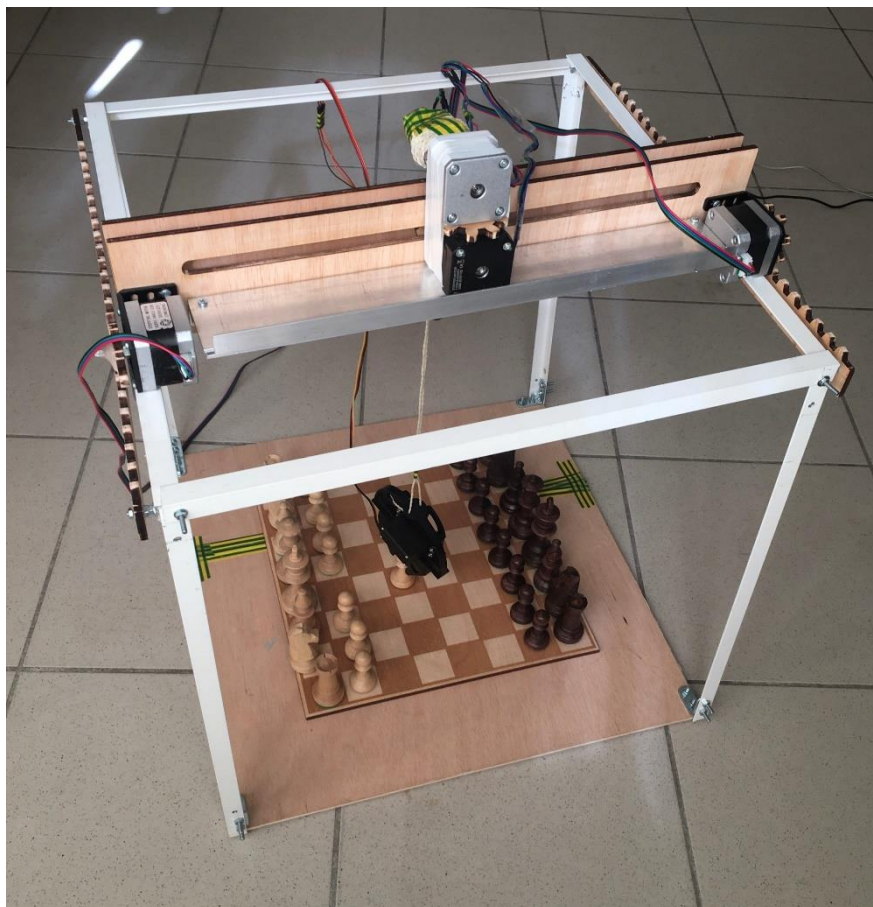


Projet Arduino Rapport Final



JEU D'ECHECS ELECTRONIQUE

Binôme : Glesser Emma
Soens Arthur

Encadrants : P. Masson
N. Adberrahmane

SOMMAIRE

1/ Notre projet

1.1/ Motivations, problématique	3
1.2/ Cahier des charges	3

2/ Objectifs

2.1/ Objectifs à réaliser	4
2.2/ Liste des objectifs atteints	4

3/ Planning, diagramme de Gant

5

4/ Matériel et utilisation

4.1/ Composants électroniques	6
4.2/ Composants additionnels	6

5/ Fonctions et schémas

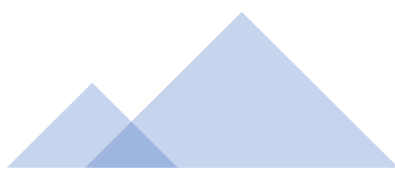
5.1/ Déplacements latéraux	7
5.2/ Plateau de jeu	7
5.3/ Pince	8

6/ Problèmes rencontrés et solutions trouvées

9

7/ Conclusion et bibliographie

10



1/ Notre projet

1.1/ Motivations et problématique

L'idée de départ était d'avoir un projet à la fois ludique et accessible à un grand nombre de personnes. De là, l'un des membres du binôme ayant un intérêt pour les échecs, est venue l'idée d'un tel jeu mais pouvant être partagé.

Il fallait pour cela ajouter la motricité nécessaire au déplacement des pièces sur un échiquier et une interface de commande.

Ce sont ces réflexions qui ont menées à l'élaboration de notre projet du jeu d'échecs électronique.

Problématique : « Comment rendre un jeu manuel accessible au plus grand nombre ? »

1.2/ Cahier des charges

Notre projet de jeu d'échecs électronique devait disposer des fonctions suivantes :

- réaliser la structure mécanique permettant de déplacer une pince dans les 3 axes au-dessus d'un plateau d'échiquier.
- recevoir l'instruction de jeu d'un ou des 2 joueurs sur une interface électronique comme un ordinateur ou un smartphone.
- calculer le mouvement à effectuer en vérifiant les règles du jeu d'échecs puis envoyer les commandes aux différents organes moteurs.
- le cas échéant, libérer au préalable en la déplaçant, une pièce déjà présente sur la case d'arrivée du plateau.

En option, et en fonction du temps encore disponible, les fonctions suivantes pourraient être ajoutées:

- ajouter une interface graphique permettant une meilleure représentation du jeu et des coups à jouer.
- ajouter une reconnaissance vocale pour les personnes dans l'incapacité de taper une instruction.



2/ Objectifs

2.1/ Objectifs à réaliser

Pour pouvoir réaliser notre projet, nous avons défini les principales tâches :

- déplacer une pince le long d'un plan, plus exactement déplacer la pince à une case de l'échiquier donnée.
- se déplacer précisément dans l'espace, cela inclut de monter ou descendre la pince en fonction des pièces concernées.
- prendre une pièce quelconque et la déplacer. Ici il s'agit non seulement de savoir prendre une pièce mais aussi de déterminer si la case d'arrivée est libre. Dans le cas contraire, il faut enlever la pièce occupant cet espace avant de mouvoir la première.
- ajouter une communication RF pour pouvoir transmettre les instructions.
- intégrer une interface graphique pour jouer.
- inclure les règles du jeu d'échecs pour déterminer si un coup est autorisé ou non et renvoyer un message d'erreur si ce n'est pas le cas.

2.2/ Liste des objectifs atteints

Compte-tenu des nombreux problèmes auxquels nous avons été confrontés, le temps est venu à manquer et plusieurs objectifs n'ont pas pu être réalisés. Toutefois, les fonctions primaires ont été réalisées et le jeu est presque opérationnel.

Nous pouvons ainsi :

- déplacer une pièce d'une case à l'autre de l'échiquier,
- recevoir le coup effectué par un joueur via un smartphone et un module Bluetooth,
- le transmettre et effectuer le déplacement adéquat y compris le déplacement d'une pièce préalable.

3/ Planning, diagramme de Gant

3.1/ Diagramme de Gant (planning prévisionnel)

	Séance 1	Séance 2	Séance 3	Séance 4	Séance 5	Séance 6	Séance 7	Séance 8
Monter la structure (rails, moteurs, carte, ...)	Emma							
Réaliser le programme pour se déplacer à une case donnée								
Ajout de la pince et des pièces du jeu								
Programme prise d'une pièce								
Communication RF								
Interface graphique								
Ajout du clic sur l'interface graphique								

3.2/ Planning réel

	Séance 1	Séance 2	Séance 3	Séance 4	Séance 5	Séance 6	Séance 7	Séance 8
Montage de la structure								
Se déplacer sur un plan, à une case donnée								
Se déplacer dans l'espace								
Pince et fonctions reliées								
Communication RF								
Interface	Objectifs non atteints							
Commande vocale								

Comme le montre le diagramme de Gant, le planning de départ n'a pas pu être tenu. Cela s'explique notamment par les problèmes rencontrés dont la résolution a mis plus de temps que prévu. Il a aussi fallu monter toute la structure, ce qui nous a pris énormément de temps. De plus, chacun a aidé l'autre à un moment sur une tâche qui lui a été attribué car tout n'était pas réalisable seul. En particulier en ce qui concerne le bricolage et l'ajustement des programmes à la réalité du plateau, c'est pourquoi il n'y a pas de noms sur le deuxième planning.

C'est pourquoi nous n'avons pas réussi à implémenter une interface graphique et une reconnaissance vocale ; la mise au point des fonctions primaires n'étant pas encore terminée.

4/ Matériel et utilisation

4.1/ Composants électroniques



4 moteur Pas à Pas Nema17 :

2 pour l'axe x, 1 pour l'axe y et 1 pour l'axe z. Il s'accompagne de roues dentées et de crémaillères pour transmettre le mouvement.



4 drivers A4988 :

il permet de contrôler les moteurs Pas à Pas



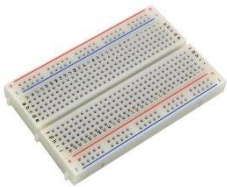
1 servomoteur HS 422 : il permet d'ouvrir et fermer la pince à un angle programmable



1 module Bluetooth HC-06 : pour établir une communication RF



1 pince : Elle sert à attraper les pièces d'échecs



1 plaque de test : pour relier tous les composants entre eux et à la carte Arduino



1 carte Arduino Uno

2.2/ Composants additionnels

Des condensateurs chimiques 100μF 25V : ils permettent de lisser les appels de courant des bobines des Nema17 et ainsi de ne pas faire s'écrouler la tension de la carte Arduino et de la relancer en boucle.

Des résistances entre 30kΩ et 100kΩ : il s'agit de résistances câblées en pull-up afin de pallier les tensions flottantes des entrées I/O de l'Arduino avant leur initialisation en entrée ou sortie. (cf 6/Problèmes rencontrés et solutions trouvées)

Beaucoup d'adhésifs (pour les faux-contacts), de colle, de vis et de câbles pour relier toutes les entrées entre elles et faire fonctionner notre projet !

5/ Fonctions et schémas

5.1/ Déplacements latéraux

Nous avons ici les fonctions de notre programme qui concernent les déplacements sur un plan, c'est-à-dire se déplacer sur l'échiquier. Elles contrôlent les mouvements effectués par 3 moteurs Nema17 :

- *avancer_X_caseX(int x)* et *avancer_X_caseY(int y)* : ces deux fonctions permettent de se déplacer respectivement sur les colonnes → puis les lignes. ↑
- *revenirAOrigine(int x, int y)* : elle permet à la structure de revenir se placer à la case d'origine en dehors de l'échiquier. ↓ ←
- *quelNbrCase(char a)* : comme les cases sont codées avec des lettres pour les colonnes il a fallu convertir cette donnée en nombre pour utiliser une matrice 8x8 représentant l'échiquier.

(voir schéma 1 p.8)

5.2/ Plateau de jeu

Nous avons également des fonctions concernant le jeu et plus précisément des fonctions de vérifications. Toutes dépendent de la matrice 8x8 (échiquier) qui se met à jour à chaque coup :



- *verifierPiecePrise(int x, int y)* : elle renvoie un booléen indiquant la présence ou non d'une pièce sur la case d'arrivée.
- *verifierTaille(int x, int y)* : elle renvoie aussi un booléen qui indique s'il s'agit d'une grande ou d'une petite pièce.
- *piecePrise(int x, int y)* : c'est la fonction qui s'occupe d'enlever au préalable la pièce prise sur l'échiquier avant de bouger la première pièce.
- *changerPiecePlateau(int xD, int yD, int xA, int yA)* : elle permet de mettre la matrice à jour sur des coups joués.

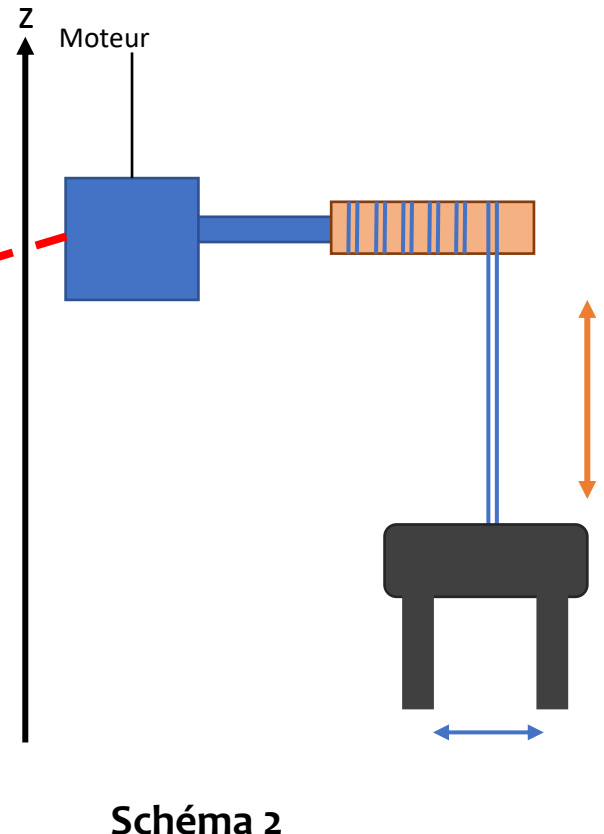
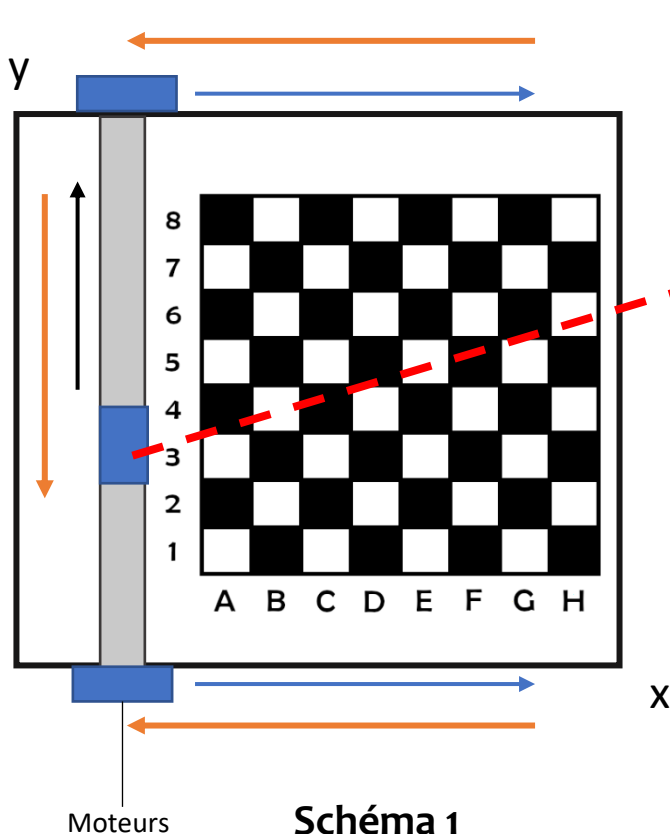


5/ Fonctions et schémas (suite)

5.3/ La pince (voir schéma 2)

Concernant la pince, nous avons plusieurs fonctions permettant de la mouvoir :

- `ouvrirPince()` et `fermerPince()` : elles contrôlent le servomoteur de la pince pour saisir les pièces. 
- `descendreLaPince(int x, int y)` et `remonterLaPince(int x, int y)` : elles permettent de descendre ou remonter la pince en fonction de la pièce sur la case (x, y). Si c'est une petite pièce elle descendra jusqu'au bout de sa course. Sinon elle s'arrête un quart de tour avant. 
- `remonterPinceDepart()` : fonction du setup. Avant de brancher la carte Arduino les moteurs sont hors tension et la pince descend au plus bas par la gravité. On la remonte donc au préalable pour les déplacements futurs.



6/ Problèmes rencontrés et solutions trouvées

Vibrations lors de l'initialisation de l'Arduino :

Lorsque nous connectons l'Arduino à l'ordinateur pour lancer notre programme, les 4 moteurs Nema17 se mettaient à tourner de manière totalement aléatoire. Cela était dû aux entrées I/O de l'Arduino qui par défaut sont en input, de ce fait la tension présente à leur borne est totalement flottante et varie entre 0 et 5V. Par conséquent les moteurs recevaient des informations aléatoires et effectuaient des mouvements incompréhensibles. Nous avons résolu le problème en mettant sur les signaux Enable de chaque driver des résistances pull-up connectées au 5V afin de forcer chaque driver en position off quelque soient les signaux aléatoires captés par les entrées step et dir durant cette phase de mise sous tension.

Moteurs Nema17 tournant dans le vide :

Les moteurs entraînent des roues dentées, mais du fait des frottements répétés le trou pour glisser l'axe des Nema17 dans les roues dentées s'est lissé et les moteurs tournaient dans le vide. On a résolu le problème en mettant un clou et beaucoup de thermocolle pour fixer l'axe et la roue dentée ensemble.

Frottements importants :

Nous avons pas mal de pièces en bois qui engendraient des frottements. Pour certaines nous les avons remplacées par du métal car le coefficient de frottement est moindre.

Oscillations de la pince :

Du fait des déplacements x-y, la pince avait des oscillations empêchant une bonne précision, notamment pour la case d'arrivée. On a réduit le problème en rajoutant un poids supplémentaire. La solution définitive aurait été de mettre un axe fixe relié au Nema17_z comme une autre crémaillère.

Communication Wifi remplacée par du Bluetooth :

Au départ, nous avons tenté d'établir une connexion wifi via le module ESP8266, mais créer un réseau wifi était trop complexe pour nos connaissances. De plus, il y avait beaucoup d'interférences donc la communication n'était pas fiable. Nous nous sommes alors tournés vers le module Bluetooth HC-06 assez simple d'utilisation et l'application Bluetooth Electronics.

7/ Conclusion et bibliographie

7.1/ Conclusion

Bien que n'ayant pas toutes les options prévues au départ notre projet est fonctionnel. Il reçoit les instructions, les transmet à la carte Arduino via une interface Bluetooth et effectue le mouvement demandé.

En travaillant en binôme sur ce projet, nous en avons retiré plusieurs bénéfices comme par exemple le fait de mener un premier projet à bien en travaillant en équipe. Nous avons appris à surmonter beaucoup de problèmes insoupçonnés tout en continuant d'avancer sur d'autres tâches pour ne pas prendre trop de retard.

En résumé, cela a surtout été une première approche de notre futur métier pour nous préparer doucement à ce que sera notre quotidien dans quelques années : le respect d'un cahier des charges et des dates limites en ayant des contretemps.

Nous avons réfléchi à plusieurs améliorations possibles de notre projet. On peut rajouter la dimension sonore et commander le programme par reconnaissance vocale via l'API du MIT ou tout simplement inclure les règles du jeu d'échecs pour le rendre vraiment opérationnel.

Si nous nous sommes concentrés sur les échecs, la structure peut être réutilisée pour d'autres applications qui dépendent des goûts et besoins de chacun.

7.2/ Bibliographie

Librairie Arduino : <https://www.arduino.cc/en/Reference/Libraries>

Cours de P. Masson sur les différents composants électroniques et robotiques :

<http://users.polytech.unice.fr/~pmasson/Enseignement-arduino.htm>