

497 A ATK implementation details

498 A.1 Mask network

499 To ensure the selection of the minimal set of N keypoints, we enforce a sparsity penalty on the
500 mask network. We denote the masking distribution over binary masks $m \in \{0, 1\}^M$ for candidate
501 keypoints K as $q_\phi(m | K)$, and the action distribution conditioned on the masked keypoints
502 as $\pi_\theta(a | m \odot K)$. Because sampling $m \sim q_\phi(m | K)$ is discrete and non-differentiable, we
503 employ the Gumbel–softmax relaxation [22] to enable gradient-based optimization. Concretely, a
504 K -dimensional learnable parameters are used as logits for M independent Bernoulli probabilities,
505 with each representing the likelihood of retaining the corresponding keypoint from the input candidate
506 set. We then add Gumbel noise to each logit, divide by a temperature τ , and apply softmax to form a
507 continuous “soft” mask y_M^{Soft} . During the forward pass, we take $\arg \max_i y_i^{\text{Soft}}$, to get a hard one-hot
508 selection, but in the backward pass we use the smooth “soft” mask for gradient calculation. The
509 policy $\pi_\theta(a | m \odot K)$ can be instantiated by any policy class model—e.g., a multi-layer perceptron,
510 a Gaussian policy [45] or a score-based diffusion model [46]. The final objective jointly optimizes
511 the policy $\pi_\theta(a | m \odot K)$ and the masking distribution $q_\phi(m | K)$, yielding a minimal, task-relevant
512 keypoint representation.

513 A.2 Viewpoint robustness

514 “Viewpoint robustness” is another critical metric for gauging how well a policy holds up when the
515 camera’s perspective shifts. It measures the policy’s performance under changed camera viewpoints.
516 In our method, we assume access to both the original and shifted camera intrinsic and extrinsic
517 matrices—a practical assumption given modern computer vision advances(e.g., extrinsic estimation
518 via CtRNet [47]). In our experiments, we utilized a fixed ArUco marker’s coordinate to get the
519 cameras’ extrinsic. We use h_C to find correspondence 2D keypoints in the new camera view and
520 project them back into the original camera frame where the policy was trained. This reprojection
521 compensates for viewpoint shifts, allowing the policy to operate as if the view had not changed.
522 We show that in our video, after changing the camera to three different angles like Figure 8, the
reprojected keypoints still enable the policy to succeed.



Figure 8: Different camera viewpoint

523 B Tasks evaluation procedure

525 B.1 High precision task – shoe lacing:

526 We also evaluate our method on a high-precision manipulation task: inserting a shoelace into a shoe
527 eyelet. The lace is already securely held by the robot, and the task involves inserting it through the
528 eyelet. The training dataset comprises 80 demonstration trajectories, collected with a fixed initial pose
529 of the end effector and randomized shoe positions. This fine-grained task requires high precision, as
530 the shoe eyelet has a diameter of 5 mm while the lace’s radius is approximately 3.2 mm, leaving only a
531 1.7 mm tolerance—significantly tighter than the tolerances typically encountered in standard picking
532 or grasping tasks. We further tested the robustness of our method under challenging conditions,

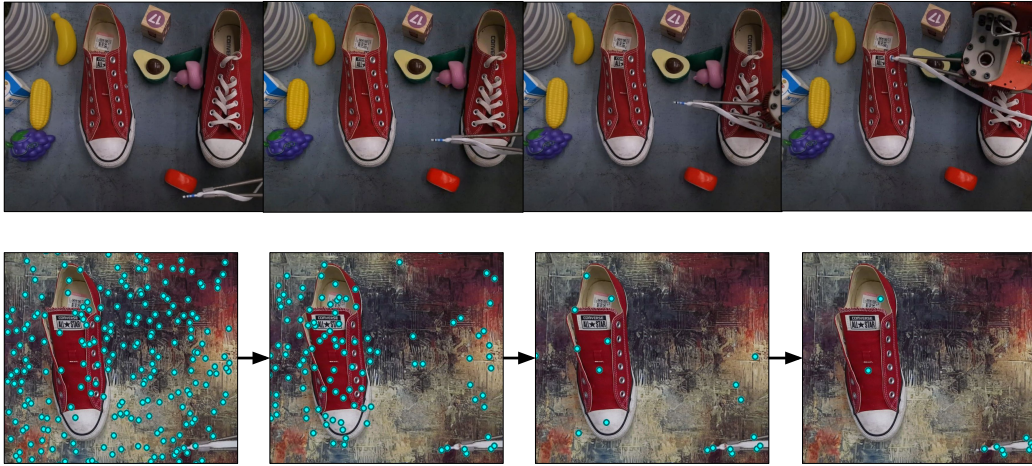


Figure 9: Visualization of the shoe lacing task. The first row shows a successful rollout of the policy performing shoelace insertion with varying backgrounds and distractors. The second row illustrates the distillation process of keypoint filtering.

including varying background textures, random distractors, and changes in lighting. Despite these perturbations, our approach demonstrates high performance in this fine-grained insertion task.

B.2 Evaluation metrics

Random pose (RP): As shown in Figure 10, we show the distribution of the object’s pose during task evaluation.

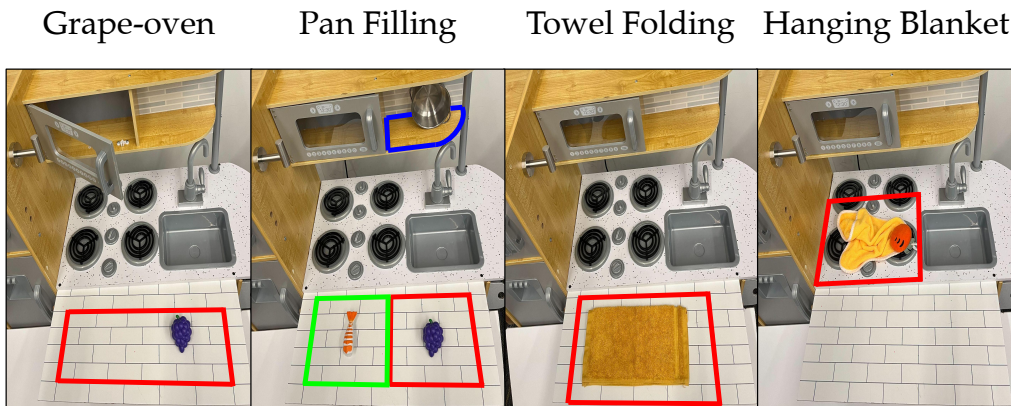


Figure 10: Highlighted regions for different objects during evaluation.

537
538

Random background (RB): As shown in Figure 11, we show different backgrounds used for evaluation in each task

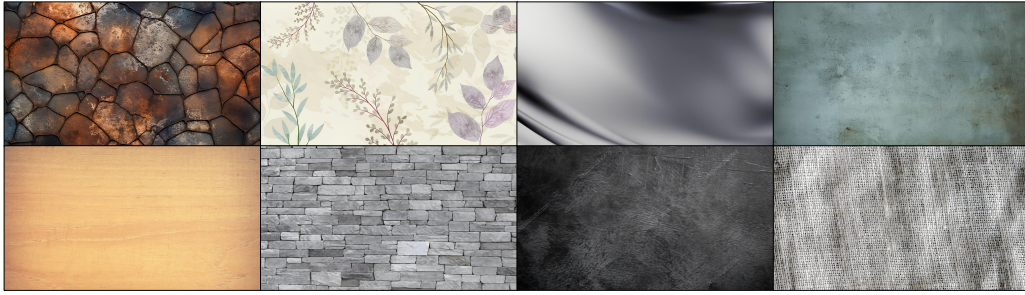


Figure 11: Different backgrounds used for evaluation

541 **Random distractor objects (RO):** As shown in Figure 12, we show distractor objects used for
542 evaluation in each task.



Figure 12: Different Distractor Objects used in evaluation

Light: As shown in Figure 13, we show the different colored lights used for evaluation in each task



Figure 13: Different Light Conditions used for evaluation

544 C Infrastructural setup

Sim-to-Real Setup: We create MuJoCo [48] simulation environment using an iPhone app, Scani-verse, to scan and import the meshes of real-world objects and add joints for articulated objects. We conduct real-world transfer experiments using a 6-DOF Hebi robot arm equipped with chopsticks, following [49]. For RGB and depth streaming, we employ Azure Kinect RGB-D cameras.

Imitation Setup: Imitation learning is purely tested in the real world. We test these methods on the UR5e robot equipped with a Robotiq 2F-145 gripper, running a joint PD controller. This robot is tasked with manipulating various objects in a miniature kitchen. As previously mentioned, for RGB and depth streaming, we employ Azure Kinect RGB-D cameras. We used 80, 50, 50, 30 number of demonstrations to train policies for pan filling, grape-oven, towel folding, hanging blanket respectively.

555 D Detailed evaluation results

556 D.1 Sim-to-real performance

| | Sushi | | | | Glass | | | |
|------------|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|
| | RP | RB | +RO | + Light | RP | RB | +RO | + Light |
| RGB | 0.453±0.262 | 0.076±0.041 | 0.027±0.020 | 0.010±0.014 | 0.253±0.154 | 0.109±0.098 | 0.020±0.021 | 0.000±0.000 |
| Depth | 0.255±0.199 | 0.255±0.199 | 0.020±0.021 | 0.010±0.014 | 0.110±0.001 | 0.110±0.001 | 0.000±0.000 | 0.000±0.000 |
| Pointcloud | 0.277±0.088 | 0.277±0.088 | 0.020±0.021 | 0.000±0.000 | 0.033±0.047 | 0.033±0.047 | 0.000±0.000 | 0.000±0.000 |
| ATK | 0.893±0.073 | 0.893±0.073 | 0.893±0.073 | 0.893±0.073 | 0.933±0.034 | 0.933±0.034 | 0.933±0.034 | 0.933±0.034 |
| | Clock button | | | | Clock turning | | | |
| | RP | RB | +RO | + Light | RP | RB | +RO | + Light |
| RGB | 0.456±0.293 | 0.046±0.017 | 0.013±0.019 | 0.000±0.000 | 0.367±0.205 | 0.093±0.020 | 0.013±0.012 | 0.000±0.000 |
| Depth | 0.290±0.150 | 0.290±0.150 | 0.000±0.000 | 0.000±0.000 | 0.256±0.264 | 0.256±0.264 | 0.000±0.000 | 0.020±0.021 |
| Pointcloud | 0.107±0.056 | 0.107±0.056 | 0.010±0.014 | 0.000±0.000 | 0.077±0.056 | 0.077±0.056 | 0.010±0.014 | 0.010±0.014 |
| ATK | 0.970±0.024 | 0.970±0.024 | 0.970±0.024 | 0.970±0.024 | 0.903±0.028 | 0.903±0.028 | 0.903±0.028 | 0.903±0.028 |

Table 1: **Simulator** Policy Success Rates using *different input modalities* over 3 random seeds. Keypoint-based policies are easier to distill in simulator than other baselines with alternative sensor modalities.

| | Sushi Pick-n-Place | | | | GlassPot Lift | | | | Clock Button Press | | | | Clock Hand Turning | | | | Total |
|------------|--------------------|-------------|-------------|-------------|---------------|-------------|-------------|-------------|--------------------|-------------|-------------|-------------|--------------------|-------------|-------------|-------------|-------------|
| | RP | RB | +RO | + Light | RP | RB | +RO | + Light | RP | RB | +RO | + Light | RP | RB | +RO | + Light | |
| RGB | 0.30 | 0.00 | 0.00 | 0.00 | 0.10 | 0.00 | 0.00 | 0.00 | 0.25 | 0.00 | 0.00 | 0.00 | 0.05 | 0.00 | 0.00 | 0.00 | 0.04 |
| Depth | 0.25 | 0.20 | 0.00 | 0.00 | 0.05 | 0.00 | 0.00 | 0.00 | 0.10 | 0.10 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.04 |
| Pointcloud | 0.10 | 0.10 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.05 | 0.05 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.02 |
| ATK | 0.85 | 0.80 | 0.55 | 0.45 | 0.75 | 0.65 | 0.60 | 0.60 | 0.90 | 0.90 | 0.80 | 0.75 | 0.50 | 0.50 | 0.40 | 0.35 | 0.64 |

Table 2: **Real-world** Policy Success Rates. Varying conditions including RP (random pose), RB (background), RO (distractor object), Light. *ATK* consistently outperforms baseline methods using alternative modalities in sim-to-real transfer.

| | Sushi | | | | Glass | | | |
|--------------|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|
| | RP | RB | +RO | + Light | RP | RB | +RO | + Light |
| FullSet | 0.122±0.057 | 0.053±0.036 | 0.010±0.008 | 0.013±0.012 | 0.311±0.150 | 0.069±0.056 | 0.013±0.012 | 0.013±0.012 |
| RandomSelect | 0.337±0.315 | 0.246±0.360 | 0.233±0.370 | 0.226±0.375 | 0.120±0.082 | 0.031±0.044 | 0.116±0.151 | 0.006±0.009 |
| GPTSelect | 0.032±0.009 | 0.020±0.008 | 0.013±0.005 | 0.006±0.004 | 0.133±0.188 | 0.020±0.028 | 0.010±0.014 | 0.010±0.014 |
| ATK | 0.893±0.073 | 0.893±0.073 | 0.893±0.073 | 0.893±0.073 | 0.933±0.034 | 0.933±0.034 | 0.933±0.034 | 0.933±0.034 |
| | Clock button | | | | Clock turning | | | |
| | RP | RB | +RO | + Light | RP | RB | +RO | + Light |
| FullSet | 0.474±0.317 | 0.126±0.090 | 0.026±0.030 | 0.020±0.016 | 0.253±0.183 | 0.083±0.880 | 0.010±0.014 | 0.010±0.014 |
| RandomSelect | 0.107±0.030 | 0.080±0.045 | 0.036±0.032 | 0.026±0.020 | 0.253±0.166 | 0.076±0.088 | 0.000±0.000 | 0.000±0.000 |
| GPTSelect | 0.913±0.041 | 0.913±0.041 | 0.913±0.041 | 0.913±0.041 | 0.065±0.053 | 0.146±0.179 | 0.077±0.088 | 0.020±0.028 |
| ATK | 0.970±0.024 | 0.970±0.024 | 0.970±0.024 | 0.970±0.024 | 0.903±0.028 | 0.903±0.028 | 0.903±0.028 | 0.903±0.028 |

Table 3: **Simulator** Policy Success rate using **different keypoint selection methods** over 3 random seeds. *ATK* consistently outperforms alternative keypoint selection methods using random sampling or ChatGPT selection.

557 D.2 Imitation learning performance

| | Towel Hanging | | | | Towel Folding | | | | Grape Oven | | | | Pan Filling | | | |
|------------|---------------|-------------|-------------|-------------|---------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
| | RP | RB | +RO | + Light | RP | RB | +RO | + Light | RP | RB | +RO | + Light | RP | RB | +RO | + Light |
| RGB | 0.40 | 0.00 | 0.00 | 0.00 | 0.60 | 0.00 | 0.00 | 0.00 | 0.45 | 0.15 | 0.00 | 0.00 | 0.25 | 0.10 | 0.00 | 0.00 |
| Depth | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.35 | 0.25 | 0.00 | 0.00 | 0.50 | 0.45 | 0.00 | 0.00 |
| Pointcloud | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.02 | 0.02 | 0.00 | 0.00 | 0.01 | 0.01 | 0.00 | 0.00 |
| ATK | 0.85 | 0.85 | 0.80 | 0.60 | 1.00 | 1.00 | 1.00 | 0.70 | 0.80 | 0.80 | 0.75 | 0.65 | 0.60 | 0.55 | 0.40 | 0.25 |

Table 4: **Real World Imitation Learning** Success Rates. Varying conditions including RP (random pose), RB (background), RO (distractor object), Light. *ATK* consistently outperforms baseline methods using alternative modalities.

| | Towel Hanging | | | | Towel Folding | | | | Grape Oven | | | | Pan Filling | | | |
|--------------|---------------|-------------|-------------|-------------|---------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
| | RP | RB | +RO | + Light | RP | RB | +RO | + Light | RP | RB | +RO | + Light | RP | RB | +RO | + Light |
| FullSet | 0.10 | 0.05 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.10 | 0.05 | 0.00 | 0.00 |
| RandomSelect | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| GPTSelect | 0.60 | 0.60 | 0.50 | 0.25 | 1.00 | 0.60 | 0.50 | 0.35 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| ATK | 0.85 | 0.85 | 0.80 | 0.60 | 1.00 | 1.00 | 1.00 | 0.70 | 0.80 | 0.80 | 0.75 | 0.65 | 0.60 | 0.55 | 0.40 | 0.25 |

Table 5: **Real World Imitation Learning** Success rate using **different keypoint selection methods**. *ATK* consistently outperforms alternative keypoint selection methods using random sampling or ChatGPT selection.

558 E Baseline implementation details

559 E.1 Encoders

560 **ResNet18 Encoder:** We use a modified ResNet18 architecture as the encoder for depth inputs. The
561 depth data is first resized to 224×224 and duplicated across three channels to match the expected input
562 format of the ResNet18 backbone. Then, we take the ResNet’s 512-dimensional feature embedding
563 as a feature map and concatenate it with a 7-dimensional robot state (six joint values and a binary
564 gripper state). The resulting vector is then passed to the diffusion policy [46] for action prediction.

565 **R3M Encoder:** To handle RGB inputs, we first resize the RGB images from 640×480 resolution
566 to 224×224 . Then, we use R3M[50] to extract a 512-dimensional feature embedding given image
567 inputs and concatenate this embedding with the 7-dimensional joint and gripper state vector to form
568 the input to the diffusion policy [46].

569 **DP3 Encoder:** For encoding 3D point cloud data, we follow the encoder from DP3[51], a CNN-based
570 architecture that utilizes layer normalization. We first project the depth map into 3D and do a spatial
571 crop to get dense point clouds. We then uniformly down-sample 4096 points as the input for the 3D
572 encoder.

573 The output 3D feature embedding with size 1024 is concatenated with the same 7-dimensional robot
574 state vector (six joint values and a binary gripper flag) before being used as input to the diffusion
575 policy [46].

Prompt for keypoint selection using GPT for Grape-oven task

Consider the image:



Task Description:

You are training a robotic arm policy to perform a task that involves:

1. Identifying and picking up a grape object in an image.
2. Placing the object inside an open gated oven.
3. Closing the oven door.

Keypoint Requirements:

The robot's control policy uses relevant visual keypoints as input to solve the above task. Your objective is to:

- Identify and output 4 critical keypoints in the provided image that will help the policy learn this task.
- Draw your output keypoints on the above given image

577

GPT Output



Here are the 2D pixel coordinates for the selected keypoints:

On the Grape Object

- Left side of the object: **(210, 375)**
- Right/top side of the object: **(225, 368)**

On the Oven (Above the Stove)

- Inside the oven tray (for placing the object): **(305, 190)**
- Oven handle (for closing the door): **(270, 160)**

578

Consider the image:



Task Description:

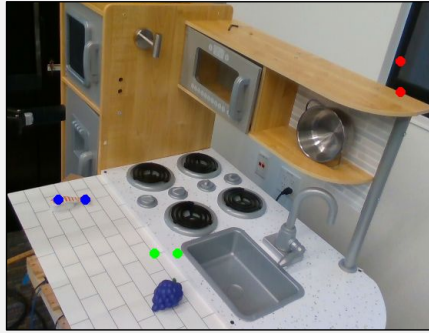
You are training a robotic arm policy to perform a task that involves:

1. Identifying and picking up a pan placed on a shelf above the sink in the above image.
2. Placing the pan on the right bottom gas burner.
3. Picking up sushi, placing it inside the pan, and finally picking up a grape and placing it inside the pan.

Keypoint Requirements:

The robot's control policy uses relevant visual keypoints as input to solve the above task. Your objective is to:

- Identify and output 6 critical 2D keypoints in the provided image that will help the policy learn this task.
- Draw your output keypoints on the above given image



Here are the 2D pixel coordinates for the selected keypoints: Sushi Keypoints:

- sushi_1: (105, 230)
- sushi_2: (125, 240)

Grape Keypoints:

- grape_1: (215, 350)
- grape_2: (225, 370)

Pan Keypoints:

- pan_1: (495, 160)
- pan_2: (525, 180)

580

Prompt for keypoint selection using GPT for Towel Folding task

Consider the image:



Task Description: You are training a robotic arm policy to perform a task that involves identifying and folding a towel lying flat on the kitchen counter:

Keypoint Requirements:

The robot's control policy uses relevant visual keypoints as input to solve the above task. Your objective is to:

- Identify and output 6 critical 2D keypoints in the provided image that will help the policy learn this task.
- Draw your output keypoints on the above given image

581

GPT Output



Here are the 2D pixel coordinates for the selected keypoints:

- Left Edge: (85, 365)
- Right Edge: (225, 365)
- Top Edge: (225, 320)

582

Prompt for keypoint selection using GPT for Hanging Blanket task

Consider the image:



Task Description: You are training a robotic arm policy to perform a task that involves identifying and picking up a blanket lying flat on the gas stove and hang it on a hook attached on a side wall:

Keypoint Requirements:

The robot's control policy uses relevant visual keypoints as input to solve the above task. Your objective is to:

- Identify and output 4 critical 2D keypoints in the provided image that will help the policy learn this task.
- Draw your output keypoints on the above given image

583



Here are the 2D pixel coordinates for the selected keypoints:

- Grasp Ball (Pickup): (330, 300)
- Towel Top Edge: (280, 280)
- Towel Center: (300, 320)
- Towel Side Edge: (260, 340)