Computer Vision with Kinect

CS 7670

Zhaoyin Jia

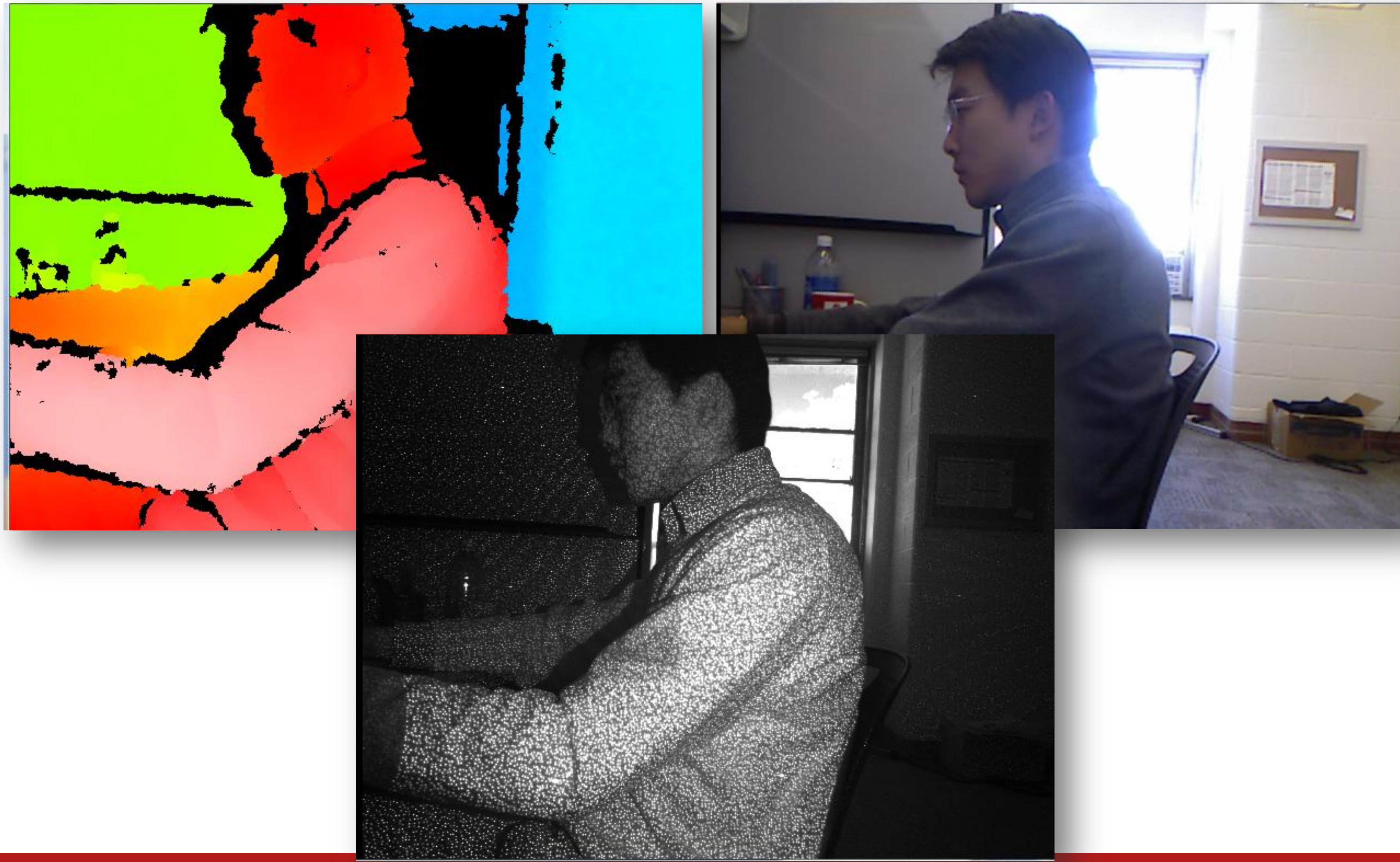
Fall, 2011

Kinect

- $150 color image and 3D sensor

- An Infrared projector
- A color camera
- An Infrared sensor

Kinect



Kinect

- Official SDK from Microsoft released on Jun 16th
- Better depth image and alignment, Skeleton tracking
  - *Real-time Human Pose Recognition in Parts from Single Depth Images*. Jamie Shotton, et.al, CVPR 2011, (Best paper award).
- Online hacks: OpenNI, open-kinect

Resources

- Real time capturing depth and color image
- Microsoft SDK gives better alignment.
- Online calibration toolbox available
  - http://www.ee.oulu.fi/~dherrera/kinect/

Topics

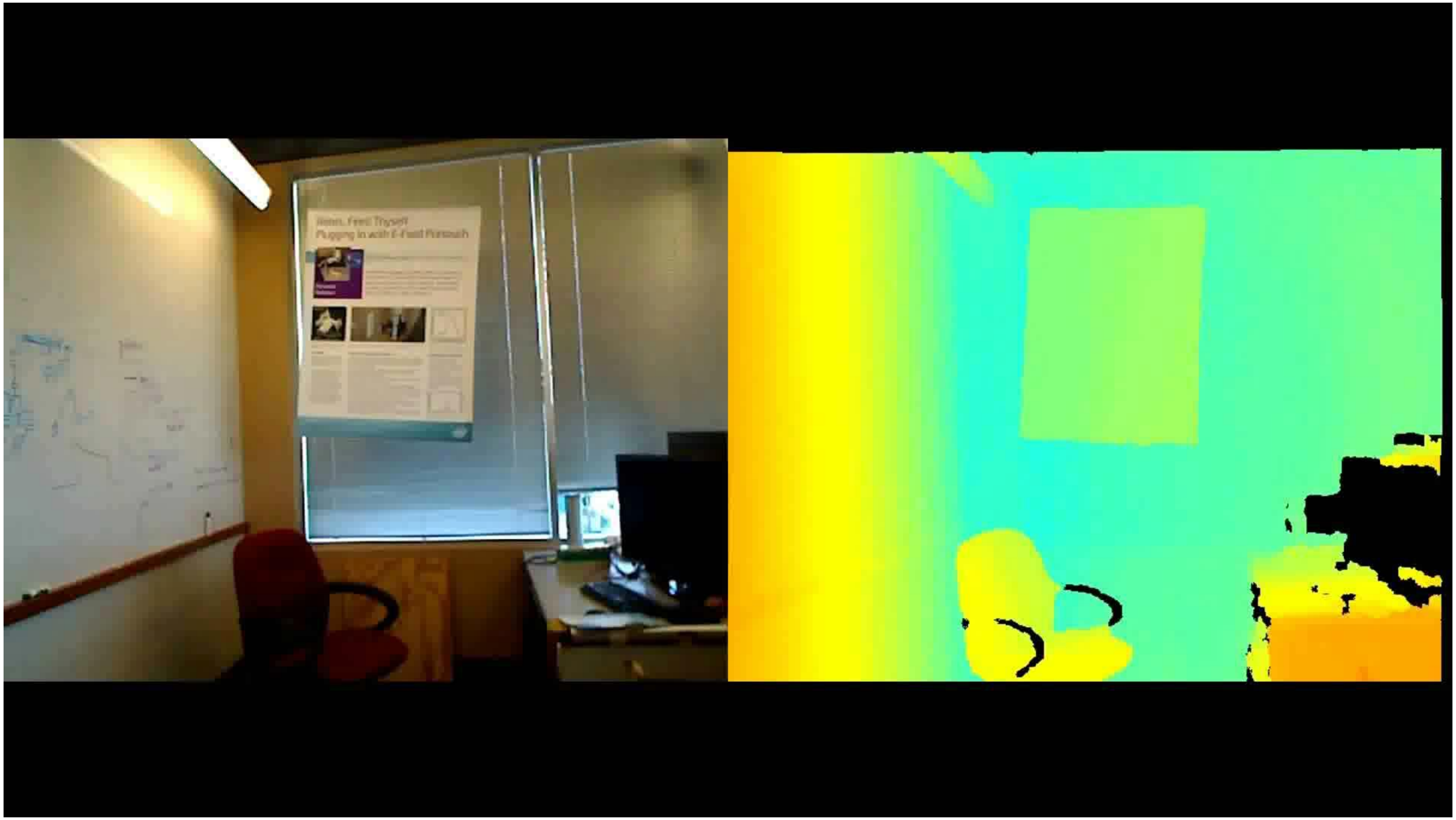
- RGB-D Mapping
- Robotics grasping
- Object recognition
- Human tracking

RGB-D Mapping

- Align the “frames” from a Kinect to create a single 3D map (or model) of the environment

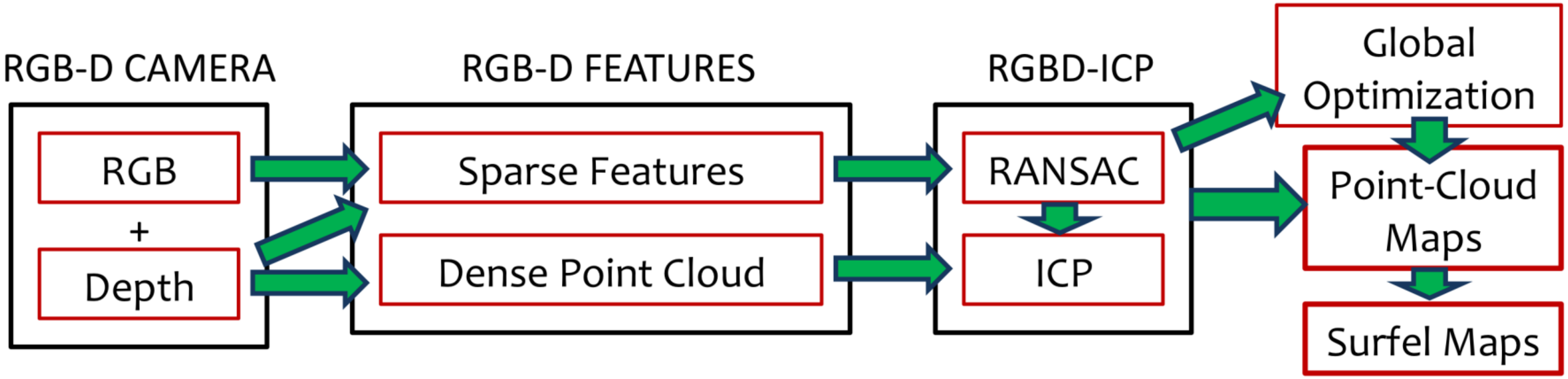
RGB-D Mapping: Using depth cameras for dense 3D modeling of indoor environments. Henry, Krainin, Herbst, Ren, Fox. ISER 2010.

RGB-D Mapping



640x480, 30Hz, color + dense depth

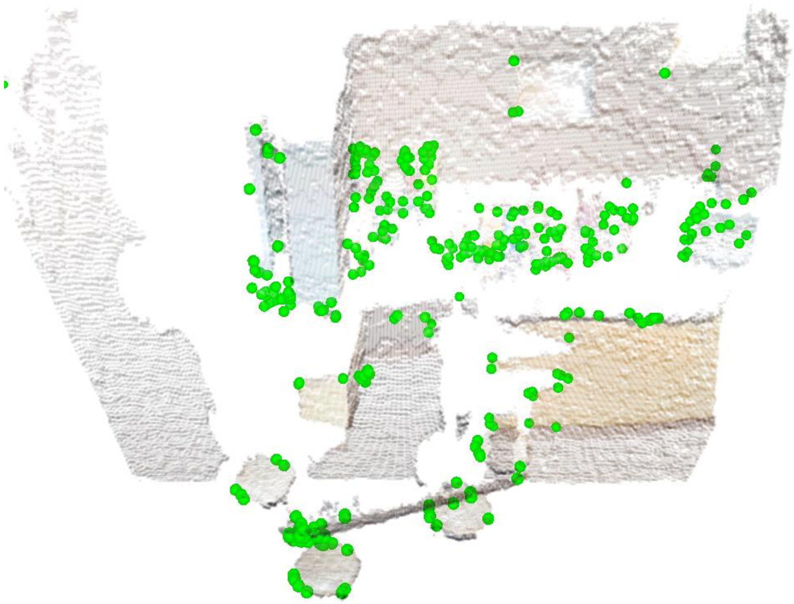
System Overview



- Frame-to-frame alignment
- Global Optimization (SBA for Loop Closure)
- Map representation

SIFT matching

- Visual features (from image) in 3D (from depth)
- Figure out how the camera moved by matching these feature



*Slide from Peter Henry

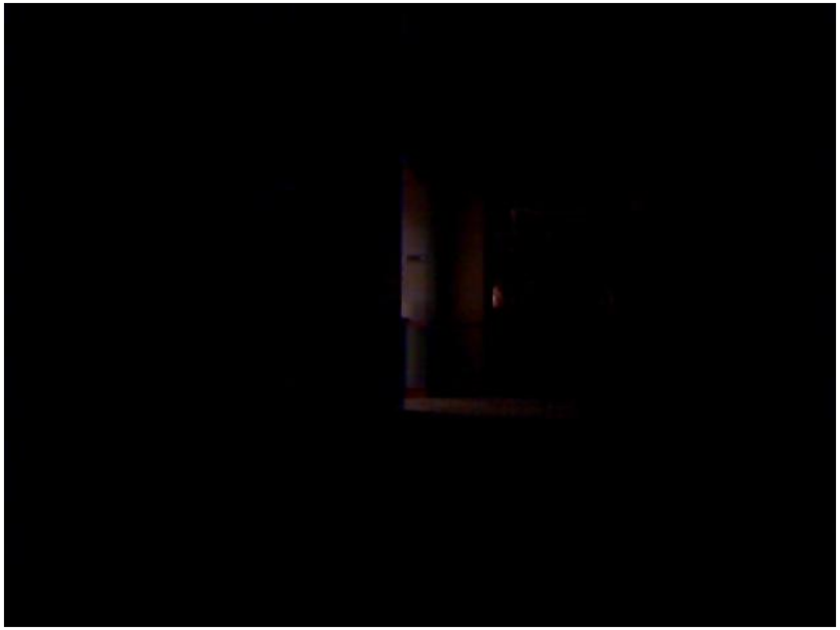
RANSAC

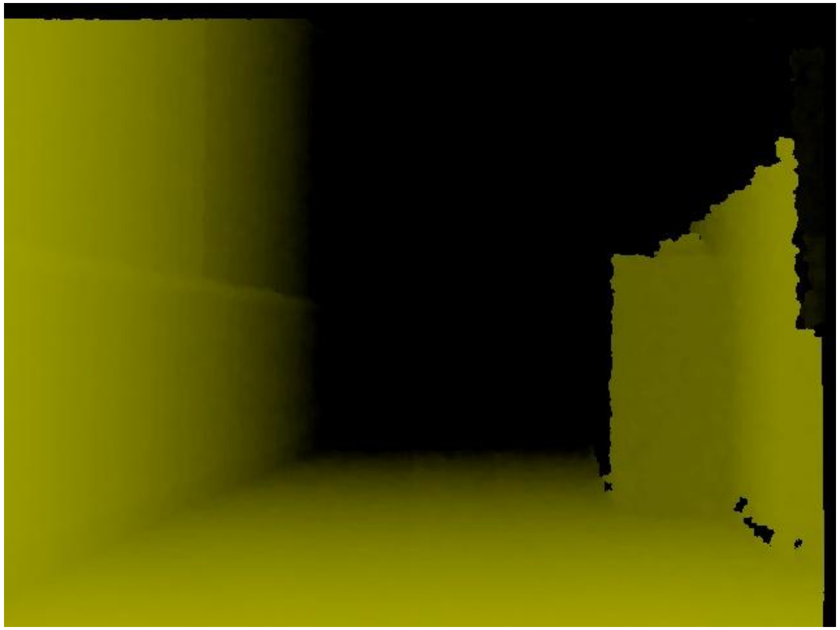
- For each feature point, find the most similar descriptor in the other frame
- Find largest set of consistent matches
- Move the new frame to align these matches

*Slide from Peter Henry

Using ICP

- Low light / Lack of visual "texture" or features
- Kinect still provides depth or "shape" information





Joint Optimization (RGBD-ICP)

RGBD-ICP ($\mathbf{P}_s, \mathbf{P}_t$):

1: $F = \mathit{Extract\_RGB\_point\_features}(\mathbf{P}_s)$

2: $F_{\text{target}} = \mathit{Extract\_RGB\_point\_features}(\mathbf{P}_t)$

3: $(\mathbf{t}^*, A_f) = \mathit{Perform\_RANSAC\_Alignment}(F, F_{\text{target}})$

4: ***repeat***

5: $\quad A_d = \mathit{Compute\_Closest\_Points}(\mathbf{t}^*, \mathbf{P}_s, \mathbf{P}_t)$

6: $\quad \mathbf{t}^* = \mathit{Optimize\_Alignment}(\mathbf{t}^*, A_f, A_d)$

7: ***until*** $(\mathit{Change}(\mathbf{t}^*) \leq \theta)$ *or* $(\mathit{maxIter\ reached})$

8: ***return*** $\mathbf{t}^*$

Resulting Map

[Henry-Krainin-Herbst-Ren-Fox]

3D mapping

- Our implementation on SIFT only
- Kinect fusion

Topics

- RGB-D Mapping
- Robotics grasping
- Object recognition
- Human tracking

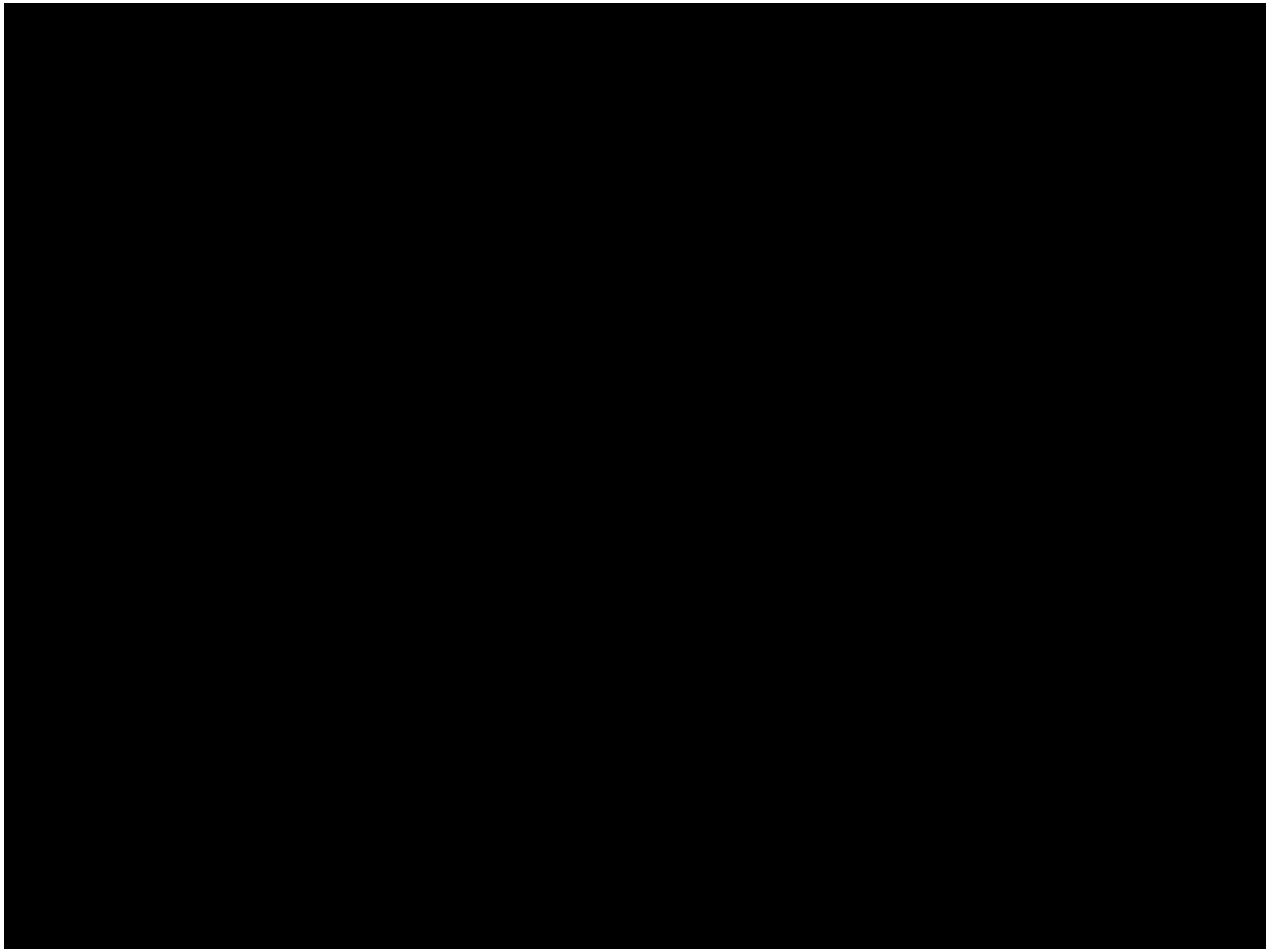
Robot manipulation: Big Picture

- Personal robots should learn incrementally from experience

- Robots can later perform useful actions with models:
  - Recognition
  - Pose estimation
  - Reliable grasping

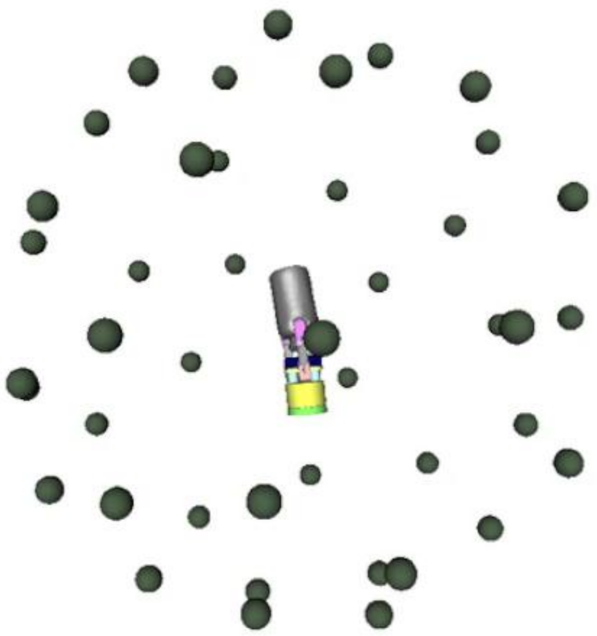
Autonomous Generation of Complete 3D Object Models Using Next Best View Manipulation Planning. Krainin, Curless, Fox, ICRA 2011

IJRR 11

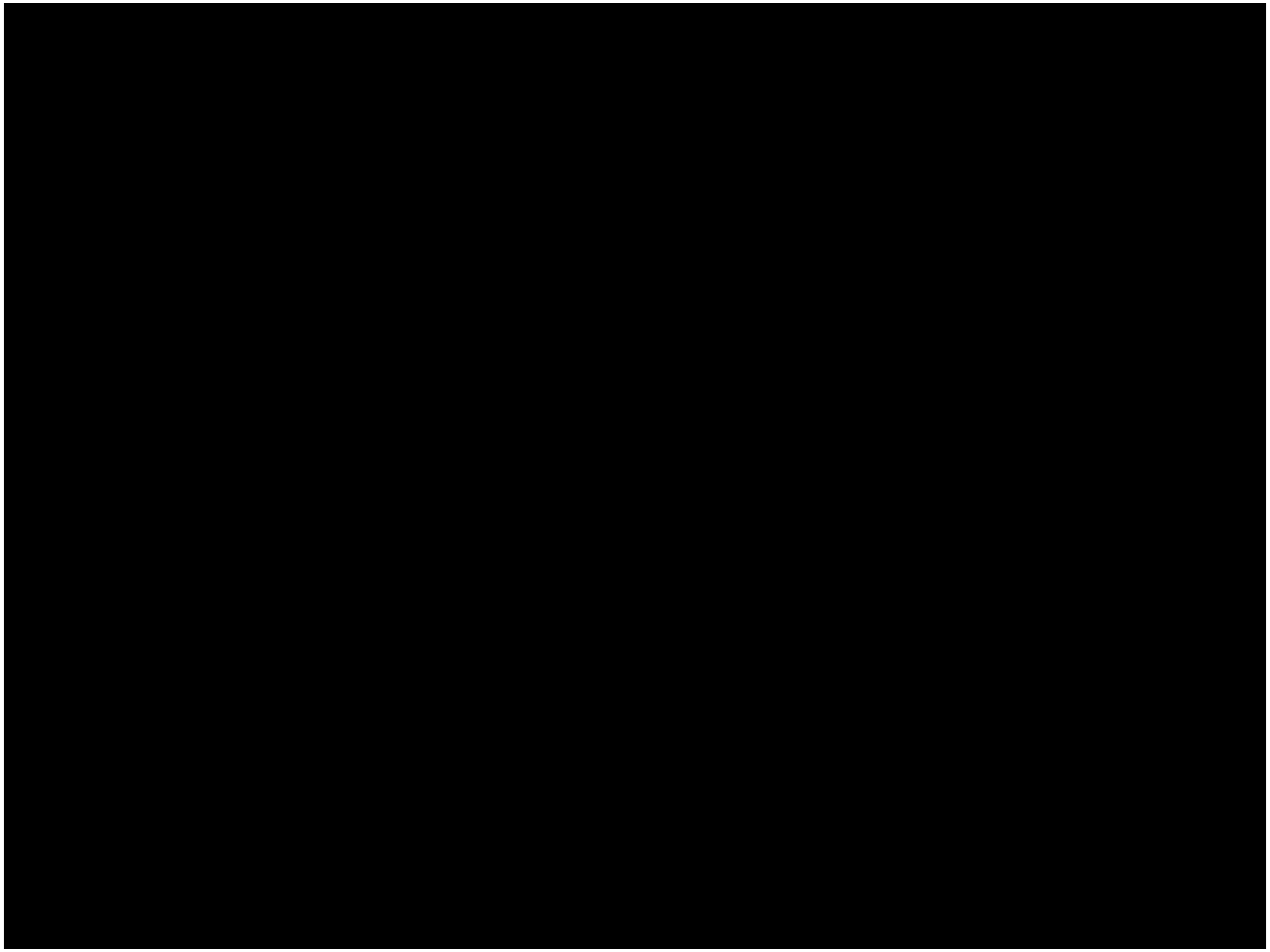


View Selection Algorithm

- Conceptually similar to Planetarium Algorithm [Connolly ’85]
- Procedure:
  – Extract object isosurface with confidences
  – Generate kinematically achievable viewpoints
  – Compute information gain (quality) for each viewpoint
  – Select view as tradeoff between quality and cost

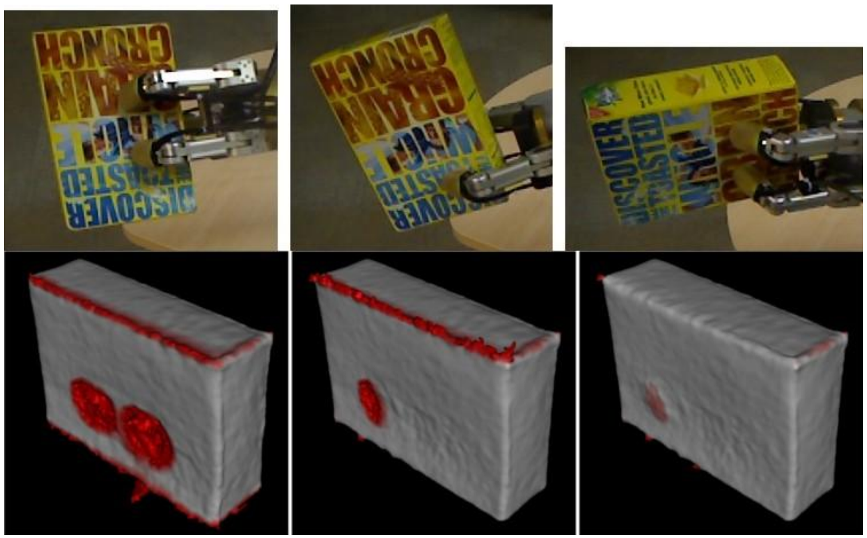


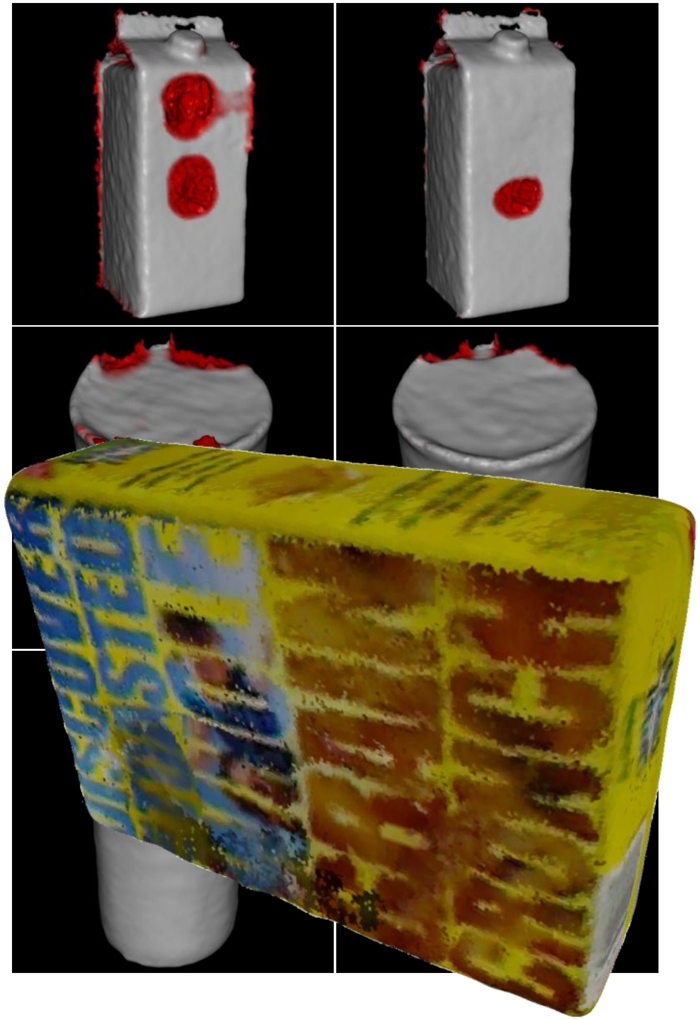
Manipulation Planning



Multiple Grasp Results

- Evaluated regrasping on four objects
- Includes box with three grasps





Topics

- RGB-D Mapping
- Robotics grasping
- Object recognition
- Human tracking

Object recognition

- List of papers:

"Object Recognition with Hierarchical Kernel Descriptors" ,Liefeng Bo,et al . CVPR 11

"A Large-Scale Hierarchical Multi-View RGB-D Object Dataset“, Kevin Lai et al. ICRA 11

"Sparse Distance Learning for Object Recognition Combining RGB and Depth Information" , Kevin Lai et al. ICRA 11

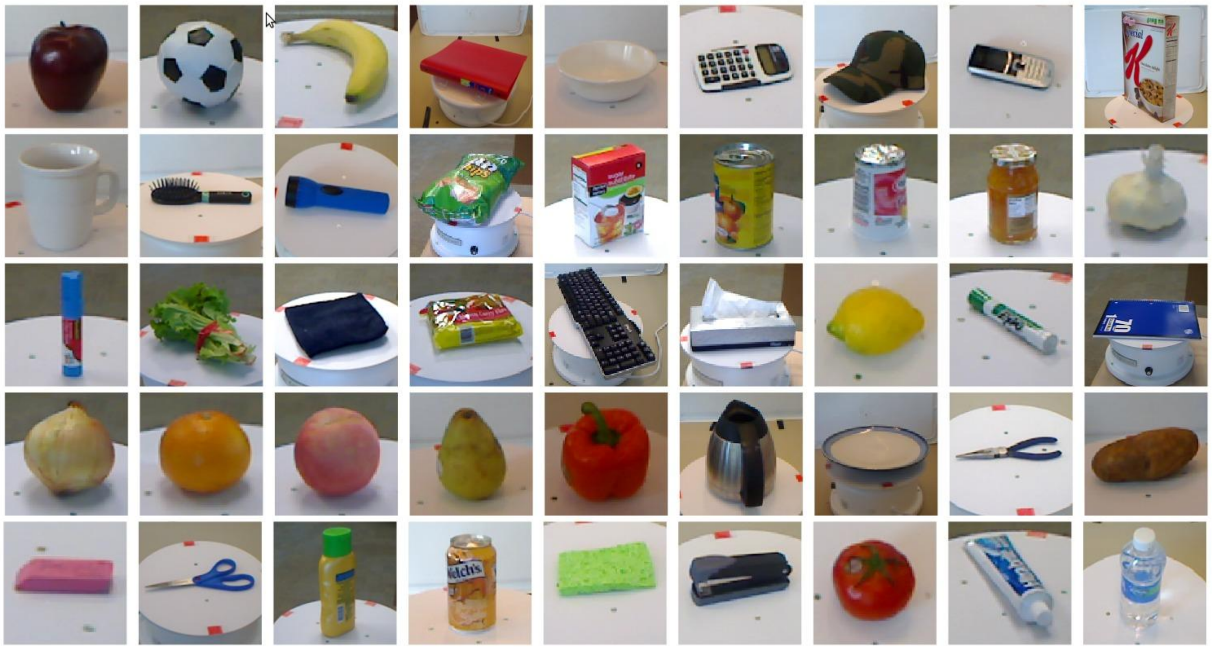
“Depth Kernel Descriptors for Object Recognition”, Liefeng Bo et al. IROS 11

“RGB-D Object Discovery via Multi-Scene Analysis.” Evan Herbst et al. IROS 11

“A Scalable Tree-based Approach for Joint Object and Pose Recognition”, Kevin Lai et al. AAAI 11

“Kernel Descriptors for Visual Recognition”, Liefeng Bo et al. NIPS 10

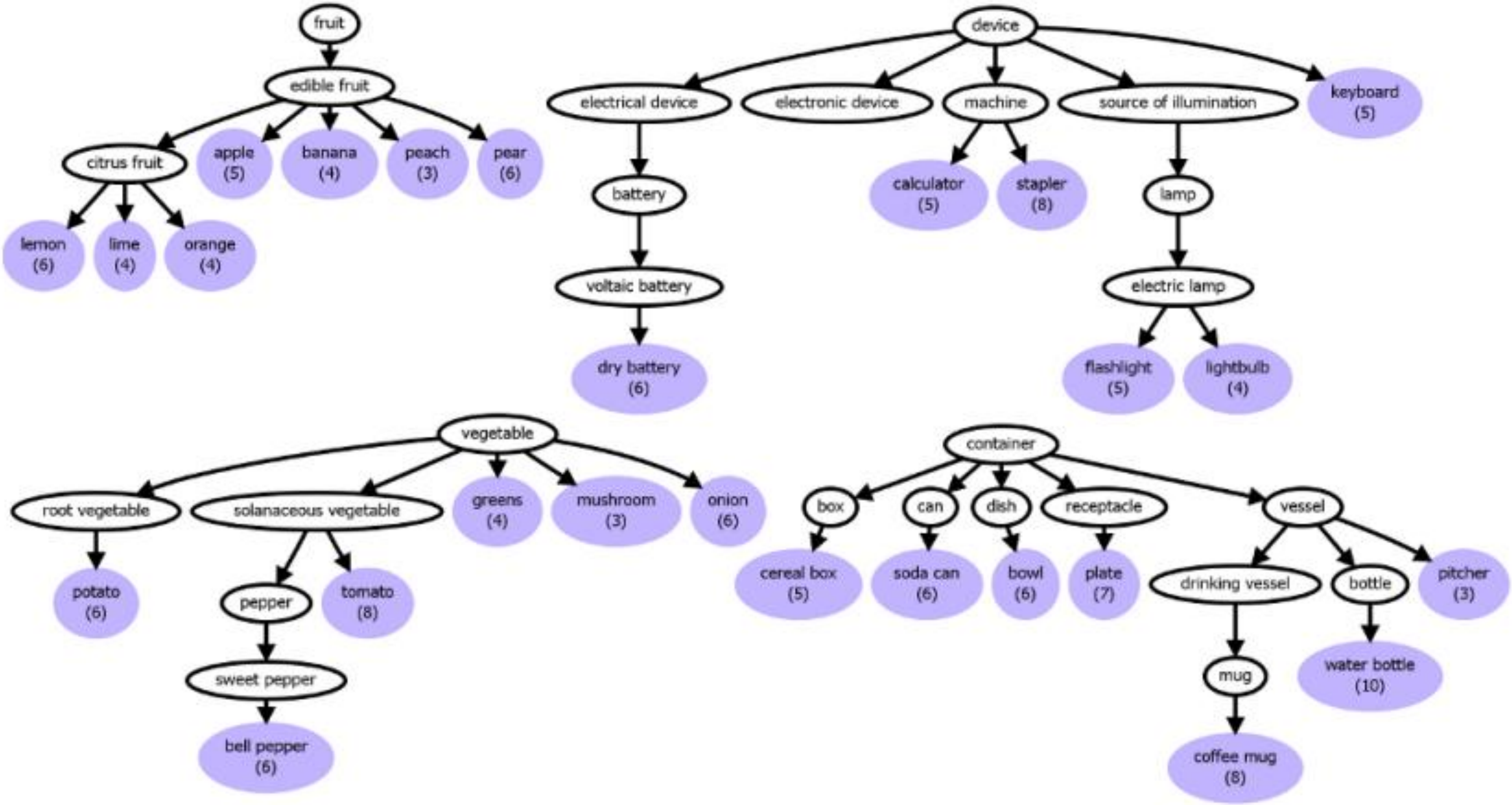
RGB-D Object Dataset



300 objects from 51 categories,
250,000 RGB-D views

Cluttered scenes

RGB-D Object Dataset



Benchmarking RGB-D Recognition

Category-Level Recognition (51 categories)

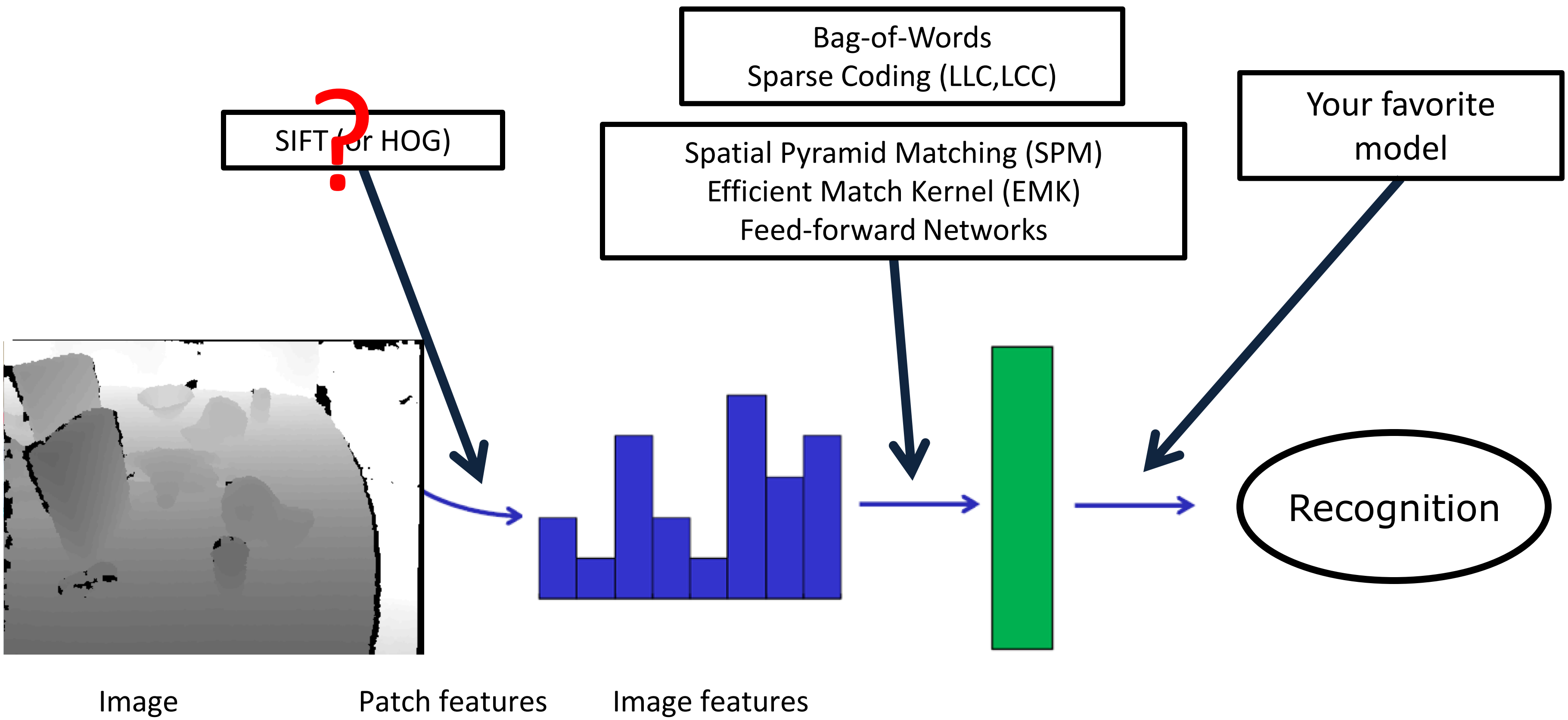
Classifier	Shape (Depth)	Vision (RGB)	RGB-D
Linear SVM	51.7±1.8	72.7±3.2	80.5±2.9
Kernel SVM	63.5±2.3	72.9±3.2	83.0±3.7
RandomForest	65.5±2.4	73.1±3.7	78.5±4.1

Instance-Level Recognition (303 instances)

Classifier	Shape (Depth)	Vision (RGB)	RGB-D
Linear SVM	29.4±0.5	90.4±0.5	89.6±0.5
Kernel SVM	50.1±0.9	90.8±0.5	90.4±0.6
RandomForest	51.6±1.1	89.6±0.7	90.2±0.3

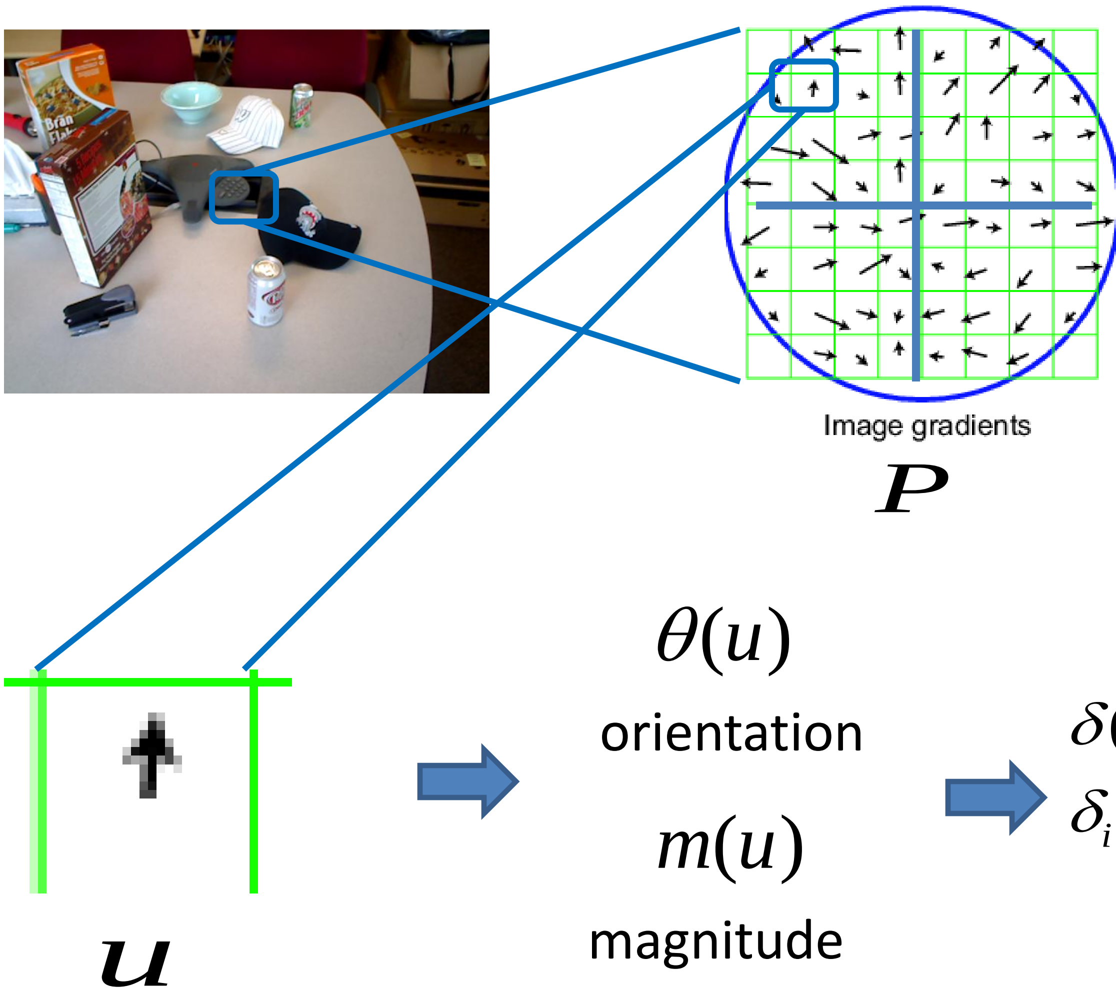
Slides from Ren. 1% Lowered than number reported in the paper

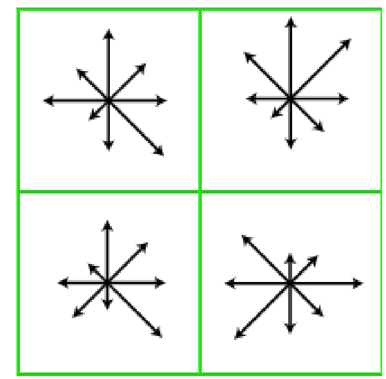
RGB-D Object Recognition



*Slide from Ren Xiaofeng

A Kernel view of SIFT/HoG





$\theta(u)$

orientation

$m(u)$

magnitude

1. Find the corresponding histogram bin (suppose each bin contains 45 degree)

$$\delta(u) = [\delta_1(u), \delta_2(u), \ldots, \delta_8(u)]$$

$$\delta_i(u) = \mathrm{mod}(\theta(u), 45^o) == i$$

2. Compute normalize term

$$m(P) = \sqrt{\sum_{u \in P} m(u)^2}$$

A Kernel view of SIFT/HoG

$$F(P) = \frac{1}{m(P)} \sum_{u \in P} m(u)\delta(u)$$

$$F(P) = \sum_{u \in P} \tilde{m}_u \boxed{\times sv(}\theta_u)$$

Some vector

- Suppose we have trained linear SVM classifier
- Have support vectors $(F_i, y_i), i = 1, \ldots, N$
- With one testing F_t
- Decision $y = wF_t - b = \sum_i \alpha_i y_i \boxed{F_i F_t} - b$

[Bo-Ren-Fox; CVPR 10 ;NIPS 2010; IROS 11;]

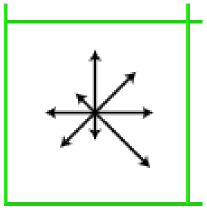
A Kernel view of SIFT/HoG

- When two SIFT/HoG meet

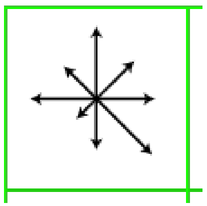
$$F(P) = \sum_{u \in P} \tilde{m}_u sv(\theta(u)) \qquad F(Q) = \sum_{v \in Q} \tilde{m}_v sv(\theta(v))$$

$$F(P) \bullet F(Q) = \sum_{u \in P} \sum_{v \in Q} \tilde{m}_u \tilde{m}_v sv(\theta_u) \bullet sv(\theta_v) = \sum_{u \in P} \sum_{v \in Q} \tilde{m}_u \tilde{m}_v K_o(\theta_u, \theta_v)$$

P

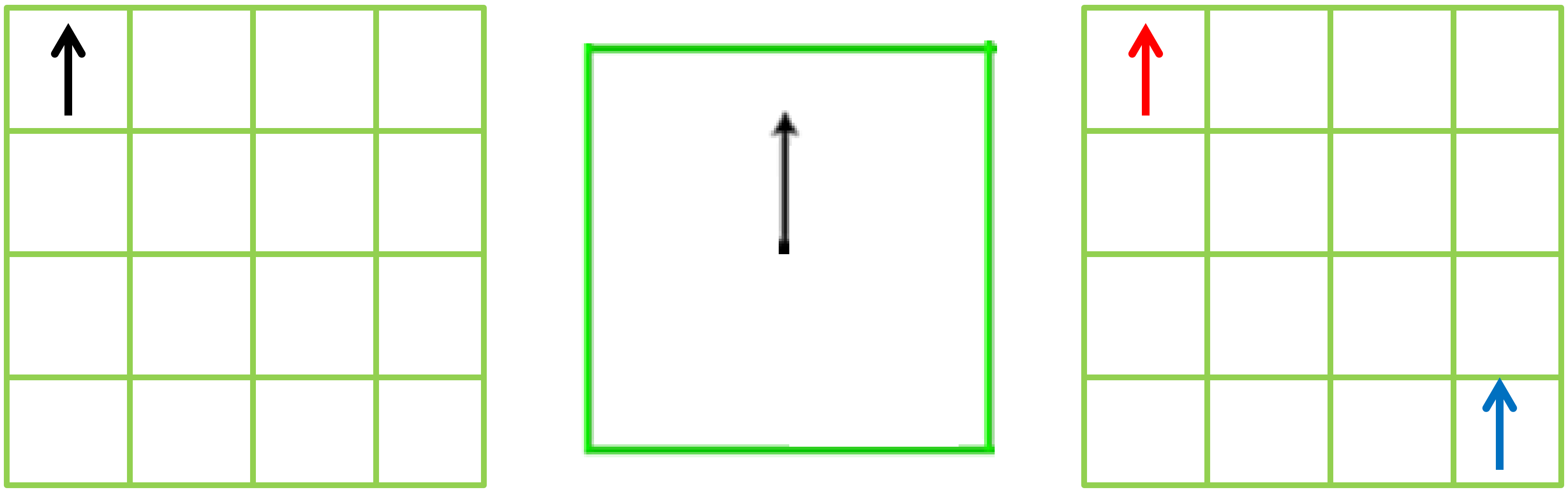


Q



How well these orientations match each other

Soft Binning/Considering the position



Some vector orientation **sv**

Give the same **sv** and thus the same matching

- Add another kernel considering the position of the pixel

$$K_p(u,v) = e^{-\gamma(u-v)^2}$$

Kernel Descriptors

normalized gradient magnitude

gradient orientation

pixel coordinates

Gradient Match Kernel

$$K_{\text{grad}}(P,Q)=\sum_{u\in P}\sum_{v\in Q}\tilde{m}_u\tilde{m}_v k_o(\tilde{\theta}_u,\tilde{\theta}_v)k_p(u,v)$$

image patch

kernels

- Propose several other kernel feaetures (Color, Shape etc) in NIPS
- Propose hierarchy kernel (kernel of kernel) in CVPR

(treat Depth as gray image)

- Propose depth kernel in IROS (PCA, shape, edge)

[Bo-Ren-Fox; CVPR 10 ;NIPS 2010; IROS 11;]

Experiment: on RGB-D dataset (average accuracy)

Category

kSVM	83.8 ± 3.5

ICRA 10, Fig 8

IDL	$\mathbf{85.4 \pm 3.2}$

ICRA 10, Fig 4

Combination of all HKDES	**84.1±2.2**

CVPR 11, table 4

This work	**86.2±2.1**

IROS 11, table 9

Instance

kSVM	74.8

ICRA 10, Fig 8

Combination of all HKDES	**82.4**

CVPR 11, table 4

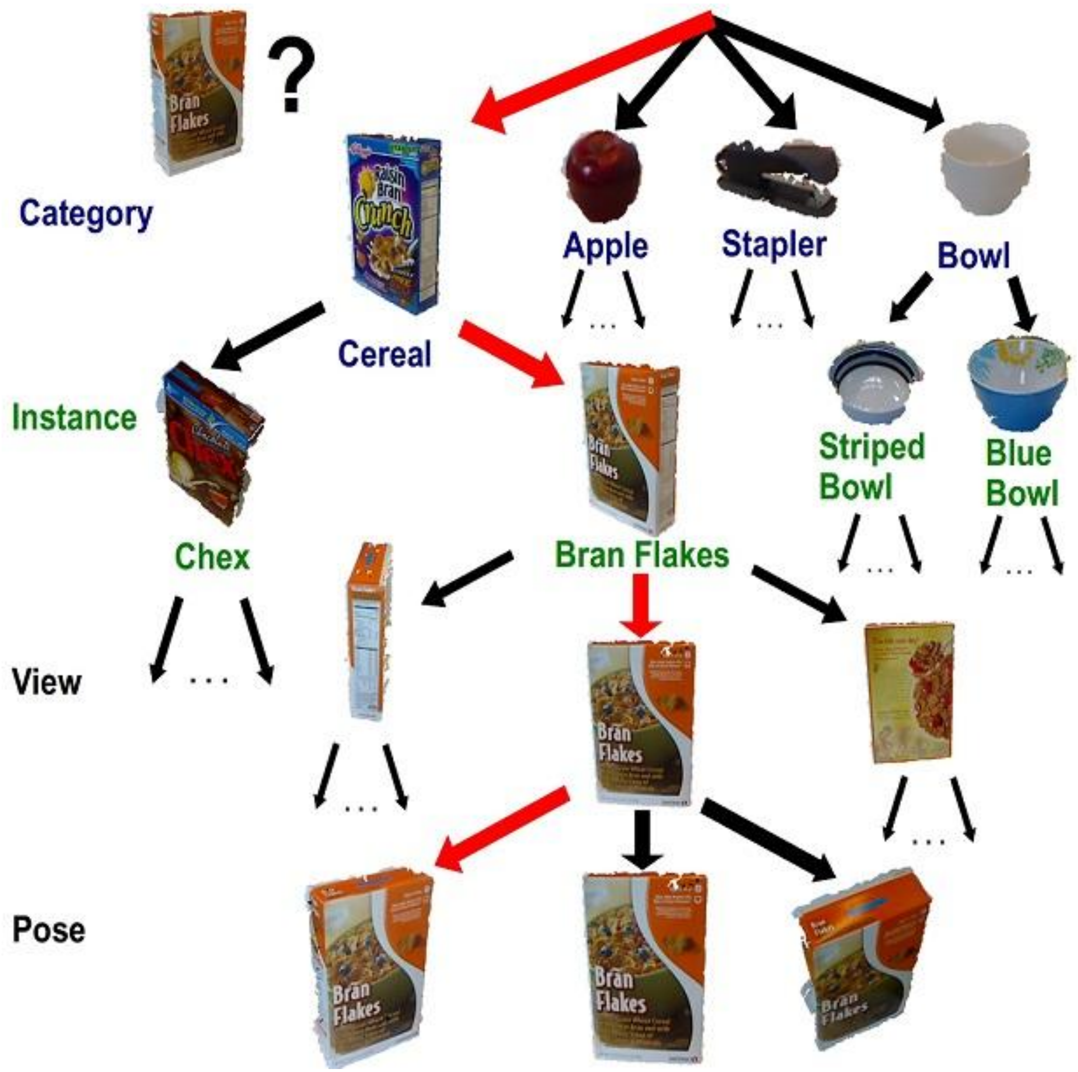
This work	**84.5**

IROS 11, table 8

Scalable and Hierarchical Recognition

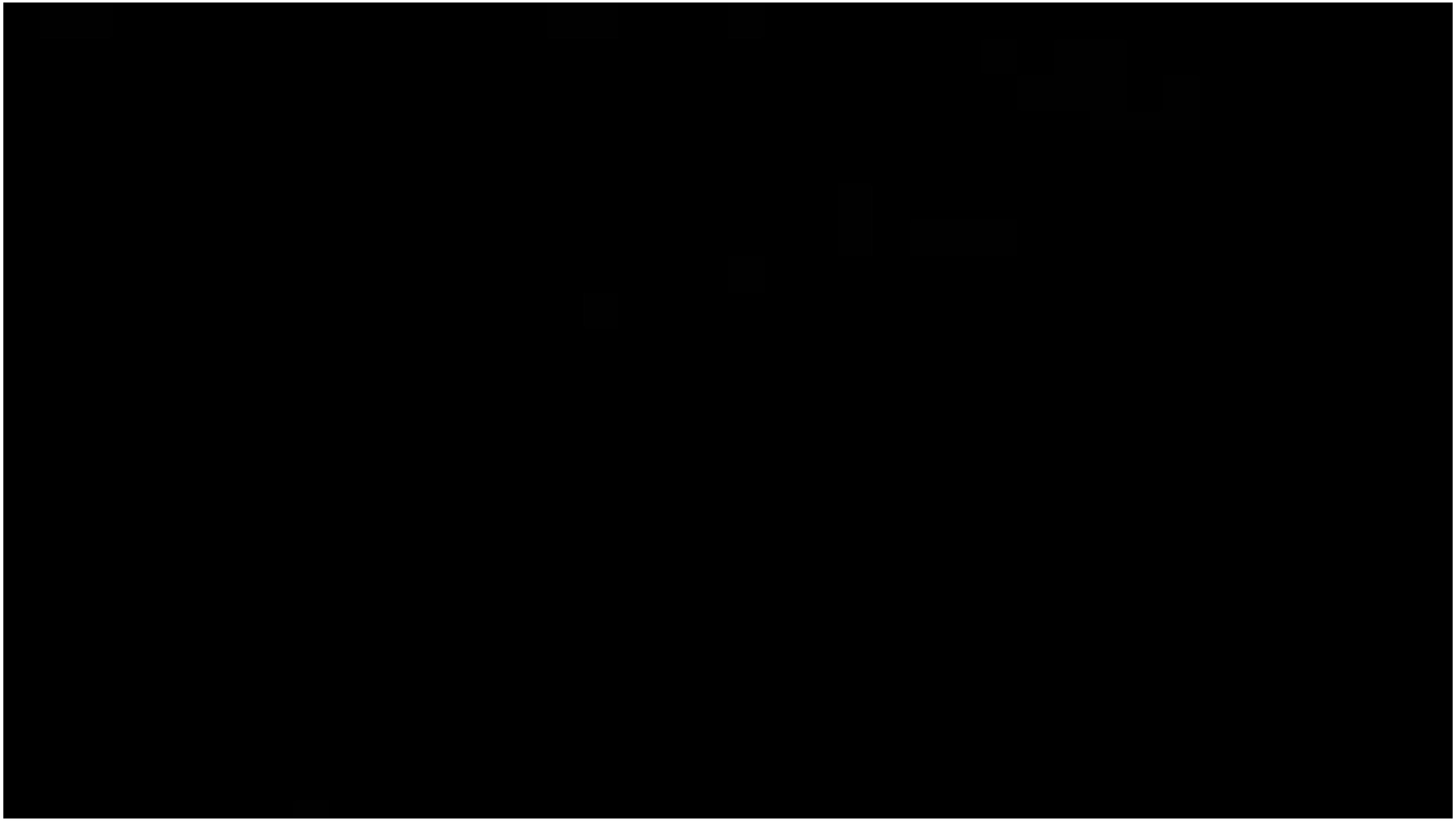
8 discrete views

continuous angles



[Lai-Bo-Ren-Fox; AAAI 2011]

Application: Interactive LEGO



RGB-D used for object recognition and hand tracking

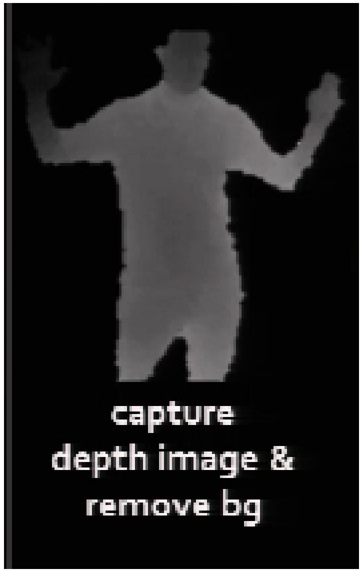
[Ziola-Harrison-Powledge-Lai-Bo-Ren-Fox]

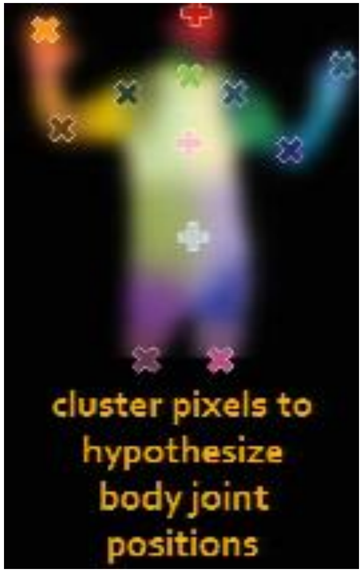
Application: Chess Playing Robot

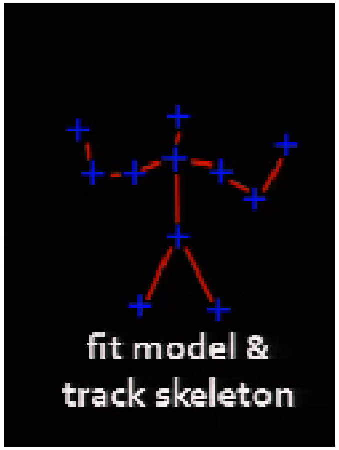
Topics

- RGB-D Mapping
- Robotics grasping
- Object recognition
- Human tracking

Kinect: Real time human tracking



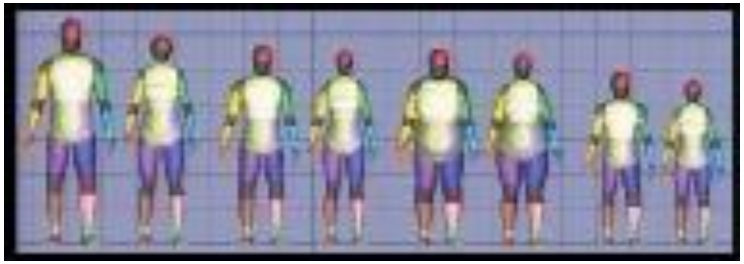




* Real-Time Human Pose Recognition in Parts from Single Depth Images. Shotton, et al, CVPR 2011.

Synthesizing Training data

- Motion capture to 100 k poses
- Retargeting to different models



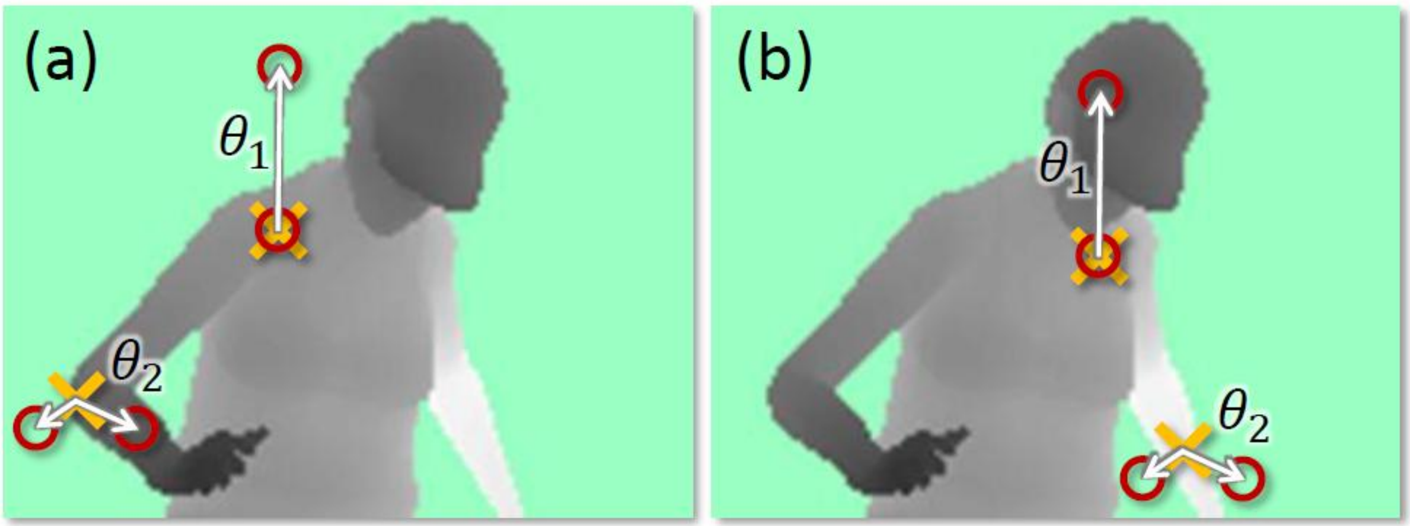
- Render depth and body parts

- Use the real and synthetic training data (1m)

Training data

Features

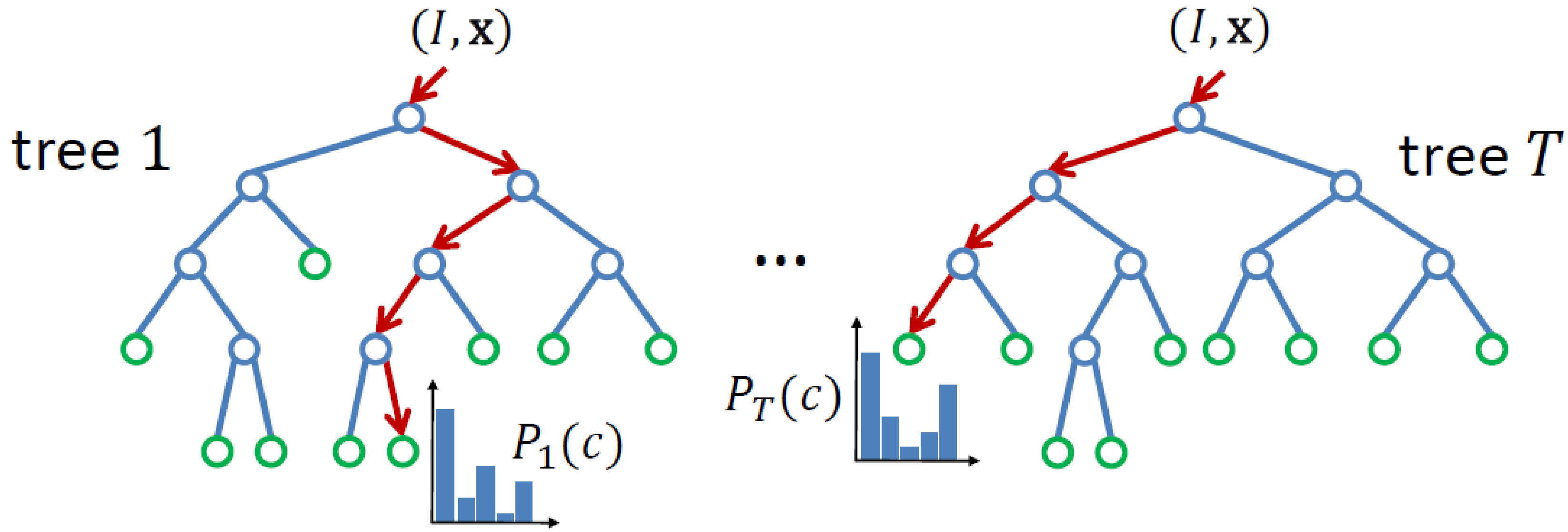
- $d_I(x)$: depth at pixel x



$$f_\theta(I, \mathbf{x}) = d_I\left(\boxed{\mathbf{x} + \frac{\mathbf{u}}{d_I(\mathbf{x})}}\right) - d_I\left(\boxed{\mathbf{x} + \frac{\mathbf{v}}{d_I(\mathbf{x})}}\right) , \quad (1)$$

Two offset from x

Decision Tree



- P(c|I,x): distribution of pixel x over labels c

$$P(c|I,\mathbf{x}) = \frac{1}{T}\sum_{t=1}^{T} P_t(c|I,\mathbf{x}) \; .$$

Training decision tree

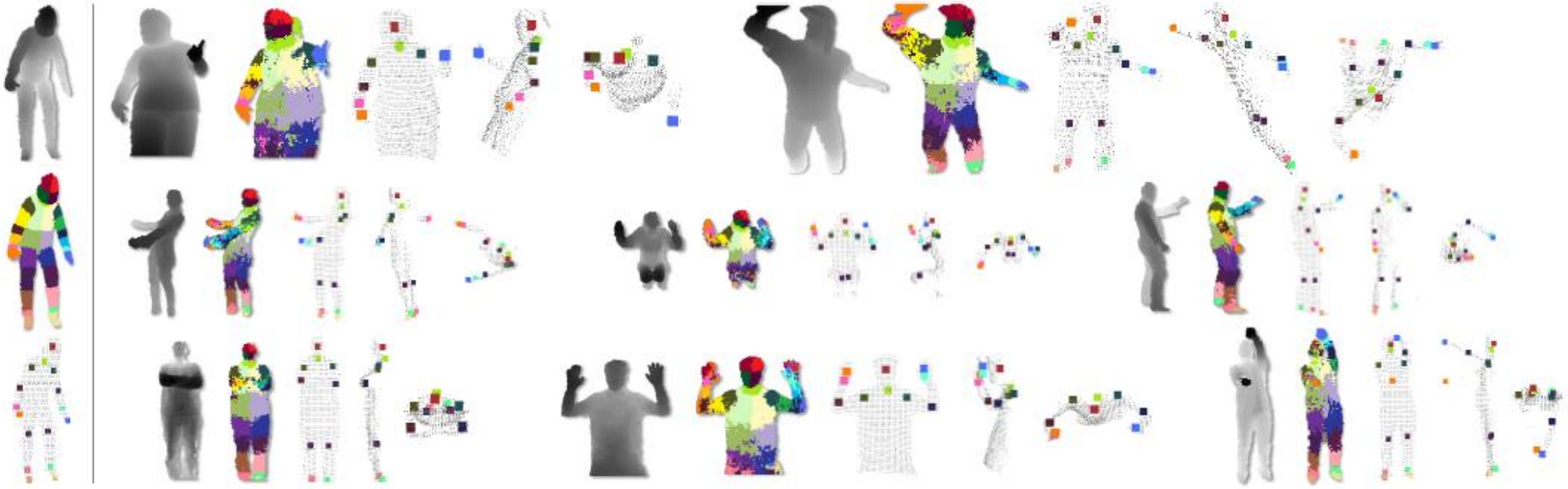
- Randomly select a set of θ and τ (a set of splits)
- Split training examples by each split
- Choose the split with maximum information gain
- Move into next layer
- 3 trees to depth 20 from 1 million images =1 day training on 1000 cores

Speed

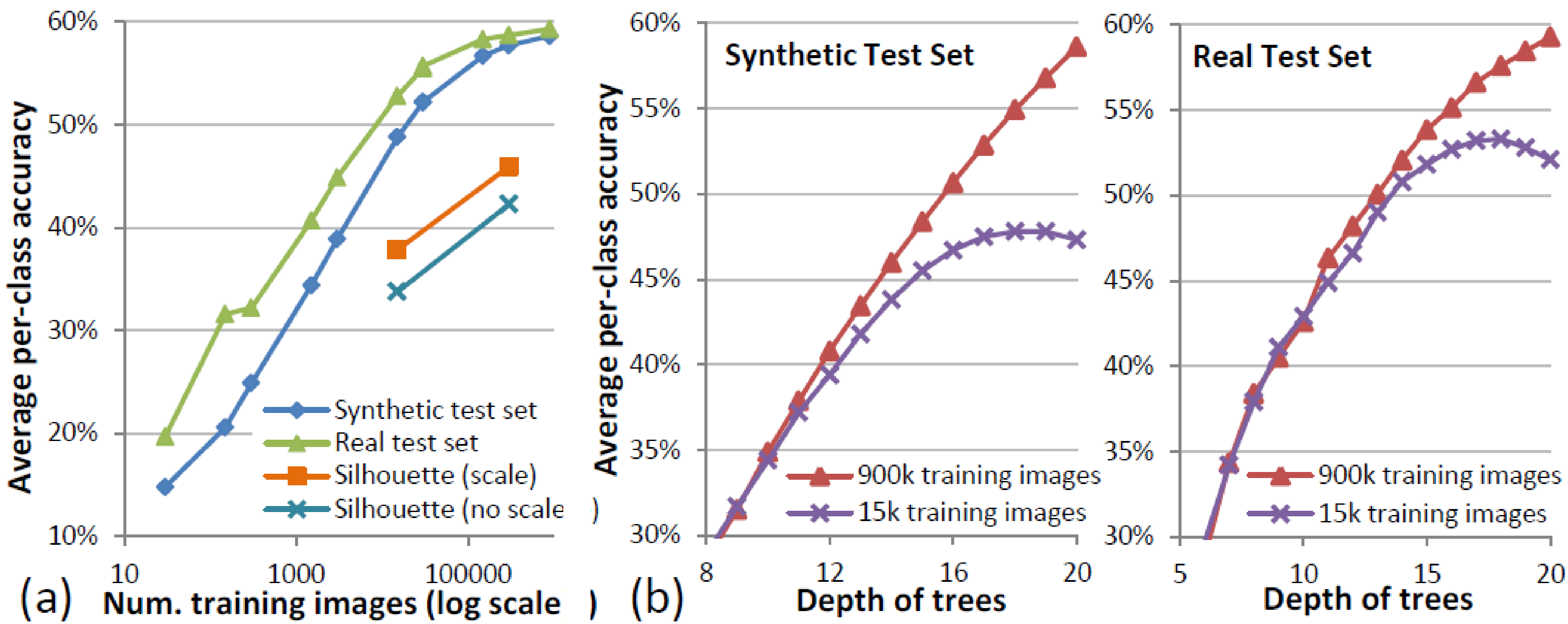
- Each feature computation:
  - read 3 image pixels
  - 5 arithmetic operations
  - Straight forward to implemented on GPU
- Decision trees:
  - Fast computing
  - Can be parallel between trees

Experiment

- Use mean-shift to find the joint.



Experiment



- demo

Conclusion

- Kinect is helpful
  - 3D modeling
  - Robotics
- Kinect introduces new data and features
  - Object recognition/scene understanding
- Many interesting applications on-going

Future

- Will RGB-D have a deep impact on vision applications?

  Yes. It's already happening, faster than we can track.

- Will RGB-D start a revolution in vision applications?

  No. We still need to solve recognition, segmentation, tracking, scene understanding, etc. etc.

  Yes. RGB-D helps address two issues in computer vision: loss of 3D from projection; lighting conditions.

  RGB-D helps "abstract away" many low-level problems.

Zhaoyin Jia

THANKS