Robotics and Computer-Integrated Manufacturing 35 (2015) 96–103

Contents lists available at ScienceDirect

Robotics and Computer-Integrated Manufacturing

journal homepage: www.elsevier.com/locate/rcim

Nonparametric statistical learning control of robot manipulators for trajectory or contour tracking☆,☆☆

Cong Wang [a,*], Yu Zhao [a], Yubei Chen [b], Masayoshi Tomizuka [a]

[a] Department of Mechanical Engineering, University of California, Berkeley, CA 94720-1740, United States
[b] Department of Electrical Engineering and Computer Science, University of California, Berkeley, CA 94720-1776, United States

ARTICLE INFO

Article history:
Received 31 July 2014
Received in revised form
15 February 2015
Accepted 10 March 2015
Available online 19 March 2015

Keywords:
Trajectory tracking
Contour tracking
Torque compensation
Reference compensation
Learning control
Nonparametric statistical learning
Gaussian process regression

ABSTRACT

This paper presents a method of precision tracking control for industrial robot manipulators. For robotic laser and plasma cutting tasks, the required tracking performance is much more demanding than that for material handling, spot welding, and machine tending tasks. Challenges in control come from the nonlinear coupled multi-body dynamics of robot manipulators, as well as the transmission error in the geared joints. The proposed method features data-driven iterative compensation of torque and motor reference. Motor side tracking and transmission error are handled by separate learning modules in a two-part compensation structure. Depending on the specific setup of end-effector sensing, the method can utilize either timed trajectory measurement or untimed two-dimensional contour inspection. Nonparametric statistical learning is used for the compensation. Considerations on incorporating analytical models and selecting data subsets for more efficient learning are discussed. The method is validated using a six-axis industrial robot.

1. Introduction

In factory automation systems, robot manipulators and machine tools serve different roles. Although a robot manipulator and a machine tool are both multi-axis servo systems and share similar control hardware, it is significantly more difficult to realize precision contouring control with robot manipulators. This is mainly because machine tools have much higher drive-train stiffness. In addition, the degrees of freedom of a machine tool are arranged in a manner such that basic contours like straight lines and circular arcs can be easily realized by moving as few as one or two motors, whereas even a simple straight line requires that all axes of a robot manipulator move coordinately. Due to these fundamental differences, robot manipulators are conventionally used for material handling, spot welding, and machine tending, but not for machining.

However, recent years have witnessed a fast growing trend of robotic machining. Many advantages of robot manipulators, such as large work range and relatively lower cost, are favorable to flexible manufacturing. Although using robot manipulators for turning and milling still suffers fundamental limitations brought by the large cutting force, robotic laser and plasma cutting are becoming increasingly popular. The tracking performance required by laser and plasma cutting tasks is more demanding than that by material handling, spot welding, or machine tending tasks. In the latter cases, there are typically few requirements on contour tracking. The main emphasis is on final positioning, which can be easily achieved based on the high repeatability and careful calibration of the robot. Precision contour tracking, on the other hand, requires advanced real-time control.

The multi-body dynamics of a robot manipulator is very nonlinear and has strong coupling among individual axes. The computed torque method [1] has become a major technique for controlling robot manipulators. As illustrated in Fig. 1, the computed torque method combines model-based torque feedforward control with decentralized linear feedback control. Accurate torque feedforward control is essential to precision tracking. Dynamics of a robot manipulator, however, is difficult to model and identify precisely. A major challenge lies in the transmission error of the drive-train. Industrial robot manipulators usually have high gear ratio reducers in the drive-train. This allows the use of high-speed low-torque motors which have relatively low costs and light weights. Compliance, backlash, and manufacturing inaccuracy of the reducers, however, introduce transmission error, and the classic multi-rigid body dynamics model cannot describe the actual system behavior perfectly. In addition, the encoders measuring joint rotation are commonly installed on the motor shafts

☆This work was supported by FANUC Corporation.
☆☆Some control hardware and software were donated by National Instruments Corporation.

* Corresponding author.
E-mail address: wangcong@berkeley.edu (C. Wang).

http://dx.doi.org/10.1016/j.rcim.2015.03.002
0736-5845/

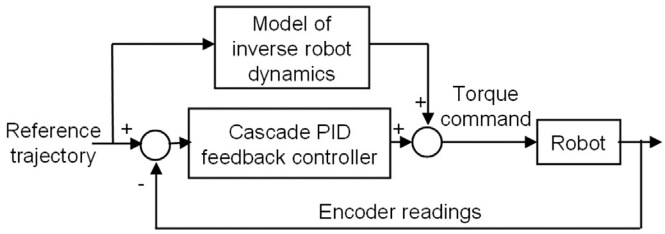

Fig. 1. Computed torque method.

instead on the output shafts of the reducers. In this way, the resolution of the encoders (in terms of measuring joint rotation) can be effectively scaled up by the gear ratio of the reducer. Due to the large reduction ratio, however, deflection caused by the reducer compliance can barely be perceived by the encoders on the motor side [2]. While contour tracking usually requires sub-millimeter accuracy, transmission error and other uncertainties in the kinematic chain can easily cause several millimeters of deviation at the tool. According to [3], 8–10% of the end-effector tracking error comes from drive-train compliance. Precision measurement of the tool motion therefore requires additional sensing such as a laser tracker or a machine vision sensor.

This paper presents a method to realize precision robot tracking control based on trajectory or contour inspection. The method features data-driven iterative compensation that benefits from the repetitive nature of industrial robots' motion. Nonparametric statistical learning is used to compensate for both the feedforward torque and the motor reference. Motor side tracking and uncertainties in kinematic chain are handled by separate learning modules in a two-part compensation structure. Compensation methods based on different setups of end-effector sensing are discussed. In the following sections, different aspects of the method are introduced in detail. Previous work related to each aspect of the proposed method is reviewed in their respective sections. Experimental validation using a benchmark hole cutting trajectory is discussed at the end.

2. Data-driven compensation through learning

Like a human worker refines his skills through experience, machines that execute repetitive tasks can improve their performance by learning from data collected in previous executions. A classic method based on this idea is iterative learning control (ILC). As shown in Fig. 3, the feedforward control is iteratively learned from previous tracking errors by using a learning filter, whose design is based on a convergence condition. Specifically, the convergence conditions of most ILC methods have a similar form as [4–6]

$$\rho(Q(I - LP)) < 1 \tag{1}$$

where $\rho(\bullet)$ is a norm function, I is an identity matrix. L, P, and Q are the descriptions (either in time domain or frequency domain) of the learning filter, the actual system dynamics, and a Q-filter respectively. The low-pass Q-filter improves the transient learning behavior and robustness by restraining the learning activity within finite frequency range. A convergence condition in the form of (1) tolerates uncertainty of the actual system behavior to a certain extent. Effectiveness of disturbance rejection, however, still depends much on how well the actual system behavior is known. Instead of refining feedforward control based on available knowledge of the system response, adaptive control techniques refine the system model online. The dynamic model of a tree-like multibody system is linear with respect to the model parameters [7]. This property allows data-driven online estimate of the model parameters using least squares regression [8]. The multi-rigid body dynamics model, however, cannot accommodate many complex characteristics of the drive-train, such as the transmission error caused by joint compliance, backlash, and manufacturing inaccuracy. Meanwhile, a more sophisticated model that attempts to include those factors always turns out to be overly complicated and difficult to identify.

2.1. Nonparametric statistical learning

The limitation of parameterized models motivates the application of nonparametric statistical learning methods. Rather than indicating that the methods are parameter-free, the term nonparametric means that the mapping from input to output (also called a target) is learned without assuming a model with specific parameterized structure. To avoid confusion, the parameters used in nonparametric learning are called hyper-parameters. A representative method in this class is locally weighted regression (LWR) [9]. Ref. [10] presents a comprehensive discussion on using LWR for learning the inverse dynamics of robot manipulators. One disadvantage of LWR is the use of a large number of hyper-parameters which are difficult to tune [11].

In recent years, Gaussian process regression (GPR) is becoming increasingly popular due to simple implementation and reliable hyper-parameter tuning. Consider a mapping f from a vector input $\mathbf{x}$ to a scalar output $y = f(\mathbf{x}) + \varepsilon$, where ε is the sensing noise with a covariance σ_ε^2. Note that the concepts of input and output used here should not be confused with those of general dynamic systems. Rather than assuming a parameterized structure for f, GPR assumes f to be random, and can be characterized using a mean function and a covariance function (also called a kernel) [12]:

$$m\left(\mathbf{x}\right) = E\left\{f(\mathbf{x})\right\}$$
$$k\left(\mathbf{x}_i, \mathbf{x}_j\right) = E\left\{\left(f(\mathbf{x}_i) - m(\mathbf{x}_i)\right)\left(f\left(\mathbf{x}_j\right) - m\left(\mathbf{x}_j\right)\right)\right\} \tag{2}$$

The kernel characterizes the correlation among input variables. If little a priori information is known about the mapping, a zero mean and a Gaussian kernel are often assumed. Then, given a set of training data points $\left\{\left(\mathbf{x}_{t,i}, y_{t,i}\right) | i = 1 \sim n\right\}$ collected from previous measurement, and a set of query points $\left\{\left(\mathbf{x}_{q,i}, y_{q,i}\right) | i = 1 \sim m\right\}$ whose output $y_{q,i}$'s are to be inferred, the joint distribution of the training data and query points is

$$\begin{pmatrix} \mathbf{y}_t \\ \mathbf{f}_q \end{pmatrix} \sim N\left(\mathbf{0}, \begin{pmatrix} K_{t,t} + \sigma_\varepsilon^2 I & K_{t,q} \\ K_{q,t} & K_{q,q} \end{pmatrix}\right) \tag{3}$$

where $K_{\bullet,*}$ denotes the covariance matrix of $X_\bullet$ and X_*. X_t, X_q, $\mathbf{y}_t$, and $\mathbf{f}_q$ are aggregations of $\mathbf{x}_{t,i}$'s, $\mathbf{x}_{q,i}$'s, $y_{t,i}$'s, and $f\left(\mathbf{x}_{q,i}\right)$'s respectively. Given X_q, X_t, and an actual observation of $\mathbf{y}_t$, the conditional distribution of $\mathbf{f}_q$ has a mean

$$\bar{\mathbf{f}}_q = K_{q,t}\left(K_{t,t} + \sigma_\varepsilon^2 I\right)^{-1}\mathbf{y}_t \tag{4}$$

and a conditional covariance

$$\mathrm{cov}\left(\mathbf{f}_q\right) = K_{q,q} - K_{q,t}\left(K_{t,t} + \sigma_\varepsilon^2 I\right)^{-1}K_{t,q} \tag{5}$$

The conditional mean $\bar{\mathbf{f}}_q$ serves as the estimate of the output of the query points. The conditional covariance $\mathrm{cov}\left(\mathbf{f}_q\right)$ gives the confidence interval of the estimate. Note that if the kernel function indicates that a query point is barely correlated to any training point, GPR still gives an estimate, however, with a wide confident interval. In this sense, GPR does not require the execution to be