# Generalization of Manipulation Skills using Keypoint Priors

**Giovanni Franzese  Daniel Sotelo  Mohammed Osama Elnour  Pavel Kopanev**
**Abdul Rehman  Mothilal Asokan  Danilo Caporale  Paola Ardon**

Autonomous Robotics Research Center, Technology Innovation Institute, Abu Dhabi, UAE
{giovanni.franzese, daniel.aguirre, paola.ardon}@tii.ae

**Abstract:** Generalizing robot behavior across diverse contexts remains a core challenge in robotics. A common workaround is to collect large-scale datasets, effectively avoiding extrapolation by covering more of the input space. However, this strategy requires increasingly complex and opaque models without guaranteeing robustness under out-of-distribution conditions. In this work, we propose grounding behavioral models in 3D space using a sparse set of semantic keypoints. This representation provides a compact and structured encoding of motion that supports improved generalization in low-data regimes. We evaluate the use of nonlinear mappings between source and target keypoints to transport demonstrations across different scenes without requiring additional supervision. Experiments are conducted on non-prehensile manipulation and pick-and-place tasks.

**Keywords:** Transportation Maps, 3D Keypoint Priors, Zero-Shot Generalization

## 1 Introduction

One of the central challenges in robot learning is the generalization of manipulation policies to novel environments and object configurations. Although large-scale demonstrations have fueled impressive progress, the ability to generalize with limited data and high precision remains limited, especially in scenarios requiring geometric or structural variation reasoning.

This work focuses on the problem of generalizing manipulation behaviors across deformed task configurations by leveraging keypoint-based representations. These representations are sparse, interpretable, and allow a flexible description of object geometries and poses across different settings [1]. We propose a method that learns a transportation map that aligns keypoints from a reference task configuration, e.g., a starting and goal object configuration, to a novel, unseen configuration. This enables generalization of policy behavior without requiring full retraining or large-scale data collection. To enable this generalization, we incorporate three key types of priors: structural, matching, and transportation priors.

**Structural prior:** We assume that the task space can be encoded in a sparse set of keypoints whose configuration captures the relevant geometry of the environment. Keypoints observed during training define the source distribution, while those required at test time define the target distribution. These keypoints can be predicted by a deep neural network trained with human-annotated examples that mark object affordances [1, 2]. Alternatively, they can be obtained from feature correspondences between source and target images using Vision Transformers [3, 4], or from the foundation model itself [5]. Keypoints can also be discovered through robot interaction with 3D point clouds in a self-supervised, trial-and-error manner [6], or by identifying constrained anchors across a few demonstration videos [7].

**Matching prior:** We assume that for each new configuration, there exists a consistent correspondence (or match) between the source and target keypoints, such that aligning them captures the

Workshop on Generalizable Priors for Robot Manipulation at 9th Conference on Robot Learning, Seoul, Korea.

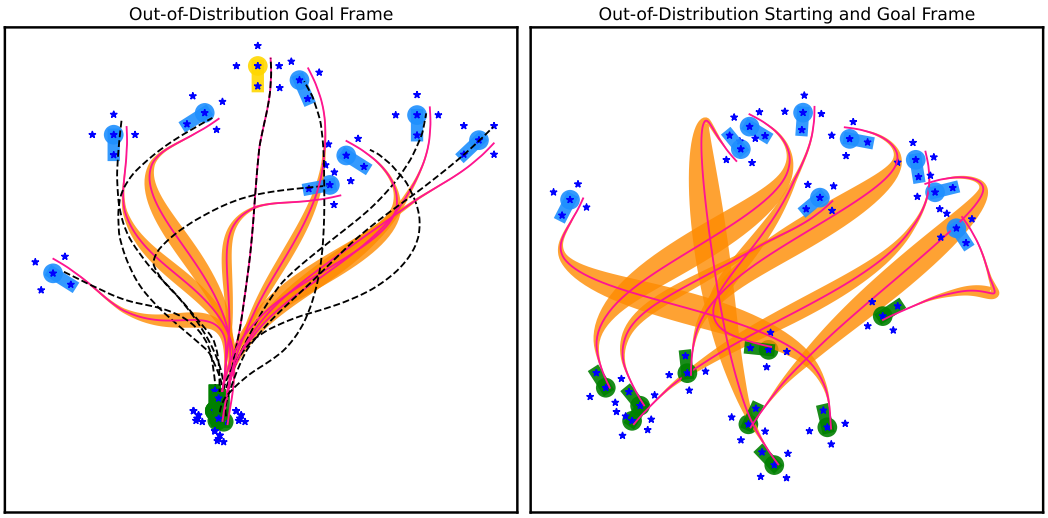


Figure 1: The **yellow** frame is the *source* goal frame, the **blue** frames are the *target* goal frames and the **green** frames are the start frames. Five keypoints are extracted per frame; that is, each demonstration is parameterized with ten keypoints. The **left** picture depicts the trajectory generalization when the goal frame is moved. The dashed line is the "ground truth" human demonstration. The **right** picture depicts the motion generalization when both frames are perturbed. The orange area highlights the uncertainty quantification in the transportation process. The uncertainty grows when transporting points of the trajectory that are further away from any task keypoint.

underlying transformation between task instances. This matching prior allows us to frame the generalization problem as keypoint alignment via a learned deformation field, similar in spirit to flow-based models in computer vision [8, 9].

**Transportation Prior:** Although keypoints provide a more expressive description of the task [1], if they do not move rigidly, we cannot simply roto-translate our original policy [3, 10]. This implies that variations between task instances (e.g., object deformations or locations) must be modeled as smooth, nonlinear deformations.

Our contribution lies in showing that a combination of keypoint-based structural and transportation priors can enable zero-shot generalization to novel task instances. We do not rely on planning algorithms [5, 6], sim-to-real reinforcement learning [11], or prompting large language models with the source, target, and trajectories to retrieve the generalized behavior [4]. Instead, we rely solely on fitting a nonlinear transformation in the context of imitation learning to "transport" the original labels, demonstrating successful zero-shot motion generalization. This article contributes to the transportation theory [12] with novel formalism on learning from multiple source demonstrations (Sec. 2.3), novel experiments on Push-T task (Sec. 3.1), and on real robot packing arm (Sec. 3.3).

## 2 Policy Transportation

We can learn a policy not only as a function of the robot's state, but also as a function of descriptive features of the environment, namely, the keypoints of an object's position or the goal position being tracked. We can define a generic policy as a predictive distribution conditioned on the current state and the data that were observed during data collection. For example, we can predict the desired robot goal, as a function of the current state, conditioning (or training) on a set of demonstration labels, i.e.,

$$\underbrace{x_g}_{\text{action}} = f\left(\underbrace{x}_{\text{state}} \mid \underbrace{\mathcal{X}}_{\text{labels}}\right). \tag{1}$$

When integrating task parameters, such as keypoints, the policy is going to be a function of the training (source) keypoints, $\mathcal{S}$, and of the testing (target) keypoints, $\mathcal{T}$, i.e.,

$$x_g = \mathcal{F}\left(\begin{bmatrix} x \\ \mathcal{T} \end{bmatrix} \middle\| \begin{bmatrix} \mathcal{X} \\ \mathcal{S} \end{bmatrix}\right). \tag{2}$$

With potentially hundreds of keypoints serving as input to the policy, a model that concatenates them may struggle with out-of-distribution configurations and is prone to the curse of dimensionality. Our goal is to have a policy that is not explicitly parameterized with respect to the keypoints $\mathcal{S}$ and $\mathcal{T}$. We encode the dependency on the keypoint parameters by fitting and exploiting a differential map. We call this a transportation map, and we use it to *transport* our policy from one situation to another.

### 2.1 Learning a Transportation Map

Given an original policy (1) that was trained with a set of labels $\mathcal{X}$ in the context $\mathcal{S}$, we retrieve the desirable action goal given the current context $\mathcal{T}$. By exploiting the existence of a transportation map,

$$\boldsymbol{\phi}_{\mathcal{S}\to\mathcal{T}} : \mathbb{R}^3 \to \mathbb{R}^3 \text{ s.t. } \boldsymbol{\mathcal{T}} = \boldsymbol{\phi}(\boldsymbol{\mathcal{S}}), \tag{3}$$

we can learn to convert the current state in the target context into the source context and project the original action in the new context.

In other words, with this map we can deform the space such that source and target keypoints are overlapping, hence for the policy there would be no change in the task parametrization, since $\boldsymbol{\mathcal{T}} = \boldsymbol{\phi}(\boldsymbol{\mathcal{S}})$. This satisfies our goal of removing the explicit dependency of the motion policy from the parameterization keypoints.

The map $\phi$ is generally a nonlinear function approximator, e.g., a Gaussian Process, a multilayer perceptron, a Bijective Flow [13], an Iterative Locally Weighted Regressor [14], etc. The nature of the regressor influences the inductive bias of the transportation, for example, enforcing invertibility, quantifying uncertainties, or having a zero-transportation prior when going out of distribution.

We can approach the problem from two different perspectives: **online** and **offline** policy transportation. The online perspective consists of applying the transportation to the state-action inputs, while the offline perspective applies the transportation to the training demonstration and re-trains a new policy with them, or simply executes the transported demonstration.

### 2.2 Online vs. Offline Policy Transportation

The online approach applies the (inverse of the) transportation to the current state-action of the trained policy. Mathematically,

$$\phi^{-1}(\boldsymbol{x}_g) = \mathcal{F}\left(\phi^{-1}\left(\begin{bmatrix} \boldsymbol{x} \\ \mathcal{T} \end{bmatrix}\right) \middle\| \begin{bmatrix} \mathcal{X} \\ \mathcal{S} \end{bmatrix}\right). \tag{4}$$

This makes the dependency on the source and target distributions spurious; hence, it can be removed as a policy dependency, i.e.,

$$\boldsymbol{x}_g = \phi(f(\phi^{-1}(\boldsymbol{x})|\mathcal{X})). \tag{5}$$

where $f$ is the policy trained only on the demonstration labels, no task parameters. All the task information is encoded in the transportation map $\phi$. However, it is evident from (5) that the main limitation of the online approach is the computation at run time of the direct and the inverse transportation map, hence requiring the fitting and existence of the invertible map. Nevertheless, this formulation has the great advantage of reusing the original policy $f$ that was trained on a source environment, for example, a certain surface shape, position, or object and goal. Moreover, this strategy is ideal when the function approximator $f$ is slow or expensive to train, for example with RL,

while the transportation map $\phi$ and its inverse have a (very) fast inference speed. The idea of projecting between a source and a target space using a smooth and invertible map has been extensively exploited in the stable motion learning from demonstration [14, 15, 16] or from reinforcement [17].

An alternative solution is to fit a new policy $g$ using as labels the transported data from the source scenario to the target space, i.e.,

$$\boldsymbol{x}_g = g(\boldsymbol{x}|\phi(\mathcal{X})). \tag{6}$$

This offline approach has the advantage of relying only on direct transportation, with no requirements on the inference speed of the map $\phi$. When dealing with robot policies that include forces, stiffness, velocity, orientation, and twist, the partial derivative of the transportation map can be exploited to transport all demonstration state-actions in the current task configuration; see [12] for more insights.

However, when dealing with task parameters that rapidly change, we need a *fast* fitting of the transportation map $\phi$ and refitting of the policy $g$. Let us consider the example of Figure 1 where the user is recording a single demonstration to go from the starting **green** frame to the **yellow** frame. Given a novel configuration of the starting and the goal frame, we can transport the original demonstration by knowing the new keypoint locations. In the figure, we also highlight the epistemic uncertainty prediction since the transportation map is a Gaussian Process [12, 18].

### 2.3 Dealing with a Distribution of Source Keypoints

When dealing with a set of demonstrations, provided for a different set of keypoint configurations, we have two options when generalizing to a novel target configuration:

1. Transport all the demonstrations from each source distribution to the target distribution.
2. Select the nearest or most correlated source distribution(s) to the current target and only transport the relevant demonstration(s).

The first strategy is ideal when dealing with a handful of demonstrations, but it will not scale when dealing with tens of thousands. Figure 2 uses the same data as 1, but given each of the eight frame configurations, it projects the demonstrations from the other seven configurations and fits a motion behavior, $g(\boldsymbol{x})$, depicted as a vector field. Transporting more demonstrations has the main advantage of fitting a richer, more informative motion behavior in the current target situation.

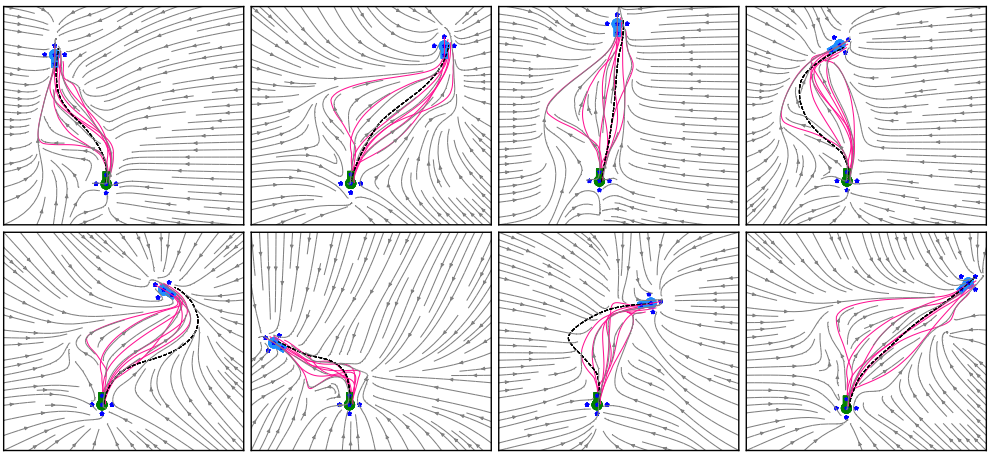

Figure 2: Transporting demonstrations from multiple sources to a single target scenario. We fit *m* transportation maps, one for each source keypoint distribution and demonstration, to project the data into the new frame configuration. A motion policy is trained on the projected data, and the depicted vector field shows its prediction across the workspace. The dashed line represents the human demonstration for that scenario used for validation, which is not used during training.

## 3 Robot Experiments

This section illustrates the performance of the policy transportation in three scenarios: (i) learning non-prehensile manipulation from a large dataset, (ii) using automatic keypoint selection with a Vision Transformer [19], and (iii) generalizing pick-and-place trajectories in a retail environment.

### 3.1 Transportation of Push-T Policy

A Push-T task, illustrated in Figure 3, is a robotic non-prehensile manipulation benchmark where an object must be pushed by a robot agent [20]. It requires the learning of a multi-modal behavioral model. In our experiment, we extracted twelve keypoints for the object, twelve for the goal, and one for the agent position for a total of 25 keypoints. We recorded thirty-five demonstrations using keyboard teleoperation for a total of 13,685 action chunks. The action chunks are trajectory segments of twenty steps. Since the policy transportation strategy does not require any offline fitting, we are deploying the policy and aggregating more data interactively with human interventions when the robot gets stuck or fails to perform the pushing task [21]. The total aggregated dataset has 53,731 action chunks.

Table 1: Evaluation of Push-T task in and out of training distribution

| **Method** | **In distribution** | **Out of distribution** | |
|---|---|---|---|
| | | Robot | Object |
| Nearest Neighborhood | 85.0% | ✗ | ✗ |
| Policy Transportation | 99.5% | 85.5% | 35.5% |
| Conditional Flow Matching | 97.0% | 88.0% | 40.0% |

For the performance evaluation, we propose and compare three strategies that use keypoints as encoding:

1. **Nearest Neighbor**: The action chunk of the nearest neighborhood distribution in the dataset is executed. The nearest neighbor is selected after the execution of the whole chunk, i.e., 20 steps.
2. **Policy Transportation**: The source distribution is obtained as the nearest neighborhood keypoint distribution with respect to the current (target) one. A transportation map is regressed using the source and the target distribution. Then, the map transforms the nearest action chunk. This is performed after the rollout of the previous chunk. We use Iterative Locally Weighted Regression [14] as the transportation method.
3. **Keypoint Conditional Flow Matching (K-CFM)**: We train a denoising process to generate action chunks given the current keypoint observation. This is a special case of Point Cloud Flow Matching, as proposed by Chisari *et al.* [22]. In our case, we do not condition the denoising process on a PointNet++ [23] embedding of the (large) point cloud but directly on a sparse set of keypoints.

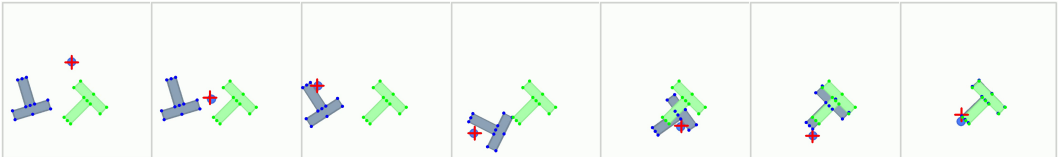

Figure 3: Push-T task using keypoint parameterization. The task is described as keypoints on the perimeter of the T-shaped object to push and on the desired goal configuration. The behavior is learned from a set of offline and interactive demonstrations.

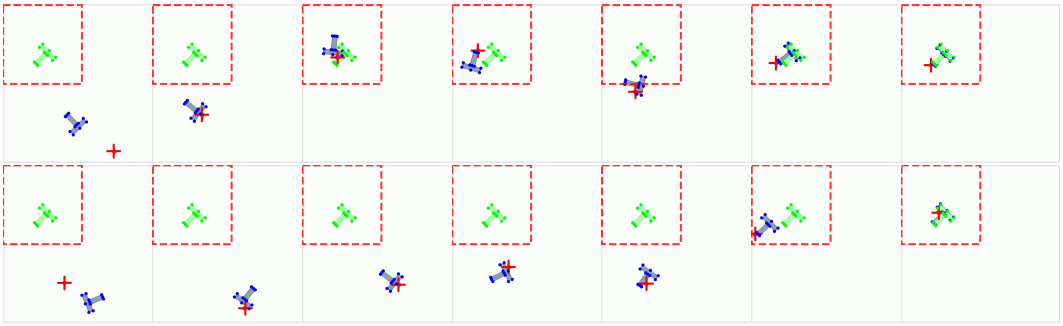

Figure 4: Out-of-distribution successful generalization of a Policy transportation (**top**) and Conditional Flow Matching Policy (**bottom**). The red square is the area where the demonstrations were collected.

For the three strategies, we evaluate the policy on 200 different initial configurations where only the position and orientation of the object and the goal are randomized. We evaluated the policy both in an environment that is the same dimension as the training environment, but also on another that is 4 times bigger, where the original training environment coincides with the top-left corner. In the large workspace, there is a high chance of initializing the policy outside the distribution of the recorded data. Figure 4 highlights two examples of successful out-of-distribution (OOD) generalization when using policy transportation or conditional flow matching. Moreover, Table 1 summarizes the performance of the different methods. In the training environment, even the simplest Nearest Neighbor performs with high accuracy. The policy transportation is the best performing. Among 200 trials, it reached a timeout, set to 2 minutes, only once.

When evaluating OOD generalization, we investigated two scenarios: (i) the robot spawns outside the training box while the object remains inside; and (ii) the object spawns completely or partially outside the training box. In the first case, we assess the robot's ability to return to the familiar training scenario, where it should know how to perform the pushing task. The second case is more challenging, as the robot must manipulate the object under OOD conditions, transferring its learned skills to push the object back into the training area and then towards the goal. This experiment aims to systematically evaluate the OOD performance of a task-parameterized behavioral policy.

The table highlights that the main drop in performance for both policy transportation and the conditional flow matching happens when the objects are initialized outside the original workspace. Figure 4 showcases some noteworthy, albeit cherry-picked, examples of generalization for both the policy distribution and flow matching. A key failure point in policy transportation appears to be its reliance on selecting the closest point in the training set, followed by a highly nonlinear deformation of the state space. Conversely, when the denoising process is conditioned on unseen inputs, the resulting action chunk can remain noisy, leading to suboptimal manipulation behaviors.

### 3.2 Pick-and-Place with Vision Transformer-Based Keypoint Extraction

Rather than hand-picking the keypoint placement, given a stereoscopic input, we can leverage new foundation models to automatically extract and find the corresponding 3D keypoints [19]. However, rather than roto-translating the policy [3, 10], we apply the nonlinear transportation map. Figure 5 shows the source, recorded before giving the demonstration, and the target image on a new OOD configuration of the objects. A stereoscopic depth map is used to estimate the 3D location of each feature, so as to generate the necessary keypoints to fit the transportation map. We ask the DINO model to find a maximum of 7 key points, and a threshold of 0.2 was applied to the saliency maps, retaining only features with higher attention values; this avoids finding features on the table. We moved the objects around the workspace, always with the (toy) cheese on the left and the (toy) lettuce on the right, and used a transportation policy with a Gaussian Process to perform the generalization of the original position and orientation labels [12]. Despite successful results in generalizing in

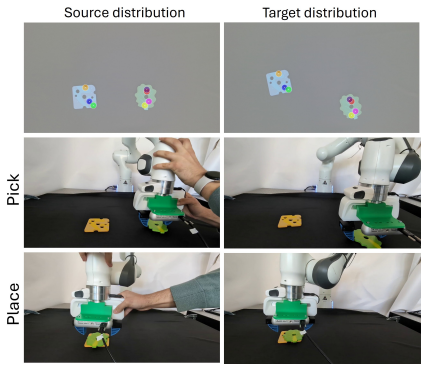


Figure 5: Food Stacking Task. The single demonstration is shown on the left, and the corresponding motion generalization on the right. Source and target keypoints are automatically extracted using DINO features correspondences [19], with their 3D positions estimated from a stereoscopic depth map. A kinesthetic demonstration is performed by the user, and a transportation policy is used to generalize the motion across different object configurations.

10 different configurations, we identify poor robot orientation generalization when swapping the location of the objects, i.e., lettuce on the left and cheese on the right side of the table.

### 3.3 Policy Transportation of an Object Packing Policy

In this section, we investigate how a learned object-packing policy can be transported to novel task configurations using keypoint-based representations. Our setting involves a single UR5e robot manipulator equipped with an OnRobot VGC10 suction gripper that has been demonstrated through a planner to pack an object into a paper bag placed on a table. At test time, we aim to show the zero-shot generalization of this behavior with different object positions, object types, and bag placements.

Figure 6 illustrates the experimental setup that consists of a paper bag with an attached AprilTag, an object to be packed, and a table-mounted RGB-D camera providing a top-down view of the workspace. To detect and segment the object on the table, Grounding DINO [24], a language-conditioned object detector, identifies the bounding box corresponding to the text prompt, and Segment Anything [25] prompted with the bounding box extracts the granular mask. The camera intrinsics and the stereoscopic depth map are then used to extract the object-centric point cloud.

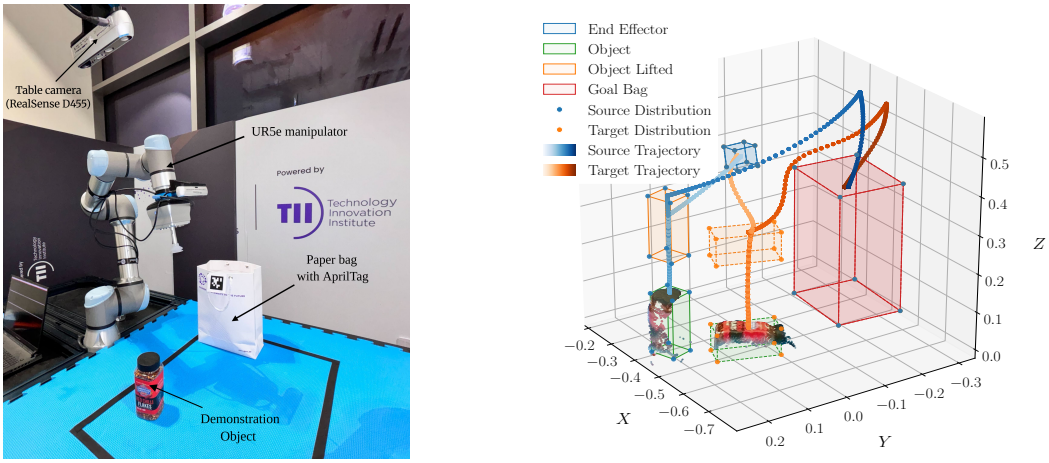


Figure 6: The **left** picture depicts the setup used for performing the experiments. The **right** picture shows the source keypoints and trajectory as well as the target ones for one of the tests. It can be observed how the source trajectory gets deformed and successfully places the object inside the bag.

We represent each task instance with four groups of eight keypoints extracted from the initial position of the end effector, the object at rest, the object after lifting, and the target location represented by the paper bag. For the end-effector and bag, keypoints match the vertices of a parallelepiped obtained by applying fixed offsets to their respective poses, defining an oriented 3D bounding shape for each component. Regarding the object, the keypoints were obtained by projecting the 3D point

Figure 7: Source (**top**) and target (**bottom**) trajectories under full scene variation.

cloud into the XY plane and fitting a minimum-area rectangle around the object's 2D footprint. The z-component of the top keypoints was set to the average height of the highest voxel, providing a robust estimate of the object's top surface. A Gaussian Process transportation map was fitted between the source and the target keypoints to generalize the only source trajectory to any novel task configuration. A video of the robot performance can be found at: https://youtu.be/MIkD2adbI8w. We evaluated the performance under two different scenarios over 15 trials each varying the position and orientation in a table top rectangular area of dimension $60\,\mathrm{cm} \times 40\,\mathrm{cm}$:

1. **Object relocation with fixed goal:** We evaluated how well the transported policy adapts to different object positions while keeping the object type and bag location fixed. The same object was placed at various positions across the table, including locations far from the demonstration. In all 15 trials, the object was successfully placed inside the bag. However, when the object was far from the demonstration region, the arm occasionally encountered kinematic singularities, and the resulting trajectories were noticeably less efficient.
2. **Full scene variation:** In this more challenging scenario, both the object and the goal differ from the source demonstration. We varied object **type** and **position**, and **bag location** to test generalization under compound distribution shifts. The method succeeded in 11 out of 15 trials, with failures caused by the arm reaching singularities when the object or bag was placed far from the base. Good generalization was still observed, as shown in Figure 7, though trajectories became more non-linear when object and bag positions were swapped.

## 4 Limitations and Opportunities

Despite the successful application of the proposed policy transportation on different challenging tasks, we can foresee some limitations and future challenges to improve the applicability and have a broader impact. For example, we assume knowing the matching between the points of the source and the target distribution. However, in many complex scenarios, this can limit the effectiveness of the method, and it becomes necessary to adopt alternative pre-processing algorithms such as Optimal Transport [26], Iterative Closest Point (ICP), or Coherent Point Drift [8] for the matching. In addition, current Vision Transformers like DINO are a very practical tool for keypoint extraction and matching, but they still make mistakes when dealing with symmetric objects.

Moreover, the keypoint placement prior introduces a strong inductive bias. Since we are using a transportation map, this implies trajectory points closer to the keypoints are moved more during the generalization. This also implies there is a decaying effect for points further away. However, this may not always be the desired generalization. We foresee an exciting research direction in automatic keypoint extraction based on a provided set of human demonstrations and corrections [6, 7].

In general, whenever we are performing an OOD generalization, such as when only one demonstration is provided, we lack evidence on the effectiveness or optimality of the generalization. If we are out of distribution, the model cannot infer what the human would have intended in that novel situation. The robotics community calls for better benchmarks for testing truly OOD behaviors, along with policies with better uncertainty quantification and calibration when facing unknown situations.

## References


[1] L. Manuelli, W. Gao, P. Florence, and R. Tedrake. KPAM: KeyPoint Affordances for Category-Level Robotic Manipulation. In *Proceedings of the 19th International Symposium on Robotics Research (ISRR)*, pages 132–157, 2022. DOI:10.1007/978-3-030-95459-8_9.

[2] A. Simeonov, Y. Du, A. Tagliasacchi, J. B. Tenenbaum, A. Rodriguez, P. Agrawal, and V. Sitzmann. Neural Descriptor Fields: SE(3)-Equivariant Object Representations for Manipulation. In *Proceedings of the 2022 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6394–6400, 2022. DOI:10.1109/ICRA46639.2022.9812146.

[3] N. Di Palo and E. Johns. DINOBot: Robot Manipulation via Retrieval and Alignment with Vision Foundation Models. In *Proceedings of the 2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2798–2805, 2024. DOI:10.48550/arXiv.2402.13181.

[4] N. Di Palo and E. Johns. Keypoint Action Tokens Enable In-Context Imitation Learning in Robotics. In *Proceedings of Robotics: Science and Systems (RSS)*, 2024. DOI:10.48550/arXiv.2403.19578.

[5] W. Huang, C. Wang, Y. Li, R. Zhang, and L. Fei-Fei. ReKep: Spatio-Temporal Reasoning of Relational Keypoint Constraints for Robotic Manipulation. In *Proceedings of the 8th Annual Conference on Robot Learning (CoRL)*, 2024. DOI:10.48550/arXiv.2409.01652.

[6] Z. Qin, K. Fang, Y. Zhu, L. Fei-Fei, and S. Savarese. KETO: Learning Keypoint Representations for Tool Manipulation. In *Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7278–7285, 2020. DOI:10.48550/arXiv.1910.11977.

[7] J. Gao, Z. Tao, N. Jaquier, and T. Asfour. K-VIL: Keypoints-based Visual Imitation Learning. *IEEE Transactions on Robotics*, 39(5):3888–3908, 2023. DOI:10.1109/TRO.2023.3286074.

[8] A. Myronenko and X. Song. Point-Set Registration: Coherent Point Drift. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(12):2262–2275, 2010. DOI:10.1109/tpami.2010.46.

[9] G. Balakrishnan, A. Zhao, M. R. Sabuncu, J. Guttag, and A. V. Dalca. VoxelMorph: A Learning Framework for Deformable Medical Image Registration. *IEEE Transactions on Medical Imaging*, 38(8):1788–1800, 2019. DOI:10.1109/tmi.2019.2897538.

[10] J. Yang, Z. Cao, C. Deng, R. Antonova, S. Song, and J. Bohg. EquiBot: SIM(3)-Equivariant Diffusion Policy for Generalizable and Data Efficient Learning. In *Proceedings of the 8th Annual Conference on Robot Learning (CoRL)*, 2024. DOI:10.48550/arXiv.2407.01479.

[11] S. Patel, X. Yin, W. Huang, S. Garg, H. Nayyeri, L. Fei-Fei, S. Lazebnik, and Y. Li. A Real-to-Sim-to-Real Approach to Robotic Manipulation with VLM-Generated Iterative Keypoint Rewards. In *Proceedings of the 2025 IEEE International Conference on Robotics and Automation (ICRA)*, 2025. DOI:10.48550/arXiv.2502.08643.

[12] G. Franzese, R. Prakash, C. Della Santina, and J. Kober. Generalizable Motion Policies through Keypoint Parameterization and Transportation Maps. *IEEE Transactions on Robotics*, 41:4557–4573, 2025. DOI:10.1109/TRO.2025.3582821.

[13] I. Kobyzev, S. J. Prince, and M. A. Brubaker. Normalizing Flows: An Introduction and Review of Current Methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(11):3964–3979, 2020. DOI:10.1109/tpami.2020.2992934.

[14] N. Perrin and P. Schlehuber-Caissier. Fast Diffeomorphic Matching to Learn Globally Asymptotically Stable Nonlinear Dynamical Systems. *Systems & Control Letters*, 96:51–59, 2016. DOI:10.1016/j.sysconle.2016.06.018.

[15] R. Pérez-Dattari and J. Kober. Stable Motion Primitives via Imitation and Contrastive Learning. *IEEE Transactions on Robotics*, 39(5):3909–3928, 2023. DOI:10.1109/TRO.2023.3289597.

[16] M. A. Rana, A. Li, D. Fox, B. Boots, F. Ramos, and N. Ratliff. Euclideanizing Flows: Diffeomorphic Reduction for Learning Stable Dynamical Systems. In *Proceedings of the 2nd Conference on Learning for Dynamics and Control*, pages 630–639, 2020. DOI:10.48550/arXiv.2005.13143.

[17] S. A. Khader, H. Yin, P. Falco, and D. Kragic. Learning Stable Normalizing-Flow Control for Robotic Manipulation. In *Proceedings of the 2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1644–1650, 2021. DOI:10.48550/arXiv.2011.00072.

[18] C. K. Williams and C. E. Rasmussen. *Gaussian Processes for Machine Learning*, volume 2. MIT Press, 2006. DOI:10.7551/mitpress/3206.001.0001.

[19] S. Amir, Y. Gandelsman, S. Bagon, and T. Dekel. Deep ViT Features as Dense Visual Descriptors. In *Proceedings of the 2022 European Conference on Computer Vision (ECCV) Workshop on What is Motion For?*, 2022. DOI:10.48550/arXiv.2112.05814.

[20] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song. Diffusion Policy: Visuomotor Policy Learning via Action Diffusion. *The International Journal of Robotics Research*, 2024. DOI:10.1177/02783649241273668.

[21] C. Celemin, R. Pérez-Dattari, E. Chisari, G. Franzese, L. de Souza Rosa, R. Prakash, Z. Ajanović, M. Ferraz, A. Valada, and J. Kober. Interactive Imitation Learning in Robotics: A Survey. *Foundations and Trends in Robotics*, 10(1-2):1–197, 2022. DOI:10.1561/2300000072.

[22] E. Chisari, N. Heppert, M. Argus, T. Welschehold, T. Brox, and A. Valada. Learning Robotic Manipulation Policies from Point Clouds with Conditional Flow Matching. In *Proceedings of the 8th Annual Conference on Robot Learning (CoRL)*, 2024. DOI:10.48550/arXiv.2409.07343.

[23] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In *Proceedings of the 31st International Conference on Neural Information Processing Systems (NeurIPS)*, pages 5105–5114, 2017. DOI:10.48550/arXiv.1706.02413.

[24] S. Liu, Z. Zeng, T. Ren, F. Li, H. Zhang, J. Yang, Q. Jiang, C. Li, J. Yang, H. Su, J. Zhu, and L. Zhang. Grounding DINO: Marrying DINO with Grounded Pre-training for Open-Set Object Detection. In *Proceedings of the 18th European Conference on Computer Vision (ECCV)*, pages 38–55, 2024. DOI:10.48550/arXiv.2303.05499.

[25] N. Ravi, V. Gabeur, Y.-T. Hu, R. Hu, C. Ryali, T. Ma, H. Khedr, R. Rädle, C. Rolland, L. Gustafson, E. Mintun, J. Pan, K. V. Alwala, N. Carion, C.-Y. Wu, R. Girshick, P. Dollar, and C. Feichtenhofer. SAM 2: Segment Anything in Images and Videos. In *Proceedings of the 13th International Conference on Learning Representations (ICLR)*, 2025. DOI:10.48550/arXiv.2408.00714.

[26] Y. Luo, Z. Jiang, S. Cohen, E. Grefenstette, and M. P. Deisenroth. Optimal Transport for Offline Imitation Learning. In *Proceedings of the 11th International Conference on Learning Representations (ICLR)*, 2023. DOI:10.48550/arXiv.2303.13971.