Object 6D Pose Estimation from single RGB image

QIU Xuchong

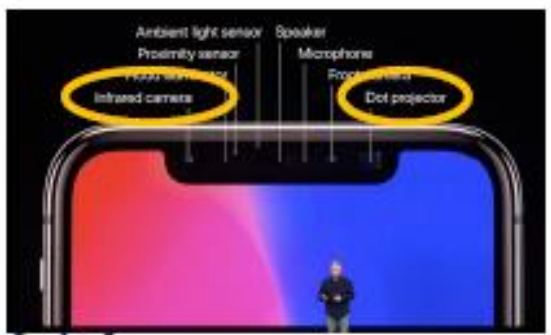
2019/02/17

Outline

- Objective
- Architecture
- Dataset
- Evaluation
- Challenges

Today

INPUT DATA	Mono		Consumer depth		Lidar / sparse
APPLICATIONS	AR for Smartphones	HMDs	Personal Robotics	Industrial Robotics	Autonomous Driving



[Federico Tombari, 2018]

Objective

- **Input:** RGB image, camera intrinsic(optional), 3D model(optional)
- **Output:** 6D pose (3D translation + 3D rotation)
- **Variants of task:**

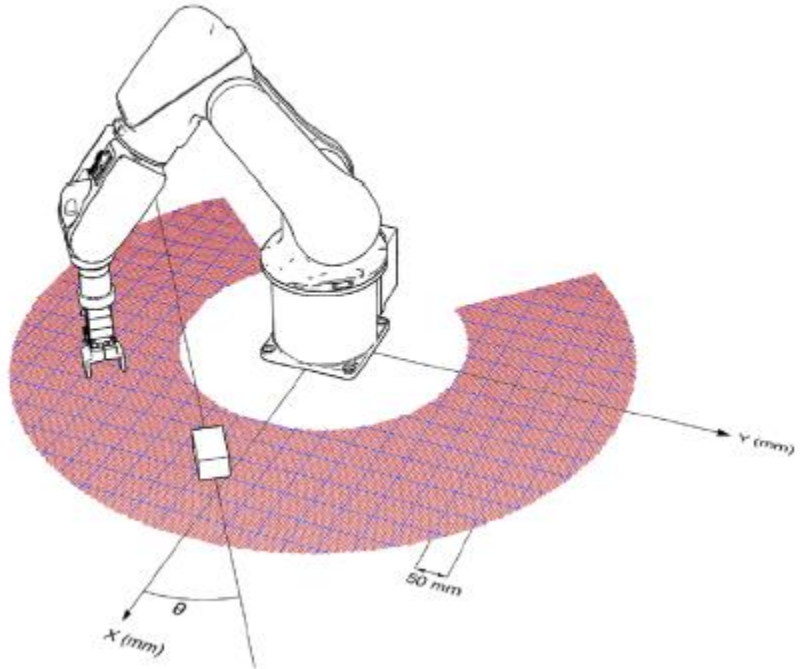
  Object pose with reference to camera frame

  Object pose with reference to robot frame

  Camera pose with reference to scene frame

  ……

[Richard Szeliski, 2010]



(a)

(b)

(c)

[Vianney et al, 2017]

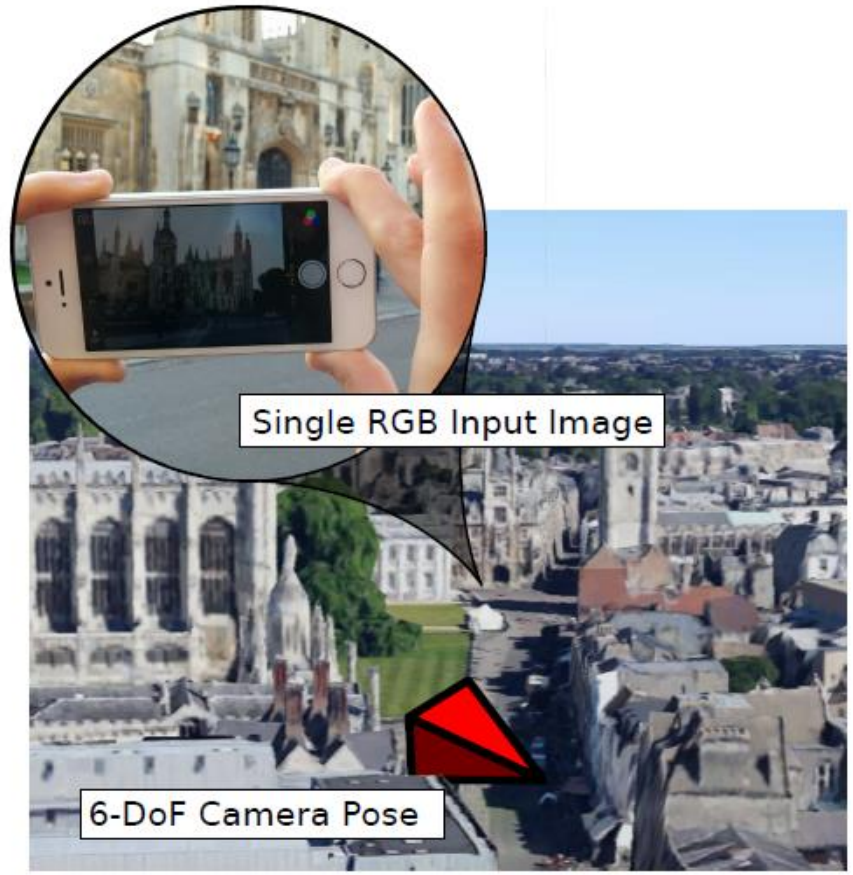


Figure 1: **PoseNet** [22] is trained end-to-end to estimate the camera's six degree of freedom pose from a single monocular image. In this paper we show how to apply a principled loss function based on the scene's geometry to learn camera pose without any hyper-parameters.

[Alex Kendall et al, 2017]

3D Euclidean transformation

- Also known as 3D rigid body motion, a geometric transformation of a Euclidean space that preserves the Euclidean distance between every pair of points

Translation. 3D translations can be written as $x' = x + t$ or

$$x' = \begin{bmatrix} I & t \end{bmatrix} \bar{x} \qquad where\ \bar{x} = [x, y, z, 1] \tag{2.23}$$

Rotation + translation. Also known as 3D *rigid body motion* or the 3D *Euclidean transformation*, it can be written as $x' = Rx + t$ or

$$x' = \begin{bmatrix} R & t \end{bmatrix} \bar{x} \tag{2.24}$$

where R is a 3×3 orthonormal rotation matrix with $RR^T = I$ and $|R| = 1$.

[Richard Szeliski, 2010]

Parameterization of 3D rotation matrix

- **Euler angles**

A rotation matrix can be formed as the product of three rotations around three cardinal axes, e.g. x, y and z or x, y, and x.

- **Axis/angle**

A rotation can be represented by a rotation axis $\hat{n}$ and an angle θ, or equivalently by a 3D vector $\omega = \theta\hat{n}$.

- **Unit quaternion**

Quaternion can be derived from the axis/angle representation:

$$\boldsymbol{q} = (x, y, z, w)$$

$$\boldsymbol{q} = (\boldsymbol{v}, w) = (\sin\frac{\theta}{2}\hat{\boldsymbol{n}}, \cos\frac{\theta}{2})$$

- **Viewpoint + in-plane rotation**

[Richard Szeliski, 2010]

Object pose estimation / camera from RGB

Objective: $[{}^{c}_{m}R|{}^{c}_{m}t]$

- Geometric relationship between object 2D projections on image and relevant object 3D position:

$$s\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = K[{}^{c}_{m}R|{}^{c}_{m}t]\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

- Learn the mapping from RGB image space to object pose space(rigid body transformation, SE(3)) using supervision

Regression variants

Direct translation/rotation regression:

- 3D translation vector
- 2D object translation + distance from camera to object
- 4D unit quaternion vector
- Viewpoint + in-plane rotation

Use geometric relationship:

- 3D object point position in camera frame
- 2D object point projections in image + PnP

Loss function

- **Pose regression loss**(translation **x** + quaternion **q**)

$$loss(I) = \|\hat{\mathbf{x}} - \mathbf{x}\|_2 + \beta \left\| \hat{\mathbf{q}} - \frac{\mathbf{q}}{\|\mathbf{q}\|} \right\|_2$$

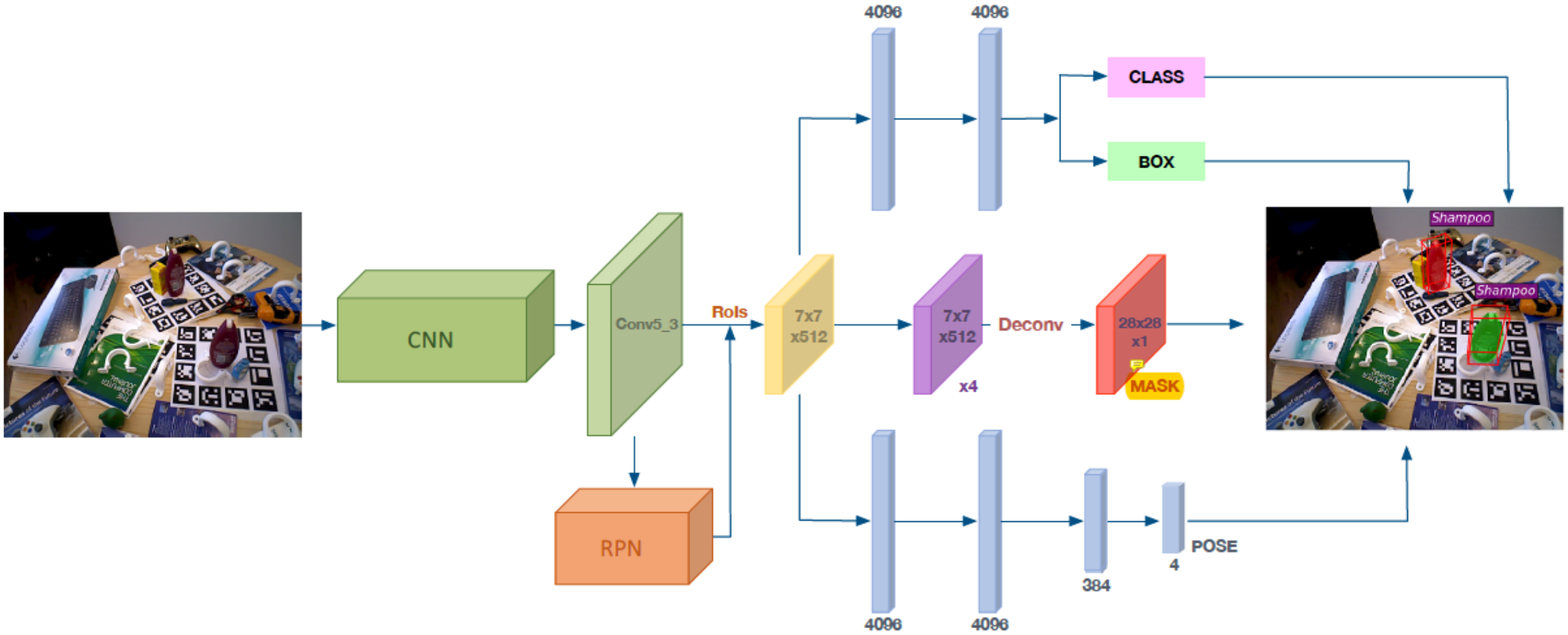
- **Point matching loss**(translation **t** + rotation **R**)

$$L_{\text{pose}}(\mathbf{p}, \hat{\mathbf{p}}) = \frac{1}{n} \sum_{j=1}^{n} L_1\big((\mathbf{R}\mathbf{x}_j + \mathbf{t}) - (\hat{\mathbf{R}}\mathbf{x}_j + \hat{\mathbf{t}})\big),$$

- **Rotation regression loss**(quaternion **q**)

$$\text{PLoss}(\tilde{\mathbf{q}}, \mathbf{q}) = \frac{1}{2m} \sum_{\mathbf{x} \in \mathcal{M}} \|R(\tilde{\mathbf{q}})\mathbf{x} - R(\mathbf{q})\mathbf{x}\|^2,$$

Multi-task learning architecture



$$L_{pose} = \|r - \hat{r}\|_p + \beta \left\|t_z - \hat{t}_z\right\|_p$$

$$L = \alpha_1 L_{cls} + \alpha_2 L_{box} + \alpha_3 L_{mask} + \alpha_4 L_{pose}$$

[Thanh-Toan Do et al, 2018]

Composition of different modules

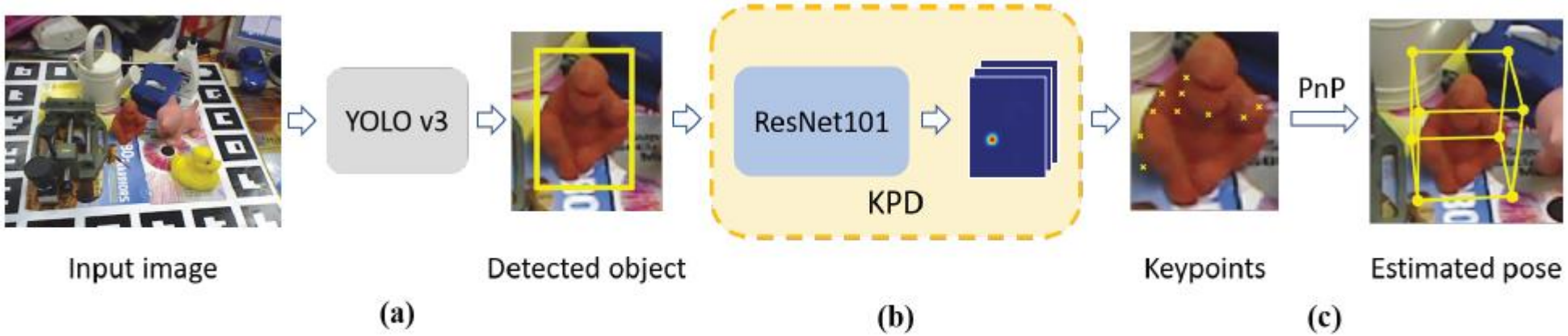
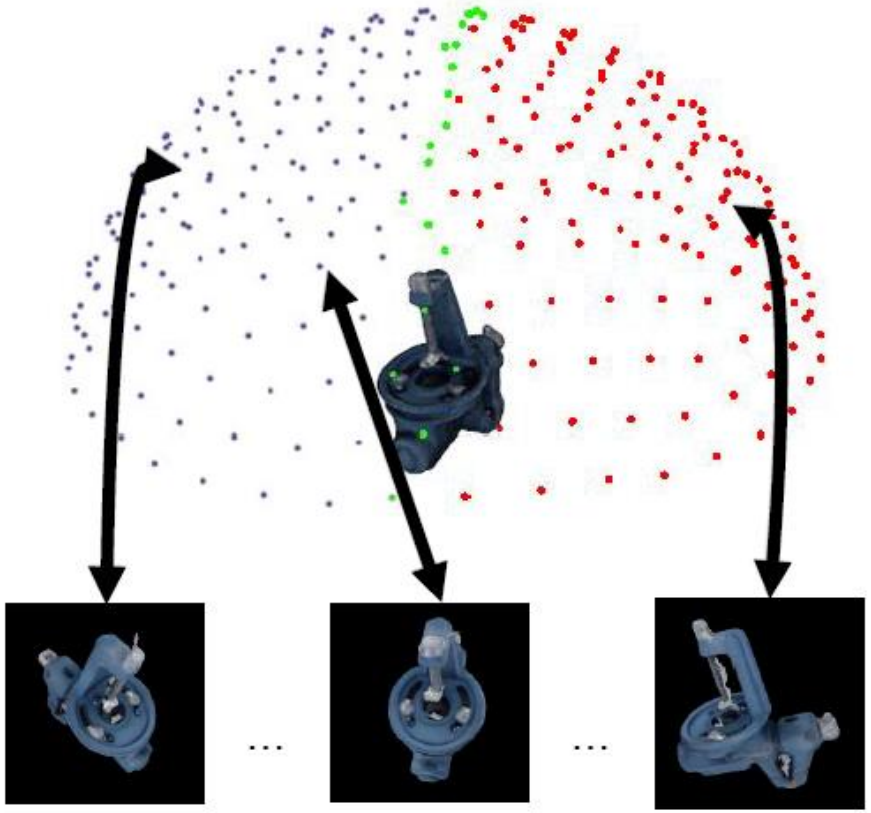


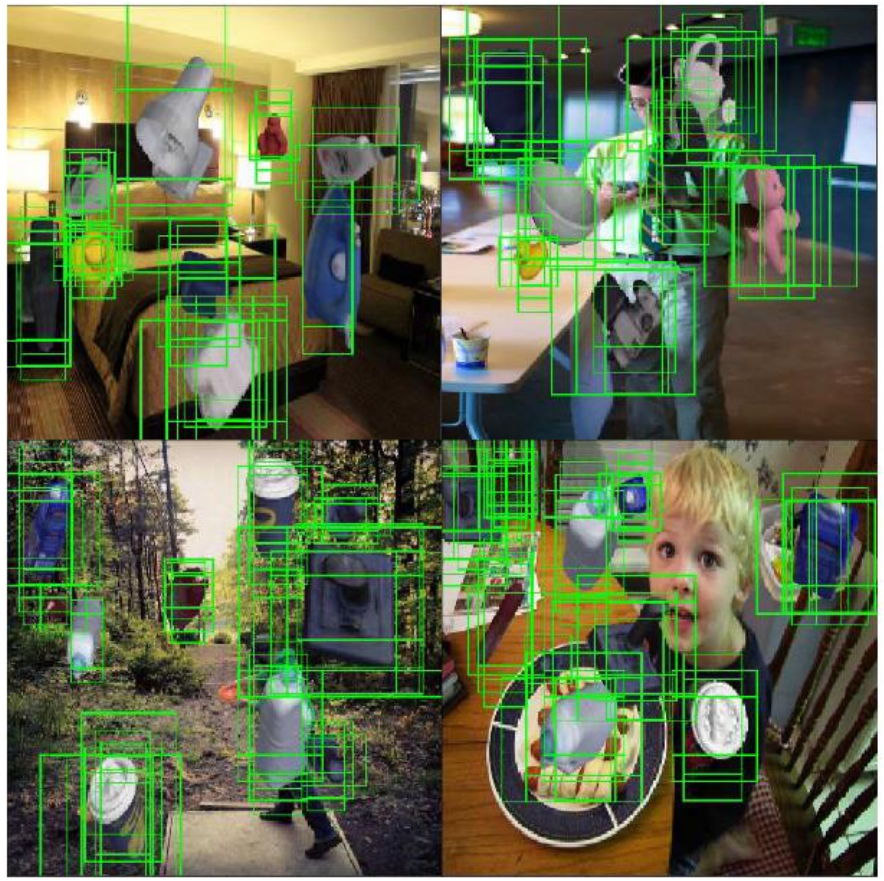
Figure 1. Visualization of our proposed pipeline. We first (a) detect bounding box then (b) localize the designated keypoints using keypoint detector (KPD). Finally (c) we use a PnP algorithm to recover the 6D pose.

[Zelin Zhao et al, 2018]

Training data

- (image, label) pair with discrete 6D pose





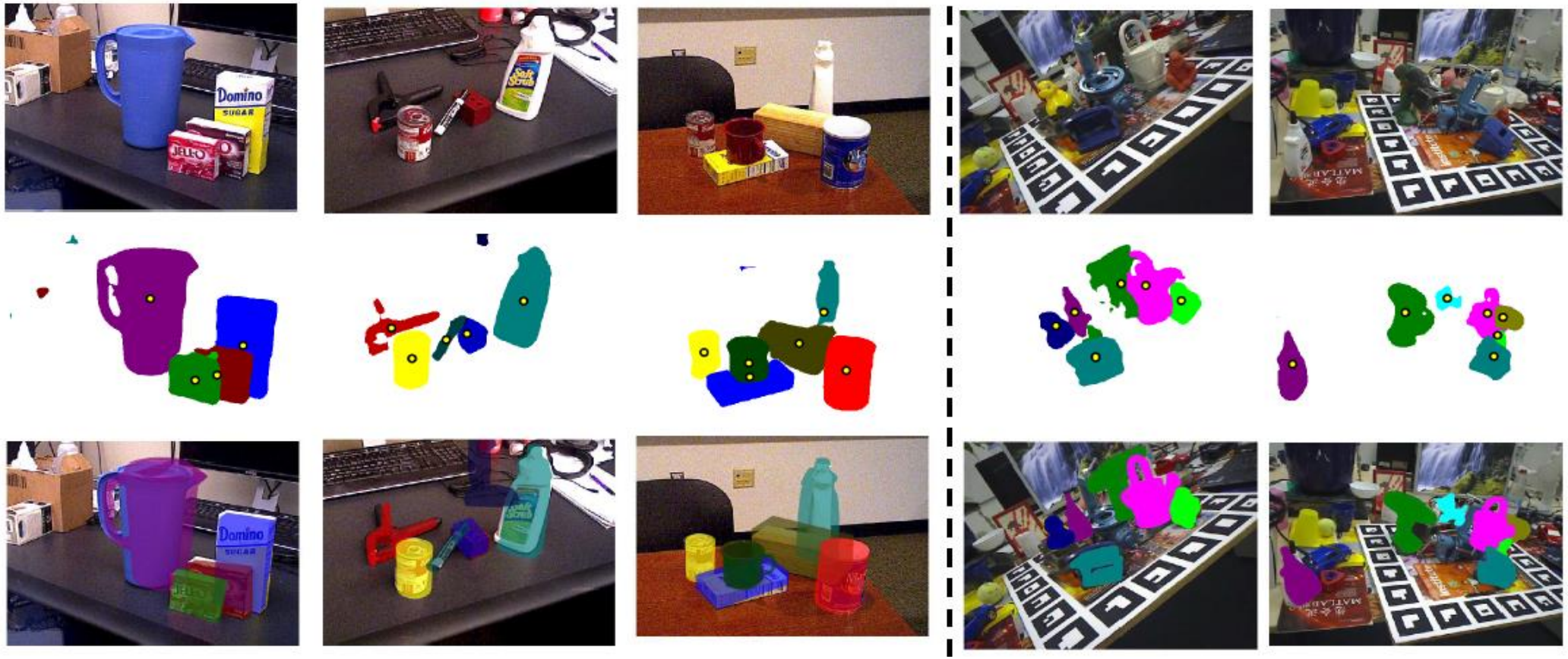
[Federico Tombari, 2018]

Test

Input Image

Labeling & Centers

PoseCNN Color



[Yu Xiang et al, 2018]

Evaluation metric

- **Translation estimation error:**

  L2 distance between prediction t and ground truth t*

- **Rotation estimation error:**

The **distance between rotations** represented by rotation matrices P and Q is the *angle of the difference rotation* represented by the rotation matrix $R = PQ^*$.

We can retrieve the angle of the difference rotation from the trace of R

$$\operatorname{tr} R = 1 + 2\cos\theta$$

again using the arccos

$$\theta = \arccos \frac{\operatorname{tr} R - 1}{2}. \qquad (2)$$

Please refer to http://www.boris-belousov.net/2016/12/01/quat-dist/

Evaluation metric(success rate)

- **5cm5degree metric**
- **ADD metric**(so-called 6D pose metric)

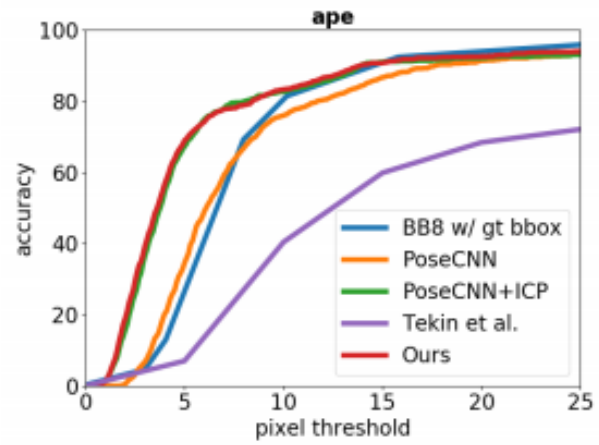
$$\frac{1}{|\mathcal{V}|}\sum_{\mathbf{M}\in\mathcal{V}}\|\mathrm{Tr}_{\hat{\mathbf{e}},\hat{\mathbf{t}}}(\mathbf{M})-\mathrm{Tr}_{\bar{\mathbf{e}},\bar{\mathbf{t}}}(\mathbf{M})\|_2$$

Threshold: 10%, 20%, 30% of object size(diagonal)

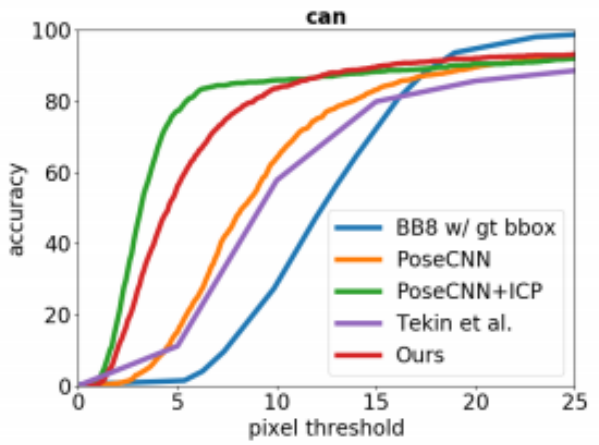
- **2D Projections metric**

A pose is considered correct if the average of 2D distance between the projections of the object's vertices from the estimated pose and the GT pose is less than a threshold(5 pix, 10 pix, …).

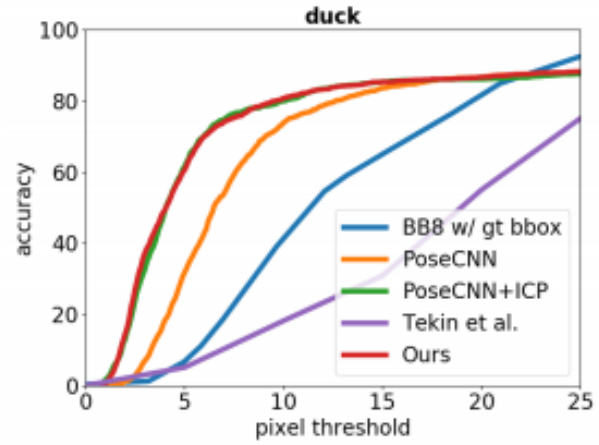
2D Projections metric



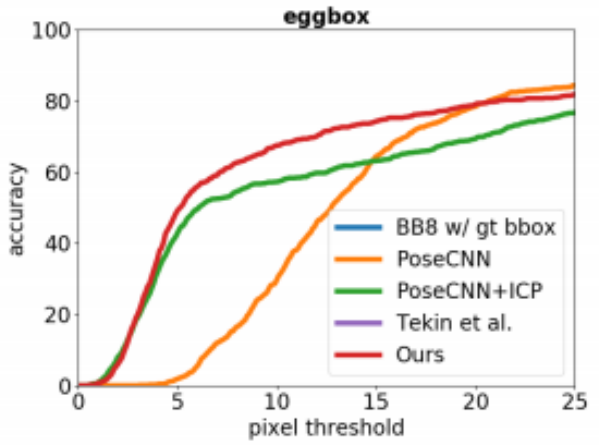
(a) ape



(b) can



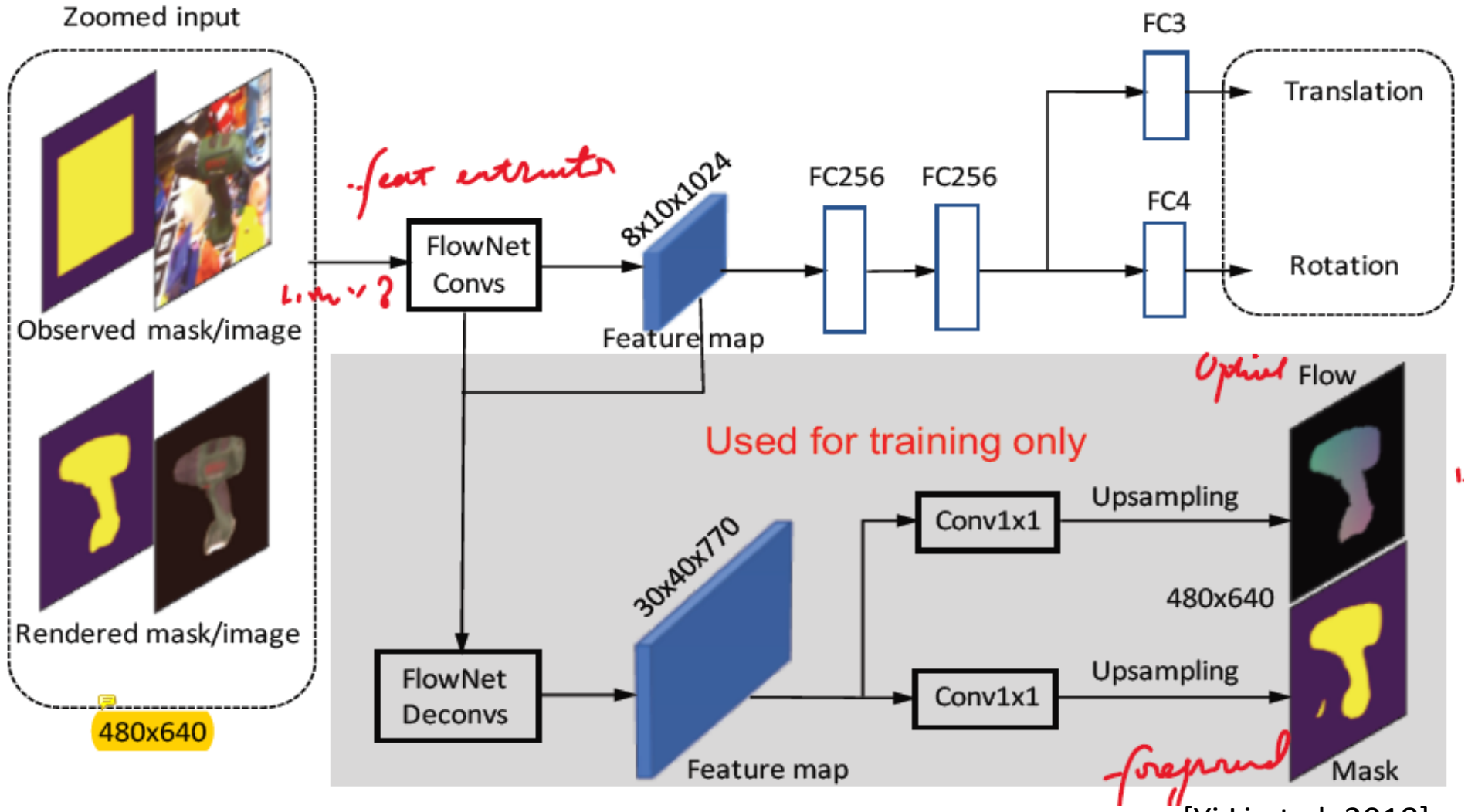
(e) duck



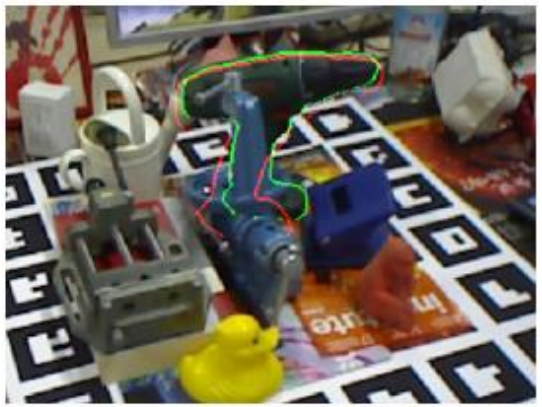
(f) eggbox

[Yi Li et al, 2018]

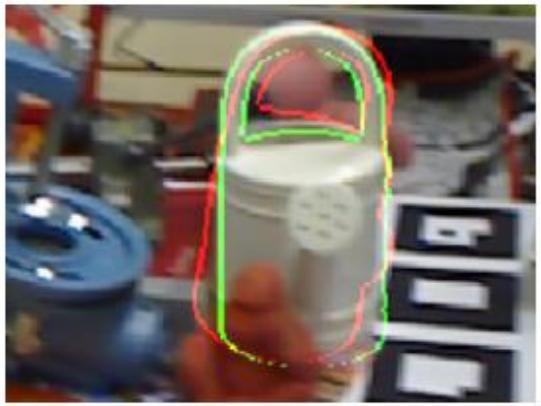
6D pose refinement



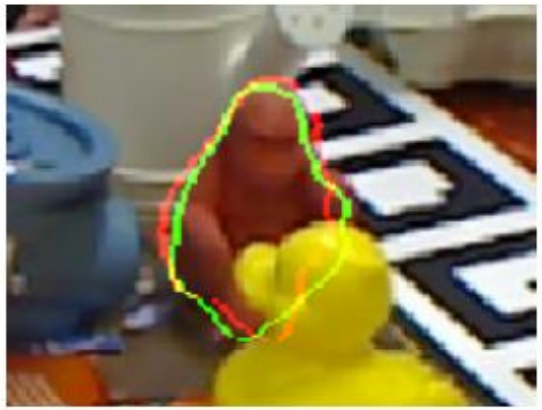
[Yi Li et al, 2018]



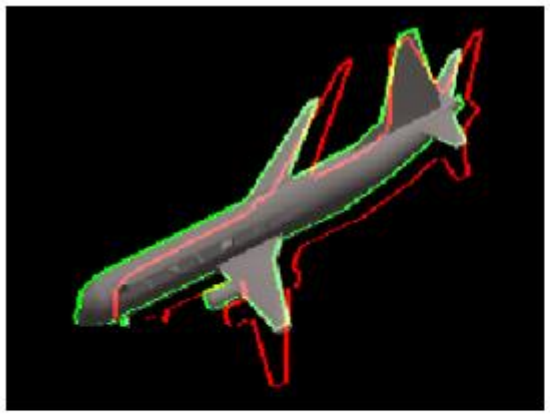
(a) driller



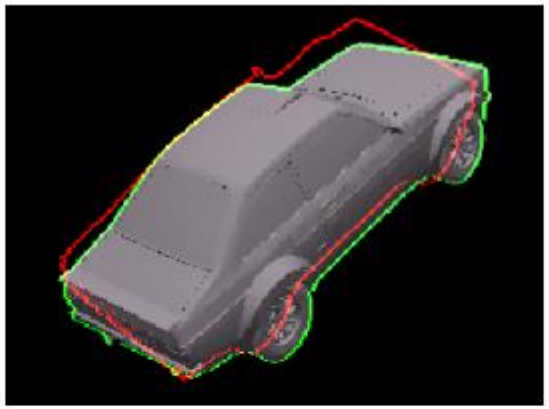
(b) can



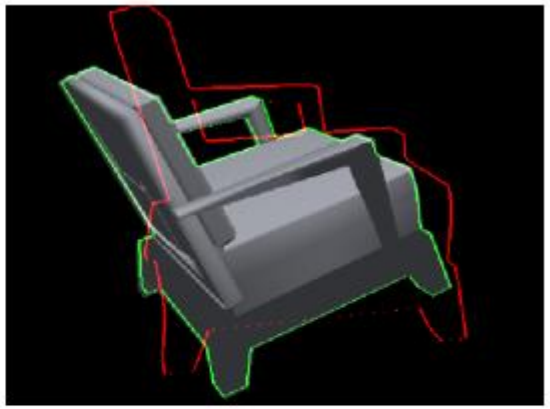
(c) ape



(a) airplane



(b) car



(c) chair

Fig. 6: Results on pose refinement of 3D models from the ModelNet dataset. These instances were not seen in training. The red and green lines represent the edges of the initial estimates and our refined poses.

[Yi Li et al, 2018]

Some Benchmarks

- **SIXD Challenge**

http://cmp.felk.cvut.cz/sixd/challenge_2017/

- **YCB-video**

https://github.com/yuxng/YCB_Video_toolbox

Monocular 6D object pose estimation – state of the art

Method	BACKBONE	Network Output	Pose Computation/Refinement
BB-8 [RAD2017]	VGG 16	8 corners of the projected 3D Bounding Box	PnP / VGG
[TEKIN2018]	YOLO V2	8 corners of the projected 3D Bounding Box + 3D centroid projection	PnP
POSECNN [XIANG2018]	VGG 16	Semantic Labeling + Regression of 6D pose	
DEEP 6D POSE [DO2018]	Mask R-CNN	Object Instance Segmentation + Regression of 6D pose	
SSD-6D [KEHL 2017]	SSD 300	Viewpoint and In-Plane rotation classification	Contour-based

- [Rad2017] Rad and Lepetit BB8: A Scalable, Accurate, Robust to Partial Occlusion Method for Predicting the 3D Poses of Challenging Objects without Using Depth, ICCV2017
- [Kehl2017] Kehl et al. SSD-6D: Making RGB-Based 3D Detection and 6D Pose Estimation Great Again, ICCV 2017
- [Tekin2018] Tekin et al. Real-Time Seamless Single Shot 6D Object Pose Prediction, CVPR 2018
- [Xiang2018] Xiang et al. PoseCNN: A Convolutional Neural Network for 6D Object Pose Estimation in Cluttered Scenes, RSS 2018
- [Do2018] Do et al. Deep-6DPose: Recovering 6D Object Pose from a Single RGB Image, Arxive 2018

[Federico Tombari, 2018]

Challenges

- Lack of depth information, the distance estimation from camera to object is not so accurate and contains ambiguity.
- When object is partially occluded, estimation accuracy decreases.
- Always fully supervised, can not handle unseen objects during training.

Reference

- Federico Tombari. From 3D descriptors to monocular 6D pose: what have we learned? In ECCV workshop, 2018
- Richard Szeliski. Computer Vision: Algorithms and Applications. Springer, 2010
- Yu Xiang, Tanner Schmidt, Venkatraman Narayanan and Dieter Fox. PoseCNN: A Convolutional Neural Network for 6D Object Pose Estimation in Cluttered Scenes. In CVPR, 2018
- Thanh-Toan, Ming Cai, Trung Pham, Ian Reid. Deep-6DPose: Recovering 6D Object Pose from a Single RGB Image. In ArXiv, 2018
- Zelin Zhao, Gao Peng, Haoyu Wang, Hao-shu Fang, Chengkun Li, Cewu Lu. Eestinating 6D pose from localizing designated surface keypoints. In arXiv, 2018
- Yi Li, Gu Wang, Xiangyang Ji, Yu Xiang and Dieter Fox. DeepIM: Deep Iterative Matching for 6D pose estimation. In ECCV, 2018