



CREPP

Feutrage de la laine

Club de Robotique et d'Electronique
Programmable de Ploemeur

31 juillet 2022

Table des matières

	Page
Préambule	2
1 Introduction	4
1.1 Objectif	4
1.2 Les versions	4
1.3 État des lieux	4
1.4 Liste du matériel	4
1.5 Communication avec la CNC	4
2 Rappels du G-Code	6
2.1 Le référentiel	6
2.2 Les déplacements	6
2.3 Le retour à l'origine	7
3 Algorithme de déplacement	8
4 Programmation Python	10
4.1 Configuration	10
4.2 Bibliothèques requises	10
4.3 Un premier programme	10
4.4 Une interface graphique	13
Liste des figures	15

Préambule

Informations

- ▶ Document réalisé en \LaTeX par Nicolas Le Guerroué pour le Club de Robotique et d'Electronique Programmable de Ploemeur (CREPP)
- ▶ Permission vous est donnée de copier, distribuer et/ou modifier ce document sous quelque forme et de quelque manière que ce soit.
- ▶ Version du 31 juillet 2022
- ▶ Taille de police : 11pt
- ✉ nicolasleguerroue@gmail.com

Section 1

Introduction

1.1 Objectif

L'objectif est d'automatiser le feutrage de la laine, étape fastidieuse à la main.

On part du principe que si, sur une surface en laine (80%), on pose une petite quantité de laine pure, on pourra avec une aiguille spéciale enfoncer des brins de laine dans le support et ceux-ci y resteront à la remontée de l'aiguille (les brins de laine s'accrochent entre eux avec une facilité incroyable).

Pour une petite surface on utilise un support avec 6 à 9 aiguilles et tenir dans la main le dit support avec un aller et retour vertical devient fastidieux avec des risques de blessures.

C'est cette action qu'il faut automatiser.

1.2 Les versions

Une première version avec un simple balayage du plateau est en cours de constitution.

La seconde version devrait comporter un traitement d'image avec une caméra pour adapter le balayage.

1.3 État des lieux

Actuellement, des essais ont été réalisés avec la CNC qui était stocké au FabLab.

De nombreuses tentatives (parfois infructueuses) pour prendre en main le GCode ont donc été réalisées.

1.4 Liste du matériel

- ▶ Une ancienne CNC (modèle de Xavier Hinault)
- ▶ Support d'aiguilles (fait par Michel)

1.5 Communication avec la CNC

L'ensemble des communications ont lieu avec une liaison série (UART)

La liaison UART est une liaison série avec deux broches :

► RX

► TX

Protocole

- Un bit de start toujours à 0 pour synchroniser la communication
- Un champ de données de 8 bits
- Un bit de parité (dans notre cas aucune parité)
- Un bit de stop

Vitesse de communication

La liaison étant asynchrone, il faut que les périphériques communiquent à la même vitesse. Cette dernière est normalisée et représente le nombre de bit par seconde (baud ¹)

En l'occurrence, la CNC communique à 115200 bauds.

1. 1 baud représente 1 symbole par seconde.

Section 2

Rappels du G-Code

Le G-Code est un langage machine utilisé par les machines numériques. Chaque instruction G-Code se termine un retour à la ligne (valeur 13 dans la table ASCII). Par souci de lisibilité, il ne sera pas affiché.

2.1 Le référentiel

Soit on effectue les déplacements de manière relative (+xmm depuis la position actuelle) ou absolue (depuis l'origine avec les capteurs fin de course) Pour définir le référentiel, on utilise soit la commande suivante pour le référentiel absolu :

G90

Référentiel absolu

et pour le référentiel relatif :

G91

Référentiel relatif

Par la suite, on va utiliser le mode relatif.

2.2 Les déplacements

Pour se déplacer on va utiliser la commande suivante :

G01 X[mm] Y[mm] Z[mm]

La commande de déplacement

Les arguments entre crochets corresponde au déplacement voulu sur l'axe concerné. Ainsi, si on souhaite me déplacer de 20 mm vers le centre sur l'axe X, on va utiliser la commande :

G01 X20

Les déplacements

Les axes non concernés par le déplacement ne sont pas présents dans la commande.

2.3 Le retour à l'origine

Le retour en position d'origine se fait via la commande suivante :

```
G28 [X] [Y] [Z]
```

Retour à l'origine

Ainsi, pour effectuer un retour à l'origine sur l'axe X on fait la commande

```
G28 X
```

Retour à l'origine sur X

Section 3

Algorithme de déplacement

L'objectif est de ne pas faire de déplacement sur X ou Y lorsque le moteur Z est activé afin d'éviter de casser les aiguilles.

Au début du programme, le porte-aiguille est placé aux coordonnées (0,0) avec le porte-aiguille en position de repos (position haute).

Pendant toutes les durées des déplacements en X et Y, le porte-aiguille reste en position de repos.

- ▶ Le porte-aiguille se déplace en Y de la moitié de la largeur de la tête contenant les aiguilles (disons 8mm)
- ▶ Le cycle Z correspond à plusieurs aller-retour du porte-aiguille et à leur retour en position de départ (capteur de fin de course activé)

Ce cycle se reproduit pour balayer l'ensemble du plateau.

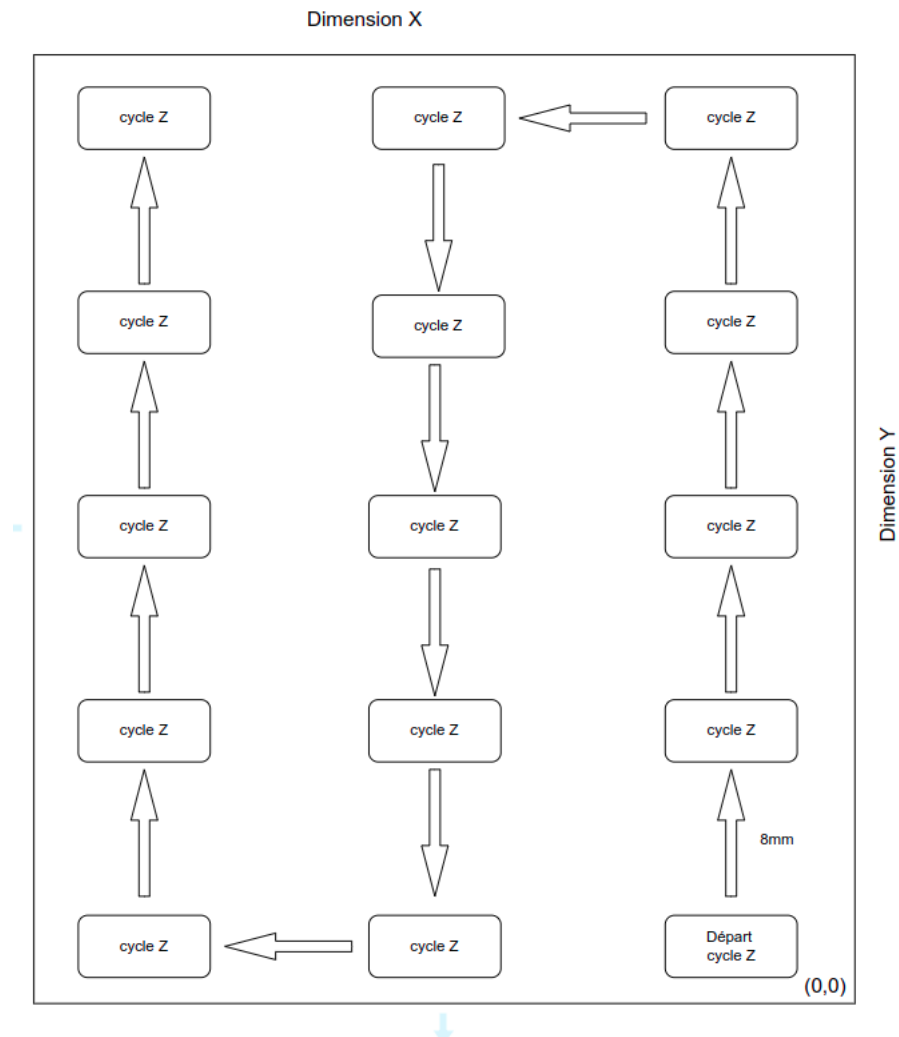


Figure 3.1 – Cycle de poinçonnage

Section 4

Programmation Python

4.1 Configuration

Le programme a été fait sous la version 2.7 de Python car une erreur de bibliothèque `LIB Serial` est survenue.

A moyen terme il faudra passer sous la version 3.X.

4.2 Bibliothèques requises

► Bibliothèque `LIB serial` :

```
sudo apt-get install python3-serial
```

Installation de la bibliothèque pyuic5

4.3 Un premier programme

Ce premier programme permet de vérifier le bon déplacement de la tête sur l'ensemble du plateau.

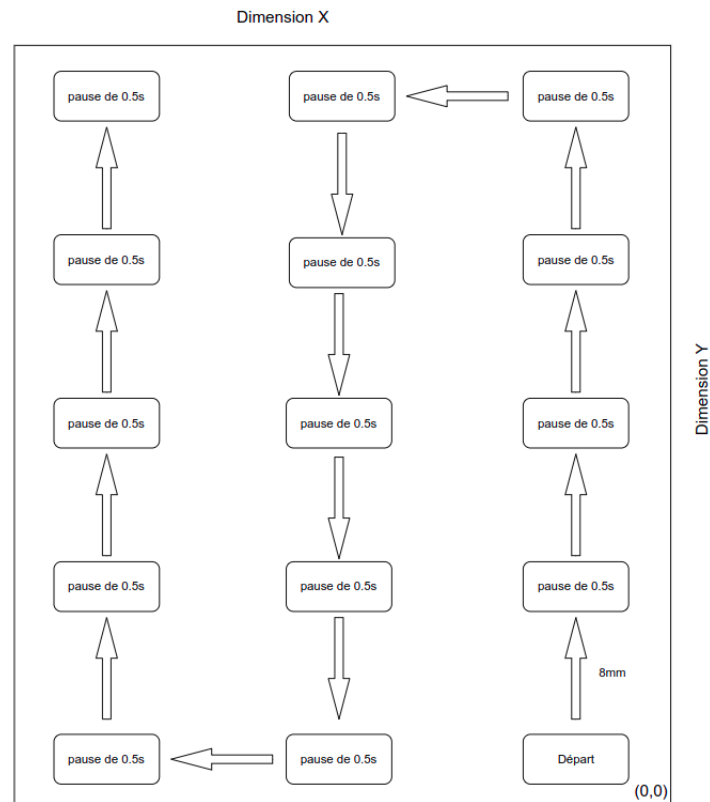


Figure 4.1 – Déplacement de la tête

Pour lancer le programme, il faut saisir la commande suivante :

```
python GCode.py
```

Lancement du programme

Le programme demande ensuite les dimensions du plateau (X et Y) et il va générer au fur et à mesure le G-Code pour balayer le plateau.

4.3.1 Code source

```
# -*- coding: utf-8 -*-

from math import floor
import serial
import time

ser = serial.Serial( '/dev/ttyACM0', 115200, serial.EIGHTBITS, serial.
    PARITY_NONE, serial.STOPBITS_ONE, timeout=0.1)
ser.flushInput() #Vide le buffer

if(open):
    print("Communication fonctionnelle !")
else:
    print("Mauvaise communication")
```

```
decalagePas = 8 #moitie du porte outil en mm

yWidth = input("Quelle est la longueur du plateau ? (mm) : ")
xWidth = input("Quelle est la largeur du plateau ? (mm) : ")

yIter = floor(int(yWidth)/8)
xIter = floor(int(xWidth)/8)

nbBlocX = xIter/2
deplacementZ = 10 #mm
vitesse=4 #max 6

print("Nombre d'arret sur X : "+str(xIter))
print("Nombre d'arret sur Y : "+str(yIter))

def sendCommand(command):
    print("Commande : "+str(command))
    ser.write(command+'\n')
    result=""
    waitingOK = True
    while (waitingOK==True): # tant que au moins un caractere en reception
        char=ser.read() # on lit le caractere
        if char=='\n': # si saut de ligne, on sort du while
            result=""
        else: #tant que c'est pas le saut de ligne, on l'ajoute à la chaine
            result=result+char
            if(result=="<ok>"):
                print("Commande valide")
                waitingOK = False

#Init
print("Mode relatif")
sendCommand("G91") #Mode relatif

#Retour Home
print("Retour origine")
sendCommand("G28 X")
sendCommand("G28 Y")

#Generation du GCode
gcode = ""
for xBloc in range(0, int(nbBlocX)):
    #Code bloc

    for xStep in range(0,int(yIter)):
        gcode += "G01 Y"+str(decalagePas)+" F4"+"\\n"
        gcode += "G04 P0.1"+"\\n"

    gcode += "G01 X"+str(decalagePas)+" F"+str(vitesse)+"\\n"

    for xStep in range(0,int(yIter)):
        gcode += "G01 Y-"+str(decalagePas)+" F4"+"\\n"
        gcode += "G04 P0.1"+"\\n"
```

```
gcode += "G01 X"+str(decalagePas)+" F"+str(vitesse)+"\n"

print(gcode)

allCommands = gcode.split("\n")
print("Nombre d'instructions a executer :" +str(len(allCommands)))

for line in allCommands:
    sendCommand(line)
```

4.4 Une interface graphique

4.4.1 Bibliothèques requises

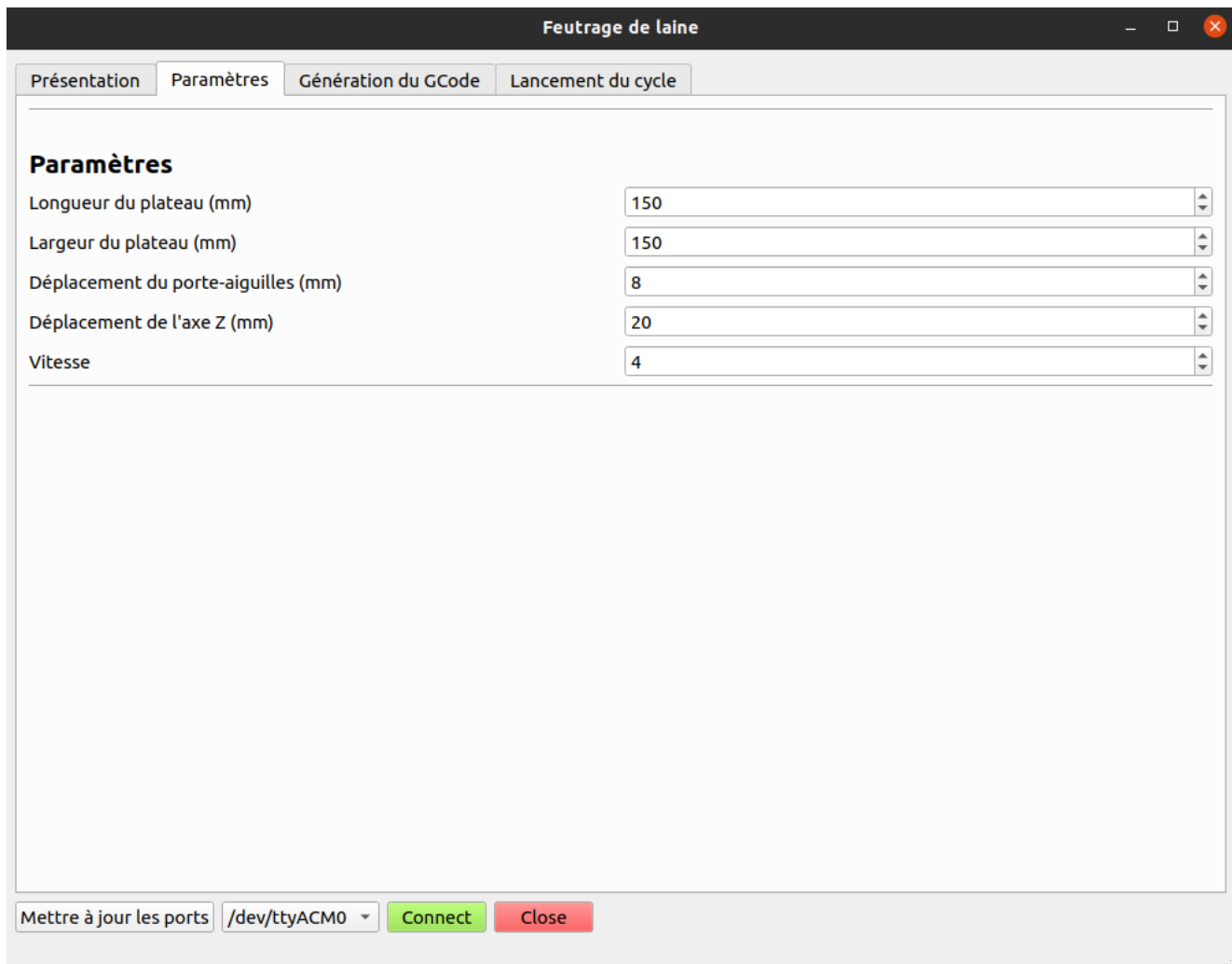
```
sudo apt-get install pyqt5-dev-tools
sudo apt-get install python3-pyqt5
sudo apt-get install python3-pyqt5.qtserialport
```

Installation des outils graphiques

Voici la première version d'une interface graphique réalisée avec PyQt5.

Le code est disponible sur le Git du club à

l'adresse [LINK https://github.com/CREPP-PLOEMEUR/Feutrage-de-laine](https://github.com/CREPP-PLOEMEUR/Feutrage-de-laine)



The screenshot shows a software window titled "Feutrage de laine" with a dark header bar. Below the header is a tabbed interface with four tabs: "Présentation", "Paramètres", "Génération du GCode", and "Lancement du cycle". The "Paramètres" tab is currently selected. The main area of the window is titled "Paramètres" and contains five input fields, each with a label and a numerical value in a text box with up/down arrow controls:

Paramètre	Valeur
Longueur du plateau (mm)	150
Largeur du plateau (mm)	150
Déplacement du porte-aiguilles (mm)	8
Déplacement de l'axe Z (mm)	20
Vitesse	4

At the bottom of the window, there is a status bar containing a button "Mettre à jour les ports", a dropdown menu showing "/dev/ttyACM0", a green "Connect" button, and a red "Close" button.

Figure 4.2 – Paramètres de l'interface

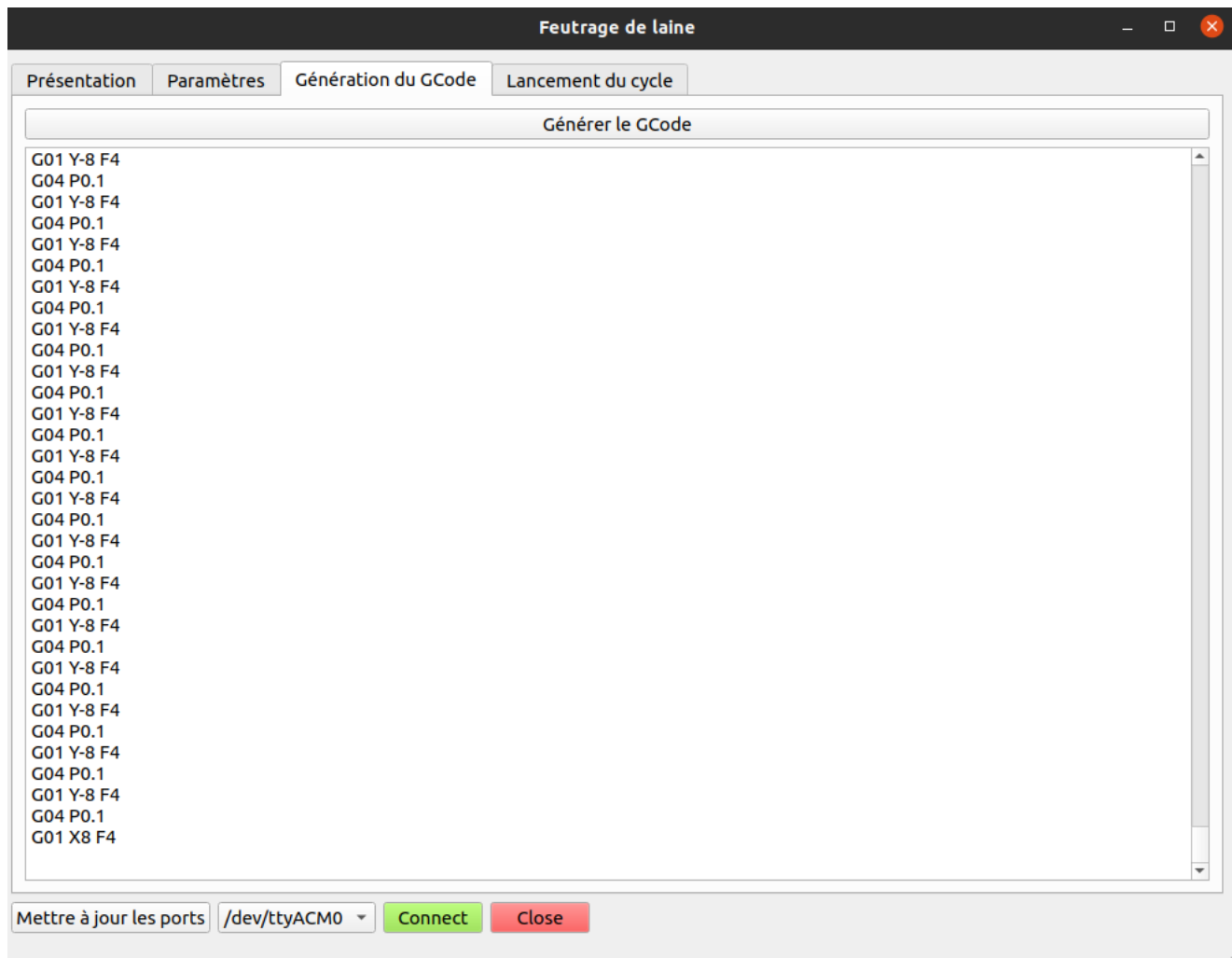


Figure 4.3 – Visualisation du GCode

L'interface est en cours de construction.

Liste des figures

3.1	Cycle de poinçonnage	9
4.1	Déplacement de la tête	11
4.2	Paramètres de l'interface	14
4.3	Visualisation du GCode	15