Appl. Math. and Comp. Sci., 1997, Vol.7, No.2, 433–458

TIME-OPTIMAL TRAJECTORY PLANNING IN DYNAMIC ENVIRONMENTS

PAOLO FIORINI*, ZVI SHILLER**

This paper presents a direct method for computing the time-optimal trajectory for a robot among stationary and moving obstacles, subject to the robot's dynamics and actuator constraints. The motion planning problem is first formulated as an optimization problem, and then solved numerically using a gradient descent. The initial guess for the optimization is generated using a method based on the concept of Velocity Obstacles. The method is demonstrated for a 2-DOF planar manipulator moving in static and dynamic environments.

1. Introduction

Motion planning is central to the operation of autonomous robots. It concerns the generation of a trajectory from start to goal that satisfies objectives, such as minimizing path distance or motion time, while avoiding obstacles in the environment and satisfying the robot mechanics (kinematics and dynamics). We distinguish between *planning* and *control* in that the former generates a nominal trajectory, whereas the latter tracks that trajectory. Robot motion planning is generally too complex to be handled by on-line feedback controllers due to the nonlinear state constrains introduced by the obstacles, and the highly nonlinear and coupled nature of robot mechanics.

Traditionally, motion planning has been treated as a *kinematic* problem, i.e. determining the path that avoids obstacles without concern to robot speeds. This was first extensively addressed for articulated robots by transforming the problem into the *configuration space*, in which the robot reduces to a point and the obstacles map into C-space obstacles (Latombe, 1991; Lozano-Pérez and Wesley, 1979). The focus in this body of work has centered on computational complexity and completeness (the ability of the algorithm to find a path if one exists). More recently the *kinematic* problem was extended to car-like robots, which are subject to *non-holonomic* kinematic constraints due to the assumption of no slip between the wheels and ground. Here the focus has centered on obstacle avoidance (Laumond *et al.*, 1994) and on minimizing path distance (Laumond, 1987).

While solving a problem fundamental to robotics, *kinematic* motion planning ignores the important effects of robot dynamics which become significant at all but

* Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA 91109 USA, e-mail: fiorini@telerobotics.jpl.nasa.gov.

** Department of Mechanical and Aerospace Engineering, University of California, Los Angeles, CA 900 95, USA, e-mail: shiller@seas.ucla.edu.

the lowest speeds. For example, non-holonomic motion planning of a car is useful for parking (Murray and Sastry, 1993), which is usually done at very low speeds, but is all but meaningless for high speed emergency maneuvers (Shiller and Sundar, 1996). Similarly, obstacle-free paths, computed using robot kinematics only may be dynamically infeasible at even moderate speeds, causing the robot to deviate from the *kinematic* path due to its dynamics and limited actuator efforts. This gave rise to *dynamic* motion planning,[1] which produces a *trajectory* in the state space rather than just a *path* in the configuration space. Planning in the state space, while computationally more extensive, allows one to minimize *dynamic* cost functions, such as time or energy. These problems have been treated previously for both articulated (Shiller, 1996; Shiller and Dubowsky, 1989) and mobile robots (Shiller and Gwo, 1991).

We distinguish between motion planning in *static* and in *dynamic* environments. In static environments, the obstacles are static, and the robot is the only one that moves, whereas in dynamic environments, both the robot and obstacles move. Typical examples of dynamic environments include manufacturing tasks in which robot manipulators track and retrieve parts from moving conveyers, and intelligent vehicles negotiating freeway traffic.

Motion planning in dynamic environments was originally addressed by adding the time dimension to the robot's configuration space, assuming bounded velocity and known trajectories of the obstacles. Reif and Sharir (1985) solved the planar problem for a polygonal robot among many moving polygonal obstacles by searching a visibility graph in the configuration-time space. Erdmann and Lozano-Pérez (1987) discretized the configuration-time space to result in a sequence of configuration space slices at successive time intervals. This method essentially solves the static planning problem at every slice and joins adjacent solutions. Fujimura and Samet (1989a) used cell decomposition to represent the configuration-time space, and joined empty cells to connect start to goal.

Another approach to dynamic motion planning is to decompose the problem into smaller problems: path planning and velocity planning. This method first computes a feasible path among the static obstacles, and represents it as a parametric curve in the arc length. Then, the intersections of the moving obstacles with the path are represented as forbidden regions in an arc length-time plane. The velocity along the path is chosen to avoid the forbidden regions. Kant and Zucker (1986) selected both the path and velocity profile using a visibility graph approach. Lee and Lee (1987) developed independently a similar approach for two cooperating robots, and compared the effects of delay and velocity reduction on motion time. Fraichard (1993) considered acceleration bounds, and used a search in a state-time space $(s, \dot{s}, t)$ to compute the velocity profile yielding a minimum-time trajectory. Fraichard and Laugier (1993) considered adjacent paths that could be reached from the nominal path when the nominal path becomes blocked by a moving obstacle. Fujimura (1995) considered the case of a robot moving on a fixed time-dependent network, whose nodes could be temporarily occluded by moving obstacles.

[1] Others use *dynamic* motion planning to denote motion planning in *dynamic* environments (Latombe, 1991), which is a subset of our definition.

A different approach consists in generating the accessibility graph of the environment, which is an extension of the visibility graph (Fujimura and Samet, 1989b; 1990). Fujimura and Samet (1989b) defined it as the locus of points on the obstacles which are reachable by the robot moving at maximum speed. These points form the *collision front*, and can be linked together to construct a path from start to goal. The accessibility graph has the property that, if the robot moves faster than the obstacles, the path computed by searching the graph is time-minimal. This concept was extended in (Fujimura, 1994) to the case of slowly moving robots and transient obstacles, i.e. obstacles that could appear and disappear in the environment.

None of the previous methods considered the non-linear robot dynamics, and none produced time optimal motions. Time-optimal motions have obvious benefits in industrial applications by reducing cycle times and thus increasing the productivity of automated manufacturing systems. Other application domains, such as intelligent vehicles and air traffic control, may benefit from time-optimal motions by minimizing the recovery time from emergency situations.

The time-optimal motion planning problem in static environments has been treated previously, beginning with the work by Kahn and Roth (1971) who solved the problem for a linearized robot model, using the Pontryagin Minimum Principle (PMP). The full robot model was used in (Meier, 1987), assuming bang-bang control and using a steepest descent over the switching times. However, the most efficient methods to date seem to consist of parameter optimizations over the trajectory (Bobrow, 1988; Johnson and Gilbert, 1985; Shiller and Dubowsky, 1989), which are similar to the Differential Inclusions introduced in (Seywald, 1994), and the Inverse Dynamic Optimization introduced by Bryson (1995).

In this paper, we present a method for computing the time-optimal trajectories of a robot moving in a dynamic environment. To make the problem computationally tractable, we restrict the treatment to the plane and assume a point robot and circular obstacles. We also assume full knowledge of the environment.

Central to this approach is the computation of the initial guess for the optimization. This is done by utilizing the concept of Velocity Obstacle (Fiorini, 1995), which maps the dynamic environment into the robot velocity space. The velocity obstacle is the first-order approximation of the robot's velocities that would cause a collision with an obstacle at some future time, within a given time horizon. Feasible avoidance maneuvers are computed simply by selecting velocities outside the velocity obstacle, and satisfying additional velocity constraints computed from robot dynamics and actuator constraints. The initial guess of the optimal trajectory is computed by a global search over a tree of feasible avoidance maneuvers, generated at discrete time intervals so as to minimize time to the goal.

The optimal trajectory is computed using a steepest descent algorithm over the admissible controls (Bryson and Denham, 1962; Bryson and Ho, 1975; Denham and Bryson, 1964), modified to consider time varying state inequality constraints. The state inequality constraints due to the moving obstacles are considered by transforming them into state-dependent control constraints. The method was implemented for intelligent vehicles negotiating freeway traffic (Shiller and Fiorini, 1995), and for a

planar SCARA robot, considering its full nonlinear dynamics and moving circular obstacles (Fiorini, 1995). Examples of the latter are presented in this paper.

The paper is organized as follows. Section 2 formulates the motion planning problem as a minimum-time problem. It also presents the numerical method for computing the optimal solution satisfying state inequality constraints and state-dependent control constraints. Section 3 addresses the problem of generating the nominal trajectory for the numerical optimization. Finally, examples of optimal trajectories of a SCARA robot avoiding fixed and moving obstacles are presented in Section 4.

2. Dynamic Optimization

The dynamic motion planning problem in the context of this paper consists in determining the trajectory between two specified boundary conditions that avoids all static and moving obstacles and minimizes motion time. This is formulated as an optimization problem with time-varying state constraints, and is solved numerically using the steepest descent method (Denham and Bryson, 1964), as discussed next.

2.1. Problem Formulation

The motion planning problem can be formulated as follows: Find the control $\boldsymbol{u}^*(t) \in \boldsymbol{U}$ in $t_0 \leq t \leq t_f$, which minimizes the cost function J:

$$\min_{\boldsymbol{u}(t)\in\boldsymbol{U}} J = \min_{\boldsymbol{u}(t)\in\boldsymbol{U}} \int_{t_0}^{t_f} L\big(\boldsymbol{x}(t),\boldsymbol{u}(t)\big)\,\mathrm{d}t = \phi_{\min}\big(\boldsymbol{x}(t_f),t_f\big) = t_f \tag{1}$$

where t_f is free, subject to robot dynamics

$$\dot{\boldsymbol{x}} = \mathcal{F}(\boldsymbol{x},\boldsymbol{u}) = f(\boldsymbol{x}) + \boldsymbol{g}(\boldsymbol{x})\boldsymbol{u} \tag{2}$$

admissible controls

$$\boldsymbol{U} = \{\boldsymbol{u} \mid \boldsymbol{u}_{\min} \leq \boldsymbol{u} \leq \boldsymbol{u}_{\max}\} \tag{3}$$

initial conditions

$$\boldsymbol{x}(t_0) = \boldsymbol{x}_0 \tag{4}$$

terminal manifold

$$\Omega\big(\boldsymbol{x}(t_f),t_f\big) = 0 \tag{5}$$

and state inequality constraints due to the moving obstacles:

$$\Psi : \bigcup_{i=1}^{n} \Big[S_i\big(\boldsymbol{x}(t),t\big) \geq 0\Big] \tag{6}$$

where $S_i(\boldsymbol{x}(t),t)$ represents the time-varying boundaries of the moving obstacles.

The original problem calls for a fixed final point. However, we assume instead a terminal manifold (a hyper-sphere around the final point) so that we can use influence

functions to compute the initial conditions of the Lagrange multipliers, and thus avoid using the more sensitive shooting method (Bryson, 1992).

State inequality constraints are generally difficult to satisfy although necessary conditions for optimality have been developed for such problems (Jacobson *et al.*, 1971; Speyer and Bryson, 1968). One way to consider state inequality constraints is to transform them into state-dependent control *equality* constraints, active only when the robot slides along the obstacle boundary (Bryson and Ho, 1975; Denham and Bryson, 1964).

To demonstrate the treatment of the state inequality constraints, we consider a single obstacle:

$$\Psi : S\big(\boldsymbol{x}(t), t\big) \geq 0, \quad S(\boldsymbol{x}) \in \mathbb{R}^m \tag{7}$$

where m is the dimension of the position space. Differentiating (7) with respect to time p times until it becomes explicit in the control $\boldsymbol{u}$, and assuming an active constraint, yields the state-dependent control constraint

$$S^{(p)}(\boldsymbol{x}, \boldsymbol{u}) = 0 \tag{8}$$

where $S^{(p)}$ denotes the p-th derivative of S, with p being the order of the constraint.

A solution satisfying (8) does not necessarily satisfy (7), unless it passes through at least one point satisfying (7) and all the derivatives of order less than p. We choose this point to be the initial entry point of the constrained arc, at time $t_1 > t_0$. The inequality constraint (7) is thus replaced by the tangency point condition, Ψ_1, and the state-dependent equality constraint, Ψ_2:

$$\Psi_1 : \begin{pmatrix} S\big(\boldsymbol{x}(t_1), t_1\big) = 0 \\ \dot{S}\big(\boldsymbol{x}(t_1), t_1\big) = 0 \\ \vdots \\ S^{(p-1)}\big(\boldsymbol{x}(t_1), t_1\big) = 0 \end{pmatrix} \tag{9}$$

$$\Psi_2 : S^{(p)}\Big(\boldsymbol{x}(t), \boldsymbol{u}(t), t\Big) = 0, \qquad t_1 < t \leq t_2 \tag{10}$$

where t_1 is the entry time, and t_2 is the exit time of the constrained arc. This also modifies the admissible controls (3) to:

$$\boldsymbol{U} : \begin{cases} \boldsymbol{U} : \boldsymbol{u}_{\min} \leq \boldsymbol{u} \leq \boldsymbol{u}_{\max} \\ S^{(p)}(\boldsymbol{x}(t), \boldsymbol{u}(t), t) = 0 \quad \text{for} \quad t \in (t_1, t_2] \end{cases} \tag{11}$$

The addition of the tangency constraint, Ψ_1, thus transforms the original Two-Point Boundary-Value Problem (1) into a Three-Point Boundary-Value Problem (for a single moving obstacle), which is solved numerically using the method discussed later.

Note that this treatment of the state inequality constraints may over-constrain the problem since the trajectory is forced to satisfy the state constraint as an equality

along a finite arc. Consequently, this approach cannot find solutions that touch the state constraint at multiple isolated points (Jacobson *et al.*, 1971). This, however, has been shown to affect only constraints of order higher than two, and hence is not an issue for the circular obstacles treated here (Jacobson *et al.*, 1971).

2.2. Numerical Computation

We apply the steepest descent method which satisfies rigorously a set of necessary optimality conditions. This method was originally developed in (Bryson and Denham, 1962), modified to include state dependent control inequality constraints in (Denham and Bryson, 1964), and modified to consider bang-bang controls in (Meier, 1987).

The steepest descent method computes iteratively the optimal controls by following the negative gradient of the augmented cost function with respect to the controls and the final time. The gradient is derived by adjoining the differential of the cost function with the differentials of the terminal manifold and the tangency-point constraint, as discussed below.

2.2.1. Differential of the Performance Index

Following the classic approach to constrained optimization (Bryson and Ho, 1975), system dynamics and control constraints are adjoined to the performance index J using two arrays of Lagrange functions, $\lambda_\phi(t) \in \mathbb{R}^n$ and $\mu(t) \in \mathbb{R}^k$, where n is the dimension of the state space, and k is the number of active state-dependent control constraints. This leads to the performance index $\widetilde{J}$:

$$\widetilde{J} = \phi\big(\boldsymbol{x}(t_f)\big) + \int_{t_0}^{t_f} \Big[\lambda_\phi^T\big(\mathcal{F}(\boldsymbol{x},\boldsymbol{u}) - \dot{\boldsymbol{x}}\big) + \mu^T\varphi(\boldsymbol{x},\boldsymbol{u})\Big]\mathrm{d}\tau \tag{12}$$

where

$$\varphi = \begin{cases} 0 & \text{if} \quad t \notin (t_1, t_2) \\ S^{(p)} & \text{if} \quad t \in (t_1, t_2) \end{cases} \tag{13}$$

and μ is a vector of Kuhn-Tucker multipliers (Bryson and Ho, 1975):

$$\mu = \begin{cases} 0 & \text{when } \varphi = 0 \\ -\boldsymbol{\Lambda}^T\boldsymbol{g}(\boldsymbol{x})\left(\dfrac{\partial\varphi(\boldsymbol{x},\boldsymbol{u})}{\partial\boldsymbol{u}}\right)^{-1} & \text{otherwise} \end{cases} \tag{14}$$

By defining the Hamiltonian as:

$$\mathcal{H}(\lambda_\phi, \boldsymbol{x}, \boldsymbol{u}) = \lambda_\phi^T\mathcal{F}(\boldsymbol{x},\boldsymbol{u}) + \mu^T\varphi(\boldsymbol{x},\boldsymbol{u}) \tag{15}$$

and by choosing:

$$\dot{\lambda}_\phi(t) = -\left(\frac{\partial\mathcal{H}}{\partial\boldsymbol{x}}\right)^T \tag{16}$$

$$\lambda_\phi(t_f) = \left(\frac{\partial \phi}{\partial \boldsymbol{x}}\right)_{t_f} \tag{17}$$

we reduce the differential $\mathrm{d}\widetilde{J}$ to:

$$\mathrm{d}\widetilde{J} = \int_{t_0}^{t_f} \frac{\partial \mathcal{H}}{\partial \boldsymbol{u}} \delta \boldsymbol{u}\, \mathrm{d}\tau + \left(\frac{\partial \phi}{\partial t} + \mathcal{H}\right)_{t_f} \mathrm{d}t_f \tag{18}$$

This establishes the relations between variations in the independent variables, $\boldsymbol{u}$ and t_f, and variations in the cost function for the unconstrained problem.

2.2.2. Differential of the Terminal Constraint

The differential of the terminal constraint Ω is:

$$(\mathrm{d}\Omega)_{t_f} = \left(\frac{\partial \Omega}{\partial \boldsymbol{x}}\mathrm{d}\boldsymbol{x} + \frac{\partial \Omega}{\partial t}\mathrm{d}t\right)_{t_f} \tag{19}$$

Following the derivation in Appendix A, and choosing multipliers $\lambda_\Omega \in \mathbb{R}^n \times \mathbb{R}^l$ (where l is the number of terminal constraints) to satisfy:

$$\dot{\lambda}_\Omega(t) = -\left(\frac{\partial \mathcal{F}}{\partial \boldsymbol{x}}\right)^T \lambda_\Omega(t) \tag{20}$$

$$\lambda_\Omega(t_f) = \left(\frac{\partial \Omega}{\partial \boldsymbol{x}}\right)_{t_f} \tag{21}$$

the differential $\mathrm{d}\Omega$ (19) reduces to:

$$(\mathrm{d}\Omega)_{t_f} = \int_{t_0}^{t_f} \lambda_\Omega^T \frac{\partial \mathcal{F}}{\partial \boldsymbol{u}} \delta \boldsymbol{u}\, \mathrm{d}\tau + \left(\frac{\partial \Omega}{\partial \boldsymbol{x}}\dot{\boldsymbol{x}} + \frac{\partial \Omega}{\partial t}\right)_{t_f} \mathrm{d}t_f \tag{22}$$

2.2.3. Differential of the Point Constraint

Similarly, the differential of the intermediate tangency constraints, Ψ_1, at time t_1, is:

$$(\mathrm{d}\Psi_1)_{t_1} = \left(\frac{\partial \Psi_1}{\partial \boldsymbol{x}}\mathrm{d}\boldsymbol{x} + \frac{\partial \Psi_1}{\partial t}\mathrm{d}t\right)_{t_1} \tag{23}$$

Following the derivation in Appendix A, and choosing the Lagrange functions $\lambda_\Psi \in \mathbb{R}^n \times \mathbb{R}^k$ (where k is the number of constraints Ψ_1)

$$\dot{\lambda}_\Psi(t) = -\left(\frac{\partial \mathcal{F}}{\partial \boldsymbol{x}}\right)^T \lambda_\Psi(t) \tag{24}$$

$$\lambda_\Psi(t_1) = \left(\frac{\partial \Psi}{\partial \boldsymbol{x}}\right)_{t_1} \tag{25}$$

the differential $\mathrm{d}\Psi$ (23) reduces to:

$$(\mathrm{d}\Psi_1)_{t_1} = \int_{t_0}^{t_1} \lambda_\Psi^T \frac{\partial \mathcal{F}}{\partial \boldsymbol{u}} \delta \boldsymbol{u}\, \mathrm{d}\tau + \left(\frac{\partial \Psi}{\partial \boldsymbol{x}}\dot{\boldsymbol{x}} + \frac{\partial \Psi}{\partial t}\right)_{t_1} \mathrm{d}t_1 \tag{26}$$

2.2.4. Discontinuity of the Lagrange Functions

The Lagrange functions, λ_ϕ and λ_Ω, are integrated through the entry point of the constrained arc at t_1, where they are discontinuous. This discontinuity is computed as a function of the jump in the acceleration (for a second-order system) across the entry point to the constrained arc (Bryson *et al.*, 1963) (see also Appendix B):

$$\lambda_\phi^T(t_1^-) = \lambda_\phi^T(t_1^+)\left(I - \frac{\dot{\boldsymbol{x}}(t_1^-) - \dot{\boldsymbol{x}}(t_1^+)}{S^{(p)}(t_1)} \left.\frac{\partial S^{(p-1)}}{\partial \boldsymbol{x}}\right|_{t_1}\right) \tag{27}$$

$$\lambda_\Omega^T(t_1^-) = \lambda_\Omega^T(t_1^+)\left(I - \frac{\dot{\boldsymbol{x}}(t_1^-) - \dot{\boldsymbol{x}}(t_1^+)}{S^{(p)}(t_1)} \left.\frac{\partial S^{(p-1)}}{\partial \boldsymbol{x}}\right|_{t_1}\right) \tag{28}$$

2.2.5. Differential of the Augmented Performance Index

The differential of the augmented performance index $\mathrm{d}\underline{J}$ consists of the differentials (22) and (26), appended to the differential $\mathrm{d}\tilde{J}$ with constant multipliers η and ν:

$$\begin{aligned}\mathrm{d}\underline{J} = &\left[\frac{\partial \phi}{\partial t} + \nu^T \frac{\partial \Omega}{\partial t} + \left(\frac{\partial \phi}{\partial \boldsymbol{x}} + \nu^T \frac{\partial \Omega}{\partial \boldsymbol{x}}\right)\dot{\boldsymbol{x}} + \mu^T \varphi(\boldsymbol{x}, \boldsymbol{u})\right]_{t_f} \mathrm{d}t_f \\ &+ \int_{t_0}^{t_1^-} \left[\frac{(\lambda_\phi^T + \nu^T \lambda_\Omega^T + \eta^T \lambda_\Psi^T)\mathcal{F} + \mu^T \varphi(\boldsymbol{x}, \boldsymbol{u})}{\partial \boldsymbol{u}}\right] \delta \boldsymbol{u}\, \mathrm{d}\tau \\ &+ \int_{t_1^+}^{t_f} \left[\frac{(\lambda_\phi^T + \nu^T \lambda_\Omega^T)\mathcal{F} + \mu^T \varphi(\boldsymbol{x}, \boldsymbol{u})}{\partial \boldsymbol{u}}\right] \delta \boldsymbol{u}\, \mathrm{d}\tau\end{aligned} \tag{29}$$

Note that the multipliers λ_Ψ are defined only between t_0 to t_1, since Ψ_1 is not affected by the states after t_1. Setting $\lambda_\Psi(t) = 0$ for $t > t_1$ we define an augmented Lagrange function, $\boldsymbol{\Lambda}$:

$$\boldsymbol{\Lambda}^T = \lambda_\phi^T + \nu^T \lambda_\Omega^T + \eta^T \lambda_\Psi^T \tag{30}$$

which yields the Hamiltonian:

$$\mathcal{H}(\boldsymbol{\Lambda}, \boldsymbol{x}, \boldsymbol{u}) = \Lambda^T \mathcal{F}(\boldsymbol{x}, \boldsymbol{u}) + \mu^T \varphi(\boldsymbol{x}, \boldsymbol{u}) \tag{31}$$

and reduces (29) to:

$$\mathrm{d}\underline{J} = \left(\frac{\partial \phi}{\partial t} + \nu^T \frac{\partial \Omega}{\partial t} + \mathcal{H}\right)_{t_f} \mathrm{d}t_f + \int_{t_0}^{t_1^-} \mathcal{H}_u \delta \boldsymbol{u}\, \mathrm{d}\tau + \int_{t_1^+}^{t_f} \mathcal{H}_u \delta \boldsymbol{u}\, \mathrm{d}\tau \tag{32}$$

This establishes the relations between variations in the independent variables, $\boldsymbol{u}$ and t_f, and variations in the cost function for the *constrained* problem, including the terminal manifold, the tangency point, and the state-dependent control constraint. Assuming bang-bang control, we use these relations to compute the variations in the switching times that would zero the differential of the augmented cost function.

2.2.6. Bang-Bang Solution

It is easy to show that the solution for minimum-time problems for systems linear in the controls consists of bang-bang controls, excluding singular arcs (Bryson and Ho, 1975; Weinreb and Bryson, 1985). By assuming bang-bang control we reduce the functional optimization to a parameter optimization over the switching times. The number of switches is approximated from the initial guess, as discussed later, and the singular arcs are approximated by a finite number of switches (Meier, 1987).

For bang-bang controls, the variations δu_i in (32) are replaced with:

$$\delta u_i = (\alpha_m - \alpha_m)\,\mathrm{sgn}(\mathrm{d}t_{ij}) \tag{33}$$

where sgn is the signum function, and $\mathrm{d}t_{ij}$ is the change of the j-th switching time for control u_i. Note that $\delta u_i \neq 0$ only at the switching times where u_i switches between the extremes. Therefore, δu_i is represented by

$$\delta u_i = (-1)^{j-1}\Delta\alpha\,\mathrm{d}t_{ij} \tag{34}$$

where

$$\Delta\alpha = \alpha_M - \alpha_m \tag{35}$$

Using (34) we now discretize the augmented cost function $\mathrm{d}\underline{J}$ of (32) as a function of the switching times:

$$\begin{aligned}\mathrm{d}\underline{J} = &\sum_{i=1}^{m}\sum_{j=1}^{s_{1,i}}(\mathcal{H}_{u_i})_{t_{ij}}\Delta\alpha\,\mathrm{d}t_{ij} + \sum_{i=1}^{m}\sum_{j=1}^{s_{2,i}}(\mathcal{H}_{u_i})_{t_{ij}}\Delta\alpha\,\mathrm{d}t_{ij}\\ &+\sum_{i=1}^{m}\sum_{j=1}^{s_{3,i}}(\mathcal{H}_{u_i})_{t_{ij}}\Delta\alpha\,\mathrm{d}t_{ij} + \left(\frac{\partial\phi}{\partial t} + \nu^T\frac{\partial\Omega}{\partial t} + \mathcal{H}\right)_{t_f}\mathrm{d}t_f\end{aligned} \tag{36}$$

where s_1 represents the segment of the trajectory before the obstacle, s_2 represents the constrained arc, s_3 is the segment of the trajectory from the obstacle to the target, and m is the dimension of $\boldsymbol{u}$. Since the second term in (36) corresponds to the constrained arc, the corrections $\mathrm{d}t_{ij}$ are computed only for the controls not determined from $S^{(p)}(\boldsymbol{x},\boldsymbol{u},t) = 0$.

The objective now is to determine the variations $\mathrm{d}t_{ij}$ that would minimize the differential $\mathrm{d}\underline{J}$. This can be done by following the negative gradient of $\mathrm{d}\underline{J}$ defined by the coefficients of the $\mathrm{d}t_{ij}$ in (36). The step size of each move is determined by adding a quadratic term in $\mathrm{d}t_{ij}$ and $\mathrm{d}t_f$ to (36) (Meier, 1987):

$$\begin{aligned}\mathrm{d}\widehat{J} = &\sum_{i=1}^{m}\sum_{j=1}^{s_{1,i}}(\mathcal{H}_{u_i})_{t_{ij}}\Delta\alpha\,\mathrm{d}t_{ij} + \frac{1}{2}w_{ii}\Delta\alpha_i^2\,\mathrm{d}t_{ij}^2\\ &+\sum_{i=1}^{m}\sum_{j=1}^{s_{2,i}}(\mathcal{H}_{u_i})_{t_{ij}}\Delta\alpha\,\mathrm{d}t_{ij} + \frac{1}{2}w_{ii}\Delta\alpha_i^2\,\mathrm{d}t_{ij}^2\end{aligned}$$

$$+\sum_{i=1}^{m}\sum_{j=1}^{s_{3,i}}(\mathcal{H}_{u_i})_{t_{ij}}\Delta\alpha\,\mathrm{d}t_{ij}+\frac{1}{2}w_{ii}\Delta\alpha_i^2\,\mathrm{d}t_{ij}^2$$

$$+\left(\frac{\partial\phi}{\partial t}+\nu^T\frac{\partial\Omega}{\partial t}+\mathcal{H}\right)_{t_f}\mathrm{d}t_f+\frac{1}{2}b(\mathrm{d}t_f)^2 \tag{37}$$

where b is a positive constant, and w_{ii} are the elements of a diagonal positive definite matrix.

. The step size that minimizes (37) is given by:

$$\mathrm{d}t_{ij}=-\frac{(\mathcal{H}_{u_i})_{t_{ij}}}{w_{ii}\Delta\alpha} \tag{38}$$

$$\mathrm{d}t_f=-\frac{1}{b}\left(\mathcal{H}+\frac{\partial\phi}{\partial t}+\nu^T\frac{\partial\Omega}{\partial t}\right)_{t_f} \tag{39}$$

The values of $\mathrm{d}t_{ij}$ and $\mathrm{d}t_f$ in eqns. (38) and (39) depend on the multipliers η and ν which are computed by back-substituting (38) and (39) in (22) and (26), and by multiplying $\mathrm{d}\Psi_1(t_1)$ and $\mathrm{d}\Omega(t_f)$ by $-\epsilon$, with ϵ a being small positive constant. This scales the improvements in η and ν to satisfy the first-order necessary conditions of optimality.

With this substitution, eqns. (22) and (26) yield:

$$\eta=-I_{\Psi\Psi}^{-1}(-\epsilon\,\mathrm{d}\Psi_{t_1}+I_{\Psi\Omega}\nu+I_{\Psi\phi})$$

$$\nu=-\left({}^1I_{\Omega\Omega}+{}^1I_{\Omega\Psi}{}^1I_{\Psi\Psi}^{-1}{}^1I_{\Psi\Omega}+{}^2I_{\Omega\Omega}+{}^3I_{\Omega\Omega}+\frac{1}{b}\left(\frac{\mathrm{d}\Omega}{\mathrm{d}t}\frac{\mathrm{d}\Omega}{\mathrm{d}t}^T\right)_{t_f}\right)^{-1}$$

$$\times\left(-\epsilon\,\mathrm{d}\Omega_{t_f}+{}^1I_{\Omega\phi}-{}^1I_{\Omega\Psi}{}^1I_{\Psi\Psi}^{-1}\left(-\epsilon\,\mathrm{d}\Psi_{t_1}+{}^1I_{\Psi\phi}\right)\right.$$

$$\left.+{}^2I_{\Omega\phi}+{}^3I_{\Omega\phi}+\frac{1}{b}\left(\frac{\mathrm{d}\Omega}{\mathrm{d}t}\frac{\mathrm{d}\phi}{\mathrm{d}t}\right)_{t_f}\right) \tag{40}$$

where the terms ${}^lI_{hk}$ are defined as:

$${}^lI_{hk}=\sum_{i=1}^{m}\sum_{j=1}^{s_{l,i}}\left(\lambda_h^T\frac{\partial\mathcal{F}}{\partial u_i}w_{ii}^{-1}\frac{\partial\mathcal{F}}{\partial u_i}^T\lambda_k\right)_{t_{ij}} \tag{41}$$

with $h=\Psi_1,\Omega$, $k=\Psi_1,\Omega,\phi$, $l=1,2,3$, representing *before*, *on* and *after* the state constraint, and i indicating the independent controls.

This procedure reduces the differential defined in (32) to zero, which also satisfies the necessary conditions of optimality stated by the Pontryagin Minimum Principle, as discussed in (Fiorini, 1995).

3. Initial Guess

The dynamic optimization discussed earlier converges only to a local minimum, which depends on the initial guess. Since the dynamic motion planning problem is generally not convex, i.e. it has multiple local minima, selecting the appropriate initial guess would determine the quality of the solution. While it is generally desirable to compute the global minimal trajectory, it is equally important to obtain a trajectory specified in terms of the sequence of avoidance and the side from which each obstacle is being avoided. Selecting an initial guess in dynamic environments is in itself a dynamic motion planning problem, as discussed earlier in the Introduction. Imposing a desired structure makes the problem only harder.

An efficient method for solving both problems has been recently developed (Fiorini and Shiller, 1995). It generates trajectories that are both collision-free and dynamically feasible. Below, we first briefly summarize this approach, and then compute a bang-bang approximation for the controls.

3.1. Generating the Trajectory

The method for generating feasible trajectories in dynamic environments is based on the concept of velocity obstacles, which is a first-order approximation of the robot velocities that would cause a collision with some obstacle at some future time (Fiorini, 1995; Fiorini and Shiller, 1995). Collision is avoided by selecting velocities outside the union of the velocity obstacles due to all moving and static obstacles.

To ensure that the selected maneuver is also dynamically feasible, we impose additional velocity constraints due to robot dynamics and actuator constraints, as shown in Fig. 1. Figure 1 shows the velocity obstacle of $\widehat{B}$, moving at some velocity v_B, with respect to a point robot, $\widehat{A}$. Also shown are the feasible velocities RAV, which for a planar robot are represented by a parallelogram. The feasible avoidance velocities are confined to the set defined by the difference between the feasible avoidance velocities and the velocity obstacle.

An avoidance maneuver consists of a velocity vector and a time interval over which that velocity is applied. Maneuvers can be selected to minimize a global cost function, such as motion time, or to satisfy local objectives, such as passing an obstacle from the front rather than from the rear.

A trajectory consists of a sequence of avoidance maneuvers. A trajectory that minimizes motion time can be generated by searching over a tree of feasible avoidance maneuvers, generated at discrete time intervals. Figure 2 shows two branches of the tree, rooted in node n_i at time i and reaching nodes n_{i+1}. The feasible avoidance velocities at times i and $i+1$ are represented by RAV^i and RAV^{i+1}. A trajectory generated by this search is a good initial guess for the dynamic optimization, since it is quasi optimal, and it has the desired topological properties (i.e. a sequence of avoidance and type of maneuvers). A drawback of this trajectory is that its velocity profile is discontinuous, and hence cannot be differentiated to compute the nominal controls. This is resolved by first smoothing the trajectory using Hermite splines, as discussed next.

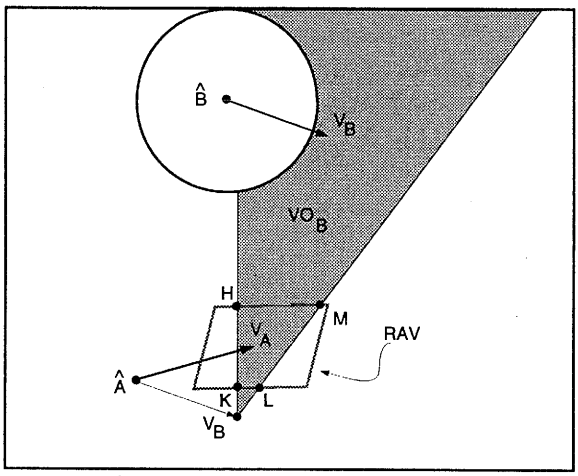

Fig. 1. The feasible avoidance velocities RAV.

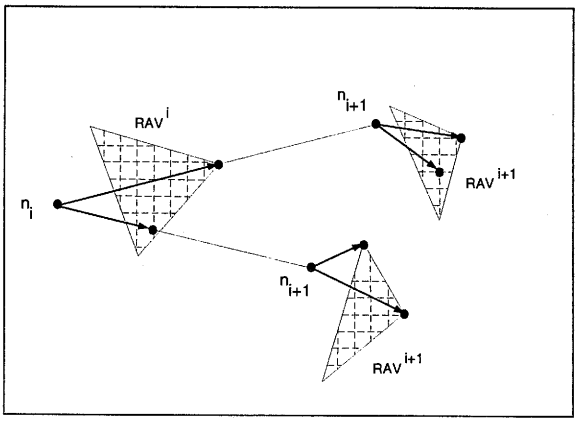

Fig. 2. Tree representation for the global search.

3.2. Generating the Controls

To compute the controls, we first smooth the trajectory, consisting of a sequence of avoidance maneuvers, using a spline interpolation. First, the path is smoothed by joining the mid-points of every consecutive path segment with a third order Hermite spline that matches the slopes of the path segments (Foley *et al.*, 1990). Then, the velocity profile along the resulting path is smoothed using a cycloid between the mid-points of consecutive velocity segments, given by:

$$v(t) = \frac{\omega t - \sin(\omega t)}{2\pi} \tag{42}$$

where $\omega = 2\pi/T$, and T is the motion time between the two mid-points.

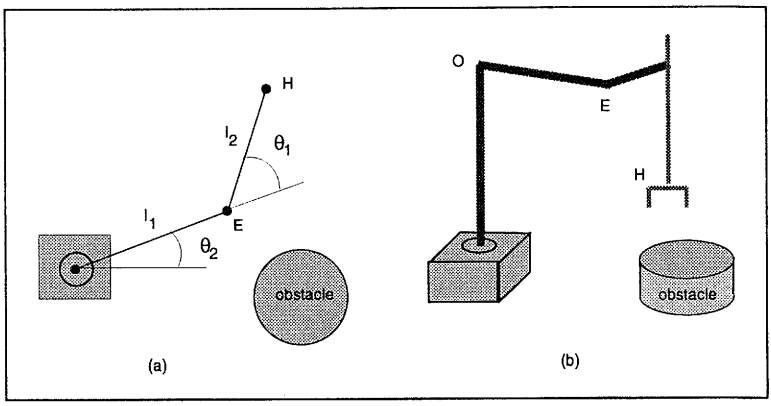

Fig. 3. Planar 2-DOF manipulator: (a) top view, (b) side view.

Using inverse dynamics, we now compute the controls associated with the smoothed trajectory. The resulting actuator efforts are approximated by bang-bang controls by choosing the switching times at the zero crossings of the smooth controls, with a dead-band to avoid chatter (Fiorini, 1995; Fiorini and Shiller, 1995).

4. Examples

Here we present examples for the two degree-of-freedom planar manipulator shown in Fig. 3. The problem is greatly simplified by assuming a planar SCARA manipulator with two uniform links and with only the end-effector reaching the plane of the obstacles. Then, only the end-effector trajectory among the obstacles needs to be computed. The parameters of the arm are: $l_1 = 1.5\,\mathrm{m}$, $l_2 = 1.3\,\mathrm{m}$, $m_1 = 10.0\,\mathrm{Kg}$, $m_2 = 10.0\,\mathrm{Kg}$, $\tau_1 = 10.0\,\mathrm{Nm}$, $\tau_2 = 3.0\,\mathrm{Nm}$.

4.1. Single Obstacle

The objective in the following examples is to move the end-effector from rest at the starting position $\boldsymbol{x} = (-.15\ \mathrm{m},\ .55\ \mathrm{m})$, to rest at the goal position $\boldsymbol{x} = (1.5\,\mathrm{m},\ -.5\,\mathrm{m})$, in minimum time.

First, the optimal path, computed with no obstacles, is shown in Fig. 4. The actuator torques for this solution are shown in Fig. 5. For this case, the second joint has one switch, whereas the first joint has two switches and a possible singular arc (multiple switches) near the start point. This singular arc may be explained by the smaller angular rotation of the first joint compared with the rotation of the second joint. This solution closely satisfies the necessary conditions of optimality, and is similar to the solution computed by the parameter optimization presented in (Shiller and Dubowsky, 1989). The optimal time for this case is 3.59 s.

The second case considers a static obstacle, represented by a circle of radius $r = .4$ m, centered at $\boldsymbol{C} = (.6\ \mathrm{m},\ -.2\ \mathrm{m})$. The constraints Ψ_1 and Ψ_2 due to this

obstacle are:

$$\Psi_1 : \begin{pmatrix} (x-x_o)^2 + (y-y_o)^2 - r^2 = 0 \\ (x-x_o)v_x + (y-y_o)v_y = 0 \end{pmatrix}, \qquad t = t_1$$

$$\Psi_2 : \; v_x^2 + (x-x_o)a_x + v_y^2 + (y-y_o)a_y = 0, \quad t_1 \le t \le t_2$$

The optimal path for this case is shown in Fig. 6, and the actuator torques are shown in Fig. 7. Here the path grazes the obstacle at one point, but does not follow the obstacle because of its high curvature. The optimal time for this case is 5.17 s.

This case was repeated with a larger obstacle, as shown in Fig. 8, where the path follows the obstacle boundary. Here, the obstacle is of radius $r = .6$ m, located at $C = (.8 \text{ m}, \; -.15 \text{ m})$. The optimal time for this case is 5.38 s, and the controls are shown in Fig. 9.

Finally, the third case considers a moving obstacle, as shown in Fig. 10. The constraints Ψ_1 and Ψ_2 are now:

$$\Psi_1 : \begin{pmatrix} (x-(v_{ox}t+x_o))^2 + (y-(v_{oy}t+y_o))^2 - r^2 = 0 \\ (x-(v_{ox}t+x_o))(v_x - v_{ox}) \\ \quad + (y-(v_{oy}t+y_o))(v_y - v_{oy}) = 0 \end{pmatrix}, \quad t = t_1$$

$$\Psi_2 : \Big((v_x - v_{ox})^2 + (x-(v_{ox}t+x_o)\Big)a_x$$
$$+ \Big((v_y - v_{oy})^2 + (y-(v_{oy}t+y_o)\Big)a_y = 0, \qquad t_1 \le t \le t_2$$

The optimal path for this case, shown in Fig. 10, slides along the moving obstacle. The actuator torques for this case are shown in Fig. 11. The motion time for this case is $t = 4.36$ s, which is longer than the unconstrained time, but shorter than the time with a fixed obstacle.

4.2. Multiple Obstacles

In this example, the optimal trajectory is computed for two moving obstacles, using the SCARA manipulator as in the previous examples. The obstacles are moving at constant velocities: Obstacle 1 at $(.045, .045)$ m/s and obstacle 2 at $(-.007, -.03)$ m/s, starting at time t_0 from the positions $(.1, -.5)$ m and $(1.15, .7)$ m, respectively. The end-effector starts at rest from $(.3, .2)$ m, and ends at rest at $(1.5, -.5)$ m.

The initial guess for this case is shown in Fig. 12, with the motion time of 4.81 s. The bang-bang controls approximated for this trajectory are shown in Fig. 13. Optimizing from this initial guess resulted in the path shown in Fig. 12, and the actuator torques shown in Fig. 14. The optimal motion time for this case is 2.6 s.

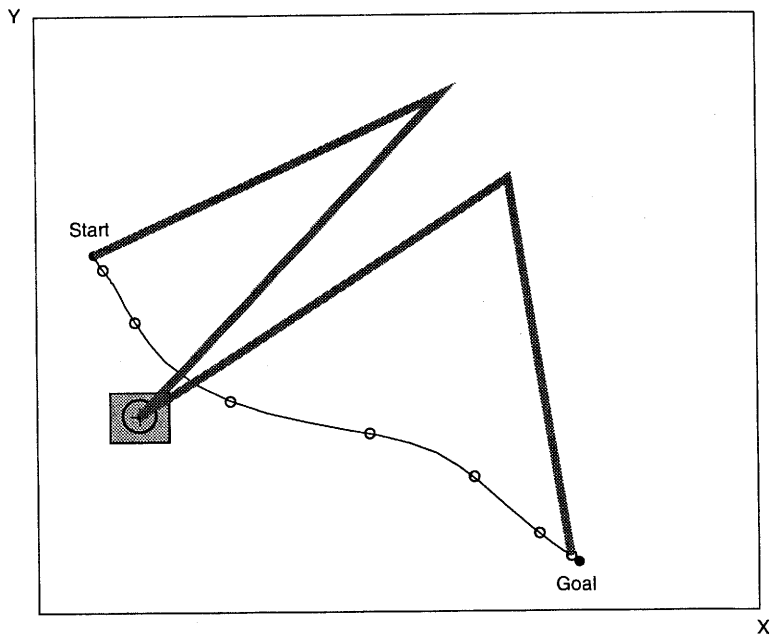

Fig. 4. Optimal trajectory in the free environment.

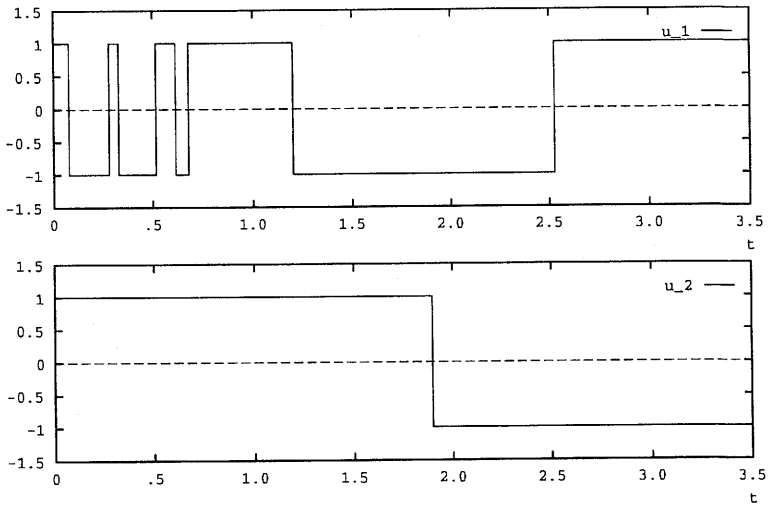

Fig. 5. Optimal controls in the free environment.

The improvement in motion time of the optimal trajectory compared to the initial guess is due to the fact that avoiding the velocity obstacles produces conservative tra-

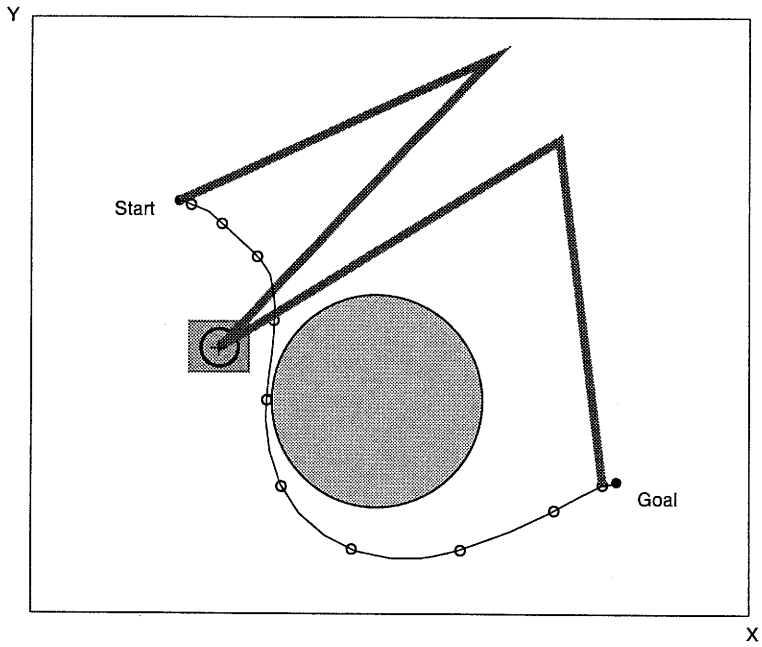

Fig. 6. Optimal trajectory with a static obstacle.

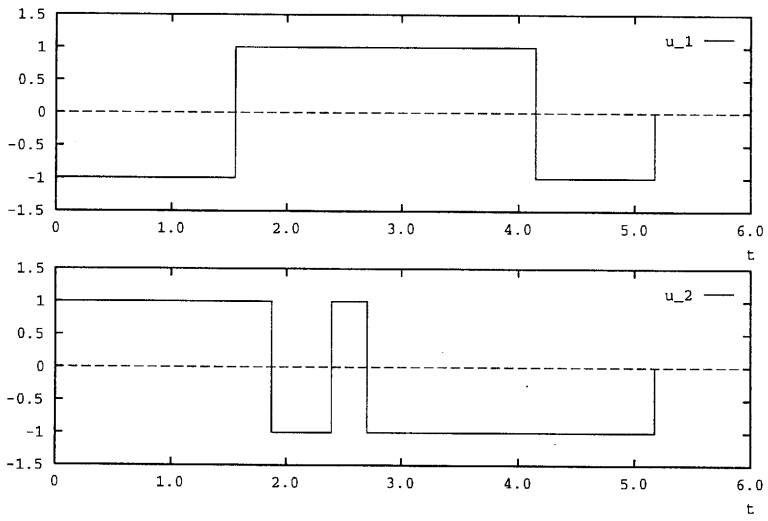

Fig. 7. Optimal controls with a static obstacle.

jectories, i.e. trajectories consisting of velocity segments that are guaranteed to avoid both obstacles at *all* times (Fiorini and Shiller, 1993). For this reason, the initial guess passes both obstacles from behind. The optimal trajectory, on the other hand, passes

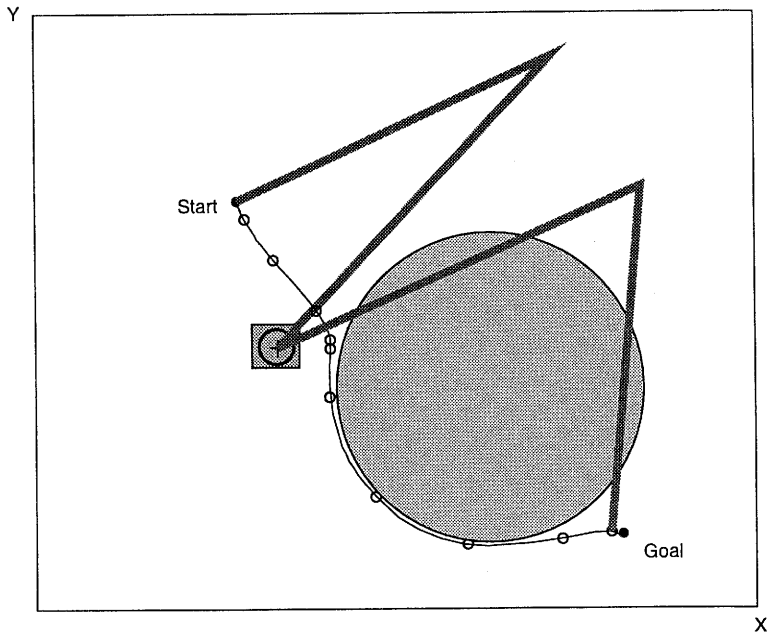

Fig. 8. Optimal trajectory with a large static obstacle.

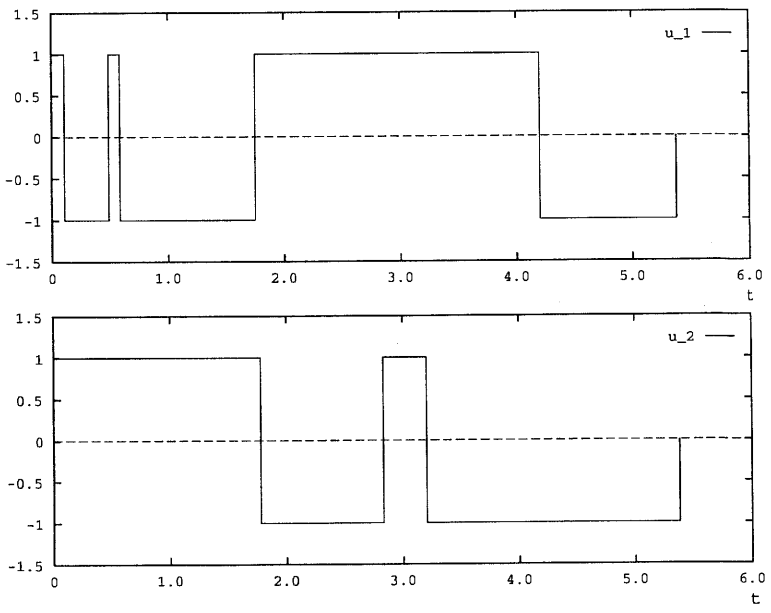

Fig. 9. Optimal controls for a large static obstacle.

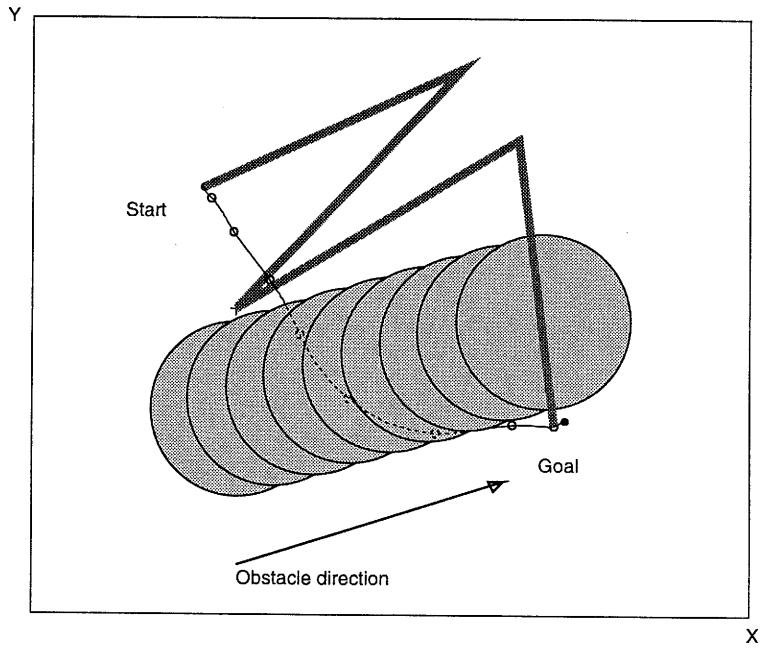

Fig. 10. Optimal trajectory with a moving obstacle.

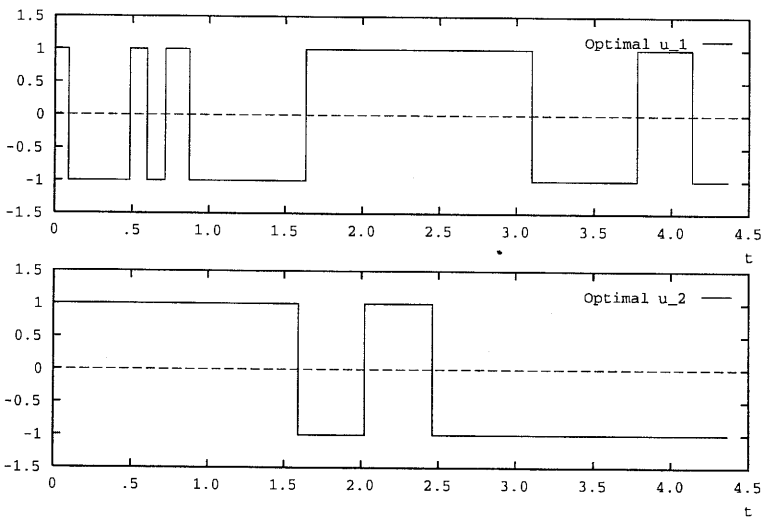

Fig. 11. Optimal controls with a moving obstacle.

both obstacles from the front, which explains the significant reduction in motion time.

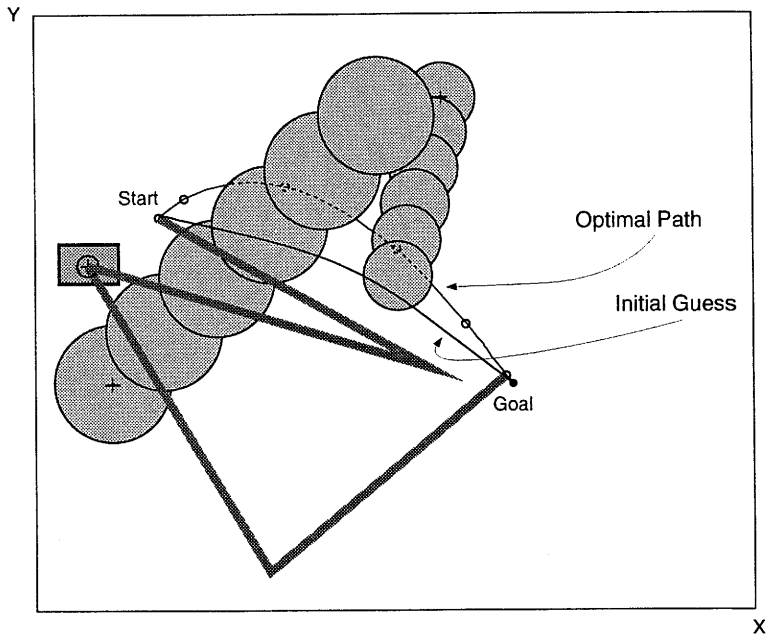

Fig. 12. Optimal trajectory with two moving obstacles.

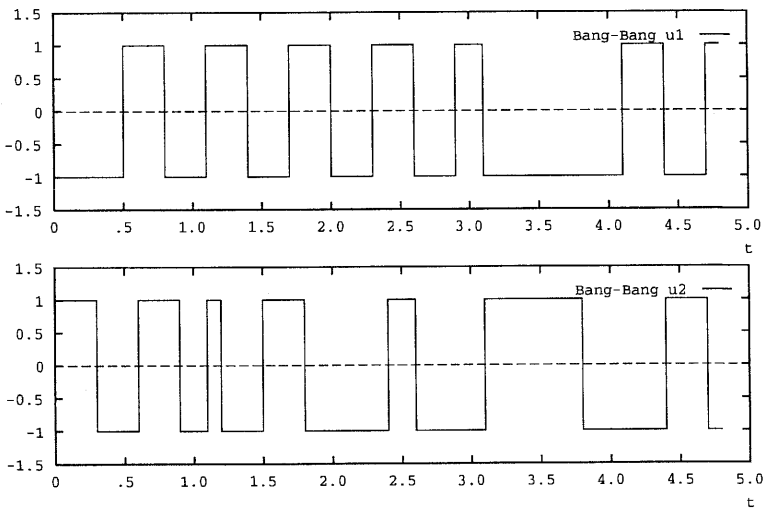

Fig. 13. Bang-bang controls for the initial guess shown in Fig. 12.

5. Summary

This paper presented a method for computing the time-optimal trajectories of a manipulator moving in dynamic environments, subject to system dynamics and actuator

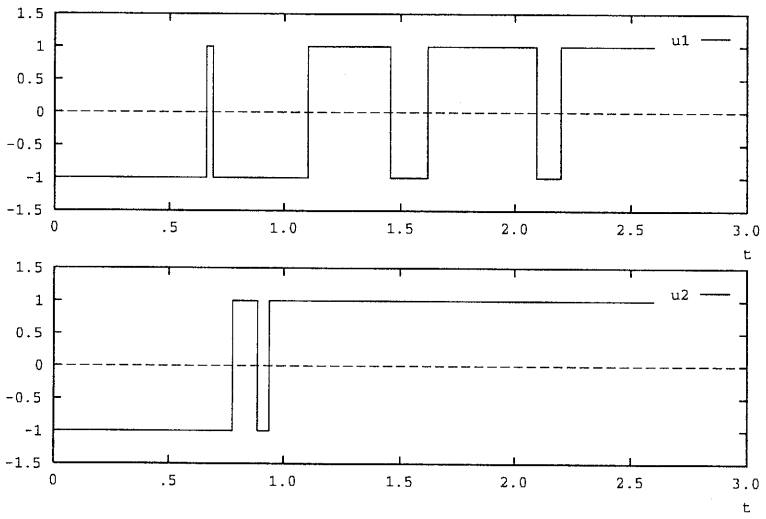

Fig. 14. Bang-bang controls for the optimal path shown in Fig. 12.

constraints. While formulating the problem as a time-minimization, the state inequality constraints due to the moving obstacles are transformed to state-dependent control constraints and a tangency point constraint at the entry point of the constrained arc. Assuming bang-bang controls, this optimization problem is solved numerically as a parameter optimization over the switching times and the final time, using a steepest descent algorithm. The initial guess for the optimization is computed using the previously developed concept of the *Velocity Obstacle* (Fiorini, 1995). The velocity obstacles allow one to select an initial guess that has a desirable structure, i.e. a desirable sequence of avoidance and a desirable side from which each obstacle should be avoided. The method is demonstrated in several examples for a 2-DOF planar manipulator moving amongst static and moving circular obstacles.

This method is meant for off-line computations, and is thus applicable to repetitive tasks, such as manipulators operating between moving conveyor belts, or manipulators operating off moving platforms. A more efficient method for on-line planning (with no guarantee of optimality) in dynamic environments has been presented in (Fiorini and Shiller, 1995).

Acknowledgment

The research described in this paper has been partially carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

Appendices

A. Derivation of the Terminal Differential

The differential of the terminal constraint Ω can be computed using (Bryson and Denham, 1962):

$$(\mathrm{d}\Omega)_{t_f} = \left(\frac{\partial \Omega}{\partial \boldsymbol{x}}\mathrm{d}\boldsymbol{x} + \frac{\partial \Omega}{\partial t}\mathrm{d}t\right)_{t_f} \tag{43}$$

Using $\mathrm{d}\boldsymbol{x} = \delta\boldsymbol{x} + \dot{\boldsymbol{x}}\mathrm{d}t$ it follows that

$$(d\Omega)_{t_f} = (\delta\Omega)_{t_f} + \dot{\Omega}_{t_f}\mathrm{d}t_f \tag{44}$$

The variation $\delta\boldsymbol{x}$ satisfies the first-order perturbation equation:

$$\delta\dot{\boldsymbol{x}} = \frac{\partial \mathcal{F}}{\partial \boldsymbol{x}}\delta\boldsymbol{x} + \frac{\partial \mathcal{F}}{\partial \boldsymbol{u}}\delta\boldsymbol{u} \tag{45}$$

Therefore, there exists a state transition matrix $\Phi(t,\tau)$ expressing the variation $(\delta\boldsymbol{x})_{t_f}$ (Bryson and Ho, 1975). The variation $\delta\Omega$ is then:

$$(\delta\Omega)_{t_f} = \left.\frac{\partial \Omega}{\partial \boldsymbol{x}}\right|_{t_f}\left(\Phi(t_f,t_0)\delta\boldsymbol{x}(t_0) + \int_{t_0}^{t_f}\Phi(t_f,\tau)\frac{\partial \mathcal{F}}{\partial \boldsymbol{u}}\delta\boldsymbol{u}(\tau)\,\mathrm{d}\tau\right) \tag{46}$$

This expression can be simplified by defining a multiplier $\lambda_\Omega \in \mathbb{R}^n \times \mathbb{R}^l$ as:

$$\lambda_\Omega^T(t) = \left(\frac{\partial \Omega}{\partial \boldsymbol{x}}\right)_{t_f}\Phi(t_f,t) \tag{47}$$

where n is the dimension of $\boldsymbol{x}$ and l is the number of terminal constraints. Taking advantage of the properties of the state transition matrix Φ (Bryson and Ho, 1975), a set of adjoint equations for λ_Ω can be written as:

$$\dot{\lambda}_\Omega(t) = -\left(\frac{\partial \mathcal{F}}{\partial \boldsymbol{x}}\right)^T\lambda_\Omega(t) \tag{48}$$

$$\lambda_\Omega(t_f) = \left(\frac{\partial \Omega}{\partial \boldsymbol{x}}\right)_{t_f} \tag{49}$$

Therefore, using

$$(\delta\Omega)_{t_f} = \int_{t_0}^{t_f}\lambda_\Omega^T\frac{\partial \mathcal{F}}{\partial \boldsymbol{u}}\delta\boldsymbol{u}\,\mathrm{d}\tau + \delta\Omega_0 \tag{50}$$

in (44), and assuming fixed initial conditions, the total differential of Ω becomes:

$$\mathrm{d}\Omega(t_f) = \int_{t_0}^{t_f}\lambda_\Omega^T\frac{\partial \mathcal{F}}{\partial \boldsymbol{u}}\delta\boldsymbol{u}\,\mathrm{d}\tau + \left(\frac{\partial \Omega}{\partial \boldsymbol{x}}\dot{\boldsymbol{x}} + \frac{\partial \Omega}{\partial t}\right)_{t_f}\mathrm{d}t_f \tag{51}$$

B. Effect of the Point Constraint on the Multipliers

The co-state equations for λ_{phi}, λ_Ω used in the previous sections do not take into account the effects of the constraints Ψ_1, given by:

$$\mathbf{\Lambda}^T_{t_1^-} = \mathbf{\Lambda}^T_{t_1^+} + \eta^T \left.\frac{\partial \Psi_1}{\partial \boldsymbol{x}(t)}\right|_{t_1} \tag{52}$$

This discontinuity affects the co-state equations, as illustrated in the following using multipliers λ_Ω (Denham and Bryson, 1964).

The unknown η is computed by relating the value of λ_Ω at t_1^-, i.e. just before reaching the constraint Ψ_1, to its value at t_1^+, i.e. just after reaching Ψ_1. To do this, $\lambda_\Omega(t_1^+)$ and $\lambda_\Omega(t_1^-)$ are first computed independently, using the expressions for $\mathrm{d}\Omega$ at t_f and t_1.

The value of $\lambda_\Omega(t_1^+)$ is computed from the expression of the changes in $\mathrm{d}\Omega(t_f)$ due to the variation of $\boldsymbol{x}$, $\delta\boldsymbol{x}(t_1^+)$:

$$\mathrm{d}\Omega(t_f) = \lambda^T_\Omega \delta\boldsymbol{x}(t_1^+) = \lambda^T_\Omega \left(\mathrm{d}\boldsymbol{x} - \dot{\boldsymbol{x}}\,\mathrm{d}t_1\right)_{t_1^+} \tag{53}$$

that can be rewritten as:

$$\mathrm{d}\Omega(t_f) = \left(\frac{\partial \Omega}{\partial \boldsymbol{x}}\mathrm{d}\boldsymbol{x} + \frac{\partial \Omega}{\partial t}\mathrm{d}t\right)_{t_1^+} \tag{54}$$

from which:

$$\left.\frac{\partial \Omega}{\partial \boldsymbol{x}}\right|_{t_1} = \lambda^T_\Omega(t_1^+) \tag{55}$$

$$\left.\frac{\partial \Omega}{\partial t_1}\right|_{t_1} = -\lambda^T_\Omega(t_1^+)\dot{\boldsymbol{x}}(t_1^+) \tag{56}$$

The value of $\mathrm{d}\Omega$ at t_1^- is computed using:

$$\mathrm{d}\Omega(t_1^-) = (\delta\Omega)_{t_1^-} + \dot{\Omega}_{t_1^-}\mathrm{d}t_1^- \tag{57}$$

where:

$$(\delta\Omega)_{t_1^-} = (\lambda_\Omega \delta\boldsymbol{x})_{t_0} + \int_{t_0}^{t_1^-} \lambda^T_\Omega \frac{\partial \mathcal{F}}{\partial \boldsymbol{u}}\delta\boldsymbol{u}\,\mathrm{d}\tau$$

Since $\mathrm{d}S^{(p-1)}(\boldsymbol{x}) = 0$, the value of $\mathrm{d}t_1$ is:

$$\mathrm{d}t_1 = \frac{1}{\dot{S}^{(p-1)}}\left[-\left(\lambda^T_{S^{(p-1)}}\delta\boldsymbol{x}\right)_{t_0} - \int_{t_0}^{t_1} \lambda^T_{S^{(p-1)}}\frac{\partial \mathcal{F}}{\partial \boldsymbol{u}}\delta\boldsymbol{u}\,\mathrm{d}t\right] \tag{58}$$

By replacing $\mathrm{d}t_1$ in $\mathrm{d}\Omega$ of (57) with (58), and since $\dot{S}^{(p-1)}$ and $\dot{\Omega}$ are both independent of the integration variable, $\mathrm{d}\Omega(t_1^-)$ becomes:

$$\mathrm{d}\Omega(t_1^-) = \left(\lambda_\Omega^T \delta \boldsymbol{x}\right)_{t_0} + \int_{t_0}^{t_1^-} \left(\lambda_\Omega^T - \frac{\dot{\Omega}}{\dot{S}^{(p-1)}} \lambda_{S^{(p-1)}}^T\right) \frac{\partial \mathcal{F}}{\partial \boldsymbol{u}} \delta \boldsymbol{u} \, \mathrm{d}\tau$$

$$- \left.\frac{\dot{\Omega}}{\dot{S}^{(p-1)}}\right|_{t_1^-} \left(\lambda_{S^{(p-1)}}^T \delta \boldsymbol{x}\right)_{t_0} \tag{59}$$

The desired expression of λ_Ω at t_1^-, satisfying $\mathrm{d}S^{(p-1)} = 0$ is then:

$$\lambda_{\Omega, S^{(p-1)}}^T(t_1^-) = \left(\lambda_\Omega^T - \frac{\dot{\Omega}}{\dot{S}^{(p-1)}} \lambda_{S^{(p-1)}}^T\right)_{t_1^-} \tag{60}$$

This equation can be further simplified by replacing $\dot{\Omega}(t_{-1})$ with:

$$\dot{\Omega}(t_1^-) = \left.\frac{\partial \Omega}{\partial \boldsymbol{x}}\right|_{t_1^-} \dot{\boldsymbol{x}}(t_1^-) - \lambda_\Omega^T(t_1^+) \dot{\boldsymbol{x}}(t_1^+) \tag{61}$$

Since the differentials $\mathrm{d}\boldsymbol{x}(t_1)$ and $\mathrm{d}t_1$ are the same at t_1^- and t_1^+, eqn. (55) gives:

$$\left.\frac{\partial \Omega}{\partial \boldsymbol{x}}\right|_{t_1^-} = \lambda_\Omega^T(t_1^+) \tag{62}$$

and similarly

$$\lambda_{S^{(p-1)}}^T(t_1^-) = \left.\frac{\partial S^{(p-1)}}{\partial \boldsymbol{x}}\right|_{t_1} \tag{63}$$

By using eqns. (62), (63), and (61) in (60), the discontinuity of λ_Ω at t_1 becomes:

$$\lambda_{\Omega, S^{(p-1)}}^T(t_1^-) = \lambda_\Omega^T(t_1^+) \left(I - \frac{\dot{\boldsymbol{x}}(t_1^-) - \dot{\boldsymbol{x}}(t_1^+)}{S^{(p)}(t_1)} \left.\frac{\partial S^{(p-1)}}{\partial \boldsymbol{x}}\right|_{t_1}\right) \tag{64}$$

which is equivalent to the necessary condition (52) if the multiplier η is equal to:

$$\eta^T = -\lambda_\Omega^T(t_1^+) \left(\frac{\dot{\boldsymbol{x}}(t_1^-) - \dot{\boldsymbol{x}}(t_1^+)}{S^{(p)}(t_1^-)}\right) \tag{65}$$

References

Bobrow J.E. (1988): *Optimal robot path planning using the minimum time criterion.* — IEEE Trans. Robot. Automat., Vol.4, No.4, pp.443–450.

Bryson A.E. (1992): *Dynamic Optimization.* — Palo Alto, CA: Stanford University.

Bryson A.E. (1995): *Inverse Dynamic Optimization.* — Los Angeles, CA: MANE Seminar, UCLA.

Bryson A.E., Denham S.E. and Dreyfus W.F. (1963): *Optimal programming problems with inequality constraints i : Necessary conditions for extremal solutions.* — AIAA Journal, Vol.1, No.11, pp.2544–2550.

Bryson A.E. and Denham S.E. (1962): *A steepest-ascent method for solving, optimum programming problems.* — ASME J. Appl. Mech., Vol.29, No.2, pp.247–257.

Bryson A.E. and Ho Y.C. (1975): *Applied Optimal Control.* — New York, NY: Hemisphere Publishing Corp.

Denham W.F. and Bryson A.E. (1964): *Optimal programming problems with inequality constraints ii : Solution by steepest ascent.* — AIAA Journal, Vol.2, No.2, pp.25–34.

Erdmann M. and Lozano-Perez T. (1987): *On multiple moving objects.* — Algorithmica, Vol.2, No.4, pp.477–521.

Fiorini P. (1995): *Robot Motion Planning among Moving Obstacles.* — Ph.D. Thesis, University of California, Los Angeles.

Fiorini P. and Shiller Z. (1993): *Motion planning in dynamic environments using the relative velocity paradigm.* — Proc. IEEE Int. Conf. *Automat. Robot.*, Atlanta, GA, Vol.1, pp.560–566.

Fiorini P. and Shiller Z. (1995): *Robot motion planning in dynamic environments*, In: Proc. Int. Symp. *Robotic Res.* (G. Girald and G. Hirzinger, Eds.). — Munich, Germany, 20–24 October 1995, Berlin: Springer-Verlag, pp.237–248.

Fiorini P. and Shiller Z. (1996): *Time optimal trajectory planning in dynamic environments.* — Proc. IEEE Int. Conf. *Automat. and Robotics*, Minneapolis, MN, 22–28 April, Vol.2, pp.1553–1558.

Foley J.D., van Dam A., Feiner S.K. and Hughes J.F. (1990): *Computer Graphics.* — Reading, MA: Addison-Wesley.

Fraichard T. (1993): *Dynamic trajectory planning with dynamic constraints: A state-time space approach.* — Proc. IEEE/RSJ Int. Conf. *Intelligent Robots and Systems*, Yokohama, Japan, 26–30 July 1993, pp.1393–1400.

Fraichard T. and Laugier C. (1993): *Path-velocity decomposition revisited and applied to dynamic trajectory planning.* — Proc. IEEE Int. Conf. *Automat. Robot.*, Atlanta, GA, 2–6 May 1993, Vol.1, pp.40–45.

Fujimura K. (1994): *On motion planning amid transient obstacles.* — Int. J. Robotics Res., Vol.13, No.5, pp.395–407.

Fujimura K. (1995): *Time-minimum routes in time-dependent networks.* — IEEE Trans. Robot. Automat., Vol.11, No.3, pp.343–351.

Fujimura K. and Samet H. (1989a): *A hierarchical strategy for path planning among moving obstacles.* — IEEE Trans. Robot. and Automat., Vol.5, No.1, pp.61–69.

Fujimura K. and Samet H. (1989b): *Time-minimal paths among moving obstacles.* — Proc. IEEE Int. Conf. *Robot. and Automat.*, Scottsdale, AZ, pp.1110–1115.

Fujimura K. and Samet H. (1990) *Motion planning in a dynamic domain.* — Proc. IEEE Int. Conf. *Robot. Automat.*, Cincinnati, OH, pp.324–330.

Jacobson D.H., Lele M.M. and Speyer J.L. (1971): *New necessary conditions of optimality for control problems with state-variable inequality constraints.* — J. Math. Anal. Applics., Vol.35, No.2, pp.255–284.

Johnson D.W. and Gilbert E.G. (1985): *Minimum time robot planning in the presence of obstacles.* — Proc. IEEE Conf. *Decision and Control*, Ft. Lauderdale, FL, 1748–1753.

Kahn M.E. and Roth B. (1971): *The near-minimum time control of open loop kinematic chains.* — ASME T., DCMS, Vol.93, No.3, pp.164–172.

Kant K. and Zucker S.W. (1986): *Towards efficient trajectory planning: The path-velocity decomposition.* — Int. J. Robotic Res., Vol.5, No.3, pp.72–89.

Latombe J.C. (Ed.) (1991): *Robot Motion Planning.* — Boston, MA: Kluwer Academic.

Laumond J.P. (1987): *Feasible trajectories for mobile robots with kinematic and environment constraints*, In: Intelligent Autonomous Systems (L.O. Hertzberger and F.C.A. Groen, Eds.). — New York, NY: North Holland, pp.346–354.

Laumond J.P., Jacobs P.E., Taix M. and Murray R.M. (1994): *Motion planner for non-holonomic mobile robots.* — IEEE Trans. Robot. Automat., Vol.RA-10, No.5, pp.573–593.

Lee B.H. and Lee C.S.G. *Collision-free motion planning of two robots.* — IEEE Trans. Syst. Man Cybern., Vol.SMC-17, No.1, pp.21–32.

Lozano-Pérez T. and Wesley M. A. (1979): *An algorithm for planning collision-free paths among polyhedral obstacles.* — Comm. ACM, Vol.22, No.10, pp.560–570.

Meier E.B. (1987): *An Efficient Algorithm for Bang-Bang Control Systems Applied to a Two-Link Manipulator.* — Ph.D. Thesis, Stanford.

Murray R.M. and Sastry S.S. (1993): *Nonholonomic motion planning: Steering with sinusoids.* — IEEE Trans. Automat. Contr., Vol.AC-38, No.5, pp.700–716.

Reif J. and Sharir M. (1985): *Motion planning in the presence of moving obstacles.* — Proc. 26th IEEE Symp. *Foundations of Computer Science*, Portland, Oregon, pp.144–153.

Seywald H. (1994): *Trajectory optimization based on differential inclusion.* — AIAA J. Guidance, Control and Dynamics, Vol.17, No.3, pp.480–487.

Shiller Z. (1996): *Time-energy optimal control of articulated systems with geometric path constraints.* — ASME J. Dyn. Systems, Meas. and Contr., Vol.118, No.1, pp.139–143.

Shiller Z. and Dubowsky S. (1989): *Robot path planning with obstacles, actuators, gripper and payload constraints.* — Int. J. Robotics Res., Vol.8, No.6, pp.3–18.

Shiller Z. and Fiorini P. (1995): *Autonomous negotiation of freeway traffic.* — SPIE Conf. *Collision Avoidance and Traffic Management Sensors*, Philadelphia, PA, Vol.2592, pp.83–92.

Shiller Z. and Gwo R.Y. (1991): *Dynamic motion planning of autonomous vehicles.* — IEEE Trans. Robot. Automat., Vol.7, No.2, pp.241–249.

Shiller Z. and Sundar S. (1996): *Emergency maneuvers of autonomous vehicles.* — Proc. 13th World Congress, IFAC'96, San Francisco, CA, Vol.Q, pp.393–398.

Speyer J. and Bryson A.E. (1968): *Optimal programming problems with a bounded state Space.* — AIAA Journal, Vol.6, No.8, pp.1488–1491.

Weinreb A. and Bryson A.E. (1985): *Optimal control of systems with hard control bounds.* — IEEE Trans. Automat. Contr., Vol.AC-30, No.11, pp.1135–1138.