Noname manuscript No.
(will be inserted by the editor)

Human Robot Cooperation with Compliance Adaptation along the Motion Trajectory

Bojan Nemec, Nejc Likar, Andrej Gams, Aleš Ude

the date of receipt and acceptance should be inserted later

Abstract In this paper we propose a novel approach for intuitive and natural physical human-robot interaction in cooperative tasks. Through initial learning by demonstration, robot behavior naturally evolves into a cooperative task, where the human co-worker is allowed to modify both the spatial course of motion as well as the speed of execution at any stage. The main feature of the proposed adaptation scheme is that the robot adjusts its stiffness in path operational space, defined with a Frenet-Serret frame. Furthermore, the required dynamic capabilities of the robot are obtained by decoupling the robot dynamics in operational space, which is attached to the desired trajectory. Speed-scaled dynamic motion primitives are applied for the underlying task representation. The combination allows a human co-worker in a cooperative task to be less precise in parts of the task that require high precision, as the precision aspect is learned and provided by the robot. The user can also freely change the speed and/or the trajectory by simply applying force to the robot. The proposed scheme was experimentally validated on three illustrative tasks. The first task demonstrates novel two-stage learning by demonstration, where the spatial part of the trajectory is demonstrated independently from the velocity part. The second task shows how parts of the trajectory can be rapidly and significantly changed in one execution. The final experiment shows two Kuka LWR-4 robots in a bi-manual setting cooperating with a human while carrying an object.

Humanoid & Cognitive Robotics Lab, Department of Automatics, Biocybernetics and Robotics, Jožef Stefan Institute, Jamova 39, 1000 Ljubljana, Slovenia, `(bojan.nemec, nejc.likar, andrej.gams,ales.ude)@ijs.si`

Address(es) of author(s) should be given

1 Introduction

An important aspect for future use of robots in our home environments as well as in production plants is their ability to cooperate with humans. Robots dominate over human capabilities in precision and efficiency while performing repetitive and monotonous tasks, while human are unbeaten in adaptation to new situations and upcoming problems (Faber et al, 2015). Joining both worlds by means of direct human-robot cooperation brings new advantages and potentially solves many open problems in robotics.

Cooperative task execution in physical human-robot interaction can be classified based on the level of control the robot assumes (Adorno et al, 2011). Most commonly, the human and the robot are in master-slave control mode, with the operator – master – retaining complete control over the evolution of the cooperative task. The interaction is provided through force or visual feedback (Evrard et al, 2009). Alternatively, the task can be controlled by the robot and only initiated by the human (Soyama et al, 2004). In some applications, e. g., in rehabilitation robotics (Krebs et al, 1998), the control over the task evolution is dynamically shared between the human and the robot (Mortl et al, 2012). The level of control can also change based on the current situation. For example, when the human is transferring knowledge to the robot, the robot will often be controlled differently than during the actual execution. The behavior of the robot might also be personalized to suit each coworker perfectly. The learning of robot behavior is thus crucial for effective cooperative task execution.

However, robotic learning can be applied to various aspects of the task; for example, the position, the velocity, the level of adaptation and/or autonomy, etc.

Furthermore, different feedback options are available through visual, haptic, or direct physical interaction Gams et al (2016). Finally, many of these aspects and conditions need to be combined in a single, preferably intuitive, system that allows interaction similar to that between two humans.

1.1 Problem statement

In this paper we investigate the learning of robot behavior during human-robot cooperative tasks. Cooperative task control should allow natural learning and adaptive execution. Therefore, it must:

- provide physical human-robot interaction,
- allow non-uniform changes of execution speeds,
- enable adaptation of a trajectory during its execution, without the need to re-plan the whole task when a new situation arises,
- provide a certain degree of cooperative intelligence, i. e., it should be compliant when accuracy is not needed, but stiff when it is needed,
- it should be applicable to both single arm and bimanual human-robot cooperation

pHRI has been heavily investigated in the past (Evrard et al, 2009; Soyama et al, 2004; Krebs et al, 1998; Calinon et al, 2010), including for bimanual robot operations (Adorno et al, 2010; Mortl et al, 2012; Park and Lee, 2015). Several papers explore sub-aspects of the stated problem. For example, in a recent paper Ramacciotti et al. (Ramacciotti et al, 2016) explore shared control for motion speed and trajectory adaptation. However, to the best of our knowledge, a complete approach that fulfills the given problem statement, has not been proposed yet.

In this paper we propose a control architecture, which fulfills the above problem statement. Throughout execution, the robot constantly learns from the interaction with the human. First, during the initial learning of the task, the control is completely handled by the human operator. However, during the task repetition, the task is analyzed and the robot gradually takes control over the parts with low variance of executed trajectories. The human can at any time take back the control over the task execution by again increasing the variance through physical interaction.

Initial idea at such a framework was published in (Nemec et al, 2016a). In this paper we further extend the approach with a)passivity based control framework for HRI scheme; b)improved speed adaptation scheme; c)additional experiments, that prove the validity of the proposed concept.

1.2 Related work

The implementation of a pHRI scheme depends primarily on the underlaying policy representation, which determines also subsequent methods for learning by demonstration, calculation of the variability distribution and implementation of non-uniform speed changes. A motor skill necessary to accomplish the given task does not only comprehend the path that the robot should follow, but also the variation of coordination patterns during the movement Calinon et al (2012). A well known paradigm to cope with such requirements is to encode the task as a dynamical system. Khansari-Zadeh&Billard (Khansari-Zadeh and Billard, 2011) have introduced a method of encoding motion as a nonlinear autonomous Dynamical System (DS) and sufficient conditions to ensure global asymptotic stability at the target. The method uses several demonstrations and ensures that all motions follow closely the demonstrations while ultimately reaching and stopping at the target. It was expanded also for fast motions, for example for catching objects (Salehian et al, 2016). The method relies on Gaussian mixture model (GMM) representation, which was also used by Calinon et al. (Calinon et al, 2014). It sequentially superimposes dynamical systems with varying full stiffness matrices. The method has been extensively applied, for example also for virtual guides, where the robot is compliant only in the direction of the trajectory (Raiola et al, 2015), and for learning of physical collaborative human-robot actions (Rozo et al, 2016). Other approaches have been proposed for both interactive tasks and for learning actions from several demonstrations. Mixture of interaction primitives has also been proposed (Ewerton et al, 2015). An example of learning from several demonstrations are the probabilistic motion primitives (ProMP) (Paraschos et al, 2013). These enable the encoding of stochastic system behavior. Modified DMPs to include coupling terms of different kinds have also been proposed for interaction with humans in single-arm and bimanual settings (Gams et al, 2014). Although these representations allow speed scaling, non-uniform speed scaling in its original form is not possible by either. Recently, interaction tasks have been discussed in a variation of motor primitives called interaction primitives (Amor et al, 2014), which maintain a distribution over the DMP parameters and synchronizes phase with external agents (e.g. humans) by dynamic time warping. In our approach we rely on the framework of dynamic motion primitives (DMP) (Ijspeert et al, 2013) and its extension to cope with non-uniform speed changes (Nemec et al, 2013).

This paper is organized as follows. Section 2 outlines the framework of dynamic movement primitives along with the speed profile encoding extension. The novelty of the paper is described in Section 3, which combines the separate sub-aspects into a complete algorithm. Applications of the proposed approach to Learning by Demonstration (LbD) and to bimanual physical human-robot cooperation are presented in section 4. A discussion concludes the paper.

2 Learning by demonstration for Human-Robot Cooperation scheme

In our work we rely on motion representation with dynamic motion primitives (DMPs) (Ijspeert et al, 2013), extended for Cartesian space movements (Ude et al, 2014). These are used to encode the demonstrated cooperative human-robot task. Kinesthetic guiding can be used to capture the desired robot motion. The original DMP formulation does not provide the means to variate the speed of movement in a non-uniform way without changing the course of movement. However, in our approach we *do* need to apply non-uniform speed changes, prompting the requirement for appropriate trajectory representation. A suitable representation is Speed-Scaled Dynamic Motion Primitives (SS-DMPs), which we originally proposed in (Nemec et al, 2013).

Through kinesthetic guiding we first acquire the initial movement policy in Cartesian coordinates

$$\mathcal{G} = \{\mathbf{p}_k, \mathbf{q}_k, \dot{\mathbf{p}}_k, \boldsymbol{\omega}_k, \ddot{\mathbf{p}}_k, \dot{\boldsymbol{\omega}}_k, t_k\}_{k=1}^{\mathrm{T}}. \tag{1}$$

$\mathbf{p}_k \in \mathbb{R}^3$ are the positions, while $\mathbf{q}_k \in \mathrm{S}^3$ are the unit quaternions describing orientation, with S^3 denoting a unit sphere in $\mathbb{R}^4$. Besides the positions and orientations, we also record the position and orientation velocities ($\dot{\mathbf{p}}_k$, $\boldsymbol{\omega}_k$) and accelerations ($\ddot{\mathbf{p}}_k$,$\dot{\boldsymbol{\omega}}_k$). k are trajectory samples, and T is the number of samples.

We parameterize this demonstrated policy with a nonlinear dynamical system that enables the encoding of general trajectories (Ijspeert et al, 2013; Nemec et al, 2013; Ude et al, 2014). The trajectory can be specified by the following system of nonlinear differential equations for positions $\mathbf{p}$ and orientations $\mathbf{q}$

$$\nu(s)\tau\dot{\mathbf{z}} = \alpha_z(\beta_z(\mathbf{g}_p - \mathbf{p}) - \mathbf{z} + \mathbf{f}_p(s), \tag{2}$$

$$\nu(s)\tau\dot{\mathbf{p}} = \mathbf{z}, \tag{3}$$

$$\nu(s)\tau\dot{\boldsymbol{\eta}} = \alpha_z\left(\beta_z 2\log\left(\mathbf{g}_o * \bar{\mathbf{q}}\right) - \boldsymbol{\eta}\right) + \mathbf{f}_o(s), \tag{4}$$

$$\nu(s)\tau\dot{\mathbf{q}} = \frac{1}{2}\boldsymbol{\eta} * \mathbf{q}, \tag{5}$$

$$\nu(s)\tau\dot{s} = -\alpha_s s. \tag{6}$$

Here s denotes the phase and $\mathbf{z}$ and $\boldsymbol{\eta}$ are auxiliary variables. The above system (2) – (6) converges to the unique equilibrium point at $\mathbf{p} = \mathbf{g}_p$, $\mathbf{z} = 0$, $\mathbf{q} = \mathbf{g}_o$, $\boldsymbol{\eta} = 0$, and $s = 0$. Asterisk $*$ denotes quaternion multiplication and $\bar{\mathbf{q}}$ quaternion conjugation. Eq. (15) provides the definition of quaternion logarithm. The nonlinear forcing terms $\mathbf{f}_p(s)$ and $\mathbf{f}_o(s)$ are formed in such a way that the response of the second-order differential equation system (2) – (6) can approximate any smooth point-to-point trajectory from the initial position $\boldsymbol{p}_0$ and orientation $\boldsymbol{q}_0$ to the final position $\mathbf{g}_p$ and orientation $\mathbf{g}_o$. The nonlinear forcing terms are defined as linear combinations of M radial basis functions (RBFs)

$$\mathbf{f}_p(s) = \frac{\sum_{i=1}^{M} \mathbf{w}_{i,p}\Psi_i(s)}{\sum_{i=1}^{M} \Psi_i(s)} s, \tag{7}$$

$$\mathbf{f}_o(s) = \frac{\sum_{i=1}^{M} \mathbf{w}_{i,o}\Psi_i(s)}{\sum_{i=1}^{M} \Psi_i(s)} s, \tag{8}$$

$$\Psi_i(s) = \exp\left(-h_i\,(s - c_i)^2\right), \tag{9}$$

where free parameters $\mathbf{w}_{i,p}$, $\mathbf{w}_{i,o}$ determine the shape of position and orientation trajectories. c_i are the centers of RBFs, evenly distributed along the trajectory, with h_i their widths. By setting $\alpha_z = 4\beta_z > 0$ and $\alpha_s > 0$, the underlying second order linear dynamic system (2) – (6) becomes critically damped.

Compared to (Ude et al, 2014) and analogous to (Nemec et al, 2013), we introduced the temporal scaling function $\nu(s)$ which is used to specify variations from the demonstrated speed profile. Similarly to the forcing terms (7) and (8), it is encoded as a linear combination of M_v RBFs

$$\nu(s) = 1 + \frac{\sum_{j=1}^{M_v} v_j\Psi_j(s)}{\sum_{j=1}^{M_v} \Psi_j(s)}, \tag{10}$$

where v_j are the corresponding free parameters (weights).

In order to parameterize the demonstrated control policy with a DMP, the weights $\mathbf{w}_{i,p}$, $\mathbf{w}_{i,o}$ and v_j need to be calculated. The shape weights $\mathbf{w}_{i,p}$ and $\mathbf{w}_{i,o}$ are calculated by applying standard regression techniques (Ude et al, 2014) and using the demonstrated trajectory (1) as the target for weight fitting. For ν we initially set $v_j = 0$, i. e. $\nu = 1$, meaning that the demonstrated speed profile is left unchanged. v_j are assigned a different value only through the change of the execution speed.

Robot control

Position and orientation trajectories, obtained from DMP, are fed directly to the robot controller as reference values. In pHRI, safety is the primary concern and the robot controller should exhibit stable operation in all possible interactions with environments and humans, and should not produce unexpected motions. To achieve this goal we applied the passivity paradigm for the controller design. It has been widely used in robotics as it preserves stable operation with respect to the feedback and parallel interconnections of passive systems (Hatanaka et al, 2015; Zhang and Cheah, 2015). In our study we applied the two level passivity based impedance controller for manipulators with flexible joints in the form (Albu-Schaffer et al, 2007)

$$\boldsymbol{\rho_c} = \mathbf{B}\mathbf{B}_\Theta^{-1}\mathbf{u} + (\mathbf{I} - \mathbf{B}\mathbf{B}_\Theta^{-1})\boldsymbol{\rho} \tag{11}$$

$$\mathbf{u} = \mathbf{J}^{\mathrm{T}}(\boldsymbol{\theta})\ddot{\mathcal{X}}_c + \bar{\mathbf{g}}(\boldsymbol{\theta}) \tag{12}$$

where $\boldsymbol{\rho_c} \in \mathbb{R}^N$ is the control torque input for the motors, N is number of robot joints, $\boldsymbol{\theta} \in \mathbb{R}^N$ is the joint position measured at the motor side, $\mathbf{J} \in \mathbb{R}^{N\times 6}$ is the manipulator Jacobian, $\mathbf{B}$ and $\mathbf{B}_\Theta \in \mathbb{R}^{6\times 6}$ denote the positive definite diagonal matrix of joint and desired joint inertia, respectively. $\boldsymbol{\rho}$ are measured joint torques and $\bar{\mathbf{g}}(\boldsymbol{\theta})$ is the gravity vector estimated in such a way, that it provides exact gravity compensation in static case using the signals measured at the motor side (Ott et al, 2004). Basically, the role of the motor torque controller (11) is to reduce the motor inertia and to compensate for the robot non-linear dynamics. Desired impedance and damping is provided with (12).

The task command input $\ddot{\mathcal{X}}_c = [\ddot{\mathbf{p}}_c^{\mathrm{T}}, \dot{\boldsymbol{\omega}}_c^{\mathrm{T}}]^{\mathrm{T}}$ is chosen as

$$\ddot{\mathbf{p}}_c = -\mathbf{D}_p\dot{\mathbf{p}} + \mathbf{K}_p\mathbf{e}_p, \tag{13}$$

$$\dot{\boldsymbol{\omega}}_c = -\mathbf{D}_q\omega + \mathbf{K}_q\mathbf{e}_q, \tag{14}$$

where position and orientation tracking errors are defined as $\mathbf{e}_p = \mathbf{p}_d - \mathbf{p}$ and $\mathbf{e}_q = 2\log(\overline{\mathbf{q}}_p * \mathbf{q}_d)$. The quaternion logarithm $\log : \mathbf{S} \mapsto \mathbb{R}^3$ is given as

$$\log(\mathbf{q}) = \log(v, \mathbf{u}) = \begin{cases} \arccos(v)\dfrac{\mathbf{u}}{\|\mathbf{u}\|}, \ \mathbf{u} \neq 0 \\ [0,0,0]^{\mathrm{T}}, \ \text{otherwise} \end{cases}. \tag{15}$$

Its inverse, i. e., the exponential map $\exp : \mathbb{R}^3 \mapsto \mathbf{S}$, is defined as

$$\exp(\mathbf{r}) = \begin{cases} \cos(\|\mathbf{r}\|) + \sin(\|\mathbf{r}\|)\dfrac{\mathbf{r}}{\|\mathbf{r}\|}, \ \mathbf{r} \neq 0 \\ 1 + [0,0,0]^{\mathrm{T}}, \ \text{otherwise} \end{cases}. \tag{16}$$

Subscript $(.)_d$ denotes the desired values. Variables without a subscript denote the current values calculated from the robot joints at the motor side. $\mathbf{K}_p$, $\mathbf{K}_q \in \mathbb{R}^{3\times 3}$ are diagonal, positive definite positional stiffness and rotational stiffness matrices, respectively. They specify the properties of the controller in Cartesian coordinate system. Positional damping and rotational damping matrices $\mathbf{D}_p$, $\mathbf{D}_q \in \mathbb{R}^{3\times 3}$ are positive definite, but not necessary diagonal matrices. Proper damping design is crucial for preserving stability properties of the controller. Unlike in classical computed torque robot controller, damping matrices are configuration dependent. Let's express total manipulator inertia in the task coordinates as

$$\boldsymbol{\Lambda}(\boldsymbol{\theta}) = (\mathbf{J}(\boldsymbol{\theta})(\mathbf{H}(\boldsymbol{\theta}) + \mathbf{B}_{\boldsymbol{\theta}})^{-1}\mathbf{J}(\boldsymbol{\theta})^{\mathrm{T}})^{-1}, \tag{17}$$

where $\mathbf{H} \in \mathbb{R}^{N\times N}$ is the manipulator inertia in joint space. Next, we factorize task inertia as $\boldsymbol{\Lambda} = \bar{\boldsymbol{\Lambda}}\bar{\boldsymbol{\Lambda}}$ and proportional gain matrix as $\mathbf{K} = \bar{\mathbf{K}}\bar{\mathbf{K}}$. $\mathbf{K} \in \mathbb{R}^{6\times 6}$ is composed of $\mathbf{K}_p$ and $\mathbf{K}_d$. Damping matrix is then calculated as

$$\mathbf{D} = \bar{\boldsymbol{\Lambda}}\mathbf{D}_\xi\bar{\mathbf{K}} + \bar{\mathbf{K}}\mathbf{D}_\xi\bar{\boldsymbol{\Lambda}}, \tag{18}$$

where $\mathbf{D}_\xi \in \mathbb{R}^{6\times 6}$ is a diagonal matrix with the desired damping. Usually it is set to $\mathbf{I}$ for critically damped response. Corresponding $\mathbf{D}_p$ and $\mathbf{D}_q$ are obtained as upper and lower part of the $\mathbf{D}$, respectively. More details about the factorization based design of damping matrices can be found in (Albu-Schaffer et al, 2004)

3 Human-robot cooperation scheme

The operation of the proposed system is as follows. First, the human operator demonstrates the desired cooperative human-robot motion by kinesthetically guiding the robot arms. The demonstrated motion is then encoded by SS-DMPs for position and orientation ($\mathbf{p}$, $\mathbf{q}$) as explained in Section 2. The demonstration of the motion is typically performed slower than what is actually the final desired motion, because typically it is not possible to demonstrate the movement with both high speed and high accuracy. Hence we should allow the human operator to non-uniformly speed up or slow down the execution. In our proposed approach this happens on-line during the task execution, when the human coworker is allowed to modify the motion.

Second, the human operator and the robot iteratively perform the task several times. The learning of the course of motion as well as the learning of the speed profile is based on the adaptation of the desired trajectory and the estimation of trajectory variances across task repetitions. As suggested in (Calinon et al, 2010), low variance of motion indicates that the corresponding part of the task should be executed with high precision and that no further variations from the course of motion should be allowed. If little variance occurs in a few executions of the cooperative task, the robot should ensure precise trajectory tracking by increasing its stiffness in the directions perpendicular to the direction of motion. This allows the human co-worker to decrease his/her own precision as the stiffer robot provides disturbance rejection. Still, the human should be able to speed up the trajectory without affecting the course of motion.

To achieve such behavior, the robot system has to be compliant in the direction of motion. To the best of our knowledge, none of the previously proposed adaptation algorithms can simultaneously address these issues.

3.1 Trajectory adaptation

Initially, in the first iteration of the cooperative task, the robot is uniformly compliant in all directions. Consequently, the commanded trajectory $\mathbf{p}_l$ in task repetition cycle l is not the same as the actually executed trajectory $\mathbf{p}_m$ due to the input of the human. Here $(.)_l$ is the index of the task repetition, referred to also as learning cycle. Subscript $(.)_m$ referrers to the measured coordinates. The proposed adaptation algorithm updates the desired trajectory $(\mathbf{p}_l(s), \mathbf{q}_l(s))$, $l = 1, \ldots, L$, where the initial SS-DMP is taken from human demonstration $\mathbf{p}_1$, $\mathbf{q}_1$, and calculates its variance after each task execution.

We update the trajectory and associated covariance matrix using the following formulas

$$\mathbf{p}_{l+1}(s) = \zeta \varDelta \mathbf{p}(s) + \mathbf{p}_l(s), \tag{19}$$

$$\boldsymbol{\varSigma}_{p,l+1}(s) = (1-\zeta)(\boldsymbol{\varSigma}_{p,l}(s) + \zeta \varDelta \mathbf{p}(s) \varDelta \mathbf{p}(s)^{\mathrm{T}}), \tag{20}$$

$$\varDelta \mathbf{p}(s) = \mathbf{p}_m(s) - \mathbf{p}_l(s),$$

where $\mathbf{p}_m(s)$ denotes the measured position of the robot, $\boldsymbol{\varSigma}_{p,l}(s)$ is the current cycle covariance of $\mathbf{p}_l(s)$, all computed at phase x, and $\zeta \in \mathbb{R}^{[0,1]}$ is the exponentially weighting factor that defines the learning speed (Knuth, 1997). If we set $\zeta = 1$, the updated trajectory $\mathbf{p}_{l+1}$ is equal to the measured trajectory $\mathbf{p}_m$. On the other hand, if we set $\zeta = 0$, the trajectory $\mathbf{p}_{l+1}$ does not change and the system stops learning. After each learning cycle, the updated trajectory $\mathbf{p}_{l+1}$ is encoded into SS-DMP. It is used as the reference trajectory to control the robot in the next cycle. Note that all trajectories are phase dependent, sampled at $x(t), t = t_1, \ldots, t_T$. The coefficients of covariance matrix $\boldsymbol{\varSigma}_{p,l+1}$ are approximated with a linear combination of radial basis functions (RBFs). Eq. (19) cannot be used for orientation trajectories. Instead we apply the following update rule for quaternions

$$\mathbf{q}_{l+1}(s) = \exp\left(\zeta \frac{\boldsymbol{\omega}(s)}{2}\right) * \mathbf{q}_l(s), \tag{21}$$

$$\boldsymbol{\omega}(s) = 2\log(\mathbf{q}_m(s) * \overline{\mathbf{q}}_l(s)).$$

Similarly, the update rule for variation of orientation trajectories can be expressed with

$$\boldsymbol{\varSigma}_{q,l+1}(s) = (1-\zeta)(\boldsymbol{\varSigma}_{q,l}(s) + \zeta \boldsymbol{\omega}_l(s)\boldsymbol{\omega}_l(s)^{\mathrm{T}}). \tag{22}$$

3.2 Stiffness adaptation

We dynamically set the desired stiffness of the robot in order to improve the ease of adaptation. It is well known that the precision and speed of human motion are related – to be precise, humans reduce their speed (Fitts, 1954). While Calinon et al. (Calinon et al, 2010) proposed to decrease the stiffness in the parts of the trajectory with higher variability and vice versa, we propose to make the change of stiffness dependent not only on the variance but also on the speed of motion. The idea here is to make the robot compliant when the typically slow fine-tuning of the trajectory is required.

Let $\mathbf{R}_p$ denote the rotation matrix of the coordinate frame $\boldsymbol{\xi}_p$ with x coordinate specified in the desired direction of motion, i. e., $\dot{\mathbf{p}}_l$, and the other two coordinates orthogonal to it, as illustrated in Fig. 1. This matrix can be obtained by forming the Frenet-Serret frame (Ravani and Meghdari, 2006; Chiaverini and D., 2008) at each sampling time. The Frenet-Serret frame consists of three orthogonal directions defined by the path's tangent (direction of motion), normal, and binormal. We obtain the following expression for $\mathbf{R}_p$

$$\mathbf{R}_p = \begin{bmatrix} \mathbf{t} \; \mathbf{n} \; \mathbf{b} \end{bmatrix}, \tag{23}$$

$$\mathbf{t} = \frac{\dot{\mathbf{p}}_l}{\|\dot{\mathbf{p}}_l\|}, \; \mathbf{b} = \frac{\dot{\mathbf{p}}_l \times \ddot{\mathbf{p}}_l}{\|\dot{\mathbf{p}}_l \times \ddot{\mathbf{p}}_l\|}, \; \mathbf{n} = \mathbf{b} \times \mathbf{t}.$$

Note that the absolute velocity $\dot{\mathbf{p}}_l$ and acceleration $\ddot{\mathbf{p}}_l$ are provided by DMP integration at every phase s, which ensures smoothness. $\|\dot{\mathbf{p}}_l\| < \varepsilon$ or $\|\dot{\mathbf{p}}_l \times \ddot{\mathbf{p}}_l\| < \varepsilon$, where $\varepsilon > 0$ is a predefined threshold, means that the motion is slow or linear. Thus in such cases we suspend the updating of $\mathbf{R}_p$ until the motion becomes faster again. The same problem might appear at the beginning of the trajectory, which might start with zero speed and accelerations. In such a case we have to integrate SS-DMP a few steps ahead until $\|\dot{\mathbf{p}}_l \times \ddot{\mathbf{p}}_l\| > \varepsilon$ and set the corresponding Frenet-Serret frame as the initial $\mathbf{R}_p$. We also compute the robot's speed, i. e., $v = \|\dot{\mathbf{p}}_l\|$ and define scalar v_0, which specifies the threshold between the low and high speed. The appropriate control gain $\mathbf{K}_p$ at each sampling time is computed as follows

$$\mathbf{K}_p(s) = \mathbf{R}_p^{\mathrm{T}} \begin{bmatrix} k_x & 0 & 0 \\ 0 & \dfrac{k_o \rho}{\varSigma_{yy} + \epsilon} & 0 \\ 0 & 0 & \dfrac{k_o \rho}{\varSigma_{zz} + \epsilon} \end{bmatrix} \mathbf{R}_p, \tag{24}$$

where $\epsilon > 0$ is an empirically chosen constant which sets the upper bound for the controller gain and k_x and k_o are the gain constants in the direction of motion and orthogonal to it, respectively. With this choice, the

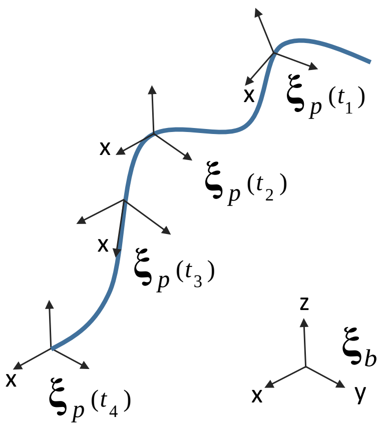

Fig. 1 Operational space $\boldsymbol{\xi}_p$ is defined by path orientation.

control error is actually transformed from the global coordinated system $\boldsymbol{\xi}_b$ to the trajectory operational space $\boldsymbol{\xi}_p$, multiplied by the gains defined in the trajectory operational space and rotated back to the global coordinate system (Khatib, 1987). Σ_{yy} and Σ_{zz} are the second and the third diagonal coefficient of $\boldsymbol{\Sigma}$, respectively. Transformation of $\boldsymbol{\Sigma}_{p,l}$ calculated from errors measured in global coordinate system $\boldsymbol{\xi}_b$ to the trajectory coordinate system is done by (Soler and Chin, 1985)

$$\boldsymbol{\Sigma} = \mathbf{R}_p \boldsymbol{\Sigma}_{p,l}(s) \mathbf{R}_p^{\mathrm{T}}. \tag{25}$$

Scalar ρ scales the control gains with respect to the calculated velocity v. To assure smooth transition of control gains at low velocities, it is calculated as

$$\rho = \gamma_1 \left(\tanh \left(\frac{v - v_0}{\gamma_3} \right) - 1 \right) + \gamma_2. \tag{26}$$

In the above equation, $\gamma_1, \gamma_2, \gamma_3 > 0$ determine the range, lower bound and the speed of transition between the lower and upper bound of the switching function defined by tanh, respectively. The initial value for covariance matrix $\boldsymbol{\Sigma}_{p,1}$ is set to $s_0 \mathbf{I}$, where s_0 is specified so that we obtain the desired initial stiffness orthogonal to the direction of motion. By pre-multiplying and post-multiplying gains with $\mathbf{R}_p^{\mathrm{T}}$ and $\mathbf{R}_p$, we can set significantly different stiffnesses in the direction of motion and orthogonal to it.

By choosing a constantly low value for k_x in $\mathbf{K}_p$, the robot is always compliant in the direction of motion, while the stiffness orthogonal to this direction is set according to the learned variance and speed of motion.

3.3 Speed adaptation

As previously explained, low gain k_x enables the human operator to freely move the robot in the tangent direction of the Frenet-Serret frame. At the same time, we would like that the robot stays on the commanded (learned trajectory) which means, that the human can only anticipate or lag behind the commanded trajectory. Obviously, this changes the speed of the trajectory, which is determined by the speed scaling factor ν in Eqs. $(2-6)$. Since all signals which determine the commanded robot pose are phase dependent, our task is to find the corresponding phase, which minimizes the difference between the current perturbed robot pose and a pose on the learned trajectory.

Lets define the tracking error $e_{xp} = [1 \ 0 \ 0] \ \mathbf{R}_p \mathbf{e}_p$, $\mathbf{e}_p = \mathbf{p}_l - \mathbf{p}_m$, which is the x component of the tracking error in path operational space (see Fig. 2). This error then determines the speed scaling factor, calculated as

$$\dot{\nu}(s) = \lambda \nu(s) e_{xp}, \tag{27}$$

where $\lambda > 0$ is an empirically set constant. This equation is executed at each sample time in the loop until it converges, typically in a few samples. The initial value is set to the speed scaling learned in previous cycles with $\nu(s) = \nu_l(s)$. With this procedure we actually speed up or slow down the commanded trajectory according to the human interaction.

Note that negative e_{xp} means that the actual robot position is anticipating the desired trajectory. In this case we have to speed up the desired trajectory by decreasing the scaling factor $\nu(s)$, and vice versa, with positive e_{xp} we slow down the desired trajectory by increasing the scaling factor $\nu(s)$. Values $\nu = (0, 1)$ speeds up the demonstrated trajectory, while $\nu = (1, \infty)$ slows down the trajectory. To compensate for this non-linear span, the update rate in (27) is weighted with the current value $\nu(s)$.

After sampling we compute the weights v_i that specify $\nu(s)$ defined as in (10). In this way we achieve faster convergence towards the desired trajectory in the direction of motion. The described procedure calculates the speed scale factor in the current learning cycle. Based on this we calculate the expected speed scale for the next learning cycle similar as for the positional part of the trajectory using

$$\begin{aligned} \nu_{l+1}(s) &= \zeta \Delta\nu(s) + \nu_l(s), \\ \Delta\nu(s) &= \nu(s) - \nu_l(s). \end{aligned} \tag{28}$$

The overall learning and adaptation algorithm is summarized in Algorithm 1.

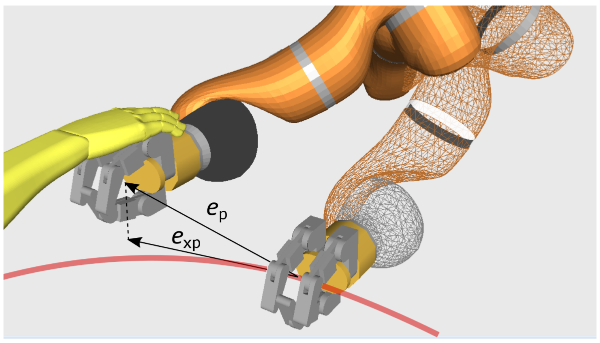

Fig. 2 Position tracking error e_p is projected to the tangential axis of the Frenet-Serret frame at each sampling instance. Wire frame model shows the commanded robot pose. Actual robot pose, denoted with solid model, is displaced due to the human operator interaction.

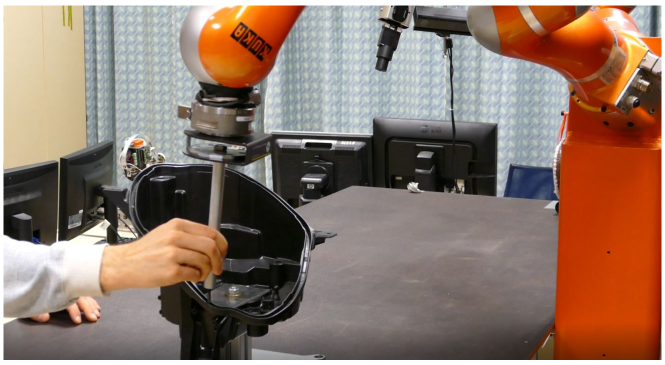
Fig. 3 Learning of a complex trajectory, where the spatial and the velocity part of the trajectory were demonstrated separately.

4 Experimental evaluation

4.1 LbD with separate learning of spacial policy and velocity profiles

The proposed human-robot cooperation scheme allows implementing two phase LbD learning, where the demonstration of the spacial part of the trajectory is followed by the demonstration of the velocity part. As humans can not be precise and fast at the same time (Fitts, 1954), complex trajectories can only be demonstrated at low speeds. During the execution, the learned trajectory needs to be accelerated. In many cases, this acceleration is non-uniform. Some parts may be executed faster and some parts not, often due to the technological requirements, e.g., when applying adhesives, welding, etc. The main idea is to demonstrate a complex policy at an arbitrary low speed with an arbitrary velocity profile. Next, the human demonstrates also the velocity part of the trajectory, while the robot maintains the learned spacial trajectory. Learning of the spatial part of the trajectory is performed as usual, e.g. by capturing the desired policy by kinesthetic guidance as described in Section 2 using (2) – (6). In the next step, Frenet-Serret frames are computed at each sampling time (23). While learning the velocity part, the robot is set compliant along the tangential direction of the Frenet-Serret frames and stiff orthogonal to this direction by selecting appropriate control gains $k_y,\ k_z \gg k_x$ in

Algorithm 1: Human-robot cooperation algorithm

```
1  Record $\{\mathbf{p}(k), \mathbf{q}(k), t_k\}_{k=1}^{T}$ using kinesthetic guiding
     and calculate SS-DMP parameters from the
     demonstrated data $(\mathbf{p}_1, \mathbf{q}_1)$
2  Initialize gains $k_x, k_o$ and set initial covariance
     matrices $\boldsymbol{\Sigma}_{p,1} = s_0\mathbf{I}$. Approximate coefficients of
     $\boldsymbol{\Sigma}_{p,1}$ with a linear combination of RBFs.
3  set $l = 1$
4  while cooperating do
5      set initial phase $s = 1$
6      while $s \le s_{min}$ do
7          integrate SS-DMP to obtain $\mathbf{p}_l(s), \mathbf{q}_l(s)$ as
             well as their velocities and accelerations
8          calculate path rotation $\mathbf{R}_p(s)$ using (23) and
             speed $v(s)$
9          calculate $\mathbf{K}_p(s)$ and $\mathbf{D}_p(s)$ using (24) and
             (18), respectively
10         execute control law (12) with $\mathbf{p}_l(s)$, $\mathbf{q}_l(s)$ as
             the desired trajectory
11         sample new trajectories $\mathbf{p}_{l+1}(s), \mathbf{q}_{l+1}(s)$,
             covariance matrices $\boldsymbol{\Sigma}_{p,l+1}(s)$, and calculate
             speed scaling factor $\nu_{l+1}$, all at phase $s$,
             using (19) – (21), (27)
12     calculate SS-DMP parameters of $\mathbf{p}_{l+1}, \mathbf{q}_{l+1}$,
         including $\nu_{l+1}$
13     approximate coefficients of $\boldsymbol{\Sigma}_{p,l+1}$ with linear
         combinations of RBFs
14     set $l = l + 1$
```

$$\mathbf{K}_p(s) = \mathbf{R}_p^{\mathrm{T}} \begin{bmatrix} k_x & 0 & 0 \\ 0 & k_y & 0 \\ 0 & 0 & k_z \end{bmatrix} \mathbf{R}_p. \tag{29}$$

This allows the human demonstrator to push and pull the robot end-effector in the direction of the trajectory, which actually modifies the execution speed. In order to maintain the previously demonstrated spatial part of the trajectory, the robot has to set the corresponding previously learned reference point $(\mathbf{p}_d(s), \mathbf{q}_d(s))$ by determining the speed scaling factor $\nu(s)$ which minimizes the difference between the current robot pose and the learned spatial trajectory using (27). In this way, it learns also the speed scaling. The corresponding weights v_j in (10) are then calculated using regression (Ude et al, 2010).

We implemented the described LbD procedure on a Kuka LWR-4 robot arm, where the task was to follow the gap of the automotive lamp, as shown in Fig. 3. The

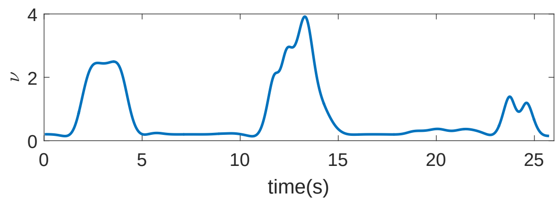

Fig. 4 Estimated speed scaling factor ν during the demonstration of the speed profile.

exact task was not an issue in this case, but it might be glue application, grinding, inspection, etc. The spatial part of this complex trajectory was demonstrated with kinesthetic guiding, where the control gains were set to $\mathbf{K}_p = 1\,\mathbf{I}$ N/m and $\mathbf{K}_q = 0.2\,\mathbf{I}$ Nm/rd. The trajectory was captured in Cartesian coordinates. After that, the captured trajectory was encoded with SS-DMPs, where the speed scaling factor $\nu(s)$ was set to 1. For the demonstration of the velocity part of the policy, we raised the control gains to $k_x = 500$ N/m, $k_y, k_z = 2000$ N/m, and $\mathbf{K}_q = 200\,\mathbf{I}$ Nm/rd. During the speed learning, we calculate $\mathbf{R}_p$ at each sampling time by (23), control gains by (29) and speed scaling by (27). The human operator was able to guide the robot along the previously learned trajectory with arbitrary speed with very low physical effort. For practical reasons, we limited the learned velocity scale factor to the interval $\nu = [0.2, 5]$. Fig. 4 shows learned speed scale during this experiment.

Video of this experiment is available at `http://abr.ijs.si/upload/1483017570-TwoPhaseLbD.mp4`.

4.2 Coaching with variable stiffness and variable weighting factor

The next experiment demonstrates how to apply the proposed approach to coaching. The goal of the coaching is to modify only a part of the previously learned trajectory while leaving the rest of the trajectory unchanged (Gams et al, 2016). During the coaching, it is desirable that we could learn a new part of the trajectory in single pass while obtaining good disturbance rejection in the part where we would like to preserve the previously learned trajectory. To achieve this goal, we apply the compliance adaptation scheme given by the Eqs. (24),(26) and introduce a variable weighting factor ζ in trajectory update (19)–(21). We associate ζ with the tracking error,

$$\zeta(k) = \begin{cases} \zeta_{max}, & \|e_p(k)\| > d\frac{\zeta_{max}}{\zeta_{min}} \\ \zeta_{min}, & \|e_p(k)\| < d \\ \frac{\zeta_{min}}{d}\|e_p(k)\|, & \text{otherwise} \end{cases}, \tag{30}$$

where ζ_{min} and $\zeta_{max} \in \mathbb{R}^{[0,1]}$ are minimal and maximal exponential weighting factors respectively, and d is the position error at which point the variable weighting factor starts to change. In exactly the same way we can associate the weighting factor to the orientation error. The coaching works as follows. We drive the robot along the previously learned trajectory. Low value of the weighing factor ζ_{min} and high control gains as a consequence of low values of the covariance matrices Σ_p and Σ_q provide good disturbance rejection. When we would like to modify the trajectory, we first decrease the speed below v_0 (26). The system drops the stiffness and allows us to modify the trajectory. Consequently, the weighting factor changes to ζ_{max}. By setting $\zeta_{max} \approx 1$ the system learns the new trajectory in a single pass, as it updates it using the current robot configurations only. When we re-approach the previously learned trajectory, Eq. (30) decreases ζ. Note that it is required to slow down the trajectory below v_0 only when we want to initiate coaching. After that, low compliance will be provided by increased values of the covariance matrices Σ_p and Σ_q. Note also that it is necessary to calculate the current speed scaling factor $\nu(s)$ using (27) in each sampling interval in order to get the corresponding trajectory reference values.

The coaching was tested for a simple case where the initial trajectory was learned with a single demonstration. Consequently, all of the elements of the covariance matrices Σ_p and Σ_q were 0 and the system calculated high control gains. In the next cycle, the operator decreased the speed in order to decrease the stiffness and demonstrated a new part of the trajectory. The original and the modified trajectories are pictured in Fig. 5 with red and blue lines respectively. In this experiment we applied the following settings: $\zeta_{min} = 0.4$, $\zeta_{max} = 0.8$, $v_0 = 0.07$ m/s, $d = 0.02$ m. In order to make the coaching even more efficient, the robot reference was actually the measured position whenever the $\zeta = \zeta_{max}$. This way, we can apply extensive position and orientation changes with very little physical effort. Fig. 6 shows how the system adjusted the control gains and weighting factor ζ during the coaching. The coaching area is marked with a shaded background in the corresponding plots. Note that the gains are expressed in the trajectory coordinate system, i.e. calculated by (24), before they were pre and post-multiplied with $\mathbf{R}_p^T$ and $\mathbf{R}_p$, respectively. Resulting robot trajectories are displayed in Fig . 7. In this plot, the initial trajectory is marked in red. The actual demonstrated trajectory (blue) differs in the middle part. The learned trajectory is denoted with a black dotted line. Since ζ_{max} was 0.8, it slightly differs from the demonstrated one. With $\zeta_{max} = 1$ it would perfectly follow the demonstrated trajectory. However, it would be less smooth due to the poorer disturbance rejection.

Video of this experiment is available at `http://abr.ijs.si/upload/1494512444-Coaching.mp4`.

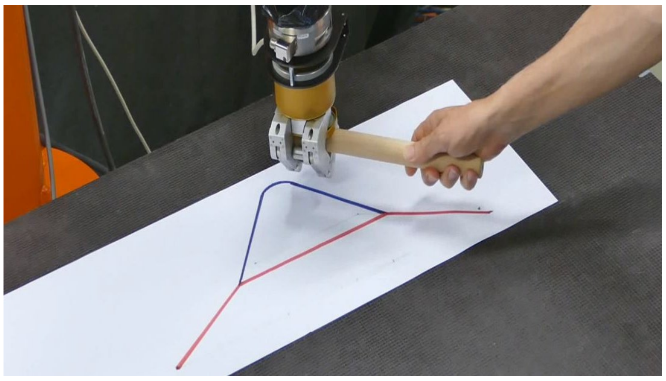
Fig. 5 Coaching; The red line is the original trajectory. The blue line denotes the desired change of the original trajectory.

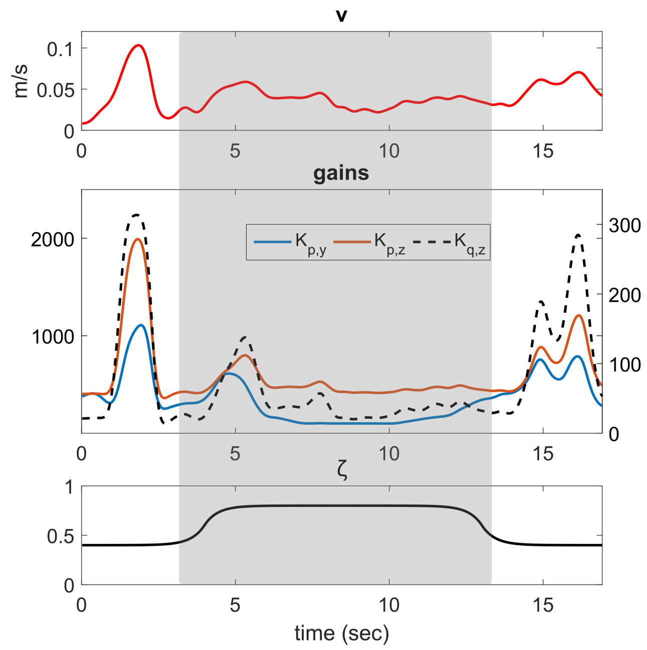

Fig. 6 Speed v, controller gains $\mathbf{R}_p\mathbf{K}_p\mathbf{R}_p^{\mathrm{T}}$ and weighting factor ζ during the coaching. The shaded area denotes the coaching phase.

4.3 Bimanual human robot cooperation in object transportation

The third experiment involves human robot cooperation (HRC) in transportation of a (potentially large and heavy) object. The task of the robot was to learn how

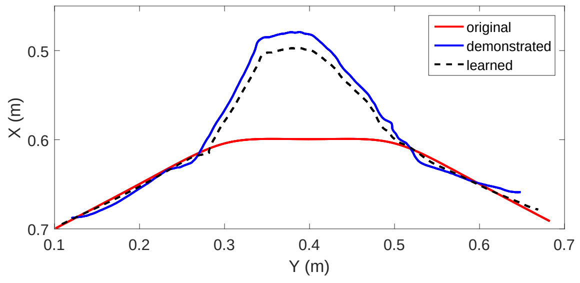

Fig. 7 The original, demonstrated and learned trajectories during the coaching.

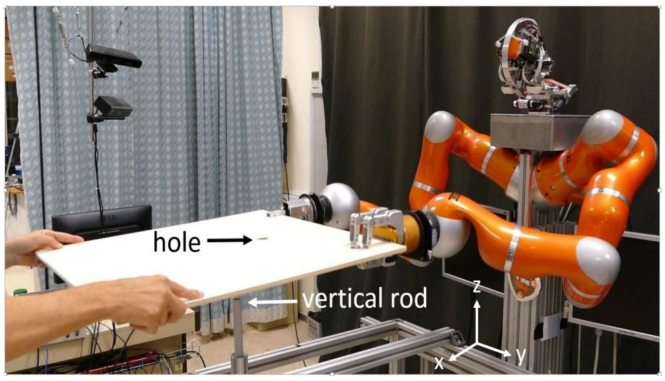

Fig. 8 Cooperating humanoid robot and human.

to cooperate with the human while transporting a rigid plate from the initial point to the final point and simultaneously avoiding an obstacle. The final point had to be precisely learned, as it was necessary to insert a hole in the panel on the vertical rod, as shown in Fig. 8. Due to the dimensions of the plate, this task can not be successfully accomplished using a single robot arm. Therefore, we applied bi-manual robot setup composed of two 7 degree of freedom Kuka LWR-4 robot arms. For bi-manual robot control we applied coordinated task-space framework as presented in (Nemec et al, 2016a). It fully characterizes a cooperative operational space and allows the user to specify the task in relative and absolute coordinates, resulting in geometrically meaningful motion variables defined at the position/orientation level (Caccavale et al, 2000). With a proper choice of relative and absolute coordinates (which might be also task dependent), both subspaces are decoupled both on kinematic as well as on dynamics level. Consequently, we can control each subsystem independently. In our experiment, the robot firmly holds the shared object. This means that the corresponding relative coordinates do not change during the task. However, in order to minimize internal wrench, it was necessary to control also the relative coordinates. These coordinates remained unchanged during entire task. The gains of the relative coordinates control law were set to 800 **I** N/m and 300 **I** N/rd, respectively.

Our human robot task coordination is fully characterized with absolute coordinates, which were the subject of our adaptation scheme (Nemec et al, 2016b). Therefore, all learning and adaptation procedures are applied to absolute coordinates only. The cooperation scheme adapts autonomously. The only parameters we have to set are initial gains, speed threshold and smoothing coefficients γ_1, γ_2 and γ_3. The initial gains $\mathbf{K}_p$ and $\mathbf{K}_q$ were set to $100\,\mathbf{I}$ N/m and $80\,\mathbf{I}$ Nm/rd, respectively. Thus, the system was initially very compliant in absolute coordinates. The speed threshold v_0, where the system starts adjusting the stiffness, was empirically set to 0.1 m/s.

After the first cycle, which corresponds to the initial task demonstration, we performed 8 cooperative repetitions of the task. The learning factor ζ was set to 0.4. Fig. 9 shows the 3-D plot of the trajectories $\mathbf{p}_l$ in absolute coordinate system. The execution speed v_a and the learned gains $\mathbf{R}_p\mathbf{K}_p\mathbf{R}_p^{\mathrm{T}}$ during subsequent executions are shown in Fig. 11. Please note that the control gains in these plots are expressed in the trajectory coordinate system in order to gain better insight into the behavior of the system. To get the control gains which are passed to the robot controller, these gains have to be pre and post-multiplied with $\mathbf{R}_p^T$ and $\mathbf{R}_p$ respectively (see Eq. (24). Control gains at the initial and final parts of the trajectory are always low as we start and end with low velocity. Note also that the control gains calculated from (24) have to be filtered in order to prevent jerky movements. For the sake of clarity only the 4th, 6th and 8th cooperation cycles are displayed. Note that in this experiment the spatial and the velocity parts were learned concurrently and not separately as in the first experiment. From the plots we can see how the speed has increased during subsequent cycles.

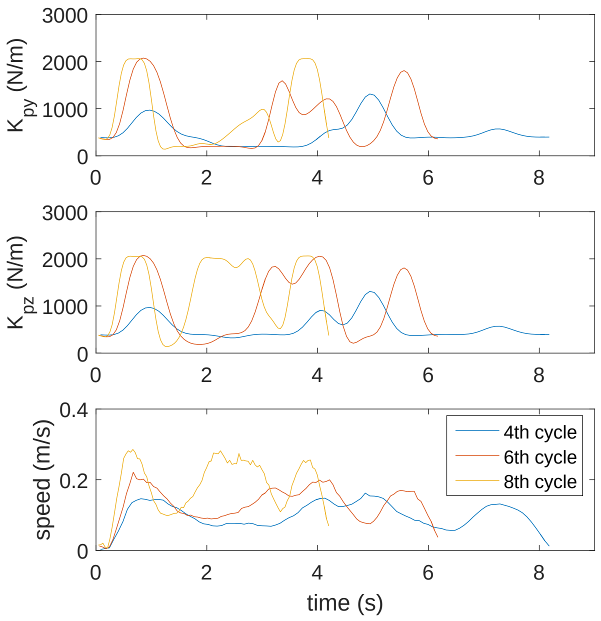

Fig. 11 Learned gains $\mathbf{R}_p\mathbf{K}_p\mathbf{R}_p^{\mathrm{T}}$ and speed v before the vertical rod displacement.

After 8 repetitions we displaced the vertical rod by 10 cm in the global y-direction. Thus, the final part of the task had to be modified. By lowering the speed in that part of the trajectory through interaction, the system immediately decreased the stiffness and allowed for the robot to be guided to the new position. In order to speed up learning of a new part of the trajectory, we increased weighting factor ζ to 0.9 in a similar way as in the previous experiment. In a few repetitions, the system learned the new task and re-set the high stiffness gains. This enabled the human operator to accomplish the task by allowing the robot to guide him. Also here, for the sake of clarity, we show only the 9th, 10th, 12th and 14th cooperation cycles.

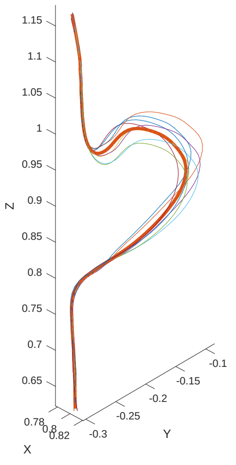

Fig. 9 3-D plot of trajectories of absolute coordinates before the vertical rod displacement. The thick line shows the final learned trajectory.

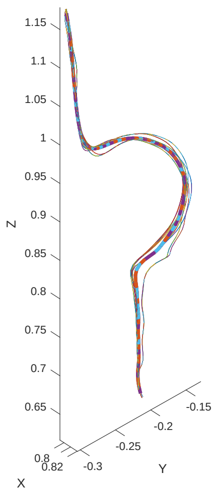

Fig. 10 3-D plot of trajectories of absolute coordinates after the vertical rod displacement in y direction.

Fig. 10 shows the 3-D plot of trajectories $\mathbf{p}_l$ after the displacement. Execution speed v and controller gains $\mathbf{R}_p\mathbf{K}_p\mathbf{R}_p^{\mathrm{T}}$ are displayed in Fig. 12.

This experiment is shown also in the video available at `http://abr.ijs.si/upload/1483017628-HRC-Plate.mp4`.

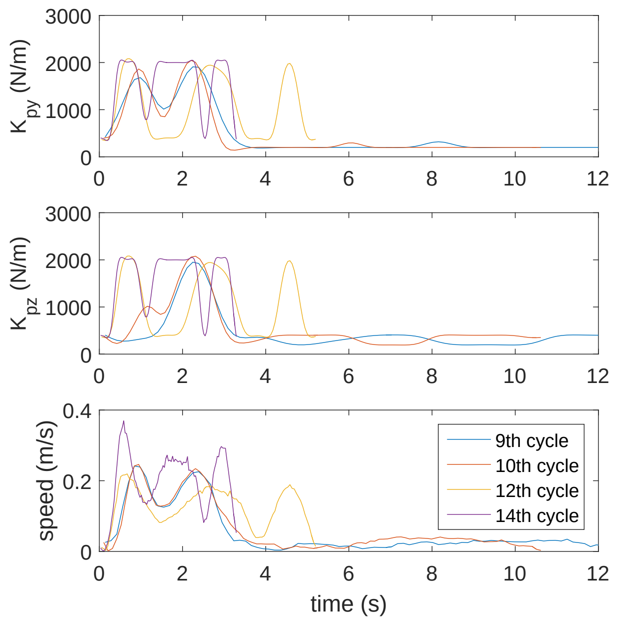

Fig. 12 Learned gains $\mathbf{R}_p\mathbf{K}_p\mathbf{R}_p^{\mathrm{T}}$ and speed v after the vertical rod displacement.

5 Conclusions

In this work we proposed a new human-robot cooperation scheme. It can be applied to various tasks, e.g. for a) two phase LbD with separate demonstration of the spatial and velocity part of the trajectory, b) human robot cooperation during the transportation of heavy and bulky objects, c) for human-robot cooperation during assembly tasks, etc. The algorithm can be applied to both single arm as well as bi-manual robotic systems and it doesn't require force sensing. The developed algorithm is based on the previously proposed SS-DMPs (Nemec et al, 2013) and extended cooperative task approach for bi-manual robots (Likar et al, 2015). There are several novelties in the proposed approach:

- Speed-scaled DMPs in Cartesian space have been introduced.
- Both spatial movement and the speed of cooperative motion can be adapted.
- Stiffness of the cooperative task is adjusted taking into account the variance of motion across several executions of the task and the current speed of motion. This enables the human to override the learned high stiffness when necessary.
- Task compliance is defined with respect to the trajectory operational space, which allows for varying the dynamic properties of the system along the direction of motion.

Note that no force sensing is necessary in the proposed approach, which might decrease the final cost of the setup and increase the robustness, since force measurement is usually noisy. Another advantage of the proposed approach is the possibility to trim the learning speed and the disturbance rejection with the variable exponential weighting factor ζ. On the other hand, there are many parameters such as threshold velocities v_0 in (26), weighting factor ζ, initial gains k_0 and k_x in (24), λ in (27) which need careful tuning. In the future, we will focus on procedures that will either diminish the number of tuning parameters or learn their values from previous experience by means of reinforcement learning.

The proposed scheme was experimentally verified in three exemplary use-cases. The first is a novel two-phase LbD scheme, which has the potential to be used for learning of complex tasks in industrial setups as well as for robots applied in home environments. With the second experiment we show how to use the proposed scheme in coaching and how to adapt the weighting factor ζ in order to obtain an appropriate trade-off between the learning speed and disturbance rejection. The third experiment deals with human-robot cooperation in object transportation, which could be applied in assembly processes in production plants or in civil engineering for transportation of heavy and bulky objects.

Acknowledgment

The research leading to these results has received partial funding from the Horizon 2020 FoF Research & Innovation Action no. 723909, AUTOWARE, and by the GOSTOP programme, contract no C3330-16-529000, co-financed by Slovenia and the EU under the ERDF.

References

Adorno B, Fraisse P, Druon S (2010) Dual position control strategies using the cooperative dual task-space framework. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Taipei, Taiwan, pp 3955–3960

Adorno BV, Bó APL, Fraisse P, Poignet P (2011) Towards a cooperative framework for interactive manipulation involving a human and a humanoid. International Conference on Robotics and Automation (ICRA) pp 3777–3783

Albu-Schaffer a, Ott C, Hirzinger G (2004) A passivity based Cartesian impedance controller for flexible joint robots - part II: full state feedback, impedance design and experiments. IEEE International Conference on Robotics and Automation, 2004 Proceedings ICRA '04 2004 3(5):2659–2665

Albu-Schaffer A, Ott C, Hirzinger G (2007) A Unified Passivity-based Control Framework for Position, Torque and Impedance Control of Flexible Joint Robots. The International Journal of Robotics Research 26(1):23–39

Amor HB, Neumann G, Kamthe S, Kroemer O, Peters J (2014) Interaction primitives for human-robot cooperation tasks. In: 2014 IEEE International Conference on Robotics and Automation (ICRA), pp 2831–2837

Caccavale F, Chiacchio P, Chiaverini S (2000) Task-space regulation of cooperative manipulators. Automatica 36:879–887

Calinon S, Sardellitti I, Caldwell DG (2010) 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

Calinon S, Li Z, Alizadeh T, Tsagarakis NG, Caldwell DG (2012) Statistical dynamical systems for skills acquisition in humanoids. In: 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids), Osaka, Japan, pp 323–329

Calinon S, Bruno D, Caldwell DG (2014) A task-parameterized probabilistic model with minimal intervention control. In: IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, pp 3339–3344

Chiaverini G S Oriolo, D WI (2008) Chapter 11: Kinematically redundant manipulators. In: Springer Handbook of Robotics, Springer Berlin Heidelberg, pp 245–268

Evrard P, Mansard N, Stasse O, Kheddar A, Schauss T, Weber C, Peer A, Buss M (2009) Intercontinental, multimodal, wide-range telecooperation using a humanoid robot. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp 5635–5640

Ewerton M, Neumann G, Lioutikov R, Amor HB, Peters J, Maeda G (2015) Learning multiple collaborative tasks with a mixture of interaction primitives. In: 2015 IEEE International Conference on Robotics and Automation (ICRA), pp 1535–1542

Faber M, Bützler J, Schlick CM (2015) Human-robot Cooperation in Future Production Systems: Analysis of Requirements for Designing an Ergonomic Work System. Procedia Manufacturing 3:510–517

Fitts PM (1954) The information capacity of the human motor system in controlling the amplitude of movement. Journal of Experimental Psychology 47(6):381–391

Gams A, Nemec B, Ijspeert AJ, Ude A (2014) Coupling movement primitives: Interaction with the environment and bimanual tasks. IEEE Transactions on Robotics 30(4):816–830

Gams A, Petrič T, Do M, Nemec B, Morimoto J, Asfour T, Ude A (2016) Adaptation and coaching of periodic motion primitives through physical and visual interaction. Robotics and Autonomous Systems 75, Part B:340 – 351

Hatanaka T, Chopra N, Spong MW (2015) Passivity-Based Control of Robots : Historical Perspective and Contemporary Issues. Conference on Decision and Control (CDC) pp 2450–2452

Ijspeert AJ, Nakanishi J, Hoffmann H, Pastor P, Schaal S (2013) Dynamical movement primitives: Learning attractor models for motor behaviors. Neural Computation 25(2):328–73

Khansari-Zadeh SM, Billard A (2011) Learning stable nonlinear dynamical systems with gaussian mixture models. IEEE Transactions on Robotics 27(5):943–957

Khatib O (1987) A unified approach for motion and force control of robot manipulators: The operational space formulation. IEEE Journal of Robotics and Automation 3:43–53

Knuth DE (1997) The Art of Computer Programming, Volume 2 (3rd Ed.): Seminumerical Algorithms. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA

Krebs HI, Hogan N, Aisen ML, Volpe BT (1998) Robot-aided neurorehabilitation. IEEE Transactions on Rehabilitation Engineering 6(December):75–87

Likar N, Nemec B, Zlajpah L, Ando S, Ude A (2015) Adaptation of bimanual assembly tasks using iterative learning framework. In: IEEE-RAS International Conference on Humanoid Robots (Humanoids), Seoul, Korea, pp 771–776

Mortl A, Lawitzky M, Kucukyilmaz A, Sezgin M, Basdogan C, Hirche S (2012) The role of roles: Physical cooperation between humans and robots. The International Journal of Robotics Research 31(13):1656–1674

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

Nemec B, Likar N, Gams A, Ude A (2016a) Adaptive human robot cooperation scheme for bimanual robots. In: Lenarcic J, Merlet J (eds) Advances in Robot Kinematics, INRIA, pp 385–393

Nemec B, Likar N, Gams A, Ude A (2016b) Bimanual human robot cooperation with adaptive stiffness control. In: 16th IEEE-RAS International Conference on Humanoid Robots, Cancun, Mexico, pp 607–613

Ott C, Albu-Schaffer A, Kugi A, Stramigioli S, Hirzinger G (2004) A passivity based cartesian impedance controller for flexible joint robots-part I: Torque feedback and gravity compensation. IEEE International Conference on Robotics & Automation pp 2659–2665

Paraschos A, Daniel C, Peters J, Neumann G (2013) Probabilistic movement primitives. In: Neural Information Processing Systems 26, pp 2616–2624

Park HA, Lee CSG (2015) Extended cooperative task space for manipulation tasks of humanoid robots. In: IEEE International Conference on Robotics and Automation (ICRA), Seattle, WA, pp 6088–6093

Raiola G, Lamy X, Stulp F (2015) Co-manipulation with multiple probabilistic virtual guides. In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp 7–13

Ramacciotti M, Milazzo M, Leoni F, Roccella S, Stefanini C (2016) A novel shared control algorithm for industrial robots. International Journal of Advanced Robotic Systems 13(6):1729881416682,701, DOI 10.1177/1729881416682701

Ravani R, Meghdari A (2006) Velocity distribution profile for robot arm motion using rational Frenet-Serret curves. Informatica 17(1):69–84

Rozo L, Calinon S, Caldwell DG, Jimnez P, Torras C (2016) Learning physical collaborative robot behaviors from human demonstrations. IEEE Transactions on Robotics 32(3):513–527

Salehian SSM, Khoramshahi M, Billard A (2016) A dynamical system approach for softly catching a flying object: Theory and experiment. IEEE Transactions on Robotics 32(2):462–471

Soler T, Chin M (1985) On transformation of covariance matrices between local cartesian coordinate systems and commutative diagrams

Soyama R, Ishii S, Fukase A (2004) Selectable operating interfaces of the meal-assistance device "my spoon". In: Bien Z, Stefanov D (eds) Rehabilitation, Springer Berlin / Heidelberg, pp 155–163

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

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

Zhang J, Cheah CC (2015) Passivity and Stability of Human-Robot Interaction Control for Upper-Limb Rehabilitation Robots. IEEE Transactions on Robotics 31(2):233–245