UC San Diego

Learning Deep Visuomotor Policies for Dexterous Hand Manipulation

Presenter: Shuo Cheng
May 26, 2020

Outline

- Introduction
- Related works
- Method
- Experiments
- Conclusion

Task intro

Visuomotor Policy: control the movement to achieve some goals based on visual feedbacks.

Why learn visuomotor policy?

Task intro

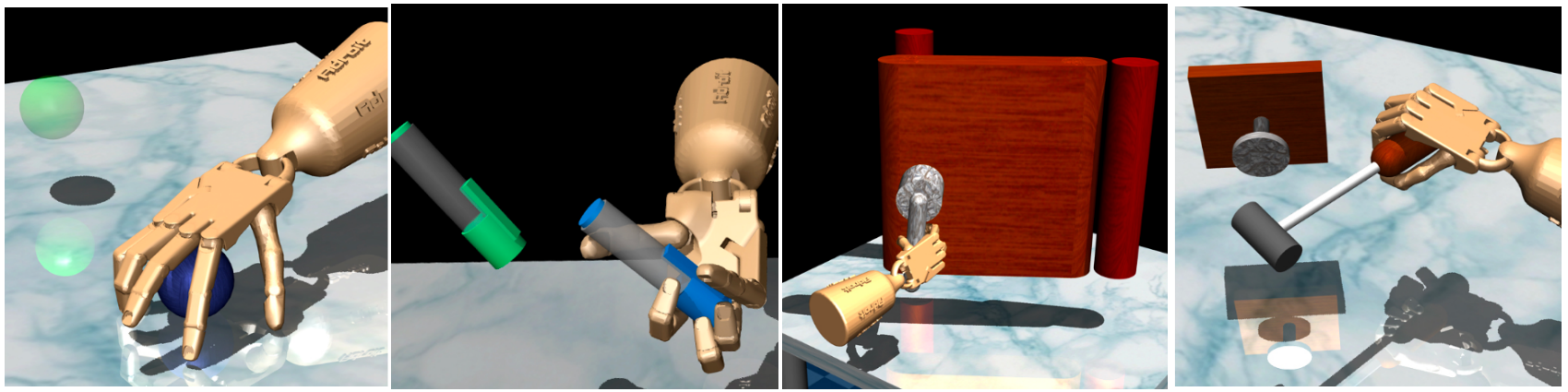
Why learn visuomotor policy?

- Designing a manageable and compact representation for the output of the vision system is non-trivial;
- Vision system can be complex and prone to errors, learning perception and control together allows the whole system to adapt to the goal of the task.

End-to-End Training of Deep Visuomotor Policies, Levine et al, JMLR 16

Task intro

Dexterous Hand Manipulation: using multi-fingered hand to perform skills like grasping and relocating objects, in-hand manipulation (e.g., rotating object), and tool usage.



Related works

For robot manipulation:

- Classical approaches
- Reinforcement learning (short as RL)
- Imitation learning (short as IL)

(The last two mainly focused on the control part …)

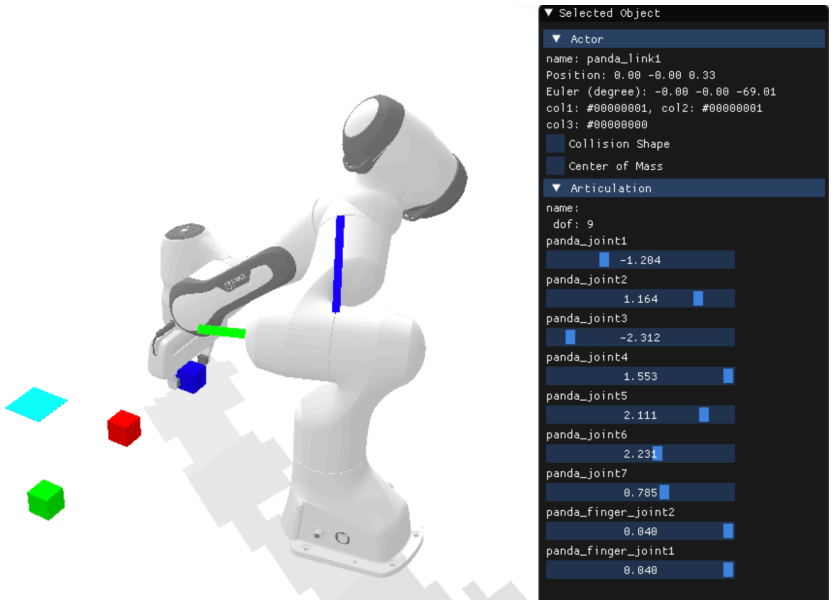
Related works

Classical approaches (model-based):

rely on accurate modeling of the environment, model-based trajectory optimization, analytic dynamics or kinematics.

The model: regular box

We can derive the state of end-effector once we know the pose of the object.



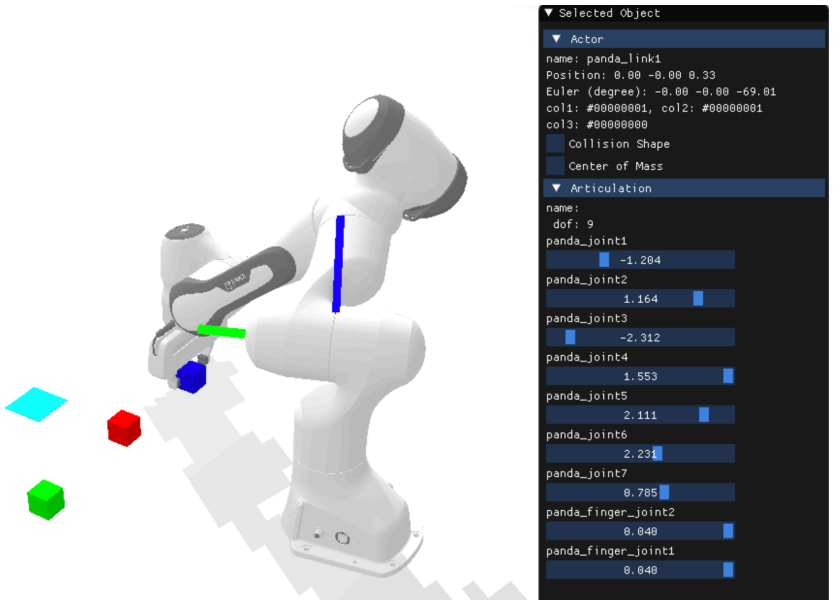
Related works

Classical approaches (model-based):

rely on accurate modeling of the environment, model-based trajectory optimization, analytic dynamics or kinematics.

Limitations:

- The state of end-effector can be complex (i.e., multi-fingered hand).
- We can not always obtain the detailed model, especially in real situations.



Related works

RL-based approaches (model-agnostic):

learn control policies without knowing the exact object model and system dynamics or kinematics.

Limitations:

Require large number of samples to learn a optimal policy or value function (sample inefficiency).

Related works

Visual imitation learning:

different from RL, imitation learning assumes that expert demonstrations are available, it learns the state-action mapping in the fashion of supervised learning.

Contributions

- Proof the concept that sensorimotor policies can be learned directly from raw visual information and tactile feedback.

- Faster training and better performance.

Proposed method

Formulation:

$$(\text{POMDP}) \quad \mathcal{M} = \{\mathcal{S}, \mathcal{X}, \mathcal{A}, \mathcal{R}, \mathcal{T}, \mathcal{G}, \rho_0, \gamma\}.$$

S: state space, not available in this setting.
X: observation space, come from sensory measurement (visual, tactile).
A: action space, the configuration of the hand.
ρ_0 : the probability distribution of the initial state.
γ : the discount factor.

$\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}^+$ and $\mathcal{G} : \mathcal{S} \times \mathcal{X} \to \mathbb{R}^+$ represent the state transition and observation models respectively.

When expert demonstrations are available, how to solve the POMDP problem efficiently?

A naïve method: behavior cloning.

$$(\mathrm{BC}) \quad \boldsymbol{\pi}_{\boldsymbol{\theta}^*} = \arg\max_{\boldsymbol{\pi}_{\boldsymbol{\theta}} \in \Pi} \mathbb{E}_{\boldsymbol{\tau}^e} \left[\ln \boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{a}_t^e | \mathbf{h}_t) \right]$$

$\boldsymbol{\pi}^{\boldsymbol{e}}$: the policy of the expert.
$\boldsymbol{\pi}_{\boldsymbol{\theta}}$: the policy need to be optimized.

For every state, treat the action of the expert as ground truth and do the supervised learning.

The problem: distribution drift.

$$\text{(BC)} \quad \boldsymbol{\pi}_{\boldsymbol{\theta}^*} = \arg\max_{\boldsymbol{\pi}_{\boldsymbol{\theta}} \in \Pi} \mathbb{E}_{\boldsymbol{\tau}^e} \left[\ln \boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{a}_t^e | \mathbf{h}_t) \right]$$

Optimizing the function over all the trajectory sampled with the expert policy. The trajectory of the agent may not exact the same as the expert's.

Can we improve the generalization of the agent by sampling more trajectories using current policy $\boldsymbol{\pi}_{\boldsymbol{\theta}^*}$?

Proposed method

A smarter way: imitation learning.

$$(\mathrm{IL}) \quad \boldsymbol{\pi}_{\boldsymbol{\theta}^*} = \arg\max_{\boldsymbol{\pi}_{\boldsymbol{\theta}} \in \Pi} \mathbb{E}_{\mathcal{D}} [\ln \boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{a}_t^e | \mathbf{h}_t)]$$

Training the agent on the trajectories collected through a mixed policy $\boldsymbol{\pi}^{\beta}$:

$$\boldsymbol{\pi}^{\beta} = \beta \boldsymbol{\pi}^{\boldsymbol{e}} + (1 - \beta) \boldsymbol{\pi}_{\boldsymbol{\theta}}$$

Proposed method

Algorithm 1 DAgger for imitation learning

1: Input expert policy $\boldsymbol{\pi}^{\boldsymbol{e}}$, mixing coefficient β, decay rate $\omega < 1$, and batch size N. Initialize $\mathcal{D} = \{\}$ and $\boldsymbol{\pi}_{\boldsymbol{\theta}}$.
2: **for** $k = 1$ **to** K **do**
3: Define mixture policy $\boldsymbol{\pi}^{\beta} = \beta\boldsymbol{\pi}^{\boldsymbol{e}} + (1-\beta)\boldsymbol{\pi}_{\boldsymbol{\theta}}$
4: Collect dataset $\mathcal{D}_k = \{\boldsymbol{\tau}^{(1)}, \ldots \boldsymbol{\tau}^{(N)}\}$ by rolling out $\boldsymbol{\pi}^{\beta}$. Also collect $\mathbf{a}_t^e \sim \boldsymbol{\pi}^{\boldsymbol{e}}(\cdot|\mathbf{s}_t)$ at every state visited in the rollouts.
5: Aggregate dataset: $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_k$
6: Update $\boldsymbol{\pi}_{\boldsymbol{\theta}}$ by re-solving optimization problem in (3)
7: Decay the mixing coefficient: $\beta \leftarrow \omega \times \beta$
8: **end for**

Gradually increase the coefficient for the optimized policy, which ensures the sampled trajectories will finally cover the states that the agent will encounter.

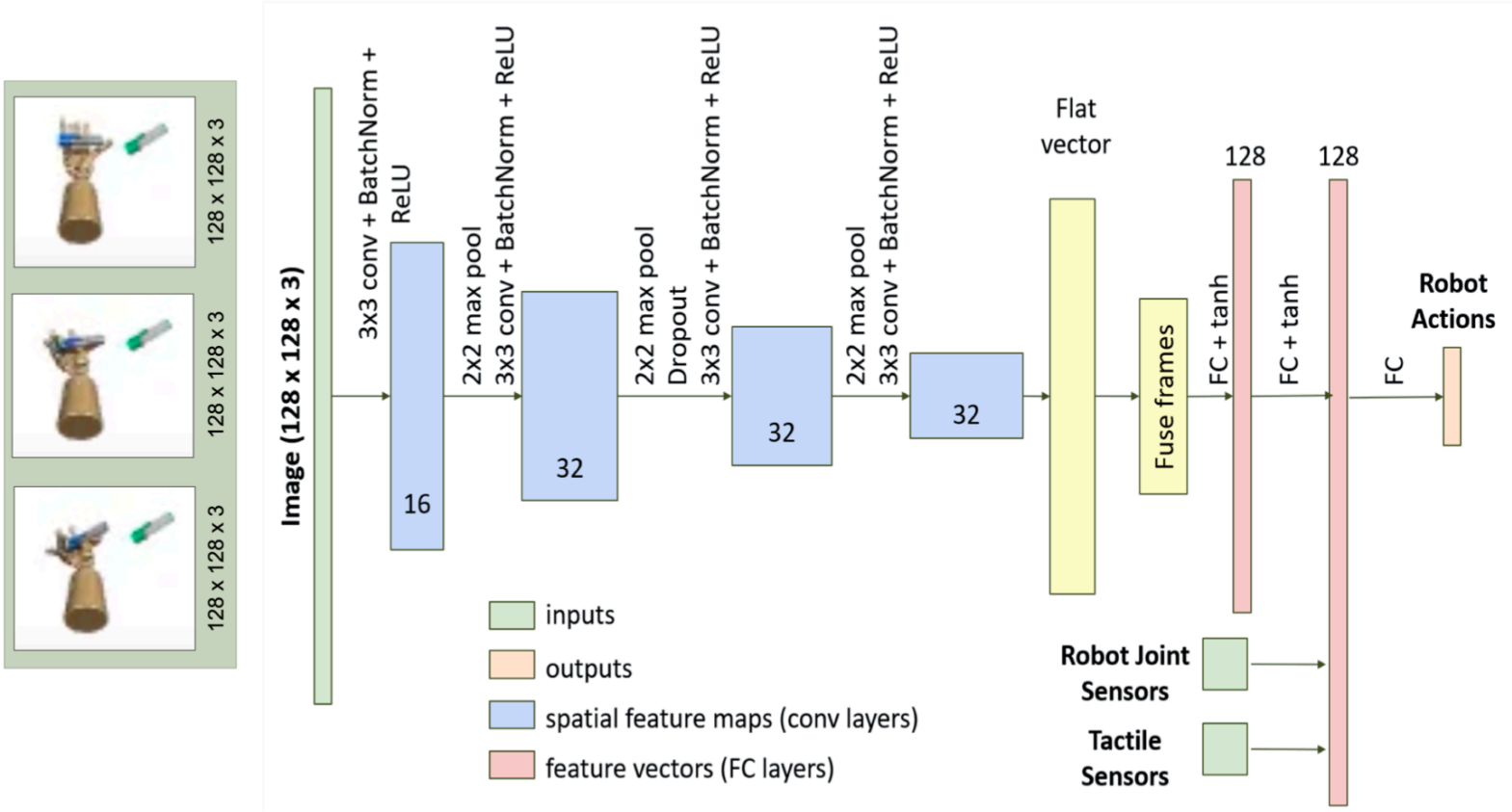
Proposed method

Algorithm 1 DAgger for imitation learning

1: Input expert policy $\boldsymbol{\pi}^{\boldsymbol{e}}$, mixing coefficient β, decay rate $\omega < 1$, and batch size N. Initialize $\mathcal{D} = \{\}$ and $\boldsymbol{\pi}_{\boldsymbol{\theta}}$.
2: **for** $k = 1$ **to** K **do**
3: Define mixture policy $\boldsymbol{\pi}^{\beta} = \beta\boldsymbol{\pi}^{\boldsymbol{e}} + (1-\beta)\boldsymbol{\pi}_{\boldsymbol{\theta}}$
4: Collect dataset $\mathcal{D}_k = \{\boldsymbol{\tau}^{(1)}, \ldots \boldsymbol{\tau}^{(N)}\}$ by rolling out $\boldsymbol{\pi}^{\beta}$. Also collect $\mathbf{a}_t^e \sim \boldsymbol{\pi}^{\boldsymbol{e}}(\cdot|\mathbf{s}_t)$ at every state visited in the rollouts.
5: Aggregate dataset: $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_k$
6: Update $\boldsymbol{\pi}_{\boldsymbol{\theta}}$ by re-solving optimization problem in (3)
7: Decay the mixing coefficient: $\beta \leftarrow \omega \times \beta$
8: **end for**

The state space is continuous and smooth, interpolation allows us to collect novel trajectories and makes the agent generalize better.

Proposed method



Inputs: RGB frames, joint position/velocity, contact information

Experiments

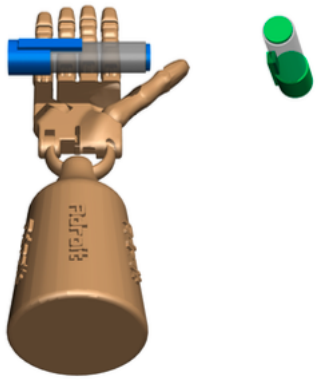
Experiment settings

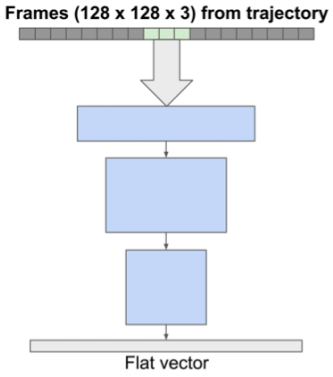
- Environment: MuJoCo
- Robot: five-fingered Adroit hand
- Sensors: camera, proprioceptive sensors (joint velocity and location), touch sensors

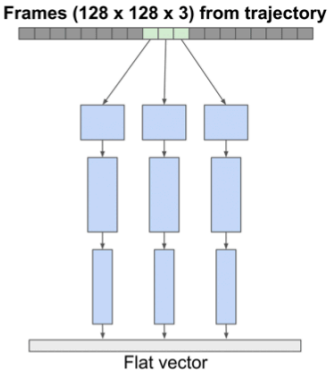
Tasks:

- **Object relocation:** grasp a sphere and move it to a target location.
- **In-hand manipulation:** reposition the pen to a desired target pose.
- **Tool usage:** pick up a hammer and use it to drive the nail into the board.
- **Door Opening:** undo a latch and open the door using the door handle.

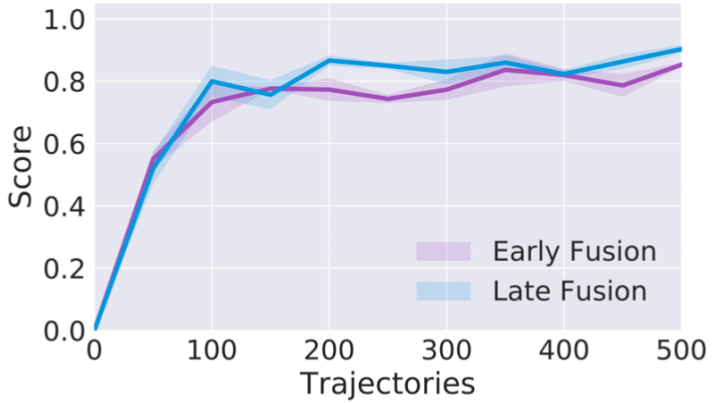
Compare different fusion strategy:





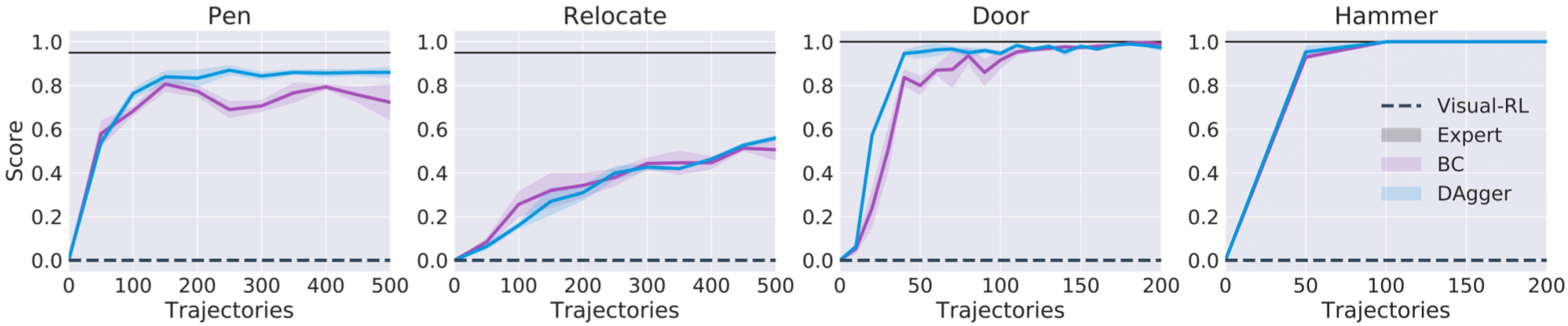


(a) Early (left) vs late (right) fusion architecture.

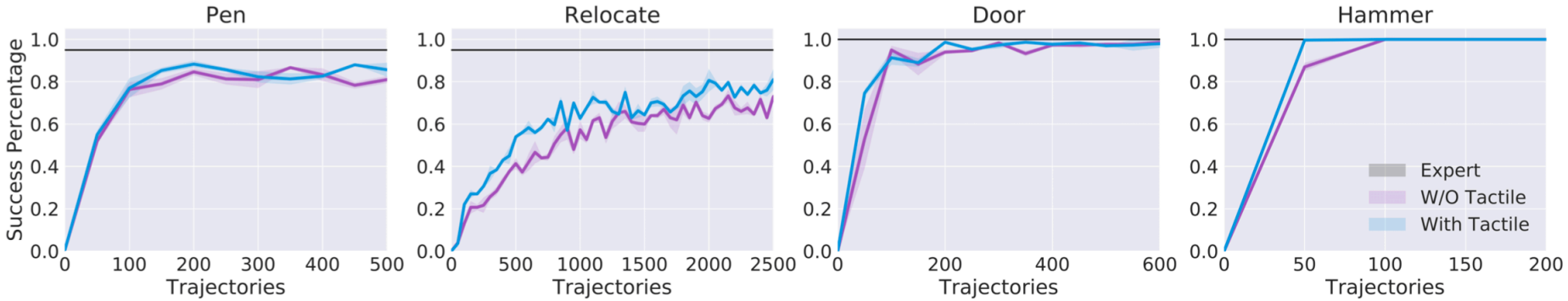


(b) Learning curve on pen repositioning task.

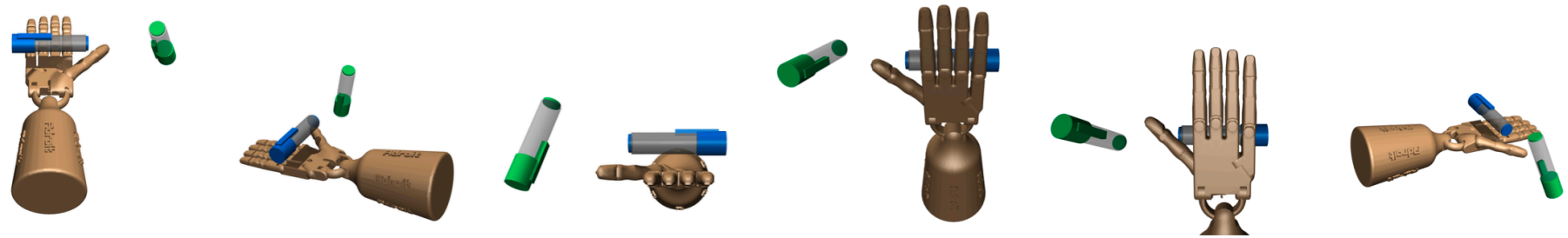
Compare different method:



Ablation study on tactile sensor:



Effectiveness of tactile sensing:



(a) View 1 (b) View 2 (c) View 3 (d) View 4 (e) View 5 (f) View 6

*CS denotes contact sensor

Sensing \ Views	1	2	3	4	5	6
Without CS	88	91	88	83	76	91
With CS	89	93	93	87	84	93

For viewpoint with heavy occlusion, contact sensor can improve the number of success.

Summary

- Learning control policies for dexterous tasks from raw sensor signals is feasible.
- When demonstrations are available, imitation learning can be more efficient than reinforcement learning.

Summary

- Learning control policies for dexterous tasks from raw sensor signals is feasible.
- When demonstrations are available, imitation learning can be more efficient than reinforcement learning.

Limitations:

Demonstrations are hard to be obtained.

The performance is bounded by the demonstrator.

Summary

Combining RL with demonstrations …

Learning Complex Dexterous Manipulation with Deep Reinforcement Learning and Demonstrations, Rajeswaran et al, RSS 2018.

Reinforcement and Imitation Learning for Diverse Visuomotor Skills, Zhu et al, RSS 2018.

Dexterous Manipulation with Deep Reinforcement Learning: Efficient, General, and Low-Cost, Zhu et al, ICRA 2019

Summary

Other issues (probably wrong):

- MDP assumes very short-term state dependency.
- RL/IL does not explicitly tackle trajectory/task planning.

Thank you!