A Virtual Mechanism Approach for Exploiting Functional Redundancy in Finishing Operations

Bojan Nemec, *Member, IEEE*, Kenichi Yasuda, and Aleš Ude, *Member, IEEE*

***Abstract*— We propose a new approach to programming by the demonstration of finishing operations. Such operations can be carried out by industrial robots in multiple ways because an industrial robot is typically functionally redundant with respect to a finishing task. In the proposed system, a human expert demonstrates a finishing operation, and the demonstrated motion is recorded in the Cartesian space. The robot's kinematic model is augmented with a virtual mechanism, which is defined according to the applied finishing tool. This way, the kinematic model is expanded with additional degrees of freedom that can be exploited to compute the optimal joint space motion of the robot without altering the essential aspects of the Cartesian space task execution as demonstrated by the human expert. Finishing operations, such as polishing and grinding, occur in contact with the treated workpiece. Since information about the contact point position is needed to control the robot during the operation, we have developed a novel approach for accurate estimation of contact points using the measured forces and torques. Finally, we applied iterative learning control to refine the demonstrated operations and compensate for inaccurate calibration and different dynamics of the robot and human demonstrator. The proposed method was verified on real robots and real polishing and grinding tasks.**

***Note to Practitioners*—This work was motivated by the need for automation of finishing operations, such as polishing and grinding, on contemporary industrial robots. Existing approaches are both too complex and too time-consuming to be applied in flexible and small-scale production, which often requires the frequent deployment of new applications. Our approach is based on programming by demonstration and enables the programming of finishing operations also for users who are not specialists in robot programming. Programming by demonstration is especially useful for teaching finishing operations because it enables the transfer of expert knowledge about finishing skills to robots without providing lengthy task descriptions or manual coding. Besides the human demonstration of the desired operation, the proposed approach also requires the availability of the kinematic model for the machine tool applied to carry out the finishing operation. We provide several practical examples of grinding and polishing tools and how to integrate them into our approach. Another feature of the proposed system is that user demonstrations of finishing operations can be transferred between different combinations of robots and machine tools.**

***Index Terms*— Functional redundancy resolution, industrial robots, programming by demonstration (PbD), virtual mechanism.**

Manuscript received May 13, 2020; revised June 28, 2020 and August 22, 2020; accepted October 14, 2020. Date of publication November 2, 2020; date of current version October 6, 2021. This article was recommended for publication by Associate Editor X. Wu and Editor C. Seatzu upon evaluation of the reviewers' comments. This work was supported in part by the Program Group P2-0076 Automation, Robotics, and Biocybernetics funded by the Slovenian Research Agency, in part by the EU Horizon 2020 Innovation Action TRINITY under Grant 825196, and in part by the GOSTOP Programme, co-financed by Slovenia and EU through ERDF under Grant C3330-16-529000. *(Corresponding author: Bojan Nemec.)*

Bojan Nemec is with the Humanoid and Cognitive Robotics Laboratory, Department of Automatics, Biocybernetics and Robotics, Jožef Stefan Institute, 1000 Ljubljana, Slovenia (e-mail: bojan.nemec@ijs.si).

Kenichi Yasuda is with the Robotics Technology Group, Yaskawa Electric Corporation, Kitakyushu 803-8530, Japan (e-mail: kenichi.yasuda@yaskawa.co.jp).

Aleš Ude is with the Humanoid and Cognitive Robotics Laboratory, Department of Automatics, Biocybernetics and Robotics, Jožef Stefan Institute, 1000 Ljubljana, Slovenia, and also with the Faculty of Electrical Engineering, University of Ljubljana, 1000 Ljubljana, Slovenia (e-mail: ales.ude@ijs.si).

This article has supplementary downloadable material available at https://ieeexplore.ieee.org, provided by the authors.

Color versions of one or more of the figures in this article are available online at https://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TASE.2020.3032075

I. Introduction

NOWADAYS, we can no longer imagine large-scale industrial production without the use of robots. While robots are predominantly used in the automotive, home appliance, chemical, and manufacturing industries, their use is still lagging behind in small-scale and artisan manufacturing [1], [2]. One of the main reasons for this is the often excessive programming effort required to program new robot tasks. Programming by demonstration (PbD) is a promising technology that enables end-users to teach new robot tasks without manual programming. Instead of requiring users to decompose and program the desired task analytically, an appropriate policy is derived from observations of human task performance. This way, the efficiency and ease of preparation of new robot tasks can be improved [3].

Finishing operations such as polishing and grinding are considered hard to automate, especially using classical robot programming approaches. The reason for this is that they entail a sophisticated trajectory and force control. Moreover, batch sizes requiring finishing operations are usually not very large, thus making it necessary to reprogram the production cell many times [4]. For these reasons, the finishing operations are still predominately performed by skilled operators, who aim to modify the treated object's surface in multiple passes involving complex movements. Even skilled operators have difficulties to explain how they perform the desired tasks. Hence, PbD is a suitable approach to program such tasks.

When workpieces are treated with tools, it is usually not important, where, in 3-D space, the operation takes place. What matters is the relative orientation between the tool and the workpiece and the applied forces. Thus, even though the exact replication of the demonstrated motion might require

all the degrees of freedom (DOF) of a robot, replicating the relevant part of the task, i.e., the relative orientation between the robot and the workpiece, usually requires fewer DOFs, thus making the robot functionally redundant [5]. In the case of functional redundancy, the Jacobian is often a full rank matrix, i.e., its null space does not exist, which prevents the application of standard redundancy resolution schemes [6].

Direct copying of finishing operations from human demonstrators to industrial robots is—in most cases—not successful due to the different kinematic and dynamic capabilities of humans and industrial robots. This is called the correspondence problem [7]. In this article, we propose a novel approach to address this problem by applying a virtual mechanism methodology. We model machine tools as virtual robotic mechanisms and treat the robot and the machine tool as a bimanual robot. Even though the degrees of freedom of the virtual mechanism are not controlled, they are making the bimanual system redundant with respect to the task, thus providing additional DOFs that can be used to optimize the real robot motion.

Another significant but not yet solved problem is how to reuse previously learned policies and apply them to an arbitrary combination of industrial robots and machine tools. Our approach supports performing the same demonstrated task with different combinations of robots and machine tools without changing the Cartesian space task execution.

Finally, the developed framework supports iterative task refinement during the task execution to account for possible calibration errors and compensate for the different dynamical properties of a robot and human demonstrator. We deal with these differences by applying the iterative learning control (ILC) paradigm [8] to the bimanual system involving the robot and the tool.

A. State of the Art in Automation of Finishing Operations

Much effort was made to automate tasks, such as polishing and grinding with industrial robots [9], [10]. Early attempts involved simple point-to-point programming and optimization using trial and error approaches [11], [12]. Besides being time-consuming, these approaches cannot transfer the learned policies to different robots and tools. More efficient programming relies on transferring the policy from a skilled operator, which comprises both kinematic motion and the applied forces and torques. Various motion capture systems can be used to acquire such policies in manufacturing tasks, e.g., 3-D optical trackers [13]–[15] or kinesthetic guidance [3].

Forces and torques exerted on the workpiece during the finishing operation are important parameters of the finishing policy. Typically, they are captured during the policy demonstration using universal force–torque sensors, which can be incorporated either in the polishing/grinding machine [13] or in special sensorized tools [16]. Yet, another approach is to perform task demonstrations in a virtual environment with haptic devices, where the virtual environment is built either using CAD models or online using additional sensors, such as laser scanners [17]–[19]. However, finishing processes are generally very complex and require precise tuning of many process parameters in order to achieve the desired final quality.

In a recent research effort, Ng *et al.* [13], [18], [20] proposed an advanced programming by demonstration approach for finishing operations. It is based on recognizing the operator's skills using key process variables and generating the appropriate robot motion from a predefined skills library, optimized for the robot, rather than transferring the operator's motion patterns directly.

The analysis of previous research on automation of finishing operations shows that, unlike our work, none of them utilizes the virtual mechanism approach. Consequently, these approaches cannot deal with the correspondence problem as effectively as the methodology proposed in this article.

An early version of our work was presented in a conference paper [21]. Here, we introduce several significant theoretical and practical improvements to this initial approach. We optimized the performance of the bimanual robot by appropriately weighting the joints of the virtual mechanism in the redundancy resolution scheme and present an improved algorithm for computing the joint motion of the virtual mechanism and the robot. A convergence proof for the algorithm used to determine the contact point between the tool and the workpiece has also been provided. Finally, we address the correspondence problem of PbD and explain how to accomplish policy transfer between different tools and robots.

II. Policy Demonstration Measurement System

The first step in our approach is the demonstration of the task policy, where the required movements, forces, and torques are demonstrated by a skilled operator. To be able to capture both the movements and the arising forces and torques, we built a custom learning device using a passive six-axis mechanical digitizer (MicroScribe G2LX6). The end link of the digitizer is attached to a specially designed handle, which incorporates a universal force–torque sensor (Dyn Pick WDF-6M200-3). The handle allows the operator to move the attached workpiece along with the machine tool, which is placed at the fixed position in space. This setup is shown in Fig. 1, where the operator demonstrates the polishing of a faucet handle.

The demonstrated motion is captured as a set of points in the Cartesian space calculated from the digitizer joint angles

$$\mathcal{G} = \{\mathbf{p}_{1,k}, \mathbf{q}_{1,k}, t_k\}_{k=1}^{T}. \tag{1}$$

Here, $\mathbf{p}_{1,k} \in \mathbb{R}^3$ are the tool positions and $\mathbf{q}_{1,k} \in \mathrm{S}^3$ are the unit quaternions describing the tool orientation, with S^3 denoting the unit sphere in $\mathbb{R}^4$. T is the number of samples, and t_k is the time at sample k. The force–torque sensor built into the handle is used to capture the applied forces along the demonstrated trajectory

$$\mathcal{F} = \{\mathbf{F}_k, \mathbf{M}_k, t_k\}_{k=1}^{T} \tag{2}$$

where $\mathbf{F}_k, \mathbf{M}_k \in \mathbb{R}^3$ are the measured forces and torques at times t_k. They are measured in the tool coordinate frame, which must be aligned with the force–torque sensor coordinate system, but we transform the measurements to the robot base coordinate frame as most of the calculations are performed in this frame.

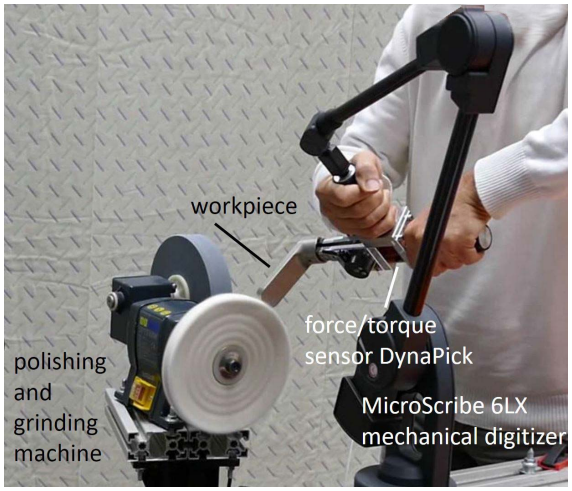


Fig. 1. Learning by demonstration setup for polishing/grinding. In some cases, it is necessary to wear protective gloves and mask during the demonstration process.

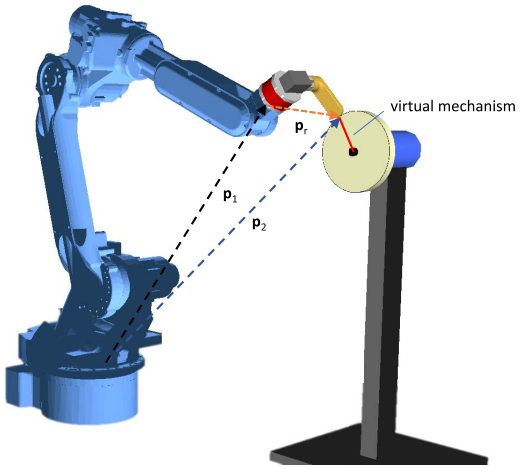


Fig. 2. Polishing machine and the robot are modeled as a bimanual system. The common coordinate frame is placed at the robot's base. $\mathbf{p}_1$ and $\mathbf{p}_2$ are the vectors, respectively, describing the position of the robot and the virtual mechanism. $\mathbf{p}_r$ denotes the relative coordinates that define the task.

III. Describing the Task With a Bimanual Robot and Virtual Mechanism

The success of a finishing operation does not depend on where precisely the contact between the machine tool and the workpiece occurs; it only matters that the workpiece touches the tool with the demonstrated orientation. Although the robot needs 6 DOFs to exactly replicate the demonstrated motion, fewer DOFs are needed to keep the contact of the workpiece with the tool at the desired orientation. Thus, the robot is functionally redundant. This redundancy arises from the shape of the tool because the workpiece can touch the tool at any position on the tool's surface.

To exploit this situation, we propose to model the machine tool as a serial kinematic chain, here called a virtual mechanism. The joints of the virtual mechanism define the point where the treated object held by a robot touches the machine tool. In order to represent the robot and the virtual mechanism in a unified system, we describe the resulting kinematic system as a bimanual robot [22] consisting of the real robot and the virtual mechanism. Only the relative coordinates of the bimanual setup are important for the accomplishment of the desired task. Fig. 2 shows an example where the robot and the polishing machine are modeled as a bimanual system.

With this setup, the overall system becomes kinematically redundant in a standard definition of redundancy even though the robot itself is not. Thus, standard redundancy resolution schemes [23] can be applied to the resulting bimanual system, where the redundant DOFs can be used to optimize the robot joint space motion, e.g., by minimizing the joint velocities or by avoiding the joint limits, singularities, and collisions.

The relative position and orientation of end-effectors of a general bimanual robot are defined as follows [22]:

$$\mathbf{p}_r = \bar{\mathbf{q}}_1 * (\mathbf{p}_2 - \mathbf{p}_1) \tag{3}$$

$$\mathbf{q}_r = \bar{\mathbf{q}}_1 * \mathbf{q}_2 \tag{4}$$

where $\mathbf{p}_1, \mathbf{p}_2 \in \mathbb{R}^3$, $\mathbf{q}_1, \mathbf{q}_2 \in \mathbf{S}^3$ are the position vectors and quaternions that, respectively, describe the position and orientation of the end-effectors of the two robot arms. Here, $\bar{\mathbf{q}}$ denotes the quaternion conjugate, and * denotes the quaternion product. In our setup, the bimanual system consists of the industrial robot arm and the virtual mechanism describing the tool.

To compute the desired relative Cartesian motion $\mathbf{p}_r, \mathbf{q}_r$, we need to know both the Cartesian space robot motion $\mathbf{p}_1, \mathbf{q}_1$ and the virtual mechanism configurations $\mathbf{p}_2, \mathbf{q}_2$. The Cartesian space robot motion, as defined in (1), is obtained by human demonstration. In Section IV, we explain how to compute the corresponding virtual mechanism configurations.

To control the robot in relative coordinates, we must be able to compute the relative Jacobian $\mathbf{J}_r = [\mathbf{J}_{r,p}^\top \mathbf{J}_{r,\omega}^\top]^\top \in \mathbb{R}^{6\times n_1+n_2}$, where n_1 and n_2 denote the number of degrees of freedom of the robot and the virtual mechanism, respectively, and $\mathbf{J}_{r,p}$ and $\mathbf{J}_{r,\omega}$ denote the position- and orientation-related parts of the relative Jacobian. The relative Jacobian maps the joint velocities $\dot{\boldsymbol{\theta}}$ of the bimanual setup onto the associated relative velocities in Cartesian space.

Let us denote the position and orientation parts of the robot's geometric Jacobian as $\mathbf{J}_1 = [\mathbf{J}_{1,p}^\top \mathbf{J}_{1,\omega}^\top]^\top \in \mathbb{R}^{6\times n_1}$ and the geometric Jacobian of the virtual mechanism as $\mathbf{J}_2 = [\mathbf{J}_{2,p}^\top \mathbf{J}_{2,\omega}^\top]^\top \in \mathbb{R}^{6\times n_2}$, where both kinematic chains are expressed in a common coordinate system. We selected the real robot base coordinate system as the common coordinate system. The relative Jacobian can then be derived from (3) and (4) as follows [22]:

$$\mathbf{J}_r = \begin{bmatrix} \mathbf{R}_1^\top(-\mathbf{J}_{1,p} + \mathbf{S}(\mathbf{p}_2 - \mathbf{p}_1)\mathbf{J}_{1,\omega}) & \mathbf{R}_1^\top \mathbf{J}_{2,p} \\ -\mathbf{R}_1^\top \mathbf{J}_{1,\omega} & \mathbf{R}_1^\top \mathbf{J}_{2,\omega} \end{bmatrix} \tag{5}$$

where $\mathbf{R}_1, \mathbf{R}_2 \in \mathbb{R}^{3\times 3}$ are the rotation matrices corresponding to quaternions $\mathbf{q}_1$ and $\mathbf{q}_2$, and $\mathbf{S}(\mathbf{p}_2 - \mathbf{p}_1)$ is the skew-symmetric matrix

$$\mathbf{S}\left(\begin{bmatrix} x \\ y \\ z \end{bmatrix}\right) = \begin{bmatrix} 0 & -z & y \\ z & 0 & -x \\ -y & x & 0 \end{bmatrix}. \tag{6}$$

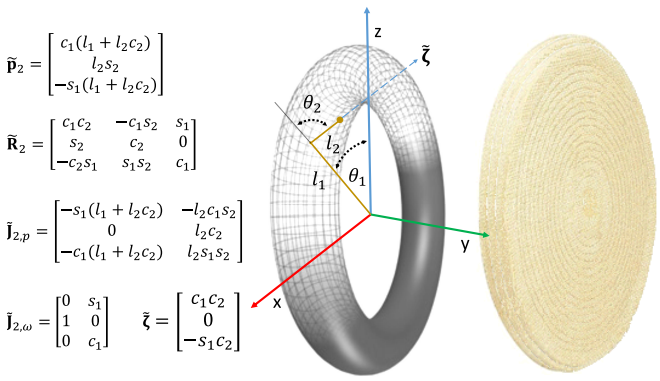


Fig. 3. Polishing disk is modeled as two rotational degrees of freedom mechanism, where l_1 and l_2 are link lengths, θ_1 and θ_2 are the joint coordinates, and c_* and s_* are the abbreviations for $\cos(\theta_*)$ and $\sin(\theta_*)$, respectively. $\boldsymbol{\zeta}$ is the directional vector of the last link.

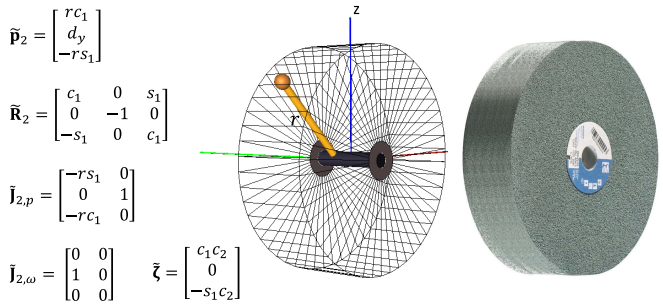


Fig. 4. Grinding disk is modeled as a two degrees of freedom mechanism, where r is the disk radius and θ and d_y are the joint coordinates, respectively.

In the above derivation, we focused on the case when a robot manipulates a workpiece. In this case, the robot represents one arm of the bimanual mechanism and the machine tool the second arm. Another possibility is that the machine tool is mounted on the robot, and the workpiece is positioned at a fixed location. In this case, we can augment the robot links with the virtual links of the machine tool, and the overall system can be treated as one robot arm. In this article, we concentrate on the first setup, which results in the above-described bimanual system.

A few examples of how to form the virtual mechanisms and the corresponding Jacobians are given in Section III-A.

A. Example Virtual Mechanisms

Some examples of the kinematic structure and the corresponding Jacobians for the most common machine tools are shown in Figs. 3–5. For the sake of simplicity and clarity, the tool kinematics is presented in the machine tool coordinate system. However, our approach requires that all entities are expressed in the robot base coordinate system. The mapping from the tool to the robot base coordinate system is given by the following formulas: $\mathbf{p}_2 = \mathbf{R}_t\tilde{\mathbf{p}}_2 + \mathbf{p}_t$, $\mathbf{R}_2 = \mathbf{R}_t\tilde{\mathbf{R}}_2$, $\mathbf{J}_{2,p} = \mathbf{R}_t\tilde{\mathbf{J}}_{2,p}$, $\mathbf{J}_{2,\omega} = \mathbf{R}_t\tilde{\mathbf{J}}_{2,\omega}$, and $\boldsymbol{\zeta} = \mathbf{R}_t\tilde{\boldsymbol{\zeta}}$, where tilde denotes the tool coordinates and $\mathbf{p}_t$ and $\mathbf{R}_t$ are the position and orientation of the machine tool base in the robot base coordinate system, respectively. The position $\mathbf{p}_t$ and orientation $\mathbf{R}_t$ must be carefully determined by a calibration procedure to ensure the operation of the complete system.

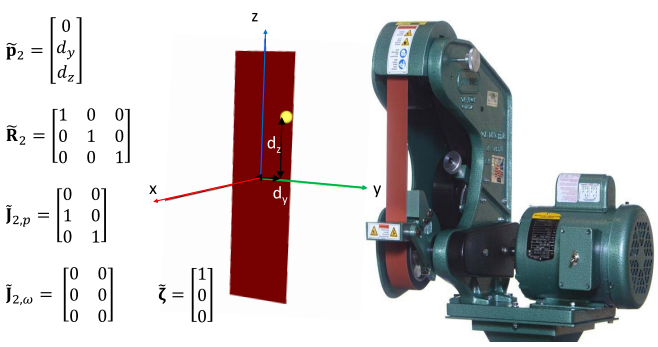


Fig. 5. Flat belt grinder is modeled as a two translational degrees of freedom mechanism with coordinates d_y and d_z.

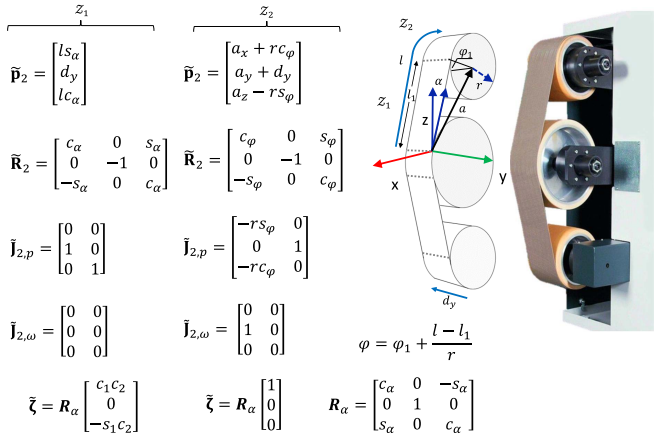


Fig. 6. Curved belt grinder is modeled as two degrees of freedom mechanism with coordinates d_y and l. Coordinate l denotes the distance on the belt from the origin. The kinematic description of a compound mechanism is divided into zones $\mathcal{Z}_i$. For sake of simplicity, only two zones are presented here.

More complex virtual mechanisms can be derived by combining these three basic shapes. One such case is shown in Fig. 6. Note that none of these virtual mechanisms has a kinematic singularity.

IV. Online Estimation of the Contact Point Using Force–Torque Measurements

The position of the tool center point of the virtual mechanism, which is needed to compute the relative coordinates of the bimanual system (3), is equal to the position of the contact between the treated workpiece and the machine tool. If precise mathematical models were available, the contact points could be determined using geometrical relations between the robot, workpiece, and machine tool [20]. While the robot's kinematic model and the model of the machine tool are usually available, this is often not the case for the workpiece models, especially in low batch size production. In such cases, the contact point cannot be calculated geometrically. Here, we propose an optimization-based approach to estimate the contact point from the measured forces and torques.

The idea to exploit force–torque measurements for the determination of contact points is not new. Bicchi *et al.* [24] investigated the contact point identification in the context of object recognition. Kitagaki *et al.* [25], [26] proposed to determine a pseudocontact point to detect contact state

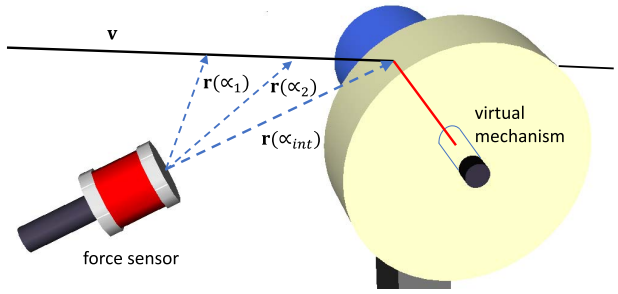


Fig. 7. Finding the contact point by computing the intersection/closest point of the line $\mathbf{r}(\alpha)$ with the machine tool (in this case grinding disk). All quantities are given in the robot base coordinate system.

transitions. The problem of online contact point estimation was also considered in [27] and [28], where an adaptive controller was proposed to solve the problem. Liu *et al.* [29] studied contact status perception between two objects involved in the simultaneous precision assembly of multiple objects. The evaluation of contact forces has also recently been used to identify successful snap-fit assemblies [30].

External forces and torques acting on the end-effector are related by the cross product with position vector $\mathbf{r}$

$$\mathbf{M} = \mathbf{r} \times \mathbf{F} = -\mathbf{S}(\mathbf{F})\mathbf{r} = -\begin{bmatrix} 0 & -F_z & F_y \\ F_z & 0 & -F_x \\ -F_y & F_x & 0 \end{bmatrix} \mathbf{r} \quad (7)$$

where $\mathbf{r}$ denotes the vector from the end-effector to the contact point. Keep in mind that all these quantities are given in the robot base coordinate system, matrix $\mathbf{S}(\mathbf{F}) \in \mathbb{R}^{3\times 3}$ is skew-symmetric; hence, its rank is equal to 2 $\forall \mathbf{S}(\mathbf{F}) \neq 0$. Consequently, there exist multiple solutions for $\mathbf{r}$ satisfying (7). The space of all possible solutions forms a line defined by the following equation (see Fig. 7):

$$\mathbf{r}(\alpha) = -\mathbf{S}(\mathbf{F})^{+}\mathbf{M} + \alpha\mathbf{v} \quad (8)$$

where $\mathbf{S}(\mathbf{F})^{+}$ denotes the pseudoinverse of $\mathbf{S}(\mathbf{F})$, $\alpha \in \mathbb{R}$ is an arbitrary scalar value, and $\mathbf{v} = \mathbf{F}/\|\mathbf{F}\| \in \mathbb{R}^3$ belongs to the null-space of $-\mathbf{S}(\mathbf{F})$ (since $-\mathbf{S}(\mathbf{F})\mathbf{v} = (\mathbf{F}/\|\mathbf{F}\|) \times \mathbf{F} = 0$).

To avoid the computation of the pseudoinverse of $\mathbf{S}(\mathbf{F})$, we can use an alternative formulation

$$\mathbf{r}(\alpha) = (\mathbf{F} \times \mathbf{M})/\|\mathbf{F}\|^2 + \alpha\mathbf{v}. \quad (9)$$

The proof of equivalence of both formulations is given in Appendix A.

Thus, the point of contact cannot be uniquely determined unless additional restrictions are imposed. Assuming that the geometry of the machine tool is known, e.g., disk, torus, or belt, we can restrict the solution to lie at the intersection of the line with the tool. The contact point can then be determined by solving the following optimization problem (see Fig. 7):

$$\arg\min_{\boldsymbol{\theta}_2,\alpha} \frac{1}{2}\|\mathbf{d}(\boldsymbol{\theta}_2, \alpha)\|^2 \quad (10)$$

where

$$\mathbf{d}(\boldsymbol{\theta}_2, \alpha) = \mathbf{p}_2(\boldsymbol{\theta}_2) - \mathbf{r}(\alpha) \quad (11)$$

is the difference function between an arbitrary point on the line $\mathbf{r}(\alpha)$ and an arbitrary point $\mathbf{p}_2(\boldsymbol{\theta}_2)$ on the surface of the machine tool, which is modeled as a virtual mechanism with joints $\boldsymbol{\theta}_2$.

To solve the optimization problem (10), we need to calculate the Jacobian of the difference function $\mathbf{d}(\boldsymbol{\theta}_2, \alpha)$

$$\mathbf{J}_d(\boldsymbol{\theta}_2, \alpha) = [\,\mathbf{J}_{2,p}(\boldsymbol{\theta}_2) \quad -\mathbf{v}\,] \quad (12)$$

where $\mathbf{J}_{2,p}$ is the positional part of the Jacobian of the virtual mechanism. We can then apply Gauss–Newton iteration to compute the optimal α^* and joints $\boldsymbol{\theta}_2^*$ of the virtual mechanism

$$\begin{bmatrix} \boldsymbol{\theta}_{2,j+1} \\ \alpha_{j+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\theta}_{2,j} \\ \alpha_j \end{bmatrix} - \mathbf{J}_d^{+}(\boldsymbol{\theta}_{2,j}, \alpha_j)\mathbf{d}(\boldsymbol{\theta}_{2,j}, \alpha_j). \quad (13)$$

We prove, in Appendix B, that the Gauss–Newton iteration is guaranteed to find the solution of optimization problem (10) provided that the procedure is given a good enough initial estimate. This is always the case because the contact point changes continuously and the solution at the previous time step is known.

To objectively evaluate the performance of the proposed method, ATI Delta universal force–torque sensor was mounted under the grinding machine, as shown in Fig. 8. The contact point was precisely measured with the mechanical digitizer MicroScribe G2LX6. We performed a few typical grinding movements and compared the contact points measured by Microscribe with the contact points estimated by the proposed algorithm. The experiment was carried out both with the grinding machine turned on and off. When the grinding machine was running, the measured forces were filtered with the zero-lag Rauch–Tung–Striebel filter. Fig. 8 shows that the estimated contact point is more accurate when the machine is switched off and when the error is in the range of a few millimeters. However, the contact point estimated with the grinding machine switched on is not far off. We conclude that it is better to demonstrate the task when the grinding machine is switched off, but the contact points estimated when the grinding machine is ON are also useable.

V. Tracking of the Desired Relative Motion

Using the approach described in Section IV, we can augment the demonstrated motion (1) with the corresponding virtual mechanism coordinates

$$\{\mathbf{p}_{2,k}, \mathbf{q}_{2,k}, t_k\}_{k=1}^{T}. \quad (14)$$

The demonstrated motion (1) can then be transformed into the relative motion of the demonstrator with respect to the machine tool using (3) and (4) and the recorded data (14)

$$\{\mathbf{p}_{r,k}, \mathbf{q}_{r,k}, \dot{\mathbf{p}}_{r,k}, \boldsymbol{\omega}_{r,k}, \ddot{\mathbf{p}}_{r,k}, \dot{\boldsymbol{\omega}}_{r,k}, t_k\}_{k=1}^{T}. \quad (15)$$

The relative position and orientation velocities $\dot{\mathbf{p}}_k$, $\boldsymbol{\omega}_k$ and relative accelerations $\ddot{\mathbf{p}}_k$, $\dot{\boldsymbol{\omega}}_k$ are computed from the relative positions $\mathbf{p}_r$ and orientations $\mathbf{q}_r$ using numerical differentiation. These data are needed to compute an effective task policy representation for robot control.

Thus, instead of directly following the human demonstrator motion (1), we propose to track the relative motion as specified by the data set (15). In the following, we first describe how

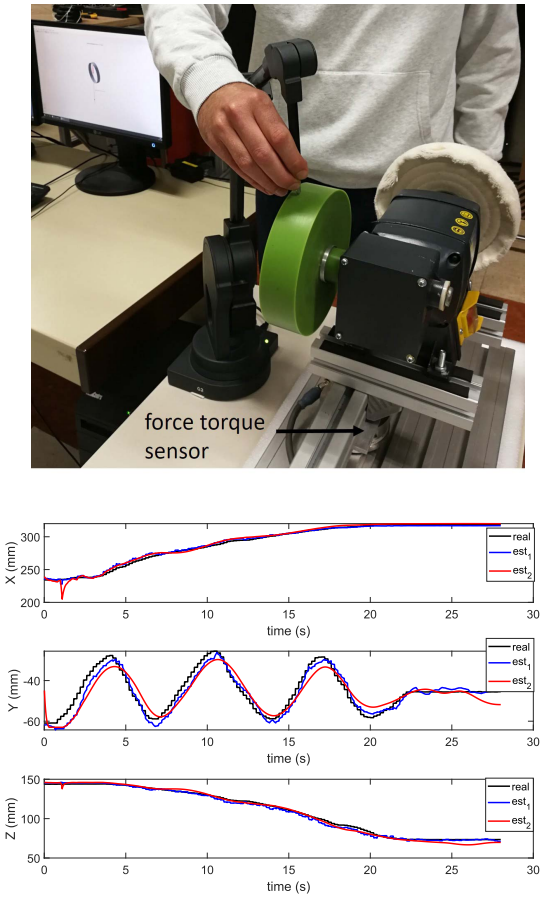


Fig. 8. Top: experimental setup for evaluation of the contact point estimation. The contact point is estimated: 1) by mechanical digitizer and 2) from the measured force–torque data. Force sensor was mounted under the base of the brush machine. Bottom: contact point calculated from force–torque data using the Gauss–Newton iteration (blue and red line) is close to the one measured by the digitizer (black line). The blue line refers to the online estimation with the grinding machine switched off and the red line to the filtered estimation with the grinding machine switched on.

to compute a suitable representation for the Cartesian space relative motion and then how the actual robot motion can be generated.

A. Policy Representation Using Dynamic Movement Primitives Framework

For efficient application and to enable online policy adaptation, it is important that trajectories are represented in a compact form. We have chosen dynamic movement primitives (DMPs) [31] for this purpose, modified to facilitate nonuniform speed scaling [32]. Furthermore, we make use of Cartesian space DMP representation proposed in [33]. Thus, the Cartesian space trajectory is encoded by the following system of nonlinear differential equations for positions $\mathbf{p}$ and orientations $\mathbf{q}$:

$$\upsilon(s)\tau\dot{\mathbf{z}} = \alpha_z(\beta_z(\mathbf{g}_p - \mathbf{p}) - \mathbf{z}) + \mathbf{f}_p(s) \tag{16}$$

$$\upsilon(s)\tau\dot{\mathbf{p}} = \mathbf{z} \tag{17}$$

$$\upsilon(s)\tau\dot{\boldsymbol{\eta}} = \alpha_z(\beta_z 2\log(\mathbf{g}_o * \overline{\mathbf{q}}) - \boldsymbol{\eta}) + \mathbf{f}_o(s) \tag{18}$$

$$\upsilon(s)\tau\dot{\mathbf{q}} = \frac{1}{2}\boldsymbol{\eta} * \mathbf{q} \tag{19}$$

$$\upsilon(s)\tau\dot{s} = -\alpha_s s. \tag{20}$$

In the above set of equations, s denotes the phase, τ is the duration of the policy, and $\mathbf{z}$ and $\boldsymbol{\eta}$ are auxiliary variables. With the proper selection of parameters, e.g., $\alpha_z = 4\beta_z > 0$ and $\alpha_s > 0$, the system (16)–(20) becomes critically damped and 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 $\bar{\mathbf{q}}$ the quaternion conjugation. The quaternion logarithm $\log : \mathbf{S} \mapsto \mathbb{R}^3$ is defined as

$$\log(\mathbf{q}) = \log(\upsilon, \mathbf{u}) = \begin{cases} \arccos(\upsilon)\dfrac{\mathbf{u}}{\|\mathbf{u}\|}, & \mathbf{u} \neq 0 \\ [0,0,0]^{\mathrm{T}}, & \text{otherwise.} \end{cases} \tag{21}$$

It maps the quaternion difference calculated as $\mathbf{q}_1 * \overline{\mathbf{q}}_2$ to the corresponding angular velocity needed to rotate quaternion $\mathbf{q}_2$ to quaternion $\mathbf{q}_1$ in unit time.

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 (16)–(20) 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)

$$\mathbf{f}_p(s) = \frac{\sum_{i=1}^{M} \mathbf{w}_{i,p}\Psi_i(s)}{\sum_{i=1}^{M}\Psi_i(s)}s \tag{22}$$

$$\mathbf{f}_o(s) = \frac{\sum_{i=1}^{M} \mathbf{w}_{i,o}\Psi_i(s)}{\sum_{i=1}^{M}\Psi_i(s)}s \tag{23}$$

$$\Psi_i(s) = \exp(-h_i(s - c_i)^2) \tag{24}$$

where free parameters $\mathbf{w}_{i,p}$, $\mathbf{w}_{i,o}$ determine the shape of position and orientation trajectories, M is the number of basis functions, and c_i are the centers of RBFs, evenly distributed along the trajectory, with h_i being their widths. In this formulation, we introduced the temporal scaling function $\upsilon(s)$, which is used to specify nonlinear variations from the demonstrated speed profile. It is defined as follows:

$$\upsilon(s) = 1 + \frac{\sum_{i=1}^{M_\upsilon} \upsilon_i\Psi_i(s)}{\sum_{i=1}^{M_\upsilon}\Psi_i(s)} \tag{25}$$

where υ_i are the free parameters (weights) and M_υ is the number of basis functions.

In order to parameterize the demonstrated control policy with a DMP, the weights $\mathbf{w}_{i,p}$ and $\mathbf{w}_{i,o}$ need to be calculated and also the parameters $\mathbf{g}_p$, $\mathbf{g}_o$, and τ need to be determined. The shape weights $\mathbf{w}_{i,p}$ and $\mathbf{w}_{i,o}$ are calculated by applying standard regression techniques [33], using the demonstrated relative trajectory (15) as the target for weight fitting. The other parameters are simply set as follows: $\mathbf{g}_p = \mathbf{p}_{r,T}$, $\mathbf{g}_o = \mathbf{q}_{r,T}$, and $\tau = t_T - t_1$. We initially set $\upsilon_i = 0$, i.e., $\upsilon = 1$, meaning that the demonstrated speed profile is left unchanged.

Unlike motion trajectories, the forces do not need to generalize the goal value. Therefore, we simply encode the recorded forces as a linear combination of radial basis functions

$$\mathbf{F}(s) = \frac{\sum_{i=1}^{M_F} \mathbf{w}_{i,F}\Psi_i(s)}{\sum_{i=1}^{M_F}\Psi_i(s)} \tag{26}$$

where Ψ_i are defined as in (24) and $\mathbf{w}_{i,F} \in \mathbb{R}^3$ are computed from data (2) using regression techniques.

B. Tracking of the Relative DMPs

Given the relative Cartesian motion of the robot and the virtual mechanism as specified by the DMP, the joint values $\boldsymbol{\theta}$ of both can be obtained using resolved rate motion control, which is a standard Jacobian-based control scheme for tracking the desired end-effector motion at the specified velocity without having to compute inverse kinematics. In this control scheme, the joint velocities of the bimanual system are calculated by the following formula:

$$\dot{\boldsymbol{\theta}} = \mathbf{J}_{W,r}^{+}\left(\begin{bmatrix}\dot{\mathbf{p}}_{r,\mathrm{DMP}}\\ \boldsymbol{\omega}_{r,\mathrm{DMP}}\end{bmatrix} + \mathbf{K}_k\mathbf{e}_r\right) + \left(\mathbf{I} - \mathbf{J}_r^{+}\mathbf{J}_r\right)\dot{\boldsymbol{\theta}}_0. \quad (27)$$

Here, $\mathbf{e}_r \in \mathbb{R}^6$ is the error between the desired relative coordinates and the actual relative coordinates

$$\mathbf{e}_r = \begin{bmatrix}\mathbf{p}_{r,\mathrm{DMP}} - \mathbf{p}_r\\ 2\log(\mathbf{q}_{r,\mathrm{DMP}} * \bar{\mathbf{q}}_r)\end{bmatrix} \quad (28)$$

and the subscripts $()_r$ and $()_{r,\mathrm{DMP}}$ denote the actual relative coordinates and the desired relative coordinates computed by the DMP, respectively. $\dot{\boldsymbol{\theta}}_0$ denotes the null space joint velocities that are used to optimize the self-motion of the bimanual mechanism. $\mathbf{K}_k \in \mathbb{R}^{(n_1+n_2)\times(n_1+n_2)}$ is the positive definite diagonal matrix with kinematic controller gains. $\mathbf{J}_{W,r}^{+}$ is the weighted pseudoinverse of the relative Jacobian $\mathbf{J}_r$ calculated as

$$\mathbf{J}_{W,r}^{+} = \mathbf{W}^{-1}\mathbf{J}_r^T\left(\mathbf{J}_r\mathbf{W}^{-1}\mathbf{J}_r^T\right)^{-1}. \quad (29)$$

The weighting matrix $\mathbf{W} \in \mathbb{R}^{(n_1+n_2)\times(n_1+n_2)}$ is a diagonal positive-definite matrix. Its function is to individually scale the robot and the virtual mechanism velocities. By setting the coefficients belonging to the virtual mechanism to higher values, one can diminish the velocity of changing the contact point between the machine tool and the treated object.

The algorithm returns both joint values of the real robot $\boldsymbol{\theta}_1$ and joint values of the virtual robot $\boldsymbol{\theta}_2$, $\boldsymbol{\theta} = [\boldsymbol{\theta}_1^\top, \boldsymbol{\theta}_2^\top]^\top$. Only the robot joints $\boldsymbol{\theta}_1$ are, of course, used to control the robot. Virtual joints $\boldsymbol{\theta}_2$ can be used to detect and limit excessive rotation of the virtual mechanism, which could sometimes cause collisions with the environment. Note that the weighted pseudoinverse (29) minimizes the joint velocities and, thus, inherently avoids singularities. If the system, nevertheless, approached a singular configuration, this could be resolved by optimizing the null-space velocities in (27).

VI. Adaptation Using Iterative Learning Control

In finishing tasks, forces exerted against the machine tool are the key process parameters [18] and should be precisely tracked in order to achieve the desired quality of the demonstrated finishing operation. Unlike in assembly operations, torques are usually not relevant in finishing operations. By tracking the demonstrated forces, we can also eliminate small calibration errors that accumulate during the transfer of the captured trajectories to the robot mechanism.

Tracking of the demonstrated forces is especially demanding with faster movements, which often arises during the finishing operations. Typically, a skilled operator moves at much higher speeds than what an average industrial robot is capable of tracking [20], [34]. To overcome this problem, Ng *et al.* [13], [20] proposed to scale the desired velocities and forces, as well as all other relevant process parameters, e.g., grinding belt speed or disk speed. Unfortunately, it is not straightforward to relate force and speed scaling.

Note that the mechanical digitizer used to capture expert demonstrations in our system already compels the operator to use slower movements during the task demonstration. The demonstrated velocities are, thus, generally below maximal robot velocities. Nevertheless, the problem of accurate force tracking remains. In the following, we propose to apply learning to address this issue. We ensure accurate tracking of the demonstrated forces by applying the ILC framework. In the past, ILC was, for example, successfully applied for edge chamfering operations [35].

A. Adaptation Framework

ILC is often used in robotics due to its simplicity, effectiveness, and robustness when dealing with repetitive tasks, such as finishing operations. It is particularly effective when the robot follows a similar trajectory repeatedly and updates the underlying parameters in order to perfect the skill. The general aim of ILC is to improve the behavior of the control system by learning the feedforward compensation signal [8] from control errors in previous execution cycles. In this work, the feedforward compensation signal was chosen as an offset to the demonstrated position trajectory. It is learned in such a way that it minimizes the errors between the demonstrated and the actually applied forces. To achieve this aim, ILC updates the desired relative robot positions as follows:

$$\mathbf{p}_{r,\mathrm{DMP},l}(t_k) = \mathbf{p}_{r,\mathrm{DMP}}(t_k) + \Delta\mathbf{p}_{k,l} + \mathbf{K}_p\boldsymbol{\zeta}_k\|\mathbf{e}_{k,l}\| \quad (30)$$

$$\Delta\mathbf{p}_{k,l} = \mathcal{Q}(\Delta\mathbf{p}_{k,l-1} + \mathbf{L}_{p,l}\boldsymbol{\zeta}_k\|\mathbf{e}_{k+1,l}\|) \quad (31)$$

where k is the sampling step on the trajectory, $k = 1, \ldots, T$, l is the iteration index of ILC, and $\mathbf{e}_{k,l} = \mathbf{F}_k - \mathbf{F}_{k,l}$ is the force tracking error, where $\mathbf{F}_k$ are the desired forces as recorded during user demonstration (2) and $\mathbf{F}_{k,l}$ are the actually applied forces at ILC iteration step l. $\mathbf{p}_{r,\mathrm{DMP}}(t_k)$ are the demonstrated relative positions encoded with a DMP. Orientations, however, remain unchanged, i.e., as demonstrated. In the above equations, $\mathbf{K}_p, \mathbf{L}_{p,l} \in \mathbb{R}^{3\times3}$ are positive definite diagonal matrices with control gains on the diagonal. $\mathcal{Q}$ denotes discrete-time low pass filter and provides the learning stability [36]. $\boldsymbol{\zeta}_k \in \mathbb{R}^3$ is the unity vector, which determines the direction of force adaptation. This vector coincides with the normal to the tool at the point of contact with the treated part [35], [37]. It is obvious that, in all our virtual mechanism formulations, this is the direction of the end link of the virtual mechanism. It can be easily calculated from the virtual joints $\boldsymbol{\theta}_2$, which are anyway computed in (27) at each time step (look for the examples of vector $\boldsymbol{\zeta}$ calculated for the most common grinding tools in Figs. 3–5). The updated term $\Delta\mathbf{p}_l$ provides the learned feedforward compensation signal.

The proposed update rule (30) is a variant of current-iteration force control [8]. Our approach differs from the more usual formulations, where forces are tracked along each direction in tool coordinates. Due to sensor noise

and high-force measurements arising from the contact of the workpiece with the tool, we obtain far better results when applying force adaptation only in a single direction, here described with vector $\boldsymbol{\zeta}$. Note that vector $\boldsymbol{\zeta}$ is not affected by the force sensor noise and vibrations, as it is calculated from the task policy.

Another difference is in the current-iteration force controller. In most of the admittance force controllers, force error is related to the velocity and not to the position, as shown in (30). The former formulation is more appropriate in schemes without learning. However, in ILC-based schemes, the position-based current-iteration force control (30) is more appropriate, as it has no stability issues and can learn feedforward compensation signals more effectively [38].

The learned compensation signal and the tracking error from the previous cycle are encoded as a linear combination of RBFs in exactly the same way as the demonstrated forces. This has several benefits: 1) more compact representation of trajectories reduces computer memory requirements; 2) we can apply speed scaling of trajectories and control signals provided by speed scaled DMPs framework; and 3) we can omit filtering, i.e., we can set $\mathcal{Q} = 1$, since the appropriate filtering is provided by encoding control signals with DMPs [39].

The proposed learning framework can account for calibration errors and generally improves the overall performance of the demonstrated policy. However, it cannot compensate for random errors resulting, for example, from variations in the geometry of the workpieces. These types of deviations can be compensated by the DMP phase-stopping technique, as proposed in [38]. When there is a large deviation between the measured and the demonstrated force, we slow down the execution of the task by means of nonuniform speed scaling factor $v(s)$ in (16)–(20). This way we provide enough time for the impedance force controller to adapt.

VII. Implementation and Experimental Evaluation

In this section, we outline the implementation of the proposed framework for finishing operations, evaluate its performance, and demonstrate the policy transfer between different combinations of robots and machine tools. The overall framework architecture is presented in Fig. 9. Within this architecture, the skills, robots, and machine tools descriptions are stored and can be reused to program other workcells. It can be implemented on any robot that allows the execution of free-form Cartesian space trajectories. The developed architecture has another benefit. Once the Cartesian space trajectory has been downloaded to the robot controller, the robot can be controlled in real time and adapt the learned skill using phase stopping and ILC.

Experimental evaluation was performed on the system shown in Fig. 10 consisting of the following components:

1) policy demonstration system described in Section II;
2) six degrees of freedom industrial robot MOTOMAN MH6 equipped with force sensor Dyn Pick WEF-6A100-30 and controlled by MOTOMAN DX100 controller;
3) seven degrees of freedom collaborative robot Franka Emika Panda;

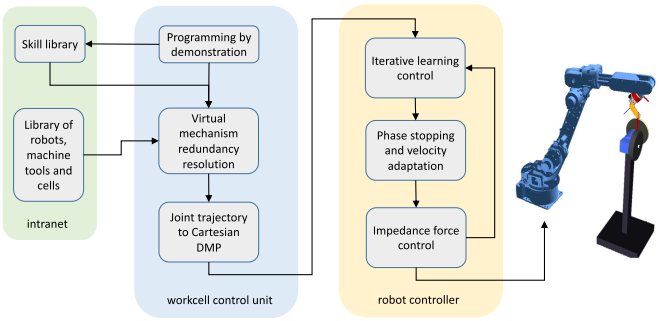
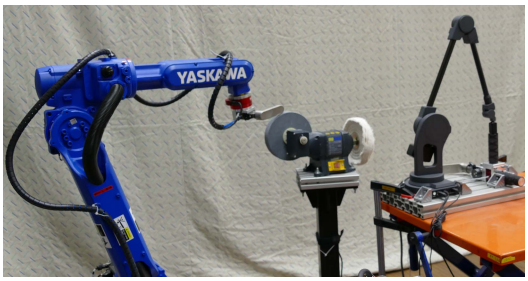
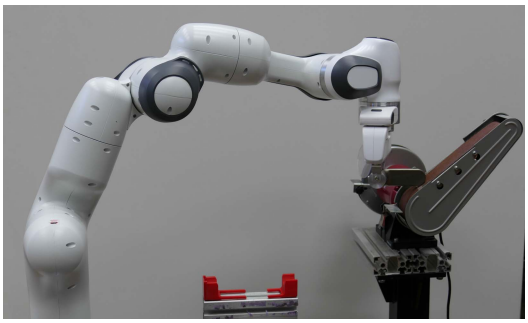


Fig. 9. Block scheme of the implemented framework.



(a)



(b)

Fig. 10. Two experimental workcells for polishing and grinding. (a) Experimental workcell with Yaskawa MOTOMAN MH6 robot, polishing/grinding machine, and policy demonstration system. (b) Experimental workcell with Franka-Emika robot and flat belt polishing machine.

4) workcell control unit consisting of an industrial PC;
5) polishing/grinding machine with a polishing and grinding disk;
6) curved belt grinder;
7) flat belt polishing machine.

The workcell control unit handles the programming by demonstration process and calculates contact points and relative coordinates from the demonstrated motion. Next, it encodes the relative motion as a DMP and the arising forces as a linear combination of RBFs. Redundancy resolution (27) is also implemented on the main control unit. It is performed in simulation and generates a suitable robot joint trajectory to perform the desired task. The robot joint trajectory is then mapped to the Cartesian space trajectory, encoded as a DMP, and downloaded to the robot controller. Transformation to the Cartesian space is necessary to enable online trajectory adaptation. The robot then executes the computed DMP policy, handles phase stopping, and provides for the force and

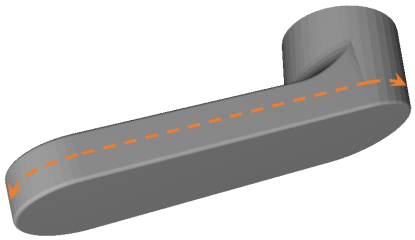


Fig. 11. Workpiece to be polished. The dotted line presents the polishing trajectory.

Fig. 12. Image sequence showing the task execution without processing the demonstrated motion with the proposed redundancy resolution scheme. The execution failed due to the violation of joint limits.

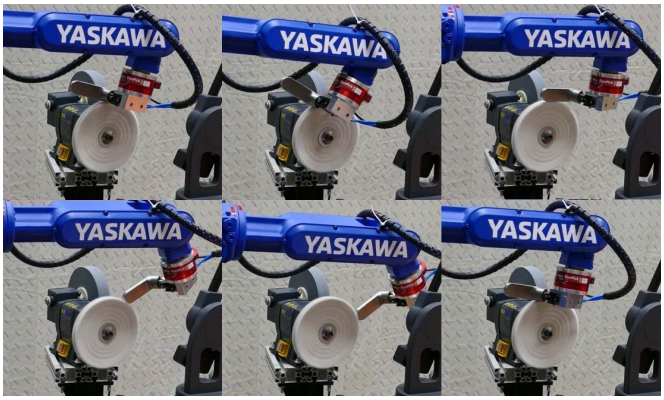


Fig. 13. Image sequence showing the task execution with the proposed redundancy resolution scheme. The robot executed the task successfully.

trajectory tracking using impedance control law. The controllers were implemented for the MOTOMAN MH6 robot (using Yaskawa MotoPlus C library) and Franka Emika Panda robot (using the FrankaLib C library in ROS environment).

To evaluate the effectiveness of the proposed framework, we demonstrated a faucet polishing trajectory and executed it with and without exploiting the functional redundancy of the task with the virtual mechanism. The task was to polish the edge of the faucet handle along the trajectory, as shown in Fig. 11. The initial task was demonstrated on the rotary polishing machine, as described in Section V-A.

We first performed the demonstrated task with the MH6 robot on the polishing machine with round polishing brushes. Without exploiting functional redundancy, the robot was not able to perform the demonstrated trajectory since the execution violated the joint limits, as shown in the image sequence presented in Fig. 12. On the other hand, when the demonstrated trajectory was processed by the proposed redundancy resolution approach, the robot avoided the joint limits and successfully executed the task, as shown in Fig. 13.

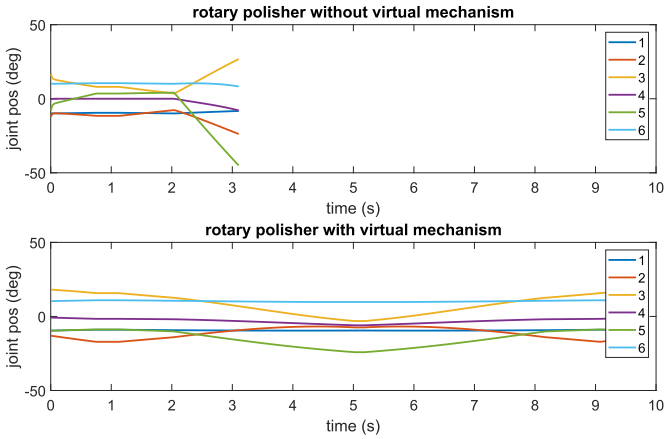


Fig. 14. Graphs of the executed robot joint trajectories. Top: executed trajectories generated without using the proposed redundancy resolution scheme. The robot stopped after 3.2 s as it violated the joint limits. Bottom: executed trajectories generated by the proposed redundancy resolution scheme.

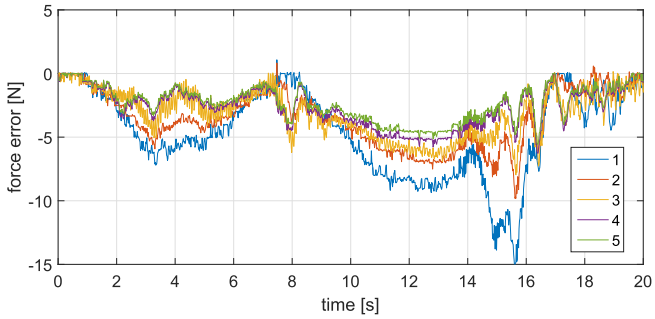


Fig. 15. Norm of the difference between the applied and the demonstrated forces in five adaptation cycles. Note that the error is significantly reduced after five iterations.

The resulting robot joint trajectories for both cases are shown in Fig. 14. Note that, without VM, the robot stopped after 3.2 s due to the violation of the joint limits.

Fig. 15 shows policy adaptation on the rotary polisher using the trajectory generated by the proposed redundancy resolution scheme. After five adaptation cycles, the robot substantially reduced the error between the demonstrated and the measured forces. Note that the implemented redundancy resolution approach minimizes the joint velocities, which additionally improves force tracking.

In the next experiment, we moved the grinding machine to several different locations within the robot's workspace and performed the same grinding task at each new location. We did not change the orientation of the grinding machine with respect to the robot, nor did we change the height of the base. The aim of the experiment was to evaluate how effective the virtual mechanism approach is at enlarging the area in the robot's workspace where the desired finishing operation can be successfully performed. Besides observing whether or not the robot could perform the given task at each location, we also measured the maximum joint speeds arising during the task execution. The results are shown in Fig. 16. They clearly show that the area in which the robot can successfully accomplish the task is much larger when the virtual mechanism approach is used. In addition, the maximum

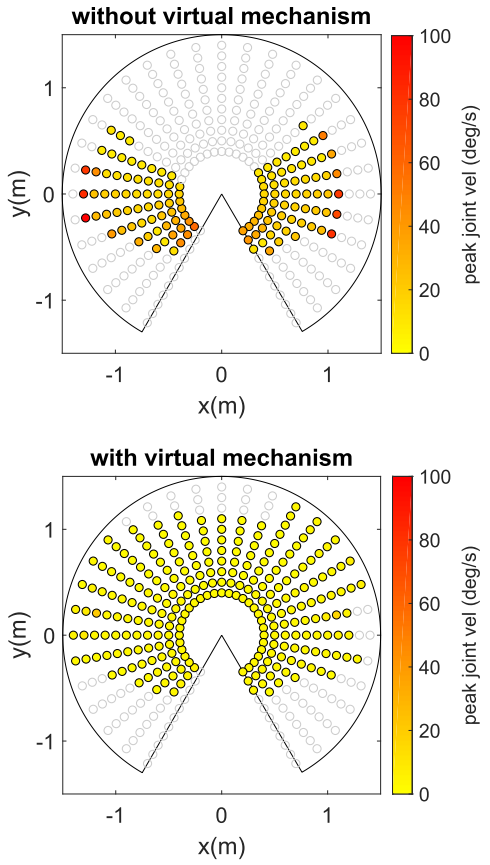


Fig. 16. Changing the location of the grinding machine with respect to the robot. The robot was placed at the center of the workcell. Empty circles denote the grinding machine locations where the robot failed to execute the demonstrated finishing operation. The color of the filled circles reflects the peak joint speed necessary to execute the task.

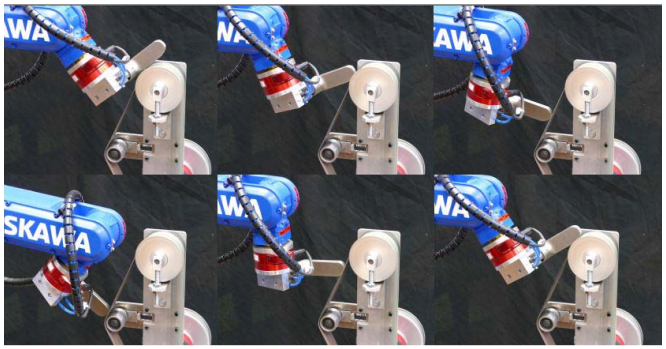


Fig. 17. Image sequence showing task transfer to a different tool. The robot executed the task successfully.

joint speeds were much lower in this case. This also means that with the virtual mechanism approach, we can perform more complex movements, treat more complex workpieces, and use machine tools of different geometrical forms.

The next experiment shows the task transfer, where we utilized the same robot, but a different tool. The virtual mechanism of the curved belt grinder has rather complex kinematics, as shown in Fig. 6. Fig. 17 depicts the image sequences showing the resulting robot motion. Note that the task was executed successfully without needing a new user demonstration. The corresponding joint trajectories are plotted

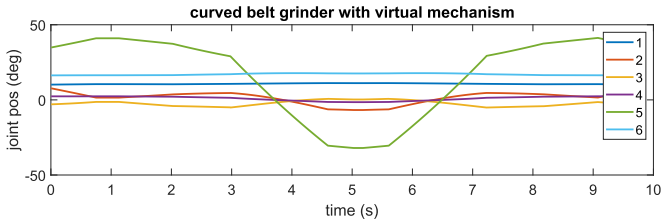


Fig. 18. Graphs of the executed robot joint trajectories. Compared with Fig. 14, the demonstrated task policy was the same, but the polishing tool was different.

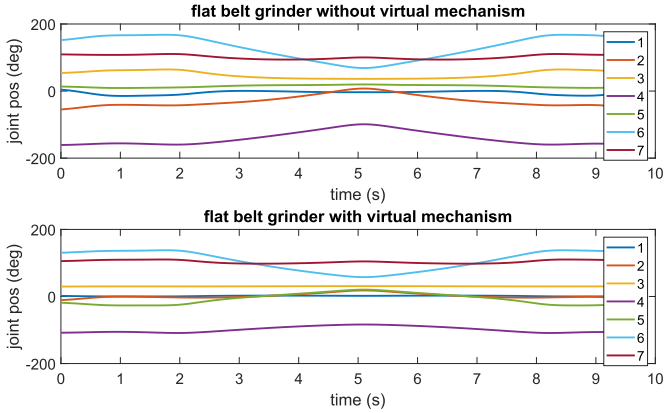


Fig. 19. Robot joint trajectories for the same task executed on the belt grinding machine with and without applying the virtual mechanism approach. The legends refer to the robot joints.

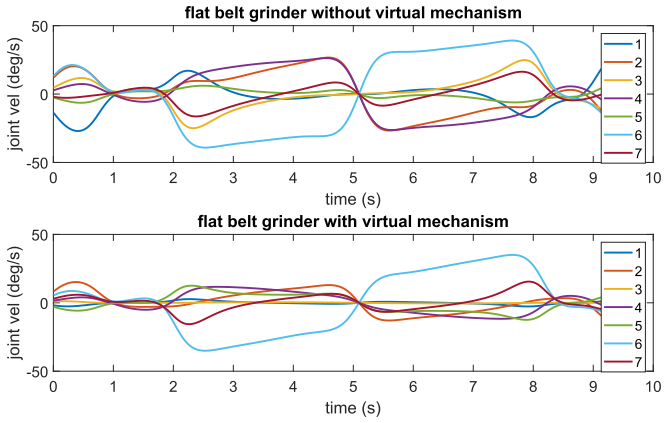


Fig. 20. Robot joint velocities for the same task executed on the belt grinding machine with and without applying the virtual mechanism approach. The legends refer to the robot joints.

in Fig. 18. They are different from the trajectories in Fig. 14 due to the differences in the geometry of both machine tools.

The final experiment dealt with policy transfer to another robot and machine tool. The machine tool, in this case, was the flat belt grinder. Its kinematics is shown in Fig. 5. Again, we utilized the same policy demonstration, but, this time, the task was performed by a seven DOF robot arm Franka Emika Panda. The redundancy resolution scheme generated the position at the top of the belt and moved the contact point along the belt during the policy execution. As the robot is intrinsically redundant, the benefit of augmentation with

the virtual mechanism is less evident in this case. Namely, also with a fixed point on the belt, the robot could successfully accomplish the task. However, with the proposed approach, the task was executed with smaller joint velocities. The resulting joint trajectories and joint velocities are shown in Figs. 19 and 20, respectively.

Videos of all experiments are available as a Supplementary Material to this article.

VIII. Conclusion

In this article, we proposed a new approach to learning finishing operations by demonstration. Several important novelties were introduced in the proposed framework.

1) Using the virtual mechanism concept, the functional redundancy of the robot can be exploited so that only the relevant part of the demonstrated motion is reproduced. The system becomes redundant even though the applied robot only has six degrees of freedom. With the proposed approach, the robot can reproduce the demonstrated motions that would be outside of the robot's workspace if the robot just played back the demonstrated trajectories.
2) We developed a new optimization approach to estimate the contact point and the virtual mechanism coordinates of the tool from the measured forces and torques. A geometrical model of the workpiece is not needed, which is an important advantage in many industrial applications.
3) The tracking accuracy of the force controller is enhanced by applying ILC, which is further improved by performing adaptation in the direction determined by the orientation of the virtual mechanism and by applying the phase stopping mechanism.
4) The presented methodology enables efficient policy transfer between different combinations of robots and machine tools without requiring additional user demonstrations. This is because: 1) the acquired task policy is represented as the Cartesian space DMP, which is independent of the robot; 2) the proposed augmentation of the robot with the virtual mechanism and the associated redundancy resolution scheme greatly improves the chance that the demonstrated task can be reproduced by the robot; and 3) the iterative learning control allows the robot to alter the motion and compensate for differences in the dynamic models of different robots and tools.

An immediate benefit of the proposed approach is that the actual finishing policy is optimized, taking into account the kinematic and dynamic capabilities of the robot rather than the human operator. Another advantage is that the robot can modify the trajectory online to compensate for the wear and tear of the tools. Furthermore, the application of the virtual mechanism approach enables the robot to perform more complex movements and reduces the possibility of encountering joint limits. Finally, the robot can perform different tasks at lower joint speeds, which improves the tracking accuracy both in terms of the trajectory and the applied forces and torques.

In our experiments, we captured the policy performed by a skilled operator using the mechanical digitizer, as explained in Section II. It is possible to demonstrate finishing operations also with other approaches, e.g., by kinesthetic guiding a collaborative robot operating in the zero gravity mode. However, most industrial robots still do not support kinesthetic guidance. The approach with mechanical digitizer enables learning of finishing operations by human demonstration also for standard industrial robots. Comparing the two approaches, the mechanical digitizer has lower inertia and, thus, enables a more natural demonstration of finishing operations. Kinesthetic guidance, on the other hand, has the advantage of directly gathering the joint space robot trajectories, thus avoiding the problems arising from joint limits and singularities when converting Cartesian space motion to the robot joint space motion.

We performed a few hundreds of different polishing and grinding experiments. The results of our experiments show clear advantages of the proposed redundancy resolution scheme based on the virtual mechanism approach. Without using the proposed approach, the robot was unsuccessful in performing about half of the demonstrated tasks, but they could all be successfully performed with the proposed approach. We did not encounter any example where the proposed approach would degrade the performance.

The main objective of our research was to enable effective imitation of finishing operations demonstrated by human experts. It is, however, clear that real finishing operations in industrial environments require the acquisition of several different finishing policies, which must be automatically combined and sequenced to treat different workpieces successfully. Another important aspect not considered in this work is the material removal control, which is tightly connected with the finishing technology. It is far from straightforward to control material removal during robotic finishing operations. These issues are left for future investigations.

Appendix A
Equivalence of Two Forms for the Estimation of the Contact Point

To prove the equivalence of formulations given by (8) and (9), it is necessary to show

$$-\mathbf{S}(\mathbf{F})^{+}\mathbf{M} = \frac{\mathbf{F}\times\mathbf{M}}{\|\mathbf{F}\|^2}. \tag{32}$$

We start with the cross product of forces $\mathbf{F}$ and torques $\mathbf{M}$, substitute $\mathbf{M}$ with $\mathbf{r}\times\mathbf{F}$, and apply the formula of the triple cross product

$$\begin{aligned}\mathbf{F}\times\mathbf{M} &= \mathbf{F}\times(\mathbf{r}\times\mathbf{F}) = (\mathbf{F}^{\top}\mathbf{F})\mathbf{r} - (\mathbf{F}^{\top}\mathbf{r})\mathbf{F}\\ &= \|\mathbf{F}\|^2\mathbf{r} - (\mathbf{F}^{\top}\mathbf{r})\mathbf{F}.\end{aligned} \tag{33}$$

Inserting $\mathbf{r}$ from (8) into (33) yields

$$\begin{aligned}\mathbf{F}\times\mathbf{M} = &-\|\mathbf{F}\|^2\mathbf{S}(\mathbf{F})^{+}\mathbf{M} + \|\mathbf{F}\|^2\alpha\mathbf{v}\\ &+ (\mathbf{F}^{\top}\mathbf{S}(\mathbf{F})^{+}\mathbf{M})\mathbf{F} - (\mathbf{F}^{\top}\alpha\mathbf{v})\mathbf{F}.\end{aligned} \tag{34}$$

The last term in the above equation can be rearranged as $(\mathbf{F}^{\top}\alpha\mathbf{v})\mathbf{F} = \alpha(\mathbf{F}^{\top}\mathbf{F}/\|\mathbf{F}\|)\mathbf{F} = \alpha\|\mathbf{F}\|^2\mathbf{v}$; thus, the second term cancels the fourth one.

Since $\mathbf{S}(\mathbf{F})\mathbf{F} = -\mathbf{F}\times\mathbf{F} = 0$, $\mathbf{F}$ belongs to the null space of $\mathbf{S}(\mathbf{F})$. Using SVD decomposition, it is possible to prove that, for any matrix, the null space of its pseudoinverse

is equal to the null space of its transpose. Since $\mathbf{S}(\mathbf{F})$ is skew symmetric, it holds $\mathbf{S}(\mathbf{F})^\top = -\mathbf{S}(\mathbf{F})$; thus, the null space of the pseudoinverse of a skew symmetric matrix is equal to the null space of the original matrix. It is, therefore, true that $\mathbf{S}(\mathbf{F})^+\mathbf{F} = 0$, and consequently, $(\mathbf{F}^\top\mathbf{S}(\mathbf{F})^+\mathbf{M})\mathbf{F} = (-(\mathbf{S}(\mathbf{F})^+\mathbf{F})^\top\mathbf{M})\mathbf{F} = 0$, which proves the relation (32).

APPENDIX B
STABILITY AND CONVERGENCE OF GAUSS–NEWTON ITERATION FOR CONTACT ESTIMATION

The Gauss–Newton iteration is in general not guaranteed to converge. In this section, we prove its convergence for contact point estimation using the Lyapunov stability theory. The proof is based on the observation that solving optimization problem (10) by the Gauss–Newton iteration (13) is equivalent to solving the following differential equation system:

$$\begin{bmatrix} \dot{\boldsymbol{\theta}}_2 \\ \dot{\alpha} \end{bmatrix} = -\mathbf{J}_d^+(\boldsymbol{\theta}_2, \alpha)\mathbf{d}(\boldsymbol{\theta}_2, \alpha) \tag{35}$$

where the difference function $\mathbf{d}(\boldsymbol{\theta}_2, \alpha)$ and its Jacobian $\mathbf{J}_d(\boldsymbol{\theta}_2, \alpha) \in \mathbb{R}^{3\times(n_2+1)}$ are defined as in (11) and (12), respectively.

Recall that, according to the Lyapunov stability theorem, a dynamical system of the form (35) is locally asymptotically stable at the point $(\boldsymbol{\theta}_2^*, \alpha^*)$ if there exists a continuous and continuously differentiable Lyapunov function $V(\boldsymbol{\theta}_2, \alpha) : \mathcal{B} \subset \mathbb{R}^{n_2+1} \mapsto \mathbb{R}$ such that

$$\begin{aligned}
&\text{a) } V(\boldsymbol{\theta}_2, \alpha) > 0 \quad \forall(\boldsymbol{\theta}_2, \alpha) \in \mathcal{B} \subset \mathbb{R}^{n_2+1},\ (\boldsymbol{\theta}_2, \alpha) \neq (\boldsymbol{\theta}_2^*, \alpha^*) \\
&\text{b) } \dot{V}(\boldsymbol{\theta}_2, \alpha) < 0 \quad \forall(\boldsymbol{\theta}_2, \alpha) \in \mathcal{B} \subset \mathbb{R}^{n_2+1},\ (\boldsymbol{\theta}_2, \alpha) \neq (\boldsymbol{\theta}_2^*, \alpha^*) \\
&\text{c) } V(\boldsymbol{\theta}_2^*, \alpha^*) = 0, \quad \dot{V}(\boldsymbol{\theta}_2^*, \alpha^*) = 0.
\end{aligned} \tag{36}$$

Note that, in all our examples, the number of degrees of freedom of the virtual mechanism $n_2 = 2$ and the virtual mechanism has no kinematic singularities.

Let us define the following Lyapunov function candidate:

$$V = \frac{1}{2}\mathbf{d}(\boldsymbol{\theta}_2, \alpha)^\top\mathbf{d}(\boldsymbol{\theta}_2, \alpha). \tag{37}$$

Note that condition *a)* is fulfilled by the definition of the candidate function (37), provided $\mathcal{B}$ does not contain any other solution besides $(\boldsymbol{\theta}_2^*, \alpha^*)$. Since we assume that the workpiece is in contact with the tool, there always exist parameters $(\boldsymbol{\theta}_2^*, \alpha^*)$ so that the difference vector $\mathbf{d}(\boldsymbol{\theta}_2^*, \alpha^*) = 0$; thus, $V(\boldsymbol{\theta}_2^*, \alpha^*) = 0$.

To prove conditions *b)* and *c)*, we next compute the derivative $\dot{V}$

$$\begin{aligned}
\dot{V} &= \mathbf{d}(\boldsymbol{\theta}_2, \alpha)^\top\dot{\mathbf{d}}(\boldsymbol{\theta}_2, \alpha) \\
&= \mathbf{d}(\boldsymbol{\theta}_2, \alpha)^\top(\mathbf{J}_{2,p}(\boldsymbol{\theta}_2)\dot{\boldsymbol{\theta}}_2 - \dot{\alpha}\mathbf{v}) \qquad (38) \\
&= \mathbf{d}(\boldsymbol{\theta}_2, \alpha)^\top\left(\mathbf{J}_d(\boldsymbol{\theta}_2, \alpha)\begin{bmatrix} \dot{\boldsymbol{\theta}}_2 \\ \dot{\alpha} \end{bmatrix}\right) \\
&= -\mathbf{d}(\boldsymbol{\theta}_2, \alpha)^\top\mathbf{J}_d(\boldsymbol{\theta}_2, \alpha)\mathbf{J}_d^+(\boldsymbol{\theta}_2, \alpha)\mathbf{d}(\boldsymbol{\theta}_2, \alpha) \qquad (39)
\end{aligned}$$

where (38) is obtained by computing the derivative of difference vector (11), while (39) follows if we replace $[\dot{\boldsymbol{\theta}}_2^\top, \dot{\alpha}]^\top$ with the expression given in (35). The matrix $\mathbf{J}_d\mathbf{J}_d^+$ is positive definite provided that $\mathbf{J}_d(\boldsymbol{\theta}_2, \alpha)$ has full rank. In this case, the derivative of the Lyapunov function V is negative, i.e., $\dot{V} < 0$. Thus, condition *b)* also holds since there is no singularity in the virtual mechanism in the parameters subset $\mathcal{B}$, in particular at solution $(\boldsymbol{\theta}_2^*, \alpha^*)$. It follows from (39) that $\dot{V}(\boldsymbol{\theta}_2, \alpha^*) = 0$ if $\mathbf{d}(\boldsymbol{\theta}_2^*, \alpha^*) = 0$. Thus, condition *c)* is also fulfilled.

The above derivation proves that the differential equation (35) is locally asymptotically stable. This means that if we start the iteration close enough to the solution, the parameters obtained by the Gauss–Newton iteration are guaranteed to converge to the solution $(\boldsymbol{\theta}_2^*, \alpha^*)$.

REFERENCES

[1] D. Antonelli, S. Astanin, and G. Bruno, "Applicability of human-robot collaboration to small batch production," in *Collaboration in a Hyperconnected World* (IFIP Advances in Information and Communication Technology), H. Afsarmanesh, L. M. Camarinha-Matos, and A. L. Soares, Eds. Springer, 2016, pp. 24–32.

[2] T. Gašpar *et al.*, "Smart hardware integration with advanced robot programming technologies for efficient reconfiguration of robot workcells," *Robot. Comput.-Integr. Manuf.*, vol. 66, Dec. 2020, Art. no. 101979.

[3] P. Kormushev, S. Calinon, and D. G. Caldwell, "Imitation learning of positional and force skills demonstrated via kinesthetic teaching and haptic input," *Adv. Robot.*, vol. 25, no. 5, pp. 581–603, Jan. 2011.

[4] J. Roseli, L. Basañez, and I. Díaz, "Graphical task-level robot programming for polishing and grinding," *IFAC Proc. Volumes*, vol. 35, no. 1, pp. 235–240, 2002.

[5] L. Sciavicco and B. Siciliano, *Modeling and Control of Robot Manipulators*. London, U.K.: Springer, 2008.

[6] L. Huo and L. Baron, "The joint-limits and singularity avoidance in robotic welding," *Ind. Robot, Int. J.*, vol. 35, no. 5, pp. 456–464, Aug. 2008.

[7] C. L. Nehaniv and K. Dautenhahn, "The correspondence problem," in *Imitation in Animals and Artifacts*, K. Dautenhahn and C. L. Nehaniv, Eds. Cambridge, MA, USA: MIT Press, 2002.

[8] D. A. Bristow, M. Tharayil, and A. G. Alleyne, "A survey of iterative learning control," *IEEE Control Syst. Mag.*, vol. 26, no. 3, pp. 96–114, Jun. 2006.

[9] L. Basanez and J. Rosell, "Robotic polishing systems," *IEEE Robot. Autom. Mag.*, vol. 12, no. 3, pp. 35–43, Sep. 2005.

[10] A. D. Wilbert, B. Behrens, C. Zymla, O. Dambon, and F. Klocke, "Robotic finishing process: An extrusion die case study," *CIRP J. Manuf. Sci. Technol.*, vol. 11, pp. 45–52, Nov. 2015.

[11] L. Zhou and H. Huang, "An automated robotic system for jet engine overhaul. System design and development for honeycomb repair," *Int. J. Adv. Manuf. Technol.*, vol. 19, no. 5, pp. 370–376, Mar. 2002.

[12] H. Huang, L. Zhou, X. Q. Chen, and Z. M. Gong, "SMART robotic system for 3D profile turbine vane airfoil repair," *Int. J. Adv. Manuf. Technol.*, vol. 21, no. 4, pp. 275–283, Feb. 2003.

[13] C. W. X. Ng, K. H. K. Chan, W. K. Teo, and I.-M. Chen, "A method for capturing the tacit knowledge in the surface finishing skill by demonstration for programming a robot," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Hong Kong, May 2014, pp. 1374–1379.

[14] J. Zhang, Y. Wang, and R. Xiong, "Industrial robot programming by demonstration," in *Proc. Int. Conf. Adv. Robot. Mechatronics (ICARM)*, Macau, China, Aug. 2016, pp. 300–305.

[15] E. Kalt, R. Monfared, and M. Jackson, "Development of an intelligent automated polishing system," in *Proc. 16th Int. Conf. Eur. Soc. Precis. Eng. Nanotechnol. (EUSPEN)*, 2016, pp. 1–2.

[16] E. Kalt, R. Monfared, and M. Jackson, "Towards an automated polishing system: Capturing manual polishing operations," *Int. J. Res. Eng. Technol.*, vol. 7, no. 5, pp. 182–192, 2016.

[17] A. Balijepalli and T. Kesavadas, "A haptic based virtual grinding tool," in *Proc. 11th Symp. Haptic Interfaces Virtual Environ. Teleoperator Syst. (HAPTICS)*, Mar. 2003, pp. 390–396.

[18] W. X. Ng, H. K. Chan, W. K. Teo, and I. M. Chen, "Programming robotic tool-path and tool-orientations for conformance grinding based on human demonstration," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Deajeon, South Korea, Oct. 2016, pp. 1246–1253.

[19] A. Kharidege, D. T. Ting, and Z. Yajun, "A practical approach for automated polishing system of free-form surface path generation based on industrial arm robot," *Int. J. Adv. Manuf. Technol.*, vol. 93, nos. 9–12, pp. 3921–3934, Dec. 2017.

[20] W. X. Ng, H. K. Chan, W. K. Teo, and I.-M. Chen, "Programming a robot for conformance grinding of complex shapes by capturing the tacit knowledge of a skilled operator," *IEEE Trans. Autom. Sci. Eng.*, vol. 14, no. 2, pp. 1020–1030, Apr. 2017.
[21] B. Nemec, K. Yasuda, N. Mullennix, N. Likar, and A. Ude, "Learning by demonstration and adaptation of finishing operations using virtual mechanism approach," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Brisbane, QLD, Australia, May 2018, pp. 7219–7225.
[22] N. Likar, B. Nemec, L. Zlajpah, S. Ando, and A. Ude, "Adaptation of bimanual assembly tasks using iterative learning framework," in *Proc. IEEE-RAS 15th Int. Conf. Humanoid Robots (Humanoids)*, Seoul, South Korea, Nov. 2015, pp. 771–776.
[23] S. Chiaverini, G. Oriolo, and I. D. Walker, "Kinematically redundant manipulators," in *Springer Handbook of Robotics*, B. Siciliano and O. Khatib, Eds. Berlin, Germany: Springer 2008, pp. 245–268.
[24] A. Bicchi, J. K. Salisbury, and D. L. Brock, "Contact sensing from force measurements," *Int. J. Robot. Res.*, vol. 12, no. 3, pp. 249–262, Jun. 1993.
[25] K. Kitagaki, T. Ogasawara, and T. Suehiro, "Contact state detection by force sensing for assembly tasks," in *Proc. IEEE Int. Conf. MFI. Multisensor Fusion Integr. Intell. Syst.*, Las Vegas, NV, USA, Oct. 1994, pp. 366–370.
[26] K. Kitagaki, M. Fujiwara, T. Suehiro, and T. Ogasawara, "Pseudo contact point monitoring for contact state estimation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. Hum. Environ. Friendly Robots High Intell. Emotional Quotients*, Kyongju, South Korea, Oct. 1999, pp. 832–837.
[27] Y. Karayiannidis, C. Smith, F. E. Vina, and D. Kragic, "Online contact point estimation for uncalibrated tool use," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Hong Kong, May 2014, pp. 2488–2494.
[28] F. Viña, C. Smith, D. Kragic, and Y. Karayiannidis, "Adaptive contact point estimation for autonomous tool manipulation," in *Proc. IEEE Int. Conf. Robot Autom. (ICRA), Auton. Grasping Manipulation Workshop*, Hong Kong, May 2014, pp. 1–2.
[29] S. Liu, Y.-F. Li, and D. Xing, "Sensing and control for simultaneous precision peg-in-hole assembly of multiple objects," *IEEE Trans. Autom. Sci. Eng.*, vol. 17, no. 1, pp. 310–324, Jan. 2020.
[30] S. Doltsinis, M. Krestenitis, and Z. Doulgeri, "A machine learning framework for real-time identification of successful snap-fit assemblies," *IEEE Trans. Autom. Sci. Eng.*, vol. 17, no. 1, pp. 513–523, Jan. 2020.
[31] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal, "Dynamical movement primitives: Learning attractor models for motor behaviors," *Neural Comput.*, vol. 25, no. 2, pp. 328–373, Feb. 2013.
[32] R. Vuga, B. Nemec, and A. Ude, "Speed adaptation for self-improvement of skills learned from user demonstrations," *Robotica*, vol. 34, no. 12, pp. 2806–2822, Dec. 2016.
[33] A. Ude, B. Nemec, T. Petric, and J. Morimoto, "Orientation in Cartesian space dynamic movement primitives," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Hong Kong, May 2014, pp. 2997–3004.
[34] E. A. Erlbacher, "Force control basics," *Ind. Robot, Int. J.*, vol. 27, no. 1, pp. 20–29, Feb. 2000.
[35] M. Sonehara, K. Hayashi, K. Nishijima, and H. Murakami, "Development of automation technology for precision finishing works employing a robot arm," *IHI Eng. Rev.*, vol. 45, no. 2, pp. 16–22, 2013.
[36] M. Norrlof and S. Gunnarsson, "Experimental comparison of some classical iterative learning control algorithms," *IEEE Trans. Robot. Autom.*, vol. 18, no. 4, pp. 636–641, Aug. 2002.
[37] F. Tian, Z. Li, C. Lv, and G. Liu, "Polishing pressure investigations of robot automatic polishing on curved surfaces," *Int. J. Adv. Manuf. Technol.*, vol. 87, nos. 1–4, pp. 639–646, Oct. 2016.
[38] F. J. Abu-Dakka, B. Nemec, J. A. Jørgensen, T. R. Savarimuthu, N. Krüger, and A. Ude, "Adaptation of manipulation skills in physical contact with the environment to reference force profiles," *Auto. Robots*, vol. 39, no. 2, pp. 199–217, Aug. 2015.
[39] B. Nemec, T. Petric, and A. Ude, "Force adaptation with recursive regression iterative learning controller," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Hamburg, Germany, Sep. 2015, pp. 2835–2841.

Bojan Nemec (Member, IEEE) received the diploma degree in electrical engineering and the M.Sc. and Ph.D. degrees in robotics from the University of Ljubljana, Ljubljana, Slovenia, in 1979, 1982, and 1988, respectively.

He is currently the Head of the Humanoid and Cognitive Robotics Laboratory and Research Councilor, Department of Automatics, Biocybernetics, and Robotics, Jožef Stefan Institute, Ljubljana. His research interests include robot control, robot learning, service robotics, and sports biomechanics.

Kenichi Yasuda joined the Yaskawa Electric Corporation, Kitakyushu, Japan, in 1990, where he has been engaged in research on control of industrial robots and service robots. These include redundant control, master-slave control, force-feedback control, and motion planning, and more. He is currently the Chief Manager of the Technology and Planning Department, Corporate Technology Division.

Aleš Ude (Member, IEEE) received the diploma degree in applied mathematics from the University of Ljubljana, Ljubljana, Slovenia, and the Dr.-Eng. degree from the University of Karlsruhe, Karlsruhe, Germany, in 1990 and 1995, respectively.

He is currently the Head of the Department of Automatics, Biocybernetics, and Robotics, Jožef Stefan Institute, Ljubljana. He is also associated with the ATR Computational Neuroscience Laboratory, Kyoto, Japan. His research interests include robot learning, imitation learning, reconfigurable robotic systems, and humanoid robotics.