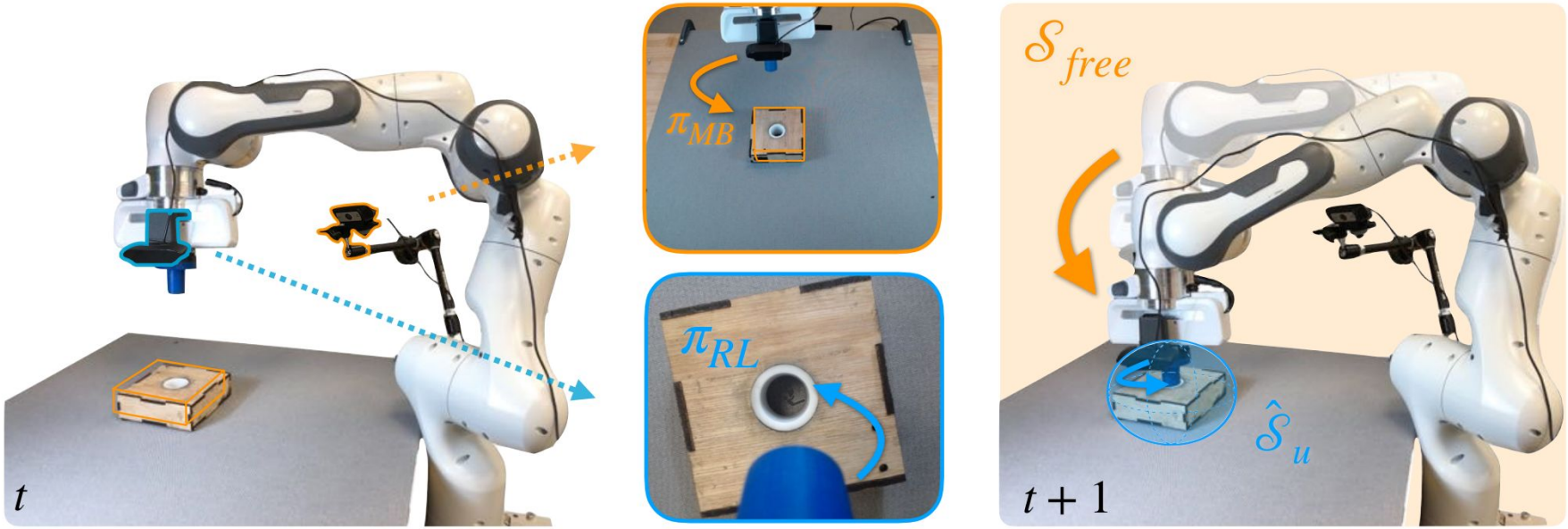
Guided Uncertainty-Aware Policy Optimization: Combining Learning and Model-Based Strategies for Sample-Efficient Policy Learning

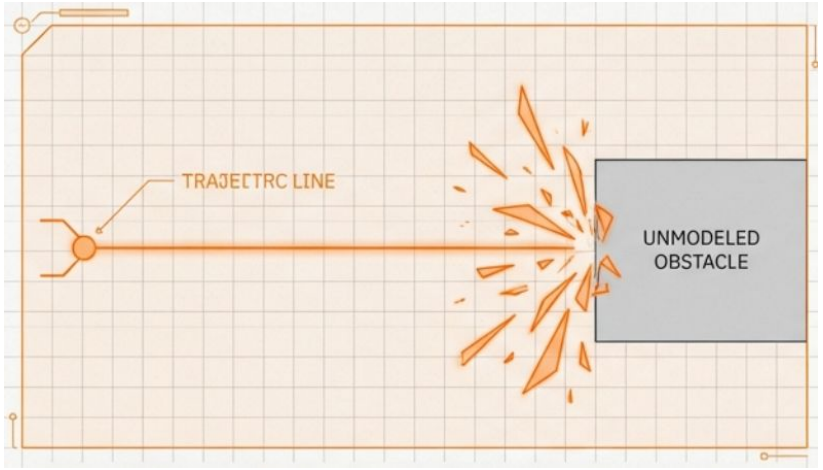
Michelle A. Lee* , Carlos Florensa*, Jonathan Tremblay , Nathan Ratliff, Animesh Garg, Fabio Ramos, Dieter Fox



16832 - Integrated Planning and Learning

Presenter: Yifu Yuan

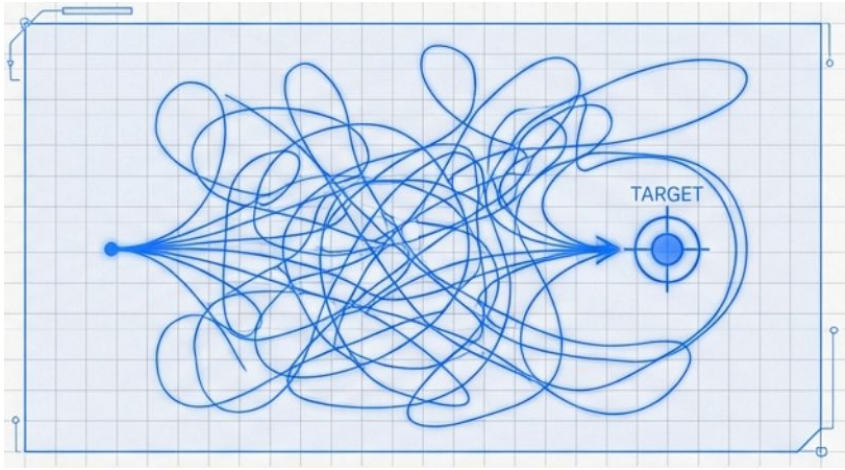
Motivation



Model-based Approach

Fast in free space; highly efficient

Sensitive to perception noise, unmodeled physical contact

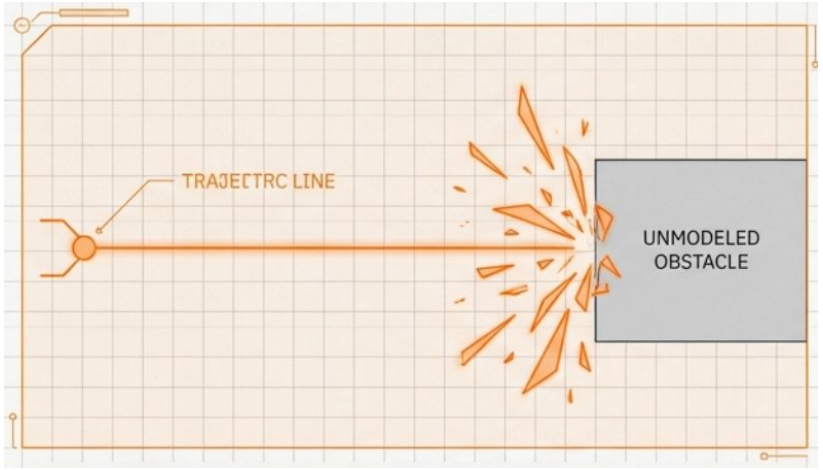


Learning-based Approach

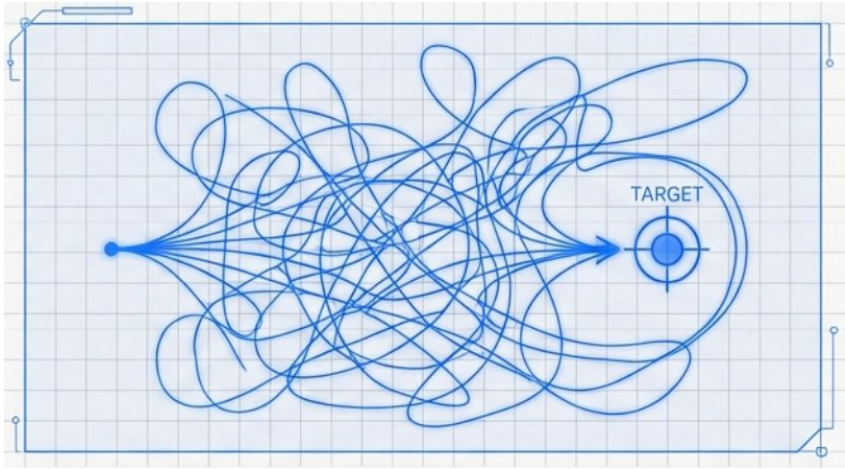
Adapts to raw sensors inputs and unstructured physical contacts

Sample-inefficient; complex reward design

Motivation

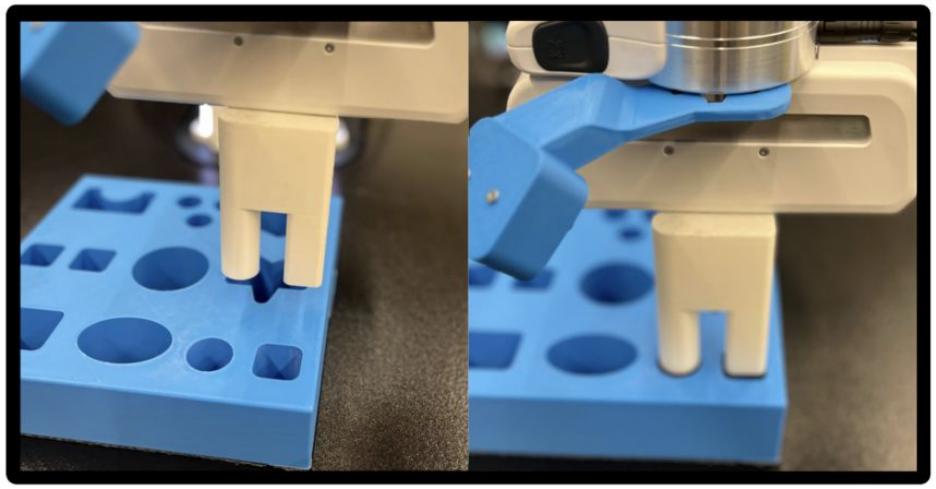


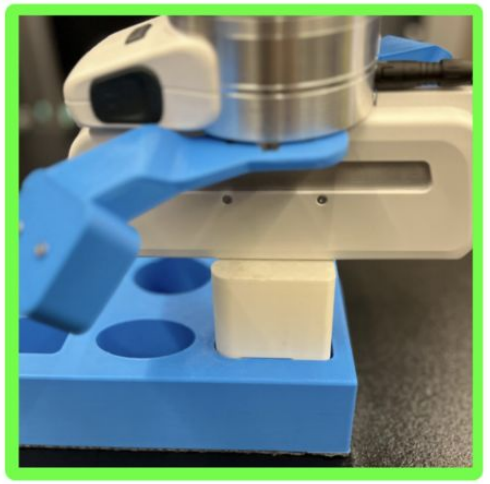
Model-based Approach
Navigate free space with model-based confidence and efficiency



Learning-based Approach
Handle contact with learning-based adaptability

Related Work

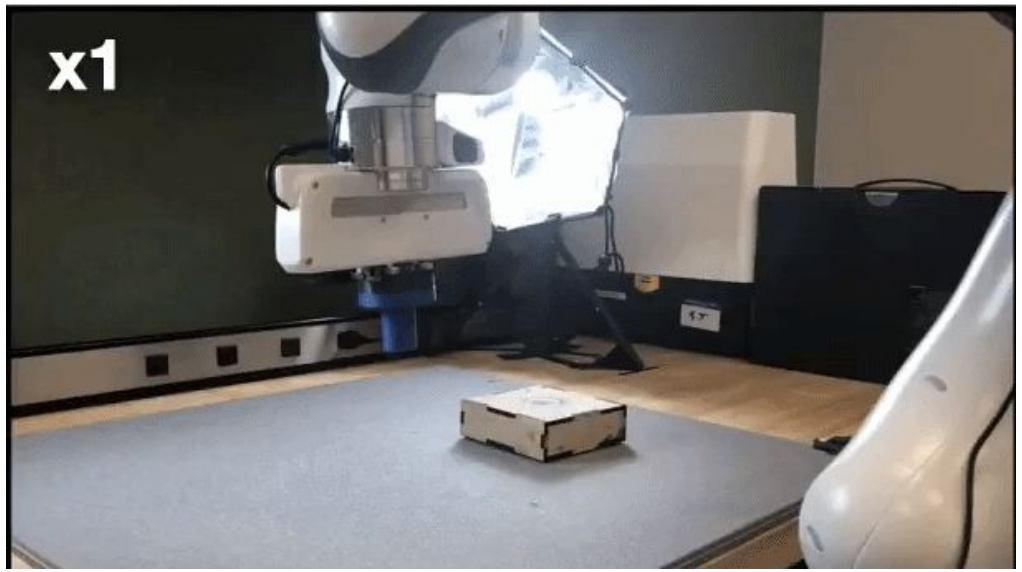




- **Traditional MB**: Requires exact CAD models and perfect state estimation; brittle to unmodeled noise.
- **Pure RL**: Requires highly engineered shaped rewards; heavily tuned reward functions.
- **Residual Learning**: Adds an RL policy on top of an MB policy, but is hard to tune, takes unsafe actions, and often loses MB benefits

Related Work

Traditional MB

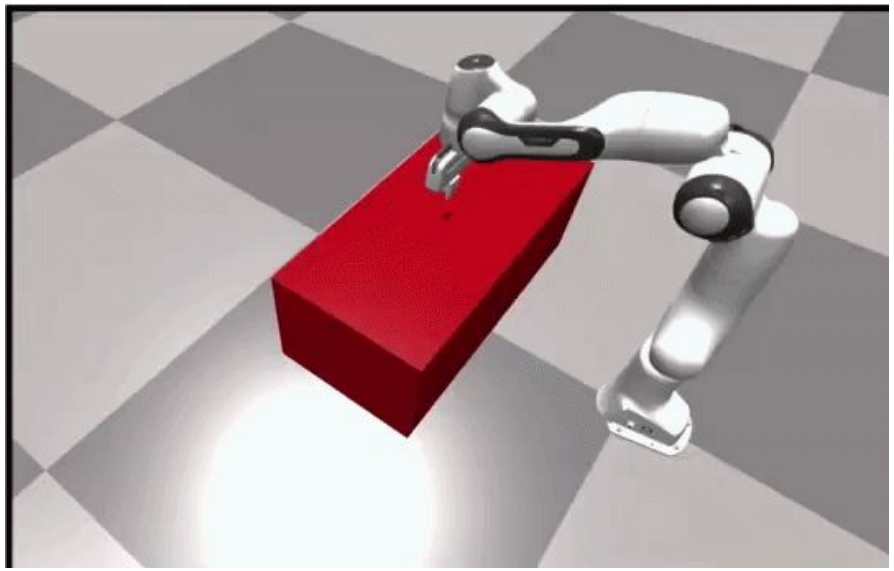


Efficient motion-planning and generalization

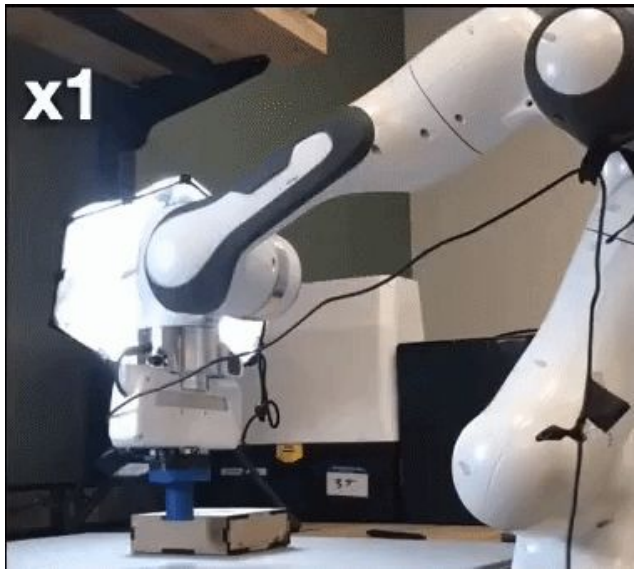
Fails under model and perception uncertainty

Related Work

RL

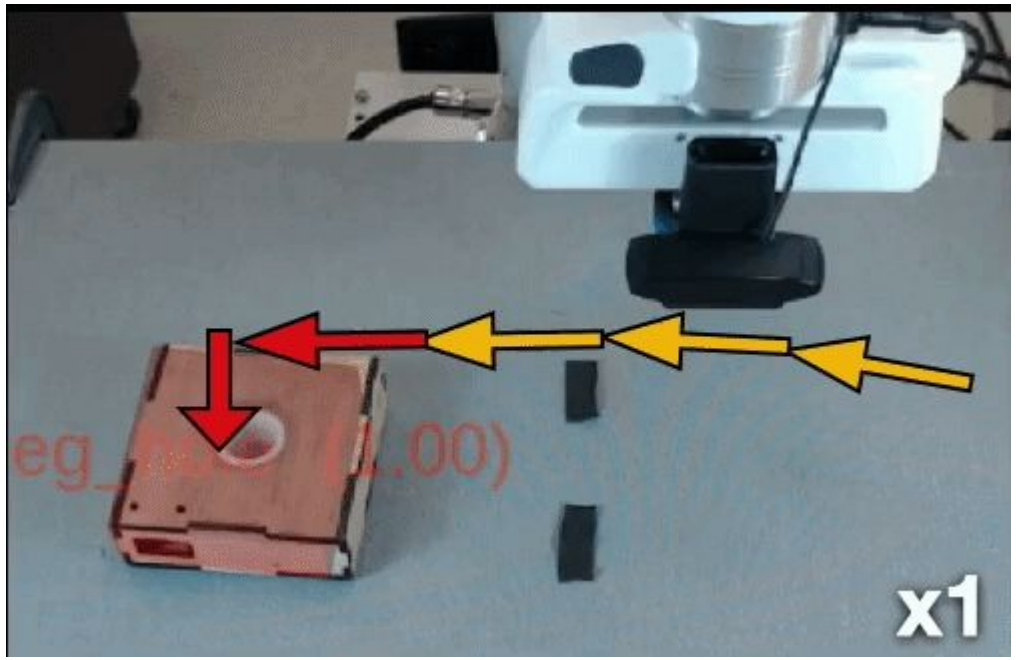


Learn from raw sensory inputs and reward, more generalized



Sample inefficient, unsafe

Methodology



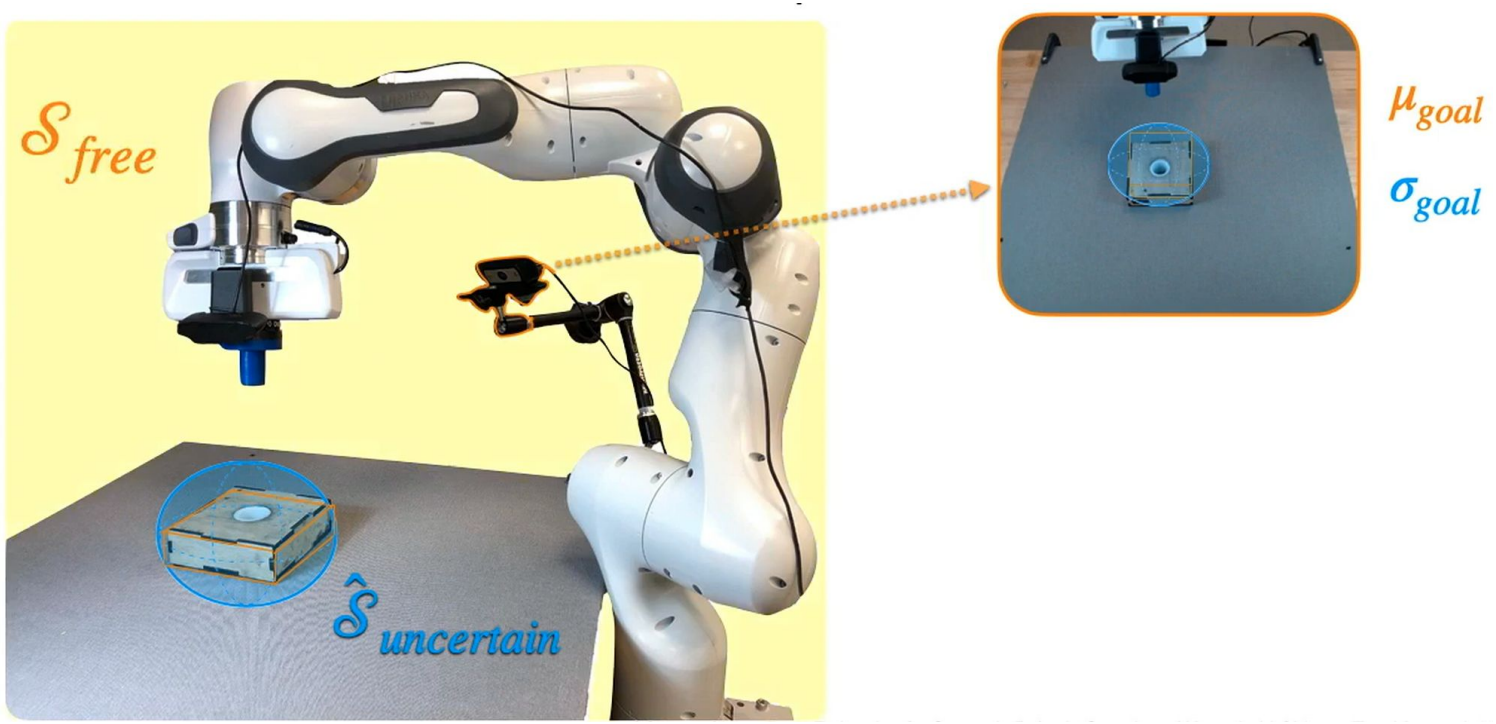
MB: are great at navigating to the blocks, but fail to finish when there is uncertainty in the estimation

Uncertainty Region: a position uncertainty estimate from the perception system to describe the region, which the action score, position, or dynamics are uncertain

RL: handle the uncertainty with learning-based adaptability

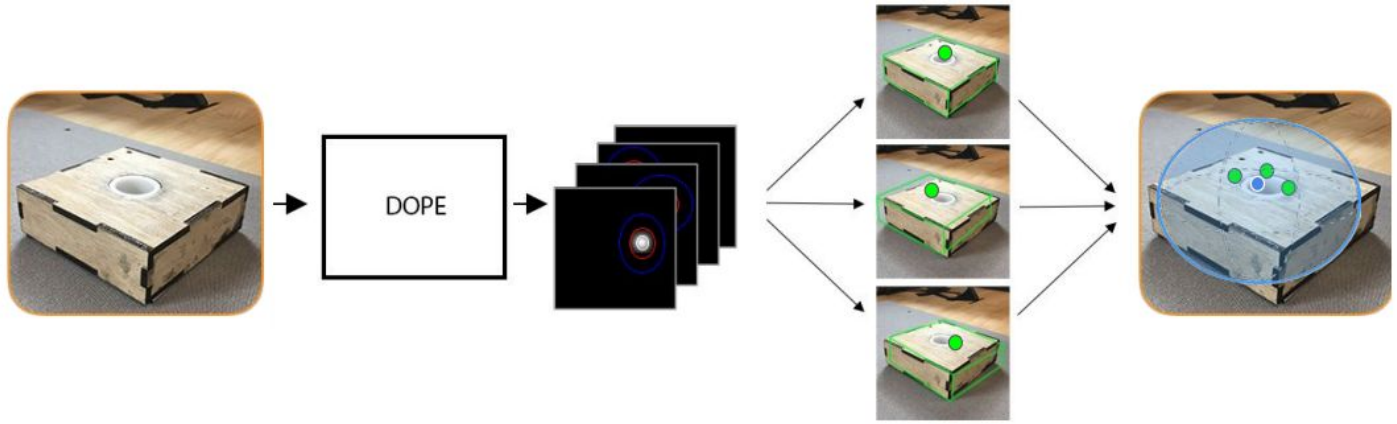
Methodology

Step1: use perception and uncertainty to partition the state-space



Methodology

Step1: use perception and uncertainty to partition the state-space

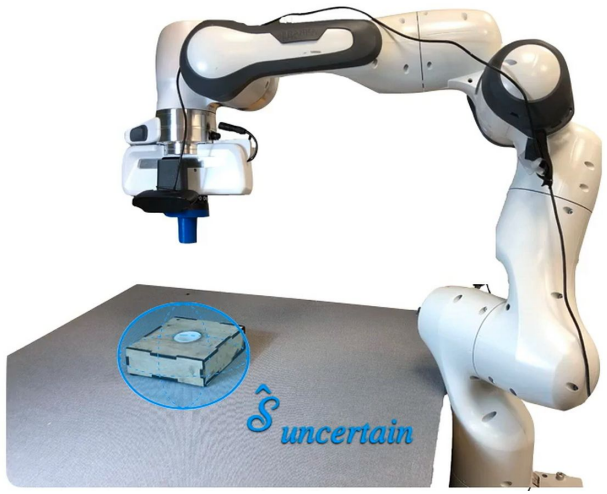


- Extends the Deep Object Pose Estimator (DOPE) to detect object keypoints from an overhead RGB camera
- Models uncertainty by fitting 2D Gaussians around detected keypoints to sample multiple potential object poses
- Aggregates poses to generate a 3D Gaussian representing the estimated target location
- The center of the uncertainty space is defined as the mean of this 3D Gaussian, and the physical boundary of the uncertainty space is calculated by expanding the base model outward by one standard deviation

$$p(s \in \mathcal{S}_u) = \sum_{i=1}^{n} \mathbb{1}\left[s \in \mathcal{S}_u^i\right] p(\mathcal{S}_u^i).$$

Methodology

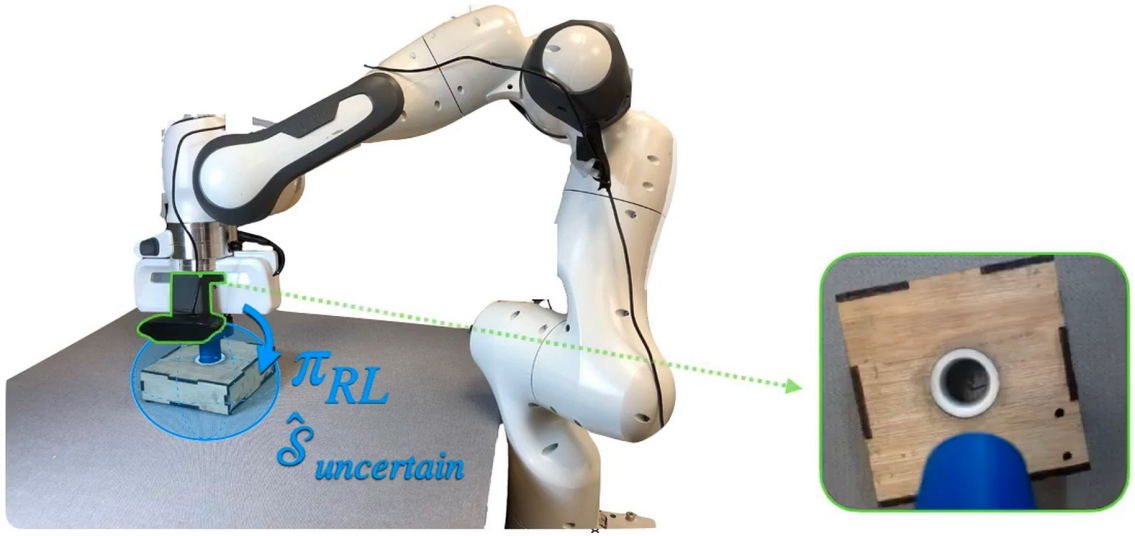
Step2: use motion planner to navigate to $\mathcal{S}_u$



- Activate only in free space
- Utilizes Riemannian Motion Policies (RMPs) as the model-based controller
- Takes in a desired end-effector position in Cartesian space
- Targets the centroid of uncertainty space as the destination coordinate
- Capable of integrating obstacle avoidance by removing uncertain obstacle regions from free space

Methodology

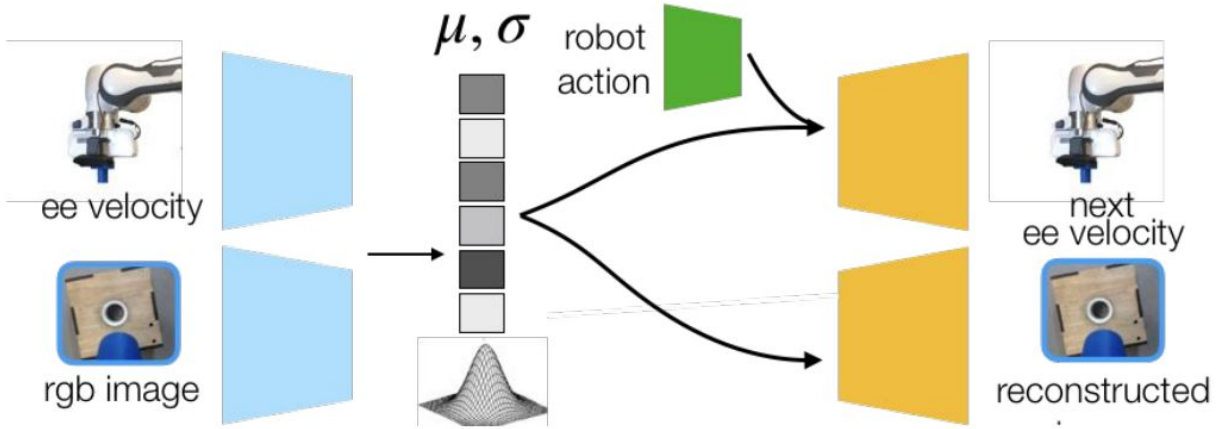
Step3: use RL to learn a policy within $\mathcal{S}_u$



- trained with raw sensory input:
  - RGB images from wrist-mounted camera
  - robot joint velocities
- If the RL policy steps outside the uncertainty region, the MB policy regains control and safely funnels it back

Methodology

Step3: use RL to learn a policy within $\mathcal{S}_u$

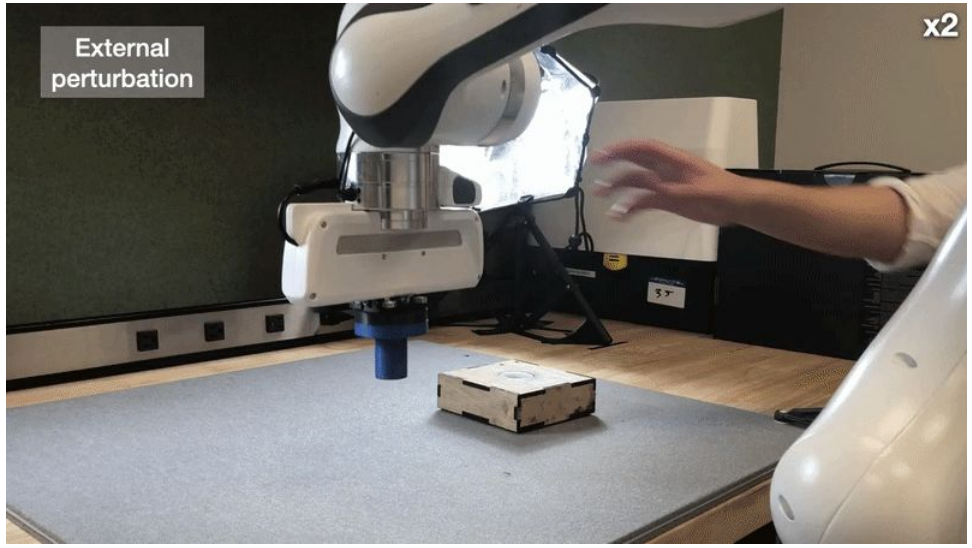


- **Policy**: Soft Actor-Critic (SAC)
- **Input**: RGB images (64x64x3) from wrist camera + EE velocity
- **Latent Representation**: The raw images and velocities are passed through a β-Variational Autoencoder (β-VAE) to compress the data into a low-dimensional, 64D latent-space representation
- **Policy Network**: A 2-layer MLP outputs 3D position displacements for the end-effector
- **Reward Design**: -1 everywhere, 0 when it finishes the task

Results

Start in different positions

Results



Results

	MB-Perfect	MB-DOPE	MB-Rand-Perfect	MB-Rand-DOPE	SAC [24]	RESIDUAL [9]	GUAPO (ours)
Success Rate	100%	0%	86.67%	26.6%	0%	0%	93%
Avg. Steps for Task Completion	158.3	n/a	554.1	925.4	n/a	n/a	469.6
In $\mathcal{S}_u$	100%	0%	100%	70.0%	0%	0%	100%
In $\hat{\mathcal{S}}_u$	100%	100%	100%	93.3%	0%	100%	100%

- **MB-Perfect**: This method consists of a scripted policy under perfect state estimation.
- **MB-Rand-Perfect**: This method uses the same policy as MB-Perfect where injected random actions, which sample from a normal distribution with 0 mean and a standard deviation defined by the perception uncertainty from DOPE
- **MB-DOPE**: This method is similar to MB-Perfect, but instead uses the pose estimator prediction to servo to the hole and accomplish insertion.
- **MB-Rand-Dope**: This method uses the same policy as MB-Dope where injected random actions, which is sampled in the same way as MB-Rand-Perfect.
- **SAC**: This uses just the policy learned from the RL algorithm, Soft-Actor Critic (SAC), to accomplish the task.
- **Residual**: This method is based on residual learning techniques that combine model-based and reinforcement learning methods

Limitations

- **Only goal-reaching task**: GUAPO Only works with goal-reaching tasks where the target can be identified with perception system, does not work with tasks involves reference trajectory tracking, like wiping or pushing taks.
- **Need for Object Models**: Requires access to a coarse model or rough description of the target object and the area of interest to train the initial DOPE perception system

Q&A