Learning Robot Skills From Video Data

Daniel Musekamp [*1] **Marco Rettig** [*1]

Abstract

Imitation learning is a common way to teach new skills to robots by learning from human demonstrations. While many existing approaches deploy kinesthetic teaching or teleoperation to gather demonstrations, these methods make data collection cumbersome and laborious for the user. In contrast, learning from external observation provides a potentially more intuitive way of teaching new skills to a robot. Here, the robot can only observe a human demonstrating the task, without any physical interaction or direct target action information given. In this work, we present a method to learn new robotic skills in task space by extracting hand trajectories from human demonstration videos. Afterwards, robotic movement skills are encoded with Probabilistic Movement Primitives from the recorded human trajectories. Our system provides a simple data collection pipeline and requires only few demonstrations to teach a new skill. We show the feasibility of our approach with a simulated 7-DOF Franka Emika Panda arm by learning to draw digits and on a pick-and-place task.

1. Introduction

In imitation learning, a policy is learned from demonstrations (Osa et al., 2018; Fang et al., 2019). It is a promising approach for quickly enabling a robot to perform a new task without requiring the help of expert users, which would be very useful in areas like elderly care or manufacturing. Traditionally, the demonstrations had to be gathered using teleoperation or kinesthetic teaching (Maeda et al., 2016). However, these methods make data collection very laborious, are difficult to use for inexperienced users and will typically prevent naturally looking movements (Maeda et al., 2016). A more convenient setting for the user would be if the robot could simply be trained by watching a human performing the task, which is the goal of *learning from observation* (Torabi et al., 2019). Regardless of its benefits, learning from observation poses new challenges: Firstly, the data lacks target action labels, i.e. at each timestep only the visual state is given, but not the action the robot should take (Liu et al., 2018). Secondly, the robot's kinematics differ greatly from those of the human demonstrating the task, which makes translation to robot movements hard and the demonstrated trajectories may be infeasible for the robot. This is known as the embodiment mismatch (Torabi et al., 2019).

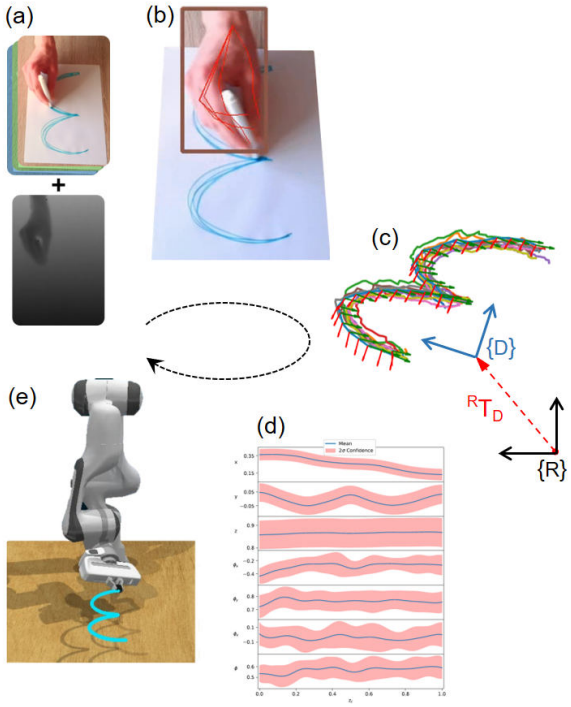

Figure 1. Overview of the proposed pipeline. (a) RGB-D videos of human demonstrations. (b) The human hand poses are estimated from the videos and mapped to 6-DOF end-effector poses. (c) The end-effector trajectories are transformed from the reference frame of the camera to that of the robot. (d) The transformed trajectories are used to learn ProMPs in task space. (e) The learned movements are applied to a robot arm in simulation.

[*]Equal contribution [1]Technische Universität Darmstadt, Germany. Correspondence to: Daniel Musekamp <daniel.musekamp@stud.tu-darmstadt.de>, Marco Rettig <marco.rettig@stud.tu-darmstadt.de>.

Proceedings of the 37^{th} International Conference on Machine Learning, Online, PMLR 119, 2020.

In this work, we present a system that can learn a new skill from only a few human video demonstrations, using an RGB-D camera. First, we build upon the FrankMocap system (Rong et al., 2020) to measure 3D hand trajectories and transform them to the robot task space. The demonstrations are treated as the target robot end-effector trajectories in task space, what reduces the problem of imitating the human motions to the default inverse kinematics problem. In order to generalize and adapt the human demonstrations to new situations, Probabilistic Movement Primitives (Paraschos et al., 2013) are applied, which represent a distribution over the demonstrated movements. We demonstrate the capabilities of our approach in a simulation study by learning from video how to draw digits and to place an object.

2. Related Work

The main challenge of learning from observation is the lack of a directly interpretable target action for each state. To overcome this problem, multiple formulations of the problem have been proposed. (Yu et al., 2018) used a meta-learning approach: During training on a dataset with both human and robot demonstrations, the model does not learn a specific set of behaviors, but rather how to generate a new behavior from only a single video demonstration. While this allows one-shot learning for new demonstrations, which are similar to those in the training set, low generalization abilities to completely new motions are reported. In contrast, time-contrastive networks (Sermanet et al., 2018) learn a view-point independent encoding of human video demonstrations in an unsupervised manner, which can subsequently be transformed by a decoder into robot joint states. (Torabi et al., 2018) showed that the generative adversarial imitation learning algorithm (Ho & Ermon, 2016) can also be adapted to learning from observation.
While the above approaches try to learn the policy from video directly in an end-to-end fashion, a different line of work aims at reconstructing the target action labels from the videos first in order to reduce the problem to the standard imitation learning case. This can, for instance, be achieved by learning a direct mapping from human video demonstrations to robot trajectories (Sharma et al., 2018). Instead, in this work we extract human poses from the videos first and use them as the target action labels. (Zimmermann et al., 2018) combine a module for body pose estimation from RGB-D images with a network that predicts the hand normal vector. The estimated 3D position of the hand is combined with the hand normal vector to form the target end-effector trajectory. In a second step, the raw trajectories are transformed using a graph-based optimization procedure to be successfully executed by the robot.

In contrast to the work of (Zimmermann et al., 2018) we decided to use only a hand joint position estimator instead of one for the whole body for the following reasons: First, this allows to measure demonstrations even if the whole body is not fully visible. This facilitates the recording of typical application scenarios of imitation learning with a single robot arm like pick-and-place, since they are often performed on top of a table, where the requirement to film the whole body would conflict with the optimal recording of the actually important actions above the table. Second, the whole body information would be helpful to fully understand the human movement, but transferring the human joint positions to the robot is hard due to the embodiment mismatch, i.e. it would be hard to transfer the human arm joint positions to the 7-DOF robot arm. Hand pose estimation has been used in imitation learning (Sieb et al., 2020) or in the context of teleoperation (Kofman et al., 2007; Li et al., 2019).

2.1. Hand Pose Estimation

Hand pose estimation refers to the computer vision problem of estimating the position of each hand joint. This can be done either from depth (Moon et al., 2018; Xiong et al., 2019; Huang et al., 2020) or RGB images (Zimmermann & Brox, 2017; Panteleris et al., 2018; Rong et al., 2020). While the problem is considered to be easier using depth data (Doosti, 2019), RGB methods have the advantage that cameras are much more widespread than depth image sensors and hence a much larger amount of data is publicly available. Estimating a 3D pose from 2D images is ill-posed and imposes ambiguities, for example regarding the scale (Zimmermann & Brox, 2017). (Zimmermann & Brox, 2017) presented the first learning based approach to the problem and resolve some of the ambiguities using deep learning to incorporate a learned prior. The output of their system is, however, not in world or camera coordinates, but in a normalized coordinate system relative to the hand. (Panteleris et al., 2018) try to overcome this by formulating an inverse kinematics problem and can thus predict absolute 3D joint positions, but report a large error on the axis perpendicular to the camera, which underlines the difficulty to estimate the depth from RGB images. Another possibility to reconstruct the absolute 3D position is to use RGB video instead of a single image and apply energy minimization to integrate, among other loss terms, temporal consistency constraints (Mueller et al., 2018).

The recently presented FrankMocap system (Rong et al., 2020) provides a full and publicly available pipeline to estimate the 3D hand joint position and 3D hand shape. It uses a hand bounding box detector (Shan et al., 2020) to get the relevant image crops for the subsequent hand pose estimator. The hand pose estimator uses an encoder-decoder architecture to predict the hand pose, shape parameters and global orientation. The predictions are fed into the SMPL-X

body model (Pavlakos et al., 2019) to estimate a mesh representation of the hand and to ensure realistic hand joint configurations. FrankMocap is optimized to work under in-the-wild situations, for example by using motion blur data augmentation. Since their hand pose estimator showed state-of-the-art performance and the system is published as open-source, we decided to use this system to predict the hand trajectories from human demonstration videos.

2.2. Learning Skills using Movement Primitives

Complex tasks in robotics are often solved by combining multiple basic movements. Movement primitives (MPs) enable a compact representation of such basic movements using a set of learned parameters (Paraschos et al., 2013). A powerful MP formulation should allow for a parallel activation and smooth blending of several MPs, their adaptation to varying target positions, velocities and via points or the execution of movements at different speeds. Many of these properties can be incorporated using deterministic MP representations such as Dynamic Movement Primitives (DMPs) (Kober, 2014; Ude et al., 2010). Such representations, however, only capture the mean of the demonstrations of the teacher (Gomez-Gonzalez et al., 2020). In contrast, probabilistic MP formulations also encode the correlations between different degrees of freedom of the robot as well as the variability of the demonstrations and thus can enable a more sensible exploration for robotic systems. For our work, we use Probabilistic Movement Primitives (ProMPs) (Paraschos et al., 2013) which represent a MP as a probability distribution over robot trajectories.

A MP can be represented either in joint space or in task space. (Prasad et al., 2021) used the joint angle trajectories extracted from 3D skeleton data of human demonstrations to learn ProMPs in joint space. They justified their choice with the similarity of the kinematic structure between the human skeleton and the humanoid robot used in their work. In contrast, we are working with a non-humanoid Franka Emika Panda robot arm and mapping directly to joint positions is not possible. As the used FrankMocap system returns 3D hand poses and our learned movements will potentially be adapted to reach desired 3D locations - which is easier when formulating ProMPs in task space (Gomez-Gonzalez et al., 2020) - we choose the task space representation. This, however, involves the risk of not being able to properly perform the learned movements due to the kinematic limitations of the robot. On the other hand, this kind of problem arises whenever trying to map a given task space trajectory to a kinematically feasible joint space trajectory and thus could also not be avoided when using the joint space ProMP representation.

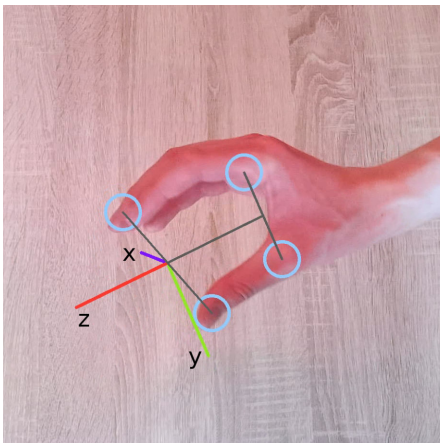

Figure 2. Visualization of the transformation from hand joint estimations to a 6D pose. From four estimated hand joint and tip positions (light blue circles) the end-effector coordinate system is estimated whose x, y, and z-axis are depicted by the blue, green, and red line, respectively.

3. Method

Figure 1 shows the five steps of our pipeline. In the first step, a human performs a set of demonstrations of the desired motion, which are recorded with an RGB-D camera. Second, the hand trajectories are estimated using a hand pose estimator and mapped to 6-DOF poses representing the robot end-effector position and orientation in task space. The trajectories are then transformed from the reference frame of the camera to that of the robot and used to learn a ProMP in task space. In the last step, the actual movement is performed by sampling a single trajectory from the learned ProMP and handing it over to the simulated robot arm via a task space controller.

3.1. Hand Tracking

We build upon the FrankMocap system (Rong et al., 2020) to capture the human hand trajectories. A bounding box tracker (Bewley et al., 2016) is integrated to associate hand bounding boxes between frames and to handle short occlusions. In the next step, the estimated hand joint and finger tip positions have to be mapped to a 6-DOF pose representing the robot end-effector position and orientation in task space. Defining this mapping for every possible hand joint configuration is very hard, since the human hand with its 16 joints provides a large flexibility in the possible hand configurations compared to a two-finger robot gripper. We use the following heuristic to approach the problem (Figure 2): We use only the second thumb joint, the thumb tip, the first index finger joint and the index finger tip. Similar to the work of (Kofman et al., 2007) we set the tool center point to the midpoint between thumb and index finger tip. The end-effector z-axis is defined as the vector from the

midpoint between the second thumb joint and the first index finger joint to the tool center point. We did not choose this midpoint instead of the wrist as proposed in (Kofman et al., 2007) to prevent wrong orientations when the wrist is angled. The end-effector x-axis is defined as the cross product between the vector from the first index to the second hand joint (rightmost gray line in Figure 2) and the z-axis, in order to ensure orthogonality. Since FrankMocap only estimates the hand joint positions in a normalized coordinate system, we use the depth images provided by the RGB-D camera to get an offset to the camera coordinate system.

3.2. Reference Frame Transformation and Scaling

The estimated hand pose trajectories are represented in the reference frame $\{D\}$ of the RGB-D camera which can potentially have an arbitrary orientation and position relative to the demonstrations. Therefore, we have to find a suitable transformation ${}^{R}\boldsymbol{T}_{D}$ relating the coordinates from $\{D\}$ to the reference frame $\{R\}$ of the robot such that the trajectories expressed w.r.t. $\{R\}$ are reachable by the robot. If our goal was only to find a transformation such that the robot could reproduce the shape of the demonstrations w.r.t. to its own reference frame as precisely as possible, we could adapt an approach suggested by (Maeda et al., 2016) by parameterizing the transformation as ${}^{R}\boldsymbol{T}_{D}(\boldsymbol{\theta})$ with $\boldsymbol{\theta} = \{\alpha, \beta, \gamma, r_1, r_2, r_3\}$ where α, β, γ are the angles of the applied rotation and r_1, r_2, r_3 are the coordinates of the applied translation. We could then define a reproduction error which quantifies deviations of a transformed trajectory from the closest kinematically feasible trajectory in terms of position and orientation and set up an optimization procedure returning those parameters $\boldsymbol{\theta}$ that minimize the reproduction error. However, this can result in the transformed trajectories having a completely different position and orientation w.r.t. objects in the scene than before. This is problematic as we ideally want to learn interactions of the robot with objects in the scene. To overcome this problem, we would need to impose constraints on the transformation ${}^{R}\boldsymbol{T}_{D}(\boldsymbol{\theta})$ that take the relative position and orientation of the trajectories w.r.t. the considered objects as well as the position and orientation of the objects w.r.t. the robot into account. This would require, on the one hand, to localize the relevant objects in the given image sequence. Moreover, we would need to estimate the position and orientation of the robot w.r.t. the localized objects only based on the image information, leading to poor estimates in cases where only a small part of the human demonstrator is visible. As we only capture the trajectories of the human hand and do not take any further parts of the human skeleton into account, the latter problem would be even harder to solve.

Therefore, we decided to manually define a transformation from the reference frame of the camera to that of the robot. For this purpose, we assume the position and orientation of the human demonstrator to be known and the robot to have the same position and orientation as the human. We exploit that all our demonstrations are executed on a table and the recorded videos thus always contain major parts of the table plate. To define the common reference frame $\{R\}$ of human and robot, we first extract three points ${}^{\mathrm{D}}\mathbf{o}$, ${}^{\mathrm{D}}\mathbf{c}$ and ${}^{\mathrm{D}}\mathbf{p}$ on the table plate from the RGB-D images which are expressed w.r.t. to the reference frame $\{D\}$ of the RGB-D camera. The point ${}^{\mathrm{D}}\mathbf{o}$ represents the origin of $\{R\}$ and is located near the human demonstrator, while ${}^{\mathrm{D}}\mathbf{c}$ represents the center of the table plate. The point ${}^{\mathrm{D}}\mathbf{p}$ has an arbitrary location on the table plate. We then define the x-axis as the vector ${}^{\mathrm{D}}\mathbf{c}$ - ${}^{\mathrm{D}}\mathbf{o}$ pointing from the origin to the table plate center and the z-axis as the table plate normal vector corresponding to the cross product between ${}^{\mathrm{D}}\mathbf{c}$ - ${}^{\mathrm{D}}\mathbf{o}$ and ${}^{\mathrm{D}}\mathbf{p}$ - ${}^{\mathrm{D}}\mathbf{o}$. Taking the cross product between the z- and x-axis yields the y-axis. Using the unit vectors ${}^{D}\mathbf{e}_{x_R}$, ${}^{D}\mathbf{e}_{y_R}$ and ${}^{D}\mathbf{e}_{z_R}$ of these three axes, we can build the matrix ${}^{D}\boldsymbol{R}_{R}$ that describes the rotation from $\{D\}$ to $\{R\}$:

$$ {}^{D}\boldsymbol{R}_{R} = \left[{}^{D}\mathbf{e}_{x_R} \mid {}^{D}\mathbf{e}_{y_R} \mid {}^{D}\mathbf{e}_{z_R} \right] \tag{1} $$

The homogeneous transformation ${}^{R}\boldsymbol{T}_{D}$ can now be computed via

$$ {}^{R}\boldsymbol{T}_{D} = \left[\begin{array}{c|c} {}^{R}\boldsymbol{R}_{D} & -{}^{R}\boldsymbol{R}_{D}\,{}^{D}\mathbf{o} \\ \mathbf{0}^{T} & 1 \end{array}\right] \tag{2} $$

where ${}^{R}\boldsymbol{R}_{D}$ is the transposed rotation matrix ${}^{D}\boldsymbol{R}_{R}$.

After the transformation, we verify whether the trajectories fit entirely inside the robot workspace and apply a downscaling if necessary. For this purpose, we compute the ratio between the workspace range $\Delta r^{\mathrm{WS}} = r^{\mathrm{WS}}_{\mathrm{max}} - r^{\mathrm{WS}}_{\mathrm{min}}$ and the trajectory range $\Delta r^{\mathrm{T}} = r^{\mathrm{T}}_{\mathrm{max}} - r^{\mathrm{T}}_{\mathrm{min}}$ for each dimension $r \in \{x, y, z\}$ and define the scaling factor s as the smallest among the three computed ratios

$$ s := \min \left\{ \frac{\Delta x^{\mathrm{WS}}}{\Delta x^{\mathrm{T}}}, \frac{\Delta y^{\mathrm{WS}}}{\Delta y^{\mathrm{T}}}, \frac{\Delta z^{\mathrm{WS}}}{\Delta z^{\mathrm{T}}} \right\} \tag{3} $$

If $s < 1$, each trajectory is downscaled by first subtracting the initial trajectory position $\boldsymbol{x}_0$ from all positions $\boldsymbol{x}_{\mathrm{i=0...T}}$, then multiplying the resulting positions $\boldsymbol{x}_{\mathrm{i}} - \boldsymbol{x}_0$ with s and finally adding the initial position again:

$$ \boldsymbol{x}_i^{\mathrm{scaled}} = s(\boldsymbol{x}_i - \boldsymbol{x}_0) + \boldsymbol{x}_0 \tag{4} $$

Subtracting and then adding $\boldsymbol{x}_0$ again ensures that the initial position of every trajectory remains the same as before the scaling. The resulting trajectories are then translated such that they lie entirely within the robot workspace while being centered in x-y direction and lying slightly above the lower workspace boundary in z direction.

3.3. Learning ProMPs

We use ProMPs to learn parameterized distributions over the transformed task space trajectories. Learning a ProMP

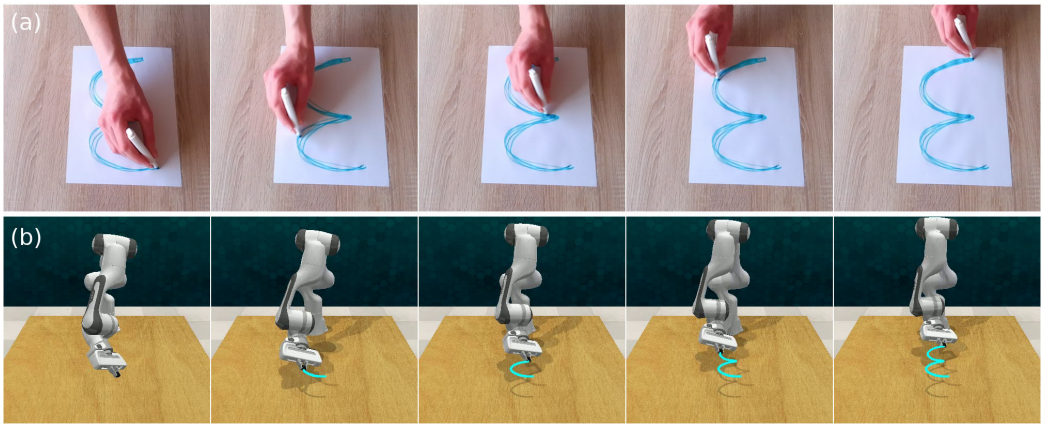

Figure 3. (a) Example of a demonstrated motion. (b) Robot following the mean of the learned ProMP in simulation.

requires at least two demonstrations of the same type of movement (Paraschos et al., 2013). A single demonstration thereby corresponds to a trajectory $\boldsymbol{\tau} = [\mathbf{y}_1 ... \mathbf{y}_\mathrm{T}]$, where $\mathbf{y}_\mathrm{t}$ is the vector of all joint positions and velocities at time t. The probability of observing $\mathbf{y}_\mathrm{t}$ is given as a linear basis function model

$$p(\mathbf{y}_t|\boldsymbol{\omega}) = \mathcal{N}\left(\mathbf{y}_t|\boldsymbol{\Psi}_t^T\boldsymbol{\omega}, \boldsymbol{\Sigma}_y\right) \tag{5}$$

with

$$\boldsymbol{\Psi}_t = \begin{bmatrix} \boldsymbol{\Phi}_t & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \boldsymbol{\Phi}_t \end{bmatrix} \tag{6}$$

where $\boldsymbol{\Phi}_\mathrm{t} = [\phi_\mathrm{t}, \dot{\phi}_\mathrm{t}]$ is the $n \times 2$ dimensional time-dependant basis matrix for the joint positions and velocities, the vector $\boldsymbol{\omega}$ contains the weights of the n basis functions for each joint and $\boldsymbol{\Sigma}_y$ is Gaussian noise.

Variations in the trajectories as well as the covariance between joints are captured by introducing a Gaussian distribution $p(\boldsymbol{\omega}; \boldsymbol{\theta}_\omega) = \mathcal{N}(\boldsymbol{\omega}|\boldsymbol{\mu}_\omega, \boldsymbol{\Sigma}_\omega)$ over the weight vector $\boldsymbol{\omega}$ with mean $\boldsymbol{\mu}_\omega$ and covariance $\boldsymbol{\Sigma}_\omega$. The distribution of $\mathbf{y}_\mathrm{t}$ can now be expressed in terms of these parameters by marginalizing out $\boldsymbol{\omega}$

$$p(\mathbf{y}_t; \boldsymbol{\theta}_\omega) = \mathcal{N}\left(\mathbf{y}_t|\boldsymbol{\Psi}_t^T\boldsymbol{\mu}_\omega, \boldsymbol{\Psi}_t^T\boldsymbol{\Sigma}_\omega\boldsymbol{\Psi}_t + \boldsymbol{\Sigma}_y\right) \tag{7}$$

The ProMP parameters $\boldsymbol{\mu}_\omega$ and $\boldsymbol{\Sigma}_\omega$ can be learned by determining an optimal weight vector $\boldsymbol{\omega}_\mathrm{i}$ for every trajectory $\boldsymbol{\tau}_\mathrm{i}$ using ridge regression,

$$\boldsymbol{\omega}_i = (\boldsymbol{\Psi}^T\boldsymbol{\Psi} + \lambda\boldsymbol{I})^{-1}\boldsymbol{\Psi}^T\boldsymbol{\tau}_i \tag{8}$$

where $\boldsymbol{\Psi} = [\boldsymbol{\Psi}_1 ... \boldsymbol{\Psi}_T]$, and then computing $\boldsymbol{\mu}_\omega$ and $\boldsymbol{\Sigma}_\omega$ as the mean and covariance of the estimated weight vectors (Prasad et al., 2021).

Adaptation of a learned ProMP to new target positions, velocities or via points can be achieved by means of conditioning. For this purpose, we apply Bayes theorem with a desired observation $\{\mathbf{y}_t^*, \boldsymbol{\Sigma}_y^*\}$, where $\mathbf{y}_t^*$ is the desired position and velocity vector at time t and $\boldsymbol{\Sigma}_y^*$ describes the accuracy of the desired observation:

$$p(\boldsymbol{\omega}|\mathbf{y}_t^*, \boldsymbol{\Sigma}_y^*) \propto \mathcal{N}\left(\mathbf{y}_t^*|\boldsymbol{\Psi}_t^T\boldsymbol{\omega}, \boldsymbol{\Sigma}_y^*\right) p(\boldsymbol{\omega}) \tag{9}$$

Maximizing this expression yields again a Gaussian distribution for $\boldsymbol{\omega}$ with updated mean and covariance

$$\boldsymbol{\mu}_\omega^* = \boldsymbol{\mu}_\omega + \boldsymbol{K}(\mathbf{y}_t^* - \boldsymbol{\Psi}_t^T\boldsymbol{\mu}_\omega) \tag{10}$$

$$\boldsymbol{\Sigma}_\omega^* = \boldsymbol{\Sigma}_\omega - \boldsymbol{K}\boldsymbol{\Psi}_t^T\boldsymbol{\Sigma}_\omega \tag{11}$$

where

$$\boldsymbol{K} = \boldsymbol{\Sigma}_\omega\boldsymbol{\Psi}_t(\boldsymbol{\Sigma}_y^* + \boldsymbol{\Psi}_t^T\boldsymbol{\Sigma}_\omega\boldsymbol{\Psi}_t)^{-1} \tag{12}$$

To be able to modulate the execution speed of a movement, the actual trajectory time T is normalized by introducing a phase variable $z_\mathrm{t} = (t - t_0)/T \in [0, 1]$ which decouples the movement from the time signal, such that $\boldsymbol{\phi}_\mathrm{t} = \boldsymbol{\phi}(z_\mathrm{t})$.

4. Experiments

We evaluate our approach on two example tasks. The first is to learn how to reproduce handwritten digits and the second one a common pick-and-place application. The experiments are conducted in simulation with CoppeliaSim (Rohmer et al., 2013) and PyRep (James et al., 2019).

4.1. Learning to Draw Digits

The goal of the first experiment is to reproduce handwritten digits, which is a toy task to show that complex, non-linear trajectories can be reproduced. For each of the ten digits, ten demonstrations are recorded using an Azure Kinect RGB-D camera. In each demonstration video, one digit is drawn on an A4 sheet. An example demonstration can be seen in Figure 3(a).

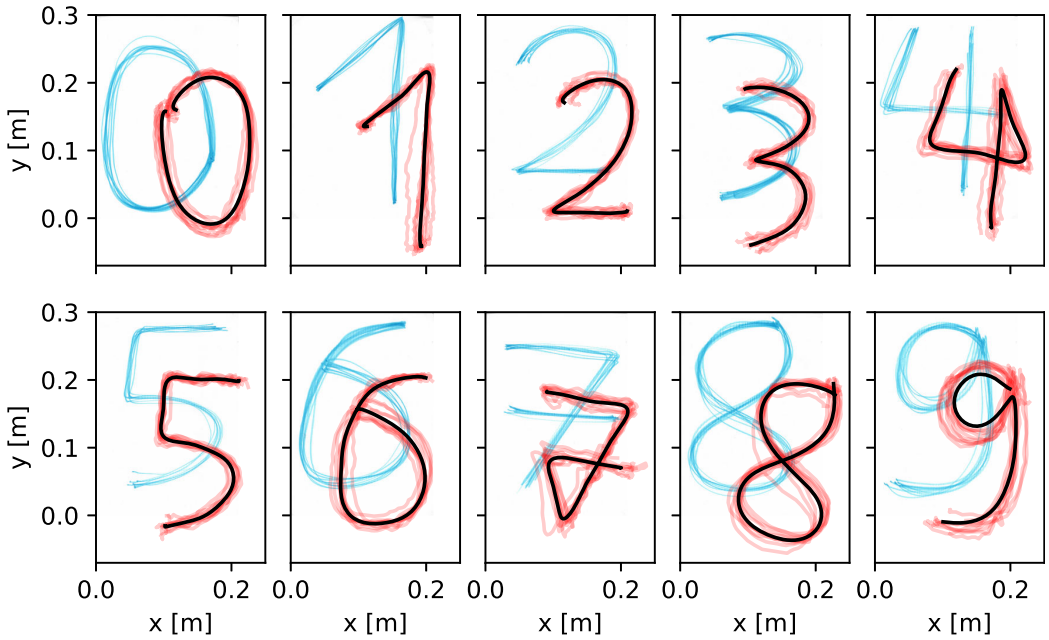

Figure 4. 2D components of the estimated trajectories (red), visualized over the scanned sheets of the drawn numbers (blue). The trajectory components have been transformed to the sheet's position in the world frame. The black lines indicate the mean of the learned ProMPs.

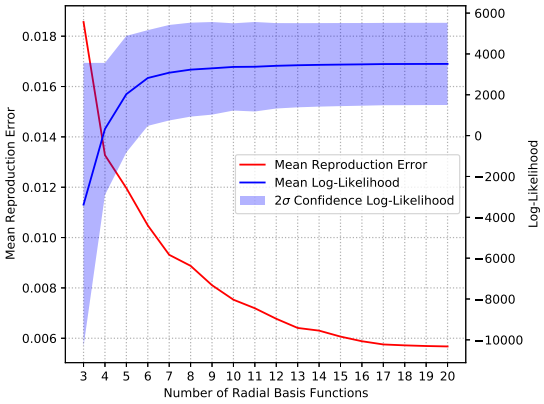

Figure 5. Mean reproduction error and log-likelihood of the demonstrations given the learned ProMPs depending on the number of basis functions.

We estimate a task space trajectory of the human hand from each video using the FrankMocap system and the integrated bounding box tracker. Figure 4 shows the distribution of the estimated trajectories in the table plane. The shape of the demonstrations is well captured by the estimates. However, there is also a shift between the handwritten digits and the task space trajectories. One part of this difference is natural because the finger tips are not exactly located at the tip of the pen. However, there are also two sources of error leading to the shift. Firstly FrankMocap can mislocate the finger tip at a position closer to the palm, especially when being in contact with an object and/or under partial occlusion. Secondly, the shift may also be caused by the normalized image coordinate system. We estimate only one offset from the camera system to the hand coordinate system using the depth data and add this offset to all joint positions, but do not rescale the hand coordinate system. Since the hand pose estimation using only RGB images is not able to reproduce world coordinates, the hand coordinate system is simply normalized to an average hand size. This causes the differences between the joints which are used to calculate the tool center point to be under- or overestimated, depending on the person's hands, which leads to a system error depending on the demonstrator. Simply using the depth data to get real world coordinates for each joint position in the image would, however, also be difficult since a small displacement of the estimated position from a finger joint to a background pixel would cause big errors. The usage of a depth based hand pose estimator would overcome this problem, but the existing open source systems tend to be harder to use and adapt.

In the next step, we evaluate the learned ProMPs. The estimated task space trajectories are temporally aligned to account for different movement speeds. Afterwards, the coordinate system transformation is performed as described in section 3.2. Given the centered trajectories, the ProMPs are learned using radial basis functions with a basis width of 0.01, as proposed by (Prasad et al., 2021). Based on an analysis of the mean reproduction error depending on the number of basis functions (Figure 5), we choose 18 basis

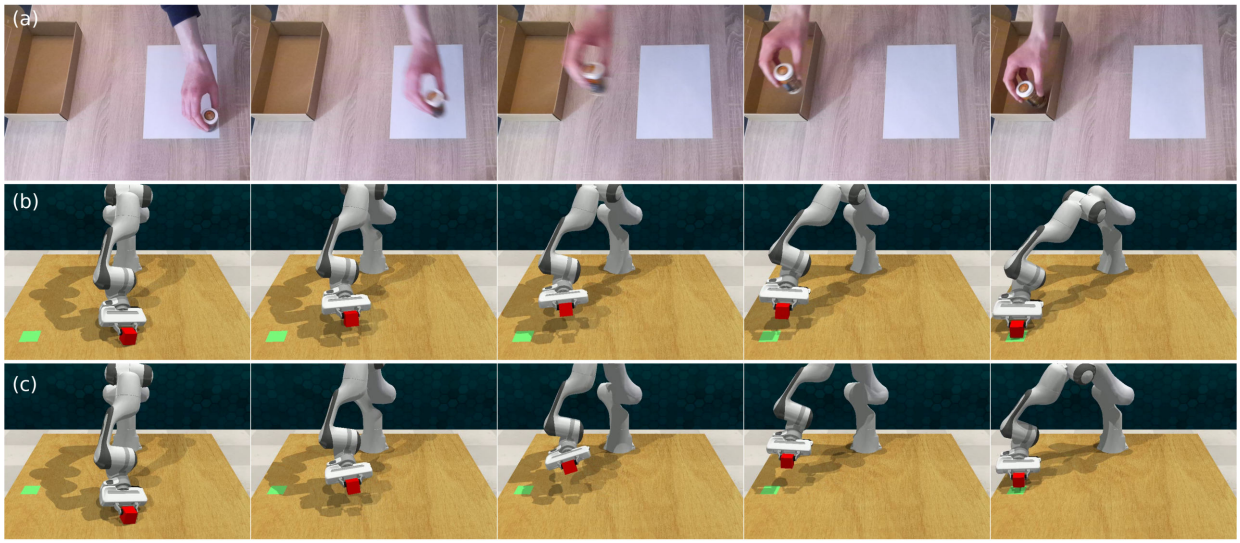

Figure 6. Example demonstration of the placing (a) and execution of the conditioned ProMP to two different target positions (b-c)

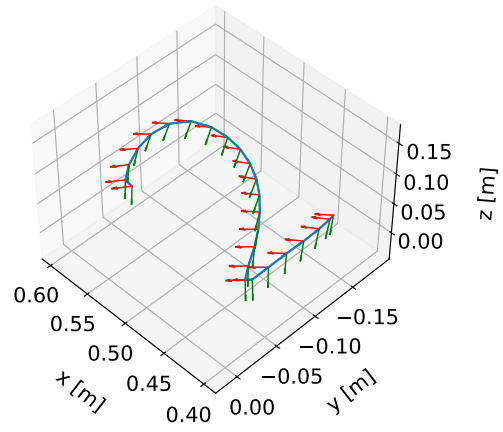

Figure 7. Mean trajectory of the learned ProMP. The red and green arrows represent the z- and y-axis of the end-effector, respectively.

functions. As Figure 4 shows, the ProMPs can capture the shape of the demonstrations. This holds also for the 3D position and orientation. Figure 7 shows the mean of the ProMP learned from the demonstrations of the number 2 including the end-effector orientation. The trajectory has only a small variation along the z-axis, showing the good replication of the demonstrations which are also performed in the plane. The smooth end-effector orientation is also in good correspondence with the demonstrated movements, where the hand orientation was also slanted and almost constant in time.

All learned ProMPs could be executed by the robot in simulation without any further adaptation. Figure 3(b) shows the robot following the mean of the trajectory distribution next to one of the corresponding demonstrations.

4.2. Pick and Place

In the second experiment we evaluate our approach on the common pick-and-place scenario. We gather ten demonstrations in which small objects are picked from the table center and placed on the right table side onto a target area. We manually segment the placing motion from the demonstration video. The hand detector's ability (Shan et al., 2020) to detect the contact state and contact object bounding boxes could also be used for segmenting the placing motion. However, prior experiments showed that the hand detector often had false positives, e.g. it often predicts the table as the object in contact. Since this task consists of an object interaction which is done at different table positions, we subtract the starting point from each demonstration. We assume that the object position is known when applying the ProMP and shift the ProMP in space according to it. The grasping problem is not considered part of the method, and thus we assume it to be given. The robot is brought to the placing ProMP's initial pose with a simple linear trajectory. We select 12 radial basis functions with a basis width of 0.01. Figure 6(a) shows an example of a demonstration. The robot has successfully learned the grasping pose from the human and also follows the demonstrated trajectories (Figure 6(b-c)). Using the conditioning property of the ProMPs, the robot is able to place the objects at different target locations. While the robot could follow the trajectory when starting in the center, we noticed that it can fail to execute the trajectory when shifting the starting point out of the center. The problem is that the human grasped the objects from a relatively low angle, while it would be more natural for the robot to grasp them more from above. In future work, these feasibility problems could be treated by integrating a better motion planner or task space controller, which tries to stay

as close as possible to the demonstration while taking the kinematic constraints into account (Kang et al., 2020).

5. Conclusion and Future Work

In this work, we present a system that is capable of learning robot skills only based on RGB-D videos of human demonstrations. We use the FrankMocap system (Rong et al., 2020) to estimate only the hand poses of the demonstrator instead of its entire arm or body poses. In this way, we avoid the problem of relating the joint positions of the human skeleton to the kinematic structure of the robot and can also use videos where a large part of the human body is not visible. We define a simple mapping from the estimated hand joint and finger tip positions to 6-DOF task space poses of the two-finger robot gripper used in our work. Applying a manual transformation of the resulting end-effector trajectories from the reference frame of the camera to that of the robot allows us to preserve the original orientation of the trajectories w.r.t. relevant objects in the scene. We use the transformed trajectories to learn ProMPs in task space which are applied to a Franka Emika Panda robot arm in simulation. Our experiments show that, using the implemented system, the robot is able to accurately reproduce the drawing of digits and the placing of objects on a table plate only from observations. Due to the ProMP representation of the demonstrated trajectories, the robot can also adapt the placing movement to new target positions while staying close to the learned trajectory distribution.

While, in our experiments, the estimated hand trajectories were feasible for the robot when centered in the robot workspace, they could often not be executed by the robot when being shifted outside of the workspace center. The feasibility of the desired trajectories could be guaranteed by integrating an optimization procedure that generates a feasible joint trajectory for a given task space trajectory (Kang et al., 2020). This could be expanded by directly incorporating a loss for the likelihood of the optimized trajectory under the ProMP. Moreover, our system is currently based on RGB-D videos of which only small quantities are publicly available. Using an RGB-only hand pose estimator which can also predict world coordinates would allow to learn robot skills from any RGB video and therefore to leverage huge public data sets. This could be achieved by augmenting the FrankMocap system with an optimization procedure to ensure temporal consistency (Mueller et al., 2018).

References

Bewley, A., Ge, Z., Ott, L., Ramos, F., and Upcroft, B. Simple Online and Realtime Tracking. In *2016 IEEE International Conference on Image Processing (ICIP)*, pp. 3464–3468. IEEE, 2016.

Doosti, B. Hand Pose Estimation: A Survey. *arXiv preprint arXiv:1903.01013*, 2019.

Fang, B., Jia, S., Guo, D., Xu, M., Wen, S., and Sun, F. Survey of imitation learning for robotic manipulation. *International Journal of Intelligent Robotics and Applications*, 3(4):362–369, 2019.

Gomez-Gonzalez, S., Neumann, G., Schölkopf, B., and Peters, J. Adaptation and Robust Learning of Probabilistic Movement Primitives. *IEEE Transactions on Robotics*, 36(2):366–379, 2020.

Ho, J. and Ermon, S. Generative Adversarial Imitation Learning. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pp. 4572–4580, 2016.

Huang, W., Ren, P., Wang, J., Qi, Q., and Sun, H. AWR: Adaptive Weighting Regression for 3D Hand Pose Estimation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 11061–11068, 2020.

James, S., Freese, M., and Davison, A. J. PyRep: Bringing V-REP to Deep Robot Learning. *arXiv preprint arXiv:1906.11176*, 2019.

Kang, M., Shin, H., Kim, D., and Yoon, S.-E. TORM: Fast and Accurate Trajectory Optimization of Redundant Manipulator given an End-Effector Path. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2020)*, pp. 9417–9424. IEEE Robotics and Automation Society/Robotics Society of Japan, 2020.

Kober, J. Learning Motor Skills: From Algorithms to Robot Experiments. *it-Information Technology*, 56(3):141–146, 2014.

Kofman, J., Verma, S., and Wu, X. Robot-Manipulator Teleoperation by Markerless Vision-Based Hand-Arm Tracking. *International Journal of Optomechatronics*, 1 (3):331–357, 2007.

Li, S., Ma, X., Liang, H., Görner, M., Ruppel, P., Fang, B., Sun, F., and Zhang, J. Vision-based Teleoperation of Shadow Dexterous Hand using End-to-End Deep Neural Network. In *2019 International Conference on Robotics and Automation (ICRA)*, pp. 416–422. IEEE, 2019.

Liu, Y., Gupta, A., Abbeel, P., and Levine, S. Imitation from Observation: Learning to Imitate Behaviors from Raw Video via Context Translation . In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1118–1125. IEEE, 2018.

Maeda, G., Ewerton, M., Koert, D., and Peters, J. Acquiring and Generalizing the Embodiment Mapping From Human Observations to Robot Skills. *IEEE Robotics and Automation Letters*, 1(2):784–791, 2016.

Moon, G., Chang, J. Y., and Lee, K. M. V2V-PoseNet: Voxel-to-Voxel Prediction Network for Accurate 3D Hand and Human Pose Estimation from a Single Depth Map. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5079–5088, 2018.

Mueller, F., Bernard, F., Sotnychenko, O., Mehta, D., Sridhar, S., Casas, D., and Theobalt, C. GANerated Hands for Real-Time 3D Hand Tracking from Monocular RGB. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 49–59, 2018.

Osa, T., Pajarinen, J., Neumann, G., Bagnell, J. A., Abbeel, P., Peters, J., et al. An Algorithmic Perspective on Imitation Learning. *Foundations and Trends in Robotics*, 7 (1-2):1–179, 2018.

Panteleris, P., Oikonomidis, I., and Argyros, A. Using a Single RGB Frame for Real Time 3D Hand Pose Estimation in the Wild. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 436–445. IEEE, 2018.

Paraschos, A., Daniel, C., Peters, J., and Neumann, G. Probabilistic Movement Primitives. In *Advances in Neural Information Processing Systems*, pp. 2616–2624, 2013.

Pavlakos, G., Choutas, V., Ghorbani, N., Bolkart, T., Osman, A. A., Tzionas, D., and Black, M. J. Expressive Body Capture: 3D Hands, Face, and Body from a Single Image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10975–10985, 2019.

Prasad, V., Stock-Homburg, R., and Peters, J. Learning Human-like Hand Reaching for Human-Robot Handshaking. *arXiv preprint arXiv:2103.00616*, 2021.

Rohmer, E., Singh, S. P., and Freese, M. V-REP: A versatile and scalable robot simulation framework. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1321–1326. IEEE, 2013.

Rong, Y., Shiratori, T., and Joo, H. FrankMocap: Fast Monocular 3D Hand and Body Motion Capture by Regression and Integration. *arXiv preprint arXiv:2008.08324*, 2020.

Sermanet, P., Lynch, C., Chebotar, Y., Hsu, J., Jang, E., Schaal, S., Levine, S., and Brain, G. Time-Contrastive Networks: Self-Supervised Learning from Video . In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1134–1141. IEEE, 2018.

Shan, D., Geng, J., Shu, M., and Fouhey, D. F. Understanding Human Hands in Contact at Internet Scale. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9869–9878, 2020.

Sharma, P., Mohan, L., Pinto, L., and Gupta, A. Multiple Interactions Made Easy (MIME) : Large Scale Demonstrations Data for Imitation. In *Conference on Robot Learning*, pp. 906–915. PMLR, 2018.

Sieb, M., Xian, Z., Huang, A., Kroemer, O., and Fragkiadaki, K. Graph-Structured Visual Imitation . In *Conference on Robot Learning*, pp. 979–989. PMLR, 2020.

Torabi, F., Warnell, G., and Stone, P. Generative Adversarial Imitation from Observation. *arXiv preprint arXiv:1807.06158*, 2018.

Torabi, F., Warnell, G., and Stone, P. Recent Advances in Imitation Learning from Observation. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI)*, 2019.

Ude, A., Gams, A., Asfour, T., and Morimoto, J. Task-Specific Generalization of Discrete and Periodic Dynamic Movement Primitives. *IEEE Transactions on Robotics*, 26(5):800–815, 2010.

Xiong, F., Zhang, B., Xiao, Y., Cao, Z., Yu, T., Zhou, J. T., and Yuan, J. A2J: Anchor-to-Joint Regression Network for 3D Articulated Pose Estimation from a Single Depth Image . In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 793–802, 2019.

Yu, T., Finn, C., Xie, A., Dasari, S., Zhang, T., Abbeel, P., and Levine, S. One-Shot Imitation from Observing Humans via Domain-Adaptive Meta-Learning . *arXiv preprint arXiv:1802.01557*, 2018.

Zimmermann, C. and Brox, T. Learning to Estimate 3D Hand Pose from Single RGB Images. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 4903–4911, 2017.

Zimmermann, C., Welschehold, T., Dornhege, C., Burgard, W., and Brox, T. 3D Human Pose Estimation in RGBD Images for Robotic Task Learning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1986–1992. IEEE, 2018.