**University of Bordeaux 1**
**Science and Technology**
**Computer Science specialty**


Second year of Master - Study and Research Project

# Mobility Models for UAV Group Reconnaissance Applications

MEMORY

**Customers :** AUTEFAGE Vincent and CHAUMETTE Serge

**Responsible of Directed Works :** AUTEFAGE Vincent and CHAUMETTE Serge

**Authors :** CASTAGNET Florian, ETCHEVERRY Jérémy, PAZIEWSKI Hayley,
TESSIER Alexis & TESTA Mickael

# Contents

# List of Figures

# Thanks

Thanks for the many councils of our teaching assistant mister Pascal DESBARATS and our instructors misters Philippe NARBEL and Adrien BOUSSICAULT, we were able to realize a serious and steady work. They indicated us methods to make a success of this project and explained us what they expected from us. We thank them for the help provided throughout the project.

The help of our clients, misters Serge CHAUMETTE and Vincent AUTEFAGE, was also very precious for us. The proposed article was really very interesting. The main subject of this study concerns a matter seen during this half-year. We thank them to have provided an interesting article.

We also wish to thank mister Arnaud CASTEIGTS, who provided us a lot of help for the understanding of the JBotsim library, and on the difficulties which we met concerning the implementation on JBotsim.

To finish, we thank the English professors, misters Jean-Christophe COQUIHAT and Jean-Jacques BERNAULTE. They helped us to correct this report and to improve it in linguistic term.

# Introduction

As part of our Master Computer formation at the University of Bordeaux 1 particularly in the matter Study and Research Project, we had to study a research article in groups of 5 people. Every group had to choose a subject of research among those proposed. Our group chose a topic related to the drones. The title of this article is *Mobility Models for UAV Group Reconnaissance Applications* written in collaboration by *Erik Kuiper* and *Simin Nadjm-Tehrani* published in 2006. The customers who suggested us studying this article are misters Serge Chaumette and Vincent Autefage. Our teaching assistant is mister Pascal Desbarats.

This project, for a period of 3 months, we were able to see the progress of the study of a research article, its reading in English, with the implementation of an algorithm, and going through a study of the existing. The aim of this matter is to make us study a research article, to extract an algorithm from it and to implement it. We must therefore study a model that meets the aerodynamic properties of drones.

Many mobility models exist and is often based on the real wolrd. This article specifically discusses two mobility models that are Random Mobility Model and distibuted Pheromon Model.

We tried to give the best possible image of our project through this memory. It represents each stage, from the study of the existing to the implementation of an algorithm. It also shows our organization as well as the multiple tests which we established and realized.

# Chapter 1

# Context

A drone is an unmanned aircraft remote-controlled or autonomous. It is equipped with different sensors (accelerometer, gyroscope, magnetometer, camera, etc..) And they allow it to move in its environment.

Today, UAVs are increasingly present on French territory and around the world. There are different types of drones with different missions. For example, some drones are monitoring (building, historical monuments ...), others are designed to collect information, others have military uses or transport.Two types of UAVs are currently present: the drones said civilian and military drones. In the military context, UAVs can be used to penetrate areas that may be too dangerous for humans. They may include mapping areas for the tracking area.

Much research has been dedicated to these devices for many years, including on cooperation and communication between them several drones. These fleets of drone used to perform tasks as efficiently as possible. We need to know how they will move together, collision avoidance, avoid them to do twice the same work etc ... These constraints can be resolved through mobility models based on communication.
Many research topics are dedicated to these types of mobility within fleets drones. They study the different ways of drones' moving in their environment all together.

The purpose of this article is to scan an area previously defined as soon as possible and if possible once per hour. The problem of this article is: **How well scan an area? As much as Quickly as possible in a limited time and at least, once every hour**.

# Chapter 2

# Analysis Of Existing

As said before, drone swarm are governed by these models mobility. Our study of the existing is therefore divided into several parts.

## 2.1 Summary of the problem

The issue of this paper is to scan an area as efficiently as possible and as quickly as possible, at least once per hour. This area will be scanned by a UAV swarm that will cooperate and communicate each other.

## 2.2 Existing models

Many mobility models exist with different characteristics. All these models are used to define movements for swarm of UAV. Most models are based on real-life situations such as road maps, or on the behavior of many animals (ants, birds, termites, wolves etc). We will expose some of them we consider the most important and most interesting for our case.

### 2.2.1 Random Walk

First described by Einstein in 1926, it was developed to mimic the extremely unpredictable movement of many entities in nature. An mobility node (MN) moves from its position to a new one by randomly choosing a direction and speed chosen by pre-defined ranges, [speedmin, speedmax] and [0, 2PI]. At the end of a constant time interval t or a constant distance traveled d, a new speed and direction are calculated. If a MN during his travel reaches a simulation boundary, it « bounces » off the simulation border with an angle determined by the incoming direction and continues to this new path. It's a memory-less mobility pattern because it retains no knowledge of its old locations and speed values. Therefore, the current position and speed of a MN is independent of its past location and speed. This can generate unrealistic movement because of sudden stops and sharp turns. If the specified time or distance of a MN motion is short, the resulting movement pattern will be a random roaming pattern restricted to a small part of the simulation area. The Figure **??** shows an example of the movement observed from this

model. All the MN begin in the center of the 300mx600m simulation area (at the position (150,300)). Every 60 seconds, each MN randomly choose a direction between 0 and 2PI, and a speed between 0 and 10m/s.

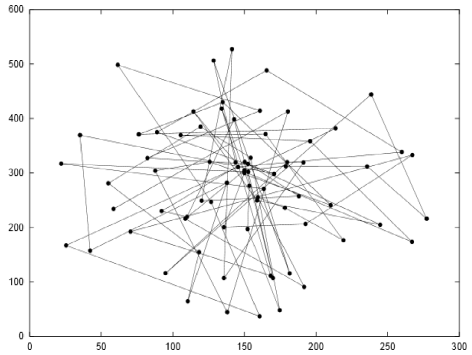



Figure 2.1: Resulting pattern of a MN using the Random Walk Mobility Model

**METTRE DES SCHEMA ET DECRIRE PLUS...et maintenant?**

### 2.2.2 Random Waypoint

Includes pause times between changes in direction and/or speed.Once this time expires, the MN chooses a random destination and a speed, that is uniformly distributed between [minspeed, maxspeed]. The movement pattern of a MN who uses the RWaypointMM is similar to the RWalkMM if pause time is zero and [minspeed, maxspeed] = [speedmin, speedmax]. The Figure 2.2 shows an example of the resulting pattern from the Random Waypoint Mobility Model.

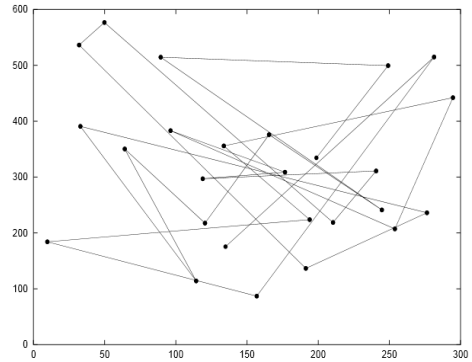



Figure 2.2: Resulting pattern of a MN using the Random Waypoint Mobility Model

During a run, MNs are initially distributed randomly around the simulation area. Figure 2.3 shows the average MN neighbors percentage of the MNs. For example, if there are 50 MNs in the network and a node has 10 neighbors, then the node's current neighbors percentage is 20%. Moreover, during the first 600 seconds, due to initially distributed randomly of the MNs around the simulation area, there is an high variability in the average MN neighbors percentage. In the paper, there is presented three possible

solutions to avoid this initialization problem. The first is to save the location of the MNs after the initial high variability and use this position as the initial starting point of the MNs in all future simulations. Second, initially distribute the MNs in a specific area to be a distribution more common the model, like fore example, initially placing the MNs in a triangle distribution. Lastly, discard the initial 1000 seconds of simulation time in each simulation trials, to ensure that the initialization problem is removed even if the MNs move slowly. But if the MNs move fastly, we can discard fewer seconds of simulation time. This third solution ensures that each simulation has a random initial configuration. Figure 5 : there is a complex relationship between node speed and pause time. A scenario with fast MNs and long pause times actually produces a more stable networks than a scenario with slower MNs and shorter pause times. Hence, long pause times(over 20 seconds) produce a stable network even at high speeds.

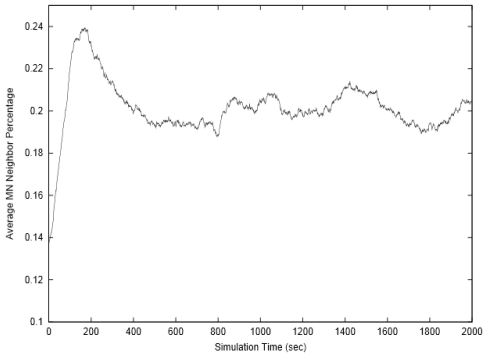



Figure 2.3: Average neighbor percentage vs. time

**METTRE DES SCHEMA ET a revoir**

### 2.2.3 City section

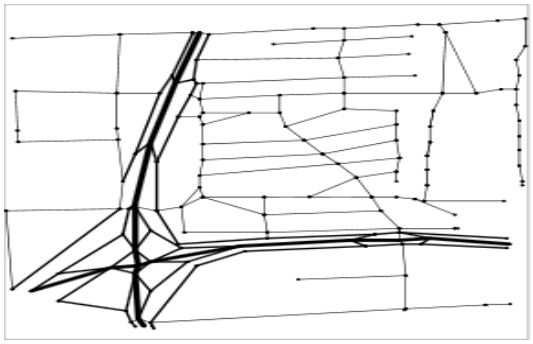
The aim of this model is to propose a realistic model. It bases on the vehicle's movement on road maps. The specificity of this model is the map depends on each geographical zone. Here, vehicles are mobile nodes.

The mapping data blocks are available from the office Americans of the census. The map information is retrieved from text files. These files are composed of several elements:

- The unique identifier for a specified road.
- The nature of the road: it can be a highway or a street.
- The longitude of the start of position.
- The latitude of the start of departure
- The latitude of arrived.
- The longitude of arrived.

All the intersections of the map are represented by nodes not mobile. This model also considers the volume of traffic : more traffic, the more the line representing the road will be thick. This model is also sensitive at the speed of cars (depending in particular on the hour). For example, we can consider that the speed of vehicular traffic in the morning is much smaller than at the beginning of the afternoon.

Here is the progress of this model. Every node begins in a point chosen randomly and its destination is chosen also randomly. The strategy of every node is to look for the shortest way until the destination. To achieve it, the algorithm of Dijkstra is used. Several parameters has to be considered in this algorithm as the speed or the traffic. This route can be dynamically changed. When a node reachs its destination, it begins again all the process described previously. Concerning the node's speed, it is limited to more or less 5% of the speed registered in the file.
For example, here is a representation of the model. We see here that the traffic is little sparse, but with very different speeds.



**(b) Region *B***: Street scenario corresponding to a square area of size 1900 m×1900 m

Figure 2.4: City application

**METTRE DES SCHEMA ET a compléter**

### 2.2.4 Natural Agent

**Ants**

Le modèle de mobilité des fourmis se base sur le fait que les fourmis construisent des réseaux de sentiers qui relient leurs nids avec des sources de nourritures disponible. Chaque fourmi qui se nourrit à le même programme :

- Elle doit éviter les obstacles.
- Aller ou elle veut (déplacement aléatoire) tant qu'il n'y a pas de phéromones.
- Si elle arrive à trouver de la nourriture, elle laisse des phéromones pendant un temps t pour pouvoir indiquer aux autres fourmis l'emplacement de la nourriture.
- Si la fourmi trouve de la nourriture, elle l'a ramène à son nid.

Les fourmis peuvent mourir. Soit elles ne trouvent pas de la nourriture assez rapidement et meurt de faim, soit elles se font tuer par d'autres prédateurs. Tous les chemins de phéromones conduisent à de la nourriture. Mais les phéromones s'évaporent dans le temps. Le taux de phéromone peut être renforcé si plusieurs fourmis passent par le même chemin.

**METTRE DES SCHEMA ET a compléter**

**Termites**

**Wasps**

le modèle basé sur les guêpes est composé de plusieurs caractéristiques :

- un chef qui répartit les différentes guêpes dans différents groupes
-

### 2.2.5 Birds and Fish

### 2.2.6 Wolves

# Chapter 3

# Scenarios

We saw that the issue of this article is "how well scan an area?".
So the scenario is the reconnaissance of an area. It takes place in a square with a side length of 30 kilometers.
For this scenario, we use 10 UAVs and a command and control center (C&C). Each UAV starts at the middle of the south edge heading north near the C&C. The UAVs have to scan entirely the map once per hour.

For the reconnaissance data, to be of any use, it needs to be transmitted to the users that need it. In this scenario, it's the C&C which needs it. So each UAV need a communication path to C&C to transmit the data collected. They have to maintain a contact with the C&C, so they need to be close enough a neighbour UAV.
Fortunately a very simple model of communication is chosen for the scenario. UAVs within 8000 meters of each other can communicate with infinite bandwidth. If they are further away no communication is possible.

Concerning the reconnaissance scan, an image resolution of 0.5 meters is chosen. The scan area is 2000x1000 meters thanks to an 8 megapixel camera and the image proportions 2:1 (width:lenght).

# Chapter 4

# Needs Analysis

## 4.1 Functional Requirements

- The area coverage
  We need to cover the area the most efficiently and rapidly.
- The connectivity
  UAVs need to be always in contact with the C&C to transmit the information at any time.

## 4.2 Non-Functional Requirements

# Chapter 5

# Schedule

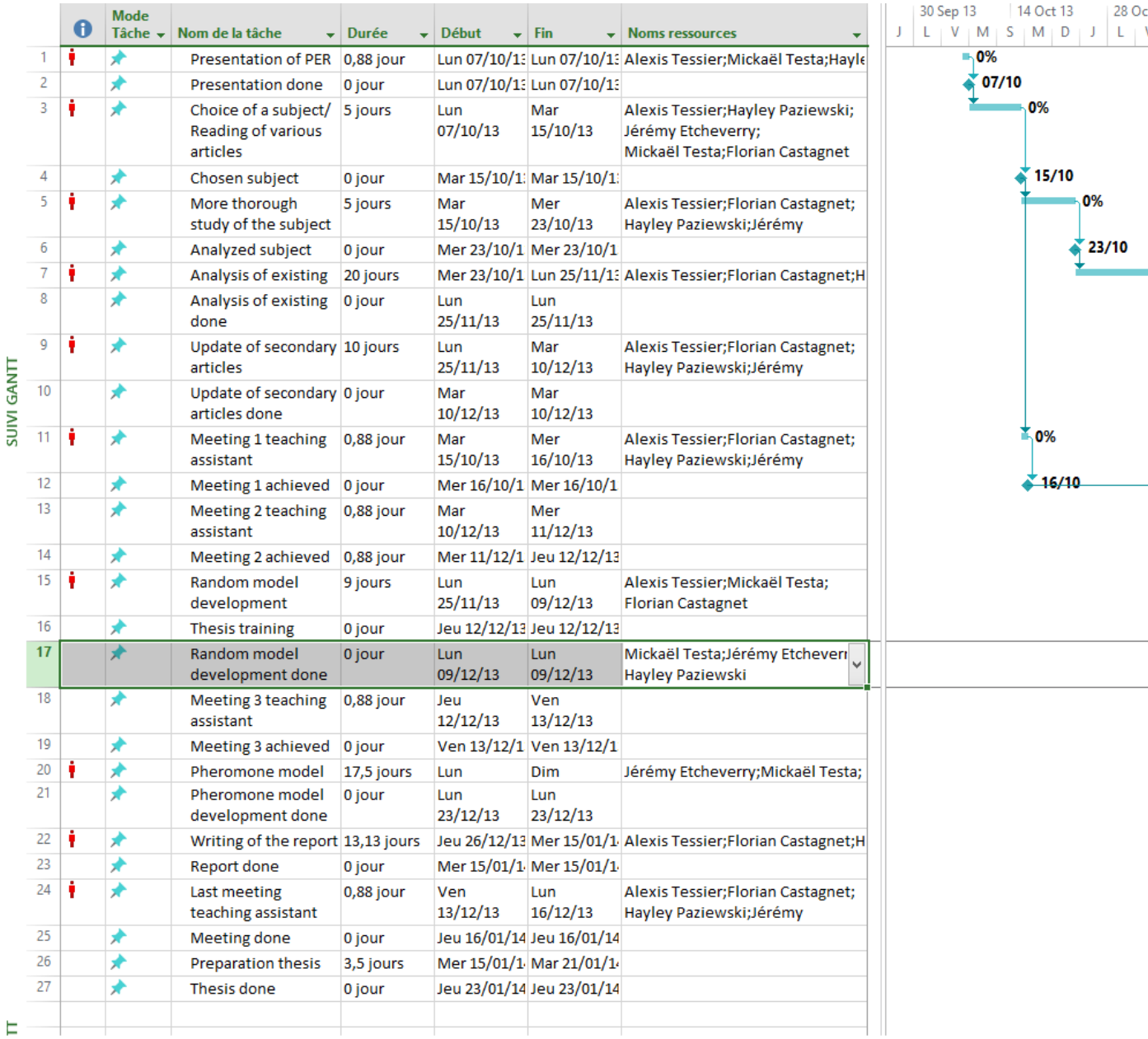
In this part, we are going to deal with the organization of the team on this project. TFirst, we identified the different tasks which will constitute this project. Thanks to elements identified previously, we made a Gantt diagram which is a tool used in project management, and allowing the affectation of a task for one person or a group of person, the time which this task will have to take. He allows to visualize the fulfillment of the tasks according to time.

## 5.1 Tasks List

A list of task was identified from the beginning by the project to be able to lead it most effectively possible from the beginning to the end.

- The choice of the subject. We took a subject which concerned drones because the majority of our team follow the subject Communicating Autonomous Systems.
- An analysis of existing concerning the communicating autonomous systems. We read about ten articles in touch with our main subject.
- An update of various articles previously studied is essential. Indeed, the speed of evolution of the technology today is so important that a written article two or three years ago may be out of date.
- The development of both present models in the studied article.
- The writing of this report.
- The preparation of the orals.

## 5.2 GANTT



| | Mode Tâche | Nom de la tâche | Durée | Début | Fin | Noms ressources |
|---|---|---|---|---|---|---|
| 1 | | Presentation of PER | 0,88 jour | Lun 07/10/13 | Lun 07/10/13 | Alexis Tessier;Mickaël Testa;Hayle |
| 2 | | Presentation done | 0 jour | Lun 07/10/13 | Lun 07/10/13 | |
| 3 | | Choice of a subject/ Reading of various articles | 5 jours | Lun 07/10/13 | Mar 15/10/13 | Alexis Tessier;Hayley Paziewski; Jérémy Etcheverry; Mickaël Testa;Florian Castagnet |
| 4 | | Chosen subject | 0 jour | Mar 15/10/1 | Mar 15/10/1 | |
| 5 | | More thorough study of the subject | 5 jours | Mar 15/10/13 | Mer 23/10/13 | Alexis Tessier;Florian Castagnet; Hayley Paziewski;Jérémy |
| 6 | | Analyzed subject | 0 jour | Mer 23/10/1 | Mer 23/10/1 | |
| 7 | | Analysis of existing | 20 jours | Mer 23/10/1 | Lun 25/11/13 | Alexis Tessier;Florian Castagnet;H |
| 8 | | Analysis of existing done | 0 jour | Lun 25/11/13 | Lun 25/11/13 | |
| 9 | | Update of secondary articles | 10 jours | Lun 25/11/13 | Mar 10/12/13 | Alexis Tessier;Florian Castagnet; Hayley Paziewski;Jérémy |
| 10 | | Update of secondary articles done | 0 jour | Mar 10/12/13 | Mar 10/12/13 | |
| 11 | | Meeting 1 teaching assistant | 0,88 jour | Mar 15/10/13 | Mer 16/10/13 | Alexis Tessier;Florian Castagnet; Hayley Paziewski;Jérémy |
| 12 | | Meeting 1 achieved | 0 jour | Mer 16/10/1 | Mer 16/10/1 | |
| 13 | | Meeting 2 teaching assistant | 0,88 jour | Mar 10/12/13 | Mer 11/12/13 | |
| 14 | | Meeting 2 achieved | 0,88 jour | Mer 11/12/1 | Jeu 12/12/13 | |
| 15 | | Random model development | 9 jours | Lun 25/11/13 | Lun 09/12/13 | Alexis Tessier;Mickaël Testa; Florian Castagnet |
| 16 | | Thesis training | 0 jour | Jeu 12/12/13 | Jeu 12/12/13 | |
| 17 | | Random model development done | 0 jour | Lun 09/12/13 | Lun 09/12/13 | Mickaël Testa;Jérémy Etcheveri Hayley Paziewski |
| 18 | | Meeting 3 teaching assistant | 0,88 jour | Jeu 12/12/13 | Ven 13/12/13 | |
| 19 | | Meeting 3 achieved | 0 jour | Ven 13/12/1 | Ven 13/12/1 | |
| 20 | | Pheromone model | 17,5 jours | Lun | Dim | Jérémy Etcheverry;Mickaël Testa; |
| 21 | | Pheromone model development done | 0 jour | Lun 23/12/13 | Lun 23/12/13 | |
| 22 | | Writing of the report | 13,13 jours | Jeu 26/12/13 | Mer 15/01/1 | Alexis Tessier;Florian Castagnet;H |
| 23 | | Report done | 0 jour | Mer 15/01/1 | Mer 15/01/1 | |
| 24 | | Last meeting teaching assistant | 0,88 jour | Ven 13/12/13 | Lun 16/12/13 | Alexis Tessier;Florian Castagnet; Hayley Paziewski;Jérémy |
| 25 | | Meeting done | 0 jour | Jeu 16/01/14 | Jeu 16/01/14 | |
| 26 | | Preparation thesis | 3,5 jours | Mer 15/01/1 | Mar 21/01/1 | |
| 27 | | Thesis done | 0 jour | Jeu 23/01/14 | Jeu 23/01/14 | |



Figure 5.1: Diagram of Gantt of our project

# Chapter 6

# Works Done

## 6.1 What we implemented

During two and a half months of this project, we implemented both models which were described in the article. As a reminder, these two models are a simple random and a model concerning pheromones. We also implemented the Random Waypoint model on the demand of our customers. In this section, we will explain the development work that we have provided and the different choices. We didn't find utility to make a UML diagram seen the simplicity of the architecture.

### 6.1.1 Architecture

#### Random Model

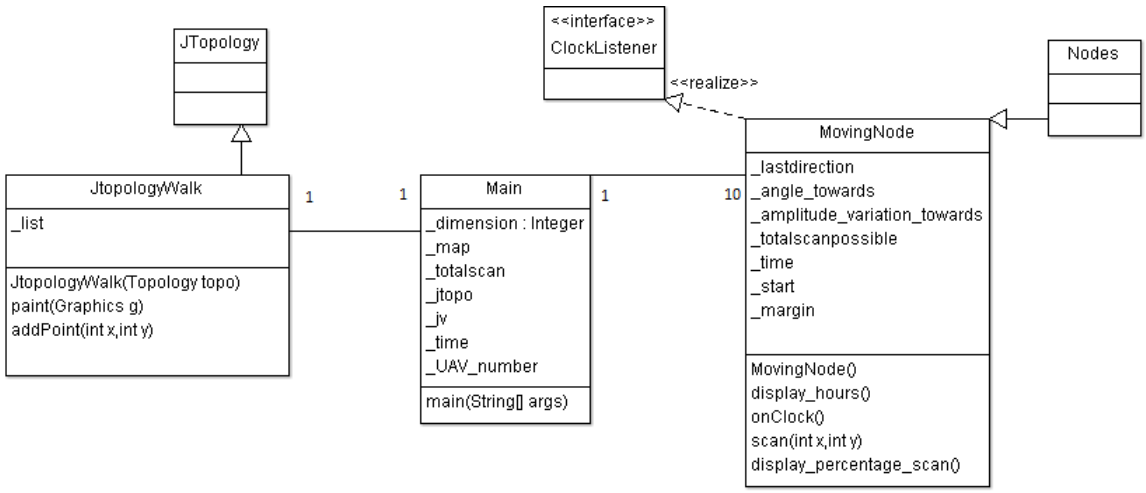



Figure 6.1: Diagram of class Random model

The description of all the classes will be made in the next section of this chapter.

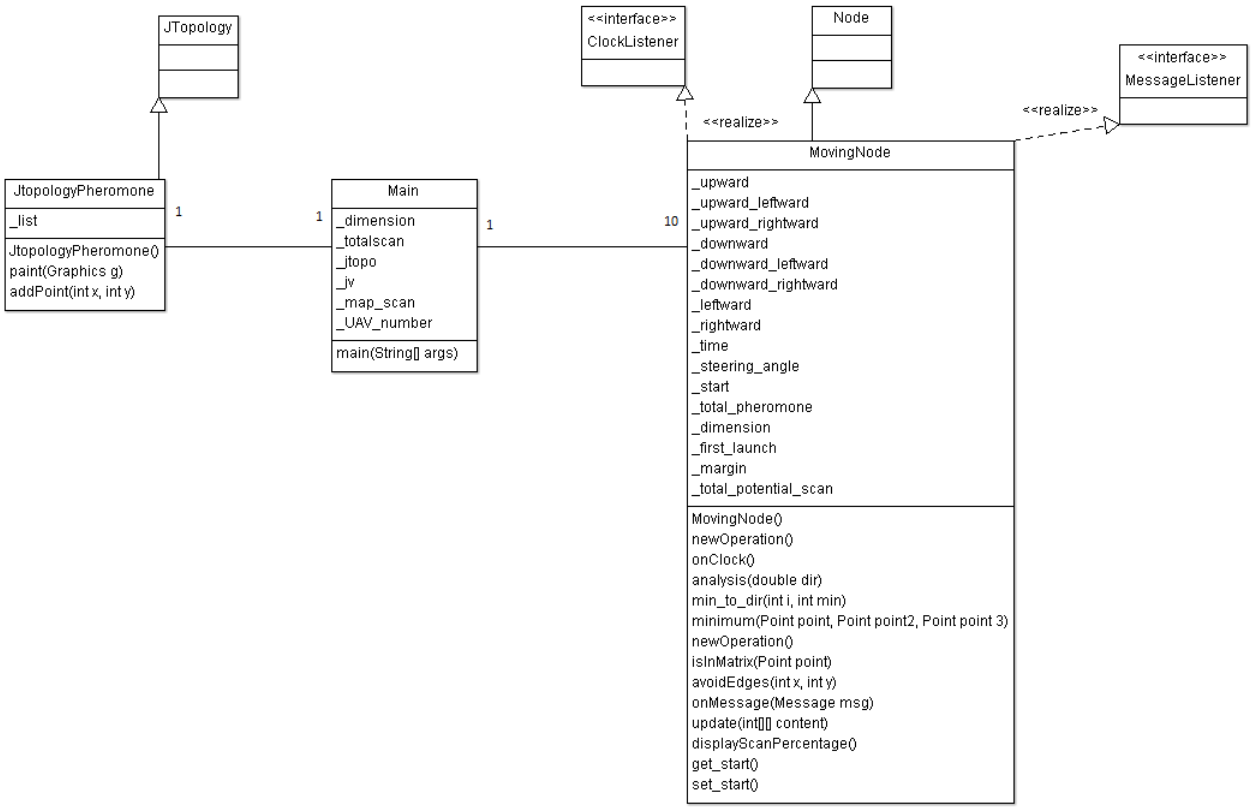



Figure 6.2: Diagram of class Pheromone model

**Pheromone model**

The description of all the classes will be made in the next section of this chapter.

### 6.1.2 Random model

To implement this model, we needed three classes.
The first class, which is called *Main.java*, allows as its name indicates it, launching the program. It is in this class that we define the size of the window, the matrix which will serve to simulate the scan of a zone (all the nodes will have the same matrix), and the instanciation of the various objects coming of JBotSim library. We have for example, an object named *topo* which is a topology. This object is the basis for our simulator. It is to this topology that we are going to be able to add mobile nodes (here planes).

We also instantiated a Jtopology's object type. We will explain this choice during the presentation of entities.

The second class of our architecture is called *MovingNode*. It is in this class that will be handled the movement of the various nodes. That's why this class inherits from the class *Node* and implement the interface *ClockListesner*, both coming from the JBotSim librairies. It inherits from the class *Node* because planes will be represented by mobile node, so we need the characteristics of that class. Moreover, It implements the interface *ClockListener* because we need that our nodes move all the seconds, so we need a top of clock to realize it. We will have a listener for time , as well as a method *onClock* which will be called to every top of horloge.

This class contains a constructor by default, allowing to instantiate and to initialize the different variable, and to associate images with the mobile nodes (call of the method *setProperty*). The random model gets characteristic to have no communication between the different nodes. To realize it, we pass an argument of 1 in the method *setCommunicationRange*.

As indicated in the article, the nodes of the random model change actions every seconds in functions of the last undertaken action. As a reminder, here is the table of action for the random model:

**Table 1. UAV random action table.**

| | Probability of action | | |
|---|---|---|---|
| **Last action** | **Turn left** | **Straight ahead** | **Turn right** |
| **Straight ahead** | 10% | 80% | 10% |
| **Turn left** | 70% | 30% | 0% |
| **Turn right** | 0% | 30% | 70% |

Figure 6.3: Random model

In our implementation, we have a variable *lastdirection* allowing to know the last action which was made. We associated figures with the last actions. If the last direction was the left, then the variable *lastdirection* takes the value 1, if it was straight, then we associate the value 2, and if it was to the right, we associate the value 3.

To respect as much as possible the reality, we had to modify the management of the

edges of the window. Indeed, originally in JBotSim, nodes can pass from an edge to the other one. For example, if they reached the left edge, they went to the right of the window, what is not possible in the real life. We blocked this problem by rectifying the position of the node, and by changing its angle of direction.

Each time a node scans a zone, it looks in the matrix if this position was already scanned (0 if the zone was never scanned, 1 otherwise). If it is, then it makes nothing, otherwise it puts the value to 1.

The third class of our architecture is called *Jtopology_Random*. It inherits from JTopology and is going to draw the scans footprint on the map. This class JTopology inherits from JPanel, and is situated in fact between a JViewer and a Topology.

### 6.1.3 Random Waypoint Model

This model is very similar to the previous. The only difference is their way to move.
In this model, the node choose one point randomly and move to it.
So the only modification compared to the previous model is the class *MovingNode* and more precisely the method *onClock*.
In this method, we choose a random point and we move the node until it. To do this, we change the direction of the node to the direction of the point chosen. When the node arrive at this point, we choose an other random point and so on.
In this model, we don't manage the edge of the window because all destination point are chosen on the window and so none UAV can go out the window.

### 6.1.4 Pheromone model

Concerning the architecture, the implementation of the model of pheromones is very similar. We find practically 3 even classy. Only the class *MovingNode* is really going to be different because the movements of nodes are going to depend on a lot of parameter. First of all, this class implements a new interface which is *MessageListener*. Indeed, in this model, nodes have to sent data each others (in this particular case their respective map).

The method which is going to change compared with the random model is the method *onClock*. Indeed, it is here that we are going to apply rules as described on the following image:

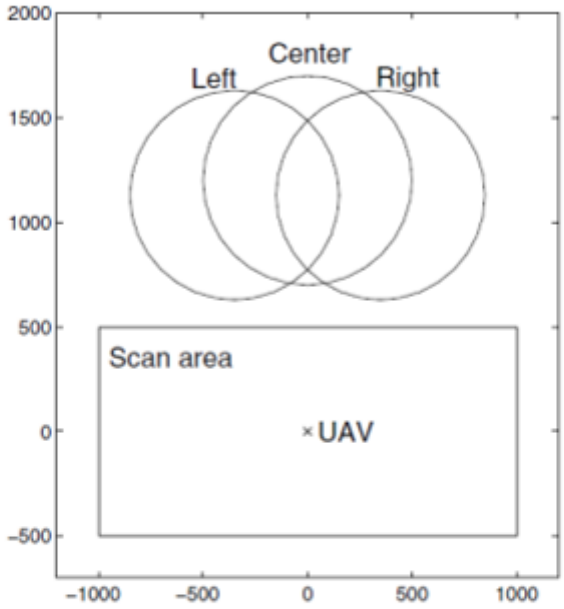



**Figure 2. Pheromone search pattern**

**Table 2. UAV pheromone action table.**

| Probability of action | | |
|---|---|---|
| **Turn left** | **Straight ahead** | **Turn right** |
| (Total – Left) / (2 * Total) | (Total – Center) / (2 * Total) | (Total – Right) / (2 * Total) |

Figure 6.4: Pheromone model

We handled all the cases described in the article, meaning node looks at the values of pheromones in its left diagonal, in front of him and in its right diagonal.

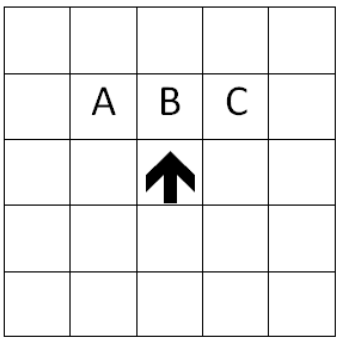



Figure 6.5: case 1

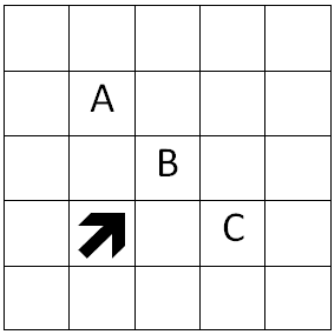



Figure 6.6: case 2

These both cases 6.5 and 6.6 show how a node scan the map.

- the area A represents the left diagonal of the node.
- the area B represents the front of the node.
- the area C represents the right diagonal of the node.

Another characteristic of our implementation is that this time, we don't scan anymore a point of the map but 5 described previously like the picture 6.7. We do that to correspond better to the reality. An UAV scan an area and doesn't scan just a point. So if the UAV choose to scan the point (X,Y), it will move to it and will scan also each point around this position. This method permits us to have results more rapidly.

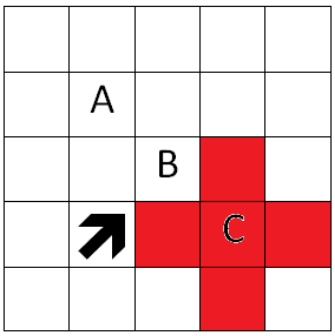



Figure 6.7: scan of area C

We handle the communication between each node with the *MessageListener* and the method *onMessage*. This method is called when a node do the method *send*. We have no destination in the method *send* to broadcast the message. We send message all ten top of clock to respect the conditions of the scenario of the article.

## 6.2 Difficulties Encountered & Solutions

La plus grosse difficultés rencontrées a été la lecture des nombreux articles en correspondance avec notre article d'étude. Ils comportaient de nombreuses pages et la lecture de

ceux-ci en Anglais nous a prit énormément de temps. Nous avons dû par la suite mettre à jour les articles lus en recherchant sur internet les mises à jour.

Concerning the implementation, we did not meet many difficulties. The only big problem is the display of the scanned zones. On the advice of Mister Casteigts, we had to create a class inheriting from JTopology and redefine the method *paint*. Indeed, the display refreshment of the scanned zones is a problem because we lost zones scanned previously. We created an ArrayList to save area scanned to redraw them in every times.

# Chapter 7

# Results

## 7.1 The article

### 7.1.1 Scan Coverage

The goal of the UAVs is to scan the full map once per hour. In theory, with a speed of 150 km/h, the maximum scan speed is 0.083 $km^2$/second/UAV. So with an area of 900 $km^2$, the fastest time to cover the full area is 18 minutes. The figure 7.1 and the figure 7.2 represent the coverage data from the mobility models.

Figure 7.1: Random mobility coverage

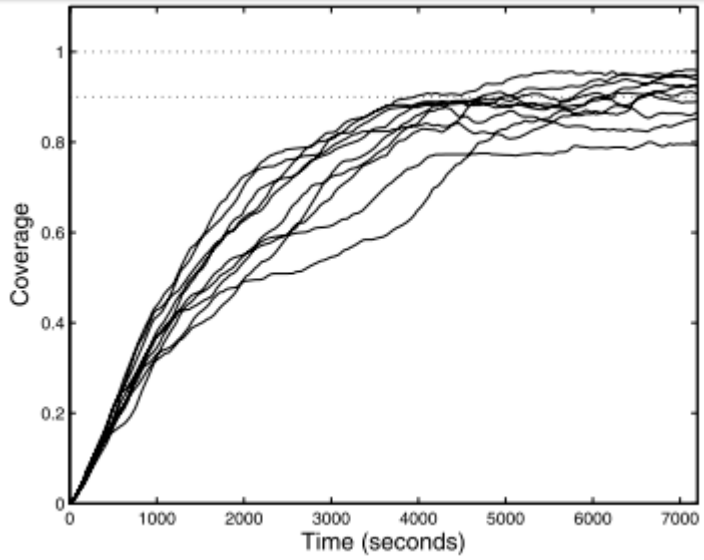
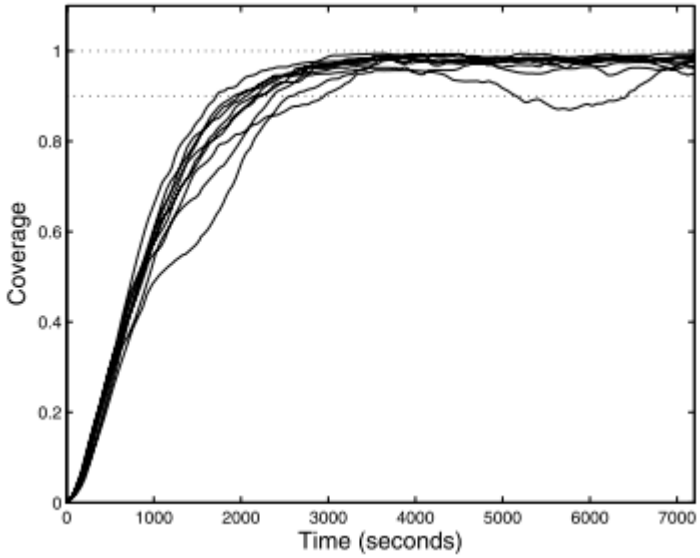


Figure 7.2: Pheromone mobility coverage




For the random model, we see that it need more than 83 minutes to reach 80 percents of coverage. Whereas the pheromone model reach the 90 percents in 60 minutes. Comparing the two models, the pheromone model has a much higher coverage rate.

### 7.1.2 Scan Characteristic

In Figure 7.3 and Figure 7.4, the solid lines represent the probability distribution of each models. This curve permits to calculate the probability of the next scan.
To know the probability of the next scan between the time t1 and t2, we have to calculate the area under the curve between t1 and t2.
So to meet with the requirement of one scan of the map per hour, the curve must be 0 at 1 hour. However none models reach 0 before one hour, because no model manages to achieve full coverage, the pheromone model manages well to avoid rescanning a recently scanned area. We can see that on the graphic : the median curve follows the dashed line at time between 0 seconds and 1000 seconds.

Figure 7.3: Random mobility

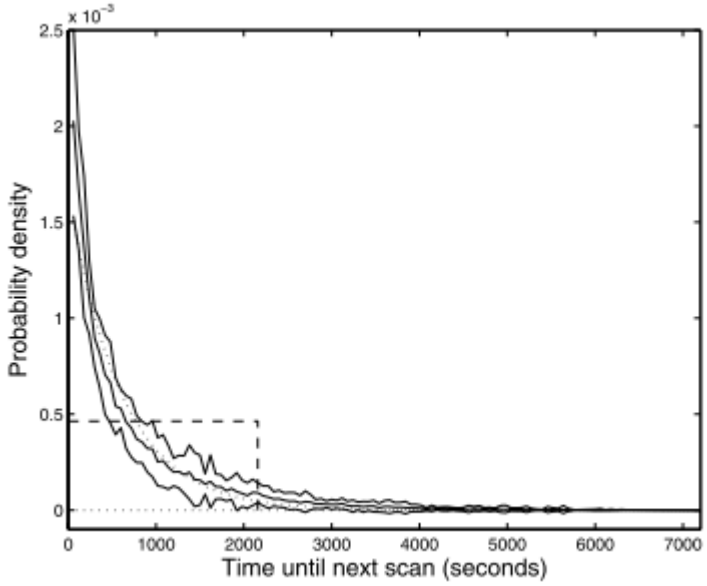
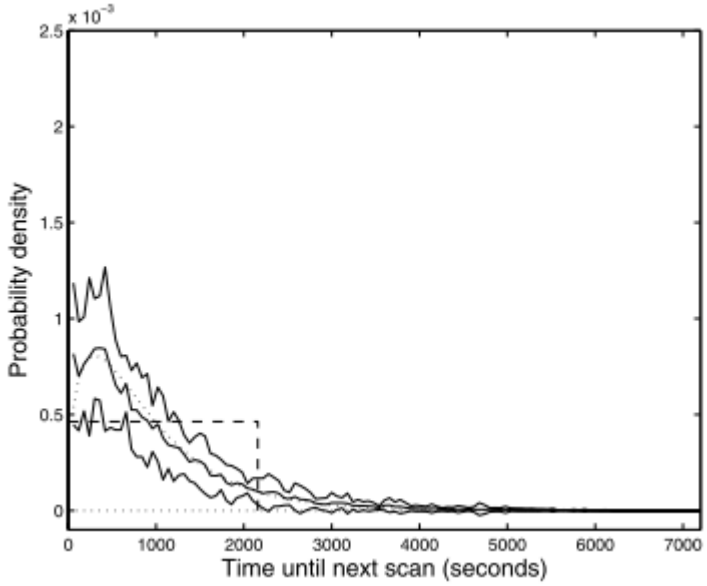


Figure 7.4: Pheromone mobility




### 7.1.3 Never scanned area

In this tableau below, we can see the results about the percentage of the map never scanned.

| | Max | Median | Min |
|---|---|---|---|
| Random | 16.2% | 3.2% | 0.5% |
| Pheromone | 0.21% | 0.03% | 0.01% |

It represents the maximum, median and minimum uncovered area for the ten runs. These numbers clearly show the ability of the pheromone model to cover the complete area.

### 7.1.4 Connectivity

Figure 7.5: Random. Number of UAVs in contact with C&C (max, average, min)

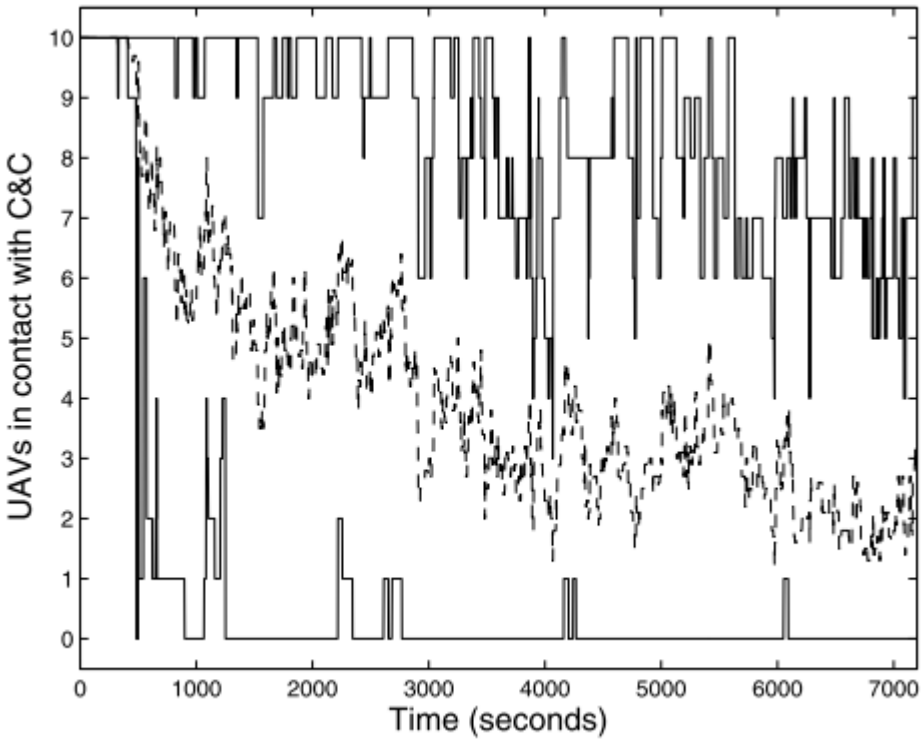
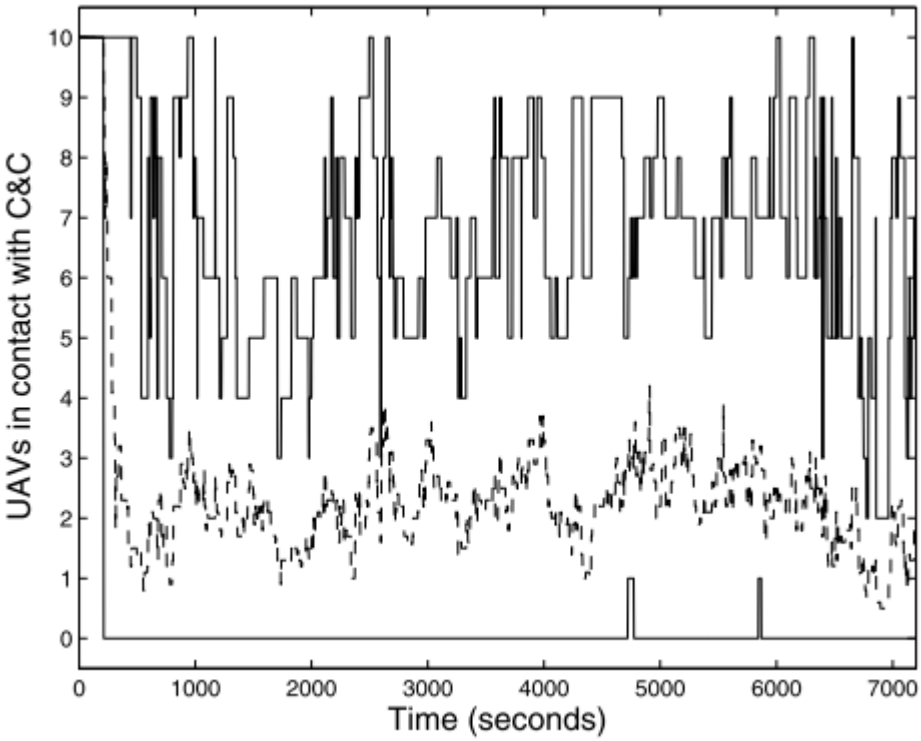


Figure 7.6: Pheromone. Number of UAVs in contact with C&C (max, average, min)




In Figure 7.5 and Figure 7.6,we can see the number of UAV wich can communicate with the C&C. The graphs show the maximum, average, and minimum of this number for the ten runs. We saw that both models don't provide good connectivity. There are not enough UAV to have a fully connected communication graph. Because of the pheromone logic that pushes the UAVs away from each other, this model have low connectivity. Whereas the random model have a high connectivity at the beginning of the run. Then with the random trajectory of each UAV, they move away from each others and finally this model finishes to have the same poor connectivity as the pheromone model.

# Chapter 8

# Tests

# Chapter 9

# Improvements

## 9.1 Mobility models

- In this article, they use The pheromone repel model wich had a bad connectivity. To improve the connectivity, we can add attractive pheromone. At regulary time, the C&C can add attractive pheromone in its place to force the UAVs to move to it in order to have a regulary update of the full map that the C&C gets.
- An other way to improve this model is to add more C&C on the area. Therefore each UAV has a map more often update.
- We can also split the map in equivalent part considering the number of drones. We give a different piece of an area to each UAV. To do that we can add attractive pheromone in these area. So each UAV scans their area and goes to the C&C to transmit his map.
- We can increase highly the connectivity adding others small UAV wich could maintain the communication with the C&C. These small UAV can make the communication path between an UAV and the C&C. Be aware of placing the small drones above the standard drones to avoid collisions.

# Chapter 10

# Development Environment and Conventions

In this part, we will see the development environment we used, and the programming conventions we've applied.

## 10.1 Development Environment

### 10.1.1 github

To write this report and to be able to implement both models most effectively possible, we used a tool of hosting and management of program which is called Github. It is a tools of versionning which allows the collaborative work and allows a simplification of the methods of work.
We used it throughout the project to put the summaries of articles that we read, the tasks which we had to make, this report or still the implementation.
Our Git repository is located at the following address :
`https://github.com/jetcheve/M2_PER_MMUAVGRA`.

Our deposit decomposes into three branches :

- A master branch which is general to the whole deposit.
- A branch called Develop. It is in this branch that there will be all the code for implementing different models. It also contains a test folder.
- An branch report, which as its name suggests, contain all the parts of our report and the pictures have inside it.

### 10.1.2 Eclipse

Eclipse is an integrated development environment for creating projects in any programming language. It is used to develop programs in JAVA, Android, C, etc.

We used Eclipse throughout the project implementation.

As a supplement to Eclipse, we used a library that is called JBotsim [1] proposed by Mr. Casteigts. This allows to simulate a dynamic network. This network is simulated with mobile nodes that interact. These nodes can send messages (in our case, maps of each UAVs).

We used this library because it is very easy to use, and it adapt well to the needs which we had.

## 10.2 Programming Conventions

First, we've choosed to used the Java Coding Conventions, which we can see in the following link `http://www.oracle.com/technetwork/java/codeconventions-150003.pdf`

Then, in order to create our documentation, we've used the Doxygen Convention, viewable here `http://www.stack.nl/~dimitri/doxygen/manual/docblocks.html`
We've relied heavily on this documentation and have mostly employed these protocoles below.

Doxygen Convention for classes

```
/**
* @class name of the class
* @brief Description of herself
*/
```

Doxygen Convention for methods

```
/**
* @brief Description of the method
* @param the parameters and their descriptions
* @return the description of what return the method (optional)
*/
```

Doxygen Convention for members

```
int var; /**< Detailed description after the member */
```

We've also resorted to programming conventions that we defined between us.
For example, when a part of a code, was not finished yet, we put the following lines above the concerning part.

Programming convention for unfinished code

```
/**
* @TO_DO
* Description
*/
```

We've also used a convention for the bugs found and wrote these protocoles, depending on whether the bug was resolved or not.
We used it, in line with the Bug Tracking of GitHub.

Programming convention for bugs

```
/**
* @BUG
* @Unfinished/finished
* Description
*/
```

To finish, we've created the five essential files for a project :

- INSTALL.txt : Installation instructions for the project,
- LICENCE.txt : Licence and copyright © of the project,
- README.txt : General description of the project,
- AUTHORS.txt : Authors of the project,
- MANIFEST.txt : Tree structure and files list of the project.

# Chapter 11

# Conclusion

This article shows that we can't have a good coverage with an adequate communication connectivity. We must find the balance between this two points to have the best model. The random model is simple but it has low connectivity with a bad coverage of area. The best model presented on the article is the pheromone model. It has a good scanning properties thanks to its pheromone repel. But like the random model, it has a low connectivity. A good connectivity is essential to collect data in a more efficient way.

We have seen that the pheromone model isn't perfect and however it has ideal characteristique that are not possible in real condition. Like the communication's range of each UAV is impossible in real life. They can't have a range of 8 km with illimited bandwidth. They have no lost packets, no contention, no overloads. With all these constraints, the connectivity of the pheromone model will be lower than those in article.
Both models can't be used to real scenario of reconnaissance.

# Bibliography


[1] Arnaud Casteigts. The JBotSim library. *CoRR*, abs/1001.1435, 2013.

[2] E. Kuiper and S. Nadjm-Tehrani. *Mobility Models for UAV Group Reconnaissance Applications*. Wireless and Mobile Communications, 2006. ICWMC '06. International Conference on, Bucharest, July 2006.

[3] Mirco Musolesi. An ad hoc mobility model founded on social network theory. In *In Proceedings of the 7th ACM international symposium on Modeling, analysis and simulation of wireless and mobile systems*, pages 20–24. ACM Press, 2004.

[4] Van Dyke Parunak. "go to the ant": Engineering principles from natural multi-agent systems, 1997.

[5] Amit Kumar Saha and David B. Johnson. Modeling mobility for vehicular ad hoc networks, Oct 2004.

[6] Tracy Camp, Jeff Boleng, and Vanessa Davies. A survey of mobility models for ad hoc network research. *WIRELESS COMMUNICATIONS AND MOBILE COMPUTING (WCMC): SPECIAL ISSUE ON MOBILE AD HOC NETWORKING: RESEARCH, TRENDS AND APPLICATIONS*, 2:483–502, 2002.

[7] Amit Jardosh, Elizabeth M. Belding-Royer, Kevin C. Almeroth, Subhash Suri, and Amit Jardosh Elizabeth M. Belding-royer. Towards realistic mobility models for mobile ad hoc networks. pages 217–229, 2003.

[8] Uichin Lee, Eugenio Magistretti, Biao Zhou, Mario Gerla, Paolo Bellavista, and Antonio Corradi. Efficient data harvesting in mobile sensor platforms. In *In PERCOMW '06*, page 352. IEEE Computer Society, 2006.

[9] John A. Sauter, Robert Matthews, H. Van, Dyke Parunak, and Sven A. Brueckner. Performance of digital pheromones for swarming vehicle control. In *In Proceedings of Fourth International Joint Conference on Autonomous Agents and Multi-Agent Systems*, pages 903–910. ACM Press, 2005.

[10] Biao Zhou, Kaixin Xu, and Mario Gerla. Group and swarm mobility models for ad hoc network scenarios using virtual tracks. In *In Proceedings of MILCOM*, pages 289–294. IEEE, 2004.