

Supplementary Material for Rethinking Camera Choice: An Empirical Study on Fisheye Camera Properties in Robotic Manipulation

A. Overview

This supplementary material provides comprehensive implementation details, extended analyses, and additional real-world verifications to substantiate the findings presented in the main paper. The material is organized as follows:

- **Section B (Experiment Setup Details)** focuses on the experiment setup details, covering the visualization of each simulation task, the double camera setup, and the comparison of experimental scenes.
- **Section C (Implementation Details)** provides the specific hyperparameters for training and the definitions of normalized score for real-world experiments.
- **Section D (RQ1: Spatial Localization)** details the probing model and presents additional ablation studies on proprioception.
- **Section E (RQ2: Scene Generalization)** details the scene datasets used in our experiments and provides granular curves for both simulation and real-world tasks.
- **Section F (RQ3: Hardware Generalization)** specifies the camera parameters in simulation and presents the quantitative results of cross-camera experiments in simulation and the real-world.

B. Experiment Setup Details

B.1. Simulation Tasks

We show all the simulation tasks in Fig. S1. They span a wide variety of behaviors including pick-and-place, precise insertion(e.g., Threading and Square), and include long-horizon tasks requiring chaining several behaviors together(e.g., Tool Hang and Mug Cleanup). The detailed configuration, including trajectory counts and data sources for each task, is provided in Tab. S1.

B.2. Double Camera Setup in Simulation

To expand the field of view for more comprehensive scene perception, we added a new camera opposite to the original wrist camera in the simulation environment. The original wrist camera has a position parameter of $\text{pos}=[0.05 \ 0 \ 0]$, and the newly added camera is placed on its opposite side with a position parameter of $\text{pos}=[-0.05 \ 0 \ 0]$. They are symmetrically distributed, thus achieving an effective expansion of the field of view. The visualization of different cameras are shown in Fig. S2

Table S1. **Simulation Tasks Overview.** PH: Proficient-Human datasets; D0: Default reset distribution; D1: Broadened reset distribution.

Task	Trajectory Counts	Data Source	Data Type
Square	200	RoboMimic	PH
Tool Hang	200	RoboMimic	PH
Coffee	500	MimicGen	D1
Threading	500	MimicGen	D0
Assembly	500	MimicGen	D0
Mug Cleanup	500	MimicGen	D0

C. Implementation Details

C.1. Training Hyperparameters

Our policy implementation is built upon the Diffusion Policy framework [4]. Following the protocol established in [18], we ensure rigorous control over hyperparameters to allow for fair comparisons between fisheye and pinhole cameras.

The specific hyperparameters for simulation and real-world experiments are detailed in Tab. S2.

Table S2. **Detailed hyperparameters.** We report the specific settings used for Simulation (Sim) and Real-World (Real) experiments. The hyperparameter values are aligned with [4] and [18].

Config	Simulation	Real-World
Model Architecture		
Visual Backbone	ResNet-18 (No Pretrain)	CLIP ViT-B/16
Pooling Method	Spatial Softmax	Spatial Softmax
Denosing Network	Conditional U-Net1D	Conditional U-Net1D
Action Space	Relative Action	Relative Action
Action Horizon (T_{act})	8	8
Observation Horizon (T_{obs})	2	2
Prediction Horizon (T_{pred})	16	16
Input Data		
Image Resolution	128×128	224×224
Image Preprocessing	Random Crop	Random Crop
Proprioceptive Input	None (State-free)	None (State-free)
Optimization		
Optimizer	AdamW	AdamW
Weight Decay	1×10^{-6}	1×10^{-6}
LR Schedule	Cosine Decay	Cosine Decay
Learning Rate (UNet)	1×10^{-4}	1×10^{-4}
Learning Rate (Encoder)	1×10^{-4}	1×10^{-5}
Batch Size	16	64
Training Epochs	2000	500
EMA Decay	0.75	0.9999

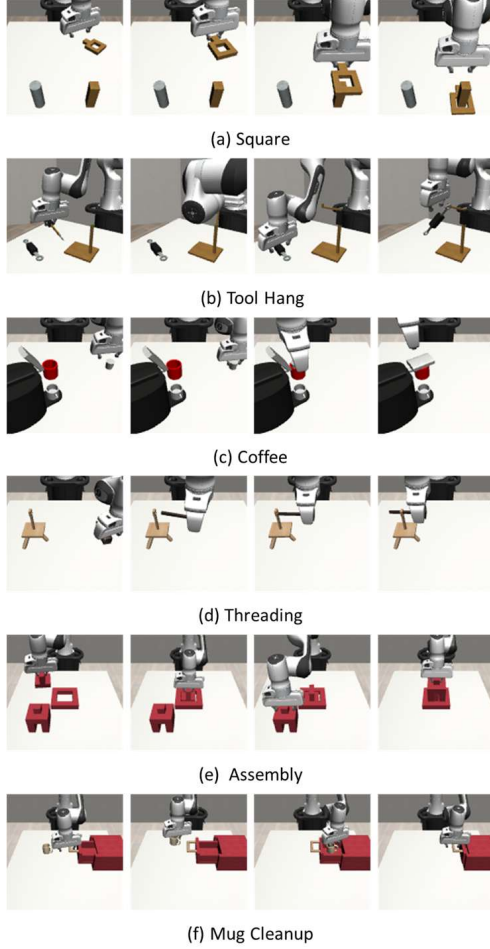


Figure S1. **Examples of simulation tasks.** Each row illustrates sequential snapshots of a distinct task: (a) Square, grasp the yellow square block and insert it into the yellow target slot. (b) Tool Hang, insert the needle-shaped hook and hang the tool. (c) Coffee, place the coffee pod into the machine and close the lid. (d) Threading, grasp the needle and insert it into the pinhole. (e) Assembly, insert two irregular blocks in sequence. (f) Mug Cleanup, open the drawer, place the mug inside, and close it.

C.2. Real-World Task Score Metric Definitions

As described in the main paper, we employ a normalized, multi-stage scoring metric for real-world evaluation. This metric provides a more granular assessment of policy capability than binary success rates. The detailed breakdown for each task is defined below:

- **Pick Cup:** The goal is to pick up a cup and place it upright onto a coaster.
 - *Stage 1 (0.00 pts):* Failed to grasp the cup, or grasped the cup but failed to place it onto the coaster (e.g., dropped it or missed the target).
 - *Stage 2 (0.50 pts):* Successfully grasped and placed the cup onto the coaster, but the cup toppled over (not up-

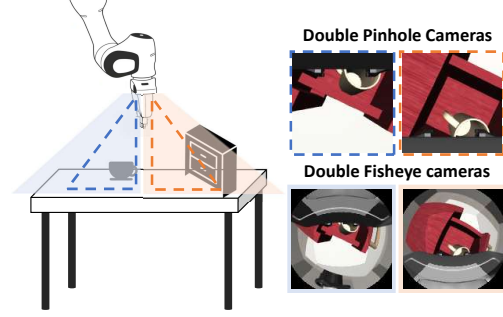


Figure S2. **Visualization of different cameras.** The orange area represents the field of view of the pinhole wrist camera, and the blue area represents the field of view of the newly added symmetrically arranged camera. The upper part shows the images from the pinhole wrist camera, and the lower part shows the images from the double fisheye cameras, with blue and orange borders distinguishing the imaging effects corresponding to their respective fields of view.

right).

- *Stage 3 (1.00 pts):* Successfully grasped and placed the cup onto the coaster, maintaining an upright orientation.
- **Fold Towel:** The goal is to perform two consecutive folds on a towel. The robot must grasp a corner, fold it to the diagonal line, and then grasp the other corner to complete the second fold.
 - *Stage 1 (0.25 pts):* Successfully grasped the first corner and lifted it off the table surface.
 - *Stage 2 (0.50 pts):* Successfully released the gripper and completed the first fold.
 - *Stage 3 (0.75 pts):* Successfully localized and grasped the second corner (after the first fold) and lifted it off the table.
 - *Stage 4 (1.00 pts):* Successfully released the gripper and completed the second fold.
- **Hang Chinese Knot:** The goal is to hang a Chinese knot onto a designated hook on a stand. Since the initial grasping phase is successfully completed by most baselines, we do not include it as a scoring criterion. We focus solely on the precise placement required to secure the knot.
 - *Stage 1 (0.00 pts):* Failed to hang the knot onto the hook (e.g., dropped midway or missed the hook).
 - *Stage 2 (1.00 pts):* Successfully moved the knot to the target location and secured the knot onto the hook.

For each evaluation setup, we conduct 20 trials and report the **cumulative score** (sum of scores across all trials).

D. Additional Experiments for RQ1 (Spatial Localization)

In this section, we first show the scenes in RQ1 simulation experiments (Sec. D.1). Then we elaborate on the model de-

tails for probing the spatial awareness of the visual encoders (Sec. D.2). Additionally, to further validate our findings on spatial localization, we evaluate the policy’s performance with proprioceptive input (Sec. D.3 and Sec. D.4) and third-view input (Sec. D.5).

D.1. Experimental Scenes in Simulation

The Fig. S3 presents a scene example from the Tool Hang task in the RQ1 experiment, comparing visualization effects across different cameras including third-view camera, pinhole camera, and fisheye camera under two distinct environmental settings: a poor scene (single dark scene) and a rich scene (scene with diverse elements).

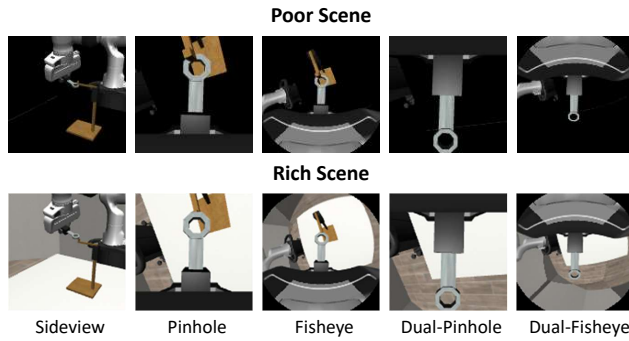


Figure S3. **Scene comparison in RQ1.** Visualization of sideview, pinhole, fisheye, dual-pinhole, and dual-fisheye cameras in poor (single dark) vs. rich (diverse elements) scenes.

D.2. Implementation Details: Proprioception Prediction Task

Methodology. In the main paper, we demonstrated that fisheye-based policies achieve lower proprioception prediction errors in real-world tasks. Here, we provide the detailed experimental setup for this probing task. To explicitly quantify the spatial information captured by the visual representations, we fine-tune the visual encoder on a proprioception prediction task. Specifically, we extract the visual encoder (CLIP ViT [33]) from the learned manipulation policy and attach a lightweight Multi-Layer Perceptron (MLP) head. The MLP consists of two fully connected layers with 256 hidden units and ReLU activations. We fine-tune the entire network (encoder + MLP head) to regress the robot’s end-effector pose (3D position and quaternion orientation) using Mean Squared Error (MSE) loss. To rigorously evaluate the generalization of the learned spatial representations, we do not simply split the training data. Instead, we collected a dedicated **test set comprising 30 additional trajectories** for each task setup. These trajectories were collected under the exact same environmental settings as the training data but were kept strictly held-out during the training phase.

D.3. Ablation Study: Simulation Policy Performance with Proprioception

Motivation. As detailed in Section 3.3, we deliberately excluded proprioceptive state (e.g., end-effector pose and joint positions) from the policy input to rigorously isolate and evaluate the visual spatial localization capabilities of different camera models (RQ1). However, in practical robotic applications, proprioception is often available. In this section, we first conduct an ablation study in simulation to investigate how the addition of proprioceptive state affects policies trained with Pinhole and Fisheye cameras. All experiments in this section are conducted under the “Feature-Rich” background setting to ensure fair comparison.

Simulation Experimental Setup. All experiments in this section are conducted under the “Feature-Rich” background setting to ensure a fair comparison. We evaluate performance across six simulation tasks. For brevity in Tab. S3, we use the following abbreviations: **Tool** (Tool Hang), **Sqr** (Square), **Cof** (Coffee), **Thrd** (Threading), **Asm** (Assembly) and **Mug** (Mug Cleanup). We compare two wrist-camera configurations:

- **Single:** The robot is equipped with a single wrist-mounted camera.
- **Double:** The robot is equipped with two wrist-mounted cameras (providing different views) to reduce occlusion.

Results and Analysis. The quantitative results are summarized in Tab. S3. The **Average** column clearly illustrates the divergent reliance on proprioception between the two camera settings:

1. **Pinhole Sensitivity:** Pinhole-based policies suffer a catastrophic performance drop when proprioception is removed. For the *Single Pinhole* configuration, the mean success rate plummets from 0.62 to 0.34 (a drop of **45%**). For the *double Pinhole* configuration, the mean success rate plummets from 0.64 to 0.45 (a drop of **30%**). This confirms that without the explicit guidance of robot state, the narrow FoV struggles to maintain consistent localization.
2. **Fisheye Robustness:** In sharp contrast, Fisheye-based policies exhibit remarkable robustness. The *Single Fisheye* configuration maintains a high mean success rate (dropping only slightly from 0.75 to **0.66**), and the *Double Fisheye* setup sees a negligible decline (0.81 to **0.75**). This empirically proves that the fisheye’s wide contextual view implicitly encodes the robot’s spatial relationship with the environment effectively, rendering explicit state input largely redundant.

D.4. Ablation Study: Policy Performance with Proprioception

Real-World Experimental Setup. To validate our simulation findings in the real-world, we conducted the same ablation study using the Real-World setup described in the main

Table S3. **Ablation study on proprioception input in simulation.** We compare the success rates between (*w/ State*) and without (*w/o State*) proprioceptive input. Values in (·) indicate the performance drop (or gain) relative to the baseline *w/ State* performance. Pinhole cameras show a sharp relative decline (e.g., -41% performance drop on average), whereas Fisheye cameras maintain robust performance (e.g., only -6% drop on average).

Experimental Factors		Simulation Task Success Rate						Average
CAMERA	STATE	SQUARE	TOOL_HANG	COFFEE	THREADING	ASSEMBLY	MUG_CLEAN	
Pinhole(Single)	w/ State	0.82	0.68	0.38	0.72	0.46	0.66	0.62
	w/o State	0.48 (-41%)	0.56 (-18%)	0.34 (-11%)	0.18 (-75%)	0.12 (-74%)	0.38 (-42%)	0.34 (-45%)
Fisheye(Single)	w/ State	0.86	0.88	0.88	0.72	0.58	0.60	0.75
	w/o State	0.74 (-14%)	0.84 (-5%)	0.76 (-14%)	0.56 (-22%)	0.48 (-17%)	0.60 (0%)	0.66 (-12%)
Pinhole(Double)	w/ State	0.92	0.54	0.38	0.76	0.58	0.66	0.64
	w/o State	0.70 (-24%)	0.34 (-37%)	0.36 (-5%)	0.38 (-50%)	0.34 (-41%)	0.56 (-15%)	0.45 (-30%)
Fisheye(Double)	w/ State	0.94	0.88	0.88	0.78	0.56	0.80	0.81
	w/o State	0.88 (-6%)	0.88 (0%)	0.86 (-2%)	0.66 (-15%)	0.44 (-21%)	0.80 (0%)	0.75 (-7%)

paper. We utilized the “Feature-Rich” (Changeable Background with rich textures) setting to maximize the potential for visual feature extraction. We compare the Normalized Score of the policy with and without proprioception across three real-world tasks: **Pick Cup**, **Fold Towel**, and **Hang Chinese Knot**.

Table S4. **Real-World ablation on proprioception.** We report the Normalized Score in the feature-rich setting. Values in (·) indicate the performance drop relative to the baseline *w/ State* performance. Notably, the Fisheye policy without proprioception (0.67) outperforms the Pinhole policy even with proprioception (0.52).

Experimental Factors		Real-World Normalized Score			Average
CAMERA	STATE	PICK CUP	FOLD TOWEL	HANG KNOT	
Pinhole	w/ State	0.75	0.37	0.45	0.52
	w/o State	0.65 (-13%)	0.32 (-14%)	0.15 (-67%)	0.37 (-29%)
Fisheye	w/ State	0.98	0.92	0.70	0.87
	w/o State	0.80 (-18%)	0.70 (-24%)	0.50 (-29%)	0.67 (-23%)

Results. The real-world results, presented in Tab. S4, reveal an even more pronounced advantage for fisheye cameras compared to simulation:

1. **Superior Spatial Localization:** The most critical metric is the performance *without* proprioception (*w/o State*), which represents the camera’s pure visual localization capability. Fisheye cameras significantly outperform Pinhole cameras in this regime. For instance, in the challenging deformable object task (*Fold Towel*), the Fisheye policy achieves a score of **0.70** without state, whereas the Pinhole policy struggles at **0.32**. On average, the Fisheye camera achieves a mean score of **0.67** using only vision, nearly doubling the Pinhole camera’s mean of **0.37**.
2. **Reduced State Dependency:** While adding proprioception improves performance for both cameras (likely due to the inherent noise and dynamics of the real world), Pinhole cameras are far more dependent on it. In the

Hang Chinese Knot task, the Pinhole policy relies on state to jump from a failing score of 0.15 to 0.45. In contrast, the Fisheye policy already starts at a strong baseline of 0.50 purely from vision.

Summary of Ablation. These real-world experiments, consistent with our simulation findings, corroborate our hypothesis: the wide FoV of the fisheye camera captures sufficient global context to enable high-precision manipulation even in the absence of robot state information. Collectively, these results reinforce our conclusion in RQ1 that fisheye cameras inherently provide superior spatial localization capabilities, significantly reducing the dependency on precise robot state input.

D.5. Ablation Study: Impact of Third-view Camera Integration

While the primary study isolates the effects of wrist-mounted cameras by excluding additional sensors, real-world robotic deployment often incorporates multi-modal setups, such as combining wrist cameras with proprioception or third-person views. To investigate how fisheye cameras behave in these more complex sensing paradigms, we evaluated a “Wrist + Third-person” configuration in simulation. Although recent state-of-the-art frameworks like UMI[5] and GEN-0[46] primarily rely on wrist cameras, our exploration provides informative insights for broader deployment scenarios.

Table S5. Third-view Camera Ablation in Simulation.

Config (Double Cam + 3rd)	Sqr	Tool	Cof	Thrd	Asm	Mug	Mean
Pinhole baseline	0.94	0.78	0.78	0.80	0.56	0.66	0.75
Fisheye (Ours)	0.96	0.84	0.82	0.82	0.66	0.72	0.80
Improvement	+0.02	+0.06	+0.04	+0.02	+0.10	+0.06	+0.05

As summarized in Tab. S5, the fisheye camera consistently maintains a 5% mean performance gain over the pinhole

baseline even when a third-person view is available. Notably, in the high-precision "Asm" (Assembly) task, the fisheye configuration achieves a 10% improvement, demonstrating that the wide-FoV benefits of fisheye cameras are not redundant when global views are present. Instead, fish-eye lenses provide essential local context—such as precise gripper-object relative poses—that fixed global cameras may struggle to capture due to occlusions or limited resolution. This consistent gain proves that the advantages of fisheye cameras identified in our study are robust and carry over to more comprehensive sensor suites, further justifying their adoption in future generalist robot policies.

E. Additional Experiments for RQ2 (Scene Generalization)

E.1. Visualization of Environmental Diversity

Motivation. In the main paper, we established that the wide FoV of fisheye cameras significantly enhances spatial localization, particularly in feature-rich environments (RQ1), and that this capability scales with scene diversity (RQ2). To validate that our experimental setup provides a rigorous assessment of generalization rather than simple memorization, we provide both a qualitative visualization and a quantitative distribution analysis of the background datasets.

Visual Setup. Fig. S4 visualizes the diverse textures employed in our study. When constructing datasets with varying numbers of scenes (N), A critical aspect of our Scene Generalization experiment is to isolate the benefit of environmental diversity from the benefit of increased data scale. Therefore, unlike prior work that scales up data volume [18], we employ a *Fixed Total Data Volume* protocol.

- **Simulation Environments:** The textures were sourced from the MimicLab [40] asset library for the substitution of the original scene materials. These range from geometric patterns to natural materials, introducing high-frequency visual features that challenge the encoder. A set of 32 textures was utilized for training, while a separate set of 5 previously unseen backgrounds was utilized for testing.

We employ Coffee tasks with a fixed budget of **500 trajectories** across all experiments. As we scale the number of distinct scenes (N), we cycle through different scene renderings to ensure uniform trajectory distribution across environments. For example, in the $N = 32$ setting, we employ a balanced combination of 20 scenes contributing 16 trajectories each and 12 scenes contributing 15 trajectories each, maintaining the total of 500 trajectories.

- **Real-World Scenes:** We engineered a variable background system using a collection of patterned cloths (e.g., abstract art, grids) to introduce diverse visual appearances. A total of 8 distinct background scenes were con-

structed for training. To evaluate zero-shot generalization, we employed a separate set of 4 previously unseen background scenes for testing.

"Pick Cup" task was tested in real-world. We fix the total training budget at **200 trajectories** for all experiments. When increasing the number of unique scenes (N), we uniformly distribute the trajectory budget across scenes. For instance, in the $N = 1$ setting, we use 200 trajectories from a single scene; in the $N = 8$ setting, we use 25 trajectories from each of the 8 scenes. This ensures that any performance gain is attributable solely to the increased diversity of the visual data, rather than the quantity of demonstrations.

Data Distribution Analysis. To ensure that our train/test split is statistically rigorous and covers the semantic space of possible environments, we employed a data-driven selection strategy rather than arbitrary manual selection. Specifically, we extracted the global semantic features of all candidate scenes using the CLS token of a pre-trained CLIP visual encoder. We then performed K-Means clustering on these embeddings to identify distinct visual clusters.

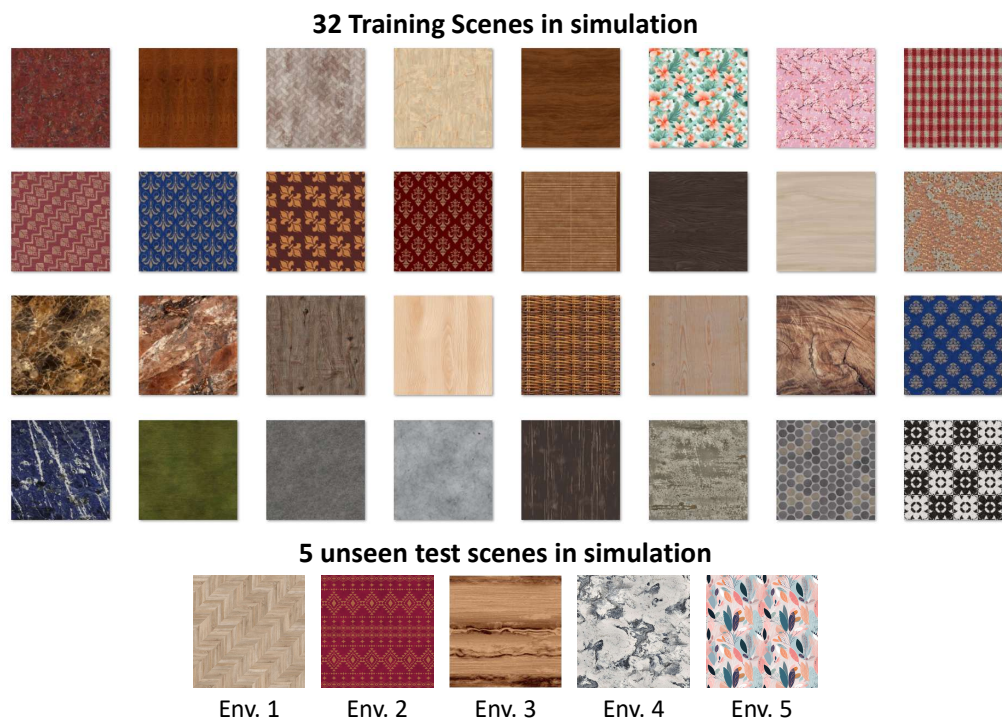
- **Simulation ($K = 8$):** As shown in Fig. S5(a), the simulation textures cluster into 8 distinct groups.
- **Real-World ($K = 4$):** As shown in Fig. S5(b), the real-world scene cluster into 8 distinct groups.

To construct a balanced evaluation protocol, we sampled exactly one representative scene from *each* cluster to serve as the **Held-out Test Set** (indicated by red boxes), while the remaining scenes formed the training set. This method guarantees that the test set is not biased towards any specific texture type and rigorously tests the policy's ability to generalize across the full spectrum of visual distributions. We adopted the settings $N \in \{1, 8, 16, 32\}$ for simulation experiments and $N \in \{1, 2, 4, 6, 8\}$ for real-world experiments, ensuring progressively diverse environmental coverage.

E.2. Per-Scene Generalization Analysis

Motivation. In the main paper, we demonstrated that increasing the diversity of training scenes (N) significantly improves the average zero-shot generalization performance in unseen environments. However, an aggregated mean metric can potentially obscure the variance in difficulty across different test scenes. For instance, a policy might perform exceptionally well on a visually simple background while failing on a more complex one, creating a misleadingly high average. To investigate the consistency of generalization, we provide a granular, disaggregated analysis, plotting performance scaling curves for *each individual* unseen test scene.

Analysis Setup. We decouple the aggregated results from Figure 8 into specific performance trajectories for every held-out environment:

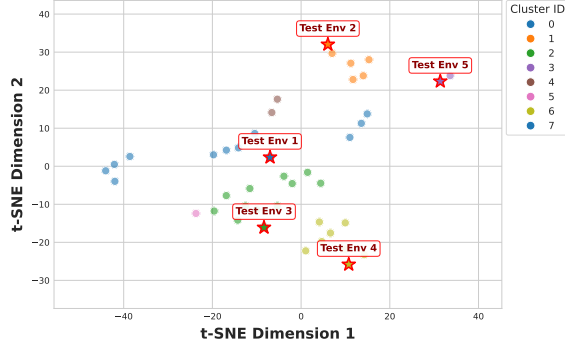


(a) Simulation Texture Library: High-frequency textures providing randomized visual noise in simulation.

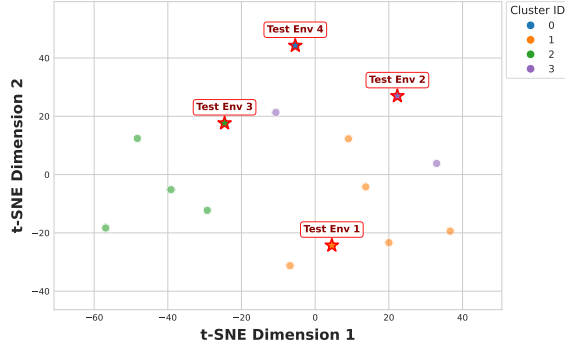


(b) Real-World Backgrounds: Patterned cloths providing diverse visual features in the real-world experiments.

Figure S4. Comprehensive visualization of scene diversity. We explicitly bridge the visual gap between simulation and real-world by ensuring high scene complexity in both domains. **(a)** In simulation, we randomize 32 distinct textures (wood, marble, fabric, tiles) onto background surfaces. **(b)** In the real world, we utilize a changeable background with visually distinct textured cloths to rigorously test spatial localization and generalization.



(a) Simulation Scene Distribution ($K = 8$)



(b) Real-World Scene Distribution ($K = 4$)

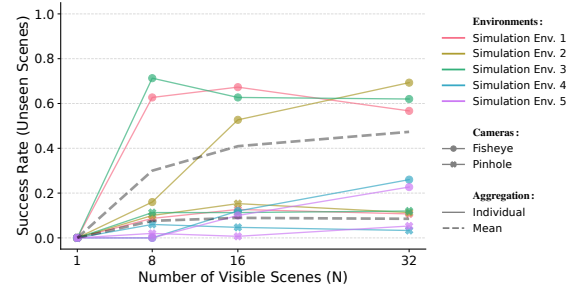
Figure S5. **Visualization of scene distribution in the generalization experiment.** We visualize the scene distributions by clustering the CLIP embeddings of scene images with K-Means. The **Red Stars** highlight the **Held-out Test Scenes**.

- **Simulation Breakdown (5 Curves):** We report the success rate scaling on each of the 5 unseen simulation test scenes (shown in Fig. S6) as the training diversity increases ($N \in \{1, 8, 16, 32\}$).
- **Real-World Breakdown (4 Curves):** We report the normalized score scaling on each of the 4 unseen real-world scenes (shown in Fig. S6)—including the challenging “Starry Night” and “Sunflowers” patterns—as the training diversity increases ($N \in \{1, 2, 4, 6, 8\}$).

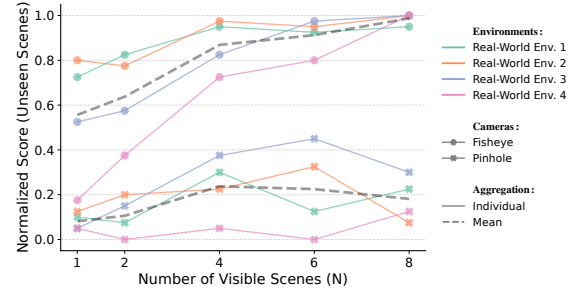
These per-scene visualizations aim to verify whether the fisheye camera’s wide FoV confers a universal generalization advantage that is robust across distinct visual distributions, rather than being specific to certain texture types.

E.3. Analysis of Environmental Complexity and Encoder Selection

To address the impact of environmental variations on policy performance, we provide a quantitative analysis of scene representativeness and justify the selection of visual encoders across different domains. We utilize the average ORB[38] feature density (pts/frame) as a metric to quantify the “visual richness” of each environment. This metric allows us to address two critical questions: **Scene Repre-**



(a) Simulation Task (5 Scenes)



(b) Real-World Task (4 Scenes)

Figure S6. **Generalization Performance on each scene.** We visualize the performance scaling curves for *each individual* held-out test scene to investigate the consistency of generalization. The **Dashed Lines** represent the aggregated mean performance, while solid lines represent specific test environments.

sentativeness: By mapping success rates to specific feature densities, we evaluate whether the advantages of fisheye cameras persist in typical real-world environments rather than being confined to extreme cases (Tab. S6). **Encoder Selection:** Feature density provides a technical justification for our use of different visual encoders across domains. It reveals a vast complexity gap between simulation and real-world environments, which necessitates scaling the model’s representation capacity—moving from ResNet-18 in sparse simulation to CLIP in the feature-rich real world—to effectively process the captured context (Tab. S7).

Scene Representativeness and Texture Sensitivity: As summarized in Tab. S6, we evaluated the policies in real-world environments with varying degrees of visual texture. Even in “Typical” settings, such as a standard wooden desk with median feature density (2111.43 ± 341.87), the fisheye policy consistently outperforms the pinhole baseline by a significant margin (+0.4250). While the performance gain is most pronounced in highly textured environments (+0.8062), the robust success in low-texture scenes confirms that our findings generalize to practical, everyday deployment settings.

Visual Encoder Choice: The choice of visual encoders

Table S6. Performance in Different Real-world Environments.

Typical Test Scene (Real-world Variations)	Feature Density (ORB[38] pts/frame)	Score		Fisheye Improvement
		Pinhole	Fisheye	
Textureless (Poor)	1299.35 ± 176.81 (Low)	0.1250	0.5250	+0.4000
Wooden Desk (Typical)	2111.43 ± 341.87 (Median)	0.1250	0.5500	+0.4250
Highly Textured (Rich)	3574.99 ± 102.54 (High)	0.1813	0.9875	+0.8062

(ResNet-18 for simulation and CLIP for real-world) stems from the vast discrepancy in visual complexity between domains. As quantified in Tab. S7, real-world scenes exhibit a feature density nearly 13 times higher than our simulated environments (3574.99 vs. 268.77). While a lightweight ResNet-18 suffices for processing visually sparse simulation data, the high-density information captured by fisheye lenses in the real world necessitates the robust representation capabilities of CLIP. Crucially, as shown in the ablation results, the fisheye configuration consistently outperforms the pinhole baseline across both domains, regardless of the encoder used, demonstrating that the benefits of a wide Field of View (FoV) are independent of the specific neural architecture.

Table S7. Ablation Study on Visual Encoders.

Domain Feat. Density(ORB pts/frame)	Simulation (Success Rate)		Real-World (Mean Score)	
	268.77 \pm 30.27		3574.99 \pm 102.54	
Encoder	ResNet-18	CLIP	ResNet-18	CLIP
Pinhole	0.4467	0.5333	0.4250	0.7000
Fisheye	0.7533	0.77	0.7000	0.8875

F. Additional Experiments for RQ3 (Hardware Generalization)

In this section, we provide a comprehensive analysis of cross-camera generalization across five subsections. We first detail the experimental protocols (Sec. F.1). To uncover the underlying causes of cross-camera failure, we perform scale sensitivity analyses in both simulation (Sec. F.2) and real-world settings (Sec. F.3). Leveraging insights from these analyses, we demonstrate the effectiveness of Random Scale Augmentation in mitigating scale overfitting within a simulated environment (Sec. F.4). Finally, we present extensive zero-shot real-world verification across diverse physical camera (Sec. F.5), confirming the practical robustness of our proposed approach.

F.1. Simulation Protocols for Cross-Camera Evaluation

Camera Modeling. To rigorously evaluate the zero-shot hardware generalization, we established “seen” camera configurations for training and prepared a diverse set of “unseen” configurations for testing. After acquiring the

equirectangular image, we can simulate various fisheye effects by applying projection models with different distortion parameters. As shown in Tab. S8, we employ different fisheye models with configured parameters to mimic the geometric domain shifts encountered when changing lenses (e.g., switching from a wide-angle fisheye lens to a narrower one). The Fig. S7 shows the visualization effects of simulations with different parameters.

The specific models used are as follows:

- **Extended Unified Camera Model (EUCM):** This model is an extension of the Unified Camera Model (UCM). By introducing two parameters, it improves the accuracy for wide field-of-view lenses. Its parameters are:
 - **f**: Focal length, controlling the zoom level of the image.
 - **a₋**: Shape parameter with a range of (0, 1], controlling the shape of the projection curve.
 - **b₋**: Distortion parameter, adjusting the extent of non-linear distortion.
- **Double Sphere Camera Model (DS):** This model describes light paths through a combination of two spheres, effectively modeling the projection geometry for large field-of-view cameras. Its parameters are:
 - **f**: Focal length, controlling the zoom level of the image.
 - **a₋**: Blending parameter, controlling the mixing ratio between the two sphere models.
 - **xi₋**: Distortion parameter, representing the offset between the centers of the two spheres.

Table S8. Simulation camera parameters for RQ3. We define one seen configuration for training and distinct configurations for zero-shot evaluation. Variations in focal length and distortion simulate significant geometric shifts.

Config Name	Method	Focal Length	Distortion	Scale
Seen Param	EUCM	45	$a_- : 0.4$ $b_- : 2.0$	0.9
Param 1	EUCM	60	$a_- : 0.5$ $b_- : 2.0$	1.0
Param 2	DS	50	$a_- : 0.5$ $xi_- : 0.1$	1.0
Param 3	EUCM	45	$a_- : 0.4$ $b_- : 2.0$	1.0
Param 4	EUCM	45	$a_- : 0.4$ $b_- : 2.5$	1.0
Param 5	EUCM	35	$a_- : 0.4$ $b_- : 1.2$	1.0

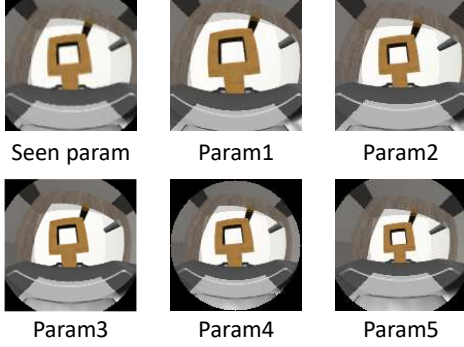


Figure S7. **Simulated fisheye effects using projection models with different distortion parameters.** Each subfigure demonstrates the visual output when applying a distinct set of parameters: “Seen param” represents a baseline parameter set, while “Param1” to “Param5” illustrate variations in distortion intensity, field of view, and other projection characteristics, showcasing how different parameter configurations alter the fisheye rendering.

F.2. Mechanism Analysis: Scale Overfitting vs. Scale Invariance

Hypothesis. In the main paper, we hypothesized that the primary failure mode for cross-camera transfer is “**Scale Overfitting.**” Since different lens intrinsics project objects at different sizes (e.g., a wider FoV makes objects appear smaller), a standard policy overfits to the absolute pixel scale of objects in the training set. When the lens changes, this absolute scale prior breaks, leading to failure.

Quantitative Analysis. To verify this, we visualize the performances for all six tasks in simulation as shown in Fig. S8.

- **Baseline Failure (The “Peaked” Curve):** The policy trained with standard augmentation (Blue Line) performs well *only* on the “Seen Param” and “Param 2”, which has a similar visual scale. Performance collapses on configurations that introduce significant scale shifts (e.g., Param 4 and Param 5), confirming that the policy relies heavily on specific geometric cues from the training lens.
- **RSA Success (The “Plateau” Curve):** In contrast, the policy trained with our **Random Scale Augmentation (RSA)** (Orange Line) maintains a high success rate across a wide range of parameters. RSA effectively forces the network to learn **scale-invariant features** (e.g., relative spatial relationships) rather than memorizing absolute pixel sizes.

F.3. Real-World Cross-Camera Verification

To verify our hypothesis in simulation, we conducted a scale sensitivity analysis on the real robot. Since physically swapping sensor modules to precisely control intrinsic parameters is impractical, we simulate geometric domain shifts by applying varying center-crop scale factors

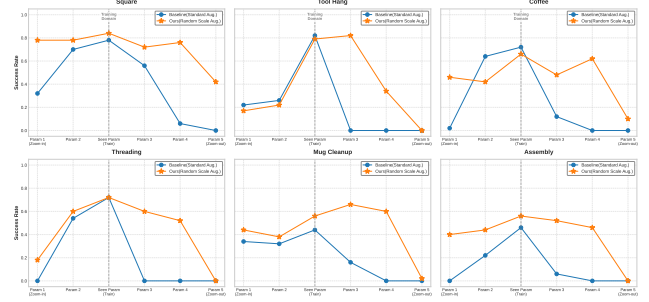


Figure S8. **Cross-Camera generalization performance in simulation for each task.** We evaluate the zero-shot camera-transfer performance of policies trained on a single “Seen Param” configuration across several unseen camera settings. The **Baseline (Blue)** exhibits a sharp performance drop as camera parameters deviate from the training domain, indicating severe overfitting to absolute object scales. In contrast, our **RSA method (Orange)** demonstrates a broad generalization plateau, maintaining robust performance across a wide range of unseen camera parameters.

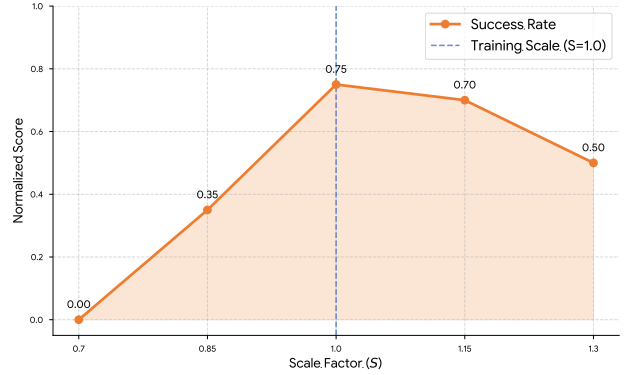


Figure S9. **Real-world Scale Sensitivity Analysis.** To investigate the root cause of cross-camera transfer failure, we simulate geometric domain shifts by applying center crops with varying scale factors (S) to the fisheye input. The results show a characteristic “**inverted-V**” performance drop: the policy performs robustly near the training scale ($S = 1.0$) but suffers catastrophic degradation as the scale deviates significantly (e.g., 0% success rate at $S = 0.7$). This confirms that the standard policy is highly sensitive to absolute object scale, validating the necessity of our Random Scale Augmentation (RSA) strategy.

(S) to the input images during inference, effectively mimicking changes in focal length and FoV. This operation effectively mimics the “Zoom In” (narrower FoV) and “Zoom Out” (wider FoV) effects caused by changing lens parameters. We evaluated the policy’s performance across a scale range of $S \in [0.7, 1.3]$, where $S = 1.0$ represents the training distribution.

Quantitative Analysis: Sensitivity to Scale. As illustrated in Fig. S9 (Scale Sensitivity Analysis), the baseline policy exhibits a characteristic “**inverted-V**” performance curve.

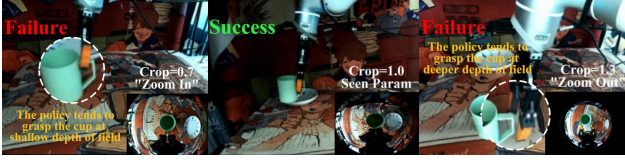


Figure S10. **Qualitative Visualization of Scale-Induced Failures.** We visualize the specific failure modes of the baseline policy under geometric scale shifts (simulated via center cropping). **(Left)** When zoomed in (Crop=0.7), the object appears larger, misleading the policy to perceive it as closer; consequently, the robot attempts to grasp at a *shallower* depth (undershooting the target). **(Right)** When zoomed out (Crop=1.3), the object appears smaller, causing the policy to perceive it as farther away; this leads to grasping at a *deeper* depth (overshooting or colliding with the target). **(Middle)** Accurate manipulation is only achieved at the training scale (Crop=1.0). These behaviors explicitly confirm that the cross-camera failure stems from the policy’s overfitting to the absolute pixel scale of objects. **Please see the video in the supplementary files for more details.**

While the policy maintains robust performance near the training scale ($S = 1.0$, score 0.75), it suffers catastrophic degradation as the scale deviates. Notably, a “Zoom In” operation ($S = 0.7$) causes the success rate to plummet to **0.0**, while a “Zoom Out” ($S = 1.3$) drops the performance to **0.5**. This sharp decline confirms that the standard fisheye policy is highly sensitive to absolute object scale, corroborating our simulation hypothesis that scale overfitting is the primary bottleneck for cross-camera transfer.

Qualitative Analysis: The Depth-Scale Ambiguity. To understand the failure mechanism, we visualize specific rollout behaviors in Fig. S10. The results reveal a distinct correlation between visual scale and depth estimation errors:

- **Underestimation of Depth (Zoom In):** At $S = 0.7$, the object appears significantly larger in the image frame. The policy misinterprets this visual cue as the object being *closer* than it actually is, resulting in a grasp attempt at a **shallower depth** (undershooting the target).
- **Overestimation of Depth (Zoom Out):** Conversely, at $S = 1.3$, the object appears smaller. The policy perceives it as being *farther away*, leading to a grasp at a **deeper depth** (often colliding with or overshooting the target).

These findings provide empirical evidence that, in the absence of scale-invariant training (e.g., RSA), fisheye-based policies rely heavily on absolute pixel size for spatial reasoning, making direct cross-camera transfer inherently difficult.

F.4. Effectiveness of Random Scale Augmentation

Building upon the scale-shift protocols defined in Sec. F.3, we evaluate the effectiveness of Random Scale Augmentation (RSA) in enhancing policy robustness. Our results

demonstrate that RSA provides consistent generalization improvements, serving as a defense against geometric domain shifts that typically cause standard imitation learning policies to fail. The detailed results of our evaluation are presented in Tab. S9, which quantifies the performance of different policies under the varying scale factors.

- **Consistent Generalization Improvements:** Policies trained with RSA demonstrate steady performance gains over the Standard Aug. (Standard Augmentation) baseline across all evaluated scales. For instance, at a scale factor of $S = 0.85$, the RSA-trained Diffusion Policy achieves a score of 0.950, whereas Standard Aug. yields 0.350.
- **Reducing Scale-Induced Performance Degradation:** Standard Augmentation shows sensitivity to scale shifts, particularly under zoom-in ($S = 0.70$), where the Diffusion Policy’s performance reaches zero. In contrast, RSA maintains a score of 0.725 under the same conditions, mitigating failures caused by object magnification.
- **Integration with Large-Scale Models:** When integrated with the $\pi_{0.5}$ architecture, RSA facilitates improved scale invariance. Notably, $\pi_{0.5}$ equipped with RSA maintains a score of 1.000 even under significant zoom-out ($S = 1.30$), a scenario where the Standard Aug. score falls to 0.150.

These results suggest that RSA encourages the visual encoder to prioritize relative spatial relationships—such as the target object’s size relative to the gripper—over absolute pixel scales. This prioritization supports more robust cross-hardware deployment by reducing the impact of lens-specific geometric shifts.

Table S9. Normalized scores for the “Pick and Place” task under simulated scale shifts. We compare **Standard Aug.** (Standard Augmentation) and **RSA** (Random Scale Augmentation) across different parameters (S).

Policy Model	Aug. Strategy	$S = 0.70$ (Zoom-in)	Param 1	$S = 1.0$ (Seen)	$S = 1.15$	$S = 1.30$ (Zoom-out)
Diffusion Policy [4]	Standard Aug.	0.000	0.350	0.750	0.750	0.500
	RSA (Ours)	0.725	0.950	1.000	0.750	0.650
$\pi_{0.5}$ [2]	Standard Aug.	0.375	0.875	1.000	0.600	0.150
	RSA (Ours)	0.900	1.000	1.000	0.975	1.000

F.5. Zero-shot Cross-Camera Validation

To evaluate the practical utility of RSA, we conducted zero-shot transfer experiments between different physical fisheye lenses in the real world. As detailed in Tab. S10, a policy trained on a standard 180° lens was deployed directly onto hardware equipped with Narrow (150°) and Wide (220°) lenses, introducing distinct geometric and scale shifts. To ensure the findings are representative of modern generalist policies, we conduct this validation using the state-of-the-art $\pi_{0.5}$ architecture.

- **Addressing Hardware-Induced Scale Shifts:** The Standard Aug. (Baseline) policy shows an observable performance decline when encountering hardware variations. Specifically, when transitioning to a Wide Lens ($\sim 0.8\times$ scale shift), the baseline score reaches 0.0025.
- **Real-world Generalization:** In contrast, RSA mitigates these hardware-induced shifts across different physical lenses. For the Narrow Lens ($\sim 1.2\times$ scale shift), RSA increases the score from 0.5000 to 0.9500. Even in the more challenging Wide Lens scenario, RSA recovers the performance to 0.6000.
- **Practical Deployment Implications:** These results suggest that RSA is an effective strategy for real-world robotics beyond simulation-based heuristics. By decoupling the policy from absolute pixel scales, RSA facilitates the reuse of existing datasets across diverse camera systems.

Table S10. Real-world Cross-camera Generalization with RSA.

Test Camera	FOV Angle	Induced Scale Shift	Baseline(Standard Aug.)	RSA (Ours)
Seen Camera	180°	1.0× (Seen)	1.0000	1.0000
Narrow Lens	150°	$\sim 1.2\times$ (Zoom In)	0.5000	0.9500
Wide Lens	220°	$\sim 0.8\times$ (Zoom Out)	0.0025	0.6000

In summary, these findings demonstrate that RSA effectively bridges the gap between diverse hardware configurations. By encouraging the learning of scale-invariant features, RSA offers a practical path for deploying vision-based policies across varied robotic platforms without the need for hardware-specific fine-tuning.