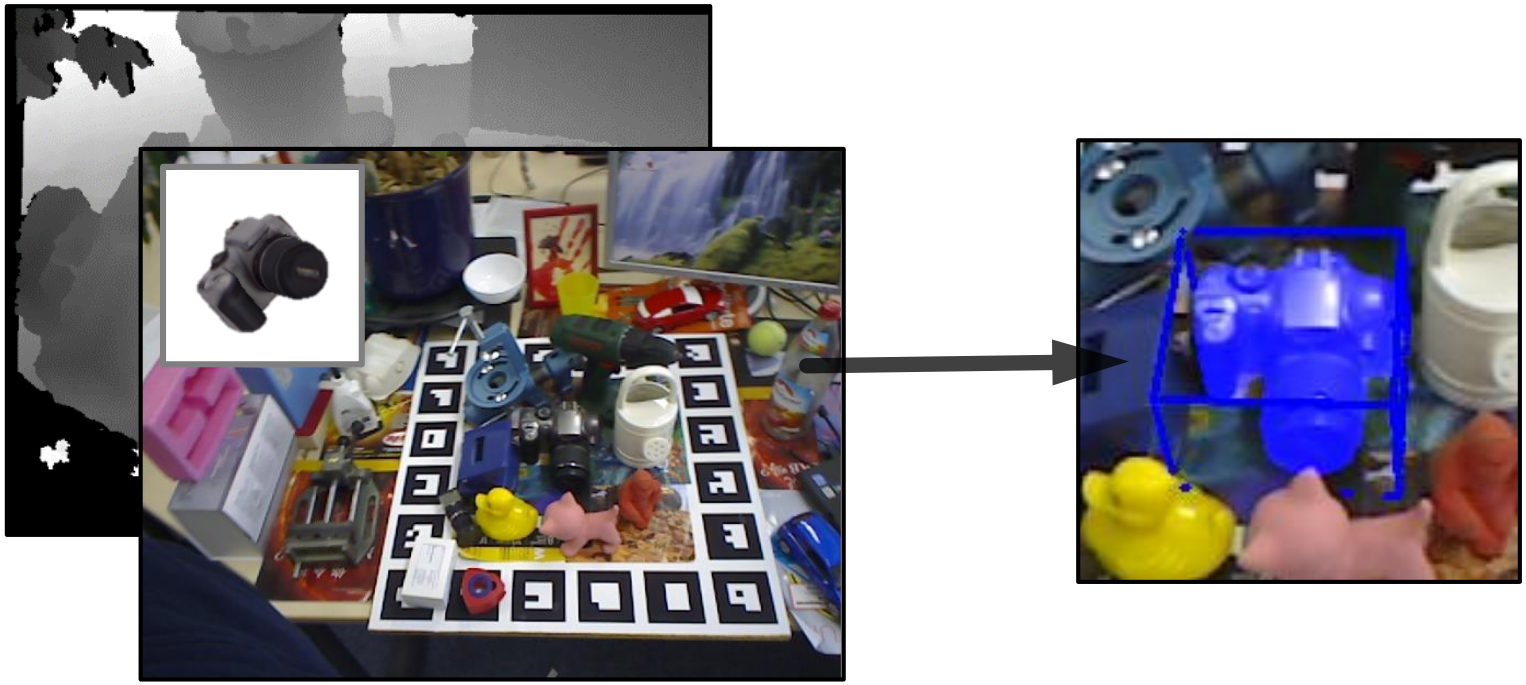
Learning 6D Object Pose Estimation and Tracking

Carsten Rother

presented by:

Alexander Krull

6D Pose Estimation



Input:

- RGBD-image
- Known 3D model

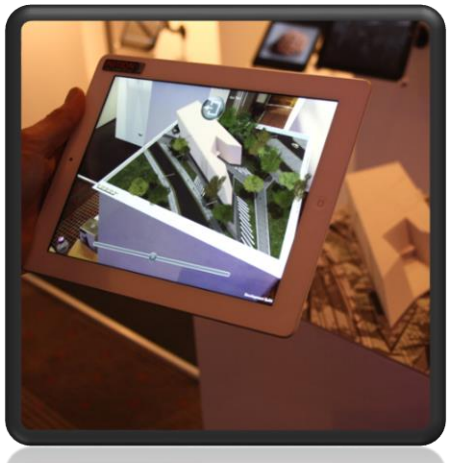
Output:

- 6D rigid body transform of object

Application Scenarios

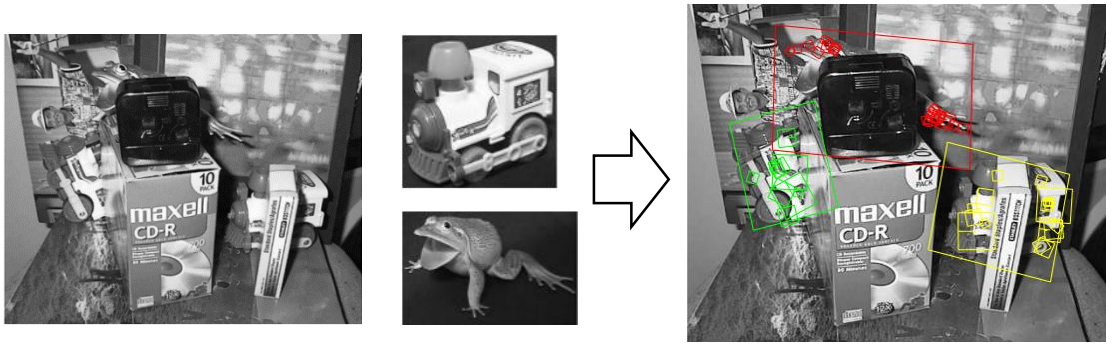
- Robotics
  - Recognition/Tracking
  - Automatic Grasping

- Augmented Reality
  - Alteration
  - Annotation
  - Substitution

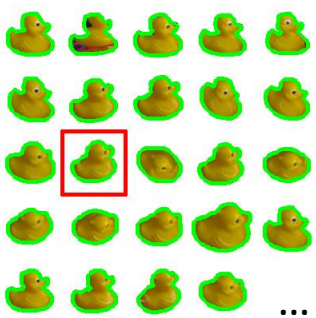


Possible Approaches

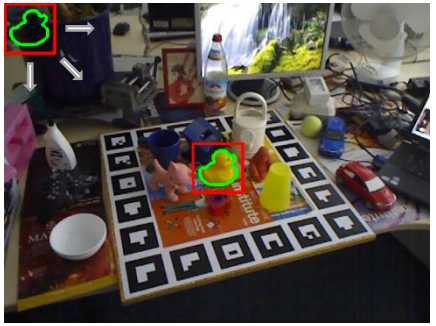
- Sparse features (Lowe 1999, 2004)
  - For textured objects



- Templates (Hinterstoisser et al. 2010, 2012)
  - For texture-less objects



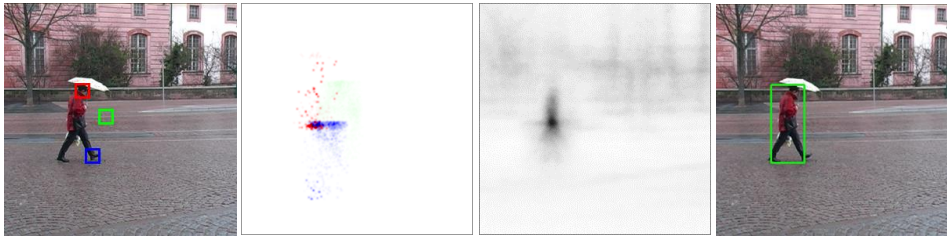
Templates



Sliding Window

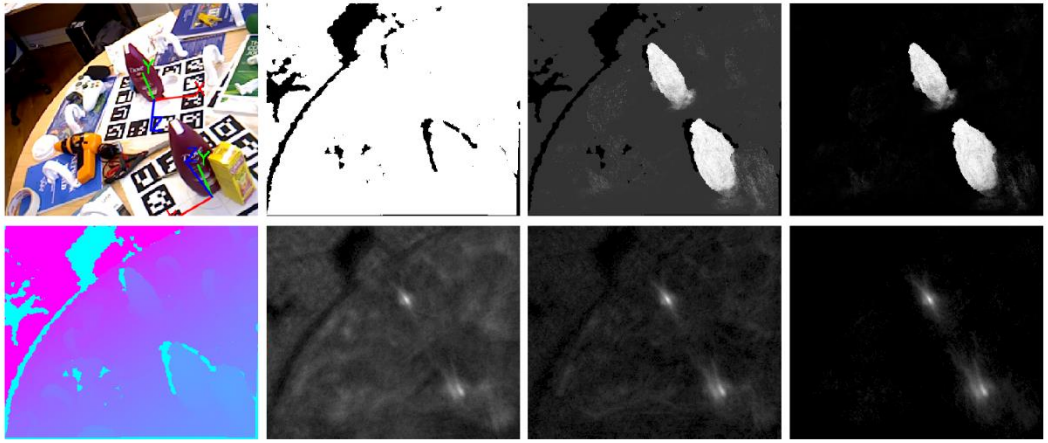
Possible Approaches

- Dense voting approaches:
  - Hough forests (Gall et al. 2009)



Input | Example Votes | Hough Space | Output

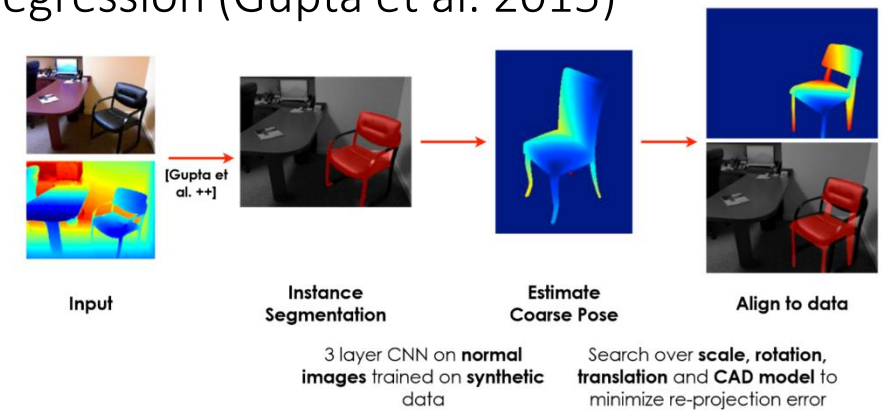
  - Latent-Class Hough Forests (Tejani et al. 2014)



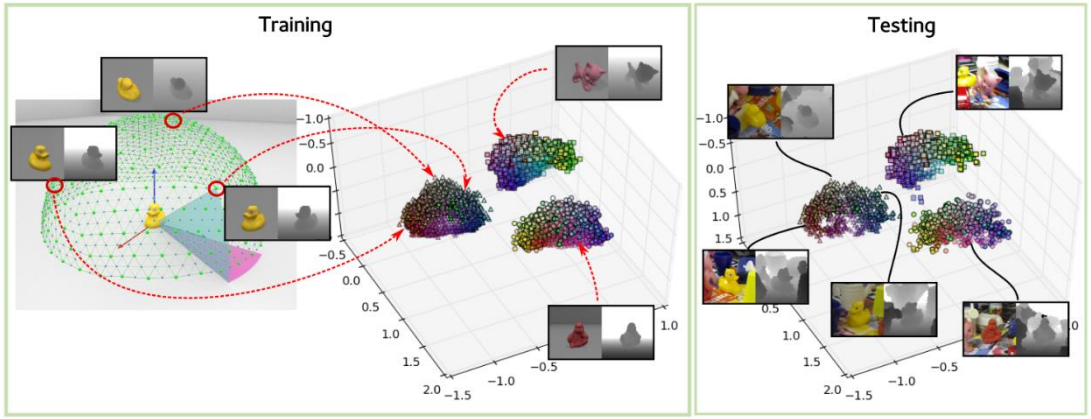
RGB-D images | #iterations=0 | #iterations=2 | #iterations=10

Possible Approaches

- Deep Learning:
  - CNN for pose regression (Gupta et al. 2015)

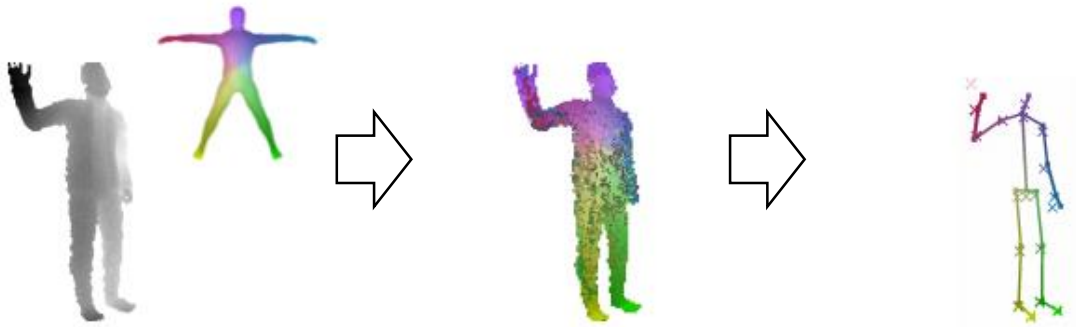


  - CNN maps to descriptor space (Wohlhart et al. 2015)



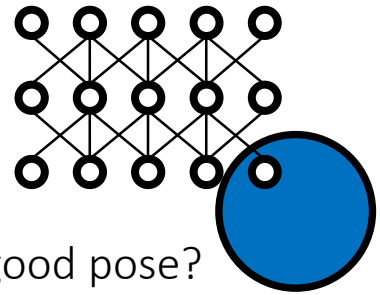
Possible Approaches

- Dense Intermediate respresentation: Vitruvian manifold (Taylor et al. 2012)



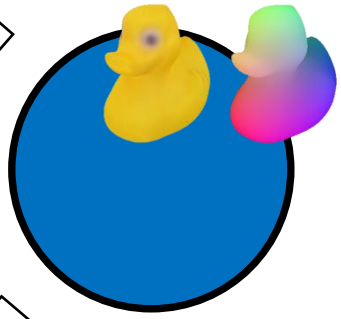
- This is the direction we take...

Overview



What is a good pose?
Can a CNN help?
(Krull, 2015)

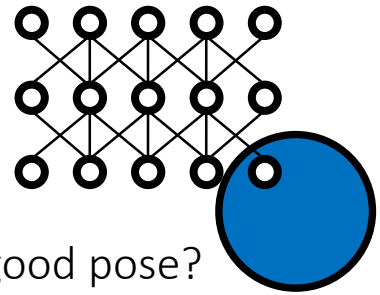
How to deal with articulted objects? (Michel, 2015)



What about tracking? (Krull, 2014)

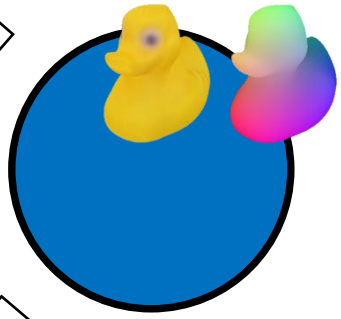
Object ooordinates for pose estimation
(Brachmann, 2014)

Overview



What is a good pose?
Can a CNN help?
(Krull, 2015)

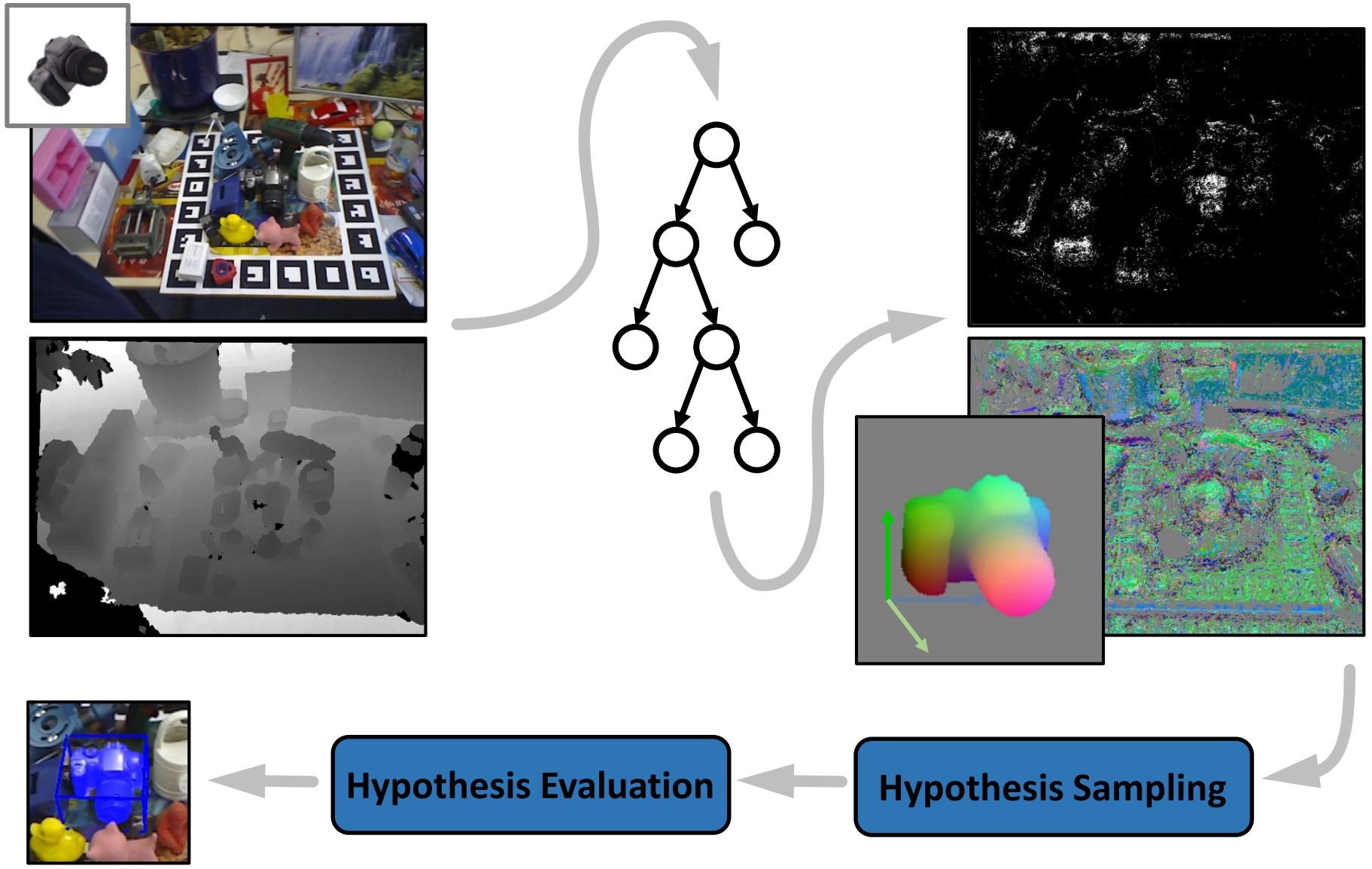
How to deal with articulted objects? (Michel, 2015)



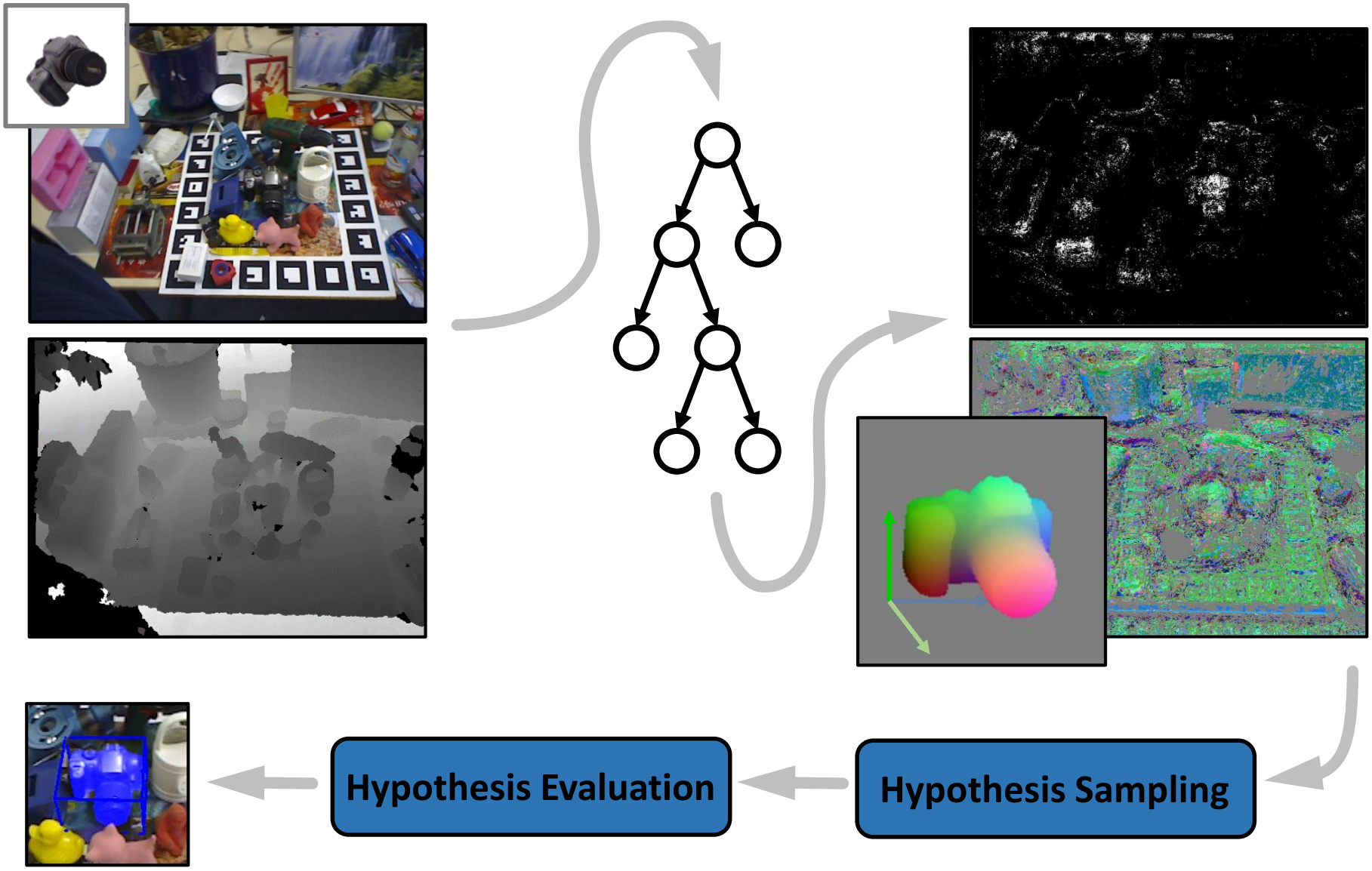
What about tracking? (Krull, 2014)

Object ooordinates for pose estimation
(Brachmann, 2014)

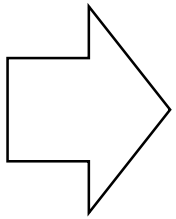
Object Coordinate Regression

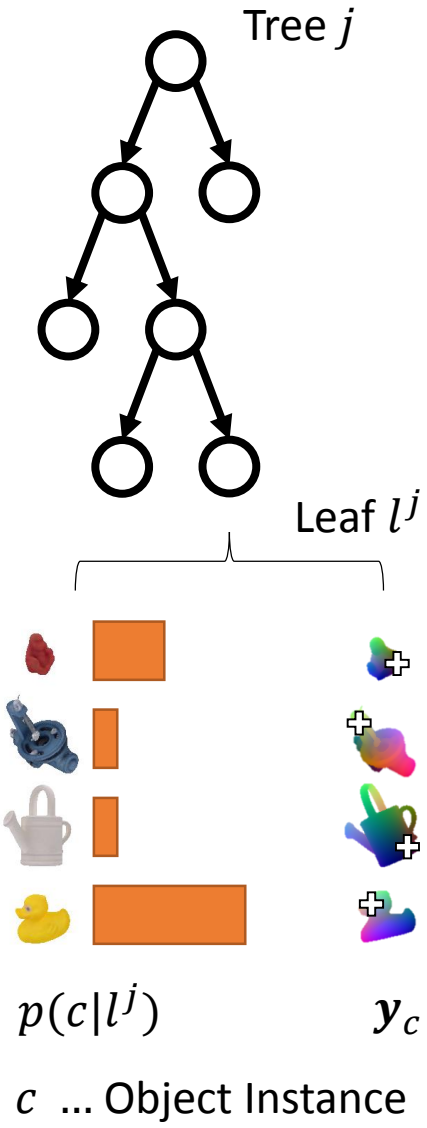


Object Coordinate Regression



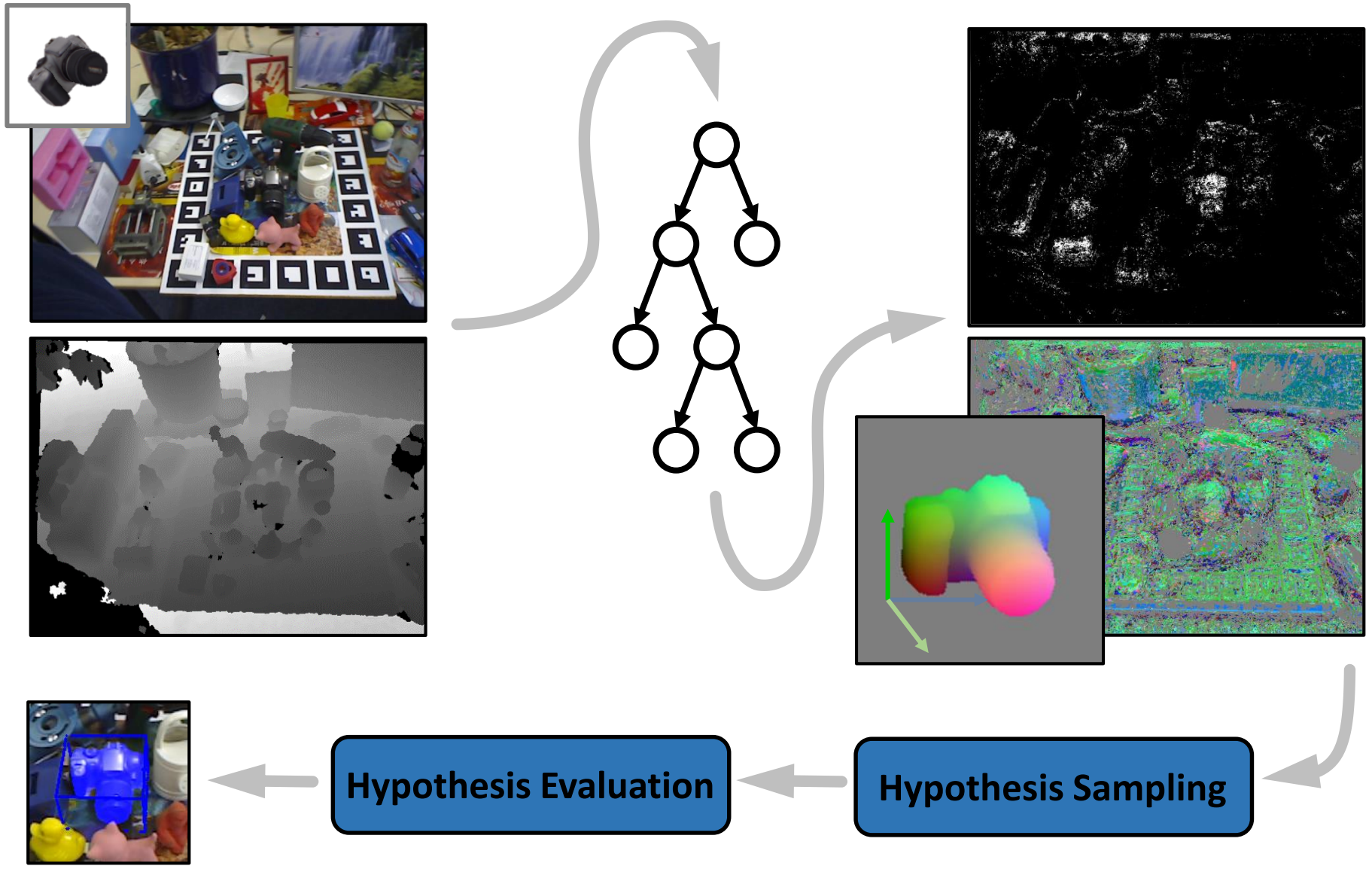
Forest Training



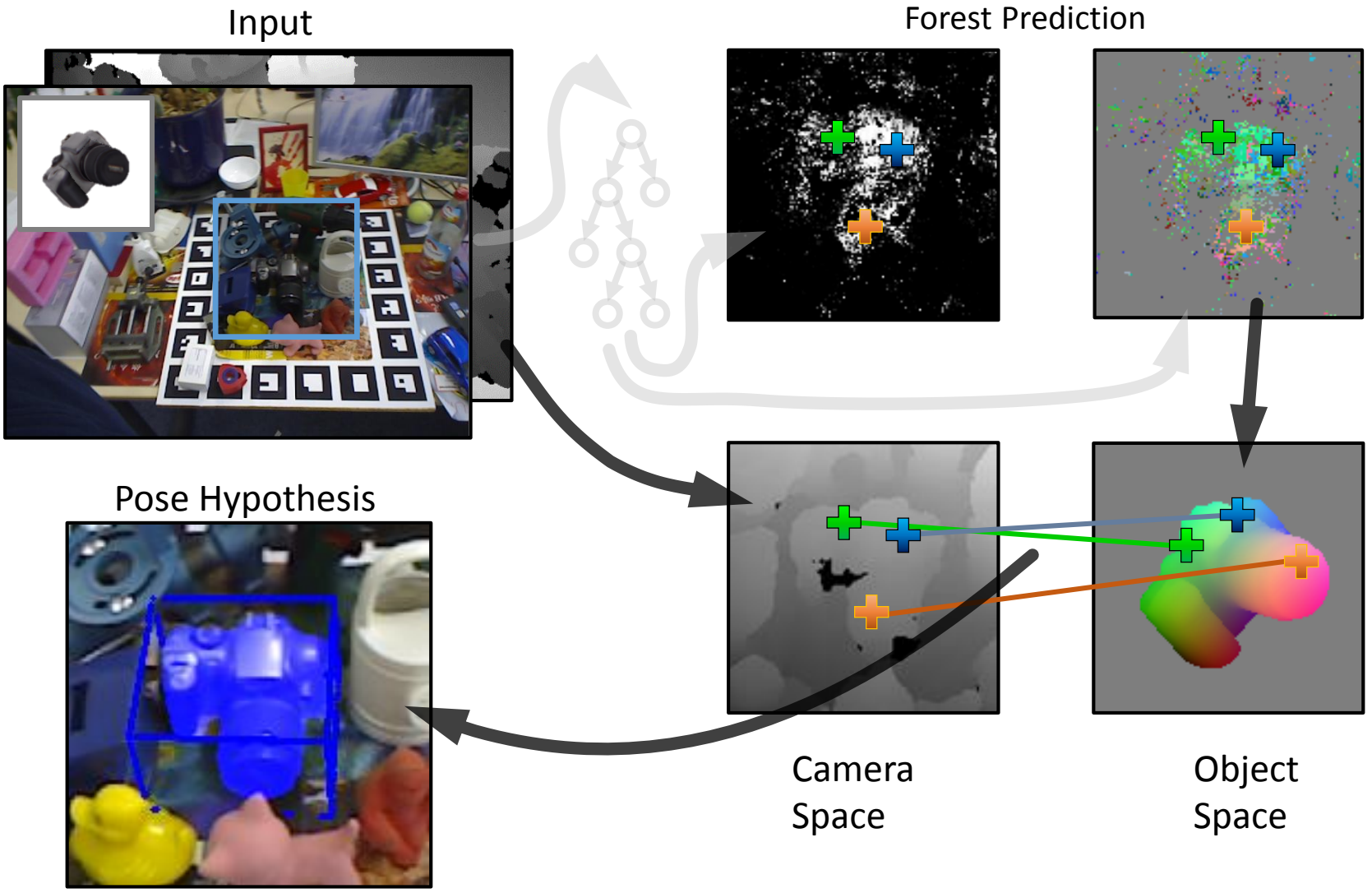


- Learns mapping of RGB-D patches to
  - Object instance probabilities $p(c|l^j)$
  - Object coordinates $\boldsymbol{y}_c$
- Randomized training procedure
  - Simple pixel difference features
  - Information gain over discretized object coordinates

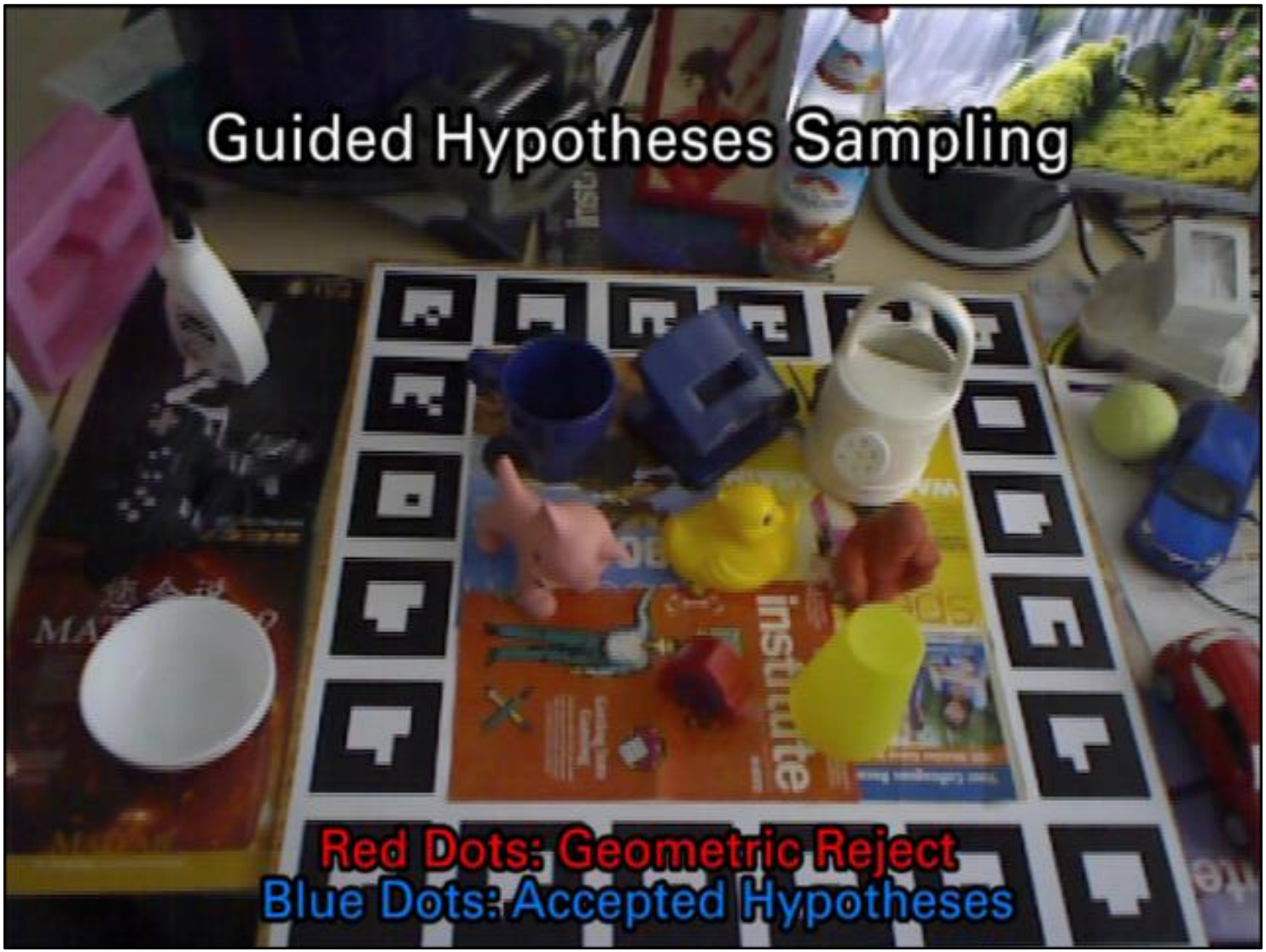
Object Coordinate Regression



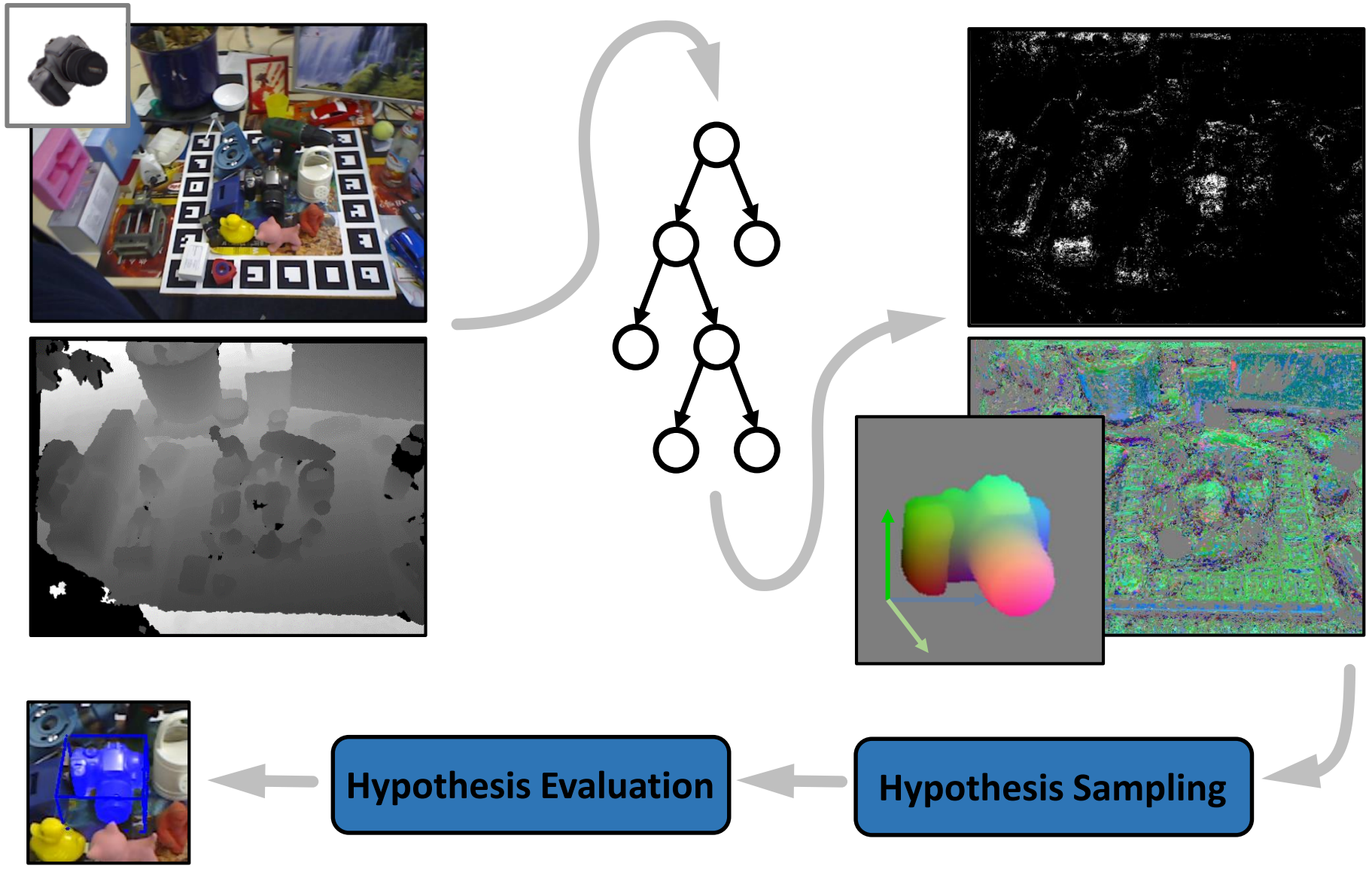
Hypothesis Sampling



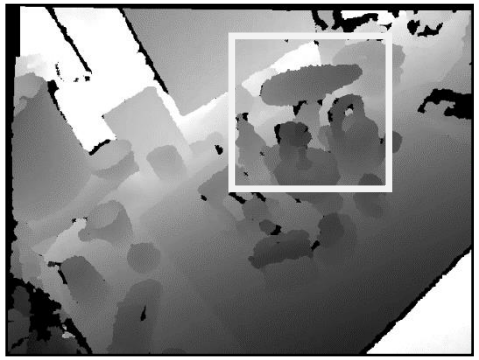
Hypothesis Sampling

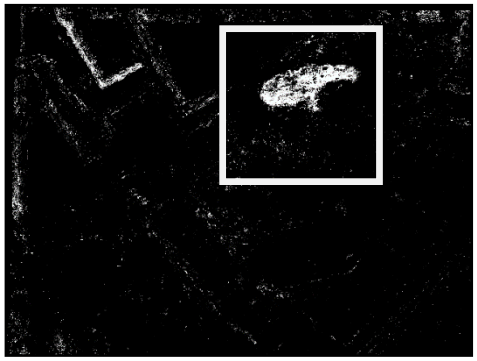


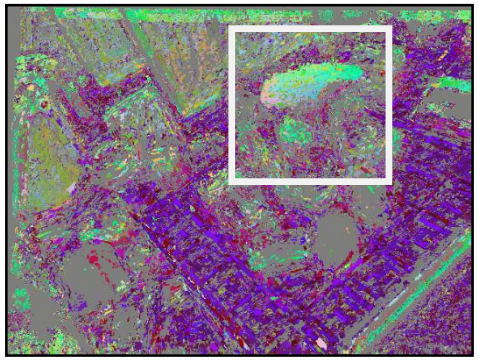
Object Coordinate Regression

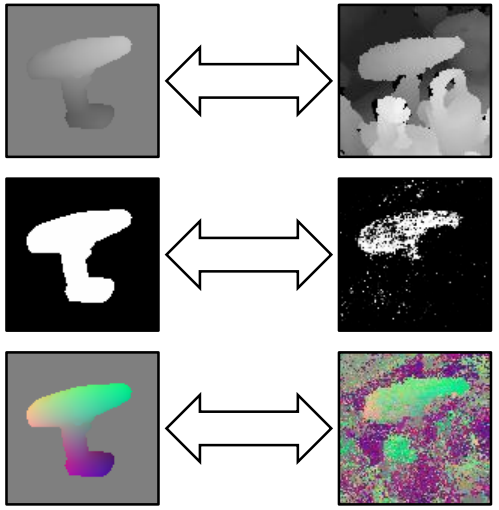


Hypothesis Evaluation







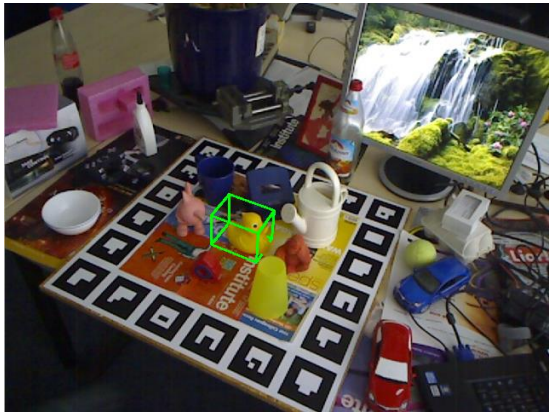


rendered observed

$$E(H) = \frac{\sum_{i \in M} d_i(\dots)}{|M|}$$

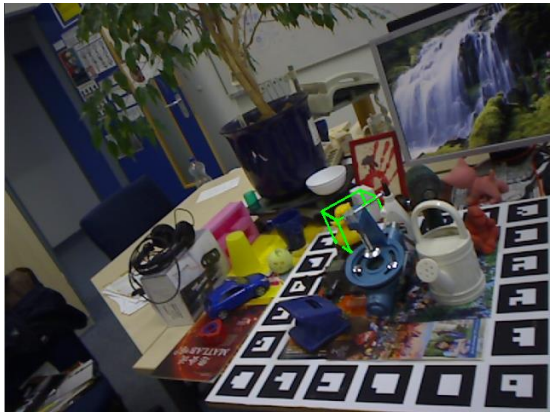
Experimental Results

No Occlusion:



	Correctly Estimated Poses		
	Linemod[1]	DTT-3D[2]	Our
Avg.	96.6%	97.2%	**98.3%**
Max.	99.9%	99.8%	**100.0%**
Min.	91.8%	94.2%	**95.8%**

Occlusion:

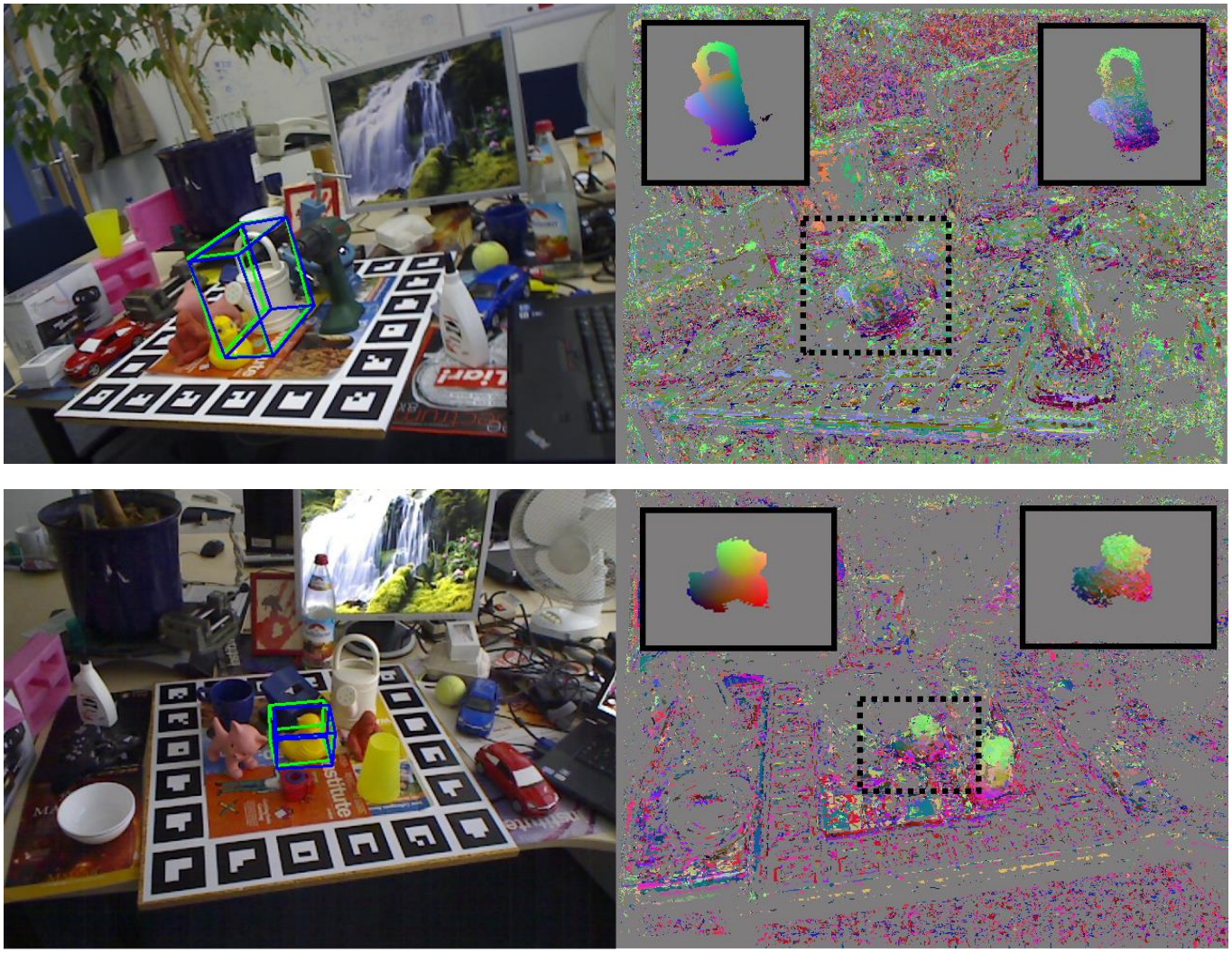


	Correctly Estimated Poses	
	Linemod[1]	Our
Avg.	54.4%	**67.3%**
Max.	98.7%	**100.0%**
Min.	**23.3%**	8.5%

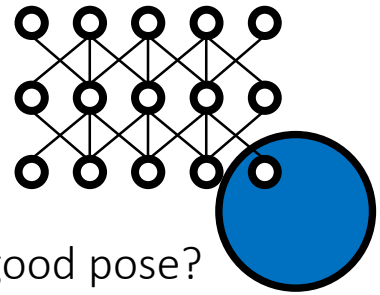
[1] Hinterstoisser et al., Model Based Training, Detection and Pose Estimation of Texture-Less 3D Objects in Heavily Cluttered Scenes, ACCV 12
[2] Rios-Cabrera et al., Discriminatively Trained Templates for 3D Object Detection: A Real Time Scalable Approach, ICCV 13

Experimental Results

Qualitative Results Ground Truth Best in Forest

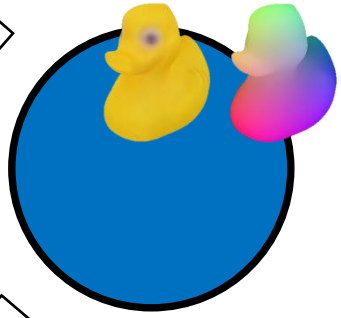


Overview



What is a good pose?
Can a CNN help?
(Krull, 2015)

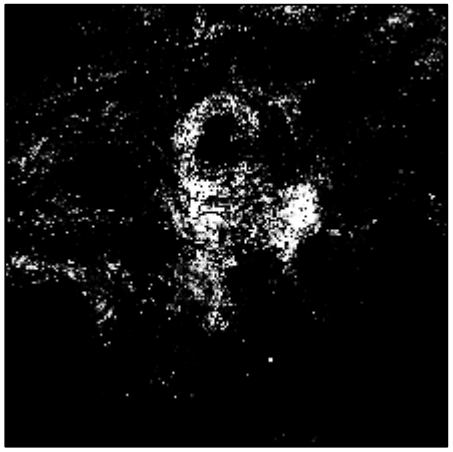
How to deal with articulted objects? (Michel, 2015)

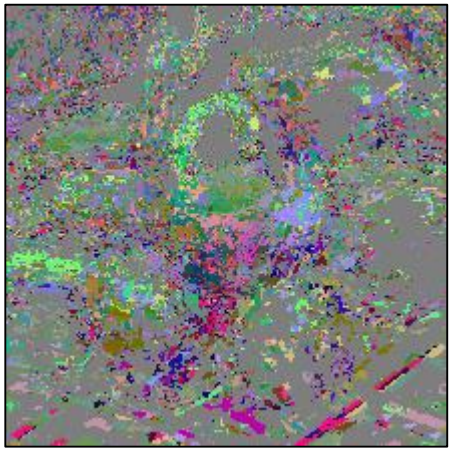


What about tracking? (Krull, 2014)

Object ooordinates for pose estimation
(Brachmann, 2014)

Challenges (and Chances) for the Energy Function

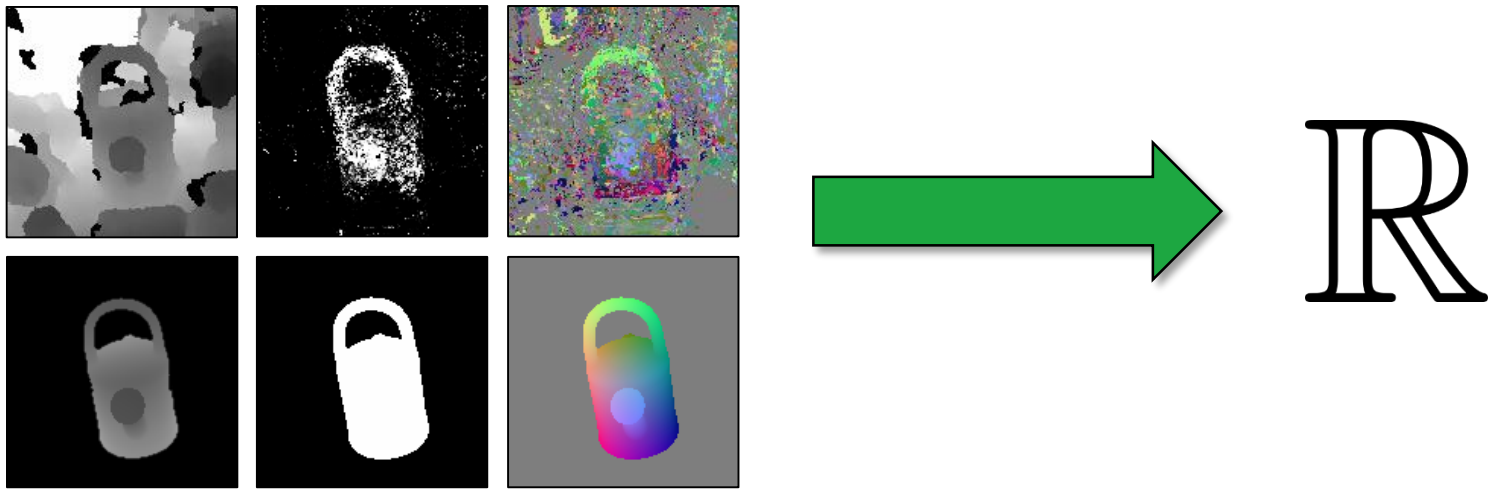




- occlusion
- missing depth values from:
  - depth shadows
  - poor reflection
  - steep angles
- noisy forest prediction
- all Issues are interconnected!
- very complex

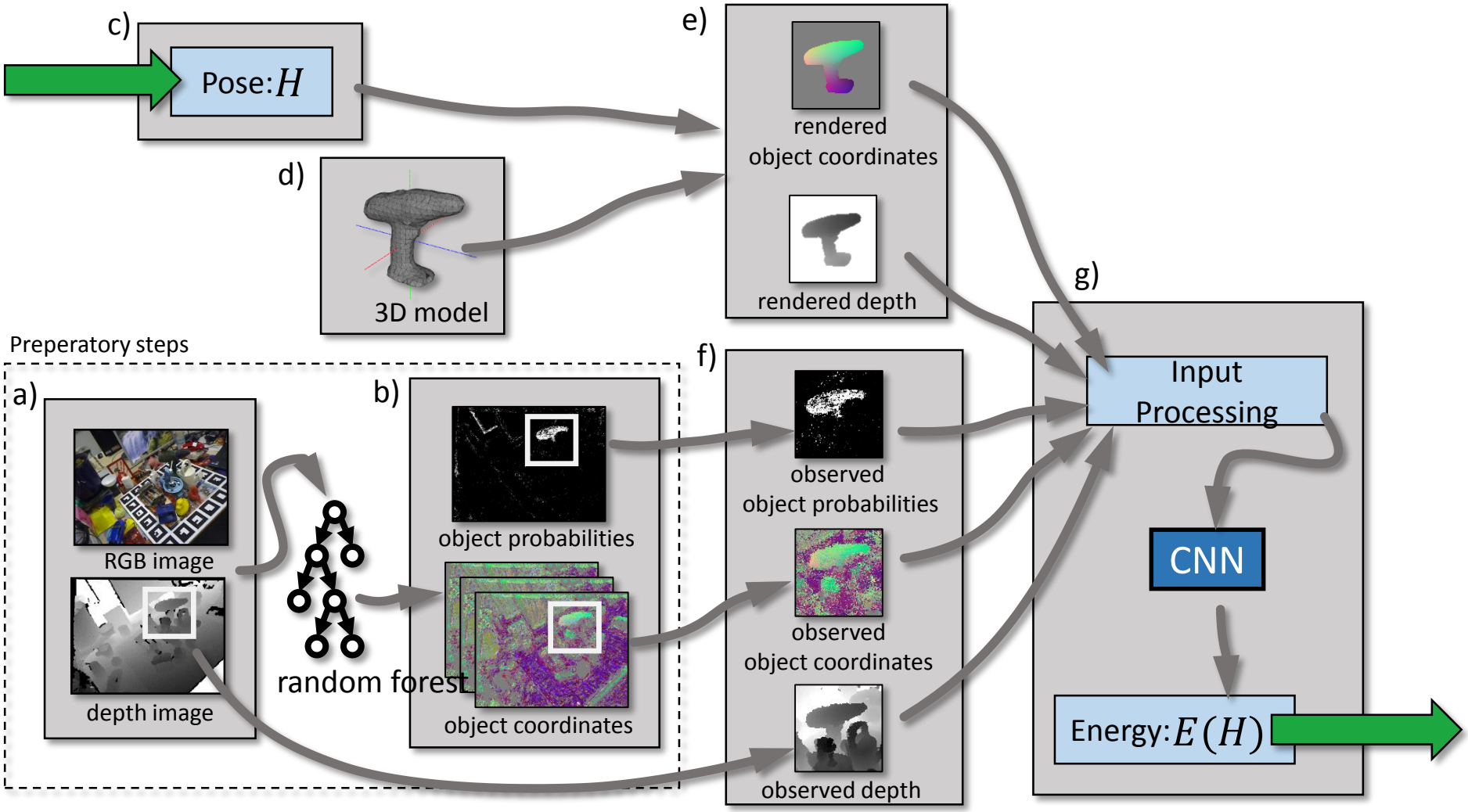
Learning an Energy Function

Comparison is mapping of **images** to **real numbers**:



- use **CNN** to implement the mapping
- try **not to learn specific properties of object**
- **learn** how **to compare** rendered and observed images

Energy Calculation



Training the Network

Maximum Likelihood Training:

Model the **posterior as Gibbs distribution**:

$$p(H|\boldsymbol{x};\boldsymbol{\theta}) = \frac{\exp\left(-E(H,\boldsymbol{x};\boldsymbol{\theta})\right)}{\int \exp\left(-E\left(\widehat{H},\boldsymbol{x};\boldsymbol{\theta}\right)\right) d\widehat{H}}$$

Use maximum likelihood with labeled ground truth data: $(H_i^*, \boldsymbol{x}_i)$

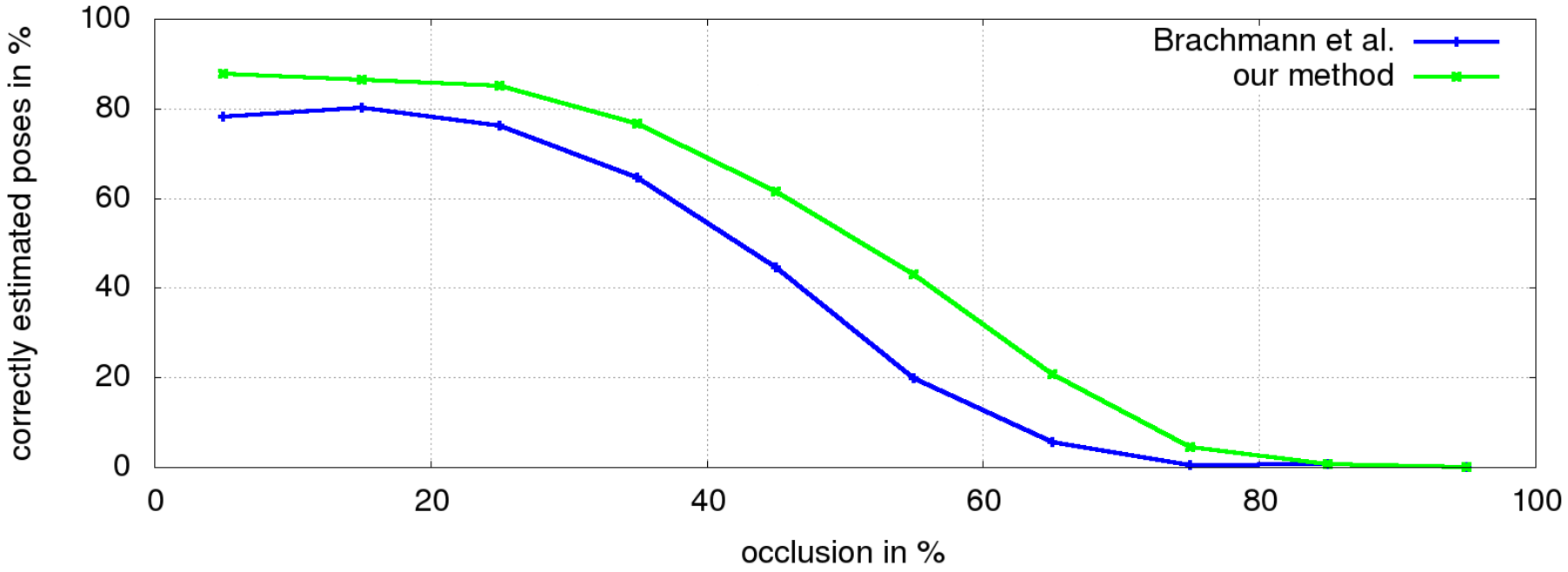
The partial derivatives are:

calculate via **error back propagation**

$$\frac{\partial}{\partial \boldsymbol{\theta}_j} \ln p(H_i^*|\boldsymbol{x}_i;\boldsymbol{\theta}) = -\frac{\partial}{\partial \boldsymbol{\theta}_j} E(H_i^*, \boldsymbol{x}_i;\boldsymbol{\theta}) + \mathbb{E}\left[\frac{\partial}{\partial \boldsymbol{\theta}_j} E(H, \boldsymbol{x}_i;\boldsymbol{\theta})|\boldsymbol{x}_i;\boldsymbol{\theta}\right]$$

approximate via **MCMC sampling**

Results on Occlusion Dataset [Hi12,Br14]



Test Objects From [Hi12,Br14]

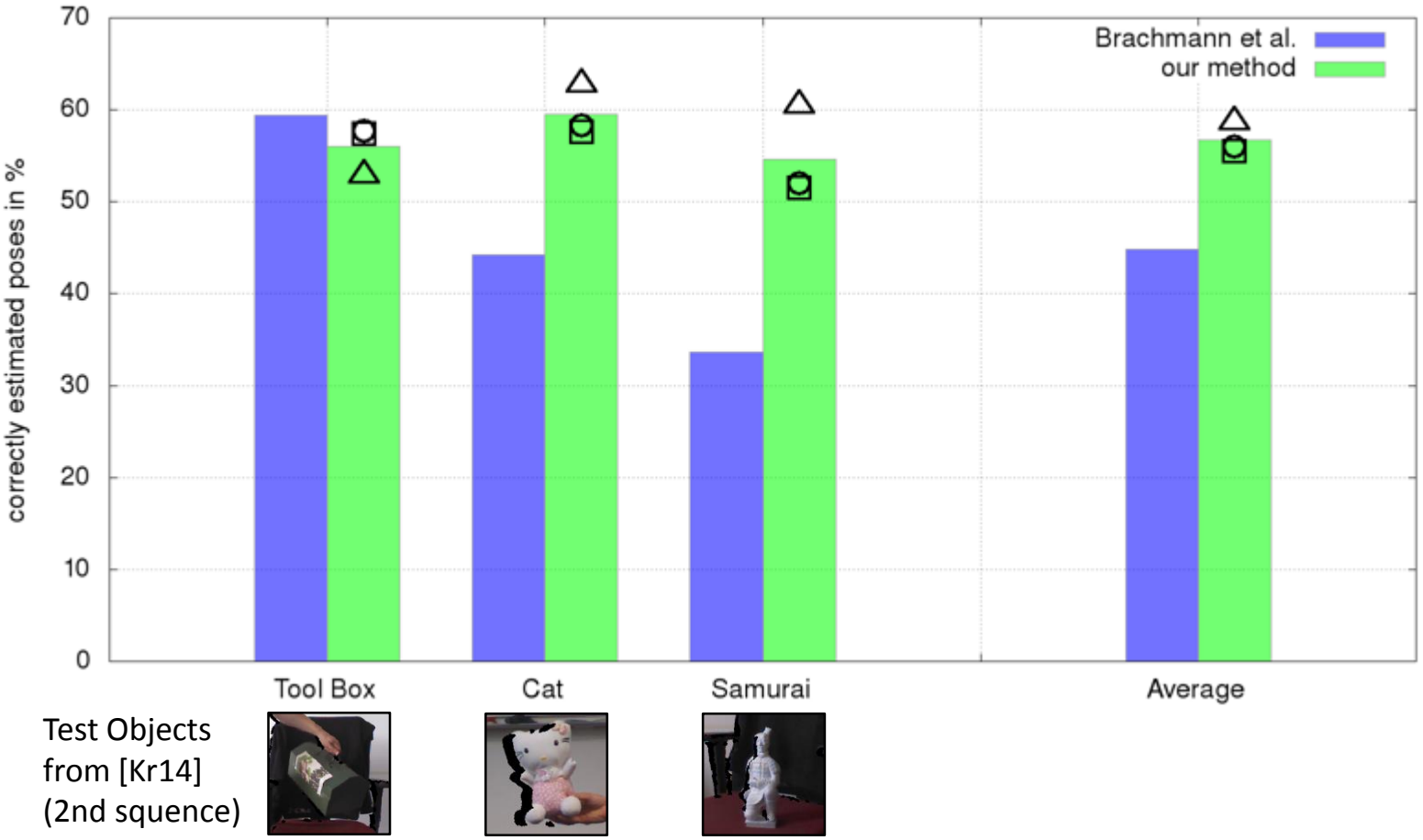
Training Object from [Kr14] (1st squence)

[Hi12]: Stefan Hinterstoisser, Vincent Lepetit, Slobodan Ilic, Stefan Holzer, Gary R. Bradski, Kurt Konolige, Nassir Navab: *Model Based Training, Detection and Pose Estimation of Texture-Less 3D Objects in Heavily Cluttered Scenes.* ACCV 2012

[Br14]: Eric Brachmann, Alexander Krull, Frank Michel, Stefan Gumhold, Jamie Shotton, Carsten Rother: *Learning 6D Object Pose Estimation using 3D Object Coordinates.* ECCV 2014

[Kr14]: Alexander Krull, Frank Michel, Eric Brachmann, Stefan Gumhold, Stephan Ihrke, Carsten Rother: *6-DOF Model Based Tracking via Object Coordinate Regression*. ACCV 2014

Results on Dataset from [Kr14]



Test Objects from [Kr14] (2nd squence)

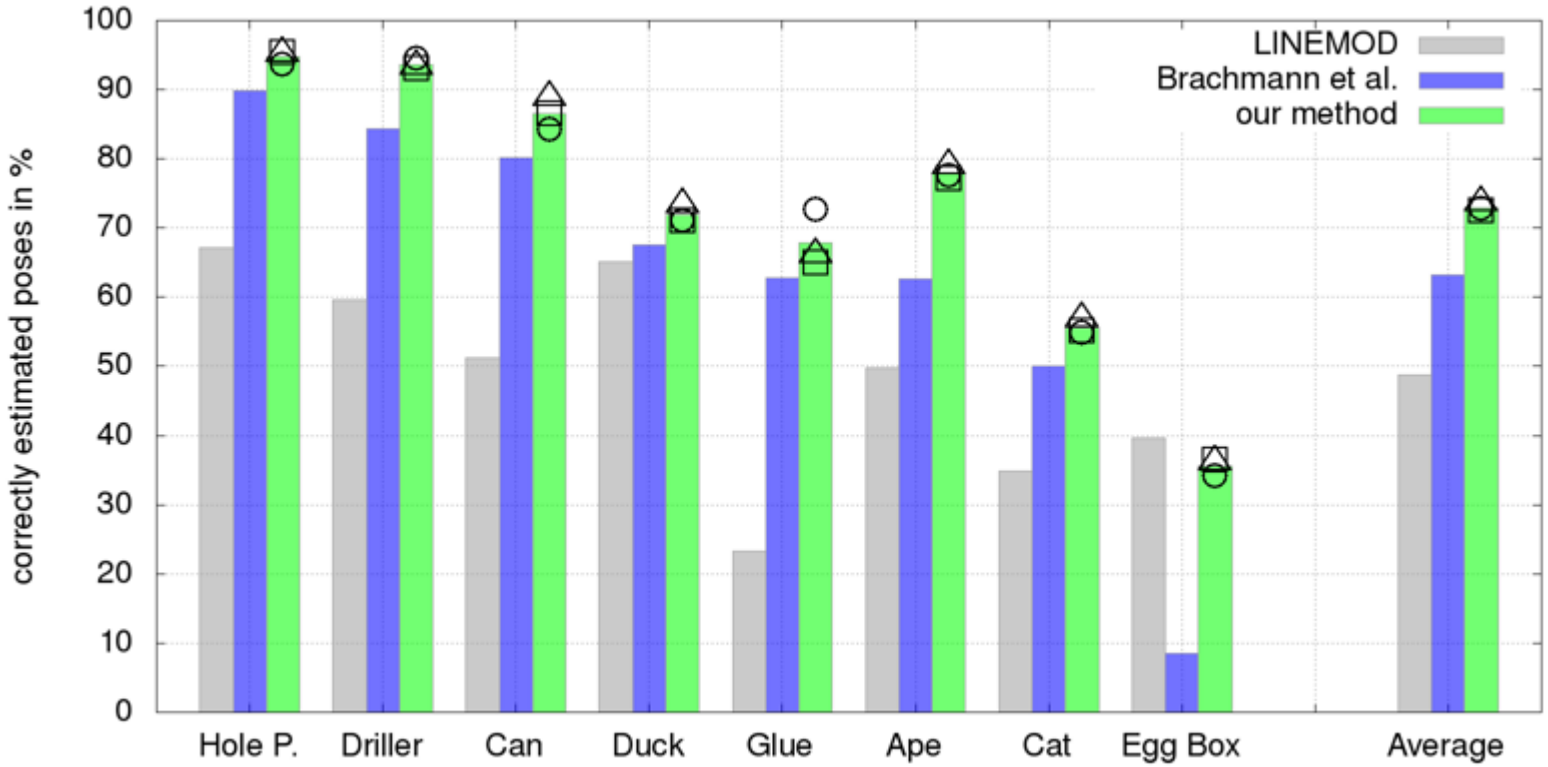
Training Objects from [Kr14] (1st squence)

[Hi12]: Stefan Hinterstoisser, Vincent Lepetit, Slobodan Ilic, Stefan Holzer, Gary R. Bradski, Kurt Konolige, Nassir Navab: *Model Based Training, Detection and Pose Estimation of Texture-Less 3D Objects in Heavily Cluttered Scenes.* ACCV 2012

[Br14]: Eric Brachmann, Alexander Krull, Frank Michel, Stefan Gumhold, Jamie Shotton, Carsten Rother: *Learning 6D Object Pose Estimation using 3D Object Coordinates.* ECCV 2014

[Kr14]: Alexander Krull, Frank Michel, Eric Brachmann, Stefan Gumhold, Stephan Ihrke, Carsten Rother: *6-DOF Model Based Tracking via Object Coordinate Regression*. ACCV 2014

Results on Occlusion Dataset [Hi12,Br14]



Test Objects From [Hi12,Br14]

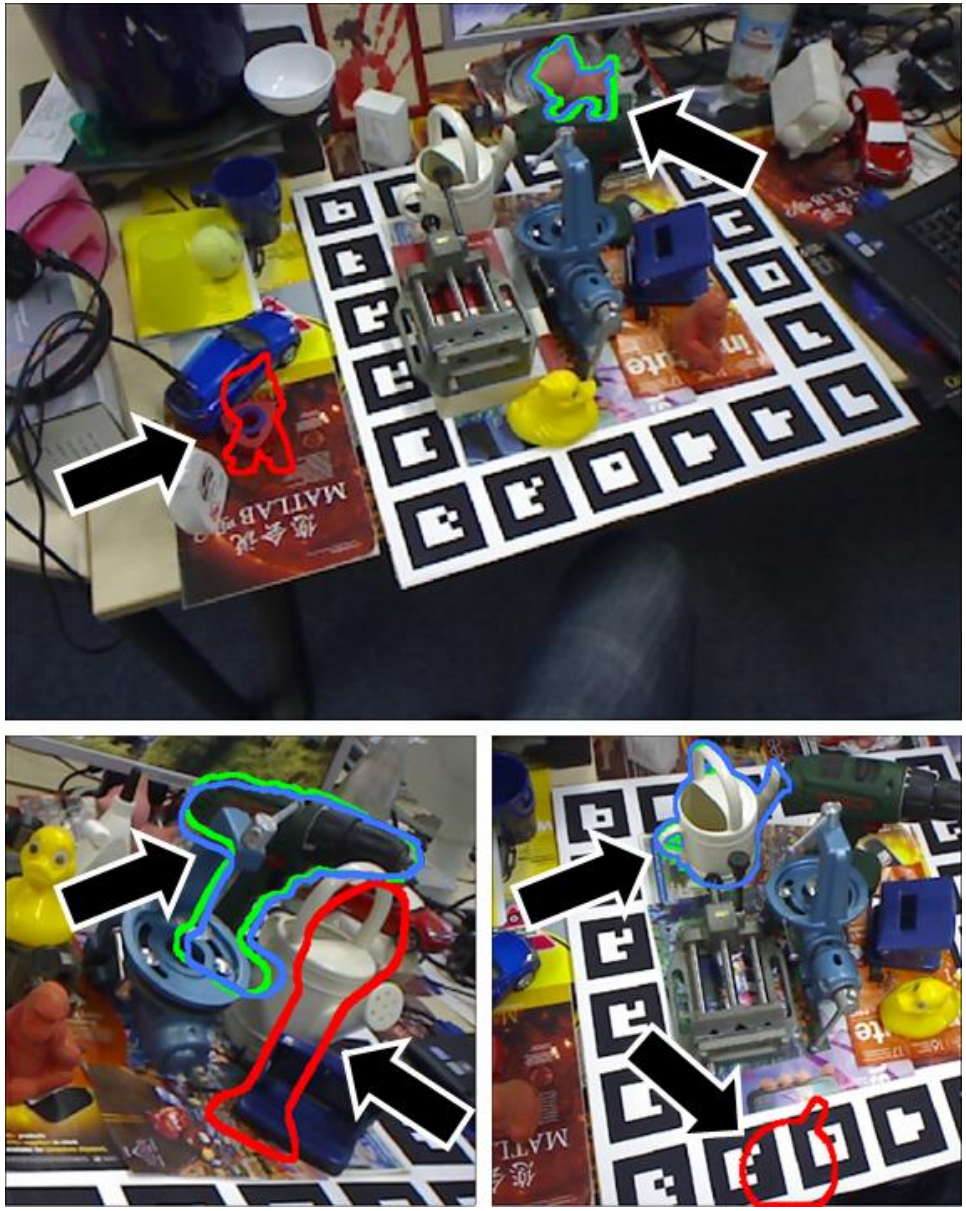
Training Objects from [Kr14] (1st squence)

[Hi12]: Stefan Hinterstoisser, Vincent Lepetit, Slobodan Ilic, Stefan Holzer, Gary R. Bradski, Kurt Konolige, Nassir Navab: *Model Based Training, Detection and Pose Estimation of Texture-Less 3D Objects in Heavily Cluttered Scenes.* ACCV 2012

[Br14]: Eric Brachmann, Alexander Krull, Frank Michel, Stefan Gumhold, Jamie Shotton, Carsten Rother: *Learning 6D Object Pose Estimation using 3D Object Coordinates.* ECCV 2014

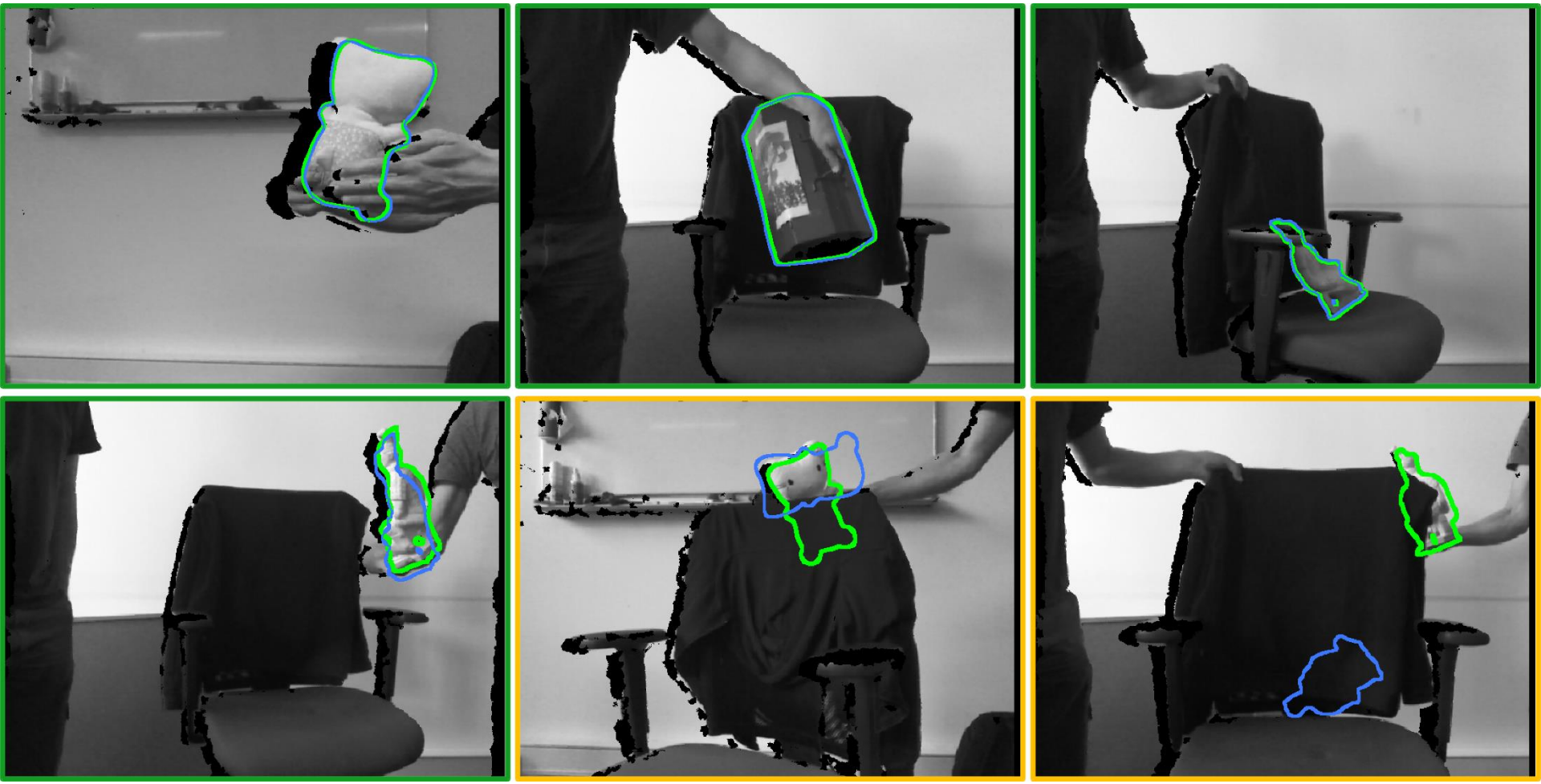
[Kr14]: Alexander Krull, Frank Michel, Eric Brachmann, Stefan Gumhold, Stephan Ihrke, Carsten Rother: *6-DOF Model Based Tracking via Object Coordinate Regression*. ACCV 2014

Qualitative Results

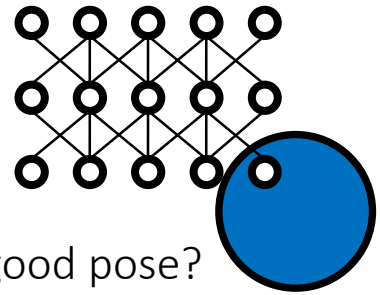


Qualitative Results

Qualitative Results

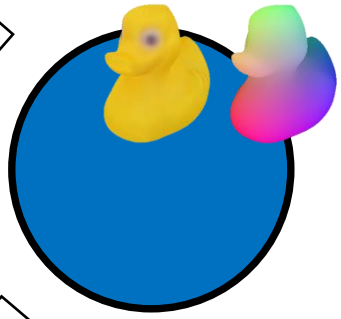


Overview



What is a good pose?
Can a CNN help?
(Krull, 2015)

How to deal with articulted objects? (Michel, 2015)

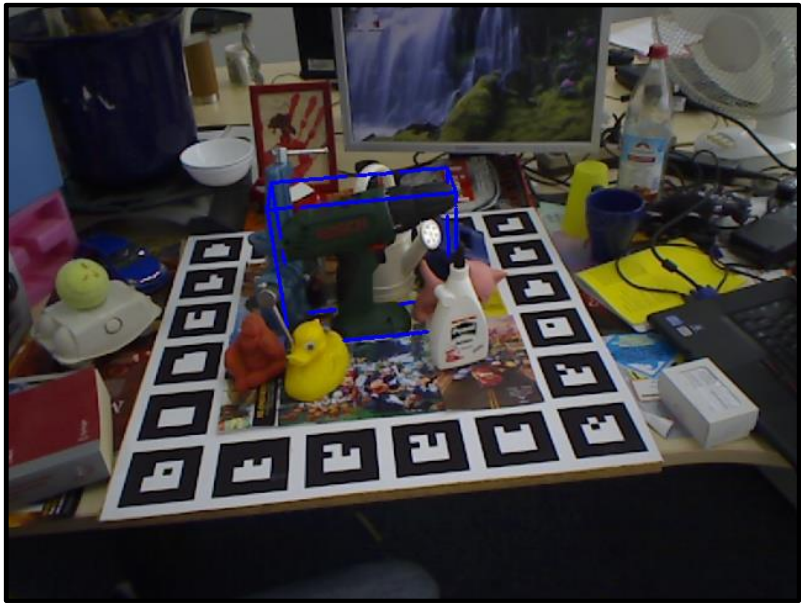


What about tracking? (Krull, 2014)

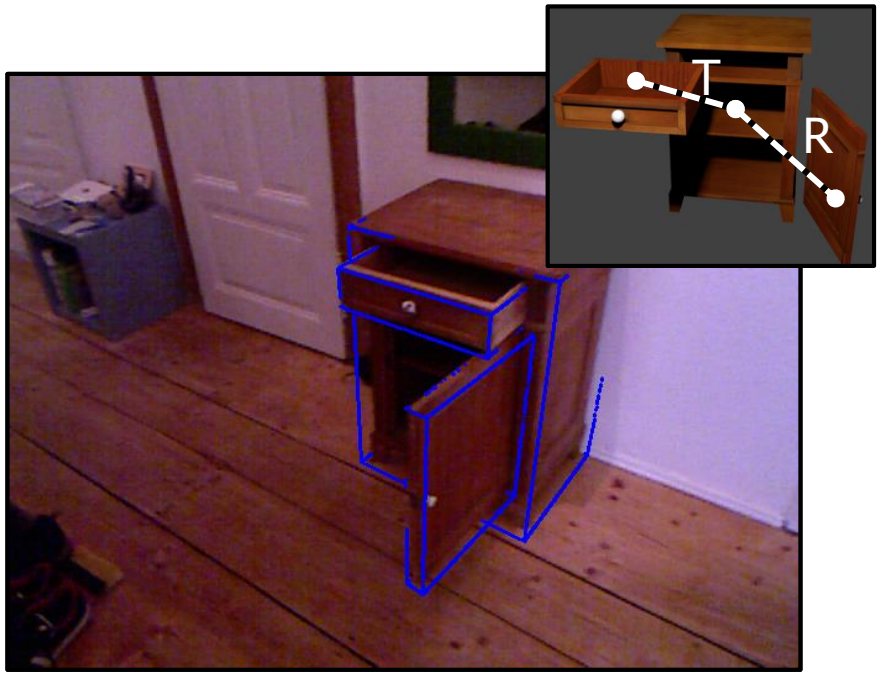
Object ooordinates for pose estimation
(Brachmann, 2014)

How to deal with articulated objects?

- 6D Rigid Object Pose Estimation [Br14]

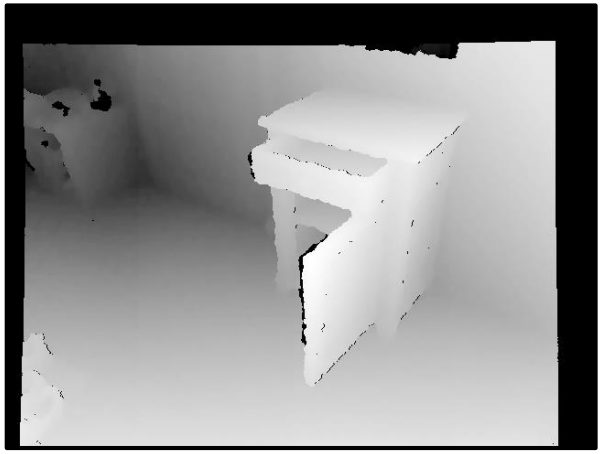


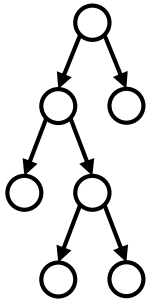
- Articulated Object Pose Estimation



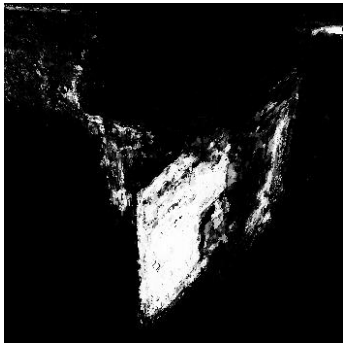
Articulated Pose Sampling

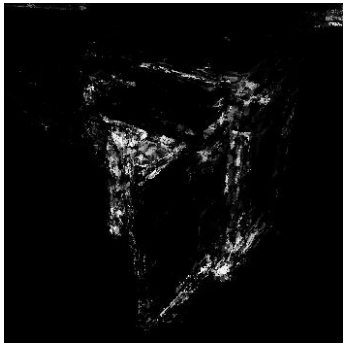
Depth input

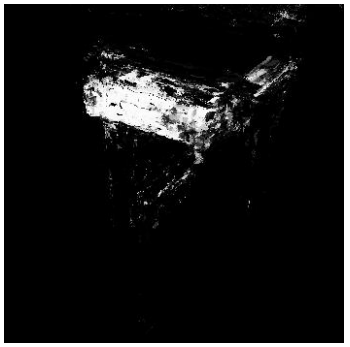


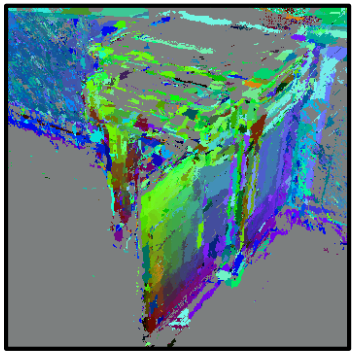


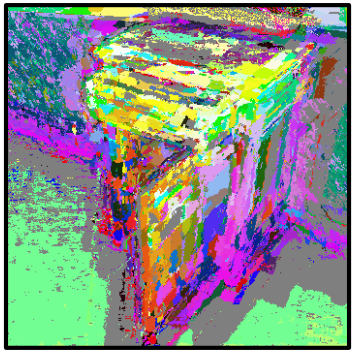
Random forest prediction

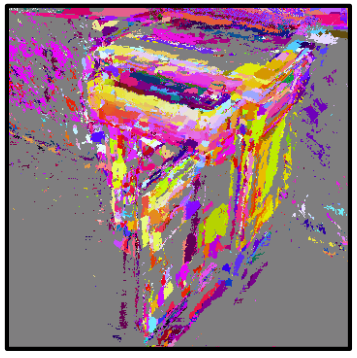










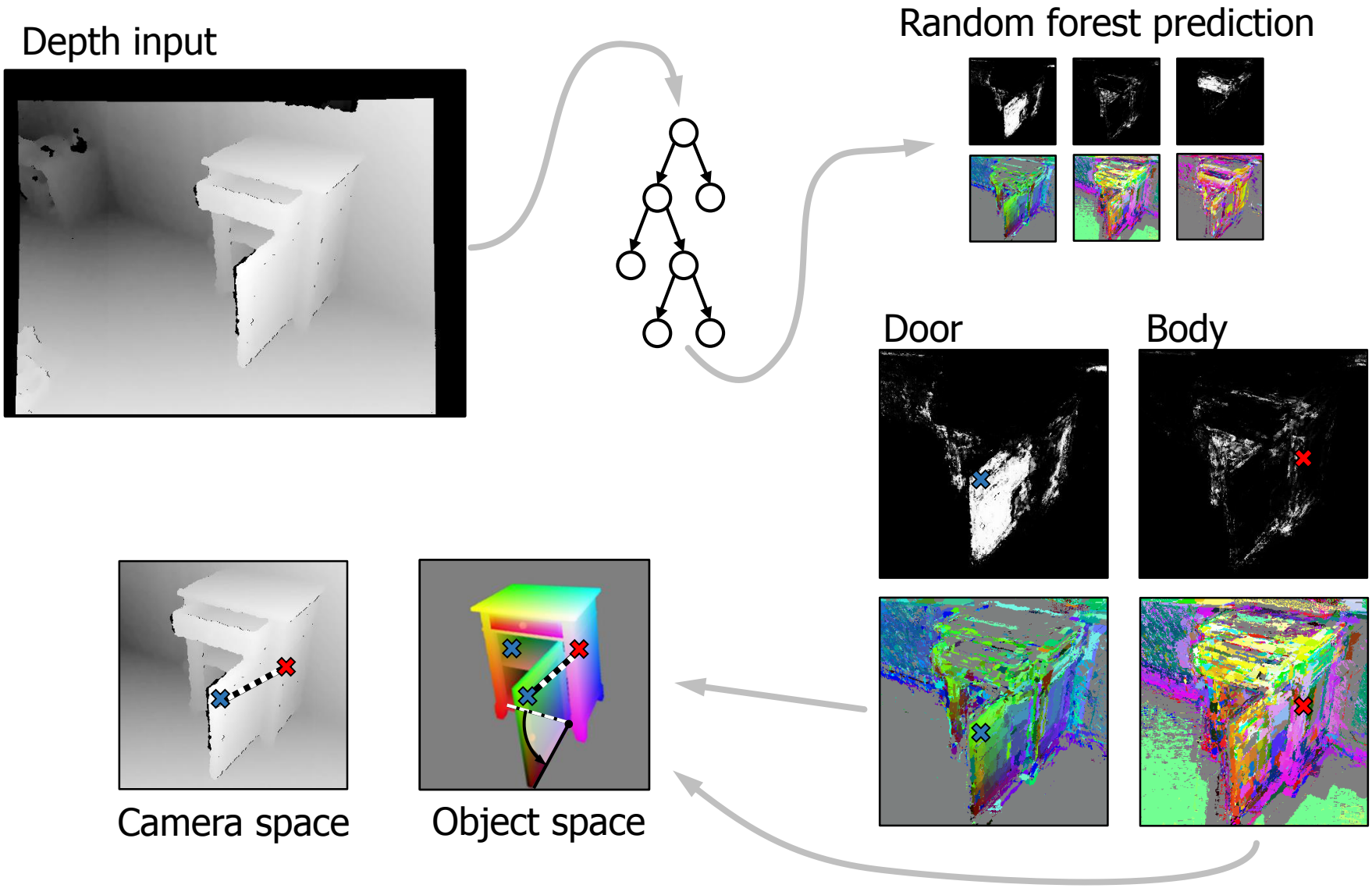


Door

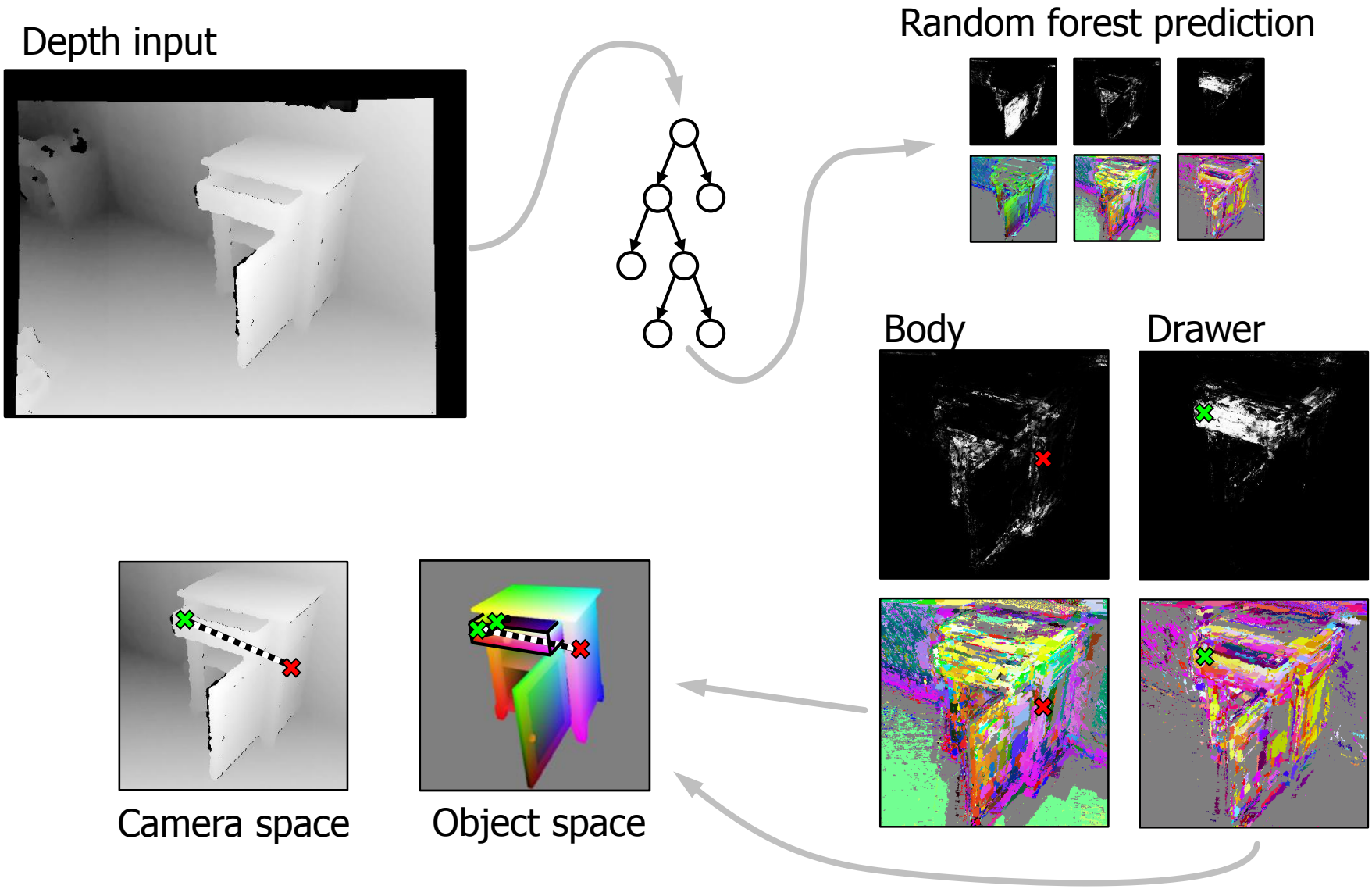
Body

Drawer

Articulated Pose Sampling

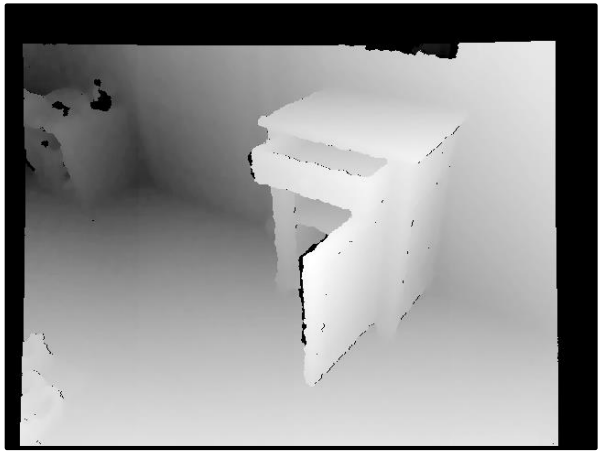


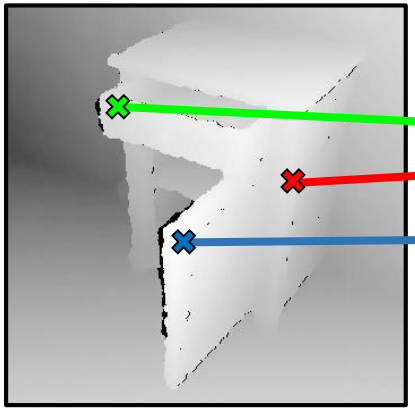
Articulated Pose Sampling



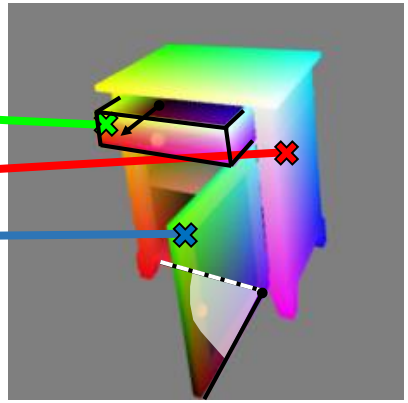
Articulated Pose Sampling

Depth input





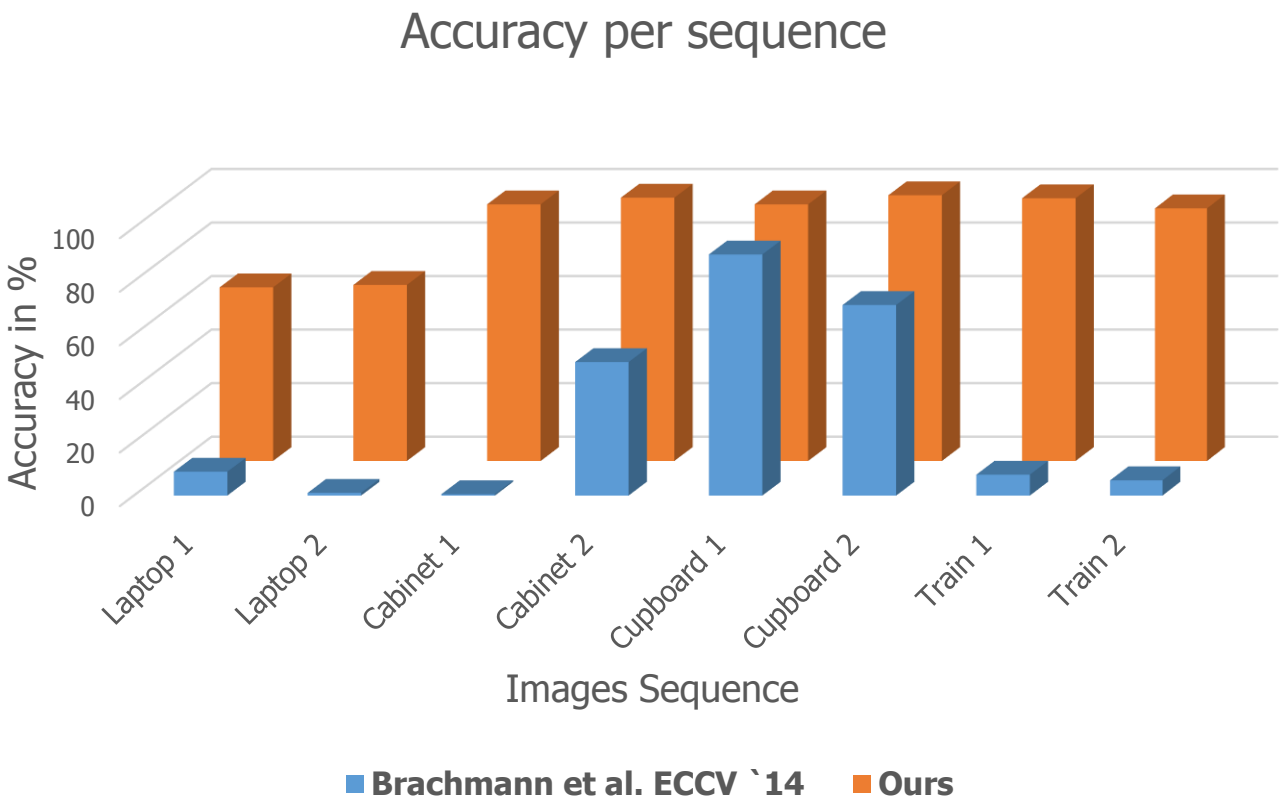
Camera space

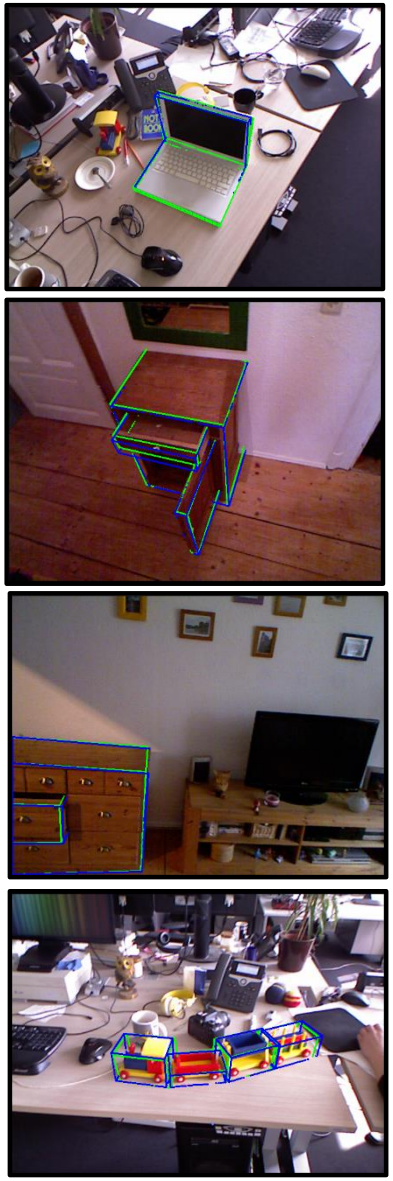


Object space

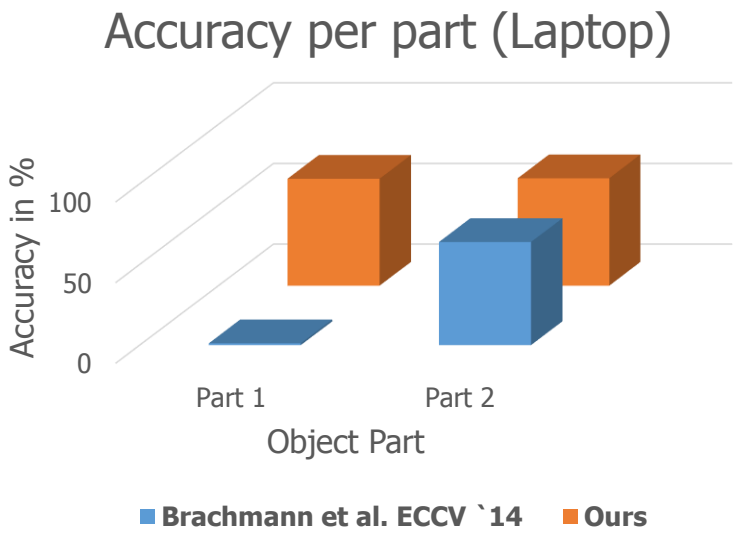
Pose hypothesis

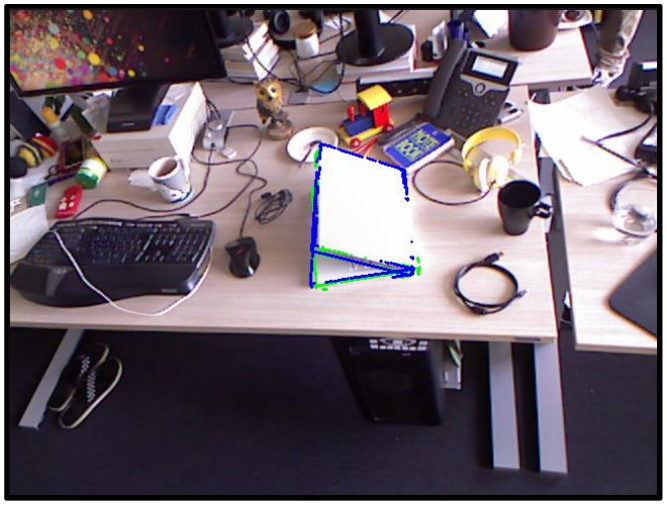
Evaluation

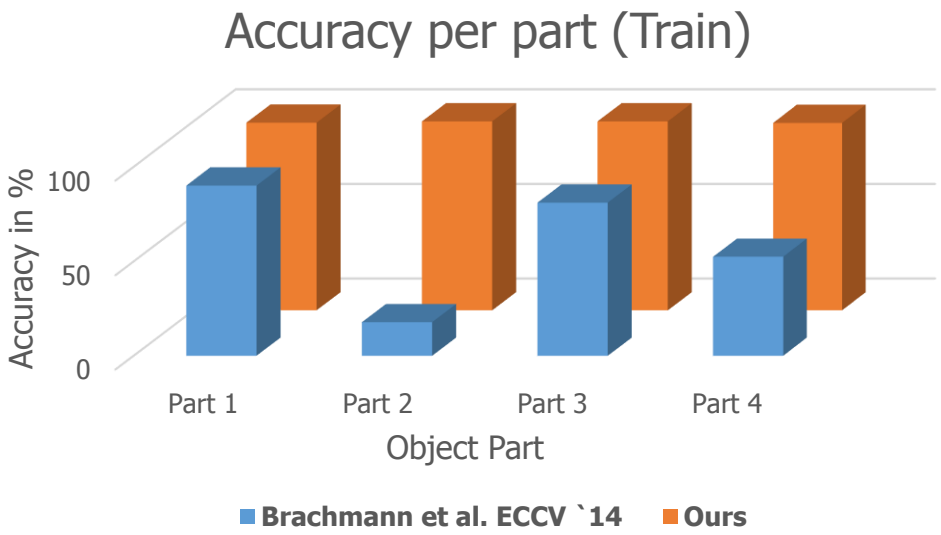

Accuracy per sequence
Accuracy in %
100
80
60
40
20
0
Laptop 1
Laptop 2
Cabinet 1
Cabinet 2
Cupboard 1
Cupboard 2
Train 1
Train 2
Images Sequence
Brachmann et al. ECCV `14
Ours

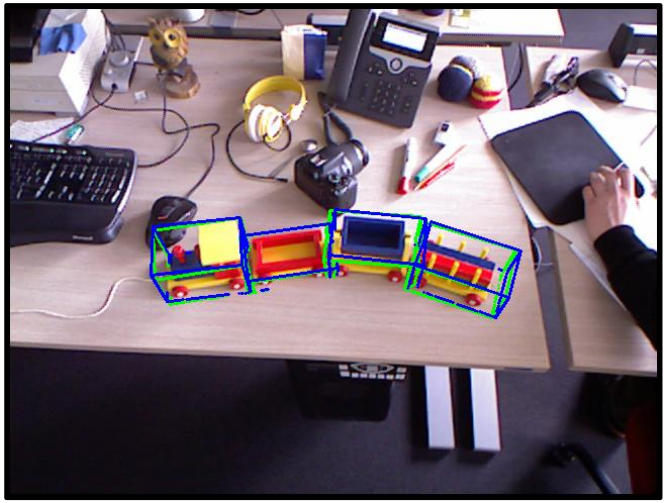


Evaluation

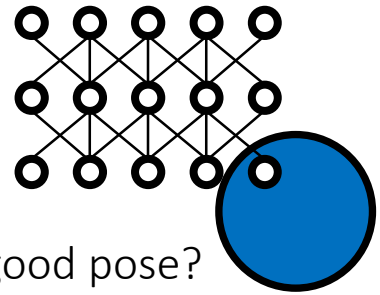

Accuracy per part (Laptop)
Accuracy in %
100
50
0
Part 1
Part 2
Object Part
Brachmann et al. ECCV `14
Ours




Accuracy per part (Train)
Accuracy in %
100
50
0
Part 1
Part 2
Part 3
Part 4
Object Part
Brachmann et al. ECCV `14
Ours

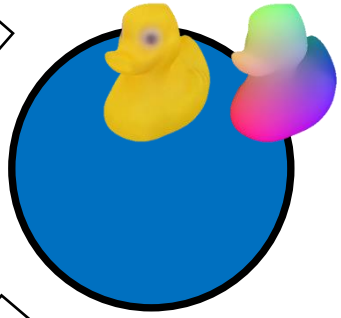


Overview



What is a good pose?
Can a CNN help?
(Krull, 2015)

How to deal with articulted objects? (Michel, 2015)

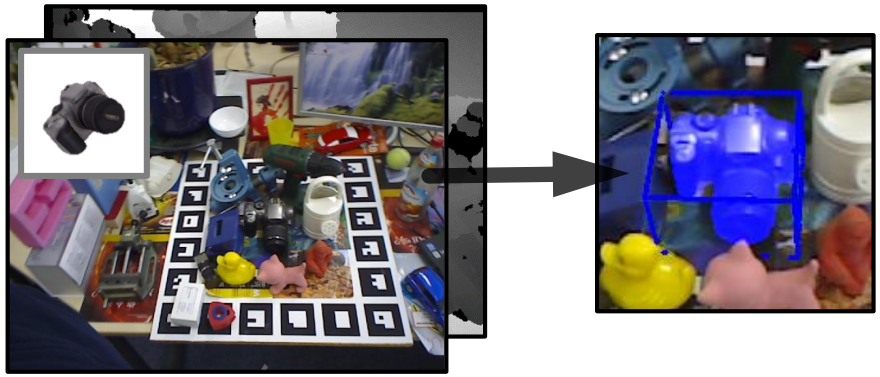


What about tracking? (Krull, 2014)

Object ooordinates for pose estimation
(Brachmann, 2014)

Tracking Task

One Shot Pose Estimation [Br14]



- Estimate 6D Pose from **single** RGB-D image
- Use Object Coordinate Regression

Pose Tracking

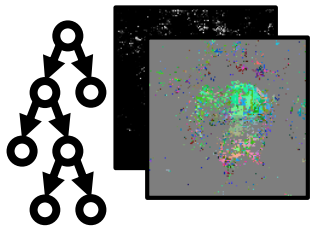
- **Stream** of RDB-D images
- Use information from previous frames
  - Realtime
  - Increase robustness, accuracy

[Br14] Brachmann, E., Krull, A., Michel, F., Shotton, J., Gumhold, S., Rother, C.: Learning 6d object pose estimation using 3d object coordinates, ECCV (2014)

How to Adapt it for Pose Tracking?

Ingredients:

Object Coordinate Regression



Particle Filter

Observation Model:

Energy

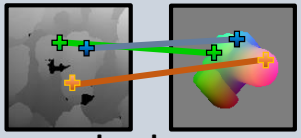
Motion Model:

- Assume continous motion

Proposal Distribution:

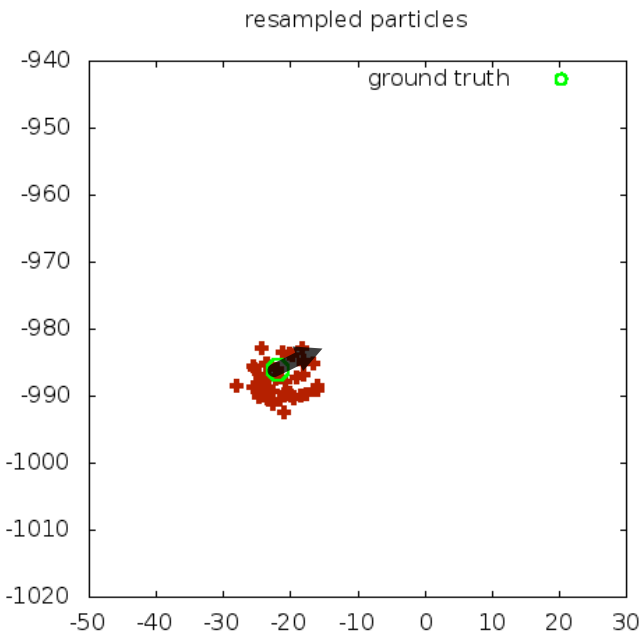
- Find rough estimate using

Efficient Search

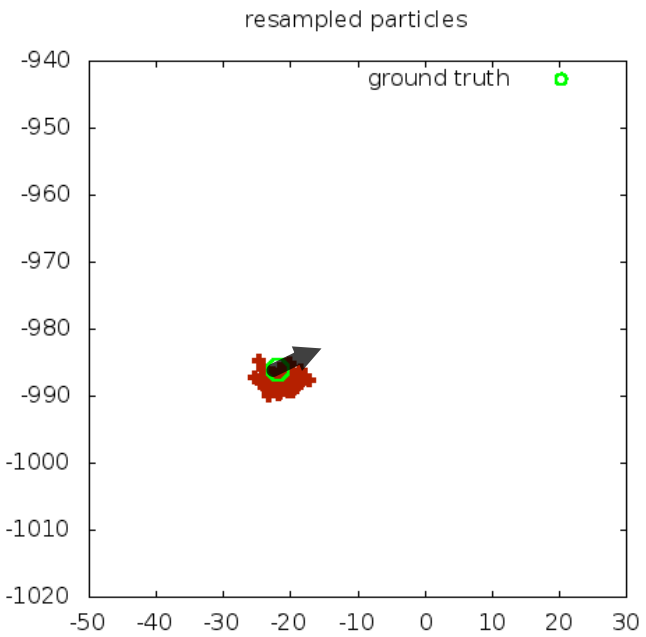


- Concentrate samples around rough estimate

Tracking Algorithm

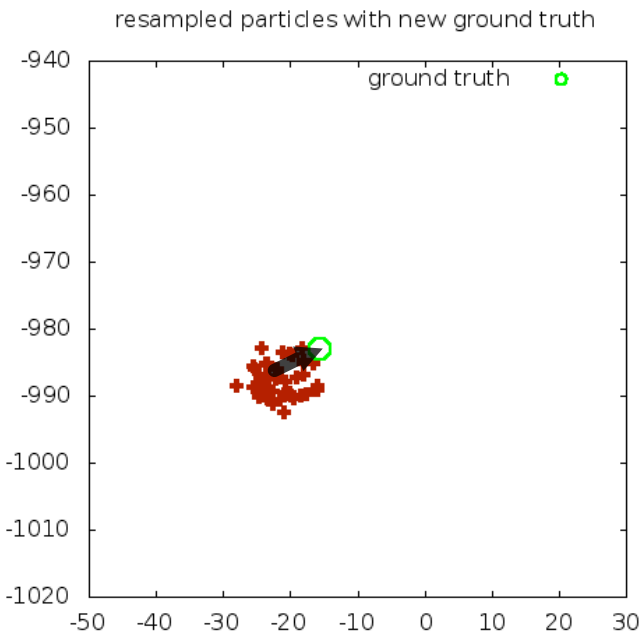


Sampling from Motion Model

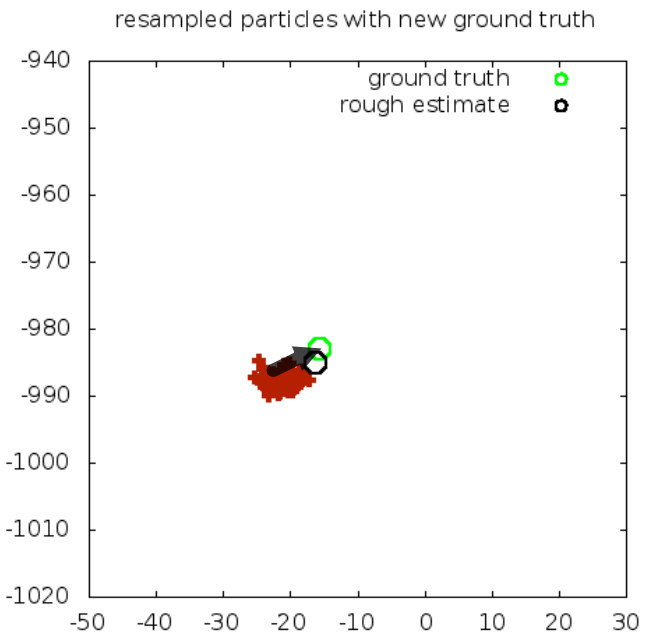


Sampling from Proposal Distribution

Tracking Algorithm

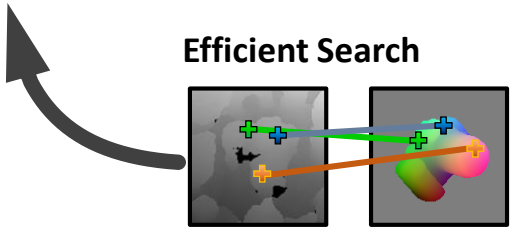


Sampling from Motion Model

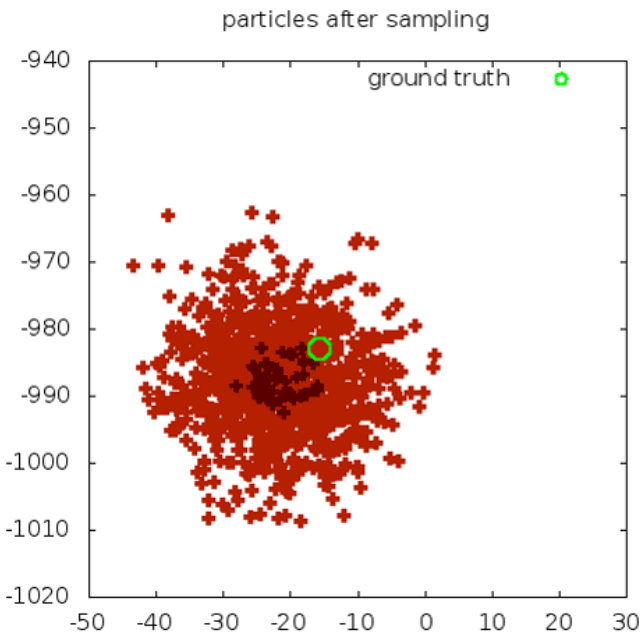


Sampling from Proposal Distribution

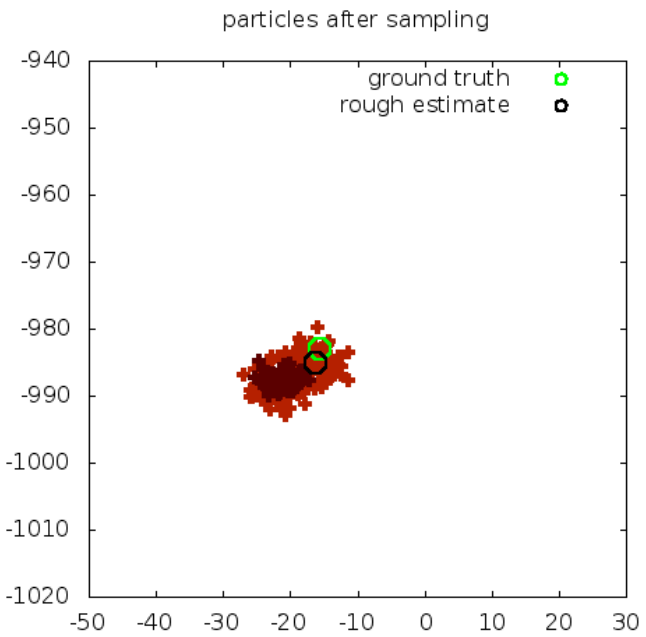
- Find a rough estimate for current pose



Tracking Algorithm



Sampling from Motion Model

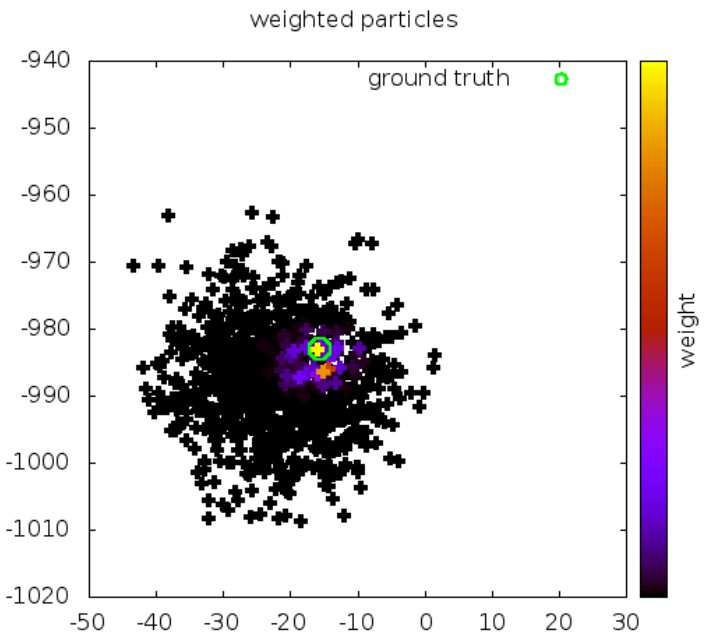


Sampling from Proposal Distribution

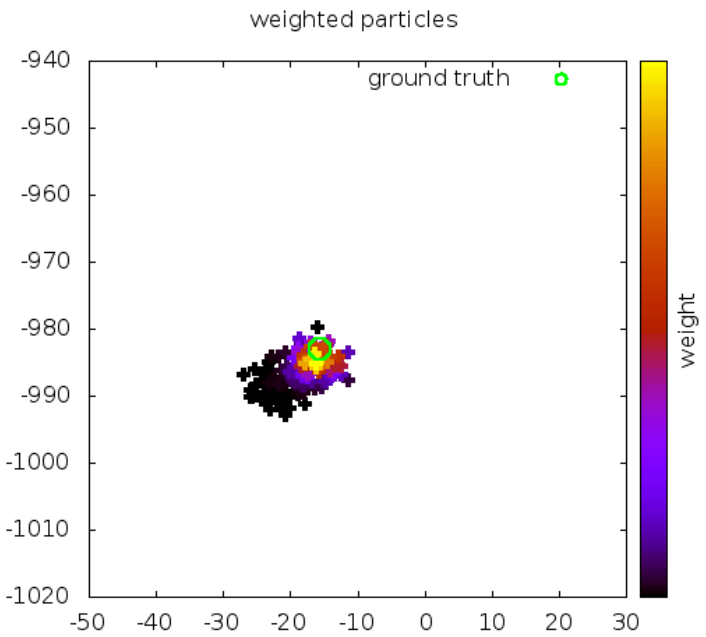
- Samples are spread out very far

- Samples are concentrated around rough estimate

Tracking Algorithm



Sampling from Motion Model

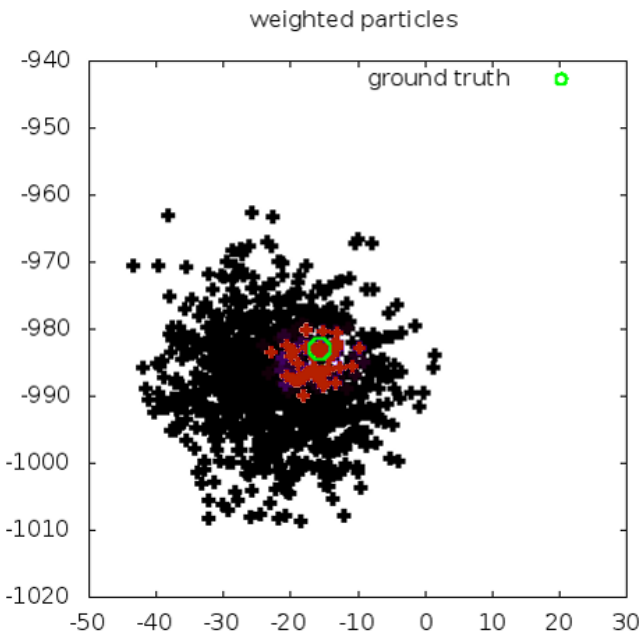


Sampling from Proposal Distribution

- Most samples have a very low weight

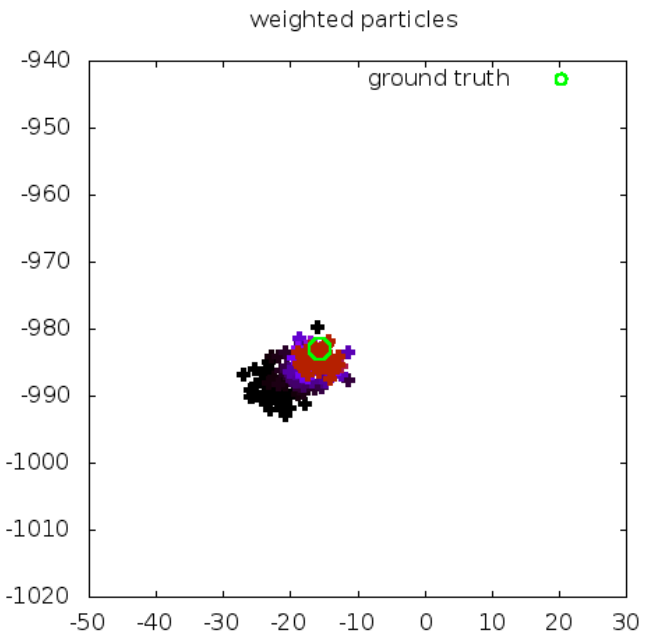
- Few samples have very low weight

Tracking Algorithm



Sampling from Motion Model

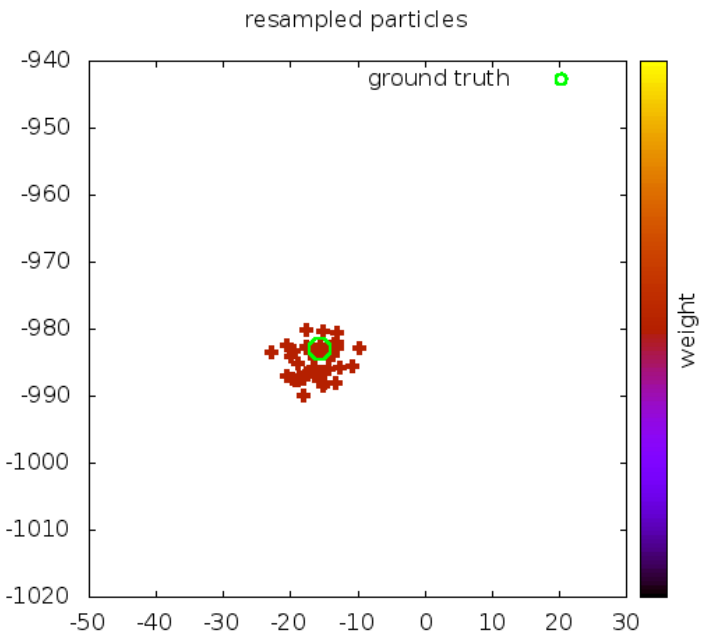
- Most samples have a very low weight



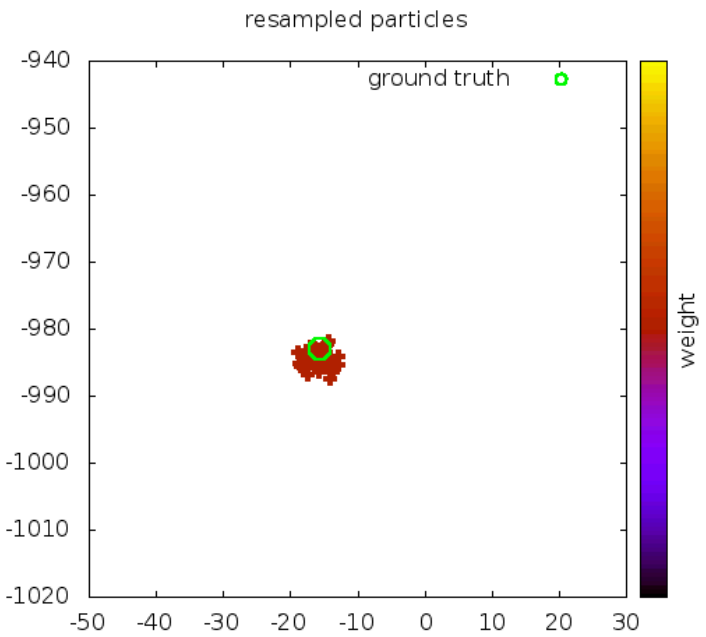
Sampling from Proposal Distribution

- Few samples have very low weight

Tracking Algorithm



Sampling from Motion Model

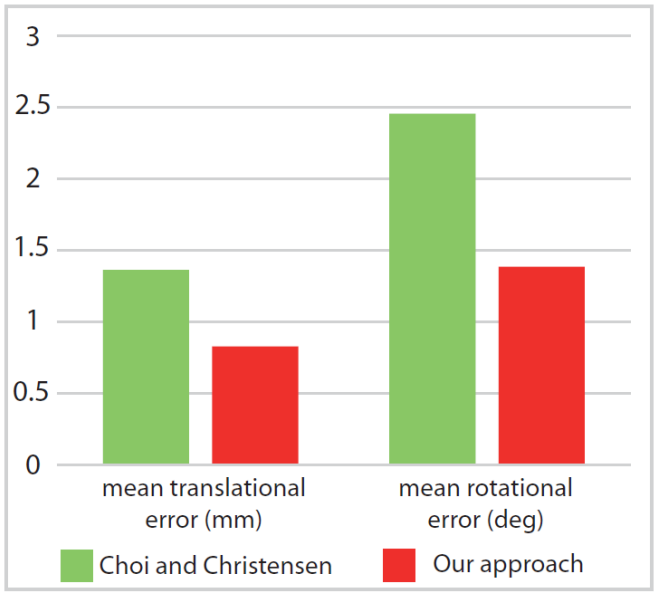


Sampling from Proposal Distribution

- More efficient
- Number of Particles can be reduced
- Frame rate can be increased

Evaluation: Dataset of [Ch13]

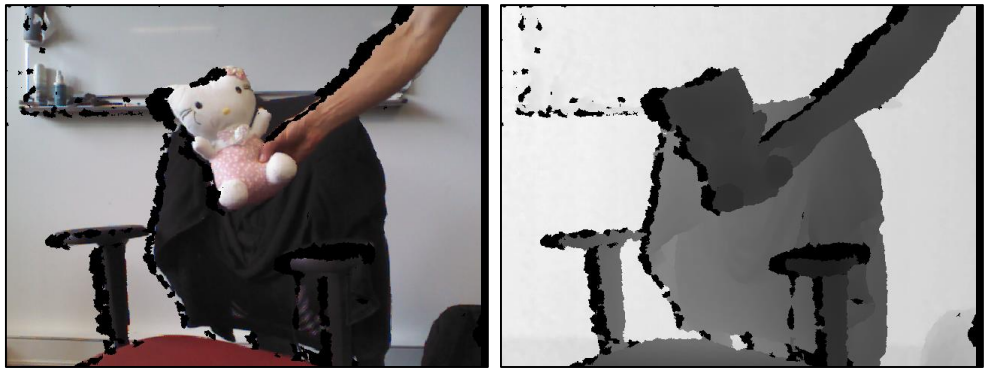
- A total of 4 synthetic sequences with 4 objects
- Objects placed in static environment
- Camera moving around object
- Partial occlusion
- Very exact ground truth



[Ch13] Changhyun Choi, Henrik I. Christensen, RGB-D Object Tracking: *A Particle Filter Approach on GPU*, IROS, 2013

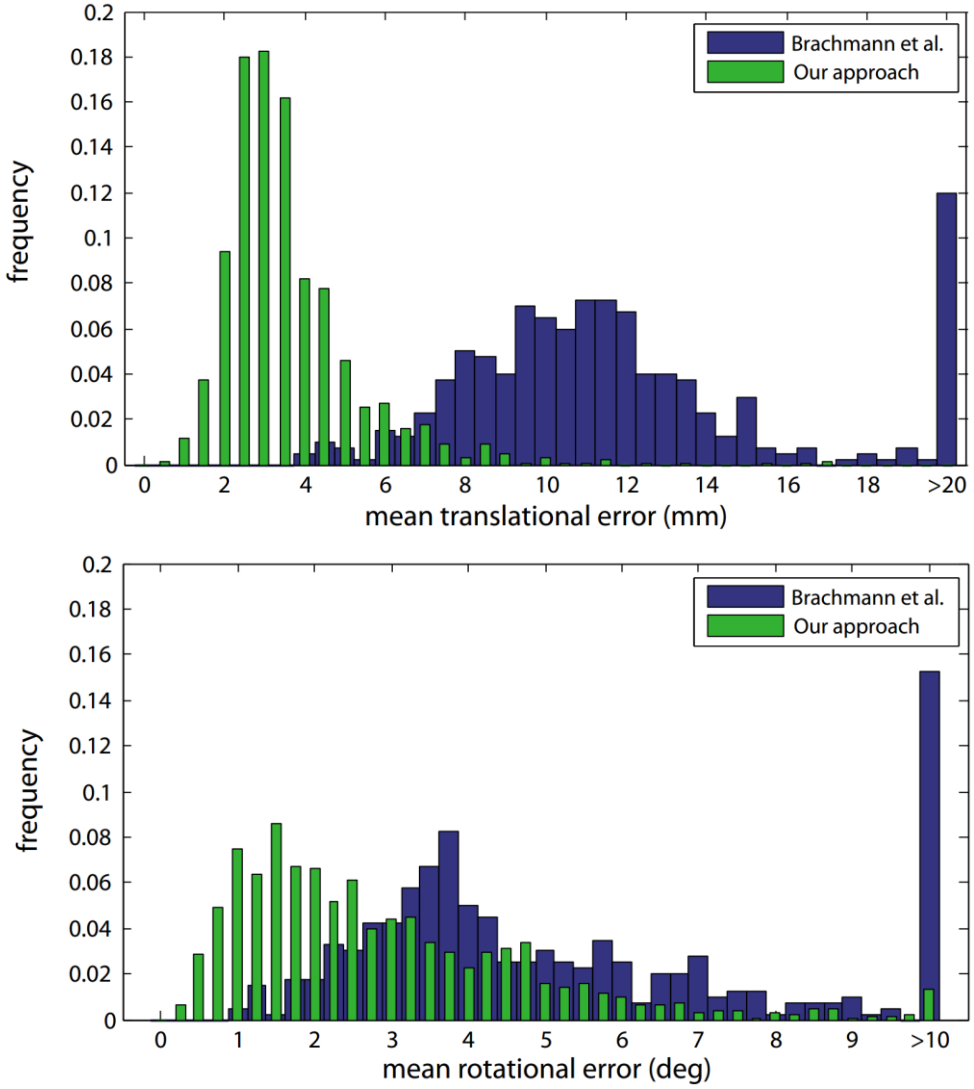
Evaluation: Our Dataset

- A total of 6 captured sequences of with 3 objects
- Manually annotated ground truth
- Moving objects in front of dynamic background
- Fast erratic movement
- Strong occlusions



Evaluation: Our Dataset

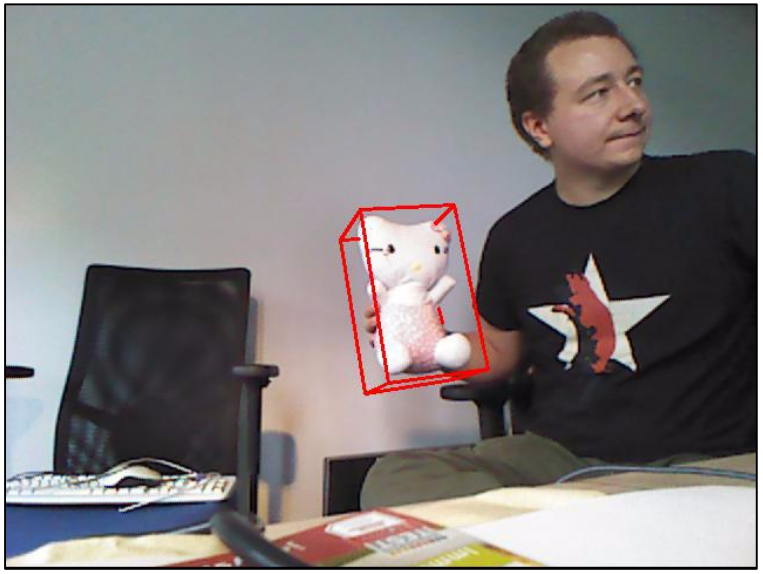
- Compared to [Br14] applied to each frame separately
- Lower average error
- Almost no outliers



[Br14] Brachmann, E., Krull, A., Michel, F., Shotton, J., Gumhold, S., Rother, C.: Learning 6d object pose estimation using 3d object coordinates, ECCV (2014)

Conclusion

- We have adapted the system from [Br14] for real time pose tracking
- We have designed a proposal distribution making efficient use of object coordinates
- Method is robust against:
  - Quick motion
  - Strong occlusion
  - Shadows and changing light conditions
  - Deformation



Where to go from here?

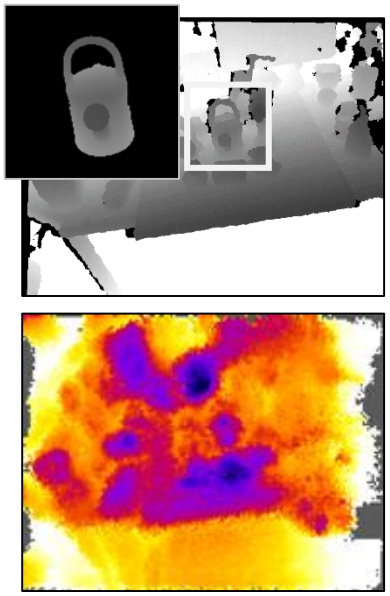
- Experiments on RGB
  - It looks promising.

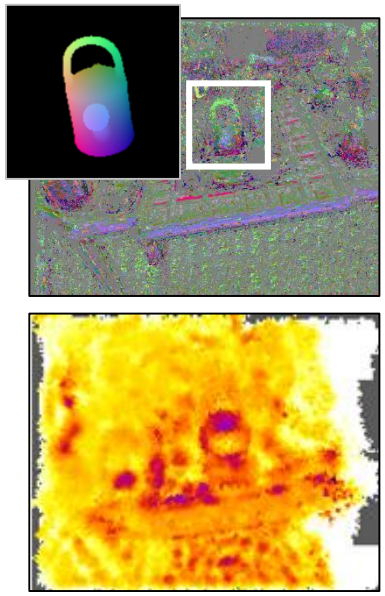
- CNN energy:
  - Optimize the search for the pose
  - Can we deal with reflectance or transparency?
  - How about discriminative training?

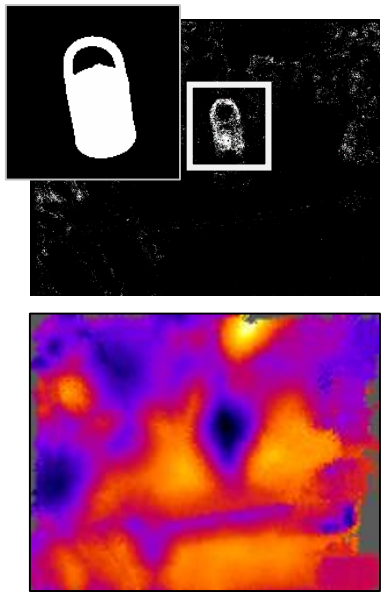
- Object classes:
  - Can object coordinates help with the task?

Extra Slides

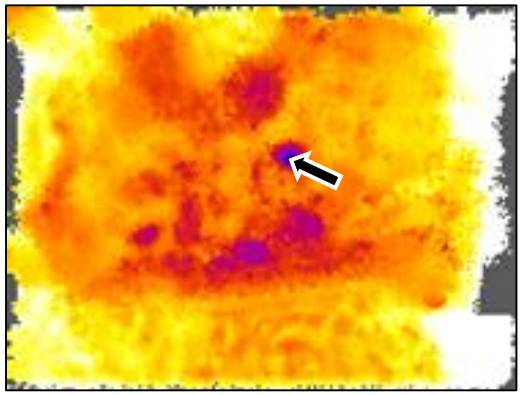
Hypothesis Evaluation







$$E(H) = \lambda^{\text{depth}} E^{\text{depth}}(H) + \lambda^{coord} E^{coord}(H) + \lambda^{obj} E^{obj}(H)$$



How to Train the CNN?

We do not know the ground truth energy

-> standard CNN learning not possible:

- view it as a **classification problem**:
  - good pose vs. bad pose
  - quadratic loss function
  - **traditional training**

- probabilistically:
  - model the **posterior as Gibbs distribution**:

$$p(y|x;\boldsymbol{\theta}) = \frac{\exp(-E(x,y;\boldsymbol{\theta}))}{\int \exp(-E(x,\hat{y};\boldsymbol{\theta}))d\hat{y}} \qquad \boldsymbol{\theta} = (\theta_1, \dots, \theta_m)$$

  - **maximum likelihood learning**

[Peng09]: J. Peng,L. Bo,J. Xu, *Conditional Neural Fields*. NIPS 2009
[Do10] Trinh Minh Tri Do, Thierry Arti: *Neural conditional random fields*. AISTATS 201

Maximum Likelihood Learning

- model the posterior as Gibbs distribution:

$$p(y|x;\boldsymbol{\theta}) = \frac{\exp\left(-E(x,y;\boldsymbol{\theta})\right)}{\int \exp\left(-E(x,\hat{y};\boldsymbol{\theta})\right) d\hat{y}} \qquad \boldsymbol{\theta} = (\theta_1, \ldots, \theta_m)$$

- maximize the log-likelihood:

$$\ell(\mathcal{D},\boldsymbol{\theta}) = \ln \prod_{i=1}^{n} p(y_i|x_i;\boldsymbol{\theta}) = \sum_{i=1}^{n} \ln p(y_i|x_i) \qquad \mathcal{D} = \left((x_1,y_1), \ldots, (x_n,y_n)\right)$$

- stochastic gradient descent, gradient for single training sample:

$$\frac{\partial \ln p(y_i|x_i;\boldsymbol{\theta})}{\partial \theta_j} = \frac{\partial E(y_\mathrm{i}, x_\mathrm{i};\boldsymbol{\theta})}{\partial \theta_j} - \mathrm{E}\left[\frac{\partial E(y, x_i;\boldsymbol{\theta})}{\partial \theta_j} | x_i;\boldsymbol{\theta}\right]$$

Use back propagation Approximate via Sampling

Training with Stochastic Gradient Descent

- repeat:
  1. start with random initial network $\boldsymbol{\theta_0}$
  2. pick a random training example (x_i, y_i)
  3. calculate $\frac{\partial E(y,x;\boldsymbol{\theta_t})}{\partial \theta_j}$ at pose y_i using back propagation
  4. do MCMC sampling for image x_i:
     - run pose estimation for intitialization
     - disregard first 30 samples
  5. update $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \alpha \nabla \ln p(y_i|x_i;\boldsymbol{\theta})$

- to avoid overfitting:
  - periodically test on validation set
  - use best performing weights