Take a ~~Deeper~~ Look at Object 6D Pose Estimation

Siyu ZHANG

Research Engineer

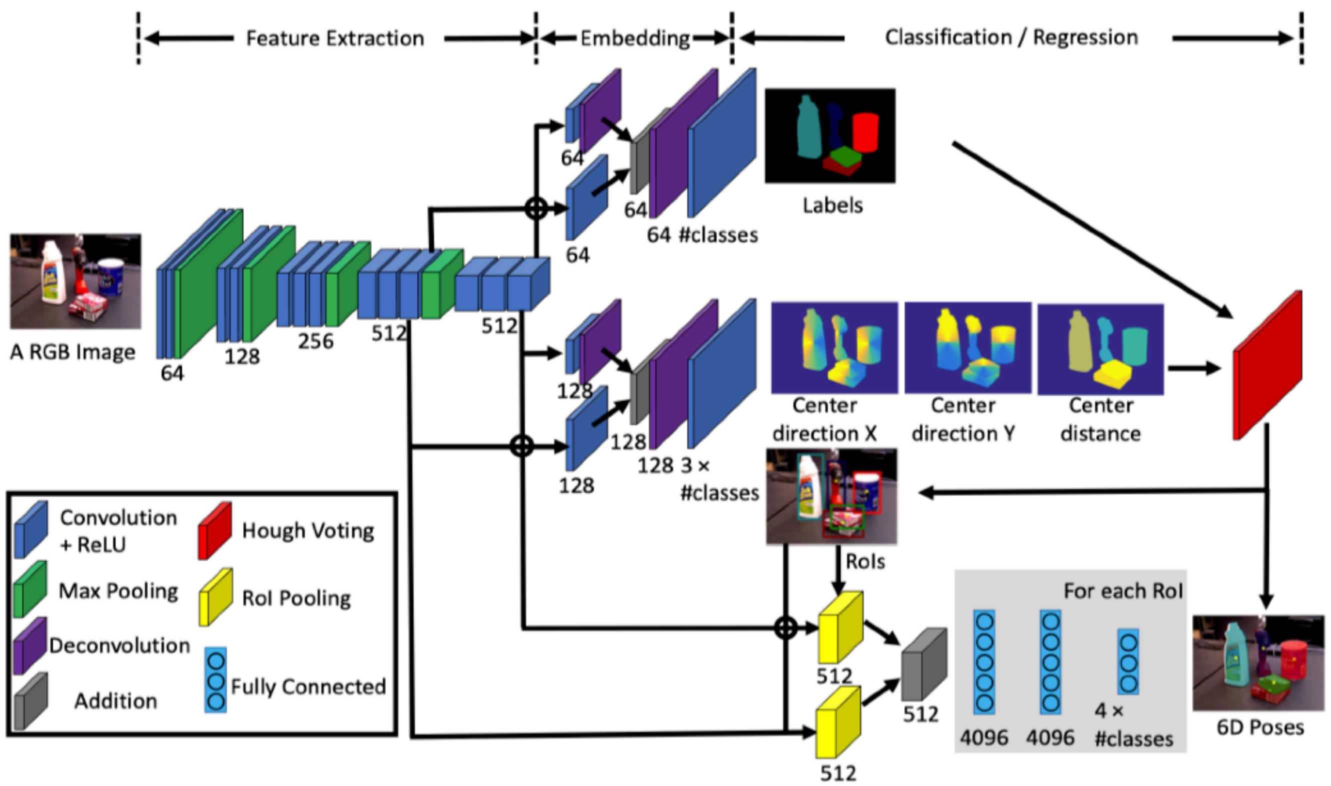
ZJU-SenseTime Joint Lab of 3D Vision

Overview

- Some terminologies …
  - Pose Estimation
    - Given: Image_t
    - Target: Pose_t
  - Pose Refinement
    - Given: Image_t, Pose_t_init
    - Target: Pose_t_refined
  - Pose Tracking
    - Given: Image_t-1, Pose_t-1, Image_t
    - Target: Pose_t

Overview

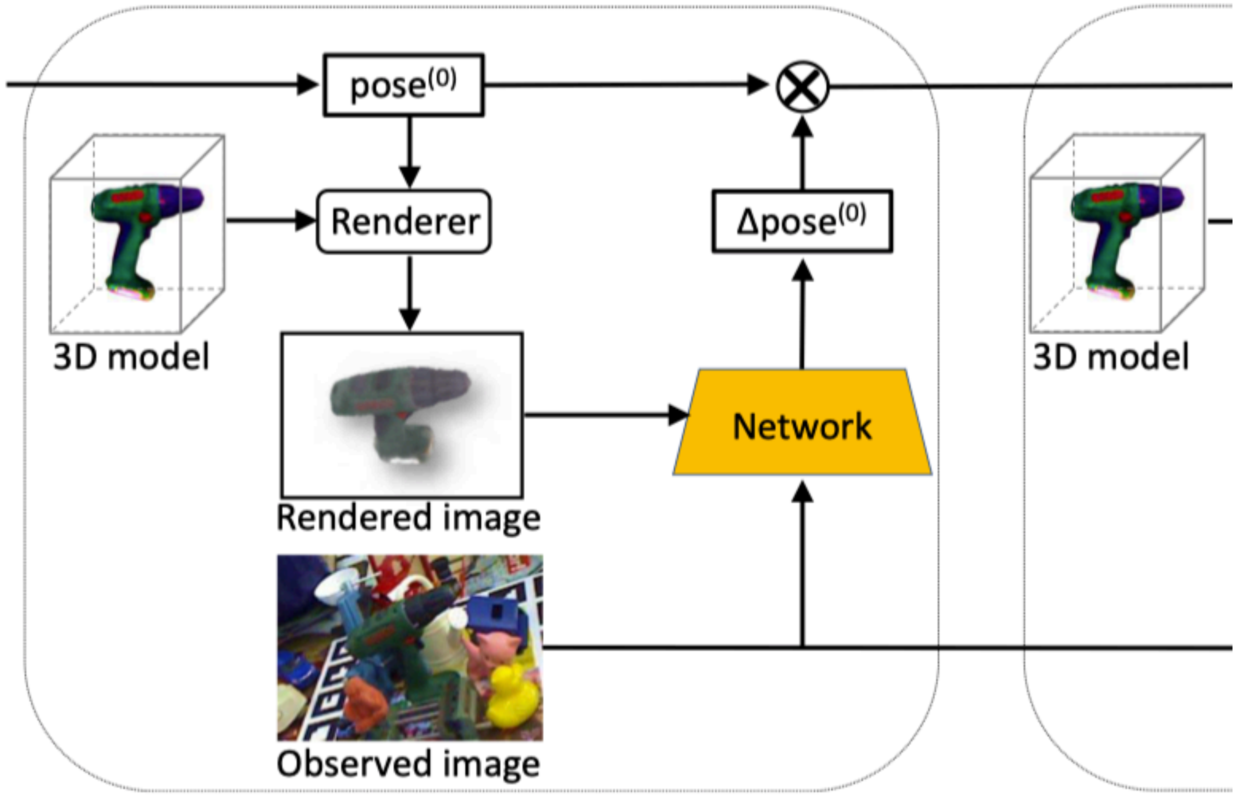
- As a **regression** problem
  - Pose Estimation: direct regression
    - Extract feature from image
    - Directly regress the translation and orientation of target object



PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scenes. RSS, 2018

Overview

- As a **regression** problem
  - Pose Tracking: render and regression
    - Render the image of target object with initial pose and object 3D model
    - Feed the rendered image and input image into Neural Network to extract feature
    - Regress the additive pose for target object



DeepIM: Deep Iterative Matching for 6D Pose Estimation, ECCV, 2018

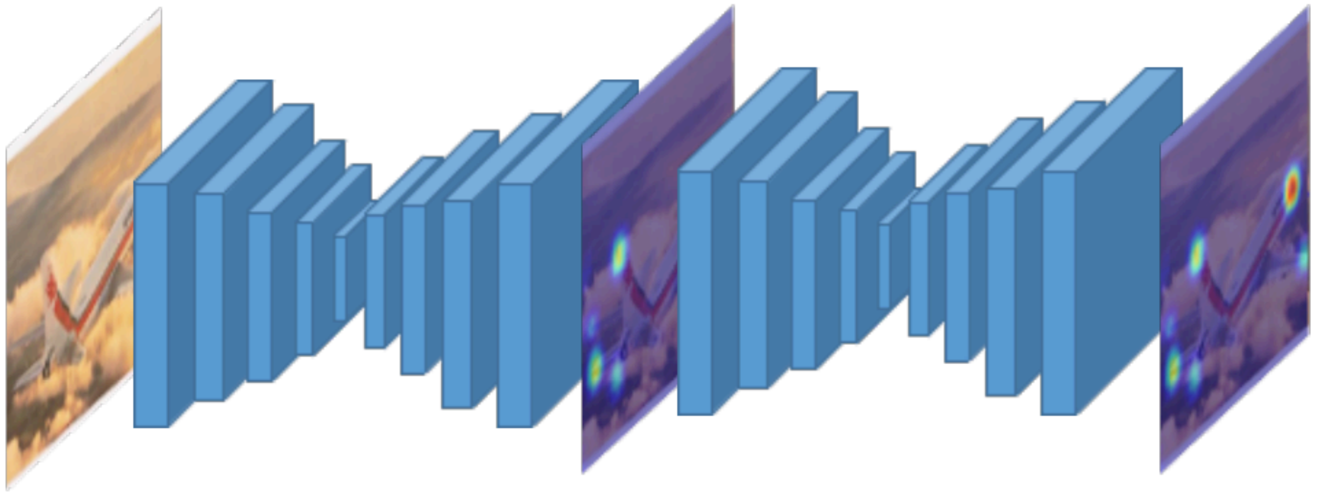
Overview

- As a **regression** problem
  - Pose Estimation: direct regression from image feature
  - Pose Tracking: render, compare with observed image, then regress relative (additive) pose

- As a **matching** problem

Overview

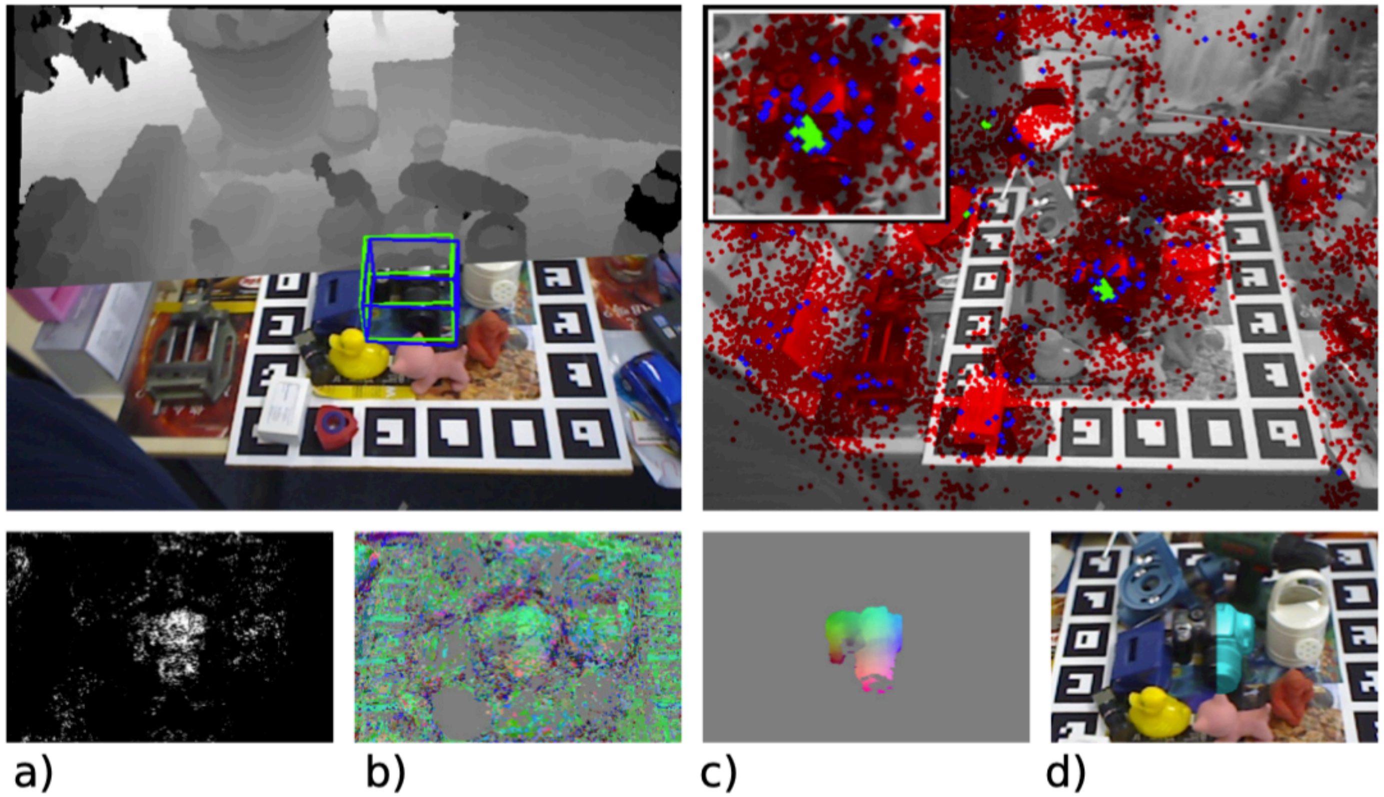
- As a **matching** problem
  - Pose Estimation by matching sparse correspondences
    - Extract semantic 2D keypoint from the image
    - Solve for PnP with corresponding 3D keypoints in object frame



6-DoF Object Pose from Semantic Keypoints, ICRA, 2017

Overview

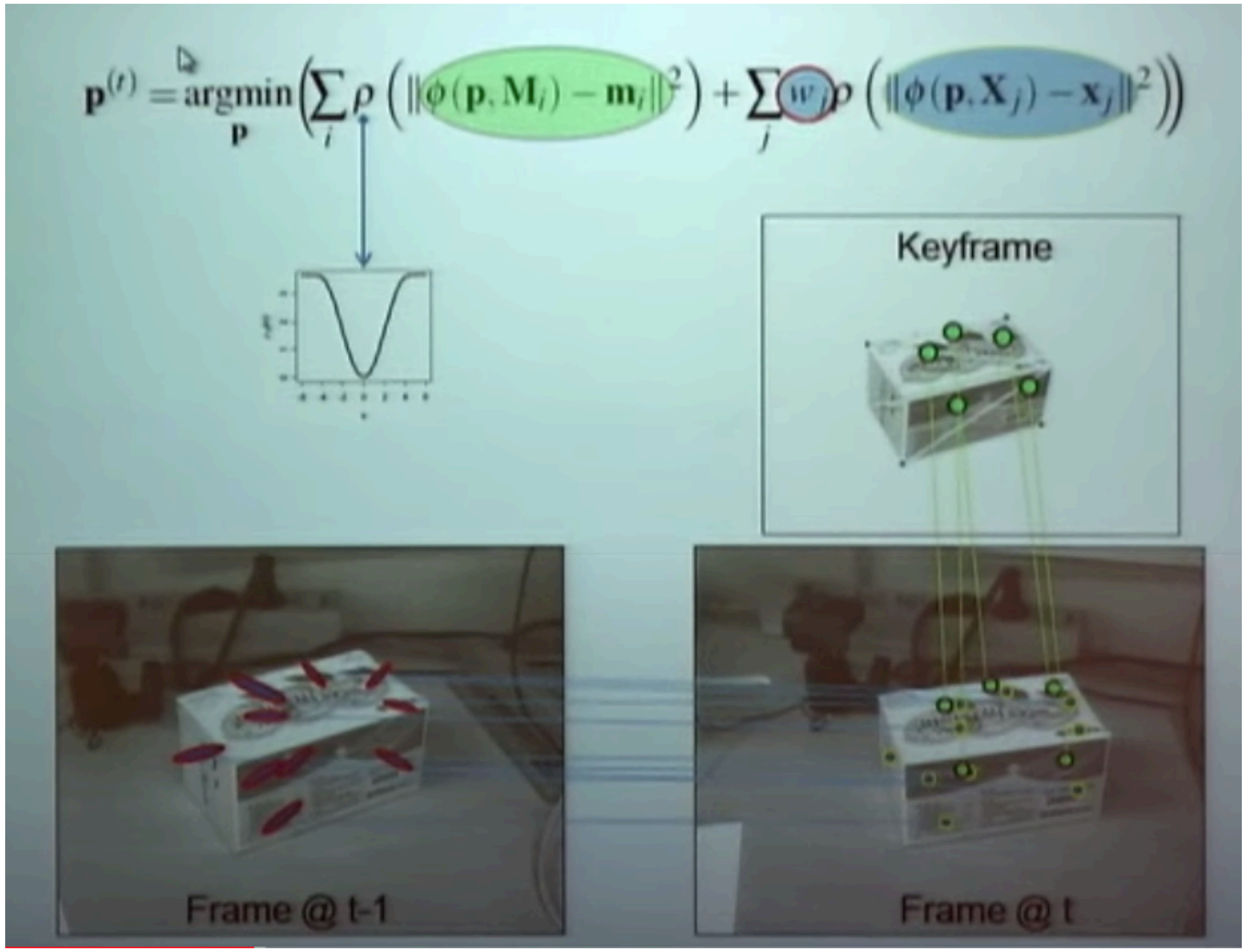
- As a **matching** problem
  - Pose Estimation by matching dense correspondences
    - Estimate pixel-wise object coordinates for all foreground pixels
    - Apply RANSAC + PnP to solve for object pose



Learning 6D Object Pose Estimation using 3D Object Coordinates, ECCV, 2014

Overview

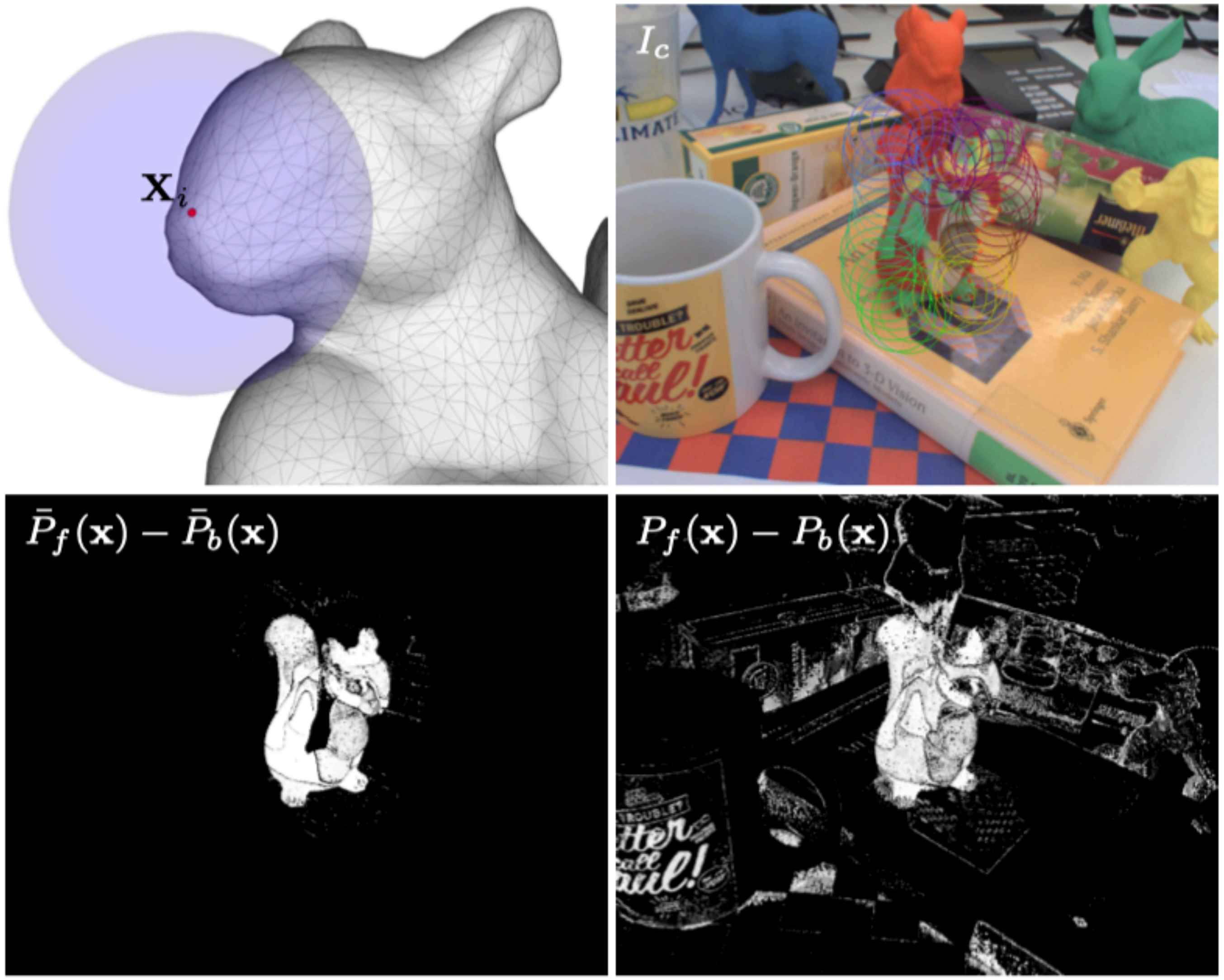
- As a **matching** problem
  - Pose Tracking by keypoint tracking
    - Extract keypoint and descriptor from input images of different frames
    - Using matched keypiont to estimate relative pose between different frames



Multiple 3D Object Tracking, TVCG, 2011; [*https://www.youtube.com/watch?v=eqlEzWmuijs*](https://www.youtube.com/watch?v=eqlEzWmuijs)

Overview

- As a **matching** problem
  - Pose Tracking by silhouette tracking:
    - Project object mesh with initial object pose and extract silhouette
    - Compute residual of current silhouette according to local foreground-background similarity
    - Optimize object pose to minimize the residual b gradient-based approach



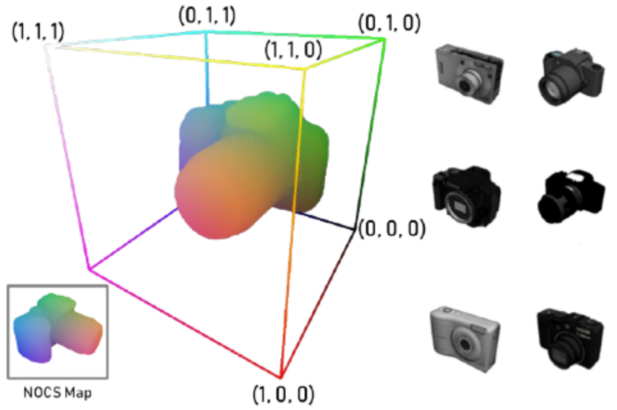
A Region-based Gauss-Newton Approach to Real-Time Monocular Multiple Object Tracking, TPAMI, 2018

Overview

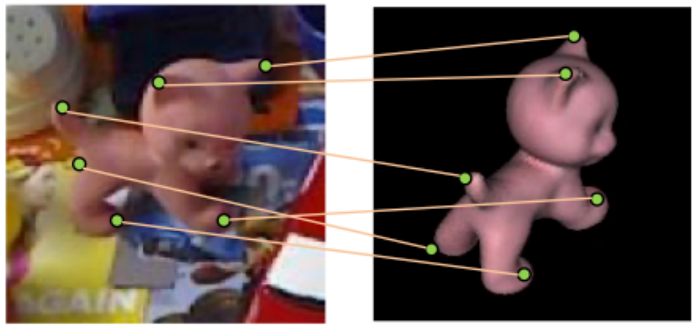
- As a **regression** problem
  - Pose Estimation: direct regression
  - Pose Tracking: render and regression
- As a **matching** problem
  - Pose Estimation: matching from image pixels to points in object frame
  - Pose Tracking： matching between frames
- Msic.
  - Tracking by Detection: Single frame estimator + filtering (smoothing)
  - Coarse-to-fine estimation: Coarse pose initialization (or sampling) + Iterative refinement

Overview

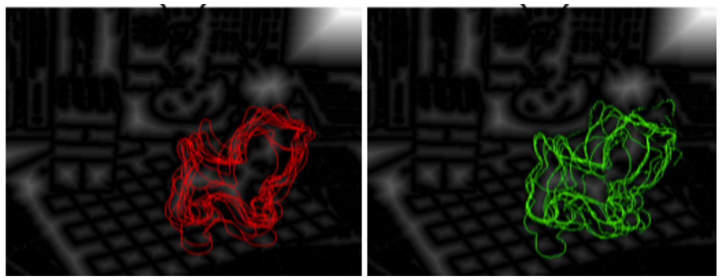
- Pose estimation is a solved problem when:
  - Using sparse keypoint as representation
  - Given adequate well-annotated data and precise object model
- Some recent trends
  - sparse representation to denser representation
  - instance-level to category-level
  - model-based to model-free



Dense - Object Coordinates



Sparse - Key Points



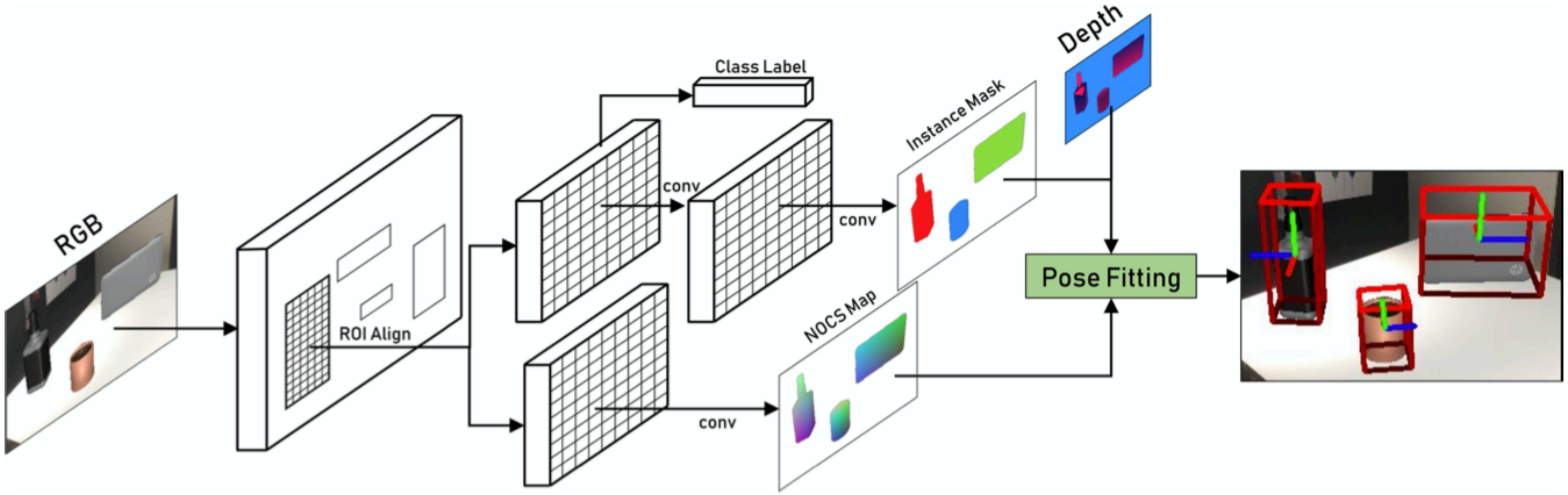
Semi-dense - Silhouette/Edges

Overview

- Paper to cover

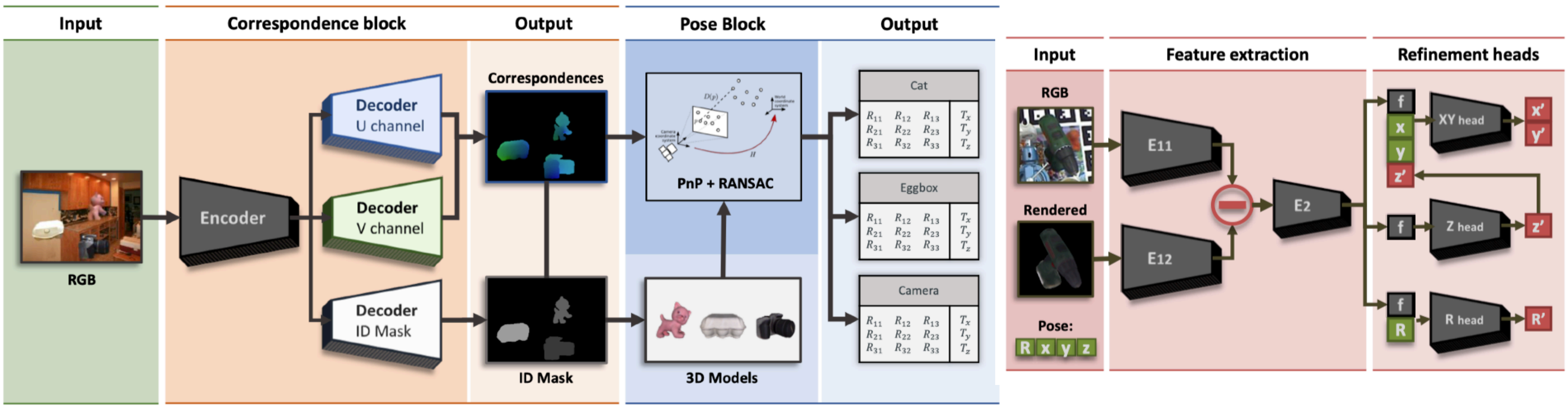
Paper Name	Conference	Model-Free	Type	Representation
NOCS	CVPR19	at inference	category-level	NOCS
DPOD	ICCV19	No	instance-level	UV Map
Hybrid Pose	CVPR20	No	instance-level	hybrid of geometric primitives
6-Pack	ICRA20	at training and inference	category-level	keypoints
Category Level Object Pose Estimation via Neural Analysis-by-Synthesis	ECCV20	at training and inference	category-level	latent vector
Shape Prior Deformation for Categorical 6D Object Pose and Size Estimation	ECCV20	at inference	category-level	NOCS
LatentFusion	CVPR20	at training and inference	unrestricted	latent volume
Reconstruct Locally, Locally Globally	CVPR20	at training and inference	instance-level	object coordinates

Pose Estimation with Various Representations



- NOCS
  - Representation: Normalize Object Coordinate Space
  - Approach:
    - Estimate NOCS Map, lift to original scale with depth
    - Obtain object point cloud with depth and instance mask
    - Estimate pose with transformation from object coordinate to object point cloud

Pose Estimation with Various Representations



- DPOD
  - Representation: UV Map
  - Approach:
    - Estimate UV Map and instance mask
    - Estimate pose using RANSAC + PnP
    - Refine pose with render and compare

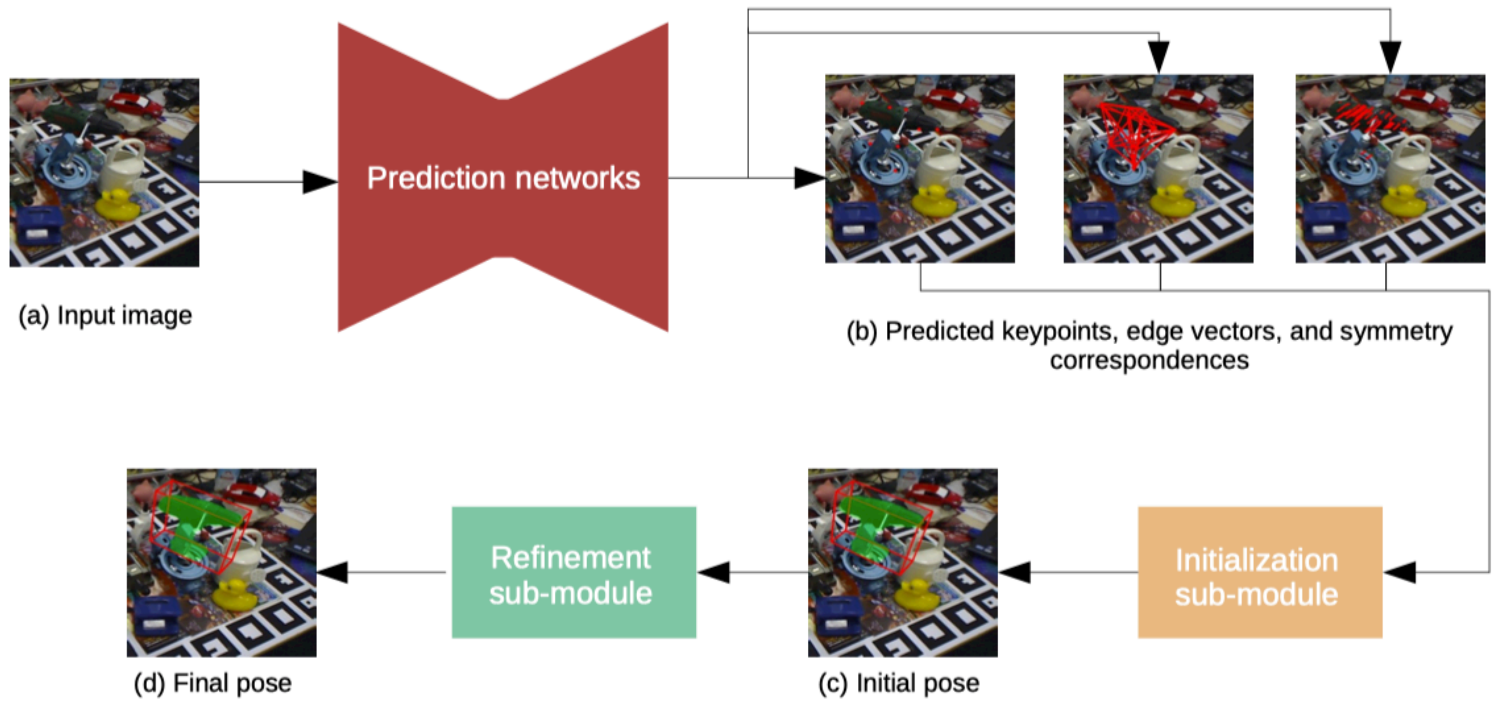
Pose Estimation with Various Representations

- DPOD
  - Experiment result on LineMOD
  - Discussion: UV Map VS. NOCS

w/o Refinement		+ Refinement	
PVNet [25]	**Ours**	**DeepIM** [18]	**Ours**
43.62	**53.28**	77.0	**87.73**
99.90	95.34	97.5	**98.45**
86.86	**90.36**	93.5	**96.07**
95.47	94.10	96.5	**99.71**
79.34	60.38	82.1	**94.71**
96.43	**97.72**	95.0	**98.80**
52.58	**66.01**	77.7	**86.29**
99.15	**99.72**	97.1	**99.91**
95.66	93.83	**99.4**	96.82
81.92	65.83	52.8	**86.87**
98.88	**99.80**	98.3	**100**
99.33	88.11	**97.5**	96.84
92.41	74.24	87.7	**94.69**
86.27	82.98	88.6	**95.15**

Pose Estimation with Various Representations

- Hybrid-Pose



$$\overline{\boldsymbol{r}}^{\mathcal{K}}_{R,\boldsymbol{t}}(\boldsymbol{p}_k) := \hat{\boldsymbol{p}}_k \times (R\overline{\boldsymbol{p}}_k + \boldsymbol{t}), \quad (1)$$

$$\overline{\boldsymbol{r}}^{\mathcal{E}}_{R,\boldsymbol{t}}(\boldsymbol{v}_e, \boldsymbol{p}_{e_s}) := \hat{\boldsymbol{v}}_e \times (R\overline{\boldsymbol{p}}_{e_t} + \boldsymbol{t}) + \hat{\boldsymbol{p}}_{e_s} \times (R\overline{\boldsymbol{v}}_e) \quad (2)$$

$$r^{\mathcal{S}}_{R,\boldsymbol{t}}(\boldsymbol{q}_{s,1}, \boldsymbol{q}_{s,2}) := (\hat{\boldsymbol{q}}_{s,1} \times \hat{\boldsymbol{q}}_{s,2})^T \boldsymbol{R}\overline{\boldsymbol{n}}_r. \quad (3)$$

Residual for initial pose estimation

$$\min_{R,\boldsymbol{t}} \sum_{k=1}^{|\mathcal{K}|} \rho(\|\boldsymbol{r}^{\mathcal{K}}_{R,\boldsymbol{t}}(\boldsymbol{p}_k)\|, \beta_{\mathcal{K}}) \|\boldsymbol{r}^{\mathcal{K}}_{R,\boldsymbol{t}}(\boldsymbol{p}_k)\|^2_{\Sigma_k} + \frac{|\mathcal{K}|}{|\mathcal{E}|} \sum_{e=1}^{|\mathcal{E}|} \rho(\|\boldsymbol{r}^{\mathcal{E}}_{R,\boldsymbol{t}}(\boldsymbol{v}_e)\|, \beta_{\mathcal{E}}) \|\boldsymbol{r}^{\mathcal{E}}_{R,\boldsymbol{t}}(\boldsymbol{v}_e)\|^2_{\Sigma_e} + \frac{|\mathcal{K}|}{|\mathcal{S}|} \sum_{s=1}^{|\mathcal{S}|} \rho(r^{\mathcal{S}}_{R,\boldsymbol{t}}(\boldsymbol{q}_{s,1}, \boldsymbol{q}_{s,2}), \beta_{\mathcal{S}}) \quad (9)$$

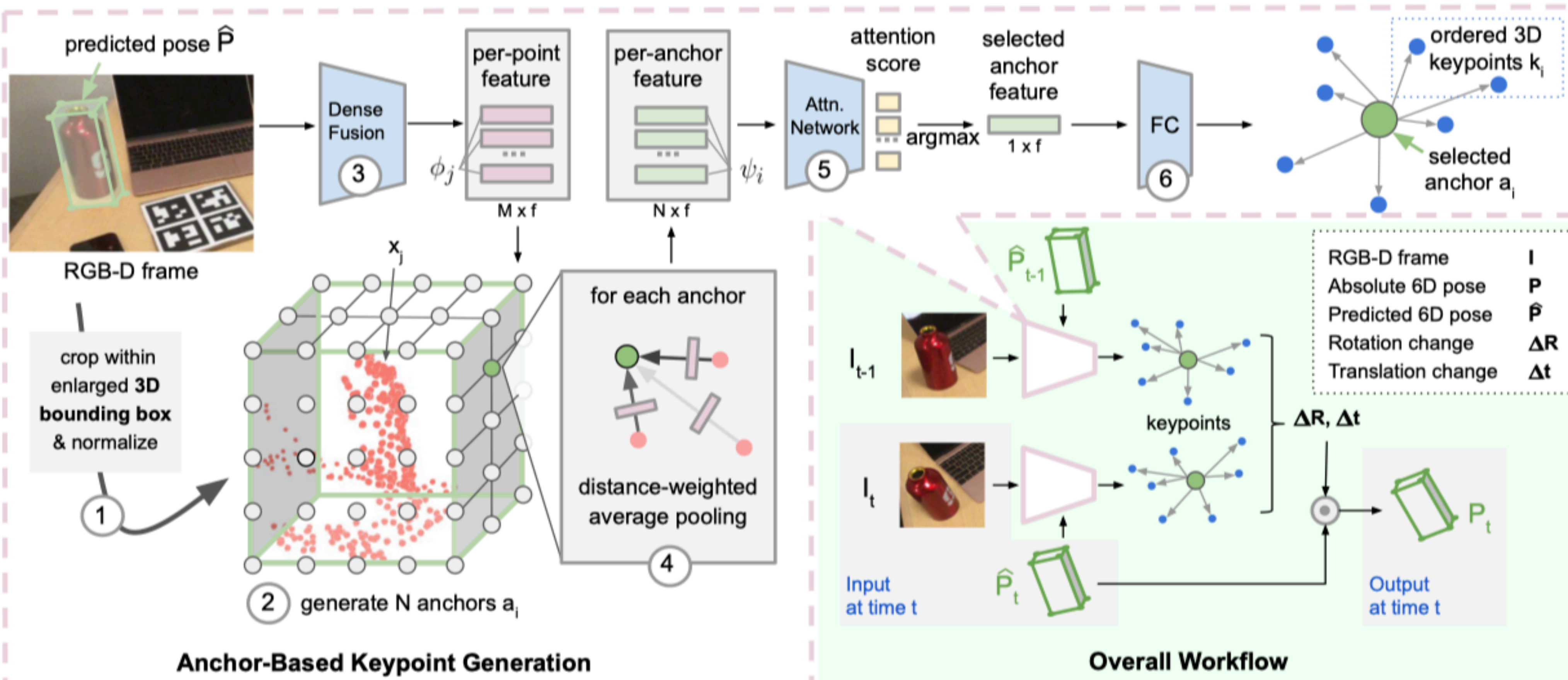
Residual for pose refinement

- <u>In a nut shell:</u> estimate multiple geometric primitives with network, then optimize over them
  - Geometric primitives:
    - Pixel coordinate Keypoint
    - 2D edges between pairwise key points
    - Pairs of symmetric points

- Different formulations are used for pose estimation and refinement
  - For pose estimation, the target is to form an Ax=b linear system
  - For pose refinement, the gradient-based optimization approach is used

Category-Level Pose Estimation

- 6-Pack



- Approach:
  - 3D anchor generation (3D grid)
  - Point-wise feature aggregated to anchor feature
  - Anchor scoring and 3D keypoint regression
- Intuition: coarse anchor selection + fine-grained keypoint selection to enlarge search space

$$L_{anc} = \frac{1}{N}\sum_i c_i(||a_i - o_{gt}||_2 - \beta)$$

anchor scoring loss

$$L_{mvc} = \frac{1}{K}\sum_i ||k_i^t - [\Delta R_t^{gt}|\Delta t_t^{gt}] \cdot k_i^{t-1}||$$

multi-view consistency loss

$$L_{tra} = ||(\bar{k}^t - \bar{k}^{t-1}) - \Delta t_t^{gt}||$$

$$L_{rot} = 2\arcsin(\frac{1}{2\sqrt{2}}||\Delta \hat{R}_t - \Delta R_t^{gt}||)$$

pose estimation loss

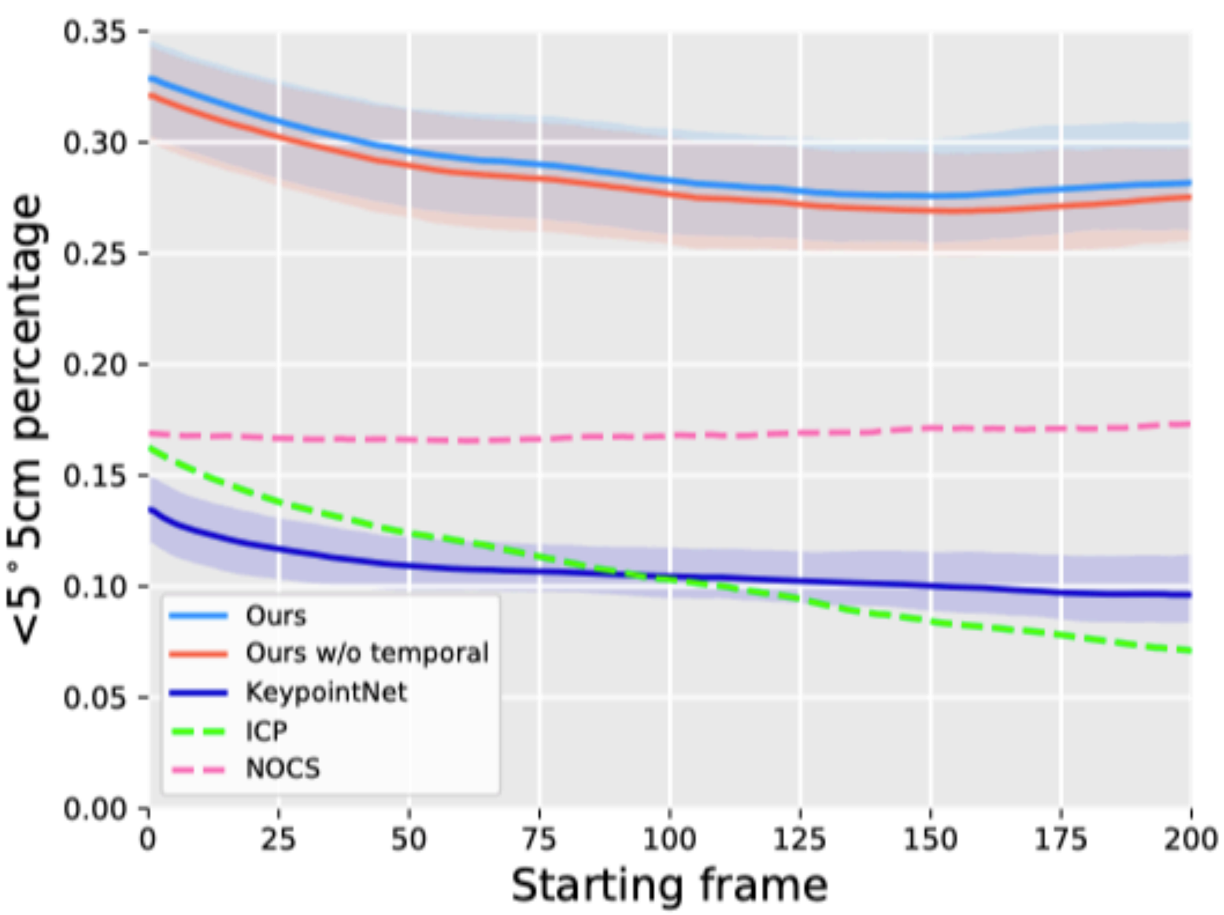
Category-Level Pose Estimation

- 6-Pack

  - Results on NOCS-REAL275

    - Accurate and stable

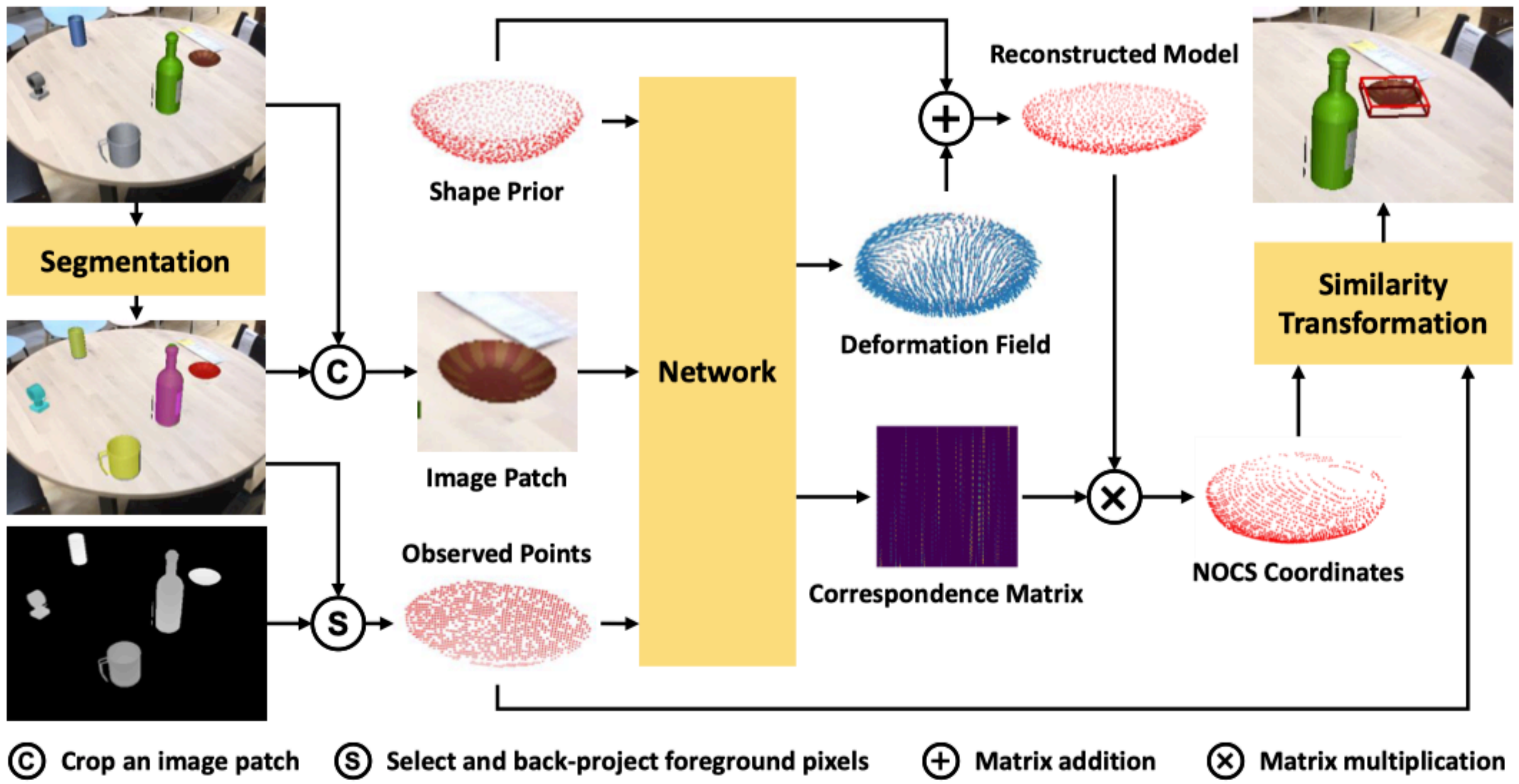
    - Run at 10 HZ with GTX1070

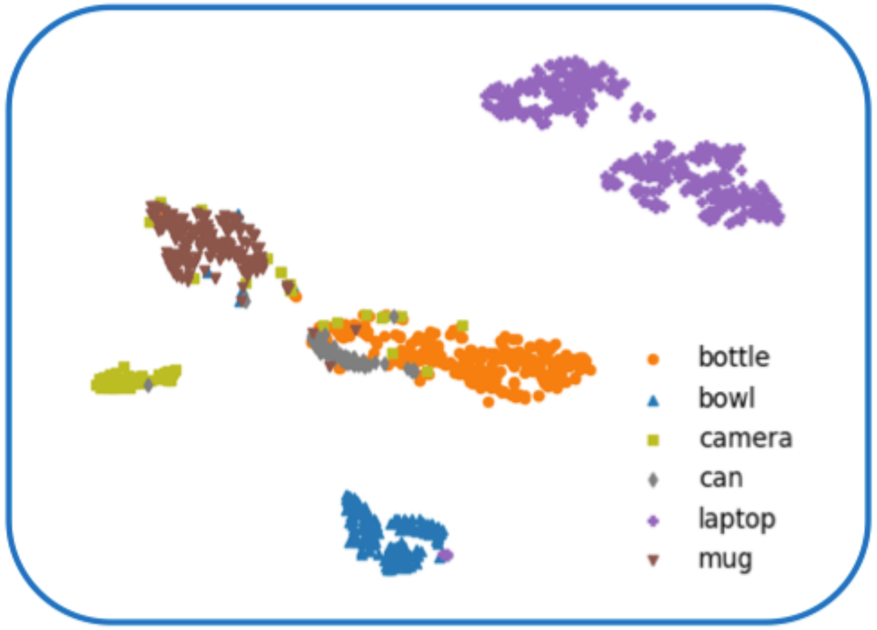


		NOCS [46]	ICP [50]	Keypoint Net [41]	Ours w/o temporal	Ours
bottle	5°5cm	5.5	10.1	5.9	23.7	**24.5**
	IoU25	48.7	29.9	23.1	**92.0**	91.1
	R_{err}	25.6	48.0	28.5	15.7	**15.6**
	T_{err}	14.4	15.7	9.5	4.2	**4.0**
bowl	5°5cm	**62.2**	40.3	16.8	53.0	55.0
	IoU25	99.6	79.7	74.7	**100.0**	**100.0**
	R_{err}	**4.7**	19.0	9.8	5.3	5.2
	T_{err}	**1.2**	4.7	8.2	1.6	1.7
camera	5°5cm	0.6	**12.6**	1.8	8.4	10.1
	IoU25	90.6	53.1	30.9	**91.0**	87.6
	R_{err}	**33.8**	80.5	45.2	43.9	35.7
	T_{err}	**3.1**	12.2	8.5	5.5	5.6
can	5°5cm	7.1	17.2	4.3	**25.0**	22.6
	IoU25	77.0	40.5	42.6	89.9	**92.6**
	R_{err}	16.9	47.1	28.8	**12.5**	13.9
	T_{err}	**4.0**	9.4	13.1	5.0	4.8
laptop	5°5cm	25.5	14.8	49.2	62.4	**63.5**
	IoU25	94.7	50.9	94.6	97.8	**98.1**
	R_{err}	8.6	37.7	6.5	4.9	**4.7**
	T_{err}	**2.4**	9.2	4.4	2.5	2.5
mug	5°5cm	0.9	6.2	3.1	22.4	**24.1**
	IoU25	82.8	27.7	52.0	**100.0**	95.2
	R_{err}	31.5	56.3	61.2	**20.3**	21.3
	T_{err}	4.0	9.2	6.7	**1.8**	2.3
Overall	5°5cm	17.0	16.9	13.5	32.5	**33.3**
	IoU25	82.2	47.0	53.0	**95.1**	94.2
	R_{err}	20.2	48.1	30.0	17.1	**16.0**
	T_{err}	4.9	10.5	8.4	**3.4**	3.5

Category-Level Pose Estimation

- Shape Prior Deformation for Categorical 6D Object Pose and Size Estimation

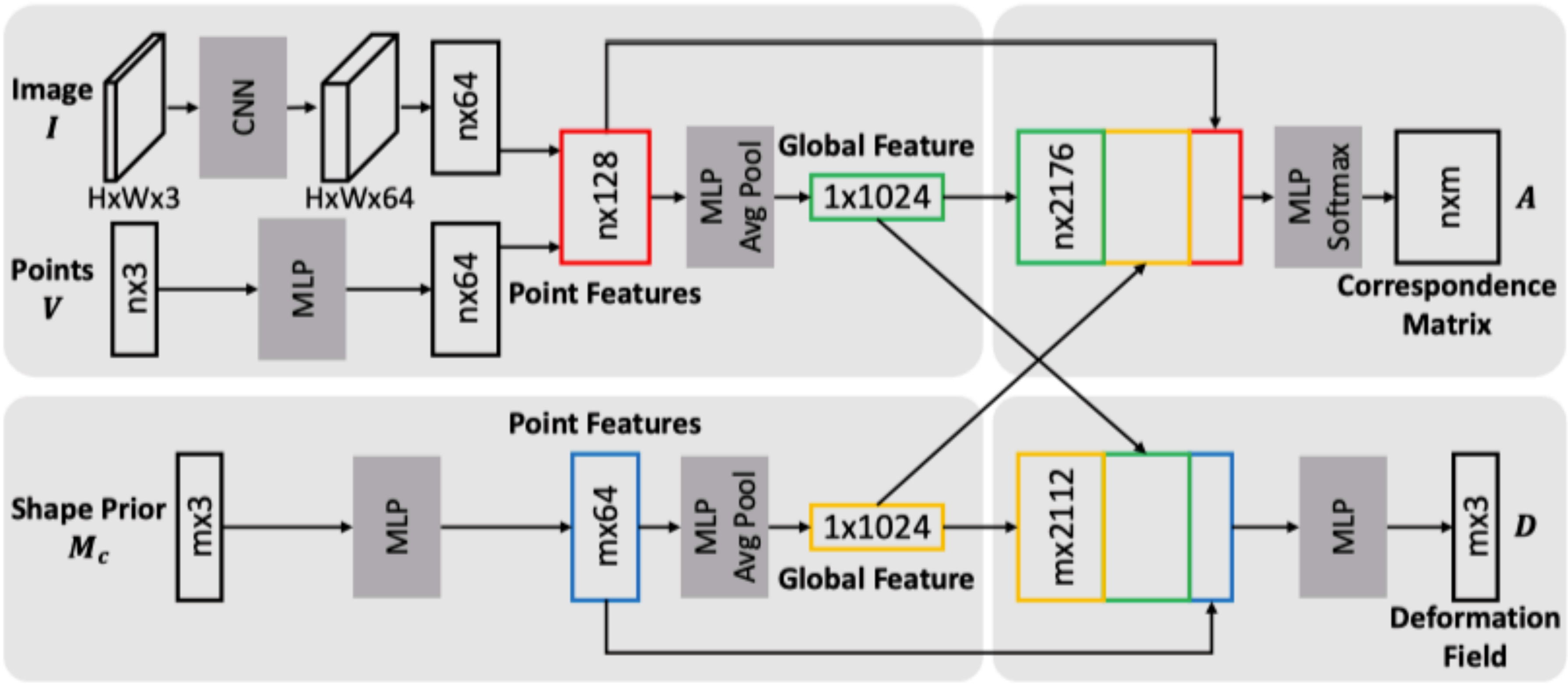




- In a nut shell: estimate NOCS with explicit category-level shape prior
  - Shape prior: mean (normalized) model point cloud by decoding mean latent vector for each category
  - Fuse feature from model point cloud, generated deformation field and correspondence.

Category-Level Pose Estimation

- Shape Prior Deformation for Categorical 6D Object Pose and Size Estimation



- Loss
  - For autoencoder & deformation field:reconstruction loss (Chamfer distance)

$$d_{\mathrm{CD}}(M_c^i, \hat{M}_c^i) = \sum_{x \in M_c^i} \min_{y \in \hat{M}_c^i} \|x - y\|_2^2 + \sum_{y \in \hat{M}_c^i} \min_{x \in M_c^i} \|x - y\|_2^2.$$

  - For A: correspondence loss (smooth L1 with gt NOCS)

$$L_{\mathrm{corr}}(P, P_{gt}) = \frac{1}{N_v} \sum_{\mathbf{x} \in P} \sum_{i=1,2,3} \begin{cases} 5(x_i - y_i)^2, & \text{if } |x_i - y_i| \leq 0.1 \\ |x_i - y_i| - 0.05, & \text{otherwise} \end{cases},$$

Category-Level Pose Estimation

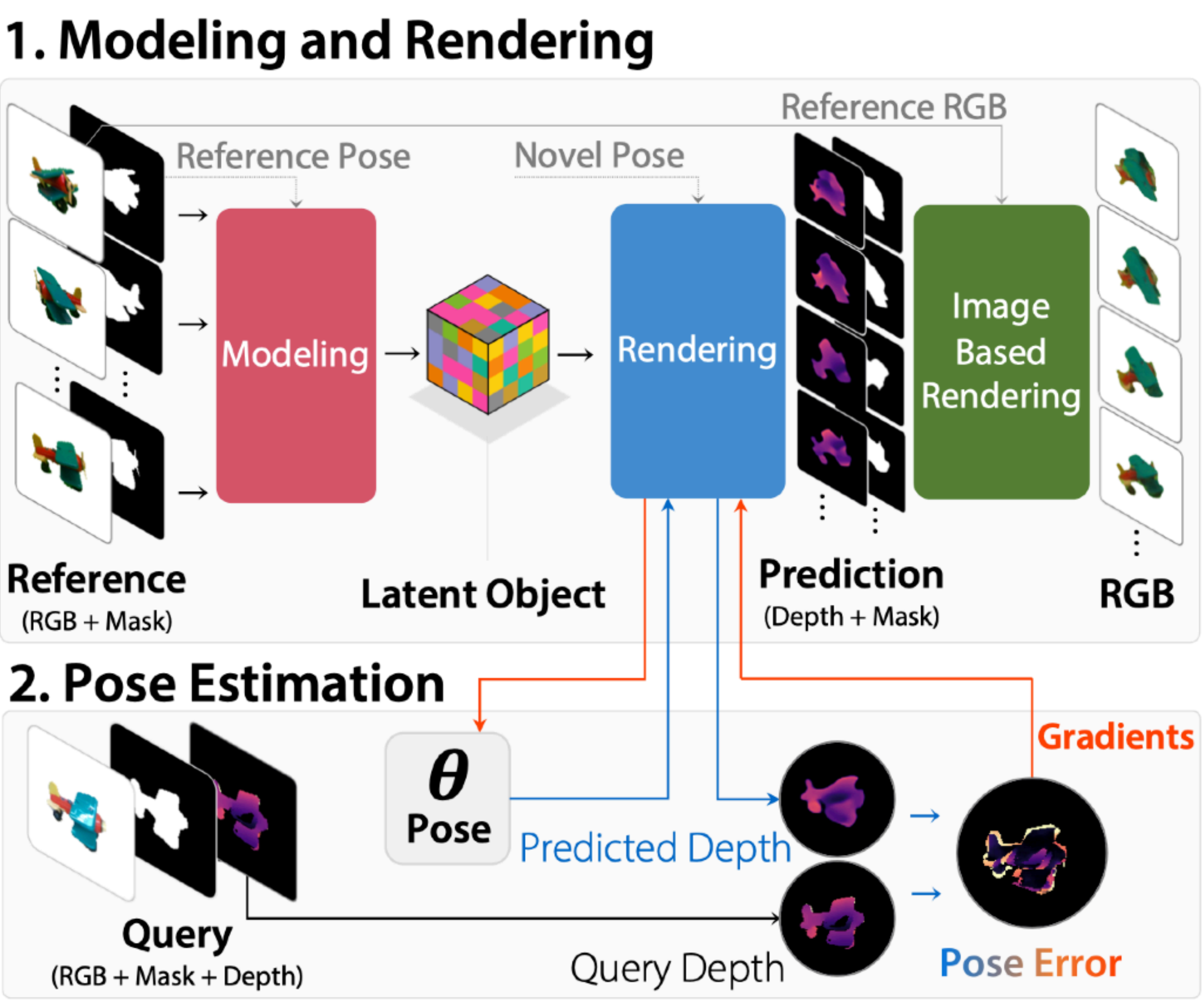
- Shape Prior Deformation for Categorical 6D Object Pose and Size Estimation

- Experiment on NOCS dataset

Data	Method	mAP					
		$3D_{50}$	$3D_{75}$	5° 2cm	5° 5cm	10° 2cm	10° 5cm
CAMERA25	Baseline [33]	83.9	69.5	32.3	40.9	48.2	64.6
	Ours (RGB)	93.1	**84.6**	50.2	54.5	70.4	78.6
	Ours (RGB-D)	**93.2**	83.1	**54.3**	**59.0**	**73.3**	**81.5**
REAL275	Baseline [33]	**78.0**	30.1	7.2	10.0	13.8	25.2
	Ours (RGB)	75.2	46.5	15.7	18.8	33.7	47.4
	Ours (RGB-D)	77.3	**53.2**	**19.3**	**21.4**	**43.2**	**54.1**

Model-Free Pose Estimation

- Latent Fusion
  - Reconstruct with reference frames
    - Estimate 2D feature, lift to 3D feature voxels
    - Feature aggregation across views
  - Inference with optimization
    - Estimate initial translation, sample rotation
    - Refine pose by optimization over depth error and latent error



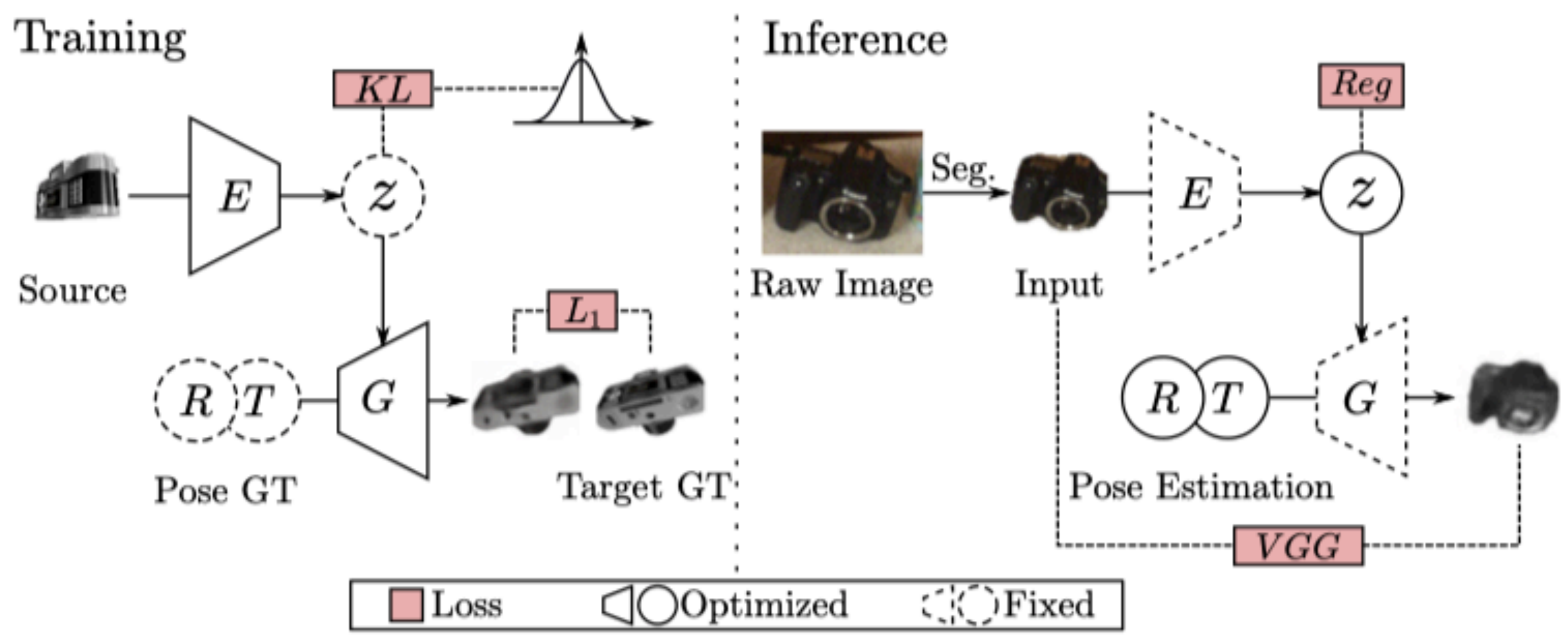
Model-Free Pose Estimation

- Latent Fusion
  - Experiment on MOPED dataset
  - Comparable results with model-based approach

	PoseRBPF [6]			IBR-LD		
Input	*Textured 3D Mesh*					
Training	*Yes*					
# Networks	*Per-Object*					
Pose Loss	-			$\mathcal{L}_{\text{latent}} + \mathcal{L}_{\text{depth}}$		
	ADD	**ADD-S**	**Proj.2D**	**ADD**	**ADD-S**	**Proj.2D**
black_drill	59.78	**82.94**	49.80	56.67	79.06	53.77
cheezit	57.78	82.45	48.47	**61.31**	**91.63**	**55.24**
duplo_dude	56.91	82.14	47.11	74.02	89.55	52.49
duster	**58.91**	82.78	**46.66**	49.13	91.56	19.33
graphics_card	59.13	83.20	49.85	**80.71**	**91.25**	**67.71**
orange_drill	**58.23**	82.68	**49.08**	51.84	70.95	46.12
pouch	57.74	82.16	49.01	**60.43**	**89.60**	**49.80**
remote	56.87	82.04	**48.06**	55.38	94.80	37.73
rinse_aid	57.74	82.53	**48.13**	65.63	92.58	28.61
toy_plane	**62.41**	85.10	49.81	60.18	**90.24**	**51.70**
vim_mug	**58.09**	**82.38**	**48.08**	30.11	80.76	14.38
mean	58.51	82.76	**48.55**	58.67	87.45	43.35

Model-Free Pose Estimation

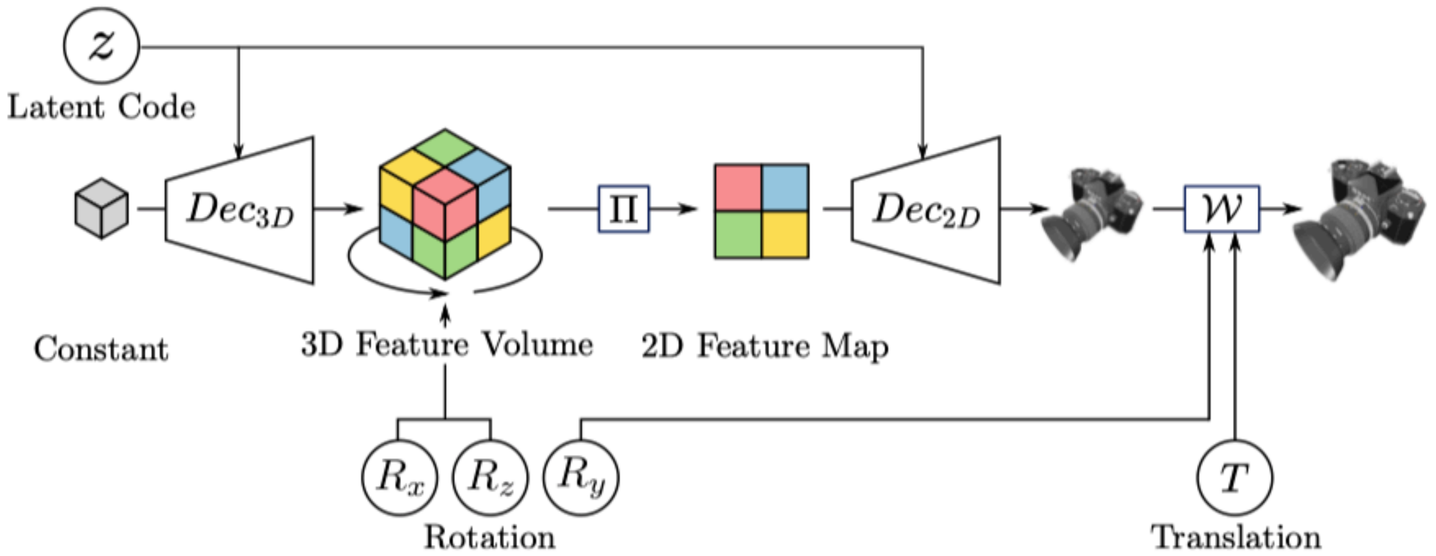
- Category Level Object Pose Estimation via Neural Analysis-by-Synthesis



- Approach:
    - Generate synthetic image given object pose and latent code
    - Compare feature metric error between synthesized and observed images to optimize object pose and object shape (latent code)

Model-Free Pose Estimation

- Category Level Object Pose Estimation via Neural Analysis-by-Synthesis



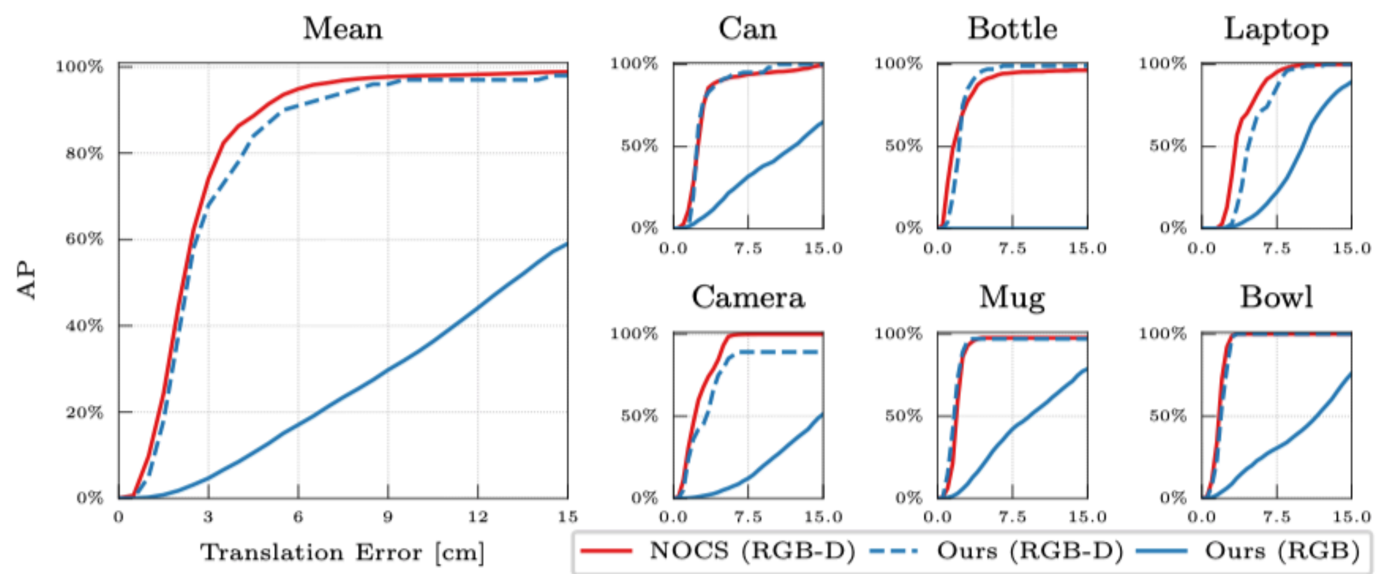
$$\mathcal{W}(T, R_z) : \begin{bmatrix} u \\ v \end{bmatrix} \mapsto \frac{f}{t_z} \cdot \left(R_z \begin{bmatrix} u \\ v \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix} \right)$$

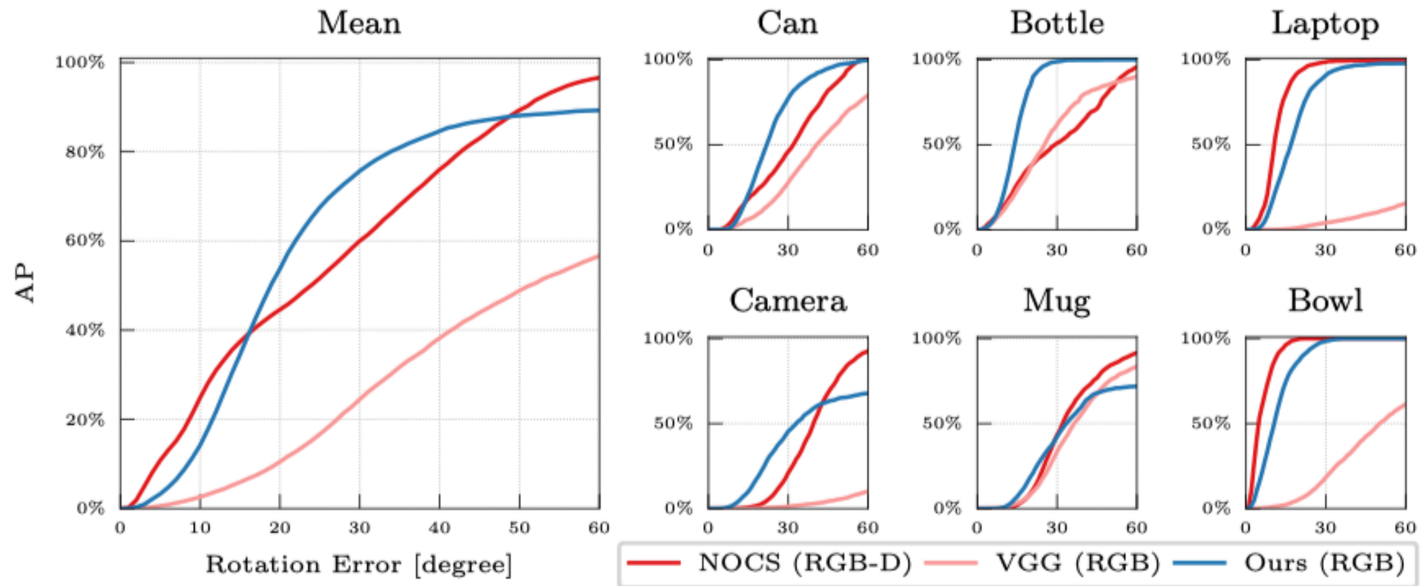
$$\hat{I} = G(R, T, z) = \mathcal{W}(T, R_z) \circ G_{3D}(R_x, R_y, z)$$

- Details:
  - Train generator using **synthetic data only**, use some tricks to reduce the need of network capacity
  - Initial pose are randomly sampled… $$E(I, R, T, z) = \|F_{vgg}(I) - F_{vgg}(\hat{I})\|_2 + \|z\|_2,$$
  - Optimize with deep feature (pre-trained VGG) + regularization on latent code

Model-Free Pose Estimation

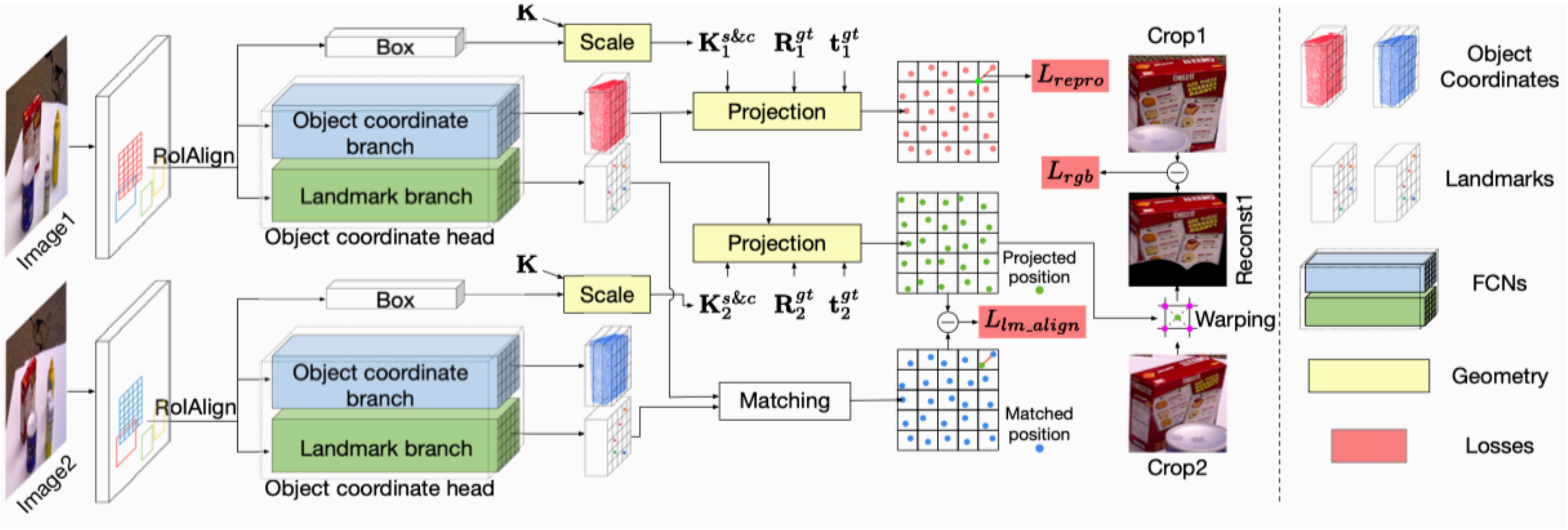
- Category Level Object Pose Estimation via Neural Analysis-by-Synthesis
  - Experiment on NOCS dataset
    - Comparable to NOCS when using RGB-D input
    - Does not require pose annotation
  - Compare with *LatentFusion*
    - Pros: No reference frames needed
    - Cons: can it extend to arbitrary unseen object?





Model-Free Pose Estimation

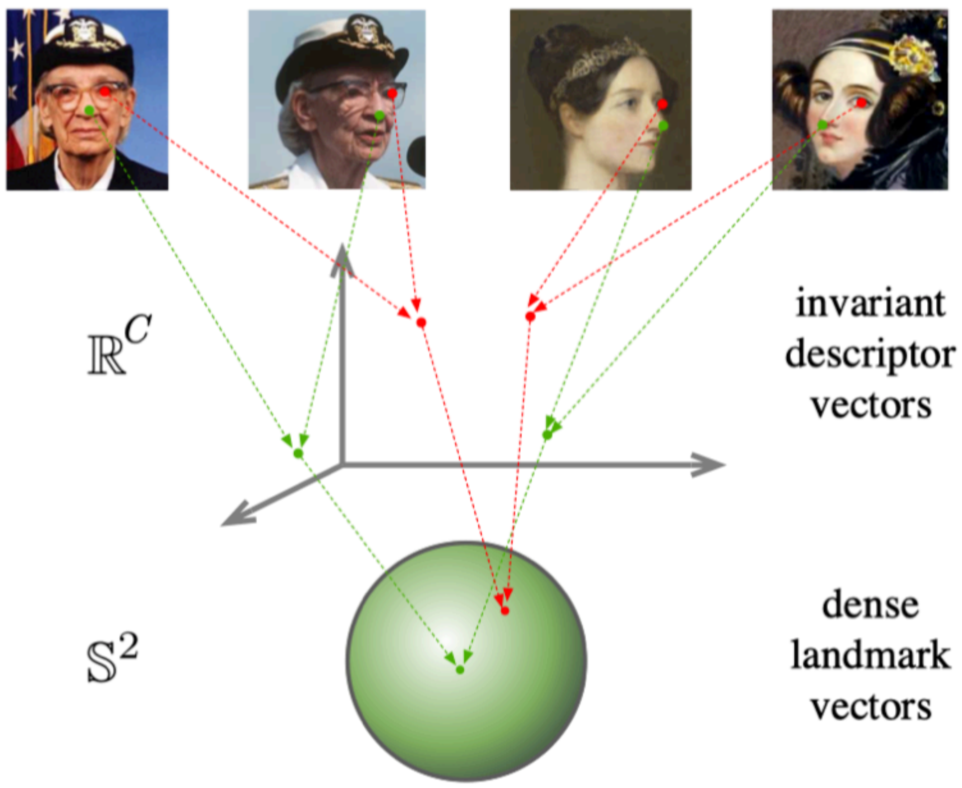
- Reconstruct Locally, Localize Globally



- Overall target: learn object coordinate estimation without CAD model
- Approach:
  - predict mask, object coordinates and landmark
  - Single-frame + cross frame consistency supervision

Model-Free Pose Estimation

- Reconstruct Locally, Locally Globally
  - What is a *landmark*?
    - Descriptor with fewer channels.
    - Descriptors that gain 'robustness' to intra-class variations
  - In this work: make no difference with using descriptor…
  - May be used to extend the work to category-level (?)



marks [45, 9, 42]. A dense descriptor associates to each image pixel a C-dimensional vector, whereas a dense landmark detector associates to each pixel a 2D vector, which is the index of the landmark in a (u, v) parameterisation of the object surface. Thus we can interpret a landmark as a tiny 2D descriptor. Due to its small dimensionality, a landmark loses the ability to encode instance-specific details of the appearance, but gains robustness to intra-class variations.

Model-Free Pose Estimation

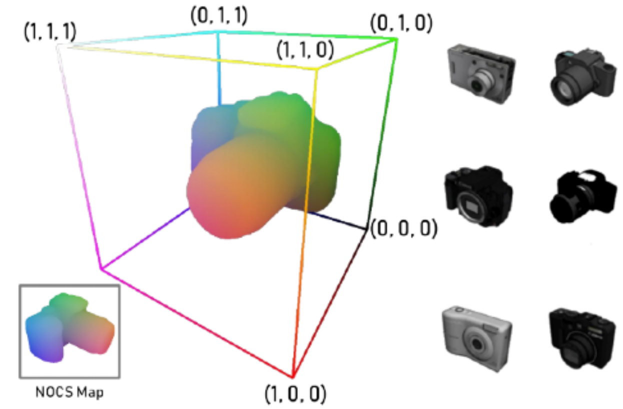
- Reconstruct Locally, Locally Globally
  - Results: Comparable with SoTA approaches

	w/ CAD model								w/o CAD model		
method	BB8 [41]	BB8 w/ r	SSD-6D w/ r [22]	Tekin [48]	DeepIM w/ r [27]	Dense-Fusion [52]	Pix2-Pose [38]	PVNet w/ r [40]	SSD-6D [22]	LieNet [11]	Ours
ape	27.9	40.4	65	21.62	77.0	92	58.1	43.62	0.00	38.8	52.91
benchwise	62.0	91.8	80	81.80	97.5	93	91.0	99.90	0.18	71.2	96.51
cam	40.1	55.7	78	36.57	93.5	94	60.0	86.86	0.41	52.5	87.84
can	48.1	64.1	86	68.80	96.5	93	84.4	95.47	1.35	86.1	86.81
cat	45.2	62.6	70	41.82	82.1	97	65.0	79.34	0.51	66.2	67.30
driller	58.6	74.4	73	63.51	95.0	87	76.3	96.43	2.58	82.3	88.70
duck	32.8	44.3	66	27.23	77.7	92	43.8	52.58	0.00	32.5	54.74
eggbox*	40.0	57.8	100	69.58	97.1	100	96.8	99.15	8.90	79.4	94.74
glue*	27.0	41.2	100	80.02	99.4	100	79.4	95.66	0.00	63.7	91.98
holepuncher	42.4	67.2	49	42.63	52.8	92	74.8	81.92	0.30	56.4	75.41
iron	67.0	84.7	78	74.97	98.3	97	83.4	98.88	8.86	65.1	94.59
lamp	39.9	76.5	73	71.11	97.5	95	82.0	99.33	8.20	89.4	96.64
phone	35.2	54.0	79	47.74	87.7	93	45.0	92.41	0.18	65.0	89.24
average	43.6	62.7	79	55.95	88.6	94	72.4	86.27	2.42	65.2	82.88

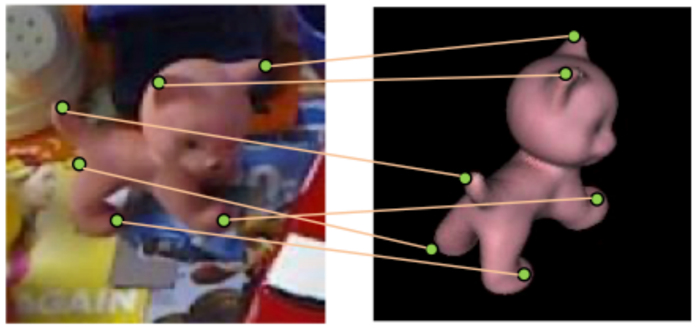
Table 2. **LineMOD: Percentages of correct pose estimates in ADD-10**. * denotes that the object is symmetric and is evaluated in ADD-S. w/r means the pose is refined with 3D model.

Discussion

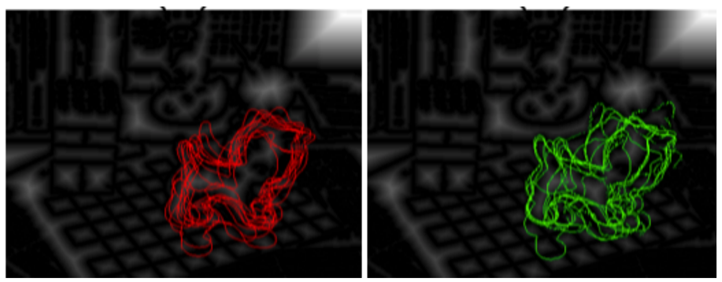
- Solved problem
  - Pose estimation with sparse keypoint set
  - Instance-level pose estimation
  - Given accurate CAD model and pose annotation
- Some recent trends
  - sparse representation to denser representation
  - instance-level to category level
  - model-based to model-free



Dense - Object Coordinates



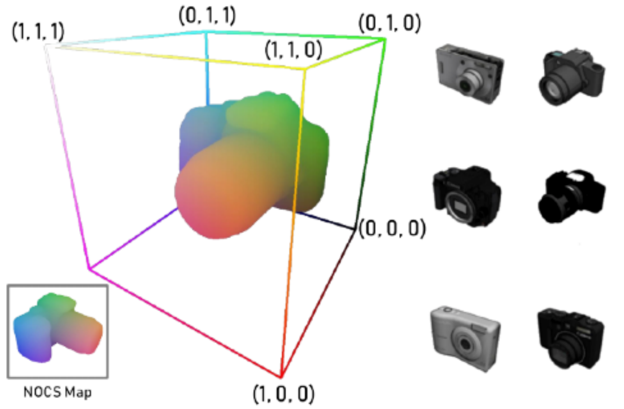
Sparse - Key Points



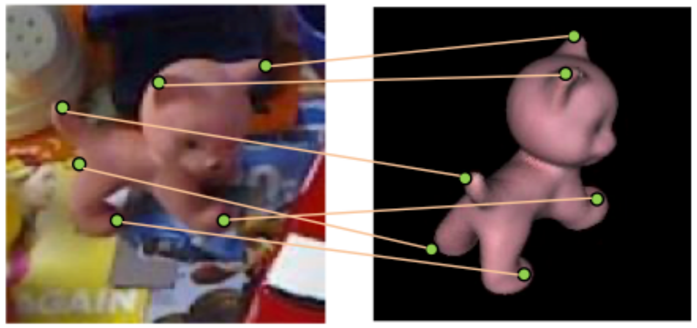
Semi-dense - Silhouette/Edges

Discussion

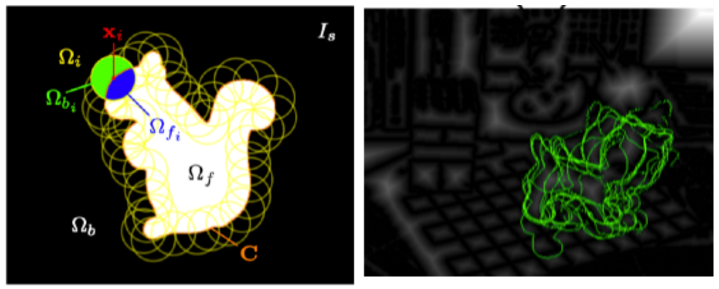
- Optimal representation of an object?
  - Preferred properties:
    - Available on weakly-textured object
    - Generalizable beyond instance-level
    - Available without accurate geometric models
    - <u>Trackable across time</u>
- Potential solution:
  - Sparse keypoint
  - Dense coordinate map
  - Silhouette + appearance cue
  - Latent representation
  - Hybrid of geometric/appearance primitives



Dense - Object Coordinates



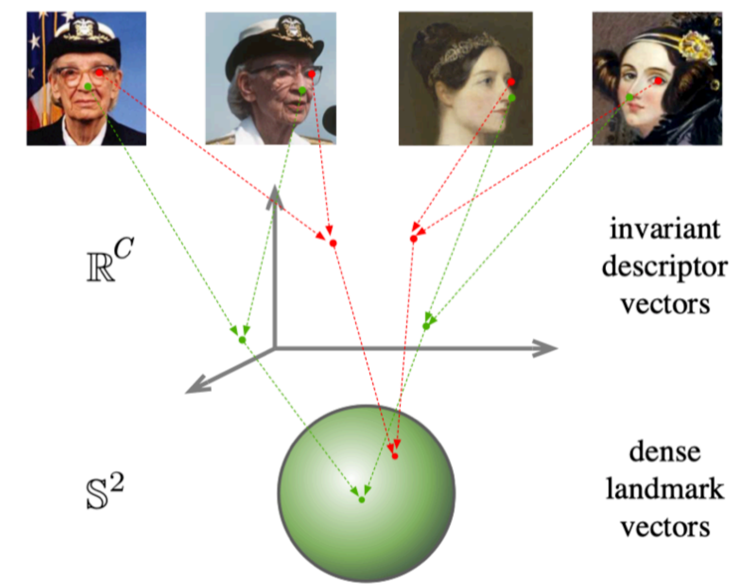
Sparse - Keypoints

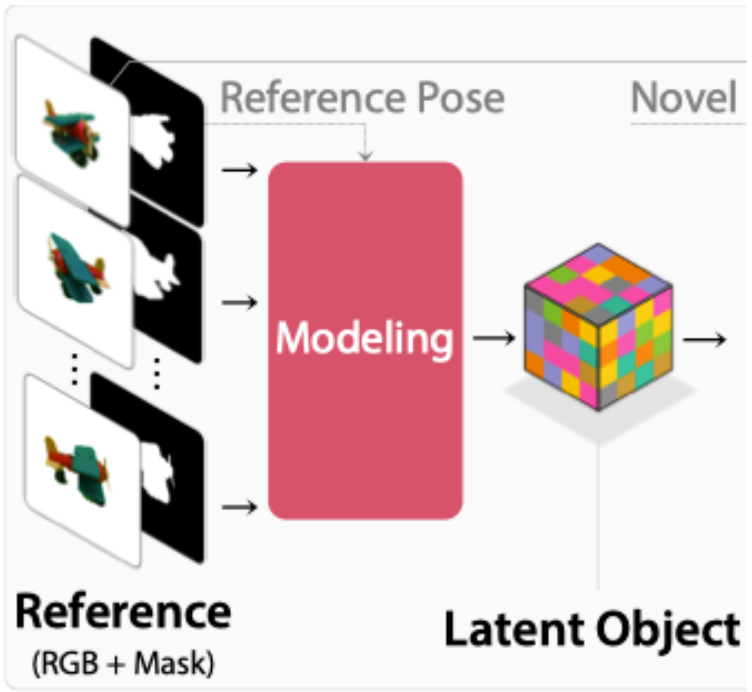


Hybrid - Silhouette + TCLC Hist.

Discussion

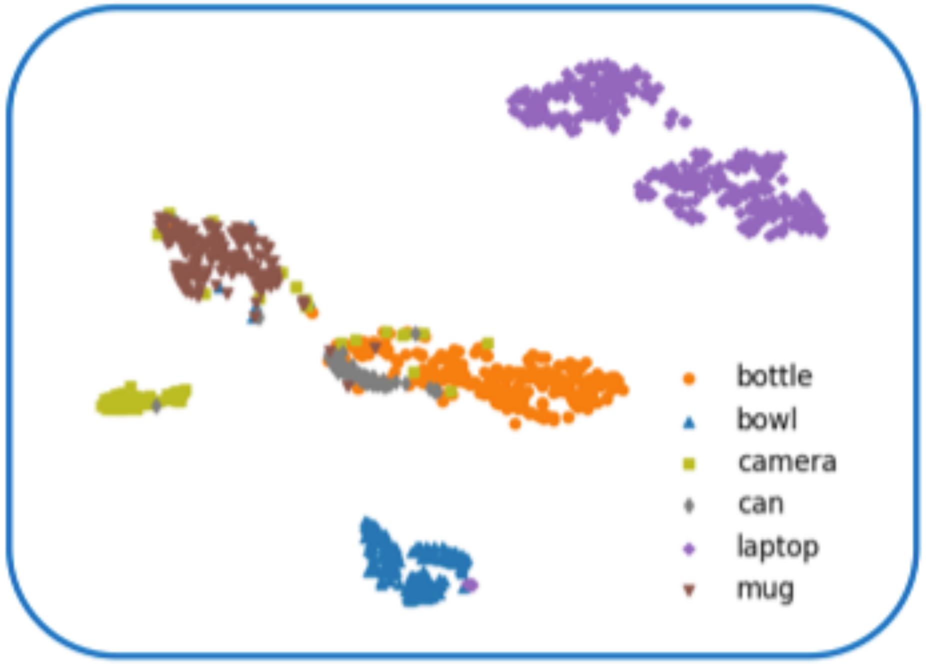
- Optimal representation of an object?
- <u>How to achieve model-free pose estimation?</u>
  - Learn to reconstruct geometry without accurate model
  - Neural synthesis to generate RGB image (and depth) for later optimization

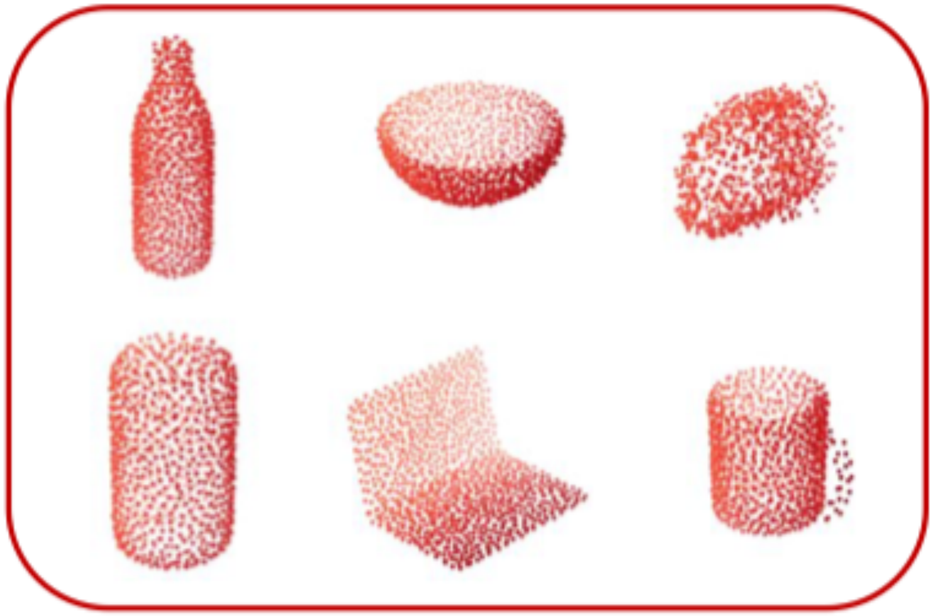




Discussion

- Optimal representation of an object?
- How to achieve model-free pose estimation?
- <u>How to achieve category-level pose estimation?</u>
  - With intra-category shape prior, either implicit (encoded in network) or explicit (mean shape)
  - Generalizable neural reconstruction





From academia to industry

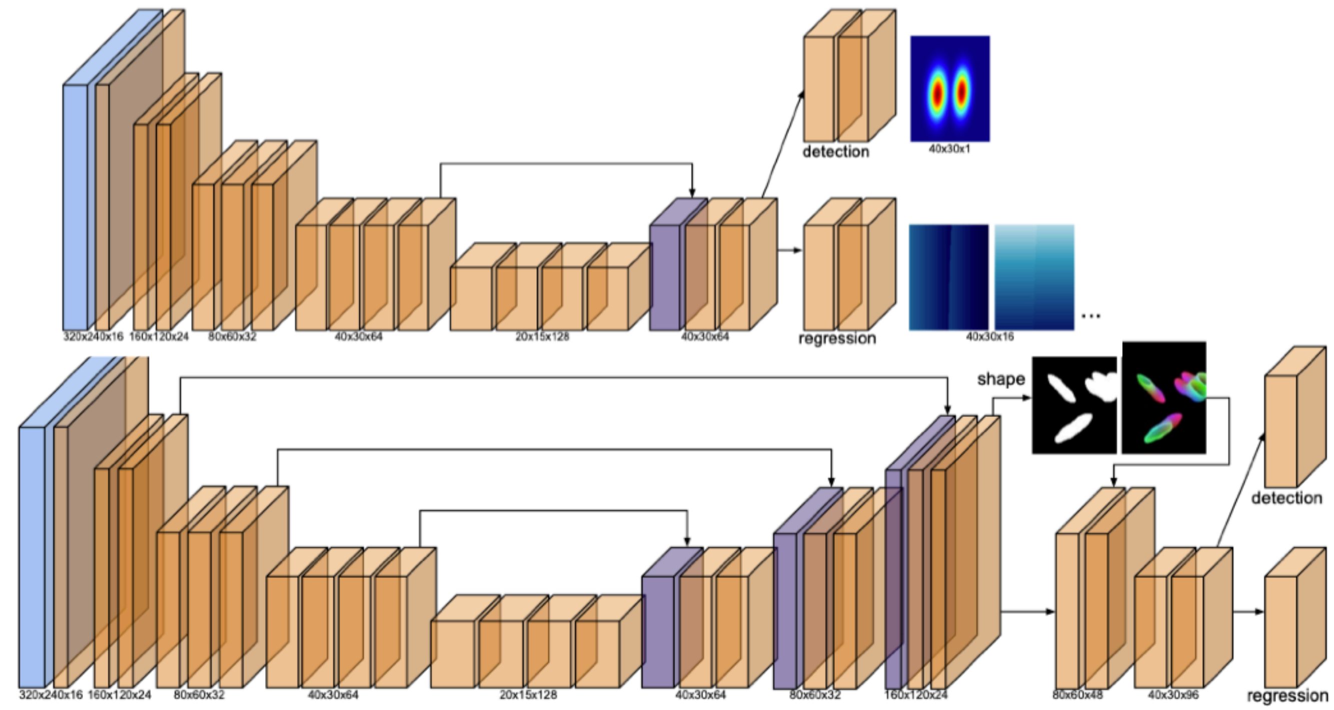
- Needs for 6DoF Pose Estimation in real application
  - Accuracy VS. Stability
  - Data VS. Algorithm
  - Scalability intra (or even inter) categories

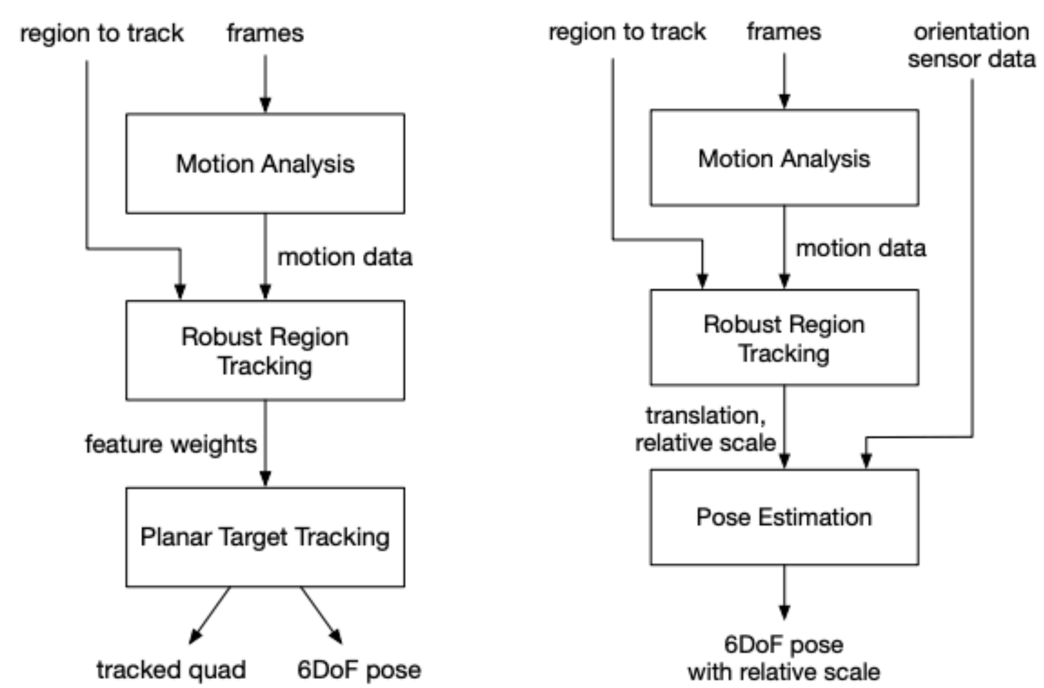
From academia to industry

- Hierarchy of problem to solve
  - Instance-level object 6DoF pose estimation in varied scenes and on varied sensor without fine-tune and adaptation
    - more of an engineering problem of scalable data collection
  - Category-level (with unseen instance) object 6DoF pose estimation on some common categories
    - worth working as academic problem
  - Estimate the pose of arbitrary unseen object
    - zero-shot learning, final target.. but hard if not impossible

The Solution by Google Media Pipe

- MobilePose
  - Approach
    - Joint 2D detection and regression (for 3D box corners and center)
    - Estimate mask and object coordinate if available to augment feature
  - Data pipeline
    - Hand-annotated for the first frame
    - Propagate along the sequence using camera pose from ARCore
- Instant motion tracking
  - Track the 9 keypoints by motion analysis





(a) Planar target tracking (b) 6DoF tracking

Thanks for your Attention

Siyu ZHANG

Research Engineer

ZJU-SenseTime Joint Lab of 3D Vision

zhangsiyu1@sensetime.com