Bimanual Human Robot Cooperation with Adaptive Stiffness Control

Bojan Nemec, Nejc Likar, Andrej Gams, Aleš Ude

***Abstract*— We propose a human-robot cooperation scheme for bimanual robots. After the initial task demonstration, the human co-worker can modify both the spatial course of motion as well as the speed of execution in an intuitive way. To achieve this goal, speed-scaled dynamic motion primitives are applied for the underlying task representation. The proposed adaptation scheme adjusts the robot's stiffness in path operational space, i. e. along the trajectory. It allows a human co-worker to be less precise in the parts of the task that require high precision, as the precision aspect can be provided by the robot. The required dynamic capabilities of the robot were obtained by decoupling the bimanual robot dynamics in operational space, which is attached to the desired trajectory. The proposed scheme was validated in a task where two Kuka LWR-4 robot arms cooperate with a human to carry an object.**

I. Introduction

Physical human-robot interaction (pHRI) with an aim towards cooperatively performing a task (human-robot co-operation) distinguishes between two levels of control [1]. In the most common one, the human operator has complete control over the evolution of the joint task and plays the role of a master, while the robot plays the role of the slave. The interaction is provided through force or visual feedback [2]. Alternatively, the task can be controlled by the robot and only initiated by the human [3]. Finally, in some applications, e. g. in rehabilitation robotics [4], the control over the task evolution is dynamically shared between the human and the robot [5].

A. Problem statement

In this paper we investigate the learning of robot behavior during bimanual human-robot cooperative tasks. In order to achieve natural learning and adaptive execution of human robot cooperation, the task control must

- allow non-uniform changes of execution speeds,
- enable effective bimanual robot control by properly representing the kinematics and dynamics of a bimanual system,
- 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.

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

Published in the proceedings of IEEE-RAS International Conference on Humanoid Robots (Humanoids), Cancun, Mexico, pp. 607-613, 2016.

While pHRI has been heavily investigated in the past [2], [3], [4], [6], including for bimanual robot operations [7], [5], [8], an approach that fulfils the given problem statement, to the best of our knowledge, has not yet been proposed.

In this paper we propose a bimanual control scheme where the robot constantly learns from the interaction with the human. During the initial learning of the task, the control is handled by the human operator. 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.

B. Related work

Related work can be separated into two distinct research topics: policy representations and bimanual robot control. Policy representations include learning by demonstration, variability distribution and non-uniform speed changes.

Policy representation with variability distribution

One of the important characteristics of motor skills is not only the path that the robot should follow, but also the variation of coordination patterns during the movement [9]. A promising paradigm to cope with such requirements is to encode the task as a dynamical system. Calinon et al. [10] proposed a representation based on Gaussian mixture model (GMM), which sequentially superimposes dynamical systems with varying full stiffness matrices. Another suitable representation is probabilistic motion primitives (ProMP) [11], which allows to encode behavior of a stochastic system. Although both representations allow speed scaling, they can not handle non-uniform speed scaling in its original form. In our approach we will apply the framework of dynamic motion primitives (DMP) [12] and its extension to cope with non-uniform speed changes [13].

Bimanual control

There are several approaches how to deal with the bimanual robot. Earlier control architectures [14], which exploit physical interaction between the robots are often enhanced with learning to synchronize motion of both arms [15], [16]. Using the concept of augmented object and virtual linkage model, Khatib [17] implemented a decentralized control between multiple mobile manipulators, applicable also to the bimanual control. Nowadays, most of the bimanual control architectures are based on the decomposition of the task into so called absolute and relative coordinates and underlying

internal and external forces [18]. We proposed an extension of this scheme [19] that properly decouples both subspaces – motion in absolute coordinates does not affects relative coordinates and vice versa. This property allows us to apply kinematic control to both subsystems independently. In this paper we further extend this approach to decouple also the dynamic properties of both subspaces.

The paper is organized as follows. In section II we outline the dynamic movement primitives framework along with the extension for speed profile encoding. Next, in section III we present the bimanual control architecture with our extensions. The main contribution is described in Section IV, where we combine the sub-aspects of the solution. An application of the proposed approach to bimanual physical human-robot cooperation is presented in section V. A short summary concludes the paper.

II. Learning by demonstration for Human-Robot Cooperation scheme

Human-robot cooperative task execution is usually initiated by demonstration, where kinesthetic guiding can be used to capture the desired robot motion. This process is followed by encoding trajectories in a more compact parametric representation, which allows us to store lengthy demonstration with a limited set of parameters. In our work we rely on motion representation with dynamical motion primitives (DMPs) [12], extended for Cartesian space movements [20]. In the original formulation of DMPs it is not possible to variate the speed of movement in a non-uniform way without changing the course of movement. In our approach we need to apply non-uniform speed changes, thus an appropriate trajectory representation is required. A suitable representation is Speed-Scaled Dynamic Motion Primitives (SS-DMPs), which we originally developed in [13].

First we acquire the initial movement policy in Cartesian coordinates by kinesthetic gudining

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

where $\mathbf{p}_k \in \mathbb{R}^3$ are the positions, $\mathbf{q}_k \in \mathrm{S}^3$ are the unit quaternions describing orientation, S^3 is a unit sphere in $\mathbb{R}^4$, k are trajectory samples, and T is the number of samples.

Next, we parameterize the demonstrated policy with a nonlinear dynamical system [12], [13], [20], which enables the encoding of general trajectories. For positions $\mathbf{p}$ and orientations $\mathbf{q}$, the trajectory can be specified by the following system of nonlinear differential equations

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

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

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

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

$$\nu(x)\tau\dot{x} = -\alpha x. \tag{6}$$

where x is the phase variable and $\boldsymbol{z}$ and $\boldsymbol{\eta}$ are auxiliary variables. The system (2) – (6) converges to the unique equilibrium point at $\mathbf{p} = \boldsymbol{g}_p$, $\boldsymbol{z} = 0$, $\mathbf{q} = \boldsymbol{g}_o$, $\boldsymbol{\eta} = 0$, and $x = 0$. Asterix $*$ denotes the quaternion multiplication and $\bar{\boldsymbol{q}}$ quaternion conjugation. See Eq. (28) for the definition of quaternion logarithm. The nonlinear forcing terms $\mathbf{f}_p(x)$ and $\mathbf{f}_o(x)$ 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 $\boldsymbol{g}_p$ and orientation $\boldsymbol{g}_o$. They are defined as linear combinations of radial basis functions (RBFs)

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

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

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

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

Compared to [20] and analogous to [13], we introduced the temporal scaling function $\nu(x)$ with which we can specify variations form the demonstrated speed profile. Similarly to the forcing terms (7) and (8), it is encoded as a linear combination of RBFs

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

where v_j are the corresponding weights.

In order to parameterize the demonstrated control policy with a DMP, the weights $\boldsymbol{w}_{i,p}$ $\boldsymbol{w}_{i,o}$ and v_j need to be calculated. The shape weights $\boldsymbol{w}_{i,p}$ and $\boldsymbol{w}_{i,o}$ are calculated by applying standard regression techniques [20], using the demonstrated trajectory (1). For ν we initially set $v_j = 0$, i. e. $\nu = 1$, meaning that the demonstrated speed profile is left unchanged.

III. Bimanual robot control

The control scheme used in our bimanual human-robot cooperation scheme is an extension of the previously proposed approach [21]. It allows the specification of the task in terms of geometrically meaningful motion variables, referred to as relative and absolute task motion of the cooperative system [7], [18]. In this work we further extend the control scheme in order to independently set also the dynamic properties in both subspaces.

First, we define the common base coordinate systems τ_b for both arms, as illustrated in Fig. 1. We use the notation where superscript $\{1, 2, b\}$ denotes that the given quantity is specified relative to the coordinate system τ_j[1], while the subscript i, $i \in \{1, 2\}$ denotes the arm of a bimanual system

[1] 1 relates to the tool center point frame of the first robot arm, 2 to the tool center point frame of the second robot arm, and b to the common base of the dual arm system.

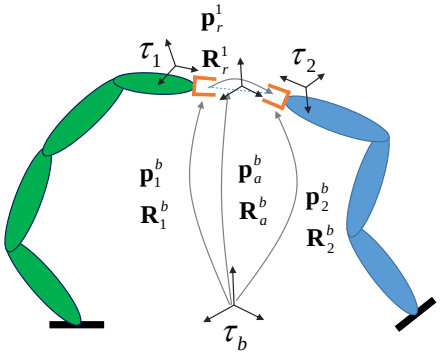

Fig. 1. Dual arm manipulator and the corresponding notation used in the paper.

and j, $j \in \{a, r\}$ denotes relative and absolute coordinates. Absolute and relative task coordinates are defined as

$$\mathbf{p}_a = \frac{1}{2}(\mathbf{p}_1^b + \mathbf{p}_2^b), \quad (11)$$

$$\mathbf{R}_a = \mathbf{R}_1^b \mathbf{R}_{\mathbf{k}_{21}^1}^1(\vartheta_{21}/2) \quad (12)$$

$$\mathbf{p}_r = {\mathbf{R}_1^b}^{\mathrm{T}}(\mathbf{p}_2^b - \mathbf{p}_1^b), \quad (13)$$

$$\mathbf{R}_r = \mathbf{R}_2^1 = {\mathbf{R}_1^b}^{\mathrm{T}}\mathbf{R}_2^b, \quad (14)$$

where $\mathbf{p}_a, \mathbf{p}_r \in \mathbb{R}^3$ are the position vectors and $\mathbf{R}_a, \mathbf{R}_r \in \mathbb{R}^{3\times 3}$ the rotational matrices. $\mathbf{k}_{21}^1$ and ϑ_{21} are the axis and angle that realize the rotation that describes the orientation of tool center point frame 2 with respect to frame 1. Note that our definition of relative coordinates is different from [21] because we multiply the difference vector $\mathbf{p}_2^b - \mathbf{p}_1^b$ by ${\mathbf{R}_1^b}^{\mathrm{T}}$. This makes the relative motion independent of the absolute motion. In quaternion notation, (12) – (14) are in the form

$$\mathbf{q}_a = \mathbf{q}_1^b * \mathbf{q}_{\mathbf{k}_{21}^1}^1(\vartheta_{21}/2), \quad (15)$$

$$(0, \mathbf{p}_r) = \bar{\mathbf{q}}_1^b * (0, \mathbf{p}_2^b - \mathbf{p}_1^b) * \mathbf{q}_1^b \quad (16)$$

$$\mathbf{q}_r = \mathbf{q}_2^1 = \bar{\mathbf{q}}_1^b * \mathbf{q}_2^b, \quad (17)$$

where the unit quaternions $\mathbf{q}_1^b$, $\mathbf{q}_2^b \in \mathbb{R}^4$ express the orientation of the tool center point of the first and the second robot in the common base coordinate frame τ_b, respectively. $\mathbf{q}_{\mathbf{k}_{21}^1}^1(\vartheta_{21}/2)$ denotes the unit quaternion corresponding to $\mathbf{R}_{\mathbf{k}_{21}^1}^1(\vartheta_{21}/2)$.

In order to control the robot, we have to map the desired relative and absolute task coordinates to the corresponding joint coordinates of both robots, denoted with $\boldsymbol{\theta} = [\boldsymbol{\theta}_1^{\mathrm{T}}, \boldsymbol{\theta}_2^{\mathrm{T}}]^{\mathrm{T}} \in \mathbb{R}^{N_1+N_2}$, where N_1 and N_2 is the number of joints of the first and the second robot, respectively. This transformation is obtained through relative and absolute geometrical Jacobian, which maps the corresponding translational and angular velocities to the joint velocities

$$\begin{bmatrix} \dot{\mathbf{p}}_r \\ \boldsymbol{\omega}_r \end{bmatrix} = \mathbf{J}_r\dot{\boldsymbol{\theta}}, \quad \begin{bmatrix} \dot{\mathbf{p}}_a \\ \boldsymbol{\omega}_a \end{bmatrix} = \mathbf{J}_a\dot{\boldsymbol{\theta}}. \quad (18)$$

As explained in [21], the absolute Jacobian can be computed as follows

$$\mathbf{J}_a = \begin{bmatrix} \frac{1}{2}\mathbf{J}_1 & \frac{1}{2}\mathbf{J}_2 \end{bmatrix}. \quad (19)$$

The derivation of relative coordinates (13) is more complex because they involve multiplication with ${\mathbf{R}_1^b}^{\mathrm{T}}$. Since these formulas are different from [21], we provide the full derivation. By differentiating (13) we obtain

$$\begin{aligned}\mathbf{J}_{r,p}\dot{\boldsymbol{\theta}} = \dot{\mathbf{p}}_r = {} & \mathbf{S}(-\boldsymbol{\omega}_1^b){\mathbf{R}_1^b}^{\mathrm{T}}(\mathbf{p}_2^b - \mathbf{p}_1^b) + {\mathbf{R}_1^b}^{\mathrm{T}}(\dot{\mathbf{p}}_2^b - \dot{\mathbf{p}}_1^b) \\ = {} & {\mathbf{R}_1^b}^{\mathrm{T}}(\mathbf{S}(-\mathbf{R}_1^b\boldsymbol{\omega}_1^b)(\mathbf{p}_2^b - \mathbf{p}_1^b) + \dot{\mathbf{p}}_2^b - \dot{\mathbf{p}}_1^b) \\ = {} & {\mathbf{R}_1^b}^{\mathrm{T}}(-\mathbf{S}(\boldsymbol{\omega}_1^b)(\mathbf{p}_2^b - \mathbf{p}_1^b) + \dot{\mathbf{p}}_2^b - \dot{\mathbf{p}}_1^b) \\ = {} & {\mathbf{R}_1^b}^{\mathrm{T}}(-\dot{\mathbf{p}}_1^b + \mathbf{S}(\mathbf{p}_2^b - \mathbf{p}_1^b)\boldsymbol{\omega}_1^b + \dot{\mathbf{p}}_2^b) \\ = {} & {\mathbf{R}_1^b}^{\mathrm{T}}(-\mathbf{J}_{1,p} + \mathbf{S}(\mathbf{p}_2^b - \mathbf{p}_1^b)\mathbf{J}_{1,\omega} + \mathbf{J}_{2,p})\dot{\boldsymbol{\theta}}, \end{aligned} \quad (20)$$

where $\mathbf{S}(\boldsymbol{x})$ is a skew-symmetric matrix constructed from vector $\boldsymbol{x} \in \mathbb{R}^3$. Here we used the fact that $\dot{\mathbf{R}} = \mathbf{S}(\boldsymbol{\omega})\mathbf{R}$, $\mathbf{S}(\mathbf{R}\boldsymbol{x})\,\mathbf{R}\boldsymbol{y} = \mathbf{R}\,(\mathbf{S}(\boldsymbol{x})\boldsymbol{y})$, $\mathbf{R}_1^b\boldsymbol{\omega}_1^b = \boldsymbol{\omega}_1^b$ (because rotation doesn't change the axis it is rotating about), and $\mathbf{S}(\boldsymbol{x})\boldsymbol{y} = -\mathbf{S}(\boldsymbol{y})\boldsymbol{x}$. Similarly, by differentiating $\mathbf{R}_r$ we obtain

$$\begin{aligned}\mathbf{S}(\boldsymbol{\omega}_r)\mathbf{R}_r = \dot{\mathbf{R}}_r = {} & \mathbf{S}(-\boldsymbol{\omega}_1^b){\mathbf{R}_1^b}^{\mathrm{T}}\mathbf{R}_2^b + {\mathbf{R}_1^b}^{\mathrm{T}}\mathbf{S}(\boldsymbol{\omega}_2^b)\mathbf{R}_2^b \\ = {} & \mathbf{S}(-\boldsymbol{\omega}_1^b)\mathbf{R}_r + {\mathbf{R}_1^b}^{\mathrm{T}}\mathbf{S}(\boldsymbol{\omega}_2^b)\mathbf{R}_1^b\mathbf{R}_r \\ = {} & (\mathbf{S}(-\boldsymbol{\omega}_1^b) + \mathbf{S}({\mathbf{R}_1^b}^{\mathrm{T}}\boldsymbol{\omega}_2^b))\mathbf{R}_r \\ = {} & \mathbf{S}({\mathbf{R}_1^b}^{\mathrm{T}}(-\boldsymbol{\omega}_1^b + \boldsymbol{\omega}_2^b))\mathbf{R}_r, \end{aligned} \quad (21)$$

where we used the fact that $\mathbf{R}\mathbf{S}(\boldsymbol{x})\mathbf{R}^{\mathrm{T}} = \mathbf{S}(\mathbf{R}\boldsymbol{x})$ and ${\mathbf{R}_1^b}^{\mathrm{T}}\boldsymbol{\omega}_1^b = \boldsymbol{\omega}_1^b$. It follows from (20) and (21) that

$$\mathbf{J}_r = \begin{bmatrix} \mathbf{R}_1^b & 0 \\ 0 & \mathbf{R}_1^b \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} -\mathbf{J}_{1,p} + \mathbf{S}(\mathbf{p}_2 - \mathbf{p}_1)\mathbf{J}_{1,\omega} & \mathbf{J}_{2,p} \\ -\mathbf{J}_{1,\omega} & \mathbf{J}_{2,\omega} \end{bmatrix}. \quad (22)$$

To control both absolute and relative coordinates, we define extended task coordinates $\mathcal{X}_e = [\mathbf{p}_a^{\mathrm{T}}, \mathbf{q}_a^{\mathrm{T}}, \mathbf{p}_r^{\mathrm{T}}, \mathbf{q}_r^{\mathrm{T}}]^{\mathrm{T}}$ and extended Jacobian $\mathbf{J}_e = [\mathbf{J}_a^{\mathrm{T}}, \mathbf{J}_r^{\mathrm{T}}]^{\mathrm{T}}$. We apply the well known impedance control law for kinematically redundant robots [22] in the form

$$\boldsymbol{\rho_c} = \mathbf{J}_e^{\mathrm{T}}\mathbf{M}_e\left(\ddot{\mathcal{X}}_c - \dot{\mathbf{J}}_e\dot{\boldsymbol{\theta}}\right) + \mathbf{H}_e\mathbf{N}\boldsymbol{\theta}_0 + \mathbf{h}_e, \quad (23)$$

where $\boldsymbol{\rho_c}$ is the control torque input for the motors, $\mathbf{M}_e = (\mathbf{J}_e\mathbf{H}_e^{-1}\mathbf{J}_e^{\mathrm{T}})^{-1}$ denotes the positive definite matrix of inertia expressed in rotated operational space coordinates, $\mathbf{H}_e = \begin{bmatrix} \mathbf{H}_1 & 0 \\ 0 & \mathbf{H}_2 \end{bmatrix}$ is the extended joint space inertia matrix composed of inertia matrices of both arms, $\mathbf{N} = \mathbf{I} - \bar{\mathbf{J}}_e\mathbf{J}_e$ is the null space matrix that maps manipulator joint velocities and manipulator torques to the null space of $\mathbf{J}_e$. $\mathbf{h}_e = [\mathbf{h}_1^{\mathrm{T}}, \mathbf{h}_2^{\mathrm{T}}]^{\mathrm{T}}$ is a vector that compensates for Coriolis, radial and gravity forces of the manipulator and the load. $\bar{\mathbf{J}}_e = \mathbf{H}_e^{-1}\mathbf{J}_e^{\mathrm{T}}\mathbf{M}$ denotes the inertia weighted pseudo-inverse of $\mathbf{J}_e$. $\boldsymbol{\theta}_0 \in \mathbb{R}^{N_1+N_2}$ is a vector that defines the null space motion. The task command input $\ddot{\mathcal{X}}_c = [\ddot{\mathbf{p}}_{ac}^{\mathrm{T}}, \dot{\boldsymbol{\omega}}_{ac}^{\mathrm{T}}, \ddot{\mathbf{p}}_{rc}^{\mathrm{T}}, \dot{\boldsymbol{\omega}}_{rc}^{\mathrm{T}}]^{\mathrm{T}}$ is chosen as

$$\ddot{\mathbf{p}}_{ac} = \ddot{\mathbf{p}}_{ad} + \mathbf{D}_{ap}\dot{\mathbf{e}}_{ap} + \mathbf{K}_{ap}\mathbf{e}_{ap}, \quad (24)$$

$$\dot{\boldsymbol{\omega}}_{ac} = \dot{\boldsymbol{\omega}}_{ad} + \mathbf{D}_{aq}\mathbf{e}_{a\omega} + \mathbf{K}_{aq}\mathbf{e}_{aq}, \quad (25)$$

$$\ddot{\mathbf{p}}_{rc} = \ddot{\mathbf{p}}_{rd} + \mathbf{D}_{rp}\dot{\mathbf{e}}_{rp} + \mathbf{K}_{rp}\mathbf{e}_{rp}, \quad (26)$$

$$\dot{\boldsymbol{\omega}}_{rc} = \dot{\boldsymbol{\omega}}_{rd} + \mathbf{D}_{rq}\mathbf{e}_{r\omega} + \mathbf{K}_{rq}\mathbf{e}_{rq}, \quad (27)$$

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

$$\log(\mathbf{q}) = \log(v, \mathbf{u}) = \begin{cases} \arccos(v)\dfrac{\mathbf{u}}{\|\mathbf{u}\|}, & \mathbf{u} \neq 0 \\ [0,0,0]^{\mathrm{T}}, & \text{otherwise} \end{cases}. \quad (28)$$

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

$$\exp(\mathbf{r}) = \begin{cases} \cos(\|\mathbf{r}\|) + \sin(\|\mathbf{r}\|)\dfrac{\mathbf{r}}{\|\mathbf{r}\|}, & \mathbf{r} \neq 0 \\ 1 + [0,0,0]^{\mathrm{T}}, & \text{otherwise} \end{cases}. \quad (29)$$

Subscripts $(.)_a$ and $(.)_r$ stand for absolute and relative coordinates, respectively. Subscript $(.)_d$ denotes the desired values and variables without the subscript denote the current entities calculated from the robot joints. $\mathbf{D}_{ap}$, $\mathbf{D}_{aq}$, $\mathbf{D}_{rp}$, $\mathbf{D}_{rq}$, $\mathbf{K}_{ap}$, $\mathbf{K}_{aq}$, $\mathbf{K}_{rp}$, $\mathbf{K}_{rq}$ are diagonal, positive definite positional damping, rotational damping, positional stiffness and rotational stiffness matrices for absolute and relative part of motion, respectively. They specify the properties of the controller in the Cartesian coordinate system.

In this study we are not interested in the null-space optimization. However, the null space has to be controlled in order to prevent non-conservative motion. One appropriate choice is given in the form $\boldsymbol{\theta}_0 = -\mathbf{K}_n\dot{\boldsymbol{\theta}}$ [22] , where $\mathbf{K}_n \in \mathbb{R}^{(N_1+N_2)\times(N_1+N_2)}$ is a positive definite diagonal gain matrix.

IV. Human-robot cooperation scheme

In the proposed system, the human operator first demonstrates the desired cooperative humanoid robot motion by kinesthetically guiding the robot arms. The demonstrated motion is transformed into relative and absolute coordinates as described in Section III and encoded by SS-DMPs $(\mathbf{p}_a, \mathbf{q}_a, \mathbf{p}_r, \mathbf{q}_r)$ as explained in Section II. The demonstration of the absolute motion is typically performed slower than actually desired because 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 co-worker is allowed also to modify the course of motion.

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 [6], 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 a behavior, the bimanual robot system has to be compliant in the direction of motion. We apply SS-DMPs to enable non-uniform speed variations without affecting the course of motion. To the best of our knowledge, none of the previously proposed adaptation algorithms can simultaneously address these issues.

A. Trajectory adaptation

In the proposed approach we deal with the adaptation of absolute robot motion, whereas the relative motion of the two arms is left unchanged. Since the motion is performed in collaboration with a human and the robot is initially compliant in all directions, the commanded trajectory $\mathbf{p}_{a,l}$ in task repetition cycle l is not the same as the actually executed trajectory $\mathbf{p}_{a,m}$ due to the input of a human. Here l is the index of the task repetition, referred to also as learning cycle. The proposed adaptation algorithm updates the desired trajectory $(\mathbf{p}_{a,l}(x), \mathbf{q}_{a,l}(x))$, $l = 1, \ldots, L$, where the initial SS-DMP is taken from human demonstration $\mathbf{p}_{a,1} = \mathbf{p}_a$, $\mathbf{q}_{a,1} = \mathbf{q}_a$, and calculates its variance after each task execution.

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

$$\mathbf{p}_{a,l+1}(x) = \zeta\Delta\mathbf{p}(x) + \mathbf{p}_{a,l}(x), \quad (30)$$

$$\boldsymbol{\Sigma}_{ap,l+1}(x) = (1-\zeta)\boldsymbol{\Sigma}_{ap,l}(x) + \zeta\Delta\mathbf{p}(x)\Delta\mathbf{p}(x)^{\mathrm{T}}, \quad (31)$$

$$\Delta\mathbf{p}(x) = \mathbf{p}_{a,m}(x) - \mathbf{p}_{a,l}(x),$$

where $\mathbf{p}_{a,m}(x)$ denotes the measured absolute position of the robot, $\boldsymbol{\Sigma}_{ap,l}(x)$ is the current cycle covariance of $\mathbf{p}_{a,l}(x)$, all computed at phase x, and $0 \leq \zeta \leq 1$ is the weighting factor that defines the learning speed. If we set $\zeta = 1$, the updated absolute trajectory $\mathbf{p}_{a,l+1}$ is equal to the measured trajectory $\mathbf{p}_{a,m}$. On the other hand, if we set $\zeta = 0$, the absolute trajectory $\mathbf{p}_{a,l+1}$ does not change and the system stops learning. After each learning cycle, the updated trajectory $\mathbf{p}_{a,l+1}$ is encoded into SS-DMP. It is used as command 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 coeffcients of covariance matrix $\boldsymbol{\Sigma}_{ap,l+1}$ are approximated with a linear combination of radial basis functions (RBFs).

Eq. (30) cannot be used for orientation trajectories. Instead we apply the following update rule

$$\mathbf{q}_{a,l+1}(x) = \exp\left(\zeta\frac{\boldsymbol{\omega}(x)}{2}\right) * \mathbf{q}_{a,l}(x), \quad (32)$$

$$\boldsymbol{\omega}(x) = 2\log(\mathbf{q}_{a,m}(x)) * \overline{\mathbf{q}}_{a,l}(x).$$

As in this work we do not need the variance of orientation trajectories, we skip describing its estimation here.

B. Stiffness adaptation

To improve the ease of adaptation we dynamically set the desired stiffness of the robot. It is well known that the precision and speed of human motion are related – to be precise, humans reduce their speed [23]. While Calinon et al. [6] 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 coordinate frame with z coordinate specified in the desired direction of motion, i. e. $\dot{\mathbf{p}}_{a,l}$, and the other two coordinates orthogonal to it, as illustrated in Fig. 2. This matrix can be obtained by forming the Frenet-Serret frame [24] at each sampling time. The Frenet-Serret frame consists of three orthogonal direction defined by the path's tangent (direction of motion), normal, and binormal. We obtain the following expression for $\mathbf{R}_p$

$$\mathbf{R}_p = \begin{bmatrix} \mathbf{n} & \mathbf{b} & \mathbf{t} \end{bmatrix}, \tag{33}$$

$$\mathbf{t} = \frac{\dot{\mathbf{p}}_{a,l}}{\|\dot{\mathbf{p}}_{a,l}\|},\ \mathbf{b} = \frac{\dot{\mathbf{p}}_{a,l} \times \ddot{\mathbf{p}}_{a_l}}{\|\dot{\mathbf{p}}_{a,l} \times \ddot{\mathbf{p}}_{a,l}\|},\ \mathbf{n} = \mathbf{b} \times \mathbf{t}.$$

Note that the absolute velocity $\dot{\mathbf{p}}_{a,l}$ and acceleration $\ddot{\mathbf{p}}_{a,l}$ are provided by DMP integration at every phase x, which ensures smoothness. $\|\dot{\mathbf{p}}_{a,l}\| < \varepsilon$ or $\|\dot{\mathbf{p}}_{a,l} \times \ddot{\mathbf{p}}_{a,l}\| < \varepsilon$, 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. We also compute the robot's speed in absolute coordinates, i. e. $v_a = \|\dot{\mathbf{p}}_{a,l}\|$ and define scalar v_0, which specifies the threshold between the low and high speed. The appropriate control gain $\mathbf{K}_{ap}$ at each sampling time is computed as follows

$$\mathbf{K}_{ap}(x) = \mathbf{R}_p^{\mathrm{T}} \begin{bmatrix} \frac{k_{a,o}\rho}{\Sigma_{xx}+\epsilon} & 0 & 0 \\ 0 & \frac{k_{a,o}\rho}{\Sigma_{yy}+\epsilon} & 0 \\ 0 & 0 & k_{a,z} \end{bmatrix} \mathbf{R}_p, \tag{34}$$

$$\mathbf{\Sigma} = \mathbf{R}_p \mathbf{\Sigma}_{ap,l}(x) \mathbf{R}_p^{\mathrm{T}}, \tag{35}$$

$$\rho = \gamma_1 \left(1 + \tanh\left(\frac{v_a - v_0}{\gamma_3}\right)\right) + \gamma_2, \tag{36}$$

where Σ_{xx} and Σ_{yy} are the first and second diagonal coefficient of $\mathbf{\Sigma}$, respectively. $\epsilon > 0$ is an empirically chosen constant which sets the upper bound for the controller gain. $k_{a,z}$ and $k_{a,o}$ are the gain constants in the direction of motion and orthogonal to it, respectively. $\gamma_1, \gamma_2, \gamma_3 > 0$ respectively determine the range, lower bound and the speed of transition between the lower and upper bound of the switching function defined by $\tanh$. The initial value for covariance matrix $\mathbf{\Sigma}_{ap,1}$ 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. The derivative part of the gain is computed as below

$$\mathbf{D}_{ap}(x) = 2\sqrt{\mathbf{K}_{ap}(x)} \tag{37}$$

for a critically damped response.

By choosing a constantly low value for $k_{a,z}$ in $\mathbf{K}_{ap}$, 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.

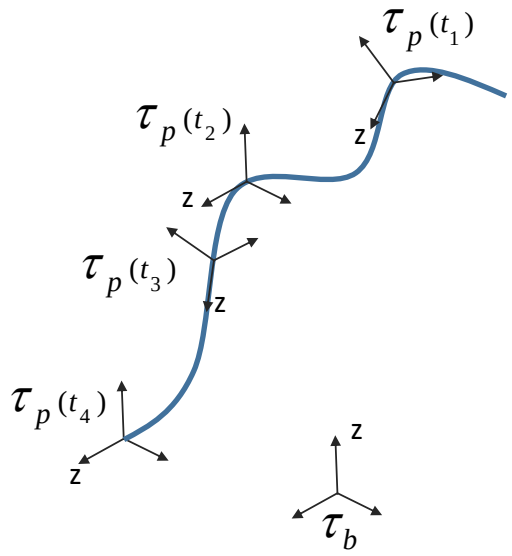

Fig. 2. Operational space τ_p is defined by path orientation.

C. Speed adaptation

The full specification of SS-DMPs requires us to set also the speed scaling factor ν in each sampling interval. Lets

Algorithm 1: Human-robot cooperation algorithm

Record $\{\mathbf{p}_a(k), \mathbf{q}_a(k), \mathbf{p}_r(k), \mathbf{q}_r(k), t_k\}_{k=1}^{T}$ using kinesthetic guiding and calculate SS-DMP parameters from the demonstrated data $(\mathbf{p}_{a,1}, \mathbf{q}_{a,1}, \mathbf{p}_r, \mathbf{q}_r)$
Initialize gains $k_{a,o}, k_{a,z}$ and set initial covariance matrices $\mathbf{\Sigma}_{ap,1} = s_0\mathbf{I}$. Approximate coefficients of $\mathbf{\Sigma}_{ap,1}$ with a linear combination of RBFs.
set $l = 1$
while *cooperating* **do**
  set initial phase $x = 1$
  while $x \leq x_{min}$ **do**
    integrate SS-DMP to obtain $\mathbf{p}_{a,l}(x), \mathbf{q}_{a,l}(x), \mathbf{p}_r(x), \mathbf{q}_r(x)$ as well as their velocities and accelerations
    calculate path rotation $\mathbf{R}_p(x)$ using (33) and speed $v_a(x)$
    calculate $\mathbf{K}_{ap}(x)$ and $\mathbf{D}_{ap}(x)$ using (34) and (37), respectively
    execute control law (23) – (27) with $\mathbf{p}_{a,l}(x), \mathbf{q}_{a,l}(x), \mathbf{p}_r(x), \mathbf{q}_r(x)$ as desired trajectories
    sample new trajectories $\mathbf{p}_{a,l+1}(x), \mathbf{q}_{a,l+1}(x)$, covariance matrices $\mathbf{\Sigma}_{ap,l+1}(x)$, and speed scaling factor ν_{l+1}, all at phase x, using (30) – (32), (38)
  calculate SS-DMP parameters of $\mathbf{p}_{a,l+1}, \mathbf{q}_{a,l+1}$, including ν_{l+1}
  approximate coefficients of $\mathbf{\Sigma}_{ap,l+1}$ with linear combinations of RBFs
  set $l = l + 1$

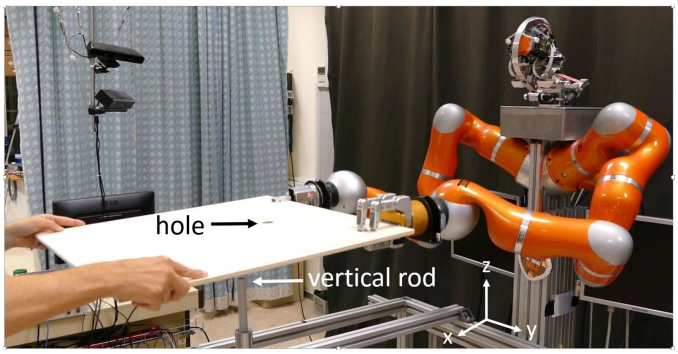

Fig. 3. Cooperating humanoid robot and human.

define the tracking error $e_{zp} = [0\,0\,1]\,\mathbf{R}_p\mathbf{e}_{ap}$, $\mathbf{e}_{ap} = \mathbf{p}_{a,l} - \mathbf{p}_{a,m}$, which is the z component of the tracking error in path operational space. This error then determines the speed scaling factor, calculated as

$$\nu_{l+1}(x) = \exp(\lambda e_{zp})\nu_l(x), \tag{38}$$

where $\lambda > 0$ is an appropriately set constant. With this equation we speed up or slow down the trajectory. Note that negative e_{zp} means that the actual robot position is anticipating the desired trajectory. In this case we have to speed up the desired trajectory, and vice versa, with positive e_{zp} we slow down the desired trajectory. After sampling we compute the coefficients v_i that specify $\nu(x)$ defined as in (10). In this way we achieve faster convergence towards the desired trajectory in the direction of motion.

The learning algorithm is summarized in Algorithm 1.

V. Experimental evaluation

The proposed human-robot cooperation scheme was experimentally verified using a humanoid robot composed of two 7 degree of freedom Kuka LWR-4 robot arms equipped with Barret hands and controlled with Fast Research Interface (FRI). The task of the robot was to learn how to cooperate with the human while transporting a rigid plate from the initial point to the final point and 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. 3.

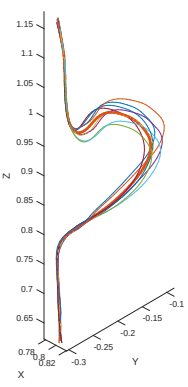

Fig. 4. 3-D plot of trajectories of absolute coordinates before the vertical rod displacement. The thick line shows the final learned trajectory.

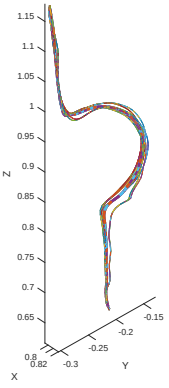

Fig. 5. 3-D plot of trajectories of absolute coordinates after the vertical rod displacement.

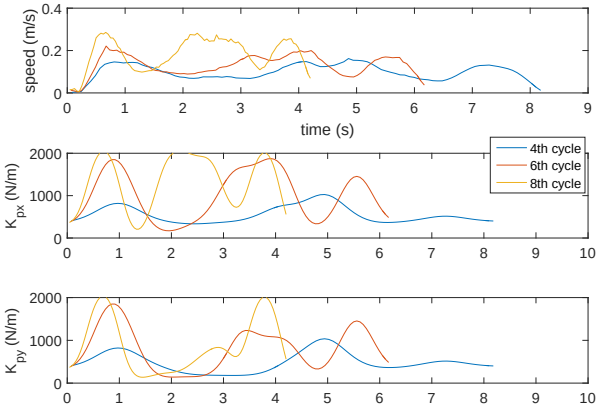

Fig. 6. Evolution of speed v_a and the learned gains $\mathbf{R}_p\mathbf{K}_{ap}\mathbf{R}_p^{\mathrm{T}}$ before the vertical rod displacement.

The initial gains $\mathbf{K}_{rp}$ and $\mathbf{K}_{ap}$ were set to $800\mathbf{I}$ N/m and $100\mathbf{I}$ N/m, respectively. Thus, the system was initially stiff in relative coordinates and very compliant in absolute coordinates. Compliance of the relative coordinates was not adapted. The speed threshold v_0, where the system starts adjusting the stiffness, was empirically set to 0.1 m/s.

After the initial task demonstration, we performed 8 cooperative repetitions of the task. The learning factor ζ was set to 0.4. Fig. 4 shows the 3-D plot of the trajectories $\mathbf{p}_{a,l}$. The execution speed v_a and the learned gains $\mathbf{R}_p\mathbf{K}_{ap}\mathbf{R}_p^{\mathrm{T}}$ during subsequent executions are shown in Fig. 6. After 8 repetitions we displaced the vertical rod for -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 guiding the robot to the new position. In a few repetitions the system learned the new task and reset the high stiffness gains. This enabled the human operator to accomplish the task by allowing the robot to guide him.

Fig. 5 shows the 3-D plot of trajectories $\mathbf{p}_{a,l}$ after the displacement. Execution speed v_a and controller gains $\mathbf{R}_p\mathbf{K}_{ap}\mathbf{R}_p^{\mathrm{T}}$ are displayed in Fig. 7.

VI. Conclusions

In this work we proposed a new human-robot cooperation scheme, where a humanoid robot and a human collaborate in manipulating an object. The developed algorithm is based on the previously proposed SS-DMPs [13] and extended cooperative task approach for bimanual robots. There are several novelties in the proposed approach:

- Speed-scaled DMPs in Cartesian space have been introduced.

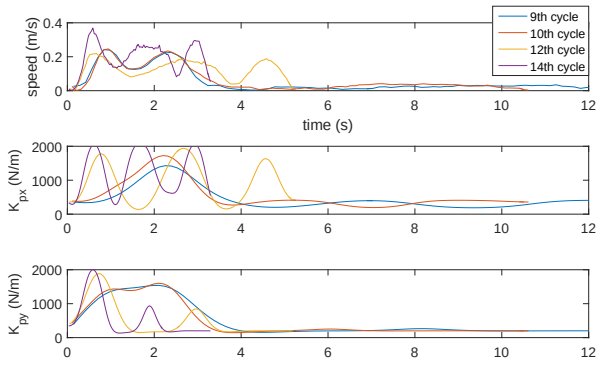

Fig. 7. Evolution of speed v_a and the learned gains $\mathbf{R}_p\mathbf{K}_{ap}\mathbf{R}_p^{\mathrm{T}}$ after the vertical rod displacement.

- 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.

ACKNOWLEDGMENT

This work was supported by EU Horizon 2020 Programme grant no. 680431, ReconCell and by the Slovenian Research Agency grant J2-7360, "Learning and autonomous adaptation of dual arm assembly and service tasks".

REFERENCES

[1] B. V. Adorno, A. P. L. Bó, P. Fraisse, and P. Poignet, "Towards a cooperative framework for interactive manipulation involving a human and a humanoid," *International Conference on Robotics and Automation (ICRA)*, pp. 3777–3783, 2011.

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

[3] R. Soyama, S. Ishii, and A. Fukase, "Selectable operating interfaces of the meal-assistance device "my spoon"," in *Rehabilitation*, Z. Bien and D. Stefanov, Eds. Springer Berlin / Heidelberg, 2004, pp. 155–163.

[4] H. I. Krebs, N. Hogan, M. L. Aisen, and B. T. Volpe, "Robot-aided neurorehabilitation," *IEEE Transactions on Rehabilitation Engineering*, vol. 6, no. December, pp. 75–87, 1998.

[5] A. Mortl, M. Lawitzky, A. Kucukyilmaz, M. Sezgin, C. Basdogan, and S. Hirche, "The role of roles: Physical cooperation between humans and robots," *The International Journal of Robotics Research*, vol. 31, no. 13, pp. 1656–1674, 2012.

[6] 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.

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

[8] H. A. Park and C. S. G. Lee, "Extended cooperative task space for manipulation tasks of humanoid robots," in *IEEE International Conference on Robotics and Automation (ICRA)*, Seattle, WA, 2015, pp. 6088–6093.

[9] S. Calinon, Z. Li, T. Alizadeh, N. G. Tsagarakis, and D. G. Caldwell, "Statistical dynamical systems for skills acquisition in humanoids," in *12th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, Osaka, Japan, 2012, pp. 323–329.

[10] 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.

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

[12] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal, "Dynamical movement primitives: Learning attractor models for motor behaviors," *Neural Computation*, vol. 25, no. 2, pp. 328–73, 2013.

[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] T. Tarn, A. Bejczy, and X. Yun, "Coordinated control of two robot arms," *Proceedings 1984 IEEE International Conference on Robotics and Automation*, vol. 3, pp. 468–473, 1986.

[15] E. Gribovskaya and A. Billard, "Combining dynamical systems control and programming by demonstration for teaching discrete bimanual coordination tasks to a humanoid robot," in *ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, Amsterdam, 2008, pp. 33–40.

[16] A. Gams, B. Nemec, A. J. Ijspeert, and A. Ude, "Coupling Movement Primitives: Interaction With the Environment and Bimanual Tasks," *IEEE Transactions on Robotics*, pp. 1–15, 2014.

[17] O. Khatib, "Robots in Human Environments: Basic Autonomous Capabilities," *The International Journal of Robotics Research*, vol. 18, pp. 684–696, 1999.

[18] F. Caccavale, P. Chiacchio, and S. Chiaverini, "Task-space regulation of cooperative manipulators," *Automatica*, vol. 36, pp. 879–887, 2000.

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

[20] 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.

[21] P. Chiacchio, S. Chiaverini, and B. Siciliano, "Direct and inverse kinematics for coordinated motion tasks of a two-manipulator system," *Journal of Dynamic Systems, Measurement, and Control*, vol. 118, no. 4, pp. 691–697, 1996.

[22] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *IEEE Journal of Robotics and Automation*, vol. 3, pp. 43–53, 1987.

[23] P. M. Fitts, "The information capacity of the human motor system in controlling the amplitude of movement." *Journal of Experimental Psychology*, vol. 47, no. 6, pp. 381–391, 1954.

[24] G. Chiaverini, S. Oriolo and W. I. D., "Chapter 11: Kinematically redundant manipulators," in *Springer Handbook of Robotics*. Springer Berlin Heidelberg, 2008, pp. 245–268.