SCUOLA SUPERIORE DI CATANIA

Marco Pavone

Architectures for Autonomous Robots: Adaptive Locomotion and Distributed Covering

Diploma Thesis

ADVISERS:

Chiar.mo Prof. Ing. P. ARENA

Chiar.mo Prof. Ing. E. FRAZZOLI

Academic Year 2004/2005

Contents

List of Figures

List of Tables

Introduction

The concept of fully autonomous robots able to perform missions in harsh and hazardous environments is nowadays the Holy Grail of robotics research. *Civilian* applications of autonomous robotics are tremendous. Landmines, for example, are one of the biggest problems in many countries throughout the world; the so-called Ottawa convention, signed at the end of 1997, requires that all landmines are eliminated before year 2010. Unfortunately, demining is an extremely dangerous task that consequently requires a long time to completion. On the other hand, if a team of inexpensive and simple robots is used, safety does not represent a problem and therefore demining operations can be much faster.
Explorative space missions represent another important application: nowadays it is commonly accepted that rover autonomy is a fundamental keyword in successful planetary explorations. In fact, in order to compensate the fact that the robotic platform can be too far to be safely reached in case of malfunctions or in case of unexpected situations, it is preferable that an extreme self-reliance and functional autonomy capabilities are present on the rover.
Autonomous robots for hazardous environment exploration could also be helpful in more "practical" problem like pipe inspection or sewer maintenance.

Unfortunately, autonomous and self-organizing robots aimed at hazardous missions represent a huge technological challenge. The two fundamental issues to be addressed are:

- rover autonomous locomotion: robot should transverse uneven terrains with large obstacles without man control;
- team coordination: robots should autonomously self-organize to accomplish the mission.

In this thesis, we deal with both problems: on one hand we study the design of a bio-inspired autonomous hexapod robot, called GregorI, on the other hand we propose a simple algorithm

to accomplish, in a distributed way, a prototypical problem: complete coverage of hazardous environments.

As far as robot design is concerned, biology provides a wealth of inspiration: insects are able to transverse harsh terrains, to climb over obstacles or even to walk upside down. Moreover, essential aspects in unmanned missions, like reconfigurability of locomotion strategies, navigation capabilities and robustness, are common features among insects. Thus, several efforts, both from a behavioral and an architectural viewpoint, have been performed to design insect-like robots [9, 7, 14]. Strongly believing that a bio-inspired approach can largely benefit the design of an autonomous robot, we took explicitly inspiration from cockroach experimental observations. Biological results inspired both the hexapod structure and the control system architecture design. The peculiarity of the proposed structure is that, in order to replicate at least in part the cockroach extraordinary agility, each of the three leg pairs has a unique design: front legs and middle legs have 3 degree of freedoms and a kind of pantograph mechanism aimed at facilitating the climbing obstacle task, while the rear legs have 2 degrees of freedom and a piston-like design suitable for powerful forward thrusting. Dynamical simulations prove that, thanks to the careful linear/rotational actuation, GregorI is able to successfully overcome high obstacles.
The control system architecture has a two-levels hierarchical organization and is based on biological results discussed in [38] and on the behavior-based control theory. The low level control is based on a CNN-based Central Pattern Generator (CPG), discussed in details in [16]; the CPG provides the basic rhythmic signals needed for locomotion. The main focus is on the high level control, whose purpose is adaptively handling complex tasks like obstacle climbing and target pursuing. We adopted a behavior-based approach, since behavior-based systems have been praised for their robustness and simplicity of construction and seem to be a really suitable approach to perform autonomously exploration task in a unknown environment. We chose to formalize a behavior as a Motor Map that adaptively maps sensory stimuli in signals that modulate the CPG outputs; in order to confer the plasticity needed for autonomous systems, we further considered an adaptive *reward function* able to learn its structure basing on experience. To the best of our knowledge, it is the first time that a behavior is formalized with a Motor Map and an adaptive reward is introduced.

Unfortunately, in a hazardous mission a robot can break, thus it is necessary to use swarms

of autonomous robots. The potential advantages of employing teams of agents are numerous. For instance, the intrinsic parallelism of a multi-agent system can guarantee better time performance. Further, a group of robots inherently provides robustness to failures of single agents. A prototypical problem that could benefit from a multi-agent approach is coverage path planning in hazardous environments, where the aim is to sweep all points in the target environment facing disturbances and agent losses.
Multi-agent coverage path planning algorithm raises several challenging issues as coverage completeness, robustness and coverage redundancy.
Our main concern is to study the feasibility of a coverage algorithm for non-holonomic robots (like an hexapod) that:

1. does not rely on costly grid maps;
2. provides complete coverage;
3. takes into account only local sensory information, thus avoiding inter-agent communication, landmark deployment and other possibly costly mechanisms for information exchange.

Our main assumption, necessary to achieve proof of correctness, is that robots can overcome all obstacles placed in the environment without modifying their desired trajectory.
The key idea is to consider a modified version of the classical cycling pursuit control strategy for non-holonomic robots, previously extensively studied in [25], where it is shown that system's equilibrium formations are generalized regular polygons. In our strategy, instead of pursuing the leading neighbor along the instantaneous line of sight, each agent pursues its leading neighbor along the line of sight rotated by an offset angle, function of locally available sensory information. It is shown that the paths described by the system at equilibrium are Archimede's spirals able to provide complete coverage of the target environment. These spiral-like paths appear to be robust against sensory noise, odometry error and agent loss.

The thesis is articulated into two parts: in chapters $1-3$ the hexapod robot is described, while in chapters $4-5$ the distributed path planning algorithm is discussed. In detail, in chapter 1 we discuss GregorI structure and its biological inspiration; in chapter 2 both the low level and the high level locomotion control are presented and discussed. In chapter 3 we report simulation results

that validate the proposed structure and control system design. Chapter 4 provides the statement of the distributed path planning coverage problem and the mathematical analysis of the proposed control algorithm. In chapter 5 simulation results show the effectiveness of the proposed algorithm. Finally we present our conclusion.

The first part of the thesis has been developed at the Electrical, Electronic and Systems Engineering Department of the University of Catania under the supervision of prof. P. Arena, while the second part has been developed at the Department of Mechanical and Aerospace Engineering of the University of California at Los Angeles, USA, under the supervision of prof. E. Frazzoli.

Part of the work presented in this thesis has been published or is going to be published in [4, 5, 31, 32]. Moreover, the part on the hexapod design is going to be presented at the 56^{th} *International Astronautical Congress 2005, Fukuoka, Japan*, under the European Space Agency Sponsorship.

Chapter 1

Structure of *Blaberus Discoidalis* and Robot Design

Structure and kinematics of cockroach guided the robot structure design in all its aspects, from leg design to body design to joint kinematics. Therefore, before discussing the proposed robot structure, we outline some of the most important results coming from cockroach experimental observations, with particular emphasis on the *Blaberus Discoidalis*. Then, after a literature review of previous hexapod robots, we describe in detail the various steps that led to the final robot design, emphasizing the parallelism with *Blaberus Discoidalis*.

1.1 Structure and kinematics of *Blaberus Discoidalis*

Structure and kinematics of *Blaberus Discoidalis* inspired leg design, body design and leg-body articulation.

1.1.1 Structure

Most important structural features of *Blaberus Discoidalis* from an engineering viewpoint are:

- leg structure;
- leg articulation;
- body structure.

Each cockroach leg is divided into several segments. Although the segments are reproduced in each of the three pairs of legs, their dimensions are very different in the front, middle and rear legs. The leg segments from the most proximal to the most distal segment are called coxa, trochanter, femur, tibia and tarsus; the last one is indeed constituted by a series of foot joints.
The complex musculature coupled with complex mechanics confers upon the joint between body and coxa three degrees of freedom (DOF), much like that of a ball and socket joint. The joints between the coxa and trochanter, between the trochanter and femur, and between the femur and tibia are, instead, simple one DOF rotational joints. The joint between the trochanter and femur makes only a small movement and has often been referred to as fused. Each tarsal joint has several passive DOF, guaranteeing agile foot placement. Finally, a claw located on the end of the tarsus can be raised or lowered to engage the substrate during locomotion on slippery surfaces for climbing [35].

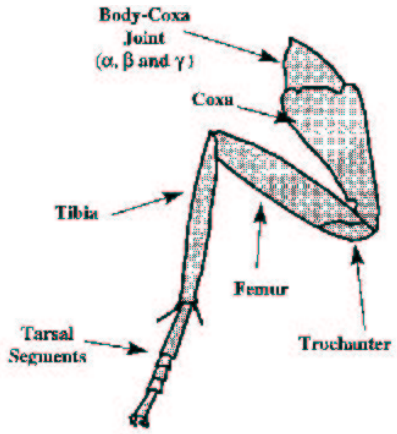


Figure 1.1: Leg structure in *Blaberus discoidalis* [35].

Although, as stated above, front, middle and rear legs have the same segments, they are different in lengths, yielding a ratio of front:middle:rear leg lengths of 1:1.2:1.7 [14]. Leg pairs with different length provide agility and adaptability. Cockroach legs articulate differently with the body, with the front legs oriented almost vertically at rest and middle and rear legs angled posteriorly of about 30° and 50° respectively [14]. This configuration confers a *sprawled posture* able to guarantee a statically stable posture and thus a high margin of stability [39], [13] . In fact the center of mass

of the system is always inside the support polygon, obtained connecting all the legs in stance phase. Finally, body is divided into three articulated segments called prothoracic, mesothoracic and metathoracic segments. Anyway, dorsal flexion is seldom accomplished [41].

Legs perform different functions [41]:

- **Front legs** – are mainly use to push the body of the cockroaches over obstacles. They also play an important role in turning and in decelerating their body.
- **Middle legs** – act to push the cockroaches forward but also push the body of the cockroaches over obstacles.
- **Rear legs** generate the major part of the forward motion. They push directly toward the mass center and the contact point is far behind to prevent the cockroaches falling on their back when climbing obstacles.

1.1.2 Kinematics: horizontal walking

Electrophysiological recordings of motor activity show that the joints more involved during horizontal walking are the CTF joint (coxa-trochanter-femur) and the FT joint (femur-tibia). Both in the rear legs and in the middle legs CTF and FT joints move in synchrony (Fig. 1.2). Front legs movements are, instead, unique and more complex, since they are also aimed at exploring the environment [41], [35].

Figure 1.2: Typical joint angle data from a rear leg of a cockroach [35].

1.1.3 Kinematics: climbing obstacle

Electrophysiological recordings of motor activity show also the strategies adopted by cockroaches to climb over an obstacle. To overcome an obstacle, animals must elevate their CoM over the barrier; in general three different strategies can be adopted:

- overcoming the obstacle without any changes of the locomotion gait;

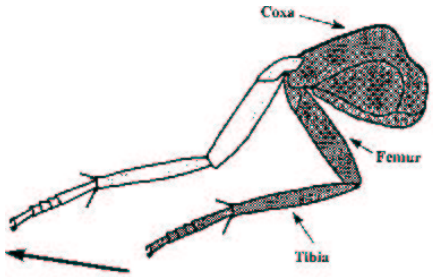


Figure 1.3: Rear leg movement [35].

- performing a drastic change of the leg movements;
- carrying out slight changes in the locomotion pattern and at the same time making some postural adjustments.

In [41, 42] an exhaustive set of experimental data referring to kinematic changes associated with climbing in the *Blaberus Discoidalis* is reported. Experimental data show that cockroaches do not deviate from normal running kinematic in surmounting obstacles whose height is smaller than one reached by front legs during swing trajectory: once one or both front tarsi are naturally placed on top of the barrier, they push downward, changing the animal's posture so that the subsequent movements of all legs drive the Center of Mass (CoM) upward; therefore for small barriers there is not an anticipatory change in running strategy. A remarkable stability of the structure, guaranteed by a sprawled posture, is fundamental to perform this strategy. On the other hand, several different strategies have been observed when an hexapod insect faces an high obstacle whose top is beyond the height of front legs during swing phase [41]:

- **Elevate** – Simultaneous extension of all the legs with consequent elevation of the CoM.
- **Elevator Reflex** – Increase of leg elevation during the swing phase when the body is still in the horizontal position.
- **Head Butt** – The collision between the head of the insect and the obstacle increases the elevation of the body.
- **Jumping** – Jump on the top of the obstacle.

- **Rear Up** – An anticipatory postural adjustment occurs when an obstacle is detected. The pitch angle is increased before the collision and the insect is able to move the frontal legs on the step.

The Rear Up strategy is the most common strategy performed by cockroaches, that normally accomplish an anticipatory attitude change tilting the body upward. The animal performs this postural adjustment *before* front legs are placed on top of the barrier, principally by rotating the middle legs in order to bring them perpendicular to the ground. In the subsequent phase, the animal's CoM is raised upward with little or no further change in body-substrate angle.

Thus, climbing high barriers is accomplished in two stages:

- rearing stage: cockroaches change the body-substrate angle *before* any leg reaches the barrier;
- rising stage: animal's CoM is raised upward.

The main point is that climbing does not require radical departures from running control mechanism, but possibly just an anticipatory rearing stage.

Since reorientation of middle legs in rearing stage is initiated only after the height of the obstacle has been evaluated, postural changes appear to be directed, at least in part, by higher centers, as *supraesophageal ganglia*, driven by sensory feedback (presumably by visual feedback and antennae) [41].

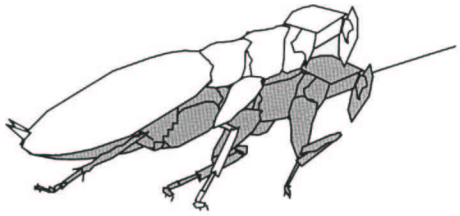


Figure 1.4: Postural adjustment in cockroaches [35].

1.2 Previous hexapod design

In this section, we review some literature results regarding hexapod robot design, analyzing in detail the adopted criteria for leg design and actuator selection.

Figure 1.5: Sprawlita robot [13].

1.2.1 Leg design

A fundamental issue in leg design is the number of degrees of freedom that each leg should possess: many DOF imply better agility and flexibility, but a more difficult control. An example of robot with just one DOF per leg is Rhex [1]. Robot structure consists of a rigid body with six equal compliant legs, each possessing only one independently actuated revolute degree of freedom. The attachment points of the legs as well as the joint orientations are all fixed relative to the body. Basically, spoked wheel concept is exploited. This very simple design guarantees surprisingly good locomotion properties, but lacks of the agility needed for more complex tasks.
Several hexapod robots reported in literature have legs with 2 DOF; one example is Sprawlita [13]. Sprawlita has 6 identical legs with 2 degrees of freedom each. The primary thrusting action in Sprawlita is performed by a prismatic actuator, implemented by a pneumatic piston. This piston is attached to the body through a compliant rotary joint at the hip. This unactuated rotary joint is based on studies of the cockroachs compliant trochanter-femur joint, which, as stated above, is largely passive. In the prototype, the compliant hip joint allows rotation mainly in the sagittal plane. These active-prismatic, passive-rotary legs are sprawled in the sagittal plane to provide specialized leg function (although all legs are indeed identical). Servo motors rotate the base of the hip with respect to the body, thus setting the nominal, or equilibrium, angle about which the leg will rotate. By changing this angle, the function that the leg performs is affected; *e.g.* aiming the thrusting action towards the back, robot accelerates, on the contrary towards the front robot decelerates. Sprawlita is fast and able to overcome small obstacles, but can not perform more

Figure 1.6: Pantograph mechanism of Boadicea robot leg [44].

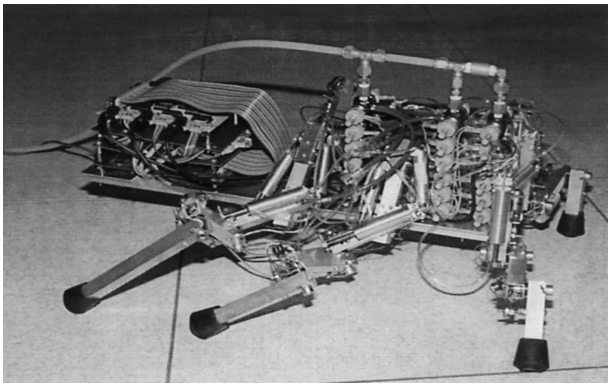


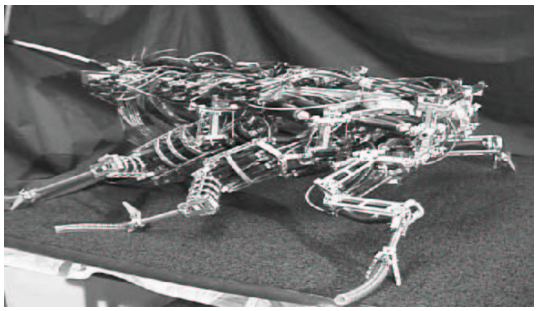
Figure 1.7: UIUC Robot [14].

complex task like climbing high obstacle due to its structure and control simplicity.
Another interesting example of two DOF robot is Boadicea [44]; its legs use a 2 dimensional pantograph mechanism that produces linear foot motions, with the advantage of simpler software control. A second advantage of the pantograph mechanism is that it provides a large leg workspace with a relatively simple and compact mechanism. Like an insect, Boadicea has different front, middle, and rear legs.

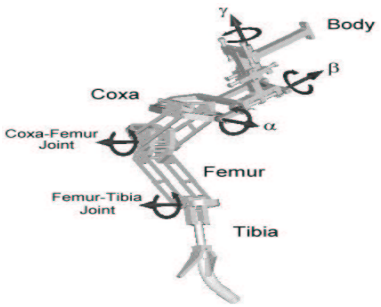
A very interesting example of robot with 3 DOF per leg is UIUC, discussed in [14]. Legs are divided into three segments, corresponding to the three main segments of insect legs: coxa, femur, and tibia. The coxa articulates with the body, the femur with the coxa, and the tibia with the femur. Each of the joints between leg segments and between the coxa and the body is a simple hinge joint. The length ratio for the robots legs is $1 : 1.1 : 1.5$. The coxae of the front legs are attached vertically, while the middle leg coxae are attached at an angle of about $75°$ from horizontal. Finally, rear legs are attached at an angle of about $30°$. This structure, taking into account the most important features of a cockroach, confers to the robot a high stability and avoids a useless

complexity.

An hexapod robot with kinematics remarkably similar to those of the *Blaberus Discoidalis* is Robot V, discussed in [19] and shown in Fig. 1.8. Rear legs have three DOF, middle legs have four DOF and front legs have five DOF. Leg design attempts to capture in detail all cockroach leg faetures, but the resulting robot is more useful from a theoretical than from a practical viewpoint due to its complexity.



(a) Overall structure



(b) Leg - Detail

Figure 1.8: Robot V.

1.2.2 Actuators

Actuator selection represents a fundamental issue in robot design, since the shape, size, weight and strength of an actuator and its power source provides the greatest constraint on robots potential abilities. Biological organisms have a great advantage over mechanical systems in that muscles, the biological actuators, have a favorable power-to-weight ratio and require low levels of activation energy, compared to any actuator.

The most frequently used actuators are electric motors and pneumatic/hydraulic cylinders. Electric motors are the most commonly used actuators since they are readily available in a wide range of sizes and are very easy to control and integrate in a hardware scheme. However, electric motors have some disadvantages: they can provide just a rotational motion and, most importantly, they have a low power-to-weight ratio. On the contrary, pneumatic and hydraulic actuators have a high power-to-weight ratio and produce linear motion. Unfortunately, pneumatic/hydraulic cylinders are better suited to "bang-bang" operations, need a complex mechanics and require a sophisticated

control; furthermore, pneumatic actuators need an expensive and heavy compressor. Recently, many new types of actuators are being introduced like Shape Memory Alloys, Piezoelectric Motors and Electroactive Polymers. In [33] the feasibility of a worm-like robot actuated by IPMC is discussed.
Referring to previous robots, Rhex is electrically actuated, Sprawlita, Boadicea and UIUC are pneumatically actuated, Robot V is actuated by means of McKibben artificial muscles.

1.3 Robot design

Biological principles and previous hexapod prototypes guided the structure design phase. Our main concern was to replicate the cockroach features that are mainly responsible of its extreme agility, adaptability and stability. We also took into careful consideration fundamental engineering issues like actuator selection.
In this section, we outline the various steps that led to the final robot structure, from the leg design to the overall structure.
Before the structure design is started, it is necessary to specify what GregorI is intended to do, since the final task deeply affect the overall design: *e.g.*, as far as leg design is concerned, if the focus is just on horizontal walking, two DOF per leg are enough.
Final task of GregorI is efficiently walking on uneven terrains and overcoming obstacles with height at least equal to robot mass center height.

The dynamic robot model was built in a C++ environment basing on `DynaMechs` libraries [27]. Dynamical simulation of the model allowed us to asses structure and control suitability, as it will be discussed in chapter 3.

1.3.1 Leg design

Let us briefly recapitulate the lesson learnt form the previous sections:

- **Leg function** – front legs are mainly used to push the body upward, to explore and to prevent falling on the front part; middle legs act to push the cockroaches forward but also push the body of the cockroaches over obstacles; finally, rear legs generate the major part of

the forward motion and confer to the overall structure high stability.

- **Sprawled posture** – a sprawled posture is essential to guarantee a statically stable posture.
- **DOF** – many DOF guarantee high flexibility but at the cost of a complex control. We can allege that an agile hexapod robot should possess $3-5$ DOF in front and medium legs and $2-3$ DOF in rear legs.
- **Mechanics** – center of mass position has to be placed carefully.
- **Actuators** – just referring to conventional robot actuators, we have

  Pneumatic actuators *Advantages*: powerful, able to deliver high forces for linear motion actuation. *Disadvantages*: "bang-bang" actuation, complex control, need of a compressor.

  Electrical motors *Advantages*: ease of control, low payload for power supply. *Disadvantages*: unable to deliver high torques, low stiffness.

Clearly, all these issues are intrinsically interlaced.

Front legs

Front legs have to provide enough flexibility to guarantee an efficient obstacle approach and an effective postural control. Toward this end, front legs are divided into three segments (analog to coxa, femur and tibia), articulated through 3 DOF rotational joints (α, β and γ joints in Fig. 1.9). In order to facilitate climbing obstacle task, we chose a segment assembly that resembles the pantograph mechanism; in Fig. 1.9, rotation axes are depicted. These rotational joints clearly require a precise control, but they do not need high actuation torques, therefore they can be actuated by conventional electrical motors.

As far as mechanical details are concerned, all segments are simply modelled as cylinders whose weights are reported in Tab. 1.1 together with the physical dimensions (radius and height respectively). The apparently ridiculous coxa height is due to the fact that we modelled the coxa segment just as a motor placed on the body structure. As far as the dynamics is concerned, joint

Table 1.1: Front Coxa, Femur and Tibia properties

Segment	Weight g	Dimensions cm
Coxa	60	0.5×0.1
Femur	60	0.5×3
Tibia	60	0.5×5

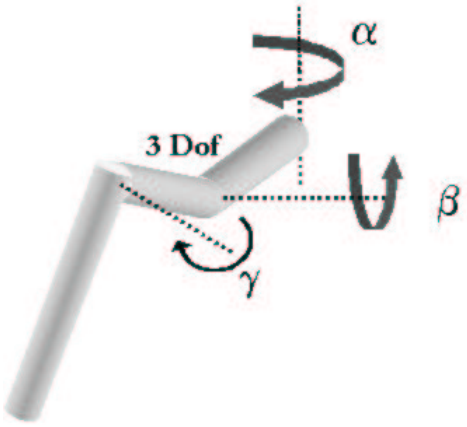


Figure 1.9: Front leg structure (not in scale).

α allows leg forward movement, joint β allows raising movement and, finally, joint γ guarantees a roll and pitch angles control. The γ joint plays a fundamental role in the attitude control, but is not needed for basic locomotion; therefore, as far as basic locomotion is concerned, γ joint is kept at the constant value -1.2 rad.

Middle Legs

Basically, middle leg design is identical to front leg design as far as number of segments, physical dimensions, joint type and actuation are concerned; nevertheless, middle legs have to provide part of the forward thrust, so a different segment assembly is needed; in particular, referring to Fig. 1.10, the α axis is modified. As far as the dynamics is concerned, similarly to front legs, joint α allows leg forward movement, joint β allows raising movement and joint γ guarantees a roll and pitch angles control. For basic locomotion the γ joint is kept at the constant value -1.2 rad.

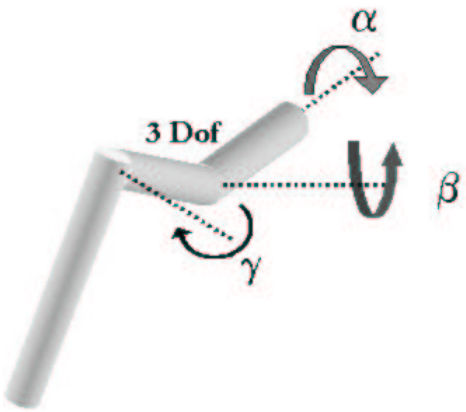


Figure 1.10: Middle leg structure (not in scale).

Rear Legs

In the rear leg design we bring the most important innovation as far as the robot structure is concerned. Since main function of rear legs is powerful thrust, we considered a robust and compact design that allows the use of a "bang-bang" pneumatic actuation.
Rear legs are divided into two segments (coxa and tibia respectively); coxa segment is articulated with the body with a rotational joint (α joint), while the coxa-tibia joint is prismatic (d joint), as shown in Fig. 1.11. Therefore, rear legs possess a peculiar hybrid linear/rotational actuation that could allow the use of a combination of electrical and pneumatic actuators.
Rear legs are considerably longer in order to facilitate the climbing obstacle task. Both segments are simply modelled as cylinders whose weights and dimensions are reported in Tab. 1.2. As far as

Table 1.2: Rear Coxa and Tibia properties

Segment	Weight g	Dimensions cm
Coxa	60	0.5×0.1
Femur	60	0.5×12

the dynamics is concerned, joint α allows leg forward movement, joint d allows raising movement.

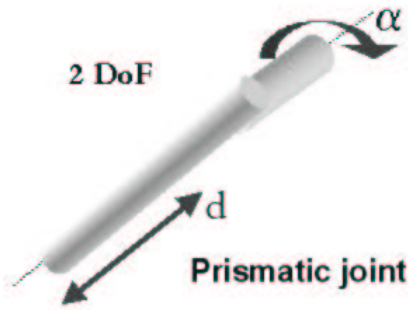


Figure 1.11: Rear leg structure (not in scale).

1.3.2 Body

Body is modelled by just one segment with parallelepiped shape; we also considered the presence of a payload, similarly modelled as a parallelepiped. Weights and dimensions are reported in Tab. 1.3 and Tab. 1.4. Position of the mass center is critical for good performance, as observed in several

Table 1.3: Body properties

Mass g	Length cm	Width cm	Height cm
420	20	8	2

Table 1.4: Payload properties

Mass g	Length cm	Width cm	Height cm
300	5	5	1

simulations. Therefore, we studied in detail payload placement through a trial and error process. Assuming a body reference system with the z-axis aligned along the body longitudinal axis and the origin located at the body mass center, we finally placed the payload mass center at coordinates, in the body reference system, $(0, 0, -3.5)$. Simulation proved that with this arrangement speed and stability are enhanced. In Fig. 1.12 the body is shown (dark part represents the payload).

1.3.3 Overall structure

In order to confer a sprawled posture with a pitch angle of $\varphi \simeq 15°$, legs articulate differently with the body; articulation angles between leg and body are shown in Tab. 1.5. The sprawl, or

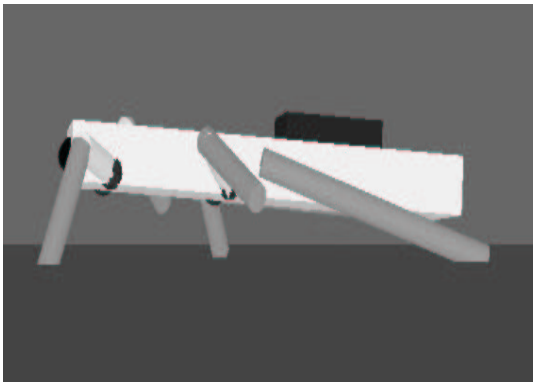
Figure 1.12: Body structure.

Table 1.5: Leg angles from horizontal at rest

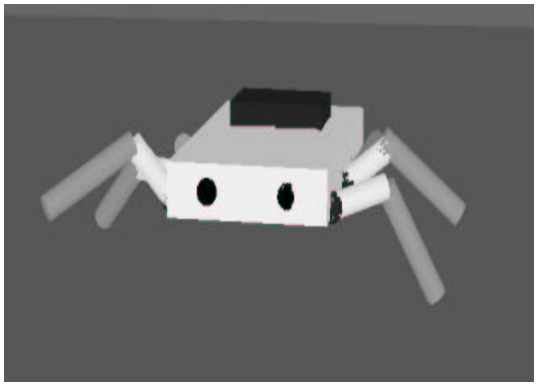
Front legs	90°
Middle legs	60°
Rear legs	20°

inclination, angle for each leg is limited by foot traction: for larger animals (or robots), it becomes progressively harder to sustain the necessary tangential forces.

The overall structure is shown in Fig.1.13, where the sprawled posture, the peculiar leg articulation and the payload placement are evident. Overall, robot length is 20 cm and robot CoM height is



(a) Overall structure: side view



(b) Overall structure: front view

Figure 1.13: Overall structure.

3.5 cm.

Chapter 2

Locomotion control

As for the structure design, biological results guided the design of the control system architecture. The proposed locomotion control is divided into two levels; the low level control provides, according to the *Central Pattern Generator* theory, the basic rhythmic movements needed for locomotion, while the adaptive high level control, modulating the rhythmic signals, is responsible for complex task execution, like pursuing a target.

The main focus in this thesis is on the high level control, based on the behavioral approach. We introduce a novel formalization of "behavior" based on the Motor Map theory and, in order to confer plasticity to the whole system, we study the feasibility of Motor Maps with adaptive *reward functions*.

Before outlining the proposed control system architecture, we discuss some essential biological results and their implication in our work.

2.1 Control system architecture: biological inspiration

In this section we discuss in some detail the Central Pattern Generator theory, that led to the design of the low level control, and the bio-inspired Motor Map theory at the base of the high level control.

2.1.1 CPG and locomotion gaits

Most insects exhibit a locomotion hierarchical control and use a modular organization of the control elements. The activation of the appropriate muscles in the legs and their coordination take place locally by means of groups of neurons functionally organized in modules called Central Pattern Generators (CPG). Walking insects adopt several distinct periodic patterns of leg movements, called gaits, which are believed due to patterns of neural activity within the CPG [37]. The output signals of the CPG control directly the effector organs. The CPG receives stimuli from the high level control layers that monitor overall locomotion and take decisions about the high level task for example by changing the locomotion gait. Three different gaits are typically shown by hexapods during walking: fast, medium and slow gait. They are adopted under different conditions to perform high speed locomotion (fast gait) or extremely stable and secure movements (slow gait). The characteristics of these locomotion gaits can be rigorously defined through the concepts of *cycle time*, *duty factor*, and *leg phases*. The cycle time is the time required for a leg to complete a locomotion cycle. The duty factor df_i is the time fraction of a cycle time in which the leg i is in the power stroke phase. The leg phase φ_i is the fraction of a cycle period by which the beginning of the return stroke of leg i lags behind the beginning of the return stroke of the left front leg (L1), chosen as a reference. Basing on these quantities, a precise gait classification is shown in Tab. 2.1:

Table 2.1: Classification of fast, medium and slow gaits

Fast	$\varphi_{L2} = \frac{1}{2}$	$\varphi_{L3} = 0$	$\varphi_{R1} = \frac{1}{2}$	$\varphi_{R2} = 0$	$\varphi_{R3} = \frac{1}{2}$	$df_i = 1/2$
Medium	$\varphi_{L2} = \frac{3}{4}$	$\varphi_{L3} = \frac{2}{4}$	$\varphi_{R1} = \frac{2}{4}$	$\varphi_{R2} = \frac{1}{4}$	$\varphi_{R3} = 0$	$df_i = 5/8$
Slow	$\varphi_{L2} = \frac{4}{6}$	$\varphi_{L3} = \frac{2}{6}$	$\varphi_{R1} = \frac{3}{6}$	$\varphi_{R2} = \frac{1}{6}$	$\varphi_{R3} = \frac{5}{6}$	$df_i = 9/12$

2.1.2 Brain topology-preserving maps and Motor Maps

The importance of topology-preserving maps in the brain relies on both the representation of sensory input signals and the ability to perform an action in response to a given stimulus. Neurons in the brain are organized in different local assemblies which are able to perform a given task such as sending appropriate signals to muscles. These neural assemblies constitute two-dimensional layers in which the locations of the excitation are mapped into movements.

In this thesis, to address the stimulus-action relation issue we considered the Motor Map paradigm. Motor Maps are artificial neural networks based, like neural assemblies, on a two-dimensional neural layer, a set of synaptic weights that determine the correspondence between input signals and neurons and a set of output values [37].
The learning algorithm is the key to obtain a correct mapping between stimuli and actions. This is achieved by considering an extension of the winner-take-all algorithm; at each learning step, when a pattern is given as input, the winner neuron is identified: this is the neuron that best matches the input pattern. Then, a neighborhood of the winner neuron is considered and an update involving both the input and output weights for neurons belonging to this neighborhood is performed according to the *reward function*. The reward function, measuring how well the control is being performed, plays the central role in Motor Maps learning. Weight updating takes place only if the corresponding control action leads to an improvement in the system being controlled, *i.e.* an increase in the reward function; it should be point out that no *a priori* information, **except the form of the reward function**, is taken into account during the learning process.
Basically, the learning process is articulated as follows:

- **Step 1** – Determine the lattice site s (winner neuron) whose input weight vector best matches the input $v(t)$ according to some distance function $\psi(\cdot)$
- **Step 2** – Perform the control action using as output value:

$$\zeta(t) = w_{\text{winner,out}} + a_s \lambda \tag{2.1}$$

  $w_{\text{winner,out}}$ is the output value of the winner neuron, a_s is a parameter determining the mean value of the search step for the neuron and λ is a Gaussian random variable with zero mean aimed at guaranteeing a random search for all possible solutions.

- **Step 3** – Compute the actual increase ΔR in the reward function; if ΔR exceeds the mean increase b_s of the reward function at lattice site s update the input weights $w_{\text{i,in}}$ and the output weights $w_{\text{i,out}}$ of neuron s and its neighboring neurons as follows:

$$\begin{aligned} w_{\text{i,in}}(t + \Delta t) &= w_{\text{i,in}}(t) + \eta\xi\,(v(t) - w_{\text{i,in}}) \\ w_{\text{i,out}}(t + \Delta t) &= w_{\text{i,out}}(t) + \eta\xi\,(\zeta(t) - w_{\text{i,out}}) \end{aligned} \tag{2.2}$$

where η is the learning rate, $\xi(\cdot)$ is a neighborhood function and the index i spans through the neighbors of the winner neuron s.

- **Step 4** – Update the mean increase in the reward function:

$$b_s^{\text{new}} = b_s^{\text{old}} + \gamma \left(\Delta R - b_s^{\text{old}} \right) \tag{2.3}$$

where γ is the learning rate; update a_i of neuron s and its neighboring neurons:

$$a_i^{\text{new}} = a_i^{\text{old}} + \eta_a \xi_a \left(a - a_i^{\text{old}} \right) \tag{2.4}$$

where η_a is the learning rate, ξ_a is the neighborhood function, a is a threshold value and the index i spans, again, through the neighbors of the winner neuron s.

- **Step 5** – Repeat steps 1) - 4). If $a = 0$ the learning phase stops when the weights converge.

2.2 Control system architecture

Most of researches on locomotion control in insects reveal the presence of a hierarchical organized neural system [38]. Therefore, following a bio-inspired approach, most of the proposed control schemes for legged robots use a hierarchical organization [8].
Accordingly, we subdivided the locomotion control into two levels: a low level control implementing a Central Pattern Generator and an high level control implementing basic behaviors. In this section we will discuss in detail both the low level and the high level control.

2.2.1 CNN-CPG and leg dynamics

The basic units of the adopted artificial CPG are nonlinear oscillators coupled together to form a network able to generate a pattern of synchronization that is used to coordinate the robot actuators. The dynamics of each oscillator can be efficiently exploited to control the leg kinematics of an insect-like hexapod robot by carefully mapping oscillator limit cycles in the limit cycles performed by legs in the joint space.

Cellular Nonlinear Network paradigm, introduced in [12], provides a framework for the implementation of these nonlinear oscillators: each oscillator is simply viewed as a cell of a CNN.

This technique has been used to control the locomotion of several different bio-inspired robotic structures: hexapods, octopods and lamprey-like robots [16, 3]. One fundamental advantage of CNN implementation is that a direct VLSI realization of the control system is possible: a chip for locomotion control implemented by a CNN-based CPG is introduced in [6]. Further advantages are ease of implementation, flexibility and modularity, allowing an arbitrarily large number of actuators to be controlled.

The following equations describe the nonlinear oscillator (CNN cell) acting as a neuron of the artificial CPG:

$$\begin{cases} \dot{x}_1 = k(-x_1 + (1+\mu)y_1 - sy_2 + i_1 + \sum_s I_{1,s}) \\ \dot{x}_2 = k(-x_2 + sy_1 + (1+\mu)y_2 + i2 + \sum_s I_{2,s}) \end{cases} \tag{2.5}$$

where $y_i = \frac{1}{2}(|x_i + 1| - |x_i - 1|)$ with $i = 1, 2$. The terms $\sum_s I_{1,s}$ and $\sum_s I_{2,s}$ represent the sum of all the synaptic inputs coming from the other neurons, *i.e.* represent the interconnection with the other neurons. For the choice of the parameters given in Tab. 2.2, system (2.5) admits a periodic solution with slow-fast dynamics (in particular the outputs y_1 and y_2 perform a unitary and square-shaped limit cycle): these regular oscillations provide the rhythmic movements for the robot actuators.

Table 2.2: CNN Parameters

μ	s	i_1	i_2	k
0.5	1.2	-0.3	0.3	$\frac{10}{3}$

To coordinate the movements it is necessary to properly synchronize the CPG neurons. This can be done by establishing suitable connections among the nonlinear oscillators, as discussed in [2].

As stated above, legs perform in the joint space a limit cycle, therefore it is natural to map the dynamics of the neuron into leg dynamics through suitable transformation functions. In [2] this approach is applied to a robot equipped with identical legs based on a pantograph mechanism.

GregorI actuation is more complex, since each leg has a unique design; anyway it is still possible to adopt a CNN-CPG with identical cells, basing on the following considerations. As far as basic locomotion is concerned, we can just actuate the α and β joints in front and middle legs and α and d joints in rear legs, since the γ joint actuation is needed just for postural adjustments. The key point is that these joints play an analogous role during locomotion (forward movement and raising respectively) and, therefore, perform a similar limit cycle in the joint space.
Thus, we can still consider, through *ad hoc* transformations, a mapping between CNN cell limit cycles and leg dynamics limit cycles.

In detail, the CPG is composed by six neurons, each one controlling through its two outputs y^1_{CPG} and y^2_{CPG} a leg (the α and β joints in front and middle legs and α and d joints in rear legs). The CNN outputs do not directly control the actuators; they instead undergo a two stages transformation in order to fit the peculiar leg design.

In the first stage, the identical and unitary limit cycles performed by CNN cells are mapped through the maps Z_{front}, Z_{middle} and Z_{rear} into three different unitary limit cycles. These transformations are graphically shown in Fig. 2.1. After this stage, the following new outputs for front, middle and rear legs are obtained:

$$\zeta_i^{front} = Z_{front}\left(y_i^{front}\right) \tag{2.6a}$$

$$\zeta_i^{middle} = Z_{middle}\left(y_i^{middle}\right) \tag{2.6b}$$

$$\zeta_i^{rear} = Z_{rear}\left(y_i^{rear}\right) \tag{2.6c}$$

In the second stage the new outputs are differently scaled and biased and the suitable actuation

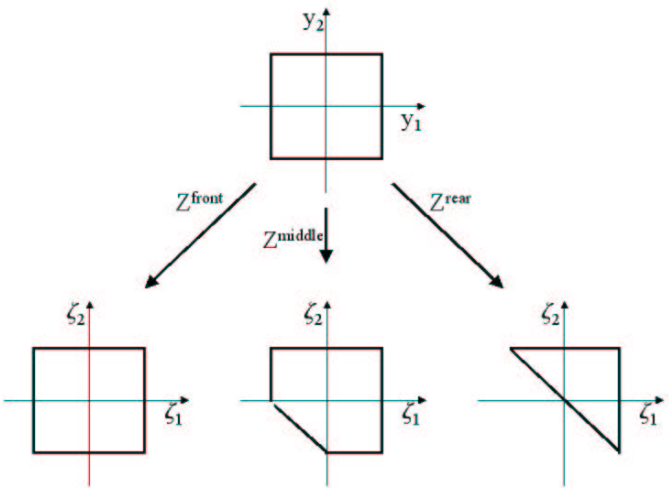


Figure 2.1: CPG signals transformation.

signals are obtained:

$$\alpha_{front} = a_{front}\zeta_2^{front} \tag{2.7a}$$

$$\beta_{front} = b_{front}\zeta_1^{front} \tag{2.7b}$$

$$\alpha_{middle} = a_{middle}\zeta_2^{middle} \tag{2.7c}$$

$$\beta_{middle} = b_{middle}\zeta_1^{middle} \tag{2.7d}$$

$$\alpha_{rear} = \lambda + a_{rear}\zeta_2^{rear} \tag{2.7e}$$

$$d_{rear} = b_{rear}\zeta_1^{rear} \tag{2.7f}$$

Synchronization is achieved through suitable connections among the neurons depending on the adopted gait, as discussed in [2]. Actuation values are shown in Tab. 2.3

Table 2.3: Actuation parameters

a_{front}	b_{front}	a_{middle}	b_{middle}	λ	a_{rear}	b_{rear}
0.6 rad	0.9 rad	0.4 rad	0.7 rad	0 rad	0.2 rad	6 cm

2.2.2 High level control

High level control has to deal with complex tasks like climbing obstacles or looking for a target. There are four basic classes of high level control for autonomous mobile robots, that can be briefly

summarized as follows:

- **Reactive control** – Don't think, act.
- **Deliberative control** – Think hard, then act.
- **Hybrid control** – Think and act independently, in parallel.
- **Behavior-based control** – Think the way you act.

Reactive control tightly couples sensory inputs and effector outputs, to allow the robot to quickly respond to changing and unstructured environments, according to a biological "stimulus-response" scheme. In *deliberative control*, the robot uses all of the available sensory information and all of the internally stored knowledge to reason about what actions to take, where reasoning is typically in the form of a planning algorithm. *Hybrid control* combines the real-time response of reactivity with the rationality and optimality of deliberation. As a result, the control system contains two different components, the reactive and the deliberative ones, which must interact in order to produce a coherent output. Finally, *behavior based systems* are based on a representational substrate, the *behaviors*, which are observable patterns of activity emerging from interactions between the robot and its environment. Like hybrid systems, behavior based systems may have different layers, but the layers do not differ drastically in terms of time-scale and representation used [26].

Behavior-based systems have been praised for their robustness and simplicity of construction [28] and seem to be a really suitable approach to perform autonomously exploration task in a unknown environment. Therefore, in our control scheme we have adopted a behavior-based approach.

Basically, behavior-based systems are composed of a collection of "behavior producing" modules that map environment states into low-level actions, and a coordination mechanism that decides, basing on the state of the environment, which behavior has to be executed. Some learning algorithms have been proposed both for the behavior learning task and the coordination task. However, the task of programming in each individual behavior remains the burden of a human designer since it requires a deep knowledge of the interactions between a particular robot and its application to the environment, typically not available if an exploration mission is considered [28].

In this thesis, we focus the attention on behavior formalization, since it plays the crucial role to achieve an adaptive control scheme. On the contrary, the behavior coordination issue exceeds the objectives of this work.

2.2.3 Behavior formalization

Different ways have been proposed to implement a behavior, like stimulus-response diagrams, mathematical functional and finite state machines. Basically, a behavior is such any mapping from possible stimuli to possible responses. The difference with a reactive control is that behaviors do not require a tight coupling between stimuli and actions in the form of a *look up* table.

Since an adaptive mapping between stimuli and actions is needed, we chose to formalize a behavior as a Motor Map. For example, the behavior "Avoiding obstacle" is translated in a Motor Map whose input is the obstacle distance and whose outputs are the motor commands. In this way it is possible to learn a behavior through the classical Motor Map learning. Motor Map Behaviors (MMB) are thus adaptive and self organizing. On the other hand, if the robot has to explore dynamic, hazardous and unknown environments, the definition of the Motor Map reward function could represent a challenging task. Let us consider, for example, the "Pursuing target" behavior and let us suppose that the robot measures the intensity of two different chemicals: sensor performance could be not known in advance for the target environment; moreover, in case of sensor malfunctioning, the reward function could take into account a wrong signal with detrimental consequences for the behavior execution. As a consequence, the need of an adaptive reward arises.

We therefore introduce a learning system for the reward function of a MMB. To the best of our knowledge, it is the first time that an adaptive reward is considered. Aim of the adaptive reward is to select the most significant sensory inputs and to combine them in the best way. The most important challenge for this approach, as it will be discussed in the next session, is to keep small the search space.

2.2.4 Reinforcement learning for MMB reward: overview

For MMB reward learning we exploit the classical reinforcement learning theory, that requires just an external feedback.

Reinforcement learning is usually formulated mathematically as an optimization problem with the objective of finding an action, in our case a *reward function*, that is optimal in some well-defined way. Optimality objectives provide a useful categorization of reinforcement learning into three basic types, in order of increasing complexity: non-associative, associative, and sequential. *Non-associative* reinforcement learning involves determining which of a set of actions is best in bringing about a satisfactory state of affairs. In *associative* reinforcement learning, different actions are best in different situations. The objective is to form an optimal associative mapping between a set of stimuli and the actions having the best immediate consequences when executed in the situations signaled by those stimuli (by the way, Motor Maps belong to this type). *Sequential* reinforcement learning retains the objective of forming an optimal associative mapping but is concerned with more complex problems in which the relevant consequences of an action are not available immediately after the action is taken [30].

The MMB reward learning can be achieved just with a *non-associative reinforcement* learning, since the reward function form does not depend on the the particular learning example, *i.e.* it does not make any difference if for the MMB reward learning we consider, referring for example to the "Climbing obstacle" behavior, an obstacle with height x instead of an obstacle with height $x + \Delta x$.

Fig. 2.2 shows the basic components of a non-associative reinforcement learning. The learning

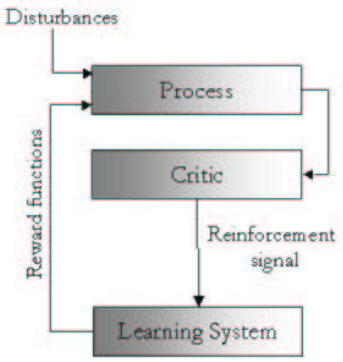


Figure 2.2: Non-associative reinforcement learning [30].

system's actions influence the behavior of some process, which might also be influenced by random or unknown factors (labelled "disturbances" in Fig. 2.2). A critic sends the learning system a reinforcement signal whose value at any time is a measure of the "goodness" of the current process behavior. Using this information, the learning system updates its action-generation rule, generates another action, and the process repeats [30]. In our framework, the action is the reward function form and the signal critic is the evaluation of behavior execution.
Unfortunately, *before* a specific reward function form can be tested, the underlined Motor Map has to be trained. Thus, at each MMB reward learning step, the learning system has to:

- select a reward function form;
- train the Motor Map with the selected reward function;
- evaluate the critic signal;
- update reward-generation rule.

We added to the classical non-associative learning algorithm the further step represented by Motor Map training, necessary to test a selected reward form. At first glance, Motor Map training represents a prohibitive obstacle, since it makes MMB reward learning excessively long. Luckily, it is enough to train the Motor Map *just in one input configuration*; in fact, coherently with the fact that a non associative learning is used, reward function form can be tested referring just to a particular learning example.

2.2.5 Reinforcement learning for MMB reward: algorithm

Let us consider a behavior p [1]; to this behavior we associate a set S of basic reward units r_j, basing on some guidelines discussed later. The S set induces, according to the exclusivity rule described below, a general reward function:

$$R = \sum_{j}^{N} -\alpha_j r_j^2 \tag{2.8}$$

[1] For sake of clarity, henceforth we will suppress the index p.

where N is the total number of basic units and α_j is a binary weight (*i.e.* α can take values 0 or 1).

Eq. 2.8 represents a family of reward functions. A particular reward function is obtained by setting α_j weights to the values 0 or 1; we will refer to a vector like $(1, 0, 1, 1 \ldots 0)$ as a reward instantiation k and we will denote it with $\bar{\alpha}_k$. The problem is how to determine the optimal reward instantiation, *i.e.* the reward the best fits the environment.

Let us now associate to a reward instantiation k a success probability d_k: d_k is the probability that the execution of behavior p succeeds, given that the controlling Motor Map with the reward function induced by $\bar{\alpha}_k$ has been trained and used. Each d_k can be any number between 0 and 1 (the d_ks do not have to sum to one), and the learning system has no initial knowledge of these values. The learning system's objective is to asymptotically maximize the probability that a behavior is successfully performed, which is accomplished when it is always used the reward function induced by the instantiation $\bar{\alpha}_h$ such that $d_h = \max(d_k | k = 1, \ldots, M)$, where M is the total number of instantiations.

Suppose now that, on each trial, the learning system selects a reward instantiation $\bar{\alpha}_k$ from the set of reward instantiations $\mathcal{R}$ according to a probability vector $(p_1(t), \ldots, p_M(t))$, where $p_k(t) = \Pr(\bar{\alpha}(t) = \bar{\alpha}_k(t))$, *i.e.* the probability that at trial t the actual reward instantiation is the k^{th} one.

The proposed learning algorithm performs the following linear reward-penalty (L_{R-P}) method [30]: if instantiation k is chosen on trial t and the critics feedback is "success", then $p_k(t)$ is increased and the probabilities of the other reward instantiations are decreased; whereas if the critic indicates "failure", then $p_k(t)$ is decreased and the probabilities of the other reward instantiations are appropriately adjusted.

In detail, if $\bar{\alpha}(t) = \bar{\alpha}_k(t)$ and, after Motor Map training, the critic says "success", then

$$\begin{aligned} p_k(t+1) &= p_k(t) + \beta(1 - p_k(t)) \\ p_l(t+1) &= (1-\beta)p_l(t), \quad l \neq k \end{aligned} \tag{2.9}$$

If $\bar{\alpha}(t) = \bar{\alpha}_k(t)$ and, after Motor Map training, the critic says "failure", then

$$\begin{aligned} p_k(t+1) &= (1-\gamma)p_k(t) \\ p_l(t+1) &= \frac{\gamma}{M-1} + (1-\gamma)p_l(t), \quad l \neq k \end{aligned} \tag{2.10}$$

where $0 < \beta < 1$ and $0 \leq \gamma < 1$.

In this way, after a transient, the most suitable reward instantiations (*i.e.* the reward instantiations with the highest d_k) are detected.

2.2.6 S set and $\mathcal{R}$ set

If we assume that the robot does not create new sensors, all sensory information available to the robot are known *a priori*; what is unknown and has to be learnt is how to combine all sensory inputs. Therefore the definition of the basic reward units is indeed simple and can be made in the designing phase as follows.

Firstly, we have to associate to a behavior just the sensory information on which a behavior could indeed rely; *e.g*, it does not make sense to consider for the "Climbing obstacles" behavior a smell information. Once a proper set of sensory information is formed, we have to build the basic reward units.
Toward this end, sensory information can be divided into four types:

- $\mathbf{U_1}$ – proprioceptive information with exteroceptive analogue, henceforth denoted π_+, *e.g* tarsus height and obstacle height.
- $\mathbf{U_2}$ –proprioceptive information without exteroceptive analogue, henceforth denoted π_-, *e.g* Euler angles.
- $\mathbf{U_3}$ –exteroceptive information with proprioceptive analogue, henceforth denoted η_+, *e.g*, trivially, obstacle height and tarsus height.
- $\mathbf{U_4}$ –exteroceptive information without proprioceptive analogue, henceforth denoted η_-, *e.g* smell.

This subdivision is correct, since, trivially, $\bigcup_{i=1}^{4} \mathbf{U}_i = \mathbf{U}$ and $\mathbf{U}_i \cap \mathbf{U}_j = 0$ if $i \neq j$.
According to this subdivision, possible types of basic reward units are:

- a) – combination between analog proprioceptive and exteroceptive information: $r_i = f_i(\pi_+ - \pi_-)$ (and its inverse), *e.g* $r_i = h_{tarsus} - h_{obs}$ where h_{tarsus} is the tarsus height and h_{obs} is the obstacle height;
- b) – comparison between a space-varying or time-varying information I and its maximal (or minimal) value, if it exists: $r_i = f_i(I - I_{max})$ (and its inverse), *e.g.* $r_i = h_{tarsus} - h_{max}$, where h_{max} is the maximum tarsus height;
- c) – maximization (or minimization) of a space-varying or time-varying information I: $r_i = f_i(I)$ (and its inverse), *e.g.* $r_i = 1/I$;
- d) – tracking of a reference I_{ref}: $r_i = f_i(I - I_{ref})$, *e.g.* $r_i = v - v_{ref}$, where v is robot speed and v_{ref} is a speed reference.

Clearly, S set definition is somewhat arbitrary; the key point is that the designer, basing on the fact that he knows which are the available information (the signals coming from the sensors placed on the robot), can easily decide generality and potentiality of the reward family $\mathcal{R}$; the reward learning system will find during the mission the best choice for the unknown environment. Therefore, in this design stage, the designer has to accomplish two competing tasks:

- placing all his available *a priori* information in order to minimize the search space;
- guaranteeing behavior adaptability.

Once S set is formed, it is straightforward to form the $\mathcal{R}$ set. One rule to follow, in order to avoid ridiculous rewards, is the obvious exclusivity rule: *in a reward instantiation the direct form and inverse form of a basic reward unit can not be present at the same time.*

2.2.7 Remarks

Uncertainty

Uncertainty plays a key role in non-associative reinforcement learning. For example, if the critic in the example above evaluated actions deterministically (i.e., $d_k = 1$ or 0 for each k), then the problem would be a much simpler optimization problem.

Critic

The critic is an abstract entity that evaluates the learning systems actions. The critic does not need to have direct access to the actions or have any knowledge about the interior workings of the process influenced by those actions. The critic has just to measure the "goodness" of the current behavior according to some criteria; clearly, critic criteria depend on the specific robot behavior currently under learning. In the next chapter, critic criteria for different behavior learning examples will be discussed.

Memory

The vector probability represents a sort of memory of past actions; this, in turn, it is fundamental as far as the adaptability of the approach is concerned. If, in fact, due to sensory malfunctioning or different environmental conditions the best reward function changes structure, the new learning phase is greatly facilitated by previous trials.

Chapter 3

Simulation results

In this chapter we study the effectiveness of the proposed structure and control architecture. In order to asses the suitability of the structure, we firstly present simulation results for horizontal walking and obstacle climbing, obtained by just exploiting in the control system architecture the CPG level. Then we test the overall control system with the learning of some classical behaviors.

3.1 Simulation environment

The dynamic robot model was tested in a C++ environment based on `DynaMechs` library [27]. The library efficiently simulates the dynamics of robotic articulations and provides a comfortable framework to translate in C++ the control system architecture. The overall C++ program is available upon request.

3.1.1 Environmental properties

Environmental properties to be set are: Ground Normal Spring Constant k_N, Ground Planar Spring Constant k_P, Ground Normal Damper Constant γ_N, Ground Planar Damper Constant γ_P, Coefficient of Static Friction μ_s and Coefficient of Kinetic Friction μ_d.

Ground normal and planar spring and damper constants are used to define how the robot interacts with the surface. The values chosen for GregorI are typical values for a hard terrain. Coefficient of Static Friction and Coefficient of Kinetic Friction model sliding across the surface.

The chosen values for friction parameters are typical values for a normal terrain. All values are shown in Tab. 3.1

Table 3.1: Terrain properties

k_N	g/s^2	k_P	g/s^2	γ_N	g/s	γ_P	g/s	μ_s	μ_d
75000		75000		2000		2000		1.5	1

3.1.2 Integration algorithm

Runge-Kutta of 4^{th} Order was selected for GregorI simulation as it provides a good numerical approximation with acceptable computational overhead. A step size of step = 0.001 was found to be appropriate for the GregorI simulation, as values larger than this may cause the controller to become unstable. All simulations were conducted on a 2.8 GHz Pentium class machine, running Microsoft Windows XP. On this machine, 10 seconds of dynamical simulation correspond to 4 seconds of computer computation.

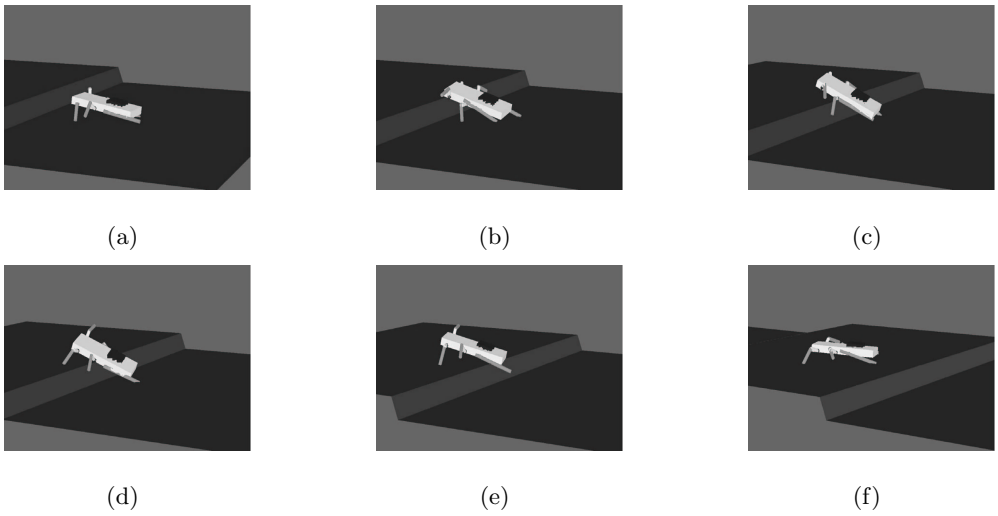
3.2 Horizontal walking and climbing obstacles

Firstly we tested the structure functionality and stability in two conditions: horizontal walking and climbing obstacles. Since in these simulations our main concern was on structure testing, we just took into account the low level control, *i.e.* the CPG controller. In all simulations we considered a fast gait.

Unfortunately, there is not enough performance detail documented in the published robotics literature to rigorously compare GregorI structure to previous structure. Thus structure analysis is somewhat qualitative. Horizontal walking simulations show, overall, a great movement agility.

We can gain more insights into the efficiency of the proposed structure by simulating the robot during obstacle climbing. Several simulations showed that thanks to the particular leg-body articulation and, above all, to the innovative linear piston-like actuation of rear legs, GregorI is able to overcome obstacles even beyond its CoM height (at least up to 5 cm) without any postural

adjustment. This result proves the high stability of the overall structure and the efficiency of rear legs thrust. In Fig. 3.1 some snapshots of GregorI climbing a 5 cm obstacle are shown. It is worth noticing how the rear legs propel the robot forward.



(a) (b) (c)

(d) (e) (f)

Figure 3.1: Obstacle climbing snapshots.

3.3 Learning behaviors

In this section we test the capabilities of the proposed high level control. We focus, in particular, the attention on learning "Climbing obstacle" behavior and "Pursuing target" behavior; the former is more interesting from a theoretical viewpoint, while the latter is more interesting from a practical viewpoint. In all simulations we consider a fast gait. We also assume that there are two chemicals originating at the target location.

Let us suppose that the following information are available to the robot:

- front tarsus height $h_{front} \in \pi_+$;
- body rear part height $h_{post} \in \pi_+$;
- obstacle height $h_{obs} \in \eta_+$;
- chemical 1 intensity $I_1 \in \eta_-$;

- chemical 2 intensity $I_2 \in \eta_-$.

The first step is associating to each behavior a Motor Map. The climbing Motor Map has:

- 12 neurons;
- 1 input (the obstacle height);
- 2 outputs (front femur joint gain b_{front} and rear coxa joint bias λ).

The Pursuing Motor Map has:

- 12 neurons;
- 2 input (I_1 and I_2 intensity);
- 1 output (yaw angle in the inertial reference frame Ξ).

Both Motor Maps possess the same learning parameters. In order to simplify the learning phase, we consider a winner-take-all strategy by selecting unitary neighborhood functions $\xi(\cdot)$ and $\xi_a(\cdot)$. The threshold value is $a = 0.01$, so that, after the learning phase, a residual plasticity for a later re-adaptation is guaranteed. The learning rate is $\eta = 0.5$ as a trade-off between speed and accuracy of learning, while the two adaptive rates are $\gamma = 0.1$ and $\eta_a = 0.05$.

The second step requires S set and $\mathcal{R}$ set definition for each Motor Map. It is natural to consider for the S set concerning the Climbing Motor Map just the sensory information h_{front}, h_{post} and h_{obs}, while for the S set concerning the Pursuing Motor Map just I_1 and I_2. Following the guidelines outlined in the previous chapter, we define the S set for the Climbing Motor Map as:

$$S_{climbing} = \{(h_{obs} - h_{front}), (h_{obs} - h_{post}), 1/(h_{obs} - h_{front}), 1/(h_{obs} - h_{post})\} \tag{3.1}$$

In the same way, we define the S set for the Pursuing Motor Map as:

$$S_{climbing} = \{I_1, I_2, 1/I_1, 1/I_2\} \tag{3.2}$$

From the S set we can easily derive the corresponding $\mathcal{R}$ set basing on the exclusivity rule. Considering a instantiation vector notation, both $\mathcal{R}$ sets have the same following elements:

- $1, 0, 0, 0$
- $0, 0, 1, 0$
- $0, 1, 0, 0$
- $0, 0, 0, 1$
- $0, 0, 1, 1$
- $0, 1, 1, 0$
- $1, 0, 0, 1$
- $1, 1, 0, 0$

The final step is defining the critic signal: trivially, the critic for the Climbing Motor Map indicates a "success" if after an evaluation time t_{max} the robot is over the obstacle, while the critic for the Pursuing Motor Map indicates a "success" if, after an evaluation time t_{max}, the distance from the target is below a threshold $d_{treshold}$.

Behavior learning simulations are articulated as follows: at the beginning the robot is placed in a starting position; a reward instantiation k is selected and the Motor Map is congruently trained for 100 epoches with the same particular learning example (as a consequence of the non-associative learning of the reward function). Each epoch lasts four times the cycle time, *i.e.* in one epoch the robot performs four steps. At the end of each epoch the robot is placed again at the starting position for ease of simulation. After 100 epoches the reinforcement signal is evaluated and a new *reward learning epoch* begins, with different initial conditions in robot configuration in order to provide *uncertainty*.

The simulation ends if a $p_k > 0.9$ is found, otherwise the simulation keep running for ever. We set the reward structure learning constants to the values $\beta = 0.5$ and $\gamma = 0.6$.

We are now ready to analyze the learning results.

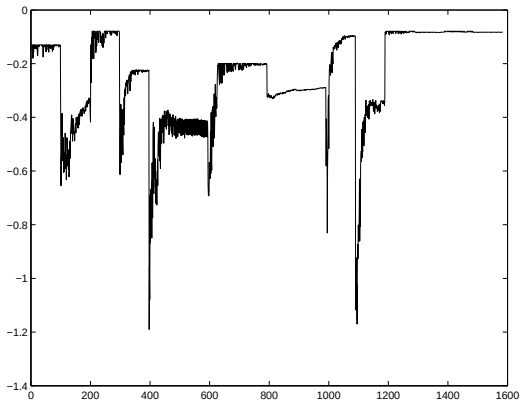


Figure 3.2: Climbing Motor Map training.

3.3.1 Climbing obstacle learning

Setting $t_{max} = 55s$ we obtain that, on 10 different simulations, there is not a prevalent probability p_k; there are, instead, three preferred behaviors corresponding to the following reward instantiations:

- $1, 0, 0, 0$
- $1, 0, 0, 1$
- $1, 1, 0, 0$

The first instantiation induces a rearing strategy, the second one induces a slightly different rearing strategy, while the last one induces an elevate strategy.

Setting, instead, $t_{max} = 35s$, *i.e.* a stricter critic signal, we obtain that the rearing strategy tends to be, on average, prevalent (8 times over 10), with a surprising congruence to the biological case.

In Fig. 3.2, the training of the Climbing Motor Map, *i.e.* the trend of the relative reward function, is shown; on the x-axis epoch counter is represented. The sharp change in the reward trend is due to the change after 100 epoches of the reward structure. In Fig. 3.3 temporal evolution of $S_{climbing}$ set is reported, referring to a simulation with $t_{max} = 35s$; lighter bar refers to the

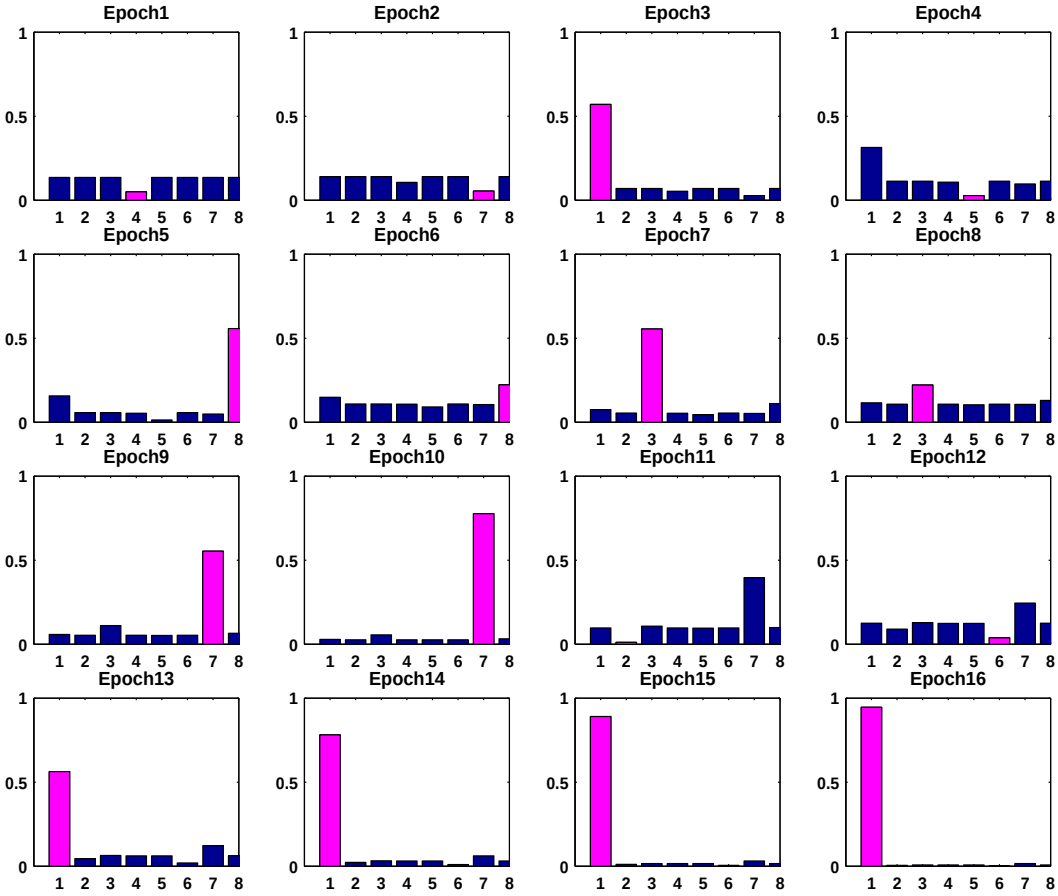


Figure 3.3: Temporal evolution of $S_{climbing}$ set.

selected reward instantiation. It is worth noticing how the uncertainty affects the learning phase: for example, at reward learning epoch 5, reward instantiation 8 leads to a success, while in the next reward learning epoch the same instantiation leads to a failure.

3.3.2 Pursuing target learning

Firstly we assume that chemical 1 follows a gaussian distribution centered in the target location, while the chemical 2 is constant, thus not providing any information. The evaluation time is $t_{max} = 25$ s.

It is easy to notice that the suitable reward instantiations are:

- $0, 0, 1, 0$
- $0, 0, 1, 1$
- $0, 1, 1, 0$

All instantiations, in this case, induce a strategy that leads to a maximization of the chemical 1 intensity. Congruently, in simulation we obtain that all and only the three instantiations above are selected as suitable reward candidates.
In Fig. 3.4, the training of the Pursuing Motor Map, *i.e.* the trend of the relative reward function, is shown; on the x-axis epoch counter is represented. Again, the sharp change in the reward trend is due to the change after 100 epoches of the reward structure.

In Fig. 3.5 the temporal evolution of $S_{pursuing}$ set is shown; lighter bar refers to the selected reward instantiation. In the reported case, instantiation 6 becomes prevalent.

In a second experiment we assume that both chemicals spread with the same gaussian distribution, but the detector of second chemical measures with some noise. The evaluation time is again $t_{max} = 25$ s. Simulation results show that the robot is able to learn to disregard the second chemical and to just rely on the maximization of the first chemical by selecting the instantiation 2. This last example shows **the effectiveness of the proposed high level control in case of sensory malfunctioning**.

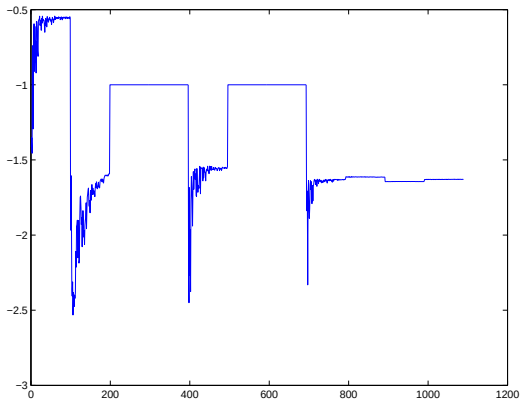


Figure 3.4: Pursuing Motor Map training.

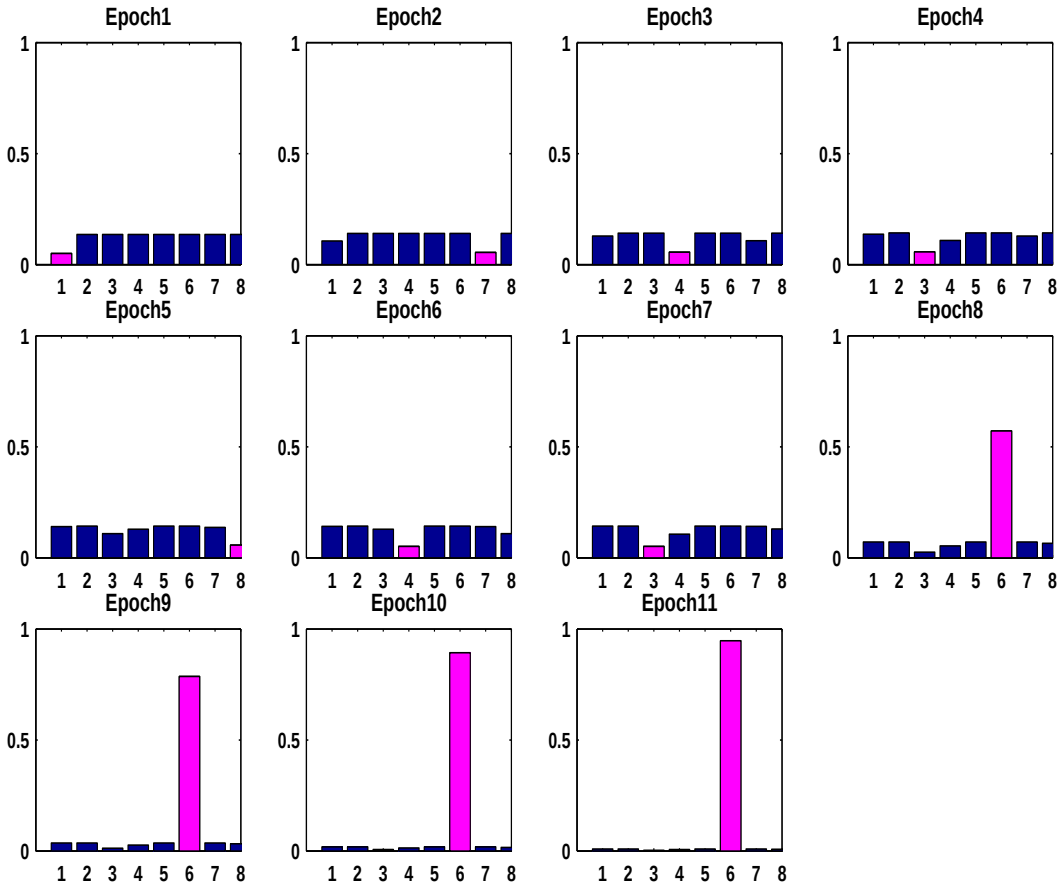


Figure 3.5: Temporal evolution of $S_{pursuing}$ set.

Chapter 4

Cyclic-pursuit approach to multi-agent coverage path planning

We now turn our attention to robot coordination. A prototypical problem that could benefit from a multi-agent approach is coverage path planning in hazardous environments, where the aim is to sweep all points in the target environment facing disturbances and agent losses. Specifically, we study the following motion coordination problem: given n non-holonomic robots arbitrarily deployed within a convex region in the plane, develop a static (*i.e.*, memoryless) decentralized control strategy able to guarantee efficient path coverage of the region, taking into account failures and the possible loss of one or more agents.

4.1 Path planning coverage: overview

Focusing firstly on a single-agent approach, most of coverage algorithms rely either implicitly or explicitly on a cellular decomposition of the free space to complete the task. A cellular decomposition breaks down the target region into cells such that coverage in each cell is simple. Provably complete coverage is attained by ensuring the robot visits each cell in the decomposition [11]. One popular exact cellular decomposition technique, which can yield a complete coverage path solution, is the trapezoidal decomposition [21]. Since each cell is a trapezoid, coverage in each cell can easily be achieved with simple back and forth motions. Coverage of the environment is

achieved by visiting each cell in the adjacency graph. An enhancement of trapezoidal decomposition is represented by the boustrophedon cellular decomposition [10], designed to minimize the number of excess lengthwise motions. The Backtracking spiral algorithm makes use of spiral filling paths instead of back and forth motion [17].
Unfortunately, grid maps based algorithms require considerable memory, centralized off-line computation and costly localization sensors (*e.g.*, GPS).
On the contrary, heuristic navigation methods, as presented in [18] for path planning of an autonomous mobile cleaning robot and in [23] for a multi agent search-and-retrieve object application, allow the use of far simpler robots, but do not guarantee complete coverage.
Some algorithms have been also proposed for a multi-agent approach, like in [43], where a set of robots help clean a railway station, using magnetic lines on the floor as guidelines. [20] suggests an off-line multi-robot coverage strategy using a Voronoi diagram-like and boustrophedon approach. They define a cost function to pseudo-optimize the collective coverage task. In [40] a decentralized covering algorithm is presented for covering of an un-mapped region by means of a group of robots leaving traces: here, issue of memorization of already covered regions is solved trough abstract chemical traces that play the role of a shared memory.

Our main concern is to study the feasibility of a coverage algorithm for non-holonomic robots that:

1. does not rely on costly grid maps (unlike Cellular Decomposition approach)
2. provides complete coverage (unlike heuristic approach)
3. takes into account only local sensory information, thus avoiding inter-agent communication, landmark deployment and other possibly costly mechanisms for information exchange.

Our main assumption, necessary to achieve proof of correctness, is that robots can overcome all obstacles placed in the environment without modifying their desired trajectory. The key idea is to consider a modified version of the classical cycling pursuit control strategy for non-holonomic robots, previously extensively studied in [25], where it is shown that system's equilibrium formations are generalized regular polygons. In our strategy, instead of pursuing the leading neighbor along the instantaneous line of sight, each agent pursues its leading neighbor along the line of sight

rotated by an offset angle, function of locally available sensory information. It is shown that the paths described by the system at equilibrium are Archimede's spirals able to provide complete coverage of the target environment. These spiral-like paths appear to be robust against sensory noise, odometry error and agent loss.

4.2 Problem formulation

Let us consider n ordered robots; we associate to each robot a sensing region, modelled as a circle of radius d, attached to the robot frame. The objective is to design a static, spatially-decentralized feedback control policy that guarantees that the union of the sets swept by the sensing regions as robots move covers the whole environment. In order to do this efficiently, we desire to exploit the intrinsic parallelism of a multi-vehicle approach, under the following assumptions:

1. robots are unable to recall past actions and observations (i.e.,they are oblivious);
2. no robot can access the absolute positions of other robots or its own. Specifically, robot i can measure only the relative positions of neighbor robots and their relative orientation;
3. robots are unable to communicate in any other ways than by observing each others position;
4. robots i pursues the next, $i+1$, modulo[1] n;
5. circular environment with surmountable obstacles (more complex environment will be analyzed in future).

4.3 Formation control for holonomic robots

Let us firstly model the agents as freely mobile robots, *i.e.* holonomic robots. In the next section we will extend our results to non-holonomic robots with a single *Pfaffian* constraint.
Let us, therefore, consider n ordered holonomic agents $\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_n$; the robot dynamics is simply:

$$\dot{\mathbf{h}}_i = \mu_i \tag{4.1}$$

[1]Henceforth, all vehicle indices should be evaluated modulo n.

where μ_i is a reference speed function of locally available information.
Considering the geometry of the problem, if all agents perform Archimede's spiral-like trajectories, complete coverage is guaranteed. Therefore, our desired feedback control law for the n dynamical systems (4.1) must guarantee Archimede's spiral-like trajectories and be robust against robots losses in such a way that, when an agent disappears from the arena, a new spiral-like formation is achieved without leaving uncovered any portion of the environment. Since each sensor has a circular footprint with radius d, the feedback control law should provide Archimede spiral-like trajectories with step at most equal to $2nd$.

Let us assume, as stated above, that each robot does not have memory of the past (static feedback) and can only measure the relative position of the leading robot:

$$\mathbf{r}_i = \mathbf{h}_{i+1} - \mathbf{h}_{i+1} \tag{4.2}$$

and the angle:

$$2\gamma_i = \arccos\left(\left\langle \frac{(\mathbf{h}_{i+1} - \mathbf{h}_i)}{\|(\mathbf{h}_{i+1} - \mathbf{h}_i\|}, \frac{(\mathbf{h}_i - \mathbf{h}_{i-1})}{\|(\mathbf{h}_i - \mathbf{h}_{i-1})\|} \right\rangle\right) \tag{4.3}$$

The key idea is to consider a modified version of the classical cycling pursuit control strategy: instead of pursuing the front neighbor along the instantaneous line of sight, each agent pursues its front neighbor along the line of sight rotated by an offset angle, function of (at most) $\mathbf{r}_i$ and γ_i, *i.e.*:

$$\dot{\mathbf{h}}_i = \lambda R(\alpha_i)\mathbf{r}_i \tag{4.4}$$

where

$$\alpha_i = \alpha(\|\mathbf{r}_i\|, \gamma_i) \tag{4.5}$$

Let us now pose:

$$\alpha_i = \arctan\left(\frac{2d\sin\gamma_i}{\|\mathbf{r}_i\|\gamma_i}\right) + \gamma_i \tag{4.6}$$

note that $\alpha(\|\mathbf{r}\|, \gamma)$ is differentiable for $r \neq 0$ and can be made continuous everywhere by setting $\alpha(\|\mathbf{r}\|, 0) = \arctan(2d/\|\mathbf{r}\|)$.
Before stating our main theorem, we define a regular configuration as follows.

Definition 4.3.1. A regular configuration is a configuration where $2\gamma_i = \frac{2\pi}{n}$ and $\|\mathbf{r}_i\| = \|\mathbf{r}_j\| \quad \forall i, j$.

Theorem 4.3.1. *Let us assume that the initial configuration is a regular configuration. Then the feedback control law:*

$$\mu_i = \lambda R(\alpha_i)\mathbf{r}_i \tag{4.7}$$

guarantees that each robot performs Archimede's spiral-like trajectories with step $\lambda = 2 \cdot n \cdot d$

Proof. To prove the claim, we should prove that:

$$\lambda = \int_0^{2\pi} \frac{\mathrm{d}\varrho_i}{\mathrm{d}\varphi_i}\mathrm{d}\varphi_i = 2 \cdot n \cdot d \tag{4.8}$$

Let us express the position of the i^{th} robot in polar coordinates:

$$\begin{aligned} h_x^i &= \varrho_i \cos\varphi_i \\ h_y^i &= \varrho_i \sin\varphi_i \end{aligned} \tag{4.9}$$

Hence, deriving with respect to time we get:

$$\left[\begin{array}{c} \dot{\varrho}_i \cos\varphi_i - \varrho_i\dot{\varphi}_i \sin\varphi_i \\ \dot{\varrho}_i \sin\varphi_i + \varrho_i\dot{\varphi}_i \cos\varphi_i \end{array} \right] = \mu_i \tag{4.10}$$

Let us now derive an expression of μ_i in polar coordinates. Basing on the definition of regular configuration and on the symmetry of the problem we can write $\varrho_{i+1} = \varrho_i$ and $\varphi_{i+1} = \varphi_i + 2\gamma_i$; hence we can express μ_i as follows:

$$\mu_i = \lambda \left(\begin{array}{cc} \cos\alpha_i & \sin\alpha_i \\ -\sin\alpha_i & \cos\alpha_i \end{array} \right) \left(\begin{array}{c} \varrho_i \cos(\varphi_i + 2\gamma_i) - \varrho_i \cos(\varphi_i) \\ \varrho_i \sin(\varphi_i + 2\gamma_i) - \varrho_i \sin(\varphi_i) \end{array} \right) \tag{4.11}$$

After some (rather tedious) algebraic manipulation we get:

$$\left(\begin{array}{c} \dot{\varrho}_i \\ \dot{\varphi}_i \end{array} \right) = \left(\begin{array}{c} 2\lambda\varrho_i \sin(\alpha_i - \gamma_i)\sin\gamma_i \\ 2\lambda\cos(\alpha_i - \gamma_i)\sin\gamma_i \end{array} \right) \tag{4.12}$$

Therefore we obtain:

$$\frac{\mathrm{d}\varrho_i}{\mathrm{d}\varphi_i} = \varrho_i \tan(\alpha_i - \gamma_i) = 2d\frac{\sin\gamma_i}{\gamma_i}\frac{\varrho_i}{\|\mathbf{r}_i\|} \tag{4.13}$$

By noticing that, for symmetry, γ_i keeps constant and equal to π/n and that we can write $\|\mathbf{r}_i\| = 2\varrho_i \sin\gamma_i$, we can conclude that $\lambda = 2 \cdot n \cdot d$ □

Clearly, in practical application, it would be better to choose d smaller than the actual footprint radius (d is indeed a control parameter).

Since with the proposed control law we are able to provide Archimede's spiral like trajectories, **in the *nominal* case complete coverage of target environment is guaranteed**.

4.4 Formation control for non-holonomic robots

It is possible to extend the previous formation control law to non-holonomic robots with a single *Pfaffian* constraint.

Let us model each vehicle as a Hilare-type mobile robot with nonlinear state model:

$$\begin{pmatrix} \dot{x}_i \\ \dot{y}_i \\ \dot{\theta}_i \\ \dot{v}_i \\ \dot{\omega}_i \end{pmatrix} = \begin{pmatrix} v_i \cos\theta_i \\ v_i \sin\theta_i \\ \omega_i \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \eta_i^1 \\ \eta_i^2 \end{pmatrix} \tag{4.14}$$

where $\mathbf{u}_i = (\eta_i^1 \eta_i^2)^T$ are the control inputs. In Eq. 4.14 it is evident the *Pfaffian* constraint:

$$\dot{x}_i \sin\theta_i - \dot{y}_i \cos\theta_i = 0 \tag{4.15}$$

The design of a decentralized feedback control policy for the system (4.14) requires nonholonomic control laws due to the *Pfaffian* constraint 4.15. However, if we only require that a point off the wheel axis of robots be maintained in formation, than the system can be feedback linearized. Feedback linearization about off wheel axis point was used in [34] and [22].
Specifically, let us define, as in [22], the "hand" position of a robot to be the point $\mathbf{h} = (h_x, h_y)$ that lies a distance L along the line that is normal to the wheel axis and intersects the wheel axis at the center point $\mathbf{r} = (x, y)$. The kinematics of the hand position is holonomic for $L \neq 0$.
Since we can reasonably consider sensor located at the hand, this approach does not represent a significant limitation.
The hand position is given by the equation:

$$\mathbf{h}_i = \mathbf{p}_i + L \begin{pmatrix} \cos\theta_i \\ \sin\theta_i \end{pmatrix} \tag{4.16}$$

To feedback linearize system (4.14), let us differentiate to get:

$$\ddot{\mathbf{h}}_i = \begin{pmatrix} -v_i\omega_i \sin\theta_i - L\omega_i^2 \cos\theta_i \\ v_i\omega_i \cos\theta_i - L\omega_i^2 \sin\theta_i \end{pmatrix} + \begin{pmatrix} \cos\theta_i & -L\sin\theta_i \\ \sin\theta_i & L\cos\theta_i \end{pmatrix} \begin{pmatrix} \eta_i^1 \\ \eta_i^2 \end{pmatrix} \tag{4.17}$$

Since:

$$\det \begin{pmatrix} \cos\theta_i & -L\sin\theta_i \\ \sin\theta_i & L\cos\theta_i \end{pmatrix} = L \neq 0 \tag{4.18}$$

the system (4.14) with output (4.16) has constant relative degree equal to two and can, therefore, be output feedback linearized [22].

Let us define the map g as in [29]:

$$\xi_i = g(\mathbf{x}_i) \triangleq \begin{pmatrix} x_i + L\cos\theta_i \\ y_i + L\sin\theta_i \\ v_i\cos\theta_i - L\omega_i\sin\theta_i \\ v_i\sin\theta_i + L\omega_i cos\theta_i \\ \theta_i \end{pmatrix} \tag{4.19}$$

The map g is a diffeomorphism [29]. In the transformed coordinates we get:

$$\begin{aligned} \begin{pmatrix} \dot{\xi}_{1i} \\ \dot{\xi}_{2i} \end{pmatrix} &= \begin{pmatrix} \xi_{3i} \\ \xi_{4i} \end{pmatrix} \\ \begin{pmatrix} \dot{\xi}_{3i} \\ \dot{\xi}_{4i} \end{pmatrix} &= \begin{pmatrix} -v_i\omega_i\sin\theta_i - L\omega_i^2\cos\theta_i \\ v_i\omega_i\cos\theta_i - L\omega_i^2\sin\theta_i \end{pmatrix} + \begin{pmatrix} \cos\theta_i & -L\sin\theta_i \\ \sin\theta_i & L\cos\theta_i \end{pmatrix}\mathbf{u}_i \\ \dot{\xi}_{5i} &= -\frac{\xi_{3i}}{L}\sin\xi_{5i} + \frac{\xi_{4i}}{L}\cos\xi_{5i} \end{aligned} \tag{4.20}$$

Therefore a natural choice for the output feedback linearizing control is given by:

$$\mathbf{u}_i = \begin{pmatrix} \cos\theta_i & -L\sin\theta_i \\ \sin\theta_i & L\cos\theta_i \end{pmatrix}^{-1} \left[\nu_i - \begin{pmatrix} -v_i\omega_i\sin\theta_i - L\omega_i^2\cos\theta_i \\ v_i\omega_i\cos\theta_i - L\omega_i^2\sin\theta_i \end{pmatrix} \right] \tag{4.21}$$

which gives:

$$\begin{aligned} \begin{pmatrix} \dot{\xi}_{1i} \\ \dot{\xi}_{2i} \end{pmatrix} &= \begin{pmatrix} \xi_{3i} \\ \xi_{4i} \end{pmatrix} \\ \begin{pmatrix} \dot{\xi}_{3i} \\ \dot{\xi}_{4i} \end{pmatrix} &= \nu_i \\ \dot{\xi}_{5i} &= -\frac{\xi_{3i}}{L}\sin\xi_{5i} + \frac{\xi_{4i}}{L}\cos\xi_{5i} \end{aligned} \tag{4.22}$$

where:

$$\mathbf{h}_i = \begin{pmatrix} \xi_{1i} \\ \xi_{2i} \end{pmatrix} \tag{4.23}$$

The last equation represents the internal dynamics rendered unobservable by the coordinate transformation $g(\cdot)$. Setting $\xi_{1i} = \xi_{2i} = \xi_{3i} = \xi_{4i} = 0$ we find that the internal dynamics is stable but not asymptotically stable, as observed in [22].
The input output dynamics of each robot will be represented by the double integrator system:

$$\ddot{\mathbf{h}}_i = \nu_i \tag{4.24}$$

. We also set:

$$\begin{pmatrix} \xi_{1i} \\ \xi_{2i} \end{pmatrix} = \xi_i^a \tag{4.25}$$

and

$$\begin{pmatrix} \xi_{3i} \\ \xi_{4i} \end{pmatrix} = \xi_i^b \tag{4.26}$$

The proposed formation control law can, therefore, be simply extended to non-holonomic robots by setting 4.7 as a reference speed for the system 4.24 through a speed tracking control law as follows:

$$\begin{aligned} \dot{\xi}_i^a &= \xi_i^b \\ \dot{\xi}_i^b &= k(\xi_i^b - \mu_i) \end{aligned} \tag{4.27}$$

where μ_i is given in 4.7, remembering that now $\mathbf{h}_i = \xi_i^a$

Chapter 5

Simulation results

Simulations confirm that the proposed control law guarantees Archimede's spiral path with step $2 \cdot n \cdot d$ and hence complete coverage of a circular environment with surmountable obstacles. In the next sections we present simulation results for both holonomic robots and non-holonomic robots.

5.1 Simulation parameters and performance criteria

In all simulations we assume a footprint radius $d = 0.2$ and a gain $\lambda = 1$; moreover, we consider a gain $k = 800$ in the speed tracking control law. In trajectory figures, one trajectory is dotted for image clarity.

Agents are deployed on a regular configuration with radius $\rho = 1$ (in practical application, in order to avoid the initial central hole it is enough to deploy agents "sufficient" close each other). To add perturbation, we simply add noise to the initial position and orientation of each robot with magnitude σ_{pos} and σ_{or} respectively.

To assess robustness against perturbation and agent loss, we define two error functions. Let us define the Phase Error as:

$$E_\gamma = \sum_{i=1}^{n} (\gamma_i - \pi/n)^2 \tag{5.1}$$

and the Distance Error as:

$$E_r = \sum_{i=1}^{n} \left(\|\mathbf{r}_i\|^2 - \|\mathbf{r}_{i-1}\|^2 \right)^2 \tag{5.2}$$

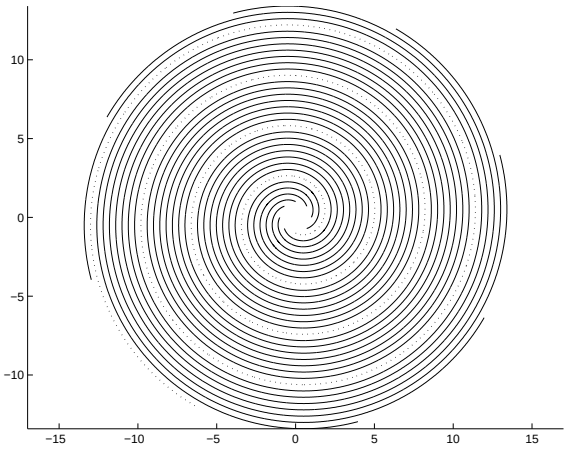


Figure 5.1: $N = 8$ holonomic robots starting from a regular configuration.

Phase error and Distance error "measure" the distance from a regular configuration. Clearly, if $E_\gamma = E_r = 0$, agents are on a regular configuration and therefore perform, as proved, the desired Archimede's spiral trajectories. Thus, the two error functions give us a way to evaluate the robustness of the system: a fast convergence of both functions to zero means robustness against perturbations.

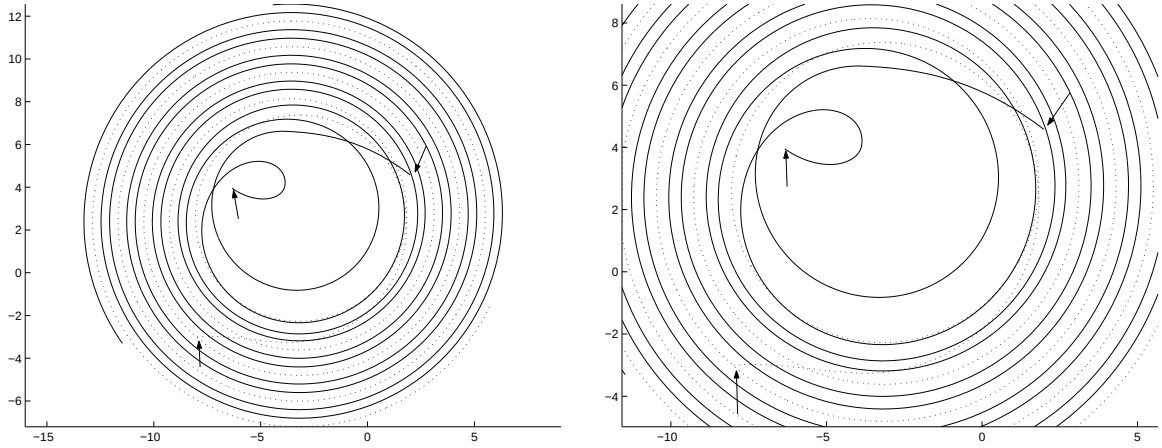
5.2 Formation of holonomic robots

Results shown in Fig. 5.1 refers to a scenario with $n = 8$ robots starting from a regular configuration. Resultant trajectories are Archimede's spirals as desired; the spiral step sequence corresponding to the dotted trajectory is $S = [3.200, 3.199, 3.200]$, in agreement with the desired value $s = 2 \cdot n \cdot d = 3.2$.

As far as robustness is concerned, simulation results show that, thanks to the proposed control law, all robots, starting from a perturbed configuration, gather on a regular configuration and hence they start performing Archimede's spiral paths; equivalently, error functions converge quickly to zero. This behavior (gathering on a regular configuration) was previously observed for a conceptually analogous control law in [24].

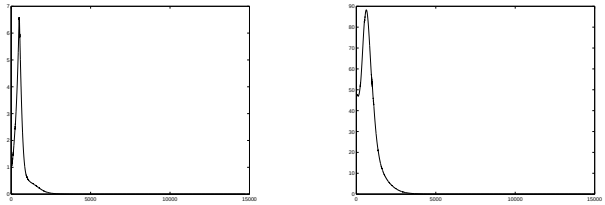
Several simulations (more than 100) have been carried out to assess robustness against perturbation and agent losses.

In the first reported experiment, we added to a regular configuration with $n = 3$ holonomic agents



(a) Spiral trajectories (b) Spiral trajectories - detail

Figure 5.2: $N = 3$ holonomic vehicles starting from a perturbed configuration.

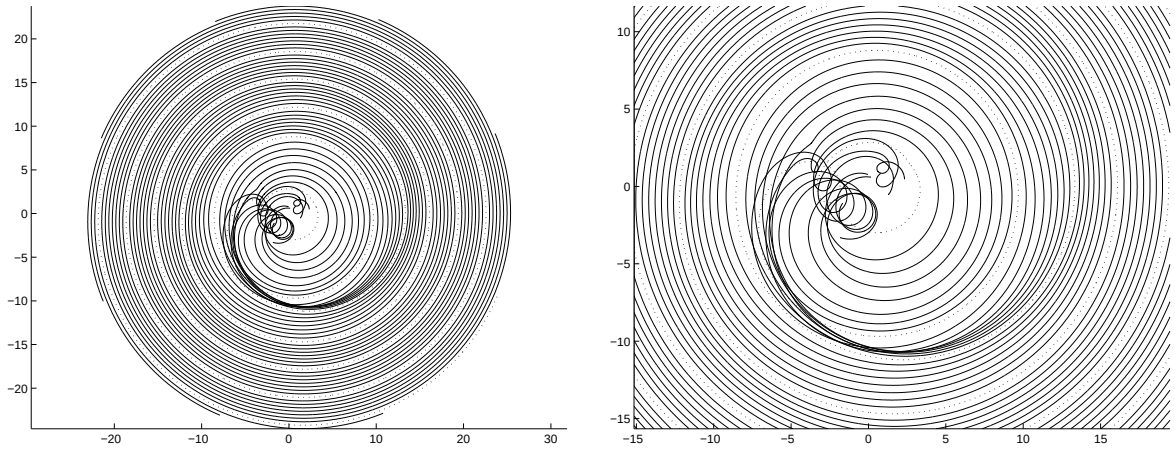


(a) Phase Error (b) Distance Error

Figure 5.3: Error function for the perturbed holonomic system with $N = 3$.

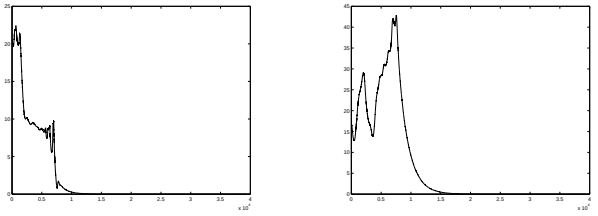
a position perturbation with magnitude $\sigma_{pos} = 5$ and a heading perturbation with magnitude $\sigma_{pos} = 1.3$ rad. Fig. 5.2 shows resulting trajectories and a zoom in the first part of the simulation; arrows indicate initial agent positions. Fig. 5.3 shows that the two error functions go quickly to zero, thus indicating a fast convergence to a regular configuration and hence to the desired Archimede's spiral-like trajectories with step $s = 1.2$.

We then repeat the experiment with $n = 8$ holonomic robots, adding to the regular initial configuration a position perturbation with magnitude $\sigma_{pos} = 1.5$ and a heading perturbation with magnitude $\sigma_{pos} = 0.6$ rad. Again, after a fast transient, robots start performing spiral-like trajectories with the desired step $s = 3.2$. Simulation results are shown in Figg. 5.4 and 5.5.



(a) Spiral trajectories (b) Spiral trajectories - detail

Figure 5.4: $N = 8$ holonomic vehicles starting from a perturbed configuration.



(a) Phase Error (b) Distance Error

Figure 5.5: Error function for the perturbed holonomic system with $N = 8$.

In Fig. 5.6 we report a simulation in which an agent is lost; in detail, we simulate that $n = 7$ holonomic robots start from a $n = 8$ regular configuration with an agent missing (*i.e.* from a regular configuration for $n = 8$ with a hole). As shown, robots reconfigure autonomously their trajectories, just relying on locally available information, to provide Archimede's spiral paths with the desired step $s = 2.8$. The arrow in Fig. 5.6 indicate the missing agent.

5.3 Formation of non-holonomic robots

Results shown in Fig. 5.7 refers to a scenario with $n = 8$ non-holonomic robots starting from a regular configuration. Resultant trajectories are Archimede's spirals as desired; the spiral step sequence corresponding to the dotted trajectory is $S = [3.201, 3.204, 3.211]$, in agreement with the

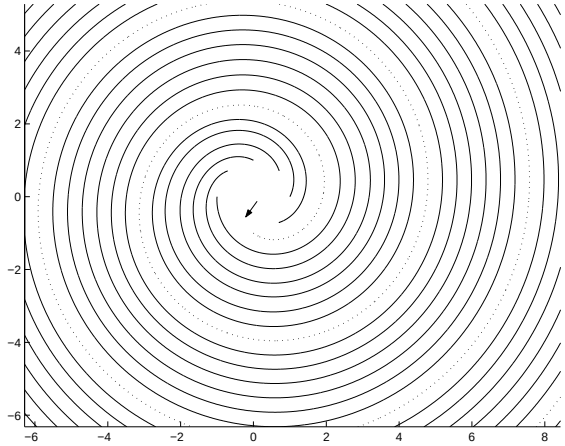


Figure 5.6: Trajectories after one holonomic agent loss.

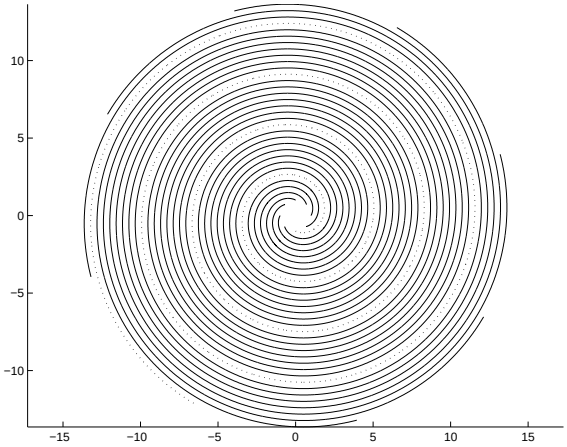


Figure 5.7: $N = 8$ non-holonomic robots starting from a regular configuration.

desired value $s = 2 \cdot n \cdot d = 3.2$. Fig. 5.7 shows the tracking error for the reference speed.

Again, several simulations have been carried out to assess robustness against perturbation and agent losses.

In the reported experiment, we added to a regular configuration with $n = 8$ non-holonomic agents a position perturbation with magnitude $\sigma_{pos} = 1.5$ and a heading perturbation with magnitude $\sigma_{pos} = 0.6$ rad. Fig. 5.9 shows resulting trajectories and a zoom in the first part of the simulation. Fig. 5.10 shows that the two error functions go quickly to zero, thus indicating a fast convergence to a regular configuration and hence to the desired Archimede's spiral-like trajectories with step $s = 3.2$.

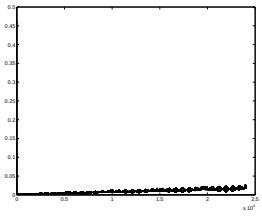
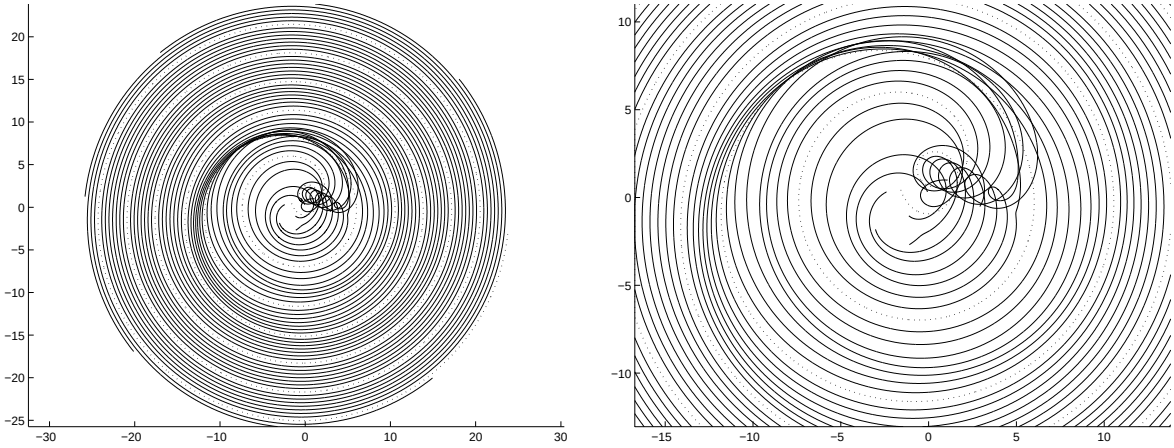


Figure 5.8: Tracking error for the reference speed.

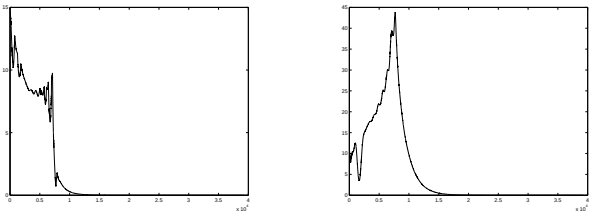


(a) Spiral trajectories

(b) Spiral trajectories - detail

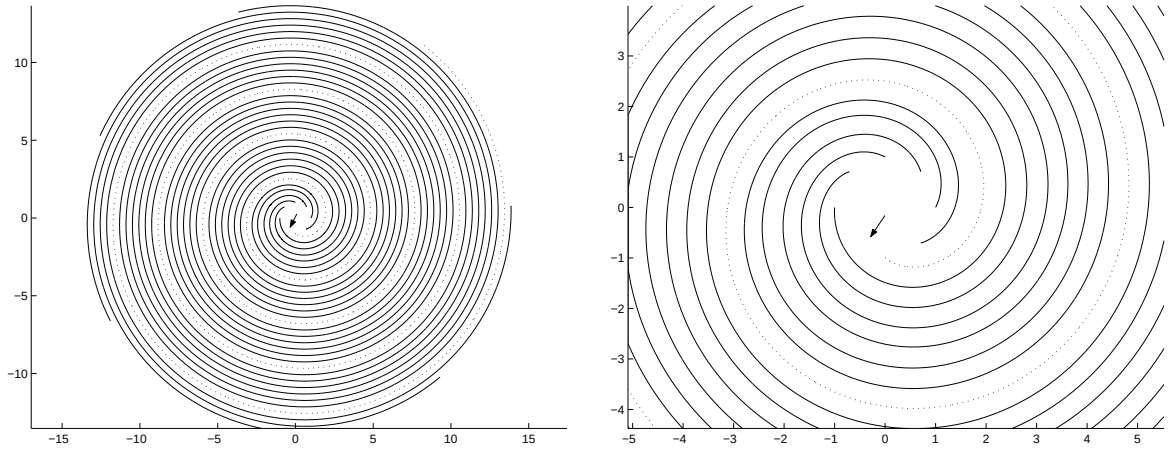
Figure 5.9: $N = 8$ non-holonomic vehicles starting from a perturbed configuration.

In Fig. 5.11 we report a simulation in which an agent is lost; in detail, we simulate that $n = 7$ non-holonomic robots start from a $n = 8$ regular configuration with an agent missing (*i.e.* from a regular configuration for $n = 8$ with a hole). As shown, robots reconfigure autonomously their trajectories, just relying on locally available information, to provide Archimede's spiral paths with the desired step $s = 2.8$. The arrow in Fig. 5.11 indicate the missing agent.



(a) Phase Error (b) Distance Error

Figure 5.10: Error function for the perturbed non-holonomic system with $N = 8$.



(a) Spiral trajectories (b) Spiral trajectories - detail

Figure 5.11: Trajectories after one non-holonomic agent loss.

Conclusion

In this thesis we studied from two different but complementary perspectives the problem of robot autonomy.

On one hand we proposed a novel structure and control system architecture for an hexapod robot. As far as the structure is concerned, each leg has a unique design and a peculiar articulation with the body; moreover an innovative linear/rotational actuation is introduced. Dynamical simulations proved that this design provides superior agility. The control system architecture is based on a behavior-approach: each behavior is modelled as a Motor Map with an adaptive reward. Reward learning is achieved through a reinforcement learning. To the best of our knowledge, it is the first time that an adaptive reward for a Motor Map is introduced. Dynamical simulations showed the suitability of the approach.
On the other hand, we dealt with a classical coordination problem for autonomous robots: distributed coverage path planning. We firstly studied the case when all robot are holonomic. We presented an innovative distributed algorithm, based on a cyclic pursuit approach, whose correctness has been analytically proved. Then we successfully applied the developed algorithm, through input-output feedback linearization, to the case when all robots have one non-holonomic constraint. Simulation results confirm the effectiveness of the proposed algorithm.

The first part of the thesis has been developed at the Electrical, Electronic and Systems Engineering Department of the University of Catania under the supervision of prof. P. Arena, while the second part has been developed at the Department of Mechanical and Aerospace Engineering of the University of California at Los Angeles, USA, under the supervision of prof. E. Frazzoli.

Acknowledgements

First and foremost, I would like to thank my two advisers, Professor Paolo Arena at the University of Catania and Professor Emilio Frazzoli at the University of California at Los Angeles. Their ideas and points of view have been a constant source of inspiration, and our discussions have taught me to be more inquisitive. Thanks for widening my views and helping me achieving my goals.

Thanks to Mattia Frasca and Luca Patané for their help: their suggestions have been essential for my work.

Thanks to all the staff of the Scuola Superiore di Catania, and in particular to the Professor Emanuele Rimini, for their encouragement and support throughout all my university studies.

Thanks to my parents Piero and Lella. To them and their smile I owe everything I attained.

Finally, thanks to Manuela. It has been hard to spend five months far from each other, but the distance strengthened our connection even more.

Bibliography

[1] R. Altendorfer, N. Moore, H. Komsuo, M. Buehler, H.B. Brown Jr., D. Mcmordie, U. Saranli, R. Full And D.E. Koditschek. *RHex: A Biologically Inspired Hexapod Runner.* Autonomous Robots, 11, pp. 207–213, 2001.

[2] P. Arena, L. Fortuna, M. Frasca and G. Sicurella. *An Adaptive, Self-Organizing Dynamical System for Hierarchical Control of Bio-Inspired Locomotion.* IEEE Transactions On Systems, Man, And Cybernetics Part B: Cybernetics, 34, No. 4, pp. 1823–1837, 2004.

[3] P. Arena, L. Fortuna, M. Frasca, L. Patané, G. Vagliasindi. *CPG-MTA implementation for locomotion control.* Proc. of IEEE International Symposium on Circuits and Systems, Kobe, Japan, May 23-26, 2005.

[4] P. Arena, L. Fortuna, M. Frasca, L. Patané, M. Pavone. *Climbing Obstacles via Bio-Inspired CNN-CPG and Adaptive Attitude Control.* Proc. of IEEE International Symposium on Circuits and Systems, Kobe, Japan, May 23-26, 2005.

[5] P. Arena, L. Fortuna, M. Frasca, L. Patané, M. Pavone. *Climbing Obstacle in Bio-robots via CNN and Adaptive Attitude Control.* International Journal of Circuit Theory and Applications, under review.

[6] P. Arena, S. Castorina, L. Fortuna, M. Frasca, and M. Ruta *A CNN-based chip for robot locomotion control.* Proc. ISCAS03, 2003.

[7] J. Ayers, J. Davis, and A. Rudolph. *Neurotechnology for Biomimetic Robots.* MIT Press, 2002.

[8] R. D. Beer, H. J. Chiel, R. D. Quinn, K. S. Espenschied, and P. Larsson. *A distributed neural network architecture for hexapod robot locomotion.*Neural Computation,4, pp. 356–365, 1992.

[9] R.D. Beer, R.D. Quinn, H.J. Chiel and R.E. Ritzmann. *Biologically inspired approaches to robotics.* Communications of the ACM, 40, No. 3, 1997.

[10] H. Choset and P. Pignon. *Coverage path planning: The boustrophedon decomposition.* Proc. of the International Conference on Field and Service Robotics, Canberra, Australia, December 1997.

[11] H. Choset. *Coverage for robotics A survey of recent results.* Annals of Mathematics and Artificial Intelligence 31, pp. 113–126, 2001, Kluwer Academic Publishers.

[12] L. O. Chua and L. Yang. *Cellular neural networks: Theory.* IEEE Trans. Circuits Syst. I, 35, pp. 1257–1272, 1988.

[13] J. E. Clark, J. G. Cham, S. A. Bailey, E. M. Froehlich, P. K. Nahata, R. J. Full, M. R. Cutkosky. *Biomimetic Design and Fabrication of a Hexapedal Running Robot.* Proc. of IEEE International Conference on Robotics and Automation, 2001.

[14] F. Delcomyn and M. E. Nelson. *Architectures for a biomimetic hexapod robot.* Robotics and Autonomous Systems, 30, pp. 5–15, 2000.

[15] K. S. Espenschied, R. D. Quinn, R. D. Beer and H. J. Chiel. *Biologically based distributed control and local reflexes improve rough terrain locomotion in a hexapod robot.* Robotics and Autonomous Systems, 18, pp. 59–64, 1996.

[16] M. Frasca, P. Arena, L. Fortuna. *Bio-Inspired Emergent Control Of Locomotion Systems.* World Scientific Series on Nonlinear Science, Series A - Vol. 48, 2004.

[17] E. Gonzalez, M. Alarcon, P. Aristizabal, C. Parra. *BSA: A coverage Algorithm.* Proc. of the 2003 IEEE/RSJ Intl. Conference on Intelligent Robots and Systems, Las Vegas, Nevada, October 2003.

[18] C. Hofner and G. Schmidt. *Path planning and guidance techniques for an autonomous mobile cleaning robot.* Robot. Auton. Syst., 14, pp. 199–212, 1995.

[19] D. Kingsley, R. Quinn, and R. Ritzmann. *A cockroach inspired robot with artificial muscles.* Proc. of International Symposium on Adaptive Motion of Animals and Machines (AMAM), Kyoto, Japan, 2003.

[20] D. Kurabayashi, J. Ota, T. Arai and E. Yoshida. *Cooperative sweeping by multiple mobile robots.* Int. Conf. on Robotics and Automation, 1996.

[21] J.C. Latombe. *Robot Motion Planning.* Kluwer Academic, Boston, MA, 1991.

[22] J. R. T. Lawton, R. W. Beard and B. J. Young *A Decentralized Approach to Formation Maneuvers.* IEEE Transactions On Automatic Control, Vol. 19, No. 6, pp. 933–941 December 2003.

[23] D. MacKenzie and T. Balch. *Making a clean sweep: Behavior based vacuuming.* AAAI Fall Symposium, Instationating Real-World Agents, 1996.

[24] J. A. Marshall, M. E. Broucke and B. A. Francis. *A Pursuit Strategy for Wheeled-vehicle Formations.* Proc. of the 42nd IEEE Conference on Decision and Control, Maui, Hawaii, USA, December 2003.

[25] J. A. Marshall, M. E. Broucke, B. A. Francis *Formations of Vehicles in Cyclic Pursuit.* IEEE Transactions On Automatic Control, 49, No. 11, pp. 1963–1974, November 2004.

[26] M.J. Mataric *Learning in behavior-based multi-robot systems: policies, models, and other agents.* Journal of Cognitive System Research, 2(1), pp. 81–93, 2001.

[27] S. McMillan, D. E. Orin and R. B. McGhee. *A computational framework of underwater robotic vehicle systems.* J. Auton. Robots - Special Issue On Autonomous Underwater Robots, 3, pp. 253–268, 1996.

[28] F. Michaud and M.J. Mataric. *Learning from History for Behavior-Based Mobile Robots in Non-Stationary Conditions.* Autonomous Robots, 5, pp. 335–354, 1998.

[29] B. A. Novel, G. Bastin and G. Campion. *Modelling and Control of Non Holonomic Wheeled Mobile Robots* Proc. of the 1991 IEEE International Conference on Robotics and Automation, Sacramento, California, April, 1991.

[30] O. M. Omidvar and D. L. Elliott. *Neural Systems for Control.* Academic Press, 1997, available at http://www.isr.umd.edu/ delliott/NeuralSystemsForControl.pdf.

[31] M. Pavone and E. Frazzoli. *A Cyclic-Pursuit Approach To Multi-Agent Coverage Path Planning.* Proc. of ASME 2005 Design Engineering Technical Conference, November 6-11, 2005, Orlando, Florida, USA, abstract accepted, under review.

[32] M. Pavone, P. Arena, M. Frasca and L. Patané. *An innovative mechanical and control architecture for a biomimetic hexapod for planetary exploration.* in preparation.

[33] M. Pavone. *Modelli matematici per soluzioni biomimetiche nell'ingegneria aerospaziale.* M.Sc. Thesis, 2004.

[34] J. B. Pomet, B. Thuilot, G. Bastin, G. Campion. *A hybrid strategy for the feedback stablization of nonholonomic mobile robots.* Proceedings of the 1992 IEEE Int. Conf. Robotics and Automation, pp. 129–134, May 1992.

[35] R. D. Quinn and R. E. Ritzmann. *Construction of a hexapod robot with cockroach kinematics benefits both robotics and biology.* Connect. Sci., 10, No. 3-4, pp. 239–254, 1998.

[36] R. D. Quinn, G. M. Nelson, and R. J. Bachmann. *Toward mission capable legged robots through biological inspiration.* Autonomous Robots, 11, pp. 215–220, 2001.

[37] H. Ritter, T. Martinetz and K. Schulten. *Neural Computing and Self-Organizing Maps Reading.* MA: Addison-Wesley, 1992.

[38] G. M. Shepherd *Neurobiology.* Oxford Univ. Press, 1997.

[39] L. H. Ting, R. Blickhan And R. J. Full. *Dynamic And Static Stability In Hexapedal Runners.* J. exp. Biol., 197, pp. 251–269, 1994.

[40] I. A. Wagner, M. Lindenbaum and A. M. Bruckstein. *Distributed Covering by Ant-Robots Using Evaporating Traces.* IEEE Transactions On Robotics And Automation, Vol. 15, No. 5, October 1999.

[41] J. T. Watson, R. E. Ritzmann, S. N. Zill and S.J. Pollack. *Control of obstacle climbing in the cockroach,* Blaberus disoidalis*, I. Kinematics.* J. Comp. Physiol. A., 188, pp. 39–53, 2002.

[42] J. T. Watson, R. E. Ritzmann and S.J. Pollack. *Control of climbing behavior in the cockroach,* Blaberus discoidalis*. II. Motor activities associated with joint movement.* J. Comp. Physiol. A., 188, pp. 55–69, 2002.

[43] H. Yaguchi. *Robot introduction to cleaning work in the East Japan Railway Co.* Adv. Robot., Vol. 10, No. 4, pp. 403–414, 1996.

[44] www.ai.mit.edu/projects/boadicea/boadicea.html