OBSTACLE AVOIDANCE FOR MOBILE ROBOTS USING SWITCHING SURFACE OPTIMIZATION

Mauro Boccadoro * **Magnus Egerstedt** ** **Yorai Wardi** **

* `boccadoro@diei.unipg.it`
Dipartimento di Ingegneria Elettronica e dell'Informazione
Università di Perugia
06125, Perugia – Italy
** {`magnus,ywardi`}`@ece.gatech.edu`
Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, GA 30332

Abstract: This paper studies the problem of letting an autonomous mobile robot negotiate obstacles in an optimal manner. In particular, a multi-modal control problem is addressed, where different modes of operation control the robot at different locations in the state space. The specification of the optimal discrete event dynamics is pursued through the design of optimal, parametrized switching surfaces, using results on switching surface optimization. *Copyright© 2005 IFAC*

Keywords: Optimal Control, Hybrid Systems, Switching Surfaces, Gradient Descent Algorithms, Mobile Robot Navigation

1. INTRODUCTION

Consider a switched system with autonomous continuous dynamics,

$$\dot{x}(t) = f_{q(t)}(x(t)), \tag{1}$$
$$q^+(t) = s(x(t), q(t)). \tag{2}$$

where (1) describes the continuous dynamics of the state variable $x \in \mathcal{X} \subseteq \mathbb{R}^n$ and (2) describes the discrete event dynamics of the system. Considering an initial condition $x_0 := x(t_0)$, the switching law (2) determines the switching instants t_i, $i = 1, 2, \ldots$, and thus the intervals where a certain modal function is active, as well as the initial condition for the o.d.e. which defines the evolution under the next mode. The discrete variable q is piecewise constant in time and belongs to a finite or countable set Q, hence, it can be expressed in terms of the index i as $q(i)$. In terms of such index the dynamics of a switched system is:

$$\dot{x}(t) = f_i(x(t)), \quad t \in (t_{i-1}, t_i] \tag{3}$$
$$i^+ = s(x(t), i, t). \tag{4}$$

with the understanding that $f_i := f_{q(i)}$, for a given map $q(i)$, i.e., in this case (4) only expresses the occurrence of the i^{th} switch, the specification of the next active mode being given by the map $q(i)$.

Since the continuous modes are autonomous, the evolution of the system is determined by the active modes, according to (4). When the function s does not depend by the (continuous) state variable x, the switching instants are determined as exogenous inputs, and the system is controlled in open loop (timing control); when s is dependent only on the state variables, the switching law is given in a feedback form, and it may be defined by switching surfaces in the state space. To illustrate, assume the system is evolving under mode i until the switching condition $g_i(x(t)) = 0$ is satisfied, for some $t = t_i$, the i^{th} switching instant, and at the i^{th} switching state $x_i := x(t_i)$ (see figure 1).

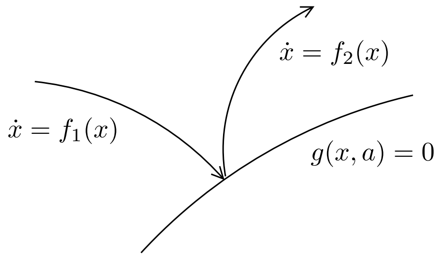


Fig. 1. Transition between discrete modes at a switching surface.

For such a system, to be able to design the switching surfaces amounts to design a feedback control law, which may be pursued for specifications of stability or optimal control. In the latter case, Refs. (Boccadoro *et al.*, 2004; Wardi *et al.*, 2004) addressed the problem of the optimal design of parametrized switching surfaces, i.e., where the generic i^{th} condition is expressed by:

$$g_i(x(t), a_i) = 0 \tag{5}$$

and determined the derivative of a given cost functional $J(x(\cdot))$ with respect to the switching parameters $a_1, a_2, \ldots$ This derivative allows for gradient descent methods to be applied resulting in locally optimal solutions.

In the literature on robot navigation, two distinctly different approaches have emerged. The first approach is generally referred to as the *reactive* approach, where the robot switches between different behaviors, or modes of operation, as changes in the environment are encountered. This approach has the general advantage in that the design process becomes modular in the sense that different modes of operation can be designed for very specific tasks. The other approach is generally referred to as the *deliberative* approach, where elaborate, optimal paths may be planned. The obvious disadvantage with such an approach is that the computational burden may be too high. On the other hand, optimality considerations can be addressed explicitly already at the design stage.

The point of view taken here is that a reactive navigation system is useful but optimality may still have an important role to play. In particular, the issue of designing optimal switching laws, given a set of behaviors, is taken in consideration. These behaviors will be given by an *approach-goal* behavior and an *avoid-obstacle* behavior, and the optimal, parametrized surface around a given obstacle has to be designed to determine *where* to switch between these two behaviors. Previously proposed such switching laws typically involve a safety distance Δ. The robot should thus switch to obstacle-avoidance when it is closer to the obstacle than Δ. However, finding the optimal Δ or even to establish if such a circular switching surface is to prefer has not been attempted, to the authors' knowledge.

This paper is organized as follows: In Section 2 is introduced the problem at hand. In Section 3 the results relative to the switching surface optimization are presented. Section 4 addresses the problem of discontinuities in the system trajectories which are encountered in the optimization procedure. In Section 5 various obstacle avoidance strategies are discussed according to the experimental results obtained.

2. PROBLEM FORMULATION

Consider a mobile robot modeled as an unicycle, which moves, in the plane, toward a target point x_g while avoiding an obstacle located at x_o. Assume such a robot is given two behaviors $Q = \{g, o\}$: mode g steers toward the goal point; mode o steers in the direction opposite to the obstacle. More in detail, the dynamics of the robot are described by:

$$\dot{x_1} = v_0 \cos\phi \tag{6}$$

$$\dot{x_2} = v_0 \sin\phi \tag{7}$$

$$\dot{\phi} = f_q(\phi, x_1, x_2) \tag{8}$$

where $q \in Q$ and

$$f_g(x, \phi) = k_g(\phi - \phi_g),$$
$$f_o(x, \phi) = -k_o(\phi - \phi_o),$$

with k_g, k_o positive constants, and $x = (x_1, x_2)$, ϕ, the state variables, denote the current coordinates in the plane, and current direction of the robot; ϕ_g, ϕ_o denote the directions from robot's current position to the target and to the obstacle, respectively. It is assumed the robot moves with a constant velocity v_0, and incurs a cost expressed by:

$$J = \frac{1}{2}\int_{t_0}^{t_g} \|x(t) - x_g\|^2 + \rho e^{-\sigma\|x(t)-x_o\|^2} dt \tag{9}$$

where t_g may be either the time instant that the robot reaches the goal or a fixed final time, and ρ, σ are given constants. Notice that the first term of the cost function penalizes trajectories getting close to the target too slowly; the second term penalizes proximity to the obstacle.

Starting with mode g from x_0 with direction ϕ_0, assuming that the respective locations of the starting point, the obstacle and goal are chosen so that the problem is not trivial (e.g. it can be assumed that x_0, x_o and x_g are aligned), the system has to be controlled by defining the "boundaries" of each behavior, which is achieved through the design of switching surfaces.

Referring to the general framework introduced in the previous Chapter, the dynamics of the system are (here x is state variable and not a planar position):

$$\dot{x}(t) = f_i(x(t)) \tag{10}$$

$$i^+(t) = \begin{cases} i & x \notin G_i(a_i) \\ i+1 & x \in G_i(a_i) \end{cases} \tag{11}$$

and the switching surfaces G_i are defined as the solution sets of the equations

$$g_i(x(t), a_i) \;=\; 0, \tag{12}$$

where $a_i \in R^h$ is the control parameter of the i^{th} switching surface, and $g_i : R^n \times R^h \to R^m$ are given functions differentiable in the x and a variables. In general, $m < n$, but typically $m = 1$; the switching surfaces are $n-1$ dimensional manifolds.

For this type of dynamics it has to be taken into account that more than one switch may occur at the same surface, and that a single surface parameter may be the argument of more than one switching surface. The "indicial" notation introduced to label continuous modes, by which f_i is a simple convention to denote $f_{q(i)}$, for a given map $q(i)$, was implicitly extended, in (11), also to the switching surfaces and their parameters, i.e., G_i represents $G_{q_g(i)}$, for a given map $q_g(i)$, which is in general different from q, and $q_g \in Q_g$, the set of labels of the switching surfaces. Also the case of different surfaces dependent on the same parameter must be considered. To this end, represent by the "vector" [1] a the set of independent parameters, and introduce the function q_a defining the argument of the i^{th} switching surface as $a_i = q_a(a, i)$ (here, the image of q_a is not a set of labels).

To illustrate, consider the switching surfaces given by two concentric circles G_r and G_R with radius r and R, $r < R$, centered on the obstacle. The inner circle determines the switch from the *approach-goal* behavior (g mode) to the *avoid-obstacle* (o mode); the outer circle the switch from mode o to mode g. Assume that the optimization parameters are the radii of the two circles with the constraint $R = 2r$, and consider an evolution starting with mode g, switching to mode o at surface G_r, then to mode g at G_R, and again to mode o at G_r (the figures in Section 5 illustrate similar situations). In this case, $Q_g = \{r,\ R\}$ and the indicial notation reads: $f_1 = f_3 = f_g,\ \ f_2 = f_4 = f_o$, and $G_1 = G_3 = G_r,\ \ G_2 = G_R$; the maps q and q_g are defined accordingly, and choosing $a = [r]$, it results, $q_a([r], 1) = q_a([r], 3) = r;\ \ q_a([r], 2) = 2r$.

[1] Since each parameter is in $\mathbb{R}^h$, a is a matrix of appropriate size

Notice that the optimal solution is constrained to the initial choice of a certain functional form for the switching surfaces (in this case, two circular shapes); this issue is going to be discussed later, at the light of the experimental results presented in Section 5.

3. ENABLING RESULTS

This Section presents the gradient formulae to be used for the optimization of the switching surfaces as derived in (Boccadoro, 2005), which hold under the following transversality assumption.

Assumption 1. For an execution [2] of (3,4) with N switches $L_i \neq 0, \forall i = 1, \ldots, N$.

where $L_i \in \mathbb{R}$ is the Lie derivative of g_i along f_i, i.e.,

$$L_i := \frac{\partial g_i}{\partial x}(x_i, a_i) f_i(x_i), \tag{13}$$

Here the vector fields f_i in (3) are Lipschitz in $\mathbb{R}^n$, and the switching surfaces are determined by continuously differentiable functions $g : \mathbb{R}^n \times \mathbb{R}^h \mapsto \mathbb{R}$, hence h is the number of parameters in the switching functions. Some shorthand notations is introduced next. Define $\Lambda_i \in \mathbb{R}^h$ as

$$\Lambda_i^T := \frac{\partial g_i}{\partial a}(x_i, a_i)\frac{1}{L_i}; \tag{14}$$

$\Gamma_i \in R^n$ as,

$$\Gamma_i^T := \frac{\partial g_i}{\partial x}(x_i, a_i)\frac{1}{L_i}, \tag{15}$$

and let $R_k := f_{k+1}(x_k) - f_k(x_k)$. The derivative of the cost function J w.r.t. the switching surface parameter a_i is given by:

$$\frac{dJ}{da_i} = -\sum_{k=i}^{N-1} p^T(t_k^+) R_k \Lambda_k^T \frac{da_k}{da_i} \tag{16}$$

where the costate variable is given by the backward o.d.e.

$$\dot{p}^T = -p^T \frac{\partial f_k}{\partial x} - \frac{\partial L}{\partial x}; \quad t \in [t_{k-1}, t_k) \quad k = 1 \ldots N \tag{17}$$

with terminal conditions

$$\begin{array}{ll} p^T(t_N) = 0 & \text{(fixed final time)} \\ p^T(t_N) = -L(x(t_N))\Gamma_N^T & \text{(terminal manifold)} \end{array} \tag{18}$$

and reset conditions:

$$p^T(t_k^-) = p^T(t_k^+)(I - R_k \Gamma_k^T) \tag{19}$$

[2] An *execution* of a switched system is defined as the trajectory followed in the continuous and discrete state space, for a given dynamic evolution of the system, which depends on the initial condition $x(t_0) := x_0$, on the switching events, and on a terminating condition.

Inspection of the gradient formula (16) shows that the set of switching states x_k is needed, in order to compute the terms R_k and Λ_k, as well as the costate, evaluated at the switching instants. Such quantities can be computed by two numerical integrations; the first, forward, of the dynamics of the system, and the second, backwards, for the costate. Notice that the boundary condition (18) for the costate, and its resets (19) also need quantities computed in the forward integration.

4. DISCONTINUOUS TRAJECTORIES

The gradient formulae presented in the previous section (16) can be exploited in a gradient descent algorithm which allows to find local optimal solutions for the switching surfaces optimization problem.

Such formulae were derived under the condition that the switching surfaces were reachable by the continuous trajectory in a transversal fashion (Assumption 1). This guarantees that infinitesimal variations in the switching surface parameters, while inducing infinitesimal variations in the continuous state space trajectory, does not change the trajectory in the discrete state space (Broucke and Arapostathis, 2002). In other words, the trajectory is deformed continuously by a change in the surface parameters, and its "qualitative" structure, i.e., the cardinality and the ordering of the active discrete modes, is invariant. Such continuity of the solutions of (3,4) implies continuity for the cost function $J(a)$, hence, a first order approximation of $J(\hat{a} + \Delta a)$ in a small neighborhood of $\hat{a}$, can be computed as

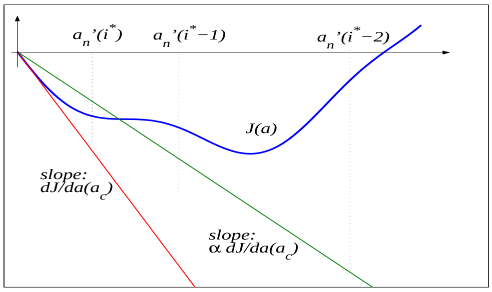
$$J(\hat{a} + \Delta a) = J(\hat{a}) + \frac{dJ}{da}(\hat{a})\Delta a + o(\Delta a), \qquad (20)$$

where $a = (a_1, a_2, \ldots)$.

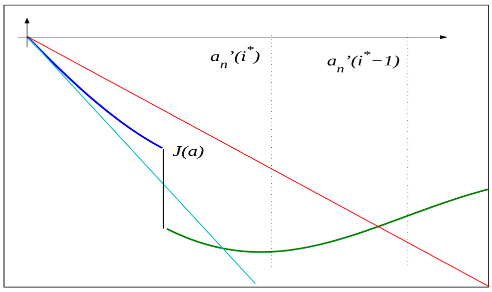
However, while proceeding in the optimization by a gradient descent algorithm, it may happen that non transversality of the trajectory and a switching surface may be approached [3], for some value $\tilde{a}$, in whose neighborhood both the system trajectory and the cost function are discontinuous. In such critical cases, taking a step in the direction of the gradient is likely to "jump over" the discontinuity, since the first order approximation (20) holds as long as Δa is sufficiently small to guarantee that the discrete trajectory remains the same, and the direction opposite to the gradient is not guaranteed to be a descent direction.

The optimization of discontinuous functions has been addressed in (Moreau and Aeyels, 2000) by introducing the constructs of the *semigradient*. As a first attempt to tackle this problem from the standpoint of numerical methods, in this paper a gradient descent algorithm with Armijo stepsize is adopted, which provides a criterion to check that a *sufficient decrease* in the objective function is obtained at each optimization step. Since by such method the *actual* decrease in cost is also guaranteed, the gradient is set, even close to critical values of parameters, as an "exploration" direction. According to this strategy, the optimization procedure follows "ordinarily" on those subsets of the parameter space yielding the same discrete trajectories; the algorithm passes from one such subset to another (and the system jumps to a qualitatively different trajectory), only if a decrease in the objective function is checked by the Armijo algorithm, presented next.

[3] Since numerical integrations of the o.d.e's (10) and numerical evaluations of the switching conditions (11) are performed, a perfect tangentiality of the trajectory and switching surfaces cannot be verified.



(a) Continuous cost function.



(b) Discontinuous cost function.

Fig. 2. Finding the stepsize of a gradient descent algorithm according to Armijo's criterion (21). In the discontinuous case a parameter value, if it satisfies the criterion, may be chosen, regardless that it lies over the discontinuity.

4.1 Gradient descent algorithm with Armijo stepsize

The general gradient descent algorithm is based on the following basic iteration: at each step the gradient dJ/da is computed for the current value for the parameters, a_c, and the parameters value for the next iteration set to:

$$a_n = a_c - \lambda \frac{dJ}{da}(a_c)$$

Here, λ determines the stepsize, in the direction of the gradient, for the current iteration. The Armijo gradient descent algorithm is characterized by a criterion for choosing an optimal stepsize, in a feedback fashion (see e.g., (Polak, 1997)). Given two scalars α, $\beta \in (0,1)$, the stepsize λ is determined, at each iteration of the optimization procedure, by an algorithm which computes the least index $i \in \{0,1,\ldots\}$ such that the following (Armijo's) condition is satisfied:

$$J(a_n(i)) < J(a_c) - \alpha\beta^i \frac{dJ}{da}(a_c), \tag{21}$$

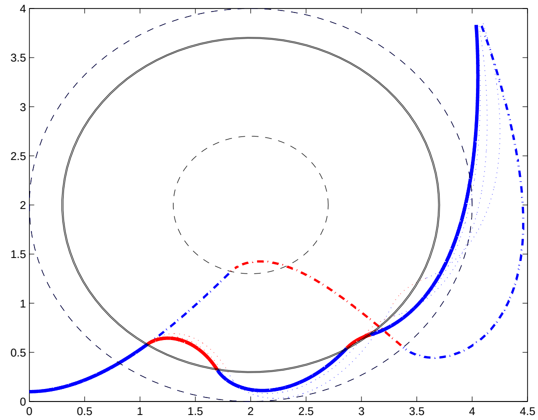
where $a_n(i) = a_c - \beta^i \frac{dJ}{da}(a_c)$. The stepsize at the current iteration is then set as $\lambda = \beta^i$ for such a minimum index i. As a rule of thumb, the parameters of the Armijo stepsize algorithm are usually set to $\alpha = \beta = 0.5$.

To illustrate Armijo's criterion, consider figure 2(a), which shows the plot of a cost functional $J(a)$, and two lines from the point $(a_c, J(a_c))$, set as the origin, one with slope equal to the gradient, and the second with slope $\alpha\frac{dJ}{da}(a_c)$. The various $a_n(i)$ and the corresponding evaluations of J are depicted, showing as $J(a_n(i))$ lies above the line passing through $(a_c, J(a_c))$ with slope $\alpha\frac{dJ}{da}$ for $i < i^*$, whereas for $i = i^*$, $J(a_n(i))$ lies below such a line.

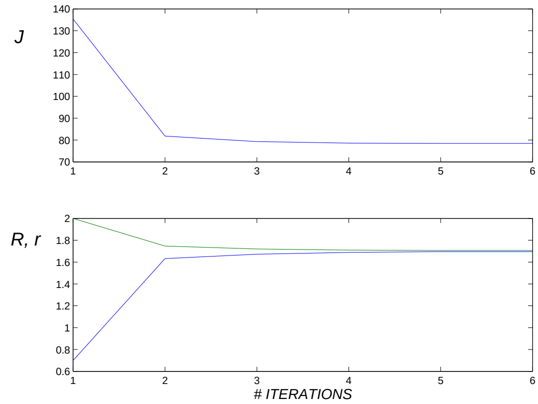
The feedback mechanism of the Armijo algorithm makes it possible to proceed without chattering along the gradient direction until convergence to some local minima, at the cost of greater computational burden, with respect to an ordinary gradient descent algorithm. Indeed, in order to check condition (21) the cost function J must be sampled each time a stepsize $\lambda(i)$ is attempted, until (21) is satisfied.

Since Armijo's algorithm checks the effective decrease in cost for the current step of optimization, this algorithm may be adopted also when the cost function J features discontinuities. To illustrate this, assume that during Armijo's algorithm the stepsize currently being checked is such to jump over a discontinuity of the cost function J. In such a case, if the region "sensed" is characterized by a lower cost, then a point sampled according to the Armijo's algorithm is likely to satisfy criterion (21), and the next value of parameter a can be changed accordingly (see figure 2(b)).

To define a terminating condition for the proposed algorithm, consider that no condition may be imposed in terms of a small norm of the gradient (typical of convex continuous problems); hence, the algorithm may be stopped when a given iteration yields no substantial change in the parameter values together with no substantial decrease in the cost function.



(a) Initial surfaces and corresponding trajectory dashed. Final surfaces and corresponding trajectory solid.



(b) Cost function and radius values at each iteration step.

Fig. 3. Optimization for two circular switching surfaces: fast robot ($v_0 = 0.7$).

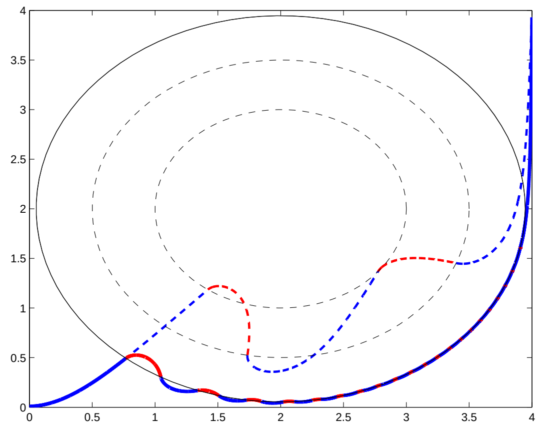


Fig. 4. Optimization for two circular switching surfaces: slow robot ($v_0 = 0.3$).

5. EXPERIMENTAL RESULTS

Some experimental results obtained for different choices of the switching surfaces structure are illustrated next. First, the optimization algorithm was run for two independent concentric circles, showing that for several choices of the system parameters v_0, ρ and α, and for different initial conditions the two radii converge to the same

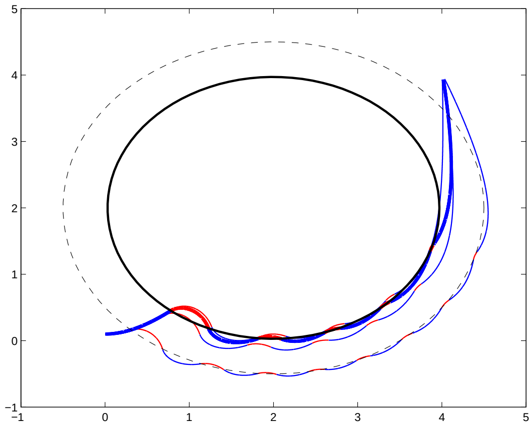


Fig. 5. Optimal navigation determined by a singular switching surface.

value. Examples of such phenomenon is shown in figures 3 and 4. This indicates that "smooth" navigation during obstacle avoidance is preferable.

According to such emerged hint, other optimization runs were carried out constraining the two radii to have a constant and very small difference. As the optimization algorithm relies on a numerical integration for the system and costate dynamics, as discussed in previous Section, the constraint $R = r + Kv_0\Delta T$ was chosen, where ΔT is the integration time step, so that the distance between the two surfaces could be covered in no less than K steps in discretized evolution of the robot, for some small integer K. Adopting two switching surfaces very close to each other may be viewed as a regularized version (see e.g. (Johansson *et al.*, 1999)) of a dynamics defined by a single switching surface. An example of single parameter optimization is given in figure 5.

Notice how in all such examples the executions undergo qualitative changes (e.g., in figure 3 the initial execution switches twice, the final execution four times) which are reflected in jumps in the cost functions.

The choice of a circular switching surface models a reactive strategy by which the robot's behavior is changed when a critical distance is reached, as mentioned above. Although such strategy presents an advantage in terms of implementation, there is no evidence for considering optimal a circular shape for the obstacle avoidance problem. Another choice for switching surfaces may be ellipses with the two semiaxis as free parameters. According to the "sliding hint" (i.e., the observation that two distinct switching surfaces tend to collapse in one), an optimization is performed for two ellipses, constrained to be close each other. The orientation of the ellipse is such that one of the semiaxis is aligned with the obstacle and the goal. More in detail $x_0 = (0, 0.1)$, $x_o = (2, 0)$, $x_g = (4, 0)$, and the switching surface given by

$$g(x, a, b) = \frac{x_1^2}{a^2} + \frac{x_2^2}{b^2} - 1 = 0.$$

A local optimal value for the ellipse parameters is $a = 1.8$, $b = 1.6$, to which correspond a trajectory with cost $J_e^* = 130.6$. For the same system, the optimality for the case of two close circular surfaces, computed by brute force, is achieved for $r = 1.97$ yielding a trajectory with cost $J_c^* = 133.2$.

6. CONCLUSIONS

In this paper, a general solution to the problem of optimizing parametrized switching surfaces with respect to the free surface parameters, is presented. The resulting solution consists of a gradient computation in combination with a numerically straightforward algorithm for obtaining the locally optimal parameters.

The proposed method is moreover applied to the problem of autonomous robot navigation, where a robot has to switch between different behaviors as it is negotiating cluttered environments. Preliminary simulation results show the feasibility of the proposed approach that constitutes a first step toward understanding how to design switch laws in the active area of behavior based robotics.

REFERENCES

Boccadoro, M., M. Egerstedt and Y. Wardi (2004). Optimal control of switching surfaces in hybrid dynamic systems. In: 7^{th} *IFAC Workshop on Discrete Event Systems, (WODES '04)*. Reims, France. pp. 127–132.

Boccadoro, M. (2005). Optimal Control of Switched Systems with applications to Robotics and Manufacturing. PhD dissertation. University of Perugia. D.I.E.I. available at www.diei.unipg.it/STAFF/boccadoro.htm

Broucke, M. and A. Arapostathis (2002). Continuous selections of trajectories of hybrid systems. *Systems and Control Letters* **47**, 149–157.

Johansson, K.H., M. Egerstedt, J. Lygeros and S. Sastry (1999). On the regularization of zeno hybrid automata. *Systems and Control Letters* **38**, 141–150.

Moreau, L. and D. Aeyels (2000). Optimization of discontinuous functions: A generalized theory of differentiation. *SIAM Journal on Optimization* **11**(4), 53–69.

Polak, E. (1997). *Optimization: Algorithms and Consistent Approximations*. Springer.

Wardi, Y., M. Egerstedt, M. Boccadoro and E. Verriest (2004). Optimal control of switching surfaces. In: 43^{th} *IEEE Conf. on Decision and Control (CDC 2004)*. Atlantis, Bahamas.