Learning to Open New Doors

Ellen Klingbeil, Ashutosh Saxena, Andrew Y. Ng

***Abstract*— We consider the problem of enabling a robot to autonomously open doors, including novel ones that the robot has not previously seen. Given the large variation in the appearances and locations of doors and door handles, this is a challenging perception and control problem; but this capability will significantly enlarge the range of environments that our robots can autonomously navigate through. In this paper, we focus on the case of doors with door handles. We propose an approach that, rather than trying to build a full 3d model of the door/door handle—which is challenging because of occlusion, specularity of many door handles, and the limited accuracy of our 3d sensors—instead uses computer vision to choose a manipulation strategy. Specifically, it uses an image of the door handle to identify a small number of "3d key locations," such as the axis of rotation of the door handle, and the location of the end-point of the door-handle. These key locations then completely define a trajectory for the robot end-effector (hand) that successfully turns the door handle and opens the door. Evaluated on a large set of doors that the robot had not previously seen, it successfully opened 31 out of 34 doors. We also show that this approach of using vision to identify a small number of key locations also generalizes to a range of other tasks, including turning a thermostat knob, pulling open a drawer, and pushing elevator buttons.**

Fig. 1. STAIR robot opening a novel door and images of the types of doors used in our experiments.

I. INTRODUCTION

There is growing interest in using robots not only in controlled factory environments, but also in unstructured home and office environments. Although robots are able to successfully navigate open spaces even of novel environments, it is desirable that robots also be able to manipulate their environment to gain access to new spaces. In this paper, we propose an algorithm that enables a robot to open and navigate through doors, even ones that it has not previously seen.

Because of the wide range of variation in the shapes and appearances of doors, door handles, etc., it is a challenging perception and control problem to open a novel door. Further, because a door handle almost invariably occludes part of itself or part of the door (and is often specular or has little texture) using our robot's sensors to build a full 3d model of the door also proves challenging. Rather than build a full 3d model, we instead proposes an approach that uses computer vision to identify a small number of "3d key locations" or "3d features," that completely define a trajectory for the robot arm to open the door. More formally, these 3d features comprise some number of points in 3d (such as the location of the endpoint of the handle) as well as some vector directions (such as the direction in which to push the handle). We can then straightforwardly map from the 3d features to an arm trajectory.

Our robot identifies 3d locations on a door handle using computer vision and supervised learning. Specifically, we train an object detection algorithm to accurately place a bounding box around door handles. By incorporating an appropriate Bayesian prior into the system, our object detection algorithm also takes into account useful *contextual* information, such as that door handles are most likely found at a certain height. And further, the information that there are usually 1-2 door handles per door, and that pairs of door handles (as in a pair of double doors) are usually at the same height.

Because of mechanical limitations of our robot (limited strength of the arm), our work focuses primarily on doors with door handles (i.e., similar to the ones shown in Figure 1), and that open away from the robot. Our approach results in our robot robustly opening a wide range of doors, even ones very different from the training set. Tested on a large set of novel doors, our robot successfully opens 31 out of 34 doors. In addition, we show that our approach of identifying 3d features also generalizes to a range of other manipulation tasks, ones where a full 3d model of the object may not be available, but where a small number of 3d points (or vector directions) suffices to define a complete manipulation

E. Klingbeil is with Department of Aeronautics and Astronautics, Stanford University, Stanford, CA 94305, USA ellenrk7@stanford.edu

A. Saxena is with the Faculty of Computer Science, Cornell University, Ithica, NY 14853, USA asaxena@cs.cornell.edu

A. Ng is with the Faculty of Computer Science, Stanford University, Stanford, CA 94305, USA ang@cs.stanford.edu

trajectory. Specifically, we show that by using computer vision to identify 3d features, we can also carry out such tasks as turn a thermostat knob, push an elevator button, or pull open a drawer.

Much prior work in manipulation tasks such as grasping and door opening assumes that an accurate 3d model of the object is available and focuses more on developing the control actions required to manipulate the single known object ([1], [2], [3]). Recently some researchers have developed control strategies that allow a robot to manipulate unknown doors [4], but these assume the robot has already accurately located the unknown door handle and established an appropriate grasp. When entering a novel building, a robot must rely on only its sensors to perform manipulation, but many modern 3d imaging sensors (such as a laser range finder, time-of-flight depth camera, or stereo camera) often provide sparse and noisy point clouds.

In this paper, we focus on the problem of manipulation in novel environments where an accurate 3d model of the object is not available and its location is unknown. Some recent efforts in grasping [5] use learning to address this problem. However, their system identifies only a single grasp point, which is not sufficient information for the more complex task of opening a door, where the appropriate grasp and subsequent control actions depend on the type of object and the task to be performed.

We use a mobile manipulation platform, where we integrate vision, navigation, planning, and control, to perform the task of opening the door. We validate our perception algorithms in a set of experiments, where the robot was able to autonomously open novel doors in 31/34 trials. We also evaluate the extensibility of our algorithms by applying the perception algorithms, trained on a different data sets for each task to enable our robot to perform the additional manipulation tasks of pressing an elevator call button, pulling out a drawer, and turning a thermostat knob.

II. RELATED WORK

In robotic manipulation, much effort has been focused on developing control actions for manipulation tasks assuming complete knowledge of the environment in the form of an accurate 3d model of an object, including its position and orientation ([1], [2], [3]). Nagatani and Yuta [6] designed a door opening control strategy for a mobile robot on a single door; however they provide the robot with the handle location in advance. Petersson et al. [7] looked at developing compliant control actions and performing online estimation of the center of rotation and radius for a single door. Schmid et al. [4] demonstrated their manipulator opening a door without knowledge of the door geometry, using a combination of tactile sensing in the fingers and force/torque sensing in the wrist. However, in both of these works a human positioned the robot end-effector on the handle manually. Jain et al. [8] demonstrate their robot opening novel doors using a force-sensing manipulator to infer how to turn the handle once a human has pointed out the location of the handle. For a fully autonomous system, the perception system must be capable of locating the object and inferring some additional information, since a push-bar or spherical door knob must be manipulated much differently than a turn lever door handle.

Some researchers address the problem of identifying the 3d location of an object through human assistance. Kemp et al. [9] consider having a human use a laser pointer to specify the 3d position of objects for the robot to grasp. Other researchers have recently considered using vision-based learning techniques to enable robots to grasp unknown objects [10]. However, since only a single grasp point is identified, these algorithms are insufficient for more complex manipulation tasks, such as opening a door, where the appropriate grasp and subsequent control actions depend on the type of object and the task to be performed.

In contrast to many of these previous endeavors, our work does not assume the existence of a known, 3d model of the door handle or knowledge of its location. However we do assume that the robot is provided with a rough 2d map of the building which allows it to navigate to a door (with an error of standard navigation algorithms). We focus on the problem of autonomous manipulation in novel environments where the robot must rely on its onboard sensor data only. We do not attempt to reconstruct a 3d model of the object, but instead recognize the object and identify only the small set of 3d features sufficient to compute the required control actions.

The difficult task of recognizing objects in an image is a large area of ongoing research. A number of algorithms have been developed that achieve good performance on object recognition [11]. However, since scenes are often cluttered with objects (or portions of objects) that look very similar to the object of interest, these 2d image based vision algorithms tend to have lots of false positives. To locate objects in the world, a human often uses prior knowledge about where certain objects tend to be located. For example, a door handle usually lies on a vertical plane at some distance above the floor. Using contextual cues (for example, the location-based priors mentioned above) has been shown to improve object detection [12].

III. APPROACH

To open a door, we must first locate the door handle and infer how to manipulate it. We approach the task of door opening by using vision and machine learning to identify the door handle and its type (whether it turns clockwise or anti-clockwise), and to identify a small set of 3d features which map straightforwardly to a trajectory for the arm. Concretely, to reliably open a door, we identify three 3d features, comprising two 3d points $p_1, p_2 \in \Re^3$, and a unit-length vector $v_1 \in \Re^3$ ($||v_1|| = 1$). These are illustrated in Figure 2(a). Concretely:

- The point p_1 is the surface location of the handle at the axis of rotation (where the keyhole would be, if the door has one).
- The vector v_1 indicates the axis of rotation of the door handle. Further, v_1 is usually the normal to the plane

TABLE I

3D FEATURES FOR SEVERAL MANIPULATION TASKS.

MANIPULATION TASK	DESCRIPTORS
OPEN A DOOR	1. LOCATION OF ROTATION AXIS OF HANDLE p_1 2. AXIS OF ROTATION v_1 3. LOCATION ALONG HANDLE AT WHICH TO PUSH DOWN p_2
PRESS AN ELEVATOR BUTTON	1. LOCATION OF BUTTON p_1 2. DIRECTION TO PRESS v_1
PULL OPEN A DRAWER	1. MIDPOINT OF THE HANDLE p_1 2. DIRECTION TO PULL v_1
TURN A THERMOSTAT KNOB	1. MIDPOINT OF THE KNOB p_1 2. AXIS OF ROTATION v_1

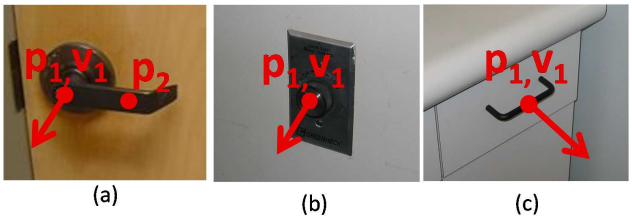

Fig. 2. 3d point/vector pairs extracted from the perception algorithm which, along with the action classification, define control actions to manipulate a (a)door handle, (b)thermostat knob, (c)drawer.

of the door; and the direction $-v_1$ usually specifies the direction in which to push the door (see Figure 2(a)).

- The point p_2 is an appropriate location on the door handle to push down on.

These 3d features can be straightforwardly used to parameterize a trajectory for the arm to open the door. Even if no 3d model of the door is available, as long as our algorithm is able to find p_1, v_1, p_2 accurately, we can robustly open a door. Specifically, we map from these 3d features to a series of waypoints, each consisting of a Cartesian position and orientation of the robot end-effector; we then command our position-controlled end-effector through these waypoints.

In our experiments, we consider only doors which open away from the robot due to mechanical limitations and lack of sensing in our manipulator gripper. We note that to apply these ideas to doors that open towards the robot, one may additionally have to estimate the location of the door hinge axis.

We propose that the idea of extracting control actions from only a small set of 3d features identified by the perception algorithm can also be extended to other manipulation tasks (Table I). For example, to turn a thermostat knob, we typically require knowledge of the position of the center of the knob p_1, and the axis of rotation v_1. (See Figure 2(b).) To pull open a drawer, we need to know the location of a drawer handle p_1, and the direction in which to pull v_2 (Figure 2(c).)

A. Object Classifier

In order to open a previously unknown door, we must first detect the door handle. This is challenging, because door handles can vary significantly in appearance (see Figure 3). We use computer vision and a state of the art 2d-sliding window object detection algorithm due to Torralba et al. [13] and implemented by [14].[1]

The sliding window detector samples rectangular shaped regions in the image, and tries to determine if each rectangular region roughly bounds a door handle. It then outputs the detections with confidence values greater than a specified threshold. A cross validation set was used to determine the classifier threshold. Because the sliding window detector uses only the information inside a rectangular image region to identify possible door handles, it thus ignores the rest of the image outside that rectangle, which may also have useful "contextual" information for detecting the handle. In our experiments, simply applying a sliding window detector (with non-maximal suppression) without considering any contextual information results in many false positives—12.2% on the training set and 15.6% on the test set.

To improve performance over this classifier, we incorporate three additional types of contextual information into the classifier:

1) Location. E.g., a door handle is most likely to be found at a certain range of heights above the floor.
2) Expected number of handles in the image (at least 1 but no more than 2, the latter in the case of double doors),
3) Expected relative location of 2 handles (in the case of double doors, the two door handles are almost always at the same height).

To incorporate the first contextual cue, each pixel in the image is assigned a prior probability of containing a door handle. This probability distribution is estimated from the training data (using maximum likelihood estimation with Laplace smoothing). Thus, pixels near the bottom (points near the floor) have a small prior probability of containing a door, whereas points corresponding to approximately waist height are assigned a higher probability of containing a door. This location-based prior is used to re-weight the probability for each detection, thus giving us a posterior probability for an image region containing a door, that takes into account location in the image.

Because the sliding window detector returns many detections (see Figure 3; each green rectangle is a separate detection reported by the sliding window algorithm), we also post-process the detector output using K-means clustering on these detections. Here, the number of clusters k is set to be the number of detections remaining after applying non-maximal suppression, and each cluster is assigned a weight

[1]The supervised training procedure produces a dictionary of features consisting of localized templates from cropped training examples. The relevance of each of these features in identifying whether an object is present in an image patch is evaluated by computing the normalized cross correlation of each template with the image patch. A binary classifier is learned by using boosted decision trees to select the set of feature patches which are most effective at recognizing the object in the training examples (see [13] for more details). We then use this classifier within a sliding-window detector framework to compute the probability of the presence of the candidate object within each rectangular subwindow of a test image.

Fig. 3. Test set results with raw classifier output outlined in green, and the result of applying context and clustering outlined in blue. Notice how the bounding boxes are very accurately centered on the handle. (Best viewed in color)

proportional to the sum of the confidences of the detections associated with it. The clustering results in a bounding box that is better localized on the center of the handle.

We explored a number of algorithms for taking into account the second and third contextual cues described above. The best method turned out to be relatively simple. We choose the cluster centroid with the highest weight from K-means as the most likely candidate for a handle and search in the horizontal band containing this detection to locate the next most likely candidate. If a second high-confidence detection is found, then we identify that as a second door handle; otherwise, we return only the first detection. This incorporates the primary knowledge that we expect at most 2 door handles, and that if there are 2 handles, they should occur at the same height. This characterizes all double doors that occured in our training set.[2]

Given the rectangular patch(es) containing a door handle, we use a second trained binary classifier to recognize the type of handle (specifically, whether it turns clockwise or anti-clockwise). [3] This indicates the desired action — push down and rotate clockwise or push down and rotate anti-clockwise. The object location and clockwise/anti-clockwise classification provides sufficient information to extract a small set of 3d features sufficient to compute control actions, as described in the next section. Figure 3 shows images of several doors (some with very cluttered backgrounds) and the resulting detections produced by our algorithm. The blue rectangles are the final output of our door handle detection algorithm.

[2]One can also envision a more complex algorithm that increases the confidence of detections which lie in close proximity to the left or right of each other. However, since there are often horizontal edge features on doors (due to windows, for example) which produce a large number of neighboring detections along a horizontal band, this approach overly increased the weight of these false positives, and gave poor performance.

[3]We resize the image patches containing the door handle to a fixed size and train a sliding window classifier. The positive and negative training examples are the left and right oriented handles, respectively.

B. Extracting 3d Features

Once our perception algorithm has identified the location of the handle in an image and classified the action (clockwise/anti-clockwise) to perform, the next step is to infer the control actions to execute the task. Concretely, we will identify the 3d features p_1, v_1, p_2, and use them to infer a trajectory for the arm.

In detail, we use the results from the vision-based classifier, along with a 2d laser scan of the door plane to extract the location of the axis of rotation of the door handle, the location to apply force, and the door plane normal. As seen in Figure 3, the object recognizer provides a tight, well-centered bounding box around the handle and a classification for the direction the handle rotates. Given a bounding box surrounding the door handle, we determine p_1 and p_2 using a fixed pair of points specified relative to the corners of the bounding box. In detail, if (x, y) is the top left corner, w is the width, and h is the height of the bounding box in image coordinates, the handle rotational axis and location to apply force are approximately located at $(x + \alpha w, y + \beta h)$ and $(x + \gamma w, y + \beta h)$, respectively, where α, β, γ are empirically chosen constants. Since the robot can localize well with respect to the door and its camera is rigidly mounted, this heuristic performed well in our experiments.

The method described above gives a set of 2d coordinates in the image plane. These 2d coordinates are transformed into

3d coordinates in the world frame using our laser scanner and the approach described in section III-B.1. Specifically, our robot uses a Hokuyo URG scanner that is mounted horizontally. Using the laser scanner output, we can extract the door plane and normal to the door plane. Making the assumption that the door is a large vertical plane (and that the door handle is a fixed distance offset from the door, usually about 4cm), we can identify the 3d locations of p_1 and p_2. Figure 2 shows the 3d points extracted from the location and action classification for a door handle (and additionally a thermostat knob and drawer). The normal of the door plane, extracted from the laser data (and assuming a vertical door plane), is used as v_1.

1) Data Fusion: We use a 2d laser scan (in the horizontal plane) along with a vertical-wall assumption to obtain the 3d location of 2d points in the image. [4] Using the vertical-wall assumption, we want to find the intersection of the camera ray passing through the door handle with the door plane. If $v \in \Re^3$ is the unit ray which passes from the camera origin through the predicted location of the door handle in the image plane, t is the distance from the camera origin to the door handle, and $T_0 \in \Re^{2\times 3}$ is a matrix which projects the camera ray into the plane of the laser, then the camera ray projected into the laser plane is defined by $r = t(T_0 v)$. We denote the planar laser readings as $l_i = (x(\theta_i), y(\theta_i))$ and the location of the laser relative to the camera in the horizontal plane as d. We must solve for the location where the camera ray intersects the door plane.

$$t^* = \min_t \sum_{i\in\Psi} ||T_0(tv) - (l_i + d)||_2^2 \qquad (1)$$

where Ψ is a small neighborhood around the ray r. The solution provides the location of the handle in the camera frame, $r^* = t^*(T_0 v)$. Finally, the 3d location in the robot's coordinate frame is found by applying the coordinate transform, $T_1 \in \Re^{4\times 4}$, from the camera frame to the robot arm frame, $p = T_1 r^*$.

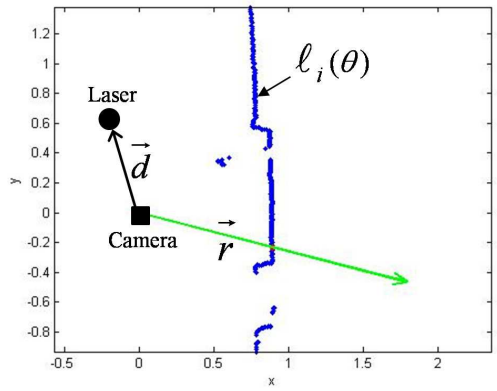

Fig. 4. Example of a 2d laser scan with the camera ray projected into the horizontal plane.

[4]We do not use a stereo camera because we found that the 3d point cloud obtained from the sensor was very noisy and sparse, often resulting in a large area of erroneous depth values in the center of the handle.

C. Mapping to Control Actions

Given the 3d features corresponding to the location of the door handle axis of rotation p_1, the location to apply force p_2, and the normal vector to the door plane v_1, as well as the action classification (push down and rotate anti-clockwise or clockwise), we need to extract 3d waypoints which can be used to plan a path in cartesian space for the end-effector. For our position-controlled robot to turn the door handle, the robot must establish contact with the handle and push down on the handle by moving the end-effector along an arc (which is centered on the handle axis of rotation and lying in the plane normal to this axis) while at the same time rotating the wrist (Figure 5). The 3D features are used to geometrically compute this arc, and the correspondiong rotation of the wrist. The handle axis of rotation provides the preferred pitch angle for the wrist (aligned with the door plane normal vector). The normal to the door plane and the location of the handle on the door (left or right side) indicates which direction the door will swing (left or right) so the robot end-effector knows approximately which direction to push the door open.

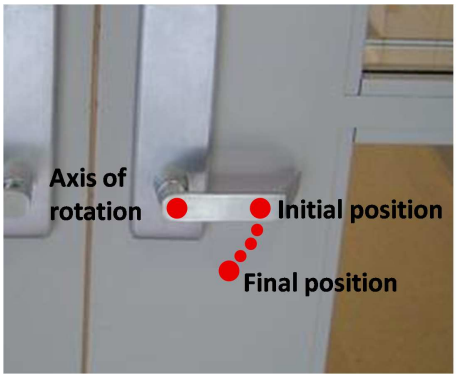

Fig. 5. An illustration of the end effector trajectory computed for the robot to manipulate a door handle.

We can compute end-effector trajectories for the additional tasks we consider (pressing an elevator button, turning a knob, and pulling a drawer) in a similar manner as the door opening task. For pressing an elevator button, the 3d location of the center of the button p_1 and the surface normal v_1 define a straight-line trajectory for the robot end-effector to move along (preferably with the wrist aligned along the normal) to make solid contact in the center of the button. In the case of pulling out a drawer, the location of the center of the handle p_1 and the surface normal v_1 provide enough information to compute a trajectory for the robot end-effector to grasp the handle and pull along a straight-line trajectory with the wrist approximately aligned with the direction of motion. For a thermostat knob, the 3d location of the center of the knob p_1 and the rotation axis v_1 defines the end-effector location and wrist orientation for the robot to grasp the knob.

Given the waypoints consisting of 3d cartesian positions and wrist orientations, we transform the trajectory into configuration space and use a Probabilistic RoadMap (PRM) [15] motion planning algorithm to generate smooth

paths which execute the task while avoiding obstacles and joint limits.

IV. PERCEPTION RESULTS

To train and test our algorithm, we use a diverse set of 335 images of doors taken with a digital camera. To evaluate the performance of our vision algorithms, we randomly split the data into 270 images for training and 65 images for testing.

The door handle training set consisted of 292 positive examples (some images had double doors) and 10,000 negative examples. Table II shows the recognition and localization accuracies. Recognition is considered correct if the area of overlap between the predicted detection bounding box and the ground truth bounding box exceeds 50% (which is standard practice in object recognition [16]). However, manipulation tasks require higher accuracy in localizing the object to be manipulated. Thus, a door handle was considered to be correctly localized if its estimated location was within 2 cm of the correct location, since an error larger than this would cause our robot arm to fail to grasp the door handle and open the door. The accuracy of the classifier that distinguishes left-turn from right-turn handles was 97.3%.

TABLE II
DOOR HANDLE CLASSIFICATION ACCURACY.

	RECOGNITION	LOCALIZATION
DOOR HANDLE	94.5%	93.2%

V. EXPERIMENTAL RESULTS

We demonstrate our perception algorithms on a mobile robot platform in a number of experiments. In the door opening experiments, the robot was required to autonomously locate the handle and push the door open in order for the trial to be considered a success.

A. Hardware

Our robotic platform consists of a Neuronics Katana450 5-DOF robotic arm, with angle gripper, mounted on a Segway base. Our vision system consists of a pan-tilt-zoom (Sony DV100) camera with a 2d laser scanner (Hokuyo). We use the ROS software framework to integrate all the software and hardware components of our system [17].

B. Door Opening

We tested our algorithm by having the robot autonomously open doors in two different buildings on a total of five different floors (approximately 20 different doors). In the experiments, our robot saw all of the test locations for the first time. The training images for our vision-based learning algorithm were collected in completely separate buildings, with different doors and door handle shapes, structure, decoration, and ambient lighting. Test cases were run with the robot placed within reach of the handle and at different angles with respect to the door, (typically between +/- 20 degrees of facing the door) in order to be robust to maximum positioning errors expected from a standard 2d map-based navigation algorithm.

Our robot was able to successfully open the door 31 times out of 34 trials. Notable failures among the test cases included glass doors (erroneous laser readings), doors with numeric keypads, and very dim/poor lighting conditions.

Our experiments discussed above only included doors where the robot could push the door open. Pulling the door open is much more difficult for our robotic arm because of the small effective workspace in which it can exert enough torque to open a door and its fairly weak gripper. Additionally, many of our doors are spring-loaded, making it impossible for this particular arm to pull open these doors. Thus, we have only performed a single demonstration of our robot pulling open one door which was light and not spring-loaded.

To demonstrate our robot navigating to an unknown door using a rough map of the building and autonomously opening the door, we integrate a standard 2d navigation system with our vision, planning, and control [17]. The navigation localized the robot within 10 cm of the center of the door and within ± 10 degrees of facing the door. Our vision-based classifiers identify the 3d position of the door handle and command the robot navigation system to drive up to the handle. Then the vision-based classifiers infer the desired 3d positions and orientations for the arm to perform the task of opening the door. The robot was able to navigate into 3 different offices in an uninterupted sequence, by opening doors not seen in the training set.

C. Additional Manipulation Tasks

To demonstrate that our ideas can be easily extended to more manipulation tasks, we tested our algorithms on three additional tasks—pressing an elevator call button, turning a thermostat knob, and pulling out a drawer. From our vision-based classifiers, we identify a small set of 3d features for each task (see Table I) and use these to infer a set of 3d waypoints to plan a path for the robot end-effector. [5] In experiments where the robot was commanded to operate novel elevator call buttons, the robot succeeded in pressing the button in 22/24 trials. In single demonstrations, the robot was also able to sucessfully turn the knob and pull out the drawer.

See Figure 6 for snapshots of the robot experiments. Videos of these experiments are available at:

http://www.stanford.edu/∼ellenrk7/OpeningNewDoors

VI. CONCLUSION

To navigate and perform tasks in unstructured environments, robots must be able to perceive their environments to identify what objects to manipulate and how they can

[5]The elevator call button training set consisted of approximately 400 positive and 1500 negative samples. Due to difficulty in collecting sufficient training data, the image-based classifiers for the drawer and thermostat knob were trained and tested on the same objects but the robot autonomously located the object, classified the appropriate action, and planned the path to manipulate it.

TABLE III

ERROR RATES FOR ROBOT OPENING DOORS IN TOTAL OF 34 TRIALS.

HANDLE TYPE	NUMBER OF TRIALS	LOCALIZATION (%)	TYPE CLASSIFICATION (%)	SUCCESS-RATE
ANTI-CLOCKWISE	19	89.5%	94.7%	84.2%
CLOCKWISE	15	100%	100%	100%
TOTAL	34	94.1%	97.1%	91.2%

Fig. 6. Snapshots of our robot opening doors and performing several other manipulation tasks.

be manipulated to perform the desired tasks. We present an approach that uses state of the art computer vision and supervised learning to identify a small number of 3d features that are sufficient to compute a trajectory for the robot end-effector to turn the handle and open the door. This strategy enabled our robot to navigate to unexplored locations in an unfamiliar building by opening 31/34 doors it had not seen before. Our robot was also able to demonstrate the application of this approach to perform several other manipulation tasks.

We believe our approach is complementary to those which focus more on developing compliant controllers with force feedback to deal with uncertainty. Tactile sensors in the gripper or a wrist-mounted force sensor could provide supplementary information to better estimate some of the 3D features once the robot has made contact with the object to be manipulated.

VII. ACKNOWLEDGMENTS

We thank Andrei Iancu and Srinivasa Rangan for their help in the experiments.

REFERENCES

[1] A. Bicchi and V. Kumar, "Robotic grasping and contact: a review," in *ICRA*, 2000.

[2] M. Prats, P. Sanz, and A. P. del Pobil, "Task planning for intelligent robot manipulation," in *IASTED Artificial Intel App*, 2007.

[3] A. Petrovskaya and A. Y. Ng, "Probabilistic mobile manipulation in dynamic environments, with application to opening doors," in *IJCAI*, 2007.

[4] A. J. Schmid, N. Gorges, D. Goger, and H. Worn, "Opening a door with a humanoid robot using multi-sensory tactile feedback," in *ICRA*, 2008.

[5] A. Saxena, J. Driemeyer, and A. Y. Ng, "Robotic grasping of novel objects using vision," *IJRR*, vol. 27, no. 2, pp. 157–173, 2008.

[6] K. Nagatani and S. Yuta, "Designing strategy and implementation of mobile manipulator control system for opening door," in *ICRA*, 1996.

[7] L. Petersson, D. Austin, and D. Kragic, "High-level control of a mobile manipulator for door opening," in *IROS*, 2000.

[8] A. Jain and C. C. Kemp, "Behaviors for robust door opening and doorway traversal with a force-sensing mobile manipulator," in *RSS Workshop on Robotic Manipulation: Intelligence in Human Environments*, 2008.

[9] C. C. Kemp, C. Anderson, H. Nguyen, A. Trevor, and Z. Xu, "A point-and-click interface for the real world: Laser designation of objects for mobile manipulation," in *HRI*, 2008.

[10] A. Saxena, L. Wong, and A. Y. Ng, "Learning grasp strategies with partial shape information," in *AAAI*, 2008.

[11] T. Serre, L. Wolf, and T. Poggio, "Object recognition with features inspired by visual cortex," in *CVPR*, 2005.

[12] A. Oliva and A. Torralba, "The role of context in object recognition," *Trends in Cogn Sci*, Nov. 2007.

[13] A. Torralba, "Contextual priming for object detection," *IJCV*, vol. 53, no. 2, pp. 169–191, 2003.

[14] S. Gould, O. Russakovsky, I. Goodfellow, P. Baumstarck, A. Y. Ng, and D. Koller. (2008) The STAIR vision library (v2.1). [Online]. Available: http://ai.stanford.edu/ sgould/svl

[15] F. Schwarzer, M. Saha, and J.-C. Latombe, "Adaptive dynamic collision checking for single and multiple articulated robots in complex environments," *IEEE Tr Robotics Auto*, vol. 21, no. 3, pp. 338–353, 2005.

[16] M. Everingham and A. Zisserman, "The pascal visual object classes challenge 2006 (voc2006) results," 2006. [Online]. Available: http://eprints.pascal-network.org/archive/00002404

[17] The ROS (Robot Operating System) framework is an open-source, peer-to-peer, cross-platform message-passing system being jointly developed by Stanford University and Willow Garage. The ROS distribution wraps many popular code bases, the navigation stack from Player, and provides drivers to various pieces of robotics hardware. [Online]. Available: http://pr.willowgarage.com/wiki/ROS