Proceedings of 2004 IEEE/RSJ International Conference on
Intelligent Robots and Systems
September 28 - October 2, 2004, Sendai, Japan

Hybrid control of semi-autonomous robots

João Sequeira

Institute for Systems and Robotics
Instituto Superior Técnico
Lisbon, Portugal
Email: jseq@isr.ist.utl.pt

Isabel Ribeiro

Institute for Systems and Robotics
Instituto Superior Técnico
Lisbon, Portugal
Email: mir@isr.ist.utl.pt

Abstract— This paper presents a hybrid architecture for robot control supported on basic concepts from the geometry of Hilbert spaces and nonsmooth calculus. The architecture develops in two classic layers. A supervising layer chooses which motion strategies to apply from a set contained in an execution layer.

The paper focuses mainly in the lower execution layer, motivated by the control of semi-autonomous robots. A set of experiments on the control of unicycle robots is presented.

I. Introduction

The paper describes a framework for the control of robots in underspecified missions. Underspecification tends to reduce the complexity of decision making and hence is specially interesting in the control of semi-autonomous robots by unskilled operators.

The architecture described in this paper has a classic two layer: an execution layer that contains a set of robot motion strategies and a supervising layer that chooses which one to apply at each instant. The novelty of the approach lies in the use of basic tools from nonsmooth calculus to capture key features of the interaction between humans and robots.

The supervising layer is implemented by a finite state automaton. The execution layer is developed around basic concepts of differential inclusions and nonsmooth systems. The overall result is a controller with a fairly intuitive structure, suitable to a large class of applications in semi-autonomous robotics.

In parallel with the development of fully autonomous robot systems (FAR), since the 80's, and even before, there has been a great interest in semi-autonomous robots (SAR). In the realm of SAR control is the human-in-the-loop to provide/improve decision capabilities.

Multiple control architectures, namely the CAMPOUT, [7], [11], the Georgia Tech, [17], the MACTA, [4], and the MAUV, [1] have been proposed along the years. Some of these share conceptual relationships with FAR control architectures from which milestones are the subsumption, [5], and the hierarchical paradigms, [9]. For instance, the CAMPOUT uses a behavior hierarchy that may also be identified in most of the FAR behavior based architectures. The Georgia Tech considers a reactive behavioral layer and additional layers to input mission specifications. The MACTA is also behavior based, though the interaction with humans is made through a mission organizer subsystem. The MAUV implements a sense-process-act loop supported on Artificial Intelligence techniques.

The recent interest in Mars exploration further pushed the development of control architectures for SAR systems. The main reason is that FAR systems are still not robust and intelligent enough to be left alone in a remote place such as Mars. Human supervision is paramount to handle, for instance, contingency scenarios, that often arise in complex real environments.

Besides planetary exploration, a vast number of SAR applications can be foreseen. Remote surveying, aerial mapping, power line inspection, crop dusting, fire fighting, search and rescue in catastrophe scenarios and movie making are just a few examples of socially and economically relevant applications of SAR. In military applications, UAV vehicles, such as the Predator and the Global Hawk, are among the most representative examples, revealing various degrees of autonomy.

A key feature common to any SAR architecture is the interfacing between the human operators and the robots. Whether it is done under graphics, voice or command line based environments, the key problem in SAR control is the mapping between commands specified in the operators language and motion commands. The natural differences in the interpretation of a mission by different operators, i.e., the ambiguity in mission interpretation, lead easily to different commands being sent to the robot and hence to different mission executions. Eventually, these different interpretations of a same mission may be successful.

The standard mission definition, e.g., reach a point or follow a reference path, fails to capture ambiguity. Instead of reaching a specific configuration or following a specific trajectory, the mission objective is changed to reaching a region or tracking a sequence of regions in the configuration space. As explained in the remaining of the paper, this change in the mission objectives sets

0-7803-8463-6/04/$20.00 ©2004 IEEE

the control problem in the framework of differential inclusions and leads to the hybrid architecture proposed. It is worth to emphasize that the architecture described in the paper is not restricted to SAR but can also be used in FAR control.

Two kinds of experiments are considered: (i) with the robot being simulated in Matlab, and (ii) with a real robot made of Lego parts. The former experiment assesses the proposed architecture under controlled conditions In the later, the sensing data, obtained from a video camera mounted on top of the robot (see Figure 2), includes calibration and environment uncertainties and hence allows the assessment of the architecture under realistic conditions.

The paper is organized as follows. Section II details the architecture. Practical implementation aspects are discussed in Section III. Section IV describes the experiments. Section V discusses the results and future work.

II. The hybrid control architecture

For a general robot, reaching a specific region in a configuration space can be done using a diversity of controls. The mapping between the control space and the velocity space can be expressed, as a function of the current configuration, in the form of a differential inclusion

$$\dot{q} \in f(q, U)$$

where $q \in Q$ stands for the configuration, U is the compact control space and $f(q, U) = \{f(q, u), \forall u \in U\}$.

Whenever controlling a SAR, an operator often finds that it is not necessary to accurately specify locations where to drive the robot. For instance, if a video image is used to interface the operator and the robot it may be enough to specify a region of interest, in the image plane, instead of a specific point. Such regions, which may also be specified by some automatic procedure, are arbitrary subsets $K \subset Q$[1].

In an Euclidean space, the set of motion directions

$$\Delta_K(q) = K - q \tag{1}$$

leads from q to any point in K. A sufficient condition for a robot at configuration q to reach a goal set K is that $f(q, U) \cap \Delta_K(q) \neq \emptyset$ be always verified after some time instant on. Assuming an adequate control set U, this condition is easy to match by holonomic robots. For nonholonomic robots, the motion constraints often lead to the non existence of such controls.

In addition to specifying a goal region, the operator may want to specify a region, R, where the trajectory of the robot should be contained. This problem has been studied within Viability theory, [2], [3]. From Nagumo's theorem, [2], the controls that keep the robot inside the region R are defined by $\{u \in U | f(q, u) \in T_R(q)\}$, with R a compact convex set and $T_R(q)$ the contingent cone to R at q. This means that $f(q, U) \cap T_R(q) \neq \emptyset$, i.e., there must be at least one motion direction that is, simultaneously, feasible and that maintains the robot inside R. In practice, the set R is often obtained after sensor information and hence is seldom convex. In such case a generalised Dirichlet tessellation (see [10] for an introduction to the convex partitioning of simple polygons) on R readily enables the application of Nagumo theorem in each of the Dirichlet regions.

Composing the aformentioned sets of motion constraints, the SAR control problem can be defined more generally by,

$$\dot{q} \in f(q, U) \cap T_R(q) \cap \Delta_K(q). \tag{2}$$

For nonholonomic systems it is straightforward to design a situation in which $f(q, U) \cap T_R(q) \cap \Delta_K(q) = \emptyset$ and hence the robot may be forced to move outside R. As an example, consider a car-like robot with limited turning radius and R a small compact region in the C-space centered at the robot configuration. For some orientations, the tightest maneuver the robot can do may lead to a trajectory outside R.

The existence of solutions of (2) requires various types of regularity and continuity conditions such as lower or upper semi-continuity of the set-valued map in the righthand side, [3]. However, even if R is wide enough (and hence does not constrain the motion) the map $\Delta_K(q)$ often may not have connected values[2], thus it is not upper semi-continuous and frequently also not lower semi-continuous[3] and hence generalised solutions are required.

Whenever $f(q, U) \cap T_R(q) \cap \Delta_K(q) = \emptyset$ the control strategy has to be modified to have the two sets $f(q, U)$ and $T_R(q) \cap \Delta_K(q)$ converging to each other[4]. From a purposeful robotics perspective, any robot is a controllable dynamic system. Therefore, whenever the set R is not wide enough to accomodate a solution for (2) there exists a compact set, larger than R, such that a solution exists. The convergence between the above sets corresponds to obtaining a generalized (Filippov) solution for problem (2). Clearly, generalised solutions may not be acceptable for a mission, e.g., the region

[1] For the sake of simplicity one can assume that the reachable space coincides with Q and hence K can be arbitrarily placed.

[2] For example, the "naive" cone (1), for some $K \subset R$, may become not connected when an obstacle is placed in the line of sight of K.

[3] Lower semi-continuity of the map $\Delta_K : Q \rightsquigarrow Q$ requires that any sequence of configurations converging to q leads to a corresponding sequence of images converging to $\Delta_K(q)$. Upper semi-continuity requires that configurations in a neighborhood of q be mapped into images in some compact neighborhood of $\Delta_K(q)$.

[4] For the sake of simplicity in this paper $R \equiv Q$, meaning that no special constraints are imposed on the motion of the robot and hence $T_R(q) \equiv Q$.

R may not be further enlarged if the environment is cluttered with obstacles.

Since $\dot{q} \in f(q, U)$, the set convergence is equivalent to the convergence of $\dot{q}$ to the set $T_R(q) \cap \Delta_K(q)$. Assuming that R is wide enough, the following lemma expresses sufficient conditions for a point $q \in Q$ to converge to a goal set $K \subset Q$ (see [13] for a demonstration).

Lemma 1 (Point to set convergence): Let Q be a Hilbert space, $K \subset Q$ a convex set, $q \in Q \backslash K$, and $T^\star_K(q)$ the conjugate cone to K at a point $\pi_K(q)$, i.e., the set of vectors orthogonal to the space tangent to the border of K at $\pi_K(q)$, with π_K standing for the best approximation projection of q relative to K. Define

$$\lambda = \langle \dot{q}(t), T^\star_K(q(t)) \rangle \tag{3}$$

where $\langle \, , \rangle$ stands for an inner product. If

$$\exists_{T>0} : \forall_{t>T}, \ \lambda > 0. \tag{4}$$

then the distance between $q(t)$ and K monotonically decreases to 0.

□

At each time instant, the condition (4) in the lemma defines a partition $U_i, i = 1, 2, 3$ in the control space U. Figure 1 gives a pictorial view of this partition without intending to give any information on the topological properties of the partition.

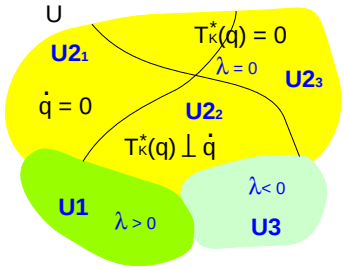


Fig. 1. Control space partition

From a task execution perspective, controls in U_1 are always admissible as they lead to the convergence between q and K. Controls in $U_2 = U_{2_1} \cup U_{2_2} \cup U_{2_3}$ are, in general, admissible but the distance between q and K may stay constant instead of diminishing. Controls in U_3 are in general not admissible as they do not guarantee the convergence between q and K. However, since Lemma 1 only states sufficient conditions, it may happen that controls defining points in this region are useful (see [15] for examples).

The switching between the regions U_i is a discrete event process (the events being determined by the transitions in the sign of λ) that can be modelled by a finite state automaton (FSA). Therefore, the overall architecture encompasses a discrete time layer that supervises a continuous time layer (defined by (2) and Lemma 1) and hence can be considered a hybrid system.

From a mission execution perspective, once the supervising scheme indicates which partition to use, any control value belonging to that partition can be chosen.

The basic structure of the supervision FSA is derived from a direct application of the Lyapunov direct method for non smooth systems. Alternatively, well known results on the stability of hybrid systems can also be used (see for instance [8]). Table I illustrates the chain structure of the supervising FSA where the function $w(q)$ stands for a monotonically decreasing, positive definite, function. This function can be interpreted as the envelop of the values of the function λ^2 (see [15] for details).

The first test in Table I searches for controls in the best region of the partition. If this search is not successful the supervisor layer searches (second test) for controls leading to $\|T^\star_K(q)\| = 0$ which indicate that the robot reached K. The alternative to the above are either controls in U_{2_2}, holding the distance to the goal set, or in U_3, leading to an increase in the distance but potentially simplifying the maneuvering that may be required.

if	$U_1 \neq \emptyset$ choose any control in U_1
elseif	$U_{2_3} \neq \emptyset$ choose any control in U_{2_3}
elseif	$U_{2_2} \cup U_3 \neq \emptyset$ either try to choose a control in U_3 such that $w(q)$ decreases or to enlarge the control space (if possible) or signal mission failure (if all of the above fail)
else	signal error

TABLE I

Structure of the supervising FSA

III. Practical implementation

The practical computation of λ relies on the definition of the operators involved. For a wide range of applications the shape of the goal sets K can be chosen such that it simplifies the computation.

Assuming Q is Euclidean and choosing K as a ball hugely simplifies the computation of $T^\star_K(q)$ as

$$T^\star_K(q) = \alpha \frac{\pi_K(q) - q}{\|\pi_K(q) - q\|} \tag{5}$$

where α is an arbitrary positive constant and the best approximation projector of q into K, defined by $\pi_K(q) = \arg_K \min \|K - q\|$ (see [3] for additional details), is

$$\pi_K(q) = q_K - \rho \frac{q_K - q}{\|q_K - q\|}$$

where q_K is the center of the ball with image K and radius ρ.

It is worth to emphasize that when Q is $\mathbb{R}^2$ or $\mathbb{R}^3$ the motion directions $T^\star_K(q)$ and Lemma 1 have a very intuitive meaning. Since $\pi_K(q)$ represents the point in the boundary of K that is closest to q, the elements in $T^\star_K$ are simply the directions orthogonal to the border

of K at the $\pi_K(q)$. This argument favors the use of this framework for SAR control.

Choosing specific controls in each of the U_i partitions can be done using a variety of techniques. Choosing controls in U_1 with

$$u = \arg_{U_1} \max(\lambda)$$

has an intuitive interpretation when the C-space is $\mathbb{R}^2$ as the controls that maximize the decreasing rate of the distance between q and K. Such scheme leads to nonsmooth control signals, always lying on the boundary of U_1, and hence is not suitable in some circumstances. The smoothness issue also arises when choosing controls in U_2 and U_3. The strategy adopted in this paper for the experiment with the real robot in Section IV is to a posteriori filter the controls.

Table II describes the complete control algorithm (supervision FSA and continuous time controller) for a given partition (the output filtering stage is not included). Controls in U_3 are never chosen. The control set is enlarged by setting the linear velocity to 0 and the angular velocity to a constant value high enough to allow the robot to reorient itself towards the goal set.

if necessary, convexify the goal set $\Delta_K(q)$;
compute $U_1 = \{u \in U : \lambda > 0\}$;
if $U_1 \neq \emptyset$ choose $u = \arg_{U_1} \max(\lambda)$;
otherwise
compute $U_2 = \{u \in U : \lambda = 0\}$;
compute the partition $U_2 = U_{2_1} \cup U_{2_2} \cup U_{2_3}$
where
$U_{2_1} = \{u \in U_2 : \dot{q} = 0\}$
$U_{2_2} = \{u \in U_2 : \dot{q} \perp T^\star_K(q)\}$
$U_{2_3} = \{u \in U_2 : \|T^\star_K(q)\| = 0\}$;
if $U_{2_3} \neq \emptyset$ then choose any $u \in U_{2_3}$;
elseif $U_{2_2} \neq \emptyset$ enlarge the control set with $u = (v, \omega) = (0, cte)$
else signal mission failure
repeat until goal is reached

TABLE II
Robot control algorithm

IV. Experimental results

This section presents two experiments on the control of a unicycle robot.

The first experiment aims at demonstrating the framework when the sequence of goal sets does not exhibit any special regularity properties (as is often the case in SARs controlled by unskilled operators). Video data acquired in real time is used to extract a sequence of goal sets.

The robot is made out of Lego parts and is equipped with an onboard Lego-RCX controller to interface to the motors. The RCX also handles the communications with a laptop computer via the Lego infrared link. The whole architecture together with the image acquisition and processing is implemented in the laptop computer under Matlab. The average sampling time is around 1s. A standard webcam (shown overhead mounted in the robot in Figure 2) is the unique sensor of the system. The images acquired are processed to extract the lightest areas which are then used as raw goal sets. To cope with the conditions in Lemma 1 the raw goal set is first processed to extract the convex hull of the boundary points. A second processing step extracts a circular region that is entirely contained inside the convex hull to simplify the computation of λ (as referred in Section III).

The vision system was subject to a rough calibration procedure to determine a matrix that maps points in the surface into points in the image plane. A retraction of $T^\star_K(q)$ to the image plane is straighforwardly computed using the procedure described in Section III. The expansion to the (x, y, θ) configuration space is made simply by joining the argument of $T^\star_K(q)$, measured directly on the image plane.

The control space is restricted to $v = [0 : 0.01 : 0.2]$ m/s and $\omega = [-1 : 0.1 : 1]$ rad/s, respectively for the linear and angular velocities. The output of the algorithm in Table II is smoothed by a first order filter[5].

Figure 2 shows a snapshot sequence showing the evolution of the robot chasing a rough paper ball (the total distance travelled is around 2m). The position of the ball was changing in real time by action of an external agent (as can be seen by comparing the frames). The practical effect of this change is similar to a change in the region of interest by an operator of the robot. Figure 3 shows a sample of the first 6 images acquired by the vision system mounted on the robot (the complete movie can be found in [16]). In the image plane, the robot is positioned in the bottom middle point. The tracking of the white ball is clearly demonstrated as it approaches the middle lowest part of the image. The convex hull of the light areas extracted and the circular region inside that represents the current goal set are superimposed on the raw image. For such a simple experiment no enlargement of the control space is used.

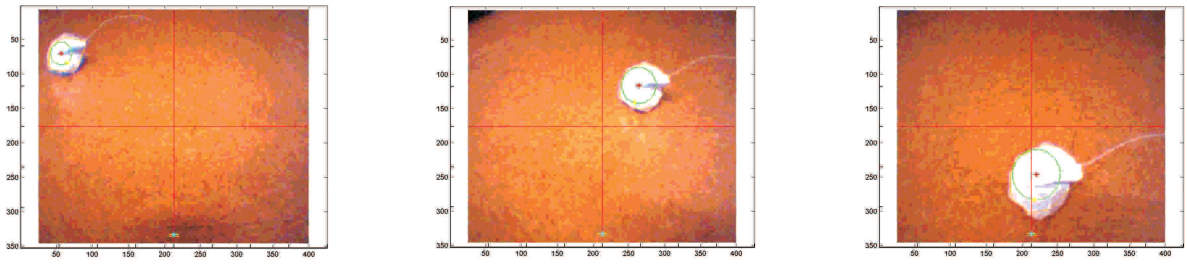
The second experiment, carried out in simulation, illustrates the application of the hybrid architecture framework directly in the $[x, y, \theta]$ C-space of the unicycle robot (see [15] for an application to a car-like robot in the usual C-space $[x, y, \theta, \phi]$). This experiments aims at assessing the framework in the absence of any uncertainty.

The mission of the robot is to move from point $[x, y, \theta] = [2, 1, 2]$ to a ball of radius 0.1 centered at $[0, 0, 0]$. The control space is restricted to $v = [-0.1 : 0.05 : 0.1]$ and $\omega = [-0.1 : 0.05 : 0.1]$, respectively for the linear and angular velocities. No control smoothing was used.

[5]The pole was placed in 0.5 though this is not a critical issue given the dynamics of the Lego robot considered.

1 2

Fig. 2. Snapshot sequence, obtained from an external fixed camera, with the target changing location between frames



1 3 6

Fig. 3. Snapshot sequence obtained with the onboard camera (frames 1, 3 and 6)

Figure 4 illustrates the behavior of a simulated cart robot when the framework is applied directly to the C-space $[x, y, \theta]$. Whenever λ approaches 0, U_1 tends to the empty set. The change in motion strategies imposed by Table II when $U_1 = \emptyset$ and $U_{2_2} \neq \emptyset$ is clearly visible in the nonsmooth parts of the trajectory in the C-space and in the abrupt changes in the λ plot (note that the behavior of the distance between the robot and the control set follows that of λ). Controls in U_{2_1} are avoided as they stop the robot. For this particular robot, controls in U_{2_2} are also not useful as they simply make the robot go around in a circular trajectory around the (ball) goal region. Instead of using controls in U_3, the strategy was to use the knowledge of the structure of the λ function and enlarge the original control set. By judicious choice of the enlargement, the robot "jumps" to a configuration where the λ is again positive (see [12]). For the presented experiment only pure rotations were considered. These are visible at the peaks shown in the controls plot for the angular velocity (note that the linear velocity trajectory is 0.2 during the entire mission and hence it appears as a horizontal line close to 0).

V. Conclusions

The paper presented a framework tailored to the control of SAR. Two very simple experiments using real and simulated unicycle robots demonstrated the main aspects of the framework.

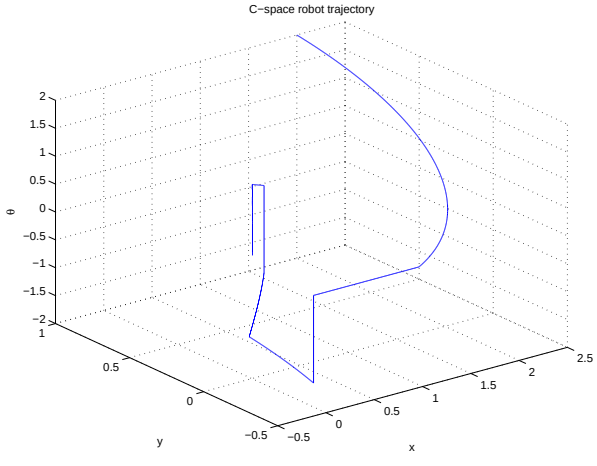
The result of the application of basic concepts in the geometry of Hilbert spaces led to a sound framework and a hybrid control architecture. The experiments presented illustrate the ability to cope with the uncertainty in sensor data, generating a fairly acceptable robot motion.

Rescue and surveillance applications in harsh terrain conditions are potential applications for a mobile robot where (roughly) similar conditions to the first experiment can be found. The experiment resembles a basic visual servoing application for which extensive literature is available (see for instance [6], [18]) though the practical consequences on the stability of the overall system are obtained from very different frameworks.

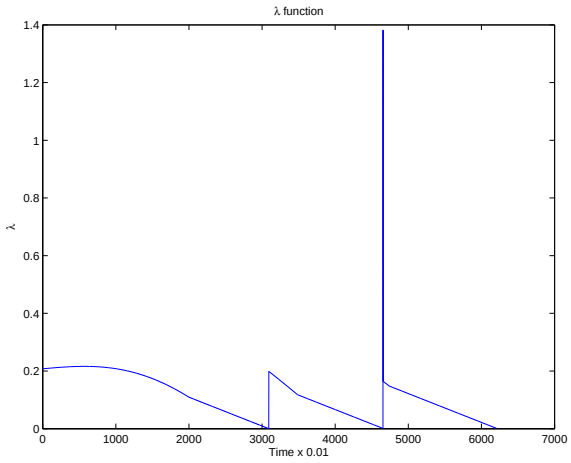
The hybrid framework presented can be extended to the control of robot teams (see [12], [14] for additional examples). Ongoing work includes the test of extensions of this architecture to the control of multiple real robots.

Acknowledgments

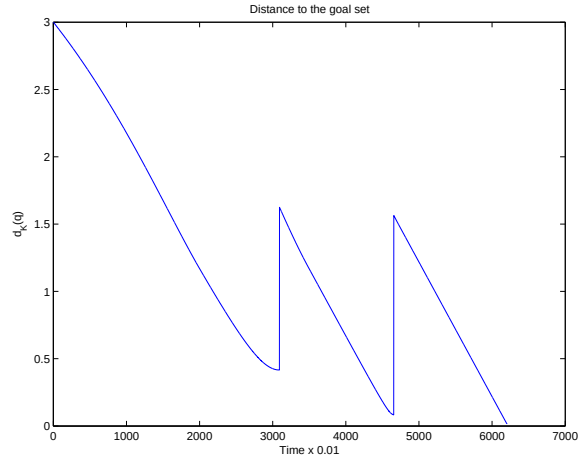
This work was supported by the FCT project SACOR - POSI/SRI/40999/2001 and Programa Operacional Sociedade de Informação (POSI) in the frame of QCA III.



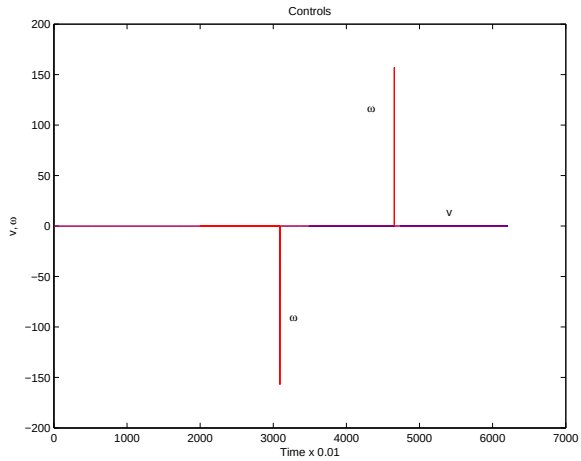
Trajectory in the (x, y, θ) space



Time evolution of the λ function



Time evolution of the distance to the goal set



Time evolution of the controls

Fig. 4. Simulated cart robot

References

[1] J. Albus. A control system architecture for intelligent systems. In Procs. of the 1987 IEEE Intl. Conf. on Systems, Man and Cybernetics, October, 20-23 1987. Alexandria, VA.

[2] J. P. Aubin. Viability Theory. Birkhäuser, 1991.

[3] J.P. Aubin and A. Cellina. Differential Inclusions. Springer-Verlag, 1984.

[4] R. Aylett and D. Barnes. A multi-robot architecture for planetary rovers. In Procs. of the 5th European Space Agency Workshop on Advanced Space Technologies for Robotics & Automation, December 1998. ESTEC, The Netherlands.

[5] R. Brooks. A Robust Layered Control System for a Mobile Robot. IEEE Journal of Robotics and Automation, RA-2(1), 1986.

[6] A. Comport, M. Pressigout, E. Marchand, and F. Chaumette. A visual servoing control law that is robust to image outliers. In Procs. of the 2003 IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems, Las Vegas, Nevada, October 2003.

[7] T. Huntsberger, P. Pirjanian, A. Trebi-Ollennu, H.D. Nayar, H. Aghazarian, A. Ganino, M. Garrett, S.S. Joshi, and P.S. Schenker. Campout: A control architecture for tightly coupled coordination of multi-robot systems for planetary surface exploration. IEEE Trans. Systems, Man & Cybernetics, Part A: Systems and Humans, 33(5):550–559, 2003. Special Issue on Collective Intelligence.

[8] J. Lygeros. Hybrid systems: Modelling analysis and control. Teaching/ee291E.html, 1999. http://robotics.eecs.berkeley.edu/∼lygeros/.

[9] A. Meystel and J. Albus. Intelligent Systems: Architecture, Design and Control. John Wiley & Sons, Inc., 2002.

[10] J. O'Rourke. Computational Geometry in C. Cambridge University Press, 2nd edition, 1998.

[11] P.S. Schenker, T.L Huntsberger, P. Pirjanian, and G.T. McKee. Robotic autonomy for space:cooperative and reconfigurable mobile surface systems. In Procs. of the 6th Int. SYmp. on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS'01), 2001. Montreal, Canada, June 18-21.

[12] J. Sequeira and M. I. Ribeiro. A geometric approach to single and multiple robot control. In Proceedings of the 7th IFAC Symposium on Robot Control, Syroco 2003, Wroclaw, Poland, September, 1-3 2003.

[13] J. Sequeira and M. I. Ribeiro. Geometric control of robot teams. Technical Report RT-602-03, Institute for Systems and Robotics, 2003.

[14] J. Sequeira and M. I. Ribeiro. A layered approach to multiple robot control. In Procs. of the 9th IEEE Intl. Conference on Methods in Automation and Robotics, MMAR 2003, August 25-28, Miedzyzdroje, Poland 2003.

[15] J. Sequeira and M. I. Ribeiro. A geometric approach to single and multiple robot control, 2004. To be presented at the 5th IFAC Symposium on Intelligent Autonomous Vehicles, IAV2004, Lisbon, Portugal, July 5-7.

[16] J. Sequeira and M.I. Ribeiro. Basic visual servoing in SAR systems, 2004. http://omni.isr.ist.utl.pt/∼jseq/filme6_1fps.avi.

[17] Georgia Tech. Real-time cooperative behavior for tactical mobile robot teams - subsystems specification a002, 1984.

Georgia Tech College of Computing and Georgia Tech Research Institute, October.

[18] J.S. Vitor. Visual servoing of cellular robots. In Procs. of the European Control Conference 2001, ECC 2001, September 4-7, Porto, Portugal 2001.