A Path Space Approach to Nonholonomic Motion Planning in the Presence of Obstacles

Adam W. Divelbiss and John T. Wen

***Abstract*—This paper presents an algorithm for finding a kinematically feasible path for a nonholonomic system in the presence of obstacles. We first consider the path planning problem without obstacles by transforming it into a nonlinear least squares problem in an augmented space which is then iteratively solved. Obstacle avoidance is included as inequality constraints. Exterior penalty functions are used to convert the inequality constraints into equality constraints. Then the same nonlinear least squares approach is applied. We demonstrate the efficacy of the approach by solving some challenging problems, including a tractor-trailer and a tractor with a steerable trailer backing in a loading dock. These examples demonstrate the performance of the algorithm in the presence of obstacles and steering and jackknife angle constraints.**

***Index Terms*—Mobile robots, nonholonomic motion planning, obstacle avoidance, path planning, tractor-trailer.**

I. Introduction

The nonholonomic motion planning (NMP) problem involves finding a path which links some initial configuration to some desired final configuration for a system with nonholonomic velocity constraints. Nonholonomic constraints occur in physical systems for which constraints on the generalized velocity vector are nonintegrable to equivalent configuration space constraints. As a result of this condition, the instantaneous velocity of a system under nonholonomic constraints is limited to certain directions. Constraints of this kind occur for any wheeled vehicle under the no-slip condition, for any two bodies in rolling contact, and also for systems in free fall where the conservation of angular momentum applies. Examples of these types of nonholonomic systems include mobile robots, automobiles, tractor-trailer vehicles, finger contacts in robot hand grasping, orbiting satellites, space-based robot manipulators and even falling cats. Nonholonomic systems, such as these, possess some peculiar characteristics which make motion planning for them challenging. For instance, nonholonomic systems are frequently globally controllable and yet the linearized system about an equilibrium point is not controllable. Further, there exists no continuous time-invariant stabilizing feedback for driftless nonholonomic systems [1], [2]. Due to these difficulties, standard motion planning techniques for holonomic systems are not directly applicable to nonholonomic motion planning.

In the rapidly accumulating literature on nonholonomic motion planning, there are mainly two classes of approaches. The search based methods, including those found in [3]–[9], generally involve some type of decomposition of the configuration space into cells. A graph is then constructed whose nodes are configurations and whose arcs are some type of path (shortest, optimal, etc.) connecting two configurations. The graph is searched using an A^* or a similar algorithm. One of the best characteristics of these approaches is that they can plan paths in highly cluttered or constrained environments. Many search based methods are also global in nature, meaning that they search over the entire configuration space. However, these methods can be computationally inefficient for complicated vehicles such as a tractor with two or more trailers. Other methods involve neural-networks and fuzzy controllers as in [10] but they require extensive training and tuning. Control theoretic methods such as [11]–[18], apply the tools in geometric control theory and provide elegant insights into the structure of the solution. A popular approach among these methods converts the system into some canonical form and then uses cyclic motion in certain base space variables to cause a net motion of the remaining variables [19]. Inequality constraints are rarely considered, and physically impractical paths frequently result.

In this paper, we present a new approach to NMP. An initial path is iteratively warped until all constraints are satisfied. Thus it is a local *path space* method. The local nature comes from the fact that convergence depends on the closeness of the initial path to a feasible solution: one that satisfying all the constraints. The initial path is itself not required to be feasible. Both equality and inequality constraints can be included in this formulation. Task space constraints involving nonlinear functions of the configuration variables are also allowed. In contrast to other path space approaches, this approach emphasizes feasibility over optimality. Once a feasible solution is found, optimality can be incorporated as a secondary objective.

A common starting point for the control theoretic approaches is to pose the NMP problem as a general nonlinear control problem. Our formulation is based on converting the nonlinear control problem into a nonlinear algebraic equation. The problem then becomes a nonlinear root finding problem in which the dimension of the search space is very high (infinite if $\{u(t) : t \in [0, 1]\}$ is taken from an infinite dimensional functional space) compared to the number of equality constraints (for the end point constraint). The nonlinear root finding problem can be solved by many familiar numerical algorithms including the Newton-Raphson method, the steepest descent method, the conjugate gradient method, etc. A similar approach has also been proposed independently in [20]–[22] for kinematic path planning with only the end point constraint. In all of these algorithms, convergence relies on a certain path gradient operator being full rank. This full rank condition has been shown to be true generically in a certain sense [22]. Coupled with a clever generic loop argument, convergence of this algorithm class is virtually assured. Along this same line, a conservative strong bracket generating condition is shown to be sufficient for the wellposedness of the initial value problem arising from the Newton-Raphson algorithm [20], [23].

There are typically many possible solutions to the root finding problem resulting from the path planning problem. Since many of these solutions will result in physically unrealizable paths, additional constraints must be enforced on the path for the algorithm to be useful in any practical situation. To enforce these constrains, we have adopted an approach similar to the exterior penalty function method [24], [25] which converts inequality constraints into a zero finding problem by solving a sequence of unconstrained minimizations. In contrast to the standard application of this method, there is no need to successively increase the penalty weights since the minimum values

Manuscript received January 22, 1996. This work was supported in part by the National Science Foundation under Grant IRI-9408874 and by the New York State Center for Advanced Technology (CAT) in Automation, Robotics and Manufacturing at Rensselaer Polytechnic Institute. The CAT is partially funded by a block grant from the New York State Science and Technology Foundation. This paper was recommended for publication by Associate Editor A. De Luca and Editor S. Salcudean upon evaluation of the reviewers' comments.

A. W. Divelbiss is with the Reveo Corporation, Hawthorne, NY 10532 USA.

J. T. Wen is with the New York State Center for Advanced Technology in Automation, Robotics and Manufacturing, Department of Electrical, Computer, and Systems Engineering, Rensselaer Polytechnic Institute, Troy, NY 12180 USA.

Publisher Item Identifier S 1042-296X(97)03823-8.

(i.e., zeros) of the penalty functions are by definition solutions to the problem. Our method also differs from the familiar artificial potential field method (popular in the holonomic path planning literature [26]–[28]) which is an interior penalty function (or barrier function) method in that the initial guess may be infeasible. The exterior penalty functions associated with the inequality constraints are combined with the equality end point constraint to form an augmented zero finding problem. Nonconfiguration space constraints including constraints on the corners of a vehicle, body of the robot, etc., can also be incorporated in this formulation. For certain convex polyhedral constraints, we show that the exterior penalty function does not add extra path singularities. In general, however, there may be local minima in the path space.

The remainder of this paper is organized as follows. In Section II, some of the theory behind the basic algorithm for enforcing end point equality constraints is presented. In Section III, the algorithm is extended to incorporate inequality constraints on the path by using the exterior penalty function method. In Section IV, simulations involving tractor-trailers are presented.

II. NMP Subject to Equality Constraints

Given a mechanical system with nonholonomic equality constraints, an initial configuration and a desired final configuration, we are interested in finding a path which 1) satisfies the nonholonomic equality constraint and 2) satisfies the inequality constraints associated with internal limits of the system, external limits imposed by the environment, and limits imposed on the control. We assume that the system is globally controllable, the system kinematic equation is as smooth as required, and that obstacles and boundaries in the environment are static. In this section, we first consider the path planning problem with only the equality constraints.

We shall need the following definitions. A path connecting some initial configuration, denoted by x_0, to some final configuration, denoted by x_f, is defined as a sequence of consecutive system configurations evenly spaced over a specified normalized time interval and is denoted by $\underline{x} = \{x(t_j) \in \mathbf{R}^n : t_j \in [0,1],\ j = 0, 1, \cdots, N\}$ where n is the state dimension, $(t_j - t_{j-1}) = h = 1/(N-1)$, $x(t_0) = x_0$, and $x(t_N) = x_f$. We denote a specific path point, $x(t_j) \in \underline{x}$, by x_j. The ith element of x_j is denoted by x_j^i. The continuous form of the path is denoted by $x = \{x(t) \in \mathbf{R}^n : t \in [0,1]\}$. The control space is defined as $\mathcal{U} \triangleq L_2^m[0,1]$. The path space, $\mathcal{P}$, is the collection of all paths generated by all possible controls.

We shall consider the problem of imposing an an equality constraint on the final point of the path (i.e., $x_N = x_f$) while ensuring the nonholonomic equality constraint is satisfied at each point along the path. General nonholonomic velocity equality constraints are often represented by

$$\mathcal{G}(x, \dot{x}, t) = 0 \tag{1}$$

where x is the generalized coordinate vector and $\dot{x}$ is the generalized velocity vector. The constraints represented by (1) are said to be *nonholonomic* if the equation is nonintegrable, i.e., there exists no function of the form $F(x,t) = 0$ such that $\mathcal{G}$ is the differential of F. There are also nonholonomic constraints involving accelerations [29], called second order nonholonomic constraints, which we do not consider here, but our method can in principle be applied to such systems as well.

In most physical cases (e.g., the no-slip rolling constraint), (1) is linear in $\dot{x}$ and time-invariant, and may be rewritten as

$$\mathcal{G}(x)\dot{x} = 0 \tag{2}$$

where $\mathcal{G}$ is a fat matrix (has more columns than rows).

A. The Basic Algorithm

The nonholonomic constraint (2) can be written in the form

$$\dot{x} = f(x)u; \quad x(0) = x_0 \tag{3}$$

where $x(t) \in \mathbf{R}^n$ is the configuration variable, $u(t) \in \mathbf{R}^m$ is the admissible velocity input, and the columns of $f(x)$ span the null space of $\mathcal{G}(x)$, i.e., $\mathcal{G}(x)f(x) = 0_{m\times m}$. Using this form, the path planning problem can be posed as a general nonlinear control problem: find an input, u, to drive the system from the initial configuration x_0 to the desired final configuration x_f. Even though the problem can be stated in such a deceptively simple form, its solution is very challenging due in part to the fact tha Furthermore, there exists no time-invariant stabilizing feedback [1], [2].

Our approach is based on converting (3) to a nonlinear algebraic equation

$$x(1) = \hat{F}(x_0, u) \quad \hat{F} : \mathbf{R}^n \times L_2^m[0,1] \to \mathbf{R}^n \tag{4}$$

where x_0 is the initial configuration, $u \triangleq \{u(t) : t \in [0,1]\}$, and where the final time has been normalized to 1. The analytic form of $\hat{F}$ is in general difficult to find but will not be explicitly required. In this perspective, (3) is globally controllable if and only if the nonlinear mapping $\hat{F}(x_0, .) : L_2^m[0,1] \to \mathbf{R}^n$ is onto for all $x_0 \in \mathbf{R}^n$. The system is locally controllable around u if and only if the Fréchet derivative of $\hat{F}$ with respect to u, $\nabla_u \hat{F}(u) : L_2^m[0,1] \to \mathbf{R}^n$, is a linear onto map. Since the mapping, $\nabla_u \hat{F}(x_0, u)$ relates variations in $x(1)$ to variations in u, it may be thought of as the operator corresponding to the solution of the variational time-varying linear system

$$\dot{\delta}x = A(t)\delta x + B(t)\delta u, \quad \delta x(0) = 0 \tag{5}$$

where $A(t) \triangleq [\frac{\partial f}{\partial x^1}(x(t))u(t) \quad \cdots \quad \frac{\partial f}{\partial x^n}(x(t))u(t)]$ and $B(t) \triangleq f(x(t))$. Since $\delta x(0) = 0$, the solution to this equation is written as

$$\delta x(1) = \nabla_u \hat{F}(x_0, u)\delta u \triangleq \int_0^1 \Phi(1,s)B(s)\delta u(s)\, ds \tag{6}$$

where Φ is the state transition matrix of the linearized system and x_0 is the given initial configuration. Controllability of the system in (5) is equivalent to $\nabla_u \hat{F}(x_0, u)$ being onto, or, $\operatorname{rank}(\nabla_u \hat{F}(x_0, u)) = n$.

We now pose the path planning problem as a nonlinear root finding problem as follows. Given an initial configuration x_0, find a control $u \in L_2^m[0,1]$ such that

$$y(x_0, u) \triangleq \hat{F}(x_0, u) - x_f = 0. \tag{7}$$

Since the dimension of the search space is much higher than the dimension of the constraint space, there are typically a large number of solutions, many of which are physically unrealizable. Additional constraints must be imposed to obtain implementable paths.

Many methods are available for solving the nonlinear root finding problem for a fixed initial configuration x_0, such as Newton-Raphson, gradient descent, conjugate-gradient, Levenberg-Marquardt, etc. We have used the Newton-Raphson algorithm successfully. In our implementation, the control u is iteratively updated by

$$u^{(k+1)} = u^{(k)} - \alpha_k \left[\nabla_u \hat{F}\left(x_0, u^{(k)}\right)\right]^+ y\left(x_0, u^{(k)}\right) \tag{8}$$

where α_k is found through a line search to maximize the decrease in y. A sufficient condition for the algorithm to converge is that the gradient operator $\nabla_u \hat{F}(x_0, u)$ be full rank for each iteration. This condition has been shown to be generic in some sense in [22] and an explicit characterization of all the singular controls (i.e., u for which $\nabla_u \hat{F}(x_0, u)$ loses rank) has been found for the N-trailer system [30].

B. Fourier Basis Representation

For implementation, the control function u in (8) needs to be approximated by a finite dimensional vector. We have used the Fourier series expansion

$$u(t) = \sum_{i=1}^{\infty} \phi_i(t)\lambda_i \quad \forall\, t \in [0,1] \tag{9}$$

where the $\phi_i(t)$'s are the standard Fourier basis elements. Substituting into (4), we get

$$x(1) = F(x_0, \lambda) = \hat{F}(x_0, T^{-1}\lambda). \tag{10}$$

where we have written $u = T^{-1}\lambda$. If the Newton-Raphson algorithm is used, the update law for λ is

$$\lambda^{(k+1)} = \lambda^{(k)} - \alpha_k \left[\nabla_\lambda F\left(x_0, \lambda^{(k)}\right)\right]^+ F\left(x_0, \lambda^{(k)}\right) \tag{11}$$

where $\nabla_\lambda F(x_0, \lambda) = \nabla_u \hat{F}(T^{-1}\lambda)T^{-1}$.

III. NMP Subject to Inequality Constraints

Many of the possible solutions to (11) will result in physically unrealizable motions. In order for the algorithm to be of any practical use, it must be modified to enforce inequality constraints, such as joint limits or task space constraints, on intermediate points along the path.

Suppose that a feasible region in the path space $\mathcal{P}$ is defined by a set of p inequalities

$$c(\underline{x}) \le 0 \tag{12}$$

where $\le$ is interpreted as a component-wise relationship. This condition requires that each point along the path to lie in the feasible region.

In the previous section, the mapping F was defined to relate the Fourier coefficients of the control, λ, to the final configuration. In the same manner, functions relating each point in $\underline{x}$ to λ are defined as $x(t_j) = \hat{F}_j(T^{-1}\lambda) = F_j(\lambda)$, for each $j = 1, 2, \cdots, N$, where the initial configuration, x_0, is assumed given. Stacking each of these functions into a single vector yields a new function $\mathcal{F}$ where $\underline{x} = \mathcal{F}(\lambda)$. Substituting into (12) defines a feasible region in the control coefficient space

$$c(\mathcal{F}(\lambda)) \le 0. \tag{13}$$

To enforce the p inequality constraints represented by (13) the inequality constraints are converted into equality constraints by defining an exterior penalty function corresponding to the ith constraint as

$$z_i(\lambda) = \gamma_i \sum_{j=1}^{N} g(c_i(F_j(\lambda))) \tag{14}$$

where $\gamma_i > 0$, c_i is the ith constraint, and g is a continuous scalar function with the property that g is positive and strictly monotonically increasing for $c > 0$ and zero for $c \le 0$. We have used g of the following form:

$$g(c) = \begin{cases} (1 - e^{-rc})^2 & \text{if } c > 0 \\ 0 & \text{if } c \le 0 \end{cases} \qquad r > 0. \tag{15}$$

Each penalty function z_i forces λ toward the feasible region when the constraint is violated and has no effect when the constraint is satisfied. The composite penalty is written as $z = [z_1, \cdots, z_p]^T$. Note that only the active constraints (i.e., $z_i > 0$) need to be included z.

The motion planning problem can now be restated as follows. Given an initial configuration, x_0, find λ such that $y(\lambda) = 0$ and $z(\lambda) = 0$. The same Newton-Raphson approach can be applied

$$\lambda^{(k+1)} = \lambda^{(k)} - \alpha_k [G(\lambda^{(k)})]^+ \psi(\lambda^{(k)}) \tag{16}$$

where

$$\psi(\lambda) = \begin{bmatrix} y(\lambda) \\ z(\lambda) \end{bmatrix}, \quad G(\lambda) \triangleq \begin{bmatrix} \nabla_\lambda F(\lambda) \\ \nabla_\lambda z(\lambda) \end{bmatrix} = \nabla_\lambda \psi(\lambda). \tag{17}$$

Again, a sufficient condition for convergence is that G be full rank for each iteration. To consider this issue further, we first decompose G as $G(\lambda) = \mathcal{K}(\lambda)\mathcal{D}(\lambda)$ where $\mathcal{K}(\lambda)$ is the gradient of the cost vector ψ with respect to the path $\underline{x}$ (i.e., $\mathcal{K}(\lambda) \triangleq \nabla_{\underline{x}}\psi(\lambda)$) and $\mathcal{D}(\lambda)$ is the gradient of the path $\underline{x} = \mathcal{F}(\lambda)$ with respect to the Fourier coefficient vector λ (i.e., $\mathcal{D}(\lambda) \triangleq \nabla_\lambda \mathcal{F}(\lambda)$). By taking the gradient of (14) and putting it into matrix form, $\mathcal{K}$ and $\mathcal{D}$ can be found as

$$\begin{aligned} \mathcal{K}(\lambda) &\triangleq \begin{bmatrix} 0_{n,n} & \cdots & 0_{n,n} & I_n \\ \gamma_1 \bar{k}_{11}(\lambda) & \cdots & \gamma_1 \bar{k}_{1N-1}(\lambda) & \gamma_1 \bar{k}_{1N}(\lambda) \\ & \vdots & & \\ \gamma_p \bar{k}_{p1}(\lambda) & \cdots & \gamma_p \bar{k}_{pN-1}(\lambda) & \gamma_p \bar{k}_{pN}(\lambda) \end{bmatrix} \\ \mathcal{D}(\lambda) &\triangleq \begin{bmatrix} D_1(\lambda) \\ \vdots \\ D_{N-1}(\lambda) \\ D_N(\lambda) \end{bmatrix}. \end{aligned} \tag{18}$$

where

$$\begin{aligned} \bar{k}_{ij}(\lambda) &\triangleq g'(c_i(\mathcal{F}_j(\lambda)))[\nabla_x c_i(\mathcal{F}_j(\lambda))]^T \\ D_j(\lambda) &\triangleq \nabla_\lambda \mathcal{F}_j(\lambda). \end{aligned} \tag{19}$$

The augmented gradient G is full rank if 1) $\mathcal{D}$ is of full row rank and 2) $\mathcal{K}^T(\lambda)\psi(\lambda) = 0$ if and only if $\psi(\lambda) = 0$.

First consider $\mathcal{D}$. A necessary condition for $\mathcal{D}$ to be of full row rank is that it be fat. This condition means that there must be a sufficiently number of Fourier basis elements M, i.e., $M \ge N \cdot n$. We further note that the system is causal (future input does not affect past state), therefore, $\mathcal{D}(\lambda)T$ (where T maps u to λ) is block lower triangular. Hence, $\mathcal{D}(\lambda)T$ is of full row rank if and only if each of its diagonal blocks (corresponding to each constrained path point) is of full row rank. This condition is equivalent to the time-varying linearized system between any two consecutive path points being controllable. The same genericity result and characterization of singular control for the full path (as mentioned in the previous section) can also be applied to these path segments.

Now consider $\mathcal{K}^T\psi = 0$. From (14) and (19), the jth equality, $1 \le j \le N-1$ in this equation can be written as

$$0 = \sum_{i=1}^{p'} \gamma_i \bar{k}_{ij}^T z_i = \sum_{i=1}^{p'} \gamma_i g'_{ij} \left[\nabla_x^T c_{ij}\right] z_i \tag{20}$$

where g'_{ij} and c_{ij} denote $g'(c_i(x_j))$ and $c_i(x_j)$, respectively, and p' constraints are assumed to be active. This can be written in a more compact form

$$(\nabla_x c(x_j))^T w(x_j) = 0 \tag{21}$$

where the ith element of $w(x_j)$ is $w_i(x_j) = \gamma_i g'_{ij} z_i$. If the worst case simultaneous active constraints satisfy $\mathcal{N}(\nabla_x c^T) = \{0\}$ ($\mathcal{N}$ denotes the null space), then $w(x_j) = 0, j = 1, \cdots, N-1$, which in turn implies $z_i = 0$. A necessary condition for $\mathcal{N}(\nabla_x c)^T = \{0\}$ is that $n \ge p'$. For example, suppose that each state is bounded above and below. Even though there are $2 \cdot n$ constraints ($p = 2 \cdot n$), there can be at most n simultaneous constraints.

The Nth equality in $\mathcal{K}^T\psi = 0$ can be written in a similar fashion

$$(x_N - x_f) + \sum_{i=1}^{p'} \gamma_i g'_{iN} \nabla_x^T c_{iN} z_i = 0. \tag{22}$$

If $z_i = 0$ for $i = 1, \cdots, N$, then $x_N = x_f$, and $\psi = 0$ as required. The only remaining case is that only the path end point constraint is

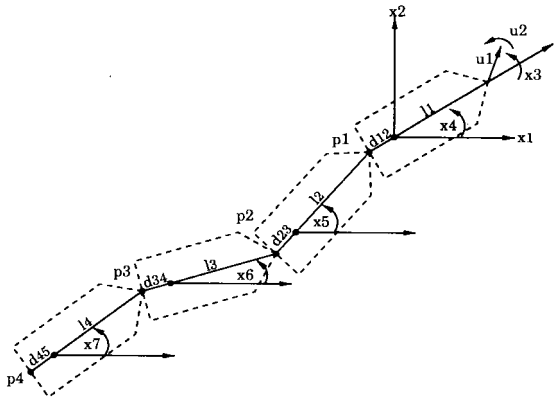

Fig. 1. Triple tractor-trailer model.

active. In this case, $\mathcal{K}^T\psi = 0$ reduces to (22). This condition may be rare, but we can only conclusively eliminate it in certain polyhedral constraint cases. Consider the case where

$$c(x) = Ax + b \leq 0. \tag{23}$$

Equation (22) then becomes

$$(x_N - x_f) + A^T w(x_N) = 0 \tag{24}$$

where A only needs to include the worst case simultaneous constraints. Suppose A is of full row rank (implying that the maximum number of simultaneous active constraints is less than the number of states), then

$$Ax_f + b = Ax_N + b + AA^T w(x_N). \tag{25}$$

Suppose x_N violates only the ith constraint in (23) (this can be easily generalized to x_N violating multiple constraints), i.e., $e_i^T(Ax_N+b) > 0$, $w_i(x_N) > 0$, and $w(x_N) = e_i w_i(x_N)$ where e_i is the ith unit vector. Then

$$\begin{aligned} e_i^T Ax_f + b &= e_i^T(Ax_N + b) + e_i^T AA^T w(x_N) \\ &> 0 + e_i^T AA^T e_i w_i(x_N) \\ &> 0. \end{aligned}$$

This condition implies that x_f violates the constraints which contradicts the assumption. Therefore, $\mathcal{K}^T\psi = 0$ implies that $\psi = 0$.

Inequality constraints may not always be directly expressed in terms of the state variable. For example, collision avoidance may be required for the boundary points of a vehicle. These constraints can be converted to inequalities in the path space as in (12) through the vehicle kinematics.

In many situations, analytic representation of the constraint may be tedious or not possible. The penalty function can be directly constructed based on a contour map from the task space obstacle boundaries. The contour map assigns a cost and a gradient direction for each point in a two-dimensional grid. Linear interpolation of the four closest map grid points is used to compute the cost and gradient for any point in the map, which are in turn used in the computation of the exterior penalty function and its gradient.

IV. Models and Simulations

In this section, we apply the path planning algorithm described in the previous sections to several different types of articulated vehicles to demonstrate its versatility and usefulness in planning paths under a wide variety of constraints. To begin, the kinematic models for a general tractor-trailer vehicle with n trailers and a general tractor-trailer vehicle with n steerable trailers are presented. Derivations for the models are found in the [31].

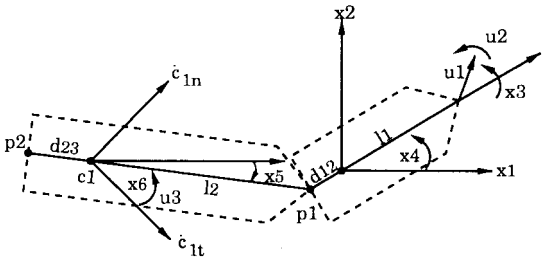

Fig. 2. Steerable tractor-trailer model.

A. Kinematic Models

There have been several multibody vehicle models proposed in [32] and [33]. These models are derived with the trailer pivot points located at the center of the rear axle of the preceding body. This situation is almost never the case for real tractor-trailer systems. For instance, a car with a boat trailer has the trailer attached to a point *behind* the rear axle on the rear bumper. An example of a tractor-trailer vehicle where the pivot point is actually in front of the rear axle is a heavy pickup truck towing a gooseneck trailer. A gooseneck trailer has its central beam bent up and over the tailgate of the truck and attached to a ball joint in the middle of the bed. The pivot point is placed slightly forward of the rear axle to help stabilize the vehicle at highway speeds.

In our examples, we consider a kinematic model for multiple tractor-trailer vehicles which have attachment points offset from the rear axle of the preceding body; see Fig. 1 for an example of a tractor with three trailers. The kinematic model for this vehicle has been derived in [31]

$$\begin{aligned} \dot{x}_1 &= c_3 c_4 u_1 \\ \dot{x}_2 &= c_3 s_4 u_1 \\ \dot{x}_3 &= u_2 \\ \dot{x}_4 &= \frac{1}{l_1} s_3 u_1 \\ \dot{x}_5 &= f_5(x) u_1 \\ &= \frac{1}{l_2}\left[\left(c_3 s_4 - \frac{d_{12}}{l_1} s_3 c_4\right) c_5 - \left(c_3 c_4 + \frac{d_{12}}{l_1} s_3 s_4\right) s_5\right] u_1 \\ &\vdots \\ \dot{x}_n &= f_n(x) u_1 \quad n = 6, 7, \cdots \end{aligned} \tag{26}$$

$$\begin{aligned} &f_n(x) \\ &= \left[\left(c_3 s_4 - \frac{d_{12}}{l_1} s_3 c_4 - \sum_{i=5}^{n-1} (l_{i-3} + d_{i-3,i-2}) c_i f_i(x)\right) c_n \right. \\ &\quad \left. - \left(c_3 c_4 + \frac{d_{12}}{l_1} s_3 s_4 + \sum_{i=5}^{n-1} (l_{i-3} + d_{i-3,i-2}) s_i f_i(x)\right) s_n\right] \Big/ l_{n-3} \end{aligned}$$

where u_1 and u_2 denote the driving and steering velocities, $l_{i-3} + d_{i-3,i-2}$ is the total length of the $(i-4)$th trailer, c_i and s_i denote $\cos(x_i)$ and $\sin(x_i)$, respectively. Notice that even though Fig. 1 is drawn with the pivot points behind the axles, d's can be chosen negative, causing the pivot to be in front of the axle as in the case of the gooseneck trailer.

Consider now a chain of n steerable trailers pulled by the front wheel drive tractor; see Fig. 2 for an example of a tractor pulling a single steerable trailer. The kinematic equations for the front wheel drive tractor are exactly the same as in the previous case. However in the steerable trailer case, there is both an orientation and a steering angle associated with each trailer. By letting x_j denote the orientation

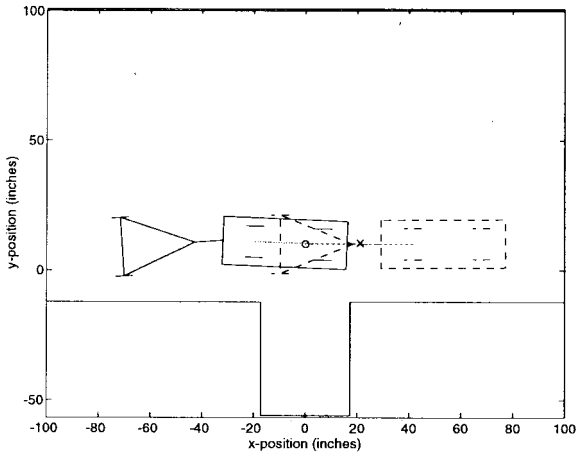

Fig. 3. Tractor-trailer initial guess path.

and x_k denote the steering angle, the kinematic model for the ith steerable trailer can be written in the following form:

$$\dot{x}_j = \left[\left(c_3 s_4 - \frac{d_{12}}{l_1} s_3 c_4 - \sum_{p=1}^{i-1} (l_{p+1} + d_{p+1,p+2}) c_{j_p} f_{j_p}(x)\right) \times \frac{c_{jk}}{c_k} - \left(c_3 c_4 + \frac{d_{12}}{l_1} s_3 s_4 + \sum_{p=1}^{i-1} (l_{p+1} + d_{p+1,p+2}) s_{j_p} f_{j_p}(x)\right) \frac{s_{jk}}{c_k}\right] \frac{u_1}{l_{i+1}}$$

$$\dot{x}_k = u_{i+2}$$

where $j_i = 2i + 3$ and $k = 2i + 4$ are the state indices for the ith trailer, l_{p+1} is the length of the pth trailer, $d_{p+1,p+2}$ is the distance between the pth trailer rear axle and the $(p + 1)$th trailer pivot point. Note that the model for steerable trailers has a singularity at $x_k = \pm\pi/2$. The physical significance of this singularity is that when the steering wheels of a trailer are pointing perpendicular to the central axis of the trailer it is not possible for the vehicle to move. Because of this fact, care must be taken in motion planning to ensure that the trailer steering wheels do not approach the singularity.

B. Simulation Results

To evaluate the usefulness of our path planning algorithm, we have applied it to a wide variety of wheeled vehicles in various situations. The examples range from the simplest nonholonomic system, a planar unicycle, to complicated vehicles such as a tractor with two trailers and a tractor with a steerable trailer. Examples presented in this section are intended to demonstrate the algorithm's versatility and its efficiency in planning paths which are challenging for average human drivers. The simulations presented in this section involve enforcing both convex and nonconvex constraints on configuration and nonconfiguration variables. The algorithm was implemented in C++ on an IBM compatible 80 486 33 MHz computer. In each of the following examples, the number of path points used (for constraint checking) is 101 unless otherwise specified. The absolute path error is calculated as the maximum excursion of the vehicle into the infeasible region over the entire path. The stopping criterion is chosen to be 0.01.

Tractor-Trailer: We present three cases for a tractor-trailer backing up into a loading dock. The first is totally unconstrained. The second case enforces the steering and jackknife angle constraints. The third case adds the no-collision constraint. The vehicle in this example consists of a front wheel drive car with a trailer attached to the rear bumper. The car has a wheel base of 26.5 in, is 48 in long, and 22 in wide. The trailer is 39 in long from pivot to rear axle, and is 22 in wide. The goal is to perform a docking motion in which the car must back the trailer into a 34 in wide loading bay. The kinematic model used for each of the three simulations is

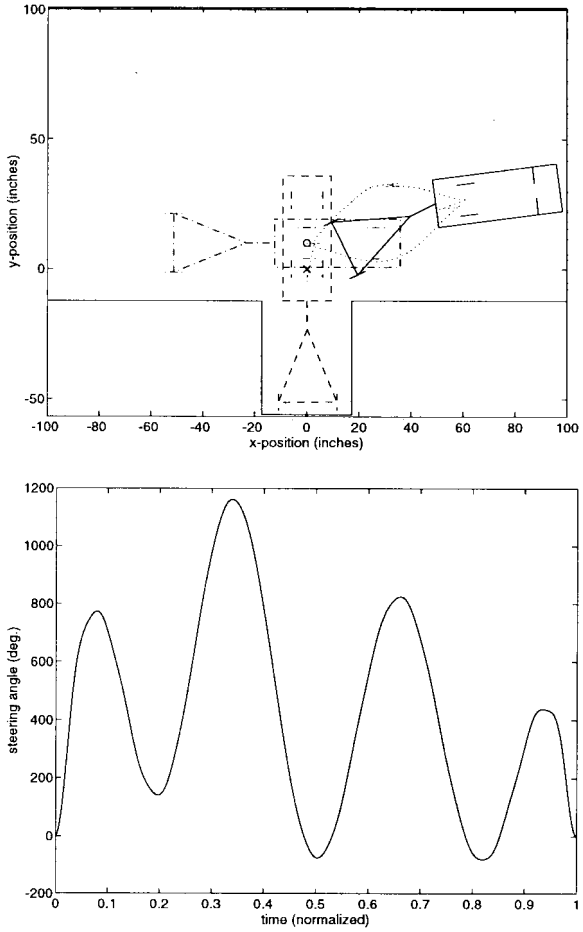

Fig. 4. Tractor-trailer unconstrained docking motion and steering angle.

$$\begin{aligned}
\dot{x}_1 &= c_3 c_4 u_1 \\
\dot{x}_2 &= c_3 s_4 u_1 \\
\dot{x}_3 &= u_2 \\
\dot{x}_4 &= 0.037\,74 s_3 u_1 \\
\dot{x}_5 &= 0.0256[(c_3 s_4 - 0.4622 s_3 c_4) c_5 \\
&\quad - (c_3 c_4 + 0.4622 s_3 s_4) s_5] u_1
\end{aligned} \tag{27}$$

where x_5 denotes the absolute orientation of the trailer. In the following discussions, the jackknife angle is defined as the difference between the car orientation and the trailer orientation.

In the first case, the initial guess simply involves the vehicle drives forward, backward, and then forward and is shown in Fig. 3. The number of Fourier basis elements used is 41. In computing the path, the algorithm took seven iterations with an average iteration time of 18.06 s. Fig. 4 shows the results. Notice that the steering angles go through some very unrealistically large values. This fact demonstrates the need to enforce inequality constraints in the algorithm.

In the second case, the initial guess of Fig. 3 is again used. The steering angle is now limited to $\pm 30^\circ$ from center and the jackknife angle is limited to $\pm 60^\circ$ from center. Under these conditions, the algorithm took six iterations to converge to an absolute error tolerance of 0.01 with the average iteration taking 17.71 s. Fig. 5 shows the

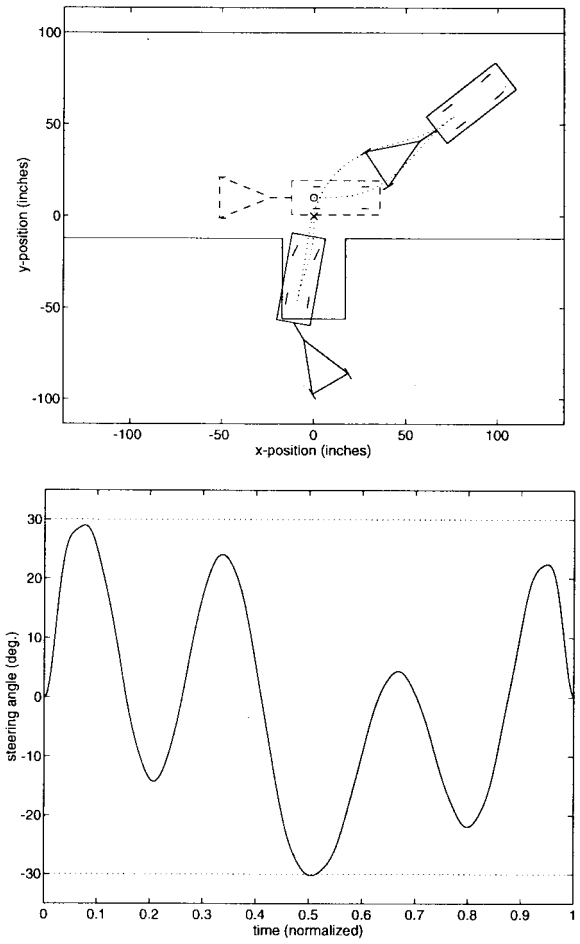

Fig. 5. Tractor-trailer docking with steering and jackknife angles constrained: Motion and steering angle.

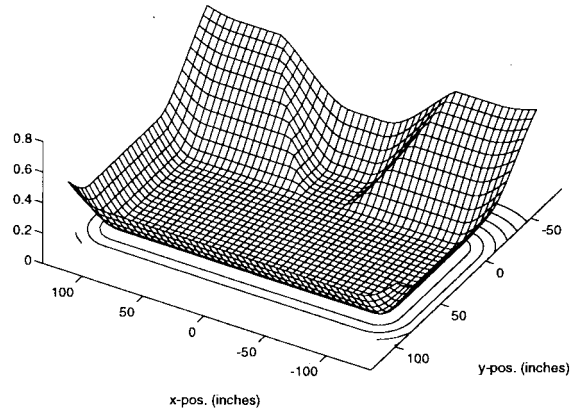

Fig. 6. Docking example contour map.

resulting path and steering angle. The vehicle begins at the "o" (represented by the dashed figure), drives forward and upward to the upper right hand location, back down to the lower location, and finally pulls forward to the desired final configuration. The final configuration is not shown in the figure but is the same as in Fig. 4. Notice that the steering angle stays within the specified tolerance.

In the final case, the path of Fig. 5 is used as the initial guess to enforce the task space constraints. The nonconvex task space constraints are enforced using the contour map method on a total of 28 points located on the periphery of both the tractor and the trailer.

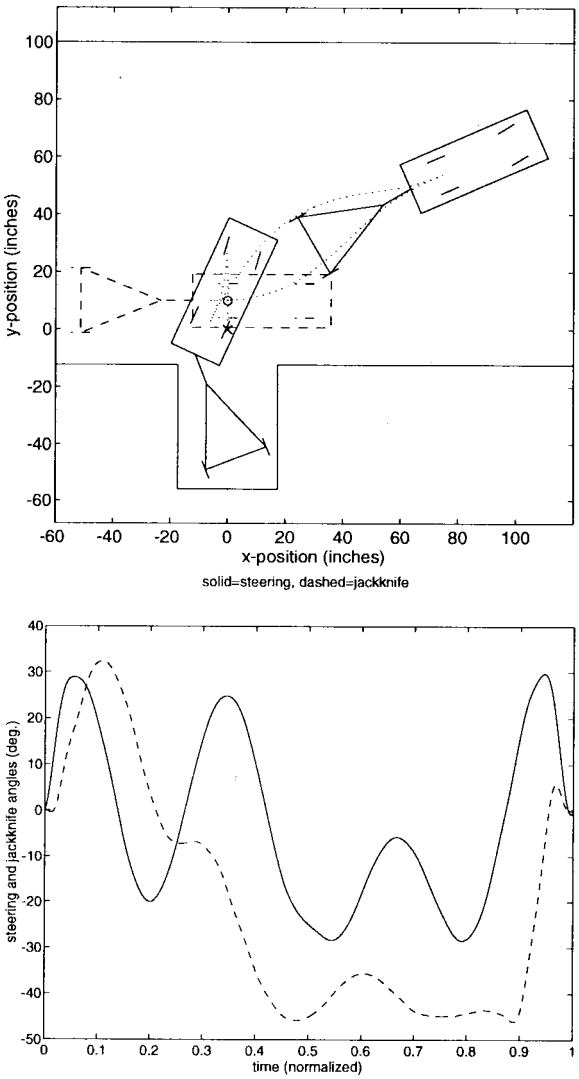

Fig. 7. Tractor-trailer docking all constraints enforced: Motion and steering and jackknife angles.

A coarse plot of the contour map used for this simulation is shown in Fig. 6. Notice the rectangular indentation representing the docking space in the plot. As in the previous simulations, 41 Fourier basis elements are used by the algorithm in computing the new path. The jackknife angle of the trailer was further limited in this simulation to $\pm 60^{\circ}$ from center. In this case the algorithm took 10 iterations to converge to an absolute error tolerance of 0.01. The average iteration time was 24.12 s. The resulting path is shown in Fig. 7 where the final configuration is the same as before. Fig. 7 also shows a plot of the steering and jackknife angles. Notice in these examples that as the vehicle and driving situation become more complicated, the number of iterations, and the average time per iteration increases.

Tractor with Steerable Trailer: Steerable trailer vehicles are of increasing interest because they present the possibility of increasing the maximum allowable trailer length for tractor-trailer vehicles used on our roads and highways. The problem with long trailers is that in turning a corner they tend to off-track more than shorter trailers. Off-tracking of a tractor-trailer at slow speeds occurs when the rear wheels of the trailer take a path inside that of the other wheels. A trailer equipped with steerable rear wheels can make turning corners at road intersections more manageable by eliminating off-tracking, as in the case of a ladder truck used by fire departments.

In this example, we require a tractor with a steerable trailer to perform the exact same task as in the previous case. In the first simulation, the docking maneuver is performed with only the steering and jackknife angles constrained. In the second simulation, the nonconvex task space constraints are applied in addition to the steering and jackknife angle constraints. As in the previous case, the front wheel steering angle is limited to $\pm 30^\circ$ from center and the jackknife angle is limited to $\pm 60^\circ$ from center. The rear wheels of the trailer are also limited to $\pm 30^\circ$ from center. The initial guess for this simulation is the same as for the fixed-trailer case but with the trailer steering input set to zero. The kinematic model used for this example is similar to the fixed model and is written as follows:

$$\begin{aligned}
\dot{x}_1 &= c_3 c_4 u_1 \\
\dot{x}_2 &= c_3 s_4 u_1 \\
\dot{x}_3 &= u_2 \\
\dot{x}_4 &= 0.037\,74 s_3 u_1 \\
\dot{x}_5 &= 0.0256\Big[(c_3 s_4 - 0.4622 s_3 c_4)\frac{c_{56}}{c_6} \\
&\qquad -(c_3 c_4 + 0.4622 s_3 s_4)\frac{s_{56}}{c_6}\Big] u_1 \\
\dot{x}_6 &= u_3
\end{aligned} \tag{28}$$

where x_6 is the steering angle of the trailer steering wheels, and u_3 is the trailer steering control input.

As in the previous trailer case, for each of the following simulations, the initial configuration has the vehicle centered at the point marked "o" and facing toward the right with the trailer in line with the tractor. The final configuration has the vehicle centered at the point marked "x" and facing upward with both tractor and trailer aligned. In each path plot, the initial and final configurations are not shown in order to reduce the clutter of the plots.

In the first simulation, 31 Fourier basis elements are used to represent the path and the control, respectively. In this case, the algorithm required five iterations with an average iteration time of 23.54 s. Compare these values with the six iterations at an average iteration time of 17.71 s in the fixed-trailer example. The added degree of freedom provided by the steerable trailer wheels causes the algorithm to take fewer iterations, but the time required to complete the average iteration increases since the input vector is larger than before. The resulting path is shown in Fig. 8. Notice that in this case, the path is much shorter and makes a much sharper turn around the right corner than does the fixed-trailer example. Fig. 8 also shows a plot of the steering and jackknife angles for this simulation. Notice that the trailer steering angle becomes quite large during the middle portion of the path. This portion corresponds to when the vehicle makes the corner around the obstacle.

The second simulation uses the result of the first simulation as an initial guess and enforces the nonconvex task space constraints on the same 28 points around the vehicle as in the fixed trailer example. The same contour map, shown in Fig. 6, was also used to represent the dock. In this case, the algorithm took five iterations to converge to a solution with an average iteration time of 23.54 s which is actually smaller than for the fixed trailer case. Fig. 9 shows the resulting path with the vehicle located at its farthest excursions in both the negative and positive x-directions. In both positions, the steering of the trailer can be clearly seen. Fig. 9 also shows the steering and jackknife angles achieved in this simulation. Notice that both the front wheel and trailer steering angles approach their $\pm 30^\circ$ limits in several places.

The examples in this section were chosen to demonstrate the versatility of the path planning algorithm and to demonstrate its applicability to challenging driving situations. However, even though the algorithm worked well for the nonconvex task space examples presented, there are many other situations for which the algorithm does not work. An illustrative example of one such scenario is the case where a vehicle is required to drive around an elongated obstacle. If the initial path is one in which the vehicle drives straight through the obstacle to the final end point, the algorithm will definitely have a local minima problem. This is because, if we think of the obstacle as a hill in the contour map with the negative gradient direction pointing straight out to the nearest boundary point, path points on either side of the obstacle will be pushed in opposite directions. As the algorithm progresses, it will reach a point where the effects of points on opposite sides of the obstacle cancel each other, thus causing the gradient matrix $G(\lambda)$ to lose rank. This example serves to illustrate that there is still much work to be done in developing a general algorithm.

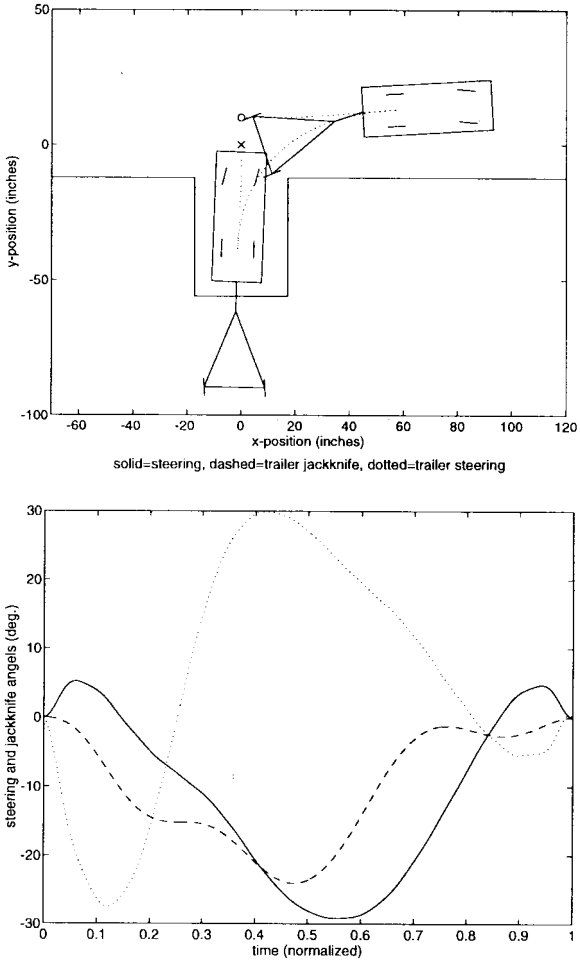

Fig. 8. Steerable tractor-trailer docking with steering and jackknife angles constrained: Motion and steering and jackknife angles.

V. Conclusion

We have presented an iterative path space algorithm for solving the nonholonomic motion planning problem. The method warps the entire path at each iteration to satisfy both equality and inequality constraints. We first transform the general nonlinear control problem associated with motion planning to a nonlinear least squares problem which can then be solved using many different algorithms. However, the resulting paths will often be physically unrealizable due to unrealistically large joint motions. To correct this problem, inequality constraints on the path need to be imposed. We use an exterior

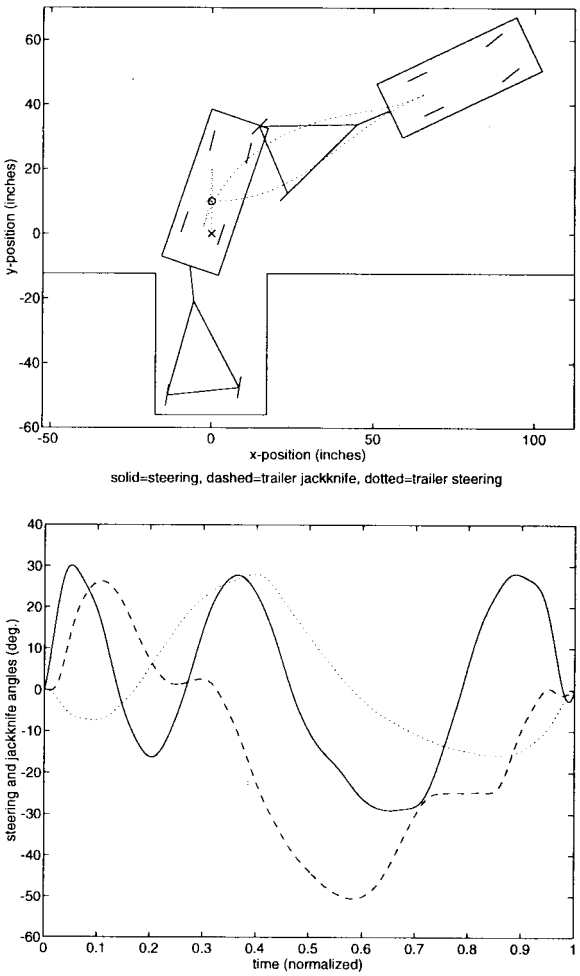

Fig. 9. Steerable tractor-trailer docking all constraints enforced: Motion and steering and jackknife angles.

penalty function to convert the inequality constraints into equality constraints which are in turn folded into the nonlinear least squares problem. Using this scheme, it is possible to enforce a wide variety of constraint types including joint limits, convex and nonconvex task space constraints, and constraints on both the configuration and nonconfiguration variables. The convergence of the algorithm depends on the avoidance of singular controls. In the absence of inequality constraints, certain genericity results are available in the literature. When inequality constraints are considered, we can show that exterior penalty functions do not add any singular control for the case of certain convex polyhedral constraints. Though in general the convergence has not been shown, and indeed may not always be true, we have demonstrated that the algorithm can tackle a large number of challenging problems including path planning for a tractor with multiple trailers. We have included the examples of a tractor trailer and a tractor and steerable trailer backing into a loading dock while satisfying a number of joint limit and collision avoidance constraints. We are currently working on expanding our understanding of singular control of specific classes of nonholonomic systems, developing general rules for the construction of the penalty functions, and extend our approach to include consideration of optimality.

ACKNOWLEDGMENT

The authors wish to acknowledge helpful discussion with D. Popa concerning exterior penalty functions and S. Middlebrooks concerning gooseneck trailers.

REFERENCES

[1] R. W. Brockett, "Asymptotic stability and feedback stabilization," in *Proc. Conf. Differential Geometric Contr. Theory*, Michigan Technol. Univ., June–July 1982, Progress in Math. Boston, MA: Birkhauser, 1983, vol. 27, pp. 181–208.
[2] J. Zabczyk, "Some comments on stabilizability," *Applied Math. Optimiz.*, vol. 19, pp. 1–9, 1989.
[3] P. Jacobs and J. Canny, "Robust motion planning for mobile robots," in *1991 IEEE R&A Workshop Nonholonomic Motion Planning*, Sacramento, CA, Apr. 1991.
[4] J. Barraquand, B. Langlois, and J.-C. Latombe, "Numerical potential field techniques for robot path planning," in *Proc. IEEE 5th Int. Conf. Advanced Robot.*, Pisa, Italy, June 1991, pp. 1012–1027.
[5] P. Jacobs, J.-P. Laumond, and M. Taix, "A complete iterative motion planner for a car-like robot," *J. Geom. Algorithm.*, 1990.
[6] L. Dorst, I. Mandhyan, and K. Trovato, "The geometrical representation of path planning problems," Tech. Rep., Philips Lab. North American Philips Corp., Briarcliff Manor, NY, 1991.
[7] B. Mirtich and J. Canny, "Using skeletons for nonholonomic path planning among obstacles," in *Proc. 1992 IEEE Robot. Automat. Conf.*, Nice, France, May 1992, pp. 2533–2540.
[8] T. Hague and S. Cameron, "Motion planning for nonholonomic industrial robot vehicles," in *IEEE/RSJ Int. Workshop Intell. Robots Syst.*, Osaka, Japan, Nov. 1991, pp. 1275–1280.
[9] T. Fraichard and C. Laugier, "On line reactive planning for nonholonomic mobile in a dynamic world," in *Proc. 1991 IEEE Robot. Automat. Conf.*, Sacramento, CA, Apr. 1991, pp. 432–437.
[10] S. G. Kong and B. Kosko, "Adaptive fuzzy systems for backing up a truck-and-trailer," *IEEE Trans. Neural Networks*, vol. 3, pp. 211–223, Mar. 1992.
[11] Z. X. Li and J. Canny, "Motion of two rigid bodies with rolling constraint," *IEEE Trans. Robot. Automat.*, vol. 6, pp. 62–72, Feb. 1990.
[12] G. Lafferriere and H. J. Sussmann, "Motion planning for controllable systems without drift: A preliminary report," Tech. Rep. SYCON-90-04, Rutgers Center for Syst. Contr., June 1990.
[13] R. M. Murray and S. S. Sastry, "Steering nonholonomic systems using sinusoids," in *Proc. 29th IEEE Conf. Decision and Contr.*, Honolulu, HI, 1990, pp. 2097–2101.
[14] Z. Li and L. Gurvits, "Theory and applications of nonholonomic motion planning," Tech. Rep., New York Univ., Courant Inst. of Math. Sci., July 1990.
[15] Y. Nakamura and R. Mukherjee, "Nonholonomic motion planning of space robots via bi-directional approach," in *Proc. 1990 IEEE Robot. Automat. Conf.*, Cincinnati, OH, 1990, pp. 1764–1769.
[16] Z. Vafa and S. Dubowsky, "On the dynamics of manipulators in space using the virtual manipulator approach," in *Proc. 1987 IEEE Robot. Automat. Conf.*, Raleigh, NC, Mar. 1987.
[17] A. M. Bloch, N. H. McClamroch, and M. Reyhanoglu, "Controllability and stabilizability properties of a nonholonomic control system," in *Proc. 29th IEEE Conf. Decision Contr.*, Honolulu, HI, 1990, pp. 1312–1314.
[18] B. d'Andrea Novel, G. Bastin, and G. Campion, "Modeling and control of nonholonomic wheeled mobile robots," in *Proc. 1991 IEEE Robot. Automat. Conf.*, Apr. 1991, pp. 1130–1135.
[19] R. M. Murray and S. S. Sastry, "Nonholonomic motion planning: Steering using sinusoids," *IEEE Trans. Automat. Contr.*, vol. 38, pp. 700–716, 1993.
[20] H. J. Sussmann and Y. Chitour, "A continuation method for nonholonomic path finding problem," in *IMA Workshop Robot.*, Jan. 1993.
[21] E. D. Sontag, "Non-singular trajectories, path planning, and time-varying feedback for analytic systems without drift," *IMA Workshop Robot.*, Jan. 1993.
[22] ——, "Control of systems without drift via generic loops," *IEEE Trans. Automat. Contr.*, vol. 40, pp. 1210–1219, July 1995.
[23] H. Sussman, "A continuation method for nonholonomic path-finding problems," in *Proc. 32nd IEEE Conf. Decision Contr.*, San Antonio, TX, 1993, pp. 2718–2723.
[24] C. N. Dorny, *A Vector Space Approach to Models and Optimization*. FL: R. E. Krieger, 1986.
[25] D. G. Luenberger, *Linear and Nonlinear Programming*. Reading, MA: Addison-Wesley, 1984, 2nd ed.
[26] D. E. Koditschek, "Natural motion for robot arms," in *Proc. IEEE Conf. Decision Contr.*, Las Vegas, NV, 1984, pp. 733–735.
[27] J.-C. Latombe, *Robot Motion Planning*. Norwell, MA: Kluwer Academic, 1991.
[28] O. Khatib, "Augmented object concept in multi-robot manipulator

control," in *Proc. 1988 Amer. Contr. Conf.*, Atlanta, GA, June 1988, pp. 2140–2147.
[29] G. Oriolo and Y. Nakamura, "Free-joint manipulators: Motion control under second-order nonholonomic constraints," in *IEEE/RSJ Int. Workshop Intell. Robots Syst.*, Osaka, Japan, Nov. 1991, pp. 1248–1253.
[30] D. Popa and J. T. Wen, "Nonholonomic path planning with obstacle avoidance: A path space approach," in *Proc. 1996 IEEE Robot. Automat. Conf.*, Minneapolis, MN, Apr. 1996.
[31] A. W. Divelbiss, "Nonholonomic motion planning in the presence of obstacles," Ph.D. dissertation, Rensselaer Polytech. Inst., Troy, NY, Dec. 1993.
[32] J.-P. Laumond, "Controllability of a multibody mobile robot," in *Int. Conf. Advanced Robot.*, Pisa, Italy, June 1991, pp. 1033–1038.
[33] O. J. Sørdalen, "Conversion of the inematics of a car with n trailers into a chained form," in *Proc. 1993 IEEE Robot. Automat. Conf.*, May 1993, pp. 382–387.

High-Order Neural Networks for the Learning of Robot Contact Surface Shape

Elias B. Kosmatopoulos and Manolis A. Christodoulou

***Abstract*—It is known that the problem of learning the shape parameters of unknown surfaces that are in contact with a robot end-effector can be formulated as a nonlinear parameter estimation problem and an extended Kalman filter can be applied in order to estimate the surface shape parameters. In this paper, we show that the problem of learning the shape parameters of unknown contact surfaces can be formulated as a linear parameter estimation problem and thus globally convergent learning laws can be applied. Moreover, we show that by using appropriate neural network approximators, the unknown surfaces can be learned even if there are no force measurements, i.e., the robot is not provided with any force or tactile sensors.**

***Index Terms*—Adaptive observers, high order neural networks, parameter estimation, robot contact surfaces.**

I. Introduction

When it is desired to design robot manipulators that will operate in uncertain and unknown environments, it is sometimes necessary to provide them with appropriate devices and algorithms that are capable of estimating and learning unknown surfaces that are in contact with the robots' end-effectors. In fact, it can be shown that the contact force F is given by $F = \Lambda(t, \vartheta)$ where ϑ is the vector of the unknown surface shape parameters, and $\Lambda(t, \cdot)$ is a known nonlinear function. Therefore, we can apply an extended Kalman filter or any other nonlinear parameter estimation method to estimate the unknown surface parameters ϑ [1]. However, nonlinear parameter estimation algorithms do not guarantee the convergence of the estimated parameter vector to the actual parameter vector ϑ; due to the nonlinear dependence of the function $\Lambda(\cdot)$ on the parameter vector ϑ, there are many local minima of the error functional that the parameter estimation algorithm minimizes. Therefore the nonlinear parameter estimation algorithm might get trapped into a local minimum, and thus, the estimation procedure will fail.

In this paper, we show that by using appropriate neural network approximators we can formulate the unknown surface estimation problem as a linear parameter estimation problem, and therefore a linear parameter estimation algorithm can be applied. The linear parameter estimation algorithms, contrary to nonlinear ones, ensure stability and convergence to the unique global minimum. Furthermore, we propose a new learning architecture that is capable of estimating the unknown surface shape parameters even in the case where the contact force F is not available for measurement. As we show, the whole scheme is globally stable and convergent. The proposed algorithms make use of high-order neural network approximators, which are linear with respect to their adjustable parameters. Our methodology is applicable to any surface that can be described by a smooth function of the Cartesian coordinates.

A. Notations and Preliminaries

$f \in \mathcal{L}_\infty$ means that $\sup_{t\geq 0} |f(t)| < \infty$; ∇ and ∇^2 denote the gradient and Hessian operators, respectively. $f \to \mathcal{R}(g_1, g_2, \cdots, g_n)$ means that $f(t)$ converges to the set $\{f : |f|^2 \leq \sum_{i=1}^n a_i \sup_{t\geq 0} |g_i(t)|^2\}$, $a_i \geq 0$. $f \overset{exp}{\to} \mathcal{R}(g_1, g_2, \cdots, g_n)$ means that the convergence is exponential. We say that the function $f(t)$ is persistently exciting (symbolically $f \in PE$) if there are positive constants β_1, β_2, δ such that $\beta_1 I \leq \int_t^{t+\delta} f(s) f^T(s)\, ds \leq \beta_2 I < \infty$, $\forall t \geq 0$. Consider now the following error systems, which will be frequently met in the rest of the paper **(E1)** $e(t) = \tilde{W}(t)\zeta(t) + \mu(t)$ and **(E2)** $\dot{e} = Ae + \tilde{W}\zeta(t) + \mu(t)$ where $e, \tilde{W}, \zeta, \mu$ denote the so-called identification error, parameter estimation error, regressor vector, and modeling error term, respectively. The matrix A is a constant stability matrix. The parameter estimation error is defined as the difference between an unknown constant matrix W^* and W which is the known current estimate of W^*. Consider now the following learning law

$$\dot{W} = \mathcal{P}_{\mathcal{C}}\Big[-\Pi e(t)\zeta^T(t); W(t)\Big] \tag{1.1}$$

where Π is a positive definite design matrix and $\mathcal{P}_{\mathcal{C}}[\cdot;\cdot]$ denotes the projection operator defined as follows: $\mathcal{P}_{\mathcal{C}}[y, W] = y$ if $C(W) \leq 0$ or $\frac{\partial C}{\partial W}(W)y \leq 0$ and

$$\mathcal{P}_{\mathcal{C}}[y, W] = y - \frac{C(W)\frac{\partial C}{\partial W}(W)y}{|\frac{\partial C}{\partial W}(W)|^2}\frac{\partial C}{\partial W}(W)^T$$

otherwise, where $\mathcal{C}$ is a convex set satisfying $W^* \in \mathcal{C}$, and $C(W)$ is a function satisfying $C(W) \leq 0$ iff $W \in \mathcal{C}$. The learning law (1.1) is known as the parameter projection adaptive law [9], [3]. The next theorem summarizes some of the properties of this adaptive law [3].

Theorem 1: Consider either the error system **(E1)** or the system **(E2)** and the learning law (1.1). Assume that $\zeta, \dot{\zeta}, \mu \in \mathcal{L}_\infty$. Then the following statements hold: 1) $e, W, \dot{W} \in \mathcal{L}_\infty$ and $W(t) \in \mathcal{C}$ for all t, provided that $W(0) \in \mathcal{C}$, 2) $e \to \mathcal{R}(\mu)$, 3) if $\mu(t) = 0$ then $\lim_{t\to\infty} e(t) = 0$, 4) $\tilde{W}\zeta \to \mathcal{R}(\mu)$, and finally 5) If $|W^*| \in \mathcal{C}$ and $\zeta \in PE$ then $\tilde{W} \overset{exp}{\to} \mathcal{R}(\mu)$.

Manuscript received August 9, 1995; revised April 23, 1996. This paper was recommended for publication by Associate Editor M. Spong and Editor S. Salcudean upon evaluation of the reviewers' comments.

E. B. Kosmatopoulos is with the Department of Electrical Engineering — Systems, University of Southern California, Los Angeles, CA 90089 USA (e-mail: kosmatop@bode.usc.edu).

M. A. Christodoulou is with the Department of Electronic and Computer Engineering, Technical University of Crete, 73100 Chania, Crete, Greece (e-mail: manolis@ced2.tuc.gr).

Publisher Item Identifier S 1042-296X(97)03816-0.