Proceedings of the
American Control Conference
San Francisco, California • June 1993

A Suboptimal Robot Path Planning Scheme for Loosely Constrained Trajectories

Dan Simon *
TRW, SB2-1062
PO Box 1310
San Bernardino, CA 92402

Can Isik
Syracuse University
Dept. of Electrical Engineering
Syracuse, NY 13244

Abstract

Approximation of a desired robot path can be accomplished by interpolating a curve through a sequence of joint-space knots. A smooth interpolated trajectory can be realized by using trigonometric splines (TSs). But sometimes the joint trajectory is not required to exactly pass through the given knots. The knots may rather be centers of tolerances *near* which the trajectory is required to pass. In this paper, we optimize TSs through a given set of knots subject to user-specified knot tolerances. The contribution of this paper is the straightforward way in which intermediate constraints (i.e. knot angles) are incorporated into the parameter optimization problem. Another contribution is the exploitation of the decoupled nature of TSs to reduce the computational expense of the problem. The additional freedom of varying the knot angles results in a lower objective function and a higher computational expense, compared to the case where the knot angles are constrained to exact values. The specific objective functions considered are minimum jerk and minimum energy. In the minimum jerk case, the optimization problem reduces to a quadratic programming problem. Simulation results for a two-link manipulator are presented to support the results of this paper.

I. Introduction

The industrial robot is a highly nonlinear, coupled multivariable system with nonlinear constraints. For this reason, robot control algorithms are often divided into two stages: *path planning* and *path tracking*. A conceptually simple approach to the path planning problem is to generate a joint-space trajectory based on interpolation of a sequence of desired joint angles. This approach ignores most of the dynamics of the robot, so the resultant trajectories do not take full advantage of the robot's capabilities. But the trajectories are typically computationally inexpensive, making this approach a popular method [1]. In this approach, a number of knot points are chosen along the desired Cartesian path. The number of knots chosen is a tradeoff between exactness and computational expense. The Cartesian knots are then mapped into joint knots using inverse kinematics. Finally, an analytic interpolating curve is fit to the joint knots. This curve provides the path tracker with joint angles and derivatives at the controller rate.

A recent development [2, 3] is the use of trigonometric polynomials to efficiently generate joint trajectories with little overshoot but continuous velocity, acceleration, and jerk. Trigonometric polynomials have the characteristic that if they are appropriately normalized in time, they are very smooth [4]. That is, the magnitude of the derivatives are relatively low, and the overshoot is relatively small. If piecewise continuous trigonometric polynomials are joined together, the computational expense is low [5], and each polynomial is of low order, preventing oscillations between knots. These piecewise continuous trigonometric polynomials are called *trigonometric splines*, hereafter referred to as TSs.

In this paper, we show how a TS which is required to pas *near* a given set of joint knots can be optimized. The objective functions which are used are minimum jerk and minimum energy. In the minimum jerk case, the problem reduces to a quadratic programming problem with linear constraints. The maximum error at the knots is specified by the user. The unique contribution of this paper is the straightforward way in which the intermediate knot angle constraints are incorporated into the optimization problem. In addition, the decoupled nature of TSs can be taken advantage of to reduce the computational expense of the problem.

II. Trigonometric Splines

In this section, the TSs used in this paper will be defined, and their application to robot path planning will be summarized. See [2, 3] for details.

A desired Cartesian trajectory can be discretized into $(n+1)$ Cartesian goal points at times $t_0 < t_1 < \ldots < t_n$. Then inverse kinematics can be performed at each of these goal points, resulting in a set of $(n+1)$ joint space goal points y_i for each joint. Then n fourth-order trigonometric polynomials $y_i(t)$ can be generated. Fourth-order polynomials are used so that the first three derivatives at each endpoint can be constrained. This allows the user to join the polynomials together so as to have a joint-space path with continuous derivatives up to the third order. The TS segment $y_i(t)$ $(i = 1, \ldots, n)$ is defined *only* on the time interval $[t_{i-1}, t_i]$. The time interval of each segment can be normalized to a *fixed* t_{i-1} and t_i. In this paper, it will be assumed that $t_{i-1} = 0$ and $t_i = \pi/4$, $(i = 1, \ldots, n)$. These values give computational stability and smoothness of motion [4]. Then each TS segment can be written as

$$\begin{aligned} y_i(t) &= a_{i,0} + \sum_{k=1}^{3}(a_{i,k}\cos kt + b_{i,k}\sin kt) + a_{i,4}\cos 4t \\ & \quad t \in [0, \pi/4]. \end{aligned} \tag{1}$$

These n trigonometric polynomials are joined together to form a TS. The eight constraints used to determine the coefficients of $y_i(t)$ are

$$\begin{array}{rclcll} y_i(t_{i-1}) &=& y_{i-1} &\equiv& y(t_{i-1}) & \\ y_i(t_i) &=& y_i &\equiv& y(t_i) & \\ y_i^{(r)}(t_{i-1}) &=& y_{i-1}^{(r)} &\equiv& y^{(r)}(t_{i-1}) & (r = 1,2,3) \\ y_i^{(r)}(t_i) &=& y_i^{(r)} &\equiv& y^{(r)}(t_i) & (r = 1,2,3). \end{array} \tag{2}$$

The first two constraints of (2) are given by the inverse kinematics solution of the Cartesian trajectory. There are several

*This work was performed while the author was with Syracuse University.

different ways to specify the last six constraints of (2). One way is that the user may desire certain joint derivatives at the knots. Another possibility is that these constraints could be determined to minimise some objective function (see Section III). Yet another possibility is that these constraints could be chosen using some simple, heuristic method [5].

Equations (1) and (2) are used to determine the coefficients of the spline segments $y_i(t)$. The multiplication of an 8×8 matrix A_i^{-1} by an eight-element vector gives the eight coefficients of $y_i(t)$ as follows.

$$\left[\begin{array}{ccc} a_{i,0} & \cdots & a_{i,4} \end{array}\right]^T = A_i^{-1} \left[\begin{array}{ccc} y_{i-1} & \cdots & y_i''' \end{array}\right]^T. \quad (3)$$

See [2, 3] for the numerical value of A_i and the existence theorem which guarantees the invertibility of A_i. Since the segment $y_i(t)$ is defined on $t \in [0, \pi/4]$ for all i, the time-scaled TS $y(t)$ is given by

$$y(t + (i-1)\pi/4) = y_i(t), \quad t \in [0, \pi/4], \quad (i = 1, \ldots, n). \quad (4)$$

The function $y(t)$ is a TS which satisfies the desired interpolation conditions, and which has length $n\pi/4$ seconds. The unscaled spline $\theta(t)$ given by

$$\theta(t) = y(n\pi t/4T) \qquad t \in [0, T] \quad (5)$$

stretches the trajectory from its normalized length $n\pi/4$ to a desired length T.

III. Optimisation

The user of the trajectory formulation algorithm described in the previous section is free to choose the first three trajectory derivatives at each knot. The knot derivatives can be chosen to minimise some objective function. Since $\vec{\theta}(t)$ is a time-scaled and time-shifted version of $\vec{y}(t)$ (5) where $\vec{y}(t)$ is the P-vector of normalised TSs, and $\vec{y}(t)$ is composed of the n spline segments $\vec{y}_i(t)$ (4), a general objective function can be written in the form

$$J = g[\sum_{k=1}^{n} \vec{y}_k(t)] \quad (6)$$

where $g(\cdot)$ is a general nonlinear function, $\vec{y}_k(t)$ is a P-vector of TS segments, and P is the number of joints that the robot has. The minimisation of this general objective function becomes a parameter optimisation problem, since $\vec{y}(t)$ is a function of the $3P(n-1)$ free knot derivatives, where $(n-1)$ is the number of interior knots of each joint trajectory.

If J is known to have only one minimum, then (6) is minimised when

$$\frac{\partial}{\partial y_{i,j}^{(r)}} g[\sum_{k=1}^{n} \vec{y}_k(t)] = 0, \quad (i = 1, \ldots, n-1), \quad (r = 1, 2, 3), \quad (j = 1, \ldots, P) \quad (7)$$

where $y_{i,j}^{(r)}$ is the r-th derivative of the i-th normalised spline segment of the j-th joint of the robot. Since $\vec{y}_k(t)$ is a function of $y_{i,j}^{(r)}$ only for $k \in \{i, i+1\}$ (1,3), we can write (7) as

$$\frac{\partial}{\partial y_{i,j}^{(r)}} g[\vec{y}_i(t) + \vec{y}_{i+1}(t)] = 0. \quad (8)$$

Further simplification from this point depends on the form of (6).

So the optimal control problem is simplified by reducing its dimension, thereby converting it to a parameter optimisation problem. This general approach to optimal control is similar to that taken by others [6]–[8]. But their formulations are applicable only for constraints at the initial and final time, and do not allow for constraints at specific times in between. In other words, as applied to the robot path planning problem, their approaches are valid only for path planning between an initial point and a final point, and do not allow for intermediate knots. Two specific examples of optimisation (minimum jerk and minimum energy) are considered in the following sections.

A. Minimum Jerk Trajectory

Suppose that the user desires to minimise the jerk of each joint throughout its trajectory. Kyriakopoulos and Saridis [9] report that the joint position errors of the path tracker increase with the magnitude of joint jerk. Also, Flanagan and Ostry [10] present evidence that the human brain plans arm movements so as to minimise a function of joint jerk. So minimizing some function of joint jerk would seem to be desirable, resulting in a coordinated motion which could be accurately followed by the robot path tracker. The objective function could then be written as

$$J = \int_0^T [\theta'''(t)]^2 dt = \int_0^{n\pi/4} [y'''(t)]^2 dt \quad (9)$$

where the second equality follows from (4) and (5). In order to minimise (9), we want to set each of the partial derivatives with respect to the $(n-1)$ normalised interior knot derivatives equal to zero. So (9) will be minimised when

$$\frac{\partial}{\partial y_i^{(r)}} \left\{ \int_0^{n\pi/4} [y'''(t)]^2 dt \right\} = 0$$
$$(i = 1, \ldots, n-1), \quad (r = 1, 2, 3). \quad (10)$$

Recall that $y(t)$ is composed of the functions $y_i(t)$ $(i = 1, \ldots, n)$, each of which is an analytic function of the eight parameters $(y_j^{(r)})$, $(j = i-1, i)$, $(r = 0, 1, 2, 3)$. So the differentiation and integration of (9) can be performed analytically to obtain the $(n-1)$ matrix equations

$$D_1 \begin{pmatrix} y_{i-1} \\ y_i \\ y_{i+1} \end{pmatrix} + D_2 Y_{i-1}^{(r)} + D_3 Y_i^{(r)} + D_4 Y_{i+1}^{(r)} = 0 \quad (11)$$

where we assume that the derivatives of the trajectory are constrained at the endpoints, the D_k are 3×3 matrices, and the $Y_i^{(r)}$ vectors are defined by

$$Y_i^{(r)} \equiv (\; y_i' \quad y_i'' \quad y_i''' \;)^T. \quad (12)$$

Then (11) can be written as the single matrix equation

$$\Delta \begin{pmatrix} Y_1^{(r)} \\ \vdots \\ Y_{n-1}^{(r)} \end{pmatrix} = C \quad (13)$$

where the block tridiagonal matrix Δ has constant blocks with known numerical entries on its diagonal, upper diagonal, and lower diagonal, and the vector C is a constant with known numerical entries [2, 3]. It can be shown that Δ is always nonsingular. This property follows from the fact that (9) is always greater than zero unless all of the knot derivatives are zero.

B. Minimum Energy Trajectory

Recall that the P-element torque vector of a P-joint robot can be given as

$$\mathcal{T}(\vec{\theta}) = M(\vec{\theta})\vec{\theta}\,'' + S(\vec{\theta},\vec{\theta}\,') \tag{14}$$

where M is the $P \times P$ mass matrix, and S is a vector of centrifugal, Coriolis, and gravity terms. Suppose the user wants to choose the interior knot derivatives of the TS for each joint so as to achieve a minimum energy trajectory. Then the objective function could be written as

$$J = \int_0^T \mathcal{T}^T(\vec{\theta})R\mathcal{T}(\vec{\theta})d\tau \tag{15}$$

where R is a $P \times P$ positive-definite weighting matrix. Using the fact (5) that $\vec{\theta}(\tau) = \vec{y}(n\pi\tau/4T)$, the torque vector $\mathcal{T}$ can be written as a function of the normalized joint trajectory $\vec{y}(t)$, and the objective function of (15) can be rewritten as

$$J = \frac{4T}{n\pi}\int_0^{n\pi/4} \mathcal{T}_y^T R \mathcal{T}_y dt \tag{16}$$

where $\mathcal{T}_y$ is given by

$$\mathcal{T}_y = M(\vec{y})(n\pi/4T)^2\vec{y}\,'' + S(\vec{y},(n\pi/4T)\vec{y}\,'). \tag{17}$$

Since $\vec{y}$ is formed by joining together the individual $\vec{y}_i$ components, each of which is defined only on the time interval $t \in [0,\pi/4]$ (4), we obtain

$$J = \frac{4T}{n\pi}\int_0^{\pi/4}\sum_{i=1}^{n}\mathcal{T}_{yi}^T R\mathcal{T}_{yi}dt \equiv \sum_{i=1}^{n} J_i \tag{18}$$

where $\mathcal{T}_{yi}$ is the normalised torque vector which is applied during the i-th spline segment. That is, $\mathcal{T}_{yi}$ is equal to (17) when $\vec{y}$ is replaced by $\vec{y}_i$.

So J is completely determined by the $3P(n-1)$ free parameters $\vec{y}_i^{(r)}$, $(r = 1,2,3)$, $(i = 1,\ldots,n-1)$. The optimal control problem of (15) has thus been converted into a parameter optimisation problem. Note that the objective function could also be minimised with respect to T (the actual path length) using some parameter optimisation scheme.

A significant computational savings in the solution of (18) can be realised by taking advantage of the fact that $\mathcal{T}_{yj}$ is an explicit function of $\vec{y}_i^{(r)}$ only for $j \in \{i, i+1\}$ (due to the decoupling of the spline segments). Therefore (18) can be solved as

$$\min_{\vec{y}_i^{(r)}}(J_i + J_{i+1}) \qquad (i = 1,\ldots,n-1) \tag{19}$$

where J_i is defined in (18). So the $3P(n-1)$-dimensional minimisation problem of (18) has been converted into $(n-1)$ separate minimisation problems, each of dimension $3P$. Of course, (19) is a highly nonlinear function of the parameters $\vec{y}_i^{(r)}$ and must be solved using some numerical method.

C. Optimization With Nonzero Knot Tolerances

The user may not really require the robot trajectory to pass exactly through the knots. The knots may be more like "centers of tolerance" *near* which the robot is required to pass. The TSs discussed earlier in this paper, and most algebraic splines, are planned so as to exactly pass through the given knots. The remainder of this section discusses the use of TSs when the robot is not required to pass exactly through the given knots. This additional freedom is used to improve the performance of the robot trajectory with respect to the objective functions discussed earlier in this section: minimum jerk and minimum energy.

C.1. Minimum Jerk Trajectory with Nonzero Knot Tolerances

As before, we desire to minimize the integral of the square of the jerk of the joint trajectory

$$\int_0^T [\theta'''(t)]^2 dt = \int_0^{n\pi/4}[y'''(t)]^2 dt = \sum_{i=1}^{n}\int_0^{\pi/4}[y_i'''(t)]^2 dt \tag{20}$$

where $y_i(t)$ is the i-th normalized TS segment. As described in Subsection A, $y_i'''(t)$ is a linear function of the eight parameters $y_j^{(r)}$, $(j = i-1, i)$, $(r = 0,1,2,3)$. Therefore

$$\int_0^{\pi/4}[y_i'''(t)]^2 dt = \frac{1}{2}x_i^T\hat{Q}x_i \tag{21}$$

where x_i^T is given by

$$x_i^T = (\ y_{i-1} \quad y_{i-1}' \cdots y_i'''\ ) \tag{22}$$

and $\hat{Q}$ is an 8×8 symmetric positive semidefinite matrix. Now we partition the vectors x_i and the matrix $\hat{Q}$ as

$$x_i = (\ \phi_{i-1}^T \quad \phi_i^T\ )^T \tag{23}$$

$$\hat{Q} = \begin{pmatrix} \hat{Q}_{11} & \hat{Q}_{12} \\ \hat{Q}_{12}^T & \hat{Q}_{22} \end{pmatrix} \tag{24}$$

where each submatrix in $\hat{Q}$ is a 4×4 matrix. Further matrix manipulations [4, 14] eventually yield

$$\min \int_0^T [\theta'''(t)]^2 dt = \min_x(\frac{1}{2}x^TQx - b^Tx) \tag{25}$$

where the $4(n-1) \times 4(n-1)$ block tridiagonal matrix Q is given by

$$Q = \begin{pmatrix} \hat{Q}_{11}+\hat{Q}_{22} & \hat{Q}_{12} & \cdots & 0 \\ \hat{Q}_{12}^T & \hat{Q}_{11}+\hat{Q}_{22} & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & \hat{Q}_{11}+\hat{Q}_{22} & \hat{Q}_{12} \\ 0 & \cdots & \hat{Q}_{12}^T & \hat{Q}_{11}+\hat{Q}_{22} \end{pmatrix} \tag{26}$$

and the $4(n-1)$-vectors b and x are given by

$$x^T = (\ \phi_1^T \ \ \ldots \ \ \phi_{n-1}^T\ ). \tag{27}$$

$$b^T = -(\ \phi_0^T\hat{Q}_{12} \ \ 0 \ \ \ldots \ \ 0 \ \ \phi_n^T\hat{Q}_{12}^T\ ) \tag{28}$$

The $\hat{Q}$ matrices of (24) and following are given numerically in [4].

Now the user may not require the TS to pass exactly through the given interior knots. The user may rather specify a desired tolerance for each knot. This increases the domain of the optimization problem, and thus results in a lower objective function value and a larger computational effort. The knot tolerances result in the following inequality constraints being associated with the minimization problem of (20).

$$|y_i - y_{ci}| \le y_{toli} \qquad (i = 1,\ldots,n-1) \tag{29}$$

where y_i is the angle of the TS at knot i, y_{ci} is the desired knot angle (the center of tolerance), and y_{toli} is the allowable joint angle error at knot i. These constraints can be written as

$$Ax \le c \tag{30}$$

where A is the $2(n-1) \times 4(n-1)$ matrix

$$A = \mathrm{diag}(\ A_s \ \cdots \ A_s\ ) \tag{31}$$

$$\text{and } A_s = \begin{pmatrix} -1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}. \tag{32}$$

The vector x is the $4(n-1)$-vector given in (27), and c is the $2(n-1)$-vector given by

$$c^T = [\ (y_{tol1} - y_{c1}) \quad (y_{tol1} + y_{c1}) \quad (y_{tol2} - y_{c2}) \quad \cdots \quad (y_{tol(n-1)} + y_{c(n-1)})]. \tag{33}$$

So by combining (25) and (30), the minimum jerk problem with nonzero knot tolerances can be written as

$$\min_x(\frac{1}{2}x^TQx - b^Tx) \quad \text{subject to } Ax \le c. \tag{34}$$

Matrix Q can be shown to be positive definite by similar reasoning as used for the matrix in (13).

So the problem has been reduced to a quadratic programming problem with linear constraints. This type of problem can be solved by several different methods, among which is Hildreth's Algorithm [12, 13]. Note that trajectory derivative inequality constraints at the knots can easily be incorporated into this problem by a straightforward modification of matrix A in (30).

C.2. Minimum Energy Trajectory with Nonzero Knot Tolerances

Now suppose we desire to find a minimum energy TS through a given sequence of knots, but with a specified allowable knot tolerance (29) given by

$$|\vec{y}_i - \vec{y}_{ci}| \le \vec{y}_{toli} \qquad (i = 1, \ldots, n-1). \tag{35}$$

Each vector in (35) has P elements, with P being the number of joints of the robot. The vector inequality in (35) is taken component by component.

In principle, the energy objective function could be augmented with a penalty function [13] corresponding to (35). Unfortunately, when the objective function is highly nonlinear, the penalty function approach may result in an augmented objective function with many hills and valleys. So a local minimum of might be significantly larger than other nearby minima. This indeed turns out to be the case for the minimum energy TS problem with nonzero knot tolerances. So rather than using a penalty function method, the following method, which makes use of the physical significance of the constraints of (35), is proposed.

$$\min_{\vec{y}_i}\left[\min_{\vec{y}_i^{(r)}}\int_0^{\pi/4}(T_i^T R T_i + T_{i+1}^T R T_{i+1})d\tau\right], \qquad (i = 1, \ldots, n-1), \qquad (r = 1, 2, 3). \tag{36}$$

This method is a set of $(n-1)$ minimizations (the outer minimization) over P-dimensional domains ($\vec{y}_i$). The function which each of these $(n-1)$ minimizations minimizes is itself the solution to a minimization problem (the inner minimization) over $3P$-dimensional domains ($\vec{y}_i^{(r)}$).

The above algorithm recognizes the increased number of hills and valleys in the objective function due to the increased number of free parameters (i.e. the knot angles). The algorithm also recognizes that the optimum knot derivatives $\vec{y}_i^{(r)}$ are functions of the knot angles $\vec{y}_i$.

IV. Simulation Results

In this section, simulation results will be presented to support the work done in the previous section. The robot manipulator which is considered is a two-link robot which is described in Craig [1, Section 6.7]. The robot operates in the vertical plane, and each link's mass ($m_1 = 4.6$ kg and $m_2 = 2.3$ kg) is concentrated at its distal end. Each link is 0.5 meters in length. The torque (in Newton-meters) due to viscous friction for each joint is assumed to be five times the joint velocity (in radians/second).

Seven Cartesian knots are specified. The TS is required to pass through (or near in the case of nonzero knot tolerances) these seven knots. The seven knots are given in Table I, along with the corresponding joint angles at the knots (obtained by inverse kinematics). There is currently no other literature which discusses optimum robot path planning through a given set of knots. So these knots were chosen somewhat arbitrarily to represent what might be a typical task for an industrial robot. The length of the path was fixed at 30 seconds. In this section five different types of TSs are computed: nominal splines (no optimization), minimum jerk splines, minimum energy splines, minimum jerk splines with nonzero (four degree) knot tolerances, and minimum energy splines with nonzero (four degree) knot tolerances.

Knot Number	Cartesian Knots (meters) x	y	Joint Angles 1	2
1	$\sqrt{2}/2$	0	45	-90
2	$\sqrt{2}/2$	$\sqrt{2}/4$	64	-76
3	$\sqrt{2}/4$	$\sqrt{2}/2$	101	-76
4	0	1	90	0
5	$-\sqrt{2}/4$	$\sqrt{2}/2$	79	76
6	$-\sqrt{2}/2$	$\sqrt{2}/4$	116	76
7	$-\sqrt{2}/2$	0	135	90

Table I: Seven Cartesian and Joint Space Knots

Hildreth's algorithm was used for the minimum jerk trajectory with nonzero knot tolerances. Since Hildreth's algorithm is iterative in nature, it could theoretically take an infinite number of iterations before convergence is achieved. Therefore, some error y_e in the solution is allowed. Once Hildreth's algorithm achieves a solution with knot errors within $\pm(y_{toli} + y_e)$, the algorithm is considered to have converged. The allowable error y_e was chosen to be 0.1 degrees. So the actual allowable knot tolerances were 4.1 degrees, while the parameters y_{toli} were fixed at four degrees.

For the minimum energy trajectories, Powell's method of nonlinear parameter optimization was implemented on a DEC Vax 8820. Powell's method was used to perform $(n-1)$ separate $3P$-dimensional minimization problems, as indicated in (19). The number of knots is $(n+1)$, and the number of joints is P (two for the manipulator considered in this section). The weighting matrix R of (15) was taken to be the identity matrix. The algorithm was considered to have converged when the objective function decreased by less than 0.5 percent. The additional minimization with respect to the knot angles (for the case of minimum energy with nonzero knot tolerances) was considered to have converged when the knot angle under consideration changed by less than 0.5 degrees.

The resulting trajectories are shown in [4, 14]. A compari-

		Type of Trigonometric Spline			
		Minimum Jerk		Minimum Energy	
	Nominal	Zero Knot Tolerance	Nonzero Knot Tolerance	Zero Knot Tolerance	Nonzero Knot Tolerance
Joint 1 Jerk	1.453	0.2229	0.1741	1.608	2.355
Joint 2 Jerk	2.351	0.8190	0.4028	2.342	1.278
Energy	27341	3674	3021	557	507
Computational Effort	760 flops	767 flops	2.7 sec	55 sec	390 sec

Table II: Objective Functions and VAX 8820 Computational Effort for Trigonometric Splines

son of the various objective functions is given in Table II. The numbers in Table II are radians^2/second^5 for the jerk objective function, and (Newton-meter)^2·seconds for the energy objective function. Note from Table II the sizeable improvement in the energy objective function when optimization is used. Even the minimum jerk splines improve the energy consumption by a factor of five or six when compared to the nominal splines. This indicates that the minimization of jerk is a big step towards the minimization of energy. In contrast, the use of minimum energy splines does not result in any improvement of the jerk objective function when compared to the nominal splines.

Table II shows that the introduction of nonzero knot tolerances results in a decrease of the objective function under consideration. This is as expected. The optimization algorithm is given more free parameters, and this results in better performance.

Table II also shows the computational effort which was required for each spline. The nominal spline and minimum jerk splines have closed-form solutions, and so their computation effort can be measured in *flops* (floating point operations). The other splines in Table II require iterative solutions, and so their computational effort is measured in CPU time on a Vax 8820 computer. It has been shown that the computational effort increases linearly with the number of knots [5].

V. Conclusion

It has been shown that the use of trigonometric splines for robot path planning is amenable to path optimization subject to user-specified knot tolerances. The knots may be chosen to avoid obstacles. So the robot path may not need to path exactly through the knots, but rather *near* the knots. This possibility makes optimization subject to user specified knot tolerances a desirable feature of a path planning method. The objective function under consideration can decrease significantly if the knot tolerances are used wisely. The optimization procedures presented in this paper are iterative, and thus cannot be performed in real time. But if the objective function is minimum jerk subject to knot tolerances, then the optimization problem reduces to a quadratic programming problem with linear constraints. This is a well known problem, and there are several ways of solving it.

If the objective function includes the dynamics of the robot, then the optimization problem must be solved using an iterative method due to the nonlinearity of robot dynamics. But the decoupled nature of TS segments means that the optimization problem can be split into many smaller subproblems (one for each knot). This decreases the computational effort by a significant amount. The simulation results of this paper indicate that the minimization of an energy objective function can result in a decrease of energy by a factor of 25 or more. This would result in less wear and tear on the robot, and lower power requirements. Both of these results would be attractive to robot users.

References

[1] J. Craig, *Introduction to Robotics.* Reading, MA: Addison-Wesley, 1989.

[2] D. Simon and C. Isik, "Optimal Trigonometric Robot Joint Trajectories," *Robotica*, 9:379-386, Oct. 1991.

[3] D. Simon and C. Isik, "A Trigonometric Trajectory Generator for Robotic Arms," *International Journal of Control*, in print.

[4] D. Simon, "A Unified Approach to Robot Path Planning Using Trigonometric Splines," Ph.D. Dissertation, Syracuse University, Department of Electrical Engineering, Syracuse, NY, Aug. 1991.

[5] D. Simon and C. Isik, "Computational Complexity and Path Error Analyses of Trigonometric Joint Trajectories," *International Journal of Robotics and Automation*, in print.

[6] C. Neuman and A. Sen, "A Suboptimal Control Algorithm for Constrained Problems Using Cubic Splines," *Automatica*, 9:601-613, 1973.

[7] J. Vlassenbroeck and R. Van Dooren, "A Chebyshev Technique for Solving Nonlinear Optimal Control Problems," *IEEE Transactions on Automatic Control*, AC-33:333-340, April 1988.

[8] V. Yen and M. Nagurka, "Fourier-Based Optimal Control Approach for Structural Systems," *Journal of Guidance, Control, and Dynamics*, 13:265-276, March 1990.

[9] K. Kyriakopoulos and G. Saridis, "Minimum Jerk Path Generation," *IEEE International Conference on Robotics and Automation*, 1:364-369, 1988.

[10] J. Flanagan and D. Ostry, "Trajectories of Human Multi-Joint Arm Movements: Evidence of Joint Level Planning," in *Experimental Robotics I, The First International Symposium*, ed. by V. Hayward and O. Khatib, New York: Springer-Verlag, 1990.

[11] W. Press, B. Flannery, S. Teukolsky, and W. Vetterling, "Numerical Recipes," New York: Cambridge University Press, 1986.

[12] C. Hildreth, "A Quadratic Programming Procedure," *Naval Research Logistics Quarterly*, 4:79-85, 1957.

[13] D. Luenberger, "Optimization by Vector Space Methods," New York: Wiley, 1968.

[14] D. Simon and C. Isik, "Suboptimal Robot Joint Interpolation Within User-Specified Knot Tolerances," *Journal of Robotic Systems*, submitted for review.