MoPA-RL:
Motion Planner Augmented RL for Robot Manipulation in Obstructed Environments

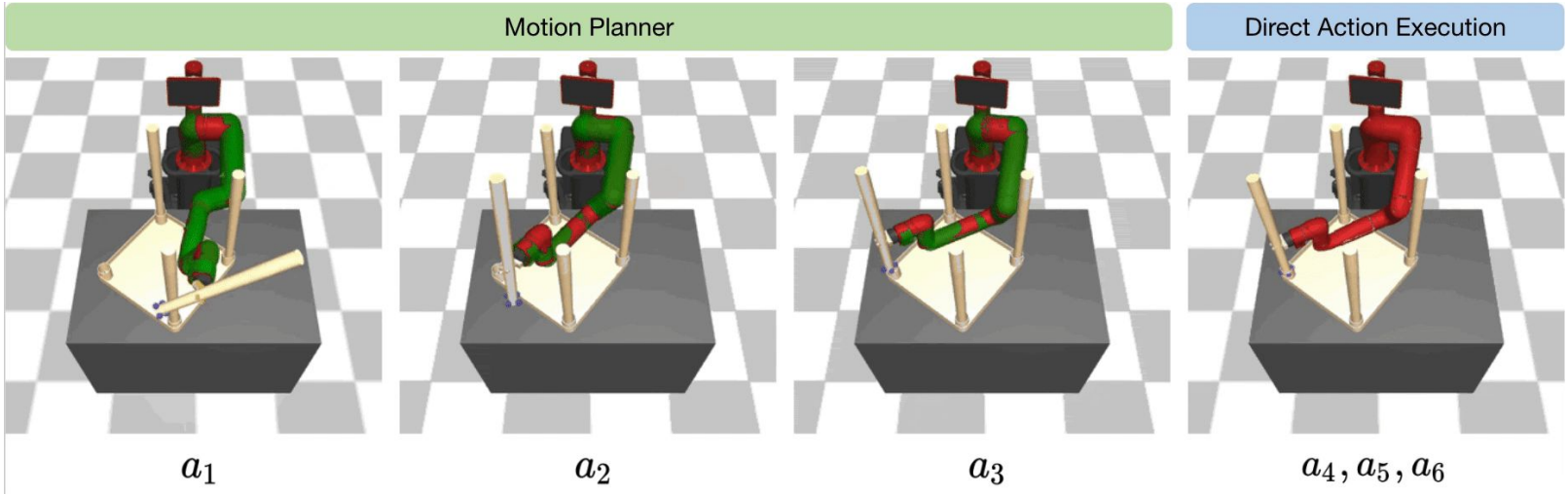
J. Yamada, Y. Lee, G. Salhotra, K. Pertsch, M. Pflueger, G. Sukhatme, J. Lim, and P. Englert
Published at CoRL 2020

16-832 Integrated Planning and Learning
Presented by Tom Gao

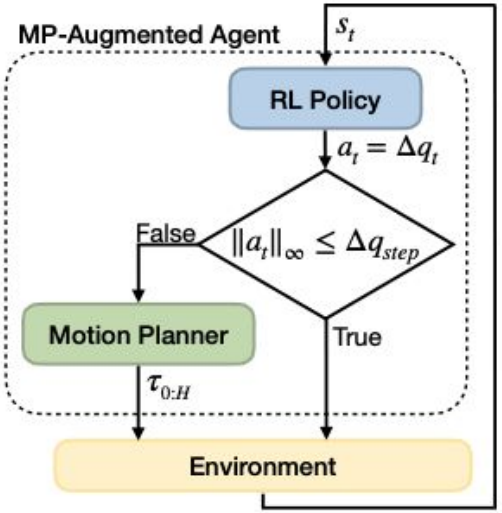
Background

Given a task that requires free-space travel and contact-rich manipulation:

- Hard to adapt traditional motion planners to handle contact-rich tasks, since they inherently avoid collision...
- Hard to train a manipulation-specific network to also plan arbitrary long-distance travel while avoiding collisions...



What is MoPA-RL?



Use action magnitude (delta joint angles) as a metric.

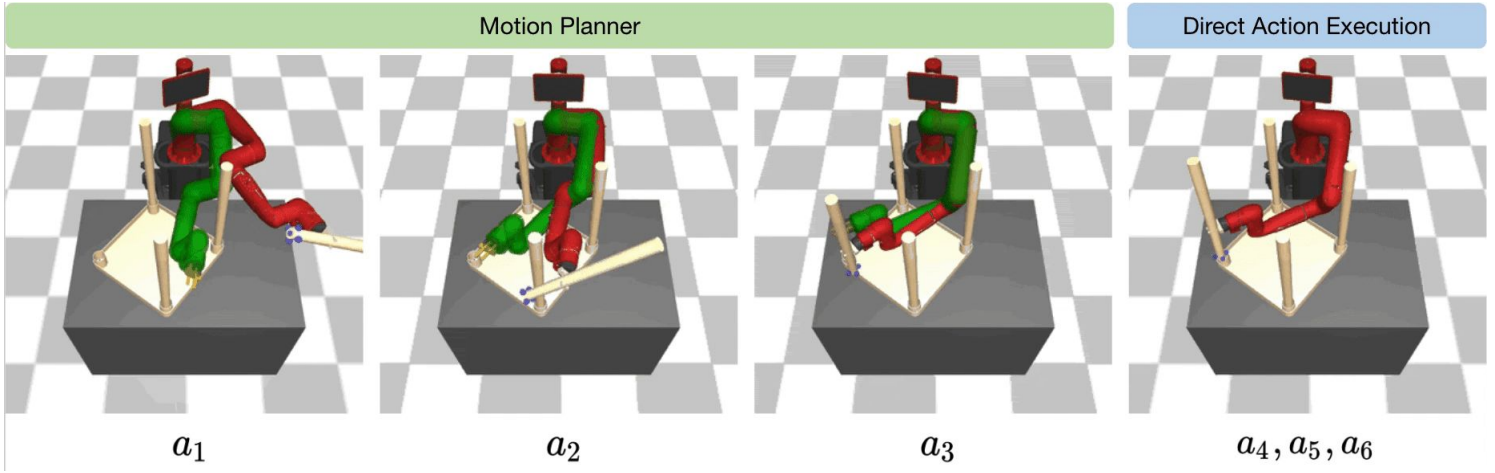
If sampled RL actions are within some magnitude, execute those actions directly, otherwise invoke the Motion Planner for long horizon travel.

Why MoPA-RL?

Significantly improve sample efficiency during training.

Solve contact-rich manipulation problems in obstructed environments.

Agent can learn to exploit motion planner and relegate long-range, collision aware motion to MP.



Related Works Analysis

Approach	Advantages	Drawbacks
Use Classical Motion Planning with explicit models only	Excellent long-horizon planning capabilities; generalizable	Struggles with dynamic, contact-rich physical interactions; requires high-quality models

Related Works Analysis

Approach	Advantages	Drawbacks
Use Classical Motion Planning with explicit models only	Excellent long-horizon planning capabilities; generalizable	Struggles with dynamic, contact-rich physical interactions; requires high-quality models
Use model-free Standard/Hierarchical RL only	Very capable of learning contact-rich skills by maximizing reward; no explicit model needed	Requires massive amounts of experience, explores very inefficiently (local optima/stuck…)

Related Works Analysis

Approach	Advantages	Drawbacks
Use Classical Motion Planning with explicit models only	Excellent long-horizon planning capabilities; generalizable	Struggles with dynamic, contact-rich physical interactions; requires high-quality models
Use model-free Standard/Hierarchical RL only	Very capable of learning contact-rich skills by maximizing reward; no explicit model needed	Requires massive amounts of experience, explores very inefficiently (local optima/stuck…)
Decompose task into parts; Use task heuristics to determine RL/MP switching, such as [1]	Well-explored, performant and data-efficient for learning specific skill.	Relies on task-specific heuristics. Heuristics not generalizable outside of limited task range.

[1] Xia et al.: Relmogen: Leveraging motion generation in reinforcement learning for mobile manipulation. 2020

Related Works Analysis

Approach	Advantages	Drawbacks
Use Classical Motion Planning with explicit models only	Excellent long-horizon planning capabilities; generalizable	Struggles with dynamic, contact-rich physical interactions; requires high-quality models
Use model-free Standard/Hierarchical RL only	Very capable of learning contact-rich skills by maximizing reward; no explicit model needed	Requires massive amounts of experience, explores very inefficiently (local optima/stuck…)
Decompose task into parts; Use task heuristics to determine RL/MP switching, such as [1]	Well-explored, performant and data-efficient for learning specific skill.	Relies on task-specific heuristics. Heuristics not generalizable outside of limited task range.
Hierarchical RL, with modular task-specific switching policies [2]	More integrated than strict task decomposition, and allows some flexibility in tasks	Depends on task-specific module switching, or hand-crafted goal definitions

[2] Lee et al.: Guided uncertainty-aware policy optimization: Combining learning and model-based strategies for sample-efficient policy learning. 2020, ICRA

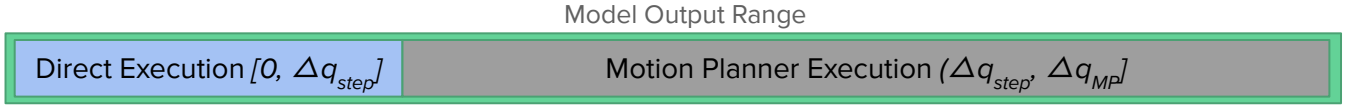
Defining the Problem - A Typical Manipulation Problem...

Markov Decision Process (MDP): $(\mathcal{S}, \mathcal{A}, P, R, \rho_0, \gamma)$ where actions are d-dimensional

- States, Actions, Transition Function, Reward, Init. State Distr., Discount Factor
- Agent executes action generated by policy $\pi_\phi(a_t|s_t)$ for reward $r_t = R(s_t, a_t)$
- Episode (length T) return is $\sum_{t=0}^{T-1} \gamma^t R(s_t, a_t)$

Actions are joint displacements: $a_t = \Delta q_t$

- We define Δq_{step} as a constraint on how large each joint delta is allowed to be when directly executing RL model actions.



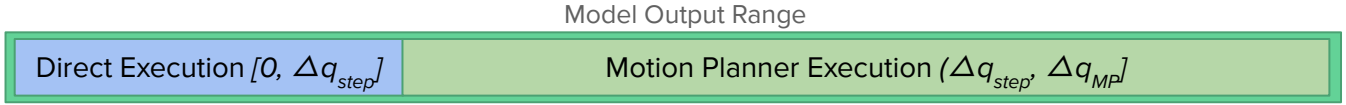
Defining the Problem - Augmenting MDP for sequences

Motion Planner creates a trajectory made up of many actions...

- But a traditional MDP only operates off of singular actions.
- Change Transition and Reward functions to use sequences of actions

Create an augmented MDP: $\tilde{\mathcal{M}}(\mathcal{S}, \tilde{\mathcal{A}}, \tilde{P}, \tilde{R}, \rho_0, \gamma)$

- Enlarged action space $\tilde{\mathcal{A}} = [-\Delta q_{\text{MP}}, \Delta q_{\text{MP}}]^d$, where $\Delta q_{\text{MP}} > \Delta q_{\text{step}}$ is a constraint on how far MP is allowed to travel
- Now each MP trajectory $\tau_{0:H}$ is also executed under one 'timestep'

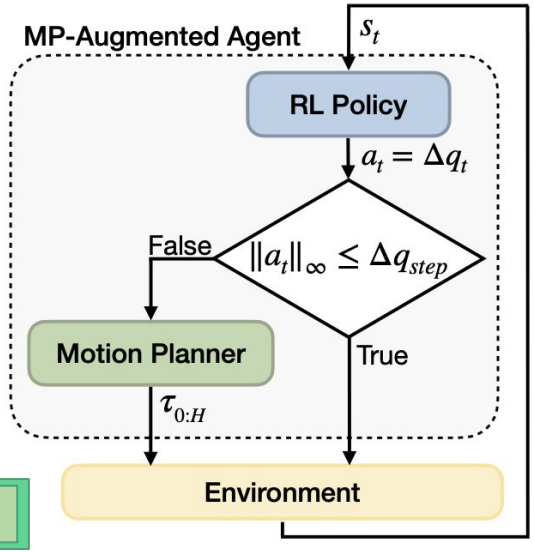


Assumptions in the Paper

- Explicit models are provided, and environment is fully observable
- MP trajectories can be treated as one logical, atomic step
  - Assumes manipulation motion is a solved problem and cannot fail in between
- Free-space navigation and contact-rich manipulation are distinct phases
  - Once an object is grasped, it is considered a rigid attachment to the gripper
- All obstacles are static (at least during execution of MP trajectories)
  - No dynamic interactions or external changes to world state

Explanation of Methodology - Maximum Magnitude

- Policy chooses an action $\tilde{a}$ within $\tilde{\mathcal{A}} = [-\Delta q_{\text{MP}}, \Delta q_{\text{MP}}]^d$
- The maximum magnitude of the action $||\tilde{a}||_\infty$ is examined:
- If $||\tilde{a}||_\infty > \Delta q_{\text{step}}$, call MP with goal $g = q + \tilde{a}$ to get path $\tau_{0:H}$
  - Execute actions in path over H steps.
- If $||\tilde{a}||_\infty \leq \Delta q_{\text{step}}$, directly execute $\tilde{a}$.

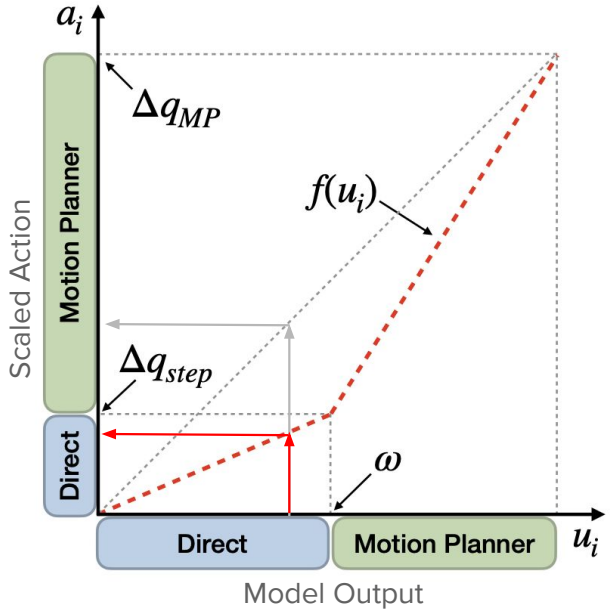


Model Output Range

Direct Execution *[0, Δq_{step}]*	Motion Planner Execution *(Δq_{step}, Δq_{MP}]*

Explanation of Methodology - Action Space Rescaling

- In practice, Δq_{MP} is much larger than Δq_{step}
  - With these tasks, 0.5 rad vs. 0.05 rad
- Probability of executing RL actions is very low
  - RL agent doesn't get frequent feedback
  - Precise manipulation only uses 10% of the model range
- Rescale the action space!
- $\omega \in [0, 1]$ is ratio between the action spaces
  - Best success rate in ablation study uses ω = 0.5 or 0.7
- With ω = 0.5, model can use half of its output range for the 0.05rad Δq_{step} and thus specify more precise actions.



Model Output Range

Direct Execution $[0, \Delta q_{step}]$	Motion Planner Execution $(\Delta q_{step}, \Delta q_{MP}]$

Explanation of Methodology - The Implementation

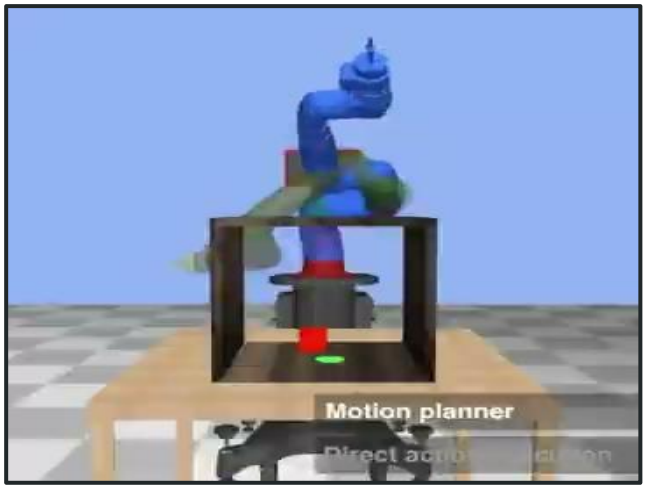
RL:

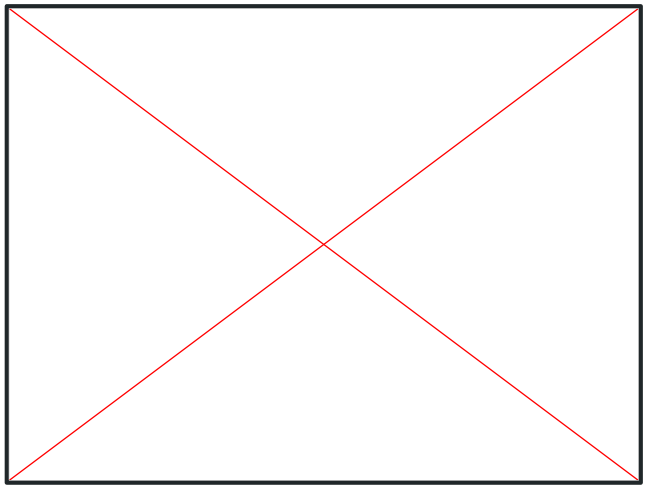
- Soft Actor-Critic: Both networks use 3 FC Layers, size 256, ReLU activations
- Policy outputs mean and stdev of a Gaussian distribution in action space
- Output passes through tanh(), then through action rescaling

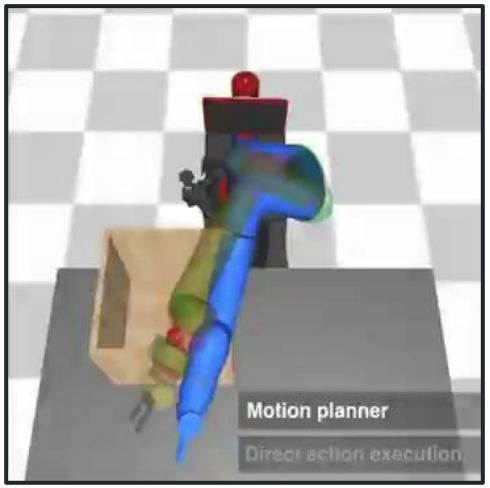
MP:

- Use collision checking function from MuJoCo engine
- Use OMPL implementation of RRT-Connect with shortcutting post-processing
- If predicted goal invalid, reduce action magnitude and check collision again
    - Reduce wastage of invalid samples, mainly to increase training efficiency

Explanation of Methodology - Examples







Blue = Motion Planner
Red = Direct Execution

Action norms are large except for object interactions

MP traversals can surround RL actions

Theoretical Guarantees (or, lack thereof...)

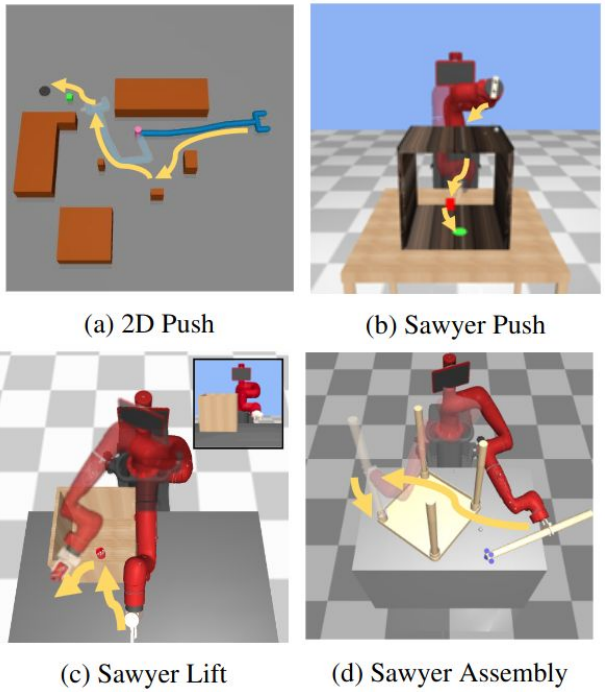
- MoPA-RL does NOT offer any theoretical guarantees for the solution
- The underlying MP implementation, such as RRT, may have guarantees
  - However, RL output dictates where MP plans towards
  - Bad RL model can still lead to task failure

Experimental Results - Models and Tasks

Models:

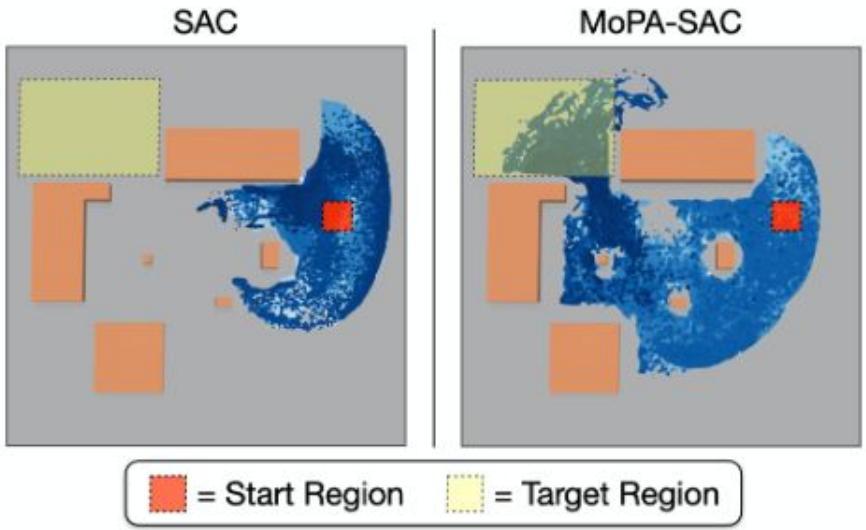
- SAC (predicts within Δq_{step})
- SAC Large (within Δq_{MP})
- SAC IK (predicts cartesian delta)
- MoPA-SAC (“Ours”)
- MoPA-SAC Discrete (explicitly predict MP/RL)
- MoPA-SAC IK (predicts cartesian delta)

Tasks:



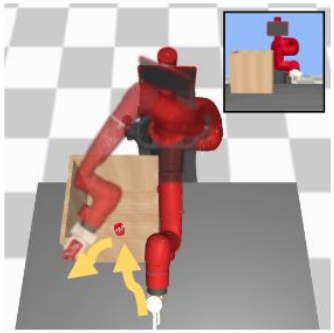
Experimental Results - 2D Intuition

- On the 2D example, within 100k training steps
- Motion Planning usage allows agent to explore more of the environment, earlier on
- Significant density of agents for MoPA-SAC already within the goal region

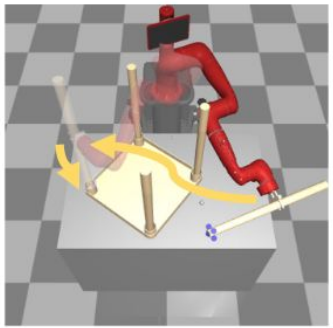


Experimental Results

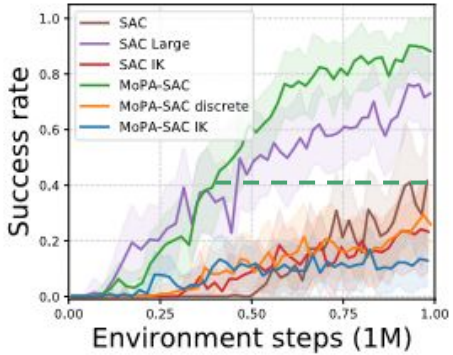
- “Ours” significantly outperformed baselines, especially in Lift/Assembly
- Sample efficiency is 4-5x that of vanilla SAC with the same training steps
- IK versions performed poorly
- Discrete results show decrease in success rate when MP/RL switch is being predicted



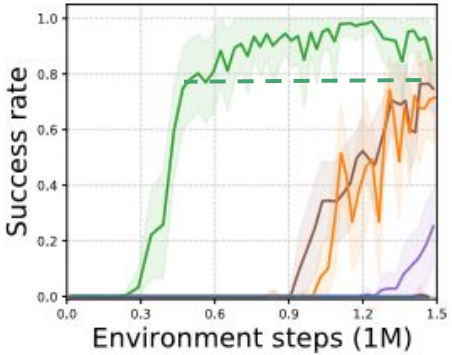
(c) Sawyer Lift



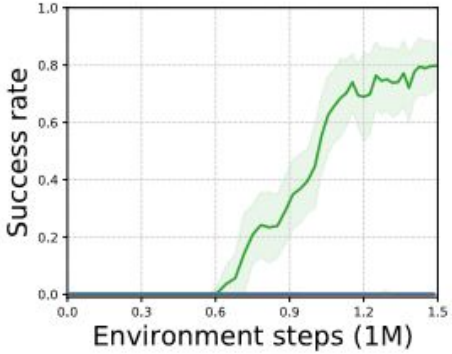
(d) Sawyer Assembly



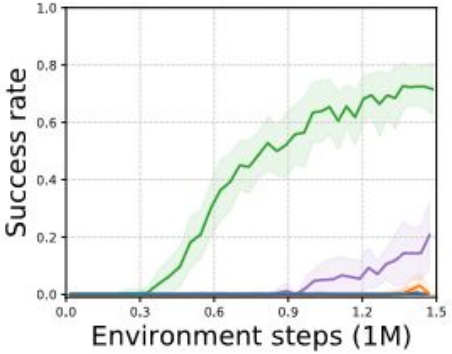
(a) 2D Push



(b) Sawyer Push



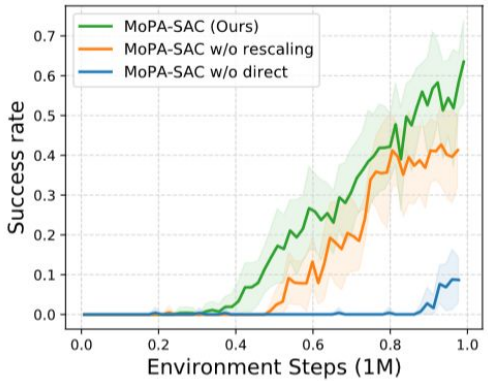
(c) Sawyer Lift



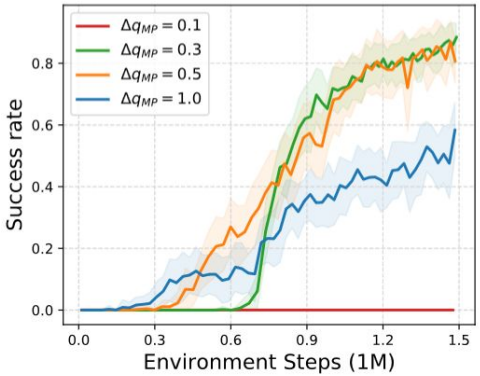
(d) Sawyer Assembly

Experimental Results - Additional Ablations

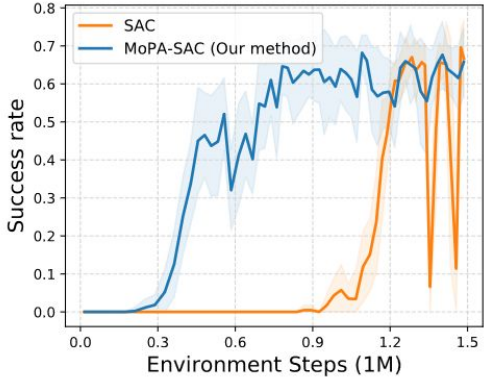
Action space rescaling is more performant (green vs. orange)



Sweet-spot for model action range is 0.3-0.5 rads per joint



On unobstructed tasks, MoPA is still 2-3x more sample-efficient than SAC



Limitations and Concerns

- MoPA-RL uses explicit models and assumes full observability
  - MoPA-PD (Liu et al., 2021) uses MoPA-RL as a teacher in simulation, then uses Behavior Cloning to distill this into a visual agent.
- MoPA-RL does not handle dynamic obstacles
  - Limits usability in tasks with movable scene or external interference
- Focuses on individual, simplistic tasks
  - Each Sawyer Assembly task for the table is actually FOUR simple episodes
  - Does not explicitly mention multi-stage tasks with complex interactions
- RL/MP switching metric does not generalize to all tasks
  - Assumes smaller actions <-> contact-rich tasking, which isn't necessarily always true
  - For example, unscrewing my cup requires large actions AND contact-rich tasking...

QnA

Thanks for Listening!