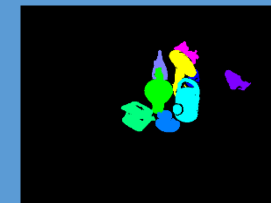
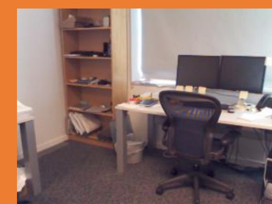
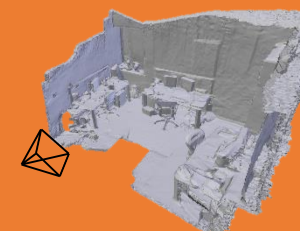
Case Study: Pose Estimation and Pose Tracking

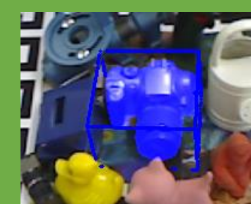
Image Segmentation:
- Decision Forest
- Features, Training Recap

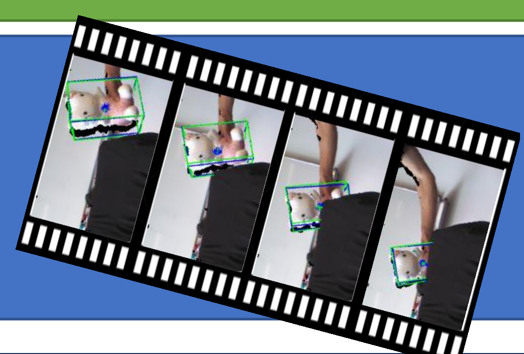
Camera Re-Localization:
- Regression Forest
- Split-Criteria, Kabsch, RANSAC

One-Shot Pose Estimation:
- Decision-Regression Forest
- Hypothesis Sampling, Hypotheses Energy

Object Pose Tracking
- Particle Filter
- Importance Sampling



Application Scenarios

- Robotics
  - Recognition/Tracking
  - Automatic Grasping

- Augmented Reality
  - Alteration
  - Annotation
  - Substitution

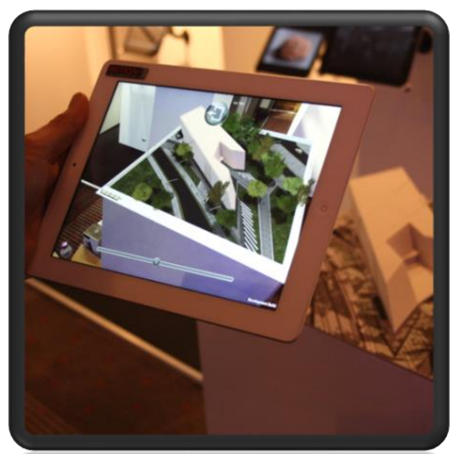
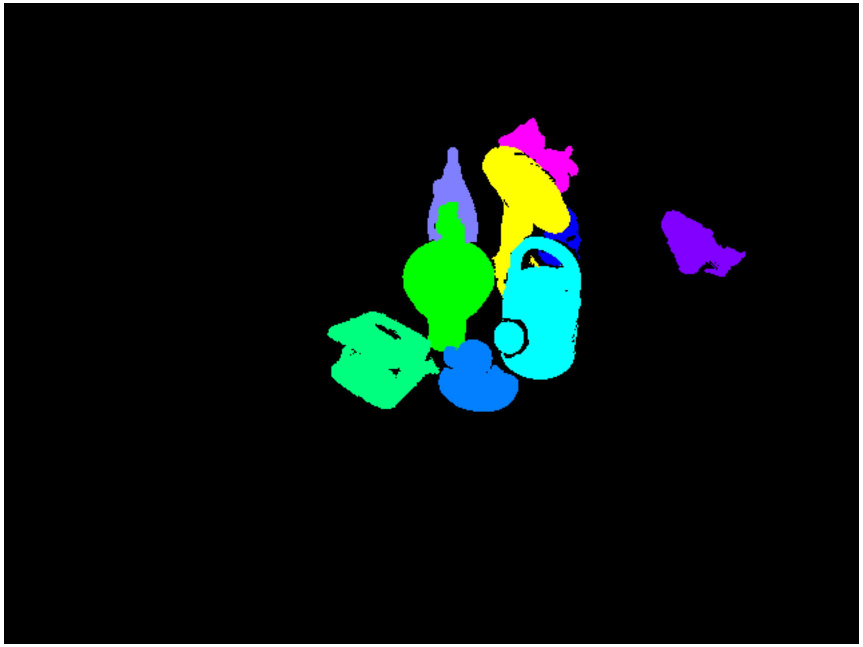


Image Segmentation

Input: RGB-D image

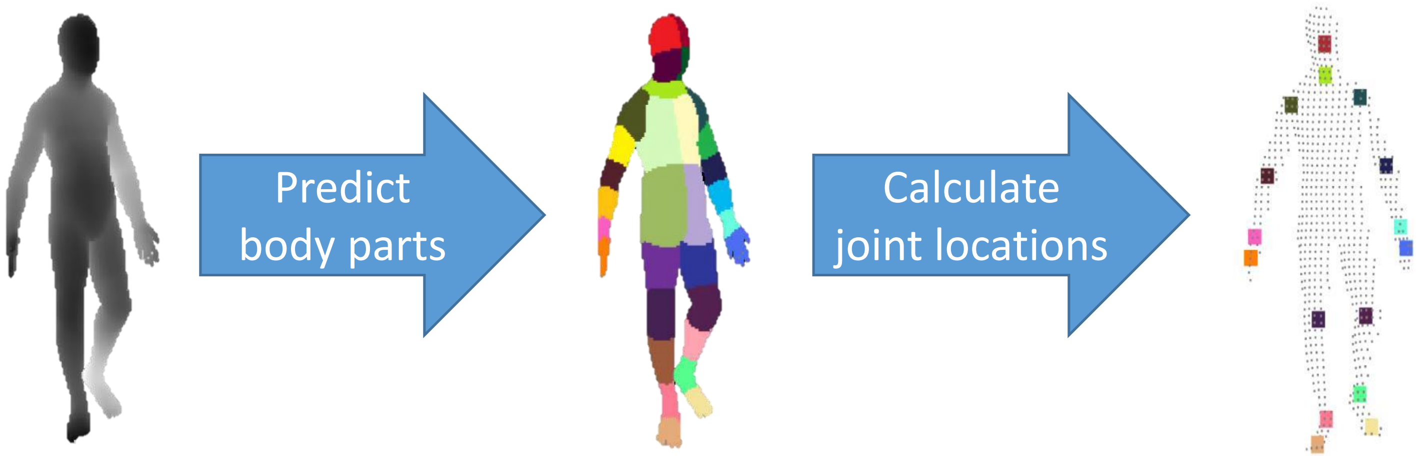
Output: Segmentation



Dataset of Hinterstoisser et al., Model Based Training, Detection and Pose Estimation of Texture-Less 3D Objects in Heavily Cluttered Scenes, ACCV 12

Decision Forest: Recap

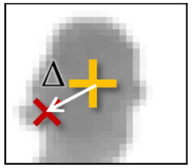
Reminder: Body pose estimation

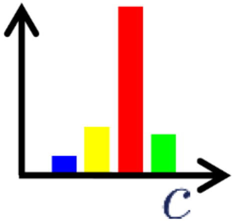


A decision forest assigns for each pixel its body part label.

What has to be defined?

Features:	Pixel differences
Split criterium:	Information gain
Stopping rules:	Max. tree depth
What to store at the leafs:	Body part histograms

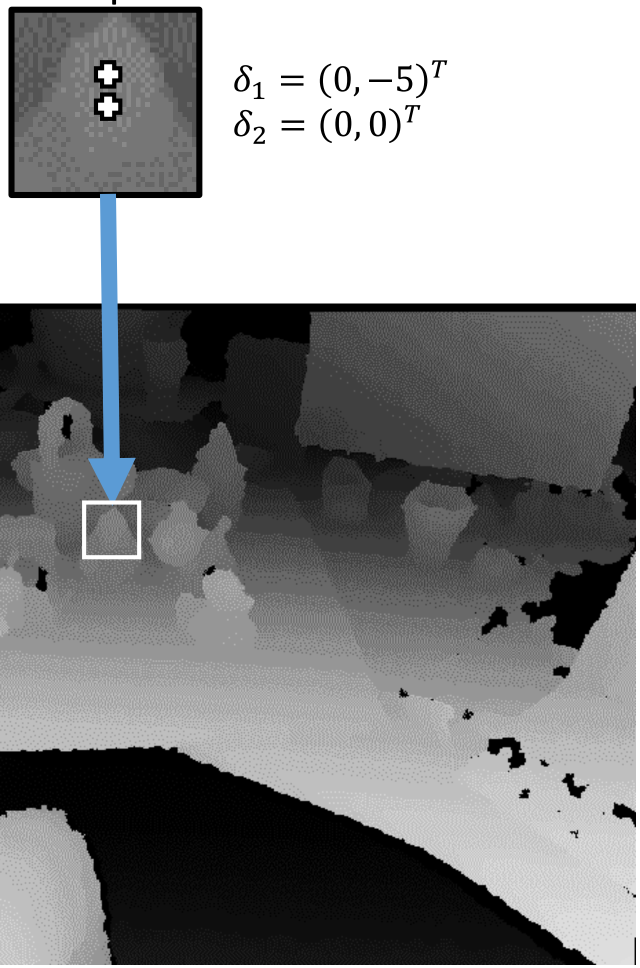




Features: Depth Feature

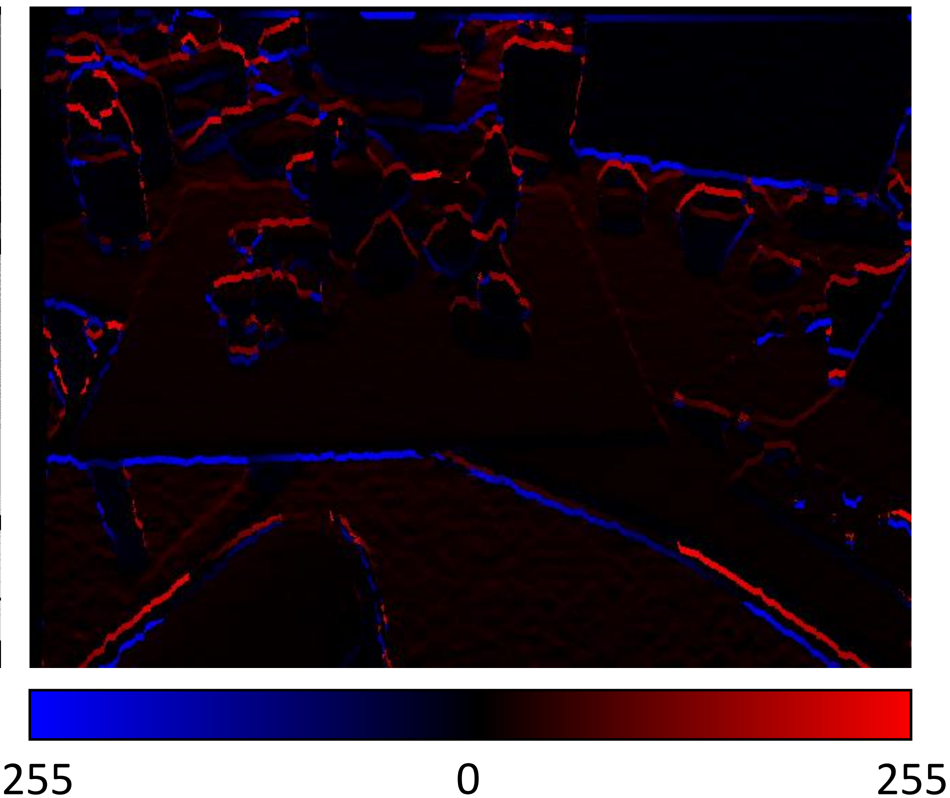
40px

$\delta_1 = (0, -5)^T$
$\delta_2 = (0, 0)^T$

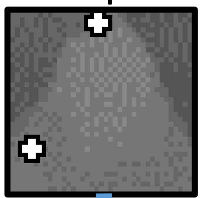


$$f^D(\boldsymbol{p}) = I^D(\boldsymbol{p} + \boldsymbol{\delta}_1) - I^D(\boldsymbol{p} + \boldsymbol{\delta}_2)$$

f^D	… depth feature
I^D	… depth image
$\boldsymbol{p}$	… current pixel
$\boldsymbol{\delta}_{1\backslash 2}$	… offset



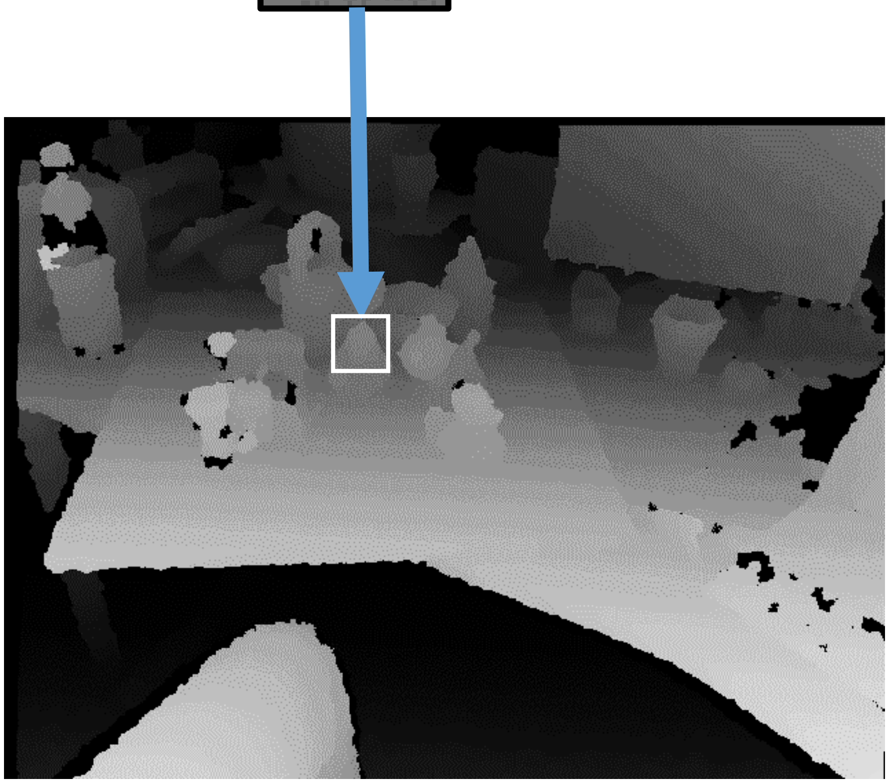
Features: Depth Feature

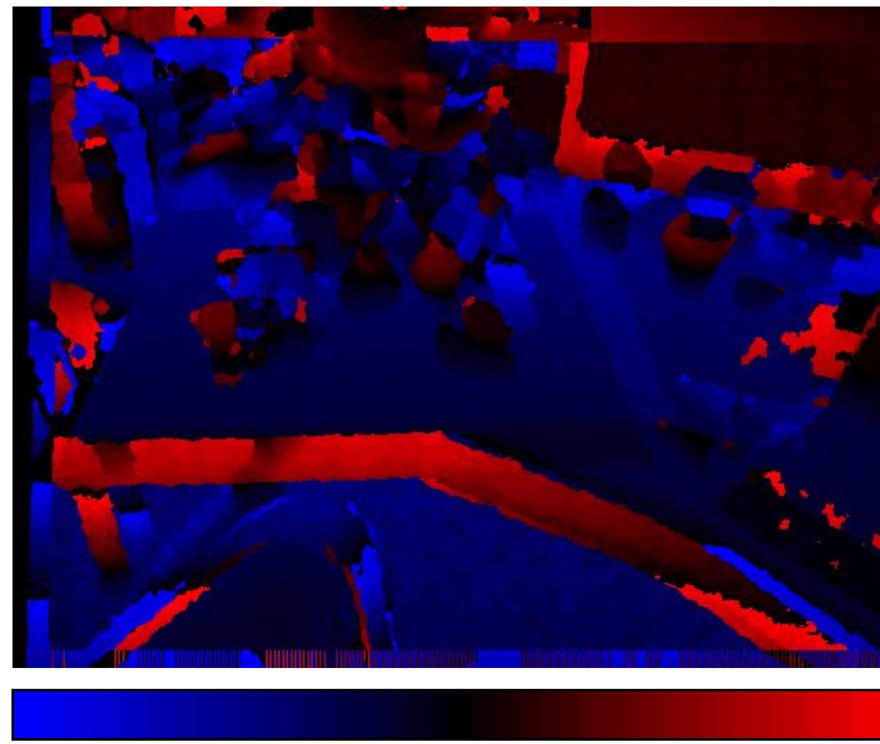


$\delta_1 = (-17, 13)^T$
$\delta_2 = (-1, 20)^T$

$$f^D(\boldsymbol{p}) = I^D(\boldsymbol{p} + \boldsymbol{\delta}_1) - I^D(\boldsymbol{p} + \boldsymbol{\delta}_2)$$

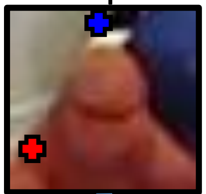
f^D	… depth feature
I^D	… depth image
$\boldsymbol{p}$	… current pixel
$\boldsymbol{\delta}_{1\backslash 2}$	… offset





Features: RGB Feature

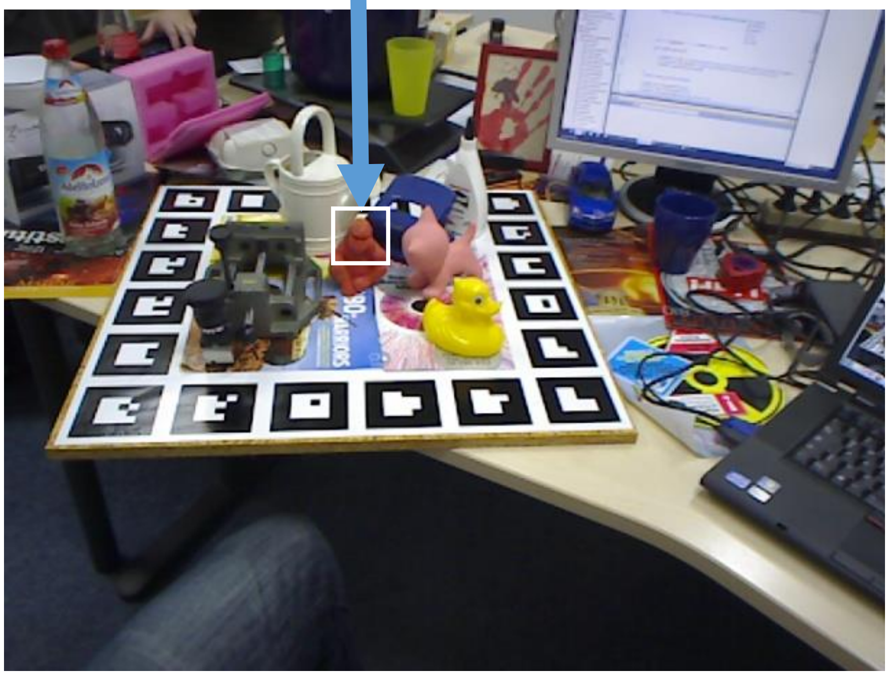
40px

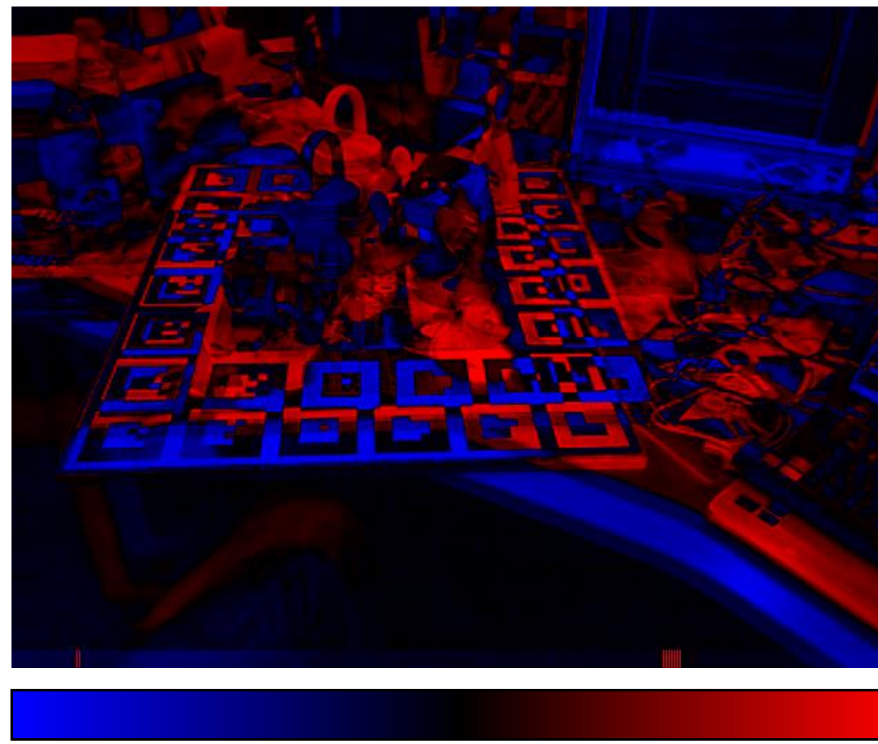


$\delta_1 = (-17, 13)^T$
$c_1 = 0$
$\delta_2 = (-1, -20)^T$
$c_2 = 2$

$$f^{RGB}(\boldsymbol{p}) = I^{RGB}(\boldsymbol{p} + \boldsymbol{\delta}_1, c_1) - I^{RGB}(\boldsymbol{p} + \boldsymbol{\delta}_2, c_2)$$

f^{RGB}	… RGB feature
I^{RGB}	… RGB image
$\boldsymbol{p}$	… current pixel
$\boldsymbol{\delta}_{1\backslash 2}$	… offset
$c_{1\backslash 2}$	… RGB channel





-255 0 255

Features: Depth Adaptation

How to deal with scale?

$$f^{RGB}(\boldsymbol{p}) = I^{RGB}(\boldsymbol{p} + \boldsymbol{\delta}_1, c_1) - I^{RGB}(\boldsymbol{p} + \boldsymbol{\delta}_2, c_2)$$

$I^D(\boldsymbol{p}) = 1.0m$

Patch: 40px

$I^D(\boldsymbol{p}) = 0.7m$

Patch: 40px

Features: Depth Adaptation

Scale Invariance

$$f^{DA-RGB}(\boldsymbol{p}) = I^{RGB}\left(\boldsymbol{p} + \frac{\boldsymbol{\delta}_1}{I^D(\boldsymbol{p})}, c_1\right) - I^{RGB}\left(\boldsymbol{p} + \frac{\boldsymbol{\delta}_2}{I^D(\boldsymbol{p})}, c_2\right)$$

$$f^{DA-D}(\boldsymbol{p}) = I^{D}\left(\boldsymbol{p} + \frac{\boldsymbol{\delta}_1}{I^D(\boldsymbol{p})}\right) - I^{D}\left(\boldsymbol{p} + \frac{\boldsymbol{\delta}_2}{I^D(\boldsymbol{p})}\right)$$

$I^D(\boldsymbol{p}) = 1.0m$

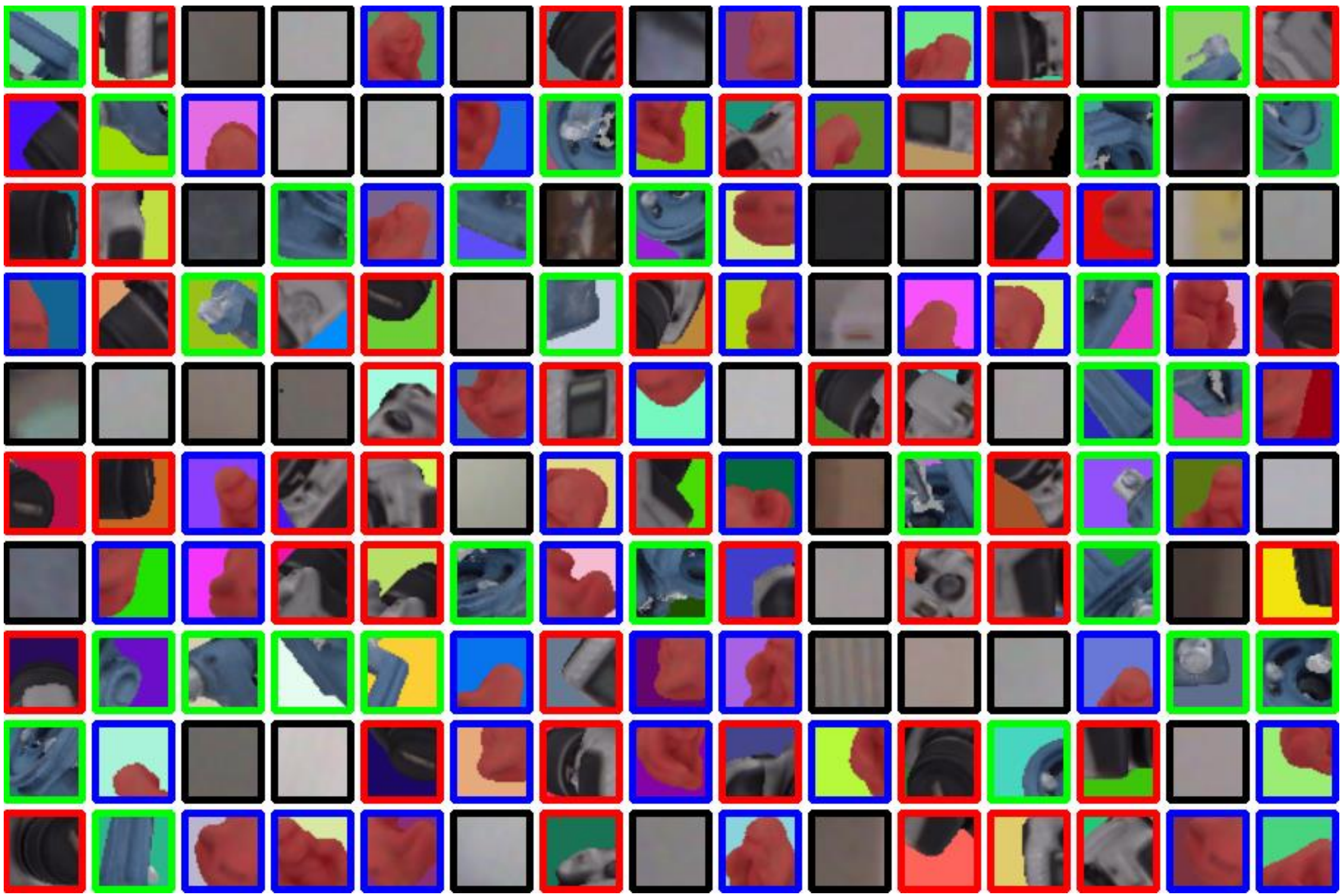
Patch: 40px

$I^D(\boldsymbol{p}) = 0.7m$

Patch: 57px

Sample Set of Training Patches

Object 1 Object 2 Object 3 … Background

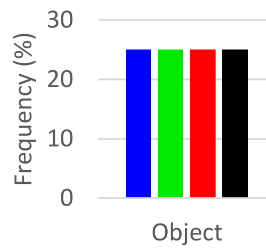


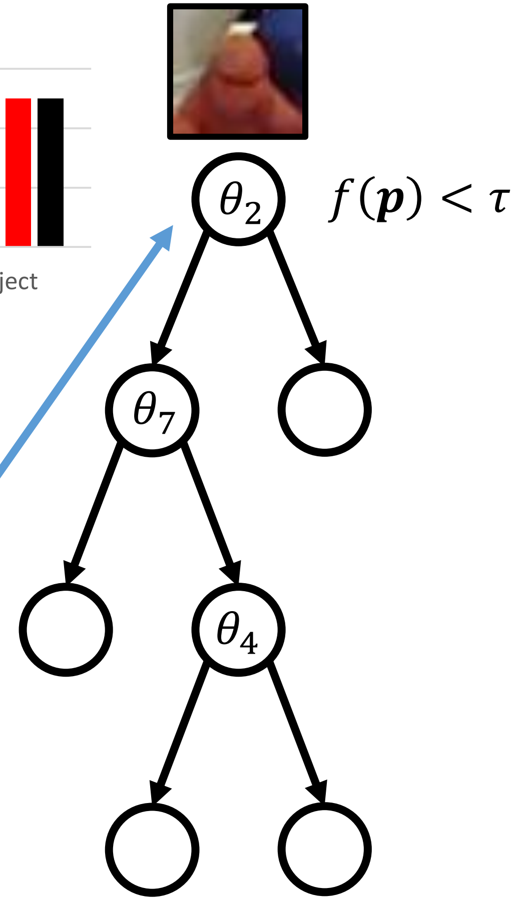
Building the Decision Forest

Sample feature parameters

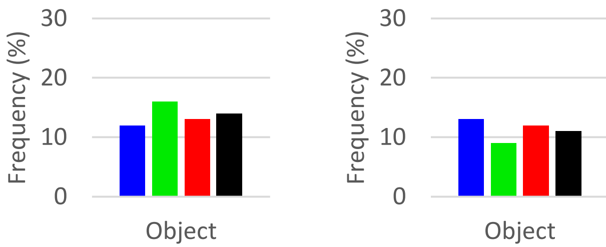
Evaluate split candiates

Apply best split



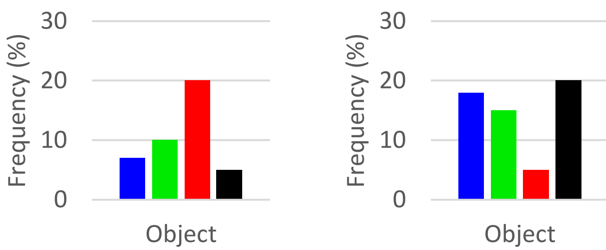


θ_1



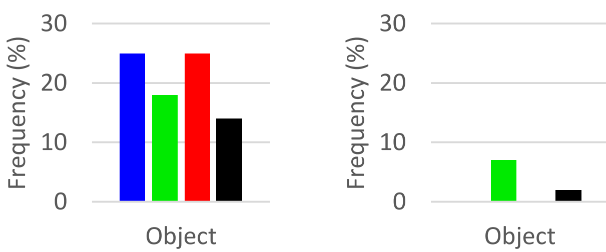
$I = 0.003$

θ_2



$I = 0.048$

θ_3



$I = 0.038$

…

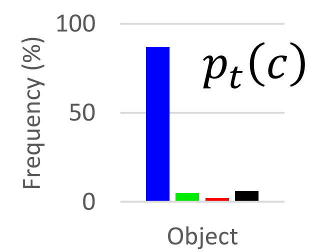
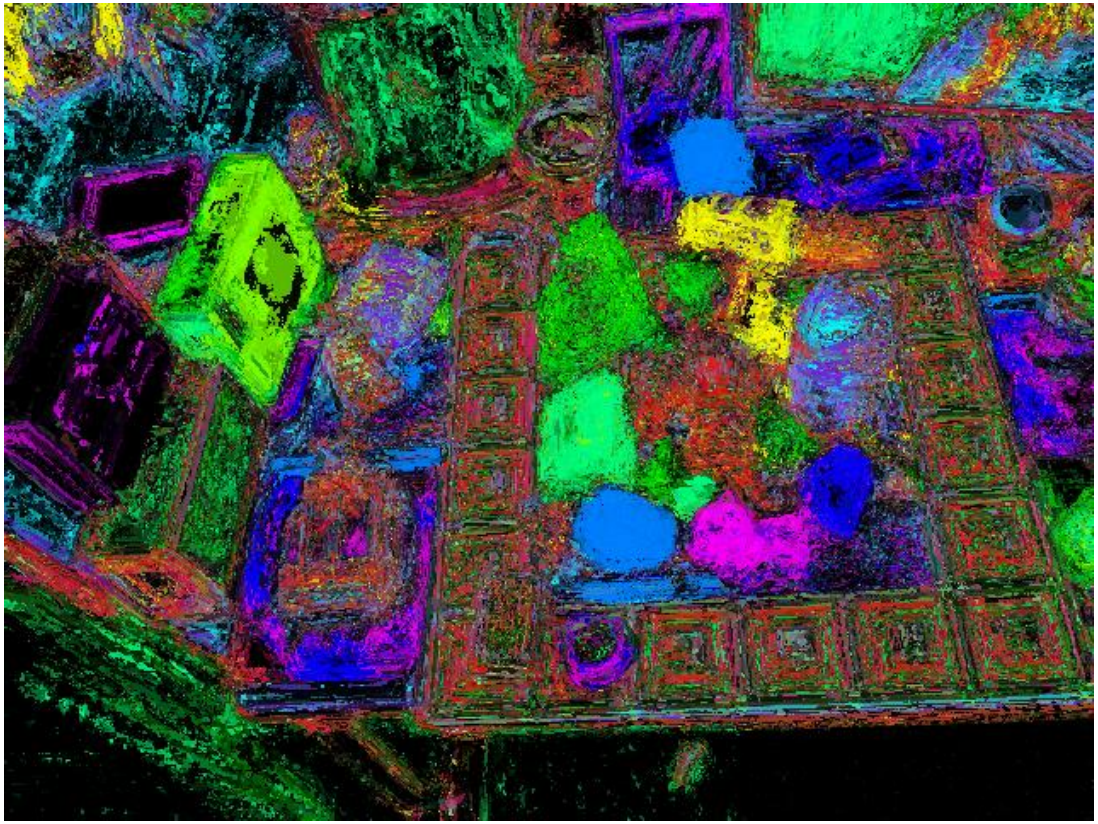


Image Segmentation Results

Input: RGB-D

Output of 1 tree: $p_t(c)$



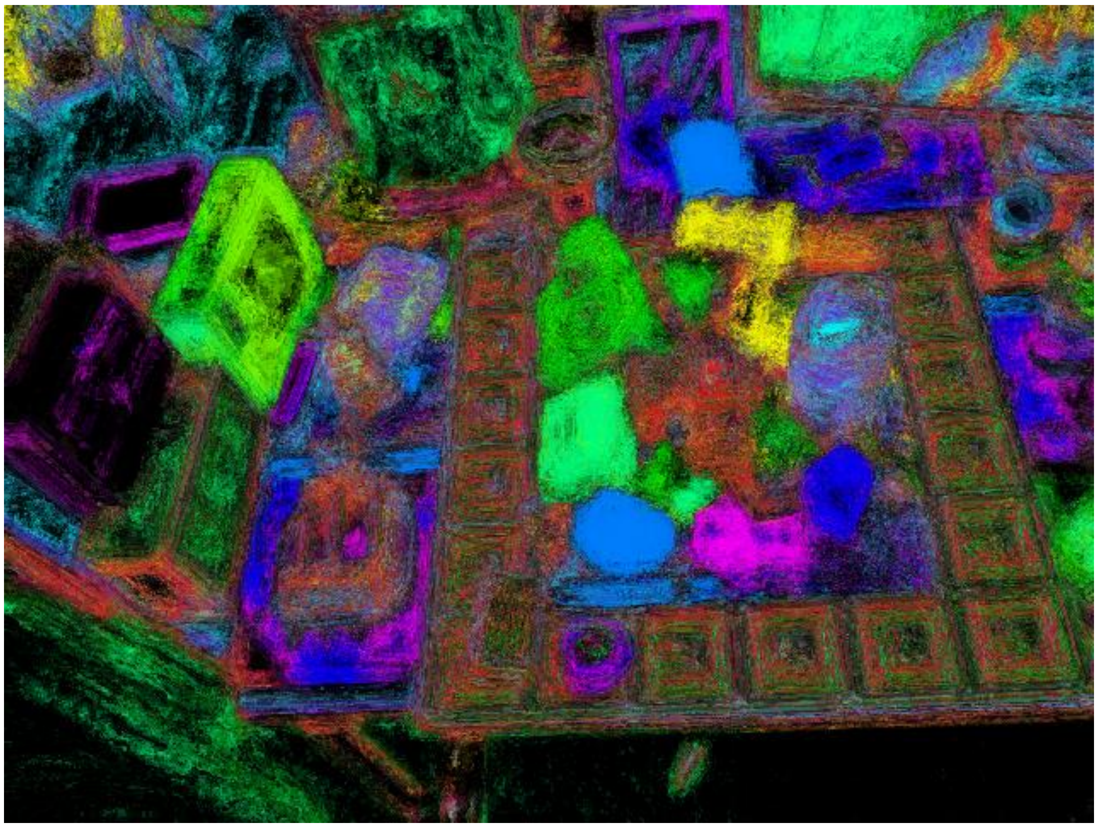
c ... object index
C ... object count
T ... tree count
$p(c)$ ... object probability
$p_t(c)$ ... object probability of tree t

Image Segmentation Results

Input: RGB-D

Output of 3 trees:

$$p(c) = \frac{1}{T}\sum_{t}^{T} p_t(c)$$



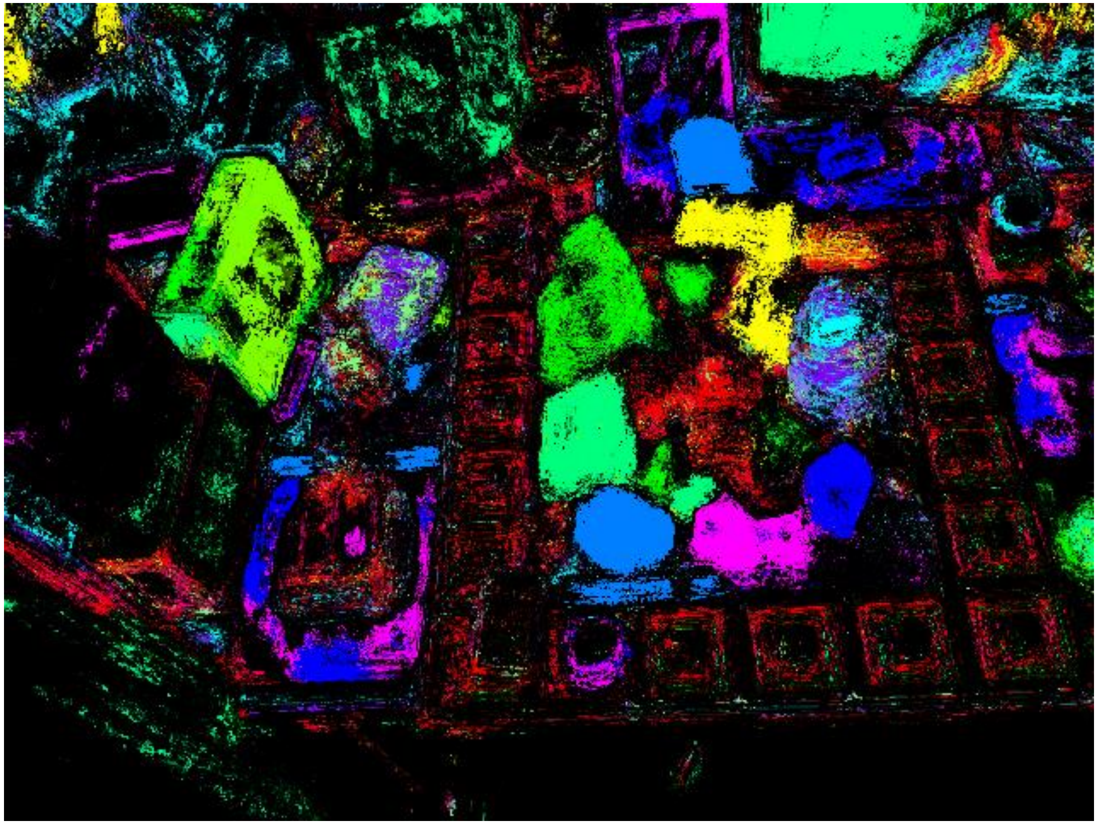
c	… object index
C	… object count
T	… tree count
$p(c)$	… object probability
$p_t(c)$	… object probability of tree t

Image Segmentation Results

Input: RGB-D

Output of 3 trees:

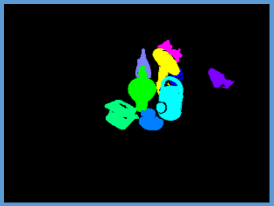
$$p(c) = \frac{\prod_t^T p_t(c)}{\sum_{c'}^C \prod_t^T p_t(c')}$$



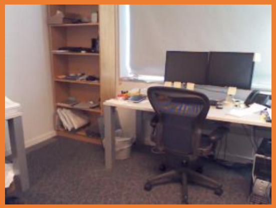
c ... object index
C ... object count
T ... tree count
$p(c)$ ... object probability
$p_t(c)$ ... object probability of tree t

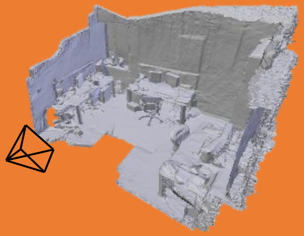
Case Study: Pose Estimation and Pose Tracking

Image Segmentation:
- Decision Forest
- Features, Training Recap



Camera Re-Localization:
- Regression Forest
- Split-Criteria, Kabsch, RANSAC

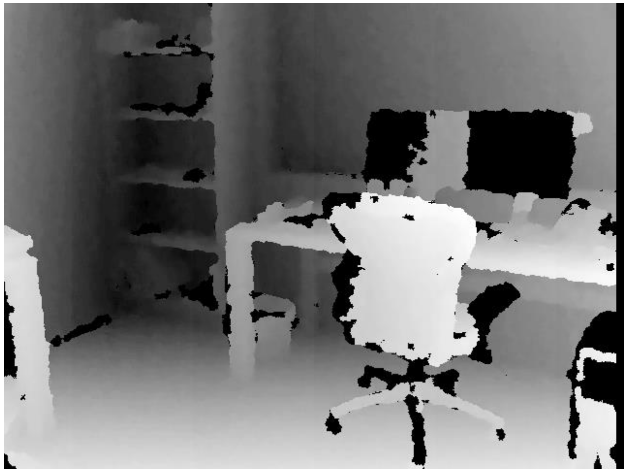


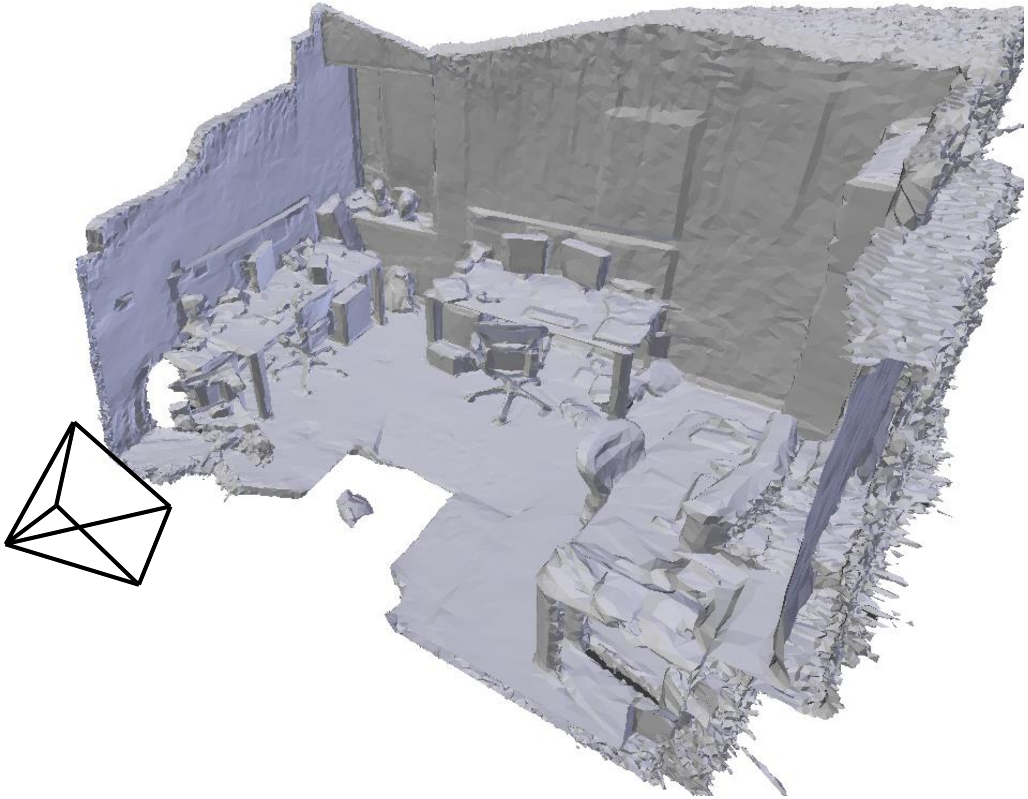


Camera Re-Localization

Input: RGB-D image

Output: 6 DoF camera pose





Dataset of Shotton et al., Scene Coordinate Regression Forests for Camera Relocalization in RGB-D Images, CVPR 13
In the following we discuss a variant of the method of this paper.

Camera Pose

$$\boldsymbol{x} = \boldsymbol{K}\ \underbrace{\boldsymbol{R}\ (\boldsymbol{I}_{3\times3}| - \widetilde{\boldsymbol{C}})}_{\text{Camera pose } (\boldsymbol{R}, \boldsymbol{C})}\boldsymbol{X}^{Obj} \ = \boldsymbol{K}\boldsymbol{X}^{Cam}$$

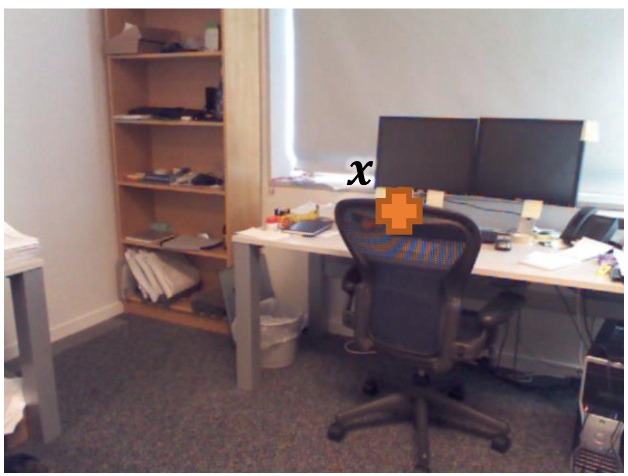
Point in camera coordinates, $\boldsymbol{X}^{Cam} = \boldsymbol{R}(\boldsymbol{I}_{3\times3}| - \widetilde{\boldsymbol{C}})\boldsymbol{X}^{Obj}$

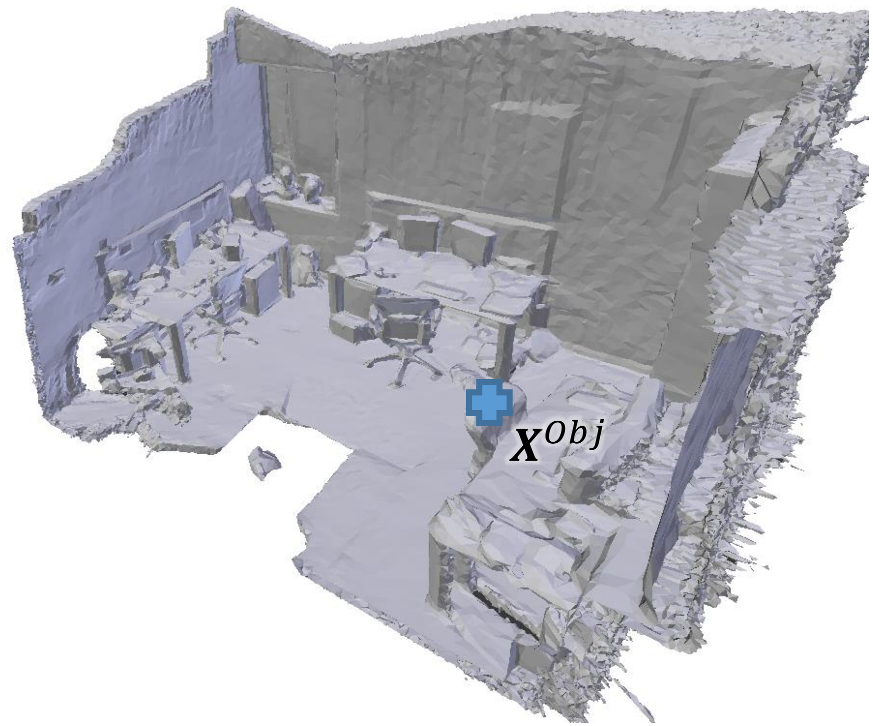
Point in object (or world) coordinates

Camera pose $(\boldsymbol{R}, \boldsymbol{C})$

Calibration matrix

Image point





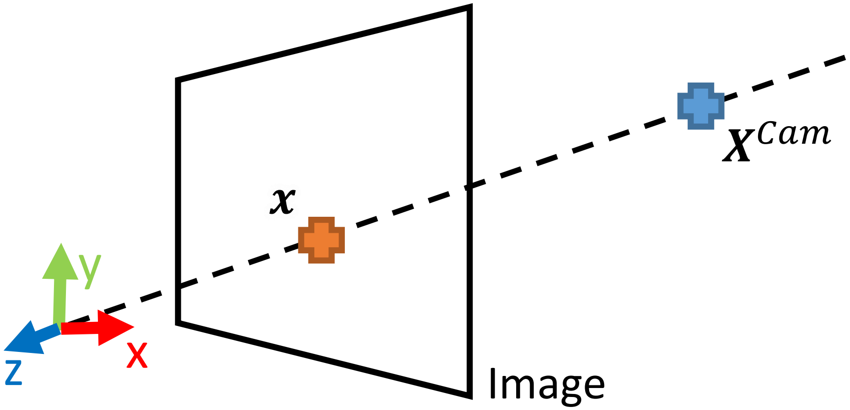
Camera Coordinates and Object Coordinates

Kinect allows us to work in 3D

Calibration matrix K is known

Pixel depth is known

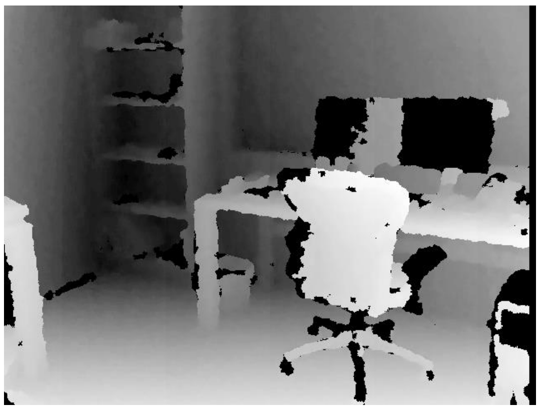
$$\boldsymbol{X}^{Cam} = \boldsymbol{K}^{-1}\boldsymbol{x}$$



Camera Coordinates and Object Coordinates

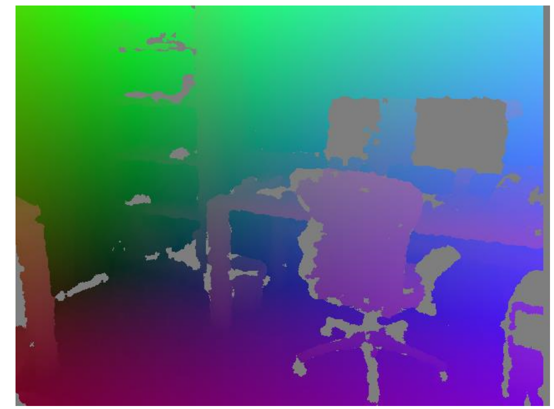
Kinect allows us to work in 3D

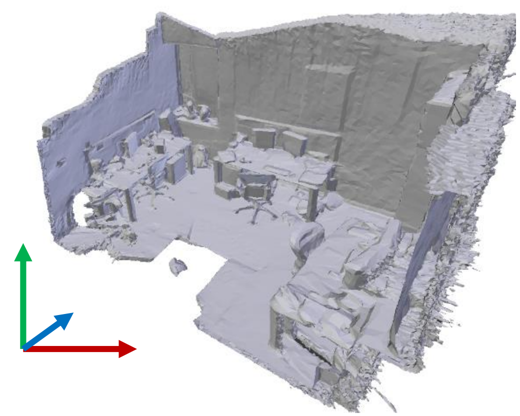
Depth: x, y, Z



$$X = \frac{xZ}{f}, Y = \frac{yZ}{f}$$

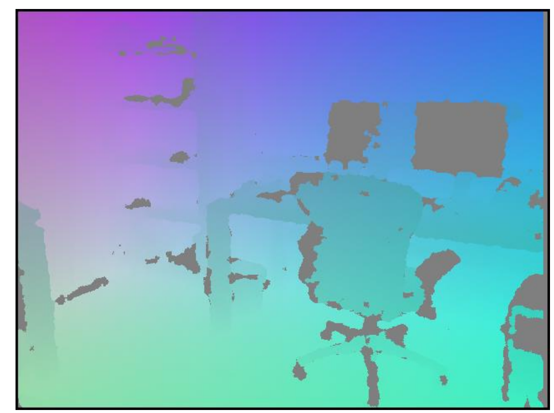
$\boldsymbol{X}^{Cam}$(XYZ mapped to RGB)





Object model

XYZ mapped to RGB



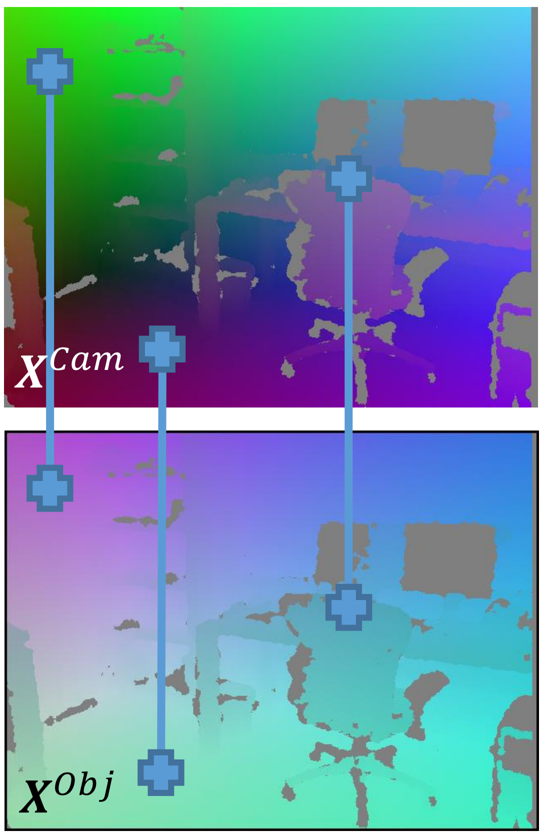
$\boldsymbol{X}^{Obj}$

Kabsch Algorithm

$$\boldsymbol{X}^{Cam} = \boldsymbol{R}(\boldsymbol{I}_{3\times3} | -\widetilde{\boldsymbol{C}})\boldsymbol{X}^{Obj}$$

Calculate camera pose $(\boldsymbol{R}, \boldsymbol{C})$ from correspondences

$\left\{(\boldsymbol{X}_1^{Cam}, \boldsymbol{X}_1^{Obj}), (\boldsymbol{X}_2^{Cam}, \boldsymbol{X}_2^{Obj}), \dots, (\boldsymbol{X}_n^{Cam}, \boldsymbol{X}_n^{Obj})\right\}$



Goal:

$$(\boldsymbol{R}^*, \boldsymbol{C}^*) = \underset{\boldsymbol{R},\boldsymbol{C}|\boldsymbol{R}\boldsymbol{R}^T=1}{\text{minarg}} \sum_{i=1}^{n} \left\| \boldsymbol{X}_i^{Cam} - \left(\boldsymbol{R}\boldsymbol{X}_i^{Obj} - \boldsymbol{C}\right) \right\|^2$$

Algorithm:

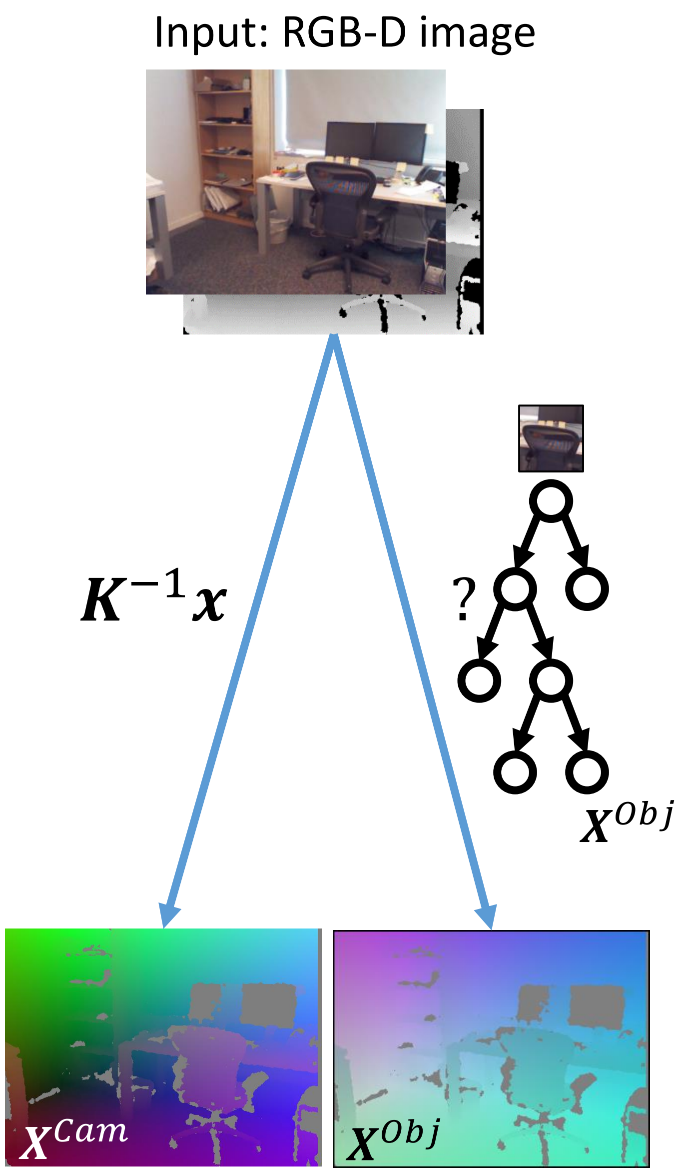
1) $\overline{\boldsymbol{X}}^{Cam} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{X}_i^{Cam}, \overline{\boldsymbol{X}}^{Obj} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{X}_i^{Obj}$

2) $cov\left(\boldsymbol{X}^{Cam}, \boldsymbol{X}^{Obj}\right) = \sum_{i=1}^{n}\left(\boldsymbol{X}_i^{Obj} - \overline{\boldsymbol{X}}^{Obj}\right)\left(\boldsymbol{X}_i^{Cam} - \overline{\boldsymbol{X}}^{Cam}\right)^T$

3) $cov\left(\boldsymbol{X}^{Cam}, \boldsymbol{X}^{Obj}\right) = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$

4) $\boldsymbol{R}^* = \boldsymbol{V}\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \det(\boldsymbol{V}\boldsymbol{U}^T) \end{pmatrix}\boldsymbol{U}^T$ 5) $\boldsymbol{C}^* = \boldsymbol{R}^*\overline{\boldsymbol{X}}^{Obj} - \overline{\boldsymbol{X}}^{Cam}$

Object Coordinate Regression



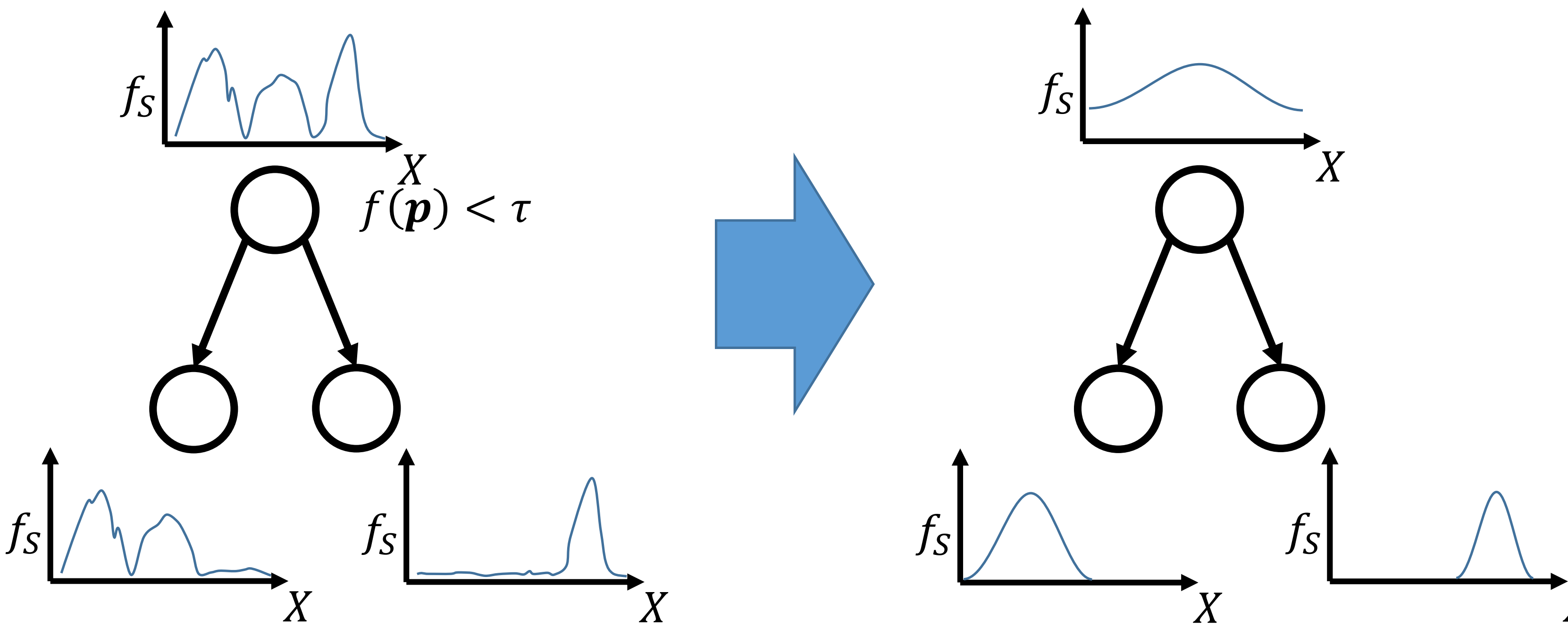
- Regression Tree
  - Continuous Output
- What has to be defined?
  - Features: f^{DA-D}, f^{DA-RGB}
  - Split score: ?
  - Stopping criterium: Max. tree depth
  - Leaf distributions: ?

Regression Split Scores

Gaussian distributions:

$$f_S(\boldsymbol{X}) = \mathcal{N}(\boldsymbol{X}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{\sqrt{2\pi}^3 \sqrt{|\boldsymbol{\Sigma}|}} e^{-\frac{1}{2}(\boldsymbol{X}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{X}-\boldsymbol{\mu})}$$

S	... Point Set
μ	... mean
$\boldsymbol{\Sigma}$	... covariance
$I(S)$	... information gain
$H(S)$	... entropy

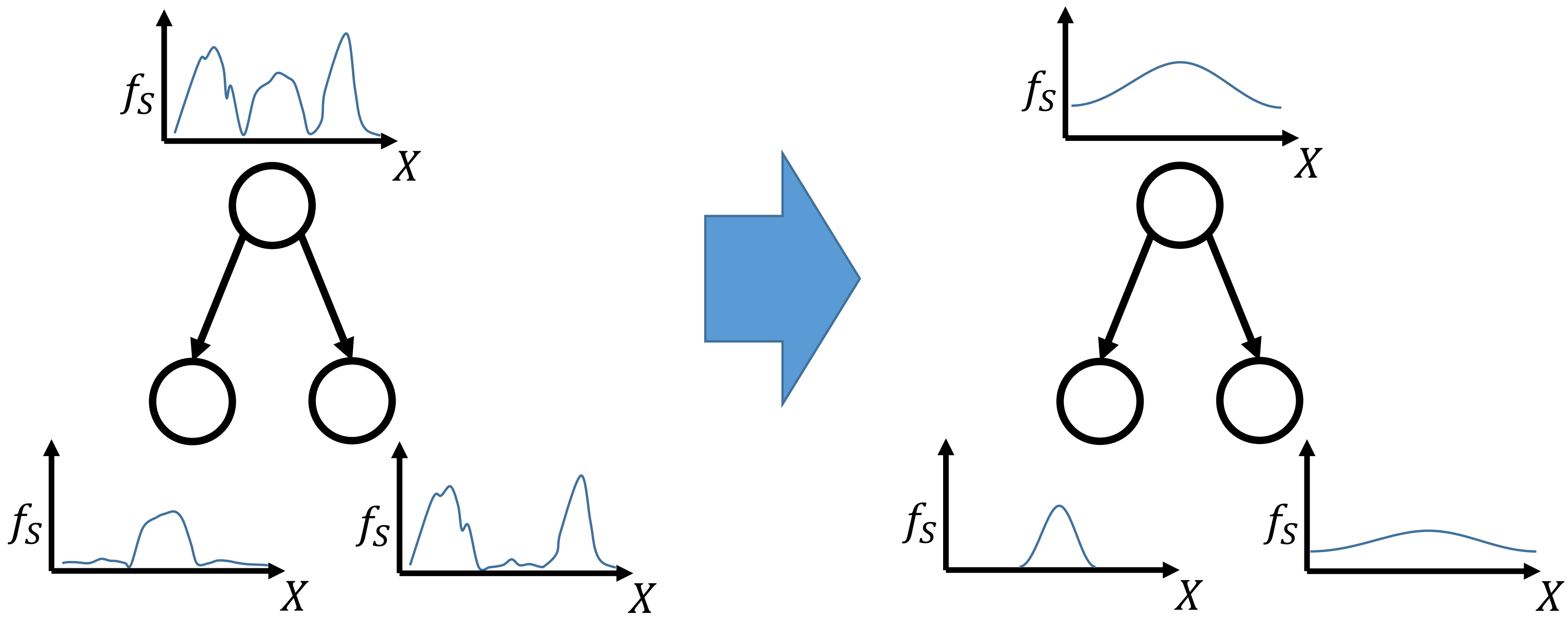


$$I(S) = H(S) - \sum_{i \in \{L,R\}} \frac{|S^i|}{|S|} H(S^i)$$

$$H(S) = \frac{1}{2} \ln\{(2\pi e)^3 |\Sigma|\}$$

Regression Split Scores

Gaussian distributions:



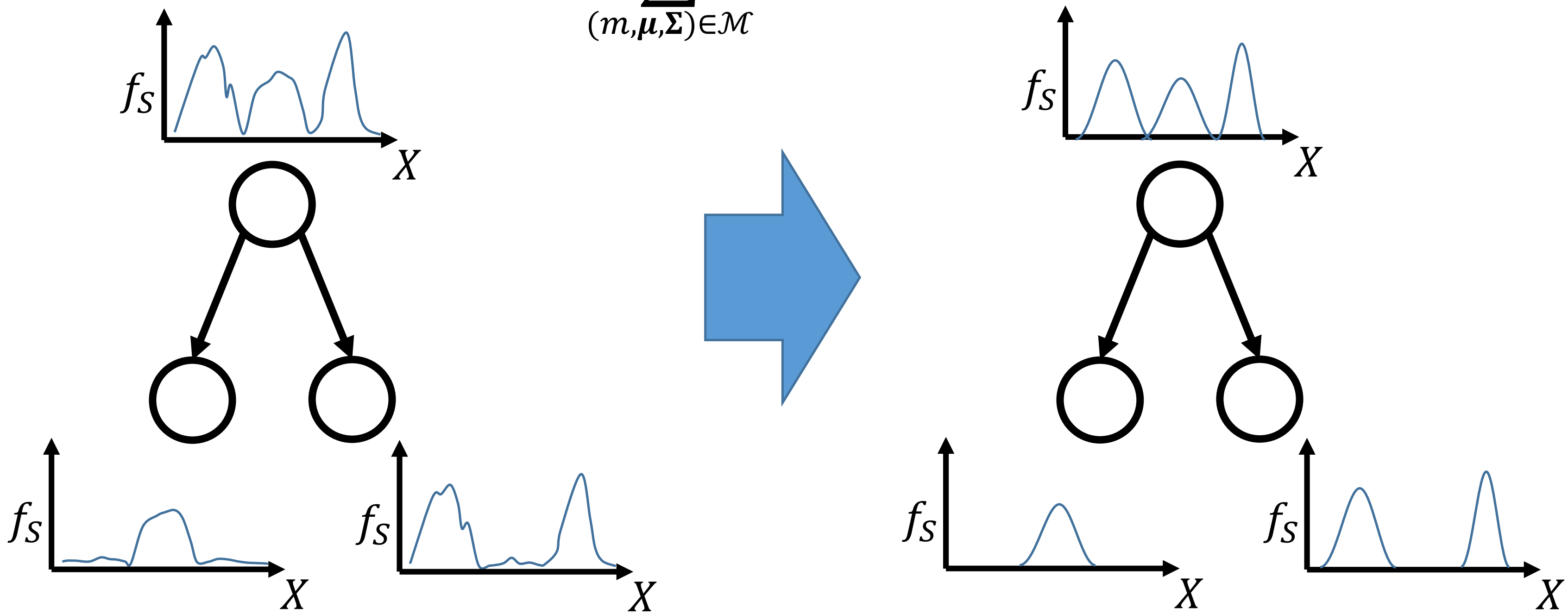
Properties:

- Fast
- Not very powerful (uni-modal)

Regression Split Scores

Mixture of Gaussians: $f_S(\boldsymbol{X}) = \sum_{(m,\boldsymbol{\mu},\boldsymbol{\Sigma})\in\mathcal{M}} \mathcal{N}(\boldsymbol{X}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$

$\mathcal{M}$ ... mixture components
m ... mixture weight



Properties:

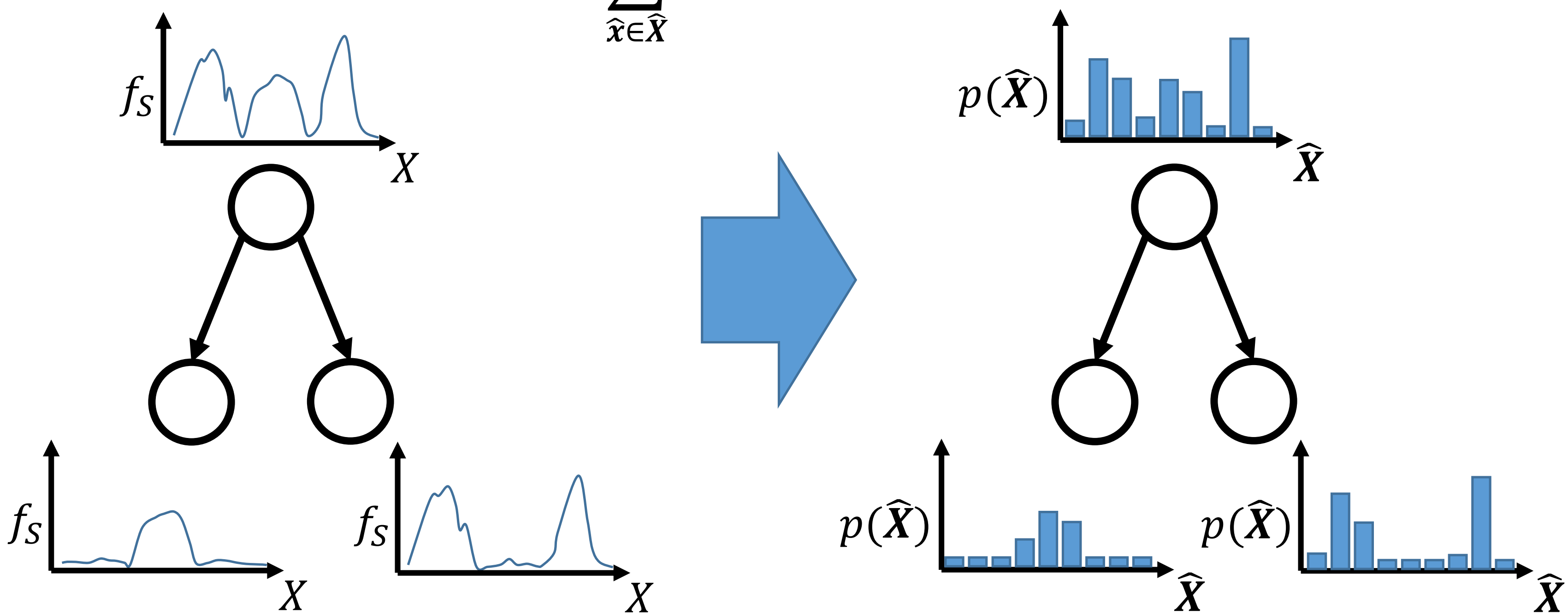
- Powerful (multi-modal)
- Very Slow
  - Expectation Maximation or Mean-Shift
  - For each feature candidate (10^3) for each node $(10^6) = 10^9$ distributions

Regression Split Scores

$\hat{\boldsymbol{X}}$... discretized object coordinate

Proxy classes:

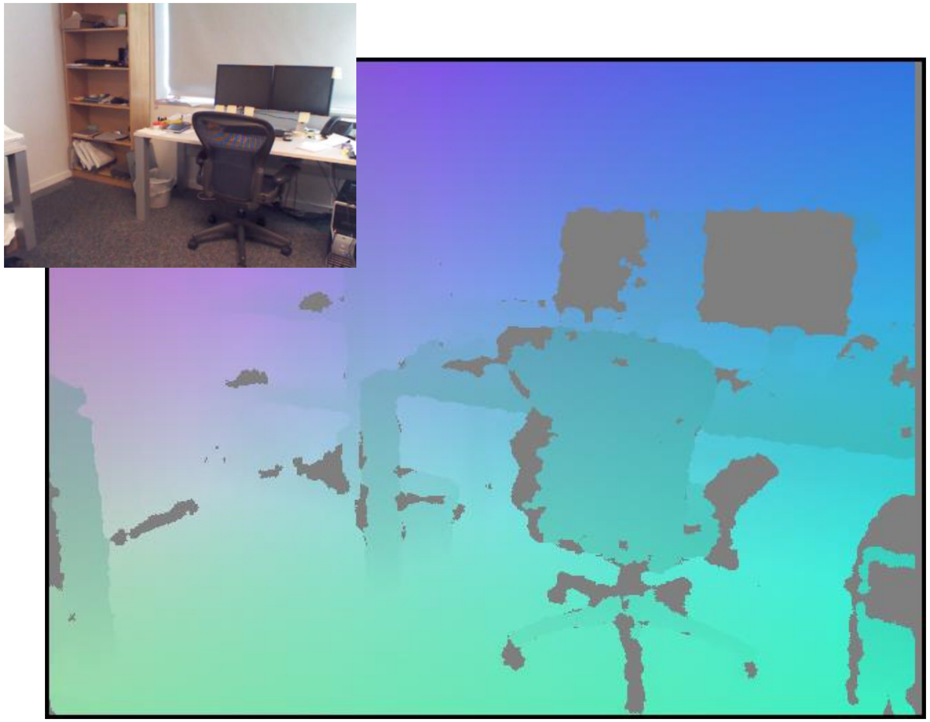
$$H(S) = -\sum_{\hat{x} \in \hat{\boldsymbol{X}}} p(\hat{x}) \log p(\hat{x})$$



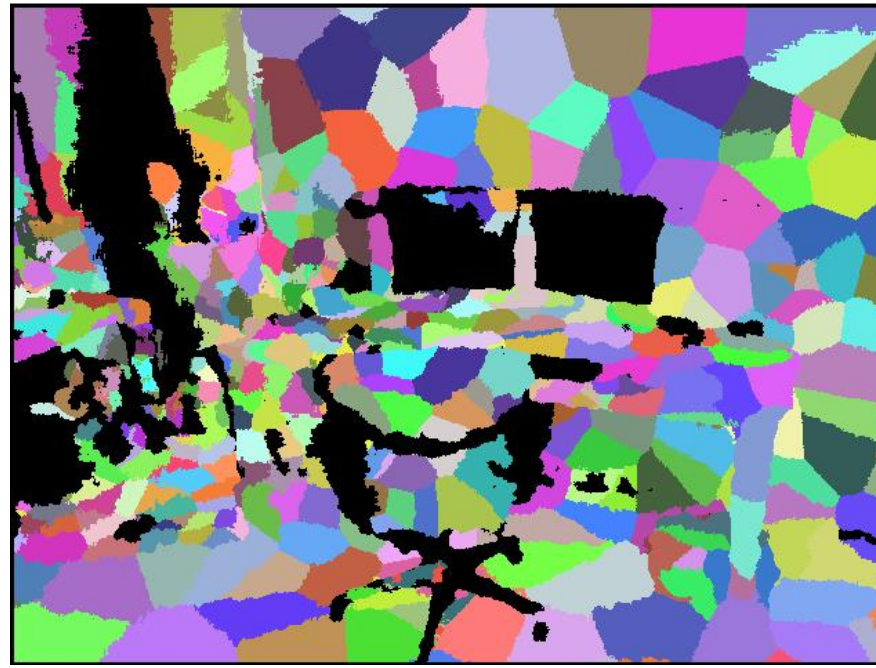
Properties:

- Powerful (multi-modal)
- Fast

Proxy Classes

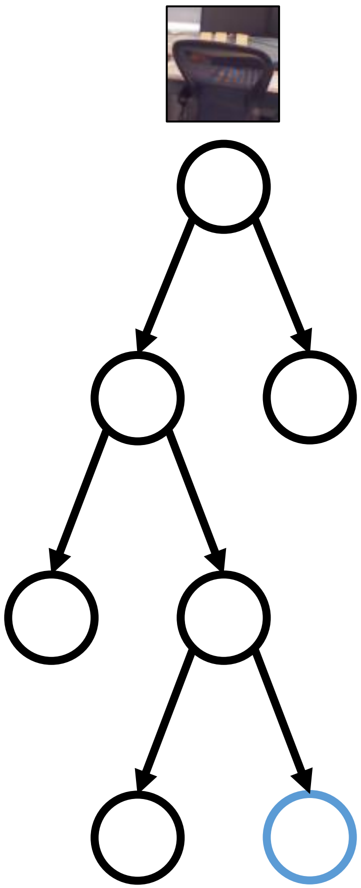


$\boldsymbol{X}^{Obj}$



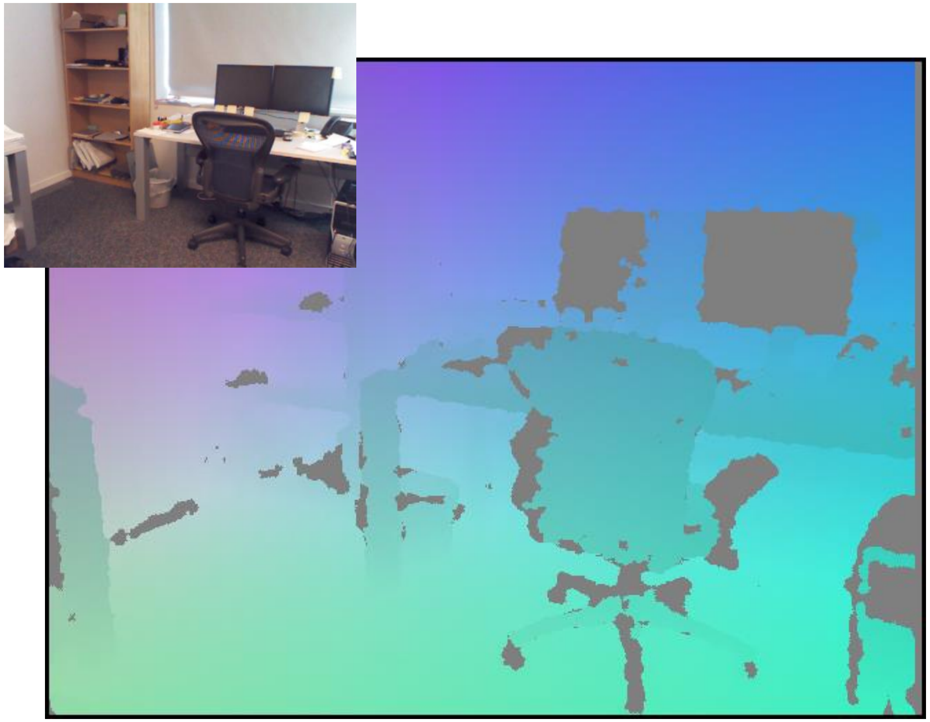
$\hat{\boldsymbol{X}}^{Obj}$

Leaf Distributions



- Find modes
- Store mean of largest mode

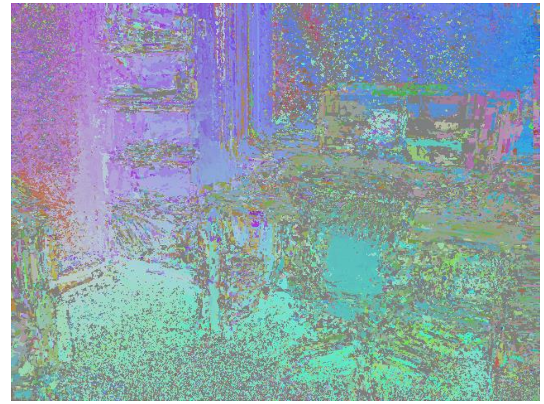
Object Coordinate Regression Results



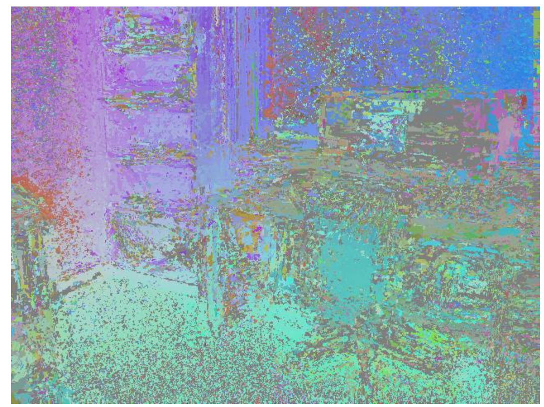
$\boldsymbol{X}^{Obj}$ (ground truth)

Problem: Outliers

Solution: RANSAC



$\boldsymbol{X}^{Obj}$ (tree 1)



$\boldsymbol{X}^{Obj}$ (tree 2)

$\boldsymbol{X}^{Obj}$ (tree 3)