Learning by Demonstration and Adaptation of Finishing Operations using Virtual Mechanism Approach

Bojan Nemec[1], Kenichi Yasuda[2],Nathanael Mullennix[2], Nejc Likar[1] and Aleš Ude[1]

Abstract— **In this paper we propose a new approach for efficient programming of grinding and polishing operation. In the proposed system, the initial policy is performed by a skilled operator and recorded with a passive digitizer. The demonstrated policy comprises both position and force data. The optimal robot execution of the task is provided by applying a virtual mechanism approach, which models the polishing/grinding tool as a serial kinematic chain. By joining the robot and the virtual mechanism in an augmented system, additional degrees of freedom are obtained and redundancy resolution can be applied to optimize the demonstrated motion. Another benefit of the proposed approach is that the same policy can be transferred to different combination of robots and grinding/polishing tools without any modification of the captured motion. The proposed approach requires known contact point between the treated object and the polishing/grinding tool. We propose a novel approach for accurate estimation of this point using data obtained from the force-torque sensor. Finally, the demonstrated path is refined to compensate for inaccurate calibration and different dynamics of a robot and the human demonstrator using iterative learning controller. The proposed method was verified in a real industrial environment.**

I. Introduction

While the robots are widely applied in large and medium-sized batch production, deployment of robots in small enterprises is still lagging behind [1]. One of the major reasons for this is excessive programming effort that is often required for programming new robot tasks. Learning from demonstration (LfD) is a promising technology aiming to increase the efficiency and ease of preparation of new robot tasks [2]. A typical example of processes that are considered hard to automate are finishing tasks such as polishing and grinding. Typically, they are performed by a skilled operator which aims to modify the treated object's surface in multiple passes of complex trajectories.

Since manual polishing and grinding requires a lot of skill and is also labor intensive and a potential health hazard, a lot of effort was made to efficiently automate such tasks using industrial robots [3], [4]. Early attempts involve simple point-to point programming and optimization using trial and error approach [5], [6]. Beside being time consuming, this approach lacks the possibility to transfer the learned policies to different robots and tools. More efficient programming relies on transferring the policy from a skilled operator, which involves both kinematic motion and the applied forces and torques. Various motion capturing system can be used to acquire such ploicies, such as 3D optical trackers [7], [8], [9], kinesthetic guiding [2] and special sensors which capture motion of the human arm [10]. Yet another LfD approach is based on CAD models (which might be built on-line with laser scanners) and the learning is performed in virtual environment using haptic devices [11], [12]. Forces and torques are captured using universal force/torque sensors, which can be incorporated into the polishing/grinding machine [7] or in special senzorized tools [13]. Finishing processes are generally very complex and require precise tuning of many process parameters in order to achieve the desired final quality. More advanced approaches are 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 [7], [12], [14], rather than transferring the operator's motion patterns directly.

One of the major problem with LfD for polishing and grinding are different kinematic and dynamic capabilities of human demonstrator and industrial robot. Therefore, direct copying of a captured motion to the industrial robot is in most cases not successful. Another unsolved problem remains how to use previously demonstrated skill applied to the different polishing and grinding machines and different robots. Our approach aims to solve the above mentioned problems by introducing a virtual mechanism paradigm, which models the polishing/grinding tool as an active robot mechanism. This results in additional degrees of freedom (D.O.F) of the augumented robot and virtual mechanism, which enables to apply on-line redundancy resolution algorithms to optimize the motion of the expanded mechanism. In order to eliminate the possible calibration error and to account for the different dynamics of a robot and the human demonstrator, we apply Iterative Learning Controller (ILC)

The rest of this paper is organized as follows: In Section II we describe the developed motion and force capture system. We also briefly introduce the speed-scaled Cartesian dynamic movement primitives, which is the underlying representation for the demonstrated motion. In Section III we introduce a virtual mechanism (VM) approach in the context of a bi-manual robot. Contact point estimation between the treated object and the grinding/polishing tool is presented in Section IV. Section V deals with iterative learning controller frame-

[1]Humanoid and Cognitive Robotics Lab, Department of Automatics, Biocybernetics and Robotics, Jožef Stean Institute, Ljubljana, Slovenia, bojan.nemec@ijs.si, nejc.likar@ijs.si, ales.ude@ijs.si [2]Robotics Technology Group, Yaskawa Electric Corporation, Kitakyushu, Japan, (e-mail: kenichi.yasuda@yaskawa.co.jp, nathanael.mullennix@yaskawa.co.jp)

work for refining polishing and grinding movements. The experimental verification of the overall scheme is described in Section VI. We summarize our approach and propose further research in Section VII.

II. Learning by demonstration using Dynamic Motion Primitives (DMP) Framework

We start by describing our learning from demonstration (LfD) setup and procedures necessary to capture the polishing/grinding motions and forces as demonstrated by a skilled operator. For this purpose we use a passive 6 axis mechanical digitizer (MicroScribe 6LX). The end link of the digitizer is attached to the specially designed handle, which incorporates universal force-torque sensor (DynaPick WEF-6H200). The handle allows the user to quickly change the faucet parts, which are the subject of polishing and grinding operation. The whole setup is shown in Fig. 1. During the demonstration, we acquire movement policy in Cartesian coordinates, calculated from the digitizer joint angles, as a set of points,

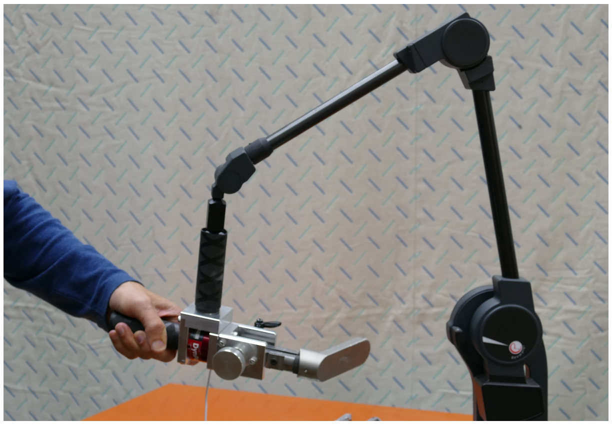

Fig. 1. Learning by demonstration setup for polishing/grinding.

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

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

Next, we parameterize this demonstrated policy with a nonlinear dynamical system that enables the encoding of general trajectories [15], [16], [17]. The trajectory is encoded by the following system of nonlinear differential equations for positions $\mathbf{p}$ and orientations $\mathbf{q}$,

$$\begin{aligned}
\nu(s)\tau\dot{\mathbf{z}} &= \alpha_z(\beta_z(\mathbf{g}_p - \mathbf{p}) - \mathbf{z}) + \mathbf{f}_p(s), && (2)\\
\nu(s)\tau\dot{\mathbf{p}} &= \mathbf{z}, && (3)\\
\nu(s)\tau\dot{\boldsymbol{\eta}} &= \alpha_z\left(\beta_z 2\log\left(\mathbf{g}_o * \overline{\mathbf{q}}\right) - \boldsymbol{\eta}\right) + \mathbf{f}_o(s), && (4)\\
\nu(s)\tau\dot{\mathbf{q}} &= \frac{1}{2}\boldsymbol{\eta} * \mathbf{q}, && (5)\\
\nu(s)\tau\dot{s} &= -\alpha_s s. && (6)
\end{aligned}$$

In the above set of equations s denotes the phase, $\tau = t_T$ is the duration of the policy and $\mathbf{z}$ and $\boldsymbol{\eta}$ are auxiliary variables. With the proper selection of parameters $\alpha_z = 4\beta_z > 0$ and $\alpha_s > 0$, the system (2) – (6) 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 $\bar{\mathbf{q}}$ quaternion conjugation. The quaternion logarithm $\log : \mathbf{S} \mapsto \mathbb{R}^3$ is defined as as

$$\log(\mathbf{q}) = \log(v, \mathbf{u}) = \begin{cases} \arccos(v)\dfrac{\mathbf{u}}{\|\mathbf{u}\|}, & \mathbf{u} \neq 0 \\ [0,0,0]^{\mathrm{T}}, & \text{otherwise} \end{cases} . \tag{7}$$

The nonlinear forcing terms $\mathbf{f}_p(s)$ and $\mathbf{f}_o(s)$ are formed in such a way that the response of the second-order differential equation system (2) – (6) can approximate any smooth point-to-point trajectory from the initial position $\boldsymbol{p}_0$ and orientation $\boldsymbol{q}_0$ to the final position $\mathbf{g}_p$ and orientation $\mathbf{g}_o$. The nonlinear forcing terms are defined as linear combinations of M radial basis functions (RBFs)

$$\begin{aligned}
\mathbf{f}_p(s) &= \frac{\sum_{i=1}^{M}\mathbf{w}_{i,p}\Psi_i(s)}{\sum_{i=1}^{M}\Psi_i(s)}s, && (8)\\
\mathbf{f}_o(s) &= \frac{\sum_{i=1}^{M}\mathbf{w}_{i,o}\Psi_i(s)}{\sum_{i=1}^{M}\Psi_i(s)}s, && (9)\\
\Psi_i(s) &= \exp\left(-h_i\left(s - c_i\right)^2\right), && (10)
\end{aligned}$$

where free parameters $\mathbf{w}_{i,p}$, $\mathbf{w}_{i,o}$ determine the shape of position and orientation trajectories and c_i are the centers of RBFs, evenly distributed along the trajectory, with h_i their widths. In this formulation we introduced the temporal scaling function $\nu(s)$ which is used to specify variations from the demonstrated speed profile. This allows to continuously and non-uniformly scale the speed of the executed motions, which is an important property for adaptation of the polishing/grinding policy.

In order to parameterize the demonstrated control policy with a DMP, the weights $\mathbf{w}_{i,p}$, $\mathbf{w}_{i,o}$ need to be calculated. The shape weights $\mathbf{w}_{i,p}$ and $\mathbf{w}_{i,o}$ are calculated by applying standard regression techniques [17] and using the demonstrated trajectory (1) as the target for weight fitting.

Similarly to the forcing terms (8) and (9), $\nu(s)$ is encoded as a linear combination of M_v RBFs

$$\nu(s) = 1 + \frac{\sum_{j=1}^{M_v} v_j\Psi_j(s)}{\sum_{j=1}^{M_v}\Psi_j(s)}, \tag{11}$$

where v_j are the corresponding free parameters (weights). For ν we initially set $v_j = 0$, i. e. $\nu = 1$, meaning that the demonstrated speed profile is left unchanged. v_j are assigned a different value only through the change of the execution speed. Besides movements (1) we also capture forces along the demonstrated trajectories

$$\mathcal{F} = \{\mathbf{F}_k, t_k\}_{k=1}^{\mathrm{T}}, \tag{12}$$

where $\mathbf{F}_k \in \mathbb{R}^3$ are the measured forces at times t_k. We encode these forces as a linear combination of radial basis functions

$$\mathbf{F}(s)=\frac{\sum_{j=1}^{M_F}\mathbf{v}_j\Psi_j(s)}{\sum_{j=1}^{M_F}\Psi_j(s)}, \tag{13}$$

where Ψ_j are defined as in (10) and $\mathbf{v}_j \in \mathbb{R}^3$ are determined using regression techniques.

III. Virtual Mechanism Framework as Bi-manual Robot Mechanism

Redundancy resolution schemes rely on a non-square Jacobian, which maps the joint velocities to the task space, in most cases given in Cartesian coordinates. Obviously, the kinematic redundancy appears if we have more degrees of freedom than needed to accomplish the task. In some cases the redundancy is not so obvious, as it arises from the shape of the tool. The task can be kinematically redundant even in cases when we have a 6 degrees of freedom robots and all 6 Cartesian coordinates are needed to describe the movements. Typical examples of such tasks are polishing, brushing and grinding. In such tasks it is not important where exactly the contact between the brush machine and the object appears; it only matters that the objects touches the brush machine with the desired orientation.

A framework to cope with this kind of redundancy is a virtual mechanism approach, where a polishing/grinding machine is modeled as a serial kinematic chain, called virtual mechanism (VM). Additional degrees of freedom arising from these virtual degrees of freedom can be exploited to optimize the robot motion. The virtual degrees of freedom allow the robot to change the contact point between the object and the polishing/grinding tool in such a way that it optimizes the desired criterion, which could be the minimization of joint velocities, joint limit avoidance, singularity avoidance, or collision avoidance. In this paper, we describe the additional degrees of freedom by exploiting a framework for describing bi-manual robots [18]. Within this framework, the polishing/grinding machine is modeled as one robot arm and only relative coordinates of the bi-manual setup are needed to accomplish the given task. Fig. 2 shows an example, where a robot and the polishing machine are modeled as a bi-manual system.

The relative position vector and quaternion of a bi-manual setup are defined as

$$\mathbf{p}_r = \bar{\mathbf{q}}_1 * (\mathbf{p}_2 - \mathbf{p}_1) \tag{14}$$

$$\mathbf{q}_r = \bar{\mathbf{q}}_1 * \mathbf{q}_2, \tag{15}$$

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 respectively describing the position and orientation of the robot's end-effector and the virtual linkage. $\bar{q}$ denotes the quaternion conjugate. Let's denote the position and orientation part of the geometric Jacobian of the robot as $\mathbf{J}_1 \in \mathbb{R}^{6\times n_1} = \begin{bmatrix}\mathbf{J}_{1,p}\\ \mathbf{J}_{1,\omega}\end{bmatrix}$, and the geometric Jacobian of the polishing/griding machine as $\mathbf{J}_2 \in \mathbb{R}^{6\times n_2} = \begin{bmatrix}\mathbf{J}_{2,p}\\ \mathbf{J}_{2,\omega}\end{bmatrix}$, both expressed in a common (usually robot base) coordinate system, where n_1 and n_2 is the number of degrees of freedom of the robot and virtual mechanism, respectively. A few examples of how to form the virtual mechanism and the corresponding Jacobian are given at the end of this section. The relative Jacobian relates joint velocities $\dot{\boldsymbol{\theta}}$ of the bi-manual setup and the resulting relative velocities in Cartesian coordinates. It can be derived from Eq. (14) and (15) in the form

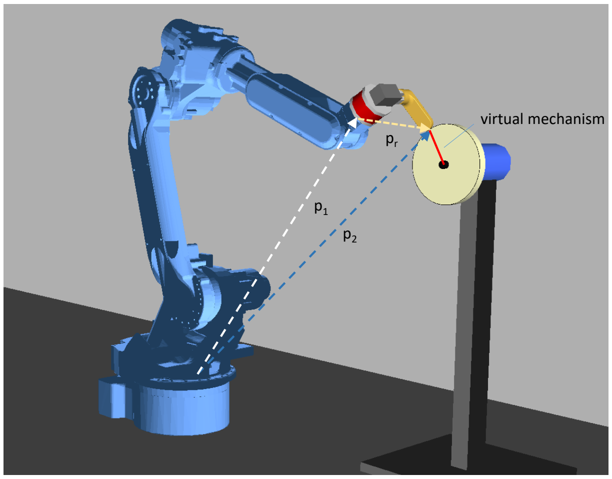

Fig. 2. Polishing machine and the robot are modeled as a bi-manual system. The common coordinate frame is placed at the robot's base. $\mathbf{p}_1$ and $\mathbf{p}_2$ are the vectors describing the robot's position and the position of the virtual mechanism, respectively. $\mathbf{p}_r$ denotes relative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

$$\mathbf{J}_r = \begin{bmatrix} \mathbf{R}_1{}^{\mathrm{T}}(-\mathbf{J}_{1,p}+\mathbf{S}(\mathbf{p}_2-\mathbf{p}_1)\mathbf{J}_{1,\omega}) & \mathbf{R}_1{}^{T}\mathbf{J}_{2,p} \\ -\mathbf{R}_1{}^{\mathrm{T}}\mathbf{J}_{1,\omega} & \mathbf{R}_1{}^{\mathrm{T}}\mathbf{J}_{2,\omega}, \end{bmatrix} \tag{16}$$

where $\mathbf{R}_1, \mathbf{R}_2 \in \mathbb{R}^{3\times 3}$ are the rotation matrices corresponding to quaternions $\mathbf{q}_1$, q_2 and $\mathbf{S}(\mathbf{p}_2 - \mathbf{p}_1)$ is the skew-symmetric matrix

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

Given the relative Cartesian motion of the robot and the virtual mechanism, the joint values $\boldsymbol{\theta}$ of both are obtained by integrating the relative joint velocities calculated by the following formula

$$\dot{\boldsymbol{\theta}} = \mathbf{J}_r^+\left(\begin{bmatrix}\dot{\mathbf{p}}_{r,d}\\ \boldsymbol{\omega}_{r,d}\end{bmatrix} + \mathbf{K}_k\mathbf{e}_r\right) + (\mathbf{I}-\mathbf{J}_r^+\mathbf{J}_r)\dot{\boldsymbol{\theta}}_0, \tag{17}$$

where $\mathbf{e}_r \in \mathbb{R}^6$ is the error between the desired relative coordinates (obtained with DMP integration) and actual relative coordinates,

$$\mathbf{e}_r = \begin{bmatrix}\mathbf{p}_{r,d} - \mathbf{p}_r\\ 2\log(\mathbf{q}_{r,d} * \bar{\mathbf{q}}_r)\end{bmatrix}, \tag{18}$$

where subscript $()_r$ and $()_d$ denotes the relative and the desired coordinates, respectively. $\dot{\boldsymbol{\theta}}_0$ are null space joint velocities, used to optimize the self-motion of the composed bi-manual mechanism. $\mathbf{K}_k \in \mathbb{R}^{(n_1+n_2)\times(n_1+n_2)}$ is a positive definite diagonal matrix with kinematic controller gains.

The algorithm returns both joint values of the real robot $\boldsymbol{\theta}_1$ and joint values of the virtual robot $\boldsymbol{\theta}_2$, $\boldsymbol{\theta} = \begin{bmatrix} \boldsymbol{\theta}_1 \\ \boldsymbol{\theta}_2 \end{bmatrix}$. Only joints $\boldsymbol{\theta}_1$ corresponding to the real robot 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 (VM), which might results e.g. in colliding with the environment.

In our setup, the desired relative coordinates $\mathbf{p}_{r,d}$, $\mathbf{q}_{r,d}$ are encoded with DMPs using equations (2) – (5). They are calculated from the demonstrated coordinates and from the estimated VM coordinates using (14). A procedure how to estimate the VM coordinates during the demonstration is proposed in the next section.

The measured relative coordinates $\mathbf{p}_r$ and $\mathbf{q}_r$ are calculated from the current actual configuration of the real robot $(\mathbf{p}_1, \mathbf{q}_1)$ and the virtual robot $(\mathbf{p}_2, \mathbf{q}_2)$ (solving direct kinematic for both robots with $\boldsymbol{\theta}$, obtained by integration of (17)).

Some examples of the kinematics structure and the corresponding Jacobian $\mathbf{J}_2$ of the polishing/brushing machine for three most common cases are given in Figs. 3, 4 and 5. More complex VM can be derived by combining these three basic shapes.

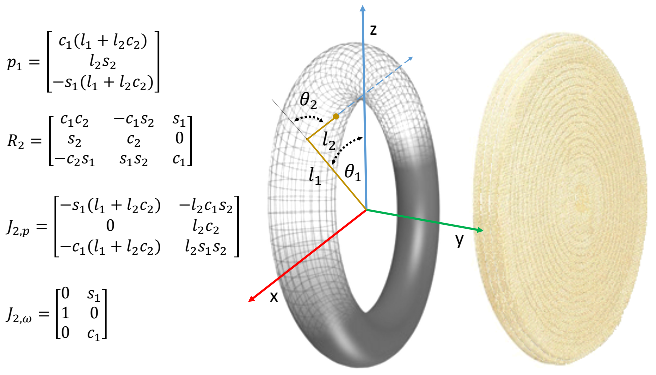
$p_1 = \begin{bmatrix} c_1(l_1 + l_2c_2) \\ l_2s_2 \\ -s_1(l_1 + l_2c_2) \end{bmatrix}$
$R_2 = \begin{bmatrix} c_1c_2 & -c_1s_2 & s_1 \\ s_2 & c_2 & 0 \\ -c_2s_1 & s_1s_2 & c_1 \end{bmatrix}$
$J_{2,p} = \begin{bmatrix} -s_1(l_1 + l_2c_2) & -l_2c_1s_2 \\ 0 & l_2c_2 \\ -c_1(l_1 + l_2c_2) & l_2s_1s_2 \end{bmatrix}$
$J_{2,\omega} = \begin{bmatrix} 0 & s_1 \\ 1 & 0 \\ 0 & c_1 \end{bmatrix}$
z
θ_2
l_2
l_1
θ_1
y
x

Fig. 3. Polishing disc is modeled as 2 rotational degrees of freedom mechanism, where l_1 and l_2 are link lengths, θ_1 and θ_2 are joint coordinates and c_n and s_n are the abbreviations for $\cos(\theta_n)$ and $\sin(\theta_n)$, respectively.

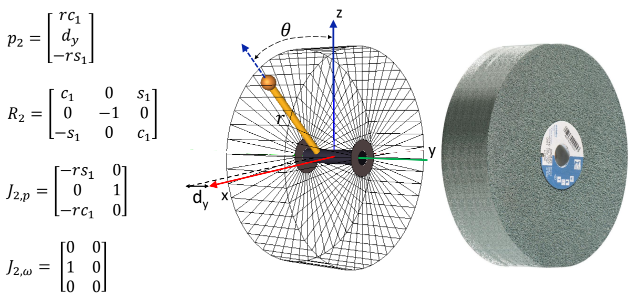
$p_2 = \begin{bmatrix} rc_1 \\ d_y \\ -rs_1 \end{bmatrix}$
$R_2 = \begin{bmatrix} c_1 & 0 & s_1 \\ 0 & -1 & 0 \\ -s_1 & 0 & c_1 \end{bmatrix}$
$J_{2,p} = \begin{bmatrix} -rs_1 & 0 \\ 0 & 1 \\ -rc_1 & 0 \end{bmatrix}$
$J_{2,\omega} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \end{bmatrix}$
θ
z
r
y
d_y
x

Fig. 4. Grinding disc is modeled as 2 degrees of freedom mechanism, where r is the disc radius and θ and $\mathbf{d}_y$ are joint coordinates, respectively.

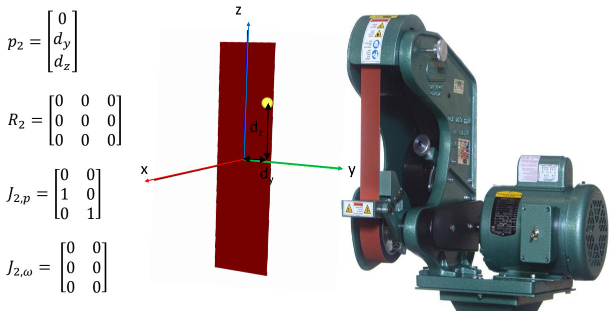
$p_2 = \begin{bmatrix} 0 \\ d_y \\ d_z \end{bmatrix}$
$R_2 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$
$J_{2,p} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$
$J_{2,\omega} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$
z
d_z
x
d_y
y

Fig. 5. Belt grinder is modeled as 2 translational degrees of freedom mechanism with coordinates d_y and d_z.

IV. Estimation of the Contact Point using Measured Forces and Torques

In order to determine the relative coordinates during LfD, it is necessary to determine the contact point between the polishing/grinding tool and the treated object. While the robot kinematic models and the model of polishing/grinding tool are often available, in industry this is often not the case for the treated part. In such cases this point cannot be calculated geometrically. Here we therefore propose to estimate the contact point from measured forces and torques.

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

$$\mathbf{M} = \mathbf{r} \times \mathbf{F} = -\mathbf{S}(\mathbf{F})\mathbf{r} = -\begin{bmatrix} 0 & -F_z & F_y \\ F_z & 0 & -F_x \\ -F_y & F_x & 0 \end{bmatrix} \mathbf{r}. \tag{19}$$

Keep in mind that all these quantities are given in the tool coordinate system, which is assumed to be aligned with the coordinate frame of the force-torque sensor. Eq. (19) can be used to determine the contact point [19], [20]. Matrix $\mathbf{S}(\mathbf{F}) \in \mathbb{R}^{3\times 3}$ is skew-symmetric, hence its rank is equal to 2, $\forall\ \mathbf{S}(\mathbf{F}) \neq 0$. It follows that there exist multiple solutions for $\mathbf{r}$ satisfying Eq. (19) for the measured external forces $\mathbf{F}$ and torques $\mathbf{M}$ (see Fig. 6). The space of all possible solutions of (19) is linear and forms a line defined by

$$\mathbf{r}(\alpha) = -\mathbf{S}(\mathbf{F})^{+}\mathbf{M} + \alpha\mathbf{v}, \tag{20}$$

where $\alpha \in \mathbb{R}$ is an arbitrary scalar value and $\mathbf{v} = \mathbf{F}/\|\mathbf{F}\| \in \mathbb{R}^3$. This is because $-\mathbf{S}(\mathbf{F})\mathbf{v} = (\mathbf{F}/\|\mathbf{F}\|) \times \mathbf{F} = 0$.

Among all possible solutions we choose the one which intersects the tool (disc, torus or belt) of a polishing/grinding machine (or is closest to the tool in case the solution line $\mathbf{r}(\alpha)$ does not intersect the tool due to the sensor noise). We denote the point at which the line intersects or is closest to the tool by $\mathbf{r}(\alpha_{int})$ (see Fig. 6). To determine the current configuration of the virtual mechanism $\boldsymbol{\theta}_2$ and the optimal α, which determines the intersection point, we solve the following optimization problem

$$\underset{\boldsymbol{\theta}_2,\alpha}{\operatorname{argmin}} \frac{1}{2}\|\mathbf{d}(\boldsymbol{\theta}_2,\alpha)\|^2 = \underset{\boldsymbol{\theta}_2,\alpha}{\operatorname{argmin}} \frac{1}{2}\|\mathbf{p}_2(\boldsymbol{\theta}_2) + \mathbf{S}(\mathbf{F})^{+}\mathbf{M} - \alpha\mathbf{v}\|^2, \tag{21}$$

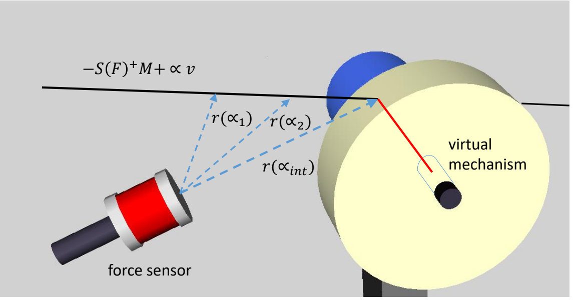

Fig. 6. Finding the contact point by computing the intersection/closest point of the line $\mathbf{r}(\alpha)$ with the grinding disc.

where vector $\mathbf{d}(\boldsymbol{\theta}_2, \alpha)$ is the distance between the line $\mathbf{r}(\alpha)$ and the polishing/grinding tool modeled as virtual mechanism with joints $\boldsymbol{\theta}_2$. The Jacobian $\mathbf{J}_d(\boldsymbol{\theta}_2, \alpha) \in \mathbb{R}^{3\times(n_2+1)}$ of vector function $\mathbf{d}(\boldsymbol{\theta}_2, \alpha)$ is given by

$$\mathbf{J}_d(\boldsymbol{\theta}_2, \alpha) = \left[\begin{array}{cc} \mathbf{J}_{2,p} & -\mathbf{v} \end{array}\right] \quad (22)$$

Note that in our experiments $n_2 \leq 2$ and the above optimization problem can be solved using any standard optimization method. We applied Gauss-Newton method. The optimal estimate from the previous time step on the demonstrated trajectory is used to initialize the estimation at the next step.

Experimental verification proved, that despite sensor noise, the contact point estimation remains within few millimeters of the optimal contact point as estimated by using geometrical models, providing proper force sensor calibration and gravity compensation of the sensor load.

V. Adaptation using Iterative Learning Controller

For finishing operations, forces exerted against the polishing/grinding tool are the key process parameter [12] and should be precisely tracked in order to achieve the desired polishing/grinding quality. Unlike in assembly operations, torques are usually not relevant for finishing operations. Tracking the demonstrated force also eliminates small calibration errors that accumulate during the transfer of the captured trajectories to the robot mechanism.

In has been reported in previous works that it is difficult to accurately track forces at high velocities [21], [14], which often arise during the human demonstration. To overcome this problem, it has been proposed to scale the desired velocities and forces [14], [7]. Unfortunately, it is not straightforward to relate force and speed scaling. In this work, we propose another solution to this problem, which relies on learning. We provide for accurate tracking of the demonstrated force profile by applying iterative learning control (ILC) framework.

ILC is often used in robotics due to its simplicity, effectiveness and robustness when dealing with repetitive tasks, such as finishing operations. The general aim of ILC is to improve the behavior of the control system by learning a feedforward compensation signal [22] 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. ILC updates the desired relative robot positions as follows

$$\begin{aligned} \mathbf{p}_{d,l}(k) &= \mathbf{p}_{DMP}(k) + \Delta\mathbf{p}_l(k) + \mathbf{K}_p\zeta\|\mathbf{e}_l(k)\|, & (23)\\ \Delta\mathbf{p}_l(k) &= \mathcal{Q}(\Delta\mathbf{p}_{l-1}(k) + \mathbf{L}_{p,l}\zeta\|\mathbf{e}_l(k+1)\|), & (24) \end{aligned}$$

where k is the sampling step on the trajectory, l is the iteration index of ILC, and $\mathbf{e}_l(k) = \mathbf{F}_d(k) - \mathbf{F}_l(k)$ is the force tracking error with $\mathbf{F}_d(k)$ and $\mathbf{F}_l(k)$ being the desired and actual forces at iteration step l, respectively. $\mathbf{p}_{DMP}(k)$ are the demonstrated relative positions which are encoded with DMPs. Orientations, however, remain unchanged, i.e. as demonstrated. In the above equations, $\mathbf{K}_p$, and $\mathbf{L}_{p,l}$ are $\mathbb{R}^{3\times 3}$ positive definite diagonal matrices with control gains. $\mathcal{Q}$ denotes a discrete-time low pass filter and provides the learning stability [23]. $\zeta \in \mathbb{R}^3$ is a unity vector, which determines the direction of force adaptation. This vector coincides with the normal to the polishing/grinding tool at the point of the contact with the treated part [24], [25]. Obviously, in the formulation of the virtual mechanism, this is the orientation of the end link of the virtual mechanism. It can be easily calculated from the virtual joints $\boldsymbol{\theta}_2$, which are anyway computed at each cycle time (17). The update term $\Delta\mathbf{p}_l$ provides the learned feedforward compensation signal.

The proposed update rule (23) is a variant of current-iteration force control [22]. Our approach differs from more usual formulations, where forces are tracked along each direction in tool coordinates. Due to sensor noise we obtain far better results by force adaptation in the direction of vector ζ, which is obtained from the current orientation of the virtual mechanism and is not subject to the force sensor noise and vibrations.

Another difference is in the current-iteration force controller. In most of admittance force controllers the force error is related to the velocity and not to the position as in our case. The former formulation is more appropriate in schemes without ILC. In ILC-based schemes, however, the position related current-iteration force control (23) is more appropriate, as it has no stability issues and learns better feed-forward compensation signals [26].

The learned compensation signal and the tracking error from the previous cycle are encoded as linear combinations of radial basis functions in exactly the same way as the demonstrated forces. This has several benefits: a) more compact representation of trajectories reduces computer memory requirements, b) we can apply velocity scaling of trajectories and control signals provided by speed scaled dynamic movement primitives framework, c) we can omit filtering, i.e. we can set $\mathcal{Q} = 1$, since the appropriate filtering is provided by encoding control signals with DMPs [27].

VI. Implementation and Experimental Evaluation

In this section we outline the implementation of the proposed framework for finishing operations. The developed system (see Fig. 7) consists of

- learning from demonstration unit, which was already described in Section II,
- 6 degrees of freedom industrial robot Motoman MH-6,
- control unit, consisting of Motoman DX100 controller and industrial PC,
- polishing/grinding machine.

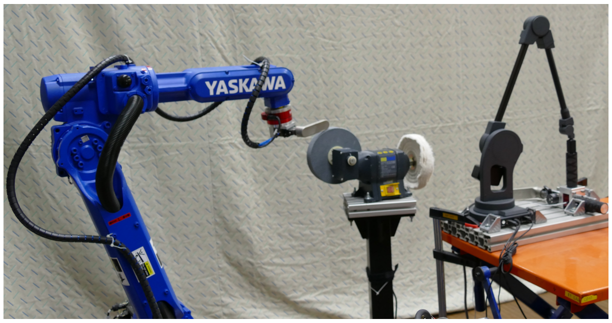

Fig. 7. Experimental polishing/grinding cell.

The PC computer handles the learning by demonstration process and calculates contact points and relative coordinates from the demonstrated motion. It is also used to encode relative coordinates as DMPs and forces as a linear combination of radial basis functions. DMPs encoding the relative coordinates and the demonstrated forces encoded as a linear combination of radial basis functions are passed to the MOTOMAN FS100 controller. Redundancy resolution (17) is implemented on the robot controller FS100 in Yaskawa MotoPlus programming language (C library for robot control).

To show the efficiency and performance of the proposed framework, we demonstrated many faucet polishing trajectories and executed them with and without applying virtual mechanism. Without the VM, the robot was not able to perform the demonstrated trajectory as the execution violated the joint limits, as it can be seen from the image sequence in Fig 8. On the other hand, when VM approach was used, the robot did not need to play back the demonstrated motion directly. Instead, only the relative motion that is relevant for the task was executed. Thus the robot could successfully execute the demonstrated trajectory while minimizing the joint velocities. The corresponding robot motion is shown in Fig 9. See also the video attached to this paper. Fig. 10 shows that after 5 adaptation cycles, the robot also substantially reduced the difference between the demonstrated and the measured forces. Note that VM minimizes joint velocities, which results in generally better force tracking. We can additionally diminish the force tracking error by speed scaling of the learned DMP using the temporary scaling function $\nu(s)$.

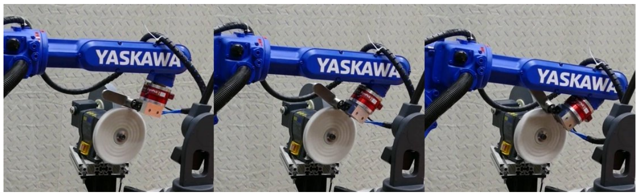

Fig. 8. Image sequence for the task execution without VM. The robot fails due to the joint limits violation.

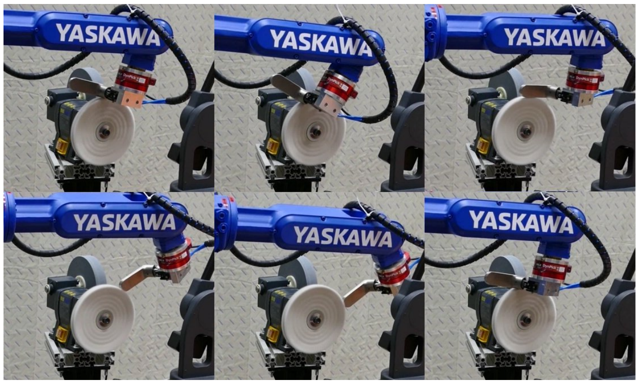

Fig. 9. Image sequence for the task execution with VM. The robot executed the task successfully.

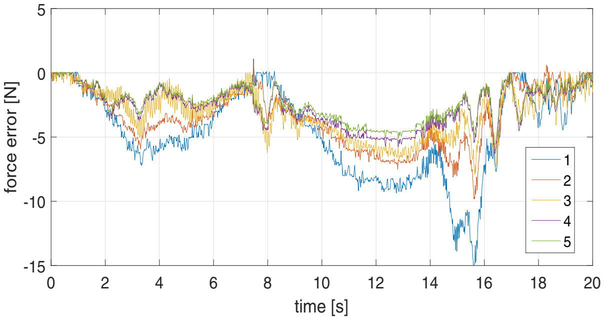

Fig. 10. Norm of the force error in 5 adaptation cycles. Legend relates to the number of the adaptation cycle.

VII. Conclusions

In this paper we proposed a new approach for learning finishing operations by demonstration. Several important novelties were introduced:

- Using the VM approach, only the relevant part of the demonstrated motion can be reproduced, i.e. the relative motion between the treated object and the polishing/grinding tool. Thus the system becomes redundant even though the applied robot only has six degrees of freedom and 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.
- The contact point and the VM coordinates are determined from the measured forces and torques. A geometrical model of the treated object is thus not needed, which is important for many industrial applications.

- 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.

The direct benefits of the proposed approach is that the actual finishing policy is optimized taking into account the kinematic/dynamic capabilities of the robot rather than human operator. The proposed VM approach enables the transfer of the demonstrated policy to different robots and tools without any modification.

We have performed more than 100 polishing and grinding experiments. Our experiments show clear advantages of the VM approach. In more than half of experiments, the robot was able to perform the demonstrated motion by applying the VM, whereas it failed without it. We did not encounter any case where the VM approach would degrade the performance.

Our future work will focus on dynamics of the VM approach, modeling the impedance of the virtual mechanism, and implementation of the proposed framework in industrial plants.

Acknowledgement: This research has been funded in part by the EU's Horizon 2020 Innovation Action ReconCell (GA no. 680431) and by GOSTOP programme C3330-16-529000 co-financed by Slovenia and EU under ERDF.

References

[1] B. G. Antonelli D., Astanin S., "Applicability of human-robot collaboration to small batch production," in *Collaboration in a Hyperconnected World. PRO-VE 2016. IFIP Advances in Information and Communication Technology*, L. S. A. Afsarmanesh H., Camarinha-Matos L., Ed. Springer, 2016, vol. 480.

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

[3] L. Basanez and J. Rosell, "Robotic polishing systems," *IEEE Robotics Automation Magazine*, vol. 12, no. 3, pp. 35–43, 2005.

[4] A. D. Wilbert, B. Behrens, C. Zymla, O. Dambon, and F. Klocke, "Robotic finishing process: An extrusion die case study," *CIRP Journal of Manufacturing Science and Technology*, vol. 11, pp. 45 – 52, 2015.

[5] L. Zhou and H. Huang, "An automated robotic system for jet engine overhaul. system design and development for honeycomb repair," *The International Journal of Advanced Manufacturing Technology*, vol. 19, no. 5, pp. 370–376, 2002.

[6] H. Huang, L. Zhou, X. Chen, and Z. Gong, "Smart robotic system for 3d profile turbine vane airfoil repair," *The International Journal of Advanced Manufacturing Technology*, vol. 21, no. 4, pp. 275–283, 2003.

[7] 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 *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 1374–1379.

[8] J. Zhang, Y. Wang, and R. Xiong, "Industrial robot programming by demonstration," in *2016 International Conference on Advanced Robotics and Mechatronics (ICARM)*, 2016, pp. 300–305.

[9] M. J. E. Kalt, R.P. Monfared, "Development of an intelligent automated polishing system," in *Proceedings of the 16th International Conference of the European Society for Precision Engineering and Nanotechnology (EUSPEN)*, 2016.

[10] A. Skoglund, R. Palm, and T. Duckett, "Towards a supervised dyna-q application on a robotic manipulator," in *Advances in Artificial Intelligence in Sweden*, T. R. P. Funk and N. Xiong, Eds., 2005, pp. 148–153.

[11] A. Balijepalli and T. Kesavadas, "A haptic based virtual grinding tool," in *11th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems, 2003. HAPTICS 2003. Proceedings.*, 2003, pp. 390–396.

[12] 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 *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 1246–1253.

[13] E. Kalt, R. Monfared, and M. Jackson, "Towards an automated polishing system: capturing manual polishing operations," *International Journal of Research in Engineering and Technology*, vol. 7, no. 5, pp. 182–192, 2016.

[14] 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 Transactions on Automation Science and Engineering*, vol. 14, no. 2, pp. 1020–1030, 2017.

[15] 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. 328–373, 2013.

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

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

[18] N. Likar, B. Nemec, L. Žlajpah, S. Ando, and A. Ude, "Adaptation of bimanual assembly tasks using iterative learning framework," in *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, 2015, pp. 771–776.

[19] K. Kitagaki, T. Ogasawara, and T. Suehiro, "Contact state detection by force sensing for assembly tasks," Las Vegas, NV, pp. 366–370, 1994.

[20] K. Kitagaki, M. Fujiwara, T. Suehiro, and T. Ogasawara, "Pseudo contact point monitoring for contact state estimation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Kyongju, Korea, 1999, pp. 832–837.

[21] E. A. Erlbacher, "Force control basics," *Industrial Robot: An International Journal*, vol. 27, no. 1, pp. 20–29, 2000.

[22] D. Bristow, M. Tharayil, and A. Alleyne, "A survey of iterative learning control," *IEEE Control Systems Magazine*, vol. 26, no. 3, pp. 96–114, 2006.

[23] M. Norrlof and S. Gunnarsson, "Experimental comparison of some classical iterative learning control algorithms," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 4, pp. 636–641, 2002.

[24] F. Tian, Z. Li, C. Lv, and G. Liu, "Polishing pressure investigations of robot automatic polishing on curved surfaces," *The International Journal of Advanced Manufacturing Technology*, vol. 87, no. 1, pp. 639–646, 2016.

[25] M. Sonehara, K. Hayashi, K. Nishijima, and H. Murakami, "Development of automation technology for precision finishing works employing a robot arm," *IHI Engineering Review*, vol. 45, no. 2, pp. 16–22, 2013.

[26] F. J. Abu-Dakka, B. Nemec, J. A. Jorgensen, T. R. Savarimuthu, N. Krüger, and A. Ude, "Adaptation of manipulation skills in physical contact with the environment to reference force profiles," *Autonomous Robots*, vol. 39, no. 2, pp. 199–217, 2015.

[27] B. Nemec, T. Petrič, and A. Ude, "Force adaptation with recursive regression iterative learning controller," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015, pp. 2835–2841.