

Paper Presentation:

**Efficient Residual Learning with Mixture-of-Experts
for Universal Dexterous Grasping (ICLR 2025)**

Author: Z. Huang, H. Yuan, Y. Fu, Z. Lu†

Team 3

Seungwan Kang, Hajun Park

| Contents

- 1. Introduction & Motivation**
- 2. Related Work**
- 3. Preliminaries**
- 4. Method**
- 5. Results**
- 6. Conclusion & Limitations**

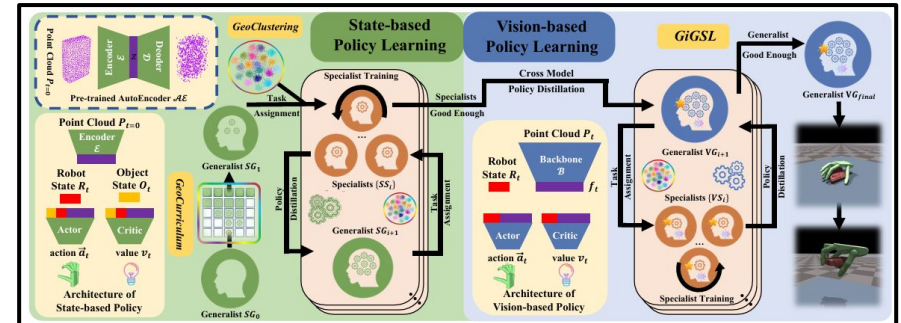
1. Introduction & Motivations

1.1. Problem Definition

1.2. Objectives

1.1. Problem Definition

- Universal dexterous grasping in robotics is important but difficult due to
 - high-DOFs
 - various object geometries
- Several works...
 - UniDexGrasp (CVPR, 2023)
 - UniDexGrasp++ (ICCV, 2023)



Problems?

- ① Complicated curriculum design
- ② Iterative training with huge amount of objects



Training time \uparrow
Overfitting

| 1.2. Objectives

Multi-task dexterous grasping policy with enhanced **efficiency** & **generalization**

2. Related Work

2.1. Dexterous Grasping

2.2. Residual Policy Learning

2.3. Geometry-Unaware policy

2.4. Mixture of Experts

| 2.1. Dexterous Grasping

- Some approaches for **generating target grasp poses**
 - Contact points
 - Affordance maps
 - Natural hand annotations
- **Closed-loop policies** managing the entire trajectory is also important!
 - Imitation learning
 - Dexmv: Imitation learning for dexterous manipulation from human videos (ECCV, 2022)
 - Reinforcement learning (**more scalable!**)
 - Learning complex dexterous manipulation with deep reinforcement learning and demonstrations (RSS, 2018)
 - Unidexgrasp: Universal robotic dexterous grasping via learning diverse proposal generation and goal-conditioned policy (CVPR, 2023)
 - Unidexgrasp++: Improving dexterous grasping policy learning via geometry-aware curriculum and iterative generalist-specialist learning (ICCV, 2023)

| 2.2. Residual Policy Learning

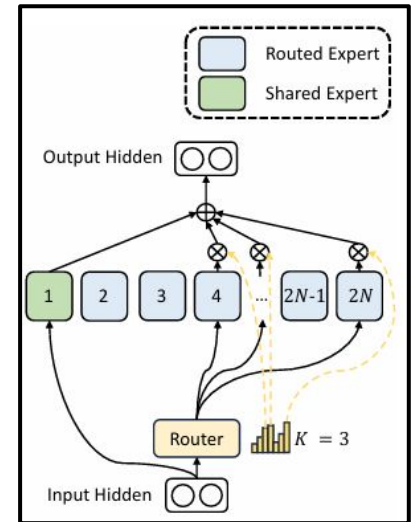
- Residual policy learning
 - **base policy(suboptimal) + residual policy**
 - train a policy to output residual actions using RL
 - when a base policy is provided.
 - **efficient** for learning challenging RL tasks
- Applied examples
 - Manipulation
 - Residual learning from demonstration: Adapting dmps for contact-rich manipulation (RA-L, 2022)
 - Control
 - Residual reinforcement learning for robot control (ICRA, 2019)
 - Sim-to-Real
 - Efficient sim-to-real transfer of contact-rich manipulation skills with online admittance residual learning (CoRL, 2023)
- Let's use this concept to improve **efficiency** of training grasping policy!

| 2.3. Geometry-Unaware Policy

- Dexterous functional grasping (ICRL, 2023)
 - propose blind grasping policy
 - rely on **robot proprioception only**
 - can robustly grasp unseen objects placed close to the palm
- Why this works?
 - policy does not overfit to specific object information
- Let's use this concept for **better generalization** to a broad range of objects.
 - Apply this to the **base policy** in our residual RL!

| 2.4. Mixture of Experts

- Comprises a set of expert models + gating network
 - learns to weight the output of each expert
- Applied examples
 - NLP
 - Mixtral of experts (arxiv, 2024)
 - DeepSeekMoE, DeepSeek-V2 (arxiv, 2024)
 - RL
 - MCP: Learning composable hierarchical control with multiplicative compositional policies (NIPS, 2019)
- Let's use this concept to improve **diversity** of grasping poses!



- image from DeepSeekMoE

3. Preliminaries

3.1. Problem Formulation

3.2. Teacher-Student Framework

| 3.1. Problem Formulation

- Goal: enabling grasping for $\forall w \in \Omega$ (Ω : large object set)
- For each w , task is formulated as POMDP (Partially Observable Markov Decision Process)

\mathcal{O} : Observation space

\mathcal{S} : State space

\mathcal{A} : Action space

$$M^w = \langle \mathcal{O}, \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \mathcal{U} \rangle$$

\mathcal{T} : Transition dynamics: $\mathcal{T}(s_{t+1}|s_t, a_t)$

\mathcal{R} : Reward function: $\mathcal{R}(s_t, a_t)$

\mathcal{U} : Observation emission function: $\mathcal{U}(o_t|s_t)$

At each t

AGENT

- ① observes $o_t \in \mathcal{O}$
- ② takes an action $a_t \in \mathcal{A}$
- ③ receives an reward $r_t = \mathcal{R}(s_t, a_t)$

ENVIRONMENT

- ① Transition to next state
 $\rightarrow s_{t+1} \sim \mathcal{T}(s_{t+1}|s_t, a_t)$

Agent's objective \rightarrow to **maximize the expected return** across all objects

$$\sum_{w \in \Omega} \mathbb{E} \left[\sum_{t=0}^{T-1} \gamma^t r_t \right]$$

T : time limit,
 γ : discount factor

| 3.1. Problem Formulation

- For task learning in simulation,

$$o \in \mathcal{O}$$

- Robot proprioception $J \in \mathbb{R}^{123}$
: wrist position+orientation, joint positions of the hand, fingertip states, forces on fingertip sensors
- Object pose b^p, b^q
: position $b^p \in \mathbb{R}^3$, quaternion $b^q \in \mathbb{R}^4$
- Object code $c^w \in \mathbb{R}^{64}$
: object geometry via a pre-trained PointNet

$$a \in \mathcal{A}$$

- Target joint position(of the hand)
- 6D force(at the wrist)

Our aim: learn a **vision-based policy** $\pi_{\theta}^V(a_t | J_t, p_t, a_{t-1})$ to maximize the expected return across all objects (p : object point cloud in real-world setting)

| 3.1. Problem Formulation

$$r \in \mathcal{R}$$

From the DexGraspNet, grasping proposal is given by $\mathbf{g} = (R, \mathbf{t}, \mathbf{q})$, where

- $R \in \mathbb{SO}(3)$: wrist's relative rotation
- $\mathbf{t} \in \mathbb{R}^3$: position to the object
- $\mathbf{q} \in \mathbb{R}^{22}$: hand's joint positions



$$r_t = r_t^{task} + \alpha \cdot r_t^{proposal}$$
$$r_t^{proposal} = -\|\mathbf{g} - \mathbf{g}_t\|$$

where

- α : hyperparameter,
- r_t^{task} : predefined reward
- $r_t^{proposal}$: reward that penalizes the distance to grasping proposal
- \mathbf{g}_t : current relative pose of the hand (to object)

3.1. Problem Formulation

$$r \in \mathcal{R}$$

r_t^{task} : predefined reward

$$r^{task} = r^{reach} + r^{lift} + r^{move} + r^{bonus}$$

① $r^{reach} = -1.0 * \|X_{obj} - X_{hand}\|_2 - 0.5 \sum \|X_{obj} - X_{finger}\|_2$

: encourages the hand to reach object

② $r^{lift} = \begin{cases} 0.1 + 0.1 * a_z, & \text{if } f_1 = 2 \\ 0, & \text{otherwise} \end{cases}$

: encourages the hand to lift the object

③ $r^{move} = \begin{cases} 0.9 - 2\|X_{obj} - X_{target}\|_2, & \text{if } f_2 = 3 \\ 0, & \text{otherwise} \end{cases}$

: encourages the hand to move the object to the target position

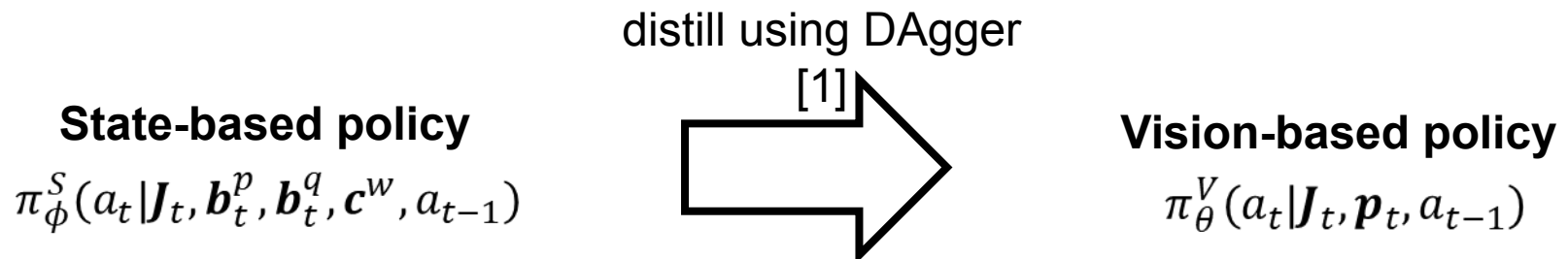
④ $r^{bonus} = \begin{cases} \frac{1}{1+10*d_{obj}}, & \text{if } d_{obj} \leq 0.05 \\ 0, & \text{otherwise} \end{cases}$

: gives an extra reward when the object is close to the target position

| 3.2. Teacher-Student Framework

Why do we use teacher-student framework?

- Directly optimizing the **vision-based policy** using RL is challenging
 - Gradient interference in multi-task RL
 - High dimensionality of point cloud observations
- Teacher-student framework to address these issues
 - UniDexGrasp: curriculum learning for state-based policy
 - UniDexGrasp++: generalist-specialist learning for state-based policy



4. Methods

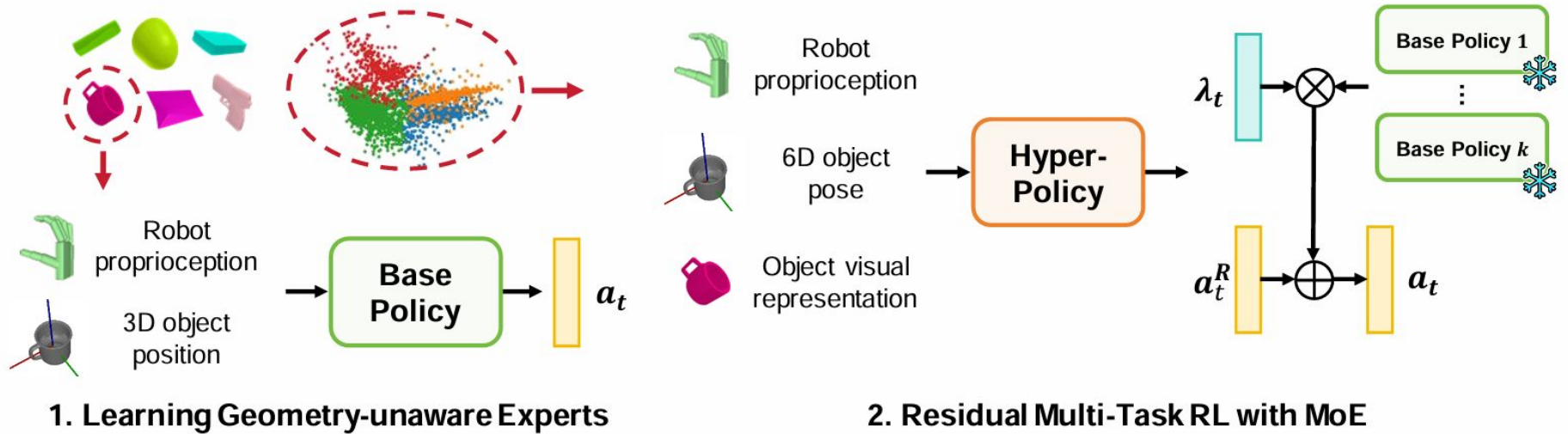
4.1. Overall framework

4.2. Learning Geometry-Unaware Experts

4.3. Residual Multi-Task RL with MoE

4.4. Training Pipeline Summary

| 4.1. Overall framework



1. Base policy (Geometry-unaware Experts)

- generalize range of objects

2. Residual policy + Weights (for each Experts)

- facilitate multi-task learning

| 4.2. Learning Geometry-Unaware Experts

- **Intention:** Using blind grasping policy for better generalization
- **Method:** geometry-agnostic **base policy**
- **Input state:** robot proprioception \mathbf{J} + 3D position of the object

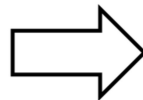
$$\pi_{\psi}^B (a_t | \mathbf{J}_t, \mathbf{b}_t^p, a_{t-1})$$

- **Reward function:** Use hand joint angles only

$$r_t^{proposal} = -\|\mathbf{g} - \mathbf{g}_t\|$$

$$\mathbf{g}_t = (R_t, \mathbf{t}_t, \mathbf{q}_t)$$

reward function in DexGraspNet



$$r_t^{pose} = -\|\mathbf{q} - \mathbf{q}_t\|$$

| 4.3. Residual Multi-Task RL with MoE (1)

- **Intention:** Only using base Policy can't achieve overall success
- **Method:** state-based **residual policy**
- Previously, base policy $a_t^B = \arg \max_{a_t} \pi_{\psi}^B (a_t | \mathbf{J}_t, \mathbf{b}_t^p, a_{t-1})$
- **Additional Input State:** Quaternion and Object code

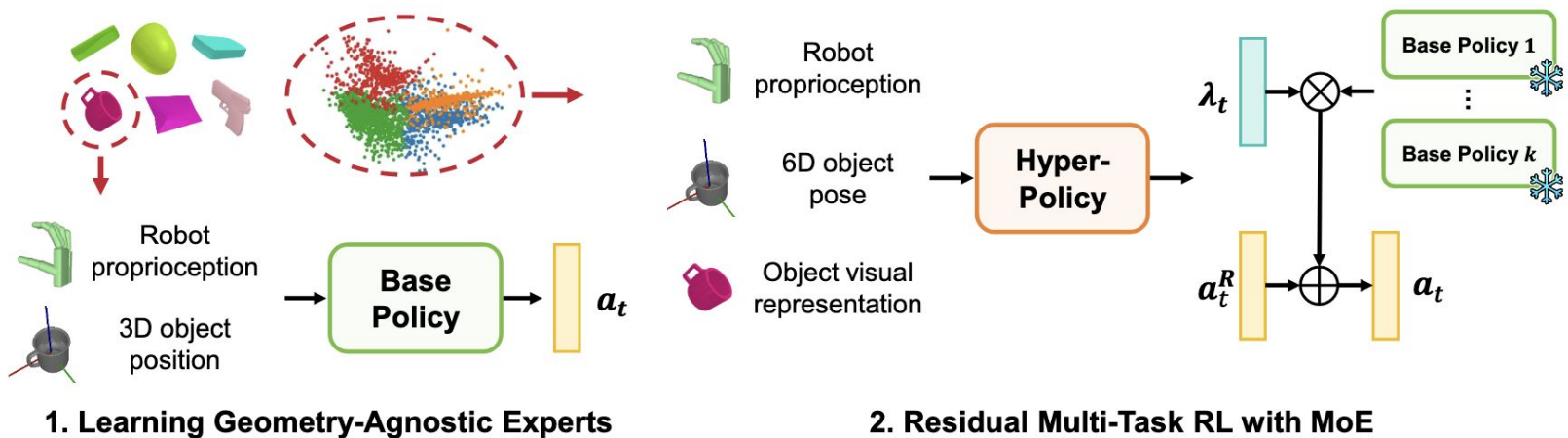
$$a_t^R \sim \pi_{\phi}^R (a_t | \mathbf{J}_t, \mathbf{b}_t^p, \mathbf{b}_t^q, \mathbf{c}^{\omega}, a_{t-1}) \quad \mathbf{b}^q \in \mathbb{R}^4$$
$$a_t = a_t^B + a_t^R \quad \mathbf{c}^{\omega} \in \mathbb{R}^{64}$$

4.3. Residual Multi-Task RL with MoE (2)

- **Intention:** Mixture of Base Policy and Residual Policy
- **Method:** **hyper-policy** (residual policy + weight)

$$\pi_{\phi}^H \left(a_t^R, \lambda_t \mid \mathbf{J}_t, \mathbf{b}_t^p, \mathbf{b}_t^q, \mathbf{c}^{\omega}, a_{t-1} \right)$$

$$a_t = a_t^R + \frac{1}{\|\lambda_t\|} \sum_{i=1}^k \lambda_{t,i} a_{t,i}^B$$



| 4.4. Training Pipeline Summary

- K-Means clustering on the PointNet and generate k clustered dataset
- **Training Base Policies**
 - Use most centered object as dataset for each cluster
 - Train base policy for each clusters
- **Training Hyper-Policy**
 - **First stage:** follow pose provided by dataset $r^{task} + r^{proposal}$
 - **Second stage:** focus terms of task completion r^{task}
- **Vision-based Distillation**
 - teacher-student framework using DAgger

5. Results

5.1. Experiment Setting

5.2. Performance of state-based policy

5.3. Performance of vision-based policy

5.4. Ablation Studies

| 5.1. Experiment Setting

Dataset

- DexGraspNet dataset (3,200 object instances)

Simulation Experiment

- Isaac Gym

Policy Learning

- state-based policy use PPO Algorithm
- vision-based policy use DAgger Algorithm

Detail

- Base policy: 5000 iteration in 4,096 parallel environment
- Hyper policy: 20000 for each training step in 11000 parallel environments
- Vision-based policy: 8000 iteration in 11000 parallel environments

| 5.2. Performance of state-based policy

- **Success rates of state-based policies**
 - Comparing methods: Much higher performance
 - Comparing success rate of train and test: **Solved overfitting**

Method	Train(%)	Test(%)	
		Uns. Obj. Seen Cat.	Uns. Cat.
UniDexGrasp	79.4	74.3	70.8
UniDexGrasp++	87.9	84.3	83.1
ResDex (stage-1)	90.6 \pm 0.6	89.7 \pm 0.8	90.9 \pm 0.1
ResDex (stage-2)	94.6\pm1.6	94.4\pm1.7	95.4\pm1.0

Table 1: Success rates of state-based policies. We evaluate our method on three different random seeds. The hyper-policy is trained with four geometry-agnostic base policies. We present the success rates after each multi-task training stage.




| 5.3. Performance of vision-based policy

- **Success rates of vision-based policies**
 - Vision-based policy achieved better performance
 - **Blind grasping policy** is meaningfully in vision condition

Methods	Train(%)	Test(%)	
		Uns. Obj. Seen Cat.	Uns. Cat.
UniDexGrasp	73.7	68.6	65.1
UniDexGrasp++	85.4	79.6	76.7
ResDex	88.8	88.5	87.2

Table 2: Success rates of vision-based policies.

| 5.4. Ablation Studies (Base Policy - 1)

- **Base policy** introduce two difference with UniDexGrasp
 - **Blind grasping policy** (limited input)
 - **Limited proposal reward** (use hand joint angles only)
- **Test for three case**
 -  **Ours:** Blind grasping policy + Limited Proposal reward
 -  **Full Obs:** Fully observed state + Limited Proposal reward
 -  **Full Pose:** Fully observed state + Full proposal reward

5.4. Ablation Studies (Base Policy - 2)

- Can check high generalization for any object

- **Ours:** Blind grasping policy + Limited Proposal reward
- **Full Obs:** Fully observed state + Limited Proposal reward
- **Full Pose:** Fully observed state + Full proposal reward

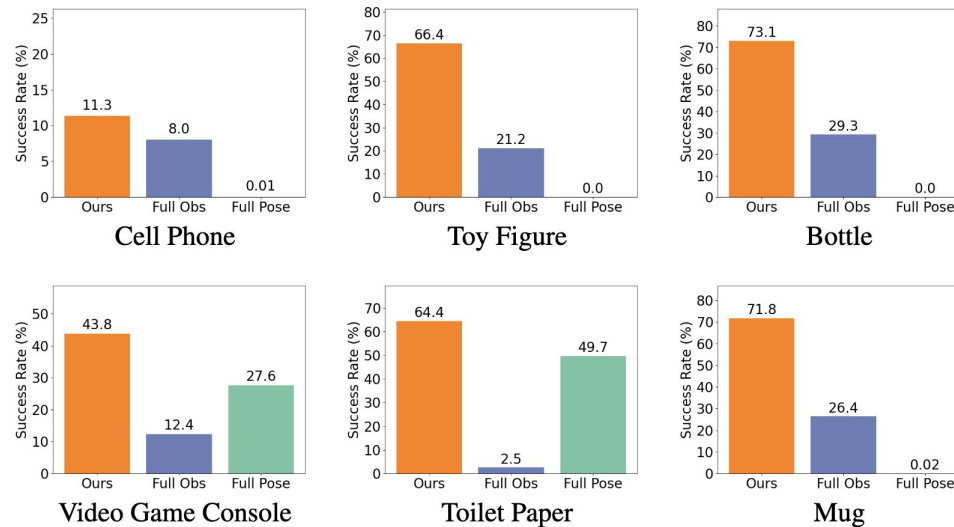


Figure 2: Generalization performance to all objects using policies with different observations and reward, each trained on a single object. Ours: Geometry-Agnostic policy. Full Obs: Policy trained with the complete state-based observations. Full Pose: Policy trained using the reward function that includes the full grasping proposal reward.

| 5.4. Ablation Studies (Hyper Policy - 1)

- Two main questions
 - necessity of **Residual Policy**
 - **Number of Clusters** to divide

Method	$k = 1$	$k = 2$	$k = 3$	$k = 4$	$k = 5$	$k = 6$
MoE	61.4	71.1	79.4	80.3	72.1	81.6
MoE+Res	83.2	82.8	88.1	90.6	87.6	88.7

Table 3: Ablation study on residual reinforcement learning. We assess success rates of policies on the training set. **Method** indicates the number of base policies used. **MoE** shows the results for hyper-policies without residual actions, while **MoE+Res** shows the results for policies that output both normalized weights for MoE and residual actions.

Methods	$k = 1$	$k = 2$	$k = 3$	$k = 4$	$k = 5$	$k = 6$
$D \downarrow$	223.6	174.5	194.3	176.3	204.6	176.1

$$D = - \sum_{t=1}^T r_t^{proposal}$$

Table 4: Quality of grasping poses achieved by different policies. We evaluate the D values of ResDex policies with various k on the test set of unseen objects in unseen categories. The lower D means the better grasping poses achieved.

5.4. Ablation Studies (Hyper Policy - 2)

- Purpose of clustering is to grasp better in different way for each objects

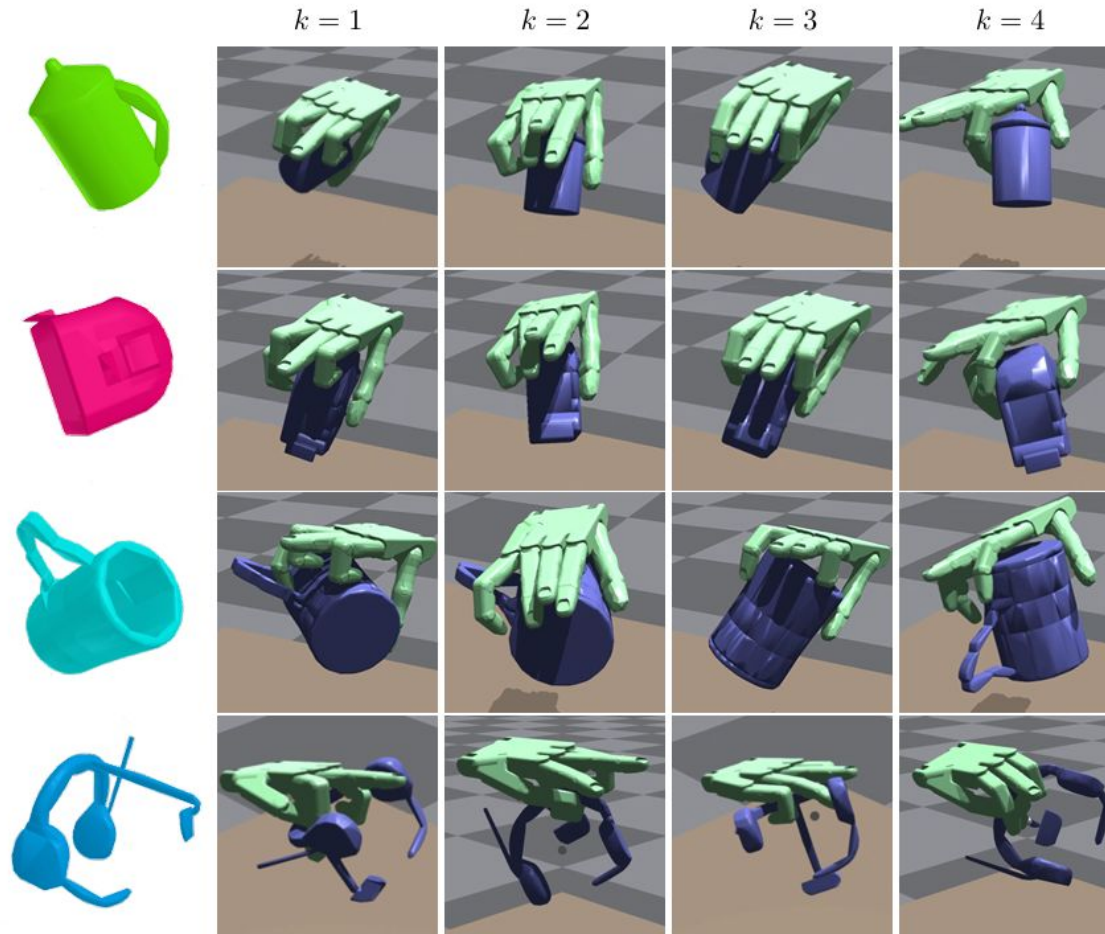


Figure 3: Grasping poses achieved by hyper-policies trained with various numbers of base policies. Each row displays grasping poses for a kettle, tape measure, mug, and headphone, respectively. Columns show hyper-policies trained with 1, 2, 3, and 4 base policies, arranged from left to right.

| 6. Conclusion & Limitation

Improvements

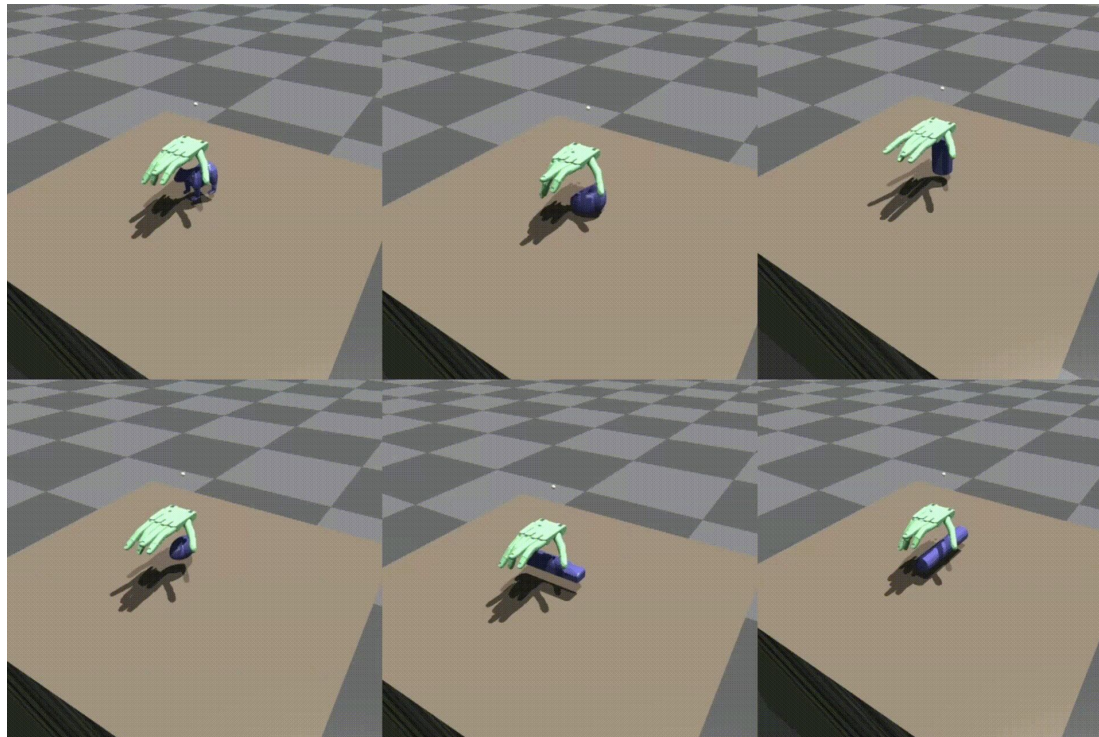
- **Residual Policy Learning Framework**
 - Efficiently trainable model
- **Geometry-Agnostic Base Policy**
 - High generalization
- **Mixture-of-Experts (MoE)**
 - High diversity and good performance

Limitation

- No functional grasping
- No experiment on hardware

Conclusion

- Perform better than previous works
- zero generalization gap



Thank you

| Appendix (Training Details)

Network Architecture We use a MLP architecture which consists of 4 layers (1024, 1024, 512, 512) for base policies and the hyper policy. For the vision-based policy, we use a simplified PointNet (Qi et al., 2017) encoder to represent the object point cloud and apply MLPs with the same hidden layer sizes for the actor and the critic. We use ELU (Clevert, 2015) as the activation function.

Training Device and Training Time All the state-based policies are trained on on a single NVIDIA RTX 4090 GPU. Training a base policy takes about 20 minutes, while training a hyper-policy takes about 11 hours. For the vision-based policy, we train on a single A800 GPU, taking about 16 hours.

Analysis of Training Efficiency To demonstrate the training efficiency of our method compared with UniDexGrasp and UniDexGrasp++, we provide a comparative analysis based on the number of training rounds, as detailed in their papers. UniDexGrasp implements a progressive training strategy — starting with a single object, expanding to several objects within the same category, and finally covering the full training set — requiring three multi-task training stages in practice. UniDexGrasp++ is more complex, involving the training of 20 multi-task policies along with several distillation stages. In contrast, our method only necessitates the training of a single multi-task policy in one trial, using between one to six low-cost, single-task base policies. Our approach is not only simpler but also efficient. As demonstrated in our experiments, our method achieves high success rates even with just one base policy. Table 7 compares the training efficiency of different methods in terms of the number of training rounds.