Rapport de projet d'IMA 4

Brique d'apprentissage informatique

Remerciements

Nous voudrions tout d'abord remercier Monsieur Boé, Madame Pichonat ainsi que Monsieur Vantroys, nos tuteurs pour ce projet, pour leur aide et leur conseils pour la réalisation de notre projet.
Nous remercions Monsieur Redon, qui même sans avoir été notre tuteur, nous a fourni un soutien inestimable, tout au long de l'année pour ce projet, et sans qui rien de tout cela n'aurait été possible.
Nous remercions également Monsieur Flamand, grâce à qui nous avons pu obtenir nos PCB, et de nombreux conseils très utiles.
Nous remercions nos camarades Ganix et Justine, pour le matériel qu'ils nous ont généreusement prêté pour la réalisation de notre projet, ainsi que tous nos camarades pour leur soutien et leur bonne humeur tout au long de semestre.

Sommaire

Introduction

Lors de ce huitième semestre d'études en IMA, nous avons eu l'occasion de réaliser un projet pluridisciplinaire, pendant quatre mois, nous permettant de mettre en pratique nos apprentissages en informatique, électronique et également en gestion de projet. En effet, outre l'importance majeure des compétences techniques, c'est nous qui étions en charge de tout finir dans les temps et de respecter les délais imposés, de remplir notre wiki pour documenter notre avancement tout au long de l'année, de rédiger ce rapport, et de présenter un prototype fonctionnel. L'organisation aura donc également été une composante très importante. Et c'est de tout cela que nous allons vous parler dans ce rapport.

1. Analyse du sujet

Notre sujet est “Réaliser une brique d’apprentissage informatique”.

Il nous appartenait ensuite d’exploiter ce sujet, éventuellement d’en modifier certaines parties avec l’accord de nos tuteurs, et de finalement réaliser ce qui nous était demandé.

Cette analyse du sujet s’est faite tout au long de nos quatres mois de projet, et ce que nous vous présentons ici n’est que la conclusion de ces quatres mois.

1.1. Description et analyse de notre projet

En résumé, notre projet consiste à réaliser des briques, qu’un enfant pourrait facilement assembler entre elles, et qui formeraient un chemin. Ce chemin devrait ensuite être envoyé à un robot, et celui-ci le suivrait. C’est sur cette base que nous avons déterminé notre cahier des charges. Le point important était notamment la façon dont le robot allait suivre le chemin établi par les briques. Les deux possibilités étaient, soit que le robot roule sur les briques, soit qu’il roule à côté de celles-ci, par exemple sur un tapis de jeu. Après en avoir discuté avec nos tuteurs, nous nous sommes orientés vers l’option ‘tapis de jeu’ pour la simple et bonne raison qu’une taille réduite des briques nous semblait plus pertinent. En effet, si le robot roulait sur les briques, cela voudrait dire que celles-ci auraient une taille assez conséquente, ce qui n’est pas pratique à transporter ou ranger pour les parents. De plus, plus les briques sont larges plus l’espace de jeu de l’enfant devra être important si il veut pouvoir faire des longs chemin, hors toutes les familles ne disposent pas d’une aire de jeux grande réservée aux petits.

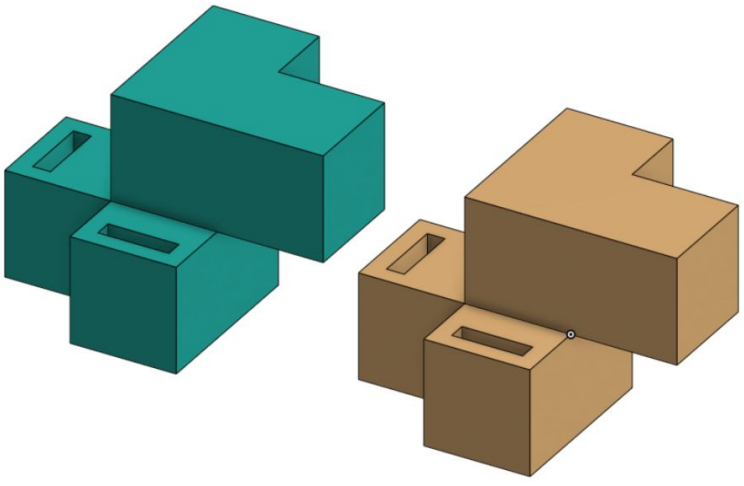
Il a ensuite fallu établir la façon dont les briques communiquaient entre elles. Nous avons choisit d’utiliser des liaisons série, quatre par brique afin de pouvoir établir un chemin dans n’importe quelle direction voulue. Le choix de la liaison série se justifie principalement par sa simplicité d’utilisation, et la possibilité d’établir quatre liaisons série en utilisant la bibliothèque software serial d’Arduino IDE.

Le principe de fonctionnement est alors qu’une brique est nommée brique principale et se charge d’alimenter toutes les autres briques, d’envoyer un message aux autres briques pour recevoir le chemin formé, et d’envoyer le

chemin formé au robot. Le programme des autres briques est ensuite relativement simple. Le premier message qu'elles reçoivent indique la direction de la brique maître. Ce message doit être retransmis sur toutes les autres liaisons si elles sont connectées. Tout message subséquent ne peut qu'être une réponse des autres briques, et doit donc être renvoyé vers la brique maître.

Nous avons ensuite dû décider quel microcontrôleur nous allions utiliser lors de la réalisation des briques. Ayant besoin de quatre liaisons séries par microcontrôleur, chacun nécessitant deux pins pour la liaison et deux pins pour le handshake, le nombre de pin requis a rapidement escaladé. Au total il nous fallait 16 pins pour les liaisons série, deux pour l'alimentation (VCC-Gnd) et quelques uns pour éventuellement ajouter des boutons, des afficheurs et des potentiomètres. Nous avons alors décidé d'utiliser des Atmega328p, qui sont les microcontrôleurs présents sur les Arduinos Uno. Ils disposent d'un nombre suffisant de pins, et il est très simple de réaliser des prototypes grâce aux Arduinos Uno. Lors de la phase de test, il est tout à fait possible d'utiliser un Arduino Uno en tant que brique esclave afin de vérifier que tout fonctionne correctement. Une fois le programme réalisé et fonctionnel, il est ensuite simple de programmer les 328p directement via Arduino IDE.

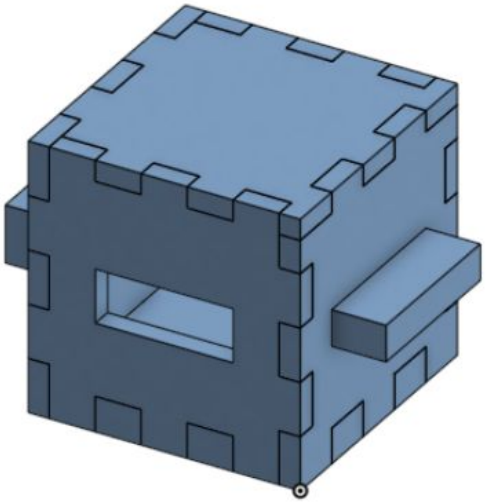
En parallèle, nous avons imaginé à quoi allaient ressembler nos boîtes, l'aspect que nous voulions leur donner et surtout comment elles allaient s'emboîter entre elles. Tout d'abord, nous nous sommes dit que la manière la plus simple qu'un enfant aurait de connecter deux boîtes entre elles serait qu'elles s'imbriquent l'une dans l'autre verticalement comme le montre l'image ci-dessous.



Version 3D de la première version des boîtes

Dans notre première version, le design était assez complexe, les fentes dans les parties basses contenait les pins reliés aux PCB et en emboîtant les briques entre elles, une connection devait se faire entre la partie haute d'une brique et la partie basse d'une autre ce qui les reliait. Bien que nous nous sommes longtemps focalisés sur ce concept au début, nous avons jugé qu'il n'était peut être pas adapté à notre clientèle. En effet, un enfant en bas âge ne comprendrait peut être pas intuitivement que ces briques se lient pour former un chemin alors que l'aspect des briques était déjà lui même compliqué. Nous avions d'ailleurs réalisé des prototypes en papiers qui nous permettaient de tester notre design sans utiliser de bois. C'est grâce à ces prototypes que nous avons eu l'idée de largement simplifier l'apparence des boîtes.

Finalement, nous avons choisis de placer les fentes sur les côtés de la brique, et cette fois, elle ne devra pas s'emboîter verticalement mais horizontalement. Ainsi une des formes géométriques les plus simples faisait l'affaire, un cube ! Un enfant de trois ans pourrait grâce à ce design jouer plus instinctivement avec nos boîtes au vu de leur simplicité.



Version 3D des boîtes finales

Mais ce n'est pas la seule raison pour laquelle nous avons opté pour ce choix. En effet, tout comme la forme des boîtes premières version était compliquée, la façon dont les connecteurs étaient assemblés pour être ensuite liés était très alambiquée.

Notre première approche consistait à utiliser des picots montés sur des ressorts pour le contact. Chaque brique aurait deux côtés mâles et deux cotés femelles, et les picots pourraient s'insérer dans les côtés femelles puis être pressés contre la piste par le ressort pour réaliser le contact. Le ressort permet de plus d'éviter qu'un enfant se blesse, si il pousse sur le picot, celui-ci va

simplement se rétracter grâce au ressort. Cependant, cette solution était extrêmement complexe à mettre en oeuvre du point de vue technique, et pas vraiment optimale, nous avons donc testé d'autres choses.

Une deuxième option dont nous disposions a été l'utilisation de lamelles pour assurer le contact, mais elle aurait pris trop de place et aurait de même été compliquée à mettre en oeuvre. De plus, il aurait alors été possible pour l'enfant de toucher simultanément les deux pôles de l'alimentation, et ainsi subir une légère décharge par les piles 9V, ce qui n'est pas très grave, mais peut devenir contrariant si il réalise le contact avec la langue.

Nous avons donc au final décidé d'utiliser de simples pins mâles et femelles qui s'emboîtent horizontalement et nous permettent donc d'avoir des boites de la forme de simples cubes. Ensuite s'est présenté le problème de ne pas blesser les enfants avec les pins pointus. Pour cela, nous avons décidé, grâce au conseil de monsieur Redon, d'utiliser les pins mâles sur les côtés femelle des boîtes et vice-versa. De cette façon, l'enfant ne peut pas se blesser sur les picots pointus des côtés mâles, ni même réaliser de faux contacts en touchant un picot par erreur. Il y aura 8 pins sur chaque côté de la boîte, ce qui est un nombre assez important mais nécessaire, comme détaillé plus haut.

1.2. Analyse de la concurrence

Une part importante de l'analyse du sujet est également l'analyse de la concurrence. En effet, ce sujet à déjà été exploité par d'autres entreprises, et à donné des jouets dont nous allons essayer de nous différencier. Nous en avons analysé deux en particulier, Cubetto et Code&Go Mouse.

Cubetto est probablement le jouet existant ressemblant le plus à notre projet. En effet, il implique également de diriger un robot en formant un chemin de briques. Le robot ne va pas rouler sur les briques, tout comme le nôtre. Nous avons cependant certains avantages notable par rapport à ce jouet.

Jeu pour enfant Cubetto

Contrairement à Cubetto, dont les briques sont disposées sur un plateau, donc pour des chemins d'une taille maximale limitée, notre chemin peut comporter en théorie un nombre infini de briques. Pour agrandir le chemin, il suffit d'ajouter des briques au bout de celui-ci. Il est également très simple pour nous d'insérer une brique en plein milieu de chemin, tandis que pour Cubetto, il faut décaler toutes les briques se trouvant derrière la brique à déplacer.

Un autre avantage sera en théorie le prix. Cubetto est un très beau jouet, mais également relativement cher (264,96€). Notre projet à été réalisé en utilisant des Atmega 328p, dont le prix est négligeable, des PCB réalisés grâce à l'école, ainsi que quelques résistances, quartz, capacités et headers mâle/femelle. La brique en elle même est réalisée grâce à la découpeuse laser du FabLab. Au total, le coût individuel d'une brique est donc très faible. Les seuls éléments relativement chers comparés au reste sont la brique maître et le robot, car nous avons pris la décision pour l'instant de conserver des Arduinos sur ces deux éléments. Ils nécessitent de plus tous deux un module Bluetooth et une alimentation. Mais même en combinant tous ces coûts, notre projet reste bien moins cher que le prix d'achat du Cubetto.

Enfin, notre projet inclus les commandes while et if d'un langage de programmation, ce qui n'est pas le cas du Cubetto à l’exception de leur bloc fonction qui permet de créer une fonction annexe telle qu’on en appellerait dans un main. Nous sommes donc en théorie plus flexibles, moins chers et plus complets que le Cubetto.

D’un autre côté, nous avons le jeu Code & go mouse. Il s’agit d’un robot en forme de souris avec des flèches directionnelles sur le dessus. La souris se trouvant au bord d’un labyrinthe, l’idée est alors de l’aider à parcourir le chemin pour qu’elle retrouve son fromage.

Jeu pour enfant Code&Go Mouse

Cette fois, le chemin que va réaliser le robot est établi en appuyant sur les différents boutons en forme de flèches au fur et à mesure. Contrairement à nous, le chemin n'est pas physique, visuel. Il faut compter le nombre de fois qu'on appuie sur chaque bouton et faire attention à ne pas se tromper d'ordre sinon le chemin que la souris va parcourir. De plus ici les choix de déplacement sont très limités : avant, arrière, gauche et droite; il n'y a ni while ni if.
Notre approche est plus accès programmation en tant que telle que notre concurrent.
Enfin, l'intérêt de ce jeu est de pouvoir bouger les murs du labyrinthe mais cela s'arrête là alors que pour notre projet, nous pourrions faire une multitude de tapis de jeu qui raconteraient des histoires différentes, ajouter de nouveaux cubes avec de nouveaux "pouvoirs" etc… Notre sujet compte donc bien plus de possibilités ludiques et éducatives que Code&go.

2. Réalisation du prototype

Dès le début de notre projet, nous avons divisé le travail en deux grandes parties distinctes. Une partie programmation, qui inclut la programmation des différentes briques esclaves, de la brique maître ainsi que la programmation du robot. La deuxième partie inclut la réalisation des briques elles-mêmes. Il faut donc pour cette partie réaliser les PCB, les fixer dans les boîtes, et réaliser les connections entre les différentes boîtes. Le robot n'étant pas le centre de notre projet, il a été monté en utilisant un kit simple. Le pcb du robot n'était également pas une priorité.

L'objectif de départ était de prototyper un programme fonctionnel. Nous avons tous les deux travaillé sur ce problème en utilisant deux techniques différentes, Simon en vérifiant les différents ports série par scrutation, et Maëva en utilisant les interruptions. C'est le programme par scrutation qui a fonctionné en premier, et donc pendant que Simon continuait à travailler sur la partie informatique, Maëva s'est chargée de la réalisation des briques. Malgré cette division du travail, nous nous sommes bien évidemment entraidés lors de nos différentes tâches.

2.1. Partie programmation

Malgré la relative simplicité du programme que l'on avait établie auparavant, il s'est présenté quelques difficultés pour cette partie. D'abord, les liaisons série soft que l'on a établies ne peuvent fonctionner qu'une seule à la fois. Il est donc nécessaire d'utiliser un processus de 'handshake' avant de démarrer la communication via la liaison série.

Nous avons utilisé deux pins par liaison série, un pour l'émission et un pour la réception. Quand l'arduino détecte qu'un de ses pins de réception passe à un, il sait qu'une brique adjacente désire communiquer avec lui. Il va alors envoyer un sur le pin d'émission, ce qui correspond à un accusé de réception et indique à la brique émettrice que la transmission peut commencer, puis va lire la liaison série correspondante. Si il est déjà occupé à lire une autre liaison série, il va simplement attendre la fin de cette communication avant d'envoyer l'ACK à la brique suivante. De cette façon, il est possible de contourner les problèmes liés à l'utilisation de multiples liaisons séries soft, au prix d'un programme un peu plus complexe et d'une sur-utilisation des pins du microprocesseur, puisque chaque liaison va utiliser 4 plutôt que 2 pins.

Un autre problème avec ce processus est lié aux choix faits lors de l'analyse du projet. Le chemin de briques formé est à priori un inconnu. Une brique esclave, ne peut donc pas savoir lesquels de ses ports série sont connectés et lesquels ne sont branchés sur rien. Si on combine ça au fait que le processus de scrutation implique qu'il puisse y avoir un délai avant qu'une brique émettrice reçoive un ACK, il devient nécessaire de trouver une façon fiable de détecter les ports branchés et les ports non branchés. Nous avons pour cela décidé d'utiliser des timeouts. Lors d'une tentative de communication, la brique émettrice va éventuellement conclure que rien n'est branché sur le port après une durée déterminée. On résout ainsi le problème de façon assez simple.

Le programme d'une brique esclave peut être divisé en deux parties, émission et réception. Son rôle est d'attendre une réception, puis en fonction des événements précédents et de son type, de retransmettre le message sur un ou plusieurs de ses ports séries. La partie réception va donc lire en permanence les quatres pins de réception des quatres liaisons série, puis va initier la communication lorsqu'un message est reçu. La partie émission entre en jeu quand la brique à fini de recevoir un message. Il y a alors deux cas possibles.

Soit c’est le premier message reçu par la brique, elle va alors déterminer que la brique maître se trouve dans la direction du port de réception, puis tenter de retransmettre le message sur les trois autres ports série en y ajoutant sa propre ID, avec un timeout afin de déterminer si une brique y est connectée ou non.

Tout message reçu par la suite sera simplement retransmis dans la direction de la brique maître, et normalement, devrait uniquement être reçu depuis une liaison autre que celle de la brique maître.

Le programme de la brique maître s’exécute en deux étapes. D’abord, il va envoyer un message à la première brique connectée à sa liaison série. Ceci va déclencher le début du mapping du chemin. Une fois le message propagé jusqu’au bout du chemin, il sera renvoyé à la brique maître, qui connaîtra alors les différentes successions de chemins possibles. Parmi celles-ci, ne seront retenues que les succession de chemins menant à une brique de fin. Après un temps défini, la brique maître va supposer que tous les chemins possibles ont été parcourus par le signal. Elle va alors envoyer une chaîne composée de tous les chemins possibles, mis bout à bout, au robot, via le module Bluetooth.

Le programme du robot ensuite, consiste à attendre la réception d’une chaîne sur la liaison série reliée au bluetooth. Il va alors parcourir cette chaîne caractère par caractère en effectuant les opérations correspondantes. Les instructions tout droite, gauche, droite et fin sont très simple, et consistent juste à activer les moteurs du robot d’une certaine manière. En revanche, les instructions while et if requièrent un traitement un peu plus complexe.

Le while va exécuter en boucle un certain nombre d'instructions. Le nombre de boucles effectuées est défini par le deuxième octet décrivant l'instruction while. En pratique, lorsque le programme voit une instruction while, il va exécuter toutes les instructions subséquentes jusqu'à rencontrer un fin de while. Il va alors incrémenter un compteur et retourner au while correspondant à ce fin de while. Une fois le compteur également au nombre de boucles à effectuer, il ne va pas retourner au while mais simplement continuer le programme après le fin de while.

Le if, la seule instruction à varier en fonction de l'environnement du robot, va utiliser le capteur ultrason. Lorsque le programme voit une instruction if, il va lire la valeur renvoyée par le capteur ultrason, puis la comparer au deuxième octet porté par l'instruction if. En fonction du résultat de la comparaison, le programme va soit continuer sur la chaîne de caractère actuelle, et donc choisir une direction du if, soit sauter jusqu'au if présent dans une autre chaîne de caractère et ayant la même ID que le if lu. Dans ce cas, l'autre direction aura été choisie.

Sans entrer dans plus de détails, notamment pour décrire tous les problèmes rencontrés lors de la réalisation des différents programmes, de la configuration des modules bluetooth et de la programmation des 328p, ceci conclut la description du fonctionnement de la partie programmation du projet.

2.2. Partie briques

Une fois une première idée de l'aspect des boîtes, de leur contenu et de comment assembler les différentes parties établi, nous pouvions débuter la réalisation d'un prototype. Bien sûr cette première idée, sera modifiée au fil du temps par les conclusions et les observations faites durant le projet.

La construction des boîtes peut être divisée en deux sous parties, la confection des PCB donc l'aspect plus électronique et celle des boîtes en elle même un aspect un peu plus mécanique dirons nous.

Nous avons choisi de commencer par les circuits imprimés car la réalisation de ceux-ci est longue, il faut les designer, puis les graver, les souder et enfin les tester et là on apporte des améliorations et c'est retour à la case départ. De plus, la taille des boîtes dépendra de la taille des cartes alors nous ne pouvions décemment pas nous lancer dans la fabrication de boîtes sans réelles mesures.

En ce qui concerne, le prototypage des PCB, la difficulté ne résidait pas dans le nombre d'éléments à placer puisqu'ils étaient en nombre restreint mais dans les croisement des pistes de routage.
Pour notre projet, nous utilisons un Atmega328p, à partir de celui-ci nous réalisons 4 liaisons séries appelées soft car nous les faisons nous même. Nous avons décidé que pour une liaison série, nous nécessitions 7 pins, le VCC et le

ground pour alimenter les briques que nous traversons puisqu'elles ne sont pas toutes équipées de piles, RX et TX pour la réception et la transmission des données, In et Out qui feront un simulacre de handshake et établir la communication, et le Reset pour réinitialiser toutes les briques depuis la brique principale sans avoir à couper l'alimentation. 7 pins nous suffisaient donc mais les barrettes femelles venant avec un nombre pair de pins, nous avons opté pour des header de taille 8.

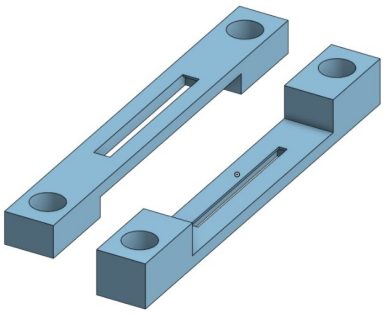
Pour emboîter les briques entre elles, nous avons deux barrettes femelles et deux barrettes mâles en face de fentes qui feront office de pont entre les PCB. Lors des essais avec notre prototype en papier, nous avons remarqué que le sens des pins des barrettes mâles devait être inversé par rapport à celui des femelles. Ainsi lorsque deux boîtes seront en face, les barrettes seront le miroir de l'autre et les bons pins s'enclencheront ensemble.

Nous avons réalisé plusieurs PCB différents, tant bien sur le logiciel que physiquement. Nous avons changé l'ordre des pins attribués sur l'Atmega328p pour optimiser l'espace utilisé sur la carte, la taille de celle-ci et réduire au maximum le nombre de fils qui se croisent. Malheureusement, au minimum, un fil croisait toujours les autres, nous avons donc mis des vias pour les relier grâce à un jumper. Une fois les PCB réalisés et soudés, il était temps de passer à la fabrication des boîtes.

Nous avons rencontrés de nombreux petits soucis lors de la conception des briques, erreurs de notre part et soucis techniques qui poussent à nous améliorer pour la suite.
Nous avions décidé assez rapidement que les briques allaient être en bois puisque c'est un matériau que nous aimons et qui est simple d'utilisation pour réaliser des cubes simples. De plus, cela nous a permis par la suite de les peindre pour leur donner un côté plus attractif pour les enfants.
Nous avons donc pris des planches de bois que nous avons découpé grâce à la découpeuse laser du fabricarium, c'était, nous le pensons, la manière la plus simple de réaliser les cubes.

Une fois les boîtes découpées, nous nous sommes alors préoccupé de maintenir les barrettes en place.
Pour ce qui est de la barrette femelle, le plus simple a été de le coller grâce à la superglue, colle très résistante et pratique d'utilisation.
Par contre les barrettes mâles devaient être maintenues à une certaine hauteur de la boîte sans pour autant sortir de celle-ci sans quoi les enfants risqueraient de se blesser.
La solution évidente était donc la conception d'une pièce 3D qui serait vissée à une des faces et qui tiendrait la barrette en face de la fente servant de connexion inter boîtes.

Nous avons alors réalisé la pièce ci dessous qui maintient la barrette mâle comme suit :



pièce barrette mâle sur Onshape

pièce barrette mâle assemblée

Nous pensions que cette pièce convenait à notre utilisation alors nous avons procédé au soudage des boîtes dans leur totalité, tous les pins des barrettes reliés aux PCB. Malheureusement, après quelques tests, nous nous sommes aperçu que la pièce était quelques millimètres trop loin de la fente et que les barrettes ne s'emboîtaient pas réellement. Par manque de temps nous avons décidé de passer outre et de ne pas réimprimer les pièces en changeant les dimensions, nous avons préféré ne pas fixer la barrette pour le jour de la présentation.

Je dirais quand même que nous avions largement sous estimé le temps de confection et de soudure des boîtes.

3. Améliorations et limitations

Dans cette partie, nous allons décrire les problèmes majeurs rencontrés lors de la réalisation de ce projet, la façon dont nous les avons résolus, ainsi que les pistes d'amélioration possibles. En effet, le projet s'est relativement bien passé, et nous sommes venus à bout de nos problèmes mais il s'est tout de même présenté de nombreuses situations qui nous ont coûtés plus ou moins longtemps.

3.1. Erreurs faites et résolues

Une première erreur à été de penser que les pins de l'arduino non branchés pourraient être dans un état constant. Ce n'est bien évidemment pas le cas, et lors de nos premiers branchements, le programme de la brique esclave n'a pas vraiment fonctionné de ce fait. Les pins sont dans un état indéterminé, et fluctue entre 0 et 1, ce qui rend la communication série ou le handshake impossibles à faire. Après nous en être rendu compte, nous avons mis des résistances pull-down sur tous les pins utilisés, afin de se s'assurer que lorsque rien n'est branché, ces pins sont à l'état bas.

Ensuite, il y a eu plusieurs erreurs réalisées au niveau du type de pins à utiliser. J'ai d'abord essayé d'utiliser des pins analogues pour le handshake, mais je me suis rendu compte des nombreux défauts de cette idée. Tout d'abord, pour le handshake, j'ai tenté de comparer la valeur reçue à un certain seuil. Cependant, je croyais que ces pins prenaient des valeurs de 0 à 255, ce qui n'est pas le cas, les valeurs vont de 0 à 1023. De plus, la valeur prise par un pin analogue peut varier en fonction de l'alimentation du microprocesseur. Plus l'on va loin sur le chemin par exemple, ou si la batterie de la brique maître faibli, la valeur renvoyée ne sera plus la même, ce qui rend la méthode de comparer la valeur reçue à un certain seuil inutilisable. A l'opposé, les pins digitaux prennent un simple Low ou High, la comparaison est simple à réaliser et plus fiable que celle des pins analogiques. De plus, tous les pins analogiques peuvent également fonctionner comme des pins digitaux, nous avons donc changer le programme pour utiliser uniquement des pins digitaux.

Une grande question a concerné les briques formant le chemin. Nous avons dû déterminer comment les différencier, quelle forme leur donner, et surtout quels types de briques choisir. Devant guider un robot, la base était évidemment

tout droit, gauche et droite. Ensuite pour l'apprentissage de la programmation, il a fallu ajouter des briques while et if, qui sont deux éléments essentiels pour programmer. L'instruction while doit être composée de deux briques, une pour le début et l'autre pour la fin du while. Enfin, il y a la brique maître, et par nécessité, une brique de fin qui indique la fin d'un chemin de briques. La forme, ensuite, qui au final est un simple carré, n'en a pas toujours été un. Il nous fallait un moyen de faire en sorte que toutes les briques puissent être connectées entre elles, peut importe la configuration. Nous avons finalement décidé d'utiliser une configuration mâle-femelle, chacun ayant deux côtés mâle et deux côtés femelle. Ce sont les côtés femelles qui ressortent afin d'éviter à un enfant de se blesser sur les pins mâles, qui sont pointus. Enfin, pour les différencier, nous avons décidé de les peindre, avec une couleur par type de brique, ce qui rend à la fois le projet plus coloré pour les enfants, et est très simple à réaliser et à comprendre.

Un autre problème rencontré a concerné la confection des briques.

Lors de la découpe, les planches étant légèrement gondolées, quand l'imprimante découpait les contours de la boîte, celle-ci se mettait à pencher et la découpe des fentes situées au milieu n'était donc pas aux bonnes dimensions. Nous avons alors appris l'existence de plusieurs niveau d'impression. En effet, la découpeuse suit un code couleur et s'occupe d'abord de la gravure en noir puis d'une découpe en rouge et ensuite une en bleu. Pour palier à notre problème il suffisait donc de jouer sur ce code couleur et de mettre en rouge les parties centrales et en bleu les contours.

3.2 Limitations et améliorations

On peut ensuite parler des limitations liées aux choix fait lors de la réalisation de ce projet, des défauts qui ne sont pas critiques et avec lesquels notre projet est fonctionnel mais que l'on pourrait résoudre afin d'améliorer l'ensemble.

La première limitation concerne les timeouts. En effet, la méthode des timeouts fonctionne uniquement si l'on fait l'approximation que le temps d'émission d'un message n'est pas suffisamment affecté par la taille de celui-ci pour qu'on en prenne compte. Dans le cas contraire, il est théoriquement possible, même si ce n'est pas le cas à l'échelle de notre projet, qu'un chemin très long génère un message de longueur correspondante et que le temps d'envoi de celui-ci soit plus long que la durée du timeout, ce qui ferait échouer la communication. C'est un problème qui n'arrive que lors de cas extrêmes mais qui

existe. Il pourrait être résolu en abandonnant les timeouts et en utilisant deux pins de plus pour chaque liaison série. Ces deux pins serviraient à indiquer si une brique est branchée ou non à l'autre extrémité de la liaison série, l'un étant toujours mit à un par la brique, et le deuxième vérifiant si il y a un un sur le pin opposé.

Une limitation qui se présente ensuite est liée au nombre de briques réalisées. Dans le temps qui nous a été attribué, nous avons eu le temps de réaliser dix briques, ce qui nous permet de disposer de briques de chaque type. Cependant, le chemin réalisé va rester assez simple et court. Faire plus de briques nous aurait permis de faire plus de stress tests et de prouver le bon fonctionnement de notre projet. De plus, nous avions à la base prévu de réaliser deux types de chemins différents, plus ou moins complexe à réaliser en fonction de l'âge de l'enfant. Cependant, le type le plus simple, qui implique que notamment que les whiles créent des véritables boucles physiques, implique d'utiliser beaucoup de briques, ce qui n'est pas un luxe dont nous disposons. Il y aura donc une seule façon de réaliser des chemins.

Le choix du microprocesseur et du type de communication faits au début du projet constituent une forme de limitation. En effet, il existe des microcontrôleurs disposant de plusieurs liaisons série par défaut, et qui donc auraient rendu inutile l'utilisation de la bibliothèque softserial, et effacé les problèmes liés à celle-ci. La communication bluetooth a été très simple à utiliser et correspond bien à ce dont nous avons besoin. Cependant, l'utilisation des liaisons série, si pratiques et simples à réaliser, aurait pu être remplacé par d'autres types de communication.

Enfin, l'amélioration la plus importante qu'il n'a pas été possible d'ajouter à notre projet par manque de temps, est l'étape supplémentaire qui nous était proposée dans le sujet, à savoir de proposer une application de programmation par blocs sur ordinateur permettant aux enfants de réaliser leur chemin virtuellement, en utilisant des briques similaires aux briques réelles. Ce pas supplémentaire nous aurait permis de transformer notre jouet 'intelligent' en un véritable outil d'apprentissage à la programmation. Cependant, la configuration actuelle de notre projet se prête tout à fait à la réalisation de ce pas supplémentaire. Le robot est commandé via un module bluetooth, auquel il suffit d'accéder pour le contrôler à distance depuis un ordinateur. La chose complexe à faire est ensuite de réaliser le programme, ce qui demanderait une grande quantité de temps. Notre priorité serait d'abord de réaliser plus de briques afin de prouver notre viabilité à plus grande échelle, ainsi que de travailler un peu plus sur le robot.

Pour finir, je dirai que nous avions aussi un manque d'organisation et de communication en début de projet qui s'est amélioré avec le temps mais nécessite encore du travail. Le mieux aurait été de prévoir réellement un planning à suivre pour ne pas trop s'attarder sans le remarquer sur certaines parties. Un peu comme en entreprise lorsqu'on nous donne des échéances.
Une autre chose aurait été judicieuse, des "réunions" de notre binôme pour expliquer les avancées de chacun. Durant ce projet, une fois nos deux parties lancées, nous ne nous sommes que peu expliqué mutuellement où en était le travail de l'autre et je pense que nous devons encore travailler sur ce point.
Après tout, l'organisation et la communication sont les points fondamentaux de tout travail en équipe.

Conclusion

Lors de ce projet, nous aurons donc appris de nombreuses compétences, tant dans les domaines de l'informatiques que de l'électronique, au niveau de la gestion de projet, et du design des composants spécifiques à l'accomplissement de nos objectifs. Nous avons eu l'occasion de créer un prototype complexe à partir d'une simple description de quelques lignes, en passant par l'analyse du sujet, le choix de tous les composants à utiliser, les méthodes de communication, jusqu'à l'obtention d'un projet réalisable dans le temps imparti. Nous avons ensuite travaillé à réaliser les différentes parties du projet, en nous basant sur une séparation du projet en deux parties, électronique et informatique, afin d'optimiser notre travail. L'entraide a bien évidemment été une composante importante afin de s'assurer d'avancer à un rythme acceptable, jusqu'à parvenir à atteindre notre objectif final, la réussite de notre projet. Ce projet aura donc été très instructif pour nous, à de nombreux niveaux, et nous sera très utile en tant que base de connaissances dans certains domaines que l'on avait pas eu l'occasion d'explorer auparavant.