Learning Transformable and Plannable se(3) Features for Scene Imitation of a Mobile Service Robot

J.hyeon Park[1], Jigang Kim[1], Youngseok Jang[1], Inkyu Jang[2], and H.Jin Kim[1]

***Abstract*—Deep neural networks facilitate visuosensory inputs for robotic systems. However, the features encoded in a network without specific constraints have little physical meaning. In this research, we add constraints on the network so that the trained features are forced to represent the actual twist coordinates of interactive objects in a scene. The trained coordinates describe 6d-pose of the objects, and $SE(3)$ transformation is applied to change the coordinate system. This algorithm is developed for a mobile service robot that imitates an object-oriented task by watching human demonstrations. As the robot has mobility, the video demonstrations are collected from different viewpoints. Our feature trajectories of twist coordinates are synthesized in the global coordinate after $SE(3)$ transformation is applied according to robot localization. Then, the trajectories are trained as probabilistic model and imitated by the robot with geometric dynamics of $se(3)$. Our main contribution is to develop a trainable robot with visually demonstrated human performances. Additionally, our algorithmic contribution is to design a scene interpretation network where $se(3)$ constraints are incorporated to estimate 6d-pose of objects.**

***Index Terms*—Deep Learning in Robotics and Automation, Visual Learning, Learning from Demonstration, Mobile Manipulation**

I. Introduction

DEEP neural networks enable robotic systems to capture task-related information from a high-dimensional sensory input such as image pixels. The networks find abstracted features of low dimension so that their dynamics or prediction is available for control [1], [2], [3], policy learning [4], [5], [6], or imitation learning [7], [8], [9]. However, the features trained without specific constraints cannot represent any physical meaning explicitly, because all the information is entangled inside the features. Due to such limitations, the control or planning is performed in the latent space which cannot be physically demonstrated. On the other hand, constrained feature learning proposed in [10] forces the features to follow user-intended dynamics. In their works, the task such as control or planning does not fit into the feature, but the feature is actively managed to fit the tasks instead. Furthermore [11] proposed $se(3)$ constrained features in 6d-pose space for geometric control. Motivated from these works, we develop $se(3)$ constrained features for geometric imitation of object manipulation.

This research aims to develop imitation learning of video demonstrations which are collected in different viewpoints, by managing the $se(3)$ constrained features to directly represent the 6d-pose of objects in a scene with a self-supervised manner. A mobile service robot equipped with a stereo camera and a manipulator moves around and observes the repeated human performance of manipulating objects. As the demonstrations are collected from the different viewpoints, the features of the scene are required to be matched in the global coordinates. Our $se(3)$ feature trajectories are transformed by $SE(3)$ transformation with the information of robot localization and synthesized in the global coordinates. Then, the coordinated trajectories are provided for trajectory learning proposed in [12] that the different demonstrations at a different speed are trained as a probabilistic model to provide a nominal trajectory. Finally, a configurable trajectory of an end-effector is planned to make the object manipulated to follow object nominal trajectory from the current robot location. Although [11] proposed $se(3)$ constrained features in 6d-pose space, their features are still abstract in that they are not compatible with the actual physical pose of the object. The distinguishable point of our research is that we generate $se(3)$ constrained features to directly represent the 6d-pose of objects in the scene so that they can be ready to be used for other robot modules such as planning or control. Although there is research on visual localization [13], or planning with $se(3)$ features [11], our work combines the visual object localization with task imitation.

The main contribution of our work is that a mobile service robot can learn tasks by watching human demonstrations. Imitation learning is a promising method to teach a robot by a non-expert end user. However, teaching by kinesthetic or teleoperated demonstration [14], [15], [16] requires robot operation knowledge and additional sensors to collect the demonstrations. On the other hand, imitation via watching provides a more convenient way of teaching in that just showing demonstration visually can teach a robot. Our algorithmic

Manuscript received: September, 10, 2019; Revised December, 10, 2019; Accepted January, 9, 2020.

This paper was recommended for publication by Editor Eric Marchand upon evaluation of the Associate Editor and Reviewers' comments. This work was supported by Institute for Information & Communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (No.2019-0-01367, Infant-Mimic Neurocognitive Developmental Machine Learning from Interaction Experience with Real World (BabyMind))

[1]J.hyeon Park, Jigang Kim, Youngseok Jang, and H.Jin Kim are with the Department of Mechanical and Aerospace Engineering, Seoul National University, Seoul, Korea, and the Automation and Systems Research Institute (ASRI), Seoul National University, Seoul, Korea {ka2hyeon, jgkim2020, duscjs59, hjinkim}@snu.ac.kr

[2]Inkyu Jang is an undergraduate student in the Department of Mechanical and Aerospace Engineering, Seoul National University, Seoul, Korea leplusbon@snu.ac.kr

Digital Object Identifier (DOI): see top of this page.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2020.2968032, IEEE Robotics and Automation Letters

contribution is to design a neural network that encodes $se(3)$ constrained features from scenic motion. As mentioned above, the proposed approach allows the features to directly represent 6d-pose of objects so that the features are transformable and plannable for various robotic tasks.

II. Related works

The basic approach for feature extraction in deep learning is to use an autoencoder which extracts representative features by self-training [4], [17], [18]. Some researchers proposed end-to-end training from an image to a task that captures task-oriented features to make learning streamlined [1], [19], [20]. Scene modeling is another approach to interpret the scene for planning so that unsupervised training of features enables to predict the scene dynamics [21], [22]. Although features in the above mentioned works contain comprehensive information about the scene, they do not have any innate constraints on the features to represent actual physical meaning. Therefore, a robot task should be defined in a latent space, which is difficult to interpret. Also, the features can be task-oriented, but they cannot be re-used for the different task, or even the same tasks if the viewpoints of the scene are changed. Therefore, additional efforts are required to match the viewpoints [23], [24].

On the other hand, [10] proposed to constrain features with user-defined dynamics. Comparing to the prior works that fit the task into the features, this work actively manipulates features to follow the user-intended characteristics. Using this concept, [11] proposed $se(3)$ constrained features that the rigid body transformation is provided as a constraint for the features from two images. Our work is influenced by this work, but we extend the $se(3)$ features to represent actual poses of the objects to obtain the 6d-pose trajectories in the Euclidean space. As the $se(3)$ constraint is originated from the rigid body motion in the scene, measuring the $SE(3)$ motion in the images is an important part of our research. The prior work [11] estimates the motion from a point cloud, but we additionally use the optical flow of an RGB image for more robust estimation. The way of estimating motion from images is called the direct method of visual motion estimation [25], which becomes popular with incorporated with the deep neural network [26], [13], [27]. Our motion estimation is motivated from [13], where a motion estimation network is trained in either an unsupervised or semi-supervised way.

III. Preliminaries

Suppose that we are given with stereo video demonstrations, which are a sequence of RGB images $I_t \in [0,1]^{(h,w)}$, and depth images $d_t \in [d_{min}, d_{max}]^{(h,w)}$ where (h,w) is the width and height of the image frame, and t is the time step. There are interactive objects o_t^k for $k = 1, ..., K$ which have continuous movement in the scene, and let us assume that we have a semantic segmentation mask $m_t^k \in [0,1]^{(h,w)}$ which refers to o_t^k in the image. For simplicity, we will omit k in the following algorithm explanation. We use $\Omega^{m_t} = \{(u,v)\}$ as the set of pixels coordinates (u,v) in the mask m_t. Let us assume that the stereo camera is calibrated with camera intrinsics $K_{in} = (c_x, c_y, f)$ where (c_x, c_y) is principal points and f is the focal length. The camera intrinsics will be used for three-dimensional reconstruction or two-dimensional projection between the image and its corresponding point cloud.

Next, we will briefly introduce twist coordinates, which is a Lie algebra $se(3)$ of a rigid body transformation $SE(3)$. The 6d-pose with respect to a frame $\{a\}$ in a three-dimensional space can be represented in twist coordinates $\xi_t^a = [v_t^a; \omega_t^a] \in \mathbb{R}^6$. Let us use $g_{ab} \in SE(3)$ as the rigid body transformation from a frame $\{a\}$ to a frame $\{b\}$. The rigid body transformation is a Lie group of $SE(3)$, and it can be mapped with the twist coordinates by an exponential map $g_{\xi_t^a} = \exp(\hat{\xi_t^a})$ where $\exp(\cdot)$ is a matrix exponential, and $\hat{(\cdot)}$ is a wedge operation. In the later section, we constrain our network features to be $se(3)$ so that they can describe the 6d-pose.

IV. Constraints for $se(3)$ features

Let us say that ξ_t^c is the twist coordinates of the object with respect to a camera frame $\{c\}$. Our network will predict ξ_t^c from an rgb-d frame (I_t, d_t) after training with the self-supervised signal of the motion estimation proposed in [13]. In the two consecutive images I_t and I_{t+1}, the motion between ξ_t^c and ξ_{t+1}^c can be described as the rigid body transformation $g_{o_t o_{t+1}}$.

$$g_{o_t o_{t+1}} = g_{\xi_t^c}^{-1} g_{\xi_{t+1}^c} \tag{1}$$

The rigid body transformation is applied to the three-dimensional points reconstructed from the two-dimensional pixels of the object. For $(u,v) \in \Omega^{m_t}$, which means the pixel coordinates belong to the object mask, the three-dimensional point $\mathbf{X}_t^c$ is reconstructed with depth measurement $d_t(u,v)$ and camera parameters (h, w, c_x, c_y, f).

$$\mathbf{X}_t^c \triangleq \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \frac{d_t(u,v)}{f} \begin{bmatrix} \frac{u}{w} - c_x \\ \frac{v}{h} - c_y \\ f \end{bmatrix} \tag{2}$$

If we assume that the object is a rigid body, all the point $\mathbf{X}_t^c$ referred by Ω^{m_t} is transformed to $\mathbf{X'^c_t}$ after the inverse of $g_{o_t o_{t+1}}$ is applied.

$$\mathbf{X'}_t^c \triangleq \begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = g_{o_t o_{t+1}}^{-1} \mathbf{X}_t^c \tag{3}$$

Finally, the projection of $\mathbf{X'}_t^c$ on the image plane provides (u', v'), which is the expected pixel coordinates of (u,v) after the motion is applied.

$$\begin{bmatrix} u'/w \\ v'/h \end{bmatrix} = \frac{f}{Z'} \begin{bmatrix} X' \\ Y' \end{bmatrix} + \begin{bmatrix} c_x \\ c_y \end{bmatrix} \tag{4}$$

We assume that the photometric intensity of pixels has consistency between the original and transformed pixels when the time interval is small. Therefore, it generates a constraint that $I_{t+1}(u', v')$ has the same value as $I_t(u,v)$. This constraint is implemented by the photometric loss $\mathcal{L}_{photo}$.

$$\mathcal{L}_{photo} = \sum_{(u,v) \in \Omega^{m_t}} ||I_t(u,v) - I_{t+1}(u', v')|| \tag{5}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2020.2968032, IEEE Robotics and Automation Letters

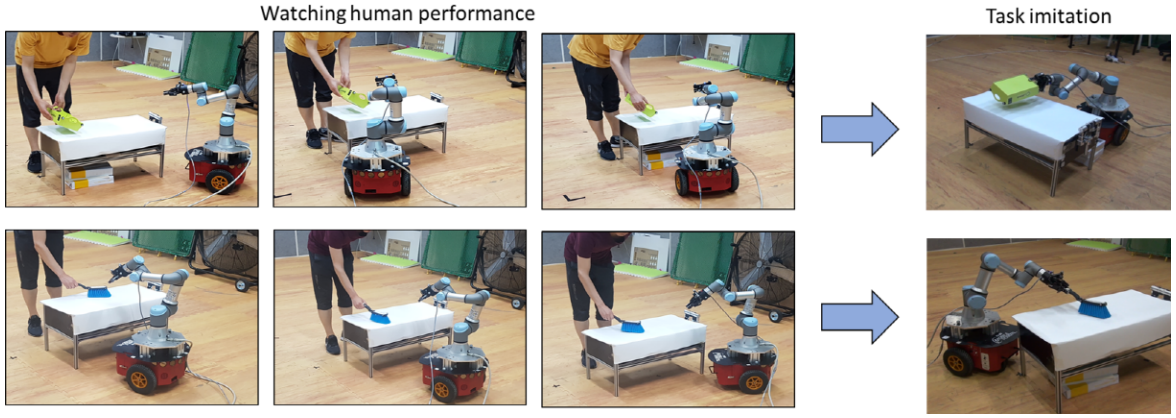

Fig. 1: Research objective: A mobile service robot watches human performances from the different viewpoints, and learns the task via imitation learning.

Likewise, $d_t(u,v)$ moves into $d_t(u,v)+W_t$ where $W_t = Z' - Z$. This motioned depth is constrained to be the same as the next step depth $d_{t+1}(u',v')$. This constraint is implemented by the depth consistency loss $\mathcal{L}_{consis}$.

$$\mathcal{L}_{consis} = \sum_{(u,v)\in\Omega^{m_t}} || \left(d_t(u,v)+W_t\right) - d_{t+1}(u',v') || \quad (6)$$

The photometric and depth losses are calculated by applying $g_{o_t o_{t+1}}$ to (I_t, d_t) forwardly. Likewise we can compute the losses backwardly by applying $g_{o_{t+1}o_t} = g^{-1}_{o_t o_{t+1}}$ to (I_{t+1}, d_{t+1}). Both forward and backward losses are used for network training.

A. Fully self-supervised learning

Our network can be trained fully self-supervised way when no benchmark pose is available. In this case, the $se(3)$ features cannot be anchored on the object only with motion estimation because it provides constraints of the relative pose, not the absolute pose. Therefore, we add the self-supervised volume loss $\mathcal{L}_{vol}$ which provides the constraint for the features to be fixated inside the volume of the object. Note that we assume the segmentation mask is given by a separate module of semantic perception to focus on the pose estimation problem. If necessary, many unsupervised methods for the segmentation of the interest region can be applied [28].

At first, we extract a three-dimensional bounding box which envelops the point cloud of the object with left-front-top coordinates of $(X_{min}, Y_{min}, Z_{min})$ and right-back-bottom coordinates of $(X_{max}, Y_{max}, Z_{max})$. This bounding box can be self-generated from the object mask and its point cloud with margin, for example,

$$X_{max} = \max_{(u,v)\in\Omega^{m_t}} X + X_{margin}. \quad (7)$$

Next, let $T_{\xi_t} = [X_{\xi_t}, Y_{\xi_t}, Z_{\xi_t}]$ be the translation element of g_{ξ_t}, and we define $\mathcal{L}_{vol}$ to be increased when ξ_t is out of the box, and have the consistent minimum value when inside the box.

$$\mathcal{L}_{vol} = \sum_{\zeta\in\{X,Y,Z\}} f(\zeta_{\xi_t}, \zeta_{min}, \zeta_{max}) \quad (8)$$

where $f(\cdot)$ is a quadratic insensitive loss in the following form:

$$f(\zeta,\zeta_{min},\zeta_{max}) = \begin{cases} (\zeta-\zeta_{max})^2 & \text{for } \zeta_{max} \le \zeta \\ 0 & \text{for } \zeta_{min} \le \zeta < \zeta_{max} \\ (\zeta_{min}-\zeta)^2 & \text{for } \zeta < \zeta_{min} \end{cases} \quad (9)$$

B. Mixed learning with sparsely annotated benchmark

Even though our algorithm can be trained in a fully self-supervised way, semi-supervised training is added for a practical reason. In an actual robot application, object localization is estimated by combining the outputs of various methods. In our paper, the optical-flow based pose estimation is used, but it is required to combine various methods, such as hand-eye coordination, structure from motion, or template-based reasoning for integrated localization. We open our algorithm to use supervised signals to incorporate the sparse result from other localization methods.

Let $\{w\}$ be the world frame, $\{r\}$ be the robot frame. As the benchmark pose is measured in the global coordinates, it needs a coordinate transformation from $\{w\}$ to $\{c\}$.

$$g_{wc} = g_{wr} g_{rc}, \quad (10)$$

where g_{wr} is obtained by the robot localization, and g_{rc} refers to the camera extrinsic parameter. Let the available benchmark pose be $\bar{\xi}^w_t$, and then the benchmark loss $\mathcal{L}_{bench}$ is defined as the supervised error from the prediction.

$$\mathcal{L}_{bench} = \left\| g_{wc}\xi^c_t - \bar{\xi}^w_t \right\|^2 \quad (11)$$

In this paper, we use this mixed learning setting to provide closed-loop pose estimation from the assumption that the pose of the static object at the start and end point can be acquired by landmark-based reasoning, and the pose of the moving object is estimated by our optical-flow based self-learning. Therefore, the experiments of the mixed learning in our paper use only both end-points of each demonstration as sparsely

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2020.2968032, IEEE Robotics and Automation Letters

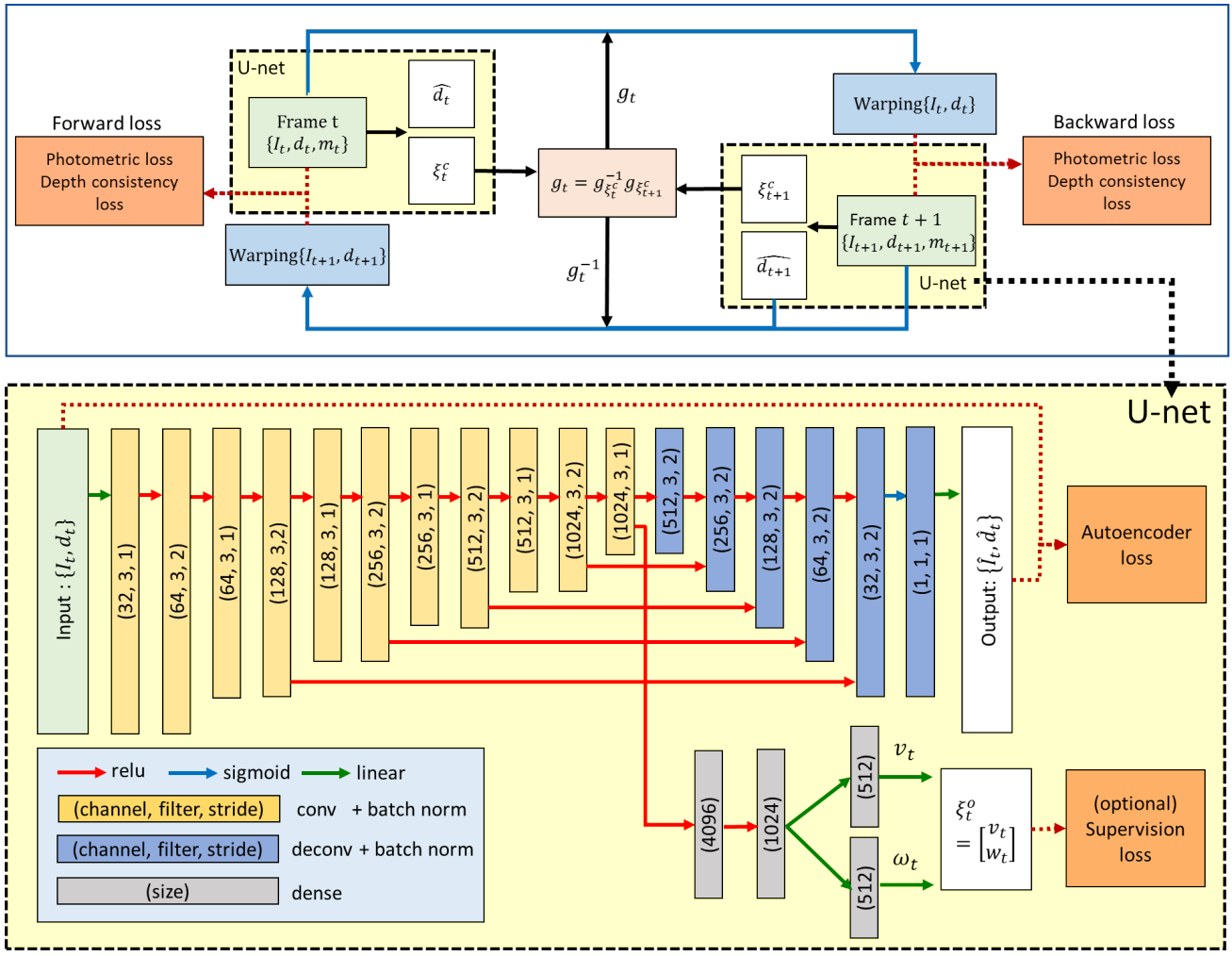

Fig. 2: Designed network and its losses for training. The top figure shows the training system of our network, and the bottom figure shows the detailed U-net structure, which is referred to in the top figure.

annotated poses. We will show that the closed-loop with these both end-poses can improve the accuracy of the estimation. Note that our main interest is self-learning of the pose, but we performed additional experiments for mixed learning for practical applications.

V. Network Design

We design the network, which predicts the object pose from a single rgb-d frame in the test phase but needs the samples of two consecutive images for the training phase. U-net [29] is adopted for a fully convolutional network whose input and output are two-dimensional with channels. The input is the concatenation of an RGB and a depth image (I_t, d_t), and the output is the reconstructed rgb-d image $(\hat{I}_t, \hat{d}_t)$ to make an autoencoder structure. In the bottleneck of the U-net, fully connected layers predict the $se(3)$ features. Two Siamese U-nets are applied to two consecutive inputs of t and $t+1$. The overall network structure is shown in Figure 2. The output channels are trained as an autoencoder to make the decoded output reconstruct the input. This autoencoder structure enhances the features to be distinguishable from two similar consecutive images. The reconstruction error for the RGB image $\mathcal{L}_{auto}$ is the following squared form.

$$\mathcal{L}_{auto} = \sum (I_t - \hat{I}_t)^2 \tag{12}$$

For the depth measurement, there are failed pixels where the depth measurement is missing. We reconstruct the complete depth as the output of the network, and it is used for the depth value in Equation 2 instead of the sensor depth. Let us use $m_t^{nan} \in \{0,1\}^{h \times w}$ as the binary mask whose value is 1 for the pixels not having depth value. Then, the depth reconstruction loss $\mathcal{L}_{depth}$ is computed excluding missing pixels.

$$\mathcal{L}_{depth} = \sum (1 - m_t^{nan}) \odot (d_t - \hat{d}_t)^2 \tag{13}$$

where $\odot$ is element-wise multiplication. The depth value for the pixels with missing depth measurement is completed by minimizing Equations 5 and 6.

VI. Imitation of Feature Trajectories

The mapping from an image to $se(3)$ features is obtained after training, and each demonstration is encoded into feature trajectory $\{\xi_t\}_{t=1}^T$ with length T. Then, trajectory learning

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2020.2968032, IEEE Robotics and Automation Letters

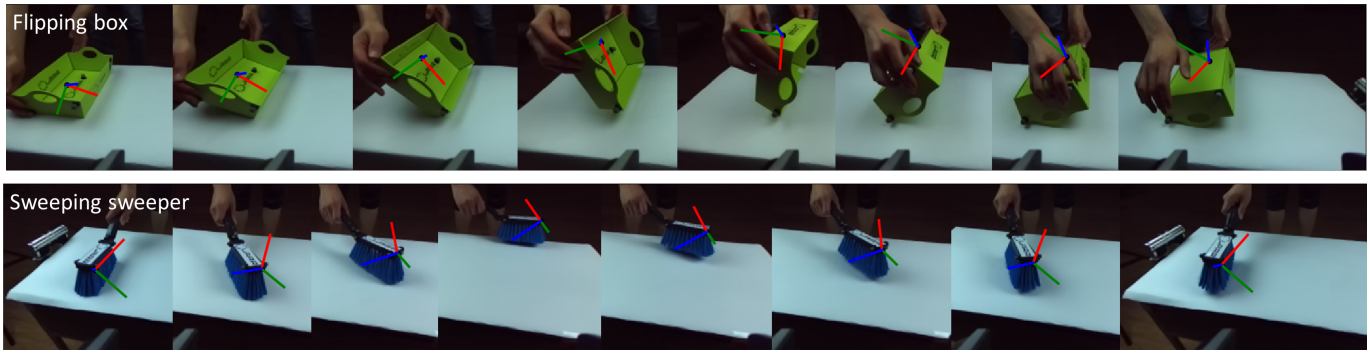

Fig. 3: Task demonstrations and the estimated 6d-pose of objects.

algorithm [12] is applied to find the latent trajectory and the corresponding time indices of the demonstrations in an iterative fashion with a series of expectation-maximization (EM) algorithms – one for the latent trajectory (Kalman smoothing), and the other for the time indices (dynamic time warping). For a detailed description of the algorithm, please refer to the original paper. Let a geometric state be $\xi_t = [v_t, \omega_t]$, and the system input is also the rigid body motion induced by $\Delta\xi$ as $g_{\xi_{t+1}} = g_{\Delta\xi} g_{\xi_t}$. This coordinate transformation can be represented as the adjoint map of the two screw axes.

$$\xi_{t+1} = \left[\mathrm{Adj}_{\Delta\xi}\right] \xi_t = \begin{bmatrix} e^{\hat{\omega}\theta} & [G(\theta)v]\, e^{\hat{\omega}\theta} \\ 0 & e^{\hat{\omega}\theta} \end{bmatrix} \xi_t, \qquad (14)$$

where $\Delta\xi = \theta\,[v, \omega]^T$ for $||\omega|| = 1$, and $G(\theta) = I\theta + \hat{\omega}\frac{\theta^2}{2!} + \hat{\omega}^2\frac{\theta^3}{3!} + \dots$. When $\Delta\xi$ is small, this equation can be linearized with small θ assumption which, yields $e^{\hat{\omega}\theta} \approx I + \hat{\omega}\theta$, and $G(\theta)v \approx \theta v$. Finally, the linearized equation is represented as the following form.

$$\xi_{t+1} \approx \xi_t + \begin{bmatrix} \hat{\omega}\theta & \hat{v}\theta \\ 0 & \hat{\omega}\theta \end{bmatrix} \xi_t = \xi_t + \begin{bmatrix} -\hat{\omega}_t & -\hat{v}_t \\ 0 & -\hat{\omega}_t \end{bmatrix} \Delta\xi. \qquad (15)$$

Then, we define a latent state $[\xi_t; \Delta\xi_t]$ and its geometric dynamics by constant $\Delta\xi_t$ assumption as following form:

$$\begin{bmatrix} \xi_{t+1} \\ \Delta\xi_{t+1} \end{bmatrix} = \begin{bmatrix} I_{6\times6} & A \\ 0 & I_{6\times6} \end{bmatrix} \begin{bmatrix} \xi_t \\ \Delta\xi_t \end{bmatrix}, \qquad (16)$$

where A is the matrix multiplied to $\Delta\xi$ in Equation 15. This linear model is provided for the dynamics of the trajectory learning algorithm, and our extracted feature trajectories are provided for the target demonstrations to be imitated. After the algorithm is optimized, a time-aligned probabilistic trajectory $\{\bar{\xi}_t\}$ with its variance is generated for the reference trajectory of the object. Then, the robot performs an end-effector planning to manipulate the object to follow the reference trajectory, and finally, the motion is executed with the closed-loop control with respect to the robot configuration.

VII. EXPERIMENT

A. Data collection

A stereo camera mounted on a mobile robot records the demonstrations from the different viewpoints while localization of the robot g_{wr} is assumed to be given. The recorded RGB and depth images have a resolution of 168×384 with 30 fps. Our dataset consists of two different household tasks: 1) flipping a box, and 2) sweeping a sweeper. Each task has three different demonstrations which contain 623 frames (flipping a box) and 1253 frames (sweeping a sweeper) in total. These image frames are both used in self-learning and testing as a manner of self-supervised learning. These two tasks contain various types of 6-dof motion as the first task includes a large rolling motion, and the second a large yawing motion with translation. Additionally, four different tasks defined in Table II are conducted to validate our algorithm for various data.

B. Network training and result

The network is trained either in a fully self-supervised or semi-supervised way according to the availability of annotated poses. In this research, the annotated poses are assumed to be given only at the both ends of the demonstrations for semi-supervised training, as we discussed in section IV-B. During the network training, two consecutive images sampled from the demonstration are provided to two Siamese U-nets, and the network parameters are trained to reduce self-supervised losses. After training, the network provides features of each image, which represent the actual 6d-pose of the object in the scene. The features of fully self-supervised training are shown in Figure 3. The RGB colored axes in the figure are the visualized features projected on the image plane. This figure indicates that our trained features represent the actual object pose correctly. While the object moves, the 6d-pose is attached on the fixed point of the object without slipping or floating even when the object is partially occluded. The pose estimation from the various data is summarized at Table II. The results in Table II show the effectiveness of our algorithm through the different type of the object and motion. Two tasks of the experiments are designed to have large occlusion during the demonstration so that our algorithm can be applied to a partially occluded object.

To show the training result in detail, we pick one demonstration from each task, and plot the estimated trajectory and its ground truth in Figure 4. On the left side of the figure, the trajectories are plotted in a three-dimensional space, and the right side shows each component of the twist coordinates with respect to the world frame w. Note that even though ω component of ξ^w ($\xi^w(4-6)$) refers to the orientation of the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2020.2968032, IEEE Robotics and Automation Letters

Method	Flipping box total motion: 0.3694 m, 18.54 rad		Sweeping sweeper total motion: 1.787 m, 8.388 rad	
	$\mathcal{E}_{pos}$ (m)	$\mathcal{E}_{ori}$ (rad)	$\mathcal{E}_{pos}$ (m)	$\mathcal{E}_{ori}$ (rad)
relative motion [13] + init pose	0.1788	3.072	5.457	4.055
point cloud [11] + volume loss	0.1284	1.687	0.3546	0.9398
ours (self-supervised pose)	**0.07486**	**0.2116**	**0.02445**	**0.1033**
ours w/o AE, depth completion	0.04634	0.2709	0.1315	0.8675
ours + sparse pose	**0.01705**	**0.1130**	**0.02413**	**0.0911**

TABLE I: Comparison of performance with prior works with and without sparse annotation.

object, v component of ξ^w ($\xi^w(1-3)$) does not directly refers to the translational motion, but its exponential map incorporated with ω component describes the translational motion. The graph shows that all the components of twist coordinates were estimated correctly in both self-supervised and semi-supervised ways. Although there are some fluctuations in v component of ξ^w due to the limit of the optical-flow based estimation, the estimation did not diverge and recovered sooner thanks to our self-supervised method.

C. Algorithm comparison

For the quantitative evaluation, we compare our algorithm with sfm-net [13] and se3-pose-net [11], and summarize it in Table I. The error was measured in both the position $\mathcal{E}_{pos}$ (m) and orientation $\mathcal{E}_{ori}$ (rad) defined as the root-mean-square from the ground truth. To show the degree of accuracy, we denote the total translation and rotation of the object for each task in the table. Because most other works about pose estimation use supervised learning or templet-based methods, these two baselines can be made comparable with our works by adding some modules for the unsupervised object localization.

At first, the sfm-net, whose original purpose is to find the motion of a camera and a moving object can find the object pose by accumulating the motion to the initial pose. However, the open-loop accumulation shows poor performance as the accumulated motion error makes the pose diverge from the ground truth. Also, the rotation motion cannot be estimated accurately in our data set. As our data set was collected from an actual robot environment, the quality of our data is lower than those used in [13] in that ours has less texture from which features can be extracted, and only a small object region of the image was used for estimation. The network was required to distinguish the smaller change of the object region motion, and it makes the network collapse in rotational motion estimation, which is more vulnerable than translational motion. In our algorithm, the use of the encoder-decoder loss helps to enhance feature representativity and prevent mode collapse. Also, direct estimation of the pose seems to be more stable than the estimation of the motion, because the pose is one-to-one mapping with the object state, and the large training data set can help the generalization of the pose estimation which can compensate the motion estimation error from the optical flow.

Next, the se3-pose-net learns the constrained se3-features from the object point cloud. As the features trained in the se3-pose network are not the actual pose of the object, we added the volume loss introduced in our paper for finding the self-supervised pose. However, self-learning of the pose from the point cloud cannot show the accurate pose estimation performance because the point cloud from sensor measurement is likely to be imperfect, as discussed in [11]. Also, point cloud

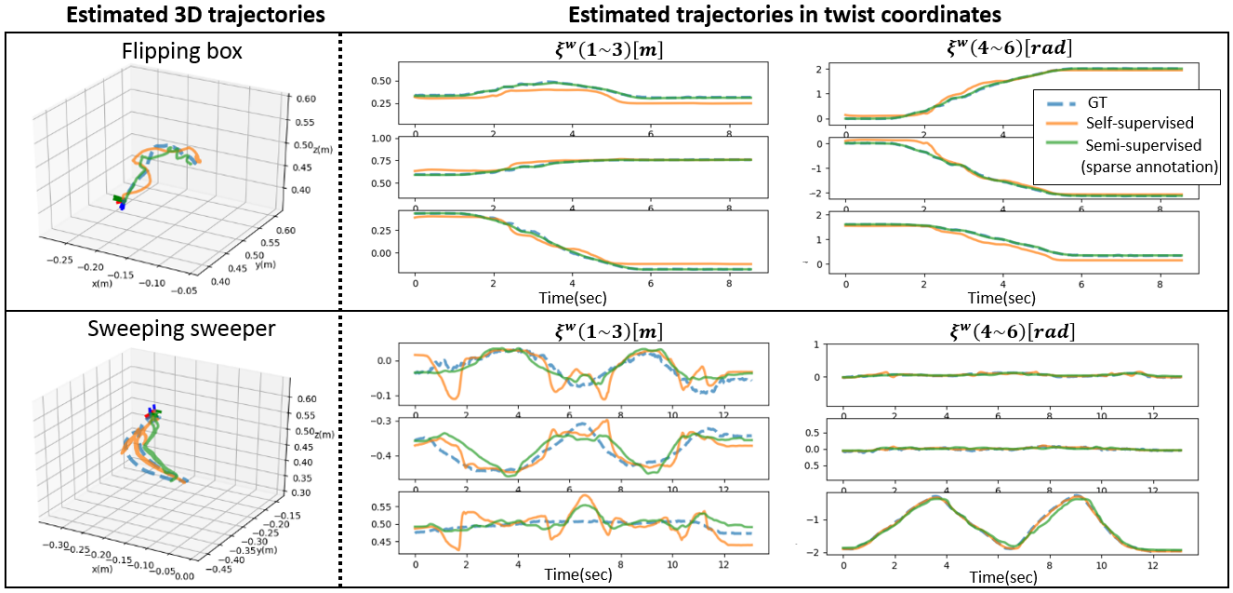

Fig. 4: Estimation results of $se(3)$ features in self-supervised and semi-supervised training.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2020.2968032, IEEE Robotics and Automation Letters

motion could fail to measure the rotation of the symmetric axis when the point cloud before and after the motion does not change. On the other hand, ours use the depth completion guided by photometric loss from the RGB image, and also the combined photometric and the depth consistency loss can prevent the above-mentioned rotation problem.

D. Task imitation

The feature trajectories were estimated from the different demonstrations by the pose estimation network. These trajectories are merged into a nominal trajectory by the trajectory learning algorithm. We plot the result of trajectory learning and the process of the imitation in Figure 5. The demonstrations which have all different temporal speed as shown in the top figure are provided for the trajectory learning algorithm. After the algorithm is optimized, the probabilistic model provides the mean and variance trajectories, which maximize the expectation of the demonstrated trajectories in the aligned time. The temporally aligned demonstrations and the trained probabilistic trajectory are shown in the bottom figure.

Next, the target nominal trajectory is sampled from the probabilistic model, which is denoted as 'Learning nominal trajectory' in the figure. The nominal trajectory is provided to a robot planner so that the corresponding end-effector trajectory is generated. The robot planner uses the object pose from our pose estimation network but requires additional information of the current robot configuration and its localization, which is assumed to be known. The generated trajectory is denoted as the 'End-effector planning' in the figure. Finally, the planned trajectory is executed so that the robot imitates the task in an object-oriented manner. Note that during all the training process in our research, only the video demonstrations are provided to teach a robot. The scenes of actual robot experiments a is denoted as the 'Robot imitation' to show our experimental set-up and task imitation. After the mobile robot approached the task space, it executes the motion to imitate the task.

	total motion		pose error	
	trans-lation (m)	rota-tion (rad)	$\mathcal{E}_{pos}$ (m)	$\mathcal{E}_{ori}$ (rad)
Flipping box	0.3694	18.54	**0.07486**	**0.2116**
Sweeping sweeper	1.787	8.388	**0.02445**	**0.1033**
Stacking block	0.5890	1.963	**0.04784**	**0.06280**
Opening door	0.2847	2.059	**0.01181**	**0.02132**
Moving block (w/ large occlusion)	0.6767	1.862	**0.09217**	**0.2417**
Moving box (w/ large occlusion)	0.5057	4.743	**0.1211**	**0.4064**

TABLE II: The translation and rotation error of fully-unsupervised training in various tasks.

VIII. Conclusion

In this research, we investigate an application of deep learning for a practical robot imitation problem of imitating human performance by video from different views. The proposed scene interpretation module has a network structure whose features are the actual twist coordinates of the interactive objects in the scene. Therefore, our trained features can be used to imitate the task by combining with other modules. For example, the feature trajectories can be transformed into the global coordinates with the robot localization, and the planner and controller can generate joint inputs to follow the geometrical motion of objects. Therefore, the proposed learning structure is more applicable in the point that physically meaningful features can be shared with other modules of the robot. We performed experiments on two different household tasks, and our algorithm showed better results for the pose estimation than prior works. In this work, we focused on learning the se(3) features, but in our future works, we will delve into developing training a robot from the routine daily-life video clips, which includes the unsupervised classification of meaningful motion in the day-life video, learning the selected motion, and the optimal motion planning to execute the imitation.

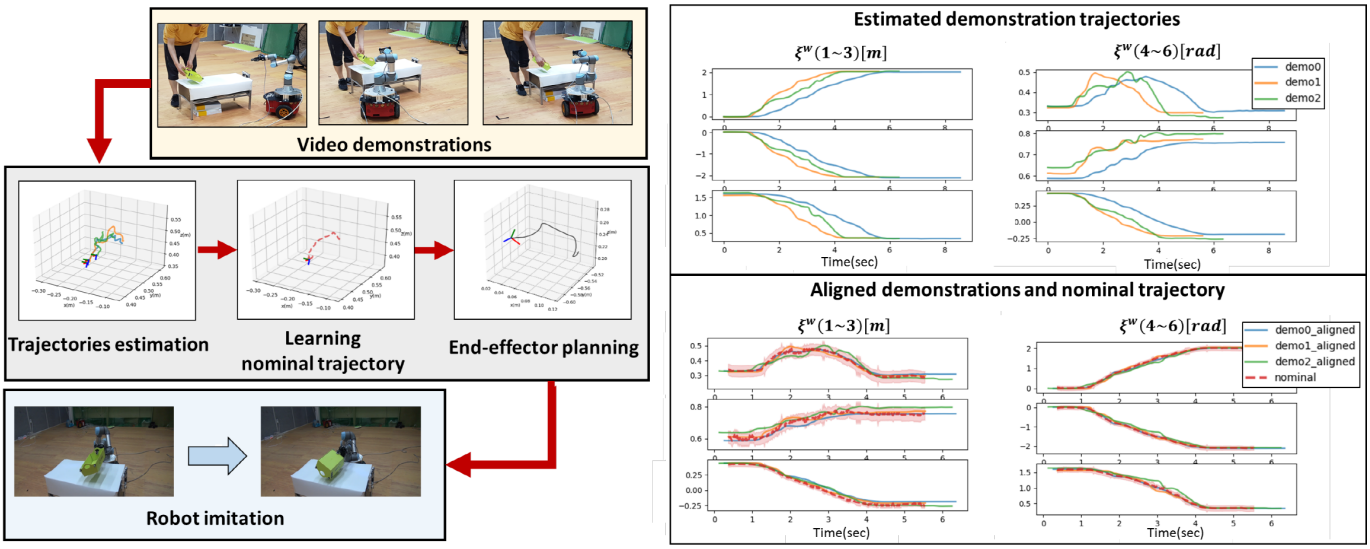

Fig. 5: Our algorithm flow: task imitation from video demonstrations (left), Trajectory learning from different demonstrations of the flipping-a-box (right)

REFERENCES

[1] Sergey Levine et al. "End-to-end training of deep visuomotor policies". In: *The Journal of Machine Learning Research* 17.1 (2016), pp. 1334–1373.

[2] Tianhe Yu et al. "Unsupervised Visuomotor Control through Distributional Planning Networks". In: *arXiv preprint arXiv:1902.05542* (2019).

[3] Josh Merel et al. "Hierarchical visuomotor control of humanoids". In: *arXiv preprint arXiv:1811.09656* (2018).

[4] Chelsea Finn et al. "Deep spatial autoencoders for visuomotor learning". In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2016, pp. 512–519.

[5] AJ Piergiovanni, Alan Wu, and Michael S Ryoo. "Learning Real-World Robot Policies by Dreaming". In: *arXiv preprint arXiv:1805.07813* (2018).

[6] Ali Ghadirzadeh et al. "Deep predictive policy training using reinforcement learning". In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2017, pp. 2351–2358.

[7] Avi Singh et al. "End-to-End Robotic Reinforcement Learning without Reward Engineering". In: *arXiv preprint arXiv:1904.07854* (2019).

[8] Yan Duan et al. "One-shot imitation learning". In: *Advances in neural information processing systems*. 2017, pp. 1087–1098.

[9] Chelsea Finn et al. "One-shot visual imitation learning via meta-learning". In: *arXiv preprint arXiv:1709.04905* (2017).

[10] Manuel Watter et al. "Embed to control: A locally linear latent dynamics model for control from raw images". In: *Advances in neural information processing systems*. 2015, pp. 2746–2754.

[11] Arunkumar Byravan et al. "Se3-pose-nets: Structured deep dynamics models for visuomotor planning and control". In: *arXiv preprint arXiv:1710.00489* (2017).

[12] Pieter Abbeel, Adam Coates, and Andrew Y Ng. "Autonomous helicopter aerobatics through apprenticeship learning". In: *The International Journal of Robotics Research* 29.13 (2010), pp. 1608–1639.

[13] Sudheendra Vijayanarasimhan et al. "Sfm-net: Learning of structure and motion from video". In: *arXiv preprint arXiv:1704.07804* (2017).

[14] Nadia Figueroa, Ana Lucia Pais Ureche, and Aude Billard. "Learning complex sequential tasks from demonstration: A pizza dough rolling case study". In: *The Eleventh ACM/IEEE International Conference on Human Robot Interaction*. IEEE Press. 2016, pp. 611–612.

[15] Tianhao Zhang et al. "Deep imitation learning for complex manipulation tasks from virtual reality teleoperation". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 1–8.

[16] Sylvain Calinon et al. "Learning collaborative manipulation tasks by demonstration using a haptic interface". In: *2009 International Conference on Advanced Robotics*. IEEE. 2009, pp. 1–6.

[17] Sascha Lange and Martin Riedmiller. "Deep auto-encoder neural networks in reinforcement learning". In: *The 2010 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2010, pp. 1–8.

[18] et al Lange Sascha. "Autonomous reinforcement learning on raw visual input data in a real world application". In: *The 2012 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2012, pp. 1–8.

[19] Timothy P Lillicrap et al. "Continuous control with deep reinforcement learning". In: *arXiv preprint arXiv:1509.02971* (2015).

[20] Mark Pfeiffer et al. "From perception to decision: A data-driven approach to end-to-end motion planning for autonomous ground robots". In: *2017 ICRA*. IEEE. 2017, pp. 1527–1533.

[21] Chelsea Finn, Ian Goodfellow, and Sergey Levine. "Unsupervised learning for physical interaction through video prediction". In: *Advances in neural information processing systems*. 2016, pp. 64–72.

[22] Frederik Ebert et al. "Self-supervised visual planning with temporal skip connections". In: *arXiv preprint arXiv:1710.05268* (2017).

[23] Pierre Sermanet et al. "Time-contrastive networks: Self-supervised learning from video". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 1134–1141.

[24] YuXuan Liu et al. "Imitation from observation: Learning to imitate behaviors from raw video via context translation". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 1118–1125.

[25] Jan Stühmer, Stefan Gumhold, and Daniel Cremers. "Real-time dense geometry from a handheld camera". In: *Joint Pattern Recognition Symposium*. Springer. 2010, pp. 11–20.

[26] Tinghui Zhou et al. "Unsupervised Learning of Depth and Ego-Motion From Video". In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017.

[27] Zhichao Yin and Jianping Shi. "GeoNet: Unsupervised Learning of Dense Depth, Optical Flow and Camera Pose". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Vol. 2. 2018.

[28] Siyang Li et al. "Unsupervised video object segmentation with motion-based bilateral networks". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 207–223.

[29] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation". In: *International Conference on Medical image computing and computer-assisted intervention*. Springer. 2015, pp. 234–241.