FROM LABS TO LIVES: PERSONALIZING PHYSICAL CAREGIVING ROBOTS TOWARD LONG-TERM ASSISTANCE

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Rajat Kumar Jenamani

December 2026

© 2026 Rajat Kumar Jenamani

ALL RIGHTS RESERVED

FROM LABS TO LIVES: PERSONALIZING PHYSICAL CAREGIVING ROBOTS TOWARD LONG-TERM ASSISTANCE

Rajat Kumar Jenamani, Ph.D.

Cornell University 2026

Physical caregiving robots could improve the lives of millions of people who require assistance with Activities of Daily Living such as feeding, dressing, and bathing. Yet despite decades of research, deployments of autonomous physical caregiving robots in real homes have lasted only a single session or a few days. In this thesis, we propose a recipe toward the long-term deployment of physical caregiving robots, using mealtime assistance as a case study. We argue that bringing these robots into people's everyday lives requires personalizing them: fitting each robot to the person it assists. With stakeholders in the loop, and through community-based participatory research with two care recipients, we identify what a robot must personalize to. We then lengthen the deployment step by step, from a single session to a month, and find that each longer deployment surfaces a new kind of personalization.

For a single session, we personalize to functional abilities, which vary widely among people with mobility limitations. Many cannot lean forward and need direct inside-mouth food placement. This demands precision, especially for those with restricted mouth openings, and appropriately reacting to various physical interactions: incidental contacts, impulsive contacts due to sudden muscle spasms, deliberate tongue maneuvers to guide the utensil, and intentional bites. We combine multi-view mouth perception, robust to occlusion by the utensil, with physical interaction-aware control that uses vision and touch to tell these

interactions apart and react to each. Our system successfully fed 13 care recipients with diverse medical conditions at a lab, a medical center, and a home.

For a few sessions, we personalize to behavioral preferences, which shift with the situation, from watching TV or eating with a friend to being in a hurry. To feed realistic plates in the order each person prefers, we combine the commonsense reasoning of foundation models with a library of food manipulation skills. Since many preferences surface only once the robot is in use, we enable care recipients to customize it in-the-wild through natural language. We deployed our system in the homes of two care recipients, who personalized it on the fly across six meals in three settings. Both reported greater control over their meals and a stronger sense of independence than with their human caregiver.

For a month, we personalize across tasks and over time. To reuse a person's preferences across tasks, we learn them incrementally from their feedback as personalized constraints that guide the robot toward the safe and competent behaviors the person prefers. To minimize corrections as preferences change over time, we hypothesize that preferences across the different aspects of a meal are correlated and have the robot predict them jointly, so that one correction informs others. This thesis culminates in a month-long in-home deployment in which the robot assisted a care recipient, Aimee Whitneck, with 22 entire meals, from the refrigerator to the sink. We observed human–robot co-adaptation: the robot adapted to her, and she to it. Her corrections fell, resurfaced as she became more comfortable with the robot, explored new settings, and changed her preferences, and fell again as each correction came to revise more of her other preferences. She adapted as well, learning to talk without triggering a bite, and by the end looked up only to take a bite. Her workload fell and her trust rose to the top of the scale. A month after the robot left, she told us, "I miss it."

BIOGRAPHICAL SKETCH

Rajat Kumar Jenamani was born in Sarangapur, Odisha, India, and was raised in New Delhi, where he attended St. Columba's School. He received his B.Tech. in Computer Science and Engineering from the Indian Institute of Technology Kharagpur in 2021, and his undergraduate thesis won the Best Undergraduate Thesis Award in Computer Science. He then started his Ph.D. in Computer Science at Cornell University, joining Tapomayukh Bhattacharjee's newly founded EmPRISE Lab, where he worked on physical caregiving robots that personalize to the people they assist. During his Ph.D., he was a visiting student researcher at the Stanford Artificial Intelligence Lab and a research scientist intern at the Robotics and AI Institute. His work received the Best Paper Award at RSS 2025, a Best Systems Paper Honorable Mention and the Best Demo Award at HRI 2024, and the Best RoboCup Paper Award at IROS 2022, along with Best Paper finalists at ICRA 2025 and IROS 2022. He was also named an HRI Pioneer in 2025.

To my family: Mummy, Bapa, and Didi,

and to all who give and receive care.

ACKNOWLEDGEMENTS

This thesis is about care, and I have been lucky to receive so much of it.

First, I thank my advisor, Tapomayukh Bhattacharjee. Tapo, especially in my first years, I often walked into your office unsure of myself and walked out with some of your energy, excited and passionate about the work ahead. Everything we have achieved together traces back to the ambition we shared, your refusal to settle for less, and the patience and trust you gave me along the way. Thank you for reviewing my drafts, however last-minute they arrived, and for shaping me into the researcher I am today, all while insisting that being a good human comes before being a good researcher. Most of all, thank you for believing in this work, and in me, from the very beginning.

I am grateful to my committee, Sarah Dean and Malte Jung at Cornell, Maja Matarić at the University of Southern California, and Siddhartha Srinivasa at the University of Washington, for their time and feedback. Sarah, thank you for so readily joining my committee on short notice. Malte, your human-facing ideas raised a question I still ponder: do robots really provide care, or are they just tools? Maja, your work brought socially assistive robots out of the lab and into people's homes, and it inspired me to do the same for physical caregiving robots. Sidd, the summer I spent in your lab as a second-year undergraduate was my first experience in a research lab, and it set me on the path to a Ph.D. Our Friday discussions taught me to understand an idea deeply before arguing for it, and to hold strong opinions weakly.

This thesis would not exist without the care recipients, caregivers, and occupational therapists who took part in our studies, most of all Aimee Whitneck and Benjamin Howe, who joined this work as community researchers and have since become friends. Aimee, thank you for trusting us with your home for a

whole summer, even while you were away, and for all your stories from "back in the day." And for laughing with me at the irony that, on some days, one feeding robot left two people hungry. Benjamin, thank you for welcoming us into your home and for letting a robot join you at your table. Thank you both for your insights and enthusiasm through years of collaboration, I am proud to share authorship with you. Katherine Dimitropoulou, you taught me to see caregiving through the eyes of the people who give and receive it, and you brought that same care to mentoring me.

The National Science Foundation (IIS #2132846 and CAREER #2238792) and DARPA (Contract HR001120C0107) partly funded this work, and a Cornell University Ph.D. Fellowship supported my first year. I also thank the funders who supported my collaborators, as the original papers acknowledge.

Along the way, I found collaborators who made the hard days lighter, many of whom became dear friends.

Tom Silver, thank you for the rigor and clarity you brought to project after project, and for turning my messy ideas into well-posed problems. My overly ambitious plans and last-minute pushes caused you more stress than I would like to admit, yet your patience never ran out. You are someone I look up to, and if I ever did another Ph.D., I would want to do it under you. Bohan Yang, a month-long deployment in someone's home is hard in ways no paper captures. Thank you for taking it on with me and bringing the skill and perseverance of a seasoned researcher to it while still an undergraduate. I know you will go on to achieve great things. I will miss the Jay Chou on repeat and the World Cup games we streamed on the various IDs you somehow had (very legally). Rishabh Madan, we started the Ph.D. and set up the lab together, and you were always patient with my questions. I learned so much from you as we co-led SPARCS,

my first project, all the way to IROS 2022 in Kyoto, my first robotics conference. Your knowledge of controls saved us more than once, and I still have flashbacks to us tuning our MPC for what felt like forever.

Ben Dodson, thank you for your help with our multi-day deployments and demos, and for putting up with me declaring, "This seems to be a hardware problem," every time something broke. In my defense, no one I have met in robotics knows hardware better than you, and your solutions felt like magic to me. I will always be thankful that, even from Yale, you kept helping with the month-long deployment. Whenever I run into a problem now, I still hear you asking, "Is this a symptom, or is this a cause?" Daniel Stabile, thank you for pushing Feel the Bite to the finish line despite every setback, and for accepting my bribes of honey oat milk matcha lattes. We spent a whole winter in the windowless Upson 522 talking about anything and everything, and by spring we had become close friends. Rohan Banerjee, I loved working with you on *To Ask or Not To Ask*, where we always managed to eat the frog first, together.

Ziang Liu, thank you for your help with our feeding studies and interfaces, especially during the Feel the Bite user studies. You also introduced me to new hobbies, from skiing to stargazing and comet hunting amid the coyotes of the Ithaca wilderness. Eric Hu, thank you for all your help on the month-long deployment and our limb manipulation work, and for not drifting or off-roading the Jeep. It has been a joy to watch you grow into a full-stack roboticist. Pranav Thakkar, you were my go-to for rants throughout the Ph.D., usually in between our endless soccer references ("it is in the history of…"). Thank you for hearing every one of them out so gracefully, and for all the advice that came with it.

To all other members of the EmPRISE Lab, including Alexis Hao, Ruolin Ye, Zhanxin Wu, Jiaying Fang, Yunting Yan, Skyler Valdez, Shubhangi Sinha, Boxin

Xu, Thomas Patton, Sidharth Vasudev, Xiaoyu Liang, David Kim, Nikki Hart, and Xilai Dai, thank you for making my years with you so enjoyable.

David Watkins, as my manager at the Robotics and AI Institute (RAI), you trusted me with projects across the institute and put my growth ahead of your own credit. I hope to one day build teams the way you built ours. Rutav Shah, thank you for our collaboration at RAI and beyond, the infinite Mirchi Nation dinners we have shared, and the sad dog GIF from *Up* that I still use. I also thank Murphy Wozniak, Ge Zhang, Peiyi Chen, Kate Zhou, Jingxi Xu, and Ondrej Biza for the work we did together and for our daily lunches, where we caught up on everything from research to life.

Dorsa Sadigh, thank you for advising me with such energy and grace during my summer at Stanford, and for teaching me how to scope problems and choose research directions. Priya Sundaresan, sharing the lead on FLAIR with you was so much fun, from countless Zoom calls to exploring Delft at RSS 2024, where we took the wrong bus to the beach instead of the conference.

Abrar Anwar, thank you for the trust it took to let me test inside-mouth bite transfer on you first, back when it ran on position control, and for your help with Feel the Bite. You have been an amazing host whenever I have visited Los Angeles, and I look forward to finally returning the favor somewhere easier to visit than Ithaca. My friends across the HRI community, including Amal Nanavati, Ethan K. Gordon, Akhil Padmanabha, Jindan Huang, Uksang Yoo, Nathan Dennler, Vinitha Ranganeni, and Jehan Yang, made every conference a reunion, especially the workshops we organized together.

Odysseus spent ten years trying to get back to Ithaca. Leaving it after five, I finally understand why. Karan Jha, you made those years survivable, whether through soccer, five sets of tomfoolery in the gym, random adventures around

Ithaca, arguing about absurdism, or late-night talks over chamomile. With you, even a car crash became a story we still laugh about, and every day felt like a Phineas and Ferb summer. Daniel Mwesigwa, the baller, thank you for your very vocal coaching (read: shouting at me) on the soccer field, for the mushrooms and beans you have cooked for me, and for always pushing me to engage with the ethnographic side of my studies. From "quick, quick, quick" and the Afrobeats dance parties we hosted to our whirlwind trip through Europe during Euro 2024, your childlike curiosity and boundless enthusiasm remind me to never stop exploring. Jack Crowley and Jenna Shields, I will miss our "chai and pancakes" mornings and the many conversations we had over them. Mohit Gurumukhani and Farhana Shahid, thank you for the countless hangouts and the many meals we cooked together, including one very ambitious Thanksgiving dinner. Mohit, your patient attempts to explain circuit lower bounds to me were, I suspect, mostly in vain. Shipra Mishra, Gaurav Rawat, Jeswin Wilson, Saksham Kaushik, and Shivani Tuli, my *gharwale* (family), you made 123 Prospect Street feel like home in my first few years here, with all our shared dinners, festivals, and parties. Jay Swarup, James Jensen, Zeba Haq, Nidhi Baderiya, and all my friends from ERG, thank you for everything we did together, from sports days to the chai competition I hosted and game nights of Werewolf that lasted until the sun came up. Nidhi, winner of that chai competition, thank you for our many expeditions, from the bathtub races of Moravia to the chai shops of Manhattan. Through rough, uncertain days or long, punishing hours, our "meow meow" kept me sane and smiling. You light up every room you walk into.

Long before Ithaca, there was Kharagpur. There, Yash Butala and Divyansh Samdhani turned a hostel room into a home. Yash, our paths have run side by side since our first year, from surviving compilers together to graduate school,

and our conversations move easily between research and life. Thank you for all of it, and also for "postponing" your girlfriend's birthday to be at my defense. Divyansh, you can get anyone to open up within minutes, and I am grateful for the countless hours you have spent letting me do exactly that. Nikita Gautam, thank you for being one of the constants of my years at Kharagpur. Years later, we can still count on each other. Siddhant Agarwal, thank you for getting me through my degree, one course at a time. Mirambika Sikdar, we met at the Nescafé between our first-year classes, and I still look forward to every coffee in Delhi and every video call with you.

Manjunath Bhat, Snehal Reddy Koukuntla, and Shamin Aggarwal, thank you for a friendship that began when we tried out for the Kharagpur RoboSoccer Students' Group. Together, we learned ROS back when every install was painful and conda was not yet in our vocabulary, taught mini robots to play soccer, and wrote our first paper. Rahul Kumar, Parth Mall, and Kushal Kedia, we were four undergraduates figuring out research entirely on our own, and looking back, I am proud of how far we got. Rahul, thank you for all your mentoring as a senior, from research to applying abroad. In Sidd's lab at the University of Washington, Sherdil Niyaz, Sanjiban Choudhury, and Aditya Vamsikrishna Mandalika mentored me through my experiments on HERB. Sherdil, thank you especially for guiding me through my graduate school applications. The following summer, Maxim Likhachev and Muhammad Suhail Saleem welcomed me into their lab at Carnegie Mellon University. Through my final year, Partha Pratim Chakrabarti advised my B.Tech. thesis. PPC, thank you for investing so much in me as an undergraduate, always with warmth and endless patience. You believed I could take on hard problems and taught me what rigor looks like in algorithmic research. I have tried to bring a similar rigor to every chapter of this thesis.

Ankur D'Cruz, Ishaan Srivastav, and Yashvardhan Bansal, you have seen every version of me and stood by each one. Decades and continents later, every conversation with you still feels like we never left the red corridors of St. Columba's School. Ishaan, thank you for pushing the people around you to be better and for your unwavering conviction in me. And for flying in from India to attend my defense, and for making sure I finish a glass of water, whether I want to or not, as I write these acknowledgements in the Grand Tetons on our cross-country road trip. Ankur, thank you for bringing the least ego and the most fun to every room, and for listening the way few people do. Yashvardhan, thank you for showing up for your friends every time, and for being the pragmatic voice whenever I have reached out for advice.

Finally, I thank my family: my mother Jharana Jena, my father Rajendra Kumar Jenamani, and my sister Sheetal Jena. Since I was young, you have believed in me and surrounded me with unconditional love. You raised me to choose my own path, trusting me even when it took me halfway across the world. Mummy, you taught me to be kind, to pursue everything with passion, and to meet every problem believing there is always a way through. You have sacrificed so much for us, and while you joke that you are the only non-doctor in the family, the rest of us could only become doctors because of you. Bapa, your playful humor taught me never to take myself too seriously, which may explain the rest of these acknowledgements. Your curiosity shaped mine. Didi, the Doraemon to my Nobita, thank you for getting me out of every mess and for listening to all my rants. I am finally a doctor like you, though not the kind anyone should call in an emergency. Thank you all for teaching me what care means.

TABLE OF CONTENTS

Biographical Sketch . . . iii
Dedication . . . iv
Acknowledgements . . . v
Table of Contents . . . xii
List of Tables . . . xvi
List of Figures . . . xvii

1 Introduction **1**
1.1 The Need for Physical Caregiving Robots . . . 1
1.2 The Long-Term Deployment Gap . . . 2
1.3 Identifying the Problem with Stakeholders in the Loop . . . 4
1.4 Personalizing to Abilities and Preferences, Across Tasks and Over Time . . . 5
1.5 A Recipe Toward Long-Term Deployment . . . 9
1.6 Bibliographic Notes . . . 15

2 Related Work **18**
2.1 Physical Caregiving Robots in the Home . . . 18
2.2 Robot-Assisted Feeding . . . 20
2.3 Foundation Models in Robotics . . . 24
2.4 Personalization and Preference Learning . . . 26
2.5 Human–Robot Co-Adaptation . . . 29

3 SPARCS: Identifying Requirements with Stakeholders in the Loop **31**
3.1 Caregiving is Multifaceted and Contextual . . . 31
3.2 Introduction . . . 32
3.3 Building Blocks of SPARCS . . . 36
3.3.1 User Functionality Model . . . 36
3.3.2 User Behavioral Model . . . 37
3.3.3 Environment Model . . . 37
3.3.4 Robot Model . . . 38
3.4 Structured Workflows: Whats and Hows of Robot Caregiving . . . 39
3.5 SPARCS-box: Bridging the Gap between Roboticists and Caregiving Community . . . 41
3.6 Importance of User Modeling in SPARCS . . . 42
3.6.1 Leveraging M_{UF} for Bite Transfer . . . 42
3.6.2 Leveraging M_{UB} for Bite Sequencing . . . 45
3.7 SPARCS in Practice: Identifying Care Requirements and Building a Caregiving Robot . . . 47
3.7.1 Identifying Care Requirements for Six Care Recipients . . . 48
3.7.2 Building a Physical Caregiving Robot . . . 49

3.8 Adapting Robot-Assisted Feeding to Different Caregiving Scenarios . . . 52
3.8.1 User Functionality and Behavioral Models . . . 52
3.8.2 Robot Model . . . 53
3.8.3 Environment Model . . . 54
3.9 Open Challenges . . . 55
3.10 Discussion . . . 57

I Deployment for a Single Session: Personalizing to Functional Abilities 58

4 Feel the Bite: Inside-Mouth Bite Transfer 59
4.1 Introduction . . . 59
4.2 Inside-Mouth Bite Transfer System . . . 62
4.2.1 Robust Multi-View Mouth Perception . . . 63
4.2.2 Physical Interaction-Aware Control . . . 65
4.3 Evaluation of System Components . . . 67
4.3.1 Mouth Perception Evaluation . . . 67
4.3.2 Physical Interaction-Aware Control Evaluation . . . 71
4.4 Full System Evaluation with Care Recipients . . . 77
4.4.1 Methods and Procedures . . . 78
4.4.2 Participants . . . 80
4.4.3 Findings . . . 80
4.5 Discussion . . . 83

II Deployment for a Few Sessions: Personalizing to Behavioral Preferences 85

5 FLAIR: Long-Horizon Bite Acquisition 86
5.1 Introduction . . . 86
5.2 FLAIR: Feeding via Long-horizon Acquisition of Realistic dishes . 89
5.2.1 Hardware System . . . 90
5.2.2 Long-Horizon Bite Acquisition Framework . . . 91
5.2.3 Bite Sequencing via Foundation Models . . . 99
5.2.4 Integration of Acquisition and Transfer . . . 101
5.3 Experiments . . . 102
5.3.1 Bite Acquisition Experiments . . . 102
5.3.2 Comparisons with Task Planning Baselines . . . 107
5.3.3 Demonstration of Real-World Feeding . . . 110
5.4 Discussion . . . 111

6 FEAST: In-the-Wild Personalization **114**
6.1 Introduction . . . 114
6.2 Identifying Personalization Needs from Care Recipients with Diverse Medical Conditions . . . 119
6.3 FEAST: A Flexible Mealtime-Assistance System Towards In-the-Wild Personalization . . . 123
6.3.1 System Hardware . . . 124
6.3.2 System Software . . . 127
6.3.3 User Interface . . . 132
6.3.4 Default Mealtime-Assistance Procedure . . . 133
6.4 FEAST Achieves the Three Tenets of Personalization . . . 136
6.4.1 Adaptability . . . 136
6.4.2 Transparency . . . 137
6.4.3 Safety . . . 139
6.5 Five-Day In-the-Wild Evaluation . . . 140
6.5.1 Study Procedure . . . 141
6.5.2 Evaluation Metrics . . . 143
6.5.3 Results . . . 144
6.5.4 Lessons Learned . . . 147
6.6 Assessing Ecological Validity with an Occupational Therapist . . . 151
6.7 Limitations and Future Work . . . 152

III Deployment for a Month: Personalizing Across Tasks and Over Time **155**

7 Coloring Between the Lines: Personalization Across Tasks **156**
7.1 Introduction . . . 156
7.2 Constraint Satisfaction Problem Formulation . . . 162
7.3 Coloring Between the Lines . . . 165
7.4 Experiments and Results . . . 170
7.4.1 Simulation Experiments . . . 171
7.4.2 Web-Based User Study . . . 176
7.4.3 In-Home Case Study with Care Recipients . . . 177
7.4.4 Generalizing Preferences Across Tasks (exploratory) . . . 181
7.5 Discussion . . . 182
7.5.1 Revisiting Our Core Hypotheses . . . 183
7.5.2 System-Level Limitations . . . 185
7.5.3 Context Within Existing Personalization Approaches . . . 188

8 Table for Two: Co-Adaptation in a Month-Long In-Home Deployment **190**
8.1 Introduction . . . 190
8.2 System . . . 194

8.3 Personalization . . . 200
8.4 Deployment . . . 202
8.4.1 Study Procedure . . . 202
8.4.2 Reliability in the Home . . . 205
8.5 Robot's Prediction Under Non-Stationarity . . . 206
8.6 Human Side of Co-Adaptation . . . 209
8.7 Rated Workload, Experience, Acceptance . . . 213
8.8 Lessons Learned . . . 215
8.9 Discussion . . . 217

9 Discussion **218**
9.1 Scaling to Many Care Recipients . . . 220
9.2 Toward a Science of Safety . . . 224
9.3 Limitations and Open Questions . . . 226

A Feel the Bite Appendix **230**

B FLAIR Appendix **255**

C FEAST Appendix **270**

D Coloring Between the Lines Appendix **306**

E Table for Two Appendix **329**

Bibliography **376**

LIST OF TABLES

3.1 Information for the six care recipients 43
3.2 Comparison of robot models (averaged over three seeds) 54

4.1 FEMTO compared with mouth perception baselines (RMSE) . . . 68
4.2 F1 scores of physical interaction classifiers 71
4.3 SVM performance on Eval. 2 study data 77
4.4 Demographics of the care recipients in the user study 81

5.1 Comparison of FLAIR's task planner with baselines 109

6.1 Comparison with other mealtime-assistance systems 118
6.2 Per-meal system performance 144

7.1 Comparison with prior work 161

8.1 Deployments of physical caregiving robots 193

A.1 F1 scores of physical interaction classifiers in four scenarios . . . 240
A.2 SVM-Features F1 scores on the Ablation II study data 250

C.1 Personalization requirements R1–R23 270
C.2 Personalization requirements R24–R46 274

D.1 Calibration and ranking quality of dipPreferred scores 315
D.2 Planning with unsatisfiable personalized constraints in Cooking 318

E.1 The pages of the web interface 329
E.2 The pages of meal 20 in order 335
E.3 The 30 personalization dimensions and their options 336
E.4 The preparation phase, day by day 345
E.5 The six settings she ate in 349
E.6 What she learned, meal by meal 351
E.7 The researcher interventions over meals 1–13 353
E.8 The researcher interventions over meals 14–22 354
E.9 Every change to the system during the deployment 356
E.10 The preference bundle the system held at the end of every meal . 366
E.11 Every correction the care recipient made, by meal 368
E.12 The survey items 371

LIST OF FIGURES

1.1 In-home deployments of physical caregiving robots 3
1.2 From labs to lives: the recipe for deployment 10
1.3 The month-long deployment in Aimee's home 14

3.1 SPARCS translates caregiving requirements into robot guidelines 32
3.2 The five steps of using SPARCS 35
3.3 Snippet of a Structured Workflow for robot-assisted feeding . . . 41
3.4 Bite transfer success rate and neck movement by policy 43
3.5 Bite sequencing accuracy and satisfaction rating by method . . . 45
3.6 Data collection for Morgan's M_{UF} 48
3.7 Robot-assisted feeding for an individual simulating Natalia while she watches TV 51
3.8 Robot-assisted feeding for an individual simulating Jose 53
3.9 Robot-assisted feeding for an individual simulating Natalia in a social dining scenario 55

4.1 Overview of the inside-mouth bite transfer system 60
4.2 FEMTO's self-supervised multi-view training 62
4.3 The robust mouth perception pipeline 63
4.4 Ratings of real-time and one-time mouth perception across scenarios 70
4.5 Confusion matrix of the multimodal SVM on a novel participant 71
4.6 SVM performance on novel participants with finetuning 73
4.7 Results of the physical interaction-aware control study 75
4.8 Care recipients fed by our system 79
4.9 Inside-mouth bite transfer challenges of four care recipients . . . 81
4.10 Care recipients' ratings of our system, with quotes 82

5.1 Overview of FLAIR 86
5.2 The feeding utensil on three robots at two institutions 90
5.3 FLAIR's skill library 92
5.4 The six evaluation plates 102
5.5 Example run on the mashed potatoes and sausage plate 103
5.6 Participant ratings with strong user preferences 107
5.7 Food pickup and participant ratings with no user preference . . 108
5.8 Feeding a care recipient with FLAIR 112

6.1 Overview of FEAST 115
6.2 Formative study using speculative videos 120
6.3 Personalization requests from the formative study 122
6.4 FEAST's skills, custom tools, and web interface 125
6.5 FEAST's three tenets of personalization 135
6.6 Meals fed to care recipients 141

6.7 Workload, acceptance, and comparison with human caregivers . 145

7.1 Coloring Between the Lines deployed in-home with two care recipients . . . 157
7.2 Overview of Coloring Between the Lines (CBTL) . . . 163
7.3 CBTL actively reduces uncertainty and generalizes from feedback 169
7.4 Simulation experiments . . . 170
7.5 Web-based user study . . . 174
7.6 Evaluation of CBTL in real-world settings . . . 179

8.1 One meal end to end . . . 191
8.2 System overview . . . 195
8.3 The 30 personalization dimensions . . . 201
8.4 Predict, correct, re-predict . . . 202
8.5 The six settings of the deployment . . . 203
8.6 The deployment, day by day . . . 204
8.7 Human–robot co-adaptation over the 22 meals . . . 207
8.8 One correction at meal 22 revised five preferences . . . 210
8.9 Four views of how the community researcher adapted to the robot211

9.1 Two kinds of risk in a home . . . 225

A.1 Qualitative comparison of DECA and FEMTO under utensil occlusion . . . 237
A.2 Confusion matrices of SVM-Features in all four scenarios . . . 241
A.3 Confusion matrices of SVM-Features with haptic or visual input only . . . 241
A.4 Questions for the mouth perception user study (Ablation I) . . . 247
A.5 Questionnaire for the physical interaction-aware control user study (Ablation II) . . . 249
A.6 Confusion matrix of SVM-Features on the Ablation II study data 251
A.7 Questionnaire for the care recipient user study . . . 254

B.1 Prompt images used for food item recognition with a VLM . . . 256
B.2 Skill parameterizations . . . 257
B.3 Task planner decision tree . . . 260
B.4 Bite transfer via visual servoing . . . 268
B.5 Efficiency across all plates . . . 268

D.1 Ablating the LLM-derived entropy term in Books . . . 313
D.2 Reliability diagram for dipPreferred scores . . . 316
D.3 CBTL adapts to shifting user preferences in nonstationary environments . . . 320
D.4 Sankey diagrams for the three categorical questions in the web study . . . 326

E.1 The care recipient's preferences are correlated, and exploiting them lowers her correction workload 342
E.2 All 22 meals of the deployment 360
E.3 The web interface at meal 20 362
E.4 Pages that meal 20 did not reach 363

"I can't remember feeding myself. It's natural, like breathing, and you don't remember natural.

What I remember is being fed at the hospital, when I first lost the use of my arms. A care attendant mixed my pasta in with my green beans. I said I like them separate, and she rolled her eyes and went, 'Ugh.' I felt like I was bossing people around. So I learned to eat them mixed.

I used to think, 'Babies get fed.' In restaurants, I'd face the wall so people couldn't tell. When my building had a Christmas party, I didn't go. I was embarrassed.

People feed me at the rate they think I should be fed. I'm done chewing, and I wait, and I wait, and I wait. Or I'm in the middle of a story, and the fork's at my face.

For them to get it right, they'd have to sit there and watch me chew and swallow, every bite. And I don't expect anyone to do that.

What I miss most is eating at my own pace."

— Aimee Whitneck
Diagnosed with Multiple Sclerosis in 2008;
community researcher since November 2022

CHAPTER 1

INTRODUCTION

1.1 The Need for Physical Caregiving Robots

The challenges that Aimee Whitneck describes in the words that open this thesis are not hers alone. In the United States, millions of adults need help with Activities of Daily Living (ADLs) [135] such as getting in or out of bed, dressing, and eating [260]. The World Health Organization estimates that between 2% and 4% of the world's population experience significant difficulties in functioning [279].

We define *physical caregiving robots* as robots that physically assist a person with ADLs. They are the subset of physically assistive robots [197] that provides hands-on care, as opposed to robots that only fetch objects for a person or wearable robots such as prostheses and exoskeletons. Physical caregiving robots have the potential to provide this assistance, improving the quality of life of care recipients [26] and reducing the physical workload on their caregivers [18, 120].

Among ADLs, feeding is one of the most time-consuming for caregivers [39], adding to their already substantial physical workload [290] and affecting their mental health and quality of life [57, 178]. Eating is also central to identity and social interaction [196], so losing the ability to feed oneself can be devastating: people who need assistance with eating express shame about it, find it hard to ask for help, and describe it as a humiliating dependence [117, 185, 246]. Moreover, assistance with eating is highly personal, since every care recipient has their own needs. In a study of assisted feeding after Spinal Cord Injury, Martinsen et al. conclude that caregivers must tailor this assistance to each individual [186]. Aimee echoes this: what she misses most is eating at her own pace.

In this thesis, we use mealtime assistance as a case study toward the long-term deployment of physical caregiving robots in people's homes.

1.2 The Long-Term Deployment Gap

Despite the need for help with ADLs such as eating, to our knowledge, no prior work has deployed an autonomous physical caregiving robot in a person's home for more than a week. Physical caregiving robots that have been used in homes fall into three groups (Figure 1.1).

Fixed-motion robots, the first group, perceive neither the person nor the scene. Each replays a preset motion, so the person and everything around them, including the food on the plate, must stay where that motion expects them. Feeding devices of this kind date back to the 1970s [253], and some people have used them for months or years: two long-term users of Bestic had used it for 14 and 5 months [202], and one of 16 people who tested the Winsford Feeder used it at home for almost three years [144]. However, their fixed motions have limited their adoption. Of the commercial feeding devices that followed, such as the Winsford Feeder, My Spoon, Neater Eater, Bestic Arm, Meal Buddy, and Obi [199], only Obi and Neater Eater remain on the market [122, 198].

Teleoperated robots, the second group, move only under the person's direct control: the person perceives the scene and commands every motion. People have used them at home for days to years. One person used a PR2 for seven days [87], participants used a Stretch for one to two weeks at a time [211, 224, 283], and owners of wheelchair-mounted arms had used them

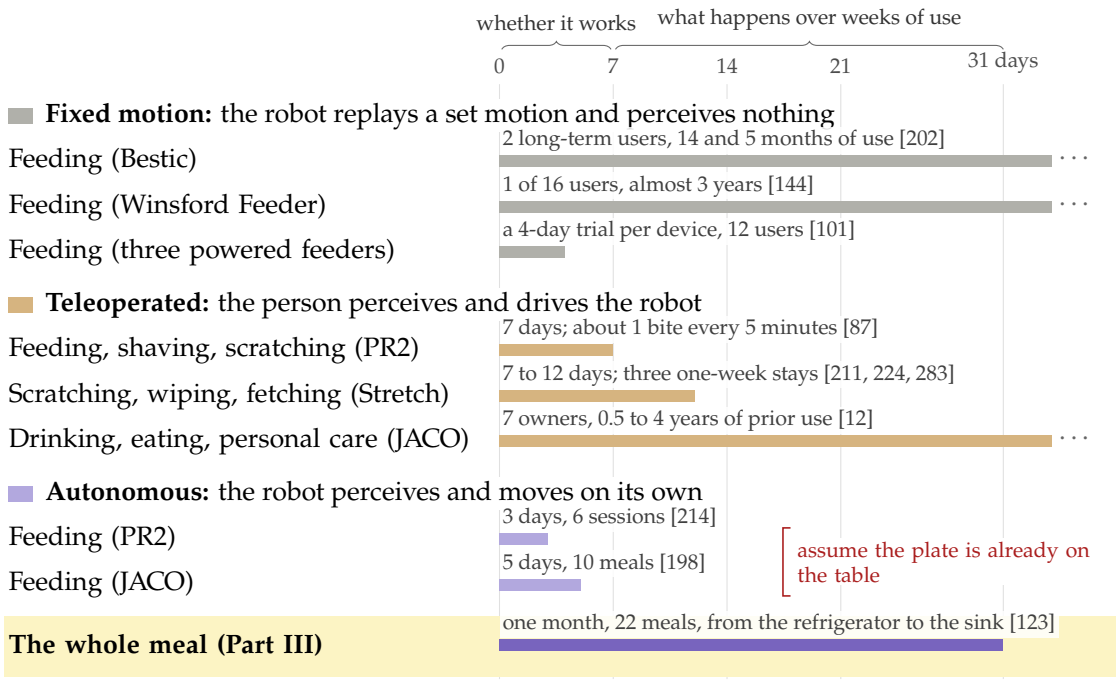


Figure 1.1: Fixed-motion and teleoperated robots have stayed in homes for months or years, whereas autonomous physical caregiving robots have stayed for days. We group deployments by how the robot moves (adapted from the survey in Jenamani et al. [123]). Each bar shows the study duration or a person's reported prior use, and we cut bars at 31 days. Both prior autonomous robots assist only with eating and begin once the meal is on the table.

for 0.5 to 4 years [12]. However, every step needs the person's attention and effort: after practice, an expert user fed himself about one bite of yogurt every 5 minutes [87].

Autonomous robots, the third group, perceive and move on their own, so they can assist without the person driving every motion and can adapt as people use them. In homes, however, they have stayed for days, not weeks: a PR2 fed a care recipient over six sessions in three days [214], and a care recipient used a JACO arm to feed himself 10 meals over five days [198]. Both robots also assist only with eating and assume that the plate is already on the table. The care

recipient still waits on a caregiver to bring and clear every meal and to help with drinking and mouth wiping.

These deployments highlight a gap: no autonomous physical caregiving robot has stayed in a person's home for weeks or helped with the whole meal. While a study of a few days shows whether a system works, it does not show what happens over weeks of use, as novelty fades [10], contexts diversify, and what the person wants from the robot changes. A social robot's first week of use is not representative of the weeks that follow [152].

1.3 Identifying the Problem with Stakeholders in the Loop

Moving toward long-term deployment requires understanding what care recipients need from a physical caregiving robot in their homes. We identify these needs with stakeholders in the loop, in two ways: (i) broadly, through studies with many care recipients, caregivers, and occupational therapists, and (ii) deeply, through a long-term collaboration with two community researchers.

Care Recipients, Caregivers, and Occupational Therapists. Across this thesis, we engage care recipients with diverse medical conditions, along with caregivers and occupational therapists. We do so through open-ended interviews and through formative studies that use speculative videos to guide in-depth conversations. In SPARCS (Structuring Physically Assistive Robotics for Caregiving with Stakeholders-in-the-loop, Chapter 3), we interview nine occupational therapists about how caregivers assist in 19 caregiving scenarios, and consult eight care recipients and two caregivers on open challenges in caregiving [180]. In the formative study of FEAST (Chapter 6), we ask 21 care recipients what they need

from a robot that feeds them [122]. In Feel the Bite (Chapter 4), 13 care recipients beyond our team evaluate a robot that places each bite inside their mouth [120].

Community Researchers. Aimee Whitneck, who has Multiple Sclerosis, and Benjamin Howe, who has a Spinal Cord Injury, are community researchers on our team. We work with them following *community-based participatory research* [89], in which community members take part in the research as members of the team rather than as its subjects. This approach, involving co-design and in-depth evaluation with one or two community researchers, is well-established in assistive technology research [69, 91, 146] and increasingly common in assistive robotics [36, 190, 196, 198, 211, 224]. The collaboration began with introductory video calls in November 2022 and has continued through regular meetings, both virtual and in person at their homes. These meetings have shaped the design of our systems and our studies. Both co-designed FEAST [122], evaluated it in their homes, and are co-authors of that work. Aimee also co-designed the month-long deployment, planned its meal schedule and settings with us, and is a co-author of Table for Two (Chapter 8) [123]. Both papers abbreviate community researcher as CR: Aimee and Benjamin are CR1 and CR2 in FEAST, and Aimee is the CR in Table for Two.

1.4 Personalizing to Abilities and Preferences, Across Tasks and Over Time

In our studies with stakeholders, such as SPARCS, in which we study how human caregivers assist care recipients [180], we find that physical caregiving is multifaceted and contextual. For example, care recipients differ in their functional abilities. One with partial neck mobility leans forward to take a bite that

their caregiver brings near their mouth, while another, with severe neck and mouth mobility limitations, needs their caregiver to place each bite inside their mouth, around their lower left molar. Their behavioral preferences differ as well. While one keeps their eyes on the television as they eat and wants to move their neck as little as possible, another turns to look at their caregiver to show that they are ready for a bite. A physical caregiving robot must address the same variability.

Functional Abilities. Robot-assisted feeding has two stages: *bite acquisition*, picking up a bite-sized food item, and *bite transfer*, moving it from the plate to the care recipient's mouth [180]. Most prior bite transfer systems bring the food just outside the mouth, and the care recipient is expected to lean forward to take the bite [13, 33, 70]. However, for many care recipients with severe mobility limitations, leaning forward is functionally impossible, and the robot has to place the food directly inside their mouth [120]. While a few earlier systems also transfer food inside the mouth, they perceive the mouth only once, so the person must stay still throughout the transfer [214, 244].

Care recipients who need food placed inside their mouth often have complex medical conditions [120]. Spinal Muscular Atrophy, for example, can reduce maximum mouth opening and decrease bite force by almost 50% [84]. Schizencephaly can cause poor jaw and tongue control. With Arthrogryposis, the bite may have to land at a specific spot in the mouth, such as above the left molar, and a care recipient may guide the utensil there with their tongue, a behavior we call *in-mouth manipulation*. Multiple Sclerosis can cause involuntary spasms that move the person suddenly and unpredictably. Consequently, a robot that places

food inside the mouth must track the mouth throughout the transfer and be precise, especially for care recipients with restricted mouth openings. It must also tell apart and react appropriately to incidental contacts, sudden spasms, tongue maneuvers that guide the utensil, and intentional bites.

Behavioral Preferences. Care recipients also differ in their behavioral preferences, such as how fast the robot should move. In Feel the Bite, one participant found the transfer "a little bit too fast" and another "too slow" [120]. In our formative study, one care recipient wanted the robot to move "as fast as it could go" [122].

Preferences also shift with the situation, starting with the plate. Meals differ from plate to plate, and people eat a wide variety of foods, which require different skills to pick up. Several prior works in bite acquisition develop individual skills targeting specific food groups, such as skewering [61, 80, 81, 83], scooping [85, 259], and twirling [256]. However, each skill is typically shown on neat plates of a single food, whereas in-the-wild meals contain several foods on one plate. A bite often cannot be picked up directly, and the food first has to be pushed aside, grouped, or cut [121]. On such a plate, people also have preferences over which bites to eat and in what order, such as "please don't feed me any meatballs" or "sausage first." Consequently, the robot must decide not only how to pick up each bite but also which bite comes next.

Preferences shift with the setting as well, even for the same care recipient. For example, in the in-home study of FEAST, Aimee set the robot to continue on its own after five seconds while she watched television. At a social meal, she raised this wait to 100 seconds and asked the robot to retract after every bite so that she could better see her companion [122]. Care recipients can also have distinct

preferences in the same setting. At social meals, conversation would trigger the standard open-mouth gesture used to initiate a bite transfer. Benjamin created his own gesture instead (a long, continuous open mouth) [122], while Aimee switched to a button [123].

Over Time and Across Tasks. Functional abilities and preferences also change over time. A participant in our formative study told us, "these things might change over time. So you might want to keep updating the robot with the progression of the disease" [122]. The deployments in this thesis do not capture such changes in functional abilities, which we return to as an open question in Section 9.3. Preferences change as people come to know the robot. For example, at the social meal in FEAST, Aimee found that even 100 seconds was too short for her to keep up with the conversation and said, "I'll get used to what the robot can do and figure out which settings actually work for me" [122]. Preferences also span tasks. In the same study, Aimee asked the robot to "bring everything as close to my mouth as possible," a preference that applies to feeding, drinking, and mouth wiping alike [122].

Mealtime assistance has no one-size-fits-all solution [122, 180]: a system must be built for all, but feel built for one. We call fitting the robot's behavior to an individual care recipient *personalization*. Fixed-motion and teleoperated robots leave the adapting to the person. An autonomous robot, however, takes over the decisions that differ from person to person and change over time, such as when to bring a bite, which bite to bring, and how fast. Because people tend to abandon assistive technology that stops fitting them [218], an autonomous robot that stays in a person's home must keep personalizing to the person it assists: to their functional abilities and behavioral preferences, across tasks, and over time.

1.5 A Recipe Toward Long-Term Deployment

Our key insight is that each longer deployment requires a further kind of personalization. Within a single session, the robot must personalize to each care recipient's functional abilities. Over a few sessions, it must personalize to behavioral preferences that shift with the situation. Over a month, it must personalize across tasks and over time. Functional abilities matter from the first bite, behavioral preferences emerge as the plate and the place vary across meals, and changes over time and across the tasks of a whole meal emerge only over weeks of use. With this insight in mind, we build toward long-term deployment through a recipe of four steps (Figure 1.2), each posing one research question. We identify the scope with stakeholders in the loop (RQ0), and then deploy for a single session (RQ1), for a few sessions (RQ2), and for a month (RQ3). After Chapter 2 reviews related work, the chapters that follow answer these questions in turn.

RQ0: What must a physical caregiving robot personalize to, and how can stakeholders shape the problem? Before any deployment, we must know what care recipients need from the robot, in a form that care recipients, caregivers, and occupational therapists can shape. In **Chapter 3**, we present SPARCS [180], a framework for physical caregiving with stakeholders in the loop. Its Building Blocks model each care recipient's functional abilities and behavioral preferences, which are what a robot must personalize to, and its Structured Workflows and SPARCS-box let stakeholders shape the problem.

RQ1 (Part I): How can a robot personalize to each care recipient's functional abilities within a single session? In a single session, the robot must adapt

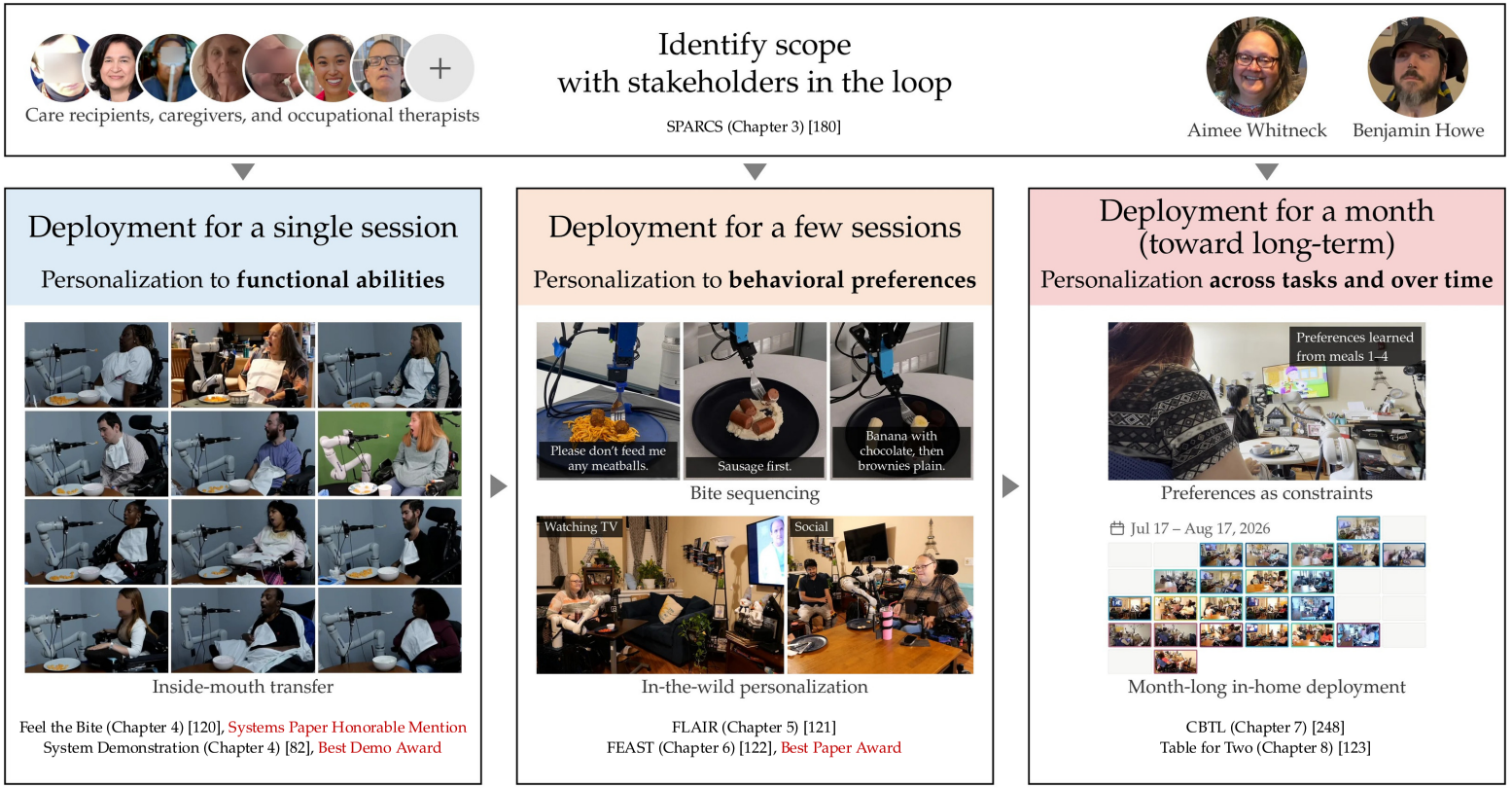


Figure 1.2: In the recipe for deployment that structures this thesis, we identify the scope with stakeholders in the loop (top), including our community researchers Aimee Whitneck and Benjamin Howe. We then deploy for a single session, personalizing to functional abilities (Part I), and for a few sessions, personalizing to behavioral preferences (Part II). Finally, we deploy for a month, personalizing across tasks and over time (Part III). The CBTL photo is from CBTL's own in-home study.

to each care recipient from the first bite. For care recipients who cannot lean forward, it must also place the bite inside the mouth. In **Chapter 4**, we present Feel the Bite [120], a robot-assisted inside-mouth bite transfer system that adapts to each care recipient's functional abilities. It combines physical interaction-aware control with multi-view mouth perception, which tracks the mouth in real time despite utensil occlusion. We demonstrate its efficacy through a user study with 13 care recipients with diverse medical conditions, all necessitating assistance in feeding, across three sites: our lab, a medical center, and a care recipient's home. Participants perceive the system as safe and comfortable, and view the technology favorably. Feel the Bite received a Best Systems Paper Honorable Mention at HRI 2024. The chapter closes with our HRI 2024 demonstration of the full feeding system, which combined our bite transfer with bite acquisition from the Personal Robotics Lab at the University of Washington [82] and received the Best Demo Award.

RQ2 (Part II): How can a robot personalize to behavioral preferences that shift with the situation, over a few sessions? Over a few sessions, people eat different meals in diverse settings, so the robot must let them state or edit what they want, in natural language or through an interface. In **Chapter 5**, we present FLAIR (Feeding via Long-horizon AcquIsition of Realistic dishes) [121], a system for long-horizon bite acquisition from in-the-wild plates that uses foundation models to sequence a library of seven food manipulation skills. It balances a preference that the care recipient states at the start of the meal against efficiency, and thus personalizes within a single meal. We deploy FLAIR across two institutions and three robots. In a user study with 42 participants without mobility limitations, we demonstrate the necessity of balancing preferences and efficiency for feeding complete, realistic meals, as compared to an efficiency-only

or a preference-only approach. Finally, it feeds a care recipient with Multiple Sclerosis a meal of various fruits and dips.

In **Chapter 6**, we present FEAST [122], a mealtime-assistance system that builds on FLAIR's bite acquisition and that lets care recipients personalize it themselves, safely and transparently, in-the-wild. It addresses 36 of the 46 types of requests from our formative study. In a five-day in-home evaluation, our two community researchers feed themselves six meals across three contexts (personal, watching TV, and social). They complete the meals with few researcher interventions and personalize the system on the fly, with low cognitive workload. Both report that FEAST gives them greater control over their meals and a stronger sense of independence than their human caregiver. FEAST received the Best Paper Award at RSS 2025.

RQ3 (Part III): How can a robot personalize across tasks and over time, when it assists a care recipient with the whole meal for a month? Over a month, the robot assists with the whole meal, in settings beyond those planned at the start, while the person's preferences and the way they work with the robot change. To keep up, the robot learns from the person's feedback to predict what they will want next, while the person keeps correcting its predictions. In **Chapter 7**, we present Coloring Between the Lines (CBTL) [248], a framework that learns personalized constraints incrementally from a care recipient's feedback and plans only within the robot's safety and competency constraints. These personalized constraints are a step toward carrying what the robot learns in one task to others. Across three simulation tasks, CBTL personalizes more effectively and efficiently than four baselines, and in a web-based study, 60 participants significantly prefer its choices over a non-personalized baseline. In an in-home case

study over five meals (the first four screen-based), separate from the month-long deployment, two care recipients prefer CBTL because it reduces the workload of specifying preferences from scratch for each new meal.

In **Chapter 8**, we present Table for Two [123], a month-long in-home deployment of an autonomous mobile manipulator for the whole meal, which addresses RQ3 in Aimee's home. Sections 8.2 to 8.4 describe the system and the deployment, including a personalization framework in which a large language model predicts all 30 personalization dimensions jointly before each meal and re-predicts the open ones after each correction. In Sections 8.5 to 8.8, we give an account of how Aimee and the robot adapted to each other over the month and draw four lessons for long-term deployment. We also compare this joint prediction with predicting each dimension on its own, as CBTL does: in a replay of the logged meals, joint prediction needs 27% fewer corrections (57 against 78).

A Month of Meals in Aimee's Home. Over 22 meals, from July 17 to August 17, 2026 (Figure 1.3), the robot took Aimee's meal from her refrigerator, heated it in her microwave, and brought it to her table. It then assisted her with feeding, drinking, and mouth wiping, and took the plate to her sink. Of the 958 skill executions that depended on perception, 93.2% succeeded without her input and 97.4% with her corrections or teleoperation. However, a researcher was present at every meal and intervened 32 times, when a skill could not recover or safety required it. She rated the robot above her human caregiver on independence and control.

Across the month, we observed *human–robot co-adaptation*, in which the robot adapted to Aimee and she adapted to it. Her corrections fell over the first six

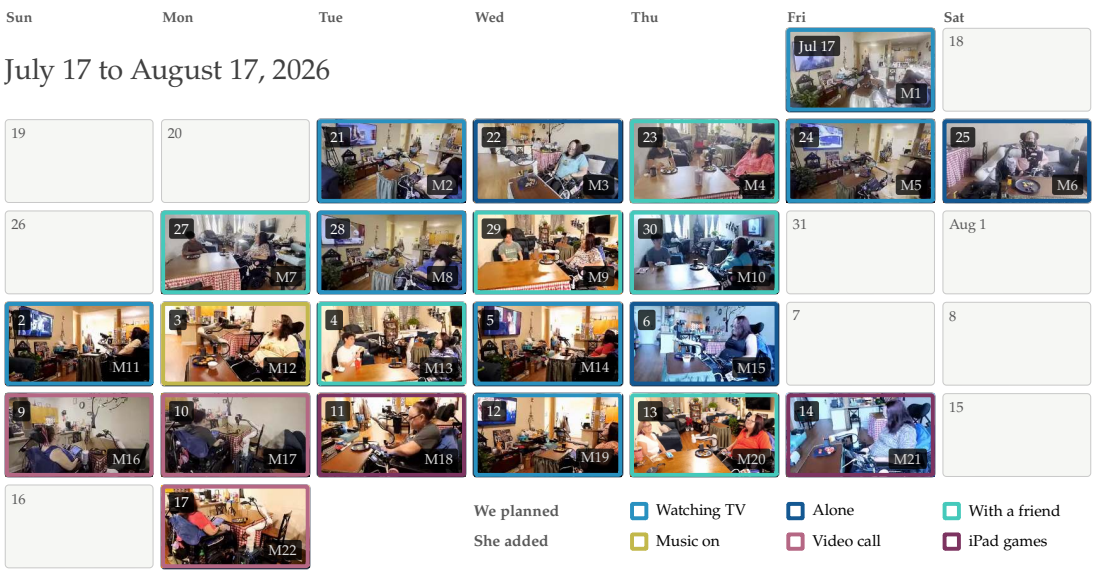


Figure 1.3: Over the month-long deployment, from July 17 to August 17, 2026, the robot assisted Aimee with 22 meals in her home (M1 to M22), one per weekday. A weekend meal (M6, M11, M16) made up any weekday she was unavailable. Each tile shows a frame from its meal, and its border marks the setting. We planned three settings with her, and she added three more as she grew comfortable with the robot.

meals, plateaued, rose again as she chose new settings and revised her preferences, and fell again. A study of 10 meals, the length of the in-home study of Nanavati et al. [198], would have ended inside this plateau and missed the second rise.

Chapter 9, Discussion, summarizes what this thesis enables, discusses scaling to many care recipients and a science of safety, and closes with the limitations of this work and the open questions they leave.

The hardware and software of our systems are open source.[1] The appendices hold the supplementary material of Chapters 4 to 8.

[1] `https://github.com/empriselab/feeding-deployment`

1.6 Bibliographic Notes

Chapters 3 to 8 and the appendices are based on published and submitted papers, written with the co-authors listed below. In each author list, * denotes equal contribution and † denotes equal advising.

Chapter 3 is based on the following work:

SPARCS: Structuring Physically Assistive Robotics for Caregiving with Stakeholders-in-the-loop, Rishabh Madan*, Rajat Kumar Jenamani*, Vy Thuy Nguyen, Ahmed Moustafa, Xuefeng Hu, Katherine Dimitropoulou, Tapomayukh Bhattacharjee, IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2022 [180].

Chapter 4 is based on the following work, and Appendix A on its supplementary material:

Feel the Bite: Robot-Assisted Inside-Mouth Bite Transfer using Robust Mouth Perception and Physical Interaction-Aware Control, Rajat Kumar Jenamani, Daniel Stabile, Ziang Liu, Abrar Anwar, Katherine Dimitropoulou, Tapomayukh Bhattacharjee, Proceedings of the ACM/IEEE International Conference on Human-Robot Interaction (HRI), 2024 [120].

The closing paragraph of Chapter 4 refers to the following demonstration:

An Adaptable, Safe, and Portable Robot-Assisted Feeding System, Ethan K. Gordon*, Rajat Kumar Jenamani*, Amal Nanavati*, Ziang Liu, Daniel Stabile, Xilai Dai, Tapomayukh Bhattacharjee†, Tyler Schrenk, Jonathan Ko, Haya Bolotski, Raida Karim, Atharva Kashyap, Bernie Hao Zhu, Taylor

Kessler Faulkner, Siddhartha S. Srinivasa†, Companion of the ACM/IEEE International Conference on Human-Robot Interaction (HRI), 2024 [82].

Chapter 5 is based on the following work, and Appendix B on its supplementary material:

FLAIR: Feeding via Long-horizon AcquIsition of Realistic dishes, Rajat Kumar Jenamani*, Priya Sundaresan*, Maram Sakr, Tapomayukh Bhattacharjee†, Dorsa Sadigh†, Robotics: Science and Systems (RSS), 2024 [121].

Chapter 6 is based on the following work, and Appendix C on its supplementary material:

FEAST: A Flexible Mealtime-Assistance System Towards In-the-Wild Personalization, Rajat Kumar Jenamani, Tom Silver, Ben Dodson, Shiqin Tong, Anthony Song, Yuting Yang, Ziang Liu, Benjamin Howe, Aimee Whitneck, Tapomayukh Bhattacharjee, Robotics: Science and Systems (RSS), 2025 [122].

Chapter 7 is based on the following work, and Appendix D on its supplementary material:

Coloring Between the Lines: Personalization in the Null Space of Planning Constraints, Tom Silver*, Rajat Kumar Jenamani*, Ziang Liu, Eric Hu, Ben Dodson, Yunting Yan, Tapomayukh Bhattacharjee, under review at ACM Transactions on Human-Robot Interaction (THRI). An earlier version appeared as an arXiv preprint in 2025 [248].

Chapter 8 is based on the following work, and Appendix E on its supplementary material:

Human–Robot Co-Adaptation in Month-Long In-Home Mealtime Assistance, Rajat Kumar Jenamani*, Bohan Yang*, Aimee Whitneck, Ben Dodson, Eric Hu, Yunting Yan, Tom Silver, Tapomayukh Bhattacharjee, under review at the ACM/IEEE International Conference on Human-Robot Interaction (HRI 2027) [123].

Chapter 2 draws on the related-work sections of these papers.

Chapter 9 is based on all of the above works.

CHAPTER 2

RELATED WORK

This chapter reviews the work on which this thesis builds, in the order of the recipe for deployment (Figure 1.2). In-home deployments of physical caregiving robots (Section 2.1) frame the gap that this thesis addresses. Robot-assisted feeding (Section 2.2) underlies Parts I and II, while foundation models in robotics (Section 2.3) and personalization and preference learning (Section 2.4) underlie Parts II and III. Finally, human–robot co-adaptation (Section 2.5) frames Part III.

2.1 Physical Caregiving Robots in the Home

Physical caregiving robots, robots that physically assist people with Activities of Daily Living (ADLs), are the subset of physically assistive robots [197] that provides hands-on care. Many in-home deployments of these robots assist with feeding, the ADL on which this thesis focuses. Building on three decades of research in mealtime assistance [13, 70, 80, 115, 120, 244, 251, 262], recent deployments have fed users in their homes over multiple days: six sessions over three days [214], six meals over five days [122], and ten meals over five days [198]. These systems assist at the table, assuming a caregiver places the meal on the table, and clears it after, whereas Table for Two (Chapter 8) assists with a meal end-to-end. Moreover, to our knowledge, no prior work has deployed an autonomous physical caregiving robot in a person's home for more than a week. Hobbit, an autonomous robot for older adults, stayed in each home for about three weeks, but it does not assist with ADLs [10]. The robots that stayed longer were teleoperated or ran a fixed motion: seven days to three one-week stays of a PR2 or Stretch commanded through a web interface, head motion, or muscle

signals [87, 211, 224, 283], episodic use of a PR2 across a year [37], wheelchair-mounted arms their owners have driven for years [12], and commercial feeders people have used for months [101, 202]. In these deployments the robot changed only when researchers added capabilities or reconfigured the interface [211, 224, 283], so only the person could adapt. We deploy an autonomous physical caregiving robot in a person's home for a month and study the co-adaptation between the robot and the person it assists (Table 8.1).

Long-Term Deployments of Other Robots. A robot's first week of use is not representative of the ones that follow [77, 130, 152], and Bajones et al. [10] argue that a home robot can only be benchmarked over three weeks. In such deployments the robot changes for the site or on a schedule rather than for the individual: STRANDS learned which of its site's routes were reliable [95], Robovie unlocked behaviors as interaction time accumulated [131], and Hobbit switched modes on a predetermined day [10]. Adaptation to a specific person over a month at home has been shown only by socially assistive robots, which set instructional difficulty from a child's performance [238] or learn a per-child policy online [46]. Clabaugh et al. call this personalization co-adaptive, but frame the learning from the robot's side alone, and both studies measure the child only through task skills, which improved, and engagement, which stayed flat; whether the person adapts to the robot is not asked.

2.2 Robot-Assisted Feeding

Research on mealtime-assistance systems dates back to the 1970s, with early examples such as the Morewood Spoon Lifter [253]. To move beyond fixed motions, recent research focuses on using sensors, such as in-hand cameras, to perceive the environment, and plan and execute autonomous motions. We review this work in the order of this thesis: bite transfer (Part I), bite acquisition (Part II), the other mealtime tasks, and complete mealtime-assistance systems.

Bite Transfer. Various works propose autonomous bite transfer methods [13, 33, 70, 120, 214, 244], which can be broadly categorized into two types. Outside-mouth bite transfer methods [13, 33, 70] bring food close to the care recipient's mouth, requiring them to lean forward to take a bite. For individuals with severe mobility limitations who cannot lean forward, inside-mouth bite transfer methods [120, 214, 244] place food directly inside their mouth. Inside-mouth transfer requires the robot to perceive the mouth throughout the transfer and to react to its physical interactions with the care recipient.

Mouth Perception. To effectively feed care recipients with complex medical conditions, it is necessary to accurately perceive their mouth for the whole duration of inside-mouth bite transfer. Contemporary robot-assisted feeding systems use single in-hand cameras, and localize the mouth by either projecting 2D mouth landmarks onto an aligned depth image [244], or fitting a fixed-head model to sparse 3D facial landmarks [70]. However, near the mouth, significant

occlusion from the utensil leads to noisy depth data and inaccuracies in single-view perception of 2D landmarks, rendering these methods ineffective for continuous mouth tracking. In Feel the Bite [120] (Chapter 4), we propose a mouth perception pipeline that leverages multiple in-hand cameras and parameterized head models to be robust to tool occlusion, thus enabling uninterrupted and accurate mouth perception. Among parameterized head models [20, 52, 159], Faces Learned with an Articulated Model and Expressions (FLAME) [159] is of special interest as it separates the representation of identity, pose, and facial expression. Detailed Expression Capture and Animation (DECA) [62] is state-of-the-art for FLAME parameter estimation, and takes a single RGB image as input. However, this reliance on monocular RGB images makes DECA vulnerable to inaccuracies in our use case where significant occlusions around the mouth are present. To overcome this limitation, we propose a novel encoder based on DECA's architecture that integrates data from multiple cameras, ensuring robust prediction of FLAME parameters.

Physical Interaction-Aware Control. Prior work in contact classification distinguishes between incidental collisions and intentional task contacts during collaborative human-robot manipulation [45, 68, 79, 143, 164]. Classical approaches [143] use the spectral norm of external force/torque signals in a specific frequency range for classification. In contrast, several other methods leverage learning-based approaches like Support Vector Machines with time-series features derived from physical contact models [45, 68, 79], or time-series models such as RNNs [164]. While these methods typically use haptic data, in Feel the Bite, we are inspired by works that integrate multiple sensing modalities [60, 150, 160, 255]. To the best of our knowledge, Feel the Bite is the first

to explore multimodal (visual + haptic) sensing for physical human-robot interaction classification. It is also the first to categorize physical interaction types beyond incidental and intentional, by additionally identifying impulsive interactions and further distinguishing intentional interactions into in-mouth manipulation and bite interactions.

Bite Acquisition. Prior work in bite acquisition has focused on developing individual food manipulation skills for specific food types. Various works [61, 80, 81, 255] tackle acquisition of solid bite-sized foods, and demonstrate effective skewering strategies based on the food item's pose and material properties. Sundaresan et al. [256] propose visually parameterized primitives for twirling and grouping noodle-like dishes, and show generalization to unseen noodles. Beyond fork-based manipulation, Grannen et al. [85] plan bimanual scooping actions with two custom utensils, while Tai et al. [259] and Zhang et al. [294] develop specialized strategies for scooping with a spoon and cutting with a knife, respectively. However, no prior work in robot-assisted feeding considers complete, in-the-wild meals containing various food types (noodles, semisolids, sauces, cuttable food items, etc.) within the same plate, as typically encountered in everyday scenarios.

In FLAIR [121] (Chapter 5), we leverage insights from the aforementioned state-of-the-art food manipulation works to develop a large library of bite acquisition skills, and use foundation models to sequence these skills for efficiently feeding realistic dishes while obeying user preferences. To the best of our knowledge, FLAIR is the first of any autonomous feeding system to tackle in-the-wild meals containing various food types, and incorporate bite sequencing preferences for long-horizon feeding. Various works have shown joint bite acquisition

with transfer [18, 70, 120]. However, they typically consider bite acquisition actions over a single timestep and not over the complete meal. In contrast, we illustrate that our long-horizon bite acquisition framework can seamlessly integrate with existing methods for bite transfer [70, 120], and demonstrate feeding of a full meal to a care recipient. Recent work, published after FLAIR, pre-acquires soft food items before picking them up [88] and sequences acquisition actions to clear a bowl of liquid, semisolid, or deformable food [16].

Other Essential Mealtime Tasks. Several works have explored assistance with drinking [78, 239], focusing on grasping a cup and having users drink from the rim. In FEAST [122] (Chapter 6), we instead focus on drinking from a cup with a straw, aligning with our community researchers' routines. Unlike prior methods, which assume specific cup colors [78] or pre-grasped cups [239], FEAST emphasizes real-world flexibility, using an adaptable handle that fits various cups without strict shape or color constraints. Some works also propose methods for mouth wiping [96, 109]. Similar to these, FEAST can pick up a custom wiping tool and position it near the user's mouth, ready for wiping.

Mealtime-Assistance Systems. In FEAST, we build on FLAIR [121] for bite acquisition and Gallenberger et al. [70] and Feel the Bite [120] for bite transfer, enabling our system to feed realistic dishes to users with severe mobility limitations. However, recent studies show that feeding robots often exhibit obtrusive motion [13, 196], limiting their use in social settings, and have restricted workspace [198], constraining plate and user placement. These limitations hinder personalization, as robots must adapt to social contexts with non-obtrusive motion and allow flexible user and plate positioning. Towards addressing these

challenges, and based on feedback from community researchers, we adapt these approaches to a novel feeding utensil (Section 6.3.1) that improves the robot's workspace and decreases obstruction.

Most relevant to our work are autonomous systems that demonstrate integration of various components for feeding a complete meal [70, 121, 198, 214] (see Table 6.1). Unlike Park et al. [214], which focuses on feeding yogurt, and Gallenberger et al. [70] and Nanavati et al. [198], which limit bite acquisition to skewering, we employ a range of skills (skewering, scooping, twirling, grouping, and more) to pick up diverse food items and sequence them over a long horizon for feeding an entire meal. We extend FLAIR [121], which integrates with Feel the Bite [120], with a novel feeding utensil and a web interface for user interaction. More significantly, FEAST is the first to integrate feeding with other essential mealtime tasks, including drinking and mouth wiping, while also automating user personalization. No prior system has explored feeding in diverse in-the-wild scenarios, except Nanavati et al. [198], which offers customization of limited system parameters.

2.3 Foundation Models in Robotics

Foundation Models for Robotic Manipulation. Two of the most challenging aspects associated with feeding are planning over available skills, and developing a library of food manipulation skills themselves. To this end, several recent works in robotic manipulation use foundation models such as vision-language models (VLMs) [1, 133, 158, 223] or large language models (LLMs) [27, 41, 55,

221, 222] towards both high-level task planning and skill instantiation. A standard approach is to prompt foundation models with context including available skills, object states, etc., and to use them to plan action sequences either for long-horizon manipulation [3, 38, 104, 163, 212, 249, 280] or grounded exploration [125]. However, these works focus on exploiting the commonsense reasoning capabilities of these models [71, 147, 234], such as inferring a sequence of skills that is feasible or user-preferred based on the provided visual and semantic context. In FLAIR (Chapter 5), we additionally care about planning *efficient* skill sequences in the setting of feeding. Reasoning about efficiency and skill affordances in a few-shot manner remains brittle and challenging for these models, due to hallucinations [129] and a lack of priors about embodied agents. Instead, we propose inferring the efficiencies of skills separately, and providing this as additional context to aid in planning.

Besides skill sequencing, several recent works show the benefits of using foundation models towards inferring the parameters of low-level skills themselves, rather than data-driven approaches to learning skill policies from scratch [6, 23, 24, 139, 179, 210, 289]. Recent approaches include instantiating skills via code skeletons generated by LLMs [105, 161], or implementing skills parameterized by open-vocabulary object detectors [280, 292] or keypoint affordances from VLMs [166]. These approaches have mainly been applied to simple quasi-static actions such as pick and place. In FLAIR, we instead apply this paradigm towards estimating the visual state of food items, and using this to parameterize a diverse library of skills such as twirling, scooping, and cutting.

Large Language Models for Human-Robot Interaction. Recent advancements in large language models (LLMs) [1, 41] enable the use of language as an interface

for human-robot interaction. Some approaches utilize LLMs for high-level planning, allowing systems to generate plans over a fixed library of skills based on user language commands [3, 38, 104, 129, 163, 212, 249]. FEAST (Chapter 6) represents a recent category of approaches that enable users to edit both high-level plans and parameters of low-level skills using LLMs [14, 86, 245]. We use an LLM to translate a user's request, formulated in natural language, into structured updates to the robot's behavior trees, which the system then statically validates for safety. However, given the physical nature of human-robot interactions in mealtime assistance, we restrict edits to carefully selected operations to ensure safety. In Part III, we also use LLMs in CBTL (Chapter 7) and Table for Two (Chapter 8), to predict a care recipient's preferences from their feedback (Section 2.4).

2.4 Personalization and Preference Learning

Various formative studies in assistive robotics emphasize the importance of user personalization based on factors such as mobility limitations, behavior, and context [17, 32, 196, 197]. The inclusion of user preferences in the design and operation of assistive robots is essential for significantly enhancing user satisfaction [32]. We review approaches in the order of this thesis: preferences that the person states (Part II), preferences that the robot learns from the person (Part III), and the constraint-based planning on which CBTL builds (Chapter 7).

Preferences Stated by the Person. Canal et al. [31] explore task planning adhering to user preferences for an assistive shoe dressing experiment. However, they explore user specification only in the form of post-hoc scoring of executed actions which is restrictive for various safety-critical applications. Nanavati

et al. [198] provide predefined customizations for a mealtime-assistance robot through a web interface, while in FLAIR [121] (Chapter 5), we explore bite-ordering preferences stated in natural language. Recently, TidyBot [280] showcased that LLMs can summarize information from limited examples and extrapolate general user preferences for determining the proper place to put each object while tidying a room. However, their approach to task planning lacks consideration of additional metrics, such as efficiency, which is crucial in the context of feeding a complete meal. In FLAIR, we instead factor in both user preference and acquisition efficiency for long-horizon feeding. In FEAST (Chapter 6), we use FLAIR for bite-ordering preferences and significantly extend personalization to other system components by enabling open-ended customization through language inputs.

Previous work considers one-time personalization in the context of single tasks [13, 21, 29, 31, 72, 235, 276, 293]. For example, given natural-language user preferences such as "sort the laundry by lights and darks" [280], robot behavior can be adapted on-the-fly to take these preferences into account. Some of these works explicitly consider safety [132, 193, 261]. INQUIRE [65] introduces continual personalization by actively selecting among different types of user feedback, but this adaptation remains task-bounded. As in these approaches, the person in FLAIR and FEAST states preferences at the start of each task. In Part III, the robot learns them instead.

Preferences Learned from the Person. One approach to achieving personalization is through implicit adaptation, where limited test-time examples guide the system to a specific preference [98, 173, 280, 281]. In FEAST, we align with

this by generating personalized gesture detectors using a small number of examples from the user. Unlike prior black-box approaches to gesture personalization [22, 127], we take a more transparent approach by composing functions from our perception library to generate customized detectors. While some studies evaluate gesture detection for care recipients [116, 258], FEAST is the first to consider personalized gesture detection for those with mobility limitations.

Closer to our goal in CBTL are approaches that learn user models reusable across multiple contexts [98, 99, 173, 272]. These works primarily focus on the structure of the user model itself, without mechanisms for active or continual adaptation. Relatedly, reward-based formulations of personalization infer user preferences from feedback and use them to guide robot behavior across tasks [19, 266]. While effective within a single latent reward function, these approaches do not capture or generalize across the diverse categories of preferences that arise in assistive settings [122]. A complementary line of research investigates lifelong personalization in human–robot interaction [43, 151, 155, 252]. While these studies highlight that ongoing personalization is critical for real-world adoption, they do not offer algorithmic frameworks that enable robots to incrementally adapt their behavior online as we propose. In CBTL and Table for Two (Chapters 7 and 8), the robot learns a care recipient's preferences from their feedback across tasks and over time.

Personalization within Planning Constraints. Our key insight in CBTL is that planning with constraints induces a "null space" that enables safe personalization. The works of Thumm et al. [261] and Wang et al. [269] are especially relevant; like us in CBTL, they consider personalization within constraint-based systems and focus respectively on safety and active learning. Prior

work has considered learning in null space, typically in the more literal case where the null space refers to the redundant degrees of freedom of a manipulator [58, 204, 233, 263, 284]. There is also work in safe reinforcement learning that generalizes the concept of null spaces to constraint manifolds and focuses on learning new low-level policies [168, 169]. These works do not consider personalization and typically focus on single tasks. CBTL's constraint generation framework takes inspiration from task and motion planning (TAMP) [74]. Learning constraints for TAMP has been previously considered [51, 106, 163, 176, 189]. Learning predicates for TAMP is also closely related [7, 157, 162, 247]. Especially relevant are the works of Wang et al. [274] and Li and Silver [156], who consider *active* constraint learning. Our use of entropy-based active learning is inspired by the latter work. These works do not consider personalization. Other recent work in TAMP has considered active learning for samplers instead of constraints [145, 188]. In principle, these efforts could be combined with our approach to develop a robot that not only personalizes, but also accelerates its planning over time.

2.5 Human–Robot Co-Adaptation

Co-adaptation names adaptation in both directions, the person's to the robot and the robot's to the person [203, 213, 265]. The person's side alone is well documented: households reorganized routines around a vacuuming robot that neither plans nor learns [63, 67, 257], and people's shared-control commands grew more aligned with a robot policy that never changed [4]. Where both sides change, the studies with a real robot last a single session [243] or 30 trials per

condition [110]; others retrain a brain–machine decoder on a user's shifting signals [217]. In Table for Two (Chapter 8), we build on CBTL (Chapter 7), which predicts each preference parameter separately with an LLM [248], and on FEAST (Chapter 6), which carries a user's last preferences forward by setting [122]. Both serve as baselines in Section 8.5. We predict all dimensions jointly, so that one correction revises the others, and we evaluate the predictions over a month of in-home use, as a user's preferences change.

CHAPTER 3

SPARCS: IDENTIFYING REQUIREMENTS WITH STAKEHOLDERS IN THE LOOP

3.1 Caregiving is Multifaceted and Contextual

Morgan (he/him) is a 58-year-old who sustained a brainstem stroke at 40 that made him quadriplegic and mute. He receives assistance from his partner and primary caregiver Riley (she/her), and his professional caregiver Spencer (they/them)[1]. Morgan's caregiving is complex and personalized. It is extremely challenging for any arbitrary person without proper training to assist him. His assistance is contextual on his functional abilities and behavior, his caregiver, and his environment:

1. Care Recipient Context: When being assisted with feeding, Morgan turns to his caregiver to show his intent to have a bite. The mobility limitations in his tongue necessitate solid food to be placed around his lower left molar for chewing.

2. Caregiver Context: Morgan is tall. Dressing him involves lifting his limbs. While Spencer is tall and muscular and does not face difficulties with this task, Riley performs it differently because she is short and lean.

3. Environment Context: Morgan lives in a house with a small bathroom. Bathing Morgan is especially challenging due to his occasional spasms. These spasms lead to involuntary movements, which can hurt him if his limbs hit the bathroom walls. This requires his caregivers to be extremely vigilant.

[1]These are real people whom we interviewed as part of a user study. We use fictitious names/pronouns for this chapter.



Figure 3.1: SPARCS enables roboticists to translate human caregiving requirements to guidelines for physical robot caregiving.

Many of Morgan's activities of daily living (ADLs) easily take more than an hour and pose a challenge to all other family routines. Caregiving for Morgan is very taxing for Riley and leaves her with barely any time for her own needs. Morgan is aware of the situation and wishes to do some of these activities by himself, not only to feel more independent but also to reduce Riley's burden.

3.2 Introduction

Morgan is among many others worldwide who require assistance with ADLs. However, existing work in physical robot caregiving is limited in its ability to consistently provide in-situ assistance to care recipients for the entire duration of an ADL. This is due to a myriad of factors such as diversity of tasks, the need for personalized assistance, and limited access to other stakeholders (care recipients, their caregivers, and occupational therapists) who have the required domain knowledge. There exists variability in problem definitions, which points to an inherent lack of common grounding in problem setup. Without a common underlying framework to systematically reason about physical robot caregiving, it is very challenging to transfer insights from one specific scenario to another.

We propose **S**tructuring **P**hysically **A**ssistive **R**obotics for **C**aregiving with **S**takeholders-in-the-loop, or **SPARCS** to address these challenges (Figure 3.1). This framework comprises **Building Blocks** (Sec. 3.3), **Structured Workflows** (Sec. 3.4), and **SPARCS-box** (Sec. 3.5). Inspired by the different contexts in human caregiving, we define four *Building Blocks*, the *user*, i.e., the care recipient, their *human caregiver*, their *environment*, and the *robot*. We build upon these blocks to form *Structured Workflows*, hierarchical workflows that enable us to answer the *whats* and *hows* of physical robot caregiving. Building these workflows requires communication between roboticists and other stakeholders who have expertise in caregiving. *SPARCS-box* is a web-based platform that facilitates this dialogue.

SPARCS is a framework for building physical caregiving robots. We perform experiments that highlight the importance of *Building Blocks*, particularly user models, to build effective and personalized systems. We show SPARCS in practice by using it to (i) identify care requirements, and (ii) utilize these requirements to build a physical caregiving robot. We consider six care recipients with varying disabilities and collect data on the *Building Blocks* for ADLs that they require assistance with. We perform a user study with expert occupational therapists to build the *whats* of *Structured Workflows* for all the identified caregiving scenarios. This can help us translate the caregiving needs of the six care recipients into well-defined research problems for roboticists. All the data and workflows are publicly available on *SPARCS-box* [182] and can be used by roboticists to reason about the *hows* for robot caregiving. We illustrate the process of generating physical robot caregiving solutions for one of the scenarios – the activity of robot-assisted feeding for one of the six care recipients. Additionally, we demonstrate how this feeding system adapts to changes in the different building

blocks. Finally, we conduct a study with care recipients and caregivers, asking them questions on general ADL assistance. We use SPARCS to examine their caregiving needs and share open challenges for the robotics community.

We summarize our contributions as follows:

- We propose SPARCS, a framework for physical robot caregiving. SPARCS comprises (i) *Building Blocks*, models that define physical robot caregiving scenarios, (ii) *Structured Workflows*, hierarchical workflows that enable us to answer the *whats* and *hows* of physical robot caregiving, and (iii) *SPARCS-box*, a web-based platform that facilitates communication between all stakeholders.
- We highlight the importance of *Building Blocks*, particularly user models, to build effective and personalized physical caregiving robots.
- We collect real-world data and create *Building Blocks* for six care recipients with varying disabilities. We conduct a user study with occupational therapists and use *Structured Workflows* to identify care requirements for these care recipients.
- We demonstrate the use of SPARCS for building a robot-assisted feeding system for one of the six care recipients. Additionally, we exhibit the adaptive nature of this system through reuse and transfer of its *Structured Workflows* across different *Building Blocks*.
- We identify open challenges in physical robot caregiving for roboticists through another user study with care recipients and caregivers.

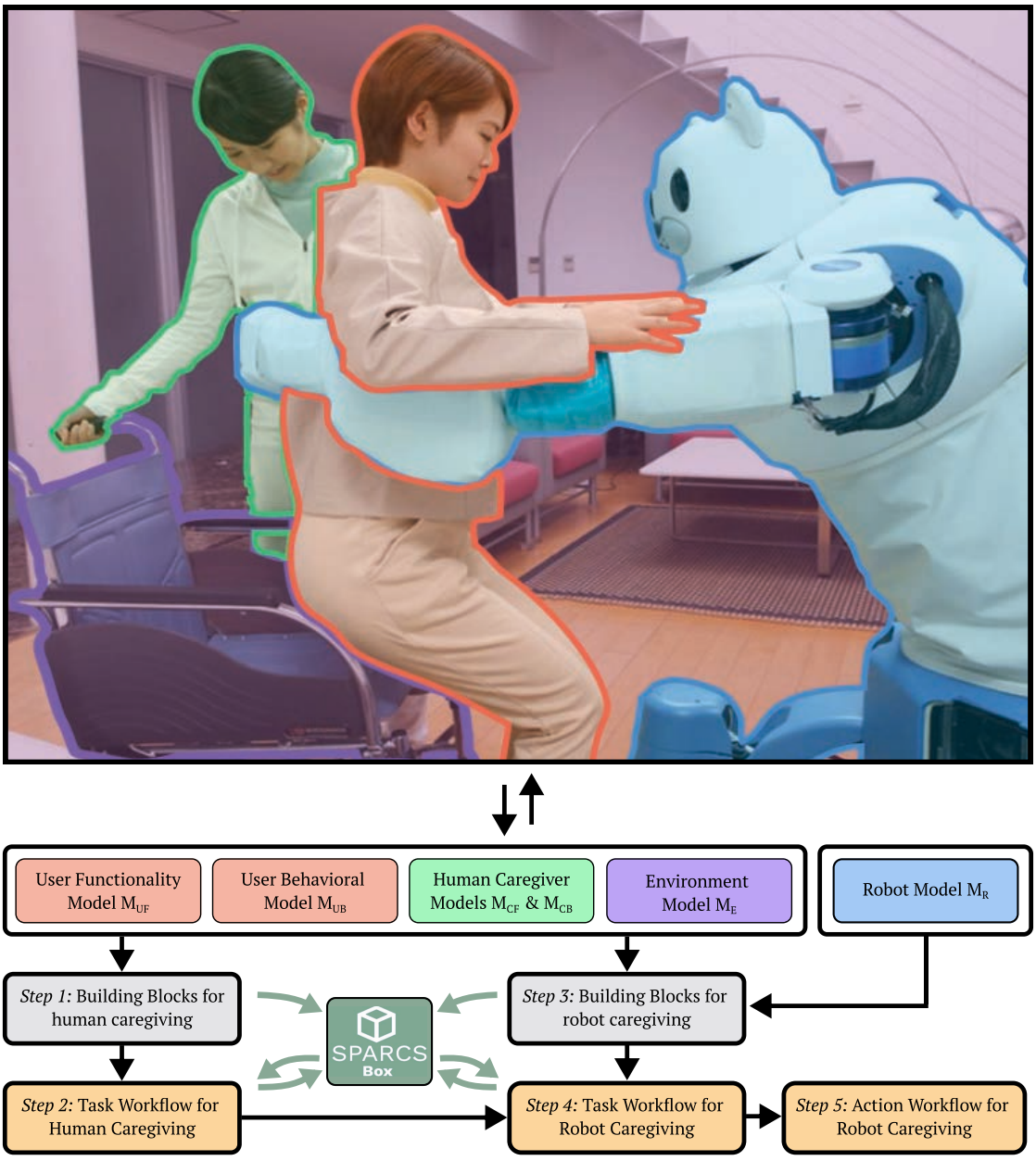


Figure 3.2: Using SPARCS involves the following steps: 1. Instantiate the *Building Blocks* for human caregiving, 2. Communicate with stakeholders using *SPARCS-box* to create the *Task Workflow for Human Caregiving*, 3. Instantiate the *Building Blocks* for robot caregiving, 4. Propose the *Task Workflow for Robot Caregiving* and incorporate feedback from stakeholders using *SPARCS-box*, and 5. Implement the *Action Workflow for Robot Caregiving*. Image shows ROBEAR from RIKEN.

3.3 Building Blocks of SPARCS

Designing a caregiving robot involves reasoning about four *Building Blocks* – the **User**, their **Human Caregiver**, their **Environment**, and the **Robot**. We define these blocks using functionality and behavioral user models, corresponding caregiver models, an environment model, and a robot model.

3.3.1 User Functionality Model

The *User Functionality Model* M_{UF} represents a care recipient's physical and cognitive functioning abilities. Physical functioning should capture body shape and structure, muscle function, joint limits, and involuntary movements. Cognitive functioning should model mental capabilities such as attention, memory, and decision-making abilities. The *Human Caregiver Functionality Model* M_{CF} follows a similar definition, but for the caregiver.

Body shape and structure in M_{UF} can be represented through articulated human-shape models such as SMPL-X [216]. Muscle function can be incorporated using high-fidelity musculoskeletal models [128, 232, 285]. Joint limits are pose-dependent and affected by the mobility limitations of the user. They can be approximated by existing learning-based approaches [72, 126] trained on user movement data. Involuntary movements have been previously simulated using parameterized offset functions [59]. All of the above models for physical functioning can be adapted to user measurements such as body dimensions, range of motion (ROM) [225], and muscle strength [44]. Cognitive functioning of the user can be characterized using measures such as MMSE [66] or SLUMS [192]. The

International Classification of Functioning, Disability and Health [278] framework defines body structures and functions, and is employed by professionals from various fields including health, rehabilitation, and community care. It can be used to model M_{UF} more extensively.

3.3.2 User Behavioral Model

The *User Behavioral Model* M_{UB} represents a care recipient's intent and preferences, which is necessary for providing personalized care [32]. The *Human Caregiver Behavioral Model* M_{CB} follows a similar definition, but for the caregiver.

Intent can be identified through explicit communication or implicit inference. Explicit communication can be established through an interface, such as speech or a GUI [264]. Implicit inference can be made using Bayesian models [118, 119] or modeling in latent space [124]. Preferences can be global or task-environment specific [30]. For example, a user may always prefer the robot to be slow-moving regardless of the scenario, whereas their preference for the level of autonomy may be task dependent. Preferences can be user-specified [31] or modeled using data-driven methods [281].

3.3.3 Environment Model

The *Environment Model* M_{E} represents physical and social information about the user's environment. Physical information constitutes the surrounding scene and the objects (including assistive devices) present in it. Social information records

the environment's social context. For example, it could be an intrapersonal setting involving only the care recipient and their caregiver, or an interpersonal setting with family members, or a community setting [17].

Physical information is necessary for most robotic applications. Environment scenes can be represented using topological maps and semantic point clouds, whereas objects can be encoded using Unified Robot Description Format (URDF) files [208]. Physical information can also be represented using more detailed data structures such as 3D scene graphs [229]. Social information can be captured through theory of mind approaches [237] or data-driven methods [206].

3.3.4 Robot Model

The *Robot Model* M_{R} represents the hardware specifications, kinematics and dynamics, visual and collision model, and onboard sensing capabilities of the robot. It also provides access to the raw sensor data. It is important to explicitly model the robot as it affects ADL assistance [17].

URDFs are commonly used to represent many of the aforementioned attributes. M_{R} can consist of these along with other configuration files that contain information on sensors and their metadata.

For the experiments in this chapter, we initialize each *Building Block* $M\colon K \rightarrow V$ as a dictionary where K and V are mutable sets of keys and values, respectively. K contains keywords corresponding to the attributes of the *Building Block*. V contains instantiations of these attributes. For example, in case of M_{UF}, K may contain keywords like "Active ROM Neck Flexion" or "Passive ROM Neck Extension" with corresponding values stored in V.

3.4 Structured Workflows: Whats and Hows of Robot Caregiving

Given the *Building Blocks*, how do we enable the robot to provide assistance for the entire duration of an ADL? This requires understanding the *whats* that the robot must address in the caregiving scenario. We call these *whats*, i.e., the set of tasks, the *Task Workflow for Robot Caregiving*.

While creating the *Task Workflow for Robot Caregiving*, roboticists must consult other stakeholders who have expertise in caregiving. However, many of these stakeholders have no experience with robotics. They cannot propose robot caregiving workflows directly. Therefore, we advocate first getting their insights on the corresponding human-human caregiving scenario. We call this the *Task Workflow for Human Caregiving*. Roboticists can then use this information to propose an initial *Task Workflow for Robot Caregiving*, and incorporate feedback from other stakeholders on it.

Once the *Task Workflow for Robot Caregiving* is finalized, the focus can shift to answering *how* a robot can perform the constituent tasks. We call this implementation workflow the *Action Workflow for Robot Caregiving*. The two *Task Workflows* and this *Action Workflow* are jointly referred to as *Structured Workflows*.

What would be a good representation for *Structured Workflows*? The representation must allow enough detailing to capture task-related considerations. For example, when feeding Morgan, his caregiver applies downward forces on Morgan's tongue to avoid gag reflex. At the same time, the representation should be hierarchical such that these details can be gradually uncovered as we move

through various layers of abstraction. Hierarchy will also promote the reusability of similar subroutines across different caregiving scenarios. For example, the subroutine of lifting a user's leg while dressing with pants can be reused for lifting their leg during a sponge bath. The representation must also support the specification of concurrent tasks like wiping off water from the user's eyes while applying shampoo on their head. Finally, the representation must be easy to understand and update for facilitating valuable discussions among all stakeholders.

Hierarchical State Machines (HSMs) [92] satisfy all of the above requirements. They offer an intuitive visual formalism for complex systems with various levels of abstraction. We propose to use HSMs as the representation for both *Task* and *Action Workflows*. Depending on the layer of abstraction, states of this machine define the routines or how they are carried out at the corresponding level of detail. Each state has pre-conditions and post-conditions that govern transitions in and out of it.

We define the abstractions for the two *Task Workflows* in a top-down manner:

$$\textit{Activity} \rightarrow \textit{Composite Task} \rightarrow \textit{Task}$$

An *Activity* is an instantiation of a given ADL. It maps to a set of *Composite Tasks* depending on the *User Functionality Model* and the initial state of the *Environment Model*. *Composite Tasks* further split into *Tasks* conditioned on all the *Building Blocks*.

In case of *Task Workflow for Robot Caregiving*, we map the constituent *Tasks* to their corresponding *hows*, i.e., the *Action Workflow for Robot Caregiving*. We define the abstractions for *Action Workflows* as:

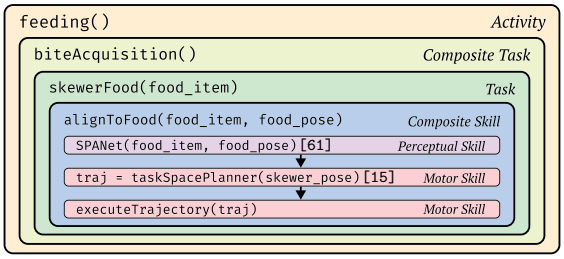


Figure 3.3: Snippet of a *Structured Workflow* for robot-assisted feeding showing all the layers of abstraction.

$$\textit{Composite Skill} \longrightarrow \textit{Motor/Perceptual Skill}$$

Each *Task* maps to a set of *Composite Skills*. *Composite Skill* further comprises *Motor* and *Perceptual Skills*. *Motor Skills* represent the robot's ability to reason about its physical movements. *Perceptual Skills* represent the robot's ability to interpret sensor data. Each layer of abstraction in *Structured Workflows* is well-scoped with respect to its parent as detailed on our website [183]. Figure 3.3 shows a snippet of a *Structured Workflow* for robot-assisted feeding.

3.5 SPARCS-box: Bridging the Gap between Roboticists and Caregiving Community

Building *Structured Workflows* requires roboticists to consult other stakeholders who have the required domain knowledge in caregiving. However, this is currently challenging due to limited access to stakeholders. To alleviate this problem, we propose *SPARCS-box* [182], a web-based platform that facilitates discussion between all stakeholders.

SPARCS-box allows stakeholders to sign up and propose caregiving scenarios of their choice. A caregiving scenario can be specified by detailing the *Activity* and its *Building Blocks*. Stakeholders can collaborate on the platform to design

its *Task Workflow for Human Caregiving*. Roboticists can then use this information to propose the corresponding *Task Workflow for Robot Caregiving*. They can get feedback on it from other stakeholders on the platform. Once this workflow is finalized, roboticists can work on well-defined robot caregiving problems. They can pick *Tasks* from this workflow and build *Skills* that contribute to the *Action Workflow for Robot Caregiving*. We provide more details on all the functionalities of *SPARCS-box* on our website [183].

3.6 Importance of User Modeling in SPARCS

Physical robot caregiving requires instantiating detailed *User Functionality Model* M_{UF} and *User Behavioral Model* M_{UB}. These models help in building more effective and personalized systems. To illustrate this, we consider *bite transfer* and *bite sequencing* for a robot-assisted feeding activity with a wheelchair-mounted robot arm. Bite transfer involves transferring a food item from a utensil into the mouth of a care recipient. Bite sequencing involves deciding the next food item to feed a care recipient. We demonstrate how algorithms that leverage user-informed *a)* M_{UF} for bite transfer and *b)* M_{UB} for bite sequencing perform better than baselines.

3.6.1 Leveraging M_{UF} for Bite Transfer

To show the effect of M_{UF} for bite transfer, we select the user Natalia (Table 3.1) for our experiments.

Experiment setup: Our bite transfer policy transfers a food item from its initial fixed position facing Natalia to near her mouth. Her head pose is denoted by

Table 3.1: Information for the six care recipients. We use initials of **F**eeding, **D**ressing, **B**athing, and **T**ransferring to denote their need for assistance with these ADLs.

Identifier	Cause of Disability	Needs Assistance
Morgan (he/him)	Brainstem Stroke	F, D, B, T
Jose (they/them)	Spinal Cord Injury (C1-C3)	F, D, B, T
Natalia (she/her)	Spinal Cord Injury (C4-C5)	F, D, B, T
Daniel (he/him)	Spinal Cord Injury (C6-C7)	D, T
Kim (she/her)	Cerebral Palsy	D, B, T
Karan (he/him)	Left-side Hemiplegia	D, T

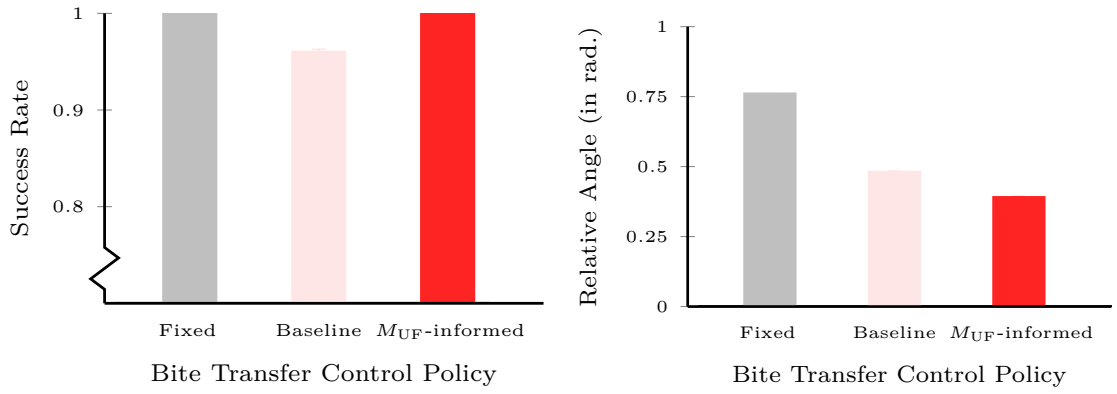


Figure 3.4: Fixed and M_{UF}-informed policies have a high bite transfer success rate, while the latter also results in minimal neck movement.

h_{user}. We use active ROM data for her neck to instantiate the manifold $H_{M_{\text{UF}}}$ of attainable head poses. We denote the set of all head poses attainable by an individual with full neck mobility by H_{all}. We consider the test set spanning all possible $h_{user} \in H_{M_{\text{UF}}}$.

Methods: We compare three bite transfer control policies: (i) Fixed [70] which executes a trajectory to an assumed fixed head pose $h_{fixed} \in H_{M_{\text{UF}}}$, (ii) Baseline which uses H_{all}, and (iii) M_{UF}-informed which uses $H_{M_{\text{UF}}}$.

M_{UF}-informed begins with sampling a set H_{cand} comprising candidate head poses from M_{UF} where the bite transfer can happen. H_{cand} also includes h_{user}

and h_{fixed}. These poses are ordered according to relative angle from h_{user}. The algorithm sequentially iterates over each sample in H_{cand}. For each head pose sample $h_{cand} \in H_{cand}$, the corresponding end effector goal pose x_{goal} is found. Natalia aligns her mouth opposite to the fork while taking the food item. Thus, x_{goal} is assumed to be at a fixed transform to h_{cand}. We then use task-space-region planner [15] to find a collision-free trajectory from the initial configuration of the robot to x_{goal}. If the trajectory is successfully found, it is returned. If no trajectory is found even after iterating over all the samples, the algorithm terminates.

Baseline follows a similar approach. However, as it does not have access to $H_{M_{\text{UF}}}$ or h_{fixed}, it instead constructs H_{cand} using h_{user} and head poses sampled from H_{all}.

Metrics: Performance of the above methods is averaged over ten seeds for the test set of sampled h_{user} and evaluated on two metrics. (1) *Success rate*, which is the percentage of test samples where the bite transfer control policy can generate a feasible trajectory. A trajectory is said to be feasible if it is collision-free and reaches a pose that results in successful bite transfer. (2) *Relative angle* by which a user has to move their neck to take the bite off the fork.

Results: Among the compared policies, M_{UF}-informed can always successfully find a feasible control policy and requires the smallest neck movement (Fig. 3.4). Our experiments illustrate that access to user-informed M_{UF} leads to more efficient physical robot caregiving.

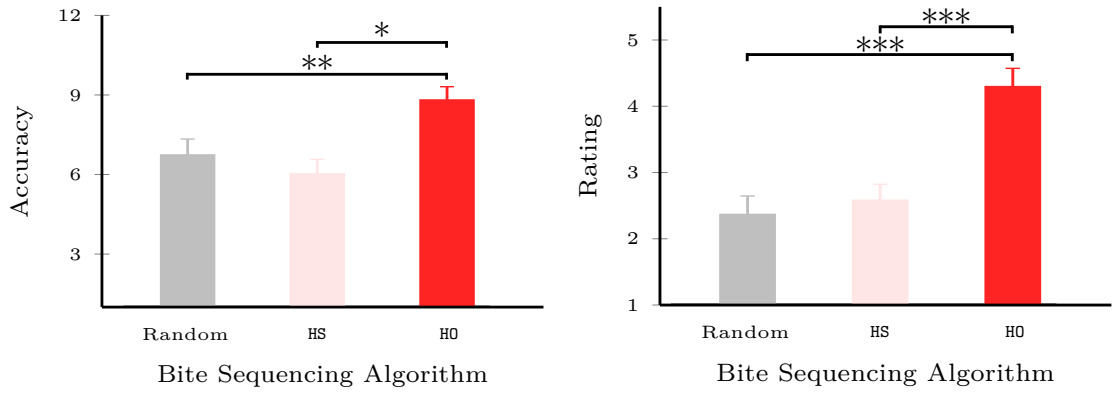


Figure 3.5: HMM-online HO outperforms other methods in both accuracy and satisfaction rating. *, **, *** denote statistically significant differences with $p_{0.05}$, $p_{0.005}$, $p_{0.0005}$ respectively.

3.6.2 Leveraging M_{UB} for Bite Sequencing

We show the effect of modeling M_{UB} for bite sequencing on user satisfaction. Among a set of candidate models for capturing M_{UB}, we perform a user study to identify the model that has the highest overall user satisfaction.

Experiment setup: We consider a meal consisting of three bites each of four unique food items – banana, kiwi, grape, and carrot. Assuming the user eats one item at a time and consumes all the food items, the bite sequence for a meal is an ordered sequence of these food items. We perform a user study with 14 participants. For each participant, we initially collect two data points. First, an affinity score $\in [1, 5]$ for each food item that implies how much they like it. Second, their high-level eating preference among (a) *I would save my favorite food for the last*, (b) *I would eat my favorite food first*, and (c) *I prefer to mix and match*. We then ask the participants to record their bite sequences for six meals. Using this data, the goal is to learn a preference model as part of M_{UB} and use it for predicting the bite sequence during the seventh meal.

Methods: Our approach comprises (i) generating simulated bite sequences for a user using their affinity score and high-level eating preference, (ii) training a hidden Markov Model (HMM) with discrete emissions using the simulated data, and (iii) updating the model using the six bite sequences obtained from the user. We use HMM-simulated (`HS`) to refer to the model obtained after (ii) and HMM-online (`HO`) for the final updated model after (iii). We compare with a baseline that randomly selects food items.

Metrics: Performance of the above methods is evaluated on two metrics. (1) *Prediction accuracy* during the seventh meal. This is calculated by presenting the predictions from all the methods as choices to the participant and comparing the predictions with the item selected by the participant. (2) For each method, we generate a bite sequence assuming the participant always chooses the item predicted by that method, and ask them to provide a *satisfaction rating* $\in [1, 5]$ for the generated sequence.

Results: We perform Kruskal-Wallis H-tests and Tukey HSD post-hoc tests and observe that `HO` significantly outperforms other methods in both accuracy and satisfaction rating (Fig. 3.5). This user study demonstrates that modeling the user preferences for bite sequencing and adapting it to user data leads to improved satisfaction. Refer to our website [183] for more details on the user study.

3.7 SPARCS in Practice: Identifying Care Requirements and Building a Caregiving Robot

SPARCS is a framework for building physical caregiving robots. Using SPARCS involves the following steps (Fig. 3.2):
A. Identifying care requirements:

1. Instantiate *Building Blocks* for human caregiving.
2. Communicate with stakeholders using *SPARCS-box* to create *Task Workflow for Human Caregiving*.

B. Building a physical caregiving robot:

3. Instantiate *Building Blocks* for robot caregiving.
4. Propose *Task Workflow for Robot Caregiving* and incorporate feedback from stakeholders using *SPARCS-box*.
5. Implement *Action Workflow for Robot Caregiving*.

We demonstrate each of these steps in this section. For steps 1 and 2, we consider six care recipients with varying functional abilities (Table 3.1). We instantiate various caregiving scenarios for each of these care recipients and identify their care requirements (Sec. 3.7.1). For steps 3, 4, and 5, we consider one of the above identified scenarios – robot-assisted feeding of bite-sized food items to Natalia while she is watching television (Sec. 3.7.2).



Figure 3.6: Data Collection for Morgan's M_{UF}, *left to right*: Active Range of Motion, Passive Range of Motion, Manual Muscle Testing

3.7.1 Identifying Care Requirements for Six Care Recipients

We consider six care recipients with different functional abilities to capture variability in care requirements (Table 3.1).

Step 1. *Instantiate Building Blocks for Human Caregiving:*

For each of the six care recipients, we instantiate corresponding *User Functionality Model* M_{UF}, *User Behavioral Model* M_{UB}, *Caregiver Models* (M_{CF} and M_{CB}), and *Environment Model* M_{E}. We collect clinical data for information on M_{UF} (Fig. 3.6) – body dimensions, weight, active and passive ROM [225] and manual muscle testing [44] – along with textual descriptions, and videos of non-disabled medical professionals simulating the functional abilities of these care recipients. For each care recipient, we look at ADLs – feeding, dressing, bathing, and transferring. Among these ADLs, we identify scenarios they require assistance with based on their functional abilities. We identify a total of 19 caregiving scenarios and provide the corresponding *Building Blocks*.

Step 2. *Create Task Workflow for Human Caregiving:*

One should use *SPARCS-box* to create *Task Workflow for Human Caregiving*. For this chapter, we conduct a user study interviewing 9 occupational therapists (2 male; 7 female), between the ages of 27 and 51. Through this study, we record *Task Workflows* for the 19 caregiving scenarios.

The *Building Blocks* and above workflows for all the identified caregiving scenarios are publicly available on *SPARCS-box*. This information can be leveraged by roboticists to build robots that can address the identified care requirements.

3.7.2 Building a Physical Caregiving Robot

In this section, we focus on the scenario of feeding Natalia while she is watching television.

Step 3. *Instantiate Building Blocks for Robot Caregiving:*

We instantiate the *Building Blocks* for the above scenario from the data collected in *Step 1* and include our *Robot Model* in it.

- *Environment Model* M_{E}: Natalia is sitting at the dining table on a wheelchair with a robot arm mounted beside its right armrest. The dining table is in her living room in front of the television. There is a plate in front of her, consisting of solid bite-sized food items. This scene is captured using a pointcloud obtained from an RGBD camera. Objects in this scene are represented using URDFs [208]. The shape and pose of Natalia's head are represented using FLAME [62].

- *User Functionality Model* M_{UF}: Natalia has complete paralysis in all limbs. She has partial mobility in her neck. We use the data collected in *Step 1* to represent her M_{UF}.
- *User Behavioral Model* M_{UB}: Natalia prefers the robot to autonomously take decisions without any input from her, and shows her intent of taking a bite by opening her mouth. She tends to focus on the television while eating and thus favors making minimal neck movements to transfer the food from the fork to her mouth. Among the set of food items, she has a preferred order for eating the food items. She expects the robot to learn this ordering. We use HMM-online (Sec. 3.6.2) to model this preference.
- *Robot Model* M_{R}: We use the Kinova Gen3 6 degrees-of-freedom (DoF) robot arm with a Robotiq 2F-85 gripper, holding a custom fork fitted with an ATI Force/Torque sensor.

Step 4. *Create Task Workflow for Robot Caregiving:*

We build the *Task Workflow for Robot Caregiving* using the *Task Workflow for Human Caregiving* identified in *Step 2*. This *Activity* can be broken down into two *Composite Tasks*: (i) *Bite Acquisition*: acquiring a food item from the plate, and (ii) *Bite Transfer*: transferring this food item into the mouth of the care recipient. Natalia's M_{UB} specifies that she prefers the caregiver to feed her without any input. Thus, *Bite Acquisition* comprises the *Task* of the robot autonomously moving above the plate and then skewering the required food item. According to Natalia's M_{UF}, she has partial mobility in her neck. She can eat the food item off the fork if the robot brings it to a region within her functional abilities. *Bite Transfer* begins with the robot moving to a fixed pose in front of Natalia such that her head is visible to its camera. According to her M_{UB}, the robot waits for her to

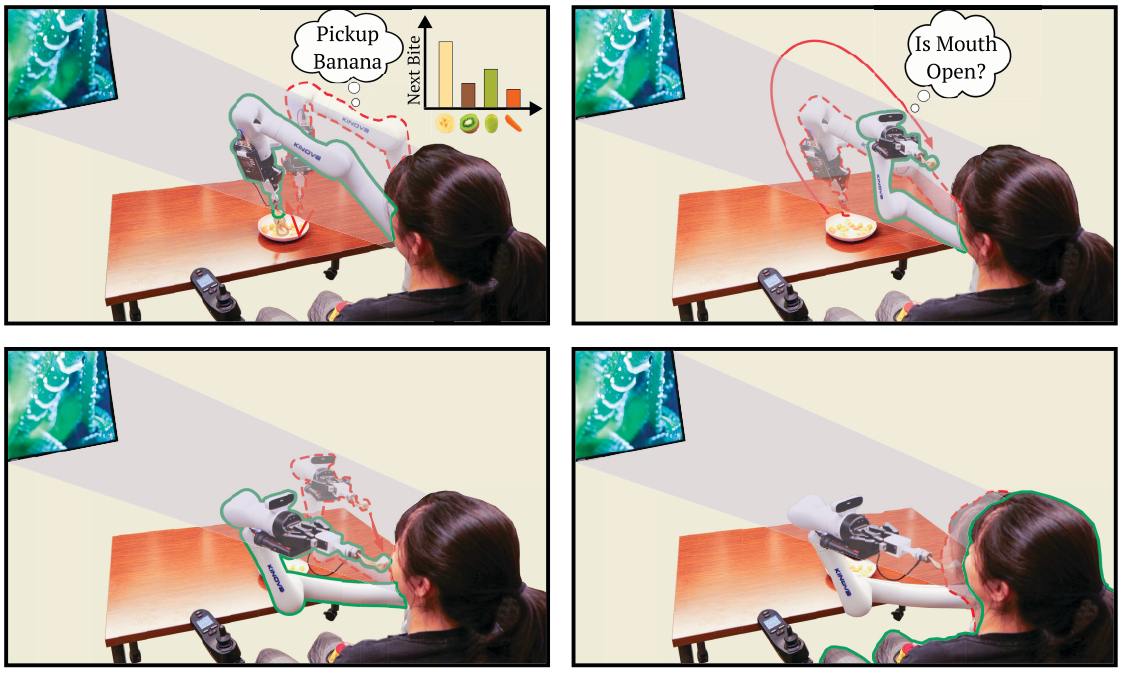


Figure 3.7: Robot-assisted feeding for an individual simulating Natalia while she watches TV. *Top left*: Robot performs *Bite Acquisition* using SPANet [61]. *Top right*: It enters *Bite Transfer* and waits for the user to open mouth. *Bottom left*: It moves close to Natalia's mouth. *Bottom right*: Natalia leans forward and takes the bite.

open her mouth. Once it is open, the robot brings the food item near her mouth, ensuring minimal neck movement for transfer. Natalia then leans towards the food item and takes it off the fork.

Step 5. *Implement Action Workflow for Robot Caregiving:*

We demonstrate the *Action Workflow for Robot Caregiving* with an individual simulating Natalia. For *Bite Acquisition*, we deploy SPANet [61] for selecting how to skewer a food item from a set of candidate skewering actions. For *Bite Transfer*, we deploy the M_{UF}-informed policy (described in Sec. 3.6.1). The demonstration of our robot feeding an individual simulating Natalia can be seen in Fig. 3.7 and found on our website [183].

3.8 Adapting Robot-Assisted Feeding to Different Caregiving Scenarios

In this section, we perform experiments to exhibit the adaptability of our system. We show how robot caregiving scenarios instantiated using SPARCS allow us to reuse and transfer insights across differing users, environments, and robot models.

3.8.1 User Functionality and Behavioral Models

We adapt the *Structured Workflows* for Natalia to Jose (Table 3.1) while keeping all other Building Blocks fixed:

- New M_{UF}: Jose has severe neck mobility limitations with partial neck rotation ROM. Unlike Natalia, they require their caregiver to place a food item inside their mouth cavity for *Bite Transfer*.
- New M_{UB}: Unlike Natalia, Jose shows their intent for wanting a bite by turning towards the robot, and expresses their consent for the food item to be placed inside their mouth by opening it. They prefer the food item to be placed one-third inside their mouth cavity. Similar to Natalia, they expect the robot to learn their preferred bite sequence.

In the *Structured Workflows* for this scenario, while *Bite Acquisition* remains similar, the *Task* breakdown for *Bite Transfer* changes. *Bite Transfer* begins with the robot moving to a fixed pose in front of Jose such that their head is visible to its camera. In accordance with their M_{UB}, the robot waits for them to turn towards the robot and then moves the food item to in front of their mouth. Once Jose's



Figure 3.8: Robot-assisted feeding for an individual simulating Jose. *Top left*: Jose watches television. *Top right*: They show the intent of taking a bite by looking at the robot. *Bottom left*: Robot performs an inside-the-mouth transfer due to Jose's mobility limitations. *Bottom right*: Robot retracts leaving food item inside the mouth.

mouth is open, the robot moves the food item inside their mouth cavity to successfully transfer the bite. The demonstration of our robot feeding an individual simulating Jose can be seen in Fig. 3.8 and found on our website [183].

3.8.2 Robot Model

We consider a different *Robot Model* M_R while keeping the same user and environment models:

- New M_R: We use the Kinova Gen3 7-DoF robot arm with a Robotiq 2F-85 gripper, holding a custom fork fitted with an ATI Force/Torque sensor.

As the robot has a similar morphology to the one considered in Sec. 3.7.2, the *Structured Workflows* remain the same. However, the change in its dimensions

Table 3.2: M_{R} comparison (averaged over three seeds)

Robot Arm	Success Rate	Relative Angle (in rad.)
Kinova Gen3 6-DoF	1.0	0.3996 ± 0.0018
Kinova Gen3 7-DoF	1.0	**0.3496 ± 0.0008**

and DoF affects its performance in *Bite Transfer* as shown in Table 3.2. We observe that the new M_{R} performs better than the previous model which could be an effect of having better manipulability.

3.8.3 Environment Model

We consider a different *Environment Model* M_{E} while keeping the same user and robot models:

- New M_{E}: Natalia is sitting at her dining table on a wheelchair with a robot arm mounted beside its right armrest. There is a plate in front of her, consisting of solid bite-sized food items. She is dining along with her friends in a social setting.

While dining in this social setting, Natalia prefers the robot to bring the food item to a fixed position on her right side for *Bite Transfer*. This is to avoid any form of social distraction due to the robot motion. The demonstration of our robot feeding an individual simulating Natalia in the new M_{E} can be seen in Fig. 3.9 and found on our website [183].

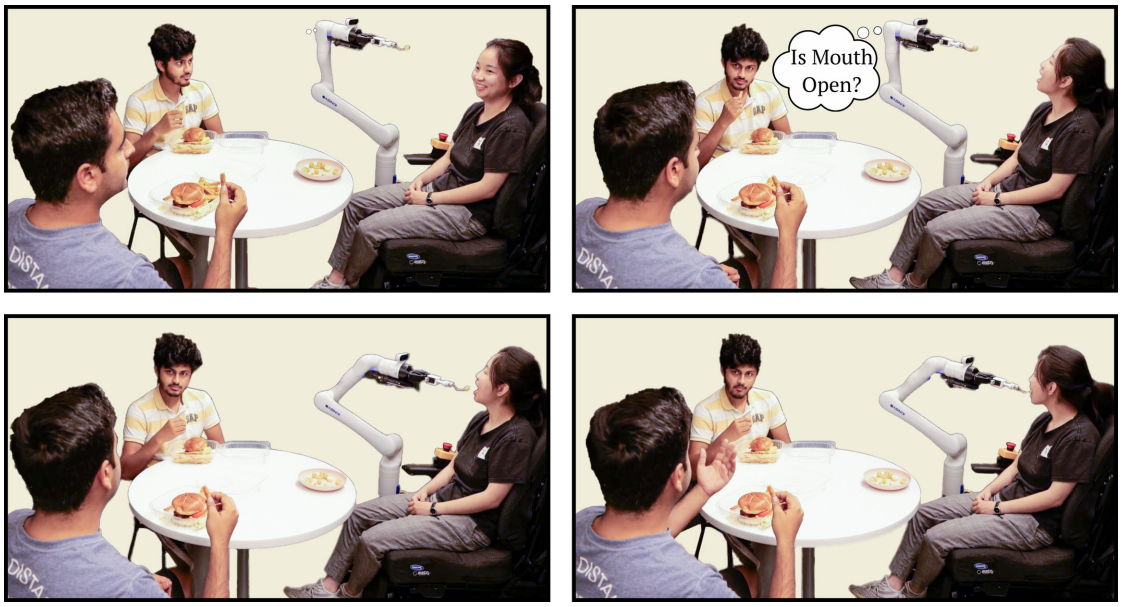


Figure 3.9: Robot-assisted feeding for an individual simulating Natalia in a social dining scenario. *Top left*: Natalia having a conversation. *Top right*: In this M_E, Natalia faces the robot and opens her mouth. *Bottom left*: Robot moves close to her mouth. *Bottom right*: Natalia leans forward and takes the bite.

3.9 Open Challenges

As more roboticists get involved in physical robot caregiving, it is important to identify open problems that one can work on. Therefore, we reached out to care recipients and caregivers to better understand the critical factors that affect caregiving for ADLs. We used SPARCS to create caregiving scenarios around them and obtained recommendations on ADL assistance. We conducted a user study with 8 care recipients (6 male; 2 female) and 2 caregivers (both female), between the ages of 21 and 60. All data collection and user studies in this chapter were approved by the Cornell University Institutional Review Board. Here we list some key open challenges:

Feeding. While there has been a lot of work on robot-assisted feeding, many challenges still need to be addressed for real-world applicability. Through our user study, we identify four unique challenges in feeding: (i) determining the level of autonomy because it is not only personalized to a user but also to different tasks within feeding, (ii) acquiring a variety of food items with different physical characteristics across shape, size, texture, compliance, etc., (iii) designing the embodiment such that it can potentially perform other tasks while still being well suited for feeding, and (iv) determining when to switch between human supervision and autonomous control with our users preferring to switch to human supervision at the tail-end of a task (e.g. precisely aligning to a food item for *Bite Acquisition*) for finer control.

Dressing and Bathing. Shared autonomy methods purely based on joystick-based control have been considered viable for controlling caregiving robots. However, they may not be feasible for many tasks within dressing and bathing. For example, when dressing a user with a t-shirt, the user may not be able to see around and control the joystick due to occlusion. Thus, a significant challenge here is to build caregiving robots that allow multiple modes of interfaces to control the robot. Participants also highlighted that dressing the lower body is more complex than the upper body. In most cases, lower body dressing is performed while the care recipient is lying on their bed. Dressing them in this state requires high payload capacity and reasoning under partial observability. These requirements make the physical interaction in this task particularly challenging. Also, bathing and dressing tasks would conceivably require a robot to collaborate with the user and their caregiver(s), and task allocation during physical caregiving is a major challenge.

Transferring. At first glance, though transferring may seem to require a robot with high payload capacity, our users highlight that existing robot hardware can still be used. This is possible by collaborating with human caregivers to operate assistive devices such as Hoyer lift, sit-to-stand lift, sliding board, etc. A major challenge is to design intelligent collaboration policies, and enable seamless integration between these robots and other assistive devices. As pointed out by a care recipient, transferring can begin with coarsely moving them to a bed or a wheelchair using an assistive device such as a Hoyer lift. The robot can then be used for finer limb-repositioning to successfully complete this ADL.

3.10 Discussion

We introduced SPARCS, a framework for physical robot caregiving. SPARCS enables roboticists to translate real-world care requirements into guidelines for physical robot caregiving. Occupational therapists use similar frameworks when designing caregiving interventions. Compared to these frameworks, we define the components of SPARCS by grounding them in robotics. We release *SPARCS-box*, which allows roboticists to communicate with stakeholders from the caregiving community.

SPARCS currently defines *Building Blocks* and *Structured Workflows* in a fairly subjective manner. Though we highlighted some possible ways of representing them, it remains to be explored what an ideal representation would be. Through this work, we are taking the first steps towards systematically structuring this impactful but scattered field of physical caregiving robotics. We hope that our framework will speed up the progress in this domain, bringing us one step closer to caregiving robots that can provide long-term assistance.

Part I

Deployment for a Single Session: Personalizing to Functional Abilities

CHAPTER 4

FEEL THE BITE: INSIDE-MOUTH BITE TRANSFER

4.1 Introduction

Robot-assisted feeding can be broken down into two stages: bite acquisition and bite transfer [180] (Chapter 3). Bite acquisition involves acquiring a bite-sized food item with an appropriate utensil [61, 80, 81, 83], whereas bite transfer entails moving it from above the plate to the care recipient's mouth [13, 70, 244]. In this chapter, we focus on bite transfer. Most prior works bring a food item in front of the mouth of a care recipient, who is then expected to lean forward to take the bite. However, for those with severe upper body and neck mobility limitations, leaning forward for an outside-mouth bite transfer can be functionally impossible, necessitating the direct placement of food in their mouths. Even for care recipients who can lean forward, having to repeatedly make this movement can be exhausting. In this work, we present a robot-assisted feeding system that can perform inside-mouth bite transfer and demonstrate its utility for assisting a diverse group of care recipients with severe mobility limitations.

Care recipients who need inside-mouth bite transfers often have complex medical conditions – including limited mouth opening [84], involuntary movements (spasms), and requirement of food transfer at a specific location inside their mouth – which makes feeding them extremely challenging. Involuntary motions, diverse in their type and occurrence, demand alert and adaptable feeding strategies. For example, if an involuntary forward spasm occurs just as the utensil approaches the care recipient's mouth, the caregiver must quickly retract

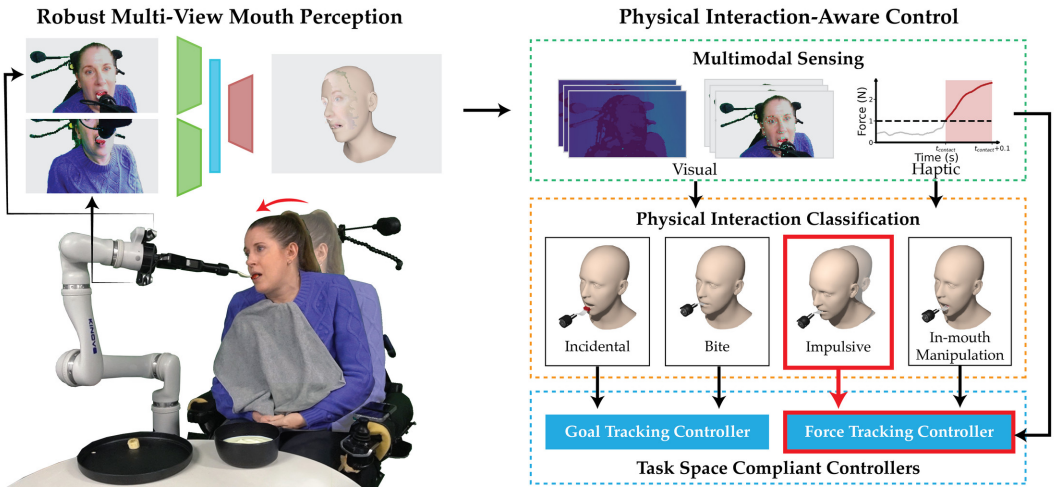


Figure 4.1: We propose an inside-mouth bite transfer system that uses two key components, (i) robust multi-view mouth perception, and (ii) physical interaction-aware control, to successfully feed care recipients with diverse mobility limitations.

the utensil to avoid harm. If such a motion happens with the utensil already inside the mouth, the caregiver should comply with the motion, ensuring the utensil moves in harmony with it to avoid injury. Another layer of intricacy emerges when food must be placed precisely within the mouth at a preferred transfer location. In these instances, care recipients may use their tongue to guide the utensil, necessitating that caregivers recognize and act upon this cue. In addition to this, physical interactions also occur when slight errors in sensing/control lead to incidental contacts, and when care recipients intentionally bite down on the food. Consequently, feeding individuals with such complex needs requires precision in perception and control, and identifying and appropriately reacting to various types of physical interactions.

In this chapter, we make the following contributions:

Inside-Mouth Bite Transfer System. We propose a system that leverages two key components – robust multi-view mouth perception, and physical

interaction-aware control – to address the aforementioned challenges and successfully feed care recipients with severe mobility limitations food inside their mouth.

Robust Multi-View Mouth Perception Method. Existing methods for mouth perception in robot-assisted feeding rely on a single in-hand camera view, and thus face challenges during inside-mouth bite transfer due to significant occlusion from the feeding utensil. To address this, we introduce a novel multi-view mouth-perception approach that is robust to tool occlusion (Section 4.2.1). This method allows for uninterrupted real-time mouth perception throughout the transfer process, enabling detection and adaptation to both voluntary and involuntary head movements.

Physical Interaction-Aware Control Method. We propose an interaction-aware controller that uses multimodal sensing for classifying the nature of physical interactions in real-time and reacts accordingly (Section 4.2.2). Adopting a data-driven approach, we collect a multimodal dataset comprising various types of physical interactions that can occur and train time-series classification models. Based on the detected interaction type, we switch between a goal-tracking controller and a force-tracking controller.

Evaluation of System Components. We demonstrate the necessity of both novel components through evaluations with baselines and two ablation studies involving participants without mobility limitations (Section 4.3).

Full System Evaluation with Care Recipients. We demonstrate our system's efficacy through a user study with 13 care recipients with diverse medical conditions, all necessitating assistance in feeding. Findings suggest users perceive our

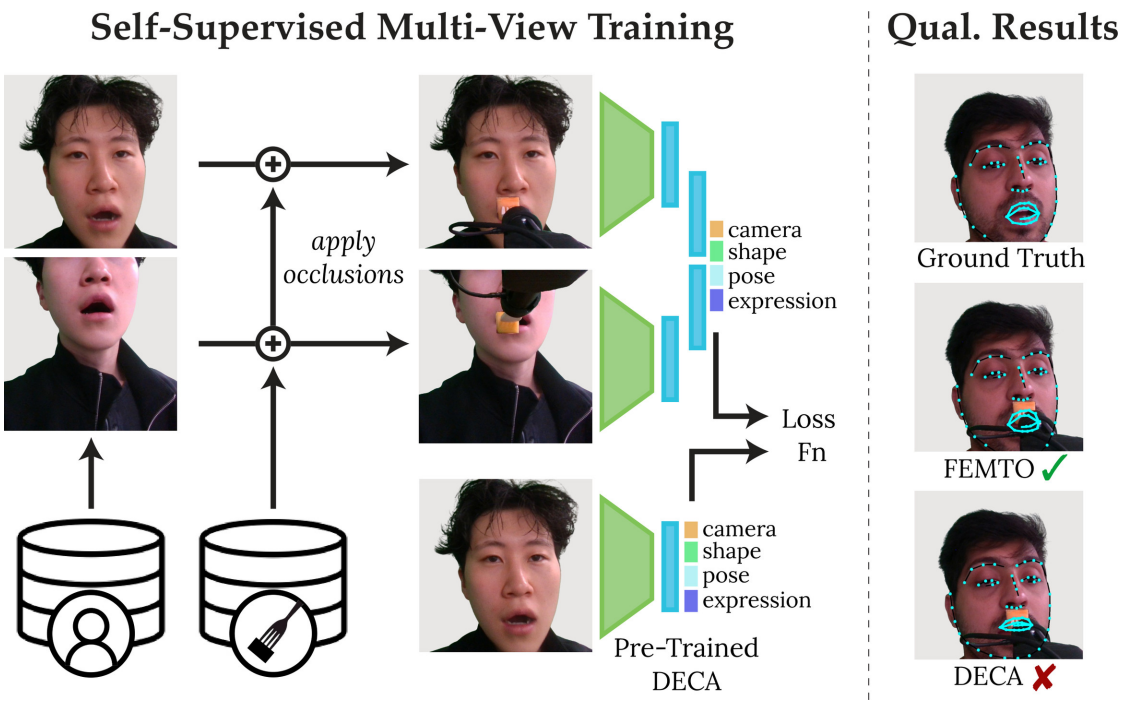


Figure 4.2: The self-supervised multi-view training of FEMTO (left) enables robust mouth perception under tool occlusion (right).

system as safe and comfortable, and view the technology favorably as measured using a Technology Acceptance Model (TAM) survey [53].

4.2 Inside-Mouth Bite Transfer System

Our inside-mouth bite transfer system (Figure 4.1) consists of a Kinova Gen3 6-degree-of-freedom robotic arm [140] with a Robotiq 2F-85 gripper [227] grasping a custom-built feeding tool. It uses two Intel RealSense D415 RGBD cameras mounted on the robot's wrist, one above and one below the utensil, for visual input, and a 6-axis ATI Nano25 F/T sensor [8] for haptic feedback. Our system leverages two key novel components – robust multi-view mouth perception (Section 4.2.1) and physical interaction-aware control (Section 4.2.2).

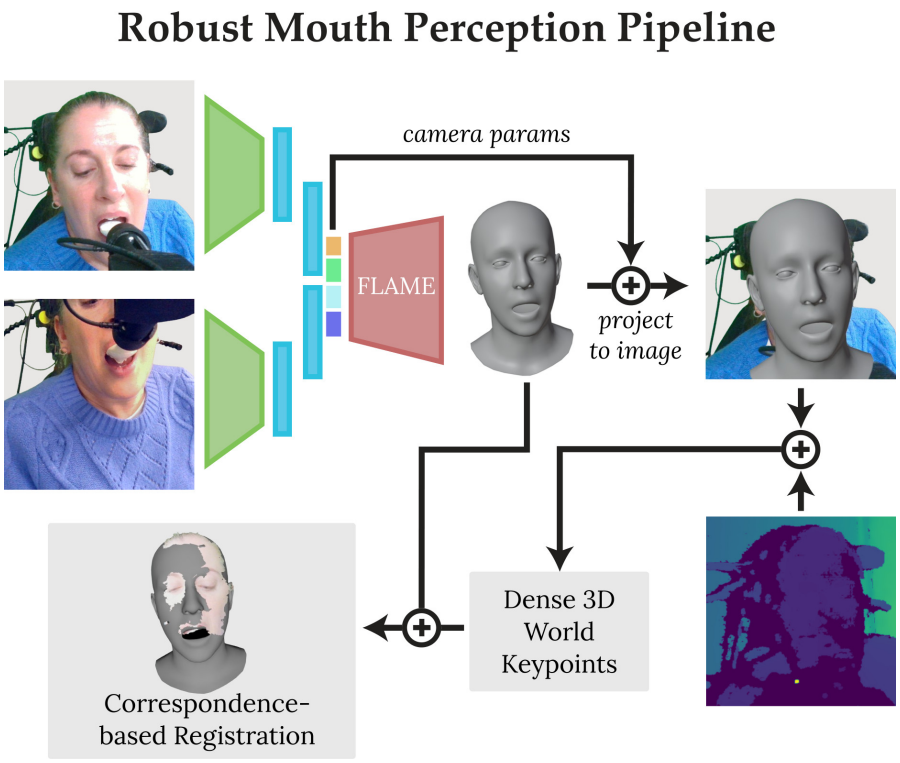


Figure 4.3: Our perception pipeline leverages FEMTO to accurately estimate mouth pose despite noisy depth due to occlusion.

4.2.1 Robust Multi-View Mouth Perception

We require accurate mouth perception for the entire duration of inside-mouth bite transfer to successfully feed care recipients having small mouth openings, involuntary motions, and requirements of food placements at specific locations inside their mouth. However, real-time mouth perception using a single in-hand camera is challenging as there is significant occlusion from the utensil due to which: (i) state-of-the-art monocular methods such as DECA [62] fail at mouth keypoint detection, and (ii) even if we obtain the mouth keypoints, projecting them on the depth image for pose estimation in the real world fails as the depth image is noisy in the vicinity of the utensil. We propose a mouth perception pipeline that is robust to this occlusion challenge. Central to this pipeline is

a novel method – Face Estimation from Multiple Views under Tool Occlusion (FEMTO). FEMTO uses inputs from two in-hand cameras mounted diametrically opposite to reconstruct a personalized head model for the care recipient, and estimates dense 2D facial keypoints. For the personalized head model, we use the parameterized head model FLAME [159] which is represented by a small number of shape, pose, and expression parameters.

Model Architecture. FEMTO's architecture consists of two parallel encoders: one encoder for the top image and the other for the bottom. These encoders use frozen weights from DECA [62], a single-view model pre-trained on 2 million images from large datasets [34, 42, 271]. The outputs of these encoders are concatenated and further processed through two fully-connected layers. These layers integrate information from both cameras, and, similar to DECA, generate FLAME parameters and an orthographic camera pose for projecting the 3D FLAME mesh into image space.

Self-Supervised Finetuning on Self-Curated Dataset. To make FEMTO robust to occlusions, we finetune it on a self-curated visual dataset. We collect about 5000 unoccluded multi-view images with different head poses and facial expressions from 10 participants. Data from 8 participants are used for training FEMTO, and data from the remaining 2 participants are used for comparison against baselines. The latter is detailed in Section 4.3.1. These images are captured without the utensil in place, and thus DECA can typically generate accurate FLAME parameters and camera pose for these images. We then synthetically occlude these images by in-painting the utensil (Figure 4.2). We use various utensils holding distinct food items at this step to ensure that FEMTO

can generalize to new utensils and food items. We provide these occluded images to FEMTO, and use annotations generated by DECA for the corresponding un-occluded image as ground truth for supervision. We evaluate all the generated ground truth parameters and manually update them wherever necessary. FLAME's modular parameterization allows for focusing training on parameters where DECA especially struggles under utensil occlusion, notably jaw pose prediction which is vital for detecting whether the mouth is open or closed.

Robust Mouth Perception Pipeline. FEMTO processes the top and bottom images to generate FLAME parameters and the orthographic camera pose of the top RGB image relative to the head model (Figure 4.3). These FLAME parameters decode into a custom 3D head model, and the generated camera pose can be used to project this model to the top RGB image and obtain dense 2D facial keypoints. We combine these 2D keypoints with the aligned depth image from the top camera to get dense 3D world keypoints. Finally, we use a correspondence-based registration robust to outliers to pose the generated head model to the 3D world keypoints. Despite the depth data around the mouth region being noisy due to occlusion from the utensil, the customized head model enables us to use depth data from other regions of the head for accurate mouth pose estimation. This proposed pipeline runs real-time (5–10 Hz) on a system with RTX 3090.

Details on data collection, model training, and correspondence-based registration method used are in Appendix A.3.

4.2.2 Physical Interaction-Aware Control

Our physical interaction-aware control method first classifies the subtle physical interactions during bite transfer and switches appropriate compliant controllers

accordingly. We take a data-driven approach to identify the nature of physical interaction, and collect visual and haptic data for four types of physical interactions:

1. Incidental interactions (collisions) between the utensil and user's mouth that occur outside the mouth due to sensing/control errors as the robot attempts to move inside.
2. In-mouth manipulation interactions initiated by the user using their tongue to guide the utensil to a desired transfer location inside their mouth.
3. Impulsive interactions due to involuntary spasms occurring while the utensil is inside the mouth.
4. Bite interactions that occur when the user takes a bite.

Data Collection. We collect a total of 3072 physical interactions (512 interactions $\times$ 6 participants without mobility limitations), varying both the utensil and the robot controller. For each physical interaction, we ask participants to perform different variations of the interaction. For example, when taking bites, participants are instructed to use only their teeth around the food item, only their lips around the utensil, or both.

Multimodal Classification. For each physical interaction, we analyze visual and haptic data from a 100 ms window starting at contact initiation. Visual features include 2D and 3D face keypoints generated from our mouth perception pipeline (Section 4.2.1). In line with related haptic classification work [68, 79], we augment raw haptic data with time-series features computed over the window, such as mean, range, kurtosis, Hjorth complexity, and frequency domain.

We evaluate four models: Support Vector Machines (SVM) [48], Multi-Layer Perceptron (MLP) [97], Temporal Convolutional Networks (TCN) [148], and Time-Series Transformers (TST) [291].

Controller Switching. Once the robot recognizes the nature of physical interaction, it switches to an appropriate controller using an event-driven control method. We use two compliant controllers during inside-mouth bite transfer: a goal-tracking controller and a force-tracking controller. The goal-tracking controller tracks a given goal pose and is ideal for moving the food item to a desired position inside the user's mouth and moving the utensil outside their mouth as soon as a bite is detected. The force-tracking controller minimizes contact force on the F/T sensor at the end-effector and thus is very compliant and reactive. This is ideal for physical interactions such as impulsive and in-mouth manipulation, where the robot must quickly respond to forces applied to the utensil.

Details on data collection, model training, and controller implementation are provided in Appendix A.4.

4.3 Evaluation of System Components

4.3.1 Mouth Perception Evaluation

We compare the performance of the proposed mouth perception method against baselines for accurate 3D mouth keypoint generation, and perform an ablation study to evaluate the necessity of a key feature our mouth perception pipeline enables: continuous, real-time mouth perception for inside-mouth bite transfer.

Eval. 1: Comparison with Baselines - We compare the root mean square error (RMSE) of 3D mouth keypoints generated by our novel mouth perception method (FEMTO) against a pre-trained DECA using our proposed pipeline, and contemporary mouth perception methods in robot-assisted feeding [70, 244] (Table 4.1). For this evaluation, we use data from the 2 test participants in Section 4.2.1, introducing synthetic utensil occlusion to color and depth images. Ground truth keypoints are generated by applying our mouth perception pipeline with DECA to the unoccluded originals.

Table 4.1: FEMTO outperforms baselines adapted from robot-assisted feeding [70, 244] and head perception [62] literature.

Method	RMSE (in mm)
Gallenberger et al. 2019 [70]	110.454
Shaikewitz et al. 2023 [244]	82.946
DECA [62]	4.72
FEMTO (Ours)	**3.69**

Eval. 2: Necessity of Real-Time Mouth Perception - We perform an ablation study to evaluate the necessity of continuous, real-time mouth perception for the whole duration of inside-mouth bite transfer. This study examines three scenarios, selected based on feedback from individuals with mobility limitations, that impact their feeding process: (1) voluntary head movement, (2) voluntary speaking, and (3) involuntary head movement (spasm).

Methods. In each scenario, we compare two methods: our approach (RealTime), which employs continuous, real-time mouth perception, and the baseline (OneTime) [244] which estimates the user's mouth pose only once when they first open their mouth.

Procedure. We conducted the study with 15 participants (11 male, 4 female; ages 20-65) who had no mobility limitations. After participants provide

their demographics and complete pre-study questionnaires, they are seated and strapped into a wheelchair, and asked to simulate three scenarios (Figure 4.4):

S1. Voluntary Head Movement: Participants initially face person A, and open their mouth to initiate a bite. While the robot is moving towards them to feed, upon an audio cue, they turn to person B as if following a conversation.

S2. Voluntary Speaking: Participants initially face person C, and open their mouth to initiate a bite. While the robot is moving towards them to feed, upon an audio cue, they start reciting a preset script, as if conversing with person C.

S3. Involuntary Head Movement (Spasm): For this scenario, participants are first trained to mimic involuntary impulses, following guidance from an occupational therapist. During the trial, participants initially face person C, and open their mouth to initiate a bite. While the robot is moving towards them to feed, upon an audio cue, participants simulate a sudden forward impulse.

Post-feeding, participants fill out a Likert survey on perceived satisfaction, safety, and comfort [all scenarios], control and ease [S1], control and bite timing [S2], and robot responsiveness [S3]. They are fed a total of 24 times (3 scenarios $\times$ 2 methods $\times$ 2 utensils $\times$ 2 trials), with utensil and method orderings counterbalanced.

Study Results. RealTime consistently outperforms OneTime in all scenarios (Figure 4.4). In S1, OneTime often misses the target by aiming at the initial mouth position, while RealTime adapts to mouth movements for accurate feeding. In S2, OneTime disrupts speech by continuing movement, whereas RealTime is able to perceive the participant's mouth closing and pauses while they are speaking. In S3, OneTime maintains its movement towards the participant,

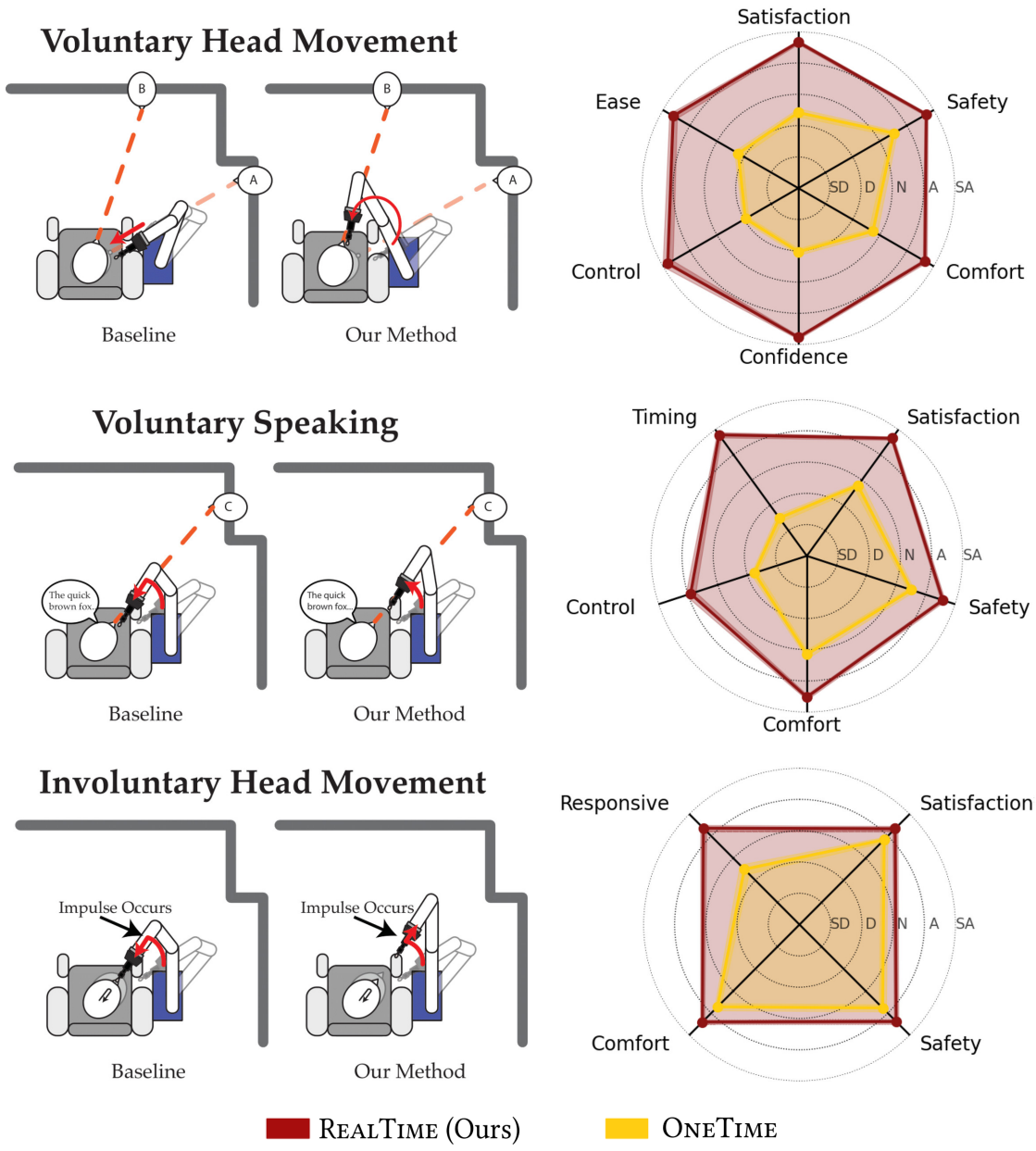


Figure 4.4: RealTime (Ours) enables the robot to track the mouth's pose and state, which enhances satisfaction, safety, comfort, confidence, control, ease of use, bite timing, and responsiveness across various scenarios in comparison to OneTime [244]. Response options used across studies: SD (Strongly Disagree), D (Disagree), N (Neutral), A (Agree), SA (Strongly Agree).

Table 4.2: Performance (F1 score) of physical interaction classification methods for inside-mouth bite transfer.

Method	All Participants Aggregated			Novel Participant		
	All	Haptic	Visual	All	Haptic	Visual
SVM [48]	**0.903**	**0.857**	**0.860**	**0.872**	0.834	0.649
MLP [97]	0.892	0.845	0.842	0.871	**0.842**	**0.712**
TCN [148]	0.887	0.787	0.634	0.862	0.784	0.545
TST [291]	0.902	0.831	0.822	0.856	0.819	0.689

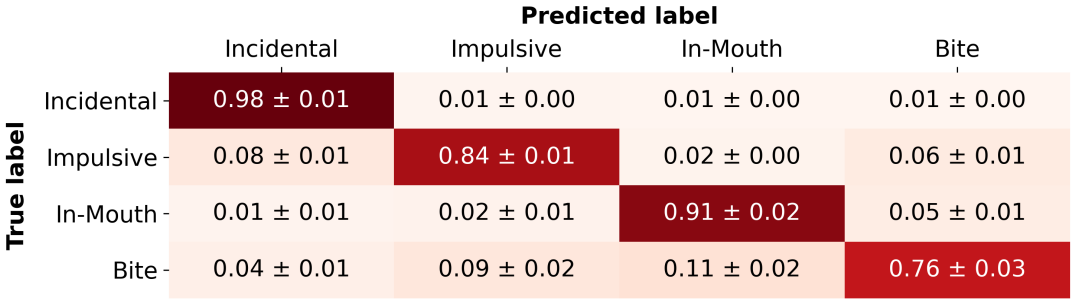


Figure 4.5: Confusion matrix showing multimodal SVM's performance for Condition B (Novel Participant).

even if they spasm. In contrast, RealTime immediately retracts upon detecting impulse, resuming feeding only when safe.

4.3.2 Physical Interaction-Aware Control Evaluation

We compare the performance of various time-series classification models for physical interaction classification, and perform an ablation study to evaluate the necessity of physical interaction-aware control for inside-mouth bite transfer.

Eval. 1: Comparison between Models - We use data collected with the 6 participants in Section 4.2.2, and consider two conditions:

A - Aggregated training and testing datasets from all 6 participants, maintaining an 80:20 split of each participant's data.

B - Train on 5 participants, and test on the 6th, novel participant in leave-one-participant-out cross validation fashion.

SVM exhibits superior performance across both conditions (Table 4.2 and Figure 4.5), which is likely attributable to the low data regime. For all models, combining haptic and visual modalities leads to superior performance compared to using either modality in isolation. This finding strongly advocates for the use of multimodal sensing for classifying physical interactions. The reduction in performance from condition A to B could stem from the unique traits of individual participants affecting physical interaction, aligning with findings from previous studies [45, 164].

Performance Improvement with Finetuning. We investigate finetuning as a solution to the challenge of unique participant traits impacting zero-shot performance of models on novel participants. We split the novel participant's data into an 80:20 ratio, allocating 80% for finetuning and 20% for testing. We evaluate the increase in performance of our best performing model, SVM, with incremental amounts of finetuning datapoints. Results (Figure 4.6) show steady performance improvement across modalities, with a significant performance improvement even with a small number of data points from the novel participant.

Eval. 2: Necessity of Physical Interaction-Aware Control - We perform an ablation study to assess the importance of physical interaction-aware control for inside-mouth bite transfers, especially for individuals with severe mobility limitations. We consider a scenario where an individual must have food placed on the right side of their mouth due to an inability to bite in the center. When caregivers place food in the center, the person guides them to the correct position using their tongue. Once shown, they expect the caregiver to remember this

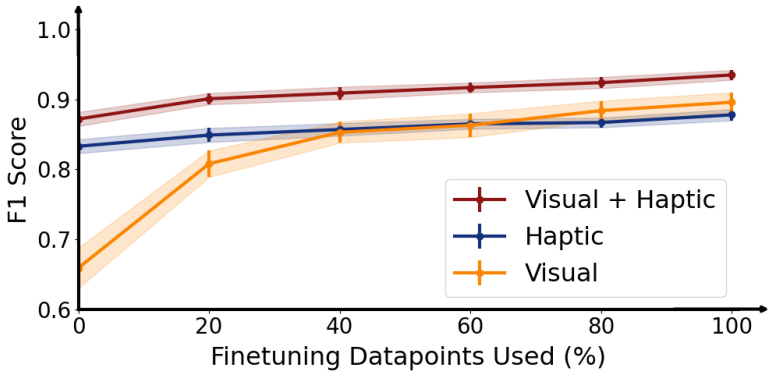


Figure 4.6: SVM's classification performance on novel participants improves steadily with finetuning using their data.

preference. This person also experiences involuntary head movements, which can occur even while they eat.

Methods. We manipulate one within-subject variable: the robot's level of physical-interaction awareness during feeding. We use four methods that vary in their ability to distinguish between different contact types, prompting the robot to switch to the appropriate controller accordingly. In order of decreasing awareness:

- Method A: Incidental vs. Impulsive vs. In-mouth Manipulation vs. Bite classification.
- Method B: Incidental vs. Inside Mouth Non-bite (combining impulsive and in-mouth manipulation) vs. Bite classification.
- Method C: Non-bite vs. Bite classification.
- Method D: Any Contact is Bite [244].

Methods have access to the ground truth contact type for this study; we also report offline classification accuracy.

Procedure. We conducted the study with 14 participants (8 males, 6 females; ages 21-27) who had no mobility limitations. After providing demographics and completing pre-study questionnaires, participants are seated and strapped into a wheelchair. They then undergo training, under an occupational therapist's guidance, to realistically simulate the actions of the illustrated care recipient. A trial in this study consists of two successive feedings using the same method, in the following sequence:

1st Feeding: This feeding focuses on in-mouth manipulation. The robot, initially stationary, begins moving when participants open their mouth. Once the robot moves to the center of their mouth, it says "push," prompting participants to move the food to the right side with their tongue while keeping their head still. When the food is correctly positioned, the robot signals "bite," and participants bite down and wait for the robot to withdraw.

2nd Feeding: This feeding focuses on reaction to impulsive physical interactions and remembering preferred bite transfer location. The robot, initially stationary, begins moving when participants open their mouth. Once inside the mouth, the robot says "impulse," prompting participants to simulate an impulse. After the movement, they keep their mouth open for five seconds. If the robot doesn't place the food on the right side of their mouth within that time, participants have to adjust it with their tongue. When the food is correctly positioned, the robot signals "bite," and participants bite down and remain still until the robot withdraws.

Post-feeding, participants complete a Likert survey assessing satisfaction, safety, and comfort for both feedings. Additionally, the survey evaluates ease of moving the utensil for 1st Feeding and whether the robot automatically moved

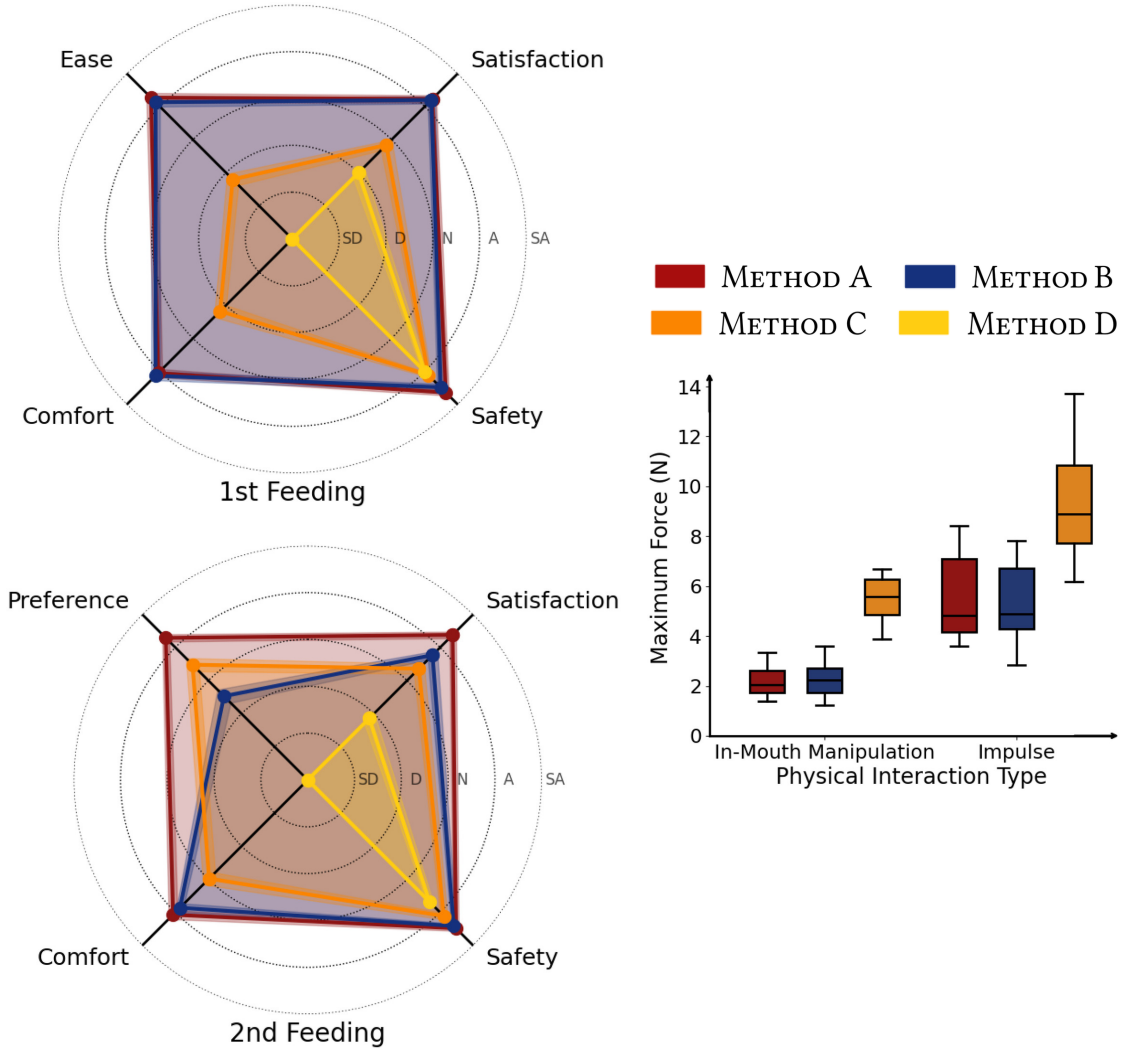


Figure 4.7: More physical-interaction aware control enhances perceived safety, satisfaction and comfort. This is supported by quantitative data recorded by the force sensor.

food to the preferred location for 2nd Feeding. Participants are fed a total of 32 times (2 feedings × 4 methods × 2 utensils × 2 trials), with utensil and method orderings counterbalanced.

Study Results. Figure 4.7 presents the study results. In the 1st Feeding, which focuses on in-mouth manipulation, participants predominantly prefer Method A and Method B. These methods switch to a force-tracking controller

upon detecting in-mouth manipulation, making them easier for participants to manipulate with their tongue compared to METHOD C, which constantly maintains a goal-tracking controller. This preference is quantitatively supported by the recorded force-torque data. In the 2nd Feeding, centered on reacting to impulsive physical interactions and remembering user preferences, participants rate METHOD A the highest. Both METHOD A and METHOD B transition to a force-tracking controller when impulsive contact is detected, and are perceived as comfortable by participants. However, participants note that METHOD A remembers their preferred transfer location, unlike METHOD B, which cannot distinguish between in-mouth manipulation and impulsive contacts. METHOD C, which continues with the goal-tracking controller during the impulsive motion, is rated as uncomfortable. The quantitative force-torque data show that METHOD C exerts significantly higher maximum force during an impulse as compared to METHOD A and METHOD B. As METHOD C does not respond to the impulsive contacts, it continues to move towards the preferred position even after the impulse. METHOD D [244] retracts from the mouth as soon as any contact is initiated during both 1st Feeding and 2nd Feeding. Consequently, it is ranked as the least satisfying method by the participants.

Classification Accuracy on Eval. 2 Study Data. For this analysis, we use our best-performing model, SVM, trained on 3072 physical interaction data points collected in Section 4.2.2, and test on interaction data from this Eval. 2 study. This evaluation tests the model's generalization capabilities to more unstructured study settings, compared to the controlled conditions of the prior data collection. SVM achieves an F1 score of 0.719 zero-shot (Table 4.3). In addition to distribution shifts caused by the unstructured study setting and novel participants, this performance drop could also be due to changes made to the

Table 4.3: SVM's performance (F1 Score) on Eval. 2 study data.

	All	Haptic	Visual
Zero-Shot	0.719	0.675	0.326
With Finetuning	0.772	0.697	0.605

robot system design between the initial data collection and this ablation study, based on end-user feedback. In line with improvements seen with finetuning in Eval. 1, the model's performance after further finetuning on 80% of each participant's data (approximately 10 data points per interaction type) and testing on the remaining 20% of their data, significantly improves to F1=0.772. These results underscore the importance of evaluating finetuning in physical interaction classification models as a means to address various sources of distribution shifts in real-world settings.

All study results are statistically significant (Wilcoxon Signed Rank Test, $p < 0.05$). Details on pilot studies, questions for each measure, and classification results are in Appendices A.5 and A.6.

4.4 Full System Evaluation with Care Recipients

We evaluate our full system through a user study with 13 individuals with severe mobility limitations, all of whom require assistance with feeding. The objectives of this study were:

O1. To evaluate the effectiveness and acceptance of our inside-mouth bite transfer system among its intended end users.

O2. Compare participant preferences between inside-mouth and outside-mouth bite transfer [18] systems, when both options are functionally possible for an individual.

Due to challenges in recruiting many individuals with mobility limitations at any one place, this study took place at three sites: EmPRISE Lab in Ithaca, NY, Columbia University Medical Center in NYC, NY, and a participant's home (Figure 4.8(b)) in Taftville, CT.

4.4.1 Methods and Procedures

We considered two systems in this user study: (1) our proposed inside-mouth bite transfer system, and (2) an outside-mouth bite transfer system adapted from Bhattacharjee et al. 2020 [18] that stops at a fixed distance (5 cm) from the user's mouth.

We informed participants to focus on the interaction from when the robot picks up a food item until they take a bite. Initially, we guided them through practice runs for each method. We used two utensils, a fork for cantaloupe and a spoon for yogurt. A trial involved three successive feedings using the same utensil and bite transfer system, after which we asked questions about their perceived safety and comfort. We evaluated the inside-mouth bite transfer system (O1) with all participants. We compared with the outside-mouth bite transfer system (O2) only with participants who demonstrated the required functional capability during practice runs. Order of the two systems and utensils were counterbalanced. After all trials, a final evaluation phase occurred where we fed participants using inside-mouth bite transfer and asked them to complete



(a)



(b)

Figure 4.8: Our system fed 13 care recipients with diverse mobility limitations across three sites. (a) Twelve of them, each labeled with their self-described impairment. (b) The thirteenth, fed in their home.

a TAM survey [53]. This was followed by a feedback interview to perceive the significance of the key components of our system and identify areas for improvement.

4.4.2 Participants

The study involved 13 participants with diverse ages, genders, self-described impairments, duration of impairments, and care providers (Table 4.4). Some participants had complex medical conditions, making feeding particularly challenging (Figure 4.9). For instance, P6 and P9 have Spinal Muscular Atrophy, a condition known to reduce maximum mouth opening and decrease bite force by almost 50% [84]. P3 has an overbite and a sensitive gag reflex, which can be triggered if a utensil presses on their tongue. For solid foods, they prefer the utensil to approach from the left side of their mouth, depositing the food on their left molars. P1, P2 and P7 experience spasms that can happen unexpectedly. P5 is diagnosed with Schizencephaly, which manifests with severe motor limitations (quadriplegia) and poor jaw and tongue control for eating. They have severe spasticity in their tongue, gag and bite reflexes, and an overbite.

All data collection and user studies in this chapter were approved by the Cornell University Institutional Review Board.

4.4.3 Findings

Our inside-mouth bite transfer system is perceived as safe and comfortable. Figure 4.10 displays the results. P7 mentioned, "... going in and out, I felt very

Table 4.4: Demographics of user study participants. All participants require assistance with feeding.

BiR - Bite Reflex, GaR - Gag Reflex, LR - Limited head/neck ROM, MaS - Masseter Spasticity, TS - Tongue Spasticity

ID	Age	Gender	Race	Self-described impairment	Impairment time	Challenges with Feeding
P1	44	Female	Caucasian/White	Multiple Sclerosis	25 years	LR, Spasms
P2	33	Female	African American	C3-C4 Spinal Cord Injury	13 years	LR, Spasms
P3	30	Male	Caucasian/White	Arthrogryposis	Since Birth	GaR, LR, MaS, TS
P4	45	Female	Caucasian/White	Multiple Sclerosis	15 years	LR
P5	26	Female	Hispanic	Schizencephaly Quadriplegia	Since Birth	BiR, GaR, LR, MaS, Overbite, TS
P6	27	Male	Caucasian/White	Spinal Muscular Atrophy	Since Birth	LR, MaS
P7	49	Female	Hispanic	C4-C5 Spinal Cord Injury	28 years	LR, Spasms
P8	44	Male	Caucasian/White	C4-C5 Spinal Cord Injury	24 years	LR
P9	39	Female	Asian/Pacific Islander	Spinal Muscular Atrophy	Since Birth	LR, MaS
P10	48	Male	African American	C5-C6 Spinal Cord Injury	2.5 years	LR, TS
P11	31	Female	African American	Cerebral Palsy Quadriplegia	Since Birth	LR, TS
P12	25	Male	Caucasian/White	C4-C5 Spinal Cord Injury	6 years	LR
P13	31	Female	African American	Arthrogryposis	Since Birth	LR, TS

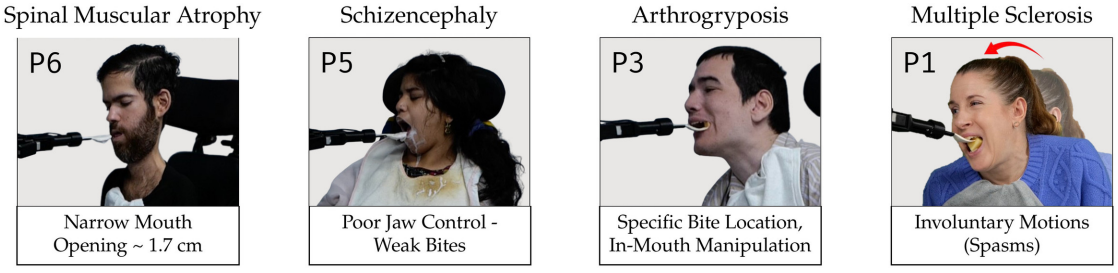


Figure 4.9: Inside-mouth bite transfer challenges, from left to right: P6 has limited mouth opening, P5's weak jaw control causes incidental contacts, P3 can bite only at left molars, and P1's muscle spasms move her towards the utensil. The label above each photo names the care recipient's self-described impairment, and the box below it describes the challenge.

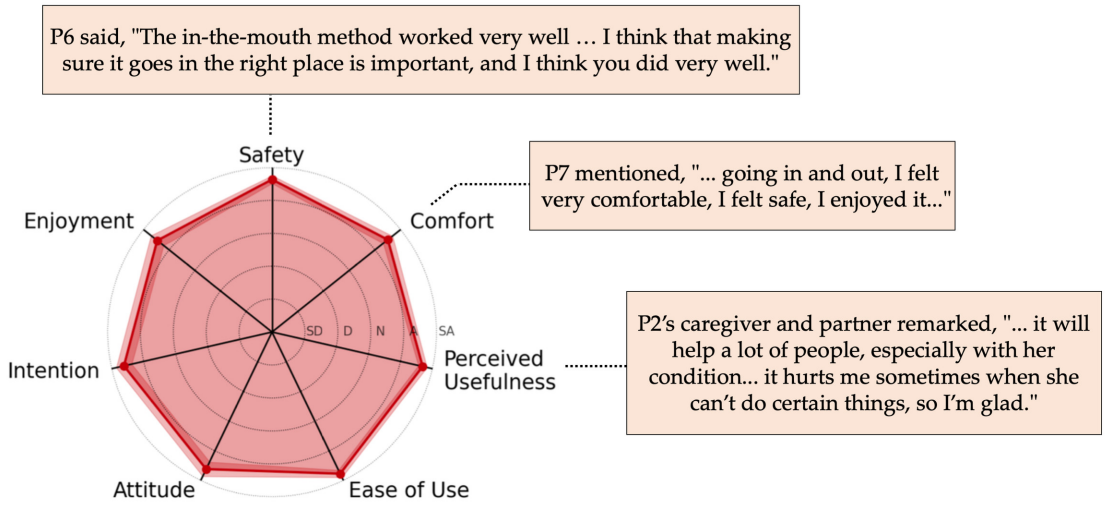


Figure 4.10: Our system received high ratings from care recipients for its safety and comfort, and was favorably evaluated in terms of technology acceptance, as indicated by the results of the TAM survey. The quotes are from care recipients and caregivers in the study.

comfortable, I felt safe, I enjoyed it..." P9 echoed, "... I think I was actually surprised... I liked it, the entrance in and out..." P5's parents were pleasantly surprised, "... I think it worked really much better than I expected. When she moved her mouth, it moved with her... She had fun with it!" P6 emphasized the importance of placement, "The in-the-mouth method worked very well. I was worried before doing it that it would go too deep, and it didn't. I think that making sure it goes in the right place is important, and I think you did very well."

Users view the technology favorably. As per Figure 4.10, users found the inside-mouth bite transfer system to be useful, easy to use, have a positive attitude and intend to use it, and enjoy the process. P2's caregiver remarked, "... it will help a lot of people, especially with her condition... it hurts me sometimes when she can't do certain things, so I'm glad." P5's parents highlighted its potential for social engagement, "[in a social setting,] the caregivers all interact,

and the kids are just passive recipients of food. So I think this will involve them more in the eating."

Even among the few users able to perform outside-mouth bite transfer, some preferred inside-mouth bite transfer. Out of the 13 users, 9 were unable to lean forward for outside-mouth bite transfer. Among the 4 users who evaluated both systems, some preferred inside-mouth as they felt that outside-mouth bite transfer requires more physical effort. P10 favored inside-mouth, saying it "requires no movement of the neck." P12 and P4's preferences varied based on their physical state. P12 stated, "If I was having neck pain, maybe I would prefer inside mouth so that I am not moving as much." P11 consistently preferred outside-mouth, noting, "I have the ability to position the food in my mouth myself, with my own comfort."

Users highlight that continuous, real-time mouth perception and adaptive compliance are essential. P12 noted, "I could be sitting in front of the TV watching something... I'm ready to take a bite and then it catches my attention, or yeah, talking to someone." P9 added another context: "When a phone rings, and I turn around..." P7 emphasized the importance of device adaptability during spontaneous movements, "On spasms. I mean, if it wiggles with you, that would be perfect. The body can move when there is a spasm, so it could go off a little and if it follows me, that's great."

4.5 Discussion

Our study demonstrates promising results, underlining the potential feeding systems have to help improve the quality of life of care recipients. However, the

wide range of disabilities and individual user needs makes crafting universal solutions complex. Participants with unique challenges, like limited mouth opening, stressed the importance of personalized adaptations. For instance, P6 and P9, who have smaller mouth openings, suggested specifically designed utensils, with P9 pointing out, "The spoon was too big." Similarly, opinions on transfer speed varied: P11 described the robot as "a little bit too fast," while P12 found it "too slow." These insights highlight the need for adaptive algorithms and interfaces that are tailored to individual needs, moving away from a one-size-fits-all model. Additionally, while our study focuses on short-term interactions, future work needs to explore long-term usability. Longitudinal studies will offer a deeper insight into the prolonged impact of this technology on users.

To deploy our inside-mouth bite transfer system, we combined it with bite acquisition from the Personal Robotics Lab at the University of Washington into a full feeding system, co-led with Ethan K. Gordon and Amal Nanavati [82]. We demonstrated this system at HRI 2024, where it received the Best Demo Award.

Part II

Deployment for a Few Sessions: Personalizing to Behavioral Preferences

CHAPTER 5

FLAIR: LONG-HORIZON BITE ACQUISITION

5.1 Introduction

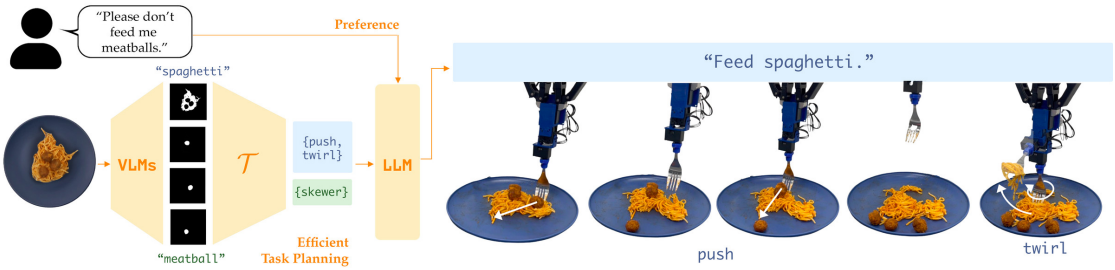


Figure 5.1: We propose FLAIR, a system for long-horizon robot-assisted feeding that combines the commonsense and few-shot reasoning capabilities of foundation models with a library of parameterized skills. Above, FLAIR takes visual observations and a given user preference ("Please don't feed me any meatballs") to plan a sequence of actions that pushes aside meatballs and twirls spaghetti.

Robot-assisted feeding entails first performing *bite acquisition* [61, 80, 81, 83, 85, 136, 255, 256], where the robot must manipulate a utensil to pick up a bite of food, followed by *bite transfer* [13, 70, 120, 214, 244], or bringing a bite of food to the mouth for consumption. In this chapter, we primarily focus on bite acquisition. Several prior works in bite acquisition develop individual skills targeting specific food groups. This includes policies for skewering firm foods [61, 80, 81, 83, 255], scooping soft foods [85, 259], or rearranging and twirling noodles [256]. These works, however, mostly operate over a single bite horizon or consider plates with a homogeneous type of food, such as only noodles or only bite-sized fruits and vegetables. However, the challenge of achieving reliable bite acquisition for dishes encountered *in-the-wild*, which contain multiple different food types within the same meal and require strategic skill sequencing over many timesteps, persists.

Consider a robot tasked with feeding a meal with a fruit appetizer—bananas, celery, and watermelon with chocolate sauce and ranch dressing—and spaghetti and meatballs for the main course. The robot must not only execute specialized strategies, such as cutting bananas, skewering fruits, dipping in sauces, and grouping and twirling noodles, but also infer how to sequence them over a *long horizon*, considering:

Efficiency: For the main course, if meatballs incidentally rest on top of the spaghetti, the robot should prioritize efficiency by serving the meatball first. This sequencing exposes the spaghetti for subsequent bites, avoiding the inefficiency of pushing the meatball aside to access the spaghetti initially.

User Preferences: However, if the user prefers to not eat meatballs, the robot must adjust the bite sequence accordingly.

Commonsense Reasoning: In the absence of explicit user preferences, the robot must employ commonsense reasoning to correctly order / combine bites for human-like feeding. For the appetizer, it should pair celery with ranch, bananas with chocolate, and feed watermelon standalone, reflecting typical food pairings. For the main course, it should vary the serving order between spaghetti and meatballs to avoid repetitions.

We desire a system that considers all these criteria to achieve long-horizon bite acquisition via a library of skills, and finally integrates with frameworks for bite transfer [13, 70, 120, 214, 244] to effectively feed complete meals.

In this work, we introduce FLAIR (Feeding via Long-horizon AcquIsition of Realistic dishes), a robot-assisted feeding system capable of feeding a complete meal to a care recipient (Figure 5.1). Given a plate image, and an optional user-provided natural language preference specifying their desired feeding strategy (i.e. 'I prefer to alternate bites of X and Y' or 'Don't feed me X'), FLAIR executes a

sequence of actions that efficiently feeds the items on the plate while adhering to the preference. The framework starts by detecting food items and their semantic labels (i.e. `'spaghetti'`) via Vision-Language Models (VLMs). We then pass the visual state estimate and semantic label for all items to a hierarchical task planner, which outputs per-item efficiencies by proxy of inferring a sequence of skills to achieve acquisition for each item. Finally, we pass all of this context – the food item labels, the optional user's preference, and per-item efficiencies – to a Large Language Model (LLM)-based planner which outputs the next bite to feed. The few-shot reasoning capabilities of LLMs allow for reasoning about the available context in a chain-of-thought manner, and planning sequences of bites that cater to both preference and efficiency. We carry out these action sequences via a library of parameterized food manipulation skills implemented on custom hardware. Finally, FLAIR's modular approach to long-horizon bite acquisition enables seamless integration with existing outside-mouth bite transfer [70] and inside-mouth bite transfer [120] (Chapter 4) frameworks.

We deploy FLAIR across two institutions and three robots: a Kinova 6-degree-of-freedom (DoF) arm at Cornell University and a Franka Emika Panda and a Kinova 7-DoF at Stanford University, demonstrating its adaptability to various robotic platforms. We validate FLAIR for long-horizon food pickup across six diverse plates, ranging from DoorDash orders and prepared grocery store meals to homemade meals. In a user study across 42 individuals without mobility limitations, we use FLAIR to demonstrate the necessity of balancing between both preferences and efficiency for feeding complete, realistic meals, as compared to an efficiency-only or preference-only approach. Moreover, we compare FLAIR's hierarchical task planner against three state-of-the-art baselines [1, 172, 256] on two different datasets, demonstrating that it significantly

outperforms these baselines. Finally, we demonstrate the real-world effectiveness of our system in feeding a care recipient with Multiple Sclerosis a meal consisting of various fruits and dips.

Overall, our contributions include:

- FLAIR: A system for long-horizon feeding which leverages foundation models to sequence a library of diverse skills towards in-the-wild long-horizon bite acquisition.
- Deployment of FLAIR across two institutions and three different robots, demonstrating its versatility.
- A user study with 42 individuals without mobility limitations across 6 diverse plates validating the effectiveness of considering both preferences and efficiency for feeding.
- Demonstration of the real-world efficacy of our system by feeding a care recipient with mobility limitations.

5.2 FLAIR: Feeding via Long-horizon Acquisition of Realistic dishes

In this section, we present FLAIR, a system for feeding complete meals which combines existing foundation models in a novel way towards personalized and efficient bite sequencing. We first give an overview of our custom system hardware, then outline our approach to long-horizon bite acquisition, and finally discuss integration of our method with existing bite transfer frameworks [70, 120] for feeding of in-the-wild dishes.

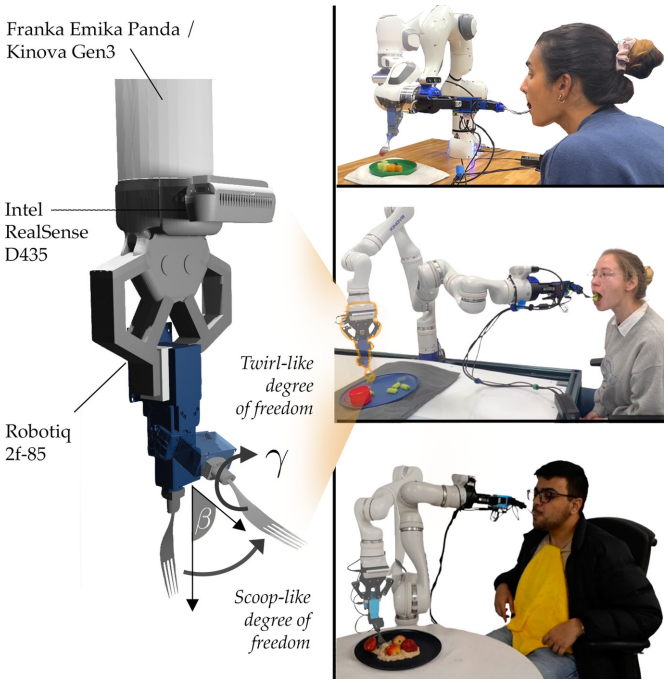


Figure 5.2: We implement our skill library using a custom feeding utensil (adapted from [244]) having two degrees of freedom for easy twirling and scooping at the end effector. We deploy the full feeding stack on three robots and two institutions: the 7-DoF Franka Emika Panda (top) and 7-DoF Kinova Gen 3 (middle) at Stanford University, and the 6-DoF Kinova Gen 3 (bottom) at Cornell University.

5.2.1 Hardware System

We tackle a wide range of food categories in this work such as fruits, vegetables, noodles, meat, soft foods, dipping sauces, and non-bite-sized items that require cutting. Many of these foods require specialized, dynamic manipulation strategies that typical 6 or 7-DoF robots struggle with due to their limited workspace. We thus implement FLAIR on Kinova and Franka robot arms equipped with a motorized feeding utensil mounted at the end-effector, adapting the design from [244]. The utensil contains a fork attachment and has two degrees of freedom corresponding to the orientation of the fork tines and the tilt angle. This allows for directly controlling the utensil to perform dynamic movements like

twirling and scooping, while the robot handles moving between waypoints in the workspace via Cartesian position control. We also use a wrist-mounted RGB-D RealSense camera with a known end-effector to camera transformation. This enables perceiving plates of food and localizing food items in the 3D workspace. We note that the same hardware was replicated on two different Kinova arms and one Franka Emika Panda, each with their separate fork attachment and sensors across two different institutions (detailed in Appendix B.1), demonstrating the reproducibility of our method and hardware (Fig. 5.2).

5.2.2 Long-Horizon Bite Acquisition Framework

With access to a hardware platform that supports dexterous food manipulation strategies, our goal is to plan and execute long-horizon bite sequences that cater to a user's preference while efficiently feeding a meal.

Problem Formulation. We assume access to an RGB-D plate image observation $o_t \in O = \mathbb{R}_+^{W \times H \times 4}$ of width W and height H, and an optional natural language instruction ℓ_{pref} from the user, representing their preferred feeding strategy at a high-level (i.e., ℓ_{pref} = "Feed me alternating bites of X and Y" or "Only feed me X"). X and Y can denote an arbitrary food item semantic label (i.e. "spaghetti", "strawberry", "caramel") or category (i.e. "noodles", "fruit", "sauce").

We further assume access to a library $\mathcal{L} = \{\phi^1, \ldots, \phi^N\}$ of N skills that the robot can use to manipulate food items. Each skill $\phi^i(p)$ represents a parameterized manipulation primitive that takes in parameters p and outputs low-level motor commands. We represent a low-level action at time t by $a_t = (x, y, z, \beta, \gamma, \psi)$, where (x, y, z) denotes the position of the feeding utensil tip, β and γ denote

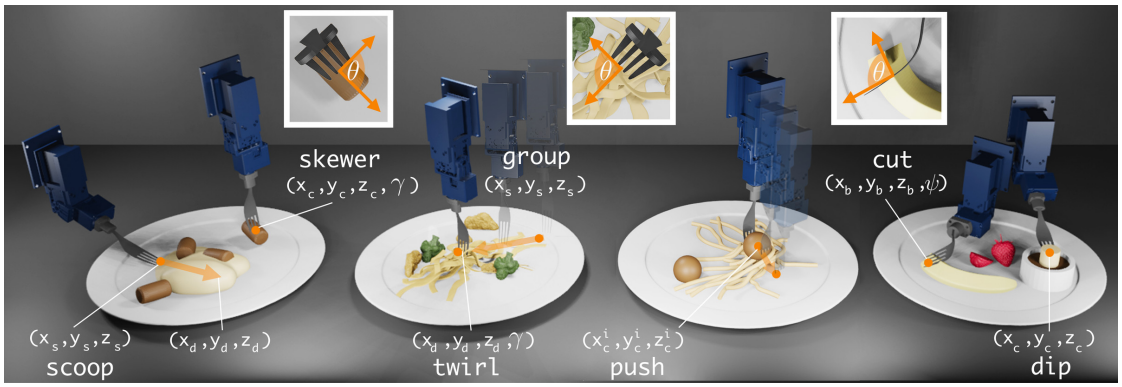


Figure 5.3: Our skill library consists of 7 parameterized manipulation skills: 4 acquisition (skewer, twirl, scoop, dip) and 3 pre-acquisition (group, push, cut).

pitch and roll of the utensil respectively, and ψ denotes the robot's end effector roll angle. Thus, the output of any skill is a sequence of T actions $\{a_t, a_{t+1}, \ldots, a_{t+T}\}$ that the robot takes to execute the particular strategy. For instance, a skewering skill may take the position and orientation of a desired food item as input, and output a trajectory that skewers the item of choice. Our goal is to plan and execute a sequence of parameterized skills $\{\phi_1(p_1), \phi_2(p_2), \ldots, \phi_H(p_H)\}$ which results in efficient and user-preferred bite acquisition, where H is the total number of skills to execute to complete feeding a plate and p_h refers to the parameters of skill $\phi_h \in \mathcal{L}$.

State Representations for Food. Our approach addresses the main challenges in long-horizon bite acquisition—parameterizing low-level skills and sequencing them—by integrating state-of-the-art visual-language models. We use visual state estimates and semantic features of food items to guide skill parameterization and sequencing.

For a given plate observation o_t at time t, we first query GPT-4V [1] in a few-shot manner to recognize which food items are present. We prompt the model

with a few in-context examples of plate images and their corresponding ground truth food item semantic labels, and ask the model to complete the prompt for the test image o_t. GPT-4V outputs a list of semantic labels l_t that are present (i.e., l_t = `['fettuccine', 'chicken', 'broccoli']`) along with their corresponding categories c_t (i.e., c_t = `['noodles','meat/seafood','vegetable', 'cuttable']`). These categories are relevant for associating the appropriate skill to each food item for bite acquisition. We then pass the recognized semantic labels to GroundingDINO [171], an open-vocabulary VLM, for bounding box detection. For each bounding box, we use SegmentAnything (SAM) [142] to refine these bounding boxes into segmentation masks $\{m_t^1, m_t^2, \ldots, m_t^D\}$ for all D items detected.

Skill Library. The segmented representations of food we obtain from VLMs provide a useful way to parameterize food manipulation skills, which we split into *acquisition* and *pre-acquisition* skills. Fig. 5.3 visualizes all skill parameterizations.

Acquisition skills

Acquisition skills refer to those that pick up food, such as skewering a food item, twirling a pile of noodles, scooping a soft pile of food, or dipping an item to coat it in sauce. We parameterize them as follows, assuming access to a segmentation mask m_t^i for the item of interest:

- `skewer`(x_c, y_c, z_c, γ): We detect the centroid of m_t^i and deproject this 2D pixel coordinate to a 3D coordinate (x_c, y_c, z_c) representing the center of a food item in the robot's frame of reference. We also estimate the major axis orientation θ of an item from m_t^i analytically. Following [61, 255], we

bring the utensil above the food item center with $\gamma = 90° + \theta$ and execute a swift downward trajectory skewering perpendicular to the main axis of the item. This encourages the tines of the fork to pierce the item. If the tines align parallel to the item's major axis, they may run along its longer length and miss the shorter breadth due to slight calibration challenges, leading to unsuccessful skewering.

- `twirl`(x_d, y_d, z_d, γ): We adopt the parameterization from VAPORS [256], a long-horizon system for noodle acquisition. Specifically, we twirl noodles by bringing the fork to the sensed *densest* pile (x_d, y_d, z_d) on the plate, estimated via 2D Gaussian filtering on m_t^i, and with γ identical to the parameterization for skewering (orthogonal to the major axis of the noodle pile sensed via a pose estimation network from [256]). We actuate the roll joint of the fork to complete two full twirls, wrapping noodles on the fork.
- `scoop`$(x_s, y_s, z_s, x_d, y_d, z_d)$: The fork starts with tines horizontal to the plate and scoops from the *sparsest* region (x_s, y_s, z_s) to the densest region (x_d, y_d, z_d) on the plate, up to a pre-defined maximum distance empirically selected to pick up a bite-sized amount. We define the sparsest region as the point on the boundary of the food item mask m_t^i that is furthest from the densest region, with the condition that the line connecting these points is not intersected by other food items, such as toppings.
- `dip`(x_c, y_c, z_c): Finally, dipping entails bringing a fork containing a food item into the center (x_c, y_c, z_c) of a small dish containing sauce. We initially orient the fork with tines horizontal to the plate to avoid the food item slipping off the utensil during dipping.

Immediately following each of these actions, the robot moves the fork tines in a scooping motion by actuating the utensil's pitch joint. The resulting horizontal fork helps prevent items from slipping off the fork after being picked up.

Pre-acquisition skills

When the above acquisition skills are not immediately feasible due to occlusion from other items or the anticipated amount of food to be picked up being insufficient, we employ a number of auxiliary strategies which we refer to as *pre-acquisition* skills. These actions do not directly pick up food but rearrange or manipulate items to facilitate subsequent acquisition. Grouping noodles into a pile before twirling, pushing a meatball off of a bed of spaghetti before twirling, or cutting banana into a bite-sized piece before pickup are all examples of pre-acquisition. We parameterize them as follows:

- `group`$(x_s, y_s, z_s, x_d, y_d, z_d)$: For a pile of food distributed on the plate, we sense the densest (x_d, y_d, z_d) and sparsest (x_s, y_s, z_s) regions via m_t^i and execute a linear push with the fork from the sparsest to densest point.
- `push`(x_c^i, y_c^i, z_c^i): For a food item with mask m_t^i obstructing a pile of food (such as noodles or a semisolid) with mask m_t^j, we can execute a linear push motion starting at the centroid of the obstructing item, to the nearest boundary point of the underlying food bed m_t^j.
- `cut`(x_b, y_b, z_b, ψ): To cut a food item, we estimate a point on the object (x_b, y_b, z_b) that would result in a bite-sized portion once cut. In practice, we detect the major axis of m_t^i and traverse a fixed unit length from the one end of the axis to estimate this. We then bring the fork horizontal ($\beta = 90°$)

and with sideways tines ($\gamma = 90°$). Finally, we set the end-effector roll ψ such that the lateral side of the fork is orthogonal to the major axis angle θ as in skewering and twirling. Then, we execute a swift downward trajectory to slice the soft item.

$\mathcal{L} = \{\texttt{skewer}, \texttt{twirl}, \texttt{scoop}, \texttt{dip}, \texttt{group}, \texttt{push}, \texttt{cut}\}$ forms the library of vision-parameterized skills at the core of FLAIR.

We provide further details on the vision-based parameterizations for each of these skills in Appendix B.3.

Task Planning for Acquisition. We plan a sequence of bites that both satisfies the preference of the user, and is efficient for the robot to acquire. The latter consideration requires reasoning over the sequence of pre-acquisition and acquisition skills needed to pick up an item, for which we introduce a hierarchical task planner $\mathcal{T}$. Our task planner relies on vision modules which post-process the segmented plate observations to quantify the density and spread of food items, along with checking for appropriate bite sizes and collision with other food items. While it uses a few key parameters, the overall pipeline and these parameters are shared across different categories like `'noodles'`, `'semisolid'`, and `'cuttable'`, rendering the approach versatile for diverse plates.

The task planner takes as input a particular food item category c_t^i along with the detected segmentation mask m_t^i and outputs a sequence of skills to acquire the item. The skill library in this work addresses the following categories of food items: {`'meat/seafood'`, `'fruit'`, `'vegetable'`, `'sauce'`, `'noodles'`, `'semisolid'`, `'cuttable'`}.

For most categories, acquisition tends to be immediately possible. Food items such as a bite of {`'meat/seafood'`, `'fruit'`, `'vegetable'`} tend to be isolated on a plate and immediately acquirable. Thus, we plan the following acquisition skills, where p_t^i denotes the parameters of the skill to manipulate the i-th food item, sensed from m_t^i and o_t:

- $\mathcal{T}(c_t^i, m_t^i) = \{\texttt{skewer}(p_t^i)\}$ for $c_t^i \in \{$`'meat/seafood'`, `'fruit'`, `'vegetable'`$\}$
- $\mathcal{T}($`'sauce'`$, m_t^i) = \{\texttt{dip}(p_t^i)\}$

Food items that are instead in the `'noodles'`, `'semisolid'`, or `'cuttable'` category require more nuanced reasoning about pre-acquisition depending on the distribution of the food on the plate, and whether other food items are intermixed, on top, or to the side. We critically observe that the segmentation mask m_t^i obtained from the VLM provides a useful prior over the spread of food on the plate, which can guide action selection. We apply a Gaussian smoothing kernel over m_t^i which has the effect of producing a normalized density heatmap of the food, and use simple pre-conditions to determine a sequence of skills to pick up a bite of noodles or a semisolid. Specifically, we measure the maximum *density* and the 2D *entropy* of the heatmap and plan actions as follows.

If the density exceeds a pre-defined threshold `DENSITY_THRESH`, this indicates the presence of a large pile of food that can be immediately acquired:

- $\mathcal{T}($`'noodles'`$, m_t^i) = \{\texttt{twirl}(p_t^i)\}$
- $\mathcal{T}($`'semisolid'`$, m_t^i) = \{\texttt{scoop}(p_t^i)\}$.

However, when twirling is obstructed by another item mask m_t^j, such as a meatball too close to a planned spaghetti twirling action, the obstructing food

must be pushed aside first. Similarly, if toppings such as sausages block all viable scooping actions for mashed potatoes, the sausage nearest to the boundary of the mashed potato mask should be pushed.

- $\mathcal{T}(\texttt{`noodles'}, m_t^i) = \{\texttt{push}(p^i_{t,\texttt{push}}), \texttt{twirl}(p^i_{t,\texttt{twirl}})\}$ if the robot must push aside a topping before twirling.
- $\mathcal{T}(\texttt{`semisolid'}, m_t^i) = \{\texttt{push}(p^i_{t,\texttt{push}}), \texttt{scoop}(p^i_{t,\texttt{scoop}})\}$ if the robot must push aside a topping before scooping.

For food items in the `noodles` category, we also consider grouping actions. If the entropy exceeds a pre-defined threshold ENTROPY_THRESH, indicating that the food item is spread out on the plate, grouping can be helpful. We can directly execute unobstructed grouping actions. However, if all viable grouping actions are blocked by toppings, we instead push the topping closest to the boundary of the underlying food aside, group the food, and then acquire.

- $\mathcal{T}(\texttt{`noodles'}, m_t^i) = \{\texttt{group}(p^i_{t,\texttt{group}}), \texttt{twirl}(p^i_{t,\texttt{twirl}})\}$ if grouping is unobstructed.
- $\mathcal{T}(\texttt{`noodles'}, m_t^i) = \{\texttt{push}(p^i_{t,\texttt{push}}), \texttt{group}(p^i_{t,\texttt{group}}), \texttt{twirl}(p^i_{t,\texttt{twirl}})\}$ if the robot must push aside an obstructing topping before grouping.

If neither acquisition nor grouping is feasible according to the set thresholds, we push the topping within the food item mask nearest to its boundary, hoping to expose more of the food item for future acquisition. If no such topping is available, we default to acquisition.

Finally, for `cuttable` items like cake, we use a max major axis length threshold to determine if m_t^i is bite-sized or not, and either cut and then skewer, or skewer immediately:

- $\mathcal{T}(\texttt{`cuttable'}, m_t^i) = \{\texttt{cut}(p^i_{t,\texttt{cut}}), \texttt{skewer}(p^i_{t,\texttt{skewer}})\}$, if the length of major axis of $m_t^i$ exceeds `BITE_LENGTH`,
- $\mathcal{T}(\texttt{`cuttable'}, m_t^i) = \{\texttt{skewer}(p_t^i)\}$, otherwise.

We provide further details on task planning for acquisition in Appendix B.4.

5.2.3 Bite Sequencing via Foundation Models

We introduce a unified framework for planning and executing bite sequences that are efficient and adhere to user preferences. With access to a library of skills $\mathcal{L}$, task planner $\mathcal{T}$, and user preference ℓ_{pref}, we show how the commonsense-reasoning capabilities of LLMs enable them to act as few-shot planners for bite sequencing, inherently balancing preference and efficiency.

We prompt an LLM, in our case GPT-4V, with relevant context about the meal. This includes the semantic food item labels l_t and the user's preference ℓ_{pref}. We augment this context with: (i) a *history* of bites taken so far, (ii) an estimate of the *portions* of each food type remaining, and (iii) the *per-item efficiencies* which correspond to the number of actions required to pick up a food item (i.e. $|\mathcal{T}(c_t^i, m_t^i)|$). A higher number indicates a less efficient bite option since more pre-acquisition skills are required to pick up the item. To estimate quantities, we simply count the number of instances detected of the food item present if the item category is [`'fruit'`, `'meat/seafood'`, `'vegetable'`]. For `'cuttable'` food items, we use the major axis length of m_t^i divided by `BITE_LENGTH`. Otherwise for food items where the ground truth quantity is not countable, such as for `'noodles'` or `'semisolid'`, we use $\lceil |m_t^i| / \texttt{PORTION_SIZE} \rceil$, where `PORTION_SIZE`

is empirically determined for approximating the number of bite portions in a mask.

Below, we show an example input prompt in gray with the immediate next bite planned by asking the LLM to perform completion. We abridge the prompt here for brevity, but include our full prompting strategy in Appendix B.2.

```
Items remaining: ["fettuccine", "chicken", "broccoli"]
Preference: "Alternating bites of each"
History: ["chicken"]
Portions remaining: [5, 1, 2]
Efficiencies: [3, 1, 1]
---
Decide what bite to feed me next.
Format your response as follows:
Strategy: Sentence describing your high-level strategy
Next bite: Phrase describing the next bite you will feed
Next bite as list: ['item'] # Or ['item', 'dip'] or []
---
Output your response here.
```

```
Strategy: Given that you want to alternate amongst the
three types of items present, and you just ate chicken,
I will feed you either fettucine or broccoli.
Even though broccoli is a more efficient option, with only
1 action required, I will feed you fettuccine since
there are more portions of it.
Next bite: Feed fettuccine.
Next bite as list: ['fettuccine']
```

By reasoning about the provided context in a chain-of-thought style, the LLM generates a subsequent bite. Importantly, there are no explicit tradeoffs between efficiency, portion size, or preference which we embed into the prompt, allowing the LLM to reason about the most sensible strategy based on the intensity of the user's preference and available plate context. Given a next bite, in this case `'fettuccine'`, we then plan the appropriate skill sequence via $\mathcal{T}$(`'noodles'`, $m_t^{\texttt{'fettuccine'}}$) and execute it via $\mathcal{L}$.

5.2.4 Integration of Acquisition and Transfer

The self-contained nature of our *bite acquisition* framework allows for straightforward integration with *bite transfer* frameworks, and is agnostic to the exact approach used. We demonstrate this easy combination with two existing methods: (i) an outside-mouth bite transfer method [70] that features visual servoing capabilities, and (ii) a recent method for inside-mouth transfer [120] that leverages robust mouth tracking and physical interaction-aware control.

A significant challenge in this integration is ensuring that food, particularly semi-solid items such as mashed potatoes or noodle-like items such as spaghetti, does not spill while it moves from above the plate to the pre-transfer pose in front of the mouth. Prior works with non-actuated utensils [214] use an approach based on model predictive control to generate robot trajectories that constrain the orientation of the utensil to remain upright. However, these methods often require complex tuning and can be prone to getting trapped in local minima. Our feeding utensil (Fig. 5.2) enables us to uniquely circumvent this challenge. We leverage its roll (γ) and pitch (β) degrees of freedom, distinct from the robot's own degrees of freedom, to consistently keep the fork's tines horizontal regardless of the robot's motion. We continuously monitor the robot's end-effector pose at 10 Hz and adjust the feeding utensil's joints accordingly, ensuring a smooth and spill-free transfer of food to the user's mouth.

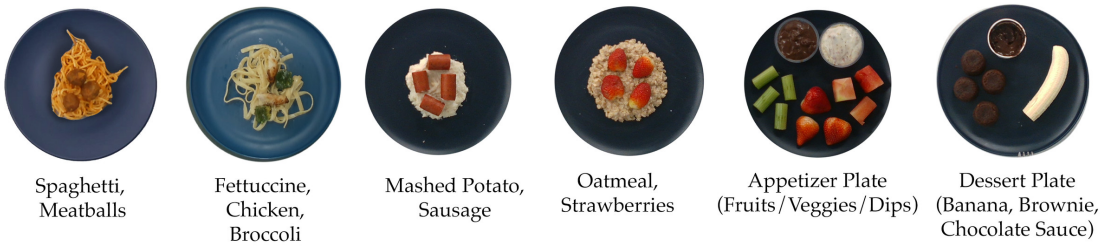


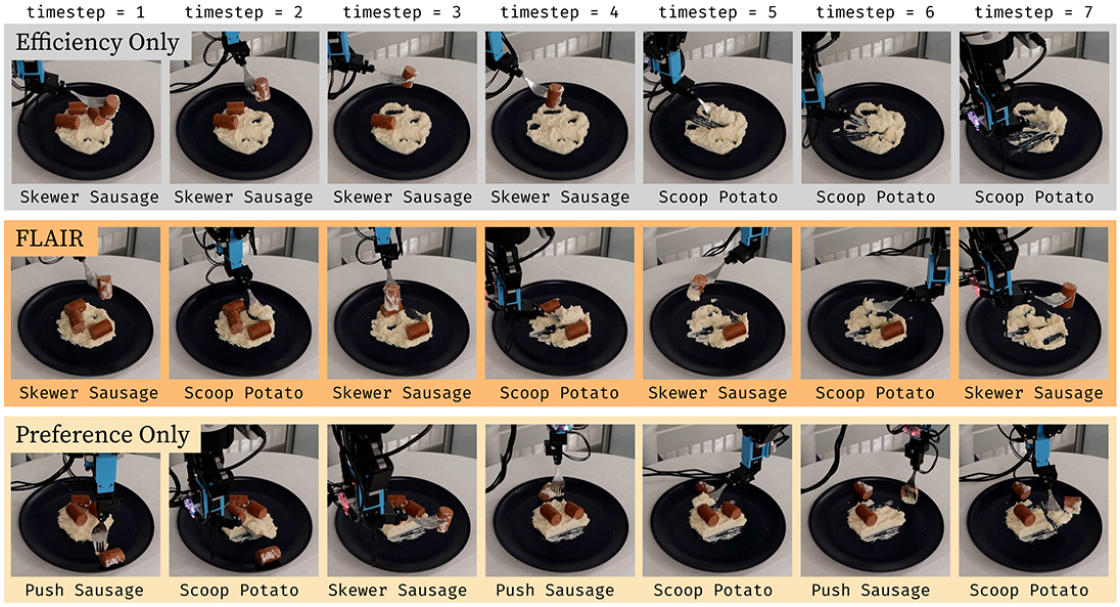
Figure 5.4: **Plates**: We evaluate our system on the following six plates containing a variety of food items, each necessitating highly different manipulation skills.

5.3 Experiments

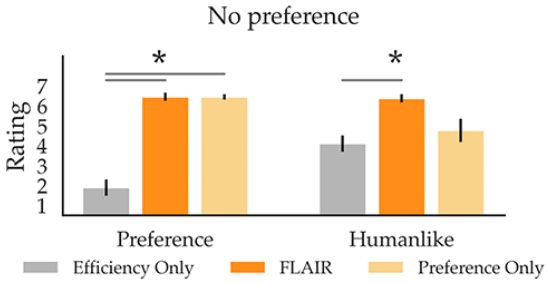
We evaluate the effectiveness of FLAIR for feeding diverse plates each containing various types of food items. We first conduct a user study to assess FLAIR's ability to perform long-horizon bite acquisition of in-the-wild plates, while adhering to user preferences and efficiently feeding bites. For all acquisition experiments, we interchangeably use 2 Kinova Gen3 arms (one 6-DoF, and another 7-DoF), and a 7-DoF Franka Emika Panda. We then ablate our hierarchical task planner $\mathcal{T}$ against various state-of-the-art baselines [1, 172, 256]. Finally, we evaluate the real-world efficacy of our system for feeding a complete plate to a care recipient with mobility limitations.

5.3.1 Bite Acquisition Experiments

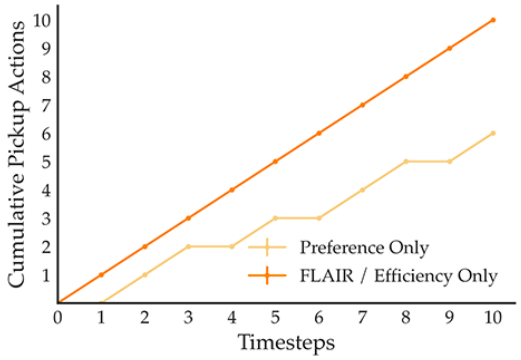
Baselines: FLAIR presents a unique approach of taking into account both preference and efficiency considerations for bite sequencing. This raises the question of how an *Efficiency-Only* or *Preference-Only* approach would compare. We implement an Efficiency-Only baseline which greedily selects the next bite as the item which requires the least number of pre-acquisition and acquisition skills



(a)



(b)



(c)

Figure 5.5: Example run on a plate with mashed potatoes and sausages where the user specified no preference. FLAIR, which balances user preferences (bite variety) and efficiency, is judged by users to better adhere to preferences than Efficiency-Only and outperforms Preference-Only (Commonsense-Only) in plate clearance. Consequently, FLAIR is considered to provide a more human-like feeding experience compared to the baseline methods. Note that * indicates statistical significance (p-value < 0.05), determined via a Mann-Whitney U test.

for pickup in the current instant, as dictated by the task planner $|\mathcal{T}(\cdot,\cdot)|$. The Preference-Only baseline is identical to FLAIR in implementation, but notably omits efficiency scores when prompting the LLM to generate a next bite. This encourages the LLM to only respect a user's preference without consideration for how efficient a particular bite may be. In the case that a user has no preference for feeding, we refer to the Preference-Only baseline as *Commonsense-Only.*

Evaluation Plates: We consider an evaluation suite of 6 diverse plates of food spanning a wide range of food categories, visualized in Fig. 5.4. We include 2 in-the-wild noodle dishes: a spaghetti and meatballs plate which is a prepared frozen meal from a grocery store, and a fettuccine alfredo dish with chicken and broccoli ordered from Applebee's via DoorDash. We also consider 2 homemade semisolid dishes: mashed potatoes with sausage, and oatmeal with strawberries. Lastly, we evaluate an appetizer plate of strawberry, watermelon, celery, ranch, and chocolate dipping sauce, as well as a dessert plate of a whole banana, brownie bites, and chocolate dipping sauce.

User Study Design: We evaluate FLAIR's ability to cater to user preferences via a two-phase user study across 42 individuals without mobility limitations (Ages: 19-64, Genders: 22F, 20M). In the first phase, we present participants with a survey showing images of all 6 evaluation plates, and solicit their natural language preference over how they would prefer to be fed each plate. In the survey, we specify the capabilities of our skill library to the participants of our user study, and ask them to note preferences over their preferred order of bites, or pairings of food items with sauces. Details on the reported user preferences are provided in Appendix B.5. Since evaluating each submitted preference across all of the plates and baselines is not scalable, we cluster the submitted preferences into common

shared responses via LLM summarization (GPT-4V). We focus on cases where users have either no preference or strong preferences, as slight preferences are not informative for comparing method behaviors. Thus, we specifically prompt GPT to filter for strong preferences (i.e. 'Always feed me alternating bites of X and Y' or 'Please do not feed me X') and group them accordingly. For each of the six plates, we then evaluate our system on the 2 most popular strong preferences summarized per plate, as well as an 'I have no preference' setting for completeness.

We hypothesized the following:

- **H1**: Compared to the Preference-Only baseline, FLAIR's consideration of efficiency in bite sequencing will lead to more bites across all settings.
- **H2**: Compared to the Efficiency-Only baseline, FLAIR's consideration of user preferences in bite sequencing in presence of strong preferences will lead to more perceived adherence to preferences, and more human-like feeding, rated based on the statement "This method is similar to the strategy I would use to feed myself."
- **H3**: Compared to the Efficiency-Only baseline, FLAIR's consideration of commonsense reasoning in bite sequencing in the absence of preferences will lead to more perceived adherence to bite variety and common food item pairings, and more human-like feeding.

This user study was approved by the Institutional Review Boards of both Cornell University and Stanford University.

Food Pickup Results: Fig. 5.7 displays the results of food pickup over time across methods in the no preference scenario. We provide additional results for food

pickup efficiency for all methods averaged across all plates (both strong preferences and no preferences) in Appendix B.6.1, noting a similar trend. Due to its consideration of efficiency in bite sequencing, FLAIR executes a greater number of pickup skills compared to Preference-Only, validating **H1**. This is because when faced with multiple valid candidate bites, FLAIR, informed with efficiency scores for each bite, is able to choose the bite that optimizes for efficiency. In contrast, Preference-Only randomly selects one bite from this set, often leading to inefficient acquisition trajectories (Fig. 5.5). The efficiency disparity between Efficiency-Only and FLAIR can be linked to settings where bite variety or strong preferences require the robot to pick up bites that are less efficient than those of a method which does not take such preferences into account. For instance, in a scenario where a robot is instructed to feed a bed of spaghetti hidden beneath multiple meatballs, methods that consider preferences must undertake multiple pre-acquisition skills to push away the meatballs.

User Evaluation: Fig. 5.6 presents average participant ratings comparing FLAIR with baseline approaches for settings with strong user preferences. We conduct a Mann-Whitney U test for statistical significance, and indicate pairs of methods for which the average participant ratings were significant (p-value < 0.05). This is a non-parametric test compatible with ordinal Likert data, and without specific assumptions on the normality or variance of the data distributions. By integrating user preferences into task planning, FLAIR substantially surpasses the Efficiency-Only baseline in terms of adherence to user preferences and human-like feeding across various settings, as hypothesized (**H2**). The exceptions, where the performance difference between Efficiency-Only and FLAIR is not significant, occur in settings where the bite sequence, generated based solely

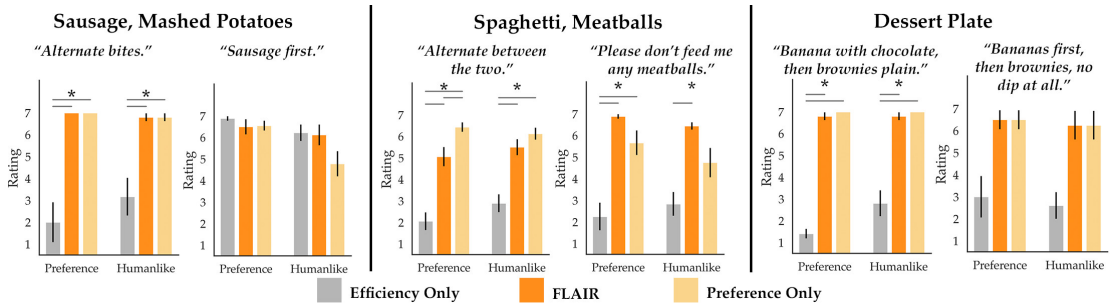


Figure 5.6: Average participant ratings for settings with strong user preferences show FLAIR significantly outperforms Efficiency-Only baseline in aligning with user preferences and achieving human-like feeding in all scenarios, except cases where the efficiency-based bite sequence coincidentally aligns with user preferences. Note that * indicates statistical significance (p-value $<$ 0.05), determined via a Mann-Whitney U test.

on efficiency, inadvertently matches the user's preferences. Fig. 5.7 shows participant ratings that compare FLAIR with baseline approaches in settings where no user preferences were specified. By leveraging commonsense reasoning, FLAIR significantly outperforms the Efficiency-Only baseline across most plates by ensuring bite variety and appropriately pairing food items with dips, resulting in a more human-like feeding experience (**H3**).

5.3.2 Comparisons with Task Planning Baselines

For evaluating necessity of pre-acquisition actions, FLAIR first estimates the distribution of food items by sensing density and entropy metrics from segmented observations, and then uses a hierarchical decision-tree style approach. In this section, we compare this approach against other task planning baselines. The closest relevant work, VAPORS [256], concentrates on noodle dishes and employs physics-based simulations for decision-making between twirling and grouping noodles. Our work, however, encompasses a broader spectrum of food

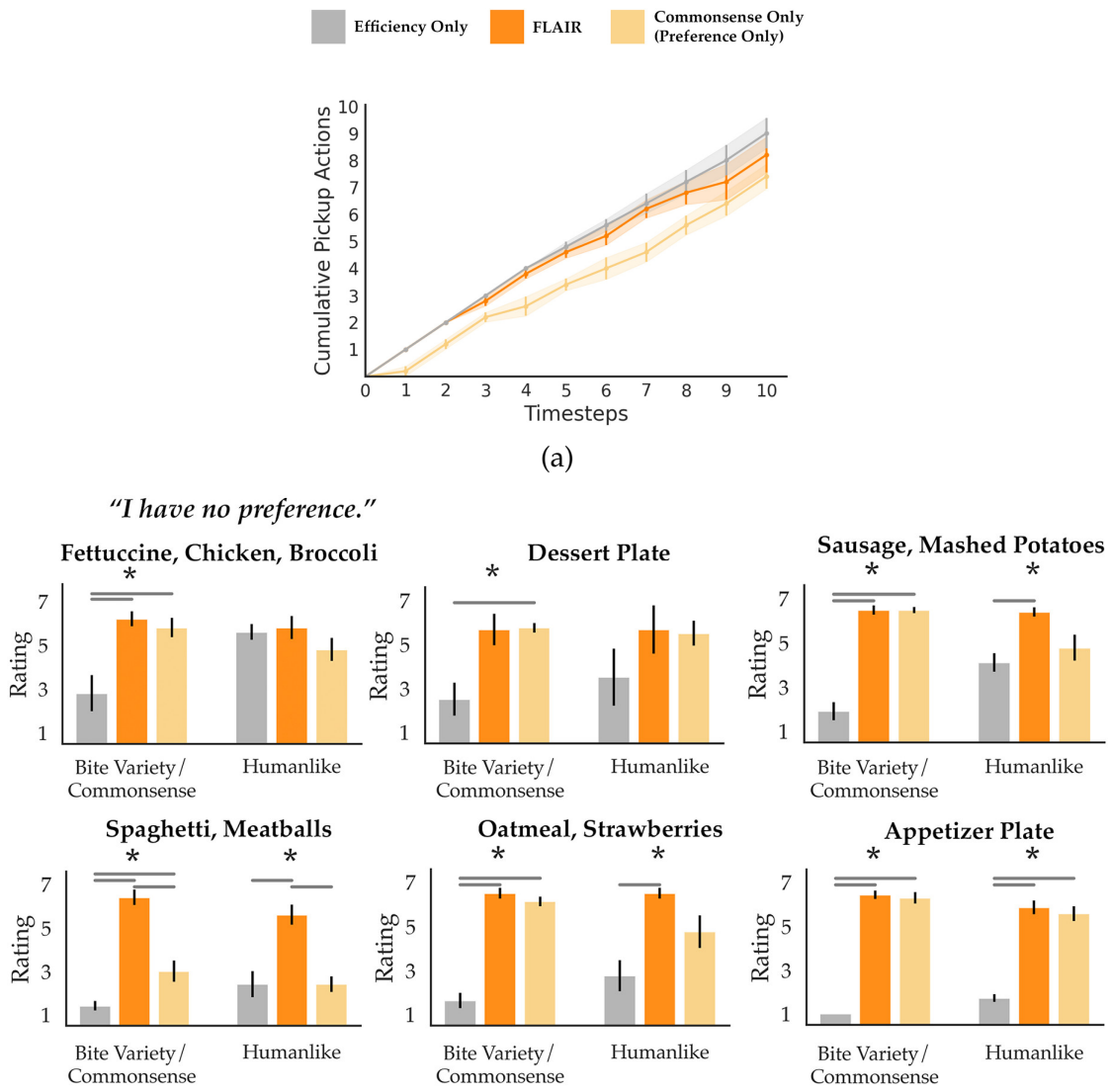


Figure 5.7: (a) FLAIR picks up more bites than Preference-Only (Commonsense-Only) accumulated across all plates for no preference scenarios. (b) For most no preference scenarios, average participant ratings show FLAIR significantly outperforms the Efficiency-Only baseline in aligning with user preferences (bite variety) and achieving human-like feeding. Note that * indicates statistical significance (p-value < 0.05), determined via a Mann-Whitney U test.

Table 5.1: Comparison of FLAIR's task planner with baselines

	User Study Plates	Noodle-Only Plates
FLAIR	**0.817**	**0.854**
VAPORS [256]	-	0.415
VLM-TaskPlanner [1]	0.518	0.683
Swin-Transformer [172]	0.720	0.785

textures and types (such as solids like meatballs, semi-solids like mashed potatoes, and noodle-like items like spaghetti), making direct adaptation of VAPORS challenging due to the complex physics simulations required for accurately representing their varied interactions. Taking this gap into account, we compare FLAIR's task planning accuracy against 3 established baselines: (i) VAPORS, (ii) VLM-TaskPlanner, which queries a VLM (GPT-4V [1]) using 10 in-context examples from the training set to decide between candidate actions, and (iii) image classification using a pre-trained Swin-Transformer [172] fine-tuned on the training set. We leverage 2 datasets of plate images with next action ground truth labels as a test-bed:

Evaluation on noodle-only plates from VAPORS ([256]). We use the $\sim$ 100 held-out noodle-only image dataset from VAPORS and have two third-party human annotators label each image with "Twirl" or "Group," corresponding to the action they deem appropriate. We extract the images with intercoder agreement and evenly split them into training and test sets. We compare our task planning approach (FLAIR), against the three baselines on this set.

Evaluation on logged user study plates. We further compare FLAIR to baselines on the $\sim$ 100 images per plate logged from our user study. Each image is

labeled by two third-party human annotators with appropriate ground truth labels for food items where pre-acquisition is relevant: twirl/group/push for noodles, push/scoop for semisolids (oatmeal/mashed potatoes), and cut/acquire for a whole banana. We extract the images with intercoder agreement, split them evenly into training and test sets, and report the mean accuracy across plates.

FLAIR uses identical parameters for all plates, whereas other baselines use plate-specific parameters inferred from their respective training data. Table 5.1 shows the results. FLAIR significantly outperforms all other baselines on both datasets: the noodle-only and user study plates. We posit that VAPORS may suffer due to the sim-to-real gap present in real vs. simulated observations, and the black-box VLM-TaskPlanner struggles without reasoning in a hierarchical manner. The Swin-Transformer classifier is the most competitive baseline, but likely suffers due to a lack of large-scale training data.

5.3.3 Demonstration of Real-World Feeding

We demonstrate FLAIR's effectiveness in helping a care recipient with severe mobility restrictions eat a plate of celery, watermelon, and strawberries with ranch dressing and chocolate sauce. The care recipient, a 44-year-old Caucasian/White female with Multiple Sclerosis, has a severely limited range of motion in their head and neck. Consequently, they require inside-mouth transfer [120] (Chapter 4) of acquired bites for successful feeding.

In the pre-study questionnaire, the care recipient mentioned that they typically have a preferred order in which they like to eat their meal. They convey this preference to their caregivers through natural language, and when caregivers adhere to this preference, it "definitely enhances my eating experiences." For

the considered plate, the care recipient specified "I want to first finish all the celery with ranch dressing, then eat watermelon without any dips, and finally end with strawberries dipped in chocolate sauce." Adhering to this preference, FLAIR begins by skewering celery, dipping them in ranch dressing, and transferring them inside the mouth of the user. Once it detects there is no more celery on the plate, it switches to skewering watermelons and feeding them without dips as instructed. Finally, it skewers the strawberries, dips them in chocolate sauce, and feeds them to complete the meal (Fig. 5.8). Following successful feeding, we posed two questions on a seven-point Likert scale on the necessity of a robot-assisted feeding system to (i) have a diverse bite acquisition skill library, and (ii) adherence to meal preferences, for acceptance for day to day usage. The care recipient strongly agreed (rating = 7) with both, emphasizing the core contributions of this chapter as critical aspects for an in-the-wild feeding system.

In the post-study questionnaire, the care recipient noted that while they often have specific preferences for the order in which they eat their food, they often refrain from sharing these preferences with their human caregivers. They expressed concern that such requests might impose additional burdens on caregivers who are already assisting them with feeding. However, they were hopeful that a robot designed to assist with feeding could accommodate their preferences seamlessly, thus enhancing their mealtime experience by respecting their autonomy and enabling them to better enjoy their meals.

5.4 Discussion

FLAIR is a first step towards robot-assisted feeding in real-world scenarios, adeptly handling various in-the-wild meals composed of diverse food items.



Figure 5.8: We demonstrate the real-world effectiveness of our method by feeding a plate of fruits and vegetables to a care recipient with severe mobility limitations.

We deploy FLAIR across 2 institutions and 3 different embodiments with a library of 7 dexterous skills. Our evaluations include both bite acquisition and bite transfer, along with a demonstration feeding a complete plate to a care recipient. FLAIR showcases the ability to abide by preferences across 42 individuals and a range of diverse plates, without compromising on efficient food pickup. Through our extensive evaluations, we identify the following limitations to guide future work in robot-assisted feeding.

Limitations of Food Perception using VLMs. While current VLMs are capable of identifying food items on a plate, using these generated identifiers with open-set object detectors can lead to inaccuracies. FLAIR addresses this challenge by enriching the identifiers with a set of hand-coded descriptors tailored to the typical type of the food item, e.g., specifying `'banana'` as [`'banana piece'`, `'sliced banana'`] and `'fettuccine'` as [`'fettuccine pasta'`, `'fettuccine noodles'`]. In the future, advancements in open-set object detection may make such specific enhancements unnecessary.

Limitations of Food Manipulation Skills. FLAIR leverages a library of skills inspired by state-of-the-art food manipulation methods, but open challenges include: slippage during skewering, failing to twirl noodles or scoop mashed potatoes into reasonable bite sizes, failing to cut tough items, and errors due to perception (erroneous depth sensing or imprecise food detection) which can cause manipulation imprecision. Although some of these failures can be addressed by re-trying (as long as the item is re-detected), these challenges can be mitigated in the future by making the skills themselves reactive, enabling adaptive utensil trajectories that adjust to food slippage or deformation on the fly.

Limitations of Bite Sequencing. We harness LLMs to plan efficient bite sequences that adhere to user preferences. However, today's language models can sometimes generate unrealistic or irrelevant outputs ("hallucinations"). In FLAIR, we reduce hallucinated artifacts in bite sequencing by using prompt-engineering strategies which we detail in Appendix B.2. However, even with templated prompts, FLAIR is still limited by the tendency of language models to occasionally neglect context, such as the manipulation efficiency of food items and their remaining portions. In the future, we are excited by structured prompting strategies [56] and incorporation of real-time corrections from the user [245] to address these challenges.

Although these are current limitations, **FLAIR's modular system design allows for easy interchange of the perception/planning stacks or even skills themselves.** Thus, it will be able to take full advantage of future advances in VLMs or better low-level skill policies that are learned or engineered.

CHAPTER 6

FEAST: IN-THE-WILD PERSONALIZATION

6.1 Introduction

Recent advancements have significantly improved various aspects of mealtime assistance, including food manipulation [61, 80, 81, 83, 85, 170, 256, 259], skill sequencing [16, 88, 121], and bite transfer [13, 70, 120, 214, 244]. Although further progress is needed before robot mealtime-assistance systems can operate long-term without expert supervision, these recent works significantly enhance their robustness and autonomy.

Building upon these efforts, this work considers the need for *personalization* in mealtime assistance. A one-size-fits-all mealtime assistance system is not enough to address the wide range of preferences, functional abilities, behaviors, and environmental contexts that vary between care recipients and between meals [180]. For example, consider the two community researchers (CRs) shown in Figure 6.1, who are both co-authors of the work in this chapter. CR1, who has Multiple Sclerosis, prefers to lean forward to take a bite. CR2, who has a C4-C6 Spinal Cord Injury and very limited head and neck mobility, requires inside-mouth bite transfer [120] (Chapter 4). In a social dining scenario, CR1 prefers that the robot retract after bite transfer so they can better see their companion. In another social scenario, CR2 prefers to control the robot using a custom *long-continuous-open-mouth* gesture that they create themselves through the user interface (a standard open-mouth gesture would be falsely triggered in conversation). These are a few examples among many underscoring the need for a mealtime assistance system that can be personalized in-the-wild by the users themselves.



Figure 6.1: Informed by a formative study with 19 care recipients and 2 community researchers, we propose FEAST: a flexible mealtime-assistance system grounded in adaptability, transparency, and safety. Through in-home evaluations, we demonstrate that FEAST can personalize to the needs of two care recipients, assisting each with three meals in diverse in-the-wild contexts.

To better understand the nature of personalization in mealtime assistance, we start our work with a *formative user study* (Section 6.2). We use speculative videos to guide in-depth conversations with 21 care recipients who have diverse medical conditions and mobility limitations. This study has two outcomes. The first is a collection of specific personalization requests that we use to develop our system (Section 6.4.1). The second is the realization of three key tenets crucial for personalization in mealtime assistance: *adaptability*, *transparency*, and *safety* (Section 6.4). Care recipients voice the need for systems that can *adapt* to their needs and preferences—not just once, but from meal to meal, and over time as their needs change. They also express that any adaptations should be *transparent*

so that they are able to understand and predict system behavior. Finally, they underscore the importance of *safety*, especially when the system changes.

With these key tenets in mind, we propose FEAST, a flexible mealtime-assistance system towards in-the-wild personalization (Section 6.3). FEAST is designed for personalization at both the hardware and software level. The hardware features a modular tool-change apparatus so that a single robot arm can assist with feeding, drinking, and mouth wiping. The system also features accessible buttons, status LEDs, cameras, microphones, and speakers to enable a wide range of customizable interactions with the user. We also propose a novel feeding utensil that increases the robot's workspace and decreases obstructions to the user's view.

On the software side, to strike a balance between adaptability, transparency, and safety, we propose to sequence together parameterized behavior-tree-based skills to achieve user-specified goals. The parameterized behavior trees provide a mechanism for personalization: user requests, formulated in natural language, are translated with a large language model (LLM) into structured updates to the behavior trees, which can then be statically validated for safety. This structured skill representation can also be analyzed by an LLM in response to user transparency requests. Finally, using the code synthesis capabilities of LLMs, we enable users to create their own custom head gestures that can be added to the behavior trees and used to interact with the robot. Users engage with the robot through a flexible web-based user interface.

We develop FEAST using *community-based participatory research* [89] in collaboration with two CRs (Figure 6.1). This approach, involving co-design and

in-depth evaluation with one or two CRs, is well-established in assistive technology research [69, 91, 146] and increasingly common in assistive robotics [36, 190, 196, 198, 211, 224]. Our collaborations began with introductory video calls in November 2022 and have continued through regular meetings—both virtual and in-person at the CRs' homes. These sessions have been instrumental in gathering feedback on system design, brainstorming studies to identify personalization needs (Section 6.2), and piloting developments.

This iterative process led to a five-day in-home evaluation in January 2025 (Section 6.5), where CRs fed themselves six meals across three distinct contexts: personal, watching TV, and social (Figure 6.1). Results show that CRs successfully completed meals with few researcher interventions and personalized the system on the fly to their preferences and abilities while reporting low cognitive workload, as indicated by NASA Task Load Index (NASA-TLX) surveys [93]. CRs also rated FEAST highly for real-world applicability, as reflected in Technology Acceptance Model [53] survey results. Furthermore, both CRs reported that FEAST provided greater control over their meals and a stronger sense of independence compared to their human caregiver, with one noting that conveying preferences was easier with our system. Finally, to further assess ecological validity, we evaluated FEAST with an Occupational Therapist unfamiliar with the system, who confirms its merits over a no-personalization baseline (Section 6.6).

Overall, our contributions include:

- FEAST: A flexible **mealtime-assistance system** built with **community-based participatory research**, tackling in-the-wild scenarios by integrating diverse skills, custom tools, a flexible interface, and user personalization.

Table 6.1: Comparison between FEAST and other mealtime assistance systems. Adaptability is assessed based on ability to handle open-ended user requests, safety by self-reported adherence to ISO 13482 [113], and transparency by Levels 1-5 of the IEEE Transparency Standard [108].

	Obi [205]	Neater Eater [200]	Park et al. [214]	Bhattacharjee et al. [18]	Feel the Bite [120]	FLAIR [121]	REPEAT [88]	Nanavati et al. [198]	**FEAST (ours)**
Adaptability	-	-	-	-	-	-	-	-	✓
Safety	✓	✓	-	-	-	-	-	✓	✓
Transparency	-	-	-	-	-	-	-	-	✓
Interface	✓	✓	✓	✓	-	-	-	✓	✓
Autonomous Bite Acquisition									
Sequencing	-	-	-	-	-	✓	-	-	✓
Pre-acquisition	-	-	-	-	-	✓	✓	-	✓
Acquisition	-	✓	✓	✓	-	✓	✓	✓	✓
Autonomous Bite Transfer									
Outside Mouth	-	-	✓	-	-	✓	-	✓	✓
Inside Mouth	-	-	✓	✓	✓	✓	-	-	✓
Minimal Occlusion	✓	✓	-	-	-	-	-	-	✓
Workspace Reachability	-	-	✓	-	-	-	-	-	✓
Other Tasks	-	-	-	-	-	-	-	-	✓

● for commercial systems, ● for academic systems

- A user study involving 21 care recipients, which identifies diverse **personalization needs** for mealtime assistance.
- A **personalization framework** built on three key tenets: (i) adaptability via LLM-based code synthesis, (ii) transparency (Levels 1-5) through LLM summarization, and (iii) safety aligned with ISO/TS 13482 safety principles.

- A **five-day in-the-wild system evaluation** with two community researchers, spanning six realistic meals across three distinct environmental contexts, showcasing FEAST's real-world applicability.
- An **evaluation with an Occupational Therapist** unfamiliar with our system, assessing ecological validity and demonstrating improved performance over a non-personalized baseline.

6.2 Identifying Personalization Needs from Care Recipients with Diverse Medical Conditions

To guide the development of our flexible mealtime-assistance system, we conducted a formative user study with 21 care recipients (including 2 CRs) with diverse mobility limitations. Our objectives in this study were two-fold: (1) to collect *specific personalization requests*; and (2) to identify *key tenets* of personalization in the context of mealtime assistance.

Participants. We recruited 19 care recipients with diverse medical conditions such as Cerebral Palsy, Multiple Sclerosis, Muscular Dystrophy, Spinal Cord Injury, etc. (Figure 6.2). There were 8 female and 11 male participants who ranged in age from 20 to 62 years old and who were located throughout the United States. The participants were also diverse in terms of their prior experience with mealtime-assistance systems: some had no experience; others had participated in academic studies; and some had extensive experience (e.g., P6 noted *"I had a Winsford Self-Feeder for over 20 years until the second one I had stopped working 4 months ago"*).



Demographics of Study Participants

ID	Age	Gender	Impairment	Onset Time
P1	46	Male	Arthrogryposis	Since birth
P2	32	Male	Arthrogryposis	Since birth
P3	28	Female	Schizencephaly	Since birth
P4	32	Male	Spinal Muscular Atrophy	Since birth
P5	43	Male	Spinal Cord Injury C2	Since birth
P6	43	Male	Cerebral Palsy	Since birth
P7	20	Male	Cerebral Palsy	Since birth
P8	51	Male	Multiple Sclerosis	Since 30 years
P9	28	Female	Muscular Dystrophy	Since birth
P10	30	Female	Fibromatosis Scoliosis	Since birth
P11	29	Female	Quadriplegic	Since 19 years
P12	42	Female	Spinal Muscular Atrophy	Since birth
P13	62	Female	Spinal Cord Injury C5-C6	Since 21 years
P14	50	Female	Charcot-Marie-Tooth	Since birth
P15	43	Male	Spinal Cord Injury C4	Since 9 years
P16	28	Male	Spinal Muscular Atrophy	Since birth
P17	42	Male	Spinal Muscular Atrophy	Since birth
P18	46	Female	Multiple Sclerosis	27 years
P19	34	Male	Cerebral Palsy	Since birth
CR1	47	Female	Multiple Sclerosis	Since 17 years
CR2	46	Male	Spinal Cord Injury C4-C6	Since 27 years

Figure 6.2: We use speculative videos recorded with community researchers (left) to conduct a formative study with 19 care recipients and the 2 community researchers (demographics shown on the right), to identify user personalization requirements.

Design Materials. Previous work [180] (Chapter 3) has shown that caregiving scenarios vary depending upon (i) care recipient functionality, (ii) care recipient behavior, and (iii) environment. Working with our community researchers, we identified realistic mealtime assistance scenarios where each of these three components vary. We then recorded speculative videos of the scenarios using our robot mealtime-assistance system (see Section 6.3 for system details). The videos were organized into a slideshow and supplemented with conversation prompts (Figure 6.2). For example, one video showed the robot successfully feeding a care recipient with outside-mouth bite transfer; the next video showed the robot failing to feed a different care recipient who has very limited neck mobility; a final video in the sequence showed the robot successfully feeding the latter care recipient with inside-mouth bite transfer. After these three videos, we posed the question: "What unique needs might your own functional abilities pose for the robot?" Other speculative videos preceded discussion questions including: "What unique needs might your own behavioral preferences pose for

the robot?" and "In what scenarios do you typically eat? What unique needs might these scenarios pose for the robot?"

Study Design. To broaden participation in the study, we conducted conversations virtually. Caregivers were also invited to join and actively contribute to the discussion. Each session consisted of the following steps: (1) introductions and agenda; (2) a brief review of recent progress in robot mealtime-assistance systems and an extension of gratitude to the study participants for facilitating this progress; (3) speculative videos and discussion questions; (4) final open-ended discussion and conclusion. Care recipients were compensated $30 USD for their participation in the 90-minute study.

Outcomes. This formative study successfully accomplished its two objectives. First, we collected specific personalization requests, which we group together into various categories and visualize in Figure 6.3. We detail each of these requests in Appendix C.1. These diverse requests highlight the need for personalization in a mealtime-assistance system and suggest where we should focus our efforts to meet the needs of users.

Our second objective was to identify key tenets for personalization. From our conversations, we identified three themes: adaptability, transparency, and safety.

1. *Adaptability:* The first clear tenet of personalization is adaptability: changing system behavior in response to user requests. Study participants highlighted the need for adaptability beyond one-time system setup. For example, they requested the ability to customize the system depending on the feeding scenario—when watching TV, controlling the robot with a button

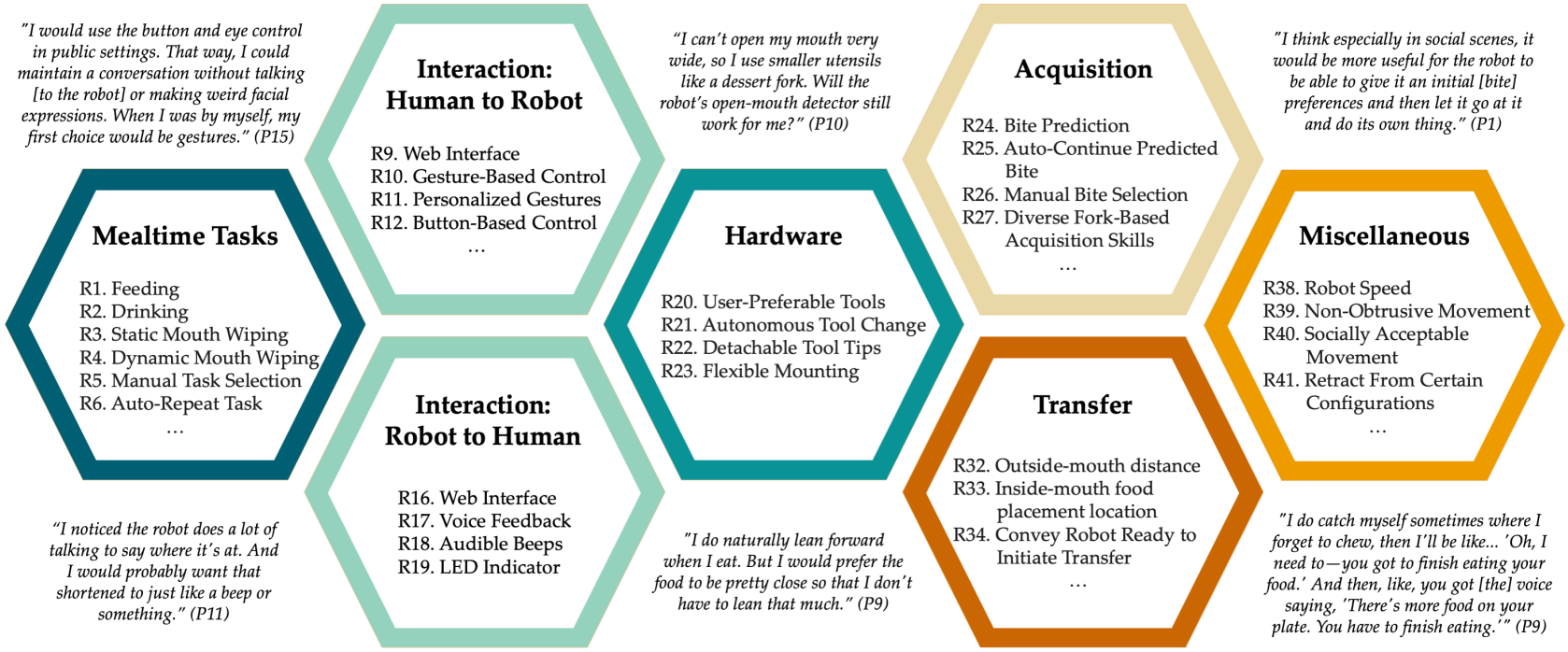


Figure 6.3: We extract and categorize personalization requests from our formative study with 21 care recipients. See Appendix C.1 for details.

may be preferable to verbal commands; or when dining socially, the robot should retract to a resting position immediately after bite transfer. Study participants also described how their preferred interaction with the robot may change throughout the day, depending on their energy level, and over time, as their medical conditions change.

2. *Transparency:* Study participants also emphasized the importance of being able to understand the behavior of the robot, especially if that behavior may change. For example, describing their ideal relationship with the robot, P8 said: *"It would be a symbiotic relationship; I would like to understand the robot capabilities and the robot would understand my requirements."*

3. *Safety:* A final key tenet of personalization that emerged from the formative study was safety. For example, P12 said: *"Safety is the most important part. If I turn my head to look somewhere the robot should not try to feed me at that position which might be unsafe."*

These specific examples and key tenets informed the design of our mealtime-assistance system, which we describe next.

6.3 FEAST: A Flexible Mealtime-Assistance System Towards In-the-Wild Personalization

In this section, we present FEAST, a mealtime-assistance system that enables users to personalize to in-the-wild eating scenarios commonly encountered in

real-world settings. *All hardware and software components of FEAST are open-sourced on our website*[1]. In the following subsections, we describe our system hardware (Section 6.3.1), software (Section 6.3.2), and user interface (Section 6.3.3), explaining how each component can be personalized while adhering to the tenets of adaptability, transparency, and safety.

6.3.1 System Hardware

FEAST (see Figure 6.4) uses a Kinova Gen3 7-degree-of-freedom robot arm [141] and a Robotiq 2F-85 gripper [227]. It can be flexibly mounted either on the user's ROVI wheelchair [231], powered by the wheelchair's battery, or on a movable Vention stand [267], powered by a wall outlet.

Tool-Change Apparatus. FEAST employs three custom tools. First, a novel feeding utensil with integrated motors provides wrist-like degrees of freedom, enabling tasks such as twirling, scooping, and maintaining an upright orientation when holding food. The utensil's fork, made of compliant silicone, is connected to a 6-axis Nano25 ATI force/torque sensor [8]. This default fork is detachable, allowing users to exchange it with a metal fork if desired. To power and control the utensil without dangling wires, the robot's gripper fingertips are replaced with custom fingertips featuring magnetic electrical connections that engage with complementary connectors on the utensil when grasped. Second, for drinking, FEAST uses an adaptable mug handle inspired by adaptable mug holders [288], which accommodates various cup shapes without strict dimensional constraints and features an ArUco marker [75] for autonomous grasping. Third, for mouth wiping, a custom tool with a removable microfiber cloth

[1]Supplementary materials and videos can be found at: `http://emprise.cs.cornell.edu/feast`.

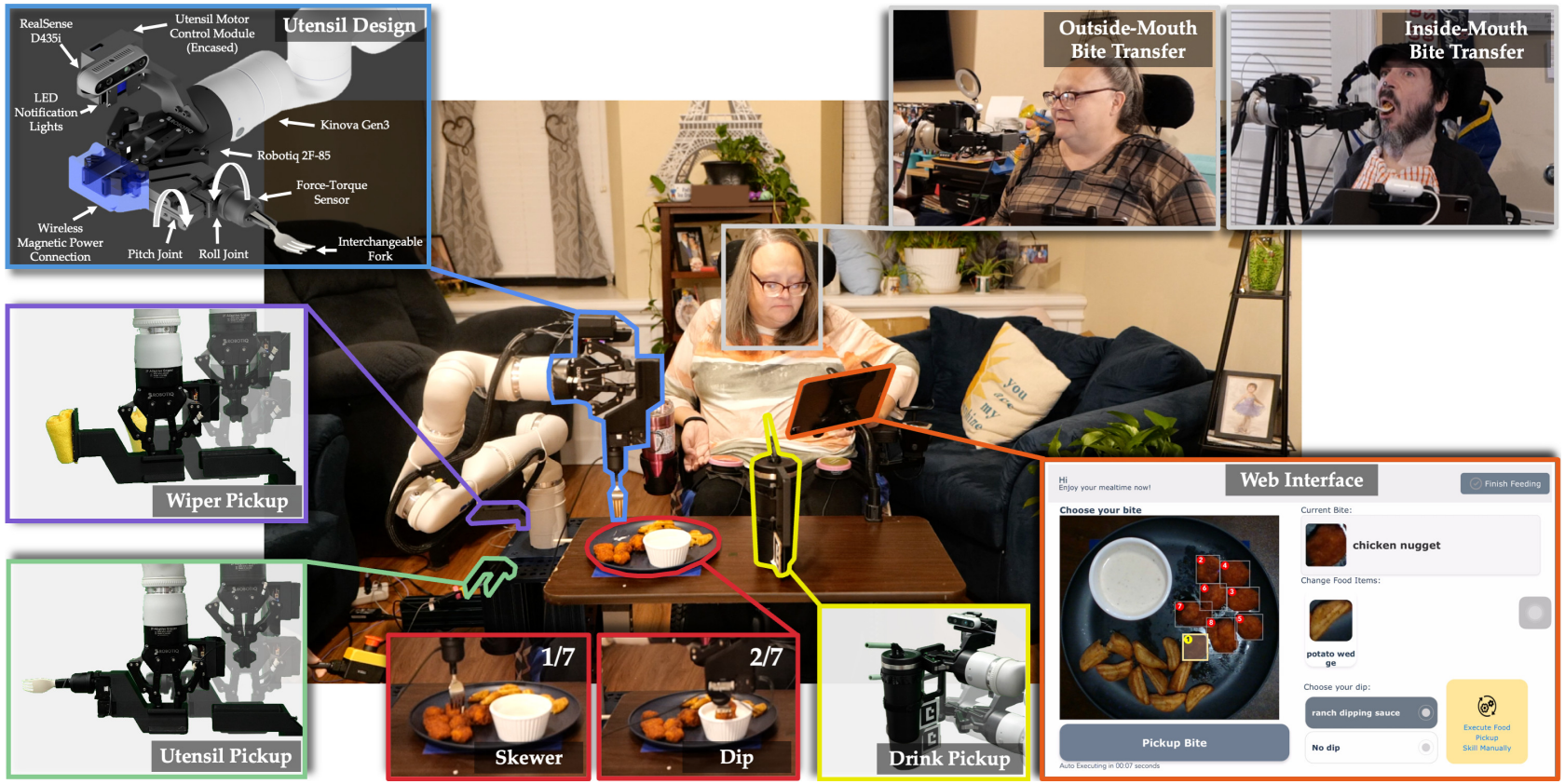


Figure 6.4: FEAST features diverse mealtime-assistance skills such as feeding, drinking and mouth wiping, custom tools, and a flexible web interface, all of which can be personalized in-the-wild by end users.

provides gentle cleaning. Each tool is mounted and dismounted by the robot opening and closing its finger tips (see Figure 6.4).

Novel Utensil Orientation and Camera Mount. Previous state-of-the-art feeding systems use forward-facing in-hand cameras and similarly oriented utensils [70, 120, 121, 198]. However, this design restricts the robot's workspace, requires large movements to transition between acquisition and transfer, and can obstruct the user's view during feeding. Based on feedback from CRs, FEAST introduces a simple and effective solution: positioning tools at a perpendicular angle, mimicking natural human wrist movements during eating. This change, however, requires the camera to align accordingly. To address this, FEAST employs a custom camera mount with an RGB-D Intel RealSense Camera [111], oriented perpendicularly away.

Compute and Networking. FEAST employs custom controllers, such as a task-space compliant controller operating at approximately 1 kHz. This necessitates a dedicated real-time control system, for which we use an Intel NUC [112]. The primary computing platform is a Lenovo Legion Pro 7i laptop equipped with a 16GB RTX 4090 GPU [154]. Communication between the main compute unit, the NUC, the robot, and the web interface is managed via a Nighthawk RAX43 router [201]. When FEAST is mounted on a wheelchair, these components can also be securely mounted to ensure they move along.

Accessible Buttons. FEAST utilizes three accessible buttons [76]. Two are intended for the user: one customizable through our personalization pipeline for interactions, and another serving as an emergency stop. The third button functions as an experimenter emergency stop.

Status LED. FEAST features a status LED within the camera mount to alert users when their attention is needed, based on feedback from CRs. This is especially useful for users multitasking, such as watching TV while eating.

6.3.2 System Software

FEAST sequences together skills to accomplish goals given by the user. For example, if the user requests a drink while the robot is holding the utensil, the robot would invoke skills to (1) dismount the utensil; (2) mount the drink; and (3) transfer the drink to the user. We next describe how skills are generally implemented, adapted, and sequenced together, before detailing the specific skills used in this work.

Skills as Parameterized Behavior Trees. We implement skills as behavior trees [47]. As an extension of standard behavior trees, we expose *node parameters* that can be adapted in response to user requests (see below). For example, the behavior tree for bite acquisition includes three parameters: `Speed`, `TimeToWaitBeforeAutocontinue`, and `AskUserForConfirmation`. Every parameter is associated with a domain of possible values. For example:

- `Speed` $\in$ {`low`, `medium`, `high`}
- `TimeToWaitBeforeAutocontinue` $\in [5, 100]$
- `AskUserForConfirmation` $\in$ {`true`, `false`}.

All nodes and parameters are given human-readable names and descriptions to facilitate LLM-based adaptations.

Personalizing Skills from Natural Language. FEAST users can make personalization requests through spoken or typed natural language. Our pipeline for processing these requests is as follows:

1. The natural language request is converted into structured behavior tree updates using an LLM. (adaptability)
2. Each potential update is checked for safety. If the updates are deemed safe, the behavior trees are updated. Otherwise, a failure is reported. (safety)
3. The updates and outcomes are briefly summarized with an LLM and reported back to the user. (transparency)

We now describe these steps in more detail.

1) Language → Structured Updates. We use an LLM (GPT-4o [107]) to translate natural language personalization requests into structured behavior tree updates. The LLM is prompted with a brief explanation of the scenario, the personalization request, the behavior trees, and the form of an expected response.[2] The output of the LLM is statically checked to ensure that the structured requests are in the expected form and the names of behavior tree nodes and parameters are valid. If any of these checks fail, the LLM is automatically re-prompted (up to 3 times in experiments) with failure information. It is important to note that one personalization request can elicit multiple updates. For example, if the user requests for the robot to "always move as fast as possible," multiple behavior trees with speed parameters would be updated.

[2]We also found that prompting the LLM to rephrase the original request into a "specific setting that should be changed in the robot's software" helped with certain requests such as "The robot is too slow right now."

2) Safety Checks. If the structured behavior tree requests pass static checks, we pass them through another round of safety checks. We consider two types of updates: node additions and parameter changes. In this work, to guarantee safety, we permit only three types of node additions: "pause", "wait for gesture", and "retract." We further restrict "retract" node additions to ensure that the robot only retracts from a limited set of configurations that have been empirically tested. For parameter changes, we ensure that the proposed values lie within the parameter domains. These domains are carefully chosen to ensure that safety is guaranteed, while also giving the user enough flexibility to meet their personalization needs.

3) Update Results → Language. Given the history of structured behavior tree updates and the check outcomes, we again use an LLM to generate a brief summary. The LLM is prompted with a brief explanation of the context, the original user request, the structured updates, and the outcomes, and asked to generate a short summary that a non-technical end user would understand. This summary is displayed on the user interface (Section 6.3.3).

This process—converting natural language requests into structured behavior tree updates that can be checked for safety and summarized back to the user—addresses our three key tenets of personalization. By exposing parameters and allowing for certain node additions, the system is *adaptable*; in summarizing the process, the system is *transparent*; and by restricting and checking the adaptations, the system is *safe*.

Synthesizing New Gestures with LLMs. As an additional step towards open-ended personalization, we allow the user to synthesize new gestures that can then be integrated into the behavior tree—for example, if the user wishes to

use a custom head shaking motion to indicate that they are ready for bite transfer. Through the interface (Section 6.3.3), the user is guided through the process of collecting a few positive and negative examples of their new gesture. The user also gives the gesture a name (e.g., "head shake") and a brief description (e.g., "shaking my head from right to left"). FEAST then uses LLM-based program synthesis to generate a programmatic gesture detector that takes a head tracking module as input. Our program synthesis pipeline is also able to propose and optimize hyperparameters, e.g., `NumberHeadShakes` or `NoiseTolerance`. See Appendix C.2 for details.

Sequencing Skills with PDDL. Towards generating a sequence of skills to achieve a user's goal, we associate each skill with a Planning Domain Definition Language (PDDL) operator [187]. For example:

```
(:action PickTool
  :parameters (?tool - tool)
  :precondition (and (GripperFree) (Reachable ?tool))
  :effect (and (Holding ?tool) (not (GripperFree))
               (not (Reachable ?tool)))
)
```

where `?tool` can be grounded with `utensil`, `mug`, or `wiper`. See McDermott et al. [187] for a formal introduction to PDDL. We use an optimal task planner (FastDownward [100] with alias `seq-opt-lmcut`) to sequence the skills together given a known initial propositional state and a propositional goal derived from the user's input. For example, when the user requests a bite, FastDownward generates a three-step plan: `PickTool(utensil)`, `AcquireBite(utensil)`, `TransferTool(utensil)`. The plan is executed open-loop and the propositional state is updated using the operator effects. If the user subsequently requests a drink after requesting a bite, the second plan would be: `PlaceTool(utensil)`, `PickTool(mug)`, `TransferTool(mug)`.

Skills Included in FEAST. We now detail the specific skills that are currently implemented in FEAST.

- `PickTool`: Grasping routine for *utensil*, *mug*, or *wiper*. The utensil and wiper are placed in a fixed tool mount relative to the robot's base, enabling a predefined pickup motion. For the mug, the robot detects the ArUco marker on the adaptable handle and executes an appropriate grasp.
- `PlaceTool`: Returns the tool to its last known location. For utensils or wipers, this is a predefined position on the tool mount; for the mug, it is the original pickup position.
- `AcquireBite`: Picks up food from the plate. FEAST builds on the bite acquisition framework proposed by FLAIR [121] (Chapter 5), adapting it to our novel utensil. This framework includes a vision-parameterized food manipulation skill library with four pickup actions—skewering, scooping, twirling, and dipping—and three pre-acquisition actions—pushing, grouping, and cutting—to prepare for future pickups. We incorporate force thresholds via the force-torque sensor to ensure safe operation. For each new bite, FEAST processes the plate image using vision-language foundation models to identify plate contents and predict the required skill sequence for each food type. These predictions, along with the user's bite order preferences and bite history, are provided to an LLM, which determines the next bite while balancing preference adherence and efficiency. The robot then executes the predicted skills to pick up the bite.
- `TransferTool`: By default, FEAST performs outside-mouth transfer [70], where it detects the user's head pose once and moves to a predefined distance outside the mouth. Upon user request, FEAST can switch to inside-mouth

transfer [120], using a task-space compliant controller to continuously track the user's head pose and bring the tool inside the mouth. After detecting that the user has finished taking a bite, sip, or completed mouth wiping, it returns to its initial position. Both transfer methods utilize the head perception pipeline proposed by Feel the Bite [120] (Chapter 4).

- `EmulateTransfer`: Moves the gripper to a configuration just in front of the user's mouth. Useful for recording and testing new gestures as we detail in the next section.
- `Retract`: Moves the robot to the retract configuration with an empty gripper.

6.3.3 User Interface

FEAST features a web-based interface accessible on personal devices, such as tablets or phones, enabling integration with assistive technologies that care recipients already use. For example, CR2 uses a tracker that follows a reflective dot on their nose [184] (Figure 6.4). We refine the interface design through multiple iterations with CRs to address their needs; they highlight that the interface should have big buttons that are easy to click, and should support speech to text.

We implement the interface using Vue.js [287], an open-source front-end JavaScript framework, while the Robot Operating System [207] facilitates communication with the mealtime-assistance skills. FEAST's web interface displays pages on demand from the robot, automatically updating to reflect system behavior as users personalize it to meet their evolving needs. The interface supports system adaptability and transparency via a personalization webpage, as we detail in Section 6.3.4.

Beyond fully autonomous workflows that can continuously feed users without intervention, FEAST's interface allows on-demand user control. Users can override the robot's autonomous predictions, for instance, by manually selecting the next bite or pinpointing key points on a displayed plate image as parameters to bite acquisition skills. Such overrides help users recover from autonomous errors, increasing the robustness of the system in in-the-wild scenarios [95, 273].

6.3.4 Default Mealtime-Assistance Procedure

Mealtime Assistance. At the start of a new meal, the robot begins in the "retract configuration" without holding any tool. The user is presented with the *New Meal Page* on the web interface, where they specify the food items on the plate and their preferred bite order (e.g., "Feed me all of X first, then Y"). The interface then transitions to the *Task Selection Page*, allowing the user to choose from three primary actions: requesting a bite, taking a sip, or wiping their mouth, which sends `TransferTool(utensil)`, `TransferTool(mug)`, and `TransferTool(wiper)` commands respectively. If the robot is not holding the appropriate tool, the task planner directs it to place the current tool and pick up the correct one. Moreover, for a bite, the task planner invokes `AcquireBite`, which looks at the plate contents and predicts the next bite in adherence to the user's ordering preferences. The interface subsequently opens a *Bite Acquisition Page*, where the user can:

1. Switch to a different bite of the same or another food type.
2. Switch to a *Manual Bite Acquisition Page* to choose a specific skill (e.g., skewering) and set keypoint parameters (e.g., the exact skewering point on the plate image).

If the user clicks "acquire bite" or does not press any button within 10 seconds, the robot proceeds with bite acquisition. For `TransferTool`, the interface prompts the user to confirm readiness for transfer. Once confirmed, the robot moves to a ready-to-transfer configuration and announces, "Please open your mouth when ready." It waits for the user to open their mouth and then moves the fork, straw, or wiping tool tip near the user's mouth, saying, "Ready for transfer." After detecting a bite via the force/torque sensor, or a completed sip or wipe through a head nod, the robot retracts to the ready-to-transfer position. It then returns to the Next Task Selection Page, with a 10-second auto-continue countdown to repeat the last task if it was a bite or sip. At any point during the meal, the user can click a "finish eating" button that triggers the `Retract` skill.

Personalization. Before and during a meal, users can access the *Personalization Page*, where they can send:

1. Adaptability Requests: Users can input commands to personalize components of the mealtime-assistance system to meet their needs (details in Section 6.4.1).
2. Transparency Requests: Users can ask about the robot's behavior, and the system responds through textual answers (details in Section 6.4.2).

The page also allows users to transition to the *Gestures Page*, which focuses on adding personalized gestures to the system's library. To add a new gesture, users provide a language label and description. The interface then triggers `EmulateTransfer`, moving the robot to a position just outside the user's mouth without holding a utensil to record positive and negative gesture examples. Once the recordings are complete, the robot returns to the ready-to-transfer

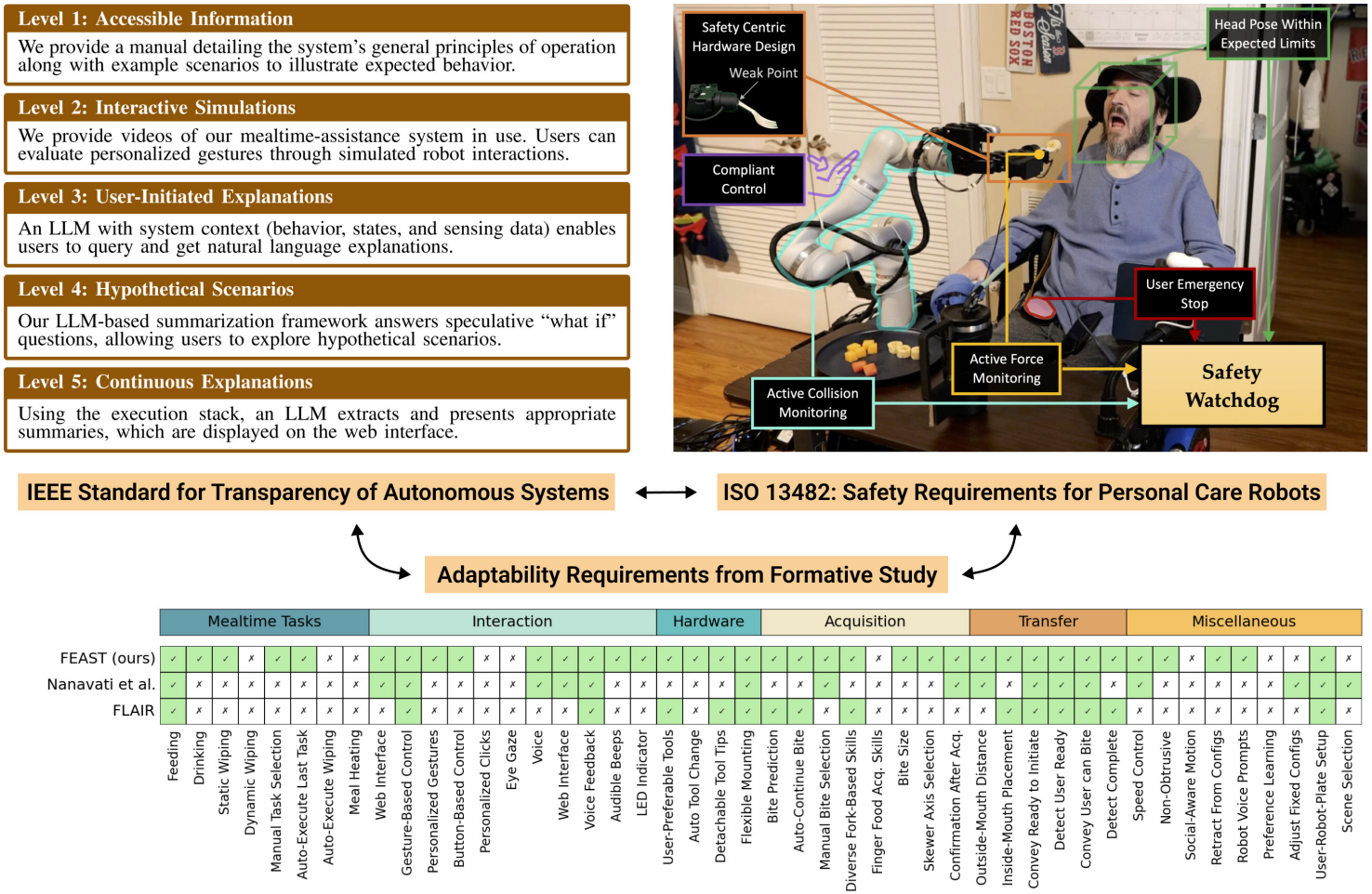


Figure 6.5: FEAST proposes a personalization framework for mealtime assistance built on three key tenets: (i) adaptability towards tackling the various request types identified through our formative study, (ii) transparency towards Levels 1-5 on the IEEE Transparency Standard, and (iii) safety aligned with ISO/TS 13482 safety principles.

configuration, and the interface displays the accuracy of the newly trained gesture detector. Users can also test gesture detectors by having the robot move to the position just outside their mouth (without holding a utensil) while the interface displays whether the detector is triggered by their movements. When switching between webpages, the interface displays a white screen with dynamic text, continuously explaining its current actions. Together, these personalization features enhance adaptability and align with the IEEE Transparency framework (Section 6.4.2).

6.4 FEAST Achieves the Three Tenets of Personalization

Our formative study affirmed the importance of adaptability, transparency, and safety for a mealtime assistance system that is personalized in-the-wild. In this section, we detail how the system design of FEAST addresses these three tenets.

6.4.1 Adaptability

In our formative user study, care recipients voiced 46 types of adaptability requests across 7 categories (Figure 6.3). Requests of the same type often differed from user to user. For example, P1 said: *"I would want the robot to be fast, as fast as it could go maybe up until those very last moments of moving towards my head,"* while P9 said, *"I have a trach and have to eat slowly."* P10 said: *"It would be useful to give the robot more autonomy during those social situations, to pick the bite or whatever, rather than me specifying something exactly,"* and P5 said, *"This might be just a personal thing, but the idea of hitting a switch and then waiting for something to happen makes me then feel like I'm no longer in control."* Users also voiced the importance

of adapting over time: for example, P8 mentioned, *"I'm able to do that right now. But these things might change over time. So you might want to keep updating the robot with the progression of the disease."*

In Figure 6.5, we compare FEAST to two state-of-the-art feeding systems, Nanavati et al. [198] and FLAIR [121], based on their ability to address the 46 adaptability request types identified in our study. FEAST covers 36, compared to 17 for Nanavati et al. and 15 for FLAIR. Appendix C.1 provides detailed justifications for each system. Appendix C.2 presents an ablation study demonstrating that our LLM-based personalized gesture synthesis outperforms a no-personalization baseline.

6.4.2 Transparency

The IEEE Standard for Transparency of Autonomous Systems [108, 277], which explicitly includes care robots within its scope, defines five levels of transparency requirements, ranging from 1 to 5 (Figure 6.5). FEAST aims to meet all these levels.

Level 1 specifies that users must have access to information such as example scenarios and general principles of operation. FEAST satisfies this requirement with an instruction manual provided to new users. Level 2 requires interactive training material, enabling users to rehearse their interactions with the system in virtual scenarios. To address this, we provide demonstration videos of the system across various use cases. Moreover, after adding personalized gestures, users can test them directly on the real robot in isolation.

Levels 3 and 4 focus on providing user-initiated explanations. Level 3 requires explanations of the system's most recent activity, while Level 4 demands answers to "what if" queries about system behavior. FEAST uses LLM summarization to fulfill these requirements. We define the following to encapsulate the current system state:

1. Current Behavior: Current behavior tree encodings for mealtime-assistance skills, with text descriptions for each node and its parameters.
2. Node Execution History: A real-time log of behavior tree nodes in-execution and completed.
3. Perception Log: A real-time log of perceived data, such as plate contents, drink pose, user head pose, and gestures.
4. Safety Log: A real-time log of safety checks, including any invalidated specifications.

On the Personalization Page of the web interface, users can submit language queries. The LLM processes these queries—along with the current system state, detailed descriptions of system operations (as outlined in Section 6.3), and prior user queries and the corresponding LLM's responses—to generate concise, understandable explanations for non-expert users.

Level 5 transparency requires the system to provide continuous explanations of its behavior. FEAST aims to achieve this using LLM summarization. At fixed-time intervals, the system checks for changes in the system state as defined above. If a change is detected, the LLM is queried with detailed descriptions of

system operations, the previous state, and the current state, prompting it to generate an explanation of what changed during that period. This explanation is displayed on the web interface whenever no other page is active.

6.4.3 Safety

FEAST fits within the scope of ISO 13482: Robots and Robotic Devices–Safety Requirements for Personal Care Robots [113], categorized as a "physical assistant robot", with the following hardware and software checks (Figure 6.5):

Pose filtering. Our pose detection methods (food, drink, and head pose) use predefined zones based on expected plate, drink, and head positions, initialized at the start of each mealtime assistance scenario. Head pose limits are tailored to the user's range of motion [225]. If detections occur outside these zones—such as mistakenly identifying a caregiver's head pose behind the user during feeding (a scenario that can arise in real-world settings)—the robot transitions to a safety state.

Active collision monitoring. While our robot verifies its motion plans for collisions with known obstacles, it also uses active collision monitoring to handle unexpected collisions when not using compliant control. This is achieved by comparing torque feedback from the joint F/T sensors with torque predictions from a nominal model-based control law [90].

Compliant control. For inside-mouth transfer, the most physical human-robot interaction intensive aspect of mealtime assistance, our robot switches to a custom Cartesian-space compliant controller that leverages Damped Least Squares [194].

Safety-centric hardware design. Within the feeding utensil, the fork is attached to a custom-designed holder with a mechanical weak point that yields under excessive force, breaking the utensil into two pieces. This safety feature ensures that if excessive force is applied while the utensil is in the user's mouth, only the fork tip remains, halting all physical interaction with the robot. For safety, we restrict outside-mouth bite transfers to the default silicone fork. Similarly, the drinking utensil uses a silicone straw, and the mouth wiper features a soft tip.

Emergency Stops. At all times, both the user and an experimenter have access to physical emergency stop buttons, which immediately transition the robot into a safety state.

Watchdog. A watchdog continuously monitors the functionality of all robot sensors and emergency stops, transitioning the robot to a safety state if an issue is detected.

We provide implementation details for the above and further discuss alignment with ISO 13482 guidelines in Appendix C.3.

6.5 Five-Day In-the-Wild Evaluation

To showcase the real-world applicability of FEAST, we evaluate our system with the two community researchers (CR1 and CR2) in a five-day in-home study spanning six realistic meals across three distinct environmental contexts:

1. **Personal context** where they focus solely on eating.
2. **Watching television** while they are eating.
3. **Social context** where they eat with another researcher.

This study asks the question: **can end-users effectively personalize FEAST to meet their requirements across various mealtime scenarios?**

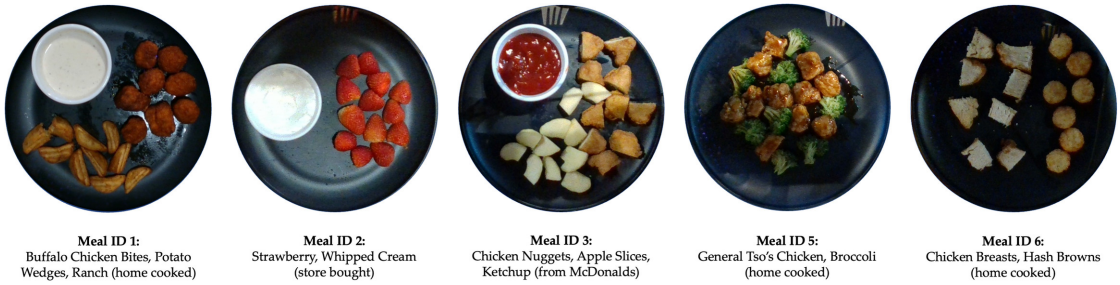


Figure 6.6: Meals fed to care recipients. The images, captured by the robot's in-hand camera, highlight the variability in lighting conditions. Each meal also included water. Meal ID 4 consisted only of a protein shake, which CR2 usually has for lunch.

6.5.1 Study Procedure

Introductory Meals and System Familiarization. Before any evaluation meals, we conduct one introductory meal with each care recipient on Day 1 of the study to familiarize them with the system. During these training meals, we explain the components of the mealtime-assistance system, demonstrate its various personalization features, and answer their questions.

Evaluation Meals. Over the course of the next 4 days, we feed 3 meals each to the two CRs, in their own homes (Figure 6.6), in the following chronological order:

- **Meal ID 1** (on Day 2): CR1 Dinner in Personal Context – buffalo chicken bites, potato wedges, and ranch dressing (home cooked meal), served with water.
- **Meal ID 2** (on Day 2): CR1 After-dinner dessert while Watching TV – strawberries with whipped cream (store-bought meal), served with water.

- **Meal ID 3** (on Day 3): CR2 Dinner in Personal Context – chicken nuggets, apple slices, and ketchup purchased from McDonald's, served with water.
- **Meal ID 4** (on Day 4): CR2 Lunch in Social Context – protein shake (CR2's usual lunch). Mug is cleaned and refilled with water after the shake is finished upon request.
- **Meal ID 5** (on Day 5): CR2 Dinner while Watching TV – General Tso's Chicken and Broccoli (home cooked meal), served with water.
- **Meal ID 6** (on Day 5): CR1 Dinner in Social Context – chicken breast strips and hash browns (home cooked meal), served with water.

We select the meals for the study based on the care recipient's eating habits, the robot's capabilities, and the need for variation. While we validate our system with similar meal types during study preparation, the exact meals naturally vary based on how they are prepared and what is available. Many adjustments were made on the fly; for example, hash browns from CR1's fridge were added to Meal ID 6 at the last minute.

Environmental and User-Specific Variability. Care recipients live in different homes and have unique preferences for their environment setup. This leads to variations both across and within the same context (see Figure 6.1), including:

- **Tools:** CR1 uses a metal fork for all meals and a pink mug for the last meal. CR2 uses a silicone fork (safer for inside-mouth transfer) and a black mug for all meals.
- **Web Interface Interaction:** Both use iPads. CR1, with limited limb mobility, interacts using their left arm, while CR2, with very limited limb mobility, uses a mouse tracker that follows a reflective dot on their nose.

- **Eating Tables:** Different table types include multiple overbed tables, a coffee table, and a dining table.
- **Seating Arrangements:** While watching TV, both CR1 and CR2 face the screen. In a social setting, CR1's social partner sits to their right, whereas CR2's social partner sits directly in front.

Meal Setup and Scene Configuration. For each new scene, experimenters update scene parameters used for safety checks, such as table height (ensuring a lower limit for acquisition actions) and the user's head pose (used for filtering head perception). They also adjust the fixed configurations that the robot moves to before detecting the plate, drink, and user's head to ensure their visibility. After the CRs personalize the robot for their personal contexts in their first evaluation meals, we initialize their other contexts with those settings.

6.5.2 Evaluation Metrics

Before the study, participants rate their experiences with human caregivers during mealtime on a 5-point Likert scale, which is later compared to FEAST. To evaluate system performance, we log experimenter interventions for each meal, categorized into hardware, web interface, skills, and personalization. We also track experimenter explanations, recording the number of questions participants ask. Skill success rates are measured for autonomous actions, which rely on perception, such as bite acquisition, bite transfer, drink pickup, drink transfer, and wipe transfer. Fixed-motion actions, such as utensil and mouth wipe pickup from the tool mount and subsequent stow, always succeed. After each meal, participants complete a NASA-TLX [93] survey (7-point Likert scale) to assess cognitive workload. At the end of the study, a Technology Acceptance

Model (TAM) [53] survey evaluates overall satisfaction and usability, along with additional questions assessing key personalization tenets: adaptability, transparency, and safety. Full questionnaires are in Appendix C.4.

Table 6.2: Per-meal system performance, including experimenter interventions, explanations, and skill success rates.

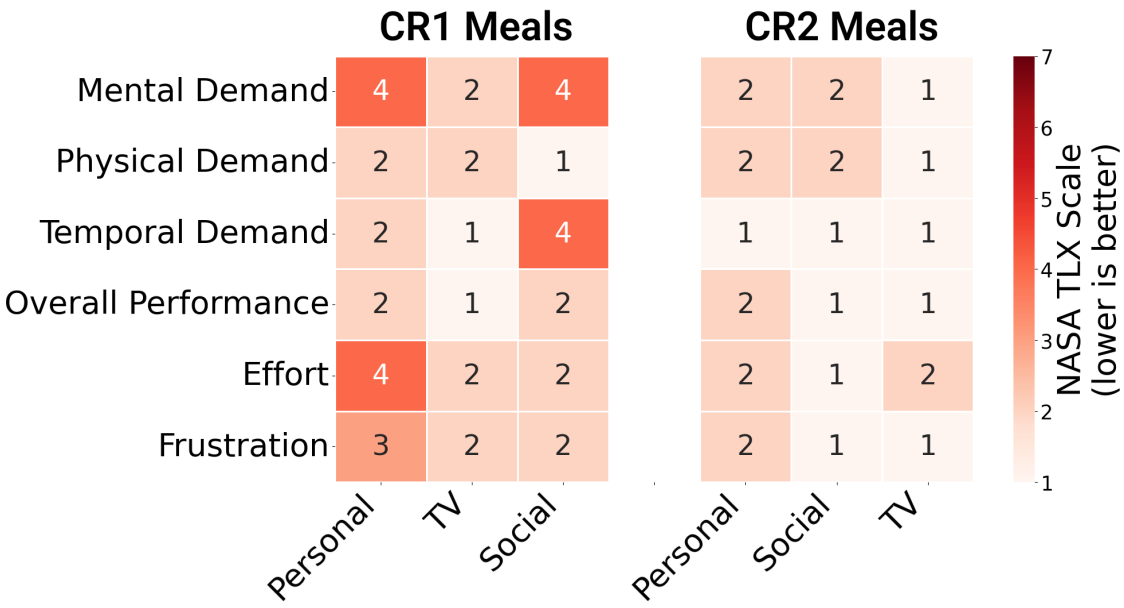
Meal	Interventions				Explanations	Skill Success Rate				
	Hardware	Web Interface	Skill	Personalization		Bite Acq.	Bite Transfer	Drink Acq.	Drink Transfer	Wipe Transfer
#1: CR1 Personal	4	2	1	0	8	24/25	14/17	3/7	2/3	1/1
#2: CR1 TV	1	1	0	0	1	22/27	11/11	1/1	1/1	-
#3: CR2 Personal	1	1	0	0	6	29/37	15/17	3/3	3/3	1/1
#4: CR2 Social	1	0	1	0	2	-	-	2/2	5/5	-
#5: CR2 TV	0	4	0	0	1	19/21	18/19	4/4	2/2	2/2
#6: CR1 Social	0	0	0	1	2	12/12	12/12	2/2	2/2	1/1

6.5.3 Results

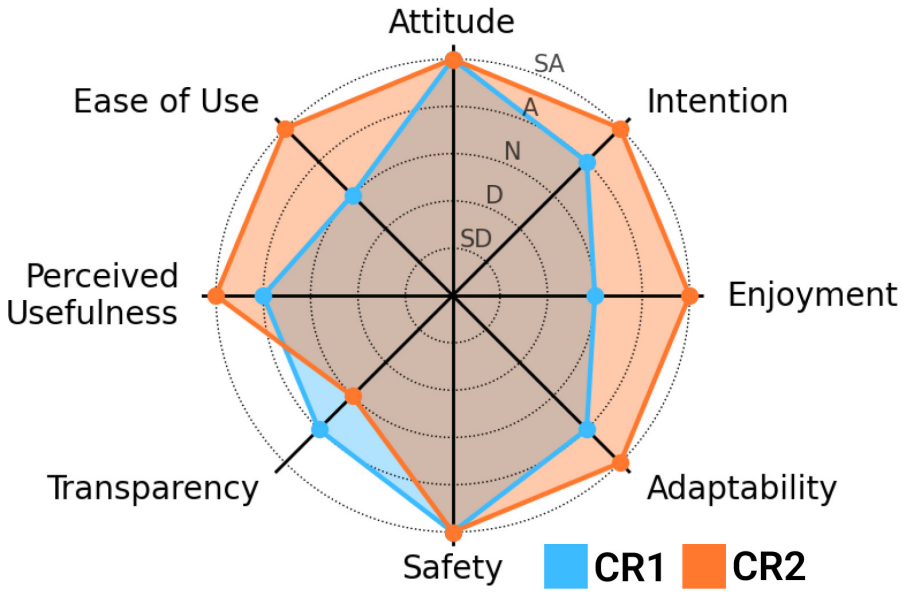
For each meal, CR1 and CR2 reported either fully clearing their plate or eating until they were full. Table 6.2 summarizes objective system performance per meal, whereas Figure 6.7 shows subjective metrics. Details on each of the interventions, explanations, and personalization requests made during the study can be found in Appendix C.5.

Interventions. A total of 18 experimenter interventions occurred across six meals ($\mathbf{3 \pm 2}$). Four hardware interventions in the first meal were due to a damaged ArUco marker on the mug handle, which was later replaced. All but one web interface intervention resulted from a recurring network issue that intermittently blocked commands, requiring a refresh.

Explanations. Users asked experimenters 20 questions across six meals ($\mathbf{3.33 \pm 2.69}$). Many could have been directed to the robot, but the transparency page was only accessible from the task selection page, underscoring the need for transparency throughout the eating process.

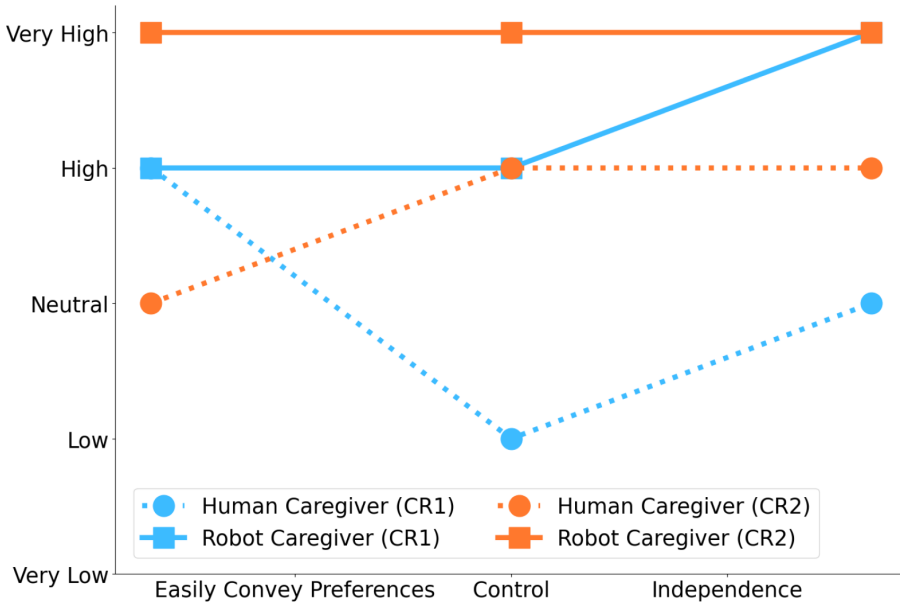


(a)



(b)

Figure 6.7: (a) Per-meal NASA-TLX survey results, with CR1 and CR2 meals shown in chronological order. (b) TAM results, including questions on personalization tenets. (c) Comparison between human caregivers and FEAST. (continued on the next page)



(c)

Figure 6.7: (continued) Comparison between human caregivers and FEAST.

Skill Success Rates. FEAST demonstrates robustness to different food items, with a bite acquisition success rate of **89.27 ± 9.24**% and a bite transfer success rate of **93.07 ± 7.70**%. Prior work suggests that an 80% acquisition rate is sufficient for use [18]. While drink acquisition success was low for Meal ID 1 due to a damaged marker on the drink handle during transit, it improved significantly after replacement, averaging **100.0 ± 0.0**% across Meal IDs 2-5. Drink and wipe transfers were also highly successful at **94.44 ± 13.61**% and **100.0 ± 0.0**% respectively.

Cognitive Workload Per Meal. The average workload imposed by FEAST was relatively low. On a 0–100 NASA-TLX scale (where lower scores indicate lower workload), CR1 had a mean of **22.22 ± 10.02**, while CR2 had a mean of **7.41 ± 5.79**, compared to a baseline of 37 [102, 196]. Both care recipients rated

their success in achieving mealtime goals very positively: the inverted Performance metric (0–100, lower = better) is **11.11 ± 9.62** for CR1 and **5.56 ± 9.62** for CR2.

Comparison with Human Caregiver. Both CRs rated FEAST as providing greater control over their meals and a greater sense of independence compared to their professional caregivers. CR2 also reported less effort in conveying preferences to the robot than to their human caregiver.

TAMs and Personalization Tenets. Results from the TAM survey indicate that both CRs rated FEAST highly across all metrics, with a mean score of $\geq 4/5$ for Perceived Usefulness, Ease of Use, Attitude, Intention, and Enjoyment. They also rated the system highly for adaptability and safety but noted that transparency could be improved. This may be because the transparency page was only accessible from the task selection page on the web interface, which was not always available, requiring them to ask experimenters for explanations.

6.5.4 Lessons Learned

Lesson 1
Significant variability exists across in-home eating scenarios, and in-the-wild personalization allows users to uniquely adapt the system to these variations.

Across different meals, both CR1 and CR2 made a range of personalization requests. These included adjusting robot speed (*"Feed me as fast as you can,"* Meal ID 1), modifying skill parameters (*"Dip the strawberry deeper into the whipped cream,"* Meal ID 2), changing web interface workflows (*"Don't show continue pages on the web interface,"* Meal ID 3), and customizing interactions between the robot and user, such as muting the robot (*"Be quiet and do not talk at all,"* Meal ID 5) or

changing how transfers are confirmed (*"Use the button to complete a transfer only when taking a sip,"* Meal ID 1).

Even under similar contexts, the CRs had different personalization needs. For example, CR2 added a *"continuous mouth open"* personalized gesture detector for initiating transfers in a social setting, to differentiate from when they are talking. On the other hand, CR1 realized the default mouth-open detector required opening their mouth very wide. This design allowed CR1 to talk without inadvertently triggering the robot, and then open their mouth wide when ready to receive a bite. Thus, CR1 continued using it in social settings.

The spatial arrangement of people and objects also influenced personalization preferences. In CR2's social context, the social partner sat across the table, so CR2 rarely felt obstructed by the robot: *"I completely forgot about the robot sometimes and had to remember that I had to take a drink."* Conversely, in CR1's social context, the social partner was seated to the right—directly across from the robot—prompting CR1 to request the robot retract after every bite (*"Move to retract position after every bite"*) to avoid interference.

Lesson 2
Transparency helps users iteratively refine the system to meet their preferences, even when adaptability commands are not always effective.

FEAST's adaptability leverages an LLM to process language commands and update the robot's behavior. However, LLMs can sometimes hallucinate or make mistakes. For example, when CR1 tried to switch to a button for wipe transfers with the command, *"Use button when completing a transfer when taking a sip,"* the LLM mistakenly applied this change to all tools, responding, *"The robot-assisted feeding system has been updated to use a button to complete actions when transferring*

drinks, utensils, and wipes." Despite the error, the system's transparency features allowed the user to recover. CR1 utilized transparency to verify the transfer completion method by asking, *"What is the default action to complete a transfer?"* and *"What other ways can I end a transfer besides pushing the button?"* After several transparency-adaptability iterations, CR1 submitted a detailed adaptability request: *"Use sensors to end transactions with utensil use, the button to end transactions when taking a sip, and the button to end transactions for face wipes."* Through these transparent interactions, CR1 was able to iteratively refine the robot's behavior to match their preferences, even when sometimes the initial adaptability command did not function as intended.

Lesson 3

Providing multiple interfaces is essential for transparency, as users may not always be able to interact with a single interface due to situational and environmental constraints.

Over multiple days of evaluating a flexible system, users often forgot the robot's settings and expectations (which gesture it is waiting to detect). FEAST provides transparency through voice prompts and a web interface, with multiple options proving crucial. In social settings, CR1 and CR2 switched off voice prompts and relied on the web interface for guidance. However, in cases where neither was accessible—such as when CR2 used a personalized gesture detector for *"keep head still"* during mouth wiping—the wiper's position right at their mouth prevented them from checking the interface. As CR2 noted, *"A small display on the gripper would help. I can't check the web interface while the fork or another tool is at my mouth and the robot expects something. Or maybe I could ask directly, like 'Hey robot, what am I supposed to do right now?' "*

Lesson 4
Cognitive workload generally decreases as users become more familiar with the system, but it also depends on the context and specific settings they choose.

A flexible system allows users to tinker and personalize their experience, but it can also impose cognitive workload. While NASA-TLX surveys indicated that our system did not impose significant cognitive workload during evaluation meals, this may be because introductory meals helped users acclimate. Over three meals, cognitive workload generally trended downward (Figure 6.7) as users became more familiar with the system, except for Meal ID 6 (a social meal for CR1). CR1 noted that the mental and temporal demand was not caused by the personalization process itself but rather by the specific settings they had chosen, which were not optimal for the context. *"I set the [auto-continue task selection] timer to 100 seconds, but that wasn't enough. I kept watching the countdown while trying to keep up with the conversation, making sure I was ready for my next bite before time ran out. Next time, I'd set it much longer."* Despite this challenge, they acknowledged that familiarity would make the process easier over time: *"The more I eat in this setting, the easier it'll get. I'll get used to what the robot can do and figure out which settings actually work for me."*

Lesson 5
System failures can occur in-the-wild for various reasons, but system flexibility and keeping the user in the loop improves the robot's ability to recover.

During our in-home evaluation, several unexpected skill failures occurred. For example, varying lighting conditions affected food perception, making it difficult for the robot to detect food items on the plate. However, the system provided a manual skill execution option on the web interface, allowing users to pick up undetected bites by selecting the skewering skill and a keypoint on

a plate image. In cases where bite transfer failed and the robot was unaware, it would return to the plate to reacquire food, even though a food item was already on the fork. To recover, users switched to manual skill selection, skewering an empty space on the plate to retain the same food item on the fork and retry bite transfer.

Similarly, the default head nod detector—used to detect when a user was finished drinking or wiping their mouth—did not work well for CR1. When leaning forward to take a sip or wipe, they inadvertently leaned downward, which the system misinterpreted as a nod, causing the robot to move away too soon. However, because the system was flexible, CR1 was able to switch from gesture detection to a button for this task.

These examples highlight how system flexibility, combined with user involvement, allows the robot to recover from difficult to predict failures that arise in the real world.

6.6 Assessing Ecological Validity with an Occupational Therapist

To assess FEAST's performance beyond the two community researchers, we evaluate our system with an Occupational Therapist (OT) not involved in system development procedure. Since OTs work with diverse care recipients with varying mobility limitations, their evaluation offers a comprehensive measure of the system's ecological validity. The OT was introduced to FEAST as in our in-the-wild evaluations—through the training manual and a training meal. After reviewing

personalization categories from our formative study, the OT independently created 10 realistic personalization scenarios. For each scenario, they personalized the default FEAST system to their requirements using adaptability and transparency requests. We report: (1) user-reported personalization success rated at 5/5, averaged across scenarios; (2) user-reported ease of personalization rated at 4.6 ± 0.16 out of 5, averaged across scenarios; (3) a preference for the personalized system over the default (no-personalization) system in all 10 scenarios; (4) a total of 13 adaptability and 2 transparency requests across all scenarios; and (5) a total of 2 experimenter interventions and 1 experimenter explanation throughout the evaluation. User ratings are on a 1–5 Likert scale (1 = Strongly Disagree, 5 = Strongly Agree). Appendix C.6 details the personalization scenarios, including the adaptability and transparency requests made and any experimenter interventions or explanations. The OT also offered feedback on the interface design, stating: "Visually, it looks really good; the contrast is good, it should work for most care recipients... having speech-to-text is helpful because that is how many communicate with their devices."

6.7 Limitations and Future Work

Open-Set Food Detection. FEAST receives user input about food items on the plate and uses an open-set object detector to locate them. However, the detector often mislabels items, particularly when plates contain multiple food types (e.g., chicken nuggets and potato wedges). During evaluation, additional qualifiers like "yellow potato wedge piece" or "cut-up chicken nugget piece" were required. While users could potentially provide such qualifiers to improve detection accuracy, the effectiveness of this approach remains to be evaluated. In

the future, more accurate open-set detectors are needed to address this limitation.

Adaptability Based on Plate Context: Some user adaptability requests required the system to have additional context about plate contents and selectively adjust behavior tree parameters. For instance, in Meal ID 1, CR1 requested more ranch for potato wedges and less for chicken bites. However, since FEAST currently uses a single dipping parameter, this required constant adjustments between bites. Future work should explore adaptability mechanisms that dynamically adjust behavior based on real-time plate contents.

LLM Hallucinations. FEAST leverages LLMs to interpret user requests and modify robot behavior, but hallucinations can lead to incorrect or incomplete adaptations, causing user confusion. While FEAST's transparency features help users identify and correct such errors, these features themselves rely on LLMs, which may also hallucinate. Future research should focus on reducing LLM-induced errors and exploring fallback mechanisms to ensure reliability.

Long-term Preference Learning: CR1 noted that generating a full set of personalization requests from scratch for a new setting could impose unnecessary cognitive workload. They suggested allowing users to describe their dining environment so that the system could initialize a relevant configuration, from which they could make refinements. Achieving this would require long-horizon preference learning, where FEAST analyzes past user preferences to propose suitable initial settings for different contexts.

Evaluation Beyond Community Researchers. Our evaluation involved two care recipients who are also community researchers. Since they were familiar

with FEAST's development, they may have had a lower cognitive workload than other users. Future studies should evaluate FEAST with care recipients who are not community researchers to better understand how unfamiliar users perceive the system's usability.

Open-Loop Skill Limitations. FEAST employs open-loop skills for bite acquisition, which can fail when food items slip. Moreover, while users can adjust skill parameters to control bite size—such as scooping more or less—these settings often require frequent adjustments within a single meal as food properties change over time. For example, the same scooping distance may yield smaller portions as food consistency shifts. Incorporating closed-loop approaches to bite acquisition could improve robustness, allowing the system to adapt dynamically to food slippage and changing food properties.

Part III

Deployment for a Month: Personalizing Across Tasks and Over Time

CHAPTER 7

COLORING BETWEEN THE LINES: PERSONALIZATION ACROSS TASKS

7.1 Introduction

When an assistive robot first arrives at a new home, restaurant, or hospital, it should already know a great deal—how to move, how to manipulate objects, and how to interpret human-given commands, among many other general competencies. But it will not yet know the details that make each individual and environment unique. It will not know that one care recipient enjoys tacos but dislikes cilantro; that another has limited range of motion [173]; or that another likes to watch TV while eating dinner. To be truly effective in the long term, the robot should learn these nuances over time and *personalize* its behavior accordingly [114, 165, 180, 215].

Continual personalization has been a longstanding goal in human-robot interaction, but it has remained elusive because of the inherent tension between *safety* and *generalizability* of preferences across tasks. An over-constrained robot cannot adapt; an under-constrained robot can be dangerous. Previous work on personalization has largely deferred this challenge by focusing on task-specific forms of personalization [29, 31, 72, 122, 235, 276, 280, 293] or by working in settings without critical safety requirements [35, 275, 295]. In this work, we propose a general framework for resolving these competing concerns.

This requires a task representation that is expressive enough to generalize across tasks yet structured enough to enforce safety. To this end, we leverage *planning with compositional constraints* [51, 74, 106, 177, 261] as a framework

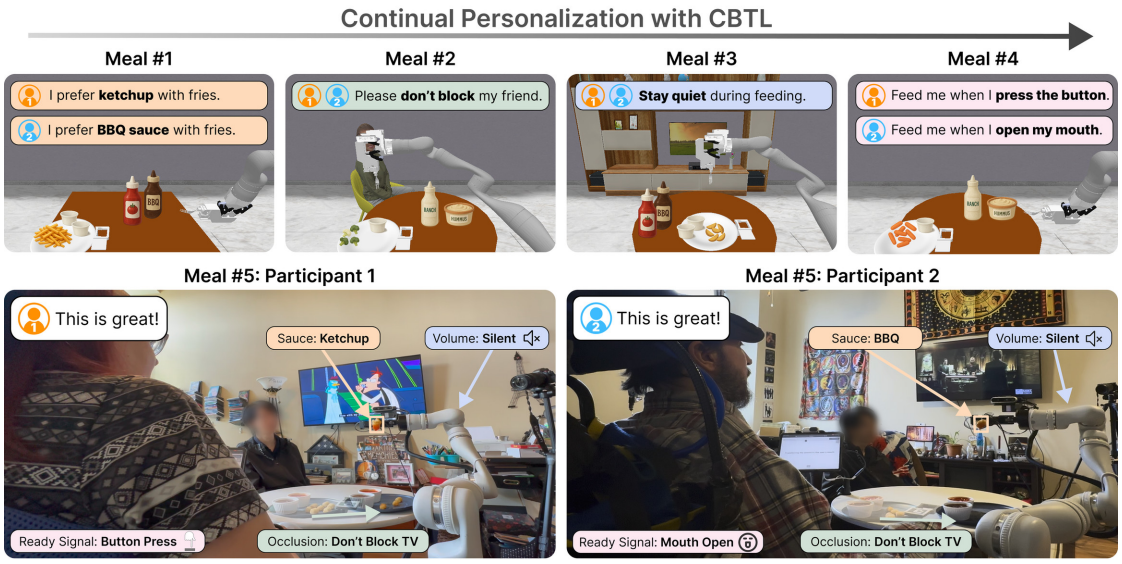


Figure 7.1: Coloring Between the Lines deployed in-home with two care recipients on a mealtime-assistance system, personalizing over five meals (four simulated) in diverse contexts. By the final meal, the robot automatically repositions the plate to respect learned occlusion preferences and correctly predicts food–dip choices, ready signals, and verbal feedback preferences.

to enable a broad range of behavior while adhering to explicit programmer-specified restrictions [268]. For example, consider the mealtime assistance system shown in Figure 7.1. Given a request for a bite of food, the robot uses its given non-personalized capabilities to generate a constraint satisfaction problem (CSP) with variables platePose $\in$ SE(2), utensilAbovePlatePose $\in$ SE(3), robotAbovePlateConf $\in \mathbb{R}^7$, dip $\in$ {ketchup, BBQ sauce, none}, and readySignal $\in$ {mouth open, button, auto}, among others, which are mutually constrained, e.g., to enforce kinematic feasibility and safe collision-free motion.

CSPs generally have multiple solutions, especially when continuous variables are involved. We refer to the set of solutions to a CSP as a *null space* by analogy to the concept in kinematics [138, 195, 240] which refers to the fact that

a manipulator with redundant degrees of freedom (DoF) can move without affecting the end-effector's task, e.g., repositioning the elbow while keeping the hand fixed. Our key insight is that *this CSP null space provides an opportunity for personalization.* Rather than picking arbitrary solutions, the robot should "color between the lines" of the constraints to select solutions in the null space that are personalized to the user. In particular, we propose that robot engineers should not only implement one-size-fits-all constraints to ensure safety and general competency—they should also implement constraints with learnable parameters that can be rapidly personalized in-the-wild.

For example, previous work in assisted feeding has shown that users vary in their bite preferences (e.g., dipping sauce) [121] (Chapter 5), their interaction modes (e.g., ready signals) [122] (Chapter 6), and their sensitivities to visual occlusions (e.g., by the robot arm) [13]. Our approach unifies these and other aspects of personalization that previous work has treated task-dependently. For instance, we generate a dipPreferred(dip, θ_1) constraint where $\theta_1 \in$ Text is a natural language summary of the user's dip preferences, and the constraint itself is implemented with a large language model (LLM) [209]; a similar signalPreferred(readySignal, θ_2) constraint where $\theta_2 \in$ Text; and an occlusionFree(conf, θ_3) constraint, where $\theta_3 \in [0, 1]$ encodes the user's occlusion sensitivity and where the constraint itself uses a model from prior work [13]. As the robot gathers experience, it adapts the parameters $\theta_1, \theta_2, \theta_3$ and continually personalizes.

To maintain generalizability of preferences across tasks, it is essential that the robot learns full personalized constraints and does not simply memorize CSP solutions. For example, plate and drink positions that satisfied the user in the past

may become impossible if the table changes or if obstacles are introduced. We therefore propose to *actively learn* personalized constraint parameters using an entropy-based method [156, 241]. This enables the robot to rapidly learn personalized constraints that generalize across scenarios.

We call this overall framework—planning with compositional constraints, learning personalized constraint parameters, and actively gathering data—*Coloring Between the Lines (CBTL)*. In experiments across various assistive robots and tasks, we evaluate the extent to which CBTL achieves efficient and effective personalization. We first compare CBTL to four baselines across three simulation tasks—adapting meal preparation to user preferences, selecting preferred surfaces to tidy, and handing over preferred books within a care recipient's range of motion—and find that CBTL consistently personalizes more effectively and efficiently. We then evaluate a mealtime-assistance system in a web-based study with 60 participants, each assessing the system across five simulated meals, and find that participants significantly prefer CBTL's choices over a non-personalized baseline.

Most importantly, we conduct an in-home case study of CBTL on the same mealtime-assistance system with two care recipients with mobility limitations, who evaluate the system over five meals—four of which use screen-based training in which the participant views first-person renderings, with only the fifth involving physical feeding. We treat this as a case study, intended to surface qualitative feasibility insights with a population that is otherwise difficult to recruit at scale, rather than as a statistical validation. Within this scope, both users prefer CBTL because it reduces the workload of specifying preferences from scratch for each new meal, while also correctly predicting their preferences across food

and dip combinations, ready signals, verbosity, and occlusion sensitivity. In a separate real-world demonstration, we also observe transfer: by reusing a personalized occlusion constraint learned from feeding, it anticipates new occlusions when a drink is introduced, repositioning both the plate and the drink to avoid collisions and maintain reachability. We report this qualitatively, as an exploratory finding. Together, these results show that CBTL offers a unified and practical framework for incremental, active, and safety-preserving personalization of assistive robots.

A note on terminology before proceeding. We use *continual*, *generalizing*, and *safe* in specific senses throughout this chapter. *Continual* refers to incremental online updates of personalized constraint parameters during interaction, without environment resets; we do not address the catastrophic forgetting and preference-drift concerns that are central to continual learning in the broader machine-learning sense [43, 165]. *Generalizing* refers to the transfer of learned preference parameters across tasks within a fixed family of programmer-specified constraint templates, rather than the autonomous discovery of new constraint types. Our evidence for this transfer is exploratory and secondary to the chapter's core contribution, which is incremental, active, safety-preserving personalization. And *safe* refers to the guarantee that programmer-specified universal constraints—such as collision avoidance—are never violated, contingent on those constraints being correctly authored. We discuss the implications of each of these scopes in §7.5.

We summarize our contributions as follows:

Table 7.1: Comparison with prior work. Entries use the senses of *incremental*, *generalizing*, *active*, and *safe* defined in §7.1, not all interpretations in the broader literature; "partial" marks a related but narrower capability, as noted below. CBTL is *incremental*, as it learns online without environment resets; *generalizing*, through compositional adaptation; *active*, via maximum-entropy exploration; *safe*, since universal constraints are never violated; and uniquely evaluated through an *in-home case study* with two care recipients with mobility limitations.

Framework / Method	Incremental	Generalizing[a]	Active	Safe[b]	In-Home Case Study
INQUIRE [65], Bıyık et al. [19], Vats et al. [266]	partial[c]	partial[d]	✓	✓	✗
Lee et al. [151], Leyzberg et al. [155], Churamani et al. [43], Spaulding et al. [252]	✓[e]	✗	✗	✓	✗
Liu et al. [173], Hellou et al. [99], He et al. [98], Wang et al. [272]	✗	✓	✗	✓	✗
Karagulle et al. [132], Mukherjee et al. [193]	✗	✗	✓	✓	✗
Thumm et al. [261], Wang et al. [269]	partial[f]	✓	✓	✓[f]	✗
CBTL (ours)	✓	✓	✓	✓	✓

[a] Reuse of learned parameters within pre-defined constraint templates, not discovery of new constraint types; our own evidence is exploratory (§7.4.4). [b] No constraint violations beyond those the underlying planner already permits; not new verification, monitoring, or formal guarantees. [c] INQUIRE adapts continually, but within a single task. [d] Reward-learning transfers one latent reward across tasks, but not heterogeneous preference categories. [e] These works additionally address preference drift, forgetting, and affect-aware modeling, which CBTL does not. [f] APRICOT updates beliefs within an interaction but not across sessions; Text2Interaction obtains safety through an added formal verification step rather than through the constraints alone.

- **Coloring Between the Lines:** A framework for incremental, active, and safety-preserving robot personalization that leverages the null space of constraint satisfaction problems (CSPs).
- **Evaluations across diverse tasks:** Simulated experiments on meal preparation, surface cleaning, and book handovers show that CBTL consistently personalizes more effectively and efficiently than various baselines.

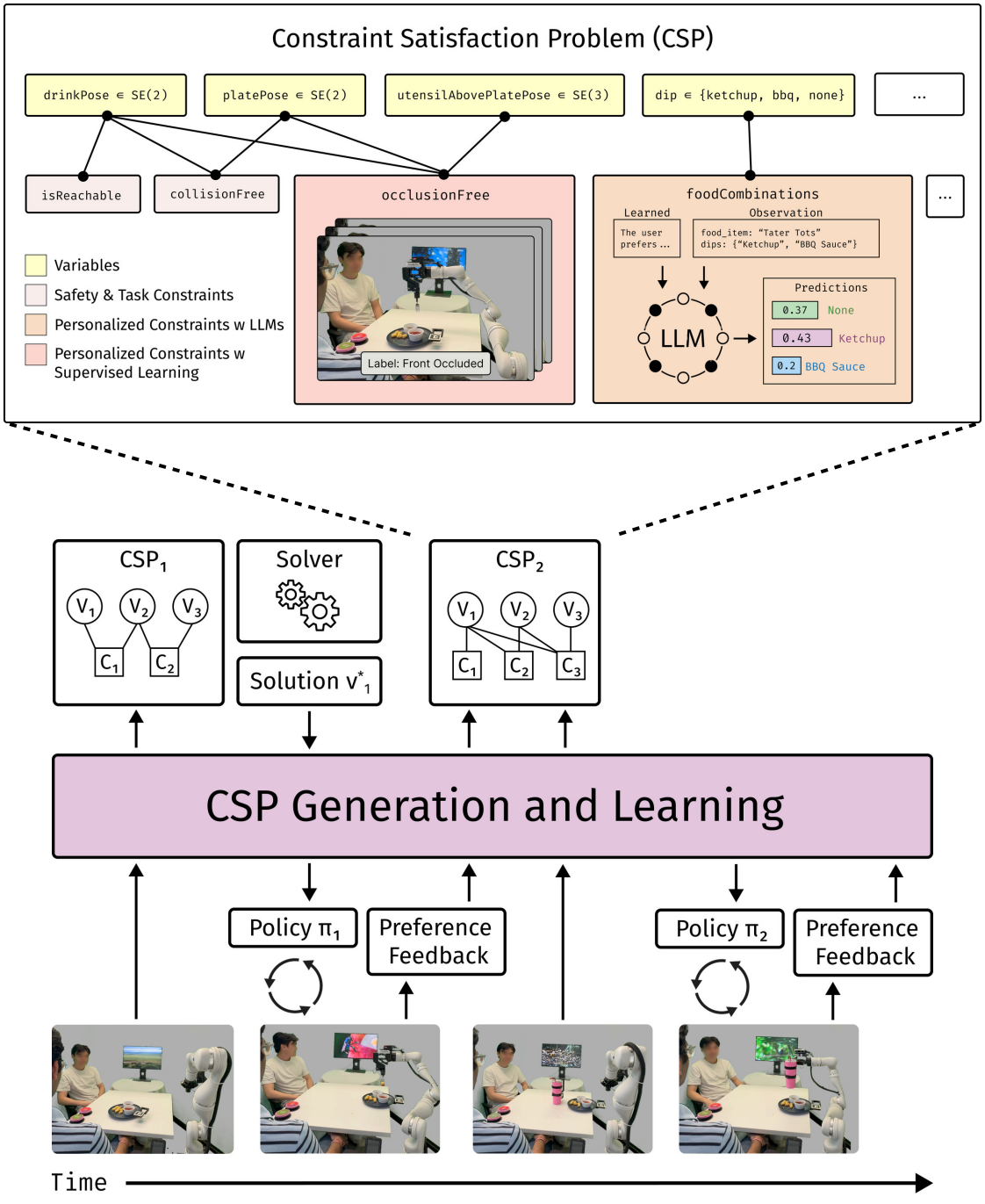
- **User study with 60 participants:** A web-based study in a mealtime-assistance setting demonstrates that users significantly prefer CBTL's choices over a non-personalized baseline.
- **In-home case study:** A case study with two care recipients—who train the system primarily through screen-based simulation and complete one meal in physical interaction—suggests that CBTL reduces user workload and accurately predicts preferences across food–dip choices, ready signals, verbosity, and occlusion sensitivity.

7.2 Constraint Satisfaction Problem Formulation

We consider a robot that takes actions $a \in \mathcal{A}$ and receives observations $o \in O$ in an environment that is governed by unknown transition and observation models. The environment is never reset—the robot has a single trajectory of experience. Let $h_t = (o_0, a_1, o_1, \ldots, a_t, o_t)$ denote the history of observations and actions through time t. Across tasks, the robot maintains an observation state that may include proprioception, symbolic task state (such as object poses or task progress indicators), and natural-language descriptions of the task context. In experiments, we measure the performance of the robot according to a domain-specific objective over histories $J(h_t) \in \mathbb{R}$. For example, in assistive feeding, we use a 5-point Likert scale to measure overall user satisfaction. Unlike in reinforcement learning and related paradigms, we do not assume that the robot can observe $J(h_t)$; the objective is only used to externally evaluate the robot's behavior in experiments.

Decision-Making with CSPs. We propose that robots in-the-wild should make decisions by repeatedly generating and solving constraint satisfaction

(a) Constraint Satisfaction Problem (CSP): CSP_2 of (b), expanded (dashed lines)



(b) CSP Generation and Learning

Figure 7.2: **Overview of Coloring Between the Lines (CBTL)**, our proposed method for robot personalization.

problems (CSPs). A CSP $(\mathcal{V}, \mathcal{C})$ is defined by variables $V \in \mathcal{V}$ and constraints $C \in \mathcal{C}$. Each variable V has a unique name $N(V)$ and a domain $D(V)$ that may be, for example, a real-valued vector space, a finite set, the set of all natural language strings, or a space of object poses. A constraint C is defined over one or more variables $V_1, \ldots, V_k$ via a function $c : D(V_1) \times \cdots \times D(V_k) \to \{\text{True}, \text{False}\}$. A CSP solution $\mathbf{v}^*$ is a joint assignment of all variables to values that satisfies all constraints. There are many techniques for solving CSPs. In this work, we use a simple sampling-based solver that performs a random walk over candidate solutions starting from an initialization produced by the CSP generator. At each step, the solver applies a sampler that perturbs a subset of variables and checks all constraints; the first candidate that satisfies all constraints is returned. The solver assumes only cheap constraint evaluations and a reasonable initialization, with no smoothness or convexity requirements. Full details appear in Appendix D.2.

Generating CSPs. We propose that robots should be equipped with *CSP generators*. A CSP generator CSPGen takes a history h_t as input and produces (1) a CSP, (2) a solution-conditioned policy $\pi(a_t \mid h_t, \mathbf{v}^*)$, and (3) a solution-conditioned termination condition $\tau(h_t, \mathbf{v}^*)$. After finding a solution $\mathbf{v}^*$ to the CSP, the robot takes actions following the policy until the termination condition is satisfied, at which point the next CSP is generated. If no CSP solution is found, a default fallback policy is used. This work is agnostic to the implementation of the CSP generator, but in practice, we take inspiration from task and motion planning (TAMP) methods such as PDDLStream [73].

Robot policy and fallback. The solution-conditioned policy $\pi(a_t \mid h_t, \mathbf{v}^*)$ composes environment-specific skills (for example, moving to a shelf, picking

an object, moving to the user, and handing over) using the variables in $\mathbf{v}^*$ as waypoints and targets. The termination condition $\tau(h_t, \mathbf{v}^*)$ checks for skill completion events or a new user request, at which point the next CSP is generated. If the CSP with personalized constraints is infeasible within a generous timeout, CBTL falls back to re-planning without personalized constraints, which guarantees that the robot can still execute a conservative, non-personalized solution.

7.3 Coloring Between the Lines

We now present Coloring Between the Lines (CBTL), our approach for incremental, active, and safety-preserving robot personalization. See Figure 7.2 for an overview of the approach.

The CSPs generated by the robot will typically have many solutions. Our key insight in this work is that this *null space* provides an opportunity for personalization. The robot should "color between the lines" by generating and adapting personalized constraints that further reduce the null space.

Scope of the Safety Claim. All safety in this framework comes from the CSP generator: CBTL neither authors, verifies, nor monitors the programmer-specified constraints C. Personalized constraints are only ever *added* to C, so personalization can shrink the solution set—in which case the robot re-plans without them (§7.2)—but can never admit a solution that violates a programmer-specified constraint. CBTL thus preserves the safety properties of the generator it is given; it adds no verification, runtime monitoring, or formal guarantee.

The physical system used in our real-robot experiments (§7.4.3) adds runtime safety measures below the level of the CSP: user- and experimenter-held

emergency-stop buttons, a watchdog over the sensors and stop buttons, force thresholds from a six-axis force/torque sensor behind the compliant silicone fork, and a fork holder with a mechanical weak point that yields under excessive force. For inside-mouth bite transfer, which one of our two participants requires, the robot switches to a Cartesian-space compliant controller that continuously tracks the user's head pose within limits tailored to that user's range of motion; torque-based collision monitoring covers the motions where compliant control is inactive. These measures belong to the hardware and low-level controller, not to CBTL.

Generating Personalized Constraints. Given a CSP $(\mathcal{V}, \mathcal{C})$ produced by CSPGen from history h_t, CBTL augments it with a set of *personalized constraints* $\mathcal{C}_p$, yielding $(\mathcal{V}, \mathcal{C} \cup \mathcal{C}_p)$. The solution to this CSP is then used to condition the policy and termination condition as before. Personalized constraints $C_p \in \mathcal{C}_p$ are produced by a collection of generators $\textsc{Gen}_p^1, \ldots, \textsc{Gen}_p^n$ to enable compositional generalization. Each generator $\textsc{Gen}_p^i$ takes the original CSP $(\mathcal{V}, \mathcal{C})$ and the history h_t as input and returns a (possibly empty) set of personalized constraints $\mathcal{C}_p^i$. The final set of personalized constraints is the union $\mathcal{C}_p = \bigcup_{i=1}^n \mathcal{C}_p^i$.

Each generator $\textsc{Gen}_p$ is characterized by an *arity* m, an *initiation condition* ι_p, and a *parameterized constraint function* c_θ. The arity refers to the number of variables that the constraint function expects as input. The initiation condition takes in a tuple of m variables from $\mathcal{V}$ and determines based on their names and domains whether or not a constraint should be produced for those variables. The

function c_θ is used to define the constraints. The parameters θ are constant during generation but updated through learning (see below). All together, the generator outputs:

$$\{((V_1, \ldots, V_m), c_\theta) : \iota_p(V_1, \ldots, V_m) = \text{True}, V_i \in \mathcal{V}\}.$$

Learning Personalized Constraints. CBTL uses the history h_t collected by the robot to update personalized constraint parameters θ for each generator $\textsc{Gen}_p$. We consider two types of learning.

Supervised Classification. Our first approach uses the history of experience to derive a classification dataset $\mathcal{D}$ with elements of the form $(\mathbf{v}, \ell)$, where $\mathbf{v}$ is an input to the classifier c_θ and ℓ is True or False. The logic for deriving these inputs and outputs varies per constraint. Preference queries are issued only at natural decision boundaries, typically after executing a CSP solution. The resulting feedback is added to h_t and incorporated into $\mathcal{D}$ when updating the relevant constraint. Given the (initially empty) dataset, standard techniques can be used to optimize the parameters θ (see Appendix D.2). Many personalized constraints reduce to checking whether a scalar feature $f(v)$ lies within a learnable interval $[\theta_0, \theta_1]$. In these cases, updating $\theta = (\theta_0, \theta_1)$ amounts to solving a simple likelihood-based optimization over the accumulated positive and negative examples in $\mathcal{D}$.

Learning with Natural Language. Our second approach similarly uses history to derive a dataset $\mathcal{D}$, but with elements of the form $(\mathbf{v}, o, w)$ where o is an observation and w is a natural-language string. These data are used to learn constraints that are implemented with large language models (LLMs). In particular,

we consider constraints of the form

$$c_\phi(v_1, \ldots, v_m) = \textsc{QueryLLM}(\textsc{ConstraintPrompt}(v_1, \ldots, v_m, o, \phi))$$

where ϕ is a natural-language string[1] and ConstraintPrompt is a constraint-specific function. A second constraint-specific function LearningPrompt is used to update the parameter constraints:

$$\phi \leftarrow \textsc{QueryLLM}(\textsc{LearningPrompt}(\mathcal{D})).$$

Each ConstraintPrompt includes (i) the candidate solution, (ii) relevant observations, and (iii) the current natural-language parameter ϕ. Each LearningPrompt summarizes the accumulated dataset $\mathcal{D}$ (e.g., recent labeled outcomes) and asks the LLM to update ϕ. As with supervised constraints, the robot can ask a brief clarifying question at natural decision boundaries (e.g., after a bite request or handover). The user's response is appended to $\mathcal{D}$ and used to update ϕ. See Appendix D.3 for LLM constraint generation and learning details and prompt templates.

Active Learning for Data-Efficient Personalization. Just as the null space enables personalization, it also enables exploration—the robot should choose actions within the null space that lead to efficient learning (Figure 7.3). We propose a maximum-entropy-based approach [156, 241]. When the robot is permitted to explore, rather than treating C_p as personalized constraints that need to be satisfied, we create a mathematical program

$$\max_{\mathbf{v}} \frac{1}{|C_p|} \sum_{C_p \in C_p} \mathcal{H}(C_p(\mathbf{v}))$$

$$\text{subject to } C(\mathbf{v}) = \text{True } \forall C \in \mathcal{C}$$

[1]Natural-language constraint parameters are written as ϕ to avoid overloading the numeric parameter symbol θ.

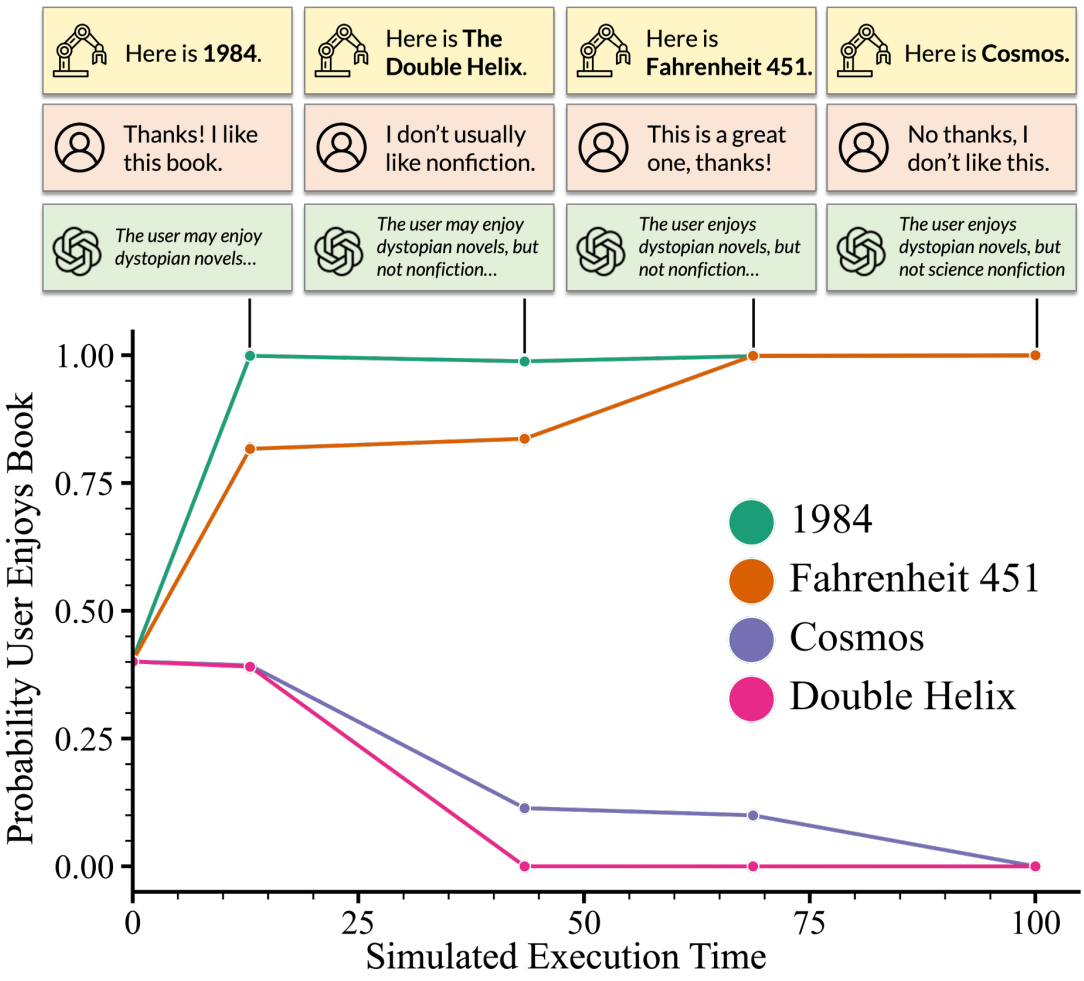


Figure 7.3: CBTL actively reduces uncertainty and generalizes from feedback.

where $\mathcal{H}$ is the entropy of the Bernoulli distribution $P(C_p(\mathbf{v}) = \text{True})$. Thus, the robot will choose CSP solutions that maximize its uncertainty about the personalized constraint parameters that it is learning, while still satisfying all non-personalized constraints to guarantee safety and competency. The reading of this objective depends on where $P(C_p(\mathbf{v}) = \text{True})$ comes from. For supervised classification constraints, it is the predictive probability of a classifier with an explicit posterior, and the entropy carries its standard information-theoretic meaning. For LLM-based constraints, we derive it from a 0–10 LLM rating normalized to $[0, 1]$ (Appendix D.3), which we treat as an *approximate confidence value* rather

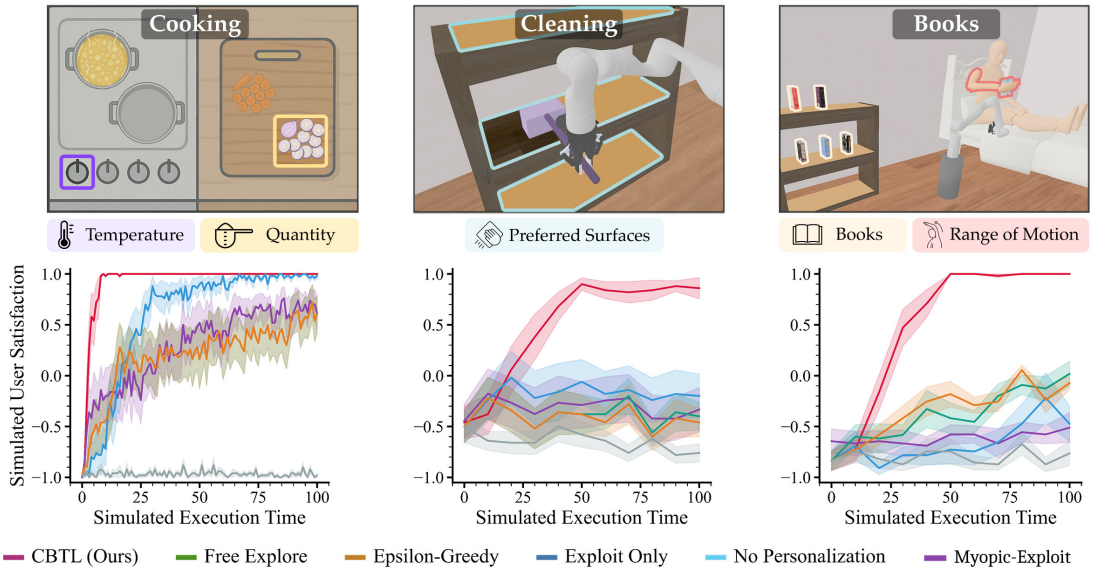


Figure 7.4: Simulation experiments. Lines are means and shaded regions are std. errors over 10 seeds. Units are deliberately omitted (time is arbitrary in simulation and user satisfaction is environment-specific). In Cooking, Exploit Only and Epsilon-Greedy obtain equal performance; see Appendix D.7 for discussion. CBTL consistently personalizes better and faster than baselines.

than a calibrated posterior. We examine that approximation empirically in Appendix D.5 and ablate it in Appendix D.4, and discuss the residual risk of being confidently wrong in §7.5.

7.4 Experiments and Results

We evaluate *Coloring Between the Lines (CBTL)* across diverse simulation tasks, a large-scale web-based user study with 60 participants in a mealtime-assistance scenario, and an in-home case study of the same system with two care recipients. Across all experiments, CSPs are generated from structured symbolic observations (e.g., object poses, meal context variables, and user-interface inputs); no raw images are used for learning. Actions correspond to invoking high-level

skills (e.g., move-to-pose, pick, place, handover) with targets specified by the CSP solution. In addition to assessing the overall *effectiveness* and *efficiency* of CBTL, we consider the following questions:

1. To what extent is *active learning* important for sample-efficient personalization? (Section 7.4.1)
2. To what extent is CBTL able to personalize to *real human preferences*? (Section 7.4.2)
3. To what extent does CBTL show feasibility in an in-home case study with care recipients? (Section 7.4.3)
4. As an exploratory question, to what extent can CBTL *rapidly generalize* during continual personalization? (Section 7.4.4)

Moreover, we also investigate scenarios such as unsatisfiable user preferences and nonstationary user preferences; results for these additional experiments are presented in Appendix D.6.

7.4.1 Simulation Experiments

We first compare CBTL to four baselines in three diverse simulation environments. All baselines are given the same prior knowledge in the form of CSP generators. For each approach in each environment, we run 10 random seeds. We evaluate performance by loading checkpoints and running held-out tasks with simulated user satisfaction metrics. For each checkpoint, we evaluate on 10 held-out tasks per environment and report cumulative user satisfaction, where each task contributes a scalar score defined by the hidden user preferences. Figure 7.4 plots the mean cumulative satisfaction across 10 seeds, with shaded regions indicating standard errors. We now briefly describe the environments and

baselines with details in Appendix D.7. We also release code for reproducing all results in our supplementary materials[2].

Environments. We begin with a brief description of the simulation environments, along with a summary of the task-level metric used to compute simulated user satisfaction.

- **Cooking**: a 2D stove-top environment that features *pots* with continuous positions and sizes; *ingredients* with continuous quantities and temperatures; a *user* with unknown preferences. The robot must move pots onto the stove, put ingredients in the pot, wait for them to heat, and then serve the dish to the user. When served a meal, the user gives feedback about ingredient quantity and temperature. CSP non-personalized constraints include: pots cannot overlap on the stove; ingredients can only be used in the quantities available; ingredients must comprise some known meal. A learned personalized constraint enforces that the user enjoys the served meal. A held-out task receives a score of +1 if all hidden temperature and quantity preferences are satisfied when the meal is served and −1 otherwise.
- **Cleaning**: a PyBullet [49] environment where a Kinova Gen3 7-DoF arm [141] on a holonomic mobile base must use a dusting tool to clean tables and shelves. A user watching has unknown preferences about which surfaces should be cleaned and admonishes the robot if it violates preferences. CSP non-personalized constraints enforce kinematic feasibility and collision-free motion for moving the robot base, grasping the dusting tool, and positioning the tool to wipe. A learned personalized constraint enforces that only

[2] `https://emprise.cs.cornell.edu/cbtl/`

preferred surfaces are cleaned. A held-out task receives +1 if exactly the preferred surfaces are cleaned and no non-preferred surface is wiped, and −1 otherwise.

- **Books**: a PyBullet [49] environment where a Kinova Gen3 7-DoF arm [141] on a holonomic mobile base must fetch books from a shelf and hand them over to a user in bed. The user has unknown book preferences and an unknown functional range of motion [173], outside of which handovers fail. CSP non-personalized constraints enforce kinematic feasibility and collision-free motion for moving the robot base, grasping and placing books, and handing over books. Learned personalized constraints enforce reachable handovers and book choices the user prefers. A held-out task receives +1 if the handover both lies within the user's functional range of motion and delivers a preferred book, and −1 otherwise.

Baselines. We now briefly describe the baselines used in the simulation experiments.

- **No Personalization**: A naive baseline where personalized constraints are never updated.
- **Free Explore**: A baseline that explores by fully excluding all personalized constraints during CSP generation. Personalized constraints are still learned and used during evaluation.
- **Exploit Only**: A baseline that always enforces the learned personalized constraints during both exploration and evaluation.
- **Epsilon-Greedy**: A baseline that uses Free Explore ϵ-percent of the time and Exploit Only otherwise, similar to epsilon-greedy exploration in reinforcement learning.

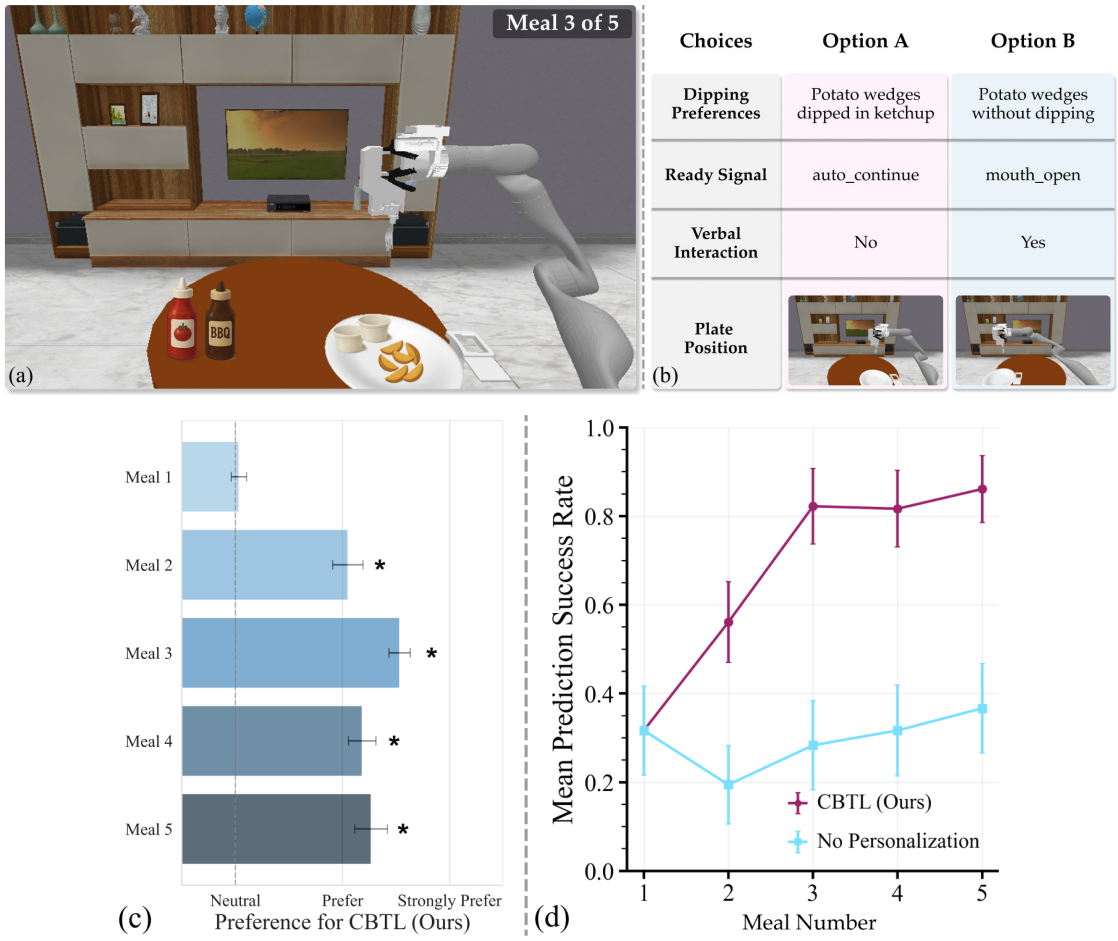


Figure 7.5: **Web-based user study.** (a) Example meal. (b) The user rates their preference on a 5-point Likert scale. (c) Users significantly ($p < 0.005$) prefer CBTL after the first meal. (d) CBTL predicts user responses with increasing accuracy over time.

- **Myopic-Exploit**: An exploit-only baseline that estimates personalized constraints using only the most recent feedback sample, rather than the full history.

Results and Analysis. In Figure 7.4, we find that CBTL consistently personalizes better and faster than baselines across all three environments. For example, in Books, the Exploit Only baseline overfits to the first book and handover pose that satisfy the user, failing to learn about the user's full preferences and

range of motion. Free Explore does not overfit, but its exploration is unguided, leading the robot to waste time. Epsilon-Greedy balances exploration and exploitation, but it is also unguided. Overall, these findings confirm the importance of entropy-based exploration in CBTL. A further ablation in Books isolates the LLM-derived term within the entropy objective. It places the benefit in the entropy objective rather than in the availability of LLM preference estimates, and finds the LLM-derived term itself positive but not separable from noise at this sample size. Appendix D.4 reports the full comparison.

Generalization of Personalized Constraints (exploratory). We also probe, as a secondary question, whether learned parameters transfer across tasks. We constructed a cross-domain task that combines elements from the simulated environments. After learning a user's seasoning preference in Cooking and their functional abilities in Books, the robot is placed in a new scenario where the user is seated in a wheelchair, eating a store-bought meal, and requests seasoning. The robot must both select the correct seasoning and perform a reachable handover, while contending with new physical constraints introduced by the wheelchair, such as armrests that block certain grasps. We compare two conditions: (1) CBTL initialized with the preferences learned in the earlier environments and (2) CBTL trained from scratch. The initialized version adapts more quickly and identifies the correct seasoning and reachable handover region in fewer interactions, illustrating how learned constraint parameters can accelerate personalization in new contexts. Demonstrations are available on our project website[3]. This comparison is qualitative, and we report it as an exploratory observation.

[3] `https://emprise.cs.cornell.edu/cbtl/`

7.4.2 Web-Based User Study

We next conduct a web-based user study to evaluate the extent to which CBTL can personalize to real human preferences. We briefly describe the study and give additional details in Appendix D.8.

Study Design. We consider the assisted feeding scenario introduced in Section 7.1. 60 participants completed the study, ranging in age from 19–68 (mean 31), with 36 identifying as male and 24 as female; 29 reported prior robot experience and 13 reported having been fed as adults. Through a custom website (see Appendix D.8), they were then presented with a sequence of five meals:

- *Meal 1:* Alone at a rectangular table with french fries and ketchup and BBQ sauce.
- *Meal 2:* With friend at a circular table with raw vegetables and ranch dressing and hummus.
- *Meal 3:* With TV at a circular table with potato wedges and ketchup and BBQ sauce.
- *Meal 4:* Alone at a circular table with carrot sticks and ranch dressing and hummus.
- *Meal 5:* With friend and TV at a rectangular table with tater tots and ketchup and BBQ sauce.

For each meal, the robot must choose (1) the plate pose, which affects the above-plate utensil pose and robot joint positions; (2) the dip, if any; (3) the ready signal (button, auto-continue, or mouth open); and (4) whether to speak to the user verbally. We compare CBTL and No Personalization, which is nontrivial here: it does not know if the user prefers ketchup or BBQ sauce for fries, but it

does know that peanut butter would be unusual, given its LLM access. Before each meal, participants see CBTL and No Personalization predictions labeled "Option A" or "Option B," with the label–method correspondence randomized each meal, and rate their preference on a 5-point Likert scale. The participant is then asked about their ideal dip, ready signal, and verbosity, as well as whether their view would be occluded in different egocentric images rendered from the CBTL-selected plate pose and robot joint positions. CBTL learns from this feedback and adapts for subsequent meals.

Results and Analysis. In Figure 7.5(c), we see that participants significantly prefer ($p < 0.005$, Wilcoxon signed-rank test) CBTL for every meal after the first. In Figure 7.5(d), we also see that the choices made by CBTL increasingly align with the user's reported choices for each categorical option (dip, ready signal, verbosity). Together, these results show that CBTL can rapidly and continually personalize to real humans. See Appendix D.8 for additional analysis. We also reuse the web-study feedback to measure whether the LLM-derived probabilities in the active learning objective are calibrated. They are not: the raw scores run systematically high, though their ordering remains informative, and a post-hoc isotonic correction recovers calibration. The correction moves most scores across the peak of the entropy curve, which changes which candidates the active learner would query, so we recommend correcting these scores before using them in the objective. Appendix D.5 reports the protocol and the full results.

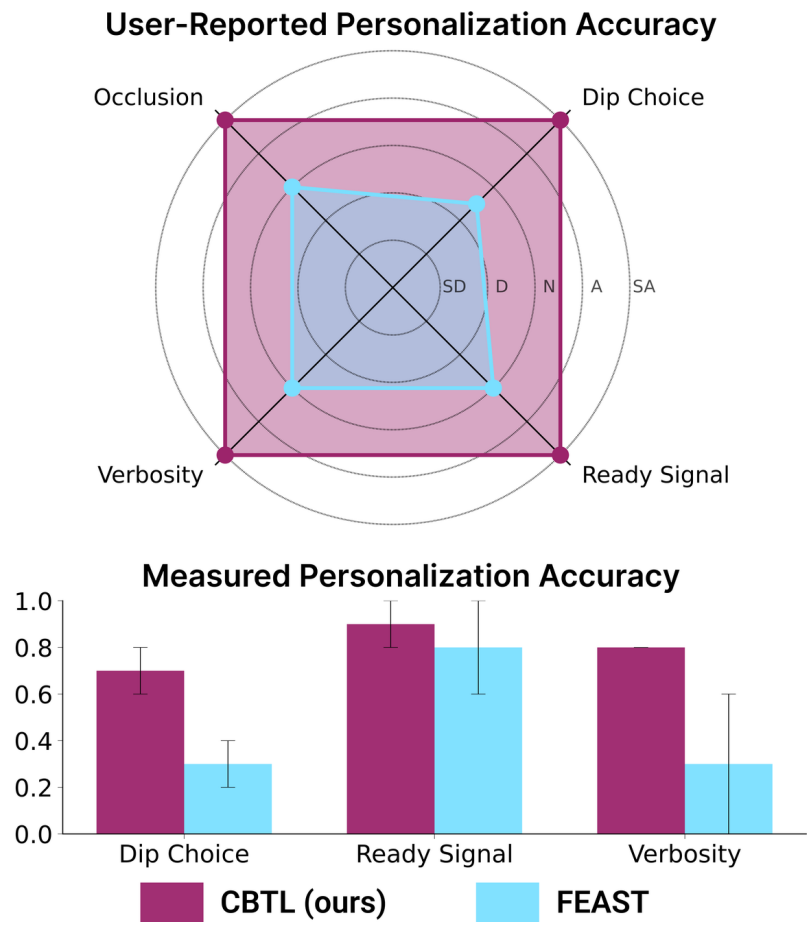
7.4.3 In-Home Case Study with Care Recipients

We next conduct a case study of CBTL with two care recipients using the same mealtime-assistance system (Figure 7.1). We frame this evaluation as a case

study rather than a controlled validation: with $N = 2$ participants, the goal is to surface qualitative insights and feasibility evidence about how CBTL behaves with users who have mobility limitations—a population that is difficult to recruit at scale—rather than to produce statistical claims that generalize to a broader user base. We also note upfront that, of each participant's five meals, four (Meals 1–4) used screen-based training in which the participant viewed first-person renderings rather than interacting with the physical robot, and only Meal 5 involved physical feeding. We discuss the implications of this design choice for ecological validity in §7.5.

System Details. The system consists of a 7-DoF Kinova Gen3 robotic arm [141] mounted on a custom base, equipped with an Intel RealSense RGB-D in-hand camera [111] and a modified Robotiq 2F-85 gripper [227] with custom fingers. For feeding, the arm picks up a fork-tipped utensil from a tool mount. The utensil has two onboard motors that enable scooping and twirling motions, allowing the robot to dexterously pick up food items and deliver them to the user's mouth. The system can also reposition plates using custom handles. Users interact with the robot via a web interface, which enables preference specification through speech-to-text.

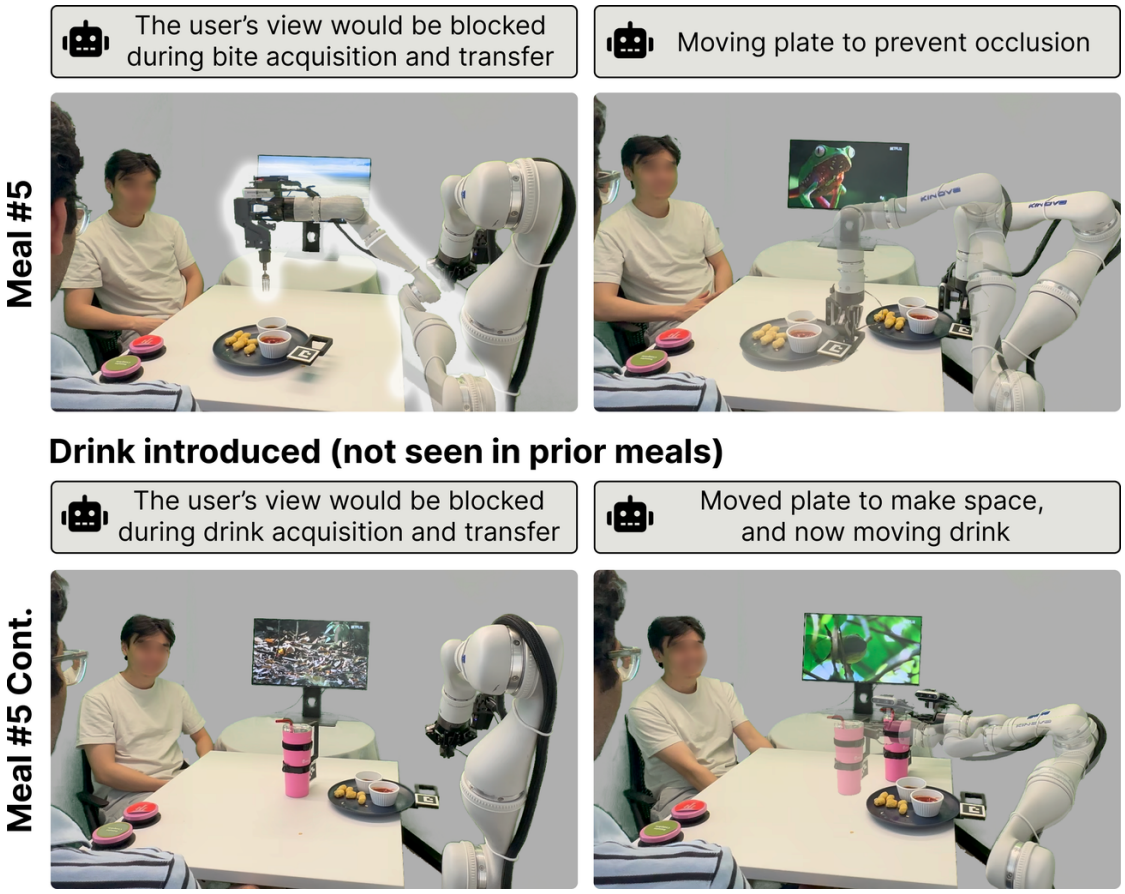
Participants. P1 and P2 are the two community researchers of Chapter 6, CR1 and CR2. P1 is a 48-year-old female with Multiple Sclerosis who retains some arm and neck mobility. She prefers leaning forward for bite transfer and uses her left arm to interact with the web interface. P2 is a 47-year-old male with a C4–C6 Spinal Cord Injury, with limited head and neck mobility, requiring placement of food items inside the mouth. He uses a reflective nose dot to control the cursor via head movements and registers clicks through pauses.



(a) Results from in-home deployment with two care recipients. CBTL outperforms FEAST in both user-perceived and measured personalization accuracy across all categories.

Figure 7.6: Evaluation of CBTL in real-world settings.

Methods. We compare CBTL against FEAST [122] (Chapter 6), a state-of-the-art mealtime-assistance system. The two methods use the same hardware and robot skills; the only difference lies in how preferences are handled. CBTL initializes each meal with personalized preferences inferred from prior interactions, while FEAST always begins from generic defaults. In addition, CBTL can reason over user-preferred occlusion constraints and reposition the plate if necessary, a capability not supported by FEAST.



(b) Generalization of CBTL. After personalizing to a participant's plate occlusion preferences, CBTL generalizes to avoid occlusions due to a previously unseen drink.

Figure 7.6: (continued) Evaluation of CBTL in real-world settings.

Procedure. Each participant, masked to the method, completes two sessions (one with CBTL, one with FEAST), counterbalanced across users. In each session, the participant experiences five meals as described in the web study (Section 7.4.2). For Meals 1–4, the scenarios are simulated: participants view first-person renderings of the meal context and food items on the plate and personalize the robot's choices using the same web interface they would use during real feeding. Meal 5 is conducted in the real world: the participant watches TV in front of them while socializing with a researcher seated to their left, eating tater tots with ketchup and BBQ sauce. Same as the web study, the robot must

predict four preference categories across meals: (1) plate pose (which affects occlusion); (2) dip choice; (3) ready signal; and (4) verbosity. At the end of each session, participants complete a brief questionnaire rating how accurately the system predicted their dip, ready-signal, verbosity, and occlusion preferences. After both sessions, they also answer a forced-choice question indicating which method required less effort to align with their preferences. Full questionnaire items appear in Appendix D.9.

Results. CBTL consistently outperformed FEAST both objectively and subjectively (Figure 7.6a): it achieved higher accuracy in predicting user preferences across the five meals, and participants rated its predictions as more aligned with their actual choices. Importantly, both participants reported that CBTL required substantially less effort to ensure the robot matched their preferences.

Limitations of This Case Study. With $N = 2$ we report no inferential tests. Four of the five meals per condition were screen-based, and each method received exactly one physical feeding meal per participant, too little to characterize robustness under the variability of real mealtimes. Both sessions fell within a short window, so nothing here speaks to longitudinal deployment or to the durability of learned parameters. The study supports feasibility and user acceptance among users who are difficult to recruit at scale; it does not support claims about long-term effectiveness or about the broader population of care recipients.

7.4.4 Generalizing Preferences Across Tasks (exploratory)

We next demonstrate CBTL's generalizability on the same mealtime-assistance system. This is a single-participant, single-episode demonstration with no quantitative measures and no control condition, so we present it as an illustration

rather than as evidence that CBTL generalizes reliably. In addition to assisting with feeding, the robot can also pick up a mug leveraging a custom handle with an attached AR tag, which is affixed to the mug, and bring it to the user for sipping. A study participant completes Meals 1–4 in simulation, and then has Meal 5 in the real world (Figure 7.6b). The plate is initially placed in front. Without any further feedback, CBTL optimizes the plate pose to avoid violating the user's occlusion preferences. After serving a few bites, a drink (which was not in previous meals) is added to the table. The robot reuses the personalized occlusion constraint from feeding and repositions the plate and the drink to avoid occlusions during drinking. The transfer here is parameter reuse within a constraint template the developer already authored; whether it holds across a wider range of tasks and users is an open question this demonstration does not settle.

7.5 Discussion

In safety-critical applications such as robot-assisted feeding, the extreme consequences of violating constraints make it tempting to engineer systems that are entirely inflexible. But users with diverse needs consistently voice the importance of personalization [121, 122, 196]. As we continue to work towards robots that assist with not only feeding, but also cooking, cleaning, and countless other activities, we must develop task-independent personalization techniques that maintain generalizability without sacrificing safety or competency. Coloring Between the Lines in CSP null spaces offers a unified and practical path toward addressing these challenges.

7.5.1 Revisiting Our Core Hypotheses

Our work set out to examine whether CBTL enables incremental, active, and safety-preserving robot personalization, and, secondarily, whether learned parameters transfer across related tasks. Across simulation, a 60-participant web study, and an in-home case study with two care recipients, our findings support the three core properties, with transfer supported only qualitatively, while also revealing important limitations. It is worth separating two kinds of evidence at the outset. The simulation experiments and the web study provide quantitative evidence about effectiveness and sample efficiency under controlled conditions. The in-home case study provides evidence of feasibility and user acceptance with two care recipients, and nothing more: it does not establish long-term effectiveness, robustness across users, or durability of learned preferences. We are careful below to attribute each claim to the evidence that actually supports it.

Incremental. CBTL adapts incrementally during interaction without retraining, enabling online personalization of both parametric and language-derived constraints. We use the term *incremental* rather than *continual* to be explicit about scope: our method updates parameters online without environment resets, but it does not include mechanisms from the continual-learning literature for detecting distribution shift, mitigating catastrophic forgetting, or measuring backward transfer [43, 165]. In particular, if a user's preferences change abruptly (e.g., they begin disliking a previously preferred dip), CBTL has no built-in drift detector and would adapt only at the rate at which new contradicting evidence overrides accumulated history. Detecting changes in user preferences and adapting without manual resets remains an open challenge.

Generalizing (exploratory). The form of generalization CBTL supports is *parameter reuse within pre-specified constraint templates*, rather than the structural discovery of new constraint types. We present it as a secondary, exploratory finding. Learned constraint parameters transfer across related tasks, for example generalizing functional-ability preferences from the books environment and seasoning preferences from the cooking environment to a novel environment, and occlusion constraints from feeding to drinks in the real world. This accelerates adaptation in new settings when CSP generators share structure. Both demonstrations are qualitative, however: neither measures how reliably transfer occurs, across how wide a range of task pairs, or for which users. A systematic sweep of source-target pairs with controls and effect sizes is future work. More fundamentally, CBTL cannot learn preferences that fall outside the constraint families a developer has authored: if a user's preference is structurally novel (e.g., a sensitivity to food texture when no texture-related constraint template exists), the system cannot represent or learn it without a code change. We elaborate on this dependence on pre-specified generators in the System-Level Limitations below.

Active. Maximum-entropy active learning efficiently reduces uncertainty with minimal queries, allowing the robot to refine preferences online, as demonstrated in our simulation experiments against multiple baselines. This formulation is well suited to low-dimensional preference spaces, but future work is needed to scale active learning to richer or more interdependent preferences. It rests on firmer ground for supervised constraints, whose probabilities are classifier posteriors, than for LLM-based constraints, whose probabilities are approximate ratings; Appendix D.5 and the Books ablation of Appendix D.4 characterize the difference.

Safe (preserving, not adding). Because all personalization occurs within the null space of task constraints, safety and task feasibility are preserved throughout learning. CBTL contributes no safety mechanism of its own, however: every safety property comes from the programmer-authored CSP generator, and CBTL's contribution is that personalization does not erode it. However, the extent to which safety and task competency are guaranteed by CBTL depends on the implementation of the CSP generators. This limitation is consistent with prior work on safety [268] and TAMP [74], which similarly assume that constraints are given and correct. Nevertheless, further work is needed on methods that verify safety and competency.

7.5.2 System-Level Limitations

Real-time performance and scalability. CBTL relies on repeatedly solving CSPs, but many personalization updates modify only low-dimensional parameters and act on relatively simple CSPs, allowing the robot to respond quickly in practice. Still, highly reactive tasks or environments with larger symbolic state spaces may require incremental or approximate CSP solving. Scaling CBTL to more complex domains will also require improved tools for composing, debugging, and validating CSP generators.

Reliability of LLM-based constraints. LLM-based constraints provide expressive, contextual evaluations of user behavior and preferences, but they introduce sensitivity to phrasing, occasional noise, and calibration challenges. We mitigated these issues through templated prompts, numeric outputs, and low-temperature inference, but long-term reliability and uncertainty estimation for

LLM-generated constraints remain important directions for future work, especially when paired with active learning. Our active-learning objective treats the normalized 0–10 LLM rating as a Bernoulli probability when computing entropy (Appendix D.3), and a rating of 0.8 carries no guarantee of corresponding to an 80% user-preference frequency. We therefore measured the correspondence directly. The ablation of Appendix D.4 locates the gain in the entropy objective rather than in the availability of LLM preference estimates, which earns the LLM-derived term a qualified place in the objective. The calibration analysis of Appendix D.5 sharpens the remaining concern rather than settling it: the raw ratings run systematically high and score worse than a base-rate predictor, so they should not be read as probabilities, and although a post-hoc isotonic correction restores calibration without disturbing the ranking, it moves most scores across the peak of the entropy curve, so it also reorders which candidates the active learner would query. The principal risk is that the robot can be *confidently wrong*: if the LLM concentrates probability mass on an incorrect outcome, the corresponding entropy is low and the active learner is unlikely to query that case again, slowing or stalling correction. Promising mitigations include post-hoc calibration of LLM scores against accumulated $\mathcal{D}$ (e.g., via Platt scaling or isotonic regression), conformal-prediction wrappers that produce coverage-guaranteed confidence sets, or replacing the LLM scorer with a small calibrated learned model trained on the same data. Our analysis covers one constraint, one domain, and one model, so whether these findings extend further remains open.

Limitations of hand-designed CSP generators. CBTL's generality is bounded by the expressiveness of its CSP generators. Designing these generators requires domain knowledge and engineering effort, and personalization is limited to the parameterized constraints that are made available. Concretely,

this manifests as an inability to learn any preference whose dimension is not already represented by some constraint template. Consider, as a representative failure case, a user who finds certain food textures aversive—perhaps preferring purées over solids, or crunchy over mushy items depending on the meal. Our current mealtime CSP generator does not include a texture-aware constraint template, and so even with extended interaction CBTL cannot learn or act on this preference: a parameterized texture constraint must first be added to the generator by the developer. The same is true for any structurally novel preference dimension—thermal comfort, social-distance norms, spatial laterality, and so on—that falls outside the authored templates. Once such a template is in place, however, parameter learning over its values is fully automatic.

The engineering cost of extending the generator depends substantially on what is being added. Authoring an additive constraint template that fits the existing infrastructure—for example, a new LLM-scored preference, or a scalar-feature constraint analogous to our occlusion-sensitivity parameter—typically requires on the order of minutes to a few hours of work: the developer specifies the constraint's input variables, a parameterized scoring function, and (where relevant) an LLM prompt template, and the existing active-learning loop then applies to the new constraint without modification. By contrast, building a generator from scratch for a fundamentally new task or robot embodiment—introducing new variables, kinematic and collision constraints, and skill-level interfaces—can take days or longer. A practical advantage of CBTL's compositional structure is that incremental extensions tend to fall in the former regime: the cross-task generalization shown in §7.4.4, where the personalized occlusion constraint learned during feeding was reused for drink delivery, required no changes to the constraint family itself and only the addition of drink-specific

variables to the CSP. We view this modularity as an important strength of the framework, while also acknowledging that the upfront cost of authoring the initial generator remains a non-trivial barrier. Automatically synthesizing or proposing new constraints from demonstrations or language could expand the range of personalizable behaviors and reduce manual authoring effort.

Ecological validity of screen-based training. In our in-home case study, four of the five training meals per participant used first-person screen renderings rather than physical interaction with the robot. This design choice was a pragmatic response to session-length constraints—each physical meal in our system typically takes 40–60 minutes—but it also abstracts away cues that may matter for the very preferences being learned. Occlusion judgments are a clear example: rendered in two dimensions on a screen, occlusions are perceived without the head-position parallax, depth perception, and shared physical space that shape what a co-present user actually finds visually intrusive. Verbosity and ready-signal preferences may be similarly modulated by physical co-presence (e.g., the perceived loudness or proximity of the robot's voice and movement). Whether the preferences elicited under screen-based training transfer cleanly to extended physical interaction remains an open empirical question, and we view follow-up studies with predominantly physical sessions as the most important next step for validating the case-study findings.

7.5.3 Context Within Existing Personalization Approaches

To clarify where CBTL sits within recent approaches to robot personalization, we qualitatively compare it to representative methods including Text2Interaction [261], APRICOT [269], and reward-learning approaches. For

occlusion preferences, Text2Interaction can interpret commands such as "don't block my view" and replan trajectories, but it cannot actively query users or estimate a continuous preferred field of view. APRICOT learns preferences through Bayesian language queries but cannot refine them in continuous spaces. Reward-learning methods can model visibility preferences but typically merge safety and satisfaction into a single reward, which introduces sample inefficiency and can lead to constraint violations. In contrast, CBTL refines visibility constraints through maximum-entropy active learning, enabling safe online adaptation and reuse across related tasks. For food-and-dip preferences, CBTL uses LLM-based summarization to encode contextual information from prior meals. Text2Interaction must be re-prompted for each meal, APRICOT re-asks clarifying questions for unseen combinations and does not maintain preferences across interaction sessions (not continual), and reward-learning approaches often entangle flavor preferences with unrelated task factors such as collision avoidance. By separating universal task constraints from personalized ones, CBTL isolates, refines, and reuses user preferences across scenarios such as cooking, feeding, and drink handling.

CHAPTER 8

TABLE FOR TWO: CO-ADAPTATION IN A MONTH-LONG IN-HOME DEPLOYMENT

8.1 Introduction

Progress in mobile manipulation [137, 167, 242], physical human-robot interaction [181, 270, 286], and robot learning with foundation models [54, 139, 149, 219, 220] is bringing autonomous robots closer to providing assistance with Activities of Daily Living in people's homes. Yet most in-home deployments of autonomous physical caregiving robots run for a single session or a few days [120, 122, 198]. These reveal whether a system works. They cannot reveal what happens over weeks of real use, when novelty fades [10], contexts diversify, and the relationship between a user and a robot begins to evolve.

This evolution is not one-sided. The robot must track preferences that shift with context and over time. Unlike short-term settings, long-term deployment exposes it to the full diversity of a user's life, demanding efficient personalization across many contexts with minimal workload on the user. Assistive technology that does not fit its user tends to be set aside [218]. At the same time, users accumulate experience with the robot: they learn its capabilities and failure modes, discover when to intervene, and adapt their behavior and environment accordingly. Following prior work on mutual adaptation [203, 213, 265], we call this bidirectional process *co-adaptation*, and study it as a defining feature of long-term human-robot deployment. In prior long-term deployments, however, only one side could change: physical caregiving robots [197] that stayed in homes for

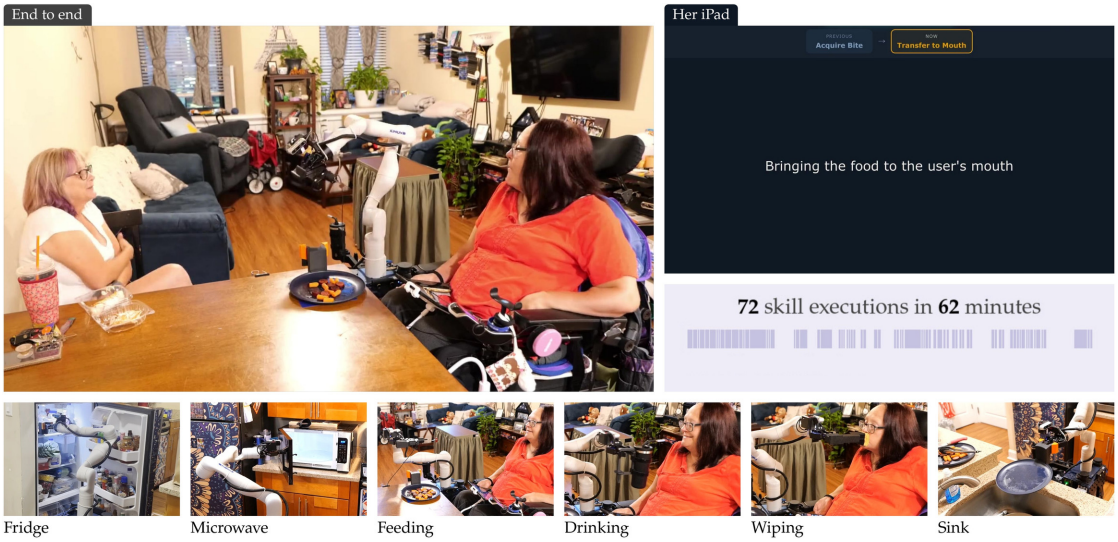


Figure 8.1: **One meal end to end.** Meal 20, a social meal with a friend, run autonomously by the mobile manipulator in the community researcher's home. *Left:* a still of the meal. *Right:* her iPad during a bite transfer, naming the previous and the current skill and describing what the robot is doing. Below it, each bar marks one of the meal's 72 skill executions over 62 minutes. *Bottom:* stills of its six stages (the refrigerator, the microwave, feeding, drinking, mouth wiping, and the sink).

months were teleoperated [12, 37] or ran pre-programmed motions [202], leaving adaptation to the user alone.

In this work, we study co-adaptation through **the first month-long in-home deployment of an autonomous physical caregiving robot**, with a community researcher (CR) who has Multiple Sclerosis and co-designed the system under *community-based participatory research* (CBPR) [89]. Our system operates on a mobile manipulator and executes the **full mealtime-assistance pipeline autonomously**: retrieving a meal from the refrigerator, heating it in the microwave,

placing it on the table, assisting the user with feeding, drinking, and mouth wiping, and dropping the plate in the sink (Fig. 8.1, Section 8.2). A pipeline execution takes about an hour, so preferences surface across many subtasks and failures accumulate. The CR can let a meal run with no input, or intervene at various levels, from confirming an action or correcting a preference to teleoperating the base and the arm. To support the robot's side of adaptation, we propose **a personalization framework** (Section 8.3) grounded in the hypothesis that user preferences are correlated, such that correcting one preference carries signal about the others. A large language model predicts her preferences jointly from her interaction history before each meal and, after each correction, re-predicts those she has not confirmed. Over the month, the CR eats 22 meals with the robot, one per weekday, in various scenarios: alone, with the TV on, with music, with games on a tablet, on video calls with family, and with a friend at the table.

We find that **both sides adapt over the deployment** (Sections 8.4 to 8.6). On the robot's side, her corrections fall to near zero by the second week, resurface as she becomes more comfortable with the robot, starts exploring new scenarios, and her preferences change, and fall again as the robot adapts through this non-stationarity. In a retrospective evaluation on the logs, joint prediction requires 27% fewer corrections than per-dimension prediction. At the final meal, over a video call, one correction leads the robot to revise five other predictions, all matching hers. She adapts as well, changing her own behavior to fit the robot. When talking at social meals confuses its open-mouth detector, she first switches to a button, then changes how she talks: by the final meal she speaks with her mouth slightly open and opens it wide only to request a bite. She learns not to over-exert herself, opening only as wide as the detector needs, and to engage more with her surroundings while the robot works: "I barely noticed the robot, I

Table 8.1: **Deployments of physical caregiving robots.** We situate our work among physical caregiving robots (physically assistive robots [197] that assist with Activities of Daily Living) that meet one of three criteria: deployed for at least a week or five sessions, studied while already in a person's everyday use, or evaluated in-home as an autonomous feeding system.

Assistance provided	Deployment			Change over the deployment	
	Setting	Study duration	Participants	Robot's behavior	Person's behavior
Fetching, scratching, wiping [87, 211, 224, 283]	Home	7–12 days; 3 × 1 week [224]	1 each	Researchers added modules [211, 224, 283]	Control strategy [283]
Grasping, pouring, pressing (person owns the robot) [12]	Home	One 4-h session[a]	7 users	No	Not measured
Feeding, fixed motion [101, 202]	Facility, home	4-day trials [101]; meal observations[a] [202]	12 in trials [101]; 2 long-term users [202]	No	Not measured
Feeding[b] [198, 214]	Home	3 days, 6 sessions [214]; 5 days, 10 meals [198]	1 care recipient each	User edits a setting [198]	Not measured
Feeding, drinking, wiping [122]	Home	5 days, 6 meals	2 care recipients	Language request	Not measured
Whole mealtime (ours)	Home	One month, 22 meals	1 person with MS	Learned from corrections, transfers across dimensions	Cueing, takeovers, lighting, plate

teleoperated or preprogrammed autonomous

[a] Prior use: JACO owned ≥6 months [12]; Bestic used 14 and 5 months [202].

[b] Input per bite: judging the scoop and releasing each step [214]; locating the food [198]. Ours needs input only to switch task.

was looking up only when it was time to take a bite." She comes to anticipate its failures: having noticed that it is sensitive to lighting, she has the lights turned on as a room darkens, and on one day, wearing red, she asks the robot to check its plate detections with her before acting, in case it confuses her with the plate attachment.

Across the month the robot executed 1,335 skills; of the 958 that depend on perception, 93.2% ran without her input and 97.4% with her corrections or teleoperation. Her reported workload falls and her trust rises to the top of the scale; she grades the system's usability an A on the System Usability Scale [25], gives it high Technology Acceptance Model [53] scores, and rates the robot above her human caregiver on independence and control. From the month we draw **four lessons** for long-term deployment of physical caregiving robots (Section 8.8).

8.2 System

Our system is a mobile manipulator that assists with a whole meal in the CR's home: it retrieves the plate from the refrigerator, heats it in the microwave, places it on the table, assists with feeding, drinking and mouth wiping, and drops the plate in the sink (Fig. 8.2). Prior mealtime-assistance systems begin once the plate is on the table [18, 88, 120–122, 198, 214], and a caregiver or researcher still brings the meal and clears it afterwards; ours is, to our knowledge, the first autonomous system to span the meal end to end (Table 8.1). We open-source the code and hardware designs on our website[1].

Hardware. Our system hardware consists of the following.

[1] `https://table-for-two.github.io`

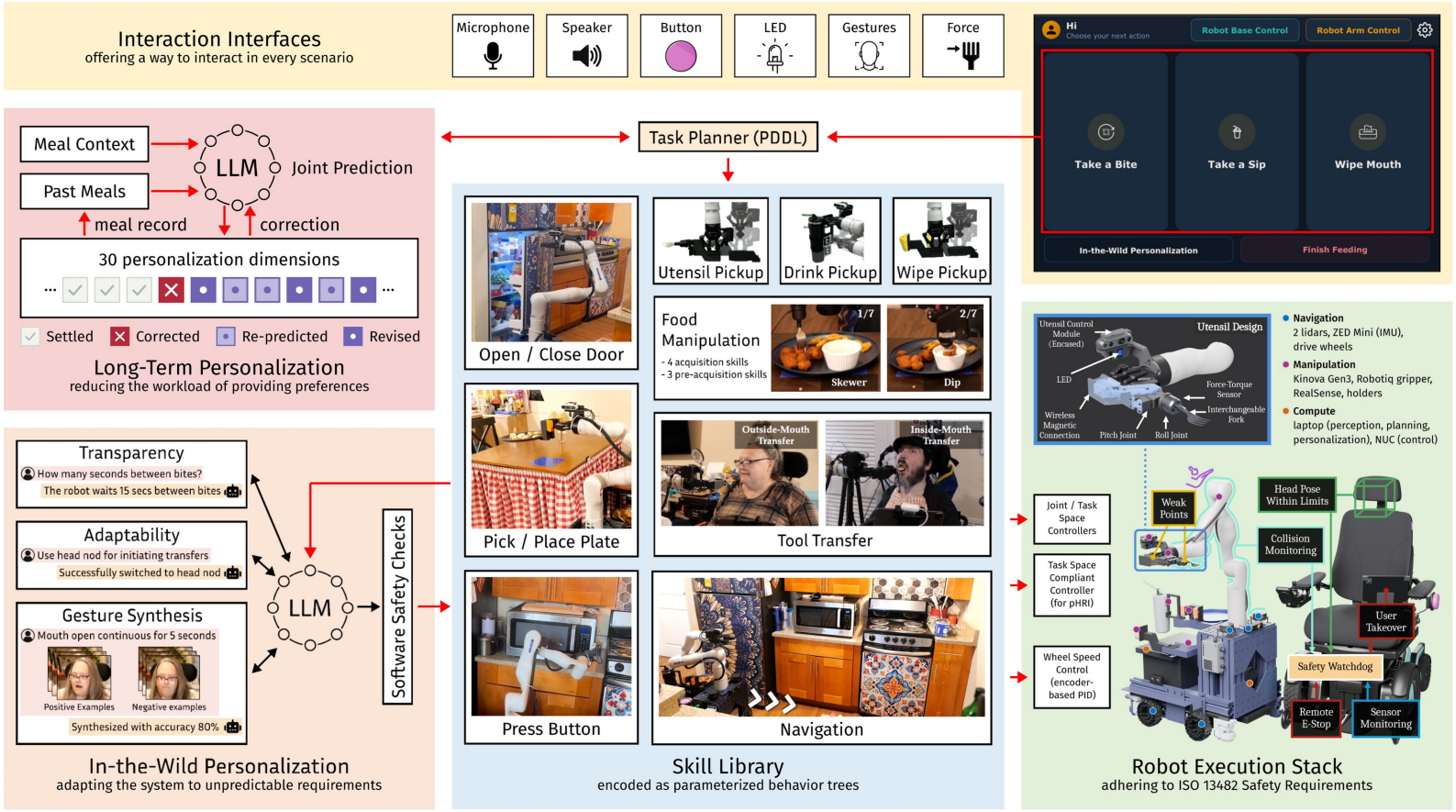


Figure 8.2: **System overview.** *Top:* the interaction interfaces and the task-selection page on the community researcher's iPad. *Left:* long-term personalization, which predicts the 30 personalization dimensions jointly and re-predicts them after each correction (Section 8.3), and FEAST's in-the-wild personalization [122] (Chapter 6), illustrated with examples from the FEAST study. *Center:* the Planning Domain Definition Language (PDDL) task planner and the skill library of parameterized behavior trees. *Right:* the robot's controllers, hardware, and safety features.

Mobile Base. The base is a custom Vention [267] stand carrying the arm, tools and compute. We add two motorized high-grip drive wheels to provide traction and convert the movable stand to a base that can autonomously navigate. An Arduino [5] relays commanded wheel speeds to two motor drivers that hold them under encoder-based proportional-integral-derivative control. Two RPLidar A1 [250] sensors provide 360° 2D lidar for localization and mapping, and a ZED Mini [254] stereo camera with an integrated IMU provides visual-inertial odometry.

Robot Arm. The arm is a 7-degree-of-freedom Kinova Gen3 [141] with a Robotiq 2F-85 [227] gripper and an eye-in-hand RealSense [111] RGB-D camera.

Tool Apparatus. The stand carries a plate holder, a drink holder, and a tool stand holding the feeding and wiping utensils open-sourced by FEAST [122] (Chapter 6); the feeding utensil is instrumented with an ATI Nano25 [8] force/-torque sensor. The plate is fitted with a 3D-printed, microwave-safe polypropylene plate attachment that gives the gripper a rigid, repeatable grasp and docks the plate in its holder. A mug is fitted with a similar drink attachment.

Interaction Interfaces. Besides the web interface described below, the robot cues the CR through an LED on the gripper and a speaker on the base, and she can signal the robot through a wired button at her iPad, for example to request a bite.

Emergency Stop. A remote emergency stop, wired to the researcher's laptop, halts the arm and base.

Compute. A Lenovo Legion Pro 7i laptop [153] with an RTX 4090 GPU runs perception, planning and preference prediction; a separate Intel NUC [112] runs the arm controllers.

Navigation. During the preparation phase we build a 2D occupancy map of the CR's home with Cartographer [103] from the two lidars, and record waypoints for five task locations: the refrigerator, the microwave, the movable table for personal and television meals, the dining table for social meals, and the sink. During deployment, Cartographer localizes against this map. Odometry comes from the wheel encoders and the yaw rate of the ZED Mini's IMU, fused by an extended Kalman filter [191]. Visual-inertial odometry from the ZED's cameras, our initial approach, proved unreliable in the home due to sunlight, reflections and low-texture walls. The Robot Operating System package `move_base` [228] with the Timed Elastic Band local planner [230] commands the base's velocity along collision-free paths it plans to the waypoints.

Manipulation. The manipulation library comprises OpenDoor and CloseDoor for the refrigerator and microwave; PressMicrowaveButton; PickPlate and PlacePlate at the refrigerator, microwave, tables, base holder and sink; PickTool and StowTool for the feeding and wiping utensils; and AcquireBite and TransferTool. The last two use FEAST's food-manipulation skills [121, 122] (Chapters 6 and 5) and head-pose-based transfer [120] (Chapter 4). A PDDL task planner (FastDownward [100]) sequences these with navigation into a valid plan given the current propositional state.

The skills use 3D keypoints as the intermediate representation between perception and action, so that the CR can verify a plan before execution and correct

it if necessary (see Levels of Autonomy below). The plate skills find the plate attachment by color, a per-location environment parameter the robot learns from her corrections (Section 8.3). The other skills use vision-language foundation models: GroundingDINO [171] detects objects such as door handles and the sink basin as open-set detections and SAM [142] refines them into masks, from which the skill takes its 2D keypoint; MolmoPoint-7B [54], prompted in natural language, returns the microwave's start button as a point directly. The skills back-project their 2D keypoints through the RealSense depth map to 3D.

Each skill is implemented as a behavior tree [47] that anchors a waypoint trajectory on its keypoints and on typed parameters, such as `Speed` $\in$ {`slow, medium, fast`}, that the robot predicts from the CR's history before each meal (Section 8.3), and executes it through task-space and joint-space controllers.

User Interface. The web interface is a Vue.js [287] application on the CR's iPad, communicating with the robot over rosbridge [50]. Its pages fall into four groups. *Confirmation pages* precede each step that depends on perception or ungrasps the plate: for plate detection, a color-correction page where she taps the attachment and adjusts the matched range; for the VLM keypoints, a detection page with Looks Correct or Redo; and before each plate release, a confirmation. Each auto-continues after a countdown she can set. *Preference pages* present each predicted dimension at the point it takes effect, with the predicted value preselected, and a settings overlay for revisiting any answered dimension (Section 8.3). *Feeding pages* build on FEAST's open-source interface [122]: task selection, bite selection with a manual keypoint mode, and personalization through language and gestures. *Teleoperation pages* are reachable from any page: the base page moves the robot base with a joystick, and the arm page offers joint-space

and task-space control as well as predefined motions such as retract. While a skill executes, the screen shows the previous, current and next skill and a language description of what the robot is doing. Appendix E.1 shows every page and the order in which they appear over a meal.

Levels of Autonomy. The system supports variable levels of autonomy. The CR can run a meal end to end without input: the robot predicts her preferences and, while executing the whole pipeline, auto-continues through each confirmation page after a countdown. She can also intervene at two levels. *High-level corrections* act on a skill's inputs through the interface: correcting a predicted preference, triggering a re-detection, selecting the plate color, or manually selecting the keypoint for a bite. *Low-level teleoperation* takes over the motion itself: driving the base, for example to reposition it when navigation parks short, or commanding the arm directly to help it recover from a joint limit. She can then have the robot redo or resume the skill it was on, or move on to the next one.

Safety. Two watchdogs run continuously: one on the NUC treats the emergency stop's signal as a heartbeat and halts the arm and base if it stops, whether from a press, a disconnected button or a lost network link; one on the laptop halts the arm if any sensor stops streaming or leaves its expected range. A collision monitor compares the arm's joint torques with a dynamic model and stops motion when the discrepancy exceeds a threshold each skill sets for its phase, from 15 Nm to press the microwave button to 40 Nm to firmly close its door. The feeding utensil and the gripper's 3D-printed fingertips each have a designed weak point that breaks under excess force [122]. Detections are accepted only within expected zones, and every predicted preference is checked against its parameter domain before it takes effect.

8.3 Personalization

Hypothesis. The robot exposes 30 *personalization dimensions*, each a choice such as how fast the arm moves or the cue she uses to request a bite. We hypothesize that they do not vary independently: how fast she wants the robot to move and how much she wants it to check with her respond to the same latent factors, such as who else is at the table, whether she is in a hurry, and how much she trusts the robot. A correction to one dimension is therefore evidence about the dimensions she has not been asked about.

Personalization dimensions. Of the 30 dimensions (Fig. 8.3), 21 are *meal preferences*, drawn from requirements identified in a prior formative study with 21 care recipients [122] (Chapter 6): five dimensions on how much the robot checks with her; eight on how she and the robot signal each other, such as whether she requests a bite by opening her mouth or pressing a button; five on the bite and how it is taken, such as how far the utensil stops from her mouth and the order in which she eats what is on the plate; two on whether the arm retracts between bites and how fast it moves; and one on how long the meal is heated. The remaining nine are *environment parameters* that describe her home, learned from her corrections in the same way: the plate attachment's color at four pickup locations, and at five, a parking offset for the base, taken from the adjustment she makes to its position on arrival. Appendix E.2 lists all 30 with their options.

Prediction and correction loop. The robot predicts all 30 dimensions jointly (Fig. 8.4), with a single call to Claude Opus 4.8 at medium reasoning effort, from a description of the CR's physical functioning, every prior meal's accepted values and context in full, and this meal's context: the food, the setting label (personal,

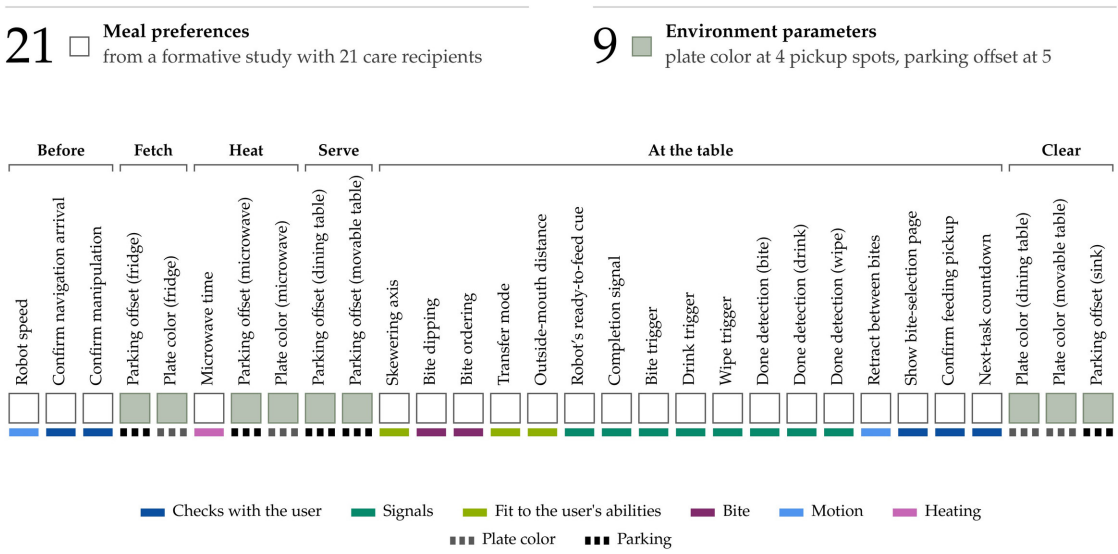


Figure 8.3: **The 30 personalization dimensions, across the whole meal.** The dimensions are ordered along the meal, from before the robot leaves for the refrigerator to clearing the plate. Of the 30, 21 are meal preferences (white), drawn from a formative study with 21 care recipients [122], and 9 are environment parameters (gray): the plate attachment's color at four pickup locations and a parking offset at five. The bar under each dimension gives its group, as in Table E.3, which lists all 30 with their options.

TV or social), and the time of day. It shows the CR each predicted dimension where it takes effect (Section 8.2). She accepts the predicted value or replaces it, which fixes that dimension for the meal and re-predicts every dimension still open, conditioned additionally on what she has confirmed or corrected so far, so one correction can revise several others. She can also revise an already-decided dimension mid-meal from the settings overlay, which takes effect at once and likewise re-predicts the open ones. The values she ends the meal with become that meal's record.

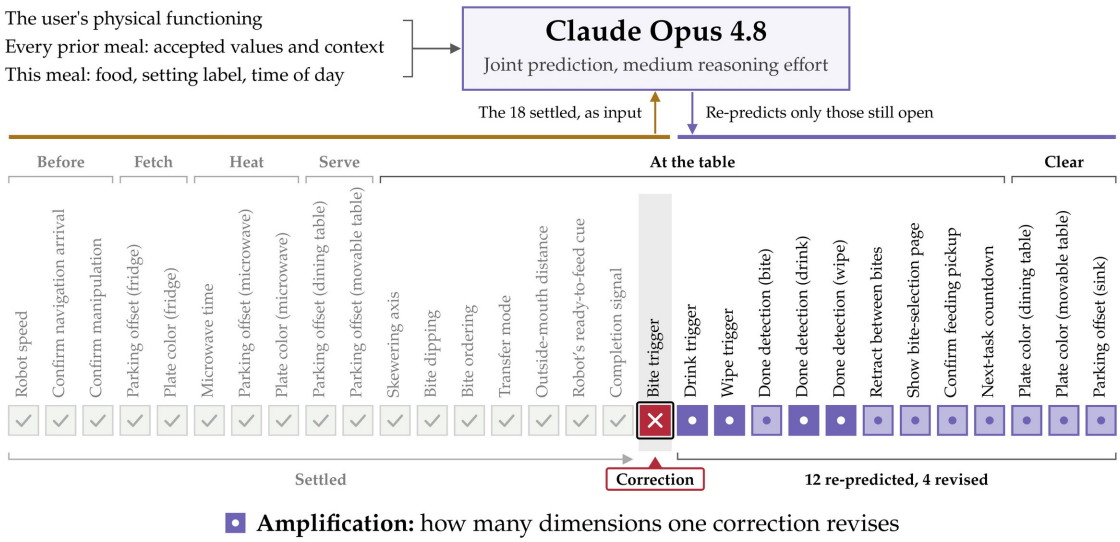


Figure 8.4: **Predict, correct, re-predict.** Before the meal, Claude Opus 4.8 predicts all 30 dimensions jointly, at medium reasoning effort, from the community researcher's physical functioning, every prior meal's accepted values and context, and this meal's food, setting label, and time of day. In this illustration, she has settled the first 17 dimensions and corrects the bite trigger. With the 18 settled as input, the robot re-predicts only the 12 still open and revises 4 of them. The number of dimensions one correction revises is its amplification (Section 8.5).

8.4 Deployment

8.4.1 Study Procedure

We conducted the deployment as community-based participatory research (CBPR) [89] with a community researcher (CR) who is a co-author of the work in this chapter. Our collaboration began in Fall 2022, and she has since evaluated several components of the system in her home, though never the system as a whole. She has Multiple Sclerosis, retains partial arm and neck mobility, and cannot feed herself; she operates the robot's web interface on her iPad with her left arm. Before the study, we spend five weeks developing and testing the system in her home. Most of this time goes to navigation: the house's narrow



Figure 8.5: **A month of meals in her home.** The community researcher's 22 meals in order, each shown by its plate and ringed in the color of its setting. We planned three settings with her: watching TV, alone, and with a friend (*top*). From meal 12 on, she proposed three more as she grew comfortable with the robot: music on, a video call, and iPad games (*bottom*). Table E.5 lists the meals in each setting.

corridors leave little margin for error, and we make minor additions to her home, such as skirts on her tables so that the lidars see a surface rather than legs. We also plan the study with her: the meal schedule, set around her availability, and the three settings in which she expects to use the robot, eating on her own or while watching television at a movable table, and with a friend at the dining table. Her only exposure to the complete system before the study is a single trial meal three days before it. Appendix E.3 gives a day-by-day account of this preparation phase, including every addition to the home.

The deployment then runs for one month, July 17 to August 17, 2026: 22 meals, one per weekday, with a weekend meal making up any weekday she is unavailable. Although we plan for three settings, she proposes new ones as she grows comfortable with the robot and eats in six by the end (Fig. 8.5). A

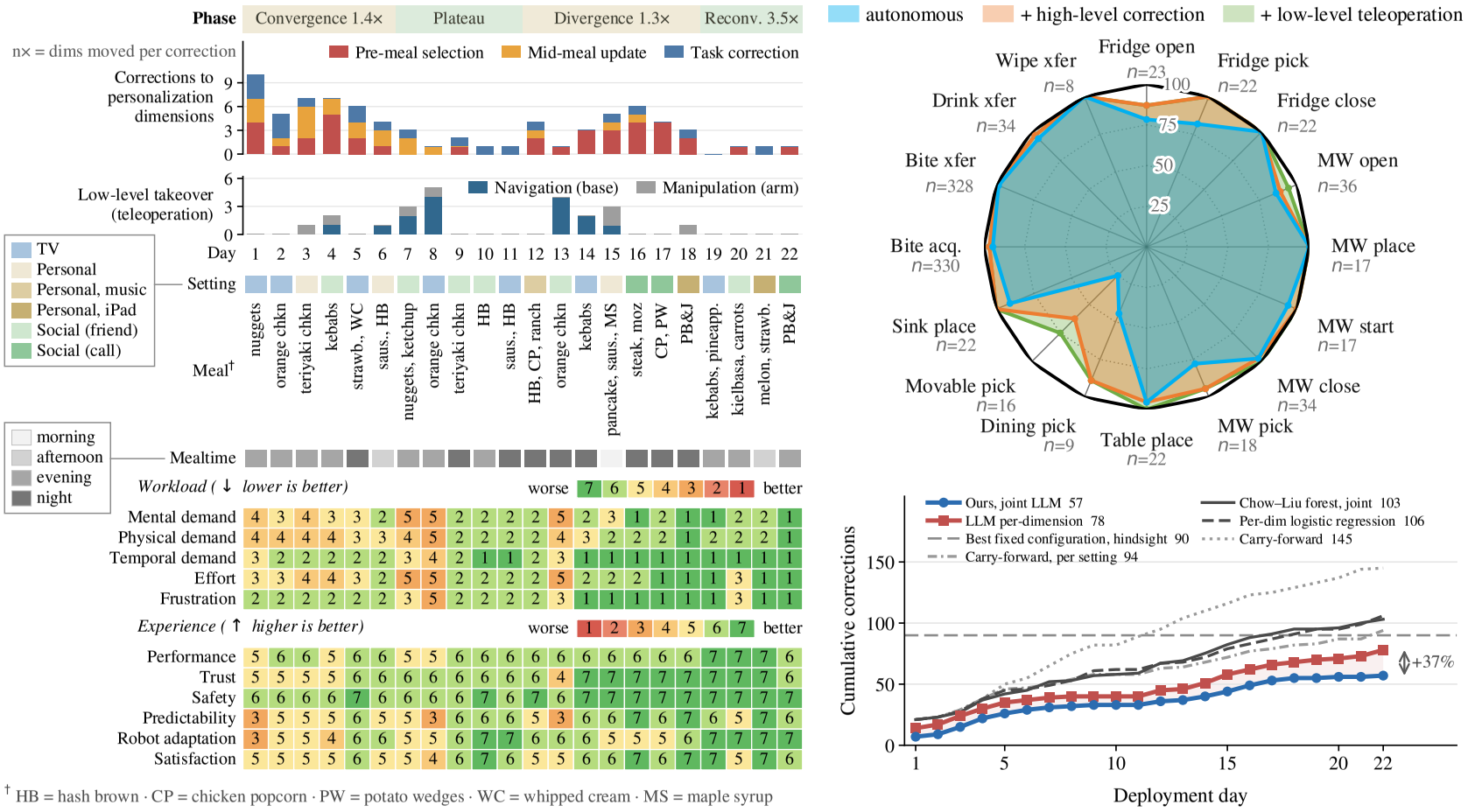


Figure 8.6: **The deployment, day by day.** *Left:* corrections to meal preferences and environment parameters, teleoperation, setting, meal, time and her daily ratings per meal, with the four segments of Section 8.5 marked. *Top right:* success rates of the perception-dependent skills. *Bottom right:* cumulative corrections against the baselines of Appendix E.8.

researcher is present, explains the system on request, and intervenes when a skill cannot recover or safety requires it (Section 8.4.2). As in prior in-home deployments [9, 211, 224] we do not freeze the system: we fix it between meals as failures arise and list every change, with the meal it applied from, in Appendix E.5 [198]. The CR chooses what she eats, when, and in what setting; every plate as served is shown in Appendix E.6. For every meal we collect the system's logs, the researcher's notes, and a post-meal survey that includes what she learned about the robot that day, and a post-study survey.

8.4.2 Reliability in the Home

A complete meal, from taking the plate from the fridge to putting it in the sink, took a median of 59 minutes excluding researcher interventions (min 36, max 82), with medians of 15 minutes for preparation, 28 for feeding and 7 for clearing. Over the month, the robot executed 1,335 skills, 958 of them perception-dependent. For these, cumulative success was 93.2% under full autonomy, 96.8% once the CR's high-level corrections through the interface are added, and 97.4% once her low-level teleoperation is added. Her corrections were crucial to this reliability: where perception was weakest, picking the plate up from the movable table, autonomous success was 25%, rising to 62.5% with her high-level corrections and to 75% with her low-level teleoperation. We show the success rate of every perception-dependent skill in Fig. 8.6 and the per-meal intervention log in Appendix E.4. The researcher intervened 32 times across 17 of 22 meals and gave 27 explanations. The logged intervention intervals lasted a median of 1.6 minutes. Time per bite, from her request through bite selection, acquisition and transfer, was 56 s at the median, against 1:00 to 2:26 for the five users and 1:41 for the community researcher in Nanavati et al. [198] over the same stages.

We changed the system between meals as failures revealed the need; Appendix E.5 lists every change and the meal it applied from. The major one was the drinking skill, which became functional from meal 12; until then we asked her to set her drinking preferences at each meal as if the skill were available. A navigation bug that spoiled the base's final parking at meals 8 and 13, leaving her to correct its position herself, was fixed after meal 13; at meal 13 it also cost 16 minutes of researcher intervention. Hardware repairs were confined to the gripper's 3D-printed plastic fingertips, which worked loose or broke on three days.

The rest of this chapter reports how the community researcher and the robot adapted to each other over the 22 meals (Figure 8.7), how she rated the robot, and the lessons we draw from the month. Figure 8.6 shows the deployment day by day.

8.5 Robot's Prediction Under Non-Stationarity

The robot adapts to the CR by predicting her meal preferences and its environment parameters. Given the meal, the time of day, and the setting, personal, social, or television, it predicted all 30 personalization dimensions before each meal. Each dimension she corrected was held fixed, and the robot re-predicted every dimension still open in light of it, so one correction could change several others; we call the number of dimensions a single correction revises its *amplification*. The nine environment parameters behaved differently from the 21 meal preferences. The five parking offsets were never corrected. The four plate colors were corrected 19 times, independently: the refrigerator and microwave light

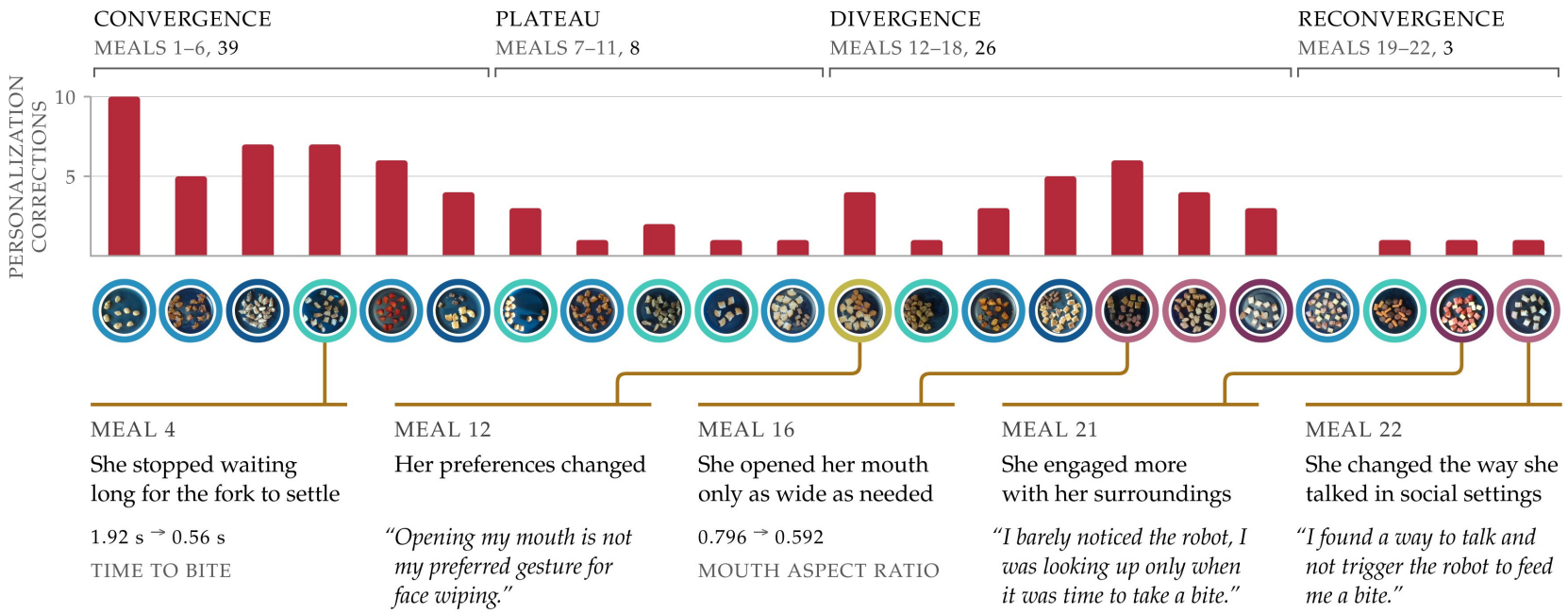


Figure 8.7: **Human–robot co-adaptation over the 22 meals.** *Top:* the robot's side, every correction the community researcher made at each meal, counting corrections to her meal preferences and to the plate colors together, under the four segments of Section 8.5. With the 19 plate-color corrections included, the segments total 39, 8, 26, and 3, against 29, 4, 22, and 2 for the 57 corrections to her meal preferences alone. *Center:* the plate of each meal, ringed in the color of its setting: watching TV (blue), alone (dark blue), with a friend (teal), music (olive), a video call with her daughter (rose), and iPad games (plum). *Bottom:* her side, five moments of her adaptation to the robot (Section 8.6), from no longer waiting long for the fork to settle at meal 4 to talking through the video call at meal 22 without triggering a bite.

their own interiors, so room lighting did not move them as a block as we had expected. The meal preferences, by contrast, were correlated, measurably: among the values she accepted, 30 of 210 dimension pairs predicted one another better than the dimension's own most common value, against 11 by chance ($p = 0.001$, Appendix E.2).

Her 57 preference corrections fall into four segments (Fig. 8.6):

Meals 1 to 6, convergence. 29 corrections as the robot learned her preferences in the three planned settings, half of them mid-meal edits as she explored what she liked and settled on it.

Meals 7 to 11, plateau. 4 corrections, and none in meals 10 and 11.

Meals 12 to 18, divergence. 22 corrections, for three reasons. She tried new settings as she grew comfortable: music at personal meals, a first morning meal, games on her iPad, and video calls with her daughter, which split the social setting in two (Section 8.6). Her preferences also changed: after meal 12 she wrote that "opening my mouth is not my preferred gesture for face wiping", since unlike a bite or a sip nothing enters her mouth; at personal meals she switched the robot's readiness cue from the LED to speech, "because I am too busy looking at the iPad screen". And drinking became functional at meal 12, bringing its dimensions into use.

Meals 19 to 22, amplified reconvergence. 2 corrections. The two descents differ in amplification: over the first six meals each of her corrections revised 1.4 other dimensions; over the final four, 3.5. The last meal is the clearest instance (Figure 8.8). On a video call with her daughter, the robot proposed her in-person social preferences: button, LED, retract. We expected her to correct the bite trigger

to open mouth and the rest of her daughter-call preferences to follow. Instead she corrected one dimension, asking to be signaled by speech, the cue she used at personal meals, and from that alone the robot revised five others, landing on her personal preferences rather than her daughter-call ones: bite and drink triggers to open mouth, the completion signal to speech, no retraction, and a countdown in place of waiting. She accepted all five. It felt natural, she said, because it was her daughter.

Over the deployment this held her correction workload to 57. Figure 8.6 (bottom right) replays the same 22 meals under five alternatives. The same large language model (LLM) predicting each dimension separately from its own history, as in CBTL [248] (Chapter 7), costs 78 corrections. Four count-based predictors with no LLM cost between 94 and 145: the best repeats her last accepted values in the same setting, as FEAST does [122] (Chapter 6); the others are carry-forward, a per-dimension logistic regression, and a Chow–Liu forest over the 21 dimensions [40]. The best fixed configuration in hindsight costs 90. Only the two LLM predictors finish below it, and the joint predictor 27% below its per-dimension counterpart (Appendix E.8).

8.6 Human Side of Co-Adaptation

A robot that learns its user is only half of a co-adaptive system; the other half is a user who learns the robot. Over the month the CR learned what the robot perceived and when it acted, changed how she used it, and the robot adapted back.

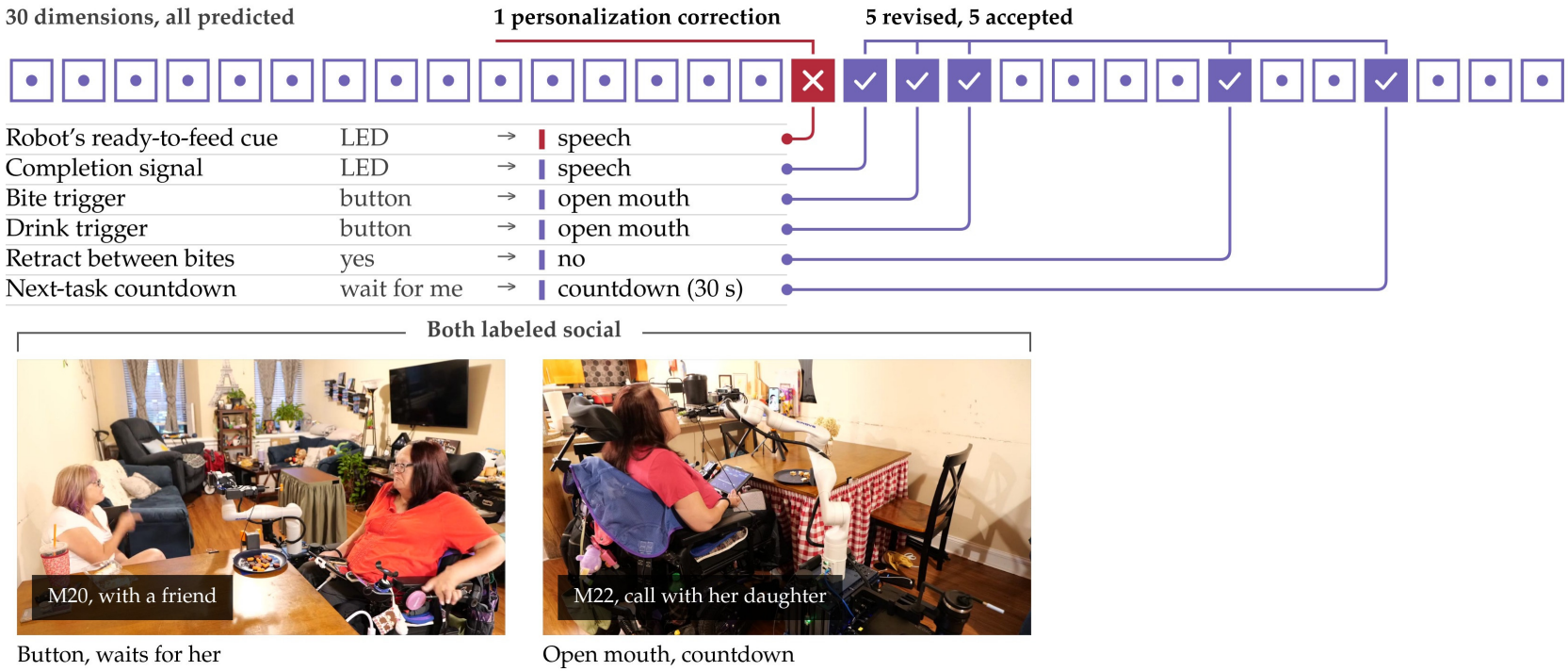


Figure 8.8: **One correction revised five preferences.** At meal 22, a video call with the community researcher's daughter, the robot predicted all 30 personalization dimensions (*top*, one square each) and proposed her in-person social preferences. She corrected one dimension (red), the robot's ready-to-feed cue, from LED to speech. From that correction alone, the robot revised five others (filled): the completion signal to speech, the bite and drink triggers to open mouth, no retraction between bites, and a 30 s countdown in place of waiting for her. She accepted all five. *Bottom:* meal 20, with a friend, where she used the button and had the robot wait for her, and meal 22, on the call, with the open-mouth trigger and a countdown. Both meals carry the social label.

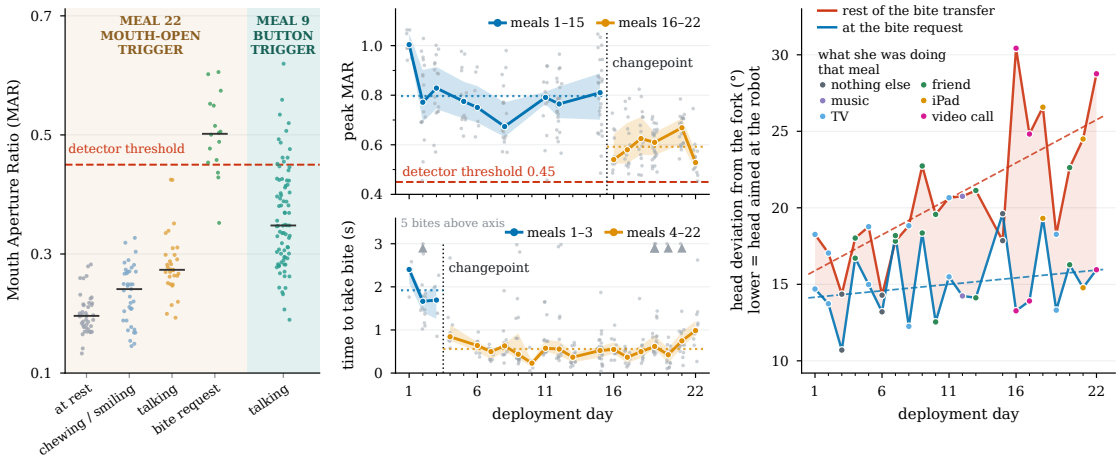


Figure 8.9: **Four views of how the community researcher adapted to the robot.** *Left:* peak mouth aspect ratio (MAR) in each second of meal 22 and meal 9, grouped by what she was doing in that second; ticks are medians. *Center top:* her triggering aperture, which steps down at meal 16. *Center bottom:* her time to take the bite, settling by meal 4; bands are interquartile ranges, dotted lines fitted levels. *Right:* head direction relative to the fork at the bite request (±1 s) and during the rest of the transfer (more than 3 s from it).

Learning to talk below the robot's threshold. The open-mouth trigger fires on an aperture threshold, so it cannot tell a bite request from a word spoken with the mouth open beyond it. She met that limit at her first social meal: at meal 4, with a friend seated to her right, she switched to a button, "because I didn't feel rushed and could carry on a conversation", and she kept the button at her friend's later visits, where the robot now proposed it (meals 9, 10 and 13). Meal 16 was her first video call with her daughter. Comfortable with her, she overrode the robot's button and went back to open mouth, and began working out where talking ends and a request begins; at meal 17, her daughter on the call again, she overrode the button once more. She switched back to the button when her friend was over (meal 20, correcting the robot), but at meal 22, her third call with her daughter, she talked through the meal on the open-mouth trigger. Figure 8.9 (left) shows how: labeling every second of that meal's video, not one of her 110 seconds of

talking, chewing, smiling or resting reaches the 0.45 threshold, while her bite requests sit at a median mouth aspect ratio (MAR) of 0.502 (Mann–Whitney $p = 6.4 \times 10^{-11}$). At meal 9, with the button and no reason to hold back, her talking sat at a median of 0.348 and 12 of 79 talking seconds, 15%, crossed 0.45. Her talking at the two meals is not the same distribution (Mann–Whitney on peak MAR $p = 2.3 \times 10^{-6}$; Fisher on the crossing rate $p = 0.018$): she had changed how she talked to accommodate the detector, keeping speech below the line and opening above it. "I found a way to talk and not trigger the robot to feed me a bite," she wrote that day.

Ceasing to over-accommodate the robot. She stopped doing more than the robot needed, in two abrupt steps: from meal 16 she opened her mouth only as wide as the detector required, and from meal 4 she waited on the arriving fork only as long as she had to. Her triggering aperture over 259 bite requests sits at two levels with one step between them, at meal 16: a median MAR of 0.796 before, 0.592 from then on (Fig. 8.9, center top; changepoint permutation $p = 7.5 \times 10^{-4}$), 59% of her margin above 0.45 given up in a single meal, the first with her daughter on the call, where she felt free to test the detector. Her time to take the bite once the fork arrived had settled earlier and as abruptly, from 1.92 s over the first three meals to 0.56 s from meal 4 (Fig. 8.9, center bottom).

Shifting engagement away from the robot. Across 357 transfers, her head direction at the moment she requested a bite, when the robot reads her head pose to aim the fork, is unchanged over the month (Fig. 8.9, right; $\rho = +0.24$, $p = 0.28$). During the rest of the transfer, when it does not require her head pose, she turned further from the robot as the deployment proceeded ($\rho = +0.74$, $p = 0.011$); by day 10 she would trigger a bite and carry on talking until it reached her. She

chose to eat in settings that asked more of her attention, games on her iPad and video calls with her daughter on a phone propped to her left on the dining table, and changed her preferences to match, moving the robot's readiness cue from the LED below the camera, which she was no longer looking at, to speech, which the robot learned to propose at those meals (Section 8.5). Of her last tablet meal she said, "I barely noticed the robot, I was looking up only when it was time to take a bite."

Anticipating the robot's failures. In the first week she learned its controls: "how to control the robot's arm myself and restart a skill in case of an error" (day 3). In the second she learned what it perceived and where it stalled: "the robot is very sensitive to lights and shadows" (day 7); "the robot base motion may be very sensitive to my wheelchair moving around it" (day 8). In the third week she acted on the model before the robot failed. On day 14 she asked for the confirmation pages back: "I am wearing red, so I will bring back confirmation pages, as the robot might confuse me and the plate attachment", a failure foreseen from a cue the robot cannot perceive. On day 18, after her wheelchair moved the movable table and a joint hit its limit reaching for the plate, "now I will be extra careful when moving around the movable table", and on day 19, when she moved it again, she asked for it to be brought back within the robot's workspace before the robot reached for the plate. On day 21, as the room darkened, she asked for the lights so the robot could still find the plate.

8.7 Rated Workload, Experience, Acceptance

After each meal the CR rated eleven 7-point items, shown day by day in Fig. 8.6: the five NASA Task Load Index workload items [93] and six on her experience

of the robot. Workload fell across the month, from a mean of 2.96 in the first week to 1.36 in the last, and every item fell individually. Her experience ratings rose: trust first reached 7 at meal 14 and performance at meal 19, her rating of the robot's adaptation climbed from 3 to 7, and she never rated safety below 6. The one departure is meal 13, where trust dropped to 4 and predictability to 3 after navigation failed repeatedly, the bug fixed after that meal (Section 8.4.2); she said she trusted the feeding but not the driving, and both rebounded the next meal, to 7 and 6.

After the final meal she completed the System Usability Scale [25], the five Technology Acceptance Model items [53], and four items comparing the robot with her caregiver. She rated usability at 82.5, an A on the Sauro–Lewis curved grading scale [236]: she strongly disagreed that the system was cumbersome or that she would need a technical person to use it (once used to it by the end of the deployment), and strongly agreed that she would use it frequently and felt confident doing so; her one neutral rating was on whether it was unnecessarily complex. She rated usefulness, attitude, intention to use, and enjoyment 5 of 5 and ease of use 4. Against her caregiver, the robot raised her sense of independence from 2 to 5 and her sense of control from 4 to 5. She rated both sides of the co-adaptation at 5: the robot got better at understanding her preferences, and she got better at working with it.

8.8 Lessons Learned

Lesson 1. *The target moves: as trust grows, the user explores unforeseen scenarios and revises preferences even at familiar meals. The first plateau is not an endpoint for the learner or the study.*

The CR's corrections plateaued from meal 7 to meal 11 (Section 8.5). A study of 10 meals, the length of the in-home study of Nanavati et al. [198], would have ended within this plateau. At that length, a plateau is indistinguishable from convergence. As her trust rose (Section 8.7) she chose unplanned scenarios, ones that asked more of her attention (Section 8.6): music, video calls with her daughter, games on her tablet (meals 12 to 21). These brought new preferences: with a game on her tablet she moved the readiness cue from the LED to speech, and on the calls, comfortable with her daughter, she went back to open mouth (Section 8.6). The wiping cue moved with no change of scenario at all (Section 8.5). A learner or a study that stopped at the plateau would have missed all of it.

Lesson 2. *Preferences for an assistive robot depend on the other people present, and so does the cost of its mistakes.*

The CR chose the button with a friend at her side, and had the robot wait for her to ask for the next bite rather than count down, at every visit (Section 8.6, Appendix E.9); on video calls with her daughter, despite the same social label, she chose open mouth and a countdown, and at the last call one correction of hers led the robot to propose her personal preferences for the rest, which matched what she wanted; it felt natural, she said, because it was her daughter (Section 8.5). The relationship, not the label, set her preferences. At meal 20, while a friend was

visiting, the button misfired and the robot moved toward her unasked, and she was more critical of it than at any other meal. Participants in a recent speculative design study asked for a robot that adapts to the relationships at the table [134].

Lesson 3. *A user is able to tinker only when they can diagnose: variable autonomy covers the scenarios where skills fail, but the user can step in only where the robot's state is legible.*

The CR stepped in throughout the month: selecting the food skewer point herself when the detector could not separate items (25 times over 11 meals), driving the base the last stretch when navigation stopped short, holding the robot above the plate until syrup stopped dripping (day 15); Fig. 8.6 shows how user intervention increased success on the weakest skill from 25% to 75% (Section 8.4.2). She could do this where she could read the cause, lighting and shadows (day 7), her red shirt against the plate attachment (day 14), joint limits from the workspace, and by the third week she acted before failures rather than after (Section 8.6). Navigation was different: she had less information about why it failed, and it was the only capability she said she did not trust (meals 8, 13). Nanavati et al. found that users overcome off-nominals when given varied control [198]; here she had control over every component and could use it only where she could see why it was failing.

Lesson 4. *As use of a physical caregiving robot grows, attention to it falls: it succeeds by fading into the background.*

During transfers the CR turned further from the robot as the month went on, while her head direction at the bite request, the moment the robot reads it, did not change (Section 8.6). Long-term studies of social robots treat a robot that

fades from attention as novelty wearing off [152] and report engagement as the outcome [46]; for a robot that feeds someone, fading is the outcome. It was not disengagement: her use of the robot widened through it (Section 8.4.1) and her trust rose (Section 8.7). "I barely noticed the robot," she said of her last tablet meal, "I was looking up only when it was time to take a bite."

8.9 Discussion

In this work, we study co-adaptation through the first month-long in-home deployment of an autonomous physical caregiving robot. Our results are promising: the robot learned the CR's preferences through non-stationarity, she co-adapted with the robot, and her workload fell and her trust rose over the month. However, these findings come from one person, one home, and a researcher present at every meal; longer deployments with more care recipients and less researcher presence are needed to test whether the lessons hold. Additionally, two events, a meal smoking in the microwave and a coughing fit that needed a drink at once, made it necessary for the researcher to intervene; a robot left alone would need to respond to such rare hazards faster than deliberation and more reliably than any skill here. Finally, the CR handed to the people around her not only tasks the robot could not do, such as cutting her steak, but also ones it could do but not fast enough for her, such as reheating her plate when she was in a hurry; how tasks should be allocated between an assistive robot and the care team around it is a question for future work.

CHAPTER 9

DISCUSSION

In this thesis, we bring physical caregiving robots out of labs and into people's homes by lengthening their deployment step by step, from a single session to a month. We first identify what a physical caregiving robot must personalize to, with stakeholders in the loop. In SPARCS, we engaged with eight care recipients, two caregivers, and nine occupational therapists to scope how caregivers assist and which challenges remain open in physical robot caregiving. Moreover, we collected clinical data for six care recipients with varying disabilities and identified their care requirements in 19 caregiving scenarios. In the formative study of FEAST, 21 care recipients voiced 46 types of adaptability requests for a robot that feeds them.

We then deploy for a single session, personalizing to each care recipient's functional abilities (Part I). Feel the Bite fed 13 care recipients with diverse medical conditions, including Multiple Sclerosis, Spinal Cord Injury, Arthrogryposis, and Schizencephaly, across our lab, a medical center, and a care recipient's home. Nine of them could not lean forward. The care recipients consistently emphasized the comfort and safety of its inside-mouth bite transfer and gave it high technology acceptance ratings.

Over a few sessions, we personalize to behavioral preferences that shift with the situation (Part II). FLAIR feeds complete in-the-wild meals by sequencing a varied library of skills. We evaluated it on six realistic plates, ranging from DoorDash orders and prepared grocery store meals to homemade meals. On

these plates, it picked up food efficiently while adhering to the diverse preferences of 42 participants without mobility limitations, as evaluated in a user study. With FEAST, the robot fed our two community researchers three meals each in their homes, across three contexts: personal, watching TV, and social. Both personalized it on the fly to meet their individual needs and preferences and rated it above their human caregiver for control and independence.

Over a month, we personalize across tasks and over time (Part III). In CBTL, the robot learns preferences incrementally from online interaction, within its planning constraints. We evaluated it in its own studies, separate from the month-long deployment. In a web-based study across five simulated meals, 60 participants significantly preferred its choices over a non-personalized baseline. In an in-home case study with two care recipients, it learned their preferences over four screen-based meals and correctly predicted them at the fifth, which involved physical feeding. Finally, over the month-long deployment, the robot assisted Aimee with 22 meals in her home, from the refrigerator to the sink. Of the 958 skill executions that depended on perception, 93.2% succeeded without her input and 97.4% with her corrections or teleoperation. We observed human–robot co-adaptation: her preference corrections fell to near zero by the second week, resurfaced as she explored new settings and her preferences changed, and fell again as the robot adapted. She adapted too, learning to talk without triggering its open-mouth detector. Her workload fell, her trust reached the top of the scale, and, as in FEAST, she rated the robot above her human caregiver on independence and control.

However, these deployments leave two pressing questions open. First, the longer the deployment, the fewer people it involved, down to one person for

one month, so we discuss how physical caregiving robots can scale to many care recipients (Section 9.1). Second, a researcher was present at every meal of the month and intervened 32 times, so we discuss the science of safety that removing the researcher requires (Section 9.2). We end with the limitations of this work and the open questions they leave (Section 9.3).

9.1 Scaling to Many Care Recipients

For physical caregiving robots to reach the many people who need assistance with eating, they must work in homes and for care recipients their developers have never met. Our approach is to build broadly for all of them and then to personalize the robot to each one. We identify requirements with stakeholders in the loop and build a system with the capabilities to address each of them. A calibration step then fits parts of this system to a care recipient's functional abilities, and over time, the system personalizes to their behavioral preferences. Below, we describe each of these four steps in turn.

Identifying Requirements with Stakeholders. We first identify the types of requirements broadly, together with care recipients, caregivers, and occupational therapists. Personalization, including in-the-wild personalization by care recipients themselves, addresses each person's specific requirements within these types. In SPARCS, occupational therapists built Structured Workflows for the caregiving of six care recipients, and care recipients and caregivers described their needs for assistance with activities of daily living (Chapter 3). SPARCS also models a care recipient's functional abilities separately from their behavior, the split that our two steps of personalization follow (Section 3.3).

In FEAST, a formative study with 21 care recipients identified 46 types of adaptability requests and three tenets of in-the-wild personalization, namely adaptability, transparency, and safety (Section 6.2). The month-long system later drew its 21 meal preferences from this study. While a formative study captures needs at one point in time, working with the same two community researchers since November 2022 also shows which needs change. For example, Aimee planned the month's schedule and its three settings with us, then proposed three more as she grew comfortable with the robot (Appendix E.4).

Building a Broad System. We then build a system with the capabilities to address these requirements, although no single care recipient needs all of them. Feel the Bite's physical interaction-aware control, for instance, distinguishes four types of physical interaction during bite transfer (Section 4.2.2). These are incidental contacts outside the mouth, in-mouth manipulation with the tongue, impulsive contacts from involuntary spasms, and bites. However, not every care recipient produces all four. Only three of the 13 care recipients it fed, for example, have spasms that can happen unexpectedly (Section 4.4).

FEAST, in turn, offers many interfaces, so that each care recipient can use the ones that suit them. These include accessible buttons, spoken or typed language, mouth and head gestures, custom gestures that care recipients create themselves, and a web interface that runs on their own devices and works with the assistive technologies they already use (Section 6.3). The month-long system also offers options that Aimee never used, such as inside-mouth bite transfer and a skewering axis that suits narrow mouth openings (Tables C.2, E.3, and E.10).

Calibrating to Functional Abilities. A calibration step then fits parts of this system to each care recipient's functional abilities. In Feel the Bite, the robot reconstructs a personalized head model of the care recipient from its two in-hand cameras. It then uses this model to estimate the pose of their head and mouth throughout bite transfer (Section 4.2.1). At the outset, the robot can also record a pose inside the mouth where the care recipient wants to be fed (Appendix A.4.2). If they prefer, the robot instead records this pose later, after an in-mouth manipulation in which they guide the utensil there with their tongue (Sections 4.2.2 and 4.3.2).

Similarly, we restrict the interaction classifier to the types of interaction that a care recipient produces. For a care recipient without spasms or in-mouth manipulation, we use only two classes, bite and non-bite. Feel the Bite also evaluated per-user fine-tuning offline, in which about 10 of a participant's own interactions per type raised the classifier's F1 score from 0.719 to 0.772 (Table 4.3).

Likewise, FEAST tailors its head pose limits to the user's range of motion (Section 6.4.3). In the month-long deployment, the personalization pipeline receives a description of the care recipient's physical functioning (Section 8.3). For Aimee, the skewering axis, transfer mode, and outside-mouth distance follow from her abilities.

Personalizing to Behavioral Preferences over Time. Behavioral preferences shift with the situation and over time, so the system keeps personalizing to them meal after meal. In FEAST, care recipients personalize the robot in-the-wild. During deployment, they select the interfaces and gestures they want to use and change them through natural-language requests. A language model turns these

requests into updates to the robot's behavior trees (Section 6.3.2). When the default head nod detector mistook Aimee's downward lean during a sip or a wipe for a nod, for example, she asked the robot to end both with a button instead (Section 6.5).

Over the month, the robot predicted her 21 meal preferences jointly before each meal and learned from her corrections (Section 8.3). Her behavioral preferences, unlike the three dimensions that follow from her abilities, kept changing. After a plateau from meal 7 to meal 11, her corrections rose again as she tried settings we had not planned and revised preferences such as her gesture for face wiping (Section 8.5). To personalize to behavior, a robot has to keep learning for as long as it is in use, from as few corrections as possible.

The Path to Many Homes. Two of these steps repeat in every home, and each has its own cost. Calibrating to a care recipient's functional abilities, and to their home, needs researchers. Feel the Bite's in-home session took an afternoon to set up, while the whole-meal system, which drives through a home and operates its appliances, took weeks of development and testing in Aimee's home, mostly for navigation (Section 8.4.1). Personalizing to behavioral preferences runs on its own, but it starts from scratch with each care recipient, as Aimee's 29 corrections over her first six meals show (Section 8.5). To reach tens of thousands of homes, and eventually millions, calibrating to each person's functional abilities and home must become something care recipients, caregivers, or occupational therapists can do themselves, and personalizing to behavioral preferences must carry what it learns from one care recipient to the next.

9.2 Toward a Science of Safety

A physical caregiving robot puts a utensil into a person's mouth, carries hot food, and drives through the narrow corridors of a home and around a wheelchair. We built layered safety into Feel the Bite, FEAST, and the month-long system, and their users rated them safe. Care recipients perceived Feel the Bite as safe and comfortable, and FEAST's users rated it highly for safety. Over the month, Aimee never rated her safety below 6 of 7 (Section 8.7).

However, a rating after a meal says little about the hazards a researcher averted. Over the month, the researcher stopped a run or repaired the robot 32 times across 17 of the 22 meals (Appendix E.4). Some hazards are also rare enough that only a continuous deployment reveals them. To remove the researcher, we need to treat safety as a science and build frameworks for it. As a starting point, we consider the two kinds of risk in Figure 9.1: known unknowns, which we can anticipate, and unknown unknowns, which no one predicted.

Known Unknowns: Systems-Level Safety. We identify known unknowns from formative studies, from the hazard identification of ISO 13482, the safety standard for personal care robots [113], which we applied to FEAST (Appendix C.3), and from hazards observed during testing, pilot studies, and deployments. For example, a participant in FEAST's formative study said, "Safety is the most important part. If I turn my head to look somewhere the robot should not try to feed me at that position which might be unsafe." We tackle known unknowns with systems-level safety. FEAST accepts food, drink, and head detections only within predefined zones, and any detection outside them, such as

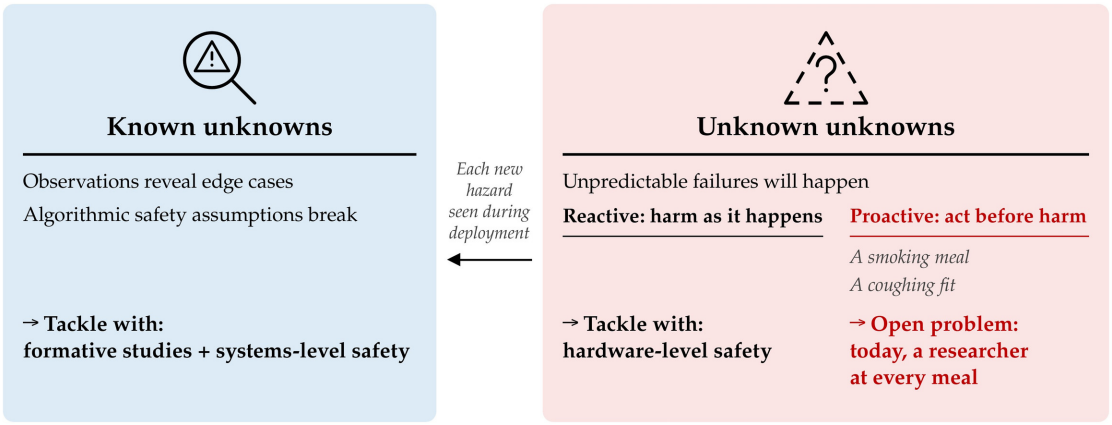


Figure 9.1: Two kinds of risk in a home. *Left:* known unknowns are tackled with formative studies and systems-level safety. *Right:* unknown unknowns are met by hardware-level safety as harm happens. Acting before harm is an open problem, and today a researcher is present at every meal. Each new hazard seen during a deployment moves from right to left.

a caregiver's head mistaken for the user's, sends the robot to a safety state (Section 6.4.3). Feel the Bite classifies the physical interactions during bite transfer and switches between compliant controllers designed around the spasms and tongue guidance of its care recipients (Section 4.2.2). In FEAST and the month-long deployment, watchdogs continuously monitor all sensors and emergency stops and send the robot to a safety state if any of them fails. In the month-long deployment, the system also checked every predicted preference against its domain, and its collision monitor stopped the arm when its joint torques departed from a dynamic model by more than a threshold set for each skill phase (Section 8.2).

Unknown Unknowns: Hardware-Level Safety. Unknown unknowns are unpredictable, so the final safeguard against them has to be physical. FEAST's fork

holder has a mechanical weak point that breaks the utensil in two under excessive force, so that only the fork tip remains in the mouth (Section 6.4.3). For the month-long deployment, we extended hardware-level safety to the gripper's fingertips, which break under excessive force. The month-long system also had two emergency stops: a physical one with the researcher, which halts the arm and the base, and a takeover button on the web interface for the care recipient (Section 8.2). Neither needs to predict a hazard: an emergency stop halts the arm, and a breaking weak point ends a dangerous physical interaction. Once such a hazard happens, it becomes a known unknown that systems-level safety can address, so the set of unknown unknowns shrinks as a deployment lengthens.

Acting Before Harm. However, hardware-level safety cannot handle every hazard. During the preparation for the deployment in Aimee's home, a meal that was too dry began smoking in the microwave. On another occasion, Aimee began coughing and needed a drink at once. Halting the arm does not stop the microwave, and the care recipient still needs a drink. Both times, a researcher stepped in (Section 8.9). A science of safety, as we see it, makes the cycle in Figure 9.1 repeatable: it measures the hazards a researcher averts, not only how safe a meal felt, and turns each hazard a deployment reveals into a systems-level check. Removing the researcher also requires acting before harm on such hazards, which remains an open problem.

9.3 Limitations and Open Questions

Beyond One Person. Each longer deployment involved fewer care recipients. Feel the Bite fed 13, FEAST and the in-home case study of CBTL involved the

same two community researchers, and the month-long deployment involved one, a community researcher who had co-designed the system. This raises three questions: (i) how much of the five weeks of preparation a second home would need, (ii) whether what the robot learns from one person can help the next, and (iii) whether the system serves care recipients who did not help design it. Among the values Aimee accepted, 30 of 210 pairs of meal preferences predicted one another better than each dimension's own most common value, against 11 by chance (Section 8.5). We have not tested whether another person's preferences share such correlations, so that one person's history can inform the initial predictions for the next. In FEAST, an occupational therapist not involved in its development personalized it in 10 scenarios of their own (Section 6.6). However, only deploying with more people in their own homes can show whether a care recipient who did not help design the robot can calibrate and correct it as Aimee did.

Beyond One Month. The longest deployment lasted one month. Whether Aimee's preferences eventually settle, or keep shifting as her life and her trust in the robot change, is unknown. While care recipients in FEAST's formative study noted that their preferred interaction may change "over time, as their medical conditions change" (Section 6.2), a month was not long enough to capture such changes in Aimee's functional abilities. To serve a care recipient for years, a robot will need to notice when their functional abilities change and recalibrate to them.

Beyond Mealtime. Our deployments assisted with a single activity of daily living, eating, and with the tasks around a meal. In Chapter 3, the stakeholders in SPARCS described what other activities of daily living would demand of a

robot. These demands include interfaces beyond a joystick, since in dressing a user may be unable to see around the task, and collaboration with caregivers and with assistive devices such as a Hoyer lift in transferring. Only deploying for these activities, with stakeholders in the loop from the start, can show whether the forms of personalization that the recipe surfaced for mealtime also matter there.

Without Researcher Oversight. A researcher was present at every in-home meal, so the reliability we report is that of a system with a researcher in the room. Over the month, the researcher's logged interventions lasted a median of 1.6 minutes each. Besides responding to hazards, the researcher repaired hardware, restarted software, and explained the system. Aimee also handed to the people around her tasks that the robot could not do, or did too slowly for her (Section 8.9). At the meals, she delegated nine caregiver tasks to the researcher as she would to a human caregiver, among them cutting her steak and reheating her food (Appendix E.4). Who should take on each of these roles, whether the robot, the care recipient, or the people around them, is a question of task allocation that the stakeholders in SPARCS raised in Section 3.9. Deploying physical caregiving robots without a researcher present remains an open question.

We hope that this thesis brings us one step closer to physical caregiving robots that provide long-term assistance in people's homes, so that those who need help with eating can eat at their own pace.

"Rajat, I miss it [the robot]."

— Aimee Whitneck
September 17, 2026, a month after her last meal with the robot

APPENDIX A

FEEL THE BITE APPENDIX

A.1 Illustrative Scenario

Jonathan is a 30-year-old male with spastic quadriplegic cerebral palsy due to prolonged anoxia at birth. He requires assistance from his caregiver for feeding and all other activities of daily living (ADLs). While he can eat solid food and thickened liquids safely, his retracted lips and anterior overbite require food items to be placed at the sides of his mouth for successful feeding. Due to better bite control and jaw mobility on the left side of his mouth, Jonathan prefers placing most food items there. However, for soft and sticky food items which can get stuck to his palate, he likes to receive them on his right side, where his tongue control is better. Jonathan's preference for transfer can also vary depending on various factors, such as the temperature, texture, and taste of the food item and changes in muscle tone. Jonathan communicates his preferred side to his caregiver by guiding the utensil to the desired position with his tongue.

Due to the spasticity in his muscles, Jonathan's voluntary control of his head is often interrupted by reflexive movements (up-down twitches) which can be particularly dangerous when the utensil is inside his mouth. His caregiver needs to react to these movements and reposition the utensil such that it does not apply large forces inside his mouth. Like Jonathan, people who experience a traumatic brain injury, severe stroke, or brain tumors may develop dystonia. This neurological disease is characterized by involuntary muscle contraction and abnormal posturing, which may require assistive feeding to be highly reactive and precise, adding challenges to an action that is already dynamic and intricate.

The scenario described above serves as an illustration of the challenges faced by various care-recipients worldwide who require in-mouth manipulation and may exhibit impulsive motions that affect their feeding. Impulsive motion is a common occurrence among care-recipients with conditions such as spinal cord injury [2] and multiple sclerosis [11]. Additionally, the need for in-mouth manipulation can be relevant to care-recipients who have a preferred location within their mouth for receiving food, such as individuals with arthrogryposis [226] and survivors of brainstem strokes.

A.2 Safety Considerations

Transferring food inside the mouth requires nuanced physical interactions, especially with care recipients who may experience involuntary movements. Given these complexities, ensuring safety is our utmost priority. To achieve this, we've developed our system with four distinct layers of safety, each designed to protect end-users during the feeding process –

Software-Based Safety Measures. Our algorithms prioritize safety. Our trials begin by recording the care recipient's resting head pose which is used to filter estimated head poses to remain within the likely range of the care recipient's head movement. This approach ensures accuracy even in uncommon scenarios that can arise during deployment in-the-wild, such as another person (their caregiver) appearing behind the user while the robot is feeding them. Importantly, in situations involving physical interaction (like reacting to a spasm while the robot is inside a care recipient's mouth), we err on the side of caution. We do not use uncertainty of prediction as a false positive (mistakenly detecting a spasm

and becoming more compliant) is preferable to a false negative (missing a real spasm).

Compliant Control. All our controllers are built on self-developed compliant control for the Kinova Gen3 (Appendix A.4.2), ensuring safety during physical interactions. This compliance means that even incidental contact between the fork and a care-recipient's mouth is only mildly uncomfortable. Additionally, using torque control allows us to always limit the robot's applied torque to safe levels.

Safety-Centric Hardware Design. We use silicone utensils attached to a custom-designed utensil holder with a deliberately engineered mechanical weak point. Should forces exceeding a certain threshold be applied to the utensil's tip, the weak point will yield, causing the utensil to break into two pieces. This feature is crucial for user safety; if too much force is applied while the utensil is in the user's mouth, it'll break and only the silicone tip will remain in the mouth, and all physical interaction with the robot will cease. In practice, our system has never required the utensil to break during inside-mouth bite transfers.

Immediate Emergency Intervention System. In all feeding trials, an experimenter sits beside the robot with an emergency stop button at hand to address any unforeseen occurrences. Activating this emergency stop immediately cuts power to the robot, ensuring user safety.

A.3 Robust Multi-View Mouth Perception

In this section, we provide details on our multi-view mouth perception method that is robust to occlusion, thus providing continuous real-time perception for the entire duration of bite transfer.

A.3.1 Data Collection and Inpainting Procedure

We train FEMTO using synthetic occlusions with ground-truth annotations generated from unoccluded originals. In this section, we provide details on the data collection procedure and inpainting method.

We conduct a three-phase data collection process with 10 participants. Each phase was designed to capture images of varying mouth openings, without any occlusion. During this process, the robot did not hold any utensils. We utilized two cameras, one positioned at the top and the other at the bottom, to record both depth and RGB images at a rate of 30 frames per second (Hz). Below is a brief overview of each phase:

- Phase 1 - Outside-Mouth Perception Data Collection
  - Participant Actions: Four different mouth states (closed, halfway open, fully open, teeth bared with varying openness) are recorded. Participants also perform random head movements centered around a reference point.
  - Robot Actions: The robot moves continuously from an initial position to a point just outside the mouth 3 cm away from lips (four iterations).

- Phase 2 - In-Front-Mouth Perception Data Collection
  - Participant Actions: Participants repeat the four mouth states, combined with a series of head movements (left/right rotation, up/down rotation, lateral rotation, left/right translation, forward/back translation, random).
  - Robot Position: Stationary, positioned just outside the participant's mouth.
- Phase 3 - Inside-Mouth Perception Data Collection
  - Participant Actions: The same mouth states and head movements are performed as in Phase 2, but this time participants simulate limited mobility as if a utensil is in their mouth.
  - Robot Position: Stationary, positioned inside the participant's mouth.

During post-processing, we extract 5000 RGB images per user spanning various head poses and mouth states. We use 8 users from this data collection for training, and reserve the rest for testing.

For inpainting, we captured images of a robot holding nine different food items – apple, cantaloupe, cherry tomato, red bell pepper, carrot, broccoli, celery, green bell pepper, and strawberry – using a fork. In addition, we recorded an image of the robot holding an empty spoon; we do not add any food on it as we noticed it does not significantly contribute to the occlusion created by the spoon itself. These images were captured from both the top and bottom cameras. The constant position of the utensil relative to the camera allows us to superimpose these masks onto unobstructed images, creating synthetic training

data. To prevent overfitting, we introduce minor variations in mask placement using Gaussian noise. Of all the masks collected, 8 are used for training, with the remaining 2 reserved for testing.

For the evaluation shown in Table 4.1, we use data collected from the test users, adding synthetic test utensil occlusion to both color and depth images. To make the depth image realistic, we add noise around the utensil, using a real depth image of the utensil in camera frame as reference. For generating ground truth 3D keypoints for this evaluation data, we apply our mouth perception pipeline with DECA to the unoccluded depth and color original images.

A.3.2 FEMTO Architecture and Training

In our experimentation with FEMTO, we tested various architectures, including those with and without frozen encoders, and compared various methods of combining encoder outputs, like concatenation versus max-pooling. Empirically, we found that FEMTO performs best with frozen encoders and concatenated outputs from these layers, followed by processing through two fully connected layers. For training, we divided the data from 8 available users into a split of 6 for training and 2 for validation. We employed Mean Squared Error (MSE) loss, AdamW optimizer [175], and Cosine Annealing [174] for the learning rate scheduler.

A.3.3 Mouth Perception Pipeline

In this section, we elaborate on the correspondence method integral to our mouth perception pipeline, which ensures accurate pose estimation even when

occlusions occur. The FLAME head model is key in this process, enabling us to estimate the mouth pose even when it is occluded by utilizing other head regions. FEMTO's camera pose output facilitates a direct correspondence between each pixel in the color image and a vertex on the custom-generated FLAME head model. However, occlusion from utensils often results in noisy depth data for some of these pixels. To address this, we need to employ outlier-resistant correspondence methods when using the entire head point cloud. While there are established methods like RANSAC [64], GORE [28], TEASER [282] for this purpose, we found that the upper part of the head remains unaffected by depth noise due to utensil occlusion. Therefore, we select a set of points from the forehead and beside the eyes for pose estimation. We extract these points from the point cloud and align them with corresponding points on the FLAME model. This alignment is achieved using the Kabsch-Umeyama algorithm, which minimizes the Root Mean Square Deviation (RMSD) between two outlier-free sets of points with established correspondences.

A.3.4 Qualitative Results

DECA often misreads the mouth as closed during occlusion – a limitation FEMTO overcomes, as qualitative results on test subjects show (Figure A.1). Correct mouth open/close prediction is essential for successful inside-mouth bite transfer, as proved in our mouth perception ablation study's Scenario 2.

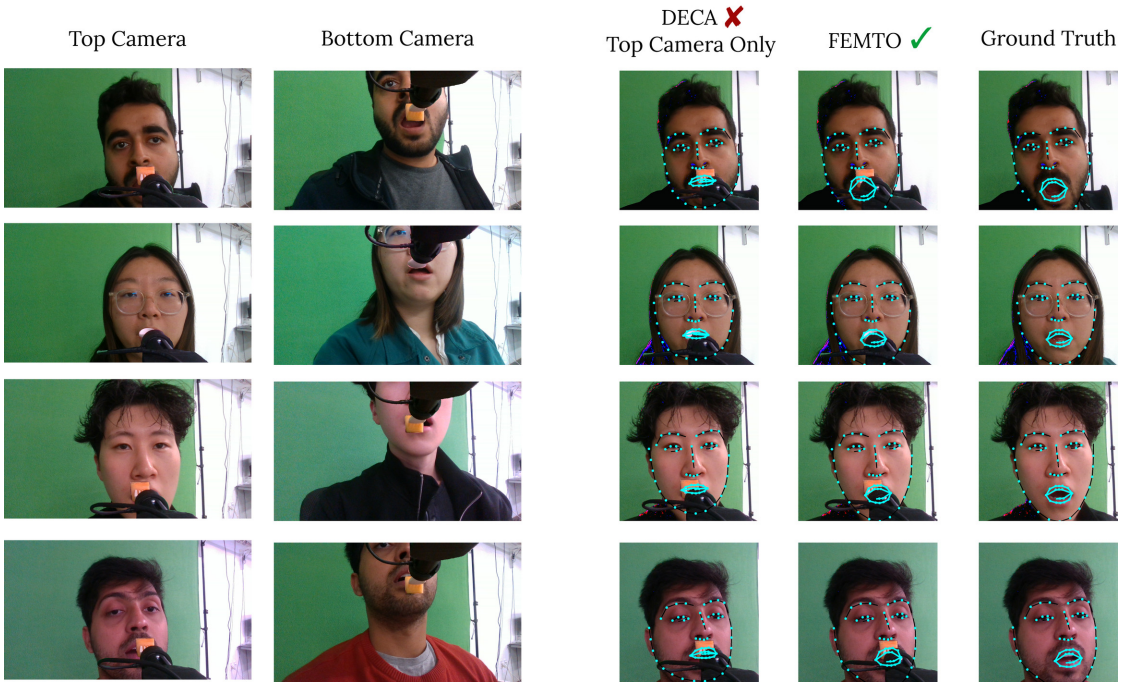


Figure A.1: Qualitative Analysis: DECA often incorrectly predicts closed mouth when it is actually open.

A.4 Physical-Interaction Aware Control

A.4.1 Physical Interaction Classification

Problem Formulation

Our method initiates upon detecting contact through the Force-Torque (FT) sensor. Following contact detection, the system pauses for 100 milliseconds to accumulate sensor data pertinent to the specific physical interaction. This data is then classified into one of four categories: incidental interaction, in-mouth manipulation, impulsive interaction, or bite. Given that the FT sensor operates at a frequency of 1kHz, it gathers a haptic observation $H_t \in \mathbb{R}^{100\times6}$. For the visual component, we utilize data from the FEMTO buffer, which includes

both 3D keypoints representing the mouth's position in world space and 2D keypoints depicting the mouth in image space. The visual observation is defined as $I_t \in \mathbb{R}^{3\times5\times20}$, where 20 are the number of keypoints being tracked on the mouth. The '3' in our visual observation's dimensionality represents past timesteps, chosen to capture the current visual state and the interaction's initiation history.

Data Collection

We gathered interaction data for six participants being fed with a fork (cantaloupe) and spoon (yogurt). Participants were provided specific instructions to induce certain contact types, ensuring a comprehensive capture of diverse scenarios for each participant as detailed below. For each utensil, we collect the following:

Goal-Tracking Controller	**Force-Tracking Controller**
Incidental interactions while moving inside mouth (64 datapoints)	(Incidental interactions do not occur with force-tracking controller)
- Lips closed, teeth closed (16 datapoints)	
- Lips open, teeth closed (16 datapoints)	
- Lips closed, teeth open (16 datapoints)	
- Lips open, teeth open (16 datapoints)	
Up/down impulsive interactions (32 datapoints)	Up/down impulsive interactions (32 datapoints)
- Tilt up (16 datapoints)	- Tilt up (16 datapoints)
- Tilt down (16 datapoints)	- Tilt down (16 datapoints)
In-mouth manipulation using tongue (32 datapoints)	In-mouth manipulation using tongue (32 datapoints)
- Move food to left, mouth open (8 datapoints)	- Move food to left, mouth open (8 datapoints)
- Move food to right, mouth open (8 datapoints)	- Move food to right, mouth open (8 datapoints)
Bite interactions (32 datapoints)	Bite interactions (32 datapoints)
- Enclosed, teeth + lips (8 datapoints)	- Enclosed, teeth + lips (8 datapoints)
- Enclosed, lips only (8 datapoints)	- Enclosed, lips only (8 datapoints)
- Enclosed, teeth only (8 datapoints)	- Enclosed, teeth only (8 datapoints)
- Partly out (8 datapoints)	- Partly out (8 datapoints)

In total, we collect 3072 datapoints, evenly distributed between the four physical interaction classes.

Feature Extraction

Following established practices in haptic classification, we extract six key time-series features from the haptic data. Based on our ablation study of the features suggested in [68], we chose the following: mean, standard deviation (std), Hjorth complexity, energy, skewness, and kurtosis. These features are extracted from each of the six force-torque channels over 100ms local time-windows, resulting in the haptic feature vector $H_{computed} \in \mathbb{R}^{6\times6}$. No additional features are extracted from the visual data, as they do not enhance performance. Consequently, our time-series classification models use the features H_t, $H_{computed}$, and I_t.

Models

We evaluate four models: Support Vector Machines (SVMs), Multi-Layer Perceptron (MLP), Temporal Convolutional Networks (TCN), and Time-Series Transformers (TST). For SVMs, we compare two variants: SVM-Raw, which utilizes H_t and I_t, and SVM-features, employing $H_{computed}$ and I_t. Prior to SVM processing, we flatten all features. For TCN and TST models, we interpolate the visual features to align with the haptic features' time-steps (100). For finetuning with SVMs, we give 20x higher weights to the novel users' datapoint while computing the decision boundary.

Evaluation

We evaluate four different scenarios:

(1) Training on 80% and testing on 20% of the same user's data.

Table A.1: Performance (F1 score) of physical interaction classification methods for inside-mouth bite transfer.

Method	A - Training Users			B - Novel Users			C - Finetuning			D - Pretraining		
	All	Haptic	Visual	All	Haptic	Visual	All	Haptic	Visual	All	Haptic	Visual
SVM-Raw	**0.908**	0.822	**0.864**	0.831	0.786	0.659	0.867	0.793	**0.785**	0.920	0.827	**0.896**
SVM-Features	0.899	**0.843**	**0.864**	**0.865**	**0.833**	0.659	**0.891**	**0.852**	**0.785**	**0.929**	**0.878**	**0.896**
MLP	0.868	0.817	0.832	0.863	0.824	**0.703**	0.869	0.832	0.737	0.906	0.862	0.884
TCN	0.811	0.776	0.688	0.863	0.772	0.620	0.867	0.788	0.765	0.878	0.801	0.760
TST	0.889	0.789	0.819	0.852	0.809	0.686	0.867	0.773	0.753	0.912	0.831	0.858

(2) Training on data from five users and testing on 20% of a sixth, novel user's data.

(3) Training on five users, finetuning with about 10 physical interaction datapoints per type from a novel user, and then testing on 20% of the same novel user's data.

(4) Pretraining on five users, finetuning on 80% of a novel user's data, and testing on the remaining 20% of this user's data.

Performance comparisons are provided for all models using both haptic and visual features, as well as their ablations for all scenarios. These results are averaged over three seeds. For each seed, we average results for all users in Scenario A and do Leave-One-User-Out testing for each user for Setting B, C and D. Across all scenarios, SVMs exhibit superior performance, likely attributable to the low data regime.

A.4.2 Controllers

Goal-Tracking Controller

For pose-tracking using torque-control for the Kinova Gen3, we developed a task-space compliant controller integrated with gravity compensation and

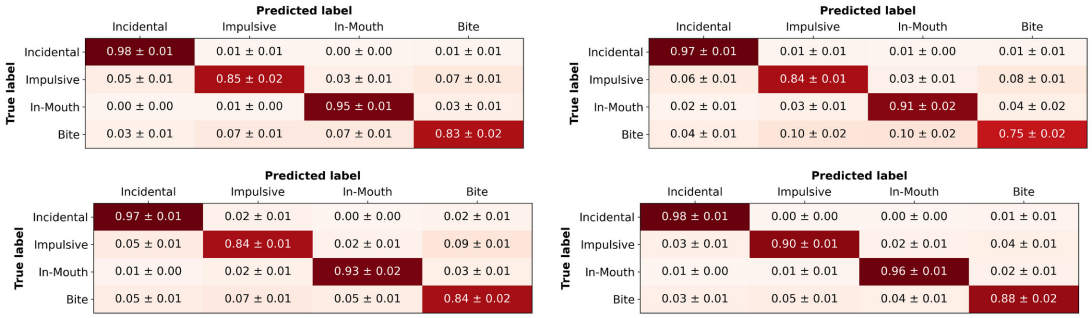


Figure A.2: Confusion matrices visualizing SVM-Features' performance using both haptic and visual modalities across all scenarios, from Left to Right, Top to Bottom: A - Training Users, B - Novel Users, C - Fine-tuning, and D - Pretraining.

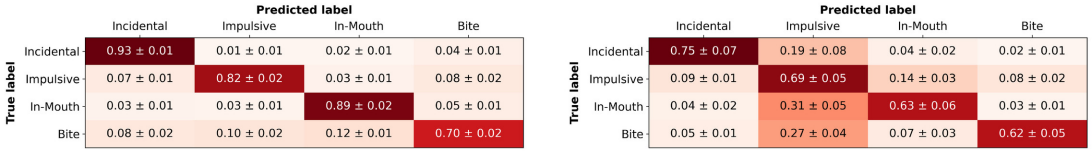


Figure A.3: Confusion matrices visualizing SVM-Features' performance with only haptic (left) and only visual (right) modality for Scenario B - Novel Users. In this scenario, the visual modality excels in distinguishing incidental interactions from other types, primarily because incidental interactions occur outside the mouth, while all other interaction types take place inside the mouth. The haptic modality faces challenges in distinguishing between impulsive (up-down) and bite interactions, as both involve sudden forces along the same axis of the force-torque sensor.

model-free friction observers. This controller operates by distinguishing between the ideal motor signal, anticipated in a frictionless scenario, and the actual motor signal, thereby enabling precise friction compensation. This enhancement significantly smoothens the control of the Gen3 arm, which typically faces challenges in performance owing to unaccounted but significant joint friction during torque control. This task-space compliant controller tracks pose by a PID term between pose and torque, with the I term capped to ensure safety. Thus, the

torque computation for the goal-tracking task is:

$$\tau_{task} = -\mathbf{J}^T \left(\mathbf{K}_p \mathbf{e} - \mathbf{K}_d \frac{d\mathbf{e}}{dt} - \mathbf{K}_i \, \text{cap} \left(\int \mathbf{e} \, dt, \text{limit} \right) \right) \tag{A.1}$$

In this equation, τ_{task} denotes the task torque, and the error term $\mathbf{e}$, representing the difference between the nominal and desired poses, is crucial in friction-observers based control. The proportional, derivative, and integral terms are scaled by their respective gain matrices $\mathbf{K}_p$, $\mathbf{K}_d$, and $\mathbf{K}_i$. The integral term is capped by $\text{cap}(\cdot, \text{limit})$ to prevent excessive accumulation and ensure stability.

Our mouth perception pipeline can be used to generate a goal pose inside the user's mouth, such as a point 2 cm within, aligned with the center of the mouth opening axis. However, using a standardized transfer pose for all users can result in frequent incidental interactions due to the unique anatomical variations in each individual's oral cavity, including differences in teeth and tongue positioning. This was evident from our preliminary tests. To mitigate these interactions, we record a customized bite transfer position for each new user right at the outset. While our system is equipped to detect and respond to incidental interactions, proactively minimizing them through this personalized approach not only enhances effectiveness but also ensures a more comfortable feeding experience for the user.

Force-Tracking Controller

For our force-tracking controller, our objective is to achieve high compliance and responsiveness to user interactions, allowing the robot to move as guided by the user. To facilitate this, we integrate an additional term into the goal-tracking controller. This term utilizes feedback from the Force-Torque (FT) sensor, directing

the end-effector to move in the applied force's direction with PD control:

$$\tau_{\text{force}} = \mathbf{J}^T \left(\mathbf{K}_p^f \mathbf{f}_m + \mathbf{K}_i^f \, \text{cap} \left(\int \mathbf{f}_m \, dt \, , \text{limit}^f \right) \right) \tag{A.2}$$

In this equation, τ_{force} denotes the force-tracking torque, with $\mathbf{f}_m$ being the measured force. The terms $\mathbf{K}_p^f$ and $\mathbf{K}_i^f$ are the proportional and integral gains specifically for force tracking, and the integral term is capped by $\text{cap}(\cdot, \text{limit}^f)$ to ensure stability and prevent excessive accumulation.

Additionally, to ensure that the torques generated by τ_{task} do not interfere with force-tracking control, we periodically reset the goal pose to the robot's current pose at a low frequency. This strategy is crucial for force-tracking control combined with stable friction compensation. We cannot completely remove the term τ_{task}, as it is integral for effective friction compensation, which is a key requirement for achieving precise torque control in the Kinova Gen3 system.

A.5 Ablation I: Mouth Perception User Study

A.5.1 Study Design and Implementation

Study Design Rationale

In this study, we aim to evaluate the necessity of continuous, real-time mouth perception throughout the entirety of the inside-mouth bite transfer process. Previous research [70, 244] perceives the mouth of user once before initiating bite transfer, thereafter assuming the user remains stationary for the duration of the transfer, which is a significant duration of time. We use these approaches as the baseline (one-time mouth perception) in our study. Initial discussions with

end-users and occupational therapists revealed that head movements – both voluntary and involuntary – are common during inside-mouth bite transfer. In collaboration with them, we have identified three main scenarios where head movement often occurs while the user is being fed:

(1) Voluntary Head Movement: Users may move their head voluntarily during the transfer and expect the robot to adjust to this movement. For instance, one participant (P12) mentioned, "I could be sitting in front of the TV watching something that I'm ready to take a bite and then it catches my attention," while another (P7) said, "I could have a conversation with somebody and just go like this [shows sudden turning of head]." We simulate this scenario by asking participants to voluntarily turn their head as the robot approaches.

(2) Talking: Participants highlighted situations where they might start talking while being fed. P7 explained, "On a normal day, you're with somebody and you're looking away and you're talking, [while] you're eating." To emulate this scenario, we ask participants to recite a sentence while the robot is moving towards them.

(3) Involuntary Head Movement: Some end-users who experience spasms raised concerns about how the robot would react if they spasm towards it. We simulate this scenario by asking participants to simulate an impulsive forward motion, similar to the spasms observed in end-user P1, while the robot is in motion.

Note: The project website[1] includes a video demonstrating the robot's response to user P1's unexpected spasm. Prior to this incident, captured

[1] `https://emprise.cs.cornell.edu/bitetransfer`

during a preliminary study, the user had not informed us that they have spasms. This incident served as a key motivator for incorporating muscle spasms into our research. At the time of recording, the robot had not been equipped with the utensil retraction on detecting spasms feature. Consequently, in the video, the robot tries to maintain a consistent approach speed towards the user's mouth. When the user experiences a spasm, the robot's real-time perception adjusts the goal pose, causing the utensil to move back in sync with the user, thereby ensuring their safety. This incident underlines the critical importance of real-time mouth perception for safely transferring bites inside the mouth.

Implementation Details

(1) Voluntary Head Movement: The user's mouth is tracked using FEMTO. The goal-tracking controller's goal pose is continuously updated to a recorded user-preferred transfer pose inside of their mouth.

(2) Talking: FEMTO is utilized to ascertain if the user is talking by monitoring the mouth's status (open or closed) as the robot approaches. If the mouth is detected as closed, the robot pauses by maintaining its current pose as the goal for the goal-tracking controller. This pause lasts until the user's mouth is perceived as open continuously for a set duration (2 seconds in our system), after which the robot resumes moving the utensil towards the user's mouth.

(3) Involuntary Head Movement: Spasms are detected by tracking the head's velocity during the robot's approach. A spasm is identified when the velocity crosses a specific threshold, prompting the robot to retreat by setting

its goal pose 30 cm behind its current pose. After waiting for a fixed period (5 seconds in our implementation), the feeding process is resumed.

Note that FEMTO uniquely enables real-time tracking of both the mouth pose (indicating where to feed) and mouth state (determining when to feed). To the best of our knowledge, no other method can continuously track these aspects while approaching the user, due to the occlusion caused by the utensil.

A.5.2 Pilot Study

We performed a pilot study with 3 participants, which indicated that 8 participants were needed to reach statistical significance ($p < 0.05$ with power = 0.8). As a precaution, we recruited double the amount of people (16 people). One participant's data was discarded prior to analysis due to difficulties following the instructions for the experiments.

A.5.3 Questionnaires

The pre-study questionnaire, as well as the questions asked for each of the three scenarios are shown in Fig. A.4.

A.6 Ablation II: Physical Interaction-Aware Control User Study

A.6.1 Study Design Rationale

In our study, we seek to evaluate the effectiveness of our interaction-aware control system. The study is modeled on the scenario of "Jonathan" (detailed in

Pre-Study Questionnaire

Do you have any prior experience with robots?

- ○ Yes
- ○ No

Scenario 1: Voluntary Head Movement

I am satisfied with the last feeding trial.

- ○ Strongly Disagree
- ○ Disgree
- ○ Neutral
- ○ Agree
- ○ Strongly Agree

I felt safe during the last feeding trial.

- ○ Strongly Disagree
- ○ Disgree
- ○ Neutral
- ○ Agree
- ○ Strongly Agree

I felt comfortable during the last feeding trial.

- ○ Strongly Disagree
- ○ Disgree
- ○ Neutral
- ○ Agree
- ○ Strongly Agree

Based on the last trial alone, I am confident the robot can bring the food item into my mouth unassisted by me.

- ○ Strongly Disagree
- ○ Disgree
- ○ Neutral
- ○ Agree
- ○ Strongly Agree

In the last trial, I felt I had control over the robots actions.

- ○ Strongly Disagree
- ○ Disgree
- ○ Neutral
- ○ Agree
- ○ Strongly Agree

In the last trial, I had to reposition my head to get the food item from the robot.

- ○ Strongly Disagree
- ○ Disgree
- ○ Neutral
- ○ Agree
- ○ Strongly Agree

Scenario 2: Voluntary Speaking

I am satisfied with the last feeding trial.

- ○ Strongly Disagree
- ○ Disgree
- ○ Neutral
- ○ Agree
- ○ Strongly Agree

I felt safe during the last feeding trial.

- ○ Strongly Disagree
- ○ Disgree
- ○ Neutral
- ○ Agree
- ○ Strongly Agree

I felt comfortable during the last feeding trial.

- ○ Strongly Disagree
- ○ Disgree
- ○ Neutral
- ○ Agree
- ○ Strongly Agree

In the last trial, I felt I had control over the robots actions.

- ○ Strongly Disagree
- ○ Disgree
- ○ Neutral
- ○ Agree
- ○ Strongly Agree

In the last trial, I felt the robot disrupted me while I was talking.

- ○ Strongly Disagree
- ○ Disagree
- ○ Neutral
- ○ Agree
- ○ Strongly Agree

Scenario 3: Involuntary Head Movement

I am satisfied with the last feeding trial.

- ○ Strongly Disagree
- ○ Disgree
- ○ Neutral
- ○ Agree
- ○ Strongly Agree

I felt safe during the last feeding trial.

- ○ Strongly Disagree
- ○ Disgree
- ○ Neutral
- ○ Agree
- ○ Strongly Agree

I felt comfortable during the last feeding trial.

- ○ Strongly Disagree
- ○ Disgree
- ○ Neutral
- ○ Agree
- ○ Strongly Agree

The robot responded to my impulsive motion.

- ○ Strongly Disagree
- ○ Disgree
- ○ Neutral
- ○ Agree
- ○ Strongly Agree

Figure A.4: Questions for Ablation I: Mouth Perception User Study

Appendix A.1). Jonathan is unable to bite in the center of his mouth and prefers soft and sticky food items, like cantaloupe and yogurt, to be placed on the right side of his mouth. When a caregiver initially places the food item at the center, Jonathan uses his tongue to move the food to the right side before taking a bite. This preference, once demonstrated, is expected to be remembered by the caregiver for subsequent feedings. Additionally, Jonathan experiences involuntary head movements, such as impulsive tilting up and down twitches, which can occur during eating.

The study is designed to have participants simulate Jonathan's condition. A trial comprises of two distinct feedings:

- Feeding 1: Participants demonstrate Jonathan's preferred position for receiving a bite. This serves to inform the robot of the specific feeding pose Jonathan needs.
- Feeding 2: Participants expect the robot to remember and automatically adjust to Jonathan's preferred feeding position, while also dealing with an unexpected impulsive head movement which occurs when the utensil is inside their mouth.

This setup allows us to compare how well interaction-aware control methods adapt to the nuanced needs and movements of individuals similar to Jonathan, enhancing the real-world applicability of assistive feeding technologies.

A.6.2 Pilot Study

We perform a pilot study with 3 participants, which indicated that 7 participants were needed to reach statistical significance ($p < 0.05$ with power = 0.8).

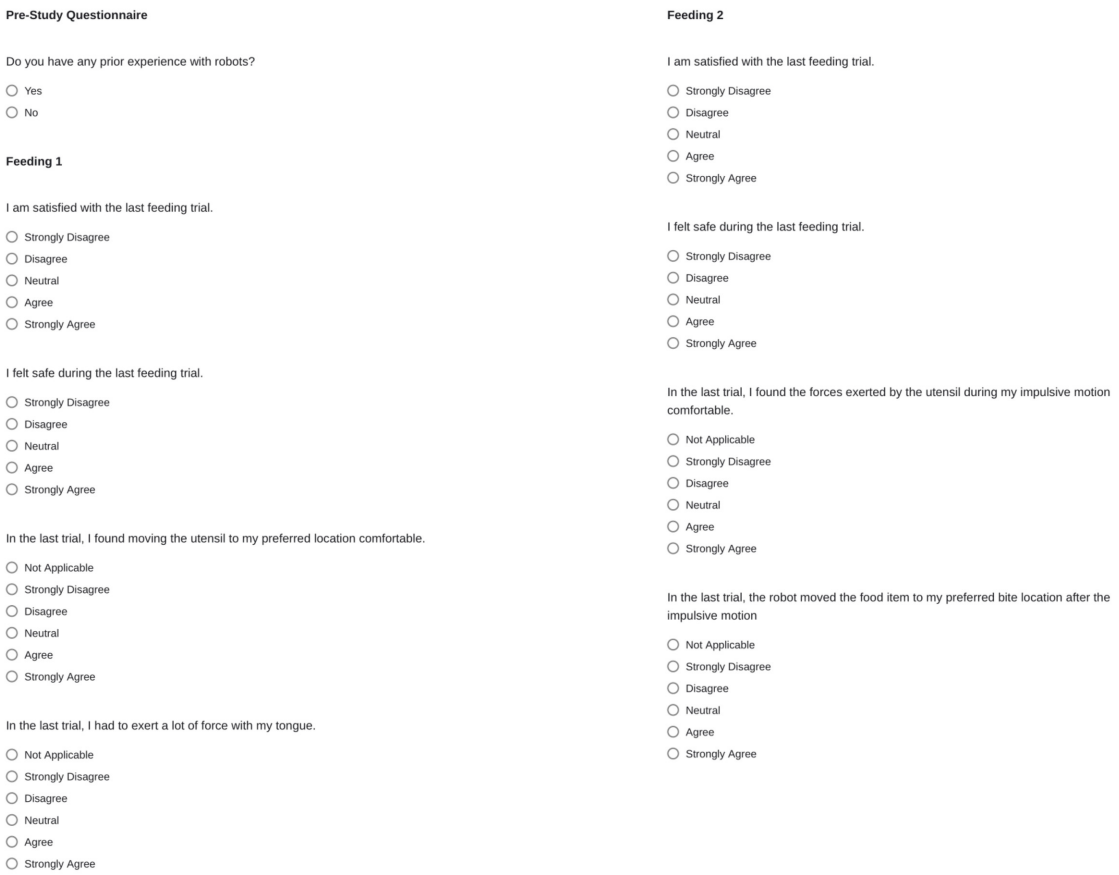


Figure A.5: Questionnaire for Ablation II: Physical-Interaction-Aware Control User Study

To maintain a number consistent with the last ablation study, we recruited 15 people. However, the study was terminated mid-way for one participant due to difficulties following the instructions for the experiments.

A.6.3 Questionnaires

The pre-study questionnaire, as well as the questions asked after the first feeding and second feeding are shown in Fig. A.5.

Table A.2: SVM-Features' performance (F1 Score) on interaction classification ablation study data.

	Without Finetuning	With Finetuning
SVM-Features		
- Haptic + Visual	0.718 (+/- 0.019)	0.794 (+/- 0.017)
- Haptic	0.675 (+/- 0.020)	0.750 (+/- 0.020)
- Visual	0.326 (+/- 0.028)	0.588 (+/- 0.027)

A.6.4 Interaction Classification Results

In this ablation study, we assess the performance of our interaction classification methods using data we collected. We automatically annotated the ablation study data using the robot's audio cues provided to participants as a reference. The first detected contact after a cue is classified as follows: an 'impulsive interaction' for the "impulse" cue, an 'in-mouth manipulation' for the "push" cue, and a 'bite' for the "bite" cue. Any other interactions are categorized as incidental. The performance of the SVM-features model, trained on the data detailed in Appendix A.4.1 and tested on each participant's interaction data from this study, is presented in Table A.2. We also report the model's performance after further fine-tuning on 80% of each participant's data (approximately 40 interactions) and testing on the remaining 20% of their data. Evaluations are conducted over three seeds and averaged across all participants of this ablation study. Consistent with findings in Appendix A.4.1, employing both haptic and visual modalities yields better results than using either modality alone, with the haptic modality outperforming the visual modality when used individually.

We've identified two reasons for the observed performance drop compared to the results in Appendix A.4.1: (i) the auto-annotation process may produce

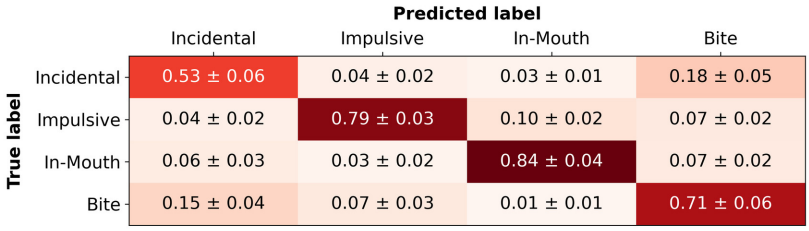


Figure A.6: Confusion matrix for multimodal SVM-Features classification on ablation study data with finetuning.

somewhat noisy labels due to users' errors in following instructions, and (ii) distribution shifts owing to changes in the robot system since the Appendix A.4.1 data collection and this ablation study. For instance, in earlier data collections, our feeding system adjusted the utensils according to the user's mouth pitch (affected by neck flexion/extension). However, based on user feedback, we later modified the feeding utensil to always remain horizontal, maintaining parallel alignment with the ground. Addressing these types of distribution shifts presents a valuable direction for future research in interaction classification methods.

A.7 Care Recipient User Study

A.7.1 Next Challenges in Inside-Mouth Bite Transfer

Through our comprehensive user study encompassing people with a broad range of mobility limitations, we have uncovered several opportunities for further enhancing our inside-mouth bite transfer system:

Longitudinal Studies on Evolving User Preferences and Trust in Robot Feeding Systems. There is need for longitudinal (long-term) studies to understand how user preferences and trust in robot-assisted feeding systems evolve

over time. As users become more accustomed to the robot's functionalities and safety measures, it may be interesting to see how a user's trust and technology acceptance may change over time and over repeated exposure to a robot's capabilities. As an example, end-user P6, due to their mobility limitations, can comfortably take bites of solid food items only at their right molar. Their human caregiver typically moves food to the center of P6's mouth before shifting it to the right side, a technique that allows for larger bite sizes to pass through their narrow mouth opening. However, P6 strongly preferred the robot to move the food item directly to their right molar, without moving through the center. This was because P6 was unsure about the robot's capability to prevent gag reflex when the utensil would touch their tongue. This is interesting because our robot is indeed equipped with the precision to mimic their human caregiver's movements, even though the user may not perceive that to be the case. A prolonged exposure to the robot's capabilities may change their preference.

Addressing Complex Medical Conditions. Our inside-mouth bite transfer system effectively accommodates users with a variety of medical limitations, as demonstrated in this study. However, we recognize that certain care-recipients with more complex medical conditions present additional challenges. For instance, individuals with advanced neurological disorders, such as severe Alzheimer's disease, Parkinson's disease, or those recovering from a major stroke, may struggle to initiate the swallowing reflex. In these situations, caregivers often need to touch a specific area in the mouth with a utensil to stimulate the swallowing reflex. This requirement introduces a unique physical interaction that our current system does not yet address, highlighting an area for future exploration and development.

A.7.2 Questionnaires

The pre-study questionnaire, questions on safety and comfort for both inside-mouth and outside-mouth bite transfers, the comparison between these methods, and the Technology Acceptance Model (TAM) survey are presented in Fig. A.7.

Pre-Study Questionnaire

What is your medical condition?

Since how long have you had this medical condition?

Please describe any challenges or difficulties you experience when a caregiver assists in transferring food from a utensil to your mouth. Are there any specific concerns or moments of discomfort during this process?

Do you have any familiarity with robotic/assistive technologies? If yes, please share your experience.

Inside-Mouth Robot-Assisted Bite Transfer

I feel safe during inside-mouth bite-transfer.

- ○ Strongly disagree
- ○ Somewhat disagree
- ○ Neither agree nor disagree
- ○ Somewhat agree
- ○ Strongly agree

I feel comfortable during inside-mouth bite-transfer.

- ○ Strongly disagree
- ○ Somewhat disagree
- ○ Neither agree nor disagree
- ○ Somewhat agree
- ○ Strongly agree

Outside-Mouth Robot-Assisted Bite Transfer

I feel safe during outside-mouth bite-transfer.

- ○ Strongly disagree
- ○ Somewhat disagree
- ○ Neither agree nor disagree
- ○ Somewhat agree
- ○ Strongly agree

I feel comfortable during outside-mouth bite-transfer.

- ○ Strongly disagree
- ○ Somewhat disagree
- ○ Neither agree nor disagree
- ○ Somewhat agree
- ○ Strongly agree

Comparison

Imagine using this system in your daily routine where you take approximately 100 bites across meals like breakfast, lunch, and dinner.

Between outside-mouth bite-transfer and inside-mouth bite-transfer, which one do you prefer and why?

Would your above specified preference change in any specific situation?

Between outside-mouth bite-transfer and inside-mouth bite transfer, which one requires more physical effort and why?

TAMs - Inside-Mouth Robot-Assisted Bite Transfer

A robot which can do inside-mouth bite-transfer will help me to become more independent in eating

- ○ Strongly disagree
- ○ Somewhat disagree
- ○ Neither agree nor disagree
- ○ Somewhat agree
- ○ Strongly agree

A robot which can do inside-mouth bite-transfer is easy to use.

- ○ Strongly disagree
- ○ Somewhat disagree
- ○ Neither agree nor disagree
- ○ Somewhat agree
- ○ Strongly agree

Using a robot which can do inside-mouth bite-transfer to improve my independence would be a good idea.

- ○ Strongly disagree
- ○ Somewhat disagree
- ○ Neither agree nor disagree
- ○ Somewhat agree
- ○ Strongly agree

Assuming I have access to a robot which can do inside-mouth bite transfer, I predict that I would use it in my daily life.

- ○ Strongly disagree
- ○ Somewhat disagree
- ○ Neither agree nor disagree
- ○ Somewhat agree
- ○ Strongly agree

Inside-mouth bite transfer is enjoyable.

- ○ Strongly disagree
- ○ Somewhat disagree
- ○ Neither agree nor disagree
- ○ Somewhat agree
- ○ Strongly agree

Open-Ended Suggestions

For the method you preferred, do you have any suggestions on improvements?

Figure A.7: Questionnaire for Care Recipient User Study.

APPENDIX B

FLAIR APPENDIX

B.1 Hardware System Details

The experiments described in this work have been conducted at two different universities, utilizing three different robot arms, each equipped with a specialized fork attachment and sensors. The design for the fork attachment, employed by both institutions, is adapted from Shaikewitz et al. [244] to accommodate the Robotiq 2F-85 gripper and to facilitate effective cutting actions. This fork attachment is illustrated in Chapter 5 (Fig. 5.2) and is publicly available as open-source hardware on the website[1]. Specifically, the system at Stanford University employs both a Kinova Gen3 7-DoF arm and a Franka Emika Panda 7-DoF arm, both mounted directly onto tables (Fig. 5.2). In contrast, the setup at Cornell University utilizes a Kinova Gen3 6-DoF robotic arm, which is mounted on a Vention stand adjacent to a feeding table.

Both institutions perform experiments to showcase how our bite acquisition framework can be integrated with an outside-mouth bite transfer method [70] with visual servoing (Fig. B.4). Additionally, we use the setup at Cornell University to feed a care recipient with Multiple Sclerosis, demonstrating integration with an inside-mouth bite transfer method [120], which is necessary for feeding people with severe mobility limitations who cannot lean forward to take a bite. The successful execution of acquisition and transfer trials at both institutions, utilizing different robotic setups, underscores the reproducibility of our methodology and hardware.

[1] `http://emprise.cs.cornell.edu/flair`



Figure B.1: Prompt images used for food item recognition with a VLM.

B.2 Prompting Details

B.2.1 Food Detection

We use GPT-4V [1] to generate a set of candidate labels for open-set object detectors, using the following prompt with accompanying images shown in Fig. B.1:

```
Objective: To determine the food items on the plate.
Available items: ['shrimp', 'chicken', 'noodles',
'chicken', 'broccoli', 'fettucine', 'meatball',
'spaghetti', 'brownie', 'chocolate sauce', 'strawberry',
'celery', 'watermelon', 'ranch dressing',
'mashed potatoes', 'sausage']
The first 3 images are in-context examples with the ground
truth food items.
Example Image 1: ['shrimp', 'chicken', 'noodles']
Example Image 2: ['chicken', 'broccoli', 'fettucine']
Example Image 3: ['meatball', 'spaghetti']
```

```
The 4th image is the test image. From the available items,
what food items are present, as a comma-separated list?
Format your answer in one-line as follows: ['first_item',
'second_item', ..., 'last_item']
```

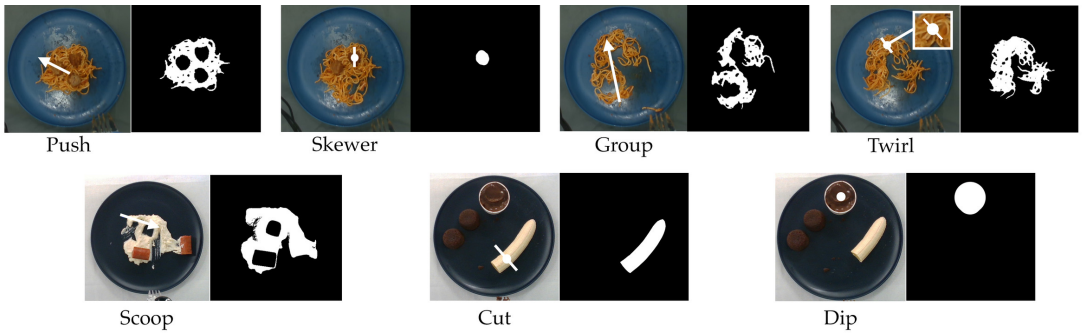


Figure B.2: **Skill Parameterizations**: We visualize the parameterization of each skill in FLAIR's skill library. White arrows indicate the direction of pushing, grouping, or scooping, while a white dot with a line through it represents the intended fork position and orientation, respectively.

B.2.2 Bite Sequencing

We use the following prompt for bite sequencing, instructing the language model to balance user preferences, commonsense orderings, and efficiency. For Preference-Only, we omit efficiency from the prompt. While LLMs offer powerful commonsense and few-shot reasoning, they sometimes produce unrealistic outputs or overlook critical context, such as manipulation efficiency and remaining portions of each food item. To guide the LLM's reasoning, we use in-context examples and meticulously identify and analyze potential failure modes through extensive pre-study testing. We update our prompt to explicitly instruct the LLM to reduce these errors (underlined).

```
You are feeding me a plate of food.
Items remaining: %s
Portions remaining (corresponding to items remaining
list): %s
Efficiency: %s
Preference: %s
Dipping sauces remaining: %s
---
Given this information, you will decide what bite to
feed me next. You may ONLY suggest a bite from the
available 'Items remaining', optionally dipped in an
```

item from 'Dipping sauces.'
1) First, summarize what food items are remaining on the plate with corresponding portions. Make sure that your summarization matches with the 'Items remaining' and 'Portions remaining' provided at the beginning.
2) Next, summarize your high-level strategy for feeding me. Describe the order of bites you would use to feed me the remaining items, and explain how you make your decision. Note that bites taken so far do not affect items remaining.
Decision Criteria:
- Prioritize my preference above all else. If I have a strong preference for a particular ordering of foods, you should obey that as best as possible. Respect user preferences for not dipping certain food items in certain sauces. If the preference specifies 'Feed me X and then Y', perceive it as me asking you to feed all of X before you feed any of Y.
- "Alternating between X and Y" means you can start with either X or Y, unless otherwise specified. If one item is more efficient, start with that.
- In the past, I have eaten the following bites: %s. The first item of this list is the first bite I had, the second item is the second bite, and so on. The last element is the most recent bite that I just had. Use this to inform what you feed me next. Note that bites taken so far are past bites and thus do not affect the specified items or portions remaining on the plate.
- In your answer, state the portions of each item remaining. Ensure that this matches with the 'Portions remaining' provided at the beginning. Note that these items haven't been consumed. If I do not have a strong preference, use commonsense. For dishes like noodles with vegetables and meat, or porridge with different toppings, prioritize feeding me a MIX of different items. This means do not feed me the same item consecutively twice if it doesn't affect efficiency. The exception to this rule is if there is a lot more of one item than another. In that case, you should prioritize feeding me the item with more quantity, until the portions of items left are more evenly distributed, even if I have eaten the larger portion item consecutively. If only a single portion of an item is left, do not feed it but leave it until later for better bite variety (unless the preference instructs otherwise). Lastly, use your best judgment to figure out what ordering makes sense. For instance, looking at a plate of carrots, ranch, cantaloupe, apples, and caramel, you should be able to tell that apples go with the caramel, and carrots go with the ranch, and that apple should not be dipped in ranch because that is not typical, and cantaloupe and other melon types should be eaten plain. Avoid feeding savory foods with sweet sauces, and vice versa (i.e. chicken

nugget and chocolate sauce is NOT a good suggestion, apple should not be dipped in ketchup; cantaloupe, honeydew, and melons are usually eaten plain without sauce). A good rule of thumb is to prioritize: common pairings (i.e. sweet savory), portion sizes, and bite variability (in that order).
3) Next, choose an item from 'Items available' to feed next, based on your answer to 2). Examples: "Feed shrimp", "Feed apple dipped in caramel", "Do not feed a bite". You can either feed a single food item which appears in 'Items remaining' ("Feed shrimp"), a single food item dipped in something ("Feed apple dipped in caramel"), or opt to not feed an item ("Do not feed a bite"); only do this if I strongly dislike the remaining bites, or I only requested a 2 bites of 'tiramisu,' for example, at the end of a meal, and you already fed me both (as mentioned in 'The bites I have taken so far'). You CANNOT feed a dip (i.e. ranch, mustard, whipped cream, bbq sauce, chocolate sauce, etc.) by itself. You can ONLY feed an item if it is present in 'Items remaining.' For example, given 'Items remaining: ["broccoli", "asparagus"]', you cannot suggest "Feed pasta", even if I have taken a bite of 'pasta' in the past.
4) Now, consider efficiency; in the numbers listed above, higher means that food item is more efficient to pick up. Would you change your answer to 3) given this information? If the portions are even (difference of <3 bites), and my preference is weak or alternating, and one item is clearly more efficient, you may change your answer. Otherwise, do not change your answer.
5) Now, summarize your planned bite from 4) in a list format. You can either output a single item in a list ['item'], where 'item' is chosen from 'Items remaining', if you would like to feed this item by itself. Or, you can output two items in a list ['item1', 'item2'], if you would like to dip 'item1' (from 'Items remaining') in 'item2' (from 'Dipping sauces remaining') i.e. ['banana', 'nutella']. However, you cannot suggest to feed 2 items if one of them is not a dip (i.e. ['ramen', 'beef'] or ['rice', 'chicken'] or ['linguini', 'mushroom'] are NOT allowed, you must feed ['ramen'] or ['beef'] by itself). Lastly, you can output an empty list [] if there is absolutely no bite that makes sense.

Format your response as follows, be concise but thorough: Food Items Left: <Sentence describing what food items are left on the plate; ensure that this matches with the 'Items remaining' provided at the beginning>

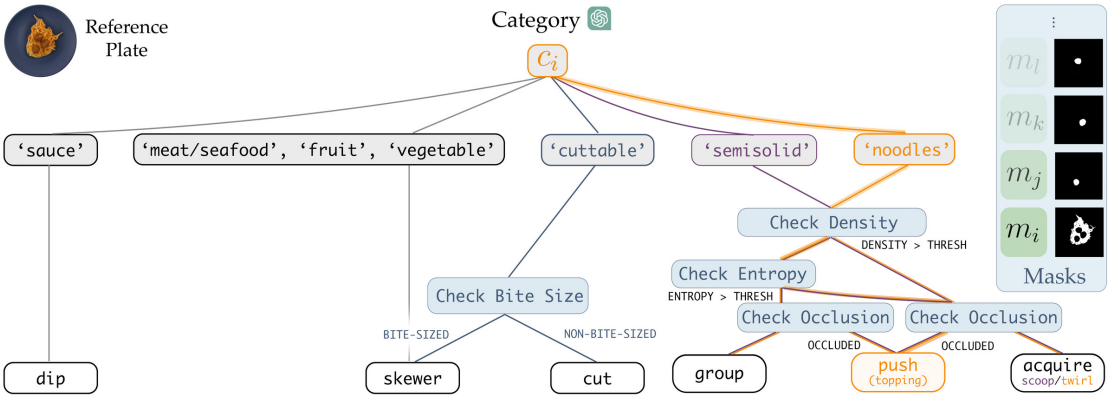


Figure B.3: **Task Planner Decision Tree:** For this reference plate and input category `'noodles'`, the task planner navigates the decision tree to decide to push aside a meatball to access the bed of spaghetti underneath. Nodes in blue leverage the segmented food item states to determine branching.

```
Strategy: <Sentence describing your high-level
strategy>
Next bite: <Phrase describing the next bite you plan to
feed>
Next bite (accounting for efficiency): <Phrase
describing the next bite you plan to feed, taking into
account efficiency>
Next bite as list: ['item1'] # Or ['item1', 'item2'] or
[]
---
Output your response here. Ensure that the last line
begins with 'Next bite as list:
```

B.3 Skill Parameterization Details

At the core of FLAIR is a large library of vision-parameterized skills for both acquisition and pre-acquisition. We visualize each skill parameterization in Fig. B.2.

The `twirl` skill and the end position for the `scoop` and `group` skills are parameterized by the densest point in a pile of noodles or semisolid mass. To sense

the densest point, we take a mask m_t, apply a 2D Gaussian filter over the image, and consider the densest pixel to be the argmax of this heatmap. The `twirl` skill is additionally parameterized by the main-axis orientation of the pile of noodles at the densest point. To determine this, we use the pose estimation network from [256] which maps a local crop of noodles to the predicted main axis orientation, and insert the fork orthogonal to this. Parameterizing the skill in this way encourages strands of noodles to be perpendicular and wrapped around the fork tines during twirling, rather than parallel and slipping off. The start positions for the `scoop` and `group` skills are parameterized by the sparsest point of a noodle or semisolid mask. To estimate this, we select the pixel on the segmentation mask that is furthest from the sensed densest point. For noodle or semisolid piles containing toppings, we update this parameterization to take into account the position of the fillings, such that the downstream skill execution is not affected by the toppings.

We parameterize the `dip` and `skewer` skills by the centroid of a predicted segmentation mask. The `skewer` skill is additionally parameterized by the mask's major-axis orientation, such that the tines of the fork run orthogonal to the item. The `push` skill starts at a constant offset from the centroid of the obstructing topping and pushes it to the nearest boundary point of the underlying mask. The fork tines are oriented perpendicular to the direction of the pushing action.

For the `cut` skill, we aim to locate a bite-sized portion from an item. We first estimate the long axis of a food item via principal component analysis, and then compute the extremities of the segmentation mask. Starting from one extremity, we take the point which is a distance of a pre-defined constant `BITE_LENGTH`

pixels away from the extremity point, along the direction of the main axis, and cut the food item orthogonal to the main axis at this point.

B.4 Task Planner Details

To better understand the task planner $\mathcal{T}$, we provide a visualization of the decision tree for different foods in Fig. B.3.

B.5 Details on User Preferences

In this work, we consider user preferences that describe a desired *ordering of bites*, and their *combination with sauces*. While there are many other potential preferences to consider during feeding, such as combinations of multiple available bites, or desired bite sizes, our skill library is only equipped to handle one food item at a time (apart from dipping), without customizing for bite size. Thus, when soliciting preferences from user study participants, we use the following form template to provide context on preferences in scope with FLAIR:

> In this study, envision yourself receiving assistance from a robot caregiver tasked with feeding you. The robot is programmed to pick up and feed you one bite at a time from a plate filled with a variety of food items. The robot can only pick up one type of food item per bite. At the beginning of your meal, the robot asks for your input on the order in which you would like these items to be fed to you.

You might have specific preferences for the sequence in which the items are presented, such as starting with a particular item and ending with another, or alternating between items to enjoy a variety of flavors. These preferences can also vary in intensity. For instance, with a plate containing asparagus and mushrooms, you might express:

- No particular preference: "I have no specific order preference for asparagus or mushrooms."
- No particular preference: "I'm fine with eating them in a random order, no specific pattern."
- Weak preference: "I sometimes begin with asparagus, but I'm not too strict about the order afterward."
- Weak preference: "I'd slightly prefer to eat all the asparagus first, but it's not crucial."
- Strong preference: "I always want to alternate between bites of asparagus and mushroom."
- Strong preference: "Please don't feed me any mushrooms."

Throughout this study, we will present you with eight different plates, each containing a unique assortment of food items. For each plate, we ask you to specify your preferred order for consuming these items. Please keep the following things in mind while responding -

1. Please write your preference in natural language, as if you are instructing your caregiver. If you don't have a preference, you can write "No preference."
2. Our robot caregiver is designed to pick up only one type of food item at a time for each bite. This means that for a plate with, say, asparagus and mushrooms, the robot can serve you bites of either asparagus or mushroom separately, but it's not able to combine them into a single bite.
3. In this study, we're keen to learn about your preferences in terms of the order in which you'd like to be served different food items, and if applicable, your choices for dipping sauces for certain dishes. However, we kindly ask that you do not specify preferences related to the size of the bites or requests for cutting specific food items.

Your responses will help us understand the various ordering preferences people might have in a setting where they are fed by a robotic caregiver.

As a representative sample of the kinds of responses received, we provide all preferences received for the spaghetti and meatballs plate below. Notably, we see that both shared and diverse preferences emerge, and very few preferences are out of scope (highlighted in red) for FLAIR.

- *"No particular preference"*
- *"Cut up the meatballs and eat it with a bite of the pasta."*
- *"Slight preference to twirl spaghetti around and to eat meatballs individually"*
- *"Would prefer to eat spaghetti only, without any meatballs."*

- *"I would alternate between one bite of spaghetti and a meatball"*
- *"Id probably prefer eating a bite of spaghetti first before going in for the meat ball, but not a strong preference."*
- *"Please feed me only the spaghetti, and skip all the metballs."*
- *"No preference"*
- *"Spaghetti then meatballs"*
- *"alternating between spaghetti and meatballs"*
- *"I have no preference as long as the three meatballs are not fed consecutively."*
- *"Eat fork of spaghetti then fork of meatball"*
- *"I would like to eat bites of spaghetti and meatballs in alternating order."*
- *"I always want to alternate between bites of meatball and pasta."*
- *"I'd slightly prefer to alternate a bit between the spaghetti and meatballs."*
- *"I prefer to eat meatballs first but it's not crucial."*
- *"I like to alternate bites: one bite spaghetti followed by one bite of meatball, and so on."*
- *"First, roll up spaghetti onto fork and eat. Part and then stab the meatball with the fork and eat. I would like to have at least one fork of spaghetti before and in between the meatballs."*
- *"I always alternate between bites of spaghetti and meatballs."*
- *"I always want to alternate between bites of meatballs and spaghetti."*
- *"I'd always like to alternate between spaghetti and meatballs, starting with spaghetti first."*
- *"Eat spagetti first, then any order would work"*

- *"Please don't feed me meatballs."*
- *"I would like to eat all of the spaghetti. Please don't feed me any meatballs."*
- *"I prefer eating more spaghetti at first followed by randomly switching between spaghetti and meatballs."*
- *"Any order is fine in the beginning as long as the last 2 bites are spaghetti"*
- *"Id prefer to alternate between bites of spaghetti and meatballs, but it is not crucial,"*
- *"I would get a few bites of spaghetti, then one meatball, then alternate so that I finish them at approximately the same time."*
- *"I would prefer to alternate between bites of spaghetti and bites of the meatballs."*
- *"Alternate between spaghetti and meatballs."*
- *"I sometimes prefer to alternate between spaghetti and meatballs"*
- *"Please feed me meatballs at regular intervals of time so that I consistently get bites of meatballs until they run out. Between bites of meatballs, feed me spaghetti. Multiple bites of spaghetti between bites of meatballs is fine. The goal is that I regularly get the meatballs and they do not run out by the end. This is not a strong requirement but it would be great if last bite is of the meatballs."*
- *"Twirl a fork for spaghetti, stab meatballs with fork"*
- *"I would alternate between cutting a meatball in half, skewering it, then eating it and twirling spaghetti onto my fork then eating it. I would repeat this process till the plate is empty."*
- *"I would prefer to eat a meat ball first."*
- *"I like to eat spaghetti first and then afterwards eat the meatballs."*
- *"I'd like to alternate between meatballs and spaghetti but it isn't crucial"*

- *"I would alternate between bites of spaghetti and bites of meatball"*
- *"I only eat the meatballs."*
- *"Alternating bites"*
- *"I have a weak preference for alternating spaghetti and meatballs, maybe a couple bites of spaghetti followed by a meatball."*
- *"I always want to alternate between bites of spaghetti and meatball."*
- *"Please don't feed me meatballs."*
- *"I'd prefer to first eat some spaghetti with meatballs in middle. Depending on how much I like the meatballs, I would decide if I want my last bite to be spaghetti or meatballs."*
- *"Please dont feed me any meatballs. I would take alternating bits of spaghetti and meatballs"*
- *"I'd like to have the spaghetti first and then a meatball, and then repeat the order"*
- *"Do not feed me meatballs"*
- *"No particular preference"*
- *"Starting with meatball, followed by alternation of meatballs and spaghetti, and with spaghetti"*

From these responses, we then use LLM summarization to find groups of common preferences (i.e. *"Alternate bites of spaghetti and meatballs"* or *"Don't feed me meatballs, only spaghetti"*) with which to evaluate.

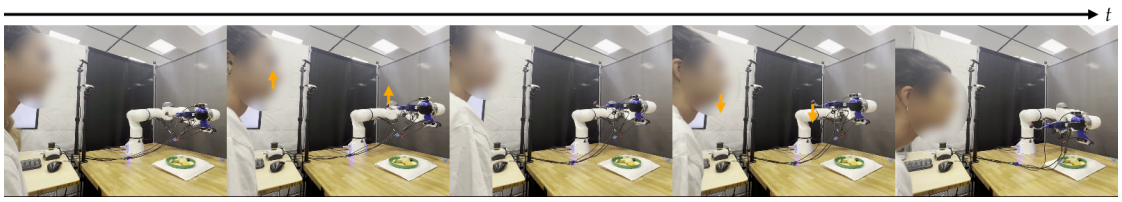


Figure B.4: **Bite Transfer via Visual Servoing:** FLAIR is compatible with any method of bite transfer, and we demonstrate this by combining acquisition with a visual-servoing based approach on the Franka Panda. We use an open-source facial landmark tracking library (`https://github.com/1adrianb/face-alignment`) to dynamically track a person's mouth position, and update the position of the fork-tip such that transfer is robust to small head movements (orange arrows).

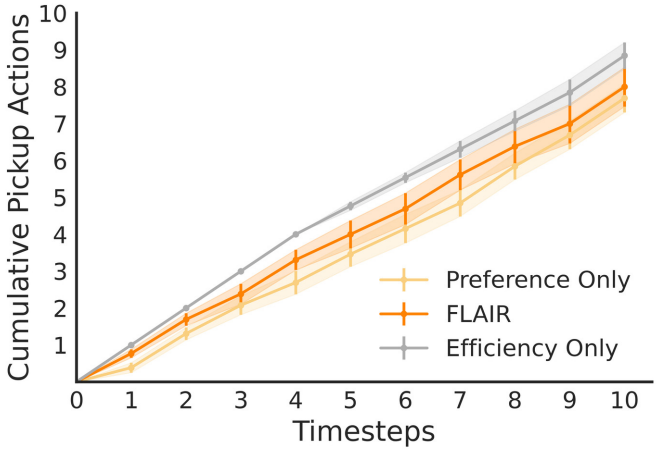


Figure B.5: **Efficiency Across All Plates:** Across all evaluated plates, including plates where both strong preferences or no preferences were provided, we find that FLAIR outperforms Preference/Commonsense Only in terms of amount of food picked up over time.

B.6 Additional Experimental Results

B.6.1 Efficiency Across Methods

We empirically find that the trend of FLAIR outperforming a Commonsense/Preference Only approach in terms of efficiency holds both in the presence and absence of strong user preferences. In Figure B.5, we visualize the

amount of food picked up over time across all methods, averaged across *all* trials including strong user preferences and none at all. We see that FLAIR yields more pickup actions over time compared to Preference Only. It is unsurprising that Efficiency Only performs the best, but this is expected as FLAIR must also balance actions that satisfy given preferences as well, when a user has strong preferences.

B.6.2 Bite Transfer

In addition to integrating FLAIR with both inside-mouth and outside-mouth bite transfer frameworks on the Kinovas, we additionally show that FLAIR is compatible with any bite transfer framework by pairing it with a visual servoing-based outside-mouth framework on the Franka. In Fig. B.4, we provide qualitative visualizations showing the ability of the system to react to head movements in real-time, and adjust the fork tip position accordingly to align with the mouth's center.

APPENDIX C

FEAST APPENDIX

C.1 Caregiving Requirements Identified from Formative Study

Refer to Table C.1.

Table C.1: Key personalization requirements for in-the-wild mealtime assistance (Part I: Requirements R1–R23). Identified from a formative study with 19 care recipients and 2 community researchers, along with a comparison of how FEAST and prior feeding systems (Nanavati et al. [198] and FLAIR [121]) address them.

ID	Requirement	Description	FEAST	Nanavati et al.	FLAIR	Explanation
	Mealtime Tasks					
R1	Feeding	Assist with eating solid food.	✓	✓	✓	All three systems can perform feeding tasks.
R2	Drinking	Assist with consuming liquids.	✓	✗	✗	Only FEAST supports drinking.
R3	Static Mouth Wiping	Assist with moving a wiper to near the user's mouth.	✓	✗	✗	Only FEAST supports static mouth wiping.
R4	Dynamic Mouth Wiping	Dynamically wipe a user's mouth who remains stationary.	✗	✗	✗	None of the systems support this.
R5	Manual Task Selection	Allows the user to choose which mealtime task (feeding, drinking, wiping) to perform.	✓	-	-	Nanavati et al. and FLAIR only consider feeding.

Table C.1 (continued)

ID	Requirement	Description	FEAST	Nanavati et al.	FLAIR	Explanation
R6	Auto-Execute Last Task	Repeat the previous mealtime action if re-selection does not occur until a certain timeout.	✓	-	-	Nanavati et al. and FLAIR only consider feeding.
R7	Auto-Detect Mouth Wiping Requirement	Perceive residue on user's mouth to trigger mouth wiping.	✗	✗	✗	None of the systems support this.
R8	Meal Heating	Keeps food warm or at a desirable temperature.	✗	✗	✗	None of the systems support this.
	Interaction: Human to Robot					
R9	Web Interface	Provides a screen-based control option accessible via a web browser.	✓	✓	✗	Both FEAST and Nanavati et al. can be operated via a web interface. FLAIR does not have one.
R10	Gesture-Based Control	Supports head gestures for interactions.	✓	✓	✓	FLAIR and Nanavati et al. use an open-mouth gesture to signal readiness for a bite. FEAST uses open-mouth or head-nod gestures by default.
R11	Personalized Gestures	Enables customization of gestures based on individual user capabilities and preferences, for example, mouth open with a smaller threshold, head shake, blinking, etc.	✓	✗	✗	FEAST allows adding custom gestures by providing a language description, a few positive examples, and a few negative examples, leveraging LLM-based code synthesis.
R12	Button-Based Control	Supports a button for interactions.	✓	✗	✗	Only FEAST supports a non e-stop button.

Table C.1 (continued)

ID	Requirement	Description	FEAST	Nanavati et al.	FLAIR	Explanation
R13	Personalized Button Clicks	Users request different button click patterns (e.g., three quick presses, one long press) to correspond to different aspects of the system.	✗	✗	✗	None of the systems support this, but the code synthesis capabilities of FEAST given positive and negative examples could be extended to support this.
R14	Eye Gaze	Supports eye gaze for interactions.	✗	✗	✗	None of the systems support this.
R15	Voice	Supports voice for interactions.	✓	✓	✗	Both FEAST and Nanavati et al. feature web interfaces which support speech-to-text / speech-based control.
	Interaction: Robot to Human					
R16	Web Interface	Allows the system to present information visually on a browser.	✓	✓	✗	FEAST and Nanavati et al. display information on a web interface; FLAIR does not.
R17	Voice Feedback	Speaks out the robot's status or actions.	✓	✓	✓	All systems can provide fixed spoken prompts. Moreover, FEAST can generate feedback adaptable to current system behavior.
R18	Audible Beeps	Offers simple sound-based signals if voice feedback is undesirable.	✓	✗	✗	FEAST can use beeps to signal robot readiness; the other two systems do not.
R19	LED Indicator	Provides a visual cue of robot state or readiness.	✓	✗	✗	FEAST features an LED near the camera mount that can be customized to indicate readiness.

Table C.1 (continued)

ID	Requirement	Description	FEAST	Nanavati et al.	FLAIR	Explanation
	Hardware					
R20	User-Preferable Tools	Enables users to choose eating tools that suit their preferences or physical needs, such as smaller tips for those with narrow mouth openings.	✓	✗	✓	FEAST supports switchable-tip silicone and metal utensils, a Velcro-attached replaceable mouth wiper, and an adaptable handle for user-provided mugs with fixed straws. Nanavati et al. use a fixed 3D-printed fork, while FLAIR also supports switchable-tip silicone or metal utensils.
R21	Autonomous Tool Change	Reduces the need for caregiver intervention when switching between tools for different tasks.	✓	✗	✗	FEAST can switch between tools (feeding utensil and mouth wiper) in its mount and also pick up a drink from a table.
R22	Detachable Tool Tips	Facilitates easier cleaning, maintenance, or replacement of utensils.	✓	✗	✓	FEAST and FLAIR tool tips can be removed easily for cleaning. Nanavati et al. use a 3D-printed metal utensil that must be intricately screwed onto the mount with three screws.
R23	Flexible Mounting	Allows system installation on various supports (e.g., a wheelchair or stand) for user convenience.	✓	✓	✓	FEAST mounts on a wheelchair or Vention stand, Nanavati et al. on a wheelchair or movable table, and FLAIR on a Vention stand or table.

Table C.2: Key personalization requirements for in-the-wild mealtime assistance (Part II: Requirements R24–R46).

ID	Requirement	Description	FEAST	Nanavati et al.	FLAIR	Explanation
	Acquisition					
R24	Bite Prediction	Automated selection of which food item to acquire next based on user preferences.	✓	✗	✓	FEAST and FLAIR use a bite-ordering preference (e.g., "feed all of X, then Y") for bite prediction. Nanavati et al. require manual bite selection.
R25	Auto-Continue Predicted Bite	Automatically continue with predicted next bite after a timeout.	✓	✗	✓	Nanavati et al. do not predict bites, so they cannot auto-continue them.
R26	Manual Bite Selection	Allows the user to specifically choose which piece of food to pick up.	✓	✓	✗	FLAIR does not feature any user interface for bite selection. Users can override FEAST's autonomously predicted bite via the web interface.
R27	Diverse Fork-Based Acquisition Skills	Supports various acquisition (skewering, scooping, twirling, dipping, etc.) and pre-acquisition actions (pushing, grouping, cutting, etc.).	✓	✗	✓	Both FLAIR and FEAST can skewer, scoop, dip, and perform limited cutting of soft foods. Nanavati et al. only support skewering.
R28	Finger Food Acquisition Skills	Supports picking up finger foods.	✗	✗	✗	None of the systems support this.
R29	Bite Size Customization	Enables adjustments to the portion size for each bite.	✓	✗	✗	FEAST allows commands like "Scoop more" or "Dip more" to change bite size for scooping or dipping. For skewering, it autonomously selects smaller food pieces. Caregivers must provide bite-sized portions for hard items.

Table C.2 (continued)

ID	Requirement	Description	FEAST	Nanavati et al.	FLAIR	Explanation
R30	Skewering Axis Selection	Allows adjusting fork orientation to trade off between easy food acquisition and transfer through narrow mouth openings.	✓	✗	✗	FEAST enables users to change the skewering axis via adaptability commands.
R31	Ask Confirmation After Acquisition	Allows users to confirm moving to transfer after acquisition.	✓	✓	✗	FEAST lets users enable or disable confirmation prompts. Nanavati et al. always ask for confirmation with a fixed auto-continue time.
	Transfer					
R32	Adjust Outside-Mouth Distance	Lets users define how close the tool should come to their mouth for outside-mouth transfers.	✓	✓	✗	Users can specify this distance in FEAST via adaptability commands. Nanavati et al. adjust it in a settings page.
R33	Adjust Inside-Mouth Placement	Allows customizing how far the utensil enters the mouth for comfort.	✓	✗	✓	Both FEAST and FLAIR let users calibrate the inside-mouth position. Nanavati et al. do not do inside-mouth transfer.
R34	Convey Robot Ready to Initiate Transfer	Notify user when robot is ready to initiate transfer	✓	✓	✓	All systems support this, but FEAST offers more interchangeable interfaces (e.g., voice, LED).
R35	Detect User Ready for Transfer	Perceive if the user is ready to initiate transfer	✓	✓	✓	All systems support this, but FEAST offers more interchangeable interfaces (e.g., button, gestures).
R36	Convey User can Take Bite / Sip / Mouth Wipe	Notify user that they can complete transfer	✓	✓	✓	All systems support this, but FEAST offers more interchangeable interfaces (e.g., voice, LED).

Table C.2 (continued)

ID	Requirement	Description	FEAST	Nanavati et al.	FLAIR	Explanation
R37	Detect Transfer Complete	Perceive that the user has completed transfer	✓	✗	✓	All systems support this, but FEAST offers more interchangeable interfaces (e.g., force-torque sensing for detecting bites, button, gestures).
	Miscellaneous					
R38	Speed Control	Allows adjusting the robot's movement speed for safety or efficiency.	✓	✓	✗	FEAST uses language commands to update speed globally or for specific skills. Nanavati et al. can change bite transfer speed in settings.
R39	Non-Obtrusive Movement	Minimizes intrusion into the user's personal space or field of view.	✓	✗	✗	FEAST's custom tools can be held from the side, like humans do, reducing occlusion compared to Nanavati et al. and FLAIR.
R40	Socially Acceptable Movement	Ensures the robot's motions appear natural and comfortable in social settings.	✗	✗	✗	None of the systems explicitly address social acceptability of movement.
R41	Retract From Certain Configurations	Allows the user to retract the robot from certain poses for safety or convenience.	✓	✗	✗	FEAST lets users add a retract node via adaptability commands at specific arm configurations.
R42	Robot Voice Prompts	Enables custom voice prompts for guidance or reminders.	✓	✗	✗	FEAST allows adding custom voice prompts at any node through adaptability commands.
R43	Long-Term Preference Learning	Learn user preferences over time to improve assistance.	✗	✗	✗	None of the systems support this.
R44	Adjust Fixed Configurations	Allows editing fixed configurations (e.g. above plate config).	✗	✓	✗	Nanavati et al. let users adjust fixed configurations; others require experimenters to do so.

Table C.2 (continued)

ID	Requirement	Description	FEAST	Nanavati et al.	FLAIR	Explanation
R45	Adapt to User-Robot-Plate Setup	Ensures the system can function with different user seating and plate placements.	✓	✓	✓	All systems adapt to variations in user-robot-plate arrangement leveraging perception.
R46	Scene Selection	Enables recalling and switching between saved settings for different scenes.	✗	✓	✗	Nanavati et al. let users switch between pre-configured scenes. FLAIR does not consider different scenes. Users can adapt FEAST to new scenes, but need experimenter help to load settings from prior scenes.

C.2 Gesture Synthesis using Large Language Models

We now describe our custom gesture synthesizer in more detail. The synthesizer takes a natural language description of the gesture (e.g., "shaking my head from left to right"), 3-5 positive examples, and 3-5 negative examples supplied by the user. Each of these training examples comprises a time series of head poses and facial key points, which are obtained from our head perception. The user controls data collection through the interface. In practice, the examples are sampled at a rate of roughly 10Hz and typically last for 5-10 seconds.

Given these inputs, we use an LLM (GPT-4o [107]) to generate a Python function capable of detecting specific gestures. The Python function takes input from the head perception model, external events that may shut down the detector, and a timeout counter; it outputs if the gesture has been detected. We prompt

the LLM with two in-context examples: one for detecting "mouth open", and the other for detecting "head nod". We also provide a description of the head perception module that is available to the Python function. Each Python function proposed by the LLM is validated with respect to the positive and negative examples. If evaluating any of the examples raises an exception, or if the Python classification misclassifies the example, we automatically re-prompt the LLM up to 3 times with failure information before returning the best-seen program.

As an additional extension, we allow the LLM to propose real-valued hyperparameters for the Python function detector. For example, in the case of "shaking my head from left to right", the LLM automatically proposes a hyperparameter for head yaw angle from left to right, $head_shake_threshold$. Following the in-context examples, each hyperparameter is proposed along with a lower and upper bound. We then optimize these hyperparameters for each proposed program using a grid search with 10 values per dimension. The objective for optimization is classification accuracy with respect to the user-provided positive and negative examples.

Once synthesized, the gesture can be tested by the user through the interface and added to behavior trees through the natural-language-based personalization procedure described in Section 6.3.2. When the generated function is invoked, it is dynamically executed within a separate memory-controlled-workspace for safety.

To evaluate FEAST's gesture personalization capabilities against a baseline that doesn't use personalized parameters, we conduct an ablation study. We compile 8 gesture types from the formative study:

- Head nod (up-down)
- Head shake (left-right)
- Head tilt (left-right)
- Head still (5 seconds)
- Mouth open
- Mouth continuously open (5 seconds)
- Three successive eye blinks
- Three successive eyebrow raises

Three individuals without mobility limitations emulate each gesture over a prescribed range of motion while secured to a wheelchair with restraint bands—a technique OTs routinely use to simulate mobility limitations. These emulations are post-validated by an occupational therapist referencing Zoom-recorded snippets from our formative study. Following the FEAST setup, we collect 5 positive and 5 negative examples per individual and gesture, along with corresponding language descriptions, to synthesize gesture detectors with personalized parameters. The personalized gesture detector achieves an F1-score of 0.9 ± 0.02 on the examples, compared to 0.63 ± 0.09 for no personalized parameters.

C.3 Adherence to ISO 13482:2014

Transferring food, drinks inside the mouth and wiping requires nuanced physical interactions. These physical interactions, if completed in an unsafe manner, may lead to hazards during human-robot collaboration. The ISO 13482:2014,

Robots and robotic devices — Safety requirements for personal care robots, is a guideline developed by the International Organization for Standardization to identify possible hazards and address them. We show that our mealtime-assistance system adheres to 13482:2014.

Our system uses a Kinova Gen3 7DoF which, per its user guide, adheres to ISO/TS 15066 Collaborative robots certification, ISO 12100 Safety of machinery, ISO 13849 Safety-related parts of control systems, ISO 10218-1 Industrial robots, IEC 60204-1 Safety of machinery – Electrical, and IEC 60950-1: Information Technology Equipment – Safety. Some of these are also requirements in 13482:2014.

Below is a summary of all the robot's safety features. We will reference them when explaining how we satisfy the requirements listed in 13482:2014. We will refer to *pose filtering* as **A**, *compliant control* as **B**, *active collision monitoring* as **C**, *safety-centric hardware design* as **D**, *emergency stops* as **E**, and *watchdog* as **F**. These are described in detail in Section 6.4.3.

C.3.1 ISO Guidelines

4 Risk assessment:

4.1 General: The system adheres to ISO 12100 as per the Kinova Gen3 7DoF user guide.

4.2 Hazard identification & 4.3 Risk estimation: (a) uncertainty of autonomous decisions made by the robot and possible hazards from wrong decisions; Autonomous decisions made by the robot are the movements to pick up food, acquisition, and path the robot takes to feed an operator. **A** reduces hazards a controlled zone. **C** monitors for collisions with this zone, further reducing risk

of danger to operator. **(b)** different levels of knowledge, experience and physical conditions of users and other exposed persons; Will provide transparency so that users know the details about how the robot works as well as all safety features at the beginning of each study. May have varying physical conditions that our robot can adjust to. **(c)** normal but unexpected movement of the personal care robot; We will demonstrate robot's movements to user pre-experiment to minimize this. **(d)** unexpected movement (e.g. jumping in front of the personal care robot from the side or from a higher level) of humans, domestic animals and other safety-related objects; During velocity control, we have **C** to stop the robot in the case of unexpected movements. During **B**, users can push the robot with minimal force to move it away. **(e)** unintended movement of the personal care robot; Unintended movements of the robot would be in the pre-defined zone by **A** and robot system has **C** and **B**. **(f)** unexpected travel surfaces and environmental conditions in the case of mobile robots; The system has two options for mounting. It can be mounted to a wheelchair system or to a Vention stand. For Vention stands, operator dictates movement by hand. For wheelchair, as products in the market, they must satisfy the ISO 7176 which regulates wheelchair safety. **(g)** uncertainty of safety-related objects to be handled in the case of mobile servant robots; **C** & **B** **(h)** conformity to the human anatomy and its variability in the case of physical assistant robots and person carrier robots; System detects head poses and dimensions, creating a custom model that fits the user that is currently being fed.

5 Safety requirements and protective measures:

5.1 General: **(a)** inherently safe design; **(b)** protective measures; **(c)** information for use; are explained in **(5.2.2)**, **(5.2.3)**, **(5.2.4)**.

5.2 Hazards related to charging battery: (5.2.1) The Kinova Gen3 7DoF robotic arm operates on an external power supply not a built-in battery charging system. Thus we do not need to satisfy requirements mentioned in **5.2.1**. **(5.2.2)** As per the Kinova Gen3 7DoF user guide, the system satisfies IEC 60204-1. All live parts are covered in the system. **(5.2.3) (a)** charging systems shall be designed in such a way that the charging connections are only activated when the personal care robot is connected to them; Battery and charging are all external and not in the robotic arm. **(b)** batteries show status of charge, and that charging connections are only activated if the personal care robot is connected. **(5.2.4)** Provided in the Kinova Gen3 7DoF user guide and in the user guide for the operator's wheelchair if arm is connected to one.

5.3 Hazards due to energy storage and supply: All parts of **5.3** are satisfied as (a) the system uses an external battery, (b) system does not have pneumatic equipment or hydraulic equipment, (c) the system follows IEC 60204-1, IEC 60950-1, ISO 10218-1, and ISO 12100. The aforementioned IECs and ISOs are extensions or incorporate all the IEC and ISO requirements mentioned in **5.3**.

5.4 Robot start-up and restart of regular operation: (5.4.1) System has two start-ups. One is hardware start-up and the other is the feeding algorithm start-up. On both start-ups, the robot stays in a neutral position away from any operator. **(5.4.2) (a)** During start-up the personal care robot shall perform internal control tests to ensure that all safety-related functions are available. Failure to do so shall avoid any hazardous operation: Kinova arm does internal control tests and won't move if anything fails. Our feeding system also has the **F** to check other safety features and on start-up a script also checks to ensure **F** is

running. **(b)** If the personal care robot's safety-related functions cannot be performed correctly after start up, then it shall immediately perform a protective stop: If the script detects that **F** is not running, robot doesn't start up. If **F** finds any other safety features are not working, stops the robot and enters a safe state. **(c)** The personal care robot shall start up in a state of restricted speed, force, etc. (see 6.4 and 6.7) and shall only return to normal levels of control by means of a mode change (see 6.11): System follows 6.4, 6.7, and 6.11. **(d)** The personal care robot shall always start up in manual mode, and shall only continue operation in autonomous mode by means of a mode change as specified in 6.11.1: System follows 6.11.1. **(5.4.3) (a)** Manipulators, mobile platforms and other moving parts shall be de-activated by safety-related functions at start-up (to prevent any unintended actions). Application functions shall only be enabled if it can be positively established, via sensors, that no hazardous situation exists. This measure shall be applied if the robot is intended to enter autonomous mode immediately after start-up: Our system does (a). **(b)** The personal care robot shall always start up in a monitored standstill and shall only return to normal operation by means of a user action: System also satisfies (b).

5.6 Hazards due to robot shape: (5.6.1 & 5.6.2) Robot system satisfies ISO 12100, ISO 13854, and ISO 15534 as per the user guide. **(5.6.3) (a)** cushioning on sharp edges and points shall be provided to eliminate shearing, stabbing, cutting hazards and reduce impact hazards (see 4.3, Note 2): **D** satisfies this. **(b)** use of fixed or movable guards to cover hazardous moving parts: Done as per Kinova user guide. Added Dynamixel motors, but these are compliant and can be easily forced around. **(c)** adjustment of the robot speed and behaviour, if hazardous loads (e.g. sharp or pointed objects) are being carried: **F** monitors speed, force,

etc., keeping them at safe levels, at all times. **(5.6.4)** system adheres to ISO 12100 and ISO 7010.

5.7 Hazards due to emission & 5.8 Hazards due to electromagnetic interference: Kinova satisfies requirements in **5.7** & **5.8** and our added features do not release new emissions or add new features that would be impacted due to electromagnetic interference.

5.9 Hazards due to stress, posture and usage: **5.9.2** addresses physical while **5.9.3** addresses mental stress. **(5.9.2.1) (a)** A personal care robot shall be designed to minimize or reduce physical stress or strain to its user due to continuous use, including but not limited to uncomfortable posture, operational environments that would directly cause physical discomfort, e.g. fatigue and tendon inflammation: We allow users to specify inside mouth or outside mouth bite transfer, with the algorithm being tuned to their preferences, to maximize comfort. **(b)** The design of the personal care robot shall take into account typical body sizes of the intended user population in order to avoid physically demanding body postures or to ensure easy operation. ISO 14738 describes how principles of ergonomic factors, by construction of workstations and machinery, should be applied. This should be considered when building a personal care robot, where somebody sits on or stands in front of the robot: System design takes ISO 14738 into account and has **A** to account for human variability in comfort. **(5.9.2.2) (a)** the design and location of manual control devices, which should ensure that they can be operated without physical stress or discomfort; Emergency stop buttons are close to user and take minimal force to press. **(b)** the proper ergonomic design and location of the seat, which should ensure that good posture can be maintained during operation of the personal care robot: We attach to user's

wheelchair, so it should already be adjusted to user's comfort. **(c)** command devices that are detachable or hand-held instead of being permanently attached to the personal care robot in an inappropriate position: **E** is detachable and one is velcroed within immediate reach of the user while the other is always close to the experimenter so that both can reach in an emergency. User's web interface's smart device is attached to a stand that will connect to the wheelchair. **(5.9.3.1) (a)** a personal care robot shall be designed to minimize or reduce mental stress to its user due to continuous use; Robot is controlled via a user web-interface where buttons are clicked to control the robot's next move. The user's web-interface was designed so that its interactive buttons are large and easy to click. **(b)** user interfaces such as controls, signalling or data display elements, shall be designed to be easily understood so that clear and unambiguous interaction between the human and the personal care robot is possible: Our user interface's buttons are clearly labeled in large font. **(c)** the personal care robot shall conform to ergonomics standards appropriate to its intended purpose (see ISO/TR 9241-100, ISO 9241-210, ISO 9241-400, ISO 9241-920 and ISO 11228): Conforms to the above ISOs. **(5.9.3.2)** Design of the user interface was created with users in loop, so users know what to expect in terms of signals, display design, etc.

5.10 Hazards due to robot motion: (5.10.2.1, 5.10.2.2, & 5.10.2.3) Kinova is mechanically stable and follows the inherently safe design points in **5.10.2.2**. Have **F** which satisfies the protective measures mentioned in **5.10.2.3**. **(5.10.3)** The user's wheelchair should adhere to the requirements about instability during travel discussed in **5.10.3**. **(5.10.4)** There is no instability while carrying the fork and food in velocity control mode. In **B**, we appropriately change the dynamic properties of the robot arm, so that there are no hazards while carrying the utensils and food. **(5.10.5.1) (a)** Risk assessment shall determine the allowable maximum

values of the appropriate parameters (e.g. contact force) that have an influence on the risk caused by contact over the entire operational range: covered in ISO 12100 and ISO 10218-1. **(b)** A personal care robot shall be designed to ensure that it does not cause any hazardous rollovers, runaways, or detachment of robot body parts, even when it receives any collision forces or safety-related obstacle detection signals during its motion, up to the limits specified for its intended operation: covered in ISO 12100 and ISO 10218-1. **(5.10.5.2 & 5.10.5.3)** Implementations to Kinova arm that satisfy requirements in **5.10.5.2 & 5.10.5.3** can be found in Kinova Gen3 7DoF user guide. Furthermore, we have **C** that stops robot when anomaly is detected. **(5.10.6)** Irrelevant as robot is not a restraint type physical assistant robot. **(5.10.7)** Irrelevant as robot is not a person carrier robot. **(5.10.8 & 5.10.9)** The system has **F** that monitors and controls velocity/force and **C** that watches for obstacles in surrounding area to minimize collisions. All requirements listed in **5.10.8 & 5.10.9** are satisfied by these safety features. Furthermore, the only safety-related obstacle is the user in our system environment and robot is only near user when doing bite, drink transfer or mouth wiping which is when the robot is in **B**. **(5.10.9)** Information to satisfy requirements in **5.10.9** can be found in Kinova user guide.

5.11 Hazards due to insufficient durability: Information to satisfy requirements in **5.11** can be found in Kinova user guide.

5.12 Hazards due to incorrect autonomous decisions and actions: (5.12.1) System can correctly detect what is food and what is not food. Has powerful force sensors, velocity sensors, etc. that are closely monitored to keep hazards during autonomous movements to a minimum. **(5.12.2) (a)** constraining the operational scenarios to reduce risk of harm due to incorrect actions: **A** satisfies (a). **(b)** use of

unique identifiers for safety-related objects, travel paths, etc: **A** also satisfies (b). **(5.12.3)** The system uses multiple checks to make sure an autonomous decision is correct and allows user input to stop any process if something goes around.

5.13 Hazards due to moving components: Requirements here are satisfied by **A**, **C**, and **E**. Furthermore, **D** makes sure that even in the instance of a collision, little harm is done to a person.

5.14 Hazards due to lack of awareness of robots by humans: Points listed in **5.13** and detector for when a user may not be paying attention to the robot when some form of transfer is occurring satisfying requirements to **5.14**.

5.15 Hazardous environmental conditions & 5.16 Hazards due to localization and navigation errors: Satisfied by the points listed in **5.13**.

6 Safety-related control system requirements:

6.1 General: Requires **(a)** emergency stop: Kinova arm provides an emergency stop (experimenter only), **(b)** protective stops: **F** & **E** (both user and experimenter), **(c)** limits to workspace: **A**, **(d)** safety-related speed control: **F**, **(e)** safety-related force control: **F**, **(f)** hazardous collision avoidance: **C** & **B**, and **(g)** stability control (including overload protection): description for these can be found in Kinova user guide & **D**. **6.1.2** Irrelevant as robot is not restraint type or person carrier type. **6.1.3** System is a physical assistant robot of type 2.4, with safety features listed above satisfying the type requirements **(d)** in **6.2.2.2**, **(c)** in **6.2.2.3**, **(d)** in **6.3**, **(d)** in **6.4**, **(b)** in **6.7**, and **(d)** in **6.5.2.1, 6.5.2.2, 6.6, & 6.7**.

C.4 In-the-wild Evaluation Questions

C.4.1 Pre-Study Questionnaire

- What is your age?
- What is your gender?
- What is your race?
- What is your medical condition?
- How long have you had this medical condition?
- What are your daily challenges with getting fed by your caregiver?
- Do you have any familiarity with robotic/assistive technologies? If yes, please share your experience.

C.4.2 Pre/Post Study Caregiving Comparison (1–5 scale, very low to very high)

- **Effort:** I have to put in significant effort to communicate my preferences during meals when assisted by my caregiver/the robot.
- **Control:** I feel in control of my feeding experience when assisted by my caregiver/the robot.
- **Independence:** I feel a sense of independence when I receive assistance from my caregiver/the robot.

C.4.3 NASA TLX (1–7 scale, very low to very high)

- **Mental Demand:** How mentally demanding was using the meal-assistance system?

- **Physical Demand:** How physically demanding was using the meal-assistance system?
- **Temporal Demand:** How much time pressure did you feel when using the meal-assistance system?
- **Performance:** How successful were you in using the meal-assistance system to achieve your goals? (We flip this while reporting scores.)
- **Effort:** How hard did you have to work to use the meal-assistance system effectively?
- **Frustration:** How frustrated were you while using the meal-assistance system?

C.4.4 Technology Acceptance Model (1–5 scale, Strongly Disagree–Strongly Agree)

- **Perceived Usefulness:** Using this meal-assistance system will make me more independent in eating.
- **Perceived Ease of Use:** This meal-assistance system is easy to use.
- **Attitude Toward Using:** Using the meal-assistance system for improving my independence is a good idea.
- **Behavioral Intention to Use:** Assuming I have access to this meal-assistance system, I predict that I would use it in my daily life.
- **Perceived Enjoyment:** I find using this meal-assistance system to be enjoyable.
- **Adaptability:** I find this meal-assistance system adaptable to my requirements.

- **Transparency:** I find this meal-assistance system to be transparent.
- **Safety:** I find this meal-assistance system to be safe.

C.5 Personalization Requests During in-the-wild Evaluation

C.5.1 Meal ID 1: CR1 in Personal Context

Meal: Buffalo Chicken Bites and Potato Wedges

Bite Ordering Preference: Buffalo chicken bite dipped in ranch dipping sauce then potato wedge without dipping

Adaptability Request: Feed me As fast as you can
Adaptability Response: The robot's speed for handling drinks and wipes has been successfully increased to make the feeding process faster. However, attempts to speed up the process for using utensils were not successful.

Transparency Request: How can I speed up Feeding me with a utensil
Transparency Response: To speed up the process of feeding you with a utensil, you can adjust the "Speed" parameter for the relevant behaviors. Specifically, you can change the speed to "high" for the following actions:

- **PickUtensil**: This will make the robot pick up the utensil faster.
- **AcquireBite**: This will speed up the process of picking up the food item.
- **TransferUtensil**: This will make the robot bring the utensil to your mouth more quickly.
- **StowUtensil**: This will speed up placing the utensil back in the tool mount.

You can adjust these settings through the user interface where these parameters are available.

Adaptability Request: Increase speed to high when feeding with a utensil
Adaptability Response: The feeding system was updated to increase the speed to high when using a utensil to acquire a bite.

Adaptability Request: Skip confirmation screens
Adaptability Response: The robot-assisted feeding system was updated to skip confirmation screens when acquiring a bite with a utensil, transferring a drink, and transferring a wipe. However, skipping the confirmation screen for transferring a utensil was not successful due to an invalid parameter.

Adaptability Request: Bring everything as close to my mouth as possible
Adaptability Response: The robot feeding system was adjusted to bring utensils, drinks, and wipes closer to the user's mouth, with a new distance of 0.07 units.

Adaptability Request: Use button when completing a transfer when taking a sip
Adaptability Response: The robot-assisted feeding system has been updated to use a button to complete actions when transferring drinks, utensils, and wipes.

Transparency Request: What is the default action to complete a transfer
Transparency Response: The default action to complete a transfer is to use a button interaction. This means the robot waits for a button press to confirm that the transfer is complete.

Transparency Request: What other ways can I end a transfer besides pushing the button

Transparency Response: Besides pushing the button, you can end a transfer by using the "sense" interaction, where the robot detects completion automatically, or by setting it to "auto_timeout," where the transfer ends after a set time.

Adaptability Request: Use the sense interaction when completing a transfer Bite

Adaptability Response: The robot-assisted feeding system has been updated to use sensory feedback when completing the transfer of utensils, drinks, and wipes.

Adaptability Request: Use the button to complete a transfer only when taking a sip

Adaptability Response: The system was updated so that a button is now used to complete the transfer when taking a sip, using utensils, or wiping.

Adaptability Request: Use sensory to complete bite transactions but not for sips or face wiping

Adaptability Response: The robot feeding system was updated to automatically complete actions for drinking and face wiping, while using sensors to ensure successful bites when using utensils.

Adaptability Request: Use sensors to end transaction with utensil use the button to end transactions when taking a sip use the button to end transaction for face wipe

Adaptability Response: The robot-assisted feeding system has been updated to improve how it completes tasks:

1. When using a utensil, the system now automatically senses when the task is done.
2. When taking a sip or wiping the face, the user needs to press a button to indicate the task is complete.

Adaptability Request: Initiate transfers with open mouth gesture

Adaptability Response: The robot-assisted feeding system has been updated to start transferring utensils, drinks, and wipes when it detects an open mouth gesture.

Explanation: Needed to rephrase and send speed requests to accelerate all tools.

Pseudo-Intervention: Had to reset because "close mouth" was not enabled (experimenter oversight).

Intervention (Web Interface): Web app failed to send information correctly, resulting in no predicted bites.

Explanation: Navigating the "Select Bite" page to override user commands.

Intervention (Skill): Lowered safety limits for outside-mouth transfers from 7cm to 6cm (necessary for CR1).

Pseudo-Intervention: Food moved away while CR1 was taking a bite. Stopped the robot but CR1 will attempt manual skill selection.

Explanation: Acquisition can fail if the selected keypoint's depth is invalid. Will need to retry.

Intervention (Hardware): Drink acquisition failed—robot moved to a seemingly random position. Had to reposition the drink.

Intervention (Hardware): Drink acquisition failed again—missed the drink during pickup. Repositioned the drink again.

Explanation: Can switch to button control for drink transfer if CR1 prefers.

Explanation: Dipping can fail due to unreliable depth detection for liquids. Will need to retry.

Explanation: How to use adaptability through prompt engineering for button/sensor-based transfers.

Intervention (Hardware): Drink acquisition failed again—robot moved to a random position. Need to check the AR tag and detection.

Intervention (Web Interface): Webpage failed to receive plate image. Page didn't populate and couldn't recover through manual pickup. Reset and refreshed the page.

Explanation: Difference in wording between "initiate transfer" and "end transfer" in adaptability settings.

Explanation: Prompt engineering for alternative bite selection.

Intervention (Hardware): Drink acquisition failed again—robot offset to the left and bumped the handle. Reset and retried.

C.5.2 Meal ID 2: CR1 Watching TV

Meal: Strawberries and whipped cream

Bite Ordering Preference: Please feed me strawberries dipped and whipped cream

Adaptability Request: Dip the strawberry deeper into the whipped cream

Adaptability Response: The system was updated to dip the strawberry deeper into the whipped cream by adjusting the dipping depth to 0.03.

Adaptability Request: Be silent

Adaptability Response: The robot-assisted feeding system has been updated to operate silently when transferring drinks, utensils, and wipes.

Adaptability Request: Change automatic timer to every five seconds

Adaptability Response: The robot-assisted feeding system was updated to automatically proceed every five seconds when acquiring a bite with a utensil and transferring a utensil or drink. However, the change could not be applied to the wiping action.

Intervention (Hardware): Strawberry got stuck in whipped cream. Simultaneously, camera anomaly due to CR1 looking away.
Explanation: How to get around webpage population error.
Intervention (Web Interface): Interface died when trying to finish the meal. Needed to refresh.

C.5.3 Meal ID 3: CR2 in Personal Context

Meal: Chicken nuggets ketchup and apple slices

Bite Ordering Preference: Dip chicken nuggets and ketchup and give apple slices after two bites of chicken nuggets

Adaptability Request: Don't show continue after picking up apples or chicken nuggets on plate or drink

Adaptability Response: The system was requested to stop showing a "continue" prompt after picking up apples, chicken nuggets, or a drink. However, the changes could not be made because the requested settings were not allowed.

Adaptability Request: Top show continue confirmation pages on the web interface

Adaptability Response: The robot-assisted feeding system has been updated to ask for confirmation before performing actions like acquiring a bite with a utensil, transferring a drink, and using a wipe.

Adaptability Request: Don't show continue pages on the web interface

Adaptability Response: The robot-assisted feeding system has been updated to no longer ask for confirmation when acquiring a bite with a utensil, transferring a drink, or using a wipe.

Adaptability Request: Dip chicken and ketchup more

Adaptability Response: The robot's feeding system was adjusted to dip the chicken deeper into the ketchup, enhancing the dipping action.

Adaptability Request: Go faster in between picking up bites of apple

Adaptability Response: The robot's speed has been increased when picking up and transferring bites of apple.

Adaptability Request: Do not dip apple and ketchup

Adaptability Response: The system was updated to ensure that when using a utensil to serve food, apples are only lightly dipped in sauces like ketchup, with minimal depth.

Explanation: Can't go back to bite pickup after dropped bite, what bite ordering should be, and how it can be personalized.

Explanation: Difference between continue on web interface and otherwise. CR2 is trying to remove confirmation pages.

Explanation: Gesture programming.

Explanation: Why it's dipping apple slices in ketchup, CR2 did not notice that he is selecting ketchup by mistake manually.

Intervention (Hardware): Bite stuck in ketchup, had to remove it.

Explanation: How to dip more ketchup.

Intervention (Web Interface): Webpage not sending message in adaptability, had to refresh web interface.

Explanation: Dipping and depth not functioning all the time.

C.5.4 Meal ID 4: CR2 in Social Context

Add Gesture:

Name: Continuous mouth open

Description: Mouth open for five seconds

Accuracy: 83.33%

Adaptability Request: Show confirmation page before transfer

Adaptability Response: Confirmation pages have been successfully added before transferring drinks and wipes. However, there was an issue adding a confirmation page for utensils.

Adaptability Request: Turn off voice

Adaptability Response: The robot's voice has been turned off for all interactions involving drinks, utensils, and wipes.

Adaptability Request: Use continuous mouth to detect I'm ready for a drink
Adaptability Response: A user requested a change to the robot's feeding system to recognize when they are ready for a drink using continuous mouth detection. However, there was an issue with the system recognizing the specified command.

Adaptability Request: Use continuous mouth open to detect for transfers
Adaptability Response: The robot-assisted feeding system has been updated to use continuous mouth opening as a signal for transferring drinks, utensils, and wipes.

Explanation: Be silent vs lower voice.
Explanation: Use continuous mouth open to show ready for a drink.
Pseudo-Intervention: Had to code back continuous transfer in (experimenter fault).
Intervention (Skill): Head perception returned none bug.
Not an Intervention, but: Went and refilled the mug with water upon CR2 request.
Intervention (Hardware): Robot fell while switching controllers after drink transfer, possibly due to a robot minimum voltage error.

C.5.5 Meal ID 5: CR2 Watching TV

Add Gesture:
Name: Head still
Description: Hold head still for five seconds
Accuracy: 83.33%

Adaptability Request: Head still for transfer completion of mouth wiping but do not make this change for drinking and eating

Adaptability Response: The robot-assisted feeding system was updated to keep the user's head still when wiping their mouth. However, the system will not keep the head still when the user is eating or drinking.

Adaptability Request: Be quiet and do not talk at all

Adaptability Response: The robot-assisted feeding system has been updated to operate silently when transferring drinks, utensils, and wipes.

Adaptability Request: Bring back confirmation page

Adaptability Response: The system now includes a confirmation step before using utensils to acquire food, transferring drinks, and using wipes.

Intervention: CR2 requested that his food be warmed up. Intervention to heat the food in the microwave.

Explanation: Auto bites got reset because of ghost clicks.

Intervention (Web Interface): Had to refresh webpage to send transfer continue for drink.

Intervention (Web Interface): Nose pointer interface CR2 uses stopped working.

Intervention (Web Interface): Web interface refresh for pickup bite.

C.5.6 Meal ID 6: CR1 in Social Context

Meal: Chicken and Hash Browns

Bite Ordering Preference: Alternate one bite of each

Adaptability Request: Move to retract position after every bite

Adaptability Response: A new step was successfully added to the system to move the utensil back after each use. However, this step could not be added for drinks or wipes.

Adaptability Request: Increase auto continue time On the webpage to 100 seconds

Adaptability Response: The system was updated to increase the wait time to 100 seconds for acquiring a bite with a utensil and transferring a drink. However, changes could not be made for transferring a wipe due to an issue with the parameter name.

Adaptability Request: Be quiet

Adaptability Response: The robot-assisted feeding system has been updated to operate more quietly when transferring utensils, drinks, and wipes.

Explanation: Auto continue time on the web interface.

Intervention (Personalization): Bug in behavior tree param for bite acquisition auto-continue for bite task selection page, manually updated to 100.

Explanation: Can move to adaptability page from task selection and back to switch off auto continue.

C.5.7 Other Meals

Training Meal 1: A training meal conducted with CR1 to explain basic usage of the system. This meal allowed the user to become acquainted with the system.

Training Meal 2: A training meal conducted with CR2 to explain basic usage of the system. This meal allowed the user to become acquainted with the system.

Aborted Meal 1: A meal was aborted part-way through due to personal time constraints of CR2.

Aborted Meal 2: A meal with CR2 was aborted part-way through due to a robot falling error. Our hypothesis for this failure is that a voltage drop, reported by the robot firmware, was caused by interference between a router and a power supply. Although the robot remained safe, we determined this error to be significant enough to warrant further testing and mitigation. Thorough testing confirmed the system's safety before we continued.

C.6 OT Evaluation Details

C.6.1 Evaluation Questions

- I was able to personalize the mealtime-assistance system to my requirements for this scenario (5 point Likert scale ranging from Strongly Disagree to Strongly Agree).
- Personalizing the mealtime-assistance system was easy for this scenario (5 point Likert scale ranging from Strongly Disagree to Strongly Agree).
- I would choose the personalized version of the system over the default version for this scenario (Yes / No).

C.6.2 Personalization Requests

Scenario 1: User has limited head mobility and cannot lean the default amount to take a bite. They prefer outside-mouth transfer over inside-mouth, and thus would still prefer to lean some amount.
Adaptability Request: Move robot closer for transfer.
Adaptability Response: The robot was moved closer to you when offering utensils, drinks, and wipes.

Scenario 2: User prefers to dip the food item less.
Adaptability Request: Dip food item less in ketchup.
Adaptability Response: Reduced how far the robot dips food into ketchup.

Scenario 3: User has limited mouth opening and prefers the robot to skewer food with fork tines along the longer axis to avoid wide mouth opening when the fork is horizontal.
Adaptability Request: Skewer item vertically.
Adaptability Response: The robot feeding system was updated to skewer food vertically.

Scenario 4: User is eating in a personal setting and wants to be done quickly, so they choose to decrease the auto-continue time for bite pickup.
Adaptability Request: Decrease auto continue time for bite pick up to five seconds.
Adaptability Response: The automatic wait time before picking up a bite has been reduced to five seconds, as requested.

Scenario 5: User is eating in a social setting and prefers using a button for transfer interactions instead of gestures that may interfere with social interaction.
Adaptability Request: Used button to initiate and show completion of transfer.
Adaptability Response: The robot feeding system now uses a button to start and confirm when handing you utensils, drinks, or wipes.

Scenario 6: User is watching TV while eating and wants to minimize interactions with the web app by removing confirmation pages.
Adaptability Request: Get rid of confirmation page for bite transfer.
Adaptability Response: Removed the confirmation step when transferring a bite, making feeding quicker and smoother.

Scenario 7: User is eating in a personal setting and wants bite transfer to initiate automatically without waiting for user's open-mouth cues.
Adaptability Request: Continue initiating bite transfer without my signal.
Adaptability Response: The system was updated to automatically continue feeding without waiting for your confirmation signal.
Note: Robot confuses request with removing the confirmation page from the web application.
Adaptability Request: Don't wait for my signal to transfer the bite.
Adaptability Response: The robot was updated to automatically feed you each bite without waiting for your confirmation.
Note: Robot confuses again.
Transparency Request: Do I need to signal with opening my mouth to have it transfer a bite or can it automatically transfer with no signal from me.
Transparency Response: You need to signal by opening your mouth to let the

robot know you're ready for a bite transfer. Currently, the robot does not automatically transfer bites without your signal.

Adaptability Request: Auto initiate bite transfer instead of waiting for me to open my mouth.

Adaptability Response: The feeding system was updated to automatically start bringing food toward your mouth without waiting for you to open it.

Scenario 8: User wants the mouth wiper to be closer to their mouth, but other tools to remain at their previous distance.

Adaptability Request: Bring wipe 3 cm from my face before transfer.

Adaptability Response: You requested bringing the wipe closer (3 cm) to your face before transferring, but this distance was outside the allowed safety limits. The system did not accept this change.

Transparency Request: How close can you bring the wipe to my mouth before transfer.

Transparency Response: I can bring the wipe as close as 15 centimeters (about 6 inches) from your mouth before you start cleaning.

Adaptability Request: Bring the wipe as close as possible to my mouth before transfer.

Adaptability Response: The wipe will now be brought closer to your mouth before transferring it.

Scenario 9: User prefers the robot to move quickly.

Adaptability Request: Make the robot as fast as possible.

Adaptability Response: The robot's movements for picking up, using, and putting away utensils, drinks, and wipes have all been set to the fastest possible speed.

Scenario 10: User is eating in a social setting and the robot's bite acquisition and transfer configurations occlude their social partner. They want the robot to reposition to a non-occluding configuration between bites.

Adaptability Request: Retract after every bite transfer.

Adaptability Response: The robot now moves back after every bite it gives you.

APPENDIX D

COLORING BETWEEN THE LINES APPENDIX

Supplementary materials can be found at `https://emprise.cs.cornell.edu/cbtl/`.

D.1 Examples

Example

In assisted feeding (Figure 7.6b), where the robot can move the drink and plate on the table before attaching a utensil to its end effector, CSP variable examples include drinkPose $\in$ SE(2), platePose $\in$ SE(2), utensilAbovePlatePose $\in$ SE(3), and robotAbovePlateConf $\in \mathbb{R}^7$, among others.[1] Constraint examples include collisionFree(drinkPose, platePose), poseIsAbove(platePose, utensilAbovePlatePose), and kinematicallyValid(utensilAbovePlatePose, robotAbovePlateConf).

Example

When the robot observes in h_t that the user is requesting a bite, it generates a bite-specific CSP that includes the variables and constraints mentioned above, among others. After solving the CSP, the robot generates a policy π over low-level robot joint commands that starts by moving the drink and plate to the poses found in $\mathbf{v}^*$. The termination condition τ checks for the successful bite transfer, or for a new request (e.g., drinking).

[1]The notation platePose $\in$ SE(2) is shorthand for V with $N(V)$ = platePose and $D(V)$ = SE(2). (Note that we use SE(2) because the plate is radially asymmetric given the handle and food.) Similarly, collisionFree(platePose, drinkPose) is shorthand for C over platePose and drinkPose.

Example

The generator for the personalized constraint occlusionFree has arity $m = 1$. The initiation condition ι_p evaluates to true on variable V if $N(V)$ indicates that the variable represents a feasible robot joint configuration (not in self collision). The constraint $c_{\theta_{\text{occ}}}$ simulates the robot joint configuration and checks if an occlusion score (computed via [13]) exceeds the learned personalized threshold θ_{occ}.

Example

The user is sometimes asked whether their view is blocked. This question and their response are included in o. When present, the robot extracts the response and its current joint configuration from o and adds the tuple to its classification dataset for the occlusionFree constraint.

Example

The user is sometimes asked which dipping sauce (if any) they would like for their next bite given the food items in the observation o. Their response w and the food items are available in $\mathcal{D}$. To learn the dipPreferred(dip, θ) constraint, a LearningPrompt($\mathcal{D}$) collates all food items and responses and asks an LLM to generate a summary of dipping preferences θ. A ConstraintPrompt(v, o, θ) extracts the current food items from o and asks an LLM whether a user with preferences θ would enjoy the dip v.

Example

For this example, we consider the Books simulation environment (Figure 7.3), where a robot must select, retrieve, and hand over books. CBTL actively learns an LLM-based userEnjoysBook(book, θ) constraint where book

consists of a well-known title and author and θ is a natural language summary of the user's book preferences. At first, the robot does not have any information about the user, but after receiving positive feedback about the book "1984", it infers that they may also enjoy "Fahrenheit 451", another dystopian novel. To maximize entropy, the robot selects "The Double Helix", and learns that the user does not enjoy nonfiction, which implies that they would not like "Cosmos." While learning book preferences, CBTL also actively selects handover poses to rapidly learn the user's functional range of motion [173] (Section 7.4).

D.2 CSP Details: Generating, Solving, and Learning

In this section, we describe details for the CSP generation, solving, and learning techniques that we use in our experiments. Coloring Between the Lines is agnostic to these details.

CSP Generation. Each environment has an associated CSP generator. The CSP generator monitors all observations and actions and is sometimes queried to produce a new CSP. When queried, the CSP generator outputs: (1) a CSP $(\mathcal{V}, C)$; (2) an initial candidate solution $\mathbf{v}_0$; (3) conditional samplers for the solver (see below); (4) a solution-conditioned policy $\pi(a_t \mid h_t, \mathbf{v}^*)$; and (5) a solution-conditioned termination condition $\tau(h_t, \mathbf{v}^*)$. The logic for CSP generation is environment-specific. In the PyBullet environments (Cleaning and Books), we factorize CSP generation into *skills*—for example, moving to a shelf, picking a book, moving to the human, and handing over the book. The logic for sequencing skills is pre-specified in our experiments; this could be generalized using

task planning [73]. Each skill contributes variables (e.g., grasp $\in$ SE(3) for picking), constraints (e.g., collisionFreeGrasp(grasp)), a subpolicy, a sub-termination condition, and zero or more samplers to the CSP generation process. Skills are reused within and between CSPs. For example, the movement skill uses a motion planner (BiRRT) to move between any two points in the environment.

CSP Solving. We use a simple and flexible sampling-based approach to solve CSPs, which performs a random walk over candidate solutions starting with the initialization $\mathbf{v}_0$ produced by the CSP generator. Each subsequent candidate $\mathbf{v}_{i+1}$ is generated from a sampler $\psi(\cdot \mid o, \mathbf{v}_i)$ where o is the most recent observation at the time of CSP generation. Each ψ typically modifies only a small subset of variable values. At every step in the random walk, a sampler is randomly selected from the set of samplers generated by the CSP generator. The first candidate that satisfies all constraints is returned by the CSP solver. We set large timeouts for the solver and do not hit the limit in any of our experiments, i.e., CSP solving always succeeds. For entropy-based active learning, where we have a mathematical program, we need to maximize an objective (mean entropy) in addition to satisfying constraints. We use the same sampling-based approach, but rather than terminating when the first solution is found, we continue until we find $N_{\text{improve}} = 500$ solutions with objective values greater than or equal to the best seen value and then return the best solution.

CSP Supervised Constraint Learning. We update personalized constraint parameters with supervised learning and LLMs. We provide details for supervised learning here and for LLMs in the section below. The following personalized constraints use supervised learning in our experiments: (1) userEnjoysMeal in Cooking; (2) surfacePreferred in Cleaning; (3) poseReachable

in Books; and (4) occlusionFree in the real-robot mealtime assistance system. The surfacePreferred model simply records, for each surface where cleaning was attempted, whether or not the user admonished the robot on the last attempt; any surface not yet cleaned is assumed preferred. The three other constraints each use one or more instantiations of a model $C(v) = 1[\theta_0 \leq f(v) \leq \theta_1]$ where $\theta_0, \theta_1 \in \mathbb{R}$ and $f(v) \in \mathbb{R}$ derives a 1D value from the CSP variable candidate. For example, the poseReachable constraint in the Books environment computes the Euclidean distance between the human's resting hand position and the candidate handover. We analytically compute a posterior distribution over θ_0 and θ_1 given positive and negative examples of the constraint. The maximum likelihood values are used for the constraints and the full posterior is used when computing entropy in active learning.

D.3 Large Language Model Constraint Details

We now provide details for the LLM-based personalized constraints used in our experiments. These include: (1) userEnjoysBook in Books; and in the real robot experiments, (2) dipPreferred; (3) signalPreferred; and (4) verbosityPreferred. As described in Chapter 7, each constraint is characterized by a ConstraintPrompt and a LearningPrompt. We use `gpt-4o-mini` through the OpenAI API for all LLM queries [209].

For entropy-based active learning, we must predict the *probability* that a constraint is satisfied. Consistent with other work, we find that the LLM next-token probabilities do not provide a useful measure of uncertainty; the LLM typically predicts values with extremely high probability, even in cases where its predictions are incorrect. The following trick mitigates this issue in our experiments.

Rather than asking the LLM to return True or False, we ask it to return an answer on a scale from 0 to 10. For example, for userEnjoysBook, we ask: "How much would the user enjoy the book on a scale from 0 to 10, where 0 means hate and 10 means love?" We interpret a response of 8 out of 10 to mean that if the user was asked 10 times whether or not they enjoy a book, they would say yes 8 times and no 2 times. Under this interpretation, the expected probability is $\sum_{i=0}^{10} \frac{i}{10} p_i$, where p_i is the LLM's predicted probability for the next token $i \in \{0, 1, \ldots, 10\}$. We note explicitly that this estimate is an *approximation* rather than a calibrated probability. The derivation above gives no formal guarantee that an aggregated rating of, say, 0.8 corresponds to an 80% probability that the user is satisfied. The aggregation across token-level next-word probabilities mitigates the well-documented tendency of LLMs to assign overconfident next-token probabilities to single-token outputs, but it does not by itself produce calibration. We measure how far this estimate departs from calibration in §D.5. We discuss the resulting risks—in particular, the possibility of being confidently wrong about a preference—as a limitation in §7.5. The full prompt is below.

ConstraintPrompt for userEnjoysBook(book, θ)

Book description: book. User description: θ. I will ask you to rate how much the user would enjoy this book on a scale from 0 to 10. IMPORTANT: if you have already seen this book before, please answer based on the user's previous enjoyment of the book. How much would the user enjoy the book on a scale from 0 to 10, where 0 means hate and 10 means love? Return a number from 0 to 10 and nothing else.

The role of the LearningPrompt is to condense the relevant experience collected thus far into a natural language summary, which serves as the constraint parameter θ. In our experiments, it was feasible to provide all feedback to the

LLM and ask it to summarize; in general, it may be better to provide only the most recent feedback and ask for an update to the most recent parameter, i.e., learn incrementally. To maintain uncertainty in the face of limited data, we found it useful to ask the LLM to generate multiple possible explanations. The entire response is used as θ. The prompt for the Books environment is shown below.

LEARNINGPROMPT for userEnjoysBook(book, θ)

You are a helpful assistant that infers a user's book preferences based on their feedback. Your current estimate of their preferences is θ. You have received the following feedback from the user: $\langle$all feedback in $\mathcal{D}\rangle$. Based on this history, provide an updated description of the user's book preferences. Your description should be in the following format: "I know that the user likes the following books: [list of books] and they do not like the following books: [list of books]. Based on this, here are some possible summaries of their preferences: 1. [summary of preferences] 2. [summary of preferences] 3. [summary of preferences]." Return this description and nothing else. Do not explain.

D.4 Ablating the LLM-Derived Entropy Term

The comparison in §7.4.1 shows that guided exploration beats unguided exploration, but it does not identify which term in the objective is responsible for the gain. In Books, the objective averages over an LLM-based book-preference constraint and a supervised reachability constraint with an analytic posterior, so CBTL's advantage over Free Explore could come entirely from the supervised term. We therefore compare full CBTL, Max-Entropy (Supervised Only), Greedy

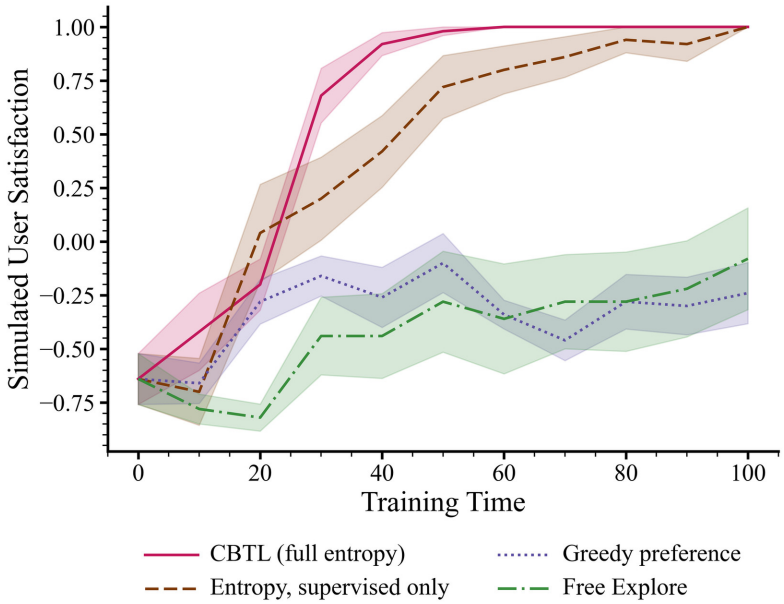


Figure D.1: Ablating the LLM-derived entropy term in Books. Lines are means and shaded regions are standard errors over 10 seeds. Max-Entropy (Supervised Only) computes the entropy objective over supervised constraints alone, excluding the LLM-based book-preference term; it is otherwise identical to CBTL. Greedy Preference replaces the entropy objective with one that maximizes the summed log probability of the same personalized constraints, isolating the choice of objective from the information the objective consumes.

Preference, and Free Explore over 10 seeds under the protocol of Figure 7.4; the gap between the first two isolates the LLM-derived signal, and Greedy Preference separates the choice of objective from the information it consumes. Arm definitions and configurations appear in §D.7.

Both entropy arms reach the maximum satisfaction of 1.0 on all 10 seeds, so they separate on speed rather than on final quality. Full CBTL reaches that ceiling after $3{,}800 \pm 327$ actions against $5{,}200 \pm 814$ for Max-Entropy (Supervised Only), and averages 0.575 ± 0.026 satisfaction across training against 0.415 ± 0.089, a paired difference that favors full CBTL without being separable from noise at this sample size (Wilcoxon signed-rank over matched seeds, $p = 0.11$ for mean

satisfaction and $p = 0.19$ for actions to ceiling). We therefore scope the active learning claim to the supervised constraints and treat the LLM-derived contribution as positive but unresolved. Greedy Preference is the more informative contrast: it consumes the same LLM probabilities under a non-entropy objective, yet never reaches the ceiling on any seed and ends at -0.240 ± 0.142, and full CBTL beats it on every one of the 10 seeds ($p = 0.002$ for both measures), which places the benefit in the entropy objective rather than in the availability of LLM preference estimates.

D.5 Calibration of LLM-Derived Preference Probabilities

This section gives the full methodology and results for the calibration analysis summarized in §7.4.2. The active learning objective of §7.3 requires a probability for every personalized constraint, and for LLM-based constraints that probability comes from a 0–10 rating normalized to $[0, 1]$. Nothing guarantees that the rating is calibrated: a score normalizing to 0.8 need not mean that eight users in ten would accept the option. We measure how far the ratings depart from calibration, whether the departure can be corrected after the fact, and what the departure implies for the exploration the objective drives. We study the dipPreferred constraint, since the web study collected an explicit ground-truth label for it—each participant's reported ideal dip—at every meal.

Protocol and Measures. For each of the 60 participants and each meal $t \in \{1, \ldots, 5\}$, we form ϕ_t by running the dipPreferred LearningPrompt over that participant's feedback from meals $1, \ldots, t-1$ only, then score each dip option available at meal t with the corresponding ConstraintPrompt, converting scores to probabilities by the expected-value aggregation of §D.3. The label is

Table D.1: Calibration and ranking quality of dipPreferred scores. ECE uses ten equal-mass bins. The isotonic rows are cross-fitted: each half of the participants is scored by a model fit on the other half. The base-rate rows predict 1/3 for every option.

		ECE ↓	Brier ↓	AUC ↑
Meals 2–5	Raw score	0.306	0.285	0.690
	After isotonic correction	0.032	0.175	0.708
	Base rate (1/3)	0.000	0.222	0.500
Meal 1 (cold start)	Raw score	0.325	0.288	0.825
	After isotonic correction	0.033	0.125	0.819
	Base rate (1/3)	0.000	0.222	0.500

whether the participant reported that option as their ideal dip. Since ϕ_1 is empty, meal 1 is a cold-start condition reported separately. Every meal offers three dip options, so this yields 720 option-level pairs over meals 2–5 and 180 cold-start pairs, each with a positive base rate of 1/3 since participants report one ideal dip per meal. We bin predictions into ten equal-mass bins and plot empirical acceptance frequency against mean predicted probability, with intervals from a cluster bootstrap that resamples participants rather than individual predictions. We report expected calibration error and Brier score against a marginal base-rate baseline, isotonic regression fit on a random half of participants and evaluated on the other half, and AUC together with top-1 agreement.

Results. Figure D.2 shows the reliability diagram and Table D.1 the summary measures. The raw scores are miscalibrated, and in a specific direction: they are too high almost everywhere. Options scoring between 0.6 and 0.8 are chosen 14% to 22% of the time, and the Brier score of 0.285 is worse than the 0.222 obtained by predicting the base rate of 1/3 for every option. The departure is not a uniform rescaling either. A cluster of options scoring near 0.33 is

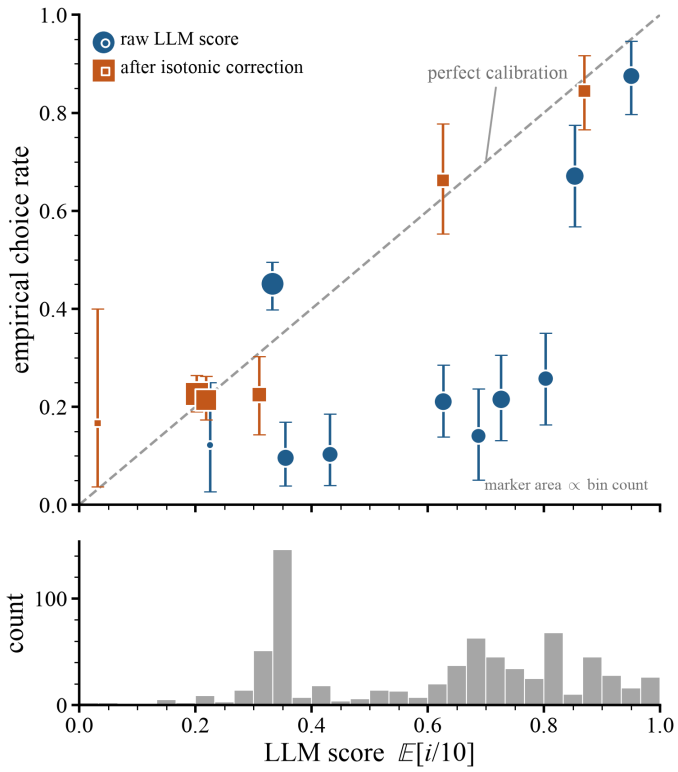


Figure D.2: Reliability diagram for dipPreferred scores on meals 2–5 of the web study. Each point is a bin of option-level predictions, plotted at its mean predicted probability against the fraction of those options participants actually chose; error bars are 95% intervals from a cluster bootstrap over participants. Raw scores (circles) fall well below the diagonal through most of the range; after a cross-fitted isotonic correction (squares) they track it. The lower panel shows the distribution of raw scores.

chosen 45% of the time, more often than any option scoring between 0.35 and 0.80, so no monotone transformation can align the whole range.

What the scores do preserve is a usable ordering. The AUC of 0.690 over meals 2–5 means that a randomly chosen ideal dip outscores a randomly chosen alternative about seven times in ten, and selecting the highest-scoring option matches the participant's choice on 73%, 72%, and 80% of meals 3, 4, and 5, against 62%, 45%, and 62% for always predicting the option most participants chose. Isotonic regression fit on half the participants and applied to the other half

reduces the calibration error from 0.306 to 0.032 and the Brier score from 0.285 to 0.175, better than the base rate, while leaving AUC essentially unchanged at 0.708. Calibration is therefore recoverable after the fact, at the cost of one held-out split.

The cold-start condition behaves differently. With ϕ_1 empty the model falls back on a population prior over food and dips, which ranks well, with an AUC of 0.825, but remains just as poorly calibrated, with an error of 0.325. Predicting the majority option is equally accurate at meal 1, so the cold-start ranking reflects a shared prior rather than anything learned about the individual.

Because the objective of §7.3 consumes these probabilities through an entropy term, we also asked whether recalibration changes which candidate looks most uncertain, and it does. Kendall's τ between the entropy orderings induced by raw and corrected scores is -0.18, and the highest-entropy option within a meal agrees in only 45% of meals, against a chance rate of 33% for three options. The mechanism is visible in the scores: 63% of raw scores exceed 0.5 while only 19% of corrected ones do, so correction moves most options across the point where binary entropy peaks and reverses their order. Miscalibration that is harmless for ranking is therefore not harmless for exploration, and any use of these scores inside the entropy objective should apply the correction first.

Reproducibility. All scores use `gpt-4o-mini` [209] with the prompts and decoding settings of the original study. The preference summaries ϕ_t are replayed from the artifacts generated for that study, so they are the summaries the deployed system used; the per-option scores were computed for this analysis, since the deployed system stored only its top choice. At meals 2–5 the resulting top-1 predictions fall within five percentage points of the accuracies in Figure 7.5(d),

and coincide exactly at meal 2. Analysis scripts and de-identified response data are available at `https://emprise.cs.cornell.edu/cbtl/`.

D.6 Additional Results

We now present additional simulation experiments and results.

Unsatisfiable Personalized Constraints. An additional advantage of CBTL is that there is natural recourse when personalized constraints are unsatisfiable: re-planning without personalized constraints. For example, in the Cooking environment, suppose that ingredient quantities are limited such that the user's preferences cannot be fully satisfied, but a meal can still be made. Given the explicit control flow of CBTL, the robot can notify the user that their preferences cannot be satisfied and give them the option of a non-personalized solution (some meal).

Table D.2: Planning with unsatisfiable personalized constraints in the Cooking environment (mean ± stdev).

Method	Task Success	Preferences Met
CBTL (Replan)	1.00 ± 0.00	0.21 ± 0.09
CBTL (No Replan)	0.31 ± 0.11	0.95 ± 0.09
No Personalization	1.00 ± 0.00	0.00 ± 0.00

To illustrate this flexibility, we run an additional experiment in the Cooking environment where ingredient quantities are limited so that meals are always possible, but when each ingredient quantity respawns, there is only a 50% chance that there will be enough to satisfy the user's preferences. We compare CBTL with and without non-personalized re-planning to No Personalization across 10 seeds with 10 evaluation tasks per seed after 2500 training steps.

In Table D.2, we show that CBTL maintains perfect task success rates (meals served) by re-planning without personalized constraints after planning in the full CSP fails. In contrast, CBTL (without replanning) consistently satisfies user preferences, but at the expense of task success; No Personalization never satisfies user preferences. In future work, we plan to explore more sophisticated fall-back strategies (cf. minimum constraint removal [94]) and questions related to human-robot interaction with unsatisfiable preferences.

Nonstationary Preferences. User preferences and functionalities can change over very long-term deployments. In this additional experiment, we consider three variations of CBTL and assess the extent to which each can handle these nonstationary distributions. In particular, we consider: (1) No Adapt, which assumes stationary user preferences; (2) User Reset, which resets the memory buffer in sync with preference shifts (analogous to explicit user feedback triggering relearning); and (3) Periodic Reset, which resets at fixed intervals (every 1000 steps) independent of preference shifts. We compare these approaches in a variation of the Cooking environment where hidden user preferences about ingredient quantity and temperature are periodically perturbed. Concretely, every 3000 simulation steps, we shift the user's preference intervals for each ingredient quantity and temperature by a scalar sampled uniformly at random from $[0.9, 1.1]$.

As shown in Figure D.3, CBTL (User Reset) rapidly relearns personalization constraints after each shift, achieving consistent high satisfaction with minimal degradation during the relearning process. CBTL (Periodic Reset) similarly recovers quickly though with more instability due to unnecessary resets. In contrast, CBTL (No Adapt) and Free Explore fail to adapt and suffer from degrading

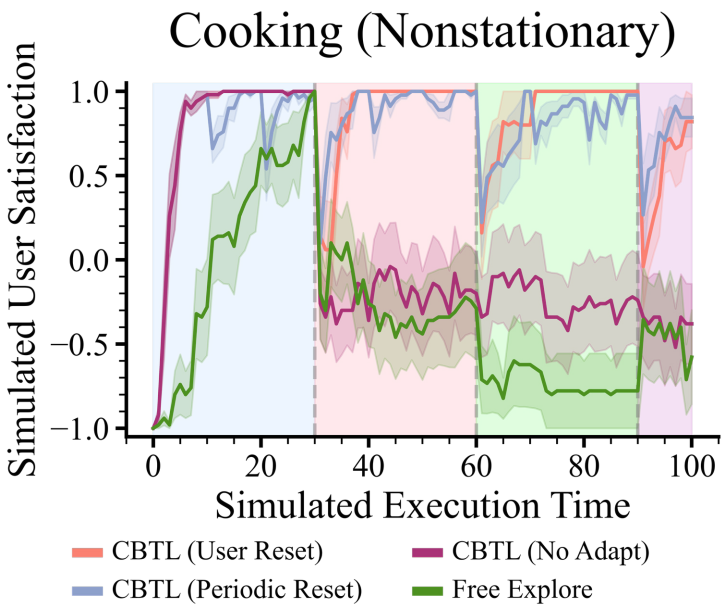


Figure D.3: CBTL adapts to shifting user preferences in nonstationary environments. CBTL (User Reset) is identical to CBTL (No Adapt) before the first preference shift (3000 steps).

performance after each shift. These findings highlight the flexibility of extending CBTL to nonstationary settings and motivate future work on automatically detecting preference shifts for even more adaptive lifelong personalization.

D.7 Simulation Experiment Details

In this section, we provide additional details about the experimental setup, environments, and approaches for the simulation results presented in Chapter 7.

Experimental Setup. All approaches and all environments are run over 10 random seeds. We use a CPU-only cluster where machines with 64 GB memory run Ubuntu 20.04. Each (approach, environment, seed) trial has a timeout of 6 hours, but typically takes between 30 and 90 minutes.

Environments. We now provide further details about the simulation environments to supplement the descriptions given in Chapter 7.

Cooking: We use two ingredients and four pots in the main experiments. Ingredients heat up at a fixed rate when in pots and respawn after meals are served. The following actions are available in the environment: (1) add a certain quantity of a certain ingredient to a certain pot; (2) move a certain pot to the stove at a certain position; (3) wait; (4) serve meal. When the meal is served, the user reports whether or not they like the temperature and quantity of each ingredient that is currently in some pot based on hidden preference specifications. A preference specification for an ingredient temperature or quantity is characterized by an upper and lower bound. User satisfaction is -1 if any preference is violated when a meal is served and 1 otherwise. We use 10 tasks to evaluate the approach checkpoints, where each task finishes when some meal is served. Evaluation is performed after every 100 actions and trials are run for a total of 10,000 actions.

Cleaning: The robot must pick up and use a dusting tool to wipe surfaces (see video)[2]. There are seven surfaces in the environment (three shelves in a bookcase and four side tables), which are spread out around the environment, necessitating robot base movement. Dust respawns quickly after the robot cleans. The user has hidden preferences that only two surfaces should be cleaned by the robot. If the robot starts to clean one of the other surfaces, the user expresses displeasure. Actions in the environment are delta joint positions for the robot arm, delta positions and rotations for the robot base, and open/close for the robot gripper. Skills implemented as part of the CSP include base movement,

[2] `https://emprise.cs.cornell.edu/cbtl/`

arm movement, picking, placing, and wiping. The movement skills use BiRRT for motion planning with IKFast for inverse kinematics and PyBullet for collision checking. Picking and placing use simple grasp and placement samplers that assume cuboid objects. Wiping uses motion planning to get into an initial position before executing a predefined up-down, left-right motion. We use 10 tasks to evaluate approach checkpoints. For each task, we randomly sample one surface to cover with obstacles (preventing wiping). The robot cleans one surface and then the user reports -1 or 1 satisfaction according to their hidden preferences. Evaluation is performed after every 500 actions and trials are run for a total of 5,000 actions.

Books: The robot must hand over books to a user lying in bed. In our main experiments, we use the following nine well-known books:

- *Cosmos* by Carl Sagan.
- *Pride and Prejudice* by Jane Austen.
- *The Hitchhiker's Guide to the Galaxy* by Douglas Adams.
- *The Immortal Life of Henrietta Lacks* by Rebecca Skloot.
- *The Diary of Anne Frank* by Anne Frank.
- *Into the Wild* by Jon Krakauer.
- *Moby Dick* by Herman Melville.
- *The Lord of the Rings* by J. R. R. Tolkien.
- *And Then There Were None* by Agatha Christie.

The user has the following hidden natural language preferences: "I only like two kinds of books: fictional romances, and grand, nonfictional reflections on the universe." When they are handed a book, short feedback is generated using

an LLM with access to the hidden preferences. The user also has a hidden functional range of motion. If the robot attempts to hand over a book outside their range of motion, the user responds: "I can't reach there." Actions and skills are the same as in the Cleaning environment. We use 10 tasks to evaluate approach checkpoints. For each task, a random subset of the books are available. The user reports 1 satisfaction only if the robot hands over a book that they enjoy on the first attempt. Evaluation is performed after every 1,000 actions and trials are run for a total of 10,000 actions.

Baselines. We next provide additional details about the baselines used in simulation experiments.

No Personalization. This baseline is the same as CBTL, except personalized constraints are never updated. For example, in Books, the user preferences are initially "unknown", and they remain unknown in perpetuity.

Free Explore. This baseline is the same as CBTL, except that whenever a CSP is generated during exploration (i.e., any time other than during the checkpoint-loaded evaluation tasks), personalized constraints are not included. We call this "Free Explore" because the baseline is free to explore any solutions that satisfy the non-personalized constraints.

Exploit Only. This baseline is the same as CBTL, except that whenever a CSP is generated during exploration, the personalized constraints (with their current learned parameters) are always included. We call this "Exploit Only" because the baseline exploits its current learned models at all times. Like in reinforcement learning, this approach can easily get stuck in local minima.

Epsilon-Greedy. This baseline is the same as CBTL, except that whenever a CSP is generated, it includes personalized constraints (all or nothing) with probability $\epsilon = 0.5$. We varied the value of ϵ and found that $\epsilon = 0.0$ and $\epsilon = 1.0$ match Free Explore and Exploit Only respectively, but there is no clear middle ground that avoids the pitfalls of both. We also note that Epsilon-Greedy and Exploit Only obtain equal performance in the Cooking results. This is due to the fact that both get trapped in the same local minima and the fact that we carefully control random seeding throughout experiments. Discrepancies emerge with lower values of ϵ and more random seeds.

Max-Entropy (Supervised Only). This ablation runs in Books, the only environment whose entropy objective mixes an LLM-based constraint with a supervised one. It is identical to CBTL except that the objective ranges over $C_p^{\text{sup}} \subset C_p$, the constraints learned by supervised classification. In Books this leaves the reachability constraint, with the analytic interval posterior of §D.2, and drops the LLM-based userEnjoysBook term. Everything else is unchanged: userEnjoysBook is still learned from feedback, still enforced when the robot is not exploring, and still applied during evaluation. The ablation therefore isolates the role of the LLM-derived probabilities in choosing what to try next. We use the same 10 seeds, 10 held-out evaluation tasks, 1,000-action evaluation interval, and 10,000-action budget as the main Books runs, so Figures 7.4 and D.1 are directly comparable.

Greedy Preference. This arm holds the information fixed and changes the objective. It is identical to CBTL except that, during exploration, the objective maximizes the summed log probability of the personalized constraints rather than

their mean Bernoulli entropy, so the robot proposes whatever it currently believes the user most prefers instead of whatever it is least certain about. The LLM-based userEnjoysBook term is still consumed, and everything else matches the ablation above. Because it reads the same LLM probabilities as CBTL, the comparison distinguishes the value of the entropy objective from the value of having LLM preference estimates at all.

D.8 Web-Based User Study Details

In this section, we provide additional details and analysis for our web-based study.

Demographics. We recruited 60 participants who completed the study online. The participants were not compensated. They ranged in age from 19 to 68 years old with a mean of 31. Given the options Male, Female, Non-binary, Prefer Not To Say, and Other, 36 participants selected Male and 24 selected Female. We asked the participants if they had prior robot experience (29 yes, 31 no) and any experience being fed as an adult (13 yes, 47 no).

Website Design. We built a custom website for the study using pure JavaScript. Although the study presents to the participant as a typical survey, the participant's earlier answers impact the choices that they are later shown, which makes it difficult to use existing survey platforms. All together, there are 228,637,728 "paths" that a participant can take. However, we can factorize over question type (occlusion, verbosity, ready signal, dip preferences). This makes it possible to pregenerate all possible model predictions offline. We did not record time, but participants reported that the survey took approximately 15 minutes.

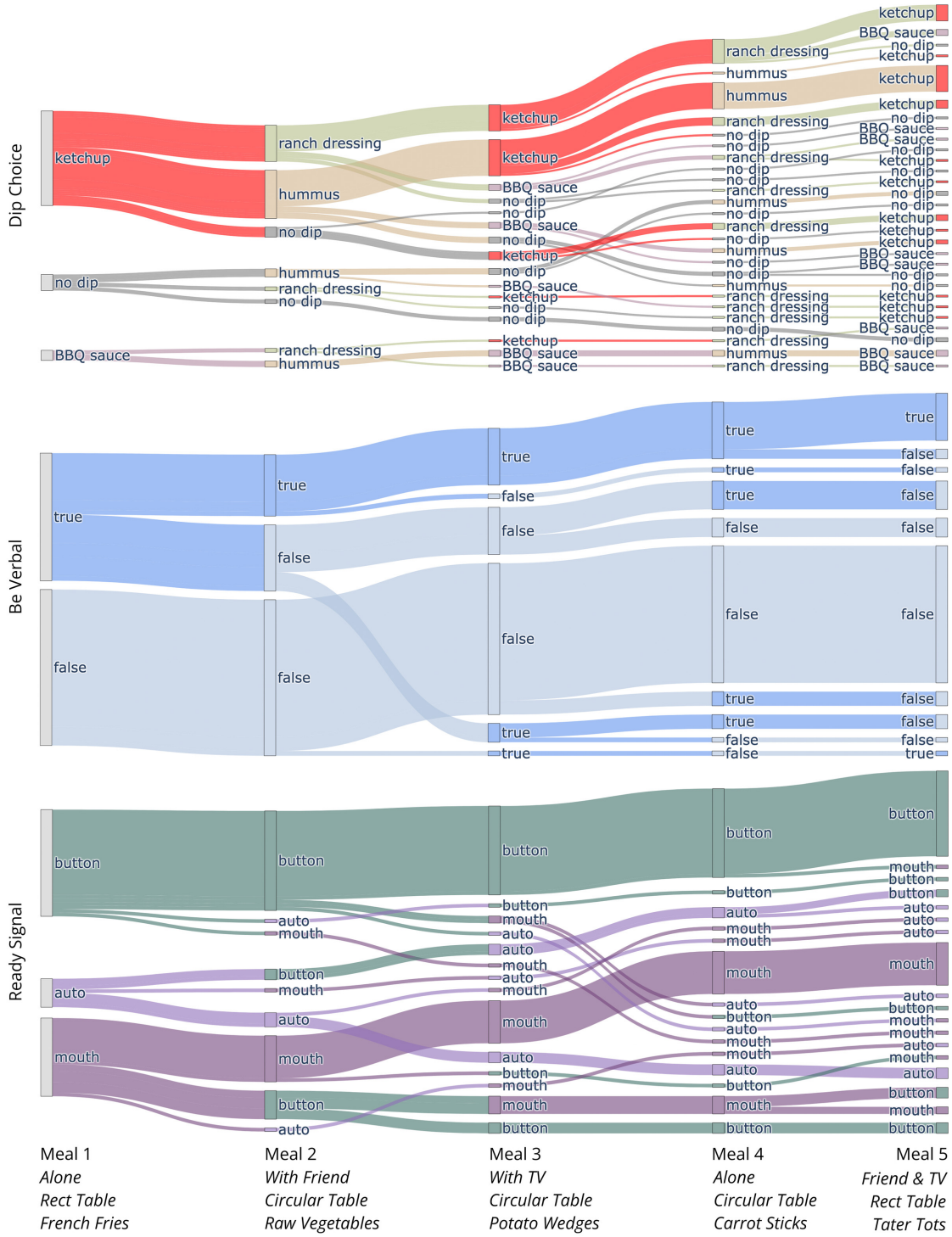


Figure D.4: Sankey diagrams for the three categorical questions in the web study. We see diversity in all three categories. This confirms that the predictive ability of CBTL is indicative of personalization.

Additional Analysis. In Chapter 7, we presented results that show (1) participants prefer CBTL after the first meal; and (2) CBTL is increasingly able to predict participant responses over time. To further confirm CBTL's personalization capabilities, we present results in Figure D.4 showing that there is substantial variation between participants in terms of their preferences. For example, in the case of dip choice, there were 29 distinct paths for 60 participants. This diversity underscores the importance of personalization and confirms that CBTL personalizes to individual users.

Returning to the main results, in Figure 7.5(c), we note the absence of a monotonic improvement in terms of preference for CBTL. For example, participants had a stronger preference for CBTL on Meal 3 than on Meal 4 on average. We hypothesize that this is primarily due to the fixed meal ordering and the fact that participants may have varying overall preference strengths for different meals. For instance, a participant may strongly prefer ketchup in Meals 1, 3, and 5, and weakly prefer hummus in Meals 2 and 4. This hypothesis is supported by the fact that CBTL's ability to predict participant responses *does* monotonically improve (Figure 7.5(d)).

D.9 In-the-wild Deployment Questionnaires

We administered the following questionnaires:

D.9.1 Post-Session Questionnaire

Q1. To what extent did the robot correctly predict your preferred *food item and dip combinations*?

- Very Low Low Neutral High Very High

Q2. To what extent did the robot correctly predict your preferred *method of signaling that you were ready for a bite*?

- Very Low Low Neutral High Very High

Q3. To what extent did the robot correctly predict whether you wanted it to *speak aloud*?

- Very Low Low Neutral High Very High

Q4. To what extent did the robot *block your view of the TV or your friend* during mealtime assistance?

- Very Low Low Neutral High Very High

D.9.2 Post-Study Questionnaire

Q1. For which method did you have to work harder to ensure that the robot aligned with your preferences?

- CBTL FEAST

APPENDIX E

TABLE FOR TWO APPENDIX

E.1 Web Interface

The community researcher (CR) operates the robot through a Vue.js [287] web application on her iPad, which exchanges JSON messages with the robot over rosbridge [50]: the application publishes her taps as messages to the robot, and the robot moves the application to a page by publishing that page's name. Every message the application sent to the robot over the deployment is logged with its timestamp, and the counts below are taken from these logs; the base joystick and the settings overlay use channels of their own. Figure E.3 shows the pages as they appeared at meal 20, from the screen recording of her iPad; Fig. E.4 shows the pages that meal 20 did not reach, from the recordings of other meals. Table E.1 lists the pages, what she can do on each, and what it sends to the robot. The screenshots are collected in Appendix E.7. Her name, which the header shows on every page, is erased from all screenshots.

Table E.1: **The pages of the web interface.** *Panel* gives the page's screenshot in Fig. E.3 or E.4; *Sends* gives what the message each page publishes to the robot on `/webapp_to_robot` carries, with fields in roman and literal values in code. The FEAST personalization pages reached from *Personalization* (language requests and gestures [122]) are omitted: she sent no request from them during the deployment.

Page	Panel	Sends	Contents and controls
Start and end of a meal (4)			
Home	E.3a	state `home`, status `start_meal`	*Start Meal*
Meal context	E.3b, c	meal, setting, time of day	Meal, setting, and time of day, one per step
End-of-meal survey	E.3t	item, value	Eleven 7-point items and one open question, one per screen

Page	Panel	Sends	Contents and controls
Closing	E.3u	—	Confirms that her answers were saved
Preference pages (2)			
Preference page	E.3d–g	dimension, value, tap or countdown	One dimension, predicted value preselected, 30 s countdown; the bite ordering takes free text
Settings overlay	E.4a	dimension, value	Every dimension answered so far this meal, editable at once; opened from the gear on every page but the context and preference pages
Confirmation pages (6)			
Detection	E.3h–l	detection type, `confirm`, `redo`, or `correct_color`	Handle, plate attachment, microwave button, table, or sink placement drawn on the camera image; *Looks Correct*, *Redo*, *Correct Color*
Color correction	E.4b	color, range, `rerun` or `confirm`	Tap the plate attachment, adjust the color range, *Rerun*, *Confirm*
Plate release	E.3m	location, `confirm`	Before the plate is let go in the microwave, on the table, or in the sink; *Continue*
Parking	E.4c	`ok` or `adjust`	On arrival at a task location; *Position OK* or *Adjust Position*, which opens the base joystick
Ready to eat	E.3o	`confirm`	Plate on the table; *Yes, I'm ready*, with no countdown
Pickup check	—	`confirm` or `cancel`	After a bite, the drink, or the wiper is picked up; *Continue*, and for a bite *Retry*
Feeding pages (5)			
Task selection	E.3n	task	*Take a Bite*, *Take a Sip*, *Wipe Mouth*, *Personalization*, *Finish Feeding*
Next task	E.4f	task, tap or countdown	Task selection after a bite or sip, with a countdown on the previous task
Bite selection	E.3q, r	food item, detection, dip	Plate with numbered detections, proposed bite, food items to switch to, dip; *Pickup Bite* with countdown; *Trouble picking this up?*
Manual keypoints	E.4d, e	skill, image points	Choose a skill, then tap one or two points on the full camera image
Executing	E.3p, s	—	Previous, current, and next skill; a description of what the robot is doing; the trigger it is waiting for

Page	Panel	Sends	Contents and controls
Teleoperation pages (3)			
Arm	E.4g	command, step size, `done`	Task-space moves and rotations at three step sizes, joint-space moves, gripper, retract, *Done*
After teleoperation	E.4h	`redo` or `next`	*Move the Base*, *Redo Skill*, *Next Skill*
Base	E.4i	velocity on `/cmd_vel_teleop`	Joystick, drag to drive and release to stop; *Resume* or *Done* during a navigation skill

Elements shared across pages. A header carries a status line describing what the robot expects of her, a *Robot Arm Control* button that opens arm teleoperation, and, on every page but the context and preference pages, a settings gear that opens the settings overlay; while the robot navigates or waits for her to choose a task, a *Robot Base Control* button opens base teleoperation as well. While a skill executes, a strip under the header shows the previous, current, and next skill of the plan, and the page shows a language description of what the robot is doing (Fig. E.3p). A wired button at her iPad, read through the iPad's audio input until meal 20 and through the robot's computer from meal 21 (Appendix E.5), can request a bite, a sip, or a wipe in place of the open-mouth trigger.

Start and end of a meal. The home page offers a single *Start Meal* button (Fig. E.3a). On the context page, she then chooses the meal, the setting, and the time of day (b, c), which form the context the predictor conditions on (Appendix E.2). After the plate is in the sink, the end-of-meal survey presents the 11 daily items and the open question of Appendix E.12 one per screen (t), and a closing page confirms that her answers were saved (u).

Preference pages. Each predicted dimension appears on its own page, at the point in the meal where it takes effect, with the predicted value preselected

and a 30-second countdown after which it is accepted (Fig. E.3d). Three dimensions are asked before the robot leaves for the refrigerator (robot speed and the two confirmation behaviors for navigation and manipulation), one after the plate leaves the refrigerator (microwave time), and the remaining 17 at the table once she presses *Yes, I'm ready* (o). When she selects a different value (e), the page sends it to the robot, which fixes it and re-predicts every dimension still open (f); the pages that follow show the re-predicted values (g). The bite ordering, the one free-text dimension, offers the predicted sentence or a field she can type or dictate into. The settings overlay (Fig. E.4a) lists every dimension answered so far in the meal and lets her change at once any that has not yet taken effect, while those set during meal preparation are shown but locked; she used it 20 times over the month. The nine environment parameters have no preference page: a plate color is set on the color-correction page and a parking offset with the base joystick; both are described below.

Confirmation pages. A confirmation page precedes each step that depends on perception or ungrasps the plate. The detection page serves five detections: door handles, found by GroundingDINO and located on the depth image, and shown with the hinge and the direction in which the door opens (Fig. E.3h); the plate attachment, found by color and shown as the pixels that matched (i); the microwave's start button, returned by MolmoPoint as a point (j); and the placement points on the table (k) and in the sink (l). She taps *Looks Correct* or *Redo*, and for the plate attachment *Correct Color*, which opens the color-correction page (Fig. E.4b): she taps the attachment in the camera image, adjusts the matched color range with a slider, reruns the detection until it is right, and confirms, which records the color for that location. The same page opens on its own when the robot cannot find the attachment at all. Before each release of the plate in

the microwave, on the table, and in the sink, a release page asks her to confirm (Fig. E.3m). On arrival at a task location, a parking page asks whether the base is where she wants it; *Adjust Position* opens the base joystick, and the adjustment she makes is recorded as that location's parking offset (Fig. E.4c). The countdown on each of these pages is itself a personalization dimension: *Confirm manipulation* sets it for the detection and release pages, and *Confirm navigation arrival* for the parking page, each to one of skip, a 15-, 30-, or 60-second countdown, or waiting for her. Skip removes the page from the meal. Over the month, she confirmed 160 detections, 8 of them by letting the countdown run out, asked for 12 re-detections and 12 color corrections, and confirmed 58 plate releases, 1 by countdown; the color-correction page, opened by her or by a failed detection, recorded the 19 plate-color corrections of Section 8.5.

Feeding pages. Task selection offers the three tasks (*Take a Bite*, *Take a Sip*, and *Wipe Mouth*), a *Personalization* menu, and *Finish Feeding* (Fig. E.3n). The bite-selection page shows the plate with each detected item boxed and numbered, the bite the robot proposes, the other foods she can switch to, and the dip (q); she can accept, tap another box, or swap to another item (r), and the proposed bite is picked up when the countdown set by *Show bite-selection page* runs out. If no bite is detected, or the robot cannot separate the item she wants, the *Trouble picking this up?* button opens the manual keypoint mode: she chooses a skill and taps one or two points on the full camera image (Fig. E.4d, e), which the robot uses instead of the detection; she did so 25 times over 11 meals. After each bite or sip, the next-task page repeats the task selection with a countdown on the previous task, set by *Next-task countdown* (Fig. E.4f). While the robot waits for her to request a bite, sip, or wipe, the executing page says which trigger it is waiting for (Fig. E.3s). The *Personalization* menu retains FEAST's language and

gesture personalization [122]; she sent no request from it during the deployment, and changed every preference on the preference pages or in the settings overlay instead.

Teleoperation pages. On the arm page (Fig. E.4g), she moves the gripper in task space, up, down, left, right, reaching out and pulling back, tilting, turning, and rolling, at a fine, medium, or coarse step, opens and closes the gripper, and runs the predefined retract motion; a joint-space tab moves one joint at a time, though she never used it. When she presses *Done*, a page asks what the robot should do next: *Move the Base*, *Redo Skill*, or *Next Skill* (h). The base page (Fig. E.4i) is a joystick she drags to drive the base and releases to stop; during a navigation skill it offers *Resume* to hand the drive back to the planner, and *Done* to end it where she stopped. Appendix E.10 counts how she used both.

Order of pages over a meal. Table E.2 gives the order in which the pages appeared at meal 20, a social meal with a friend, reconstructed from the application's message log and the robot's event log and checked against the screen recording. Meal 20 had the parking pages set to skip and reached neither teleoperation page; where a meal differs, the pages above appear at the step they belong to.

E.2 Preference Prediction

This section formally states the prediction problem summarized in Section 8.3, describes how the predictor deployed in the home realizes it, and gives the offline loop used to score predictors against the deployment logs.

Table E.2: **The pages of meal 20 in order.** *Min* counts minutes from *Start Meal*; *Detail* gives the skill or detection the page belongs to, or what happened on it. The feeding loop repeats task selection, bite selection, and execution for each of 15 bites and 3 sips.

Min	Page	Detail
Start of the meal		
0.0	Home	*Start Meal*
0.3	Meal context	meal, setting, time of day
1.2	Preference pages	1–3 of 3: robot speed, confirm navigation arrival, confirm manipulation
Preparation		
1.7	Detection	refrigerator handle
2.9	Detection	plate attachment, refrigerator
4.3	Preference page	1 of 1: microwave time
5.2	Detection	microwave handle
6.6	Plate release	microwave
7.8	Detection	microwave button
8.7	Detection	microwave handle, reopening
9.8	Detection	plate attachment, microwave
14.0	Detection	placement point, table
14.5	Plate release	table
15.2	Ready to eat	*Yes, I'm ready*
Feeding		
15.2	Preference pages	1–17 of 17; one correction, one re-prediction
17.9	Task selection	*Take a Bite*
18.5	Bite selection	she swaps the proposed item
18.8	Executing	waiting for her button press
21.5	Next task, bite selection, executing, repeated until 51.1	
53.6	Task selection	*Wipe Mouth*
54.7	Task selection	*Finish Feeding*
Clearing		
55.6	Detection	plate attachment, table
56.2	Navigation to the sink stops; the application reloads to the home page and she finishes feeding again	
63.6	Detection	placement point, sink
63.7	Plate release	sink
End of the meal		
63.8	End-of-meal survey	—
72.7	Closing	—

E.2.1 Personalization Dimensions

Table E.3 lists the 30 personalization dimensions with their options and the point in the meal at which each is asked. Every categorical dimension is predicted from its option list; the bite ordering is predicted as one sentence grounded in the foods on the plate, such as which item comes first and which dip goes with which item. The environment parameters are continuous. A plate color is a hue, saturation, and value with a matching range, set when she corrects the color on the color-correction page (Appendix E.1); a parking offset is a correction to the base's recorded waypoint, set when she drives the base on arrival. At meal 1, the plate colors began from a single calibrated value and the parking offsets from zero. The dining table and the movable table began as one location; the system began choosing the table from the setting at meal 4 (Appendix E.5).

Table E.3: **The 30 personalization dimensions and their options.** The 21 meal preferences are grouped by what they govern: when the robot checks with her, the signals between them, settings fitted to her functional abilities, the bite itself, how the robot moves, and heating; Section 8.3 counts the fit to her abilities and the bite together, as five dimensions on the bite. Within a group, dimensions are listed in the order they take effect in a meal. *Asked* is the point in the meal at which the dimension's preference page appears: before the robot leaves for the refrigerator (*start*), after the plate leaves the refrigerator (*fridge*), or at the table once she is ready to eat (*table*). The environment parameters have no preference page and are set at each location, on pickup or on arrival. Dimensions marked * held the same value at all 22 meals.

Dimension	Asked	Options
Meal preferences (21)		
Checks with her (5)		
Confirm navigation arrival	start	skip • countdown 15, 30, or 60 s • wait for me
Confirm manipulation	start	skip • countdown 15, 30, or 60 s • wait for me
Show bite-selection page*	table	countdown 15, 30, or 60 s • wait for me
Confirm feeding pickup*	table	skip • countdown 15, 30, or 60 s • wait for me
Next-task countdown	table	countdown 15, 30, or 60 s • wait for me

Dimension	Asked	Options
Signals (8)		
Robot's ready-to-feed cue	table	speech • LED • speech and LED • no cue
Completion signal	table	speech • LED • speech and LED • no cue
Bite, drink, and wipe trigger (3)	table	open mouth • button • proceed automatically after a pause
Done detection (bite)	table	perception (force/torque) • button • proceed automatically after a pause
Done detection (drink, wipe) (2)	table	perception (head nod) • button • proceed automatically after a pause
Fit to her abilities (3)		
Skewering axis*	table	parallel • perpendicular (to the item's major axis)
Transfer mode*	table	outside mouth • inside mouth
Outside-mouth distance*	table	near • medium • far • not applicable
Bite (2)		
Bite dipping	table	do not dip • less • more
Bite ordering	table	free text, such as the order of items and which dip goes with which item
Motion (2)		
Robot speed*	start	slow • medium • fast
Retract between bites	table	yes • no
Heating (1)		
Microwave time	fridge	no microwave • 30 s • 1 min • 2 min
Environment parameters (9)		
Plate color (4)		
Refrigerator, microwave, dining table, movable table	on pickup	hue, saturation, value, and a matching range; set on the color-correction page
Parking (5)		
Refrigerator, microwave, dining table, movable table, sink	on arrival	Δx, Δy within ±0.5 m and $\Delta\theta$ within ±45° of the recorded waypoint; set by driving the base

E.2.2 Problem Formulation

We consider a single user u interacting with the robot over a sequence of meals $t = 1, \ldots, T$. The user is characterized by fixed physical functioning abilities U_u^{phys} and by a slowly evolving latent state $U_{u,t}^{\text{pref}}$ governing how preferences are

formed. The latter is intrinsic to the user, is not directly observable by the robot, and may drift over the horizon of a deployment.

At each meal t, the robot observes an exogenous mealtime context X_t: the meal, the dining context (personal, social, or watching television) and the time of day. The user additionally carries a transient latent state Y_t, varying from meal to meal, that captures factors such as fatigue, hurriedness, or a willingness to try something different. Neither $U^{\text{pref}}_{u,t}$ nor Y_t is observable.

For each meal, the user holds a preference vector

$$P_t = \left(p_t^{(1)}, \ldots, p_t^{(M)}\right) \in \mathcal{P}, \qquad \mathcal{P} := \prod_{i=1}^{M} \mathcal{P}^{(i)},$$

where $\mathcal{P}^{(i)}$ is the feasible set of dimension i — a finite set of options for the categorical dimensions, a bounded continuum for the plate-attachment colors and parking-pose offsets, and free text for the bite ordering. In this deployment, $M = 30$.

We assume that preference vectors are generated by a shared structural mechanism

$$P_t \sim f\left(U^{\text{phys}}_u,\ U^{\text{pref}}_{u,t},\ X_t,\ Y_t\right),$$

where f defines a joint distribution over $\mathcal{P}$. That the distribution is joint rather than a product of per-dimension marginals is the substantive assumption of this work: the components of P_t are dependent because they are generated from shared latent quantities, which is what makes a correction on one dimension informative about the others.

Interactive prediction. At the beginning of meal t, the robot produces an initial prediction $\hat{P}_t^{(0)}$. The user then confirms or corrects dimensions one at a

time, each correction revealing the ground-truth value of a single dimension. After $m \geq 0$ corrections, let $S_{t,m} \subseteq \{1, \ldots, M\}$ denote the set of revealed indices with $|S_{t,m}| = m$, let $P_{t,S_{t,m}} = \{p_t^{(i)} : i \in S_{t,m}\}$ denote their ground-truth values, and let $\bar{S}_{t,m} = \{1, \ldots, M\} \setminus S_{t,m}$ be the dimensions still open. A dimension the user confirms is likewise fixed for the remainder of the meal and is supplied to the predictor, so $S_{t,m}$ indexes every dimension whose value has been settled, by confirmation or by correction, while m counts the corrections alone. The robot conditions on the interaction history $\mathcal{H}_{t-1}$ and on its own earlier predictions this meal, and outputs

$$\hat{P}_t^{(m)} = g_\theta\Big(U_u^{\text{phys}},\ X_t,\ \mathcal{H}_{t-1},\ S_{t,m},\ P_{t,S_{t,m}},\ \hat{P}_t^{(0:m-1)}\Big).$$

The predictor g_θ never observes $U_{u,t}^{\text{pref}}$ or Y_t directly, and must reason about them through the history and the corrections so far.

Loss and objective. Corrected dimensions are fixed, so prediction reduces to inferring the remaining components $P_{t,\bar{S}_{t,m}}$. With ℓ_i a per-dimension loss and $w_i \geq 0$ optional importance weights,

$$\mathcal{L}_{t,m}(P_t, \hat{P}_t^{(m)}) = \sum_{i \in \bar{S}_{t,m}} w_i\, \ell_i\Big(p_t^{(i)}, \hat{p}_t^{(m,i)}\Big),$$

and the learning objective is to minimize its expectation over the latent states and the stochasticity of f.

E.2.3 Realization

The predictor is a single call to a large language model, which returns the full bundle. The interaction history is carried verbatim: $\mathcal{H}_{t-1} = \{h_1, \ldots, h_{t-1}\}$, where each episode $h_\tau = (\tau,\ X_\tau,\ P_\tau^*)$ records the context of meal τ and the bundle finally accepted in it, serialized as text and supplied in chronological order.

No summarization, retrieval, or windowing is applied — the model conditions on the complete record. Alongside it, the call receives a description of U_u^{phys}, the current context X_t with the food on the plate, the dimensions confirmed or corrected so far this meal, and the model's own earlier predictions of the meal, so that it can revise its reading of Y_t where a correction has contradicted one of them. Predictions come from Claude Opus 4.8 at medium reasoning effort.

E.2.4 Offline Evaluation Loop

The deployed joint predictor cost the CR 57 corrections over the 22 meals, as counted during the deployment. To compare this observed workload with alternative predictors, we replay each meal under a common offline protocol (Algorithm E.1). Each alternative predictor is shown the meal's context and predicts the bundle; any open dimension that disagrees with the bundle the CR finally accepted is eligible to be corrected; one such dimension is revealed and the predictor runs again; the meal ends when no open dimension disagrees.

The one respect in which this departs from the deployment is the choice of which dimension to reveal. In the home, the CR corrects dimensions in the order the system surfaces them, and her counterfactual order under a different predictor is unknowable, so the loop reveals a uniformly random disagreeing dimension and results are averaged over reveal orders.

A correction is counted whenever the CR sets a value the system did not already hold. This includes the three occasions on which she revised a dimension she had already settled earlier in the same meal; these reflect her own reconsideration rather than a prediction failure, and are counted against both LLM arms

alike. The count-based arms are scored under this same protocol, each predicting day t from days 1 to $t-1$.

Algorithm E.1: Offline evaluation at meal t

Input: U_u^{phys}, history $\mathcal{H}_{t-1}$, context X_t, accepted bundle P_t^*
Output: corrections m_t^*

$m \leftarrow 0$, $S_{t,0} \leftarrow \emptyset$;
repeat
  $\hat{P}_t^{(m)} \leftarrow g_\theta(U_u^{\text{phys}},\ X_t,\ \mathcal{H}_{t-1},\ S_{t,m},\ P_{t,S_{t,m}},\ \hat{P}_t^{(0:m-1)})$;
  if $\exists\, i \in \bar{S}_{t,m} : \hat{p}_t^{(m,i)} \neq p_t^{*(i)}$ **then**
    $j \sim \text{Uniform}\Big(\{i \in \bar{S}_{t,m} : \hat{p}_t^{(m,i)} \neq p_t^{*(i)}\}\Big)$;
    $S_{t,m+1} \leftarrow S_{t,m} \cup \{j\}, \quad P_{t,S_{t,m+1}} \leftarrow P_{t,S_{t,m}} \cup \{p_t^{*(j)}\}$;
    $m \leftarrow m + 1$;
until $\hat{P}_t^{(m)} = P_t^*$ *on all* $i \in \bar{S}_{t,m}$;
$m_t^* \leftarrow m$;
$\mathcal{H}_t \leftarrow \mathcal{H}_{t-1} \cup \{h_t\}, \quad h_t = (t,\ X_t,\ P_t^*)$;

Predictors are compared on two quantities: *accuracy-after-m*, the prediction accuracy on the open dimensions $P_{t,\bar{S}_{t,m}}$ as a function of the m corrections received; and *corrections-to-stop* m_t^*, the number of corrections needed before the prediction matches P_t^* on every open dimension. Summed over the deployment, $\sum_t m_t^*$ is the total correction workload in Section 8.5.

E.2.5 Correlation Evidence

The evidence behind the correlated-bundle hypothesis is in Fig. E.1: which dimensions propagated to which (a), that the dependence exceeds chance (b), and the per-meal correction counts (c) underlying the cumulative comparison in Fig. 8.6. All three panels cover the same 21 categorical and free-text dimensions; the nine environment parameters (plate colors and parking offsets) are excluded as perceptual and geometric values, not meal preferences.

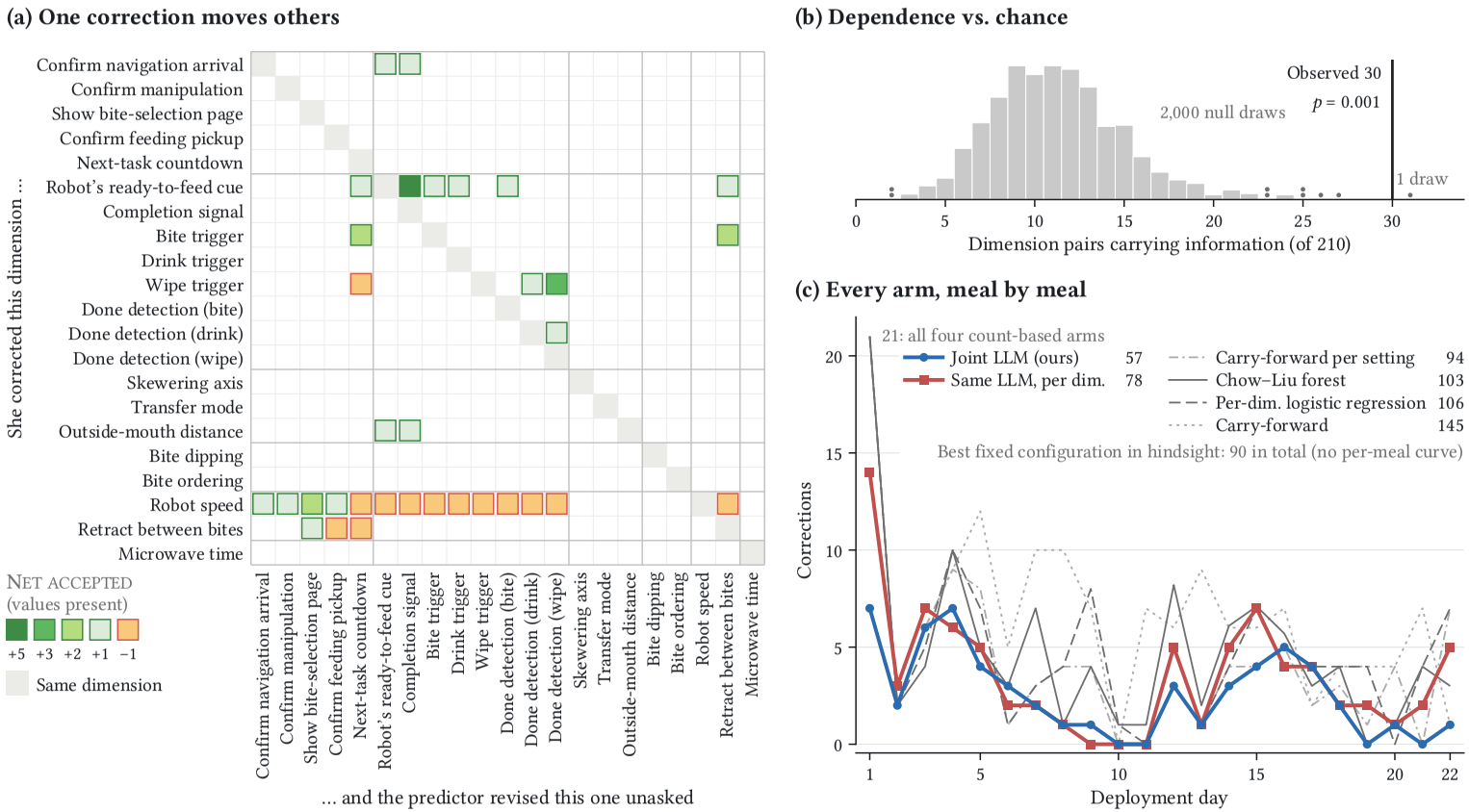


Figure E.1: **The CR's preferences are correlated, and exploiting them lowers her correction workload.** **(a)** Rows are dimensions she corrected, columns those the predictor then revised unasked, both ordered and grouped as in Table E.3; shade is accepted revisions minus overwrites of already-correct values, green where propagation net helped, amber where it hurt. **(b)** Count of informative dimension pairs under 2,000 circular shifts (gray) against the observed count. **(c)** Each arm's corrections at every meal, underlying the cumulative totals in Fig. 8.6; the key gives each arm's deployment total. *Same LLM, per dim.* predicts each dimension on its own, as CBTL [248] does; *Carry-forward per setting* is FEAST's rule [122] (Appendix E.8).

Panel (a) reads the correlation that the predictor actually used directly from the logs: a row is a dimension the CR corrected, either on its preference page or mid-meal on the settings page, and a column is a dimension the predictor then revised on its own, without being asked about it. Each cell is outlined in its arm's darkest step, which is what separates the two pale tints from one another. The effect is real but uneven: correcting the interaction cue for one task carried most reliably to the same cue for the others (*Robot's ready-to-feed cue* → *Completion signal*, five accepted revisions and none wrong), whereas a single mid-meal speed edit on meal 4 fanned out to 12 dimensions and mostly overwrote correct values (the *Robot speed* row).

Test of correlation. Panel (b) tests whether the dimensions carry information about one another in the values she accepted. Six of the 21 meal preferences held the same value at every meal (Table E.3), leaving 15 that varied and $15 \times 14 = 210$ ordered pairs. For each ordered pair, we predict the second dimension from the first by a lookup table fitted on every other meal, leaving out the meal being predicted, and compare its accuracy with that of the second dimension's own most common value, fitted the same way. In 30 of the 210 pairs, the first dimension predicts the second better. The null circularly shifts each dimension's 22-meal history by an independent random offset, which preserves how sticky each preference is over the deployment and destroys only the alignment between dimensions; over 2,000 shifts it yields 11.25 such pairs on average, and one shift reaches 30 or more ($p = 0.001$).

Propagation accounting. Every re-prediction over the deployment is logged with the values it changed. Scoring each changed meal preference against the value she held at the end of that meal, the predictor revised 62 values without

being asked: 37 revisions turned a wrong value right, 22 turned a right value wrong, and 3 replaced one wrong value with another, a net gain of 15. They came from 23 re-predictions, which changed 2.7 values on average; the largest, the mid-meal speed edit of meal 4, changed 12, most of them values that were already correct.

Amplification. The amplification reported in Section 8.5 is computed over the corrections she made on the preference pages, where each correction triggers a re-prediction of the dimensions still open: it is the number of dimensions, meal preferences and environment parameters alike, that these re-predictions changed, divided by the number of such corrections. Over meals 1 to 6, 15 corrections changed 21 dimensions (1.4); over meals 12 to 18, 19 changed 24 (1.3); over meals 19 to 22, 2 changed 7 (3.5). The one such correction of meals 7 to 11 changed none.

In panel (c), corrections are counted as individual events, so the panel agrees meal by meal with the timeline table in Fig. 8.6; the only difference is that the plate-color channel plotted there has no counterpart here. Meal 1 costs every count-based arm all 21 dimensions, having no history to predict from, whereas both LLM arms begin from a physically grounded prior; this is what sets the panel's vertical range.

E.3 Preparation Phase

We spent five weeks, from June 11 to July 16, 2026, developing and testing the system in the CR's home before the first meal of the study. Table E.4 gives a day-by-day account, reconstructed from the dated records the work left: the system's

code history, the maps built of her home, the logs of every run of the robot, and the notes we kept as we worked, and from a team member's journal and calendar of the period. We worked in her home on 29 of the 36 days; on the other seven we traveled, or she was unavailable and we worked elsewhere.

Table E.4: **The preparation phase, day by day.** Every date from June 11 to July 16, 2026, on which the records show work on the system or a change of plan; the team worked in her home on 29 of the 36 days. *Stand-in* meals were eaten by research team members, not the CR.

Date	Work
Week 1	**The robot in her kitchen**
Jun 11	Travel to her town; no work on the robot
Jun 12	Her home unavailable: design of the study and of the interface for taking over a manipulation skill
Jun 13	First day in her home: skirts on both tables, a cover on the refrigerator door, and tape marking each parking location on the floor; opening her refrigerator; teleoperation of the arm and base from the web interface as a fallback for failed skills
Jun 14	Microwave manipulation; detection confirmation pages in the interface
Jun 15	Microwave manipulation and the web interface; testing ended early because she had a commitment
Jun 16	The new microwave-safe plate attachment found to fit its holder poorly; remote emergency stop built
Jun 17	Microwave manipulation; the holder's mating part sanded until the attachment docked reliably
Week 2	**Every appliance skill and the first complete run**
Jun 18	The refrigerator, microwave, and sink skills work in her kitchen; the microwave-button model served remotely; taking over extended from arm to base; the wired button at her iPad
Jun 19	Her home unavailable: design of learning from her corrections
Jun 20	The first map of her home; refrigerator manipulation completed and table manipulation begun; navigation's goal-reaching refined and its arrival tolerance logged
Jun 21	First run of the complete pipeline, from the refrigerator to the sink, over nine hours; the second map; the color-correction page; resuming an interrupted navigation
Jun 22–23	Navigation refined; the base's controller moved to the arm's computer; the remote emergency stop put into use
Jun 24	Logging of every meal event; staged preference pages; preference prediction moved to Claude

Date	Work
Week 3	**Stand-in meals**
Jun 25	An integrated run from the refrigerator to the table; the feeding utensil's holder broke and was reprinted
Jun 26	Stand-in meals, begun the day before; emergency-stop dropouts traced to the wireless channel and fixed; resuming a skill after teleoperation
Jun 27	Third map; collision thresholds set per skill
Jun 28–30	Stand-in meals on most days; sensor diagnostics; the door of her microwave opened wider
Jul 1	Fourth map; parking offsets learned from her adjustments on arrival
Week 4	**Navigation**
Jul 2	Navigation logging; the visual-inertial odometry found to diverge while the robot's base was parked
Jul 3	Fifth map; prediction of all personalization dimensions; the confirmation pages and their countdowns designed
Jul 6	Localization found to flip by 180° at the table, whose view is nearly symmetric; the plate attachment's color changed from teal to orange
Jul 7	Wheel encoders read back to the robot's computer
Jul 8	Wheel odometry calibrated on the floor of her home; in the evening, a test of the complete pipeline
Jul 9	Visual-inertial odometry found to drift in direct sunlight with no fault reported; wheel commands made reliable; base speeds calibrated
Jul 10	Base speeds lowered; fusion of wheel odometry and the IMU tested alongside the camera; the team returned home for the weekend
Week 5	**Wheel-and-IMU odometry, the trial meal**
Jul 13	RealSense firmware downgraded to stabilize its streams; the switch to wheel-and-IMU odometry, validated in a run in which the camera's estimate drifted 5.9 m while localization held; sixth map
Jul 14	**Trial meal with the CR**; a fallback when wheel slip turns the base during a scripted drive
Jul 15	Localization protected from scans occluded by the arm and people at the table; the camera's visual odometry turned off after its software crashed during a run; the seventh and eighth maps, of which the eighth was used from meal 1
Jul 16	End-of-meal survey; a full stand-in rehearsal of the meal

From skills to a complete meal. The first two weeks brought the manipulation skills to her appliances and the interface around them: the refrigerator, microwave, and sink skills worked in her kitchen by June 18, the first map of her home was built on June 20, and the first run of the complete pipeline, from refrigerator to sink, followed on June 21. The hardware needed changes of its own:

the microwave-safe plate attachment did not dock in its holder until we sanded the holder's mating part (June 17), and the feeding utensil's holder broke during an integrated run and was reprinted (June 25). From June 25, members of the research team ate stand-in meals with the robot most days, which exercised the pipeline end to end and surfaced failures before the CR used it.

Navigation. Most of the remaining time went to navigation, for which her home leaves little margin. We built eight maps as we refined the system, and three failures shaped the configuration that ran during the study. First, the table's outline, as the lidars see it, is nearly symmetric, so localization could flip by 180° beside it (July 6). Second, the robot's odometry initially came from the ZED Mini's visual-inertial odometry, which failed in the home in three ways: it drifted under direct sunlight on the lens while reporting no fault (July 9), crept slowly off course by several meters, and its software crashed four times. We added encoder read-back to the wheels (July 7), calibrated wheel odometry on her floor (July 8), and on July 13 switched to an extended Kalman filter fusing the wheel encoders with the yaw rate of the ZED's IMU; in the run that validated it, the camera's own estimate drifted 5.9 m while localization was unaffected. On July 15, after the camera's software crashed during a run, we turned its visual odometry off and kept only its IMU. Third, while the base was parked at the table on July 14, the arm and the people around it occluded the lidars, and localization jumped by meters; from July 15 the system discards scans too sparse to localize against.

Additions to the home. On our first day in her home, June 13, we made three small additions to her home. We put a floor-length skirt on each table, gingham on the dining table and a plain one on the movable table, so that the lidars see

a surface rather than the table's legs. We put a cover on the refrigerator door, so that glare from sunlight would not corrupt the depth readings the robot uses to find its handle. And we marked each parking location on the floor with tape, as the reference against which we measured how precisely the base parked.

Planning with the CR. We planned the study together with her: the three settings in which she expected to use the robot (eating on her own or while watching television at the movable table, and eating with a friend at the dining table) and the schedule of the meals, set around her availability, which moved the start of the study by about a week.

The trial meal. The CR's only exposure to the complete system before the study was a single trial meal on July 14, three days before meal 1. The robot ran the whole pipeline, from the refrigerator to the sink. The trial did not seed her preferences: it was logged separately from the study, and the predictor's history at meal 1 was empty, so its first predictions drew only on her physical functioning and the meal's context. The one value carried into the study was the plate attachment's color, calibrated during a test that day, from which all plate-color dimensions began.

E.4 Researcher Notes and Interventions

Settings. The CR ate in six settings over the month (Table E.5). We planned three with her before the study: eating while watching television and eating on her own, both at the movable table, and eating with a friend at the dining table. She proposed the other three as she grew comfortable with the robot: music at a personal meal from meal 12, video calls with her daughter at the dining table

Table E.5: **The six settings she ate in.** *Label* is the setting she chose on the meal-context page, which the predictor conditions on; the page's labels were reduced to these three at meal 16. The dining table serves social meals and the movable table all others. Swatches are the setting colors of Fig. 8.6.

Setting	Label	Meals
Planned before the study (3)		
Watching TV	TV	1, 2, 5, 8, 11, 14, 19
Alone	Personal	3, 6, 15
With a friend	Social	4, 7, 9, 10, 13, 20
Proposed by her (3)		
Music on	Personal	12
Video call	Social	16, 17, 22
iPad games	Personal	18, 21

from meal 16, and games on her iPad from meal 18. She also ate her first morning meal with the robot at meal 15.

The corpus. The attending researcher kept timestamped notes during each meal in the system's logging interface. They cover all 22 meals; the notes of meals 5 and 20 were written into the file of the preceding meal, and are counted with the meal at which they were taken. By episode, the corpus holds 100 notes and 27 explanations, of which 91 and 20 carry free text. The researcher also marked intervals as interventions during the meal; these marks were one source for the count below, not the count itself. The researcher sometimes began the notes late, as at meals 12 and 16 ("I started taking notes late"), so the counts are totals over the month, not a per-meal series. The CR's name has been replaced with "the CR" throughout.

Interventions. We identified the 32 interventions by combining the robot's execution log, the researcher's in-meal notes, and the video recordings of the

meals, counting an episode when the run had to be stopped, or the robot repaired, by someone other than the CR, and attributing each to a cause from the same sources. Tables E.7 and E.8 list them by meal. By cause, 11 were hardware (gripper fingertips working loose or breaking on meals 4, 9, and 10; cables caught on meals 7, 9, 12, and 15; the depth sensor failing on meal 11; the battery running out on meal 17; the transfer button registering only intermittently on meal 20), 17 software (navigation on meals 7, 10, 13, 20, and 21; food detection on meals 15, 19, and 22; plate detection on meals 19 and 21, the second in a room too dark for it; a code change that broke bite transfer on meal 12; unexpected arm motion on meals 4 and 16; an arm fault that left a service unavailable on meal 17; and single faults in collision checking, mouth-open detection, and the network), and four interface. By skill, bite acquisition and navigation account for six each, bite transfer for five, and the refrigerator for four. Three of the interface interventions share one cause, an interface bug: a takeover the CR pressed during bite transfer (meals 9, 14, and 17) was not handed to her, and the system stopped responding until the researcher restarted it. This was fixed after meal 17.

Several kinds of episode are excluded from the count. The CR's own takeovers are low-level teleoperation and are reported with her corrections in Section 8.4.2. Nine caregiver tasks she delegated to the researcher, as she would to a human caregiver, are not system failures: reheating food three times, cutting steak, a towel for dripping, removing two pieces of pineapple too hard to chew, moving a food item the plate hid, and moving the plate past a joint limit. Three episodes began with an input error rather than a system fault: the CR choosing *done* instead of *resume* after a failed handle detection (meal 7), and the emergency stop struck by mistake (meals 14 and 16). Twice the researcher returned the iPad to the task-selection page while the robot kept running (meals 15 and 18).

Explanations. 27 episodes record the CR asking the research team something rather than the system: 23 in the first 11 meals, 4 in the last 11, none after meal 15. Four concern navigation or repositioning the base, two of them how to resume after it.

Corrections known to be input errors. At least four corrections in the ledger of Appendix E.11 are logged as misclicks the CR then reverted or continued with: a plate color on meal 3, a setting of a confirmation page on meal 5, an unspecified correction on meal 6, and a base-position confirmation on meal 8. They are counted in the totals of Section 8.5, the plate color among the 19 plate-color corrections and the rest among the 57 preference corrections.

What she learned each day. Each end-of-meal survey closed with the question "What, if anything, did you learn about the robot today, or how did your interaction with it change?" Table E.6 reproduces every answer as she gave it, by dictation or typing on her iPad, with capitalization, spacing, and final punctuation corrected; she left it blank at six meals.

Table E.6: **What she learned, meal by meal.** Her answer to the question that closed every end-of-meal survey, "What, if anything, did you learn about the robot today, or how did your interaction with it change?" Answers are verbatim but for capitalization, spacing and final punctuation; a bracketed word replaces one misrecognized by dictation. Quotations in Chapter 8 are lightly edited, with her permission. She left the question blank at meals 10, 11 and 14–17.

Meal	Answer
1	I learned that I can change any setting at any time and on any screen.
2	I am starting to trust the robot's movements more.
3	Today I learned how to control the robots arm myself and restart a skill in case of an error.
4	Today I used the button to cue the robot to bring the food to my mouth. I liked this option for the social setting because I didn't feel rushed and could carry on a conversation.

Meal	Answer
5	I noticed the robot predicted that the strawberries would not be microwaved. It predicted that I like 30 seconds in between bites.
6	I had to control the robot base today so it would not crash! Once I corrected the path of the robot, it did a complete 360 to reacquaint itself with its surroundings.
7	The robot is very sensitive to lights and shadows. I need to be aware if I need to use the continued option or a resume option after moving the robot arm.
8	I learned that the robot may be very sensitive to my wheelchair moving around it. Also, just a slight move of the base or arm can help complete an action.
9	I need to remember to look straight at the robots arm after I press the button for it to advance to my mouth.
10–11	*left blank*
12	I learned that opening my mouth is not my preferred gesture for face wiping. I chose which bite to take because the robot could not distinguish between the two different types of food.
13	I need to remember to look at the robot hand after pressing the button for it to bring the bite to my mouth.
14–17	*left blank*
18	Split screen works very well for playing games. Sometimes, especially on the screen used to move the robot arm, some options are not showing. I had to use the robot's voice to [tell] me to open my mouth because I am too busy looking at the screen. When the movable table was accidentally hit, the robot joints hit a limit when it tried to pick the plate up so now I will be extra careful when moving around the movable table.
19	As the lighting outside changed, the robot struggled identifying the pieces of food on my plate. I was also more careful about not hitting the table, but when I did I was sure to bring attention to it so it could be fixed.
20	Had trouble with the button signaling to give me a bite. Thinking it is a hardware problem. Once I did not hit the button but it moved towards me. Figured out it was due to the iPad microphone being on and a plastic container loudly opening.
21	As I ate, the lighting changed so at the end of my meal when the robot went to pick up my plate it could not distinguish the orange from the rest of the background due to it being too dark in the room. I had to ask for assistance turning on the lights.
22	I found a way to talk and not trigger the robot to feed me a bite when seeing my "open mouth."

Table E.7: **The researcher interventions over meals 1–13.** *Category* is the cause. *Skill* is the high-level action affected, from the robot's log. Each description combines the robot's log, the researcher's note, and the video recording of the meal. Meals not listed had no intervention.

Meal	Category	Skill	What happened
3	Software	`open_fridge`	A network error stopped the system before the refrigerator opened; the researcher restarted it and reran the skill.
4	Hardware	`open_fridge`	A gripper fingertip worked loose; after the CR's three retries, the researcher fixed it during her fourth takeover.
	Hardware	`pick_utensil`	The fingertip fix did not hold and the utensil could not be picked up; the researcher fitted the spare fingertip.
	Software	`acquire_bite`	The arm began an unexpected trajectory during bite acquisition; the researcher stopped and relaunched the system.
7	Hardware	`pick_plate_from_microwave`	A cable that had not been fully unfurled was pulled; the researcher stopped the run and freed it.
	Software	`navigate_to_dining_table`	Navigation raised transform-extrapolation errors; the researcher restarted it.
	Software	`acquire_bite`	On the CR's retry, a collision threshold set too tight stopped the software; the researcher restarted it.
9	Hardware	`open_fridge`	A fingertip broke; the researcher stopped the system and replaced it.
	Hardware	`pick_utensil`	A cable caught on the mouth wiper; the researcher stopped the system and freed it.
	Interface	`transfer_utensil`	Due to an interface bug, the robot ignored the CR's takeover during transfer and stopped responding to the iPad; the researcher restarted it.
10	Hardware	`open_fridge`	A fingertip worked loose again; the researcher tightened it during the CR's takeover.
	Software	`navigate_to_sink`	Navigation failed after the CR had moved; the researcher reset it and relaunched.
11	Hardware	`navigate_to_microwave`	The depth sensor failed; the researcher recovered it and relaunched.
12	Software	`transfer_utensil`	A code change made earlier that day broke bite transfer; the researcher fixed it and relaunched.
	Hardware	`pick_wipe`	A cable caught and the collision sensor tripped; the researcher stopped the system and freed it.

Meal	Category	Skill	What happened
13	Software	`navigate_to_dining_table`	Navigation failed twice in succession; the researcher reset it and relaunched.

Table E.8: **The researcher interventions over meals 14–22.** Columns as in Table E.7.

Meal	Category	Skill	What happened
14	Interface	`transfer_utensil`	Due to an interface bug, the robot ignored the CR's takeover during transfer of a bite she had let cool, and stopped responding to the iPad; the researcher restarted it.
15	Software	`acquire_bite`	The food-detection prompt did not detect the pancake; the researcher stopped the system and revised it.
	Hardware	`stow_utensil`	A cable lay in the stowing path; the arm struck it, triggering the emergency stop, and the researcher moved it.
16	Software	`transfer_drink`	The drink moved down and to her right, not toward her mouth; after the emergency stop, the researcher reset it.
17	Software	`open_microwave`	The arm jerked and faulted while opening the microwave, and a service it needed became unavailable; the researcher reset the system.
	Interface	`transfer_utensil`	The CR missed a bite and took over to retry the transfer; due to an interface bug, the robot ignored the takeover and stopped responding to the iPad; the researcher restarted it.
	Hardware	`acquire_bite`	The battery ran out and the arm dropped; the researcher recovered it.
18	Software	`transfer_drink`	Mouth-open detection misfired; the CR redid the sip, but detection then stopped responding and the researcher restarted the system.
	Interface	`pick_plate_from_movable_table`	The iPad showed no plate detection and the table had moved; the researcher relaunched and reset the setup.
19	Software	`acquire_bite`	Food detection missed the chicken, labeling it pineapple; the researcher restarted with a revised prompt.
	Software	`pick_plate_from_movable_table`	Plate detection failed repeatedly despite the CR's corrections; the researcher stopped the run and relaunched.
20	Hardware	`transfer_utensil`	The transfer button registered presses only intermittently; the researcher reconnected it.

Meal	Category	Skill	What happened
	Software	`navigate_to_sink`	Localization jumped while parked, leaving no feasible path; the researcher stopped the run and relaunched.
21	Software	`pick_plate_from_movable_table`	The room was too dark for plate detection; the researcher relaunched and, at the CR's request, turned on the lights.
	Software	`navigate_to_sink`	Leaving the table, the base reversed and turned; the researcher stopped it and relaunched.
22	Software	`acquire_bite`	The food-detection prompt read the whole sandwich as one region, so the plate appeared empty; the researcher revised it.

E.5 Changes to the System During Deployment

As in prior in-home deployments [198], we did not freeze the system: we changed it between meals when a failure revealed the need, and Table E.9 lists every change with the first meal it applied to. Because changes were often committed in batches after a meal, we date each one from the code that actually ran: at every start of the robot's software, the system recorded the version of the code and every edit not yet committed, which gives the exact code that ran at each of the 22 meals. We checked the dates against the robot's behavior-tree parameters and preference dimensions logged at each meal and against the researcher's notes. Changes to analysis and study tooling that did not affect the robot are excluded.

Three changes are the ones Section 8.4.2 refers to. The drinking skill became functional at meal 12, when the mug moved to a fixed holder on the base from which the robot picks and stows it; none of the drinking skills ran at meals 1 to 11, and each ran at every meal from 12 to 22. The navigation fix applied from meal 14: the gyroscope's messages carried timestamps that the odometry filter

could reject, in which case it estimated the base's rotation from the wheels alone, which under-report rotation. Which messages it rejected varied from one start of the software to the next, so the failure appeared at some meals and not others; it spoiled the base's parking at meals 8 and 13. Re-stamping each message with the time it is received removed it. The gripper's 3D-printed fingertips were repaired during meals 4, 9, and 10 without a restart of the meal.

The remaining changes fall into four kinds. Many addressed a failure that the previous meal exposed, such as the transfer target raised after the fork stopped below her mouth at meal 3, or the collision threshold raised after a false stop at meal 7. A second kind followed her use of the robot: the dining and movable tables became separate locations at meal 4, her first social meal at the dining table, and the button moved to the robot's computer after it misfired at meal 20. A third kind recalibrated the poses of her furniture and her appliances. The fourth kind, before most meals, added object-detection prompts for the foods of that meal; when a prompt failed during a meal, we corrected it and restarted, which happened at meals 15, 19, and 22.

Table E.9: **Every change to the system during the deployment.** Changes to the software on the robot's main computer and to the hardware, each listed at the first meal it applied to; the software changes run in meal order, followed by the recurring prompt change and then the hardware changes. A change made *during* a meal took effect at that meal; every other change was made between meals.

Meal	Component	Change
2	Manipulation	Arm speed presets retuned so joint and Cartesian motions match at each speed; Cartesian trajectories slow to a stop at their last waypoint
	Manipulation	Outside-mouth distances shortened to 5, 7, and 10 cm for near, medium, and far
	Manipulation	Bite-acquisition history kept per meal rather than carried across meals

Meal	Component	Change
	Safety	Speed held at *slow* while the arm picks, stows, or carries a tool or the plate, whatever her speed preference
	Personalization	The predictor also estimates her transient state and receives its own earlier predictions of the meal (Appendix E.2.3); it retries when a response lacks the full bundle
	Interface	The settings overlay lists dimensions in the order they are asked
4	Navigation	The dining table and the movable table become separate locations, chosen from the setting, each with its own parking offset; a new map of the home with waypoints for the dining table and the kitchen exit
	Navigation	The map correction frozen while the base is parked, to keep occluded scans from moving it (removed before meal 5)
	Perception	Transfer waits for a valid head-pose reading instead of failing when none arrives
	Manipulation	Outside-mouth transfer target raised by 3 cm, after the transfer at meal 3 stopped below her mouth
	Safety	The takeover button is honored during bite acquisition
	Interface	The language descriptions of the robot's actions generated by Claude Opus 4.8 in place of Claude Haiku 4.5
5	Navigation	The route out of the kitchen depends on where the base leaves from; waypoints at the microwave, sink, and movable table re-recorded
	Perception	Plate color and plate height kept separate for the dining and the movable table
	Perception	Head-pose readings accepted up to 1.5 m instead of 1 m; a 1 s pause after picking a tool
	Manipulation	Gripper opening and closing setpoints recalibrated, after a fingertip broke at meal 4
	Manipulation	Dipping: sauces detected off the plate, dip depth measured with a fallback to the plate's height
	Interface	Manual keypoint mode taps points on the full camera image and shows the chosen dip
7	Manipulation	Dips spread across the sauce; skewer target offset 5 mm along the item's major axis; the pose above the plate recalibrated
8	Safety	Default collision threshold, used wherever a skill sets none of its own, raised from 15 to 20 Nm, after a false stop when she retried a bite acquisition at meal 7
9	Perception	Skewer depth measured over a wider patch, with a fallback to the plate's height; plate-attachment detections larger than the attachment rejected
	Manipulation	The gripper closes before pushing the microwave door
	Interface	The button can also be read on the robot's computer, as a fallback to the iPad
	Interface	Continuous dictation on every text field

Meal	Component	Change
11	Study procedure	The researcher is prompted to position the recording camera before each navigation and plate pickup
12	Perception	Head-nod detection for finishing a sip or a wipe made more robust
	Manipulation	Drinking made functional: the mug is picked from and stowed in a fixed holder on the base, with its transfer distance recalibrated
	Interface	*During the meal:* a task-selection bug, introduced that morning, fixed
13	Manipulation	Plate-holder pick and place poses recalibrated
14	Navigation	The IMU's messages re-stamped with the time each is received; before this, the odometry filter could discard the gyroscope and estimate rotation from the wheels alone, spoiling parking
15	Perception	*During the meal:* the object-detection prompts for two food items corrected
16	Personalization	*During the meal:* setting labels reduced to personal, social, and television, dropping which side a companion sat on; her stored history relabeled before meal 17
17	Perception	The pose from which the plate is imaged recalibrated
18	Perception	Object-detection prompts tuned automatically from the plate image
	Safety	A takeover can end the wait for a transfer trigger
19	Perception	*During the meal:* the object-detection prompts for two food items corrected
20	Perception	The robot waits 10 s instead of 5 s for the camera to settle before imaging the plate
21	Perception	Automatic prompt tuning turned off in favor of fixed prompts for each meal's foods
	Interface	The button is read only on the robot's computer, after it misfired at meal 20
22	Perception	*During the meal:* an object-detection prompt added for a sandwich
2–22	Perception	Before most meals, object-detection prompts and meal labels added for that meal's foods
4, 9, 10	Hardware	*During the meal:* a gripper fingertip repaired (meal 4), replaced after breaking (meal 9) and retightened (meal 10)
12	Hardware	A fixed holder for the mug added to the base

E.6 Meals of the Deployment

Figure E.2 shows every meal of the deployment as served, one panel per meal. The CR chose each meal's food, so the set was not curated for the robot: it spans

dry and crumbly items (hash brown, pancake) through sauced and slippery ones (orange chicken) to soft fruit, and varies in piece size, count, and spacing. It is the range of foods the acquisition stack had to handle in the wild, and is referred to from Section 8.4.

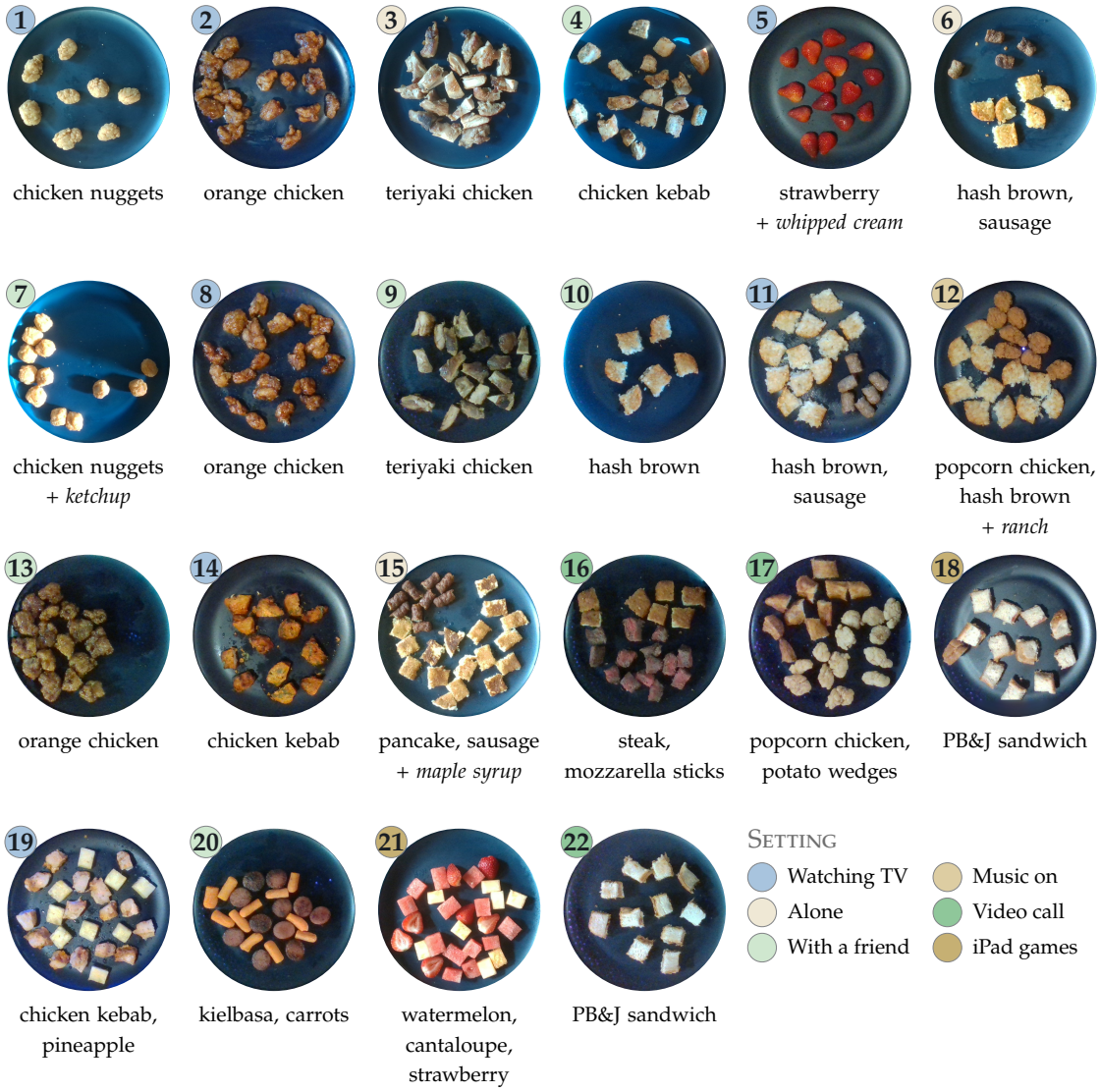


Figure E.2: All 22 meals of the deployment, in order, each plate as served before the first bite and cropped to its rim. Meals were chosen by the CR and vary in piece size, count, and spacing, from dry and crumbly (hash brown, pancake) to sauced and slippery (orange chicken); dips are named in italics. Each plate's disc carries the meal number and is filled with the color of that meal's setting (Table E.5; left panel of Fig. 8.6). Six crops contain a small region of synthesized bare plate where the plate ran past the camera frame or was occluded by the fork; food is never altered.

E.7 Interface Screenshots

Figures E.3 and E.4 show the pages of the web interface described in Appendix E.1, as recorded on the CR's iPad.

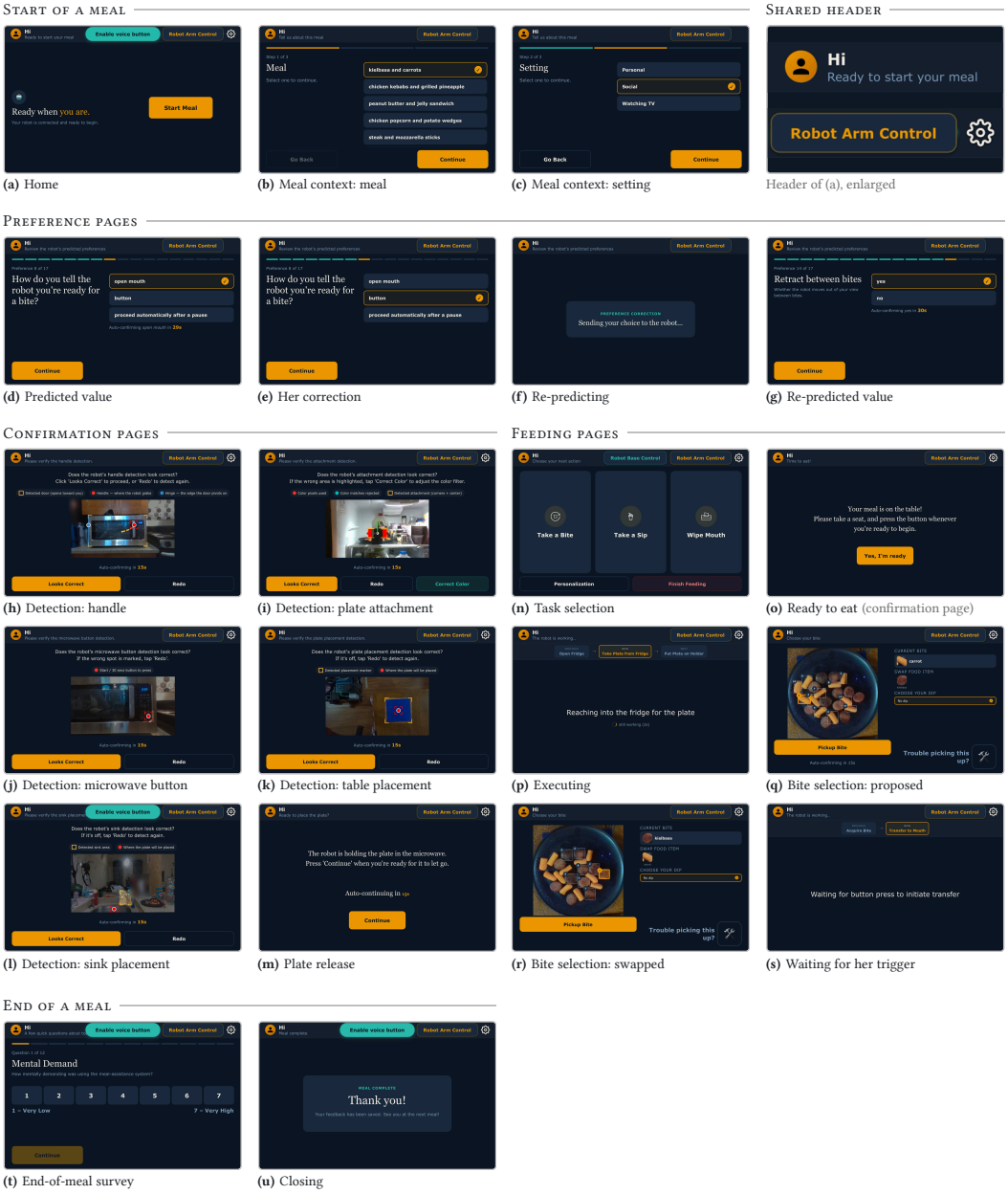
E.8 Baselines

Every alternative in Fig. 8.6 predicts meal t from meals 1 to $t-1$ and is scored under the replay protocol of Appendix E.2.4. We chose one representative of each family a designer could have deployed before LLM-based prediction, and one oracle as a ceiling.

Same LLM, per dimension. The same model and reasoning budget as the joint predictor (Claude Opus 4.8, medium reasoning), but one call per open dimension, each shown only that dimension's own history of (day, meal, setting, time, value) and her physical-functioning description. It never sees another dimension's value or her corrections that meal, and is not re-run after a correction, so its errors are its corrections. This is the structure of CBTL [248], which learns each parameter independently; against it the joint predictor isolates the value of correlation alone.

Carry-forward. For every dimension, repeats her last accepted value: the intuitive strawman, and the model an "auto-continue" setting implements. It costs 145.

Carry-forward per setting. Repeats the last value she accepted in the same setting (personal, social, or television), falling back to plain carry-forward until a setting has been seen. This is what FEAST [122] does: it stores one bundle per



(a) Home (b) Meal context: meal (c) Meal context: setting Header of (a), enlarged

(d) Predicted value (e) Her correction (f) Re-predicting (g) Re-predicted value

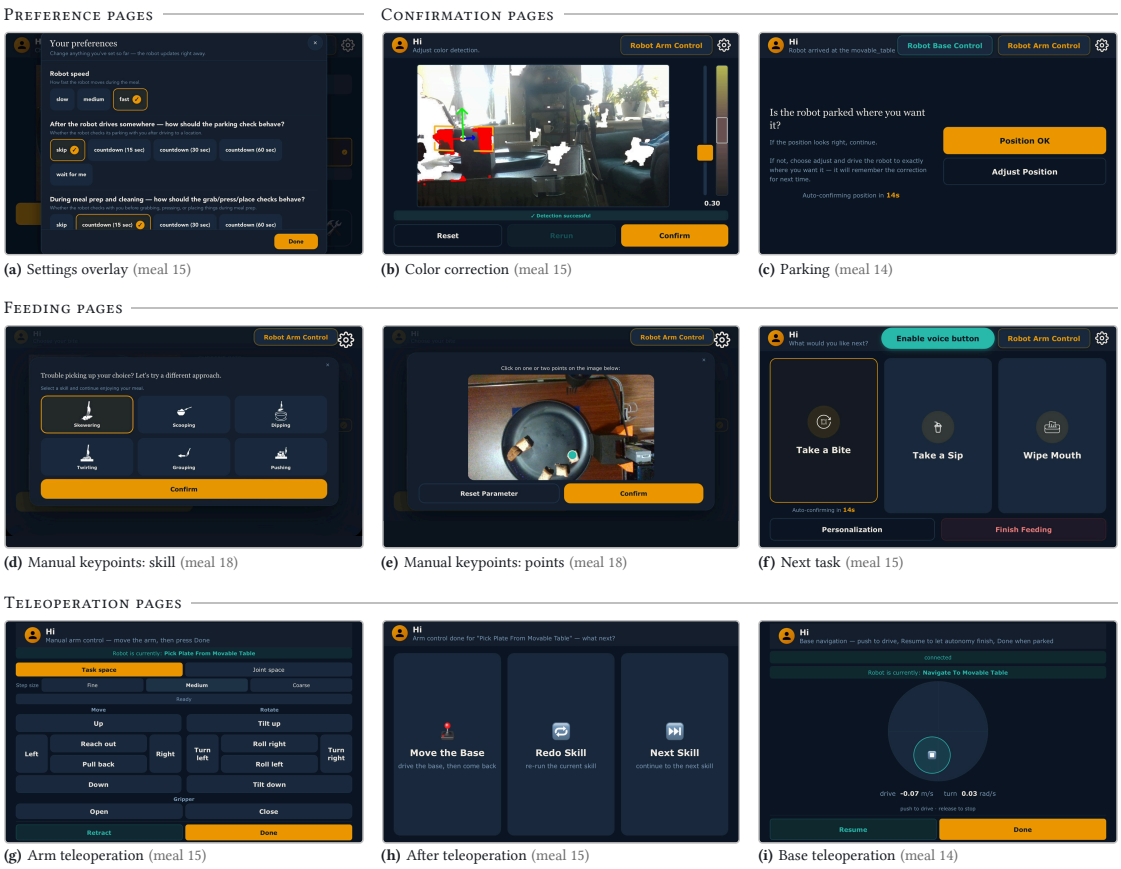
(h) Detection: handle (i) Detection: plate attachment (n) Task selection (o) Ready to eat (confirmation page)

(j) Detection: microwave button (k) Detection: table placement (p) Executing (q) Bite selection: proposed

(l) Detection: sink placement (m) Plate release (r) Bite selection: swapped (s) Waiting for her trigger

(t) End-of-meal survey (u) Closing

Figure E.3: **The web interface at meal 20**, as recorded on the CR's iPad. The header, enlarged at top right, carries the status line and the *Robot Arm Control* button on every page, and the settings gear on most; her name is erased from it. (d–g) show one correction and its re-prediction: the robot predicted open mouth as her bite trigger (d), she chose the button (e), and the robot re-predicted the open dimensions (f), revising retraction between bites (g) and the next-task countdown, to waiting for her. (h–l) are the five detections the detection page confirms. (q, r) show her swapping the proposed bite to another food item.



(a) Settings overlay (meal 15) **(b)** Color correction (meal 15) **(c)** Parking (meal 14)

(d) Manual keypoints: skill (meal 18) **(e)** Manual keypoints: points (meal 18) **(f)** Next task (meal 15)

(g) Arm teleoperation (meal 15) **(h)** After teleoperation (meal 15) **(i)** Base teleoperation (meal 14)

Figure E.4: **Pages that meal 20 did not reach**, from the CR's iPad recordings of meals 14, 15, and 18. In (b) she has tapped the plate attachment; the pixels matching its color are shown in red and the slider sets the matched range. (d, e) show the manual keypoint mode, opened from *Trouble picking this up?* on the bite-selection page: she chooses a skill and taps a point on the food. In (i) she drives the base while the robot navigates to the movable table. Background photographs in (b) are blurred. The pages are shown at a larger scale than in Fig. E.3.

context and reloads it. It is the strongest count-based arm at 94, which shows that most of plain carry-forward's cost is dragging one setting's bundle into another, not drift.

Per-dimension logistic regression. A multinomial softmax per dimension over one-hot setting, time of day, and meal name, fitted by gradient descent (400

steps, learning rate 0.5, L_2 weight 0.1) on meals 1 to $t-1$. With no history, a dimension is counted wrong; with a single observed value that value is predicted. This is the full-information reduction of the contextual-bandit view of preference learning. Adding the meal name never helps (106 to 116 across encodings); it costs 106.

Chow–Liu forest. The textbook joint categorical model for small n [40]. Pairwise mutual information is computed from counts with add-0.5 smoothing over meals 1 to $t-1$; an edge is kept if $2n$ MI exceeds the BIC penalty $\ln(n)\,(|A|-1)(|B|-1)$, and Kruskal's algorithm on the kept edges gives a forest. Prediction clamps the dimensions corrected so far and runs exact sum-product in each component, taking the argmax belief; a revealed value never seen before falls back to the marginal. With fewer than four meals of history, it predicts the per-dimension mode. Reveal order is random; the total, 103, averages 10 orders.

Best fixed configuration in hindsight. The single constant bundle minimizing total corrections over all 22 meals, which is the per-dimension mode, chosen with knowledge of the whole deployment. It costs 90 and is a ceiling on any non-adaptive predictor, not a baseline. Guessing uniformly from each dimension's option list would cost roughly 324 corrections.

Two scoring rules favor the count-based arms. Her three same-meal revisions (meals 3, 5, and 6), in which she changed a dimension she had already settled, are charged to both LLM arms and not to the count-based ones. And a count-based arm cannot produce a value it has never observed; 44 of the 462 scored slots hold one. The joint predictor faces the same cold start on meal 1, where a count-based method has no history and misses all 21 dimensions, and costs 7 corrections there, the hindsight bundle's cost that day.

E.9 Preference Bundle by Meal

Where Table E.11 records the individual overrides, Table E.10 records the result: the complete preference bundle the system held at the end of every meal, one column per meal and one row per dimension. Each column is headed by that meal's setting and plate, so the configuration can be read against the context that produced it. Cells the CR corrected during the meal are shaded and bold; the rest are values she accepted as predicted.

Read across a row, the table separates dimensions she never revisits from ones that move. Six are constant over the whole deployment: *Skewering axis*, *Transfer mode*, *Outside-mouth distance*, *Confirm feeding pickup*, *Robot speed*, and *Show bite-selection page*. The dimensions that do move cluster rather than drift independently. The done-detection dimensions settle over the first week. The drink and wipe triggers shift once those two tasks enter regular use from meal 12, and shift again in the social meals of meals 16 and 17 — the *Drink trigger* had been button in every social meal and open mouth in every TV or personal meal up to meal 16, and meal 17 is the first social meal where she asks for open mouth instead. The *Next-task countdown* moves the same way, from *wait for me* in the five earlier social meals to a 60-second countdown on meals 16 and 17.

E.10 Teleoperation Accounting

Base-driving episodes are recovered from `/cmd_vel_teleop` in the deployment rosbags, the only record that survives every route to the base controls, including a detour out of an arm session to reposition an already-parked robot. We separate the researcher's Xbox controller from the CR's web interface by publisher

Table E.10: **The preference bundle the system held at the end of every meal.** One column per meal, headed by the plate as served and that meal's setting; one row per preference dimension. **Shaded bold** cells are dimensions the CR corrected during that meal; every other cell is a value she accepted as predicted, in black where it changed from the meal before (and at meal 1) and in gray where it did not. The continuous dimensions (plate-attachment color at each pickup location, and the per-location parking offsets) and the free-text bite-ordering dimension are omitted. Setting colors are those of the left panel of Fig. 8.6 and of Fig. E.2; abbreviations are keyed below the matrix.

Plate	chicken nuggets	orange chicken	teriyaki chicken	chicken kebab	strawberry *+ whipped cream*	hash brown, sausage	chicken nuggets *+ ketchup*	orange chicken	teriyaki chicken	hash brown	hash brown, sausage	popcorn chicken, hash brown *+ ranch*	orange chicken	chicken kebab	pancake, sausage *+ maple syrup*	steak, mozzarella sticks	chicken popcorn, potato wedges	PB&J sandwich	chicken kebab, pineapple	kielbasa, carrots	watermelon, cantaloupe, strawberry	PB&J sandwich
Setting	TV	TV	PERSONAL	SOCIAL	TV	PERSONAL	SOCIAL	TV	SOCIAL	SOCIAL	TV	PERSONAL	SOCIAL	TV	PERSONAL	SOCIAL	SOCIAL	PERSONAL	TV	SOCIAL	PERSONAL	SOCIAL
Meal	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
Checks with her																						
Confirm navigation arrival	15 s	15 s	**skip**	15 s	**skip**	skip	15 s	**skip**	**skip**	skip	skip	skip	skip	**15 s**	skip	skip	skip	skip	skip	skip	skip	skip
Confirm manipulation	15 s	15 s	**15 s**	15 s	**15 s**	15 s	15 s	15 s	15 s	15 s	15 s	15 s	**skip**	15 s	15 s	15 s	15 s	15 s	15 s	15 s	15 s	15 s
Show bite-selection page	15 s	15 s	15 s	15 s	15 s	15 s	15 s	15 s	15 s	15 s	15 s	15 s	15 s	15 s	15 s	15 s	15 s	15 s	15 s	15 s	15 s	15 s
Confirm feeding pickup	**skip**	skip	skip	**skip**	skip	skip	skip	skip	skip	skip	skip	skip	skip	skip	skip	skip	skip	skip	skip	skip	skip	skip
Next-task countdown	**30 s**	30 s	**15 s**	**wait**	30 s	15 s	wait	30 s	wait	wait	30 s	15 s	wait	30 s	15 s	**60 s**	**60 s**	15 s	30 s	wait	15 s	30 s
Signals																						
Robot's ready-to-feed cue	SP	SP	**LED**	**LED**	**LED**	LED	LED	LED	LED	LED	LED	**SP**	LED	LED	**SP**	LED	LED	SP	LED	LED	SP	**SP**
Completion signal	SP	SP	LED	LED	LED	LED	LED	LED	LED	LED	LED	LED	LED	LED	SP	LED	LED	**SP**	LED	LED	SP	SP
Bite trigger	OM	OM	OM	**BT**	OM	**OM**	**AP**	OM	BT	BT	OM	OM	BT	OM	OM	**OM**	**OM**	OM	OM	**BT**	OM	OM
Drink trigger	OM	OM	OM	BT	OM	OM	BT	OM	BT	BT	OM	OM	BT	OM	OM	**BT**	**OM**	OM	OM	BT	OM	OM
Wipe trigger	OM	OM	OM	BT	OM	OM	BT	OM	BT	BT	OM	OM	BT	**BT**	**BT**	BT	**BT**	**BT**	BT	BT	BT	BT
Done detection (bite)	**AP**	**PC**	**AP**	**PC**	AP	**PC**	PC	PC	PC	PC	PC	PC	PC	PC	PC	PC	PC	PC	PC	PC	PC	PC
Done detection (drink)	PC	PC	PC	BT	PC	PC	BT	PC	BT	BT	PC	**BT**	BT	**BT**	**BT**	BT	BT	BT	BT	BT	BT	BT
Done detection (wipe)	PC	**AP**	PC	BT	PC	PC	BT	PC	BT	BT	PC	BT	BT	BT	BT	BT	BT	BT	BT	BT	BT	BT
Fit to her abilities																						
Skewering axis	⊥	⊥	⊥	⊥	⊥	⊥	⊥	⊥	⊥	⊥	⊥	⊥	⊥	⊥	⊥	⊥	⊥	⊥	⊥	⊥	⊥	⊥
Transfer mode	out	out	out	out	out	out	out	out	out	out	out	out	out	out	out	out	out	out	out	out	out	out
Outside-mouth distance	**near**	near	near	near	near	near	near	near	near	near	near	near	near	near	near	near	near	near	near	near	near	near
Bite																						
Bite dipping	none	none	none	none	more	none	**more**	none	none	none	none	more	none	none	**less**	none	none	none	none	none	none	none
Motion																						
Robot speed	**fast**	fast	fast	**fast**	fast	fast	fast	fast	fast	fast	fast	fast	fast	fast	fast	fast	fast	fast	fast	fast	fast	fast
Retract between bites	**no**	no	no	**yes**	no	no	yes	no	yes	yes	no	no	yes	no	no	**no**	no	no	no	yes	no	no
Heating																						
Microwave time	**30 s**	30 s	30 s	30 s	off	30 s	30 s	30 s	30 s	30 s	30 s	30 s	30 s	30 s	30 s	**off**	30 s	off	30 s	30 s	off	off

Cells: **BT** corrected during that meal; BT accepted, changed from the meal before; BT accepted, same as the meal before

Setting: Watching TV; Alone; With a friend; Music on; Video call; iPad games

Key: 15 s, 30 s, 60 s countdown (confirm and autocontinue); off no microwave; ⊥ perpendicular to major axis; out outside-mouth transfer; SP speech; OM open mouth; BT button; AP proceed automatically after a pause; PC perception; wait wait for me

identity, and exclude it: it is used to place the base at setup and teardown, and accounts for more base driving than the CR does. Of the 66 logged arm sessions, 51 command no motion at all — the CR uses the teleoperation screen as an escape hatch to skip past a skill, which on meal 3 she described as only wanting "to skip driving" — and a further 8 jog the arm without the arm being stuck: pressing arm-control is also how she stops the robot to reach the *Move the Base* button that follows *Done*, how a researcher pauses it to adjust hardware, and once how she nudged an idle arm between bites. We count the remaining 7, all rescues of an arm that had hit a joint limit. Bag coverage of the meal windows is 92–100% per day. Both series are shown in the left panel of Fig. 8.6.

E.11 Correction Ledger

Table E.11 lists every correction the CR made over the deployment, in the order she made them. Corrections reach the system through two channels. Each predicted dimension appears on its own preference page where it takes effect, and she overrides the ones she disagrees with; these appear as *Ask* rows. During the meal, she can open the settings page and change a dimension directly; these appear as *Set* rows. Plate colors she re-picked on the color-correction page appear as *Col* rows; they are the 19 plate-color corrections of Section 8.5, counted apart from the 57 preference corrections. All three are logged with the value the system held immediately before the override, so each row records what was predicted and what she replaced it with.

Table E.11: **Every correction the CR made, by meal.** *Ask* corrections are overrides on a dimension's preference page, where it takes effect; *Set* corrections are mid-meal changes from the settings page; *Col* corrections are plate-attachment colors she re-picked on the color-correction page. *Predicted* is the value the system held for that dimension immediately before the override. A dot marks each row's channel and a swatch each logged plate color (key below the table).

Meal	Dimension	Predicted	Corrected to
1	• Robot speed	medium	fast
	• Plate color (fridge)	12, 223, 169 (0.10)	9, 189, 189 (0.10)
	• Microwave time	1 min	30 s
	• Plate color (microwave)	12, 223, 169 (0.10)	10, 201, 240 (0.24)
	• Outside-mouth distance	medium	near
	• Retract between bites	yes	no
	• Confirm feeding pickup	countdown 15 s	skip
	• Next-task countdown	countdown 15 s	countdown 30 s
	• Done detection (bite)	perception	proceed automatically after a pause
	• Plate color (table)	10, 195, 195 (0.12)	3, 99, 129 (0.12)
2	• Plate color (fridge)	9, 189, 189 (0.10)	8, 160, 147 (0.19)
	• Plate color (microwave)	9, 175, 200 (0.24)	12, 248, 186 (0.24)
	• Done detection (wipe)	perception	proceed automatically after a pause
	• Done detection (bite)	proceed automatically after a pause	perception
	• Plate color (table)	5, 120, 140 (0.15)	7, 172, 175 (0.24)
3	• Confirm manipulation	countdown 15 s	skip
	• Confirm manipulation	skip	countdown 15 s
	• Confirm navigation arrival	countdown 15 s	skip
	• Robot's ready-to-feed cue	speech	LED
	• Done detection (bite)	perception	proceed automatically after a pause
	• Next-task countdown	countdown 30 s	countdown 15 s
	• Plate color (table)	5, 135, 150 (0.18)	7, 146, 155 (0.18)

Meal	Dimension	Predicted	Corrected to
4	Robot speed	medium	fast
	Robot's ready-to-feed cue	speech	LED
	Bite trigger	open mouth	button
	Retract between bites	no	yes
	Confirm feeding pickup	countdown 15 s	skip
	Next-task countdown	countdown 30 s	wait for me
	Done detection (bite)	proceed automatically after a pause	perception
5	Confirm navigation arrival	countdown 15 s	skip
	Plate color (fridge)	8, 175, 168 (0.15)	10, 200, 129 (0.15)
	Robot's ready-to-feed cue	speech	LED
	Confirm manipulation	countdown 15 s	skip
	Confirm manipulation	skip	countdown 15 s
	Plate color (movable table)	7, 150, 158 (0.18)	11, 191, 254 (0.18)
6	Done detection (bite)	proceed automatically after a pause	perception
	Bite trigger	open mouth	button
	Bite trigger	button	open mouth
	Plate color (movable table)	8, 170, 190 (0.18)	3, 89, 97 (0.17)
7	Bite trigger	button	proceed automatically after a pause
	Bite dipping	less	more
	Plate color (dining table)	7, 146, 155 (0.18)	10, 180, 207 (0.18)
8	Confirm navigation arrival	countdown 15 s	skip
9	Confirm navigation arrival	countdown 15 s	skip
	Plate color (dining table)	10, 180, 207 (0.18)	13, 206, 161 (0.18)
10	Plate color (dining table)	13, 206, 161 (0.18)	16, 136, 253 (0.15)
11	Plate color (fridge)	8, 170, 160 (0.17)	10, 207, 123 (0.17)
12	Bite ordering	*free text*[a]	*free text*[a]
	Done detection (drink)	perception	button
	Robot's ready-to-feed cue	LED	speech
	Plate color (movable table)	7, 172, 175 (0.20)	13, 216, 255 (0.20)
13	Confirm manipulation	countdown 15 s	skip
14	Confirm navigation arrival	skip	countdown 15 s
	Wipe trigger	open mouth	button
	Done detection (drink)	perception	button
15	Robot's ready-to-feed cue	LED	speech
	Wipe trigger	open mouth	button
	Done detection (drink)	perception	button
	Bite dipping	more	less
	Plate color (movable table)	7, 172, 175 (0.20)	4, 52, 118 (0.30)

Meal	Dimension	Predicted	Corrected to
16	• Microwave time	30 s	no microwave
	• Bite trigger	button	open mouth
	• Drink trigger	open mouth	button
	• Retract between bites	yes	no
	• Next-task countdown	countdown 30 s	countdown 60 s
	• Plate color (dining table)	13, 180, 205 (0.18)	11, 200, 140 (0.18)
17	• Bite trigger	button	open mouth
	• Drink trigger	button	open mouth
	• Wipe trigger	open mouth	button
	• Next-task countdown	countdown 30 s	countdown 60 s
18	• Completion signal	LED	speech
	• Wipe trigger	open mouth	button
	• Plate color (movable table)	4, 52, 118 (0.30)	13, 219, 238 (0.30)
19	*no corrections*		
20	• Bite trigger	open mouth	button
21	• Plate color (movable table)	7, 172, 175 (0.20)	10, 181, 192 (0.09)
22	• Robot's ready-to-feed cue	LED	speech

[a] Meal 12, bite ordering. Predicted: "Dip each chicken popcorn and hash brown in ranch dressing before feeding." Corrected to: "I would like to eat each chicken popcorn dipped in ranch dressing. I would like to eat …"

• Ask (pre-meal selection) • Set (mid-meal update) • Col (plate color) Plate colors: swatch, then hue 0–179, saturation and value 0–255 (matching range).

E.12 Measures

Daily. After every meal, the CR rated 11 items on the iPad, one per screen, on a 7-point scale anchored at *Very Low* (1) and *Very High* (7); each is shaded in the bottom-left panel of Fig. 8.6 so that green is the favorable end. Five are NASA-TLX workload items [93] and six concern her experience of the robot; Table E.12 gives the wording of each. One open question followed: "What, if anything, did you learn about the robot today, or how did your interaction with it change?" Her answers are reproduced in full in Table E.6 and quoted in Chapter 8.

End of study. After the final meal she completed the following, each on a 5-point agreement scale from *strongly disagree* (1) to *strongly agree* (5): the five items of the *Technology Acceptance Model* [53]; the 10 standard items of the *System Usability Scale* [25]; two items on *Control and Independence*, each administered twice, once for the robot and once for her human caregiver, to enable direct comparison; and two items on *Co-Adaptation*. Table E.12 gives the wording of all but the System Usability Scale items.

Table E.12: **The survey items.** Items are given as the CR saw them on her iPad, one per screen. The daily items were rated on a 7-point scale from *Very Low* (1) to *Very High* (7), and the end-of-study items on a 5-point agreement scale from *strongly disagree* (1) to *strongly agree* (5). The 10 standard items of the System Usability Scale are not repeated here.

Item	Question
After every meal: NASA-TLX workload (5)	
Mental demand	How mentally demanding was using the meal-assistance system?
Physical demand	How physically demanding was using the meal-assistance system?
Temporal demand	How much time pressure did you feel when using the meal-assistance system?
Effort	How hard did you have to work to use the meal-assistance system effectively?
Frustration	How frustrated were you while using the meal-assistance system?
After every meal: her experience of the robot (6)	
Performance	How successful were you in using the meal-assistance system to achieve your goals?
Trust	How much did you trust the robot to do the right thing during today's meal?
Safety	How safe did you feel during today's meal?
Predictability	How well could you predict what the robot was going to do during today's meal?
Robot adaptation	How much do you feel the robot has learned about your preferences?
Satisfaction	Overall, how satisfied were you with today's meal?
End of study: Technology Acceptance Model (5)	
Perceived usefulness	Using this meal-assistance system will make me more independent in eating.
Perceived ease of use	This meal-assistance system is easy to use.

Item	Question
Attitude toward using	Using the meal-assistance system for improving my independence is a good idea.
Behavioral intention to use	Assuming I have access to this meal-assistance system, I predict that I would use it in my daily life.
Perceived enjoyment	I find using this meal-assistance system to be enjoyable.
End of study: Control and Independence, each asked for her caregiver and for the robot (2)	
Control	I feel in control of my feeding experience when assisted by my caregiver / the robot.
Independence	I feel a sense of independence when I receive assistance from my caregiver / the robot.
End of study: Co-Adaptation (2)	
Robot's side	The robot got better at understanding my preferences over the course of the study.
Her side	I got better at working with the robot over the course of the study.

E.13 Human Adaptation: Protocol and Robustness

Recovering her triggering aperture. Across the 15 meals with mouth-open triggering active, we re-ran the deployed detector offline over the logged eye-in-hand video and recovered the peak mouth aperture ratio (MAR) at which each of 261 bite requests fired; it recovers an opening for 259 of them. The per-meal medians sit at two levels. Neither block trends internally (meals 1 to 15, $\rho = -0.32$, $p = 0.41$; meals 16 to 22, $\rho = +0.09$, $p = 0.87$), and they separate perfectly at the transition: the lowest median before it (0.674) exceeds the highest after it (0.669). Sweeping every candidate split and maximizing the between-group sum of squares locates the changepoint at meal 16, and a permutation test on that maximum, which corrects for searching all splits, rejects the no-change null ($p = 7.5{\times}10^{-4}$). The fitted levels are 0.796 and 0.592 MAR (95% CIs 0.727–0.864 and 0.536–0.648), and only 3% of the 103 post-transition bites exceed the pre-transition median.

Labeling meals 22 and 9. The trigger log cannot say whether the lower level still separates a request from the rest of a meal, because defining a request as whatever fired censors the distribution at 0.45 by construction. An author therefore labeled every second of meal 22's 14 detector-armed waits as talking, chewing, smiling, requesting a bite, or at rest, working from the eye-in-hand video; consecutive request seconds are merged, since a request is one continuous opening. Across 110 seconds of everyday behavior, not one reaches 0.45; the closest approach is a single talking second at 0.425. Bite requests sit at a median of 0.502, and 13 of 16 clear the threshold (Mann–Whitney $p = 6.4\times10^{-11}$). She is not opening wide to be safe: her requests clear the threshold by a median of 0.05 MAR, the widest aperture she produces anywhere in the meal is 0.99, and her median aperture while actually taking a bite is 0.756. Meal 9, a social meal in which she used the button, was labeled the same way as a contrast in which the detector was never armed: 79 seconds of talking with a median of 0.348 and a maximum of 0.620, of which 12 (15%) cross 0.45, against none of the 31 talking seconds in meal 22 (Fisher $p = 0.018$; Mann–Whitney on peak MAR $p = 2.3\times10^{-6}$).

Three caveats. The labeler saw the aperture trace alongside the video, though not the 0.45 line, so the labels are not blind to the signal; that they cut across the threshold in both directions — three requests below it (0.352, 0.429, 0.437) and three non-requests above it, a whistle at 0.464, a laugh at 0.507 and a head-pose artifact at 0.528 — indicates they track intent rather than magnitude. In meal 9, 27 of 106 talking seconds carry a note that she had turned toward the researcher or downward; we exclude them, because the inter-ocular distance confirms the turn (76 vs 107 px, $p = 2.7\times10^{-11}$) and MAR is unreliable off-axis. Including them would raise the crossing rate to 30%. Finally, this is one meal per condition, and they differ in more than the detector: meal 9 is early and

has a friend present, meal 22 is late and on a video call. A second button meal (meal 20) shows the same shift in level but only 1 of 47 talking seconds above threshold, so the crossing rate is not a stable property of the unarmed condition.

Time to take the bite. Across 273 bites in the 20 meals with usable timing — five run past the 3 s panel axis in Fig. 8.9 and are drawn at its cap, the longest at 5.6 s — the same changepoint procedure places a step before meal 4 ($p = 2.1\times10^{-3}$). The per-meal medians separate perfectly there (the lowest before it, 1.66 s; the highest after it, 0.99 s); the fitted levels fall from 1.92 s to 0.56 s, a factor of 3.4, and the later block does not trend internally ($\rho = -0.02$, $p = 0.95$). Read instead as a gradual decline across the meals, the effect is not significant ($\rho = -0.40$, $p = 0.084$ over the 20 meal medians); the data support an abrupt drop within the first week followed by 16 flat meals. The step survives aggregation to the meal level (Mann–Whitney on per-meal medians, $p = 0.002$), so it is not an artifact of treating the 273 bites as statistically independent observations.

E.14 Attention Measurement

We recover head pose per frame with a face-mesh regressor and `solvePnP`, and report the angle between her head direction and the fork; the eye-in-hand camera sits directly behind the fork, so a small angle means her head is aimed at the robot. The measure covers 357 transfers and 1.45×10^{5} frames; meal 14 has no bite-transfer recording and is absent from the per-meal panel. Two windows are compared: the bite request itself (±1 s) and the rest of the transfer (>3 s from the request).

Away from the request, her head turns further from the robot as the deployment proceeds ($\rho = +0.74$, $p = 0.011$); the permutation test is stratified by activity so that the late clustering of video-call meals cannot produce the effect. At the request, it does not ($\rho = +0.24$, $p = 0.28$). Across activity types, head direction away from the request varies strongly (Kruskal–Wallis $p = 0.006$, from 14.4° on meals with nothing else to 28.8° on video calls) while at the request it does not ($p = 0.46$, every meal within 10.7–19.6°). The activities she chose grew more demanding over the month ($\rho = +0.53$, $p = 0.011$ against an ordering assigned from the labels alone; the five most demanding meals all fall in the last third), and her attention tracks what each one asks of her ($\rho = +0.79$, $p = 1.7\times10^{-5}$). The direction of causation is not established by a single-participant observational design.

BIBLIOGRAPHY

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. GPT-4 Technical Report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] Melanie M Adams and Audrey L Hicks. Spasticity after Spinal Cord Injury. *Spinal Cord*, 43(10):577–586, 2005.

[3] Christopher Agia, Toki Migimatsu, Jiajun Wu, and Jeannette Bohg. TAPS: Task-Agnostic Policy Sequencing. *arXiv preprint arXiv:2210.12250*, 2022.

[4] Negin Amirshirzad, Asiye Kumru, and Erhan Oztop. Human Adaptation to Human-Robot Shared Control. *IEEE Transactions on Human-Machine Systems*, 49(2):126–136, 2019. doi: 10.1109/THMS.2018.2884719.

[5] Arduino. Arduino Uno. `https://store.arduino.cc/products/arduino-uno-rev3`, 2024.

[6] Montserrat Gonzalez Arenas, Ted Xiao, Sumeet Singh, Vidhi Jain, Allen Ren, Quan Vuong, Jake Varley, Alexander Herzog, Isabel Leal, Sean Kirmani, et al. How to Prompt Your Robot: A Promptbook for Manipulation Skills with Code as Policies. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4340–4348. IEEE, 2024.

[7] Ashay Athalye, Nishanth Kumar, Tom Silver, Yichao Liang, Tomás Lozano-Pérez, and Leslie Pack Kaelbling. Predicate Invention from Pixels via Pretrained Vision-Language Models. *arXiv preprint arXiv:2501.00296*, 2024.

[8] ATI. ATI F/T Sensor, 2024. URL `https://www.ati-ia.com/products/ft/sensors.aspx`. (Accessed: 1st January, 2024).

[9] Markus Bajones, David Fischinger, Astrid Weiss, Daniel Wolf, Markus Vincze, Paloma de la Puente, Tobias Körtner, Markus Weninger, Konstantinos Papoutsakis, Damien Michel, Ammar Qammaz, Paschalis Panteleris, Michalis Foukarakis, Ilia Adami, Danai Ioannidi, Asterios Leonidis, Margherita Antona, Antonis Argyros, Peter Mayer, Paul Panek, Håkan Eftring, and Susanne Frennert. Hobbit: Providing Fall Detection and Prevention for the Elderly in the Real World. *Journal of Robotics*, 2018:1754657, 2018. doi: 10.1155/2018/1754657.

[10] Markus Bajones, David Fischinger, Astrid Weiss, Paloma De La Puente, Daniel Wolf, Markus Vincze, Tobias Körtner, Markus Weninger, Konstantinos Papoutsakis, Damien Michel, et al. Results of Field Trials with a Mobile Service Robot for Older Adults in 16 Private Households. *ACM Transactions on Human-Robot Interaction*, 9(2):1–27, 2019.

[11] MP Barnes, RM Kent, JK Semlyen, and KM McMullen. Spasticity in Multiple Sclerosis. *Neurorehabilitation and Neural Repair*, 17(1):66–70, 2003.

[12] Maude Beaudoin, Josiane Lettre, François Routhier, Philippe S. Archambault, Martin Lemay, and Isabelle Gélinas. Long-Term Use of the JACO Robotic Arm: A Case Series. *Disability and Rehabilitation: Assistive Technology*, 14(3):267–275, 2019. doi: 10.1080/17483107.2018.1428692. Published online 2018.

[13] Suneel Belkhale, Ethan K Gordon, Yuxiao Chen, Siddhartha Srinivasa, Tapomayukh Bhattacharjee, and Dorsa Sadigh. Balancing Efficiency and Comfort in Robot-Assisted Bite Transfer. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 4757–4763. IEEE, 2022.

[14] Suneel Belkhale, Tianli Ding, Ted Xiao, Pierre Sermanet, Quon Vuong, Jonathan Tompson, Yevgen Chebotar, Debidatta Dwibedi, and Dorsa Sadigh. RT-H: Action Hierarchies Using Language. *arXiv preprint arXiv:2403.01823*, 2024.

[15] Dmitry Berenson, Siddhartha Srinivasa, and James Kuffner. Task Space Regions: A Framework for Pose-Constrained Manipulation Planning. *The International Journal of Robotics Research*, 2011.

[16] Amisha Bhaskar, Rui Liu, Vishnu D Sharma, Guangyao Shi, and Pratap Tokekar. LAVA: Long-horizon Visual Action based Food Acquisition. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8929–8935, 2024. doi: 10.1109/IROS58592.2024.10802738.

[17] Tapomayukh Bhattacharjee, Maria E. Cabrera, Anat Caspi, Maya Cakmak, and Siddhartha S. Srinivasa. A Community-Centered Design Framework for Robot-Assisted Feeding Systems. In *Proceedings of the 21st International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS '19, pages 482–494, New York, NY, USA, 2019. Association for Computing Machinery. doi: 10.1145/3308561.3353803.

[18] Tapomayukh Bhattacharjee, Ethan K Gordon, Rosario Scalise, Maria E Cabrera, Anat Caspi, Maya Cakmak, and Siddhartha S Srinivasa. Is More Autonomy Always Better? Exploring Preferences of Users with Mobility

Impairments in Robot-Assisted Feeding. In *2020 15th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 181–190. IEEE, 2020.

[19] Erdem Bıyık, Dylan P Losey, Malayandi Palan, Nicholas C Landolfi, Gleb Shevchuk, and Dorsa Sadigh. Learning Reward Functions from Diverse Sources of Human Feedback: Optimally Integrating Demonstrations and Preferences. *The International Journal of Robotics Research*, 41(1):45–67, 2022.

[20] James Booth, Anastasios Roussos, Stefanos Zafeiriou, Allan Ponniah, and David Dunaway. A 3D Morphable Model Learnt from 10,000 Faces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5543–5552, 2016.

[21] Jake Brawer, Debasmita Ghose, Kate Candon, Meiying Qin, Alessandro Roncone, Marynel Vázquez, and Brian Scassellati. Interactive Policy Shaping for Human-Robot Collaboration with Transparent Matrix Overlays. In *Proceedings of the 2023 ACM/IEEE International Conference on Human-Robot Interaction*, pages 525–533, 2023.

[22] Heike Brock and Randy Gomez. Personalization of Human-Robot Gestural Communication through Voice Interaction Grounding. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 846–853. IEEE, 2021.

[23] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. RT-1: Robotics Transformer for Real-World Control at Scale. *arXiv preprint arXiv:2212.06817*, 2022.

[24] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. RT-2: Vision-Language-Action Models Transfer Web Knowledge to Robotic Control. *arXiv preprint arXiv:2307.15818*, 2023.

[25] John Brooke. SUS: A Quick and Dirty Usability Scale. In Patrick W. Jordan, Bruce Thomas, Bernard A. Weerdmeester, and Ian L. McClelland, editors, *Usability Evaluation in Industry*, pages 189–194. Taylor & Francis, London, 1996.

[26] Steven W Brose, Douglas J Weber, Ben A Salatin, Garret G Grindle, Hongwu Wang, Juan J Vazquez, and Rory A Cooper. The Role of Assistive Robotics in the Lives of Persons with Disability. *American Journal of*

Physical Medicine & Rehabilitation, 89(6):509–521, 2010.

[27] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language Models Are Few-Shot Learners. *Advances in Neural Information Processing Systems*, 33:1877–1901, 2020.

[28] Alvaro Parra Bustos and Tat-Jun Chin. Guaranteed Outlier Removal for Point Cloud Registration with Correspondences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2868–2882, 2017.

[29] Gerard Canal, Guillem Alenyà, and Carme Torras. Personalization Framework for Adaptive Robotic Feeding Assistance. In *Social Robotics: 8th International Conference, ICSR 2016, Kansas City, MO, USA, November 1–3, 2016*, pages 22–31. Springer, 2016.

[30] Gerard Canal, Guillem Alenyà, and Carme Torras. A Taxonomy of Preferences for Physically Assistive Robots. *IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, 2017.

[31] Gerard Canal, Guillem Alenyà, and Carme Torras. Adapting Robot Task Planning to User Preferences: An Assistive Shoe Dressing Example. *Autonomous Robots*, 2019.

[32] Gerard Canal, Carme Torras, and Guillem Alenyà. Are Preferences Useful for Better Assistance? A Physically Assistive Robotics User Study. *ACM Transactions on Human-Robot Interaction*, 2021.

[33] Alexandre Candeias, Travers Rhodes, Manuel Marques, Manuela Veloso, et al. Vision Augmented Robot Feeding. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, 2018.

[34] Qiong Cao, Li Shen, Weidi Xie, Omkar M Parkhi, and Andrew Zisserman. VGGFace2: A Dataset for Recognising Faces across Pose and Age. In *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*, pages 67–74. IEEE, 2018.

[35] Jin Chen, Zheng Liu, Xu Huang, Chenwang Wu, Qi Liu, Gangwei Jiang, Yuanhao Pu, Yuxuan Lei, Xiaolong Chen, Xingmei Wang, et al. When Large Language Models Meet Personalization: Perspectives of Challenges and Opportunities. *World Wide Web*, 27(4):42, 2024.

[36] Tiffany L Chen, Matei Ciocarlie, Steve Cousins, Phillip Grice, Kelsey Hawkins, Kaijen Hsiao, Charles C Kemp, Chih-Hung King, Daniel A Lazewatsky, Adam Leeper, et al. Robots for Humanity: User-Centered Design

for Assistive Mobile Manipulation. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5434–5435. IEEE, 2012.

[37] Tiffany L Chen, Matei Ciocarlie, Steve Cousins, Phillip M Grice, Kelsey Hawkins, Kaijen Hsiao, Charles C Kemp, Chih-Hung King, Daniel A Lazewatsky, Adam E Leeper, et al. Robots for Humanity: Using Assistive Robotics to Empower People with Disabilities. *IEEE Robotics & Automation Magazine*, 20(1):30–39, 2013.

[38] Yongchao Chen, Jacob Arkin, Yang Zhang, Nicholas Roy, and Chuchu Fan. AutoTAMP: Autoregressive Task and Motion Planning with LLMs as Translators and Checkers. *arXiv preprint arXiv:2306.06531*, 2023.

[39] Adriano Chiò, A Gauthier, A Vignola, Andrea Calvo, Paolo Ghiglione, Enrico Cavallo, AA Terreni, and Roberto Mutani. Caregiver Time Use in ALS. *Neurology*, 67(5):902–904, 2006.

[40] C. K. Chow and C. N. Liu. Approximating Discrete Probability Distributions with Dependence Trees. *IEEE Transactions on Information Theory*, 14 (3):462–467, 1968.

[41] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. PaLM: Scaling Language Modeling with Pathways. *Journal of Machine Learning Research*, 24(240):1–113, 2023.

[42] J Chung, A Nagrani, and A Zisserman. VoxCeleb2: Deep Speaker Recognition. In *Interspeech*. International Speech Communication Association, 2018.

[43] Nikhil Churamani, Sinan Kalkan, and Hatice Gunes. Continual Learning for Affective Robotics: Why, What and How? In *2020 29th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, pages 425–431. IEEE, 2020.

[44] Nancy Ciesla, Victor Dinglas, Eddy Fan, Michelle Kho, Jill Kuramoto, and Dale Needham. Manual Muscle Testing: A Method of Measuring Extremity Muscle Strength Applied to Critically Ill Patients. *Journal of Visualized Experiments (JoVE)*, 2011.

[45] Giovanni Cioffi, Silke Klose, and Arne Wahrburg. Data-Efficient Online Classification of Human-Robot Contact Situations. In *2020 European Control Conference (ECC)*, pages 608–614. IEEE, 2020.

[46] Caitlyn Clabaugh, Kartik Mahajan, Shomik Jain, Roxanna Pakkar, David Becerra, Zhonghao Shi, Eric Deng, Rhianna Lee, Gisele Ragusa, and Maja Matarić. Long-Term Personalization of an In-Home Socially Assistive Robot for Children With Autism Spectrum Disorders. *Frontiers in Robotics and AI*, 6:110, 2019. doi: 10.3389/frobt.2019.00110.

[47] Michele Colledanchise and Petter Ögren. *Behavior Trees in Robotics and AI: An Introduction*. CRC Press, 2018.

[48] Corinna Cortes and Vladimir Vapnik. Support-Vector Networks. *Machine Learning*, 20(3):273–297, 1995.

[49] Erwin Coumans and Yunfei Bai. PyBullet, a Python Module for Physics Simulation for Games, Robotics and Machine Learning, 2016. URL `http://pybullet.org`. (Accessed: 22nd April, 2025).

[50] Christopher Crick, Graylin Jay, Sarah Osentoski, Benjamin Pitzer, and Odest Chadwicke Jenkins. Rosbridge: ROS for Non-ROS Users. In *Robotics Research: The 15th International Symposium ISRR*, pages 493–504. Springer, 2017.

[51] Aidan Curtis, Nishanth Kumar, Jing Cao, Tomás Lozano-Pérez, and Leslie Pack Kaelbling. Trust the PRoC3S: Solving Long-Horizon Robotics Problems with LLMs and Constraint Satisfaction. In *Conference on Robot Learning (CoRL)*, 2024.

[52] Hang Dai, Nick Pears, William AP Smith, and Christian Duncan. A 3D Morphable Model of Craniofacial Shape and Texture Variation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3085–3093, 2017.

[53] Fred D Davis, Richard P Bagozzi, and Paul R Warshaw. User Acceptance of Computer Technology: A Comparison of Two Theoretical Models. *Management Science*, 35(8):982–1003, 1989.

[54] Matt Deitke, Christopher Clark, Sangho Lee, Rohun Tripathi, Yue Yang, Jae Sung Park, Mohammadreza Salehi, Niklas Muennighoff, Kyle Lo, Luca Soldaini, et al. Molmo and PixMo: Open Weights and Open Data for State-of-the-Art Multimodal Models. *arXiv preprint arXiv:2409.17146*, 2024.

[55] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[56] Shehzaad Dhuliawala, Mojtaba Komeili, Jing Xu, Roberta Raileanu, Xian Li, Asli Celikyilmaz, and Jason Weston. Chain-of-Verification Reduces Hallucination in Large Language Models. *arXiv preprint arXiv:2309.11495*, 2023.

[57] Laura E Dreer, Timothy R Elliott, Richard Shewchuk, Jack W Berry, and Patricia Rivera. Family Caregivers of Persons with Spinal Cord Injury: Predicting Caregivers at Risk for Probable Depression. *Rehabilitation Psychology*, 52(3):351, 2007.

[58] Aaron D'Souza, Sethu Vijayakumar, and Stefan Schaal. Learning Inverse Kinematics. In *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 1, pages 298–303. IEEE, 2001.

[59] Zackory Erickson, Vamsee Gangaram, Ariel Kapusta, C Karen Liu, and Charles C Kemp. Assistive Gym: A Physics Simulation Framework for Assistive Robotics. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

[60] Nima Fazeli, Miquel Oller, Jiajun Wu, Zheng Wu, Joshua B Tenenbaum, and Alberto Rodriguez. See, Feel, Act: Hierarchical Learning for Complex Manipulation Skills with Multisensory Fusion. *Science Robotics*, 4(26): eaav3123, 2019.

[61] Ryan Feng, Youngsun Kim, Gilwoo Lee, Ethan K Gordon, Matt Schmittle, Shivaum Kumar, Tapomayukh Bhattacharjee, and Siddhartha S Srinivasa. Robot-Assisted Feeding: Generalizing Skewering Strategies Across Food Items on a Plate. In *The International Symposium of Robotics Research*, pages 427–442. Springer, 2019.

[62] Yao Feng, Haiwen Feng, Michael J Black, and Timo Bolkart. Learning an Animatable Detailed 3D Face Model from In-the-Wild Images. *ACM Transactions on Graphics*, pages 1–13, 2021.

[63] Julia Fink, Valérie Bauwens, Frédéric Kaplan, and Pierre Dillenbourg. Living with a Vacuum Cleaning Robot: A 6-month Ethnographic Study. *International Journal of Social Robotics*, 5(3):389–408, 2013. doi: 10.1007/s12369-013-0190-2.

[64] Martin A. Fischler and Robert C. Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Communications of the ACM*, 24(6):381–395, June 1981. doi: 10.1145/358669.358692.

[65] Tesca Fitzgerald, Pallavi Koppol, Patrick Callaghan, Russell Quinlan Jun Hei Wong, Reid Simmons, Oliver Kroemer, and Henny Admoni. INQUIRE: Interactive Querying for User-Aware Informative Reasoning. In *6th Annual Conference on Robot Learning*, 2022.

[66] Marshal F Folstein, Susan E Folstein, and Paul R McHugh. "Mini-Mental State": A Practical Method for Grading the Cognitive State of Patients for the Clinician. *Journal of Psychiatric Research*, 1975.

[67] Jodi Forlizzi and Carl DiSalvo. Service Robots in the Domestic Environment: A Study of the Roomba Vacuum in the Home. In *ACM SIGCHI/SIGART Conference on Human-Robot Interaction (HRI)*, pages 258–265, 2006. doi: 10.1145/1121241.1121286.

[68] Felix Franzel, Thomas Eiband, and Dongheui Lee. Detection of Collaboration and Collision Events during Contact Task Execution. In *2020 IEEE-RAS 20th International Conference on Humanoid Robots (Humanoids)*, pages 376–383. IEEE, 2021.

[69] Felix Fussenegger and Katta Spiel. Depending on Independence An Autoethnographic Account of Daily Use of Assistive Technologies. In *Proceedings of the 24th International ACM SIGACCESS Conference on Computers and Accessibility*, pages 1–6, 2022.

[70] Daniel Gallenberger, Tapomayukh Bhattacharjee, Youngsun Kim, and Siddhartha S Srinivasa. Transfer Depends on Acquisition: Analyzing Manipulation Strategies for Robotic Feeding. In *2019 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 267–276. IEEE, 2019.

[71] Jensen Gao, Bidipta Sarkar, Fei Xia, Ted Xiao, Jiajun Wu, Brian Ichter, Anirudha Majumdar, and Dorsa Sadigh. Physically Grounded Vision-Language Models for Robotic Manipulation. *arXiv preprint arXiv:2309.02561*, 2023.

[72] Yixing Gao, Hyung Jin Chang, and Yiannis Demiris. User Modelling for Personalised Dressing Assistance by Humanoid Robots. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1840–1845. IEEE, 2015.

[73] Caelan Reed Garrett, Tomás Lozano-Pérez, and Leslie Pack Kaelbling. PDDLStream: Integrating Symbolic Planners and Blackbox Samplers via Optimistic Adaptive Planning. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 30, pages 440–448, 2020.

[74] Caelan Reed Garrett, Rohan Chitnis, Rachel Holladay, Beomjoon Kim, Tom Silver, Leslie Pack Kaelbling, and Tomás Lozano-Pérez. Integrated Task and Motion Planning. *Annual Review of Control, Robotics, and Autonomous Systems*, 4(1):265–293, 2021.

[75] Sergio Garrido-Jurado, Rafael Muñoz-Salinas, Francisco José Madrid-Cuevas, and Manuel Jesús Marín-Jiménez. Automatic Generation and Detection of Highly Reliable Fiducial Markers under Occlusion. *Pattern Recognition*, 47(6):2280–2292, 2014.

[76] GlassOuse. GlassOuse Press Switch, 2025. URL `https://glassouse.com/product/press-switch-100gr-glassouse-assistive-device/`. (Accessed: 1st January, 2025).

[77] Rachel Gockley, Allison Bruce, Jodi Forlizzi, Marek Michalowski, Anne Mundell, Stephanie Rosenthal, Brennan Sellner, Reid Simmons, Kevin Snipes, Alan C. Schultz, and Jue Wang. Designing Robots for Long-Term Social Interaction. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1338–1343, 2005. doi: 10.1109/IROS.2005.1545303.

[78] Felix Ferdinand Goldau, Tejas Kumar Shastha, Maria Kyrarini, and Axel Gräser. Autonomous Multi-Sensory Robotic Assistant for a Drinking Task. In *2019 IEEE 16th International Conference on Rehabilitation Robotics (ICORR)*, pages 210–216. IEEE, 2019.

[79] Saskia Golz, Christian Osendorfer, and Sami Haddadin. Using Tactile Sensation for Learning Contact Knowledge: Discriminate Collision from Physical Interaction. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3788–3794. IEEE, 2015.

[80] Ethan K Gordon, Xiang Meng, Tapomayukh Bhattacharjee, Matt Barnes, and Siddhartha S Srinivasa. Adaptive Robot-Assisted Feeding: An Online Learning Framework for Acquiring Previously Unseen Food Items. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 9659–9666. IEEE, 2020.

[81] Ethan K. Gordon, Sumegh Roychowdhury, Tapomayukh Bhattacharjee, Kevin Jamieson, and Siddhartha S. Srinivasa. Leveraging Post Hoc Context for Faster Learning in Bandit Settings with Applications in Robot-Assisted Feeding. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10528–10535. IEEE Press, 2021. doi: 10.1109/ICRA48506.2021.9561520.

[82] Ethan K. Gordon, Rajat Kumar Jenamani, Amal Nanavati, Ziang Liu, Daniel Stabile, Xilai Dai, Tapomayukh Bhattacharjee, Tyler Schrenk, Jonathan Ko, Haya Bolotski, Raida Karim, Atharva Kashyap, Bernie Hao Zhu, Taylor Kessler Faulkner, and Siddhartha S. Srinivasa. An Adaptable, Safe, and Portable Robot-Assisted Feeding System. In *Companion of the 2024 ACM/IEEE International Conference on Human-Robot Interaction*, pages 74–76, 2024. doi: 10.1145/3610978.3641085.

[83] Ethan Kroll Gordon, Amal Nanavati, Ramya Challa, Bernie Hao Zhu, Taylor Annette Kessler Faulkner, and Siddhartha Srinivasa. Towards General Single-Utensil Food Acquisition with Human-Informed Actions. In *Conference on Robot Learning*, pages 2414–2428. PMLR, 2023.

[84] M.W. Granger, P.H. Buschang, G.S. Throckmorton, and S.T. Iannaccone. Masticatory Muscle Function in Patients with Spinal Muscular Atrophy. *American Journal of Orthodontics and Dentofacial Orthopedics*, 115(6):697–702, 1999. doi: 10.1016/S0889-5406(99)70296-9.

[85] Jennifer Grannen, Yilin Wu, Suneel Belkhale, and Dorsa Sadigh. Learning Bimanual Scooping Policies for Food Acquisition. In *6th Annual Conference on Robot Learning*, 2022. URL `https://openreview.net/forum?id=qDtbMK67PJG`.

[86] Jennifer Grannen, Siddharth Karamcheti, Suvir Mirchandani, Percy Liang, and Dorsa Sadigh. Vocal Sandbox: Continual Learning and Adaptation for Situated Human-Robot Collaboration. *arXiv preprint arXiv:2411.02599*, 2024.

[87] Phillip M Grice and Charles C Kemp. In-Home and Remote Use of Robotic Body Surrogates by People with Profound Motor Deficits. *PLoS ONE*, 14 (3):e0212904, 2019.

[88] Nayoung Ha, Ruolin Ye, Ziang Liu, Shubhangi Sinha, and Tapomayukh Bhattacharjee. REPeat: A Real2Sim2Real Approach for Pre-acquisition of Soft Food Items in Robot-assisted Feeding. *IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2024.

[89] Karen Hacker. *Community-Based Participatory Research*. Sage publications, 2013.

[90] Sami Haddadin, Alessandro De Luca, and Alin Albu-Schäffer. Robot Collisions: A Survey on Detection, Isolation, and Identification. *IEEE Transactions on Robotics*, 33(6):1292–1312, 2017.

[91] Choe Haidenhofer, Laurianne Sitbon, Chris P Beaumont, Maria Hoogstrate, and Jessica L Korte. From Research Participant to Co-Researcher: Chloe's Story on Co-Designing Inclusive Technologies with People with Intellectual Disability. In *Proceedings of the 26th International ACM SIGACCESS Conference on Computers and Accessibility*, pages 1–6, 2024.

[92] David Harel. Statecharts: A Visual Formalism for Complex Systems. *Science of Computer Programming*, 1987.

[93] Sandra G. Hart and Lowell E. Staveland. Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research. In Peter A. Hancock and Najmedin Meshkati, editors, *Human Mental Workload*, volume 52 of *Advances in Psychology*, pages 139–183. North-Holland, Amsterdam, 1988.

[94] Kris Hauser. The Minimum Constraint Removal Problem with Three Robotics Applications. *The International Journal of Robotics Research*, 33(1): 5–17, 2014.

[95] Nick Hawes, Christopher Burbridge, Ferdian Jovan, Lars Kunze, Bruno Lacerda, Lenka Mudrova, Jay Young, Jeremy Wyatt, Denise Hebesberger, Tobias Kortner, et al. The STRANDS Project: Long-Term Autonomy in Everyday Environments. *IEEE Robotics & Automation Magazine*, 24(3):146–156, 2017.

[96] Kelsey P Hawkins, Chih-Hung King, Tiffany L Chen, and Charles C Kemp. Informing Assistive Robots with Models of Contact Forces from Able-Bodied Face Wiping and Shaving. In *2012 IEEE RO-MAN: The 21st IEEE International Symposium on Robot and Human Interactive Communication*, pages 251–258. IEEE, 2012.

[97] Simon Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice Hall PTR, 1994.

[98] Jerry Zhi-Yang He, Zackory Erickson, Daniel S Brown, Aditi Raghunathan, and Anca Dragan. Learning Representations That Enable Generalization in Assistive Tasks. In *Conference on Robot Learning*, pages 2105–2114. PMLR, 2023.

[99] Mehdi Hellou, Norina Gasteiger, Jong Yoon Lim, Minsu Jang, and Ho Seok Ahn. Personalization and Localization in Human-Robot Interaction: A Review of Technical Methods. *Robotics*, 10(4):120, 2021.

[100] Malte Helmert. The Fast Downward Planning System. *Journal of Artificial Intelligence Research*, 26:191–246, 2006.

[101] Richard P. Hermann, Anna C. Phalangas, Richard M. Mahoney, and Michael A. Alexander. Powered Feeding Devices: An Evaluation of Three Models. *Archives of Physical Medicine and Rehabilitation*, 80(10):1237–1242, 1999. doi: 10.1016/S0003-9993(99)90022-9.

[102] Morten Hertzum. Reference Values and Subscale Patterns for the Task Load Index (TLX): A Meta-Analytic Review. *Ergonomics*, 64(7):869–878, 2021.

[103] Wolfgang Hess, Damon Kohler, Holger Rapp, and Daniel Andor. Real-Time Loop Closure in 2D LiDAR SLAM. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1271–1278. IEEE, 2016.

[104] Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, et al. Inner Monologue: Embodied Reasoning through Planning with Language Models. *arXiv preprint arXiv:2207.05608*, 2022.

[105] Wenlong Huang, Chen Wang, Ruohan Zhang, Yunzhu Li, Jiajun Wu, and Li Fei-Fei. VoxPoser: Composable 3D Value Maps for Robotic Manipulation with Language Models. *arXiv preprint arXiv:2307.05973*, 2023.

[106] Wenlong Huang, Chen Wang, Yunzhu Li, Ruohan Zhang, and Li Fei-Fei. ReKep: Spatio-Temporal Reasoning of Relational Keypoint Constraints for Robotic Manipulation. *arXiv preprint arXiv:2409.01652*, 2024.

[107] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. GPT-4o System Card. *arXiv preprint arXiv:2410.21276*, 2024.

[108] IEEE. IEEE Standard for Transparency of Autonomous Systems. Standard IEEE Std 7001-2021, IEEE, 2022.

[109] Takeshi Ikeda, Tetsuhiro Makino, Seiji Furuno, and Fusaomi Nagata. Wiping the Mouth Using the Robot Arm with Information of RGB-D Sensor. *Artificial Life and Robotics*, 24:360–367, 2019.

[110] Shuhei Ikemoto, Heni Ben Amor, Takashi Minato, Bernhard Jung, and Hiroshi Ishiguro. Physical Human-Robot Interaction: Mutual Learning and Adaptation. *IEEE Robotics & Automation Magazine*, 19(4):24–35, 2012. doi: 10.1109/MRA.2011.2181676.

[111] Intel. Intel RealSense D435i, 2025. URL https://www.intelrealsense.com/depth-camera-d435i/. (Accessed: 1st January, 2025).

[112] Intel. Intel NUC 12 Pro Mini PC Kit (NUC12WSHi7), 2025. (Accessed: 1st January, 2025).

[113] International Organization for Standardization. Robots and Robotic Devices: Safety Requirements for Personal Care Robots. ISO 13482:2014, 2014.

[114] Bahar Irfan, Nikhil Churamani, Michelle Zhao, Ali Ayub, and Silvia Rossi. Lifelong Learning and Personalization in Long-Term Human-Robot Interaction (LEAP-HRI): 5th Edition Workshop at HRI 2025, 2025. URL https://leap-hri.github.io/.

[115] Sumio Ishii, S. Tanaka, and F. Hiramatsu. Meal Assistance Robot for Severely Handicapped People. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1308–1313, 1995. doi: 10.1109/ROBOT.1995.525461.

[116] Anja Jackowski, Marion Gebhard, and Roland Thietje. Head Motion and Head Gesture-Based Robot Control: A Usability Study. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 26(1):161–170, 2017.

[117] Catrine Jacobsson, Karin Axelsson, Per Olov Österlind, and Astrid Norberg. How People with Stroke and Healthy Older People Experience the Eating Process. *Journal of Clinical Nursing*, 9(2):255–264, 2000. doi: 10.1046/j.1365-2702.2000.00355.x.

[118] Siddarth Jain and Brenna Argall. Probabilistic Human Intent Recognition for Shared Autonomy in Assistive Robotics. *ACM Transactions on Human-Robot Interaction*, 2019.

[119] S. Javdani, H. Admoni, S. Pellegrinelli, S.S. Srinivasa, and J.A. Bagnell. Shared Autonomy via Hindsight Optimization for Teleoperation and Teaming. *The International Journal of Robotics Research*, 2018.

[120] Rajat Kumar Jenamani, Daniel Stabile, Ziang Liu, Abrar Anwar, Katherine Dimitropoulou, and Tapomayukh Bhattacharjee. Feel the Bite: Robot-Assisted Inside-Mouth Bite Transfer using Robust Mouth Perception and Physical Interaction-Aware Control. In *Proceedings of the 2024 ACM/IEEE International Conference on Human-Robot Interaction*, pages 313–322, 2024.

[121] Rajat Kumar Jenamani, Priya Sundaresan, Maram Sakr, Tapomayukh Bhattacharjee, and Dorsa Sadigh. FLAIR: Feeding via Long-horizon AcquIsition of Realistic dishes. In *Proceedings of Robotics: Science and Systems (RSS)*, 2024.

[122] Rajat Kumar Jenamani, Tom Silver, Ben Dodson, Shiqin Tong, Anthony Song, Yuting Yang, Ziang Liu, Benjamin Howe, Aimee Whitneck, and Tapomayukh Bhattacharjee. FEAST: A Flexible Mealtime-Assistance System Towards In-the-Wild Personalization. In *Robotics: Science and Systems (RSS)*, 2025.

[123] Rajat Kumar Jenamani, Bohan Yang, Aimee Whitneck, Ben Dodson, Eric Hu, Yunting Yan, Tom Silver, and Tapomayukh Bhattacharjee. Human–Robot Co-Adaptation in Month-Long In-Home Mealtime Assistance. Under review at the ACM/IEEE International Conference on Human-Robot Interaction (HRI 2027), 2026.

[124] Hong Jun Jeon, Dylan P Losey, and Dorsa Sadigh. Shared Autonomy with Learned Latent Actions. *arXiv preprint arXiv:2005.03210*, 2020.

[125] Hanxiao Jiang, Binghao Huang, Ruihai Wu, Zhuoran Li, Shubham Garg, Hooshang Nayyeri, Shenlong Wang, and Yunzhu Li. RoboEXP: Action-Conditioned Scene Graph via Interactive Exploration for Robotic Manipulation. *arXiv preprint arXiv:2402.15487*, 2024.

[126] Yifeng Jiang and C Karen Liu. Data-Driven Approach to Simulating Realistic Human Joint Constraints. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[127] Ajjen Joshi, Soumya Ghosh, Margrit Betke, Stan Sclaroff, and Hanspeter Pfister. Personalizing Gesture Recognition Using Hierarchical Bayesian Neural Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6513–6522, 2017.

[128] Petr Kadleček, Alexandru-Eugen Ichim, Tiantian Liu, Jaroslav Křivánek, and Ladislav Kavan. Reconstructing Personalized Anatomical Models for Physics-Based Body Animation. *ACM Transactions on Graphics*, 2016.

[129] Subbarao Kambhampati. Can Large Language Models Reason and Plan? *Annals of the New York Academy of Sciences*, 2024.

[130] Takayuki Kanda, Takayuki Hirano, Daniel Eaton, and Hiroshi Ishiguro. Interactive Robots as Social Partners and Peer Tutors for Children: A Field Trial. *Human–Computer Interaction*, 19(1–2):61–84, 2004. doi: 10.1207/s15327051hci1901\&2_4.

[131] Takayuki Kanda, Rumi Sato, Naoki Saiwaki, and Hiroshi Ishiguro. A Two-Month Field Trial in an Elementary School for Long-Term Human–Robot Interaction. *IEEE Transactions on Robotics*, 23(5):962–971, 2007. doi: 10.1109/TRO.2007.904904.

[132] Ruya Karagulle, Nikos Aréchiga, Andrew Best, Jonathan DeCastro, and Necmiye Ozay. A Safe Preference Learning Approach for Personalization with Applications to Autonomous Vehicles. *IEEE Robotics and Automation Letters*, 2024.

[133] Siddharth Karamcheti, Suraj Nair, Annie S Chen, Thomas Kollar, Chelsea Finn, Dorsa Sadigh, and Percy Liang. Language-Driven Representation Learning for Robotics. *arXiv preprint arXiv:2302.12766*, 2023.

[134] Atharva S Kashyap, Ugne Aleksandra Morkute, and Patricia Alves-Oliveira. Robot-Assisted Social Dining as a White Glove Service. In *Proceedings of the 2026 CHI Conference on Human Factors in Computing Systems*, CHI '26, New York, NY, USA, 2026. ACM. doi: 10.1145/3772318.3790481. arXiv:2602.15767.

[135] Sidney Katz, Amasa B Ford, Roland W Moskowitz, Beverly A Jackson, and Marjorie W Jaffe. Studies of Illness in the Aged: The Index of ADL: A Standardized Measure of Biological and Psychosocial Function. *JAMA*, 185(12):914–919, 1963.

[136] Maya N Keely, Heramb Nemlekar, and Dylan P Losey. Kiri-Spoon: A Soft Shape-Changing Utensil for Robot-Assisted Feeding. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024.

[137] Charles C. Kemp, Aaron Edsinger, Henry M. Clever, and Blaine Matulevich. The Design of Stretch: A Compact, Lightweight Mobile Manipulator for Indoor Human Environments. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3150–3157, 2022. doi: 10.1109/ICRA46639.2022.9811922.

[138] Oussama Khatib. A Unified Approach for Motion and Force Control of Robot Manipulators: The Operational Space Formulation. *IEEE Journal on Robotics and Automation*, 3(1):43–53, 1987.

[139] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. OpenVLA: An Open-Source Vision-Language-Action Model. *arXiv preprint arXiv:2406.09246*, 2024.

[140] Kinova. Kinova Gen3 6DoF Robotic Arm, 2024. URL https://www.kinovarobotics.com/product/gen3-robots. (Accessed: 1st January, 2024).

[141] Kinova. Kinova Gen3 7DoF Robotic Arm, 2025. URL https://www.kinovarobotics.com/uploads/User-Guide-Gen3-R07.pdf. (Accessed: 1st January, 2025).

[142] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment Anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4015–4026, 2023.

[143] Alexandros Kouris, Fotios Dimeas, and Nikos Aspragathos. A Frequency Domain Approach for Contact Type Distinction in Human–Robot Collaboration. *IEEE Robotics and Automation Letters*, 3(2):720–727, 2018.

[144] Jane E. Kreckman and Saleem J. Sheredos. Winsford Feeder. Clinical evaluation report, Veterans Administration Rehabilitation Engineering Center, pp. 39–44, 1981. Reprinted as Appendix C in G. N. Phillips, *Feasibility Study for Assistive Feeder*, Final Report, SwRI Project No. 14-1164, Southwest Research Institute, San Antonio, TX, 1987.

[145] Nishanth Kumar, Tom Silver, Willie McClinton, Linfeng Zhao, Stephen Proulx, Tomás Lozano-Pérez, Leslie Pack Kaelbling, and Jennifer Barry. Practice Makes Perfect: Planning to Learn Skill Parameter Policies. In *Robotics: Science and Systems (RSS)*, 2024.

[146] Raja S Kushalnagar and Christian Vogler. Teleconference Accessibility and Guidelines for Deaf and Hard of Hearing Users. In *Proceedings of the 22nd International ACM SIGACCESS Conference on Computers and Accessibility*, pages 1–6, 2020.

[147] Minae Kwon, Hengyuan Hu, Vivek Myers, Siddharth Karamcheti, Anca Dragan, and Dorsa Sadigh. Toward Grounded Social Reasoning. *arXiv preprint arXiv:2306.08651*, 2023.

[148] Colin Lea, Michael D Flynn, Rene Vidal, Austin Reiter, and Gregory D Hager. Temporal Convolutional Networks for Action Segmentation and Detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 156–165, 2017.

[149] Jason Lee, Jiafei Duan, Haoquan Fang, Yuquan Deng, Shuo Liu, Boyang Li, Bohan Fang, Jieyu Zhang, Yi Ru Wang, Sangho Lee, Winson Han, Wilbert Pumacay, Angelica Wu, Rose Hendrix, Karen Farley, Eli VanderBilt, Ali Farhadi, Dieter Fox, and Ranjay Krishna. MolmoAct: Action Reasoning

Models that can Reason in Space. *arXiv preprint arXiv:2508.07917*, 2025.

[150] Michelle A Lee, Yuke Zhu, Krishnan Srinivasan, Parth Shah, Silvio Savarese, Li Fei-Fei, Animesh Garg, and Jeannette Bohg. Making Sense of Vision and Touch: Self-Supervised Learning of Multimodal Representations for Contact-Rich Tasks. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8943–8950. IEEE, 2019.

[151] Min Kyung Lee, Jodi Forlizzi, Sara Kiesler, Paul Rybski, John Antanitis, and Sarun Savetsila. Personalization in HRI: A Longitudinal Field Experiment. In *Proceedings of the Seventh Annual ACM/IEEE International Conference on Human-Robot Interaction*, pages 319–326, 2012.

[152] Iolanda Leite, Carlos Martinho, and Ana Paiva. Social Robots for Long-Term Interaction: A Survey. *International Journal of Social Robotics*, 5(2): 291–308, 2013. doi: 10.1007/s12369-013-0178-y.

[153] Lenovo. Lenovo Legion Pro 7i Gaming Laptop. `https://www.lenovo.com/us/en/p/laptops/legion-laptops/legion-7-series/lenovo-legion-pro-7i-gen-8-(16-inch-intel)/len101g0018`, 2023.

[154] Lenovo. Legion Pro 7i Gen 9 Intel (16") with RTX 4090 (83DE000AUS), 2025. URL `https://www.lenovo.com/us/en/p/laptops/legion-laptops/legion-pro-series/lenovo-legion-pro-7i-gen-9-16-inch-intel/83de000aus`. (Accessed: 1st January, 2025).

[155] Daniel Leyzberg, Aditi Ramachandran, and Brian Scassellati. The Effect of Personalization in Longer-Term Robot Tutoring. *ACM Transactions on Human-Robot Interaction*, 7(3):1–19, 2018.

[156] Amber Li and Tom Silver. Embodied Active Learning of Relational State Abstractions for Bilevel Planning. *Conference on Lifelong Learning Agents (CoLLAs)*, 2023.

[157] Bowen Li, Tom Silver, Sebastian Scherer, and Alexander Gray. Bilevel Learning for Bilevel Planning. In *Robotics: Science and Systems (RSS)*, 2025.

[158] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. BLIP: Bootstrapping Language-Image Pre-Training for Unified Vision-Language Understanding and Generation. In *International Conference on Machine Learning (ICML)*, pages 12888–12900. PMLR, 2022.

[159] Tianye Li, Timo Bolkart, Michael J Black, Hao Li, and Javier Romero. Learning a Model of Facial Shape and Expression from 4D Scans. *ACM Transactions on Graphics*, 36(6):194:1–194:17, 2017.

[160] Yunzhu Li, Jun-Yan Zhu, Russ Tedrake, and Antonio Torralba. Connecting Touch and Vision via Cross-Modal Prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10609–10618, 2019.

[161] Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as Policies: Language Model Programs for Embodied Control. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9493–9500. IEEE, 2023.

[162] Yichao Liang, Nishanth Kumar, Hao Tang, Adrian Weller, Joshua B Tenenbaum, Tom Silver, João F Henriques, and Kevin Ellis. VisualPredicator: Learning Abstract World Models with Neuro-Symbolic Predicates for Robot Planning. *arXiv preprint arXiv:2410.23156*, 2024.

[163] Kevin Lin, Christopher Agia, Toki Migimatsu, Marco Pavone, and Jeannette Bohg. Text2Motion: From Natural Language Instructions to Feasible Plans. *Autonomous Robots*, 47(8):1345–1365, 2023.

[164] Martina Lippi, Giuseppe Gillini, Alessandro Marino, and Filippo Arrichiello. A Data-Driven Approach for Contact Detection, Classification and Reaction in Physical Human-Robot Collaboration. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3597–3603. IEEE, 2021.

[165] Bing Liu. Learning on the Job: Online Lifelong and Continual Learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

[166] Fangchen Liu, Kuan Fang, Pieter Abbeel, and Sergey Levine. MOKA: Open-Vocabulary Robotic Manipulation through Mark-Based Visual Prompting. *arXiv preprint arXiv:2403.03174*, 2024.

[167] Peiqi Liu, Yaswanth Orru, Jay Vakil, Chris Paxton, Nur Muhammad Mahi Shafiullah, and Lerrel Pinto. OK-Robot: What Really Matters in Integrating Open-Knowledge Models for Robotics. In *Robotics: Science and Systems (RSS)*, 2024.

[168] Puze Liu, Davide Tateo, Haitham Bou Ammar, and Jan Peters. Robot Reinforcement Learning on the Constraint Manifold. In *Conference on Robot Learning*, pages 1357–1366. PMLR, 2022.

[169] Puze Liu, Haitham Bou-Ammar, Jan Peters, and Davide Tateo. Safe Reinforcement Learning on the Constraint Manifold: Theory and Applications. *arXiv preprint arXiv:2404.09080*, 2024.

[170] Rui Liu, Zahiruddin Mahammad, Amisha Bhaskar, and Pratap Tokekar. IMRL: Integrating Visual, Physical, Temporal, and Geometric Representations for Enhanced Food Acquisition. *arXiv preprint arXiv:2409.12092*, 2024.

[171] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, et al. Grounding DINO: Marrying DINO with Grounded Pre-Training for Open-Set Object Detection. *arXiv preprint arXiv:2303.05499*, 2023.

[172] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin Transformer: Hierarchical Vision Transformer Using Shifted Windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021.

[173] Ziang Liu, Yuanchen Ju, Yu Da, Tom Silver, Pranav N Thakkar, Jenna Li, Justin Guo, Katherine Dimitropoulou, and Tapomayukh Bhattacharjee. GRACE: Generalizing Robot-Assisted Caregiving with User Functionality Embeddings. *arXiv preprint arXiv:2501.17855*, 2025.

[174] Ilya Loshchilov and Frank Hutter. SGDR: Stochastic Gradient Descent with Warm Restarts. *arXiv preprint arXiv:1608.03983*, 2016.

[175] Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[176] Joao Loula, Kelsey Allen, Tom Silver, and Josh Tenenbaum. Learning Constraint-Based Planning Models from Demonstrations. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5410–5416. IEEE, 2020.

[177] Tomás Lozano-Pérez and Leslie Pack Kaelbling. A Constraint-Based Method for Solving Sequential Manipulation Planning Problems. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3684–3691. IEEE, 2014.

[178] J Lynch and R Cahalan. The Impact of Spinal Cord Injury on the Quality of Life of Primary Family Caregivers: A Literature Review. *Spinal Cord*, 55 (11):964–978, 2017.

[179] Yecheng Jason Ma, William Liang, Guanzhi Wang, De-An Huang, Osbert Bastani, Dinesh Jayaraman, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Eureka: Human-Level Reward Design via Coding Large Language Models. *arXiv preprint arXiv:2310.12931*, 2023.

[180] Rishabh Madan, Rajat Kumar Jenamani, Vy Thuy Nguyen, Ahmed Moustafa, Xuefeng Hu, Katherine Dimitropoulou, and Tapomayukh Bhattacharjee. SPARCS: Structuring Physically Assistive Robotics for Caregiving with Stakeholders-in-the-Loop. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 641–648. IEEE, 2022.

[181] Rishabh Madan, Jiawei Lin, Mahika Goel, Angchen Xie, Xiaoyu Liang, Marcus Lee, Justin Guo, Pranav N. Thakkar, Rohan Banerjee, Jose Barreiros, Kate Tsui, Tom Silver, and Tapomayukh Bhattacharjee. PrioriTouch: Adapting to User Contact Preferences for Whole-Arm Physical Human-Robot Interaction. In *Conference on Robot Learning (CoRL)*, 2025.

[182] Rishabh Madan et al. SPARCS-box. `https://emprise.cs.cornell.edu/sparcs`, 2022.

[183] Rishabh Madan et al. SPARCS Supplementary Website. `https://emprise.cs.cornell.edu/sparcs`, 2022.

[184] Madentec. Tracker Pro 2, 2025. URL `https://www.schoolhealth.com/tracker-pro-2`. (Accessed: 1st January, 2025).

[185] Anne Mandy, Tara Sims, Graham Stew, and Dominic Onions. Manual Feeding Device Experiences of People with a Neurodisability. *The American Journal of Occupational Therapy*, 72(3):7203345010p1–7203345010p5, 2018.

[186] Bente Martinsen, Ingegerd Harder, and Fin Biering-Sørensen. The Meaning of Assisted Feeding for People Living with Spinal Cord Injury: A Phenomenological Study. *Journal of Advanced Nursing*, 62(5):533–540, 2008. doi: 10.1111/j.1365-2648.2008.04637.x.

[187] Drew M McDermott. The 1998 AI Planning Systems Competition. *AI Magazine*, 21(2):35, 2000. doi: 10.1609/aimag.v21i2.1506.

[188] Jorge Mendez-Mendez, Leslie Pack Kaelbling, and Tomás Lozano-Pérez. Embodied Lifelong Learning for Task and Motion Planning. In *Conference on Robot Learning*, pages 2134–2150. PMLR, 2023.

[189] Utkarsh Aashu Mishra, Shangjie Xue, Yongxin Chen, and Danfei Xu. Generative Skill Chaining: Long-Horizon Skill Planning with Diffusion Models. In *Conference on Robot Learning*, pages 2905–2925. PMLR, 2023.

[190] Sanika Moharana, Alejandro E Panduro, Hee Rin Lee, and Laurel D Riek. Robots for Joy, Robots for Sorrow: Community Based Robot Design for Dementia Caregivers. In *2019 14th ACM/IEEE International Conference on*

Human-Robot Interaction (HRI), pages 458–467. IEEE, 2019.

[191] Thomas Moore and Daniel Stouch. A Generalized Extended Kalman Filter Implementation for the Robot Operating System. In *Intelligent Autonomous Systems 13*, pages 335–348. Springer, 2014.

[192] JE Morley and N Tumosa. Saint Louis University Mental Status Examination (SLUMS). *Aging Successfully*, 2002.

[193] Debasmita Mukherjee, Jayden Hong, Haripriya Vats, Sooyeon Bae, and Homayoun Najjaran. Personalization of Industrial Human–Robot Communication through Domain Adaptation Based on User Feedback. *User Modeling and User-Adapted Interaction*, pages 1–41, 2024.

[194] Yoshihiko Nakamura and Hideo Hanafusa. Inverse Kinematic Solutions with Singularity Robustness for Robot Manipulator Control. *Journal of Dynamic Systems, Measurement, and Control*, 108(3):163–171, 1986.

[195] Yoshihiko Nakamura, Hideo Hanafusa, and Tsuneo Yoshikawa. Task-Priority Based Redundancy Control of Robot Manipulators. *The International Journal of Robotics Research*, 6(2):3–15, 1987.

[196] Amal Nanavati, Patricia Alves-Oliveira, Tyler Schrenk, Ethan K. Gordon, Maya Cakmak, and Siddhartha S. Srinivasa. Design Principles for Robot-Assisted Feeding in Social Contexts. In *Proceedings of the 2023 ACM/IEEE International Conference on Human-Robot Interaction*, HRI '23, pages 24–33, New York, NY, USA, 2023. Association for Computing Machinery. doi: 10.1145/3568162.3576988.

[197] Amal Nanavati, Vinitha Ranganeni, and Maya Cakmak. Physically Assistive Robots: A Systematic Review of Mobile and Manipulator Robots That Physically Assist People with Disabilities. *Annual Review of Control, Robotics, and Autonomous Systems*, 7:123–147, 2024.

[198] Amal Nanavati, Ethan K Gordon, Taylor A Kessler Faulkner, Yuxin (Ray) Song, Johnathan Ko, Tyler Schrenk, Vy Nguyen, Bernie Hao Zhu, Haya Bolotski, Atharva Kashyap, Sriram Kutty, Raida Karim, Liander Rainbolt, Rosario Scalise, Hanjun Song, Ramon Qu, Maya Cakmak, and Siddhartha S Srinivasa. Lessons Learned from Designing and Evaluating a Robot-assisted Feeding System for Out-of-lab Use. In *Proceedings of the 2025 ACM/IEEE International Conference on Human-Robot Interaction*, 2025.

[199] Isira Naotunna, Chamika Janith Perera, Chameera Sandaruwan, RARC Gopura, and Thilina Dulantha Lalitharatne. Meal Assistance Robots: A Review on Current Status, Challenges and Future Directions. In *IEEE/SICE*

International Symposium on System Integration, pages 211–216, 2015.

[200] Neater. Neater Eater Robot, 2024. URL https://www.neater.co.uk/neater-eater-robotic. (Accessed: 1st January, 2024).

[201] NETGEAR. NETGEAR Nighthawk WiFi Router (RAX43), 2025. URL https://www.amazon.com/NETGEAR-Nighthawk-5-Stream-Router-RAX43/dp/B08JY6RMG9/. (Accessed: 1st January, 2025).

[202] Niels Christian Mossfeldt Nickelsen. Imagining and Tinkering with Assistive Robotics in Care for the Disabled. *Paladyn, Journal of Behavioral Robotics*, 10(1):128–139, 2019. doi: 10.1515/pjbr-2019-0009.

[203] Stefanos Nikolaidis, David Hsu, and Siddhartha Srinivasa. Human-Robot Mutual Adaptation in Collaborative Tasks: Models and Experiments. *The International Journal of Robotics Research*, 36(5–7):618–634, 2017. doi: 10.1177/0278364917690593.

[204] Arne Nordmann, Christian Emmerich, Stefan Rüther, Andre Lemme, Sebastian Wrede, and Jochen Steil. Teaching Nullspace Constraints in Physical Human-Robot Interaction Using Reservoir Computing. In *2012 IEEE International Conference on Robotics and Automation*, pages 1868–1875. IEEE, 2012.

[205] Obi. Obi, 2024. URL https://meetobi.com. (Accessed: 1st January, 2024).

[206] Jan Ondras, Abrar Anwar, Tong Wu, Fanjun Bu, Malte Jung, Jorge Jose Ortiz, and Tapomayukh Bhattacharjee. Human-Robot Commensality: Bite Timing Prediction for Robot-Assisted Feeding in Groups. In *Proceedings of the 6th Conference on Robot Learning (CoRL)*, volume 205 of *Proceedings of Machine Learning Research*, pages 921–933. PMLR, 2022.

[207] Open Robotics. ROS Noetic, 2020. URL https://wiki.ros.org/noetic. (Accessed: May 2022).

[208] Open Robotics. Unified Robot Description Format (URDF). http://wiki.ros.org/urdf, 2022. (Accessed: 31st July, 2022).

[209] OpenAI. GPT-4o-mini. https://openai.com, 2024. Large language model.

[210] Abhishek Padalkar, Acorn Pooley, Ajinkya Jain, Alex Bewley, Alex Herzog, Alex Irpan, Alexander Khazatsky, Anant Rai, Anikait Singh, Anthony Brohan, et al. Open X-Embodiment: Robotic Learning Datasets and RT-X Models. *arXiv preprint arXiv:2310.08864*, 2023.

[211] Akhil Padmanabha, Janavi Gupta, Chen Chen, Jehan Yang, Vy Nguyen, Douglas J Weber, Carmel Majidi, and Zackory Erickson. Independence in the Home: A Wearable Interface for a Person with Quadriplegia to Teleoperate a Mobile Manipulator. In *Proceedings of the 2024 ACM/IEEE International Conference on Human-Robot Interaction*, pages 542–551, 2024.

[212] Akhil Padmanabha, Jessie Yuan, Janavi Gupta, Zulekha Karachiwalla, Carmel Majidi, Henny Admoni, and Zackory Erickson. VoicePilot: Harnessing LLMs as Speech Interfaces for Physically Assistive Robots. *arXiv preprint arXiv:2404.04066*, 2024.

[213] Sagar Parekh and Dylan P. Losey. Learning Latent Representations to Co-Adapt to Humans. *Autonomous Robots*, 47(6):771–796, 2023. doi: 10.1007/s10514-023-10109-5.

[214] Daehyung Park, Yuuna Hoshi, Harshal P. Mahajan, Ho Keun Kim, Zackory Erickson, Wendy A. Rogers, and Charles C. Kemp. Active Robot-Assisted Feeding with a General-Purpose Mobile Manipulator: Design, Evaluation, and Lessons Learned. *Robotics and Autonomous Systems*, 124:103344, 2020. doi: 10.1016/j.robot.2019.103344.

[215] Maithili Patel and Sonia Chernova. Robot Behavior Personalization from Sparse User Feedback. *arXiv preprint arXiv:2410.19219*, 2024.

[216] Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed AA Osman, Dimitrios Tzionas, and Michael J Black. Expressive Body Capture: 3D Hands, Face, and Body from a Single Image. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[217] Serafeim Perdikis, Luca Tonin, Sareh Saeedi, Christoph Schneider, and José del R. Millán. The Cybathlon BCI Race: Successful Longitudinal Mutual Learning with Two Tetraplegic Users. *PLOS Biology*, 16(5):e2003787, 2018. doi: 10.1371/journal.pbio.2003787.

[218] Betsy Phillips and Hongxin Zhao. Predictors of Assistive Technology Abandonment. *Assistive Technology*, 5(1):36–45, 1993. doi: 10.1080/10400435.1993.10132205.

[219] Physical Intelligence, Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Kevin Black, Danny Driess, Chelsea Finn, Karol Hausman, Brian Ichter, Sergey Levine, Karl Pertsch, et al. $\pi^*_{0.6}$: a VLA That Learns From Experience. *arXiv preprint arXiv:2511.14759*, 2025.

[220] Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, et al. $\pi_{0.5}$: a Vision-Language-Action Model with Open-World Generalization. *arXiv preprint arXiv:2504.16054*, 2025.

[221] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving Language Understanding by Generative Pre-Training. Technical report, OpenAI, 2018.

[222] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language Models Are Unsupervised Multitask Learners. *OpenAI blog*, 1(8):9, 2019.

[223] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning Transferable Visual Models from Natural Language Supervision. In *International Conference on Machine Learning (ICML)*, pages 8748–8763. PMLR, 2021.

[224] Vinitha Ranganeni, Vy Nguyen, Henry Evans, Jane Evans, Julian Mehu, Samuel Olatunji, Wendy Rogers, Aaron Edsinger, Charles Kemp, and Maya Cakmak. Robots for Humanity: In-Home Deployment of Stretch RE2. In *Companion of the 2024 ACM/IEEE International Conference on Human-Robot Interaction*, pages 1299–1301, 2024.

[225] Nancy Berryman Reese and William D Bandy. *Joint Range of Motion and Muscle Length Testing-E-Book*. Elsevier Health Sciences, 2016.

[226] RO Robinson. Arthrogryposis Multiplex Congenita; Feeding, Language and Other Health Problems. *Neuropediatrics*, 21(4):177–178, 1990.

[227] Robotiq. Robotiq 2F-85 Gripper, 2025. URL `https://robotiq.com/products/2f85-140-adaptive-robot-gripper`. (Accessed: 1st January, 2025).

[228] ROS. move_base: ROS Navigation Stack. `http://wiki.ros.org/move_base`, 2024.

[229] Antoni Rosinol, Arjun Gupta, Marcus Abate, Jingnan Shi, and Luca Carlone. 3D Dynamic Scene Graphs: Actionable Spatial Perception with Places, Objects, and Humans. *arXiv preprint arXiv:2002.06289*, 2020.

[230] Christoph Rösmann, Wendelin Feiten, Thomas Wösch, Frank Hoffmann, and Torsten Bertram. Efficient Trajectory Optimization Using a Sparse Model. In *2013 European Conference on Mobile Robots (ECMR)*, pages 138–143. IEEE, 2013.

[231] Rovi. Rovi Wheelchair, 2025. URL `https://www.rovimobility.com/`. (Accessed: 1st January, 2025).

[232] Hoseok Ryu, Minseok Kim, Seungwhan Lee, Moon Seok Park, Kyoungmin Lee, and Jehee Lee. Functionality-Driven Musculature Retargeting. In *Computer Graphics Forum*, 2021.

[233] Camille Salaün, Vincent Padois, and Olivier Sigaud. Control of Redundant Robots Using Learned Models: An Operational Space Control Approach. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 878–885. IEEE, 2009.

[234] Maarten Sap, Vered Shwartz, Antoine Bosselut, Yejin Choi, and Dan Roth. Commonsense Reasoning for Natural Language Processing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*, pages 27–33, 2020.

[235] Joe Saunders, Dag Sverre Syrdal, Kheng Lee Koay, Nathan Burke, and Kerstin Dautenhahn. "Teach Me Show Me"—End-User Personalization of a Smart Home and Companion Robot. *IEEE Transactions on Human-Machine Systems*, 46(1):27–40, 2015.

[236] Jeff Sauro and James R. Lewis. *Quantifying the User Experience: Practical Statistics for User Research*. Morgan Kaufmann, Cambridge, MA, 2 edition, 2016.

[237] Brian Scassellati. Theory of Mind for a Humanoid Robot. *Autonomous Robots*, 2002.

[238] Brian Scassellati, Laura Boccanfuso, Chien-Ming Huang, Marilena Mademtzi, Meiying Qin, Nicole Salomons, Pamela Ventola, and Frederick Shic. Improving Social Skills in Children with ASD Using a Long-Term, In-Home Social Robot. *Science Robotics*, 3(21):eaat7544, 2018. doi: 10.1126/scirobotics.aat7544.

[239] Sebastian Schröer, Ingo Killmann, Barbara Frank, Martin Völker, Lukas Fiederer, Tonio Ball, and Wolfram Burgard. An Autonomous Robotic Assistant for Drinking. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6482–6487. IEEE, 2015.

[240] Lorenzo Sciavicco and Bruno Siciliano. *Modelling and Control of Robot Manipulators*. Springer Science & Business Media, 2001.

[241] Burr Settles. *Active Learning Literature Survey*. PhD thesis, University of Wisconsin-Madison Department of Computer Sciences, 2009.

[242] Nur Muhammad Mahi Shafiullah, Anant Rai, Haritheja Etukuru, Yiqian Liu, Ishan Misra, Soumith Chintala, and Lerrel Pinto. On Bringing Robots Home. In *arXiv preprint arXiv:2311.16098*, 2023.

[243] Ali Shafti, Jonas Tjomsland, William Dudley, and A. Aldo Faisal. Real-World Human-Robot Collaborative Reinforcement Learning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 11161–11166, 2020. doi: 10.1109/IROS45743.2020.9341473.

[244] Lorenzo Shaikewitz, Yilin Wu, Suneel Belkhale, Jennifer Grannen, Priya Sundaresan, and Dorsa Sadigh. In-Mouth Robotic Bite Transfer with Visual and Haptic Sensing. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9885–9895. IEEE, 2023.

[245] Lucy Xiaoyang Shi, Zheyuan Hu, Tony Z Zhao, Archit Sharma, Karl Pertsch, Jianlan Luo, Sergey Levine, and Chelsea Finn. Yell At Your Robot: Improving On-the-Fly from Language Corrections. *arXiv preprint arXiv:2403.12910*, 2024.

[246] Samantha E. Shune. An Altered Eating Experience: Attitudes Toward Feeding Assistance Among Younger and Older Adults. *Rehabilitation Nursing*, 2020.

[247] Tom Silver, Rohan Chitnis, Nishanth Kumar, Willie McClinton, Tomás Lozano-Pérez, Leslie Kaelbling, and Joshua B Tenenbaum. Predicate Invention for Bilevel Planning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023.

[248] Tom Silver, Rajat Kumar Jenamani, Ziang Liu, Ben Dodson, and Tapomayukh Bhattacharjee. Coloring Between the Lines: Personalization in the Null Space of Planning Constraints. *arXiv preprint arXiv:2505.15503*, 2025.

[249] Ishika Singh, Valts Blukis, Arsalan Mousavian, Ankit Goyal, Danfei Xu, Jonathan Tremblay, Dieter Fox, Jesse Thomason, and Animesh Garg. ProgPrompt: Generating Situated Robot Task Plans Using Large Language Models. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11523–11530. IEEE, 2023.

[250] Slamtec. RPLIDAR A1 360 Degree Laser Scanner. `https://www.slamtec.com/en/Lidar/A1`, 2024.

[251] Ryoji Soyama, Sumio Ishii, and Azuma Fukase. Selectable Operating Interfaces of the Meal-Assistance Device "My Spoon". In *Advances in Rehabilitation Robotics*, volume 306 of *Lecture Notes in Control and Information*

Sciences, pages 155–163. Springer, 2004. doi: 10.1007/10946978_8.

[252] Samuel Spaulding, Jocelyn Shen, Hae Won Park, and Cynthia Breazeal. Lifelong Personalization via Gaussian Process Modeling for Long-Term HRI. *Frontiers in Robotics and AI*, 8:683066, 2021.

[253] Anthony Staros and Edward Peizer. Veterans Administration Prosthetics Center Research Report. *Bulletin of Prosthetics Research*, 1977.

[254] Stereolabs. ZED Mini Stereo Camera. `https://www.stereolabs.com/products/zed-mini`, 2024.

[255] Priya Sundaresan, Suneel Belkhale, and Dorsa Sadigh. Learning Visuo-Haptic Skewering Strategies for Robot-Assisted Feeding. In *Conference on Robot Learning*, pages 332–341. PMLR, 2023.

[256] Priya Sundaresan, Jiajun Wu, and Dorsa Sadigh. Learning Sequential Acquisition Policies for Robot-Assisted Feeding. In *Conference on Robot Learning*, pages 1282–1299. PMLR, 2023.

[257] JaYoung Sung, Rebecca E. Grinter, and Henrik I. Christensen. Domestic Robot Ecology: An Initial Framework to Unpack Long-Term Acceptance of Robots at Home. *International Journal of Social Robotics*, 2(4):417–429, 2010. doi: 10.1007/s12369-010-0065-8.

[258] Atieh Taheri and Ziv Weissman. Designing a First Person Shooter Game for Quadriplegics. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–6, 2021.

[259] Yen-Ling Tai, Yu Chien Chiu, Yu-Wei Chao, and Yi-Ting Chen. SCONE: A Food Scooping Robot Learning Framework with Active Perception. In *Conference on Robot Learning*, pages 849–865. PMLR, 2023.

[260] Danielle M. Taylor. Americans with Disabilities: 2014. Current Population Reports P70-152, U.S. Census Bureau, Washington, DC, 2018.

[261] Jakob Thumm, Christopher Agia, Marco Pavone, and Matthias Althoff. Text2Interaction: Establishing Safe and Preferable Human-Robot Interaction. In *Conference on Robot Learning*, 2024.

[262] Mike Topping. An Overview of the Development of Handy 1, a Rehabilitation Robot to Assist the Severely Disabled. *Journal of Intelligent and Robotic Systems*, 34(3):253–263, 2002. doi: 10.1023/A:1016355418817.

[263] Chris Towell, Matthew Howard, and Sethu Vijayakumar. Learning Nullspace Policies. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 241–248. IEEE, 2010.

[264] Katherine Tsui, Holly Yanco, David Kontak, and Linda Beliveau. Development and Evaluation of a Flexible Interface for a Wheelchair Mounted Robotic Arm. In *ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2008.

[265] Emma M. van Zoelen, Karel van den Bosch, and Mark Neerincx. Becoming Team Members: Identifying Interaction Patterns of Mutual Adaptation for Human-Robot Co-Learning. *Frontiers in Robotics and AI*, 8:692811, 2021. doi: 10.3389/frobt.2021.692811.

[266] Shivam Vats, Michelle Zhao, Patrick Callaghan, Mingxi Jia, Maxim Likhachev, Oliver Kroemer, and George Konidaris. Optimal Interactive Learning on the Job via Facility Location Planning. *arXiv preprint arXiv:2505.00490*, 2025.

[267] Vention. Vention Stand, 2025. URL `https://vention.io/designs/heavy-duty-robot-pedestal-mobile-130889`. (Accessed: 1st January, 2025).

[268] Akifumi Wachi, Xun Shen, and Yanan Sui. A Survey of Constraint Formulations in Safe Reinforcement Learning. *International Joint Conference on Artificial Intelligence*, 2024.

[269] Huaxiaoyue Wang, Nathaniel Chin, Gonzalo Gonzalez-Pumariega, Xiangwan Sun, Neha Sunkara, Maximus Adrian Pace, Jeannette Bohg, and Sanjiban Choudhury. APRICOT: Active Preference Learning and Constraint-Aware Task Planning with LLMs. In *Conference on Robot Learning (CoRL)*, 2024.

[270] Junxiang Wang, Cindy Wang, Rana Soltani Zarrin, and Zackory Erickson. Bidirectional Human-Robot Communication for Physical Human-Robot Interaction. In *ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2026.

[271] Mei Wang, Weihong Deng, Jiani Hu, Xunqiang Tao, and Yaohai Huang. Racial Faces in the Wild: Reducing Racial Bias by Information Maximization Adaptation Network. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 692–702, 2019.

[272] Ruiqi Wang, Dezhong Zhao, Dayoon Suh, Ziqin Yuan, Guohua Chen, and Byung-Cheol Min. Personalization in Human-Robot Interaction through Preference-Based Action Representation Learning. *arXiv preprint*

arXiv:2409.13822, 2024.

[273] Shengye Wang and Henrik I Christensen. TritonBot: First Lessons Learned from Deployment of a Long-Term Autonomy Tour Guide Robot. In *2018 27th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pages 158–165. IEEE, 2018.

[274] Zi Wang, Caelan Reed Garrett, Leslie Pack Kaelbling, and Tomás Lozano-Pérez. Learning Compositional Models of Robot Skills for Task and Motion Planning. *The International Journal of Robotics Research*, 40(6-7):866–894, 2021.

[275] Tianxin Wei, Bowen Jin, Ruirui Li, Hansi Zeng, Zhengyang Wang, Jianhui Sun, Qingyu Yin, Hanqing Lu, Suhang Wang, Jingrui He, et al. Towards Unified Multi-Modal Personalization: Large Vision-Language Models for Generative Recommendation and Beyond. *arXiv preprint arXiv:2403.10667*, 2024.

[276] Nils Wilde, Dana Kulić, and Stephen L Smith. Learning User Preferences in Robot Motion Planning through Interaction. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 619–626. IEEE, 2018.

[277] Alan FT Winfield, Serena Booth, Louise A Dennis, Takashi Egawa, Helen Hastie, Naomi Jacobs, Roderick I Muttram, Joanna I Olszewska, Fahimeh Rajabiyazdi, Andreas Theodorou, et al. IEEE P7001: A Proposed Standard on Transparency. *Frontiers in Robotics and AI*, 8:665729, 2021.

[278] World Health Organization. *International Classification of Functioning, Disability, and Health: Children & Youth Version: ICF-CY*. World Health Organization, 2007.

[279] World Health Organization and World Bank. *World Report on Disability*. World Health Organization, Geneva, 2011. ISBN 9789241564182.

[280] Jimmy Wu, Rika Antonova, Adam Kan, Marion Lepert, Andy Zeng, Shuran Song, Jeannette Bohg, Szymon Rusinkiewicz, and Thomas Funkhouser. TidyBot: Personalized Robot Assistance with Large Language Models. *arXiv preprint arXiv:2305.05658*, 2023.

[281] Guang Yang, Shuoyu Wang, Junyou Yang, and Peng Shi. Desire-Driven Reasoning Considering Personalized Care Preferences. *Transactions on Systems, Man, and Cybernetics: Systems*, 2021.

[282] Heng Yang, Jingnan Shi, and Luca Carlone. TEASER: Fast and Certifiable Point Cloud Registration. *IEEE Transactions on Robotics*, 37(2):314–333,

2020.

[283] Jehan Yang, Eleanor Hodgson, Cindy Sun, Zackory Erickson, and Douglas J. Weber. Bimanual High-Density EMG Control for In-Home Mobile Manipulation by Users with Quadriplegia, 2026. URL https://arxiv.org/abs/2602.02773.

[284] Quantao Yang, Johannes A Stork, and Todor Stoyanov. Null Space Based Efficient Reinforcement Learning with Hierarchical Safety Constraints. In *2021 European Conference on Mobile Robots (ECMR)*, pages 1–6. IEEE, 2021.

[285] Ruolin Ye, Wenqiang Xu, Haoyuan Fu, Rajat Kumar Jenamani, Vy Nguyen, Cewu Lu, Katherine Dimitropoulou, and Tapomayukh Bhattacharjee. RCareWorld: A Human-centric Simulation World for Caregiving Robots. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 33–40, 2022. doi: 10.1109/IROS47612.2022.9982244.

[286] Uksang Yoo, Nathaniel Dennler, Eliot Xing, Maja Matarić, Stefanos Nikolaidis, Jeffrey Ichnowski, and Jean Oh. Soft and Compliant Contact-Rich Hair Manipulation and Care. *arXiv preprint arXiv:2501.02630*, 2025.

[287] Evan You. Vue.js, 2025. URL https://vuejs.org. (Accessed: 1st January, 2025).

[288] YOUCOX. Anti Slip Adjustable Mug Holder, 2025. URL https://www.amazon.com/YOUCOX-Anti-Slip-Adjustable-Tumbler-SUNWILL/dp/B0936XNR2C. (Accessed: 1st January, 2025).

[289] Wenhao Yu, Nimrod Gileadi, Chuyuan Fu, Sean Kirmani, Kuang-Huei Lee, Montse Gonzalez Arenas, Hao-Tien Lewis Chiang, Tom Erez, Leonard Hasenclever, Jan Humplik, et al. Language to Rewards for Robotic Skill Synthesis. *arXiv preprint arXiv:2306.08647*, 2023.

[290] Rachel Zenker, Connor Pardell, Andrea Gilmore-Bykovskyi, Amy JH Kind, and Nicole Werner. Exploring Workload among Informal Caregivers of Persons with Dementia. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 61(1):1297, 2017.

[291] George Zerveas, Srideepika Jayaraman, Dhaval Patel, Anuradha Bhamidipaty, and Carsten Eickhoff. A Transformer-Based Framework for Multivariate Time Series Representation Learning. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, KDD '21, pages 2114–2124, New York, NY, USA, 2021. Association for Computing Machinery. doi: 10.1145/3447548.3467401.

[292] Lihan Zha, Yuchen Cui, Li-Heng Lin, Minae Kwon, Montserrat Gonzalez Arenas, Andy Zeng, Fei Xia, and Dorsa Sadigh. Distilling and Retrieving Generalizable Knowledge for Robot Manipulation via Language Corrections. *arXiv preprint arXiv:2311.10678*, 2023.

[293] Fan Zhang, Antoine Cully, and Yiannis Demiris. Personalized Robot-Assisted Dressing Using User Modeling in Latent Spaces. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3603–3610. IEEE, 2017.

[294] Kevin Zhang, Mohit Sharma, Manuela Veloso, and Oliver Kroemer. Leveraging Multimodal Haptic Sensory Data for Robust Cutting. In *2019 IEEE-RAS 19th International Conference on Humanoid Robots (Humanoids)*, 2019.

[295] Junhao Zheng, Chengming Shi, Xidi Cai, Qiuke Li, Duzhen Zhang, Chenxing Li, Dong Yu, and Qianli Ma. Lifelong Learning of Large Language Model based Agents: A Roadmap. *arXiv preprint arXiv:2501.07278*, 2025.