Robust Instant Policy: Leveraging Student's t-Regression Model for Robust In-context Imitation Learning of Robot Manipulation

Anonymous Author(s)
Affiliation
Address
`email`

Abstract: Imitation learning (IL) aims to enable robots to perform tasks autonomously by observing a few human demonstrations. Recently, a variant of IL, called In-Context IL, utilized off-the-shelf large language models (LLMs) as instant policies that understand the context from a few given demonstrations to perform a new task, rather than explicitly updating network models with large-scale demonstrations. However, its reliability in the robotics domain is undermined by *hallucination* issues such as LLM-based instant policy, which occasionally generates poor trajectories that deviate from the given demonstrations. To alleviate this problem, we propose a new robust in-context imitation learning algorithm called the robust instant policy (RIP), which utilizes a Student's t-regression model to be robust against the hallucinated trajectories of instant policies to allow reliable trajectory generation. Specifically, RIP generates several candidate robot trajectories to complete a given task from an LLM and aggregates them using the Student's t-distribution, which is beneficial for ignoring outliers (*i.e.,* hallucinations); thereby, a robust trajectory against hallucinations is generated. Our experiments, conducted in both simulated and real-world environments, show that RIP significantly outperforms state-of-the-art IL methods, with at least 26% improvement in task success rates, particularly in low-data scenarios for everyday tasks. Video results available at `https://sites.google.com/view/robustinstantpolicy`.

Keywords: Imitation Learning, In-Context Imitation Learning

1 INTRODUCTION

Imitation learning (IL) is a promising technique for learning policies to automate robot manipulation by observing human demonstrations [1]. It has shown considerable success in a wide range of applications with large datasets and highly expressive policy models [2, 3, 4]. However, despite its capabilities, it remains limited because it requires thousands of demonstrations and/or long-term model weight tuning to be applied to new tasks or environments. This motivates a more efficient paradigm for learning an instant policy that can be immediately adapted and deployed to new tasks while minimizing costs.

A previous study attempted to achieve this by in-context imitation learning (ICIL), in which a large transformer model trained on a variety of datasets can be immediately generalized to a new task by providing a few demonstrations of the new task as a context without updating the model [5]. Although this technique is based on findings in the language and visual domains [6], where sufficient large-scale data are available, the amount of data in the robotics domain remains insufficient for widespread applications. In contrast, keypoint action tokens (KAT) translate robot trajectory data into keypoint-based text, enabling the off-the-shelf large language model (LLM) to be reused as an instant policy [7], which has shown comparable task achievement to state-of-the-art IL methods by using fewer than 10 human demonstrations of a new task without updating the model.

Although the LLM is a useful tool for instant robotic agents, it faces the inherent issue of *hallucinations* , where its responses may be out of a given context and lack accuracy [8]. Specifically, this issue is critical in the robotics domain, where even a small loss of precision in robot trajectory generation owing to hallucinations can lead to task failure. Although several methods have been proposed to mitigate these hallucinations, they mainly focus on discrete language data [9] and are unsuitable for continuous robot trajectory data. Thus, an approach that uses continuous robot data is required to enhance the robustness of LLM-based instant robot policies.

Therefore, this study proposes a novel robust ICIL method that generates a robust trajectory against hallucinations of LLM through stochastic treatment (Fig. 1): the robust instant policy (RIP). In RIP, a few demonstration trajectories of a robot for a new task are fed into an instant policy by following a prior LLM-based method (*i.e.,* KAT [7]) to generate the robot trajectory for completing a given task. This process is performed iteratively to gather multiple response trajectories. The robust trajectory is captured from the set of response trajectories by using the Student's t-regression model [10], which is a well-established model for ignoring outliers (*i.e.,* hallucinations). RIP minimizes hallucinations, thereby enabling a reliable robot agent with a robust trajectory. Validation in both simulated and real-world environments demonstrates that RIP significantly outperforms state-of-the-art imitation learning (IL) methods, particularly in low-data scenarios for everyday tasks.

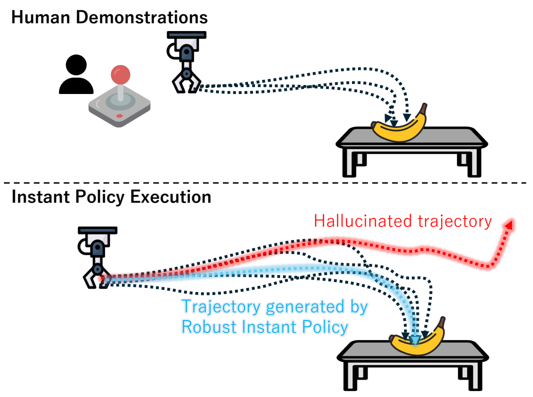

Figure 1: Robust instant policy (RIP) for a robotic banana-picking task. Given a few human demonstrations, LLM-based instant policy can capture the task's context and generate trajectories such as demonstrations, but some may deviate from demonstrations (red) owing to LLM's hallucinations. In contrast, RIP generates a robust trajectory (blue) to hallucinations using a Student's t-regression model that averages trajectories of the instant policy while ignoring hallucinations.

2 RELATED WORKS

2.1 Imitation Learning from Human Demonstration

IL is a promising approach in which robots learn to perform tasks autonomously by observing human demonstrations rather than relying on manual engineering. Recent advances in deep learning architectures have enabled the imitation of a broader range of complex human behaviors. For example, highly expressive architectures such as energy-based models [11] and diffusion models [12] can efficiently learn probabilistic human behaviors, including discontinuities and multioptimality. Furthermore, training a transformer-based architecture on large and diverse datasets containing various tasks has been shown to lead to generalizable policies for multiple tasks [2, 3, 4]. However, even when training expressive policy models on extensive datasets, these policies often require additional demonstration data and sophisticated fine-tuning to adapt to new tasks and environments.

2.2 In-context Imitation Learning for Instant Policy

ICIL is a notable paradigm that enables robotic policies to adapt instantly to new tasks using a few human demonstrations without explicit policy updates. To this end, deep learning methods, such as contrastive learning, can train a robot to identify similarities between tasks, enabling it to control a robot for test tasks based on similar training tasks [13]. Furthermore, the transformer architecture enables similarity matching based on a self-attention mechanism, which led to the proposal of a more concise and highly capable ICIL approach [5]. However, these approaches are effective for tasks similar to those they are trained on; thus, they require sufficient data to cover a vast range of tasks, which is still lacking in the robotics domain. The KAT method addresses this problem by converting robotic data into text, enabling off-the-shelf LLMs [14] to be reused as policies, yielding performance comparable to advanced IL methods with fewer than 10 demonstrations [7]. Despite their effectiveness, LLMs face the issue of hallucinations [8], where they occasionally provide responses that do not fit the given context, and LLM-based KAT is also vulnerable to this.

2.3 Mitigating Hallucination in Large Language Models

Numerous studies have been conducted to mitigate these hallucinations in LLM. Two main streams of research include the following: estimating the uncertainty of LLM predictions by designing explicit probabilistic models [15] and estimating the implicit confidence from LLM responses [16]. The estimation of the implicit confidence from LLM responses is based on the fact that sampled responses are likely to be consistent if the LLM knows a given question, whereas hallucinated responses diverge and contradict each other. This is implemented in a sampling-based approach that assigns a high confidence score to consistent text tokens among the responses sampled from the LLM, and has become more widespread because of its flexibility for diverse applications [17, 9] and statistical guarantees [18]. However, in robotics, most research has focused primarily on discrete language data [9], which poses challenges in applying these techniques to continuous robot trajectory data. To address this problem, this study proposed an approach that employs a Student's t-regression model [10] to extract reliable trajectories from the continuous robot trajectory data generated by LLM and investigated its effectiveness on the ICIL of robotics.

3 FORMULATION

In this paper, we consider an ICIL scenario in which a human expert provides a few demonstrations of task execution, and a robot immediately imitates these demonstrations to autonomously perform the new task. The expert demonstrations are represented as $\mathcal{D} = \{d_i\}_{i=0}^{I}$, where I is the total number of demonstration episodes, and each demonstration d_i consists of a sequence of observations o^i received by the robot and a sequence of desired actions a^i that the robot should exhibit to complete the task. The objective of ICIL is to develop an instant policy $\Phi(\mathcal{D}, o') \rightarrow \hat{a}$ that effectively generalizes the expert policies implicitly expressed in $\mathcal{D}$, enabling appropriate actions $\hat{a}$ to be predicted based on new observations o'.

3.1 Leveraging Large Language Models as an Instant Policy

A key feature of the ICIL is that it does not require additional updates to the policy model parameters. Although certain approaches explicitly train a specific model based on robot data to achieve this feature, KAT show that off-the-shelf LLMs can function as instant policies without fine-tuning [7]. Following this notion, this study was formulated based on the LLM-based approach as follows:

At the beginning of each episode, the robot captures an RGB-D image observation o^i. During demonstration d_i, the robot collects observations o^i along with a set of action trajectories $\{a_t^i\}_{t=1}^{T}$ of length T. When executing an instant policy, the robot records a new observation o' and infers a series of actions $\hat{\xi} = \{\hat{a}_t\}_{t=1}^{T}$ that replicate the expert behavior observed in the demonstration dataset $\mathcal{D}$.

To accelerate the inference capability of LLMs, 3D keypoints-based observation and action space are introduced: 3D visual keypoints extracted from complex RGB-D images as observation and triplet 3D points that can describe the robot's end-effector posture as action.

To extract 3D visual keypoints based on semantic and geometric similarities, a state-of-the-art vision transformer model called DINO [19] is used, and the process is as follows:

(i) Extracting DINO descriptors from RGB-D image o^i in each demonstration yields $z^i \in \mathbb{R}^{N\times 6528}$, with N as the number of patches in each image; that is, each descriptor contains informative features of the patch.

(ii) Extracting K descriptors for each image that are most similar to other images by comparing descriptors between images via a nearest neighbor search algorithm [20], from which K 2D keypoints are calculated.

(iii) Projecting each 2D keypoints into 3D space by using the depth of each image and known camera intrinsic parameters yields $\mathcal{V} \in \{V_i\}_{i=0}^{I}$, where $V_i = \{v_k\}_{k=1}^{K}$ is a set of visual 3D keypoints v_k of each image.

For a new observation o', we also extract the DINO descriptors z'. We then find the K nearest neighbors for each descriptor in $\mathcal{V}$ and extract the corresponding visual 3D keypoint V'.

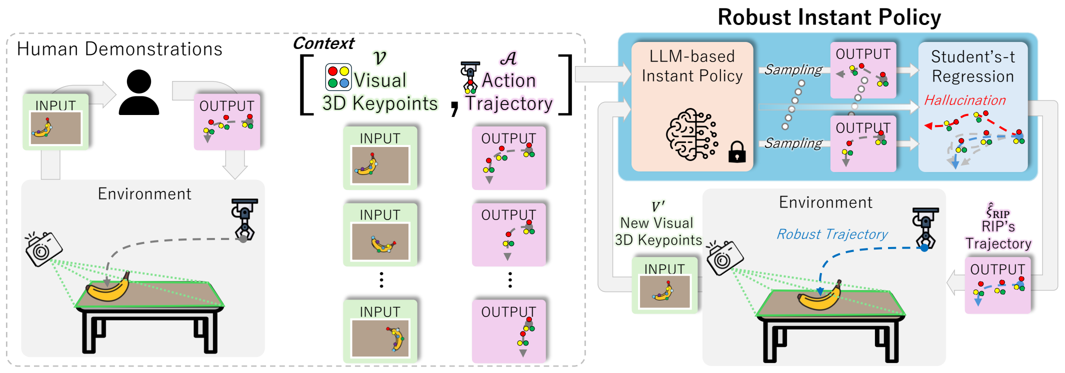

Figure 2: Overview of RIP in a banana-picking task. The input of the initial image and output of the robotic gripper's trajectory are collected through human demonstrations. From the demonstration dataset, contextual text data is tokenized: the visual 3D keypoints with semantic and geometric similarities are extracted from image inputs, and the action trajectory consists of a 3D triplet representing the gripper's posture. The LLM-based instant policy is fed the context data and new image keypoints, sampling the action trajectories multiple times. Using the Student's t-regression model, a robust trajectory is captured from the set of sampled action trajectories, and the robot performs the task following that reliable trajectory.

In addition, the action sequence is defined by the trajectory of the robot's end-effector poses. Each unique pose of the robot end-effector is defined by a triplet of 3D points ($\tau_t^i = [p_{t,0}^i, p_{t,1}^i, p_{t,2}^i]$) that represent the x, y, z positions of the gripper and both fingertips when the gripper is open. The state of the gripper is represented by the variable g_t^i, where $g_t^i = 0$ indicates that the gripper is open and $g_t^i = 1$ indicates that it is closed. Therefore, action is defined as $\mathbf{a}_t^i = [\tau_t^i, g_t^i] \in \mathbb{R}^{10}$, and all action sequences in demonstration is defined as $\mathcal{A} = \{\xi^i\}_{i=1}^{I}$, where $\xi^i = \{\mathbf{a}_t^i\}_{t=1}^{T}$. We note that all the 3D coordinates are within the world frame.

At the deployment phase, the text token consisting of a context $[\mathcal{V}, \mathcal{A}]$ and a new visual 3D keypoint V' is input into the LLM, and LLM generates a new trajectory of the desired end-effector motion to complete a task as an instant policy as follows: $\Phi_{\text{KAT}}([\mathcal{V}, \mathcal{A}], V') \to \hat{\xi}$, where $\hat{\xi} = \{\hat{\mathbf{a}}_t\}_{t=1}^{T}$. For further details, please refer to [7].

Although this instant policy performs favorably compared with other IL approaches, its applicability is not yet stable because of the LLM's hallucinations, which generate action sequences that are considerably different from those inherent in demonstrations $\mathcal{D}$. To overcome this drawback, in the next section, we present a novel approach that captures the uncertainty from an instant policy Φ_{KAT} and ignores hallucinations for robust applications.

4 ROBUST INSTANT POLICY

In this section, we introduce a novel stochastic approach for enhancing the robustness of LLM-based instant policy. We start from the simple notion that if the LLM-based policy knows a given task, the sampled trajectories are consistent, whereas the hallucinated trajectories diverge from each other. Thus, hallucinated trajectories can be recognized as outliers that deviate from a consistent trajectory. Therefore, we propose RIP that utilizes the Student's t-regression model [10] that excels at ignoring these outliers and generates a reliable robot trajectory to complete tasks (Fig. 2).

Initially, a set of action trajectories $\mathcal{A}' = \{\hat{\xi}^q\}_{q=1}^{Q}$ is generated by querying the instant policy Q times using the same text token (*i.e.*, context $[\mathcal{V}, \mathcal{A}]$ and a new observation V'). Notably, in practice, this process is concurrent to avoid computational complexity as Q increases. The length of each trajectory can also vary; therefore, the time step t is normalized to the maximum length of each trajectory T to align the generated trajectories based on the start and end of the episode, similar to a previous study [21]. For simplicity without loss of generality, all trajectories have the same length T, and the one-dimensional action $\hat{a}_t \in \hat{\mathbf{a}}_t$ is used in the subsequent formalization.

Given a set of trajectories $\mathcal{A}'$ involving hallucinations, the aim is to train an action estimator to approximate a consistent trajectory. To this end, we employ the Student's t-regression model [10] as our action estimator, which is known to enable considerable robustness to outliers than the standard Gaussian distribution [22]. Specifically, our action estimator is defined as $\mathcal{S}_\theta(\hat{a}_t|t;\nu)$, which outputs the Student's t-distribution with a mean network μ_θ and variance network σ^2_θ for a given time step t with parameter θ:

$$\mathcal{S}_\theta(\hat{a}_t|t;\nu) = \frac{\Gamma((\nu+1)/2)}{\Gamma(\nu/2)\sqrt{\nu\pi\sigma^2_\theta(t)}}\left\{1 + \frac{(\hat{a}_t - \mu_\theta(t))^2}{\nu\sigma^2_\theta(t)}\right\}^{-(\nu+1)/2}, \tag{1}$$

where $\Gamma(\cdot)$ is the gamma function and ν is the degree of freedom with a positive real value. ν is a hyperparameter that regulates the sensitivity to outliers. As it approaches infinity (*i.e.,* $\nu \to \infty$), the distribution resembles a normal Gaussian distribution. Please see details on [22].

Subsequently, to capture Student's t-distribution of action trajectories, the loss of the action estimator is defined as a negative log-likelihood:

$$\mathcal{L}(\mathcal{S}_\theta|\mathcal{A}') = \sum_{q=1}^{Q}\sum_{t=1}^{T} -\log \mathcal{S}_\theta(\hat{a}^q_t|t;\nu). \tag{2}$$

Therefore, our objective function is to train the parameter θ of the action estimator by minimizing the expected loss along the action trajectories $\mathcal{A}'$ derived from an instant policy.

$$\hat{\theta} = \arg\min_\theta \mathbb{E}_{\mathcal{A}'\sim\Phi_{\text{KAT}}}[\mathcal{L}(\mathcal{S}_\theta|\mathcal{A}')]. \tag{3}$$

Finally, using the learned action estimator $\mathcal{S}_{\hat{\theta}}$, a consistent trajectory is extracted from a set of action trajectories $\mathcal{A}'$. This calculation is performed over time steps using the mean of the learned action estimator: $\hat{\xi}_{\text{RIP}} = \{\mu_{\hat{\theta}}(t)\}^T_{t=1}$. A summary of RIP is presented in Algorithm 1.

```
Algorithm 1: Robust instant policy (RIP)
Input  : Instant policy $\Phi_{\text{KAT}}$, number of queries $Q$, context $[\mathcal{V}, \mathcal{A}]$, new observation $V'$
Output: Robust action trajectory $\hat{\xi}_{\text{RIP}}$
1 for $q = 1$ to $Q$ do
2     Get actions from an instant policy of KAT: $\hat{\xi}^q \sim \Phi_{\text{KAT}}([\mathcal{V}, \mathcal{A}], V')$ ;
3     Aggregate action trajectories: $\mathcal{A}' \leftarrow \mathcal{A}' \cup \hat{\xi}^q$;
4 end
5 Learn the action estimator parameter $\hat{\theta}$ by Eq. (3);
6 Get the consistent trajectory $\hat{\xi}_{\text{RIP}}$
```

5 EVALUATION

In this study, a novel robust ICIL approach called RIP was proposed to obtain robust instant robot policies that are immediately applicable to novel tasks. Therefore, our evaluation was conducted using simulated and real-world experiments to answer two main questions: 1) How is RIP comparable to state-of-the-art IL approaches? 2) What is the optimal RIP design for maximizing performance?

5.1 Evaluation Setting

5.1.1 Environments and Tasks

To evaluate the RIP, as shown in Fig. 3, a set of 10 daily tasks was defined for simulated and real-world environments as follows:

Simulation environment was implemented based on the Maniskill3 benchmark [23]. The initial RGB-D image of the top view was captured using the built-in Maniskill3 RGB-D camera at the start of each episode. A human provides task demonstrations for a Universal Robotics UR5e 6DOF robot equipped with a Robotiq Hand-E gripper, using a leader-follower teleoperation system called GELLO [24]. These are recorded at 55 Hz, but will be 5.5 Hz during the test phase to ensure stability. Given demonstrations, the aim is to obtain an instant robotic agent that performs the following tasks:

- Pick Banana: A task where a robot picks a banana from a table. A banana is placed randomly; a robot needs to reach it precisely to pick it up.

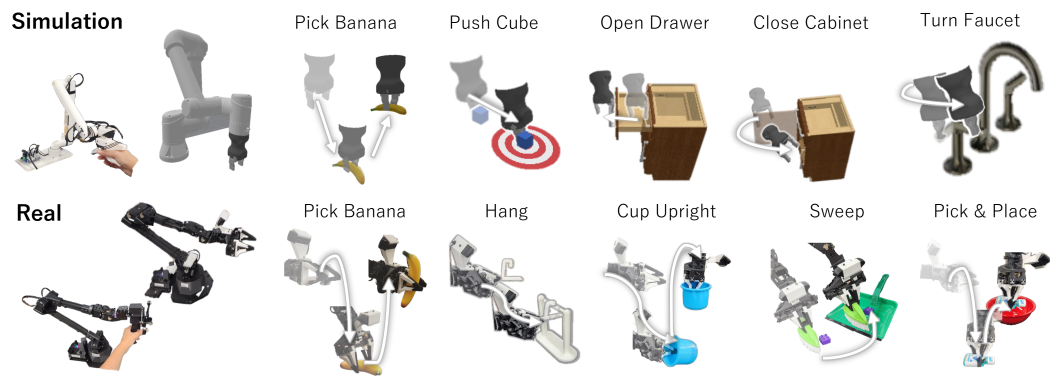

Figure 3: Environments and Tasks. In both simulated and real-world environments, human demonstrations are collected with leader-follower robot systems, where the movements of a directly human-controlled leader robot are followed by a follower robot with a similar embodiment. Under these robotic environments, RIP and baseline IL methods were evaluated for their capability on 10 everyday manipulation tasks.

- Push Cube: A task where a robot pushes a cube to the goal position. A cube is placed randomly; a robot agent's nonprehensile capability is evaluated.
- Open Drawer: A task where a robot pulls a handle to open a drawer. A drawer is placed randomly; a robot needs to pull a handle correctly to open it.
- Close Cabinet: A task where a robot pushes a cabinet door to close it. A cabinet is randomly placed, and a robot needs to push the door in the correct orientation to close it.
- Turn Faucet: A task where a robot turns the faucet. A faucet is placed randomly; a robot needs to turn it in the correct orientation.

Real-world environment was built on the ALOHA [25] robotic system that also employed leader-follower teleoperation with two 6DOF robot arms. Demonstrations are recorded at 10 Hz, but will be 1 Hz during the test phase to ensure stability. An Intel RealSense D415 camera was mounted on top of the table to capture an RGB-D image at the start of each episode. Given the demonstrations, the aim is to obtain an instant robotic agent that performs the following tasks:

- Pick Banana: Same as simulation.
- Hang: A task where a robot grabs a clothes hanger, reaches, and hangs it on a horizontal stand. A stand is placed randomly; a robot needs to bring a hanger and release it to the correct position.
- Put Cup Upright: A task where a robot picks up a horizontally placed cup and places it upright on the table. A cup is placed randomly; a robot needs to grasp and rotate a cup precisely.
- Sweep: A task where a robot grabs a brush and sweeps an object into a dustpan. A dust object is placed randomly; a robot needs to sweep it with a brush while touching the table.
- Pick-and-Place: A task where a robot picks up a bottle and places it in a red bowl. A bowl and a bottle are placed randomly; a robot needs to pick a bottle and put it in a red bowl precisely.

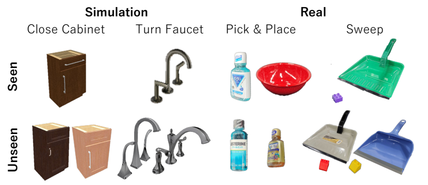

Figure 4: Objects used to evaluate the generalizability of unseen objects during training.

Inspired by [7], to evaluate the generalizability of objects unseen during training, the following four tasks also test unseen objects in the demonstration: close a cabinet and turn a faucet in a simulation and sweep and pick-and-place in a real-world environment. Specific objects are described in Fig. 4.

5.1.2 Comparison Methods

In this evaluation, we compared our method (RIP) with other IL methods, including the following:

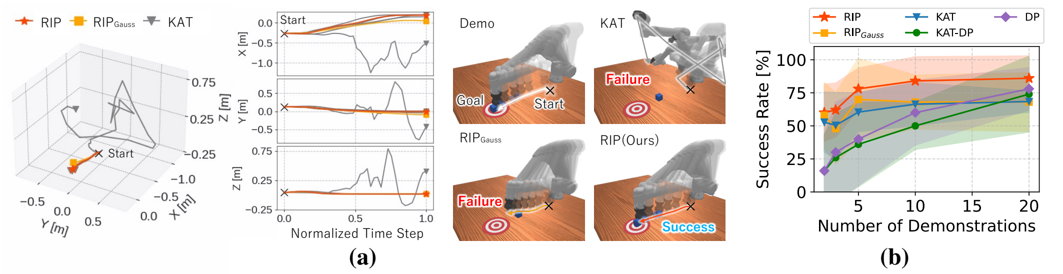

Figure 5: Evaluation Results: (a) Qualitative analysis comparing trajectories generated by LLM-based instant policies (*e.g.,* RIP, RIP$_{\text{Gauss}}$ and KAT) on push cube simulation tasks. Note that only the trajectory of the gripper position is represented for intuitive analysis. (b) Quantitative analysis comparing the success rate of RIP and baselines regarding the number of demonstrations. All results represent the mean (line) and standard deviation (shaded) of the success rate on all the simulation tasks. The success rate of each method, except KAT, was measured over 10 test executions. For KAT, all five trajectories generated for RIP from one initial image were tested, *i.e.,* 50 test executions.

- Diffusion Policy (DP) [12]: A state-of-the-art IL algorithm that demonstrated superior learning efficiency and generality compared to several previous IL methods.
- Keypoint Action Tokens (KAT) [7]: A state-of-the-art in-context IL algorithm that introduces an LLM as an instant policy by introducing the keypoint-based observation-action space described in §3.1.
- KAT-DP [7]: An IL algorithm that combines KAT and DP on [7]. Utilizing the keypoint representation of KAT for the observation-action space instead of the raw image input from the original DP resulted in performance comparable to KAT [7].
- RIP with Gaussian (RIP$_{\text{Gauss}}$): A baseline for ablation studies in RIP; it uses a normal Gaussian distribution (*i.e.,* $\mathcal{N}(\hat{a}_t|\mu_\theta(t), \sigma^2_\theta(t))$) instead of the Student's t-distribution.

Notably, the keypoint-based method extracts $K = 10$ keypoints from observations recorded at the beginning of the episode as described in §3.1. All the LLM-based methods use GPT 4o [14] as an instant policy model. RIP uses a fixed number of degrees, $\nu = 1.5$, and a query count of $Q = 5$. These hyperparameters are chosen to improve the success rate, which is analyzed in §5.2.2. See §A for detailed setting of 3D keypoints extraction and RIP model (S_θ).

5.1.3 Downsampling Action Demonstration Dataset

In practice, the redundant action trajectory length T degrades the performance of LLM-based instant policies [7]. KAT employs an approach that uniformly down-samples data from a dataset collected at a high frequency to obtain a length of 20–30. However, this downsampling may omit the key steps of the demonstration. In particular, in tasks involving grasping or releasing, downsampling reduces the gripper-action precision of the demonstration. To alleviate this, we first mask the time steps of the start and end of episodes as well as the gripper actions $|g^i_{t+1} - g^i_t| = 1$ when it is activated to open or close, and we uniformly sample the actions between these masked data to achieve a length of approximately 30. This downsampling was applied to all comparison methods. This technique improved the performance of our method (RIP) in tasks involving gripper actions. The analysis is described in §5.2.2.

5.2 Results

5.2.1 How does RIP comparable to state-of-the-art imitation learning approaches?

Qualitative and quantitative analyses were conducted to answer this question.

Qualitative Results: The results of the qualitative analysis comparing the trajectories generated by LLM-based instant policies (RIP, RIP$_{\text{Gauss}}$, and KAT) are shown in Fig. 5a. Most of the trajectories generated by KAT accomplished this task by capturing the context provided by human demonstrations. However, one outlier was significantly different from the others, and the robot failed the task, which could be attributed to hallucinations of the LLM. In the case of RIP$_{\text{Gauss}}$, averaging over the

Methods	Simulation Tasks Success Rate [%]					Real-world Tasks Success Rate [%]					Avg. [%]
	Pick Banana	Push Cube	Open Drawer	Close Cabinet	Turn Faucet	Pick Banana	Hang	Sweep	Pick & Place	Cup Upright	
DP	40	30	60	**100**	70	60	30	40	10	60	$50^{**} \pm 24$
KAT-DP	**60**	40	20	70	60	**100**	50	20	30	**70**	$52^{*} \pm 24$
KAT	48	74	42	90	78	80	50	10	30	40	$54^{*} \pm 24$
RIP$_{\text{Gauss}}$ (Ours)	40	70	50	90	**90**	70	60	60	40	**70**	64 ± 17
RIP (Ours)	**60**	**100**	**70**	**100**	**90**	**100**	**90**	**70**	**50**	**70**	80 ± 17

Table 1: Quantitative Results: Comparing the success rate with 10 demonstrations between RIP and baseline methods for each task. The success rate of each method, except KAT in simulation, is measured over 10 test executions. For KAT in simulation, all five trajectories generated for RIP from one initial image are tested, *i.e.,* 50 test executions. For each task result, the one in bold is the best. Average shows the mean and standard deviation of the results, our method is significantly better than task results marked * or ** (t-test, $p < 5e{-2}$ and $p < 5e{-3}$, respectively).

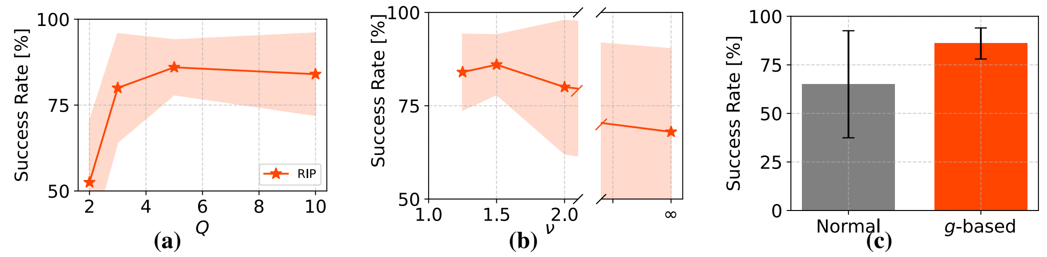

Figure 6: Design Analysis of RIP: RIP has three main design elements (Q, ν, and downsampling). (a) and (b): Success rate comparison of RIP for different $Q = \{2, 3, 5, 10\}$, $v = \{1.25, 1.5, \infty\}$, where ∞ is the same as RIP$_{\text{Gauss}}$. The results present the mean and standard deviation of the success rates obtained by testing the RIP with 20 demonstrations 10 times for each simulated task. (c): Success rate comparison of RIP without and with grasping-action-based downsampling, represented as Normal and g-based, respectively. The results present the mean and standard deviation of the success rates obtained by testing the RIP with 10 demonstrations 10 times for each task.

trajectories of the KAT, the effect of this outlier cannot be ignored. It generates a trajectory close to the hallucinated trajectory, that is, the X trajectory of RIP$_{\text{Gauss}}$ is 14 cm less than that of RIP at the final step. This deviation caused the robot to fail to reach the cube, resulting in task failure. By contrast, RIP can average the trajectories while ignoring the outliers owing to hallucinations and succeeds in the task.

Quantitative Results: The results of the quantitative analysis comparing the policy success rates of each method are presented in TABLE 1 and Fig. 5b.

TABLE 1 lists the policy success rate for each comparison method in a learning scenario where only a limited number of demonstrations ($I = 10$) is provided for each manipulation task. Although the LLM-based instant policy methods (KAT, RIP$_{\text{Gauss}}$, and RIP) do not require additional training after receiving demonstrations, their average performance outperforms that of the other baseline methods (DP and KAT-DP), which require extra training for each task. Notably, the RIP method achieved a significant improvement in the performance for KAT, with a 26% increase, and showed the best results across all tasks. Furthermore, there is a 16% improvement compared with our ablation method (RIP$_{\text{Gauss}}$), which aligns with the qualitative results and validates the effectiveness of our Student's t-based design.

In addition, Fig. 5b shows the quantitative results of simulation tasks that investigate the policy success rate for each comparison method over a wider range of demonstration amounts. The results indicate that up to 10 demonstrations, the LLM-based instant policy method outperformed the other baseline methods. This finding supports the notion that instant policies can yield higher success rates than state-of-the-art IL approaches, without the need for additional training. However, LLM-based instant policies that lack robustness against hallucinations were surpassed by other baseline methods when the number of demonstrations reached 20 and failed to achieve a success rate above 70%. In contrast, our approach (RIP) consistently demonstrated a superior probability of success across all demonstration amounts. Overall, both the qualitative and quantitative results show that our approach (RIP) is more effective than other state-of-the-art IL methods, particularly in scenarios where the number of demonstrations is limited.

5.2.2 What is the optimal design of RIP to maximize performance?

To investigate this question, we conducted a quantitative analysis of three key factors in RIP design: the number of queries (Q), degrees of freedom (ν), and downsampling methods. The results are presented in Fig. 6.

Fig. 6a displays a comparison of the success rate of RIP with varying numbers of queries (Q). RIP must extract a robust trajectory from a set containing a sufficient number of reliable trajectories. For $Q = \{2\}$, reliable trajectories may not be obtained sufficiently, making RIP vulnerable to hallucinations, and its success rate is similar to that of KAT. By contrast, for $Q \geq 3$, the RIP success rate increases with an increase in Q, peaking at $Q = 5$. After this point, performance stabilizes, showing no significant improvements; thus, $Q = 5$ was used in §5.2.1.

Fig. 6b shows the quantitative analysis of the success rate of RIP for different degrees of freedom (ν). In RIP, the degree of freedom ν can be interpreted as the level of tolerance of the hallucinated trajectories. When $\nu = \infty$, equivalent to $\text{RIP}_{\text{Gauss}}$ as described in §4, the success rate is the lowest because it is not sufficiently robust against hallucinated trajectories. As ν decreased, the success rate increased until $\nu = 1.5$. We used $\nu = 1.5$ in §5.2.1 because this setting achieved the highest success rate.

Fig. 6c shows the quantitative analysis of the success rate of RIP according to the downsampling method discussed in §5.1.3. For this evaluation, we focused on tasks in which gripper actions were essential (Simulation: Pick banana; Real: Pick banana, hang, pick-and-place, and cup upright). The use of RIP with gripper-action-based downsampling (success rate: 74%) increased the success rate by 22% compared with the method that employs normal downsampling (success rate: 52%). This outcome supports our assertion that the data triggering the gripper action may be omitted during the normal downsampling process. Such omissions can lead to less accurate demonstrations and decrease the reliability of instant policies.

6 DISCUSSION

Our experiments demonstrate that the proposed method (RIP) significantly enhances the robustness of ICIL, resulting in a reliable robotic instant policy. Although this study assumed that demonstrators consistently choose a single optimal behavior, in practice, demonstrators often choose multiple optimal behaviors, and certain research attempted to address this challenge in IL [26]. Our RIP model currently cannot capture this complexity. Thus, one important direction for future work is to introduce a Student's t-mixture model that can capture multiple optimal trajectories while providing robustness to outliers [27].

Additionally, although our method (RIP) can generate a hallucination-robust trajectory using the Student's t-regression model, it does not explicitly recognize hallucinations. Therefore, the underlying causes of the hallucinations remain unclear. Future work will be directed at identifying the causes of hallucinations in robotic instant policies by introducing an approach for recognizing hallucinations through an explicit LLM probabilistic model [15].

In addition, although the computational complexity remains constant regardless of the number of repeated queries (*i.e.,* Q) owing to parallelization, current LLMs using transformer architectures still require an output computational complexity of $O(l^2)$ for prompts of length l [28]. This limitation indicates that current transformer-based LLMs are unsuitable for scenarios that require online control. Therefore, our future work will focus on introducing alternative architectures [29] that exhibit comparable performance but lower complexity.

7 CONCLUSION

This study introduced RIP, a novel ICIL algorithm that leverages stochastic treatment to provide a reliable instant policy for robotic manipulation tasks. RIP uses a Student's t-regression model for robustness against the hallucinated trajectories of LLM-based instant policies and captures a reliable trajectory. Our findings demonstrate that robustification achieves state-of-the-art results in IL for various daily tasks. The design selection analysis reveals the factors that determine the optimal performance of the proposed method. These results show that the RIP allows for a reliable instant robot agent, particularly in scenarios where data are often scarce, such as those in the actual robotics domain.

References

[1] T. Osa, J. Pajarinen, G. Neumann, J. A. Bagnell, P. Abbeel, and J. Peters. An algorithmic perspective on imitation learning. *Foundations and Trends® in Robotics*, 7(1-2):1–179, 2018. ISSN 1935-8253.

[2] O. X.-E. Collaboration. Open X-Embodiment: Robotic learning datasets and RT-X models. In *International Conference on Robotics and Automation*, pages 6892–6903. IEEE, 2024.

[3] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

[4] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi, et al. OpenVLA: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

[5] L. Fu, H. Huang, G. Datta, L. Y. Chen, W. C.-H. Panitch, F. Liu, H. Li, and K. Goldberg. In-context imitation learning via next-token prediction. *arXiv preprint arXiv:2408.15980*, 2024.

[6] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901, 2020.

[7] N. Di Palo and E. Johns. Keypoint action tokens enable in-context imitation learning in robotics. In *Robotics: Science and Systems*, 2024.

[8] Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. J. Bang, A. Madotto, and P. Fung. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38, 2023.

[9] A. Z. Ren, A. Dixit, A. Bodrova, S. Singh, S. Tu, N. Brown, P. Xu, L. Takayama, F. Xia, J. Varley, et al. Robots that ask for help: Uncertainty alignment for large language model planners. In *Conference on Robot Learning*, pages 661–682. PMLR, 2023.

[10] K. L. Lange, R. J. Little, and J. M. Taylor. Robust statistical modeling using the t distribution. *Journal of the American Statistical Association*, 84(408):881–896, 1989.

[11] P. Florence, C. Lynch, A. Zeng, O. A. Ramirez, A. Wahid, L. Downs, A. Wong, J. Lee, I. Mordatch, and J. Tompson. Implicit behavioral cloning. In *Conference on Robot Learning*, pages 158–168. PMLR, 2022.

[12] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song. Diffusion Policy: Visuomotor policy learning via action diffusion. In *Robotics: Science and Systems*, 2023.

[13] E. Jang, A. Irpan, M. Khansari, D. Kappler, F. Ebert, C. Lynch, S. Levine, and C. Finn. Bc-z: Zero-shot task generalization with robotic imitation learning. In *Conference on Robot Learning*, pages 991–1002. PMLR, 2022.

[14] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[15] C. Ling, X. Zhao, X. Zhang, W. Cheng, Y. Liu, Y. Sun, M. Oishi, T. Osaki, K. Matsuda, J. Ji, et al. Uncertainty quantification for in-context learning of large language models. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3357–3370, 2024.

[16] P. Manakul, A. Liusie, and M. Gales. SelfCheckGPT: Zero-resource black-box hallucination detection for generative large language models. In *Conference on Empirical Methods in Natural Language Processing*, pages 9004–9017, 2023.

[17] M. Campos, A. Farinhas, C. Zerva, M. A. Figueiredo, and A. F. Martins. Conformal prediction for natural language processing: A survey. *Transactions of the Association for Computational Linguistics*, 12:1497–1516, 2024.

[18] V. Vovk, A. Gammerman, and G. Shafer. *Algorithmic learning in a random world*, volume 29. Springer, 2005.

[19] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin. Emerging properties in self-supervised vision transformers. In *International Conference on Computer Vision*, pages 9650–9660. IEEE/CVF, 2021.

[20] S. Amir, Y. Gandelsman, S. Bagon, and T. Dekel. Deep ViT features as dense visual descriptors. *ECCVW What is Motion For?*, 2022.

[21] W. J. Eerland, S. Box, and A. Sóbester. Modeling the dispersion of aircraft trajectories using Gaussian processes. *Journal of Guidance, Control, and Dynamics*, 39(12):2661–2672, 2016.

[22] C. M. Bishop. *Pattern recognition and machine learning*. springer, 2006.

[23] S. Tao, F. Xiang, A. Shukla, Y. Qin, X. Hinrichsen, X. Yuan, C. Bao, X. Lin, Y. Liu, T. kai Chan, Y. Gao, X. Li, T. Mu, N. Xiao, A. Gurha, Z. Huang, R. Calandra, R. Chen, S. Luo, and H. Su. ManiSkill3: GPU parallelized robotics simulation and rendering for generalizable embodied AI. *arXiv preprint arXiv:2410.00425*, 2024.

[24] P. Wu, Y. Shentu, Z. Yi, X. Lin, and P. Abbeel. Gello: A general, low-cost, and intuitive teleoperation framework for robot manipulators. In *International Conference on Intelligent Robots and Systems*, pages 12156–12163. IEEE/RSJ, 2024.

[25] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *Robotics: Science and Systems*, 2023.

[26] H. Oh, H. Sasaki, B. Michael, and T. Matsubara. Bayesian Disturbance Injection: Robust imitation learning of flexible policies for robot manipulation. *Neural Networks*, 158:42–58, 2023.

[27] D. Peel and G. J. McLachlan. Robust mixture modelling using the t distribution. *Statistics and computing*, 10:339–348, 2000.

[28] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In *International Conference on Neural Information Processing Systems*, page 6000–6010, 2017.

[29] A. Gu and T. Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

A Additional Details of Extracting Visual Keypoints and Learning RIP

As described in §3.1, we utilized the pretrained model provided by Amir et al. [19], known as DINO [19]. Although certain parameters (*e.g.,* image width (W) and height (H)) were adjusted, as shown in TABLE 2, this did not impact the overall evaluation's generality, and all other parameters remain unchanged from the original.

In addition, the RIP model (*i.e.,* S_θ) is optimized using Eq. (3) with parameter settings provided in TABLE 2.

Parameters of DINO			**Parameters of RIP**		
Parameters	Sim	Real	Parameters	Sim	Real
image W	100	320	hidden size	(64,64)	(64,64)
image H	100	240	ν	1.5	1.5
num_pairs	10	10	batch size	64	64
load_size	100	240	optimizer	Adam	Adam
layer	9	9	learning steps	$4e4$	$1e5$
facet	key	key	learning rate	$1e{-2}$	$5e{-2}$
bin	True	True			
thresh	0.15	0.15			
model_type	dino_vits8	dino_vits8			
stride	2	4			

Table 2: Parameter Setting of DINO and RIP.