Raphaël ABELLAN–ROMITA

Robotics

Internship Report

Obstacle avoidance with robust path generation

An Internship report submitted in fulfillment of the requirements for the degree of 2nd Year Internship

in the

SPID
ROB

"With regard to robots, in the early days of robots people said, 'Oh, let's build a robot' and what's the first thought? You make a robot look like a human and do human things. That's so 1950s. We are so past that."

Neil deGrasse Tyson

Abstract

2nd Year Internship at Institut de Robòtica i Informàtica Industrial

Obstacle avoidance with robust path generation

Internship Report

by Raphaël ABELLAN–ROMITA

During an 12-weeks long internship at the Universitat Politècnica de Catalunya, I worked on a robot in a simulated environment. In that environment, I designed a trajectory tracking strategy robust enough to work even with very noisy sensor values.

I also modified a program using box particle filtering to diminish the noise of the sensors. Such modifications allowed me to get better sensor values, although those were sometimes a bit erratic and random.

Once done, I added an obstacle avoidance algorithm through a vector field, predicting the future position of the bot, given all the boxes in which the robot had chances to be in in the next step.

Résumé

Stage de seconde année à Institut de Robòtica i Informàtica Industrial

Evitement d'obstacles avec génération robuste de trajectoire

Rapport de stage

par Raphaël ABELLAN–ROMITA

Au cours d'un stage de 12 semaines à l'Universitat Politècnica de Catalunya, j'ai travaillé sur un système robotique simulé. Dans cet environnement, j'ai conçu une stratégie de suivi de trajectoire assez robuste pour fonctionner même avec des retours de capteurs très bruités.

J'ai également modifié un programme utilisant un algorithme de box particle filtering pour diminuer le bruit des capteurs. De telles modifications m'ont permis d'obtenir de meilleurs retours de capteurs, bien qu'ils soient parfois un peu erratiques et aléatoires.

Une fois terminé, j'ai ajouté un algorithme d'évitement d'obstacles grâce un champ vectoriel, en prédisant la position future du robot, compte tenu de toutes les cases dans lesquelles le robot avait des chances d'être dans l'étape suivante.

Contents

List of Figures

List of Abbreviations

UPC	**U**niversitat **P**olitècnica de **C**atalunya
BPF	**B**ox **P**article **F**iltering
GUI	**G**raphical **U**ser **I**nterface
ERL	**E**uropean **R**obotic **L**eague
IMU	**I**nertial **M**easurement **U**nit

Chapter 1

Mission of the research center

1.1 The Polytechnic University of Catalonia

The Polytechnic University of Catalonia, named in Catalan Universitat Politècnica de Catalunya, currently referred to as BarcelonaTech and commonly named just as UPC, is the largest engineering university in Catalonia, Spain. It also offers programs in other disciplines such as mathematics and architecture.

UPC's objectives are based on internationalization, as it is one of Europe's technical universities with the most international PhD students and the university with the largest share of international master's degree students. UPC is a university aiming at achieving the highest degree of engineering/technical excellence and has bilateral agreements with several top-ranked European universities.

UPC is a member of the Top Industrial Managers for Europe network, which allows for student exchanges between leading European engineering schools. It is also a member of several university federations, including the Conference of European Schools for Advanced Engineering Education and Research (CESAER) and UNITECH.

The university was founded in March 1971 as the Universitat Politècnica de Barcelona through the merger of engineering and architecture schools founded during the 19th century. As of 2007 it has 25 schools in Catalonia located in the cities of Barcelona, Castelldefels, Manresa, Sant Cugat del Vallès, Terrassa, Igualada, Vilanova i la Geltrú and Mataró. UPC has about 30,000 students and 2,500 professors and researchers

1.2 Institut de Robòtica i Informàtica Industrial

The Institut de Robòtica i Informàtica Industrial is a Joint Research Center of the Spanish Council for Scientific Research (CSIC) and the Technical University of Catalonia (UPC).

The Institute has three main objectives: to promote fundamental research in Robotics and Applied Informatics, to cooperate with the community in industrial technological projects, and to offer scientific education through graduate courses. The Institute's research activities are organized in four research

lines.

Three of them tackle various aspects of robotics research, including indoor and outdoor human-centered human-safe robotics systems, and the design and construction of novel parallel mechanisms. Efforts in the fourth line are aimed at research on energy efficiency, in fuel cells research, and on the management of energy systems.

Chapter 2

Objectives

2.1 Objectives as presented

The internship goals were, at first, defined as such:

- Study of realistic models of autonomous robots
- Study of trajectory tracking strategies
- Implementation of trajectory tracking strategies in nominal case and robust case (in simulation)
- Implementation of Obstacle Avoidance

Although those goals seem well defined, those were given before my co-worker, Mr Bernardes had begun his internship. Eventually, those goals changed in between. When I arrived at the IRI, those goals had changed to such:

- Study of realistic models of autonomous robots
- Study and Understanding of box particle filtering for state estimation
- Study of trajectory tracking strategies
- Implementation of trajectory tracking strategies
- Implementation of a control strategy to minimize the probability of shocks

2.2 Evolutions and changes

During the Internship, I had to adapt to all the changes necessary. One of the biggest parts of my work was retro engineering the BPF code, and changing it from a once-over to a step-by-step system. Unless this change was made, the program would have been simply unusable. It eventually took almost a third of my time.

2.3 Issues and Challenges

From what I could determine, Interval analysis, and vector fields were not something used usually at the UPC laboratory. Such methods were greatly different to what they were used to, and we, Mr Bernardes and myself, brought a lot of new ideas and possibilities. Eventually, the researchers' program was functionally better, in the sense that, even though their results had a slightly bigger error margin, their program was still much faster.
Still, those new ideas may yield new methods, new algorithms combining the best of both worlds.

Chapter 3

Technical work

3.1 Code Refactoring

When working with new code, the first thing to do is to understand it. Then, you must do the necessary modifications.

3.1.1 Understanding the code

When I arrived at IRI, another student was already working on applying particle filtering on a robot simulation. He was doing his third-year internship and was working at the IRI since March. He based his work on Mr Jaulin's latest research: (L.Jaulin, 2016).

When first discovered, the code was mainly organized in three parts:

- The Interval class
- The Box particle filtering
- The Main

The Interval Class

The interval class was directly inspired by Mr Jaulin's works and courses (L. Jaulin and Walter, 2001). It's a simple class enabling the use of Interval analysis in Matlab, by creating a new object, and overloading the standard comparators and operators.

As such, this new object is in fact defined by a couple upper bound and lower bound for each dimension we are in. In this case, the simulation we work on is two dimensional, so the intervals have two bounds of each kind.

The Box particle filtering

"Resulting from the synergy between the sequential Monte Carlo method and interval analysis, box particle filtering is an approach that has recently emerged and is aimed at solving a general class of nonlinear filtering problems. This approach is particularly appealing in practical situations involving imprecise stochastic measurements that result in very broad posterior

densities.

It relies on the concept of a box particle that occupies a small and controllable rectangular region having a nonzero volume in the state space. Key advantages of the box particle filter against the standard particle filter are its reduced computational complexity and its suitability for distributed filtering.

Indeed, in some applications where the sampling importance resampling PF may require thousands of particles to achieve accurate and reliable performance, the box-PF can reach the same level of accuracy with just a few dozen box particles."(Amadou Gning, 2013)

As seen, the BPF is an efficient filter in our case. Aimed mainly at imprecise measurements, and using interval analysis, it integrates perfectly with what we are looking for.

The Main

Two more files are used in the program.

In the environment file, all the variables needed at the start of the program, such as the trajectory, the number of boxes, or the length of the simulation are defined.

The Mainbox file is simply the main loop, and the one where the graphical interface is created. It's no more than printing the trajectories for the real and the self-perceived bots.

3.1.2 Changing the way it works

The first thing to do was to change completely how the main loop worked. The new system needed to be robust, simple and calculate step after step.

In the first program, the point of interest was measuring the position and the speed of the robot. The system had no feedback loop, so there was no need to calculate step by step. All the calculation for the robot's position and movement were done before the program's actual work began. The new feedback loop needed all the parameters from a step to calculate the next one.

Information from the consign trajectory and the self-perceived robot were used to calculate the real position of the robot. The real position was then used to infer the new self-perceived position, feeding the loop again. As seen in Figure 3.1, this adds a level of complexity to the program.

From one point to another, the system had many calculations to do, primarily due to the BPF. Indeed, the BPF is a system which needs a lot of diverse points to operate efficiently. A sample size of 2000 points was sometimes reached. My feedback loop and control system needed to be light and

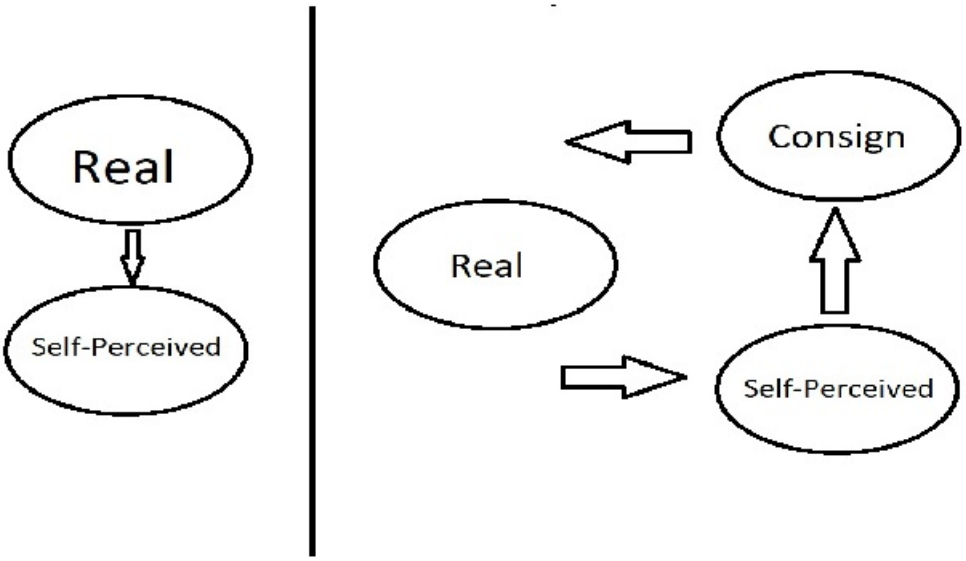

FIGURE 3.1: How the new control loop was implemented.

fast, or it would have slowed the program even more.

3.2 Simple control

After the new loop was added, I had to modify the informations sent from one part of the loop to another.

3.2.1 Adding precision

Before the work on control algorithm started, I had discovered that the information returned by the BPF were much more precise than simply using the sensor's feed, but consistency was lacking. Indeed, from one step to another, the BPF could render highly different results, and it would cause the real robot behaving erratically. To compensate, a low-pass filter was added, by using a sliding median of the last five values instead of only the last one.

Although such a system means that the correction is a bit late, it suppresses most of the erratic behaviour from the self-perceived bot. Giving more weight to the last value was tested, but the erratic behaviour came back with too much force, so the idea was dropped. More tests were made with a sliding median on more or less values, but five was the best compromise. The system was only around two steps late, so about 1/10th of a second.

This kind of treatment is often used on real robots. Usually, sensors have a high communication rate (around 500Hz for an IMU) whereas actuator control board work around 10Hz. That kind of differences means that, on real robots, sudden spikes and outliers from sensors can be minimized without

impacting the rate of information sent to actuators. Sadly, on this simulation, the sensor rate was limited by the BPF calculation rate. Eventually, I used a sliding median, but the trade-off was the slight lateness of the control algorithm.

3.2.2 Testing methods and taking control

The new loop, at the beginning, did nothing more than transmitting information from one part of the program to another. At first, a simple proportional control was added. A divergence from the original trajectory was met by a correction proportional to the deviation.

This system was too unstable or too slow. As it is often the case for proportional control, the bot would either overshoot, or not be able to reach the designated trajectory.

The next step was using a proportional derivative method. This solved the overshoot problems I had, and the corrections were much faster. This system was based on a feedback linearization, as described in (Jaulin, 2015). It was implemented in a simple way, by adding a control variable named U.

CODE 3.1: Control Code

```
function u = control(x,w,dw)
%control : creates consign vector from state and trajectory
    vector
    A = [-x(4)*sin(x(3)), cos(x(3));
          x(4)*cos(x(3)), sin(x(3)) ];
    Y = [x(1); x(2)];
    dY= [x(4)*cos(x(3));x(4)*sin(x(3))];
    V = (w-Y)+2*(dw-dY);
    u = A\V;
end
```

In this code, we suppose the existence of a nonlinear system described by:

$$\begin{cases} \dot{x} &= f(x) + g(x)u \\ y &= h(x) \end{cases} \tag{3.1}$$

This system describes the movements of our simulated robot. By differenciating y_i until inputs are involved inthe expression of the derivative, we obtain:

$$\begin{pmatrix} y_1^{(k_1)} \\ \vdots \\ y_m^{(k_m)} \end{pmatrix} = A(x)u + b(x) \tag{3.2}$$

If we suppose that A is invertible, we can write:

$$u = A^{-1}(x)(v - b(x)) \tag{3.3}$$

Thus, going back to the code 3.1, each member of the equation 3.3 can be joined with the similarly named variable in the code.

Once the control loop has been established, we need to add it in a main loop.

The Main loop in this program is copied here :

CODE 3.2: Main

```
for k=1:N
    %display iteration number
    disp(k);
    %create control vector
    % xc,dxc,ddxc,vc,thetac : full state of consign robot (position,
    % derivative, second derivative, speed, and angle)
    [xc,dxc,ddxc,vc,thetac]=consigne(k,ts);
    ur=control([x_med2 theta_measure v_measure],xc,dxc);
    %new step for the state of the system
    [x,v,theta,v_measure, theta_measure, pe, U]=realState(N, x, v, theta,ur, ts,S,NS);
    %box particle filtering
    [w_box_1,w_box_2,x_med] = Boxfilter1(Boxes,ts,stateF,U,pe,w_boxes{k});
    x_med_box(k,:)=x_med;
    w_boxes{k}=w_box_1;
    w_boxes{k+1}=w_box_2;
    %low pass filter (gliding median) on median position from BPF
    if k>5
        x_med2=sum(x_med_box(k-5:k,:))/5;
    else
        x_med2=x_med;
    end
    %creating positions of real, self-measured and consign robots
    x_tank=[x(1);x(2);theta]; % real robot
    xm_tank=[x_med2(1);x_med2(2);theta_measure]; %measured robot
    xc_tank=[xc; thetac]; %consign robot
    %saving those positions in lists.
    x_tank_list(:,k)=x_tank;
    xm_tank_list(:,k)=xm_tank;
    xc_tank_list(:,k)=xc_tank;
end
```

As seen in this code, each step of trajectory calculation is immediately followed by a step of BPF on the information given by the simulated sensors.

As a side note, I also tried to adapt a H-infinity control method, but a few problems blocked my way. The more evident one was that the model I used was far from enough for this method to yield good results. And of course, the use of a BPF resulted sometimes in singularities or in a kind of saturation that rendered the method completely unusable.

3.3 Obstacle Avoidance

Now that the robot was able to follow a trajectory, with more or less success, depending on the precision reached by the BPF, it needed to avoid obstacles and walls.

3.3.1 Trials and errors

One of the best ways to avoid obstacles in a partially known environment is a vector field. In my case, I had to add the right vector to the vector I had already created with the control system.

My first try was simply by adding repulsive vectors stemming from boxes colliding with a wall. This test was far from successful, since the self-perceived robot, a bit late because of the low-pass filter, would often make a U-turn, and completely lose itself. Indeed, the vector field, radiating only from colliding boxes, would overrun the control vector and push back the robot until the robot was far enough from the consign for shenanigans to ensue.

I then tried to use various parts of the map, first creating a single vector, then with multiple vectors, but I then determined that I had to use the whole known surroundings.

3.3.2 Creating a vector field

I had to devise a way for the robot to be aware of his surroundings as a whole.

CODE 3.3: Vector Field

```
function [Mx,My] = fieldmat(envimat,k)
%% fieldmat : creates two vector fields, acting as repulse
    fields from the walls.
% - Inputs=
%   -envimat - DOUBLE ARRAY, environement matrix
%   -k - optional - DOUBLE, strength of the vector field
% - Outputs =
%   -Mx - DOUBLE ARRAY, matrix of the x-axis part of the
    vector field
```

```
%   -My - DOUBLE ARRAY, matrix of the y-axis part of the
    vector field
    if (exist('k')==0), k=12; end;
    [m,n]=size(envimat);
    walls=envimat==ones(m,n);
    free=envimat==2*ones(m,n);
    Mx=zeros(m,n);
    My=zeros(m,n);
    Mx(:,1)=k;
    Mx(:,end)=-k;
    My(1,:)=k;
    My(end,:)=-k;
    for i=2:m-1
        for j=2:n-1
            Mx(i,j)=k*(walls(i,j-1)-walls(i,j+1));
            My(i,j)=k*(walls(i-1,j)-walls(i+1,j));
        end
    end
    Mx=imgaussfilt(Mx,5).*walls+imgaussfilt(Mx,3).*free;
    My=imgaussfilt(My,5).*walls+imgaussfilt(My,3).*free;
```

I created a small sub-program able to infer a repulsive vector field from an image of the obstacles.

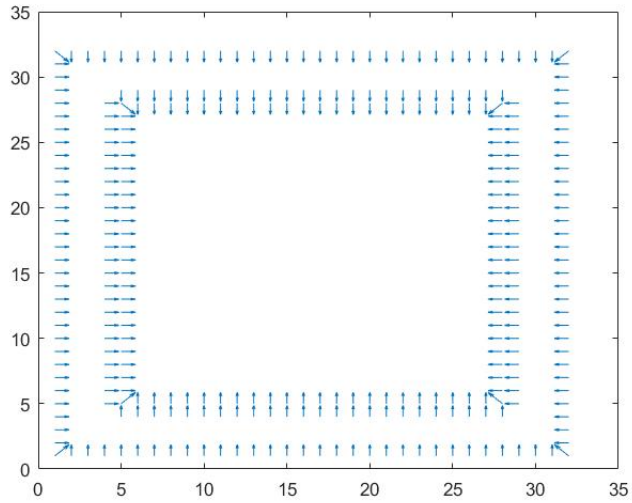

FIGURE 3.2: Quiver of basic vector field created from the obstacles.

This program has two parts. First, a repulsive vector is added in each box next to an obstacle. This creates a series of vectors each of equal power, as seen in Figure 3.2.

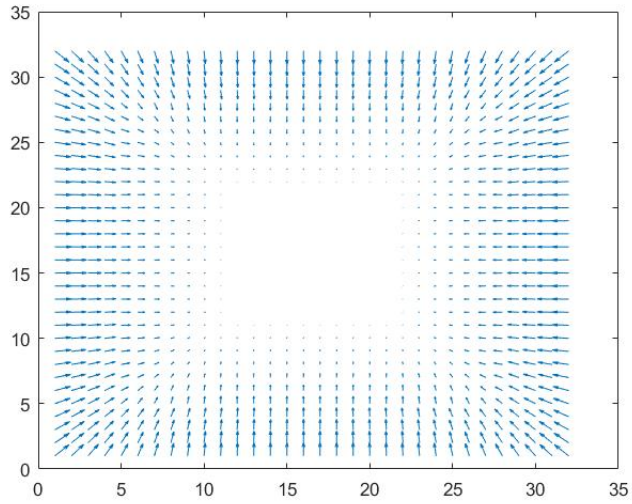

FIGURE 3.3: Gaussian blur added to the first vector field.

After that, a gaussian blur is added to the matrix, using a bigger parameter on the obstacle parts than on the free parts of the playground. This ensures that, if the self-perceived robot gets in an obstacle, it will be met by a stronger repulsive force. The result of the blur can be seen on Figure 3.3

3.4 Graphic User Interface

Since my work was a bit tedious on something so dry, I decided to add a graphical interface.

3.4.1 Adding an interface

In this interface, I decided first to add the choice to show each component separately. Checkboxes enable the visualisation of the robot, the environment, the boxes or the landmarks. Those choices will be reflected on the visualisation after clicking on show.

The simulation length and the percentile of boxes shown can be adjusted via two sliders. Those are useful to mitigate the precision of the simulation with the time spent on calculations.

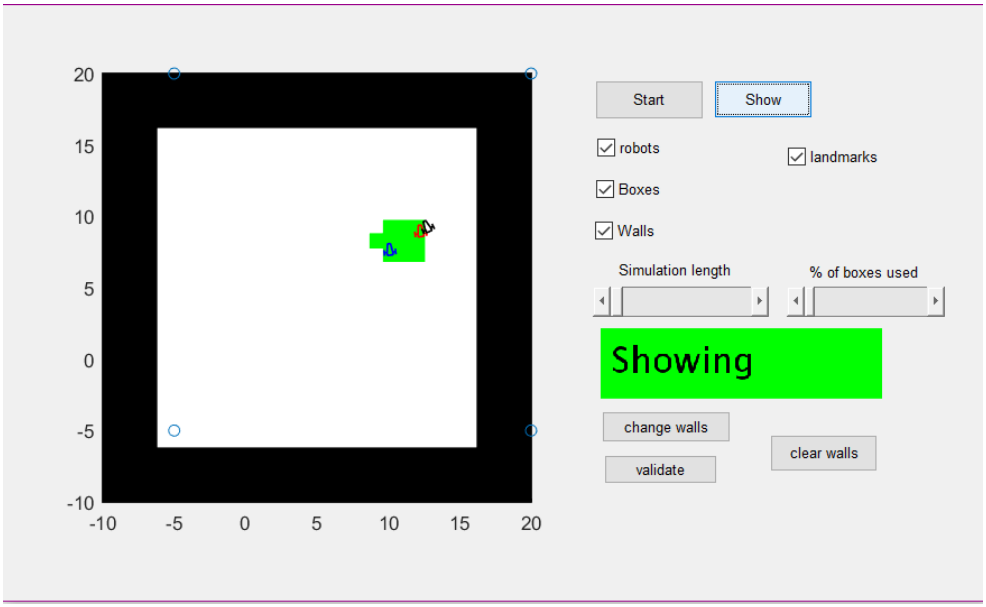

FIGURE 3.4: Full GUI for the algorithm

Part of the challenge of the algorithm was to make it reliable enough for it to run in any environment. I then added a graphical way to change the place of the obstacles, to test the program in all the possibilities. It did prove quite useful, since it was how the vector field coefficients were adjusted.

3.4.2 The full program

The program is now organized in four parts:

- the BPF
- the Interval class
- the Vector field
- the GUI

Even though the first two have the same inner workings than at the beginning, the last two are new.

the Vector field

As mentioned in 3.3.2, a subprogram was added to create the necessary vector field. This program can be called at any moment, and will create a vector field with any size of matrix. If the size of the playground or the boxes changes, this program will still be able to work.

the GUI

Due to Matlab's way, the new main is simple.m, the file containing the GUI description. Part of the Main was tranferred in this new file, and part of the envionment file can now also be found there.

This is due to the variables that can be adjusted in the GUI. A lot of variables were also duplicated in handles, because those were used in multiple parts of the program, and had to be transferred from one part to another.

Chapter 4

Economic Impact

In an environment of public research, it's always a bit hard to understand clearly the economic impact of any action.

Although no action is clearly designed to yield an economic benefit, some may, by chance, bring reputation to the laboratory. Sadly, it was not the case of the systems I worked on. But, from what I saw and what I learned from the numerous scientists of the laboratory, a non-negligible number of their work is aimed to other business.

Other part of their work was tailored to participate in competitions such as the ERL or the SEAT autonomous driving challenge. Participating in those kind of challenges does not directly bring in money, since the cash prizes are rarely high, but sponsors may be attracted by the teams engaged in those challenges. Moreover, even just participating will bring recognition to the school.

In a nutshell, the economic model of a public research laboratory is far from the one of any company, and works in some kind of closed ecosystem.

Chapter 5

Contributions to the professionnal project

5.1 Skills Learnt

Among what I learnt during this internship, most of the things were linked directly or indirectly to technical knowledge.

The technical skills I learnt were as such:

- Knowledge of state estimation techniques for autonomous robots using box particle filtering
- Knowledge of trajectory tracking and avoidance strategies
- Implementation in the Matlab environment of nominal and robust tracking error methods
- Implementation in the Matlab environment of obstacle avoidance methods

But, of course, this can't be the only contribution. A great part of what I learnt was interpersonal skill in an international environment. Although it was far from my first trip to another country, it was the first time I had to work in another country than France. And, as in many of the countries with a strong latin influence, professional and private sphere have blurred boundaries.

This is even more highlighted in laboratory work, were there are no work schedules.

As a result, a good part of my work was done in a bar, at home, or anywhere with air conditioning. Discussions with co-workers were free-flowing, and no hierarchical boundaries were felt during these times of exchange.

This motus operandi allowed me to greatly improve my conversational and interpersonal skills.

5.2 Public research and Private Business

I already had a pretty clear view of what I wanted to do after the engineering school, but this experience made it even more precise.

I have a preference for Research and Development, but sadly, the free-flowing, even laid back management of a public laboratory is not something I like. Through my two internships, one at ENSTA Bretagne, and the other at the UPC, I can now say without a doubt that something like this is not made for me.

My next internship will be done in a private structure, to understand the differences, and maybe validate my choice.

Chapter 6

Conclusion

During this internship at the Universitat Politècnica de Catalunya, I worked for the first time on a simulated environment without applying this work to the physical world. I was met with numerous challenges, from a complete refactoring of the base program that had to be done before I could even begin my work to a lengthy program that I couldn't launch frequently.

Far from being a bother, those challenges helped me grow, and nurture new skills. Working in a constrained environment can be a problem, but taken by the right side, can become a strength.

Noisy sensors, long calculation times, complex subject, all of those were seen as challenges to overcome, and not only by the simplest way. Thinking outside the box was necessary and, as always, the best solutions came from where you would expect it the less.

Appendix A

Evaluation sheet

ENSTA Bretagne	**ASSESSMENT REPORT**
	This report should be signed, endorsed with company stamp by the internship supervisor and returned to ENSTA Bretagne by email (annie.giot@ensta-bretagne.fr)

Host organisation	INSTITUT DE ROBÒTICA I INFORMÀTICA INDUSTRIAL (CSIC-UPC)
Dates of the internship	6/6/17 - 1/9/17
Name of the intern	RAPHAËL ABELLAN - ROMITA

	Check the appropriate boxes				
Criteria	Professional	Commendable	Satisfactory	Needs improvement	Unsatisfactory
Interpersonal skills					
Adaptability	X				
Availability	X				
Corporate culture	X				
Team work	X				
Listening skills (or Listening and communication skills)	X				
Professional skills					
Autonomy	X				
Technical initiative	X				
Performance	X				
Leadership	X				
Capacity to analyse and suggest improvement	X				
Internship report					
Format, style, etc.		X			
Content		X			
Practical application for the host organisation		X			

Overall appreciation

RAPHAËL ABELLAN HAS DONE A VERY GOOD WORK DURING HIS INTERNSHIP.

In the event of a position becoming available, would you consider employing this student?	☒ YES
	☐ NO

Name of the supervisor: JOAQUIM BLESA **Date :** 29/9/17

Title/Position : POSTDOCTORAL RESEARCHER **Signature :**

Bibliography

Amadou Gning Branko Ristic, Lyudmila Mihaylova Fahed Abdallah (2013). "An Introduction to Box Particle Filtering". In: *IEEE Signal Processing Magazine* 30 (4).

Jaulin, L. (2015). "Feedback linearization". In: *Mobile Robotics*, pp. 55–59.

L. Jaulin M. Kieffer, O. Didrit and E. Walter (2001). "Applied Interval Analysis with Examples in Parameter and State Estimation, Robust Control and Robotics, Springer-Verlag." In:

L.Jaulin (2016). "Inner and outer set-membership state estimation." In: *Reliable Computing* 21, pp. 44–55.