Investigating Dense Transformation vs Flow for Tool Manipulation

[redacted]

[redacted]

Abstract—**Dynamic manipulation of tools is present for many robotic tasks, such as pouring water from one cup to another. We examine the importance of action representations in performing such tasks. We build upon prior work ToolFlowNet, but instead of predicting per-point flows, we propose to predict dense transformations. The main contribution of our work is a point cloud-based architecture for imitating dense 6D actions for tool use that can perform just as well as the flow-based baseline. We show for transformation-based methods that using dense predictions and random sampling in combination achieves the best success rate and quality over single transformation and fixed point methods.**

Index Terms—**manipulation, tool use, point clouds, action representations**

I. Introduction

Manipulation for tool use requires reasoning about the geometry of the tool and how it interacts with its environment. In a water pouring task, a robot must take into account how different parts of the cup interact with the contained water to successfully pour the water while minimizing spillage. Some common modalities for observations for policy learning include RGB images [1]–[3] and robot ground truth joint angle [4], [5], however policy learning from point clouds has been less explored. Methods specifically using point clouds tend to focus on segmentation or classification [6], [7] instead of policies, perhaps due to the difficulty of reasoning about 3D point clouds. However, point clouds are a promising observation modality: point clouds offer depth information unlike 2D RGB images, and it is easier to transfer from simulation to the real world. Moreover, they are easily obtainable for most objects in the scene with depth cameras unlike ground truth poses. In this paper, we aim to explore different action representations based on the point cloud observations and examine how to utilize the point cloud structure to better learn the policy.

Prior work in ToolFlowNet [8] demonstrated the efficacy of dense flow action predictions on deformable manipulation tasks. However, these flow predictions only describe the positional motion of the tool, and not its rotation in space. To build upon this prior work, our project uses a similar point cloud backbone, but explores 6D transformation-based action representations, which include both positional and rotational information, and show that a transformation-based action space can perform just as well as a flow-based action space. We ablate against a few variations of transformation-based action spaces, with dense vs single predictions and random point vs fixed point point selection.

[redacted]

Our main contributions are:

- We propose a novel point cloud architecture for behavior cloning that predicts tool transformations along a trajectory as 6D vectors.
- We compare our method against ablations and baselines to show that dense predictions with random sampling performs the best on a water pouring task, possibly due to the dense supervision this provides during training.

II. Related Work

A. 6D transformations for policy learning:

Prior work has used 6D transformations as action spaces in robot manipulation, whether it be framed as a reinforcement learning problem [5] or a grasp success classification problem [7]. However, there has been little comparison between transformation-based action spaces and flow-based action spaces. In addition, prior work mostly explores the manipulation of rigid objects. Also, such methods often predict a single transformation and lack dense supervision. The single transformation from prior work is usually the robot gripper transformation, thus highly dependent on where the robot grasps the object or the tool. Therefore, in a more general way, the policy should learn an object-centric transformation, which can extend to more diverse settings where the relative position between the gripper and the object can change. In this work, we explore this object-centric action representation and directly compare two design choices for action spaces for deformable object manipulation and show that both can be optimized for a model to predict 6 DOF robot actions well.

B. Deformable object manipulation:

We focus on the application of deformable object manipulation. One common difficulty with deformable object manipulation is state representation for non-rigid objects. Some methods simplify the state [9] or specifically learn the environment dynamics [10]–[12] to learn policies that can handle the complexities of deformable objects like sand [13], rope [9], or cloth [14]. In contrast, in this work and in ToolFlowNet [8], we propose a general method that does not require the modeling of the dynamics of deformables.

III. Method

We are interested in the task of policy learning using segmented point cloud observations. At a given timestep t, a segmented point cloud $\mathbf{P}_t$ has N points, each with a feature dimension of d. The feature includes its 3D coordinate position $(x, y, z) \in \mathbb{R}^3$ and a one-hot vector that indicates the object class to which it belongs to.

Our focus is on studying Behavioral Cloning (BC) using segmented point clouds. BC assumes access to a dataset $D = (o_t, a_t^*)_{t=1}^M$, where o_t, a_t^* is the observation and optimal action from a demonstrator at timestep t respectively. The observation o_t we used in this project is point cloud observation $\mathbf{P}_t$. BC employs supervised learning to train a policy π with parameters θ, aiming to make the predicted action $a_t = \pi(o_t)$ closely match the ground truth label a_t^*. While BC may exhibit some error accumulation during testing, it has demonstrated remarkable effectiveness in comparison to more intricate learning-based algorithms in certain robotic manipulation scenarios. This success has prompted further investigation into its applicability with point cloud observations. For simplicity, we will omit the timtetstep t subscript in the following part of the paper.

In the following section III-A and section III-B, we are going to explain two action representations for the Behavior Cloning policy, namely Flow-based action and transformation-based action.

A. Flow as Action Representation

The flow as action representation is adopted from this paper **ToolFlowNet** [8] where we use a flow $f^{(i)} \in \mathbb{R}^3$ associated with point $p^{(i)}$ as a 3D vector. For each point on the tool we want to manipulate, the flow vector represents how the point will move in the 3D space as a result of applying the action. In details, we adopt a commonly used architecture for point cloud (segmentation PointNet++) to extract per-point feature from the tool. This backbone would take the 3d world position and a one-hot encoding of the class of each point in the point cloud, where $\mathbf{P}_t$ has the shape of (N, d_1) as input and output per point feature $\mathbb{F}$ in shape (N, d_2). The resulted 3D flow vectors $\mathbb{F} = \{f^{(i)}\}_{i=1}^N$ is fed through a differentiable, parameter-less Singular Value Decomposition (SVD) layer to get the rotation R_t of the tool /todocite here. The translation t_t is computed as the difference between $\mathbf{P_t} + \mathbf{F_t}$ and $R_t\mathbf{P_t}$. The final transformation of the tool is (R_t, t_t).

B. Transformation as Action Representation

Similar to flow-based action, transformation-based action also uses a segmentation PointNet++ based backbone to extract the per-point feature first from point cloud observation $\mathbf{P}_t$ and then it is mapped through a network to output the per-point transformation $T_t = (R_t, t_t) \in \mathbb{R}^{N\times 6}$. There are several variants of the transformation-based action representations which differ in how the final action a_t is generated from this dense transformation T_t.

*1) **SFP-Single: Single Fixed Point Transformation as Action Representation**:* The first variant creates a fake center point and replace the first point p_0 of the point cloud observation with the fake center point. In our pouring water task, the fake point is the bottom center of the pouring box. The transformation of the tool is estimated as $\hat{T}_t^{(0)}$, the first element of $\hat{T}_t$. The ground truth action is $T_t^{*(0)}$. The behavior cloning loss is shown as below:

$$L(\hat{T}_t, T_t^*) = ||\hat{T}_t^{(0)} - T_t^{*(0)}|| \tag{1}$$

*2) **RP-Dense: Random Point Transformation from Dense Transformation as Action Representation**:* Unlike the first variant which only uses a single point transformation to provide supervising signal for the training, we use dense transformation to provide guidance for the policy. The training loss is:

$$L(\hat{T}_t, T_t^*) = \sum_{i=0}^{N} ||\hat{T}_t^{(i)} - T_t^{*(i)}|| \tag{2}$$

Through training, each point should learn a transformation of the tool with regard to the coordinate frame that is centered around the point. During test time, a random point index i is selected among all possible N points to use as the coordinate frame center p_i and the corresponding transformation is retrieved as $\hat{T}_t^{(i)}$.

*3) **SFP-Dense: Single Fixed Point Transformation from Dense Transformation as Action Representation**:* The training loss is the same as Equation 2 but we propose a different selection of the point during evaluation. Instead of selecting a random point, this method always choose the first point $p^{(0)}$ as the coordinate frame center and select the corresponding transformation $\hat{T}_t^{(0)}$ from the dense transformation predicted by the network.

*4) **LWP-Dense: Largest Weight Point Transformation from Dense Transformation as Action Representation**:* In order to automatically select the most appropriate point as the coordinate frame center and apply the corresponding transformation to the tool, we modify the output of our network to not only generate a per-point transformation $\hat{T}_t^{(i)}$ but also a per point weight $\omega_t^{(i)}$. The training loss is a weighted sum of the per-point loss.

$$L(\hat{T}_t, T_t^*) = \sum_{i=0}^{N} \omega_t^{(i)} ||\hat{T}_t^{(i)} - T_t^{*(i)}|| \tag{3}$$

During evaluation, we would select the point that has the largest weight as the coordinate frame center to apply the tool transformation. The estimated action $\hat{a_t}$ is as follow:

$$\hat{a}_t = \hat{T}_t^{(arg\max_i \omega_t^{(i)})} \tag{4}$$

The detailed structure of the framework for section III-B2 and section III-B3 is shown in the figure 1

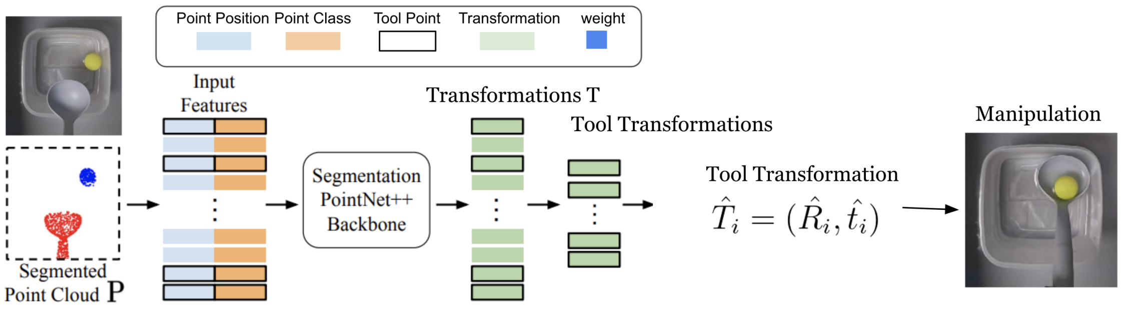

Fig. 1. Method Overview for **Random Point Transformation from Dense Transformation** as Action Representation and **Single Fixed Point Transformation from Dense Transformation** as Action Representation.

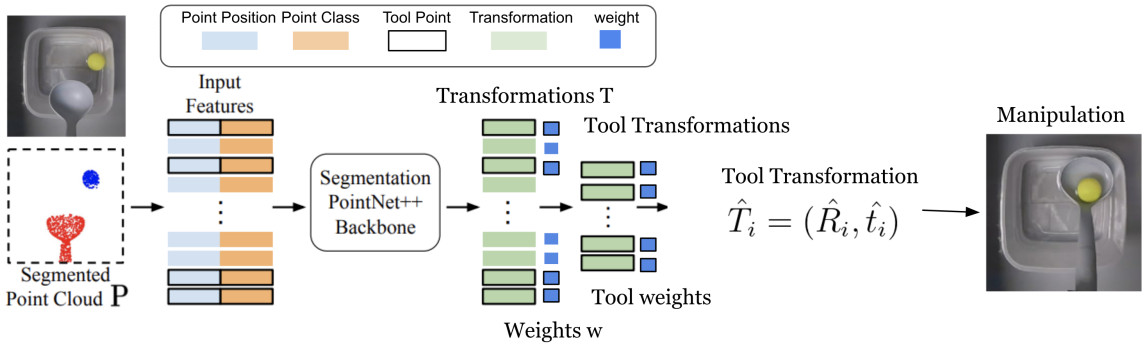

Fig. 2. Method Overview for **Largest Weight Point Transformation from Dense Transformation** as Action Representation.

IV. Experiments and Results

We evaluate our method in the simulated SoftGym [15] PourWater task shown in Figure 3, where an agent is tasked with pouring water from one box into a target box. The task is considered as a success if at least 75% of the water particles end in the target box. We used 100 demonstrations for training the behavior cloning policy for all the methods and $N = 2000$ points for point cloud observation. After the network generates the transformation, the simulation applies the corresponding transformation to the pouring box. All the methods are trained for 500 epochs and we use the checkpoint at the last epoch for evaluation. During evaluation, there are 25 fixed held-out starting configurations. The success rate averaged over 25 configurations for all the methods are shown in Figure 4.

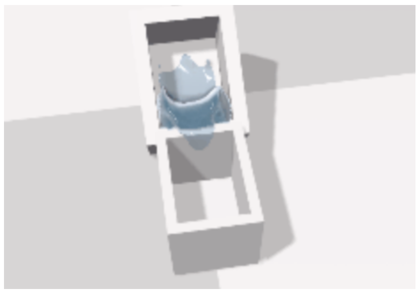

Fig. 3. The PourWater task in SoftGym [15].

The methods we are testing are listed in III-A and III-B. We used their abbreviation for simplicity. We also listed some related metrics, including the training loss, testing loss, percentage (percentage of particles poured into the target box) and success rate, in the table I.

From the result, we can see that RP-Dense achieved best performance both in success rate and pouring percentage among all the methods. ToolFlowNet also demonstrated comparable performance as RP-Dense. Therefore, both dense transformation and flow have great potential in learning such tool transformation policy. We also noticed that SFP-Dense significantly outperforms SFP-Single because SFP-Single only receives supervised signal from one point transformation so it may lose track of the whole point cloud structure and it fails to generalize to different variations of the box sizes we introduced in the demonstrations. In contrast, SFP-Single receives all the points' supervised signals and those supervised transformations are related to the distances between points. Therefore, these transformations can help the network quickly capture the structure of the point cloud, accelerating the policy learning for tool transformation. Another difference we can find from the table is that RP-Dense outpeforms SFP-Dense. This may suggest that through training with dense transformation, not necessarily the fixed center point learns a best transformation all the time. Encouraging the randomness of the choice of the transformation coordinate frame center may bring additional benefit for learning an accurate transformation. This is also

Method	Metrics			
	Training Loss	Testing Loss	Success Rate	Percentage
ToolFlowNet	**0.0018**	**0.0046**	0.6	0.72
SFP-Single	0.0081	0.0146	0.2	0.42
RP-Dense	0.0073	0.0136	**0.68**	**0.77**
SFP-Dense	0.0074	0.0140	0.4	0.53
LWP-Dense	0.0045	0.0356	0	0.08

TABLE I
THE TRAINING LOSS, TESTING LOSS, SUCCESS RATE, AND PERCENTAGE OF WATER IN THE TARGET CUP FOR DIFFERENT METHODS AVERAGED OVER 25 TRIALS WITH DIFFERENT ENVIRONMENT INITIAL CONFIGURATIONS.

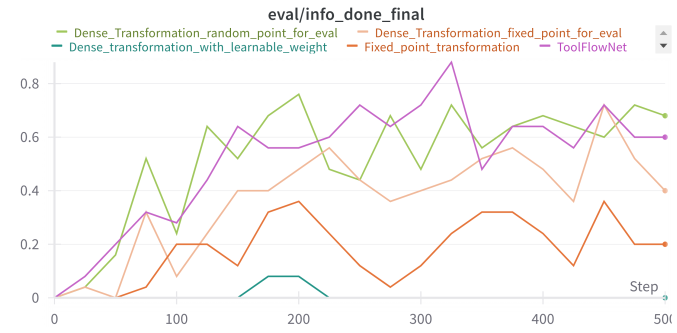

Fig. 4. The success rate for Flow-based method and Transformation-based methods.

the reason why we introduce our last variant LWP-Dense, which adds a per point weight into the architecture so that we could learn a weight for each point automatically. During the test time, if the weight is reliable, we could automatically choose the coordiante frame center based on the largest weight. However, since learning the transformation and learning the weight is a deeply coupled problem. It is very difficult for the network to learn it concurrently. Due to the limited time, we haven't fully tackled the problem of learning weight and transformation at the same time but we believe this is an interesting direction to explore in the future.

V. Conclusion

We investigated a variety of action representations for robotic manipulation with point clouds. Our method RP-Dense predicts dense 6D transformations with dense supervision and shows that transformation-based action spaces are just as good as flow-based action spaces if trained with the right supervision. We compared methods in a water pouring task for both success rate and quality. This work shows the promise of dense point cloud networks in robotic manipulation tasks for future work.

References

[1] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 1334–1373, 2016.

[2] L. Pinto and A. Gupta, "Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours," in *2016 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2016, pp. 3406–3413.

[3] I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas *et al.*, "Solving rubik's cube with a robot hand," *arXiv preprint arXiv:1910.07113*, 2019.

[4] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel *et al.*, "Soft actor-critic algorithms and applications," *arXiv preprint arXiv:1812.05905*, 2018.

[5] L. Wang, Y. Xiang, W. Yang, A. Mousavian, and D. Fox, "Goal-auxiliary actor-critic for 6d robotic grasping with point clouds," in *Conference on Robot Learning*. PMLR, 2022, pp. 70–80.

[6] K. Takeuchi, I. Yanokura, Y. Kakiuchi, K. Okada, and M. Inaba, "Automatic hanging point learning from random shape generation and physical function validation," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 4237–4243.

[7] X. Yan, J. Hsu, M. Khansari, Y. Bai, A. Pathak, A. Gupta, J. Davidson, and H. Lee, "Learning 6-dof grasping interaction via deep geometry-aware 3d representations," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 3766–3773.

[8] D. Seita, Y. Wang, S. J. Shetty, E. Y. Li, Z. Erickson, and D. Held, "Toolflownet: Robotic manipulation with tools via predicting tool flow from point clouds," in *Conference on Robot Learning*. PMLR, 2023, pp. 1038–1049.

[9] S. Wang, R. Papallas, M. Leouctti, and M. Dogar, "Goal-conditioned action space reduction for deformable object manipulation," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 3623–3630.

[10] R. Wu, C. Ning, and H. Dong, "Learning foresightful dense visual affordance for deformable object manipulation," *arXiv preprint arXiv:2303.11057*, 2023.

[11] C. Schenck and D. Fox, "Visual closed-loop control for pouring liquids," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 2629–2636.

[12] C. Matl, Y. Narang, R. Bajcsy, F. Ramos, and D. Fox, "Inferring the material properties of granular media for robotic tasks," in *2020 ieee international conference on robotics and automation (icra)*. IEEE, 2020, pp. 2770–2777.

[13] C. Schenck, J. Tompson, S. Levine, and D. Fox, "Learning robotic manipulation of granular media," in *Conference on Robot Learning*. PMLR, 2017, pp. 239–248.

[14] K. Mo, C. Xia, X. Wang, Y. Deng, X. Gao, and B. Liang, "Foldsformer: Learning sequential multi-step cloth manipulation with space-time attention," *IEEE Robotics and Automation Letters*, vol. 8, no. 2, pp. 760–767, 2022.

[15] X. Lin, Y. Wang, J. Olkin, and D. Held, "Softgym: Benchmarking deep reinforcement learning for deformable object manipulation," in *Conference on Robot Learning*. PMLR, 2021, pp. 432–448.