MITSUBISHI ELECTRIC RESEARCH LABORATORIES
https://www.merl.com

Trajectory Generation for Online Payload Estimation of Robot Manipulators: A Supervised Learning Based Approach

Duan, Xiaoming; Wang, Yebin; Romeres, Diego; Koike-Akino, Toshiaki; Orlik, Philip V.

TR2023-106 September 01, 2023

Abstract

This paper studies the optimal trajectory generation problem for online payload estimation to enable flexible manipulation, where the robotic manipulators pick, transport, and place various types of workpieces. Prevailing work focuses on offline estimation and solve time-consuming optimization problem for the optimal trajectory and initial configuration. By contrast, online estimation requires a quick trajectory generation process where the initial configuration, largely determined by the workpiece and environment layout, is not a design variable. Parameterizing joint trajectories by sinusoidal functions with the amplitudes being design variables, we adopt a supervised learning based approach to fulfill real- time trajectory generation where the mapping from the initial joint positions to the optimal amplitudes is established. This approach shifts the burden of solving computationally intensive and time-consuming trajectory design problems offline and facilitates the fast online generation of identification trajectories. The effectiveness of the trajectory generation method is demonstrated through simulation.

IEEE Conference on Automation and Science Engineering 2023

Mitsubishi Electric Research Laboratories, Inc.
201 Broadway, Cambridge, Massachusetts 02139

Trajectory Generation for Online Payload Estimation of Robot Manipulators: A Supervised Learning Based Approach

Xiaoming Duan[1], Yebin Wang[2], Diego Romeres[2], Toshiaki Koike-Akino[2], Philip V. Orlik[2]

Abstract— **This paper studies the optimal trajectory generation problem for online payload estimation to enable flexible manipulation, where the robotic manipulators pick, transport, and place various types of workpieces. Prevailing work focuses on offline estimation and solve time-consuming optimization problem for the optimal trajectory and initial configuration. By contrast, online estimation requires a quick trajectory generation process where the initial configuration, largely determined by the workpiece and environment layout, is not a design variable. Parameterizing joint trajectories by sinusoidal functions with the amplitudes being design variables, we adopt a supervised learning based approach to fulfill real-time trajectory generation where the mapping from the initial joint positions to the optimal amplitudes is established. This approach shifts the burden of solving computationally intensive and time-consuming trajectory design problems offline and facilitates the fast online generation of identification trajectories. The effectiveness of the trajectory generation method is demonstrated through simulation.**

I. Introduction

Driven by the desire to reduce human intervention in modern manufacturing processes, the past decade has witnessed the burgeoning of research on smart factory. How to enhance autonomy and flexibility of existing factory automation equipment is viewed as an intermediate step toward smart factory. As far as robotic manipulator is concerned, a key ingredient of flexibility is the capability to handle a variety of workpieces. This work investigates one of the key challenges to achieve flexible manipulation: fast trajectory generation for online payload estimation.

An accurate load estimate is crucial for the implementation of precise model-based control algorithms in robotic manipulator applications. The unknown load inertia parameters are usually identified from the dynamic motion data generated by the manipulator. It has long been recognized that sufficient excitation of the manipulator is necessary for the load parameters to be accurately identified. Thus, adequately exciting trajectories shall be designed in order to achieve desirable estimation performance. Moreover, in the context of flexible manipulation, where the load identification task has to be frequently executed, the excitation trajectory should be generated in a fast fashion.

Designing a sufficiently excited trajectory typically resorts to solving complicated optimization problems, where the objective function depends on the trajectory parameters (design variables) in a highly nonlinear manner. Therefore, it is undesirable to repeatedly solve the trajectory design problem whenever a new load is attached and needs to be identified, especially in an online setting. Motivated by this, we propose a supervised learning-based approach to fast trajectory generation, where instances of offline solved design problems are utilized to build a prediction model for online uses. The trajectory parameters can be quickly obtained through the learned model, which leads to a great reduction in the computation time.

The estimation of link and load inertia parameters for robotic manipulators has been widely studied in the literature. Early work [12] and [1] have shown that the dynamic model of the manipulator can be transformed into a linear form in terms of the link/load parameters, and a (weighted) least squares algorithm can be applied to estimate the unknown parameters. Comprehensive procedures for estimating the dynamic parameters of a robotic manipulator were described in [22] and [25]. The authors in [13] proposed an online recursive total least squares method to estimate the load parameters based on the measurements of the force-torque sensor. By exploiting the linear separation of the load and link parameters, the authors in [8] estimate the load parameters under the condition that the link parameters are already known. Identification experiments for both link and load parameters were conducted in [3] where a linear matrix inequality is imposed to ensure parameter feasibility. The authors in [9] proposed an iterative reweighted least squares method to estimate the inertia parameters of the manipulator, where the friction models and physical feasibility of parameters were taken into account. Estimation methods that ensure physical feasibility of the inertia parameters were discussed, for example, in [20], [23], and [14].

Regarding the trajectory design problem for the parameter identification, sinusoidal functions were first proposed to parameterize the excitation trajectory in [21], where the authors also developed a maximum-likelihood parameter estimation method. Fast trajectory design for the parameter identification of robotic manipulators was studied in [10], where the objective function was simplified via approximation. In [5], different excitation trajectories were designed so that different parameters of the payload can be separately identified. The authors in [17] proposed to use the optimized B-splines to parameterize the excitation trajectory. A direct computation of the gradient of a commonly used design criterion, i.e., the condition number of the linear regressor, was developed in [2]. The optimal excitation trajectory

[1]X. Duan is with the Department of Automation, Shanghai Jiao Tong University, Shanghai, China 200240. (email: xduan@sjtu.edu.cn).

[2]Y. Wang, D. Romeres, T. Koike-Akino and P. V. Orlik are with Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA 02139, USA (email: {yebinwang, romeres, koike, porlik}@merl.com).

This work was done when X. Duan was an intern at MERL.

design was also studied for other robotic structures such as in [4].

The design of the excitation trajectory in the aforementioned works is usually achieved through solving nonlinear optimization problems. This is suitable for the one-shot offline identification task where the computation time is not a main issue. However, when the identification task has to be performed from different initial joint configurations rather frequently in the context of flexible manipulation, a faster trajectory generation method becomes indispensable. The main contributions of this paper are as follows. We propose a supervised learning based framework for the fast trajectory generation in load identification tasks. The prediction model is learned offline by solving many instances of the trajectory design problem, and the optimal trajectory is generated online through prediction and correction.

The rest of this paper is organized as follows. We introduce the preliminaries of the load estimation for robotic manipulators in Section II. The main results are presented in Section III. The effectiveness of the proposed approach is demonstrated via simulation in Section IV. Finally, we conclude the paper in Section V.

Notation: Let $\mathbb{R}$, $\mathbb{R}^n$ and $\mathbb{R}^{m\times n}$ be the set of real numbers, the set of real vectors of dimension n and the set of real matrices of dimension m by n, respectively. The bold symbols will be used to denote vectors and matrices. For a vector $\mathbf{v} = \begin{bmatrix} v_1 & v_2 & v_3 \end{bmatrix}^\top \in \mathbb{R}^3$, we define the cross product operator as

$$\hat{\mathbf{v}} = \begin{bmatrix} 0 & -v_3 & v_2 \\ v_3 & 0 & -v_1 \\ -v_2 & v_1 & 0 \end{bmatrix}.$$

For vectors $\mathbf{v}_1, \mathbf{v}_2 \in \mathbb{R}^3$ and $\mathbf{v} = \begin{bmatrix} \mathbf{v}_1^\top & \mathbf{v}_2^\top \end{bmatrix}^\top$, we define

$$\hat{\mathbf{v}} = \begin{bmatrix} \hat{\mathbf{v}}_1 & \mathbf{0}_{3\times 3} \\ \hat{\mathbf{v}}_2 & \hat{\mathbf{v}}_1 \end{bmatrix}.$$

The identity matrix in dimension n is denoted by $\mathbf{I}_n$.

II. Preliminaries

A. Dynamical model of robotic manipulators

The dynamics of an n degrees of freedom serial robotic manipulator is governed by

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{N}(\mathbf{q},\dot{\mathbf{q}}) = \boldsymbol{\tau}, \tag{1}$$

where $\mathbf{q} \in \mathbb{R}^n$ is the generalized coordinate (angular and linear positions for revolute and prismatic joints, respectively), $\mathbf{M}(\mathbf{q}) \in \mathbb{R}^{n\times n}$ is a positive definite mass matrix, $\mathbf{C}(\mathbf{q},\dot{\mathbf{q}}) \in \mathbb{R}^{n\times n}$ is the Coriolis matrix, $\mathbf{N}(\mathbf{q},\dot{\mathbf{q}}) \in \mathbb{R}^n$ includes gravity and other forces such as the friction, and $\boldsymbol{\tau} \in \mathbb{R}^n$ is the actuator torque. The dynamical model (1) contains all the inertia parameters of the links. Specifically, the inertia parameters of the i-th link of the manipulator for $i \in \{1,\dots,n\}$ are given by

$$\boldsymbol{\phi}_i = \begin{bmatrix} I_i^{xx} & I_i^{xy} & I_i^{xz} & I_i^{yy} & I_i^{yz} & I_i^{zz} & m_i\mathbf{p}_i^\top & m_i \end{bmatrix}^\top, \tag{2}$$

where m_i is the mass, $\mathbf{p}_i = \begin{bmatrix} x_i & y_i & z_i \end{bmatrix}^\top$ is the center of mass and

$$\boldsymbol{\mathcal{I}}_i = \begin{bmatrix} I_i^{xx} & I_i^{xy} & I_i^{xz} \\ I_i^{xy} & I_i^{yy} & I_i^{yz} \\ I_i^{xz} & I_i^{yz} & I_i^{zz} \end{bmatrix}$$

is the moment of inertia, all with respect to the origin of the i-th joint frame. The overall inertia parameters of the manipulator can be organized as

$$\boldsymbol{\phi} = \begin{bmatrix} \boldsymbol{\phi}_1^\top & \boldsymbol{\phi}_2^\top & \cdots & \boldsymbol{\phi}_n^\top \end{bmatrix}^\top.$$

When a payload with inertia parameters m_{L}, $\mathbf{p}_{\mathrm{L}}$ and $\boldsymbol{\mathcal{I}}_{\mathrm{L}}$ (in the last joint frame) is attached to the end effector, the inertia parameters of the last link becomes [8]

$$\begin{aligned} m'_n &= m_n + m_{\mathrm{L}}, \\ \mathbf{p}'_n &= \frac{m_n\mathbf{p}_n + m_{\mathrm{L}}\mathbf{p}_{\mathrm{L}}}{m_n + m_{\mathrm{L}}}, \\ \boldsymbol{\mathcal{I}}'_n &= \boldsymbol{\mathcal{I}}_n + \boldsymbol{\mathcal{I}}_{\mathrm{L}}. \end{aligned} \tag{3}$$

As a result, the combined inertia parameters of the manipulator and the payload are

$$\boldsymbol{\phi}' = \begin{bmatrix} \boldsymbol{\phi}_1^\top & \boldsymbol{\phi}_2^\top & \cdots & \boldsymbol{\phi}_{n-1}^\top & {\boldsymbol{\phi}'_n}^\top \end{bmatrix}^\top,$$

where $\boldsymbol{\phi}'_n$ comprises of $m'_n, \mathbf{p}'_n, \boldsymbol{\mathcal{I}}'_n$ given by (3).

B. Identification model and objectives

In order to enable precise model-based control algorithms, the inertia parameters of the links and the payload must be identified. It turns out that the dynamical model (1) (in the absence of friction) can be reorganized in a linear form regarding the inertia parameters (without load)

$$\mathbf{Y}(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}})\boldsymbol{\pi} = \boldsymbol{\tau}, \tag{4}$$

where $\mathbf{Y}(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}}) \in \mathbb{R}^{n\times r}$ is the state-dependent regressor matrix with the number of columns $r \le 10n$ and $\boldsymbol{\pi} \in \mathbb{R}^r$ is the set of identifiable (base) parameters [11, Chap. 12]. The parameter vector $\boldsymbol{\pi}$ consists of linear combinations of the original inertia parameters $\boldsymbol{\phi}$, and its specific form depends on the structure of the manipulator. The identification equation (4) is linear in terms of the identifiable parameters, and the coefficient matrix (a.k.a. the regressor) depends on the state (position, velocity) and acceleration of the manipulator. Since the payload only affects the inertia parameters of the last link, a similar equation as (4) can be derived for the case when a payload is present as follows

$$\mathbf{Y}(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}})\boldsymbol{\pi}' = \boldsymbol{\tau}'. \tag{5}$$

Combining (4) and (5) and assuming that the inertia parameters of the manipulator links are known a priori, we obtain the main identification equation for the load parameters

$$\mathbf{Y}_{\mathrm{L}}(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}})\boldsymbol{\pi}_{\mathrm{L}} = \boldsymbol{\tau}_{\mathrm{L}}, \tag{6}$$

where $\mathbf{Y}_{\mathrm{L}}(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}}) \in \mathbb{R}^{n\times r_{\mathrm{L}}}$, $\boldsymbol{\pi}_{\mathrm{L}} \in \mathbb{R}^{r_{\mathrm{L}}}$ and $\boldsymbol{\tau}_{\mathrm{L}} = \boldsymbol{\tau}' - \boldsymbol{\tau} = \boldsymbol{\tau}' - \mathbf{Y}(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}})\boldsymbol{\pi}$. The identification equation (6) is usually an underdetermined system of linear equations, and multiple samples of $(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}})$ and the corresponding torques $\boldsymbol{\tau}_{\mathrm{L}}$ must be collected so that a (weighted) least squares method can be applied. Suppose N samples of the motion data and torque outputs are available, then we arrive at the following equation

$$\begin{bmatrix} \mathbf{Y}_{\mathrm{L}}(\mathbf{q}(t_1), \dot{\mathbf{q}}(t_1), \ddot{\mathbf{q}}(t_1)) \\ \mathbf{Y}_{\mathrm{L}}(\mathbf{q}(t_2), \dot{\mathbf{q}}(t_2), \ddot{\mathbf{q}}(t_2)) \\ \vdots \\ \mathbf{Y}_{\mathrm{L}}(\mathbf{q}(t_N), \dot{\mathbf{q}}(t_N), \ddot{\mathbf{q}}(t_N)) \end{bmatrix} \boldsymbol{\pi}_{\mathrm{L}} = \begin{bmatrix} \boldsymbol{\tau}_{\mathrm{L}}(t_1) \\ \boldsymbol{\tau}_{\mathrm{L}}(t_2) \\ \vdots \\ \boldsymbol{\tau}_{\mathrm{L}}(t_N) \end{bmatrix}. \quad (7)$$

With a little abuse of notation, we will use $\mathbf{Y}_{\mathrm{L}}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})$ and $\boldsymbol{\tau}_{\mathrm{L}}$ to denote the regressor and the torques in (7), respectively.

In summary, to identify the load parameters $\boldsymbol{\pi}_{\mathrm{L}}$, one needs to sample the motion states $(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})$, construct the regressor $\mathbf{Y}_{\mathrm{L}}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})$, measure and compute the corresponding actuator torques $\boldsymbol{\tau}_{\mathrm{L}}$, and finally solve a (weighted) least squares problem. The solution quality of (6) highly depends on the properties of the coefficient matrix $\mathbf{Y}_{\mathrm{L}}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})$. A well designed trajectory along which the motion samples are taken leads to a regressor with desirable properties and thus helps improve the estimation performance. Following the discussion in [16] and [11, Eq. 12.14], we adopt the optimality criterion for the regressor below in this paper[1]

$$f(\mathbf{Y}_{\mathrm{L}}) = \mathrm{cond}(\mathbf{Y}_{\mathrm{L}}) + \frac{1}{\sigma_{\min}(\mathbf{Y}_{\mathrm{L}})}, \quad (8)$$

where $\mathrm{cond}(\cdot)$ is the condition number and $\sigma_{\min}(\cdot)$ is the minimum singular value. The optimality criterion $f(\mathbf{Y}_{\mathrm{L}})$ takes value from $(1, \infty)$, and an arbitrarily generated identification trajectory can result in a large value of $f(\mathbf{Y}_{\mathrm{L}})$, making the load estimation sensitive to measurement noise. Therefore, the identification trajectory should be carefully designed so that the objective function $f(\mathbf{Y}_{\mathrm{L}})$ is minimized.

C. Trajectory parameterization

Sufficient excitation is necessary for accurate estimation of manipulator parameters. Parameterizing the trajectories by sinusoidal functions is a common practice [22]. For illustration purpose, we parameterize the trajectory of joint i as follows

$$q_i(t) = q_i(0) + \sum_{k=1}^{K} \big(A_k \sin(k\omega_f t) + B_k \cos(k\omega_f t)\big), \quad (9)$$

where $q_i(0)$ is the initial joint position, ω_f is the base frequency of the sinusoidal functions, K is the number of harmonic components, and A_k and B_k are amplitudes. The predetermined design choices for the joint trajectory (9) are ω_f and K, and the amplitudes are the optimization variables that can be tuned so as to minimize the objective function $f(\mathbf{Y}_{\mathrm{L}})$ in (8). The initial joint positions depend on the configuration where the identification task starts, and thus are not design variables. In this paper, we focus on the case when the joint trajectories all have the same amplitudes, i.e., A_k and B_k are not joint dependent. This is to simplify the notation, and the proposed method can be easily extended to heterogeneous cases. On the other hand, we allow the joints to have different initial joint positions. Based on (9), we compute the joint velocity and acceleration as

$$\dot{q}_i(t) = \sum_{k=1}^{K} k\omega_f \big(A_k \cos(k\omega_f t) - B_k \sin(k\omega_f t)\big), \quad (10)$$

$$\ddot{q}_i(t) = \sum_{k=1}^{K} (k\omega_f)^2 \big(-A_k \sin(k\omega_f t) - B_k \cos(k\omega_f t)\big). \quad (11)$$

III. Main results

In this section, for completeness, we first describe a method to numerically construct the regressor $\mathbf{Y}_{\mathrm{L}}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})$ based on the Newton-Euler inverse dynamics, which is seamlessly incorporated into the trajectory design problem later on. Then, we propose a supervised learning based framework to generate an optimal identification trajectory.

A. Regressor construction via the Newton-Euler Algorithm

The recursive computation of the regressor $\mathbf{Y}_{\mathrm{L}}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})$ has been recognized and well known in the literature [15], [24], [26]. Here, we give a description based on [7] and [6, Chap. 5] and specialize it to serial robotic manipulators (instead of manipulators with a tree structure). This treatment allows us to utilize the associated software package. To simplify the exposition, we assign frames that have origins fixed at the joints and rotate with the links. The original Newton-Euler inverse dynamics algorithm is presented in Algorithm 1 with the following notation for $i \in \{1, \dots, n\}$: $\mathbf{v}_i \in \mathbb{R}^6$, $\mathbf{a}_i \in \mathbb{R}^6$ and $\mathbf{S}_i(q_i) \in \mathbb{R}^6$ are the velocity, acceleration and twist of joint i, respectively; $\mathbf{F}_i \in \mathbb{R}^6$ and

$$\boldsymbol{\mathcal{G}}_i = \begin{bmatrix} \boldsymbol{\mathcal{I}}_i & m_i \hat{\mathbf{p}}_i \\ m_i \hat{\mathbf{p}}_i^\top & m_i \mathbf{I}_3 \end{bmatrix} \quad (12)$$

are the wrench and inertia parameters of link i; $\mathbf{Ad}_i(q_i) \in \mathbb{R}^{6\times 6}$ is the adjoint representation of the homogeneous transformation from joint $i-1$ to joint i with joint 0 being a fixed inertia frame; $\mathbf{g} = \begin{bmatrix} 0 & 0 & 9.81 \end{bmatrix}^\top$ represents the gravitational acceleration.

In order to obtain the regressor for the load identification task, we only need to keep track of the impact of the load parameters on the torque output. For a vector $\mathbf{x} \in \mathbb{R}^6$, we define a linear mapping $[\cdot] : \mathbb{R}^6 \mapsto \mathbb{R}^{6\times 10}$ as follows

$$[\mathbf{x}] = \begin{bmatrix} x_1 & x_2 & x_3 & 0 & 0 & 0 & 0 & x_6 & -x_5 & 0 \\ 0 & x_1 & 0 & x_2 & x_3 & 0 & -x_6 & 0 & x_4 & 0 \\ 0 & 0 & x_1 & 0 & x_2 & x_3 & x_5 & -x_4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -x_3 & x_2 & x_4 \\ 0 & 0 & 0 & 0 & 0 & 0 & x_3 & 0 & -x_1 & x_5 \\ 0 & 0 & 0 & 0 & 0 & 0 & -x_2 & x_1 & 0 & x_6 \end{bmatrix}.$$

Then, the following equation holds

$$\boldsymbol{\mathcal{G}}_i \mathbf{x} = [\mathbf{x}] \boldsymbol{\phi}_i,$$

where $\boldsymbol{\mathcal{G}}_i$ and $\boldsymbol{\phi}_i$ are defined in (12) and (2), respectively. Thus, line 4 of Algorithm 1 can be rewritten as

$$\mathbf{F}_i = \boldsymbol{\mathcal{G}}_i \cdot \mathbf{a}_i - \hat{\mathbf{v}}_i^\top \cdot \boldsymbol{\mathcal{G}}_i \cdot \mathbf{v}_i = ([\mathbf{a}_i] - \hat{\mathbf{v}}_i^\top \cdot [\mathbf{v}_i]) \boldsymbol{\phi}_i. \quad (13)$$

[1]The particular objective function can be chosen differently and does not affect the overall framework. For example, the regressor may need to be reweighted when the inertia parameters are badly scaled [16].

Algorithm 1 Newton-Euler Inverse Dynamics Algorithm

Input: The kinematic structure of the manipulator, the inertia parameters $\boldsymbol{\mathcal{G}}_i$ of link i, the position q_i, velocity $\dot{q}_i$ and acceleration $\ddot{q}_i$ of joints i for $i \in \{1, \ldots, n\}$, the initial conditions $\mathbf{v}_0 = \mathbf{0}_6$ and $\mathbf{a}_0 = \begin{bmatrix} 0 & 0 & 0 & \mathbf{g}^\top \end{bmatrix}^\top$

Output: The actuator torque τ_i for $i \in \{1, \ldots, n\}$

Forward Iterations

for $i = 1$ to n **do**

$\quad \mathbf{v}_i = \mathbf{Ad}_i(q_i) \cdot \mathbf{v}_{i-1} + \mathbf{S}_i(q_i) \cdot \dot{q}_i$

$\quad \mathbf{a}_i = \mathbf{Ad}_i(q_i) \cdot \mathbf{a}_{i-1} + \mathbf{S}_i(q_i) \cdot \ddot{q}_i + \hat{\mathbf{v}}_i \cdot \mathbf{S}_i(q_i) \cdot \dot{q}_i$

$\quad \mathbf{F}_i = \boldsymbol{\mathcal{G}}_i \cdot \mathbf{a}_i - \hat{\mathbf{v}}_i^\top \cdot \boldsymbol{\mathcal{G}}_i \cdot \mathbf{v}_i$

end for

Backward Iterations

for $i = n$ to 1 **do**

$\quad \tau_i = \mathbf{S}_i(q_i)^\top \cdot \mathbf{F}_i$

$\quad \mathbf{F}_{i-1} = \mathbf{F}_{i-1} + \mathbf{Ad}_i(q_i)^\top \cdot \mathbf{F}_i$

end for

return τ_i for $i \in \{1, \ldots, n\}$

Algorithm 2 Regressor Construction Algorithm

Input: The kinematic structure of the manipulator, the position q_i, velocity $\dot{q}_i$ and acceleration $\ddot{q}_i$ of joints $i \in \{1, \ldots, n\}$, the initial conditions $\mathbf{v}_0 = \mathbf{0}_6$ and $\mathbf{a}_0 = \begin{bmatrix} 0 & 0 & 0 & \mathbf{g}^\top \end{bmatrix}^\top$

Output: The regressor $\mathbf{Y}_\mathrm{L}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})$

Forward Iterations

for $i = 1$ to n **do**

$\quad \mathbf{v}_i = \mathbf{Ad}_i(q_i) \cdot \mathbf{v}_{i-1} + \mathbf{S}_i(q_i) \cdot \dot{q}_i$

$\quad \mathbf{a}_i = \mathbf{Ad}_i(q_i) \cdot \mathbf{a}_{i-1} + \mathbf{S}_i(q_i) \cdot \ddot{q}_i + \hat{\mathbf{v}}_i \cdot \mathbf{S}_i(q_i) \cdot \dot{q}_i$

end for

$\mathbf{Y}_0 = [\mathbf{a}_n] - \hat{\mathbf{v}}_n^\top \cdot [\mathbf{v}_n]$

Backward Iterations

for $i = n$ to 1 **do**

$\quad \mathbf{Y}_i = \mathbf{S}_i(q_i)^\top \cdot \mathbf{Y}_0$

$\quad \mathbf{Y}_0 = \mathbf{Ad}_i(q_i)^\top \cdot \mathbf{Y}_0$

end for

return $\mathbf{Y}_\mathrm{L}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}) = \begin{bmatrix} \mathbf{Y}_1^\top & \cdots & \mathbf{Y}_n^\top \end{bmatrix}^\top$

Finally, we obtain the regressor by propagating (13) through the backward iterations. We present the numerical method in Algorithm 2.

Since we are interested in the load identification task, the regressor constructed by Algorithm 2 only keeps track of the impacts of the load inertia parameters on the torque outputs. However, similar modifications can be easily applied to construct the regressor for the identification of link parameters. Moreover, given the regressor constructed by Algorithm 2, the base parameters can be determined by a singular value decomposition on $\mathbf{Y}_\mathrm{L}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})$. We use Algorithm 2 in the evaluation and optimization of the objective function $f(\mathbf{Y}_\mathrm{L})$.

B. Supervised learning based trajectory generation

Typically, sufficiently excited trajectories are generated by solving an optimization problem where the amplitudes A_k and B_k of the harmonic functions in (9) are optimization variables and $f(\mathbf{Y}_\mathrm{L})$ in (8) is the objective function. In this subsection, we propose a supervised learning based approach to approximate the optimal trajectory generation process. The model is trained using instances of the solved trajectory generation problems and the excitation trajectory is predicted online given the starting configuration of the joints.

In solving the trajectory generation problem, the following parameters need to be preselected:

- base frequency ω_f of the trajectory in (9);
- the number of harmonic function components K in (9);
- the total execution time t_N in (7) (usually integer multiples of $\frac{2\pi}{\omega_f}$);
- the number of samples N along the trajectory in (7) and how the samples are spread out.

Once the above parameters are fixed, given the initial joint configuration of the manipulator $\mathbf{q}(0)$, we can solve the following optimization problem.

Problem 1 (Trajectory design problem): Given the kinematic structure, the initial joint configuration $\mathbf{q}(0)$ and the motion constraints $\bar{q}$, $\bar{\dot{q}}$ and $\bar{\ddot{q}}$ of the manipulator, select the base frequency ω_f, the number of components K of the trajectory, the total execution time t_N, and the number of samples N, find a sufficiently excited trajectory, i.e., solve the following trajectory design problem:

$$\begin{aligned}
\underset{A_k, B_k \in \mathbb{R}}{\text{minimize}} \quad & \mathrm{cond}(\mathbf{Y}_\mathrm{L}) + \frac{1}{\sigma_{\min}(\mathbf{Y}_\mathrm{L})} \\
\text{subject to} \quad & \mathbf{Y}_\mathrm{L} \text{ is given by (7)}, \\
& q_i(t), \dot{q}_i(t), \text{and } \ddot{q}_i(t) \text{ satisfy (9), (10), (11), resp.,} \\
& \sum_{k=1}^{K} \sqrt{A_k^2 + B_k^2} \le \bar{q}, \qquad (14) \\
& \sum_{k=1}^{K} k\omega_f \sqrt{A_k^2 + B_k^2} \le \bar{\dot{q}}, \qquad (15) \\
& \sum_{k=1}^{K} (k\omega_f)^2 \sqrt{A_k^2 + B_k^2} \le \bar{\ddot{q}}. \qquad (16) \\
& \sum_{k=1}^{K} B_k = 0, \qquad (17) \\
& \sum_{k=1}^{K} k A_k = 0. \qquad (18)
\end{aligned}$$

In Problem 1, the inequalities (14)-(16) encode the physical constraints on the movement range, velocity and acceleration of the joints, where we overestimate the magnitude of the harmonic functions by using the following inequality

$$|A_k \sin(k\omega_f t) + B_k \cos(k\omega_f t)| \le \sqrt{A_k^2 + B_k^2}.$$

The equalities (17) and (18) ensure that the initial joint position is $\mathbf{q}(0)$ and the initial velocity is $\mathbf{0}_6$. The offline training phase consists of the following steps:

1) determine the trajectory parameters ω_f, K, t_N, N and the joint constraints $\bar{q}$, $\bar{\dot{q}}$ and $\bar{\ddot{q}}$;
2) sample initial conditions $\mathbf{q}(0)$ from $\mathcal{Q}$ where $\mathcal{Q}$ is a compact set of possible initial conditions (e.g., $\mathcal{Q} = [-\frac{\pi}{4}, \frac{\pi}{4}]^n$);

3) solve Problem 1 and obtain optimal A_k's and B_k's corresponding to the sampled initial conditions;
4) learn the mapping $\mathcal{M} : \mathbb{R}^n \mapsto \mathbb{R}^{2K}$ that maps the initial conditions $\mathbf{q}(0)$ to the amplitudes (A_k, B_k) using the data set $\{\mathbf{q}(0), \{A_k, B_k\}_1^K\}_1^M$ with $M \geq 1$.

Since the linear regressor $\mathbf{Y}_{\rm L}$ depends on the state of the manipulator, the optimal excitation trajectory is related to the starting configuration of the joints. Therefore, if the mapping $\mathcal{M}$ captures such a relationship, then the optimal trajectory can be generated without solving the optimization problem explicitly. Specifically, once the model $\mathcal{M}$ is learned by a supervised learning approach based on the input ($\mathbf{q}(0)$) and the output (optimal A_k's and B_k's), the online trajectory generation problem is transcribed to a simple function evaluation. The supervised learning method could in principle use any function approximator e.g., a linear model, a Gaussian Process or a neural network, our choices will be outlined in Sec. IV. The predicted trajectory may need to be further processed so that the constraints (14)-(18) are enforced.

Remark 1 (Solution quality): Problem 1 is a highly nonlinear and nonconvex optimization problem, and we usually settle with locally optimal solutions. In fact, finding the exact optimal trajectories may not be the main concern in the identification task; instead, excitation trajectories with small objective values are sufficient in practice.

Remark 2 (Constraints enforcement): Given a joint configuration $\mathbf{q}(0)$, it is difficult to ensure that the predicted amplitudes $\mathcal{M}(\mathbf{q}(0))$ generated by the prediction model satisfy the constraints (14)–(18). However, bringing the predicted amplitudes $\mathcal{M}(\mathbf{q}(0))$ back to the constraint set can be achieved by

1) subtracting the predicted A_k's and B_k's from their (weighted) averages, i.e.,

$$\tilde{A}_k = A_k - \frac{\sum_{k=1}^K kA_k}{\sum_{k=1}^K k},$$
$$\tilde{B}_k = B_k - \frac{\sum_{k=1}^K B_k}{K},$$

so that $\tilde{A}_k$ and $\tilde{B}_k$ satisfy constraints (17) and (18);
2) uniformly scaling down $\tilde{A}_k$'s and $\tilde{B}_k$'s (if necessary) so that constraints (14)–(16) are enforced.

IV. Simulation

This section verifies the effectiveness of the proposed supervised learning based trajectory generation method through simulation. We consider a 6 axis manipulator with all revolute joints. The kinematic structure of the manipulator is specified by the relative positions of the joint origins in the reference configuration. The detailed technical information of the manipulator is proprietary and not disclosed here.

A. Parameter settings

The trajectory parameters adopted in the simulation are documented in Table I. Note that the period of the harmonic functions with the base (lowest) frequency $\omega_f = \frac{\pi}{2}$ is exactly equal to the execution time $t_N = 4$s. We take $N = 100$ samples uniformly along the trajectory, which results in a 25Hz sampling frequency. The velocity and acceleration of the joints are computed using (10) and (11), respectively. Although we do not impose a constraint on the joint velocity, the constraints $\bar{q}$ on the movement range and $\bar{\ddot{q}}$ on the acceleration automatically ensure a reasonable range for the joint velocity. We use Algorithm 2 to construct the objective function and solve Problem 1 by the `fmincon` function in MATLAB.

TABLE I
Trajectory parameters

Parameter	ω_f	K	t_N	N	$\mathcal{Q}$	$\bar{q}$	$\bar{\ddot{q}}$
Value	$\frac{\pi}{2}$	3	4s	100	$[-\frac{\pi}{4}, \frac{\pi}{4}]^6$	$\frac{\pi}{4}$	4rad/s^2

B. Supervised learning based approaches

We consider two common supervised learning approaches in the simulation, i.e., the linear regression with polynomial basis functions and the Gaussian process (GP) [19].

1) Linear regression: In the linear regression model, we parameterize the amplitudes of the optimal trajectory by cubic functions of the initial joint positions, i.e., for $k \in \{1, \ldots, K\}$,

$$A_k = a_0^k + \sum_{i=1}^n a_i^k q_i(0) + \sum_{i,\ell \geq 1}^n a_{i\ell}^k q_i(0) q_\ell(0) + \sum_{i,\ell,s \geq 1}^n a_{i\ell s}^k q_i(0) q_\ell(0) q_s(0),$$

where a^k's are parameters to be learned from the solved trajectory design problems. Similar parameterization also applies to the amplitudes B_k's.

2) Gaussian processes: In the GP model, we parameterize each amplitude of the optimal trajectory by an independent GP, i.e.,

$$y = \mathcal{GP}(\mathbf{x}) + \eta,$$

where y is the amplitude output, $\mathbf{x} \in \mathbb{R}^6$ is the initial joint configuration, and $\eta \sim \mathcal{N}(0, \sigma^2)$ is an independent zero-mean Gaussian random variable with variance σ^2. The kernel function of the GP model is the squared exponential function

$$\mathcal{K}(\mathbf{x}_1, \mathbf{x}_2) = \alpha \exp(-\frac{\|\mathbf{x}_1 - \mathbf{x}_2\|_2^2}{2\ell^2}),$$

where $\alpha > 0$ is the length scale and $\ell > 0$ is the signal variance. The GP model is trained using the companion MATLAB toolbox `GPML` [18] of the book [19], and the hyperparameters α, ℓ and σ^2 are determined by maximizing the marginal likelihood $p(y \mid \mathbf{x})$.

Since the first joint of the studied manipulator has its rotating axis in the vertical direction, the initial position of this joint does not affect the design of the excitation trajectory. Therefore, in both of the supervised learning approaches, the input is the initial positions of the other 5 joints. Note that in this case, we have 56 parameters for each amplitude output in the linear regression model.

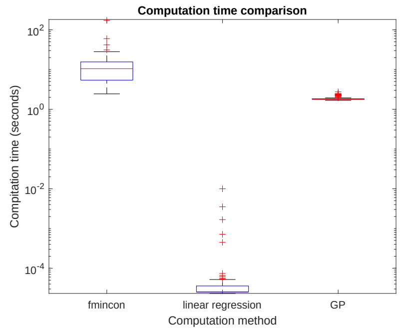

Fig. 1. Comparison of computation time (in logarithmic scale)

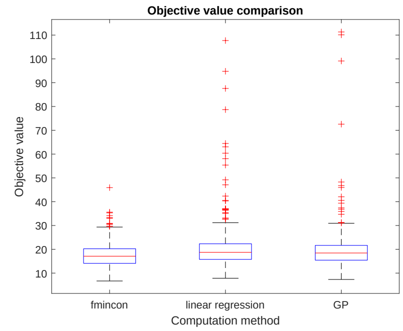

Fig. 2. Comparison of objective values

C. Simulation results

As discussed in Remark 1, the exact trajectories with the optimal amplitudes are not the main concern. Instead, the predicted trajectories should have good performance measured by the metric $f(\mathbf{Y}_{\mathrm{L}})$ as well as the estimation accuracy of load parameters. We uniformly sample $M = 2000$ initial joint configurations from $\mathcal{Q}$, solve Problem 1 and obtain the optimal amplitudes, and train the learning models using the first 1600 data points. We test the prediction performance for the rest 400 different initial configurations.

To ensure that the predicted trajectories lie in the constraint set of Problem 1, we follow the post-processing steps in Remark 2. Specifically, the predicted A_k's and B_k's are first subtracted from their (weighted) averages, respectively. Moreover, if the amplitudes still violate constraints (14)-(16), then they are uniformly scaled down until they are on the boundary of the constraint set, i.e., all the inequalities (14)-(16) are satisfied and at least one of them is tight. On the other hand, it is empirically found that the amplitudes of almost all the optimal excitation trajectories returned by the `fmincon` function are on or close to the boundary of the constraint set. Therefore, we scale up the amplitudes of the predicted trajectories if they are strictly inside the constraint set, until they hit the boundary. This is consistent with the intuition that more modes of the manipulator are excited when the excitation trajectories could reach a larger range.

Computation time: We first compare the computation time for different methods to compute the coefficients of the optimal excitation trajectories. Computations were performed on a 2.6 GHz processor. We generate optimal trajectories for the same set of 400 initial joint configurations in the test set using `fmincon`, the linear regression and GP models, respectively. The computation time are reported in Fig. 1 (in logarithmic scale) as box plots (see `boxplot` function in MATLAB). As can be seen, the offline `fmincon` function takes time that is in the order of 10 seconds or more. In contrast, both the linear regression and GP models consume much less time. In particular, the linear regression model takes time that is in the sub-millisecond range. The outliers in the linear regression case are due to the fact that the time consumption of the post-processing steps described earlier dominates that of the matrix multiplication.

Performance metric $f(\mathbf{Y}_{\mathrm{L}})$: We compare and plot the performance of the optimal trajectories in the test set and the performance of the (post-processed) predicted trajectories by the two learning approaches in Fig. 2. Although the coefficients of trajectories predicted by both learning models do not match well with that of the optimal trajectories found by the `fmincon` function, their performance is decent in terms of the objective value. In fact, from Fig. 2, the majority of the predicted trajectories have comparable performance metrics as that of the optimal ones. Moreover, the GP model has slightly better performance than the linear regression model, in the sense of a lower median value as well as a smaller variance. Since Problem 1 is highly nonlinear, there exist many locally optimal solutions. However, it appears that the objective function is relatively flat in a large region, which partly explains the good performance of the predicted trajectories despite the mismatch between the coefficients of the trajectory parameters.

Estimation accuracy: We compare the estimation accuracy of the payload parameters using optimal trajectories computed by the `fmincon` function and the learned models. Based on (7), the sampled states, and corresponding torques along a trajectory, we formulate the load estimation problem as a least-squares problem with a semidefinite constraint that enforces physical feasibility (mass needs to be nonnegative and the inertia matrix needs to be positive semidefinite). The added noise to the torque measurement on the right hand of (5) follow a Gaussian distribution with a zero mean and a standard deviation being 10^{-2} times the magnitude of the true torque[2]. The estimation accuracy of the load parameters is defined by

$$\frac{\|\hat{\boldsymbol{\pi}}_{\mathrm{L}} - \boldsymbol{\pi}_{\mathrm{L}}\|_2}{\|\boldsymbol{\pi}_{\mathrm{L}}\|_2},$$

where $\hat{\boldsymbol{\pi}}_{\mathrm{L}}$ is the estimated inertia parameters. We repeat the experiment for each trajectory for 10 times and report the average estimation accuracy. As shown in Fig. 3, the estimation performance of the trajectories predicted by both linear regression and GP are very close to those generated by the `fmincon` function (except for a few outliers). This validates the effectiveness of the proposed framework, i.e.,

[2]Since we assume the inertia parameters of the manipulator are known a priori, the torques in (7) can then be computed.

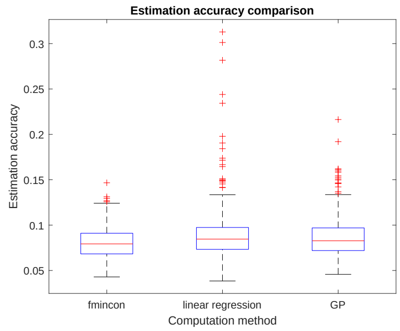

Fig. 3. Estimation accuracy using the predicted trajectories

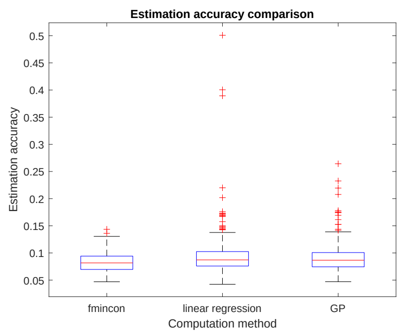

Fig. 4. Estimation accuracy for a different payload

it achieves a tremendous reduction on the computation time and retains comparable performance.

On the other hand, since the optimization of the excitation trajectory does not depend on the particular payload that is applied, we expect the designed trajectories to be robust against the change of payloads. Fig. 4 shows the results of estimation accuracy when we double the mass of the payload (and thus the moment of inertia). Again, comparable estimation accuracy is achieved by using the learned model.

V. Conclusions and Future Work

In this paper, we presented a supervised learning based framework for the trajectory generation in load identification tasks for robotic manipulators. We trained the mapping from the initial joint configuration to the optimal excitation trajectory using the instances of the solved trajectory design problems. As a result, the online trajectory generation is transcribed to simple function evaluations and thus drastically speeds up. For future work, we will study how to achieve effective online load identification without interrupting the normal operations of manipulators, and how to better parameterize the optimal trajectory and generate training data to improve prediction accuracy of the optimal trajectory.

References

[1] C. G. Atkeson, C. H. An, and J. M. Hollerbach. Estimation of inertial parameters of manipulator loads and links. *International Journal of Robotics Research*, 5(3):101–119, 1986.

[2] K. Ayusawa, A. Rioux, E. Yoshida, G. Venture, and M. Gautier. Generating persistently exciting trajectory based on condition number optimization. In *International Conference on Robotics and Automation*, pages 6518–6524, Singapore, May 2017.

[3] A. Bahloul, S. Tliba, and Y. Chitour. Dynamic parameters identification of an industrial robot with and without payload. In *IFAC Symposium on System Identification*, Stockholm, Sweden, July 2018.

[4] V. Bonnet, P. Fraisse, A. Crosnier, M. Gautier, A. González, and G. Venture. Optimal exciting dance for identifying inertial parameters of an anthropomorphic structure. *IEEE Transactions on Robotics*, 32(4):823–836, 2016.

[5] Y. Dong, T. Ren, K. Chen, and D. Wu. An efficient robot payload identification method for industrial application. *Industrial Robot*, 45(4):505–515, 2018.

[6] R. Featherstone. *Rigid Body Dynamics Algorithms*. Springer, 2008.

[7] R. Featherstone. A beginner's guide to 6-D vectors (part 2) [tutorial]. *IEEE Robotics & Automation Magazine*, 17(4):88–99, 2010.

[8] C. Gaz and A. De Luca. Payload estimation based on identified coefficients of robot dynamics — with an application to collision detection. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3033–3040, Vancouver, BC, Canada, September 2017.

[9] Y. Han, J. Wu, C. Liu, and Z. Xiong. An iterative approach for accurate dynamic model identification of industrial robots. *IEEE Transactions on Robotics*, 36(5):1577–1594, 2020.

[10] J. Jin and N. Gans. Parameter identification for industrial robots with a fast and robust trajectory design approach. *Robotics and Computer-Integrated Manufacturing*, 31:21–29, 2015.

[11] W. Khalil and E. Dombre. *Modeling, Identification and Control of Robots*. Butterworth-Heinemann, 2004.

[12] P. K. Khosla and T. Kanade. Parameter identification of robot dynamics. In *IEEE Conference on Decision and Control*, pages 1754–1760, Fort Lauderdale, FL, USA, December 1985.

[13] D. Kubus, T. Kroger, and F. M. Wahl. On-line estimation of inertial parameters using a recursive total least-squares approach. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3845–3852, Nice, France, September 2008.

[14] T. Lee, P. M. Wensing, and F. C. Park. Geometric robot dynamic identification: A convex programming approach. *IEEE Transactions on Robotics*, 36(2):348–365, 2020.

[15] W. Lu and Q. Meng. Recursive computation of manipulator regressor and its application to adaptive motion control of robots. In *IEEE Pacific Rim Conference on Communication, Computers and Signal Processing*, pages 170–173, Victoria, BC, May 1991.

[16] C. Presse and M. Gautier. New criteria of exciting trajectories for robot identification. In *International Conference on Robotics and Automation*, pages 907–912, Atlanta, GA, USA, May 1993.

[17] W. Rackl, R. Lampariello, and G. Hirzinger. Robot excitation trajectories for dynamic parameter estimation using optimized B-splines. In *International Conference on Robotics and Automation*, pages 2042–2047, Saint Paul, MN, USA, May 2012.

[18] C. D. Rasmussen and H. Nickisch. Gaussian processes for machine learning (GPML) toolbox. *Journal of Machine Learning Research*, 11:3011–3015, 2010.

[19] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[20] C. D. Sousa and R. Cortes ao. Physical feasibility of robot base inertial parameter identification: A linear matrix inequality approach. *International Journal of Robotics Research*, 33(6):931–944, 2014.

[21] J. Swevers, C. Ganseman, D. B. Tukel, J. de Schutter, and H. Van Brussel. Optimal robot excitation and identification. *IEEE Transactions on Robotics and Automation*, 13(5):730–740, 1997.

[22] J. Swevers, W. Verdonck, and J. De Schutter. Dynamic model identification for industrial robots. *IEEE Control Systems Magazine*, 27(5):58–71, 2007.

[23] S. Traversaro, S. Brossette, A. Escande, and F. Nori. Identification of fully physical consistent inertial parameters using optimization on manifolds. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5446–5451, Daejeon, South Korea, October 2016.

[24] S. Traversaro, A. D. Prete, S. Ivaldi, and F. Nori. Inertial parameters identification and joint torques estimation with proximal force/torque sensing. In *International Conference on Robotics and Automation*, pages 2105–2110, Seattle, WA, USA, May 2015.

[25] J. Wu, J. Wang, and Z. You. An overview of dynamic parameter identification of robots. *Robotics and Computer-Integrated Manufacturing*, 26(5):414–419, 2010.

[26] J. Yuan and B. Yuan. Recursive computation of the Slotine-Li regressor. In *American Control Conference*, pages 2327–2331, Seattle, WA, USA, 1995.