A New Phase Determination Algorithm for Iterative Learning of Human-Robot Collaboration

Mihael Simonič, Miha Deniša, Aleš Ude, and Bojan Nemec

Abstract— **In this paper we discuss a methodology for learning human-robot collaboration tasks by human guidance. In the proposed framework, the robot learns the task in multiple repetitions of the task by comparing and adapting the performed trajectories so that the robot's performance naturally evolves into a collaborative behavior. When comparing the trajectories of two learning cycles, the problem of accurate phase determination arises because the imprecise phase determination affects the precision of the learned collaborative behavior. To solve this issue, we propose a new projection algorithm for measuring the similarity of two trajectories. The proposed algorithm was experimentally verified and compared to the performance of dynamic time warping in learning of human-robot collaboration tasks with Franka Emika Panda collaborative robot.**

I. Introduction

An important theme in the emerging Industry 5.0 agenda is human-robot co-working [1]. The human-centred production paradigm provides tight collaboration between humans and robots whenever possible, sharing common place and common tasks. This need dictates the development of new procedures and algorithms for natural collaboration between robots and humans. One of the key requirements remains the fast deployment of robots to perform a new task, where little or even no programming is required. Learning from Demonstration (LfD) [2] is widely applied to shorten the programming time and to reduce the required skill level of the operators. LfD enables us to learn or modify robot policies in a natural way rather than by coding in a robot-oriented programming language.

LfD is in itself a kind of physical collaboration between a human and a robot, in which the human takes an active role. We say a robot is a leader and a human is a follower. This role is preferably not fixed. Instead, a humans and a robots should share the task dynamically, taking into account the requirements of the task and the capabilities of both agents. In this way, it is possible to combine the accuracy and repeatability of a robot and the adaptability and flexibility of the human operator. Several different human-robot cooperation schemes were proposed in the past decade, also characterized by the underlying policy representation. As pointed out by Calinon et al. [3], a motor skill necessary to follow the desired path should also cope with the possible variation of coordination patterns during the movement. One way to ensure this property is to encode the task as a non-linear dynamic system, which ensures global asymptotic stability at the target [4]. However, multiple policy demonstrations are required to learn such dynamic systems. Calinon et al. [5] applied Gaussian mixture model (GMM) to sequentially superimpose systems with varying full stiffness matrices. This method allows to arbitrary set the direction of the compliance and defines virtual guides [6]. It can be used for learning of physical collaborative human-robot actions [7]. Ewerton et al. [8] proposed a mixture of interaction primitives (IP). Probabilistic motion primitives (ProMP) enable the encoding of stochastic collaboration policies [9]. Many human-robot collaboration schemes rely on Dynamic Motion Primitives (DMP), where a single demonstration suffices to encode the policy, such as bi-manual crate lid placing [10]. Learning of robot stiffness according to the path variations while decoupling learning of spatial and temporal part of the policy was proposed in [11]. Another concept based on DMPs are the interaction primitives (IP) [12], where a collaborative behavior is represented by maintaining the distribution over DMP parameters of the policy.

The work presented in this paper is based on Speed-scaled Cartesian space Dynamic Movement Primitives (SCDMPs) [13], [14] and an algorithm for stiffness adaptation along the motion trajectory [11], which we extend with a more efficient velocity learning algorithm [15]. Furthermore, we discuss the precision of iterative policy learning and propose a novel technique for measuring the similarity of two trajectories based on projection.

The paper is organized into five sections. In the next section, we briefly review the procedures for physical Human-Robot Cooperation (pHRC) based on stiffness learning. To increase the accuracy of the learned policy during pHRC, we propose to apply the trajectory similarity measures explained in Section III. Section IV describes the experimental verification of our framework for learning a cooperative assembly task. Finally, we conclude with summary and a critical discussion of the proposed framework in Section V.

II. Policy adaptation during human-robot cooperation

We study physical human-robot cooperation, where both the robot and the human operator firmly hold and manipulate an object. Examples of such collaboration are numerous and can be found in civil engineering, production plants, and also in everyday life. In most cases, the problem is related to the transportation of bulky, heavy, or very long and possibly flexible objects. Our approach aims at allowing the initial demonstration to evolve naturally into a collaborative task,

All authors are with Humanoid and Cognitive Robotics Lab, Department of Automatics, Biocybernetics and Robotics, Jožef Stefan Institute, Ljubljana, Slovenia, e-mail: {mihael.simonic, miha.denisa, ales.ude, bojan.nemec}@ijs.si

where the operator can gradually change the spatial and/or temporal part of the trajectory at any time. The operator can suspend the execution of the task simply by no longer following the robot's movement and restart the task by following the robot again. This feature dictates the dynamic sharing of the leader-follower role. Being able to adapt robot stiffness is the key technology to achieve this behavior.

The collaborative task is given in a common Cartesian coordinate system and parametrized with Speed scaled Cartesian Dynamic Movement Primitives (SCDMP). Each position and orientation degree of freedom is governed by a unique dynamical system and synchronized with a common phase. The mathematical formulation of SCDMP can be found in [16], [15]. Let $\mathbf{p}(s) \in \mathbb{R}^3$ and $\mathbf{q}(s) \in \mathbb{R}^4$ represent the desired robot position vector and orientation quaternion, calculated by integrating the SCDMP,

$$\begin{bmatrix} \mathbf{p}(s) \\ \mathbf{q}(s) \end{bmatrix} = \text{integ}(\texttt{SCDMP}, \mathtt{s}, \nu), \tag{1}$$

where SCDMP denotes a set of SCDMP parameters, $s \in (0, 1]$ is the phase variable and $\nu \in [0, \infty)$ the speed scaling factor. The integration procedure as described in [16] is denoted by integ. Initially, the robot is in gravity compensation mode and compliant in all positional and rotational degrees of freedom. This means that the operator can freely move the shared object and the robot by pushing it in the desired direction. The learning system records the robot's positions and orientations and calculates the SCDMP parameters. Thus, the robot obtains an approximate trajectory for the given task without stiffness parameters. In subsequent repetition(s) of the same task, the appropriate stiffness parameters are calculated so that the robot can take on the leader role. An important feature of SCDMP is its ability to non-uniformly modulate the task's velocity profile [14]. The speed scaling factor $\nu(s)$ provides us with the ability to continuously change the execution speed from zero to any speed, where s denotes the phase variable. In the HRC scheme, this scaling factor consists of two parts:

$$\nu(s) = \nu_c(s) + \nu_h(s), \tag{2}$$

where $\nu_c(s)$ denotes the commanded speed scaling factors, which is learnt, and $\nu_h(s)$ the speed scaling factor that arises from the collaboration. The learned speed scaling factor is calculated as a weighted sum of radial basis functions (RBFs) for the given phase s [14],

$$\nu_c(s) = 1 + \frac{\sum_{i=1}^{M} w_i^{\nu} \Psi_i(s)}{\sum_{i=1}^{M} \Psi_i(s)}, \ \Psi_i(s) = \exp\left(-h_i \left(s - c_i\right)^2\right), \tag{3}$$

$w_i^{\nu} \in \mathbb{R}$ are the weights, and $c_i, h_i \in \mathbb{R}$ specify the centers and the widths of RBFs, respectively. Initial values of the weights w_i^{ν} are set to zero, meaning that the speed scaling factor is equal to one. New weights are calculated after each task cycle as explained in Section II-B. The second part of the scaling factor, ν_h, is calculated from forces and torques

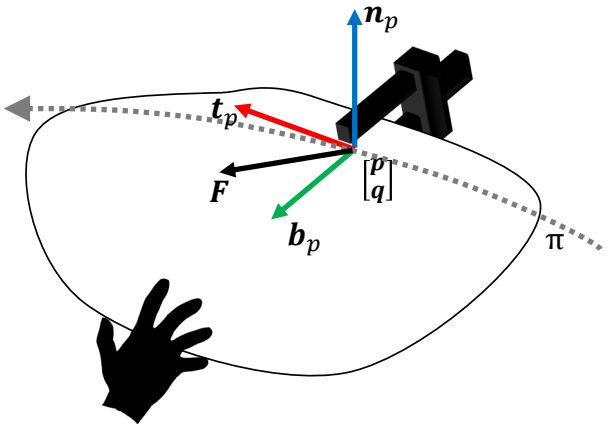

Fig. 1. The spatial part of collaborative policy π describes the motion of the robot. Vector $\mathbf{F}$ denotes forces exerted by the human co-worker expressed in the robot coordinate system. The Frenet-Serret frame is specified by the tangent $\mathbf{t}_p$, normal $\mathbf{n}_p$ and binormal $\mathbf{b}_p$. Current robot pose consists of a positional vector $\mathbf{p}$ and a quaternion $\mathbf{q}$ defining orientation.

applied by the human operator

$$\nu_h(s) = \nu_p \text{sigm}(\|\mathbf{F}\|)\mathbf{F} \cdot \mathbf{t}_p(s) + \nu_o \text{sigm}(\|\mathbf{M}\|)\mathbf{M} \cdot \mathbf{t}_r(s), \tag{4}$$

where $\cdot$ denotes the dot product and ν_p, ν_o are positive scalars used to scale the velocities of the translational and rotational motion along the desired path, $\mathbf{t}_p,\ \mathbf{t}_r \in \mathbb{R}^3$ are the tangential vectors to the robot path and $\mathbf{F}, \mathbf{M}$ are the operator's forces and torques exerted to the robot tool. $\text{sigm}(\cdot)$ denotes a sigmoid function given by $\text{sigm}(x) = 1/(1 + e^{-ax})$. It ensures a smooth start and stop when pushing along the policy and effectively deals with the jittery robot motion due to the noisy input from the force sensor. The constant scalar a specifies the smoothness/speed of the transition.

As explained above, robot learning should gradually evolve into human-robot collaboration. Initially, the robot is fully compliant. Assuming that the object's geometry and dynamic parameters are known, the robot can firmly hold the object and compensate for its gravity. Initially, the robot system does not know anything about the task. The trajectory to accomplish the desired task is learnt during the collaboration. The spatial and temporal parts of the task are learnt separately. During the collaboration, the robot system learns which part of the task should be executed with high precision and adapt its stiffness accordingly. For safety reasons or to synchronize it with an external event, the human operator can change the temporal part of the task during the collaboration using a mechanism given in Section II-B. In this way, the human operator can stop the robot and take the lead over the execution of the collaborative task and pass it back to the robot when appropriate.

The policy dictating robot's behaviour is composed from spatial and temporal parts of the motion trajectory and the corresponding stiffness parameters. In the following sections, a detailed description for how to update the policy is provided.

A. Learning of the spatial part of the trajectory

In our learning scheme, a new trajectory is learnt by specifying an offset $\tilde{\mathbf{p}},\ \tilde{\mathbf{q}}$ to the initially demonstrated trajectory $\mathbf{p}_0,\ \mathbf{q}_0$. The initially demonstrated trajectory is encoded with an SCDMP, while the offset to the trajectory are represented as a linear combination of RBFs [15]. In this way, we eliminate the influence of the DMP integration noise, which would otherwise accumulate during the iterative policy improvement.

During the collaborative task execution, we update the spatial path of robot trajectories using the following set of formulas

$$\begin{aligned} \mathbf{p}_l(s) &= \mathbf{p}_0(s) + \tilde{\mathbf{p}}_l(s) && (5)\\ \tilde{\mathbf{p}}_{l+1}(s) &= \zeta_p \Delta\mathbf{p}(s) + \tilde{\mathbf{p}}_l(s), && (6)\\ \Delta\mathbf{p}(s) &= \mathbf{p}_m - \mathbf{p}_l(s), && (7) \end{aligned}$$

where $\mathbf{p}_l(s)$ denotes the robot's position on the learnt trajectory at phase s. $\mathbf{p}_0$ denotes the originally demonstrated trajectory and $\tilde{\mathbf{p}}_l(s)$ the learned offset at phase s from the point on the original trajectory at the same phase. $\mathbf{p}_m$ denotes the measured position of the robot. $\mathbf{p}_l$ is used as a reference trajectory to control the robot in the next cycle. Additionally, factor ζ_p defines the learning rate. By setting $\zeta_p = 1$, the updated trajectory offset $\tilde{\mathbf{p}}_{l+1}$ is equal to the measured trajectory offset in the current learning cycle $\tilde{\mathbf{p}}_m$. On the other hand, by setting $\zeta_p = 0$, the trajectory offset $\tilde{\mathbf{p}}_{l+1}$ does not change and the system stops learning. After each learning cycle, the updated trajectory offset $\tilde{\mathbf{p}}_{l+1}$ is encoded as a linear combination of radial basis functions (RBF) [15].

For learning of orientational part of the trajectory we apply the following update rule for quaternions

$$\begin{aligned} \mathbf{q}_l(s) &= \tilde{\mathbf{q}}_l(s) * \mathbf{q}_0(s) && (8)\\ \tilde{\mathbf{q}}_{l+1}(s) &= \exp\left(\zeta_o \frac{\boldsymbol{\omega}(s)}{2}\right) * \tilde{\mathbf{q}}_l(s), && (9)\\ \boldsymbol{\omega}(s) &= 2\log(\mathbf{q}_m * \overline{\mathbf{q}}_l(s)), && (10) \end{aligned}$$

Symbol $*$ denotes quaternion multiplication, $\overline{\mathbf{q}}$ denotes conjugate quaternion, and $\mathbf{q}_m$ is the measured robot orientation. log and exp are quaternion logarithm and exponential maps form $S^3 \mapsto \mathbb{R}^3$ and vice versa, respectively [16].

B. Learning of the temporal part of the trajectory

Similar to positions and orientations, we also learn the speed profile of a task with respect to speed modulation in each human-robot collaboration cycle.

$$\begin{aligned} \nu_{c,l+1}(s) &= \zeta_\nu \delta\nu(s) + \nu_{c,l}(s) && (11)\\ \delta\nu(s) &= \nu_l(s) - \nu_{l-1}(s) + \nu_{h,l} && (12) \end{aligned}$$

Signal ν_c is encoded as a linear combination of RBFs as explained in Eq. (3). ζ_ν is the learning rate similarly as in Eq. (6). While performing a collaborative task, the operator may stop the robot by applying a sufficiently large force in the opposite direction of the tangent to the task. The value of this force is determined by equations (2) and (4). In such a case, however, we stop updating the speed scaling factor with the equation (11), because we would distort the speed profile of the task too much.

C. Stiffness learning

Working with a non-compliant robot can be uncomfortable and even potentially dangerous for the operator. On the other hand, the robot must be stiff where high accuracy is required [17], [11]. Here we build on the idea of decreasing the stiffness in the parts of the trajectory with higher variability and vice versa. The variability is determined by observing the task executions over successive repetitions and calculating the phase-dependent variance. Concurrently with Eqs. (5–7), we calculate the positional covariance

$$\boldsymbol{\Sigma}_{p,l+1}(s) = (1-\zeta_p)(\boldsymbol{\Sigma}_{p,l}(s) + \zeta_p \Delta\mathbf{p}(s)\Delta\mathbf{p}(s)^{\mathrm{T}}). \quad (13)$$

We set the initial value for covariance matrix $\boldsymbol{\Sigma}_{p,1}$ to zero. Similarly, the update rule for covariance of speed factors ν is given as

$$\Sigma_{\nu,l+1}(s) = (1-\zeta_\nu)(\Sigma_{\nu,l}(s) + \zeta_\nu \nu^2(s)). \quad (14)$$

The covariance matrices elements are encoded with RBFs, just like position offsets and velocity scale factor. The initial value of Σ_ν is zero. Note that, we set the stiffness in the tangential direction with respect to how precisely the robot should follow the learned speed profile.

However, this variance has to be expressed in an appropriate coordinate system. Let $\mathbf{R}_p$ denote the rotation matrix attached to the object's trajectory π with x coordinate specified in the desired direction of motion and the other two coordinates orthogonal to it, as illustrated in Fig. 1. A suitable choice for such a rotation matrix is by calculation the Frenet-Serret frame [18] at each sampling time. The Frenet-Serret (FS) frame consists of three orthogonal directions defined by the path's tangent $\mathbf{t}_p$ (direction of motion), normal $\mathbf{n}_p$, and binormal $\mathbf{b}_p$. We obtain the following expression for $\mathbf{R}_p$

$$\begin{aligned} \mathbf{R}_p(s) &= [\mathbf{t}_p(s)\ \mathbf{n}_p(s)\ \mathbf{b}_p(s)], && (15)\\ \mathbf{t}_p(s) &= \frac{\dot{\mathbf{p}}_l(s)}{\|\dot{\mathbf{p}}_l(s)\|}, && (16)\\ \mathbf{b}_p(s) &= \frac{\dot{\mathbf{p}}_l(s) \times \ddot{\mathbf{p}}_l(s)}{\|\dot{\mathbf{p}}_l(s) \times \ddot{\mathbf{p}}_l(s)\|}, && (17)\\ \mathbf{n}_p(s) &= \mathbf{b}_p(s) \times \mathbf{t}_p(s) && (18) \end{aligned}$$

Note that the velocities $\dot{\mathbf{p}}_l$ and accelerations $\ddot{\mathbf{p}}_l$ are provided by DMP (or RBF) integration at every phase s, which ensures smoothness. If the denominator in Eq. (16) gets close to zero, i.e. the motion stops, we suspend the updating of $\mathbf{R}_p$. Updating resumes when the motion becomes faster again. A similar approach is taken in Eq. (17), when the updating is resumed when the motion is no longer a straight line.

As mentioned above, the variances computed with (13) need to be expressed in the trajectory coordinate system given with FS frame. The transformation from the robot base coordinate system to the trajectory coordinate system is done by

$$\boldsymbol{\Sigma}(s) = \mathbf{R}_p(s)\boldsymbol{\Sigma}_{p,l}(s)\mathbf{R}_p(s)^{\mathrm{T}}. \quad (19)$$

We set the positional stiffness according to speed variation in the tangential direction of the motion ($\mathbf{t}_p$ in Fig. 1) and positional variation in the remaining directions ($\mathbf{b}_p, \mathbf{n}_p$ in Fig. 1). To set the stiffness according to the policy positional variation, we calculate the robot stiffness at each sampling time as

$$\mathbf{K}_p(s) = \mathbf{R}_p(s)^{\mathrm{T}} \mathbf{K}'_p(s) \mathbf{R}_p(s), \tag{20}$$

$$\mathbf{K}'_p(s) = \begin{bmatrix} \frac{k_\nu}{\Sigma_\nu(s)+\epsilon_\nu} & 0 & 0 \\ 0 & \frac{k_o}{\Sigma_{yy}(s)+\epsilon_0} & 0 \\ 0 & 0 & \frac{k_o}{\Sigma_{zz}(s)+\epsilon_0} \end{bmatrix},$$

where k_ν, k_o, ϵ_ν and ϵ_0 are empirically chosen positive constants, which set the upper bound for the controller gain. Σ_{yy} and Σ_{zz} are the second and the third diagonal coefficient of $\mathbf{\Sigma}$, respectively.

In a similar manner, we can calculate the rotational stiffness using the following update rule for orientational covariance:

$$\mathbf{\Sigma}_{q,l+1}(s) = (1-\zeta_o)(\mathbf{\Sigma}_{q,l}(s) + \zeta_o \boldsymbol{\omega}_l(s) \boldsymbol{\omega}_l(s)^{\mathrm{T}}). \tag{21}$$

For calculation of the rotational stiffness, we need to express the FS frame for the rotation part of the trajectory. Note that the tangential direction of the FS frame coincides with the normalized angular velocity $\boldsymbol{\omega}$. Note also that when changing stiffness matrix, it is necessary to adjust also the controller damping matrices of the underlying robot controller.

III. Policy Learning Using Trajectory Similarity Measure

Whenever a human co-worker needs to modify a collaborative task, it exerts a certain force on the shared object. It thus modifies both the temporal and spatial part of the trajectory. The learning is governed by the position and orientation displacement, calculated by Eqs. (7) and (10). In the ideal conditions, displacement can occur only in the normal and bi-normal coordinates of the FS frame, as any force in the tangential direction causes a change in the robot phase s. In practice, however, there is always a certain error between the commanded pose and measured pose in the tangential direction. This error is mainly caused by the robot's friction and variable controller gains, which reduces the precision of the trajectory adjustment. The situation is shown in Fig. 2.

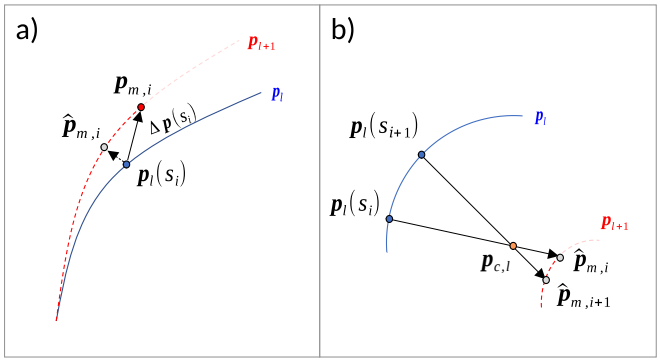

Fig. 2. Two examples of matching phases between two consecutive learning cycles. The trajectory $\mathbf{p}_l$ learned in the previous cycle l is shown with blue curve. The trajectory $\mathbf{p}_{l+1}$ being sampled in the current learning cycle $l+1$ is shown with dotted red curve. Point $\mathbf{p}_l(s_i)$ shows the learned position of the robot at phase s_i in the previous cycle and is taken as the reference position. The currently measured point $\mathbf{p}_{m,i}$ differs from the reference both due to human interaction and non-ideal robot tracking. To eliminate the later, we project the reference point $\mathbf{p}_l(s_i)$ to the known part of trajectory $\mathbf{p}_{l+1}$ and obtain $\hat{\mathbf{p}}_{m,i}$ as shown on the left. In certain cases (e.g. when the centre of curvature $\mathbf{p}_c$ lies between current and previous trajectory), the projection can yield an unwanted result, i.e. $\hat{\mathbf{p}}_{m,i+1}$ is projected behind $\hat{\mathbf{p}}_{m,i}$ as shown in the right part.

One solution is to project the commanded position and orientation at phase s from the previous cycle l to the currently sampled trajectory $\mathbf{p}_{l+1}$, as illustrated in Fig. 2a. However, pure projection is not always appropriate. For example, if the curvature of the trajectory segment is high[1], the projection $\hat{\mathbf{p}}_m$ changes the direction, creating unwanted patterns (see Fig. 2b). In this case, we consider the projected point from the previous time sample as long as the projection follows the direction of $\mathbf{p}_l$. In simple words, the projected trajectory should never go back. To calculate the projected point $\hat{\mathbf{p}}_m = \mathbf{p}_m(k_i)$ we solve the following optimization problem

$$\begin{aligned} \arg\min_{k_i} \; & |(\mathbf{p}_l(s) - \mathbf{p}_{m,k_i}) \cdot \mathbf{t}_p| + |2\log(\mathbf{q}_{m,k_i} * \overline{\mathbf{q}}_l(s)) \cdot \mathbf{t}_r| \\ s.t. \quad & k_i \geq k_{i-1} \end{aligned} \tag{22}$$

and calculate the displacement (7) and (10) as $\Delta\mathbf{p}(s) = \mathbf{p}_{m,k_i} - \mathbf{p}_l(s)$ and $\boldsymbol{\omega}(s) = 2\log(\mathbf{q}_{m,k_i} * \overline{\mathbf{q}}_l(s))$.

The above problem can also be viewed as determining the similarity between two trajectories. Different similarity measures between two series try to minimize the cumulative distance between nodes of compared trajectories, refereed as distance measure. Our proposed method has some similarities with the Fréchet Distance (FD) [19], as in both algorithms, two individual agents traveling along the compared trajectories are never allowed to move back. Our condition that the individual points of both trajectories are connected by vectors orthogonal to the tangents has a solid physical background. Namely, the proposed learning scheme, where the robot is moved by pushing in the tangential direction, negates the influence of robot tracking error in this direction.

IV. Experimental Evaluation

We evaluated the proposed pHRC scheme to realize a collaborative task where a robot and a human together insert a long metal rod into a groove (see Fig. 3). High precision is required to accomplish such a task. To perform the task, the robot firmly grasps one side of the rod and the human co-worker the other side. The position tolerances are minimal at the beginning and the end of the movement whereas they are more relaxed in between. This task mimics the elementary operations required for collaborative assembly.

[1] i.e. the center of curvature lies between the edited trajectory $\mathbf{p}_l$ and the updated $\mathbf{p}_{l+1}$

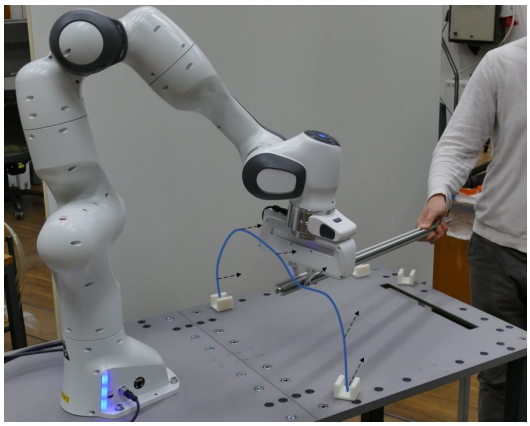

Fig. 3. The collaborative task is to move the metal rod from the initial location to the final one, where small tolerances are necessary to accomplish the task.

Our first concern was how the learning algorithm affects the precision of the task execution. The learning algorithm in all experiments was the same as described in Section II, but we applied different approaches to calculate exponential moving average given by Eqs. (7) and (10). We compared the results obtained with averaging robot trajectories using a) commanded and actual robot position b) commanded and projected actual position, as proposed in previous section and c) commanded and actual position, where the trajectory indices were obtained with well known similarity measure, Dynamic Time Warping (DTW) [20]. The results for one learning cycle are shown in Fig. 4.

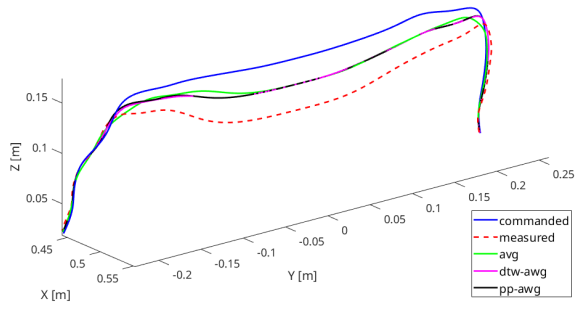

Fig. 4. Average trajectory calculation using different algorithms. Avg (green) denotes Euclidian average from commanded and actual robot position, dtw-avg (magenta) averaging using indices obtained with DTW and pp-avg (black) averaging using indices obtained with the proposed projection algorithm.

As evident from the results, DTW and projection based average calculation both returns comparable results, while simple average deviates, resulting in inaccurate learning. An additional indicator of the correctness of calculating the average is the distance measure [19]. For the given case, the distance measure for Euclidean averaging was 29.86, for DTW 14.76, and for projection algorithm 16.03. In practice, the results obtained using DTW and the projection algorithm are very similar and comparable. The advantage of the projection algorithm is that it is less computationally demanding and does not require pre-known trajectories.

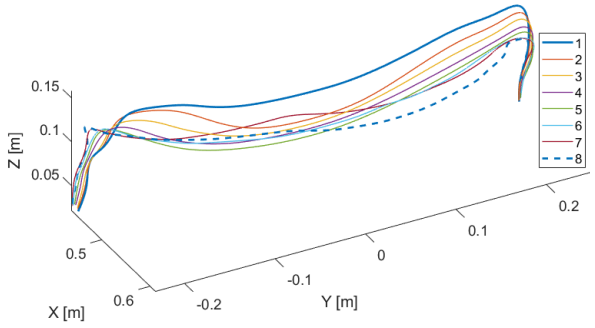

Fig. 5. Positional part of eight repetitions of collaborative policy. The final policy is denoted with a dashed line. Note that, trajectory labelled as 1 refers to the first collaborative cycle after the initial demonstration.

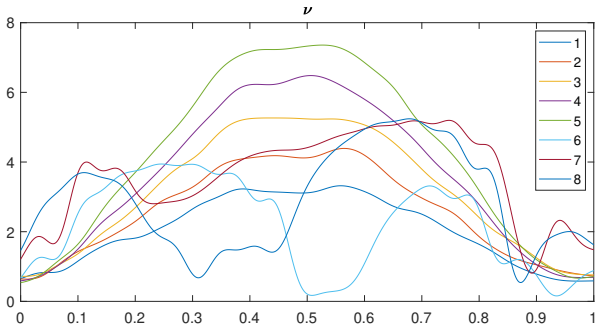

Fig. 6. Speed scaling factor ν of eight repetitions of collaborative policy with respect to the normalized trajectory length.

Thus, one can calculate the average on the fly during human-robot interaction.

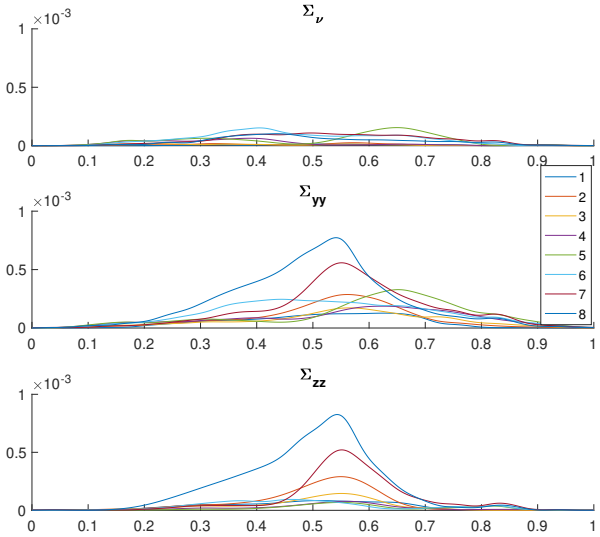

Fig. 7. Positional part of variances of eight repetitions of collaborative policy. Note that all graphs are scaled to the normalized trajectory length.

Next, we evaluated the learning of stiffness and velocity profiles. The robot was initially fully compliant. Then, the human co-worker moved the rod from the initial to the final position at low speed by exerting necessary forces and torques to the robot. As a result, the robot learned the task

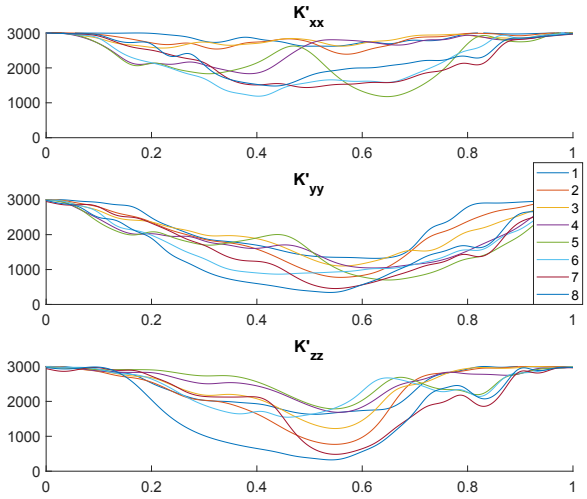

Fig. 8. Learned controller gains in eight repetitions of collaborative policy.

and adjusted its stiffness. Figs. 5 and 6 show the positional and temporal part of the eight repetitions of the task. The estimated variances were very low at the beginning and end of the task, where the tolerances needed to successfully insert the rod into grooves were also very low (see Fig. 7). As a result, the robot learned high stiffness, which prevented deviations in these critical parts of the path. In contrast, the deviation in the middle part of the trajectory, where the robot only had to avoid an obstacle, and the movement was not precisely determined, could have been greater. Therefore, the robot learned lower stiffness on this part of the path, as evident from Fig. 8.

In the first five repetitions, the operator constantly increased the execution speed, which is also shown by the results in Fig. 6. Thus, the robot also learned the new velocity profile. Regardless, the human co-worker was able to stop the movement at any time by pushing the stick in the opposite direction of the tangent. We demonstrate this property in the sixth to eighth cycle, where the human operator randomly modified the task's speed by pushing the rod in the tangential direction of the task's trajectory.

V. Conclusions

In the paper, we presented an improved approach to human-robot physical collaboration, where high precision is essential, such as for example in assembly tasks. While this approach is based on our previously proposed algorithm [11], we developed a better speed learning method and a more accurate learning of trajectories [15]. The major contribution is the novel distance measure for time series based on projecting series points to the related trajectory. The experimental evaluation showed that the proposed distance measure performs equally well in our human-robot collaboration schemes as the well established DTW. An important benefit of the proposed scheme is that our new distance measure can be calculated on the fly during human-robot interaction and does not require gathering of the entire trajectory. We evaluated the overall scheme in an HRC task, where high precision is required only in some parts of the task.

Acknowledgment

The research leading to these results has received funding from the Horizon 2020 RIA Programme grant no. 820767, CoLLaboratE.

References

[1] K. Demir, G. Döven, and B. Sezen, "Industry 5.0 and human-robot co-working," *Procedia Computer Science*, vol. 158, pp. 688–695, 2019.

[2] R. Dillmann, "Teaching and learning of robot tasks via observation of human performance," *Robotics and Autonomous Systems*, vol. 47, no. 2-3, pp. 109–116, 2004.

[3] S. Calinon, Z. Li, T. Alizadeh, N. G. Tsagarakis, and D. G. Caldwell, "Statistical dynamical systems for skills acquisition in humanoids," in *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, Osaka, Japan, 2012, pp. 323–329.

[4] S. M. Khansari-Zadeh and A. Billard, "Learning stable nonlinear dynamical systems with gaussian mixture models," *IEEE Transactions on Robotics*, vol. 27, no. 5, pp. 943–957, Oct 2011.

[5] S. Calinon, D. Bruno, and D. G. Caldwell, "A task-parameterized probabilistic model with minimal intervention control," in *IEEE International Conference on Robotics and Automation (ICRA)*, Hong Kong, 2014, pp. 3339–3344.

[6] G. Raiola, X. Lamy, and F. Stulp, "Co-manipulation with multiple probabilistic virtual guides," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sept 2015, pp. 7–13.

[7] L. Rozo, S. Calinon, D. G. Caldwell, P. Jiménez, and C. Torras, "Learning physical collaborative robot behaviors from human demonstrations," *IEEE Transactions on Robotics*, vol. 32, no. 3, pp. 513–527, June 2016.

[8] M. Ewerton, G. Neumann, R. Lioutikov, H. B. Amor, J. Peters, and G. Maeda, "Learning multiple collaborative tasks with a mixture of interaction primitives," in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2015, pp. 1535–1542.

[9] A. Paraschos, C. Daniel, J. Peters, and G. Neumann, "Probabilistic movement primitives," in *Neural Information Processing Systems 26*, 2013, pp. 2616–2624.

[10] A. Gams, B. Nemec, A. J. Ijspeert, and A. Ude, "Coupling movement primitives: Interaction with the environment and bimanual tasks," *IEEE Transactions on Robotics*, vol. 30, no. 4, pp. 816–830, Aug 2014.

[11] B. Nemec, N. Likar, A. Gams, and A. Ude, "Human robot cooperation with compliance adaptation along the motion trajectory," *Autonomous Robots*, vol. 42, no. 5, pp. 1023–1035, 2018.

[12] H. B. Amor, G. Neumann, S. Kamthe, O. Kroemer, and J. Peters, "Interaction primitives for human-robot cooperation tasks," in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2014, pp. 2831–2837.

[13] B. Nemec, A. Gams, and A. Ude, "Velocity adaptation for self-improvement of skills learned from user demonstrations," in *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, Atlanta, USA, 2013, pp. 423–428.

[14] R. Vuga, B. Nemec, and A. Ude, "Speed adaptation for self-improvement of skills learned from user demonstrations," *Robotica*, vol. 34, no. 12, pp. 2806–2822, 2016.

[15] M. Simonič, T. Petrič, A. Ude, and B. Nemec, "Analysis of methods for incremental policy refinement by kinesthetic guidance," *Journal of Intelligent & Robotic Systems*, vol. 102, no. 5, 2021.

[16] A. Ude, B. Nemec, T. Petrič, and J. Morimoto, "Orientation in cartesian space dynamic movement primitives," in *IEEE International Conference on Robotics and Automation (ICRA)*, Hong Kong, China, 2014, pp. 2997–3004.

[17] S. Calinon, I. Sardellitti, and D. G. Caldwell, "Learning-based control strategy for safe human-robot interaction exploiting task and robot redundancies," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 249–254, 2010.

[18] R. Ravani and A. Meghdari, "Velocity distribution profile for robot arm motion using rational Frenet-Serret curves," *Informatica*, vol. 17, no. 1, pp. 69–84, 2006.

[19] H. Su, S. Liu, B. Zheng, X. Zhou, and K. Zheng, "A survey of trajectory distance measures and performance evaluation," *The VLDB Journal*, no. 29, pp. 3–32, 2020.

[20] P. Senin, "Dynamic time warping algorithm review," in *University of Hawaii at Manoa, Technical report series*, 2008.