mikro rover Robot Kit

Introduction to KidMotor V4i controller board 01

1.1 Technical features

- Main microcontroller is RP2040 from Raspberry Pi Foundation
- POWER switch for turning on/off the power supply
- Requires +3.7V battery voltage source. The 18650 Lithium-Ion 3.7V 1000mAH is recommended.
- On-board +5V and +3.3V regulator circuit includes battery voltage monitor
- Upload program and charge the battery through USB-C port
- I2C bus connector with KB Chain (RH5) type and GROVE (PH4) standard
- DC motor driver circuit 2 channels for 3 to 6V 1A with LED direction status
- 3-servo motor output with +5V for motor supply voltage
- OLED 0.96" with 128x64 pixel resolution interfaced via I^2C bus.

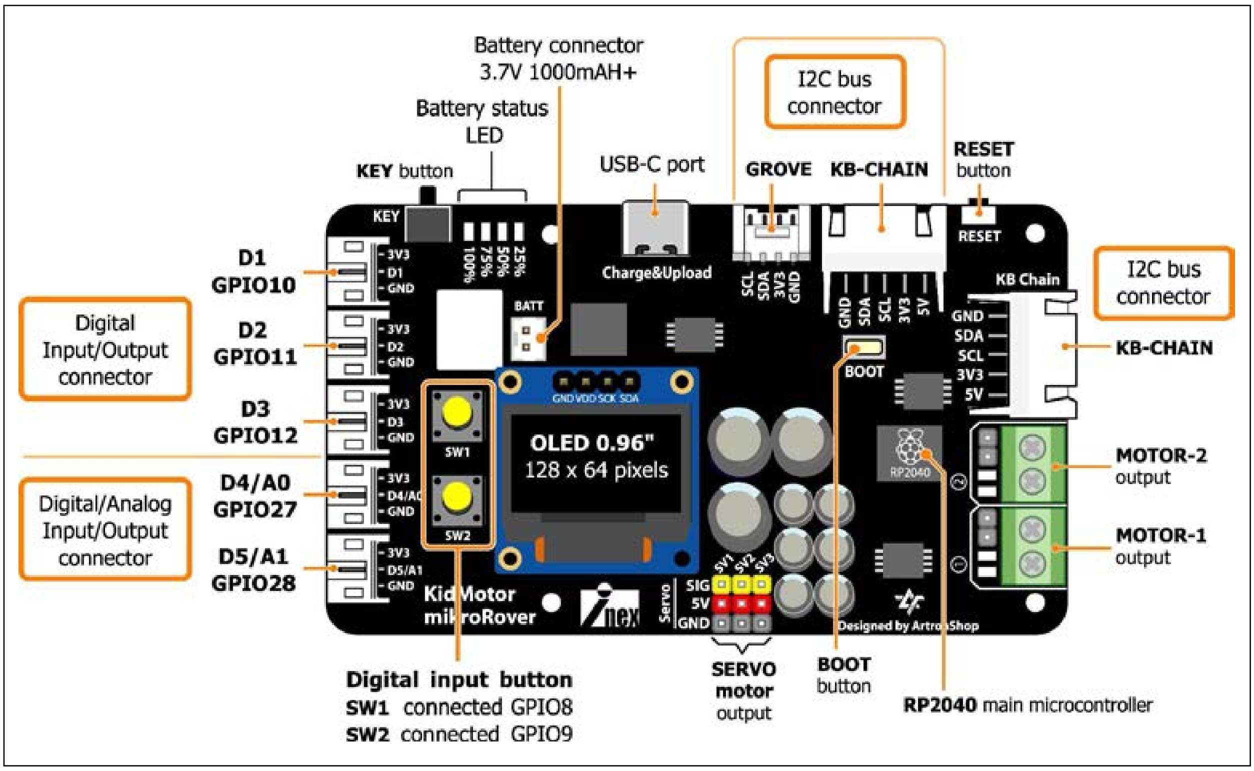


Figure 1-1 The KidMotor V4i board layout

● 2-button switch as standard digital input devices. Active with logic "0". It can be used to test the digital input devices and function as a programmable switch.

● 3-digital input/output port with JST 3-pin 2mm. connector : D1 (GPIO10), D2 (GPIO11) and D3 (GPIO12) of main microcontroller chip RP2040

● 2-analog/digital input/output port with JST 3-pin 2mm. connector : D4/ADC0 (GPIO26) and D5/ADC1 (GPIO27). Analog input range is 0 to 3.3V.

1.2 How to turn on/off

1.2.1 Turning om

Press the **KEY** button at once. The KidMotor V4i will be activated and shows LED battery status.

1.2.2 Turning off

Press the **KEY** button twice time quickly. The power supply circuit will be off and all LED battery status are off.

1.2.3 LED Battery status

There are 4 levels of battery status indicators. They are 25%, 50%, 75% and 100%. They will lit following the battery level status as shown in the Table 1-1 and Table 1-2 depends on operating condition.

Battery voltage level (running time)	**LED status**			
	25%	**50%**	**75%**	**100%**
more than 75%	**ON**	**ON**	**ON**	**ON**
more than 50% to 75%	**ON**	**ON**	**ON**	**OFF**
more than 25% to 50%	**ON**	**ON**	**OFF**	**OFF**
more than 3% to 25	**ON**	**OFF**	**OFF**	**OFF**
more than 0% to 3%	**OFF**	**OFF**	**OFF**	**OFF**

Table 1-1 Battery status LED operation of Running time

Battery voltage level (charging time)	LED status			
	25%	50%	75%	100%
100%	ON	ON	ON	ON
more than or equal 75%	ON	ON	ON	BLINK
more than or equal 50%	ON	ON	BLINK	OFF
more than or equal 25%	ON	BLINK	OFF	OFF
less than 25%	BLINK	OFF	OFF	OFF

Table 1-2 Battery status LED operation of Charging time

1.2.4 Charging battery

KidMotor V4i board charges the battery via USB-C port. Battery charge current is 1.5A. To charge, it is recommended to use a mobile charger that supply voltage 5V and current not less than 2A.

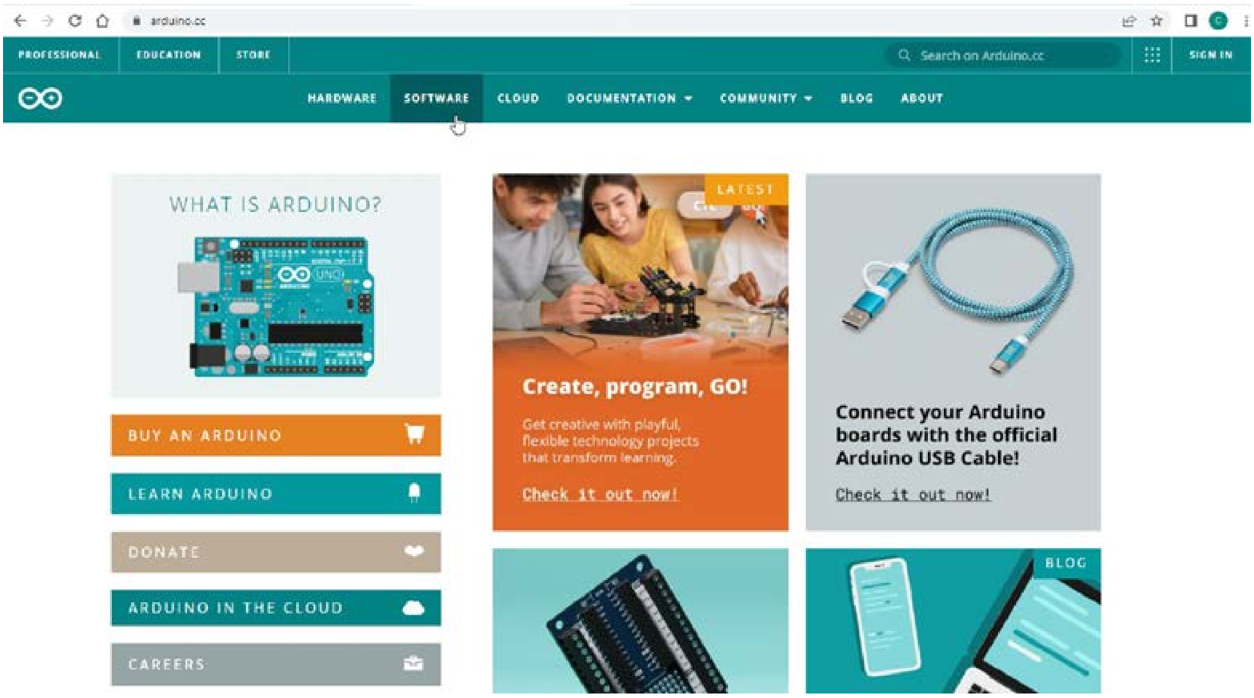
Arduino IDE Installation 02

mikroRover robot is controlled by the main microcontroller RP2040 from Raspberry Pi Foundation. There are many development software for making the code to operate. One is Arduino IDE. This documentation will be use this software tools for developing the operation code for mikroRover robot.

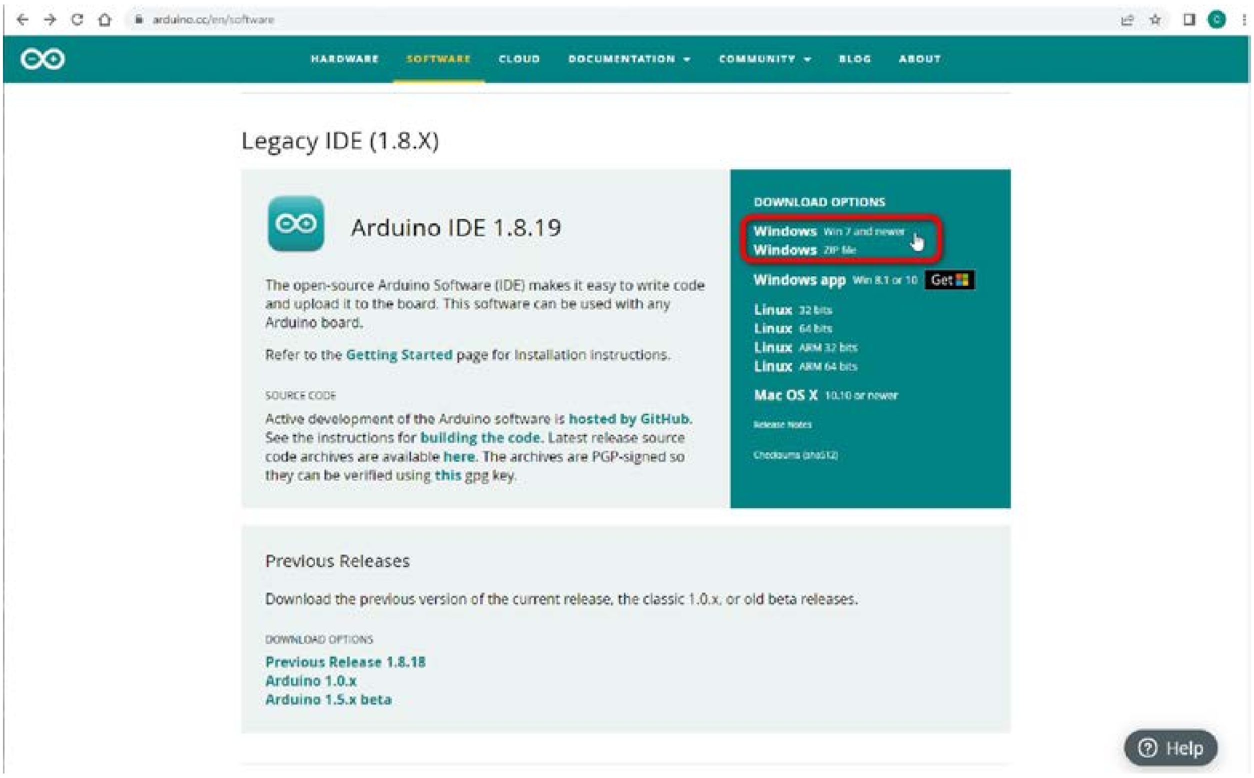
2.1 Installation

Installation step is as follows :

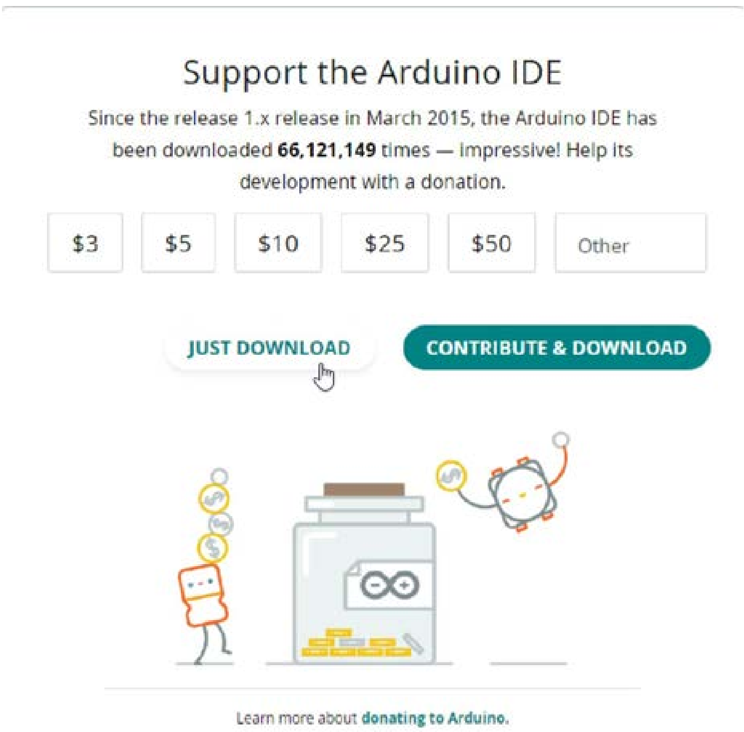
(2.1.1) Enter to the official Arduino website at **https://www.arduino.cc/**. Choose the menu **Software** and click on.



(2.1.2) Scroll to find the **Arduino Legacy IDE (1.8.X).** Select the computer operation system that use to create the Arduino code. Now select **Arduino IDE 1.8.19 Windows (Win 7 and more)**

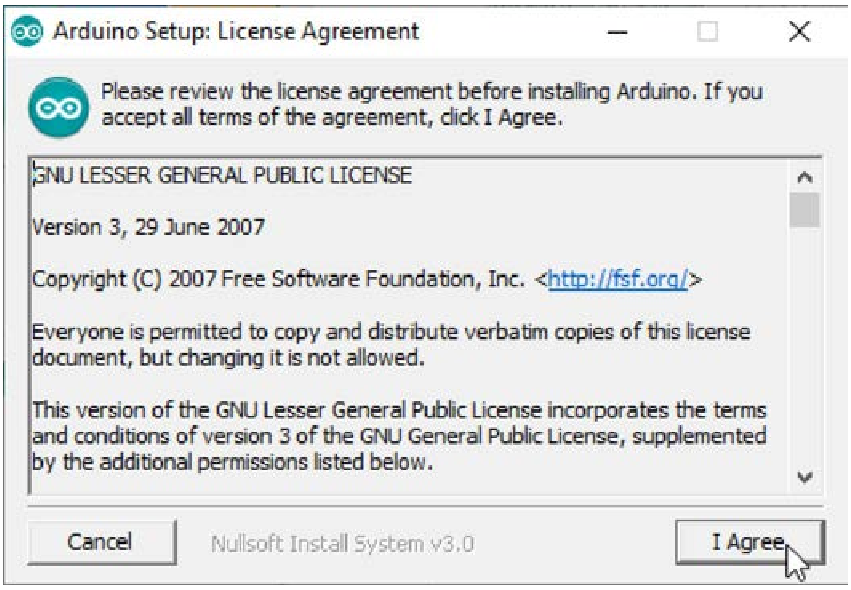


(2.1.3) Next, click on the **JUST DOWNLOAD** button.

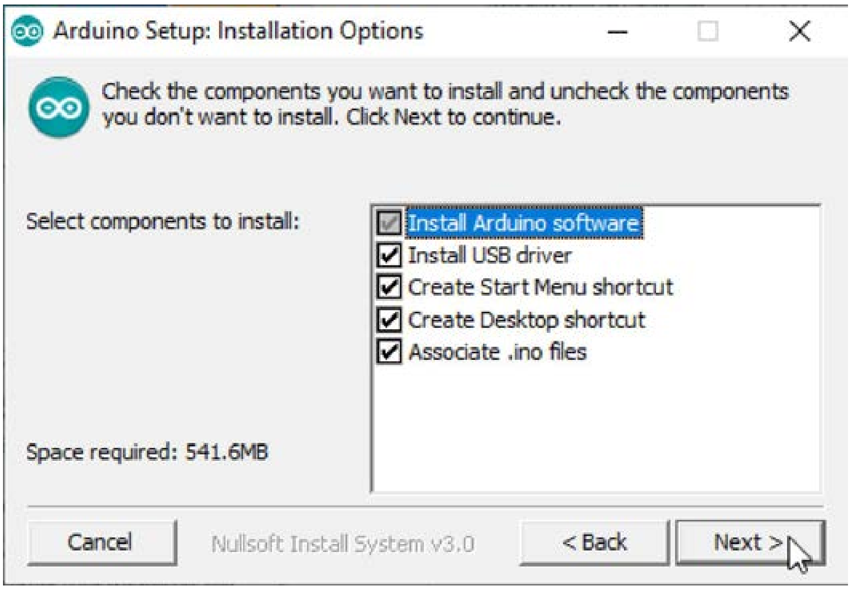


(2.1.4) Wait for downloading the Arduino IDE software. After download already, double click on the **arduino-1.8.19-windows.exe** file for starting installation.

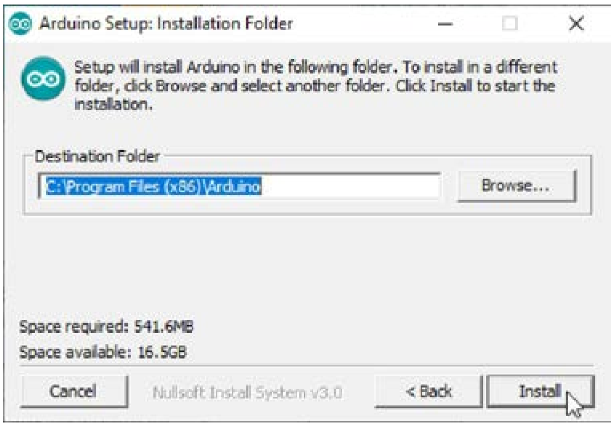
(2.1.5) The **license agreement** window will be appeared. Click on **I agree button**.

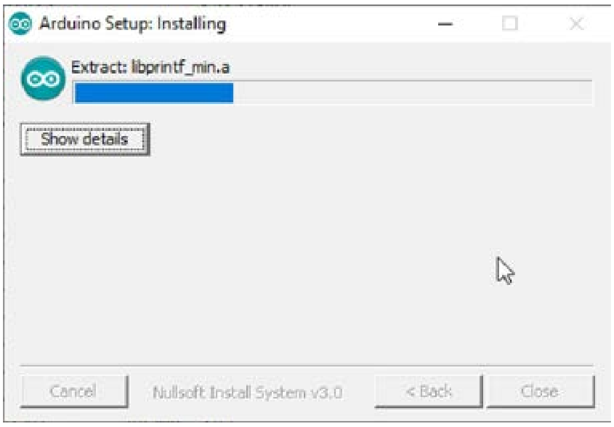


(2.1.6) Click to select all components then click on the **Next** button.



(2.1.7) Select the installation folder. Select from default then click on the **Install** button. Installation will be start. Wait until the installation is completed.

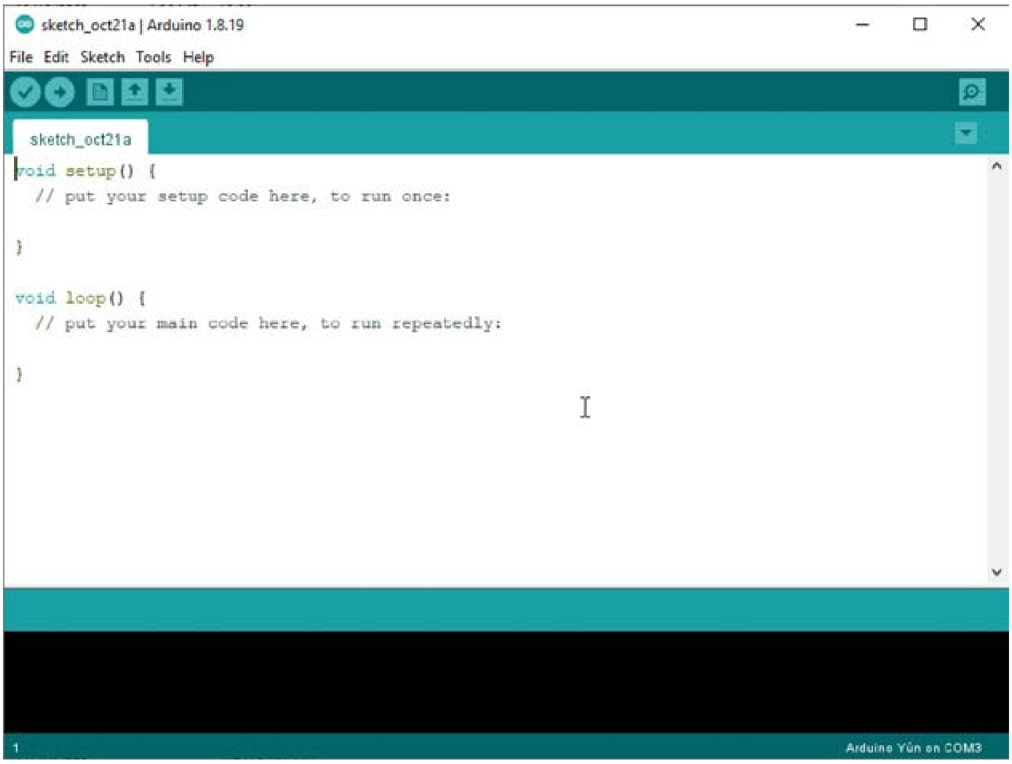




(2.1.8) Click on the **Close** button.

(2.1.9) Try to run the Arduino IDE softwate by double click on the file **arduino.exe** in **Arduino** folder or run from start menu and select the **Arduino** on the list.

(2.1.10) The main window of Arduino IDE will be appeared.

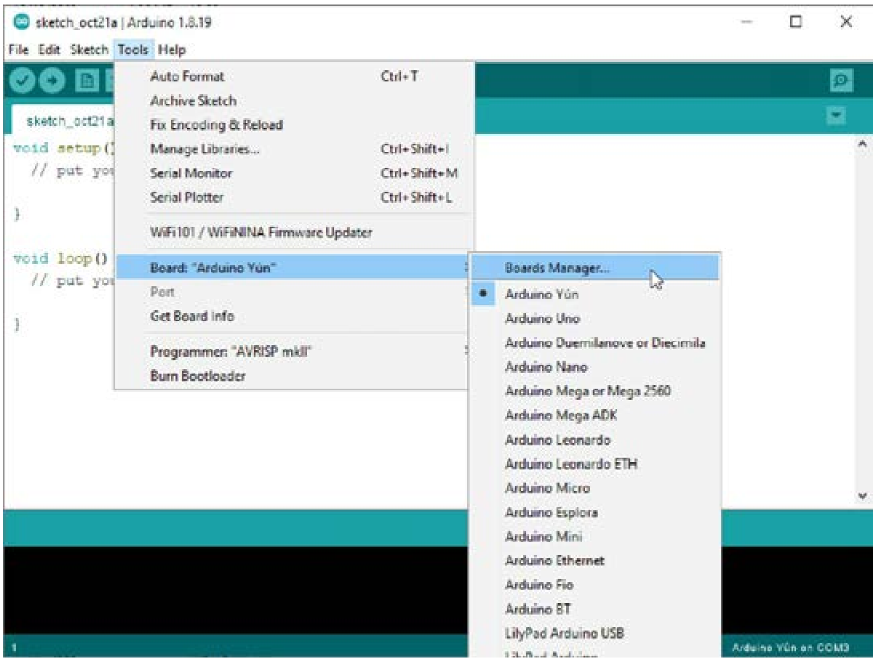


2.2 Setting Arduino IDE for RP2040 microcontroller

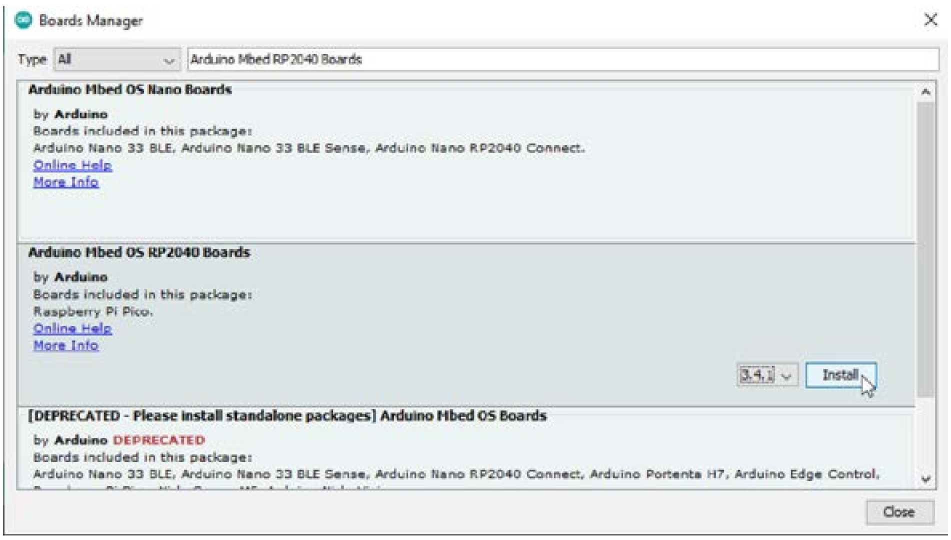
After installation Arduino IDE completed, next step is setting the Aduino IDE to work with RP2040 microcontroller board. The setting is done by using Board Manager menu in Arduino IDE.

The step is as follows :

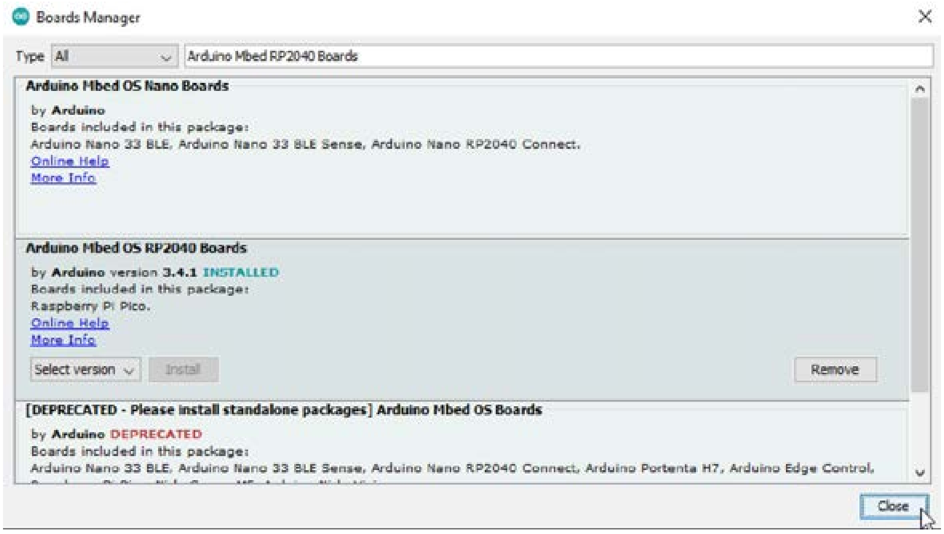
(2.2.1) At the main window of Arduino IDE, select the menu **Tools>Board>Board Manager**. The **Board Manager window** will be appeared.



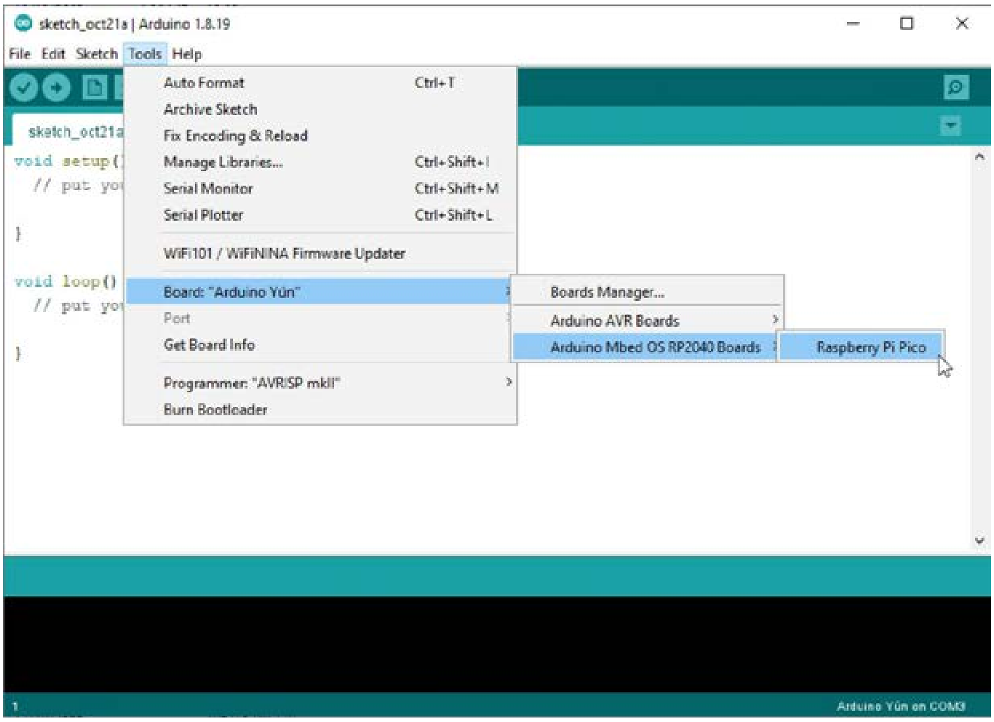
(2.2.2) Type the words "**Arduino Mbed RP2040 Boards**" into the search box. Then select the list **Arduino Mbed OS RP2040 Boards** latest version. Click on the **Install** button to add this hardware info to Arduino IDE.



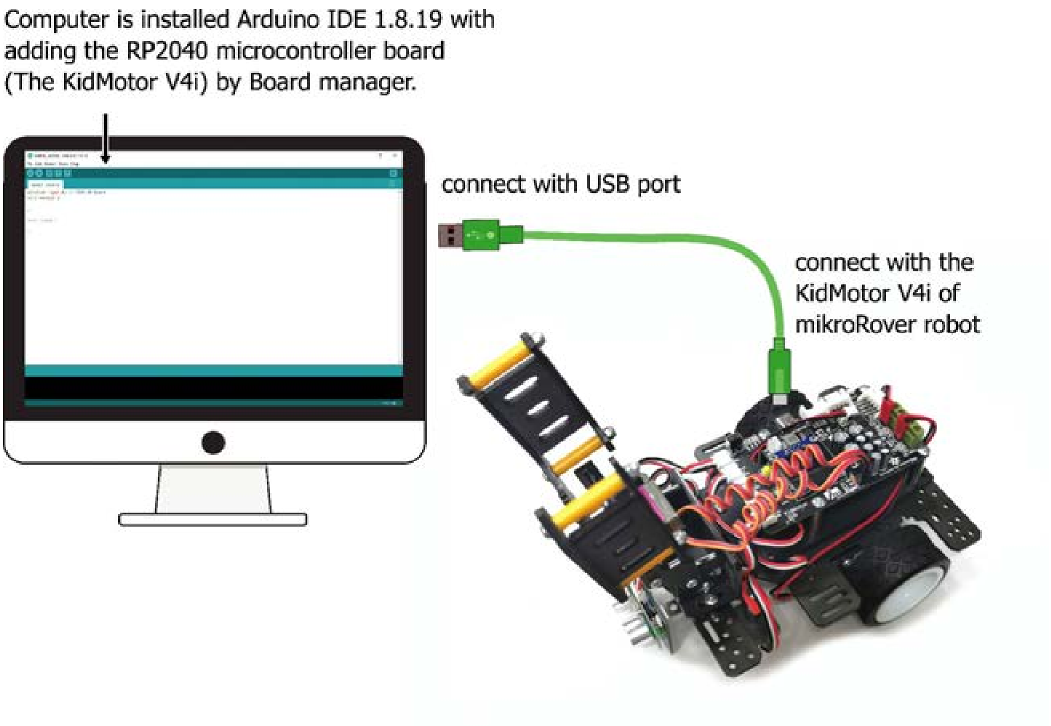
(2.2.3) Wait until the Board Manager process is finished. Close this window and back to the main window.



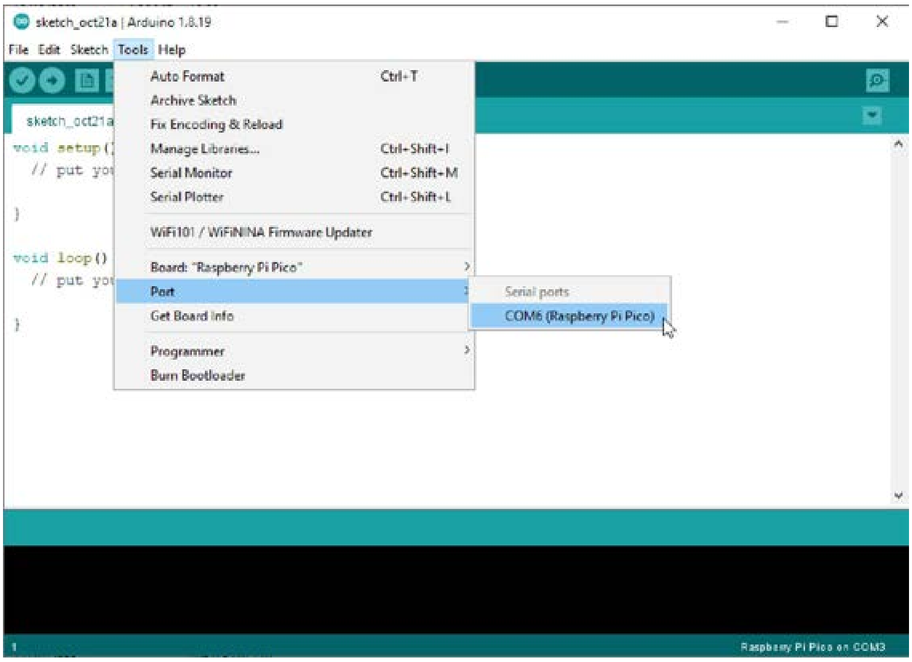
(2.2.4) Check the adding hardware complete by select to menu **Tools>Board**. User will be see the list of **Arduino Mbed OS RP2040 Boards > Raspberry Pi Pico**. The setting Arduino IDE for RP2040 microcontroller board is completed.



(2.2.5) Connect the mikroRover robot to computer by using USB-C data cable.



(2.2.6) Check the connection of RP2040 microcontroller board; KidMotor V4i by select menu **Tools > Port > COMx (Raspberry Pi Pico)**



Let's code now !

Getting start with OLED 03

The first practice for mikroRover robot is using the 0.96” OLED 128 x 64 pixel. It is a display device of KidMotor V4i controller board of mikroRover. It displays characters, number, symbol and any custom graphic on one color. The interfacing with main microcontroller RP2040 of OLED is I2C bus. This communication bus requires only 2 signal wires; **SDA** (Serial Data) and **SCL** (Serial Clock).

3.1 Adding OLED library

The development programming for OLED in C/C++ wth Arduino IDE requires some important libraries; **adafruit_SSD1306.h** and **adafruit_GFX.h**. There are installing library procedure as follows :

(3.1.1) Connect computer to internet network.

(3.1.2) Open the Arduino IDE.

(3.1.3) Select menu **Sketch > Include Library > Manage Libraries**. The Library Manager window will appear as shown in the Figure 3-2.

(3.1.4) Type “**SSD1306**” in the search box and select **Adafruit SSD1306**. Then click on the **Install** button to install the library.

(3.1.5) The dialog box of installation option about more related library is shown as figure 3-3. Click on the **Install all** button to start.

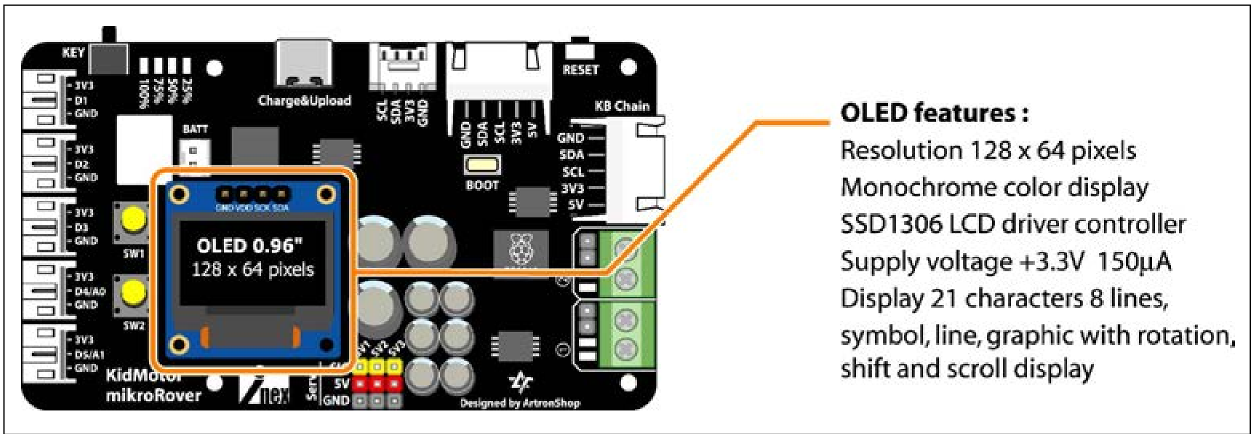


Figure 3-1 OLED of KidMotor V4i controller board

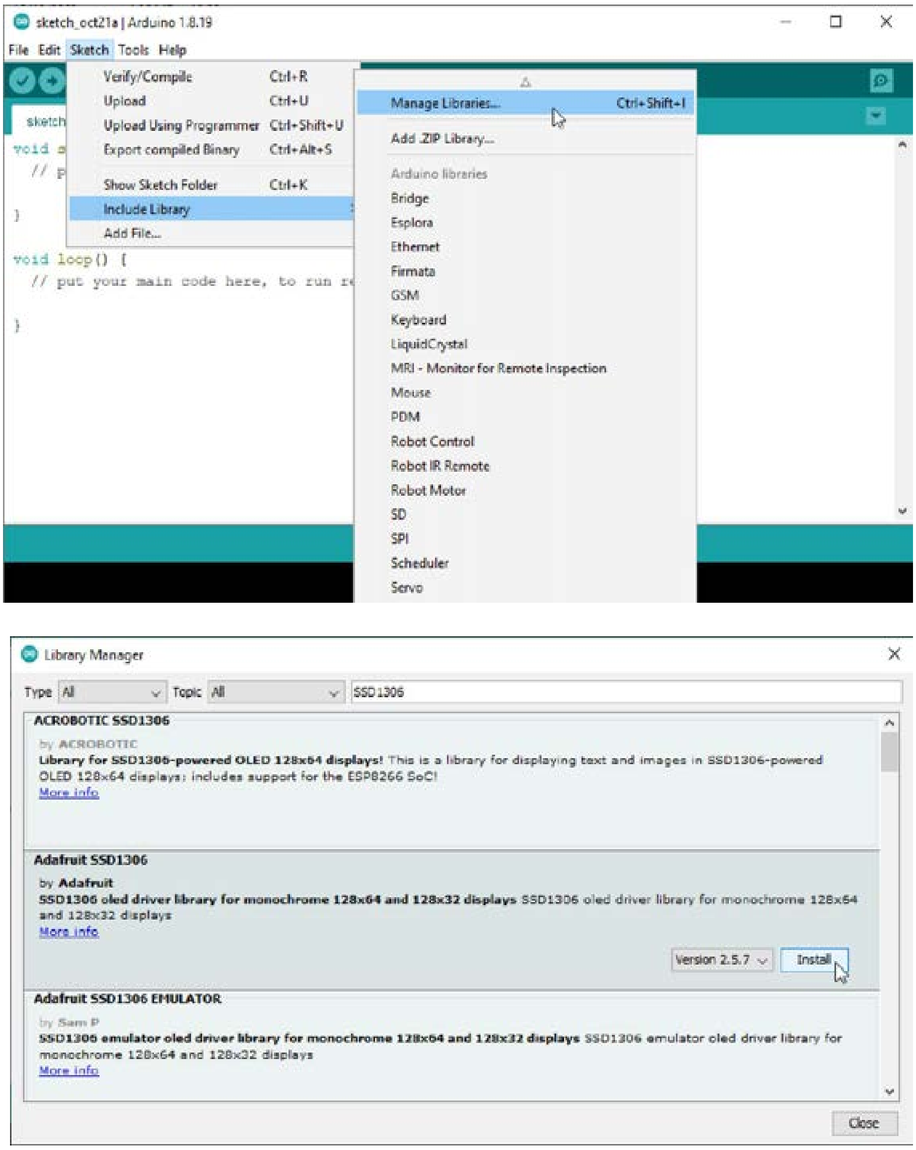


Figure 3-2 Library Manager window

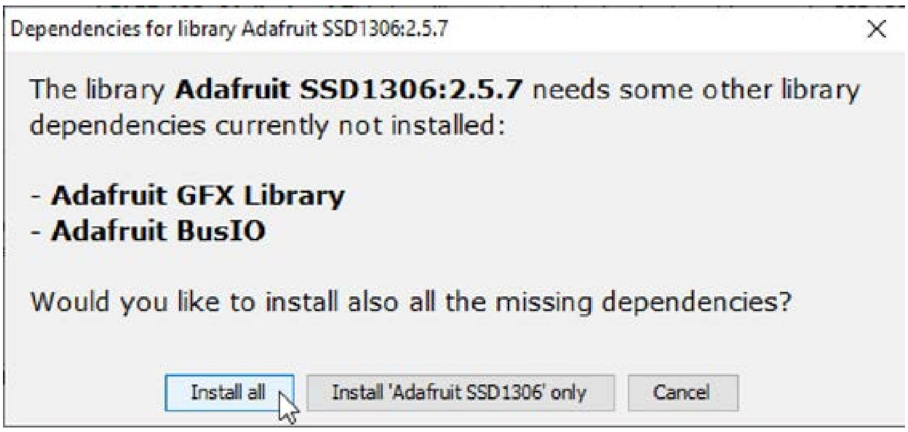


Figure 3-3 Installation option of Adafruit SSD1306 library

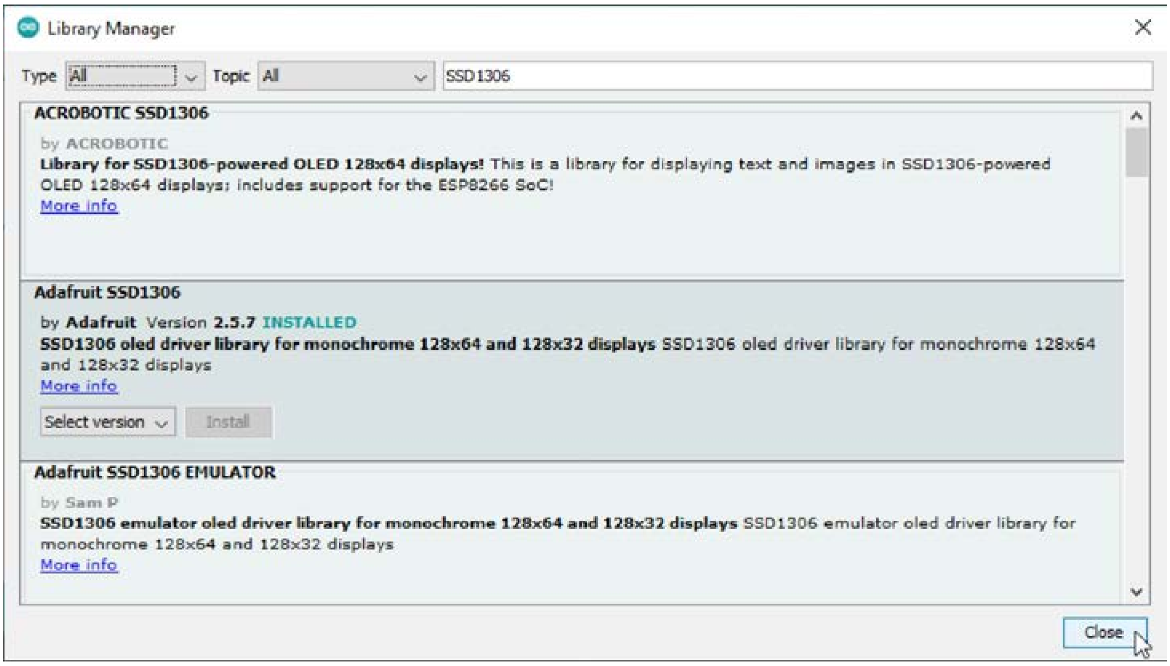


Figure 3-4 Library manager after the Adafruit SSD13606 is installed

(3.1.6) Installation library is start It install 2 important libraries :

Adafruit SSD1306

Adafruit GFX

The installation take times. Wait for the installation complete. Click on the Close button for exiting the library installation following the Figure 3-4

(3.1.7) Check the library installation by select menu **Sketch > Include Library** and scroll to find Recommended libraries header. The **Adafruit SSD1306** and **Adafruit GFX** library will appear and ready to use as shown in Figure 3-5.

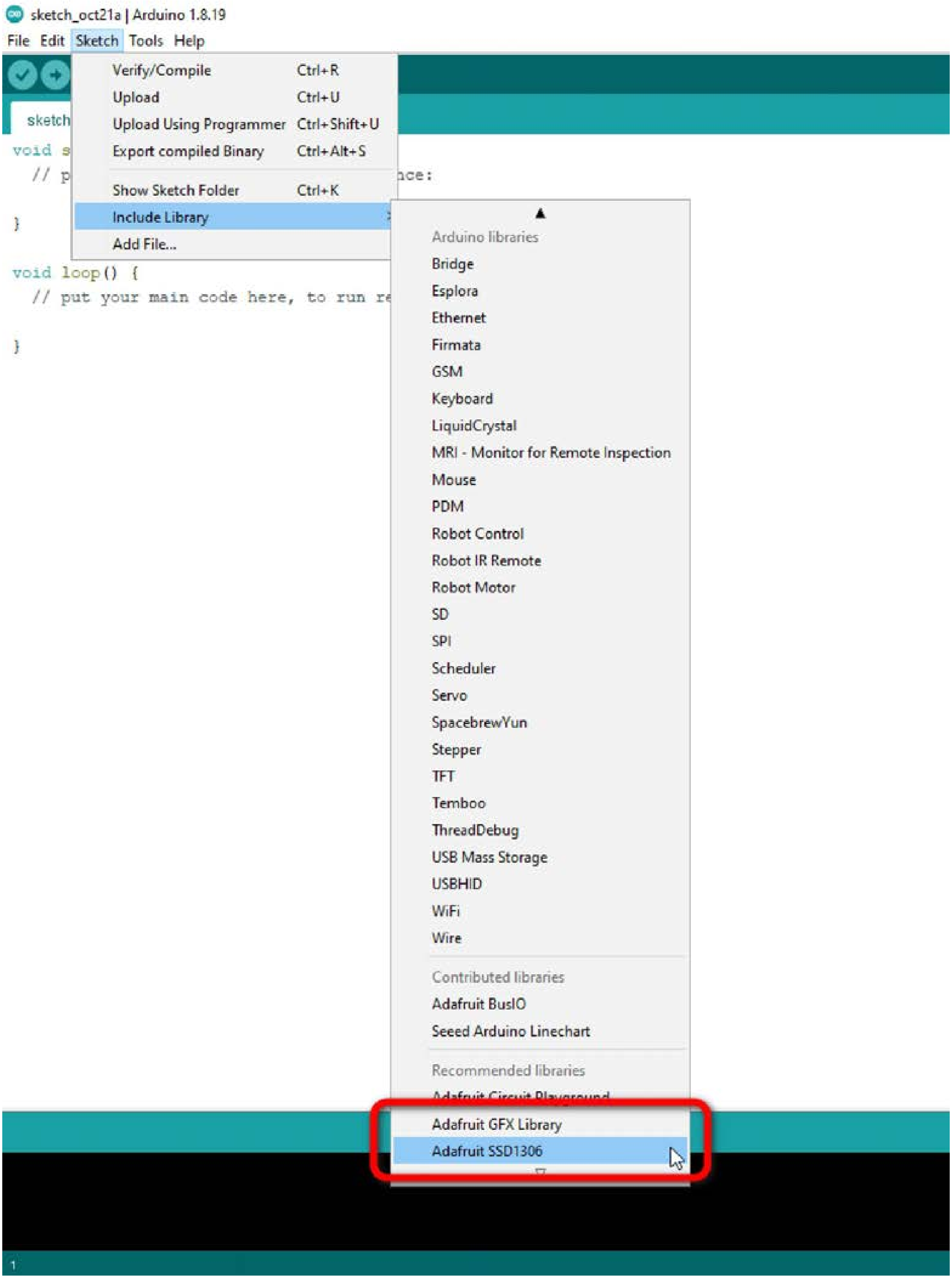


Figure 3-5 List of Adafruit SSD1306 and Adafruit GFX library in Arduino IDE after installation

3.2 The first coding for using OLED : Hello World

There are steps as follows :

(3.2.1) Open the Arduino IDE

(3.2.2) Connect the USB-C cable between the mikroRover robot with computer. Turn on power.

(3.2.3) Select the hardware as Raspberry Pi Pico following the Figure 3-6

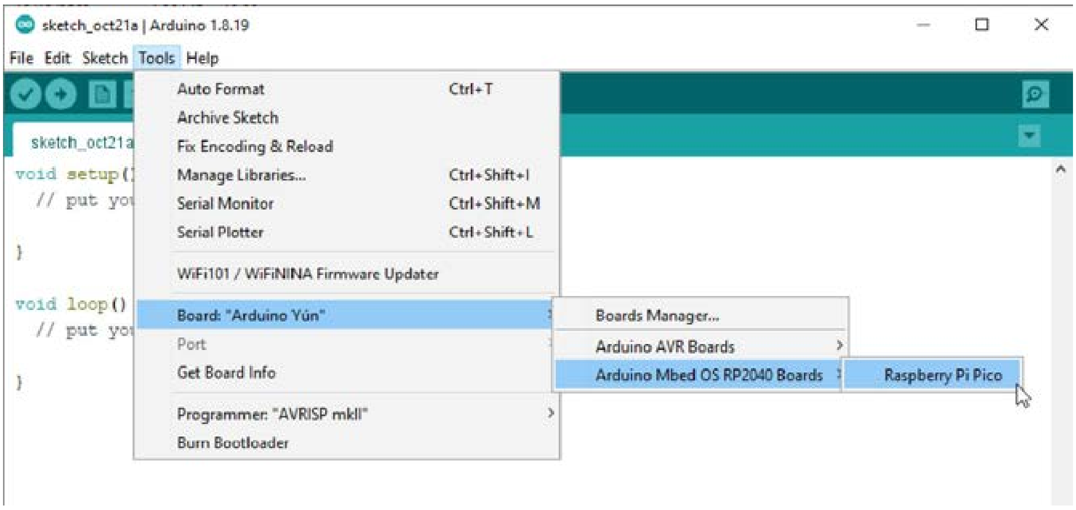


Figure 3-6 Select the board as Raspberry Pi Pico

(3.2.4) Select the interfaced COM port following the Figure 3-7

(3.2.5) Create the code following Lisitng 3-1

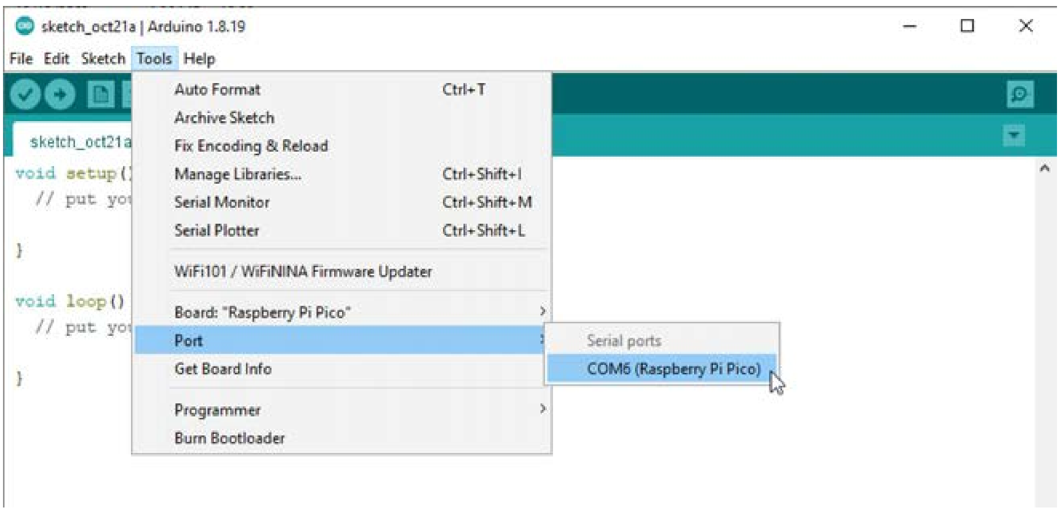


Figure 3-7 Select the COM port for serial communication between Arduino IDE and mikroRover robot

```
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 64 // OLED display height, in pixels

// Declaration for an SSD1306 display connected to I2C (SDA, SCL pins)
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);

void setup()
{
  Serial.begin(115200);

  if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3C))
  { // Address 0x3D for 128x64
    Serial.println(F("SSD1306 allocation failed"));
    for(;;);
  }
  delay(2000);
  display.clearDisplay();

  display.setTextSize(1);
  display.setTextColor(WHITE);
  display.setCursor(0, 10);
  // Display static text
  display.println("Hello, world!");
  display.display();
}
void loop()
{}
```

Lisitng 3-1 The first program for using OLED of the mikroRover robot

(3.2.6) Compile and Upload the program to mikroRoverr robot by clicking on the button or select from menu File > Upload. Wait for compiling and uploading process. After complete the hardware will be run immediatly.

The mikroRover's OLED shows the message Hello, world following the Figure 3-8. See the demonstration video of this activity at ***https:/youtu.be/Cuft0mxBhNA***

Figure 3-8 The result of Listing 1 program is shown on the OLED of mikroRover robot.

3.3 OLED library troubleshooting

When install and using the OLED library set the first time, possible found the error compiling for Raspberry Pi Pico (RP2040 microcontroller) following the Fiugure 3-9. The error is from a problem in the file **Adafruit_SSD1306.cpp** of Adafruit SSD1306 library. The **pgmspace.h** file is missing.

```
Error compiling for board Raspberry Pi Pico

C:\Users\SakuranohanaTH\Documents\Arduino\libraries\Adafruit_SSD1306\Adafruit_SSD1306.cpp:42:10: fatal error: pgmspace.h: No such file or directory
 #include <pgmspace.h>
          ^~~~~~~~~~~~
compilation terminated.
exit status 1
Error compiling for board Raspberry Pi Pico.
```

Figure 3-9 The error compiling messages is shown in the message box of Arduino IDE

Troubleshooting is as follows :

(3.3.1) Enter to folder **C:\Users\(user name)\Documents\Arduino\libraries\Adafruit_SSD1306** and open the file **Adafruit_SSD1306.cpp** by any text editor program such as Microsoft Visual Studio Code, Notepad ++ or Wordpad.

(3.3.2) Edit the line number 39 to 46 from

```
39 #ifdef __AVR__
40 #include <avr/pgmspace.h>
41 #elif defined(ESP8266) || defined(ESP32) || defined(ARDUINO_ARCH_RP2040)
42 #include <pgmspace.h>
43 #else
44 #define pgm_read_byte(addr)
45   (*(const unsigned char *)(addr)) ///< PROGMEM workaround for non-AVR
46 #endif
```

to be as shown below

```
39 #ifdef __AVR__
40 #include <avr/pgmspace.h>
41 #elif defined(ESP8266) || defined(ESP32)
42 #include <pgmspace.h>
43 #elif defined(ARDUINO_ARCH_RP2040)
44 #include <api\deprecated-avr-comp\avr\pgmspace.h>
45 #else
46 #define pgm_read_byte(addr)
47   (*(const unsigned char *)(addr)) ///< PROGMEM workaround for non-AVR
48 #endif
```

(3.3.3) Save the file and exit the Text editor program.

(3.3.4) Back to Arduino IDE. Compile the program and upload again.

3.4 More OLED example code

There are 2 example programs for using OLED features.

Listing 3-2 is showing the messages scrolling of OLED.

See the result video from ***https://youtu.be/-V8PyA6MVjk***

```
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 64 // OLED display height, in pixels

// Declaration for an SSD1306 display connected to I2C (SDA, SCL pins)
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);

void setup()
{
  Serial.begin(115200);
  if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3C))
  { // Address 0x3D for 128x64
    Serial.println(F("SSD1306 allocation failed"));
    for(;;);
  }
  delay(2000);
  display.clearDisplay();

  display.setTextSize(1);
  display.setTextColor(WHITE);
  display.setCursor(0, 0);
  // Display static text
  display.println("Scrolling Hello");
  display.display();
  delay(100);
}
void loop()
{
  // Scroll in various directions, pausing in-between:
  display.startscrollright(0x00, 0x0F);
  delay(2000);
  display.stopscroll();
  delay(1000);
  display.startscrollleft(0x00, 0x0F);
  delay(2000);
  display.stopscroll();
  delay(1000);
  display.startscrolldiagright(0x00, 0x07);
  delay(2000);
  display.startscrolldiagleft(0x00, 0x07);
  delay(2000);
  display.stopscroll();
  delay(1000);
}
```

Listing 3-2 The demonstration program for showing the message scrolling feature of OLED

Listing 3-3 is showing the font changing of OLED.

See the result video from ***https://youtu.be/o9JHBj—MBE***

```
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <Fonts/FreeSerif9pt7b.h>

#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 64 // OLED display height, in pixels

// Declaration for an SSD1306 display connected to I2C (SDA, SCL pins)
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);

void setup()
{
  Serial.begin(115200);

  if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3C))
  {
    Serial.println("SSD1306 allocation failed");
    for(;;);
  }
  delay(2000);

  display.setFont(&FreeSerif9pt7b);
  display.clearDisplay();
  display.setTextSize(1);
  display.setTextColor(WHITE);
  display.setCursor(0,20);
  display.println("Hello, world!");
  display.display();
  delay(2000);
}
void loop()
{}
```

Listing 3-3 The demonstration program for showing font changing feature of OLED

Robot movement testing 04

mikroRobot has 2 of DC motor driver outputs and 3 of servo motor outputs. The schematic for these features is shown in the Figure 4-1.

4.1 Microcontroller pin assignment for motor output

The main microcontroller RP2040 use 2 GPIO pins for connecting with one of DC motor driver circuit. These are pin assignment for all motor outputs.

DC motor output-1 : GPIO13 and 14

DC motor output-2 : GPIO16 and 17

Servo motor output : GPIO18 (SV1), GPIO19 (SV2) and GPIO20 (SV3)

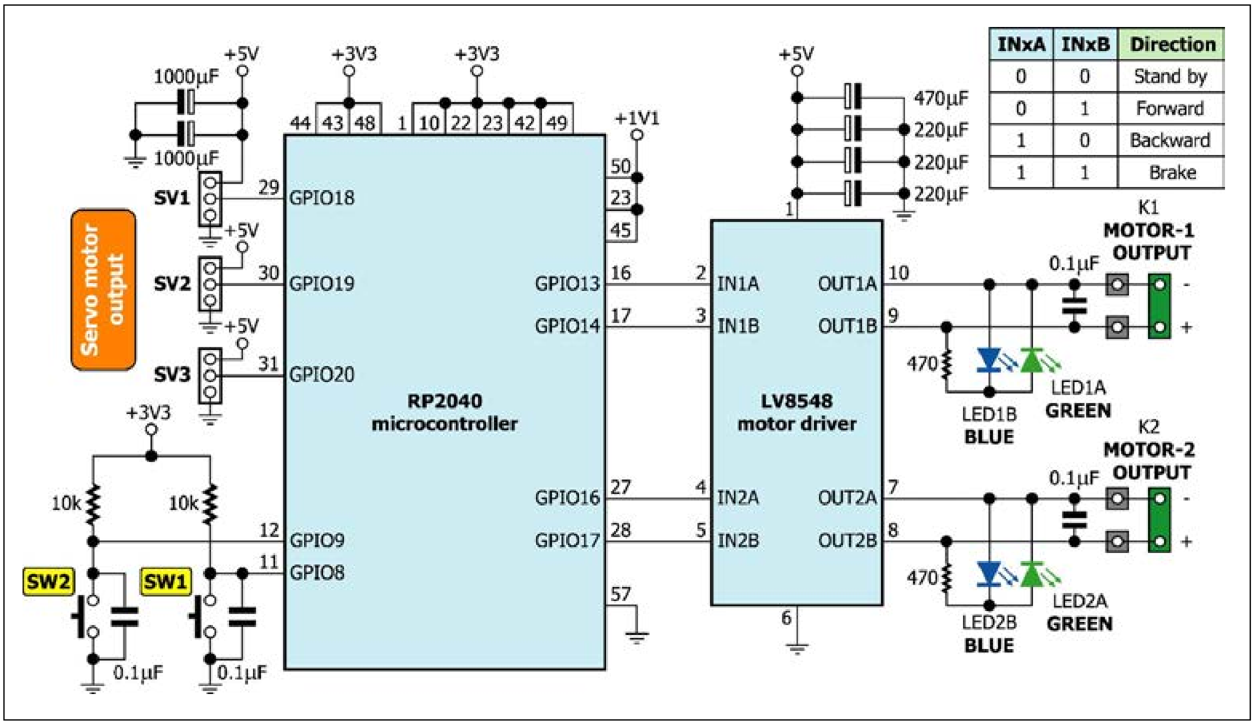


Figure 4-1 mikroRover robot's motor driver schematic

4.2 Robot moving assignment

Figure 4-2 shows the DC motor assignment of mikroRover robot.

Left motor is connected with DC motor output-1.

Right motor is connected with DC motor output-2.

All movement operation of mikroRover robot is summarized as follows :

Forward moving (fd) : both motors are moved forward.

Backward moving (bk) : both motors are moved backward.

Spin Left (sl) : Left motor is moved backward. Right motor is moved forward

Spin Right (sr) : Left motor is moved forward. Right motor is moved backward

Turn Left (tl) : Left motor is off. Right motor is moved forward.

Turn Right (tr) : Left motor is moved forward. Right motor is off.

Off (ao) : both motors are off.

Brake : Both motors are braked.

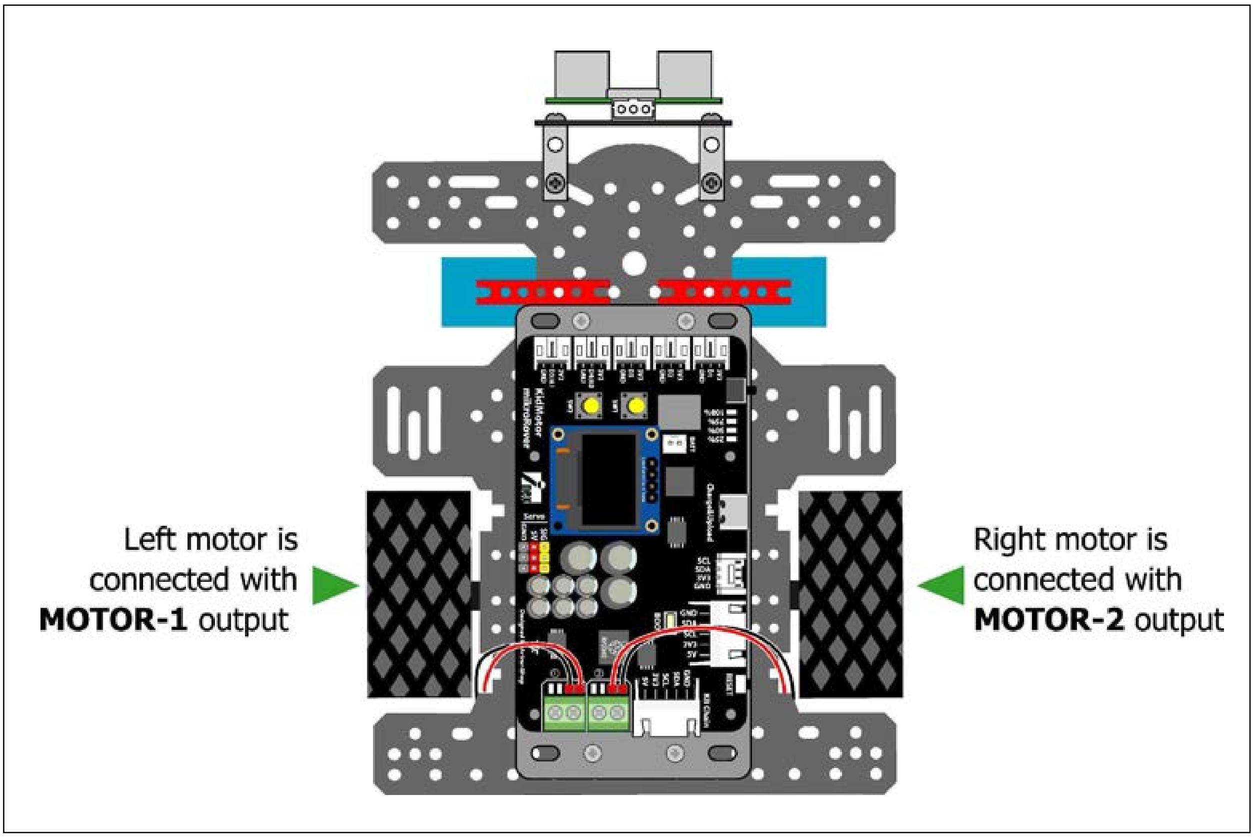


Figure 4-2 mikroRover robot's motor assignment

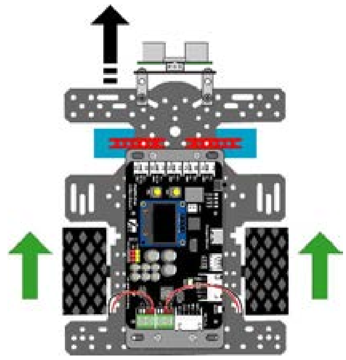
4.3 Movement function programming of mikroRover robot

About coding development for mikroRover robot's movement in C/C++ with Arduino IDE in thid chapeter will create from simple command without special library. By using **`analogWrite ()`** function, microcontroller will generate PWM signal and send to DC motor driver circuit for driving motor. Main parameter of **`analogWrite ()`** function is **0 to 255** for making the power to DC motor driver 0 to 100%.

Summary of mikroRover robot's movement function are as follows :

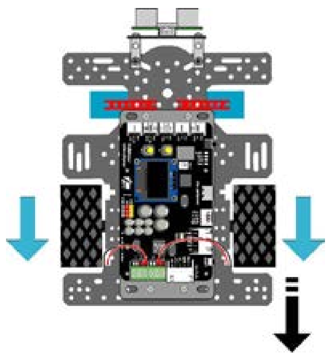
4.3.1 Forward moving (fd)

```
unsigned int pwm = map(constrain(speed, 0, 100), 0, 100, 0, 255);
analogWrite(13, 0);
analogWrite(14, pwm);
analogWrite(16, 0);
analogWrite(17, pwm);
```



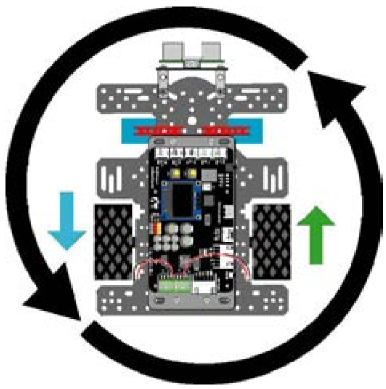
4.3.2 Backward moving (bk)

```
unsigned int pwm = map(constrain(speed, 0, 100), 0, 100, 0, 255);
analogWrite(13, pwm);
analogWrite(14, 0);
analogWrite(16, pwm);
analogWrite(17, 0);
```



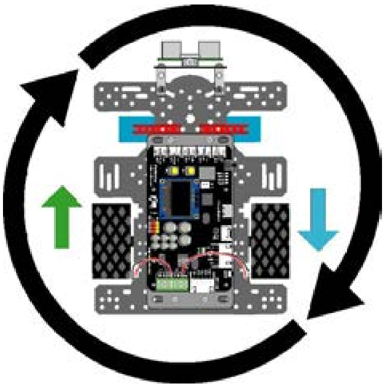
4.3.3 Spin Left (sl)

```
unsigned int pwm = map(constrain(speed, 0, 100), 0, 100, 0, 255);
analogWrite(13, pwm);
analogWrite(14, 0);
analogWrite(16, 0);
analogWrite(17, pwm);
```



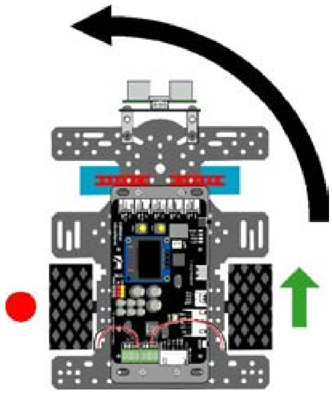
4.3.4 Spin Right (sr)

```
unsigned int pwm = map(constrain(speed, 0, 100), 0, 100, 0, 255);
analogWrite(13, 0);
analogWrite(14, pwm);
analogWrite(16, pwm);
analogWrite(17, 0);
```



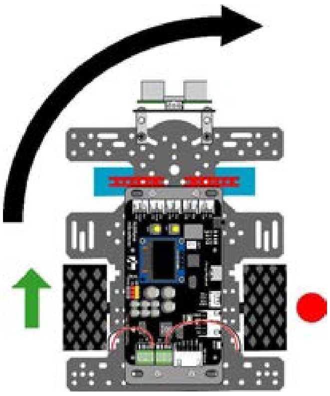
4.3.5 Turn Left (tl)

```
unsigned int pwm = map(constrain(speed, 0, 100), 0, 100, 0, 255);
analogWrite(13, 0);
analogWrite(14, 0);
analogWrite(16, 0);
analogWrite(17, pwm);
```



4.3.6 Turn Right (tr)

```
unsigned int pwm = map(constrain(speed, 0, 100), 0, 100, 0, 255);
analogWrite(13, 0);
analogWrite(14, pwm);
analogWrite(16, 0);
analogWrite(17, 0);
```



4.3.7 Off (ao)

```
analogWrite(13, 0);
analogWrite(14, 0);
analogWrite(16, 0);
analogWrite(17, 0);
```

4.4 mikroRover robot movement example

4.4.1 Simple movement control by buttons

Press SW1 and SW2 for movement control.

- SW1 for moving forward with 40% speed in 2 seconds
- SW2 for moving backward with 40% speed in 2 seconds

```
void setup()
{}
void loop()
{
  if (!digitalRead(8) &&digitalRead(9))
  {
    analogWrite(13, 0);
    analogWrite(14, 100);
    analogWrite(16, 0);
    analogWrite(17, 100);
    delay(2000);
  }
  else if (digitalRead(8) && !digitalRead(9))
  {
    analogWrite(13, 100);
    analogWrite(14, 0);
    analogWrite(16, 100);
    analogWrite(17, 0);
    delay(2000);
  }
  analogWrite(13, 0);
  analogWrite(14, 0);
  analogWrite(16, 0);
  analogWrite(17, 0);
}
```

Listing 4-1 Simple moving

4.4.2 Movement function example

This example code is shown in Listing 4-2. Watch the video at ***https://youtu.be/Ly6X_5DAQSc***.

```
void setup()
{
  while(digitalRead(8));
}

void loop()
{
   fd(50);
   delay(2000);
   ao();
   delay(1000);
   bk(50);
   delay(2000);
   ao();
   delay(1000);
   sl(50);
   delay(2000);
   ao();
   delay(1000);
   sr(50);
   delay(2000);
   ao();
   delay(1000);
   tl(50);
   delay(2000);
   ao();
   delay(1000);
   tr(50);
   delay(2000);
   ao();
   delay(1000);
}

void fd(unsigned int speed)
{
   unsigned int pwm = map(constrain(speed, 0, 100), 0, 100, 0, 255);
   analogWrite(13, 0);
   analogWrite(14, pwm);
   analogWrite(16, 0);
   analogWrite(17, pwm);
}
void bk(unsigned int speed)
{
   unsigned int pwm = map(constrain(speed, 0, 100), 0, 100, 0, 255);
   analogWrite(13, pwm);
   analogWrite(14, 0);
   analogWrite(16, pwm);
   analogWrite(17, 0);
}
```

Listing 4-2 Movement function demonstration code for mikroRover robot (continue)

```
void sl(unsigned int speed)
{
   unsigned int pwm = map(constrain(speed, 0, 100), 0, 100, 0, 255);
   analogWrite(13, pwm);
   analogWrite(14, 0);
   analogWrite(16, 0);
   analogWrite(17, pwm);
}
void sr(unsigned int speed)
{
  unsigned int pwm = map(constrain(speed, 0, 100), 0, 100, 0, 255);
   analogWrite(13, 0);
   analogWrite(14, pwm);
   analogWrite(16, pwm);
   analogWrite(17, 0);
}
void tl(unsigned int speed)
{
  unsigned int pwm = map(constrain(speed, 0, 100), 0, 100, 0, 255);
   analogWrite(13, 0);
   analogWrite(14, 0);
   analogWrite(16, 0);
   analogWrite(17, pwm);
}
void tr(unsigned int speed)
{
  unsigned int pwm = map(constrain(speed, 0, 100), 0, 100, 0, 255);
   analogWrite(13, 0);
   analogWrite(14, pwm);
   analogWrite(16, 0);
   analogWrite(17, 0);
}
void ao()
{
   analogWrite(13, 0);
   analogWrite(14, 0);
   analogWrite(16, 0);
   analogWrite(17, 0);
}
```

Listing 4-2 Movement function demonstration code for mikroRover robot (finish)

mikro rover Robot Kit

Work with Ultrasonic sensor 05

This chapter presents the example about working with the basic sensor of mikroRover robot. The mikroRover robot is installed the special function sensor as ultrasonic distance sensor***; ZX-SONAR1M*** for detecting objects or obstacles at the front. ***Connect this distance sensor to D5/A1/GPIO27 port of KidMotor V4i the controller board of mikroRover robot.***

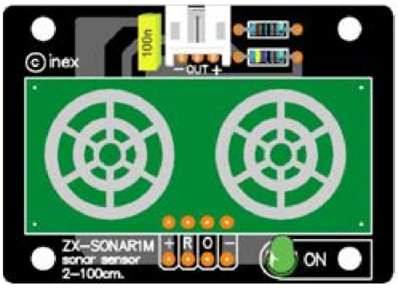
5.1 About ZX-SONAR1M sensor

The ZX-SONAR1M's layout and technical features are shown in Figure 5-1. It provides a voltage versus a detectable distance. The working principle of the ultrasonic distance measuring module is the module's transmitter spread out the 40kHz signal. When the wave hits an object, it is reflected back to the receiver. The processing circuit inside the module processes the signal and outputs it to a DC voltage of 0 to 3.3V. It is calculated to a distance of 1 to 102 centimeters by software as shown in the diagram in Figure 5-2.

5.2 Distance measurement with ZX-SONAR1M

The ZX-SONAR1M mini-board can be connected to the microcontroller by connecting the output signal to the analog input of the KidMotor V4i controller board of mikroRover robot. The programming method will read the value from the analog-to-digital converter (ADC) circuit of the pin port connected to the ZX-SONAR1M sensor and calculate it for getting the distance measuring value.

The readings from the ZX-SONAR1M are digital data 0 to 1023 (in decimal), which are not real distance values. The final result is required for a distance in centimeters then must calculate as follows:



Technical Features

- Measure distance between the target object with ultrasonic sensor
- Range is 2 to 102 centimeters in the free area
- Supply voltage 3 to 5V. Current consumption 20mA
- Output voltage 0 to 3.3V directly proportional to the distance.
- Output connector is JST type 3 pins
- LED power indicator

Figure 5-1 ZX-SONAR1M technical features

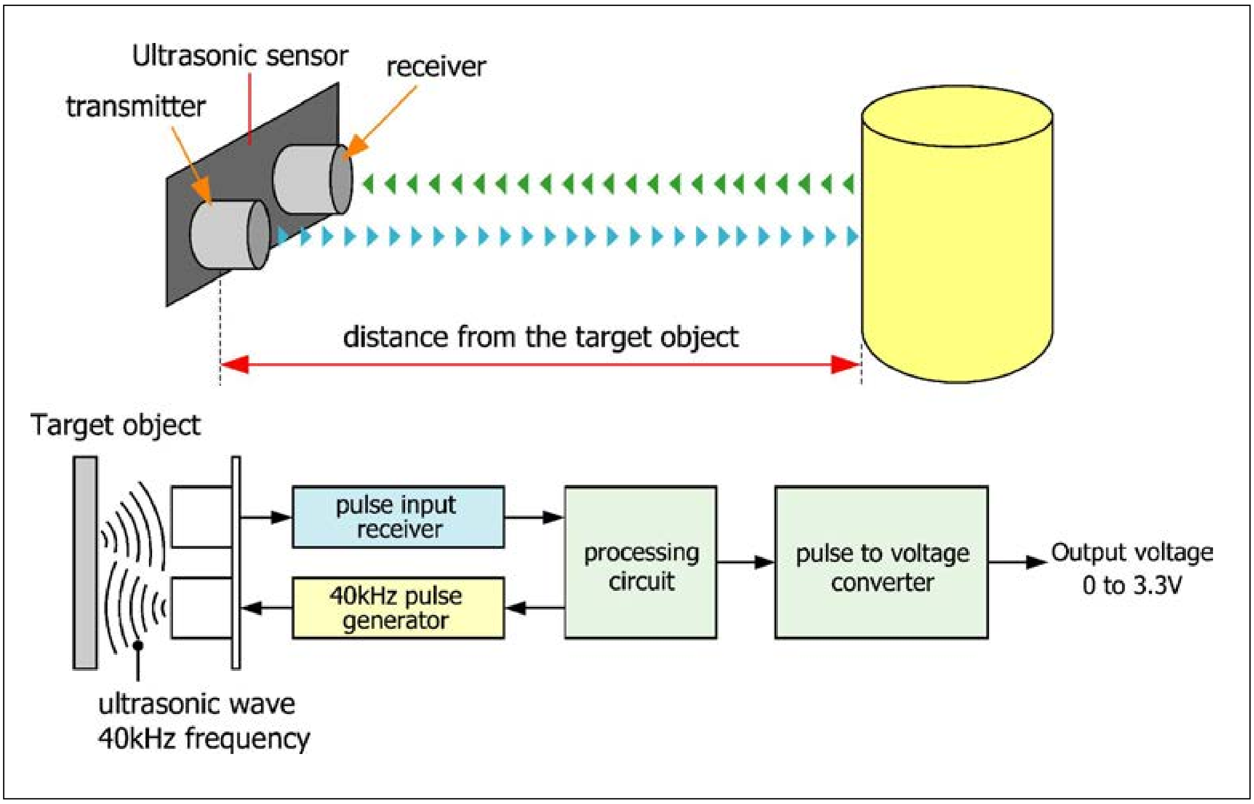


Figure 5-2 The operation diagram of ZX-SONAR1M ultrasonic distance sensor board

Distance (centimeter : cm) = ADC data x maximum distance range (equal 102cm.) / maximum 12-bit ADC data (equal 1023)

The distance is approximately equal to ADC data / 10. The summarized value is as shown in Table 5-1.

Output Voltage (Volts)	10-bit ADC data (decimal)	Distance (cm)
0	0	0
0.25	77	7
0.50	155	15
0.75	232	23
1.00	310	31
1.25	387	38
1.50	465	46
1.75	542	54
2.00	620	62

Output Voltage (Volts)	10-bit ADC data (decimal)	Distance (cm)
2.25	697	69
2.50	775	77
2.75	852	85
3.00	930	93
3.10	961	96
3.20	992	99
3.30	1023	102

Table 5-1 shows the relation of the output voltage from the ZX-SONAR1M sensor, ADC data and the measured distance

5.3 Testing the distance measurement

(5.3.1) Open the Arduino IDE and select the board as Raspberry Pi Pico and correct COM port.

(5.3.2) Create the code from Listing 5-1

(5.3.3) Upload the code from Listing 5-1 to the mikroRover robot. Then turn off the robot and unplug the USB-C cable

```
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 64 // OLED display height, in pixels

// Declaration for an SSD1306 display connected to I2C (SDA, SCL pins)
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);

void setup()
{
   Serial.begin(115200);

   if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3C))
   { // Address 0x3D for 128x64
     Serial.println(F("SSD1306 allocation failed"));
     for(;;);
   }
   delay(2000);
}
void loop()
{
   display.clearDisplay();
   display.setTextSize(3);
   display.setTextColor(WHITE);
   display.setCursor(0, 0);
   // Display static text
   display.println(analogRead(1)/10);
   display.display();
   delay(100);
}
```

Listing 5-1 Example code of reading the distance measurement from the ZX-SONAR1M and display on the OLED of the mikroRover robot.

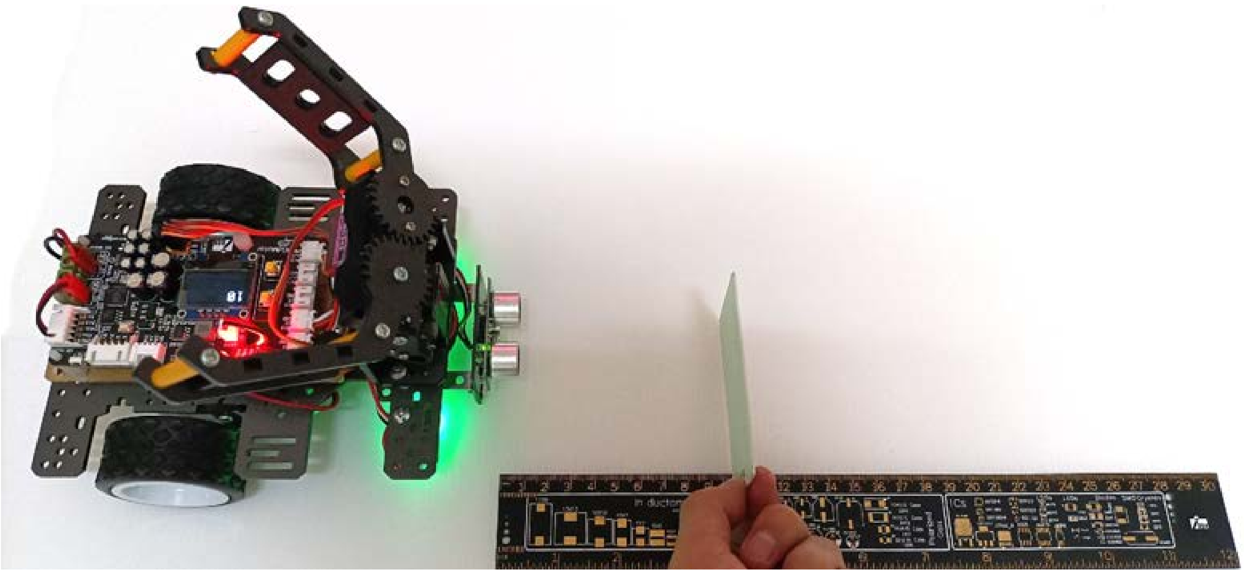


Figure 5-3 Testing the operation of ZX-SONAR1M sensor with mikroRover robot

(5.3.4) Place the robot on the floor. Turn on power. Prepare some object such as glass, paper, book or using hand to cover the front of the ZX-SONAR1M sensor following the Figure 5-3 Observe the result on the OLED screen of the mikroRover robot.

- ***If the object is moved away, the measured value will increase.***
- I***f the object moves closer, the measured value will decrease.***

Watch the operation video from ***https://youtu.be/x2rgjYPiudw.***

5.4 Contactless object avoiding robot

5.4.1 Activity material

1. mikroRover robot and USB-C data cable

2. Material for making the obstacle, allow any shapes and colors. Size is must bigger than the ZX-SONAR1M sensor board for good result.

5.4.2 Activity steps

(1) Connect the mikroRover robot with computer via USB port. The robot would be turn on automatically.

(2) Open the Arduino IDE. Select the board as Raspberry Pi Pico and set the correct COM port.

(3) Create this activity's code following the Listing 5-2. The program is made from standard function of Arduino without the custom library. It included all robot's movement functions from Chapter 4.

```
void setup()
{
  while (digitalRead(8));
}
void loop()
{
  if ((analogRead(1) / 10) < 30)
  {
    ao();
    delay(200);
    sr(50);
    delay(750);
  }
  else
  {
    fd(50);
  }
}
void fd(unsigned int speed)
{
  unsigned int pwm = map(constrain(speed, 0, 100), 0, 100, 0, 255);
  analogWrite(13, 0);
  analogWrite(14, pwm);
  analogWrite(16, 0);
  analogWrite(17, pwm);
}
void bk(unsigned int speed)
{
  unsigned int pwm = map(constrain(speed, 0, 100), 0, 100, 0, 255);
  analogWrite(13, pwm);
  analogWrite(14, 0);
  analogWrite(16, pwm);
  analogWrite(17, 0);
}
void sl(unsigned int speed)
{
  unsigned int pwm = map(constrain(speed, 0, 100), 0, 100, 0, 255);
  analogWrite(13, pwm);
analogWrite(14, 0);
analogWrite(16, 0);
analogWrite(17, pwm);
}
void sr(unsigned int speed) {
  unsigned int pwm = map(constrain(speed, 0, 100), 0, 100, 0, 255);
analogWrite(13, 0);
analogWrite(14, pwm);
analogWrite(16, pwm);
analogWrite(17, 0);
}
```

Listing 5-2 The contactless object avoiding robot program for mikroRover robot (continue)

```
void tl(unsigned int speed)
{
  unsigned int pwm = map(constrain(speed, 0, 100), 0, 100, 0, 255);
  analogWrite(13, 0);
  analogWrite(14, 0);
  analogWrite(16, 0);
  analogWrite(17, pwm);
}
void tr(unsigned int speed)
{
  unsigned int pwm = map(constrain(speed, 0, 100), 0, 100, 0, 255);
  analogWrite(13, 0);
  analogWrite(14, pwm);
  analogWrite(16, 0);
  analogWrite(17, 0);
}
void ao()
{
  analogWrite(13, 0);
  analogWrite(14, 0);
  analogWrite(16, 0);
  analogWrite(17, 0);
}
```

Listing 5-2 The contactless object avoiding robot program for mikroRover robot (final)

(4) Compile and upload the code to mikroRover robot. Unplug the USB-C cable and turn off the robot.

(5) Prepare the demonstration field by placing the obstacles similar the maze.

(6) Place the robot on the floor. Turn on power. Press the SW1 for starting the robot.

mikroRover robot will moves forward and check the distance from any obstacle.

If the distance still is more than 30cm., the robot continued to move forward.

If the distance is shorter than 30cm., the robot stop 0.2 second and turn right 0.75 second. The robot possible to move avoid the obstacle as shown in the Figure 5-4

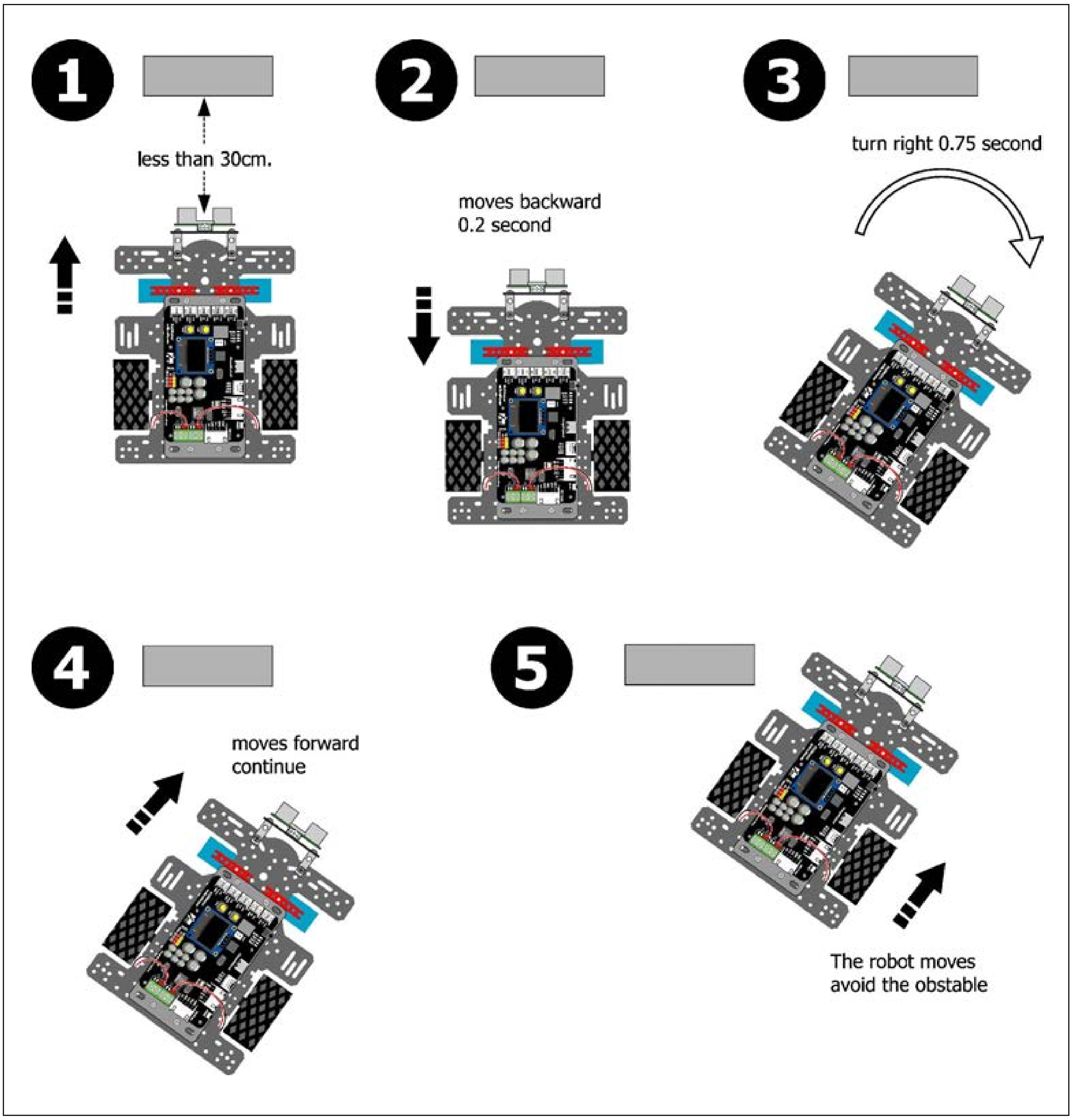


Figure 5-4 The contactless object avoiding robot operation

mikro rover Robot Kit | Moving along the line 06

One of the most popular learning of the autonomous robots is robot moving along the lines. Because it is a good example for understanding about the simple logical process. The robot's moving along the line is occurred from a logical and scientifically-explained interrelated process.

The robot must read the surface data that it must move through. Verify the color surface as white or black.

- If both robot's sensors detect a black line, how do they move ?

- If only one of the line sensor detects a black line, how should it move?

Get all conditions to set the robot's operation for moving along the line stable and do not move out of the path.

6.1 About IR reflector sensor

The mikroRover robot's line tracking sensor is made from the **Infrared reflector sensor board; ZX-03**. It has an infrared LED for emitting the infrared light and has a photo-transistor for receiving the reflected infrared light. The output voltage is depended on the intensity of reflected light. If the surface is light color like white, infrared light will reflect better than dark color or black. Figure 6-1 shows the ZX-03 IR reflector mini-board operation.

In this robot sensing, connect the ZX-03 to digital input port. Then the sensing result is as digital data "0" or "1".

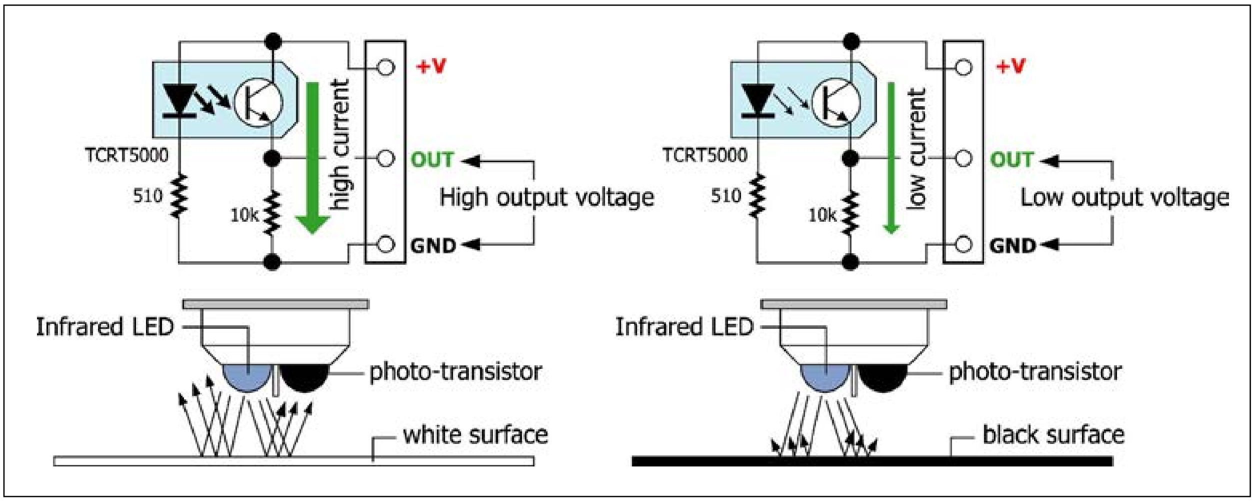


Figure 6-1 ZX-03 Infrared reflector sensor board operation

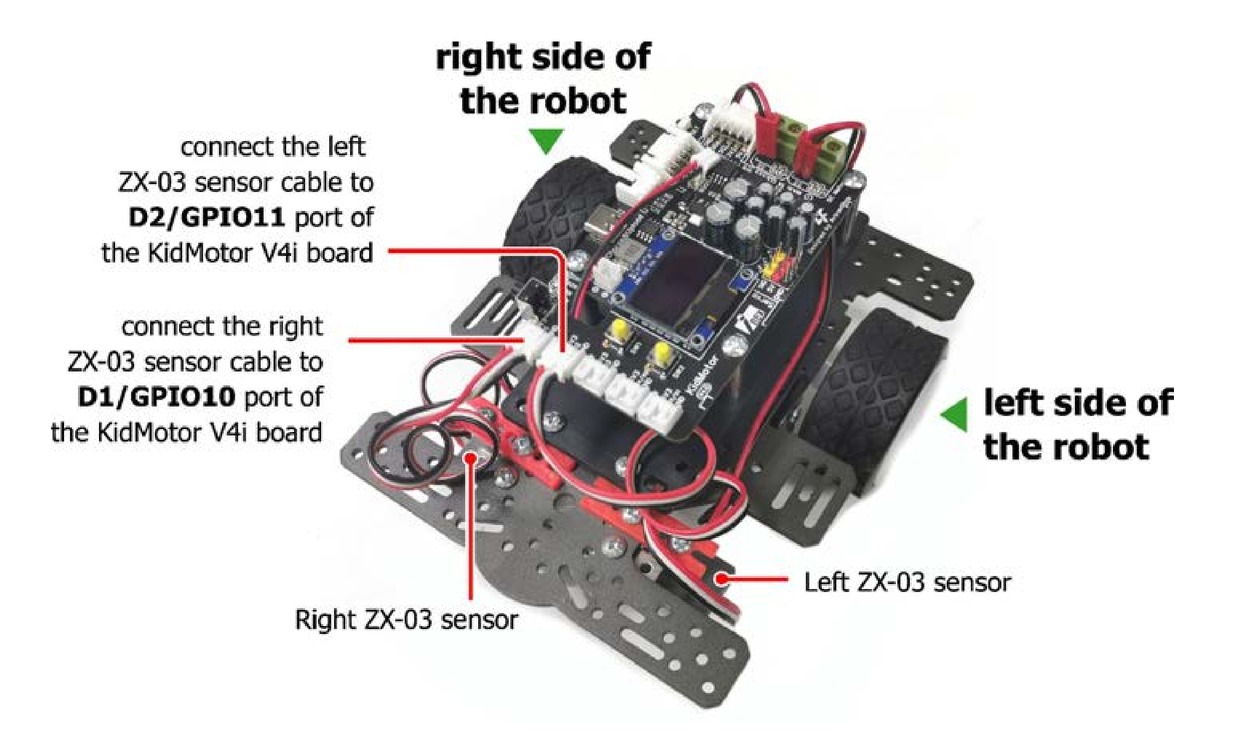


Figure 6-2 shows the line sensors configuration of the mikroRover robot

If it detects the white or light color surface, the sensor sends logical data "1" to microcontroller.

If it detects the black or dark color surface, the sensor sends logical data "0" to microcontroller.

The optimum sensing distance is 3 to 10mm. from the surface or floor.

In line tracking application, ZX-03 will be used to line tracking sensor. The connection of ZX-03 with mikroRover robot is as follows :

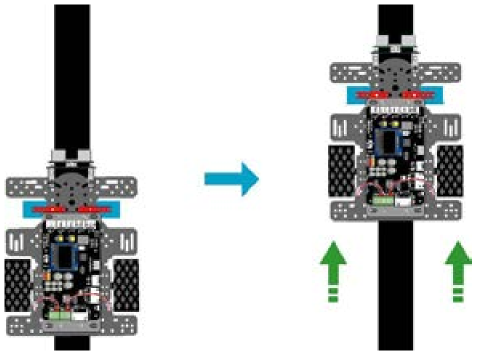
The left ZX-03 sensor is connected with D2/GPIO11 of KidMotor V4i controller board of mikroRover robot.

The right ZX-03 sensor is connected with D1/GPIO10 of KidMotor V4i controller board of mikroRover robot.

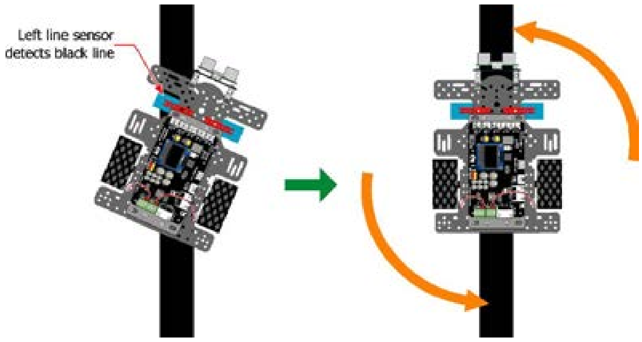
6.2 Moving along the line

There are 4 simple patterns of moving along the line of the robot as follows :

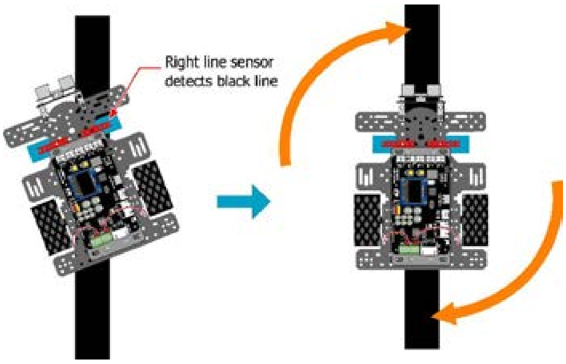
1. In case the both line sensors detect the white surface, the robot moves forward.



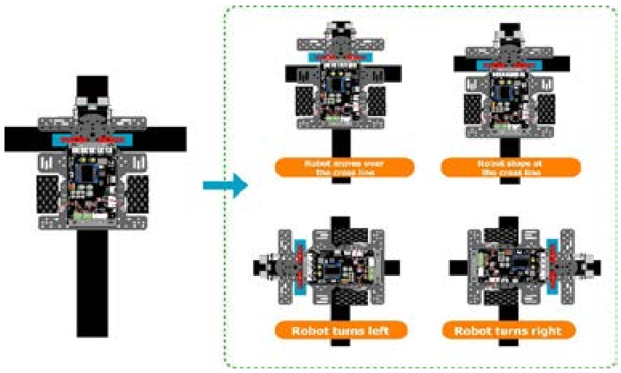
2. In case the left sensors detect the black line, it means the robot is moving out of the line on right side. The robot must spin or turn left until the robot move back to over the black line.



3. In case the right sensors detect the black line, it means the robot is moving out of the line on left side. The robot must spin or turn right until the robot move back to over the black line.



4. In case the both line sensors detect the black line, it means the robot detect the cross line or junction. There are some choices for next step : Move forward pass the cross line, Stop moving, Turn left, Turn right and more.



6.3 Making the simple line tracking robot playing field

There are material and tools used as follows.

1. The white PP flute board 90 x 60 cm. or bigger as enough for robot's movement.

2. The black electrical tape 1 inch width. 3M brand is recommended due to its high flexibility and stick to curves as well.

3. Scissor

Next is making the testing field. Put the black tape on the flute board as shown in Figure 6-3. For more accurate detection, recommended to stick the tape be overlapped as shown in Figure 6-4 to increase the width of the line to 3cm.

Figure 6-3 Example of Black line testing field for mikroRover robot

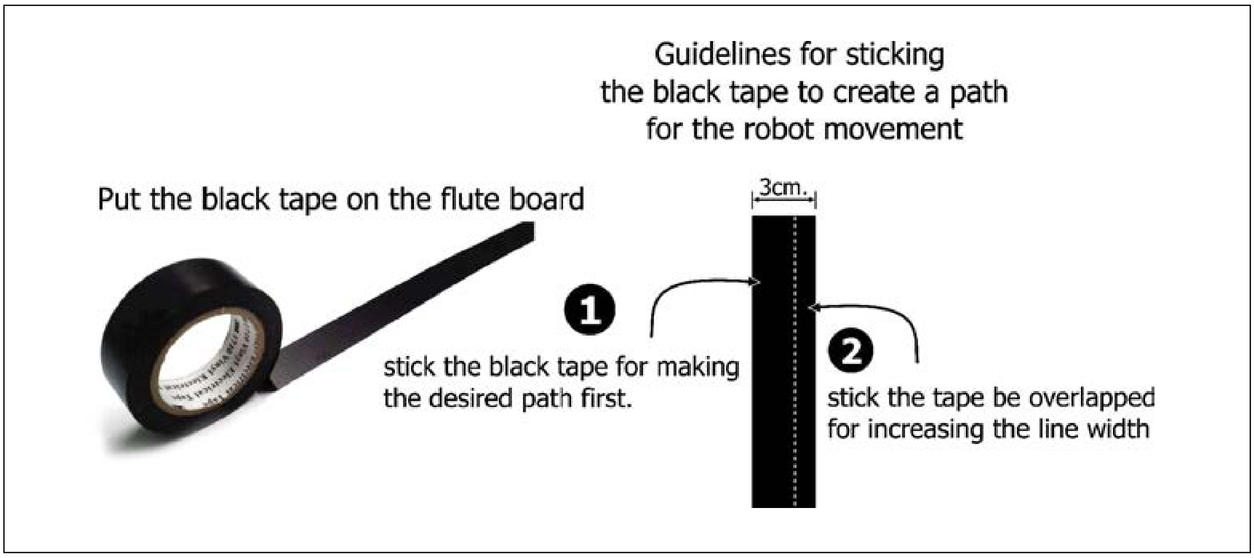


Figure 6-4 Guidelines for sticking the black tape overlapped to increase the width of the line.

6.4 About programming for Line tracking

In programming, main microcontroller will read the digital value from both line sensors at D2/GPIO11 for left sensor and D1/GPIO10 for right sensor. This is some C/C++ code for robot operation when detect the white surface or black line.

```
if (digitalRead(11) &&digitalRead(10))
// both sensors detect white surface
{
   fd(60);   // robot still move forward with 60% power
}
else if (!digitalRead(11) &&digitalRead(10))
// only the left sensor detects the black line
{
   sl(70);   // robot spin left with 70% power
}
else if (digitalRead(11) && !digitalRead(10))
// only the right sensor detects the black line
{
   sr(70);// robot spin right with 70% power
}
```

6.5 Testing the Line tracking robot

(6.5.1) Open the Arduino IDE and select the board as Raspberry Pi Pico and correct COM port.

(6.5.2) Create the code from Listing 6-1

(6.5.3) Upload the code from Listing 6-1 to the mikroRover robot. Then turn off the robot and unplug the USB-C cable

(6.5.4) Place the robot over the black line on the black line testing field. Turn on power.

```
void setup()
{
  while (digitalRead(8)); // wait for pressing the SW1 on the robot
}
void loop()
{
  if (digitalRead(11) &&digitalRead(10))
  // both line sensor detect the white surface
  {
    fd(60); // move forward continue with 60% power
  }
  else if (!digitalRead(11) &&digitalRead(10))
  // the left line sensor detects the black line
  {
    sl(70); // robot spins left with 70% power
  }
  else if (digitalRead(11) && !digitalRead(10))
  {
    sr(70); // robot spins rightt with 70% power
  }
}
void fd(unsigned int speed)
{
  unsigned int pwm = map(constrain(speed, 0, 100), 0, 100, 0, 255);
  analogWrite(13, 0);
  analogWrite(14, pwm);
  analogWrite(16, 0);
  analogWrite(17, pwm);
}
```

Listing 6-1 Simple Line tracking robot program in C/C++ for Arduino IDE (continue)

```
void bk(unsigned int speed)
{
   unsigned int pwm = map(constrain(speed, 0, 100), 0, 100, 0, 255);
   analogWrite(13, pwm);
   analogWrite(14, 0);
   analogWrite(16, pwm);
   analogWrite(17, 0);
}
void sl(unsigned int speed) {
  unsigned int pwm = map(constrain(speed, 0, 100), 0, 100, 0, 255);
analogWrite(13, pwm);
analogWrite(14, 0);
analogWrite(16, 0);
analogWrite(17, pwm);
}
void sr(unsigned int speed) {
  unsigned int pwm = map(constrain(speed, 0, 100), 0, 100, 0, 255);
analogWrite(13, 0);
analogWrite(14, pwm);
analogWrite(16, pwm);
analogWrite(17, 0);
}
void tl(unsigned int speed) {
  unsigned int pwm = map(constrain(speed, 0, 100), 0, 100, 0, 255);
analogWrite(13, 0);
analogWrite(14, 0);
analogWrite(16, 0);
analogWrite(17, pwm);
}
void tr(unsigned int speed) {
  unsigned int pwm = map(constrain(speed, 0, 100), 0, 100, 0, 255);
analogWrite(13, 0);
analogWrite(14, pwm);
analogWrite(16, 0);
analogWrite(17, 0);
}
void ao() {
analogWrite(13, 0);
analogWrite(14, 0);
analogWrite(16, 0);
analogWrite(17, 0);
}
```

Listing 6-1 Simple Line tracking robot program in C/C++ for Arduino IDE (final)

(6.5.5) Press the SW1 button on the robot. Observe the robot operation

The mikroRover robot moves along the black line continue as shown in the Figure 6-5. Watch the operation video from ***https://youtu.be/6fYv4OQRcXY***.

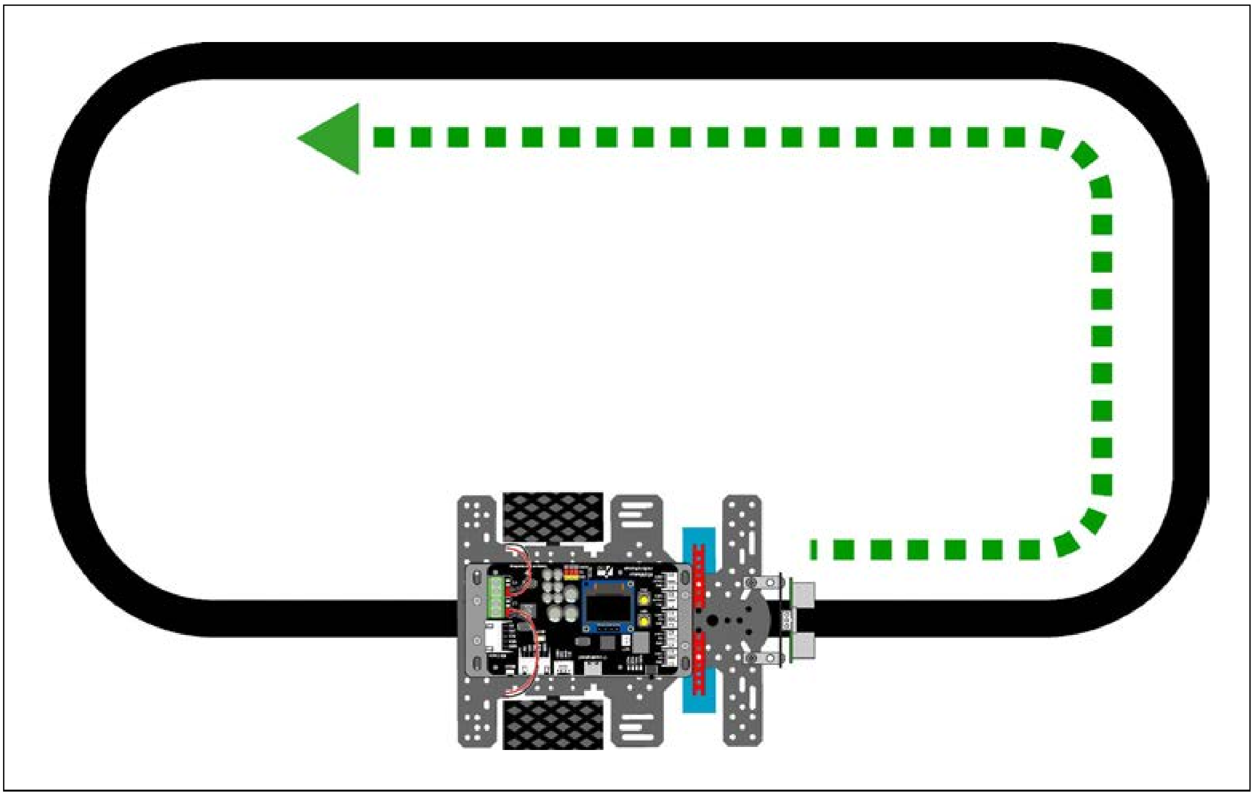


Figure 6-5 mikroRover robot's moving along the lines is from the execution of line tracking robot program following Listing 6-1

Gripper-X activity 07

7.1 Gripper-X introduction

The Gripper-X is grip and lift mechanism for small autonomous robot. The components of the Gripper-X mechanism are shown in Figure 7-1. Technical features are as follows :

● Use 2 of the metal gear servo motor with 1.5kg.cm torque assembly with the engineering plastic mechanical parts.

● Supply voltage 4.8 to 6V. Requires 100mA current consumption in normal operation and 350mA maximum

● Can be used to grip and lift any geometric shapes. The recommended object is cylindrical shape objects with a diameter of 3.5 to 9 centimeters and rectangular or more geometric shapes between 3 x 3 and 7 x 7 centimeters. Height is unlimited.

● Easy attach to the robot chassis with M3 screws and nuts.

● Working with any microcontroller board that has at least 2 servo motor outputs.

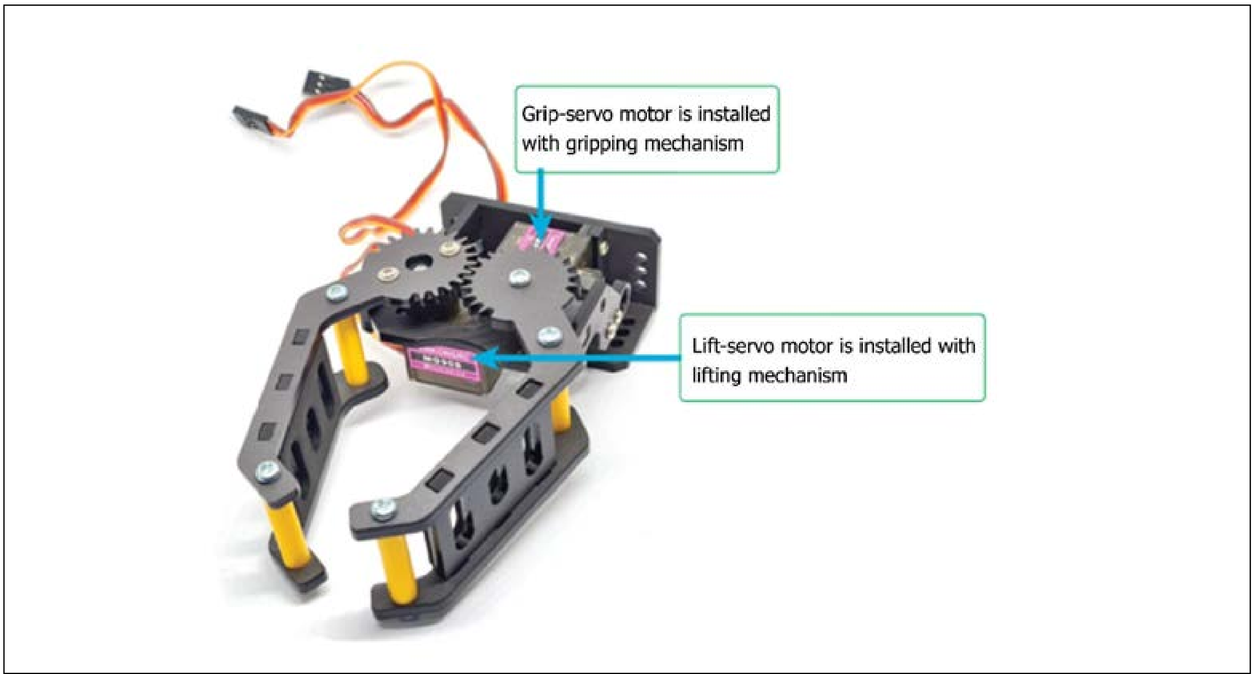


Figure 7-1 The Gripper-X component

7.2 About Gripper-X mechanism's servo motor

Summary of technical features of all servo motors of Gripper-X are as follows :

- Weight : 13.4 ¡ÃÑÁ
- Dimension : 22.5 x 12 x 35.5 mm.
- Torque : 1.8kg?cm at 4.8V supply voltage
- Rotation speed : 60 degree/0.1 second
- Supply voltage : 4.8 to 6V 100mA in normal operation and 350mA maximum current consumption.

7.3 Testing Grip function of Gripper-X

(7.3.1) Open the Arduino IDE and select the board as Raspberry Pi Pico and correct COM port.

(7.3.2) Create the code from Listing 7-1

(7.3.3) Upload the code from Listing 7-1 to the mikroRover robot. Then turn off the robot and unplug the USB-C cable

(7.3.4) Turn on the robot. Press the SW1 of the mikroRover robot to start.

(7.3.5) Press and hold the SW1. Observe the robot operation.

Grip-servo motor is driven the grip mechanism to open slowly. The OLED screen of the robot shows the servo motor shaft's position. Until the grip mechanism open the mechanism enough, record the position value.

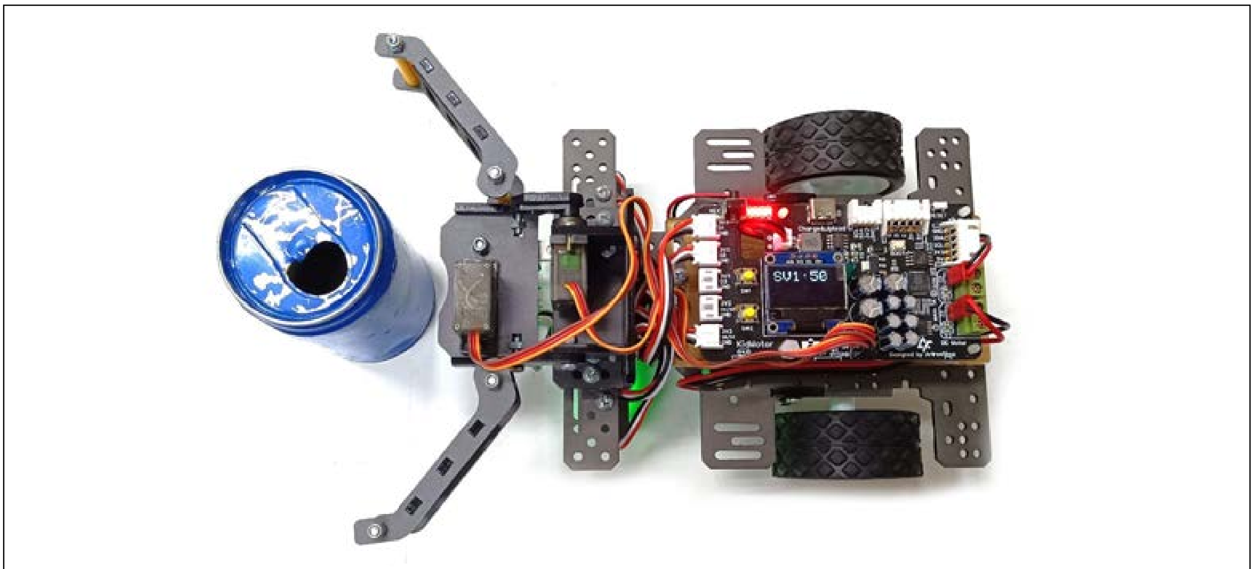


Figure 7-2 The mikroRover robot is in testing Grip function

```
#include <Servo.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 64 // OLED display height, in pixels

// Declaration for an SSD1306 display connected to I2C (SDA, SCL pins)
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);

int x = 90;
Servo sv_keep;
void setup()
{
   while (digitalRead(8));
   Serial.begin(115200);
   if (!display.begin(SSD1306_SWITCHCAPVCC, 0x3C))
   { // Address 0x3D for 128x64
     Serial.println(F("SSD1306 allocation failed"));
     for (;;);
   }
   delay(2000);
   sv_keep.attach(18);
}
void loop()
{
   display.clearDisplay();
   display.setTextSize(3);
   display.setTextColor(WHITE);
   display.setCursor(0, 0);
   // Display static text
   display.print("SV1:");
   display.println(x);
   display.display();

   if (!digitalRead(8) &&digitalRead(9) && x > 0)
   {
     x--;
   }
   else if (digitalRead(8) && !digitalRead(9) && x < 180)
   {
     x++;
   }
   sv_keep.write(x);
   delay(50);
}
```

Additional program explanation

The variable x is stored the grip-servo motor position from SW1 or SW2 button pressing. It is useful to adjust the angle position of the servo motor whether it is gripping or releasing.

Set the delay as 0.05 second to prevent damage for the grip-servo motor.

Listing 7-1 Example code for testing Grip function of Gripper-X

(7.3.6) Place the can in front of the robot as shown in the Figure 7-2. Press and hold the SW2 of the robot. Observe the robot operation.

Grip-servo motor is driven the grip mechanism to grab slowly. The OLED screen of the robot shows the servo motor shaft's position. Until the grip mechanism grabs the can completely, record the position value.

(7.3.7) Try to press the SW1 again.

Gripper-X will open the mechanism to release the can. Watch the video of this operation at https://youtu.be/XVKDXGkj530.

7.4 Testing Lift and Place function of Gripper-X

(7.4.1) Open the Arduino IDE and select the board as Raspberry Pi Pico and correct COM port.

(7.4.2) Create the code from Listing 7-2

(7.4.3) Upload the code from Listing 7-2 to the mikroRover robot. Then turn off the robot and unplug the USB-C cable

(7.4.4) Place the robot on the floor or testing filed. Turn on the robot. Press the SW1 of the mikroRover robot to start.

(7.4.5) Press and hold the SW2. Observe the robot operation.

Lift-servo motor is driven the lift mechanism to lift up slowly as shown in the Figure 7-3. The OLED screen of the robot shows the servo motor shaft's position. Until the lift mechanism lift up the mechanism enough, record the position value.

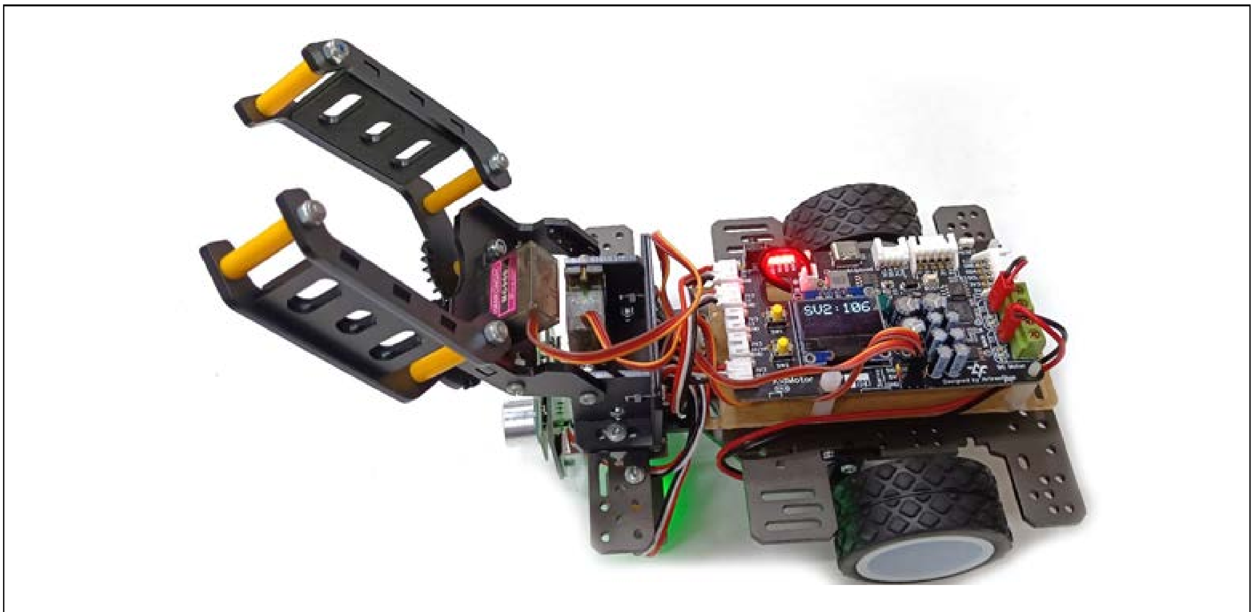


Figure 7-3 The mikroRover robot is in testing Lift function

```
#include <Servo.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 64 // OLED display height, in pixels

// Declaration for an SSD1306 display connected to I2C (SDA, SCL pins)
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);

int x = 90;
Servo sv_pick;
void setup()
{
  while (digitalRead(8));
  Serial.begin(115200);
  if (!display.begin(SSD1306_SWITCHCAPVCC, 0x3C))
  { // Address 0x3D for 128x64
    Serial.println(F("SSD1306 allocation failed"));
    for (;;);
  }
  delay(2000);
  sv_pick.attach(19);
}
void loop()
{
  display.clearDisplay();
  display.setTextSize(3);
  display.setTextColor(WHITE);
  display.setCursor(0, 0);
  // Display static text
  display.print("SV2:");
  display.println(x);
  display.display();

  if (!digitalRead(8) &&digitalRead(9) && x > 0)
  {
    x--;
  }
  else if (digitalRead(8) && !digitalRead(9) && x < 180)
  {
    x++;
  }
  sv_pick.write(x);
  delay(50);
}
```

Listing 7-2 Example code for testing Lift function of Gripper-X

(7.3.6) Press and hold the SW1 of the robot. Observe the robot operation.

Lift-servo motor is driven the lift mechanism to lift down slowly. The OLED screen of the robot also shows the servo motor shaft's position. Until the lift mechanism lift down the gripper to home position, record the home position value.

Watch the video of this operation at ***https://youtu.be/I9Aj4G7ZYXk***

7.5 Testing grab the can function

(7.5.1) Open the Arduino IDE and select the board as Raspberry Pi Pico and correct COM port.

(7.5.2) Create the code from Listing 7-3

(7.5.3) Upload the code from Listing 7-3 to the mikroRover robot. Then turn off the robot and unplug the USB-C cable

(7.5.4) Place the robot on the floor or testing filed and place the can (softdrink can 350ml.) in front of the robot.

(7.5.5) Turn on the robot. Press the SW1 of the mikroRover robot to start. Observe the robot operation.

The Gripper-X will grab the can and lift over the floor.

(7.5.6) Press the SW2 of the robot. Observe the robot operation.

The Gripper-X will place the can on the floor and release the can.

Watch the video of this operation at ***https://youtu.be/swwBweRsTmM***

```
#include <Servo.h>
Servo sv_keep, sv_pick;
void setup()
{
   sv_keep.attach(18);
   sv_pick.attach(19);
}
void loop()
{
   if (!digitalRead(8) &&digitalRead(9))
   {
      sKeep();
   }
   else if (digitalRead(8) && !digitalRead(9))
   {
      sDrop();
   }
}
void sKeep()
{
   sv_keep.write(50); delay(300);
   sv_pick.write(45); delay(300);
   sv_keep.write(112); delay(300);
   sv_pick.write(165); delay(300);
}
void sDrop()
{
   sv_pick.write(45); delay(300);
   sv_keep.write(50); delay(300);
   sv_pick.write(165); delay(300);
   sv_keep.write(112); delay(300);
}
```

Listing 7-3 Example code for testing Grab the can function of Gripper-X

7.6 Move the can mission

This demonstration is one of simple robot game. It is Move the can mission. The robot start to move along the line. Detect the crossing line then turn left. Move along the line continue to the can position. Grab and lift the can. Move to target position then place and release the can.

(7.6.1) Open the Arduino IDE and select the board as Raspberry Pi Pico and correct COM port.

(7.6.2) Create the code from Listing 7-4

(7.6.3) Upload the code from Listing 7-1 to the mikroRover robot. Then turn off the robot and unplug the USB-C cable

(7.6.4) Prepare the playing filed as shown in the Figure 7-4. Place the can at the target point.

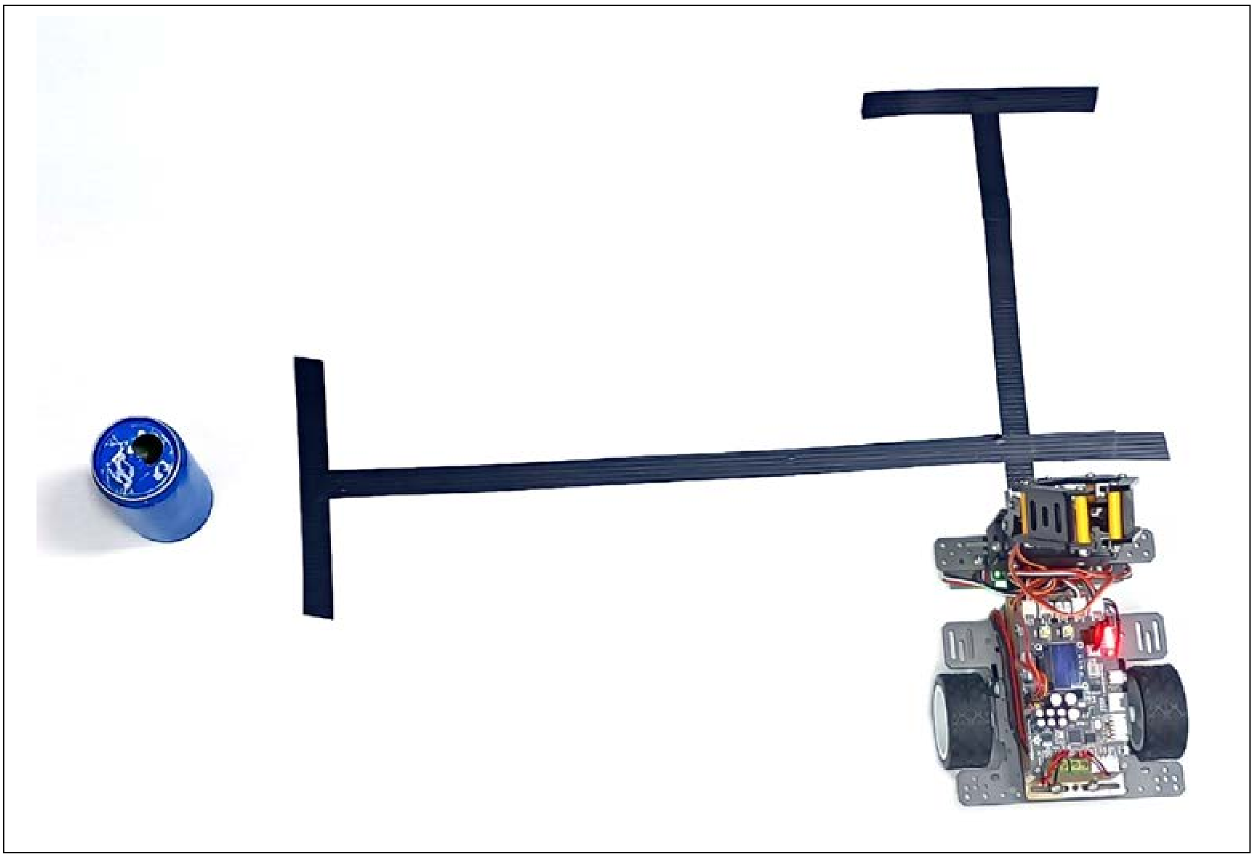


Figure 7-4 Move the can mission playing field setup.

```
#include <Servo.h>
Servo sv_keep, sv_pick;
void setup()
{
  while(digitalRead(8));

  sv_keep.attach(18);
  sv_pick.attach(19);

  cross_left();
  cross_stop();

  sKeep();

  UTURN();

  cross_left();
  cross_stop();

  sDrop();
}
void loop()
{}

void fd(unsigned int speed)
{
  unsigned int pwm = map(constrain(speed, 0, 100), 0, 100, 0, 255);
  analogWrite(13, 0);
  analogWrite(14, pwm);
  analogWrite(16, 0);
  analogWrite(17, pwm);
}
void bk(unsigned int speed)
{
  unsigned int pwm = map(constrain(speed, 0, 100), 0, 100, 0, 255);
  analogWrite(13, pwm);
  analogWrite(14, 0);
  analogWrite(16, pwm);
analogWrite(17, 0);
}
void sl(unsigned int speed)
{
  unsigned int pwm = map(constrain(speed, 0, 100), 0, 100, 0, 255);
  analogWrite(13, pwm);
  analogWrite(14, 0);
  analogWrite(16, 0);
  analogWrite(17, pwm);
}
```

Listing 7-4 Example code for Move the can mission of mikroRover robot (continue)

```
void sr(unsigned int speed)
{
  unsigned int pwm = map(constrain(speed, 0, 100), 0, 100, 0, 255);
  analogWrite(13, 0);
  analogWrite(14, pwm);
  analogWrite(16, pwm);
  analogWrite(17, 0);
}
void tl(unsigned int speed)
{
  unsigned int pwm = map(constrain(speed, 0, 100), 0, 100, 0, 255);
  analogWrite(13, 0);
  analogWrite(14, 0);
  analogWrite(16, 0);
  analogWrite(17, pwm);
}
void tr(unsigned int speed)
{
  unsigned int pwm = map(constrain(speed, 0, 100), 0, 100, 0, 255);
  analogWrite(13, 0);
  analogWrite(14, pwm);
  analogWrite(16, 0);
  analogWrite(17, 0);
}
void ao()
{
  analogWrite(13, 0);
  analogWrite(14, 0);
  analogWrite(16, 0);
  analogWrite(17, 0);
}
void sKeep()
{
  sv_keep.write(50); delay(300);
  sv_pick.write(45); delay(300);
  sv_keep.write(112); delay(300);
  sv_pick.write(165); delay(300);
}
void sDrop()
{
  sv_pick.write(45); delay(300);
  sv_keep.write(50); delay(300);
  sv_pick.write(165); delay(300);
  sv_keep.write(112); delay(300);
}
```

Listing 7-4 Example code for Move the can mission of mikroRover robot (continue)

```
void cross_pass()
{
   while (1)
   {
      if (digitalRead(10) &&digitalRead(11))
      {
         fd(60);
      }
      else if (!digitalRead(10) &&digitalRead(11))
      {
         sl(70);
      }
      else if (digitalRead(10) && !digitalRead(11))
      {
         sr(70);
      }
      else if (!digitalRead(10) && !digitalRead(11))
      {
         fd(60);
         delay(200);
         break;
      }
   }
}
void cross_stop()
{
   while (1)
   {
      if (digitalRead(10) &&digitalRead(11))
      {
         fd(60);
      }
      else if (!digitalRead(10) &&digitalRead(11))
      {
         sl(70);
      }
      else if (digitalRead(10) && !digitalRead(11))
      {
         sr(70);
      }
      else if (!digitalRead(10) && !digitalRead(11))
      {
         ao();
         break;
      }
   }
}
```

Listing 7-4 Example code for Move the can mission of mikroRover robot (continue)

```
void cross_left()
{
   while (1)
   {
      if (digitalRead(10) &&digitalRead(11))
      {
         fd(60);
      }
      else if (!digitalRead(10) &&digitalRead(11))
      {
         sl(70);
      }
      else if (digitalRead(10) && !digitalRead(11))
      {
         sr(70);
      }
      else if (!digitalRead(10) && !digitalRead(11))
      {
         fd(60);
         delay(150);
         L90();
         break;
      }
   }
}
void cross_right()
{
   while (1)
   {
      if (digitalRead(10) &&digitalRead(11))
      {
         fd(60);
      }
      else if (!digitalRead(10) &&digitalRead(11))
      {
         sl(70);
      }
      else if (digitalRead(10) && !digitalRead(11))
      {
         sr(70);
      }
      else if (!digitalRead(10) && !digitalRead(11))
      {
         fd(60);
         delay(150);
         R90();
         break;
      }
   }
}
```

Listing 7-4 Example code for Move the can mission of mikroRover robot (continue)

```
void R90()
{
   sr(70);
   while(digitalRead(11));
   while(!digitalRead(11));
   ao();
}
void L90()
{
   sl(70);
   while(digitalRead(11));
   while(!digitalRead(11));
   ao();
}
void UTURN()
{
   bk(70);
   delay(300);
   R90();
}
```

Listing 7-4 Example code for Move the can mission of mikroRover robot (final)

(7.6.5) Place the robot at the start point. Turn on the robot. Press the SW1 of the mikroRover robot to start. Observe the robot operation.

Step 1 : The mikroRover robot starts to moves along the line then turn left when detect the first crossing line.

Step 2 : The mikroRover robot still moves along the line to the Target point.

Step 3 : At the Target point, the robot opens the gripper then grabs the can following lifts it.

Step 4 : The robot moves backward and turn around to moves along the line continue.

Step 5 : The robot turns left when detects the next crossing line.

Step 6 : The robot moves along the line to the Destination point.

Step 7 : The robot reaches to the Destination point. Place the can and release the gripper. Stop moving. All are done.

Watch the video of this operation at ***https://youtu.be/o9JHBj—MBE***