



# Projet de Master IMA : Reconnaissance et tracking embarqué sur robot mobile thymio-2 équipé d'un Raspberry PI 3.

**RAPPORT DU PROJET**

**BIRABEN-BIANCHI Hugo**

21 mai 2018

Encadrants : Nicolas BREDECHE et Dominique BEREZIAT

## Table des matières

|      |   |    |
|------|---|----|
| I.   | Introduction.....                       | 3  |
| 1.   | Présentation du sujet .....             | 3  |
| 2.   | Présentation de mon travail.....        | 3  |
| 3.   | Pourquoi avoir choisi ce projet ? ..... | 3  |
| II.  | Projet .....                            | 4  |
| 1.   | Objectifs.....                          | 4  |
| 2.   | Matériel et logiciel.....               | 4  |
| 3.   | Etat de l’art .....                     | 5  |
| 4.   | Avancée du projet .....                 | 6  |
| 5.   | Méthodologie .....                      | 7  |
| 6.   | Améliorations possibles.....            | 10 |
| III. | Bibliographie.....                      | 11 |
| IV.  | Annexes .....                           | 11 |

# I. Introduction

## 1. Présentation du sujet

Le projet consiste en la réalisation d'une solution de détection de tags posés sur des robots thymios-2, avec pour unité de calcul un Raspberry Pi 3 et comme module d'acquisition une PiCamera. Il faut pouvoir détecter en temps réel les tags afin d'identifier les robots dans le champ de vision.

Le but étant de pouvoir créer une communication entre les robots dans le champ de vision, mais également des interactions (suivre, éviter ...). Cette partie-là est mise en place par un groupe d'ANDROIDE, ce qui fait qu'il y a une collaboration entre différents groupes. Il y a également une partie de monitoring des robots dans leurs déplacements et actions réalisée par Bailin CAI, un autre étudiant en M1 IMA.

## 2. Présentation de mon travail

Le sujet que je dois traiter n'est pas nouveau, en effet, plusieurs étudiants ont pu déjà travailler dessus les années précédentes. Il existe donc déjà une solution, mais qui n'est pas encore assez performante pour être utilisée en situation. Je dois donc améliorer la solution existante, mais également faire des ajouts de nouvelles fonctionnalités afin de permettre l'utilisation de la solution par des étudiants en P-ANDROIDE, ce qui est l'objectif final de mon projet.

Je m'occupe également de produire de la documentation sur mon projet, mais également sur le fonctionnement global de l'environnement de travail en salle de robotique, pour permettre aux autres utilisateurs une meilleure expérience d'utilisation et un travail plus productif.

## 3. Pourquoi avoir choisi ce projet ?

J'ai choisi ce projet car ce qui me plaît dans l'informatique, mais également dans l'imagerie, c'est l'applicatif, c'est-à-dire produire un code utilisant des outils déjà existants afin d'avoir une solution fonctionnelle avec un retour concret de la progression dans le travail. L'embarqué me semblait donc être un bon pas dans cette voie. De plus, cela permet également d'apprendre beaucoup, grâce à l'utilisation de bibliothèques, indispensable pour contrôler le matériel à notre disposition, mais aussi sur le moyen de mettre en place des solutions avec des ressources bien plus limitées qu'en programmation plus traditionnelle, comme ce que l'on trouve habituellement en imagerie.

## II. Projet

### 1. Objectifs

L'objectif principal de ce projet est de pouvoir offrir une reconnaissance performante et suffisamment robuste. Cela permettra aux étudiants de P-ANDROIDE de pouvoir détecter la face du robot dans le champ de vision, ainsi que sa distance pour adapter son comportement (suivre, éviter, se rapprocher ...). Cela donnera également la possibilité de pouvoir procéder à l'envoi de données entre les robots, pour mettre à jour leurs algorithmes de prise de décision lorsqu'ils seront dans le champ de vision.

Parmi les autres objectifs fixés dans le projet, il y a la production de documentation, afin de permettre à d'autres personnes de comprendre le fonctionnement du projet, mais également l'environnement de travail mis en place dans la salle de robotique. En effet, je me suis rendu compte qu'il était difficile de travailler efficacement sur le projet avec une documentation inexistante.

### 2. Matériel et logiciel

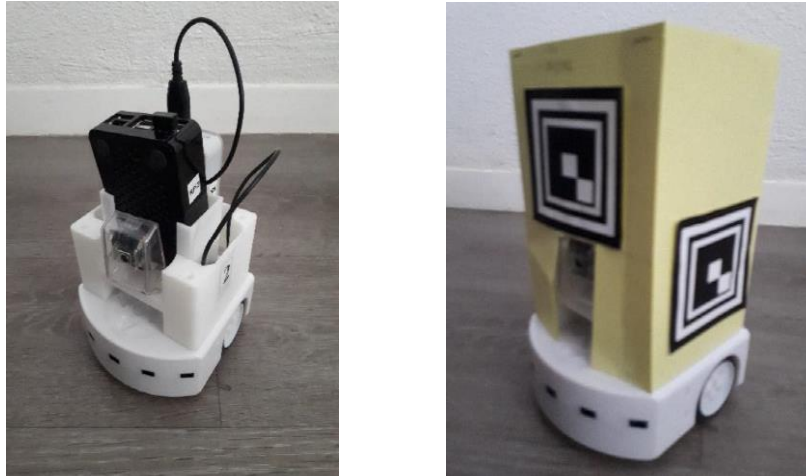


**Figure 1** – Matériel utilisé dans le projet

Le matériel constituant le cœur de ce projet a été sélectionné puisqu'il permettait d'avoir un robot complet pour un coût faible, ainsi nous disposons :

- D'une partie mobile avec le Thymio-2, qui peut se déplacer dans toutes les directions, avec 3 vitesses différentes,
- D'une partie de calcul avec le Raspberry-pi 3, sur lequel sont stockés mon algorithme d'imagerie, mais également les algos de prise de décision des P-ANDROIDE,
- D'une partie acquisition d'image, qui se fait grâce au module caméra v2 du Raspberry-pi 3, branché au port CSI de ce dernier.

Afin de pouvoir placer le Raspberry-pi 3, son module camera ainsi qu'une batterie externe pour pouvoir alimenter le tout, un module imprimé grâce à une imprimante 3D supporte le tout. Le tout est surmonté d'un carton sur lequel sont collés les tags afin de permettre la reconnaissance des robots.



**Figure 2** – A gauche, le robot avec son équipement complet, à droite le robot complet surmonté d'un carton afin de placer les tags de reconnaissance dessus.

Pour la partie logicielle, le projet est bien entendu constitué des API de la PiCamera, fourni par le constructeur. Cela permet de l'acquisition vidéo, et tous les paramètres de la PiCamera. Il est possible de choisir la résolution, le framerate, mais également la luminosité ou bien encore une acquisition couleur ou en niveau de gris par exemple. A cela s'ajoute la librairie Open CV en version 3.4.1 (dernière version à ce jour) qui est un outil essentiel pour le traitement d'image. Elle fournit les outils nécessaire et indispensable pour faire de la détection de contour, des opérations sur les images de manière rapide et avec une bonne robustesse.

### 3. Etat de l'art

La détection d'objet dans une image a pendant longtemps était un obstacle, il était difficile d'avoir de bons résultats, que ce soit en temps réel ou non. Si aujourd'hui on trouve un grand nombre de solutions faciles d'utilisation (*SIFT*, *SURF*, *FAST* [4]) et offrant de bonnes performances, les recherches dans le domaine continuent d'avancer, avec des modèles toujours plus précis et rapides.

Ainsi, si au début les solutions utilisées des descripteurs basés sur les histogrammes de couleur pour réaliser une indexation [Swain91] ou encore des invariants géométriques [Rothwell92], on a trouvé depuis de nouvelles méthodes. Avec l'apparition et la démocratisation du deep-learning et des réseaux de neurones, des algorithmes comme **You Only Look Once v3** mélange détection d'objet traditionnelle et ces nouvelles technologies afin d'accélérer la détection en temps réel.

J'ai donc dû reprendre le code de Mario VITI, réalisé l'année précédente, utilisant un détecteur de contour. Il y a ensuite une rectification de la perspective puis ensuite la détection et l'identification des tags.

#### 4. Avancée du projet

Le début du projet a été plutôt complexe, avec la découverte de l'environnement déjà existant ainsi que les problèmes matériels et logiciels liés à l'embarqué, il m'a fallu un certain temps avant de pouvoir produire quelque chose. En effet, je me rends tous les lundis après-midi en salle expérimentale de robotique à l'ISIR (tour 65-66 étage 3) pour travailler et faire un point hebdomadaire avec mes deux encadrants sur l'avancement ainsi que les problèmes rencontrés.

Ainsi, les deux premières semaines, j'ai pris connaissance du matériel constituant le robot (Raspberry-pi 3, PiCamera, Thymio-2) mais aussi du matériel à ma disposition en salle de robotique. En effet, les Raspberry-pi ne sont pas connectés à internet par mesure de sécurité, on y accède donc à l'aide d'un routeur branché à l'ordinateur mis à notre disposition. Cependant, la configuration particulière mise en place (servant pour un programme de communication des P-ANDROIDE permettant de faire un broadcast des ordres à tous les robots connectés au routeur) ne fonctionnait pas, empêchant le travail sur les Raspberry-pi.

Par la suite, il m'a fallu prendre connaissance du code correspondant à mon projet et qui existait déjà, réalisé l'année dernière par Mario VITI, un étudiant en IMA. La solution qu'il avait réalisée ne fonctionnait plus à la suite d'une mise à jour du système. Je l'ai contacté afin de voir avec lui si nous pouvions résoudre le problème, mais également avoir quelques précisions supplémentaires sur le fonctionnement de certaines parties de son code. J'ai pu prendre connaissance des problèmes qu'il avait rencontrés, mais aussi des éléments qui devaient être modifiés ou créés afin d'accomplir l'objectif du projet.

Je me suis ensuite attelé à produire de la documentation technique en anglais, expliquant le fonctionnement de l'environnement de travail dans la salle de robotique (ordinateur mis à notre disposition, routeur, OctoPy et ThymioPYPI) mais également du robot (Raspberry-pi, PiCamera, programme de reconnaissance de tag). J'ai également produit une documentation sur des expérimentations à réaliser afin de voir l'état actuel du programme, ce qu'il était capable ou non de faire (évaluer les distances de reconnaissance, actuellement entre 10 et 80 cm, le nombre de tag détecté ...) afin de comparer maintenant et plus tard les avancées qui seront réalisées. (Cf. [IV. Annexes](#))

J'ai également produit une image de l'OS utilisé sur le Raspberry Pi pour nettoyer la base utilisée et repartir de zéro avec un système plus à jour, pouvant ainsi supporter une plus longue durée de vie et être mieux maintenu.

J'ai alors pu commencer à effectuer des modifications dans le code. Mes premières modifications ont été sur les paramètres d'acquisitions, j'ai cherché quelle était la meilleure configuration pour avoir un bon angle de vision, tout en obtenant une image nette et une fréquence suffisamment élevée pour avoir des informations intéressantes pour les traiter après. Je vais donc détailler dans la prochaine partie les aspects plus techniques de mon travail.

## 5. Méthodologie

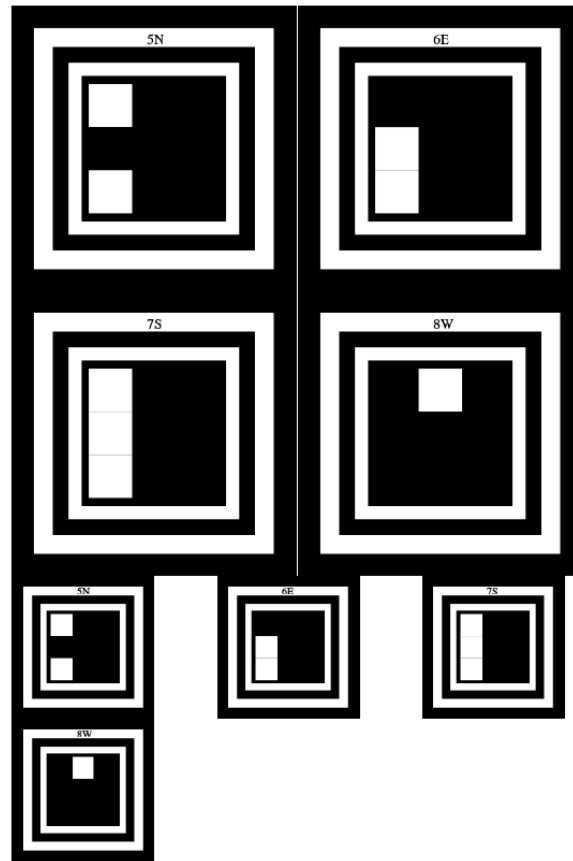
Tout d'abord, il a fallu améliorer la qualité de l'image lors de l'acquisition. Le problème qu'avait rencontré Mario VITI était d'avoir une image avec un petit champ de vision et une image avec du flou lors des mouvements des Thymios. Pour régler le problème du champ de vision, il a fallu changer la fréquence de la vidéo. En effet, diminuer le nombre d'images par seconde modifie l'angle de vision en l'agrandissant. Je n'ai trouvé aucune information sur les raisons de ce comportement, mais j'ai pu obtenir un bien meilleur champ de vision en réglant le nombre d'images par seconde à 20. Avec ce taux de rafraîchissement, j'ai également réussi à supprimer le flou lors du mouvement des robots.

Ensuite, il a également fallu fournir l'orientation du robot, connaître sa direction permettrait à d'autres projets de travailler sur le comportement des robots, comme pourrait le faire un banc de poissons. J'ai donc dû réfléchir à quelle était la solution la plus simple pour obtenir l'orientation d'un robot, tout en sachant qu'une petite précision était amplement suffisante. Ainsi, j'ai opté pour la solution de mettre un tag différent sur chaque face du robot, cela permet donc de connaître son orientation, selon que l'on voit un ou deux tags. L'ordre des tags est ainsi toujours le même, suivant le sens horaire (Nord, Est, Sud et enfin Ouest).

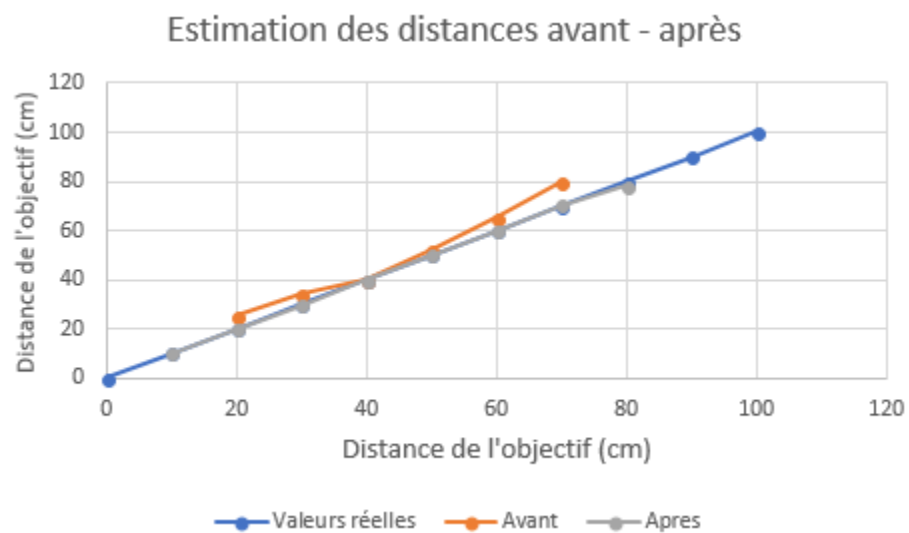
De plus, j'ai ajouté des inscriptions sur les tags afin que n'importe qui sache dans quels sens les poser sur le robot, mais aussi de pouvoir dire si l'algorithme prédit les bons numéros de tag. J'ai également produit des tags en version réduite afin qu'ils puissent être vus de plus prêt (en effet, le tag situé sur la face avant, est placé en hauteur parce qu'il doit laisser l'objectif de la caméra pouvoir filmer les autres tags).



**Figure 3** – Exemple de reconnaissance d'un tag

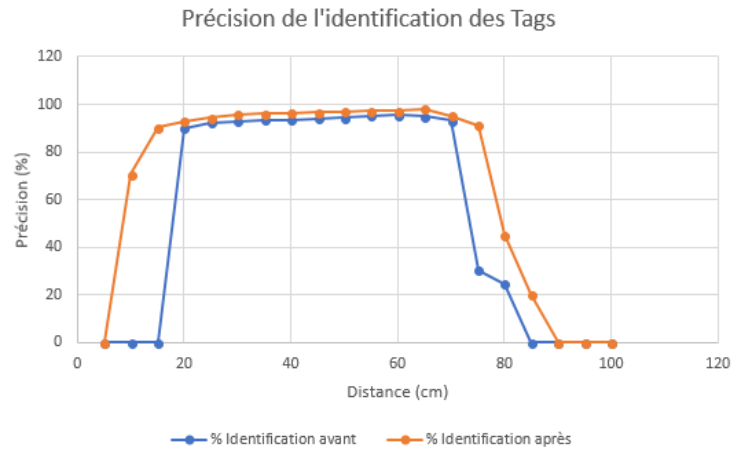


**Figure 4** – Exemple de fichier PDF des Tags pour le robot 2



**Figure 5** – Estimation des distances tel que l'était l'algorithme de Mario et après les modifications que j'ai pu apporter.

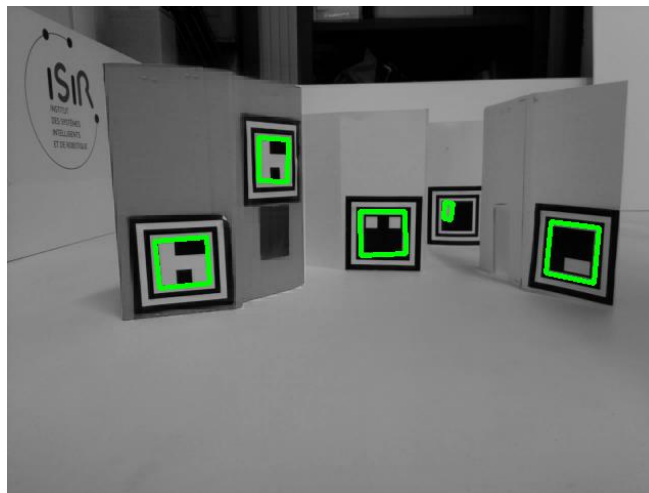




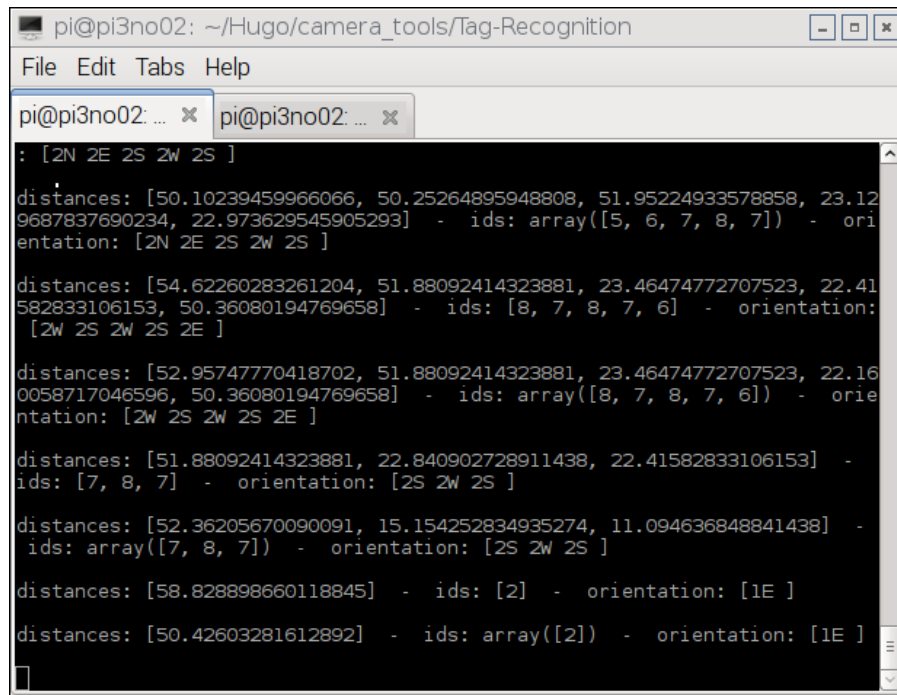
**Figure 6** – Précision de l'identification des Tags selon la distance en cm.



**Figure 7** – Détecteur de contour



**Figure 8** – Exemple en situation (avec un bug sur le 2<sup>ème</sup> tag en partant de la gauche)



```
pi@pi3no02: ~/Hugo/camera_tools/Tag-Recognition
File Edit Tabs Help
pi@pi3no02: ... x pi@pi3no02: ... x
: [2N 2E 2S 2W 2S ]
distances: [50.10239459966066, 50.25264895948808, 51.95224933578858, 23.12
9687837690234, 22.973629545905293] - ids: array([5, 6, 7, 8, 7]) - ori
entation: [2N 2E 2S 2W 2S ]
distances: [54.62260283261204, 51.88092414323881, 23.46474772707523, 22.41
582833106153, 50.36080194769658] - ids: [8, 7, 8, 7, 6] - orientation:
[2W 2S 2W 2S 2E ]
distances: [52.95747770418702, 51.88092414323881, 23.46474772707523, 22.16
0058717046596, 50.36080194769658] - ids: array([8, 7, 8, 7, 6]) - orie
ntation: [2W 2S 2W 2S 2E ]
distances: [51.88092414323881, 22.840902728911438, 22.41582833106153] -
ids: [7, 8, 7] - orientation: [2S 2W 2S ]
distances: [52.36205670090091, 15.154252834935274, 11.094636848841438] -
ids: array([7, 8, 7]) - orientation: [2S 2W 2S ]
distances: [58.828898660118845] - ids: [2] - orientation: [1E ]
distances: [50.42603281612892] - ids: array([2]) - orientation: [1E ]
```

**Figure 9** – Affichage des informations que retourne le programme lorsqu’il détecte des Tags (Figure 8)

## 6. [Améliorations possibles](#)

À la suite de l’absence de documentation sur le projet et aux nombreux problèmes rencontrés lors de la prise en main du projet dans la salle d’expérimentation, je n’ai pas réussi à aller jusqu’au bout de ce projet.

Une modification plus importante du cœur de la détection des Tags aurait été nécessaire, afin de corriger les différents bugs (Clignotement de la detection, absence de détection dans une distance pourtant fonctionnelle ou encore mauvaise reconnaissance des Tags) et améliorer la distance de reconnaissances des Tags.

Il aurait également été une bonne idée de filmer des vidéos et de modifier le programme pour qu’il puisse fonctionner avec ces vidéos pré-enregistrées afin de quantifier plus facilement les améliorations apportées au logiciel et faire des comparaisons entre les différentes versions. Je n’y avais absolument pas pensé, et Nicolas BREDECHE et Dominique BEREZIAT m’en ont parlé trop tard pour que je puisse la mettre en place.

### III. Bibliographie

[1] J. Canny. **A Computational Approach to Edge Detection**. vol. *PAMI-8*, no. 6, pp. 679-698, Nov. 1986.

[2] J.R. Bergen, P. Anandan, K. J. Hanna, R. Hingorani. **Hierarchical model-based motion estimation**. *Computer Vision - ECCV'92* pp. 237-252, May 1992.

[3] N. Bredeche, E. Haasdijk, A. E. Eiben. **On-Line, On-Board Evolution of Robot Controllers**, in *Artificial Evolution: 9th International Conference, Evolution Artificielle, EA, 2009, Strasbourg, France*, pp. 110-121, Oct 2009.

[4] **OpenCV Computer Vision Library**. [Online at <https://docs.opencv.org/3.4.1/>]. [Accessed: 15-March-2018].

[5] **PiCamera Library** [Online at <http://picamera.readthedocs.io/en/release-1.13/index.html>]. [Accessed : 12-March-2018].

### IV. Annexes

Liens vers la documentation produite :

- Documentation du projet et du fonctionnement de l'environnement en salle de robotique.

[https://docs.google.com/document/d/1VPjbkoip\\_LUNSA9O8SOeYe54Wtd8JIACi-L3X1S17dM/edit?usp=sharing](https://docs.google.com/document/d/1VPjbkoip_LUNSA9O8SOeYe54Wtd8JIACi-L3X1S17dM/edit?usp=sharing)

- Documentation sur les expérimentations à réaliser pour effectuer les mesures.

<https://docs.google.com/document/d/1S2avljW17NMtbQheUsmiPhZK3mwUSPpmK9APyRJyqLo/edit?usp=sharing>