Master 1 Informatique et Ingénierie des Systèmes Complexes (IISC)

Université de Cergy-Pontoise

Projet de synthèse

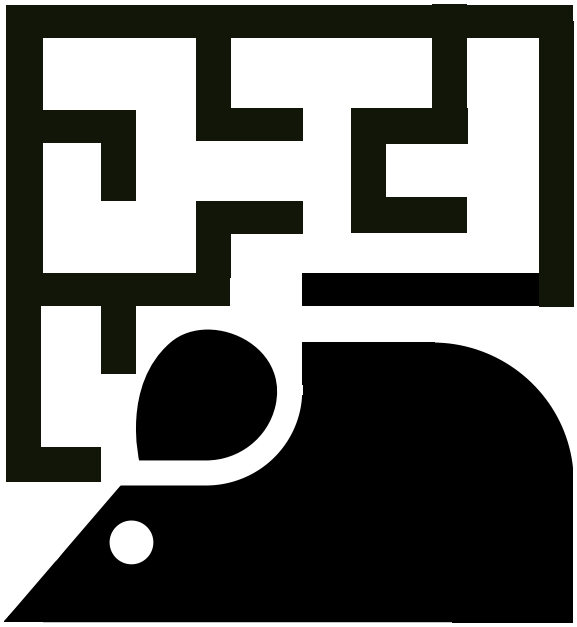
Résolution de labyrinthes par véhicule intelligent

Micromouse

rapporteur :

Auteurs :
Djahid ABDELMOUMENE
Amine AGRANE
Ishak AYAD
Donald LAY

Tuteur technique :
Pr. Alexandre PITTI
Encadrant de gestion de projet :
Pr. Tianxiao LIU



Rendu le
7 février 2020

Remerciements

Résumé et abstract

Résumé

Abstract

Table des matières

Table des figures

Chapitre 1

Introduction

Chapeau du chapitre

1.1 Contexte du projet

1.2 Objectifs du projet

Chapeau

1.3 Mise en scénario

Étant donné que notre projet est composé de plusieurs briques applicatives, nous avons imaginé une mise en situation de notre produit permettant une démonstration globale des différentes fonctionnalités de ce dernier.

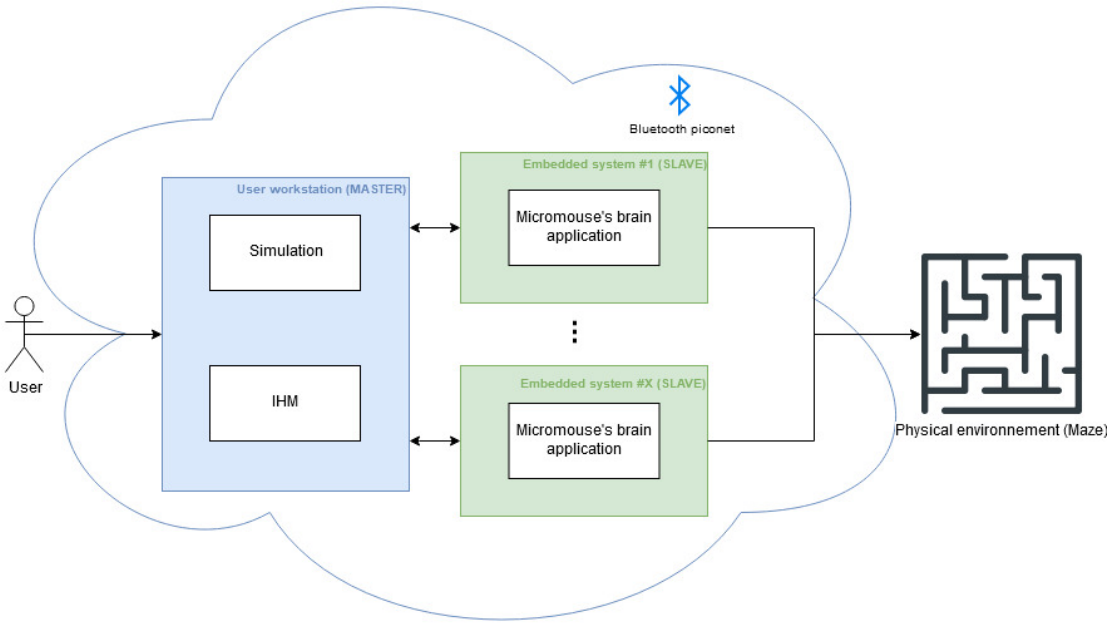


FIGURE 1.1 – Schéma d'infrastructure globale du projet.

La mise en situation que nous avons imaginée se déroule en trois étapes. Pour se faire, l'utilisateur aura accès à une interface depuis laquelle il pourra piloter les différentes phases de la démonstration (User workstation indiqué sur la Fig.[1.1]) :

1. À l'aide du module de simulation, l'utilisateur pourra simuler un environnement physique en paramétrant ou générant automatiquement un labyrinthe. L'intelligence, dont le code sera intégré au module de simulation, s'opérera sur cet environnement simulé. L'utilisateur pourra suivre les mouvements d'une micromouse virtuelle sur l'interface graphique ; mouvements qui seront synchronisés avec ceux d'une micromouse physique. Ainsi, l'utilisateur aura une double vision de l'intelligence du produit : une vision concrète avec le déplacement de la micromouse physique et une autre, plus théorique, avec l'interface graphique sur laquelle seront affichées les différentes métriques nécessaires au guidage de la micromouse ;

2. Une démonstration sur un environnement physique aura ensuite lieu sur un environnement que nous aurons préalablement construit. Cette démonstration se déroulera en deux phases : tout d'abord, une phase de mappage aura lieu dans laquelle la micromouse se déplacera dans le labyrintheen partant d'un point de départ jusqu'à une arrivée préétablie afin d'obtenir un modèle de l'environnement. Ensuite, depuis ce même point de départ, la micromouse sera amenée à se rendre au même point d'arrivée en empruntant un chemin unique qu'il aura jugé être le plus court ;

3. Enfin, sur ce même environnement physique, une démonstration mettant en scène nos quatre souris sera faite. Les micromouses se confronteront par équipe de deux dans un jeu <À DEFINIR – à priori sur un jeu type Pacman : i.e. 1v3 dans lequel la souris individuelle devra parcourir toutes les cases du labyrinthe sans rencontrer une des autres souris dans son chemin ou bien capture de drapeau >. La mise en scénario multi-micromouse pourra également être simulée depuis le module de simulation.

1.4 Organisation du rapport

— Un cahier des charges (2.1) de notre projet sera disponible. Il tirera un portrait de ce qui a déjà été fait dans le domaine dans une étude de marché. Par ailleurs, une liste des fonctionnalités attendues ainsi qu'une description de la conception globale du projet sera tenue. Cette dernière rendra compte des différentes vues (utilisateur, technique et logicielle) du projet et pourront mettre en évidence les différentes problématiques qui seront traitées au cours des chapitres techniques. Enfin, une dernière section concernera l'environnement de travail au cours de laquelle sera dressée une liste exhaustive du matériel ainsi que des logiciels et outils utilisés.

— Les problématiques soulevées précédemment dans le cahier des charges seront respectivement traitées dans des sections techniques. Ces sections, qui constituent les chapitres techniques, auront pour sujets principaux l'intelligence embarquée dans nos micromouses comprenant les algorithmes de mappage et de plus court chemin (4.1, 5.1) ainsi que les algorithmes de contrôle (6.1) mettant en oeuvres les capteurs du véhicule ; les protocoles réseaux de communication entre les différentes briques du projets (**??**).

— Un chapitre concernant le rendu final (7.1) figurera également sur ce présent rapport. Il aura vocation à fournir le reflet du résultat attendu du projet pour l'utilisateur final. Ainsi, il présentera les différents outils terminaux que l'utilisateur sera amené à manipuler pour faire fonctionner le produit ainsi qu'une batterie de tests qui témoignent du bon fonctionnement du dispositif livré.

— Enfin, un chapitre sera consacré à la gestion projet (8.1) dans laquelle figureront les méthodes de travail employées ainsi que la répartition des tâches au cours de ce projet. Par ailleurs, une partie de ce chapitre aura une vocation didactique sur le sujet. Ainsi, seront évoqués <Choix1> ainsi que <Choix2>.

Chapitre 2

Présentation et spécification du projet

Ce chapitre présente le cahier des charges (CDC) du projet et les spécifications techniques, il contient les principaux éléments nécessaires pour comprendre le positionnement marketing du futur produit et la conception technique. Il mentionne également des informations relatives au développement du produit dans sa globalité et son architecture.

2.1 Étude du marché

Une compétition Micromouse est un évènement durant lequel plusieurs équipes s'affrontent en opposant leur robot souris, l'objectif étant de résoudre un labyrinthe le plus rapidement possible. Ces compétitions sont organisées depuis la fin des années 1970 et ont lieu un peu partout autour du monde.

Historique et Origine Le magazine "IEE Spectrum magazine" est à l'origine de l'apparition des compétitions Micromouse. C'est un magazine anglophone qui vise à couvrir les tendances et avancées majeures dans les domaines des technologies, de l'ingénierie et des sciences. En 1977, le magazine met au défi ses lecteurs en leurs proposant de concevoir et construire un robot "micromouse" pouvant résoudre un labyrinthe. Le robot devait agir selon sa propre logique et résoudre un labyrinthe imaginé par les rédacteurs du magazine IEE Spectrum. Cette compétition fut appelée 'The Amazing Micromouse Competition'. Elle prit place à New York lors de la conférence nationale sur l'informatique en 1979. Cet évènement couvert par de grands médias tels que les chaînes CBS et ABC ainsi que le journal The New York Times. Le grand succès de cette première édition ainsi que sa large médiatisation participa au gain de popularité des compétitions Micromouse pour les années à venir. En quelques années, le défi des micromouses était devenu un événement mondial. En 1980, le premier concours européen a eu lieu à Londres, suivi un an plus tard par un une compétition organisée à Paris. Le Japon a annoncé l'organisation du premier tournoi mondial de Micromouse qui aura lieu à Tsubaka en août 1985. La même année, l'IEE (devenue l'Institution of Engineering and Technology) a organisé un concours international à Londres. Au début des années 1990 des clubs Micromouse ont commencé à faire leur apparition au sein des écoles et universités. La "IEE Micromouse Competition" bénéficie au aujourd'hui d'une grande popularité auprès des étudiants du domaine informatique ou électronique.

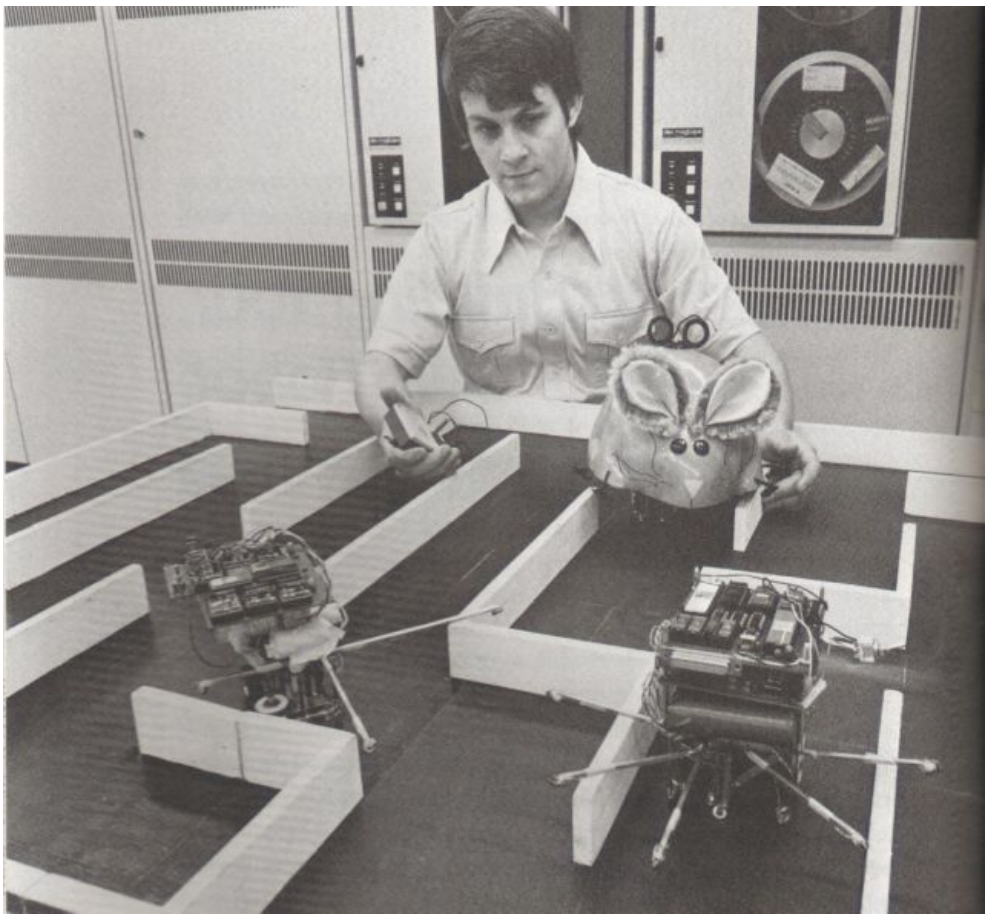


FIGURE 2.1 – Ancienne génération de Micromouse (Pacific Northwest National Laborator, 1980)

Organistaions des compétitions Au vu du nombre important de compétitions Micromouse ayant lieu chaque année, certaines règles et conditions peuvent varier d'une compétition à l'autre. Néanmoins, il existe des éléments généralement communs à toutes les compétitions :

— Le labyrinthe standard de micromouse mesure environ $2.5m^2$ et consiste en une grille de cellules (matrice) de 16 sur 16. Chaque micromouse est autorisée à effectuer un certain nombre de recherches afin de déterminer le chemin le plus court vers l'objectif. La micromouse devra garder une trace de sa position, découvrir les murs en explorant le labyrinthe et détecter quand elle a atteint son objectif (sortie du labyrinthe).

— La notation est basée à la fois sur la course la plus rapide et sur le temps total consommé pour toutes les courses. Les concurrents n'ont pas le droit de communiquer avec leur micromouse. Il existe de nombreuses versions de règles selon la compétition, et il existe un certain nombre de variations mineures sur la façon dont le score de la souris est déterminé.

2.2 Fonctionnalités attendues

Dans la section [1.3] le lecteur peut observer que le produit proposé fournit plusieurs options ie :fonctionnalités pour tester et manipuler différentes entités dans les deux environnements, physique et simulé dans cette section nous allons introduire ces fonctionnalités et expliquer les cas d'utilisation de ces derniers.

2.2.1 Vue d'ensemble du système

Le produit permet à l'utilisateur de simuler un véhicule autonome se déplaçant dans un labyrinthe et en essayant de trancher le point d'arriver, le labyrinthe est créé soit manuellement ou en insérant un fichier de configuration, ainsi la simulation pourra se lancer et l'utilisateur pourra observer le déplacement de la micro mouse et en même temps d'observer le journal et les statistiques de la simulation qui sont modifiées et stockées au fur et à mesure de la simulation, ainsi que les valeurs des différents composants de l'engin.

En outre, l'utilisateur du produit peut modifier le labyrinthe réel et d'y mettre le véhicule dedans pour lancer une observation physique, ainsi qu'il pourra voir sur sa machine sur un autre logiciel de visualisation le déplacement et les valeurs retournées par l'engin.

Le schéma suivant illustre le diagramme de cas d'utilisation du produit.

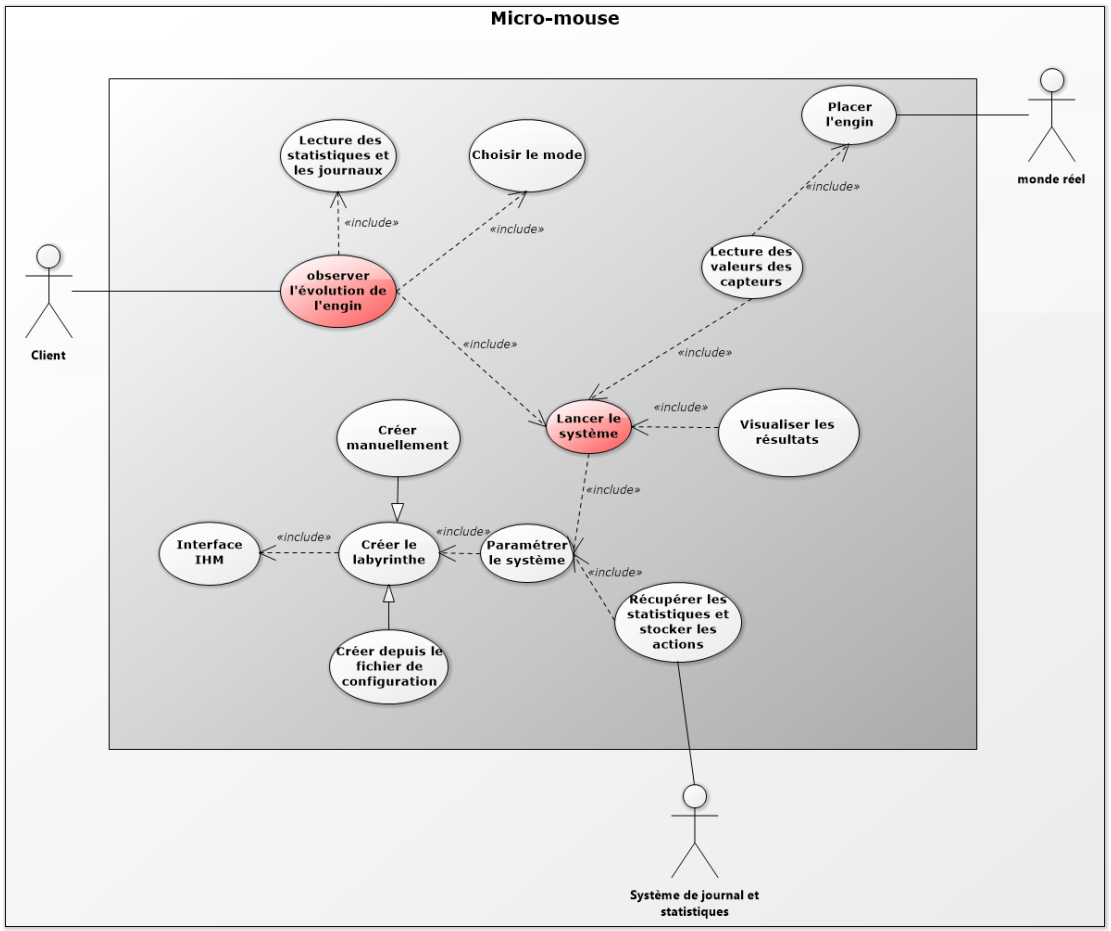


FIGURE 2.2 – Le diagramme de cas d'utilisation du système

2.2.2 Services fournis par le produit

Simulation : La simulation offre à l'utilisateur un ensemble de fonctionnalités pour paramétrer le monde, comme la création du labyrinthe soit avec un fichier de configuration ou manuellement et le choix de la complexité de l'algorithme de recherche du plus court chemin, en outre l'utilisateur peut visualiser le journal de la simulation et les statistiques sur la partie information sur la simulation, finalement l'utilisateur pourra observer les valeurs des différents composants de la micro mouse.

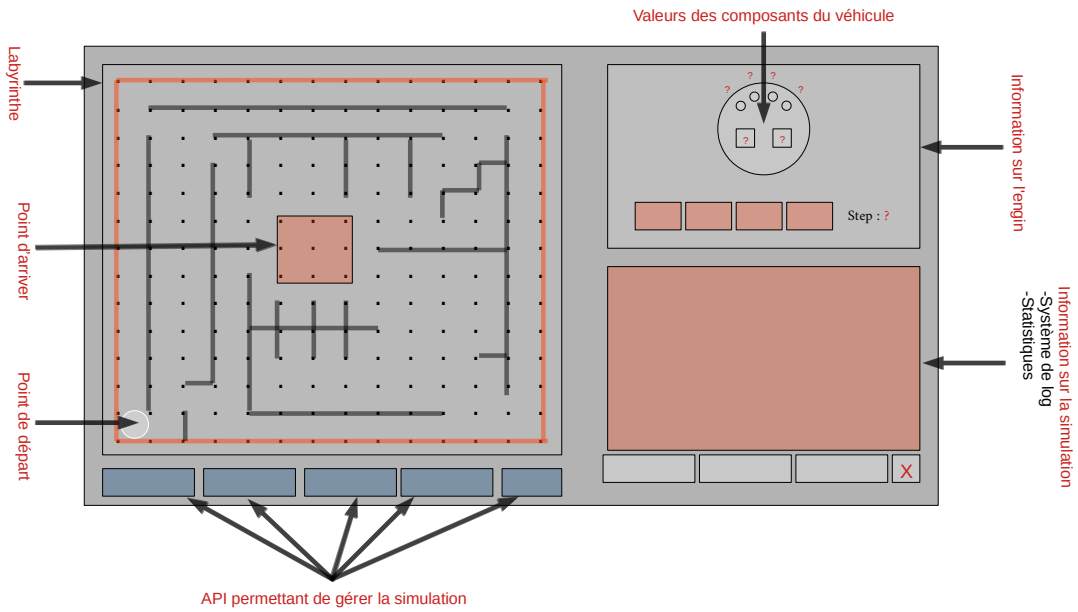


FIGURE 2.3 – Interface homme-machine pour la simulation

Véhicule réel : Notre système sera réalisé sur un prototype minimisé de micromouse appelé quarter-micromouse, le labyrinthe est constitué d'un quadrillage de cellules 8×8 de 180 mm de côté avec des murs de 50 mm de haut les souris sont des robots complètement autonomes qui doivent trouver leur chemin d'une position de départ prédéterminée à la zone centrale du labyrinthe sans aide. La souris doit savoir où elle se trouve, découvrir les murs en explorant, cartographier le labyrinthe et détecter quand elle a atteint son but. Une fois l'objectif atteint, la souris effectuera généralement des recherches supplémentaires dans le labyrinthe jusqu'à ce qu'elle ait trouvé un itinéraire optimal du début à la fin. Une fois que l'itinéraire optimal a été trouvé, la souris exécute cet itinéraire dans les plus brefs délais.

2.2.3 Fonctionnalités supplémentaires

Visualisation en temps réel : La visualisation des résultats au fur et à mesure du temps, au moment du lancement du système l'utilisateur peut lancer un logiciel, c'est le logiciel de visualisation des résultats de l'expérience, il permet à l'utilisateur de visualiser la vue du véhicule et les valeurs des capteurs au fur et à mesure de l'exploration, aussi de visualiser le journal de la simulation et les statistiques.

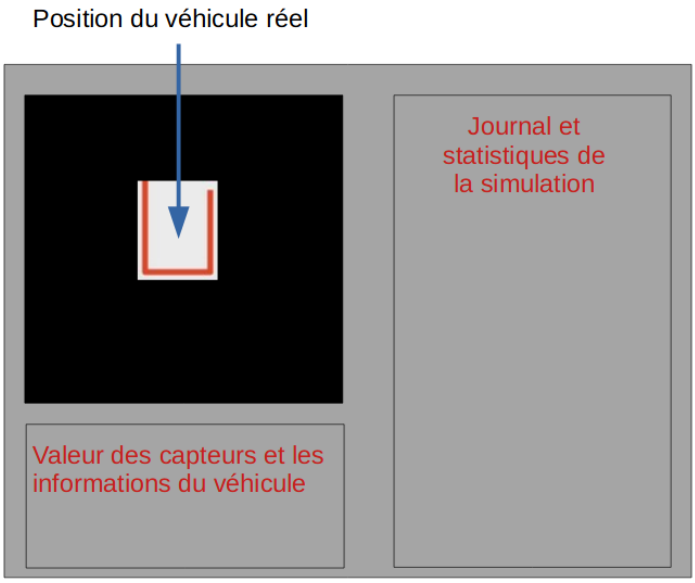


FIGURE 2.4 – Maquette du logiciel de visualisation

Multi-véhicule : Un mini-jeu est mis en place pour les deux univers, simulé et réel permettant d'introduire la notion de communication entre différents véhicules, le jeu consiste à parcourir toutes les cases du labyrinthe et à la fin se situer dans la case d'arriver par un véhicule autonome intelligent en évitant les autres véhicules gradient communiquant entre eux. Ceci en utilisant les modules déjà abordés tels que la simulation et l'univers réel.

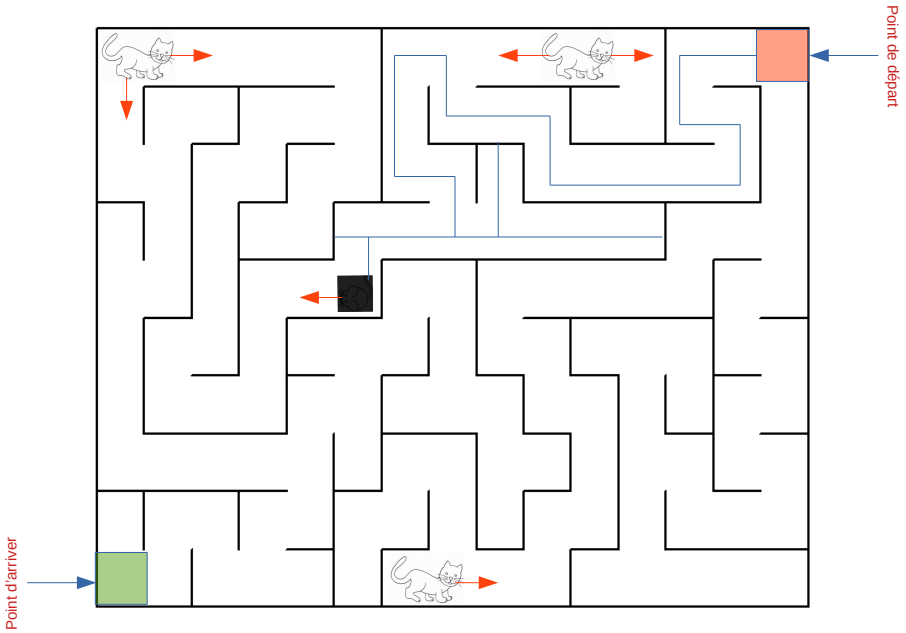


FIGURE 2.5 – Pac-Man avec l'entité micro mouse (Pac-Micro)

La figure [2.5] représente une illustration du mini-jeu, les trois chats représentent les gardiens du labyrinthe et le carré noir représente le véhicule intelligent qui doit résoudre le labyrinthe en parcourant toutes les cases tout en évitant de croiser un gradient.

2.3 Conception globale du projet

après avoir illustré les différentes fonctionnalités du système sur la section [2.2], il va falloir expliquer plus en détails la conception de ces fonctionnalités, en séparent la vue pour l'utilisateur on masquera les détails techniques informatiques et on illustrera seulement les composants ou modules fonctionnels, de l'architecturer technique et logiciel on va présenter les différentes parties techniques nécessaires permettant de réaliser les fonctionnalités décrites dans la section précédente [2.2]

2.3.1 Vue pour l'utilisateur

Le système offre à l'utilisateur deux choix pour l'exploiter :

1. Le système réalisé permet à l'utilisateur d'observer le fonctionnement et l'évolution de la micro mouse ie :véhicule autonome dans sans monde un labyrinthe de 8x8 [2.2.2] case dans un monde physique, pour but de résoudre ce labyrinthe et trouver le plus court chemin du point de départ au point final ;

   — **Le labyrinthe** est un carré de 1,44 m2 de surface composé de seize « cellules » carrées de 180 millimètres de côté et de 50 millimètres de haut. Il peut y avoir 8×8 cellules.

   — **Le robot** ne doit pas faire plus de 250 millimètres de large et de long (il n'y a pas de hauteur limite).
2. Le système permet aussi de simuler les fonctionnalités physiques dans un monde simulé, un labyrinthe de taille variante et des chemins à définir par l'utilisateur, aussi on peut charger un fichier de configuration pour créer le labyrinthe ;

   — **Le labyrinthe** peut être créé par l'utilisateur manuellement ou bien en passant un fichier de configuration.

   — **Le robot** Le robot est défini comme une entité qui se déplace dans le labyrinthe, il est sous la forme d'un cercle, surface du cercle ne dépasse pas la surface d'une case du labyrinthe.

Finalement, l'utilisateur peut observer en temps réel l'évolution de la micro mouse dans son univers, cette partie permet aussi d'afficher les valeurs des différents composants du véhicule comme les capteurs infrarouges et l'accéléromètre ainsi que d'autres informations sur la simulation.

2.3.2 Architecture technique

la micromouse sera composée de nombreuses parties electronique qui vont lui permettent de fonctionner.

Le plus important d'entre eux est le **PCB** (Printed Circuit Board) qui est le corps où tous les autres composants seront soudés, il sert aussi à les interconnecter et leur faire communiquer. Pour cela, nous avons trouvé un design open source [3] conçu pour les micro-souris qui contient

tous les modules nécessaires pour notre produit final.

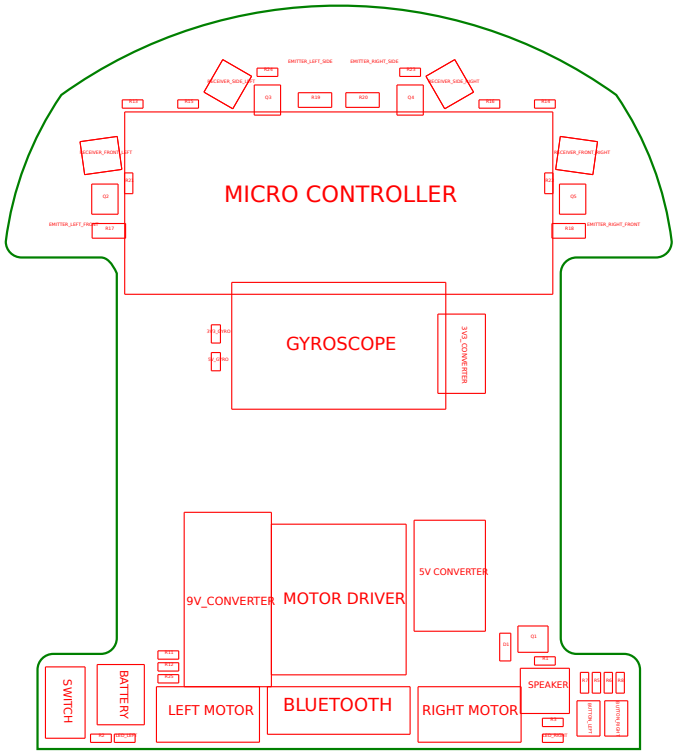


FIGURE 2.6 – digramme de composantes dans le PCB

Le cerveau de notre véhicule sera un micro contrôleur **STM32F103** [2] également connu sous le nom de Blue Pill, il a 64KB/128KB de mémoire flash et 20KB de RAM fonctionnant à 72MHz.

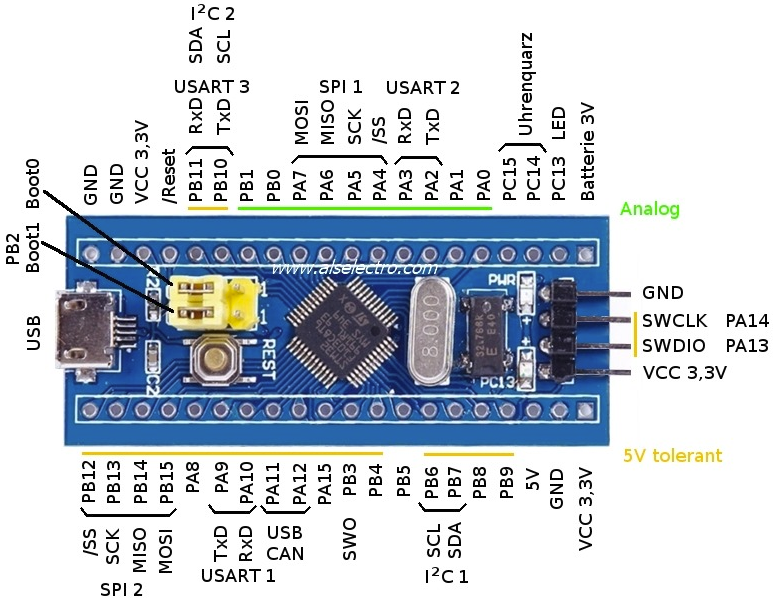


FIGURE 2.7 – Microcontrôleur, 32bit, STM32

On passe ensuite aux capteurs, qui sont les composants utilisés pour recevoir les informations du monde extérieur, qui est dans ce cas le labyrinthe.

Il y a trois types que nous allons utiliser, le premier est le **gyroscope** qui est employé pour détecter le changement de direction et l'angle de rotation, c'est important pour bien tracer le labyrinthe puisque le véhicule devra faire beaucoup de manœuvres dans le labyrinthe.

Ensuite, il y a **l'accéléromètre** qui, comme son nom l'indique, détecte l'accélération, et avec le gyroscope il peut trouver la position ou plutôt les changements de position, de micromouse, ce capteur alors sera le seul moyen de localisation pour notre véhicule.

Ces deux capteurs précédents sont généralement regroupés sur une seule circuit électronique parce qu'ils sont sovent utilisés ensemble et ont des fonctionnalités similaires, cela nous économise également de la place sur le PCB.

Enfin, il y a les **capteurs infrarouge** que nous utiliserons pour obtenir la distance du véhicule des murs et autres obstacles, ils seront placés sur toutes les directions pour évaluer plus précisément les alentours et avoir une meilleure cartographie du labyrinthe.

Nous disposerons également d'un **module bluetooth**, l'outil de communication principal. Lequel est utilisé pour envoyer des données en direct à l'interface afin de déboguer et de visualiser le processus de Mapping, ainsi que pour communiquer avec le simulateur de labyrinthe pour recevoir les signaux des capteurs artificiels.

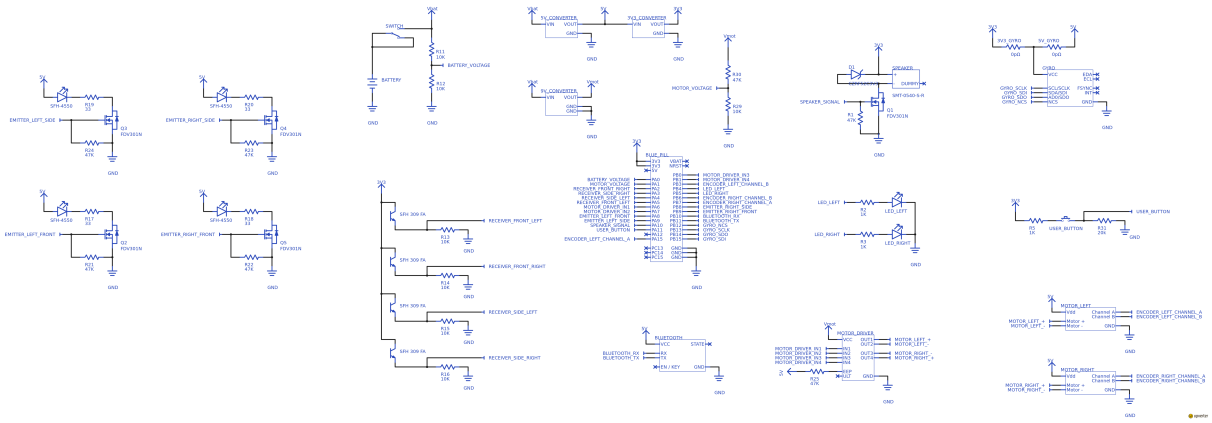


FIGURE 2.8 – Ensemble des composantes électroniques

Le contrôle du véhicule se fera avec deux **moteurs falhaber** [1] des deux côtés du véhicule. Les roues et leurs essieux sont imprimés en 3d pour s'adapter au design du circuit. Aucun volant ne sera utilisé à cause du poids supplémentaire qu'ils apportent et aussi parce que les deux que nous avons déjà peuvent faire des manœuvres très précises avec un code conducteur minimal.

Il existe d'autres petits composants électriques ou **SMD**s (Surface Mount Devices) tels que les résistances, condensateurs, interrupteurs, diodes, LEDs etc... qui sont utilisés pour contrôler le courant électrique sur le PCB.

2.3.3 Architecture logicielle

Contrôle du véhicule : à un moment de la simulation notre véhicule va décider de faire des virages, des marches arrière ...etc., des prises de décisions floues.

Pour gérer ça on utilise la logique floue. La logique floue (fuzzy logic, en anglais) est une logique polyvalente où les valeurs de vérité des variables - au lieu d'être vrai ou faux - sont des réels entre 0 et 1. En ce sens, elle étend la logique booléenne classique avec des valeurs de vérités partielles1. Elle consiste à tenir compte de divers facteurs numériques pour aboutir à une décision qu'on souhaite acceptable.

Mappage et navigation : Le véhicule a le droit de faire un repérage du labyrinthe pour déterminer le chemin vers le point d'arrivée, puis de retourner au point de départ pour réaliser l'épreuve en utilisant le chemin le plus court qu'elles ont déterminé.

Ce qui nous introduit au mappage et navigation du véhicule ie :La localisation et cartographie simultanées ; 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.

Calcul du plus court chemin : après avoir fait le mappage le robot se place sur la case départ, en le relançant le robot ne va pas refaire les mêmes pas il va calculer le plus court chemin entre la case de départ et la case d'arriver en utilisant toutes les données récupérer à l'étape de mappage et en utilisant un algorithme de calcul de plus court chemin.

En théorie des graphes, le problème de plus court chemin est le problème algorithmique qui consiste à trouver un chemin d'un sommet à un autre de façon que la somme des poids des arcs de ce chemin soit minimale.

Communication : Pour déboguer et visualiser les données du véhicule ou bien pour recevoir les signaux artificiels des capteurs de la simulation une communication entre les composants est nécessaire, ainsi le transfert de données se fait en temps réel.

2.4 Problématiques identifiées et solutions envisagées

D'un coup d'œil, plusieurs problèmes majeurs se posent d'un point de vue technique du projet. Dans cette section, nous décrivons ces problèmes et les solutions que nous y apportons.

2.4.1 Mappage et navigation

Pour notre véhicule, les seules données qu'il reçoit du monde extérieur sont celles des capteurs, ce sont des signaux en temps réel des changements de rotation et d'accélération, ainsi que les distances des murs qui sont les principales informations que nous allons utiliser pour le traçage de la mappe.

Pour transformer ces signaux en une carte 2D du labyrinthe, nous allons devoir commencer par déterminer la position et la direction du véhicule. Pour cela, nous utiliserons les informations du gyroscope afin de garder une copie de la position et de la direction et de les actualiser

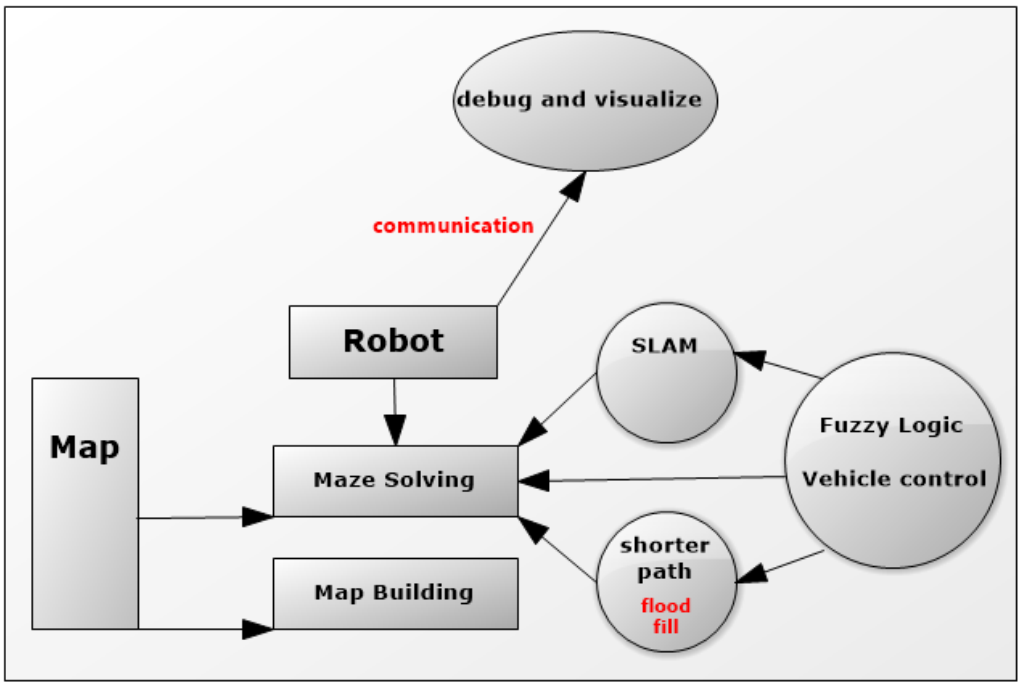


FIGURE 2.9 – schéma illustrant l'enchaînement des différentes parties

a chaque iteration en utilisant les données reçues.

Après avoir obtenu ces entrées, nous allons les utiliser avec les informations restantes des capteurs infrarouges pour faire la mappage proprement dite, nous pouvons utiliser un algorithme de Flood fill pour déterminer la géométrie des murs et/ou des obstacles. Après un peu de nettoyage et interpolation nous devrions avoir une carte finale du labyrinthe.

Une fois le tour de mapping est terminé, nous devrons trouver le chemin le plus court entre la position du véhicule et le point d'arrivée. Une solution qui vient à l'esprit est l'algorithme A* qu'il est souvent utilisé dans les labyrinthes et qui est alors idéale pour notre projet. Bien sûr, l'algorithme doit être modifié pour tenir compte des erreurs occasionnelles de contrôle qui surviennent à cause des irrégularités du monde physique.

2.4.2 Contrôle du véhicule

Notre véhicule se compose de deux moteurs que nous devrons contrôler pour parcourir les chemins avec fluidité, il devra faire des virages et éviter les collisions avec les murs et autres obstacles, tout en suivant le chemin créé pour lui.

Pour résoudre ces problèmes, nous allons utiliser la logique floue qui servira à générer des signaux pour les puissances des moteurs en temps réel. La façon dont cela fonctionne est que nous créons un moteur d'inférence et lui fournissons des règles qui décrivent la façon dont la véhicule doit se comporter en se déplaçant, et le moteur utilisera ces règles ainsi que les données du capteurs pour générer les signaux de sortie des moteurs qui seront sa décision finale.

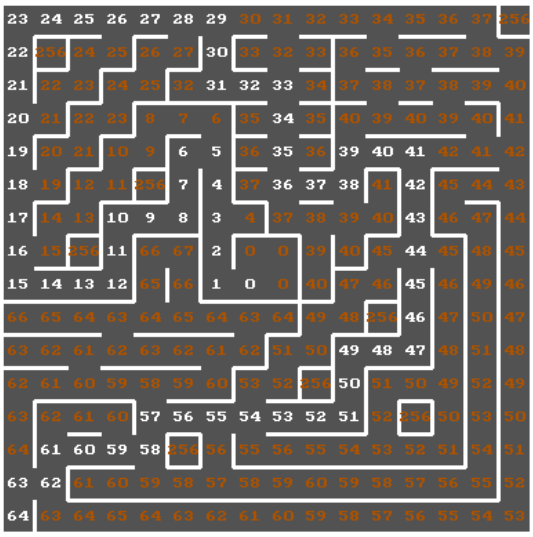


FIGURE 2.10 – Algorithme de Flood fill, Micromouse competition US 1982

2.4.3 Communication

Un problème qui se pose à cause des besoins de certaines fonctionnalités du projet est la communication, que ce soit pour déboguer et visualiser les données du véhicule ou bien pour recevoir les signaux artificiels des capteurs de la simulation.

Tous deux auront besoin de leur propre protocole de communication respectif à être efficaces, fiables et rapides puisque le transfert de données se fait en temps réel, avec les ressources limitées de micro contrôleur. et pour cela nous allons utiliser une communication serial entre le module Bluetooth de Micromouse et les autres parties communicantes.

2.5 Environnement de travail

Plusieurs outils et matériels ont été utilisés pour développer les différents composants de la micromouse. Ici ils sont décrits et leurs rôles dans la réalisation de chacune de ces parties.

2.5.1 logiciels et environnements de développement

Le simulateur de labyrinthe et l'interface temps réel de la micro mouse sont écrits en langage de programmation Processing. Le contrôleur principal de l'Arduino est écrit en C.

Processing est un langage qui se concentre sur les graphiques et les interfaces 2D et 3D, il sera utilisé pour créer des courbes et des animations en temps réel très claires et informatives, ainsi que des contrôles simples et faciles à utiliser. il fournit des bibliothèques pour la communication serial pour le transfert de signals depuis et vers le micro contrôleur.

Le **IDE**[1] **Processing** nous permettra d'organiser le projet, etd'exécuter le code de Processing. Il a également un débogueur intégré etd'autres outils de developpement, et une interface simple pour ajouter les bibliothèques et extensions de langage.

1. Integrated Development Environment

Box2D est une librairie de simulation physique 2D écrite pour C++, mais il existe des frameworks et wrappers pour cela en Java et Processing. elle sera utilisée pour simuler le véhicule et ses interactions avec son environnement. et pour générer en temps réel des signaux de capteurs artificiels pour tester la micro souris.

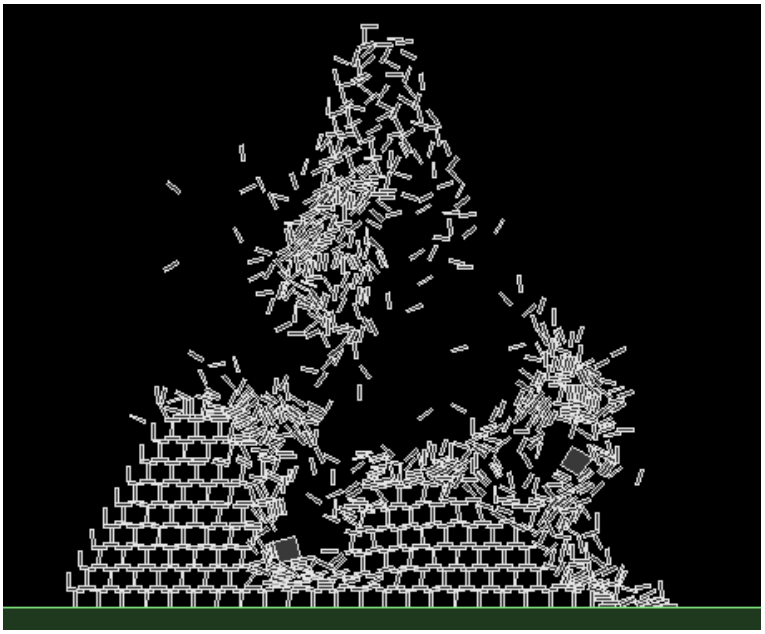


FIGURE 2.11 – Exemple d'usage de Box2D, Wikimedia creative commons

Le **IDE Arduino** sera utilisé pour développer le microcontrôleur principal en C, il dispose d'outils qui permettent la compilation et l'ectiture des sorties binaires dans le micro contrôleur.

2.5.2 Matériel informatique

Dans ce projet, nous avions une carte de circuit imprimé utilisée pour connecter les pièces de notre véhicule, celle qui devait être imprimée à l'aide d'une **imprimante PCB** qui grave les circuits de la carte pour pouvoir ensuite souder les pièces ensemble. Nous avons également des pièces qui nécessitent une **imprimante 3D** pour fabriquer les formes des roues et essieux du véhicule pour qu'ils s'adaptent au design du corps.

Chapitre 3

Partie technique construction du labyrinthe

Chapeau du chapitre

3.1 Analyse de la problématique de la construction du labyrinthe

Chapeau

3.2 État de l'art : études des solutions existantes

Chapeau

3.3 Solution proposée et sa mise en œuvre

Chapeau

3.4 Tests et certifications de la solution

Chapeau

Chapitre 4

Partie technique mappage et navigation

Chapeau du chapitre

4.1 Analyse de la problématique du mappage et navigation du véhicule

Chapeau

4.2 État de l'art : études des solutions existantes

Chapeau

4.3 Solution proposée et sa mise en œuvre

Chapeau

4.4 Tests et certifications de la solution

Chapeau

Chapitre 5

Partie technique calcul du plus court chemin

Chapeau du chapitre

5.1 Analyse de la problématique du plus court chemin

Chapeau

5.2 État de l'art : études des solutions existantes

Chapeau

5.3 Solution proposée et sa mise en œuvre

Chapeau

5.4 Tests et certifications de la solution

Chapeau

Chapitre 6

Partie technique contrôle du véhicule

Chapeau du chapitre

6.1 Analyse de la problématique du contrôle du véhicule

Chapeau

6.2 État de l'art : études des solutions existantes

Chapeau

6.3 Solution proposée et sa mise en œuvre

Chapeau

6.4 Tests et certifications de la solution

Chapeau

Chapitre 7

Rendu final

Chapeau du chapitre

7.1 Interface utilisateur finale

Chapeau

7.2 Tests utilisateur et certification

Chapeau

7.3 Autres tests et certifications

Chapeau

Chapitre 8

Gestion de projet

Chapeau du chapitre

8.1 Méthode de gestion

Chapeau

8.2 Répartition de tâches

Chapeau

8.3 Choix n°1

Chapeau

8.4 Choix n°2

Chapeau

Chapitre 9

Conclusion et perspectives

Chapeau du chapitre

Conclusion

Chapeau

Perspectives

Chapeau

Bibliographie

[1] Faulhaber motors specification. https://www.faulhaber.com/fileadmin/Import/Media/EN_1524_SR_DFF.pdf.

[2] Stm32 'blue pill' micro controller. https://web.archive.org/web/20190524151648/https://wiki.stm32duino.com/index.php?title=Blue_Pill.

[3] Bulebule. Micromouse board pcb. https://bulebule.readthedocs.io/en/latest/building.html#board, 2017.

Annexe

Chapeau