Instituto de Sistemas e Robótica

Pólo de Lisboa

Instituto Superior Técnico, Torre Norte,
Av. Rovisco Pais 1, 1049-001, Lisboa, Portugal

Single And Multiple Robot Control: A Geometric Approach

João Silva Sequeira jseq@isr.ist.utl.pt

Maria Isabel Ribeiro mir@isr.ist.utl.pt

October, 2003

RT-602-03 (revision 0.1)

ISR-Torre Norte
Av. Rovisco Pais 1
1049-001 Lisboa
PORTUGAL

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

Contents

Abstract

This report presents an approach to the control of single and multiple robot systems based on basic geometry properties of Hilbert spaces.

For single robots, the control problem is defined in terms of the convergence between the set of feasible velocities, defined by the robot kinematics, and the set of goal velocities, defined by the mission specification and the sensor data acquired along the mission. Basis concepts of the geometry of Hilbert spaces are used to derive conditions on the convergence between these sets.

The motion behavior of robot teams is classified, according to the robot neighboring relationships, into formation and free motions. The control of a team extends the results for single robots by constraining the set of goal velocities to account for the neighboring relationships among teammates. Each robot monitors these relationships for relevant changes that are mapped into an event set. A finite state automaton is used to map the event set into a set of motion strategies.

The report presents simulation results on both single and multiple robot control using 2D holonomic, cart and car robots.

1 Introduction

The feedback control problem for common mobile robots operating under perfect information, such as the unicycle and the car robots, has been solved for more than a decade (see for instance [Khatib et al., 1997, Murray and Sastry, 1993, Samson, 1992, Sordalen, 1993] for specific examples and [Hermann and Krener, 1977] for the general controllability and observability conditions). Robot control has been considered either in the framework of dynamical systems or using a miscellaneous of other techniques, ranging from artificial intelligence, [Arkin and Balch, 1998, Drogoull and Collinot, 1998, Jennings, 1999, Parker, 1998], to discrete event systems, [Košecká et al., 1997], to fuzzy and hybrid systems.

Brocket's theorem, [Brocket, 1983], stating that no smooth time-independent feedback control law is guaranteed to exist for nonholonomic systems, establishes, on one hand, a sort of boundary on the research on this topic, and, on another hand, a starting point for non-smooth, e.g., hybrid strategies.

Perfect information is seldom a realistic assumption, and hence strategies to deal with the corresponding uncertainty have been presented in the literature along the years. In addition to the uncertainties in the models, the control of a robot must also cope with the uncertainties generated by the environment such as those due to innacurate data acquired through sensors. This class of uncertainties has been tackled mainly through Kalman filtering techniques.

A large variety of tasks can be loosely specified, not requiring exact path following. Instead, any path in some class can be followed, meaning that robots are only required to stay within some bounded region in the workspace. This class of uncertainties is related to the task[1] decomposition. Tasks involving motion, such as those in robotics either being manipulation or locomotion, end up by requiring the specification of a reference path for the robot to follow. In general, the computational effort to define such paths is greater than that to define a class of paths.

Whenever the dynamics of the interaction between the environment and the robot(s) is relevant task specification becomes important. Such is the case for semi-autonomous and multiple robot systems. For the former, the human operator in the loop may have to take fast decisions to cope with a specific situation and hence time consuming procedures to define a single reference path may not be an option. For the later, similar issues arise, these being also related to computational complexity.

1.1 Single robot issues

Semi-autonomous robots, operating mainly in isolated form, have been gaining importance in the last few years after the slow increase in the use of standard autonomous robotics in practical applications. Despite the vast work in the field of autonomy, where multiple architectures have been proposed and a large number of functionalities identified, intelligent control architectures have still to evolve at the higher decision levels that are often required by full autonomy.

Applications such as surveillance in wide open areas, rescue missions in catastrophe scenarios and the maintenance of remote infrastructures are examples of socially/economically relevant applications where autonomous robots can be used. Current state of the art locomotion/manipulation technologies are able to remotely operate in most of these environments under guidance of an external operator. Relevant examples can be found in minning, sewers inspection and maintenance, in aerial surveillance and even in medicine with some remote surgical interventions. The interactions between robots and operators accounting for the fast dynamics of real environments that require fast control actions to be taken either by the human operators or the robots. The teleoperation concept from the early days of robotics evolved

[1] Along the report the terms *task* and *mission* are used interchangeably.

to control paradigms in which (i) the human operator only interacts with the robot at sparse instants providing decision capabilities whenever that of the robot reaches its limit, and (ii) the interaction is specified in loose form, i.e., goals are specified up to some uncertainty.

The single robot control problem can be formulated to account for both classes of uncertainties above by constraining the robots to stay in some bounded area of the configuration space, instead of specifying a given pose or following a particular path. In this report the control problem for single robots is set up in the framework of differential inclusions[2]. Tools from Hilbert spaces are used to develop a control algorithm supported on basic geometry concepts. These yield a partition in the control space and a finite state automaton is used to provide the switching among the elements in this partition.

1.2 Multiple robot issues

In parallel to semi-autonomous robotics, in the last years there has been an increasing interest in multiple robot problems, e.g., spacecraft and military formations, for which techniques ranging from control theory to artificial intelligence have being proposed in the literature. The use of multiple robots is justified by:

Redundancy, provided by the competition among the team members to execute the mission;
Complementarity, provided by the different specializations existing among the team members, e.g., different sensing capabilities and kinematics allowing different forms of motion in the workspace and perception of the environment.

Multi-robot systems share most of the control issues present in the control of single robots. In addition, robot teams are also subject to disturbances in the communications and to the constraints introduced by simultaneous motion of the teammates.

In [Parker, 1998] a behavior based architecture for fault tollerant cooperating robots was proposed. This three level architecture is based on the subsumption architecture. The highest level models the motivation of a robot using performance measures such as the impatience (the attitude of the robot towards the teammates) and acquiescence (the attitute towards itself). The intermediate level contains sets of specific behaviors activated by the motivational behaviors. Each of these sets emphasizes a particular global robot behavior, e.g., find a location using a methodical behavior or a wander behavior. The lowest level contains the basis competence layers for the robot to survive.

Motor schemas (primitive motion strategies) were used in [Balch and Arkin, 1999] to control teams of cart and car-like vehicles aiming at moving in formation. The formations considered were defined either using the distances between a robot and its neighbors, between a robot and a team leader or between a robot and some characteristic point of the formation, e.g. the center of mass. Formation specific motor schemas allow each robot to compute its motion direction so that no formation break occurs.

In [Egerstrdt and Hu, 2001] the formation control problem was decoupled into the planning of a reference path to be followed by a virtual leader and a tracking problem to be handled by the real robots in the team. The formation is defined by the kernel of a convex map such as sum of quadratic errors between the position of each robot and its reference trajectory. The formation control is given by a steepest descent technique on the map defining the formation.

Formations defined by smooth functions of the relative distances of fully actuated spacecrafts were also considered in [Lee and Li, 2003]. The overall system is divided into an average system, that captures the average motion of the team, and a shape system that captures the relative velocities among the robots

[2]Roughly, the control of systems described by differential inclusions is usually considered in the framework of Viability theory(see [Aubin, 1991] for details).

and hence the formation pattern. PD-like laws are used to make both the average and shape systems follow their reference trajectories.

In [Ögren and Leonard, 2003] a formation is defined by a set of vectors defined after the relative positions of the robots in the team. The formation control problem is formulated as the control of a set of double integrators, acting under a leader-follower strategy. A compactness assumption on the control space of each robot allows the definition of a compact uncertainty region around its current position. The free space the team is allowed to use to avoid formation breakings is obtained by superimposing this region on the obstacles in the environment.

The approach presented in this report differs from the ones in the current literature in that it separates the control problem into a layer that depends exclusively on the properties of the behavioral spaces of the robots and a layer that handles the problem dependent information. The extension to multiple robot systems of the framework developed for single robots is done by introducing the concept of neighboring relationships among the teammates.

The report is organised as follows: Section 2 introduces the relevant aspects of differential inclusions to robotics applications by dicussing a practical example. Section 3 describes the main results used to design the control algorithm. Sections 4 and 6 describe algorithms for the control of single and multiple robot systems supported on the framework detailed in Section 3. Section 7 illustrates the applicaction of basic results in non-smooth analysis to the proposed framework. Section 8 presents simulation examples using holonomic, cart and car-like robots in both single robot and team missions. Section 9 presents the conclusions of the report. Appendices A and B present the demonstrations and useful results used throughout the report.

2 Differential inclusions and Robotics

A robot, considered as a dynamical system, is usually represented in the form

$$\begin{aligned} &\dot{q}(t) = f(q(t), u(t), t) \\ &q(0) = q_0 \\ &u \in U \end{aligned} \tag{1}$$

where t is the time, $Q = \{q\}$ stands for the C-space, u are the exhogenous control inputs and f describes the motion capabilities of the robot. Without lack of generality, throughout the report f is assumed to describe only intrinsic properties of the robot, e.g., the kinematics[3].

The synthesis of $u(t)$ such that $q(t)$ achieves a given goal is the classical control problem.

Defining a multiple valued map (or set-valued map) such that $F(q,t) = \{\dot{q} : \dot{q} = f(q,u,t)_{u \in U}\}$, the expression (1) can be written in the differential inclusion form as

$$\begin{aligned} &\dot{q}(t) \in F(q(t), t) \\ &q(0) = q_0 \end{aligned} \tag{2}$$

$F(q(t), t)$ represents the set of motion directions available at time t, given the set of available controls U. The inclusion relationship in (2) expresses explicitly the ambiguity in the motion direction the robot can take at time t. When defining control objectives, this ambiguity can be explored as it allows the loose specification of goals mentioned in Section 1.

[3] To avoid cumbersome expressions, hereafter the explicit dependence on t in the model (1) is ommitted whenever no confusion arises.

In what concerns the design of motion controllers, the loose specification of control objectives has a number of benefits, namely

- it is less demanding in terms of accuracy in the sensed data;
- can cope with imperfect data on the environment;
- it is suitable for linguistic description

Inaccuracy and imperfect knowledge result, for example, from sensors that only scan regions in the neighbourhood of the robot or have technological constraints (e.g., ultrasound reflections and poor range measures).

The linguistic description of a task shortens the communication with the robot (either of origin/destination in a human or in another robot). For instance, the passing through a door, relying on perfect information, may be mapped into *move along the region under the door*. This description of the task does not specify an exact motion direction, or trajectory, to be taken by the robot. Instead, the fact that a whole region of space can be used to accomodate trajectories through the door introduces ambiguity that can be stated in terms of a set of motion directions that drive the robot through the door or in terms of the space the robot can use.

If, from the task description, the robot must execute a trajectory entirely contained inside a region defined by some set $K \subset Q$, then the robot control problem is a *viability* problem. This is defined as: compute a control $u \in U$ such that a trajectory *viable under* D is produced, that is, a trajectory such that, for all t and $q_0 \in D$, $q(t) \in D$, i.e. the trajectory of the robot will stay inside D[4].

The region D will often change at each event (e.g., as a result of a change in perception or in the raw sensor data) and there is the possibility that the new D and the previous one are disjoint. In such case the control must be chosen such that the actual trajectory converges (in some sense) to the new D which amounts to the convergence of the robot velocity to some region in the velocity space.

The literature on differential inclusions and related topics such as set-valued analysis is immense. The key ideas in this report come from [Aubin, 1991, Aubin and Cellina, 1984, Smirnov, 2002, Bacciotti et al., 2000].

2.1 Setting up a motivational example

Consider the following example involving a cart like robot (see Figure 1 where x, y, θ stand for the position and orientation variables, and v, ω stand for the linear and angular velocities).

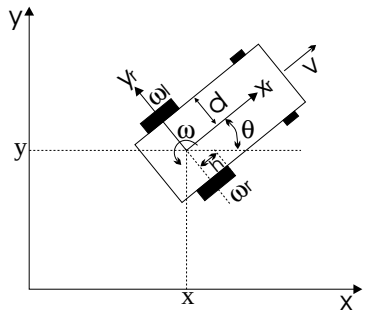
The cart is assumed to be able to turn with arbitrary curvature radius and is equipped with a single sonar for obstacle detection mounted along the robot x axis and facing ahead. Furthermore, assume that this sensor produces a sound beam of conic shape with 20° aperture[5].

At each configuration q, the information gathered by the ultrasound sensor defines a region in the robot's C-space either indicating free ou cluttered space. The robot must move in such a way that the trajectory stays inside the free space defined by the ultrasound data.

The detection of an obstacle means that the projection of the sonar beam on the xy plane (the workspace plane) represents a region where the robot has additional knowledge on the environment. For the purpose

[4] In a "standard" control problem each point of the reference D does not have a neighborhood entirely contained in D and hence it belongs to a unique trajectory.

[5] Apart from the aperture angle, ideal characteristics are assumed.



$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos(\theta) & 0 \\ \sin(\theta) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}, \qquad (3)$$

Figure 1: Cart robot kinematics

of this example, it is assumed that the information provided by the ultrasound system is processed returning the free space the robot can use whilst moving towards the goal location.

At some time instant t_0, it is assumed that the robot is at pose $q_0 = (x_0, y_0, \theta_0)$ and a snapshot of the environment is taken using the ultrasound sensor. In the workspace, the region scanned by the sensor is defined by a horizontal cross-section of the sonar conic beam in the (x, y, z) space. This is represented in the C-space by a "slice of pie" volume, meaning that the conic sector may be obtained at any orientation of the robot,

$$D(q_0, L_{\min}, L_{\max}) = \left\{ q : \begin{bmatrix} x_0 \\ y_0 \\ \theta_0 \end{bmatrix} + h \begin{bmatrix} u_1 \cos(\theta_0 + u_2) \\ u_1 \sin(\theta_0 + u_2) \\ u_3 \end{bmatrix}, \quad \begin{array}{c} \frac{L_{\min}}{h} \le u_1 \le \frac{L_{\max}}{h} \\ -\frac{10\pi}{h180} \le u_2 \le \frac{10\pi}{h180} \\ \frac{-\pi-\theta_0}{h} \le u_3 < \frac{\pi-\theta_0}{h} \end{array} \right\} \qquad (4)$$

where h is a scaling factor[6], and $L_{\min}$, $L_{\max}$ stand for the minimal and maximal sonar detection range respectively. The variables u_1, u_2 span, respectively, the length and the aperture of the conic sector. The variable u_3 forces the spanning of the entire range of orientations, meaning that the robot can moves from q_0 towards the inside of the conic region with any orientation.

Every trajectory generated by the robot must be contained inside the obstacle free region detected by the ultrasound, i.e.,

$$q \in D(q_0, L_{\min}, L_{\text{safety}}) \qquad (5)$$

for some adequate L_{safety}[7].

Let $K \subseteq D$. For any configuration $q \in K$, the vector $q - q_0$ points towards K and represents a motion direction. The set K is named a *goal set* as it represents the region of the C-space the robot would converge to in the absence of any constraints. The set of all such points, hereafter named the *sensed viable region*, is thus

$$\Delta_{K_h}(q_0) = q\big|_{q \in K} - q_0 = K - q_0 = \left\{ h \begin{bmatrix} u_1 \cos(\theta_0 + u_2) \\ u_1 \sin(\theta_0 + u_2) \\ u_3 \end{bmatrix} \quad : \quad \begin{array}{c} \frac{L_{\min}}{h} \le u_1 \le \frac{L_{\max}}{h} \\ -10\frac{\pi}{h180} \le u_2 \le 10\frac{\pi}{h180} \\ \frac{-\pi-\theta_0}{h} \le u_3 < \frac{\pi-\theta_0}{h} \end{array} \right\} \qquad (6)$$

[6] In fact, this scaling factor corresponds to the time taken by the robot to go from q_0 to the point inside the conic area specific by (u_1, u_2, u_3) in a discrete simulation.

[7] In general, $L_{\min}$ depends on some technological constraint whereas L_{safety} depends on the mission being executed. Hereafter, to avoid cumbersome notation $L_{\min}$ and $L_{\max}$ are omitted when refering to D.

If the set (6) is made to represent instantaneous velocities, the dependence on h can be removed. It is worth to emphasize that (6) does not account for the robot kinematics.

Given the cart kinematics, (3), and the motion constraints, (6), the corresponding robot control problem is stated as:

Definition 1 *Choose a motion direction in (6) under the constraint (3) or vice-versa.*

The set of admissible motion directions that is the solution to the control problem of Definition 1 satisfies

$$f(q_0, u)_{u\in U} \cap \Delta_K(q_0) \neq \emptyset. \tag{7}$$

The solution of (7) for the current example can be obtained[8] from

$$\begin{bmatrix} u_1 \cos(\theta_0 + u_2) \\ u_1 \sin(\theta_0 + u_2) \\ u_3 \end{bmatrix} = \begin{bmatrix} v\cos(\theta_0) \\ v\sin(\theta_0) \\ \omega \end{bmatrix} \tag{8}$$

resulting in $u_2 = k\pi,\ \ k = 0, \pm1, \pm2, \ldots$, from which only $u_2 = 0$ lies inside the interval $\left[\frac{-10\pi}{180}, \frac{10\pi}{180}\right]$. The solutions of (7) are,

$$\{(v,\omega)\}\,|_{u_2=0} = \left\{ (u_1, u_3) \ : \quad \begin{array}{c} 0 \leq u_1 \leq 1 \\ -\pi - \theta_0 \leq u_3 < \pi - \theta_0 \end{array} \right\} \tag{9}$$

and without further refinement criteria, every pair in (9) drives the cart towards a configuration inside the set $K(q_0)$. This solution is pretty much obvious: if the robot is already facing the conic area that defines the goal set K (as imposed by the constraint $u_2 = 0$) then moving straight by a small amount will keep it inside the sensed viable region.

Often (7) will be empty, for instance due to the intrinsic (to the model) kinematics constraints or to limitations in the control set.

Whenever there is an empty set solution to (7) it is necessary to control the robot such that the desired velocity set, defined by Δ_K, and the feasible velocity set, defined by $f(q,u)_{u\in U}$, converge to each other. The control problem in Definition 1 is then modified to reflect the (in general) existence of a convergence process between two sets.

Definition 2 (Robot control problem) *Choose a robot velocity (and hence a set of controls) such that the set* $\{f(q,u)\}\,|_{u\in U}$ *converges to the set of motion directions,* $\Delta_K(q)$*, or vice-versa.*

2.2 Loose task specification and viability problems

Following the example in the previous section, the loose specification of a task is often in the form of a region $D \subseteq Q$. This region can be further pruned from the zones cluttered with obstacles resulting in the free C-space where the robot trajectory must be contained. Such regions are named *viability domains* in Viability theory. Definition 3 extends this concept to the problem considered in this report.

Definition 3 (Viability domain) *A set D is a* viability domain *for a robot described by (2) and operating under environment velocity constraints Δ_K iff the trajectory of the robot stays inside D independently of $F(q) \cap \Delta_K$ empty*

[8] For the sake of simplicity, consider $L_{\min} = 0$ and $L_{\max} = 1$.

The solution of standard viability problems, i.e., for systems in the form (1), is given by the Nagumo theorem (see [Aubin, 1991]). Roughly speaking, Nagumo's theorem states that, if a set K is a viability domain,

- if the robot is inside K then any motion direction can be used during and infinitesimal time, as the robot will continue to stay inside K;
- if the robot is at the border of K then it must move either along the tangent to the border of K at its present location or along a direction poiting towards the interior of K.

The above two conditions can be summarized by stating that the robot must move along the contingent cone to K at the current robot configuration.

Nagumo's theorem is not applicable during the convergence process mentioned in Definition 2 (as the dynamic system would be represented by $\dot{q} \in \emptyset$). However, it still makes sense to define viability domains, for instance as the space spanned by the robot during the convergence.

Definition 3 is extended to robot teams by requiring that the trajectory generated by any of the robots in the team be contained inside the viability domain.

Once a task is specified (through a viability domain) it must be further decomposed into a sequence of goal sets. Each of these goal sets is then used to define the corresponding Δ_K sets. The robot will then proceed in sequence towards each of these subgoals. Section 5.1 further details the main aspects related to this decomposition.

3 Convergence between sets

The convergence concept between two sets is equivalent to have a distance measure converging to 0. In this section some basis results on the requirements for a convergent relative motion between two generic sets are presented. The following definition for the distance between two sets A and B is used[9].

Definition 4 (Distance between subsets of a normed space) *Let A and B be two subsets of a normed space. The distance between A and B is defined as*

$$d(A, B) = \min_{x_a \in A, x_b \in B} \|x_a - x_b\| \tag{10}$$

The distance between a point $x_a \in A$ and a set B is defined by making A a singleton

$$d(x_a, B) = \min_{x_b \in B} \|x_a - x_b\|. \tag{11}$$

Both (10) and (11) are not metrics as they do not verify the triangular inequality (see Figure 2 for an illustrative counterexample). However they are commonly referred as distances, [Simmons, 1963, Kreysig, 1978].

Note that, since a robot's C-space is in general non-euclidean[10], by using an euclidean norm it is implicitly assumed that:

[9] In this section the notation slightly differs from the one in Section 2.1 to emphasize the fact that these are generic results - in Hilbert spaces - and not tailored for use in Robotics. Points in sets are denoted by x and y.

[10] In fact a smooth manifold embedded in the euclidean $\mathbb{R}^n$, [Latombe, 1992].

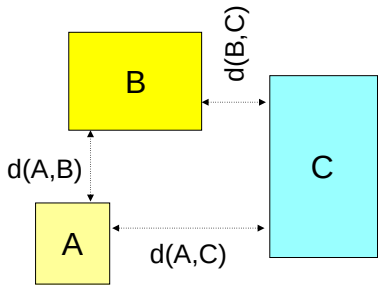


Figure 2: Distances that are not metrics

- every point in the C-space is reachable, as distances between any two points are being computed (and hence there exist at least a path between any two points);
- the distance does not measure the "effort" to travel along a path connecting any two points.

For example, if the shortest path between the points lies in a surface with non-null extrinsic curvature[11] then the distance computed with (11) will be lower than the corresponding path length.

Some basic assumptions and facts on sets of Hilbert spaces are used to design the convergence algorithm.

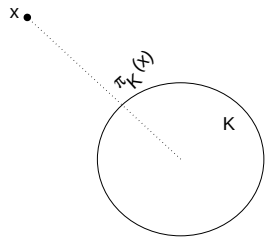
Theorem 1 (Best Approximation Theorem) *Let K be a closed convex subset of a Hilbert space, X. There is a map $\pi : X \mapsto K$ and a unique element $\pi_K(x)$ verifying*

$$\forall_{x \in X}, \quad \|x - \pi_K(x)\| = \min_{y \in K} \|x - y\|$$

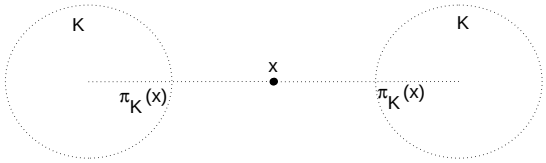
Demonstration: See [Aubin and Cellina, 1984].

□

The map $\pi : X \mapsto K$ projects X into K. Figure 3 illustrates the meaning of this projection in the 2D euclidean space.



a) Projection for a convex K



b) Ambiguity in the projection for non convex K

Figure 3: The meaning of the projection map π_K in the 2D euclidean space

[11] That is, a surface.

The π map is called a *best approximation projector* (b.a.p for short) whenever the following two conditions are verified

$$\begin{aligned} &\|\pi_K(x) - \pi_K(y)\| \leq \|x - y\| && \text{non-expansivity} \\ &\langle \pi_K(x) - \pi_K(y), x - y \rangle \geq 0 && \text{monotonicity} \end{aligned} \tag{12}$$

where the $\langle \cdot, \cdot \rangle$ stand for an inner product and $\| \cdot \|$ for the corresponding induced norm (the added subscript K is used to make explicit the dependence of this map on the projection set K).

The generation of a trajectory converging towards a convex set can be derived using some basic tools of geometry. The following lemma states sufficient conditions for this convergence to occur.

Lemma 1 (Point-to-set convergence) *Let X be a Hilbert space, $G \subset X$ a convex set, $x \in X \backslash G$ and $\pi_G(x)$ a b.a.p. of a point x onto G. Furthermore, let x_n be a sequence from a trajectory $x(t)$, outside G, and define*

$$\lambda_n = \langle x_{n+1} - x_n, \pi_G(x_{n+1}) - x_{n+1} \rangle \tag{13}$$

such that

$$\exists_{T>0} : \forall_{t_n > T}, \lambda_n > 0. \tag{14}$$

Then $d(x_n, G)$ is strictly decreasing. Furthermore, if x_{n+1} such that λ_n decreases slower than $d(x_n, G)$ then $\lim_{n\to\infty} d(x_n, G) = 0$.

Demonstration: See Appendix A.

Figure 4 illustrates a particularly intuitive interpretation of Lemma 1 when X is the 2D euclidean space. The distance λ is the result of the inner product in the theorem and represents the amount of distance to the goal set K that is gained by chosing the goal point x_{n+1} when the robot is at x_n.

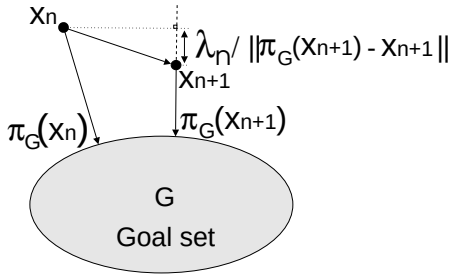


Figure 4: Interpretation of Lemma 1 when X is a 2D euclidean space.

When $x_{n+1} \to x_n$ (13) can be formulated in continuous time as

$$\lambda = \langle \dot{x}, T_K^{\perp}(x) \rangle \tag{15}$$

where $T_K^{\perp}(x)$ stands for the orthogonal contingent cone to K at x[12]. Alternatively, $T_K^{\perp}(x)$ can be written as the conjugate cone to K at $\pi_K(x)$ and denoted by $T_K^{\star}$ (see Appendix B).

Lemma 1 stands for a discrete time approach, with (15) expressing the condition for the equivalent continuous time approach.

[12] This is an informal statement meant to be intuitive. The definition of contingent cone to K at x, $T_K(x)$ requires that $x \in K$ which is not the case for $T_K^{\star}(x)$.

It is worth to emphasize that no particular physical meaning was assigned to the X space considered in this section with the two converging sets being generic. If the sets involved represent velocities, as in Section 2.1, then x_n stands for a velocity sampled at some time instant n.

If the control problem is defined by algebraic expressions the discrete convergence lemma can be used. This is the case when the robot is modelled by an algebraic model, e.g., a serial manipulator, and the goal set is defined in the configuration space. For the example in Section 2.1, the control problem was specified in terms of the convergence of the robot possible velocities, $f(q, U)$, towards a set of reference velocities obtained from sensor data, $\Delta_K(q)$. This corresponds to an algebraic formulation of the control problem and hence the discrete version of Lemma 1 can be applied.

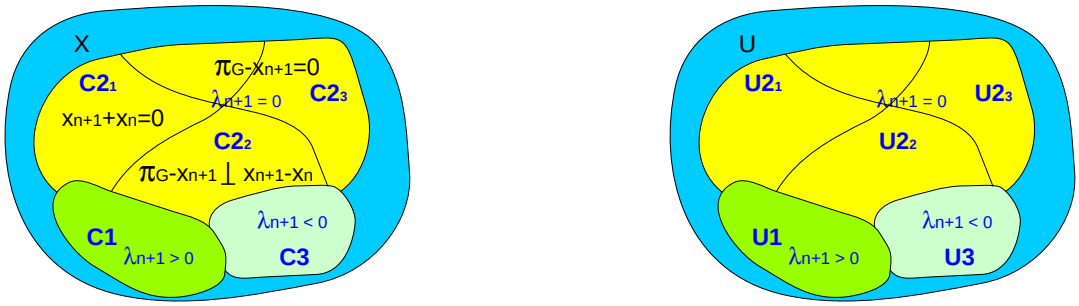
For non algebraic formulations of the control problem the continuous version of the convergence lemma is preferable. This is the case when the robot is modelled by a differential system and the goal set is expressed in the configuration space.

4 Single robot control

The control of a single robot under a loosely specified task, is based on expressions (13) and (15).

$$\begin{array}{l} \text{if } \lambda > 0 \text{ the distance between the sets is decreasing,} \\ \text{if } \lambda = 0 \text{ the distance between the sets is constant,} \\ \text{if } \lambda < 0 \text{ the distance between the sets is increasing.} \end{array} \tag{16}$$

The conditions (16) define three disjoint regions in the behavioral space X (see Figure 5). In the control space the equivalent classes form a partition $U = U_1 \cup U_2 \cup U_3 = U_1 \cup U_{21} \cup U_{22} \cup U_{23} \cup U_3$.



Classes in the behavioral space Classes in the control space

Figure 5: Equivalence classes

For a general robot, and in the absence of further refinement criteria, (most likely dependent of the goal of the particular problem), q_{n+1} (in the discrete approach) or $\dot{q}$ (in the continuous approach) can be chosen using exhaustive search in the control space. This avoids having to solve inverse problems, e.g., computing u from $\dot{q}$. By performing the search in a compact control set U, each of the controls tested is classified into one of the elements U_i of the partition according the sign of the corresponding λ.

Defining a good search strategy depends on the particular robot kinematics. For holonomic robots a greedy strategy such as

$$u = \arg_U \max(\lambda)$$

can be used as the motion of the robot is not constrained.

In the nonholonomic case the greedy approach may lead to a local minima problem. These arise when $\lambda = 0$ and hence the distance between q_n and $\Delta_K(q_n)$ (in the discrete approach) or q and the set K (in the continuous approach) is constant (even though the robot may be moving). The solution to this local minima problem depends on the control goal and requires the detection of conditions under which there is a loss in the number of degrees of freedom in the control space, that is, one or more controls do not cause any motion progress[13].

The control synthesis algorithm is sketched, in terms of the above partition for the control set, in Table 1 for the discrete case and in Table 2 for the continuous case.

if necessary, convexify the goal set $\Delta_K(\dot{q}_n)$;
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 $U_2 = U_{21} \cup U_{22} \cup U_{23}$
where
$U_{21} = \{u \in U_2 : \dot{q}_{n+1} - \dot{q}_n = 0\}$
$U_{22} = \{u \in U_2 : \dot{q}_{n+1} - \dot{q}_n \perp \pi_{\Delta_K}(\dot{q}_{n+1}) - \dot{q}_{n+1}\}$
$U_{23} = \{u \in U_2 : \pi_{\Delta_K}(\dot{q}_{n+1}) - \dot{q}_{n+1} = 0\}$;
if $U_{23} \neq \emptyset$ then choose any $u \in U_{23}$;
otherwise
compute $U_3 = U \backslash U_1 \cup U_2$
choose $u \in U_{21} \cup U_{22} \cup U_3$;
repeat until goal is reached

Table 1: Algorithm to implement the discrete version of Lemma 1

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 $U_2 = U_{21} \cup U_{22} \cup U_{23}$
where
$U_{21} = \{u \in U_2 : \dot{q} = 0\}$
$U_{22} = \{u \in U_2 : \dot{q} \perp T_K^\star(q)\}$
$U_{23} = \{u \in U_2 : \|T_K^\star(q)\| = 0\}$;
if $U_{23} \neq \emptyset$ then choose any $u \in U_{23}$;
otherwise
compute $U_3 = U \backslash U_1 \cup U_2$
choose $u \in U_{21} \cup U_{22} \cup U_3$;
repeat until goal is reached

Table 2: Algorithm to implement the continuous version of Lemma 1

The liveness properties of each of the algorithms in Tables 1 and 2 depend on the particular robot under consideration. The algorithm has no deadlocks as each state has an output transition. However, the

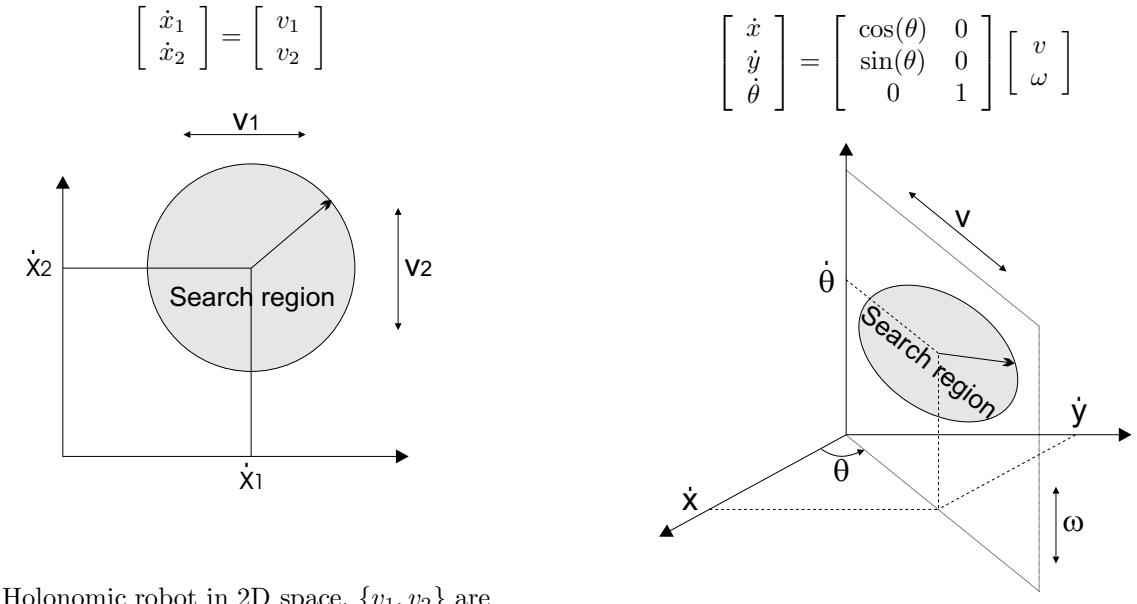
[13] These controls are in the kernel of λ.

choice of controls in $U_{21} \cup U_{22} \cup U_3$ depends on the robot capabilities and/or mission specifications. If not properly chosen, these may lead to a livelock situation in which the robot endlessly repeats a sequence of maneuvers.

Tables 1 and 2 define hybrid algorithms. The continuous state is defined by the motion strategies applied at each of the C_i regions. The discrete state is defined by the states of an FSA used to control the switch between the C_i.

The limited exhaustive search in the control space is illustrated in Figure 6 for a 2D holonomic robot and a cart robot. The two plots represent the velocity spaces for each of the robots, $\{\dot{x}_1, \dot{x}_2\}$ for the 2D holonomic robot, and $\left\{\dot{x}, \dot{y}, \dot{\theta}\right\}$ for the cart robot. The 2D holonomic robot is controlled through the variables v_1 and v_2, each defining the velocity along the basis directions in the C-space. The cart robot is controlled through the linear and angular velocities in the robot local frame, respectively v and ω.

In each plot, the shadowed area named "search region" represents the ball centered at the current robot velocity. The algorithm in Table 1 searches for the adequate controls inside these balls. For the 2D holonomic robot this ball is 2-dimensional as the ambient space. For the cart robot this ball is 2-dimensional, instead of the 3D of the ambient space, as the kinematics imposes that the robot velocity evolves along the vertical plane shown in the figure.



a) Holonomic robot in 2D space, $\{v_1, v_2\}$ are the control variables

b) Cart robot, $\{v, \omega\}$ are the control variables

Figure 6: Exhaustive search for controls.

5 Task decomposition

5.1 Convex goal sets

The convexity assumption on the goal set imposes some processing on the data acquired by the sensors. For example, if a goal set K is extracted from an image representing some environment scenery then it is likely that the raw goal set K, i.e., the area in the image where the trajectory performed by the robot must lie, has an arbitrary shape.

The extraction of a convex set out of the acquired raw K can be made using computational geometry techniques[14]. An alternative technique is to decompose the raw K into Dirichlet (or Thiessen) regions and use them all in sequence; see [O'Rourke, 1998] for an introduction to the convex partitioning for simple polygons. The same convexifying procedures can be used if the goal set is $\Delta_K(q_n)$. Depending of the particular problem, the area acquired through sensing can be expanded or contracted during the convexifying process.

The set of convex regions resulting from the decomposition of K can be retracted into a visibility graph or topological map (see [Latombe, 1992] for an introduction to retraction maps and visibility graphs in robotics).

When driving a robot towards a goal set K, any of the subsets in the partition can be used instead of K. Clearly, if the goal subset changes in time the convergence can not be guaranteed even if $\lambda > 0$ at all times (see Figure 7 where if the goal subset is constantly changing between K_1 and K_2 the robot may not approach neither of them).

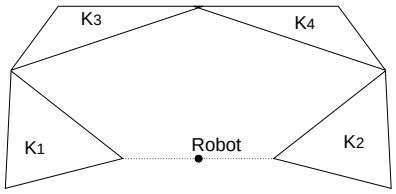


Figure 7: Switching between goal sets in a convex decomposition

5.2 Convexifying viability domains

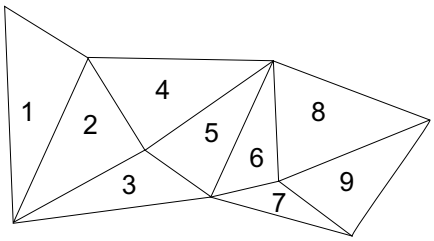
As mentioned in Section 1, driving the robot through an arbitrary region is a viability problem. The viability domain can be specified by an external (human) operator or by another robot.

For a vast class of problems, having suitable regularity properties, the solution can be constructed using Nagumo's theorem in a constructive way. An intuitive approach, suitable for the purpose of this report, relies also on the decomposition of the arbitrary region into convex regions and visibility graphs.

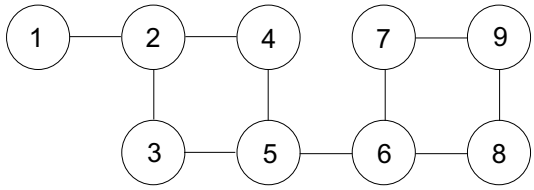
Using standard graph search techniques, a sequence of convex goal sets containing a reference path between the current robot location and the mission goal can be defined. A supervisory controller[15] is used to choose, at each time instant, the adequate goal set among the elements in the partition so that the algorithm in Table 1 (or that in Table 2) can be applied. Figure 8 illustrates an example of a convexification of a viability domain D and the corresponding visibility graph.

[14] For instance, assuming a simple polygonal region it is possible to use either polygon convexifying techniques, [Grünbaum, 1995], or the convex hull, [Preparata and Shamos, 1985].

[15] Implemented as a finite state automaton.



a) Viability domain, D, and a possible convex partition



b) Visibility graph (topological map) for the convex partition of D

Figure 8: Example of the convexification of a viability domain

Without lack of generality, the sequence of convex elements in the partition of D can be substituted by a partial covering, $D_s = \cup_i D_{s_i}$, with each element, D_{s_i}, a convex region of specific shape, e.g., a sphere. To avoid introducing forbidden regions in the D (which could endanger the mission), the covering must verify $D_s \subseteq D$. Figure 9 illustrates a possible partial covering for the example in Figure 8-a) with the shadowed areas representing the D_{s_i}, $i = 1, \ldots, 9$. By imposing that each D_{s_i} is strictly contained inside a single Dirichlet region, the partial covering generates the same visibility graph as the set of Dirichlet regions of D (and hence the graph in Figure 8-b) still applies).

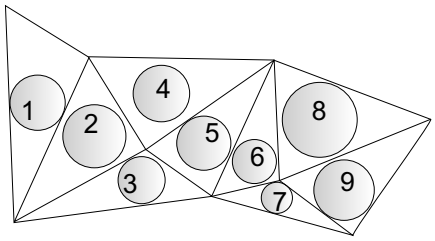


Figure 9: Example of a partial covering formed by the shadowed areas.

This substitution aims at simplifying the computation of $\pi_K(q_n)$ and $T_K(\pi_K(q))$ which, for arbitrary (convex) sets, may require the use of exhaustive search methods. Therefore, for the sake of simplifying the computation of λ, the size of the domain region is reduced, meaning that the number of potential acceptable trajectories is thus reduced.

6 Robot team control

This section extends the hybrid architecture of Section 4 to multiple robots operating within a team.

The team members may compete to have a subset of them (or the whole team) executing the mission eventually coordinating efforts to take advantage of their complementarities. The cooperation (competition and/or coordination) results from the decision (control) policies at each robot these being, in general, mission dependent. The cooperative behavior during any mission can be classified in one of two classes, *formation* and *free* behavior.

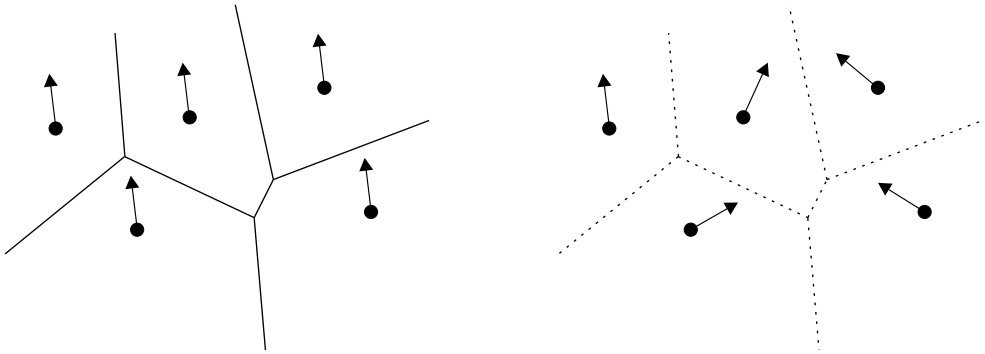
Formation, behavior arises whenever a topological invariant[16] for the space spanned by the team, that

[16] A topological invariant is a mathematical object that maintains its qualities between any two initial and final configurations (e.g., connectedness) under the motion in the space of team trajectories for the assigned mission.

also defines some time invariant relation, can be found. For example, for a team of holonomic robots in a 2D planar C-space, the topological invariant can be the Voronoi diagram of the set of robots (considered as point robots), and hence each robot is identified with a Voronoi polygon. The neighboring relationships between the Voronoi polygons, obtained for the set of robots, is time invariant[17]. In Figure 10-a, the Voronoi diagram, and consequently the neighboring relationships, shown does not change with the motion of the robots.

Free, behavior arises whenever the team motion does not preserve a formation. Figure 10-b, replicates the starting state of Figure 10-a with different robot motion directions. Assuming that the shown motions are constant, there is a time where the trajectories of some of the robots cross each other and hence the neighboring relationships among the set of Voronoi polygons (and, among the set of robots) change[18].

The behavior that is active during some part of a mission is defined by a supervisor controller at each robot and depends on both the specific mission assigned to the team and the capabilities of the team members in terms of kinematics and intra-team communications. Some missions may require that part of the team acts in formation and the remaining in free behavior.



a) Formation motion example b) Free motion example

Figure 10: Motion behaviors for a set of point robots

For the case of formations, the trajectory of a robot has to be contained inside a region defined by the motion (along the mission) of the corresponding Voronoi polygon, which in turn must preserve its neighboring relationships with the other polygons in the team. Therefore, every goal set (K for continuous time version and Δ_K for discrete time) generated along the mission must be contained inside the region spanned by the motion of the Voronoi polygon.

A robot can not generate a goal set that may destroy the neighboring relationships of the formation and hence a negotiation between the teammates has to take place whenever there is the risk that any robot breaks its relationships.

Figure 11 illustrates an example of evolution with the formation of Figure 10-a. The dashed lines indicate the trajectories followed by the robots between the initial and final configurations shown. The Voronoi polygons for the two situations are shown with the neighboring relationships being preserved[19].

[17] Further refinements of this classification are possible, e.g., rigid *vs.* loose formations, when any robot relative positioning constraints must be rigidly (loosely) verified and pre-defined *vs.* spontaneous formations, when the formation is part of the mission specification or it is formed spontaneously.

[18] In some sense, free behavior is an extreme case of a formation behavior.

[19] It is assumed that each robot preserves its neighboring relationships in every point of its trajectory between the initial and final configurations shown.

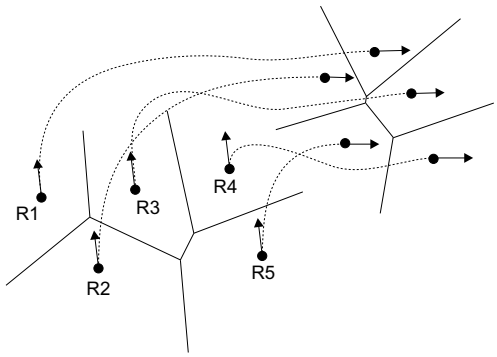


Figure 11: Example of the evolution of a formation (the orientation is not preserved)

Free behavior does not impose any specific inter-robots relationships. For example, in a free behavior the motion of a robot is not spatially constrained so it is free to enter a close neighborhood of any teammate and hence compete for the same physical space. The competition among teammates requires that each robot is able to detect relevant events (e.g., imminent collision). These events (resulting from physical type interactions) are identified with relevant changes in some of the neighboring relationships.

Figure 12 illustrates an example of free evolution. The starting configuration is identical to that in Figure 11 but the robots R_4 and R_5 exchange their neighboring relationships during the course of the mission (the Voronoi polygons are shown for the sake of comparison with Figure 11).

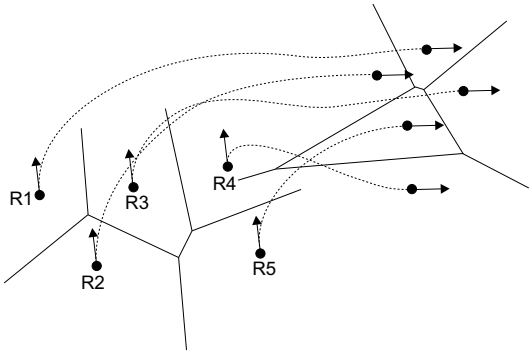


Figure 12: Example of evolution of a team under free motion

6.1 Neighboring relationships

Computing neighboring relationships requires the scanning of the C-space around each robot to check for neighbors and eventually compute the distance to those neighbors. The scanned area around the robot defines an "augmented robot", i.e., an influence region is assigned to the robot. Any physical interaction among the team members (as resulting from the motion of each of them) can be modelled as a problem of interference of these influence regions[20].

Figure 13 shows a team of three point robots, in the plane, each expanded by identical circular areas that

[20]Not every interaction is of physical type, e.g., the exchange of configuration data for the purpose of deciding which sensor to use in a specific situation.

account for some mission relevant feature. The dashed lines indicate the Voronoi polygons for the set of point robots considered[21] Robots R_2 and R_3 are interfering with each other as their areas of influence intersect. The same happens for robots R_1 and R_2.

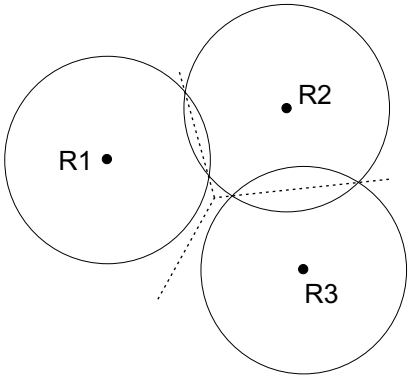


Figure 13: Interference between the regions of influence in a team.

Interference between influence regions generates events that are handled by the supervisor controller at each robot. The meaning of these events depends on the particular mission assigned to the team. For example, some may be enabled by the supervisor for obstacle detection and avoidance purposes.

It is possible to define multiple influence regions around the same robot such that multiple events can be detected. Figure 14 illustrates a situation where each robot has two interference regions. For robots 2 and 3, the inner regions intersect each other; for robots 1 and 2 the inner region of each of them intersects the outer region of the other; for robots 1 and 3 there is no intersection at all.

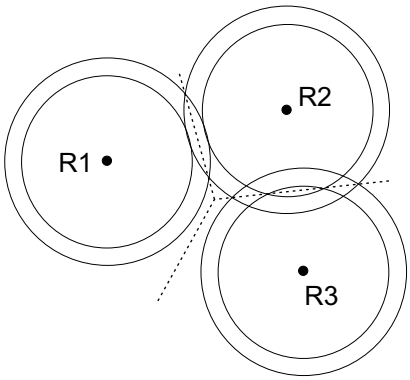


Figure 14: Multiple interference regions in a team.

These combinations of intersections can be identified with different neighboring scenarios among the robots. For the example in Figure 14, robots 2 and 3 might be in a collision avoidance mode (triggered by the intersection of the inner interference regions). The intersection of the outter regions of robots 1

[21] It is worth to note that the intersection of the interference regions of variable radius generates the Voronoi polygon. Recall that the Voronoi diagram, for a set $P \in W$ of points, is defined as the set of points verifying $V = \{v \in W : \max\min\left(d(v,p_i), d(v,p_j)\right),\ i \neq j\}$. The computation of the distance function $d(\cdot,\cdot)$ can be done by "growing" circumferences centered at each p_i and computing their intersections; the distance being the radius at which the first intersection occurs. Any point in the Voronoi diagram belongs to an intersection of, at least, two circumferences of identical radius.

and 2 might just triggered a slowing down mode in robot 1, trying to avoid the more drastic collision avoidance mode (see [Sequeira and Ribeiro, 2001] for a detailed example of a team executing a mission using free behavior and two influence regions per robot).

The negotiation procedure defines the motion strategies handling the violation of the neighboring conditions based on the information exchanged with the teammates. In general, these are mission and robot specific.

For the purpose of this report, an influence region around a robot at each configuration is represented by a set-valued map $R : Q \rightsquigarrow Q$. A neighboring relationship is a logical condition between the values of the R map for the i-th robot and those of the teammates. Table 3 summarizes the team control algorithm, implemented as a FSA at each robot.

given the neighboring relationships for the i-th robot, $R_i : {}^iQ \rightsquigarrow {}^iQ$
compute $R_{ij} = R_i({}^iq) \cap \left\{ {}^jq \right\},\ j \neq i$;
if $R_{ij} = \emptyset$, for any teammate j, then
negotiate with teammates;
otherwise, proceed to goal;

Table 3: Team control algorithm

For simple negotiation algorithms, such as *stop the robot if $R_{ij} = \emptyset$, for any teammate j, otherwise move to goal*, the algorithm in Table 3 is a straighforward two-state FSA. Complex negotiation algorithms may require arbitrarily complex automata.

Among the practical cases of formations in this class are some soccer and military tactical movements.

The composition of any of the FSA in Table 1 and Table 2 with that in Table 3 suggests that multiple formation control problems can be defined share the same control structure.

7 Nonsmooth analysis

This section presents a performance analysis of the hybrid algorithm in Table 1. A similar analysis can be made for the algorithm in Table 2.

Let $F_K(q) = f(q, U) \cap \Delta_K(q)$. A robot modelled by

$$\dot{q} \in F_K(q)$$

and controlled by the algorithm in Table 1 is described by

$$\dot{q} \in \begin{cases} f(q, U_1) & \text{if } \lambda > 0 \\ f(q, U_{21}) & \text{if } \dot{q} = 0 \\ f(q, U_{22}) & \text{if } \dot{q} \perp T_K^\star(q) \\ f(q, U_{23}) & \text{if } \|T_K^\star(q)\| = 0 \\ f(q, U_3) & \text{if } \lambda < 0 \end{cases} \tag{17}$$

where $\|T_K^\star(q)\| = 0$ indicates that the robot is at the boundary of K, i.e., the distance to K measured from the current configuration, q, along a direction in $T_K^\star(q)$ is zero.

A sufficient condition for a mission to end successfully is that $U_1 \cup U_{23} \neq \emptyset$. This means that either $U_1 \neq \emptyset$ and hence the robot is at a configuration where it can approach K or $U_{23} \neq \emptyset$ and hence the

robot has reached a point in the boundary of K, $\mathrm{Bd}(K)$. When $q \rightarrow \mathrm{Bd}(K)$ under motion directions in $F_K(q)$ then $\|\Delta_K(q)\| \rightarrow 0$, meaning that a point in the kernel of $F_K(q)$ is reached. The motion directions generating controls in U_{23} though may not stop the robot still represent a successful mission as the robot reaches a point in the boundary of K. Nevertheless, without lack of generality, $f(q, U_{23})$ can be extended by

$$f(q, U_{23}) = \begin{cases} \mathbf{0} & \text{if } \forall u \in U_{23}, \\ & \quad f(q,u) \ni \mathrm{Ker}(f(q,U)) \\ f(q, U_{23_s}) & \text{otherwise} \end{cases}$$

where $U_{23_s} \subset U_{23}$ is the set of controls such that $\forall u \in U_{23_s}, f(q,u) \in \mathrm{Ker}(f(q,U))$. In practical terms, this extension corresponds to force the robot to stop whenever it touches K (for the sake of simplicity it is assumed that the kinematics allow sudden stops and dynamics side effects are ignored). A similar extension can be made for $f(q, U_1)$. The set of equilibria defined by $\mathrm{Ker}(F_K(q))$ along with those introduced by the extensions of $f(q, U_1)$ and $f(q, U_{23})$ define the conditions for the success of a mission.

The kernels of the remaining terms in the righthand side of (17) define alternative equilibria that correspond to situations where the robot stops before reaching K and hence must be avoided.

Following the conditions in the Lyapunov direct method (see Appendix B for a brief presentation and [Smirnov, 2002, Aubin and Cellina, 1984] for further details) to hold for a system in the form (17), the upper Dini derivative of a Lyapunov function for (17) must be bounded by a negative function for stable equilibria.

Given that $\lambda > 0$ is a sufficient condition for the mission to be successfull, if there is a sequence of controls corresponding to a sequence of switchings among the set valued maps in the righthand side of (17) resulting in a successful mission, then λ^2 is a Lyapunov function for this system.

The computation of the Dini derivative for λ^2, $D^+\lambda^2$, depends on the particular robot. The existence of a Lyapunov function implies that the hybrid system (17) reaches an equilibrium and hence, for the adequate sequence of controls (which is problem dependent), any mission can be successfully completed. Under the conditions of the Lyapunov theorem, it suffices to prove that $D^+\lambda^2$ is bounded to ensure that λ^2 is a Lyapunov function. By construction, λ is bounded as it is piecewise smooth and exhibits only discontinuities of the first order (see the examples in Section 8). Therefore it is lower semi-continuous, and hence of λ^2, which means that it is upper bounded by a negative function, [Bacciotti et al., 2000].

The practical interpretation of the Lyapunov theorem is well known in hybrid systems theory, namely, the envelop of the Lyapunov function is strictly decreasing along the trajectory of the system towards the stable equilibria (see for instance [Lygeros, 1999]).

8 Experimental results

This section presents some simulation results on single robot and team control.

Sections 8.1 to 8.7 present experiments on single 2D holonomic, cart and car robots controlled under the discrete and continuous strategy. No obstacles are considered.

In Sections 8.8 and 8.9 missions to be executed under free and formation behaviors are assigned, respectively, to a team of 2D holonomic robots and a team of cart robots. For the sake of simplicity, in both classes of experiments the robots are dimensionless (point robots). This assumption avoids the use of obstacle avoidance procedures that would mask the performance of the algorithms.

For the continuous approach the goal sets K are balls of radius 0.1. For the discrete approach, the goal sets are always defined by the cone $\Delta_K(q) = K - q$, similar to the example in Section 2.1.

Following the idea in Section 4, to speed up the exhaustive search process for the b.a.p. points and the contingent cone to the goal set, the control sets were restricted to a small number of values. Moreover, the following identification are used

- $q \equiv (x, y)$ for the 2D holonomic robots,
- $q \equiv (x, y, \theta)$ for the cart robot, and
- $q \equiv (x, y, \theta, \phi)$ for the car robot

8.1 Single 2D holonomic mobile robot controlled under the discrete strategy

This section presents experiments with a holonomic 2D robot modelled by

$$\dot{q} = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$$

where $v_1 \in V_1$ and $v_2 \in V_2$ represent the control variables. The problem is to have $\dot{q}$ converging to the set $\Delta_K(q) = K - q$. The set K is a ball centered at $[x_1, x_2] = [0, 0]$.

Given that K is a ball, the b.a.p. is defined by (see Figure 15).

$$\pi_{\Delta_K}(\dot{q}) = -q + 0.1\frac{(\dot{q} - (-q))}{\|\dot{q} - (-q)\|}$$

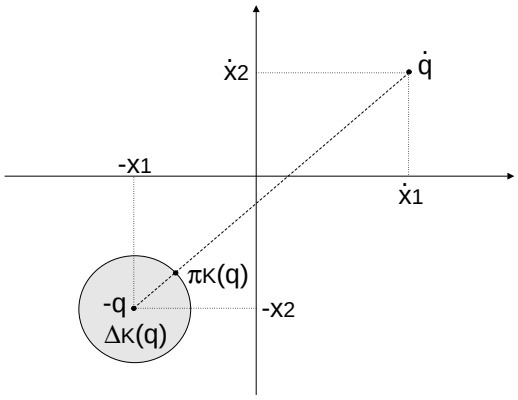


Figure 15: Computing the $\Delta_K(q)$ set and $\pi_{\Delta_K}(\dot{q})$ for the robot at $q = [x_1, x_2]$ and $\dot{q} = [\dot{x}_1, \dot{x}_2]$

The λ_n function is

$$\lambda_n = \langle -q_n + 0.1\frac{\dot{q}_{n+1} + q_n}{\|\dot{q}_{n+1} + q_n\|} - \dot{q}_{n+1}, \dot{q}_{n+1} - \dot{q}_n \rangle, \tag{18}$$

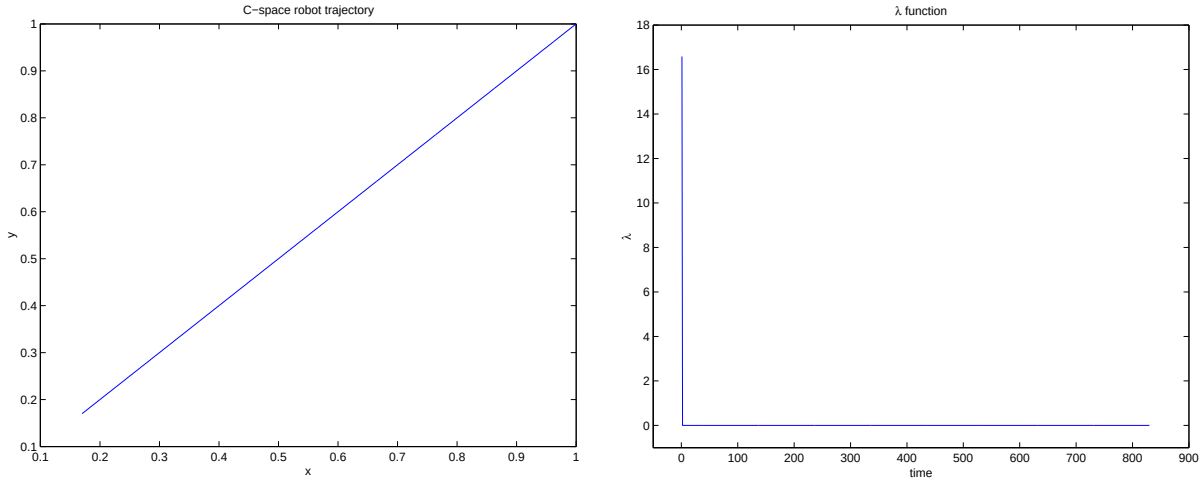
where n indicates the current step.

The control space was restricted to $V_1 \times V_2 = \{-0.1, -0.05, 0, 0.05, 0.1\} \times \{-0.1, -0.05, 0, 0.05, 0.1\}$. Whenever $\lambda > 0$ the controls are such that the decreasing rate of the distance to the goal set is maximized. The condition $\lambda = 0$ may arise either because the two vectors in the inner product of the λ function are orthogonal, because the robot has already reached the goal set or because $\dot{q}_n = \dot{q}_{n+1}$. In the first situation

the distance to the goal set will remain constant[22] whereas in the last situation it may still decrease and hence a reasonable policy is to use the last set of controls that ensured a decrease in the distance to the goal set. Table 4 shows the definition of the corresponding FSA.

Label	FSA state transition events	FSA states (Actions)
1	$\lambda < 0$	$(v_1, v_2) = \arg\max_{V_1 \times V_2}(\lambda)$
2	$\lambda = 0$	keep the previous controls
3	$\lambda < 0$	error condition

Table 4: Control FSA.



a) C-space trajectory b) Time evolution of λ

Figure 16: Holonomic 2D robot (discrete strategy) - Starting configuration $[1, 1]$.

Figures 16 and 17 show the trajectory obtained for a holonomic robot moving in the 2D plane and controlled under the discrete strategy, respectively for the starting configurations $[1, 1]$ and $[2, 1]$.

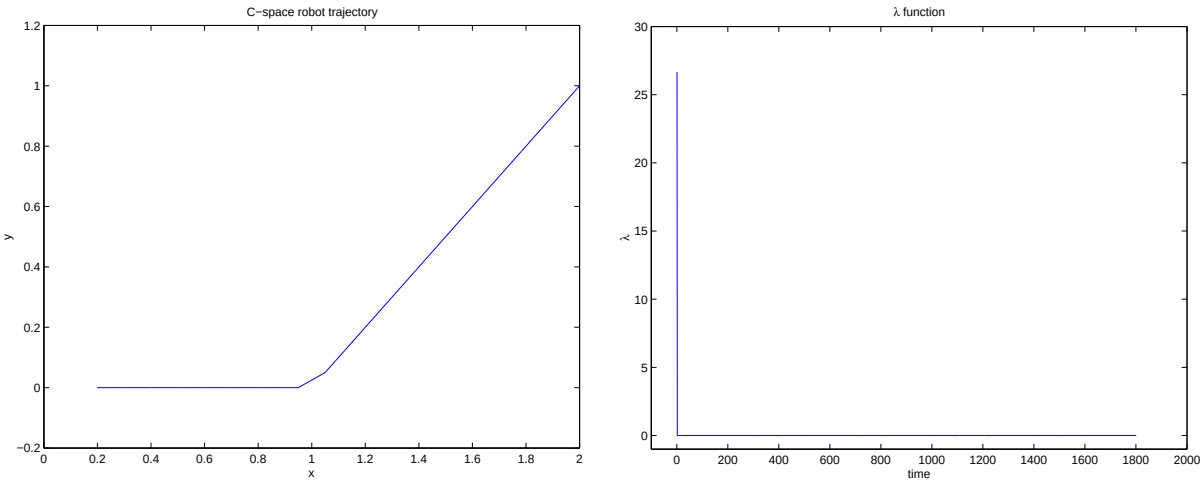
The results in Figure 16 indicate a two phase motion, visible in the λ plot. In the first phase (lasting only for the first step), the controller defines a set of controls such that the motion is straighforward towards the goal set. During the remaining time, the previous set of controls is chosen. The condition $\lambda < 0$ is never reached and hence the corresponding action need not to be further detailed.

The results in Figure 17 show a piecewise linear trajectory with 3 distinctive regions. In the corresponding λ plot only the first phase (at step 1) and the $\lambda = 0$ phases are visible. The peaks corresponding to the trajectory breaking points have values $\lambda \approx 10^{-15}$ and hence are not visible. Note that, for the available set of controls, the straight line trajectory does not maximize the decrease rate for the distance to the goal set. For instance using $V_1 = \{-0.2, \ldots, 0.2\}$ would produce a straight line trajectory similar to the example in Figure 16.

8.2 Single 2D holonomic mobile robot controlled under the continuous strategy

In this section the robot of Section 8.1 is controlled to reach a ball K centered at $[0, 0]$.

[22] This is a local minimum situation not considered in this report.



a) C-space trajectory b) Time evolution of λ

Figure 17: Holonomic 2D robot (discrete strategy) - Starting configuration $[2, 1]$.

The ball assumption for the set K simplifies the computation of $T_K^\star$ to

$$T_K^\star(q) = -\alpha \frac{\pi_K(q) - q}{\|\pi_K(q) - q\|}$$

where $\alpha > 0$ is an arbitrary constant. Given that K is a ball centered at $q_K = [0, 0]$ the b.a.p is given by

$$\pi_K(q) = q_K + 0.1\frac{q - q_K}{\|q - q_K\|} = 0.1\frac{q}{\|q\|}$$

yielding

$$T_K^\star(q) = -\alpha \frac{0.1\frac{q}{\|q\|} - q}{\|0.1\frac{q}{\|q\|} - q\|} = -\alpha \frac{0.1q - q\|q\|}{\|0.1q - q\|q\|\|} = -\alpha \frac{q}{\|q\|}$$

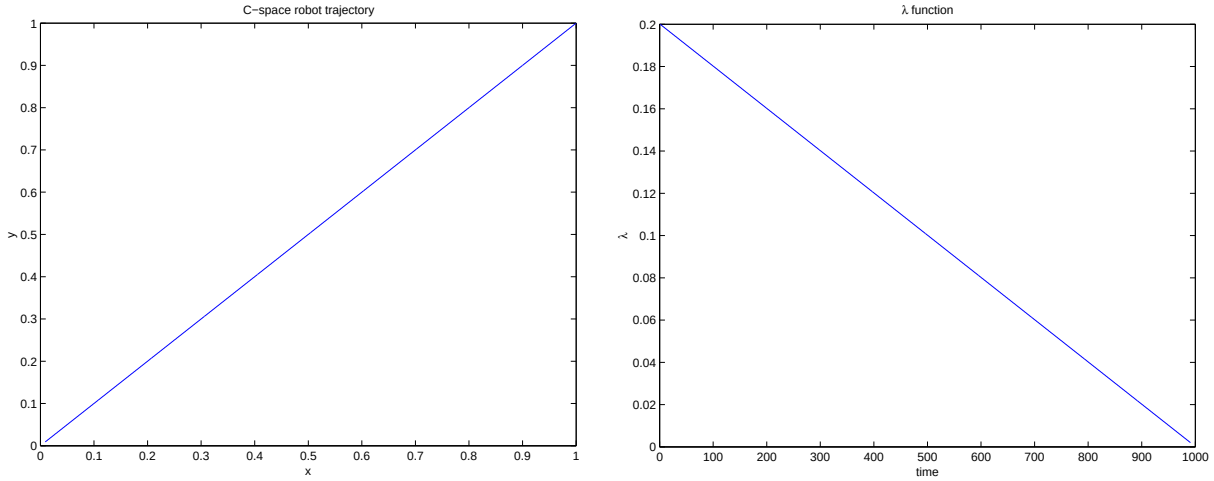
Whenever $\lambda > 0$ there are controls in the chosen control set that ensure the decrease of the distance to the goal set. When $\lambda = 0$, either $\dot{q} = 0$ and the robot is stopped, or the robot reached the goal set or $\dot{q}$ is orthogonal to the direction that ensures a decrease in the distance to the goal set and so the distance will be constant.

The results in Figure 18 obtained with the continuous strategy match those obtained with the discrete strategy, namely the robot reaches the goal set using a straight line trajectory. Unlike the previous experiments, the λ function shows a continuous behavior, not exhibiting any peaks above 0. Note that $\lambda = 0$ is never reached, meaning that the robot never stopped and that the distance is strictly diminishing along the mission.

The results in Figure 19 show a piecewise linear trajectory pattern similar to the trajectory in Figure 17. The use of the alternative control set $V_1 = \{-0.2, \ldots, 0.2\}$ would produce a straight line trajectory joining the initial and final configurations.

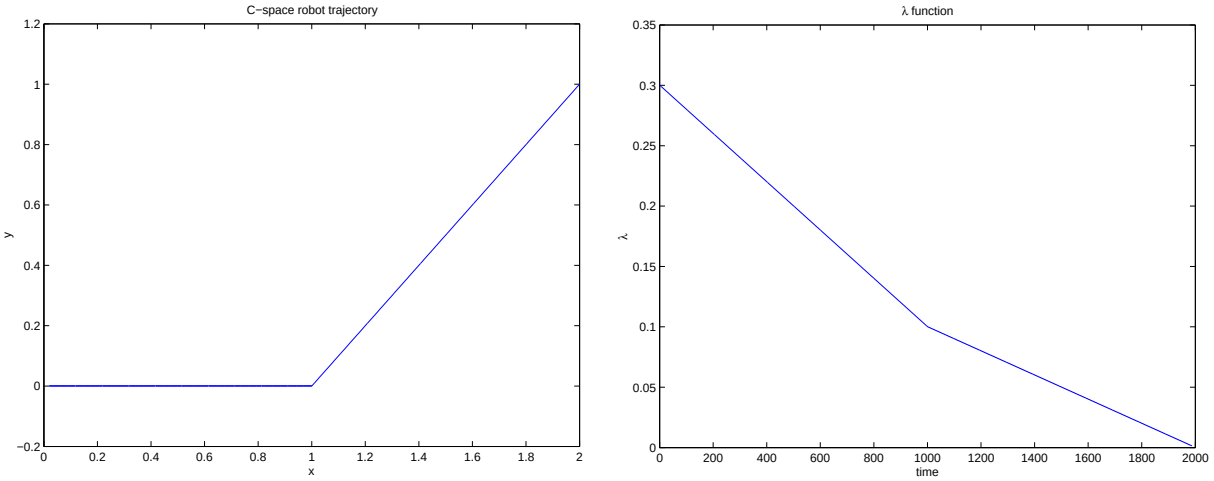
8.3 Single cart robot controlled under the discrete strategy

In this section the control set is restricted to $V \times \Omega = \{-0.1, -0.05, 0, 0.05, 0.1\} \times \{-0.1, -0.05, 0, 0.05, 0.1\}$.



a) C-space trajectory b) Time evolution of λ

Figure 18: Holonomic 2D robot (continuous strategy) - starting configuration $[1, 1]$.



a) C-space trajectory b) Time evolution of λ

Figure 19: Holonomic 2D robot (continuous strategy) - starting configuration $[2, 1]$.

The b.a.p. is given by

$$\pi_{\Delta_K}(\dot{q}) = -q + 0.1\frac{\dot{q}+q)}{\|\dot{q}+q)\|}$$

yielding

$$\lambda_n = \langle -q_n + 0.1\frac{\dot{q}_{n+1}+q_n}{\|\dot{q}_{n+1}+q_n\|} - \dot{q}_{n+1}, \dot{q}_{n+1} - \dot{q}_n\rangle,$$

Table 5 shows the corresponding FSA, where, for the sake of simplifying the description, some functions were introduced. The main structure of the FSA identifies the same 3 situations already identified in the previous examples: $\lambda > 0$, $\lambda = 0$ and $\lambda < 0$. However, the non holonomy of the robot may easily lead to local minima situations characterized by the orthogonality of the two vectors arising in the computation of the λ function. The function $d(\cdot)_{\Delta_K}$ stands for the distance between the argument and the set Δ_K as in Definition 11. The function $\text{copl}(\cdot,\cdot)$ is defined as

$$\text{copl}(a,b) = \angle(\text{Proj}_{xy}(a), \text{Proj}_{xy}(b)_{xy})$$

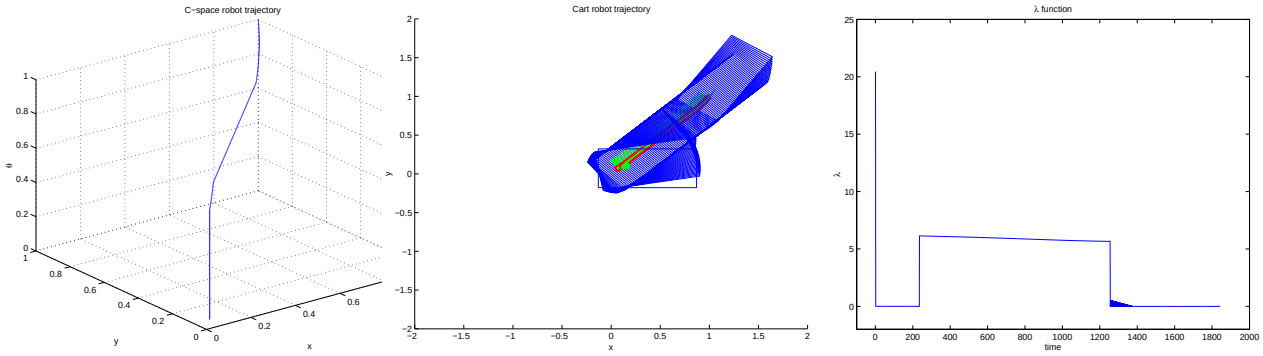
where $\text{Proj}_{xy}(\cdot)$ represents the projection of the argument over the plane $\theta = 0$. The vector $o_n = (\cos(\theta_n), \sin(\theta_n), 0)$ stands for the robot orientation vector at time instant n. The function $\text{sign}(\cdot)$ stands for the usual sign function.

Label	FSA state transition events	FSA states (Actions)
1	$\lambda > 0 \wedge d_{\Delta_K}(\dot{q}_{n+1}) < 1e^{-3}$	$(v,\omega) = (0, -\theta_n/h)$
2	$\lambda > 0 \wedge \text{copl}(\dot{q}_{n+1}, o_n) < 1e^{-3}$	$(v,\omega) = \text{sign}(\langle \dot{q}_{n+1}, o_n\rangle)\max(V), 0)$
3	$\lambda > 0$	$(v,\omega) = (\max_{(v,\omega)\in V\times\Omega}(\lambda)$
4	$\lambda = 0 \wedge d_{\Delta_K}(\text{Proj}_{xy}(\dot{q}_{n+1})) < 1e^{-3}$	$(v,\omega) = (0, -\theta_n/h)$
5	$\lambda = 0 \wedge \|\dot{q}_{n+1} - \dot{q}_n\| = 0 \wedge \max_{(v,\omega)\in V\times\Omega}(\lambda) = 0$	$(v,\omega) = (0, \text{copl}(\dot{q}_{n+1}, o_n)/h$
6	$\lambda = 0 \wedge \|\dot{q}_{n+1} - \dot{q}_n\| = 0 \wedge \max_{(v,\omega)\in V\times\Omega}(\lambda) \neq 0$	$(v,\omega) = \max_{(v,\omega)\in V\times\Omega}(\lambda)$
7	$\lambda = 0 \wedge \|\dot{q}_{n+1} - \dot{q}_n\| > 0$	$(v,\omega) = (0, \text{sign}(\max_{\omega\in\Omega}(\lambda))\max(\Omega))$
8	$\lambda < 0$	error condition

Table 5: Control FSA.

Events 1 to 3 express situations where the controls chosen by the algorithm in Table 1 lead to a decrease in the distance to the goal set. Two special situations, defined by events 1 and 2, are identified to avoid the robot being trapped in local minima: (i) when the robot requires only a rotation to reach the goal set, expressed by event 1, and (ii) when the robot requires only a translation ro reach the goal set, expressed by event 2. In both these situations the chosen controls are ignored and the specific controls shown in the Table are used instead. Event 3 expresses the situation where the controls used are the ones computed by the algorithm in Table 1.

Events 4 to 7 express situations where there is no warranty that the distance will decrease and hence it is necessary to further characterize the situations. Event 4 expresses the situation where the robot requires only a rotation to reach the goal set. Event 5 expresses the situation where the controls should stay as before, since $\dot{q}_{n+1} = \dot{q}_n$, but this leads to zero controls due to the discrete control space used (to improve exhaustive search). In this situation a pure rotation to force the alignment of the robot with the direction of the goal set is used. Event 6 expresses a similar situation to Event 5 but without the constraint imposed by the discrete control space and hence the controls chosen by the algorithm in Table 1 are used. Event 7 indicates that the two vectors in the computation of λ are orthogonal. The action is just to "jump", through a pure rotation, to another region of the C-space where there is the possibility that adequate controls can be chosen.

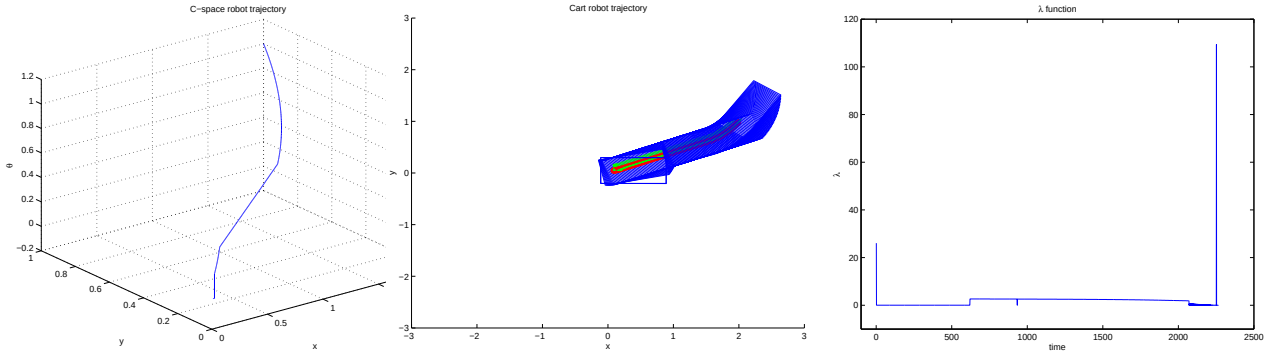


a) C-space trajectory b) Trajectory in the plane $\theta = 0$ c) Time evolution of λ

Figure 20: Single cart robot (discrete strategy) - starting configuration $[1, 1, 1]$.

Figures 20 to 22 show the trajectories of the cart robot under the control of the discrete strategy, respectively for the starting configurations $[1, 1, 1]$, $[2, 1, 1]$ and $[2, 1, 2]$.

Figure 20 shows a piecewise smooth curve with, roughly, 4 trunks. The first trunk can be identified in the λ plot from time 0 to, roughly, 200; the second trunk corresponds to the interval 200 to, roughly, 1200; the third trunk is represented by the small oscilations in the interval 1200 to 1400; the fourth trunk corresponds to the remaining of the mission.



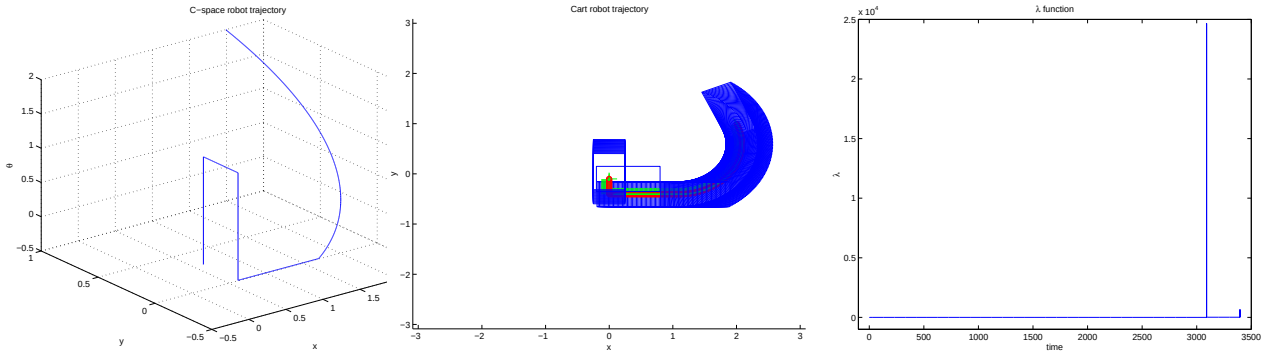
a) C-space trajectory b) Trajectory in the plane $\theta = 0$ c) Time evolution of λ

Figure 21: Single cart robot (discrete strategy) - starting configuration $[2, 1, 1]$.

Figure 21 shows the result of a mission differing from that in Figure 20 in the starting configuration of the robot. The trajectory is again piecewise differentiable and the main trunks can be identified both in the 3D trajectory and in the λ evolution. The large peak in λ at the end of the mission is caused by the pure rotation visible in the planar trajectory.

Figure 22 shows the results for a mission with a different starting configuration. The different trunk visible in the trajectory are masked in the λ plot due to the large values involved. However, the two pure rotations shown in the 3D trajectory plot are identified in the λ plot by the two peaks around the time instants 3100 and 3400.

As in the previous experiments, it was not necessary to detail the condition $\lambda < 0$.



a) C-space trajectory b) Trajectory in the plane $\theta = 0$ c) Time evolution of λ

Figure 22: Single cart robot (discrete strategy) - starting configuration $[2, 1, 2]$.

8.4 Single cart robot under continuous control strategy

The conjugate cone is given by

$$T_K^\star(q) = -\alpha \frac{\pi_K(q) - q}{\|\pi_K(q) - q\|} = -\alpha \frac{q}{\|q\|}$$

yielding

$$\lambda = \langle [\cos(\theta)v, \sin(\theta)v, \omega] \, , \, \frac{\alpha}{\|[x, y, \theta]\|} [-x, -y, -\theta]' \rangle \tag{19}$$

Figures 23 to 25 show the trajectories of the cart robot under the control of the continuous strategy, respectively for the starting configurations $[1, 1, 1]$, $[2, 1, 1]$ and $[2, 1, 2]$. Table 6 shows the corresponding FSA. Event 1 expresses the situation in which the controls chosen by the algorithm in Table 1 are used. The small, $1e^{-3}$, threshold used avoids excessive slow motion that occurs if $\dot{q}$ is too small. The specific kinematics of the cart robot allows the separation of two $\lambda = 0$ situations. From expression (19), whenever the function

$$p(q) = -cos(\theta)x - \sin(\theta)y$$

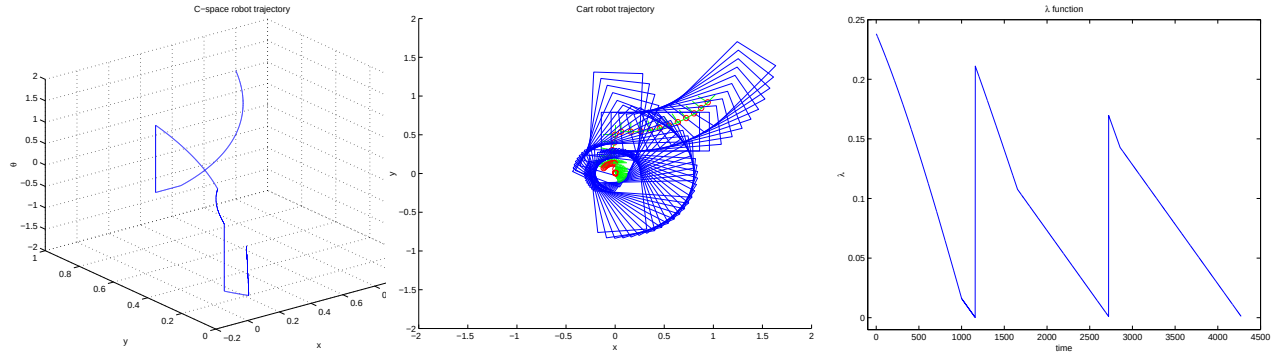
vanishes the system is loosing a degree of freedom in the control space, as there are no linear velocity control (v) that avoids a $\lambda = 0$ situation if it is already $\omega = 0$. Also, if $\theta = 0$ the angular velocity (ω) is no longer useful to control as hence the linear velocity must be forced to a value that keeps λ outside any undesired region. This situation is expressed by Event 2. Similarly, Event 3 captures the loosing of the ω degree of freedom in the control space.

Label	FSA state transition events	FSA states (Actions)
1	$\lambda > 1e^{-3}$	$(v, \omega) = \max_{(v,\omega) \in V \times \Omega}(\lambda)$
2	$\lambda \geq 0 \wedge p(q) < 1e^{-3}$	$(v, \omega) = (0, -\text{sign}(\theta)\pi/2h)$
3	$\lambda \geq 0 \wedge \lvert\theta\rvert < 1e^{-3}$	$(v, \omega) = (\text{sign}(p(q)) \max(V), 0)$
4	$\lambda < 0$	error condition

Table 6: Control FSA.

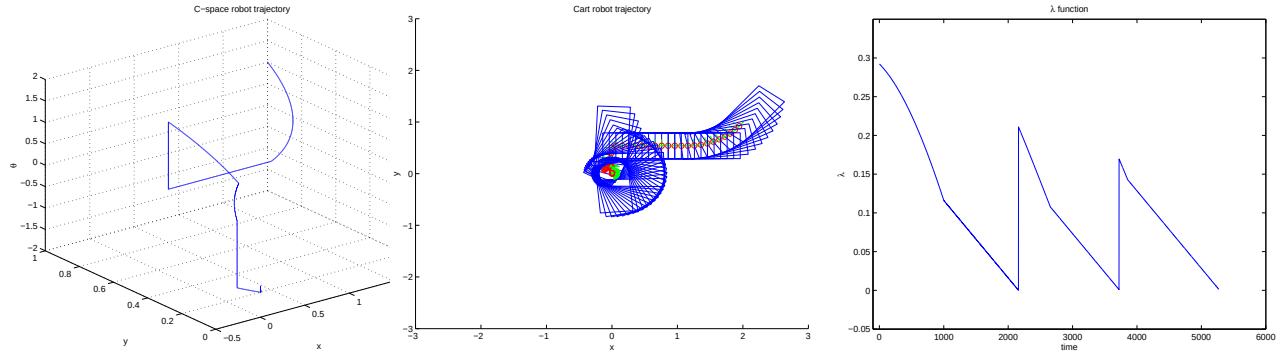
Figures 23 to 25 show the results for three missions differing in the starting locations. For all the three mission, the trajectories follow a similar pattern, identified in the similar saw-tooth behavior of the λ function. The pure rotations identified in the 3D trajectories by the vertical segments have identified in

the "jumps" shown in the λ plots (note that the rotations at the end of the mission are mapped into $\lambda = 0$).



a) C-space trajectory b) Trajectory in the plane $\theta = 0$ c) Time evolution of λ

Figure 23: Single cart robot (continuous strategy) - starting configuration $[1, 1, 1]$.



a) C-space trajectory b) Trajectory in the plane $\theta = 0$ c) Time evolution of λ

Figure 24: Single cart robot (continuous strategy) - starting configuration $[2, 1, 1]$.

As in the previous examples, the condition $\lambda < 0$ need not to be further detailed.

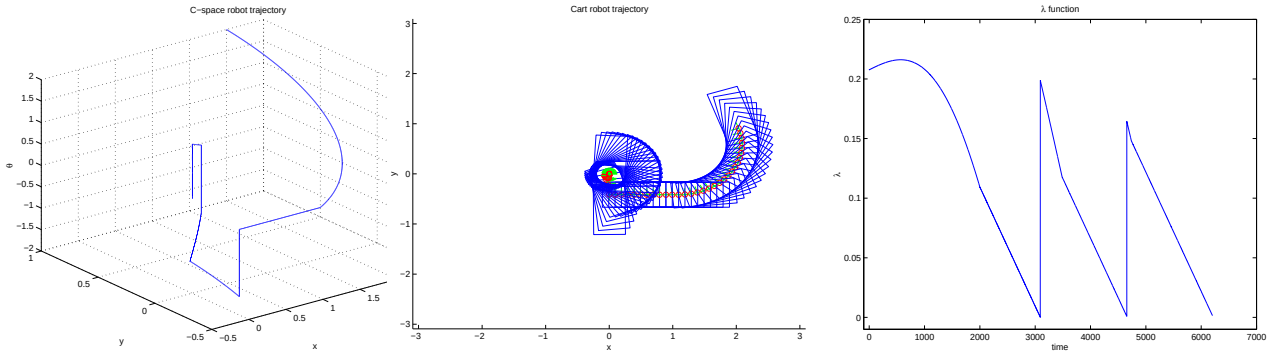
8.5 Two-link serial manipulator

In this section a RR planar manipulator is controlled to have its gripper reaching a goal set centered at point $[x, y] = [0, 1.5]$.

The manipulator model is taken from the kinematics (see Figure 26, where θ_1 and θ_2 are the joint variables of the first and second links, respectively).

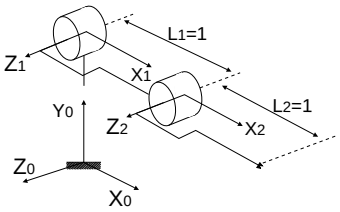
Given that the robot is holonomic, the solution is straightforward and can be achieved in one step if the controller is allowed to search the whole control space. The control values are obtained after the inverse kinematics procedure applied to a workspace region around the gripper defined by[23] $x_{\text{ref}} \in [x - 0.1 : 0.01 : x + 0.1]$, $y_{\text{ref}} \in [y - 0.1 : 0.05 : y + 0.1]$.

[23] Matlab^{TM} notation is used.



a) C-space trajectory b) Trajectory in the plane $\theta = 0$ c) Time evolution of λ

Figure 25: Single cart robot (continuous strategy) - starting configuration $[2, 1, 2]$.



$$q = \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} L_1 \cos(\theta_1) + L_2 \cos(\theta_1 + \theta_2) \\ L_1 \sin(\theta_1) + L_2 \sin(\theta_1 + \theta_2) \end{bmatrix} \tag{20}$$

Figure 26: The RR planar manipulator

The b.a.p. is given by

$$\pi_K(q) = [0, 1.5] + 0.1 \frac{q - [0, 1.5]}{\|q - [0, 1.5]\|}$$

yielding

$$\lambda_n = \langle \pi_K(q_{n+1}) - q_{n+1}, q_{n+1} - q_n \rangle$$

Figure 8.5 shows a stick diagram of the trajectory performed by the manipulator and the corresponding trajectory for the λ_n function. The gripper starts at workspace position $q = [2, 0]$. The trajectory of the gripper exhibits, roughly, three sections: a straight line followed by a curve section followed by a horizontal straight line. The behavior of λ_n matches that of the gripper xy trajectory and indicates that at each time the range of available controls was not wide enough for the gripper to move in a single straight line directly to the goal set.

8.6 Single car robot

In this section a car-like robot is required to reach a goal set centered at point $\dot{q}_K = [1, 1, 1, 0]$ in its velocity space.

The car-like robot model is taken from its differential kinematics with notation explicit from Figure 28. This is a specially interesting example as only 2 controls are used to steer a robot in a 4-dimensional space.

The b.a.p is given by

$$\pi_{\Delta_K}(\dot{q}) = \dot{q}_K + 0.1 \frac{\dot{q} - \dot{q}_K}{\|\dot{q} - \dot{q}_K\|}$$

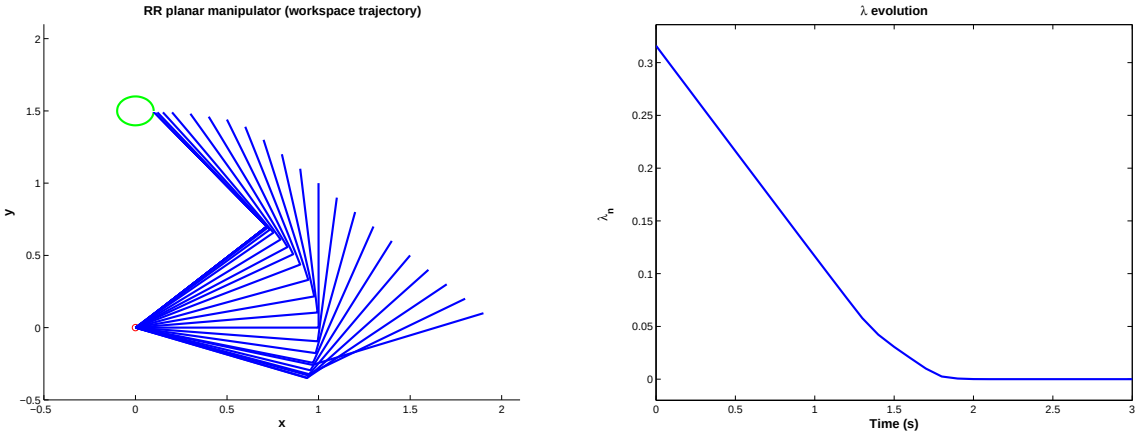
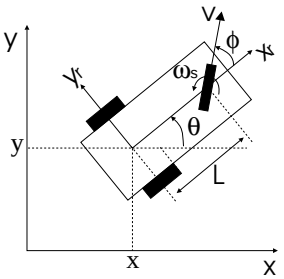


Figure 27: Trajectories performed by the RR planar manipulator



$$\dot{q} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} \cos(\theta)\cos(\phi) & 0 \\ \sin(\theta)\cos(\phi) & 0 \\ \frac{1}{L}\sin(\phi) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega_s \end{bmatrix} \tag{21}$$

Figure 28: The car-like mobile platform

yielding

$$\lambda_n = \langle \pi_{\Delta_K}(\dot{q}_{n+1}) - \dot{q}_{n+1}, \dot{q}_{n+1} - \dot{q}_n \rangle$$

As stated in Section 4, the system jumps to C_1 whenever there are controls satisfying $\lambda_n > 0$. In such case the controls yielding the maximal λ_n are chosen. Whenever there are only controls leading to jumps to C_{2_1} or C_{2_2}, the robot model (21) can be used to give some insight on the design of the actions applied at each of these states. In this problem it is worth to point out that each of the control variables affects orthogonal subspaces of the velocity space. When the robot jumps to one of the states C_{2_1} or C_{2_2} predefined maneuvers along each of these subspaces are used.

The car robot has two control variables v, the linear velocity, and ω_s, the steering rate of the steering wheel. The projection along v of the velocity space is the subpace spanned by $\text{Proj}_v(\dot{x}, \dot{y}, \dot{\theta}, \dot{\phi}) = \left\{\dot{x}, \dot{y}, \dot{\theta}\right\}$, whereas the projection along ω_s is the subspace spanned by $\text{Proj}_{\omega_s}(\dot{x}, \dot{y}, \dot{\theta}, \dot{\phi}) = \left\{\dot{\phi}\right\}$.

When $C_1 = \emptyset \ \wedge \ C_{2_1} \neq \emptyset$ the controls are defined by,

$$[v, \omega_s] = \begin{cases} [\max\{V\}, 0] & \text{if } \|\text{Proj}_v(\dot{q}_{n+1} - \dot{q}_n)|_{\lambda_n=0}\| = 0 \wedge \\ & \|\text{Proj}_{\omega_s}(\dot{q}_{n+1} - \dot{q}_n)|_{\lambda_n=0}\| \neq 0 \\ [0, \max\{\Omega\}] & \text{if } \|\text{Proj}_v(\dot{q}_{n+1} - \dot{q}_n)|_{\lambda_n=0}\| \neq 0 \wedge \\ & \|\text{Proj}_{\omega_s}(\dot{q}_{n+1} - \dot{q}_n)|_{\lambda_n=0}\| = 0 \end{cases} \tag{22}$$

Controls (22) can be interpreted as forcing the projections over each of the controls to be linearly dependent of the corresponding projections of the goal set location[24]. If both the projections are linearly dependent of the goal set the solution trivial. Therefore, even if this dependence leads to an increase in the distance between $\dot{q}$ and the goal set and hence to $\lambda_n < 0$, the following maneuvering tends to become simplified.

When $C_1 = \emptyset \ \wedge \ C_{2_1} = \emptyset \ \wedge \ C_{2_2} \neq \emptyset$ the controls are defined by,

$$\begin{aligned} v &= \begin{cases} \max\{V\} & \text{if } \text{Proj}_v^2 > 0 \\ 0 & \text{if } \text{Proj}_v^2 = 0 \\ \min\{V\} & \text{if } \text{Proj}_v^2 < 0 \end{cases}, \\ \omega_s &= \begin{cases} \max\{\Omega\} & \text{if } \text{Proj}_{\omega_s}^2 > 0 \\ 0, & \text{if } \text{Proj}_{\omega_s}^2 = 0 \\ \min\{\Omega\} & \text{if } \text{Prog}_{\omega_s}^2 < 0 \end{cases} \end{aligned} \tag{23}$$

where,

$$\text{Proj}_v^2 = \langle \text{Proj}_v(\dot{q}_{n+1} - \dot{q}_n) \text{Proj}_v(\pi_K(\dot{q}_{n+1}) - \dot{q}_{n+1}) \rangle|_{\lambda=0}$$

$$\text{Proj}_{\omega_s}^2 = \langle \text{Proj}_{\omega_s}(\dot{q}_{n+1} - \dot{q}_n) \text{Proj}_{\omega_s}(\pi_K(\dot{q}_{n+1}) - \dot{q}_{n+1}) \rangle_{\lambda=0}$$

The controls (23) are designed to counter the effect of the sign of the inner product between projections over the same control variable. This strategy can only produce negative λ_n values (if controls leading to a positive λ_n exist they would have been found while computing C_1). Since the maximal control values are used, these points in C_3 tend to be far away the goal set thus simplifying the switching to C_1 immediately after this step.

[24] Alternatively, the corresponding vectors are aligned.

Figure 29 shows the results using the above controller (the robot starts at the origin of the behavioral space). Two points in C_3, corresponding to the negative λ_n values are shown. The first point results from a transition from C_{2_1} while the second one is generated from a transition from C_{2_2}. With the exception of the final point, the remaining points are in C_1, as the smooth decrease in the λ_n is clearly visible. After the first point in C_3 is reached, the envelop of the Lyapunov function, λ^2, is strictly decreasing as mentioned in Section 4.

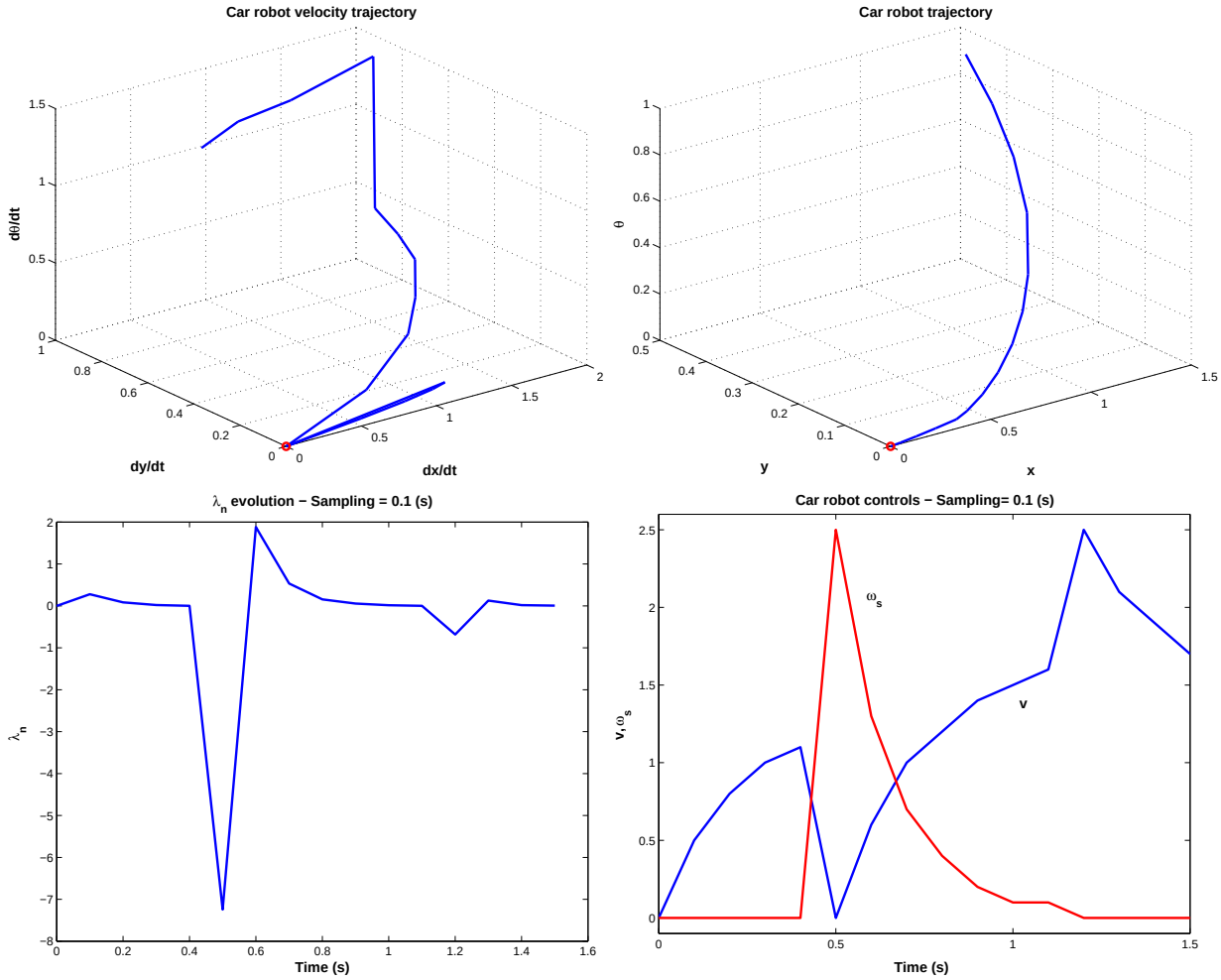


Figure 29: Velocity and position trajectories for the car robot

No formal proof is given that the controls (22) and (23) are sufficient to have the whole system verifying the Lyapunov property. However, the above interpretation for each of them supports the conjecture that the points reached in C_3 are such that either $f(q, U) \cap \Delta(q) \neq \emptyset$ (as when $\dot{q}$ is such that $\Delta(q)$ can be reached just by using constant controls) or the distance between the $f(q, U)$ and $\Delta(q)$ can be brought to 0 by jumping to C_1 and waiting for the jump to C_{2_3}.

8.7 Single car robot in reduced space

This section presents two classes of experiments on a car robot using the continuous algorithm (Table 2). The control is performed

- in the full C-space and
- only the xy workspace is used.

The robot starts from configuration $[0, 0, 0, 0]$ and the goal region K is centered at $q_K = [1, 1, 0, 0]$.

The conjugate cone to K at a point q is given by

$$T_K^\star(q) = q_K + 0.1\frac{q - q_K}{\|q - q_K\|}$$

The computation of the partition for U in Table 2 may be done using exhaustive search techniques to evaluate the projection map and avoid solving inverse dynamics problems, e.g., computing u from $\dot{q}$. The control space is discretized into a small number of admissible values thus allowing the exhaustive search technique. At each instant, a compact set of controls, $V \times \Omega_s = [-1 : 0.5 : 1] \times [-1 : 0.1 : 1]$, is used to search for U_1. The steering wheel angle is constrained by $-\pi/2 \leq \phi \leq \pi/2$. In all the experiments, whenever $\lambda = 0$ the controls are kept as before.

Figure 30 illustrates the results obtained when the algorithm is applied in the configuration space $[x, y, \theta, \phi]$.

To reach position $[x, y] = [1, 1]$ with null orientation and subject to the considered compact control set requires a highly complex trajectory. The first part is smooth and corresponds to the minimization of the distance along the x coordinate. After a re-orientation maneuver around $[x, y] = [1, 0]$, the robot proceeds to minimize the y coordinate. This maneuver can be identified in the λ plot around the instant of the first 0 crossing. The final approach to the goal set is made through a number of complex maneuvers, with the robot moving back and forth to reach a suitable orientation. The λ plot shows that for the most of the time controls in U_3 are used. These are mainly needed for the maneuvering shown in the xy plot. After a brief initial period where λ decreases to 0 (corresponding to the initial horizontal part of the xy trajectory) the system enters a heavy maneuvering phase. The final part of the mission corresponds again to a decrease of λ to 0. From the plot it is clear that, after the spike near 100 s, the envelop of λ^2 is strictly decreasing, as expected from the Lyapunov theorem (see Section 7).

Figure 31 shows the results when the behavioral space is restricted to $[x, y]$. In this case the control variable is chosen as $u = \cos(\phi)v$. Given the kinematics of the robot, a natural objective when choosing ω_s is to require that $\theta + \phi$ tends to the direction of the goal set. Thus, ω_s is chosen to approach this direction. No bounds were assumed for the controls outside U_1. The λ plot exhibits a smooth behavior decreasing to 0, indicating that no harsh maneuvers were used.

Figure 32 shows the results when controls outside U_1 are bounded, with $|v| \leq 10$ and $|\omega_s| \leq 13$. Although the xy trajectory is almost identical to the one in Figure 31, the behavior of the control system is completely different. During the first part of the experiment (visible as the horizontal region in the distance plot) the robot slowly changes the orientation of the steering wheel after which it proceeds towards the goal set. The initial horizontal trajectory for λ is caused by the reorientation of the steering wheel. Around 4.4 s the plot shows a spike that indicates the beginning of the motion of the robot in the xy plane. Clearly, the behavior of the λ^2 is also upper bounded by a decreasing curve, hence verifying the stability of the equilibrium state corresponding to the end of the mission.

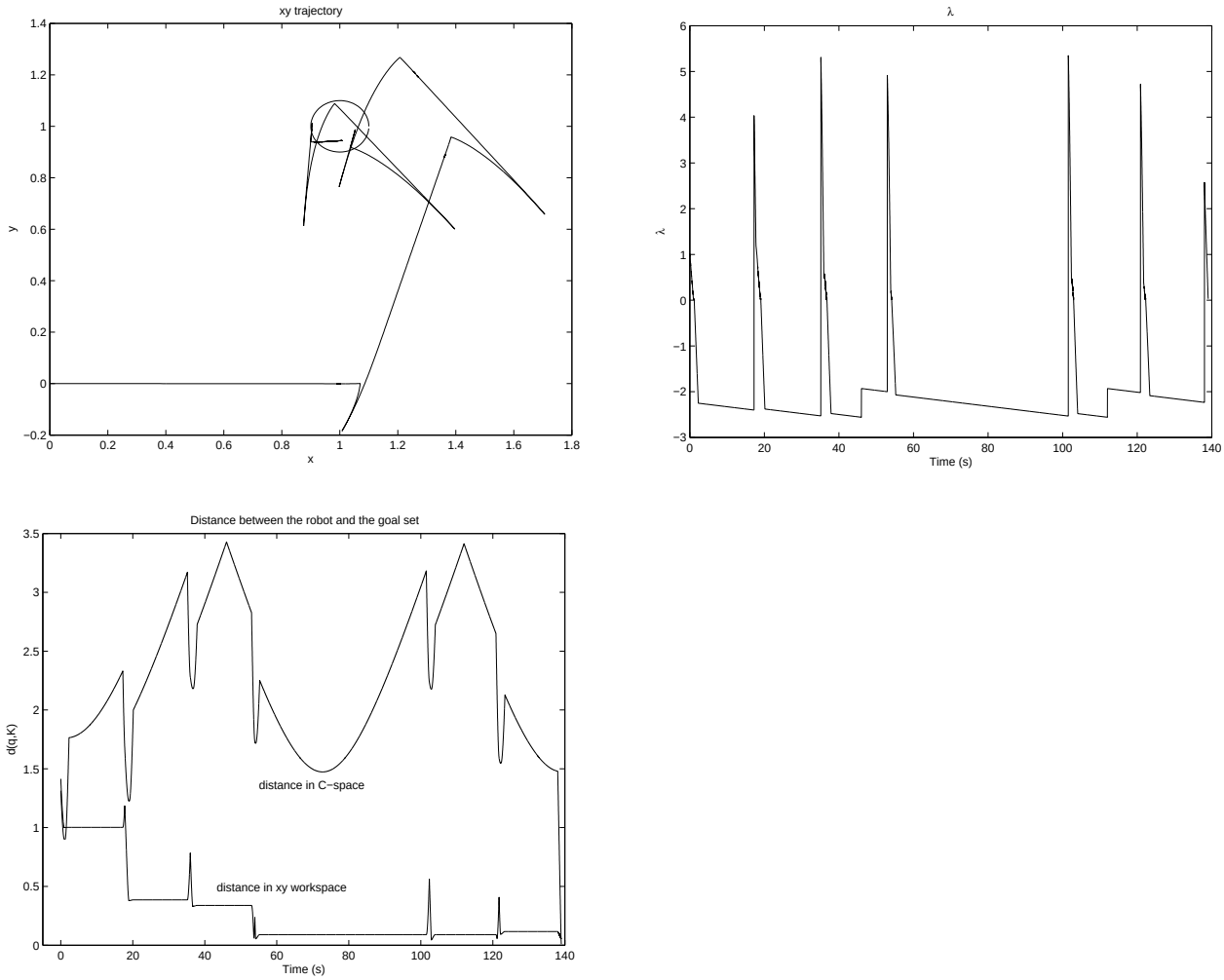


Figure 30: Car control in the (x, y, θ, ϕ) C-space

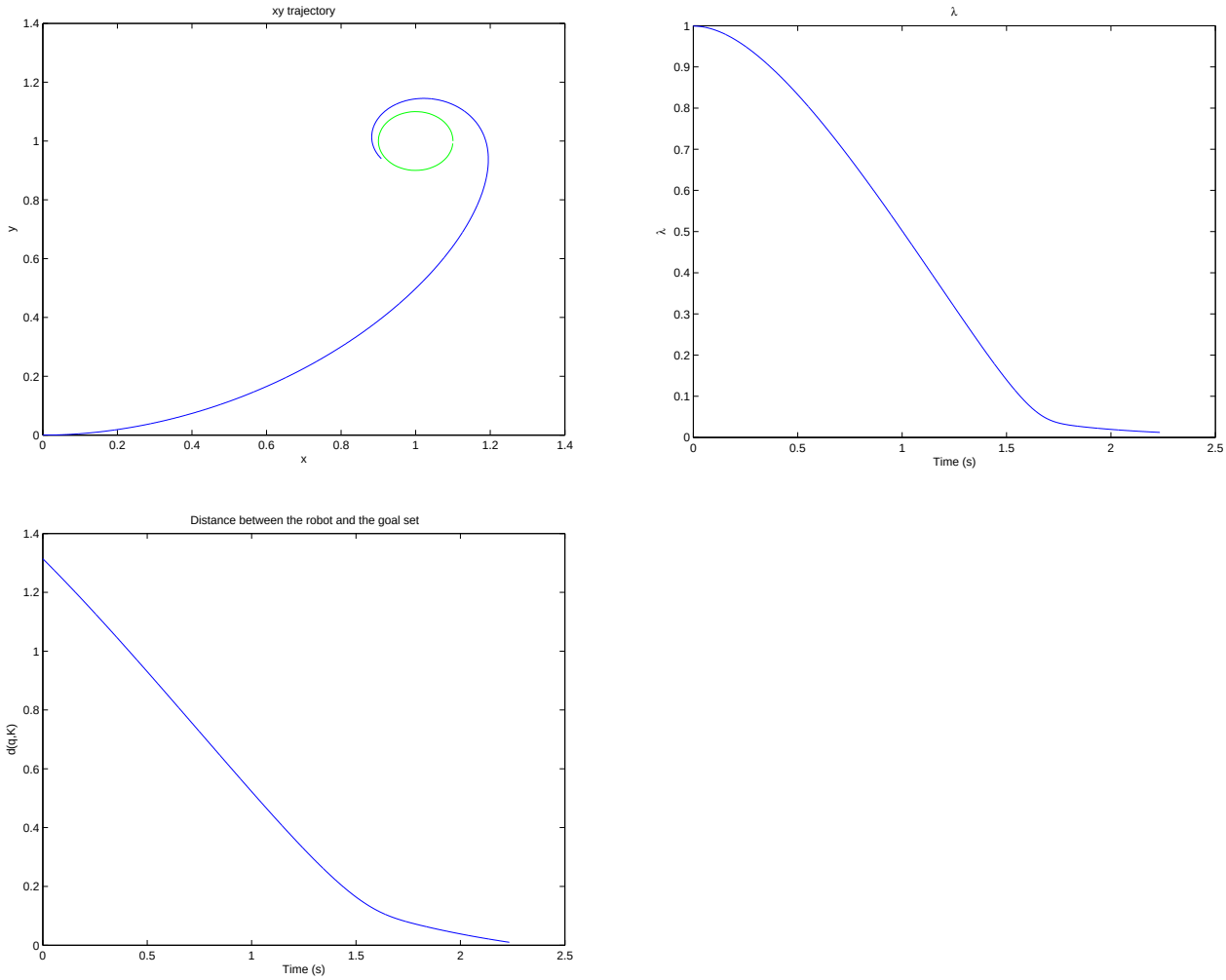


Figure 31: Car control in the (x, y) workspace

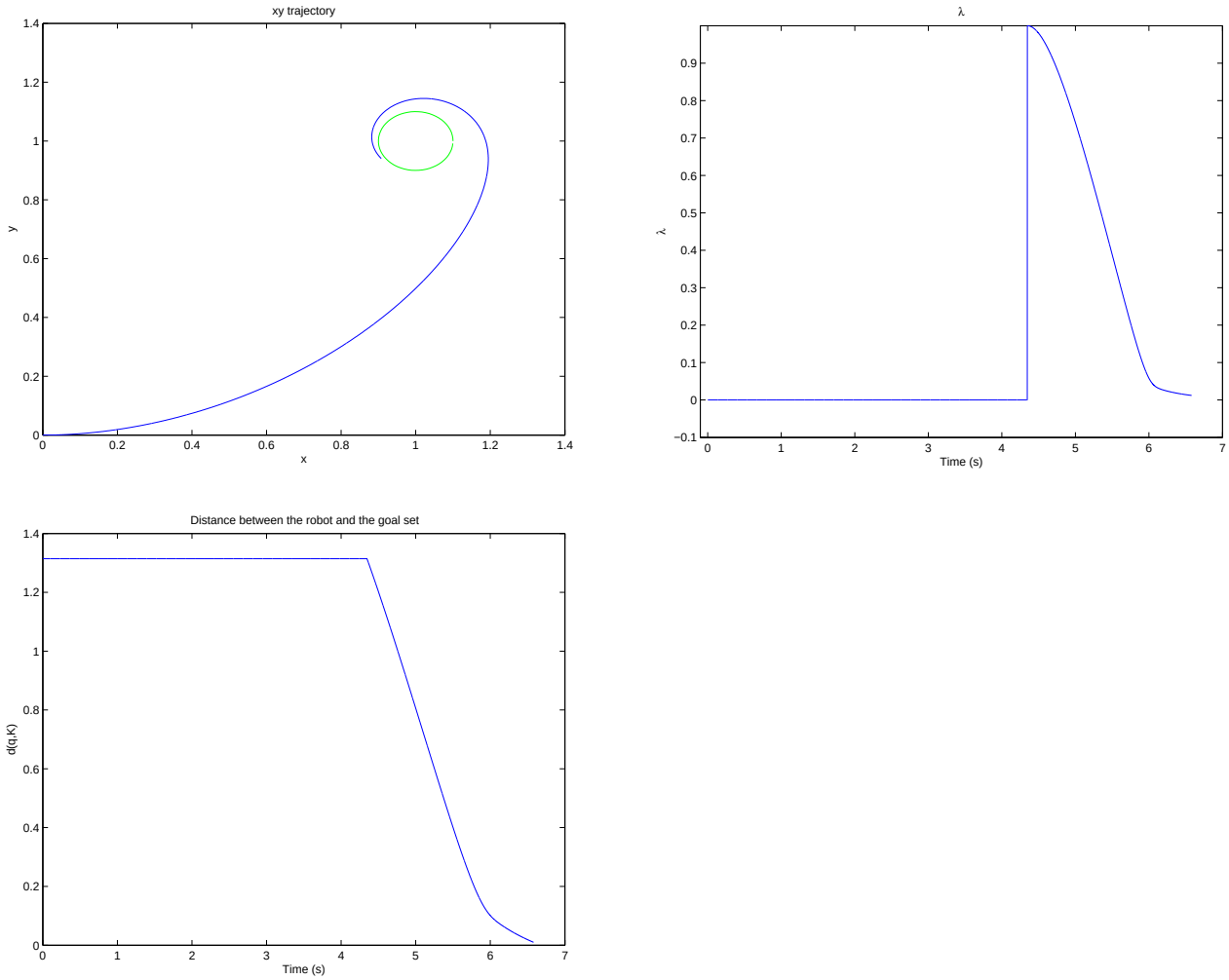
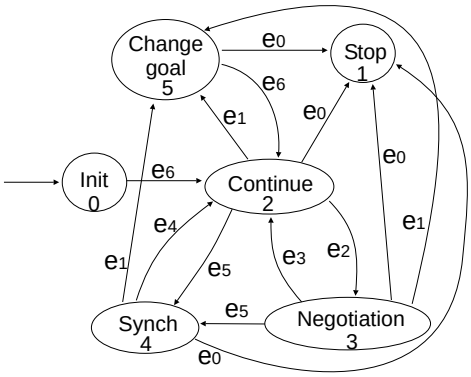


Figure 32: Car control in the (x, y) workspace - Both controls bounded

8.8 Team of 2D holonomic robots

The composition of the FSA in Tables 2 and 3 results in a FSA with the structure shown in Figure 33. The structure is similar for both experiments, differing only in the definition of the events firing the transitions between states.



Label	State	Meaning
0	Init	Initialization procedures
1	Stop	Team goal reached
2	Continue	Proceed towards current goal
3	Negotiate	Formation break
4	Synchronize	Wait for teammates to reach their current goals
5	Change goal	Change for the next goal in the mission

Figure 33: Structure of the FSA composed from the single and team control FSAs

The core states are *continue* and *negotiate*. After the initialization procedures each robot reaches the state *continue*, remaining there as long as no violation of the neighboring conditions occurs (triggering e_2) and its current goal is not yet reached (triggering e_5). The controls used are those defined by the algorithm in Table 2. In practical terms, the state *continue* aggregates a set of states handling all the single robot control motion strategies, i.e., the specific motion strategies generated by controls in each of the ${}^iU$ regions.

Whenever any of the neighboring relationships is violated e_2 is triggered and the control is transfered to the state *negotiate* where it remains until the violation disapears (triggering e_3). In this case the controls in Table 2 are overriden by specific (problem dependent) controls defined bellow for each of the examples.

When a robot reaches its current goal set e_5 is triggered and the control jumps to state *synchronize*. The robot waits in this state for the rest of the teammates to reach their goal sets before chosing another goal set (if any is still left in the mission sequence) by triggering e_1 after what it jumps to *continue* to proceed to the goal.

The goal of the mission is to have the robots reaching the sequence of goal sets with centers defined in Figure 34a while preserving a reference formation. This reference formation is defined from the initial configuration of the robots and the neighboring relationships between the i-th robot and the teammates

defined by

$$R_i = \left\{ q \in {}^iQ : \begin{array}{l} \langle {}^iq_{\text{goal}} - {}^iq, q - {}^iq \rangle \geq 0 \ \wedge \\ 0.1 < d({}^iq, q) < 0.5 \end{array} \right\} \ \cup$$

$$\left\{ q \in {}^iQ : \langle {}^iq_{\text{goal}} - {}^iq, q - {}^iq \rangle < 0 \right\}$$

This relationship defines a circular sector in front of the robot where any teammate is allowed to stay. The region in the back of the robot is also an allowed region. Any robot located outside R_i violates this neighboring condition, forcing this robot to enter a negotiation. While negotiating a robot is stopped.

The algorithm in Table 2 is instanced to that in Table 7.

(the goal velocity sets are always convex)
compute ${}^iU_1 = \left\{ u \in {}^iU : \lambda > 0 \right\}$;
if ${}^iU_1 \neq \emptyset$ choose $u = \arg_{{}^iU_1} \max(\lambda)$
otherwise
compute ${}^iU_2 = \left\{ u \in {}^iU : \lambda = 0 \right\}$;
compute the partition ${}^iU_2 = {}^iU_{21} \cup {}^iU_{22} \cup {}^iU_{23}$
if ${}^iU_{23} \neq \emptyset$ choose any $u \in {}^iU_{23}$
otherwise
if ${}^iU_{21} \cup {}^iU_{22} \neq \emptyset$
choose any $u \in {}^iU_{21} \cup {}^iU_{22}$
repeat until goal is reached

Table 7: Controller for the single 2D holonomic robot

Figure 34b shows the trajectories of the robots, with the goal sets shown as wide circles. The mission is successfully completed.

The trajectories are mainly in straight line due to the holonomy of the robots (a straightforward consequence for controls in U_1) and the motion strategy adopted when in the *negotiate* state. For the most part of the mission each of the robots acts as an isolated agent as no neighboring relationships are violated. When entering the neighborhood of the final goal sets they are forced to shorten the distances among them, resulting in the violation of the neighboring conditions and in the corresponding negotiations. The closest robot to the final goal set has no neighboring conditions violated and hence proceeds to its goal. The remaining two wait for the leader to go away to have their neighboring conditions satisfied and hence proceed also to their goals.

8.9 Team of cart robots

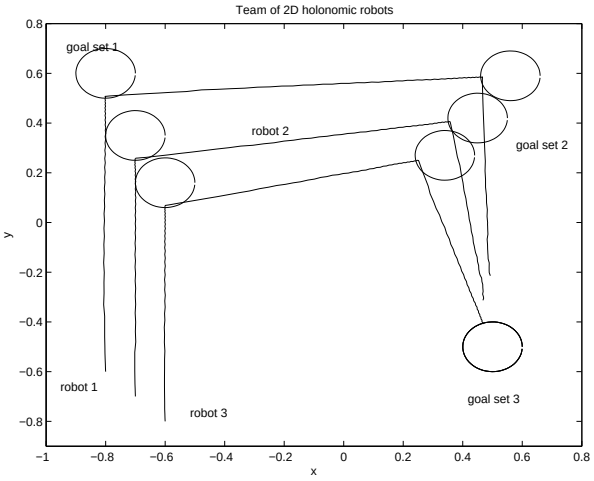
The goal of the mission is to have the robots reaching the goal sets with centers defined in Figure 35a while preserving a reference formation. The neighboring relationships for the i-th robot are defined by

$$R_i = \ \left\{ q \in {}^iQ : d({}^iq, q) \leq 0.5 \right\} \ \cup$$

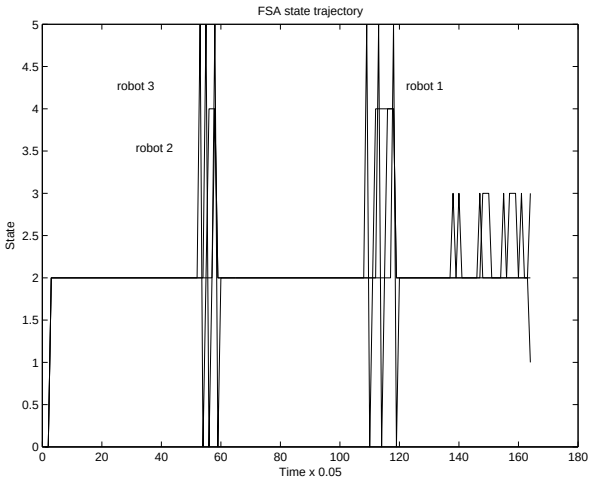
$$\left\{ q \in {}^iQ : \begin{array}{l} d({}^iq, q) > 0.5 \wedge \\ d({}^iq, {}^iq_{\text{goal}}) < d({}^jq, {}^jq_{\text{goal}}), \forall j \neq i \end{array} \right\}$$

	Robot 1		Robot 2		Robot 3
q_{start}	(-0.8,-0.6)	q_{start}	(-0.7,-0.7)	q_{start}	(-0.6,-0.8)
q_{goal_1}	(-0.8,0.6)	q_{goal_1}	(-0.7,0.35)	q_{goal_1}	(-0.6,0.16)
q_{goal_2}	(0.56,0.59)	q_{goal_2}	(0.45,0.42)	q_{goal_2}	(0.34,0.27)
q_{goal_3}	0.5,-0.5)	q_{goal_3}	(0.5,-0.5)	q_{goal_3}	(0.5,-0.5)

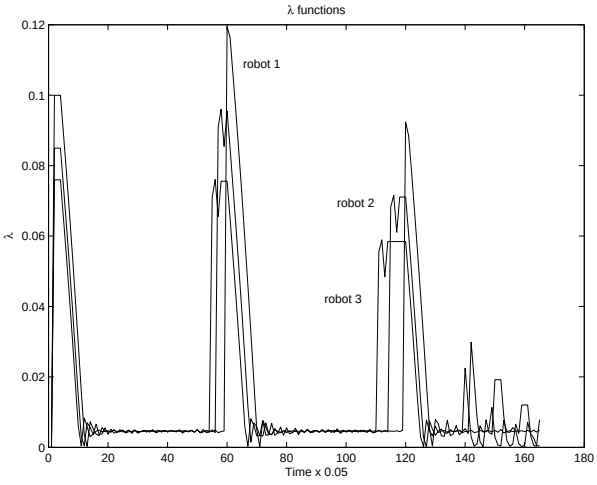
a) Mission definition



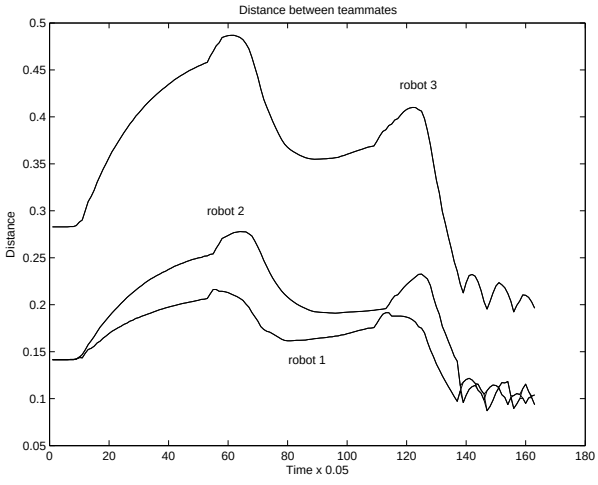
b) Robot trajectories.



c) State evolution



d) Evolution of the λ functions



e) Minimum distances among teammates

Figure 34: Mission definition and results for the 2D holonomic robot team

The first subset in the above relationship defines a region around the robot (within a distance of 0.5) where the relationship is considered verified if at least a teammate is located there. This ensures that if the robots stay close to each other no special negotiation is required and each of them proceeds towards its own goal. If the team is spliting apart and the robot is being left behind then the relationship is also verified and the robot proceeds towards its current goal. This condition ensures that the team does not span arbitrary large regions while moving. While negotiating a robot is stopped.

The control of each single robot follows the algorithm in Table 8. The controls in ${}^iU_{21} \cup {}^iU_{22}$ are divided into two of the basic maneuvers a cart robot can perform: pure translations and pure rotations.

(the goal velocity sets are always convex)
let ${}^iU_1 = \{u \in {}^iU : \lambda > 0\}$;
if ${}^iU_1 \neq \emptyset$ choose $u = \arg_{{}^iU_1} \max(\lambda)$
otherwise
let ${}^iU_2 = \{u \in {}^iU : \lambda = 0\}$;
with ${}^iU_2 = {}^iU_{21} \cup {}^iU_{22} \cup {}^iU_{23}$
if ${}^iU_{23} \neq \emptyset$ choose any $u \in {}^iU_{23}$
otherwise
if ${}^iU_{21} \cup {}^iU_{22} \neq \emptyset$
let $align = ({}^ix - {}^ix_{goal})\cos({}^i\theta) +$
$+({}^iy - {}^iy_{goal})\sin({}^i\theta)$
if $align < 1e^{-2}$
choose $u = ({}^iv_{\text{cte}}, 0)$
elseif $\|({}^ix, {}^iy) - ({}^ix_{goal}, {}^iy_{goal})\| < 1e^{-2}$
choose $u = (0, {}^i\omega_{\text{cte}})$
repeat until goal is reached

Table 8: Controller for the single cart robot

The mission was successfully completed with the plots in Figure 35 showing similar behavior by all the robots in the team. Figures 35b,d show the resulting trajectories in the 3D C-space and in the xy plane, with the goal sets shown as shaded regions in the 3D trajectories plot.

The trajectory plots show intense maneuvering as a result of the nonholonomy of the robots which also leads to the intense negotiations shown in plot 35c. This matches the behavior of the λ curves, with the spikes indicating the motion strategy changes induced by the negotiation state at each robot and the following descending slopes indicating the motion under the *continue* state.

When negotiating, each robot is stopped this being a blocking strategy. Using only the inter-robot distances would result in a blocking strategy, i.e., if all the robots violate simultaneously the neighboring condition the team stops. Adding the distance to the goal set condition allows that the farest robot to move out of the blocking formation heading towards its goal as a sort of leading robot. As this robot approaches its goal another robot yet blocked is freed and starts moving towards its own goal set, thus releasing the formation from the blocking condition.

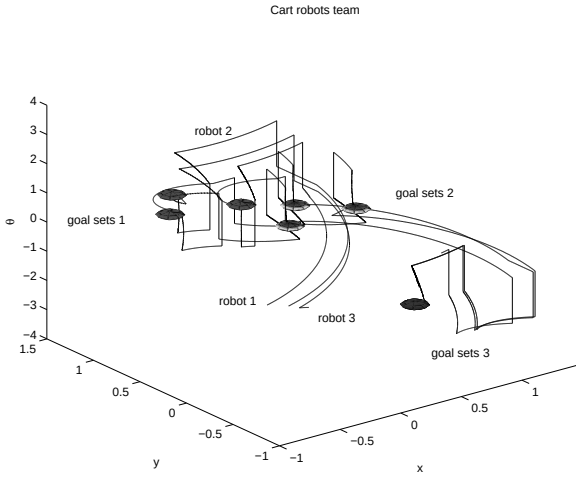
9 Conclusions

This report presented a geometric approach to the control of robots, supported on

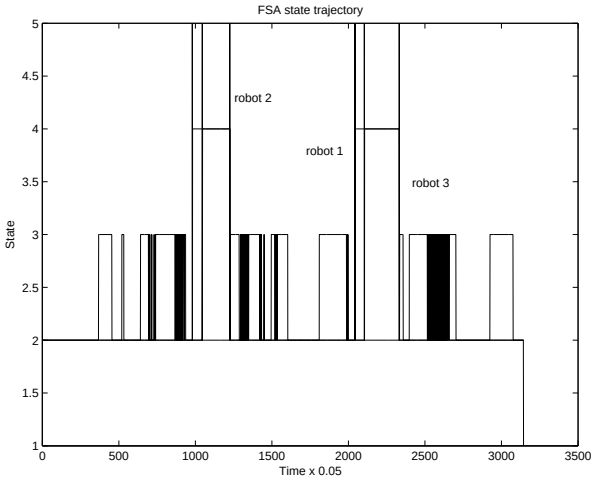
- a projection map defined from basic properties of Hilbert spaces, and

	Robot 1		Robot 2		Robot 3
q_{start}	$(-0.8, -0.6, 0)$	q_{start}	$(-0.7, -0.7, 0)$	q_{start}	$(-0.6, -0.8, 0)$
q_{goal_1}	$(-0.31, 0.31, \pi/2)$	q_{goal_1}	$(-0.56, 0.73, \pi/2)$	q_{goal_1}	$(-0.79, 0.45, \pi/2)$
q_{goal_2}	$(0.76, 0.47, 0)$	q_{goal_2}	$(0.22, 0.48, 0)$	q_{goal_2}	$(0.48, 0.76, 0)$
q_{goal_3}	$(0.5, -0.5, -\pi/2)$	q_{goal_3}	$(0.5, -0.5, -\pi/2)$	q_{goal_3}	$(0.5, -0.5, -\pi/2)$

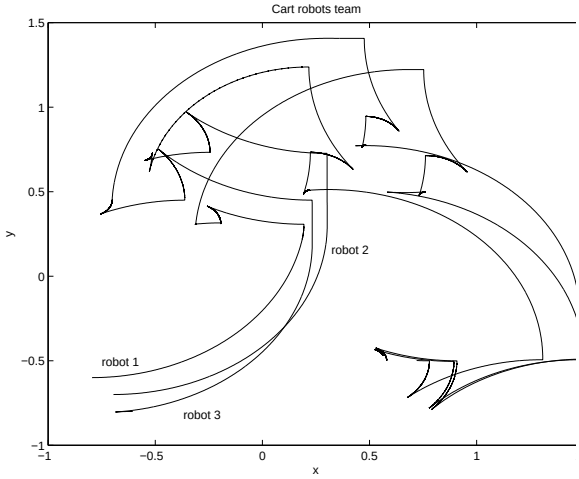
a) Mission definition



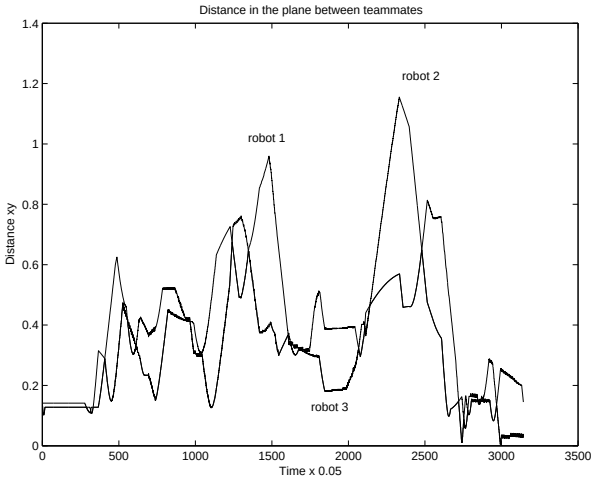
b) Robot trajectories in the C-space



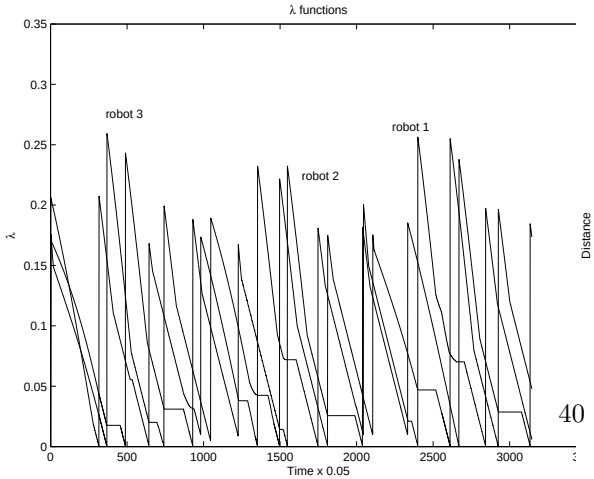
c) State evolution



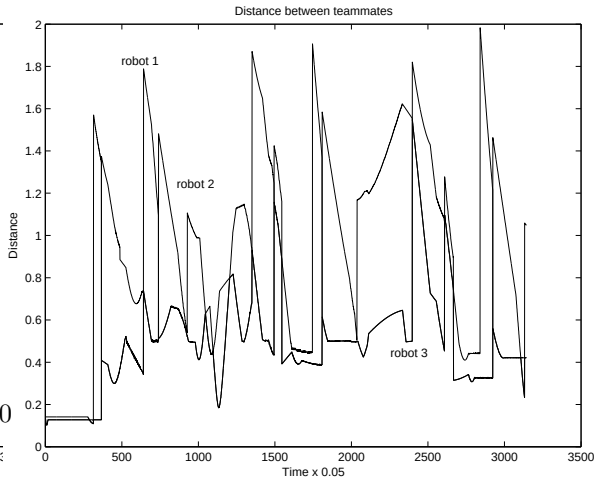
d) Robot trajectories in the plane



e) Team distances in the plane



f) λ evolution



g) Team distances in the C-space

Figure 35: Mission definition and results for the cart team

- sufficient conditions for the convergence of the robot configuration to a goal subset in the C-space or in the velocity space.

The proposed structure is of hybrid type with

- an algorithm to choose controls in one of three classes, and
- a FSA to control the switching between these three classes.

The concept of neighboring relationship is used to extend the single robot hybrid control structure to the control of teams of robots.

The success in the execution of a mission is interpreted in terms of reaching an equilibrium point in the hybrid model of system (robot and controller). Conditions for the stability of equilibria are set in terms of conditions (13) or (15). Basis concepts of non-smooth analysis are used to show that if the whole set of actions is able to search the entire space holding the Lyapunov condition then the goal region is reachable and hence the mission is successful.

The experiments on single holonomic and cart robots show that simple motion strategies can be devised to work within the proposed hybrid architecture. The experiments with the car robot also show that, although the trajectories found when the problem is set in the $\mathbb{R}^4$ embedding for the configuration space are quite complex, fairly acceptable trajectories can be easily found when the control problem is retracted to the 2D workspace. This represents a promising approach to loose specification of tasks to be used, for example by semi-autonomous robots.

The simulation examples on team control are relatively simple, aiming at illustrating that formations of holonomic and non holonomic robots share the same deep control structure.

Further work includes the study of the transient behavior of the team when a formation, defined by a specific set of neighboring relationships, is broken and how long does it take for the team to recover. Even though the presented simulation experiments validate the hybrid structure controller, a more detailed study of the design procedures for the actions at C_{21}, C_{22} and C_3 such that the Lyapunov property holds is necessary.

A Demonstration of Lemma 1

Lemma 1 follows from basic tools from Hilbert spaces. The main ideas can be found in [Aubin and Cellina, 1984, Smirnov, 2002].

The convergence between the set of admissible velocities and the set of goal velocities is a generalization of the convergence between the current velocity and the set of goal velocities. The solution of the later problem implies a solution of the former.

The demonstration of Lemma 1 is made around the discrete time version described by the following lemma.

Lemma 2 (Point-to-set convergence) *Let X be a Hilbert space, $K \subset X$ a convex set, $x \in X \backslash K$ and $\pi_K(x)$ a b.a.p. of a point x onto K. Furthermore, let $x_n = x(t_n)$ be a subsequence of a trajectory $x(t)$, outside K, and define*

$$\lambda = \langle x_{n+1} - x_n, \pi_K(x_{n+1}) - x_{n+1} \rangle, \tag{24}$$

such that, $\exists_{T>0} : \forall_{t_n>T}, \lambda > 0$. Then $d(x_n, K) \to 0$ as $n \to \infty$.

When $x_{n+1} \rightarrow x_n$ the condition in Lemma 2 can be formulated in continuous time, as

$$\langle \dot{x}, T_K^\star(x) \rangle > 0 \tag{25}$$

where $T_K^\star(x)$ represents the conjugate cone to K at $\pi_K(x)$, i.e., the set of vectors orthogonal to the set tangent to the border of K at $\pi_K(x)$.

The demonstration of Lemma 2 encompasses the following steps.

- To achieve the convergence $d(x_i, K) \rightarrow \epsilon$, for some ϵ it is sufficient to ensure Cauchy convergence, i.e., $\exists n_0 \ : \ \forall n \geq n_0, \ d(x_{n+1}, K) < d(x_n, K)$.
- Since $d(x_{n+1}, K) = \|\pi_K(x_{n+1}) - x_{n+1}\|$ then, by the properties of an inner product and Schwarz inequality, $d(x_n, K) \rightarrow 0 \ \Rightarrow \ \lambda \rightarrow 0$ (the converse is not necessarily true).
- If x_{n+1} is carefully chosen so that $\exists n_0 : \forall n \geq n_0, \quad \lambda \rightarrow 0$ slower than $d(x_n, K) \rightarrow \epsilon$ then $\epsilon = 0$.

The following lemma is used in the main demonstration.

Lemma 3 *Let $\pi_K(\cdot)$ describe a b.a.p. and*

$$\langle \pi_K(x_{n+1}) - x_{n+1}, x_{n+1} - x_n \rangle \geq 0.$$

Then $\langle \pi_K(x_n) - x_n, x_{n+1} - x_n \rangle \geq 0$.

Demonstration: Consider the identity

$$(\pi_K(x_{n+1}) - x_{n+1}) + (x_{n+1} - x_n) = (\pi_K(x_{n+1}) - \pi_K(x_n)) + (\pi_K(x_n) - x_n)$$

and compute the inner product by $x_{n+1} - x_n$ for both sides to obtain,

$$\begin{aligned} \langle \pi_K(x_{n+1}) - x_{n+1}, x_{n+1} - x_n \rangle + \|x_{n+1} - x_n\|^2 = \\ \langle \pi_K(x_{n+1}) - \pi_K(x_n), x_{n+1} - x_n \rangle + \langle \pi_K(x_n) - x_n, x_{n+1} - x_n \rangle, \end{aligned} \tag{26}$$

where the leftmost term in the lefthand side is always positive or null by assumption and the leftmost term in the righthand side is also always positive or null by the monotonicity property of b.a.p.s.

Using, simultaneously, Schwarz inequality the monotonicity and non-expansivity properties of b.a.p.s

$$\langle \pi_K(x_{n+1}) - \pi_K(x_n), x_{n+1} - x_n \rangle \leq \|\pi_K(x_{n+1}) - \pi_K(x_n)\| \|x_{n+1} - x_n\| \leq \|x_{n+1} - x_n\|^2$$

and hence (26) can be written as

$$\langle \pi_K(x_{n+1}) - x_{n+1}, x_{n+1} - x_n \rangle + \|x_{n+1} - x_n\|^2 \leq \|x_{n+1} - x_n\|^2 + \langle \pi_K(x_n) - x_n, x_{n+1} - x_n \rangle$$

or

$$\langle \pi_K(x_{n+1}) - x_{n+1}, x_{n+1} - x_n \rangle \leq \langle \pi_K(x_n) - x_n, x_{n+1} - x_n \rangle$$

from which the lemma results easily since the left side was assumed positive or null.

□

From the definition of inner product in a Banach space (see, for example, [Simmons, 1963])

$$\langle \pi_K(x_n) - x_n, \pi_K(x_n) - x_n \rangle = \|\pi_K(x_n) - x_n\|^2$$

The set distance in use can be expressed in terms of Theorem 1 by

$$d(x_n, K) = \|\pi_K(x_n) - x_n\|$$

and hence, for some x_{n+1}, the (non strict) distance diminishing consequent in the theorem must verify

$$\langle \pi_K(x_{n+1}) - x_{n+1}, \pi_K(x_{n+1}) - x_{n+1} \rangle \leq \langle \pi_K(x_n) - x_n, \pi_K(x_n) - x_n \rangle$$

$$\langle \pi_K(x_{n+1}) - x_{n+1}, \pi_K(x_{n+1}) - x_{n+1} \rangle - \langle \pi_K(x_n) - x_n, \pi_K(x_n) - x_n \rangle \leq 0$$

$$\langle \pi_K(x_{n+1}) - x_{n+1} + \pi_K(x_n) - x_n, \pi_K(x_{n+1}) - x_{n+1} - (\pi_K(x_n) - x_n) \rangle \leq 0$$

$$\langle \pi_K(x_{n+1}) - x_{n+1} + \pi_K(x_n) - x_n, (\pi_K(x_{n+1}) - \pi_K(x_n)) - (x_{n+1} - x_n) \rangle \leq 0$$

which, using the property of inner products

$$\langle a, a \rangle - \langle b, b \rangle = \langle a + b, a - b \rangle,$$

for any a, b, can be rearranged into four separate terms

$$\begin{array}{l} \langle \pi_K(x_{n+1}) - x_{n+1}, \pi_K(x_{n+1}) - \pi_K(x_n) \rangle - \langle \pi_K(x_{n+1}) - x_{n+1}, x_{n+1} - x_n \rangle + \\ + \langle \pi_K(x_n) - x_n, \pi_K(x_{n+1}) - \pi_K(x_n) \rangle - \langle \pi_K(x_n) - x_n, x_{n+1} - x_n \rangle \leq 0 \end{array} \tag{27}$$

The demonstration follows by proving the dominance of the 2nd term over the 1st term and of the 4th term over the 3rd term in expression (27).

Computing the inner product of both sides of the identity

$$\pi_K(x_{n+1}) - \pi_K(x_n) - (x_{n+1} - x_n) = \pi_K(x_{n+1}) - x_{n+1} - (\pi_K(x_n) - x_n) \tag{28}$$

by $\pi_K(x_{n+1}) - x_{n+1}$ yields

$$\begin{array}{c} \langle \pi_K(x_{n+1}) - x_{n+1}, \pi_K(x_{n+1}) - \pi_K(x_n) \rangle - \langle \pi_K(x_{n+1}) - x_{n+1}, x_{n+1} - x_n \rangle = \\ = \|\pi_K(x_{n+1}) - x_{n+1}\|^2 - \langle \pi_K(x_{n+1}) - x_{n+1}, \pi_K(x_n) - x_n \rangle \end{array} \tag{29}$$

By the Schwarz inequality and the Lemma 3,

$$\begin{array}{l} |\langle \pi_K(x_{n+1}) - x_{n+1}, \pi_K(x_n) - x_n \rangle| \leq \|\pi_K(x_{n+1}) - x_{n+1}\| \|\pi_K(x_n) - x_n\| \\ \quad \geq \|\pi_K(x_{n+1}) - x_{n+1}\|^2 \end{array}$$

If $\langle \pi_K(x_{n+1}) - x_{n+1}, \pi_K(x_n) - x_n \rangle \geq 0$ (worst case assumption) then (29) leads to

$$\langle \pi_K(x_{n+1}) - x_{n+1}, \pi_K(x_{n+1}) - \pi_K(x_n) \rangle - \langle \pi_K(x_{n+1}) - x_{n+1}, x_{n+1} - x_n \rangle \leq 0$$

and hence the 2nd term dominates over the 1st term in expression (27).

Applying again to identity (28) the inner product of both sides by $\pi_K(x_n) - x_n$ one obtains

$$\begin{array}{c} \langle \pi_K(x_n) - x_n, \pi_K(x_{n+1}) - x_{n+1} \rangle - \langle \pi_K(x_n) - x_n, x_{n+1} - x_n \rangle = \\ = \langle \pi_K(x_n) - x_n, \pi_K(x_{n+1}) - x_{n+1} \rangle - \|\pi_K(x_n) - x_n\|^2 \end{array} \tag{30}$$

Again, by the Schwarz inequality and the Lemma 3,

$$|\langle \pi_K(x_n) - x_n, \pi_K(x_{n+1}) - x_{n+1} \rangle| \leq \|\pi_K(x_n) - x_n\| \|\pi_K(x_{n+1}) - x_{n+1}\| \leq \|\pi_K(x_n) - x_n\|^2$$

If $\langle \pi_K(x_n) - x_n, \pi_K(x_{n+1}) - x_{n+1} \rangle \geq 0$ (worst case assumption) then (30) leads to

$$\langle \pi_K(x_n) - x_n, \pi_K(x_{n+1}) - x_{n+1} \rangle - \langle \pi_K(x_n) - x_n, x_{n+1} - x_n \rangle \leq 0$$

and hence the 4th term dominates the over 3rd term in expression (27) and the first part of the Lemma 2 follows

The convergence $d_(x_n, G) \to 0$ whenever x_{n+1} is chosen such that $\lambda(x_{n+1})$ decreases slower than $d(x_{n+1}, G)$ uses an inductive argument. It is assumed that x_{n+1} is chosen such that $\lambda > 0$.

Using series expansion

$$\begin{aligned} d(x_{n+1}, G) &\approx d(x_n, G) + h\dot{d}(x_n, G) \\ \lambda(x_{n+1}) &\approx \lambda(x_n) + h\dot{\lambda}(x_n) = d(x_n, G)\gamma + h\dot{\lambda}(x_n) \end{aligned}$$

for some constant $\gamma > 0$. Since λ can be defined up to a positive factor, a $\gamma > 1$ can always be found and hence $\dot{\lambda} > \dot{d}$ implies that $d(x_{n+1}, G) < \lambda(x_{n+1})$. The claim follows by induction.

□

To transform Lemma 2 to the continuous time Lemma 1 the two arguments in the inner product (24) are evaluated in the limit.

The left argument in the inner product (24) is formulated in continuous time by taking the limit when x_{n+1} converges to x_n, that is $\lim_{x_{n+1} \to x_n} x_{n+1} - x_n = \dot{x}h$, yielding the left argument in the inner product (13).

To characterize the right argument the following Theorem is used (see [Smirnov, 2002] for a demonstration)

Theorem 2 (Normal cones) *Let K be a convex set and let $x \in K$. Then the normal cone to K is*

$$N_K(x) = \{v \ : \ \langle v, y - x \rangle \leq 0, \forall y \in K\}.$$

By assumption, K is convex and, for $\lambda \geq 0$, the term $\pi_K(x_{n+1}) - x_{n+1}$ in (24) verifies

$$\langle x_{n+1} - \pi_K(x_{n+1}), y - \pi_K(x_{n+1}) \rangle \leq 0 \quad \forall y \in K$$

that is

$$-(\pi_K(x_{n+1}) - x_{n+1}) \in N_K(\pi_K(x_{n+1}))$$

or, since $-N_K(x) = T_K^\star(x)$ (see [Smirnov, 2002]), one has

$$\pi_K(x_{n+1}) - x_{n+1} \in T_K^\star(\pi_K(x_{n+1})).$$

When $x_{n+1} \to x_n$, $T_K^\star(\pi_K(x_{n+1})) \to T_K^\star(\pi_K(x_n))$ yielding the right argument in the inner product (13).

□

B Lyapunov's direct method

This section provides the basics for Lyapunov direct method theorem (see for instance [Smirnov, 2002] for details).

The upper Dini derivative of a function $V(q)$ at vector v is defined by

$$D^+V(q)(v) = \lim_{h\to 0} \sup_{v'\to v} \frac{V(q+hv') - V(q)}{h}$$

Consider the system $\dot{q} \in F(q)$, where $F : \mathbb{R}^n \mapsto \mathbb{R}^n$ and $0 \in \mathrm{Ker}(F(q))$.

Lyapunov stability theorem states that

Theorem 3 (Lyapunov stability theorem)
Assume that there exist $\eta > 0$, a positive definite function $V : \mathbb{R}^n \mapsto \mathbb{R}$ and a negative definite function $W : \mathbb{R}^n \mapsto \mathbb{R}$ such that

$$\forall v \in F(q), |q| \leq \eta, \quad D^+V(q)(v) \leq W(q)$$

Then the equilibrium position $q = 0$ is asymptotically stable.

References

[Arkin and Balch, 1998] Arkin, R. and Balch, T. (1998). Cooperative Multiagent Robotic Systems. In Kortenkamp, D., Bonasso, R., and Murphy, R., editors, *AI-Based Mobile Robots:Case Studies of Successful Robot Systems*. MIT Press. Conference and Symposium Papers, in press.

[Aubin, 1991] Aubin, J. (1991). *Viability Theory*. Birkhäuser.

[Aubin and Cellina, 1984] Aubin, J. and Cellina, A. (1984). *Differential Inclusions*. Springer-Verlag.

[Bacciotti et al., 2000] Bacciotti, A., Ceragioli, F., and Mazzi, L. (2000). Differential inclusions and monotonicity conditions for nonsmooth lyapunov functions. *Set Valued Analysis*, 8:299–309.

[Balch and Arkin, 1999] Balch, T. and Arkin, R. (1999). Behavior-based formation control for multirobot teams. *IEEE Trans. on Robotics and Automation*, 14:926–939.

[Brocket, 1983] Brocket, R. (1983). Asymptotic stability and feedback stabilization. In Millman, R., Brocket, R., and Sussmann, H., editors, *Differential Geometry and Geometrical Control*. Birkhäuser, Boston.

[Drogoull and Collinot, 1998] Drogoull, A. and Collinot, A. (1998). Autonomous Agents and Multi-Agent Systems. In *Applying and Agent-Oriented Methodology to the Design of Artificial Organizations: A Case Study in Robotic Soccer*. Kluwer Academic.

[Egerstrdt and Hu, 2001] Egerstrdt, M. and Hu, X. (2001). Formation constrained multi-agent control. *IEEE Trans. on Robotics and Automation*, 17(6):947–951.

[Grünbaum, 1995] Grünbaum, B. (1995). How to convexify a polygon. *Geocombinatorics*, 5:24–30.

[Hermann and Krener, 1977] Hermann, R. and Krener, A. (1977). Nonlinear Controllability and Observability. *IEEE Transactions on Automatic Control*, AC-22(5).

[Jennings, 1999] Jennings, N. (1999). Controlling Cooperative Problem Solving in Industrial Multi-Agent Systems Using Joint Intentions. *Artificial Intelligence*, (75):195–240.

[Khatib et al., 1997] Khatib, O., Quinlan, S., and Williams, D. (1997). Robot planning and control. *Robotics and Autonomous Systems*, (21):249–261.

[Košecká et al., 1997] Košecká, J., Christensen, H., and Bajcsy, R. (1997). Experiments in behavior composition. *Robotics and Autonomous Systems*, 19:287–298.

[Kreysig, 1978] Kreysig, E. (1978). *Functional Analysis*. Wiley.

[Latombe, 1992] Latombe, J. C. (1992). *Robot Motion Planning*. Kluwer Academic.

[Lee and Li, 2003] Lee, D. and Li, P. (2003). Formation and maneuver control of multiple spacecraft. In *Proc. of the American Control Conference*. Denver, Colorado, June 4-6.

[Lygeros, 1999] Lygeros, J. (1999). Hybrid systems: Modelling analysis and control. Teaching/ee291E.html.

[Murray and Sastry, 1993] Murray, R. and Sastry, S. (1993). Nonholonomic Motion Planning: Steering using Sinusoids. *IEEE Transactions on Automatic Control*, 38(5).

[Ögren and Leonard, 2003] Ögren, P. and Leonard, N. (2003). Obstacle avoidance in formation. In *Proc. of the IEEE Int. Conf. Robotics and Automation, 2003*.

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

[Parker, 1998] Parker, L. E. (1998). ALLIANCE: An Architecture for Fault Tolerant Multirobot Cooperation. *IEEE Transactions on Robotics and Automation*, 14(2):220–240.

[Preparata and Shamos, 1985] Preparata, F. and Shamos, M. (1985). *Computational Geometry: An Introduction*. Springer-Verlag.

[Samson, 1992] Samson, C. (1992). Path Following And Time-Varying Feedback Stabilization of a Wheeled Mobile Robot. In *Proceedings of the ICARCV'92*. Singapore.

[Sequeira and Ribeiro, 2001] Sequeira, J. and Ribeiro, M. (2001). A negotiation model for cooperation among robots. Proceedings of the European Control Conference 2001, ECC 2001, September 4-7, Porto, Portugal.

[Sequeira and Ribeiro, 2003] Sequeira, J. and Ribeiro, M. I. (2003). A geometric approach to single and multiple robot control. In *Proceedings of the 7th IFAC Symposium on Robot Control, Syroco 2003*.

[Simmons, 1963] Simmons, G. (1963). *Topology And Modern Analysis*. McGraw-Hill.

[Smirnov, 2002] Smirnov, G. (2002). *Introduction to the Theory of Differential Inclusions*, volume 41 of *Graduate Studies in Mathematics*. American Mathematical Society.

[Sordalen, 1993] Sordalen, O. (1993). *Feedback Control of Nonholonomic Mobile Robots*. PhD thesis, The Norwegian Institute of Technology. Department of Engineering Cybernetics, N-7034 Trondheim, Norway.