UNIVERSITÉ LAVAL

Design III

Remise 3

présenté à

Philippe Giguère, Dominique Grenier, Denis Laurendeau

par

Équipe 7 — Zière

matricule	*nom*	*signature*
111 114 478	Garvin, Sébastien	
111 040 128	Kedzierski, Xavier	
111 066 466	Magnan, Charles-Olivier	
111 071 384	Provencher, Jean-Michel	
111 073 630	Bourque, Emile	
111 075 478	Sylvain, Matthieu	
111 074 361	Brown, Jérémy	
907 196 009	Garneau, Laurent	

Université Laval

3 avril 2016

Historique des versions		
version	*date*	*description*
1.0	24 janvier 2016	Création du document
2.0	31 janvier 2016	Remise 1
3.0	28 février 2016	Remise 2
4.0	3 avril 2016	Remise 3

Table des matières

Chapitre 1

Diagrammes

1.1 Diagramme des fonctionnalités

1.2 Diagramme de classe

Le diagramme de classe à une taille trop importante pour être intégré au rapport, mais il se retrouve sous forme d'image dans le dossier de remise. Une bref description du flot d'éxécution de la routine suivra afin de mieu comprendre la fonction de chaque classe.

Du côté de la station de base, le main initialise l'interface. Les boutons permettent par la suite de commencer une routine en initialisant la classe StationBase avec la table de jeux et la routine voulu. Ce thread est le controleur de la station. Il commence par initialiser les autres threads :

1. StationServeur. Ce thread initialise une connection TCP avec le robot (avec TCPServeur). Il est en charge d'envoyer et de traiter tout la communication entre la station de base et le robot. Ceci est effectué par l'envoie de fichier JSON qui sont créé par la classe RequeteJSON.
2. FeedVideoStation. Ce thread a comme simple tache de lire les images de la caméra world.
3. AnalyseImageWorld. Ce thread ce sers des classes de Detection pour repérer les îles, trésors et le robot. Ceux-ci utilise les différentes intervalle de couleurs correspondante à chaque table dans IntervalleCouleur. Une fois trouvée, Les éléments cartographiques sont ajouté à la classe Carte.
4. ImageVirtuelle. Ce thread est en charge de dessiner sur l'image obtenue afin de l'afficher dans l'interface avec les informations voulu (trajectoire prévue, position robot, etc.). La classe représente donc tout simplement la carte virtuelle.

Suite à l'initialisation de StationBase, L'interface initialise un dernier thread (AfficherImageVirtuelle). Celui-ci a comme seul tâche de mettre à jour le feed vidéo (ImageVirtuelle) dans l'interface.

La classe StationBase dès le début de l'éxécution initialise la classe Trajectoire avec Grille-Cellule. Celle-ci représente la matrice de position que peut prendre le robot. Lors de la demande d'un trajet, Trajectoire appelle AlgorithmeTrajectoire avec la grille de cellule. Cette classe trouvera la trajectoire demandé.

Finalement, la classe RedirigeurTexte sers simplement à rediriger les print dans la boite de texte de l'interface.

Du coté du robot, le controleur est la classe Robot. La communication se déroule comme avec la station de base avec les classes (RequeteJSON, RobotClient, TCPClient). La classe UARTDriver est utilisé pour envoyer des commandes par UART. Tandis que la classe LectureUART est utilisé pour lire et traiter les données envoyé par le UART.

Lorsque le robot débute les phase d'alignement, il se sert de AnalyseImageEmbarquee et la Detection correspondante afin de trouver les déplacements requis.

La classe, RobotService est utilisée pour obtenir l'indice à l'aide de la lettre manchester.

Finalement, la classe ConfigPath est utilisée afin de transformer les path locale en path absolu.

Chapitre 2

Plan d'intégration

Chapitre 3

Vision

La section 3.1 et 3.2 se déroule évidamment sur la station de base. Il est a noter que lors de la réception d'une nouvelle image, celle-ci commence par la redimentionner de sorte à ne contenir que la table. Par la suite, un Gaussian blur est effectué. C'est à ce stade que les détections sont effectuée.

3.1 Localisation des iles et des trésors

Le processus de localisation des îles est effectué, au tout début de la routine. Pour localiser les îles, chacune des quatre couleurs est filtrée et placée dans un masque afin d'effectuer un traitement individuel. Ensuite, les formes ayant une aire trop grande ou trop petite sont éliminées. Par la suite, les formes possédant un trou (contour enfant) avec un aire concidérable sont éliminées (voir 3.2). Le système compare ensuite les contours restant des formes filtrées avec les formes géométriques en mémoire. De ce fait, celui-ci est en mesure d'identifier la forme qui possède le plus haut taux de compatibilité avec celles en mémoire. Si l'indice de précision est trop faible pour les quatres formes en mémoire, la forme est ignorée. Bien que ce processus soit déjà robuste, ceci est effectuée à dix reprises dans le but de conserver la liste d'île ayant la longueur la plus constante.

Pour ce qui est de la localisation des trésors, un processus similaire est utilisé. L'image est filtrée (à dix reprises) avec la couleur des trésors, puis les formes détectées sont retenues ou non dépendamment de l'aire de celles-ci. Les formes des trésors ne sont donc pas comparées, mais les trésors ne se retrouvant pas sur le long des murs sont négligés. Cette dernièr opération s'efectue facilement puisque l'image est redimentionnée de sorte à ne contenir que la table avant la détection.

3.2 Localisation du robot

Puisque la détection des îles se déroule avec succès, la même base est appliquée au robot. En effet, sur le dessus de celui-ci se retrouve deux formes (carré et cercle) de la même couleur

située sur le même axe. Par contre, afin de les différencier avec les îles, elles possèdent un trou blanc à l'intérieur (voir figure 3.1). L'image redimentionnée de la table est donc filtrée avec la couleur du robot et placée dans un masque. Ensuite (comme pour les îles), les formes ayant une aire trop grande ou trop petite sont éliminées. Par la suite, les formes ne possédant pas un trou (contour enfant) avec un aire concidérable sont éliminées. Le système compare ensuite les contours restant des formes filtrées avec les formes géométriques en mémoire. De ce fait, celui-ci est en mesure d'identifier la forme qui possède le plus haut taux de compatibilité avec celles en mémoire. Si l'indice de précision est trop faible pour les deux formes en mémoire, le robot ne sera pas détecté à cette itération.

FIGURE 3.1 – Image de la forme sur le robot

Une fois les contours détectés, un peut de géométrie permet de d'obtenir les données voulues. Les formes sont positionnées de sorte à ce que le centre des deux centres est situé au centre du robot. L'orientation du robot est simplement l'orientation du vecteur reliant le centre du cercle et du carré (sachant que le cercle est l'avant du robot).

Afin d'ajouter un peu de robustesse, suite à la première détection du robot, les détection suivante considère l'ancienne position du robot et procède à une analyse de fesabilité du déplacement. Si le déplacement est trop élevé, le robot n'est pas détecté. Dans l'éventualité ou le robot n'est pas détecté lors de dix itération consécutive, le robot est déclaré comme perdu et l'analyse de fesabilité de déplacement est ignorée jusqu'à ce qu'il soit retrouvé.

3.3 Phases d'alignement du robot

Tout d'abord, il est important de spécifier que la station de base gère les déplacements principaux du robot alors que le robot gère lui-même les phases d'alignements. Ceci dit, lorsque le robot est dirigé vers une position cible, la station de base lui indiquera quelle type d'alignement il doit effectuer. Il y a une phase d'alignement unique pour la capture d'un trésor, le dépôt de celui-ci sur l'île cible ainsi que pour l'amarrage du robot avec la station de recharge.

Les phases d'alignement regroupent plusieurs petites étapes. Tout commence avec le changement de l'orientation de la caméra embarquée. Celle-ci possède plusieurs positions prédéfinies, ce qui facilite grandement le déroulement du processus d'alignement.

3.3.1 Capture du trésor

Pour la capture du trésor, la caméra est placée en "position trésor" afin d'identifier le trésor et évaluer la distance le séparant du robot. Encore une fois, connaissant la position du préhenseur sur le robot, le robot calcule les ajustements nécessaires et les commandes appropriées sont envoyés. Une fois la commande de déplacement vertical effectuée, l'électroaimant sera activé et le robot reculera afin de valider que le trésor a bel et bien été capturé. La phase de préhension de trésor se termine donc et la station de base reprendra le contrôle du robot. **AJOUTER IMAGE**

3.3.2 Dépôt du trésor

Pour le dépôt du trésor, la caméra se positionnera face à la surface de jeu et analysera la position de l'île cible pour calculer le déplacement vertical et horizontal à effectuer. Une fois ces ajustements calculés, des commandes de déplacements sont envoyés. Finalement, une fois l'alignement terminé, le système valide la position de l'île cible par rapport à la zone de dépôt sécuritaire déterminée par des tests. Le trésor est soit déposé ou un autre alignement est effectué. **AJOUTER IMAGE**

3.3.3 Recharge du condensateur

La phase d'alignement avec la station de recharge est une étape cruciale dans le déroulement de la routine. En effet, un mauvais alignement aura comme conséquence un plus long temps de recharge ou aucune recharge dans le pire des cas. Il est donc indispensable que le robot soit parfaitement aligné avec la station de recharge afin que la recharge à induction soit optimale, mais aussi afin que le code Manchester soit transmis avec succès par le système LED à LED. **A COMPLETER**

Chapitre 4

Schémas Électroniques

4.1 Station de recharge

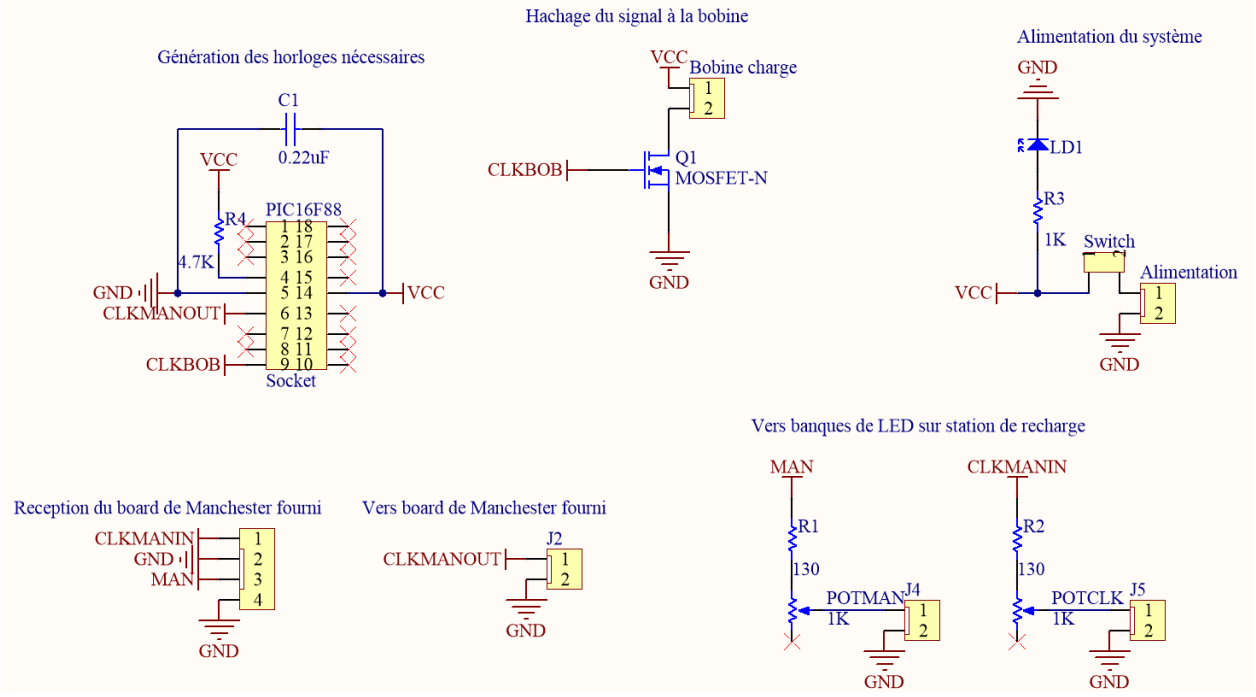

FIGURE 4.1 – Schéma électrique de la station de recharge

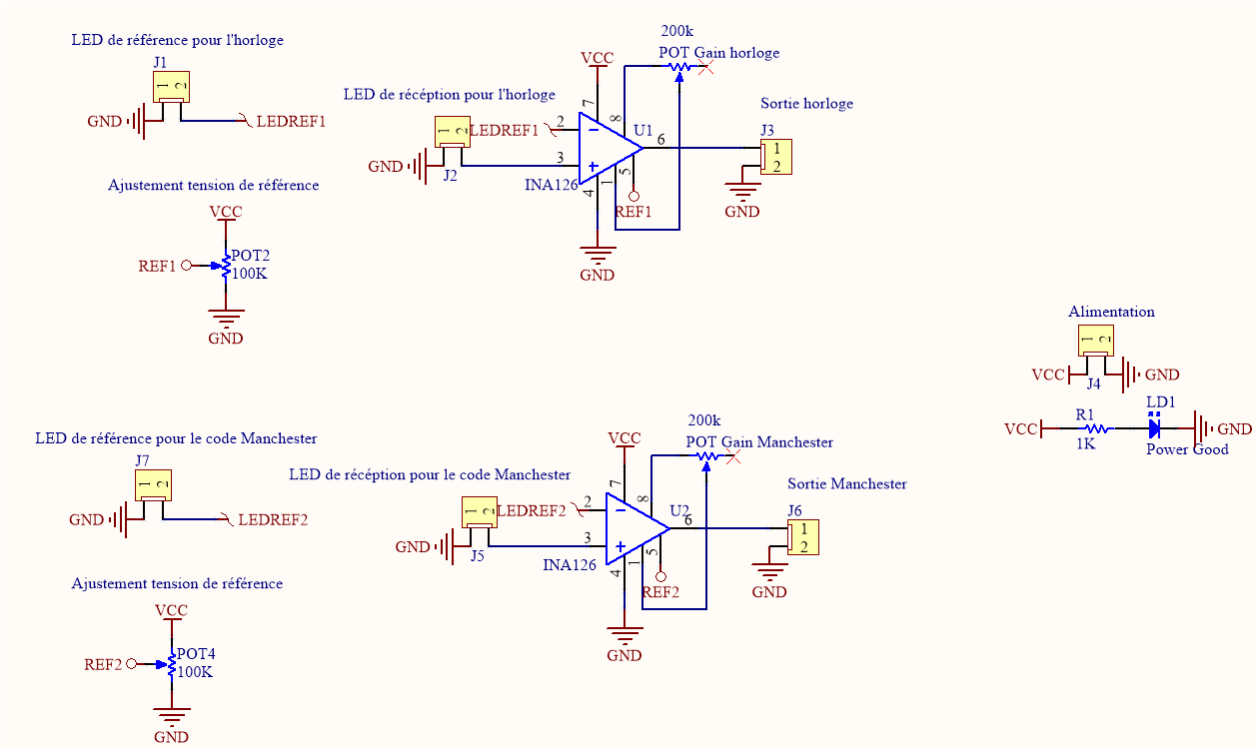

FIGURE 4.2 – Schéma électrique du circuit d'amplification du code Manchester

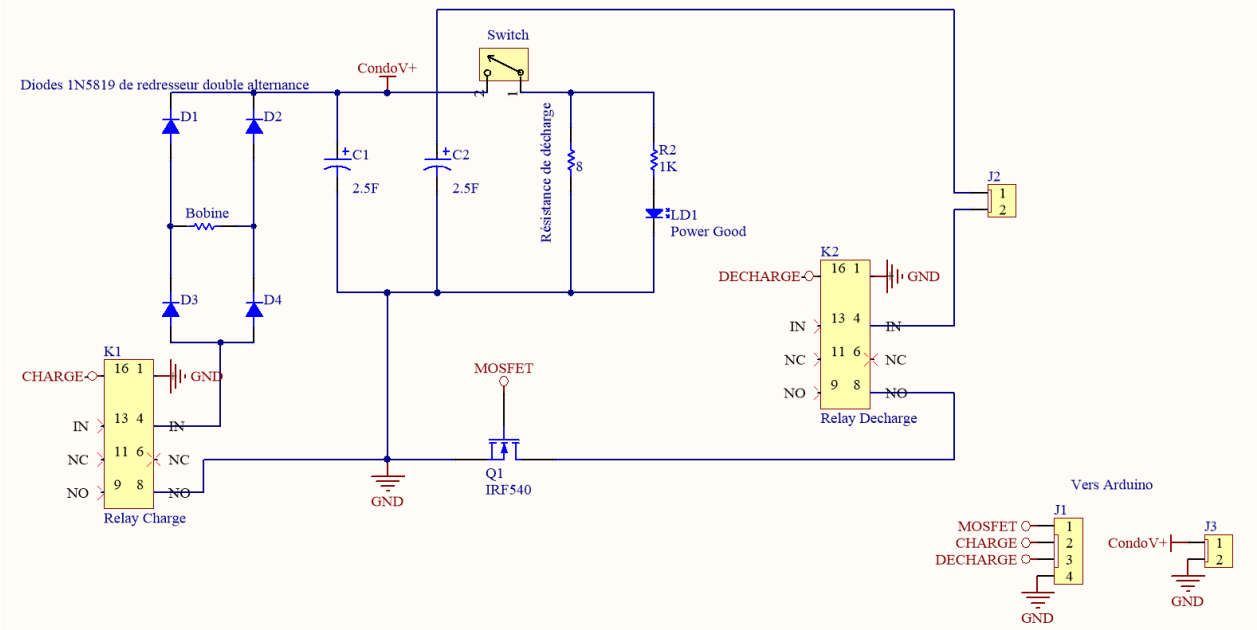

FIGURE 4.3 – Schéma électrique de la charge et la décharge du condensateur

4.2 Circuit d'amplification du code Manchester

4.3 Circuit de charge et de décharge du condensateur

4.4 Circuit de connections pour les moteurs

4.4.1 Connections moteurs et pont en H

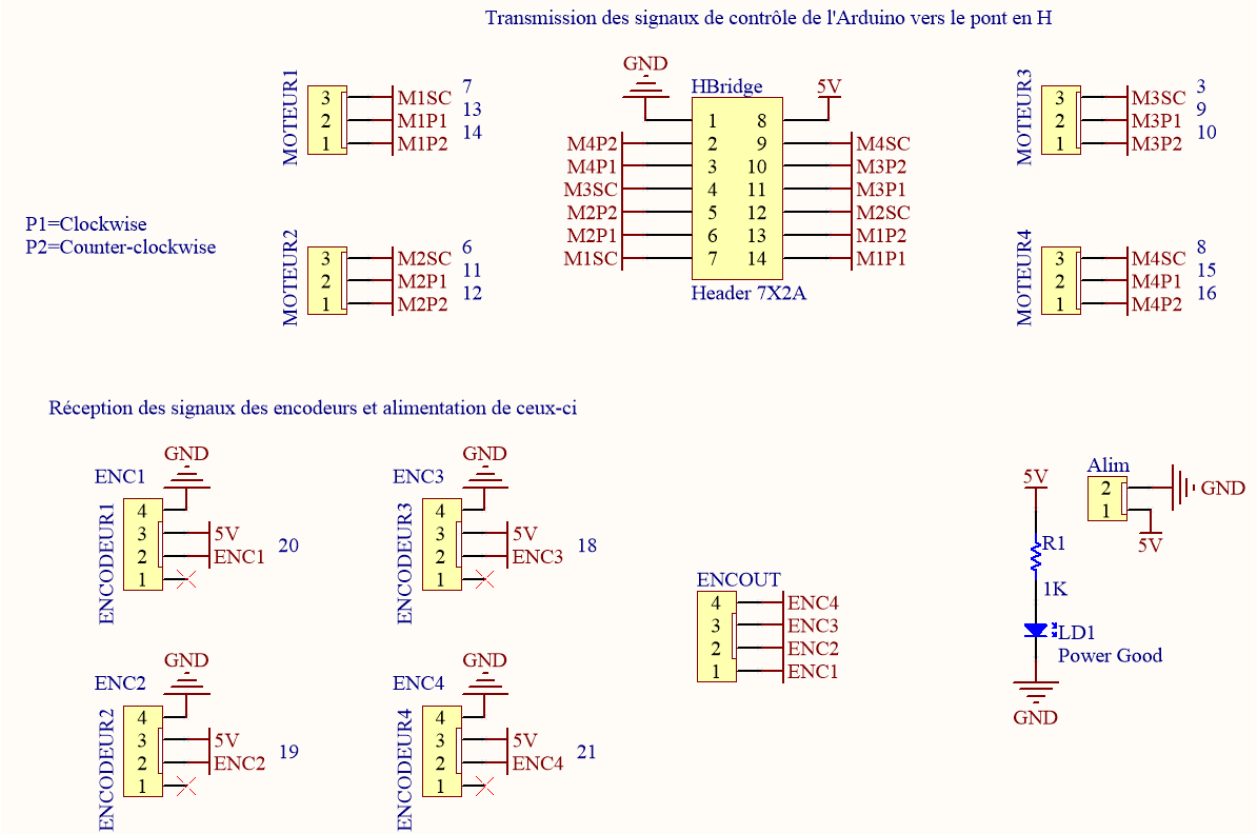

FIGURE 4.4 – Schéma électrique du circuit de connections des moteurs

4.4.2 Connections sur l'Arduino

4.5 Circuit d'alimentation

Bibliographie

[1] ETH Zurich, *An LED-to-LED Visible Light Communication System with Software-Based Synchronization*, [En ligne], http://people.inf.ethz.ch/schmist/papers/OWC12Slides.pdf, Page consultée le 27 janvier 2016

[2] Robotshop, *Arduino Mega 2560 Microcontroller*, [En ligne], http://www.robotshop.com/ca/en/arduino-mega-2560-microcontroller-rev3.html, Page consultÃ©e le 30 janvier 2016

[3] HobbyKing.com, *Turnigy 4500mAh 6S 30C Lipo Pack*, [En ligne], http://www.hobbyking.com/hobbyking/store/__10284__Turnigy_4500mAh_6S_30C_Lipo_Pack.html, Page consultée le 30 janvier 2016

[4] ebay.ca, *DC-DC Step-down Buck Converter Adjustable Power 4V-38V to 1.25V-36V 5A Voltmeter*, [En ligne], http://www.ebay.ca/itm/171445007919?_trksid=p2057872.m2749.l2649&ssPageName=STRK%3AMEBIDX%3AIT, Page consulté le 30 janvier 2016

[5] Robotshop.com, *Grove Electromagnet*, [En ligne], http://www.robotshop.com/ca/en/grove-electromagnet.html, Page consultée le 30 janvier 2016