Olivier David Laplante

Technique Systèmes Ordinés

**Rapport du Projet d'Intégration en TSO TraceurDessinateur**

Travail présenté à :

[Francis Lance](mailto:francis.lance@cegepoutaouais.qc.ca)

Dans le cadre du cours

247-6E2-HU PIT

Groupe 201

Cégep de l’Outaouais

Campus Félix-Leclerc

Le 25 Avril 2023

# Table des matières

[**Table des matières 2**](#_94pemac5ctm6)

[**Introduction 3**](#_f9k5rufk86rd)

[But 3](#_fj5qmd57bkut)

[Caractéristiques 3](#_5v601zb6nm68)

[Faisabilité 3](#_8x8cet6dy565)

[Dans le temps 3](#_gclrjf3hw2u9)

[Économiquement 3](#_63vxi6yv4agr)

[Technologiquement 3](#_kojf2qag8o8h)

[**Diagrammes 5**](#_rujsukebx6s3)

[Fonctionnel 5](#_qmp54dmpyeq0)

[Schématiques 6](#_zrzgjek79v)

[Matriciel 8](#_rv3hl2yl0wm4)

[Evaluation des coûts 9](#_vmle9cgey75e)

[**Programmatique 10**](#_7qvgzjdifhez)

[Organigrammes 10](#_4rr93ktmta7k)

[Manette 10](#_hurh22ngm1p7)

[Serveur 11](#_32pvdmoudffr)

[Dessinateur 12](#_7rq3jw1173r8)

[Pseudo-code 13](#_lb2at4enzi0u)

[Manette 13](#_4l2d7vlan2ep)

[Serveur 14](#_lobwww52jpxt)

[Dessinateur 15](#_c506n7cgiuhy)

[Code 16](#_7i6qps7v8z95)

[Dessinateur.ino 16](#_ry3qgckd5njv)

[StepperWrapper.h 22](#_bc55a0q6wlmo)

[StepperWrapper.cpp 26](#_iadxx1tyo0yo)

[Manette.ino 31](#_jgxl4t53qx8)

[MQTTBroker.ino 38](#_p6c2rzbg6p46)

[**Discussion 44**](#_xjuwmab1kmzs)

[Problèmes et Solutions 44](#_w1ay7qcrnms3)

[Tests 44](#_s11ffzfse4l3)

[Utilisations 44](#_unfio9y74h8m)

[Ameliorations 44](#_6lsuru5sgcmp)

[**Reproduction 45**](#_g315fzffpvbk)

[Matériel 45](#_94o5109q1z3z)

[Programmation 45](#_l30qo0bsu2oj)

[Environnement 45](#_sz6tmqoktoqd)

[Logiciels 45](#_8pd6tr6owshr)

[Installer les pilotes 46](#_v8woe7i3jd88)

[Procédé 46](#_i0btw3vpzxdf)

[Installer le logiciel du "Contrôleur USB" 46](#_b90afwt90ypm)

[Installer le logiciel du "Dessinateur" 46](#_3yeqdrh7mvgt)

[Installer le logiciel de la "Manette" 46](#_zbpjpqmuqhx0)

[Vérifications 46](#_2b6acaz8ynu6)

[Assemblage 47](#_uh39npw1byr5)

[Assurer le bon fonctionnement 47](#_5go5l0ifzk2n)

[**Conclusion 48**](#_merz20z3hyvc)

[**Annexe A 49**](#_f270l251sicy)

[**Annexe B 50**](#_thm6lydt39i)

[Manuel de l’utilisateur 50](#_3tdk8jdvozvo)

[Consignes de sécurité 50](#_gvvl177vk1i4)

[!Avertissements 50](#_av2oinv6417)

[!Attention 50](#_cfkw4eun6exf)

[Attention 50](#_cp0zfloszffi)

[Table des matières 50](#_n365srqjivm5)

[Caractéristiques 50](#_mo65nunrf6zq)

[Définitions 50](#_mqygrlrokwgg)

[Installation 51](#_tj4nqra7qo1d)

[Matérielle 51](#_f133oh92bg17)

[Mise en marche 51](#_c07bvr67llqd)

[Mise en arrêt 51](#_sz3iivhvlgn7)

[Utilisation 51](#_f8etzgxcr8he)

[Fonctionnement 51](#_99b0pb6zw1s6)

[Mode d'emploi 51](#_tsx35iuneyt9)

[Dépannage 52](#_qxi36fyl3fcv)

[Section avancée 53](#_mfqw3gfmqhzl)

[Prérequis 53](#_vathgg70y3rh)

[Accéder à la console de commandes 53](#_v6q5s8csl4n0)

[Les commandes 54](#_ilmvn4o9qnxj)

[Réinstaller les logiciels 54](#_nqu2tgfm4zi2)

[Installer les pilotes 54](#_s8md8gurhiq5)

[Installer le logiciel du "Contrôleur USB" 54](#_5acl81walxwz)

[Installer le logiciel du "Dessinateur" 55](#_9voadse7fne1)

[Installer le logiciel de la "Manette" 55](#_1zrspfftxbt)

[Dépannage avancé 55](#_2sh8ia1hcigb)

[**Annexe C 56**](#_ixahl8j21je6)

[Manuel de dépannage 56](#_1qwrs7k4kl0k)

[Problème 1 56](#_g4iatmnq1j8)

[Tests 56](#_4x4gerwoo1az)

[Hypothèses 56](#_137171cyik3)

[Meilleurs Tests 56](#_vk2ws6meuoyl)

[Modifications 56](#_hvodaq6iwdxy)

[Vérification 56](#_pdykjtcvpv1b)

[Problème 2 56](#_ncq6d6kft0yk)

[Tests 56](#_zadlxx1guv9q)

[Hypothèses 56](#_lb2br1n4cvxc)

[Meilleurs Tests 56](#_5bmvm1chlkz5)

[Modifications 56](#_a99go5dt4jbv)

[Vérification 56](#_xuf58pzetz2f)

[Problème 3 56](#_l0o95h5ci89b)

[Tests 56](#_h5h07ur8jq09)

[Hypothèses 56](#_o1n4dj2qnsrz)

[Meilleurs Tests 56](#_i5mw18p7rjn5)

[Modifications 56](#_irjpcwugdpfi)

[Vérification 56](#_2z7jkp9ylbq3)

# Introduction

## But

Le but était de concevoir et de réaliser un pinceau 3D connectée par wifi au dessinateur qui lui pourra dessiner sur une feuille de papier.

## Caractéristiques

## Faisabilité

### Dans le temps

Malgré que j'ai fini par manquer de temps pour achever le projet,

je le blâme purement sur la planification et sur les défis inutiles que je me suis placé pour rendre le projet plus intéressant. Si j'avais simplifié au lieu de complexifier et que mon échéancier était fondé sur mes capacités, je crois sincèrement que le temps n'aurait pas été un problème.

### Économiquement

À part la machine CNC qui m'a été prêté pour ce projet, toutes les autres composantes ne sont pas excessivement coûteuses (~100$ avec optimisations).

### Technologiquement

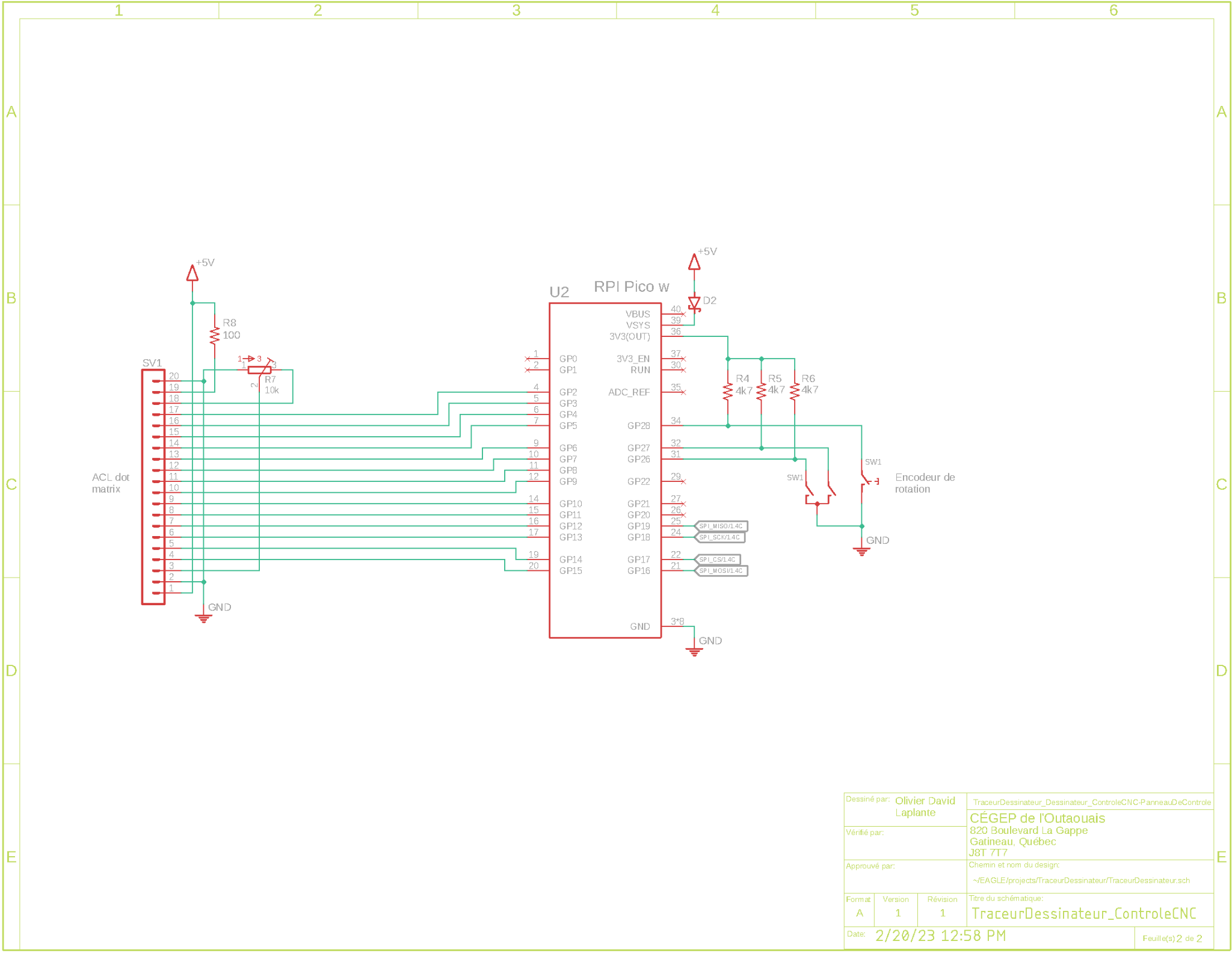
Les technologies utilisées dans ce projet sont très développées et accessibles pour tous. La composante la plus récente est le Raspberry Pi Pico W qui est sorti en juillet 2022.

# 

# Diagrammes

## Fonctionnel

## Schématiques



## Matriciel

# 

## Evaluation des coûts

# 

# Programmatique

## Organigrammes

### Manette

### Serveur

### Dessinateur

## 

## Pseudo-code

### Manette

### Serveur

### Dessinateur

## Code

### Dessinateur.ino

/\*\*

\* @file AllInOneDessinateur.ino

\* @brief Fichier principal du dessinateur

\*

\* Olivier David Laplante -- 8 Avril 2023 : Creation

\*/

//#include <wiring\_private.h>

#include "StepperWrapper.h"

#include <ArduinoJson.h>

#include <WiFi.h>

#include <PubSubClient.h>

// Definitions de constantes

#define NAME "Broker PIT"

#define PORT 1337

const IPAddress server\_ip(192, 168, 144, 1);

// Variables et instances globales

DynamicJsonDocument doc(256);

char message[256] = "";

WiFiClient wifi\_client;

PubSubClient client(wifi\_client);

// Steppers

StepperWrapper stepper\_x(0, 1, 2, 12, 12845 \* 4);

StepperWrapper stepper\_y(4, 5, 6, 13, 10752 \* 4);

StepperWrapper stepper\_z(8, 9, 10, 14, 1325 \* 4, true);

ThreeAxisStepper steppers(&stepper\_x, &stepper\_y, &stepper\_z);

bool paused = false;

// Variables pour les deltas

int deltaX = 0;

int deltaY = 0;

int deltaZ = 0;

/\*\*

\* @brief Fonction qui gère les messages admin

\*

\* @param topic : topic du message

\* @param payload : payload du message

\* @param lenght : longueur du payload

\*/

void adminMessage(char \*topic, uint8\_t \*payload, unsigned int lenght)

{

if (strcmp((char \*)payload, "home") == 0) // home : mise a zero de la position

{

paused = true;

steppers.home();

paused = false;

}

else if (strcmp((char \*)payload, "start") == 0) // start : demarrage des moteurs

{

steppers.enable();

}

else if (strcmp((char \*)payload, "stop") == 0) // stop : arret des moteurs

{

steppers.disable();

}

else if (strcmp((char \*)payload, "pause") == 0) // pause : pause les déplacements

{

paused = true;

}

else if (strcmp((char \*)payload, "resume") == 0) // resume : reprend les deplacements

{

paused = false;

}

else if (strcmp((char \*)payload, "restart") == 0) // restart : redémarre le RP2040

{

digitalWrite(LED\_BUILTIN, HIGH);

delay(1000);

digitalWrite(LED\_BUILTIN, LOW);

rp2040.reboot();

}

}

/\*\*

\* @brief Fonction qui fait clignoter la LED

\*

\*/

void vShortBlink()

{

digitalWrite(LED\_BUILTIN, HIGH);

delay(1);

digitalWrite(LED\_BUILTIN, LOW);

delay(1);

}

/\*\*

\* @brief Fonction qui gère les messages MQTT

\*

\* @param topic : topic du message

\* @param payload : payload du message

\* @param lenght : longueur du payload

\*/

void callback(char \*topic, uint8\_t \*payload, unsigned int lenght)

{

vShortBlink(); // blink led

// Si le message est un message admin

if (strcmp(topic, "admin") == 0)

{

adminMessage(topic, payload, lenght);

return;

}

if (paused) // Si le dessinateur est en pause on ne fait rien

{

return;

}

deserializeJson(doc, payload); // On désérialise le message

// On calcule les deltas

deltaY = (int)((short)doc["angle\_x"] / 2) - 63;

deltaX = (int)((short)doc["angle\_y"] / 2) - 63;

deltaZ = (int)((short)doc["bouton"] / 2) - 63;

client.flush();

}

/\*\*

\* @brief Fonction qui fait tourner les moteurs

\*

\*/

void runSteppers() {

unsigned long time = millis();

steppers.move(deltaX, deltaY, deltaZ);

delay((10 > (millis() - time)) ? (10 - (millis() - time)) : 0); // On attend jusqu'a 10ms pour avoir un delai constant

}

/\*\*

\* @brief Fonction qui reconnecte le client MQTT

\*

\*/

void reconnect()

{

// Boucle jusqu'à ce que le client se connecte

while (!client.connected())

{

Serial.print("Attempting MQTT connection...");

// essaye de se connecter

if (client.connect("arduinoClient"))

{

Serial.println("connected");

// Une fois connecté on s'abonne aux topics

client.subscribe("inTopic");

client.subscribe("admin");

}

else

{

Serial.print("failed, rc=");

Serial.print(client.state());

Serial.println(" try again in 5 seconds");

// On attend 5 secondes avant de recommencer

delay(5000);

}

}

}

/\*\*

\* @brief Fonction qui setup le dessinateur

\*

\*/

void setup()

{

Serial.begin(115200); // On initialise le port serie

pinMode(LED\_BUILTIN, OUTPUT); // On initialise la LED builtin

// On initialise le client MQTT

client.setServer(server\_ip, PORT);

client.setCallback(callback);

// On se connecte au WiFi

WiFi.mode(WIFI\_STA);

WiFi.begin(NAME);

// on active les moteurs

steppers.enable();

// On met les moteurs a zero

steppers.home();

}

void loop()

{

if (!client.connected()) // Si le client n'est pas connecte on se reconnecte

{

reconnect();

}

client.loop(); // On fait tourner le client MQTT

runSteppers(); // On fait tourner les moteurs

}

### StepperWrapper.h

/\*\*

\* @file StepperWrapper.h

\* @brief Les definitions pour manipuler des moteurs steppers

\*

\* Olivier David Laplante -- 8 Avril 2023 : Creation

\*/

#pragma once

/\*\*

\* @class StepperWrapper

\* @brief Objet pour simplifier un moteur stepper

\*

\*/

class StepperWrapper {

private:

int dir\_pin, step\_pin, en\_pin, home\_pin;

long max\_distance\_from\_origin;

bool positive\_direction;

bool direction;

/\*\*

\* @brief Initialise les pins relative au moteur

\*/

void initPins();

public:

long distance\_from\_origin;

/\*\*

\* @brief Le constructeur pour le stepper

\*

\* @param dir La pin de direction

\* @param step La pin de step

\* @param en La pin enable

\* @param home La pin de l'interrupteur de limite

\* @param max\_distance La distance maximale en pas

\* @param flip\_dir Change la direction par defaut (defaut = false)

\*/

StepperWrapper(int dir, int step, int en, int home, int max\_distance,

bool flip\_dir = false);

/\*\*

\* @brief Fait un pas

\*

\* @param force Force le pas meme si on est a la limite

\*/

void step(bool force);

/\*\*

\* @brief Change l'etat de la pin de direction

\*

\* @param dir La nouvelle direction

\*/

void setDir(bool dir);

/\*\*

\* @brief Change l'etat de la pin enable

\*

\* @param en L'etat

\*/

void setEnabled(bool en);

/\*\*

\* @brief Verifie si l'interrupteur est enfonce

\*

\* @return bool Si enfonce

\*/

bool isHome();

};

/\*\*

\* @class ThreeAxisStepper

\* @brief Manipule 3 moteurs stepper simultanement

\*

\*/

class ThreeAxisStepper {

private:

StepperWrapper \*stepper\_x, \*stepper\_y, \*stepper\_z;

/\*\*

\* @brief Utilise pour repartir egalement les pas des moteurs

\*

\* @param this\_step Le nombre de pas que le moteur a fait durant ce

\* deplacement

\* @param count Le nombre de pas de ce deplacement

\* @param steps\_moving Le nombre de pas requis pour ce moteur pour ce

\* deplacement

\* @param largest\_step\_count Le plus grand nombre de pas requis pour de

\* deplacement

\* @return Si le moteur fait un pas

\*/

bool doesStep(int this\_step, int count, int steps\_moving,

int largest\_step\_count);

public:

/\*\*

\* @brief Le constructeur

\*

\* @param stepperX Pointeur vers le StepperWrapper du moteur de l'axe X

\* @param stepperY Pointeur vers le StepperWrapper du moteur de l'axe Y

\* @param stepperZ Pointeur vers le StepperWrapper du moteur de l'axe Z

\*/

ThreeAxisStepper(StepperWrapper \*stepperX, StepperWrapper \*stepperY,

StepperWrapper \*stepperZ);

/\*\*

\* @brief Execute un deplacement

\*

\* @param deltaX Le nombre de pas pour l'axe X

\* @param deltaY Le nombre de pas pour l'axe Y

\* @param deltaZ Le nombre de pas pour l'axe Z

\*/

void move(int deltaX, int deltaY, int deltaZ);

/\*\*

\* @brief Remet a 0 les moteurs

\*/

void home();

/\*\*

\* @brief Fonctions pour activer et desactiver les moteurs

\*/

void disable();

void enable();

};

### StepperWrapper.cpp

/\*\*

\* @file StepperWrapper.cpp

\* @brief Le code pour manipuler des moteurs steppers

\*

\* Olivier David Laplante -- 8 Avril 2023 : Creation

\*/

#include "StepperWrapper.h"

#include <Arduino.h>

void StepperWrapper::initPins() {

pinMode(this->dir\_pin, OUTPUT);

pinMode(this->step\_pin, OUTPUT);

pinMode(this->en\_pin, OUTPUT);

pinMode(this->home\_pin, INPUT);

digitalWrite(this->dir\_pin, LOW);

digitalWrite(this->step\_pin, LOW);

digitalWrite(this->en\_pin, HIGH); // HIGH = disabled

}

StepperWrapper::StepperWrapper(int dir, int step, int en, int home,

int max\_distance, bool flip\_dir) {

this->dir\_pin = dir;

this->step\_pin = step;

this->en\_pin = en;

this->home\_pin = home;

this->distance\_from\_origin = 0;

this->max\_distance\_from\_origin = max\_distance;

this->positive\_direction = !flip\_dir;

this->direction = false;

this->initPins();

}

void StepperWrapper::step(bool force = false) {

// check if we are at the limit

// if (this->distance\_from\_origin >= this->max\_distance\_from\_origin && !force) {

// if (this->direction == this->positive\_direction) {

// delayMicroseconds(50);

// return;

// }

// } else if (this->distance\_from\_origin <= 0 && !force) {

// if (this->direction != this->positive\_direction) {

// delayMicroseconds(50);

// return;

// }

// }

// // keep track of distance from origin

// if (this->direction) {

// this->distance\_from\_origin++;

// } else {

// this->distance\_from\_origin--;

// }

digitalWrite(this->step\_pin, HIGH);

delayMicroseconds(25);

digitalWrite(this->step\_pin, LOW);

delayMicroseconds(25);

}

void StepperWrapper::setDir(bool dir) {

this->direction = dir;

digitalWrite(this->dir\_pin, this->positive\_direction == dir);

}

void StepperWrapper::setEnabled(bool en) {

digitalWrite(this->en\_pin, !en);

}

bool StepperWrapper::isHome() {

return digitalRead(this->home\_pin);

}

ThreeAxisStepper::ThreeAxisStepper(StepperWrapper \*stepperX,

StepperWrapper \*stepperY,

StepperWrapper \*stepperZ) {

this->stepper\_x = stepperX;

this->stepper\_y = stepperY;

this->stepper\_z = stepperZ;

}

bool ThreeAxisStepper::doesStep(int this\_step, int count, int steps, int largest\_step\_count) {

int num = (int)ceilf(count \* (float)((float)largest\_step\_count / (float)(steps + 1)));

return (num == this\_step && count <= steps);

}

void ThreeAxisStepper::move(int deltaX, int deltaY, int deltaZ) {

// On met les directions des moteurs

(\*this->stepper\_x).setDir(deltaX >= 0);

(\*this->stepper\_y).setDir(deltaY >= 0);

(\*this->stepper\_z).setDir(deltaZ >= 0);

// Attendre pour que les moteurs soient prêts

delayMicroseconds(10);

// Normaliser les pas

deltaX \*= (deltaX < 0) ? -1 : 1;

deltaY \*= (deltaY < 0) ? -1 : 1;

deltaZ \*= (deltaZ < 0) ? -1 : 1;

// Trouver le plus grand nombre de pas à faire

int largest\_step\_count = (deltaX > deltaY) ? deltaX : deltaY;

largest\_step\_count = (largest\_step\_count > deltaZ) ? largest\_step\_count : deltaZ;

// Initialiser les compteurs de pas pour chaque moteur

int count\_x = 1, count\_y = 1, count\_z = 1;

// Faire les pas un à un

for (int this\_step = 1; this\_step < largest\_step\_count + 1; this\_step++) {

if (doesStep(this\_step, count\_x, deltaX, largest\_step\_count)) { // Si on doit faire un pas pour ce moteur

(\*this->stepper\_x).step(); // Faire le pas

count\_x++; // Incrémenter le compteur de pas pour ce moteur (pour savoir quand on doit faire un pas)

} else { // Sinon, attendre pour compenser les pas manqués

delayMicroseconds(50);

}

if (doesStep(this\_step, count\_y, deltaY, largest\_step\_count)) { // Si on doit faire un pas pour ce moteur

(\*this->stepper\_y).step(); // Faire le pas

count\_y++; // Incrémenter le compteur de pas pour ce moteur (pour savoir quand on doit faire un pas)

} else { // Sinon, attendre pour compenser les pas manqués

delayMicroseconds(50);

}

if (doesStep(this\_step, count\_z, deltaZ, largest\_step\_count)) { // Si on doit faire un pas pour ce moteur

(\*this->stepper\_z).step(); // Faire le pas

count\_z++; // Incrémenter le compteur de pas pour ce moteur (pour savoir quand on doit faire un pas)

} else { // Sinon, attendre pour compenser les pas manqués

delayMicroseconds(50);

}

}

}

void ThreeAxisStepper::home() {

// un moteur à la fois

while (!(\*this->stepper\_x).isHome()) {

(\*this->stepper\_x).setDir(true);

(\*this->stepper\_x).step(true);

delayMicroseconds(90);

}

while (!(\*this->stepper\_y).isHome()) {

(\*this->stepper\_y).setDir(false);

(\*this->stepper\_y).step(true);

delayMicroseconds(90);

}

while (!(\*this->stepper\_z).isHome()) {

(\*this->stepper\_z).setDir(false);

(\*this->stepper\_z).step(true);

delayMicroseconds(90);

}

// remettre les distances à 0

(\*this->stepper\_x).distance\_from\_origin = 0;

(\*this->stepper\_y).distance\_from\_origin = 0;

(\*this->stepper\_z).distance\_from\_origin = 0;

}

void ThreeAxisStepper::disable() {

(\*this->stepper\_x).setEnabled(false);

(\*this->stepper\_y).setEnabled(false);

(\*this->stepper\_z).setEnabled(false);

}

void ThreeAxisStepper::enable() {

(\*this->stepper\_x).setEnabled(true);

(\*this->stepper\_y).setEnabled(true);

(\*this->stepper\_z).setEnabled(true);

}

### Manette.ino

/\*\*

\* @file Manette.ino

\* @brief Code Manette

\*

\* Olivier David Laplante - 11 Avril 2023 - Creation

\*/

#include <ArduinoJson.h>

#include <PubSubClient.h>

#include <MPU6050\_tockn.h>

#include <WiFi.h>

#include <Wire.h>

#include <freertos/FreeRTOS.h>

#include <freertos/task.h>

#include <esp\_task\_wdt.h>

// Configurations WiFi/Serveur/Capteurs

#define SSID "Broker PIT"

#define ADDR "192.168.144.1"

#define PORT 1337

#define PIN\_BUTTON 37

// Instantiation du Capteur/ClientWiFi/JSON

MPU6050 xManetteIMU(Wire);

WiFiClient wifi\_client;

PubSubClient client(wifi\_client);

DynamicJsonDocument doc(256);

char message[256] = "";

// Biais du IMU

float fOffsetX;

float fOffsetY;

/\*\*

\* @brief Assure la connexion WiFi et Serveur

\*/

void assertConnection()

{

while (WiFi.status() != WL\_CONNECTED)

{

Serial.print('.');

delay(500);

}

while (!client.connected()) {

connectToServer();

};

}

/\*\*

\* @brief Initialise la connexion au reseau WiFi

\*/

void initWiFi()

{

Serial.println("Connexion au controlleur...");

client.setServer(ADDR, PORT);

client.setCallback(callback);

WiFi.mode(WIFI\_STA);

WiFi.begin(SSID);

while (WiFi.status() != WL\_CONNECTED)

{

Serial.print('.');

delay(500);

}

Serial.println("OK");

Serial.println();

}

/\*\*

\* @brief Initialise la connexion au Serveur

\*/

void connectToServer()

{

Serial.println("Connexion au serveur...");

while (!client.connected()) {

Serial.print("Attempting MQTT connection...");

if (client.connect("arduinoClient")) {

Serial.println("connected");

client.subscribe("outTopic");

client.subscribe("admin");

} else {

Serial.print("failed, rc=");

Serial.print(client.state());

Serial.println(" try again in 5 seconds");

delay(5000);

}

}

Serial.println("OK");

}

/\*\*

\* @brief Calcule les biais du IMU

\*/

void recalibrateIMU()

{

float fCumulX = 0;

float fCumulY = 0;

for(int i = 0; i < 3000; i++) {

xManetteIMU.update();

fCumulX += xManetteIMU.getAccAngleX();

fCumulY += xManetteIMU.getAccAngleY();

}

fOffsetX = fCumulX / 3000;

fOffsetY = fCumulY / 3000;

}

/\*\*

\* @brief Initialise le IMU

\*/

void initIMU()

{

Wire.begin();

xManetteIMU.begin();

recalibrateIMU();

}

/\*\*

\* @brief Met a jour les donnees du IMU et du bouton

\*/

void updateDataIMU()

{

xManetteIMU.update();

float fAngleX = xManetteIMU.getAccAngleX() - fOffsetX;

float fAngleY = xManetteIMU.getAccAngleY() - fOffsetY;

int iButtonValue = analogRead(PIN\_BUTTON);

// Serial.println(round(fAngleX));

// Serial.println(round(fAngleY));

// Serial.println(iButtonValue);

/\*\*

\* @brief Formatte les donnees brutes en un nombre (short) de 0-255

\*/

short sAngleX = ((128 + round(fAngleX \* 2)) > 255) ? 255 : ((128 + round(fAngleX \* 2)) < 0) ? 0 : (128 + round(fAngleX \* 2));

short sAngleY = ((128 + round(fAngleY \* 2)) > 255) ? 255 : ((128 + round(fAngleY \* 2)) < 0) ? 0 : (128 + round(fAngleY \* 2));

short sButtonValue = floor(iButtonValue / 16);

// Serial.println(sAngleX);

// Serial.println(sAngleY);

// Serial.println(sButtonValue);

doc["angle\_x"] = sAngleX;

doc["angle\_y"] = sAngleY;

doc["bouton"] = sButtonValue;

serializeJson(doc, message, 256);

client.publish("inTopic", message);

client.flush();

}

/\*\*

\* @brief Callback pour le recalibrage du IMU sur un message MQTT admin

\*/

void callback(char \*topic, uint8\_t \*payload, unsigned int lenght)

{

// "admin" -> "recalibrate" -> recalibrateIMU()

if (strcmp(topic, "admin") == 0)

{

if (strcmp((char \*)payload, "gyrozero") == 0)

{

recalibrateIMU();

Serial.println("Recalibration du IMU");

} else if (strcmp((char \*)payload, "restart") == 0) {

delay(1000);

esp\_restart();

}

}

}

/\*\*

\* @brief Appelle les fonctions d'initialisation et demarre les taches

\*/

void setup()

{

Serial.begin(115200);

initWiFi();

connectToServer();

initIMU();

xTaskCreatePinnedToCore(vDataTask, "vDataTask", 10000, NULL, 2, NULL, 1);

xTaskCreatePinnedToCore(vMQTTTask, "vMQTTTask", 10000, NULL, 1, NULL, 1);

esp\_task\_wdt\_delete(NULL);

esp\_task\_wdt\_deinit();

}

/\*\*

\* @brief Appelle les fonctions de mise a jour du IMU et du bouton

\*/

void vDataTask(void \*pvParameters)

{

while (1)

{

updateDataIMU();

vTaskDelay(pdMS\_TO\_TICKS(100));

}

}

/\*\*

\* @brief Appelle les fonctions de connexion au serveur

\*/

void vMQTTTask(void \*pvParameters)

{

while (1)

{

if (!client.connected())

{

connectToServer();

}

client.loop();

vTaskDelay(30);

}

}

void loop() {

}

### MQTTBroker.ino

/\*\*

\* @file MQTTBroker.ino

\* @brief Programme principal du broker MQTT (Serveur)

\*

\* Olivier David Laplante - 20 Avril 2023 - Creation

\*/

// Fichiers d'inclusion

#include <WiFi.h>

#include <TinyMqtt.h>

#include <TinyConsole.h>

#include <ArduinoJson.h>

// configurations du broker

#define SSID "Broker PIT"

#define PORT 1337

#define RETAIN 10

IPAddress server\_ip(192, 168, 144, 11);

IPAddress gateway(192, 168, 144, 1);

IPAddress subnet(255, 255, 255, 0);

// broker et client

MqttBroker broker(PORT);

MqttClient adminClient(&broker);

// TinyConsole Console;

// variables pour le mode admin

bool bAdminLogging = false;

bool bAdminActive = false;

/\*\*

\* @brief Fonction qui affiche le logo du projet

\*/

void printBanner() {

// "TraceurDessinateur" en art ASCII

Console << " \_\_\_\_\_\_ " << endl;

Console << "(\_) | " << endl;

Console << " | ,\_ \_\_, \_\_ \_ ,\_ " << endl;

Console << " \_ |/ | / | / |/ | | / | " << endl;

Console << " (\_/ |\_/\\\_/|\_/\\\_\_\_/|\_\_/ \\\_/|\_/ |\_/ " << endl;

Console << endl;

Console << " \_\_\_\_ " << endl;

Console << " (| \\ o " << endl;

Console << " | | \_ , , \_ \_ \_\_, \_|\_ \_ ,\_ " << endl;

Console << " \_| ||/ / \\\_/ \\\_| / |/ | / | | |/ | | / | " << endl;

Console << "(/\\\_\_\_/ |\_\_/ \\/ \\/ |\_/ | |\_/\\\_/|\_/|\_/|\_\_/ \\\_/|\_/ |\_/" << endl;

Console << endl;

}

/\*\*

\* @brief Gere les commandes de l'admin

\*

\* @param command le texte de la commande

\*/

void consoleCallback(const std::string &command) {

if (!bAdminActive) { // Si le mode admin n'est pas active

if (command == "admin") { // Si la commande est "admin"

bAdminActive = true; // Active le mode admin

Console << "Mode admin active" << endl;

} else {

Console << "Commande inconnue" << endl;

}

Console.prompt();

return;

}

// Si le mode admin est active

if (command == "help") { // affiche l'aide

Console << endl << endl << endl;

printBanner();

Console << endl << endl;

Console << "Commandes:" << endl;

Console << "help: affiche l'aide" << endl;

Console << "log: affiche les messages du broker" << endl;

Console << "nolog: ne plus afficher les messages du broker" << endl;

Console << "gyrozero: recalibre la manette" << endl;

Console << "home: met a zero le dessinateur et le recentre" << endl;

Console << "stop: arrete le dessinateur" << endl;

Console << "start: demarre le dessinateur" << endl;

Console << "pause: met en pause le dessinateur" << endl;

Console << "resume: reprend le dessinateur" << endl;

Console << "restart: redemarre le serveur" << endl;

Console << "admin: entre dans le mode admin" << endl;

Console << "status: affiche le status des connexions" << endl;

Console << "exit: quitte le mode admin" << endl;

} else if (command == "gyrozero") { // recalibre la manette

adminClient.publish("admin", "gyrozero", RETAIN);

} else if (command == "home") { // recentre le dessinateur

adminClient.publish("admin", "home", RETAIN);

} else if (command == "stop") { // arrete le dessinateur

adminClient.publish("admin", "stop", RETAIN);

} else if (command == "start") { // demarre le dessinateur

adminClient.publish("admin", "start", RETAIN);

} else if (command == "pause") { // met en pause le dessinateur

adminClient.publish("admin", "pause", RETAIN);

} else if (command == "resume") { // reprend le dessinateur

adminClient.publish("admin", "resume", RETAIN);

} else if (command == "log") { // affiche les messages du broker

bAdminLogging = true;

} else if (command == "nolog") { // ne plus afficher les messages du broker

bAdminLogging = false;

} else if (command == "exit") { // quitte le mode admin

bAdminActive = false;

Console << "Mode admin desactive" << endl;

} else if (command == "restart") { // redemarre le serveur

adminClient.publish("admin", "restart", RETAIN);

delay(1000);

rp2040.reboot();

} else if (command == "status") { // affiche le status des connexions

Console << "Status des connexions:" << endl;

Console << " - Admin: " << (bAdminActive ? "active" : "desactive") << endl;

Console << " - Logging: " << (bAdminLogging ? "active" : "desactive") << endl;

Console << " - Clients: " << broker.clientsCount() << endl;

} else { // commande inconnue

Console << "Commande inconnue" << endl;

}

}

/\*\*

\* @brief Callback appele lorsqu'un message est recu

\*

\* @param client le client qui a recu le message

\* @param topic le topic sur lequel le message a ete recu

\* @param payload le message recu

\* @param lenght la taille du message

\*/

void logCallback(const MqttClient \*client, const Topic &topic, const char \*payload, unsigned int lenght) {

if (bAdminLogging == true) {

Console << topic.str() << ": " << payload << endl;

}

}

/\*\*

\* @brief Fonction d'initialisation

\*/

void setup() {

Serial.begin(115200);

pinMode(LED\_BUILTIN, OUTPUT);

Console.begin(Serial);

Console.setCallback(consoleCallback);

printBanner();

Console << endl << endl << endl;

Console << "MQTT Broker PIT" << endl << endl;

WiFi.mode(WIFI\_AP);

WiFi.softAPConfig(server\_ip, gateway, subnet);

WiFi.softAP(SSID);

Console << "Point d'acces WiFi configure" << endl;

Console << "SSID: " << SSID << endl;

Console << "IP: " << server\_ip.toString() << ":" << PORT << endl;

Console << "Gateway: " << gateway.toString() << endl;

Console << "Subnet mask: " << subnet.toString() << endl << endl;

// init du broker

broker.begin();

Console << "Broker initialise" << endl << endl;

// init du Callback du client admin

adminClient.setCallback(logCallback);

adminClient.subscribe("inTopic");

adminClient.subscribe("outTopic");

adminClient.subscribe("admin");

}

void loop() {

broker.loop();

adminClient.loop();

Console.loop();

// blink tres court

digitalWrite(LED\_BUILTIN, HIGH);

delay(1);

digitalWrite(LED\_BUILTIN, LOW);

delay(1);

}

# Discussion

## Problèmes et Solutions

## Tests

## Utilisations

## Ameliorations

# Reproduction

## Matériel

- Raspberry Pi Pico W (x2)

- Esp32 Pico Kit

- Joystick

- MPU-6050 GY-521

- Li-ion

- Chargeur Li-ion

- CNC 2.5 axes

- PSU d'ordinateur 12v/5v/3v3

- Connecteur 1x6 femelle (x3)

- Connecteur 1x2 femelle (x3)

- Connecteur 1x3 femelle

- Connecteur 1x20 femelle (x2)

- Breadboard medium (x2)

- Câble micro USB

- Schottky 1N5817

- Résistance 4k7 (x3)

## Programmation

### Environnement

#### Logiciels

- Visual Studio Code (+ extension Arduino) (aka VSCode)

- Arduino IDE

- Git

- GitFiend (optionnel)

- Python

- Putty

- EagleCAD

- InstalleurTD (https://github.com/Skkeye/TraceurDessinateur/releases/download/v0.1.0-alpha/InstalleurTD.zip)

- Pilotes WinUSB

- Scripts d'installation

- 7zip (https://www.7-zip.org)

- Pilotes CP210x (https://www.silabs.com/documents/public/software/CP210x\_Universal\_Windows\_Driver.zip)

- PuTTY (<https://the.earth.li/~sgtatham/putty/latest/w64/putty-64bit-0.78-installer.msi>)

#### Installer les pilotes

1. WinUSB avec Zadig (un pico doit etre connecte durant l'installation)

2. CP210x

### Procédé

1. Avant tout, installer les logiciels ci-dessus

2. Avec Git/GitFiend, faites un clone de mon Git repo : https://github.com/Skkeye/TraceurDessinateur.git

cmd>`git clone https://github.com/Skkeye/TraceurDessinateur.git`

3. Ouvrez le dossier avec VSCode

4. Naviguez dans `Code/Test/Sanity/`

5. Utilisez l'extension Arduino pour configurer l'environnement `Ctrl+Shift+p` `Arduino: Initialize`

- S'il y a une erreur, utilisez Arduino IDE pour comme environnement de programmation

6. Procédés de programmation :

\_\_\_

#### Installer le logiciel du "Contrôleur USB"

1. Prendre le "Contrôleur USB" et le brancher dans l'ordinateur en gardant le bouton blanc appuyé

2. Exécutez "Installer\_Controleur.bat" et attendre que la fenêtre se referme

3. Retirez le "Contrôleur USB" de l'ordinateur

#### Installer le logiciel du "Dessinateur"

1. Prendre le "Dessinateur" et le brancher dans l'ordinateur en gardant le bouton blanc appuyé

2. Exécutez "Installer\_Dessinateur.bat" et attendre que la fenêtre se referme

3. Retirez le "Dessinateur" de l'ordinateur

#### Installer le logiciel de la "Manette"

1. Prendre la "Manette" et le brancher dans l'ordinateur en gardant le bouton étiqueté "Boot" appuyé

2. Exécutez "Installer\_Manette.bat", vous pouvez relâcher le bouton une fois connecte

3. Une fois la fenêtre refermée, vous pouvez retirer la "Manette" de l'ordinateur

\_\_\_

## Vérifications

- Matériel

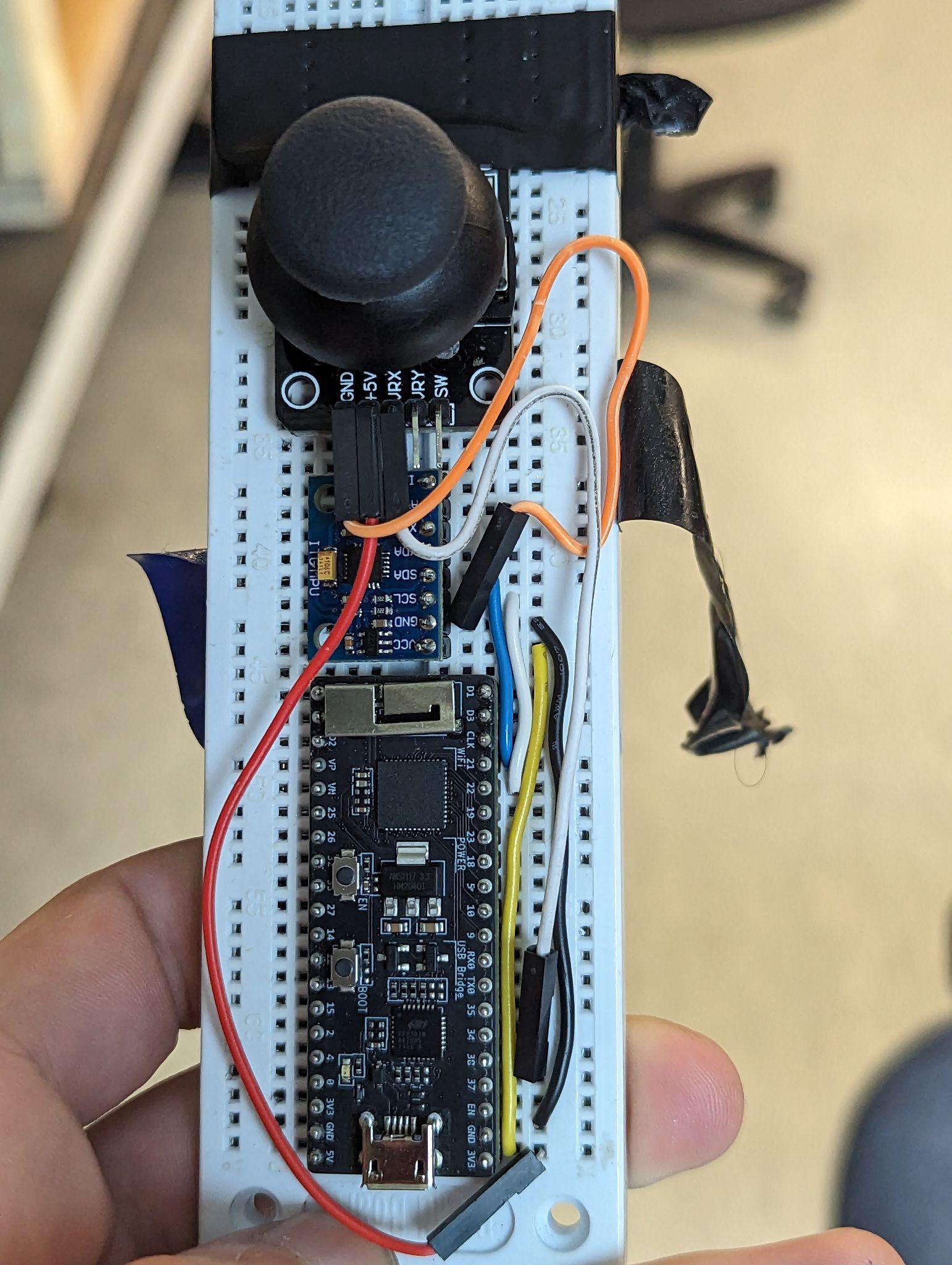
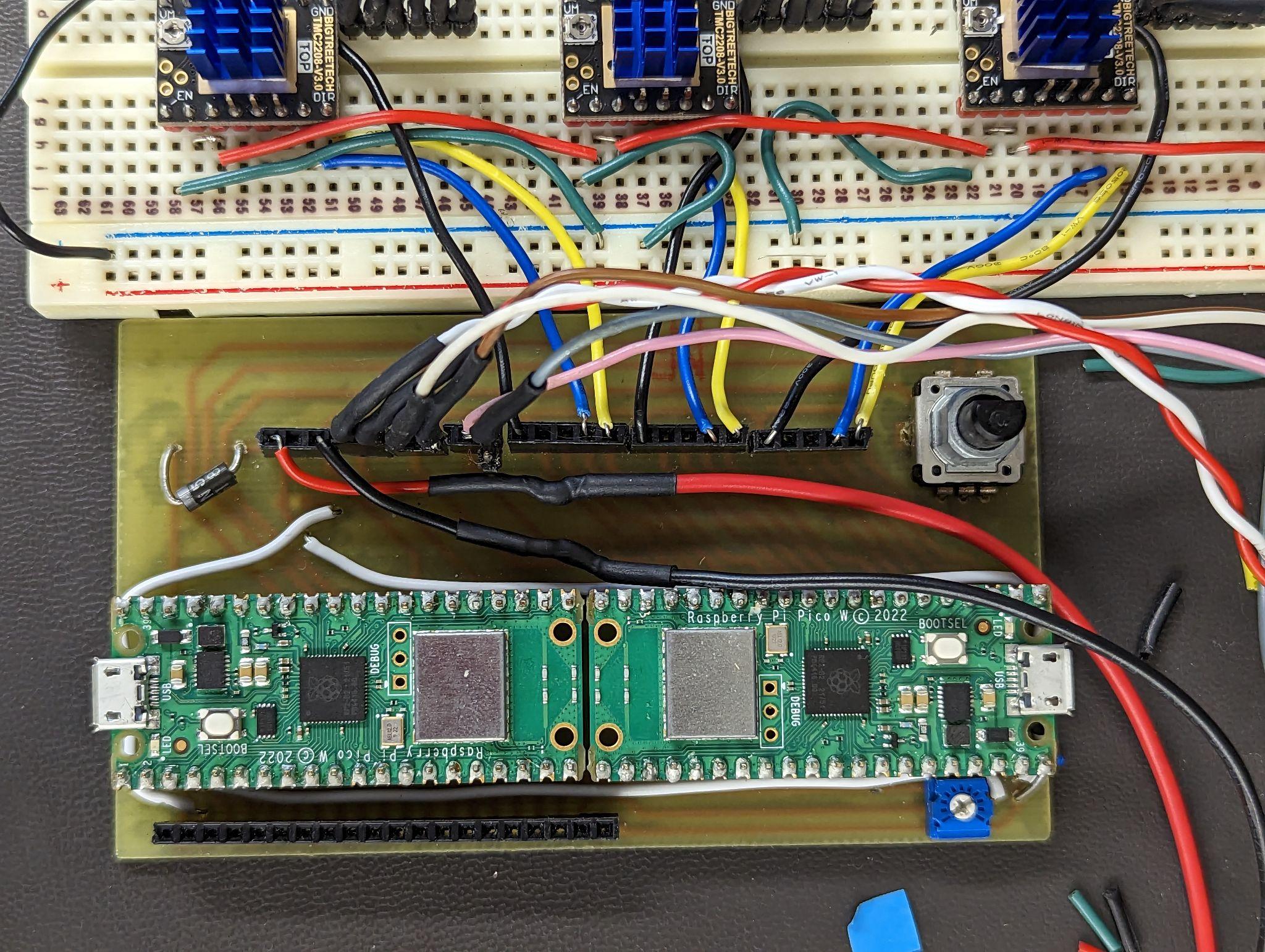
- Logiciel

## Assemblage

- Utilisez EAGLE pour générer les fichiers gerber depuis les fichiers brd

- Commandez les circuits imprimés avec les gerbers générés à la dernière étape

- Faire ces connections:



## Assurer le bon fonctionnement

- Lorsque le Dessinateur s'allume, il devrait se remettre à zéro

- Lorsque la DEL sur le Dessinateur fait une impulsion, cela signifie qu'il a reçu un message MQTT.

-

# Conclusion

# Annexe A

AllInOneDessinateur.ino : Code principal du dessinateur

StepperWrapper.h : Fichier d’inclusion du module de contrôle des steppers

StepperWrapper.cpp : Code du module de contrôle des steppers

…

# Annexe B

## Manuel de l’utilisateur

## **Consignes de sécurité**

### **!Avertissements**

* Ne pas ouvrir le bloc d'alimentation

### **!Attention**

* Ne pas être trop proche du "Dessinateur" lorsqu'il est en fonction
* Ne pas plier, couper, percer ou mettre un stress sur la batterie Lithium
* Ne pas déplacer manuellement la tête du "Dessinateur"

### **Attention**

* Ne pas laisser le "Dessinateur" en fonction pour une durée de plus de 2 heures

## **Table des matières**

## **Caractéristiques**

* Peut dessiner sur tout matériau qui n'a pas plus de 3 cm d'épaisseur et qui est compatible avec le crayon choisi
* Peut être contrôlé sur une distance de 0 à 20 mètres du "Contrôleur USB"

## **Définitions**

* Le "Dessinateur"
* Le "Câble d'alimentation"
* La "Manette"
* Le "Contrôleur USB"

## 

## 

## 

## **Installation**

### **Matérielle**

* Connectez le "Câble d'alimentation" dans la prise d'alimentation du "Dessinateur"
* Branchez le "Câble d'alimentation" dans une prise murale

### **Mise en marche**

* Branchez le "Contrôleur USB" dans une prise USB alimentée
* Allumez le "Dessinateur" avec l'interrupteur située proche de la prise d'alimentation
* Connectez la batterie à la "Manette"

### **Mise en arrêt**

* Déactivez l'interrupteur situé proche de la prise d'alimentation du "Dessinateur"
* Déconnectez la batterie de la "Manette"
* Débranchez le "Contrôleur USB"

## **Utilisation**

### **Fonctionnement**

Une fois le "Dessinateur" calibre, l'inclinaison de la "Manette" déplacera le crayon de gauche à droite et de proche a loin. Tirer ou pousser sur le joystick de la "Manette" descendera et remontera le crayon.

### **Mode d'emploi**

* Gardez la "Manette" à l'horizontale pour ne pas changer la position de la tête du "Dessinateur"
* Inclinez la "Manette" dans une direction pour contrôler la position de la tête du "Dessinateur"
* Poussez le joystick de la "Manette" pour abaisser le crayon sur la surface à dessiner
* Tirez le manche à balai de la "Manette" pour remonter le crayon
* En cas d'urgence, débranchez immédiatement le "Câble d'alimentation" de la prise murale et ensuite éteindre l'interrupteur du "Dessinateur" située proche de la prise d'alimentation

## **Dépannage**

* Si la "Manette" ne contrôle toujours pas le "Dessinateur" :
  + Éteindre tout
  + Branchez le "Controlleur USB" en premier
  + Ensuite, allumez le "Dessinateur"
  + Et finalement, allumez la "Manette"
  + Sinon se référer au Dépannage de la section avancée
* S'il y a un délai trop important :
  + Éteindre la "Manette"
  + Attendre 2 secondes
  + Allumer la "Manette"
  + Sinon tout redémarrer
  + Sinon se référer au Dépannage de la section avancée
* Si la "Manette" se déconnecte fréquemment :
  + Redémarrer la "Manette"
  + Sinon changer/recharger la batterie
  + Sinon se référer au Dépannage de la section avancée

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## **Section avancée**

### **Prérequis**

Extraire et installer ces logiciels

* PuTTY : <https://the.earth.li/~sgtatham/putty/latest/w64/putty-64bit-0.78-installer.msi>
* Pilotes CP210x : <https://www.silabs.com/documents/public/software/CP210x_Universal_Windows_Driver.zip>
* 7zip : [https://www.7-zip.org](https://www.7-zip.org/)
* InstalleurTD : <https://github.com/Skkeye/TraceurDessinateur/releases/download/v0.1.0-alpha/InstalleurTD.zip>

### **Accéder à la console de commandes**

* Ouvrez PuTTY
* Sélectionnez "Sériel" pour le type de connexion
* Entrez "115200" pour la Vitesse
* Pour trouver le port :
  + Naviguez dans InstalleurTD
  + Exécutez une première fois "Lister\_Ports.bat"
  + Branchez le "Controlleur USB" dans l'ordinateur
  + Exécutez une seconde fois "Lister\_Ports.bat"
  + Il devrait y avoir un port "COMx" de plus, c'est le port à choisir dans PuTTY
* Entrez le port trouve à l'étape 4 pour le port sériel
* Cliquez sur "Ouvrir"
* Entrez la commande "admin" suivie de la commande "help" pour voir si tout fonctionne

### 

### 

### 

### 

### 

### **Les commandes**

* "help" : Affiche les commandes disponibles
* "log" : Affiche les messages envoyés par la "Manette"
* "nolog" : Annule la commande "log"
* "gyrozero" : Recalibre la "Manette" ; garder la "Manette" immobile durant ce processus
* "home" : Faire la mise à zéro de la position du "Dessinateur"
* "stop" : Désamorcer les moteurs du "Dessinateur"
* "start" : Réamorcer les moteurs du "Dessinateur" ; il est recommandé d'exécuter la commande "home" à la suite d'un réamorçage
* "pause" : Immobilise le "Dessinateur"
* "resume" : Annule la commande "pause"
* "restart" : Redémarre tout le système
* "admin" : Entre dans le mode administrateur de la console de commandes : le mode administrateur permet l'accès aux commandes
* "status" : Affiche l'état du serveur
* "exit" : Quitte le mode administrateur de la console de commandes

### **Réinstaller les logiciels**

#### **Installer les pilotes**

* Naviguez dans le dossier InstalleurTD
* Installez les Pilotes USB :
  + Prendre le "Contrôleur USB" et le brancher dans l'ordinateur en gardant le bouton blanc appuyé
  + Exécutez "Installer\_Pilotes\_Zadig.exe"
  + Cliquez le bouton "Install Drivers" et attendre que les pilotes soient installés
  + Fermer la fenêtre
* Naviguez dans le dossier CP210x\_Universal\_Windows\_Driver
* Faites un clic droit sur "silabser.inf" et cliquez "Installer"

#### **Installer le logiciel du "Contrôleur USB"**

* Prendre le "Contrôleur USB" et le brancher dans l'ordinateur en gardant le bouton blanc appuyé
* Exécutez "Installer\_Controleur.bat" et attendre que la fenêtre se referme
* Retirez le "Contrôleur USB" de l'ordinateur

#### 

#### 

#### **Installer le logiciel du "Dessinateur"**

* Prendre le "Dessinateur" et le brancher dans l'ordinateur en gardant le bouton blanc appuyé
* Exécutez "Installer\_Dessinateur.bat" et attendre que la fenêtre se referme
* Retirez le "Dessinateur" de l'ordinateur

#### **Installer le logiciel de la "Manette"**

* Prendre la "Manette" et le brancher dans l'ordinateur en gardant le bouton étiqueté "Boot" appuyé
* Exécutez "Installer\_Manette.bat", vous pouvez relâcher le bouton une fois connecte
* Une fois la fenêtre refermée, vous pouvez retirer la "Manette" de l'ordinateur

### **Dépannage avancé**

* Si les procédés de dépannage standard ne règlent pas le problème :
  + Utiliser la commende "status" pour savoir si tous les clients sont connectés
  + Si un client n'est pas connecté, éteindre et redémarrez la "Manette" et le "Dessinateur"
  + Sinon, utilisez la commande "restart" pour redémarrer le système complètement
  + Si aucune des méthodes ne règle le problème, réinstallez les logiciels

# Annexe C

## Manuel de dépannage

## **Problème 1**

Les mouvements de la machine CNC étaient saccadés et inconsistants.

### **Tests**

J’ai vérifié ce que le Serveur recevait en imprimant sur le port sériel les informations envoyées par la manette.

J’ai envoyé des informations à partir du serveur pour contrôler directement la machine CNC.

### **Hypothèses**

Le Dessinateur laissais tomber des instructions si elles arrivaient durant un déplacement

Le réseau MQTT était surcharge dû à la quantité de messages

### **Meilleurs Tests**

En imprimant directement les messages reçus sur le Dessinateur j’aurai pu voir exactement ce qu’il recevait.

### **Modifications**

J’ai changé la façon dont le dessinateur exécutait les instructions, en passant de traiter un déplacement lorsqu’une instruction était reçue à faire des déplacements de manière consistante et modifier les déplacements lorsqu’une instruction est reçue.

J’ai aussi réduit la fréquence à laquelle la Manette envoyait des messages pour alléger la demande sur le serveur MQTT.

### **Vérification**

J’ai utilisé la Manette pour contrôler le Dessinateur en repassant les tests effectués.

J’ai constaté que le mouvement était suffisamment stable pour les standards du projet.

## 

## 

## 

## 

## **Problème 2**

La remise à zéro de la position du Dessinateur ne fonctionnait pas.

### **Tests**

J’ai vérifié les connexions des moteurs aux contrôleurs et de ceux-ci au microcontrôleur.

J’ai aussi utilisé un oscilloscope pour voir si les signaux étaient bons et bien synchronisés.

J’ai finalement mis en marche la fonction de remise a zero, et j'ai manuellement appuyer sur les interrupteurs de limites.

### **Hypothèses**

La fonction chargée d’orchestrer les pas pour la mise à zéro n'active pas correctement les conditions de limites pour arrêter le moteur.

Les calculs de protections de limites de fin (distance maximale) avaient des conditions non fonctionnelles.

### **Meilleurs Tests**

Lire les commentaires laissés pour indiquer une fonctionnalité qui reste à implémenter.

### **Modifications**

Ajouter la fonctionnalité dans la fonction de remise à zéro pour déplacer les moteurs jusqu'à leurs interrupteurs de limites et de mettre à zéro leur positions par la suite.

### **Vérification**

Exécuter la fonction avec la machine CNC et si nécessaire refaire les tests effectués

## 

## 

## 

## 

## 

## 

## 

## **Problème 3**

Un axe de la machine CNC était inversé.

### **Tests**

J’ai manuellement appuyer sur l’interrupteur de limite de l’axe inverse pour voir si ce n'était que durant la remise à zéro ou que c'était constant.

### **Hypothèses**

La direction par défaut n'était pas la bonne.

### **Meilleurs Tests**

Vérifier les connexions du dessinateur au contrôleur pour voir si le signal de direction se transmettait.

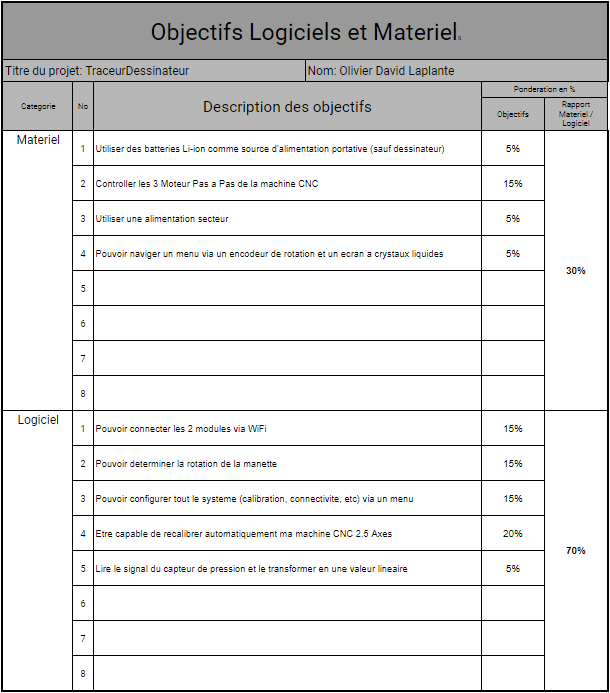
### **Modifications**

Reconnecter le fil du signal de direction correctement.

### **Vérification**

Mettre en marche le dessinateur et si nécessaire repasser les tests effectués.

## Liste des objectifs



## Gantt

## 

## Journal de Bord

| **Journal de bord : Projet d'intégration en TSO** | | | | **Jour de la semaine** | ***Description du travail accompli durant la semaine*** | ***Prolèmes rencontrés*** | ***Solution au problème*** |
| --- | --- | --- | --- | --- | --- | --- | --- |
|
| **Nom de l'étudiant: Olivier David Laplante** | | | |
|
| **Semaine 1** | | | | **Lundi** | Recherche d'information sur les composants pour le plan initial | J'ai realise que je n'avais pas suffisamment de broches pour l'ecrant ACL | J'ai ajoute un 2e RPi Pico W en communication SPI avec le premier |
| **Mardi** | Recherche d'information sur les composants pour le plan initial |  |  |
| **Mercredi** | Recherche d'information sur les composants pour le plan initial |  |  |
| **Jeudi** | Recherche d'information sur les composants pour le plan initial | L'ecran TFT commande n'etait pas compatible avec les solutions de camera de mon projet | Acheter personnellement un ecrant HDMI pour la solution du Raspberry Pi |
| **Vendredi** | Recherche d'information sur les composants pour le plan initial  Creation de la liste de pieces au magasin TGE | Le TEENSY41 avec la camera choisi n'ont pas un temps de rafraichissement assez vite pour mon application | Remplacer le TEENSY41 par un Raspberry Pi Pico W et la camera par une camera Raspberry Pi |
| **Semaine 2** | | | | **Lundi** | Réalisation de la feuille de présentation |  |  |
| **Mardi** | Réalisation de la liste des objectifs |  |  |
| **Mercredi** | Réalisation de la liste des objectifs |  |  |
| **Jeudi** | Réalisation de l'échéancier |  |  |
| **Vendredi** | Réalisation de l'échéancier |  |  |
| **Semaine 3** | | | | **Lundi** | Réalisation du Schématique du Contrôleur CNC  Réalisation du Schématique de la Manette  Réalisation du Schématique de la Caméra |  |  |
| **Mardi** | Réalisation du Schématique de la Manette  Réalisation du Schématique de la Caméra  Réalisation du Matriciel de la Caméra |  |  |
| **Mercredi** | Réalisation du Schématique de la Manette  Réalisation du Schématique du CNC  Réalisation du Matriciel de la Caméra  Réalisation du Matriciel de la Manette |  |  |
| **Jeudi** |  |  |  |
| **Vendredi** |  |  |  |
| **Semaine 4** | | | | **Lundi** | Réalisation du Schématique de la Manette  Réalisation du Schématique de la Caméra  Réalisation du Matriciel de la Manette  Réalisation du Matriciel de la Caméra | Les connecteurs de Batteries n'etait pas les bons | J'ai change les footprints pour accommoder les nouveaux connecteurs |
| **Mardi** |  |  |  |
| **Mercredi** |  |  |  |
| **Jeudi** |  |  |  |
| **Vendredi** |  |  |  |
| **Semaine 5** | | | | **Lundi** | Réalisation du Schématique du Contrôleur CNC  Réalisation du Matriciel du Contrôleur CNC |  |  |
| **Mardi** | Réalisation du Matriciel du Contrôleur CNC | Les traces sur le Matriciel CNC n'etaient pas sur le bon cote | Mettre les traces sur le "BOTTOM" |
| **Mercredi** | Réalisation du schématique du Contrôleur CNC  Réalisation du Matriciel du Contrôleur CNC  Réalisation du Matriciel de la Caméra  Réalisation du Matriciel de la Manette  Perçage du PCB du CNC  Perçage du PCB de la Caméra  Perçage du PCB de la Manette | Les Matriciels n'etaient pas aux normes du departement TGE | Ajout de texte sur les Matriciels |
| **Jeudi** | Prendre connaissance de Doxygen pour documenter le code  Soudage du PCB du CNC  Soudage du PCB de la Manette  Soudage du PCB de la Caméra |  |  |
| **Vendredi** | Installation de Doxygen  Soudage du PCB du CNC  Soudage du PCB de la Manette  Soudage du PCB de la Caméra |  |  |
| **Semaine 6** | | | | **Lundi** | Filage de la machine CNC | Le filage de la machine CNC n'etait pas clair ni assurement fonctionnel | Refaire completement le filage de la machine CNC |
| **Mardi** | Montage test du CNC  Création du code prototype pour contrôler la machine CNC  Test du CNC  Développement de l'algorithme du déplacement linéaire pour CNC | J'ai un probleme ou la liste cree dans une fonction ne pourra pas etre retournee avec un pointeur | Passer un pointeur d'une liste deja cree dans les arguments pour modifier le contenu. |
| **Mercredi** | Développement de l'algorithme du déplacement linéaire pour CNC | L'algorithme ne produit pas les valeurs attendu et demande potentionellement trop de memoire | Refaire l'algorithme completement en python pour mieux le developper et le deverminer |
| **Jeudi** | Intégration de Doxygen au projet |  |  |
| **Vendredi** | Intégration de Doxygen au projet |  |  |
| **Semaine 7** | | | | **Lundi** |  |  |  |
| **Mardi** |  |  |  |
| **Mercredi** |  |  |  |
| **Jeudi** |  |  |  |
| **Vendredi** |  |  |  |
| **Semaine 8** | | | | **Lundi** |  |  |  |
| **Mardi** |  |  |  |
| **Mercredi** |  | Le module de la camera etait trop ambitieux | Faire des modifications a la liste des objectifs avec mon superviseur de projet pour rendre le projet realisable dans le temps restant |
| **Jeudi** |  |  |  |
| **Vendredi** |  |  |  |
| **Semaine 9** | | | | **Lundi** |  |  |  |
| **Mardi** |  |  |  |
| **Mercredi** |  |  |  |
| **Jeudi** |  |  |  |
| **Vendredi** |  |  |  |
| **Semaine 10** | | | | **Lundi** |  |  |  |
| **Mardi** |  |  |  |
| **Mercredi** |  |  |  |
| **Jeudi** |  |  |  |
| **Vendredi** | Création initiale du module StepperWrapper  Création du code test pour la manette |  |  |
| **Semaine 11** | | | | **Lundi** | Documentation du code du module StepperWrapper |  |  |
| **Mardi** | Création du code final de la Manette |  |  |
| **Mercredi** | Déverminage du module StepperWrapper | Le code du module StepperWrapper ne fonctionnais pas / ne donnais pas de bonnes valeurs | Trouver la source du probleme (Implementation de l'algorithme de deplacement lineraire) et reparer le code errone |
| **Jeudi** | Création du code final du contrôleur  Création du Serveur MQTT | La librairie de creation/lecture de packets ne fonctionne pas pour mon application | Transmettre tout via MQTT et la librairie "PubSubClient" en format json avec la librairie "ArduinoJson", ajouter un serveur Broker MQTT |
| **Vendredi** |  |  |  |
| **Semaine 12** | | | | **Lundi** | Déverminage code Manette  Ajout du menu dans une pseudo-console sur le Serveur MQTT | Les mise a jours d'informations venant de la manette n'etaient pas assez stable | Integrer FreeRTOS dans le code de la manette et ralentir la frequence d'envoie de mises a jour |
| **Mardi** | Test CNC  Déverminage du code du CNC  Déverminage du code de la manette | La mise a zero de la position du dessinateur bloquais ou ne marchais pas | Simplifier la methode de mise a zero en calibrant qu'un seul axe a la fois |
| **Mercredi** | Déverminage du code du CNC  Déverminage du code de la manette | Les longueurs des axes du CNC était 4 fois trop court  Les mises à jour de la manette étaient trop fréquentes  Les déplacements étaient trop saccadés  La mise à zéro des axes semblaient ne pas mettre à zéro les axes  Les fonctions de protection de déplacement ne fonctionnaient qu'a moitie | Multiplier par 4 la longueur des axes et changer le type de variable  Ralentir la fréquence d'envoi de mise à jour de la manette  Se déplacer a une fréquence constante, modifier la vitesse sur une mise à jour  Ajouter les lignes qui remettent à zéro les distances à la fonction de remise à zéro  Retirer les fonctions de protection de déplacement |
| **Jeudi** |  |  |  |
| **Vendredi** |  |  |  |