int stepL = 3; //Globale Variablen, Definieren der Pins, die mit den zwei Schrittmotor-Treiber-Modulen verbunden sind,

int stepR = 6;//jedes Schrittmotor-Treiber-Modul hat jeweils zwei Pins, einen StepPin und einen DirectionPin,

int dirL = 5;// damit wir wissen, welche Pins benutzt werden, um die Pololu zu steuern.

int dirR = 7;

int druckTaster = 9;

int STEPS\_PER\_ROTATION = 400;

void setup() { //Funktion wird einmal aufgerufen, um festzulegen, was benutzt werden soll und was damit passieren soll.

pinMode (stepL, OUTPUT); //Schrittmotor 1

pinMode (stepR, OUTPUT);// Schrittmotor 2

pinMode (dirL, OUTPUT); //Schrittmotor 1

pinMode (dirR, OUTPUT);// Schrittmotor 2

digitalWrite (dirL, HIGH);

digitalWrite (dirR, HIGH);

pinMode (druckTaster, INPUT);

}

void loop() { //Funktion wird die ganze Zeit abgerufen, ständige Wiederholung

if (digitalRead (druckTaster) == HIGH) {

if (metalSensor()) {

rotateWheels(10, true);

}

else {

rotateWheels(10, false);

}

}

//bei "true" fahren die Schrittmotoren nach vorne, bei "false" nach hinten,

// hier wird nur die Bewegung selbst festgelegt, dass sie sich überhaupt bewegen und Richtung

}

// macht 1 Schritt mit dem bestimmten Motor in angegebene Richtung "true" oder "false", je nachdem, was angegeben wurde

void stepMove (int directionPin, int stepPin, boolean forward) { //boolean steht für true oder false

digitalWrite(directionPin, forward);

digitalWrite (stepPin, HIGH);

delay(1);

digitalWrite (stepPin, LOW);

delay (1);

}

// sobald ein Signal kommt (onSignal) wird alles unterbrochen und die Funktion "interrupt" kann beginnen

//dreht beide Räder in die angebene Richtung und die Anzahl der Umdrehungen

void rotateWheels(int rotations, boolean forward) {

for (int i = 0; i < rotations \* STEPS\_PER\_ROTATION; i = i + 1)

{

stepMove(dirL, stepL, forward);

stepMove(dirR, stepR, forward);

}

}

void onSignal() { // Funktion zur Aktivierung des Mini-Tastschalters, d.h. sobald etwas auf dem Schalter abgelegt wird, soll "interrupt" beginnen

}

boolean metalSensor() {

boolean metalDetected = digitalRead (2);

return metalDetected; //gibt ein True zurück, wenn das Metall ist oder ein false, wenn es kein Metall ist

//attachInterrupt(digitalPinToInterrupt(2), onSignal, RISING);

}

void onSignal() {

Serial.println("detected in onSignal");

metalDetected = true;

}