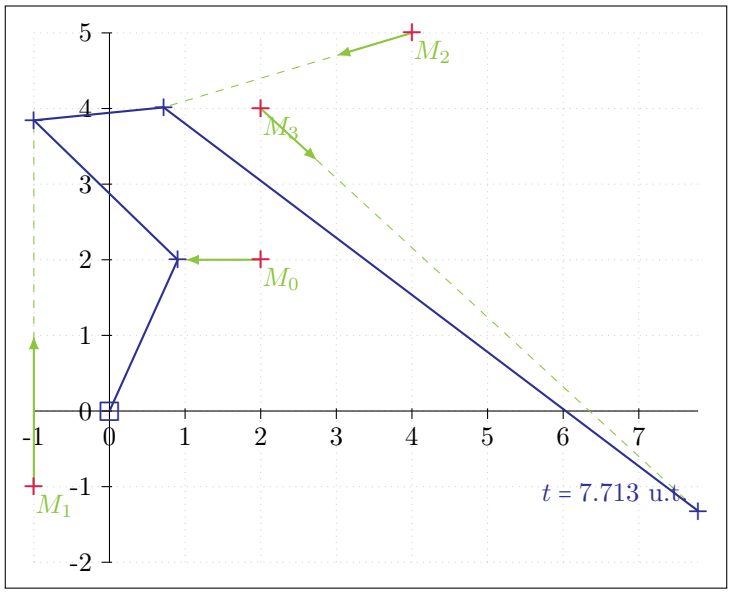
ISIMA Première Année

Projet

Interception de mobiles



Axel DELSOL
Pierre-Loup PISSAVY

Tuteur de projet :
Christophe DUHAMEL

mars – juin 2015

ISIMA

Table des matières

Table des figures

Liste des tableaux

Liste des fichiers

1 | Introduction

1.1 Problème à modéliser

Nous disposons d'un intercepteur dont la vitesse v_1 est constante et de n mobiles.

Il nous faut intercepter autant de mobiles que possible en un temps minimal.

1.2 Méthodes utilisées

1.2.1 Notations

Chaque mobile se déplace à vitesse constante $\left\|\overrightarrow{v_0}\right\|$ selon une direction déterminée par son vecteur vitesse $\overrightarrow{v_0}$.

On peut décomposer cette vitesse selon les axes du plan :

$$\overrightarrow{v_0} = \begin{pmatrix} v_0^x \\ v_0^y \end{pmatrix}$$

On note la position initiale de l'intercepteur :

$$\overrightarrow{i}\,(t=0) = \begin{pmatrix} x_1 \\ y_1 \end{pmatrix}$$

1.2.2 Idées

Une première approche nous mène à supposer qu'il peut être intéressant d'intercepter le mobile que l'on peut atteindre le plus rapidement.

Nous pouvons également tenter d'intercepter les mobiles un par un dans un ordre aléatoire. Cela revient à définir une séquence (aléatoire ou non) et à calculer le temps qu'il sera nécessaire pour les intercepter dans cet ordre (si c'est possible).

Dans tous les cas imaginables, il sera nécessaire de déterminer les positions successives de l'intercepteur au cours du temps ainsi que les différentes directions dans lesquelles il devra se déplacer.

1.3 Outils proposés

Nous avons construit un ensemble de fonctions :

- Calcul de la position d'un mobile à un un instant t,
- Calcul de l'angle que doit prendre l'intercepteur pour intercepter un mobile à partir de sa position courante,
- Calcul de la durée nécessaire pour intercepter un mobile à partir de la position courante de l'intercepteur,
- Calcul de la position future de l'intercepteur à partir de sa position courante, de l'angle, et du temps nécessaire.

Egalement, nous avons conçu une structure de fichier permettant de fournir au programme de calcul toutes les données relatives aux mobiles (position initiale, vitesse, direction) et à l'intercepteur (position initiale, vitesse). Le format de ce fichier lui permet une évolutivité : nous avons laissé la possibilité de définir plusieurs intercepteurs pour des modifications ultérieures du calcul de parcours en impliquant plusieurs.

Un exemple de fichier est présenté en fichier 1.1. Chaque ligne commençant par un croisillon (#) est un commentaire qui ne sera pas interprété par le programme de calcul.

Fichier 1.1 exemple_graph.data

```
#nb depots
1
#nb mobiles
5
#nb intercepteurs / vitesse
1 0.5
#coord intercepteurs
0 0
#coord mobiles / vitesse mobiles
1. 2. 0.5 0.4
2. 4. 0.4 0.2
4. 6. 0.1 0.05
8. -2. 0.7 0.04
-4. 5. 0.2 0.35
```

Nous avons organisé notre code de la manière suivante :

- Un dossier `tests` contenant plusieurs sous-dossiers, chacun correspondant à l'un des 4 tests proposés ici,
- Un dossier `rapport` contenant toutes les sources du présent rapport,
- Un dossier `src` contenant tous les fichiers sources,
- Un dossier `include` contenant toutes les déclarations de fonctions et de structures utilisées,
- Un dossier `bin` dans lequel sera rangé l'exécutable final.

2 | Problèmes étudiés

2.1 Calcul d'interception

On modélise le déplacement de l'intercepteur par la fonction suivante :

$$\overrightarrow{i}(t,\alpha) = \begin{pmatrix} x_1 + t \cdot v_1 \cdot \cos(\alpha) \\ y_1 + t \cdot v_1 \cdot \sin(\alpha) \end{pmatrix}$$

avec $t \in \mathbb{R}^+$ et $\alpha \in [-\pi\,;\pi]$.

On modélise de même la même manière le déplacement du mobile :

$$\overrightarrow{m}(t) = \begin{pmatrix} x_0 + t \cdot v_0^x \\ y_0 + t \cdot v_0^y \end{pmatrix}$$

avec $t \in \mathbb{R}^+$.

On doit donc résoudre le système d'équations suivant afin de calculer le temps d'interception d'un mobile :

$$\begin{cases} x_1 + t \cdot v_1 \cdot \cos(\alpha) &= x_0 + t \cdot v_0^x \\ y_1 + t \cdot v_1 \cdot \sin(\alpha) &= y_0 + t \cdot v_0^y \end{cases}$$

La valeur est donnée par la résolution de l'équation $a \cdot \cos(\alpha) + b \cdot \sin(\alpha) = c$ avec :

$$\begin{cases} a &= y_0 - y_1 \\ b &= x_1 - x_0 \\ c &= \dfrac{a \cdot v_0^x + b \cdot v_0^y}{v_1} \end{cases}$$

On obtient alors 2 possibilités pour la date t :

$$\frac{-b}{-v_0^x + v_1 \cdot \cos(\alpha)} \qquad \text{et} \qquad \frac{a}{-v_0^y + v_1 \cdot \sin(\alpha)}$$

La fonction d'interception teste alors les positions obtenues avec les deux dates et retient celle qui fonctionne et qui est minimale.

2.2 Heuristique H_0

Nous proposons une première méthode heuristique H_0 qui intercepte successivement les mobiles qui seront interceptés le plus rapidement. A chaque nouvelle interception, les temps d'interception pour atteindre les mobiles restants sont recalculés, et l'on conserve le plus faible d'entre-eux.

Ainsi le premier mobile intercepté n'est pas nécessairement le plus proche. En effet, il suffit qu'il s'éloigne de la position initiale de l'intercepteur tandis qu'un autre, plus éloigné au départ s'en rapproche suffisamment vite pour que l'intercepteur commence par intercepter ce dernier.

A chaque étape de recherche du temps d'interception minimal, nous conservons les paramètres d'orientation de l'intercepteur et le temps calculé, cela permet ainsi de limiter le nombre de calculs.

2.3 Heuristique H_1

La seconde méthode heuristique que nous proposons permet, à partir d'une séquence donnée définissant l'ordre dans lequel les mobiles doivent être interceptés, de calculer le temps qui sera nécessaire pour intercepter tous les mobiles qui sont accessibles tout en respectant l'ordre demandé.

Cette méthode demande moins de ressources de calcul que la précédente dans la mesure où l'ordre est défini par avance : il n'y a donc pas à déterminer de temps minimal.

3 | Tests réalisés

Nous présentons dans cette partie un ensemble de tests que nous avons effectués pour contrôler le bon fonctionnement de nos algorithmes.

Nous avons réalisé des graphiques afin d'avoir une meilleure interprétation des résultats obtenus.

Sur ces graphiques, nous avons choisi de représenter les mobiles M_i non-interceptés par des croix vertes (+), et les mobiles M_i interceptés par des croix rouges (+).

Les vecteurs vitesse et la trajectoire empruntée par les mobiles sont indiqués en vert par des vecteurs (⟶) et des lignes pointillées (- - - - -).

La position initiale de l'intercepteur est repérée par un carré bleu (□) et ses positions successives par des croix bleues (+). La date de la dernière interception est indiquée au-dessus de la position où elle a lieu.

3.1 Test n° 1 : Tous les mobiles interceptés, séquences différentes

Fichier 3.1 test_1.data

```
#nb depots
1
#nb mobiles
5
#nb intercepteurs / vitesse
1 2
#coord intercepteurs
0 0
#coord mobiles / vitesse mobiles
1 2 -1 0
0 3 0 -1
2 5 1 0.3
4 1 0.5 0.5
-1 -2 -0.5 0.5
```

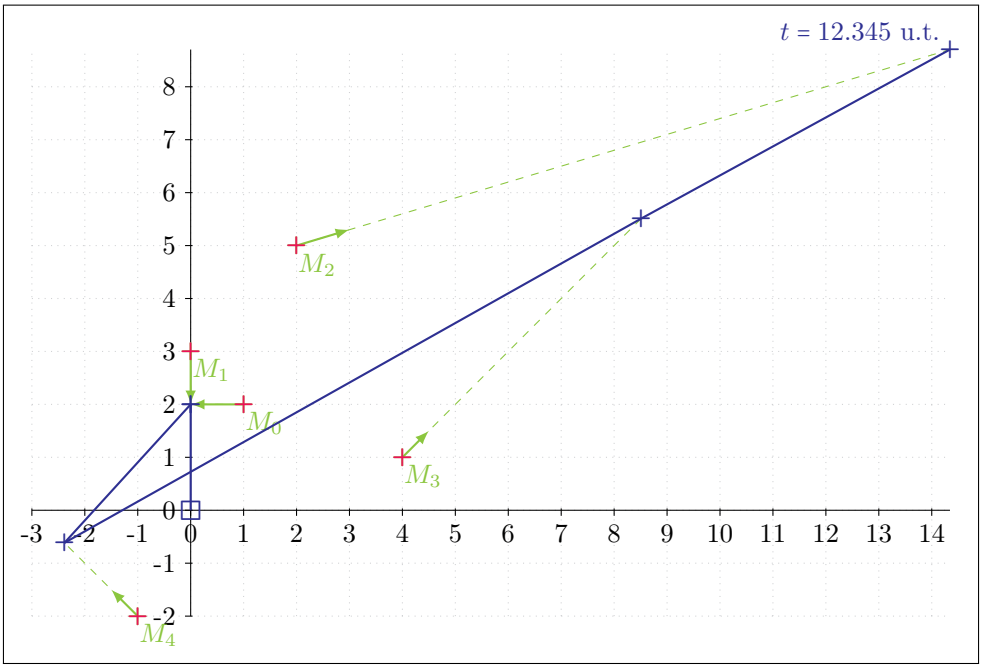


FIGURE 3.1 – Heuristique H_0 : Test n° 1

N° mobile	Position interception	Date interception (u.t.)
0	(0.000 ; 2.000)	1.0000
1	(0.000 ; 2.000)	1.0000
4	(−2.385 ; −0.615)	2.7696
3	(8.509 ; 5.509)	9.0184
2	(14.345 ; 8.703)	12.3446

TABLEAU 3.1 – Heuristique H_0 : Résultats test n° 1

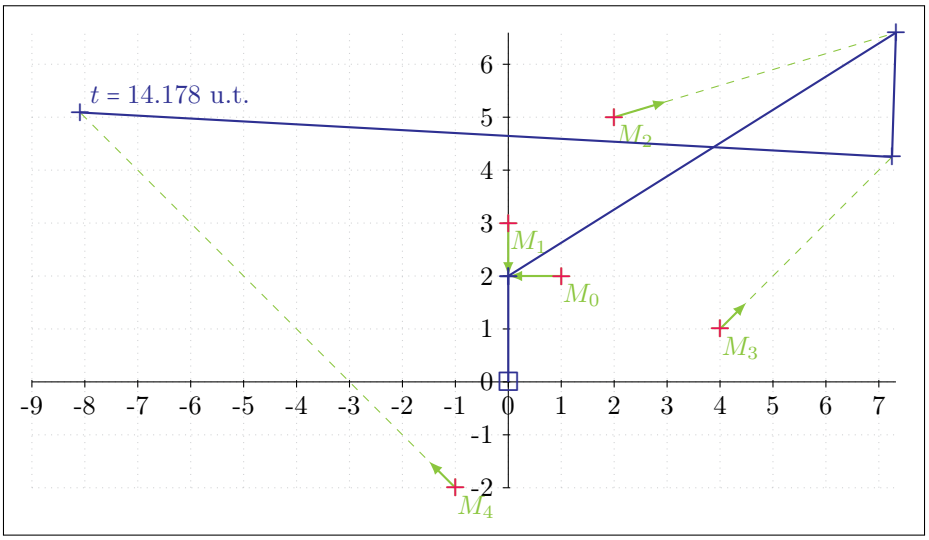


FIGURE 3.2 – Heuristique H_1 : Test n° 1

N° mobile	Position interception	Date interception (u.t.)
0	(0.000 ; 2.000)	1.0000
1	(0.000 ; 2.000)	1.0000
2	(7.323 ; 6.597)	5.3233
3	(7.249 ; 4.249)	6.4979
4	(−8.089 ; 5.089)	14.1785

TABLEAU 3.2 – Heuristique H_1 : Résultats test n° 1

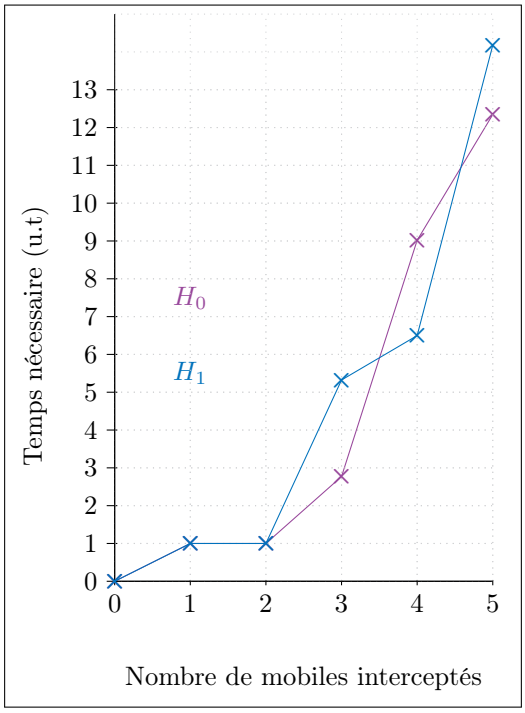


FIGURE 3.3 – Comparaison de H_0 et de H_1 : test n° 1

3.2 Test n° 2 : Résultats identiques et mobile non-intercepté

Fichier 3.2 test_2.data

```
#nb depots
1
#nb mobiles
5
#nb intercepteurs / vitesse
1 2
#coord intercepteurs
0 0
#coord mobiles / vitesse mobiles
2 2 -1.5 0
-1 -1 0 2
4 5 -1 -0.3
2 4 0.6 -0.69
-2 -1 1 6
```

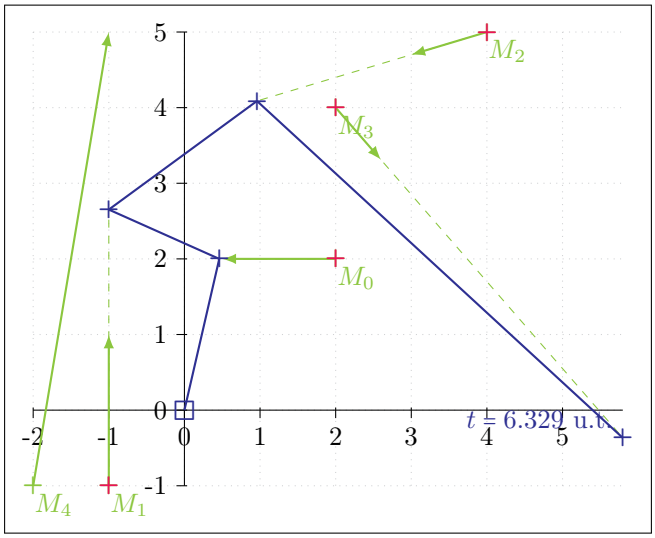


FIGURE 3.4 – Heuristique H_0 : Test n° 2

N° mobile	Position interception	Date interception (u.t.)
0	(0.461 ; 2.000)	1.0262
1	(−1.000 ; 2.652)	1.8260
2	(0.959 ; 4.088)	3.0406
3	(5.797 ; −0.367)	6.3288

TABLEAU 3.3 – Heuristique H_0 : Résultats test n° 2

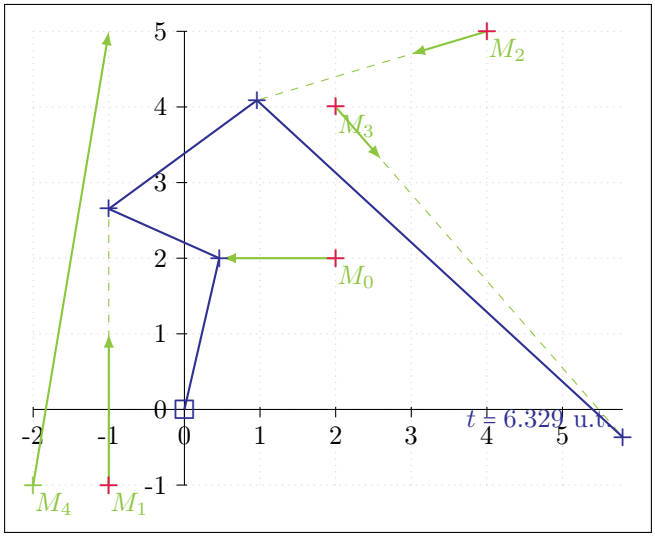


FIGURE 3.5 – Heuristique H_1 : Test n° 2

N° mobile	Position interception	Date interception (u.t.)
0	(0.461 ; 2.000)	1.0262
1	(−1.000 ; 2.652)	1.8260
2	(0.959 ; 4.088)	3.0406
3	(5.797 ; −0.367)	6.3288

TABLEAU 3.4 – Heuristique H_1 : Résultats test n° 2

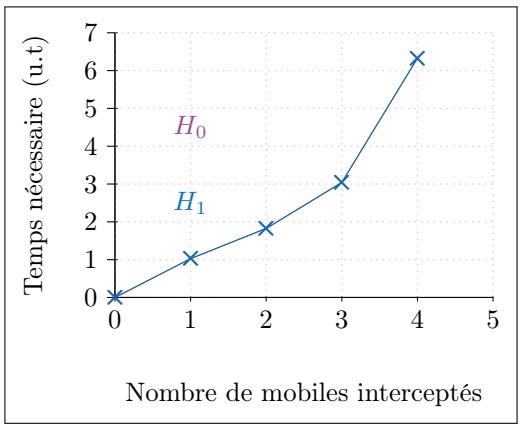


FIGURE 3.6 – Comparaison de H_0 et de H_1 : test n° 2

3.3 Test n° 3 : Mobiles positionnés aléatoirement

Fichier 3.3 test_3.data

```
#nb depots
1
#nb mobiles
6
#nb intercepteurs / vitesse
1 2
#coord intercepteurs
2 9
#coord mobiles / vitesse mobiles
7.140297 8.876400 0 0
4.767767 5.291864 -0.811979 -0.290230
-7.277143 -4.215198 0.458222 0.901331
-9.317039 -8.399123 -0.209148 0.404874
8.360891 5.869070 -0.505128 0.038969
2.371453 -8.866265 -0.225212 0.200970
```

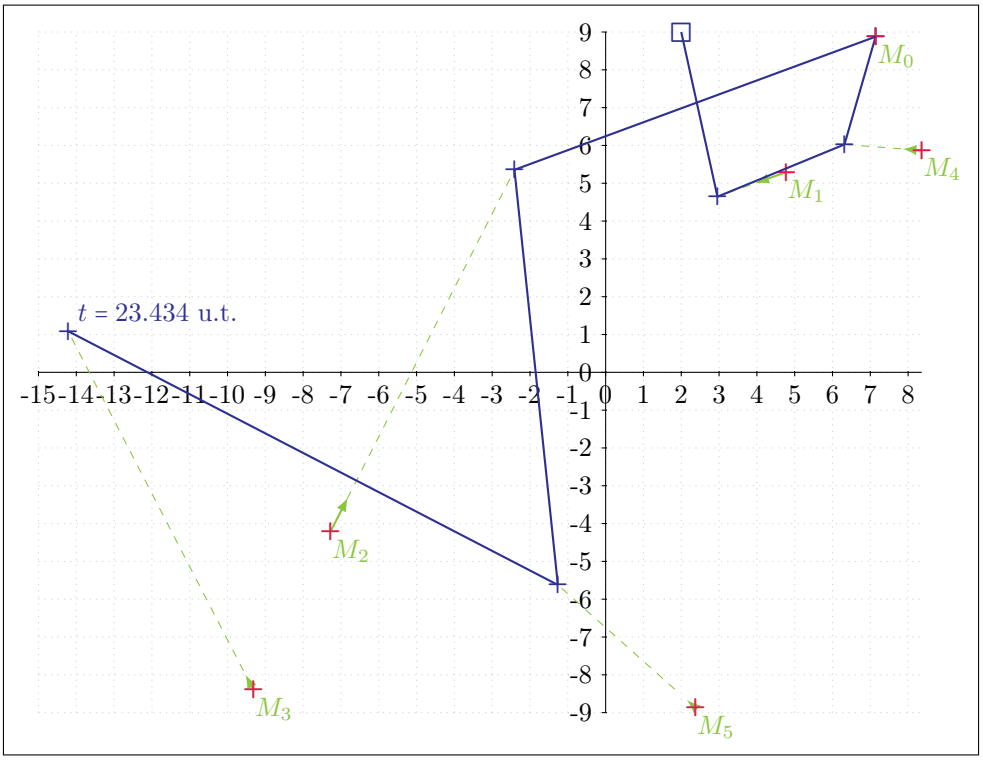


FIGURE 3.7 – Heuristique H_0 : Test n° 3

N° mobile	Position interception	Date interception (u.t.)
1	(2.957 ; 4.645)	2.2296
4	(6.317 ; 6.027)	4.0460
0	(7.140 ; 8.876)	5.5291
2	(−2.411 ; 5.356)	10.6190
5	(−1.263 ; −5.623)	16.1384
3	(−14.218 ; 1.088)	23.4335

TABLEAU 3.5 – Heuristique H_0 : Résultats test n° 3

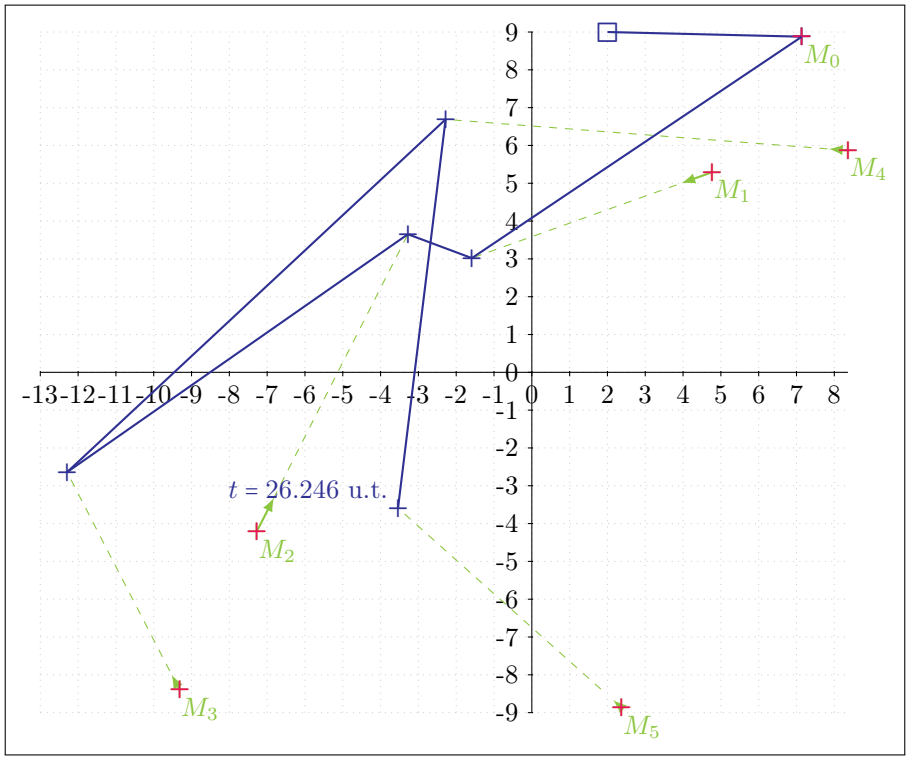


FIGURE 3.8 – Heuristique H_1 : Test n° 3

N° mobile	Position interception	Date interception (u.t.)
0	(7.140 ; 8.876)	2.5709
1	(−1.586 ; 3.021)	7.8255
2	(−3.278 ; 3.652)	8.7281
3	(−12.292 ; −2.640)	14.2246
4	(−2.281 ; 6.690)	21.0671
5	(−3.540 ; −3.592)	26.2462

TABLEAU 3.6 – Heuristique H_1 : Résultats test n° 3

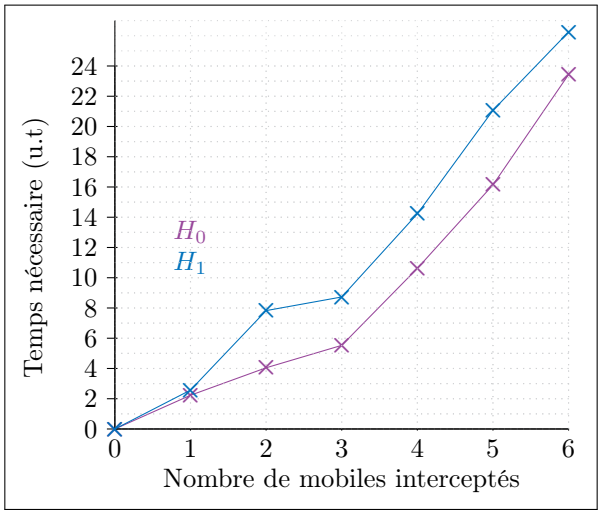


FIGURE 3.9 – Comparaison de H_0 et de H_1 : test n° 3

3.4 Test n° 4 : Heuristique H_1 plus rapide

Fichier 3.4 test_4.data

```
#nb depots
1
#nb mobiles
4
#nb intercepteurs / vitesse
1 2
#coord intercepteurs
0 0
#coord mobiles / vitesse mobiles
2 2 -1 0
-1 -1 0 2
4 5 -1 -0.3
2 4 0.75 -0.69
```

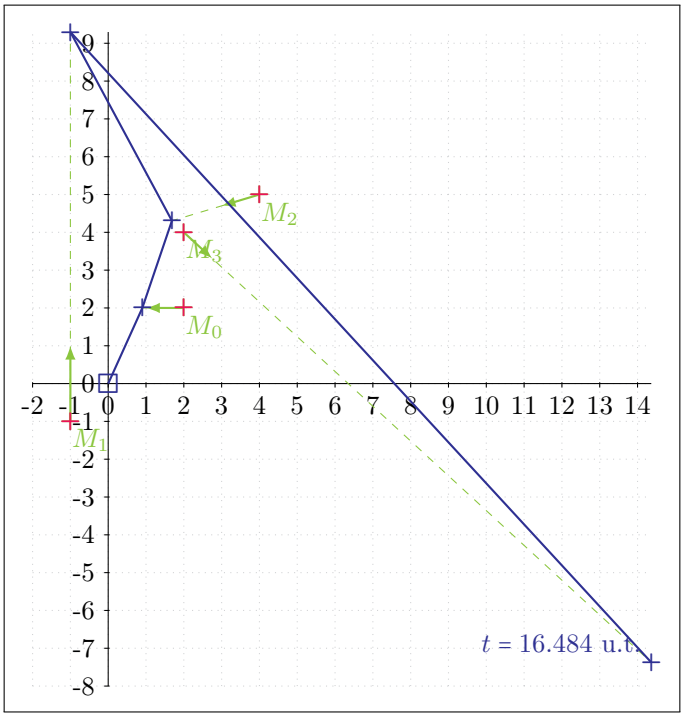


FIGURE 3.10 – Heuristique H_0 : Test n° 4

N° mobile	Position interception	Date interception (u.t.)
0	(0.903 ; 2.000)	1.0972
2	(1.685 ; 4.306)	2.3146
1	(−1.000 ; 9.298)	5.1488
3	(14.363 ; −7.374)	16.4844

TABLEAU 3.7 – Heuristique H_0 : Résultats test n° 4

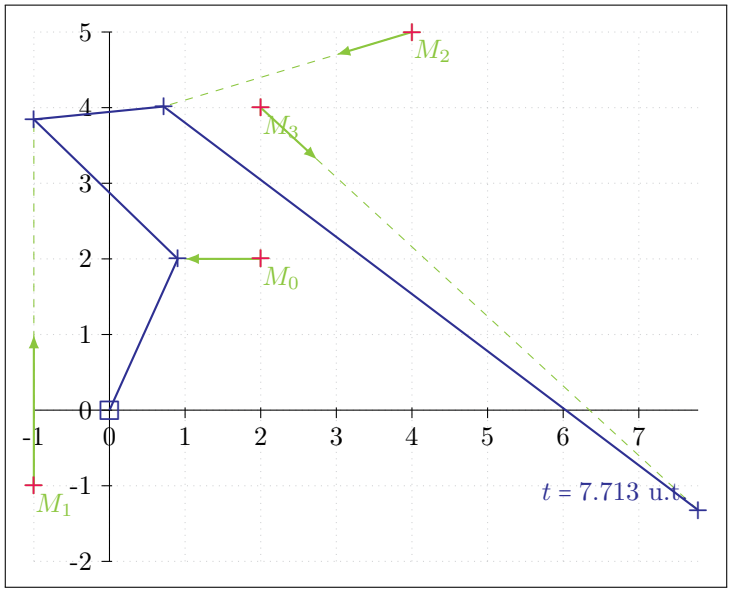


FIGURE 3.11 – Heuristique H_1 : Test n° 4

N° mobile	Position interception	Date interception (u.t.)
0	(0.903 ; 2.000)	1.0972
1	(–1.000 ; 3.844)	2.4221
2	(0.716 ; 4.015)	3.2842
3	(7.785 ; –1.322)	7.7127

TABLEAU 3.8 – Heuristique H_1 : Résultats test n° 4

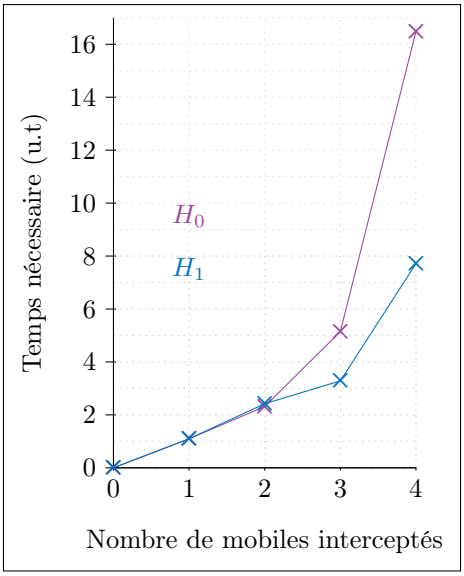


FIGURE 3.12 – Comparaison de H_0 et de H_1 : test n° 4

4 | Conclusion

Le but de ce projet était de maximiser le nombre de mobiles interceptés en minimisant le temps d'interception.

Nous avons donc construit deux heuristiques capables de fournir une solution stable et d'interpréter les résultats.

On remarque que les heuristiques tendent vers des temps d'interceptions similaires lorsque l'on essaye d'intercepter tous les mobiles mais que pour un nombre faible de mobiles, l'heuristique H_0 semble être plus efficace.

Cependant, elle est plus coûteuse en temps CPU que l'heuristique H_1 puisqu'elle demande de recalculer le temps d'interception minimum à chaque itération.

Enfin ces deux solutions ne fournissent pas la solution optimale et donc, il est possible de définir des variantes qui pourraient obtenir de meilleurs résultats.

Proposition d'améliorations

On pourrait considérer que l'intercepteur se déplace à une vitesse variable (majorée par une vitesse max v_1) et chercher à trouver la vitesse v qui nous permettrait d'intercepter un mobile plus rapidement.

Afin de trouver la séquence idéale, minimisant le temps de parcours, on peut partir d'une séquence initiale, lui appliquer des opérations particulières (intervertion de mobiles/ajout/suppression), et comparer les différents parcours possibles pour trouver le meilleur compromis.

On pourrait également concevoir une gestion de priorité au niveau des mobiles (par exemple pour inspecter des zones en priorité avec des drônes).

Il serait ainsi possible de proposer des heuristiques afin d'optimiser un rapport mobiles interceptés / temps de parcours ou bien mobiles / distance parcourue.

Remerciements

Nous tenons à remercier Christophe DUHAMEL qui nous a accompagné dans tout le cheminement de notre projet et qui nous a conseillé quant aux méthodes à employer pour réaliser nos heuristiques.

Egalement, nous souhaitons remercier Luc MARCHAND pour son aide dans la détermination du calcul nécessaire à l'obtention de l'angle α permettant d'orienter l'intercepteur, ainsi que Gilles LEBORGNE qui nous a aidé à déterminer si la solution était facilement résolvable.

Outils utilisés

Pour la rédaction de ce rapport, nous avons utilisé LaTeX avec le module TikZ pour les graphiques, et `minted` pour l'affichage des fichiers.

Le code correspondant aux graphiques est généré par le programme et inclus à la compilation du rapport.

Notre code source est écrit en langage `C` et est disponible sur la plateforme GitHub à l'adresse :

https://github.com/theCmaker/Projet_ZZ1