Object 6D Pose Estimation by Action-Decision

Siyu ZHANG

Research Engineer

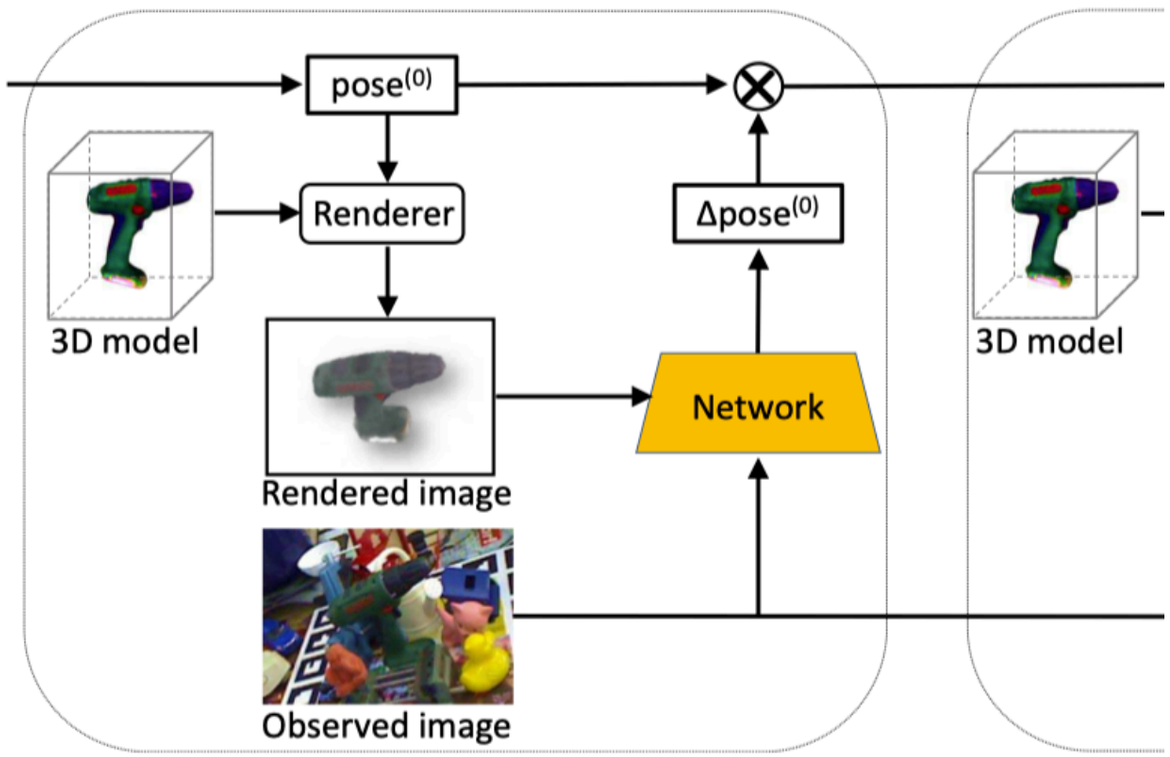
ZJU-SenseTime Joint Lab of 3D Vision

A bit of Recap of Objet 6D Pose Estimation

- 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

A bit of Recap of Objet 6D Pose Estimation

- As a **regression** problem
  - Pose Tracking: render and regression
    - Render the image of target object
    - Feed the rendered image and input image into network
    - Regress the additive pose for target object



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

A bit of Recap of Objet 6D Pose Estimation

- As a **regression** problem
  - Pose Estimation: direct regression
  - Pose Tracking: render and regression

- Are there possible improvements?

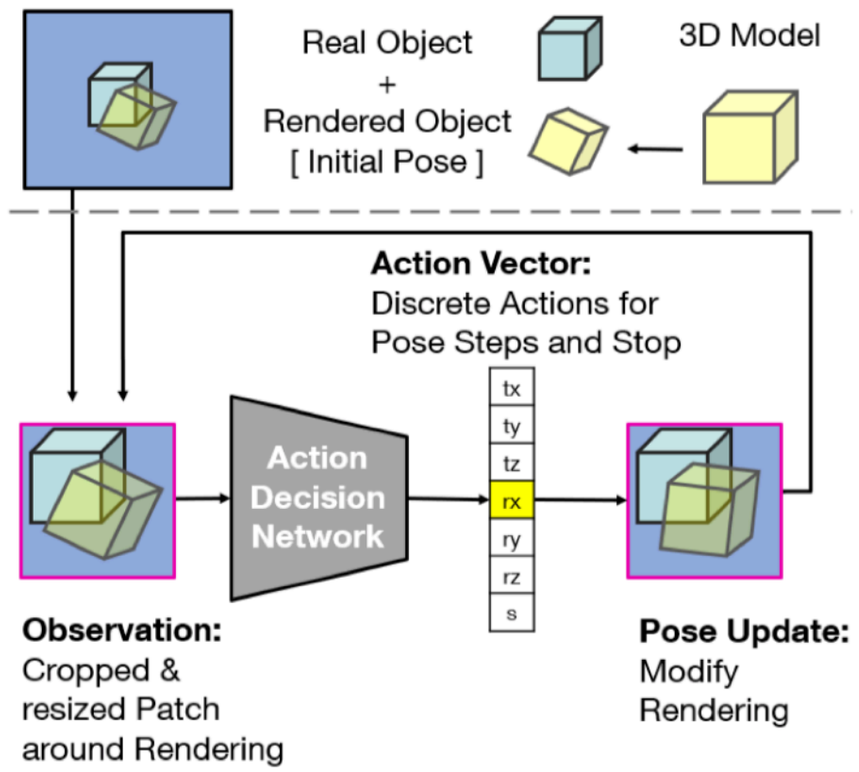
Content

- Paper 1: *I Like to Move it 6D Pose Estimation as an Action Decision Process* - model object pose refinement as **discrete decision making** process

- Paper 2: *Pose-Free Reinforcement Learning for 6D Pose Estimation* - model object pose refinement as as **reinforcement learning** problem

6D Pose as Action Decision Problem

- Methodology
  - Input:
    - Cropped image of real object + rendered object with initial pose
    - Concat with rendered depth and mask
  - Output: 13 DoF action
    - Action: +tx, -tx, +rx, -rx, …, stop
    - Step size is fixed
  - Initialization: Random seed and vote for object center (detailed next page)

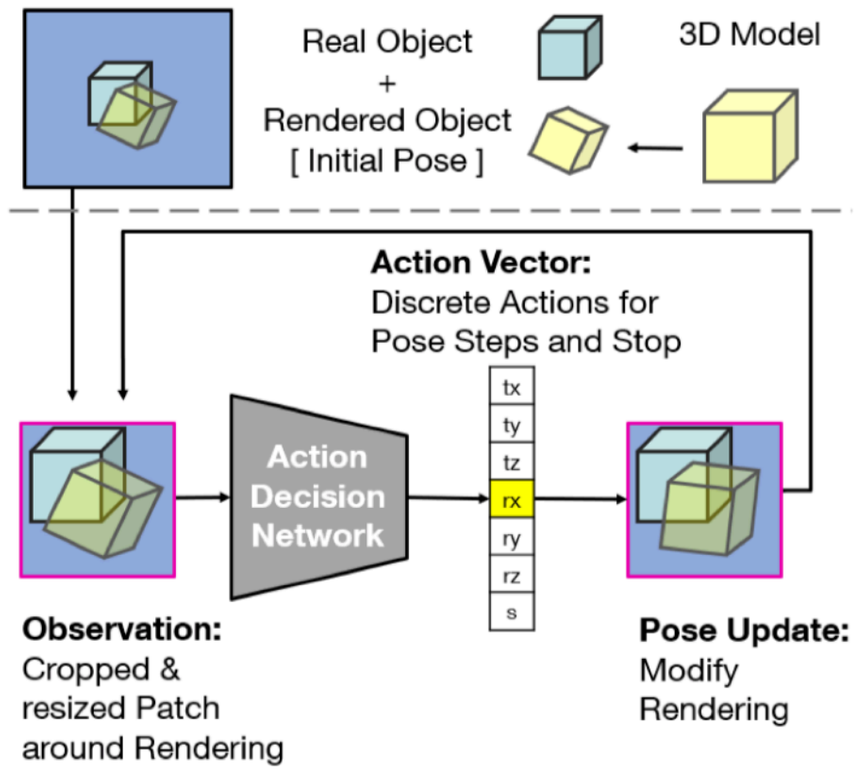


6D Pose as Action Decision Problem

- Initialization by voting
  - Observation:
    - Will usually **translate** then **rotate**
    - Action can still converge even with large offset.
  - Method:
    - Random sample seeds
    - Vote for object center by aggregate actions.

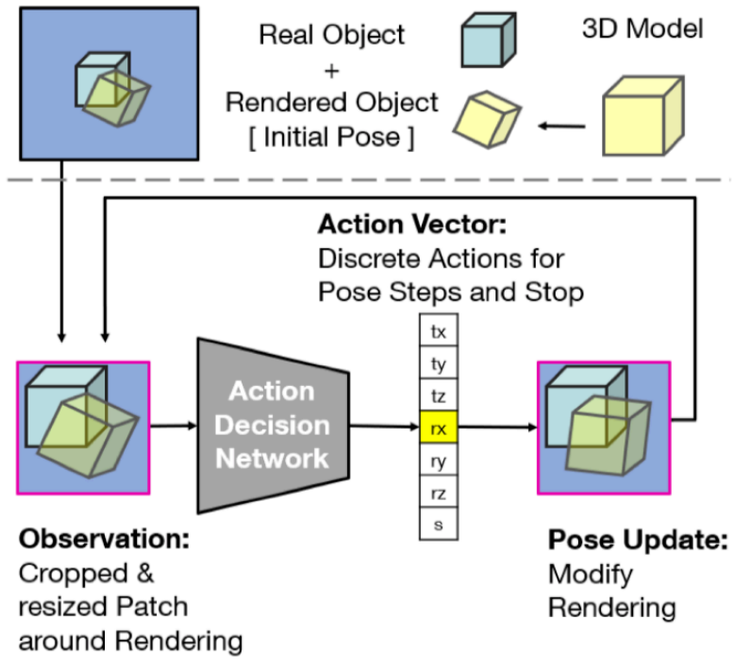
6D Pose as Action Decision Problem

- Methodology
  - Input: RGB image + rendered image
  - Output: 13 DoF action
  - Initialization: Random seed and vote for object center



6D Pose as Action Decision Problem

- Difference with existing approach:
  - Discrete action with fixed size
- Intuition: Why this is better?
  - Generalization ability
    - Wider converge basin
    - Lighter network + simpler task
    - Synthetic training



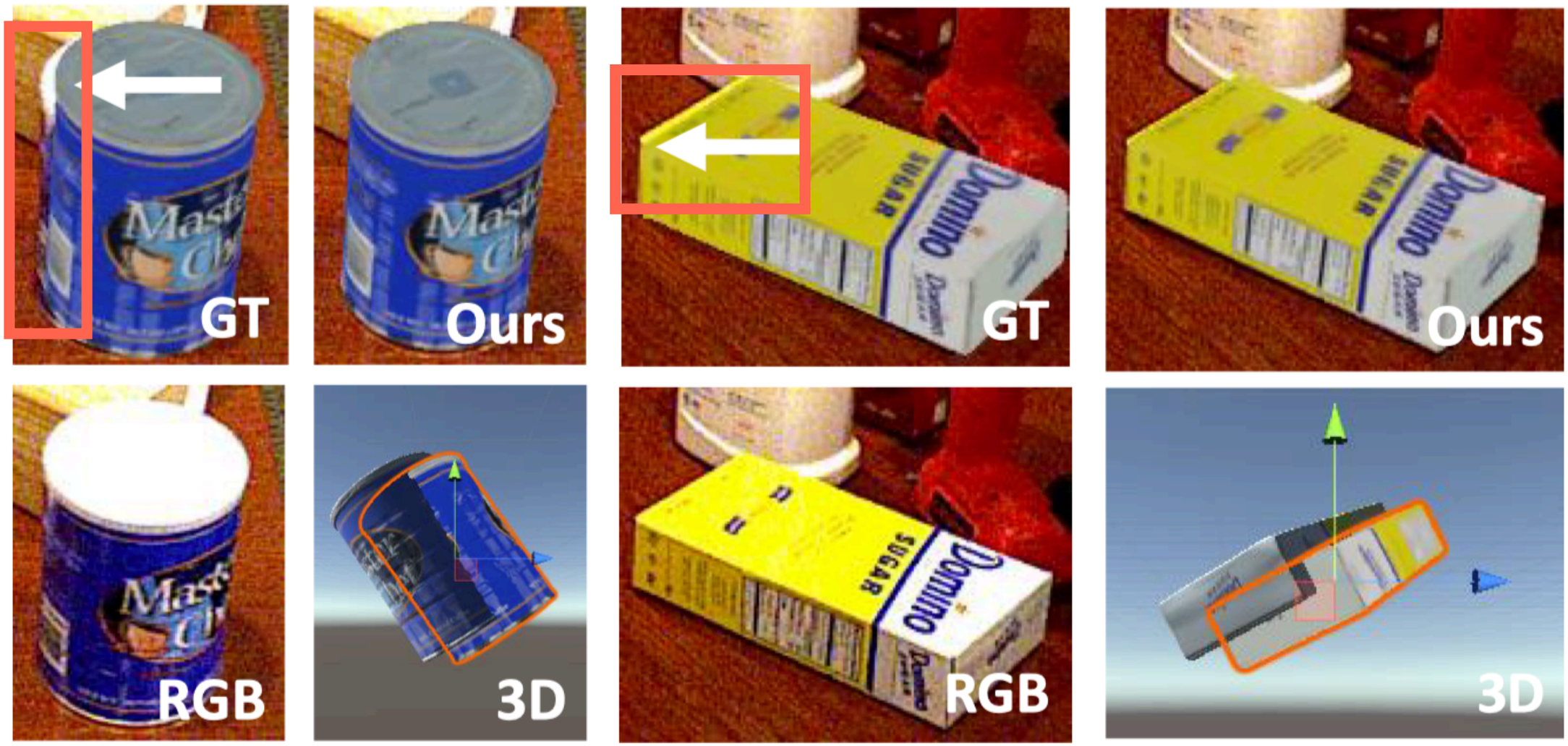
6D Pose as Action Decision Problem

- Experiment
  - Dataset: YCB-Video, LAVAL
  - **Trained only on YCB** and eval on both
  - While analysis, they **amazingly** found that:

YCB which we further analyzed. It turned out that the annotations for some of the objects are slightly shifted as shown in Fig. 5. Our method – in contrast to others with which we compare in Tab. 1 – is fully trained on synthetic data. Thus, we cannot learn an annotation offset during training time due to the fact that our training setup provides pixel-perfect ground truth. Further investigations

6D Pose as Action Decision Problem

- Problem: GT is wrong… (all methods before are trying to overfit on false GT)



6D Pose as Action Decision Problem

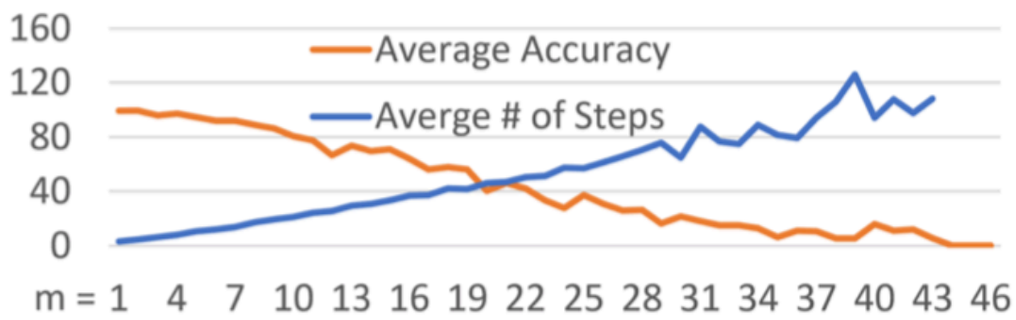
- Experiment
  - Dataset: YCB-Video, LAVAL
  - **Trained only on YCB** and eval on both.
  - Evaluation on YCB (single-object model) - **surpass SoTA method**

SD [30]	HM [53]	Ours OS	Ours + Shift	PRBPF [13]	Ours OS	+ Shift
33.00	**75.80**	7.70	91.88	63.30	65.61	91.15
46.60	86.20	**88.36**	97.76	77.80	84.34	90.74
75.60	67.70	58.35	91.95	79.60	78.43	91.05
40.80	38.10	38.23	57.99	73.00	66.83	76.06
70.60	**95.20**	87.74	98.49	84.70	86.05	94.03
18.10	5.83	**47.90**	52.89	64.20	65.90	69.12
12.20	**82.20**	58.68	76.00	64.50	79.00	83.01
59.40	**87.80**	37.08	89.20	83.00	82.92	92.78
33.30	**46.50**	45.99	60.61	51.80	75.21	79.44
16.60	30.80	**74.02**	90.43	18.40	84.99	90.19
90.00	57.90	**99.40**	100.00	63.70	85.14	94.22
70.90	73.30	**95.04**	95.30	60.50	89.27	90.68
30.50	36.90	**99.44**	99.44	28.40	85.89	87.03
40.70	17.50	**45.35**	76.59	77.90	78.95	87.83
63.50	**78.80**	52.77	97.35	71.80	76.56	91.95
27.70	33.90	**52.28**	63.48	2.30	48.62	53.52
17.10	43.10	**63.33**	81.11	38.70	79.78	83.99
4.80	8.88	**39.53**	41.73	67.10	73.27	75.31
25.60	50.10	**64.01**	82.83	38.30	56.09	65.97
8.80	32.50	**88.02**	91.37	32.30	67.31	78.06
34.70	66.30	**80.83**	80.83	84.10	86.52	86.70
39.07	53.11	**63.05**	81.75	58.35	76.03	83.47

6D Pose as Action Decision Problem

- Robustness & Convergence Analysis
  - Can still converge even when the initial pose is unreasonably bad.
    - make it possible to initialize without other approaches.
  - While previous methods (e.g. Deep 6DoF tracking) failed when overlapping is less than 50%

results are summarized in Fig. 6. Note that even for a large deviation of $m = 12$ which is significantly larger then the deviation found in the video sequence, our accuracy is ADD = 73.8%. Moreover, we can also see reasonable convergence in cases with 50% or fewer bounding box overlap where other methods [21] struggle and drift.



6D Pose as Action Decision Problem

- Experiment: Evaluation on LAVAL
  - Not good enough in rotation

	Ours full				Ours w/o D			
Occlusion	0%	15%	30%	45%	0%	15%	30%	45%
Clock								
T[mm]	14.02	20.54	25.85	51.92	9.39	9.96	32.58	15.91
R[deg]	9.40	10.84	12.74	17.05	29.15	27.92	30.72	28.40
Cookie Jar								
T[mm]	3.82	5.99	9.52	15.18	1.79	2.75	11.62	5.95
R[deg]	6.48	17.82	18.22	15.89	28.77	18.18	24.30	19.02
Dog								
T[mm]	12.09	28.37	55.48	77.91	6.10	10.76	33.89	15.62
R[deg]	11.70	14.21	22.43	23.80	20.75	26.81	24.22	22.53
Dragon								
T[mm]	22.47	29.39	36.37	40.06	25.69	25.13	27.71	30.65
R[deg]	3.34	4.89	11.65	13.39	27.16	36.40	37.61	30.94
Shoe								
T[mm]	9.72	17.91	24.33	37.34	44.61	19.90	38.04	41.90
R[deg]	5.84	9.26	17.89	16.91	62.78	39.47	43.50	24.73
Watering Can								
T[mm]	14.67	21.66	18.68	33.26	11.61	20.54	20.96	26.10
R[deg]	11.89	19.80	23.43	33.54	38.89	40.85	36.30	35.23

6D Pose as Action Decision Problem

- Experiment: Runtime

and the number of actions. In our current implementation, the runtime for one loop in the cycle breaks down in the image preprocessing done on CPU and the inference on the GPU. We performed a runtime test averaging 512 iterations. The results are shown in Tab. 3.

Average Runtime on	CPU	GPU	Total
Average Runtime in ms	14.6	5.2	19.8

Table 3: Average Runtime of Action Decision Process Cycle.

Given the average of 4.2 actions on our YCB tests, we report an overall average runtime of 83.16 ms or 12 FPS. Note that the runtime could be increased if the image processing was also ported to the GPU.

6D Pose as Reinforcement Learning Problem

- Quick recap of RL (considering only *control* problem)
  - Terminologies:
    - State: information about the world
    - Action: action to trigger next state from current state, sampled from ***policy***
    - Reward: how good is current action.
  - Target: get ***policy*** that would optimize ***value function*** (expected cumulated ***reward*** for all times)

Discussion: Compare (D)RL and Supervised Learning

- Similarities:
  - Target: get some output from network that would produce maximally possible performance
  - Method: optimize **network parameter** w.r.t. **performance measurement**

- **For Supervised Learning**
  - **Performance measurement** comes from a **differentiable Loss function of network output**
  - Supervision is **dense**

- **For Reinforcement Learning**
  - Performance measurement does **not necessarily relate to network output** (for example, it may comes from the environment)
  - Supervision is **sparse** and **temporal correlated**

Discussion: Compare (D)RL and Supervised Learning

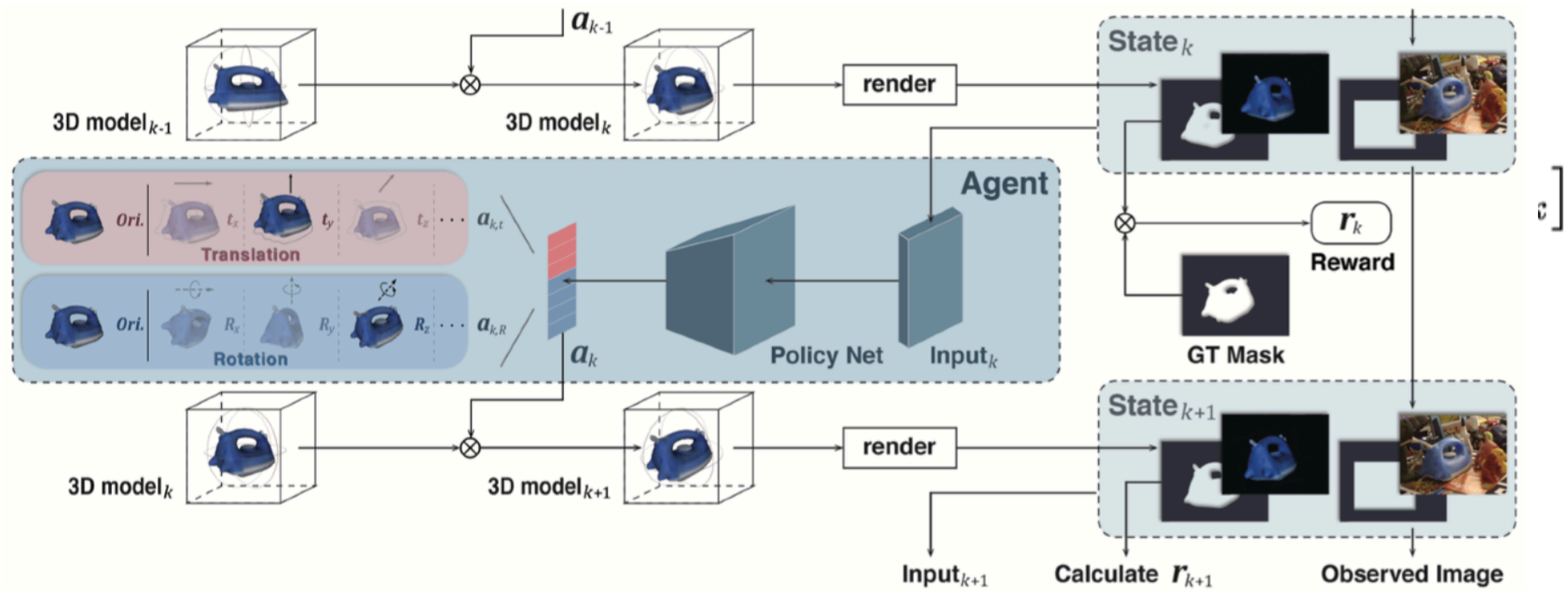
- Use DRL instead of Supervised Learning when …
  - Loss some part of the network is non- differentiable
  - Supervision is sparse
  - Task is temporal correlated (e.g. path planning)

6D Pose as Reinforcement Learning Problem

- Quick recap of RL (considering only *control* problem)
  - Terminologies:
    - State: information about the world
    - Action: action to trigger next state from current state, sampled from ***policy***
    - Reward: how good is current action.
  - Target: get ***policy*** that would optimize ***value function*** (expected cumulated ***reward*** for all times)

- **<u>Why use RL instead of Supervised Learning?</u>**
  - Using 2D mask as sparse supervision

6D Pose as Reinforcement Learning Problem



- Problem formulation
  - Maximize future discounted rewards: $V^{\pi}(\mathbf{s}) = \mathbb{E}\left[\sum_{k \geq 0} \gamma^k r_k\right]$

6D Pose as Reinforcement Learning Problem

- Reward: 2D Mask-based reward
  - IoU Difference Reward: Encourage overlapping of 2D mask

$$r_I = f_\phi(\mathrm{IoU}_{k+1}) - f_\phi(\mathrm{IoU}_k), \qquad f_\phi(x) = \begin{cases} x, & x < X_{\text{thr}} \\ \alpha x^2 - \beta x, & x \geq X_{\text{thr}} \end{cases}$$

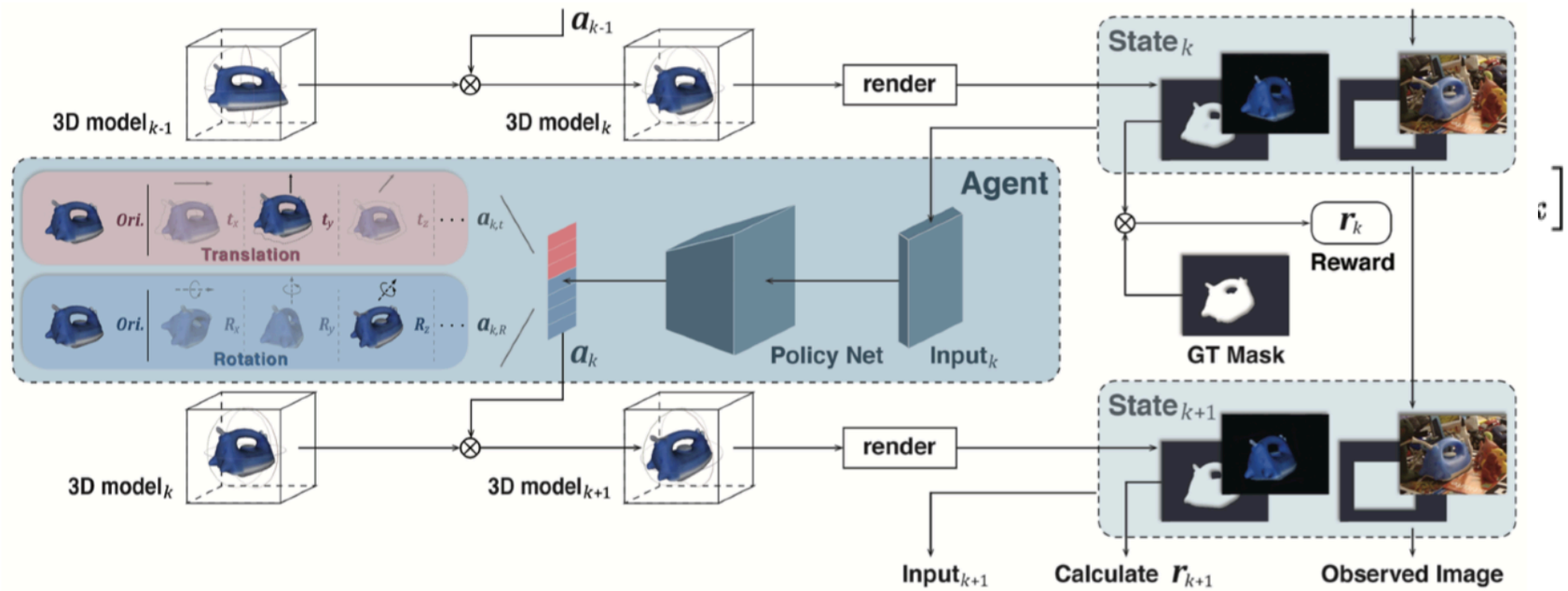
  - Goal Reached Reward: Stop refining when reach IoU_thr

$$r_G = \begin{cases} 1, & \mathrm{IoU}_k \geq \mathrm{IoU}_{\text{thr}} \\ 0, & \mathrm{IoU}_k < \mathrm{IoU}_{\text{thr}} \end{cases}$$

- Ceneralization Reward: Bootstrap the network

$$r_C = \min(||c_r - c_g||_2^{-\frac{1}{2}}, 1)$$

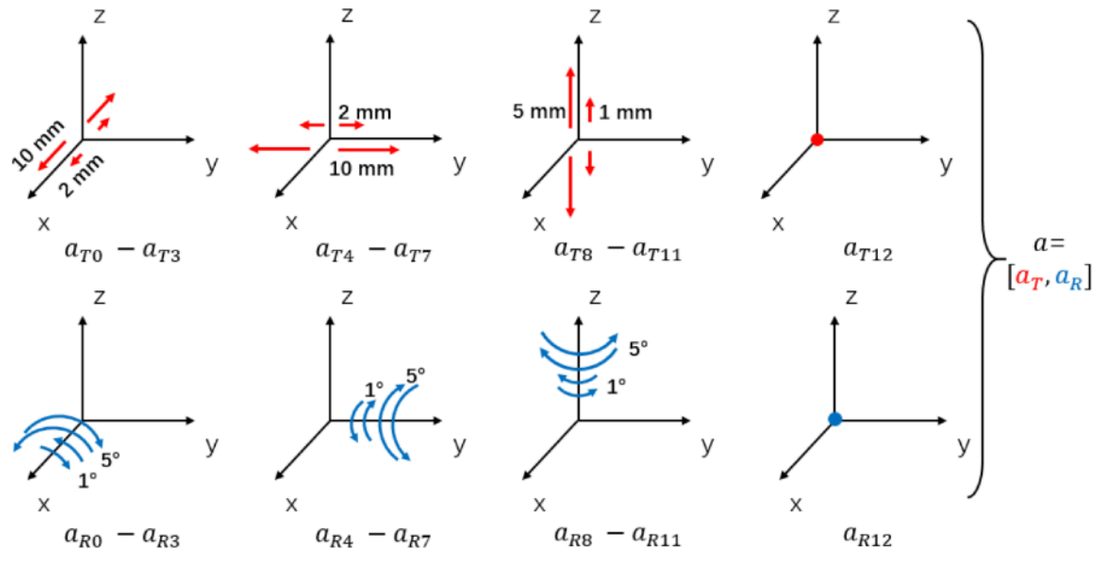
6D Pose as Reinforcement Learning Problem



- Problem formulation
  - Maximize future discounted rewards: $V^{\pi}(\mathbf{s}) = \mathbb{E}\left[\sum_{k \geq 0} \gamma^k r_k\right]$
  - **State**: rendered RGB image, projection mask, observed RGB image, gt-2D box
  - **Action**: discrete & hand-craft action

6D Pose as Reinforcement Learning Problem

- **Action**:
  - Discrete hand-craft action
  - Proved to be better than continuous action.



Action Style / Metric	Continuous Proj. 2D	Continuous ADD	Discrete Proj. 2D	Discrete ADD
Initial	14.2	35.6	**14.2**	**35.6**
Epoch: 800	37.6	43.8	**58.9**	**52.8**
Epoch:1600	45.7	46.7	**63.3**	**55.4**
Epoch:2400	57.5	56.2	**75.3**	**70.7**
Epoch:3200	60.4	60.9	**80.6**	**79.2**

Table 3. Ablation on action style.

6D Pose as Reinforcement Learning Problem

- Composite Reinforced Optimization
  - Policy-gradient optimization based on PPO
  - Off-policy optimization based with replay buffer

6D Pose as Reinforcement Learning Problem

- Experiment: Evaluation on Linemod & T-LESS

Train data	Pose-Free Init		+Pose-Free Refine			Gt Pose Init		+Pose-Free Refine			
Object	AAE[33] ADD	DPOD-SYN[40] ADD	SSD6D[12] ADD	DPOD-SYN+Refine ADD	AAE+ours ADD	YOLO6D[36] ADD	PoseCNN[38] ADD	PoseCNN+DeepIM-SYN[15] Proj.2D	PoseCNN+DeepIM-SYN[15] ADD	PoseCNN+ours Proj.2D	PoseCNN+ours ADD
Ape	3.96	37.22	-	55.23	**65.4**	21.6	27.8	81.7	23.9	**95.6**	**60.5**
Benchvise	20.92	66.76	-	72.69	**84.5**	81.8	68.9	92.2	**93.1**	**94.7**	88.9
Camera	30.47	24.22	-	34.76	**41.5**	36.6	47.5	**97.0**	**84.7**	95.0	64.6
Can	35.87	52.57	-	**83.59**	80.9	68.8	71.4	89.6	**91.5**	**93.1**	91.3
Cat	17.90	32.36	-	65.10	**80.4**	41.8	56.7	96.1	79.5	**99.3**	**82.9**
Driller	23.99	66.60	-	73.32	**77.6**	63.5	65.4	85.9	82.3	**94.8**	**92.0**
Duck	4.86	26.12	-	50.04	**52.5**	27.2	42.8	92.6	24.0	**98.2**	**55.2**
Eggbox	81.01	73.35	-	89.05	**96.1**	69.6	98.3	90.8	88.3	**97.8**	**99.4**
Glue	45.49	74.96	-	**84.37**	76.7	80.0	95.6	81.2	**96.9**	**97.1**	93.3
Holepuncher	17.60	24.50	-	35.35	**44.9**	42.6	50.9	78.0	20.6	**96.7**	**66.7**
Iron	32.03	85.02	-	**98.78**	67.3	75.0	65.6	59.3	**85.1**	**81.6**	75.8
Lamp	60.47	57.26	-	74.27	**91.1**	71.1	70.3	75.6	85.5	**96.0**	**96.6**
Phone	33.79	29.08	-	46.98	**52.7**	47.7	54.6	88.3	66.1	**91.0**	**69.1**
Mean	31.41	50.0	34.1	66.43	**70.1**	56.0	62.7	85.3	70.9	**94.7**	**79.7**

Method	AAE[33]		AAE+Ours	
Metric	Proj. 2D	VSD	Proj. 2D	VSD
19	30.65	49.95	32.79	57.39
20	23.51	41.87	25.40	45.29
21	56.55	59.06	60.58	62.50
22	42.99	46.08	44.78	48.02
23	21.88	40.38	29.32	44.44
Mean	35.12	47.47	**38.57**	**51.53**

Metric	Recall scores (%) on ADD						
Class	ape	bv.	cam	can	cat	driller	duck
DPOD-refine	52.12	64.67	22.23	77.51	56.49	65.23	49.04
PFRL	69.26	78.68	27.77	77.16	64.52	79.90	48.24
Class	egg.	glue	hol.	iron	lamp	phone	**Mean**
DPOD-refine	62.21	38.94	25.55	98.43	58.35	33.79	54.20
PFRL	67.68	37.73	27.87	88.00	73.67	37.51	**59.85**

Table 2. DPOD-refine/PFRL with DPOD init.

6D Pose as Reinforcement Learning Problem

- Experiment: Runtime

Ep Length		Ape		Benchvise		Can	
		A	P	A	P	A	P
Acc (%)	0	39.0	83.1	14.2	50.0	16.8	69.6
	5	87.0	92.0	48.2	78.4	59.8	84.8
	10	93.0	93.4	66.1	83.9	71.6	87.5
	20	95.2	94.0	78.4	88.9	77.7	90.0
	50	96.9	94.2	85.1	92.7	82.6	91.1
Time (ms)	5	62	64	67	65	68	68
	10	110	114	126	121	128	127
	20	218	220	243	234	245	243
	50	529	534	588	578	588	584

Table 4. Accuracy and testing time with respect to episode length. 'A' and 'P' denote initial poses from AAE [33] and PoseCNN [38], respectively.

Take-Home Message

- When network is not working, turn to the training & val data at first (your algorithm may really outperform ground truth!)

- When dealing with limited network capacity, learning less is more

- Consider using RL when supervision is sparse and temporal correlated.

Thanks for your Attention

Siyu ZHANG

Research Engineer

ZJU-SenseTime Joint Lab of 3D Vision

zhangsiyu1@sensetime.com