

# PROJETS ARDUINO – PEIP2

*Année scolaire 20018-2019*

*Imprimante « intelligente »*

**Etudiants : Antoine Blaud  
Raphaël Chuilon**

**Encadrants : Pascal Masson**



## REMERCIEMENTS

Nous tenons à remercier dans un premier temps Monsieur Pascal Masson, notre encadrant durant les cours d'Arduino, pour nous avoir encadré et accompagné pendant toute la réalisation du projet. Il a été disponible lorsque nous avions une question concernant le projet et malgré notre surconsommation en drivers, il a toujours été positif. Dans une même mesure, nous remercions Monsieur Nassim Abderrahmane, assistant Pascal Masson pendant les cours d'Arduino.

Nous remercions aussi Monsieur Roland Blaud, grand-père d'Antoine pour nous avoir grandement aidé pour la construction de l'imprimante.

Et pour finir, nous remercions toutes les personnes qui nous ont aidés, que ça soit en répondant à nos questions ou en nous aidant lors de branchements.



# SOMMAIRE

Introduction .....	7
Chapitre I : Présentation du projet.....	8
I.1. Définition du projet .....	8
I.2. Cahier des charges .....	8
I.2.1. Besoins .....	9
I.2.2. Description du fonctionnement .....	9
I.3. Développement du projet .....	9
I.3.1. Réflexion .....	9
I.3.2. Erreurs à éviter .....	9
Chapitre II : Mise en route du projet .....	10
II.1. Modélisation 3D.....	10
II.2. Construction de la structure .....	10
Chapitre III : Production du projet .....	13
III.1. Élaboration de la base de données .....	13
III.2. Création du réseaux de neurones .....	14
III.3. Utilisation du réseaux de neurones .....	15
III.4. Génération du texte.....	17
III.5. Finalisation de la structure.....	17
III.6. Ajouts des différents modules .....	18
III.7. Transformation de l'image en chemins .....	18
III.8. Transfert d'information et contrôle de la carte.....	20
Conclusion .....	21
Bibliographie .....	22



# Introduction

Notre projet se nomme « imprimante intelligente ». Il a pour objectif d'écrire comme une personne et avec l'encre d'un stylo Bic. On voudrait donc pouvoir analyser, puis recopier ou générer l'écriture de quelqu'un.

On aura donc comme support physique une imprimante construite par nous.

On a choisi ce projet car tous ceux qui ont déjà fait une tâche répétitive d'écriture ont déjà rêvés de pouvoir le délégué mais sans trouver de solution convenable. Avec notre imprimante, on pourra par exemple fournir un pattern qui sera recopier autant qu'on veut.

# Chapitre I : Présentation du projet

## I.1. Définition du projet

Notre projet « Imprimante Intelligente » est une imprimante ayant pour particularité « d'imprimer » avec un stylo et non un jet d'encre.

Elle aura comme fonctionnalité principale de recupérer le contenu d'un fichier texte et de produire en sortit une page de texte écrite manuscrit avec une grande ressemblance de l'écriture d'une personne sélectionnée à l'avance (dans le cadre de ce projet nous choisirons uniquement l'écriture de Raphaël).

## I.2. Cahier des charges

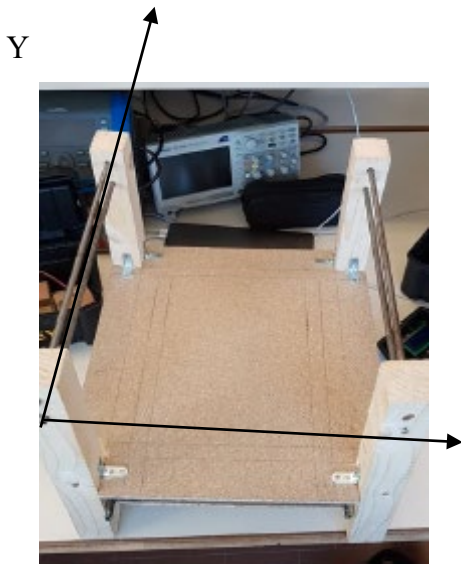
### I.2.1. Besoins

- 4 moteurs pas à pas (3 42SHDC3025-24B)*
- 4 drivers (pour les moteurs 42SHDC3025-24B et)*
- 1 écran LCD 16X2*
- 1 power supplies (source de courant)*
- 1 RFID module*
- 2 830 Breadboard*
- 1 carte Arduino Xplained mini*
- 1 extension de mémoire*
- 1 structure imprimante en bois et fer*
- 1 potentiomètre*
- 3 courroies*
- 4 poulies*
- 8 roulements*
- 1 ventilateur*
- 4 capacités 1 mf*

### I.2.2. Description du fonctionnement

Nous allons utiliser un CNN (Convolutional Neural Network), codé en Python, afin que notre Ordinateur apprenne à lire l'écriture des gens. Pour ce faire, nous avons besoin d'une base de données comprenant des documents manuscrits par des personnes (base de données EMNIST). Nous allons de notre côté écrire des pages de texte, utiliser un algorithme afin de récupérer le plus de mots et de lettres possible sous forme d'image parfaitement cadrées, puis de les fournir en donnée utilisable complémentaire à notre imprimante. Quand cela sera fait, l'ordinateur communiquera avec l'imprimante via le port serial, et cette dernière fera bouger une plateforme sur notre imprimante sur des rails.





La plateforme se déplacera sur l'axe y sur des rails à l'aide d'un moteur qui entrainera une courroie. De même, la plateforme se déplacera selon d'axe x sur des rails, entraînée par une courroie. Le stylo, accroché à la plateforme, se relèvera grâce à un dernier moteur lorsque qu'il en aura besoin, pour changer de mot par exemple.

### **I.3. Développement du projet**

#### **I.3.1. Réflexion**

Nous avons commencé par réfléchir à comment faire le mouvement d'écriture. Comme expliqué dans la description du fonctionnement, nous devions faire un chariot avec un stylo qui se déplace.

Nous devions aussi intégrer des moteurs permettant de déplacer se chariot, et d'aller à des coordonnées précises.

Afin d'imiter l'écriture, on s'est dit qu'on devait générer une image comportant le texte et qu'on devait la transformer en points, qui seront alors transmis à la carte Arduino puis au moteur afin de se déplacer

#### **I.3.2. Erreurs à éviter**

On a aussi réfléchi à ce qui ne fallait pas faire ou alors faire attention.

On s'est dit que on devait faire la structure assez large afin de pouvoir sur la page A4 en entière, car si on devait faire attention aux marges alors ce serait très compliqué.

On s'est aussi dit que on devait générer une image avec exactement la même taille qu'une page A4 afin d'éviter des problèmes d'échelles ou autres.

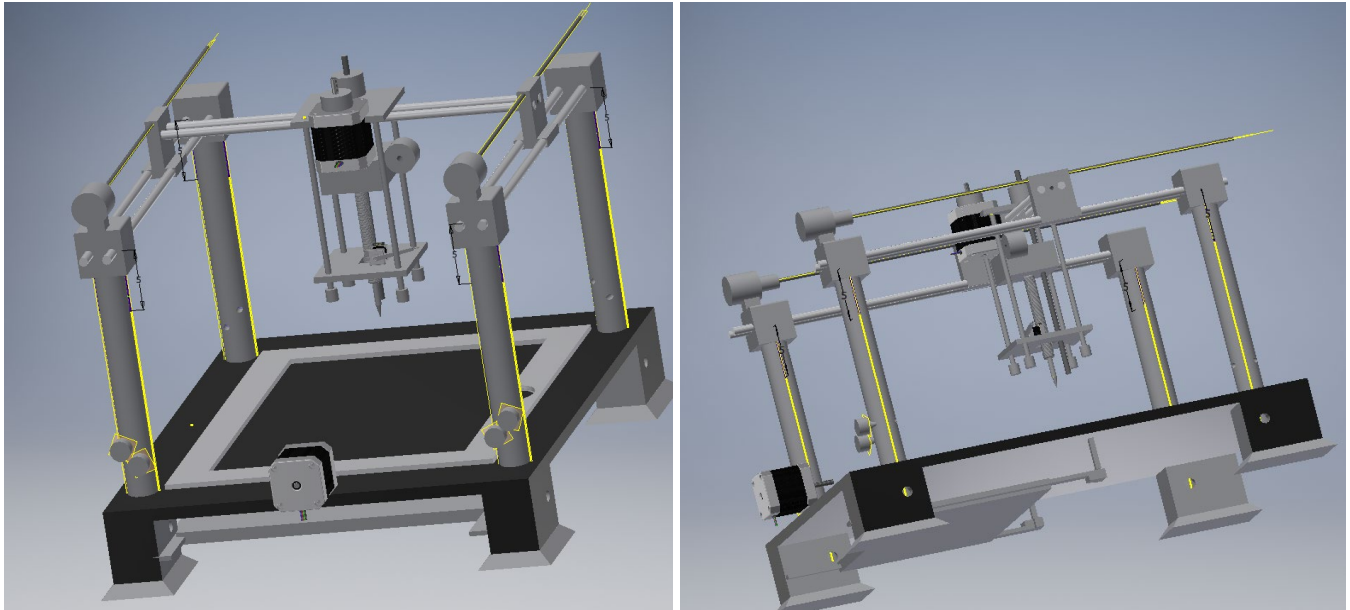
On a aussi réfléchi à mettre une sorte de tiroir sur le dessous de l'imprimante afin d'accueillir toutes la partie électronique, et à notre guise le tirer afin de par exemple modifier quelque chose, ou alors de le pousser afin de cacher les fils et autres.

# Chapitre II : Mise en route du projet

## II.1 Modélisation 3D

On a choisi de modéliser notre imprimante avant toute chose afin d'envisager le plus possible de problèmes avant qu'ils arrivent.

*Voici le modèle 3D :*



On retrouve 4 piliers qui soutiennent les rails et le chariot. La planche permet de poser la feuille, et le contour gris permet de la tenir.

Le tiroir en dessous permet de mettre les composants électroniques. On voulait à la base utiliser un seul moteur (*qui est au premier plan sur la photo de gauche*) pour manipuler le chariot sur l'axe X et on voulait aussi mettre des tiges filetées pour le mouvement mais il se trouve que les tiges filetées coûtent cher donc on a plutôt pris des courroies et installer un moteur de plus. Un micro moteur devait aussi s'occuper de lever le moteur.

## II.2. Construction de la structure

Nous avons commencé à construire notre imprimante tôt par rapport à la date butoir pour ne pas être pressé par le temps. Nous avons donc d'abord construit la base, c'est-à-dire les 4 piliers et la planche sur laquelle les feuilles seront posé. Nous avons choisi d'utiliser du bois comme matériau pour la facilité de mise en forme et coût moins important.

Il y a eu donc quelques séances de sciage (II.1) et de collage, pour arriver au résultat suivant au bout du premier jour de fabrication (II.2).

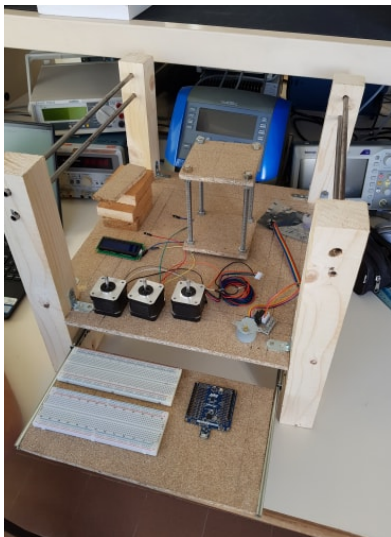


**Figure II.1 :** Découpe du bois pour la structure.

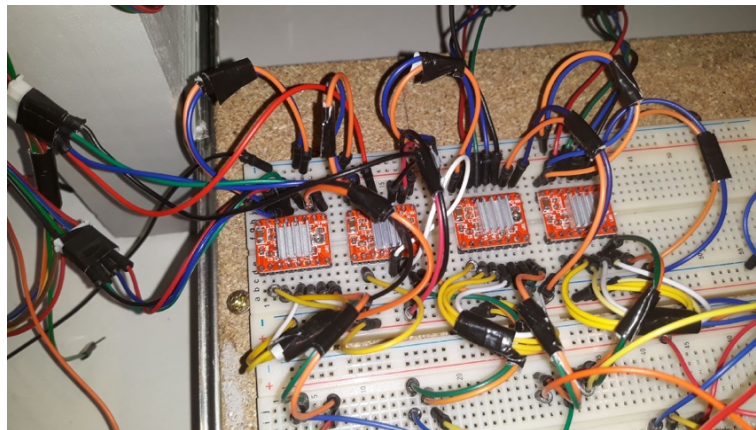


**Figure II.2 :** État de la structure au bout du premier jour de travail.

L'étape suivante a été d'installer les rails. Ci-dessous (II.3), on peut voir que nous n'avons pu faire les rails que sur un seul axe car des imprécisions sur les mesures pour faire les trous qui accueillent les rails nous ont fait perdre beaucoup de temps. Nous avons en parallèle construit le chariot qui guidera le stylo. Vous pouvez voir aussi les branchements nécessaires (II.4) pour le test des moteurs.

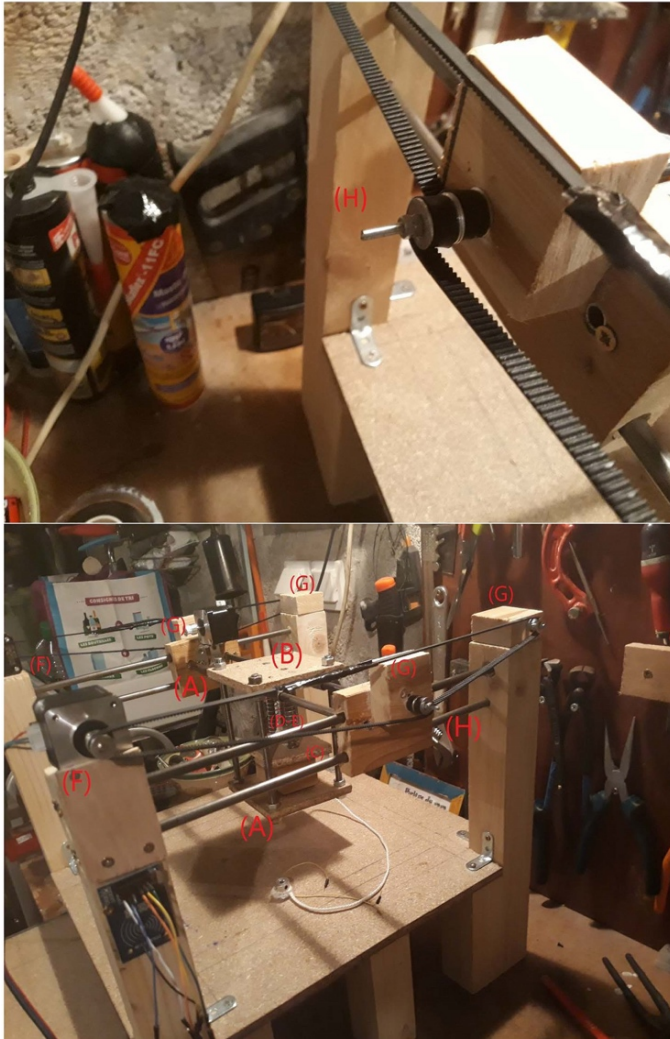


**Figure II.3 :** Structure avec le chariot et quelques composants.



**Figure II.4 :** Branchements pour tester les moteurs et les drivers.





L'étape finale a été de mettre en place les moteurs, les rails manquants et les courroies. Ci-contre (II.5), vous pouvez trouver les détails de la structure finale :

**(A)** : Ensemble de la structure.

**(B)** : Chariot conduisant le stylo.

**(C)** : Le bloc qui maintient le stylo et le relève.

**(D)** : Le bloc est guidé par 2 tubes qui le guide selon l'axe z (« hauteur »).

**(E)** : Nous souhaitions ajouter 2 ressorts permettant d'appliquer une force verticale sur le système « bloc + stylo » sur le sol. Mais nous ne les avons finalement pas ajoutés.

**(F)** : Les 2 moteurs permettent le mouvement du chariot sur l'axe y.

**(G)** : Ces parties en bois servent à aligner les éléments pour installer les poulies.

**(H)** : Les tiges en métal maintiennent les poulies qui conduisent les courroies.

**Figure II.5** : Version finale de la structure, sans les modules et quelques derniers ajouts

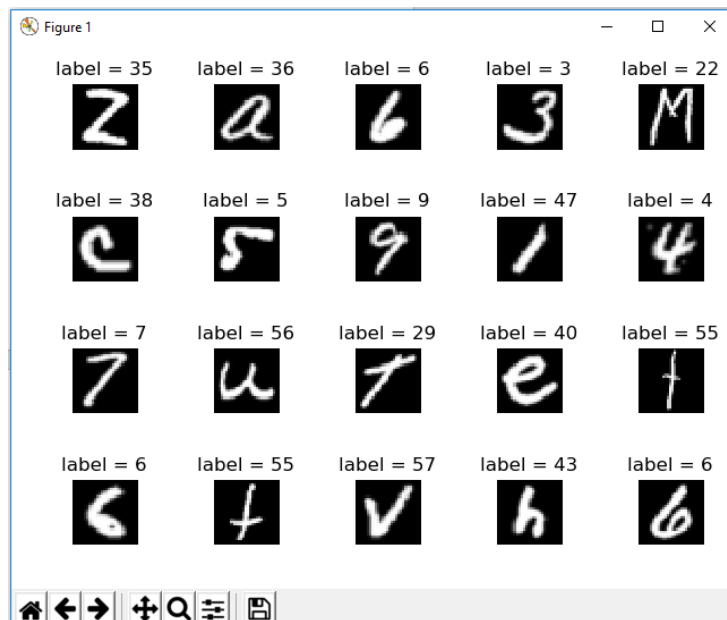
# Chapitre III : Production du projet

## III.1 Elaboration de la base de données

Pour la base de données on voulait utiliser celle de L'EMNIST qui regroupe une grande quantité d'images de caractères qu'on allait utiliser afin d'apprendre à notre programme à écrire.

A la base nous voulions utiliser celle de l'IAM qui regroupe une énorme quantité de mots écrits par plusieurs centaines de personnes et tous labélisés. Cependant ces mots sont en anglais et donc ne servent pas à grand-chose pour nous.

*Voici un petit exemple des images qui se trouvent dans cette base de données :*



*Et voici l'occurrence des caractères :*

Training dataset of NIST database (184,033 uppercase + 155,215 lowercase characters)

Character Class	Number Samples	Character Class	Number Samples	Character Class	Number Samples	Character Class	Number Samples
A	6,085	N	8,192	a	10,205	n	11,943
B	3,191	O	27,737	b	4,642	o	1,800
C	10,221	P	8,340	c	1,794	p	1,589
D	4,033	Q	1,664	d	10,497	q	2,405
E	4,652	R	4,528	e	27,408	r	15,050
F	9,325	S	22,898	f	1,608	s	1,776
G	1,707	T	9,974	g	3,050	t	19,861
H	2,425	U	13,224	h	8,755	u	1,826
I	11,847	V	3,945	i	1,838	v	1,777
J	3,063	W	4,084	j	1,446	w	1,765
K	1,662	X	1,785	k	1,740	x	1,861
L	4,341	Y	4,206	l	15,420	y	1,614
M	9,216	Z	1,778	m	1,731	z	1,814

On a aussi scanner des pages d'écriture de Raphael et élaboré un algorithme permettant de récupérer tous les mots et lettres d'une page scannée afin d'obtenir des images comme celles d'au-dessus.

On a ensuite trié ses lettres à la main et mises dans les dossier correspondant (*lettre A dans le dossier A, etc...*).

Au total on a réussi à obtenir environ 3500 lettres triées.

## III.2 Création du réseau de neurones

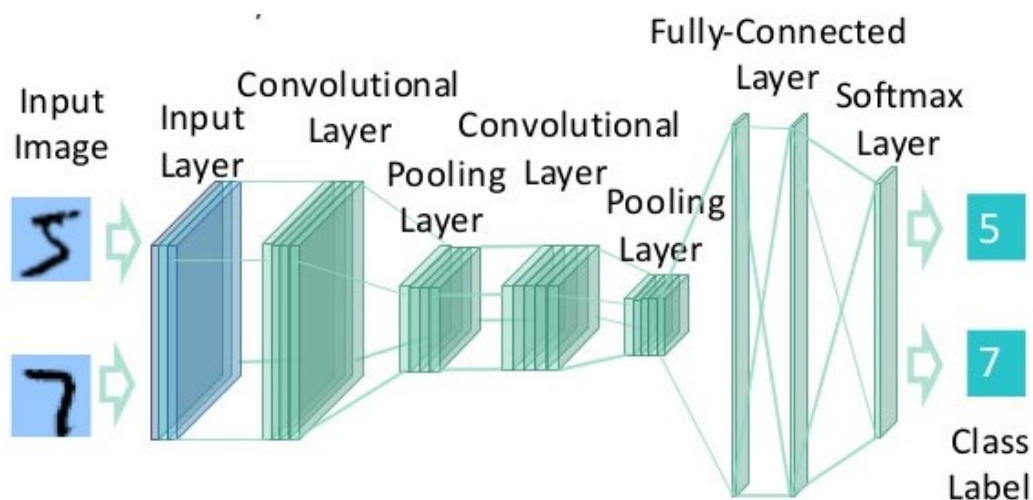
Sur certain problème informatique il est plus facile de créer une intelligence artificielle qui va essayer de le résoudre plutôt que d'essayer de le faire soi-même. Et cela est particulièrement vrai dans la reconnaissance d'image.

On a donc choisi de programmer un réseau de neurone prenant en entrées une image de 26 par 26 pixels et qui en sorti donnait :

- un caractère correspondant à celui sur l'image d'entrée si le résultat était juste
- un label « autre » si l'image correspondait a rien de connu.

On a donc fait une superposition de couches (notamment des couches de convolution ou des couches denses).

*Voici à quoi ressemble notre modèle et le principe de base du réseau de neurones:*



Des couches de convolutions sont en quelques sortent des couches filtrantes qui vont essayer de reconnaître les points clés de l'image, et les dernières couches sont des couches entièrement connectés qui sont donc des neurones reliés entre eux et qui s'ajuste pendant l'apprentissage (*phase d'entraînement*).

Certains détails sur une image étant plus important que d'autres, des neurones vont s'activer ou non à l'obtention d'une certaine information, et suivant leur importance (*poids*), ils vont pouvoir activer ou non d'autres neurones. Chaque neurone a un poids d'entrée et de sortie. Le poids d'entrée a besoin d'être dépassé afin que le neurone s'active et à son tour transmette quelque chose.

### III.3 Utilisation du réseau de neurone

L'objectif de ce réseau de neurones était de pouvoir lire.

Cependant nous avons utilisé que des lettres pour l'entraîner, mais nous voulions réussir à lire des mots. Or l'algorithme permettant d'extraire les caractères d'une page ne nous permettait pas d'extraire les caractères d'un mot.

On a donc essayé en vain de séparer les caractères d'un mot par plusieurs méthodes :

D'abord par densité de pixel, quand on passe d'une lettre à l'autre, normalement la liaison à un mot de pixel blanche qu'une lettre, seulement certains caractères sont trop collés pour pouvoir détecter cette liaison.

*Voici un exemple :*



Si on prend la première lettre par exemple, on peut s'apercevoir qu'il y a un pic de densité de pixel à 50, puis une absence totale de pixel à environ 180. On peut donc en déduire qu'il y a une lettre entre 0 et 180. Cependant si on continue l'analyse du mot on s'aperçoit que ce schéma simple ne se reproduit pas, cette technique ne marche donc pas.

On a donc tenté une autre technique, cette fois par détection de formes.

Voici le résultat :



Cette technique ne marche malheureusement pas non plus.

Il ne nous rester donc plus qu'une solution, celle de détection par balayage. Le principe est simple. On récupère l'image d'un mot, on récupère uniquement les 150 pixels par exemple. On demande à notre réseau de neurones de voir si cette partie du mot pourrai être une lettre. Si il évalue que oui, il enregistre le résultat puis avance de 150 pixels, sinon il tente d'ajuster l'image en récupérant un peu plus à droite ou à gauche puis réévalue.

Cette technique marche plutôt bien.

Voici quelques résultats :





Comme vous pouvez le voir, ces résultats sont plutôt bon, des vérification et corrections ont bien évidemment à faire afin de pouvoir les classer et les utiliser.

Notre réseau de neurone avait pour utilité de classer les images et c'est ce qu'il fait, on va pouvoir donc par la suite utiliser ces images afin de se rapprocher au mieux de l'écriture de Raphael.

### **III.4 Génération du texte**

Maintenant que nous avons nos images de lettres, on va vouloir transformer un le texte contenu dans un fichier en image de mots de Raphael correspondant au texte.

Nous avons donc créé plusieurs classes, représentant des pages, des lignes et des mots.

La technique et de récupérer les images de lettres on de mots dans les fichiers et de les coller les uns à côté des autres de manière successive afin de former des mots.

Afin de simuler les liaisons nous utilisons les mots lu et référencés par notre réseau de neurones afin de récupérer seulement les parties de mots qui nous intéresse.

Les mots sont construits en utilisant des images de lettres ou des parties de mots.

Les lignes sont construites en insérant plusieurs mots.

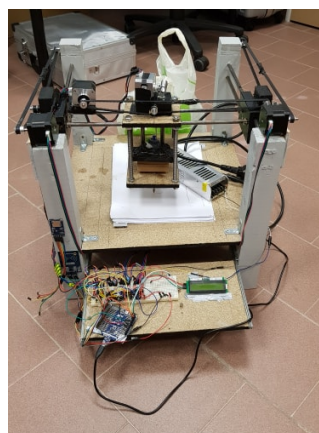
Les pages sont construites en juxtaposant des lignes.

### **III.5 Finalisation de la structure**

Comme dit dans le paragraphe II.2., après avoir terminé la construction de la structure, il nous restait quelques modifications et ajouts à finaliser pour que l'imprimante soit opérationnelle.

Premièrement, il nous manquait les différents modules que nous voulions ajouter à l'imprimante (voir paragraphe III.6.) et dans un deuxième temps, il fallait modifier le chariot pour qu'il puisse accueillir le stylet et qu'on y ajoute un moteur qui relèvera le stylet en temps voulu. Comme pour les autres moteurs, nous avons choisi un système de poulie pour tirer le stylo. Le moteur fait tourner la poulie qui est fixée à la branche du moteur et une corde, qui est nouée au stylo, est enroulée autour de la poulie. Ainsi, lorsque le moteur tourne, la corde tire le stylo vers le haut.

Après cela, l'imprimante était prête pour le fonctionnement. Nous avons ajouté des cales pour stabiliser les feuilles de papier lors de l'impression, nous avons peint l'imprimante pour qu'elle ne soit plus de couleur bois et nous avons rebouché quelques trous qui ont été créés lors l'implémentation des rails dans les piliers de la structure.



### III.6 Ajout des différents modules

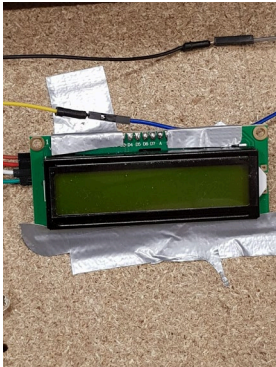
Pour finir l'imprimante, nous avons décidé de rajouter deux modules : le module RFID et un écran LCD 16x2.

Le module RFID (Radio Frequency Identification en anglais) nous permet de déverrouiller l'imprimante lorsque l'on passe notre badge sur le module. Ce système est le même que l'on peut retrouver dans des résidences pour ouvrir une porte d'entrée. Le principe est le suivant : en se rapprochant suffisamment avec le badge, le module va lire l'identifiant du badge. Si c'est celui que nous avons mis dans les conditions, l'imprimante sera déverrouillée. Si elle est déjà déverrouillée, alors l'imprimante se verrouillera.



L'écran LCD a plusieurs fonctionnalités ici :

- La première est de savoir si l'imprimante est verrouillée ou non.
- La seconde est qu'il nous permet d'afficher une barre de chargement qui nous indique là où en est le processus d'impression.

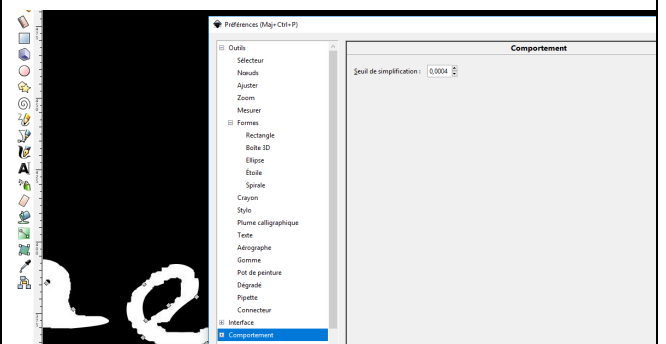
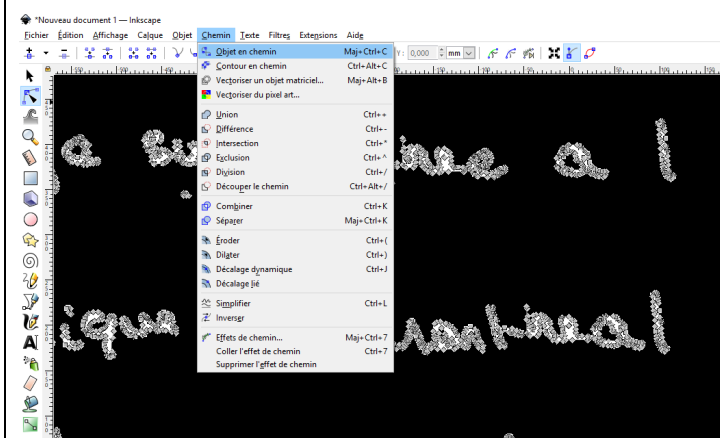
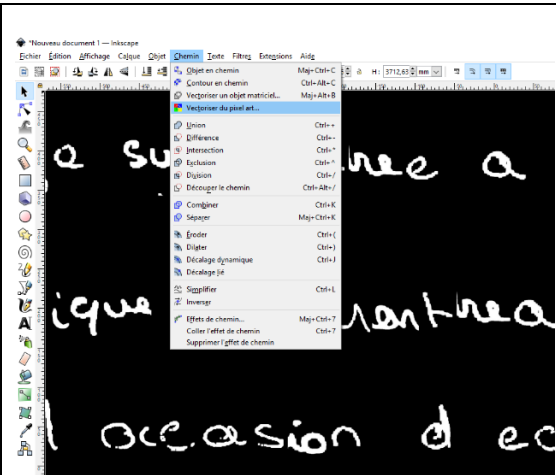
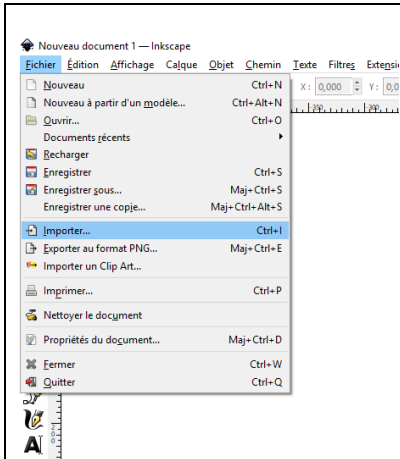


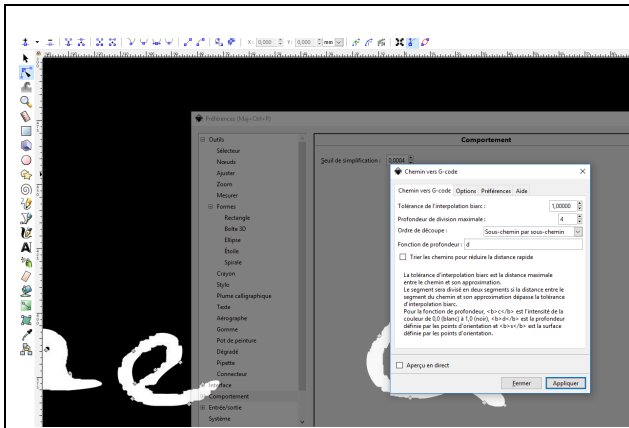
### III.7 Transformation de l'image en chemins

Notre carte Arduino ne va bien évidemment pouvoir recevoir une image et la recopier en toute simplicité. Le mieux est donc de transformer l'image en suite de point, ces points reliés entre eux formeront des lettres, et la carte Arduino n'aura plus qu'à tracer et rejoindre les points de manière successif. Afin de transformer l'image en points (*ou plus précisément chemins*), nous utilisons un logiciel spécialisé nommé Inkscape.

Celui-ci va prendre en entrée une image et va donner en sortie un fichier texte avec les différentes instructions à suivre pour pouvoir le tracer.

*Voici une petite diapo montrant les différentes étapes à suivre pour pouvoir faire la transformation.*





```

1  %
2  (Header)
3  (Generated by gcodetools from Inkscape.)
4  (Using default header. To add your own header create file "header" in the output dir.)
5  M3
6  (Header end.)
7  G21 (All units in mm)
8
9  (Start cutting path id: path596)
10 (Change tool to Default tool)
11
12 G00 Z5.000000
13 G00 X26.192413 Y29.934170
14
15 G01 Z-0.125000 F100.0(Penstrate)
16 G02 X27.399657 Y29.769872 Z-0.125000 I0.544639 J-0.515552 F400.000000
17 G02 X27.569770 Y28.638380 Z-0.125000 I-1.529440 J-0.818771
18 G02 X25.889775 Y26.988442 Z-0.125000 I-2.148365 J0.415529
19 G02 X23.584825 Y27.179460 Z-0.125000 I-0.768387 J3.470131
20 G02 X21.336991 Y30.286751 Z-0.125000 I1.767207 J3.660431
21 G02 X22.068345 Y33.045560 Z-0.125000 I5.305331 J0.836623
22 G02 X26.419560 Y36.655048 Z-0.125000 I5.093628 J-3.114977
23 G02 X31.407589 Y35.443600 Z-0.125000 I0.892102 J-7.197504
24 G02 X34.793393 Y29.756169 Z-0.125000 I-4.447736 J-6.499320
25 G02 X33.079194 Y29.415200 Z-0.125000 I-9.097259 J-0.942864
26 G02 X26.064908 Y19.447890 Z-0.125000 I-7.893307 J5.771173
27 G02 X10.369650 Y21.070840 Z-0.125000 I-0.992681 J10.997697
28 G02 X13.817584 Y30.810573 Z-0.125000 I7.888947 J9.281451
29 G02 X16.550920 Y39.060730 Z-0.125000 I12.897632 J1.042487
30 G02 X26.217396 Y44.199495 Z-0.125000 I10.666330 J-8.403184
31 G02 X36.622765 Y40.953020 Z-0.125000 I1.092510 J-14.796875
32 G02 X42.349490 Y29.960789 Z-0.125000 I-9.715011 J-12.049188
33 G02 X38.580619 Y10.200030 Z-0.125000 I-16.695559 J-1.142700
34 G02 X26.278749 Y11.884478 Z-0.125000 I-13.430741 J11.025174
35 G02 X13.154474 Y16.160610 Z-0.125000 I-1.193288 J18.593788
36 G02 X6.249486 Y29.804044 Z-0.125000 I12.334138 J14.811385
37 G02 X11.041495 Y44.275910 Z-0.125000 I20.491671 J1.243990
38 G02 X26.010444 Y51.770795 Z-0.125000 I16.191392 J-13.642204
39 G02 X41.037941 Y46.462450 Z-0.125000 I1.294859 J-22.309223
40 G02 X43.446070 Y44.954051 Z-0.125000 I-14.752316 J-17.339111
41 G02 X44.901285 Y43.297640 Z-0.125000 I-16.472098 J-15.930750
42 G00 Z5.000000
43
44 (End cutting path id: path596)
45
46
47 (Footer)
48 M5
49 G00 X0.0000 Y0.0000
50
51 (Using default footer. To add your own footer create file "footer" in the output dir.)
52 (end)
53 %

```

## III.8 Transfert d'informations et contrôle de la carte

Comme expliqué précédemment, c'est notre ordinateur qui génère les images et qui s'occupe de la plupart des tâches informatiques nécessaires.

L'ordinateur doit donc pouvoir transférer les informations de l'image à la carte Arduino. Pour cela on utilise une communication simple mais efficace, celle du serial.

Après avoir récupéré le fichier Gcode nous le transférons donc à la carte Arduino ligne par ligne.

Chaque ligne envoyée est suivie d'un traitement logiciel qui récupère les coordonnées et le type de mouvement à faire. Quand la carte Arduino les a récupérés, elle s'occupe de contrôler les moteurs afin que le mouvement soit correctement effectué. Quand le mouvement est terminé, elle envoie un message de confirmation au pc, ce qui permet d'envoyer la commande suivante etc....

Le fichier est donc lu, envoyé et traité intégralement ligne par ligne jusqu'à sa fin.

# Conclusion

Notre projet avait pour objectif de pouvoir écrire de manière très précise, seulement notre imprimante entièrement bricolé et pleine d'imprécisions, le programme permettant de transformer l'image en Gcode ne le fait pas parfaitement et la carte Arduino n'est pas assez puissante ce qui amène beaucoup de latence et certains bugs.

Nous ne sommes donc pas en mesure de pouvoir écrire comme Raphael ni comme quiconque.

Cependant on peut faire des dessins assez simples comme des hippocampes ou des oiseaux.

# Bibliographie

<http://users.polytech.unice.fr/~pmasson/Enseignement.htm>  
<http://www.ppgia.pucpr.br/~alekoe/AM/2009/private/NISTDatabase.pdf>  
<http://www.wildml.com/2015/12/implementing-a-cnn-for-text-classification-in-tensorflow/>  
<https://arxiv.org/pdf/1702.05373.pdf>  
<https://catalog.data.gov/dataset/nist-handprinted-forms-and-characters-nist-special-database-19>  
<https://codeburst.io/optical-character-recognition-recognizing-text-to-labels-on-an-android-platform-4c20bddc9175>  
<https://fr.scribd.com/document/389830227/Generation-of-Personalized-Handwriting-Written-in-Different-Language-A-Technical-Review>  
<https://github.com/0x454447415244/HandwritingRecognitionSystem/blob/master/README.md>  
<https://github.com/bbornchr/gcodesender.py/blob/master/gcodesender.py>  
<https://github.com/Dexter2389/EMNIST/blob/master/EMNIST.py>  
<https://github.com/githubharald/WordSegmentation?files=1>  
<https://github.com/Grzego/handwriting-generation>  
<https://github.com/j05t/emnist>  
<https://github.com/j05t/emnist/blob/master/emnist.ipynb>  
[https://github.com/MakerBlock/TinyCNC-Sketches/blob/master/TinyCNC\\_Gcode](https://github.com/MakerBlock/TinyCNC-Sketches/blob/master/TinyCNC_Gcode)  
<https://github.com/shubhammor0403/EMNIST/blob/master/modeltrain.ipynb>  
<https://github.com/shubhammor0403/EMNIST/blob/master/README.md>  
<https://github.com/snowkylin/rnn-handwriting-generation>  
<https://github.com/vinaychetnani/EMNIST-letter-classification>  
<https://greydanus.github.io/2016/08/21/handwriting/>  
<https://medium.com/@erikhallstrm/hello-world-rnn-83cd7105b767>  
<https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148>  
<https://stackoverflow.com/questions/36503170/python-communication-to-arduino-via-bluetooth>  
<https://stackoverflow.com/questions/50815307/extract-character-images-from-cursive-continuous-handwritten-image>  
<https://towardsdatascience.com/tutorial-alphabet-recognition-deeplearning-opencv-97e697b8fb86>  
<https://www.cs.toronto.edu/~graves/handwriting.html>  
<https://www.kaggle.com/marcose18/cnn-on-emnist-dataset>  
<https://www.nist.gov/itl/iad/image-group/emnist-dataset>  
<https://www.tensorflow.org/tutorials/sequences/recurrent>  
<https://www.thingiverse.com/thing:2349232>

