CONF-960448--1

SAND95-2197C

Kinematically Optimal Robot Placement for Minimum Time Coordinated Motion

John T. Feddema †
Sandia National Laboratories
P.O. Box 5800, MS 0949
Albuquerque, NM 87111

Abstract

This paper describes an algorithm for determining the optimal placement of a robotic manipulator within a workcell for minimum time coordinated motion. The algorithm uses a simple principle of coordinated motion to estimate the time of a joint interpolated motion. Specifically, the coordinated motion profile is limited by the slowest axis. Two and six degree of freedom (DOF) examples are presented. In experimental tests on a FANUC S-800 arm, the optimal placement of the robot can improve cycle time of a robotic operation by as much as 25%. In high volume processes where the robot motion is currently the limiting factor, this increased throughput can result in substantial cost savings.

Introduction

Industrial robots are often used in high volume manufacturing processes where meeting a specified cycle time is vital to the profitability of the process. Examples include loading a press, inserting electronic components, or spot welding a workpiece. Many times the speed of the robot between taught points is a limiting factor in the process. A good robot programmer can often reduce the cycle time by changing the trajectory of the robot or by changing acceleration/ deceleration times. However, an often overlooked point is that where a robot is mounted to the floor can substantially affect the cycle time between a sequence of points.

The problem of time-optimal control along a specified path has been investigated for several years [1,2]. The objective of these optimizations has been to find a minimum-time path of a robot (with a fixed base) which passes through a set number of points. These algorithms account for the robot's non-linear dynamics, actuator saturation characteristics, joint limits, and, more recently, the presence of obstacles in the workspace [3,4].

This paper addresses a new problem: finding the optimal position of the robot base given a fixed set of points in a world space which the robot end-effector must reach. The proposed optimization algorithm uses only the robot kinematics and the maximum acceleration of each joint as defined by the trajectory generator. For most industrial robot applications, the dynamic effects are negligible for payloads that are less than the robot's nominal payload; therefore, full robot dynamics are not considered here.

As demonstrated in some of the off-line graphical simulation packages [5], one way of finding the optimal base position is to perform an exhaustive search of the entire (x,y,z) position and (roll, pitch, yaw) orientation space of the base with respect to the world coordinate system. This approach may be reasonable if the search space is restricted to just (x,y) space. However, for a full 6 DOF search space, a gradient search method is much more efficient at finding local minimums. This paper will discuss the results of using a steepest descent method as applied to a two link manipulator and a full 6 DOF manipulator.

† This work was performed at Sandia National Laboratories and supported by the U.S. Department of Energy under contract DE-AC04-94AL85000.

Theory

It is assumed that a task has been defined where the robot arm is to move between a starting and ending point as shown in Figure 1. Homogeneous transformations (4x4 matrices representing both position and orientation (pose)) are used to represent the starting pose ${}^{w}T_{s}$ and ending pose ${}^{w}T_{e}$ with respect to the world coordinates w. The pose of the starting and ending positions are stationary because of constraints on the factory floor; therefore, ${}^{w}T_{s}$ and ${}^{w}T_{e}$ are constant matrices. Also, it is assumed that the transformation from the robot end-effector mounting plate to the tool coordinate frame has already been used to compute ${}^{w}T_{s}$ and ${}^{w}T_{e}$. The pose of the robot base with respect to the world coordinates is denoted as ${}^{w}T_{b}$. The objective is to move ${}^{w}T_{b}$ so as to minimize the time required to move between ${}^{w}T_{s}$ and ${}^{w}T_{e}$.

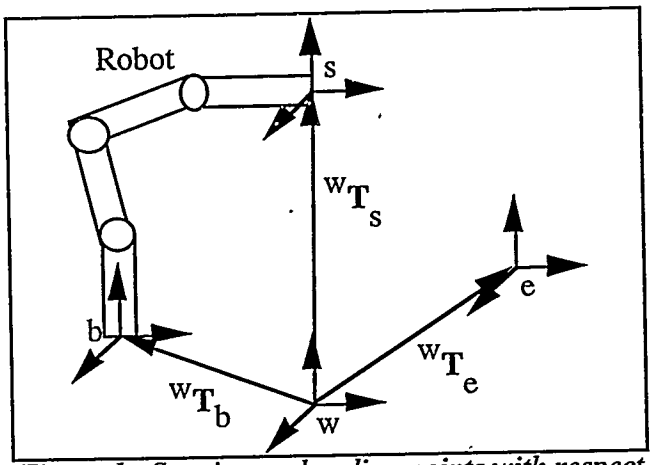

Figure 1. Starting and ending points with respect to the world coordinates are fixed, but the robot base is allowed to move.

Most all industrial robots are joint coordinated devices, meaning that all joints will complete their motion at the same time. If we assume that the maximum acceleration profile for the slowest axis i is a first order acceleration/ deceleration as shown in Figure 2, then the time of the motion for joint i is

$$t_i = 2\sqrt{\frac{|q_{e_i} - q_{s_i}|}{a_{\max_i}}} \qquad (1)$$

where $q_{e_i} - q_{s_i}$ is the joint distance moved and $a_{\max\, i}$ is the maximum acceleration of the axis. For n joints, the time of a coordinated move is

$$t = 2 \max_{i=1,\ldots,n}\left(\sqrt{\frac{|q_{e_i} - q_{s_i}|}{a_{\max_i}}}\right). \qquad (2)$$

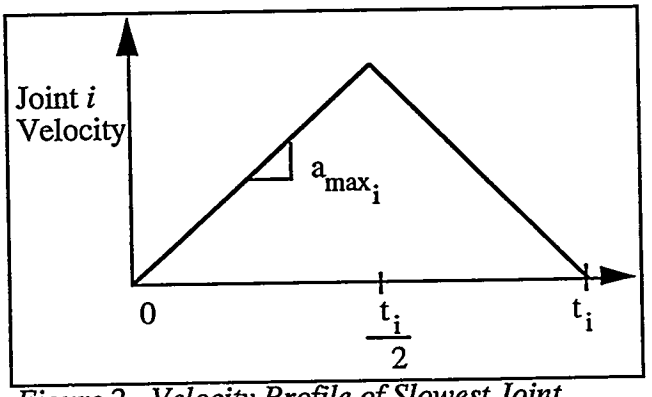

Figure 2. Velocity Profile of Slowest Joint

The vector $q_e - q_s$ is the distance traveled by each joint and is given by

$$q_e - q_s = K^{-1}({}^{b}T_e) - K^{-1}({}^{b}T_s) \qquad (3)$$

where ${}^{b}T_{s}$ and ${}^{b}T_{e}$ are the homogeneous transformations of the starting and ending pose with respect to the robot base, and $K^{-1}(\bullet)$ represents the robot's inverse kinematics. The expressions for the homogeneous transformations can be written as

$${}^{b}T_{s} = \left({}^{w}T_{b}\right)^{-1}{}^{w}T_{s} \qquad (4)$$

and

$${}^{b}T_{e} = \left({}^{w}T_{b}\right)^{-1}{}^{w}T_{e} \qquad (5)$$

where the $(\bullet)^{-1}$ denotes a matrix inversion.

The pose can also be represented as a 6x1 column vector ${}^{w}x_{b}$ with elements x, y, z, yaw, pitch, and roll. The mapping between the

homogeneous transform and ${}^{w}x_b$ will be represented as

$$ {}^{w}T_b = T\left({}^{w}x_b\right). \tag{6} $$

The problem can then be stated as the minimization of time t in Equation (2) while moving the robot base ${}^{w}T_b$ (or ${}^{w}x_b$).

To minimize Equation (2) subject to the constraints in Equations (3), (4), and (5), a steepest descent algorithm can be used to iteratively converge to the solution.

$$ {}^{w}x_b(k+1) = {}^{w}x_b(k) - \alpha_k \left(\left. \frac{\partial t}{\partial {}^{w}x_b} \right|_k \right)^T, \quad \alpha_k > 0 \tag{7} $$

The constant α_k in more elaborate algorithms such as Newton's method and Davidon-Fletcher-Powell method also includes the inverse Hessian relationship [6]. For simplicity, α_k was set to a large number (10^4) at the beginning of the search and decreased when the gradient increased time instead of decreased time. The partial of time with respect to the robot base pose is given by

$$ \frac{\partial t}{\partial {}^{w}x_b} = \frac{\partial}{\partial {}^{w}x_b} \left\langle \max_{i=1,\ldots,n} \left\{ \frac{2}{\sqrt{a_{\max_i}}} \left[\left(q_{e_i} - q_{s_i} \right)^2 \right]^{\frac{1}{4}} \right\} \right\rangle \tag{8} $$

This suggests that the gradient used in the kth iteration of the search should be along the joint which is going to take the longest time to complete it's motion. In implementation, we ranked the joints motion time from longest to shortest, and began the search using the gradient of the longest. When the local minimum along that axis was reached, the next longest ranking joint was used. This was repeated until a local minimum was reached for all joints.

The gradient along joint i may be computed as

$$ \frac{\partial t_i}{\partial {}^{w}x_b} = \frac{1}{\sqrt{a_{\max_i}}} \left[\left(q_{e_i} - q_{s_i} \right)^2 \right]^{-\frac{3}{4}} \left(q_{e_i} - q_{s_i} \right) \cdot \left(\left. \frac{\partial q_i}{\partial {}^{w}x_b} \right|_e - \left. \frac{\partial q_i}{\partial {}^{w}x_b} \right|_s \right) \tag{9} $$

where the partial $\frac{\partial q_i}{\partial {}^{w}x_b}$ is the differential change in joint angles given a differential change in the robot base pose.

This partial is related to the inverse manipulator Jacobian as follows. First notice that since the starting and ending pose do not change, the differential change of these points with respect to the world coordinates is zero.

$$ d{}^{w}x_n = \frac{\partial {}^{w}x_n}{\partial {}^{w}x_b} d{}^{w}x_b + \frac{\partial {}^{w}x_n}{\partial q} dq = 0 \tag{10} $$

Here, ${}^{w}x_n$ is the pose of the robot end-effector with respect to the world coordinates. Therefore, the partial of the joint angles with respect to the base pose is a function of the inverse manipulator Jacobian with respect to the world coordinates [7] and the Jacobian relating changes in base pose to changes in world coordinates.

$$ \frac{\partial q}{\partial {}^{w}x_b} = -\left(\frac{\partial {}^{w}x_n}{\partial q} \right)^{-1} \left(\frac{\partial {}^{w}x_n}{\partial {}^{w}x_b} \right) \tag{11} $$

Often, it is easier to express the manipulator Jacobian with respect to tool coordinates [8]. Therefore, the above expression can be written as

$$ \frac{\partial q}{\partial {}^{w}x_b} = -\left(\frac{\partial {}^{n}x_n}{\partial q} \right)^{-1} \left(\frac{\partial {}^{n}x_n}{\partial {}^{w}x_b} \right) \tag{12} $$

where $\frac{\partial {}^{n}x_n}{\partial q}$ is the manipulator Jacobian as given in [8], and

$$\frac{\partial^n x_n}{\partial^w x_b} = \begin{bmatrix} {}^w R_n^T & {}^w S_n^T \\ 0 & {}^w R_n^T \end{bmatrix}. \qquad (13)$$

The variable ${}^w R_n$ is the 3x3 rotation matrix with column vectors ${}^w n_n$, ${}^w o_n$, and ${}^w a_n$. The matrix ${}^w S_n$ is given by

$${}^w S_n = \left[\left({}^w p_n \times {}^w n_n \right) \quad \left({}^w p_n \times {}^w o_n \right) \quad \left({}^w p_n \times {}^w a_n \right) \right] \qquad (14)$$

where ${}^w p_n$ is the position vector of end-effector with respect to the world coordinates. The superscript T is a transpose operator.

In summary, the steepest descent algorithm (7) in conjunction with Equations (9), (12), (13), and (14) can be used to solve for local minimums of Equation (2).

Two Degree of Freedom Example

Insight into the optimal placement problem can be gained by first analyzing a two DOF problem as shown in Figure 3. A task has been defined which requires the 2 link robot arm to move between two points (70,100) mm and (20,50) mm. With the base of the arm at (0,0) mm, the change in joint angles between the starting and ending points are (-8.80, 43.99) degrees. The length of each link is 100 mm, and the maximum acceleration of each joint is 100 mm/s^2. The motion execution time with the base at (0,0) mm is 1.3264 seconds.

The steepest descent algorithm converged to a local minimum of 1.0804 seconds (a 18.5% improvement) at a base position of (44.56, 5.23). Figure 4 shows a contour plot of time verses robot base position, and the path traveled by the steepest descent algorithm. In the figure, darker regions correspond to shorter motion time. The algorithm converged to 4 decimal places in time in 19 iterations. At this optimal base position, the change in joint angles is (29.18, 29.18) degrees. Since the accelerations and link lengths of each degree of freedom are equal, the optimization moves the robot base so that the distance traveled by each joint is the same.

Now suppose that the acceleration of joint 1 is half that of joint 2 (50 mm/s^2 instead of 100 mm/s^2). The motion execution time with the base at (0,0) mm is still 1.3264 seconds because joint 2 is still the limiting joint. Using the steepest descent algorithm, the optimal base position is (31.87,1.36) with a motion time of 1.1807 seconds (11.0% improvement). Convergence to 4 decimal places in time occurred in 18 iterations. At this optimal base position, the change in joint angles is (17.42,34.85) degrees.

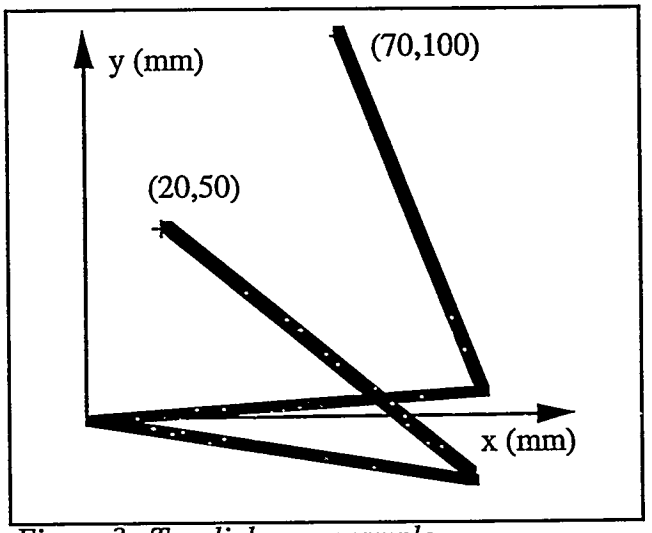

Figure 3. Two link arm example.

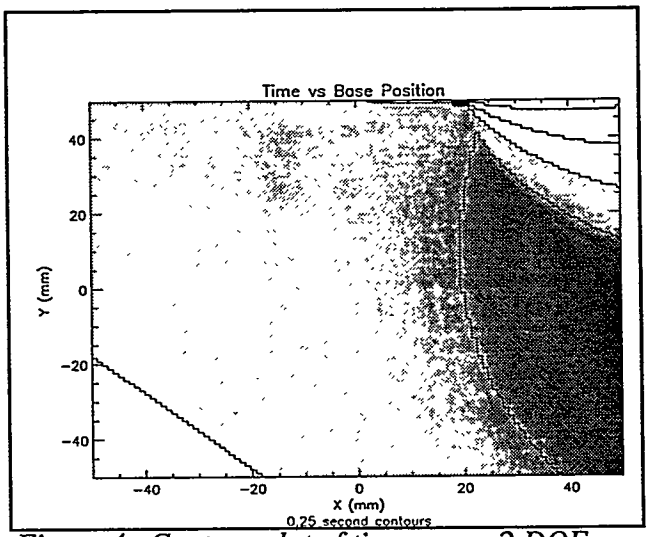

Figure 4. Contour plot of time verses 2 DOF robot base position.

Six Degree of Freedom Example

Next, let us consider the optimal base placement of a six degree of freedom FANUC S-800 robot arm. A maneuver between two points will be discussed. The maneuver is from a starting pose of $(x, y, z; w, p, r) = (1040, 0, 634; 160, -90, 19)$ to an ending pose of $(x, y, z; w, p, r) = (1616, 0, 789; 160, -90, 19)$. The position units are in mm, and the orientation units are in degrees. This motion is a simple translation of the robot end-effector in the positive x and z directions. The change in joint angles between these two points is (-0.10, 47.89, 6.30, -2.09, 6.28, 2.17) degrees. Assuming the maximum acceleration of each joint is 80 degrees/second, the estimated time of the motion is 1.547 seconds.

Using the steepest descent algorithm, the optimal base position relative to it's initial position is at $(x, y, z) = (-1, -160, 730)$. The orientation was not allowed to change appreciably by lowering the values of α_k for the orientation components. Convergence to 3 decimal places in time occurred in 163 iterations. By comparison, an exhaustive search of this precision over a 2000x2000x1000 mm volume would require approximately 10^6 iterations. Figures 5 and 6 show contour plots of time verses the x, y, and z positions. Again, darker regions correspond to shorter motion times, and black corresponds to points which are unreachable. The new starting and ending poses are (1042, 153, -102; 8, -90, 170) and (1619, 149, 50; 8, -90, 170), and the change in joint angles between these two points is (-4.42, 24.28, 24.28, 2.72, 24.20, 3.33) degrees.. The estimated time of motion is reduced to 1.102 seconds (a 29% improvement). Notice that the limiting joint's travel (joint 2) has been nearly halved. The reduction in joint 2's travel was made up for by an increase in the travel of joints 3 and 5. In the optimal position, joints 2, 3, and 5 have approximately equal travel times.

The motions in Figures 5 and 6 were experimentally tested. The non-optimal motion took 1.456 seconds, while the optimal motion took 1.176 seconds (a 19% improvement). While the estimated and actual motion times are relatively inaccurate, the fact that the optimization does result in shorter cycle times is important. In fact, improvements in the 20-25% range typically have been seen in other experiments.

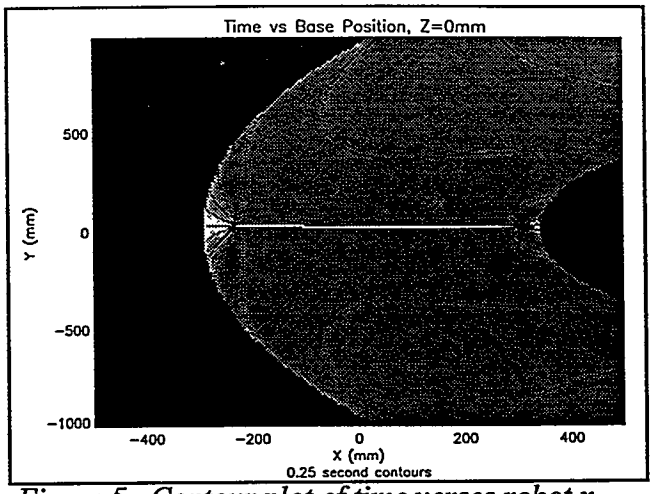

Figure 5. Contour plot of time verses robot x and y base positions.

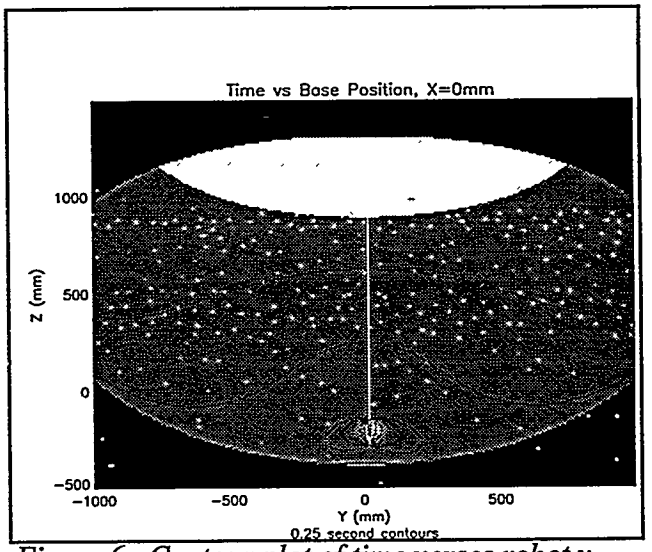

Figure 6. Contour plot of time verses robot y and z base positions.

There are several reasons for the discrepancies between the estimated times and the actual experimental times. The first is that every robot vendor performs trajectory generation differently. On the FANUC controller [9], the trajectory is specified by a minimum acceleration time and maximum joint velocities. In addition, an exponential filter is used to smooth the trajectory. In our case, this filter adds approximately 0.1 seconds to the motion time. To improve the accuracy of the motion time estimate, equation (1) and it's partial would need to be changed to the particular algorithm used by that vendor.

Second, we have not taken into account dynamic effects which may increase the settling time.

DISCLAIMER