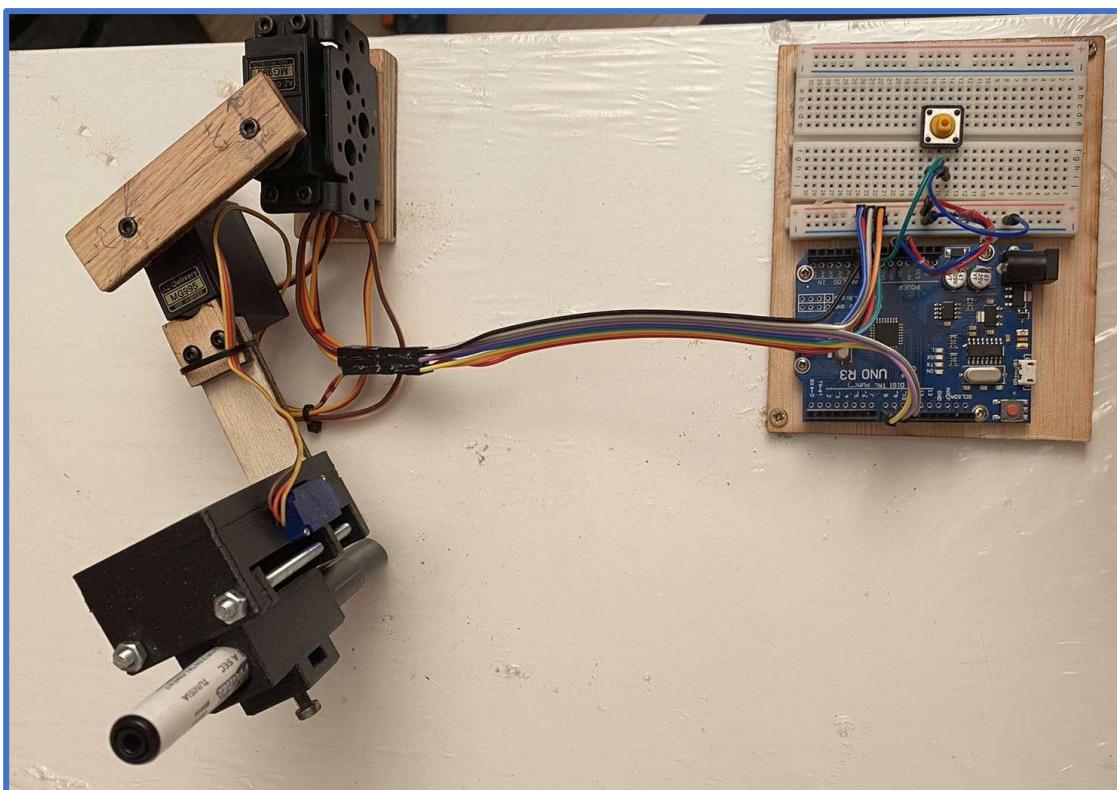


RAPPORT DE PROJET ARDUINO

LE BRAS DESSINATEUR



Ce projet Arduino est un bras composé de servomoteurs et permettant de dessiner, sur une feuille, différentes formes et lettres à partir de coordonnées définies au préalable.

SOMMAIRE

- **CAHIER DES CHARGES INITIAL & FINAL**
- **SCHÉMA ÉLECTRIQUE DU PROJET**
- **ALGORITHME DE FONCTIONNEMENT**
- **COÛT DU PROJET :**
 - **LE MATÉRIEL**
 - **COÛT INGÉNIEUR**
- **PLANNINGS PRÉVISIONNEL ET RÉEL**
- **LES PROBLÈMES**
COMMENT LES AVONS-NOUS SURMONTÉS ?
- **CONCLUSION :**
 - **LE RENDU FINAL**
 - **APPORTS DU PROJET**
 - **PERSPECTIVES D'AMÉLIORATION**

BIBLIOGRAPHIE :

- ❖ <http://users.polytech.unice.fr/~pmasson/Enseignement/Elements%20de%20robotique%20avec%20arduino%20-%20Moteurs%20-%20Projection%20-%20MASSON.pdf>
DOC SERVOS
- ❖ <http://www.jcbmathsandco.fr/le-wonder-bras-before-import/>
THÉORÈME D'AL KASHI
- ❖ <https://www.tinkercad.com/dashboard>
MODÉLISATION ARDUINO
- ❖ <https://www.locoduino.org/spip.php?article106>
LIBRAIRIE SERVO
- ❖ <https://www.locoduino.org/spip.php?article106>
<https://www.locoduino.org/spip.php?article129>
POINTEURS CODE C++

I- CAHIER DES CHARGES INITIAL et FINAL

1.Cahier des charges Initial

Présentation :

L'objectif est de réaliser un bras robotique capable de dessiner des lettres ayant des coordonnées rentrées en paramètres dans notre code. Le bras sera capable d'aller à des positions précises à partir d'angles calculés puis envoyés aux servomoteurs.

Fonctionnalités :

- Bras articulé
- Possibilité de baisser et de lever le stylo
- Capable de dessiner des lettres et une phrase

Contraintes du projet :

- Construction du bras avec des éléments que nous ne possédons pas
- Rédaction d'un rapport individuel à l'issue de chaque séance
- Présentation du projet avec diaporama

2.Cahier des charges Final

Présentation :

Nous avons réussi à réaliser un bras robotisé capable d'écrire une lettre dans une zone de dessin. Le bras peut dessiner des formes simples et n'importe quelle lettre à condition qu'on rentre les coordonnées des points caractéristiques de la forme voulue dans la zone de dessin définie.

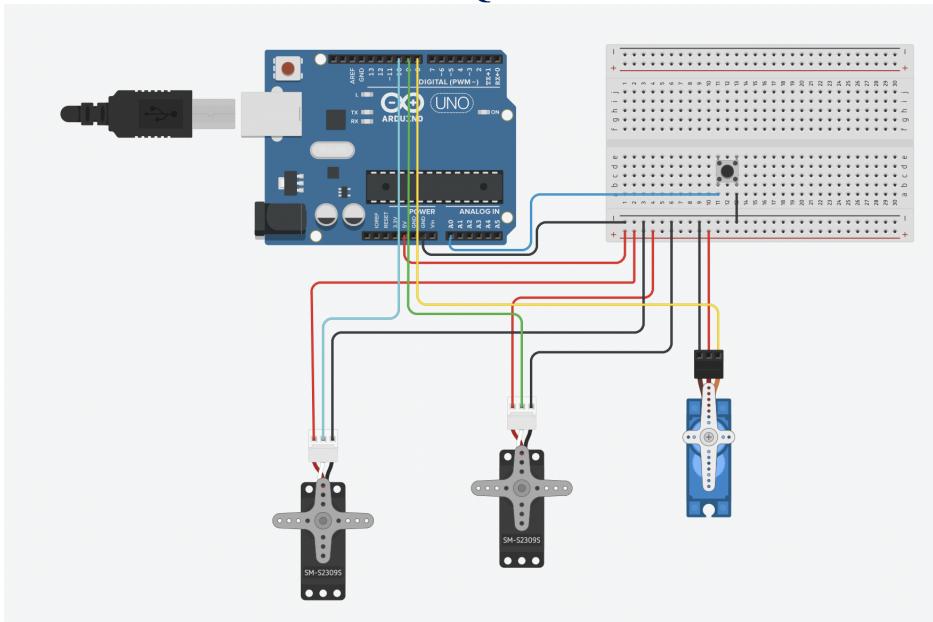
Fonctionnalités :

- Contrôle de la position du robot
- Faire lever le stylo et le baisser

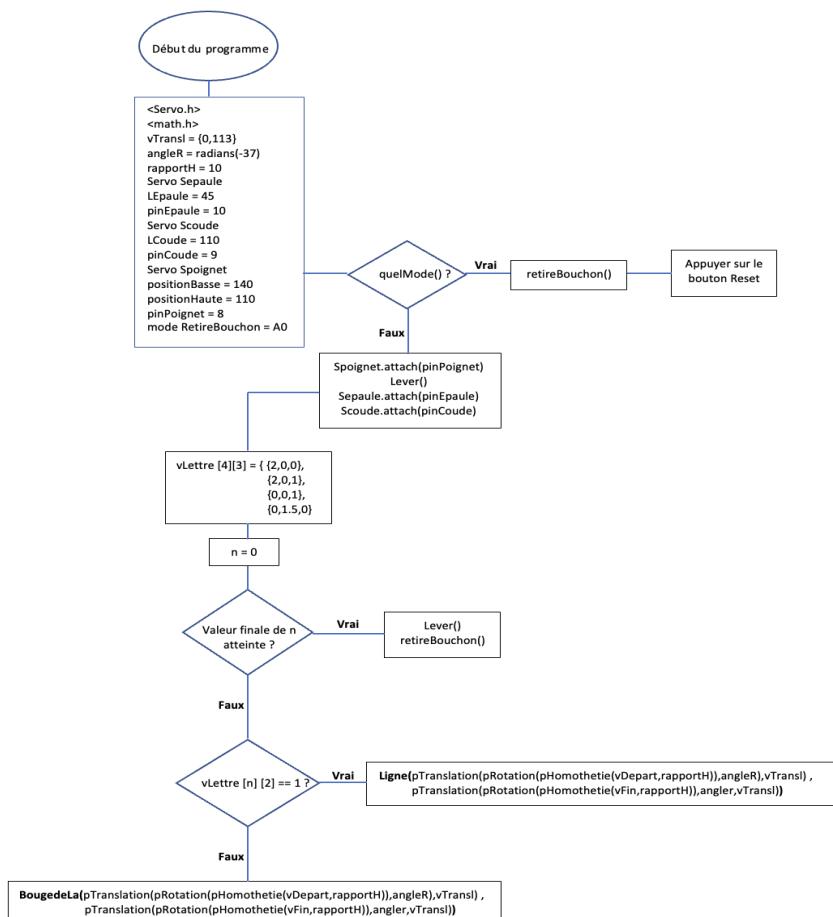
Contraintes Techniques :

- L'ensemble des contraintes techniques ont été respectées

II- SCHÉMA ÉLECTRIQUE DU PROJET

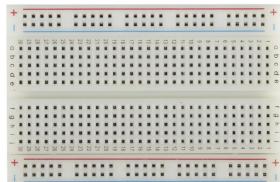


III- ALGORITHME DE FONCTIONNEMENT

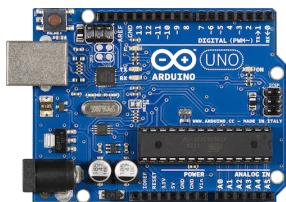


IV- COÛT DU PROJET

1.Le matériel



- ❖ Une plaquette : 3,79€



- ❖ Une carte Arduino Uno : 3,55€



- ❖ 2 "gros" servomoteurs MG995 : 15,60€



- ❖ 1 "petit" servomoteur SG90 : 3,39€

- ❖ 1 stylo Velleda & 1 table "style Ardoise Velleda"

2.Coût ingénieur

En partant de la base d'un salaire brut annuel fixé à 38k euros pour 1600h de travail, nous estimons que le coût ingénieur de ce projet a coûté **au total 3610€**.

En effet, nous avons eu 8 séances de cours, chaque séance durant 3 heures.

De plus, nous estimons notre travail en dehors des séances, bien supérieur : environ 13 séances de 4h (nous avons beaucoup travaillé sur la partie mathématique, modélisation via un tableau Excel et sur le code informatique durant les vacances, ainsi que certains après-midis après les cours).

Ce qui arrive donc à 76 h de travail par ingénieur, et donc finalement 152h pour nous deux.

V- PLANNINGS PRÉVISIONNEL ET RÉEL

Ci-contre, voici donc le planning prévisionnel que nous avions réalisé dans un premier temps.

La partie **grise** correspond à un travail mutuel

La partie **rose** correspond au travail de Charlotte

La partie **bleue** correspond au travail d'Evann

PRÉVISION	SÉANCE N°							
	1	2	3	4	5	6	7	8
Compréhension Générale								
Découpe bois								
Impression 3D								
Programmation fonctions lever/baisser + goTo								
Conversion en ms								
Montage du bras								
Programmation fonctions ligne/bougeDeLa								
Recherche tracé d'une ligne								
Programmation fonctions Homothétie/Translation/Rotation								
Recherche des angles								

Et maintenant, voici la réalité de ce que nous avons réalisé lors des séances (sachant qu'une partie majeure de notre projet a été réalisée en dehors des séances, durant les vacances entre les séances 6 à 8).

RÉALITÉ	SÉANCE N°							
	1	2	3	4	5	6	7	8
Compréhension Générale								
Découpe bois								
Impression 3D								
Programmation fonctions lever/baisser + goTo								
Conversion en ms								
Montage du bras								
Programmation fonctions ligne/bougeDeLa								
Recherche tracé d'une ligne								
Programmation fonctions Homothétie/Translation/Rotation								
Recherche des angles								

VI- LES PROBLÈMES - COMMENT LES AVONS-NOUS SURMONTÉS ?

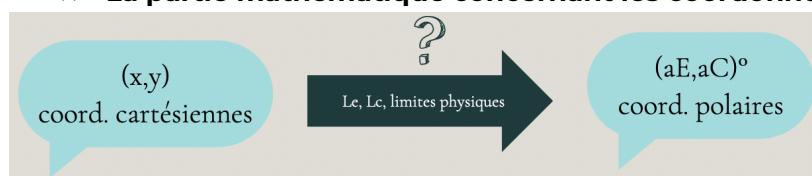
Afin de mener à bout notre projet, nous avons rencontré de nombreux problèmes dont certains étant majeurs car ils constituaient la base même du projet. Pour faire un bras robotisé capable de dessiner, la partie mathématique autour des coordonnées cartésiennes et polaires était conséquente. Mais, nous ne nous sommes pas réellement rendu compte au départ de l'importance de cette partie de réflexion et de modélisation. Nous pourrions dire que nous nous sommes "lancé un peu trop rapidement" dans le vif du projet, en construisant directement notre bras et en essayant simplement d'ajouter un code pris sur internet et voir si cela marchait. Mais la réalité "nous a vite rattrapé", nous nous sommes rendu compte que sans compréhension des formules, nous serions incapables de corriger les erreurs du code afin de parvenir à notre objectif final.

Voici le lien tout d'abord de la modélisation via OnShape que nous avons réalisée afin de modéliser notre bras et les rotations des servomoteurs associées :

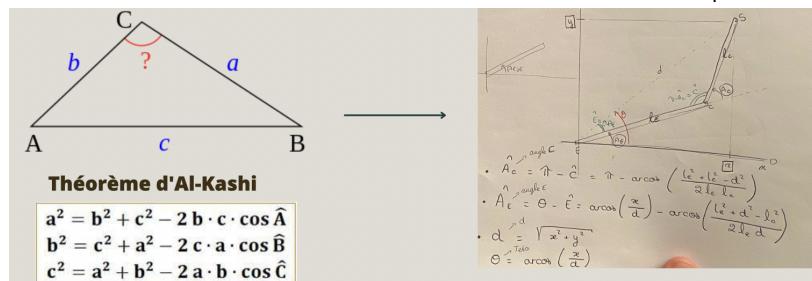
<https://youtu.be/4PCJIPXh2VQ>

Voici donc maintenant les démarches que nous avons adoptées, les problèmes liés et les solutions que nous avons utilisé pour les surmonter :

★ La partie mathématique concernant les coordonnées



- Ici, la démarche est plutôt simple. Nous partons de coordonnées cartésiennes (prédéfinies par nous-mêmes) et nous devons les transformer en angles afin de pouvoir envoyer ces angles grâce à la fonction `.write()` aux servomoteurs. Ils iront finalement à la position que nous souhaitons.



- Le problème que nous avons rencontré a été de comprendre qu'il fallait utiliser le théorème d'Al-Kashi ainsi que réussir à l'appliquer correctement dans notre cas, avec nos noms d'angles etc...
- Afin d'utiliser correctement ce théorème, nous avons représenté notre cas au brouillon sur un papier pour ensuite le transformer en code informatique.

★ La conversion des angles en microsecondes

```
//Convertit en microsecondes (= la durée d'impulsion correspondant à l'angle en degrés)
float angleE_micro = (angleEwrite * 10.3) + 544; // angleE_micro est l'angle envoyé au servo avec une fonction Sepaule.writeMicroseconds().
float angleC_micro = (angleCwrite * 10.3) + 544; // angleC_micro est l'angle envoyé au servo avec une fonction Scoude.writeMicroseconds().
```

- Convertir un angle de degrés en microsecondes permet, dans le cas de servomoteurs, d'apporter de la précision lors du tracé, ce dont nous avions besoin afin d'avoir une lettre la plus ressemblante à la réalité possible.

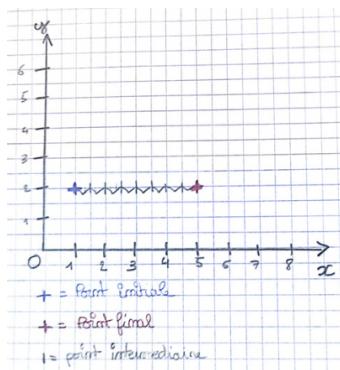
- Le problème a été de trouver la formule ci-dessus permettant la conversion.
- Pour cela, nous avons parcouru la librairie <Servo.h> où nous avons trouvé les informations nécessaires concernant les calibrations du servo MG995.

★ Trouver la bonne zone d'écriture

https://docs.google.com/spreadsheets/d/1hcmHztFNOQue9LK7nhtLLEV_sXeA19i0/edit?usp=sharing&ouid=115329628837362060140&rtpof=true&sd=true

- Voici un document Excel que nous avons réalisé permettant de trouver la bonne zone d'écriture
- En effet, nous avons rencontré de nombreux problèmes dans la fonction goTo car les angles donnés n'étaient pas acceptés par le code, ne pouvant être envoyés aux servos.
- Nous avons donc dû créer des fonctions Homothétie, Rotation et Translation permettant de définir une zone d'écriture correcte et où le stylo était apte à se positionner.

★ Trouver comment tracer une ligne avec des servomoteurs



- Ici, nous avons dû réfléchir à la façon dont les servomoteurs tracent une ligne. Nous nous sommes donc rendu compte qu'ils ne pouvaient se déplacer que sous forme de cercles.
- Pour tracer une ligne, il a donc fallu établir une fonction permettant de calculer les points intermédiaires. Ce qui permettra donc de faire se déplacer le stylo sur un endroit tout petit et le tracé ressemblera de plus en plus à une ligne et non plus à des arcs de cercle.

★ La partie informatique

Pour cette partie, nous nous sommes énormément aidés de ChatGPT afin de corriger notre code. Nous avons essayé de réaliser le code le plus propre possible en déclarant un maximum de variables au début de notre code et en utilisant des tableaux afin que toutes les coordonnées "ressemblent" à la manière dont nous les écrivons en mathématiques (sous forme de vecteurs) et afin que les formules ressemblent également aux mathématiques (sous forme de matrices).

- En effet, toutes nos fonctions et formules étaient correctes, toutes les variables étaient déclarées correctement. Mais des problèmes de bug sont apparus.
 - ➔ Premièrement, nous nous sommes rendu compte de l'importance de déclarer des int partout ou bien des float partout mais JAMAIS l'on ne peut associer un int avec un float
 - ➔ Ensuite, une des solutions majeures à nos bugs apportée, a été l'utilisation de pointeurs. Nous ne connaissons pas profondément leur signification mais permettent d'accéder directement à l'adresse mémoire associée.

VII- CONCLUSION

1.Le rendu final

Finalement, nous avons réussi à tracer un “É”.

Voici le lien de la vidéo : [Le Bras Dessinateur - Projet Arduino PEIP2 Polytech Nice Sophia](#)

Nous voulions tracer “P \heartsuit LYTECH” mais la zone de dessin n’étant pas totalement respectée, seulement le P et le \heartsuit se traçaient. Il était un petit peu trop tard pour finaliser cette phrase complète mais nous avons bien démontré que notre bras était capable de dessiner par le biais de coordonnées préalablement définies et de lignes droites.

Voici le code permettant de tracer la lettre “É” :

```
// EXEMPLE 1 : TRACER LA LETTRE "É"

//étape 1 : définir le tableau vLettre --> CHANGE EN FONCTION DE CHAQUE LETTRE QUE L'ON VEUT ÉCRIRE
float vLettre[9][3] = { {2,0,1},
{0,0,1},
{0,1.5,0},
{1.5,1.5,1},
{0,1,1},
{0,3,1},
{2,3,0}, //la valeur [0] correspond au x, [1] correspond au y, [2] correspond à la position d'écriture(levé ou baissé)
{1,3,1},
{2,4,0} }; // [9] correspond au nombre de points dont on a besoin pour tracer cette lettre --> MODIFIABLE SI ON CHANGE DE LETTRE !!

//étape 2 : calculer la taille du tableau vLettre
int longLettre = sizeof(vLettre) / sizeof(vLettre[0]);

float vD[3];
float vF[3];

//étape 3 : faire un for qui parcourt tout le tableau vLettre
for(int n=0 ; n<longLettre - 2 ; n+=3){

    // recopie les coordonnées de la lettre n dans vD et celles de la lettre n+1 dans vF
    for (int lig = 0; lig<3; lig++) {vD[lig] = vLettre[n][lig];}
    for (int lig = 0; lig<3; lig++) {vF[lig] = vLettre[n+1][lig];}

    //étape 4 : faire un if qui analyse le troisième élément du tableau vLettre => si il est égal à 1, on utilise la fonction ligne qui permet d'écrire
    if(vLettre[n][2] == 1){

        // commence le tracé de la lettre
        ligne( pTranslation(
            pRotation(
                pHomothetie(
                    vD,
                    rapportH
                ),
                angleR),
                vTransl)

            , pTranslation( pRotation(
                pHomothetie(
                    vF,
                    rapportH
                ),
                angleR),
                vTransl)
        );
    }
    //puis mettre un else => si c'est pas égal à 1, on utilise la fonction bougeDeLa qui permet juste d'aller au point suivant sans écrire
    else{
        bougeDeLa( pTranslation(
            pRotation(
                pHomothetie(
                    vD,
                    rapportH
                ),
                angleR),
                vTransl)

            , pTranslation( pRotation(
                pHomothetie(
                    vF,
                    rapportH
                ),
                angleR),
                vTransl)
        );
    };
}

} //FIN LETTRE É
```

Voici les fonctions utilisées dans notre code ainsi qu'une courte description de chacune d'entre elles :

★ **Les fonctions lever() et baisser()**

Elles permettent de mettre le stylo en position d'écriture ou en position relevée.

★ **La fonction goTo()**

Elle permet d'aller à un point d'arrivée en calculant des angles avec le théorème d'Al-Kashi, en passant à des angles en microsecondes puis grâce à la fonction préexistante ".writeMicroseconds()".

★ **La fonction ligne()**

Elle permet de tracer une ligne (avec le stylo en position d'écriture) en calculant des points intermédiaires (nommés "vPointInter") et en utilisant la fonction goTo(vPointInter).

★ **La fonction bougeDeLa()**

Elle permet d'aller d'un point de départ vers un point d'arrivée (avec la fonction goTo()) plus rapidement qu'avec la fonction ligne() car le stylo n'est pas en position d'écriture et donc le calcul des points intermédiaires n'est pas nécessaire. Cela évite de repasser sur une ligne déjà tracée, cela rend donc le tracé plus propre et cela accélère le tracé complet de la lettre.

★ **Les fonctions pTranslation(), pHomothetie() et pRotation()**

Elles permettent de réaliser une translation ou une homothétie ou encore une rotation. Cela permet, dans notre cas, d'aller dans la bonne zone de dessin, et donc de POUVOIR tracer la lettre.

★ **La fonction retireBouchon()**

Elle permet d'amener le bras avant et après le tracé de la lettre dans une position "dans le vide" et non au-dessus de la feuille, pour pouvoir enlever et remettre le bouchon sur le stylo.

2. Apports du projet

Ce projet nous a apporté de nombreuses connaissances en Arduino mais également en informatique. Nous avons également pu et dû appliquer "en pratique" ce que nous avons vu en mathématiques. Ce projet a nécessité un travail en équipe et de comprendre ensemble, sans quoi, tout aurait été bien plus complexe. Nous avons aussi pris conscience au fil des séances du peu de temps qui nous était accordé ainsi que de la complexité du projet vis-à-vis des connaissances que nous avions au départ.

Pour terminer nous avons dû prendre sur nous lors des difficultés et trouver des solutions correctes afin de pallier à chaque problème.

3. Perspectives d'amélioration

Afin de perfectionner encore et encore notre projet, nous pourrions dans un premier temps, comme cité précédemment, écrire "P \heartsuit LYTECH".

Nous aurions pu utiliser des matériaux plus solides que le bois pour relier les servomoteurs pour éviter le "jeu" présent sur le bras, mais également d'agrandir les longueur Épaule et longueur Coude pour avoir une zone de dessin plus large.

Nous pourrions également redessiner le système 3D permettant de tenir le stylo afin de l'alléger un maximum.

Pour finir, nous pourrions planter un système Bluetooth pour permettre à l'utilisateur de "jouer" avec le bras en dessinant lui-même ses lettres sur son smartphone et en l'envoyant au bras.