Paula Grzebyk 21236

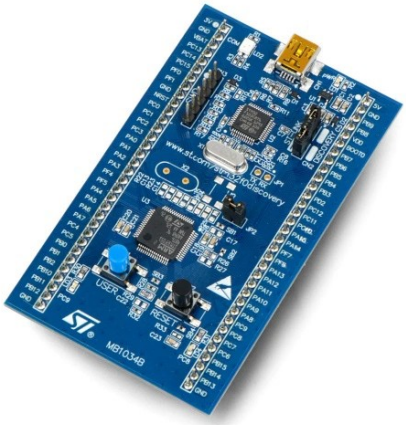
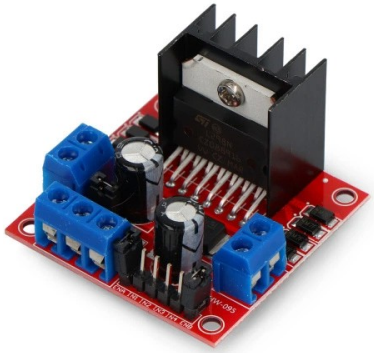
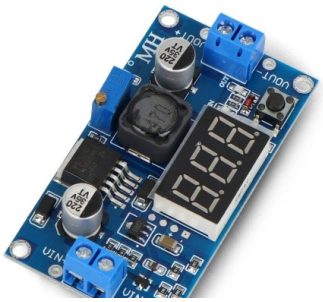
Sprawozdanie z milestone 3

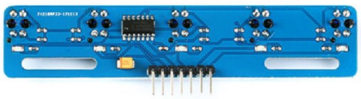
1. Opis robota

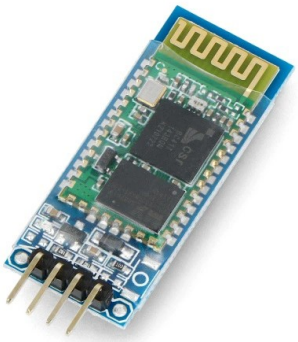
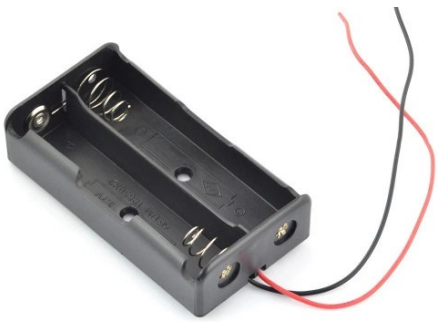
Robot trójkołowy o wymiarach ok. 24x20x15cm o napędzie kołowym za pomocą dwóch silników DC 5V, sterowany przez mikrokontroler STM32F051R8T6. Realizacja skrętu to obrót jednego koła według wskazówek zegara, a drugiego odwrotnie. Robot jest zasilany z koszyka na dwa ogniwa Li-On 18650 (łącznie 7,4V). Dodatkowo robot jest wyposażony w moduł Bluetooth HC-06 do komunikacji i sterowania z komputera za pomocą UART.

2. Elementy wybrane do budowy robota

- STM32F051R8T6
- L298N - dwukanałowy sterownik silników - moduł 12V/2A
- 2x Silnik DC 5V z oponą 65x26mm
- Kółko obrotowe
- Przetwornica step-down LM2596 3,2V-35V 3A z wyświetlaczem
- Moduł Bluetooth HC-06 ZS-040
- Koszyk na dwa ogniwa Li-On 18650
- Listwa 5 czujników odbiciowych
- Przewody

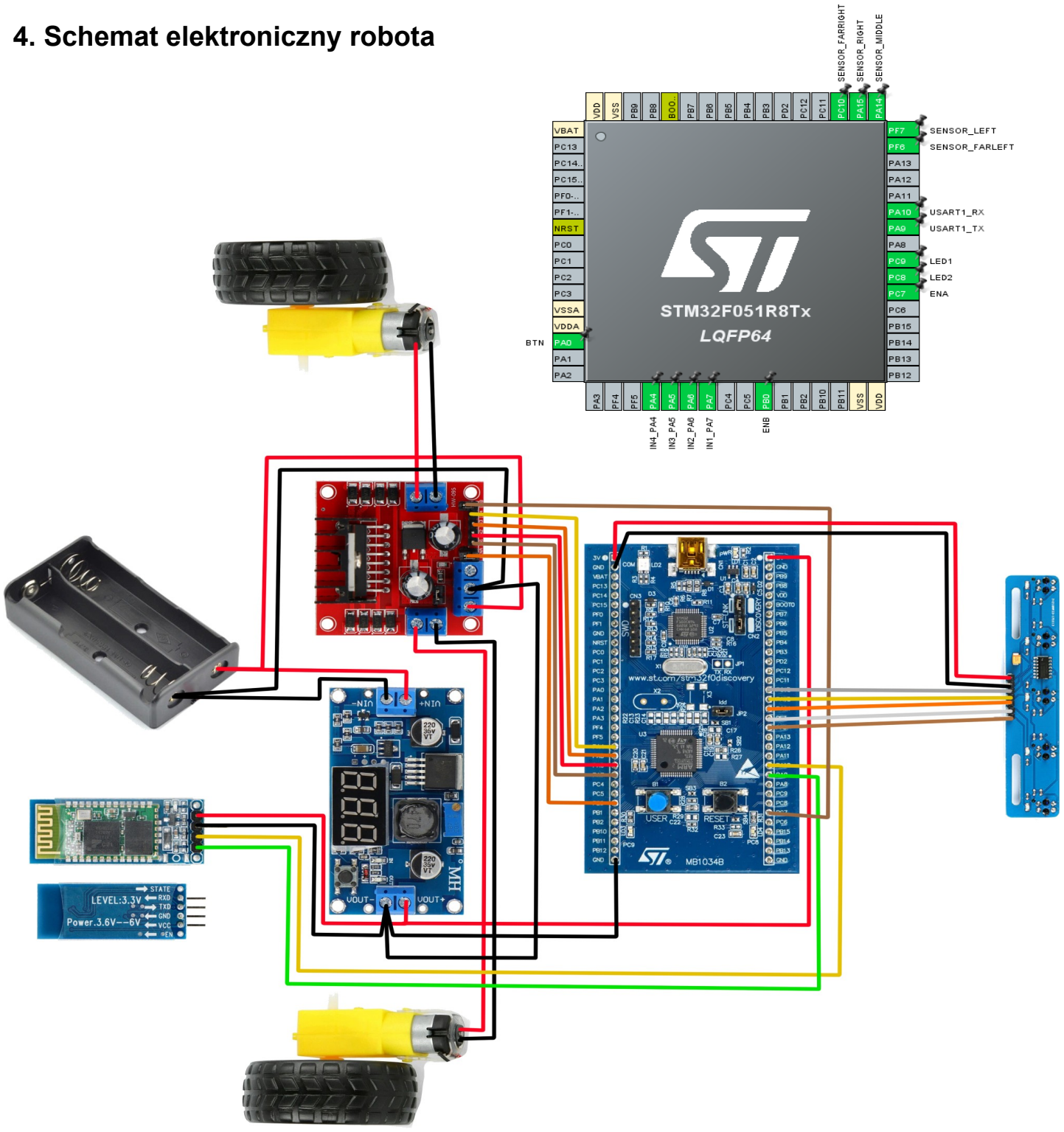



3. Mechanika robota

Podwozie zostało wykonane ze sklejki, na której umieszczono wszystkie elementy. Koła z silnikami umieszczono z tyłu a koło obrotowe pośrodku na przodzie pojazdu, i przyklejono gorącym klejem. Koszyk na akumulatory zamontowano z tyłu dla stabilizacji. L298N, przetwornicę, Bluetooth HC-06 oraz STM32F0 zamontowano przy użyciu gorącego kleju i taśmy dwustronnej. Czujniki IR zostały zamontowane na przodzie pojazdu w odstępach. Końcówki wybranych przewodów zostały cynowane aby zapobiec strzępieniu się żył miedzianych.

4. Schemat elektroniczny robota



5. Oprogramowanie sterujące

- Funkcje sterujące pojazdem – funkcje wysyłają odpowiednie sygnały do kół poprzez piny IN1-4 oraz ENA, ENB (PWM).

- (1,0) – koło kręci się według wskazówek zegara

- (0,1) – koło kręci się przeciwnie do wskazówek zegara

- (0,0) – koło nie kręci się

```
void Set_Speed(uint16_t speed){
    drivingspeed = speed; // zmienna ustalajaca predkosc (PWM)
    __HAL_TIM_SET_COMPARE(&htim3, TIM_CHANNEL_2, drivingspeed);
    __HAL_TIM_SET_COMPARE(&htim3, TIM_CHANNEL_3, drivingspeed);
}


void Go_Forward(void){
    Set_Speed(drivingspeed);
    HAL_GPIO_WritePin(IN1_PA7_GPIO_Port, IN1_PA7_Pin, 1); // oba kola do przodu
    HAL_GPIO_WritePin(IN2_PA6_GPIO_Port, IN2_PA6_Pin, 0);

    HAL_GPIO_WritePin(IN3_PA5_GPIO_Port, IN3_PA5_Pin, 1);
    HAL_GPIO_WritePin(IN4_PA4_GPIO_Port, IN4_PA4_Pin, 0);
}
void Go_Left(void){
    Set_Speed(drivingspeed);
    HAL_GPIO_WritePin(IN1_PA7_GPIO_Port, IN1_PA7_Pin, 1); // prawe do przodu
    HAL_GPIO_WritePin(IN2_PA6_GPIO_Port, IN2_PA6_Pin, 0);

    HAL_GPIO_WritePin(IN3_PA5_GPIO_Port, IN3_PA5_Pin, 0); // lewe do tylu
    HAL_GPIO_WritePin(IN4_PA4_GPIO_Port, IN4_PA4_Pin, 1);
}
void Go_Right(void){
    Set_Speed(drivingspeed);
    HAL_GPIO_WritePin(IN1_PA7_GPIO_Port, IN1_PA7_Pin, 0); // prawe do tylu
    HAL_GPIO_WritePin(IN2_PA6_GPIO_Port, IN2_PA6_Pin, 1);

    HAL_GPIO_WritePin(IN3_PA5_GPIO_Port, IN3_PA5_Pin, 1); // lewe do przodu
    HAL_GPIO_WritePin(IN4_PA4_GPIO_Port, IN4_PA4_Pin, 0);
}
void Go_Backward(void){
    Set_Speed(drivingspeed);
    HAL_GPIO_WritePin(IN1_PA7_GPIO_Port, IN1_PA7_Pin, 0); // oba kola do tylu
    HAL_GPIO_WritePin(IN2_PA6_GPIO_Port, IN2_PA6_Pin, 1);

    HAL_GPIO_WritePin(IN3_PA5_GPIO_Port, IN3_PA5_Pin, 0);
    HAL_GPIO_WritePin(IN4_PA4_GPIO_Port, IN4_PA4_Pin, 1);
}
void Stop(void){
    HAL_GPIO_WritePin(IN1_PA7_GPIO_Port, IN1_PA7_Pin, 0); // oba kola stop
    HAL_GPIO_WritePin(IN2_PA6_GPIO_Port, IN2_PA6_Pin, 0);
    __HAL_TIM_SET_COMPARE(&htim3, TIM_CHANNEL_2, 0);

    HAL_GPIO_WritePin(IN3_PA5_GPIO_Port, IN3_PA5_Pin, 0);
    HAL_GPIO_WritePin(IN4_PA4_GPIO_Port, IN4_PA4_Pin, 0);
    __HAL_TIM_SET_COMPARE(&htim3, TIM_CHANNEL_3, 0);
```

- **Milestone 1:** Zaprogramowana sekwencja jazdy – po wciśnięciu przycisku na mikrokontroletrze pojazd porusza się po sekwencji wykonanej z 10 rozkazów.

```
 while (1)
 {
     if(HAL_GPIO_ReadPin(BTN_GPIO_Port, BTN_Pin)==1){
         Drive_Sequence();
     }
void Drive_Sequence(void){
    HAL_GPIO_WritePin(LED2_GPIO_Port, LED2_Pin, 1);
    Set_Speed(100);
    HAL_Delay(2000);

    Go_Forward(); //1
    HAL_Delay(1000);
    Stop();
    HAL_Delay(500);

    Go_Left(); //2
    HAL_Delay(350);
    Stop();
    HAL_Delay(500);

    Go_Forward(); //3
    HAL_Delay(1000);
    Stop();
    HAL_Delay(500);
```

- **Milestone 2:** Po wciśnięciu przycisku program sprawdza, które czujniki wykrywają linię i odpowiednio skręca lub jedzie prosto.

```
while (1)
{
    // przycisk -> ustawia followline_flag = 1
    if(HAL_GPIO_ReadPin(BTN_GPIO_Port, BTN_Pin)==1){
        HAL_GPIO_WritePin(LED2_GPIO_Port, LED2_Pin, 1);
        HAL_Delay(2000);
        HAL_GPIO_WritePin(LED2_GPIO_Port, LED2_Pin, 0);
        followline_flag = 1;
    }
    // stan podazania za linia
    if(followline_flag == 1){
        HAL_Delay(80);
        Follow_Line();
        HAL_Delay(40);b
        Stop();
    }
```

```
void Follow_Line(void){
    //skret 90 w prawo
    if(HAL_GPIO_ReadPin(SENSOR_FARRIGHT_GPIO_Port, SENSOR_FARRIGHT_Pin)==0 && HAL_GPIO_ReadPin(SENSOR_RIGHT_GPIO_Port, SENSOR_RIGHT_Pin)==0){
        HAL_GPIO_WritePin(LED2_GPIO_Port, LED2_Pin, 1);
        HAL_UART_Transmit(&huart1, "FARRIGHT\r\n", 10, 100);
        Go_Left();
    }
    //skret 90 w lewo
    else if(HAL_GPIO_ReadPin(SENSOR_FARLEFT_GPIO_Port, SENSOR_FARLEFT_Pin)==0 && HAL_GPIO_ReadPin(SENSOR_LEFT_GPIO_Port, SENSOR_LEFT_Pin)==0){
        HAL_GPIO_WritePin(LED1_GPIO_Port, LED1_Pin, 1);
        HAL_UART_Transmit(&huart1, "FARLEFT\r\n", 9, 100);
        Go_Right();
    }
    //middle
    else if(HAL_GPIO_ReadPin(SENSOR_MIDDLE_GPIO_Port, SENSOR_MIDDLE_Pin)==0){
        HAL_GPIO_WritePin(LED1_GPIO_Port, LED1_Pin, 1);
        HAL_GPIO_WritePin(LED2_GPIO_Port, LED2_Pin, 1);
        HAL_UART_Transmit(&huart1, "MID\r\n", 6, 100);
        Go_Forward();
    }
    //right
    else if(HAL_GPIO_ReadPin(SENSOR_RIGHT_GPIO_Port, SENSOR_RIGHT_Pin)==0){
        HAL_GPIO_WritePin(LED2_GPIO_Port, LED2_Pin, 1);
        HAL_UART_Transmit(&huart1, "RIGHT\r\n", 7, 100);
        Go_Left();
    }
    //left
    else if(HAL_GPIO_ReadPin(SENSOR_LEFT_GPIO_Port, SENSOR_LEFT_Pin)==0){
        HAL_GPIO_WritePin(LED1_GPIO_Port, LED1_Pin, 1);
        HAL_UART_Transmit(&huart1, "LEFT\r\n", 6, 100);
        Go_Right();
    }
    else{
        HAL_UART_Transmit(&huart1, "NONE\r\n", 6, 100);
        Go_Forward();
    }
```

- Milestone 3: Przy użyciu modułu HC-06, pojazd można połączyć z komputerem przez Bluetooth i np. programem Putty wysyłać odpowiednie klawisze WSAD i zdalnie sterować.

W while(1): funkcje sterujące ruchem pojazdu wykonują się przez 100ms bez blokowania reszty programu.

```
// stan kontroli z bluetooth
if(action_flag != 0){
    if(action_flag == 1) Go_Forward();
    else if(action_flag == 2) Go_Backward();
    else if(action_flag == 3) Go_Left();
    else if(action_flag == 4) Go_Right();
    else if(action_flag == 5) Emergency_Stop();

    if(HAL_GetTick() > action_timeout){
        Stop();
        action_flag = 0;
    }
}
```

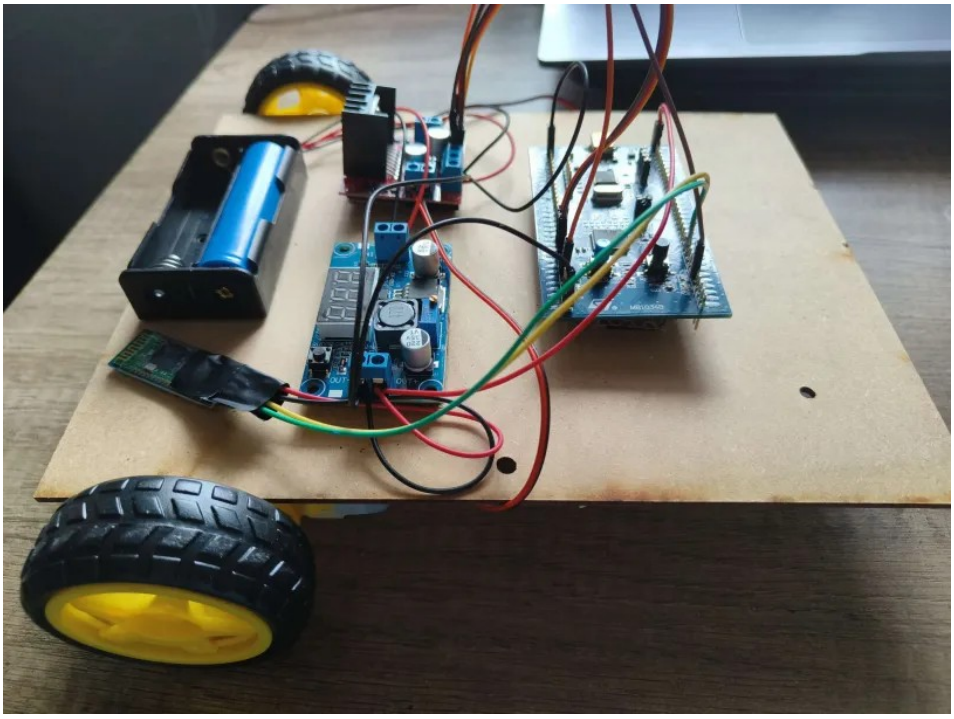
Komunikacja przez UART i przerwania, ustawia zmienną action_flag i w programie głównym w while(1) wykonuje odpowiednie funkcje.

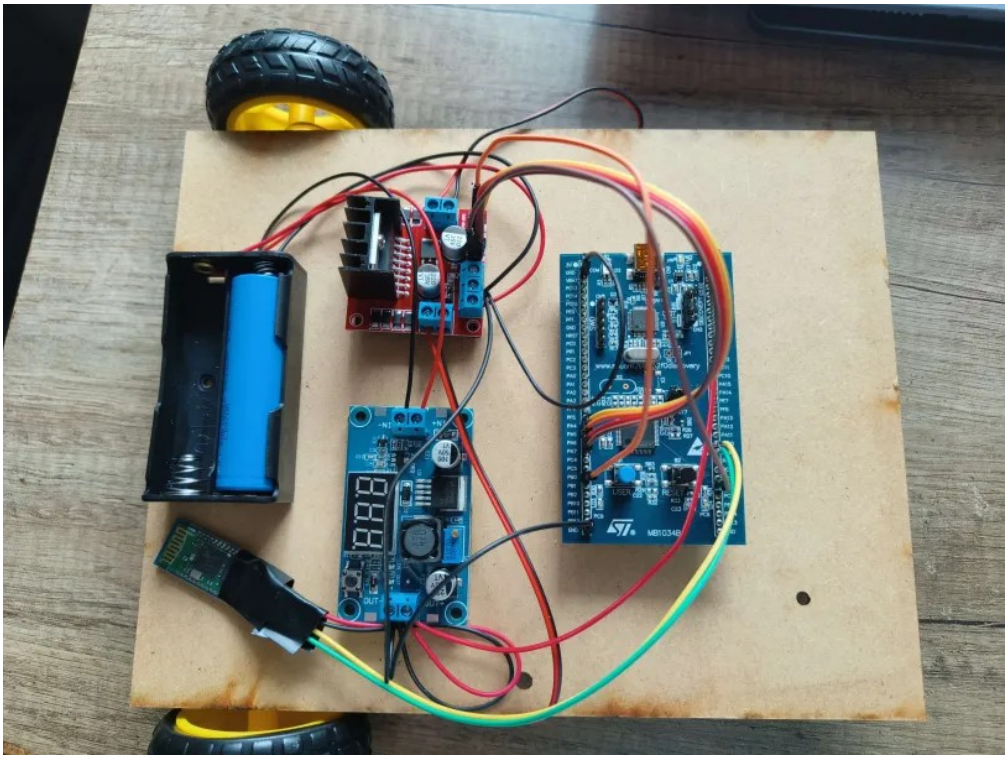
```
void HAL_UART_RxCpltCallback(UART_HandleTypeDef *huart) {
    if (huart->Instance == USART1){
        if(buffer[0] == 'w'){
            HAL_UART_Transmit(&huart1, "W\n", 2, 100);
            action_flag = 1;  // Go_Forward
            action_timeout = HAL_GetTick() + 100;
        }
        else if(buffer[0] == 's'){
            HAL_UART_Transmit(&huart1, "S\n", 2, 100);
            action_flag = 2;  // Go_Backward
            action_timeout = HAL_GetTick() + 100;
        }
        else if(buffer[0] == 'a'){
            HAL_UART_Transmit(&huart1, "A\n", 2, 100);
            action_flag = 3;  // Go_Left
            action_timeout = HAL_GetTick() + 100;
        }
        else if(buffer[0] == 'd'){
            HAL_UART_Transmit(&huart1, "D\n", 2, 100);
            action_flag = 4;  // Go_Right
            action_timeout = HAL_GetTick() + 100;
        }
        else if(buffer[0] == 'x'){
            HAL_UART_Transmit(&huart1, "X\n", 2, 100);
            action_flag = 5;  // Emergency_Stop
        }

        HAL_UART_Receive_IT(&huart1, buffer, 1);
    }
}
```

6. Zdjęcia, filmy, kod robota

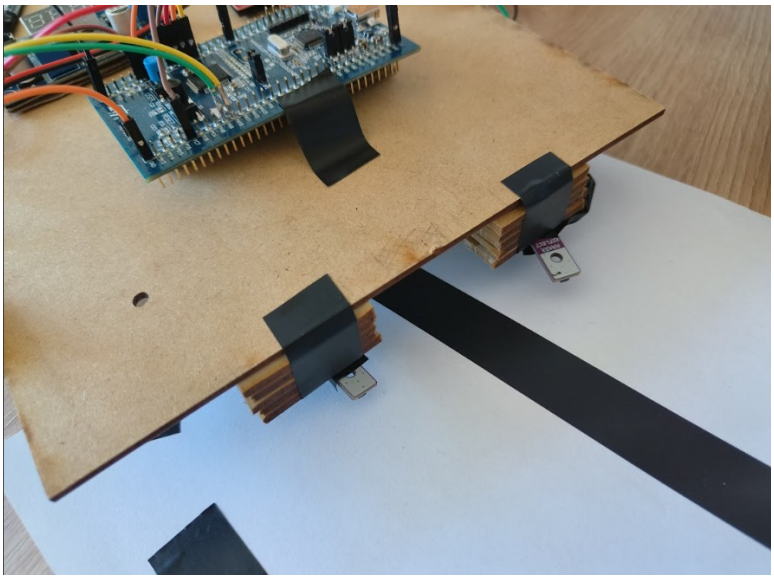
- Kod robota: https://github.com/BialaStrzala/SWIM_LineFollower
- Zdjęcia i filmy: https://drive.google.com/drive/folders/1Ak4zJQ64U5y1St-GSTiknMUM3oySruRG?usp=drive_link

- **Milestone 1:**





- Milestone 2:



- Milestone 3: