Point Policy: Unifying Observations and Actions with Key Points for Robot Manipulation

Siddhant Haldar Lerrel Pinto

New York University

***Abstract*—Building robotic agents capable of operating across diverse environments and object types remains a significant challenge, often requiring extensive data collection. This is particularly restrictive in robotics, where each data point must be physically executed in the real world. Consequently, there is a critical need for alternative data sources for robotics and frameworks that enable learning from such data. In this work, we present Point Policy, a new method for learning robot policies exclusively from offline human demonstration videos and without any teleoperation data. Point Policy leverages state-of-the-art vision models and policy architectures to translate human hand poses into robot poses while capturing object states through semantically meaningful key points. This approach yields a morphology-agnostic representation that facilitates effective policy learning. Our experiments on 8 real-world tasks demonstrate an overall 75% absolute improvement over prior works when evaluated in identical settings as training. Further, Point Policy exhibits a 74% gain across tasks for novel object instances and is robust to significant background clutter. Videos of the robot are best viewed at point-policy.github.io.**

I. Introduction

Recent years have witnessed remarkable advancements in computer vision (CV) and natural language processing (NLP), resulting in models capable of complex reasoning [2, 67, 78], generating photorealistic images [7, 70] and videos [49], and even writing code [15]. A driving force behind these breakthroughs has been the abundance of data scraped from the internet. In contrast, robotics has yet to experience a similar revolution, with most robots still confined to controlled or structured environments. While CV and NLP can readily take advantage of large-scale datasets from the internet, robotics is inherently interactive and requires physical engagement with the world for data acquisition. This makes collecting robot data significantly more challenging, both in terms of time and financial resources.

A prominent approach for training robot policies has been the collection of extensive datasets, often through contracted teleoperators [54, 12, 72], followed by training deep networks on these datasets [72, 19, 61, 41]. While effective, these methods tend to require months or even years of human effort [12, 41] and still result in datasets orders of magnitude smaller than those used in CV and NLP [61, 41]. A potential solution to this data scarcity in robotics is to tap into the vast repository of human videos available online, showcasing individuals performing a wide range of tasks in diverse scenarios.

Correspondence to: siddhanthaldar@nyu.edu

The primary challenge in learning robot policies from human videos lies in addressing the morphology gap between robots and the human body [4, 25, 10, 9, 68]. Two notable trends have emerged in efforts to utilize human data for learning robot policies: (1) first learning visual representations or coarse policies from human datasets and then finetuning them for downstream learning on robot datasets [10, 9, 68, 58, 11, 82, 52, 53, 38], and (2) using human videos to compute rewards for autonomous policy learning through reinforcement learning [87, 4, 25, 43]. While the former requires a substantial amount of robot demonstrations to learn policies for downstream tasks, the latter often requires large amounts of online robot interactions in the real world, which can be time-consuming and potentially unsafe.

In this work, we introduce Point Policy, a new technique to learn robot policies solely from offline human data without requiring robot interactions during training. Our key observation in building Point Policy is that both humans and robots occupy the same 3D space in the world, which can be tied together using key points derived from state-of-the-art vision models.

Concretely, Point Policy works in three steps. First, given a dataset of human videos, a motion track of key points on the human hand and the object is computed using hand pose detectors [51, 64] and minimal human annotation of one frame per task. These key points are computed from two camera views, which allows for projection in 3D using point triangulation. Second, a transformer-based policy [28] is trained to predict future robot points given the set of key points derived in the previous stage. Third, during inference, the predicted future robot points in 3D space are used to backtrack the 6 DOF pose of the robot's end-effector using constraints from rigid-body geometry. The gripper state of the robot end effector is predicted as an additional token. The predicted end-effector pose and gripper state are then executed on the robot at 6 Hz.

We demonstrate the effectiveness of Point Policy through experiments on 8 real-world tasks on a Franka robot. Our main findings are summarized below:

1) Point Policy exhibits an absolute improvement of 75% over prior state-of-the-art policy learning algorithms across 8 real world tasks when evaluated in identical settings as training. (Section III-B).
2) Point Policy generalizes to novel object instances, exhibited a 74% absolute improvement over prior work on a held-out set of objects unseen in the training data. (Section III-C).
3) Policies trained with Point Policy are robust to the presence of background distractors, performing at par with scenes

without clutter (Section III-D).

4) We provide an analysis of co-training Point Policy with teleoperated robot data (Section H4) and study the importance of several design choices in Point Policy (Section H5).

All of our datasets, and training and evaluation code have been made publicly available. Videos of our trained policies can be seen here: point-policy.github.io.

II. Point Policy

Point Policy seeks to learn generalizable policies exclusively from human videos that are robust to significant environmental perturbations and applicable to diverse object locations and types. An overview of our method is presented in Figure 1. Before diving into the details, we first present some of the key assumptions needed to run Point Policy.

Assumptions: (1) The pose of the human hand in the first frame is known for each task. This is needed to initialize the robot and set that pose as the base frame of operation. This assumption can be relaxed with a hand-pose estimator [64], which we do not investigate in this work. (2) We operate in a calibrated scene with the camera's intrinsic and extrinsic matrices, and the transforms between each camera and the robot base known. In practice this is a one-time process that takes under 5 minutes when the robot system is first installed.

A. Point-based Scene Representation

Our method begins by collecting human demonstrations, which are then converted to a point-based representation amenable to policy learning.

1) Human-to-Robot Pose Transfer: For each time step t of a human video, we first extract image key points on the human hand p_h^t using the MediaPipe [51] hand pose detector, focusing specifically on the index finger and thumb. The corresponding hand key points p_h^t obtained from two camera views are used to compute the 3D world coordinates $\mathcal{P}_h^t$ of the human hand through point triangulation. We use point triangulation for 3D projection due to its higher accuracy as compared to sensor depth from the camera (Section H5). The robot position $\mathcal{R}_{pos}^t$ is computed as the midpoint between the tips of the index finger and thumb in $\mathcal{P}_h^t$. The robot orientation $\mathcal{R}_{ori}^t$ is computed as

$$\begin{aligned} \Delta\mathcal{R}_{ori}^t &= \mathcal{T}(\mathcal{P}_h^0, \mathcal{P}_h^t) \\ \mathcal{R}_{ori}^t &= \Delta\mathcal{R}_{ori}^t \cdot \mathcal{R}_{ori}^0 \end{aligned} \tag{1}$$

where $\mathcal{T}$ computes the rigid transform between hand key points on the first frame of the video, $\mathcal{P}_h^0$, and $\mathcal{P}_h^t$. The robot end effector pose is then represented at $T_r^t \leftarrow \{\mathcal{R}_{pos}^t, \mathcal{R}_{ori}^t\}$. The robot's gripper state $\mathcal{R}_g$ is computed using the distance between the tip of the index finger and thumb. The gripper is considered closed when the distance is less than 7cm, otherwise open. Finally, given the robot pose T_r^t, we define a set of N rigid transformations T about the computed robot pose and compute robot key points $\mathcal{P}_r^t$ such that

$$(\mathcal{P}_r^t)^i = T_r^t \cdot T^i, \quad \forall i \in \{1, ..., N\} \tag{2}$$

This process has been demonstrated in Figure 1. This approach effectively bridges the morphological gap between human hands and robot manipulators, enabling accurate transfer of demonstrated actions to a robotic framework.

2) Environment state through point priors: To obtain key points on task-relevant objects in the scene, we adopt the method proposed by P3PO [45]. Initially, a user randomly selects one demonstration from a dataset of human videos and annotates semantically meaningful object points on the first frame that are pertinent to the task being performed. This annotation process is quick, taking only a few seconds. The user-annotated points serve as priors for subsequent data generation. Using an off-the-shelf semantic correspondence model, DIFT [76], we transfer the annotated points from the first frame to the corresponding locations in the first frames of all other demonstrations within the dataset. This approach allows us to initialize key points throughout the data set with minimal additional human effort.

For each demonstration, we then employ Co-Tracker [37], an off-the-shelf point tracker, to automatically track these initialized key points throughout the entire trajectory. By leveraging existing vision models for correspondence and tracking, we efficiently compute object key points for every frame in the dataset while requiring user input for only a single frame. This process, illustrated in Figure 2, capitalizes on large-scale pre-training of vision models to generalize across new object instances and scenes without necessitating further training. We prefer point tracking over correspondence at each frame due to its faster inference speed and its capability to handle occlusions by continuing to track points. The corresponding object points from two camera views are lifted to 3D world coordinates using point triangulation to obtain the 3D object key points $\mathcal{P}_o$. During inference, DIFT is employed to identify corresponding object key points on the first frame, followed by Co-Tracker tracking these points during execution. It is important to note that Point Policy utilizes multiple camera views only for point triangulation, with the policy being learned on 3D key points grounded in the robot's base frame.

B. Policy Learning

For policy learning, we use BAKU [28]. Instead of providing raw images as input, we provide the robot points $\mathcal{P}_r$ and object points $\mathcal{P}_o$ grounded in the robot's base frame as input to the policy. A history of observations for each key point is flattened into a single vector which is then encoded using a multilayer perceptron (MLP) encoder. The encoded representations are fed as separate tokens along with a gripper token into a BAKU [28] transformer policy, which predicts the future tracks for each robot point $\hat{\mathcal{P}}_r$ and the robot gripper state $\hat{\mathcal{G}}_r$ using a deterministic action head. Mathematically, this can be represented as

$$\begin{aligned} \mathcal{O}^{t-H:t} &= \{\mathcal{P}_r^{t-H:t},\ \mathcal{P}_o^{t-H:t}\} \\ \hat{\mathcal{P}}_r^{t+1},\ \mathcal{G}_r^{t+1} &= \pi(\cdot|\ \mathcal{O}^{t-H:t}) \end{aligned} \tag{3}$$

where H is the history length and π is the learned policy. Following prior works in policy learning [89, 14], we use

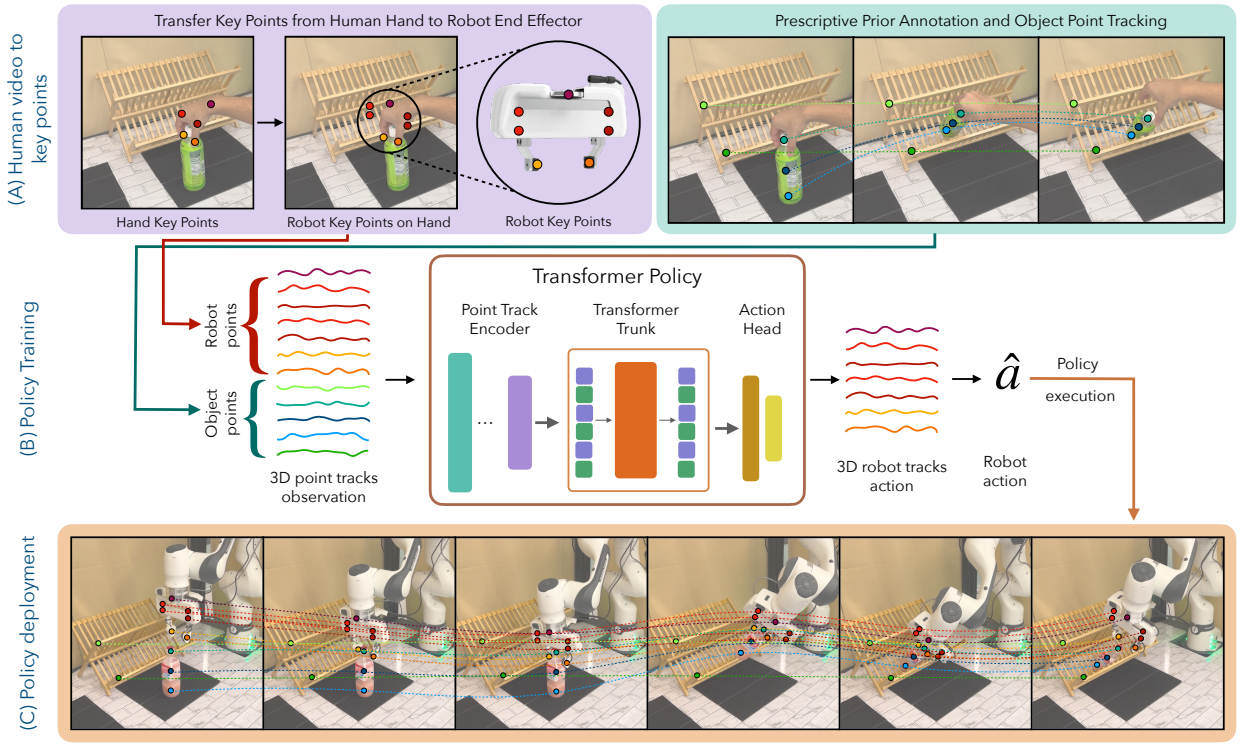

Fig. 1: Overview of the Point Policy framework. (a) Point Policy leverages state-of-the-art vision models and policy architectures to translate human hand poses into robot poses while capturing object states through sparse single-frame annotations. (b) The derived key points are fed into a transformer policy to predict the 3D future point tracks from which the robot actions are computed through rigid-body geometry constraints. (c) Finally, the computed action is executed on the robot using end-effector position control at a 6Hz frequency.

action chunking with exponential temporal averaging to ensure temporal smoothness of the predicted point tracks. The transformer is non-causal in this scenario and hence the training loss is only applied to the robot point tracks.

C. Backtrack Robot Actions from Predicted Key Points

The predicted robot points $\hat{\mathcal{P}}_r$ are mapped back to the robot pose using constraints from rigid-body geometry. We first consider the key point corresponding to the robot's wrist $\hat{\mathcal{P}}_r^{wrist}$ as the robot position $\hat{\mathcal{R}}_{pos}$. The robot orientation $\hat{\mathcal{R}}_{ori}$ is computed using Eq. 1 considering $\mathcal{R}_{ori}^0$ is fixed and known. Finally, the robot action $\mathcal{A}_r$ is defined as

$$\hat{\mathcal{A}}_r = (\hat{\mathcal{R}}_{pos},\ \hat{\mathcal{R}}_{ori},\ \hat{\mathcal{G}}_r) \tag{4}$$

Finally, the action $\hat{\mathcal{A}}_r$ is executed on the robot using end-effector position control at a 6Hz frequency.

III. Experiments

Our experiments are designed to answer the following questions: (1) How well does Point Policy work for policy learning? (2) How well does Point Policy work for novel object instances? (3) Can Point Policy handle background distractors? More analysis has been included in Appendix H.

A. Experimental Setup

Our experiments utilize a Franka Research 3 robot equipped with a Franka Hand gripper, operating in a real-world environment. We use the Deoxys [93] real-time controller for controlling the robot. The policies utilize RGB and RGB-D images captured using Intel RealSense D435 cameras from two third-person camera views. The action space encompasses the robot's end effector pose and gripper state. We collect a total of 190 human demonstrations across 8 real-world tasks, featuring diverse object positions and types. Additionally, for studying the effect of co-training with robot data (Section H4), we collect a total of 100 robot demonstrations for 4 tasks (Section H4) using a VR-based teleoperation framework [34]. All demonstrations are recorded at a 20Hz frequency and subsequently subsampled to approximately 6Hz. For methods that directly predict robot actions, we employ absolute actions during training, with orientation represented using a 6D rotation representation [91]. This representation is chosen for its continuity and fast convergence properties. The learned policies are deployed at a 6Hz frequency during execution. A description of the tasks and baselines has been provided in Appendix H.

B. How well does Point Policy work for policy learning?

We evaluate Point Policy in an in-domain setting, using the same objects seen during training. The evaluation consists of 10 trials per object for each task, resulting in a variable total number of trials per task. The results of this evaluation are summarized in Table I. Baselines that rely on RGB images as inputs (RGB, RGB-D, MT-π) perform poorly when trained exclusively on human hand videos. This is largely due to the significant visual differences between the human hand and the robot manipulator. While appearance-agnostic, P3-PO struggles

TABLE I: Policy performance of Point Policy on in-domain object instances on 8 real-world tasks.

Method	Close drawer	Put bread on plate	Fold towel	Close oven	Sweep broom	Put bottle on rack	Put bowl in oven	Make bottle upright
BC [28]	0/10	0/20	0/10	0/10	0/10	0/30	1/10	0/20
BC w/ Depth	0/10	0/20	0/10	0/10	0/10	0/30	0/10	0/20
MT-π [68]	2/10	2/20	0/10	4/10	0/10	8/30	0/10	0/20
P3-PO [45]	0/10	0/20	0/10	0/10	0/10	0/30	0/10	0/20
Point Policy **(Ours)**	**10/10**	**19/20**	**9/10**	**9/10**	**9/10**	**26/30**	**8/10**	**16/20**

TABLE II: Policy performance of Point Policy on novel object instances on 6 real-world tasks.

Method	Put bread on plate	Fold towel	Sweep broom	Put bottle on rack	Put bowl in oven	Make bottle upright
BC [28]	0/20	0/20	0/10	0/30	0/10	0/20
BC w/ Depth	0/20	0/20	0/20	0/30	0/10	0/20
MT-π [68]	1/20	0/20	0/10	0/30	0/10	0/20
P3-PO [45]	0/20	0/20	0/10	0/30	0/10	0/20
Point Policy **(Ours)**	**18/20**	**15/20**	**4/10**	**27/30**	**9/10**	**9/20**

TABLE III: Policy performance of Point Policy with background distractors on both in-domain and novel object instances.

Background distractors	Put bread on plate		Sweep broom		Put bottle on rack	
	In-domain	Novel object	In-domain	Novel object	In-domain	Novel object
✗	19/20	18/20	9/10	4/10	26/30	27/30
✓	18/20	18/20	9/10	2/10	23/30	23/30

due to noisy depth data from the camera. Point Policy achieves an average success rate of 88% across all tasks, outperforming the strongest baseline MT-π by 75%. Overall, these results demonstrate that Point Policy's ability to effectively address challenges related to visual differences and noisy depth data, achieving state-of-the-art performance in an in-domain setting.

C. How well does Point Policy work for novel object instances?

Table II compares the performance of Point Policy when evaluated on new object instances unseen in the training data. We perform this comparison on a subset of our tasks. We observe that Point Policy achieves an average success rate of 74% across all tasks, outperforming the strongest baseline by 73%. Compared to P3PO[45], where each task is trained with a variety of object sizes, most of our tasks are trained on a single object instance. Despite this limited diversity in the training data, Point Policy demonstrates robust generalization capabilities. Figure 4 depicts rollouts of Point Policy for novel object instances. For a visual reference of the novel object instances used for each task, please refer to Appendix H6. These results affirm Point Policy's strong generalization capabilities, making it suitable for real-world applications where encountering unseen objects is common.

D. Can Point Policy handle background distractors?

We evaluate the robustness of Point Policy in the presence of background clutter, as shown in Table III. This study is conducted on three tasks - *put bread on plate*, *sweep broom*, and *put bottle on rack*. Trials are conducted using both in-domain and novel object instances. Examples of the distractors used are illustrated in Figure 1, with Figure 4 depicting rollouts of Point Policy in the presence of background distractors. We observe that Point Policy is robust to background clutter, exhibiting either comparable performance or only minimal degradation in the presence of background distractors. This robustness can be attributed to Point Policy's use of point-based representations, which are decoupled from raw pixel values. By focusing on semantically meaningful points rather than image-level features, Point Policy enables policies that are resilient to environmental perturbations.

IV. Conclusion and Limitations

In this work, we presented Point Policy, a framework that enables learning robot policies exclusively from human videos, does not require real-world online interactions, and exhibits generalization to spatial variations, new object instances, and robustness to background clutter.

Limitations: We recognize a few limitations in this work: (1) Point Policy's reliance on existing vision models makes it susceptible to their failures. For instance, failures in hand pose detection or point tracking under occlusion have a detrimental effect on performance. However, with continued advances in computer vision, we believe that frameworks such as Point Policy will become stronger over time. (2) Point-based abstractions enhance generalization capabilities, but sacrifice valuable scene context information, which is crucial for navigating through cluttered or obstacle-rich environments. Future research focusing on developing algorithms that preserve sparse contextual cues in addition to the point abstractions in Point Policy might help address this. (3) While all our experiments are from a fixed third-person camera view, a large portion of human task videos on the internet are from an egocentric view [23, 50]. Extending Point Policy to egocentric

camera views can help us utilize these vast repositories of human videos readily available on the internet.

References

[1] Pieter Abbeel and Andrew Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the Twenty-First International Conference on Machine Learning*, ICML '04, page 1, New York, NY, USA, 2004. Association for Computing Machinery. ISBN 1581138385.

[2] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[3] Jake K Aggarwal and Quin Cai. Human motion analysis: A review. *Computer vision and image understanding*, 73 (3):428–440, 1999.

[4] Shikhar Bahl, Abhinav Gupta, and Deepak Pathak. Human-to-robot imitation in the wild. *arXiv preprint arXiv:2207.09450*, 2022.

[5] Dominik Bauer, Timothy Patten, and Markus Vincze. Reagent: Point cloud registration using imitation and reinforcement learning, 2021.

[6] Sarah Bechtle, Neha Das, and Franziska Meier. Multi-modal learning of keypoint predictive models for visual object manipulation. *IEEE Transactions on Robotics*, 39 (2):1212–1224, 2023.

[7] James Betker, Gabriel Goh, Li Jing, Tim Brooks, Jianfeng Wang, Linjie Li, Long Ouyang, Juntang Zhuang, Joyce Lee, Yufei Guo, et al. Improving image generation with better captions. *Computer Science. https://cdn. openai. com/papers/dall-e-3. pdf*, 2(3):8, 2023.

[8] Homanga Bharadhwaj, Jay Vakil, Mohit Sharma, Abhinav Gupta, Shubham Tulsiani, and Vikash Kumar. Roboagent: Generalization and efficiency in robot manipulation via semantic augmentations and action chunking. *arXiv preprint arXiv:2309.01918*, 2023.

[9] Homanga Bharadhwaj, Debidatta Dwibedi, Abhinav Gupta, Shubham Tulsiani, Carl Doersch, Ted Xiao, Dhruv Shah, Fei Xia, Dorsa Sadigh, and Sean Kirmani. Gen2act: Human video generation in novel scenarios enables generalizable robot manipulation. *arXiv preprint arXiv:2409.16283*, 2024.

[10] Homanga Bharadhwaj, Roozbeh Mottaghi, Abhinav Gupta, and Shubham Tulsiani. Track2act: Predicting point tracks from internet videos enables diverse zero-shot robot manipulation. *arXiv preprint arXiv:2405.01527*, 2024.

[11] Chethan Bhateja, Derek Guo, Dibya Ghosh, Anikait Singh, Manan Tomar, Quan Vuong, Yevgen Chebotar, Sergey Levine, and Aviral Kumar. Robotic offline rl from internet videos via value-function learning. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 16977–16984. IEEE, 2024.

[12] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

[13] Yuanpei Chen, Chen Wang, Li Fei-Fei, and C. Karen Liu. Sequential dexterity: Chaining dexterous policies for long-horizon manipulation, 2023.

[14] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and Systems (RSS)*, 2023.

[15] Cognition. Devin, 2025. URL https://devin.ai. Accessed: January 24, 2025.

[16] Coline Devin, Pieter Abbeel, Trevor Darrell, and Sergey Levine. Deep object-centric representations for generalizable robot learning. *CoRR*, abs/1708.04225, 2017.

[17] Carl Doersch, Yi Yang, Mel Vecerik, Dilara Gokay, Ankush Gupta, Yusuf Aytar, Joao Carreira, and Andrew Zisserman. Tapir: Tracking any point with per-frame initialization and temporal refinement. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10061–10072, 2023.

[18] Yan Duan, Marcin Andrychowicz, Bradly C. Stadie, Jonathan Ho, Jonas Schneider, Ilya Sutskever, Pieter Abbeel, and Wojciech Zaremba. One-shot imitation learning. *CoRR*, abs/1703.07326, 2017.

[19] Haritheja Etukuru, Norihito Naka, Zijin Hu, Seungjae Lee, Julian Mehu, Aaron Edsinger, Chris Paxton, Soumith Chintala, Lerrel Pinto, and Nur Muhammad Mahi Shafiullah. Robot utility models: General policies for zero-shot deployment in new environments. *arXiv preprint arXiv:2409.05865*, 2024.

[20] Hao-Shu Fang, Chenxi Wang, Hongjie Fang, Minghao Gou, Jirong Liu, Hengxu Yan, Wenhai Liu, Yichen Xie, and Cewu Lu. Anygrasp: Robust and efficient grasp perception in spatial and temporal domains. *IEEE Transactions on Robotics (T-RO)*, 2023.

[21] Xiaolin Fang, Bo-Ruei Huang, Jiayuan Mao, Jasmine Shone, Joshua B Tenenbaum, Tomás Lozano-Pérez, and Leslie Pack Kaelbling. Keypoint abstraction using large models for object-relative imitation learning. *arXiv preprint arXiv:2410.23254*, 2024.

[22] Yabo Fu, Yang Lei, Tonghe Wang, Walter J Curran, Tian Liu, and Xiaofeng Yang. Deep learning in medical image registration: a review. *Physics in Medicine & Biology*, 65 (20):20TR01, 2020.

[23] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, et al. Ego4d: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18995–19012, 2022.

[24] Kamal Gupta, Varun Jampani, Carlos Esteves, Abhinav Shrivastava, Ameesh Makadia, Noah Snavely, and Abhishek Kar. Asic: Aligning sparse in-the-wild image

collections. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4134–4145, October 2023.

[25] Irmak Guzey, Yinlong Dai, Georgy Savva, Raunaq Bhirangi, and Lerrel Pinto. Bridging the human to robot dexterity gap through object-oriented rewards. *arXiv preprint arXiv:2410.23289*, 2024.

[26] Siddhant Haldar, Vaibhav Mathur, Denis Yarats, and Lerrel Pinto. Watch and match: Supercharging imitation with regularized optimal transport. In *Conference on Robot Learning*, pages 32–43. PMLR, 2023.

[27] Siddhant Haldar, Jyothish Pari, Anant Rai, and Lerrel Pinto. Teach a robot to fish: Versatile imitation from one minute of demonstrations, 2023.

[28] Siddhant Haldar, Zhuoran Peng, and Lerrel Pinto. Baku: An efficient transformer for multi-task policy learning. *arXiv preprint arXiv:2406.07539*, 2024.

[29] Adam W. Harley, Zhaoyuan Fang, and Katerina Fragkiadaki. Particle video revisited: Tracking through occlusions using point trajectories. In *ECCV*, 2022.

[30] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. *CoRR*, abs/1606.03476, 2016.

[31] Shuaiyi Huang, Luyu Yang, Bo He, Songyang Zhang, Xuming He, and Abhinav Shrivastava. Learning semantic correspondence with sparse annotations. In *Proceedings of the European Conference on Computer Vision(ECCV)*, 2022.

[32] Wenlong Huang, Chen Wang, Yunzhu Li, Ruohan Zhang, and Li Fei-Fei. Rekep: Spatio-temporal reasoning of relational keypoint constraints for robotic manipulation. *arXiv preprint arXiv:2409.01652*, 2024.

[33] Ahmed Hussein, Mohamed Medhat Gaber, Eyad Elyan, and Chrisina Jayne. Imitation learning: A survey of learning methods. *ACM Comput. Surv.*, 50(2), apr 2017. ISSN 0360-0300.

[34] Aadhithya Iyer, Zhuoran Peng, Yinlong Dai, Irmak Guzey, Siddhant Haldar, Soumith Chintala, and Lerrel Pinto. Open teach: A versatile teleoperation system for robotic manipulation. *arXiv preprint arXiv:2403.07870*, 2024.

[35] Eric Jang, Alex Irpan, Mohi Khansari, Daniel Kappler, Frederik Ebert, Corey Lynch, Sergey Levine, and Chelsea Finn. Bc-z: Zero-shot task generalization with robotic imitation learning. In *Conference on Robot Learning*, pages 991–1002. PMLR, 2022.

[36] Yuanchen Ju, Kaizhe Hu, Guowei Zhang, Gu Zhang, Mingrun Jiang, and Huazhe Xu. Robo-abc: Affordance generalization beyond categories via semantic correspondence for robot manipulation. In *European Conference on Computer Vision*, pages 222–239. Springer, 2025.

[37] Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker: It is better to track together, 2023.

[38] Siddharth Karamcheti, Suraj Nair, Annie S Chen, Thomas Kollar, Chelsea Finn, Dorsa Sadigh, and Percy Liang. Language-driven representation learning for robotics. *arXiv preprint arXiv:2302.12766*, 2023.

[39] Andrej Karpathy. mingpt: A minimal pytorch re-implementation of the openai gpt. https://github.com/karpathy/minGPT, 2021.

[40] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4): 139–1, 2023.

[41] Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024.

[42] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything, 2023.

[43] Sateesh Kumar, Jonathan Zamora, Nicklas Hansen, Rishabh Jangir, and Xiaolong Wang. Graph inverse reinforcement learning from diverse videos. In *Conference on Robot Learning*, pages 55–66. PMLR, 2023.

[44] Seungjae Lee, Yibin Wang, Haritheja Etukuru, H Jin Kim, Nur Muhammad Mahi Shafiullah, and Lerrel Pinto. Behavior generation with latent actions. *arXiv preprint arXiv:2403.03181*, 2024.

[45] Mara Levy, Siddhant Haldar, Lerrel Pinto, and Abhinav Shirivastava. P3-po: Prescriptive point priors for visuo-spatial generalization of robot policies. *arXiv preprint arXiv:2412.06784*, 2024.

[46] Mara Levy, Nirat Saini, and Abhinav Shrivastava. Wayex: Waypoint exploration using a single demonstration, 2024.

[47] Yi Li, Yuquan Deng, Jesse Zhang, Joel Jang, Marius Memme, Raymond Yu, Caelan Reed Garrett, Fabio Ramos, Dieter Fox, Anqi Li, et al. Hamster: Hierarchical action models for open-world robot manipulation. *arXiv preprint arXiv:2502.05485*, 2025.

[48] Tony Lindeberg. *Scale Invariant Feature Transform*, volume 7. 05 2012. doi: 10.4249/scholarpedia.10491.

[49] Yixin Liu, Kai Zhang, Yuan Li, Zhiling Yan, Chujie Gao, Ruoxi Chen, Zhengqing Yuan, Yue Huang, Hanchi Sun, Jianfeng Gao, et al. Sora: A review on background, technology, limitations, and opportunities of large vision models. *arXiv preprint arXiv:2402.17177*, 2024.

[50] Yunze Liu, Yun Liu, Che Jiang, Kangbo Lyu, Weikang Wan, Hao Shen, Boqiang Liang, Zhoujie Fu, He Wang, and Li Yi. Hoi4d: A 4d egocentric dataset for category-level human-object interaction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21013–21022, 2022.

[51] Camillo Lugaresi, Jiuqiang Tang, Hadon Nash, Chris McClanahan, Esha Uboweja, Michael Hays, Fan Zhang, Chuo-Ling Chang, Ming Guang Yong, Juhyun Lee, et al. Mediapipe: A framework for building perception pipelines. *arXiv preprint arXiv:1906.08172*, 2019.

[52] Yecheng Jason Ma, Shagun Sodhani, Dinesh Jayaraman, Osbert Bastani, Vikash Kumar, and Amy Zhang. Vip: To-

wards universal visual reward and representation via value-implicit pre-training. *arXiv preprint arXiv:2210.00030*, 2022.

[53] Yecheng Jason Ma, Vikash Kumar, Amy Zhang, Osbert Bastani, and Dinesh Jayaraman. Liv: Language-image representations and rewards for robotic control. In *International Conference on Machine Learning*, pages 23301–23320. PMLR, 2023.

[54] Ajay Mandlekar, Yuke Zhu, Animesh Garg, Jonathan Booher, Max Spero, Albert Tung, Julian Gao, John Emmons, Anchit Gupta, Emre Orbay, et al. Roboturk: A crowdsourcing platform for robotic skill learning through imitation. In *Conference on Robot Learning*, pages 879–893. PMLR, 2018.

[55] Ajay Mandlekar, Danfei Xu, Roberto Martín-Martín, Silvio Savarese, and Li Fei-Fei. Learning to generalize across long-horizon tasks from human demonstrations. *CoRR*, abs/2003.06085, 2020.

[56] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

[57] Ashvin Nair, Abhishek Gupta, Murtaza Dalal, and Sergey Levine. Awac: Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*, 2020.

[58] Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. *arXiv preprint arXiv:2203.12601*, 2022.

[59] Andrew Y. Ng and Stuart J. Russell. Algorithms for inverse reinforcement learning. In *Proceedings of the Seventeenth International Conference on Machine Learning*, ICML '00, page 663–670, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc. ISBN 1558607072.

[60] David Nistér, Oleg Naroditsky, and James Bergen. Visual odometry. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, volume 1, pages I–I. Ieee, 2004.

[61] Abhishek Padalkar, Acorn Pooley, Ajinkya Jain, Alex Bewley, Alex Herzog, Alex Irpan, Alexander Khazatsky, Anant Rai, Anikait Singh, Anthony Brohan, et al. Open x-embodiment: Robotic learning datasets and rt-x models. *arXiv preprint arXiv:2310.08864*, 2023.

[62] Abhishek Padalkar, Acorn Pooley, Ajinkya Jain, Alex Bewley, Alex Herzog, Alex Irpan, Alexander Khazatsky, Anant Rai, Anikait Singh, Anthony Brohan, et al. Open x-embodiment: Robotic learning datasets and rt-x models. *arXiv preprint arXiv:2310.08864*, 2023.

[63] Georgios Papagiannis, Norman Di Palo, Pietro Vitiello, and Edward Johns. R+ x: Retrieval and execution from everyday human videos. *arXiv preprint arXiv:2407.12957*, 2024.

[64] Georgios Pavlakos, Dandan Shan, Ilija Radosavovic, Angjoo Kanazawa, David Fouhey, and Jitendra Malik. Reconstructing hands in 3d with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9826–9836, 2024.

[65] D Pomerleau. An autonomous land vehicle in a neural network. *Advances in Neural Information Processing Systems*, 1, 1998.

[66] Dean Pomerleau. Alvinn: An autonomous land vehicle in a neural network. In D.S. Touretzky, editor, *Proceedings of (NeurIPS) Neural Information Processing Systems*, pages 305 – 313. Morgan Kaufmann, December 1989.

[67] Machel Reid, Nikolay Savinov, Denis Teplyashin, Dmitry Lepikhin, Timothy Lillicrap, Jean-baptiste Alayrac, Radu Soricut, Angeliki Lazaridou, Orhan Firat, Julian Schrittwieser, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024.

[68] Juntao Ren, Priya Sundaresan, Dorsa Sadigh, Sanjiban Choudhury, and Jeannette Bohg. Motion tracks: A unified representation for human-robot transfer in few-shot imitation learning. *arXiv preprint arXiv:2501.06994*, 2025.

[69] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635. JMLR Workshop and Conference Proceedings, 2011.

[70] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems*, 35:36479–36494, 2022.

[71] Stefan Schaal. Learning from demonstration. *Advances in neural information processing systems*, 9, 1996.

[72] Nur Muhammad Mahi Shafiullah, Anant Rai, Haritheja Etukuru, Yiqian Liu, Ishan Misra, Soumith Chintala, and Lerrel Pinto. On bringing robots home. *arXiv preprint arXiv:2311.16098*, 2023.

[73] Nur Muhammad Mahi. Shafiullah, Siyuan. Feng, Lerrel. Pinto, and Russ. Tedrake. Supervised policy learning for real robots, July 2024. URL https://supervised-robot-learning.github.io. Tutorial presented at the Robotics: Science and Systems (RSS), Delft.

[74] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Cliport: What and where pathways for robotic manipulation. *CoRR*, abs/2109.12098, 2021.

[75] Kaustubh Sridhar, Souradeep Dutta, Dinesh Jayaraman, James Weimer, and Insup Lee. Memory-consistent neural networks for imitation learning. *arXiv preprint arXiv:2310.06171*, 2023.

[76] Luming Tang, Menglin Jia, Qianqian Wang, Cheng Perng Phoo, and Bharath Hariharan. Emergent correspondence from image diffusion. In *Thirty-seventh Conference on*

Neural Information Processing Systems, 2023.

[77] Faraz Torabi, Garrett Warnell, and Peter Stone. Recent advances in imitation learning from observation. *arXiv preprint arXiv:1905.13566*, 2019.

[78] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[79] Jonathan Tremblay, Thang To, Balakumar Sundaralingam, Yu Xiang, Dieter Fox, and Stan Birchfield. Deep object pose estimation for semantic robotic grasping of household objects. *CoRR*, abs/1809.10790, 2018.

[80] Stephen Tyree, Jonathan Tremblay, Thang To, Jia Cheng, Terry Mosier, Jeffrey Smith, and Stan Birchfield. 6-dof pose estimation of household objects for robotic manipulation: An accessible dataset and benchmark. In *International Conference on Intelligent Robots and Systems (IROS)*, 2022.

[81] Chuan Wen, Xingyu Lin, John So, Kai Chen, Qi Dou, Yang Gao, and Pieter Abbeel. Any-point trajectory modeling for policy learning. *arXiv preprint arXiv:2401.00025*, 2023.

[82] Hongtao Wu, Ya Jing, Chilam Cheang, Guangzeng Chen, Jiafeng Xu, Xinghang Li, Minghuan Liu, Hang Li, and Tao Kong. Unleashing large-scale video generative pre-training for visual robot manipulation. *arXiv preprint arXiv:2312.13139*, 2023.

[83] Mengda Xu, Zhenjia Xu, Yinghao Xu, Cheng Chi, Gordon Wetzstein, Manuela Veloso, and Shuran Song. Flow as the cross-domain manipulation interface. *arXiv preprint arXiv:2407.15208*, 2024.

[84] Alper Yilmaz, Omar Javed, and Mubarak Shah. Object tracking: A survey. *Acm computing surveys (CSUR)*, 38 (4):13–es, 2006.

[85] Chengbo Yuan, Chuan Wen, Tong Zhang, and Yang Gao. General flow as foundation affordance for scalable robot learning. *arXiv preprint arXiv:2401.11439*, 2024.

[86] Wentao Yuan, Jiafei Duan, Valts Blukis, Wilbert Pumacay, Ranjay Krishna, Adithyavairavan Murali, Arsalan Mousavian, and Dieter Fox. Robopoint: A vision-language model for spatial affordance prediction for robotics. *arXiv preprint arXiv:2406.10721*, 2024.

[87] Kevin Zakka, Andy Zeng, Pete Florence, Jonathan Tompson, Jeannette Bohg, and Debidatta Dwibedi. Xirl: Cross-embodiment inverse reinforcement learning. In *Conference on Robot Learning*, pages 537–546. PMLR, 2022.

[88] Tianhao Zhang, Zoe McCarthy, Owen Jow, Dennis Lee, Xi Chen, Ken Goldberg, and Pieter Abbeel. Deep imitation learning for complex manipulation tasks from virtual reality teleoperation, 2018.

[89] Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.

[90] Yang Zheng, Adam W. Harley, Bokui Shen, Gordon Wetzstein, and Leonidas J. Guibas. Pointodyssey: A large-scale synthetic dataset for long-term point tracking. In *ICCV*, 2023.

[91] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5745–5753, 2019.

[92] Yifeng Zhu, Zhenyu Jiang, Peter Stone, and Yuke Zhu. Learning generalizable manipulation policies with object-centric 3d representations. In *7th Annual Conference on Robot Learning*, 2023.

[93] Yifeng Zhu, Abhishek Joshi, Peter Stone, and Yuke Zhu. Viola: Imitation learning for vision-based manipulation with object proposal priors. In *Conference on Robot Learning*, pages 1199–1210. PMLR, 2023.

[94] Barbara Zitova and Jan Flusser. Image registration methods: a survey. *Image and vision computing*, 21(11): 977–1000, 2003.

APPENDIX

A. Related Works

1) Imitation Learning: Imitation Learning (IL) [33] refers to training policies with expert demonstrations, without requiring a predefined reward function. In the context of reinforcement learning (RL), this is often referred to as inverse RL [59, 1], where the reward function is derived from the demonstrations and used to train a policy [46, 26, 27, 30, 57]. While these methods reduce the need for extensive human demonstrations, they still suffer from significant sample inefficiency. As a result of this inefficiency in deploying RL policies in the real world, behavior cloning (BC) [66, 77, 71, 69] has become increasingly popular in robotics. Recent advances in BC have demonstrated success in learning policies for both long-horizon tasks [13, 55, 74] and multi-task scenarios [28, 8, 62, 10, 9]. However, most of these approaches rely on image-based representations [88, 28, 14, 8, 62, 35], which limits their ability to generalize to new objects and function effectively outside of controlled lab environments. In this work, we propose Point Policy, which attempts to address this reliance on image representations by directly using key points as an input to the policy instead of raw images. Through extensive experiments, we observe that such an abstraction helps learn robust policies that generalize across varying scenarios.

2) Object-centric Representation Learning: Object-centric representation learning aims to create structured representations for individual components within a scene, rather than treating the scene as a whole. Common techniques in this area include segmenting scenes into bounding boxes [16, 55, 18, 20, 93] and estimating object poses [79, 80]. While bounding boxes show promise, they share similar limitations with non object-centric image-based models, such as overfitting to specific object instances. Pose estimation, although less prone to overfitting, requires separate models for each object in a task. Another popular method involves using point clouds [92, 5], but their high dimensionality necessitates specialized models, making it difficult to accurately capture spatial relationships. Lately, several works have resorted to adopting key points [45, 36, 32, 10, 9, 68, 21, 6, 86, 47, 81, 85, 83] for policy learning due to their generalization ability. Further, key points also allow the direct injection of human priors into the policy learning pipeline [10, 9, 68] as opposed to learning representations from human videos followed by downstream learning on robot teleoperated data [58, 11, 82, 52, 53, 38]. In this work, we leverage key points as a unified observation and action space to enable learning generalizable policies exclusively from human videos.

3) Human-to-Robot Transfer for Policy Learning: There have been several attempts at learning robot policies from human videos. Some works first learn visual representations from large-scale human video datasets and learn a downstream policy on these representations using limited amounts of robot data [58, 11, 82, 52, 53, 38]. Another line of work learns coarse policies from human videos, using key points [10] and generative modeling [9], which are then improved using downstream learning on robot data. Recently proposed MT-π [68] alleviates the need for downstream learning by co-training a key point policy with human and robot data. A caveat in all these works is that despite having access to abundant human demonstrations, there is a need to collect robot data to achieve a highly performant policy. A recently emerging line of work [63] attempts to do away with this need for robot data by doing in-context learning with state-of-the-art vision-language models (VLMs) [67, 2, 78]. However, owing to the large compute times of VLMs, these policies are required to be deployed open-loop and hence, are not reactive to changes in the scene. In this work, we propose Point Policy, a new framework that learns generalizable policies from human videos, does not require robot demonstrations or online robot interactions, and can be executed in a closed-loop fashion.

B. Background

C. Imitation learning

The goal of imitation learning is to learn a behavior policy π^b given access to either the expert policy π^e or trajectories derived from the expert policy τ^e. This work operates in the setting where the agent only has access to observation-based trajectories, i.e. $\tau^e \equiv \{(o_t, a_t)_{t=0}^T\}_{n=0}^N$. Here N and T denote the number of demonstrations and episode timesteps respectively. We choose this specific setting since obtaining observations and actions from expert or near-expert demonstrators is feasible in real-world settings [89, 34] and falls in line with recent work in this area [28, 44, 89, 14, 75].

D. Behavior Cloning

Behavior Cloning (BC) [65, 73] corresponds to solving the maximum likelihood problem shown in Eq. 5. Here $\mathcal{T}^e$ refers to expert demonstrations. When parameterized by a normal distribution with fixed variance, the objective can be framed as a regression problem where, given observations o^e, π^{BC} needs to output a^e.

$$\mathcal{L}^{BC} = \mathbb{E}_{(o^e,a^e)\sim\mathcal{T}^e} \|a^e - \pi^{BC}(o^e)\|^2 \tag{5}$$

After training, it enables π^{BC} to mimic the actions corresponding to the observations seen in the demonstrations.

1) Semantic Correspondence: Finding corresponding points across multiple images of the same scene is a well-established problem in computer vision [48, 94]. Correspondence is essential for solving a range of larger challenges, including 3D reconstruction [56, 40], motion tracking [37, 29, 90, 17], image registration [94], and object recognition [42]. In contrast, semantic correspondence focuses on matching points between a source image and an image of a different scene (e.g., identifying the left eye of a cat in relation to the left eye of a dog). Traditional correspondence methods [94, 48] often struggle with semantic correspondence due to the substantial differences in features between the images. Recent advancements in semantic correspondence utilize deep learning and dense correspondence techniques to enhance robustness [22, 31, 24] across variations in background, lighting, and camera perspectives. In this work, we adopt a diffusion-based point correspondence model, DIFT [76], to establish correspondences between a reference and an observed image, which is illustrated in Figure 2.

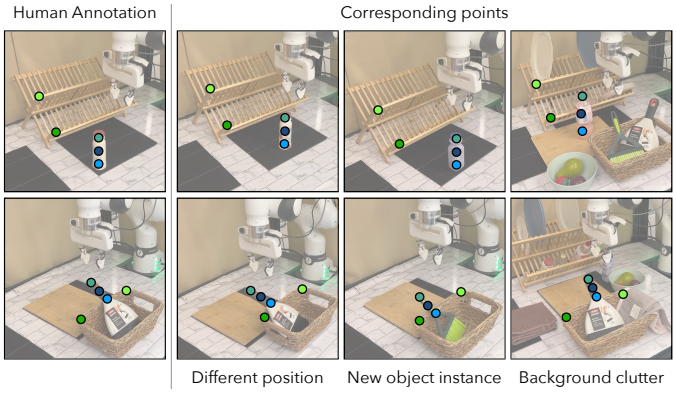

Fig. 2: Results of the correspondence model when used for the put bottle on rack and sweep broom tasks. On the left is a frame with human annotations for the object points. On the right, we show that semantic correspondence can identify the same points across different positions, new object instances, and background clutter.

2) Point Tracking: Point tracking across videos is a problem in computer vision, where a set of reference points are given in the first frame of the video, and the task is to track these points across multiple frames of the video sequence. Point tracking has proven crucial for many applications, including motion analysis [3], object tracking [84], and visual odometry [60]. The goal is to establish reliable correspondences between points in one frame and their counterparts in subsequent frames, despite challenges such as changes in illumination, occlusions, and camera motion. While traditional point tracking methods rely on detecting local features in images, more recent advancements leverage deep learning and dense correspondence methods to improve robustness and accuracy [37, 29, 90]. In this work, we use Co-Tracker [37] to track a set of reference points defined in the first frame of a robot's trajectory. These points tracked through the entire trajectory are then used to train generalizable robot policies for the real world.

E. Algorithmic Details

1) Point Triangulation: Point triangulation is a fundamental technique in computer vision used to reconstruct 3D points from their 2D projections in multiple images. Given n cameras with known projection matrices $P_1, P_2, ..., P_n$ and corresponding 2D image points $x_1, x_2, ..., x_n$, the goal is to find the 3D point X that best explains these observations.

The projection of X onto each image is given by:

$$x_i \sim P_i X$$

where $\sim$ denotes equality up to scale.

One common approach is the Direct Linear Transform (DLT) method:

1) For each view i, we can form two linear equations:

$$x_i(p_i^3 \cdot X) - (p_i^1 \cdot X) = 0$$

$$y_i(p_i^3 \cdot X) - (p_i^2 \cdot X) = 0$$

where p_i^j is the j-th row of P_i.

2) Combining equations from all views, we get a system $AX = 0$.
3) The solution is the unit vector corresponding to the smallest singular value of A, found via Singular Value Decomposition (SVD).

For optimal triangulation, we aim to minimize the geometric reprojection error.

F. Hyperparameters

The complete list of hyperparameters is provided in Table IV. Details about the number of demonstrations for each task has been included in Section H1, and summarized in Table V. All the models have been trained using a single NVIDIA RTX A4000 GPU.

G. Implementation Details for MT-π

Since the official implementation of MT-π is not yet public available, we adopt the Diffusion Transformer (DiT) based implementation of a 2D point track prediction model proposed by Bharadhwaj et al. [10]. We modify the architecture such that given a single image observation and robot motion tracks on the image, the model predicts future tracks of the robot points. These robot tracks are then converted to 3D using corresponding tracks for two camera views. The robot action is then computed from the 3D robot tracks using the same rigid-body geometry constraints as Point Policy (described in Section II-C). MT-π proposes the use of a key point retargeting network in order to convert the human hand and robot key points to the same space. Since we already convert the human hand key points to the corresponding robot points for Point Policy, we directly use these converted robot points instead of learning a separate keypoint retargeting network.

To ensure the correctness of our implementation, we evaluate MT-π in a setting identical to the one described in their paper. We conduct this evaluation on the *put bread on plate* task. We use 30 robot teleoperated demonstrations in addition to the human demonstrations, resulting in a total of 60 demonstrations. We observed a performance of 18/20, thus, confirming the correctness of the implementation.

H. Experiments

1) Task Descriptions: We experiment with manipulation tasks with significant variability in object position, type, and background context. Figure 3 depicts rollouts for all of our tasks. For each task, we collect data across various object sizes and appearances. During evaluations, we add novel object instances that are unseen during training. The variations in positions and object instances for selected tasks are depicted in Figure 5 and Figure 6. We provide a brief description of each task below.

a) Close drawer: The robot arm is tasked with pushing close a drawer placed on the table. The position of the drawer varies for each evaluation. We collect 20 demonstrations for a single drawer and run evaluations on the same drawer.

TABLE IV: List of hyperparameters.

Parameter	Value
Learning rate	$1e^{-4}$
Image size	256×256 (for BC, BC w/ Depth, MT-π)
Batch size	64
Optimizer	Adam
Number of training steps	100000
Transformer architecture	minGPT [39] (for BC, BC w/ Depth, P3PO, Point Policy)
	Diffusion Transformer [10] (for MT-π)
Hidden dim	256
Observation history length	1 (for BC, BC w/ Depth)
	10 (for MT-π, P3PO, Point Policy)
Action head	MLP
Action chunk length	20

TABLE V: Number of demonstrations.

Task	Number of object instances	Total number of demonstrations
Close drawer	1	20
Put bread on plate	1	30
Fold towel	1	20
Close oven	1	20
Sweep broom	1	20
Put bottle on rack	2	30
Put bowl in oven	1	20
Make bottle upright	2	30

b) Put bread on plate: The robot arm picks up a piece of bread from the table and places it on a plate. The positions of the bread and the plate are varied for each evaluation. We collect 30 demonstrations for the task of a single bread-plate pair. During evaluations, we introduce two new plates.

c) Fold towel: The robot arm picks up a towel placed on the table from a corner and folds it. The position of the towel varies for each evaluation. We collect 20 demonstrations for a single towel. During evaluations, we introduce two new towels.

d) Close oven: The robot arm is tasked with closing the door of an oven. The position of the oven varies for each evaluation. We collect 20 demonstrations for the task on a single oven and run evaluations on the same oven.

e) Sweep broom: The robot arm picks up a broom and sweeps the table. The position and orientation of the broom are varied across evaluations. We collect 20 demonstrations for a single broom. During evaluations, we introduce a new broom.

f) Put bottle on rack: The robot arm picks up a bottle from the table and places it on the lower level of a kitchen rack. The position of the bottle is varied for each evaluation. We collect 15 demonstrations for 2 different bottles, resulting in a total of 30 demonstrations for the task. During evaluations, we introduce three new bottles.

g) Put bowl in oven: The robot arm picks up a bowl from the table and places it inside an oven. The position of the bowl varies for each evaluation. We collect 20 demonstrations for the task with a single bowl. During evaluations, we introduce a new bowl.

h) Make bottle upright: The robot arm pick up a bottle from the table and places it in an upright position. The position of the bottle varies for each evaluation. We collect 15 demonstrations for 2 different bottles, resulting in a total of 30 demonstrations for the task. During evaluations, we introduce two new bottles.

2) Baselines: We compare Point Policy with 4 baselines - *behavior cloning (BC)* [28] with RGB and RGB-D images, *Motion Tracks* [68], and *P3-PO* [45]. We describe each method below.

a) Behavior Cloning (BC) [28]: This method performs behavior cloning (BC) using the BAKU policy learning architecture [28], which takes RGB images of the human hand as input and predicts the extracted robot actions as output.

b) Behavior Cloning (BC) with Depth: This is similar to BC but uses both RGB and depth images as input.

TABLE VI: Policy performance of Point Policy with teleoperated robot data on in-domain object instaces.

Demonstrations	Put bread on plate	Fold towel	Sweep broom	Make bottle upright
Human	19/20	**9/10**	**9/10**	**16/20**
Robot	18/20	**9/10**	4/10	12/20
Human + Robot	**20/20**	**9/10**	8/10	8/20

c) Motion Track Policy (MT-π) [68]: Given an image of the scene and robot key points on the image, MT-π predicts the future 2D robot point tracks to complete a task. This approach generates future 2D point tracks for robot points across multiple views, which are then triangulated to obtain 3D points on the robot. These 3D points are subsequently converted to the robot's absolute pose (similar to our proposed method) and treated as the robot's action. Implementation details for MT-π have been provided in Appendix G.

d) P3-PO [45]: This method utilizes image points representing both the robot and objects of interest, projecting them into 3D space using camera depth information. These 3D points serve as input to a transformer policy [28], which predicts robot actions. P3PO's 3D point representations, akin to those in Point Policy, enable spatial generalization, adaptability to novel object instances, and robustness to background clutter.

3) Considerations for policy learning: Point Policy and P3PO use a point-based representation obtained from 640×480 images. For correspondence, we use DIFT [76] using the first layer of the hundredth diffusion time step with an ensemble size of 8. Point tracking is performed using a modified version of Co-Tracker [37] that enables tracking one frame at a time, rather than chunks. Point Policy, MT-π, and P3PO use a history of 10 point observations, while the image-based baselines do not use history [28]. BC (RGB), BC (RGB-D), and MT-π are trained on images of size 256×256. All methods predict an action chunk [89] of size 20 ($\sim$ 3 seconds).

4) Can Point Policy be improved with robot demonstrations?: Table VI investigates whether Point Policy's performance can be enhanced through co-training with teleoperated robot data, collected using a VR-based teleoperation framework [34]. We conduct this study on four tasks - *put bread on plate*, *fold towel*, *sweep broom*, and *make bottle upright*. For each task, we collect an equal number of robot demonstrations as human demonstrations, resulting in 30, 20, 20, and 30 demonstrations respectively. Interestingly, our findings reveal that for tasks involving complex motions, such as *sweep broom* and *make bottle upright*, policies trained solely on robot data perform poorly with the same amount of data as compared to those trained exclusively on human data. This drop in performance stems from the complex motions in these tasks making it harder to collect robot data using VR teleoperation, resulting in noisy demos. These results highlight an important consideration: humans and robots may execute the same task in different ways. Consequently, co-training with both human and robot data requires the development of algorithms capable of dealing with these differences effectively.

TABLE VII: The effect of triangulated depth on P3PO and Point Policy.

Method	Put bread on plate	Sweep broom	Put bottle on rack
P3PO	0/20	0/10	0/30
P3PO + Triangulated Depth	17/20	4/10	23/30
Point Policy	**19/20**	**9/10**	**26/30**
Point Policy - Triangulated Depth	0/20	0/10	0/30

TABLE VIII: Importance of object point inputs for policy learning.

Method	Close drawer	Put bread on plate	Fold towel	Make bottle upright
MT-π	2/10	2/20	0/10	0/20
MT-π + object points	8/10	1/20	6/10	2/20
Point Policy	**10/10**	**19/20**	**9/10**	**16/20**

5) What design choices matter for human-to-robot learning?: This section examines the impact of key design decisions on learning from human videos.

a) Depth Sensing: In Point Policy, we utilize point triangulation from two camera views to obtain 3D key points, rather than relying on depth maps from the camera. We hypothesize that noisy camera depth leads to imprecise 3D key points, resulting in unreliable actions. Table VII tests this hypothesis on 4 real-world tasks by comparing the performance of P3PO and Point Policy with and without triangulated depth. We observe that adding triangulated depth to P3PO improves its performance from 0% to 72%. Further, removing triangulated depth from Point Policy reduces its performance from 90% to 0%. These results emphasize the importance of obtaining accurate 3D key points from human hands when learning robot policies from human videos. Appendix H7 includes an illustration of imprecise actions resulting from noisy sensor depth.

b) Significance of Object Points: While Point Policy uses robot and object key points as input to the policy, MT-π [68], the best-performing baseline in Table I, only uses robot key points and obtains information about the rest of the scene through an input image. We hypothesize that using object points can improve policy learning performance, especially when there is a morphology gap between data collection and inference. Table VIII tests this hypothesis by providing object points in addition to the robot points already passed as input into MT-π. We observe that adding object points improves the performance of MT-π on select tasks(comprehensive results on all tasks included in Appendix H8), suggesting that including object points in the input offers a potential advantage. Nevertheless, Point Policy outperforms both methods by 68% across all tasks, emphasizing the efficacy of predicting 3D key points rather than 2D key points in image space.

6) Illustration of Spatial Generalization and Novel Object Instances: Figure 5 and Figure 6 illustrate the variations in object positions and novel object instances used for each task, respectively.

7) Illustration of Depth Discrepancy: Figure 7 provides an illustration of the discrepancy in actions obtained from sensor depth and triangulated depth for the task of putting a bottle on the rack. We observe that the noise in sensor depth leads to

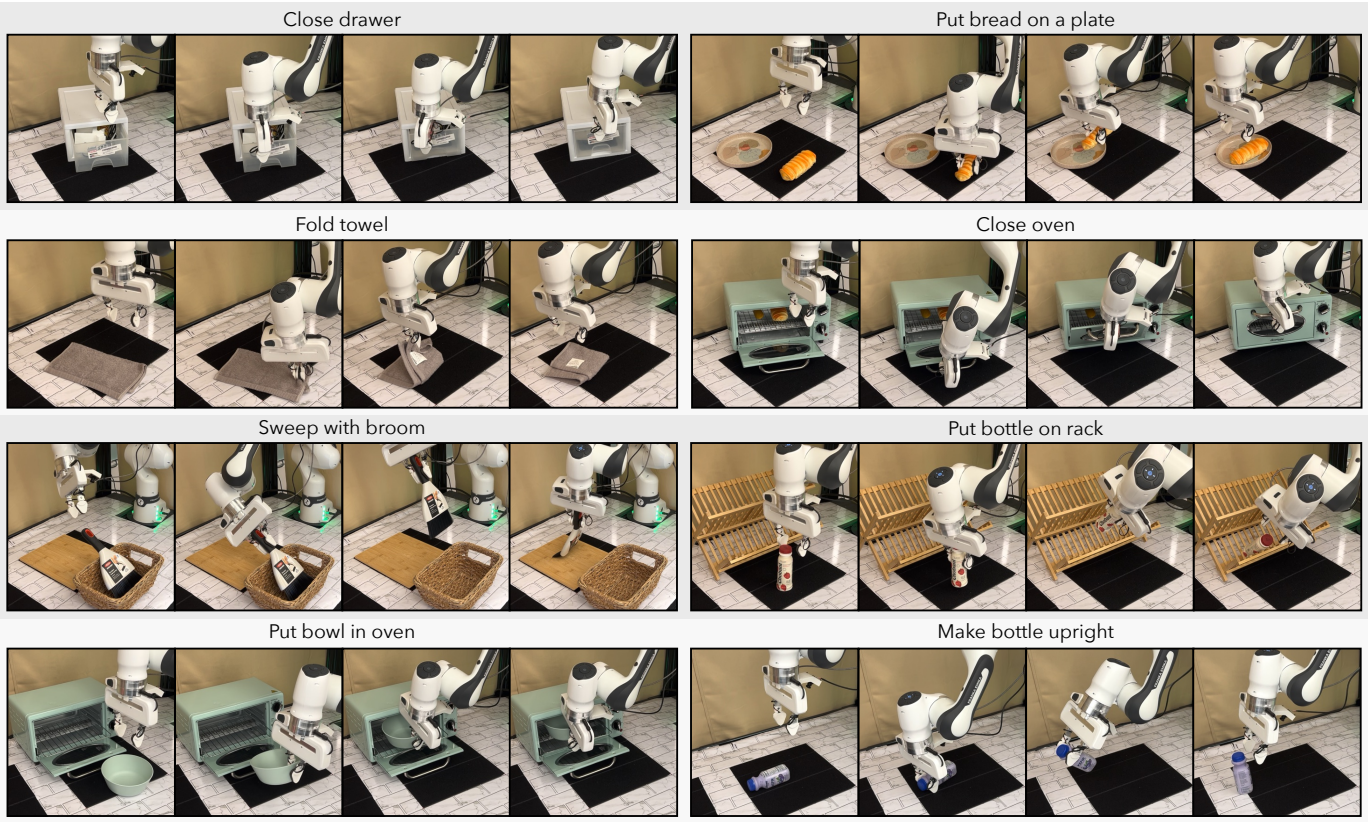

Fig. 3: Real-world rollouts showing Point Policy's ability on in-domain objects across 8 real-world tasks.

noise in robot points which is turn results in unreliable actions.

8) Significance of Object Points: Table IX and Table X study the performance of MT-π with and without object points and Point Policy across all of our tasks. We observe that MT-π with object points outperforms MT-π on select tasks, suggesting that including object points in the input offers a potential advantage.

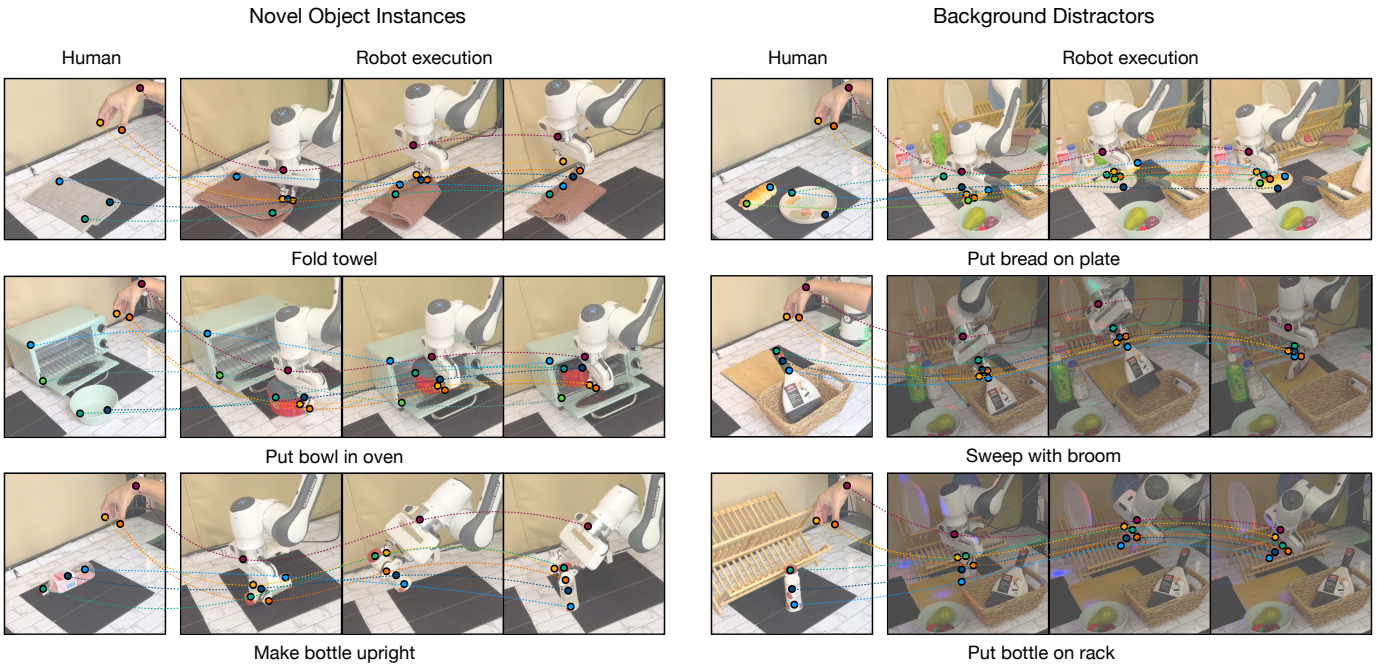

Fig. 4: Real-world rollouts showing that Point Policy generalizes to novel object instances and is robust to background distractors.

Fig. 5: Illustration of spatial variation used in our experiments.

TABLE IX: In-domain policy performance

Method	Close drawer	Put bread on plate	Fold towel	Close oven	Sweep broom	Put bottle on rack	Put bowl in oven	Make bottle upright
MT-π [68]	2/10	2/20	0/10	4/10	0/10	8/30	0/10	0/20
MT-π + object points	1/20	6/10	1/20	4/10	0/10	0/10	2/20	8/10
Point Policy **(Ours)**	**10/10**	**19/20**	**9/10**	**9/10**	**9/10**	**26/30**	**8/10**	**16/20**

TABLE X: Policy performance on novel object instances

Method	Put bread on plate	Fold towel	Sweep broom	Put bottle on rack	Put bowl in oven	Make bottle upright
MT-π [68]	1/20	0/20	0/10	0/30	0/10	0/20
MT-π + object points	2/20	0/20	0/20	1/10	0/10	1/20
Point Policy **(Ours)**	**18/20**	**15/20**	**4/10**	**27/30**	**9/10**	**9/20**

Fig. 6: Illustration of objects used in our experiments. For each task, on the left are in-domain objects while on the right are novel objects used in our generalization experiments.

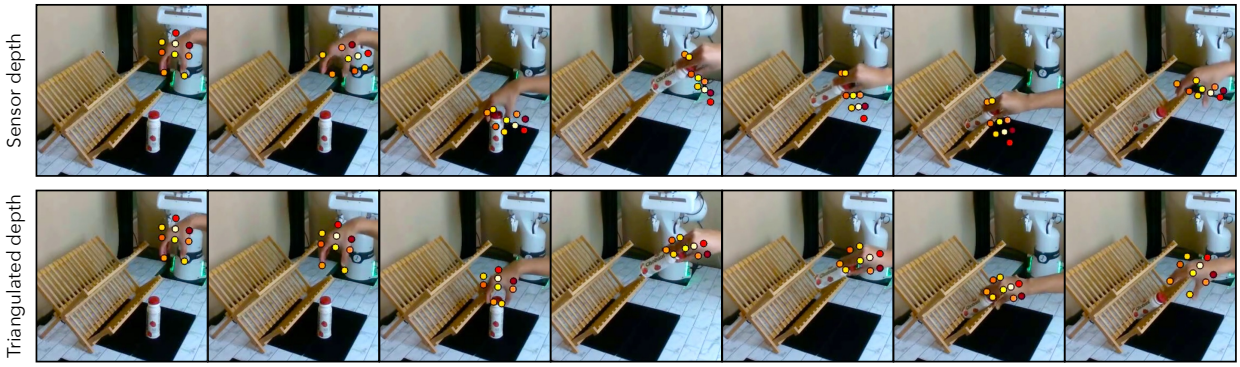

Fig. 7: Illustration of discrepancy in actions obtained from sensor depth and triangulated depth for the task of putting a bottle on the rack.