IEEE Robotics and Automation Letters (RAL) paper presented at the
2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Robotic Deep Rolling with Iterative Learning Motion and Force Control

Shuyang Chen[1], Zhigang Wang[2], Abhijit Chakraborty[2], Michael Klecka[2], Glenn Saunders[3], and John Wen[4]

***Abstract*— Large industrial robots offer an attractive option for deep rolling in terms of cost and flexibility. These robots are typically designed for fast and precise motion, but may be commanded to perform force control by adjusting the position setpoint based on the measurements from a wrist-mounted force/torque sensor. Contact force during deep rolling may be as high as 2000 N. The force control performance is affected by robot dynamics, robot joint servo controllers, and motion-induced inertial force. In this paper, we compare three deep rolling force control strategies: position-based rolling with open-loop force control, impedance control, and gradient-based iterative learning control (ILC). Open loop force control is easy to implement but does not correct for any force deviation. Impedance control uses force feedback, but does not track well non-constant force profiles. The ILC augments the impedance control by updating the commanded motion and force profiles based on the motion and force error trajectories in the previous iteration. The update is based on the gradient of the motion and force trajectories with respect to the commanded motion and force. We show that this gradient may be generated experimentally without the need of an explicit model. This is possible because the mapping from the commanded joint motion to the actual joint motion is nearly identical for all joints in industrial robots. We have evaluated the approach on the physical testbed using an ABB robot and demonstrated the convergence of the ILC scheme. The final ILC tracking performance of a trapezoidal force profile improves by over 70 % in terms of the RMS error compared with the impedance controller.**

I. INTRODUCTION

Surface treatment processes, including roller burnishing, deep rolling, and laser peening are commonly utilized to improve performance of a wide variety of commercial and aerospace products [1]. Of all these methods, the deep rolling process is of particular interest because it may be customized according to part geometry and the required compressive residual stress profile to enhance part fatigue life and strength [2]. Deep rolling is a mechanical surface treatment method in which the workpiece surface is exposed to high local mechanical load using a spherical or cylinder-like tool to induce work hardening and compressive residual stress in the near-surface material. Computer numerical controlled (CNC) platforms are typically used in industry due to their rigidity, but they are expensive with low throughput. Furthermore, many CNC platforms are not designed to accommodate the high spindle loads required for rolling/burnishing. Large robotic systems could provide an excellent alternative to traditional CNC platforms for deep rolling (as illustrated in Fig. 1) with the advantages of lower cost, higher throughput, and process flexibility for parts.

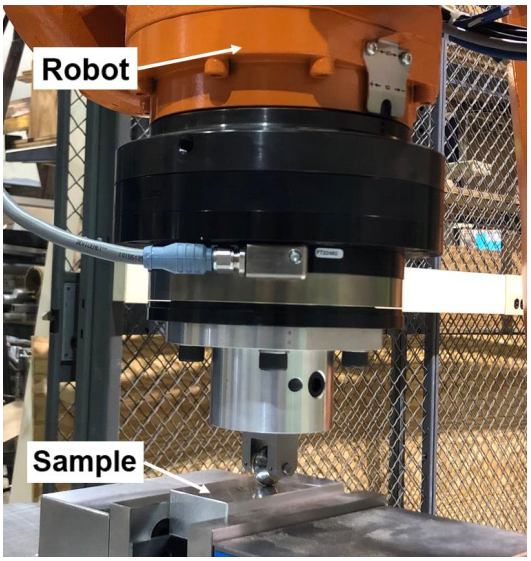


Fig. 1. The robotic deep rolling process.

Industrial robots typically only allow users to access the external position and velocity control interfaces, e.g., the Robot Sensor Interface (RSI) of Kuka (12 ms sampling rate), Low Level Interface (LLI) of Stäubli (4 ms sampling rate), and Externally Guided Motion (EGM) of ABB (4 ms sampling rate). In this paper, we present and compare three strategies to implement deep rolling using a position-controlled industrial robot. The first approach is position-based rolling with open-loop force control. In this case, a user commands a pre-computed position trajectory that produces the desired contact force and motion in orthogonal directions. However, this approach does not reject force disturbances and is difficult to adopt to non-constant force profiles or curved parts. The second method uses force feedback to implement impedance control [3] to ensure force profile tracking. Such schemes are offered as part of a

[1]S. Chen is with the Department of Mechanical, Aerospace, and Nuclear Engineering, Rensselaer Polytechnic Institute, Troy, NY 12180, USA (e-mail: chens26@rpi.edu).

[2]Z. Wang and A. Chakraborty are Staff Research Engineers, M. Klecka is a Principal Research Engineer, with the Raytheon Technologies Research Center, East Hartford, CT 06118, USA (e-mail: wangzhig@utrc.utc.com; chakraa@utrc.utc.com; kleckama@utrc.utc.com).

[3]G. Saunders is with the Manufacturing Innovation Center, Rensselaer Polytechnic Institute, Troy, NY 12180, USA (e-mail: saundg@rpi.edu).

[4]J. Wen is with the Department of Electrical, Computer, and Systems Engineering, Rensselaer Polytechnic Institute, Troy, NY 12180, USA (e-mail: wenj@rpi.edu).

proprietary add-on feature for some industrial robots, e.g., the ABB force control package [4]. Compared with the open-loop control, the impedance control is robust with respect to the robot and material properties and can reject force disturbances. However, the tracking performance for time-varying or position-dependent force profiles is frequently compromised by the robot dynamics and joint servo controllers. The third approach is based on iterative learning control [5], [6]. The idea is to iteratively update the input based on the error from previous trials. As the method is for motion control, a precise surface deformation model is needed to extend to force control.

In this paper, we introduce a robust data-driven gradient-based ILC algorithm for robotic deep rolling that does not need explicit model information of the system and environment, and only requires the position or velocity command interface for robot control. Our formulation directly takes the desired Cartesian force and motion as input and the measured Cartesian force and motion as output. The standard ILC is mostly based on linear proportional update. In contrast, we use a trajectory-based gradient descent algorithm. Explicit model is not needed for the gradient update computation. Instead, we feed the time reversed force and motion tracking error profiles directly into the physical system to generate the gradient descent direction. We have implemented all three approaches mentioned above and evaluated the tracking performance for a trapezoidal force and motion profile on an ABB industrial robot. The open loop controller is not applicable for such a time-varying force profile. The ILC controller improves the RMS tracking error by over 70 % compared with impedance control. To our best knowledge, this is the first theoretical and experimental work that applies ILC for robotic deep rolling. To summarize, the contributions of this study include:

1) Introduction of an ILC algorithm for motion and force trajectory tracking that uses the physical process itself to generate the gradient descent direction and does not require explicit model information.
2) Development of convergence condition of the ILC algorithm.
3) Demonstration of the applicability of the proposed ILC algorithm to the commanded position control interface available in many industrial robots.
4) Experimental demonstration of the feasibility of the approach for deep rolling with a desired time-varying force profile.

For the rest of the paper, Section II summarizes related work on force control. Section III describes our data-driven gradient-based ILC algorithm and its convergence condition. Section IV shows the applicability of the ILC algorithm to the motion and force trajectory tracking problem. The testbed system and experimental results are presented in Section V.

II. Related Work

Classical techniques which tackled the force control problem mostly focused on model-based control by feedback linearization [7]–[9], impedance control [3], and in some cases model-independent control. For instance, Raibert *et al.* [7] proposed one of the earliest hybrid position/force control algorithm based on feedback linearization and development of two complementary orthogonal subspaces to decouple the control of position and force. In [10], Anderson *et al.* proposed hybrid impedance control that combined impedance control and hybrid position/force control. Impedance control tries to command the robot to mimic a specified impedance. However, without cancellation of the nonlinear robot dynamics, the motion-induced inertial force can compromise the force tracking performance.

Force control techniques based on universal function approximators such as neural networks (NNs) for robot dynamics model approximation have been proposed [11]–[13]. For example, Rani *et al.* [13] designed a neural adaptive controller based on radial basis function NNs for hybrid force and position control of a simulated two-link robot manipulator.

Iterative learning control performs well in motion control [14] and has been evaluated with industrial robots [6]. It has also been extended to impedance control, admittance control, and force control. For the case of impedance control [15]–[17], a learning controller was designed such that a target impedance behavior was followed. For admittance control [18], an ILC algorithm was proposed to gradually eliminate interaction force for a peg-in-hole assembly task. For robot force control [19]–[21], experimental evaluation on physical systems were limited. Jeon *et al.* [22] proposed a torque-level learning controller that can improve position and force tracking performance based on error information from previous trials, given sufficiently accurate knowledge of the system model. Visioli *et al.* [23] presented an ILC algorithm for the contour tracking task of a piece of unknown shape without a time-based reference signal. Tahara *et al.* [24] developed an ILC method for force/position trajectory tracking with a robot subject to non-holonomic rolling constraints. However, the input was at the torque level, and only simulation results were presented. A hybrid control strategy for dual-arm manipulation task using ILC was proposed in [25], where the measured position and force errors were fused by a Kalman filter with an experimentally identified contact model, and the fused position error was utilized by ILC for manipulator position compensation.

Drawbacks of the aforementioned works which pose a problem for their application in deep rolling include dependence on an accurate model of the robot and/or the environment, the requirement of a robot joint torque control interface, computational complexity, lack of experimental evaluation for large target force levels. A preliminary investigation of the use of an industrial robot as an alternative to CNC machines for deep rolling process was reported in [26] but it did not address the force control algorithm. The goal of this study is to lower the barrier of using industrial robots for surface treatment processes by improving the robot force control accuracy while retaining the benefits of flexibility and lower cost.

III. ILC for Nonlinear Systems

Let $L_2^n[0,T]$ denote $\mathbb{R}^n$ square integrable functions on $[0,T]$ with the inner product $< v, w >= \int_0^T v(t)^T w(t)\,dt$ and norm $\|v\|^2 = \int_0^T v(t)^T v(t)\,dt$. Consider the mapping of an input trajectory $\underline{u} \in L_2^n[0,T]$ to an output trajectory $\underline{y} \in L_2^n[0,T]$ with a fixed initial condition, denoted by $\underline{y} = \mathcal{G}(\underline{u})$ where $\mathcal{G} : L_2^n[0,T] \to L_2^n[0,T]$ is an operator, possibly nonlinear. Given a desired output $\underline{y}_d \in L_2^n[0,T]$, the goal is to find $\underline{u}$ to minimize $\left\|\underline{y} - \underline{y}_d\right\|$ where the norm is in the $L_2^n[0,T]$ sense. Iterative learning control is basically an iterative algorithm to update $\underline{u}$ to reduce the tracking error $\underline{e} := \underline{y} - \underline{y}_d$:

$$\underline{u}_{k+1} = \underline{u}_k - \alpha \mathcal{L}(\underline{e}_k), \tag{1}$$

where α is sometimes called the learning rate, and k is the iteration number. In the ILC literature, the update law $\mathcal{L}$ does not explicitly depend on the model parameters, but needs certain properties of $\mathcal{G}$ to ensure convergence (e.g., gain, passivity). Many ILC algorithms have been developed for linear systems [27], [28] where $\mathcal{L}$ is chosen to ensure the convergence of $\underline{e}$ to zero. Such technique has been applied to industrial robot motion control based on linearized analysis [29]. For nonlinear systems such as robots, passivity-based convergence condition has been developed in [30].

A. Gradient-Based ILC

The standard gradient descent algorithm to minimize $\|\underline{e}\|$ results in the following ILC update law:

$$\underline{u}_{k+1} = \underline{u}_k - \alpha G_k^* \underline{e}_k, \tag{2}$$

where G_k^* is the adjoint of $G_k := \nabla_{\underline{u}} \mathcal{G}(\underline{u}_k)$, the gradient (Frechét derivative) of $\mathcal{G}$, and α is a sufficiently small constant.

Note that G_k is a linear time varying (LTV) system – the linearization of $\mathcal{G}$ about the state trajectory generated by $\underline{u}_k$. The adjoint G_k^* is also an LTV system. If G_k is approximately time-invariant, then the gradient and its adjoint, G_k and G_k^*, are related through their impulse response kernels [31]:

$$G_k^* = \mathcal{T} G_k^{\mathsf{T}} \mathcal{T}, \tag{3}$$

where $\mathcal{T}$ is the time reversal operator [32], and G_k^{T} is the LTV system with the transpose of the impulse response of G_k. Substituting into (2), we have

$$\underline{u}_{k+1} = \underline{u}_k - \alpha \mathcal{T} G_k^{\mathsf{T}} \mathcal{T} \underline{e}_k. \tag{4}$$

If the impulse response of G_k is symmetric, then $G_k^{\mathsf{T}} = G_k$. This includes the special case when $\mathcal{G}$ is diagonal. In this case, we have

$$\underline{u}_{k+1} = \underline{u}_k - \alpha \mathcal{T} G_k \mathcal{T} \underline{e}_k. \tag{5}$$

It may appear that model information, i.e., $\mathcal{G}$, is needed in implementing this update law. However, as proposed in [14] for a single-input/single-output high speed galvo positioning system, $\mathcal{T} G_k \mathcal{T} \underline{e}_k$ may be obtained by using the time reversal filter with the physical system itself instead of relying on a model.

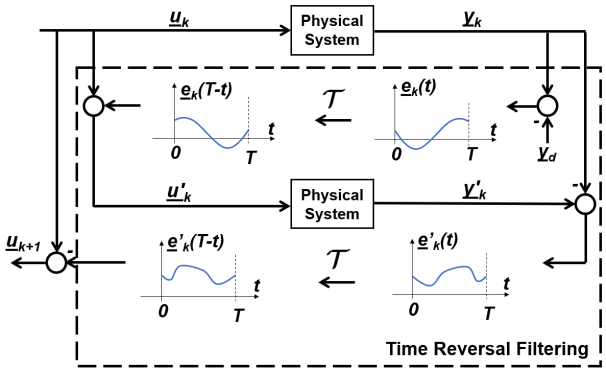


Fig. 2. Block diagram of one iteration of the gradient descent ILC algorithm using time reversal filtering.

Under the symmetric kernel assumption on $\nabla\mathcal{G}$, the $\mathcal{T} G_k \mathcal{T} \underline{e}_k$(here identical to $G_k^* \underline{e}_k$) term in (5) may be computed *without any knowledge of* $\mathcal{G}$ by using the algorithm below (illustrated in Fig. 2):

Algorithm 1. *Given $\underline{u}_k$, apply the following steps to compute $G_k^* \underline{e}_k$:*

(a) *Apply $\underline{u}_k$ to the physical system and obtain the output $\underline{y}_k = \mathcal{G}(\underline{u}_k)$ and tracking error $\underline{e}_k = \underline{y}_k - \underline{y}_d$.*
(b) *Time-reverse $\underline{e}_k(t)$ to $\tilde{\underline{e}}_k(t) := \underline{e}_k(T-t)$.*
(c) *Apply the augmented input $\underline{u}'_k(t) = \underline{u}_k(t) + \tilde{\underline{e}}_k(t)$ to the physical system and obtain the output $\underline{y}'_k = \mathcal{G}(\underline{u}'_k)$ under the same initial configuration as in step (a).*
(d) *Compute $\underline{e}'_k = \underline{y}'_k - \underline{y}_k$, and reverse it in time again to obtain the gradient direction $G_k^* \underline{e}_k(t) = \underline{e}'_k(T-t)$.*

B. Convergence Analysis

Consider the first order variation of $\delta\underline{e}$ resulting from a small change of the input $\delta\underline{u}$:

$$\delta\underline{e} = \underbrace{\nabla_{\underline{u}} \mathcal{G}(\underline{u})}_{:=G} \delta\underline{u}. \tag{6}$$

Substituting in the gradient update law

$$\delta\underline{u} = -\alpha G^* \underline{e}, \tag{7}$$

we have

$$\delta\underline{e} = -\alpha G G^* \underline{e}, \tag{8}$$

which guarantees $G^*\underline{e}$ converges to zero. In the iterative implementation, (7) becomes (2), provided α is sufficiently small. In the model-free implementation of the gradient descent algorithm, under the assumption that G is approximately time-invariant, (7) becomes

$$\delta\underline{u} = -\alpha \mathcal{T} G \mathcal{T} \underline{e}, \tag{9}$$

which becomes (5) in the iterative form. The convergence requires G to have a symmetric kernel, $G^{\mathsf{T}} = G$. However, this property may only be approximately true for the physical system. For example, for a nearly diagonal G, the diagonal portion satisfies the symmetric kernel condition, but we

need to consider the effect of the off-diagonal terms. In this section, we derive a robustness bound for the anti-symmetric portion of the kernel to ensure the convergence of the tracking error in (9).

Decompose G into a symmetric kernel and an anti-symmetric kernel part:

$$G = G_s + G_a, \tag{10}$$

where $G_s^{\mathsf{T}} = G_s$ and $G_a^{\mathsf{T}} = -G_a$. It follows that

$$\begin{aligned} G_s^* &= \mathcal{T} G_s \mathcal{T} \\ G_a^* &= \mathcal{T} G_a^{\mathsf{T}} \mathcal{T} = -\mathcal{T} G_a \mathcal{T}. \end{aligned} \tag{11}$$

The update law (9) is now

$$\begin{aligned} \delta \underline{u} &= -\alpha \mathcal{T}(G_s + G_a)\mathcal{T}\underline{e} \\ &= -\alpha \mathcal{T}(G_s - G_a + 2G_a)\mathcal{T}\underline{e} \\ &= -\alpha \mathcal{T}((G_s + G_a)^{\mathsf{T}} + 2G_a)\mathcal{T}\underline{e} \\ &= -\alpha G^* \underline{e} + 2\alpha \mathcal{T} G_a^{\mathsf{T}} \mathcal{T}\underline{e} \\ &= -\alpha G^* \underline{e} + 2\alpha G_a^* \underline{e}. \end{aligned} \tag{12}$$

Substitute the update law (12) into the error variation equation (6), we have

$$\delta \underline{e} = -\alpha G G^* \underline{e} + 2\alpha G G_a^* \underline{e}. \tag{13}$$

Assume GG^* is positive definite (required for the convergence of $\underline{e}$ in the gradient algorithm to zero), i.e.,

$$\langle \underline{e}, GG^* \underline{e} \rangle \geq \gamma_G \|e\|^2 \tag{14}$$

for some $\gamma_G > 0$. If the anti-symmetric portion of G is sufficiently small:

$$\|G_a\| \leq \frac{\gamma_G}{2\|G\|}, \tag{15}$$

then the iteration (5) will converge.

IV. ILC for Robot Motion/Force Control

Robotic deep rolling involves moving a rigid roller along a pre-determined trajectory while applying a desired force normal to the surface tangent plane containing the path. The task specification is therefore motion along the surface of the part and force orthogonal to the motion direction. However, the control input is the commanded joint position q_c and the measured outputs are the actual joint position q and joint velocity $\dot{q}$. In this section, we will transform the input/output variables in the physical system to the Cartesian Space suitable for motion/force control using the gradient ILC algorithm in Section III-A.

A. Cartesian Input/Output Map

As the desired hybrid force and motion profile is described in the Cartesian Space, we will use the resolved motion controller as shown in Fig. 3. The input is the commanded motion and force, (v_c, f_c), in the orthogonal directions. The commanded force is converted to a velocity command using impedance control (generalized damper):

$$v_s = -K(f - f_c). \tag{16}$$

The combined desired task space velocity, V_T, is converted to the commanded joint velocity and then joint position, q_c, which in turn is fed into the robot controller. The measured joint velocity is mapped to the task velocity in the motion direction. The contact force is directly measured using the force/torque sensor. We consider the transformed input/output map as $y = \mathcal{G}(u)$ where

$$u(t) = \begin{bmatrix} v_c(t) \\ f_c(t) \end{bmatrix}, \quad y(t) = \begin{bmatrix} v(t) \\ f(t) \end{bmatrix}. \tag{17}$$

Similarly, the desired motion and force profile is denoted by $y_d(t)$. Note that since the robot joint controller is stable, $\mathcal{G}$ is a stable (nonlinear) dynamical system.

B. System Characterization

As discussed in Section III-A, a sufficient condition for applying the model-free gradient-based ILC is the near-diagonal structure of the map $\mathcal{G}$. We will consider the condition for the specific robot that we use, but the methodology is applicable to other industrial robots. We employ the ABB IRB6640-180 articulated robot which has six revolute joints, large load capacity (180 kg), and high static repeatability (0.07 mm) and path repeatability (1.06 mm) [33]. We use the EGM module of the ABB IRC5 controller, which allows for joint position commands and joint position and velocity measurements at 4 ms.

The joint servo controllers for industrial robots are typically well tuned so the joint motion follows the commanded motion closely, and there is almost no cross-coupling. Denote the mapping from the commanded joint angle q_c to the actual joint angle q by:

$$q = H(q_c). \tag{18}$$

By applying a chirp signal to q_c, we may obtain the local frequency response of the linearization of H at various robot configurations. Since deep rolling does not involve large changes of the robot configuration, H is approximately a linear time invariant system. Furthermore, the joint servo controllers are tuned to achieve almost the same closed loop behavior. Therefore, H is nearly identical for all joints. This means H may be replaced by a stable scalar transfer function.

Away from singularities, the resolved motion controller is equivalent to the J^{-1} controller. So the joint angle commanded is

$$\begin{aligned} \dot{q}_c &= J^{-1} \cdot V_T = J^{-1} \cdot \begin{bmatrix} v_c \\ -K \cdot (f - f_c) \end{bmatrix} \\ &= J^{-1} \left(\begin{bmatrix} I & 0 \\ 0 & K \end{bmatrix} \begin{bmatrix} v_c \\ f_c \end{bmatrix} + \begin{bmatrix} 0 \\ -K \cdot f \end{bmatrix} \right). \end{aligned} \tag{19}$$

For the output, y, we assume a linear spring model for the surface with the spring constant K_s. When the arm is in contact with the surface, we approximate the system output as:

$$y = \begin{bmatrix} v \\ f \end{bmatrix} \approx \begin{bmatrix} J_m \dot{q} \\ K_s (J_s \Delta q) \end{bmatrix} \approx \begin{bmatrix} t_s^{-1} \cdot I & 0 \\ 0 & K_s \end{bmatrix} J \Delta q, \tag{20}$$

where $\dot{q}$ is approximated by $\Delta q / t_s$. t_s is the sampling period (4 ms for the ABB EGM), and J_m and J_s are the Jacobians

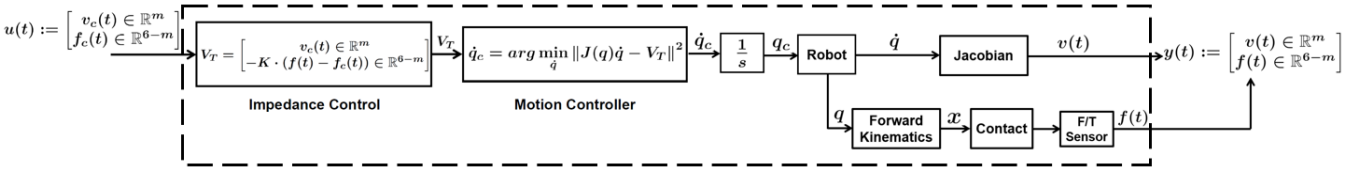


Fig. 3. Structure of the input/output map in the ILC analysis.

in the motion and force directions, respectively. Combining $\Delta q = H(\Delta q_c)$ with (19), we have

$$y \approx \begin{bmatrix} I & 0 \\ 0 & K_s t_s \end{bmatrix} JHJ^{-1} \left(\begin{bmatrix} I & 0 \\ 0 & K \end{bmatrix} u + \begin{bmatrix} 0 \\ -K \cdot f \end{bmatrix} \right). \quad (21)$$

As explained above, we approximate our industrial robot controller by a single linear transfer function for all joints. In that case, H and J^{-1} commute, and we have

$$y \approx \begin{bmatrix} H & 0 \\ 0 & (I + t_s K_s H K)^{-1} t_s K_s H K \end{bmatrix} u. \quad (22)$$

Since H is a scalar and the surface spring is assumed decoupled, we choose the force control gain K to be diagonal, so that the input/output relationship from u to y is approximately diagonal. If the approximation is sufficiently close, then as shown in Section III-B, the ILC algorithm, Algorithm 1, will ensure convergence of $\underline{y}$ to $\underline{y}_d$.

V. Results

A. Experimental Setup

With the ABB robot as described in Section IV-B, a 6-axis ATI Omega160 force/torque sensor capable of measuring forces in the range of $\pm$ 2500 N and with a measurement uncertainty of 1 % is mounted at the robot flange. A hardened-steel roller is connected to the force/torque sensor through a yoke and a roller bearing. The size of the sample is $102.7 \times 48.2 \times 25.4$ mm. The entire testbed is shown in Fig. 4. We have implemented three control approaches: position-based open-loop rolling, impedance control, and iterative learning control. As denoted in Fig. 4, for all experiments, the desired force is in the z direction, and the desired motion is along the y direction, i.e., y_d is in form of:

$$y_d := \begin{bmatrix} 0 & 0 & 0 & 0 & v_d^y & f_d^z \end{bmatrix}^T. \quad (23)$$

The total number of samples of the trajectory is $N = 2000$.

B. Implementations and Comparison

1) Open-Loop Position Control: We first evaluate the open-loop position-based force control for tracking a constant 1000 N force. With the attached force/torque sensor, we first determine the desired position of robot end effector that can produce the contact force by manually jogging the robot to just establishing contact with the material. Then, we command the pre-determined position trajectory using RAPID (the high-level programming language for ABB robots control). The force tracking performance is plotted in Fig. 5.

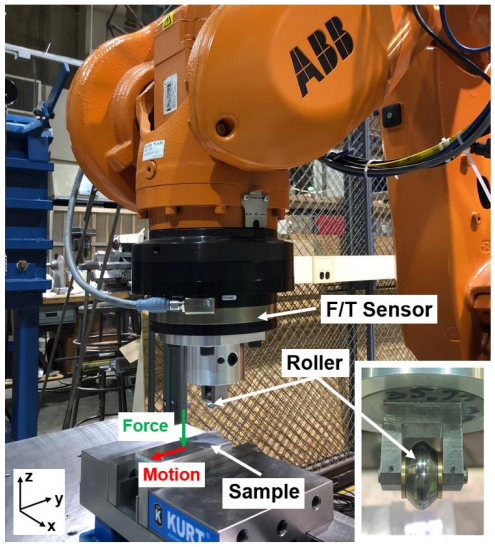


Fig. 4. Experimental setup for the robotic deep rolling.

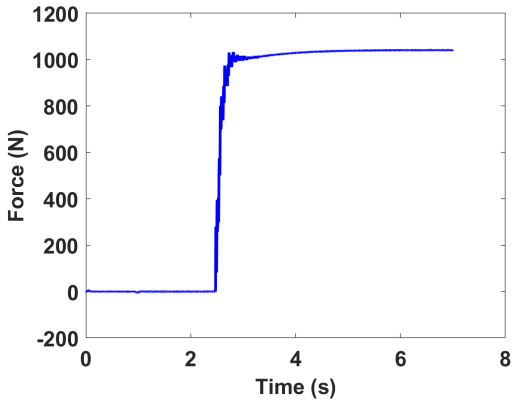
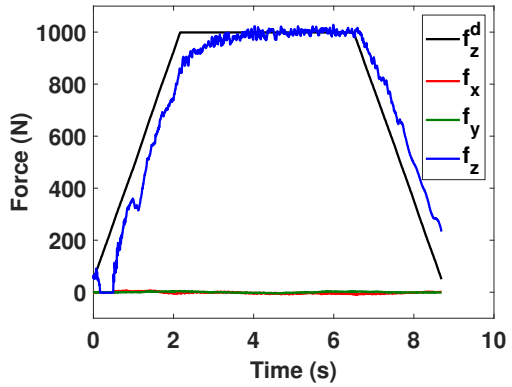


Fig. 5. Force tracking performance for 1000 N desired force in z using open-loop position-based control.

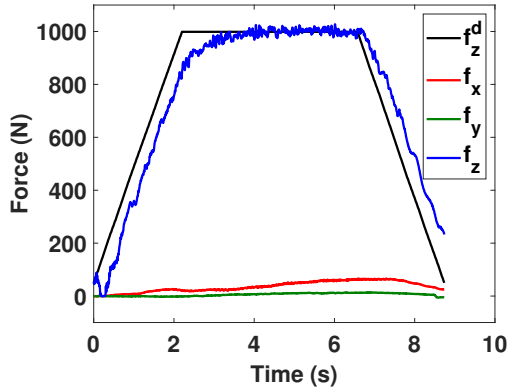
This open-loop control approach is straightforward and easy to implement using the standard robot controller. However, it is difficult to determine the desired position trajectory if the desired force profile is time-varying or position-dependent, or the the part has a curved 3d geometry.

2) Impedance Control: For impedance control, we use the full generalized mass-spring-damper desired impedance driven by the measured force to generate the commanded robot joint position:

$$V_T = \dot{X}_T = K_p(F - F_{ref}) + K_i(X_T - X_{ref}) + K_d \ddot{X}_T, \quad (24)$$



(a)



(b)

Fig. 6. Force tracking performance with different desired impedance. (a) Performance under full impedance control. (b) Performance under generalized damper only (first iteration of ILC).

where X_T is a parameterization of the task space position and orientation of the robot end effector. F_{ref}, X_{ref} are the referenced trajectories. We consider the desired force and motion profiles as trapezoidal in time with the peak force at 1000 N in the z direction. We have tested a group of different set of gains (K_p, K_i, K_d), and the tracking performance corresponding to the best set of gains ($K_p = 0.0002$, $K_d = 0.0008$, $K_i = 0.0004$) is shown in Fig. 6(a). The presence of a large delay in force tracking is evident, due to the dynamics in H.

3) ILC Control: For the ILC implementation, we consider the same force and motion profile as in the impedance control case. Note that we add a step over of 0.25 mm in the x direction for each ILC iteration to ensure rolling is performed on the unrolled material. For the impedance control component as shown in Fig. 3, we only keep the gain $K = 0.0002$ in (16), which is a generalized-damper controller as in the analysis in Section IV. The force tracking performance (shown in Fig. 6(b)) of the first roll (without any iterative refinement) is similar but slightly worse than the result in Fig. 6(a) (more obvious in the force in x direction), where we tune and use all three gains K_p, K_i, K_d. With the desired force and motion profiles as initial input $\underline{u}_0$, we implement 9 ILC runs thus 18 step overs, because for each update we need to roll twice due to the time reversal filtering procedure to experimentally generate the descent direction. For all iterations, we use a fixed learning rate $\alpha = 0.25$.

TABLE I

RMS AND L_∞ NORM OF FORCE TRACKING ERRORS (1000 N PEAK FORCE) IN z

Iteration (No.)	RMSE (N)	L_∞ Norm Error (N)
0	98.3848	185.4981
1	77.6698	150.9880
2	62.0531	149.2758
3	49.3232	122.7392
4	40.4415	119.7123
5	32.9373	101.3830
6	27.1772	81.7024
7	24.3933	103.6499
8	22.2196	103.8649
9	20.7247	83.8624

TABLE II

RMS AND L_∞ NORM OF VELOCITY TRACKING ERRORS IN y

Iteration (No.)	RMSE (mm/s)	L_∞ Norm Error (mm/s)
0	3.1506	4.7358
1	2.6329	3.8745
2	2.2173	3.1618
3	1.8800	2.6507
4	1.6033	2.2064
5	1.3743	1.8320
6	1.1838	2.6493
7	1.0217	1.4880
8	0.8856	2.1225
9	0.7688	1.8215

Figure 7 demonstrates the force tracking performance in z with number of ILC iterations. The RMS and $\|.\|_\infty$ errors for force tracking in z are summarized in Table. I. The RMS of tracking error monotonically decreases with the ILC runs. The $\|.\|_\infty$ of error, although not monotonically decreasing, demonstrates a clearly downward trend. After 9 iterations, the force tracking error in terms of RMS error improves by over 78 %. The comparison of velocity tracking performance in y direction using the impedance controller, with 0 and 9 ILC iterations is illustrated in Fig. 8. Note that as in the force tracking, the velocity tracking performance using the impedance controller is similar to using the control input without any ILC iterations. The RMS and $\|.\|_\infty$ velocity tracking errors of all 9 ILC iterations are summarized in Table. II. The figure and table demonstrate that the velocity tracking improves with increase number of ILC iterations.

The gradient update profiles for the motion and force directions over the iterations are shown in Figures 9-10. The large errors in the ramp up and ramp down portions lead to the input correction in those periods. As the error decreases, the update also becomes small over the entire trajectory.

We also evaluate the proposed ILC algorithm with the same experimental parameters but for a larger magnitude of target force. In this case, the desired force profile is still trapezoidal but with a peak of 2000 N. As summarized in Table. III, after 7 iterations, the tracking accuracy in terms

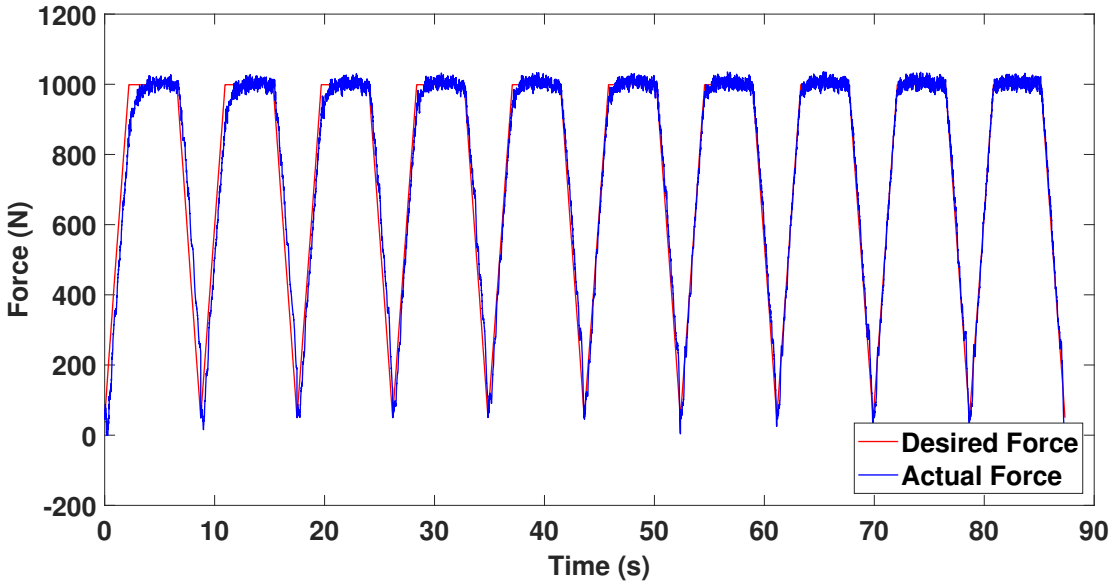


Fig. 7. Force tracking performance in z (1000 N peak force) over 9 ILC iterations. Results from the second roll in each iteration for gradient descent direction computation are omitted.

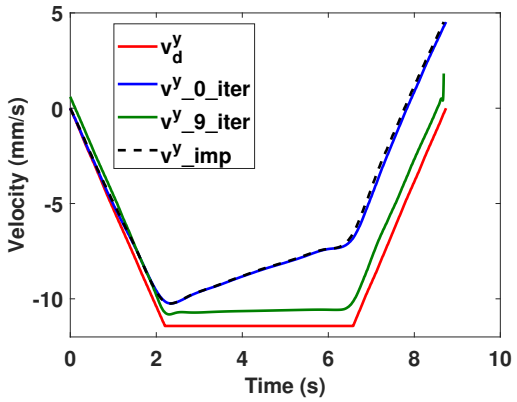


Fig. 8. Comparison of tracking of the desired velocity profile (red) with the impedance controller (black), initial ILC iteration (blue) and ninth ILC iteration (green).

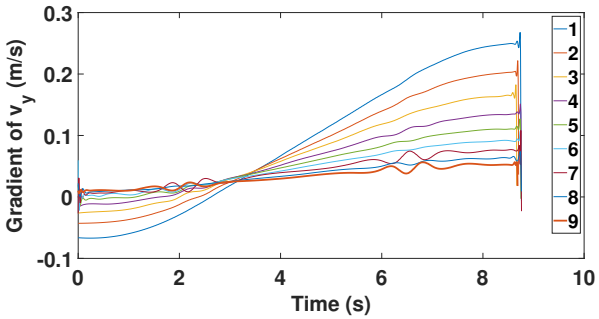


Fig. 9. Gradient update for motion over the nine iterations.

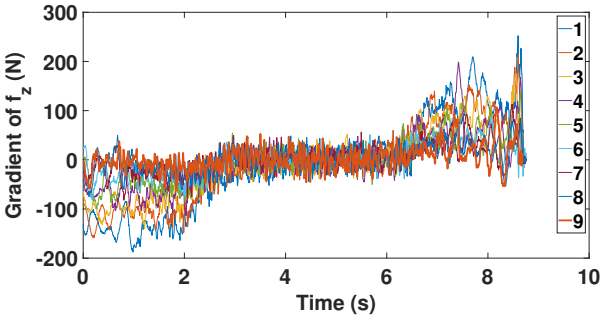


Fig. 10. Gradient update for force over the nine iterations.

of RMS error improves by over 73 %.

VI. Conclusions and Future Work

This paper presents a robust data-driven ILC approach for industrial robotic deep rolling. The method is gradient based but does not require the explicit model information. Instead, the gradient is generated by using the physical process itself. We validate the approach through experiments with a physical testbed using an ABB industrial robot. The proposed ILC method outperforms the impedance control and is applicable to other similar industrial robot platforms that allow externally commanded inputs. The current study has only been tested on a flat sample with relatively slow motion profiles. Future work will include tests of the algorithm with curved surfaces, multi-dimensional desired force profiles, and fast motion trajectories.

Acknowledgment

The authors would like to thank K. Myer for designing and fabricating the roller mount, W. Lawler for assisting with the software implementation, and A. Elias for helping with conducting the experiments.

This work was supported in part by Subaward No. ARM-

TABLE III

RMS AND L_∞ NORM OF FORCE TRACKING ERRORS (2000 N PEAK FORCE) IN z

Iteration (No.)	RMSE (N)	L_∞ Norm Error (N)
0	183.2640	413.1678
1	149.0200	345.3046
2	120.6820	300.3394
3	96.4220	251.4267
4	77.9340	214.2127
5	63.9560	192.5337
6	56.0300	175.6275
7	49.2820	150.0793

TEC-18-01-F-03 from the Advanced Robotics for Manufacturing (ARM) Institute under Agreement Number W911NF-17-3-0004 sponsored by the Office of the Secretary of Defense. ARM Project Management was provided by Cara Mazzarini. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of either ARM or the Office of the Secretary of Defense of the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes, notwithstanding any copyright notation herein. This research was also funded in part by the New York State Empire State Development Division of Science, Technology and Innovation (NYSTAR) Matching Grant Program under contract C170141.

REFERENCES

[1] L. Wagner, T. Ludian, and M. W. M, "Ball-burnishing and roller-burnishing to improve fatigue performance of structural alloys," in *Engineering Against Fracture*, S. Pantelakis and C. Rodopoulos, Eds. Springer, Dordrecht, 2009.

[2] P. Delgado, I. Cuesta, J. Alegre, and A. Díaz, "State of the art of deep rolling," *Precision Engineering*, vol. 46, pp. 1 – 10, 2016.

[3] N. Hogan, "Impedance control: An approach to manipulation," in *1984 American Control Conference*, June 1984, pp. 304–313.

[4] ABB, *Application Manual - Force Control*, ABB Robotics, Västerås Sweden, 2018.

[5] S. Arimoto, "Learning control theory for robotic motion," *International Journal of Adaptive Control and Signal Processing*, vol. 4, no. 6, pp. 543–564, 1990.

[6] M. Norrlöf, "An adaptive iterative learning control algorithm with experiments on an industrial robot," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 2, pp. 245–251, April 2002.

[7] M. H. Raibert and J. J. Craig, "Hybrid position/force control of manipulators," *Journal of Dynamic Systems, Measurement, and Control*, vol. 103, no. 2, pp. 126–133, 06 1981.

[8] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *IEEE Journal on Robotics and Automation*, vol. 3, no. 1, pp. 43–53, February 1987.

[9] T. Yoshikawa, T. Sugie, and M. Tanaka, "Dynamic hybrid position/force control of robot manipulators-controller design and experiment," *IEEE Journal on Robotics and Automation*, vol. 4, no. 6, pp. 699–705, Dec 1988.

[10] R. J. Anderson and M. W. Spong, "Hybrid impedance control of robotic manipulators," *Proceedings. 1987 IEEE International Conference on Robotics and Automation*, vol. 4, pp. 1073–1080, 1987.

[11] S. Jung and T. Hsia, "Neural network techniques for robust force control of robot manipulators," in *Proceedings of Tenth International Symposium on Intelligent Control*, Aug 1995, pp. 111–116.

[12] Y. Li, G. Wang, B. Dong, and B. Zhao, "Hybrid position–force control for constrained reconfigurable manipulators based on adaptive neural network," *Advances in Mechanical Engineering*, vol. 7, no. 9, 2015.

[13] K. Rani and N. Kumar, "Design of intelligent hybrid force and position control of robot manipulator," *Procedia Computer Science*, vol. 125, pp. 42 – 49, 2018, the 6th International Conference on Smart Computing and Communications.

[14] B. Potsaid, J. T. Wen, M. Unrath, D. Watt, and M. Alpay, "High performance motion tracking control for electronic manufacturing," *Journal of Dynamic Systems Measurement and Control*, vol. 129, p. 767–776, 11 2007.

[15] C.-C. Cheah and D. Wang, "Learning impedance control for robotic manipulators," *IEEE Transactions on Robotics and Automation*, vol. 14, no. 3, pp. 452–465, June 1998.

[16] Y. Li and S. S. Ge, "Impedance learning for robots interacting with unknown environments," *IEEE Transactions on Control Systems Technology*, vol. 22, no. 4, pp. 1422–1432, July 2014.

[17] B. Huynh, C. Wu, and Y. Kuo, "Force/position hybrid control for a Hexa robot using gradient descent iterative learning control algorithm," *IEEE Access*, vol. 7, pp. 72 329–72 342, 2019.

[18] J. Bös, A. Wahrburg, and K. D. Listmann, "Iteratively learned and temporally scaled force control with application to robotic assembly in unstructured environments," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 3000–3007.

[19] M. Aicardi, G. Cannata, and G. Casalino, "A learning procedure for position and force control of constrained manipulators," in *Fifth International Conference on Advanced Robotics 'Robots in Unstructured Environments*, June 1991, pp. 423–430 vol.1.

[20] S. R. Pandian and S. Kawamura, "Hybrid force/position control for robot manipulators based on a D-type learning law," *Robotica*, vol. 14, no. 1, p. 51–59, 1996.

[21] P. Lucibello, "A learning algorithm for improved hybrid force control of robot arms," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 28, no. 2, pp. 241–244, March 1998.

[22] D. Jeon and M. Tomizuka, "Learning hybrid force and position control of robot manipulators," in *Proceedings 1992 IEEE International Conference on Robotics and Automation*, May 1992, pp. 1455–1460 vol.2.

[23] A. Visioli, G. Ziliani, and G. Legnani, "Iterative-learning hybrid force/velocity control for contour tracking," *IEEE Transactions on Robotics*, vol. 26, no. 2, pp. 388–393, April 2010.

[24] K. Tahara, S. Arimoto, M. Sekimoto, M. Yoshida, and Z. Luo, "On iterative learning control for simultaneous force/position trajectory tracking by using a 5 DOF robotic thumb under non-holonomic rolling constraints," in *2008 IEEE International Conference on Robotics and Automation*, May 2008, pp. 2611–2616.

[25] B. Chen, Y. Wang, and P. Lin, "A hybrid control strategy for dual-arm object manipulation using fused force/position errors and iterative learning," in *2018 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, July 2018, pp. 39–44.

[26] C. C. Wong, A. Hartawan, and W. K. Teo, "Deep cold rolling of features on aero-engine components," *Procedia CIRP*, vol. 13, pp. 350 – 354, 2014, 2nd CIRP Conference on Surface Integrity (CSI).

[27] H. Ahn, Y. Chen, and K. L. Moore, "Iterative learning control: Brief survey and categorization," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 37, no. 6, pp. 1099–1121, Nov 2007.

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

[29] M. Norrlöf 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.

[30] S. Arimoto, S. Kawamura, and F. Miyazaki, "Bettering operation of dynamic systems by learning: A new control theory for servomechanism or mechatronics systems," in *The 23rd IEEE Conference on Decision and Control*, Dec 1984, pp. 1064–1069.

[31] T. He and W. Chen, "A new interpretation of adjoint method in linear time-varying system analysis," in *8th International Conference on CIS & RAM*, Ningbo, China, 2017.

[32] J. Bolder, "Flexibility and robustness in iterative learning control : with applications to industrial printers," Ph.D. dissertation, Department of Mechanical Engineering, The Eindhoven University of Technology, Eindhoven, Netherlands, 2015.

[33] ABB, "Product Manual-IRB6640," ABB Robotics, Västerås Sweden, 2010.