République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Ecole Nationale Polytechnique

Département d'Automatique

En collaboration avec l'Ecole Centrale d'Electronique (ECE Paris)

Mémoire de projet de fin d'études pour l'obtention du diplôme d'ingénieur d'état en Automatique

THEME

Conception et Réalisation d'un Robot Mobile Entraîneur des Gardiens de Buts de Football

Melle Stihi Maha

Sous la direction de :

Dr Chakir Messaoud (ENP)

Pr.A Moussaoui Zeryab (ECE)

Mr Casimir Raphaël (ECE)

Présenté et soutenu publiquement le 2 juillet 2019 devant le jury :

Président :	D. BOUDANA	Dr	ENP
Rapporteurs :	M.CHAKIR	Dr	ENP
	Z.MOUSSAOUI	Pr.A	ECE
Examinateurs :	M.S BOUCHERIT	Pr	ENP

ENP 2019

République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Ecole Nationale Polytechnique

Département d'Automatique

En collaboration avec l'Ecole Centrale d'Electronique (ECE Paris)

Mémoire de projet de fin d'études pour l'obtention du diplôme d'ingénieur d'état en Automatique

THEME

Conception et Réalisation d'un Robot Mobile Entraîneur des Gardiens de Buts de Football

Melle Stihi Maha

Sous la direction de :

Dr Chakir Messaoud (ENP)

Pr.A Moussaoui Zeryab (ECE)

Mr Casimir Raphaël (ECE)

Présenté et soutenu publiquement le 2 juillet 2019 devant le jury :

Président :	D. BOUDANA	Dr	ENP
Rapporteurs :	M.CHAKIR	Dr	ENP
	Z.MOUSSAOUI	Pr.A	ECE
Examinateurs :	M.S BOUCHERIT	Pr	ENP

ENP 2019

Dédicaces

Je dédie ce projet de fin d'études en premier, à mes très chers parents. Tous les mots du monde ne sauraient exprimer l'immense gratitude que je vous témoigne pour tous les efforts et les sacrifices que vous n'avez jamais cessé de consentir pour mon instruction et mon bien-être. J'espère avoir répondu aux espoirs que vous avez fondés en moi.

A mon frère Mehdi.

A toute ma grande famille, plus spécialement à ma grand-mère.

A tous mes amis, plus particulièrement à Isra, Assia, Houda, Hanan, Romaissa, Meriem et Abderrahim.

A tous ceux qui m'ont aidé de près ou de loin dans la concrétisation de ce modeste travail.

Remerciements

Tout d'abord, merci à **Dieu**, le Clément, le Miséricordieux de nous avoir donné la force et le courage de mener à bien ce modeste projet.

C'est avec une profonde reconnaissance et considération particulière que je remercie mon promoteur **Pr Moussaoui Zeryab** pour la sollicitude avec laquelle il m'a suivi et guidé tout au long de ce parcours.

Je tiens aussi à remercier **l'Ecole Centrale d'Electronique Paris** pour ce qu'elle m'a fourni en personnel et matériel pour la réussite de mon projet, ainsi que le stage proposé au niveau de son association **Eceborg**

Je ne manquerai pas de remercier vivement mes Co-encadreurs **Dr Chakir Messaoud** et **Mr Casimir Rafael**, qui ont répondu toujours présents à mes interrogations et ont transmis leurs passions et visions du domaine.

Je remercie les membres du jury qui me font l'honneur de présider et examiner ce modeste travail.

Toute ma gratitude à tous les enseignants qui ont contribué à ma formation à **l'Ecole Nationale Polytechnique**.

Mes remercîments vont également à tous ceux qui ont contribué de près ou de loin à l'aboutissement de ce projet.

Tout le personnel de l'Ecole Nationale Polytechnique.

ملخص

يتمحور المشروع حول نمذجة وبناء روبوت مستقل يقوم بتصويب الكرات وتقييم أداء حارس مرمى كرة القدم بناءً على ردود أفعاله.

باستخدام كاميرا ومقدّر الوضع البشري ثلاثي الأبعاد ، يمكن للنظام الكشف عن الإيماءات الجسدية لحارس المرمى ووضع نماذج لها.

أخيرًا، يقترح باستخدام الذكاء الاصطناعي القرار المناسب للتدريبات المستقبلية التي ستحسن من أداء اللاعب.

كلمات مفتاحية : حارس مرمى كرة القدم ، تدريب ، روبوت ، تقدير الوضع البشري ثلاثي الأبعاد ، الذكاء الاصطناعي.

Abstract

The project consists on the modeling and the realization of an autonomous robot that shoots and evaluates the performances of a soccer goalkeeper based on his reactions.

Using a camera and a 3D human pose estimator, the system can detect and modelize the goalkeeper's physical gestures.

Finally, it proposes using artificial intelligence the suitable decision for the future exercises, which maximizes the player's performances.

Key words: Soccer goalkeeper, Training, Robot, 3D human pose estimation, Artificial intelligence.

Résumé

Le projet consiste à modéliser et réaliser un robot tireur autonome dans le but d'aider l'évaluation et l'amélioration des performances du gardien de but de football.

En utilisant, une caméra et un estimateur de pose humaine 3D, le système peut détecter et modéliser les gestes physiques du gardien.

Il propose enfin, à l'aide d'une intelligence artificielle, la décision appropriée à prendre pour les prochains exercices, qui maximisent les performances du joueur.

Mots clés : Gardien de but de football, Entraînement, Robot, Estimation de la pose humaine en 3D, Intelligence artificielle.

Table des matières

Table des figures

Liste des tableaux

Liste des Abréviations

CML	Concurrent Mapping and Localization
CNN	Réseau de Neurones Convolutionnel
ECE	Ecole Centrale d'Electronique Paris
EKF	Extended Kalman Filter
GPU	General Processor Unit
IMC	Internal Model Controler
IMU	Unité de Mesure Inertielle
IA	Intelligence Artificielle
KBES	Knowledge Based Expert Systems
MBPC	Model Based Predictive Control
MPC	Model Predictive Control
OV	Odométrie Visuelle
PFC	Predictive Functional Control
RGB	Red-Green-Blue
RGB-D	Red-Green-Blue-Depth
SLAM	Simultaneous Localization and Mapping
VIO	Odométrie Inertielle Visuelle
3DPS	Structures Picturales 3D

Chapitre 1

Introduction Générale et Problématique

1.1 Introduction Générale

La robotique possède de nombreux domaines d'application. À l'origine, les robots ont été installés dans des industries (dès 1961), pour réaliser des tâches répétitives avec une précision constante.

Désormais, après le développement de la technologie et surtout après l'introduction de l'intelligence artificielle, on trouve des robots implantés dans presque tous les domaines et utilisés pour tous les usages, même dans un environnement proche des humains. Les algorithmes d'intelligence artificielle et les robots sont connus d'être introduits pour automatiser les taches et remplacer le travail humain. Par exemple, « LawGeex » utilise les dernières technologies en matière d'intelligence, d'apprentissage automatique, d'analyse de texte et de traitement du langage naturel pour réviser et comprendre les documents juridiques [1]. Ce système intelligent peut remplacer des centaines de conseillers juridiques qui sont souvent débordés de travail pour faire la révision des contrats quotidiens, ce qui coutait très cher en temps, argent et ressources humaines. En robotique, les humains sont concurrencés par des robots qui sont des « multiplicateurs de force » transformant de manière spectaculaire les industries de la construction et de la fabrication. Ils peuvent soulever des objets lourds plus rapidement et en toute sécurité, serrer à plusieurs reprises les vis au couple optimal et travailler sans pauses.

En revanche, la nouvelle technologie peut être utilisée aussi pour aider les êtres humains à acquérir une nouvelle compétence ou se perfectionner dans un certain domaine. On peut citer la machine qui a la capacité de vaincre les meilleurs joueurs du monde au jeu de plateau « GO ». En Mars 2016, l'intelligence artificielle (IA) « Alpha GO » développée par « DeepMind Technologies », a pu vaincre le champion du monde « Lee Sedol » dans ce jeu [2]. Alpha GO a utilisé initialement l'algorithme d'apprentissage par imitation pour apprendre en imitant les mouvements de ses adversaires humaines. Une fois un certain niveau atteint, Alpha GO s'est entraînée à jouer des millions de parties contre d'autres instances de lui-même, utilisant l'apprentissage par renforcement pour s'améliore. Cette machine et autres peuvent être utilisées par les personnes pour s'entrainer et s'améliorer.

Les athlètes sont aussi parmi les bénéficiaires de l'aide technologique. On peut citer :

— le robot tireur de balles de tennis qui donne un entrainement précis et intensif aux joueurs [3] (Figure 1.1.b) ;

— les robots mobiles développés par la « Japanese Volleyball Association » avec l'aide de chercheurs de l'Université de Tsukuba pour aider les attaquantes de l'équipe nationale féminine à s'entrainer [4] (Figure 1.1.a) ;

— l'entraineur virtuel au football, utilisé à la coupe du monde 2018, qui fournit des statistiques et propositions tactiques pour faire des changements de postes des joueurs ;

— et aussi le robot mobile « MVP » qui simule le placage des joueurs de football américain [5] (Figure 1.1.c).

FIGURE 1.1 – Exemples de robot sportifs.

1.2 Problématique

Dans le domaine du sport, certaines techniques nécessitent beaucoup d'entrainements pour les maitriser, donc, il faut prendre suffisamment de temps avec chaque joueur pour bien contrôler ses mouvements. Même dans les meilleures équipes au monde, le nombre d'entraineurs est très limité, notamment au football, il y a un seul entraineur spécialisé pour 4 gardiens de buts. Par conséquence, les joueurs de différents postes vont s'entraider pour assurer leurs entrainements. Ce qui fait que les attaquants de football, par exemple, passent une partie de leurs séances d'entrainement à aider les gardiens de buts à exercer des mouvements bien spécifiques, au lieu de se focaliser sur leurs entrainements en tant qu'attaquants.

La prise de décision a toujours était une tache difficile ; choisir quel joueur introduire dans un match est une décision faite par les entraineurs en se basant sur plusieurs critères comme les performances physiques et tactiques du joueur dans les entrainements. Cette décision est reliée aux compétences et expériences des entraineurs, donc, elle peut être biaisée.

Les blessures sont les plus grands ennemis des sportifs. Une fois la blessure traitée médialement, le joueur passe par des programmes spéciaux pour revenir à ses anciennes performances. Le suivi des joueurs après un arrêt de jeu est une partie très importante, vu qu'elle détermine le temps nécessaire pour revenir à la compétition, donc, elle peut affecter toute l'équipe s'il s'agit d'un joueur important. Les choix du type et intensité des exercices sont généralement fixés par les deux parties médicale et technique. Par conséquences, avoir un compromis entre l'équipe médicale qui veut assurer que le joueur peut rejouer en toute sécurité et l'équipe technique qui veut que le joueur revient au terrain le plus tôt possible est très difficile à assurer, alors, un troisième avis neutre est très recommandé.

Avec ce projet, nous voulons contribuer à une partie ce domaine sportif, qui est l'entrainement de gardiens de buts de football afin de minimiser les problèmes illustrées ci-dessus. Nous proposons un robot mobile, qui va être mis sur le terrain, pour accompagner l'entraineur principale des gardiens afin d'entrainer et tester leurs performances à l'aide de l'analyse 3D des réflexes du gardien pour des exercices bien étudiés. Par la suite, le robot fournira des rapports techniques détaillés pour aider l'entraineur à prendre les bonnes décisions. Il sera doté aussi d'une intelligence artificielle qui va aider à proposer les bons exercices à travailler dessus pour assurer le suivi des blessés et maximiser les performances des joueurs.

1.3 Cahier des Charges

Afin de réaliser notre stratégie, le robot devra être capable de :

— Se déplacer sur une surface plate pour se positionner aux cordonnées (x, y) du terrain choisi, de manière autonome ;

— lTourner de manière flexible afin d'orienter la partie du tir à un angle θ avec une bonne précision ;

— Se localiser sur le terrain à l'aide des odométres ;

— Stocker les balles dans le robot en exploitant la hauteur disponible ;

— Tirer les balles avec une vitesse et une trajectoire prédéfinie ;

— Estimer de la pose du gardien ;

— Analyser et étudier la réaction du gardien au tir.

Il s'agit donc d'un cahier des charges très précis. Les différents systèmes devront être modulables et le robot final le plus fiable et précis possible.

1.4 Structure du Mémoire

Nous débutons ce mémoire, qui compte huit chapitres, par la présentation du projet, l'objectif, la problématique et le contexte du sujet. Cette présentation fait l'objet du chapitre 1.

Le chapitre 2 est consacré aux notions de la robotique mobile (localisation, navigation, asservissement) ainsi que des méthodes de l'état de l'art.

Nous passons ensuite au chapitre 3 qui traite les notions et l'état de l'art d'analyse et interpretation de données, il inclue l'estimation de la pose humaine et la prise de décision en utilisant l'intelligence artificielle.

Le prochain et quatrième chapitre présente la modélisation et l'architecture du robot et le système d'analyse.

Le chapitre 5 est consacré à la conception du robot, en commençant par la conception en 3D avec le logiciel SolisWorks au dimensionnement et choix des composants.

Le chapitre 6 est dédié à la simulation du robot mobile sur le simulateur puissant et précis d'Ardupilot.

Nous passons au chapitre 7 qui présente les étapes de réalisation du prototype pour chaque partie du projet (mécanique, éléctronique et informatique).

Le chapitre 8 est consacré à l'étude et rélisation de l'analyse des réactions du gardiens de but. Il présente l'estimateur de la pose humaine ainsi qu'un cours sur l'apprentissage automatique qui nous permet de suggerer une méthode de prise de décision.

Enfin, le mémoires est complété par une conclusion générale qui fait une révision des pricipales conclusions aux quelles a abouti cette étude, met en avant les résultats et suggére quelques perspectives pour de futurs projets.

Chapitre 2

Robotique Mobile : Notions et Etat de l'Art

2.1 Système de lancement des ballons

Il existe des systèmes similaires au robot qu'on veut concevoir en ce qui concerne les mécanismes et le type de contrôle qui peuvent être utilisés comme référence. Pour la conception, la machine à lancer de baseball à roue simple conçue par « Baseball JUGS Sports » (Softball Passing Machine) [6] est conçue sur la base d'une seule roue de tirs ce qui lui donne la capacité de lancer des bales de baseball avec différentes puissances mais sur une seule trajectoire rectiligne. On va s'en inspirer pour notre système. Le fonctionnement de cette machine est basé sur la sécurité, la performance et la fiabilité comme priorité absolue pour assurer la performance des tirs. C'est précisément ces qualités qui vont être utilisées comme références pour la conception de notre robot sportif. Les composants de la machine JUGS sont visibles dans la Figure 2.1.

Des modifications dans la partie mécanique de la conception du système seront ajoutées afin de gagner en contrôlabilité du robot.

Les étudiants du « Georgia Institute of Technology » ont mené à bien un projet similaire appelé « Football Launcher » (AFL) [7] qui utilise une caméra afin de suivre la position d'un athlète et lance ainsi un ballon de football American au receveur. Leur système est un peu similaire à la machine JUGS, mais leur système de contrôle est plus développé. En plus il permet de lancer les ballons avec différentes trajectoires et orientations à l'aide d'une plate-forme rotative et basculante contrôlée par des moteurs assistés par ordinateur. Ce qui se traduit par une meilleure précision du lancement des ballons de football American. Une conception approximative de la plate-forme peut être vue dans Figure 2.2.

Cette conception a la même fonctionnalité souhaitée pour notre robot tireur, mais elle est beaucoup trop encombrante à mettre en œuvre dans la machine à lancer JUGS. Pour contrer cette difficulté, on se limite à l'utilisation de la partie basculante seulement afin de générer des trajectoires rectilignes sur un axe vertical variable. La partie rotative de notre système sera réalisée par une rotation de tout le robot vu que notre système est mobile.

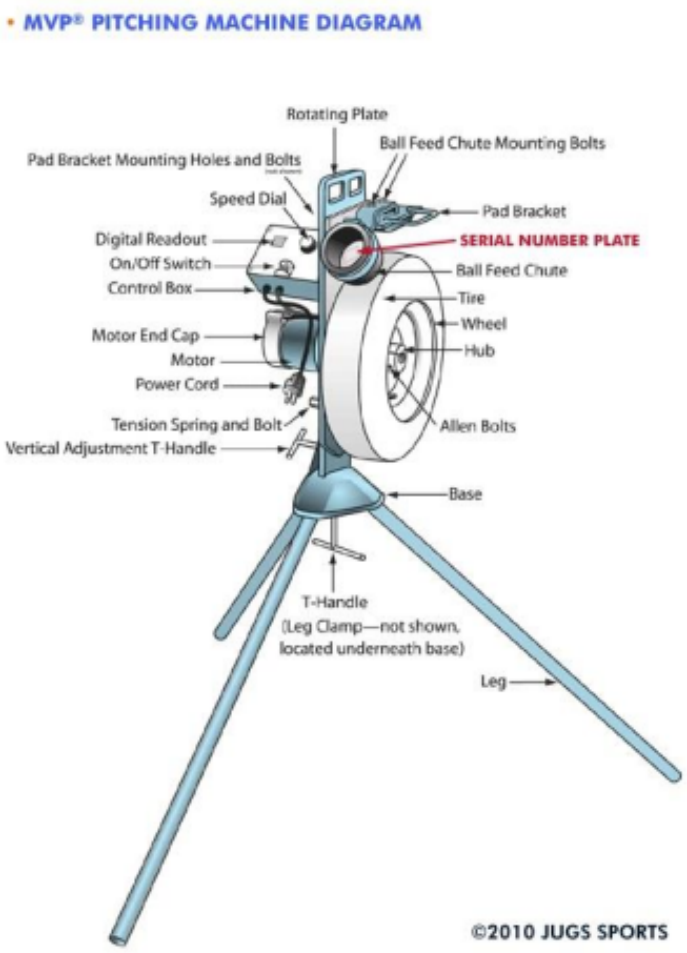


FIGURE 2.1 – Les composants du robot « Baseball JUGS Sport ».

D'autre part, « Globus EUROGOAL » est un produit qui existe déjà sur le marché et le dispositif est utilisé par certains gardiens de buts de football [8]. Cette machine est contrôlée manuellement par l'entraineur qui la déplace, configure ces paramètres des tirs (angle, force, trajectoire,..) et l'alimente de ballons. EUROGOAL offre des tirs bien contrôlés et très variables. Plusieurs configurations de trajectoires sont montrées dans la figure 2.3 ci-dessous.

Manifestement, la conception mécanique optimale qui donne une grande flexibilité et qui en plus offre différentes configurations est assurée majoritairement par un système de deux roues commandées par des moteurs séparés. Cette conception est représentée dans la Figure 2.4.

Une autre équipe d'étudiants mécaniciens ont travaillé sur un design qui consiste à s'engager de manière fiable sur des cibles fixes et mobiles de ballons balistiques de « Nerf » à une distance maximale de 4 mètres [9]. Cette conception est intéressante pour son système de stockage de balles. Le dispositif est montré dans la Figure 2.5 ci-dessous.

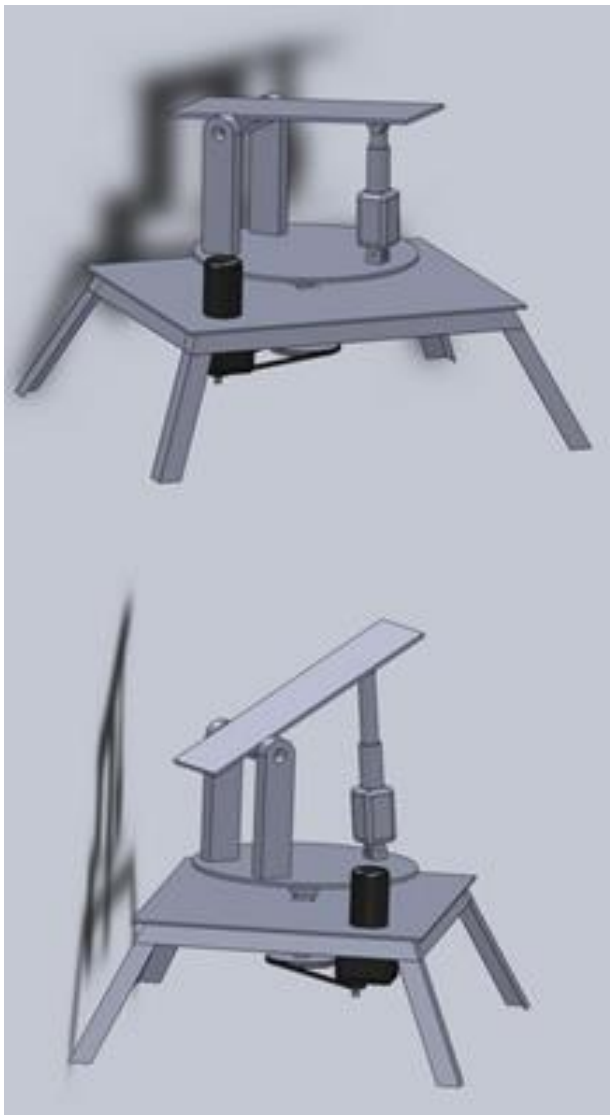


FIGURE 2.2 – Conception panoramique et inclinable du Georgia Tech AFL.

FIGURE 2.3 – Trajectoires possible d'EUROGOAL.

FIGURE 2.4 – Système de tir à deux roues.

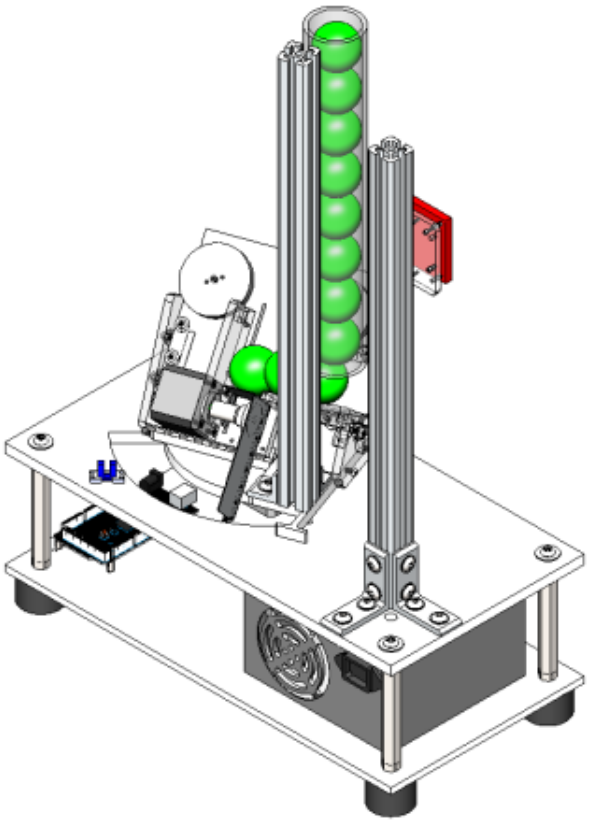


FIGURE 2.5 – Représentation par CAD de la machine à tir.

Conclusion Dans notre projet, on va s'inspirer du robot « Baseball JUGS Sports » pour la conception globale avec l'ajout d'une base mobile à quatre roues. En ce qui concerne les fonctions mécaniques assurées par le robot, on pourra reprendre les systèmes de rotation et tir des deux systèmes « Georgia Tech AFL » et « Globus EUROGOAL ». Finalement,

pour la partie stockage des ballons, on peut prendre comme référence le dispositif utilisé dans la dernière machine présentée. Pour la partie prototypage du robot, on peut prendre quelques conceptions déjà faites dans les projets cités au paravent et faire la conception des autres pièces en 3D pour l'usinage.

2.2 Localisation

Pour les différentes applications considérées par la robotique dans des environnements externes, autrement dit, pour la robotique mobile d'une manière générale, la capacité pour un robot de se localiser, c'est-à-dire de connaître à tout instant sa position avec précision est essentielle.

On peut distinguer deux grandes catégories de méthodes de localisation : en boucle ouverte (BO) et en boucle fermée (BF). On va présenter dans ce qui suit deux méthodes qui fonctionnent en BO et une autre qui représente l'état de l'art actuel et qui utilise une fermeture de la boucle comme principe.

2.2.1 Odométrie

L'odométrie est une technique permettant d'estimer la position d'un véhicule en mouvement. Cette mesure de bas niveau est présente sur quasiment tous les robots mobiles, grâce à des capteurs embarqués permettant de mesurer le déplacement du robot (plus précisément de ses roues).

L'odométrie repose sur la mesure individuelle des déplacements des roues afin de reconstituer le mouvement global du robot. En partant d'une position initiale connue et en intégrant les déplacements mesurés, on peut ainsi calculer à chaque instant la position courante du véhicule [10]. Pour calculer le mouvement global du robot à partir des mesures odométriques, il est nécessaire de disposer d'un modèle décrivant le déplacement du robot. Le mouvement global du robot peut être présenté comme une translation et une rotation. L'exemple d'un robot, dont le déplacement est contrôlé par le différentiel de vitesse entre les deux roues motrices, est décrit dans la Figure 2.6.

Connaissant la pose du robot à l'instant n-1, on cherche la pose à l'instant n [11]. On a donc :

$$\Delta moy_n = \frac{\Delta d_n + \Delta g_n}{2}$$

$$\Delta dif_n = \Delta d_n + \Delta g_n$$

$$\Delta x_n = \Delta moy_n \Delta \cos_{n-1}(\theta)$$

$$\Delta y_n = \Delta moy_n \Delta \sin_{n-1}(\theta)$$

$$\Delta \theta_n = \frac{\Delta d_i f_n}{L}$$

Sachant que :

- Δg : Distance parcourue par la roue gauche du robot pendant une période T
- Δd : Distance parcourue par la roue droite du robot pendant une période T
- L : La distance entre les 2 roues
- (x, y, θ) : La pose du robot

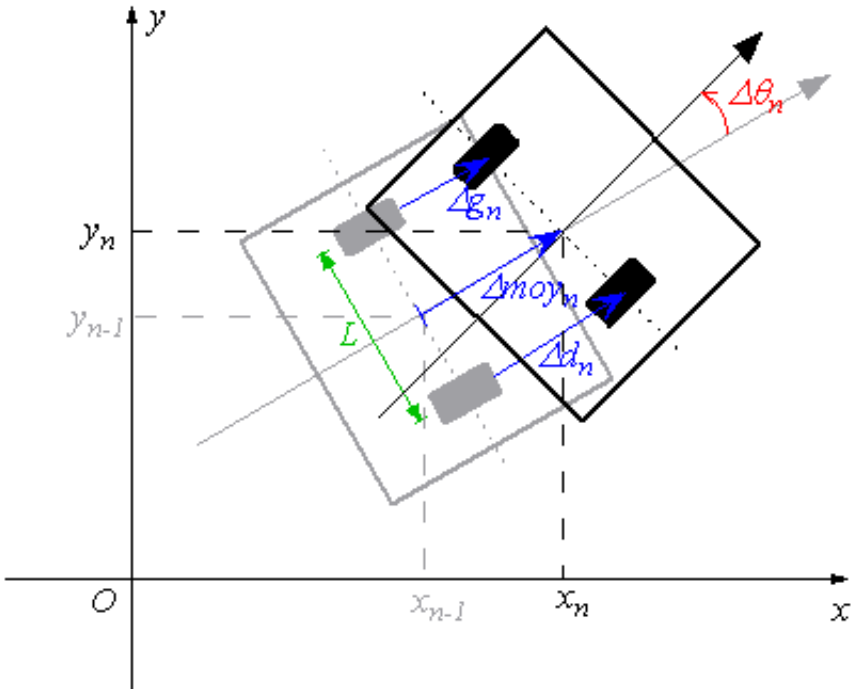


FIGURE 2.6 – Représentation du déplacement.

Ainsi, on intègre pour avoir la distance parcourue totale :

$$x_n = x_{n-1} + \Delta x_n$$
$$y_n = y_{n-1} + \Delta y_n$$
$$\theta_n = \theta_{n-1} + \Delta\theta_n$$

L'odométrie présente un problème majeur, qui, en l'occurrence, est le dérapage (drift). Celui-ci est peut être causé par les frottements des roues, la nature du sol et les mauvais points de contacts avec la surface de déplacement ce qui fausse inévitablement l'estimation du positionnement.

2.2.2 Odométrie Visuelle

En robotique, l'odométrie visuelle (OV) est le processus permettant de déterminer la position et l'orientation d'un robot en analysant les images fournies des caméras associées au robot. Cette méthode a été utilisée dans une grande variété d'applications en robotique, telles que la voiture d'exploration de Mars.

L'odométrie visuelle est le processus de détermination d'informations sur le positionnement équivalent à l'odométrie standard à l'aide d'images séquentielles d'une caméra pour estimer la distance parcourue. L'odométrie visuelle permet une précision de navigation améliorée dans les robots ou les véhicules, quel que soit le type de locomotion utilisé, sur n'importe quelle surface.

Les différents types de l'odométrie visuelle qui peuvent être combinés, sont :

Monoculaire ou Stéréo En fonction de la configuration de la caméra, l'OV peut être catégorisée en OV monoculaire (caméra unique), stéréo OV (deux caméras en configuration stéréo).

Méthode Directe ou Indirecte Les informations visuelles de l'OV traditionnelle sont obtenues par la méthode basée sur les caractéristiques de l'image collectée de l'environnement. Elle extrait les points caractéristiques des images et les suit dans la séquence d'images.
Un développement récent dans la recherche sur l'OV a fourni une alternative, appelée méthode directe, qui utilise l'intensité du pixel dans la séquence d'images directement comme entrée visuelle. Il existe également des méthodes hybrides.

Odomètre visuel inertiel Si une unité de mesure inertielle (IMU) est utilisée dans le système OV, elle est communément appelée odométrie inertielle visuelle (VIO) [12]

2.2.3 SLAM

La localisation et cartographie simultanées, connue en anglais sous le nom de SLAM « Simultaneous Localization and Mapping » ou CML « Concurrent Mapping and Localization », consiste, pour un robot ou véhicule autonome, à simultanément construire ou améliorer une carte de son environnement et de s'y localiser [13]. Le SLAM est donc défini comme le problème de la construction d'une carte en même temps que la localisation du robot dans ce plan.

Plusieurs approches ont été proposées dans la littérature pour résoudre le problème de SLAM. Celles-ci peuvent être classifiées, dans un premier lieu, en approches probabilistes et ensemblistes. L'approche dominante est l'approche probabiliste. Elle cherche à estimer, en général, la densité à posteriori de la carte et la position du robot. Par ailleurs, on peut diviser encore les approches de SLAM en approches de filtrage et approches de lissage (ou approches d'optimisation). Les approches de filtrage corrigent la position courante et la carte. Dans cette catégorie, on retrouve l'EKF-SLAM (Extended Kalman Filter for SLAM) [14] et le FastSLAM (Filtre particulaire) [15]. Les approches de lissage permettent de corriger la carte des amers et toute la trajectoire depuis l'état initial. Le SLAM par optimisation de graphe [16] est une méthode de lissage [17].

Dans la pratique, on peut découper le processus de SLAM en quatre étapes :

- Extractions des amers
- Estimation des mesures
- Mise en correspondance
- Correction

2.2.3.1 EKF-SLAM

L'EKF a été donc la première approche utilisée pour résoudre le problème de SLAM. C'est une extension du filtre de Kalman qui prend en compte les systèmes non linéaires. En effet, les modèles de mouvement et d'observation sont, en général, non linéaires. L'EKF-SLAM est une approche Bayésienne récursive qui nécessite plusieurs hypothèses :

- La trajectoire du robot forme une chaîne de Markov de premier ordre. La position courante du robot xt ne dépend que de la position précédente et de la commande ut.
- Les observations des amers sont indépendantes entre elles.
- L'observation d'un amer à l'instant t ne dépend que de la position du robot à cet instant.

- Les mesures capteurs sont affectées d'un bruit blanc Gaussien.

Sous ces hypothèses, la formulation Bayésienne de l'EKF-SLAM est donnée par :

$$p(x_1, m|z_{1:t}, u_{1:t}) = \nu p(z_t|x_t, m) \int p(x_t|x_{t-1}, u_t) p(x_{t-1}, m|z_{1:t-1}, u_{1:t-1}, x_0) dx_{t-1}$$

//équaion Où h est une constante de normalisation [18]. //équation

2.2.3.2 Graph SLAM

Cette approche constitue depuis une dizaine d'années un axe de recherche très actif au sein de la communauté de robotique. L'estimation de l'état du système est formulée par un problème d'optimisation. Ce dernier fait appel à plusieurs techniques issues essentiellement de l'algèbre linéaire et de la théorie du graphe. Le GraphSLAM ou (SLAM basé optimisation de graphe) modélise le problème de SLAM à l'aide d'un graphe. Comme l'illustre la Figure 2.7, la trajectoire et la carte des amers sont représentées par des nœuds. Associées à un bruit Gaussien, les mesures capteurs donnent des contraintes spatiales entre les nœuds. Ces contraintes spatiales sont modélisées par des arêtes. On distingue deux types d'arêtes : arêtes de mouvement et arêtes d'observation. Une arête de mouvement relie deux nœuds robot (pose) consécutifs. Une arête d'observation provient d'une observation d'un amer. Un amer est relié à une pose (nœud robot) s'il a été observé depuis celle-ci [18].

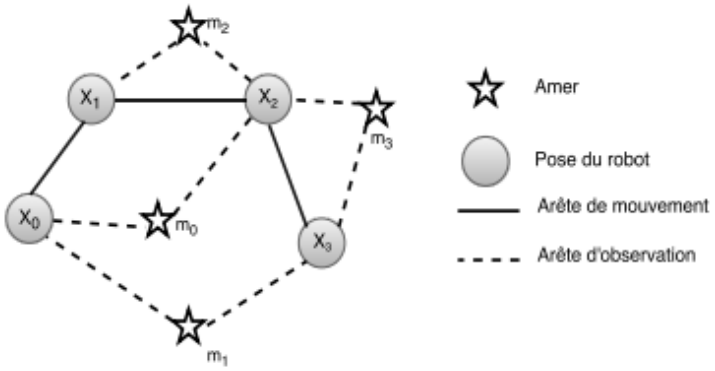


FIGURE 2.7 – Illustration du GraphSLAM.

Le GraphSLAM est une approche de lissage qui consiste, contrairement à l'EKF-SLAM, à ré-estimer, en plus de la position courante, toute la trajectoire depuis l'état initial en prenant en compte tout l'historique des mesures. De même que les autres approches Bayésiennes, le GraphSLAM suppose que les bruits sont Gaussiens. Le GraphSLAM estime la densité p(x1 :t, m|z1 :t, u1 :t, x0). A la différence du filtre particulaire qui génère plusieurs hypothèses de trajectoire, le GraphSLAM modélise toutes les contraintes spatiales entre les nœuds. Dans la pratique, le GraphSLAM est divisé en deux parties : partie en-amont « Front-end » et partie en-aval « Back-end » [18].

2.2.4 Conclusion

La localisation de notre robot est dans un environnement bien connu (carte métrique définie et fixe) donc, on n'aura pas un besoin de cartographier notre environnement par une

méthode comme le SLAM. Nous utilisons des odométres comme capteurs avec la méthode de fusion des données (méthode en boucle fermée) pour localiser notre robot sur le stade, mais elle peut poser des problèmes car le robot se déplace sur du gazon qui peut être dans certains cas moulé, ce qui augmente son coefficient de glissement et par conséquent, peut causer des glissements de roues qui fausse l'estimation du positionnement. Pour remédier à ce problème, il faut utiliser des roues adéquates c.-à-d. des roues qui ne glissent pas sur le gazon.

2.3 Navigation

Selon le standard IEEE 172-1983 : « **La navigation est le processus de guider un véhicule afin de parvenir à la destination** ».

La navigation est une tâche essentielle dans la robotique car elle définit les actions ou les mouvements générés par le robot pour atteindre son but en évitant l'ensemble des obstacles. Pour la partie qui suit, on a supposé que la navigation dynamique des obstacles n'est pas nécessaire c.-à-d. le robot n'aura pas d'obstacle comme : des personnes, ballons ou équipements d'entrainement sur sa surface de déplacement.

2.3.1 Navigation topologique

Elle fournie une capacité de navigation globale, ce qui permet le déplacement d'un lieu à l'autre sans but fixé (le but est invisible avec des amers voisins invisibles).

Cette méthode nécessite ce pendant une représentation interne de l'environnement en définissant l'ensemble de lieux et la mémorisation des relations spatiales entre les lieux. Le modèle interne est un graphe qui permet le calcul des chemins entre lieux pour proposer un chemin optimal (planification) [19].

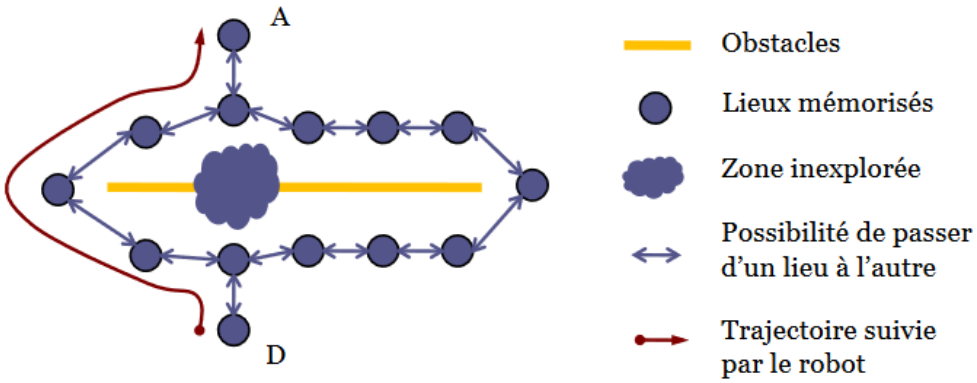


FIGURE 2.8 – Exemple d'utilisation de la méthode topologique.

2.3.2 Navigation métrique

Capacité étendue de la méthode précédente, elle permet de planifier des chemins dans des zones inexplorées par la possibilité de passage d'un lieu à l'autre sans avoir un lien pré-

définie entre les deux. Ce passage est possible par la détermination les positions métriques relatives des différents lieux [20].

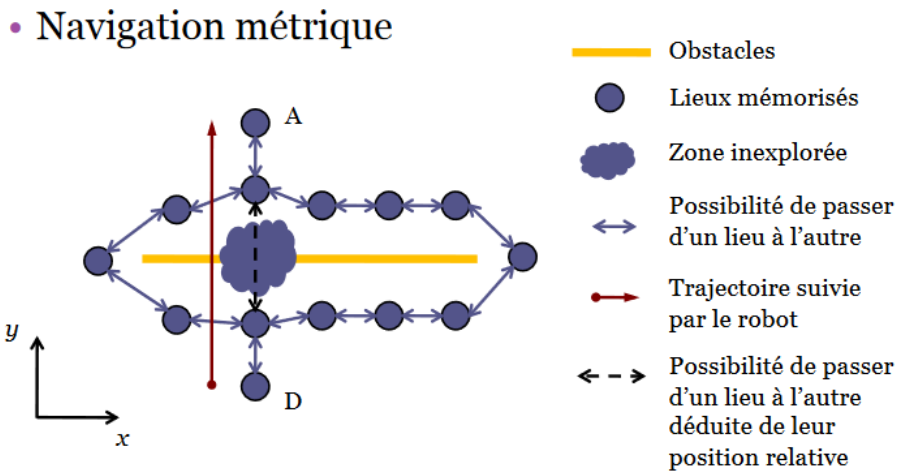


FIGURE 2.9 – Exemple d'utilisation de la méthode métrique.

Ces deux méthodes présentées au-dessus, sont utilisées pour des environnements statiques.

2.3.3 Conclusion

Pour notre robot, les dimensions du stade de football sont connues donc, on va opter pour une navigation métrique (statique avec carte).

2.4 Commande et Asservissement

Afin d'assurer l'asservissement des moteurs pour arriver à la référence désirée par le système de navigation, on utilise une commande d'asservissement.

2.4.1 Commande prédictive à base de modèle (MPC)

La stratégie appelée commande prédictive à base de modèle « Model Prédictive Control » (MPC) optimise, à partir des entrées, le comportement futur anticipé du système considéré. La prédiction est faite à partir d'un modèle du système sur un intervalle de temps fini appelé horizon de prédiction [21].

La commande prédictive fait partie des techniques de contrôle à modèle interne « Internal Model Controler » (IMC). En anglais on utilise le terme MPC ou MBPC pour qualifier la commande prédictive « Model (Based) Predictive Control » ou aussi PFC « Predictive Functional Control » [22].

Principe Elle consiste à optimiser à chaque période d'échantillonnage le critère de performance (J) soumis à des contraintes fonctionnelles, et à déterminer la meilleure séquence des commandes sur l'horizon de prédiction K. La première commande de la séquence optimale est alors appliquée et la résolution recommence en prenant en compte les informations disponibles réactualisées (Figure 2.11). La répétition de cette procédure à chaque période permet de balayer le temps avec un horizon fini.

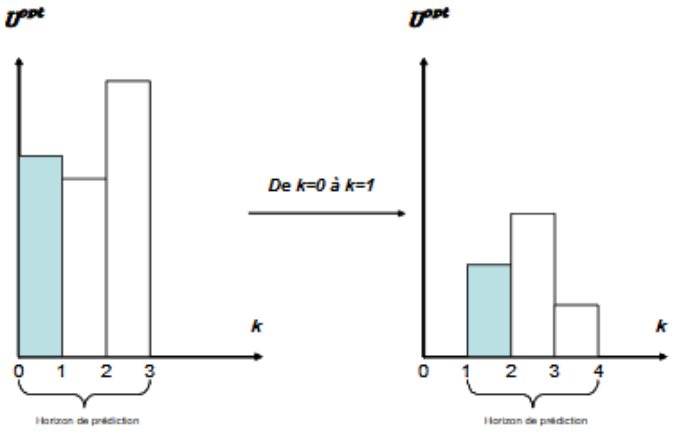


FIGURE 2.10 – Le principe de l'horizon glissant.

La commande prédictive réalise à chaque période d'échantillonnage du contrôleur les mêmes étapes, à savoir :

- Calcul des prédictions des variables contrôlées jusqu'à un horizon de temps N2 (Figure 2.11) grâce au modèle interne.
- Élaboration d'une trajectoire de référence à suivre.
- Calcul de la future loi de commande à appliquer sur les variables manipulées jusqu'à un horizon temporel Nu.
- Seul le premier élément de la loi de commande calculée, est appliqué sur le système au coup d'horloge suivant. Toutes ces étapes se répéteront ensuite, c'est le principe de l'horizon fuyant.

Il est important que la stratégie de régulation développée puisse s'appliquer en boucle fermée. Ceci est possible suivant le schéma donné sur la Figure 2.12 [22].

2.4.2 Commande par platitude

La notion platitude est une propriété caractérisant une classe de systèmes non linéaires. Elle a été définie dans le cadre de l'algèbre différentielle, puis dans le cadre de la géométrie différentielle. Le concept de la platitude introduit une notion d'équivalence entre un système non linéaire et un système linéaire commandable. Cette équivalence porte le nom d'équivalence par bouclage dynamique endogène dans le cadre de l'algèbre différentielle et d'équivalence de Lie-Bäcklund dans le cadre de la géométrie différentielle [18].

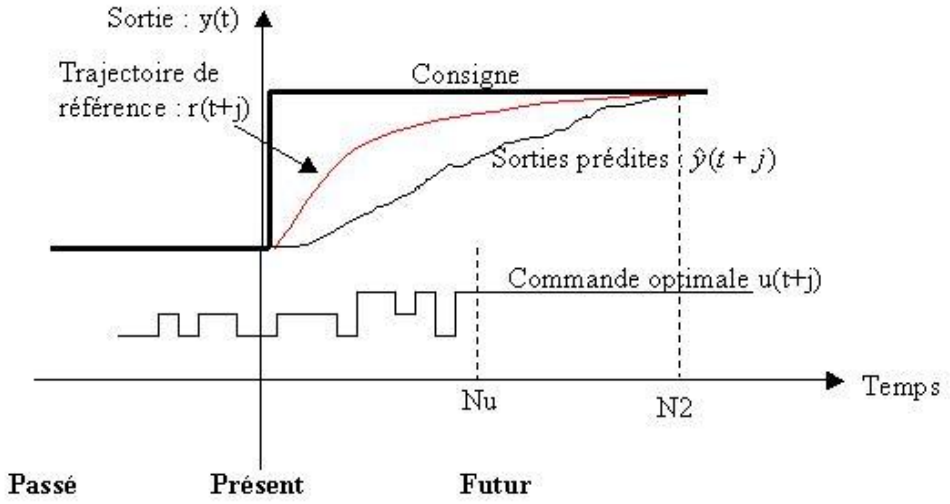


FIGURE 2.11 – Le principe de fonctionnement de MPC en horizon glissant.

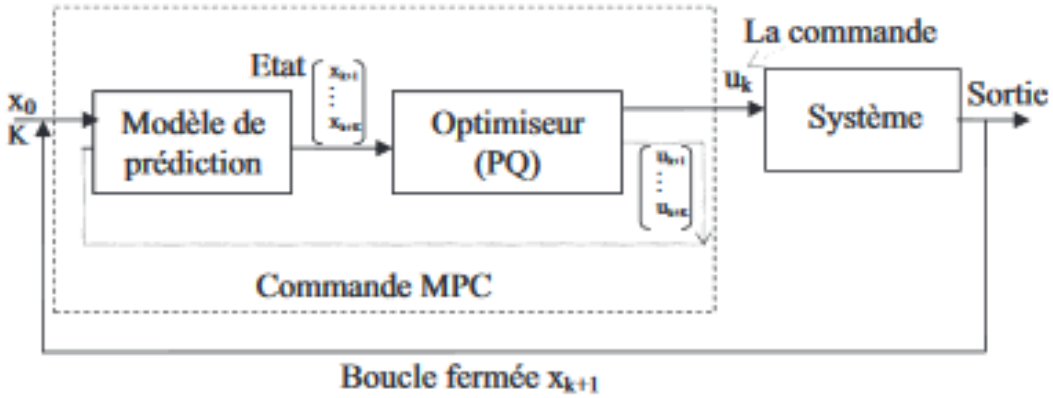


FIGURE 2.12 – Représentation de la boucle fermée de commande.

2.4.2.1 Définition d'un système plat

Soit un système non linéaire dont l'entrée u = (u1, ..., um) et l'état :

$$\begin{cases} x = (x_1, ..., x_n) \\ \dot{x} = f(x, u) \end{cases} \tag{2.1}$$

Où f est un champ de vecteur lisse.
Ce système est (différentiaient) plat s'il existe un ensemble de m quantités $z = (z1, ..., zm)$ qui ont les 3 propriétés suivantes :

- Les éléments de z sont différentiellement indépendants ;
- Les éléments de z peuvent être exprimés comme : $z = h(x, u, \dot{u}, ..., u^{(\alpha)})$ où : x ∈ N
- Toutes les variables du système peuvent être exprimées indépendamment en fonction de z et de ses dérivées, et en particulier.

$$u = \psi(z, \dot{z}, ..., z^{(\beta)})$$

$$x = \phi(z, \dot{z}, ..., z^{(\gamma)})$$

L'entier (m) est précisément le nombre de composants d'entrée du système 2.1. L'ensemble z est appelé sortie plate ou sortie linéarisante du système. La seconde condition signifie que les éléments de la sortie plate sont les variables du système 2.1 qui s'expriment simplement à partir des variables du système. La troisième condition implique que toute variable du système ou chaque fonction de variable du système est paramétrable par la sortie plate et un nombre fini de ses dérivées. Finalement, la première condition signifie que les éléments de la sortie plate ne sont pas liés entre eux par une équation différentielle : ils sont entièrement libres et ainsi il est possible de prévoir —et ainsi de concevoir— leurs trajectoires afin de réaliser le contrôle du système 2.1.

L'intérêt de la commande par la platitude vient du fait que de nombreux systèmes de la pratique sont plats et que la sortie plate est souvent reliée aux variables à commander. Dans le contexte des systèmes linéaires, les systèmes plats correspondent précisément aux systèmes commandables. La notion de la platitude peut ainsi être considérée comme une sorte de généralisation non linéaire de la commandabilité.

2.4.2.2 Planification des trajectoires

Cette section rappelle comment la platitude différentielle permet de planifier une trajectoire nominale de l'entrée (et de l'état) basée sur la propriété de paramétrage de toute variable à partir d'une sortie plate.

Soit un intervalle de temps (I) tel que : (I $\subset \mathbb{R}$), de forme I = [to, tf] avec to $\in \mathbb{R}$ et tf est un nombre réel tel que tf $>$ to ou tf $= +\infty$. La trajectoire nominale de la sortie plate z du système 2.1 est :

$$z^{\#} : I \rightarrow \mathbb{R}^{\mathrm{m}}$$

Si elle est suffisamment différentiable de sorte que les expressions de l'entrée (u) et l'état (x) sont définies sur l'intervalle (I). Ceci implique aussi que la trajectoire nominale de la sortie plate $z^{\#}$ évite d'éventuelles singularités des fonctions ψ et ϕ de u et x respectivement. En donnant une trajectoire nominale admissible à la sortie plate $z^{\#}$, on obtient la trajectoire nominale $u^{\#}$ de l'entrée u en utilisant la formule suivante :

$$u^{\#}(t) = \psi(z^{\#}(t), \dot{z}^{\#}(t), ..., z^{\#(\beta)}(t)), \forall t \in I$$

De même, une trajectoire d'état nominal $x^{\#}$ est définie :

$$x^{\#}(t) = \phi(z^{\#}(t), \dot{z}^{\#}(t), ..., z^{\#(\gamma)}(t)), \forall t \in I$$

2.4.2.3 Formes canoniques du système plat

Deux formes canoniques sont nécessaires pour la représentation du système plat : la première ressemble à une forme contrôleur est obtenue par changement d'état ; la seconde ressemble à la forme de Brunovsky des systèmes linéaires qui est obtenue par changement d'état et bouclage.

Forme contrôleur On a :

$$X = (z_1, \dot{z}_1, ..., z_1^{(k_1-1)}, z_2, ..., z_m^{(k_1-m)})$$

Où les (ki) sont des entiers qui correspondent aux indices de commandabilité et bien entendu m = n avec n la dimension de l'état du système.

$$X = (X_1, ..., X_m) = (X_1^1, X_1^2, ..., X_1^{k_1}, X_2^{k_2}, ..., X_m^1, ..., X_m^{k_m})$$

L'indice inferieur correspond —comme cela apparaît clairement dans les formes canoniques— au numéro du sous-système ; l'indice supérieur désigne l'ordre des variables d'un sous-système (et ne devrait pas être interprété comme une exponentiation). L'état du sous-système (i) est , ..., . En utilisant ce nouvel état, le système 2.1 peut être réécrit comme :

$$\dot{\chi}_1^1 = \chi_2^1$$

$$..$$

$$..$$

$$\dot{\chi}_1^{k_1} = g_1(\chi, u, \dot{u}, ..., u^{(\sigma_1)})$$

$$\dot{\chi}_1^1 = \chi_2^2$$

$$..$$

$$..$$

$$\dot{\chi}_{m-1}^{k_{m-1}} = g_{m-1}(\chi, u, \dot{u}, ..., u^{(\sigma_{m-1})})$$

$$..$$

$$..$$

$$\dot{\chi}_m^1 = \chi_m^2$$

$$..$$

$$..$$

$$\dot{\chi}_m^{k_m} = g_m(\chi, u, \dot{u}, ..., u^{(\sigma_m)})$$

Le schéma bloc de la forme canonique est montré dans la Figure 2.13

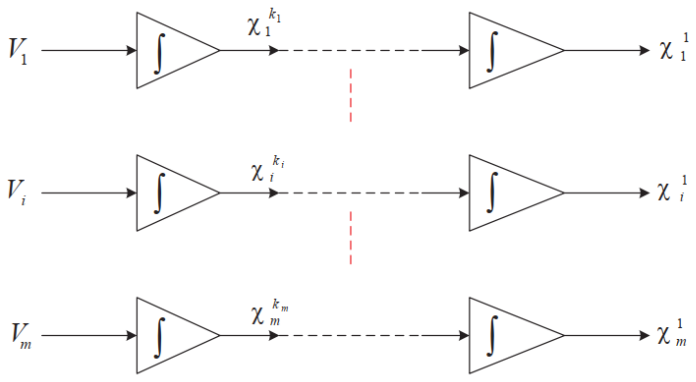


FIGURE 2.13 – Forme canonique par changement d'état.

Forme de Brunovsky On a l'état x tel que :

$$x = T(x, u, \dot{u}, ..., u^{(\sigma-1)}), i = 1, ..., m$$

Où : $\sigma = \max\{\sigma_1, ..., \sigma_m\}$ est l'application de retour endogène dynamique.
et on définit une sortie intermédiaire :

$$v_i = g_i(x, u, \dot{u}, ..., u^{(\sigma_i)})$$

Avec ces notations, on peut écrire le système sous la forme de Brunovsky :

$$\dot{\chi}_1^1 = \chi_1^2$$

..

..

$$\dot{\chi}_1^{k1} = v_1$$

..

..

$$\dot{\chi}_m^1 = \chi_m^2$$

..

..

$$\dot{\chi}_m^{km} = v_m$$

Le schéma bloc de la forme canonique [24] est représenté dans la Figure 2.14 :

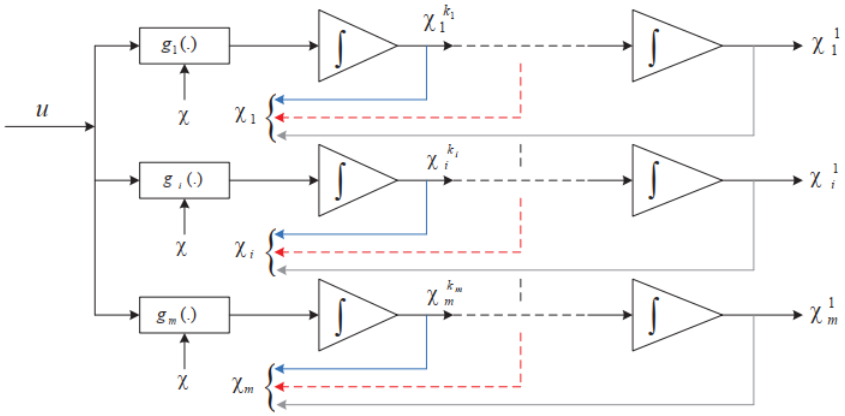


FIGURE 2.14 – Forme canonique de Brunovsky.

Chapitre 3

Analyse des Données : Notions et Etat de l'Art

3.1 Estimation de la Pose Humaine

L'estimation de la pose d'un humaine à partir d'une image ou d'une vidéo a récemment fait l'objet d'une attention particulière de la part de la communauté scientifique. Les principales raisons de cette tendance sont la nouvelle gamme toujours croissante d'applications (par exemple, l'interaction entre robots humains, les jeux, l'analyse de la performance des sportifs), qui sont guidées par les avancées technologiques actuelles [25].

On peut catégoriser les méthodes d'estimation de la pose humaine par plusieurs critères : par le format de la présentation 3D ou 6D, par le type de camera utilisée RGB ou RGB-D, par nombre de source de données (image ou vidéo) et par l'algorithme utilisé avec Deep Learning ou sans. Dans ce qui suit, on va citer quelques méthodes de chaque catégorisation.

3.1.1 Reconnaissance de la Pose Humaine en Temps Réel par Parties du Corps à l'Aide d'une Seule Image de Profondeur (RGB-D)

C'est une méthode développée par « Microsoft Research Cambridget Xbox Incubatation ». Elle est considérée comme une méthode rapide et précise de prédiction des positions en 3D des articulations du corps à partir d'une seule image de profondeur (en utilisant une Kinect), autrement dit n'utilisant aucune information temporelle.

En utilisant une approche de reconnaissance d'objet, on peut générer une conception intermédiaire qui présente les parties du corps afin de simplifier le problème de l'estimation de la pose vers un problème de classification par pixel. Avec le grand nombre et la grande variété de données d'entrainement, cela permet au classificateur d'estimer les parties du corps invariantes à la pose, la forme du corps, les vêtements, etc. Enfin, on peut générer des propositions 3D de scores de confiance.

Le système fonctionne à 200 images par seconde sur le matériel du consommateur. L'évaluation de la méthode, montre une grande précision sur les deux tests synthétiques et réels, et permet d'étudier l'effet de plusieurs paramètres d'entraînement général [26].

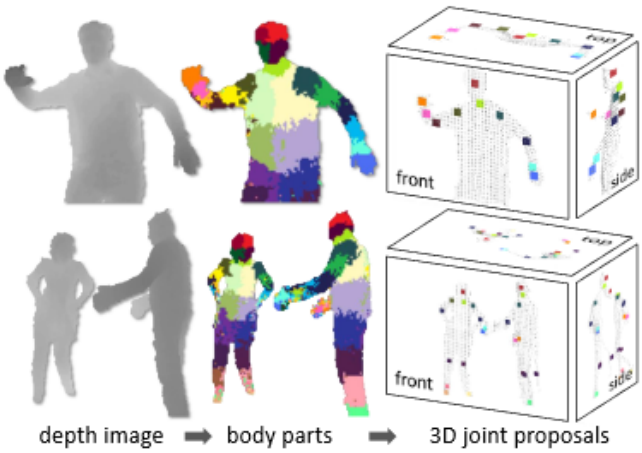


FIGURE 3.1 – Etapes de conception de la pose 3D à partir de l'image de profondeur.

3.1.2 Estimation de la Pose Humaine en Temps Réel avec une Seule Caméra RGB

Cette méthode utilise une simple camera RGB et elle donne de très bon résultats à l'intérieur comme à l'extérieur. Elle combine un nouveau régresseur de pose basé sur un réseau de neurones convolutionnel (CNN) et un ajustement cinématique du squelette. Notre nouvelle formulation de pose entièrement convolutive régresse les positions conjointes 2D et 3D conjointement en temps réel et ne nécessite pas de trames d'entrée étroitement découpées.

Une méthode d'ajustement squelettique cinématique en temps réel utilise la sortie CNN pour obtenir des reconstructions 3D globales de postures stables dans le temps sur la base d'un squelette cinématique cohérent. Cela fait de l'approche la première méthode RGB monoculaire utilisable dans les applications en temps réel telles que le contrôle de caractères 3D - à ce jour, les seules méthodes monoculaires pour de telles applications utilisent des caméras RGB-D spécialisées. La précision de cette méthode est quantitativement égale à celle des meilleures méthodes d'estimation de pose RGB monoculaire 3D hors ligne. Les résultats sont qualitativement comparables aux résultats d'approches monoculaires RGB-D, telles que Kinect, et parfois meilleurs. Cependant, cette approche est plus largement applicable que les solutions RGB-D, car elle peut être aussi utilisée pour les scènes en extérieur, les vidéos communautaires et les appareils photo RGB de qualité médiocre [27].

FIGURE 3.2 – Les étapes de la méthode avec une caméra RGB.

3.1.3 Récupération d'une Pose Humaine 3D à Partir d'une Séquence d'Images à Vues Multiples

Les approches de Belagiannis et Al. [28] et Sigal et Al. [29] ont utilisé des modèles humains 3D pour estimer la pose à partir d'une séquence d'images en vue multiple scénarios. La méthode de Belagiannis et Al estime conjointement la pose 3D de plusieurs humains dans une vue multiple scénarios.

Le premier obstacle que les auteurs ont voulu surmonter est l'espace d'états complexe de grande dimension. Au lieu de les discrétiser, ils ont utilisé la triangulation des articulations du corps échantillonnées à partir des postérieurs des détecteurs 2D de parties du corps dans toutes les paires de vues de caméra. Les auteurs ont introduit un modèle de structures picturales 3D (3DPS) qui déduit la posture articulée de plusieurs êtres humains à partir de l'espace d'états réduit, tout en résolvant les ambiguïtés qui découlent à la fois du scénario à vues multiples et de l'estimation à l'homme multiple.

3.1.4 Approche Faiblement Supervisée pour Estimation 3D de la Pose Humaine à l'Etat dans un Environnement Externe à partir des Images 2D

Récemment, Mehta et Al. [30] ont montré que le transfert de connaissances 2D vers 3D, c'est-à-dire que l'utilisation de réseaux de pose 2D préformés pour initialiser les réseaux de régression de pose 3D peut améliorer considérablement les performances d'estimation de pose 3D. Ce qui indique que les tâches d'estimation de pose 2D et 3D sont intrinsèquement empêtrées et qu'elles pourraient partager des représentations communes.

Inspirée par ces travaux, une autre équipe de recherche [31] a utilisé le transfert de connaissances inverse, c'est-à-dire à partir d'annotations 3D d'images de l'intérieur « indoor » on peut les utilisées pour l'estimation avec des images à l'extérieur « outdoor » et ça offre une solution efficace pour la prédiction de l'état 3D dans un environnement externe.

Ils ont proposé un travail unifié pouvant exploiter des annotations 2D d'images de l'extérieur en tant qu'étiquettes faibles pour la tâche d'estimation de pose 3D. En d'autres termes, nous considérons un système faiblement supervisé de problème de transfert d'apprentissage, où le domaine source est constitué d'images entièrement annotées dans un environnement intérieur restreint et le domaine cible se compose d'images faiblement étiquetées dans un environnement externe.

Ce travail apporte les avantages suivants :

- Pour la première fois, une estimation de la pose humaine 3D de bout en bout (end to end) avec des images dans la nature. Il réalise des performances de pointe sur plusieurs critères.
- Propose une contrainte géométrique 3D pour l'estimation de pose en 3D à partir d'images avec uniquement des annotations jointes 2D.
- Il a un faible coût en mémoire et en calcul. Il améliore la validité géométrique des poses estimées.

Le code est accessible au public sur : https ://github.com/xingyizhou/pose-hg-3d.

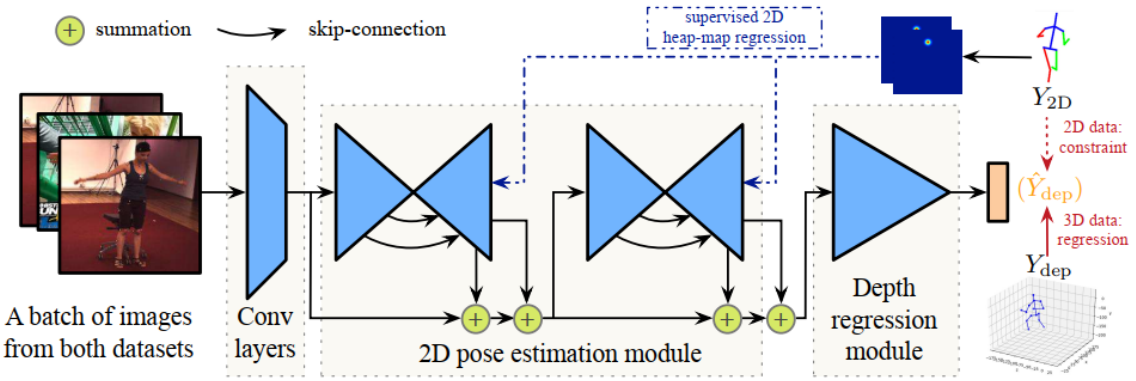


FIGURE 3.3 – La structure de test et d'entrainement proposée.

Pour le test : Les images sont transformées en cartes thermiques 2D. Ces cartes thermiques 2D et les images de couche inférieure sont additionnées en tant qu'entrée du module de régression de profondeur suivant.

Pour l'entrainement : Des images de données 2D et 3D sont mélangés dans un seul lot. Pour les données 3D, la régression standard avec perte euclidienne est appliquée. Pour les données 2D, nous proposons une perte faiblement supervisée basée sur son annotation 2D et sa connaissance préalable du squelette humain [31].

3.1.5 Pose CNN : un Réseau de Neurones Convolutif pour Estimation de la pose 6D d'Objets dans des Scènes Encombrées

La reconnaissance de l'emplacement 3D et de l'orientation des objets est importante pour la manipulation du robot. Il est également utile dans les tâches d'interaction robot/humain telle que l'apprentissage à partir d'une démonstration. Cependant, le problème est difficile en raison de la variété des objets dans le monde réel. Ils ont différentes formes en 3D, et leurs apparences sur les images sont affectées par les conditions d'éclairage, encombrement dans la scène et occlusions entre objets.

Traditionnellement, le problème de l'estimation de la pose d'objet 6D est abordé en faisant correspondre les points caractéristiques entre les modèles 3D et images [32]. Cependant, ces méthodes exigent qu'il y ait des textures riches sur les objets afin de détecter la fonctionnalité points pour l'appariement. En conséquence, ils sont incapables de gérer des objets sans texture.

Avec l'émergence des caméras de profondeur, plusieurs méthodes ont été proposées pour reconnaître des objets sans texture utilisant des données RGB-D [33]. Pour les méthodes basées sur des modèles [34], les occlusions réduisent considérablement les performances de reconnaissance. Sinon, les méthodes qui permettent d'apprendre à régresser les pixels de l'image en coordonnées d'objet 3D afin d'établir les correspondances 2D-3D pour l'estimation de pose 6D [35] ne peuvent pas gérer les objets symétriques.

Dans une recherche plus récente [36], elle propose une structure générique pour l'estimation de la pose 6D des objets où ils surmontent les limites des méthodes existantes à l'aide d'un nouveau réseau de neurones convolutifs (CNN) pour une estimation de pose 6D de bout en bout nommée PoseCNN. Une idée clé de PoseCNN est de découpler la

tâche d'estimation de pose en différents composants, ce qui permet au réseau de modéliser explicitement les dépendances et les indépendances qui les unissent. Plus précisément, PoseCNN effectue trois tâches connexes, comme illustré à la Figure 3.4.

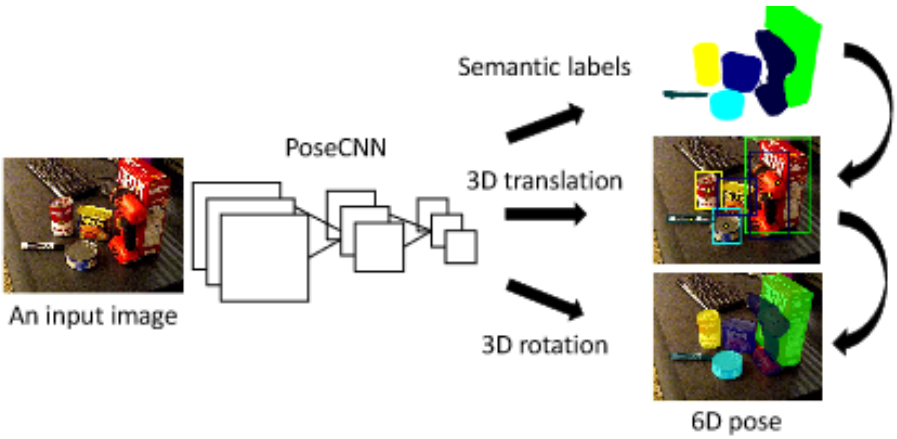


FIGURE 3.4 – Structure PoseCNN pour estimation de pose en 6D.

Pour la pose humaine, l'estimation en 6D est déconseillée pour des raisons de complexité donc, on va opter pour l'estimation de la pose du gardien de but en 3D.

Malgré que la kinect, qui fait partie des caméras RGB-D, est généralement utilisée dans un environnement interne, des expériences ont démontré qu'elle donne des résultats très acceptables en environnement externe [37], ce qui est le cas pour notre utilisation (stade à l'air libre).

En plus, la kinect est utilisée dans plusieurs applications qui nécessitent l'estimation de la pose humaine en robotique, sécurité et plus fréquemment en jeux vidéo (dance, sport, etc.). Donc, des ressources open source sont disponibles et donnent des résultats qui répondent au besoin de notre robot sportif, alors, on va opter pour une estimation de la pose humaine en utilisant une caméra RGB-D tout en respectant la distance maximale du capteur de profondeur et en évitant l'exposition directe au rayon du soleil.

3.2 Intelligence Artificielle

L'Intelligence Artificielle (IA) est une technologie qui fait désormais partie intégrante de la réalité. Elle constitue une aide précieuse pour l'homme dans des domaines comme l'automobile, la médecine ou la robotique.

3.2.1 Définition de l'Intelligence Artificielle Forte

L'intelligence artificielle forte fait référence à une machine capable de produire un comportement intelligent et aussi d'éprouver une impression d'une réelle conscience de soi, de vrais sentiments. La machine serait donc apte à comprendre ce qu'elle fait. De fait que l'intelligence est de source biologique, donc matérielle, les scientifiques ne voient pas de limites à pouvoir réaliser un jour une intelligence consciente sur un support matériel. Mais cela suscite de nombreux débats. Le fait qu'à l'heure actuelle, s'il n'y a pas d'ordinateurs ou de robots aussi intelligents que l'homme, ce n'est pas un problème de matériel,

mais de conception (d'après cela, nous pouvons donc considérer qu'il n'y a pas de limite fonctionnelle. Pour pouvoir déterminer si une machine peut être considérée comme ayant une intelligence artificielle forte, il faut que celle-ci ait réussi à passer le test de Turing [38].

3.2.2 Définition de l'Intelligence Artificielle Faible

L'intelligence artificielle est dite faible lorsqu'elle ne fait que reproduire un comportement spécifique, mais pas son fonctionnement. En d'autres termes, la machine ne comprend pas ce qu'elle fait.

L'IA faible vise essentiellement à reprendre le plus fidèlement possible, à l'aide d'un programme informatique, le résultat d'un comportement spécifique prévu à l'avance, sans aucune forme d'improvisation. C'est en fait un système qui imite un comportement intelligent dans un domaine précis.

La machine semble agir comme si elle était intelligente. Elle peut simuler le raisonnement, apprendre et résoudre des problèmes. Eliza, l'agent de dialogue simulant un échange en psychothérapie, est un cas typique de ces logiciels. Ce programme informatique de Joseph Weizenbaum, écrit entre 1964 et 1966, simule un psychothérapeute. Il reformule la plupart des affirmations du patient en questions à lui poser. Le fait qu'un humain parle à une machine sans qu'il ne s'en rende potentiellement compte a été considéré comme un critère de classification en Intelligence Artificielle [34].

3.2.3 Types d'Intelligence Artificielle

On peut distinguer trois grandes catégories de l'IA :

- IA symbolique ;
- IA par apprentissage automatique : parler des approches connexionnistes (réseau de neurones), celles par arbre et celles issues des statistiques ;
- IA évolutive : parler des algorithmes génétiques.

3.2.3.1 Intelligence Artificielle Symbolique

Elle est fondée sur la modélisation du raisonnement logique, sur la représentation et la manipulation de la connaissance par des symboles formels. L'approche top-down, s'appuyant sur l'intersection entre logiques philosophiques et mathématiques « Computational Logic ».

D'un point de vue pratique, elle s'appuie notamment sur des moteurs de règles et de faits, qui permettent notamment, mais pas seulement, de créer des systèmes experts ou de faire de la programmation par contraintes.

La terminologie de ce domaine intègre les notions de logiques d'ordre zéro, du premier ordre (dite calcul des prédicats) et second ordre qui sont liées à la complexité des problèmes logiques à résoudre au niveau d'objets et d'ensembles d'objets [40].

3.2.3.1.1 Les automates

Un automate est une machine logique qui réagit non seulement en fonction des variables d'entrées, mais aussi en fonction d'un contexte historique, en fonction de ce qui s'est

déjà passé. L'automate est le modèle générique de toute machine numérique automatique et mêmes les microprocesseurs et les ordinateurs sont des automates sophistiqués.

Principe Une fonction de sortie est égale à une combinaison logique des entrées et de variables internes témoignant du passé de l'automate.

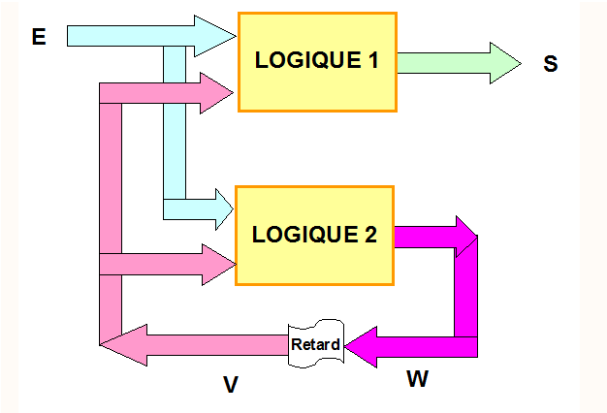


FIGURE 3.5 – Exemple d'illustration du fonctionnement d'un automate.

Illustration du fonctionnement La logique 1, tenant compte des entrées E de l'état interne V, calcule la sortie S.
La logique 2, tenant compte des entrées E et de l'état interne V, calcule un nouvel état interne W.
À la séquence suivante de nouvelles données d'entrée arrivent et l'état interne passe au suivant ; V prend la valeur de W.
Dans la pratique, tous ces étapes sont synchronisées au rythme d'une horloge qui cadence la prise en compte des données [41].

3.2.3.1.2 Les Systèmes Experts (Knowledge Based Expert Systems) (KBES)

« Le système expert est un outil de déductions rapide pour l'homme »

Les systèmes experts représentent des mécanisations du raisonnement pour obtenir des déductions et des conclusions. Ils s'appuient sur la connaissance du domaine, préalablement communiquée par un expert.

D'une liste de déclarations, le système expert cherchera toutes les déductions possibles et tentera d'aboutir à une conclusion [42].

Principe Un système expert utilise la connaissance correspondante à un domaine spécifique afin de fournir une performance comparable à l'expert humain. En général, les concepteurs de systèmes experts effectuent l'acquisition de connaissance grâce à un ou plusieurs interviews avec l'expert ou les experts du domaine. Les humains qui enrichissent le système avec leurs connaissances ne fournissent pas seulement leur connaissance théorique ou académique mais aussi des heuristiques qu'ils ont acquises grâce à l'utilisation de leurs connaissances.

L'architecture d'un système expert typique est constituée de plusieurs modules qui s'interagissent (Figure 3.6) :

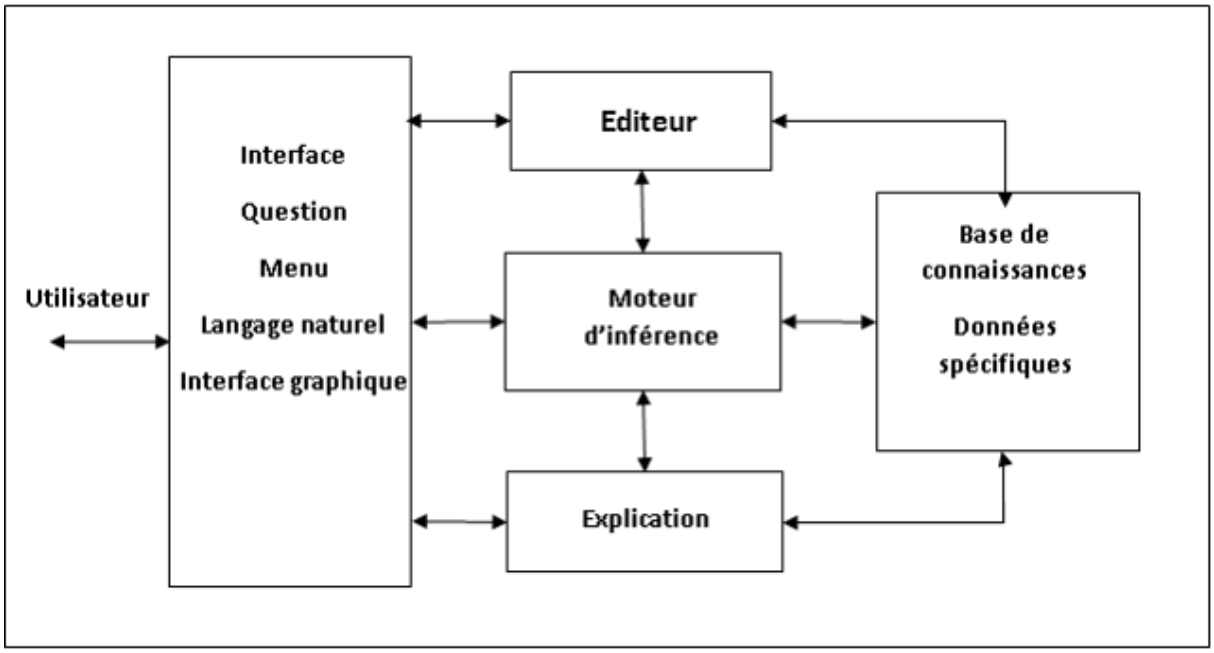


FIGURE 3.6 – Architecture d'un système expert.

- **L'interface utilisateur :** sert à simplifier la communication, elle peut utiliser la forme question-réponse, le menu, le langage naturel etc.
- **La base de connaissances :** contient les connaissances concernant la résolution du problème.
- **Le moteur d'inférence :** applique une stratégie de résolution en utilisant les connaissances et ceci pour en dériver une nouvelle information.
- **La base de faits :** contient les données spécifiques liées à l'application traitée. Elle peut contenir aussi les solutions intermédiaires ou les conclusions partielles trouvées lors de l'inférence.
- **Le module d'explication :** permet au système expert d'expliquer son raisonnement.
- **L'éditeur :** permet l'édition des connaissances dans la base.

Il est très important de remarquer que les connaissances et l'inférence sont représentées dans deux blocs séparés :

- Cette séparation permet d'utiliser un codage différent, cela nous permet par exemple d'utiliser le langage naturel pour représenter les connaissances (sous forme Si ... ALORS ... par exemple).
- Cette séparation permet au programmeur de se focaliser au codage des connaissances sans se soucier trop de la façon du codage du moteur d'inférence.
- Cette séparation permet aussi de modifier les connaissances sans avoir un effet sur le codage du moteur d'inférence.
- Cette séparation permet également de pouvoir tester plusieurs types d'inférence sur la même base de connaissances [43]

3.2.3.1.3 La Logique Floue « Fuzzy Logic »

Ensemble de principes mathématiques pour la représentation et la manipulation des connaissances en se basant sur des degrés d'appartenance compris dans [0,1] [44].

Contrairement a la logique binaire qui a deux états (0 et 1), la logique floue propose un spectre plus large et prend en compte les problèmes de la vie réelle qui peuvent présentés des états intermédiaires, donc, il faut être plus prés de la logique humain que de la logique numérique (Binaire) pour les résoudre.

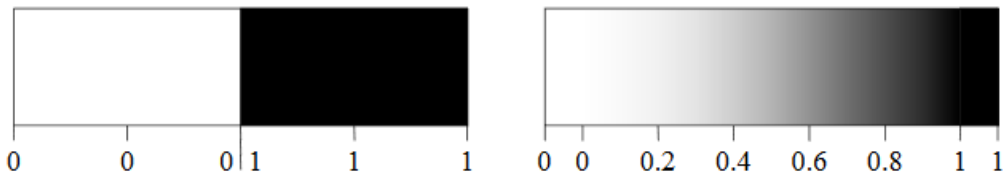


FIGURE 3.7 – Différence entre logique binaire et logique floue.

En logique binaire, une affirmation est vraie ou fausse ; son degré de vérité vaut 1 ou 0.

Valeur	Signification
1	Absolument vrai
0	Absolument faux

En logique floue, une affirmation est plus ou moins vraie (donc, plus ou moins fausse) ; son degré de vérité varie entre 0 et 1.

Valeur	Signification
0.0	Absolument faux
0.2	Plutôt faux
0.4	Quelque peu faux
0.6	Quelque peu vrai
0.8	Plutôt vrai
1.0	Absolument vrai

Principe Un système à logique floue comprend :

- Des variables d'e/s.
- Des labels qui représentent les valeurs floues de chaque variable.
- Des fonctions qui définissent le degré d'appartenance des valeurs des variables aux labels.

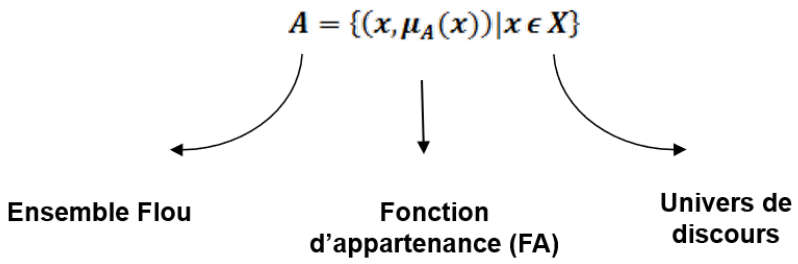
Une valeur mesurée peut appartenir à plusieurs labels, avec des degrés divers.

Variable floue Une variable floue (dite aussi linguistique) est un quintuplet x, T(x), X, G, M, où :

- X est le nom de la variable : ex. âge
- T(x) est l'ensemble de ses valeurs : ex. T (âge) = jeune, moins jeune, très jeune, ...

- X est l'univers de discours : ex. X = [0-100]
- G est une règle syntaxique qui définit l'ensemble T(x)
- M est une règle sémantique qui associe un ensemble flou avec chaque élément de T(x) ex. : M (jeune), (vieux)

Ensemble flou A sur un univers de discours X est un ensemble de paires ordonnées :



Tel que tout x en est membre à un degré $0 \leq d \leq 1$ avec A est entièrement caractérisés par sa fonction d'appartenance. Concrètement, A est un attribut qualitatif (valeur floue ou linguistique) que l'on associe avec les valeurs précises d'une variable numérique x.

Relation floue R est un ensemble flou défini sur le produit cartésien de deux univers de discours X et Y :

$$R = \{((x, y), \mu_R(x, y)) | (x, y) \in X > Y\}$$

Permet d'exprimer une relation qualitative entre deux variables numériques :

- x est près de y (nombres)
- x dépend de y (évènements)
- x et y se ressemblent (personnes ou objets)

Répartition floue Distribution des ensembles flous sur l'univers de discours. Exemple : Répartition floue des valeurs linguistiques « jeune », « ni jeune ni vieux », et « vieux »

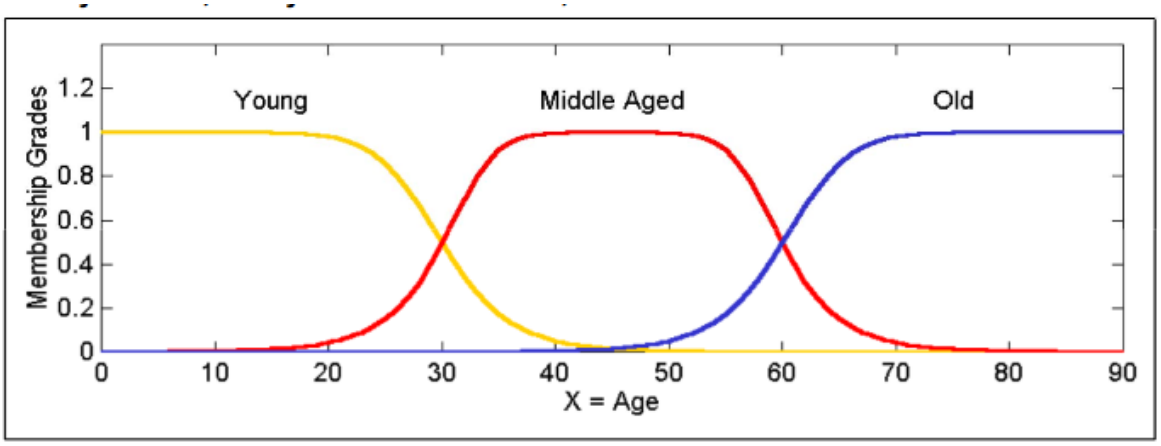


FIGURE 3.8 – Exemple répartition floue de l'âge.

Affirmation	Degré de vérité (fonction d'appartenance)
X	$\mu(X)$
Y	$\mu(Y)$
X ET Y	$\min(\mu(X), \mu(Y))$
X OU Y	$\max(\mu(X), \mu(Y))$
$\neg X$	1 - $\mu(X)$

Opération en logique floue Des opérations de logique peuvent être réalisées pour des variables floues, prenons comme exemple :

Règle d'inférence Syntaxe identique à celle de la logique binaire

$$Si < Condition > Alors < Consequence >$$

Où $< Condition >$ est une affirmation simple ou composée.

Exemple : Un régulateur de température dont les variables sont :

- **temp :** la température ambiante.
- **vac :** vitesse de rotation d'un ventilateur d'air chaud.
- **vaf :** vitesse de rotation d'un ventilateur d'air frais.

Les règles d'inférence peuvent être :

- **Si** temp est haute **alors** vac est nulle.
- **Si** temp est haute **alors** vaf est grande.
- **Si** temp est basse **alors** vac est grande.
- **Si** temp est basse **alors** vaf est nulle [44].

3.2.3.2 L'intelligence Artificielle par Apprentissage Automatique (Machine Learning)

L'apprentissage automatique est un sous-domaine de l'intelligence artificielle (IA). En général, l'objectif de l'apprentissage automatique est de comprendre la structure des données et de les intégrer dans des modèles qui peuvent être compris et utilisés.

Bien que l'apprentissage automatique soit un domaine de l'informatique, il diffère des approches informatiques traditionnelles. En effet dans cette dernière, les algorithmes sont des ensembles d'instructions explicitement programmées utilisées par les ordinateurs pour calculer ou résoudre des problèmes. Les algorithmes d'apprentissage automatique permettent aux ordinateurs de s'entraîner sur les entrées de données et utilisent l'analyse statistique pour produire des valeurs qui se situent dans une plage spécifique. Pour cette raison, l'apprentissage automatique facilite l'utilisation des ordinateurs dans la construction de modèles à partir de données d'échantillonnage afin d'automatiser les processus de prise de décision en fonction des données saisies.

Dans l'apprentissage automatique, les tâches sont généralement classées en grandes catégories. Ces catégories sont basées sur la façon dont l'apprentissage est reçu ou comment le feedback sur l'apprentissage est donné au système développé.

En tant que domaine informatique, l'apprentissage automatique est étroitement lié aux statistiques mathématiques ; disposer d'une connaissance approfondie des statistiques est donc utile pour comprendre et exploiter les algorithmes d'apprentissage automatique.

3.2.3.2.1 Arbre de Décision

Pour un usage général, les arbres de décision sont utilisés pour représenter visuellement les décisions et montrer ou éclairer la prise de décision. Lorsque vous travaillez avec l'apprentissage automatique et l'exploration de données, les arbres de décision sont utilisés comme modèle prédictif. Ces modèles cartographient les observations de données et tirent des conclusions sur la valeur cible des données en fonction de la valeur d'entrée.

Dans le modèle prédictif, les attributs des données qui sont déterminés par l'observation sont représentés par les branches, tandis que les conclusions, sur la valeur cible des données, sont représentées dans les feuilles. Lors de l'apprentissage d'un arbre, les données source sont divisées en sous-ensembles en fonction d'un test de valeur d'attribut, qui est répété récursivement sur chacun des sous-ensembles dérivés. Une fois que le sous-ensemble d'un nœud a la valeur équivalente à sa valeur cible, le processus récursif sera terminé. Regardons un exemple de diverses conditions qui peuvent déterminer si quelqu'un devrait lire à l'extérieur. Cela inclut les conditions météorologiques ainsi que les conditions de pression barométrique.

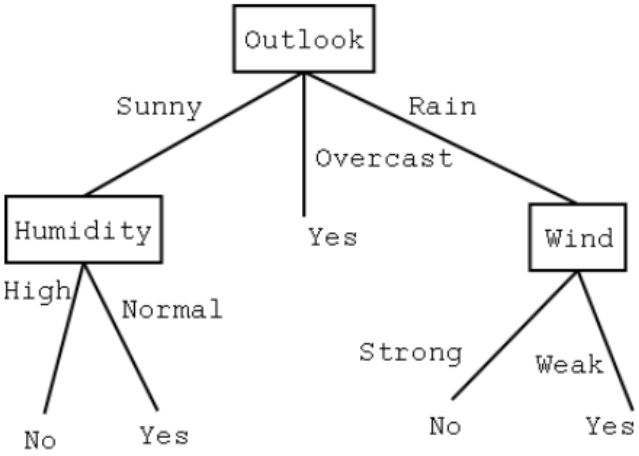


FIGURE 3.9 – Exemple d'un arbre.

3.2.3.2.2 L'Apprentissage Profond (Deep Learning)

Des tentatives profondes d'apprentissage pour imiter comment le cerveau humain traite des stimuli lumineux et sonores dans la vision et l'ouïe, sont à l'étude. Une architecture d'apprentissage en profondeur est inspirée par les réseaux neuronaux biologiques et se compose de plusieurs couches dans un réseau neuronal artificiel composé de matériel et d'un processeur GPU.

L'apprentissage en profondeur utilise une cascade de couches d'unités de traitement non linéaires afin d'extraire ou de transformer les caractéristiques (ou représentations) des données. La sortie d'une couche sert d'entrée de la couche suivante. Dans l'apprentissage en profondeur, les algorithmes peuvent être supervisés et servir à classer les données, ou non supervisés et à effectuer une analyse de modèle. Parmi les algorithmes d'apprentissage machine actuellement utilisés et développés, l'apprentissage en profondeur absorbe le plus de données et a été capable de battre les humains dans certaines tâches cognitives.

En raison de ces attributs, l'apprentissage en profondeur est devenu l'approche avec un potentiel significatif dans le monde de l'intelligence artificielle. La reconnaissance faciale par ordinateur et la reconnaissance vocale ont toutes les deux permis de réaliser des progrès significatifs grâce à des approches d'apprentissage approfondies. IBM Watson est un exemple bien connu d'un système qui exploite l'apprentissage en profondeur [45].

3.2.3.2.3 Intelligence Artificielle évolutive

Ce type d'intelligence fonctionne comme le cerveau humain et peut être catégorisée comme IA forte. Le principe est issu du domaine de la biologie, où plusieurs espèces arrivent à muter et s'adapter aux changements dans leur environnement.

L'IA est dite évolutive car, au fur et à mesure du temps le programme stocke dans sa mémoire toutes les informations ajoutées. Elle apprend à s'adapter. Cette IA peut être trouvée sous forme de boîte de dialogue (type cmd) ou encore physique (type robot) ; des vrais robots autonomes pouvant se modifier eux-mêmes et de s'adapter aux modifications de l'environnement. L'hypothèse de base est que ces « robots modernes » réuniront ces conditions nécessaires à l'apparition de la conscience, qui pourra se développer sur n'importe quel support physique [46].

Principe Premièrement, nous avons besoin d'une population d'algorithmes qui doit être générée aléatoirement dans un univers défini.

Deuxièmement, ces algorithmes vont devoir interagir avec l'univers dans lequel ceux-ci ont été construits. Ils vont devoir pouvoir prendre en compte différents éléments/sources d'informations disponibles dans l'univers, des variables mathématiques qui représentent les entrées. À partir de celles-ci, l'algorithme doit donner un résultat, les sorties. C'est les résultats de la fonction, prenant en compte les différentes variables.

Les algorithmes sont purement aléatoires car ils n'ont pas de but et n'ont pas été construits, réfléchis pour effectuer une tâche particulière. Il est donc primordial d'intégrer une notion d'objectif, de score.

Dès lors que ces trois éléments sont réunis, il ne reste plus qu'à appliquer le principe de la sélection naturelle. L'idée c'est de tester notre première population d'algorithmes (génération 1) et, à l'issue du test, on ne conserve que ceux ayant eu les meilleurs scores. A partir de ces algorithmes, une nouvelle population sera créée (génération 2). Afin que celle-ci soit de taille identique, de nouveaux algorithmes seront générés aléatoirement à partir de ceux que nous avons conservés. En appliquant ce double processus de sélection/mutation de manière répétée, on obtient assez rapidement des algorithmes très performants [47].

Notre robot a pour rôle de juger les réactions du gardien de buts (sa pose humaine) en se basant sur les exercices proposés pour qu'en suite il peut proposer les prochains exercices. Vu qu'on dispose de références fixes pour définir un comportement « parfait » du gardien donc, on peut utilisé une IA symbolique avec des règles fixées. En plus, on a besoin d'un système d'apprentissage pour le système ce qui nécessite l'utilisation du Machine Learning.

Conclusion, notre système de prise de décision est un système hybride entre symbolique et apprentissage automatique.

Concernant la fréquence d'appel du système de décision, les exercices sont en format de série d'exercices du même type, afin de confirmer le diagnostique de la pose humaine

du gardien. Donc, le besoin de prise de décision n'est pas en temps réel, ce qui favorise l'utilisation d'une machine d'états à la quelle on fait appel au moment nécessaire. Dans la Figure 3.10 on a un schéma qui représente le fonctionnement de la machine d'états.

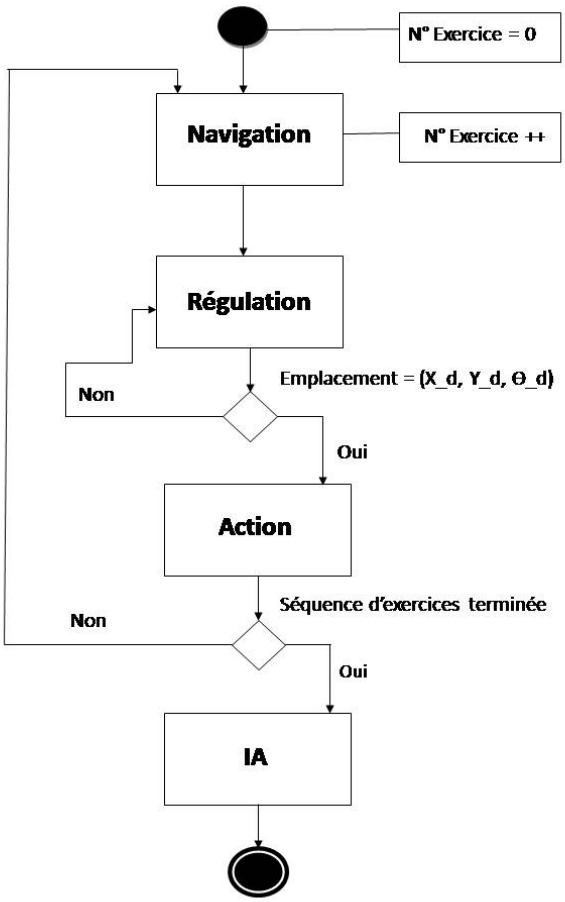


FIGURE 3.10 – Représentation de la machine d'états.

Chapitre 4

Architecture du Système

La modelisation du système nous permet d'établir un plan de travail du projet. En identifiant les parties essentielles du projet et les relations entre eux, nous pouvons modulariser le projet par spécialité. Cette modelisation nous permet d'avoir l'architecture globale du système.

L'architecture des robots mobiles se compose de quatre parties essentielles :

- Structure mécanique et la motricité ;
- Système de localisation ;
- Organes de sécurité ;
- Système de traitement des informations et gestion de tâches.

Cette architecture est généralement présentée sous forme de schéma blocs, chaque bloc représente une fonctionnalité du système, les blocs sont ensuite liés entre eux en se basant sur la logique de fonctionnement du système.

Notre projet et plus complexe qu'un simple robot mobile, car nous disposons d'une fonctionnalité de tir des ballons ainsi qu'une autre fonctionnalité de traitement des données externes du gardien. Alors, plus de blocs sont présentés dans l'architecture de notre système (Figure 4.1).

Les organes de sécurité ne sont pas pris en compte dans notre travail. Seuls les composants de sécurité de base sont mis en place dans notre système.

Notations :

(x, y) : Cordonnées du robot sur le terrain en prenant comme repère l'extrémité du terrain.

θ : L'angle d'orientation du robot sur le terrain (Figure 4.2).

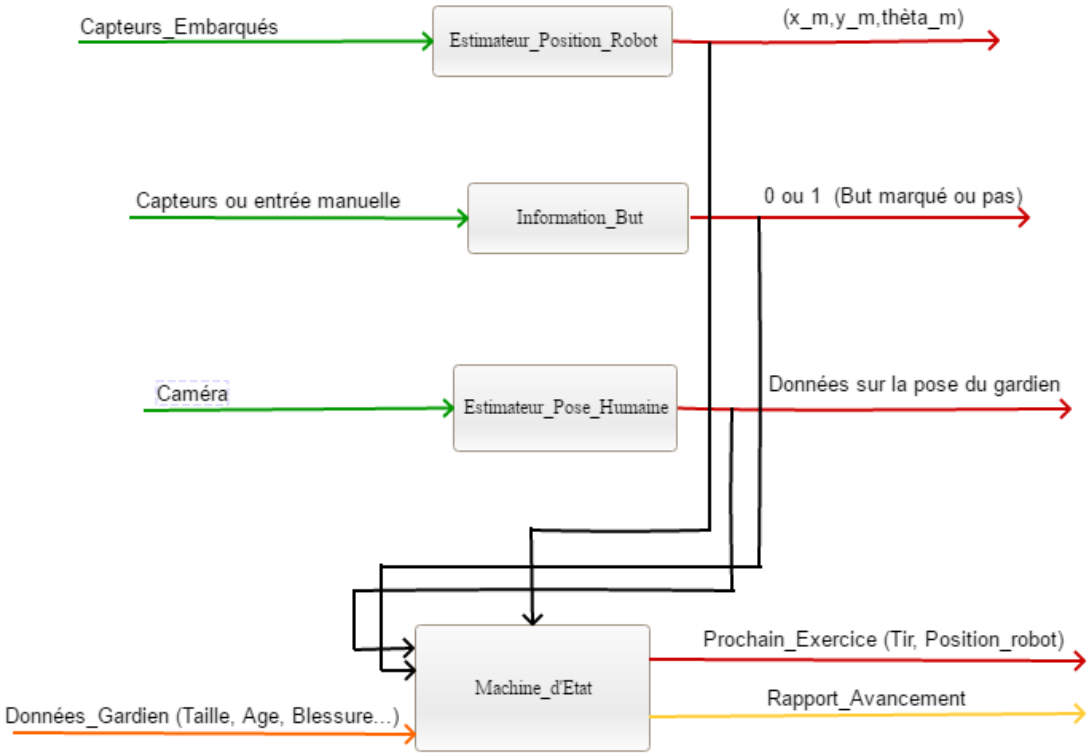


FIGURE 4.1 – Architecture du système.

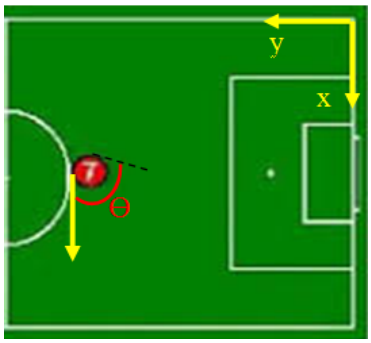


FIGURE 4.2 – Paramètres de localisation sur le terrain.

4.1 Structure Mécanique et Motricité

La partie mécanique de notre système est constituée de deux fonctionnalités : déplacement et tir (Figure 4.3).

Les blocs qui représentent ces fonctionnalités sont liés entre eux par un échange de données, par exemple : Le tir s'effectue **si** le déplacement est terminé. Donc, l'information de localisation qui est la sortie du bloc « ***Positionnement*** » est considérée comme entrée au bloc « ***Tir*** ».

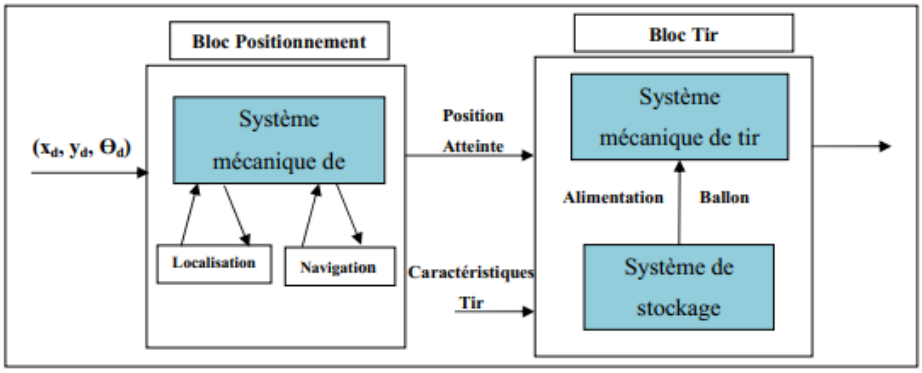


FIGURE 4.3 – Architecture mécanique.

4.2 Système de Localisation

Après avoir étudié plusieurs options de localisation utilisées pour les robots mobiles, nous choisissons d'utiliser la méthode de fusion de données (FDD) qui appartient à la catégorie de méthodes en boucle fermée.

Nous schématisons la structure de localisation avec le schéma bloc de la Figure 4.4

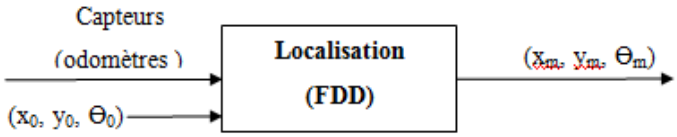


FIGURE 4.4 – Schéma blocde de la fonction de localisation.

La navigation du robot sur le terrain est basée sur sa localisation. L'architechture de navigation séléctionnée pour notre robot est une architechture lineaire (Figure 4.5) vu que notre système travaille dans environnement supposé statique.

FIGURE 4.5 – Architecture de navigation statique.

4.3 Système de Traitement des Informations et Gestion de Tâches

Le traitement de données est une fonctionnalité importante de notre système d'entrainement. En plus des données internes du robot mobile, nous avons un système d'accisition de pose humaine, ainsi qu'un système de prise de décision qui nous fournie l'évaluation des réactions du gardien (données éxternes).

L'architecture de l'estimateur de pose humaine et le systeme de prise de décision sont complexes, nous les détaillons dans la suite du projet.

4.4 Modélisation du Fonctionnement Global

Pour schématiser le fonctionnement du robot, on utilise une machine de Moore qui représente son comportement séquentiel. On a identifié à partir de l'analyse du système les variables suivantes :

Les états :

- Initialisation : Initialisation des paramètres et éléments du robot.
- Déplacement : Déplacement du robot pour se positionner à la position désirée P_d.
- Tir : Tir des ballons.
- Analyse : Analyse de la réaction du gardien de buts et prise de décision.

On code les états en binaire pour faciliter leurs notations.

Initialisation : 00 Déplacement : 01 Tir : 10 Analyse : 11

Les entrées :

- Exercice : Un nouvel exercice est en cours.
- Arrive : Le robot est arrivé à la position désirée P_d.
- Tir : Tir effectué.
- Ana_Term : Analyse terminée.

On représente les entrées par quatre bits avec le même ordre donné au-dessus.

Les sorties :

- Mot_Dep : Moteurs pour se déplacer fonctionnent.
- Mot_Tir : Moteurs pour tirer fonctionnent.

On représente les entrées par deux bits avec le même ordre donné au-dessus.

Le diagramme d'états du comportement du robot est représenté comme suit (Figure 4.6) :

Chaque état du système possède une architecture interne spécifique qui lui permet d'effectuer plusieurs opérations avec un certain ordre. En ce qui suit, nous détaillons toutes les architectures.

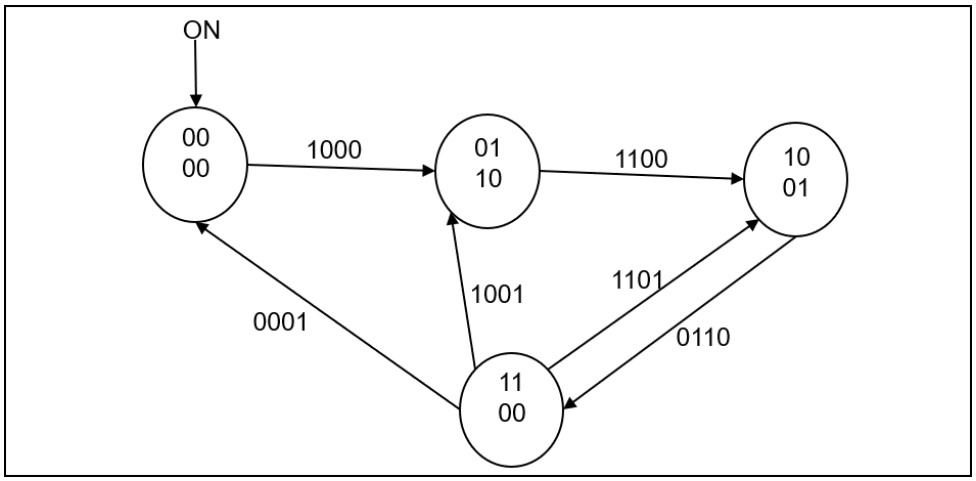


FIGURE 4.6 – Diagramme d'état du robot.

4.4.1 Initialisation

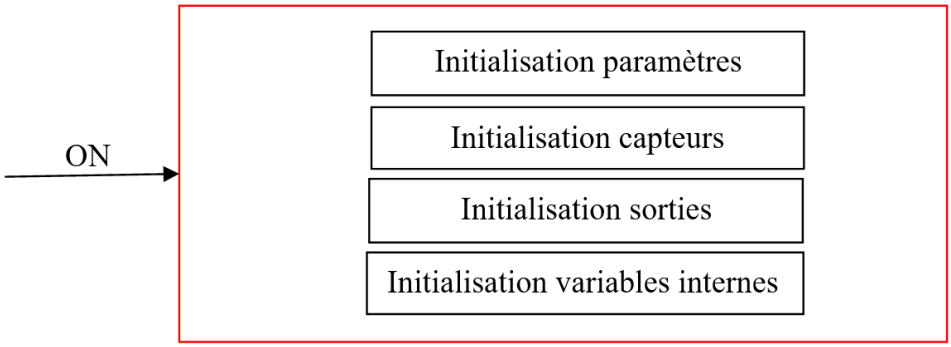


FIGURE 4.7 – Architecture interne de l'état initialisation.

4.4.2 Déplacement

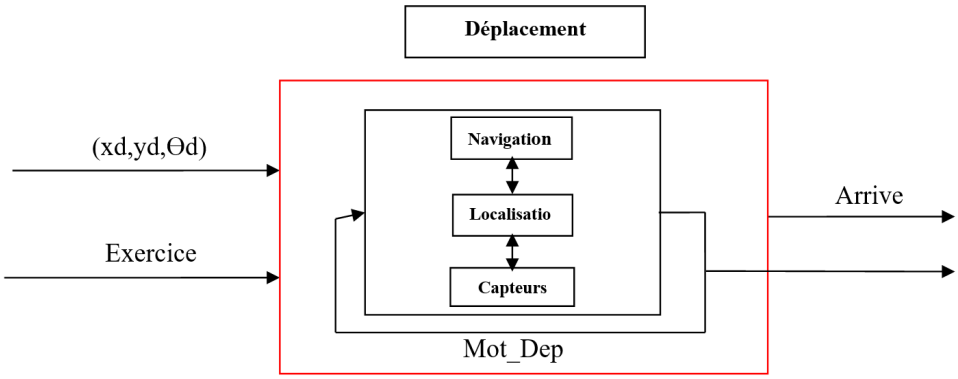


FIGURE 4.8 – Architecture interne de l'état déplacement.

4.4.3 Tir

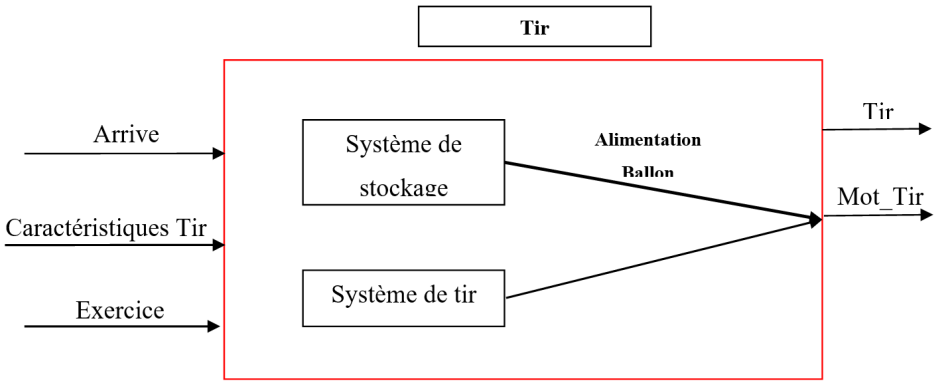


FIGURE 4.9 – Architecture interne de l'état tir.

4.4.4 Analyse

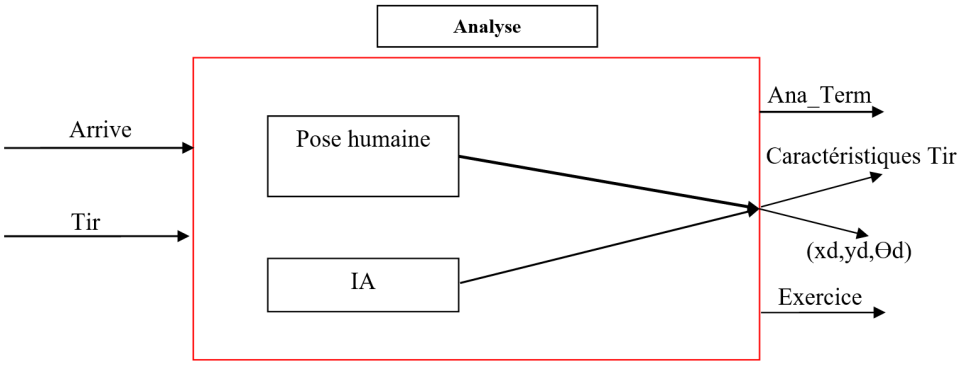


FIGURE 4.10 – Architecture interne de l'état analyse.

Chapitre 5

Conception du Robot

La conception mécanique était réalisée par SolidWorks, version 2016.

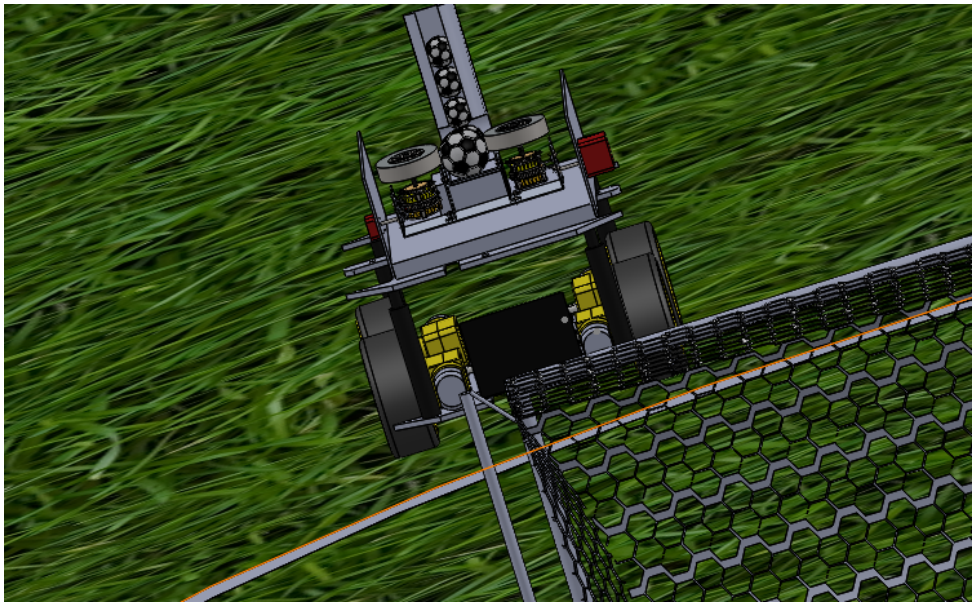


FIGURE 5.1 – Représentation du robot sur SolidWorks.

5.1 Conception Mécanique

5.1.1 La Base Mobile du Robot

La partie supérieure du robot dédié pour le tir des ballons est volumineuse et lourde par conséquent, la surface supérieure doit être spacieuse et solide. En même temps, le robot doit rester en équilibre pour assurer un bon déplacement sur le terrain. Donc, nous avons opté pour un modèle de base mobile à quatre roues « Rover » (Figure 5.2).

5.1.2 Système de Tir des Ballons

A fin d'assurer une variété d'exercices d'entrainement par notre robot, il faut que le système de tir ait la capacité de générer différents types de trajectoires pour le ballon. Pour cela, cette partie du robot ne doit pas être totalement fixe. A l'aide de deux moteurs

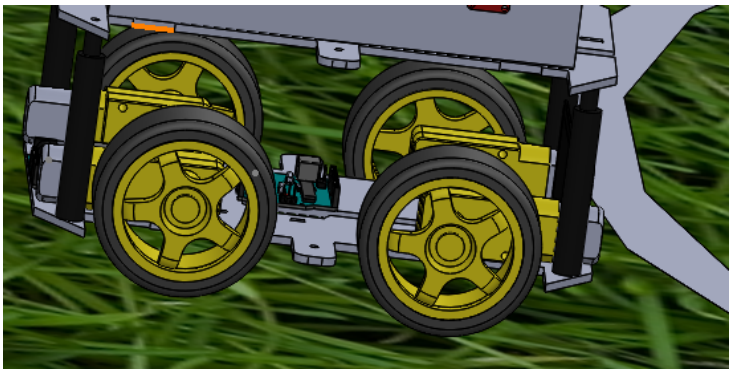


FIGURE 5.2 – Modèle de la base du robot.

(Figure 5.3), un angle de tir de 30°maximum peut être réalisé afin de viser à n'importe quel point sur la hauteur des buts.

D'un autre coté, le tir se fait à l'aide de deux roues tournantes avec des vitesses différentes, ce qui permet de varier la trajectoire de sortie tout en variant les vitesses des deux moteurs séparément (Figure 5.3).

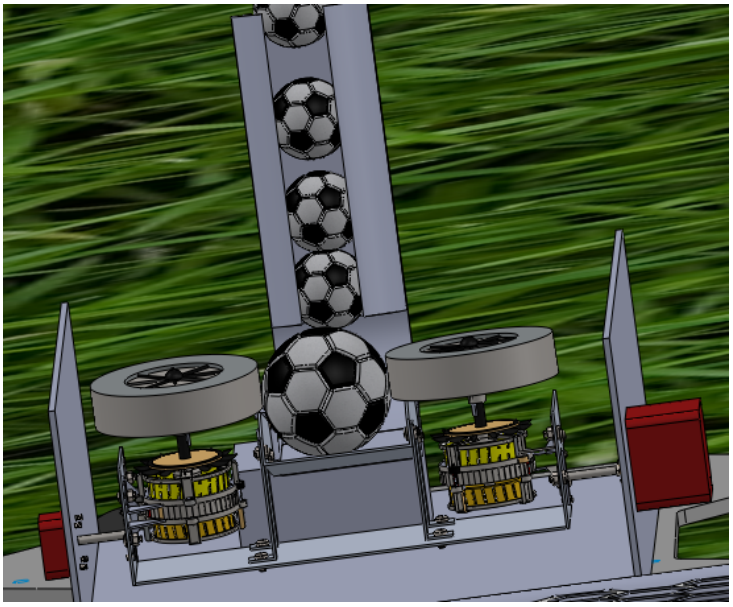


FIGURE 5.3 – Partie supérieure de tir des ballons.

5.1.3 Système de Stockage

Pour réaliser plusieurs tirs rapide et automatisé, et aussi pour éviter au mieux l'assistance humaine, un système de stockage des ballons est nécessaire. Nous avons conçu un dispositif cylindrique avec un diamètre de 28 cm, qui permet d'empiler les ballons de foot (diamètre 22 cm) verticalement.

Un seul ballon doit passer à la fois et à la demande à la partie de tir, pour cela, un système de blocage commandable à l'aide d'un moteur est implémenté (Figure 5.4).

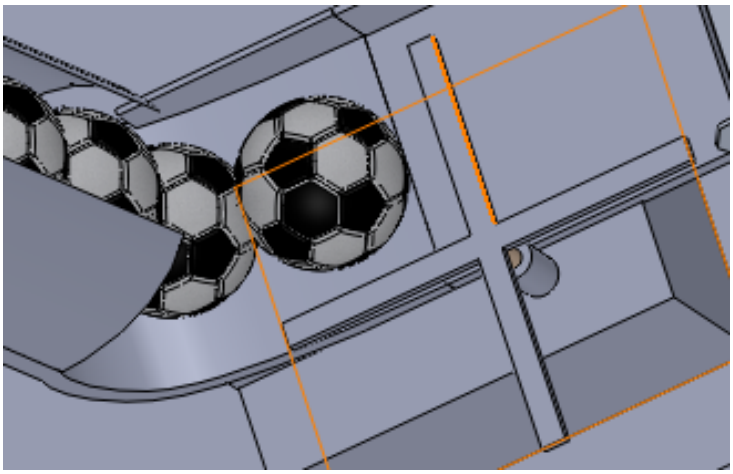


FIGURE 5.4 – Système de blocage des ballons.

5.1.4 Assemblage du Robot

Les trois parties précédentes sont assemblées pour former notre robot tireur de ballons (Figures 5.5 et 5.6).

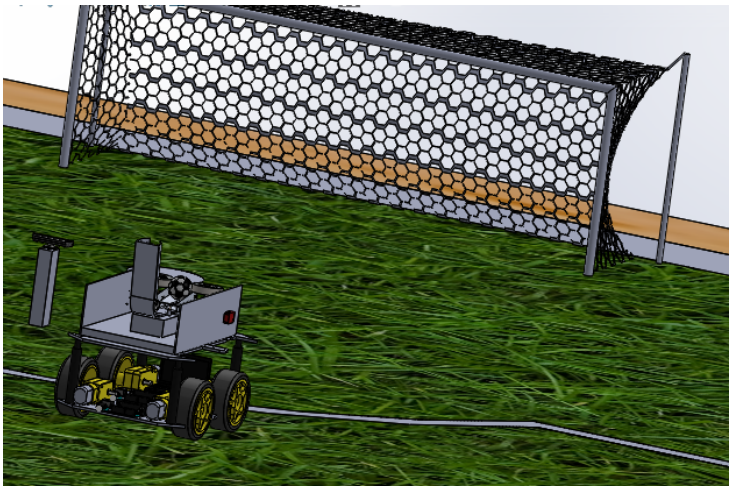


FIGURE 5.5 – Vue de derrière.

FIGURE 5.6 – Vue de face du robot.

5.1.5 Emplacement de la Caméra

Pour la partie analyse du projet, nous avons besoin d'une kinect, comme déjà conclue dans le chapitre précédant. La position et le modèle de la caméra (kinect) sont choisis en fonction des calculs faits et basés sur le champ de vision vertical et horizontal du modèle de la Kinect.

5.1.5.1 Le Plan Vertical

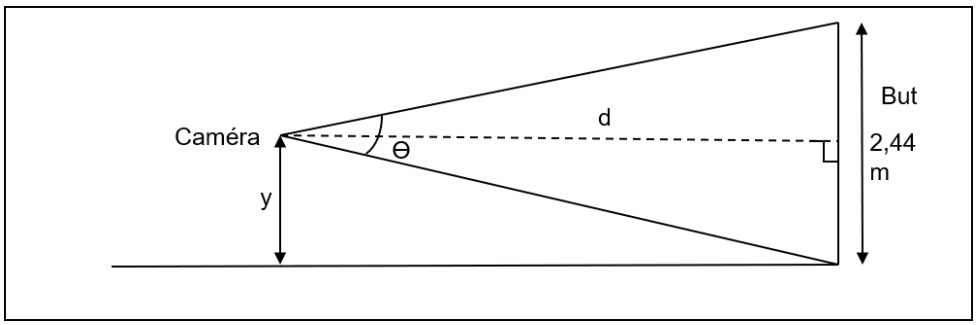


FIGURE 5.7 – Position vectirale de la caméra.

En considérant l'angle de tir comme :

$$tan(\theta/2) = y/d$$

Nous déterminons, à partir de quelle distance minimale du but (d_{min}) et à quelle hauteur minimale (y_{min}), nous pouvons positionner la camera afin de couvrir toute la zone de présence du gardien c.-à-d. tout le but. Nous supposons la distance d'éloignement du but et nous cherchons la hauteur nécessaire pour positionner la caméra. Nous présentons les résultats sous forme de tableau pour faciliter la comparaison entre les deux modèles de kinect disponibles sur le marché (Table 5.1) :

	Kinect 1 (Xbox 360 Kinect Sensor)	Kinect 2 (Xbox ONE Durango Sensor)
Champ de vision vertical	43°	60°
d_{min}	5.5 m	4m
y_{min}	0.27 m	0m

TABLE 5.1 – Positions verticales de la camera.

5.1.5.2 Le Plan Horizontal

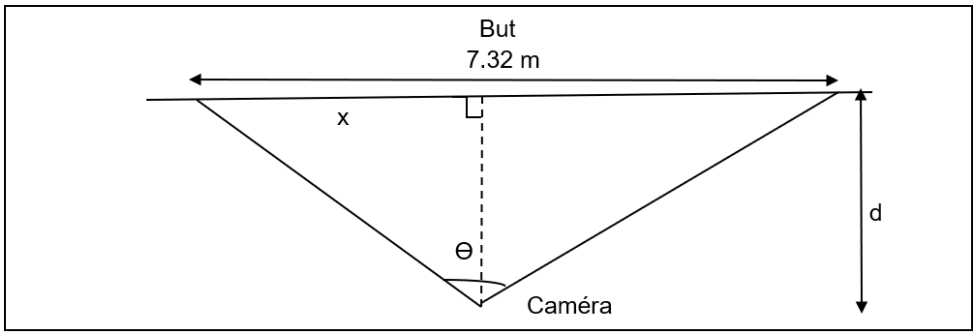


FIGURE 5.8 – Position horizontale de la caméra.

Nous adoptons la même méthode utilisée pour le plan vertical, avec :

$$tan(\theta/2) = x/d$$

Nous trouverons les résultats suivants (Table 5.2) :

	Kinect 1 (Xbox 360 Kinect Sensor)	Kinect 2 (Xbox ONE Durango Sensor)
Champ de vision vertical	57.5°	70°
d_{min}	6.67 m	5.5m

TABLE 5.2 – Positions horizontales de la camera.

Nous concluons ainsi que la Kinect modèle 2 est plus adéquate, car il est claire que nous pouvons la placer sur n'importe quelle hauteur avec une distance minimale de 5.5m ce qui est tout à fait acceptable en comparant avec la ligne de 5.5m tracée sur le terrain (Figure 5.9).

5.2 Dimensionnement

5.2.1 Partie Tir

Dans cette partie, on cherche à positionner le ballon à un point bien déterminé défini par les cordonnées (xd, yd) dans un plan. Pour éviter l'étude standard en 3D, qui est assez complexe, on divise l'étude sur deux plans indépendants en 2D :

Plan 01 : permet la détermination de l'angle de tir (α) qui va nous assurer la hauteur désirée (axe vertical) du ballon (y_d) (Figure 4.10).

Plan 02 : permet la détermination des vitesses des moteurs de tir qui vont nous assurer la position désirée sur l'axe horizontal du ballon au niveau des buts (x_d) (Figure 5.11).

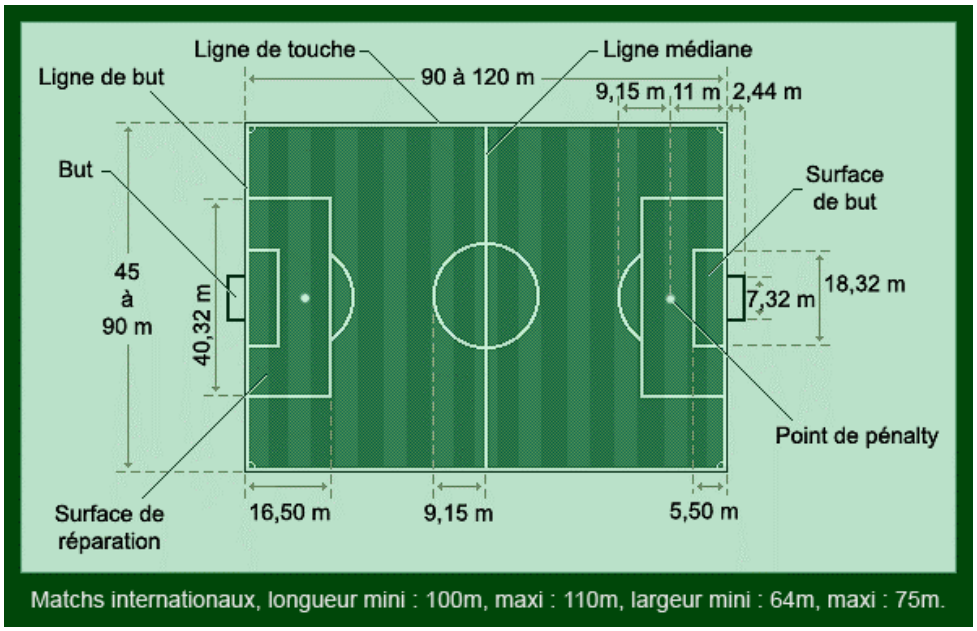


FIGURE 5.9 – Dimensions terrain de football.

FIGURE 5.10 – Vue du plan 01.

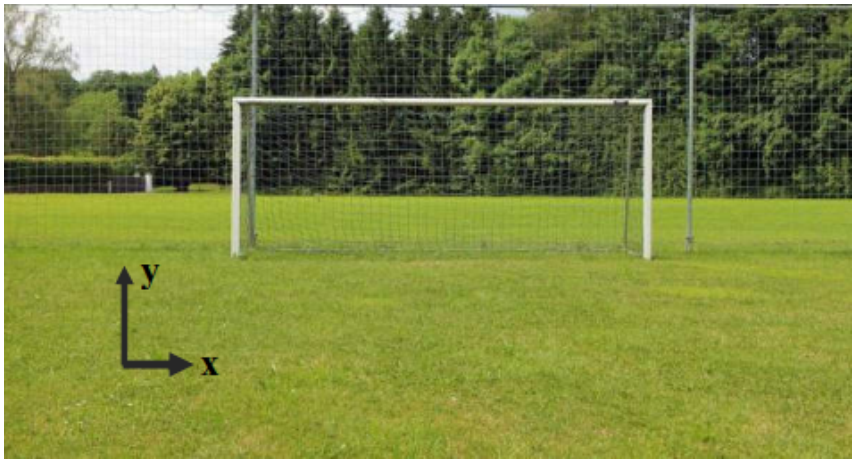


FIGURE 5.11 – Vue du plan 02.

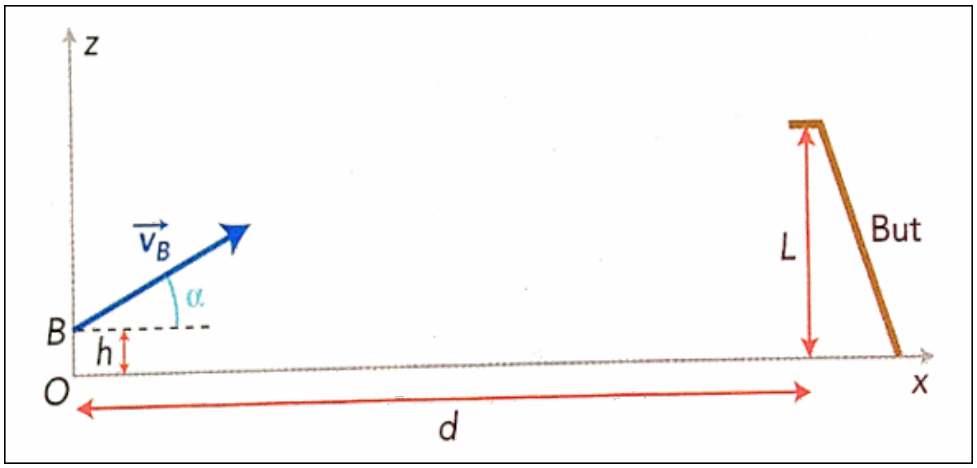


FIGURE 5.12 – Représentation du problème de projectile.

5.2.1.1 Angle de Tir

D'après la conception mécanique du robot, l'angle de tir (α) est variable par un système automatisé. On s'inspire des lois de la physique d'un système de projectile pour déterminer cet angle de tir à partir de la hauteur désirée y.

Pour atteindre une cible à distance (d) et une altitude (z) lorsque le projectile est lancé du point de coordonnées (0 ; 0) avec une vitesse (v_B), les valeurs de l'angle de portée (α) se calculent ainsi :

$$\alpha = \arctan\left(\frac{v_B^2 \pm \sqrt{v_B^4 - g(gd^2 + 2yv_B^2)}}{2}\right)$$

Pour chaque exercice, les valeurs de dd, vBd et zd sont données, tel que :

- d : la position désirée du robot lors du tir.
- v_B : la vitesse du ballon désirée, prédéfinie pour chaque exercice.
- y : la hauteur du ballon au niveau du but, prédéfinie pour chaque exercice.
- g : la gravité.

Afin de changer l'angle d'arrivée au but pour à la fois varier le type d'exercices et s'approcher des tirs effectués par des vrais joueurs de foot, on varie la distance d tout en gardant l'altitude désirée y (Figure 5.13).

5.2.1.2 Vitesses de Tir (v_d, v_g)

La variation des vitesses des deux moteurs de tir génère un changement de la direction de sortie du ballon (Figure **??**). Par exemple, si on fait tourner les deux moteurs à la même vitesse ($V_d = V_g = V$) le ballon va sortir avec un angle d'orientation horizontal nul ($\beta = 0°$).

Pour avoir le vecteur résultant de la somme des deux vecteurs, on applique la somme algébrique des deux vecteurs en utilisation les composants cartésiennes comme suit :

$$v = \overrightarrow{v_d} + \overrightarrow{v_d} = \begin{pmatrix} a \\ b \end{pmatrix} + \begin{pmatrix} c \\ d \end{pmatrix} = \begin{pmatrix} a+c \\ b+d \end{pmatrix}$$

Les combinaisons de deux vecteurs qui donnent le même vecteur résultant sont infinies, mais ils ne sont pas toutes réalisables physiquement (limitation des moteurs électriques).

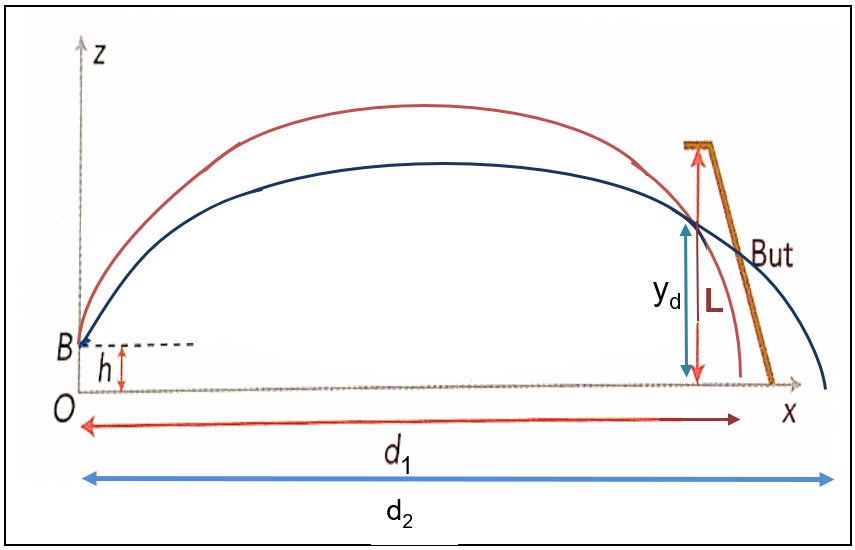


FIGURE 5.13 – Variation de l'angle d'arrivée.

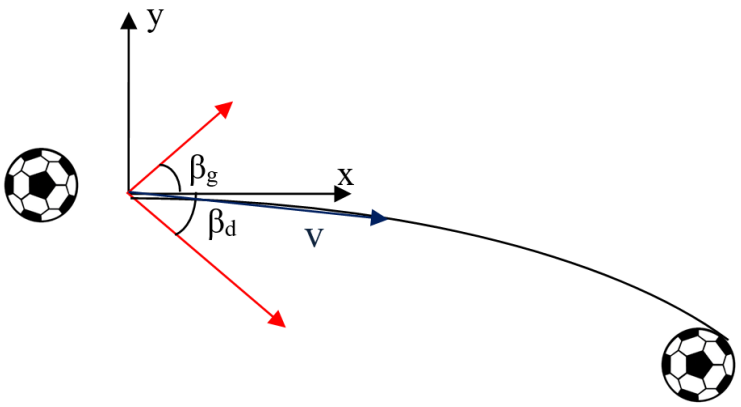


FIGURE 5.14 – Direction du ballon en fonction des vecteurs de vitesses.

Pour chaque exercice on doit prédéfinir les caractéristiques du tir c.à.d. la vitesse et la direction de sortie du ballon, donc, le vecteur $(\overrightarrow{v})$ est connu mais il faut le constituer avec la combinaison des vecteurs vitesse des moteurs droit et gauche $(\overrightarrow{v_g})$ et $(\overrightarrow{v_d})$.

Afin d'extraire les deux vecteurs dont l'addition donne le vecteur voulu (Figure **??**), on suppose une valeur réalisable pour l'un d'eux et on calcul le 2ème vecteur par les équations suivantes :

$$V_{1x} = R_x - V_{2x}$$

$$V_{1y} = R_y - V_{2y}$$

$$\theta_1 = \tan^{-1}(\frac{V_{1y}}{V_{1x}})$$

Au final, on valide la combinaison si la valeur trouvée est réalisable et optimale. Dans le cas contraire, on redéfini une nouvelle valeur du 1er vecteur de vitesse.

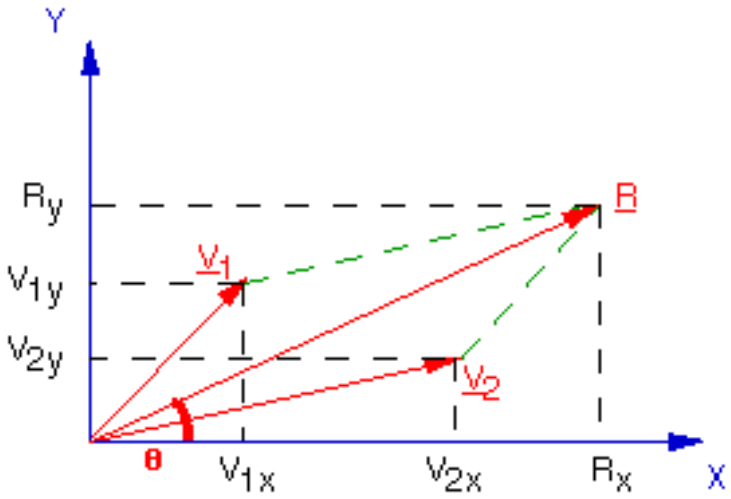


FIGURE 5.15 – Addition de deux vecteurs par l'addition des composantes.

5.2.2 Partie Déplacement

Afin d'assurer le déplacement optimal du robot sur la pelouse, il faut que les moteurs misent en place peuvent fournir suffisamment de vitesse et de couple. Donc, des calculs de puissance sont primordiaux avant de sélectionner les moteurs.

Bilan des forces Nous étudions le système mécanique représenté sur la Figure **??**.

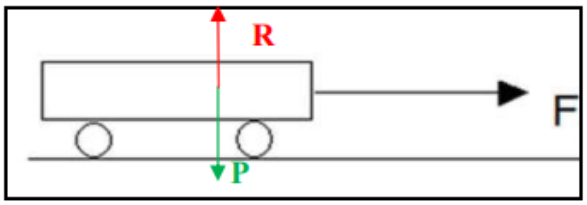


FIGURE 5.16 – Bilan des forces.

On sait que : $\sum \overrightarrow{F} = m\overrightarrow{a}$
Avec : R = - P

Bilan du travail des forces Le travail de chaque force externe est défini comme suit :

$$W_p = 0 \qquad W_R = 0 \qquad W_F = F \times d$$

Le travail du poids et du sol sont nuls car la direction du vecteur de force est perpendiculaire au sens de déplacement (d).

On a : $\Delta E_c = \sum W_f ext$
D'autre part : $E_c = \frac{1}{2}m(v(t) - v(0))^2$
Donc : $\sum W_f ext = F \times d \implies F = \frac{\frac{1}{2}m(v(t)-v(0))^2}{d}$

Energie cinétique

La formule du couple est donnée par : $C_m = F \times r$ | r : Rayon de la roue

Couple du moteur La valeur du couple trouvée doit être divisée par deux vu que l'on utilise 2 moteurs identiques pour le déplacement.

On a : $P_{moteur} = C_m \times \omega(t)$ | $\omega(t)$: vitesse angulaire désirée

Puissance du moteur

Application numérique En se basant sur les formules au dessus et les données caractéristiques du robot, on peut avoir les valeurs pratiques suivantes (Table 5.3) :

Nom	Symbole	Valeur
Distance parcourue	d	1(m)
Rayon de la roue	r	0.1(m)
Vitesse angulaire désirée	ω	1 (rad/s)
Vitesse linéaire désirée	v	0.1 (m/s)
Masse du robot	m	15 (kg)

TABLE 5.3 – Données numérique d'une application.

Nous trouvons les résultats suivants :

$$\begin{aligned} F &= 0.075 \text{ N} \\ C_m &= 0.00075 \text{ Nm} \\ P_m &= 5.625 \times 10^{-4} \text{ J} \end{aligned}$$

Ces résultats sont valables pour les deux moteurs, on divise sur 2 par la suite.

5.2.3 Choix du Matériel

En se basant sur les études faites pour les deux parties tir et déplacement, nous choisissons les moteurs et composants suivants (Table 5.4) :

Fonctionnalité	Composant	Référence	Caractéristiques
Tir	Moteurs CC (2)	Pololu 12VDC, 350rpm 30 :1 Metal Gearmotor	• Moteur à Engrenage CC • Tension nominale : 12 V • RPM Sans Charge : 350 • Courant de Démarrage : 5A • Couple de Décrochage : 110 oz-in • Diamètre de l'Axe : 6 mm
Tir	Moteur pas à pas (1)	Moteur Pas-à-Pas 2.8 V 1.68A 4.4 kg-cm RepRap SOYO	• Tension : 2,8 Vcc • Résolution de 1,8 degré/pas • Couple : Retenue 4,4 (kg-cm)/ détente 250 (g-cm) • Courant : 1,68 A • Précision de ±5 %

Tir	Pilote moteur pas à pas (1)	Driver de moteur pas à pas bipolaire simple 10-30V 4A	• Courant disponible par bobine, au maximum : 4 A • Tension d'alimentation min : 10 V DC • Tension d'alimentation max : 30 V DC • Consommation de courant min : 50 mA • Courant maximal : 7 A • Courant minimum de consommation (port VINT) : 500 μA • Consommation de courant max (port VINT) : 1 mA • Consommation électrique au repos : 200 mW
Tir	Pilote moteurs CC (1)	Cytron 10A 5-30V Dual Channel DC Motor Driver	• Pilote de moteur avec commande bi-directionnelle pour 2 moteurs CC à balais • Prend en charge les opérations MLI à opposition de phase verrouillée et à signe et grandeur • Courant maximum : 10 A en continu et 30 A en crête (10 secondes) pour chaque canal • Plage de tension du moteur : de 5 à 30 V
Déplacement	Carte de contrôle navigation (pilote automatique) (1)	Pixhawk PX4 2.4.8	• Processeur avancé Cortex® M4 ARM 32 bits utilisant NuttX RTOS • Capteurs accéléromètre/magnétomètre 14 bits ST Micro LSM303D • Langage de script Lua et le comportement de vol 14 sorties MID/servomoteur • Inclut module GPS uBlox NEO-7 avec boussole **(Annexe A)**

Déplacement	Moteurs (Avec encodeurs) (2)	Pololu 12V, 100 :1 Gear Motor w/ 64 CPR Encoder	• Motoréducteur CC • Tension nominale : 12V • Vitesse à vide : 100 RPM • Courant de décrochage : 5A • Couple de décrochage : 220 oz-in • Diamètre de l'arbre : 6 mm • 64 encodeur rotatif de RCP inclus
Déplacement et tir	Régulateur 5V (1)	L78S05CV	• Courant de sortie : 2a • Tensions de sortie 5 v • Protection contre la surcharge thermique • Protection de court circuit • Protection à transistor de sortie
Déplacement et tir	Régulateur 12V (1)	TO-3 / IP3R18K-12	• Sortie fixe : 5 V ou 12 V • Courant de sortie 5 A • Protection : limitation de courant interne/shutdown thermique/protection de zone d'utilisation sécurisée • Régulation en ligne 0,01 % • Régulation de charge 0,5 % • Boîtier TO3
Déplacement et tir	Batterie (2)	14.8V, 4000mAh, 40C, 4S LiPo Battery	• Tension nominale : 14.8V • Chimie : LiPo Capacité : 4 Ah • Décharge continue : 160 A Type de connecteur : XT-60

Déplacement et tir	Carte de commande	RP3 B+	• CPU 64 bit quad core ARM Cortex-A53 intégré et cadencé à 1,4 GHz • Contrôleur graphique Broadcom Videocore IV • 1 Go de mémoire. • 1 micro SD / SDHC/4 sorties USB 2.0/1 port RJ45 (Ethernet 10/100/300 Mbps)/1 port HDMI/1 audio Jack 3,5 mm • GPIO 40 broches + 4 nouvelles broches pour le PoE • Compatible Wifi 802.11 b/g/n/ac double bande 2,4 Ghz et 5 Ghz et Bluetooth 4.2 (Bluetooth Classique et LE) **(Annexe B)**

TABLE 5.4: Caractéristiques des composants sélectionnés.

Chapitre 6

Simulation

6.1 Simulation du Rover

La simulation permet de tester en toute sécurité le code et les paramètres expérimentaux et peut aider à s'entraîner avant de passer à la pratique. . Il est bien connu que les erreurs en simulation sont beaucoup moins couteuses que sur le terrain.

Afin de simuler la base roulante du robot et tester l'opération de navigation, on utilise un simulateur open source d'actualité nommé « *Ardupilot-Rover2* ».

6.1.1 Ardupilot

Ardupilot est un autopilote open source, il est développé à l'origine par des passionnés pour contrôler des modèles réduits d'avions et de rovers.C'est ensuite devenu un autopilote utilisé par des industries, organisations de recherche et des amateurs.

Plusieurs types de véhicule disponible, nous citons par exemple : Copter - Avion - Rover - Sous-marin.

En tant que projet open source, il évolue et se développe constamment. L'équipe de développement « DEV » utilise la communauté et les partenaires commerciaux pourlui ajouter des fonctionnalités à Ardupilot profitant à tous. Bien qu'Ardupilot ne fabrique aucun matériel, le micrologiciel Ardupilot fonctionne sur de nombreuses cartes différentes (matériel) pour contrôler des véhicules sans pilote de tous types. Couplés au logiciel de contrôle au sol, les véhicules sans pilote exécutent des missions entièrement scriptées avec des fonctionnalités avancées, notamment une communication en temps réel avec les autres opérateurs.

Cet outil possède la plus grande communauté dédiée à aider les utilisateurs avec des questions, des problèmes et des solutions [48].

Ardupilot nous permet de simuler avec grande précision la commande du véhicule en utilisant un grand choix de logiciels de simulations, par exemple : Gazebo, CRRCsim et SITL « Software In The Loop » (Figure 50). Ces simulateurs peuvent fonctionner sur plusieurs systèmes d'exploitation comme Windows, Linux et Rasbian.

Après la simulation, on peut passer à l'implémentation en utilisation des cartes de commande compatibles avec Ardupilot. On peut citer Pixhawk qui est une carte de contrôle de pilotage (autopilote) de haut de gamme avec plusieurs connexions avec des capteurs

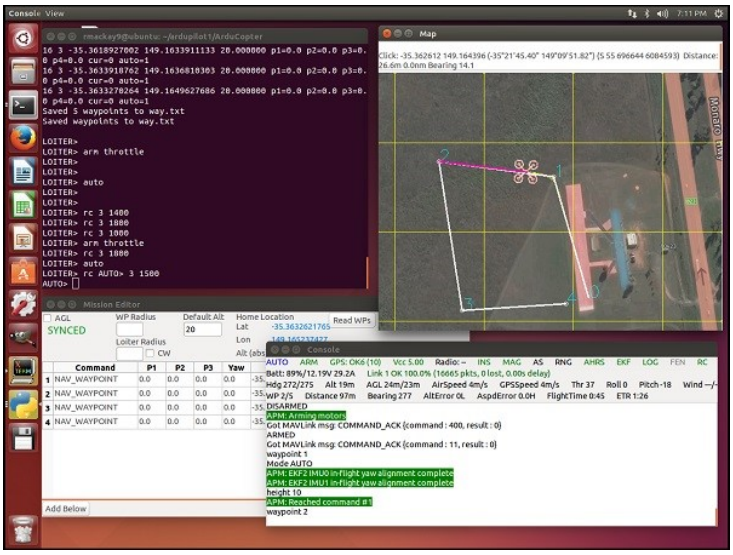


FIGURE 6.1 – SITL sur Linux.

FIGURE 6.2 – Carte de pilotage Pixhawk.

et commandes d'actionneurs (Figure 51). Cette carte peut communiquer avec une carte Raspberry Pi ou un autre support du logiciel d'Ardupilot à l'aide protocoles normalisés par exemple : MAVLink.

6.1.2 Simulation du projet

Pour notre projet, nous avons utilisé le simulateur SITL sur une machine virtuelle à système d'exploitation Linux version 16.4.

6.1.2.1 Lancement de la simulation

Le lancement de la console de commande du simulateur, la carte de simulation et le choix du vehicule se fait avec le terminal de commande (Figure 6.3).

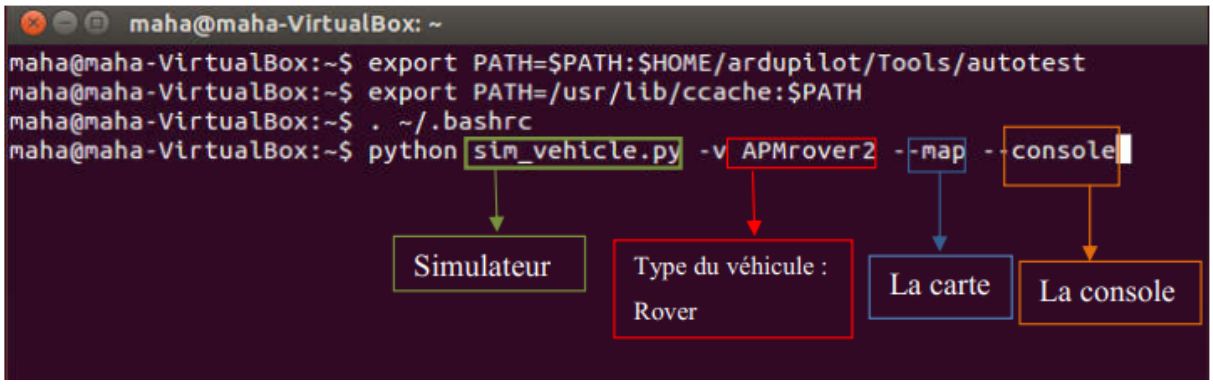


FIGURE 6.3 – Lancement de la simulation.

FIGURE 6.4 – La carte et les consoles de commande du simulateur .

6.1.2.2 Paramétrage du Simulateur

Nous modifions les paramètres relatifs à la simulation en fonction de notre environnement de travail, par exemple :

- La vitesse de déplacement ;
- La force du vent ;
- La nature du terrain de déplacement ;
- Les vibrations.

6.1.2.3 Tests du Rover

Nous passons aux tests de navigation du rover. Les moteurs du rover sont bloqués par défaut pour éviter que le robot démarre au lancement du programme. Donc, la première étape à faire est de préparer les moteurs en les débloquant « Arm throttle ». Plusieurs modes sont disponibles sur Ardupilot :

- Auto : mode automatique, le véhicule ce déplace avec un itinéraire prédéfini ;
- Guided : mode guidé, le véhicule ce déplace sur demande en précisant la position désirée à chaque fois ;
- Manual : mode manuel, le véhicule est stable et prêt à changer de mode ;
- RTL : revenir au point de lancement « *Return To Launch* » ;
- HOLD : pause.

Nous pouvons commander les déplacements du véhicule manuellement en utilisant la carte, nous cliquons sur la position cible et choisissons l'opération à faire (Figure 6.5). La deuxième option de commande qui nous intéresse plus est les lignes de commande de bas niveau avec la console de commande. Pour cela il faut que le rover soit en mode « *Guided* » (Figure 6.6).

FIGURE 6.5 – Commande du déplacement en utilisant la carte.

```
> arm throttle                    // Préparation des moteurs

MANUAL > mode guided              // Changement du mode de Manual à Guided

GUIDED > guided 10 20 0           // Déplacement vers (10,20) avec attitude=0

GUIDED > mode hold                // Pause
```

FIGURE 6.6 – Déplacement par commandes bas niveau.

6.1.2.4 Planification d'une mission

Nous pouvons planifier une mission de déplacement avec plusieurs points d'arrêt "waypoints". Le Rover suit l'itinéraire défini en fessant des arrêts un temps bien déterminé dans chaque waypoint. Ces points d'arrêts peuvent être définis soit manuellement sur la carte (Figure **) ou en utilisant les lignes de code avec la console de commande. Une autre méthode plus adéquate à la programmation des missions est d'accorder Ardupilot avec le logiciel "Mission Planner" qui est spécialisé dans la planification des circuits (Figure 6.7).

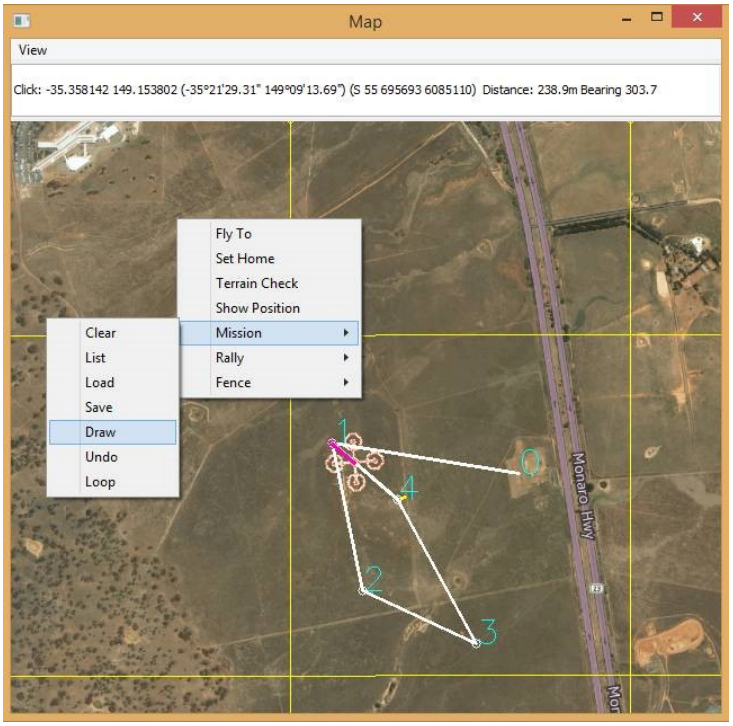


FIGURE 6.7 – Planification d'une mission sur la carte.

6.1.3 Commande haut niveau du déplacement

Le pilote automatique Ardupilot peut être contrôlé par un programme de haut niveau écrit avec un langage évolué comme, par exemple, Python.

L'étape de déplacement fait partie du diagramme d'état du système (Chapitre 4.3). Vu que le diagramme d'état est séquentiel ainsi les états sont liés, il est impératif de programmer tous les états et toutes les transactions dans un seul programme de haut niveau.

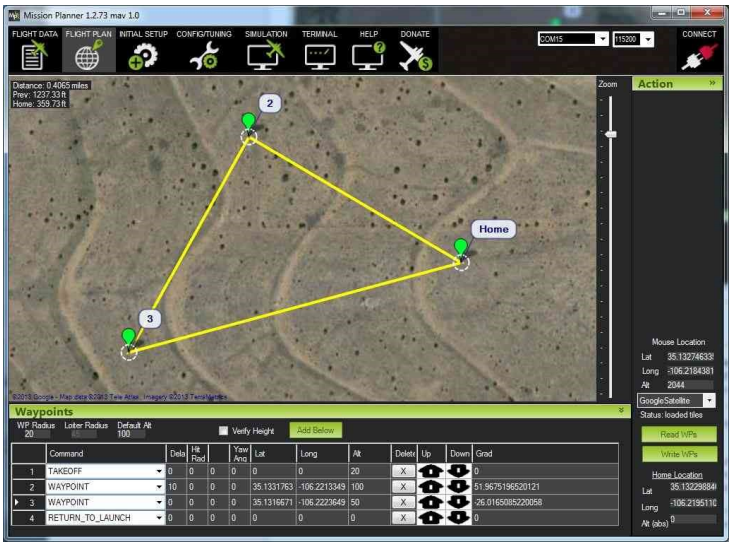


FIGURE 6.8 – Planification avec "Mission Planner".

En utilisant le protocole de communication MAVLink, nous commandons l'autopilote à distance avec un programme en Python (sans utiliser la console de commande interne d'Ardupilot).

Pour l'état déplacement, nous utilisons les lignes de codes donnés par le constructeur de l'autopilote [49].

Le programme Python complet de l'état déplacement est présenté en **Annexe C**.

Chapitre 7

Réalisation du Prototype

Dans cette partie du rapport, nous détaillerons les étapes successives de la construction mécanique de notre robot. Au départ d'une stratégie claire et précise, nous élaborerons les différents systèmes inhérents à celle-ci. Comme dans toute construction mécanique qui se respecte, un cahier des charges a été élaboré.

La réalisation du robot de taille réelle nécessite des ressources en matériel et équipements mais surtout une expertise dans le domaine de la mécanique. Afin de prouver le fonctionnement du robot, nous réalisons un prototype de taille réduite «*proof of concept* ».

7.1 Conception Mécanique

Pour des raisons de faisabilité, le prototype ressemble mais n'est pas identique au robot réel. Des fonctionnalités sont éliminées comme l'angle de tir, le prototype peut effectuer des tirs avec une pente nulle seulement.

De plus, nous avons changé la base du robot d'un rover (quatre roues) à un robot différentiel (deux roues).

7.1.1 Système de Déplacement

La plateforme mobile est placée à 30 mm du sol et elle à les dimensions suivantes :

- Largeur : 205 mm
- Longueur : 225 mm
- Epaisseur : 4mm

La base mobile est motorisée par deux moteurs identiques de référence 12DC Motor 122rpm w Encoder SKU FITO403 . Ces derniers contient des encodeurs 64 CPR intégrés qui sont utilisés pour la partie de localisation (Figure 7.1).

Afin d'assurer un mouvement de rotation précis, nous utilisons quatre roues folles positionnées aux extrémités de la base (Figure 7.2).

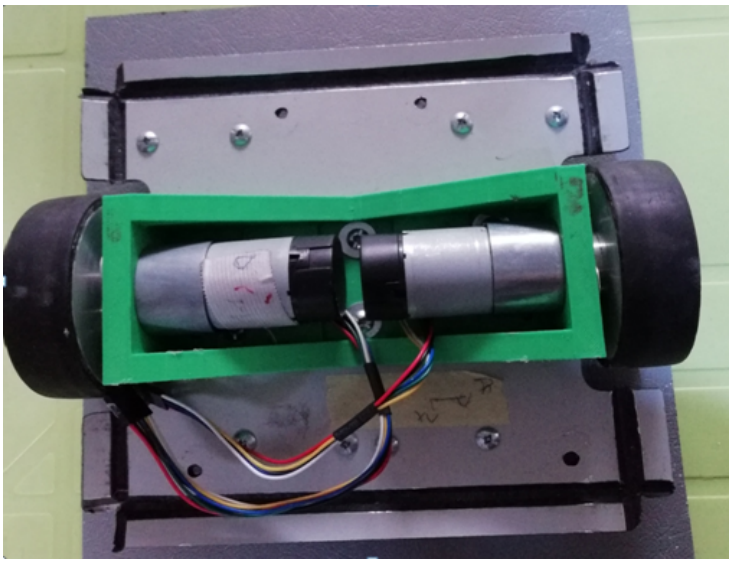


FIGURE 7.1 – Moteurs la base roulante.

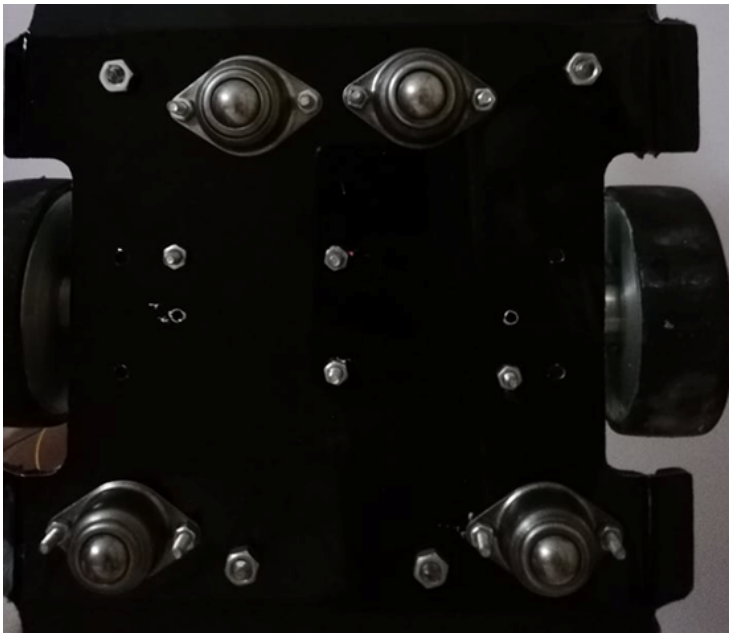


FIGURE 7.2 – Positionnement des roues folles.

Le deuxième étage du robot est réservé à la partie de tir. La plateforme supérieure est de dimension :

- Largeur : 250 mm
- Longueur : 302 mm
- Epaisseur : 4mm

Cette platforme est plus grande que la base pour contenir le système de tir et les cartes de commande ainsi que les batteries. Les étapes de construction de la partie supérieure sont détaillées dans les Figures 7.3, 7.4 et 7.5.

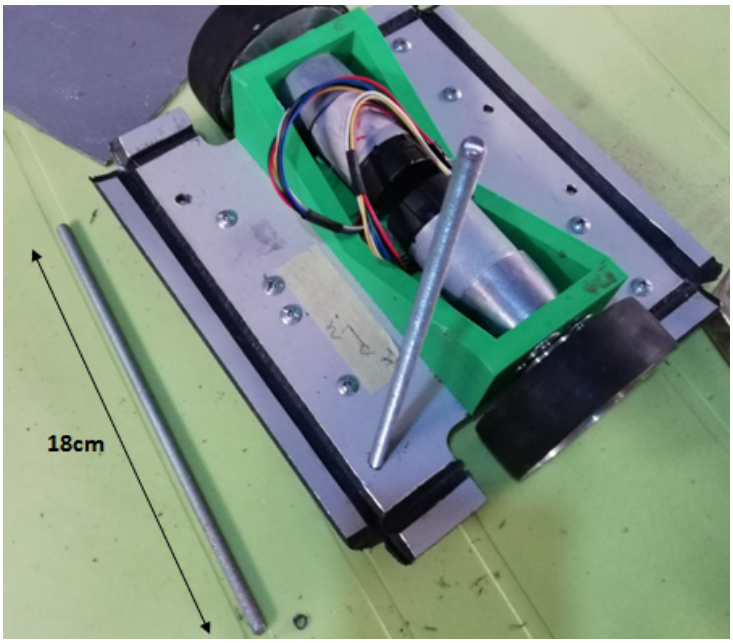


FIGURE 7.3 – Placer les trous des tiges et dimension de la tige.

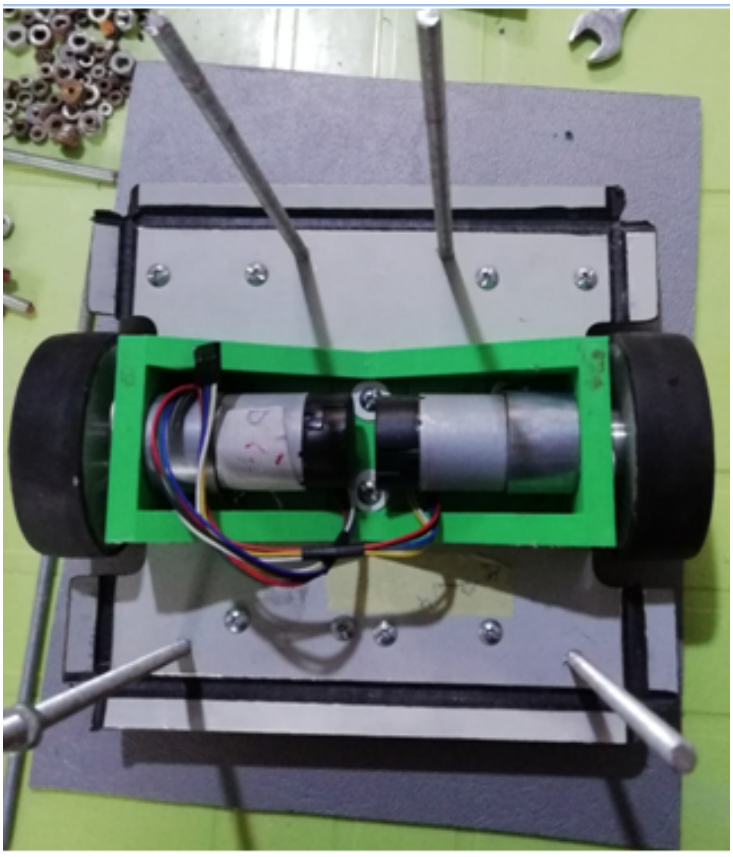


FIGURE 7.4 – Placer les quatre tiges de support.

FIGURE 7.5 – Placer la plateforme supérieure.

7.1.2 Système de Tir

Selon les choix établis au départ du projet, nous devons être capables de tirer les balles du robot vers une cible bien déterminée et à une vitesse précise.

Nous avions choisi de placer ce système au niveau supérieur pour gagner en hauteur de démarrage des ballons car, avec l'effet de la gravité, les projectiles vont forcement perdre en hauteur.

En plus, la partie de stockage des balles, est liée à la partie de tir, donc, un grand espace doit être réservé à cet effet.

Le tir est effectué par deux roues motorisées, la tige du moteur est liée à l'axe de rotation de la roue (Figure 7.6).

FIGURE 7.6 – Roue motorisée.

Ensuite, on place les deux roues motorisées centrées à l'avant du robot (Figure 7.7). Elles sont écartées de 30 mm qui est le diamètre des mini-ballons choisis pour le prototypage.

FIGURE 7.7 – Emplacement des roues tournantes.

7.1.3 Système de Stockage

En s'inspirant des machines similaire à la notre, le système de stockage est positionné à la verticale avec une capacité de stockage de six balles (Figures 7.8 et 7.9). Le diamètre du tube de stockage est de 50 mm (gaine à calibre 50).

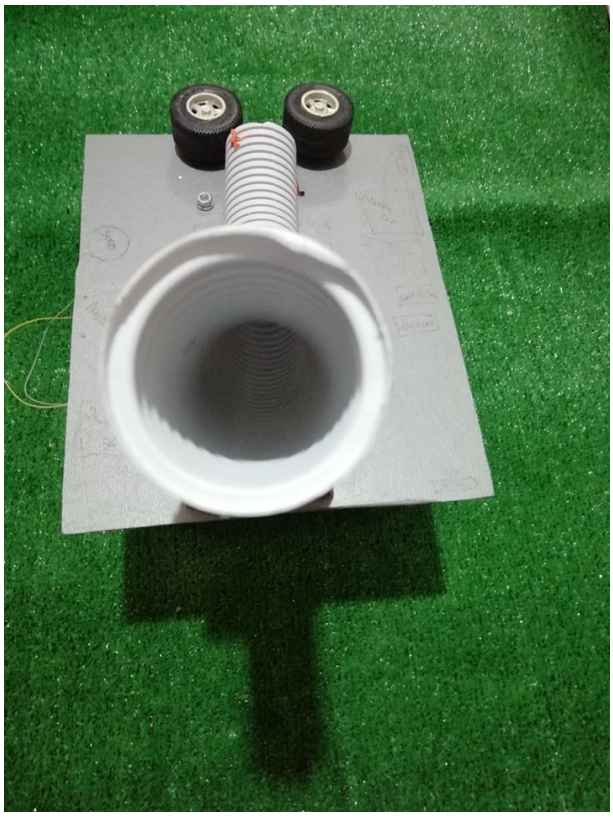


FIGURE 7.8 – Système de stockage (vue de haut).

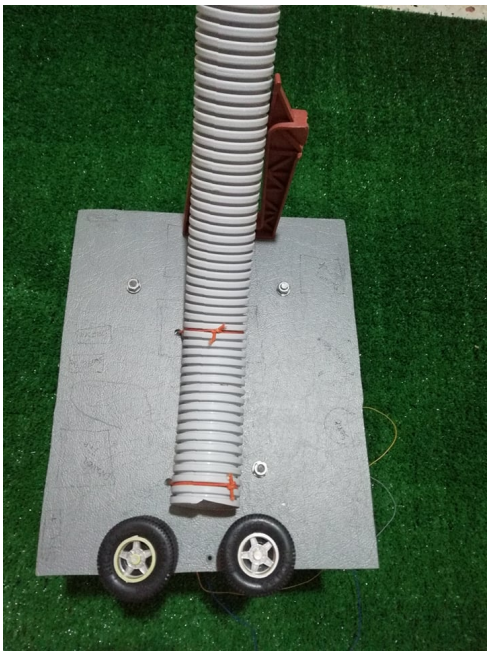


FIGURE 7.9 – Système de stockage (vue de face).

Le système de tir reçoit une balle à la fois et donc un mécanisme de blocage est nécessaire. Pour notre prototype, cette étape ce fait manuellement contrairement au système à dimension réelle.

7.1.4 Partie Électronique

L'électronique constitue un des trois piliers de la science que nous appelons « Robotique ». Elle complète parfaitement la Mécanique et l'Informatique. Elle est utilisée comme outil permettant le traitement d'informations pour nous permettre, à partir de données entrantes (inputs) de fournir des données sortantes (outputs).

En se basant sur le dimensionnement du système réel (Chapitre 4.2), nous avons utilisé les mêmes composants électroniques en réduisant les puissances dans le cas des actionneurs. La carte électronique de pilotage (pixhawk PX4), la carte de commande (Raspberry Pi 3B+) sont gardées afin de tester les programmes du robot à taille réelle.

Comme déjà mentionné, nous avons éliminé quelques fonctionnalités pour le but de simplifier le prototypage du robot, ces dernières sont remplacées avec des LEDs. Le code couleur est illustré dans la Figure 7.10.

Couleur	Signification
Vert	Nouvel exercice disponible
Rouge	Système en arrêt
Bleu	Demande de balle
Blanc	Angle de tir nécessaire.

FIGURE 7.10 – Code couleur LEDs.

Afin de faciliter le câblage du robot et éviter les mauvais branchements, nous avons constitué des schémas électriques pour le système de tir (**Annexe D**) ainsi que le système de déplacement (**Annexe E**).

7.1.5 Partie d'Informatique

Dans le cadre de ce projet, l'informatique permet d'abord et avant toute chose la gestion du déplacement et l'actionnement des différents systèmes mécaniques embarqués, c'est-à-dire la mise en mouvement ou non des moteurs à des vitesses variables ou constantes en fournissant des tensions aux bornes des moteurs.

L'informatique est également utilisée comme outil permettant de traiter les données entrantes (inputs) provenant des diverses cartes et contrôleurs électroniques et de faire réagir le système en conséquence en fournissant des données sortantes " *outputs*".

Afin de donner des bases solides à de tels traitements de données, un outil est primordial - un langage de programmation robuste. De nos jours, de nombreux langages ont fait leur apparition, mais en ce qui nous concerne, celui qui est le plus adapté à l'utilisation que l'on veut en faire et compatible avec les cartes de control que nous utilisons, est le Python.

Notre système est modélisé en plusieurs sous modules qui communiquent entre eux, ce qui rend la tache de programmation plus facile et flexible à des futures modifications. Chaque partie du système (déplacement, tir, analyse, stockage,...) est un programme Python séparé.

- **Système de Tir** En utilisant le principe étudié dans le Chapitre 4.2.1, le prototype conçu peut nous assurer quelques points de cibles sur le cadre du but comme preuve de fonctionnement avec trois niveau de vitesses **(faible, moyen, fort)**.

  Le programme détaillé en python adapter au câblage de la carte de commande Raspberry Pi défini au préalable est attaché en **Annexe F**

- **Système de Déplacement** En se basant sur les commandes de bas niveau du pilote automatique Pixhawk, nous élaborons un programme de haut niveau en Python afin de commander le logiciel du pilote. Le programme détaillé est en **Annexe G**.

- **Système de Stockage** Le remplissage du système de stockage est manuel, mais de déstockage est sur demande. Pour notre prototype, les deux fonctionnalités sont manuelles comme expliqué préalablement.

  Afin de savoir le bon moment de déstocker une balle, un voyant de couleur bleue est utilisé. La programmation de cette partie est en **Annexe H**.

- **Programme Global** Les sous-programmes précédents sont importés au programme principal. Dans ce dernier, la machine d'état prédéfinie en Chapitre 5 est codée. Un exemple de deux différents exercices est démontré pour démonstration du fonctionnement « *Proof of concept* ». L'algorithme de la machine d'état est présenté comme **Annexe I**.

Chapitre 8

Analyse et Interprétation de Données

8.1 Estimation Pose Humaine

L'estimation de la pose fait référence à des techniques de vision par ordinateur qui détectent des personnes en images et en vidéo, de manière à pouvoir déterminer, par exemple, à quel endroit le coude d'une personne apparaît dans une image. Cette technologie ne reconnaît pas qui est dans une image, donc,il n'y a pas d'informations personnelles identifiables associées à la détection de pose. L'algorithme consiste simplement à estimer l'emplacement des principales articulations du corps [50].

L'estimation de la pose a de nombreuses utilisations, allant des installations interactives qui réagissent au corps à la réalité augmentée, en passant par les utilisations d'animation, de fitness, etc.

Notre projet consiste à l'utilisation d'un estimateur de pose afin d'analyser les réactions du gardiende buts aux exercices effectués. Le développement d'un estimateur spécifique à notre application est un projet en lui-même. Pour cela, nous avons préféré utiliser un estimateur open source et ensuite le paramétrer en fonction des besoins de notre travail.

Bien que de nombreux systèmes de détection de pose existent et en plus sont à source ouverte, beaucoup d'entre eux nécessitentdu matériel et / ou des caméras spécialisés, ainsi que de nombreuses configurations système. L'estimateur de pose le plus flexible et accessible que nous avons choisi est « PoseNet » qui fonction sur « TensorFlow.js » avec un ordinateur de bureau ou un téléphone qui posède une webcam ou une caméra spécialisé comme une kinect (les calculs fait précedaments sont pour cas d'une kinect (projet réel)). Pour notre prototype, nous utilisons une caméra RGB.

8.1.1 TensorFlow.js

TensorFlow.js est une bibliothèque permettant de créer et d'exécuter des algorithmes d'apprentissage automatique en « JavaScript » [51]. Les modèles de TensorFlow.js s'exécutent dans un navigateur Web et dans l'environnement « Node.js ». La bibliothèque fait partie de l'écosystème TensorFlow, fournissant un ensemble d'interfaces de programmation d'applications (Application Programming Interface : API) compatibles avec celles de Python, permettant ainsi le portage de modèles entre les systèmes Pythonet les écosystèmes JavaScript.

Grace à la communauté très vaste de JavaScript, TensorFlow.js offre des modèles d'apprentissage automatique et de nouvelles classes de calcul surdifférentstypes d'appareils.

TensorFlow.js est conçu pour fonctionner dans le navigateur et sur le serveur (Figure 8.1). Lorsqu'il est exécuté dans le navigateur, il utilise le GPU « GraphicsProcessing Unit » de la machine via WebGL pour permettre un calcul parallèle. Dans Node.js, TensorFlow.js se lie à la bibliothèque TensorFlow C, permettant ainsi un accès à TensorFlow. TensorFlow.js fournit également une implémentation plus lente de la CPU « Central Processing Unit » en tant que solution de secours peut être exécutée dans n'importe quel environnement d'exécution et elle est automatiquement utilisée lorsque l'environnement n'a pas accès à WebGL ou au binaire TensorFlow [52].

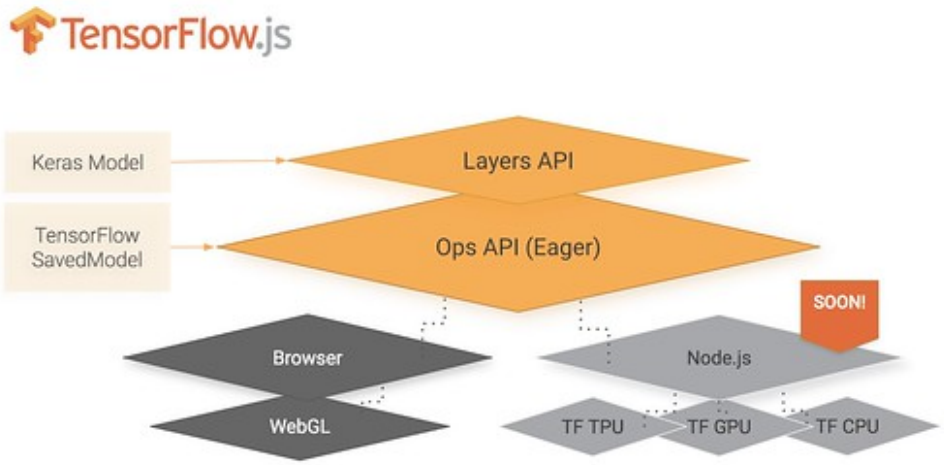


FIGURE 8.1 – Vue d'ensemble de l'architecture TensorFlow.js.

8.1.2 PoseNet

PoseNet peut être utilisé pour estimer une ou plusieurs poses, ce qui signifie qu'il existe une version de l'algorithme capable de détecter une seule personne dans une image / vidéo et une autre version capable de détecter plusieurs personnes dans une image / vidéo.

À un niveau élevé, l'estimation de la pose se déroule en deux phases :

- Une image RVB d'entrée est alimentée par un réseau de neurones à convolution.
- Un algorithme de décodage à pose unique ou à poses multiples est utilisé pour décoder les poses, les scores de confiance des poses, les positions des points clés et les scores de confiance des points clés à partir des sorties du modèle.

Nous définissons quelques termes importants :

- **Pose** : au plus haut niveau, PoseNet renvoie un objet de pose contenant une liste de points clés et un score de confiance au niveau de l'instance pour chaque personne détectée.
- **Score de confiance de la pose** : détermine la confiance globale dans l'estimation d'une pose. Il varie entre 0.0 et 1.0. Il peut être utilisé pour masquer des poses jugées insuffisantes (Figure 8.2).
- **Point-clé** : Partie de la pose d'une personne estimée, telle que le nez, l'oreille droite, le genou gauche, le pied droit, etc. Elle contient à la fois une position et un indice de confiance. PoseNet détecte actuellement 17 points clés illustrés dans la Figure 8.3

- **Score de confiance des points-clés** : détermine la confiance dans l'exactitude d'une position estimée du point-clé. Il varie entre 0.0 et 1.0. Il peut être utilisé pour masquer des points-clés jugés insuffisants (Figure 8.4).
- **Position du point-clé** : coordonnées en 2D (x, y) sur l'image d'entrée originale où un point-clé a été détecté.

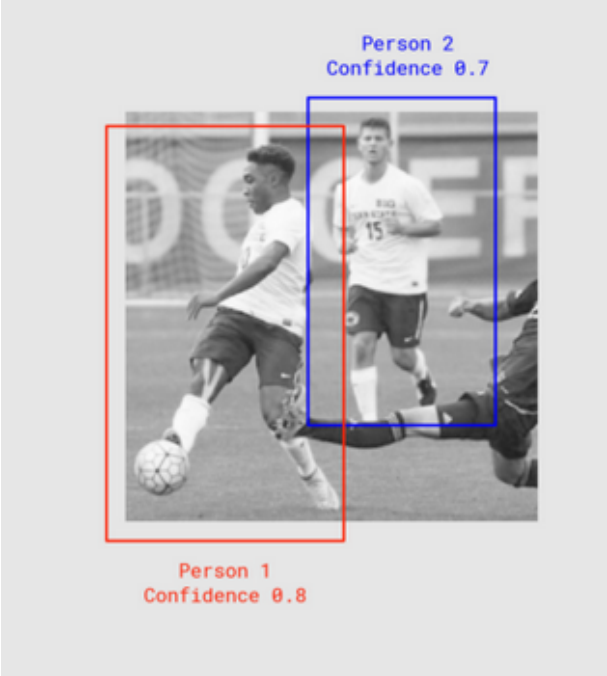


FIGURE 8.2 – Score de confiance de la pose.

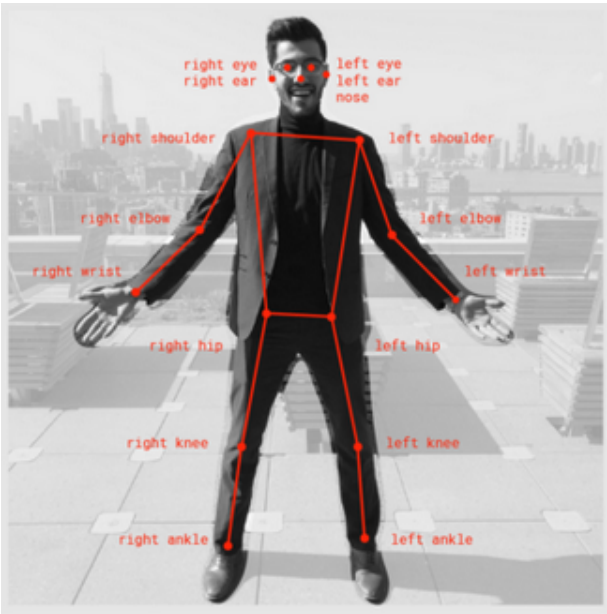


FIGURE 8.3 – Les 17 points-clés de la pose détectés par PoseNet.

FIGURE 8.4 – Score de confiance du point-clé.

8.1.3 Etapes de configuration de l'estimateur avec PoseNet

- **Importation des Bibliothèques TensorFlow.js et PoseNet**

  Beaucoup de travail a été réalisé par l'équipe de développement de PoseNet afin de résumer les complexités du modèle et pour encapsuler la fonctionnalité dans des méthodes à utiliser. Nous devons choisir les bibliothèques qui convient à notre application. La bibliothèque peut être installée avec la commande :

  Npminstall @tensorflow-models/posenet

  Et importer ses modules avec :

  Import * as posenet from '@tensorflow-models/posenet' ;
  Const net = await posenet.load() ;

- **Adaptation de l'Entrée/Sortie**

  Des conditions sur le type et la taille de l'entrée/sortie (de l'image ou la vidéo) sont nécessaires pour assurer une bonne précision de l'estimation de pose [53] :

  — La forme d'entrée, de l'image oula vidéo insérée doit être carrée ;
  — L'échelle de l'image, nombre compris entre 0,2 et 1. La valeur par défaut est 0,50 ;
  — Le sens d'orientation de l'entrée. Si les vidéos ou images sont mal orientées, elles doivent être inversés / reflétés horizontalement ;
  — La fréquence de sortie, doit être définie.Plus la valeur de la fréquence de sortie est faible, plus la précision est grande. De plus, la vitesse de mouvement du corps affect la précision, plus elle est lente, plus la valeur est élevée.

8.1.4 Implementation de l'Estimateur de Pose Humaine sur une Image

Pour commencer, nous allons coder un estimateur de pose qui prend une image comme entrée. Cet estimateur est codé avec le langage HTML (Figure 8.5 et executer sur une pageweb.

L'image choisie pour le test contient une seule personne et elle est enregistrée en interne (sur la machine).

```
<html>
  <head>
    <!-- Load TensorFlow.js -->
    <script src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs"></script>
    <!-- Load Posenet -->
    <script src="https://cdn.jsdelivr.net/npm/@tensorflow-models/posenet"></script>
  </head>

  <body>
    <img id='maha' src='/Pictures/maha.jpg '/>
  </body>
  <!-- Place your code in the script tag below. You can also use an external .js file -->
  <script>
    var flipHorizontal = false;

    var imageElement = document.getElementById('maha');

    posenet.load().then(function(net) {
      const poses = net.estimatePoses(imageElement, {
        flipHorizontal: flipHorizontal,
        decodingMethod: 'single-person'
      });
      const pose = poses[0];
      return pose;
    }).then(function(pose){
      console.log(pose);
    })
  </script>
</html>
```

FIGURE 8.5 – Programme d'estimation de pose (image).

Une pose contenant à la fois un score de confiance et un tableau de 17 points clés.Chaque point clé contient une position et un indice de confiance. De plus, toutes les positions de points-clés ont des coordonnées x et y dans l'espace d'image d'entrée et peuvent être mappées directement sur l'image.

Une partie des résultats (trois points-clés sur 17) de l'estimation de pose de la photo fournie au programme est illustrée dans la Figure 8.6.

```
    "part": "leftShoulder",
    "score": 0.99559044837952
  },
  {
    "position": {
      "y": 95.082359313965,
      "x": 458.21868896484
    },
    "part": "rightShoulder",
    "score": 0.99583911895752
  },
  {
    "position": {
      "y": 94.626205444336,
      "x": 163.94561767578
    },
    "part": "leftElbow",
    "score": 0.9518963098526
  },
  {
    "position": {
      "y": 150.2349395752,
      "x": 245.06030273438
    },
    "part": "rightElbow",
    "score": 0.98052614927292
  },
  {
```


FIGURE 8.6 – Résultats de l'estimation de pose.

8.1.5 Implementation de l'Estimateur de Pose Humaine en Temps Réel

L'estimateur de pose humaine de PoseNet en temps réel est mis en source ouverte (code et démonstration). Il emploie le même principe que l'estimation de pose avec une image en prenant des echantillions de photos à partir de la vidéo en temps réel.

Nous utilisons ces resources fournies ouvertement et nous configurons l'estimateur pour répondre à notre besoin. La sortie du systeme d'estimation de pose doit être numériquement exploitable pour l'utiliser dans la partie analyse de données.

La caméra utilisée est une Webcam (RGB), quelques photos prise de l'estimation de pose en temps réel sont illustrées dans les Figures 8.7 et 8.9.

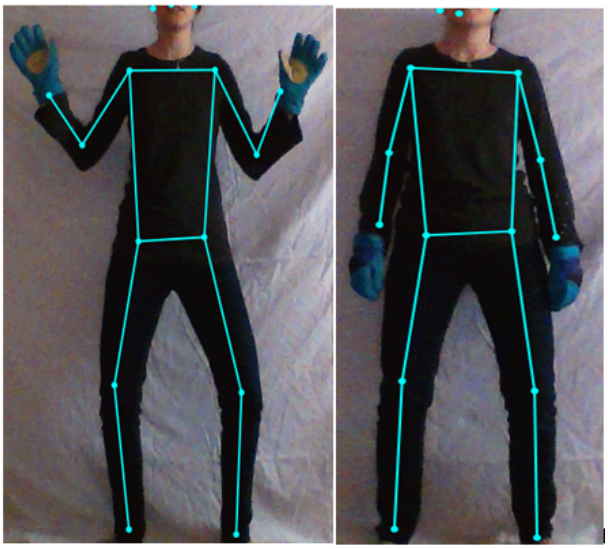


FIGURE 8.7 – Résultats de l'estimation de pose en temps réel.

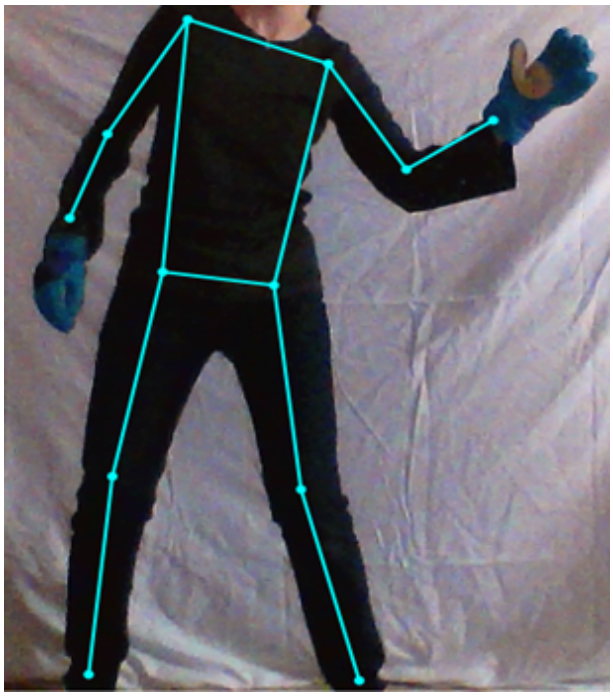


FIGURE 8.8 – Résultat de l'estimation de pose en temps réel avec mouvement.

8.2 Cours de Machine Learning

8.2.1 Définitions

8.2.1.1 Science de Données

La science des données « data science » est l'extraction de connaissance d'ensembles de données. Elle emploie des techniques et des théories tirées de plusieurs domaines des mathématiques, principalement les statistiques, la théorie de l'information et la technologie de l'information, notamment le traitement du signal, des modèles probabilistes, l'apprentissage automatique, l'apprentissage statistique, la programmation informatique,

l'ingénierie de données, la reconnaissance de formes et l'apprentissage, la visualisation, l'analytique prophétique, la modélisation d'incertitude, le stockage de données, la compression de données et le calcul à haute performance.

8.2.1.2 Machine Learning

L'apprentissage automatique "Machine Learning : ML" est un champ d'étude de l'intelligence artificielle qui se fonde sur des approches statistiques. Elles donnent aux ordinateurs la capacité d'apprendre à partir de données. C'est-à-dire d'améliorer leurs performances à résoudre des tâches sans être explicitement programmés pour chacune. Plus largement, cela concerne la conception, l'analyse, le développement et l'implémentation de telles méthodes.

Le ML comporte généralement deux phases. La première consiste à estimer un modèle f à partir de données, appelées observations, qui sont disponibles et en nombre fini, lors de la phase de conception du système. L'estimation du modèle consiste à résoudre une tâche pratique, telle que traduire un discours, reconnaître la présence d'un chat dans une photographie ou participer à la conduite d'un véhicule autonome. Cette phase dite **d'apprentissage** ou **d'entraînement** est généralement réalisée préalablement à l'utilisation pratique du modèle. La seconde phase correspond à la mise en production : le modèle étant déterminé, de nouvelles données peuvent alors être soumises afin d'obtenir le résultat correspondant à la tâche souhaitée.

Vu que l'utilisation du ML est liée à l'utilisation, l'analyse et la transformation des données, ce dernier fait appel à un autre domaine : la science des données "data science". L'utilisation de la science de données est illustrée par la partie verte de la Figure 8.5.

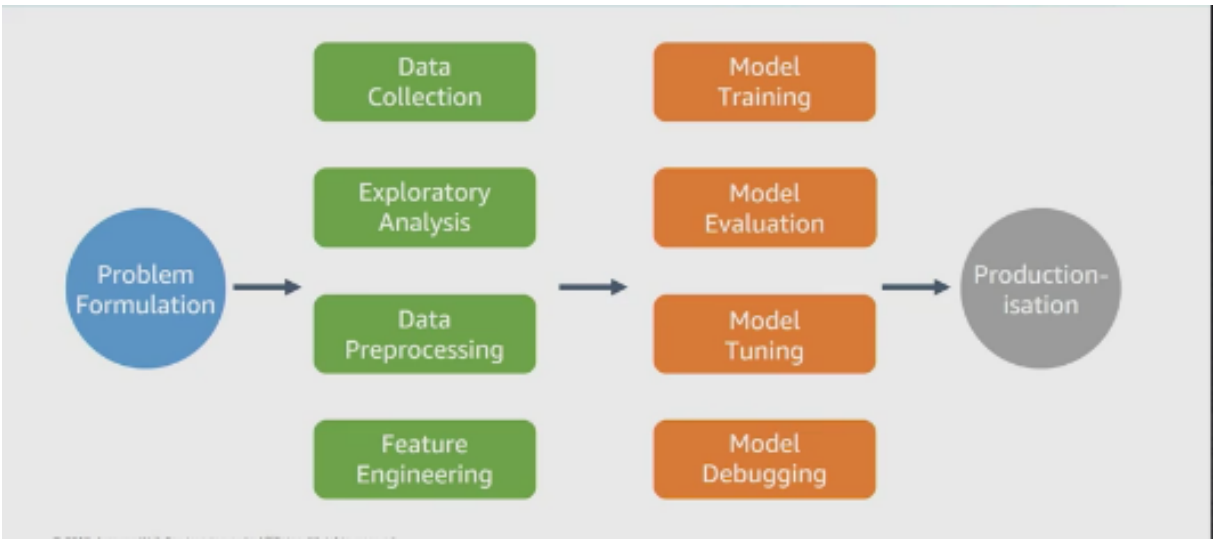


FIGURE 8.9 – L'utilisation de la data science en ML.

Pourquoi le ML ?

- Difficulté d'écrire des programmes informatiques car :
  - — Trop complexes (ex. reconnaissance faciale) ;
  - — Trop de données (ex. prédiction du marché boursier) ;

— Les informations sont dynamiques (ex. systèmes de recommandation) ;

- L'utilisation des données pour améliorer le traitement ;
- Difficulté de traiter un très grand nombre de données.

8.2.2 Les Données

Les problèmes d'apprentissage-machine commencent avec des données, de préférence avec beaucoup de données (exemples ou observations) pour lesquelles on connait déjà la réponse cible. Les données pour lesquelles la réponse cible est connue sont appelées données étiquetées.

Afin d'entrainer et tester un algorithme d'apprentissage avec un grand taux de certitude, il faut que les données utilisées soient :

- En nombre suffisant et de bonne qualité (petite variance, sans bruit, sans biais) ;
- Correctement représentatives du système à prévoir, c.-à-d. que l'échantillonnage des données doit être équitable entre tous les sous ensemble de l'environnement échantillonné.

8.2.2.1 Collecte des Données

Chaque exemple/observation figurant dans les données doit contenir deux éléments :

- La cible, c.-à-d., la réponse à prédire. Les données sont fournies étiquetées avec la cible (réponse correcte) à l'algorithme d'apprentissage-machine pour qu'il apprenne à partir d'elles. Ensuite, le modèle de ML formé pour prédire cette réponse est utilisé sur des données pour lesquelles la réponse cible n'est pas connue.
- Variables/entités : ce sont des attributs de l'exemple qui peuvent être utilisés pour identifier des tendances afin de prédire la réponse cible.

Par exemple, pour le problème de classification des e-mails, la cible est une étiquette qui indique si un e-mail correspond à du courrier indésirable ou non. Comme exemples de variables, on peut citer l'expéditeur de l'e-mail, le texte dans le corps de l'e-mail, le texte dans la ligne d'objet, l'heure à laquelle l'e-mail a été envoyé et l'existence d'une correspondance antérieure entre l'expéditeur et le destinataire.

Afin de s'assurer que les échantillons utilisés représentent équitablement le problème et ne favorisent pas une partie de la population sur une autre (ce qui peut engendrer de fausses prédictions), on utilise l'échantillonnage aléatoire. De plus, on peut utiliser cette méthode sur chaque partie de la population pour être sûr que le partage sera adéquat. Cette méthode est appelée : échantillonnage stratifié "stratified sampling".

8.2.2.1.1 Catégorisation des Données

1. **Données Quantitatives :** Les données quantitatives représentent une mesure de quelque chose. Elles peuvent être de deux types :
   - `Continue` : Les valeurs que peut prendre une donnée numérique continue sont infinies. Elles représentent une mesure d'une quantité. Par exemple le poids en Kg d'une personne, la distance entre Paris et Madrid en Km...

- `Discrète` : Il s'agit généralement d'un comptage d'un événement. Par exemple : le nombre d'enfants dans un foyer. Il ne peut pas y avoir 2.5 enfants dans un foyer.

2. **Données Qualitatives :** Les données qualitatives sont aussi nommées des modalités. Une donnée qualitative n'a pas une sémantique mathématique. Elles apportent une information sur une caractéristique d'un individu observé. Exemples de modalité : sexe d'une personne (Homme / Femme), pays de naissance d'une personne, race d'un chien, affiliation politique, etc. Elles peuvent être de deux types :
   - `Données Ordinales` : Elles peuvent être ordonnées, par exemple les tailles : $\{S, M, L\}$ on peut les ordonner comme suit : $S < M < L$. Ce type peut facilement être remplacé par des chiffres vu que les algorithmes d'apprentissage n'acceptent que les données numériques.
   - `Données Nominales` : Elles ne peuvent pas être ordonnées, par exemple les couleurs : Rouge, Vert, Noir, Rose Généralement, ce type de variables est représenté par des caractères donc, on ne peut pas les utiliser directement dans les algorithmes d'apprentissage. Par conséquent, il faut coder les variables numériquement, la Figure 8.10 illustre un exemple de codage.

On a les données de type nominal suivantes :

	Sex	Pclass
0	Male	3
1	Female	1
2	Female	3
3	Female	1
4	Male	3

Le codage en variables quantitatives est comme suit :

	Pclass	Sex_Female	Sex_Male
0	3	0	1
1	1	1	0
2	3	1	0
3	1	1	0
4	3	0	1

FIGURE 8.10 – Codage de variable nominale vers une variable quantitative.

8.2.2.2 Étiquetage des Données

Souvent, les données ne sont pas disponibles sous une forme étiquetée. La collecte et la préparation des variables et de la cible sont souvent les étapes les plus importantes dans la résolution d'un problème d'apprentissage-machine. Les exemples de données doivent être représentatifs des données pour lesquelles le modèle sera utilisé pour établir une prédiction.

Par exemple, si l'on souhaite prédire si un e-mail correspond à du courrier indésirable ou non, il faudra collecter des exemples positifs (courriers indésirables) et des négatifs (courriers non indésirables) pour que l'algorithme d'apprentissage-machine soit en mesure d'identifier des tendances qui permettront de distinguer les deux types d'e-mails.

Généralement cette étape d'étiquetage se fait manuellement par des humains. Pour permettre à la personne qui effectue ce travail de fournir de bons résultats, il faut suivre les consignes suivantes :

- Mettre des instructions détaillées et minimiser les ambiguïtés ;
- Utiliser les bons outils : Excel, Amazon Mechanical Turk...etc. ;
- Poser de bonnes et simples questions. Ex. "Est qu'il y a un chat dans l'image ?" ;
- Donner les mêmes données pour plusieurs personnes et comparer les résultats ;
- Tester la difficulté et les ambiguïtés des questions en les posant à plusieurs personnes et voir si elles sont bien comprises ou si elles nécessitent des modifications.

Pour certain cas, on peut utiliser des méthodes d'étiquetage spéciales, par exemple :

- Donner des récompenses aux personnes qui vont faire le travail.
- Demander à des volontaires si le projet a un aspect humanitaire.
- Présenter le travail en tant qu'un jeu avec comme récompenses des points/niveaux.

8.2.2.3 Entité des Données

Nous utilisons l'intuition pour rechercher des entités en répondant à la question **comment un humain pourrait prédire ça ?**

Les entités changent d'un problème à un autre, et pour les extraire on procède comme suit :

- Générer de nombreuses entités ;
- Appliquer ces entités sur de nombreux modèles ;
- Appliquer la réduction de dimensionnalité [1] ou des transformations pour augmenter le nombre d'entités si nécessaire.

Une fois les données étiquetées disponibles, il convient de les convertir dans un format acceptable par l'algorithme ou le logiciel utilisé. Par exemple, Amazon ML nécessite de convertir les données au format CSV (valeurs séparées par des virgules), chaque exemple constitue une ligne du fichier CSV et chaque colonne (hormis la dernière) contient une seule variable d'entrée. La dernière colonne contient la réponse cible.

Les données peuvent aussi être filtrées ou mises à l'échelle en cas de besoin, spécialement pour le cas des images ou son. La mise à l'échelle peut être nécessaire étant donné que certains algorithmes tels que Gradient et KNN sont sensibles aux grandes différences d'échelle.

8.2.2.4 Fractionnement des Données en Données d'Apprentissage et d'Evaluation

L'objectif fondamental de l'apprentissage-machine est de généraliser au-delà des instances de données utilisées pour former les modèles. Nous voulons évaluer le modèle pour estimer la qualité de la généralisation des tendances pour des données avec lesquelles le

1. Les techniques de réduction de dimensionnalité permettent de représenter les données dans un espace de plus petite dimension. On peut citer comme exemple, l'analyse en composantes principales (ACP).

modèle n'a pas été formé. Toutefois, comme les instances futures ont des valeurs cibles inconnues et que nous ne pouvons pas vérifier la précision de nos prédictions pour les instances futures, nous devons utiliser une partie des données étiquetées comme indicateur pour les données futures. L'évaluation du modèle avec les mêmes données qui ont été utilisées pour l'apprentissage n'est pas utile. En effet, elle récompense les modèles qui peuvent « mémoriser » les données d'apprentissage, par opposition à une généralisation à partir de celles-ci.

Une stratégie courante consiste à prendre toutes les données étiquetées disponibles, et à les fractionner en sous-ensembles d'apprentissage et d'évaluation, (généralement avec une proportion de $70-80\%$ pour l'apprentissage et de $20-30\%$ pour l'évaluation). Le système d'apprentissage-machine utilise les données d'apprentissage pour former les modèles à identifier des tendances, et utilise les données d'évaluation pour évaluer la qualité prédictive du modèle formé. Le système d'apprentissage-machine évalue les performances prédictives en comparant les prédictions sur le jeu de données d'évaluation à leurs valeurs réelles (vérité de terrain) à l'aide de diverses métriques. En règle générale, le "meilleur" modèle est utilisé sur le sous-ensemble d'évaluation pour établir des prédictions sur les instances futures pour lesquelles la réponse cible n'est pas connue.

8.2.3 Entrainement du Modèle

L'algorithme apprendra des données d'apprentissage les tendances mettant en correspondance les variables et la cible, et il fournira en sortie un modèle capturant ces relations. Le modèle d'apprentissage-machine peut alors être utilisé pour obtenir des prédictions sur de nouvelles données pour lesquelles la réponse cible n'est pas connue.

8.2.3.1 Catégorisation des Modèles

Les algorithmes d'apprentissage peuvent se catégoriser selon le mode d'apprentissage qu'ils emploient :

1. **Apprentissage Supervisé :** Le système apprend à classer selon un modèle de classification ou de classement ; on parle alors d'apprentissage supervisé (ou d'analyse discriminante). Un expert doit préalablement étiqueter des exemples. Le processus se passe en deux phases. Lors de la première phase (hors ligne, dite d'apprentissage), il s'agit de déterminer un modèle à partir des données étiquetées. La seconde phase (en ligne, dite de test) consiste à prédire l'étiquette d'une nouvelle donnée, connaissant le modèle préalablement appris.
2. **Apprentissage Non-Supervisé :** Quand le système ou l'opérateur ne dispose que d'exemples, mais non d'étiquette, et que le nombre de classes et leur nature n'ont pas été prédéterminés, on parle d'apprentissage non supervisé. L'algorithme doit découvrir par lui-même la structure plus ou moins cachée des données.
3. **Apprentissage Semi-Supervisé :** Il est mis en œuvre quand des données (ou étiquettes) manquent. Le modèle doit utiliser des exemples non étiquetés pouvant néanmoins donner des informations utiles. Exemple : En médecine, il peut constituer une aide au diagnostic ou au choix des moyens les moins onéreux de tests de diagnostic.
4. **Apprentissage par Renforcement :** L'algorithme apprend un comportement étant donné une observation. L'action de l'algorithme sur l'environnement produit

une valeur de retour qui guide l'algorithme d'apprentissage. L'algorithme de "Q-learning" est un exemple classique.

8.2.3.1.1 Méthodes Supervisées

Elles peuvent être regroupées en problèmes de régression et de classification. Les deux problèmes ont pour but la construction d'un modèle capable de prédire la valeur de l'attribut dépendant à partir des variables d'attribut. La différence entre les deux réside dans le fait que l'attribut dépendant est numérique pour la régression et catégorique pour la classification.

On va citer quelques méthodes de chaque catégorie.

Régression Logistique La régression logistique ou modèle logit est un modèle de régression binomiale. Comme pour tous les modèles de régression binomiale, il s'agit de modéliser au mieux un modèle mathématique simple à des observations réelles nombreuses. En d'autres termes d'associer à un vecteur de variables aléatoires $(x_1, \ldots, x_K)$ une variable aléatoire binomiale génériquement notée y. La régression logistique constitue un cas particulier de modèle linéaire généralisé. Elle est largement utilisée en apprentissage automatique.

La régression logistique repose sur l'hypothèse fondamentale suivante, où l'on reconnaît la mesure nommée évidence :

$$Ev(p) = \ln \frac{p}{1-p}$$

Popularisée par I.J. Good, E.T Jaynes et Myron Tribus pour les besoins de l'inférence bayésienne en évitant des renormalisations continuelles sur [0,1] :

$$\ln \frac{p(X|1)}{p(X|0} = a_0 + a_1 x_1 + ... + a_j x_j$$

Une vaste classe de distributions répond à cette spécification, la distribution multi normale décrite en analyse discriminante linéaire par exemple, mais également d'autres distributions, notamment celles où les variables explicatives sont booléennes (0/1).

Par rapport à l'analyse discriminante, ce ne sont plus les densités conditionnelles p (X | 1) et p (X | 0) qui sont modélisées mais le rapport de ces densités. La restriction introduite par l'hypothèse est moins forte.

Réseau de Neurones Un réseau neuronal est l'association, en un graphe plus ou moins complexe, d'objets élémentaires, les neurones formels. Les principaux réseaux se distinguent par l'organisation du graphe, c'est-à-dire leur architecture, son niveau de complexité (le nombre de neurones, présence ou non de boucles de rétroaction dans le réseau), par le type des neurones (leurs fonctions de transition ou d'activation) et enfin par l'objectif visé : apprentissage supervisé ou non, optimisation, systèmes dynamiques...

Neurone Formel Le neurone reçoit les entrées $x_1, ..., x_i, ..., x_n$. Le potentiel d'activation du neurone p est défini comme la somme pondérée (les poids sont les coefficients synaptiques wi) des entrées. La sortie o est alors calculée en fonction du seuil θ.

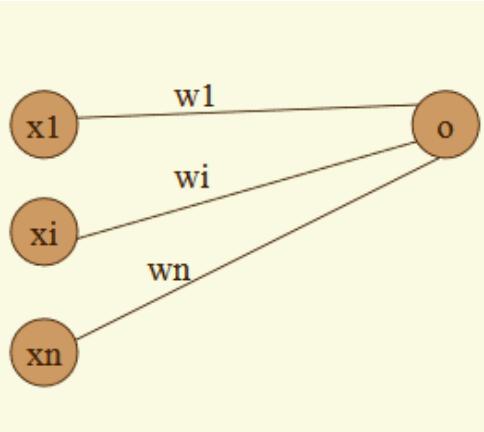


FIGURE 8.11 – Représentation d'un neurone formel.

Réseau Connexionniste / Réseau de Neurones Un réseau de neurones est un graphe valué orienté, constitué d'un ensemble d'unités (ou automates), réalisant des calculs élémentaires, structurées en couches successives capables d'échanger des informations au moyen de connexions qui les relient.

On cherche, au moyen de l'algorithme d'apprentissage à trouver des poids tels que :

- Les exemples sont reconnus : $d^k = \Psi(x^k, w)$
- On obtient une bonne généralisation : $d = \Psi(x, w)$
- Une réponse raisonnable pour l'entrée x

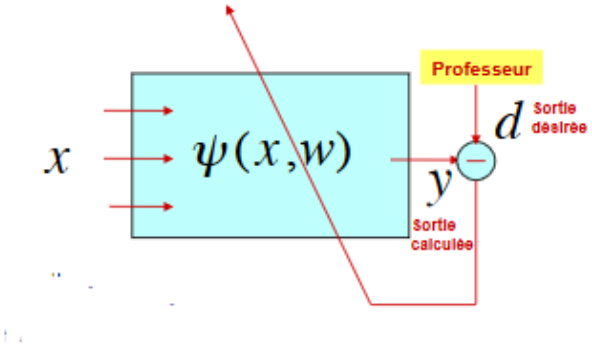


FIGURE 8.12 – Apprentissage supervisé.

Arbre de Décision L'apprentissage par arbre de décision désigne une méthode basée sur l'utilisation d'un arbre de décision comme modèle prédictif. On l'utilise notamment en fouille de données et en apprentissage automatique.

Dans ces structures d'arbre, les feuilles représentent les valeurs de la variable-cible et les embranchements correspondent à des combinaisons de variables d'entrée qui mènent à ces valeurs. En analyse de décision, un arbre de décision peut être utilisé pour représenter de manière explicite les décisions réalisées et les processus qui les amènent. En apprentissage et en fouille de données, un arbre de décision décrit les données mais pas les décisions elles-mêmes, l'arbre serait utilisé comme point de départ au processus de décision.

C'est une technique d'apprentissage supervisé : on utilise un ensemble de données pour lesquelles on connaît la valeur de la variable-cible afin de construire l'arbre (données dites étiquetées), puis on extrapole les résultats à l'ensemble des données de test. Il existe deux principaux types d'arbre de décision en fouille de données :

- Les arbres de classification « Classification Tree » permettent de prédire à quelle classe la variable-cible appartient, dans ce cas la prédiction est une étiquette de classe,
- Les arbres de régression « Regression Tree » permettent de prédire une quantité réelle par exemple, le prix d'une maison ou la durée de séjour d'un patient dans un hôpital, dans ce cas la prédiction est une valeur numérique.

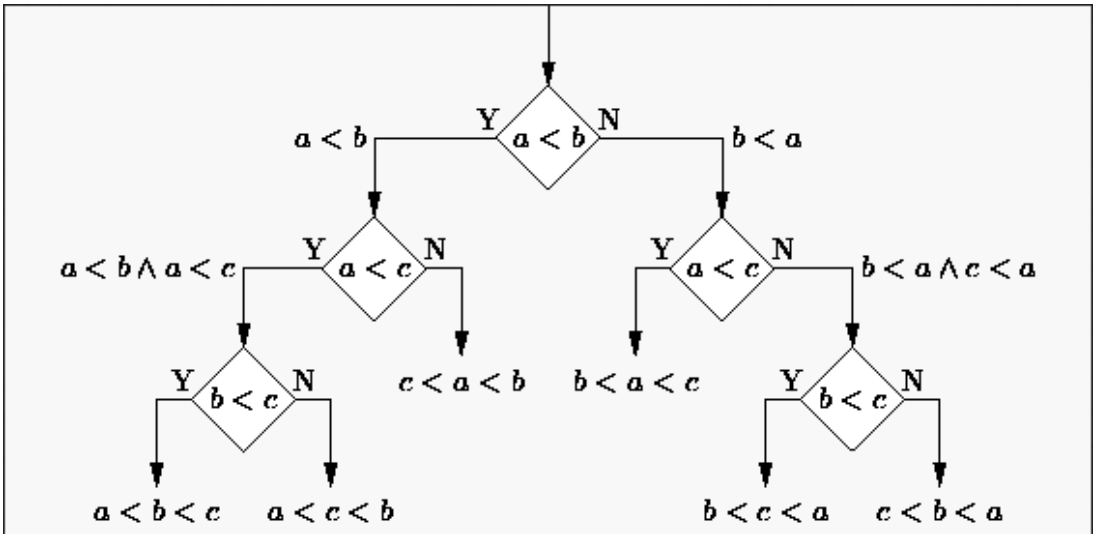


FIGURE 8.13 – Exemple d'arbre de décision.

Les Machines à Vecteurs de Support Les machines à vecteurs de support ou séparateurs à vaste marge (en anglais support vector machine, SVM) sont un ensemble de techniques d'apprentissage supervisé destinées à résoudre des problèmes de discrimination et de régression. Les SVM sont une généralisation des classifieurs linéaires.

Les séparateurs à vaste marge ont été développés dans les années 1990 à partir des considérations théoriques de Vladimir Vapnik sur le développement d'une théorie statistique de l'apprentissage : la théorie de Vapnik-Chervonenkis. Ils ont rapidement été adoptés pour leur capacité à travailler avec des données de grandes dimensions, le faible nombre d'hyperparamètres, leurs garanties théoriques, et leurs bons résultats en pratique.

Les SVM ont été appliqués à de très nombreux domaines (bio-informatique, recherche d'information, vision par ordinateur, finance...). Selon les données, la performance des machines à vecteurs de support est de même ordre, ou même supérieure, à celle d'un réseau de neurones ou d'un modèle de mélanges gaussiens.

Les SVM peuvent être utilisés pour résoudre des problèmes de discrimination (décider à quelle classe appartient un échantillon), ou de régression(prédire la valeur numérique d'une variable). La résolution de ces deux problèmes passe par la construction d'une fonction h qui à un vecteur d'entrée x fait correspondre une sortie y :

$$y = h(x)$$

8.2.3.1.2 Méthodes Non-Supervisées

K-Moyennes Le partitionnement en k-moyennes « k-means » est une méthode de partitionnement de données et un problème d'optimisation combinatoire. Étant donnés des points et un entier k, le problème est de diviser les points en k groupes, souvent appelés clusters, de façon à minimiser une certaine fonction. On considère la distance d'un point à la moyenne des points de son cluster ; la fonction à minimiser est la somme des carrés de ces distances.

Les k-moyennes sont utilisées en apprentissage non supervisé où l'on divise des observations en k partitions. Les nuées dynamiques sont une généralisation de ce principe, pour laquelle chaque partition est représentée par un noyau pouvant être plus complexe qu'une moyenne.

Pour la détermination de l'entier k, il y a plusieurs méthodes par exemple : la méthode de l'erreur et la méthode de groupage hiérarchique.

Étant donné un ensemble de points $(x_1, x_2, \ldots, x_n)$, on cherche à partitionner les n points en k ensembles S = $\{S_1, S_2, \ldots, S_k\}$ (k $\leq$ n) en minimisant la distance entre les points à l'intérieur de chaque partition :

$$\underset{s}{\operatorname{argmin}} \sum_{i=1}^{k} \sum_{x_j \in S_i} ||x_j - \mu_i||^2$$

8.2.3.2 Réglage du Modèle

Cette partie consiste à analyser le modèle générer en termes de qualité et le taux d'erreurs.

Réglage des Hyperparamètres Les hyperparamètres sont les paramètres qui ne sont pas estimés dans la partie d'entrainement du système. Ils doivent être optimisés séparément. Plusieurs techniques sont utilisées : Recherche par grille, recherche aléatoire,...

Réglage des Données d'Apprentissage Si le nombre d'observations est petit, l'échantillonnage favorise une sous-partie sur une autre. Ou, en cas de difficulté d'étiquetage des données, on collecte plus de données (si possible dans une certaine partie ou dans toutes les parties), ou bien on duplique ou on utilise une technique comme « Synthetic Minority Oversampling Technique (SMOTE) » pour avoir plus d'échantillons.

En cas d'abondance de données, on essaye de diminuer le nombre tout en s'assurant d'éviter une perte d'information.

Réglage des Entités des Données Si on dispose de données biaisées, peu de données où le modèle n'est pas suffisamment flexible, on doit rajouter des entités ou appliquer des transformations sur les entités que l'on a. Si on a beaucoup d'entités, on veut réduire la dimension des entités tout en s'assurant d'éviter une perte d'information [54].

8.3 Choix de la Méthode d'Analyse

Après avoir étudié la logique floue et le Machine Learning, nous pouvons estimer la méthode, le temps et les données nécessaires pour chaque technique d'analyse.

8.3.1 Machine Learning

Nous cherchons à créer un comportement de gardien parfait, pour ensuite le comparer au comportement de notre gardien sous test et identifier les différences entre les deux.

L'entrainement du gardien parfait utilise des vidéos extraites de jeux vidéo en mode **"entrainement"** (coup franc du joueur contre le gardien seul). Nous gardons seulement les tirs où le gardien a réussi à attraper le ballon.

Sachant que le gardien de buts dans les jeux vidéo est autonome (joue sans l'intervention du gamer), se qui signifie qu'une analyse de *"comment un gardien doit réagir"* est déjà faite et implémentée par les développeurs de ces jeux.

Les données doivent être normalisées (angle et distance de vue) pour les utiliser à l'entrainement. Nous appliquons l'estimation de la pose humaine sur toutes les séquences prises de la vidéo d'entrainement enregistrée pour déterminer quelles sont les séquences les plus déterminantes afin d'éviter le but (les séquences en commun avec les autres vidéos qui possèdent les mêmes entrées). Donc, nous aurons pour chaque combinaison d'entrées, un nombre réduit de séquences déterminantes afin de les utiliser pour la comparaison avec la vidéo du test réel par la suite.

- **Technique choisie** : apprentissage supervisé "***supervised learning***".
- **Temps estimé de collecte de données** : 40h
- **Temps estimé d'entrainement** : 10h
- **Temps estimé des testss et implementation** : 20h
- **Méthode de collecte de données** : manuelle (jeux vidéo).

8.3.2 Logique Floue

L'implémentation d'un système d'évaluation automatisé a pour but principal d'imiter l'entraineur humain afin de l'aider à prendre des décisions plus objectives et précises. Nous pouvons reproduire le même résonnement de l'être humain, en travaillant avec un expert du domaine qui identifié les lois de prise de décision en fonction de plusieurs critères en entrées (la taille du joueur, l'expérience, les caractéristiques des tirs,..).

La logique floue est la plus adaptée parmi les méthodes d'inteligence artificielle symboliques, parce que les variables (données d'entée) ne sont pas exactes et elles peuvent varier dans une plage de valeurs numériques déterminables.

- **Technique choisie** : logique floue type 1.
- **Temps estimé de collecte de données** : 8h
- **Temps estimé de l'étude du système** : 12h
- **Temps estimé des tests et implementation** : 10h
- **Méthode de collecte de données** : manuelle (avec un expert).

8.3.3 Conclusion

Le choix de la méthode d'analyse est basé sur les critaires cités au paravant (temps, resource et expertise).

La disponibilité des données en quantité importante est primordiale à l'utilisation du ML. D'autre part, le bon choix de l'expert pour la méthode de logique floue influence énormément la qualité de notre système d'analyse.

Conclusion Générale

Cette étude nous a permis de mettre en pratique toutes les connaissances acquises durant notre cursus de formation d'ingénieur, d'approfondir nos connaissances en se basant sur des documents techniques, des cours et des travaux scientifiques similaires.

Il est à souligner que ce projet est constitué de plusieurs parties de différents domaines : Mécanique, Automatique, Electronique et Informatique. Ce qui nous a obligé de toucher à des domaines complémentaires au domaine d'étude principal qui est l'Automatique et ainsi devenir plus polyvalents. Avec ce projet lourd et varié, nous avons appris à bien planifier et gérer le temps, qui sont deux qualités importantes pour chaque ingénieur.

Nous avons réussi à modéliser le système pour les parties : Mécanique, Automatique et Informatique ainsi que la construction d'un prototype de taille réduite du robot tireur de ballons.

Ce qui concerne la partie analyse, nous avons travaillé sur un estimateur de pose humaine qui fournit les données nécessaires à l'interprétation des réactions du gardien.

Notons qu'avec le peu de temps disponible et en tenant compte de la diversité et la complexité de chaque partie du projet, nous avons accompli dans ce travail une preuve de conception "Proof of concept" et non pas un produit fini. Afin de présenter un produit précis, fini et robuste nous proposons que chaque partie du système soit traitée séparément ainsi, les performances des différentes constituantes du système d'entrainement soient renforcées.

Le robot tireur de ballons conçu, effectue des tirs sur des points prédéterminés et limités des buts. Afin d'élargir le champ du tir, il est recommandé de modéliser le mouvement de tir par une équation mathématique globale.

L'amélioration de la précision de la pose humaine nécessite l'utilisation d'une caméra plus performante, par exemple la kinect. L'algorithme utilisé pour l'estimation peut être amélioré en utilisant l'une des méthodes de l'état de l'art cité au paravant.

La mise en place de l'intelligence artificielle qui a pour rôle d'interpréter les réactions du gardien et proposer des nouveaux exercices, peut se faire de deux manières : Machine Learning ou logique floue. Il est préférable de consacrer un projet dédié à cette partie d'analyse de données afin d'avoir des résultats exploitables.

Pour la continuité du projet, nous avons documenté, bien expliqué chaque détail du projet, sauvegardé les programmes et les logiciels de travail sur la plateforme de développement et coopération des projets "GuitHub", afin de faciliter la reprise du sujet ou d'une partie du sujet.

Bibliographie

[1] LAWGEEX . About us , [En ligne]. https://www.lawgeex.com/aboutus. [Page consultée le : 20/01/2019].

[2] THEATLANTIC. Technology, the invisible opponent, [En ligne]. https://www.theatlantic.com/technology/archive/2016/03/the invisibleopponent/475611/ . [Page consultée le : 20/01/2019].

[3] DOCPLAYER. Lancer Balles de Tennis Tutor 4 Plus Player [En ligne]. https://docplayer.fr/29115688-Lance-balles-tennis-tutor-4-plus-player.html . [Page consultée le : 22/01/2019].

[4] NUMERAMA. L'équipe Feminine Japonaise de Volley-ball s'Entraine avec un Robot [En ligne]. https://www.numerama.com/tech/249453-lequipe-feminine-japonaise devolley-ball-sentraine-avec-un-robot.html [Page consultée le : 22/01/2019].

[5] LE TEMPS. Un Robot Sécurise le Placage d'un Terrains de Football American, [En ligne]. https://www.letemps.ch/sciences/terrains-football-americain-un-robotsecurise-placages. [Page consultée le : 22/01/2019].

[6] BASEBALL JUGS SPORTS. Oftball Passing Machine fabriqué. [En ligne]. https://jugssports.com/products/changeup-super-softball-pitching-machine.html. .[Page consultée le : 28/01/2019].

[7] YUE, GEORGE, CARMINE MILONE, JOSEPH MILONE, ALEX HEYDARI, AND JOE FYNEFACE. *Automated Football Launcher - Design***.** *Automated Football Launcher***.** Georgia Tech University, 2010. Web. 22 Sept. 2013.

[8] GLOBUS. Eurogoal. [En ligne]. https://www.clickforfoot.com/materiel-d-entrainement/415-lanceur-de-ballon-eurogoal-1500.html.[Page consultée le : 28/01/2019].

[9] MUSGRAVE, MYERS, JANG, JIA. Mechatronic Design, Final Report- Task, 2012, 18. 18-578.

[10] CHENAVIER, LECOEUR TALBI, CROWLEY. Estimation de la Position d'un Robot par Odométrie et Vision Monoculaire. http://documents.irevues.inist.fr/bitstream/handle/2042/1909/nondispo.pdf?sequence=1.

[11] MANUBATBAT. Positionnement du robot. [Document électronique].Coco, 2002, http://manubatbat.free.fr/doc/positionning/node5.html

[12] MAIMONE, CHENG, MATTHIES . Two years of Visual Odometry on the Mars Exploration Rovers (PDF). Journal of Field Robotics. 24 (3): 169–186. doi:10.1002/rob.20184. Retrieved 2008-07-10.

[13] STACHNISS, FRESE, GRISETTI. What is OpenSLAM.org? [En ligne]. https://openslam-org.github.io/. [Page consultée le : 06/02/2018].

[14] ENGEL, SCHÖPS, CREMERS. LSD-SLAM: Large-scale direct Monocular SLAM. [En ligne]. https://vision.in.tum.de/_media/spezial/bib/engel14eccv.pdf [Page consultée le: 06/02/2018].

[15] ENGEL , STURM, CREMERS. Semi-Dense visual Odometry for a Monocular camera. [En ligne]. http://vision.in.tum.de/_media/spezial/bib/engel2013iccv.pdf [Page consultée le: 06/02/2018].

[16] KLEIN, MURRAY. Parallel tracking and mapping for small AR workspaces, in IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR), Nara, Japan, November 2007.

[17] RUBLEE, RABAUD, KONOLIGE, BRADSKI. “ORB: an efficient alternative to SIFT or SURF”. Willow Garage, Menlo Park, California.

[18] DINE. Localisation et cartographie simultanées par optimisation de graphe sur architectures hétérogènes pour l’embarqué. Systèmes embarqués. Université Paris-Saclay, 2016. Français. <NNT : 2016SACLS303>. <tel-01552178>.

[19] FILLIAT. Navigation & cartographie Master IAD. [Document électronique]. Navigation topologique, 2008, http://www-master.ufr-info p6.jussieu.fr/2007/Ajouts/Master_esj20_2007_2008/IMG/pdf/DF_Navigation.pdf page 8,9

[20] FILLIAT. Navigation & cartographie Master IAD. [Document électronique]. Navigation métrique, 2008, http://www-master.ufr-info p6.jussieu.fr/2007/Ajouts/Master_esj20_2007_2008/IMG/pdf/DF_Navigation.pdf page 8,9

[21] BHOURI, TOUAZI. Commande prédictive à base de modèle (MPC) pour le trafic urbain bi-modal. CIFA : Conférence Internationale Francophone d'Automatique, Sep 2008, Bucarest, Roumanie. CIFA : Conférence Internationale Francophone d'Automatique, 8p, 2008. <hal-01497156>

[22] RICHALET, RAULT, TESTUD. Model Predictive Heuristic Control : Applications to Industrial processes. Automatica, 14, 413-428, 1978.

[23] LAROCHE, MARTIN, PETIT. Commande par platitude. Equations différentielles ordinaires et aux dérivées partielles. DEA. 2008. <cel-00483381>

[24] ALDWAIHI. Commande non linéaire fondée sur la platitude d'un système de production éolien. Energie électrique. Université de Bretagne occidentale - Brest, 2013. Français. NNT:2013BRES0080. tel-01150682

[25] SARAFIANOS, BOTEANU, IONESCU, IOANNIS, KAKADIARIS, 3D Human Pose Estimation: A Review of the Literature and Analysis of Covariates, Computer Vision and Image Understanding (2016), doi: 10.1016/j.cviu.2016.09.002.

[26] SHOTTON, FITZGIBBON, COOK., SHARP, FINOCCHIO, MOORE, KIPMAN, BLAKE, 2011. Real-time human pose recognition in parts from a single depth image, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1297–1304.

[27] VNECT: Real-time 3D Human Pose Estimation with a Single RGB Camera Mehta, Sridhar, Sotnychenko, Rhodin Shafiei, Weipeng Xu, Casas, Theobalt, Planck. Institute for Informatics (GVV Group) Saarland University 3Universidad Rey Juan Carlos ACM Transactions on Graphics (SIGGRAPH 2017), Los Angeles, USA [http://gvv.mpi-inf.mpg.de/projects/VNect/]

[28] BELAGIANNIS, AMIN, ANDRILUKA, SCHIELE, NAVAB, ILIC. 3D pictorial structures for multiple human pose estimation, in: Proc. IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, 2014, pp. 1669–1676.

[29] SIGAL, ISARD, HAUSSECKER, BLACK. Loose-limbed. People: estimating 3D human pose and motion using nonparametric belief propagation, International Journal of Computer Vision 98 (1) (2012) 15–48

[30] MEHTA, RHODIN, CASAS, SOTNYCHENKO, XU, THEOBALT. Monocular 3d human pose estimation using transfer learning and improved cnn supervision. *arXiv preprint arXiv:1611.09813*, 2016. 2, 3, 5, 6, 7

[31] Towards 3D Human Pose Estimation in the Wild: a Weakly-supervised Approach

[32] LOWE. Object recognition from local scale-invariant features. In IEEE International Conference on Computer Vision (ICCV), volume 2, pages 1150–1157,1999.

[33] SCHWARZ, SCHULZ, BEHNKE. RGB-D object recognition and pose estimation based on pre-trained convolutional neural network features. In IEEE International Conference on Robotics and Automation (ICRA), pages 1329–1335, 2015.

[34] HINTERSTOISSER, CAGNIART, ILIC, STURM, NAVAB, FUA, LEPETIT. Gradient response maps for real-time detection of textureless objects.IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 34(5):876–888, 2012.

[35] BRACHMANN, MICHEL, KRULL, YING YANG, GUMHOLD, ROTHER. Uncertainty-driven 6D pose estimation of objects and scenes from a single RGB image. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 3364–3372, 2016

[36] XIANG, SCHMIDT, NARAYANAN, FOX. PoseCNN: A Convolutional Neural Network for 6D Object : Pose Estimation in Cluttered Scenes. [Document électronique].2018. https://arxiv.org/pdf/1711.00199.pdf.

[37] PAGLIARI, PINTO, REGUZZONI, ROSSI. INTEGRATION OF KINECT AND LOW-COST GNSS FOR OUTDOOR NAVIGATION. [Document électronique]. Prague, 2016,https://www.int-arch-photogramm-remote-sens-spatial-inf-sci.net/XLI B5/565/2016/isprs-archives-XLI-B5-565-2016.pdf

[38] TPE-INTELLIGENCE-ARTIFICIELLE. ia-forte et ia-faible. [En ligne]. http://tpe-intelligence-artificielle.e-monsite.com/pages/i-2-ia-forte-et-ia-faible.html. [Page consultée le 08/02/2019].

[39] INPRINCIPIO. L'IA faible ou descendante. [En ligne]. https://www.inprincipio.xyz/ia-faible/ [Page consultée le 08/02/2019].

[40] EZRATTY. Que devient l'IA symbolique ? . [En ligne]. https://www.oezratty.net/wordpress/2018/que-devient-ia-symbolique/ [Page consultée le 08/02/2019].

[41] VILLEMIN. Intelligence Artificielle, Automate. [Document électronique].2015, http://villemin.gerard.free.fr/Wwwgvmm/Logique/IAautoma.htm

[42] VILLEMIN. Intelligence Artificielle, Systèmes experts. [Document électronique].2015. http://villemin.gerard.free.fr/Wwwgvmm/Logique/IAexpert.htm#top

[43] MALEK. SYSTÈMES EXPERTS - Notes de cours. [Document électronique].2008, http://www.univ-tebessa.dz/fichiers/master/master_2173.pdf

[44] BOUKADOUM. Commande à logique floue. [Document électronique]. http://www.labunix.uqam.ca/~boukadoum_m/EMB7000/Notes/ch8a_commande%20floue.pdf

[45] METOMO. SUPINFO International University. Machine Learning : Introduction à l'apprentissage automatique. [En ligne]. https://www.supinfo.com/articles/single/6041-machine-learning-introduction-apprentissage-automatique [page consultée le 14/02/2019].

[46] TPE-IA. L'intelligence artificielle évolutive. 2016, http://tpe-ia-2016.emonsite.com/pages/l-intelligence-artificielle-evolutive.html

[47] CLOUX. Algorithme génétique : Darwin au service de l'intelligence artificielle [En ligne]. https://toiledefond.net/algorithme-genetique-darwin-intelligence-artificielle/ [page consultée le 15/02/2019].

[48] ARDUPILOT. Community, [En ligne]. http://ardupilot.org/ardupilot/index.html. [Page consultée: 15/05/2019].

[49] ARDUPILOT. ROVER_Mission_Commands, [En ligne]. http://ardupilot.org/rover/docs/common-mavlink-mission-command-messages-mav_cmd.html. [Page consultée: 17/05/2019].

[50] EMREDOGAN. Estimation de pose humaine et reconnaissance d'action par un système multi-robots. Modélisation et simulation. Université de Lyon, 2017. Français. NNT: 2017LYSEI060. tel-01921842

[51] DE LA MARCK, PARDANAUD (2 February 2017). Découvrez le langage JavaScript. Eyrolles. ISBN 978-2-212-30483-1.

[52] SMILKOV, THORAT, ASSOGBA, YUAN, KREEGER, YU, BILESCHI. (2019). TensorFlow. js: Machine Learning for the Web and Beyond. arXiv preprint arXiv:1901.05350.

[53] MEDIUM.Real_Time-Human-Pose-Estimation-in-the-Browser-with-TensorFlow.js.[En ligne] https://medium.com/tensorflow/real-time-human-pose-estimation-in-the-browser-with-tensorflow-js-7dd0bc881cd5. [Page Consultée: 15/06/2019]

[54] AMAZON DOCS. Amazon Machine Learning, [En ligne]. https://docs.aws.amazon.com/fr_fr/machinelearning/latest/dg/evaluating_models.html. [Page consultée: 10/04/2019].

Annexe A

Datasheet PX4

The most advanced development kit for the PX4®autopilot

Product features

- A new and small form factor
- More computing power and 2X the RAM than previous versions
- New sensors with higher temperature stability
- Integrated vibrations isolation
- Increased ease-of-use: pre-installed with most recent PX4 (v1.8)
- Additional ports for better integration and expansion

Product description

Pixhawk®4 is the latest update to the successful family of Pixhawk flight controllers. It is designed and developed in collaboration with Holybro and the PX4 team, optimized to run the full Dronecode stack and comes preinstalled with the latest PX4 firmware (v1.8).

It features the currently most advanced processor technology from STMicroelectronics®, sensor technology from Bosch®,InvenSense®, and a NuttX real-time operating system, delivering incredible performance, flexibility, and reliability for controlling any autonomous vehicle.

The Pixhawk 4's microcontroller now has a 2MB flash memory and 512KB RAM. With the increased power and RAM resources, developers can be more productive and efficient with their development work. More complex algorithms and models can be implemented on the autopilot.

High-performance, low-noise IMUs on board are designed for stabilization applications. Data-ready signals from all sensors are routed to separate interrupt and timer capture pins on the autopilot, permitting precise time-stamping of sensor data. Newly designed vibration isolations enables more accurate readings, allowing vehicles to reach better overall flight performances.

The two external SPI buses and six associated chip select lines allow to add additional sensors and SPI-interfaced payload. There are total of four I2C buses, two dedicated for external use and two grouped with serial ports for GPS/compass modules.

The Pixhawk 4 autopilot development kit is perfect for developers at corporate research labs, startups, and for academics (research, professors, students).

Technical specifications

- Main FMU Processor: STM32F765
    - 32 Bit Arm®Cortex®-M7, 216MHz, 2MB memory, 512KB RAM
- IO Processor: STM32F100
    - 32 Bit Arm®Cortex®-M3, 24MHz, 8KB SRAM
- On-board sensors
    - Accel/Gyro: ICM-20689
    - Accel/Gyro: BMI055
    - Mag: IST8310
    - Barometer: MS5611
- GPS: ublox Neo-M8N GPS/GLONASS receiver; integrated magnetometer IST8310

Interfaces

- 8-16 PWM servo outputs (8 from IO, 8 from FMU)
- 3 dedicated PWM/Capture inputs on FMU
- Dedicated R/C input for CPPM
- Dedicated R/C input for Spektrum / DSM and S.Bus with analog / PWM RSSI input
- Dedicated S.Bus servo output
- 5 general purpose serial ports
    - 2 with full flow control
    - 1 with separate 1.5A current limit
- 3 I2C ports
- 4 SPI buses
    - 1 internal high speed SPI sensor bus with 4 chip selects and 6 DRDYs
    - 1 internal low noise SPI bus dedicated for Barometer with 2 chip selects, no DRDYs
    - 1 internal SPI bus dedicated for FRAM
    - Supports dedicated SPI calibration EEPROM located on sensor module
    - 1 external SPI buses
- Up to 2 CANBuses for dual CAN with serial ESC
    - Each CANBus has individual silent controls or ESC RX-MUX control
- Analog inputs for voltage / current of 2 batteries
- 2 additional analog inputs

Electrical data

Voltage Ratings:

- Power module output: 4.9~5.5V
- Max input voltage: 6V
- Max current sensing: 120A
- USB Power Input: 4.75~5.25V
- Servo Rail Input: 0~36V

Mechanical data

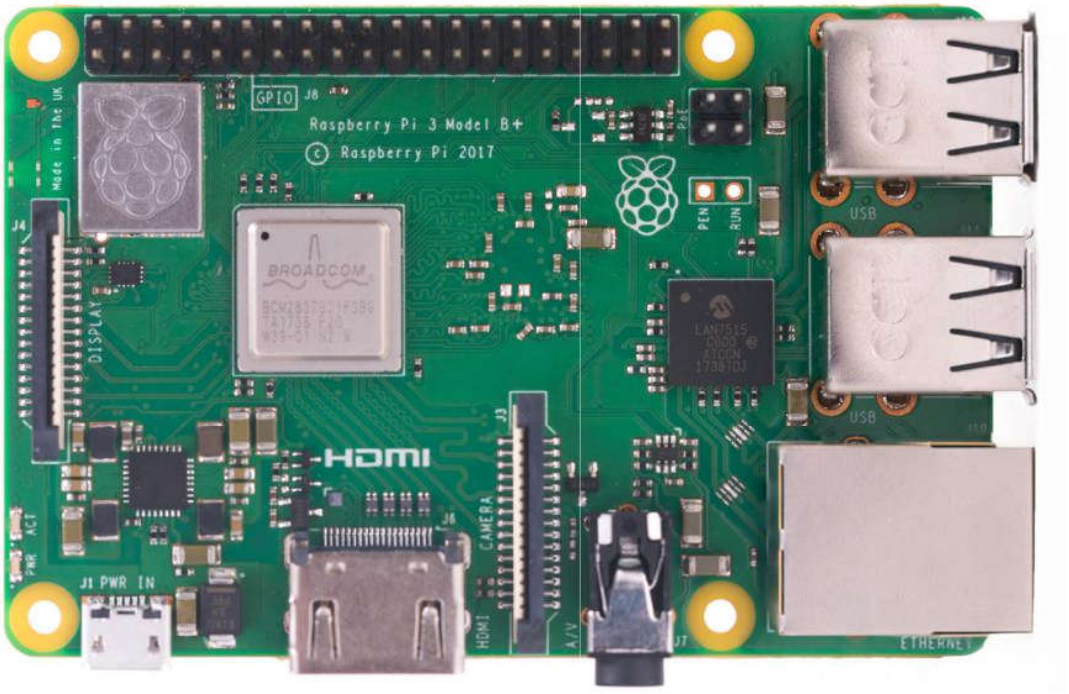
- Dimensions: 44x84x12mm
- Weight: 15.8g

Environmental data, quality & reliability

- Operating temp. ~40~85C
- Storage temp. -40~85C
- CE
- FCC
- RoHS compliant (lead-free)

Annexe B

Overview



The Raspberry Pi 3 Model B+ is the latest product in the Raspberry Pi 3 range, boasting a 64-bit quad core processor running at 1.4GHz, dual-band 2.4GHz and 5GHz wireless LAN, Bluetooth 4.2/BLE, faster Ethernet, and PoE capability via a separate PoE HAT

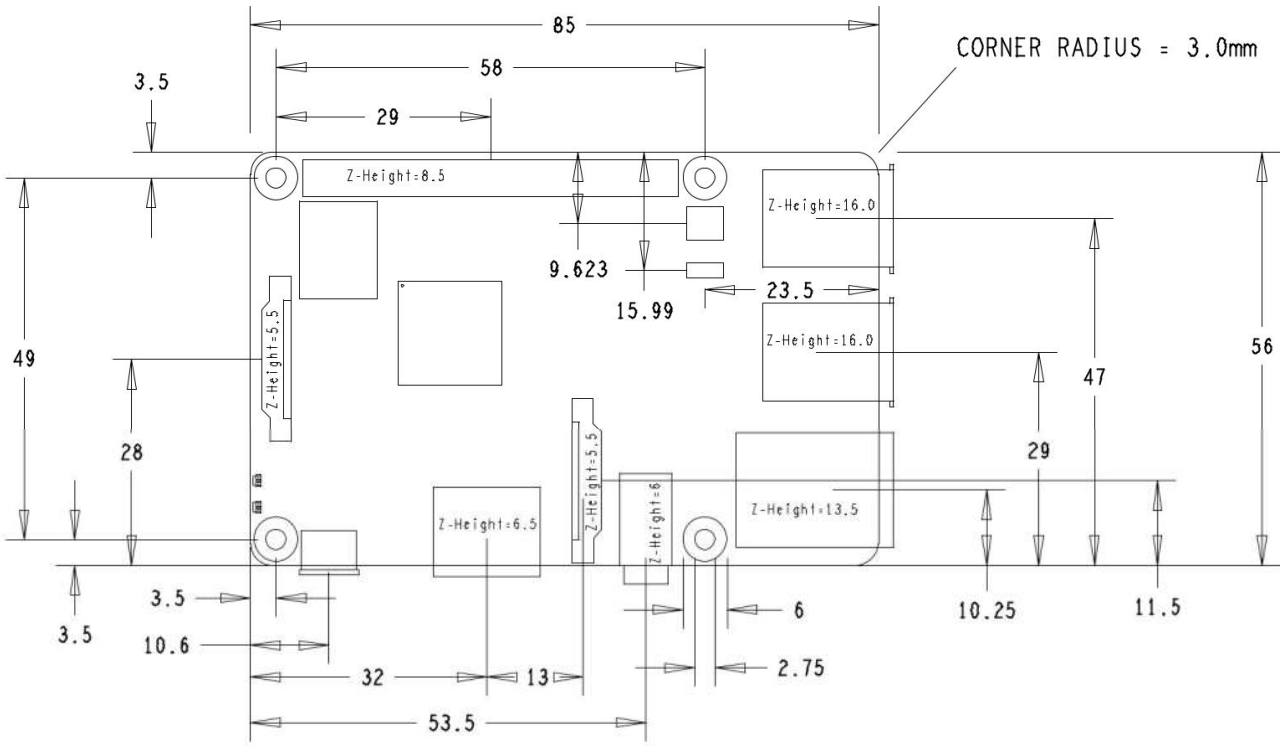
The dual-band wireless LAN comes with modular compliance certification, allowing the board to be designed into end products with significantly reduced wireless LAN compliance testing, improving both cost and time to market.

The Raspberry Pi 3 Model B+ maintains the same mechanical footprint as both the Raspberry Pi 2 Model B and the Raspberry Pi 3 Model B.

Specifications

Processor:	Broadcom BCM2837B0, Cortex-A53 64-bit SoC @ 1.4GHz
Memory:	1GB LPDDR2 SDRAM
Connectivity:	■ 2.4GHz and 5GHz IEEE 802.11.b/g/n/ac wireless LAN, Bluetooth 4.2, BLE ■ Gigabit Ethernet over USB 2.0 (maximum throughput 300Mbps) ■ 4 × USB 2.0 ports
Access:	Extended 40-pin GPIO header
Video & sound:	■ 1 × full size HDMI ■ MIPI DSI display port ■ MIPI CSI camera port ■ 4 pole stereo output and composite video port
Multimedia:	H.264, MPEG-4 decode (1080p30); H.264 encode (1080p30); OpenGL ES 1.1, 2.0 graphics
SD card support:	Micro SD format for loading operating system and data storage
Input power:	■ 5V/2.5A DC via micro USB connector ■ 5V DC via GPIO header ■ Power over Ethernet (PoE)–enabled (requires separate PoE HAT)
Environment:	Operating temperature, 0–50°C
Compliance:	For a full list of local and regional product approvals, please visit **www.raspberrypi.org/products/raspberry-pi-3-model-b+**
Production lifetime:	The Raspberry Pi 3 Model B+ will remain in production until at least January 2023.

Physical specifications



Warnings

- This product should only be connected to an external power supply rated at 5V/2.5A DC. Any external power supply used with the Raspberry Pi 3 Model B+ shall comply with relevant regulations and standards applicable in the country of intended use.
- This product should be operated in a well-ventilated environment and, if used inside a case, the case should not be covered.
- Whilst in use, this product should be placed on a stable, flat, non-conductive surface and should not be contacted by conductive items.
- The connection of incompatible devices to the GPIO connection may affect compliance, result in damage to the unit, and invalidate the warranty.
- All peripherals used with this product should comply with relevant standards for the country of use and be marked accordingly to ensure that safety and performance requirements are met. These articles include but are not limited to keyboards, monitors, and mice when used in conjunction with the Raspberry Pi.
- The cables and connectors of all peripherals used with this product must have adequate insulation so that relevant safety requirements are met.

Safety instructions

To avoid malfunction of or damage to this product, please observe the following:

- Do not expose to water or moisture, or place on a conductive surface whilst in operation.
- Do not expose to heat from any source; the Raspberry Pi 3 Model B+ is designed for reliable operation at normal ambient temperatures.
- Take care whilst handling to avoid mechanical or electrical damage to the printed circuit board and connectors.

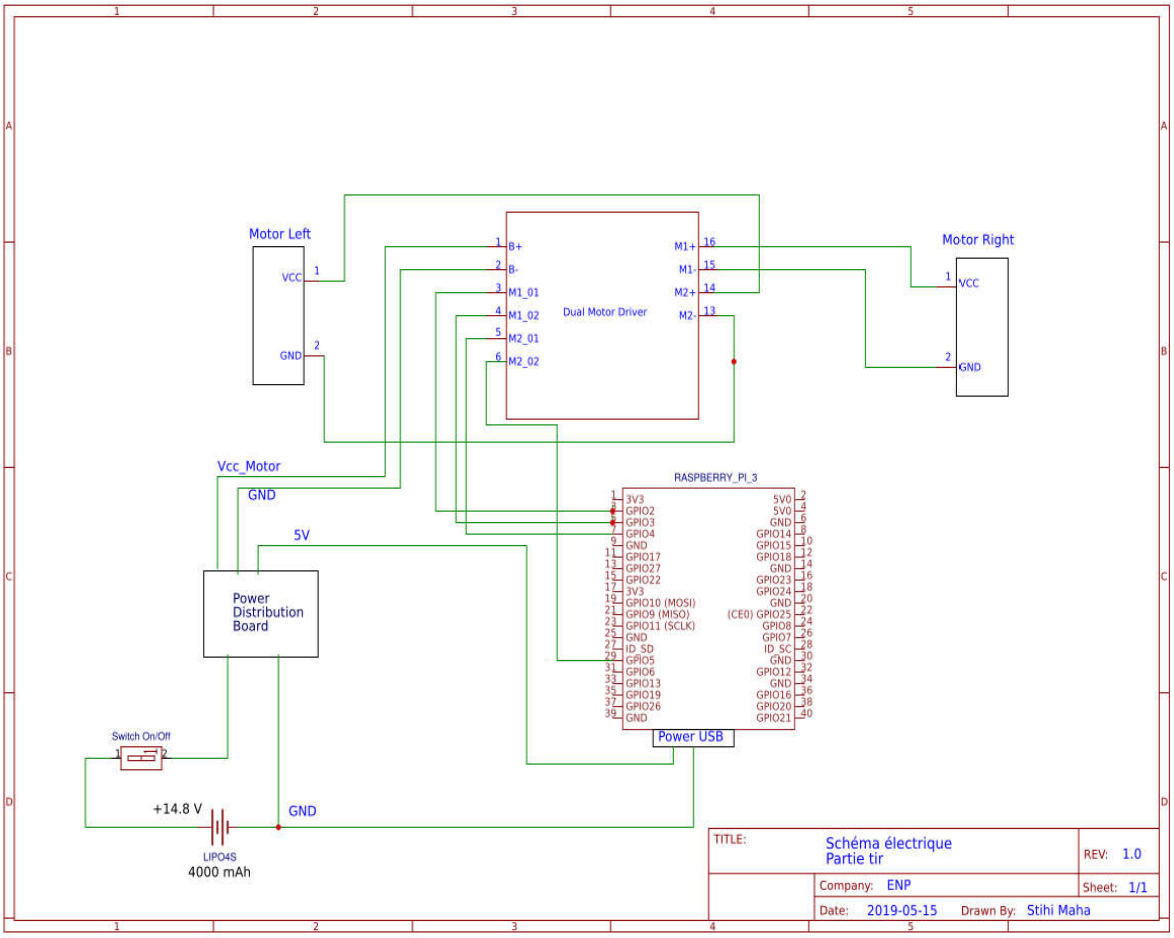
Annexe C

Programme de Commande du Simulateur

```
import time
from pymavlink import mavutil

mavutil.set_dialect("ardupilotmega")

autopilot = mavutil.mavlink_connection('tcp:localhost:5762')

msg = None

# wait for autopilot connection
while msg is None:
        msg = autopilot.recv_msg()

print msg

# The values of these heartbeat fields is not really important here
# I just used the same numbers that QGC uses
# It is standard practice for any system communicating via mavlink emit the HEARTBEAT message
autopilot.mav.heartbeat_send(
6, # type
8, # autopilot
192, # base_mode
0, # custom_mode
4, # system_status
3  # mavlink_version
)

autopilot.mav.command_long_send(
1, # autopilot system id
1, # autopilot component id
400, # command id, ARM/DISARM
0, # confirmation
1, # arm!
0,0,0,0,0,0 # unused parameters for this command
)

time.sleep(2)

autopilot.set_mode_manual()
autopilot.mav.rc_channels_override_send(autopilot.target_system, autopilot.target_component, 0, 2000, 1000, 0, 0, 0, 0, 0)
```

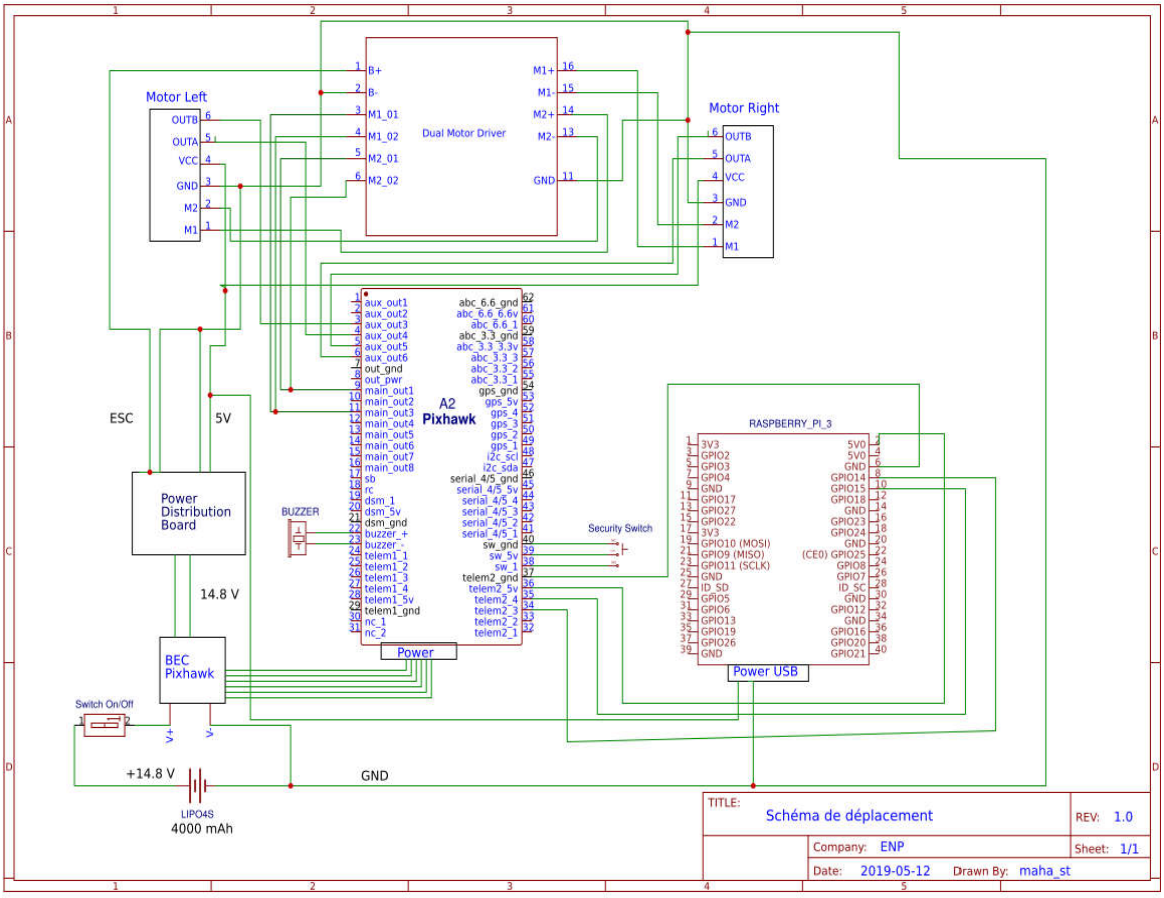
Annexe D

Circuit Electrique Système de Tir



Annexe E

Circuit Electrique Système de déplacement



Annexe F

Programme de Commande Tir

```
# Python Pittching Program
# in1 & in2 for the right motor (front view)
# in3 & in4 for the left motor (front view)
# p speed regultion for right motor
# p2 speed regultion for left motor

import RPi.GPIO as GPIO
from time import sleep

in1 = 24
in2 = 23
#in3 = *
#in4 = *
en = 25
#en2 =

def init():
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(in1,GPIO.OUT)
    GPIO.setup(in2,GPIO.OUT)
    GPIO.setup(en,GPIO.OUT)
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(in1,GPIO.OUT)
    GPIO.setup(in2,GPIO.OUT)
    GPIO.setup(en,GPIO.OUT)
    GPIO.output(in1,GPIO.LOW)
    GPIO.output(in2,GPIO.LOW)
    p=GPIO.PWM(en,1000)
    p.start(25)
    p2=GPIO.PWM(en2,1000)
    p2.start(25)


 def pitch(vitesse,te,angle):

     if angle==45:
         print("Pitching with an angle of 45")
         if vitesse == 'l':
             p.ChangeDutyCycle(50)
             p2.ChangeDutyCycle(30)
             GPIO.output(in1,GPIO.HIGH)
             GPIO.output(in2,GPIO.LOW)
             GPIO.output(in3,GPIO.HIGH)
             GPIO.output(in4,GPIO.LOW)
             print("with a low speed ")
             time.sleep(15)

         elif vitesse=='m':
             p.ChangeDutyCycle(70)
             p2.ChangeDutyCycle(50)
             GPIO.output(in1,GPIO.HIGH)
             GPIO.output(in2,GPIO.LOW)
             GPIO.output(in3,GPIO.HIGH)
             GPIO.output(in4,GPIO.LOW)
             print("with a medium speed ")
             time.sleep(15)

         elif vitesse=='h':
             p.ChangeDutyCycle(90)
             p2.ChangeDutyCycle(70)
             GPIO.output(in1,GPIO.HIGH)
             GPIO.output(in2,GPIO.LOW)
             GPIO.output(in3,GPIO.HIGH)
             GPIO.output(in4,GPIO.LOW)
             print("with a high speed ")
             time.sleep(15)
```

```
##########################################################################
    if angle==90:
        print("Pitching with an angle of 45")
        if vitesse == 'l':
            p.ChangeDutyCycle(30)
            p2.ChangeDutyCycle(30)
            GPIO.output(in1,GPIO.HIGH)
            GPIO.output(in2,GPIO.LOW)
            GPIO.output(in3,GPIO.HIGH)
            GPIO.output(in4,GPIO.LOW)
            print("with a low speed ")
            time.sleep(15)

        elif vitesse=='m':
            p.ChangeDutyCycle(60)
            p2.ChangeDutyCycle(60)
            GPIO.output(in1,GPIO.HIGH)
            GPIO.output(in2,GPIO.LOW)
            GPIO.output(in3,GPIO.HIGH)
            GPIO.output(in4,GPIO.LOW)
            print("with a medium speed ")
            time.sleep(15)

        elif vitesse=='h':
            p.ChangeDutyCycle(80)
            p2.ChangeDutyCycle(80)
            GPIO.output(in1,GPIO.HIGH)
            GPIO.output(in2,GPIO.LOW)
            GPIO.output(in3,GPIO.HIGH)
            GPIO.output(in4,GPIO.LOW)
            print("with a high speed ")
            time.sleep(15)


##########################################################################
    if angle==135:
        print("Pitching with an angle of 45")
        if vitesse == 'l':
            p.ChangeDutyCycle(30)
            p2.ChangeDutyCycle(50)
            GPIO.output(in1,GPIO.HIGH)
            GPIO.output(in2,GPIO.LOW)
            GPIO.output(in3,GPIO.HIGH)
            GPIO.output(in4,GPIO.LOW)
            print("with a low speed ")
            time.sleep(15)

        elif vitesse=='m':
            p.ChangeDutyCycle(50)
            p2.ChangeDutyCycle(70)
            GPIO.output(in1,GPIO.HIGH)
            GPIO.output(in2,GPIO.LOW)
            GPIO.output(in3,GPIO.HIGH)
            GPIO.output(in4,GPIO.LOW)
            print("with a medium speed ")
            time.sleep(15)

        elif vitesse=='h':
            p.ChangeDutyCycle(70)
            p2.ChangeDutyCycle(90)
            GPIO.output(in1,GPIO.HIGH)
            GPIO.output(in2,GPIO.LOW)
            GPIO.output(in3,GPIO.HIGH)
            GPIO.output(in4,GPIO.LOW)
            print("with a high speed ")
            time.sleep(15)
```

Annexe G

Programme de commande Déplacement

```
class Move(State):
    def __init__(self, FSM,  shoot_pml=0,shoot_pmr=0, move_pml=0, move_pmr=0, shoot_state=False, move_state=True):
        super(Move, self).__init__(FSM, shoot_pml,shoot_pmr, move_pml, move_pmr, shoot_state, move_state)

    def Enter(self):
       print "State Move activated !"

    def Execute(self):
       mavutil.set_dialect("ardupilotmega")
       autopilot = mavutil.mavlink_connection('tcp:localhost:5762')
       msg = None
# wait for autopilot connection
       while msg is None:
           msg = autopilot.recv_msg()
       print msg
# The values of these heartbeat fields is not really important here
# I just used the same numbers that QGC uses
# It is standard practice for any system communicating via mavlink emit the HEARTBEAT message at 1Hz! Your autopilot may not behave the way you want otherwise!

       autopilot.mav.heartbeat_send(
       6, # type
       8, # autopilot
       192, # base_mode
       0, # custom_mode
       4, # system_status
       3  # mavlink_version
       )


        autopilot.mav.command_long_send(
        1, # autopilot system id
        1, # autopilot component id
        400, # command id, ARM/DISARM
        0, # confirmation
        1, # arm!
        0,0,0,0,0,0 # unused parameters for this command
        )
        time.sleep(2)
        autopilot.set_mode_manual()
        autopilot.mav.rc_channels_override_send(autopilot.target_system, autopilot.target_component, 0, 2000, 2000, 0, 0, 0, 0, 0)
        time.sleep(10)
        global inputs
        inputs="1100"
        if  inputs == "1100":
            self.FSM.ToTransition("toShoot")
            self.FSM.Execute()
    def Exit(self):
        print "State Move disactivated !"
```

Annexe H

Programme de commande Stockage

```
# Python Ball Releasing System Program
# For the prototype we made, this system is manual, we replaced the stepper motor
# with a LED to indicate the exact time to release the ball from the storage system


import RPi.GPIO as GPIO
from time import sleep

LED_Ball = 12

def init ():
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(LED_Ball,GPIO.OUT)
    GPIO.setup(LED_Ball,GPIO.LOW)

def ball_release(demande):
    init()

    if demande == 1:
        GPIO.output(LED_Ball,GPIO.HIGH)
        print("Ball released")

    else:
        print ("No ball demand !")
```

Annexe I

Algorithme global de commande en Python

```
import time
from pymavlink import mavutil

inputs ="0000"
"""##================================================================================================================##"""
##Transitions

class Transition(object):
    def __init__(self, toState):
        self.toState = toState
    def Execute(self):
        print "Transitioning..."

"""##================================================================================================================##"""
##States
class State(object):
    def __init__(self, FSM, shoot_pml=0,shoot_pmr=0, move_pml=0, move_pmr=0, shoot_state=False, move_state=False):


        self.FSM = FSM
        self.shoot_pml = shoot_pml
        self.shoot_pmr = shoot_pmr
        self.move_pml = move_pml
        self.move_pmr = move_pmr
        self.shoot_state = shoot_state
        self.move_state = move_state

    def Enter(self):
        pass
    def Execute(self):
        pass
    def Exit(self):
        pass

class Initialisation(State):
    def __init__(self, FSM):
        super(Initialisation, self).__init__(FSM)

    def Enter(self):
         pass
    def Execute(self):
         if inputs == "0000":
             print "State Initialisation activated !"
         if inputs == "1000":
             print "Nouvel Exercice !"
             self.FSM.ToTransition("toMove")
             self.FSM.Execute()


    def Exit(self):
         print "State Initialisation disactivated !"

class Move(State):
    def __init__(self, FSM,  shoot_pml=0,shoot_pmr=0, move_pml=0, move_pmr=0, shoot_state=False, move_state=True):
        super(Move, self).__init__(FSM, shoot_pml,shoot_pmr, move_pml, move_pmr, shoot_state, move_state)
```

```
    def Enter(self):
       print "State Move activated !"

    def Execute(self):
       mavutil.set_dialect("ardupilotmega")
       autopilot = mavutil.mavlink_connection('tcp:localhost:5762')
       msg = None
# wait for autopilot connection
       while msg is None:
            msg = autopilot.recv_msg()
       print msg
# The values of these heartbeat fields is not really important here
# I just used the same numbers that QGC uses
# It is standard practice for any system communicating via mavlink emit the HEARTBEAT message at 1Hz! Your autopilot may not behave the way
you want otherwise!

       autopilot.mav.heartbeat_send(
       6, # type
       8, # autopilot
       192, # base_mode
       0, # custom_mode
       4, # system_status
       3  # mavlink_version
       )



        autopilot.mav.command_long_send(
        1, # autopilot system id
        1, # autopilot component id
        400, # command id, ARM/DISARM
        0, # confirmation
        1, # arm!
        0,0,0,0,0,0 # unused parameters for this command
        )
        time.sleep(2)
        autopilot.set_mode_manual()
        autopilot.mav.rc_channels_override_send(autopilot.target_system, autopilot.target_component, 0, 2000, 2000, 0, 0, 0, 0, 0)
        time.sleep(10)
        global inputs
        inputs="1100"
        if  inputs == "1100":
            self.FSM.ToTransition("toShoot")
            self.FSM.Execute()
    def Exit(self):
        print "State Move disactivated !"

class Shoot(State):

    def Enter(self):
        print "State Shoot activated !"

    def Execute(self):
        global inputs
        inputs = "0110"
        if  inputs == "0110":
            self.FSM.ToTransition("toAnalize")
```

```
            self.FSM.Execute()
    def Exit(self):
        print "State Shoot disactivated !"

class Analize(State):

    def Enter(self):
        print "State Analize activated !"

    def Execute(self):
        global inputs
        inputs = "0001"
        if  inputs == "0001":
            self.FSM.ToTransition("toAnalize")
            self.FSM.Execute()
        if  inputs == "1101":
            self.FSM.ToTransition("toShoot")
            self.FSM.Execute()
        if  inputs == "1001":
            self.FSM.ToTransition("toMove")
            self.FSM.Execute()

    def Exit(self):
        print "State Analize disactivated !"

"""##==========================================================================================================##"""
## Finite State Machine :


class FSM(object):
    def __init__(self, character):
        self.char=character
        self.states = {}
        self.transitions = {}
        self.curState = None
        self.trans = None

    def AddState(self, stateName, state):
        self.states[stateName] = state

    def AddTransition(self, transName, transition):
        self.transitions[transName] = transition

    def SetState(self, stateName):
        self.curState = self.states[stateName]

    def ToTransition(self, toTrans):
        self.trans = self.transitions[toTrans]

    def Execute(self):
        if(self.trans):
              self.curState.Exit()
              self.trans.Execute()
              self.SetState(self.trans.toState) #toState ?
              self.curState.Enter()
              self.trans = None
        self.curState.Execute()
```

```
'""##==========================================================================================================##"""
## IMPLEMENTATION

Char= type("Char",(object,),{})

class Robot(Char):
    def __init__(self):
        self.FSM=FSM(self)

        ##STATES
        self.FSM.AddState("Initialisation", Initialisation(self.FSM))
        self.FSM.AddState("Move", Move(self.FSM))
        self.FSM.AddState("Analize", Analize(self.FSM))
        self.FSM.AddState("Shoot", Shoot(self.FSM))

        ##TRANSITIONS
        self.FSM.AddTransition("toInitialisation", Transition("Initialisation"))
        self.FSM.AddTransition("toMove", Transition("Move"))
        self.FSM.AddTransition("toShoot", Transition("Shoot"))
        self.FSM.AddTransition("toAnalize", Transition("Analize"))

        self.FSM.SetState("Initialisation")

    def Execute(self):
        self.FSM.Execute()

    def SetState(self, stateName):
        self.FSM.SetState(stateName)
```