ECOLE
POLYTECHNIQUE
DE BRUXELLES

ULB

UNIVERSITÉ LIBRE DE BRUXELLES

Human - Robots Swarms Interaction

An Escorting Robot Swarm that Diverts a Human away from Dangers one cannot perceive.

Mémoire présenté en vue de l'obtention du diplôme
d'Ingénieur Civil en Informatique à finalité Intelligence Computationnelle

Anthony Debruyn

Directeur
Professeur Mauro Birattari

Co-Promoteur
Professeur Marco Dorigo

Superviseur
Gaëtan Podevijn, Andreagiovanni Reina

Service
IRIDIA

Année académique
2014 - 2015

Acknowledgements

Mauro Birattari

Gaëtan Podevijn

Andreagiovanni Reina

Anthony Antoun

Brian Delhaisse

Lorenzo Garattoni

Family & Friends

Résumé

Summary

Contents

List of Figures

List of Tables

Todo list

Chapter 1

Introduction

[I'll do this and this... blah blah blah...]

As swarm robotic systems are mostly destined to operate on risky floors, unknown environment, it would seem logical to consider their application in exploration and/or protection missions. However, at the time of writing this thesis, we could not find any study on the subject. Exploration experiments never included a human, or other living organism. The object of this thesis is to address this lack of study by designing and implementing a protective behaviour executed by a robotic swarm.

The human operator is here part of the swarm system. The swarm has to protect him by preventing him from going into dangerous areas, in the same way a group of bodyguards protects someone. The swarm has to follow the operator anywhere to ensure permanent protection.

We believe this work to be important since it could lay the foundations of a new branch in swarm engineering: human protection, escort or swarm turn-by-turn navigation.

An article will be written to expose this research to the rest of the swarm robotics community.

Chapter 2

State of the Art

In this section, we will discuss the problem that led to the creation of this thesis by first providing the reader with some general insight in the world of swarm robotics and swarm intelligence. Then we will focus on specific parts of these domains of study: feedbacks between human and single robot, and human and robots' swarm.

2.1 Human - Robot Interaction

[Work related to what I do (detect humans, protection, follow person). Conclude on why it cannot be applied to my problem.]

2.2 Swarm Robotics

[Flocking with a guide, and then Pattern Formation with a Guide. Work related to what I do. Conclude on why it cannot be applied to my problem. Make connections with this part later in the document. S'inspirer de Brambilla pour les flock et pattern.]

This section and the next one are largely inspired by Brambilla et al. (2013), a reviewing article on swarm engineering. For Şahin (2005), swarm robotics is defined as *'the study of how large numbers of relatively simple physically embodied agents can be designed such that a desired collective behavior emerges from the local interactions among agents and between the agents and the environment'* (Şahin, 2005). Swarm robotics can be separated from other robotic studies by the following characteristics (Brambilla et al., 2013):

- Robots are *autonomous*

- Robots evolve *in the environment* and can interact with it
- Robots' interactions are *local* (sensors and communications)
- No *centralised control* or *global knowledge*
- Robots *cooperate* to achieve a certain goal

As in this field of study, one is always looking for *robust*, *scalable* and *flexible* systems, the main source of inspiration is the group of social animals: ants, birds, fishes, ... When some of these simple animals gather in groups, they are able to perform tasks that could not be achieved individually (collective behaviour emerges from local interactions). Below are listed the definitions of these three terms (Brambilla et al., 2013):

Robustness: Resistance against *loss of group entities*. One can increase it by adding redundancy or remove the need for a leader.

Scalability: Low variation in the performance of a system with respect to the *size of the system*. It can be increased by encouraging local interactions, such as sensing and communications.

Flexibility: Low variation in the performance of a system with respect to the *type of environment or the task*.

With these definitions in mind, we can explain swarm engineering as:

> *'Swarm engineering is an emerging discipline that aims at defining systematic and well founded procedures for modeling, designing, realizing, verifying, validating, operating, and maintaining a swarm robotics system.'*
> *- Brambilla et al. (2013)*

Kazadi (2000) points out that *'to the swarm engineer, the important points in the design of a swarm are that the swarm will do precisely what it is designed to do, and that it will do so reliably and on time'* (Kazadi, 2000).

2.2.1 Human - Robots Swarm Interaction

[Work related to what I do (detect humans, protection, follow person). Conclude on why it cannot be applied to my problem.]

Human - Robotic swarm interaction is the study of how humans can interact with a swarm to control it and receive feedback from it (Brambilla et al., 2013). A proper feedback is needed by the operator in order to make the right decisions.

Since swarms must ideally be autonomous and make decisions in a distributed way, it is difficult to insert a communication with a human operator in the system to gain control.

Currently, little attention has been devoted to the study of the interaction between humans and robotic swarms, how one can send instructions and receive feedback. People investigating in the field encounter many difficulties, such as the difference of perspective between the swarm and the human operator (the human only observes the global collective behaviour, not the local interactions or individual behaviours driving the robots), the simplicity of the hardware found on the robots, or the efficient synthesis of all the information sent by the robots. All the existing types of interactions in the literature present a major disadvantage: they require an extra layer between the group of robots and the human. This requirement might not always be satisfied when we remember that swarms like this are mostly destined to evolve in an unknown environment. The monitoring equipment necessary to operate the swarm may not be safely deployed. Furthermore, a synthesis of all the local information pieces must be done in order to provide an understandable state of the system to the human. A supplementary step that involves modelling, additional overheads and perhaps heavy computations, and the gathering of all information at a central point (eliminating by the way the distributed and not centralised properties of the swarm system) (Podevijn et al., 2012).

Daily et al. (2003) used a head-mounted display and augmented reality to add information right on top of the robot in the environment itself, suppressing the need for an additional display. Baizid et al. (2009) proposed a platform to interact with multiple robots simultaneously through a graphical user interface, or a head-mounted display, virtual reality etc. They also studied how virtual reality abstraction affected the human perception and cognitive capabilities, i.e, they created a virtual environment by filtering useless information. McLurkin et al. (2006) developed an centralised graphical user interface taking inspiration from real-time strategy video games, where one must control armies. They also imagined a feedback approach based on LEDs and sounds. The robots transmit their internal state by applying to their LEDs and sound system a defined pattern, recognisable by the operator, now able to quickly understand the state of the swarm without looking at a supplementary interface.

Podevijn et al. (2012) argue that self-organised mechanism, as those ruling the behaviour of the swarm, should be used to provide feedback to the operator. They suggest that the best entity which could communicate the status of the system and the whole swarm is the swarm itself. They performed experiments using colour feedback to distinguish different internal states and split the swarms into groups to tackle different tasks.

As swarm robotic systems are mostly destined to operate on risky floors, unknown environment, it would seem logical to consider their application in exploration and/or protection missions. However, at the time of writing this thesis, we could not find any study on the subject. Exploration experiments never included a human, or other living organism. The object of this thesis is to address this lack of study by designing and implementing a protective behaviour executed by a robotic swarm.

The human operator is here part of the swarm system. The swarm has to protect him by preventing him from going into dangerous areas, in the same way a group of bodyguards protects someone. The swarm has to follow the operator anywhere to ensure permanent protection.

We believe this work to be important since it could lay the foundations of a new branch in swarm engineering: human protection, escort or swarm turn-by-turn navigation.

Chapter 3

An Escorting Swarm

In this chapter, we will explore the investigated problem and the proposed solution at a high level of description. In the next chapter, we will describe the implementation details.

3.1 The Problem

Since the early days, human beings have explored new territories to expand their control and get a better understanding of the world surrounding them. Among those new landscapes, some were relatively safe but some were dangerous. To overcome this, we have invented equipment, suits, and other kinds of protections. We tried, with this work, to contribute to the study of these solutions.

Figure 3.1 shows a graphical representation of a possible scenario. The dangerous areas are the red circles. Those areas could be radioactive areas, mine fields, or any other invisible threats. The human must travel from point A to B without being hurt by the danger contained in these areas. The human cannot perceive them. The protection created should prevent the user from going inside those areas.

Exploration is not the only real application for the proposed solution that comes into mind. Rescue in disaster areas would also benefit from it (evacuation of people to safe zones, etc). The solution should be able to constantly protect the person using it, and constantly provide feedback. It should be robust and fit to the destination environment.

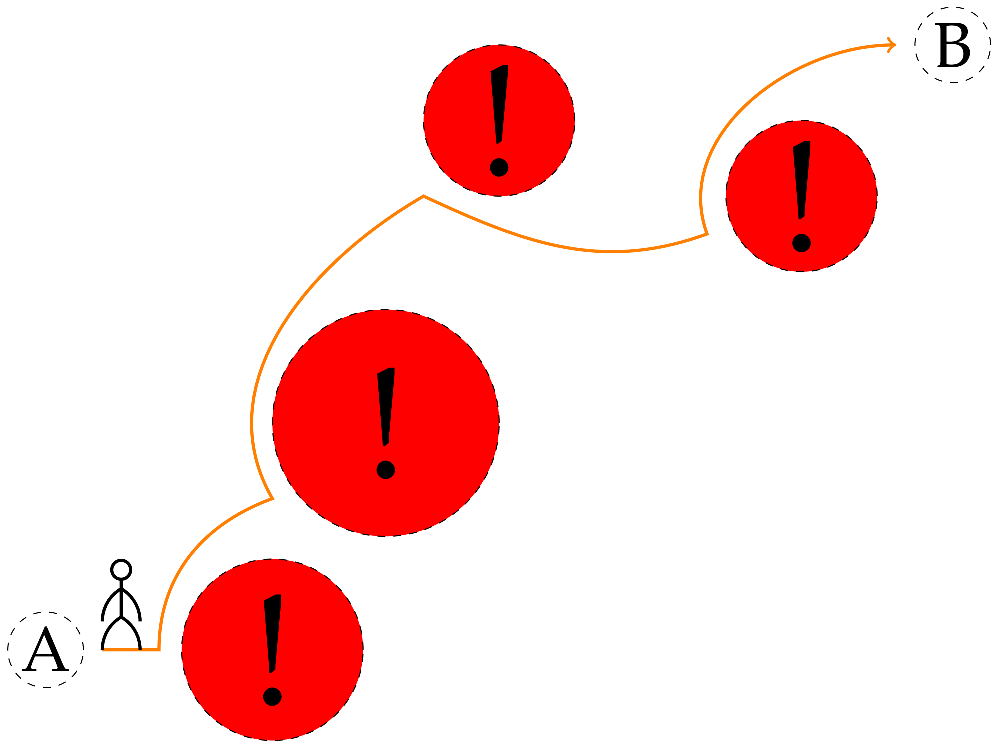


Figure 3.1 – Unknown Dangerous Environment: This image illustrates an environment, observed from above, in which a human must move from point *A* to point *B* while avoiding invisible dangerous areas. *A* is the start location, *B* is the goal ant the red circles represent dangerous zones. We provide in this thesis a solution to guarantee safeness in such circumstances. Possible applications for this type of solutions are: mine fields crossing and cleaning, radioactive areas avoidance,...

3.2 Solution

The solution we propose involves the use of a swarm of robots. Swarm robotics seems fit to this kind of application, since it is compatible with unknown environments thanks to its flexible, robust and scalable characteristics (Brambilla et al., 2013). In case of failure of one or a few robots, the system would continue to provide sufficient performance thanks to its scalability and robustness.

The swarm of robots forms a round shield around a user. The round shield formed by the swarm enables a 360° protection of the user. All the robots try to stay at the same distance from each other and the human (except when there is a danger). To achieve this, the final solution relies on the pattern formation theory widely used in swarm robotics. The corresponding techniques will be explained in the next chapter with more details. If the number of robots is not high enough to form a complete circle, an arc is formed at the front to always shield the most critical zone.

Parler du flocking et pattern.

As shown on figure 3.2, the robots in contact with a dangerous zone will report the danger through visual communication with the human. Here the robots light on their orange LEDs and stay on the boundary of the zone to prevent the human from getting into it. Since the human cannot see the danger, and only the robots can, we can see that the swarm is increasing the perception capabilities of the human.

One issue that had to be resolved was the detection of the human by the robots. As Podevijn et al. (2012) suggested, the interface between the human and the robots swarm should be restricted to the strict minimum because in the field the infrastructure needed to operate the swarm might not be easy to build and manipulate. The swarm should handle the communication on its own. Furthermore, any centralised control system would break the distributed and robust feature of swarm robotics. As a big infrastructure such as a tracking system, or any interface of the same kind would have been difficult to use in real life applications, we designed and implemented a compact, wearable device that allows a human to be recognised by the robots: a pair of shoes.

Figure 3.3 illustrates the use of the shoes (no user is wearing them to not occlude the field of view). In figure 3.3 (left), the robots have just recognised the shoes thanks to the LED system inside and begin to move in order to form a circle around the shoe. n figure 3.3 (right), we show an example of one configuration obtained after 3 minutes, viewed from above.

Our objective in this thesis, is to present an innovative protection using swarm robotics. The results obtained with real robots are presented in the chapter 5.

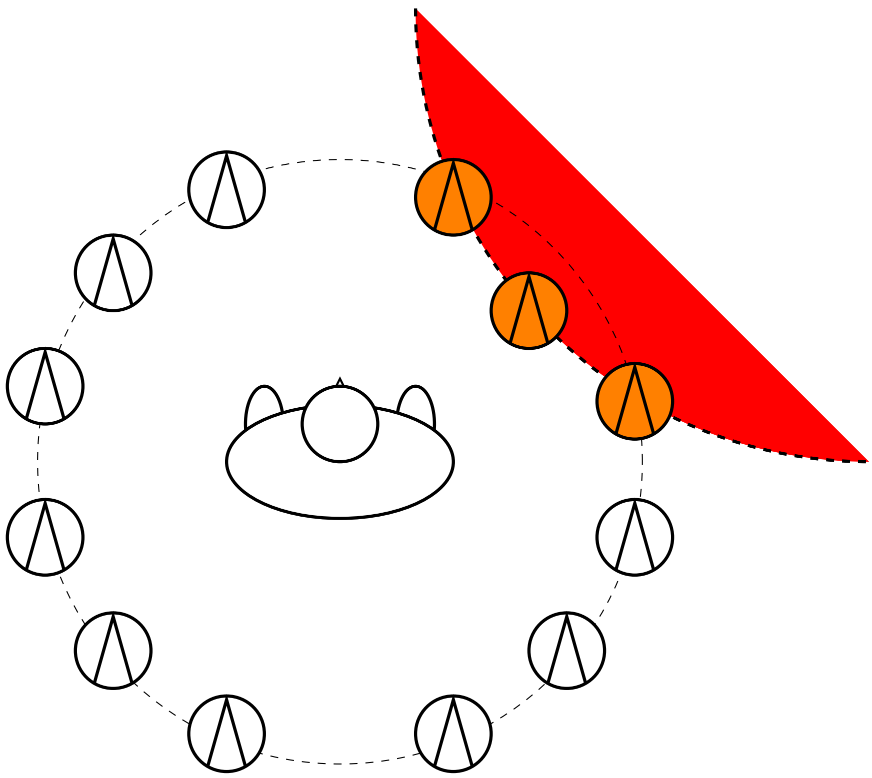


Figure 3.2 – Swarm Prevention: This figure is a symbolic representation of a human helped by a swarm. The circles with a triangle inside are representations of a robot. The swarm tells the human that a dangerous zone is located at the front right by visual communication (here the robots change their colour to red). The swarm stays at the boundary to form a 'shield'. The direction taken by each individual in the swarm is given by the triangle inside (here heading north).

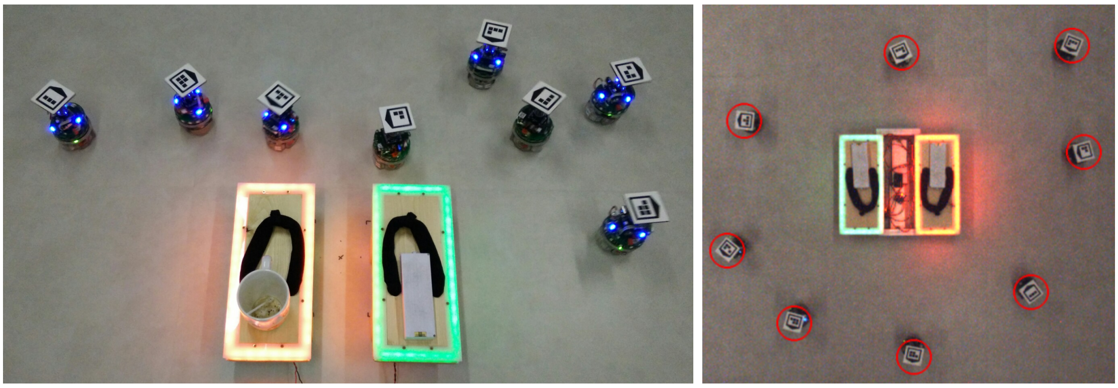


Figure 3.3 – The Shoes: This picture shows a prototype of the shoes viewed from above, and the robots interacting with it. The interaction is realised through the recognition of the colours, one for each shoe, indicating left or right side. This pair of shoes enables the robots to locate the user, allowing them to evolve at the target distance from him/her. On the left image, the robots are still in the process of placing themselves in a correct circle. The right image depicts the situation after a 3 minutes experiment where the robots were initially placed in lines around the shoes. Objects are put on the shoes to close the lights switch (normally activated by the weight of the user).

Chapter 4

Implementation

This chapter details the solution to the given problem and all the choices that resulted in it. The explanation will take a top-down approach, first reviewing the hardware and the general code architecture. It will then go deeper in the details.

The first question one could ask is: why swarm robotics for such an application? The answer to that question lays in section 2.2. Robustness, scalability and flexibility are characteristics that make swarms of robots really interesting in unknown environments (Brambilla et al., 2013). In case one of the agents is broken, we do not want to see the whole system collapse and leave the human unattended. Flexibility guarantees that the solution will work in different conditions, environments, which is an advantage for exploration and rescue. In case of loss of robots, scalability would maintain the protection performance to an acceptable level.

Insert more justification?

4.1 The Hardware

The following sections will go through the details of the robots we used to conduct our experiments, and the prototype enabling swarm control.

4.1.1 E-puck

The robotic platform chosen was the e-puck (Mondada et al., 2009). This robot model was made for educational purposes, ~~at university level~~. Its shape is cylindrical with a diameter of 7.5 cm. It is moved by two diametrically opposed wheels. The figure shows an e-puck from the laboratory. Several extensions were plugged onto it to increase its capabilities. The important sensors that were utilised are the proximity sensors, the omnidirectional camera sensor and the

Insert picture

virtual ground sensor. The proximity sensors are based on 8 infrared emitting pieces that return a value proportional to the proximity of an nearby obstacle in the [O; 1] interval. These pieces are ~~almost regularly~~ placed a along the perimeter of the robot. The omnidirectional camera is a vertical camera on top of the apparatus, aiming at a convex mirror to provide a 360° view of the environment. It translates this view into a list of colour blobs. A colour blob is a cluster of pixels presenting close colour characteristics, that is, almost being of the same colour (the degree of similarity can be tuned during calibration). During the calibration, among other parameters, one can tune the degree of similarity, the minimum size of the cluster, and the recognised colours. The last sensor is a bit special: it is not real, and not physically present on the board. It is simulated through the ARGoS simulator used to develop the controller of the robot. It returns data created inside the simulator to the robot, computed from the simulated environment. In our case, this 'simulated/false data' contains the colour of the ground, symbolising the presence of danger. We actually simulated red discs on the floor through the simulator to artificially set up dangerous areas that were not visible by the testing user. That way, the conditions of real life application were the closest possible to ours. An example of red zone is in figure 4.1.

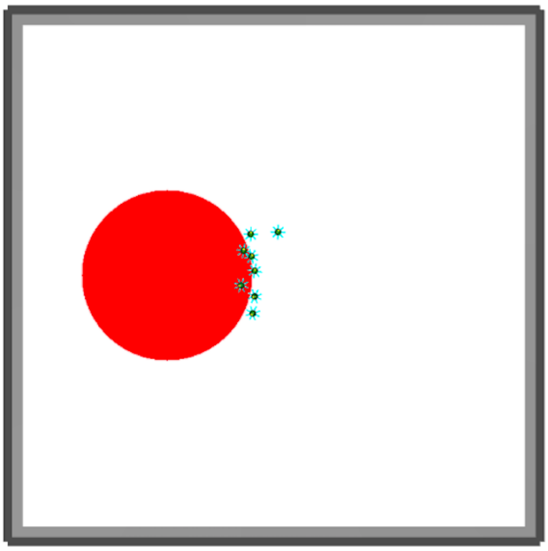


Figure 4.1 – The E-puck and its Virtual Sensor:

4.1.2 E-geta

As explained in chapter 3, one of the main issue was to enhance the robots so they could detect the user and position themselves with respect to him without any large external equipment. Large external equipments are not recommended since, in the targeted unknown environment, they might not be usable. For example, one might use a tracking system to get the position of the robots in real time and communicate it to the robots for them to adjust their speed. In controlled environment, this may work very well, but in the field it would be difficult to access that knowledge.

We thus opted for a compact, wearable device that would act as a 'landmark' for the robots: a pair of shoes. In order for the robots to understand on which side of the human they were (to go in front of the user), the two shoes had to emit a different message. The first idea the team had was to make use of the range and bearing sensors of the e-pucks to interact with the shoes. The shoes would have been equipped with infrared emitters. Unfortunately, tests with real robots demonstrated the inability of the range and bearing sensors to evaluate their distance to the human with precision. This method was abandoned for another one, more reliable: LEDs.

The omnidirectional camera sensor, explained in section 4.1.1, provided more accurate data. After validation of this method by tests on the real robots, the decision was made to create a prototype of a pair of shoes equipped with colour lights. Both shoes had to emit a different colour to give the robot information on the direction of the human. In section 4.2 we come back on the algorithm used to deduce the direction of the human from the observed colours.

The laboratory took inspiration from the Japanese 'geta' shoes for the design. Figure 4.2 shows a picture of a Japanese wooden shoe called geta. The right side of the figure is a picture of our prototype. Both have the same overall design. We chose this design in order to slow down the human's speed. Indeed, he speed of the robots is limited to maximum 10 cm/s ~~for~~ per wheel. Another advantage is the simplicity of the structure, and the low number of assembly parts. Figure 4.3 details the blueprints of the prototype.

Nice picture of shoes

Insert blue-prints

The base of the shoe is made with wood. The surrounding piece covering the LEDs was cut in sheets of semi-opaque plexiglass to diffuse the light. The LEDs are standard strip LEDs (one red strip, and one green). The final electronic circuit is very simple. The 9 volts battery is connected to two variable voltage regulators (step up) in parallel to increase the potential on the output. Each regulator is connected to a shoe LED strip. A separation is necessary since the

Insert nice photo of our shoes

Figure 4.2 – E-Geta: Left image by Haragayato [GFDL (http://www.gnu.org/copyleft/fdl.html) or CC-BY-SA-3.0 (http://creativecommons.org/licenses/by-sa/3.0/)], via Wikimedia Commons. Found on Wikipedia (2015).

Insert blueprints

Figure 4.3 – E-Geta Blueprints: Insert text

green LED strip requires more energy than the red one. To get an equivalent luminosity for both shoes, a different voltage had to be applied.

4.2 The Robot Behaviour

The first step of the design of the solution is to imagine how the system will look like and how we will implement it, the overall behaviour of the swarm. How do the robots move around the human? What shape will they ~~try~~ to form? This part is important because it will define the overall look and performance of the system.

The first shape that intuitively comes in mind is the circle. The circle is the most elementary shape in geometry. It offers the best ratio:

$$\frac{Surface}{Perimeter} = \frac{\pi r^2}{2\pi r} = \frac{r}{2}, \text{ where r is the radius of the circle.}$$

That means that fewer robots are needed for the same protected area, and more space for the human with a certain amount of robots than any other possible shape. ~~Luckily, it is also the easiest shape to realise in practice~~. The figure 4.5 represents the kind of circle that we would like to obtain for 6 robots and 1

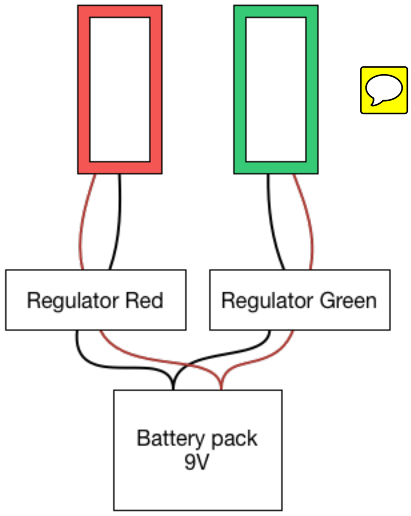


Figure 4.4 – The Circuit: The figure shows a sketch of the final electronic circuit for the shoes. The battery delivers 9 volts to two regulators in parallel, one for each shoe since each one of them need a different energy supply. Indeed, the green LED strip is more power hungry than the red one. The mass (black cable) is common for all the circuit.

human in the centre. All the robots are equally distanced from each other and the human. The human is protected in all directions. In presence of danger, the robots in contact with it should report it to the human and prevent him from going towards it, as seen on figure 3.2. In that case, the conditions concerning the target distance from the human and between robot may not be respected.

Implementing a robots swarm behaviour means writing a controller code for its individual components: the robots. The laboratory provides a template in the Lua and the C++ languages for this purpose. The logic of the individual behaviour is added inside callback methods called either by the simulator when performing simulations inside ARGoS (Pinciroli et al., 2012), or by the robot main method when testing on real robots. The first attempts to build the core of the controller were made in Lua to accelerate the process of trial and error. Indeed, Lua has a simpler syntax, is interpreted by the simulator and does not require any compilation process. Furthermore, the debugging is faster since the simulator includes an editor for the lua controller code. Although Lua is very convenient for quick implementation, it is not supported by the robots. Hence a translation was necessary to port the code to C++ in order to begin the tests on the real robots. After that, the implementation continued exclusively in C++

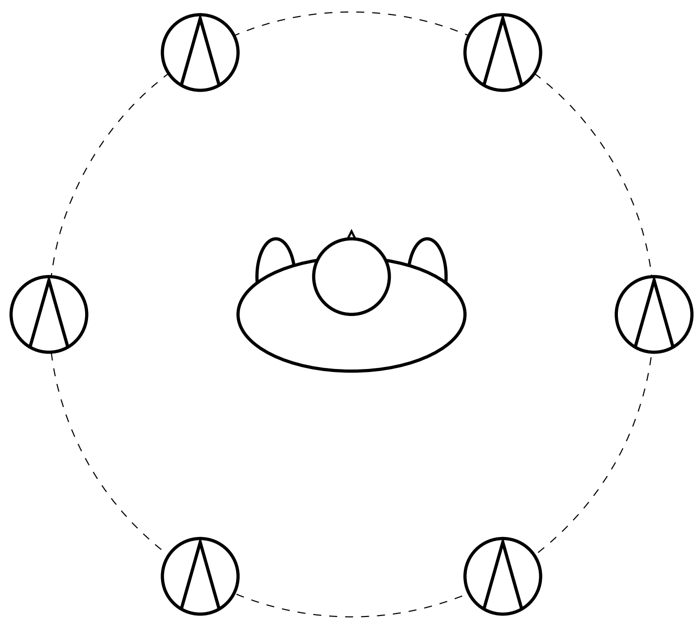


Figure 4.5 – Ideal Behaviour in Absence of Danger: This figure symbolises the ideal behaviour required in absence of danger. The swarm forms a circle to cover the widest protected surface for a given amount of robots. All the robots are equally distanced from each other and the human. The human is protected in all directions. In presence of danger, the robots in contact with it should report it to the human and prevent him from going towards it, as seen on figure 3.2. In that case, the conditions concerning the target distance from the human and between robot may not be respected.

since the tests on the robots were more frequent. The C++ template developed by the laboratory can be compiled for the ARGoS simulator and cross-compiled for the real robots (e-puck) without any modification. After the compilation, a simple transfer of the binary codes over WiFi allows the operator to store the controller on the robots to launch an experiment.

The final implementation of the controller is built on 2 layers. The upper layer is a deterministic finite state machine, containing for each state a specific behaviour in the lower layer. Figure 4.6 illustrates the whole structure of the upper layer. Figure 4.6 (left) is what could be called the 'core' of the behaviour, while fig. 4.6 (right) would be considered as additions to enhance the behaviour. The core part of the state machine is present on fig. 4.6 (right) since it is one of its constituents. ~~One can only limit himself to core part to understand the behaviour.~~ It is composed of 3 states: *Searching* when the robots do not detect any human nor obstacle, *Escorting* when a human is detected, and *Protection* when a human is detected and there is a danger nearby. If an obstacle is detected by a robot, the latter escapes from the core behaviour of the state machine and enters the *O.A. (Obstacle Avoidance* state. If the robot is blocked for too long, another state takes over to unblock it: *Unblocking*. When no more obstacle is in front of the robot, it can go back to its obstacle avoidance if any obstacle remains close. If none, it switches back to its core actions. The labels next to each transition must be read as follows: *H(uman)*, *D(anger)*, *O(bstacle)*, *F(ront obstacle)*, *T(hreshold crossed, stuck for too long and initiating rotation procedure to escape from the obstacles)*. *N* stands for the negation, so *NH* means 'no human found'. A complete state machine gathering all the states can be found in appendix A. Although the complete state machine is containing all the aspects of the behaviour, it does not allow to grasp the idea quickly. Dividing the final behaviour into several connected sub-behaviours modularises the solution. Adding a new state, a new sub-behaviour is easy.

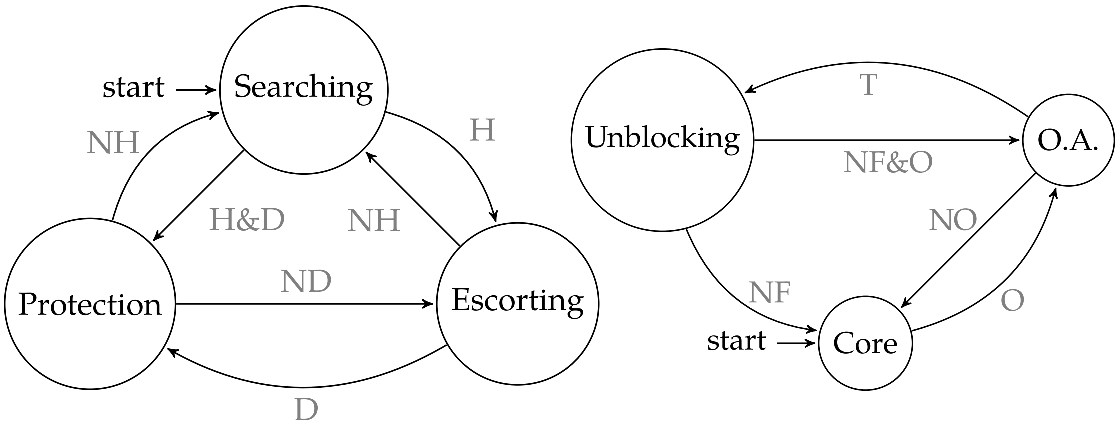


Figure 4.6 – State Machine of the Final Behaviour: This figure is the visual representation of the state machine of the final behaviour. Figure 4.6 (left) is what could be called the 'core' of the behaviour, while fig. 4.6 (right) would be considered as additions to enhance the behaviour. The core part of the state machine is present on fig. 4.6 (right) since it is one of its constituents. The abbreviation O.A. means 'obstacle avoidance'. The labels next to each transition must be read as follows: H(uman), D(anger), O(bstacle), F(ront obstacle), T(hreshold crossed, stuck for too long and initiating rotation procedure to escape from the obstacles). N stands for the negation, so NH means 'no human found'.

Once the actions the robots have to execute have been defined, we have to implement them, code them in the controller that will be run. So the next step was to find a way to translate those actions into code. Our behaviour is a kind of coordinated motion and pattern formation. Thus the intuitive way of implementing it was to make use of the virtual physics design. Using this framework, each robot is a particle subject to virtual forces exerted by the environment (the other robots, the obstacles, and other elements). Khatib (1986) was among the first to use this method. His goal was to implement an obstacle avoidance swarm behaviour where the obstacles create repulsive forces and the goals attractive forces. The overall resulting potential presented local minima at goals and maxima at obstacles. Reif and Wang (1999) introduces 'social potential fields' consisting in virtual forces applied on robots by other robots, obstacles, objective~~s,...~~ The robot resultant motion is defined by the sum of all the forces applied to it. The individual robots carry out the calculations themselves, so the final control is completely distributed. The laws they used are similar to those found in molecular dynamics (inverse-power laws). For example, a law could favour attraction over long distances and repulsion over short distances. One of these is the Lennard-Jones potential, depicted in figure 4.7. Inverse-power laws, while being extremely simple, can form interesting and elaborate patterns with molecules and plasma gases (Reif and Wang, 1999). Spears et al. (2004) proposed a framework they call 'physicomimetics' to grant distributed control over a large swarm of robots with 'artificial physics'.

More works on virtual physics?

The laws of physics force a system to go to a state of minimum energy, i.e., to reach a minimum of the potential function of the system. Since the force exerted on the system is proportional to the derivative of the corresponding potential –

$$\vec{f} = -\vec{\nabla}P,$$

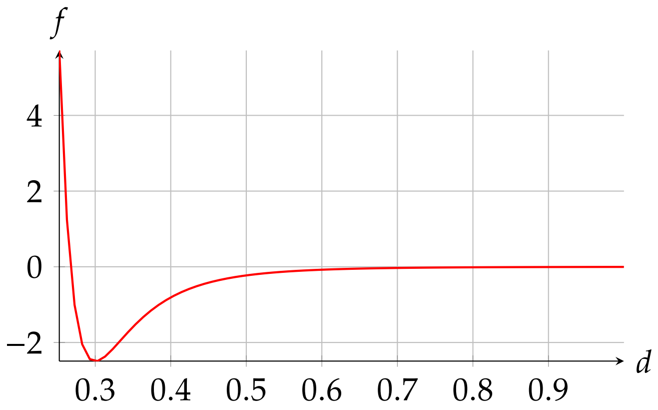
with $\vec{\nabla}$ being the nabla operator for the gradient computation, P the potential and $\vec{f}$ the force – the minimum of the potential function means the disappearance of the forces. For every robot to be at the desired location, the forces need to disappear. In this thesis, we only consider forces and not the virtual potentials associated to them. The implemented behaviour is expressed by in terms of virtual forces.

Using virtual physics offers some advantages over the other methods (Brambilla et al., 2013):

1. Only one mathematical formula fluently and elegantly converts all the inputs into outputs for the actuators. It removes the need for multiple rules and behaviours.

   $$f : \mathbb{R}^m \to \mathbb{R}^n : x_1, x_2, ..., x_m \to y_1, y_2, ..., y_n = f(x_1, x_2, ..., x_m),$$

   where m is the number of inputs, n is the number of outputs and f is the translating function.



$$f(d) = \epsilon \left[\left(\frac{\sigma}{d} \right)^{12} - 2 \left(\frac{\sigma}{d} \right)^{6} \right]$$

$$\epsilon = 2.5$$

$$\sigma = 0.3$$

Figure 4.7 – The Lennard-Jones Potential: This figure shows a graph of one of the most used virtual potentials in virtual physics, the Lennard-Jones potential, where ϵ is the gain, σ is the target distance and d is the current real distance. The equilibrium is reached at the global minimum at 0.3.

2. Multiple behaviours can be combined by simply summing the corresponding resulting vectors.

$$y_1, y_2, \dots, y_n = g(x_1, x_2, \dots, x_m) = \sum_{i=0}^{s} f_i(x_1, x_2, \dots, x_m),$$

   where s is the number of behaviour components.

One disadvantage is that it might be difficult to find an expression that implements perfectly the behaviour we want.

As written above, the robots need inputs to compute the values to send to the actuators, i.e., the wheels. The two types of inputs are the *colour blobs* and the *proximity sensor values*, respectively provided by the omnidirectional camera and the proximity sensors. Both are explained in section 4.1.1. 3 blob colours are used for our solution: red, green and blue, the three basic components of every colour in computer graphics. It was decided to a low number of colours to ease the calibration process and lower the amount of errors when detecting the blobs (wrong colour). To each blob is associated a distance - angle couple. The angle is taken from the front of the robot in radians. With these two value, the robot is able to situate all the blobs and use them as attractive or repulsive points. The proximity values are a list of 8 angle - value couples, where the angle is the position of the sensor on the perimeter of the robot. The whole perimeter of the robot is covered to detect any nearby obstacle. The value is comprised between 0 and 1, inversely proportional to the distance to the obstacle.

The following paragraphs explain in details the various forces that were implemented to obtain the desired behaviour from the fed inputs. They are grouped by states of the state machine in which they are used (see fig. 4.6). As explained earlier, each state in the upper layer of the general behaviour (the state machine) contains a sub-behaviour (a particular action) executed by means of virtual physics.

Searching When no human nor obstacle is around, the robot enters in the *Searching* mode with its LED off. It tries to stay at a constant distance from a detected colour blob, whatever colour it is. Since robots in *Escorting* or *Protecting* mode light their LEDs in blue, the searching robots always stay around the robots helping the human, whom they will detect at some point. This measure prevents the robots from going away too far from the human. This sub-behaviour is implemented through a sum of simplified Lennard-Jones virtual forces, one for each colour blob detected. The term 'simplified' is used because the force is not the real derivative of the Lennard-Jones potential, but a simplified version with lower exponents on the fractions:

Why a simplified version? Put the development.

$$\vec{f}(d) = \frac{-4\epsilon}{d}\left[\left(\frac{\sigma}{d}\right)^4 - \left(\frac{\sigma}{d}\right)^2\right]\vec{1_s} \tag{4.1}$$

The original version is:

$$\vec{f}(d) = \frac{-12\epsilon}{d}\left[\left(\frac{\sigma}{d}\right)^{12} - \left(\frac{\sigma}{d}\right)^6\right]\vec{1_s} \tag{4.2}$$

The simplified version is exposed in figure 4.8. If no blob is detected, the robot goes forward with a speed of 5 cm/s. The unit vector $\vec{1_s}$ is heading towards the source of the force (the applier).

Escorting If a human is found nearby and no danger is around, the controller enters into the *Escorting* state. The robot then tries to stay at a fixed distance from the human and other robots. If all the robots around the human follow the same pattern formation rules, a circle appears encircling him/her. The complete virtual force for this state is the sum of 3 components: the *human force*, the *robot repulsion force* and the *gravity force*. All three components are fed with the detected colour blobs as inputs to evaluate the distances and angles.

- The *human force* uses a stronger version of the Lennard-Jones force to keep the robot at a certain distance from the human (see figure 4.9). The force

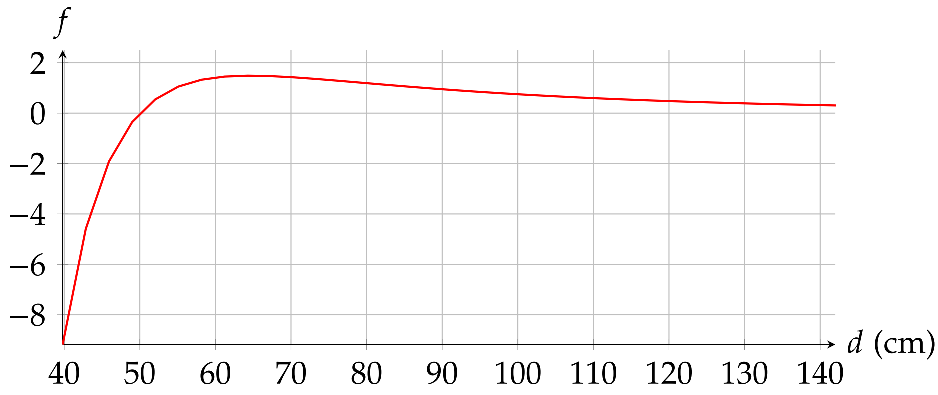


Figure 4.8 – The Simplified Lennard-Jones Virtual Force: This figure exposes a simplified version of the Lennard-Jones force derivated from the corresponding potential. Its expression is given by equation 4.1. The term ‘simplified’ is used because the force is not the real derivative of the Lennard-Jones potential, but a simplified version with lower exponents on the fractions. The parameters values are: $\epsilon = 100$ and $\sigma = 50$, and are conform to those used with robots. One can observe the root at 50 corresponding to the state of minimum energy in the system. Above 50, the force is positive, so the robot is attracted by the source applying the force. Below 50, it is negative, so the robot is repulsed from the source of the force.

is stronger in the attraction part, hence the name. The repulsion part is unchanged because of the more interesting asymptotic behaviour at $d = 0$, increasing the norm of the force quicker than the linear attraction replacement (in gray). Its expression is given by equation 4.3. Only the closest human colour blob is fed to the force computation.

$$\vec{f}(d) = \begin{cases} \frac{-4\epsilon}{d} \left[\left(\frac{\sigma}{d}\right)^4 - \left(\frac{\sigma}{d}\right)^2 \right] \vec{1_s} & \text{for} \quad d < \sigma \\ 15(d - \sigma)\, \vec{1_s} & \text{for} \quad d \geq \sigma \end{cases} \tag{4.3}$$

- The *robot repulsion force* maintains a fixed distance between the robots escorting or protecting a human (those with LEDs lit in blue). The simplified Lennard-Jones force is used, given by equation 4.1 and figure 4.8.
- The *gravity force* is the only force that can be deactivated via the experiment configuration file. The closest human colour blob to the human is taken as reference (the closest shoe). Then depending on colour of the blob (shoe), two different things can happen: the blob is red and the robot turns clockwise around it, or the blob is green and the it turns anti-clockwise. The goal is to go in front of the human like if the floor was ‘sloped down’

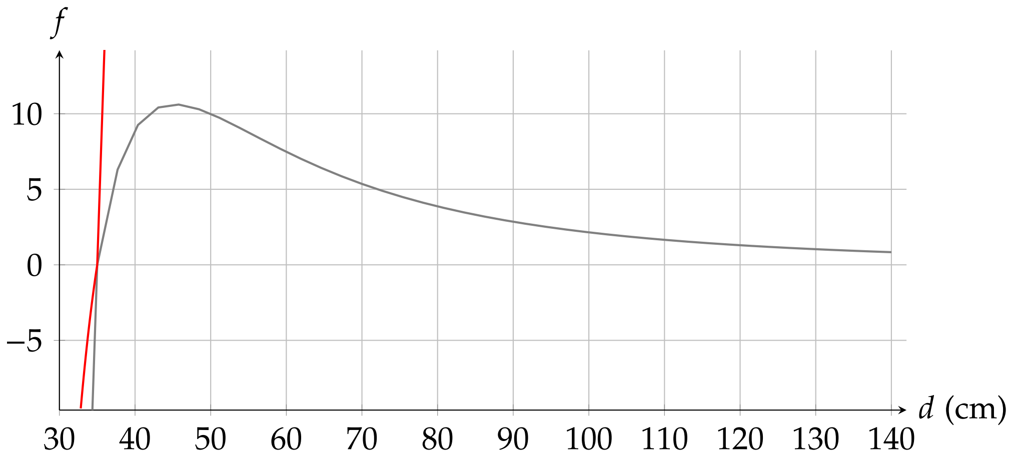


Figure 4.9 – The Stronger Lennard-Jones Virtual Force: This figure exposes a stronger version of the Lennard-Jones force derivated from the corresponding potential. Its expression is given by the equation system 4.3. The functions are plotted on the whole domain but only the red part is used. It allows to compare both values for the same distance. The force is stronger in the attraction part, hence the name. The repulsion part is unchanged because of the more interesting asymptotic behaviour at $d = 0$, increasing the norm of the force quicker than the linear attraction replacement (in gray). The parameters values are: $\epsilon = 500$ and $\sigma = 35$, and are conform to those used with robots. One can observe the root at 35 corresponding to the state of minimum energy in the system. Above 35, the force is positive, so the robot is attracted by the source applying the force. Below 35, it is negative, so the robot is repulsed from the source of the force.

to the front of the human, hence the name. Figure 4.10 illustrates the idea of the gravity force and figure 4.11 its norm. Two robots are both on the opposite side of the human. The left one sees the red shoe as the closest one and turns clockwise heading for the front. The right robot does the opposite with the same intention. This force is expressed in polar coordinates:

$$\vec{f}(\alpha) = \begin{cases} \epsilon\, \vec{1_\theta} & \text{for} \quad \alpha < 0 \\ -\epsilon\, \vec{1_\theta} & \text{for} \quad \alpha \geq 0 \end{cases} \tag{4.4}$$

$\vec{1_\theta}$ is the standard axis in polar coordinates $(\vec{1_r}, \vec{1_\theta})$ where the origin is the centre of the human. ϵ is the norm of the force.

Protection The *Protection* is reached when the robot detects a human and a danger. It makes its blue LEDs blink and stays at the border of the en-

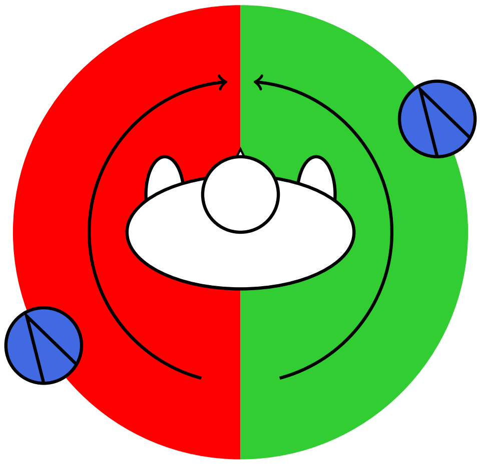


Figure 4.10 – The Gravity Virtual Force Concept: This figure illustrates the idea of the gravity virtual force, the only force that can be disabled in the configuration of the experiment. The closest human colour blob to the human is taken as reference (the closest shoe). Then depending on colour of the blob (shoe), two different things can happen: the blob is red and the robot turns clockwise around it, or the blob is green and the it turns anti-clockwise. The goal is to go in front of the human like if the floor was 'sloped down' to the front of the human, hence the name. Two robots are both on the opposite side of the human. The left one sees the red shoe as the closest one and turns clockwise heading for the front. The right robot does the opposite with the same intention.

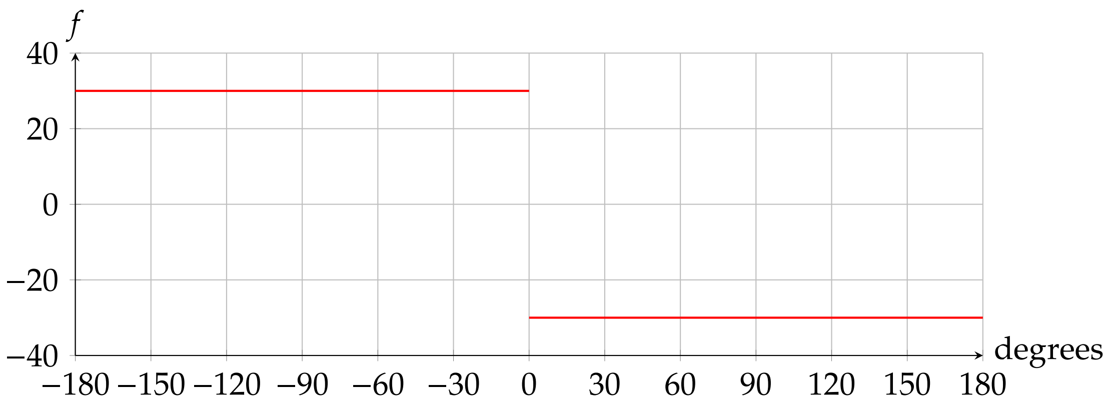


Figure 4.11 – The Gravity Virtual Force: This depicts the norm of the gravity virtual force with respect to the angle from the front of the human. Any angle outside this domain can fall back in the $[-180; 180]$ interval by normalising it. An angle of 0° means that the robot is in front of the human. The angle increases by turning anti-clockwise.

countered danger area while maintaining the escorting formation. Hence this sub-behaviour is based on the one from the *Escorting* state. The only difference is in the *human virtual force*. Since everything stays the same as in *Escorting*, except the fact that the robot must not cross the danger border, we just modify the target distance from the human. In presence of a danger, σ (the target distance) will be decreased incrementally until no danger is detected any more. As a result, the robot gets closer to the human like seen on figure 3.2 at page 9. We speak about *dynamic target distance*. Figure 4.12 illustrates the concept. On the figure, the same robot is represented at two different time steps. On the right, the robot is inside the dangerous area after the human took a step towards it. The robot enters *protection mode* and blinks. Its human target distance starts decreasing incrementally at each time steps by a user-defined amount.

Obstacle Avoidance [Idem for blocked state.]

Unblocking

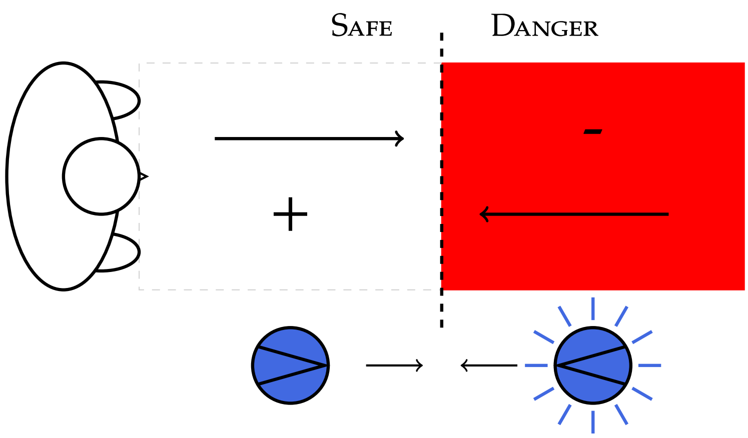
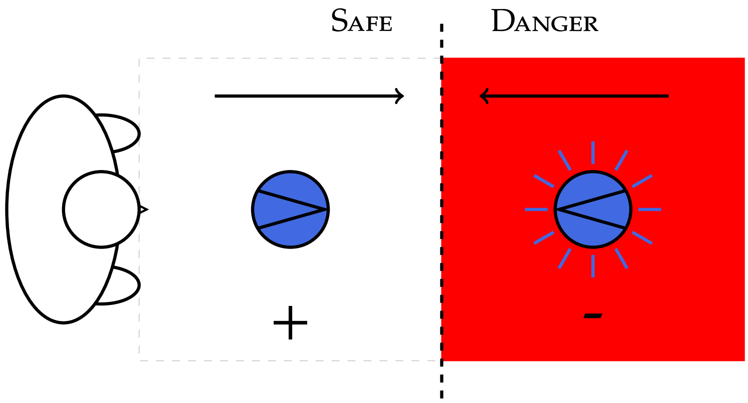


Figure 4.12 – The Dynamic Target Distance:


Safe
Danger
+
-

Chapter 5

Experiments

[Explain very clearly both types of tests (no human/human.]

5.1 Characterisation of the System

[The measurements and the tests on the final behaviour.]

5.1.1 Metrics

[Metrics I will use. Their description. How I will compute them.]

Correct Distance [Do the robots respect the correct wanted distance?]

Robot Density [Do the robots surround the human correctly?]

Time [Do they do that in a relatively low amount of time?]

5.1.2 Set-up

[How I am performing my experiments.]

5.1.3 Analysis

[All the graphs we discussed about. The evolution of the error over time. The analysis of the behaviour on basis of the criteria we defined.]

5.2 Demonstration

[What demonstration was done with the device. Add pictures. Describe perfectly.]

Chapter 6

Conclusion

[I've done this, this and this (1/2 pages). (Intro: "I'll do this, this...) Put sentences of type "so what?". Continuous text.

Future Works [The future works that would be interesting from my point of view.]

Other Robots

Guidance

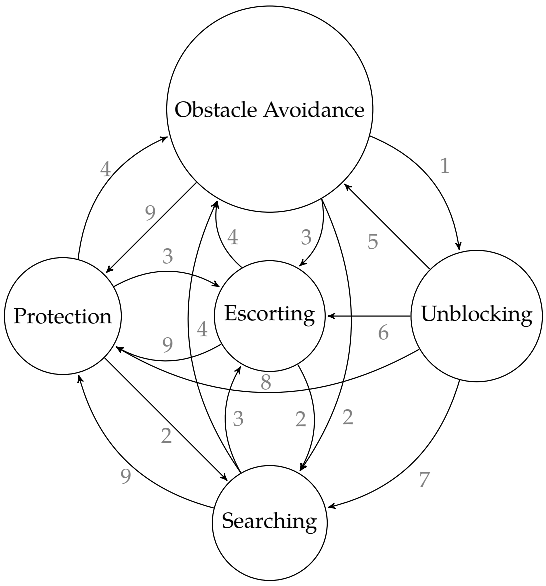
Zero Visibility Areas or Blind People:

Human Motion Synchronisation:

Vehicle Guidance:

Appendix A

The Complete State Machine



1. Amount of direction change (left/right) while having an obstacle around reaches a threshold.
2. No human & no obstacle.
3. No obstacle & human & no danger.
4. Obstacle.
5. No front obstacle & obstacle elsewhere.
6. No front obstacle & human & no danger.
7. No front obstacle & no human & no obstacle.
8. No front obstacle & human & danger.
9. No obstacle & human & danger.

Figure A.1 – State Machine of the Final Behaviour: This figure is the visual representation of the state machine of the final behaviour. On the left, the states and their connections are drawn. On the right, the information on the conditions needed to take the transitions are listed.

Appendix B

E-puck

Appendix C

ARGoS

Appendix D

Arena Tracking System

Appendix E

Range and Bearing

Appendix F

Omnidirectional Camera

Appendix G

Controller Code

Appendix H

MATLAB Scripts Code

Appendix I

Human Detection Devices Blueprints

Bibliography

Khelifa Baizid, Zhao Li, Nicolas Mollet, and Ryad Chellali. Human multi-robots interaction with high virtual reality abstraction level. In *Intelligent Robotics and Applications*, pages 23–32. Springer, 2009.

Manuele Brambilla, Eliseo Ferrante, Mauro Birattari, and Marco Dorigo. Swarm robotics: a review from the swarm engineering perspective. *Swarm Intelligence*, 7(1):1–41, 2013.

Mike Daily, Youngkwan Cho, Kevin Martin, and Dave Payton. World embedded interfaces for human-robot interaction. In *System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on*, pages 6–pp. IEEE, 2003.

Sanza T Kazadi. *Swarm engineering*. PhD thesis, California Institute of Technology, 2000.

Oussama Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *The international journal of robotics research*, 5(1):90–98, 1986.

James McLurkin, Jennifer Smith, James Frankel, David Sotkowitz, David Blau, and Brian Schmidt. Speaking swarmish: Human-robot interface design for large swarms of autonomous mobile robots. In *AAAI Spring Symposium: To Boldly Go Where No Human-Robot Team Has Gone Before*, pages 72–75, 2006.

Francesco Mondada, Michael Bonani, Xavier Raemy, James Pugh, Christopher Cianci, Adam Klaptocz, Stephane Magnenat, Jean-Christophe Zufferey, Dario Floreano, and Alcherio Martinoli. The e-puck, a robot designed for education in engineering. In *Proceedings of the 9th conference on autonomous robot systems and competitions*, volume 1, pages 59–65. IPCB: Instituto Politécnico de Castelo Branco, 2009.

Carlo Pinciroli, Vito Trianni, Rehan O'Grady, Giovanni Pini, Arne Brutschy, Manuele Brambilla, Nithin Mathews, Eliseo Ferrante, Gianni Di Caro, Frederick Ducatelle, Mauro Birattari, Luca Maria Gambardella, and Marco Dorigo.

ARGoS: a modular, parallel, multi-engine simulator for multi-robot systems. *Swarm intelligence*, 6(4):271–295, 2012.

Gaëtan Podevijn, Rehan O'Grady, and Marco Dorigo. Self-organised feedback in human swarm interaction. In *Proceedings of the workshop on robot feedback in human-robot interaction: how to make a robot readable for a human interaction partner (Ro-Man 2012)*, 2012.

John H Reif and Hongyan Wang. Social potential fields: A distributed behavioral control for autonomous robots. *Robotics and Autonomous Systems*, 27(3):171–194, 1999.

Erol Şahin. Swarm robotics: From sources of inspiration to domains of application. In *Swarm robotics*, pages 10–20. Springer, 2005.

William M Spears, Diana F Spears, Jerry C Hamann, and Rodney Heil. Distributed, physics-based control of swarms of vehicles. *Autonomous Robots*, 17(2-3): 137–162, 2004.

Wikipedia. Geta (footwear) — wikipedia, the free encyclopedia, 2015. URL `\url{http://en.wikipedia.org/w/index.php?title=Geta_(footwear)&oldid=651925222}`. [Online; accessed 20-May-2015].