Planning-Guided Diffusion Policy Learning for Generalizable Contact-Rich Bimanual Manipulation

Xuanlin Li , Tong Zhao, Xinghao Zhu, Jiuguang Wang, Tao Pang, Kuan Fang

Presenter: Megan Lee

Paper Info

Affiliations: Boston Dynamics AI Institute, UC San Diego, Cornell University

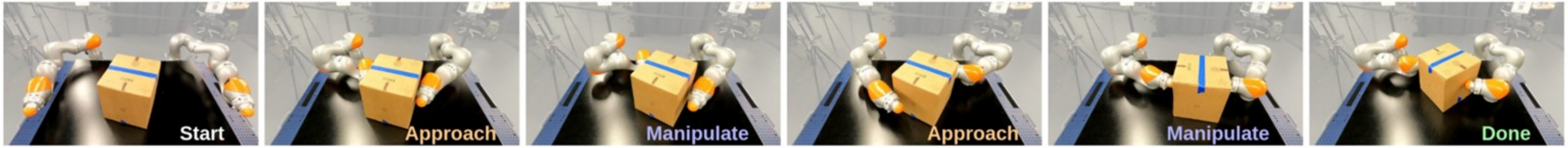
arXiv preprint https://arxiv.org/abs/2412.02676

Last Updated: Feb 14, 2025

Website: https://glide-manip.github.io/

Motivation

The challenge: contact rich bimanual manipulation for bulky objects require coordination and multiple phases



1. Model based planners:
    a. need fully observable spaces
    b. too slow for online trajectory generation
    c. can't generalize to new objects and new environments
2. Learning based methods:
    a. Require many expert demonstrations that are hard to collect
    b. Sim to real gap

The Idea

This paper introduces GLIDE (**G**eneralizable P**L**anning-Gu**I**ded **D**iffusion Policy L**E**arning)

1. Uses contact-implicit trajectory optimization solver in sim to generate large scale data
    a. Derived from smoothed linear approximation of local contact dynamics
    b. Greedily approach goal object state to speed up data generation
2. Train a task-conditioned diffusion policy to manipulate object to arbitrary target poses
3. Sim to real improvements

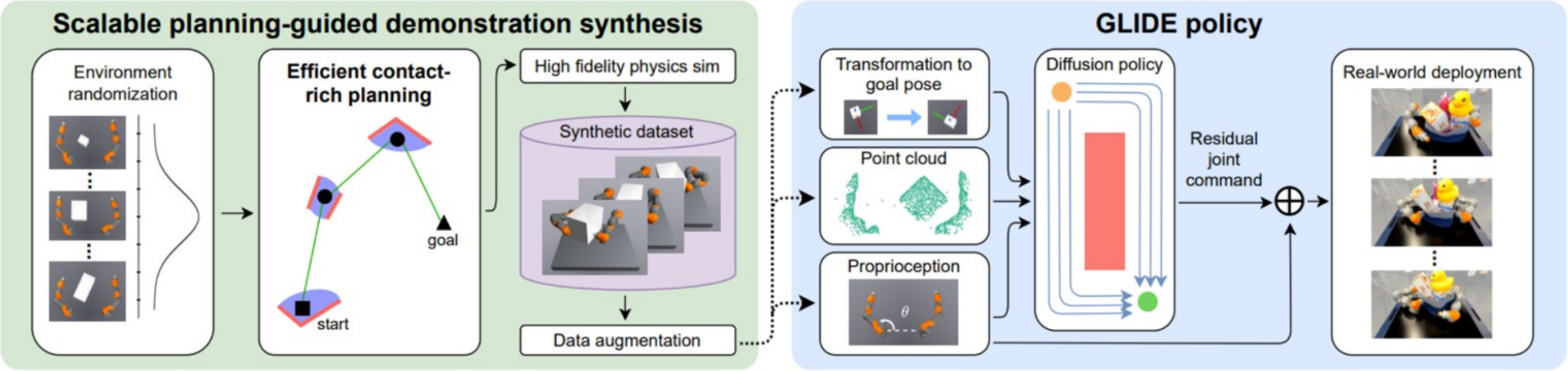
Related Work

Area	Prior Work	The gap
Contact-Rich and Bimanual Manipulation	Mixed Integer Programming (MIP) optimal but scales poorly Smooth contact models need full state knowledge	Can't generalize to new objects and environments
Visuomotor Policy Learning for Manipulation	Diffusion policy, DP3 capture multimodal actions	No goal conditioning, one policy per fixed goal
Planning-guided data synthesis	Imitating TAMP (manip), Motion Policy Networks (collision avoidance)	Not explored for contact rich bimanual manipulation

Assumptions

1. Static Environment
    a. Table top setting
    b. No dynamic obstacles
2. Rigid object, appropriate size and weight
    a. Can’t be lifted by end effectors but not too heavy that the arms can’t rotate it
3. SE(2) target goals, planer transformations only
4. Initial shape and pose of object are not known, received from visual observations
5. No real world training data is collected but should be transferable to real world

Method


Scalable planning-guided demonstration synthesis
Environment randomization
Efficient contact-rich planning
goal
start
High fidelity physics sim
Synthetic dataset
Data augmentation
GLIDE policy
Transformation to goal pose
Point cloud
Proprioception
Diffusion policy
Residual joint command
Real-world deployment

Data Synthesis

Robot joint angle

Object pose

Algorithm 1: Demonstration Synthesis via Planning

Input: $q_0^a, q_0^u, q_{\text{goal}}^u$ → Goal object pose

Output: Action trajectory T

$q^a,\ q^u,\ T \leftarrow q_0^a,\ q_0^u$, `list()`

while $q^u \neq q_{\text{goal}}^u$ **do**

  $q_{\text{grasp}}^a \leftarrow \text{SampleContact}(q^u)$

  while $q^a \neq q_{grasp}^a$ **do**

    $q^a, a \leftarrow \text{PlanCollisionFree}(q^a, q_{\text{grasp}}^a)$

    $T.\texttt{extend}(a)$

  while q^a *not at joint limit and* $q^u \neq q_{\text{goal}}^u$ **do**

    $q^a, q^u, a \leftarrow \text{PlanContact}(q^a, q^u, q_{\text{goal}}^u)$

    $T.\texttt{extend}(a)$

return T

Contact Sampler:
Generates robot joint configurations where arm makes contact with object
-> used IK to generate grasps where robot links can stably hold object

Data Synthesis

Algorithm 1: Demonstration Synthesis via Planning

Input: $q_0^a, q_0^u, q_{\text{goal}}^u$
Output: Action trajectory T
$q^a,\ q^u,\ T \leftarrow q_0^a,\ q_0^u$, `list()`
while $q^u \neq q_{\text{goal}}^u$ **do**
  $q_{\text{grasp}}^a \leftarrow \text{SAMPLECONTACT}(q^u)$
  while $q^a \neq q_{grasp}^a$ **do**
    $q^a, a \leftarrow \text{PLANCOLLISIONFREE}(q^a, q_{\text{grasp}}^a)$
    T.`extend`(a)
  while q^a *not at joint limit and* $q^u \neq q_{\text{goal}}^u$ **do**
    $q^a, q^u, a \leftarrow \text{PLANCONTACT}(q^a, q^u, q_{\text{goal}}^u)$
    T.`extend`(a)
return T

Collision free planner:
Using Bidirectional RRT with shortcutting to plan collision free trajectory to next grasp q_{grasp}^a

Data Synthesis

Algorithm 1: Demonstration Synthesis via Planning

Input: $q_0^a, q_0^u, q_{\text{goal}}^u$
Output: Action trajectory T
$q^a,\ q^u,\ T \leftarrow q_0^a,\ q_0^u$, list()
while $q^u \neq q_{\text{goal}}^u$ **do**
  $q_{\text{grasp}}^a \leftarrow \textsc{SampleContact}(q^u)$
  while $q^a \neq q_{grasp}^a$ **do**
    $q^a, a \leftarrow \textsc{PlanCollisionFree}(q^a, q_{\text{grasp}}^a)$
    T.extend(a)
  while q^a *not at joint limit and* $q^u \neq q_{\text{goal}}^u$ **do**
    $q^a, q^u, a \leftarrow \textsc{PlanContact}(q^a, q^u, q_{\text{goal}}^u)$
    T.extend(a)
return T

Contact Planner:
After contact is made with object, greedily move the object towards goal configuration as much as possible without reaching joint limit

- With contact, dynamics are non-smooth. There are ways to mitigate this but they have high computational costs
- They instead use a single step variant, optimize one step at a time which makes it inherently greedy
- Linear approximator of non-smooth dynamics f_{local}
  With objective function:

$$\min_{q_+^u, a} \ (q_+^u - q_{\text{goal}}^u)^T \mathbf{Q} (q_+^u - q_{\text{goal}}^u) + (a - q^a)^T \mathbf{R} (a - q^a)$$

$$q_+^u = f_{\text{local}}(q^u, q^a, a)$$

Object approx config after action a has been taken
Q = state cost matrix, R = control cost matrix

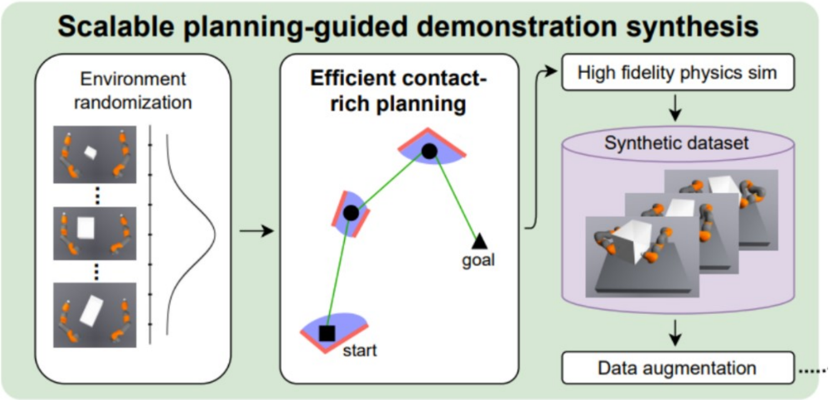
Data Synthesis

After synthesis,

Rollout trajectories in Drake to verify their accuracy

Discard suboptimal or failed trajectories

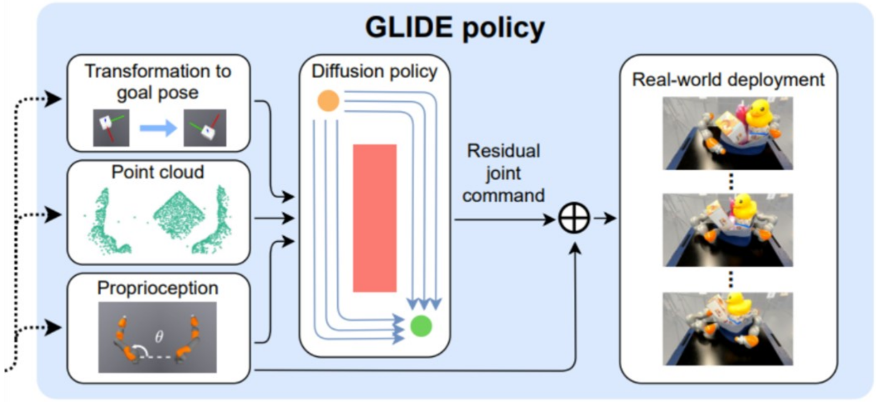
Rebalance trajectories across objects



GLIDE Policy: Feature Extraction

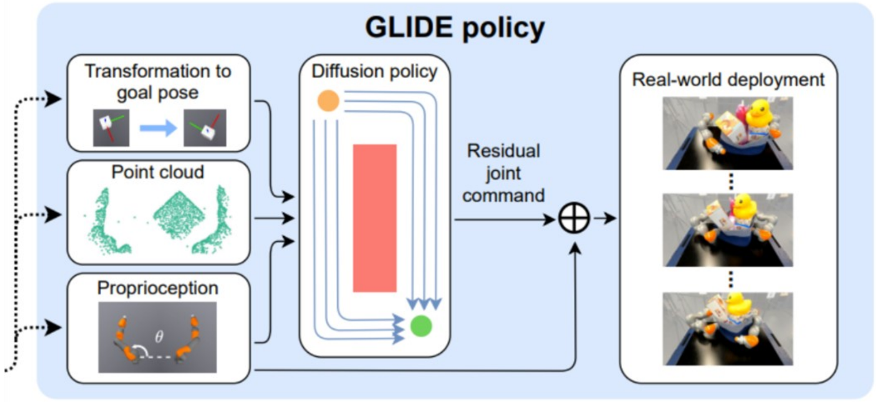
Point Cloud Processing:

- **Clip point clouds** within workspace and remove background
- Generalize to unseen environments
- **Flying Point Augmentation**: randomly add large Gaussian noise to 0.5% of points while training to simulate sensor noise



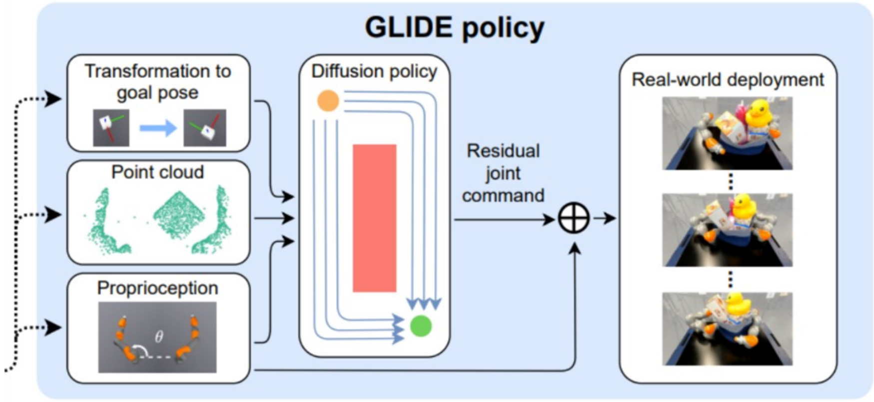
GLIDE Policy: Task Representation

- Don't assume object shape and specify target pose, instead take visual observation and **delta transformation** to the target object pose
- In every timestep, recompute transformation
  - segmenting target object using **open-vocabulary segmentation** algorithm (GroundingDINO)
  - Track keypoints with real time object tracking



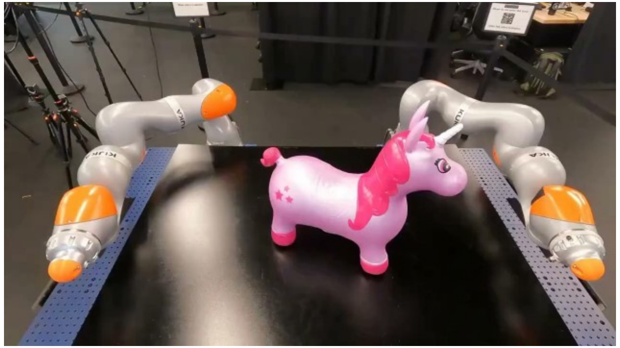
GLIDE Policy: Action Prediction

- Instead of predicting absolute joint angles, prediction head predicts **residuals** (change form current joint angles)
- Much more consistent across different configurations
- **Longer action sequences**: train with T = 64 steps, evaluate with T = 20 steps for smoother trajectories (usually T is 8 steps)



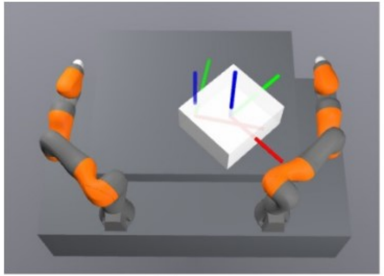
Results

- Trained on 12,000 successful trajectories, 2,000 box primitives of varying size, mass, friction coefficients
- Success evaluation: final object pose is within 10 cm and 0.2 rad of goal



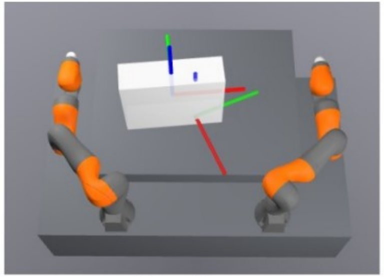
In Distribution

Task (In-Dist Eval)	Planner (Sim)	Policy (Sim)	Policy (Real)
Fixed 45° Rotation	0.337	0.740	0.800
Random Rotation (Easy)	0.227	0.610	0.600
Random Rotation (Medium)	0.141	0.410	0.360
Random Rotation (Hard)	0.099	0.180	0.200
Random Rotation (Overall)	0.156	0.400	0.387



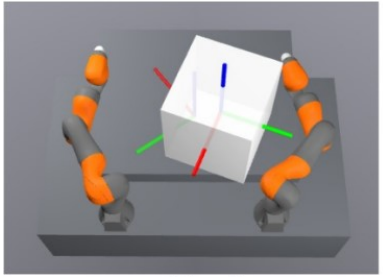
Easy

$|\Delta\theta| \leq 45°$



Medium

$45° < |\Delta\theta| \leq 90°$

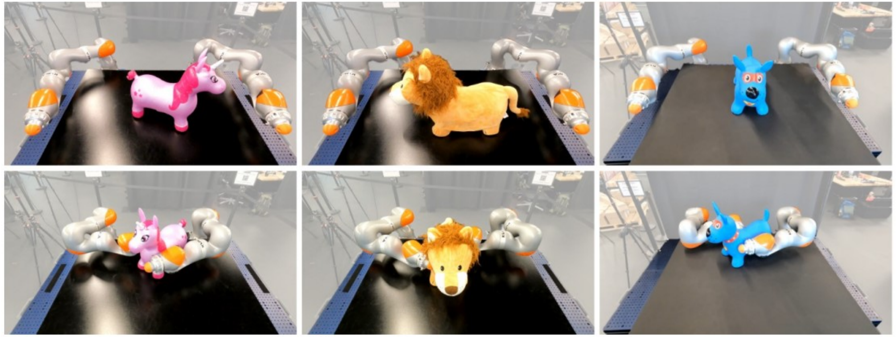


Hard

$90° < |\Delta\theta| \leq 150°$

Out of Distribution

Task (Real OOD Eval)	Empty Containers	Overfilled Containers	Overall
Fixed 45° Rotation	0.688	0.625	0.657
Random Rotation	0.250	0.313	0.282



52% for fixed rotation,
28% for random

Ablations

Residual action and flying points augmentation were very important for sim2real

Residual Action	Flying Point Aug	Success (Sim, In-Dist)	Success (Real, In-Dist)
✓	✓	0.740	0.800
✓	✗	0.750	0.320
✗	✓	0.780	0.520
✗	✗	0.750	0.000

TABLE III: Ablation of different point cloud diffusion policy design choices on our fixed 45° clockwise object rotation task.

Task (Sim, In-Dist Eval)	$T_a = 8$	$T_a = 20$	$T_a = 40$	$T_a = 64$
Fixed 45° Rotation	0.440	0.740	0.760	0.770
Random Rotation	0.270	0.400	0.340	0.200

TABLE IV: Ablation on the number of action prediction steps T_a for diffusion policy evaluation. We use $T_a = 64$ for training.

Too many prediction steps can degrade performance due to lack of real-time feedback

More demonstrations will improve policy

Task (Sim, In-Dist Eval)	500 Demos	2500 Demos	7500 Demos	12000 Demos
Fixed 45° Rotation	0.330	0.690	0.740	0.800
Random Rotation	0.030	0.170	0.330	0.400

TABLE V: Ablation on the number of expert planner demonstrations for diffusion policy learning.

Limitations & Guarantees

- 52% of failure cases in random rotation (hard) the robot gets stuck in poor joint configuration
- Object slippage due to unstable contact
- 16% cases robot exceeds torque limits and squeeze object too hard
- Only used box primitive for training
- Tabletop, rotation tasks only

- Guarantees? None

Thank you