Smarthon Documentation

Release 1.0

Smarthon Limited

Sep 03, 2022

CONTENTS

CHAPTER

ONE

TUTORIAL GUIDE

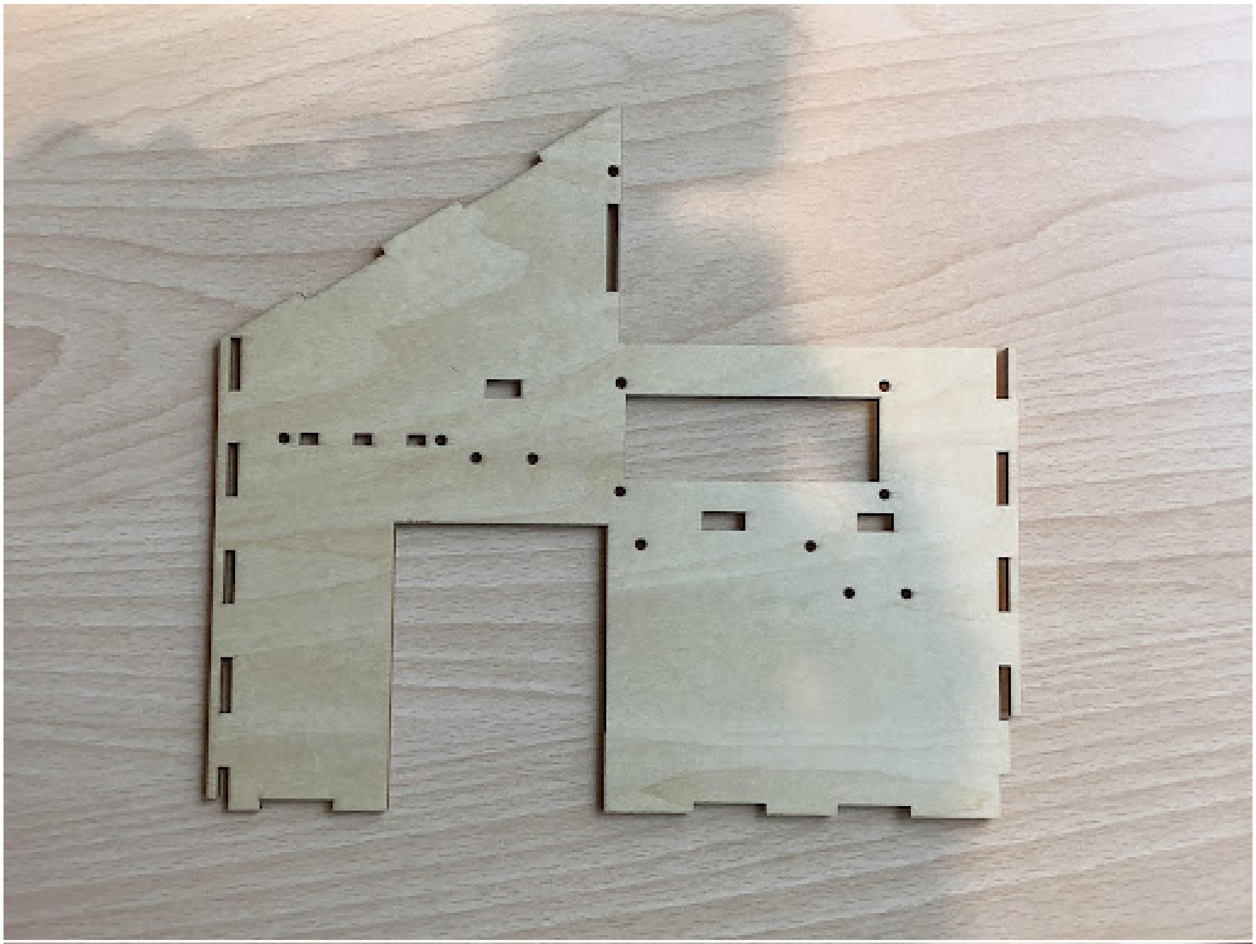
1.1 Smarthon Smart House Kit for micro:bit

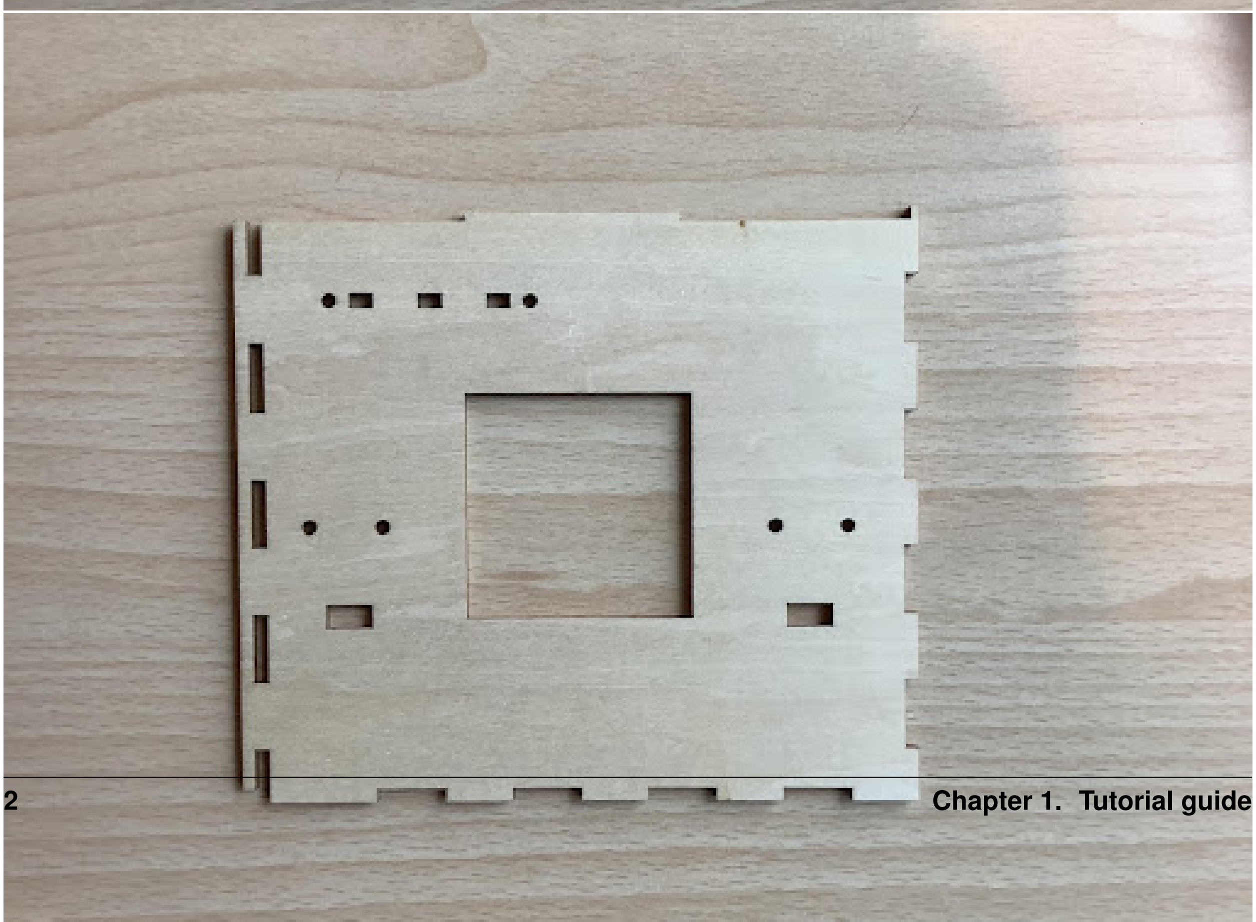
1.1.1 Chapter 1 Know More About Smart Home

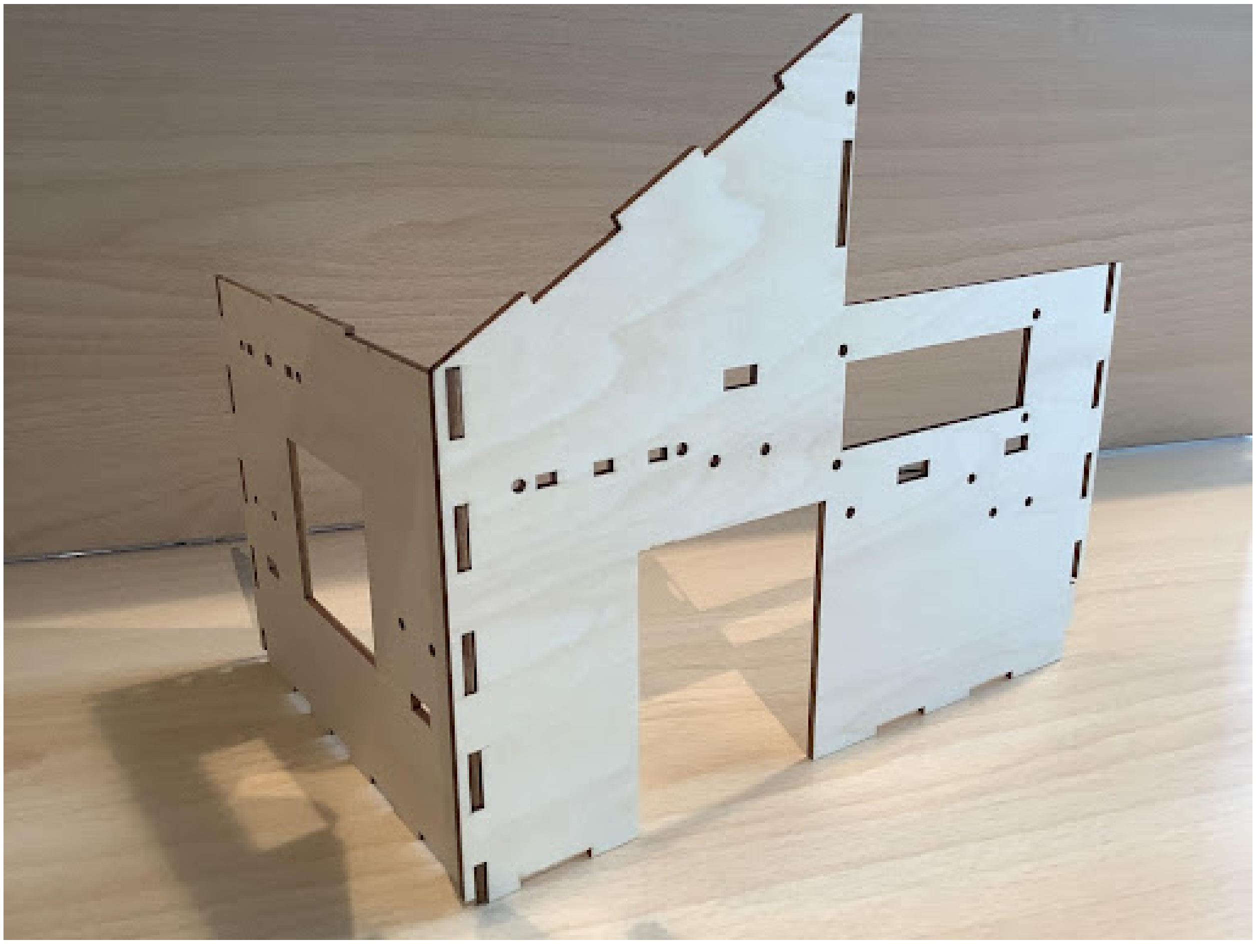
What is Smart Home

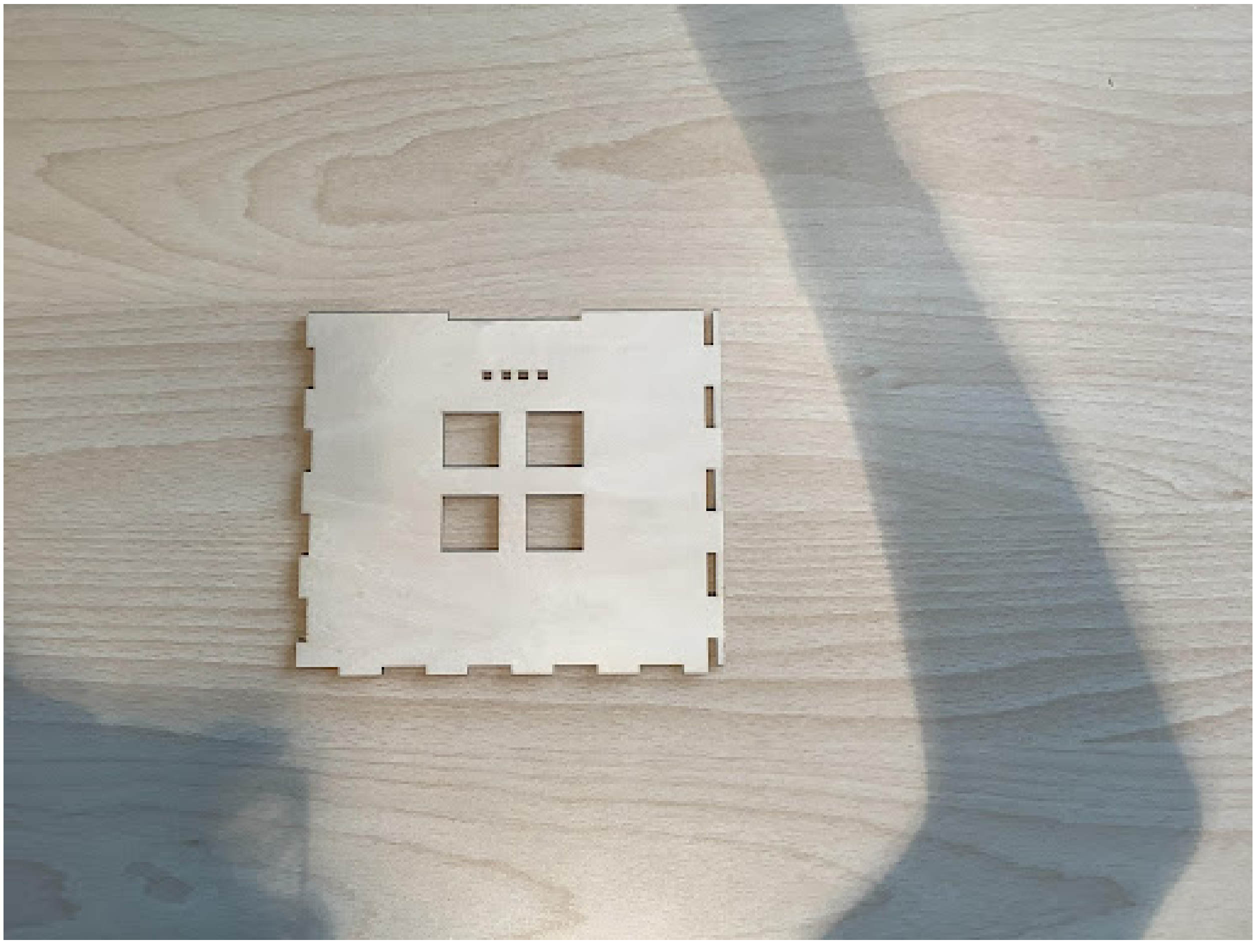
Advantages of Smart Home

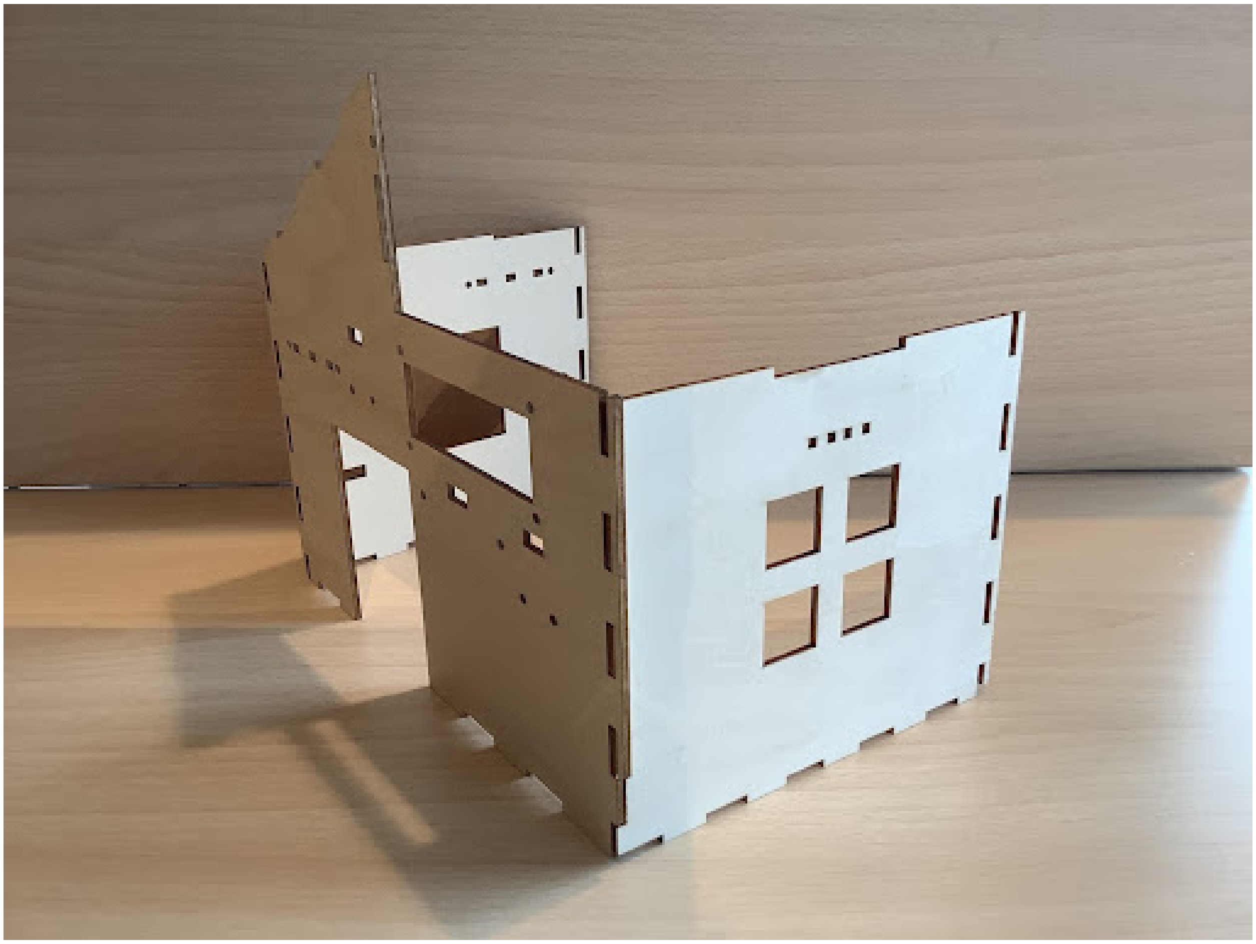
Steps of Building Smart Home Model

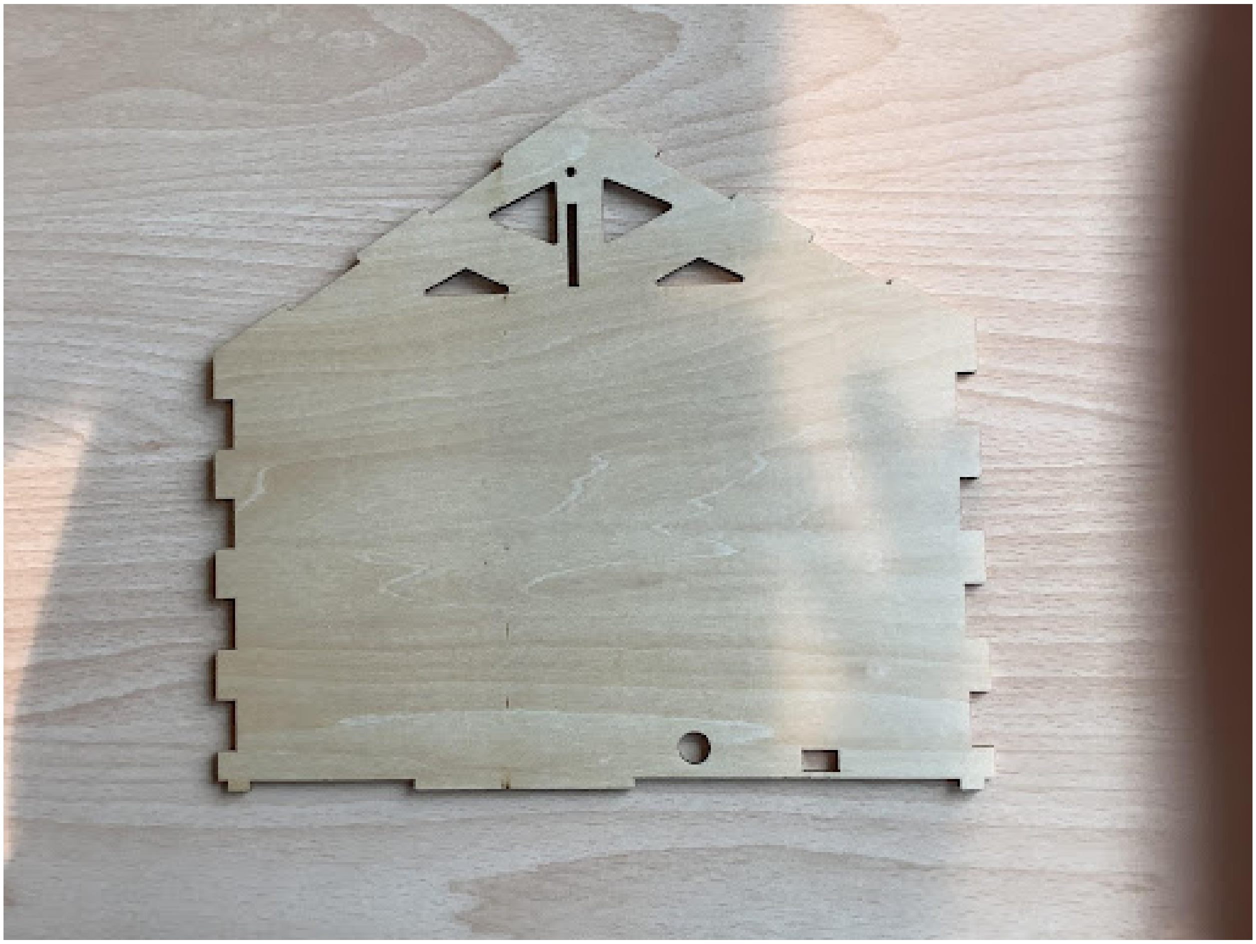


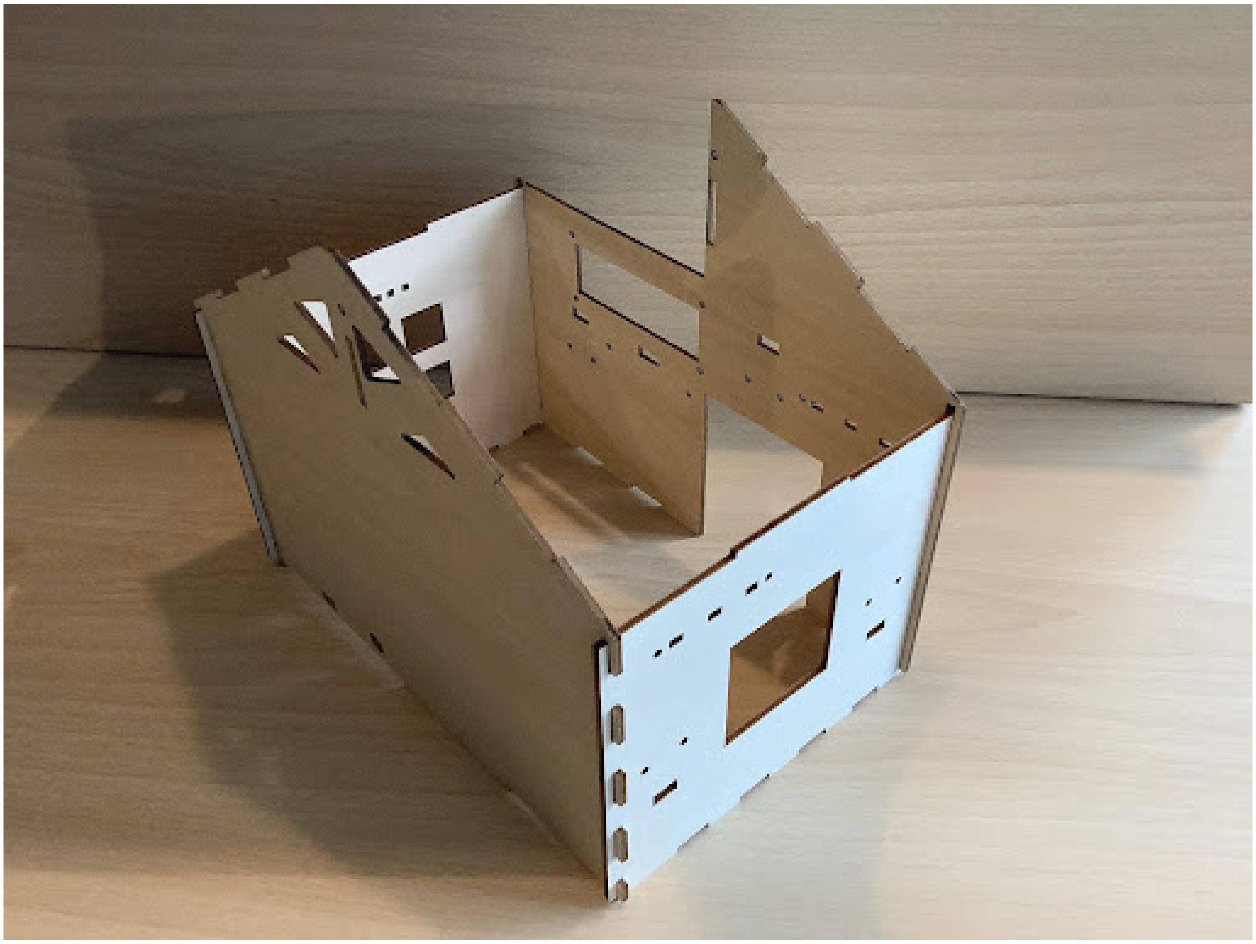


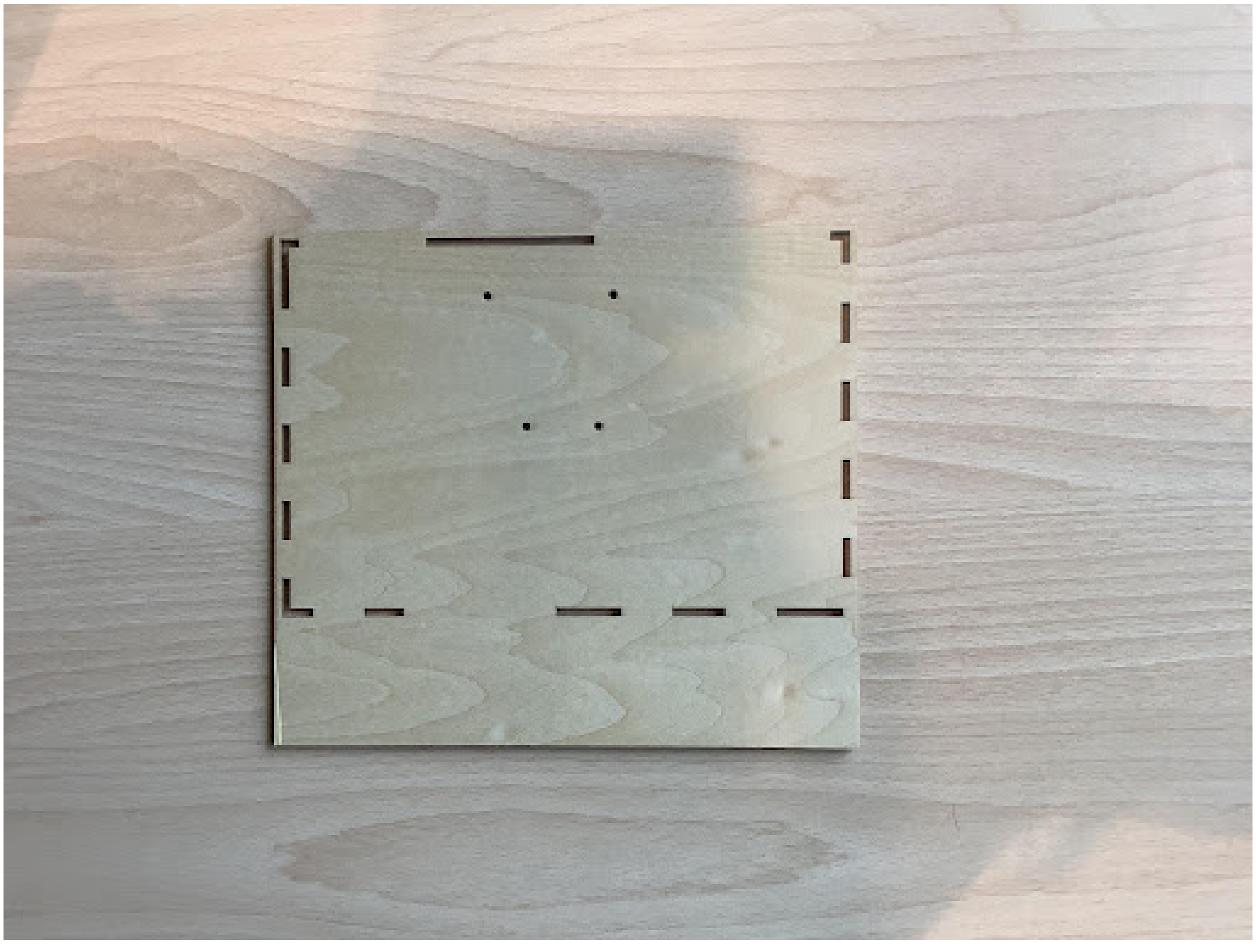


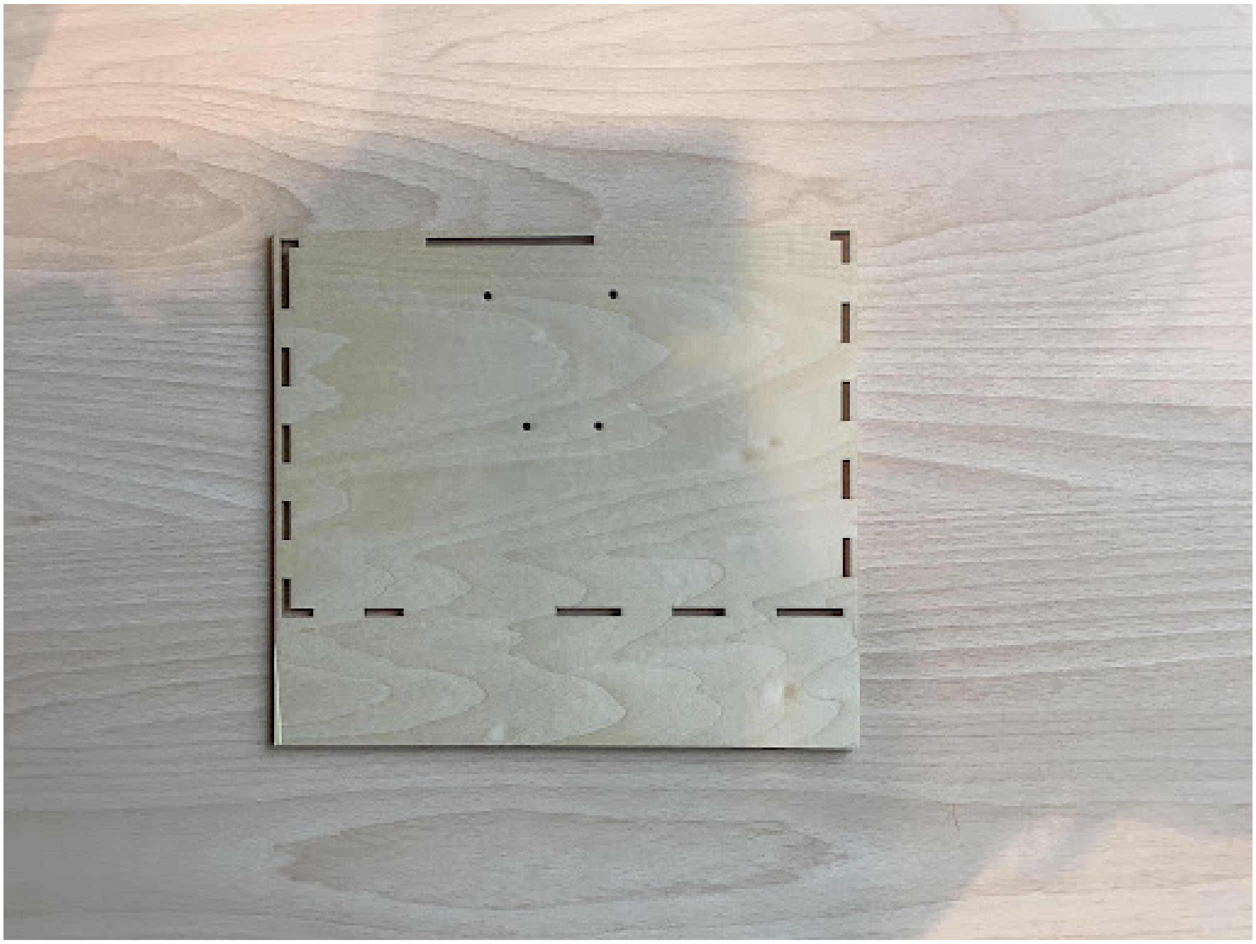


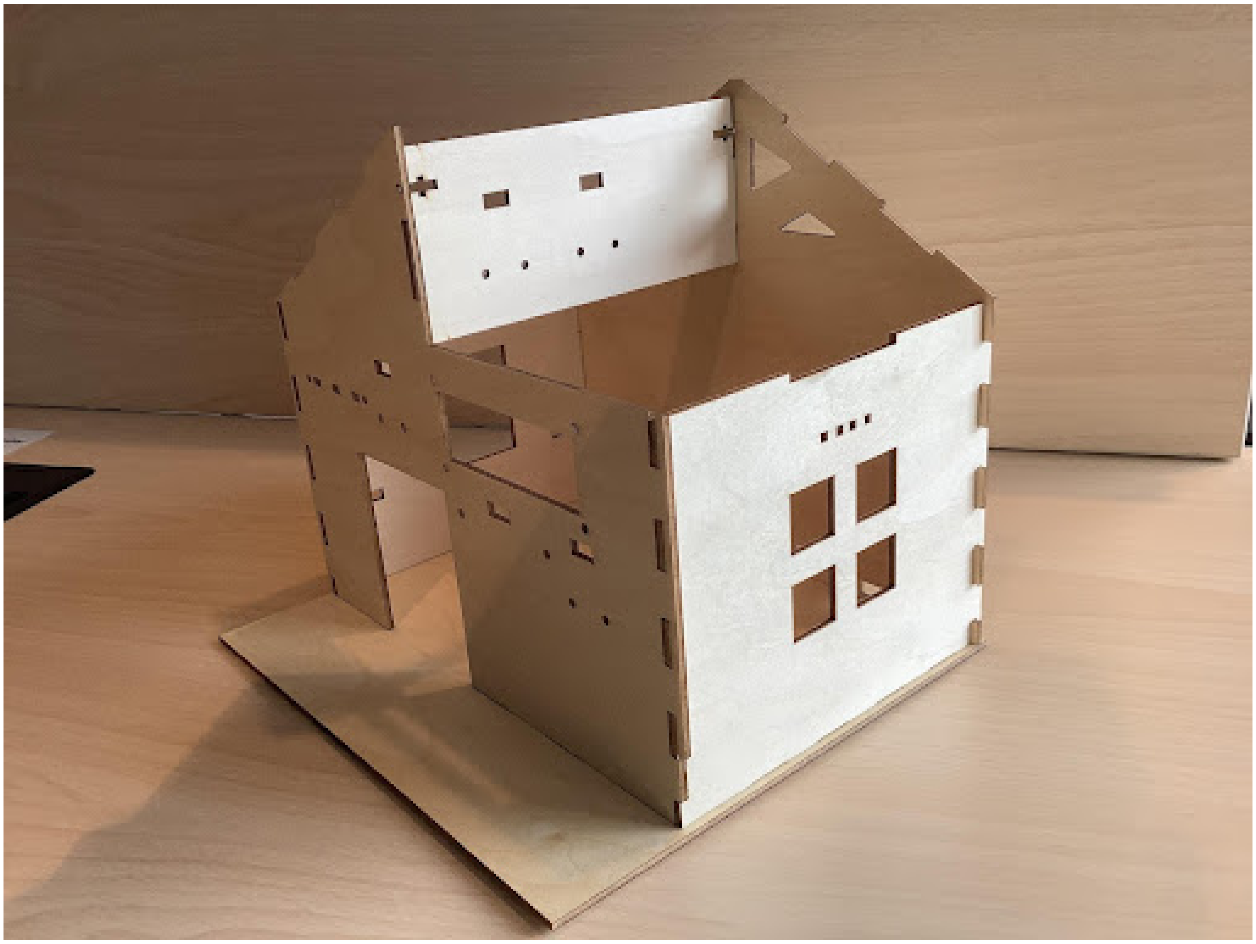


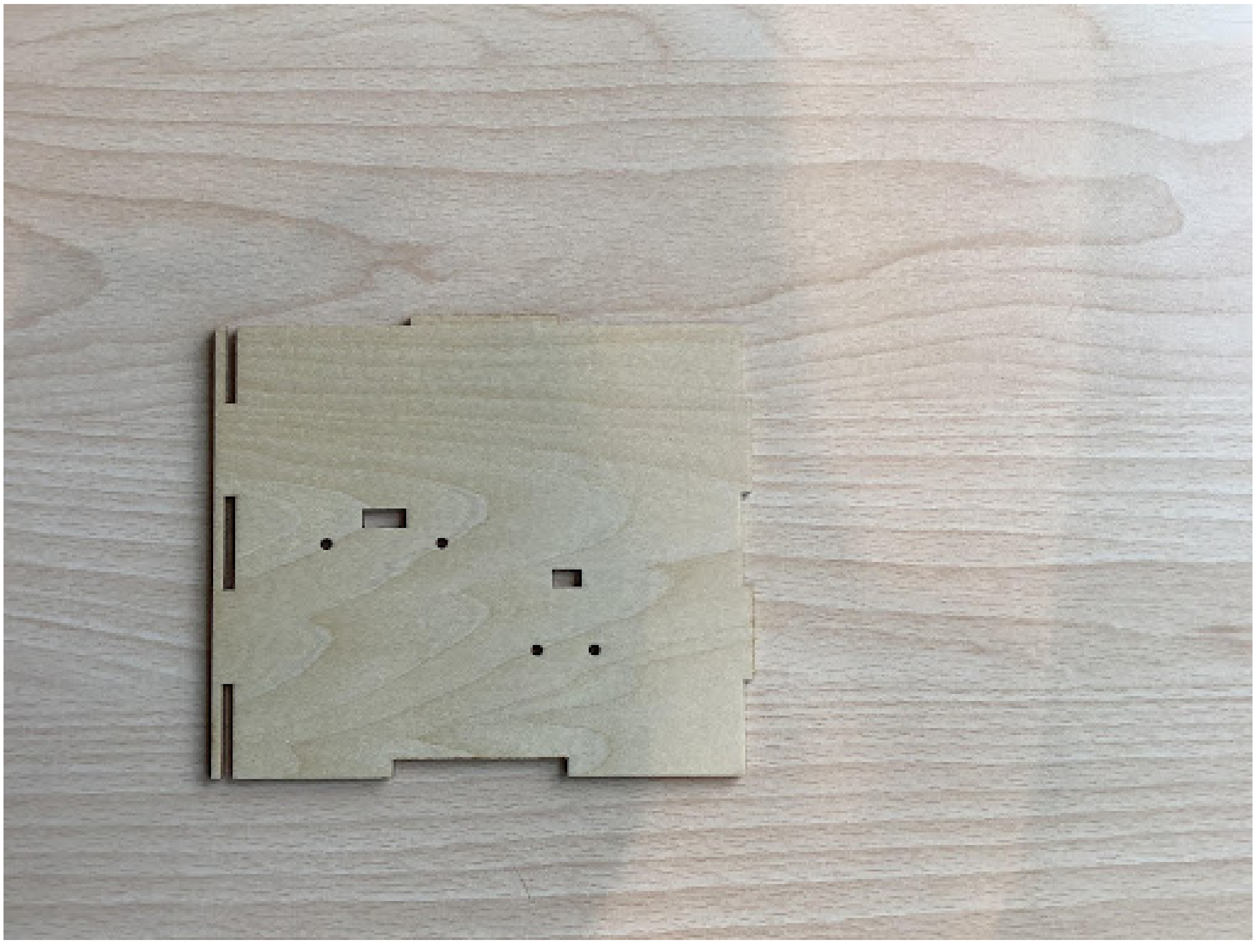


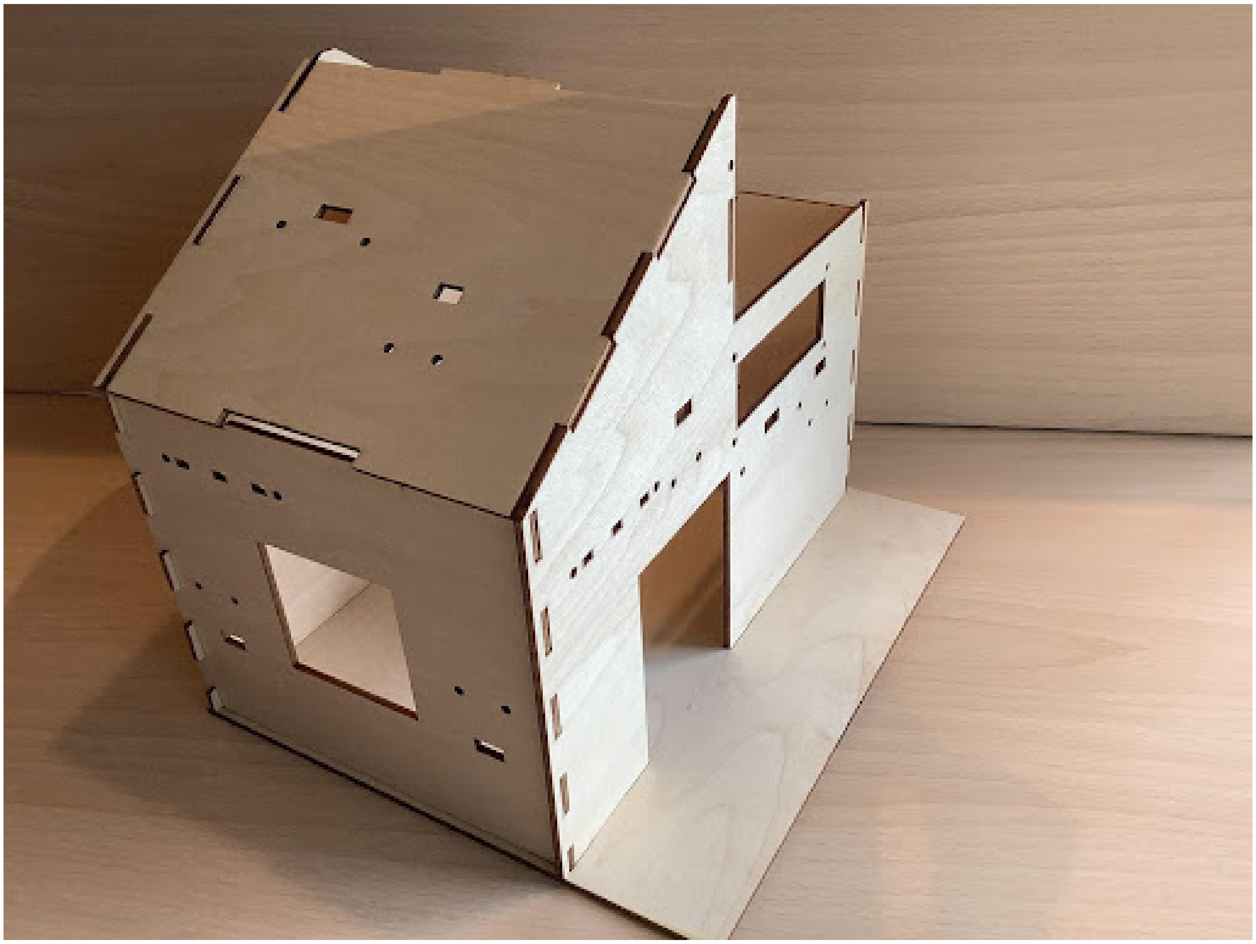




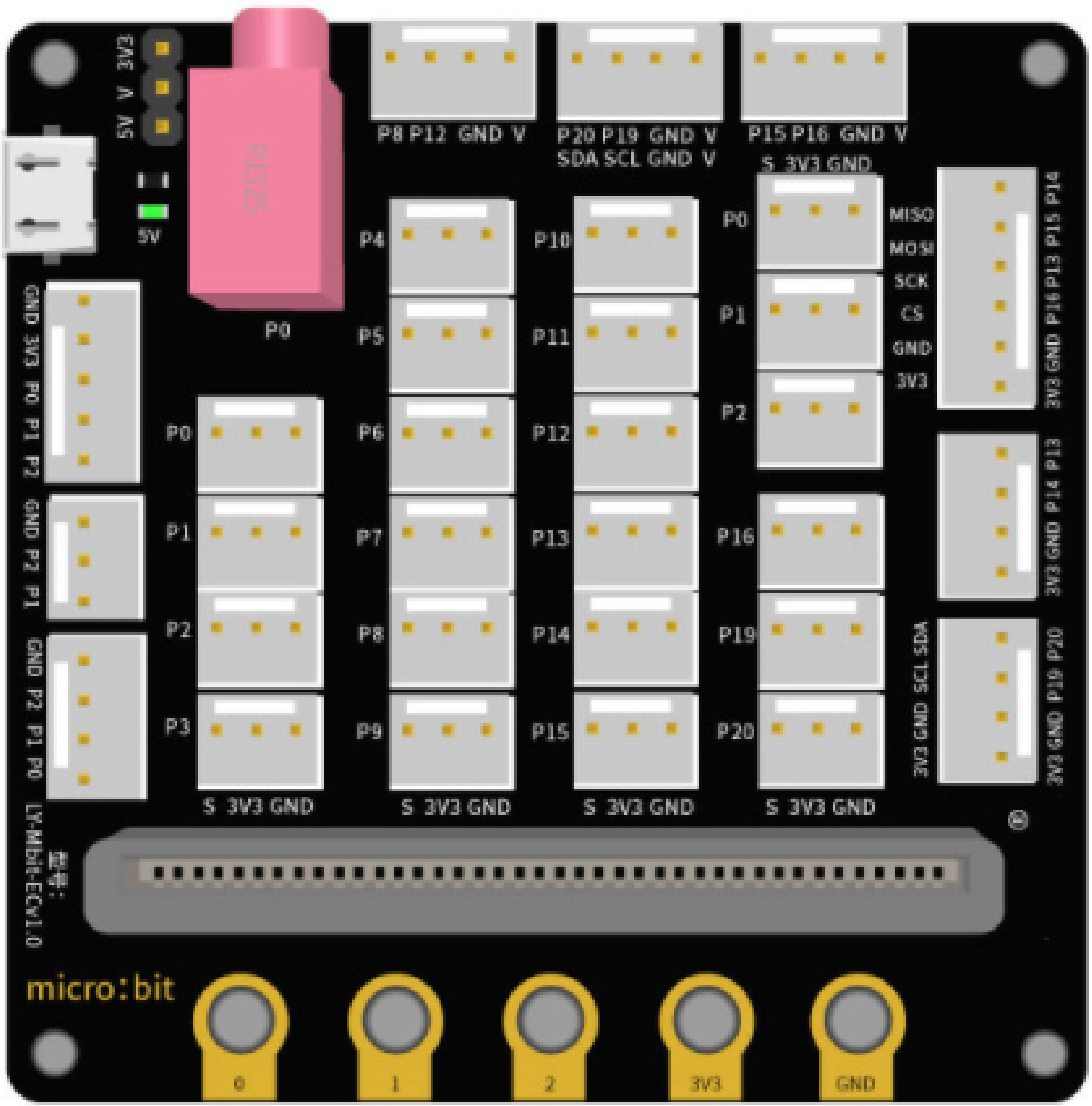




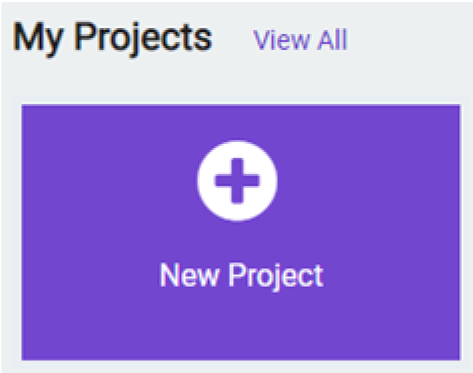


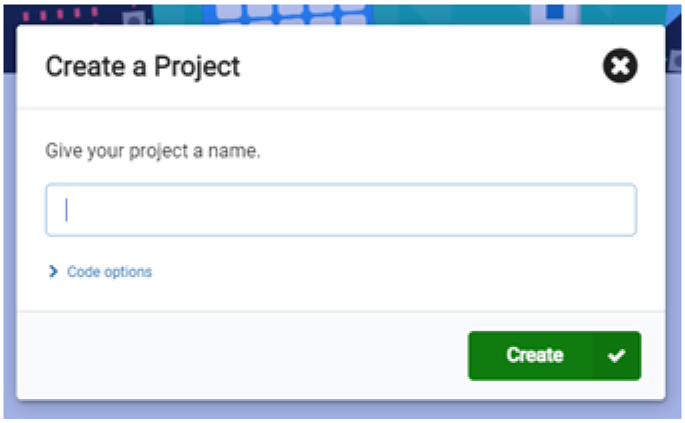


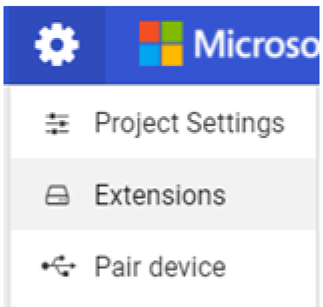
Learn About Micro:bit Expansion Board

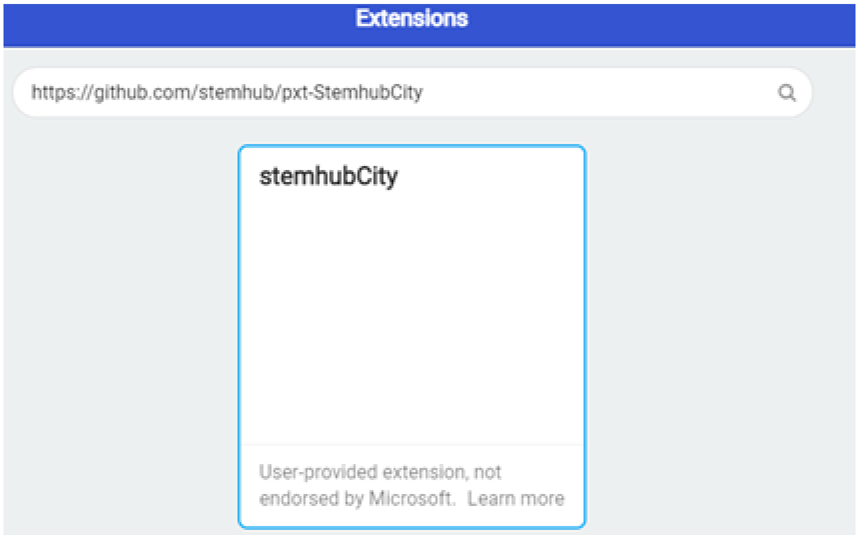


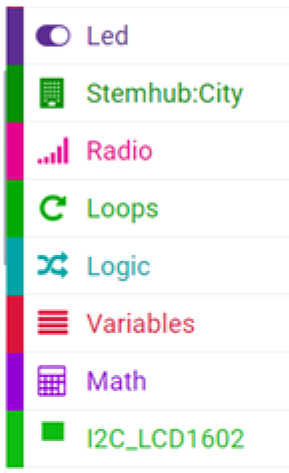
Prepare Micro:bit Programming MakeCode: Add Extensions











1.1.2 Chapter 2 Smart Human Body Induction Lamp

Background

Preparation

Learn About Smart Body Induction Lamp

Learn About Human Sensor Module and the Red and Green Light Module

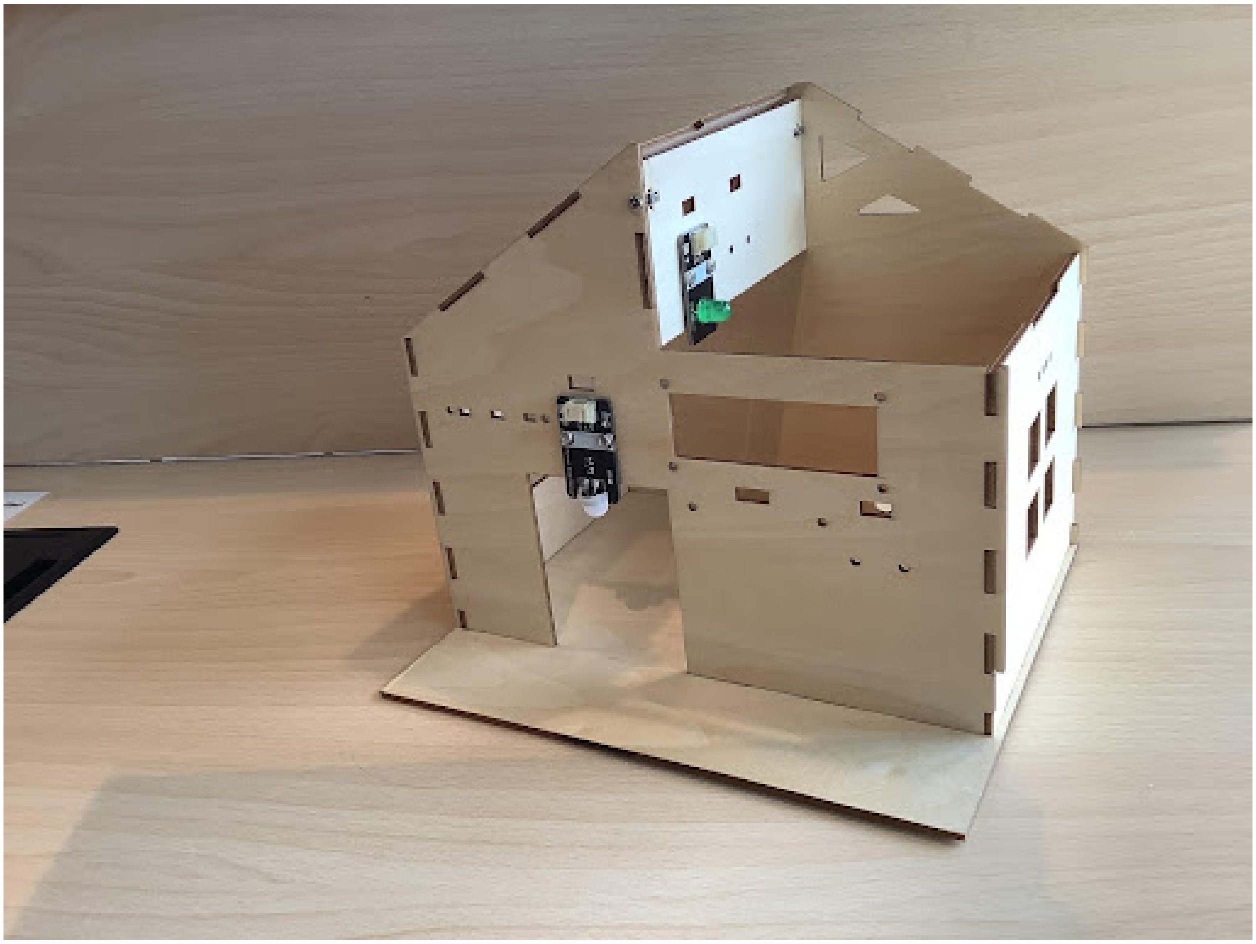
Human Body Sensor Module

Fully automatic sensing: high level is output when a person enters its sensing range, and low level is output when the person leaves the sensing range with automatic delay to turn off high level.

Repeatable triggering method: After the induction output is in high level, if a person is sensed in the induction area during the delay time period, the output will remain high level until the person leaves and then the high level will be changed to low level (the induction module will automatically delay a delay time period after detecting each human activity, and the time of the last activity will be the starting point of the delay time HC-SR505). The small human body sensor module has three pins, G for GND ground, V for VCC high level or 5v, S is the signal pin.

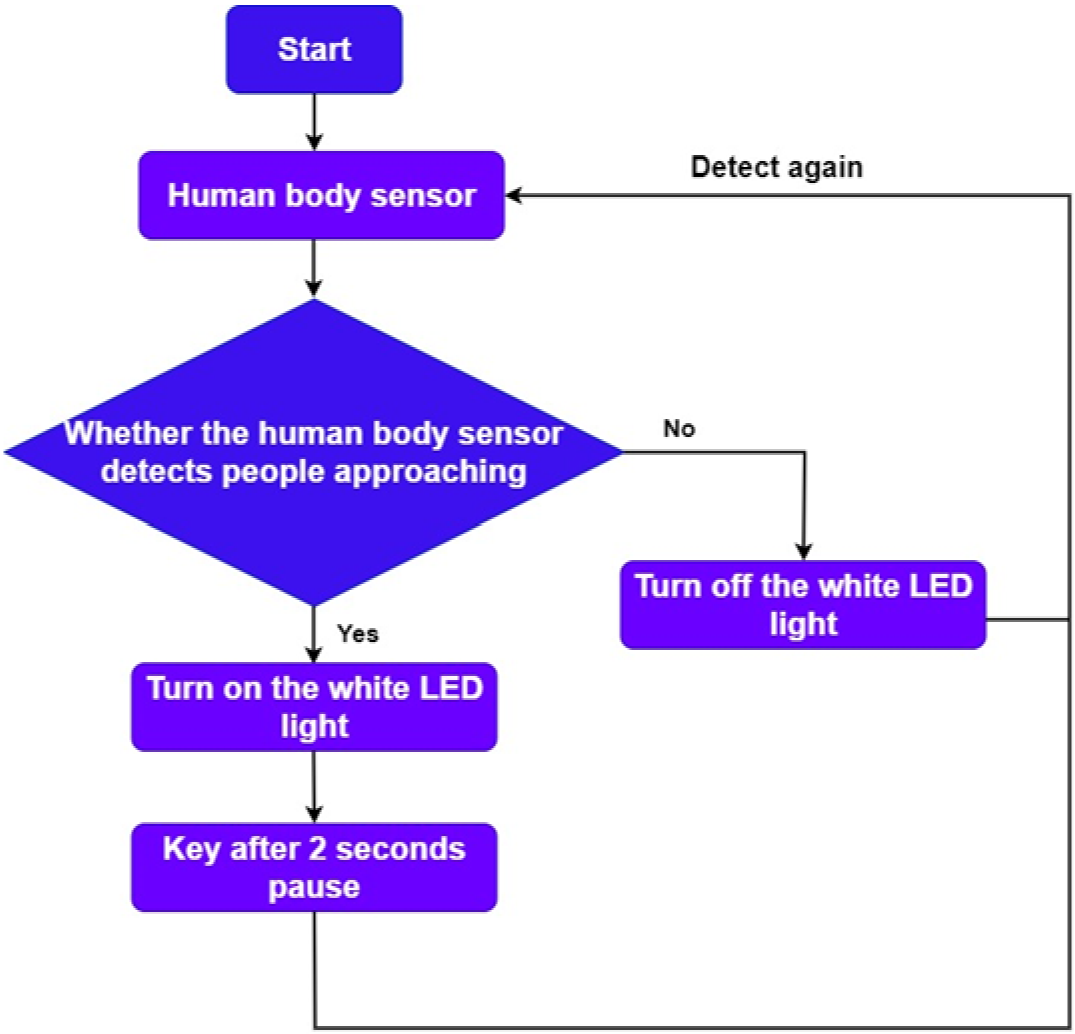
White LED light module

Installation of Human Body Sensor Light

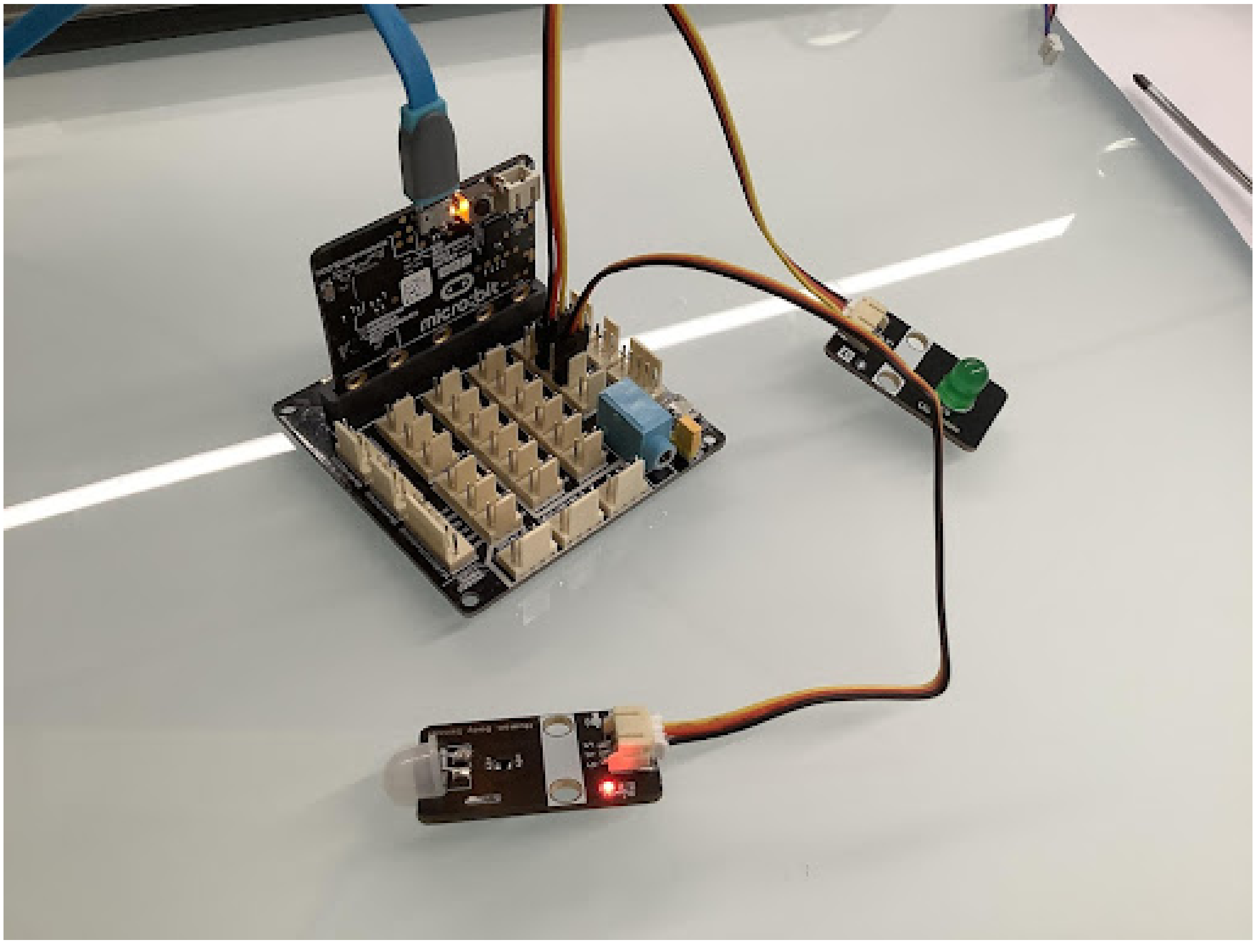


Program Design

Algorithm Design

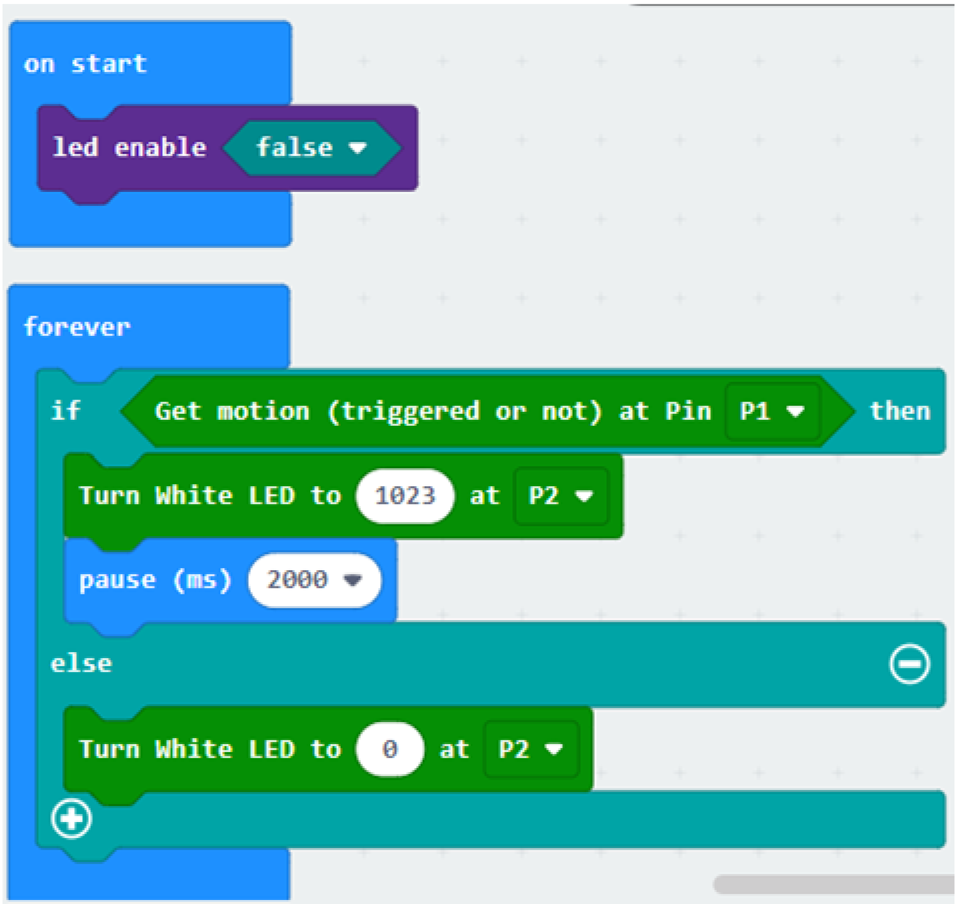


Hardware Connection



Sample Program

Makecode program



Conclusion

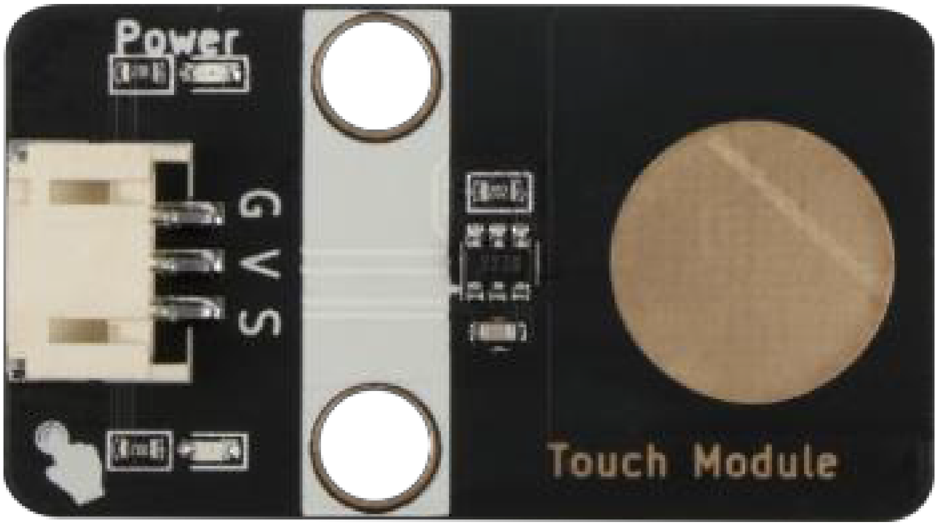
1.1.3 Chapter 3 Music Doorbell

Background

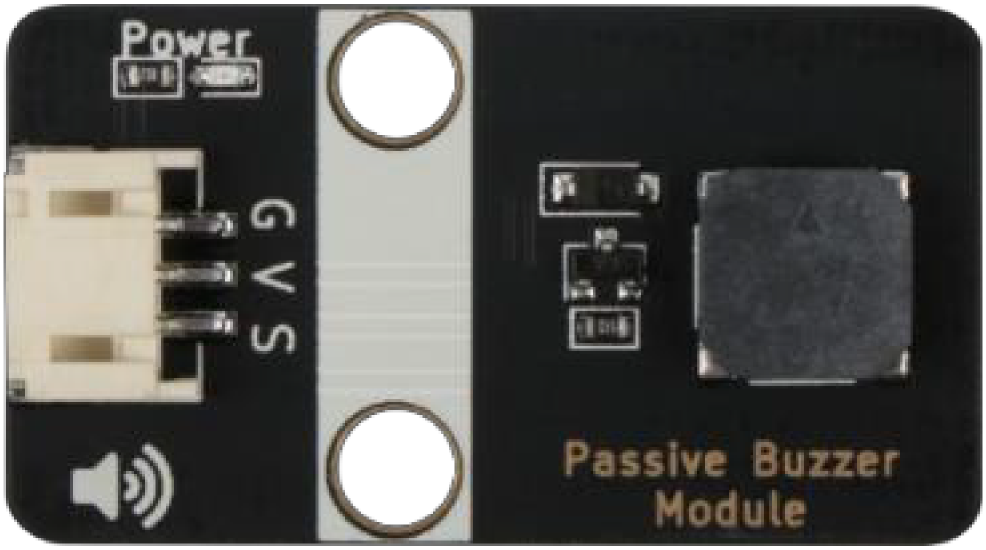
Preparation

Learn AboutTouch Sensors and Passive Buzzers

Touch Sensor

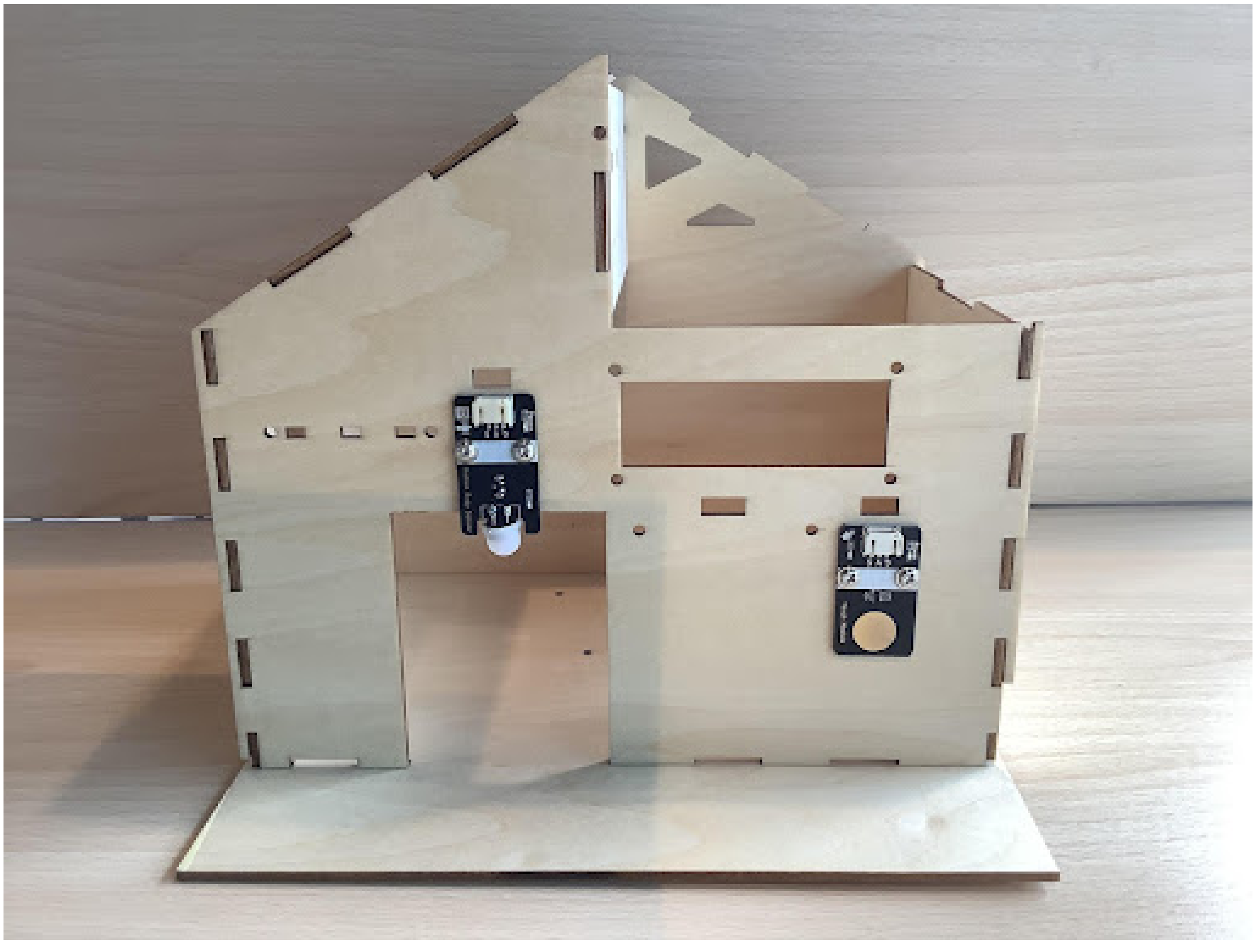


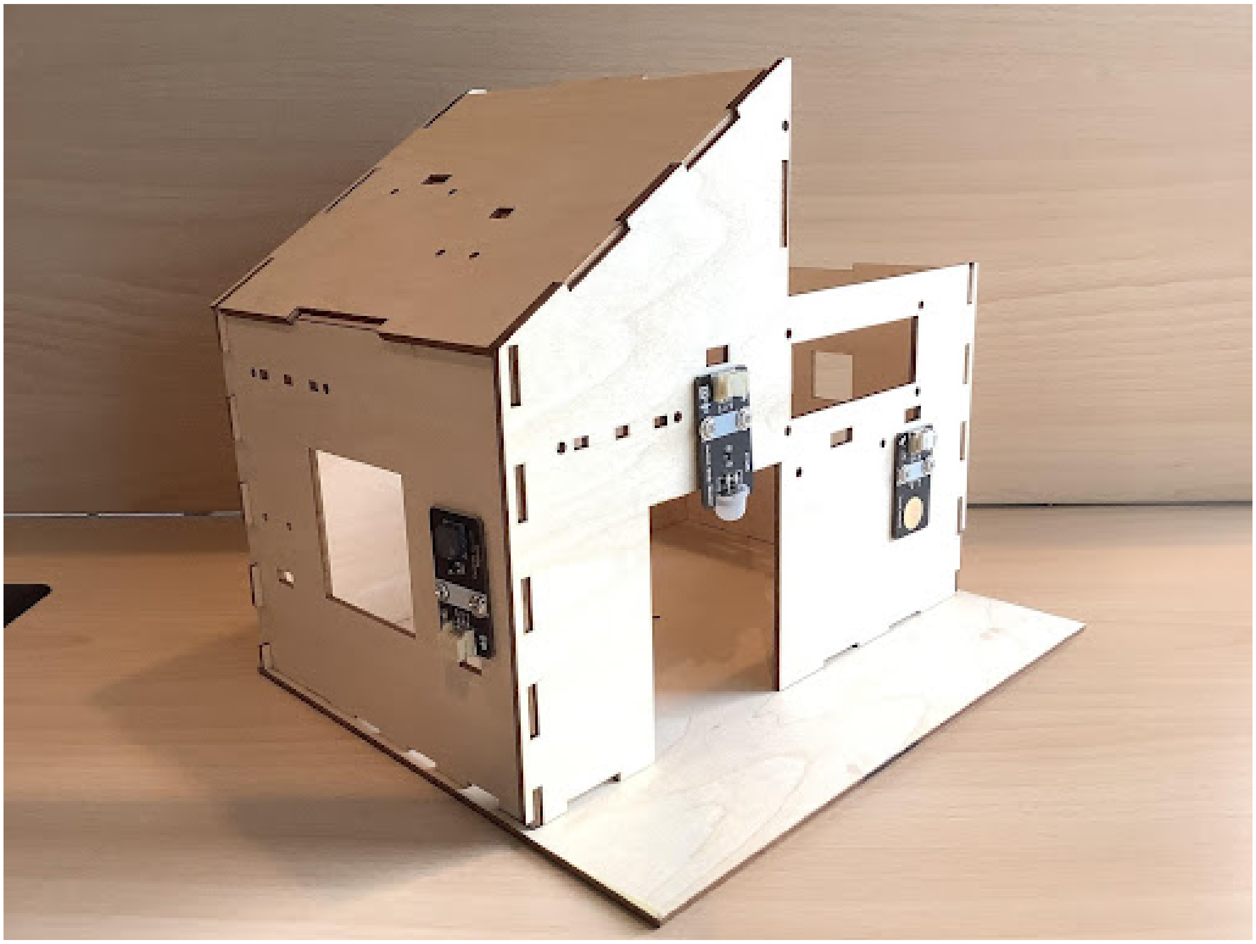
Passive buzzer module



Buzzer	BLE-UNO Main Board
G	GND
V	VCC、5V、3.3V
S	D0-D13

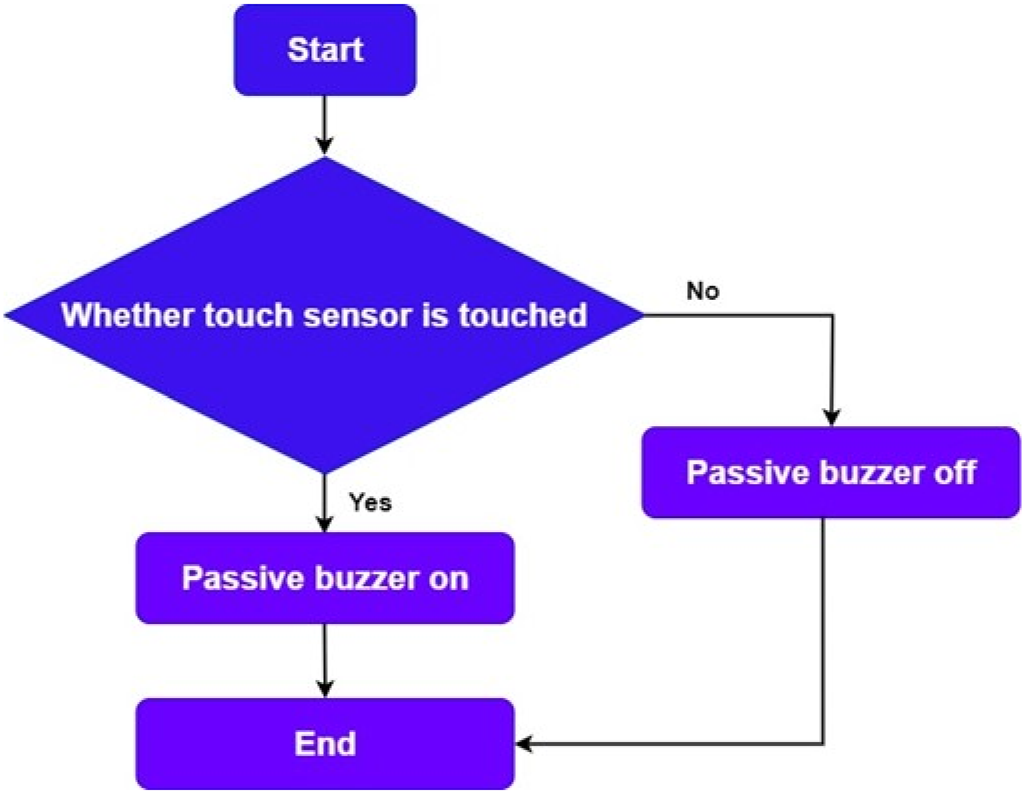
Installation of Doorbell



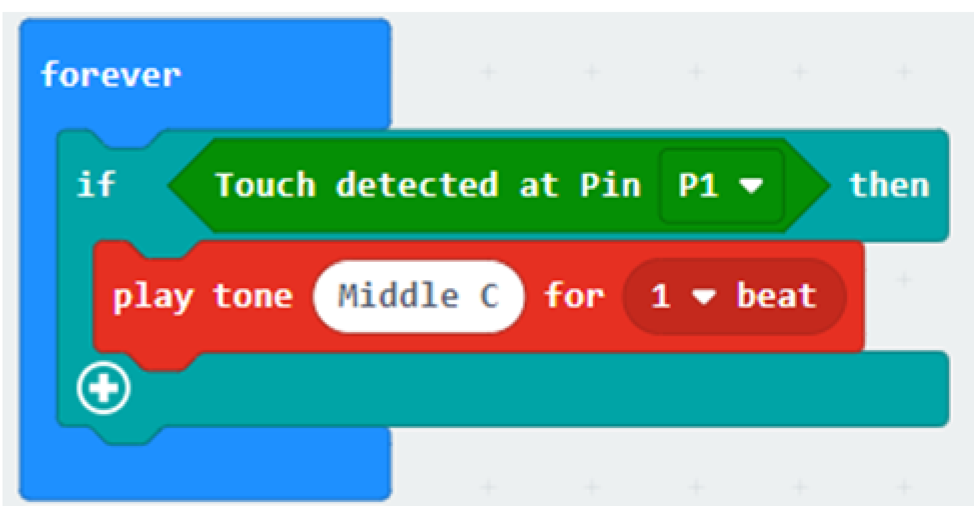


Program Design

Algorithm Design



Hardware Connection



Conclusion

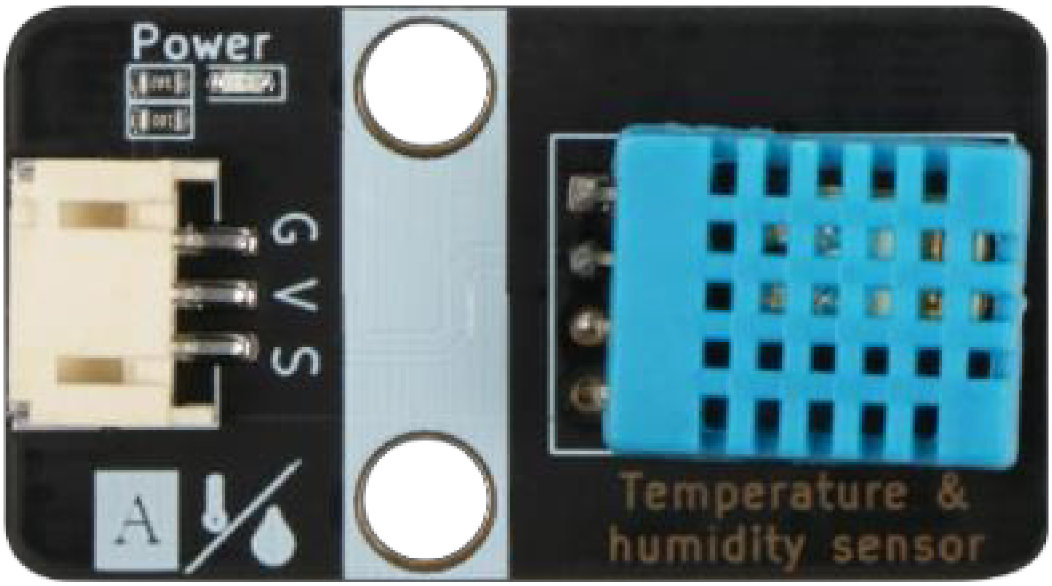
1.1.4 Chapter 4 Smart Temperature Control Fan

Background

Preparation

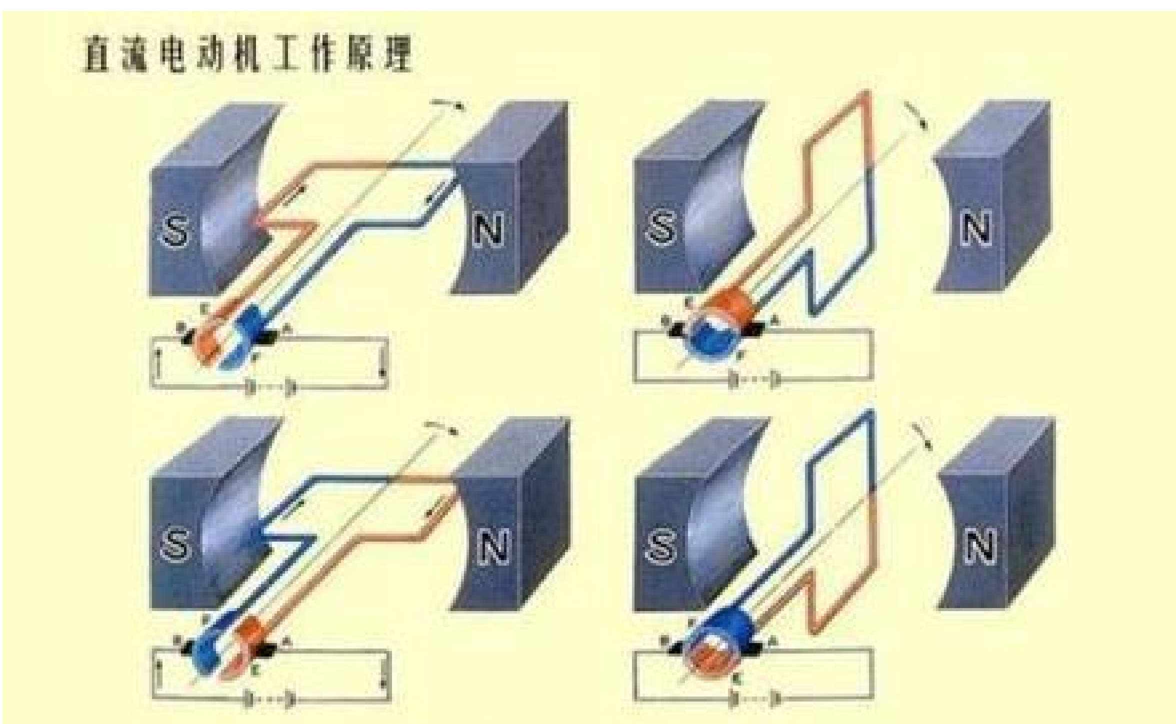
Learn About Temperature And Humidity Sensors And DC Motor Fan Modules

Temperature and Humidity Sensor Module



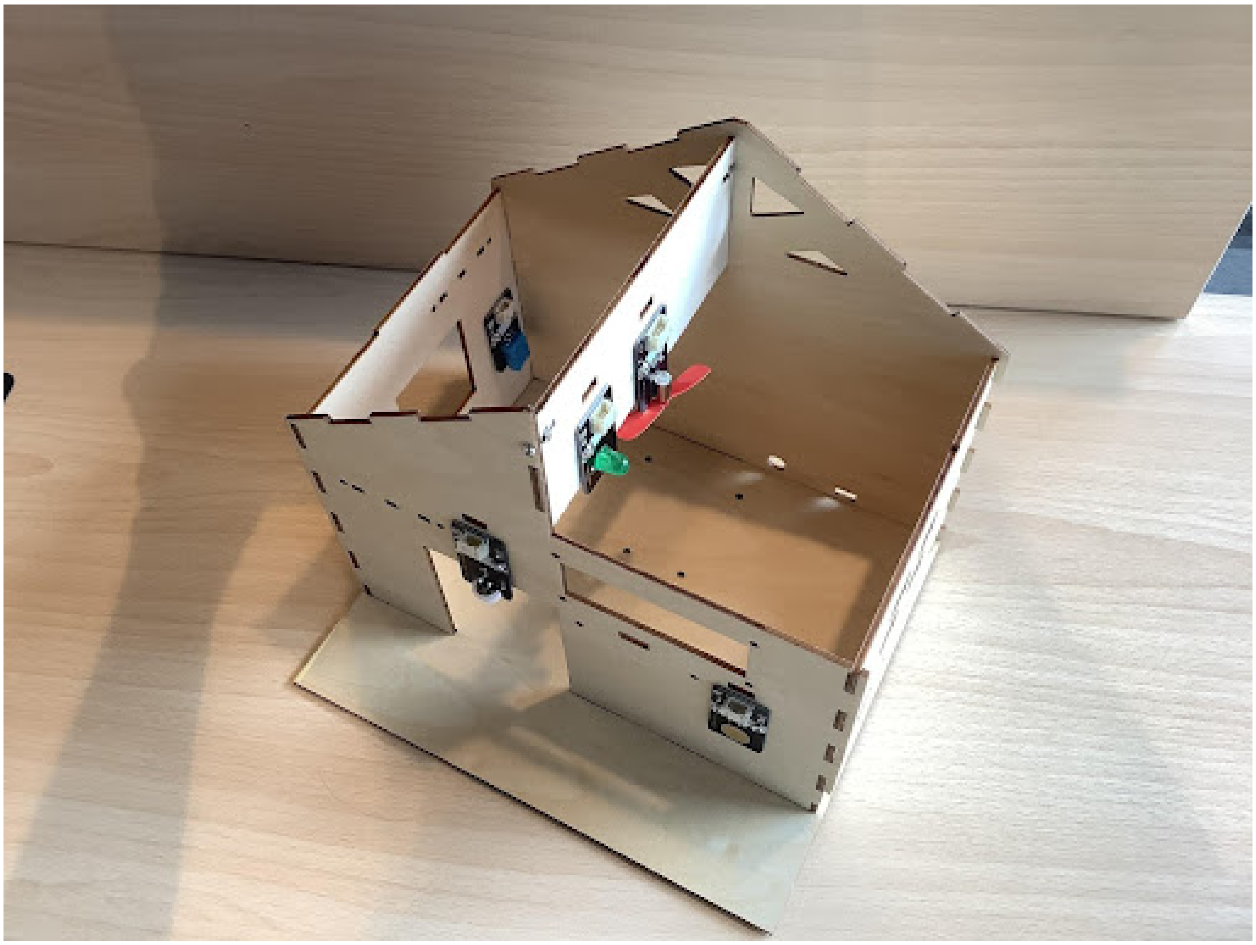
Temperature and Humidity Sensor	Arduino BLE-UNO
G	GND
V	VCC · 5V
S	A0-A5

DC Electric Motor Wind Fan Model



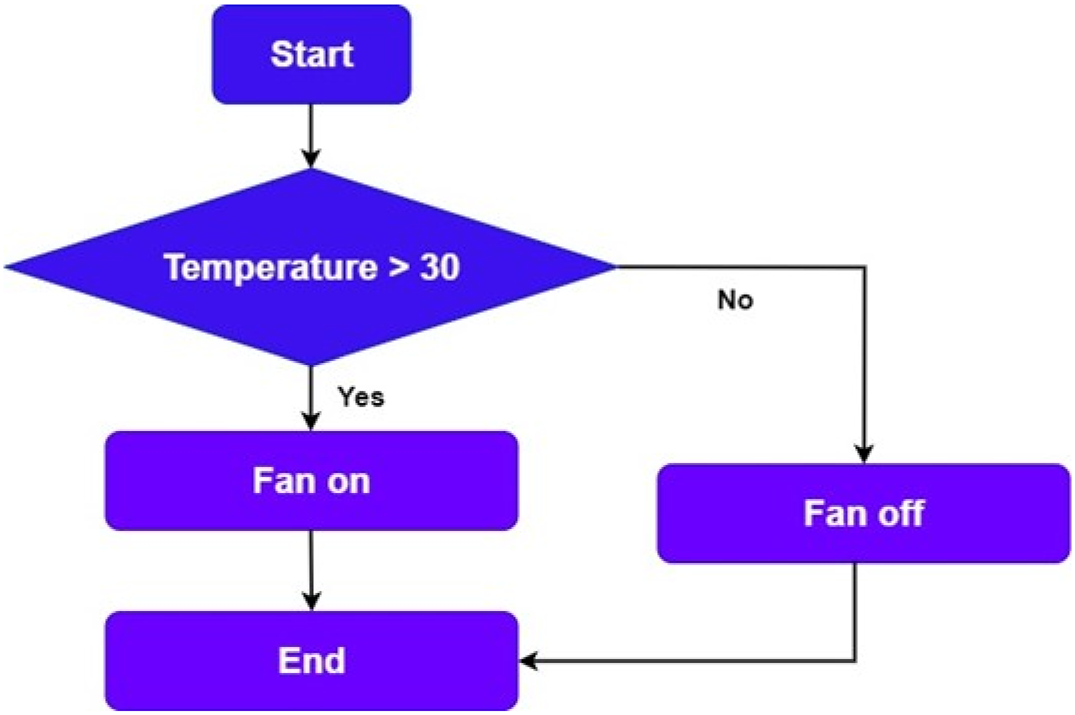
INA	INB	Power Status
0	0	Release
1	0	Turn
0	1	Reverse
1	1	Stop (Brake)

Installation of Temperature-Controlled Fan



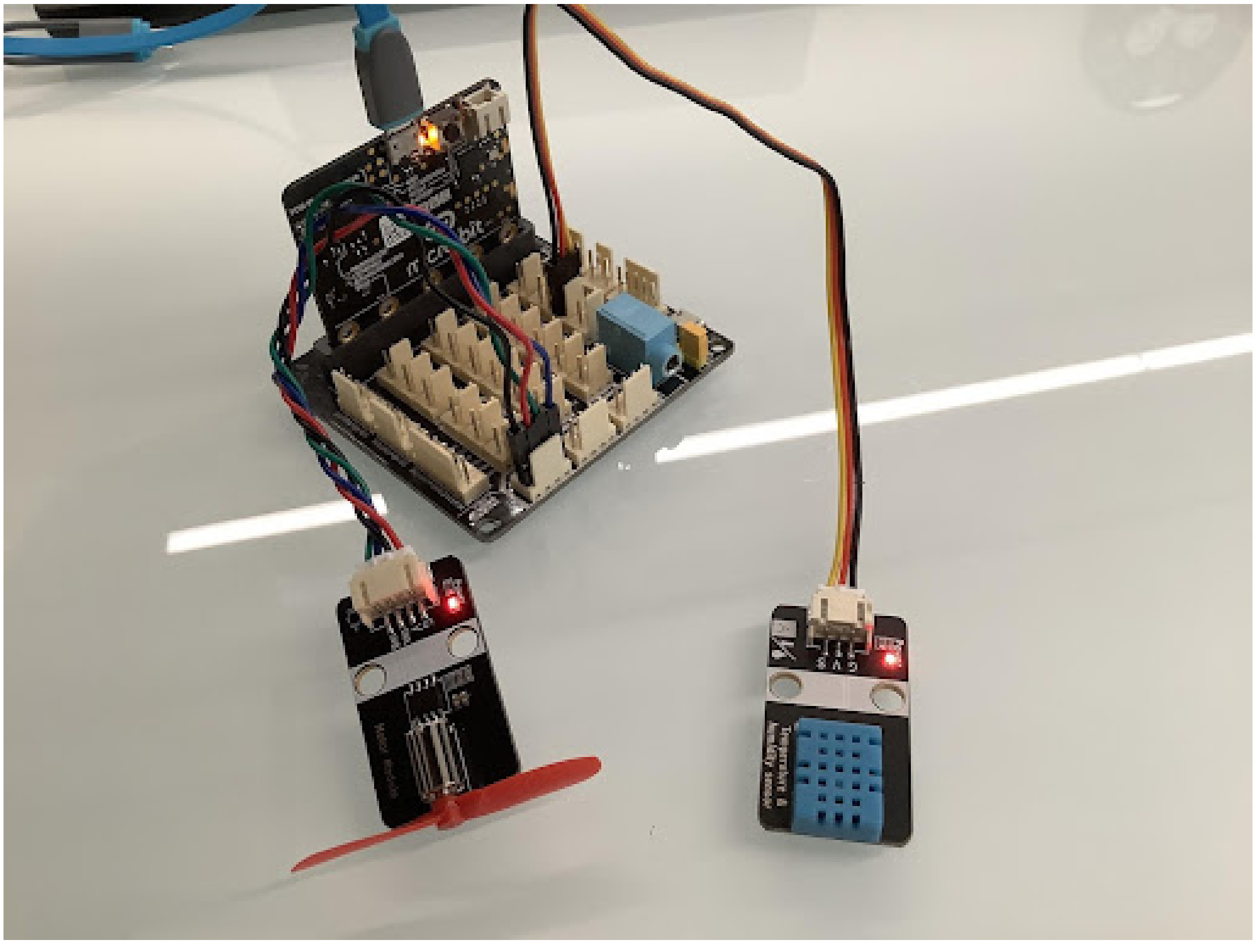
Program Design

Algorithm Design



Hardware Connections

Sensors and Actuators | Main Control Board :– | :– Temperature and Humidity Sensor | P1 DC Motor Fan Module | P15P16



Sample Program

Conclusion

1.1.5 Chapter 5 Smart Access Control

Background

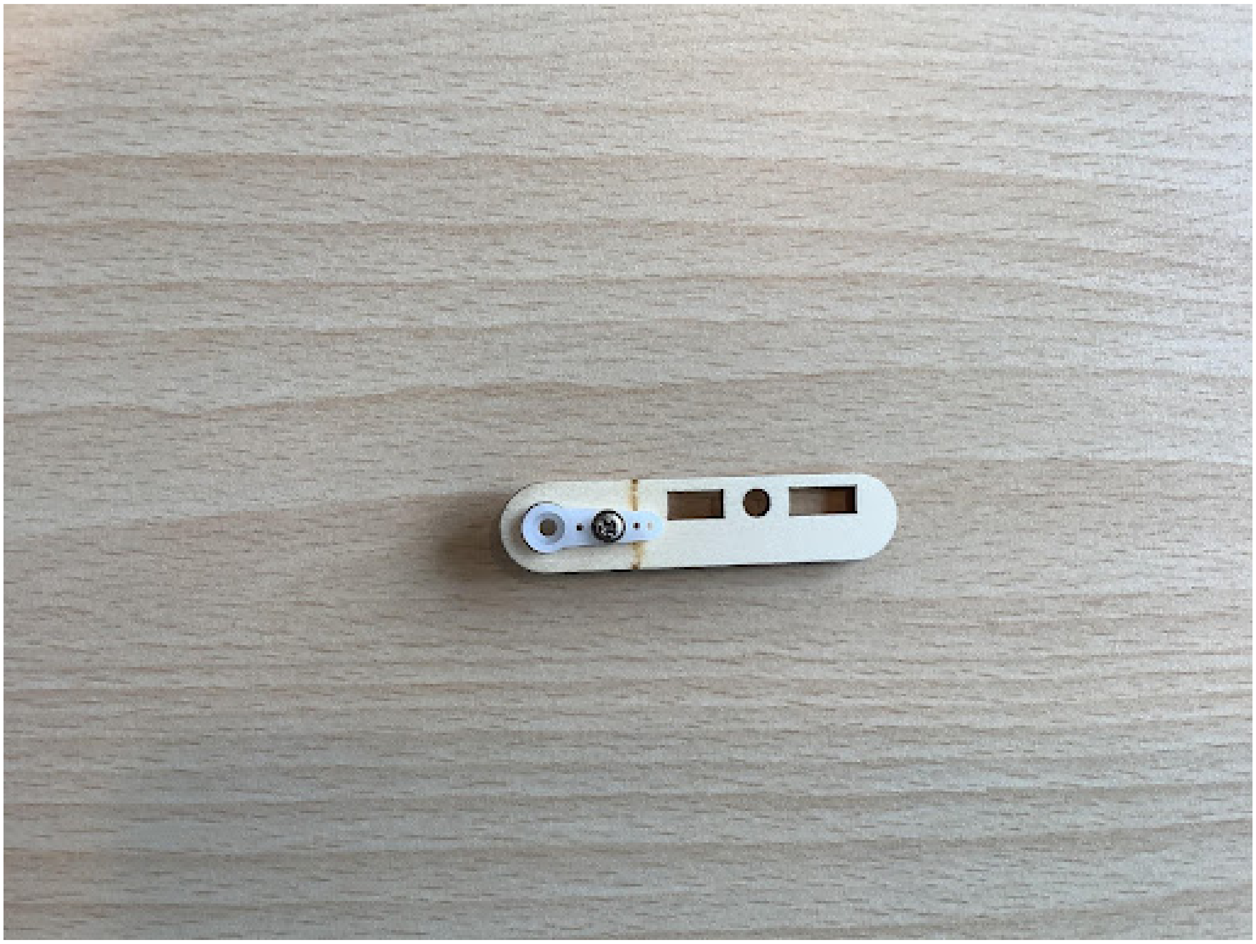
Preparation

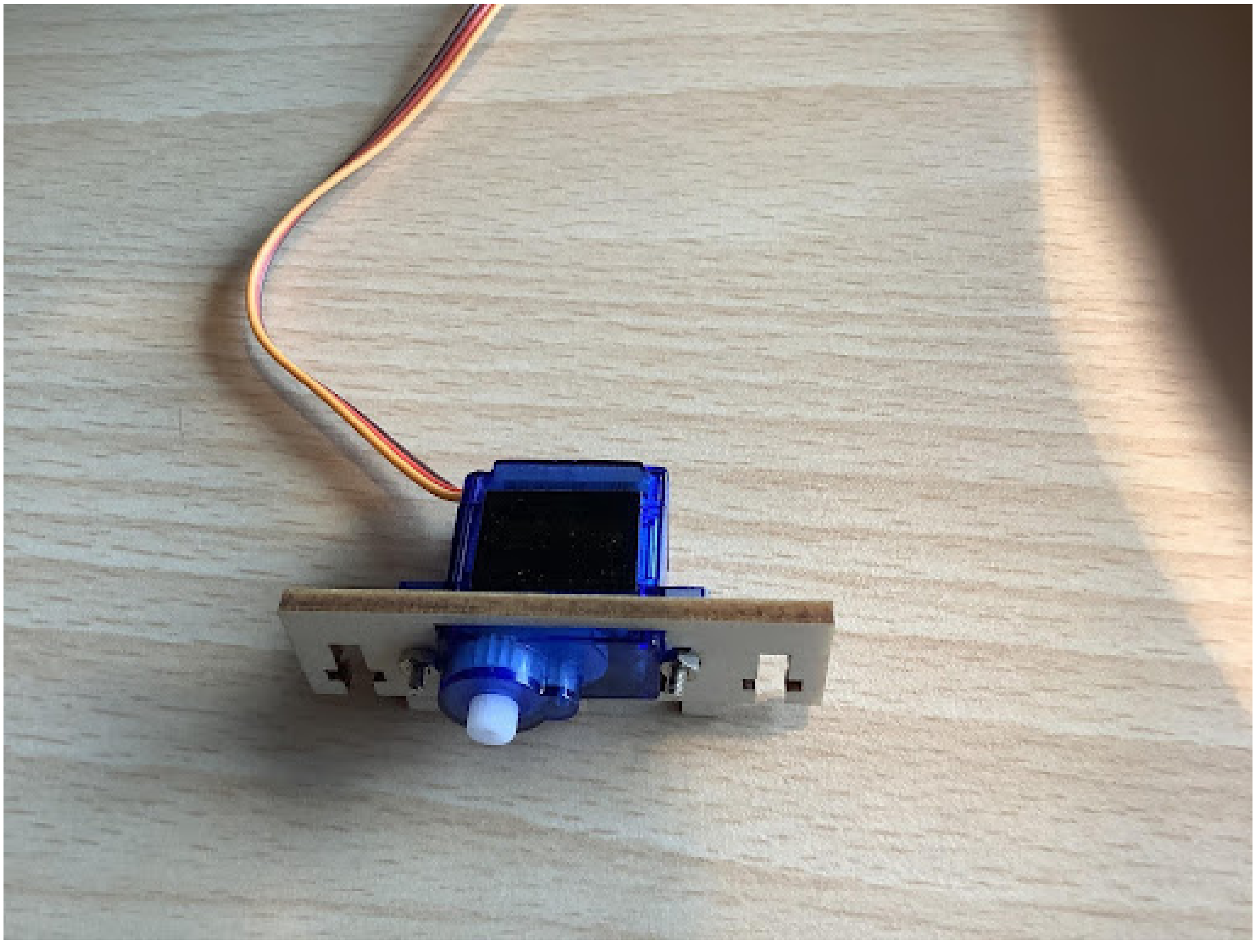
Learn About Smart Locks

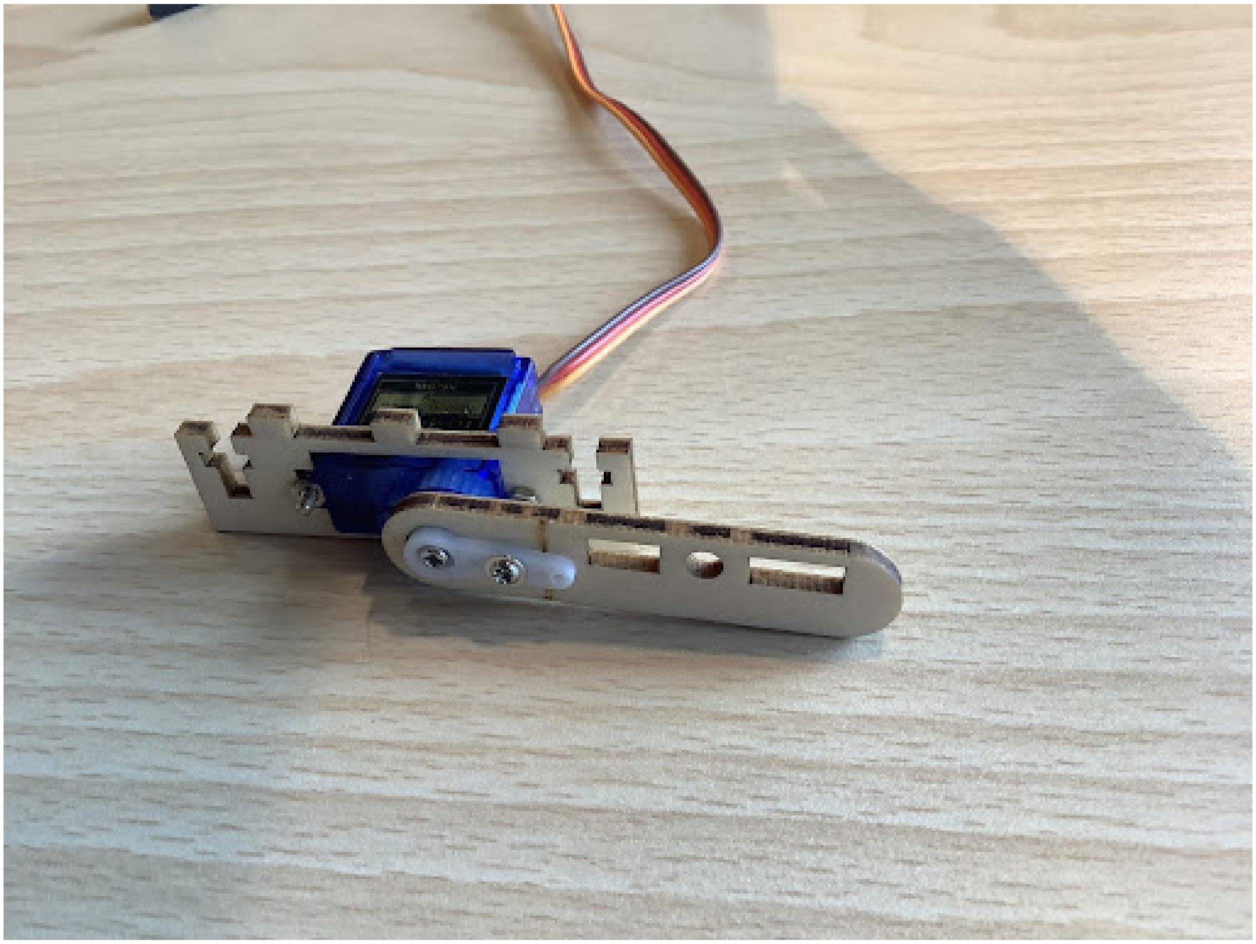
Learn About the Matrix Keyboard Sensor Module

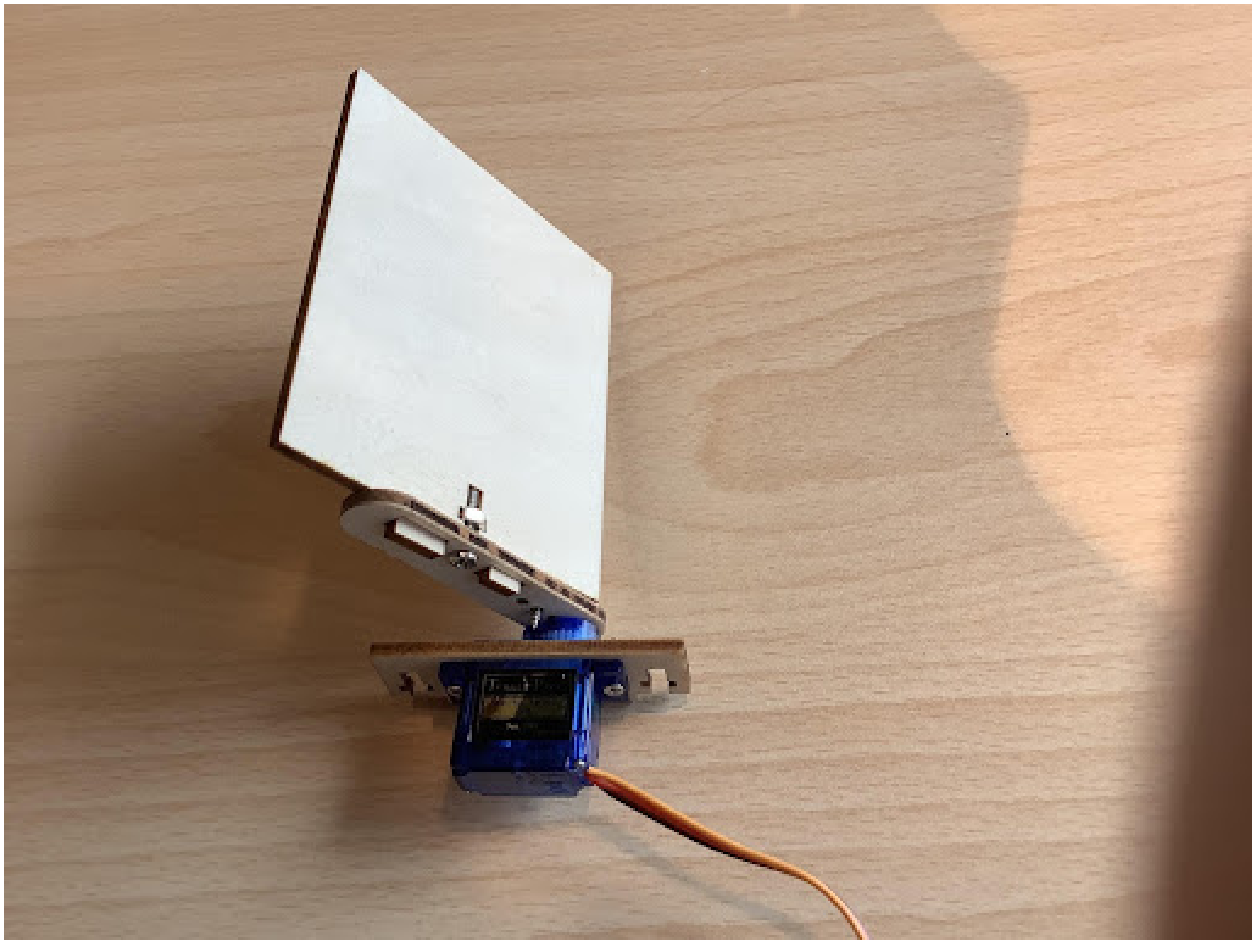
Keyboard Value | Hexadecimal value | Keyboard Value | Hexadecimal value | Keyboard Value | Hexadecimal value :– | :– | :– | :– | :– | :– 1|0xFFFE|7|0xFEFF|D|0x7FFF 2|0xFFFD|8|0xFDFF|C|0xF7FF 3|0xFFFB|9|0xFBFF|B|0xFF7F 4|0xFFEF|*|0xEFFF|A|0xFFF7 5|0xFFDF|0|0xDFFF| 6|0xFFBF|#|0xBFFF|

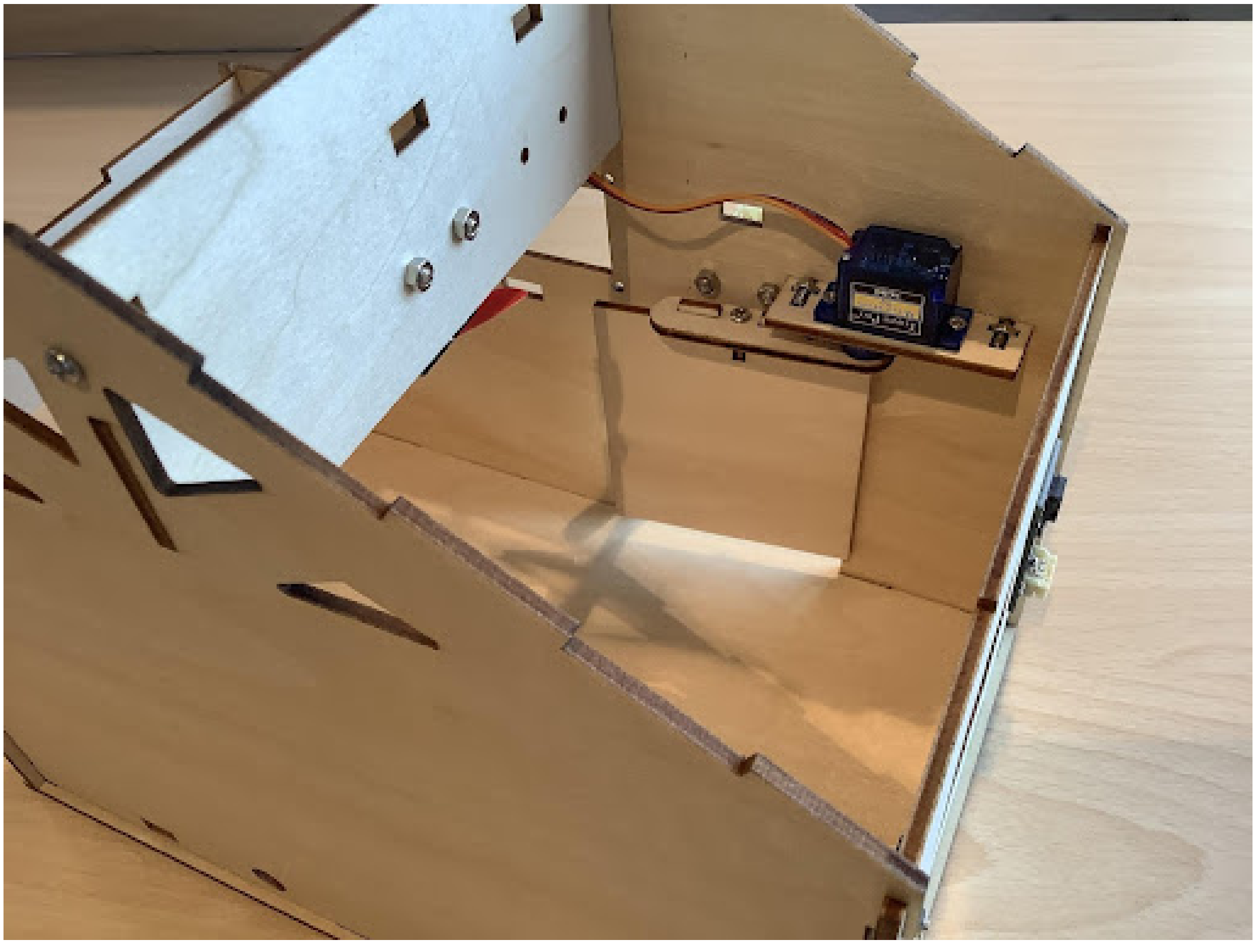
Installation of Smart Access Control

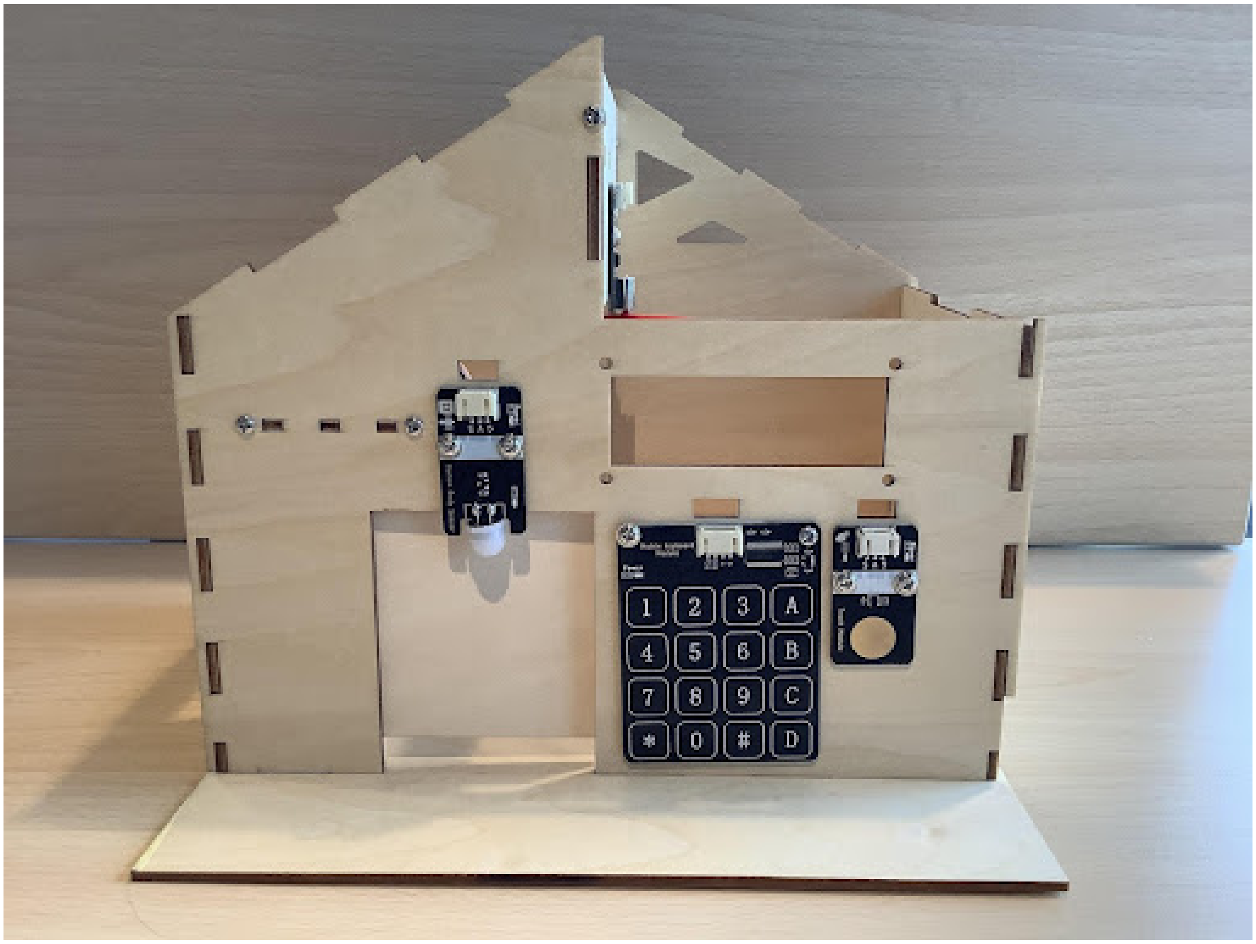










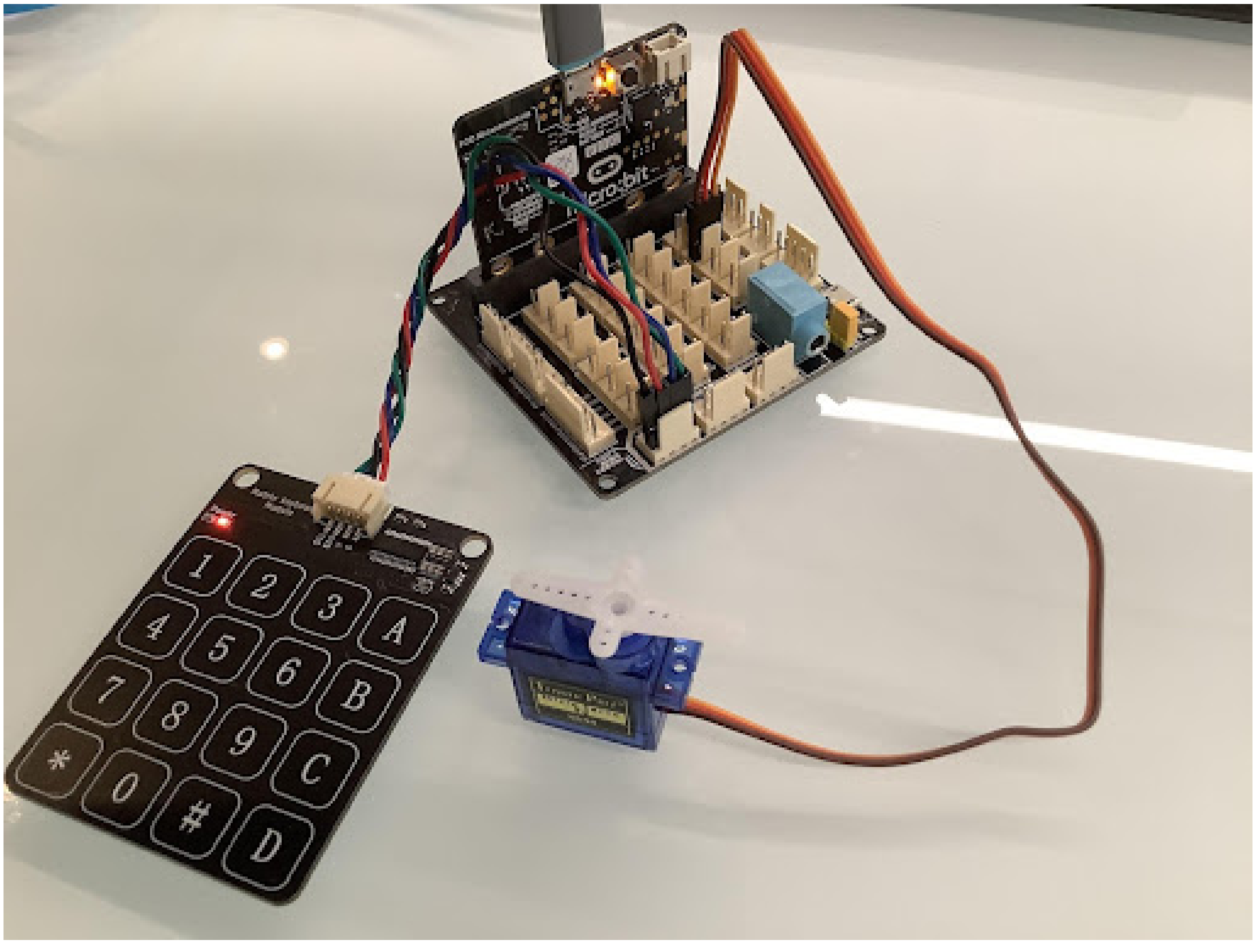


Program Design

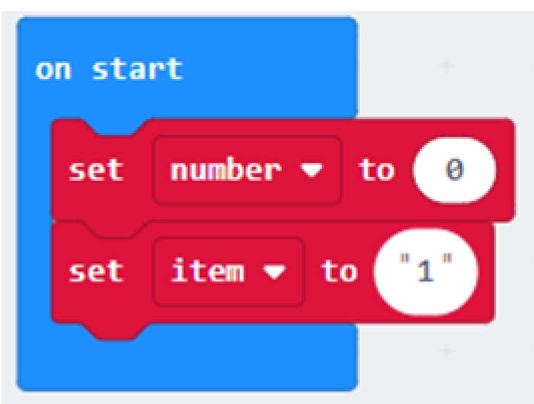
Algorithm Design

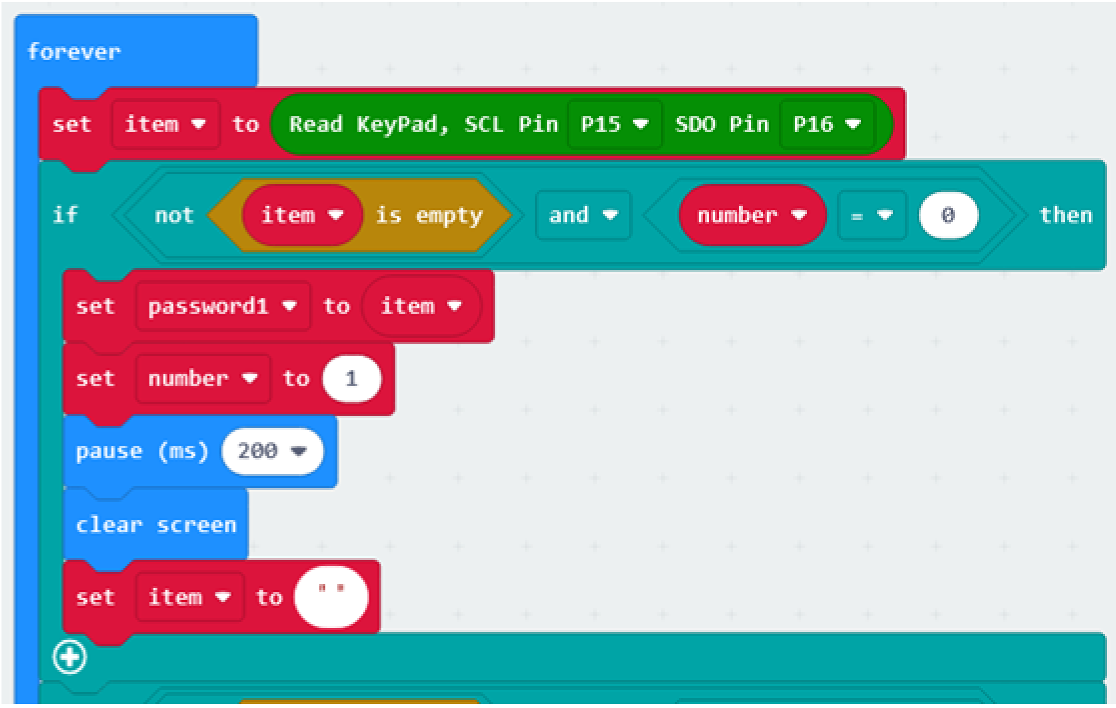
Hardware Connections

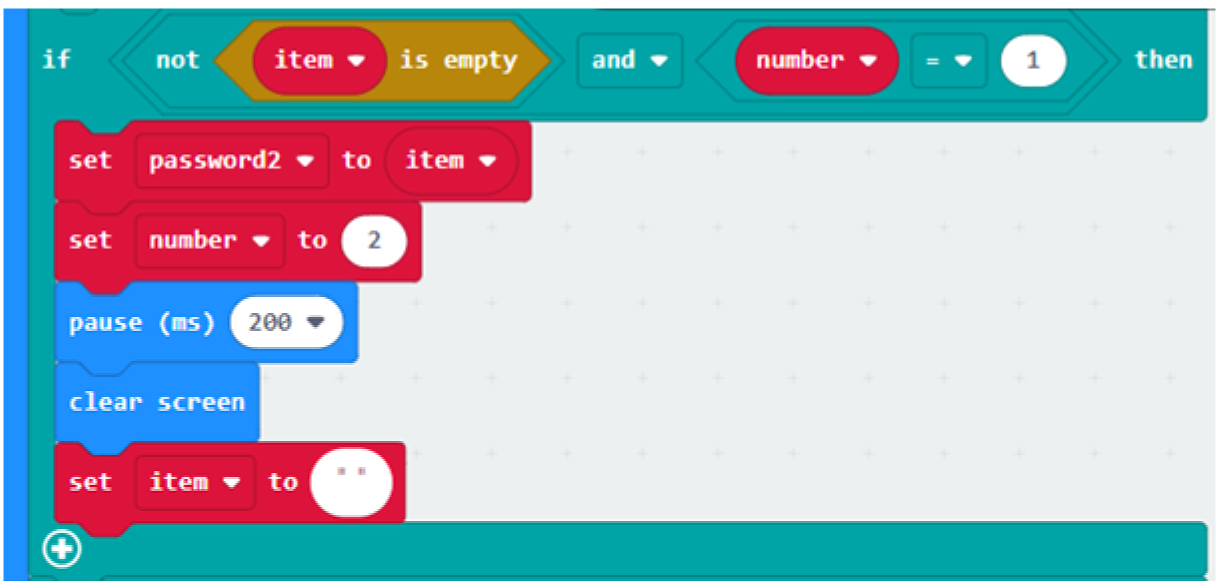
Sensors and Actuators | Main Control Board :– | :– Matrix Keyboard Sensor Module |P15(SCL)P16(SDO) Servo |P2

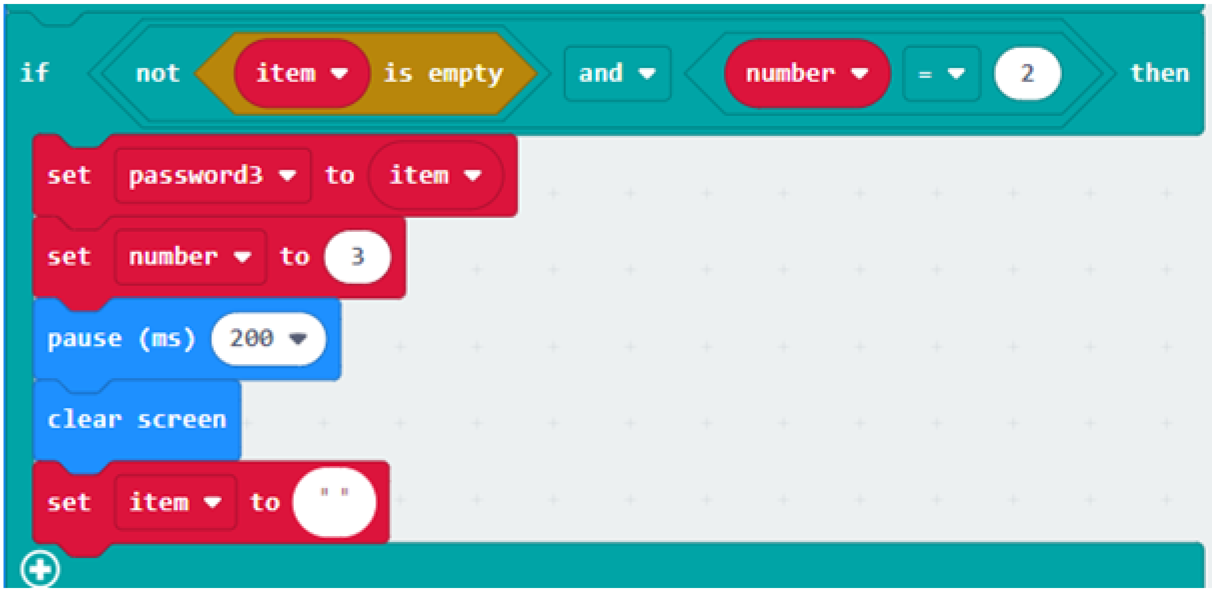


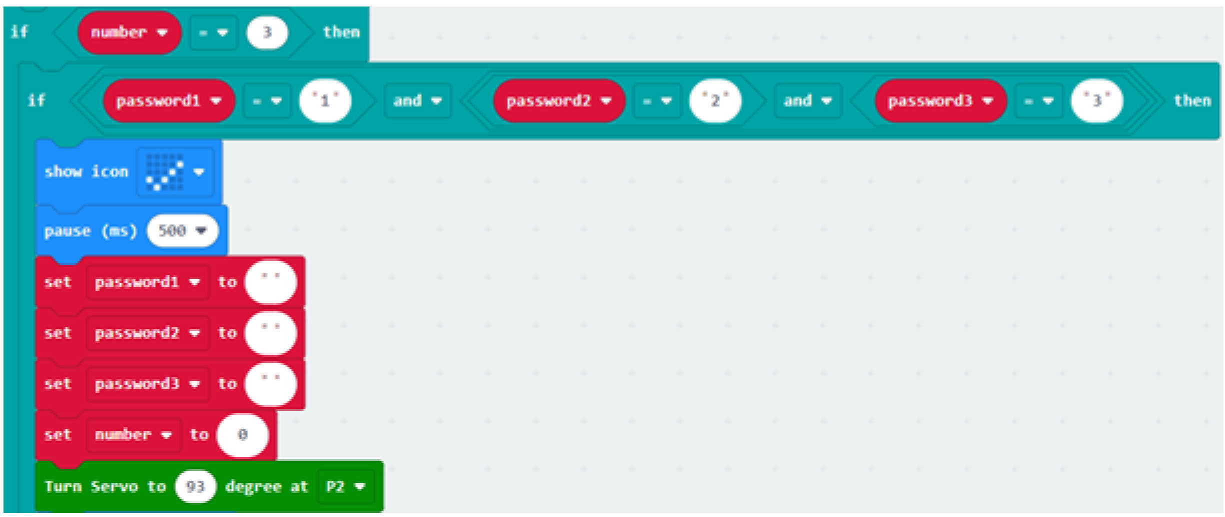
Sample Program




forever
set item to Read KeyPad, SCL Pin P15 SDO Pin P16
if not item is empty and number = 0 then
set password1 to item
set number to 1
pause (ms) 200
clear screen
set item to " "


if not item is empty and number = 1 then
set password2 to item
set number to 2
pause (ms) 200
clear screen
set item to " "


if not item is empty and number = 2 then
set password3 to item
set number to 3
pause (ms) 200
clear screen
set item to " "


if number = 3 then
if password1 = "1" and password2 = "2" and password3 = "3" then
show icon
pause (ms) 500
set password1 to " "
set password2 to " "
set password3 to " "
set number to 0
Turn Servo to 93 degree at P2


pause (ms) 5000
Turn Servo to 0 degree at P2
clear screen
else
show icon
pause (ms) 1000
set password1 to " "
set password2 to " "
set password3 to " "
set number to 0

Conclusion

1.1.6 Chapter 6 Light-controlled Automatic Windows

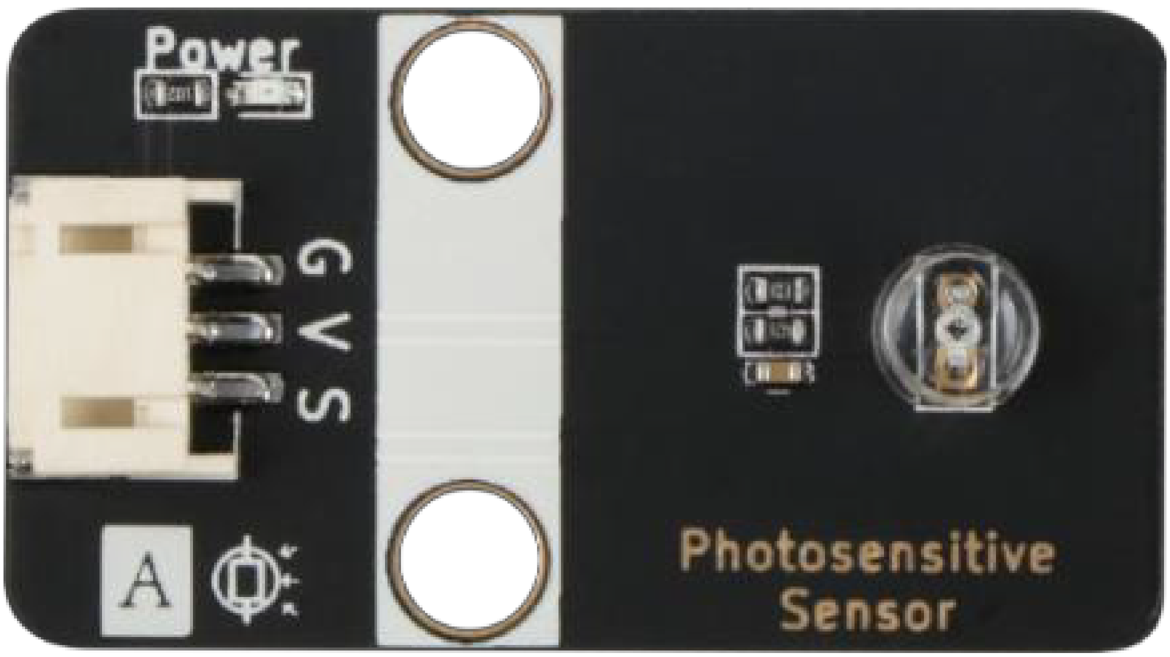
Background

Preparation

Learn About the Principle of Light-controlled Automatic Windows

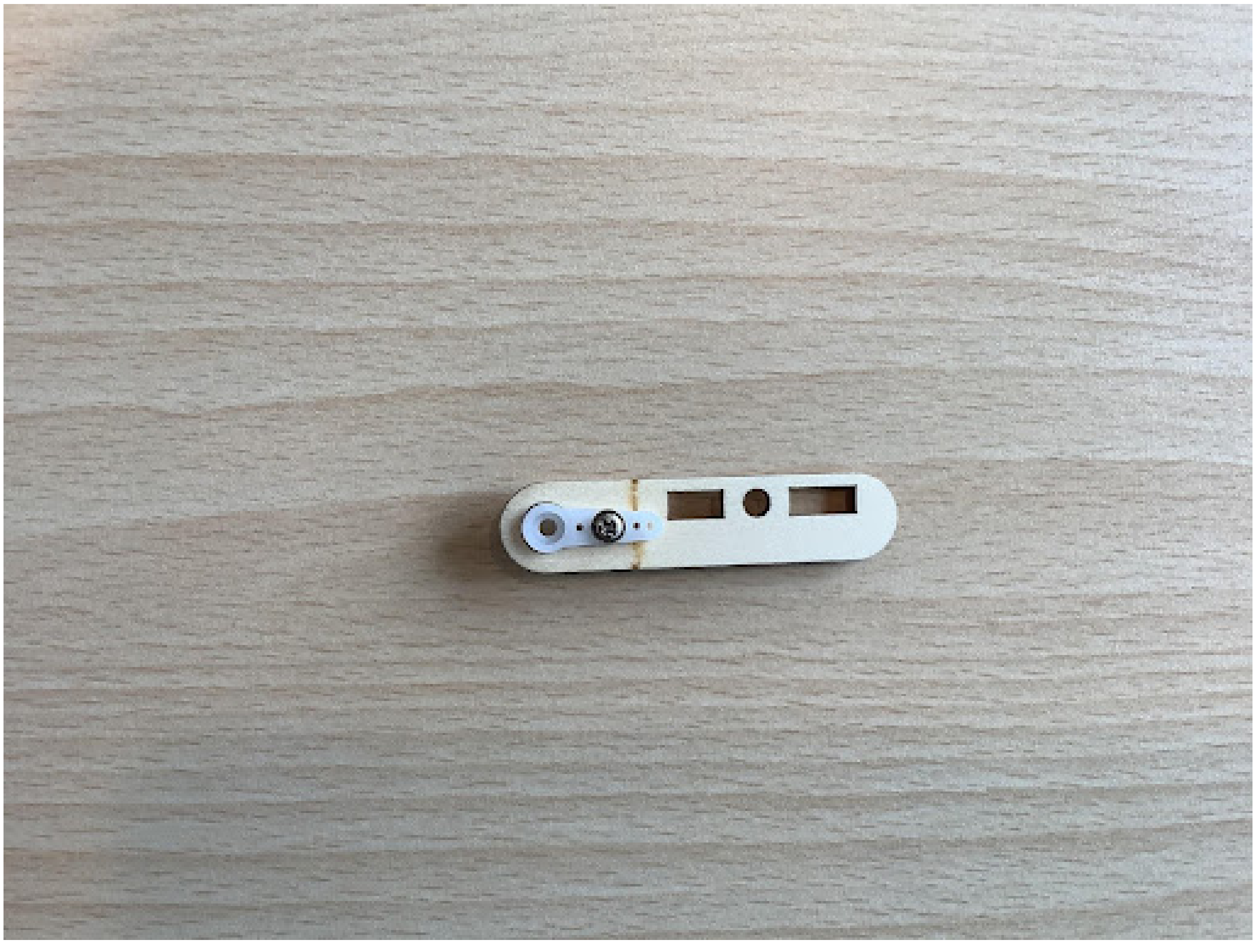
Learn About Photoreceptors and Digital Tube Displays

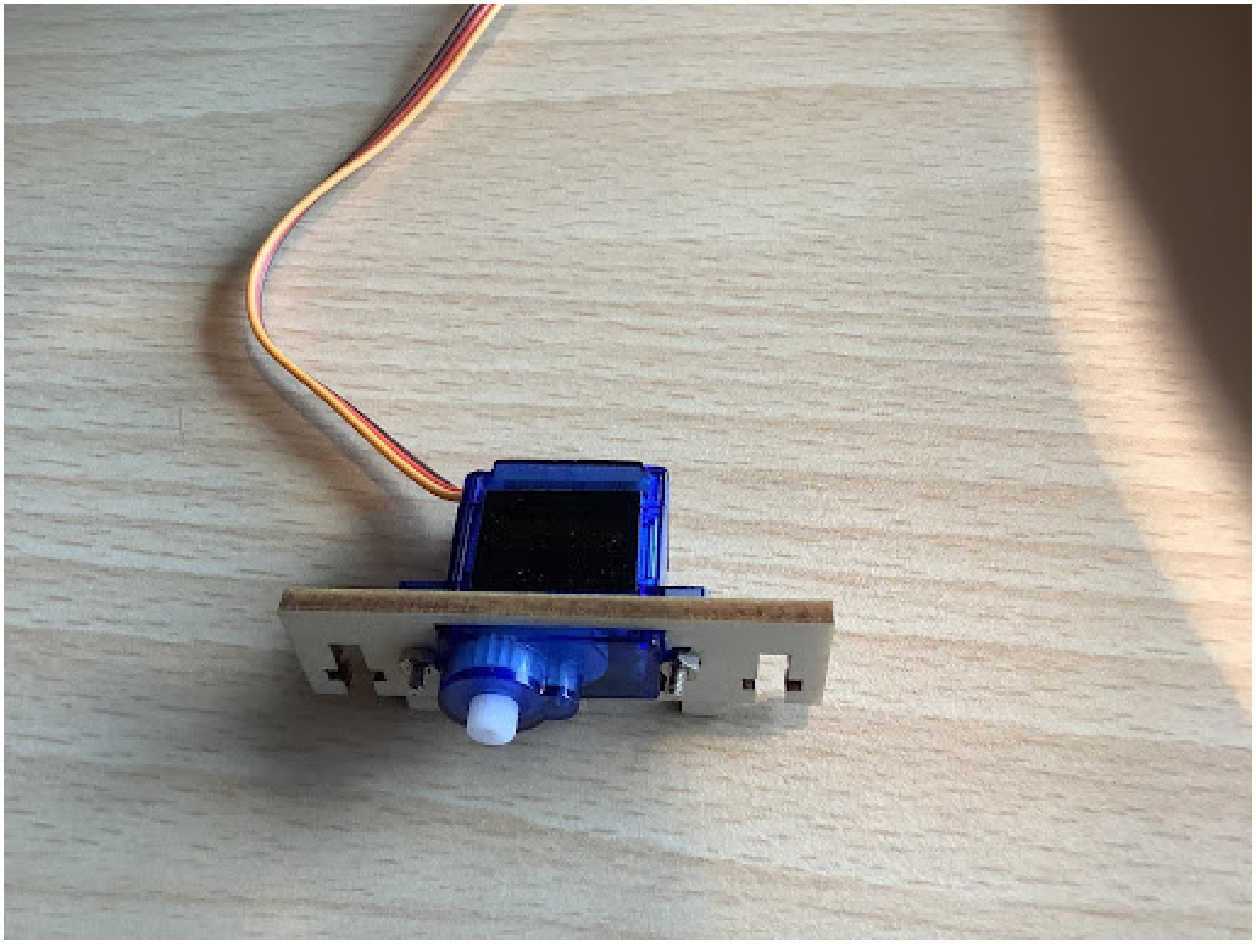
Photosensitive sensor

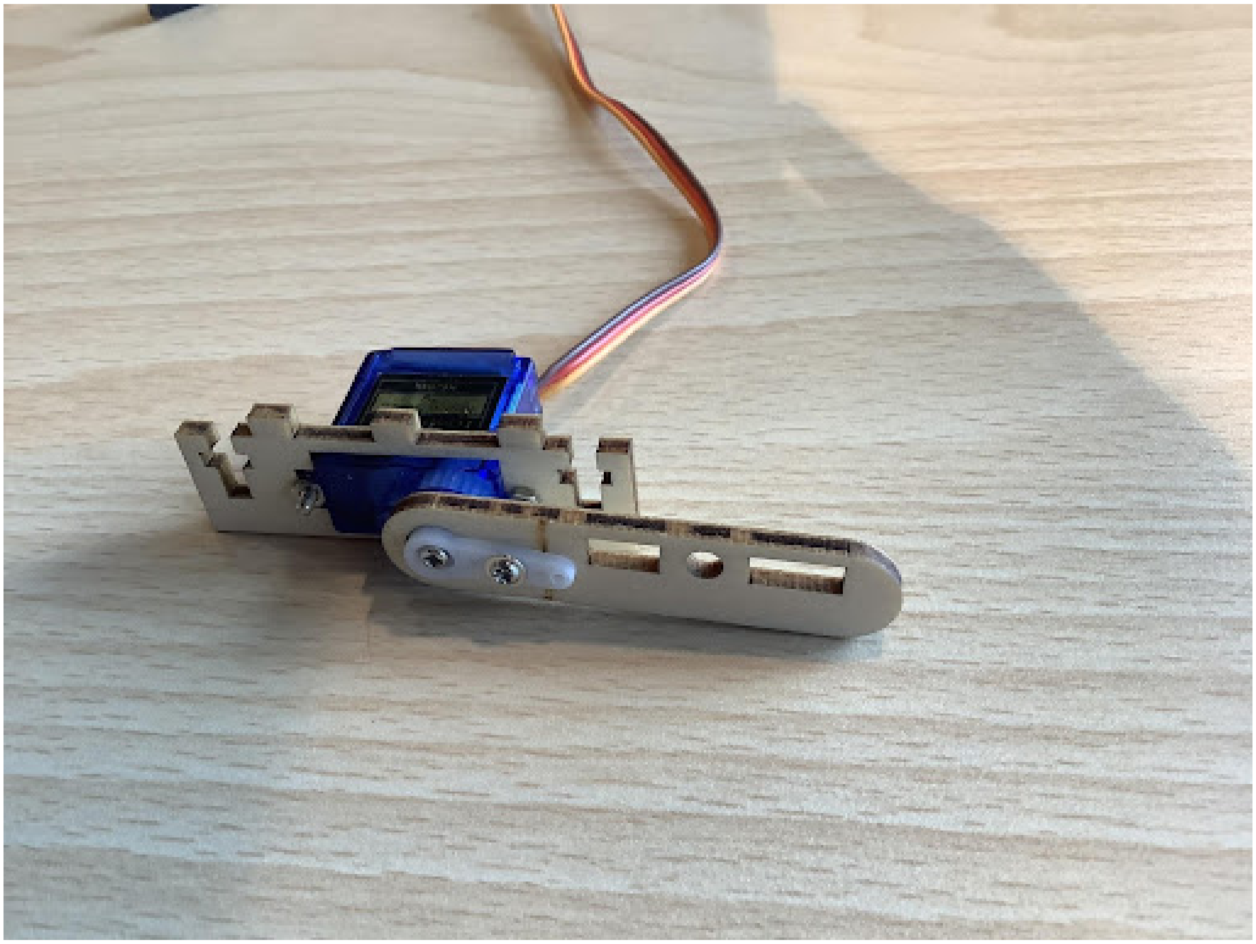


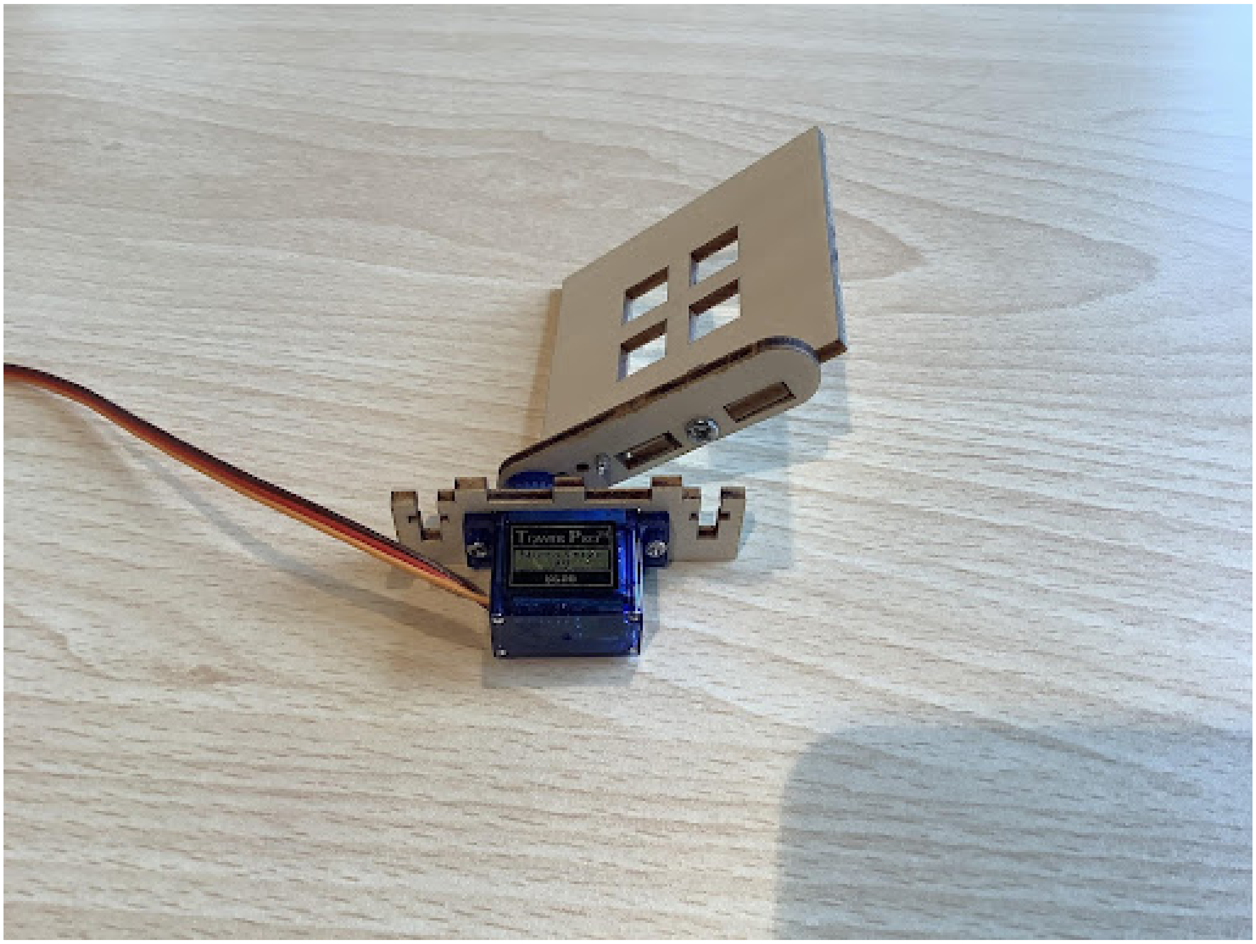
Photosensitive sensor	Arduino BLE-UNO
G	GND
V	VCC · 5V
S	A0-A5

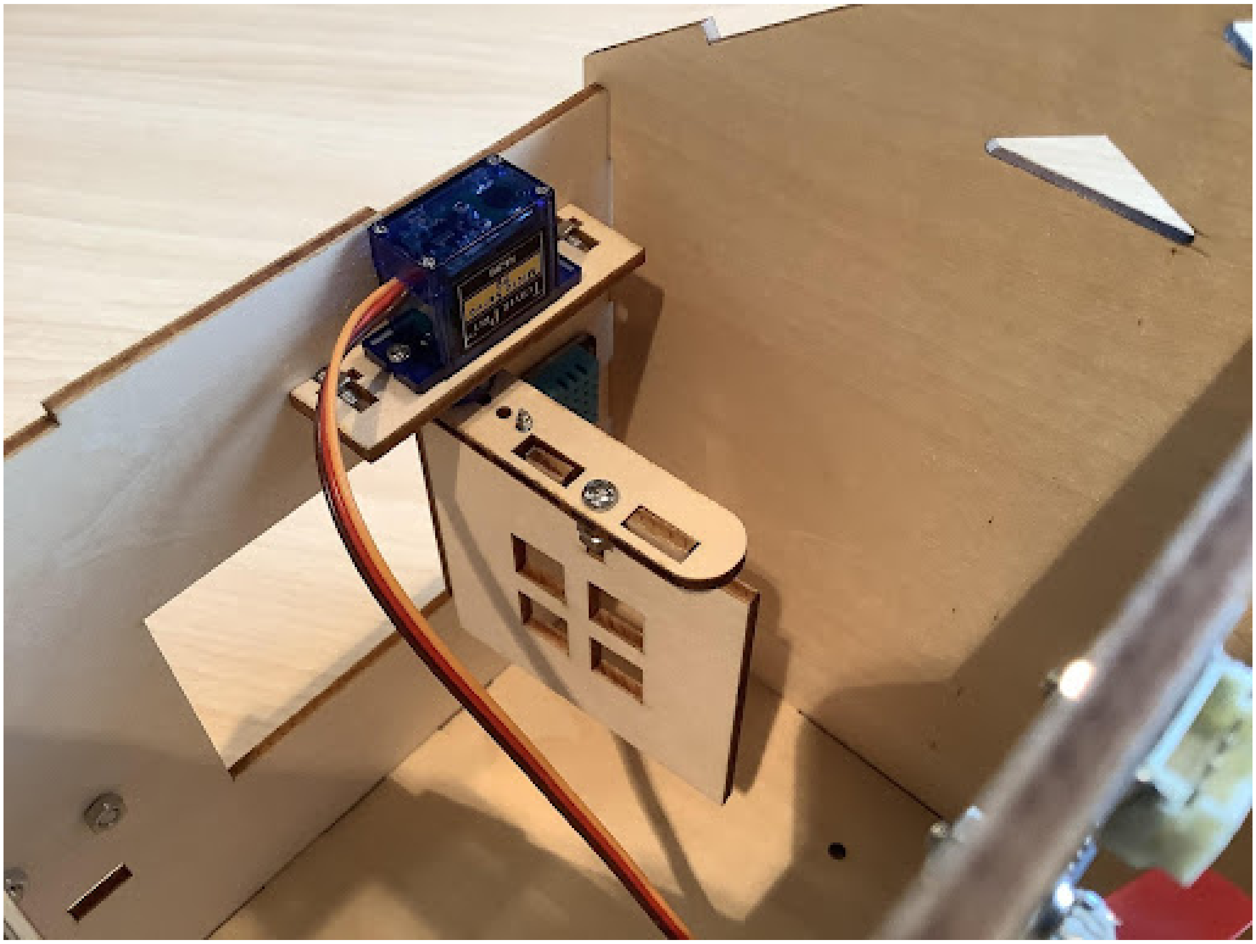
Installation Of Light-controlled Automatic Window

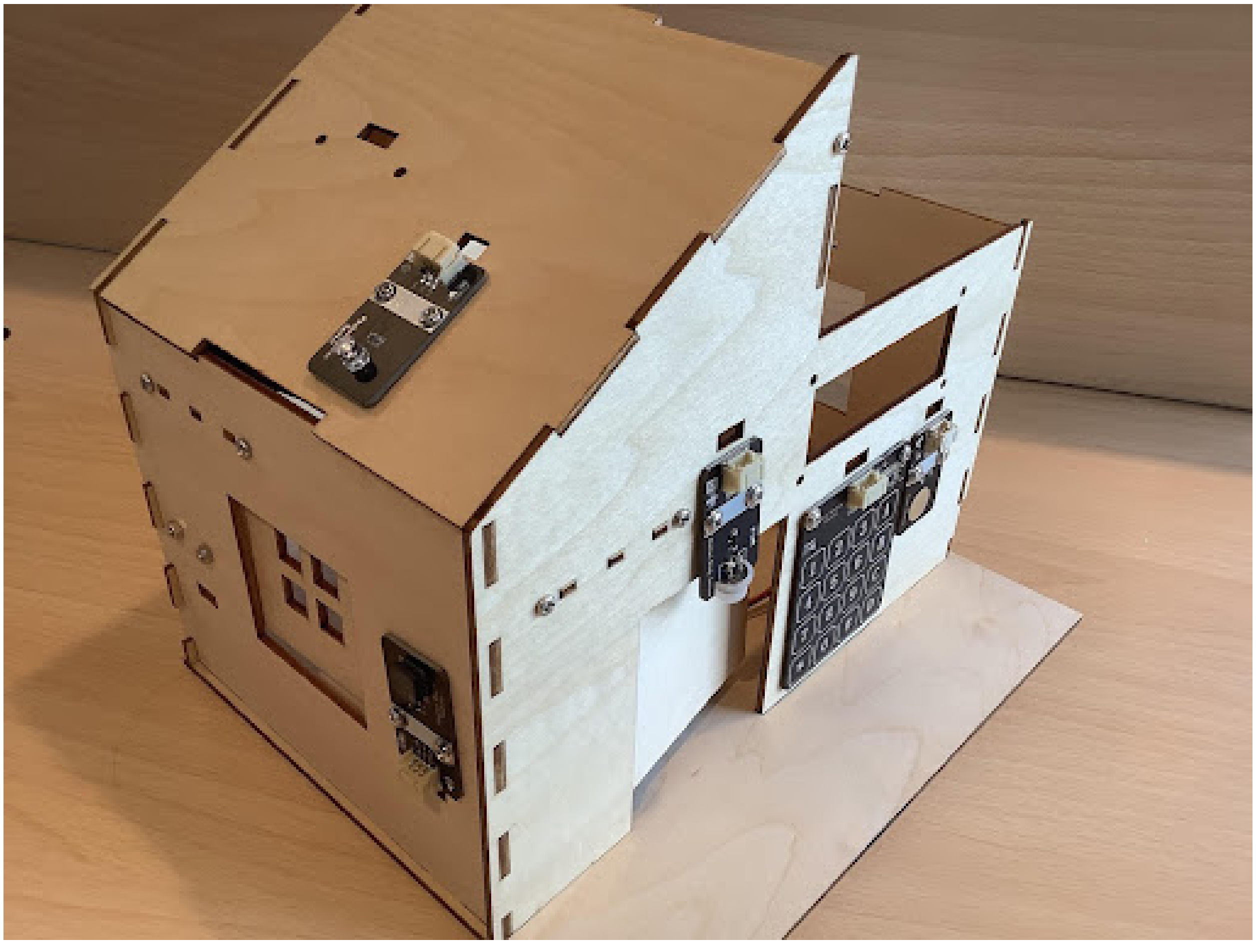






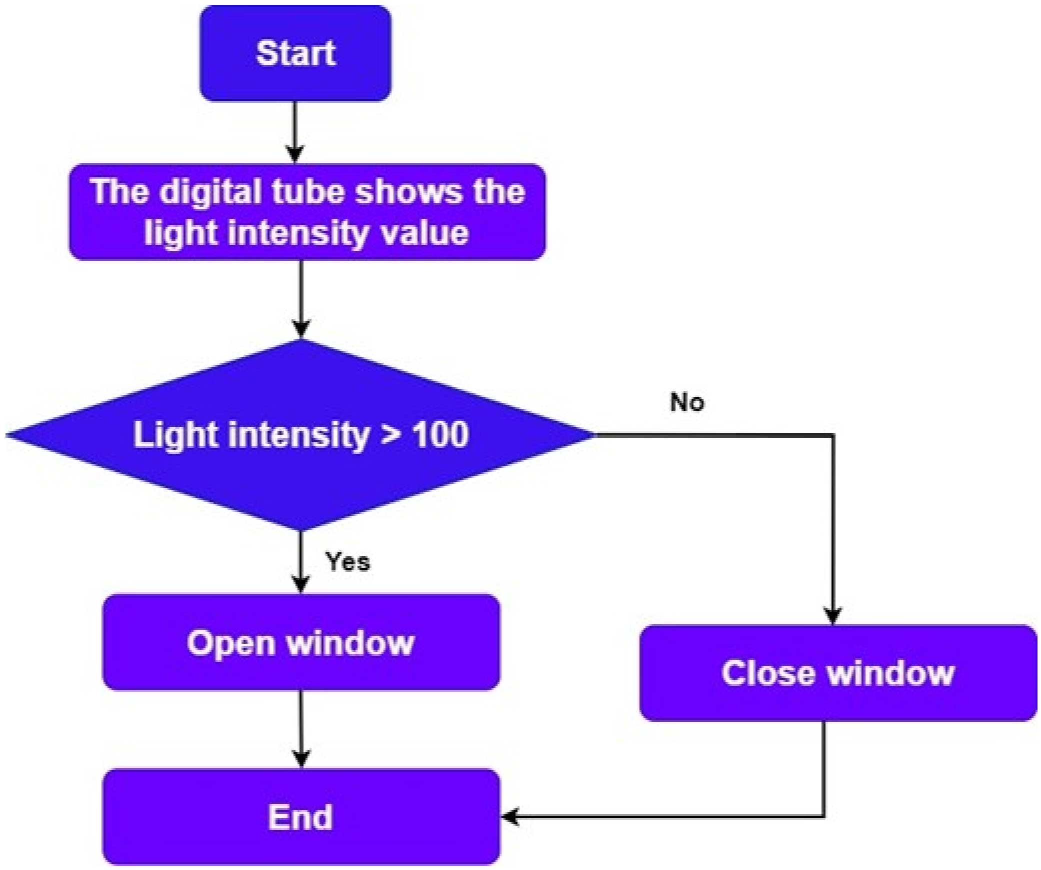






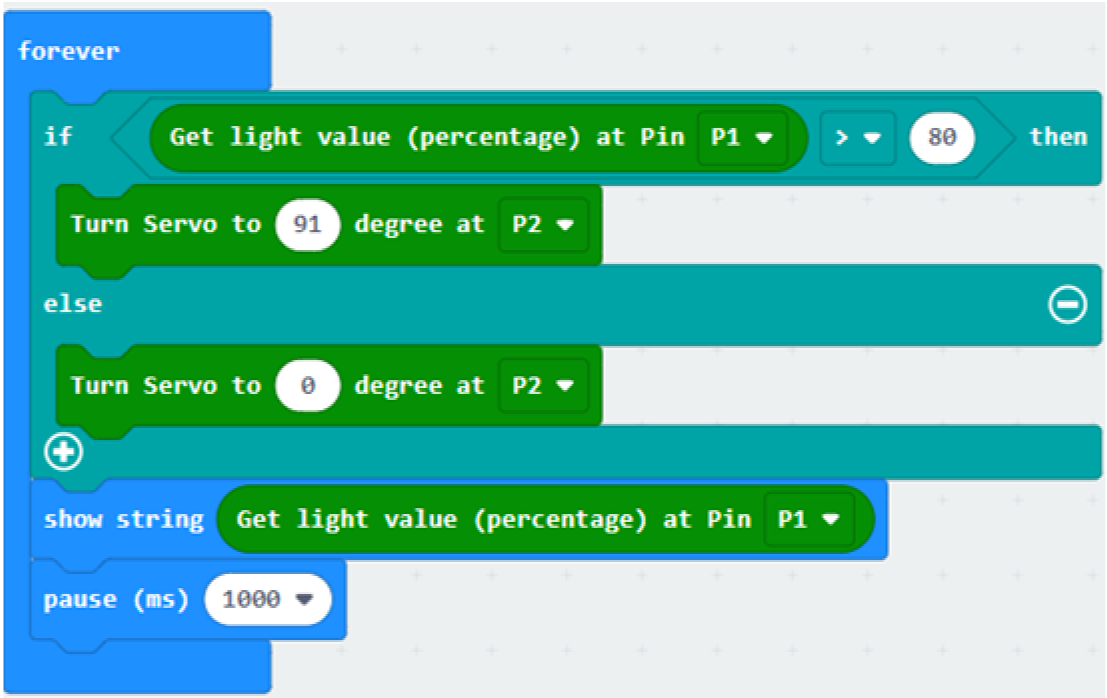
Program Design

Algorithm Design



Hardware Connections

Sample Program



Conclusion

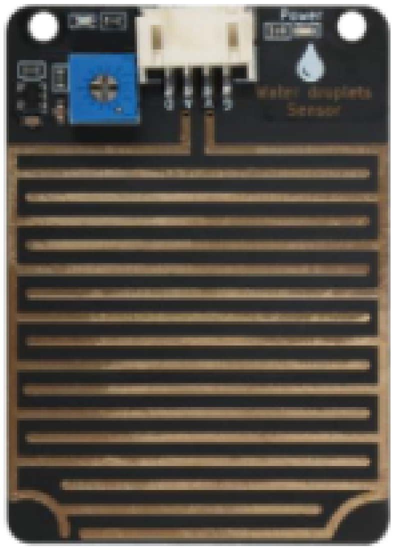
1.1.7 Chapter 7 Smart Rain Control Windows

Background

Preparation

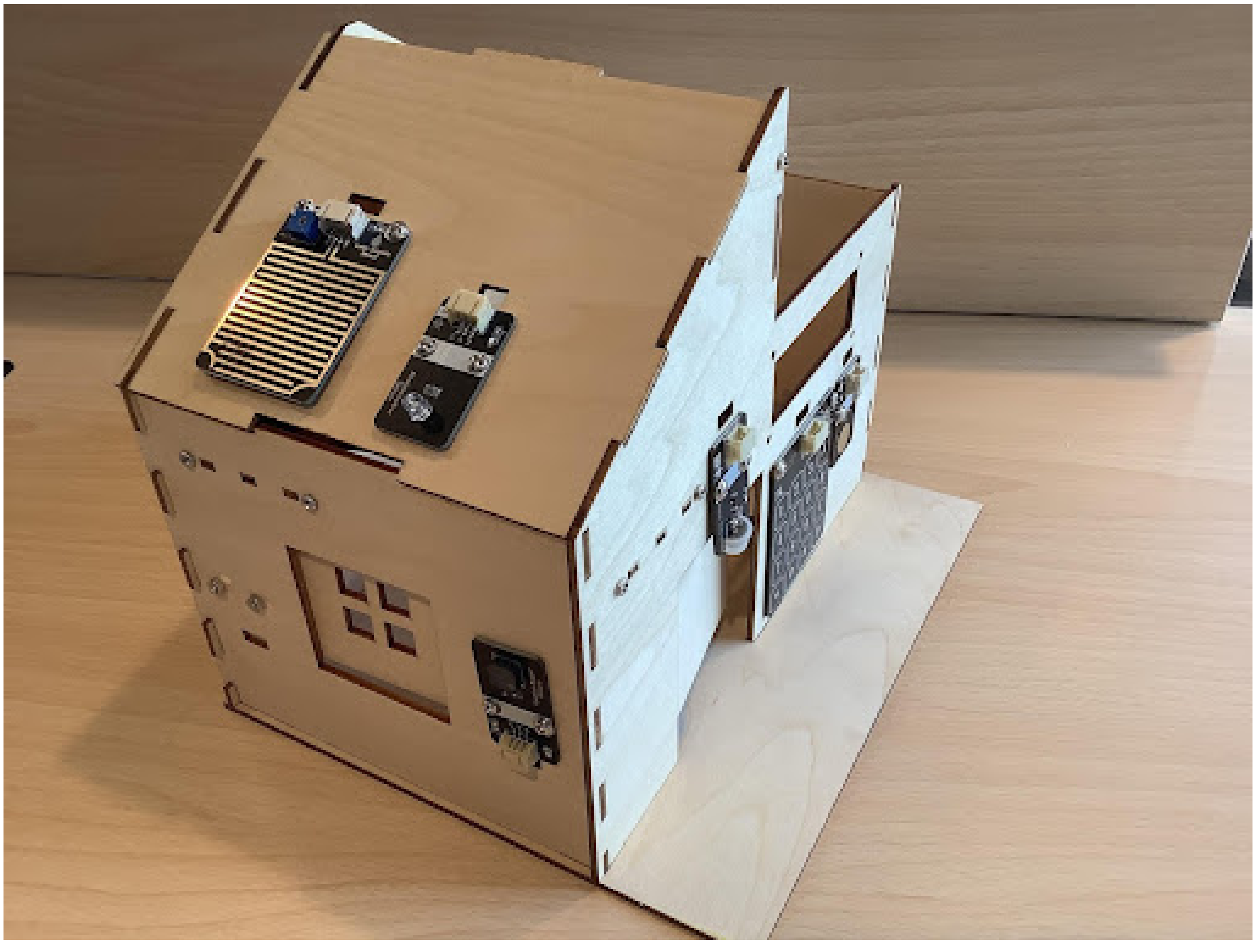
Principle of Smart Rain Control Window

Learn About Raindrop Sensor



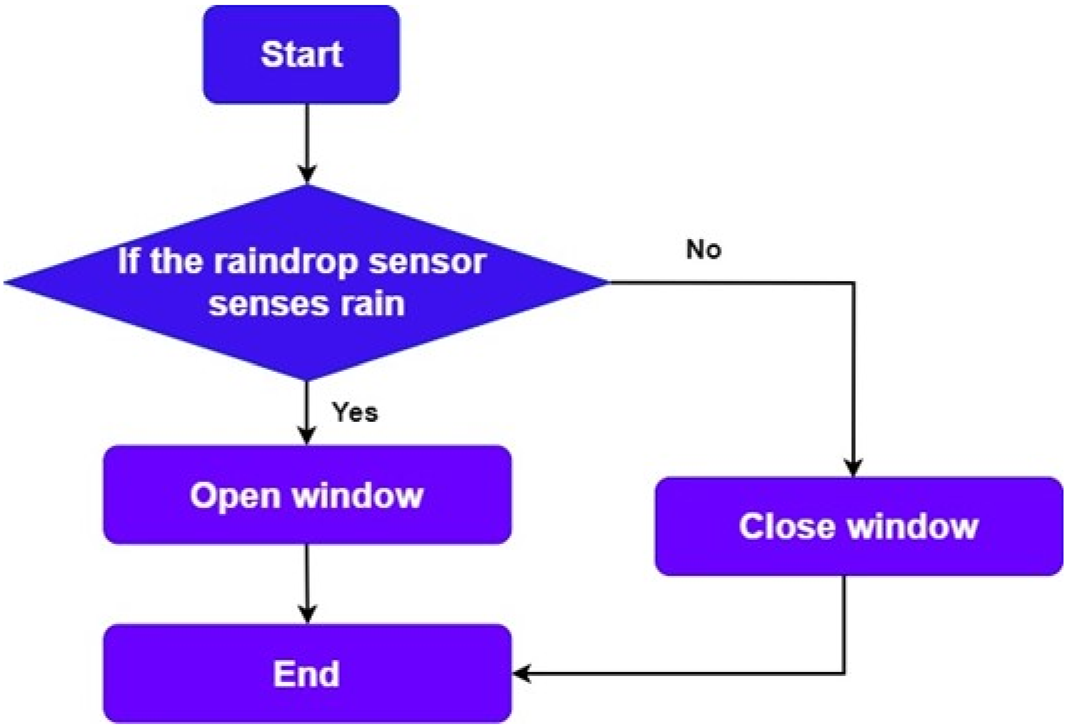
Raindrop sensor	Arduino BLE-UNO
G	GND
V	VCC、5V
A	A0-A5
D	D0-D13

Construction of Automatic Smart Window



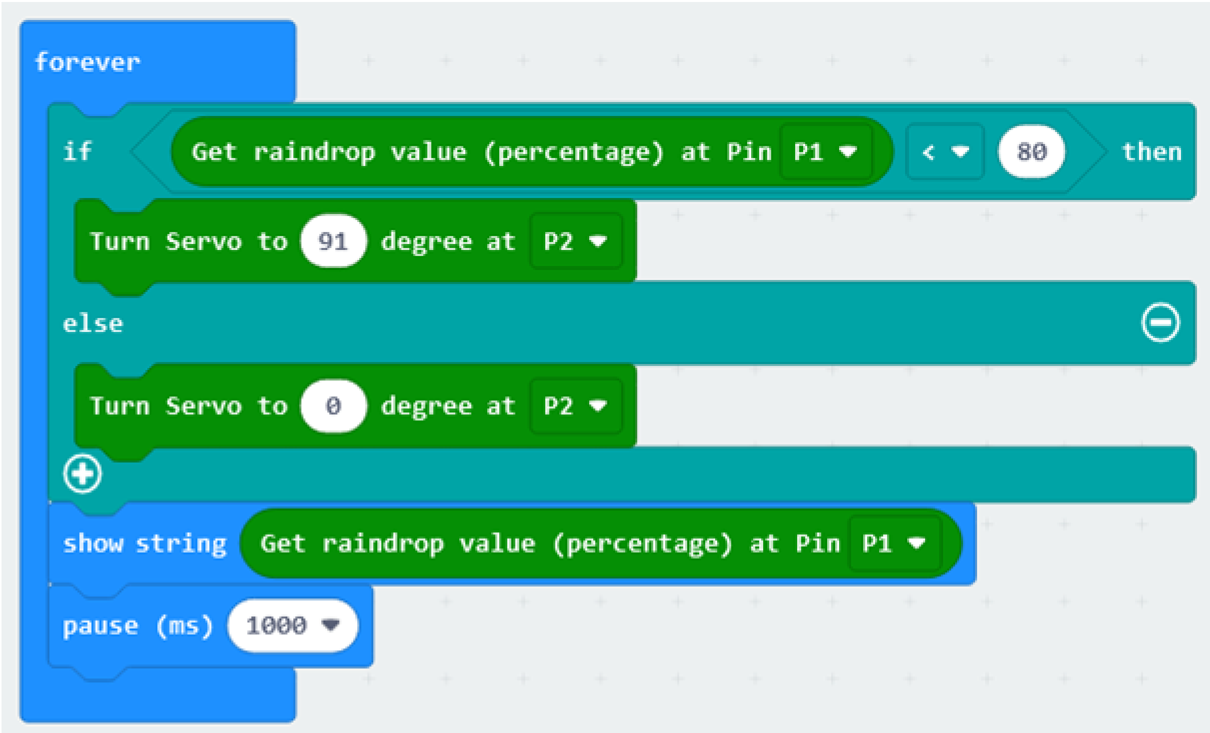
Program Design

Algorithm Design



Hardware Connections

Sample Program



Conclusion

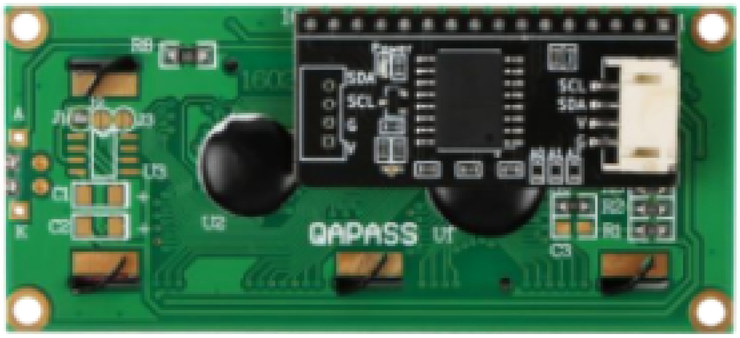
1.1.8 Chapter 8 Environmental Monitoring System

Background

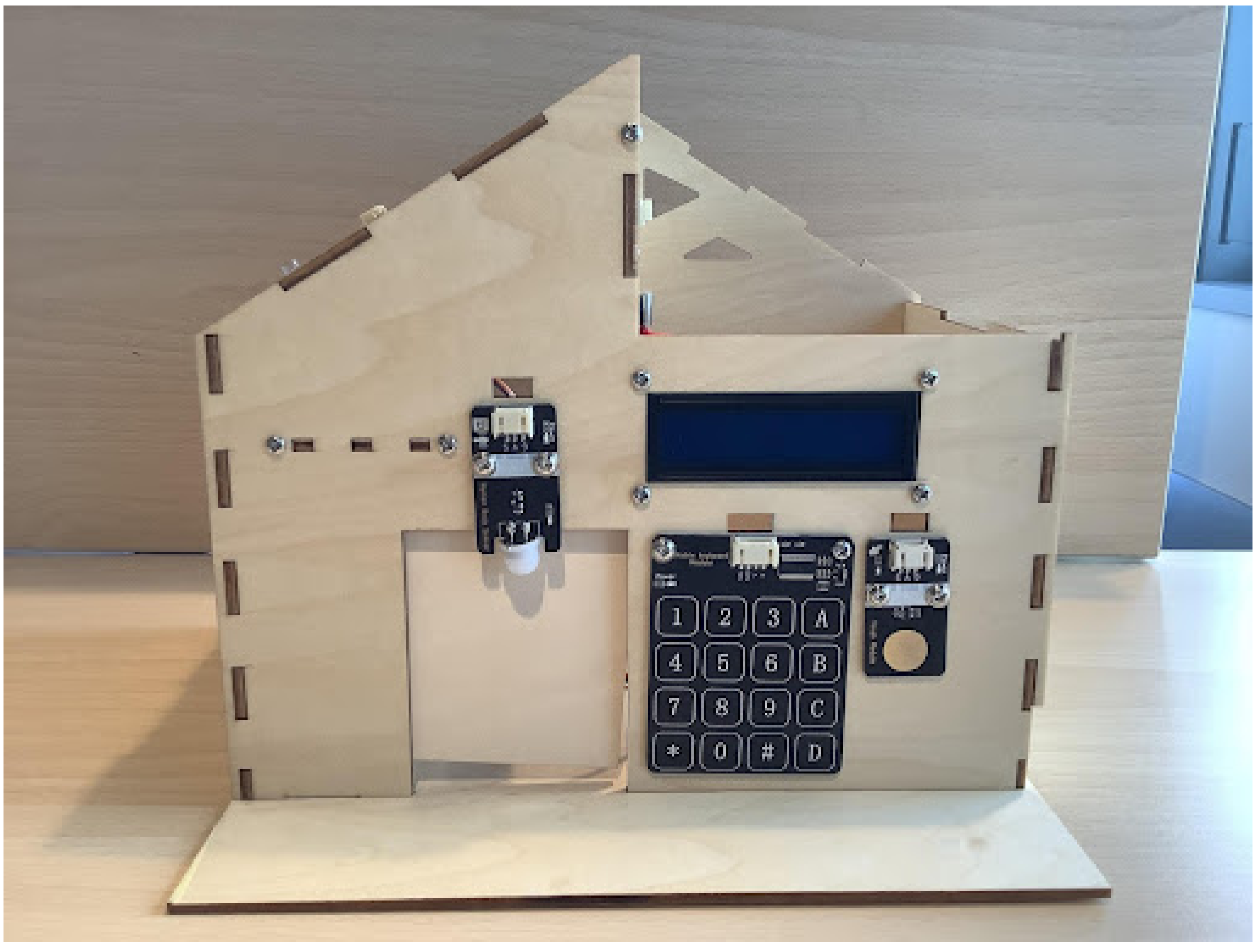
Preparation

Learn About Environmental Monitoring Systems

Learn About LCD Displays

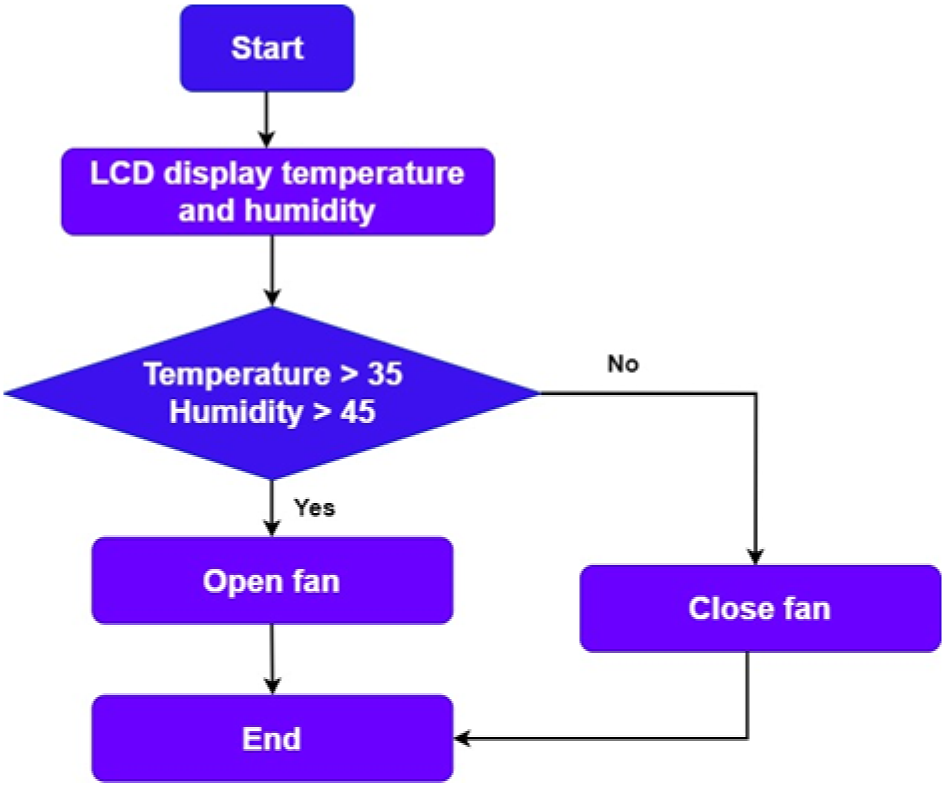


Installation of Environmental Monitoring System



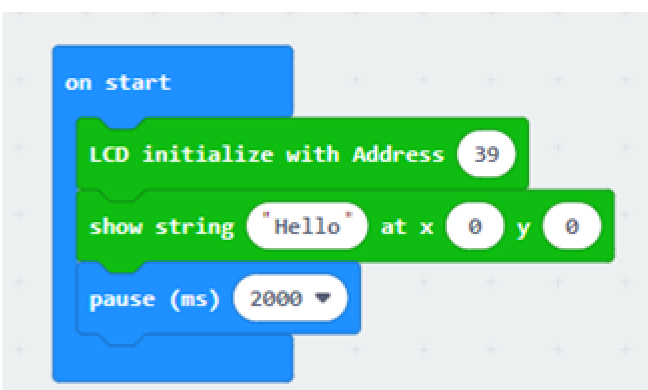
Program Design

Algorithm Design



Hardware Connections

Sample Program



Conclusion

1.2 Smart City IoT Starter Kit

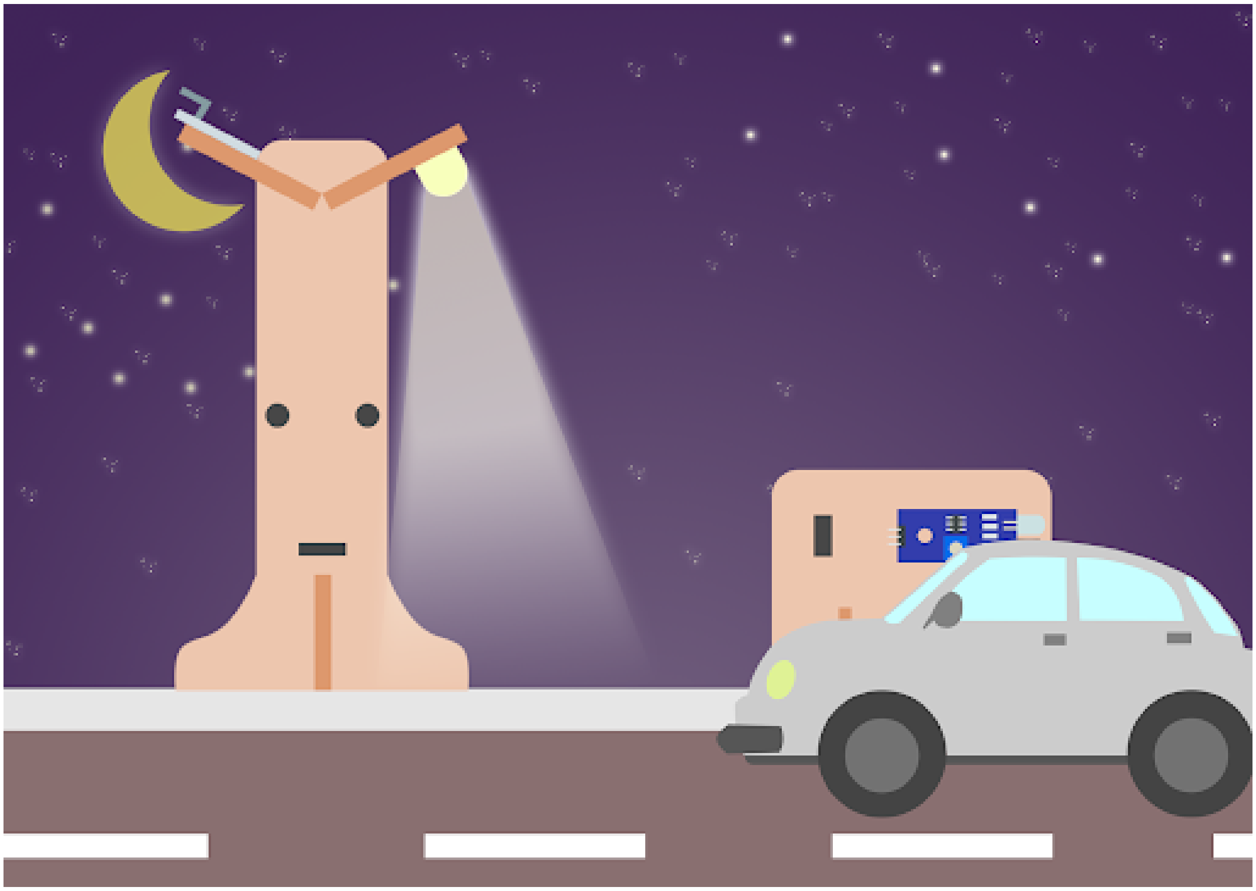
1.2.1 Know More About Smart City

What is a Smart City?

Advantages of Smart City

- More effective, data-driven decision-making
- Enhanced citizen and government engagement
- Safer communities
- Reduced environmental footprint
- Improved transportation

1.2.2 Automated Smart Street Lamp

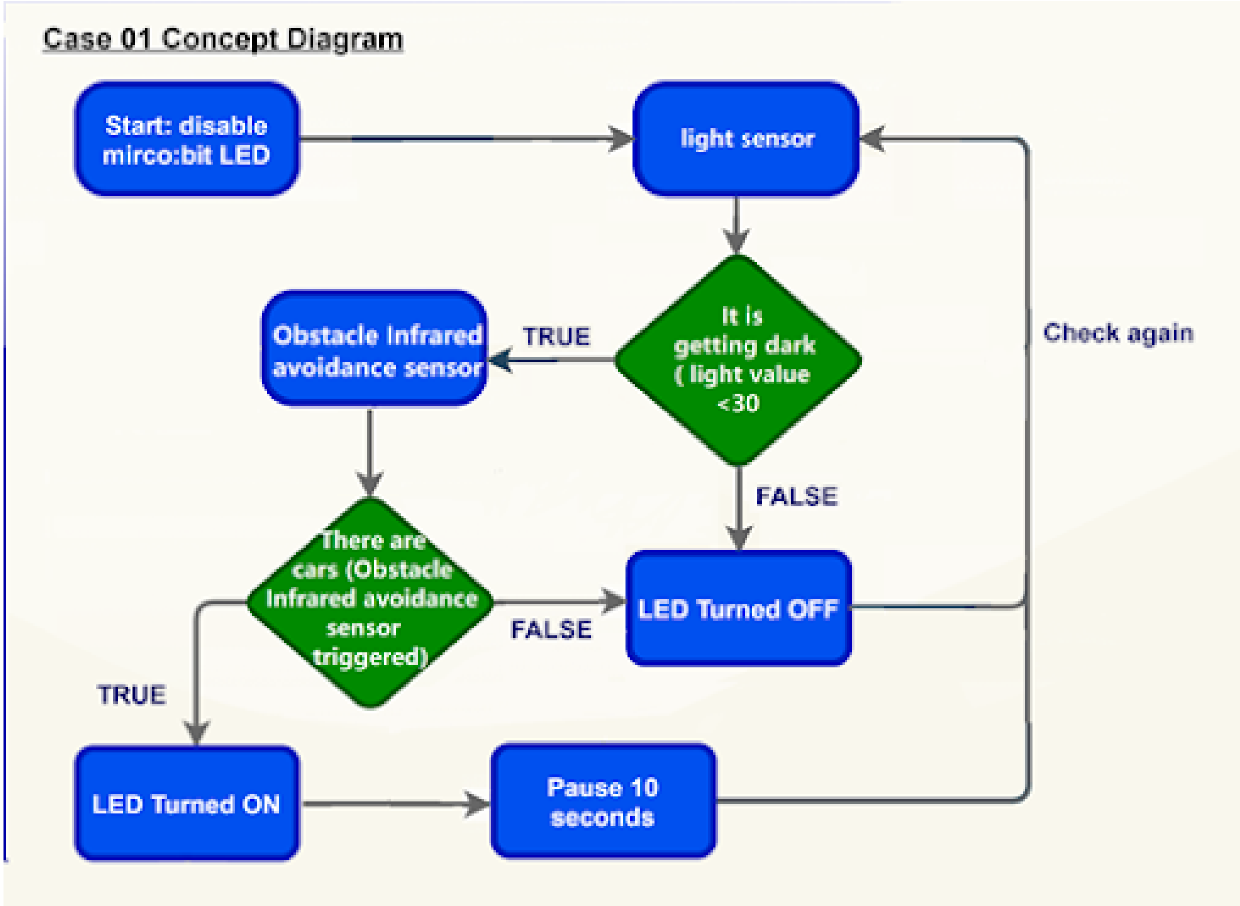


Goal

Background

What is a smart street lamp?

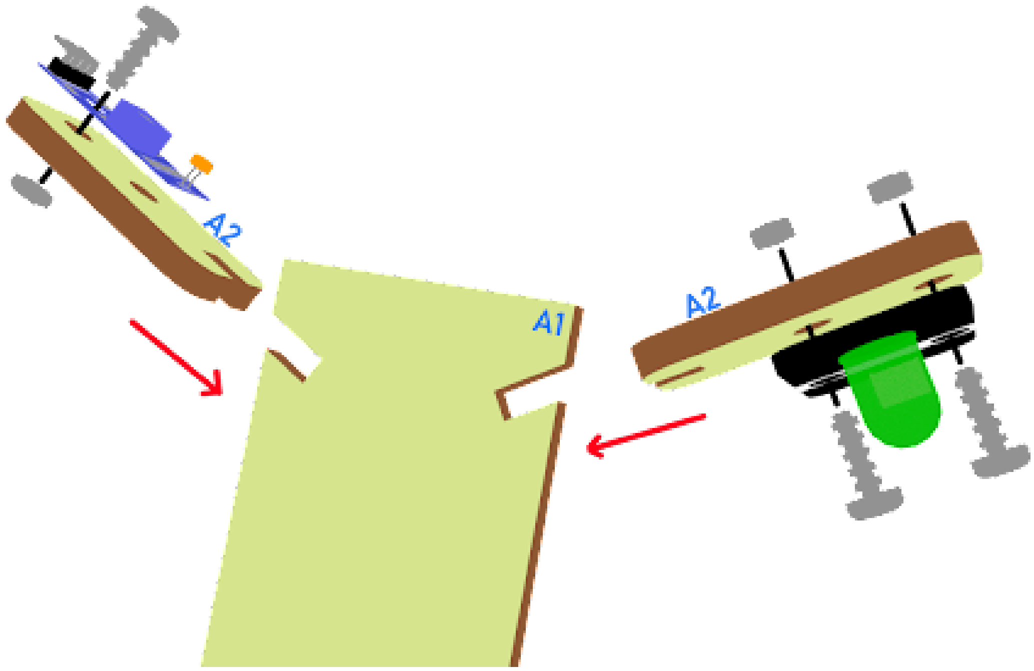
Smart street lamp operation



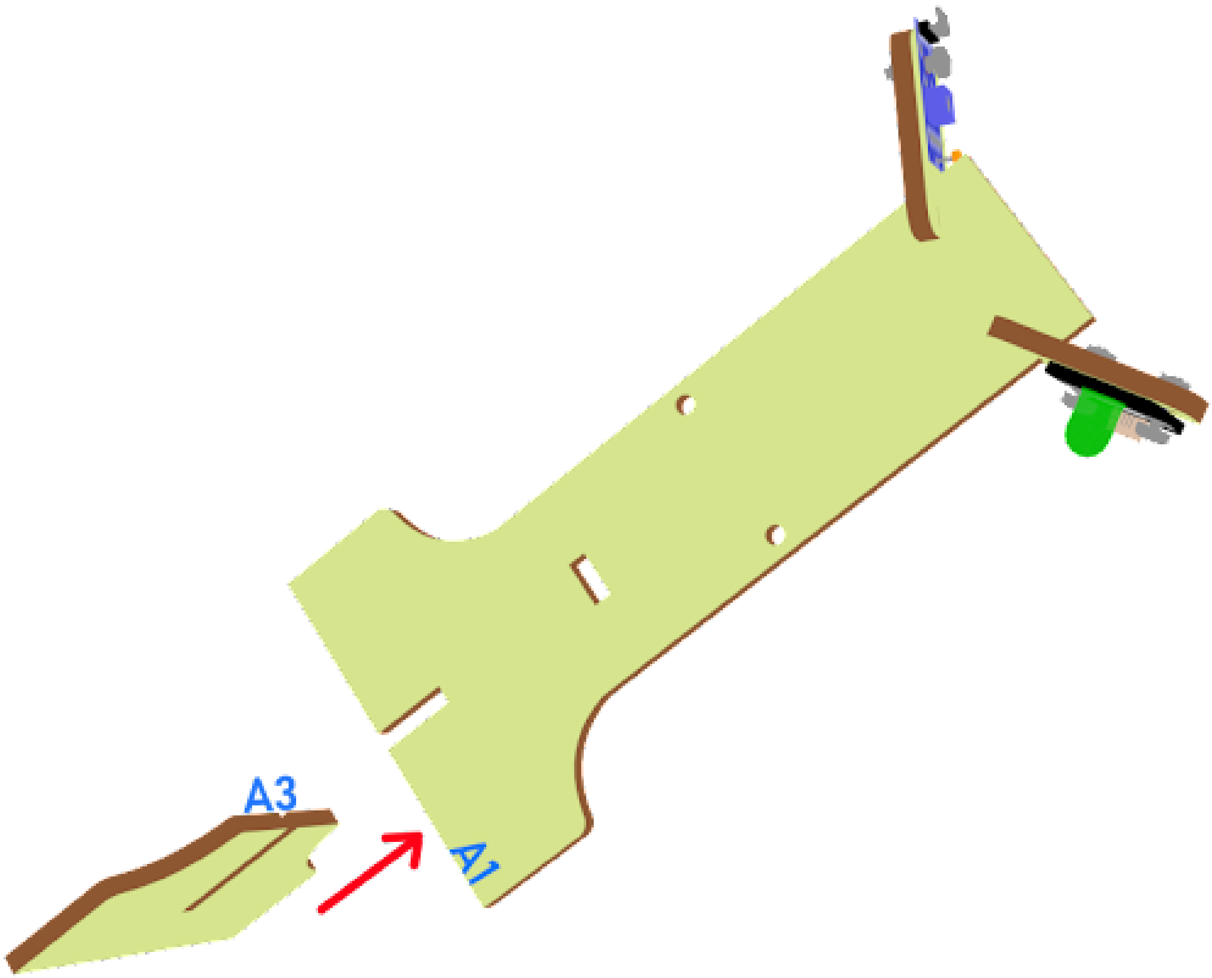
Part List

Assembly step

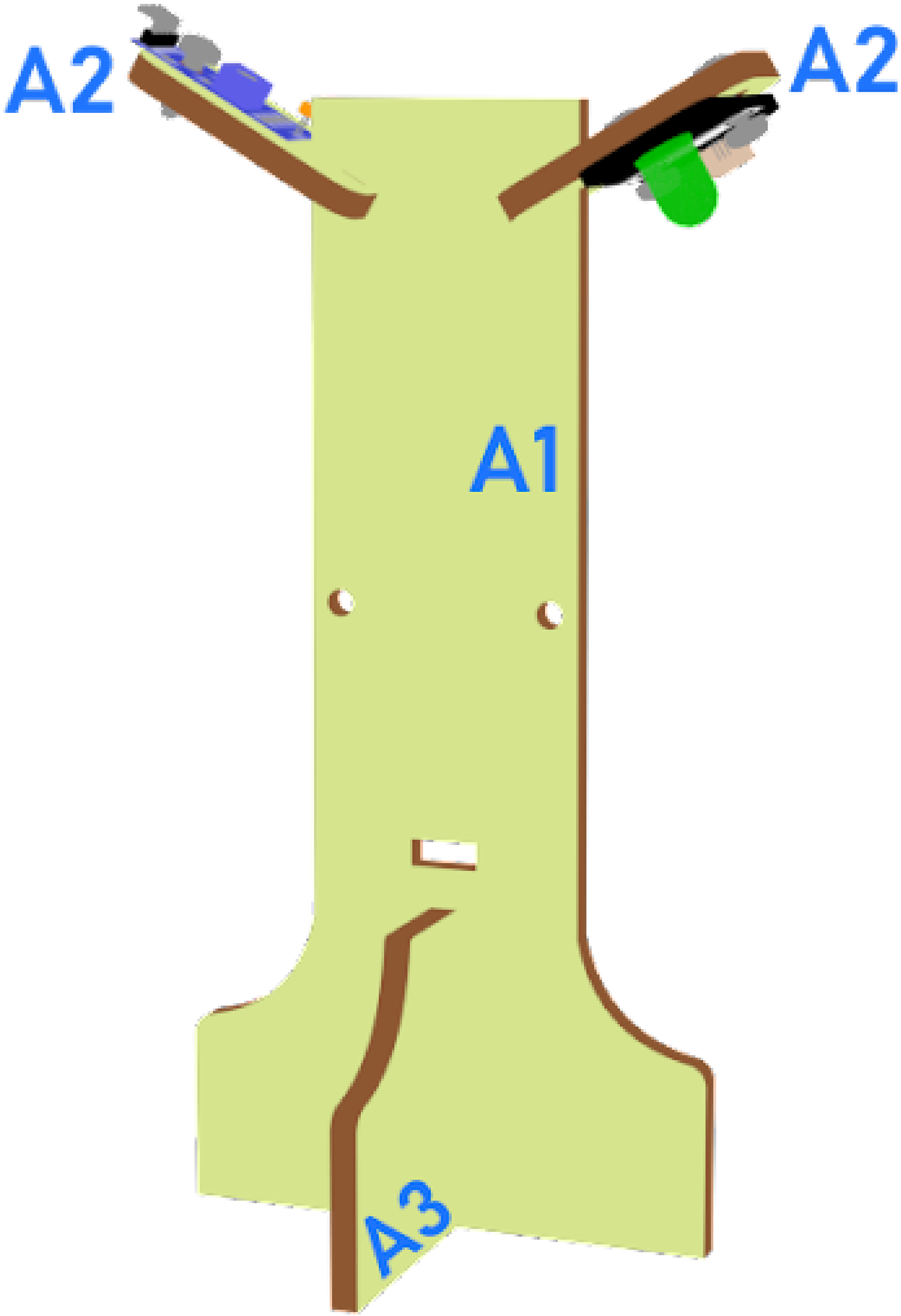
Step 1



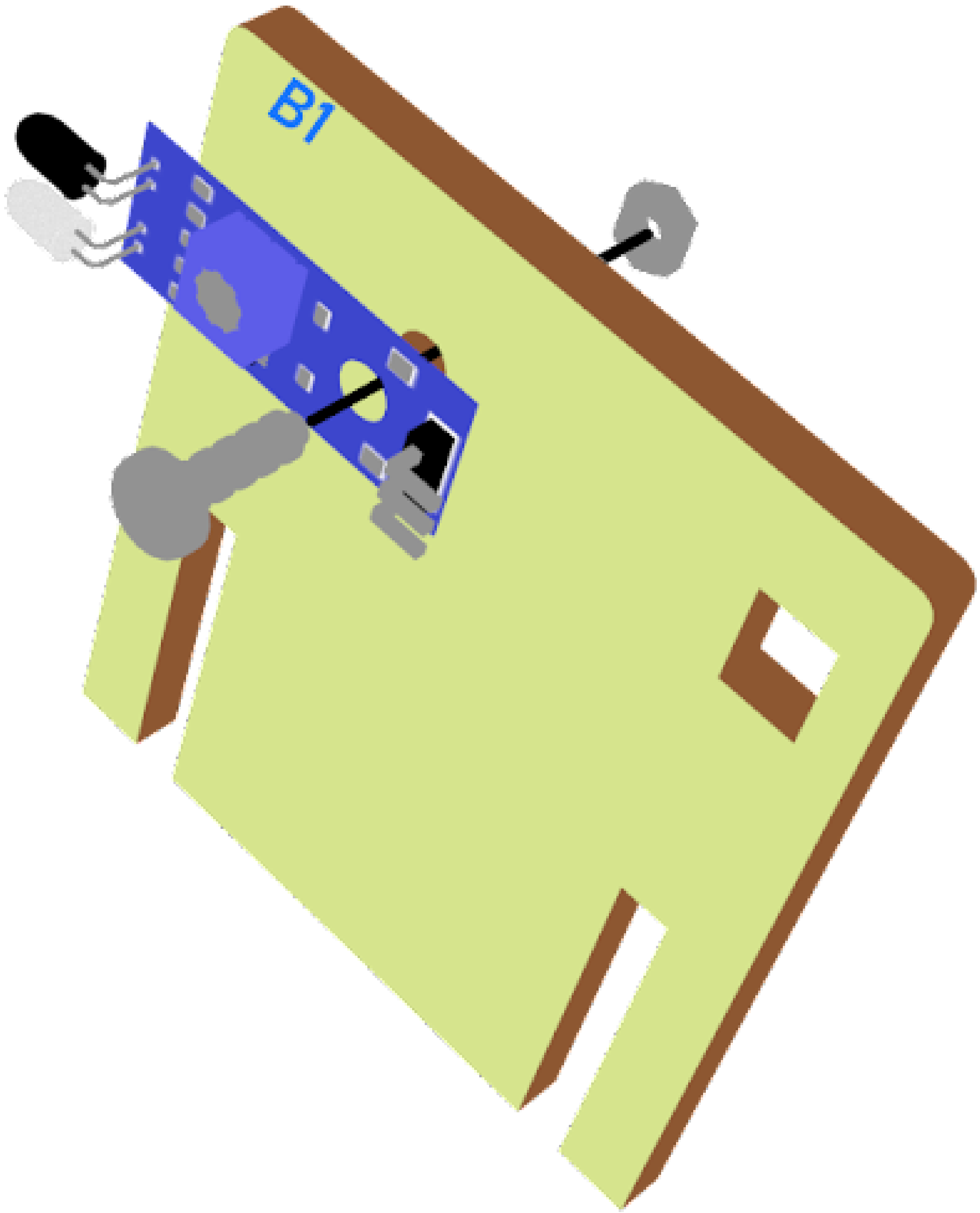
Step 2



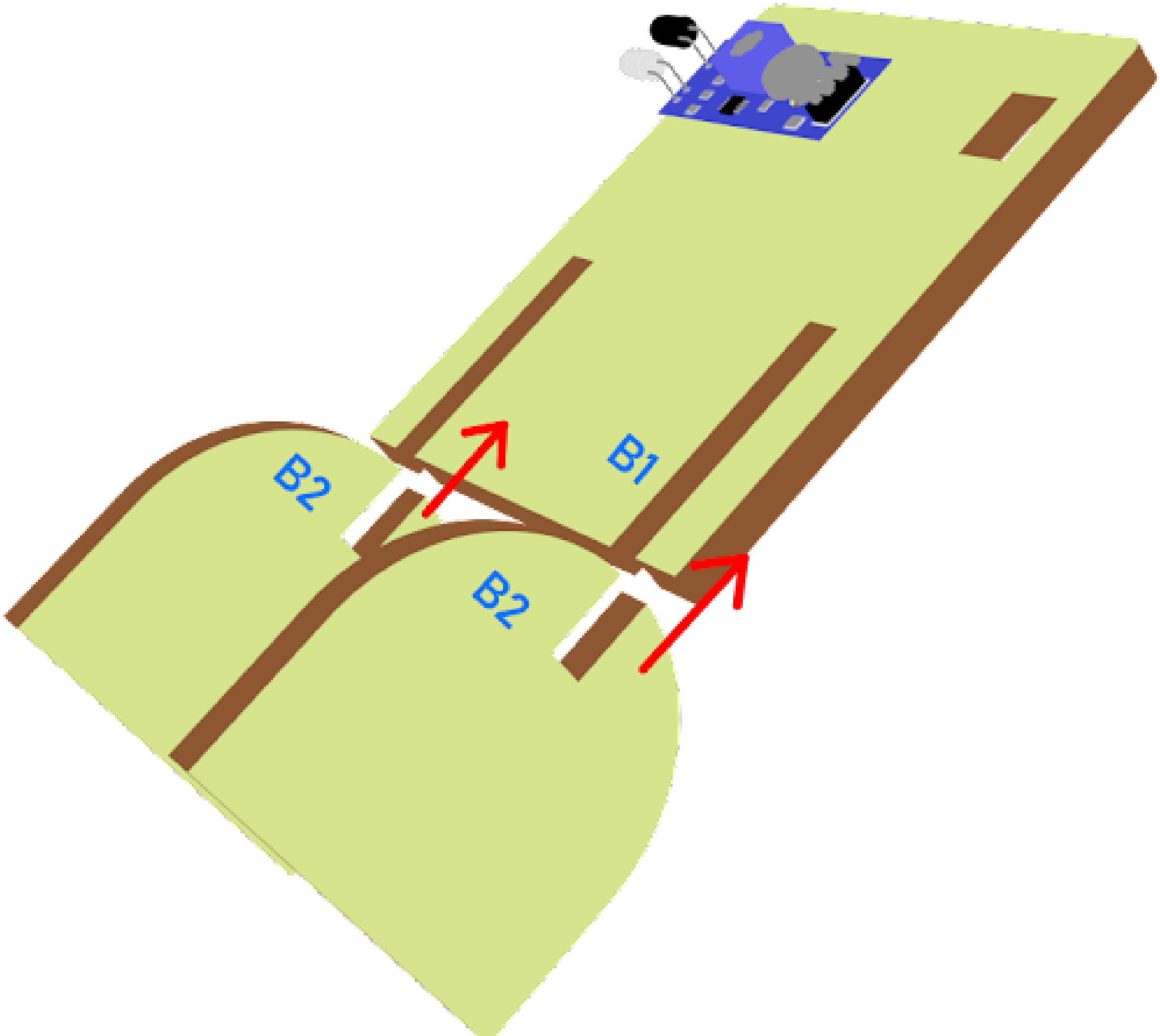
Step 3



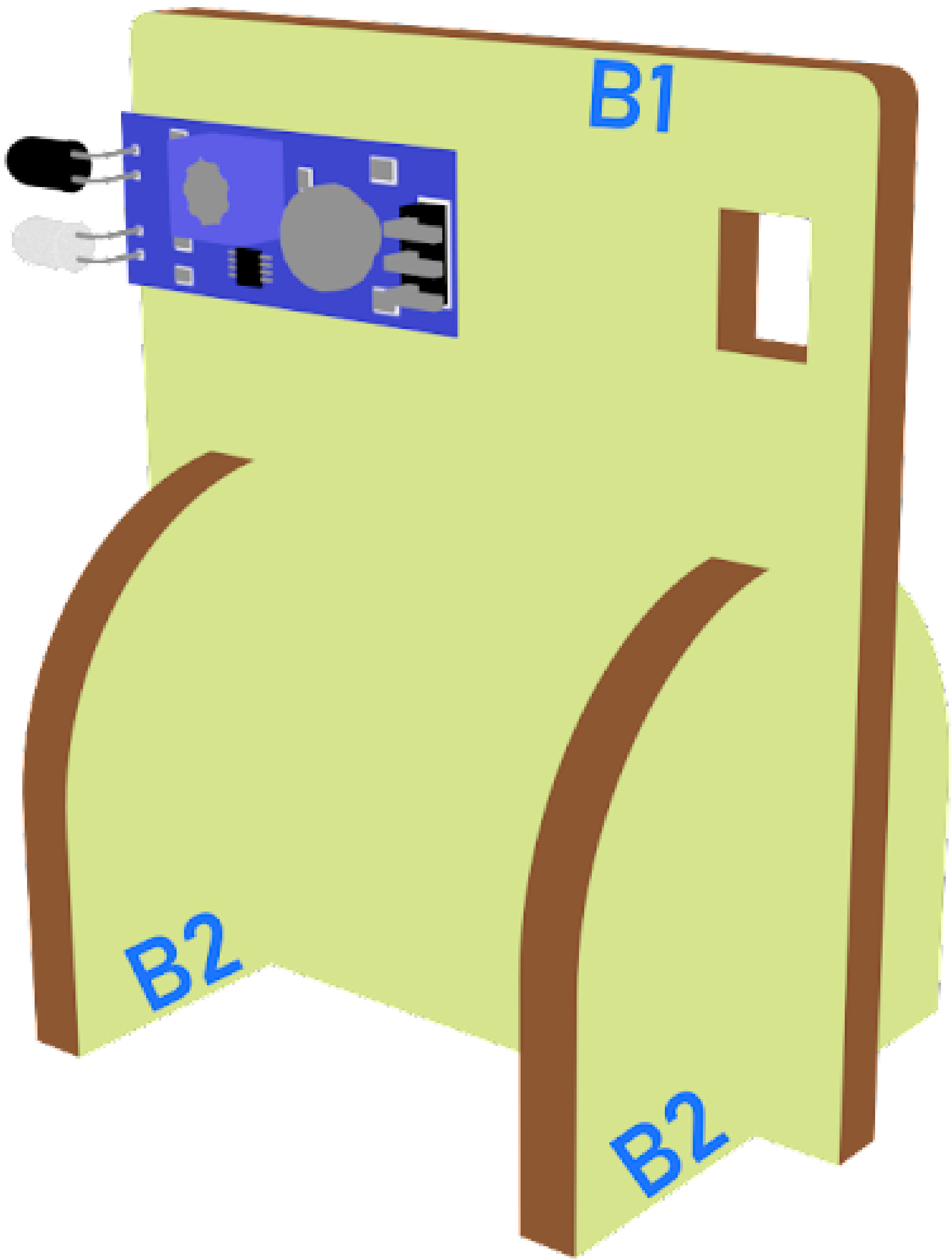
Step 4



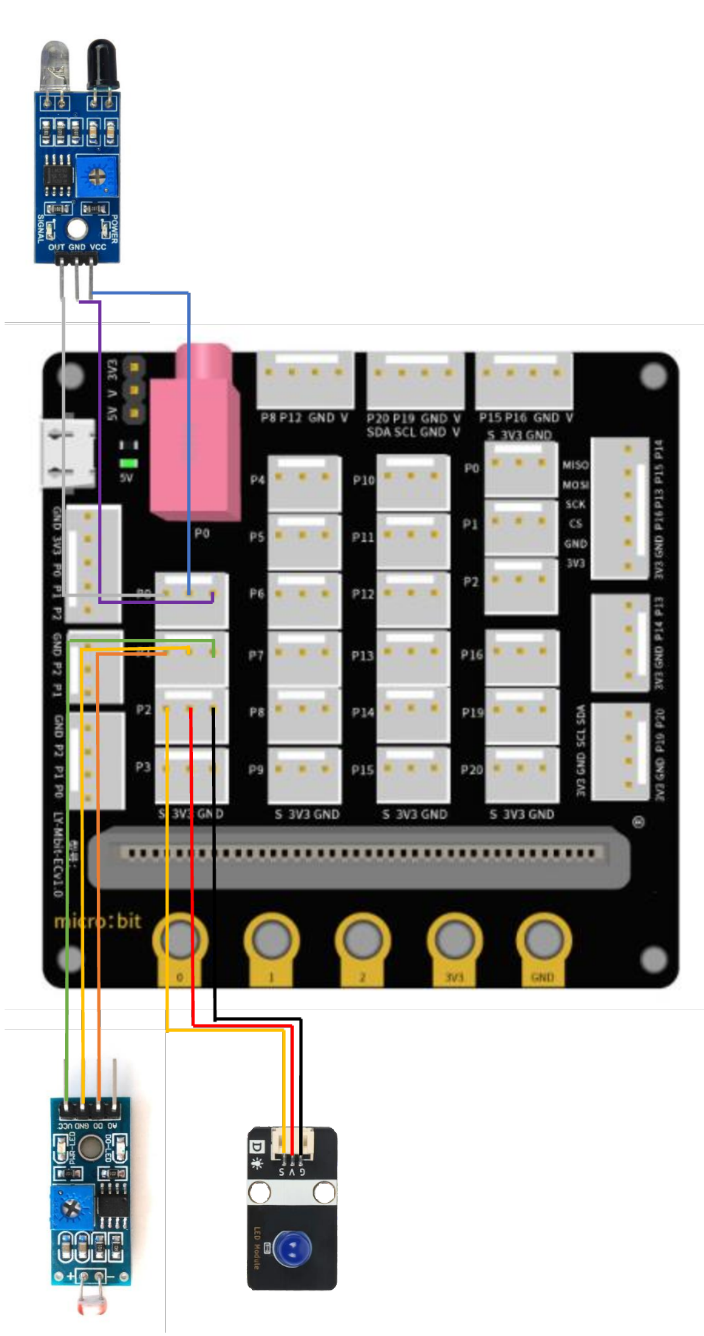
Step 5

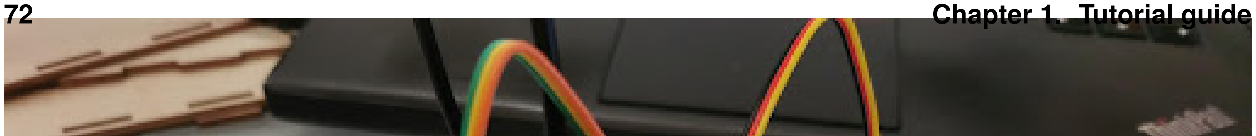


Step 6



Hardware connect

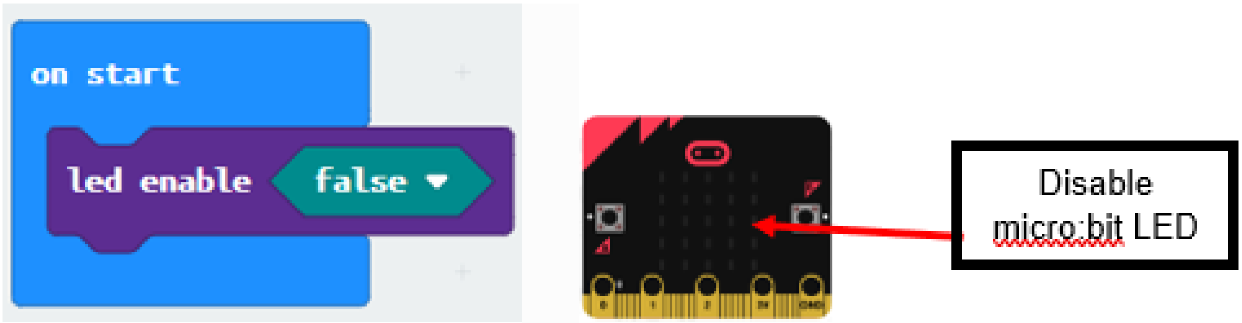




Programming (MakeCode)

Step 1. Disable micro:bit LED.

- Snap led enable false to on start
- Note that P3 is used as LED in default setting, LED need to be disable



Step 2. Turn on LED by light sensor and obstacle Infrared avoidance sensor

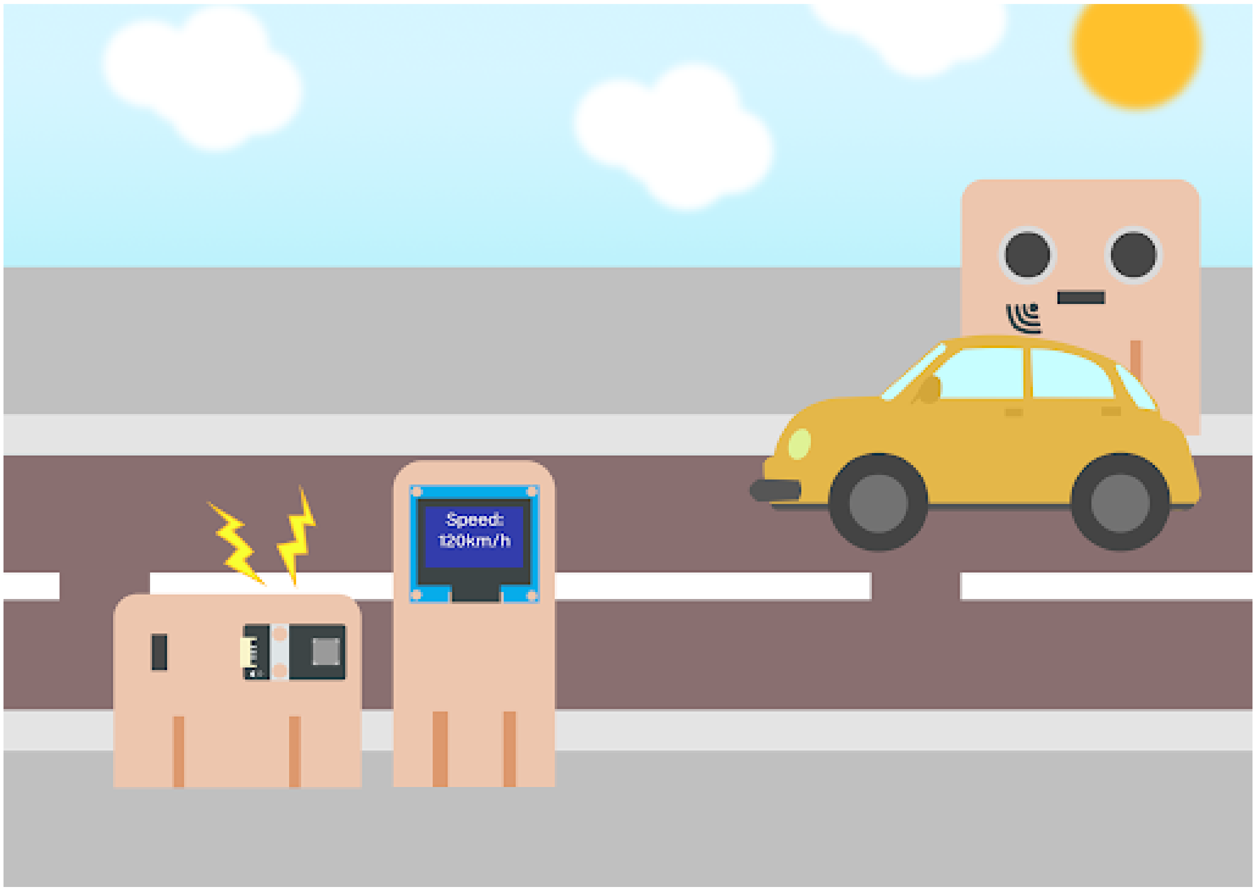
- Drag forever from Basic
- Snap if statement into forever
- Set get light value (percentage) at P1 <40 and get motion (triggered or not) at P0 = true, into if statement that says motion is triggered, someone passes by.
- Then, turn white LED to 1023 at P2 as turning on white LED and pause 10 seconds.
- Else, turn white LED at P2 to 0 as turning off.


forever
if
Get light value (percentage) at Pin P1
<
40
then
if
Get motion (triggered or not) at Pin P0
then
Turn White LED to 1023 at P2
pause (ms) 2000
else
Turn White LED to 0 at P2
else
Turn White LED to 0 at P2

Result

Think

1.2.3 Car Speed Monitoring



Goal

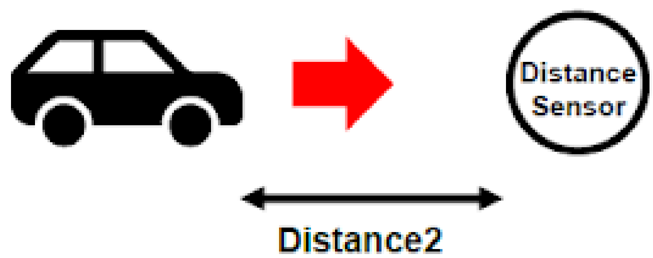
background

What is car speed monitoring?

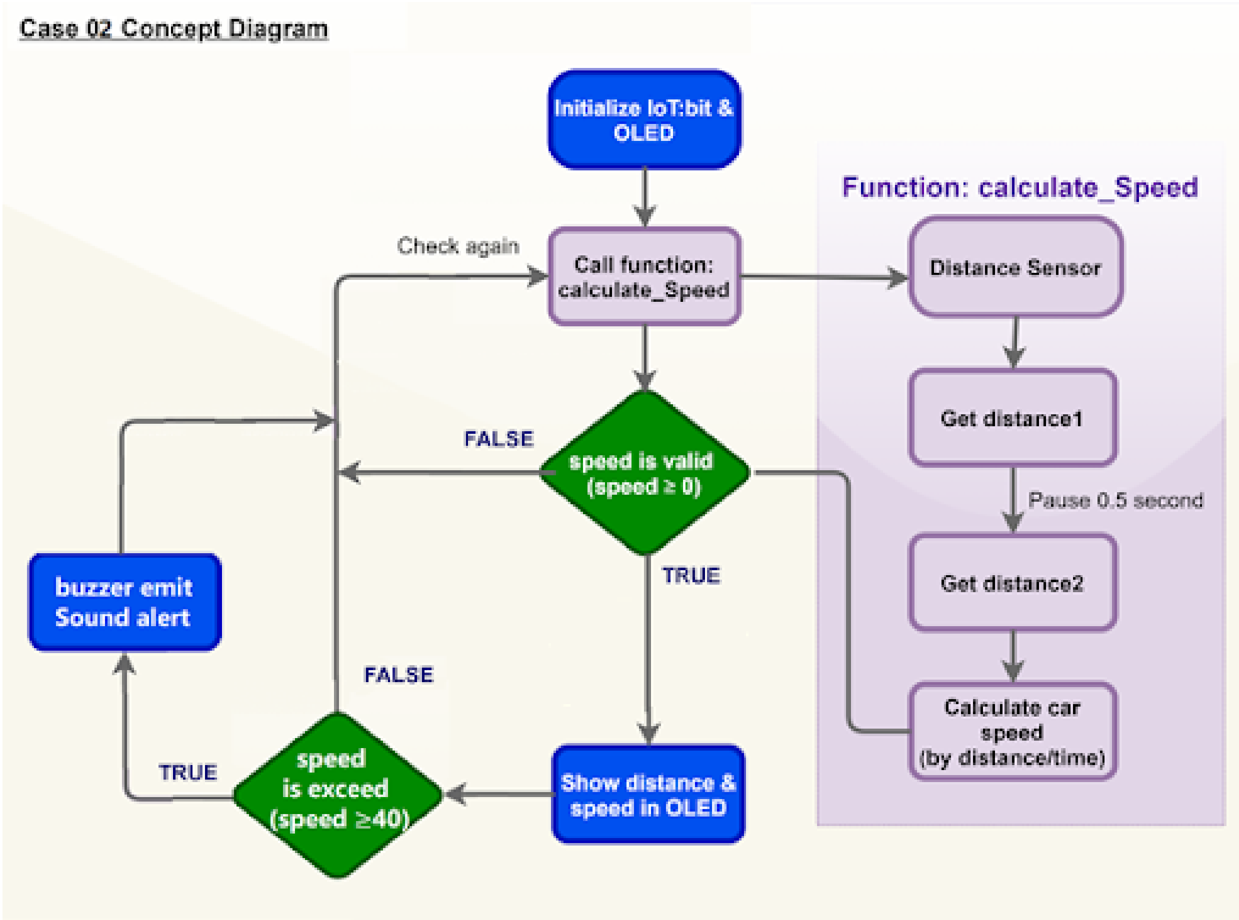
Car speed monitor operation

Distance1: initial distance

Distance2: distance after 0.5s



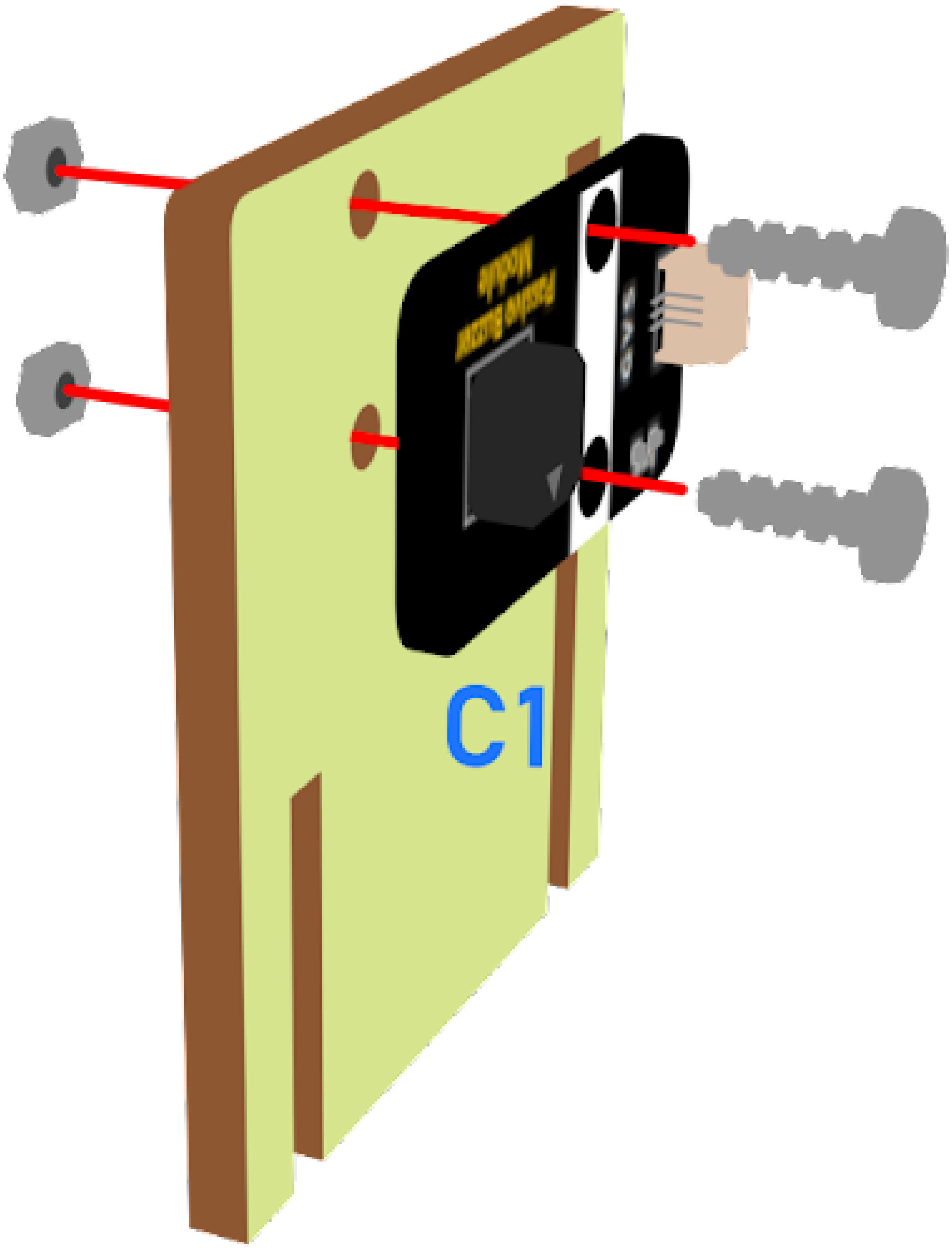
By equation, $\text{Speed} = \frac{\text{distance}}{\text{time}}$,we found that $\text{Speed} = \frac{\text{distance1} - \text{distance2}}{\text{time}}$



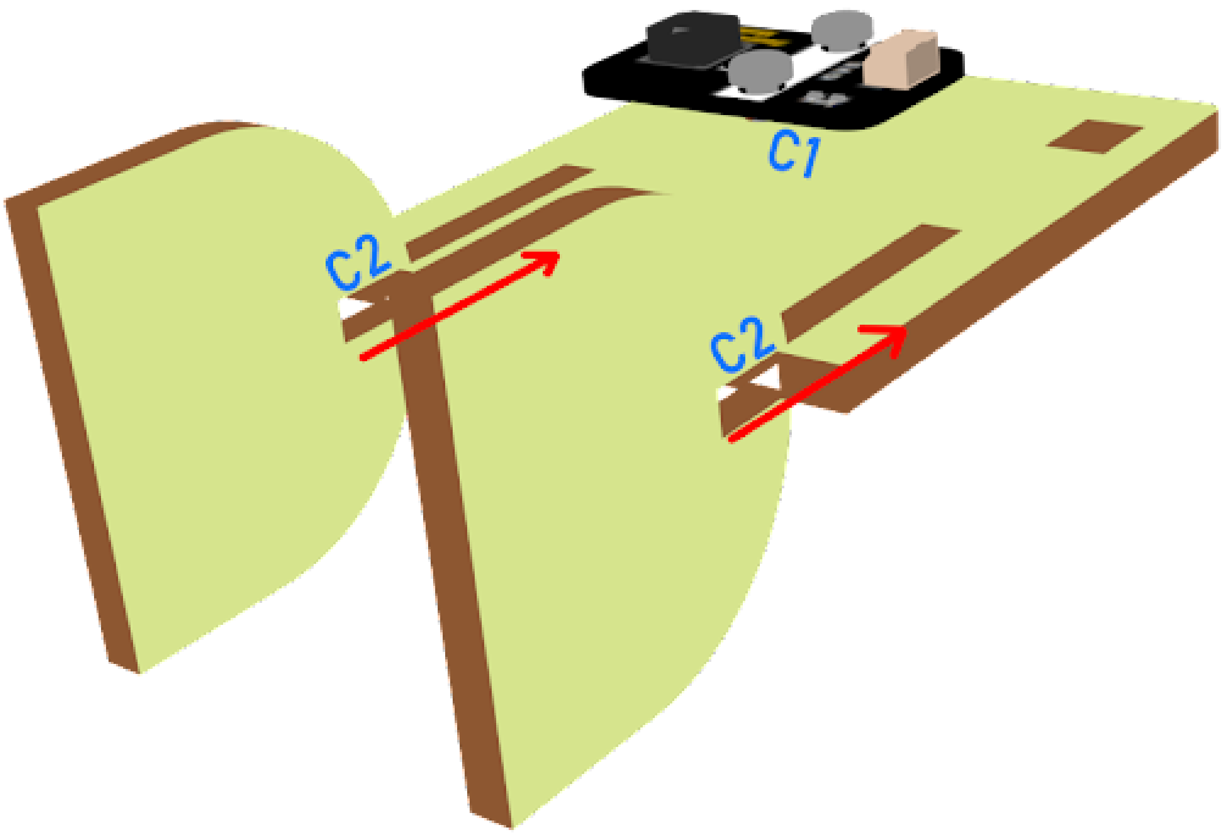
Part List

Assembly step

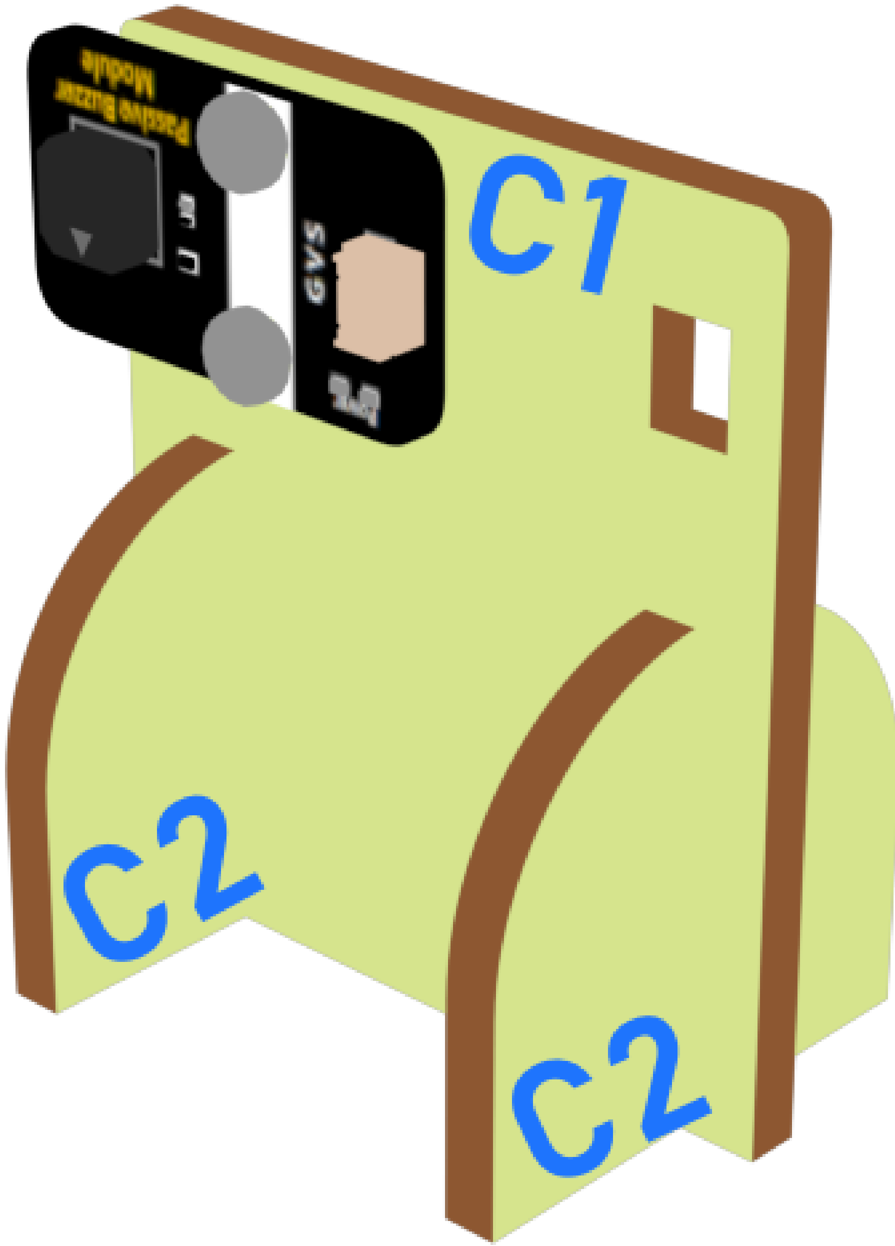
Step 1



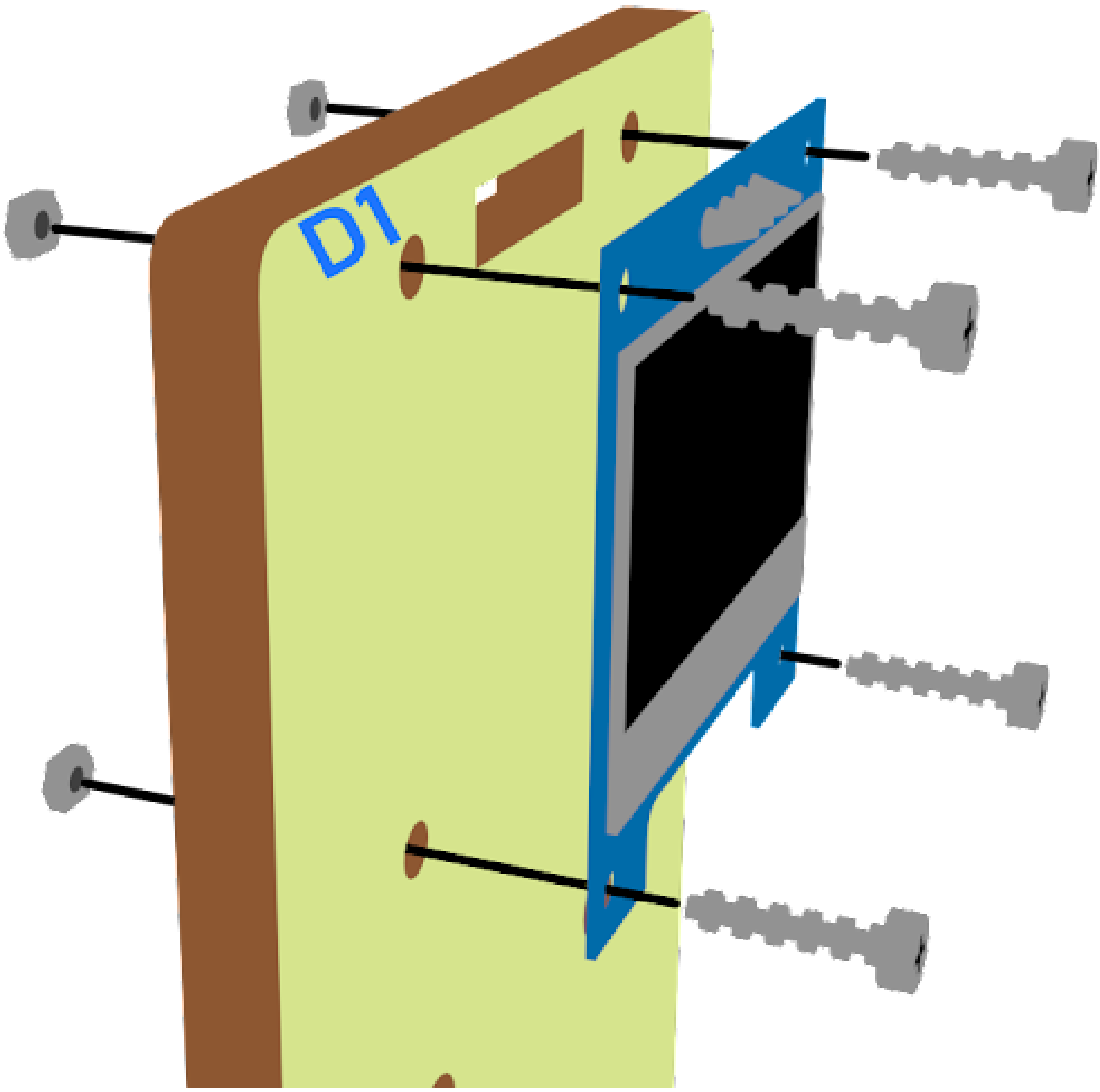
Step 2



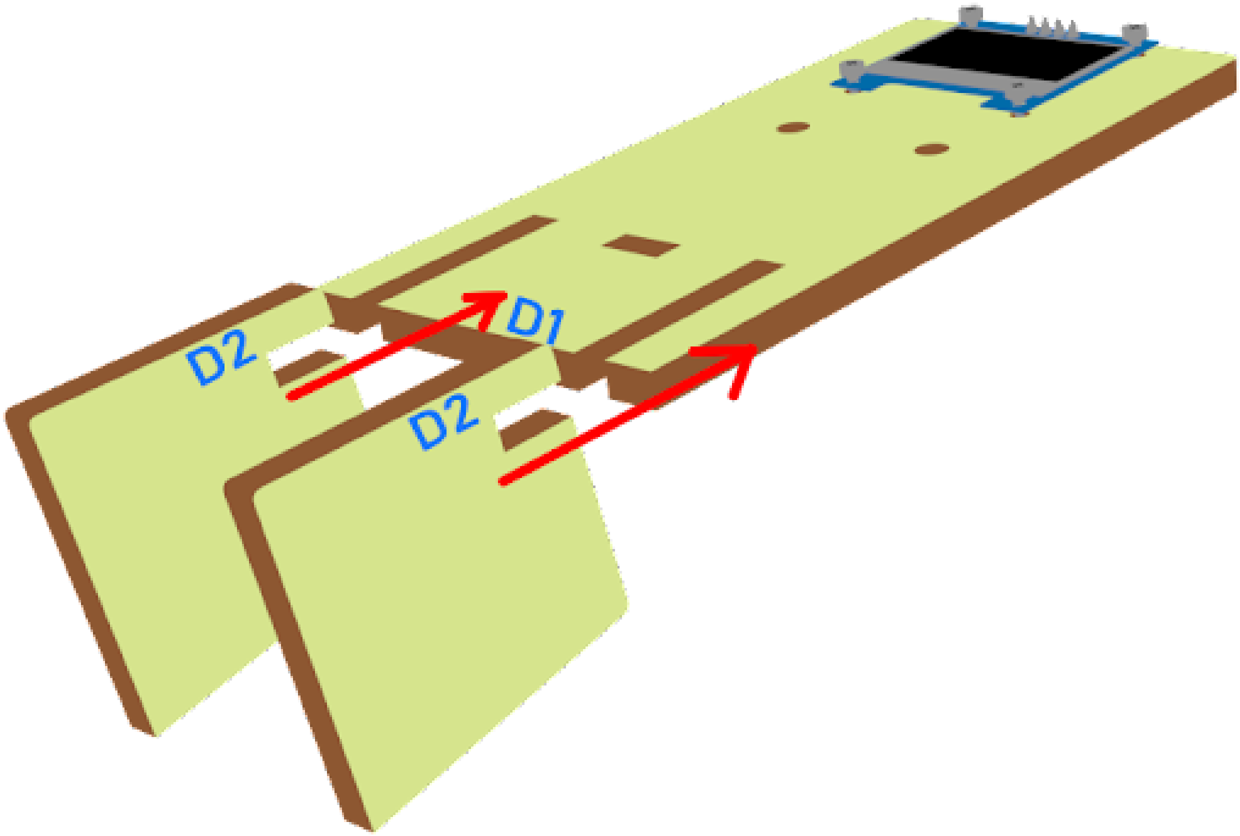
Step 3



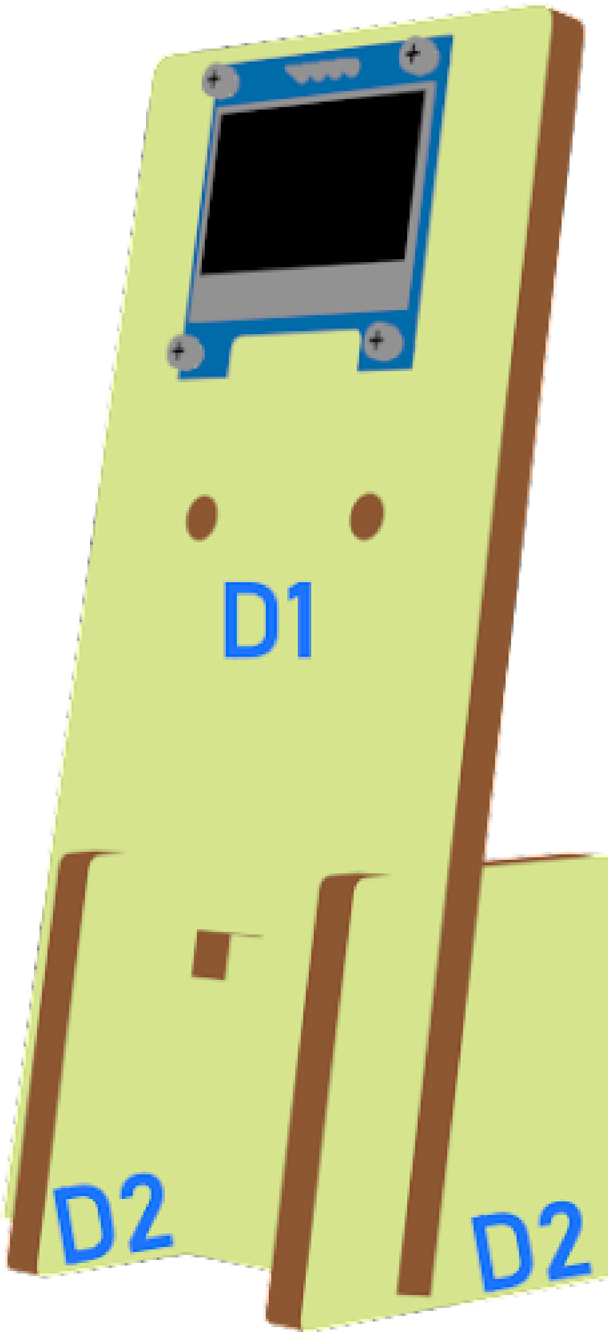
Step 4



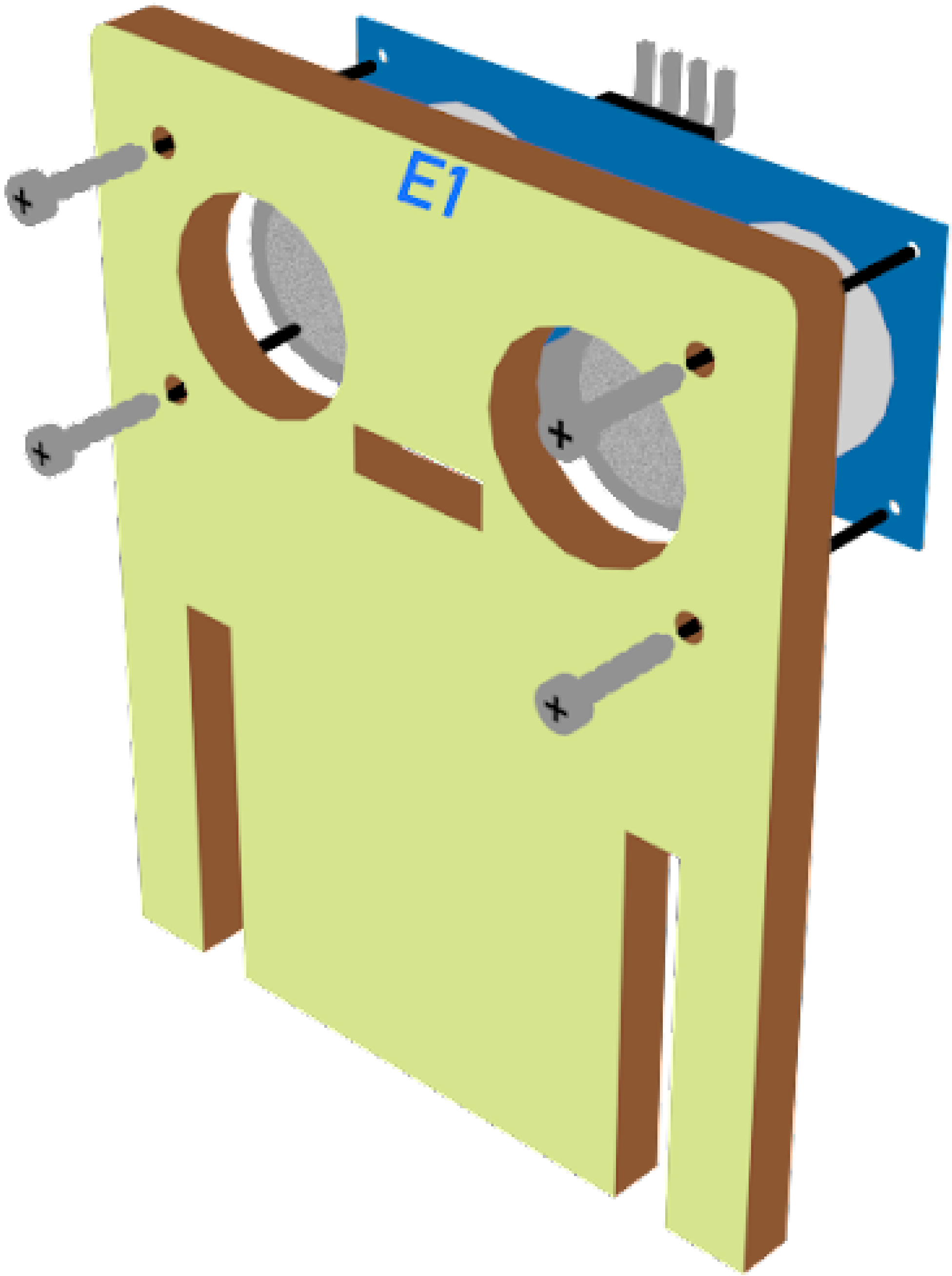
Step 5



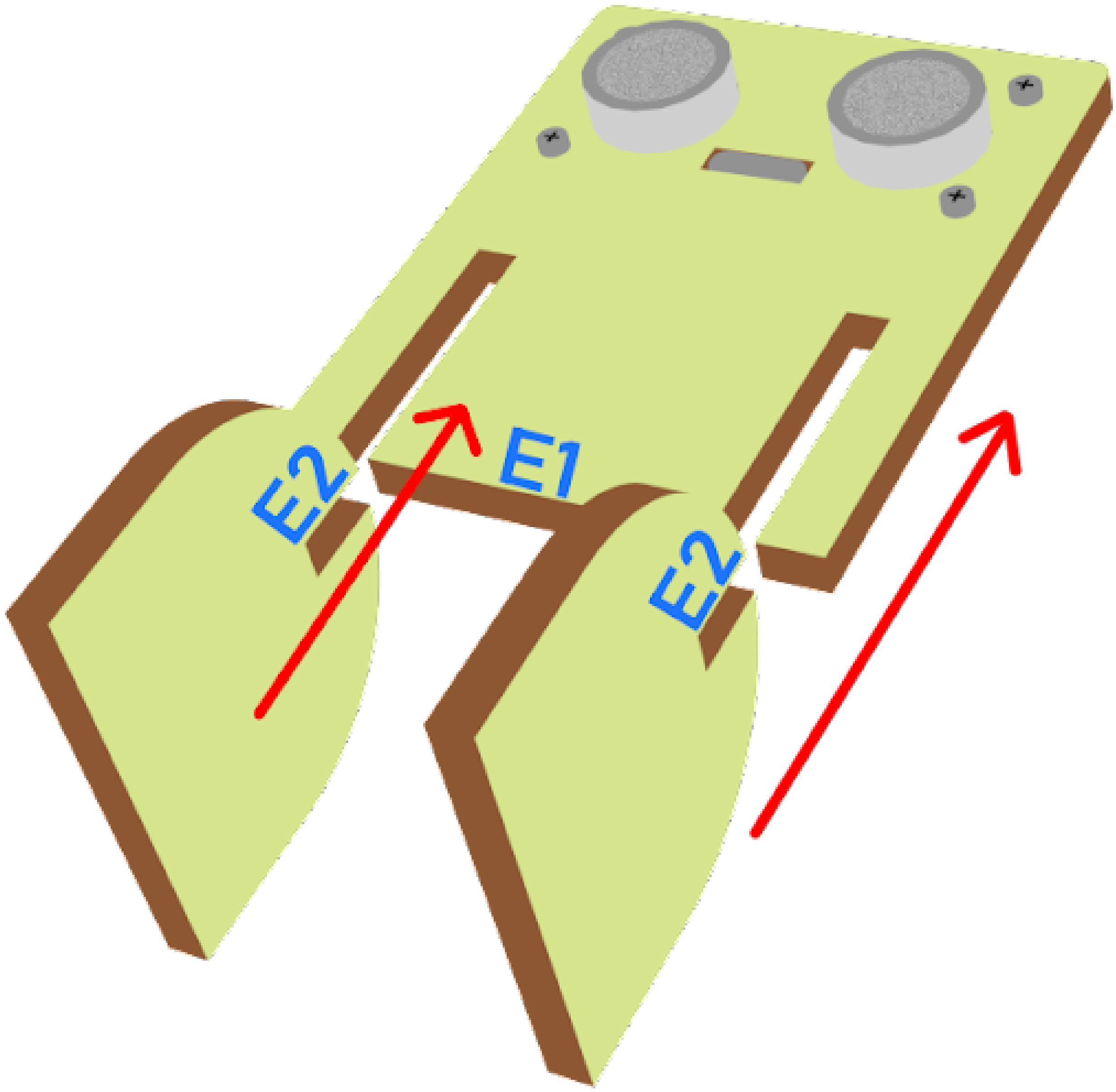
Step 6



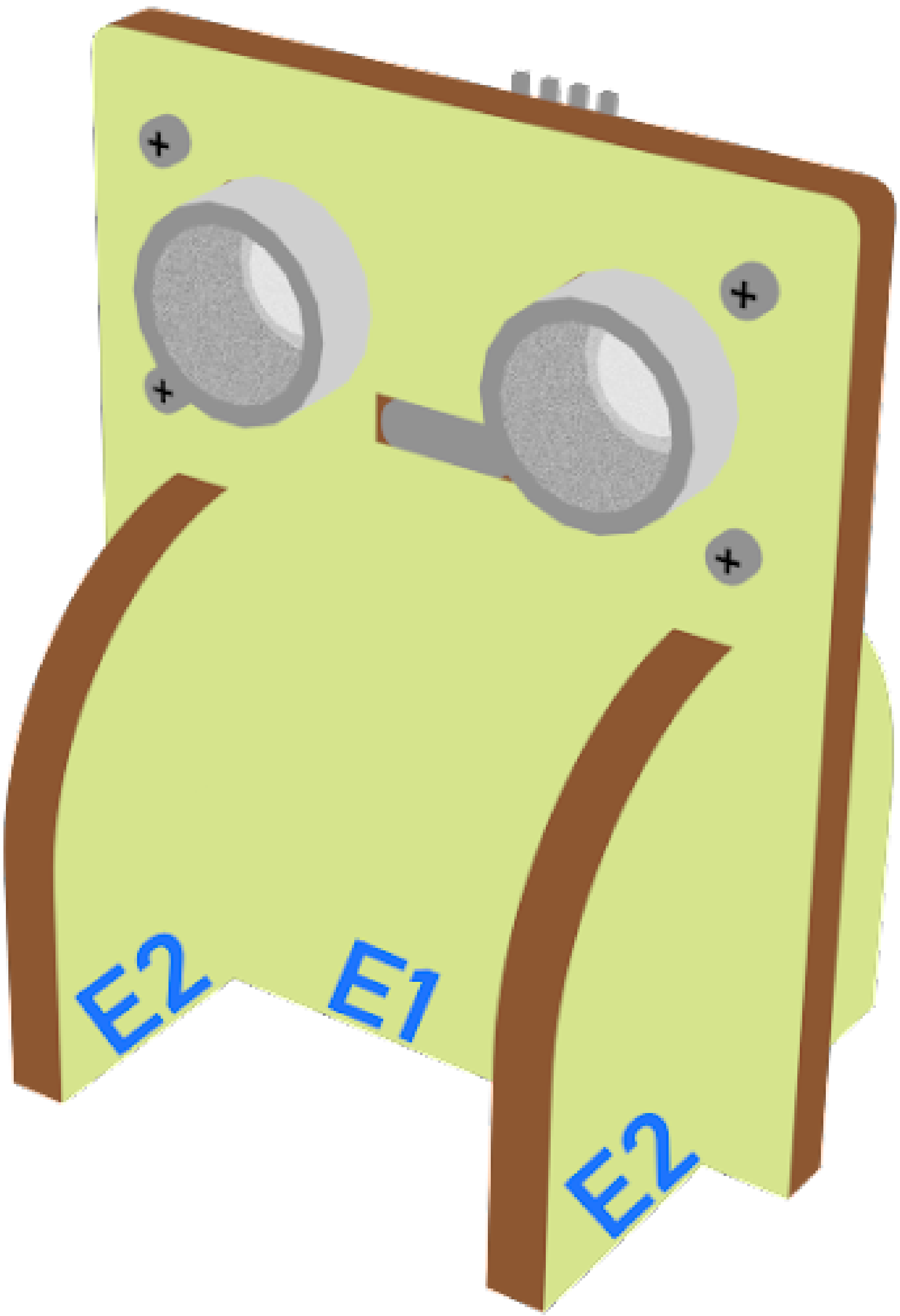
Step 7



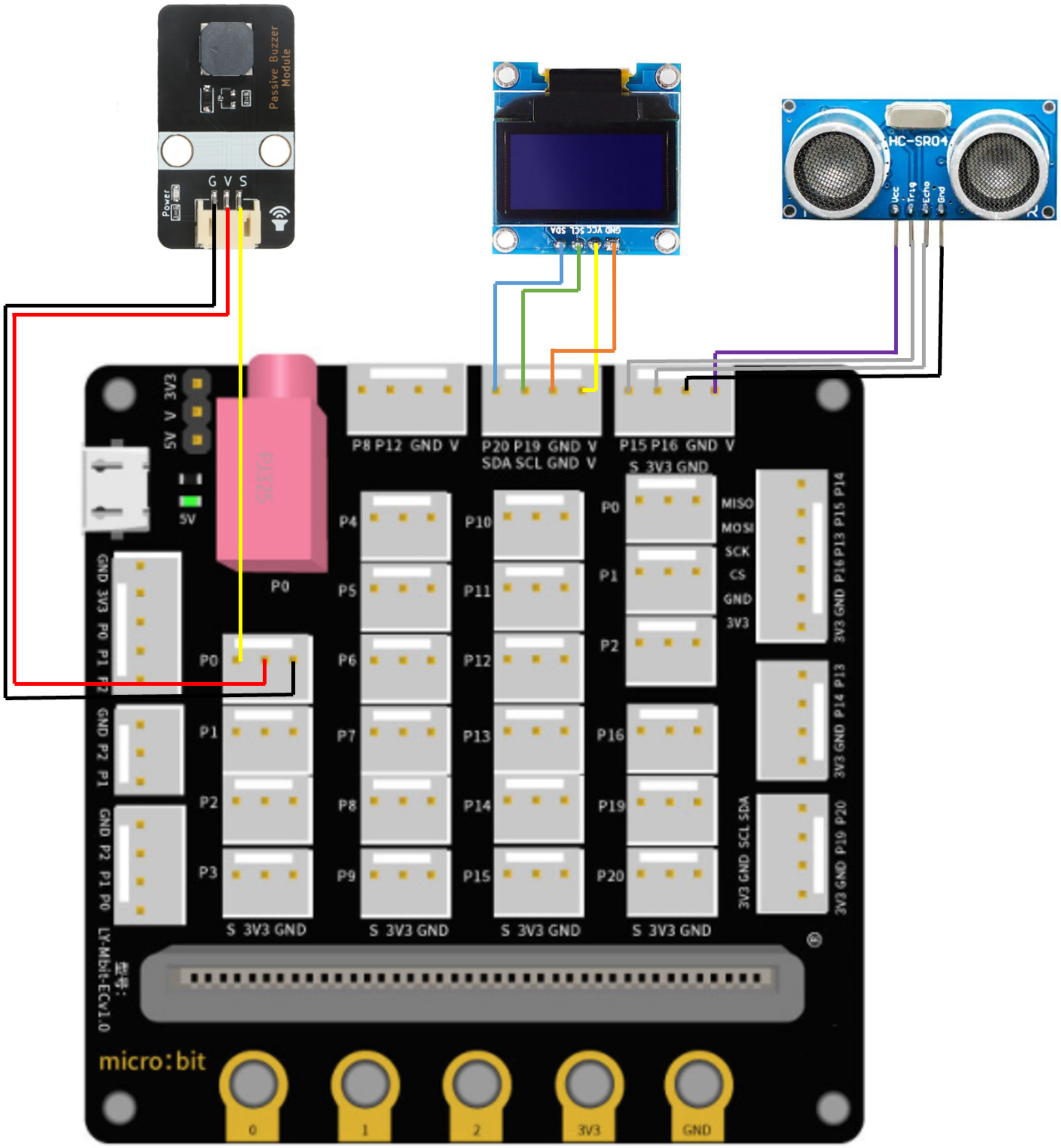
Step 8

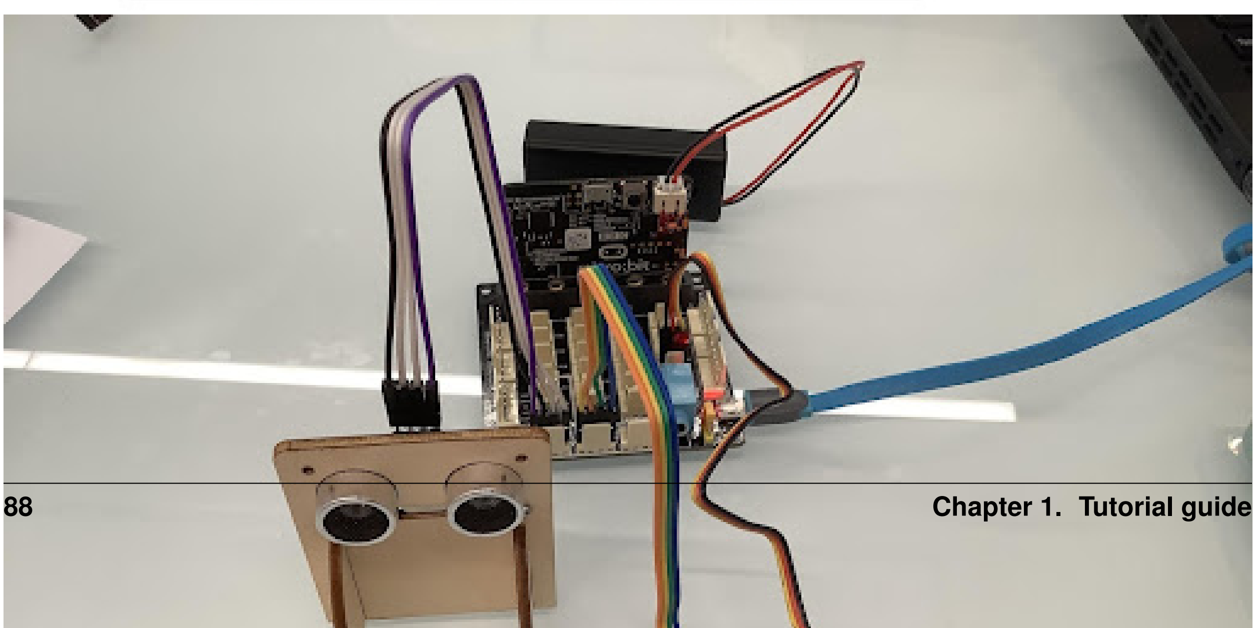


Step 9



Hardware connect

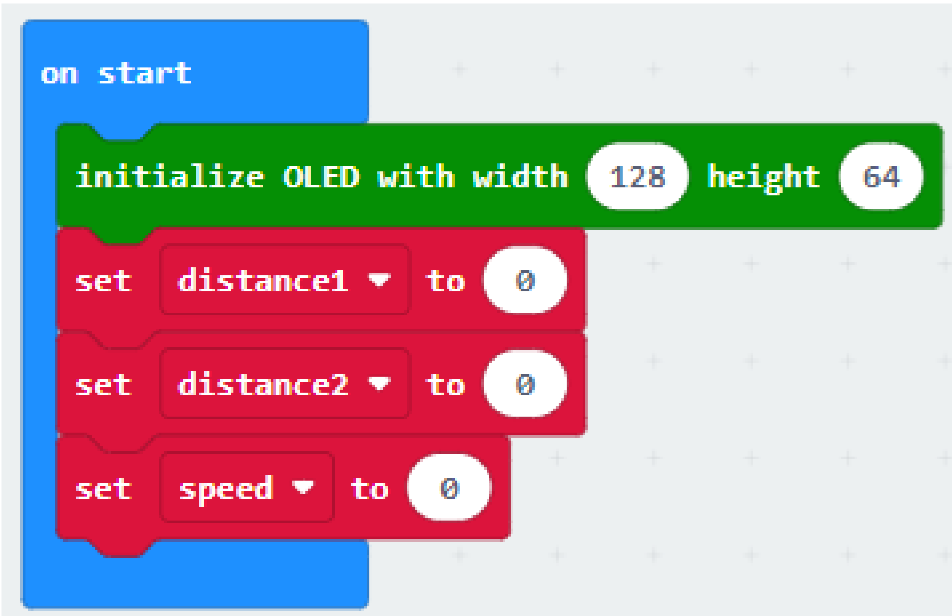




Programming (MakeCode)

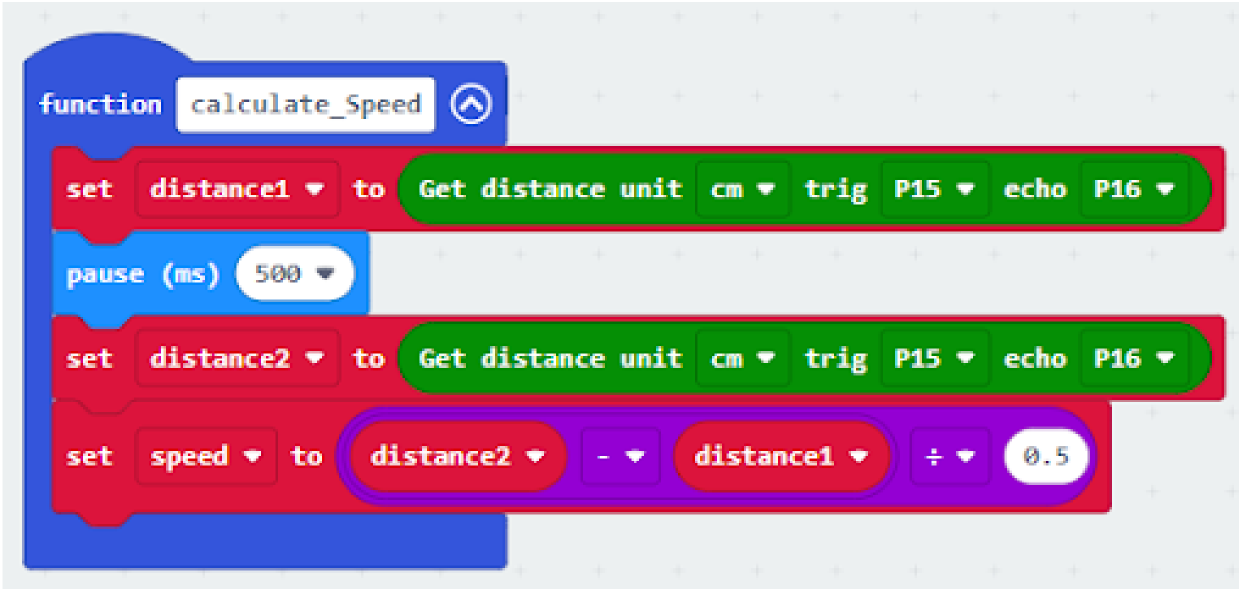
Step 1. Initialize OLED screen

- Drag Initialize OLED with width:128, height: 64 to on start
- Set distance1, distance2 and speed to 0 from variables



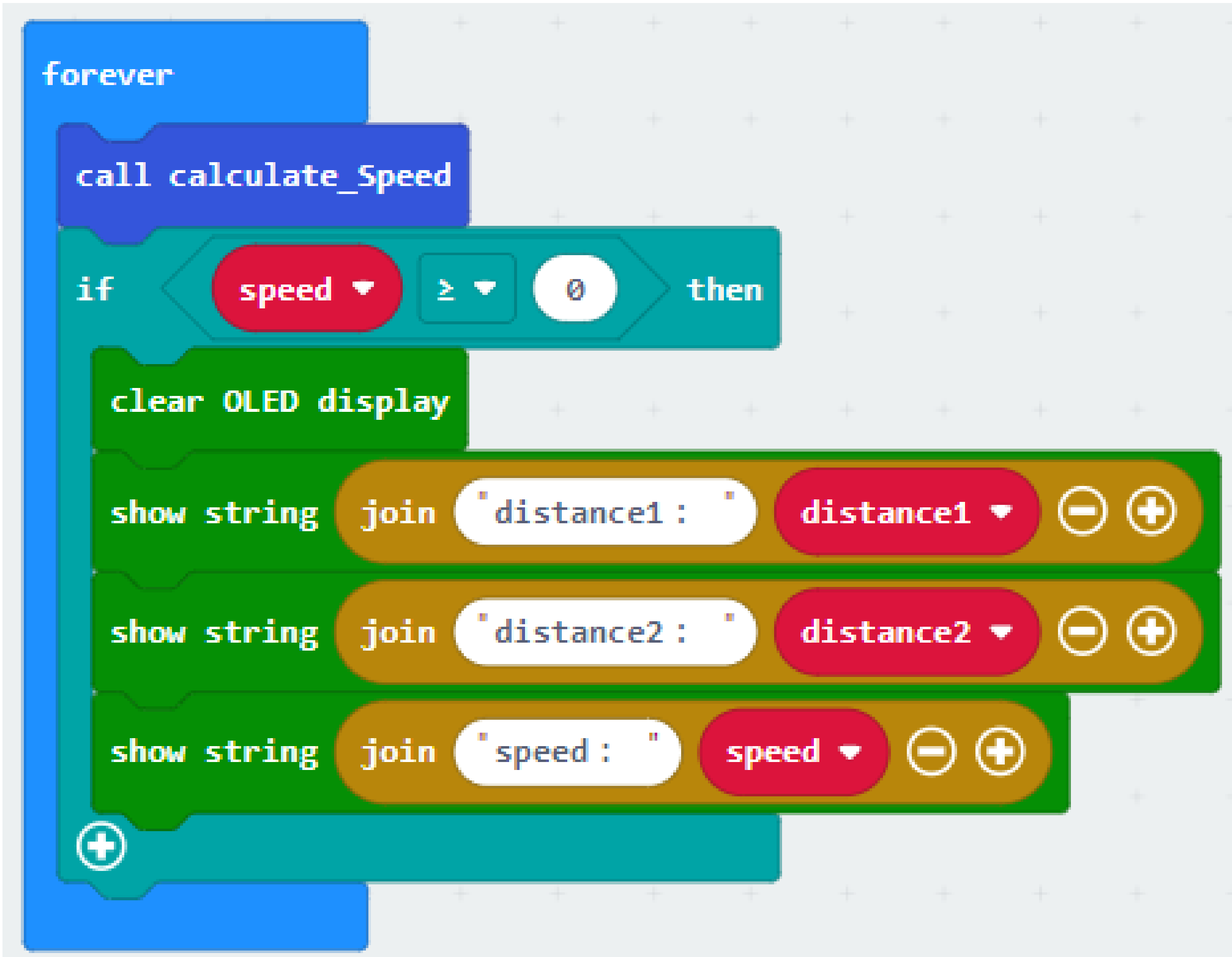
Step 2. Set up function (calculate_Speed)

- Set up a new function calculate_Speed from Advanced > Functions.
- Set distance1 to get distance unit cm trig P14 echo P15 (distance from the car to the distance sensor before 0.5 second) Drag Pause to wait 500ms and set distance2 to get distance unit cm trig P14 echo P15 (distance from the car to the distance sensor after 0.5 second)
- By the equation of speed = distance / time. We get the speed of the moving car to (distance1-distance2)/0.5 (unit: cm/s)



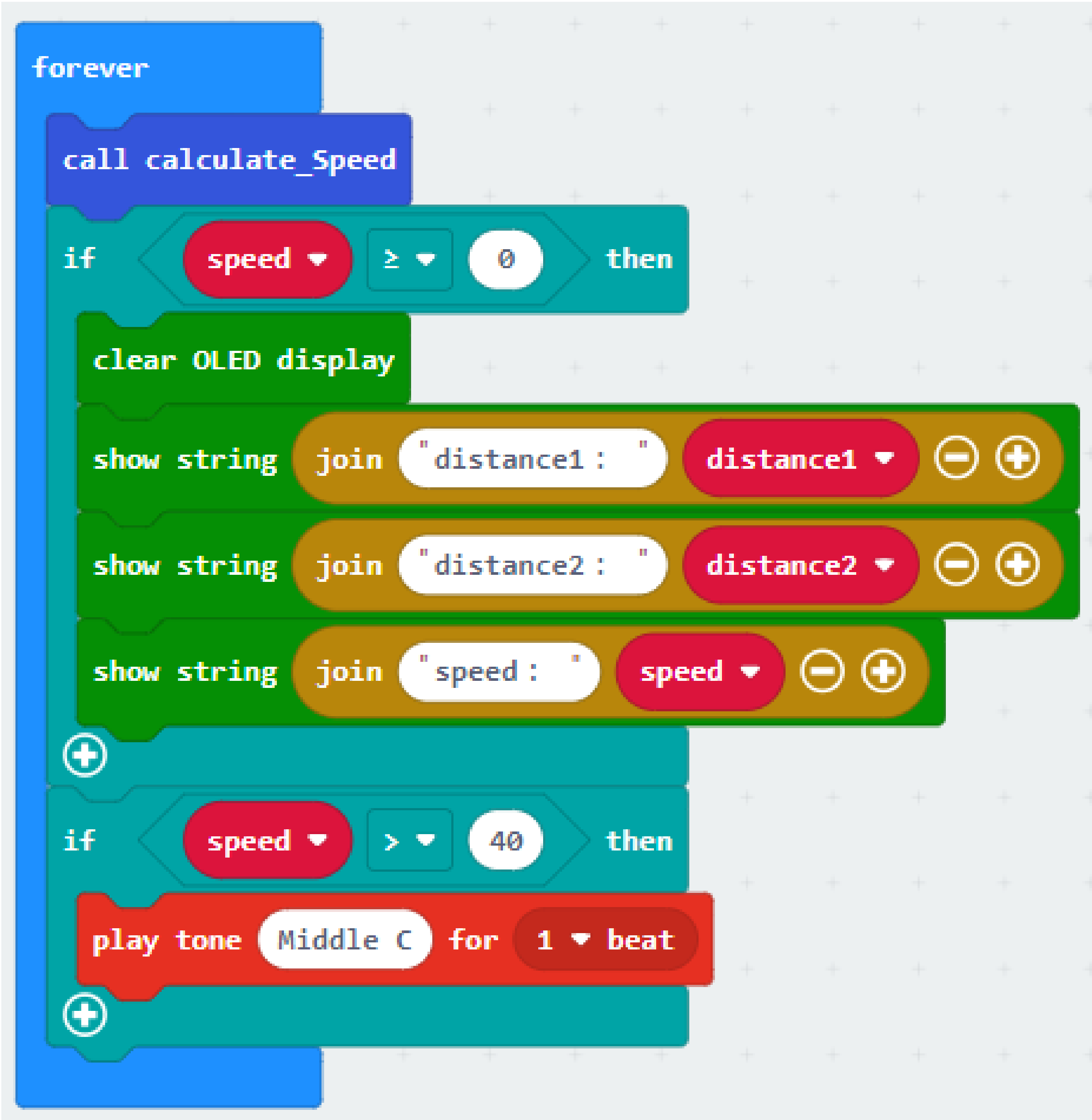
Step 3. Calculate car speed

- In block forever, call function calculate_Speed from Advanced > Functions to get the speed of the moving car
- Snap If statement into the loop
- Snap clear OLED display from OLED to avoid overlap
- Snap show string and show value of variables distance1, distance2 and speed



Step 4. buzzer

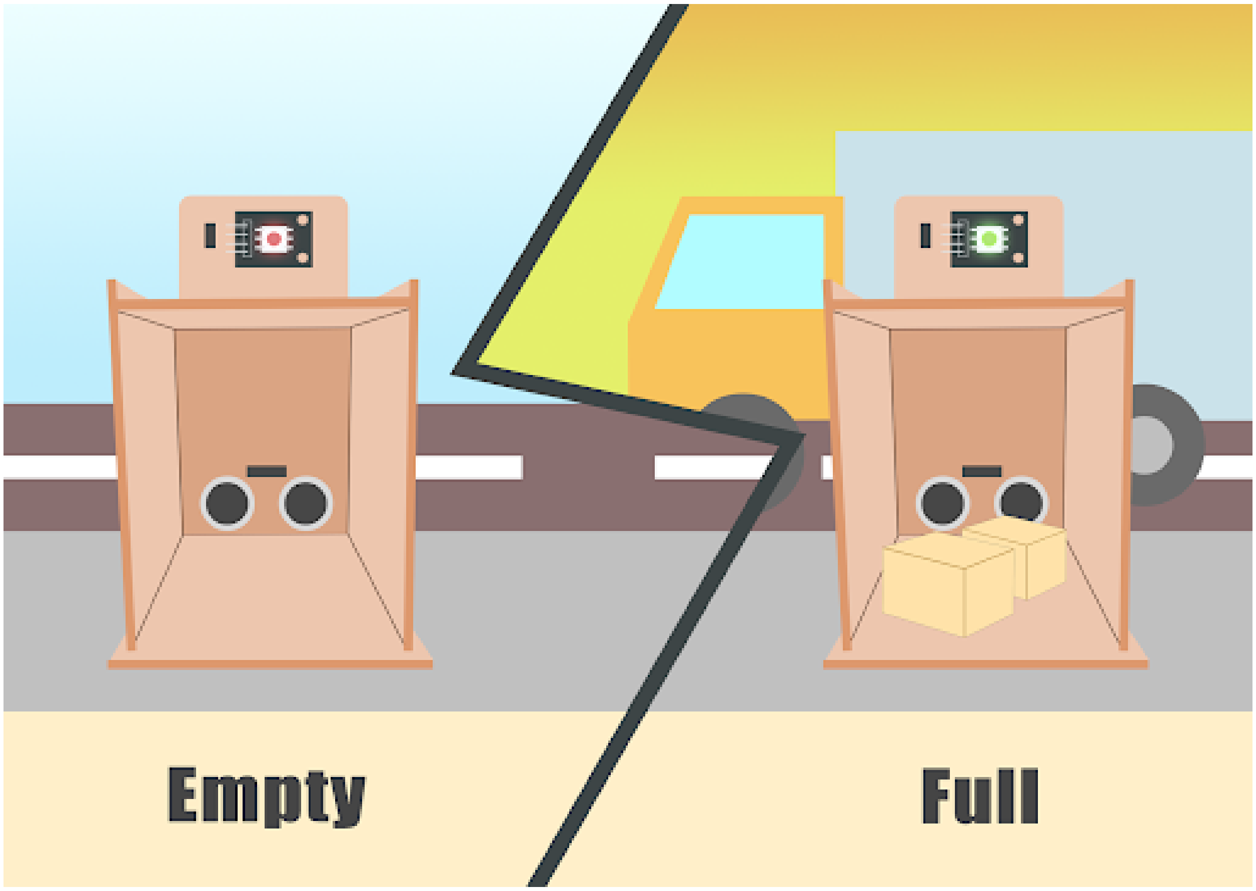
- Snap If statement into the loop
- If speed 40, then snap play tone Middle C for 1 beat from music


forever
call calculate_Speed
if speed ≥ 0 then
clear OLED display
show string join "distance1 : " distance1
show string join "distance2 : " distance2
show string join "speed : " speed
if speed > 40 then
play tone Middle C for 1 beat

Result

Think

1.2.4 Unloading Alert System

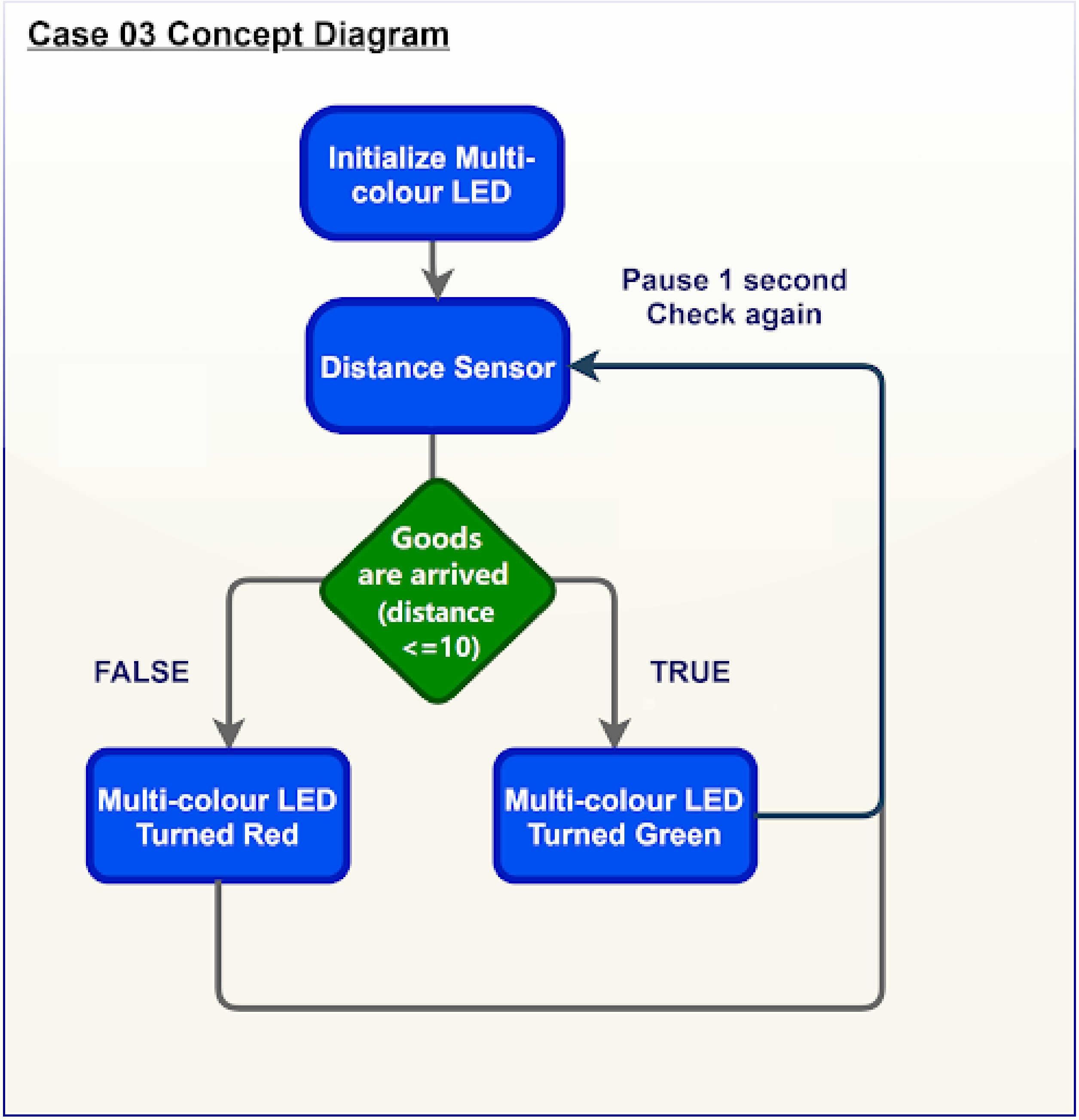


Goal

Background

What is an Unloading alert system?

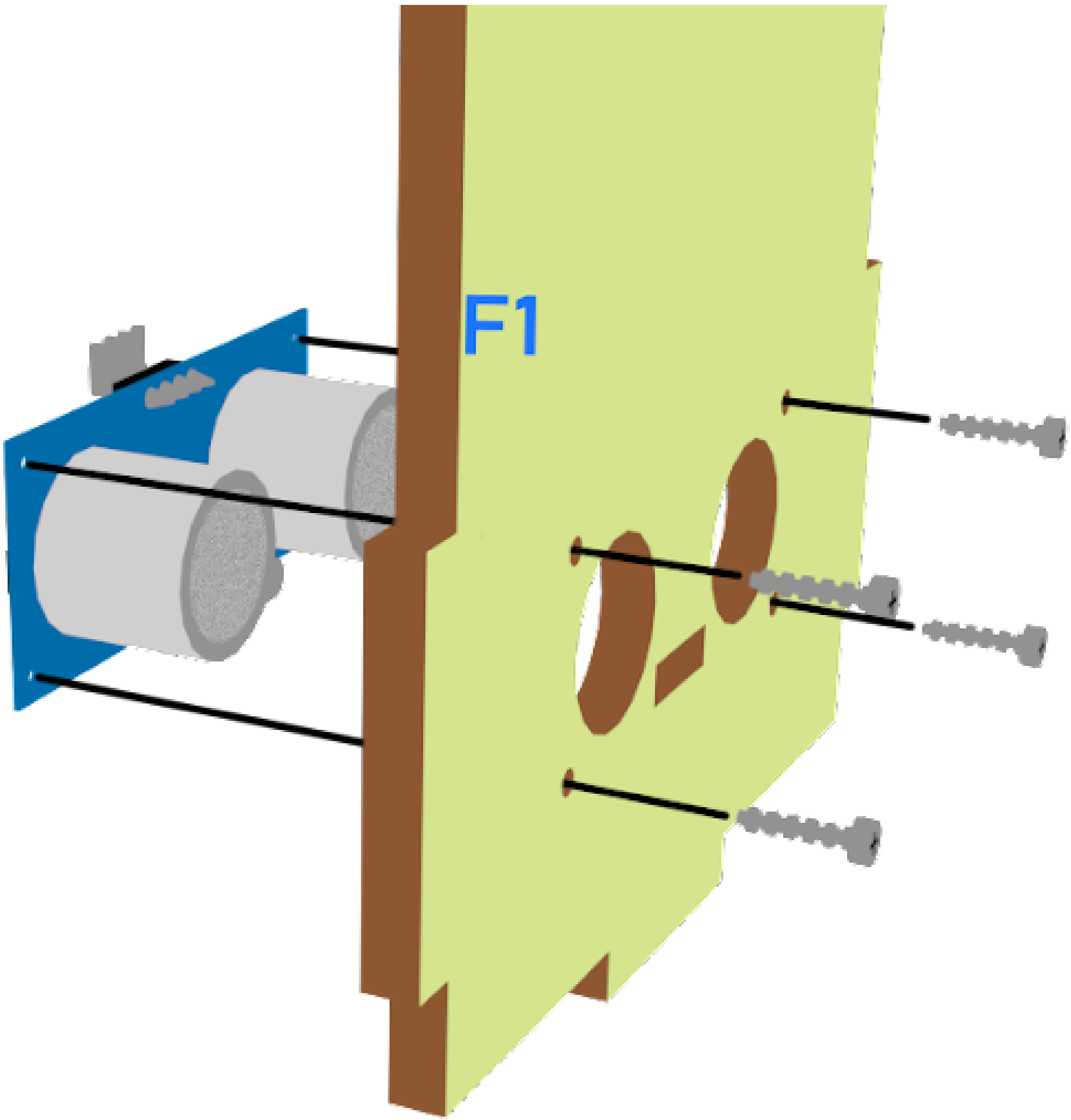
Unloading alert system operation



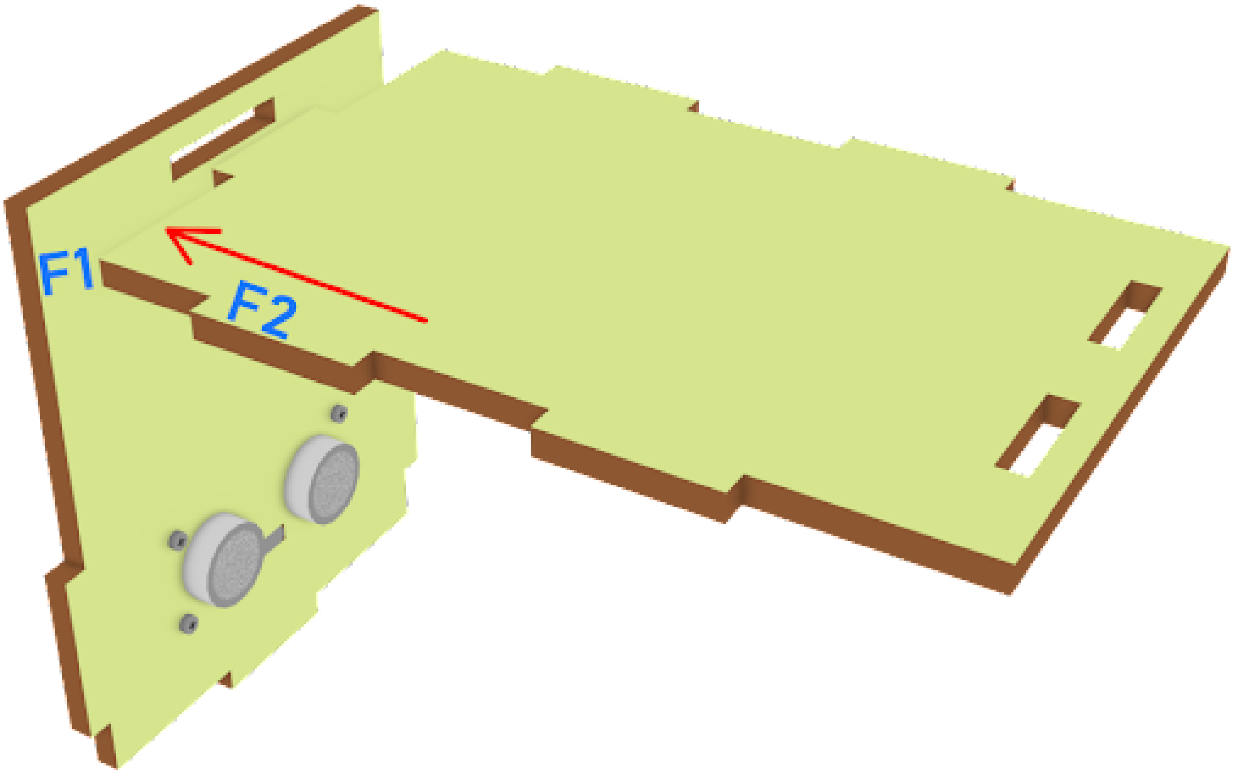
Part List

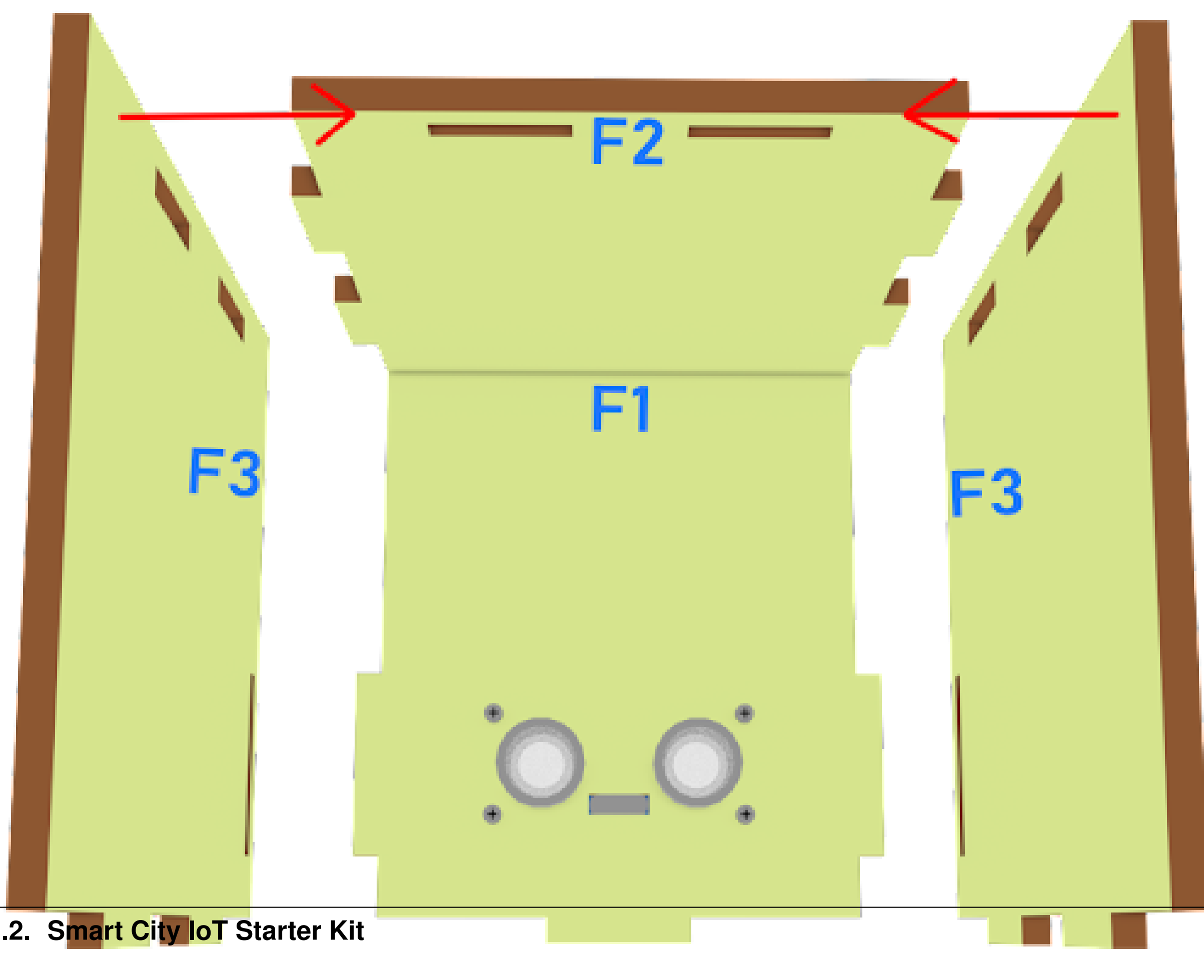
Assembly step

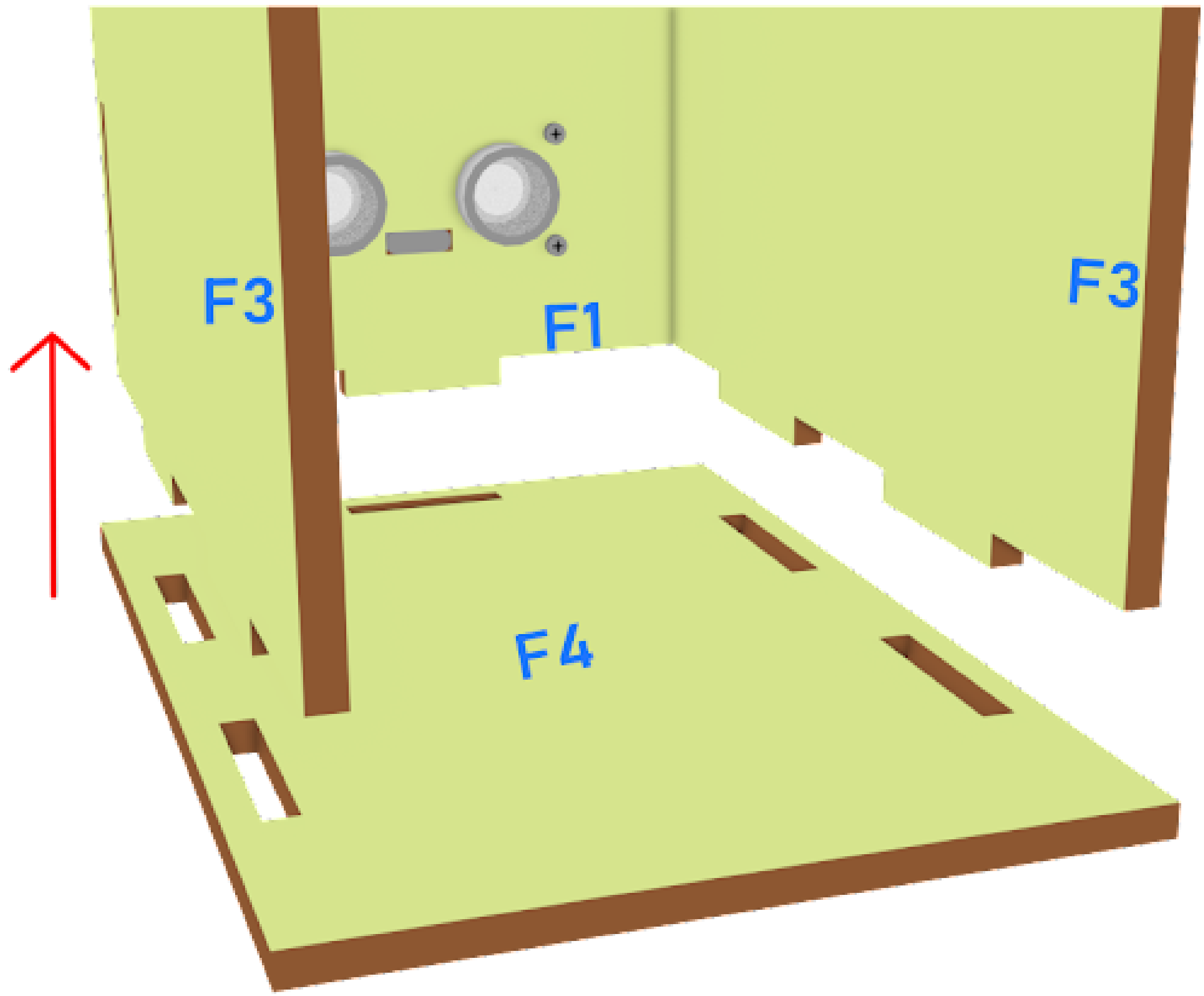
Step 1



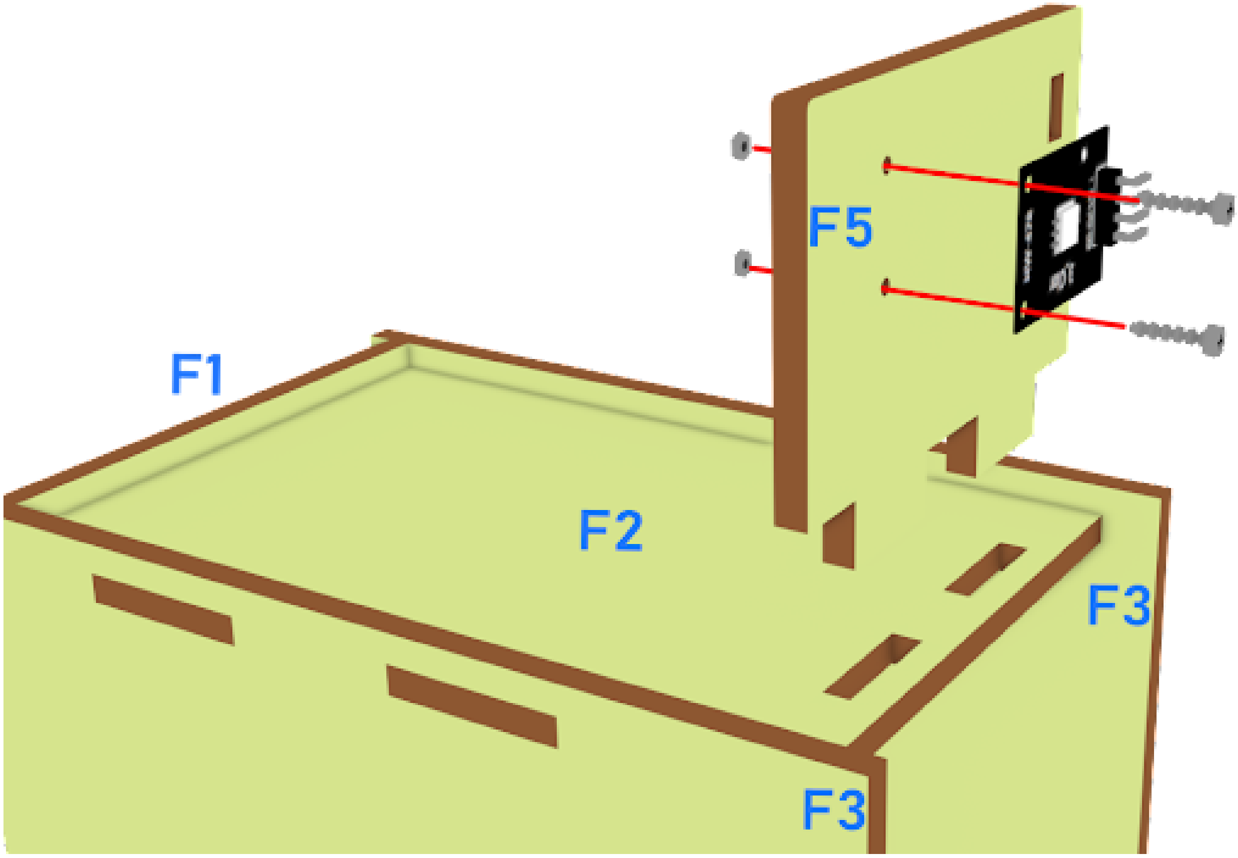
Step 2



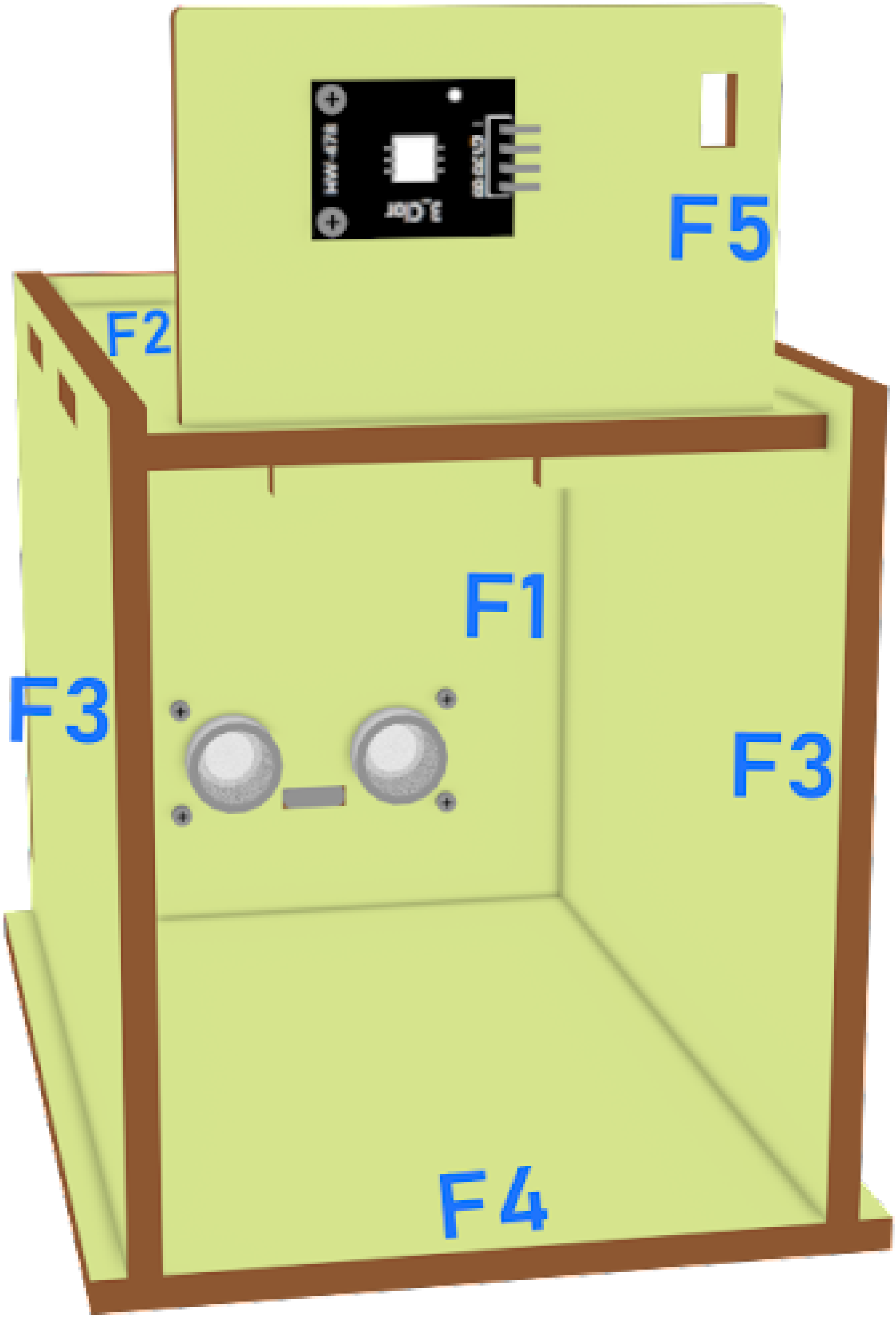



F3
F1
F3
F4

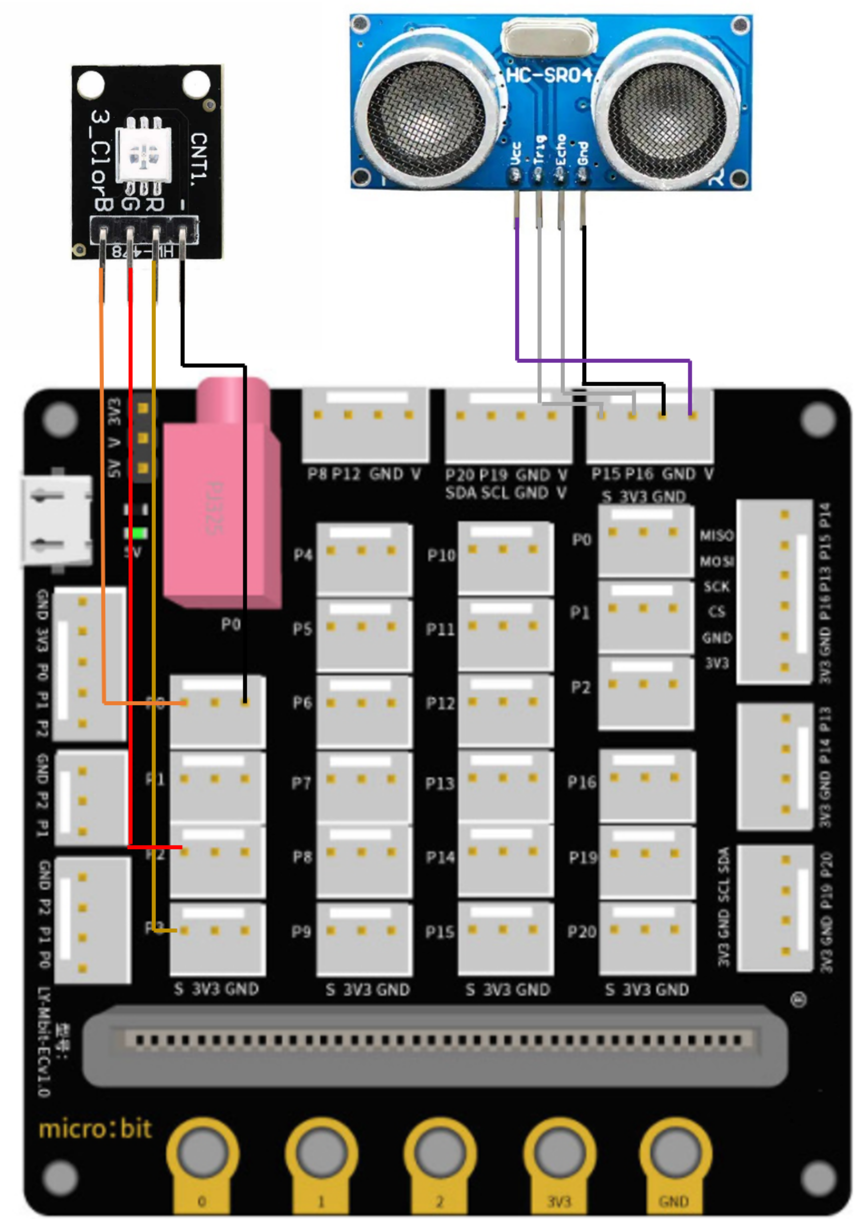
Step3



Step 4



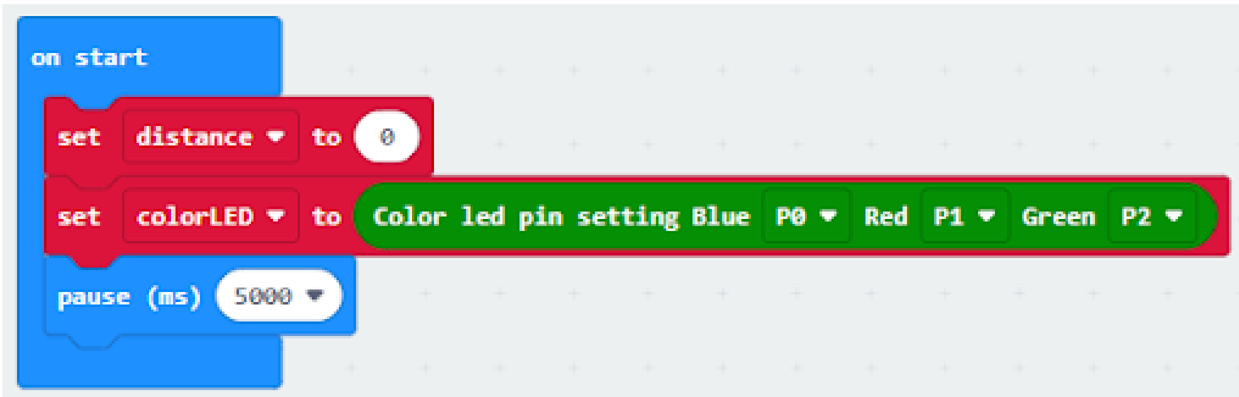
Hardware connect



Programming (MakeCode)

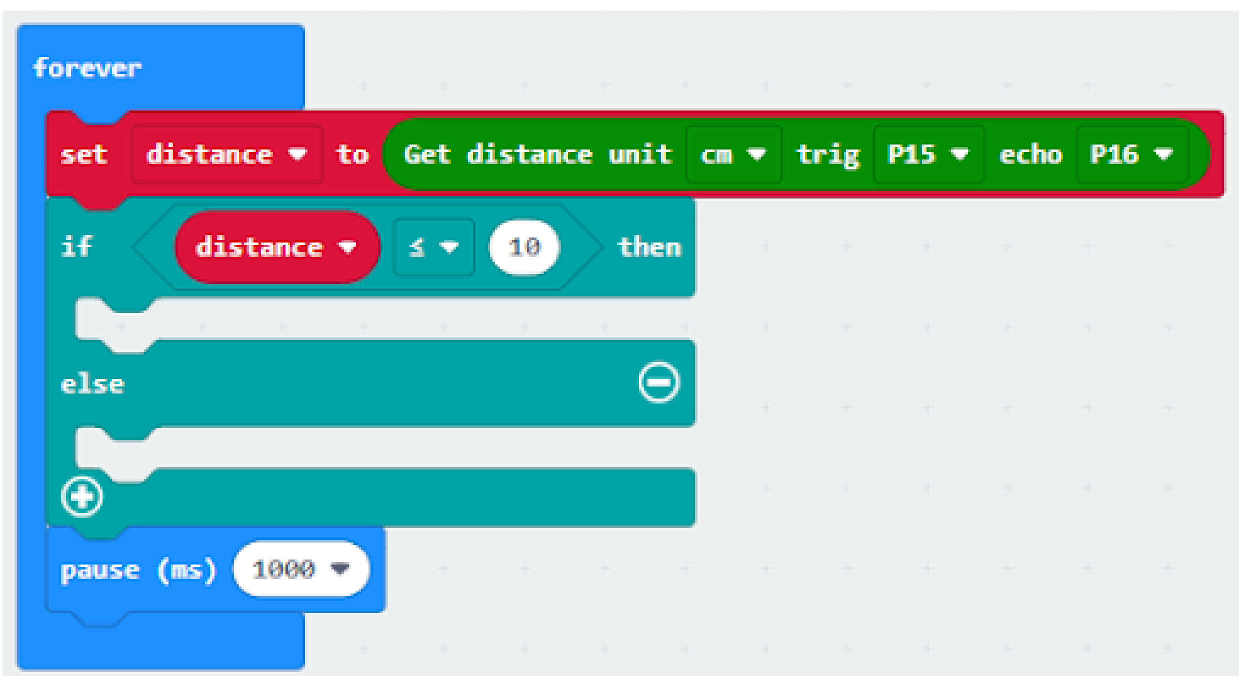
Step 1. Set variable and initialize multi-colour LED

- Inside on start, snap set variable distance to 0 from variables
- Snap set colorLED to color pin setting. . . .
- Snap pause to wait 5 seconds



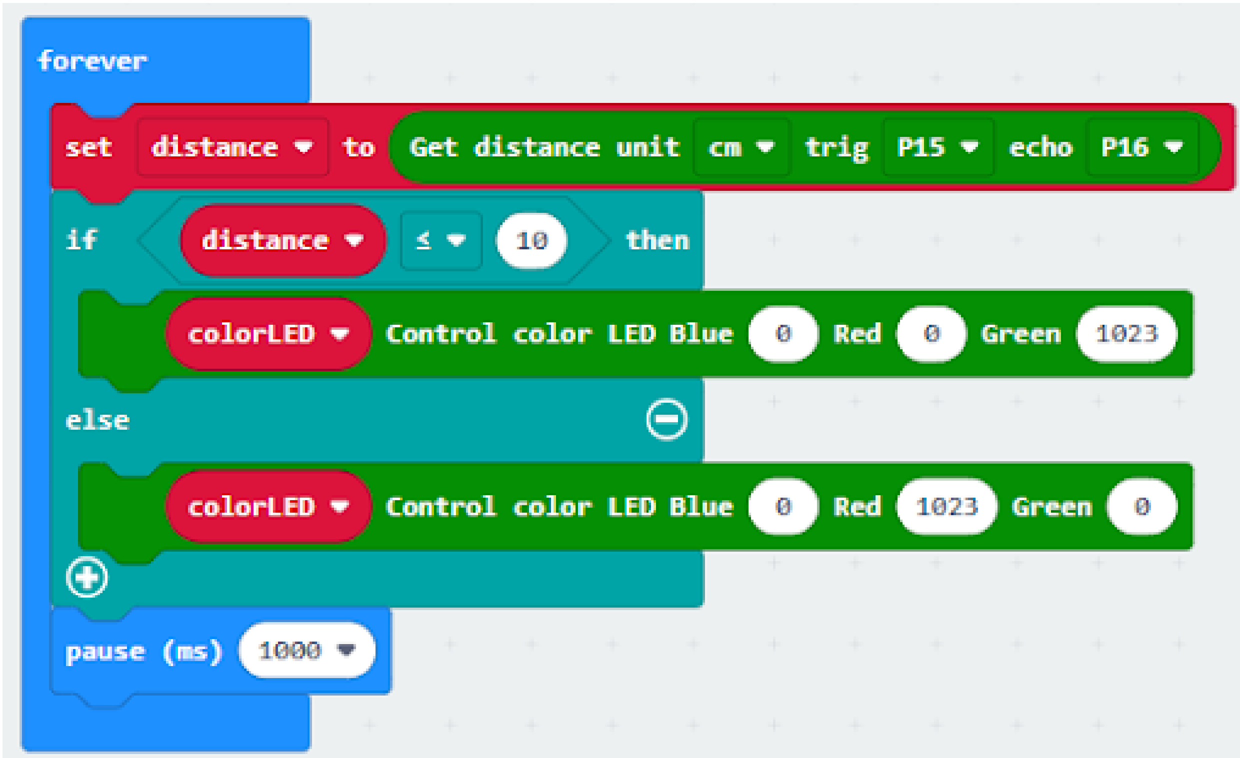
Step 2. Get distance value

- Inside block forever. Set distance to get distance unit cm trig P15 echo P16, that's say get the distance value by connecting the distance sensor to P15 and P16
- Snap if statement into forever, set distance 10 into if statement
- Snap Pause to the loop to wait 1 second for next checking



Step 3. Show indicating colours with distance value

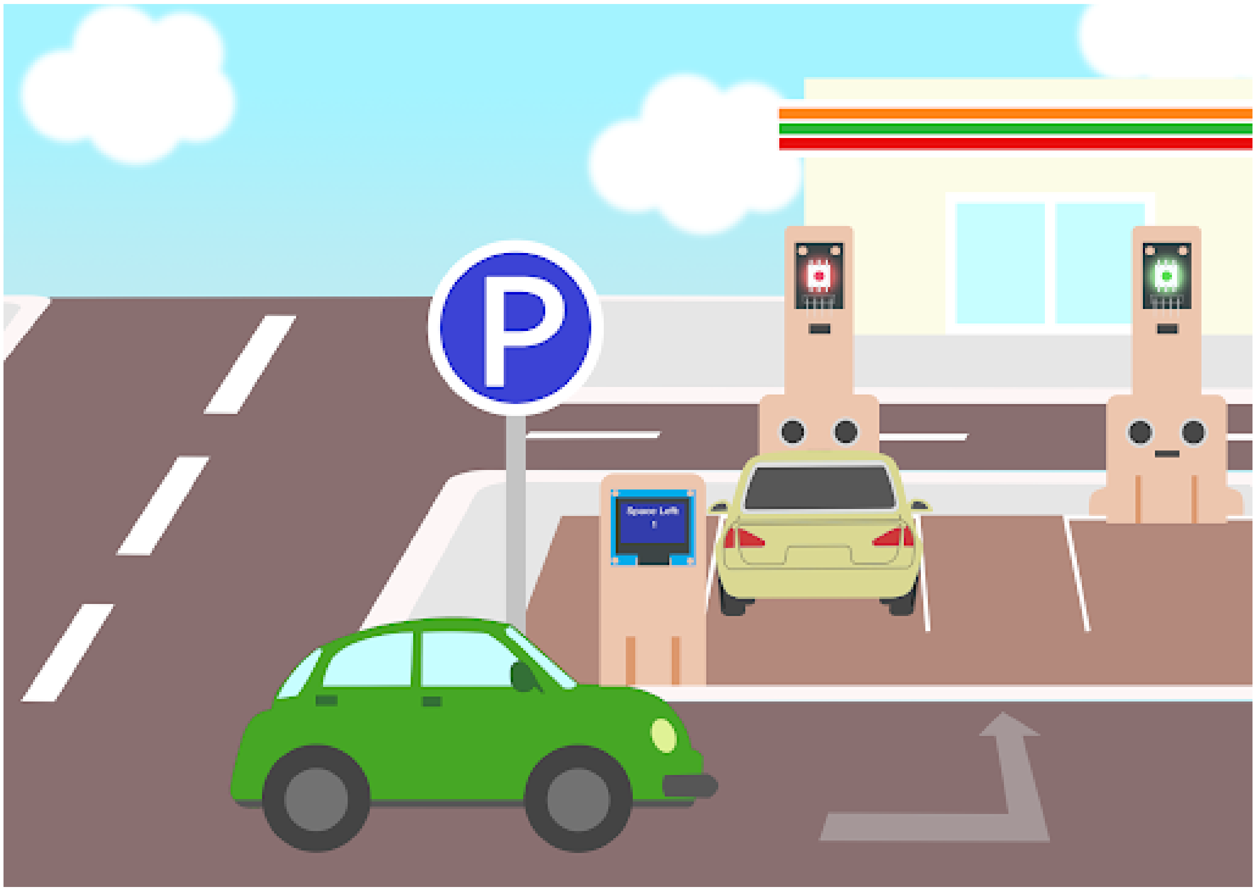
- If distance 10, then strip show color green, else strip show color red



Result

Think

1.2.5 Smart Car Park Access Barrier 1: Car Park Monitoring System

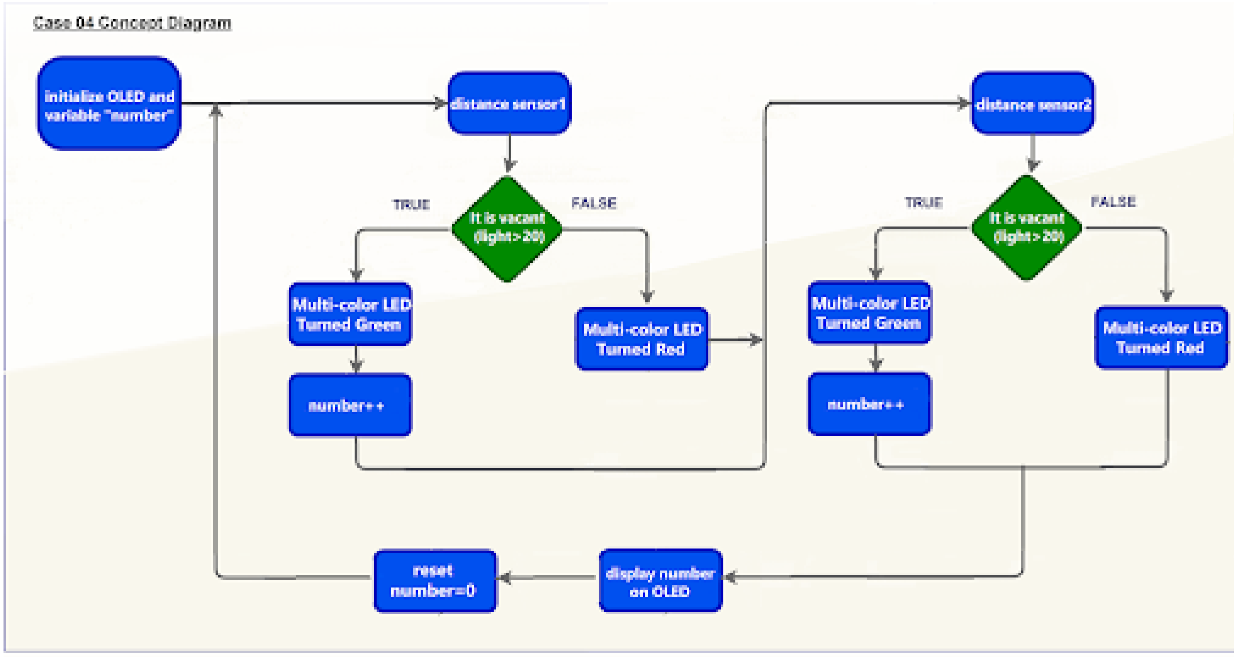


Goal

Background

What is an Smart car park monitoring systems?

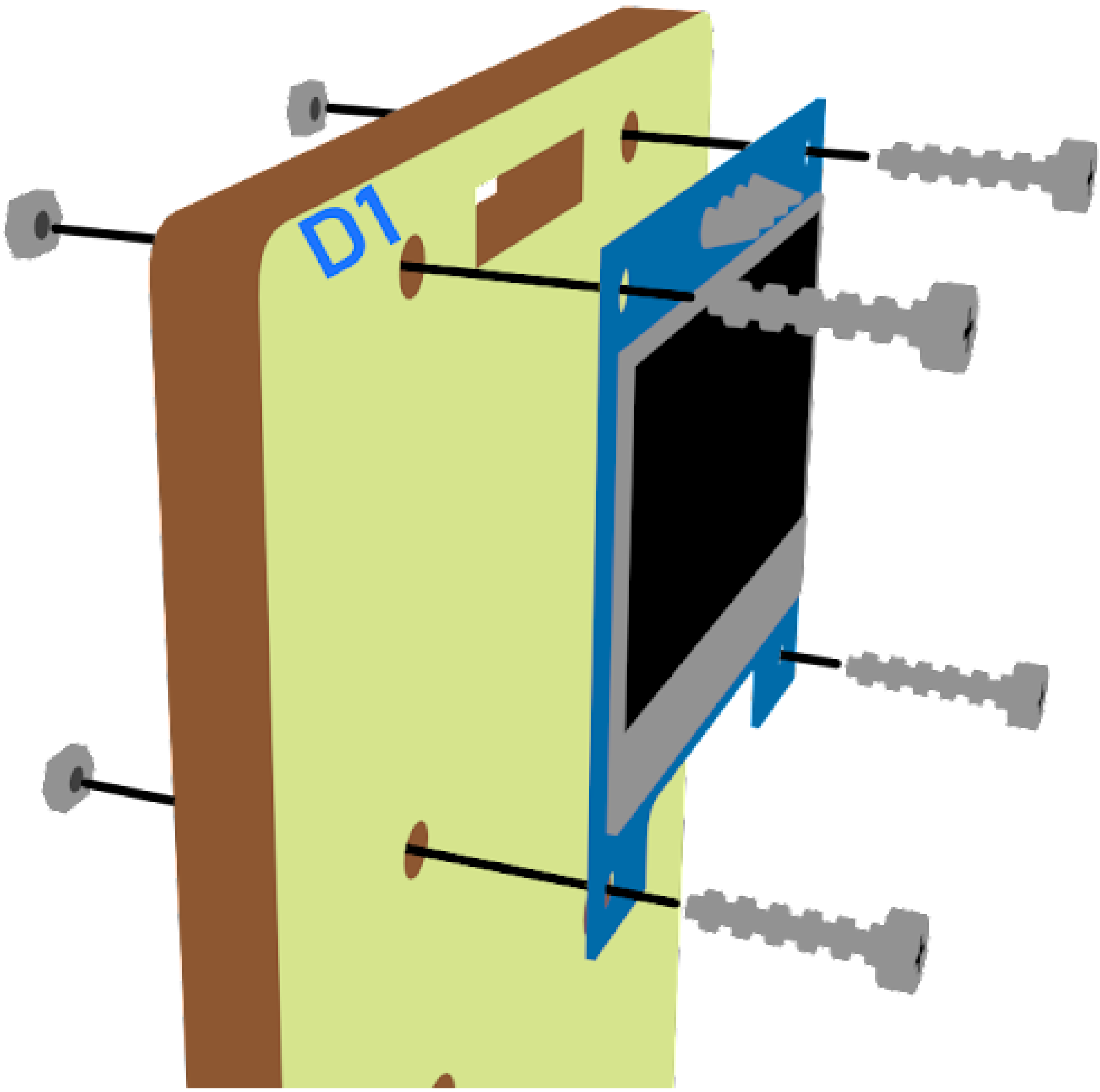
Smart car park monitoring systems operation



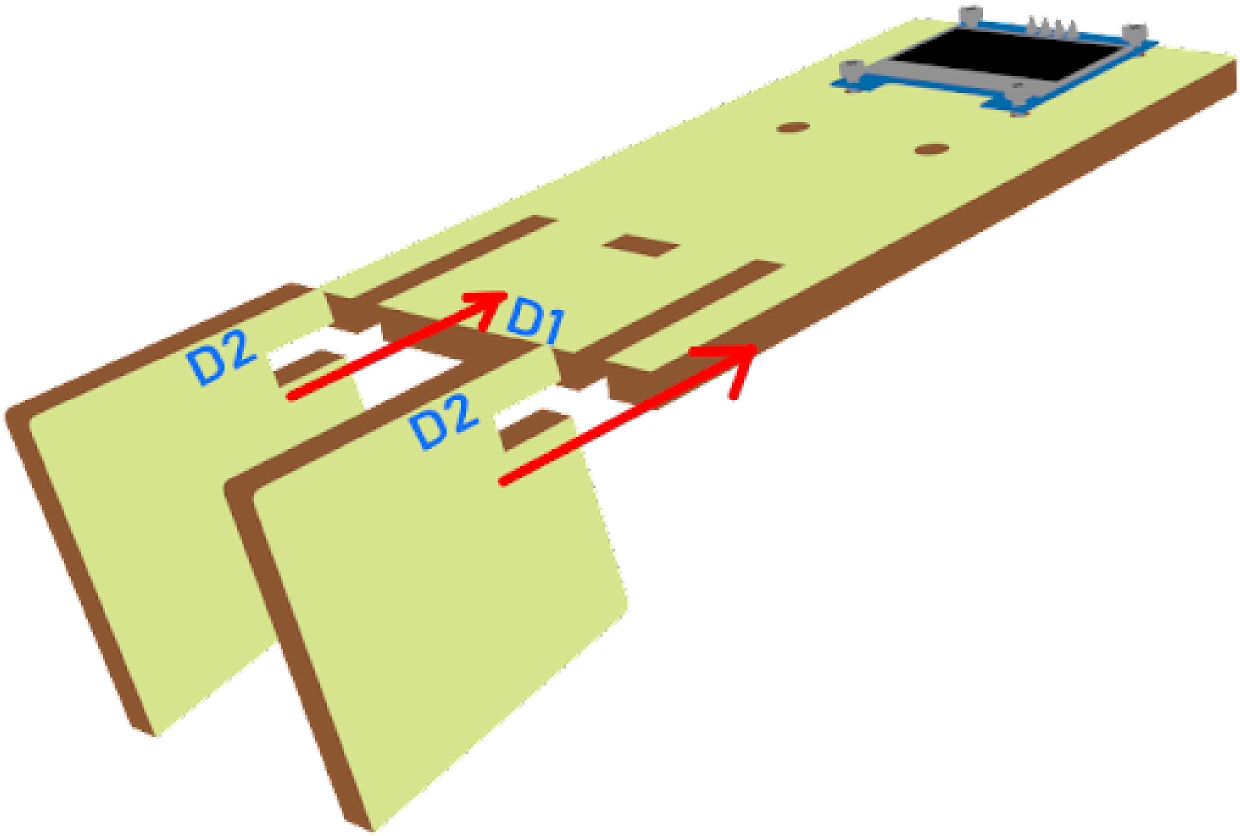
Part List

Assembly step

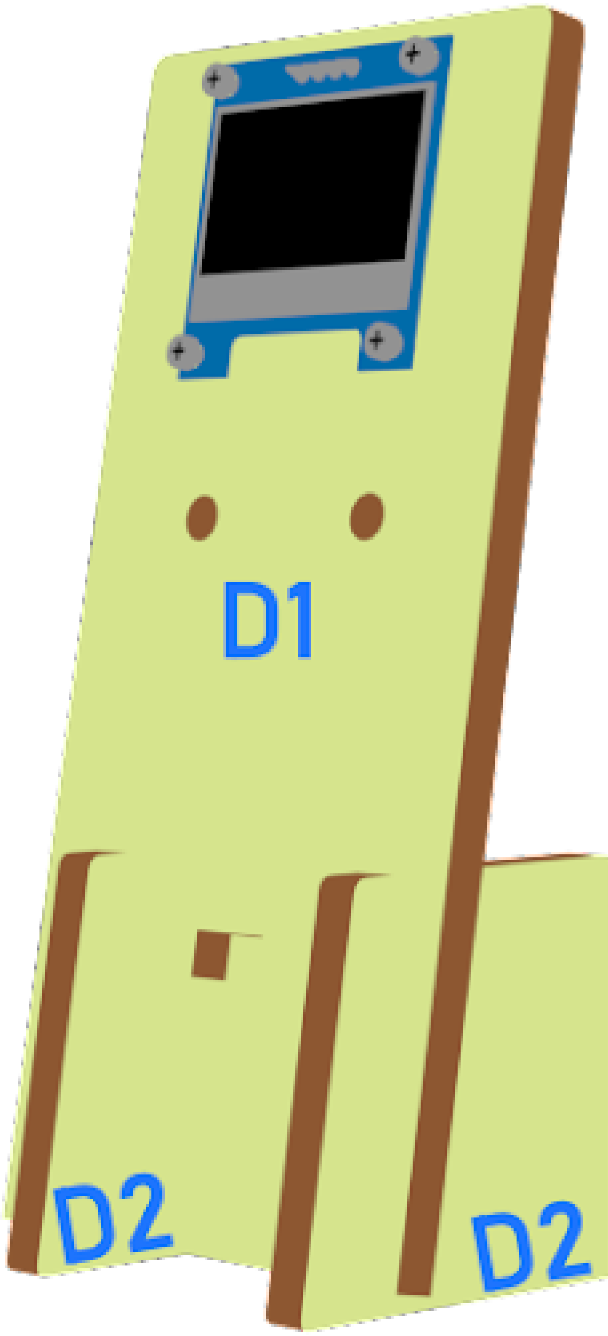
Step 1



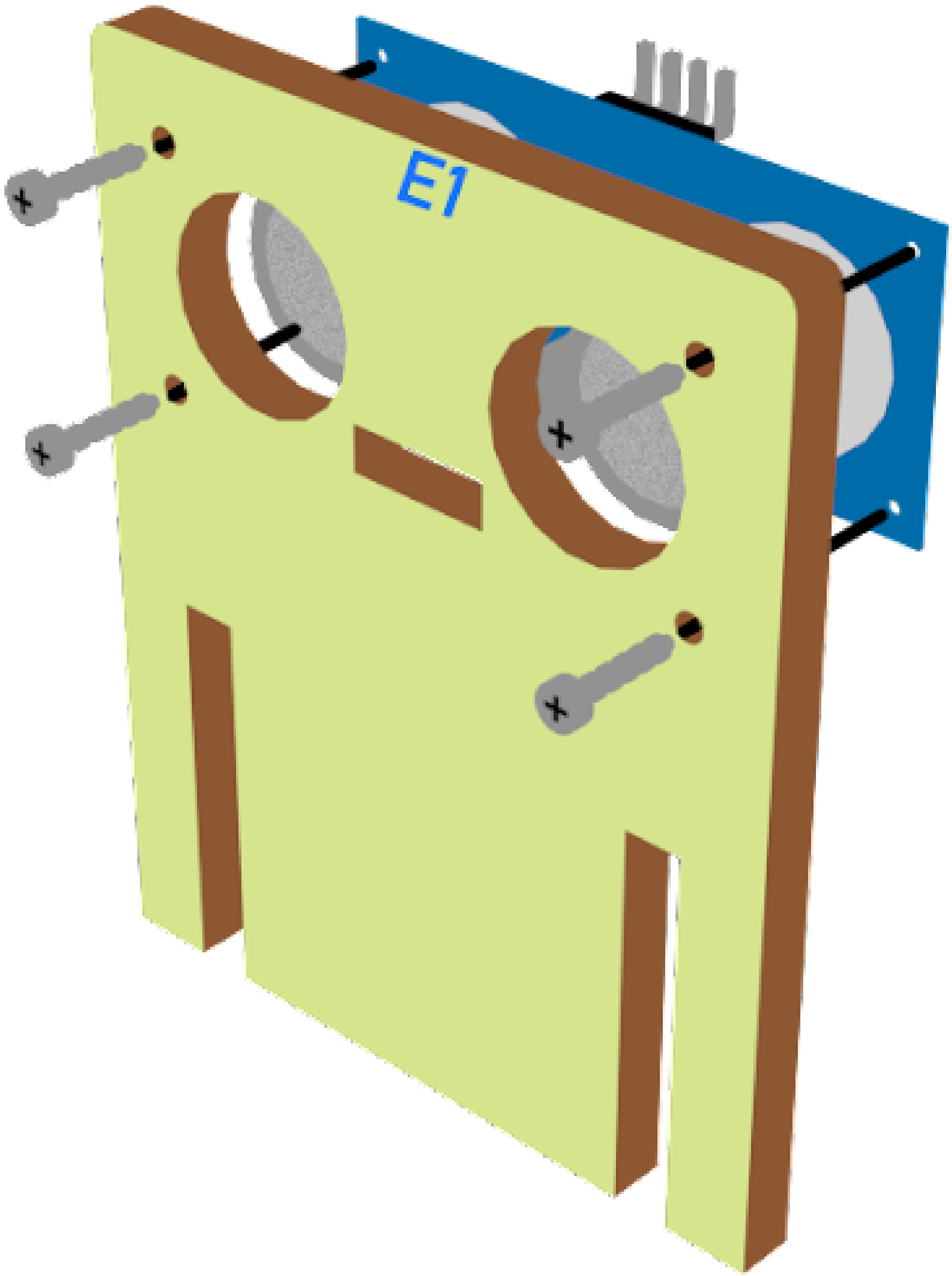
Step 2



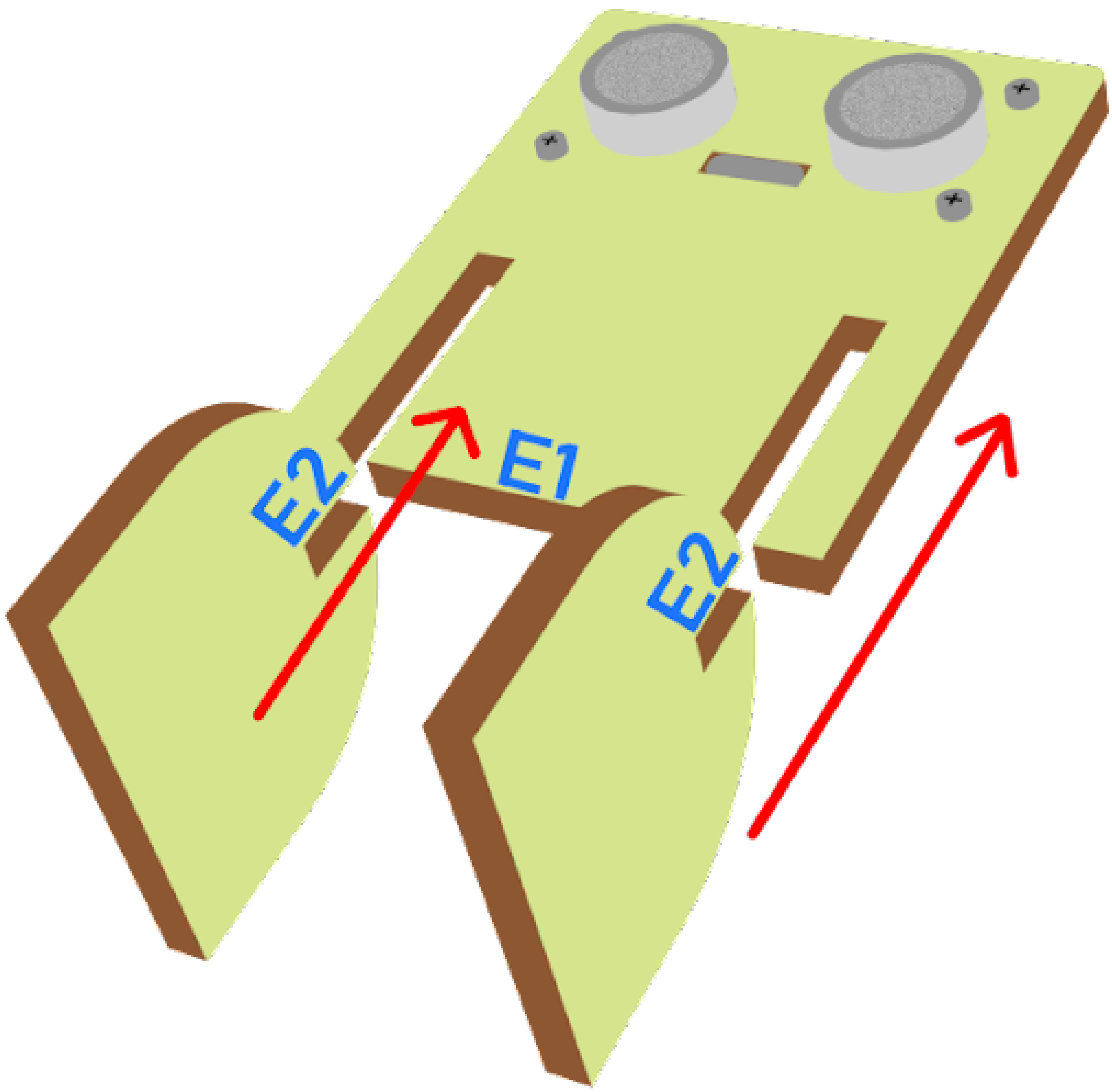
Step 3



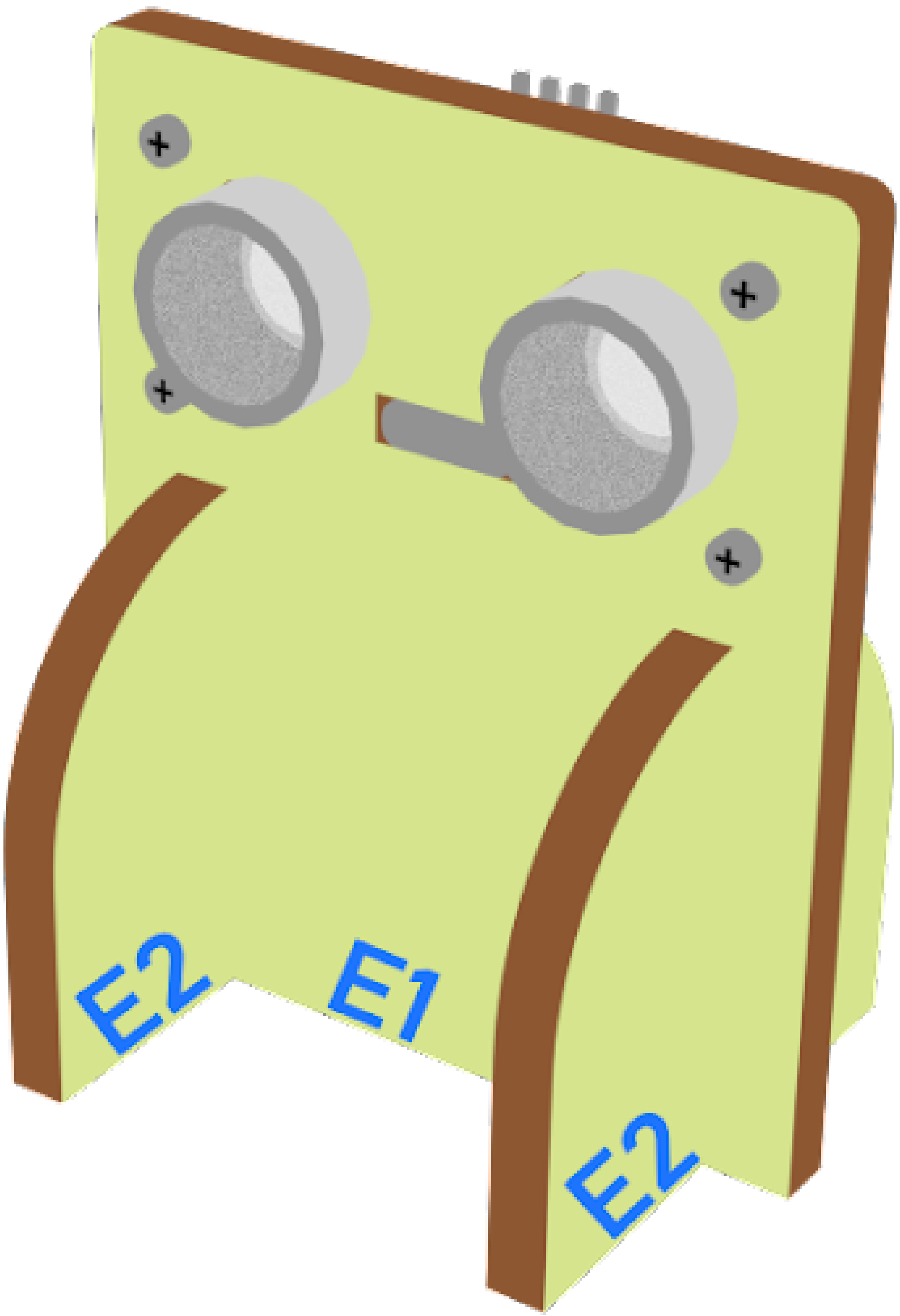
Step 4



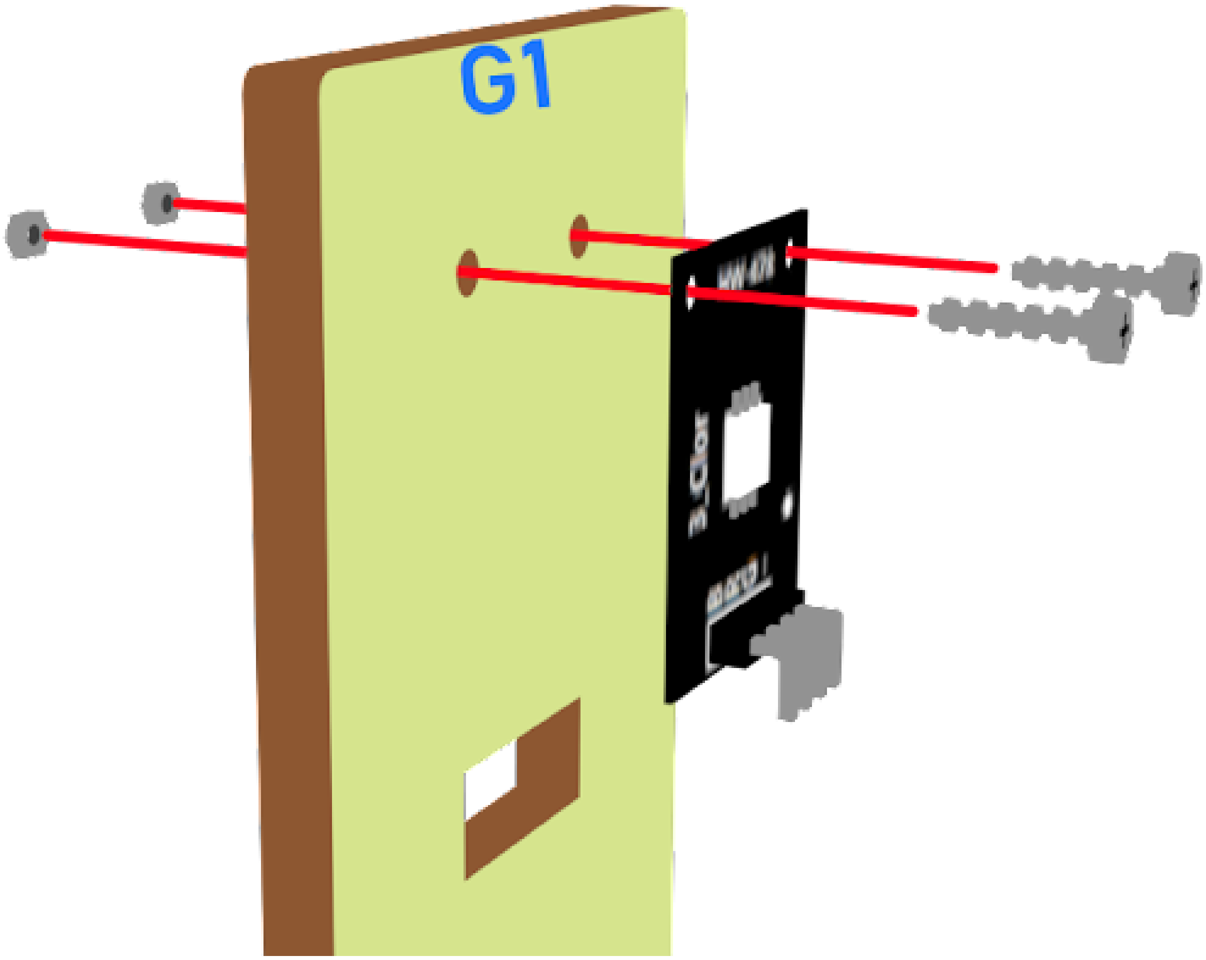
Step 5



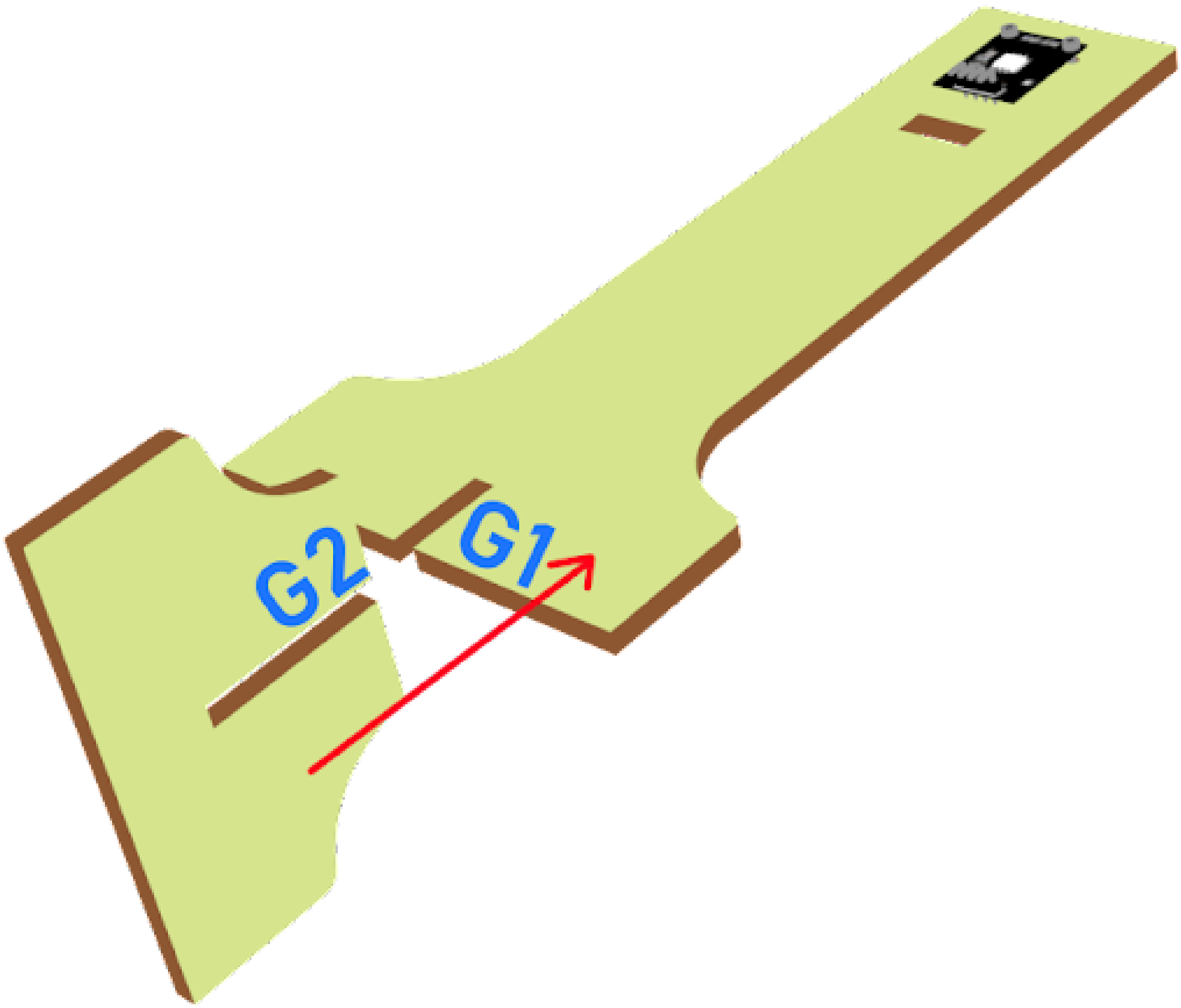
Step 6



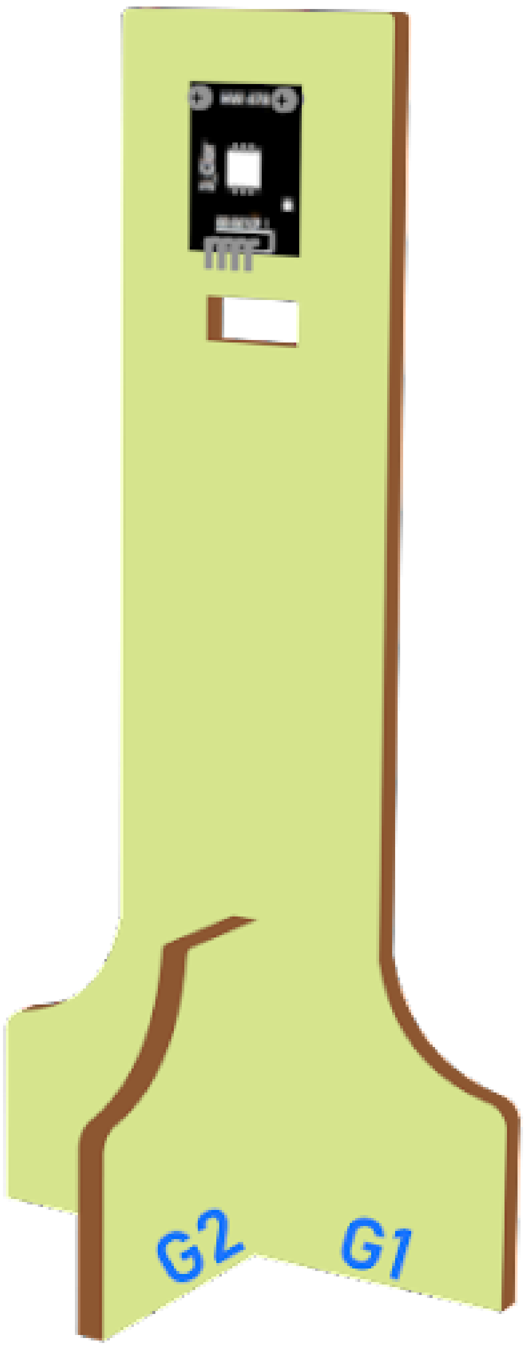
Step 7



Step 8



Step 9

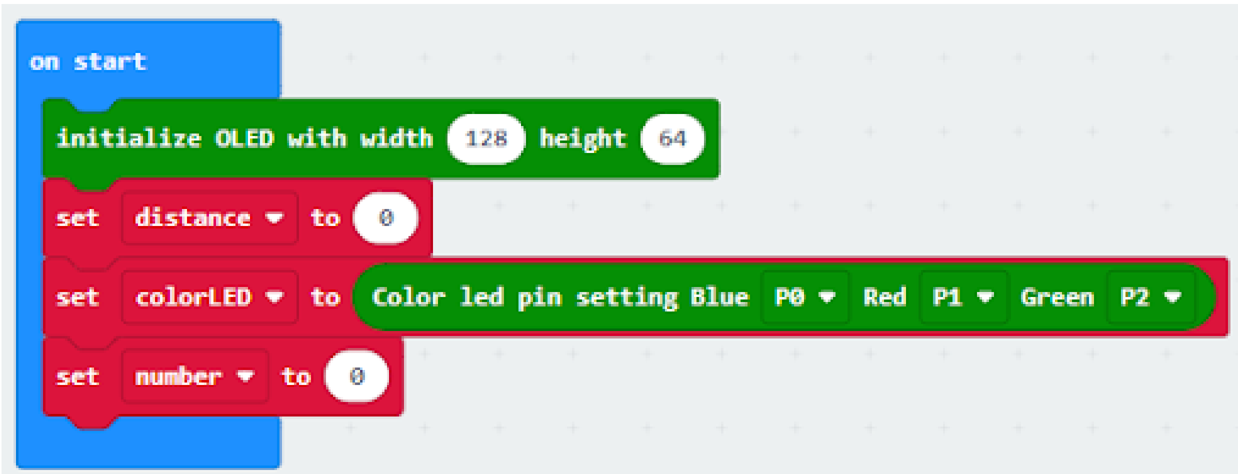


Hardware connect

Programming (MakeCode)

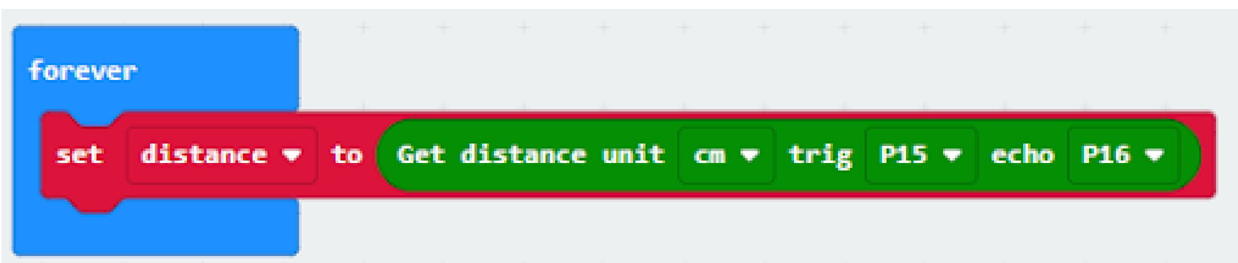
Step 1. Set variables and initialize multi-colour LED and OLED screen

- Drag Initialize OLED with width:128, height: 64 to on start
- Inside on start, snap set variable distance to 0 and set number to 0 from variables.
- Snap set colorLED to color led pin setting … set colorLED to color led pin setting …



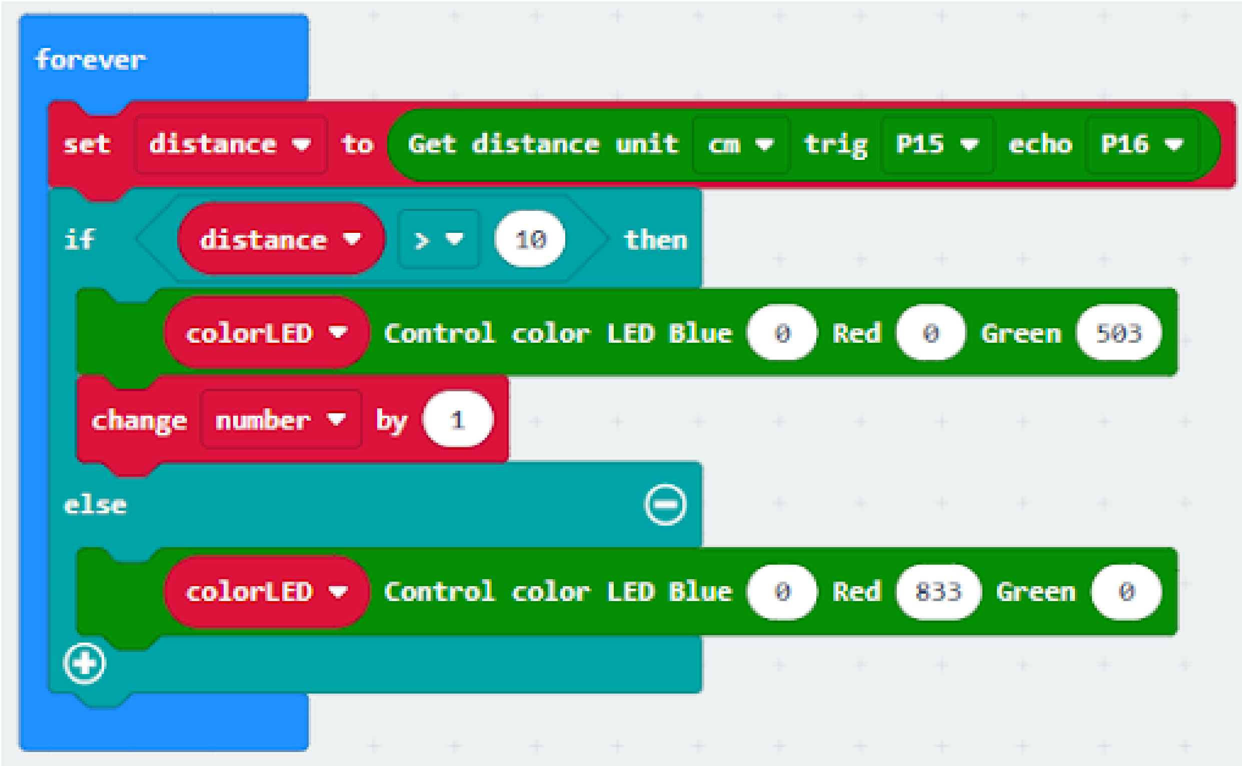
Step 2. Get distance

- Drag get distance to distance unit cm trig P15 echo P16, store the value to variable distance.



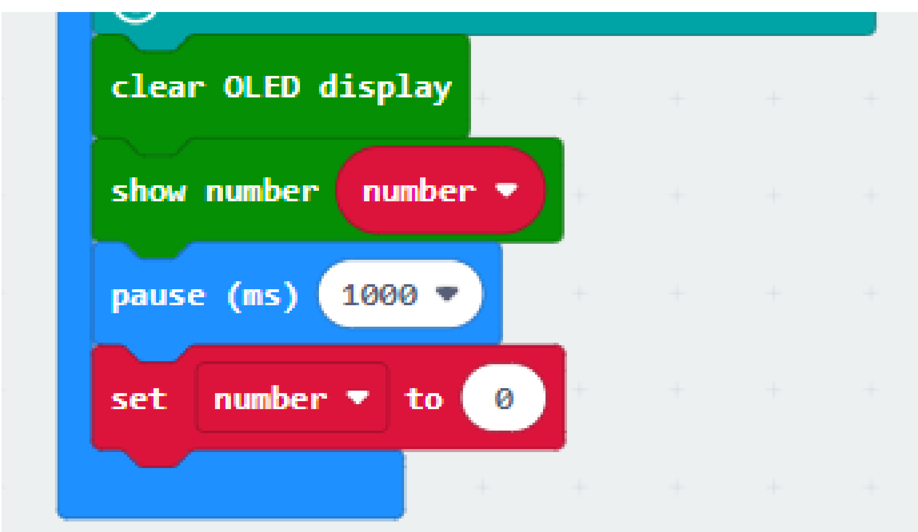
Step 3. Show indicating colours and count the number of vacancies

- Snap if statement into forever, set variable distance > 10
- If distance >10, then colorLED shows color green, else colorLED shows color red
- Snap change number by 1 if distance>10



Step 4 display on OLED

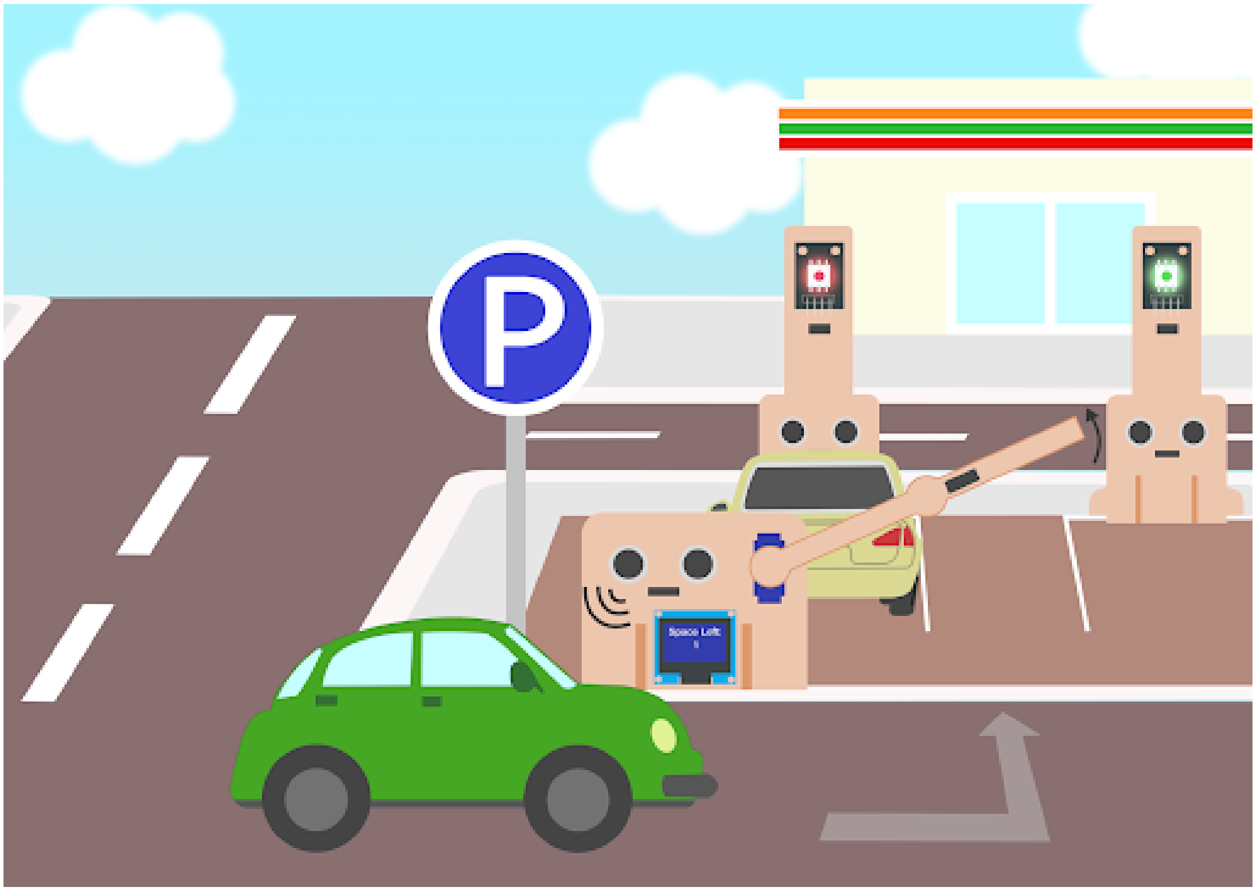
- Snap clear OLED display from OLED to avoid overlap
- Snap show number and show value of variables number
- Snap Pause to the loop to wait 1 second for next checking
- Reset number to 0 before next checking



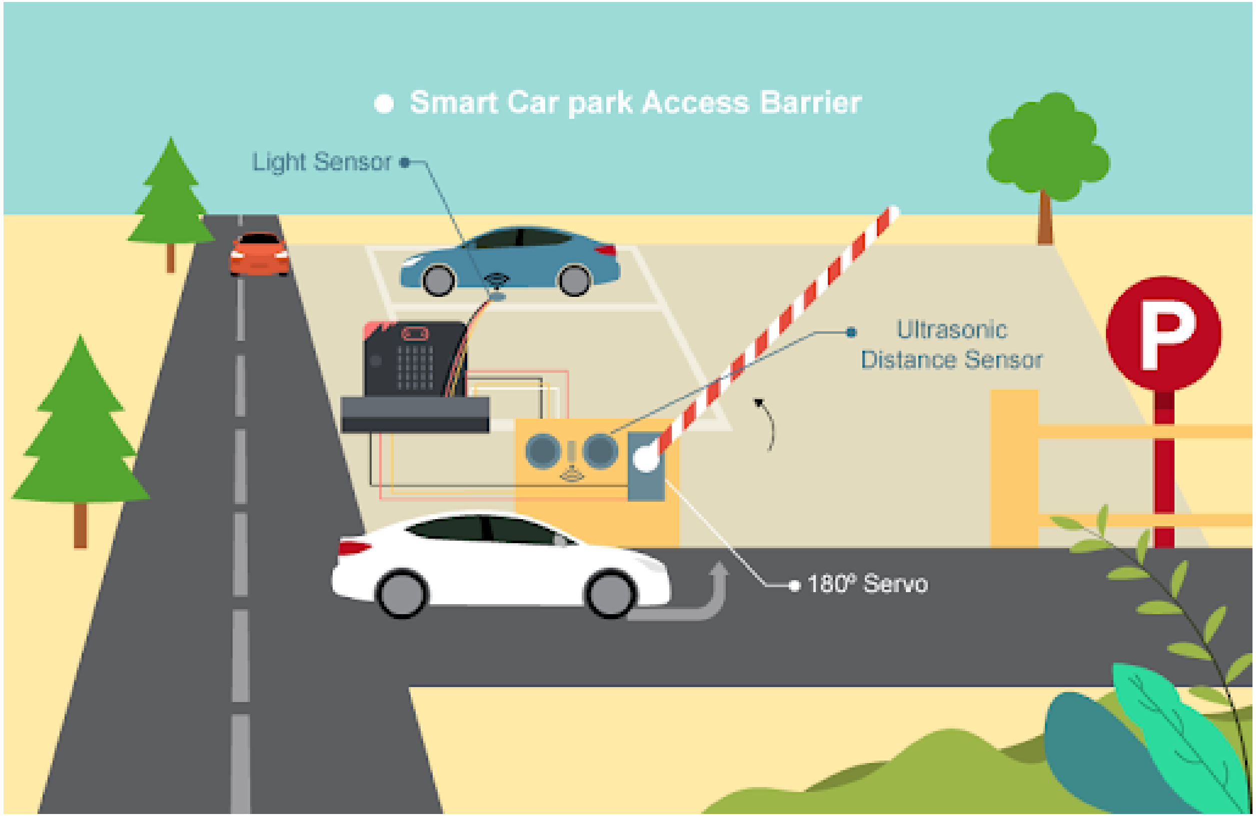
Result

Think

1.2.6 Smart Car Park Access Barrier 2: Car Park Access Barrier



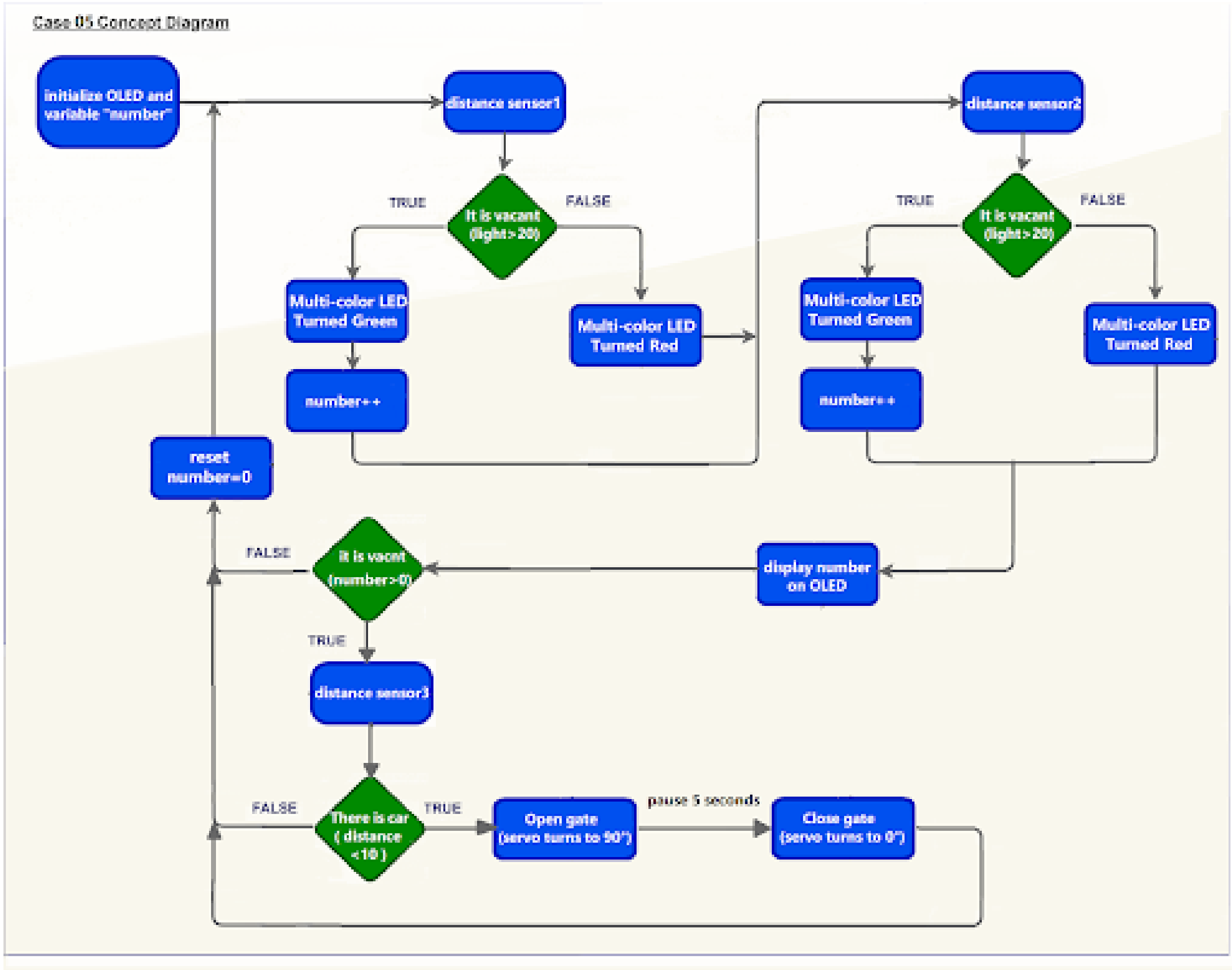
Goal



Background

What is a smart car park access barrier?

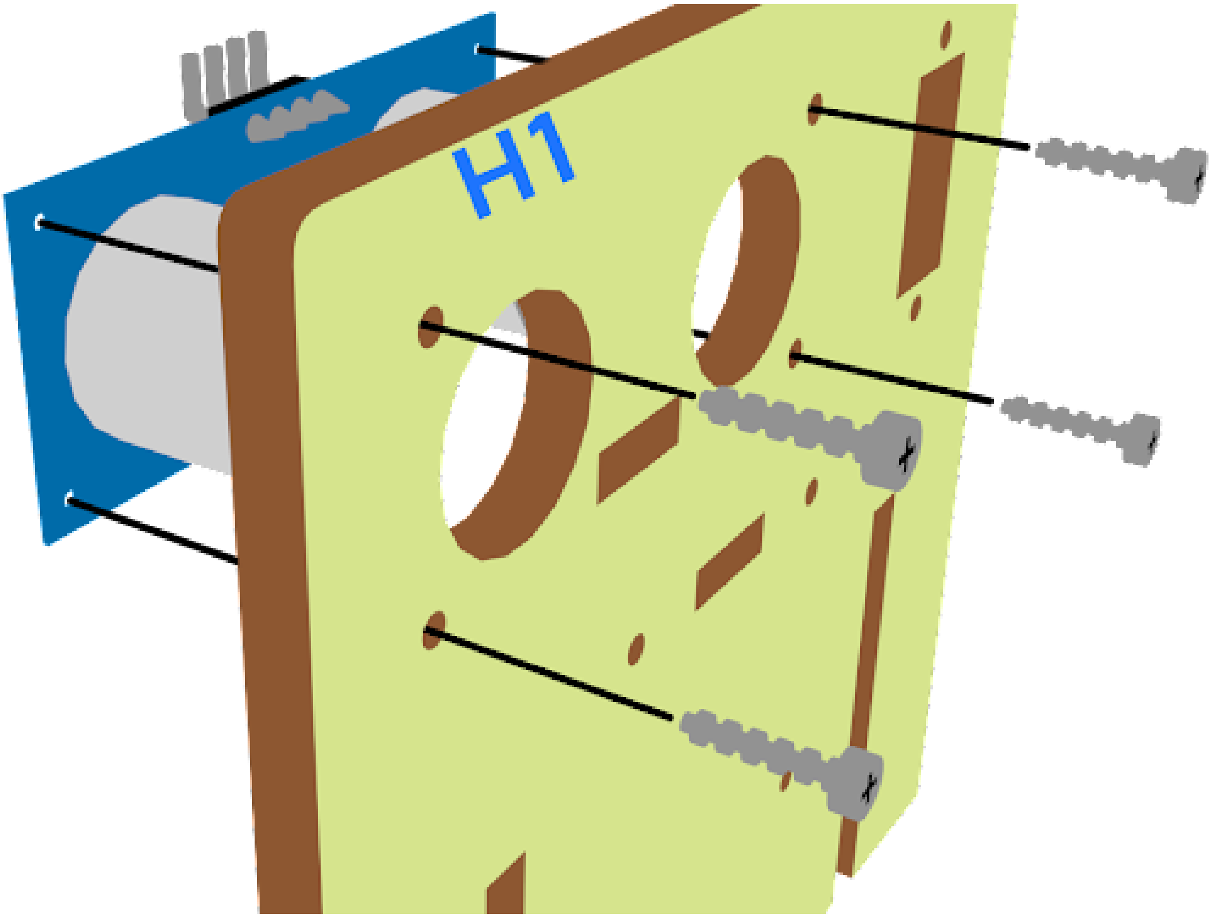
Car park access barrier operation



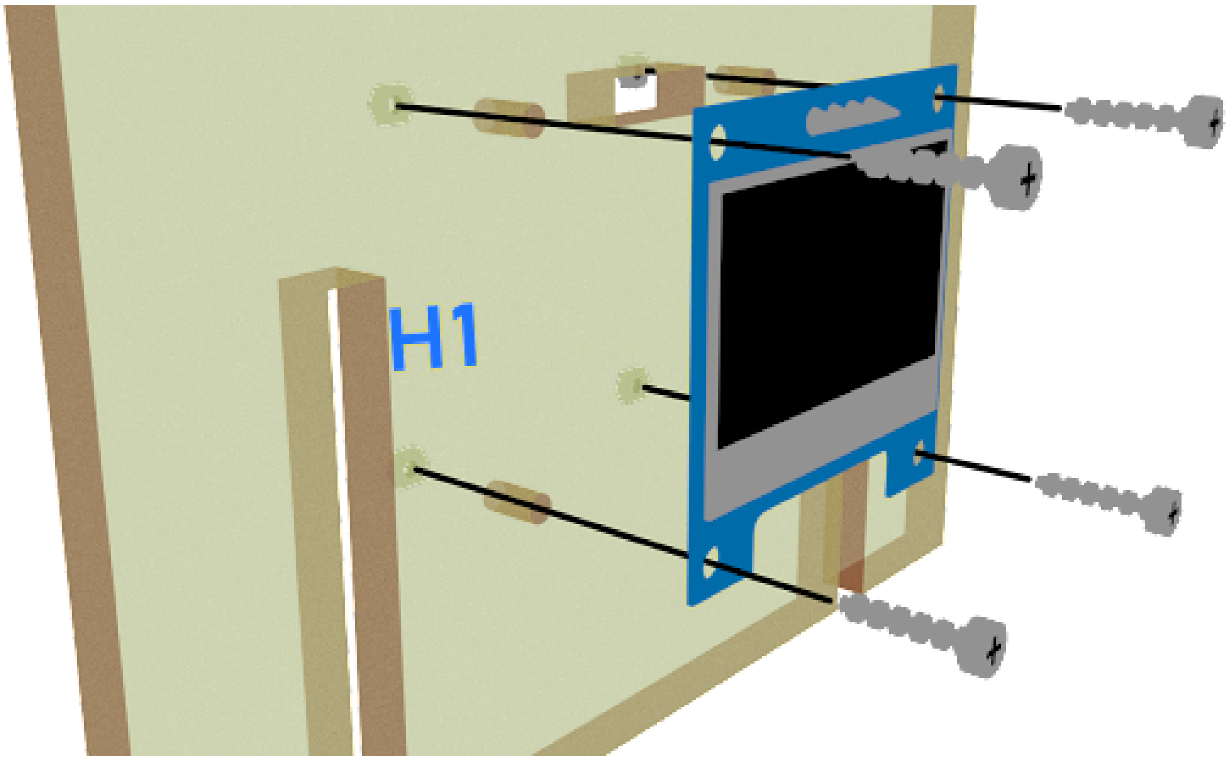
Part List

Assembly step

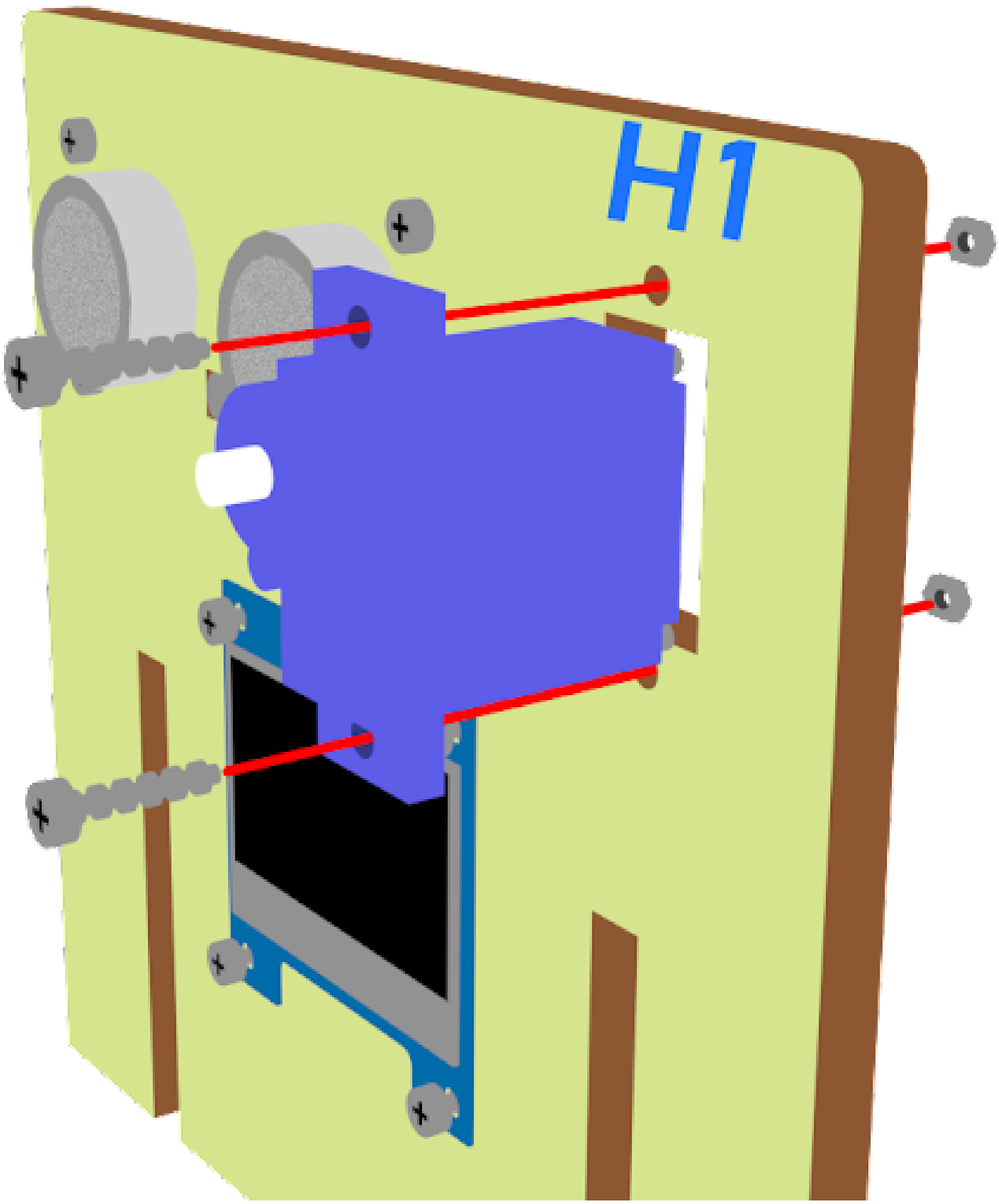
Step 1



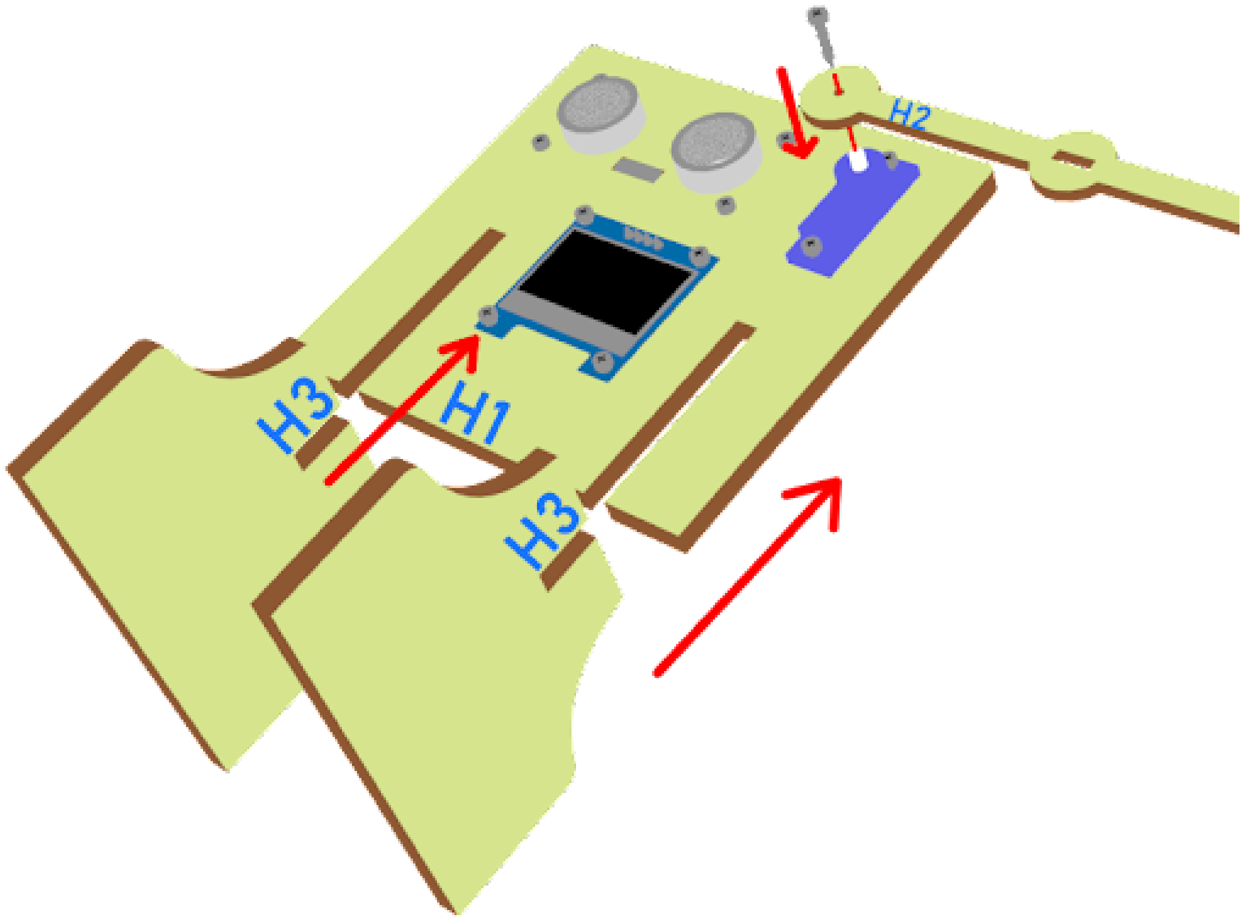
Step 2



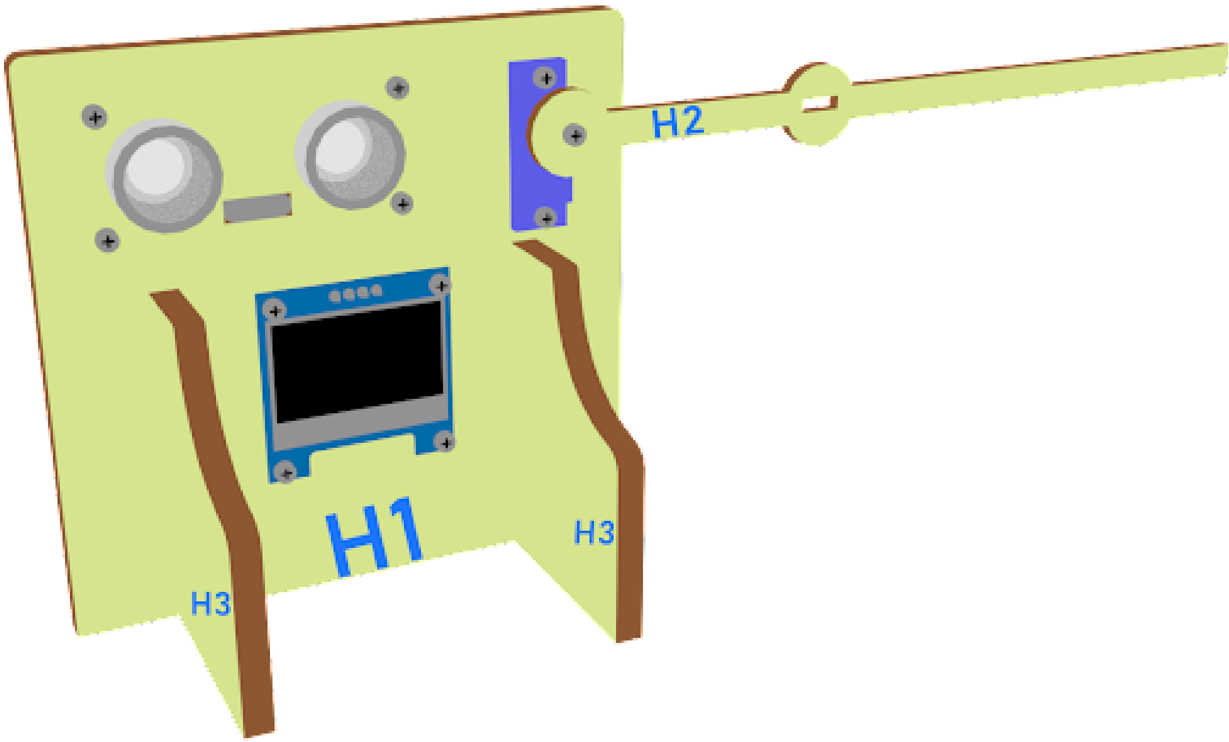
Step 3



Step 4



Step 5

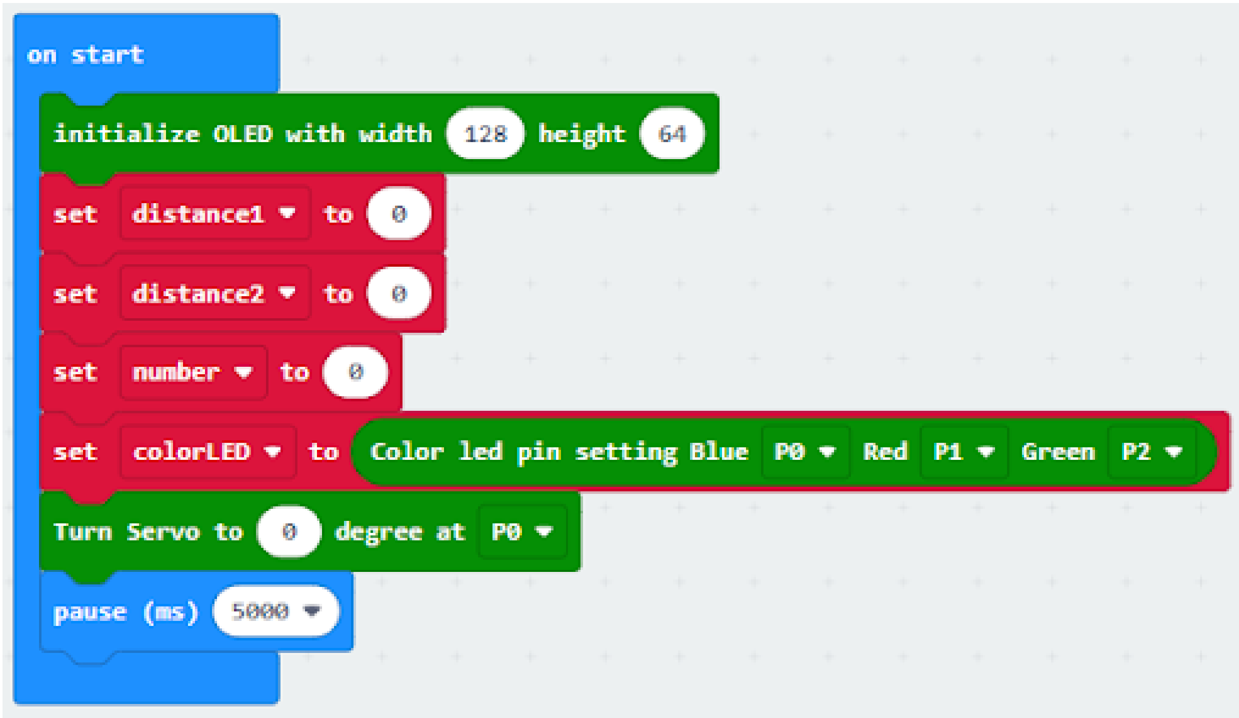


Hardware connect

Programming (MakeCode)

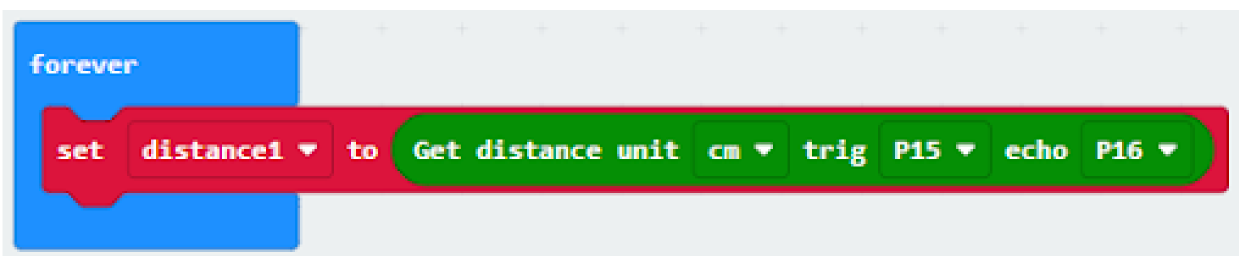
Step 1. Set variables, initialize OLED screen and servo at start position

- Drag Initialize OLED with width:128, height: 64 to on start
- Inside on start, snap set variable distance1 to 0, set variable distance2 to 0 and set number to 0 from variables.
- Snap set colorLED to color led pin setting
- Snap Turn Servo to 0 degree at P0.
- Snap pause to wait 5 seconds



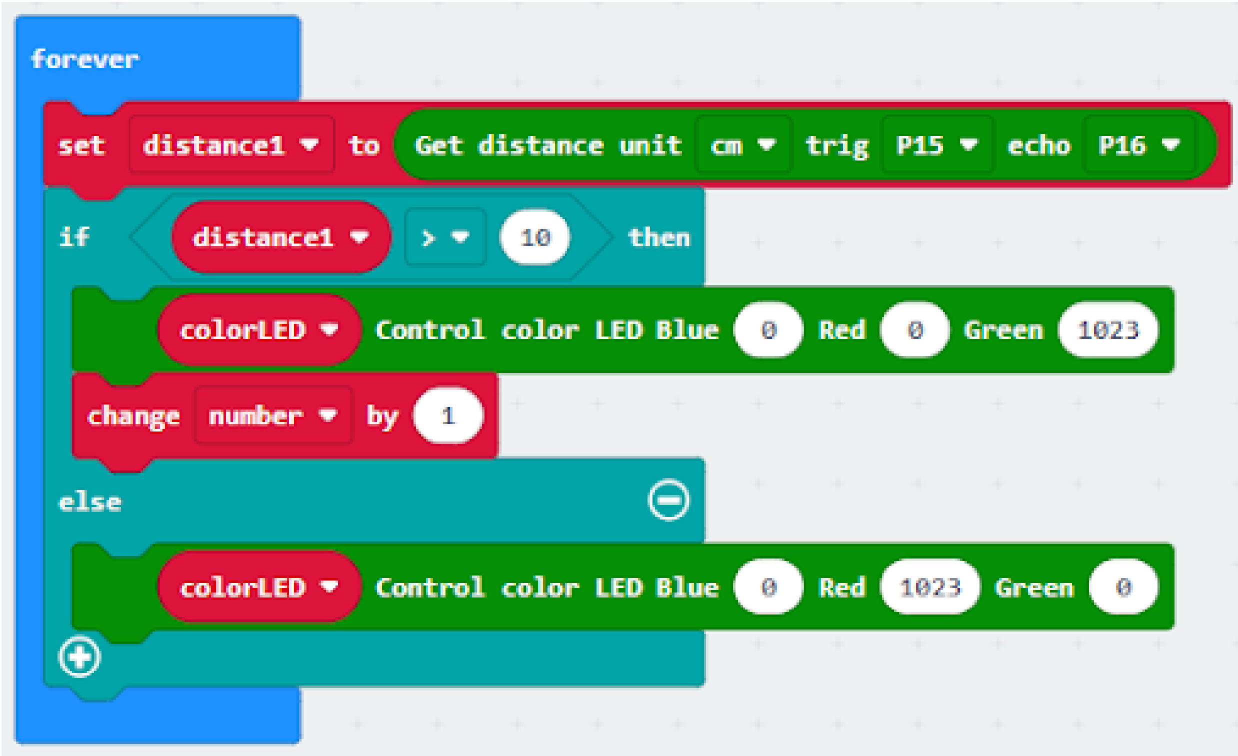
Step 2. Get distance

- Drag set distance1 to distance unit cm trig P15 echo P16, store the value to variable distance1.



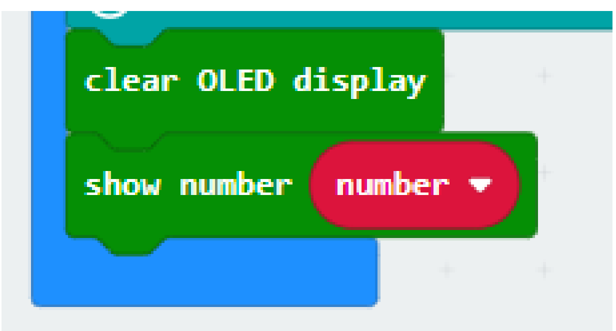
Step 3. Show indicating colours and count the number of vacancies

- Snap if statement into forever, set variable distance1 > 10
- If distance1 >10, then colorLED shows color green, else colorLED shows color red
- Snap change number by 1 if distance1>10



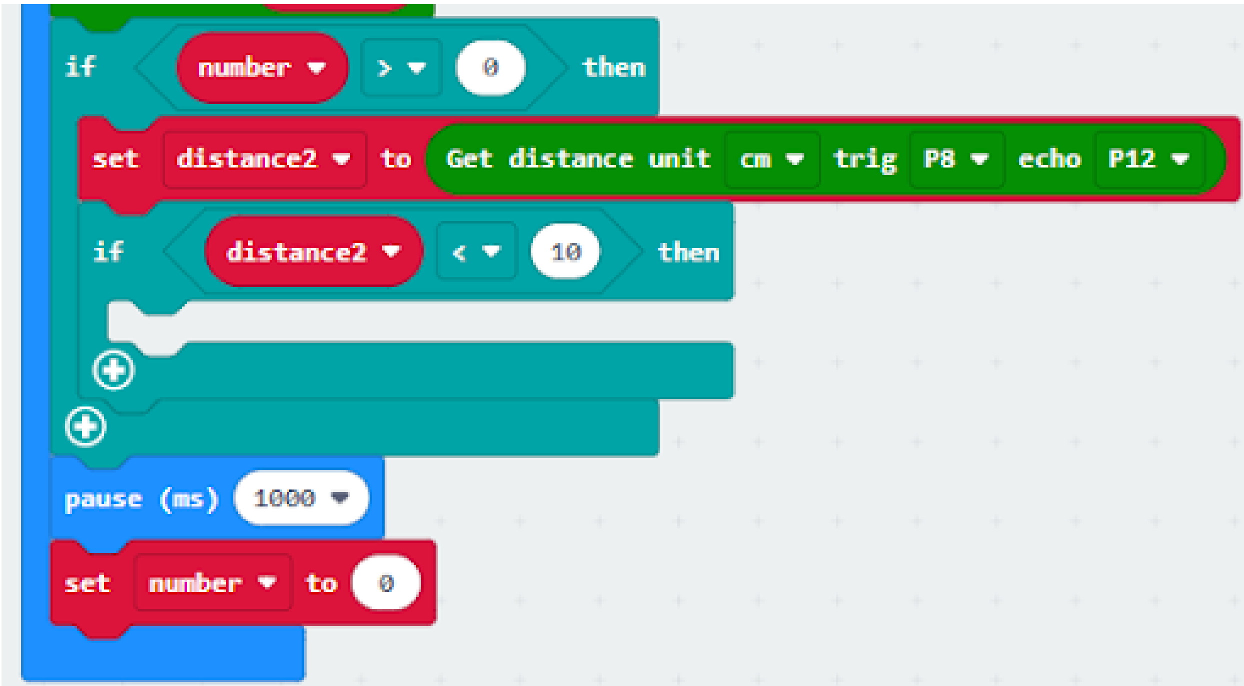
Step 4 display on OLED

- Snap clear OLED display from OLED to avoid overlap
- Snap show number and show value of variables number



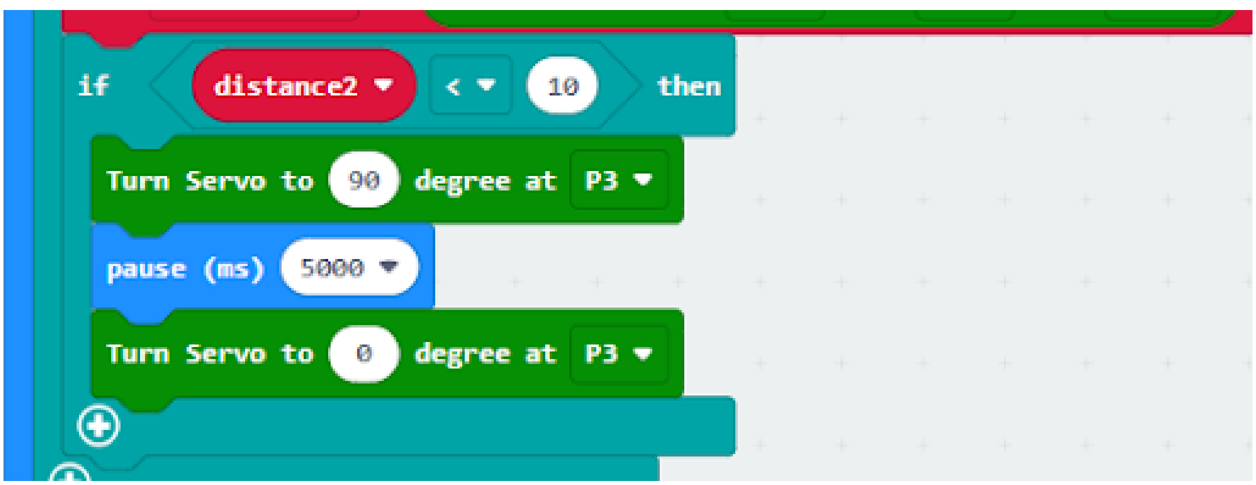
Step 5. Open/close gate with distance value

- Snap if statement into forever, set variable number>0
- Drag get distance2 to distance unit cm trig P8 echo P12, store the value to variable distance2.
- Snap if statement into forever, set variable distance2 < 10
- Snap Pause to the loop to wait 1 second for next checking
- Reset number to 0 before next checking



Step 6. Set servo position

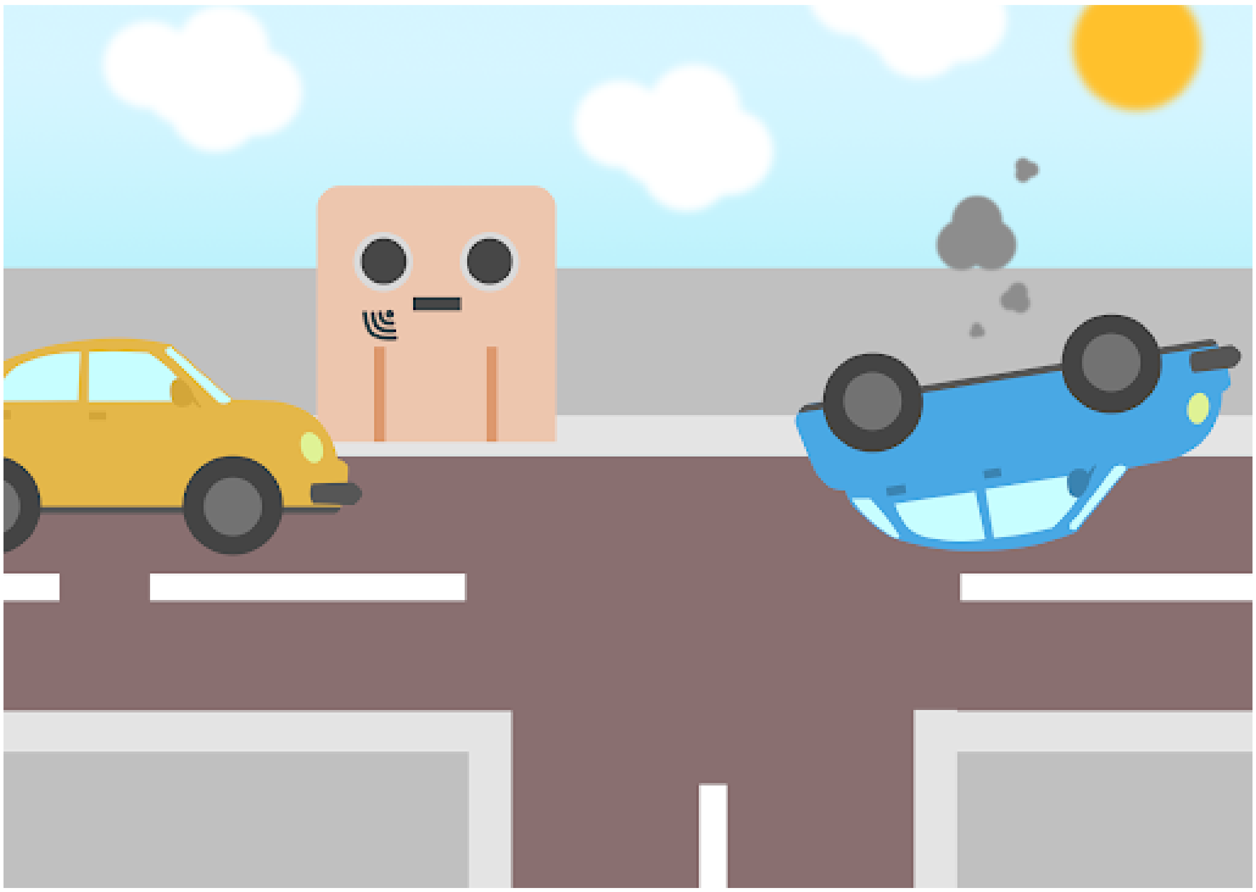
- Snap Turn Servo to 90 degree at P3 as the gate is opened.
- Snap pause to the loop to wait 5 seconds
- Snap Turn Servo to 0 degree at P3 as the gate is closed.



Result

Think

1.2.7 Broken Car

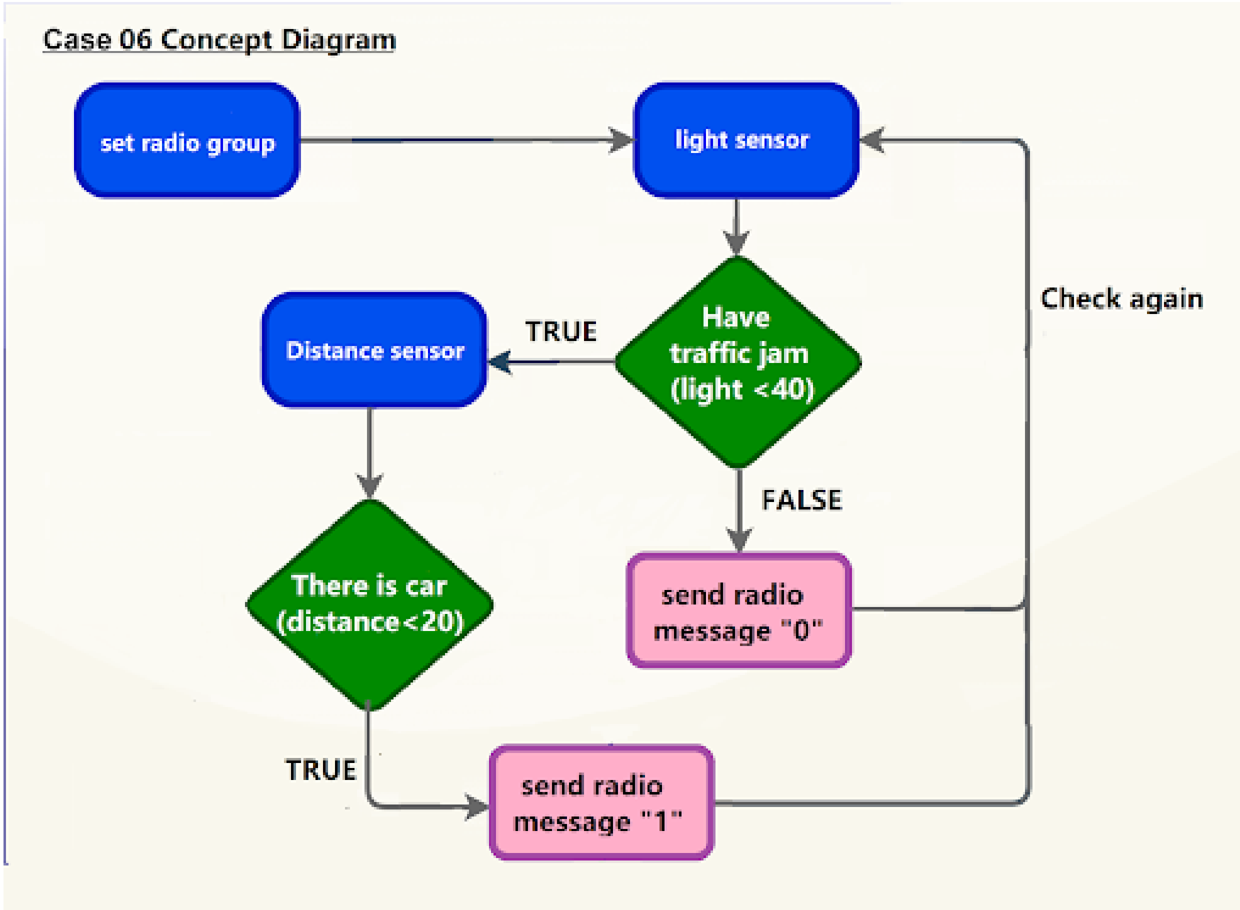


Goal

Background

What is a smart traffic system ?

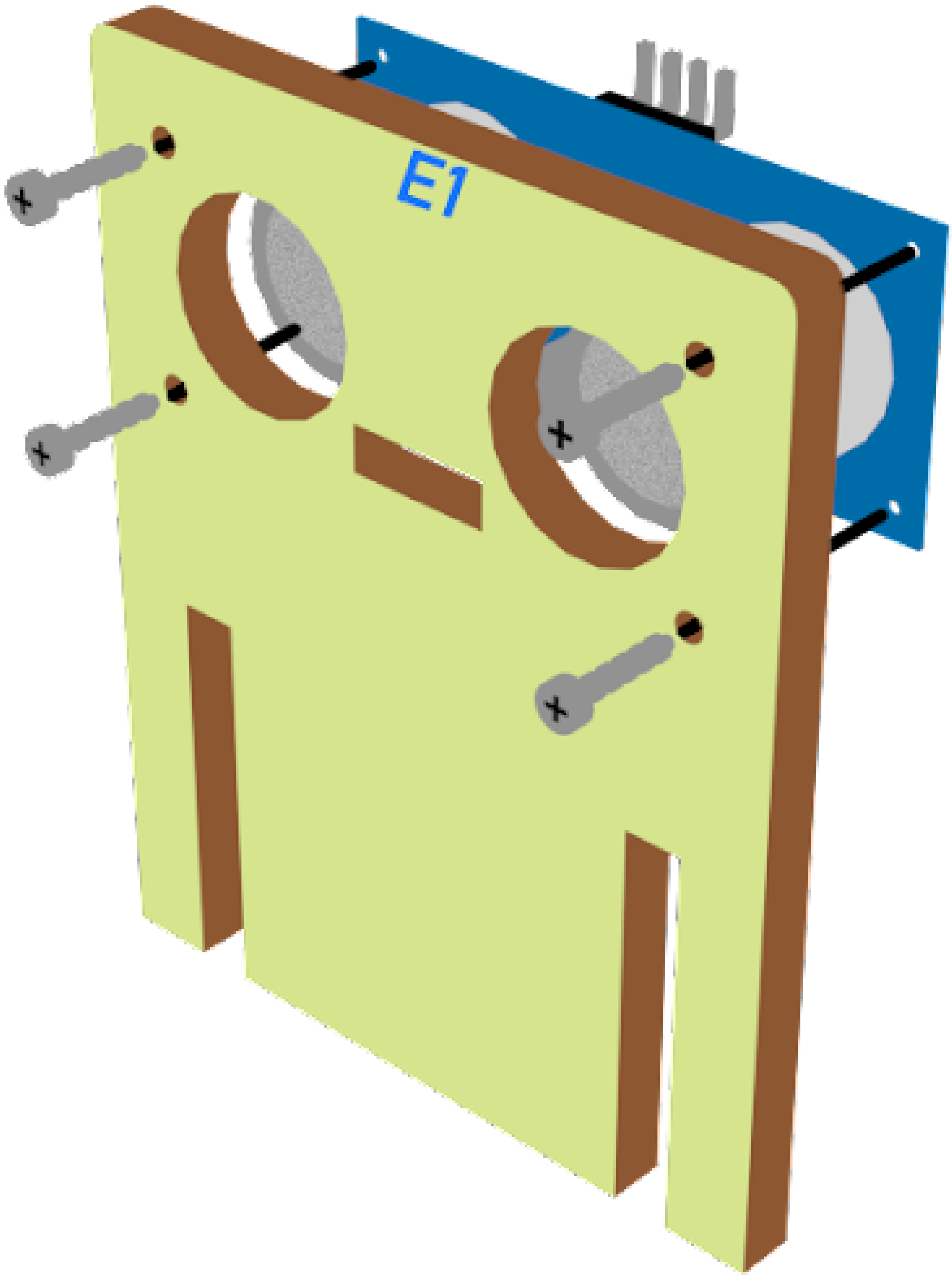
Smart traffic system Operation



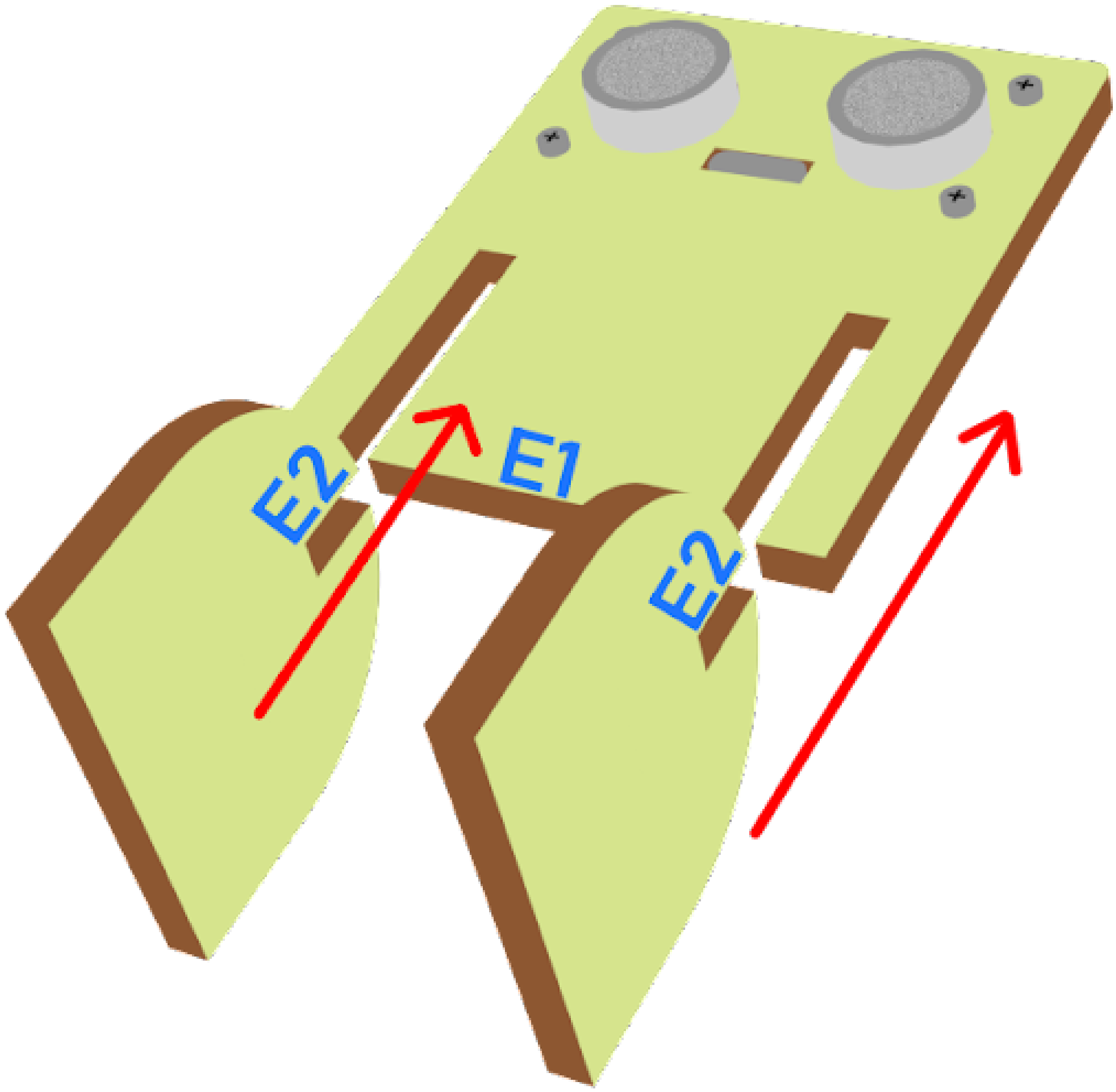
Part List

Assembly step

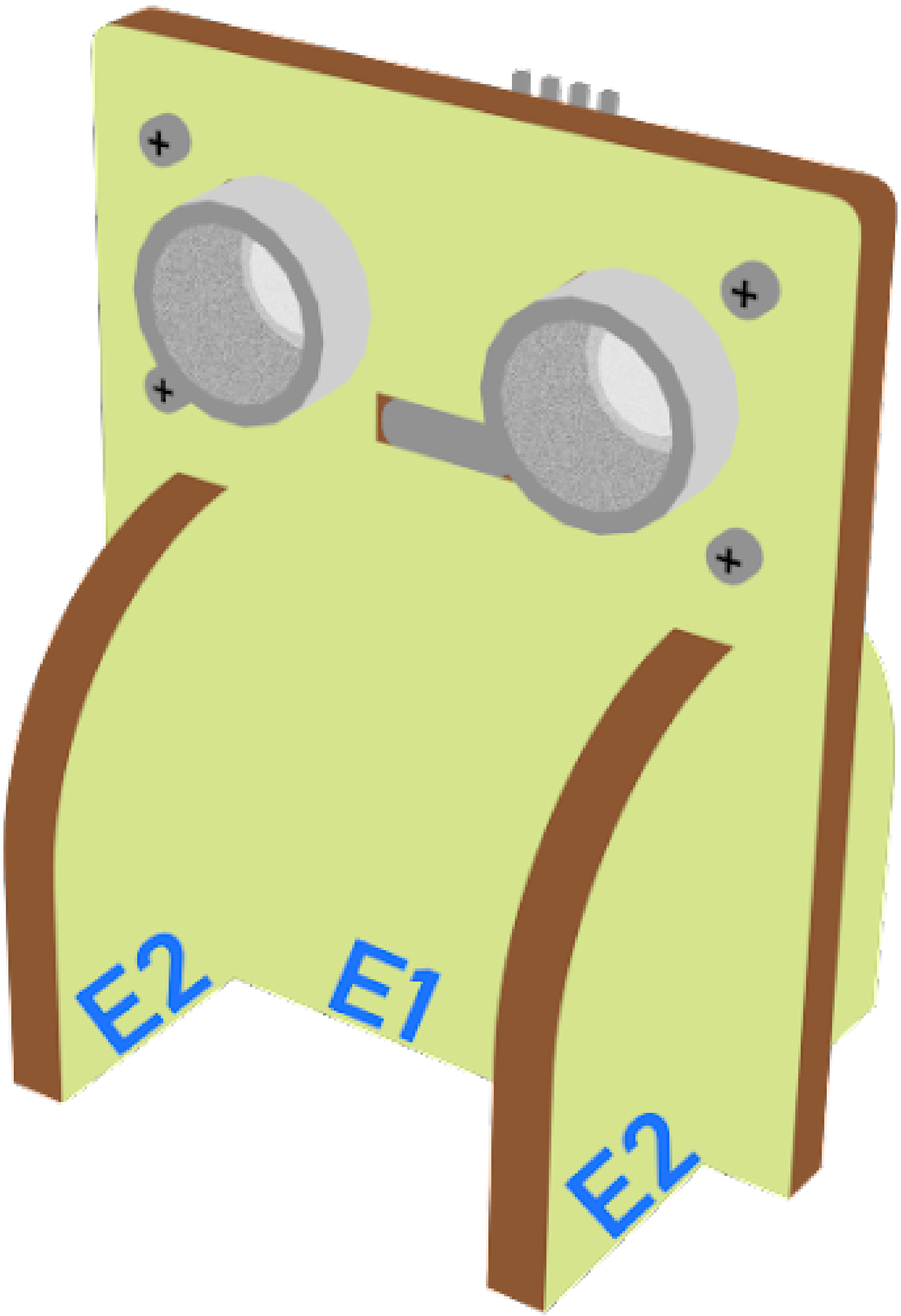
Step 1



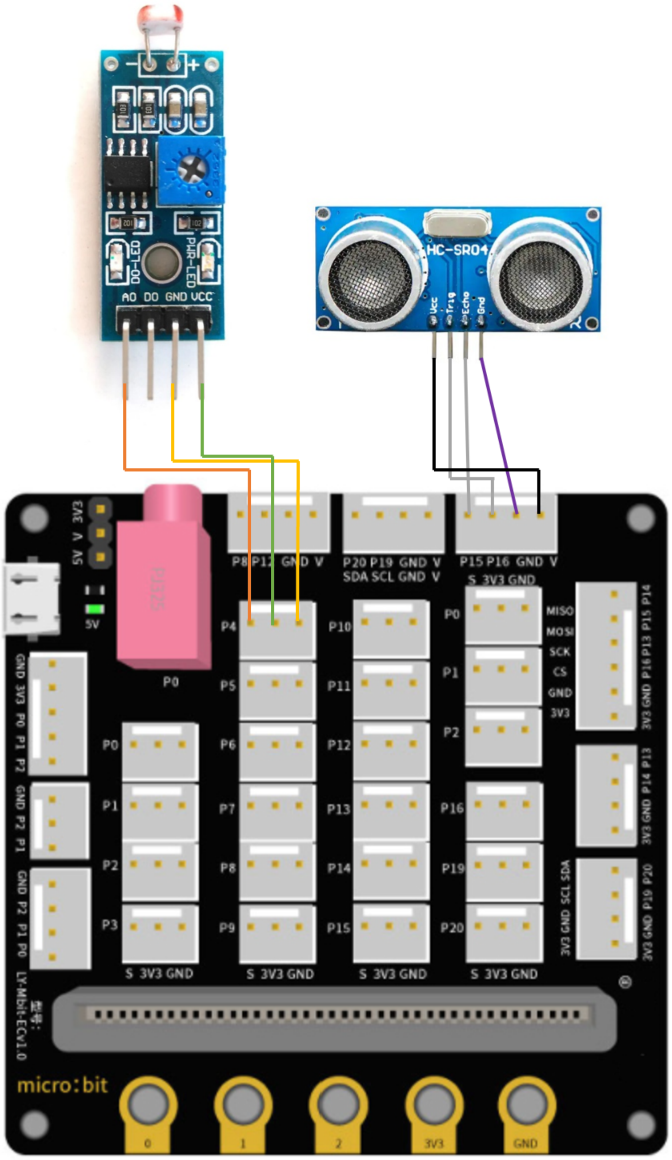
Step 2

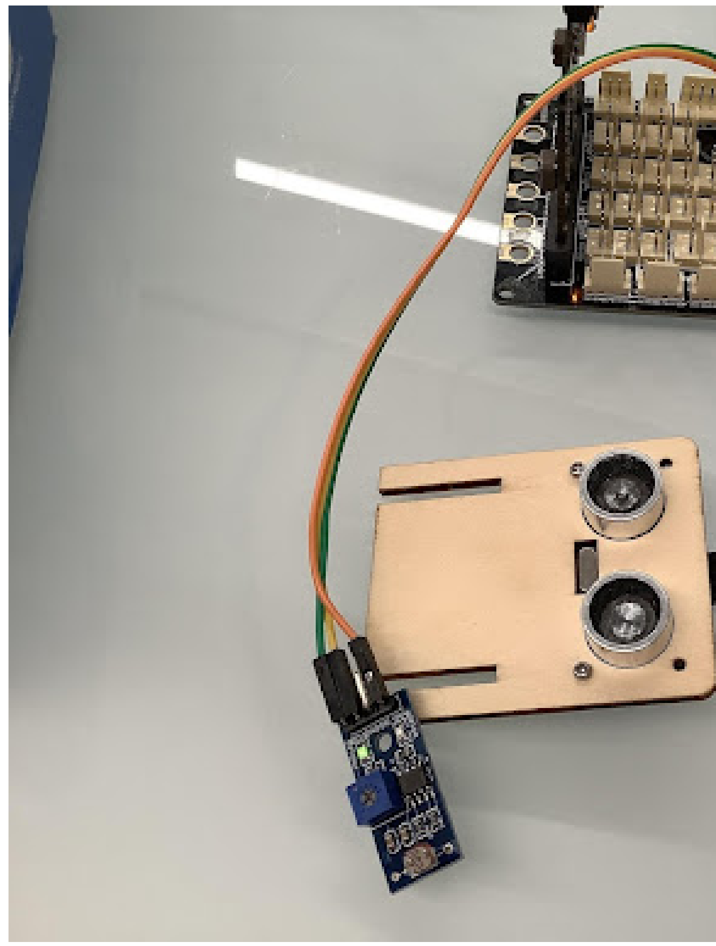


Step 3



Hardware connect



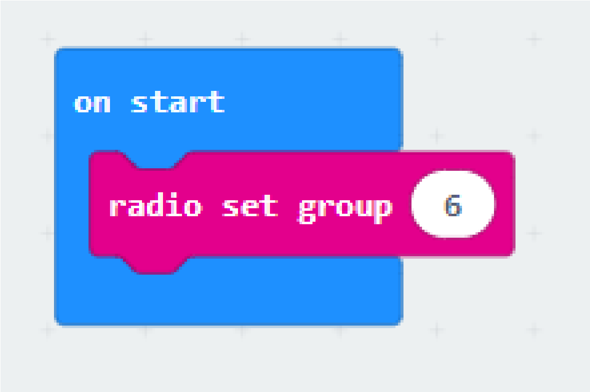


Programming (MakeCode)

Sender

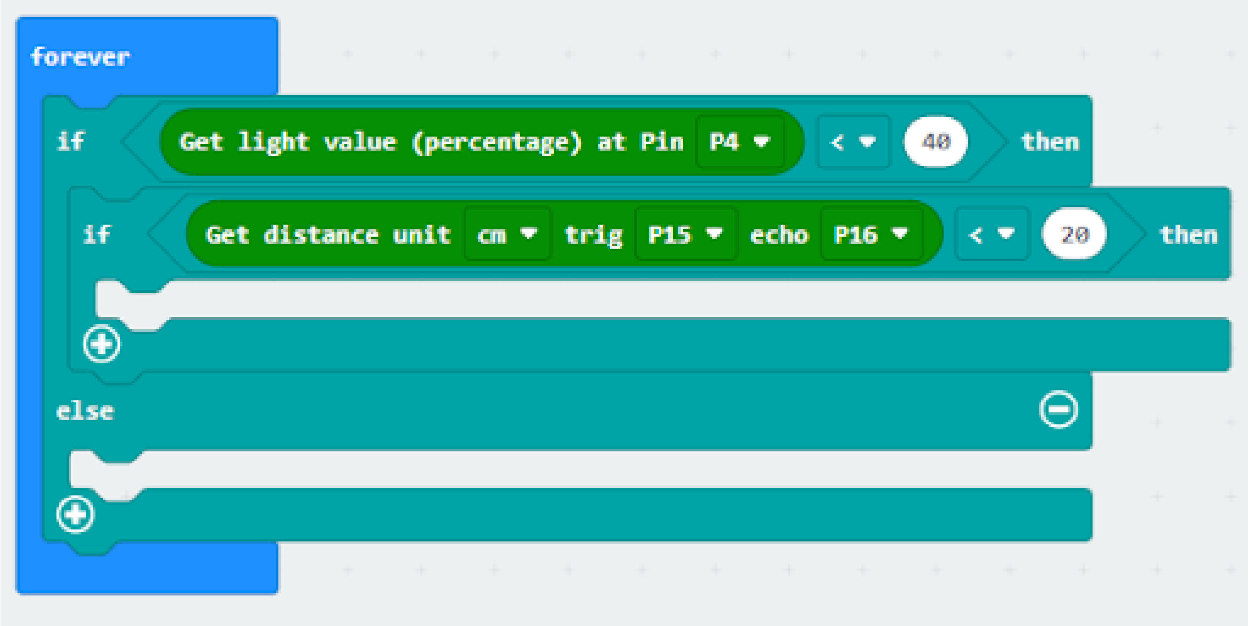
Step 1. Set radio set group at start position

- Drag radio set group 6 to on start



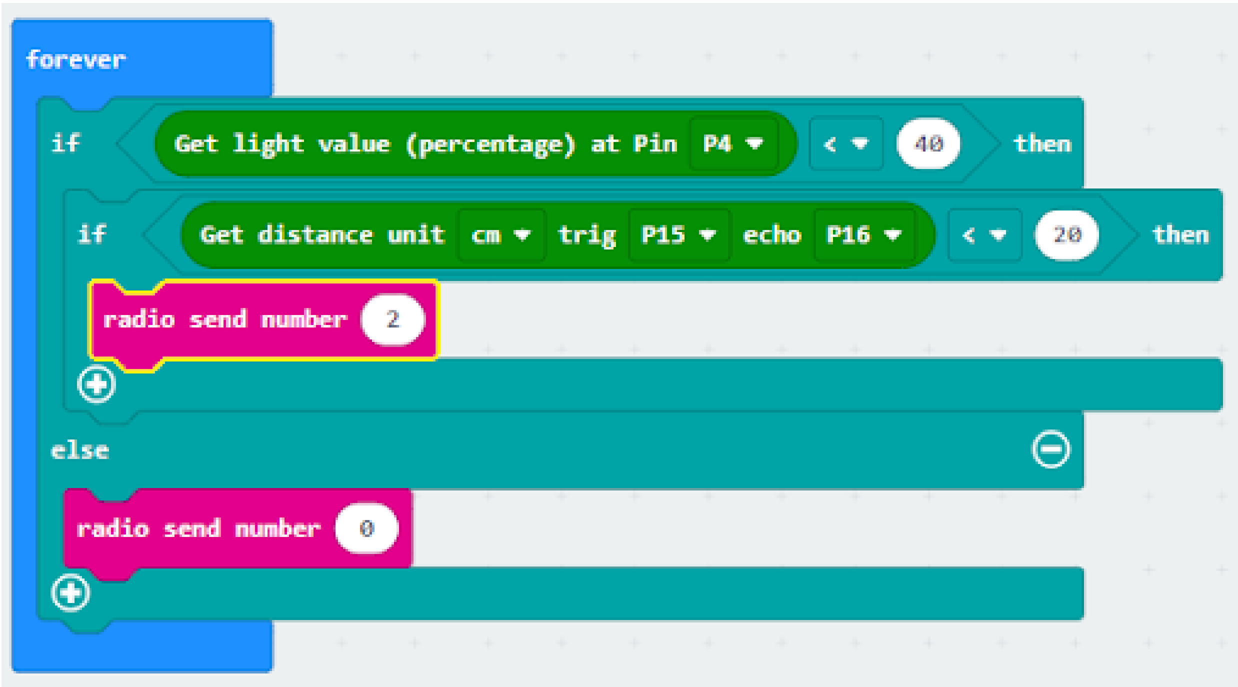
Step 2. Get light and distance value

- Snap if statement into forever, set get light value (percentage) at pin P4 < 40
- If get light value (percentage) at pin P4 < 40, and else if get distance unit cm trig P15 echo 16 < 20



Step 3. Control the car by sending radio number

- Drag radio send number to 2 into if
- Drag radio send number to 0 into else



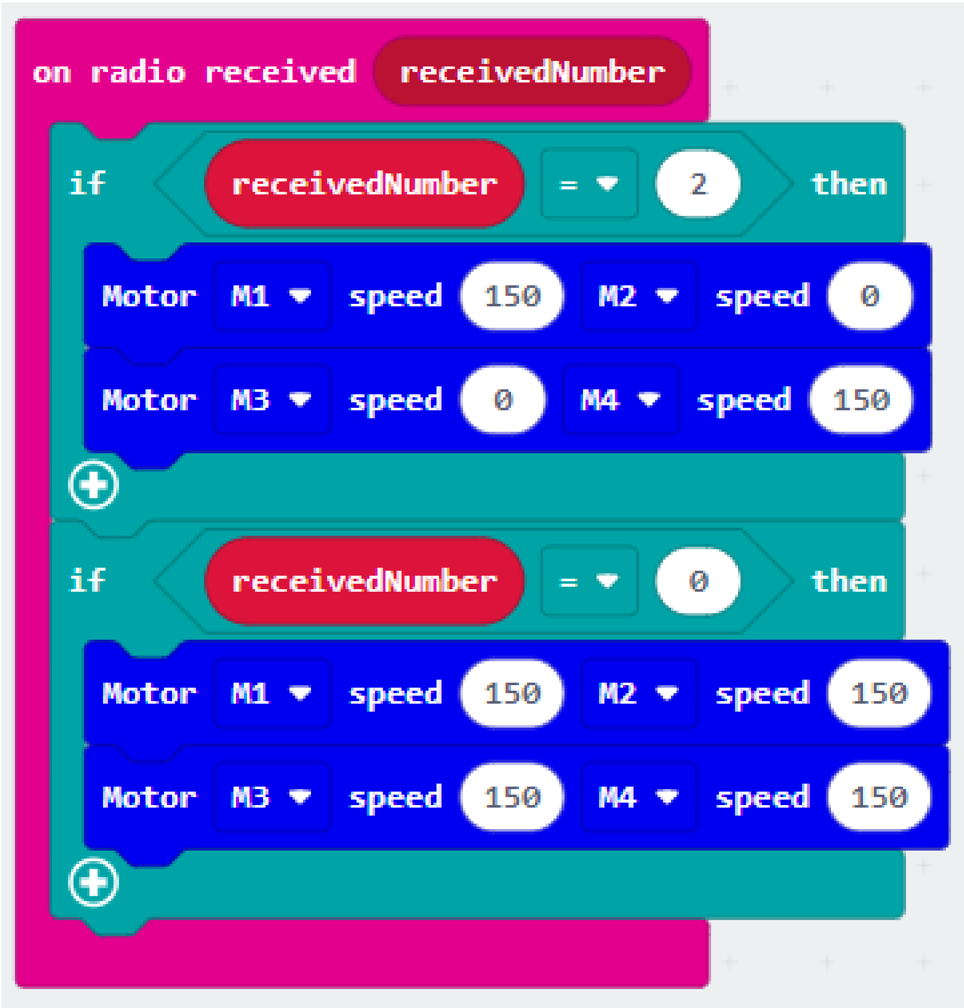
Receiver

Step 1. Set radio set group at start position

- Drag radio set group 6 to on start
- Initially, the car moves forward by default

Step 2. Control car by receiving different number

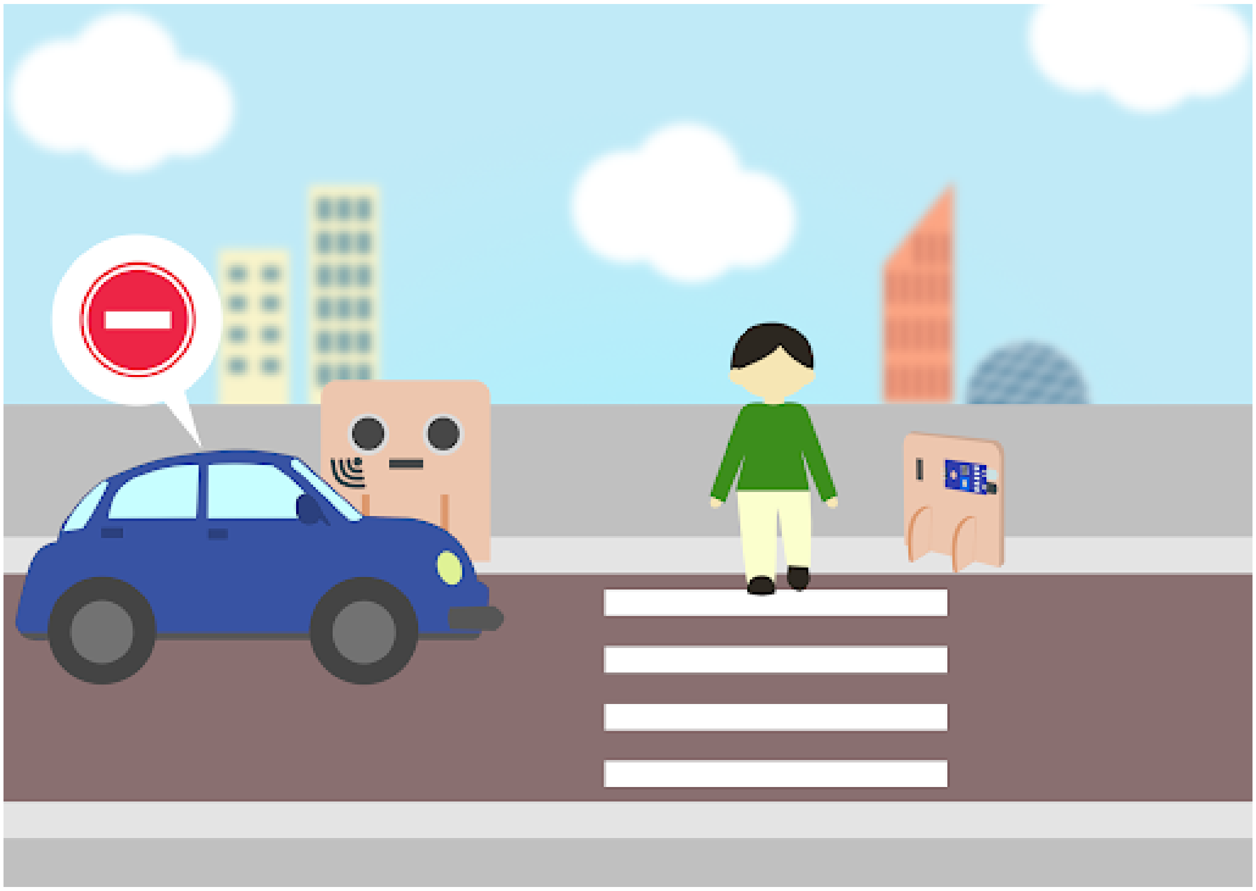
- Snap if statement into on radio received receivedNumber
- Set receivedNumber =2 and make the car turn left
- Set receivedNumber=0 and make the car move forward



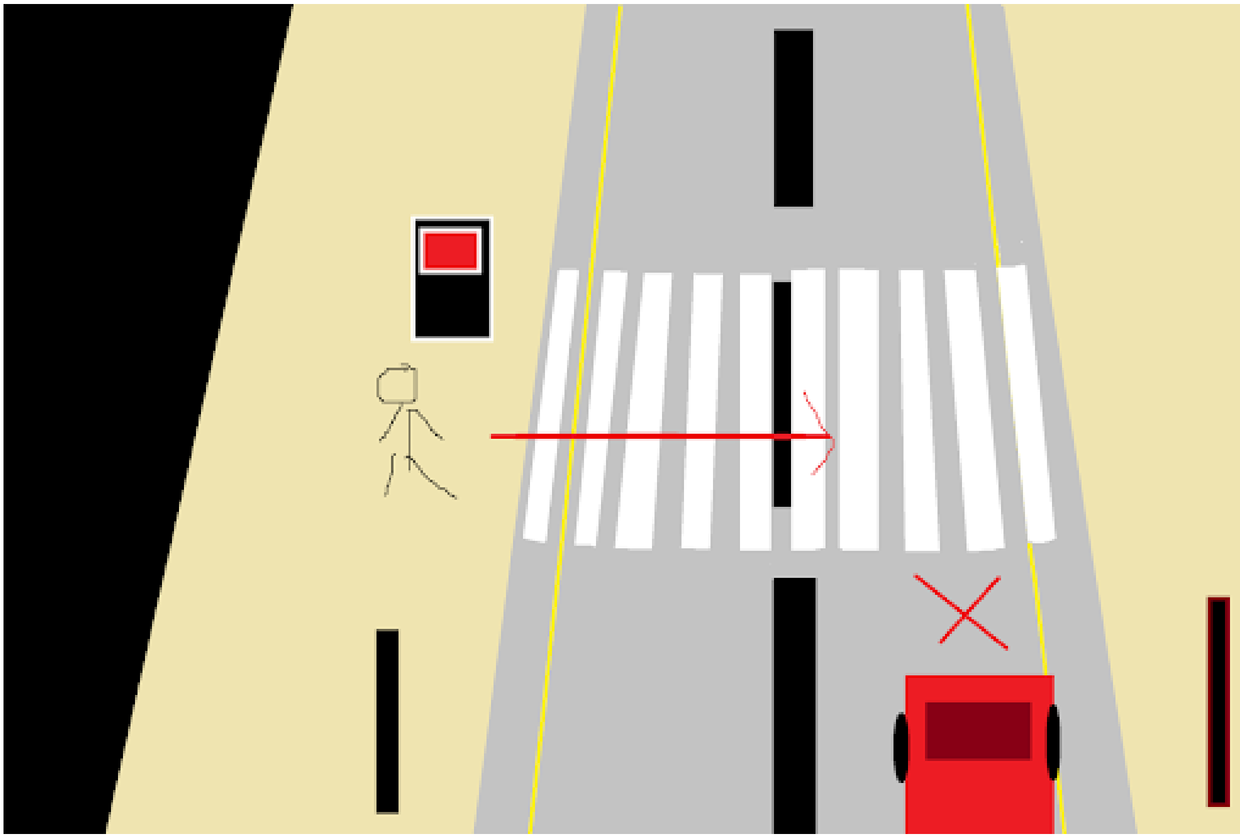
Result

Think

1.2.8 Crosswalk



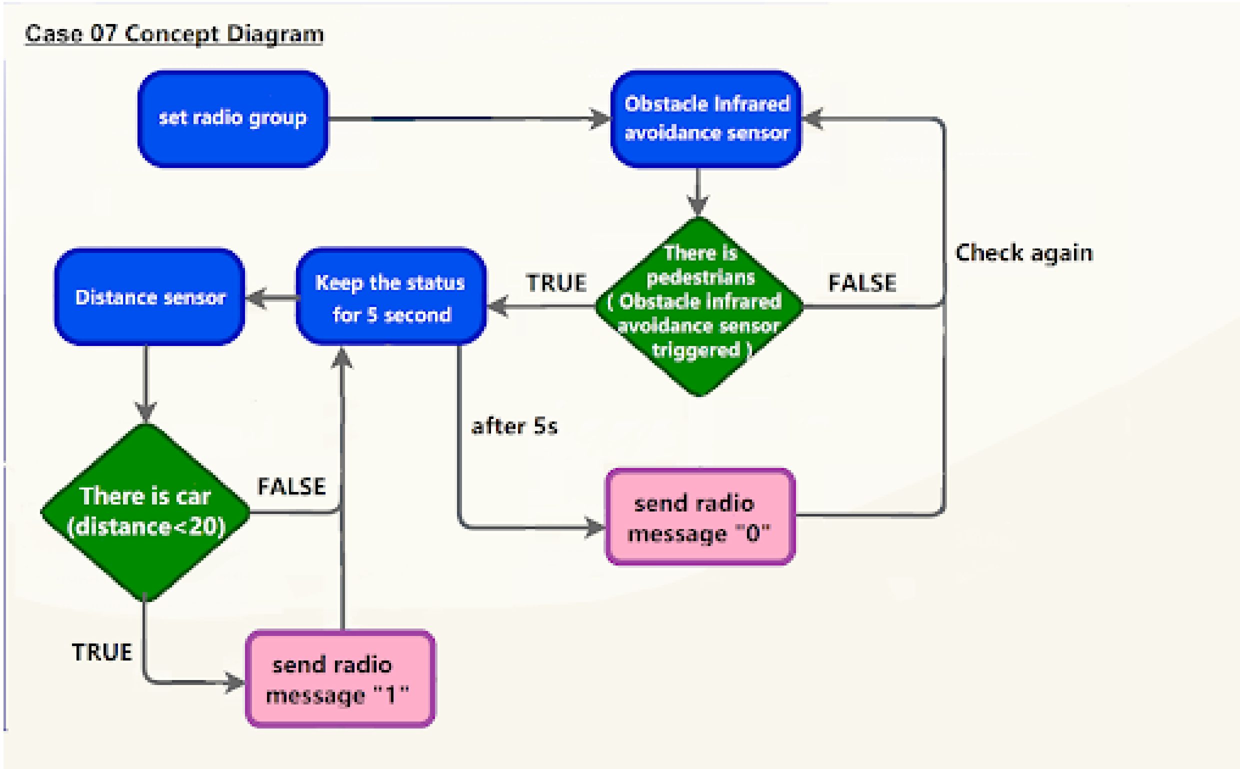
Goal



Background

What is a smart crosswalk system?

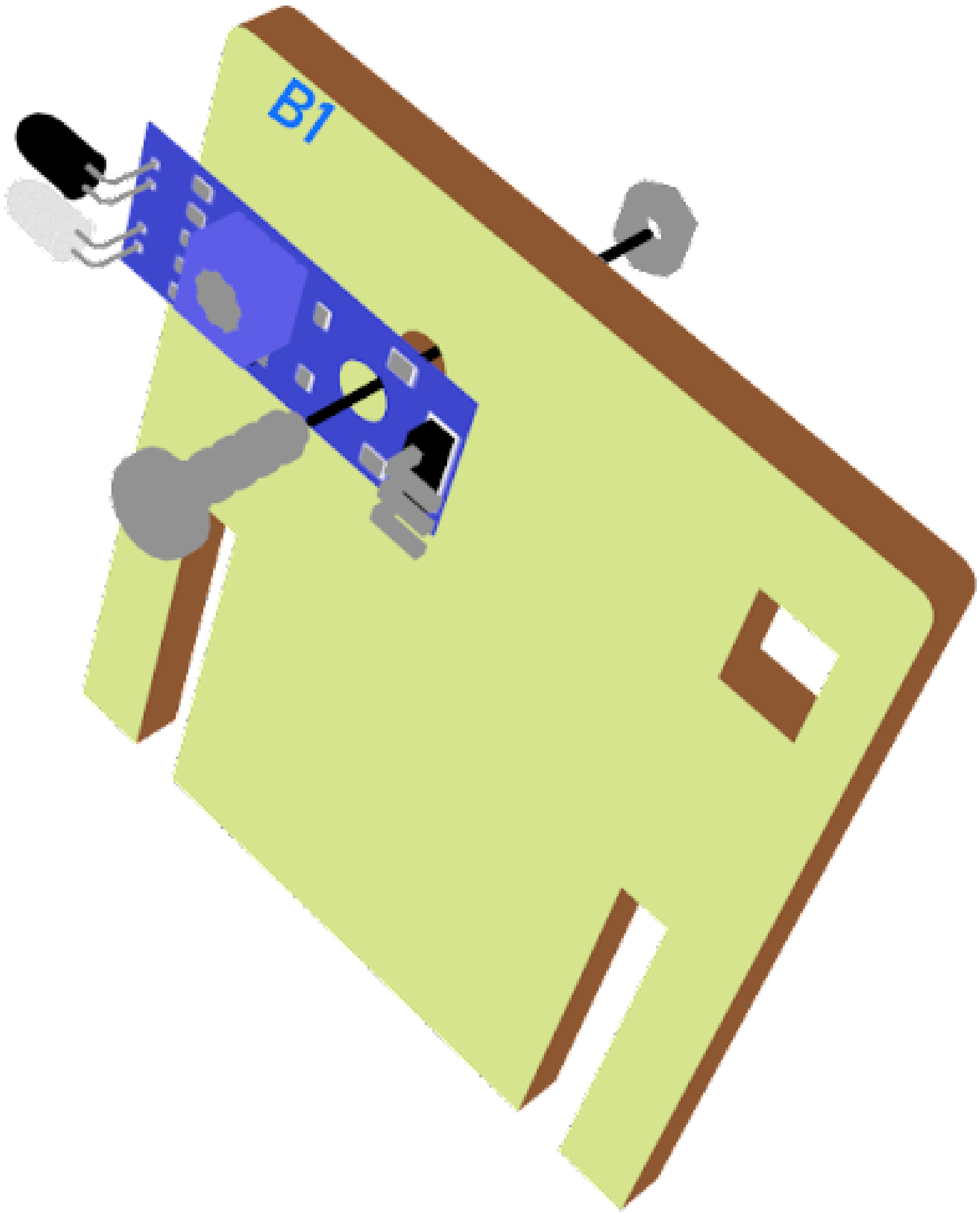
Smart crosswalk system operation



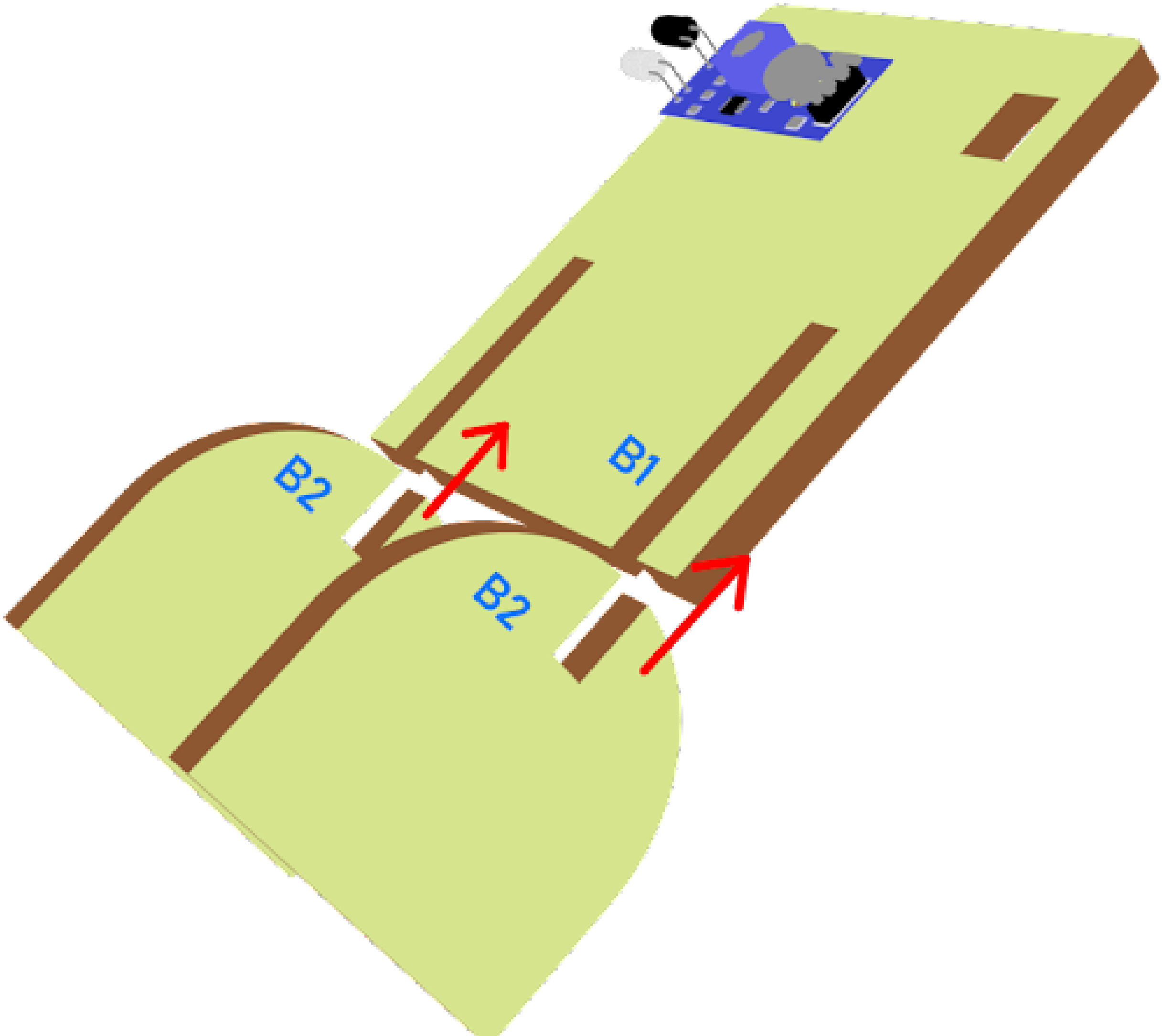
Part List

Assembly step

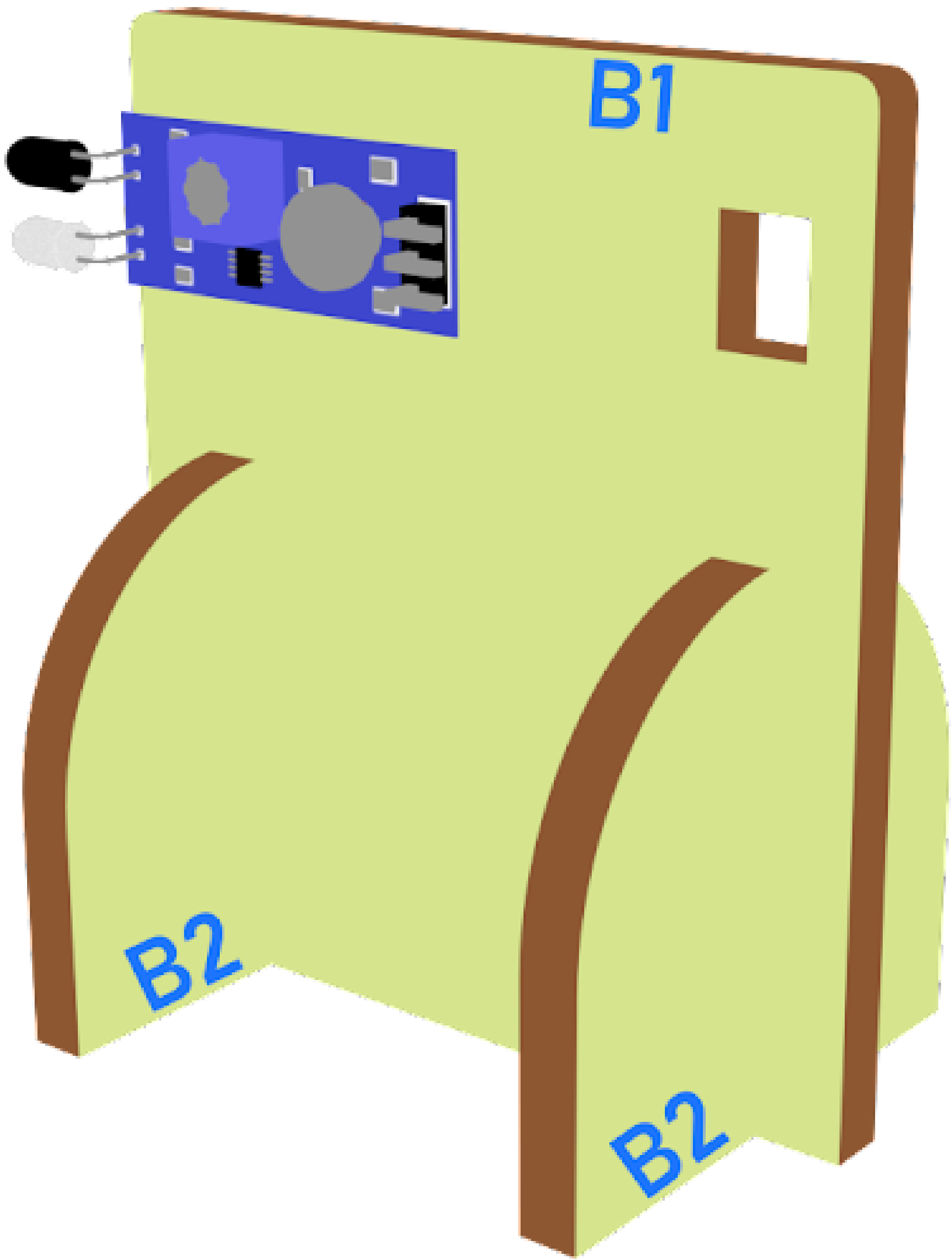
Step 1



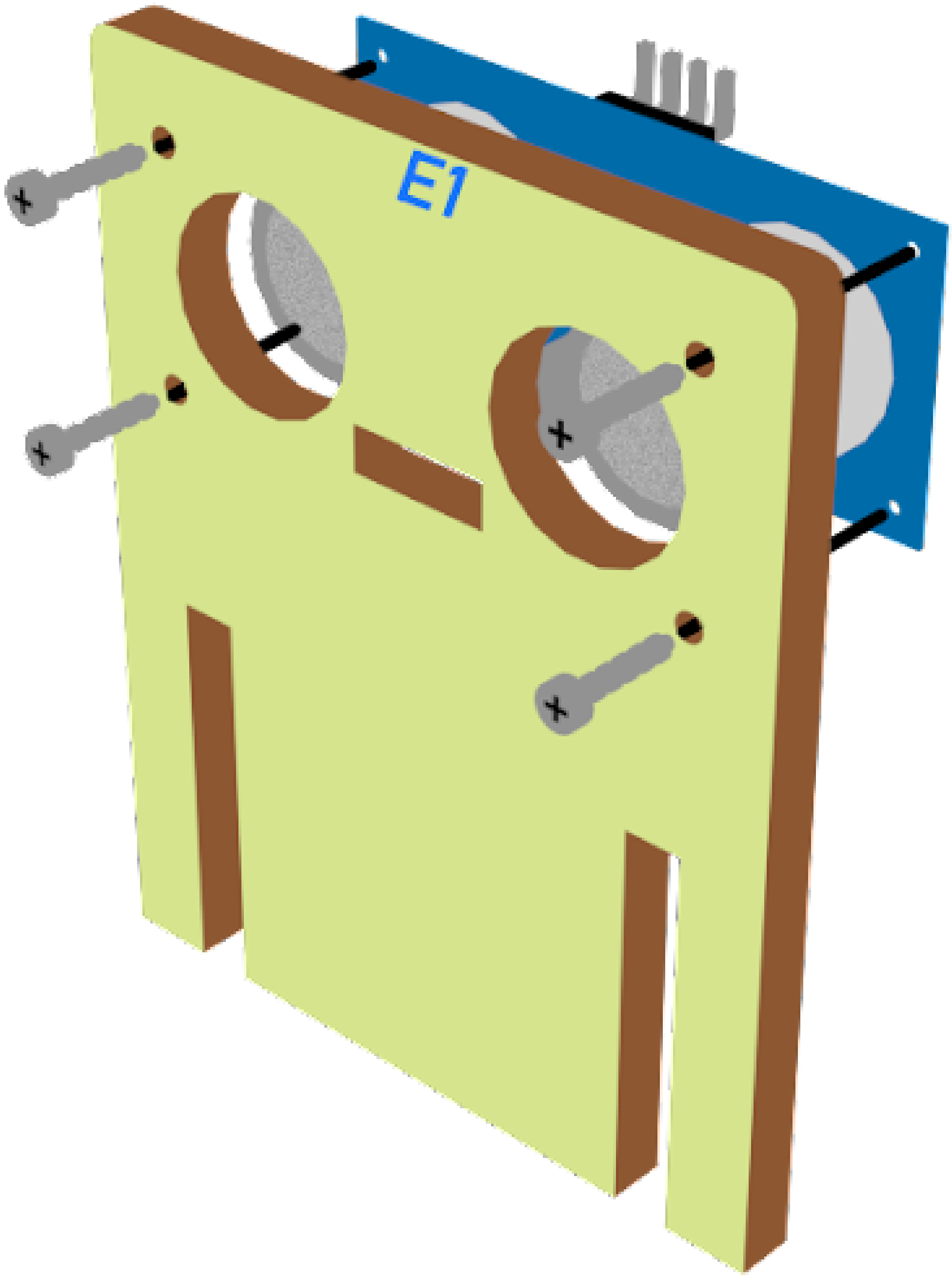
Step 2



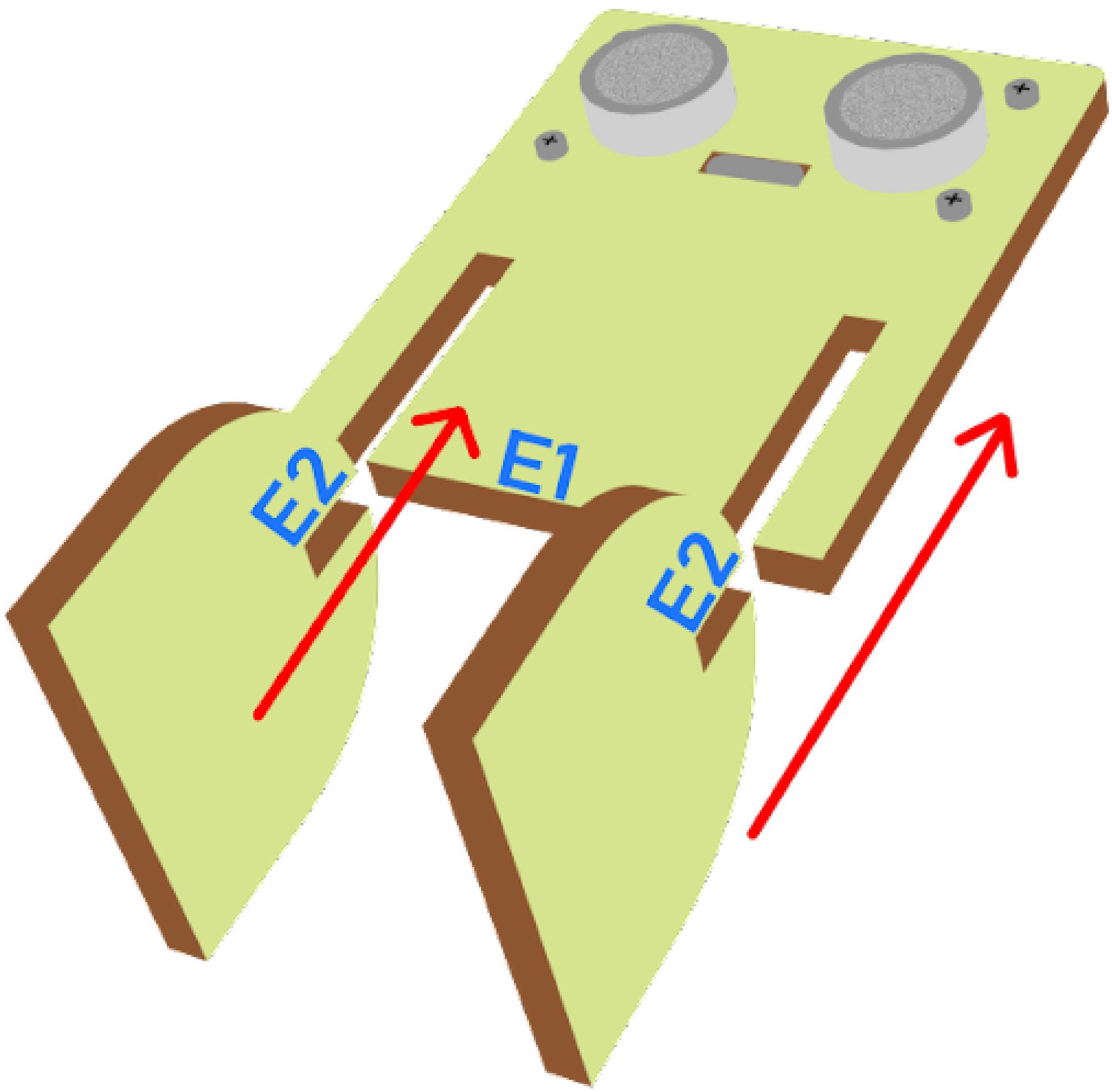
Step 3



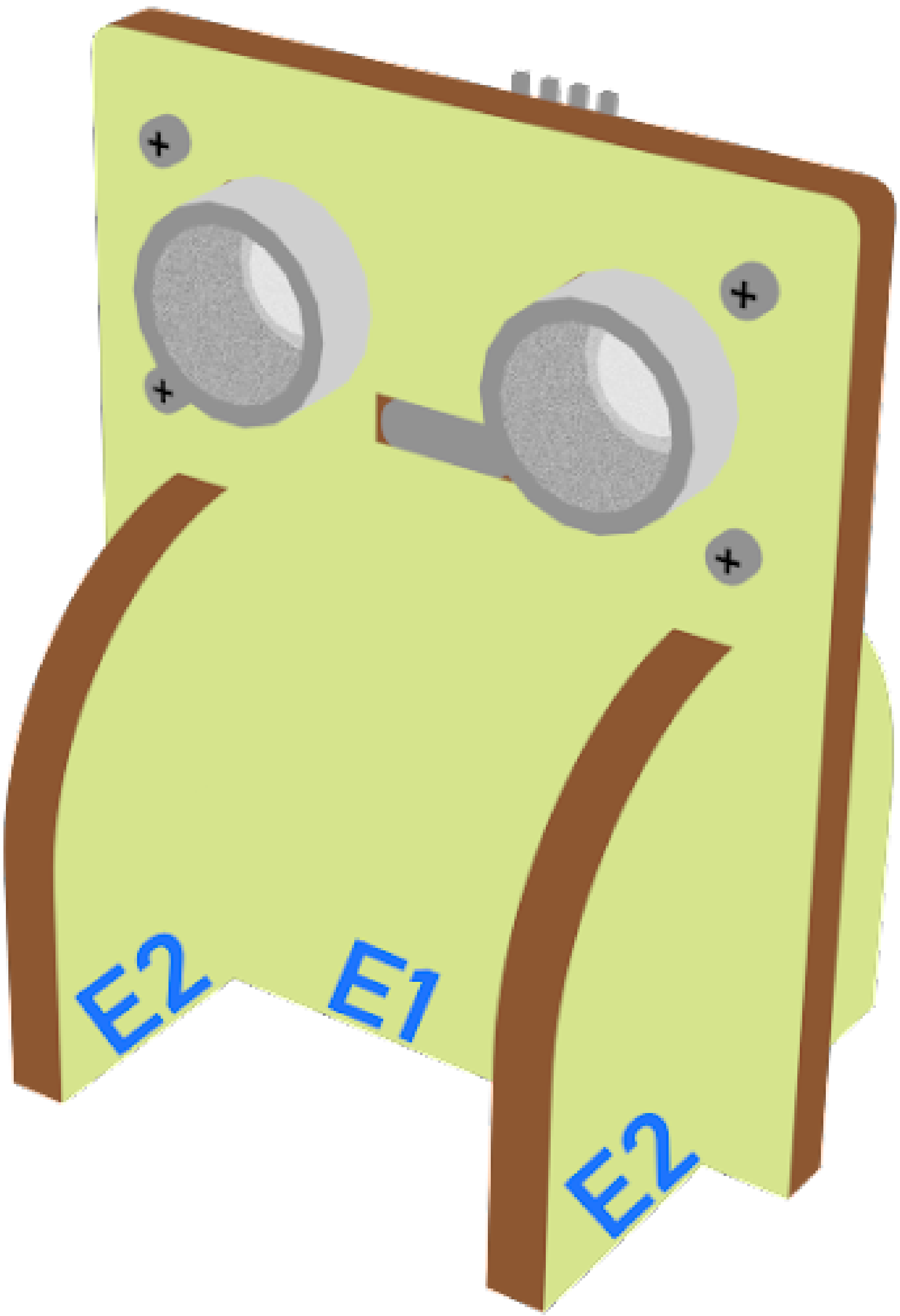
Step 4



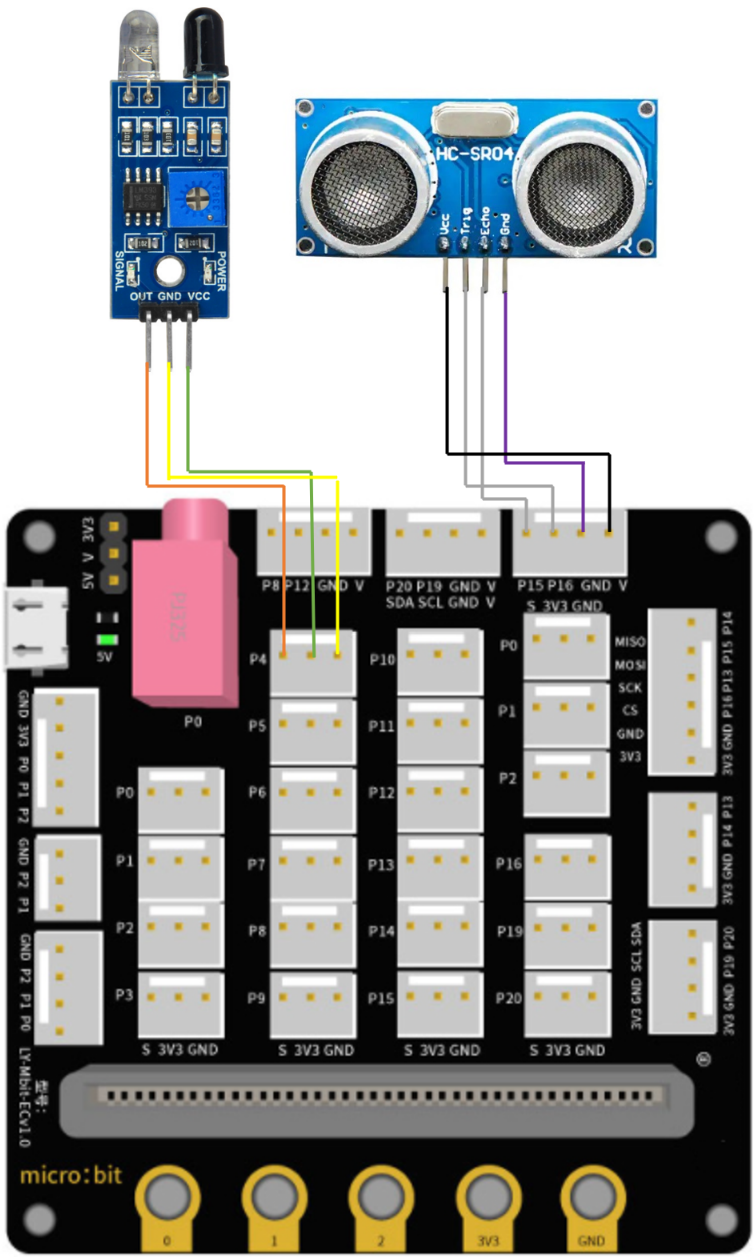
Step 5

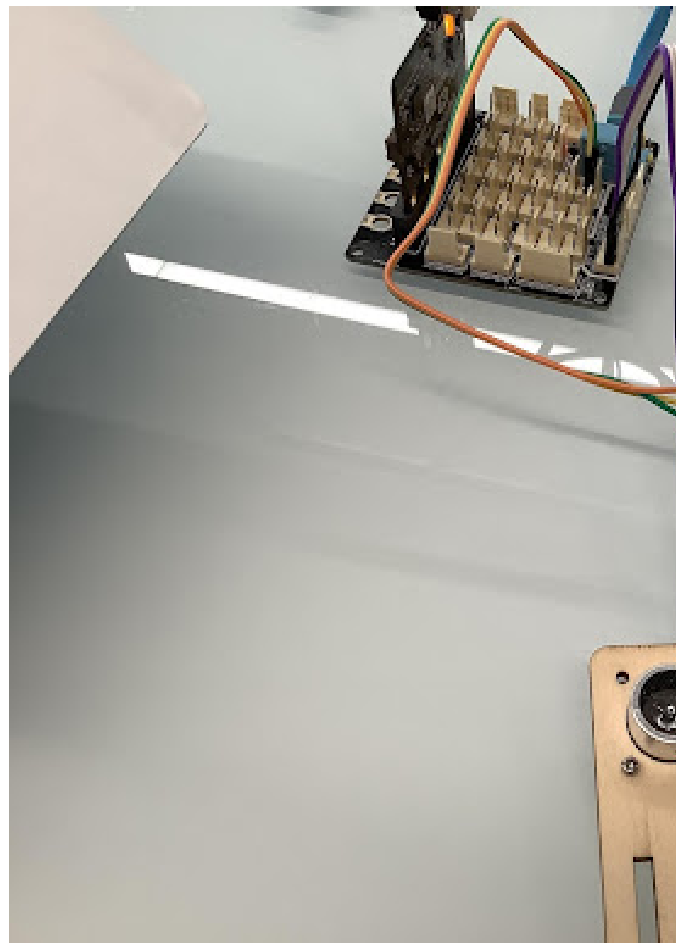


Step 6



Hardware connect





Programming (MakeCode)

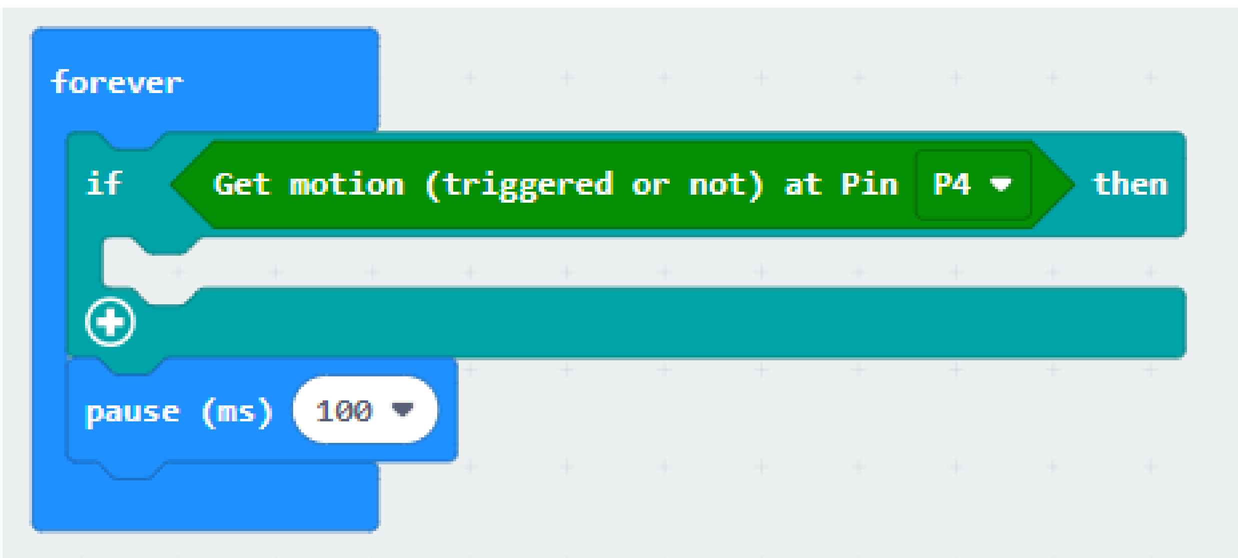
Sender:

Step 1. Set radio set group at start position

- Drag radio set group 6 to on start

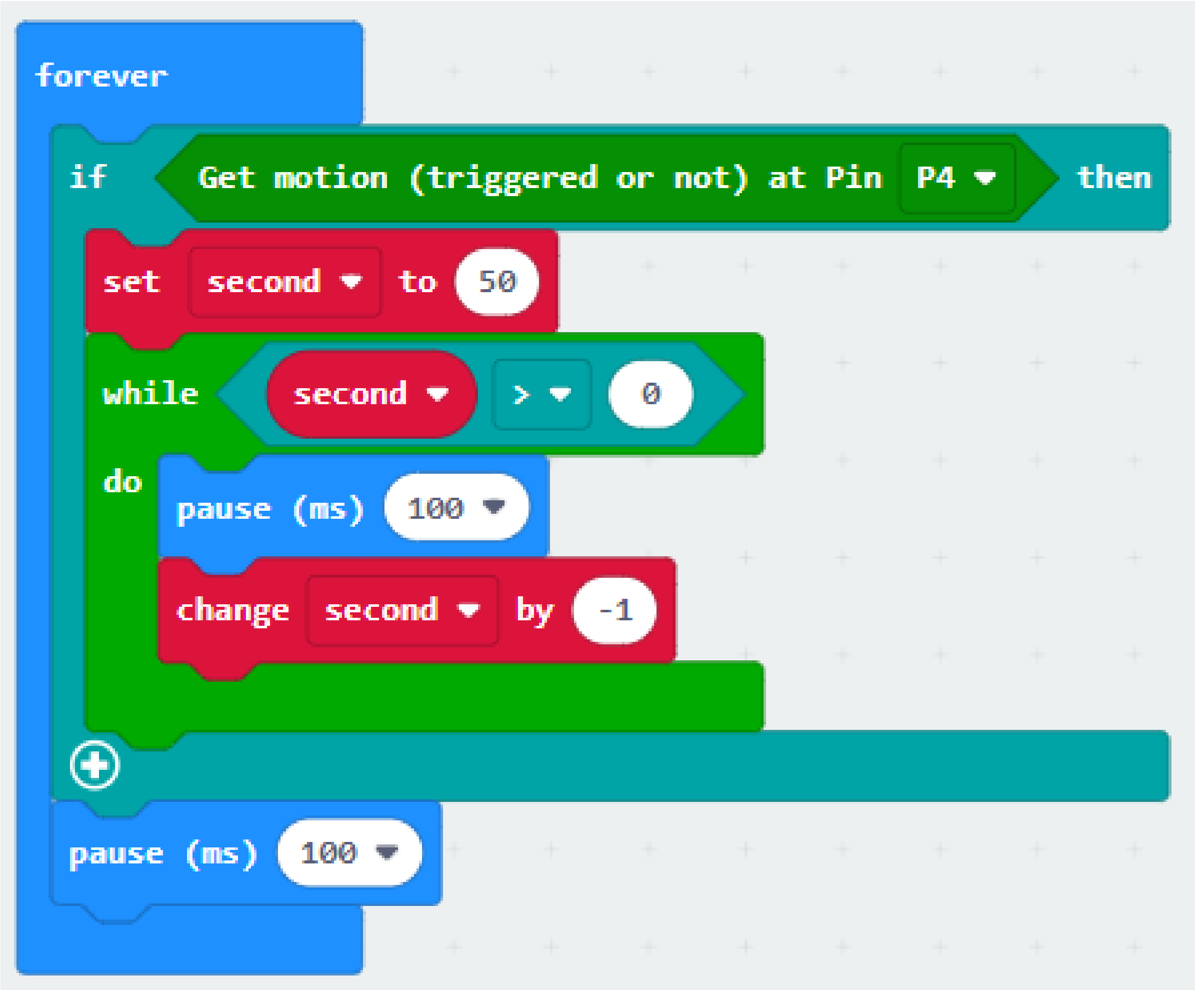
Step 2. When trigger a motion

- Snap if statement into forever, set get motion (triggered or not) at pin P4
- Snap pause to the loop to wait for 0.1 second for next checking



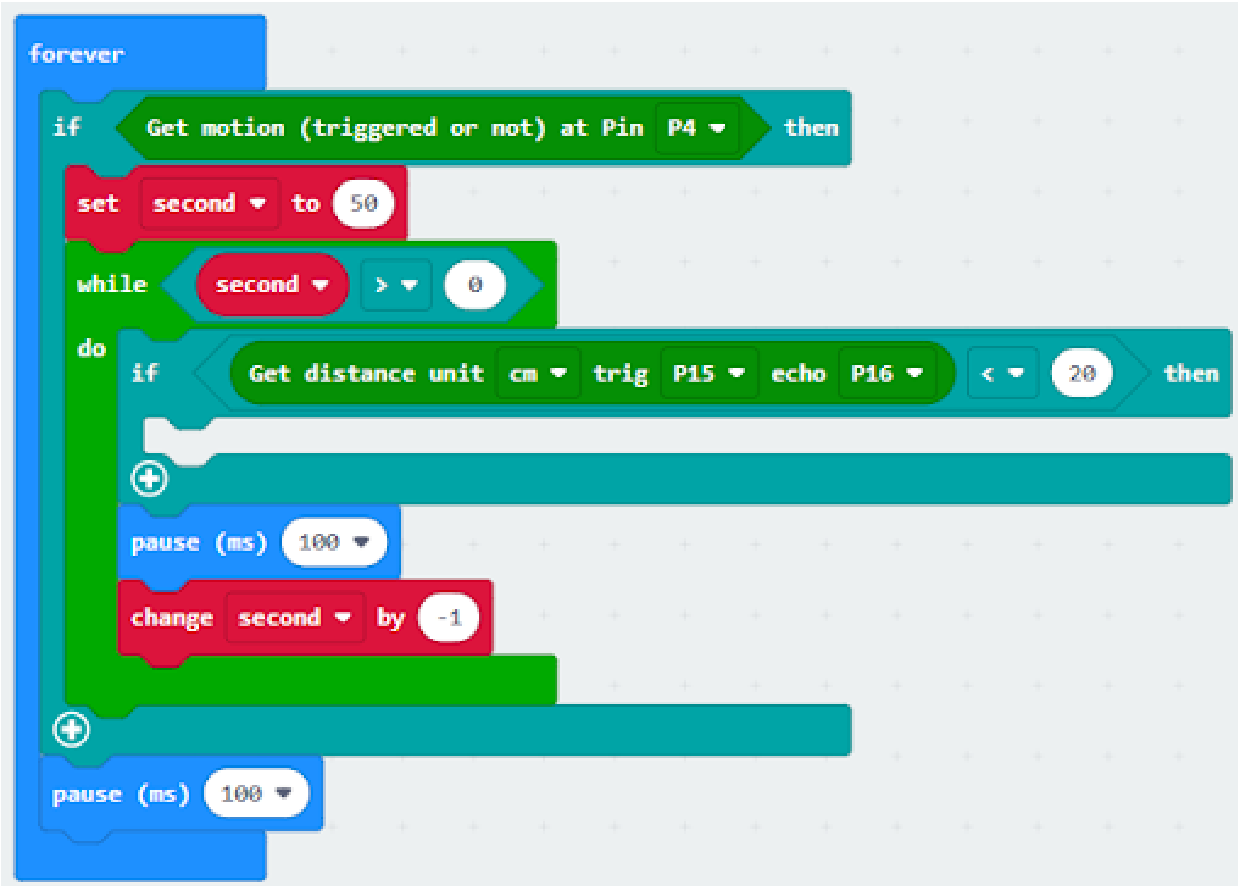
Step 3. Keep the status for 5 second

- set variable second to 50
- While second > 0, snap pause to 0.1 second and change second by -1.



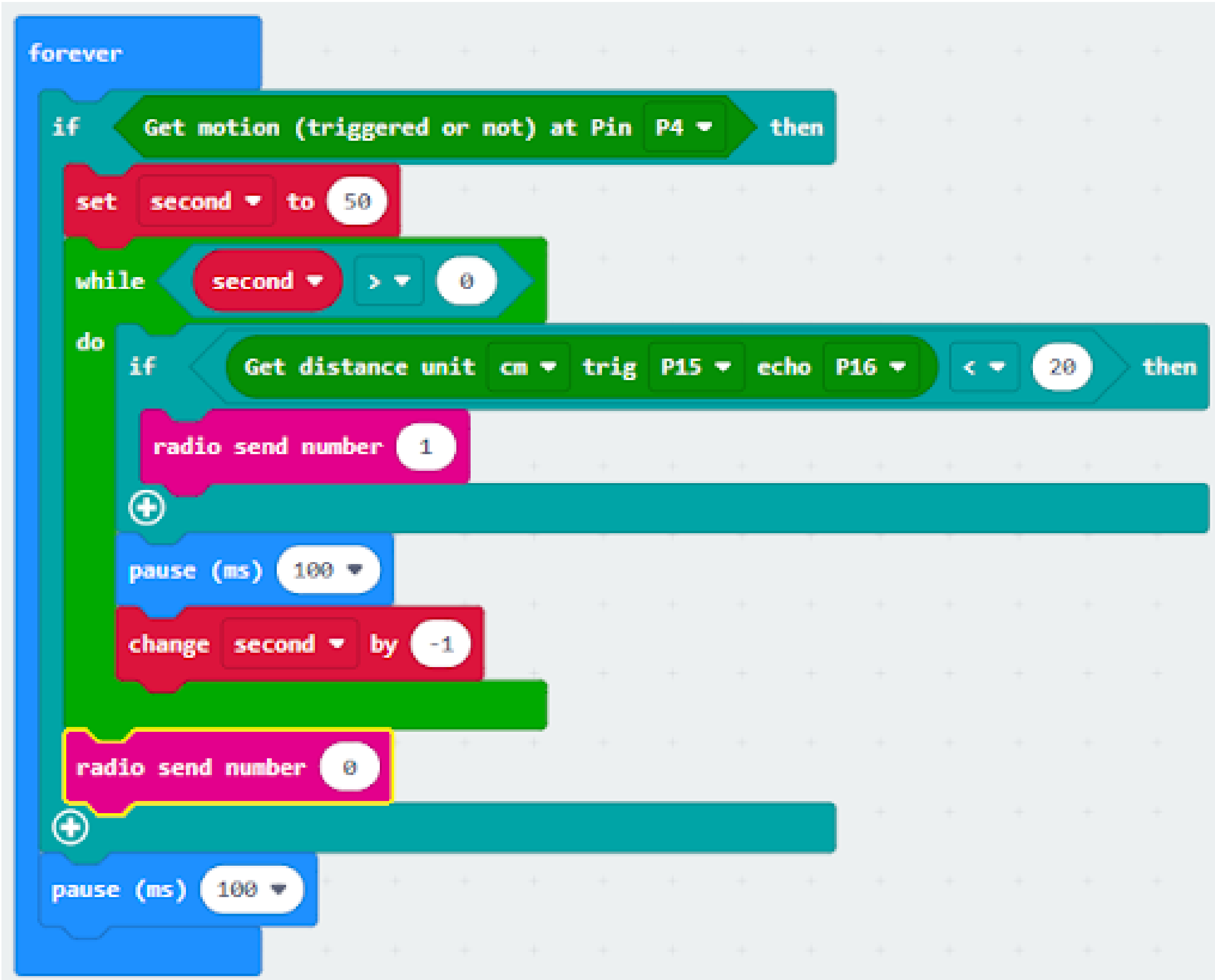
Step 4. Get distance value

- Snap if statement into while loop, set get distance unit cm trig P15 echo P16 < 20



Step 5. Control the car by sending radio number

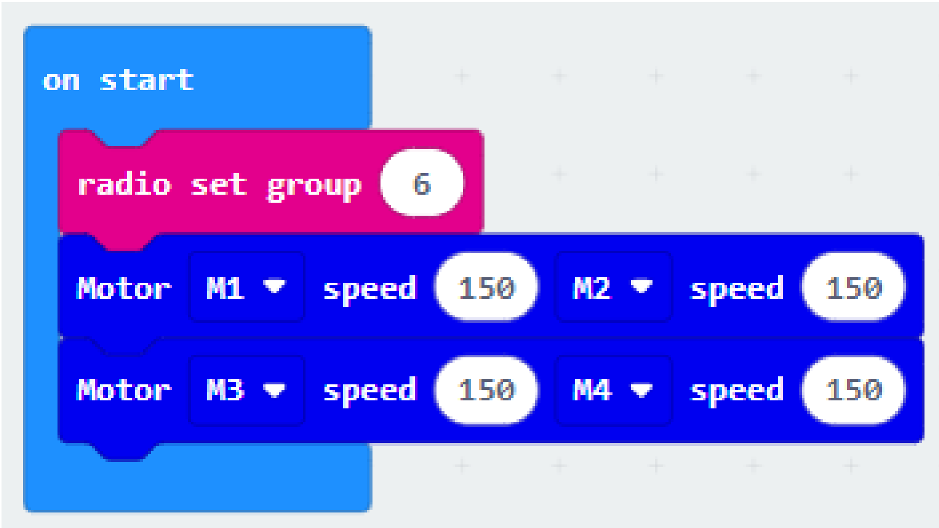
- Drag radio send number to 1 into if



Receiver

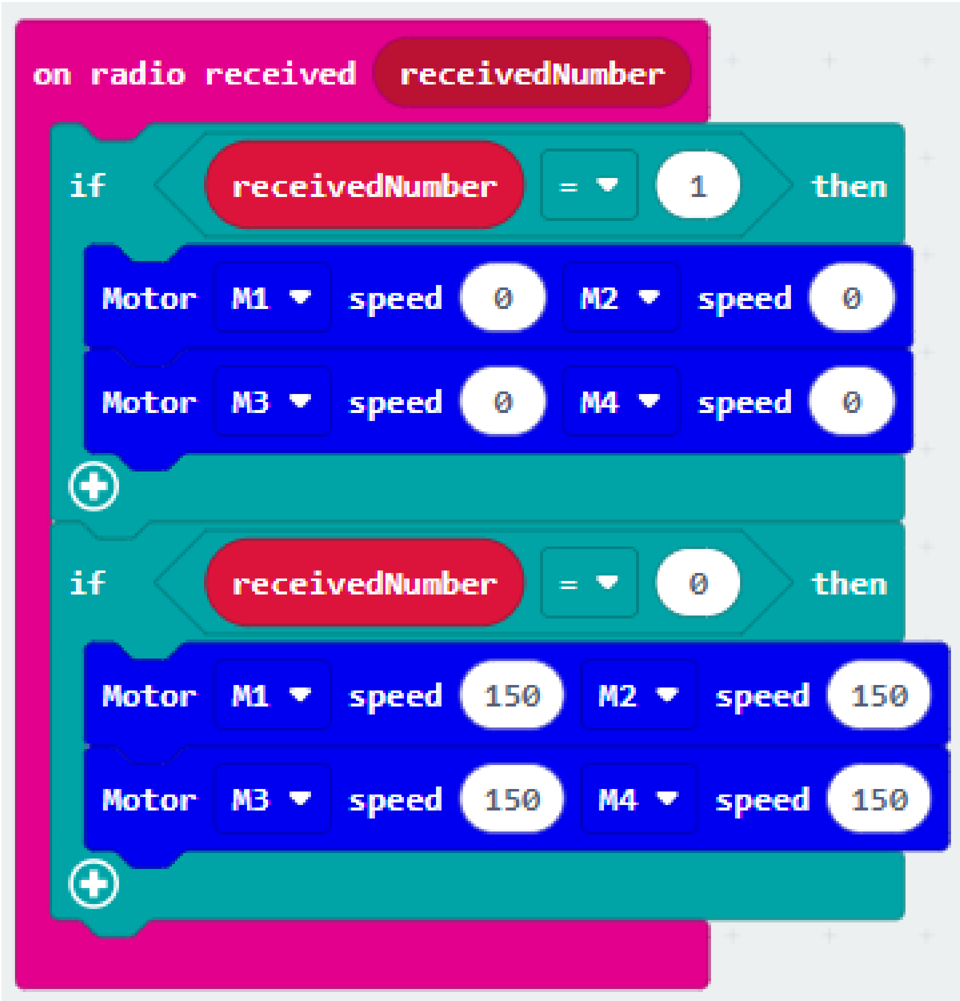
Step 1. Set radio set group at start position

- Drag radio set group 6 to on start
- Initially, the car moves forward by default



Step 2. Control car by receiving different number

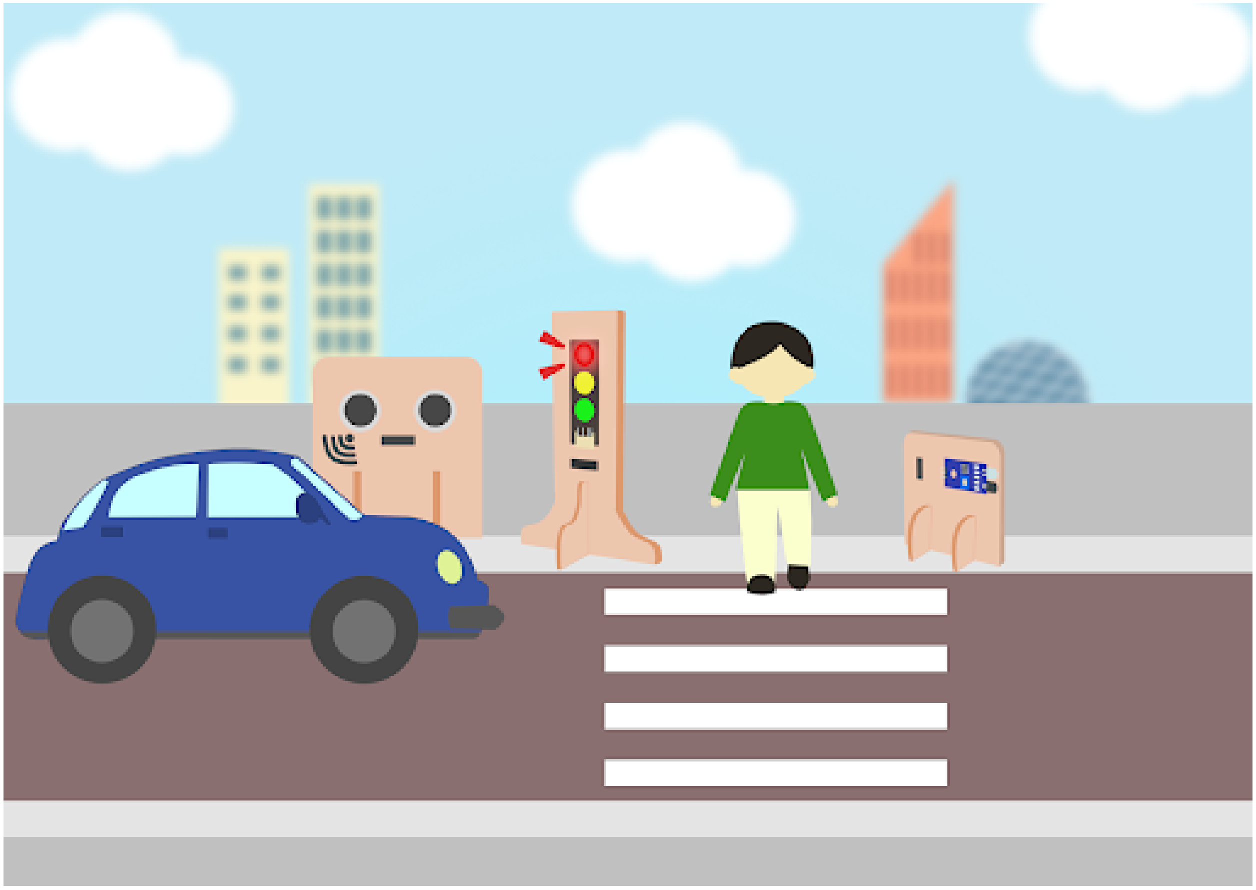
- Snap if statement into on radio received receivedNumber
- Set receivedNumber =1 and make the car stop
- Set receivedNumber=0 and make the car move forward


on radio received receivedNumber
if receivedNumber = 1 then
Motor M1 speed 0 M2 speed 0
Motor M3 speed 0 M4 speed 0
if receivedNumber = 0 then
Motor M1 speed 150 M2 speed 150
Motor M3 speed 150 M4 speed 150

Result

Think

1.2.9 Smart Traffic Lights

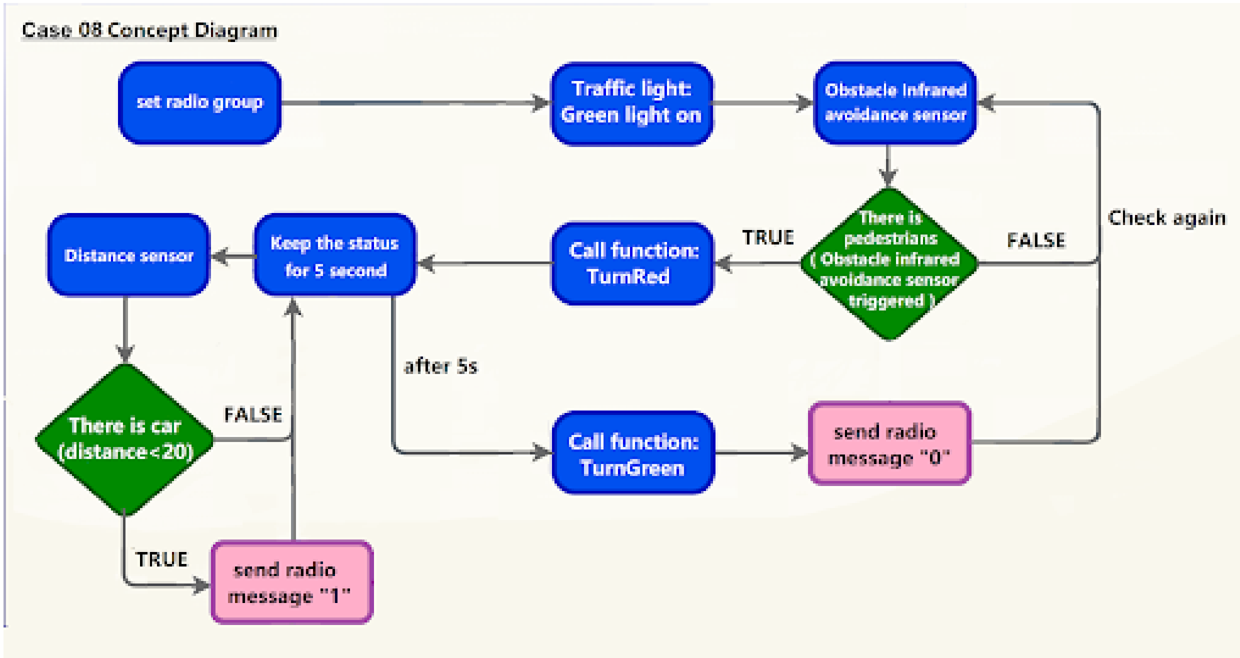


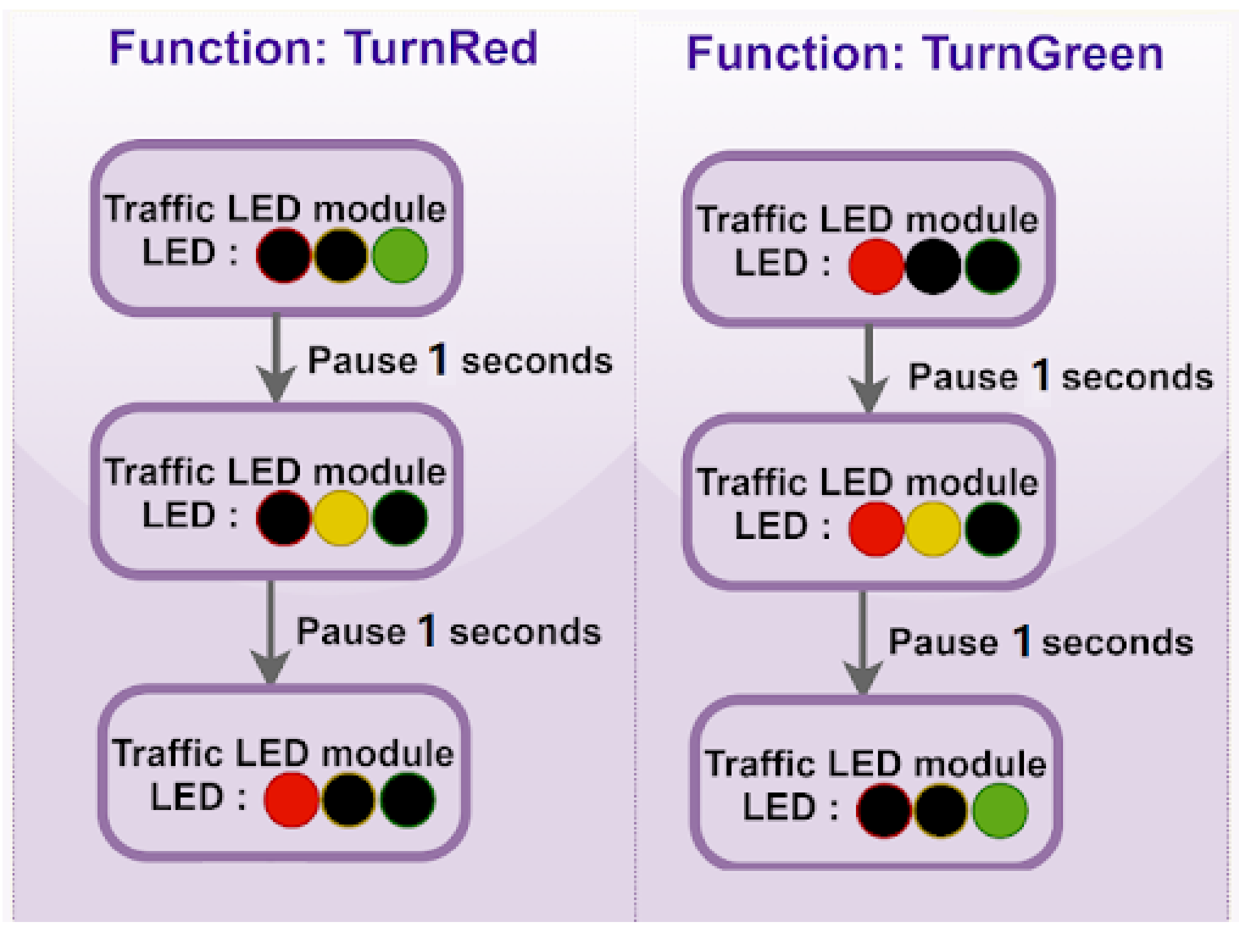
Goal

Background

What is a smart traffic light?

Smart traffic light operation

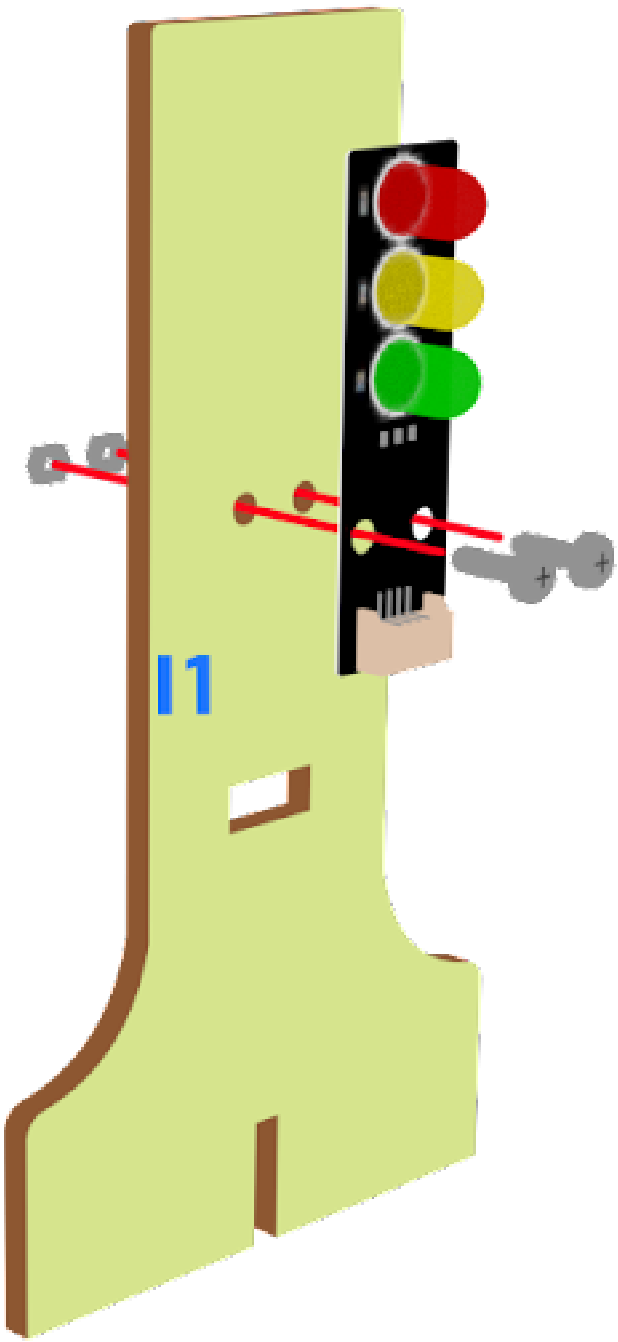




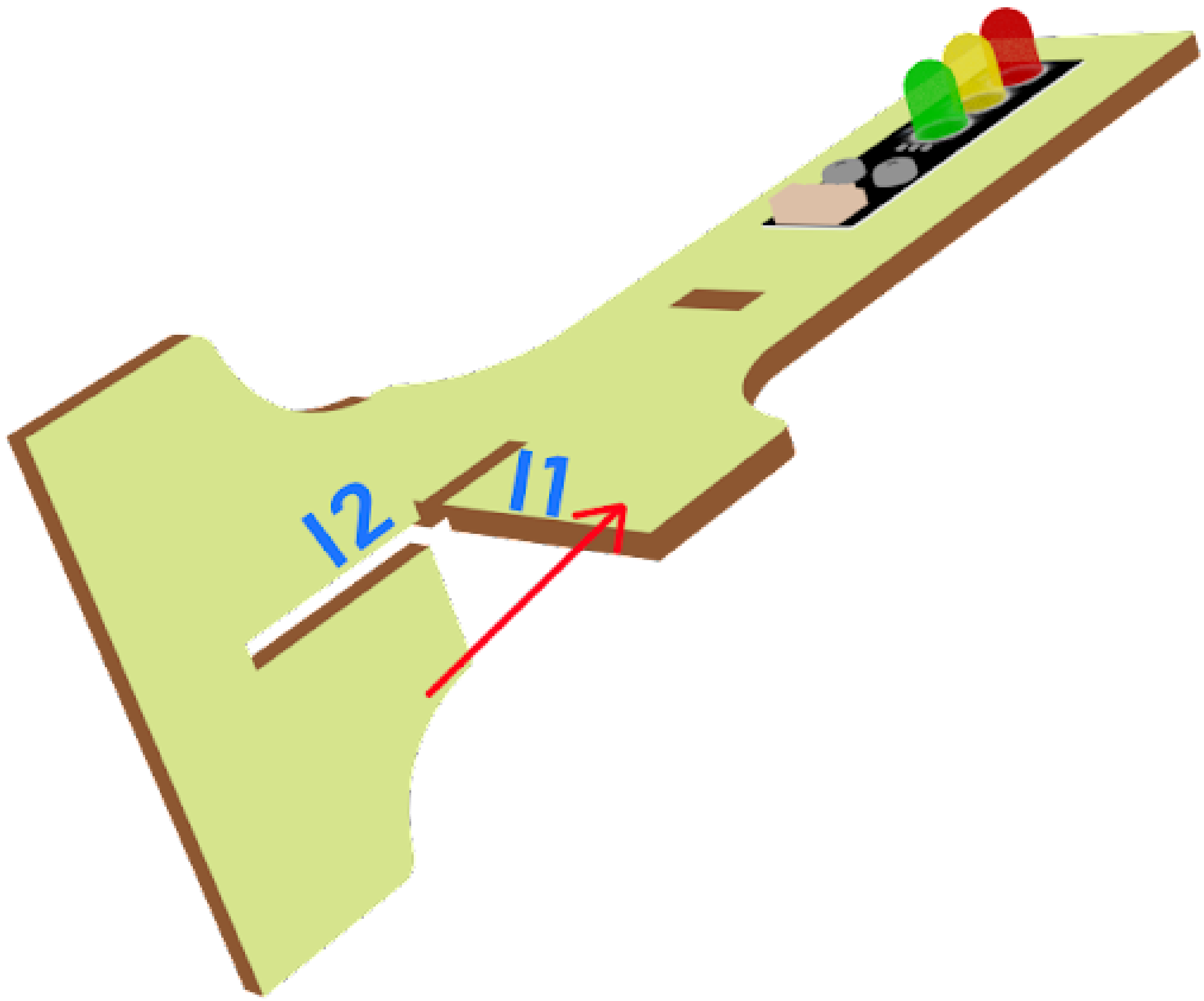
Part List

Assembly step

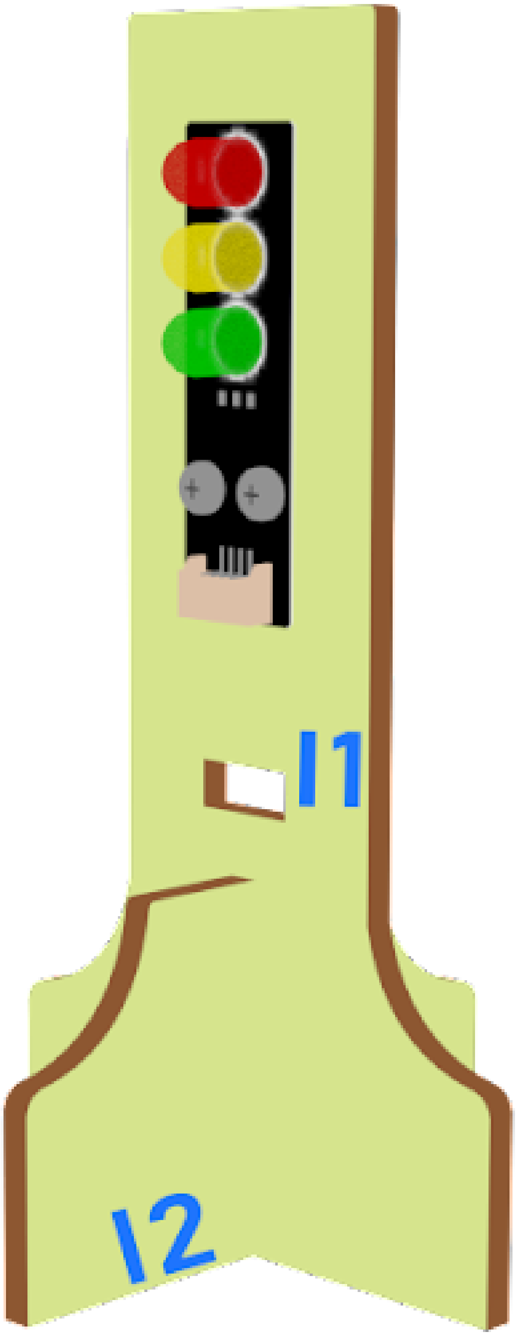
Step 1



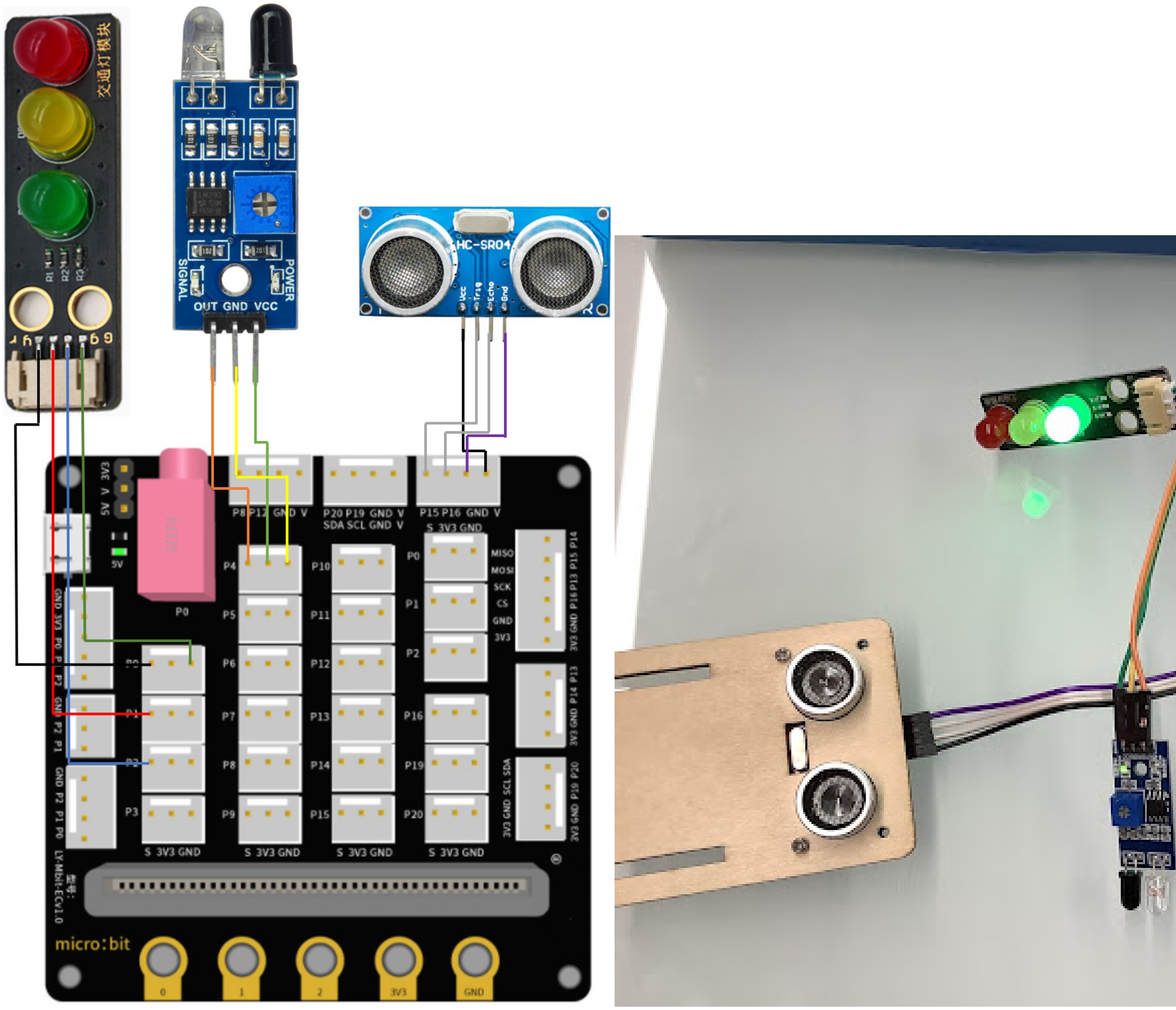
Step 2



Step 3



Hardware connect



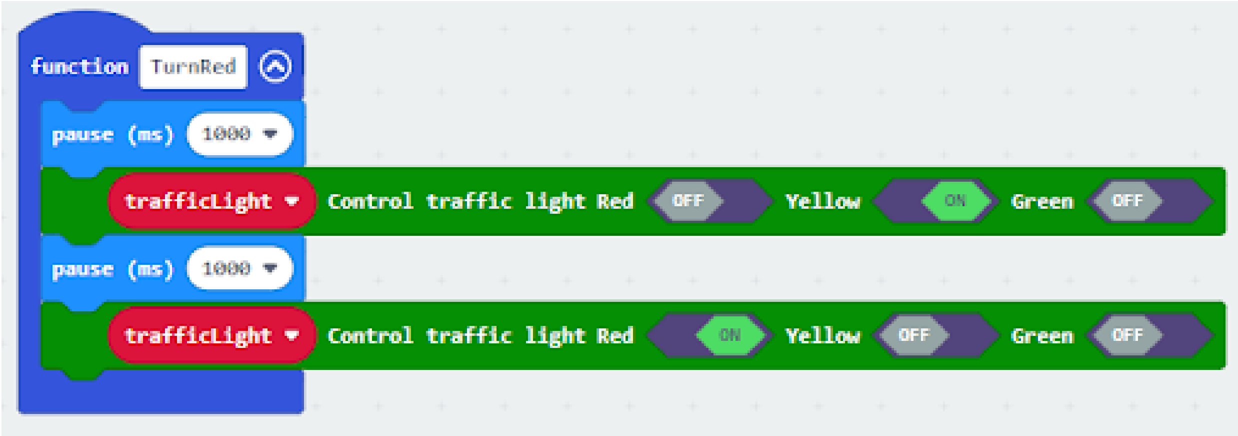
Programming (MakeCode)

Sender

Step 1. Set up a new function (TurnRed)

- Snap pause to wait 1 second
- Control traffic light yellow on
- Snap pause to wait 1 second

- Control traffic light red on

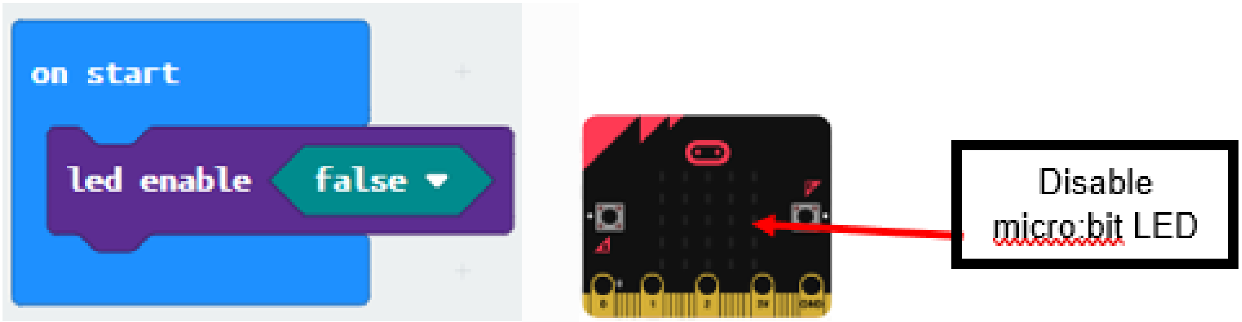


Step 2. Set up a new function (TurnGreen)

- Snap pause to wait 1 second
- Control traffic light yellow on
- Snap pause to wait 1 second
- Control traffic light green on

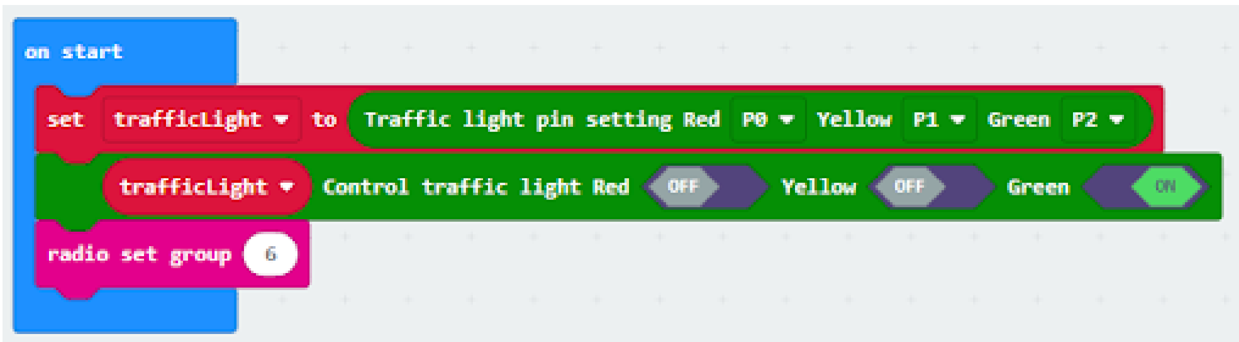
Step 3. Disable micro:bit LED.

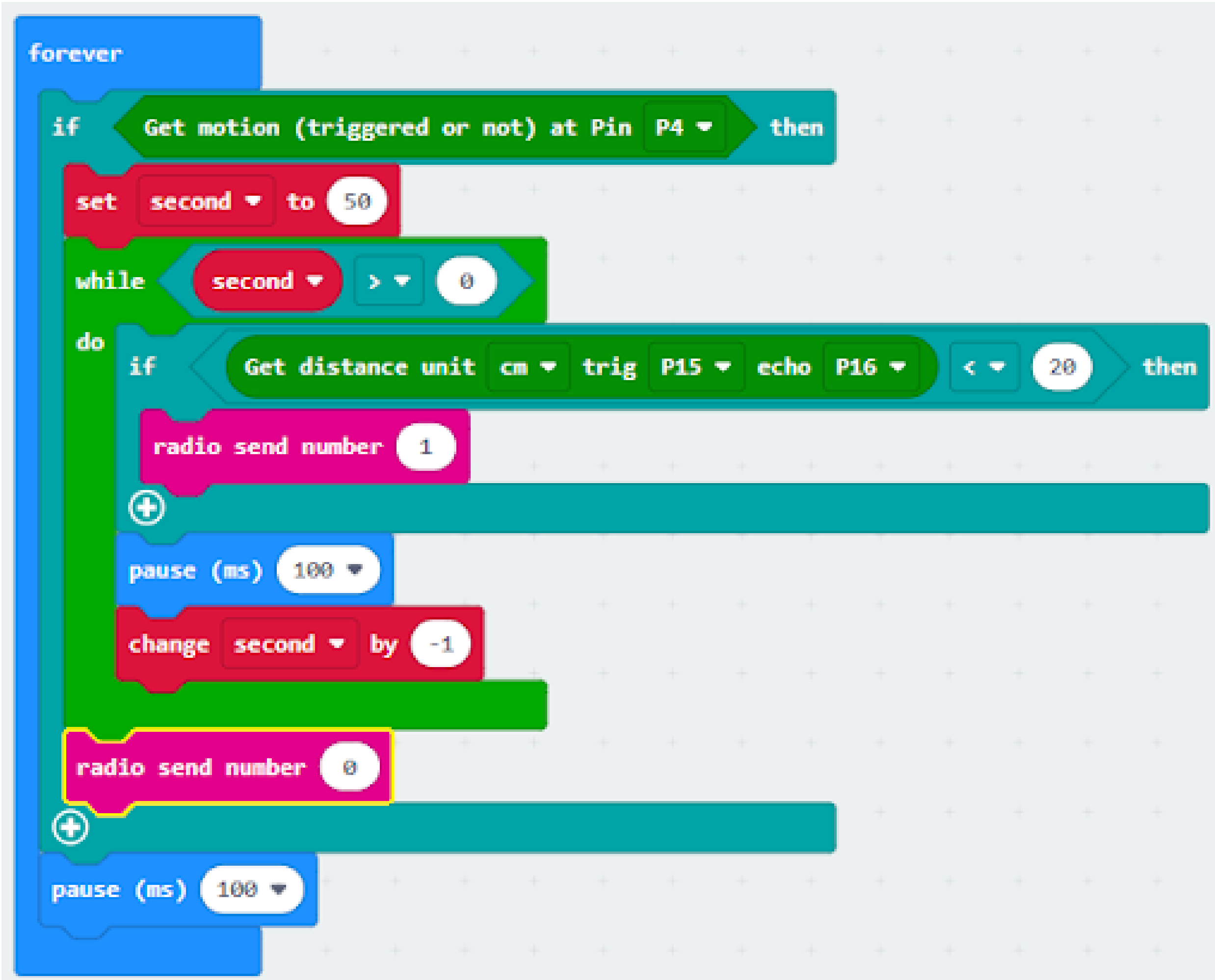
- Snap led enable false to on start
- Note that P3 is used as LED in default setting, LED need to be disable



Step 4. Initialize the program similar as last lesson

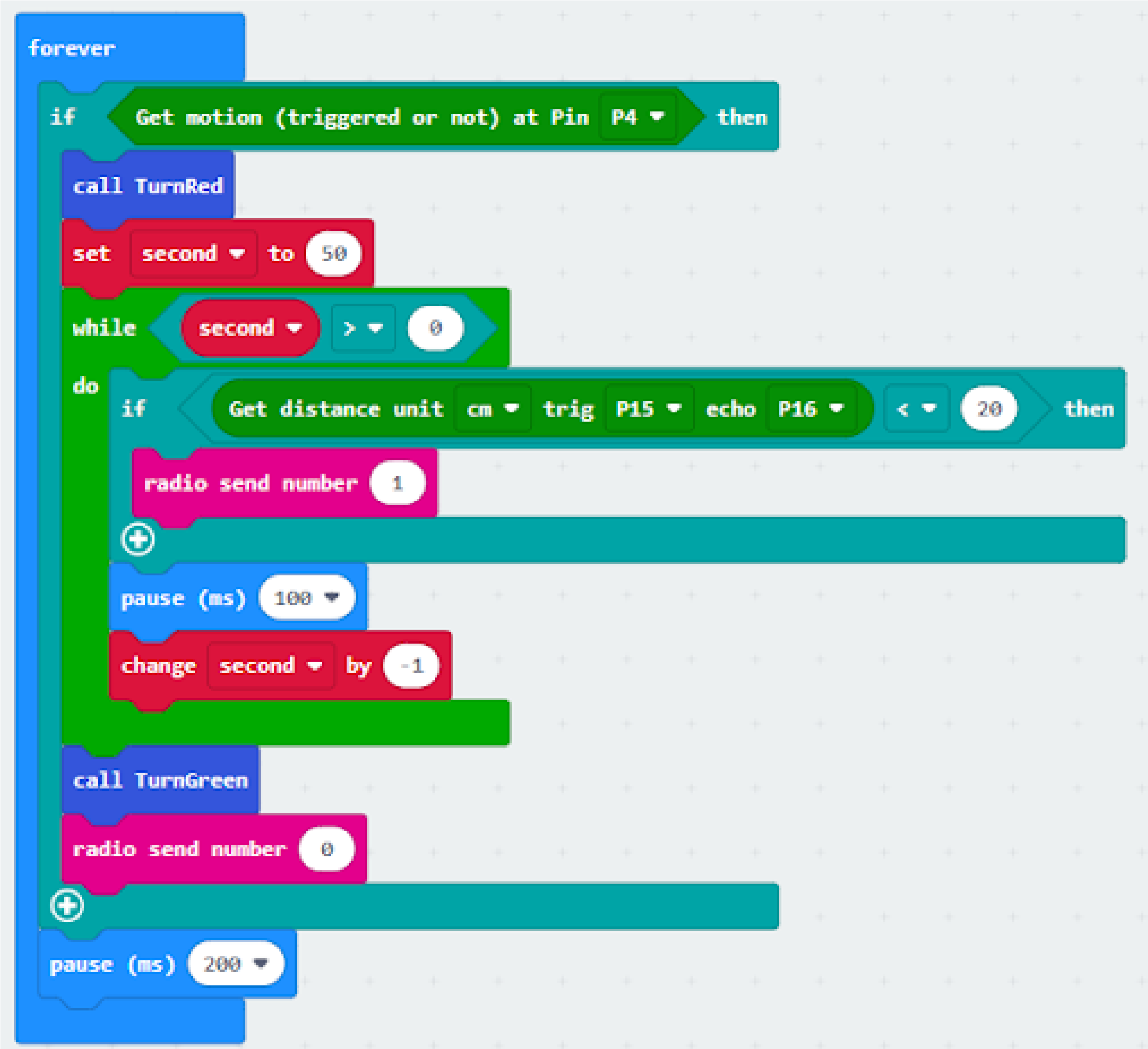
- Drag set variable trafficLight to Traffic light pin setting Red P0 Yellow P1 Green P2 to on start
- Control traffic light green on
- Drag radio set group 6 to on start





Step 5. Call function

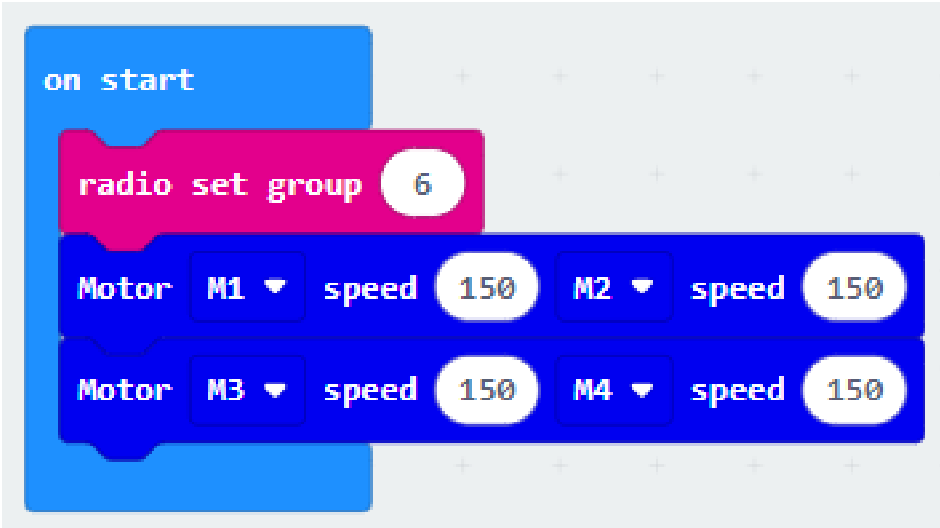
- Snap function TurnRed into if get motion (triggered or not) at pin P4 case
- Drag function TurnGreen after the while loop



Receiver

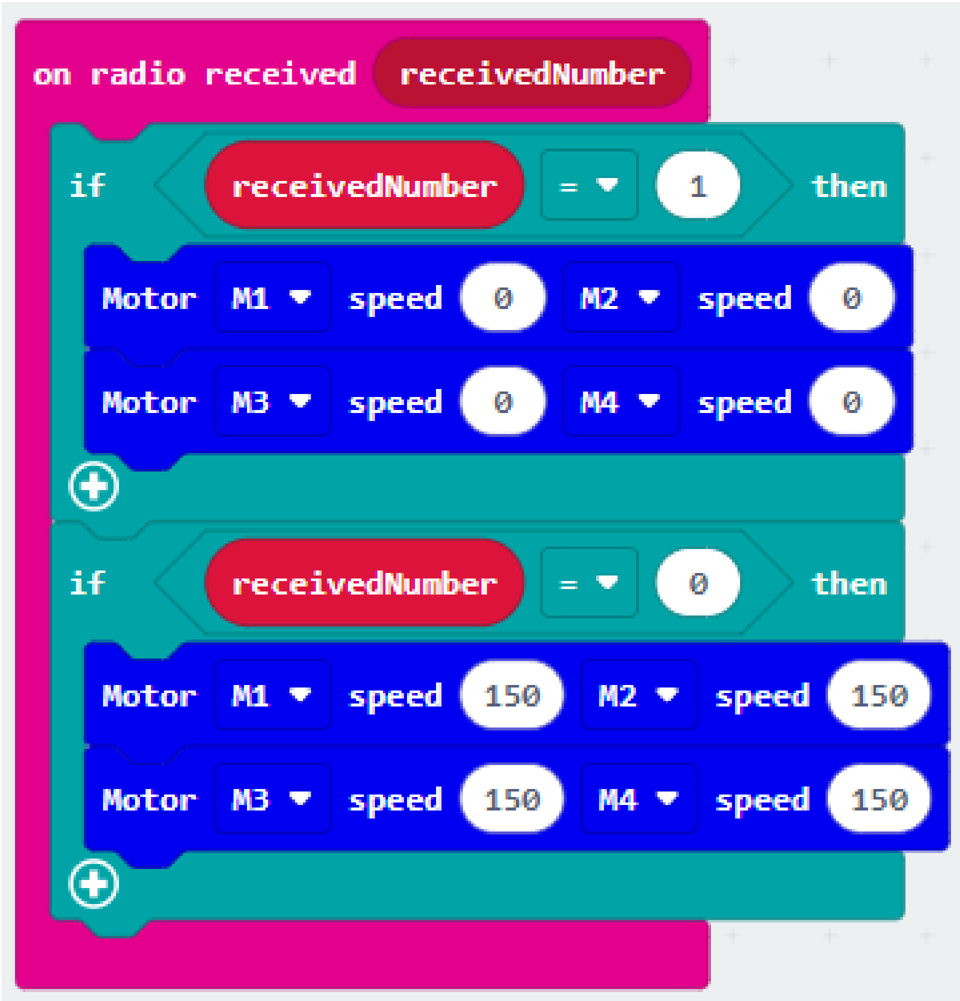
Step 1. Set radio set group at start position

- Drag radio set group 6 to on start
- Initially, the car moves forward by default



Step 2. Control car by receiving different number

- Snap if statement into on radio received receivedNumber
- Set receivedNumber =1 and make the car stop
- Set receivedNumber=0 and make the car move forward


on radio received receivedNumber
if receivedNumber = 1 then
Motor M1 speed 0 M2 speed 0
Motor M3 speed 0 M4 speed 0
if receivedNumber = 0 then
Motor M1 speed 150 M2 speed 150
Motor M3 speed 150 M4 speed 150

Result

Think

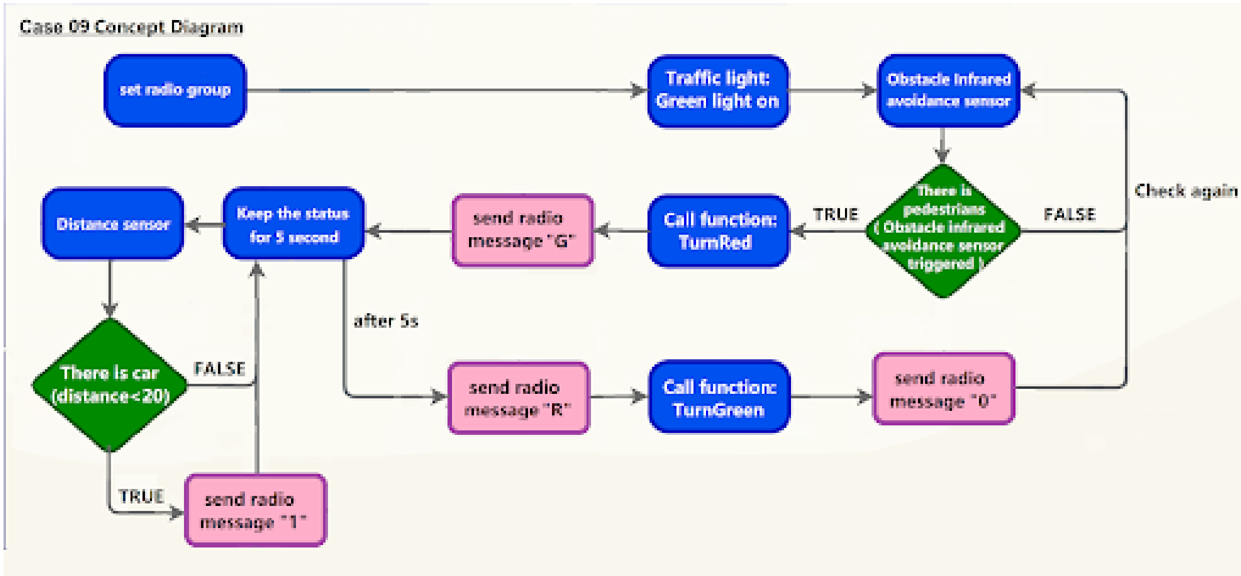
1.2.10 Smart Pedestrian Lights 2

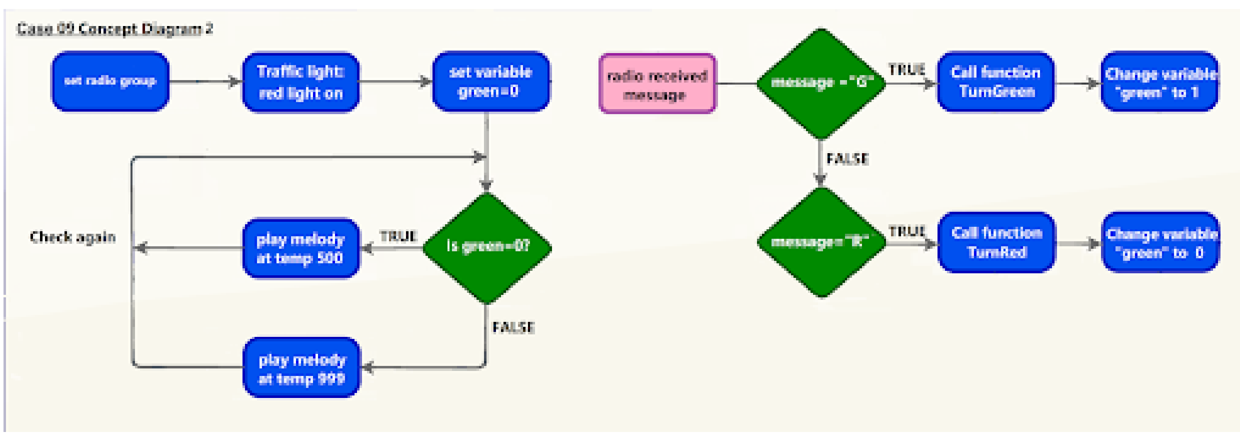
Goal

Background

What is a smart pedestrian light?

Smart traffic light operation

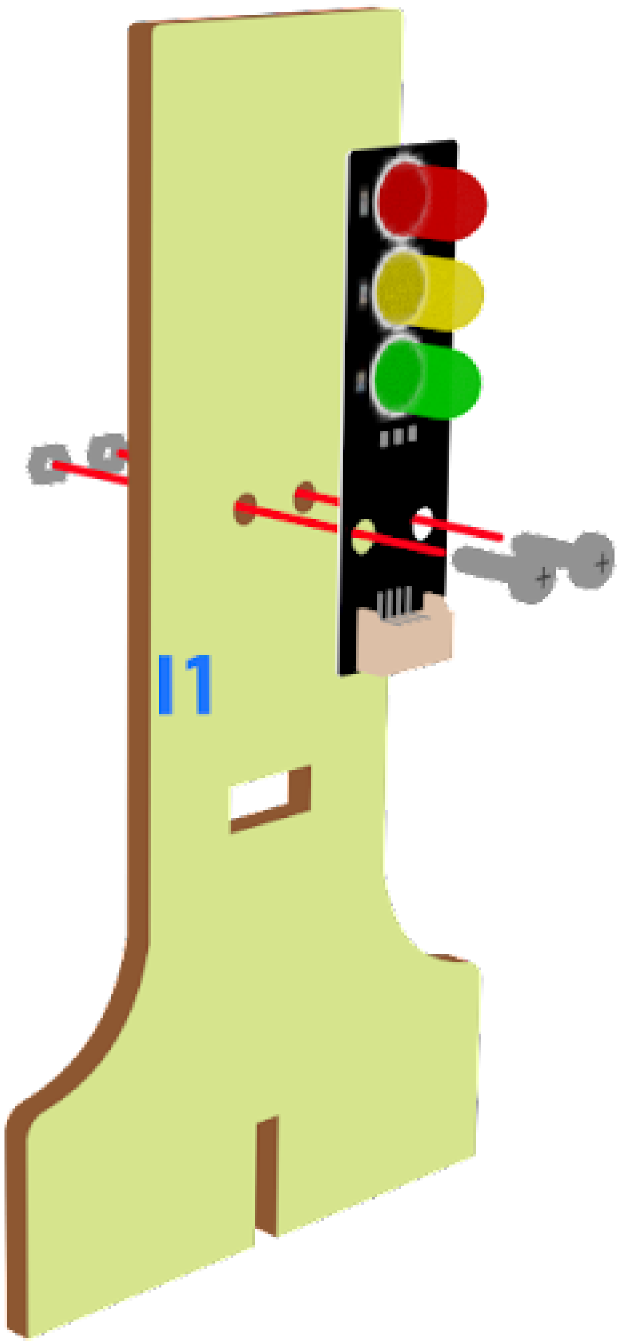




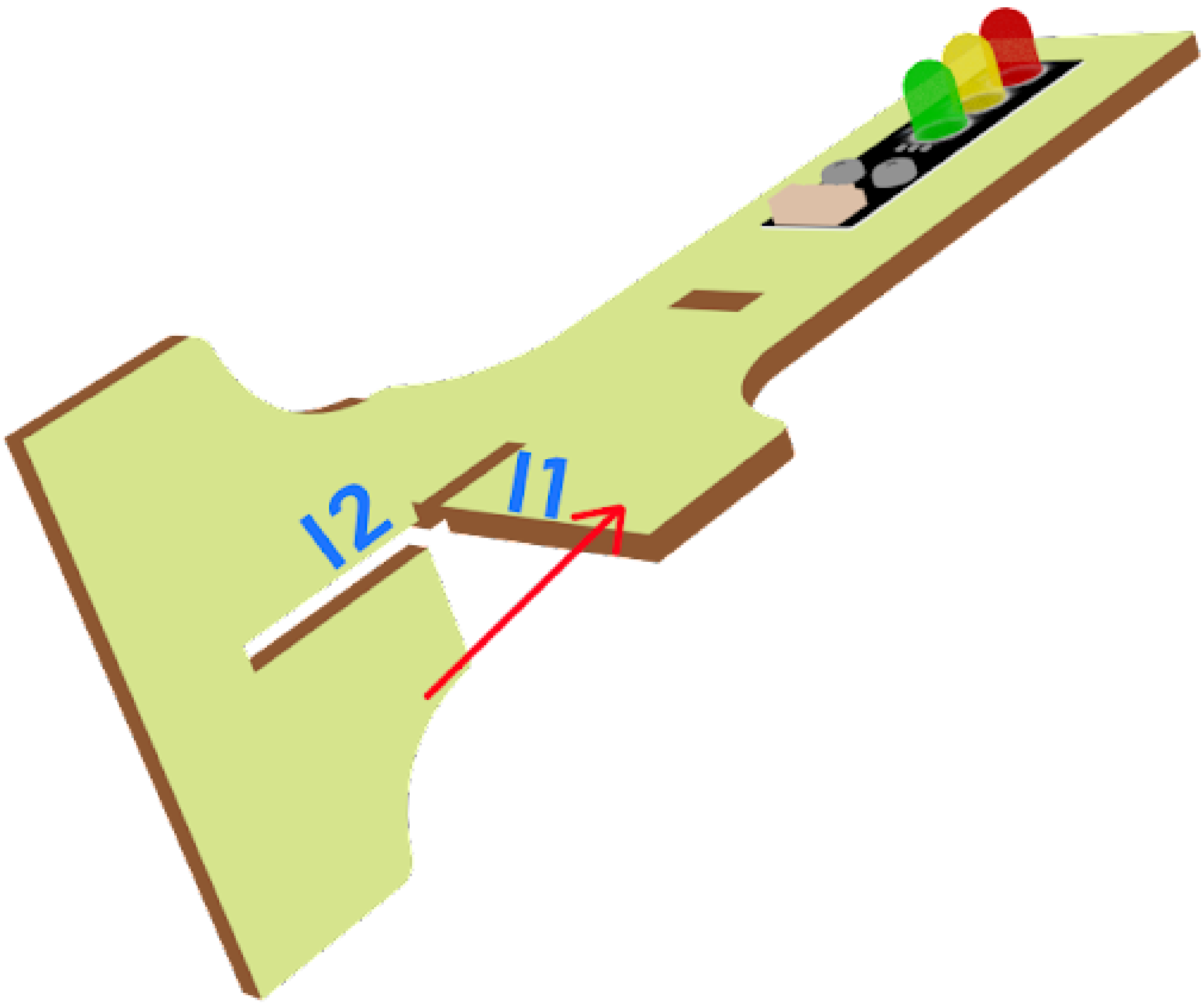
Part List

Assembly step

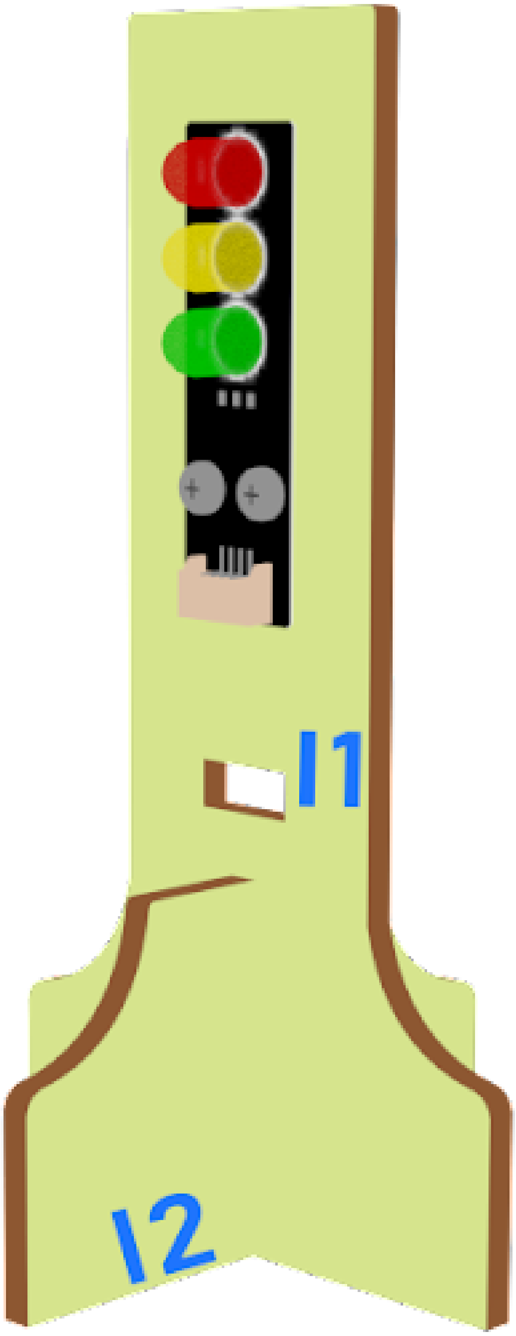
Step 1



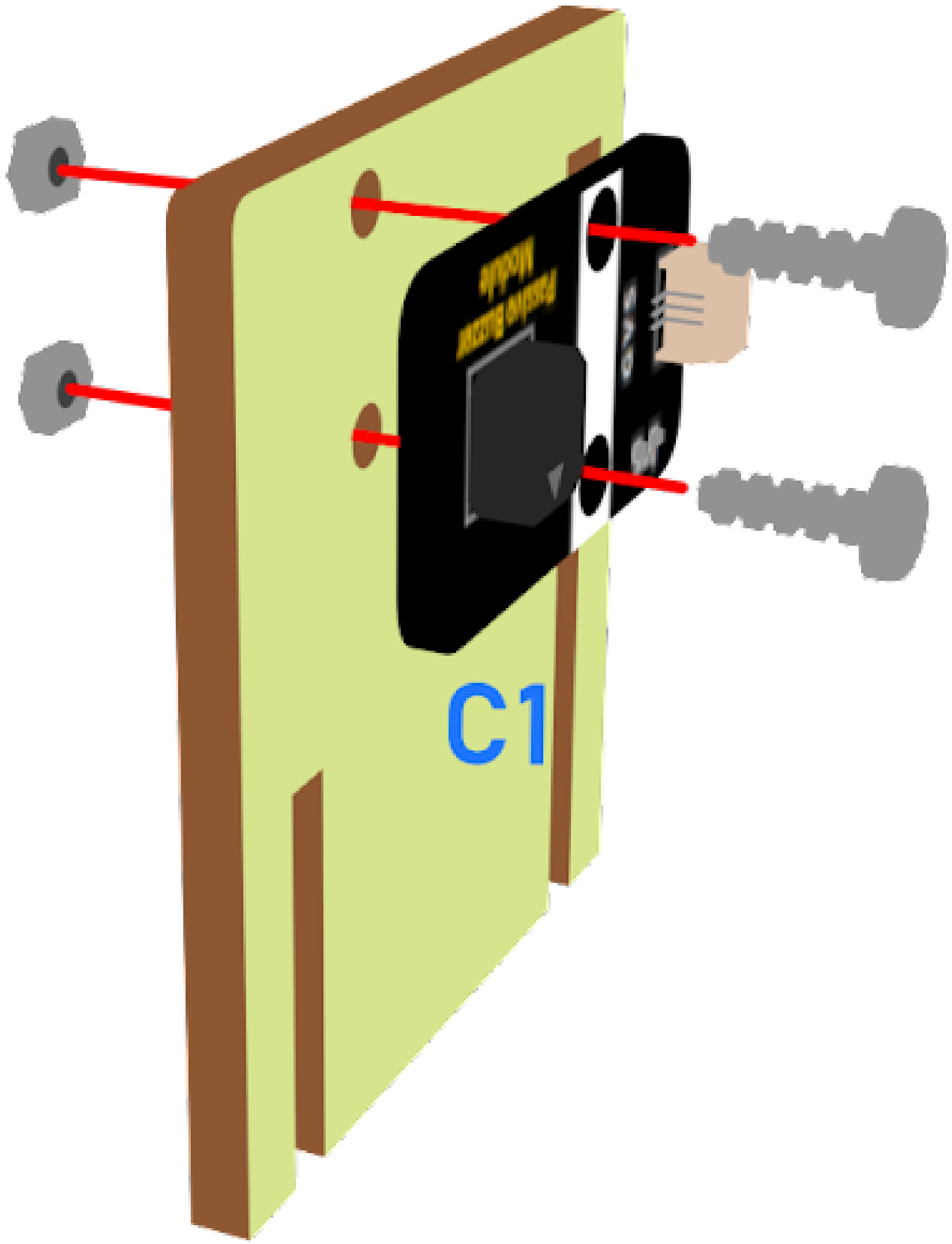
Step 2



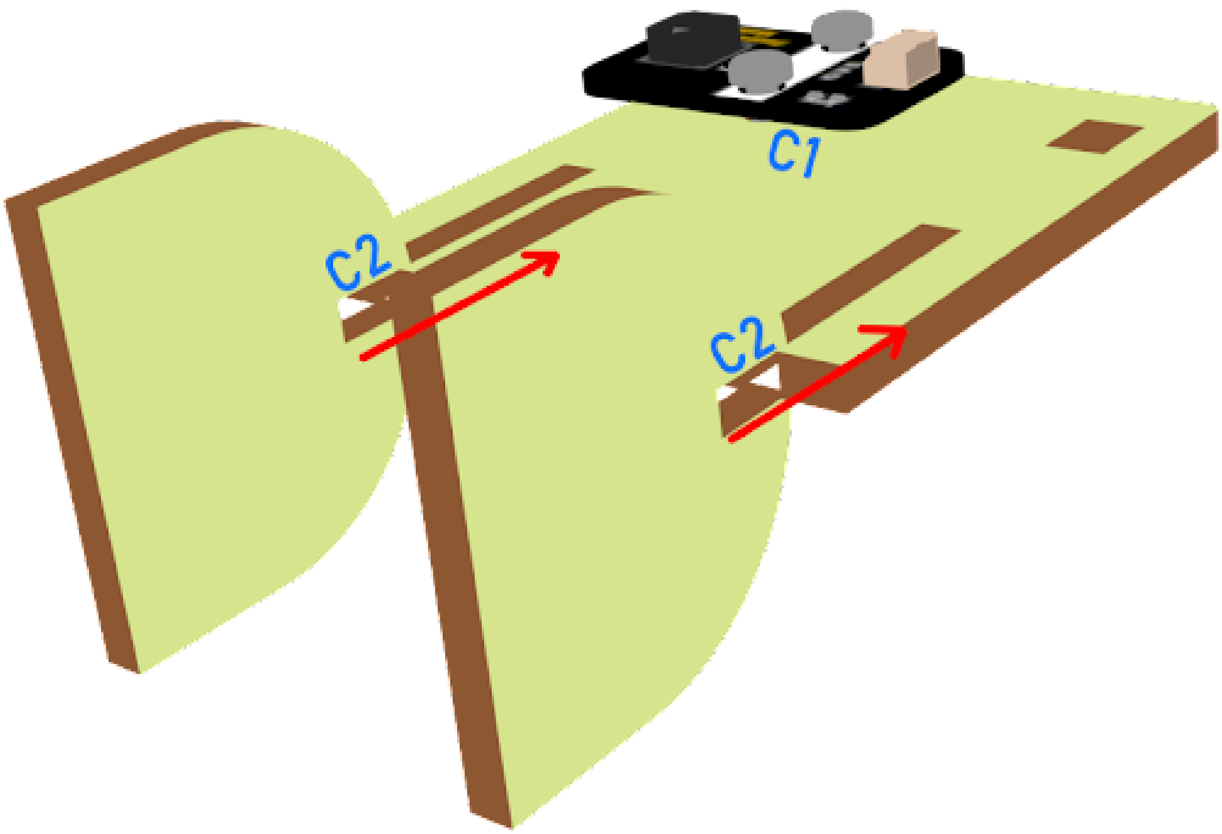
Step 3



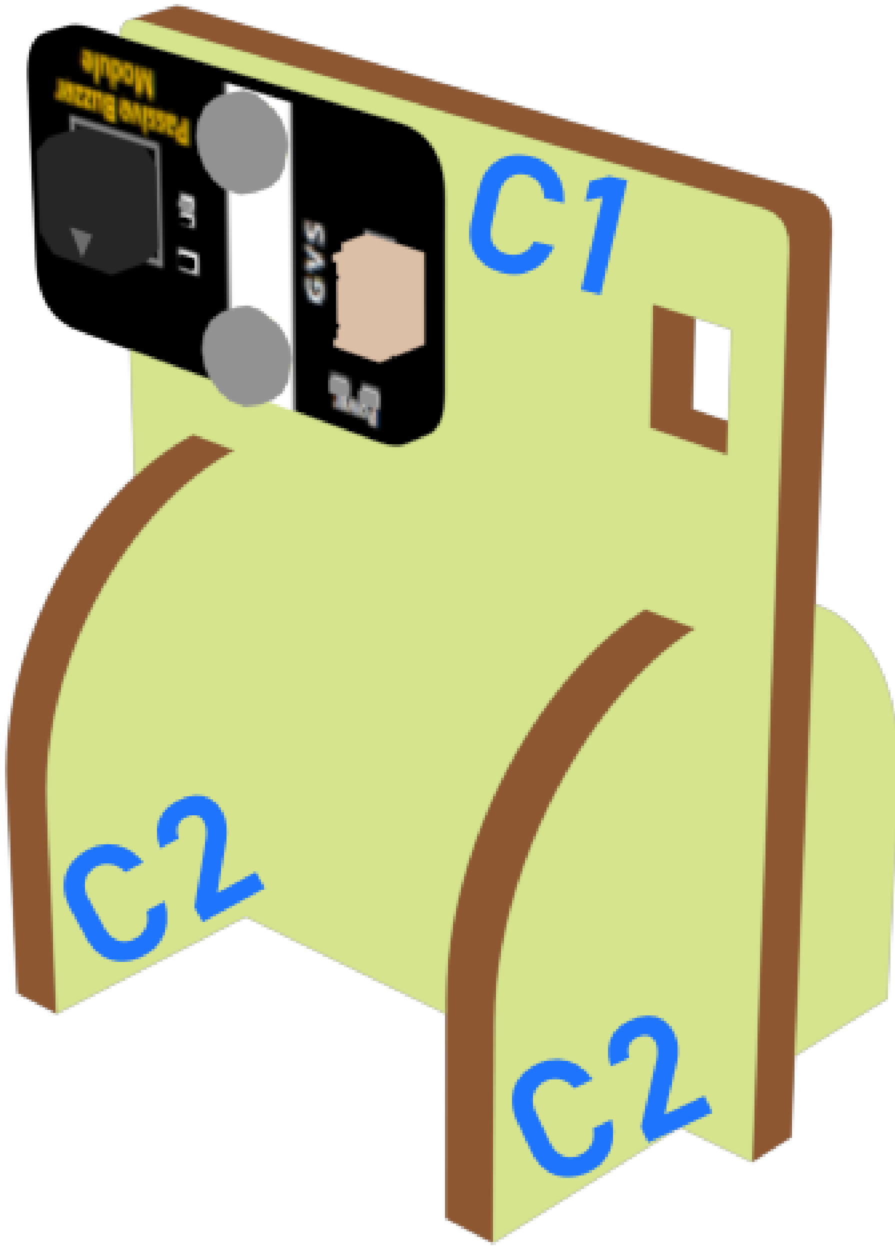
Step 4



Step 5



Step 6



Hardware connect

Programming (MakeCode)

Traffic light 1

Step 1. Set up a new function (TurnRed)

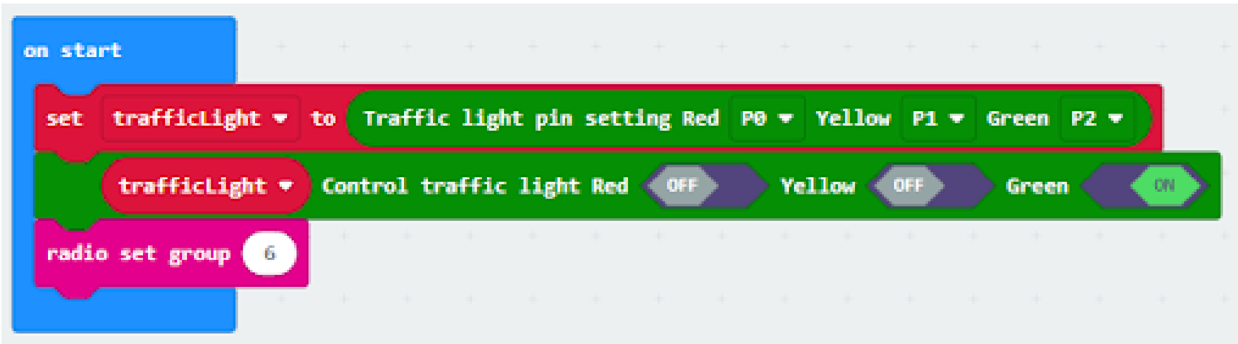
- Snap pause to wait 1 second
- Control traffic light yellow on
- Snap pause to wait 1 second
- Control traffic light red on

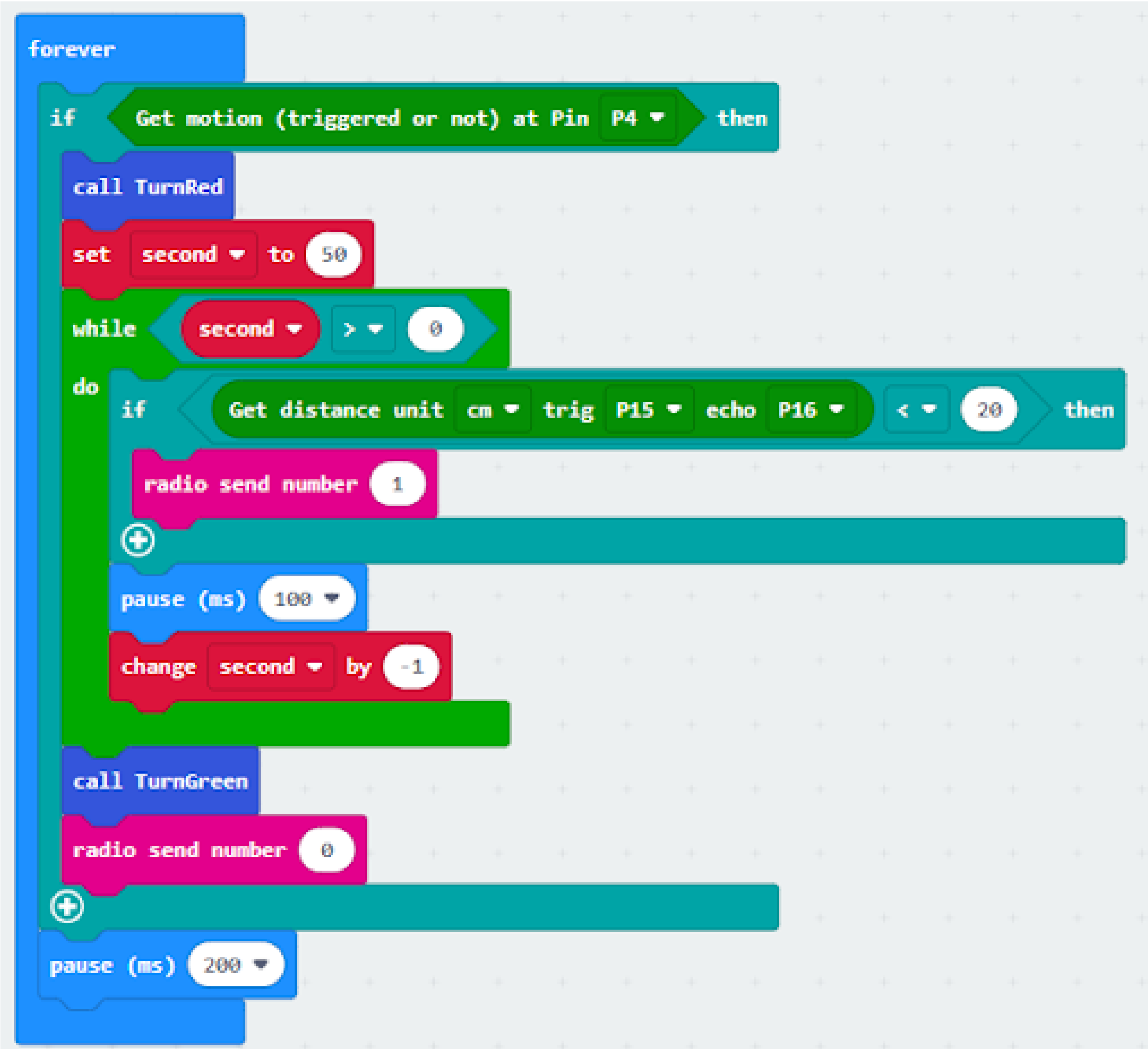
Step 2. Set up a new function (TurnGreen)

- Snap pause to wait 1 second
- Control traffic light yellow on
- Snap pause to wait 1 second
- Control traffic light green on

Step 3. Initialize the program

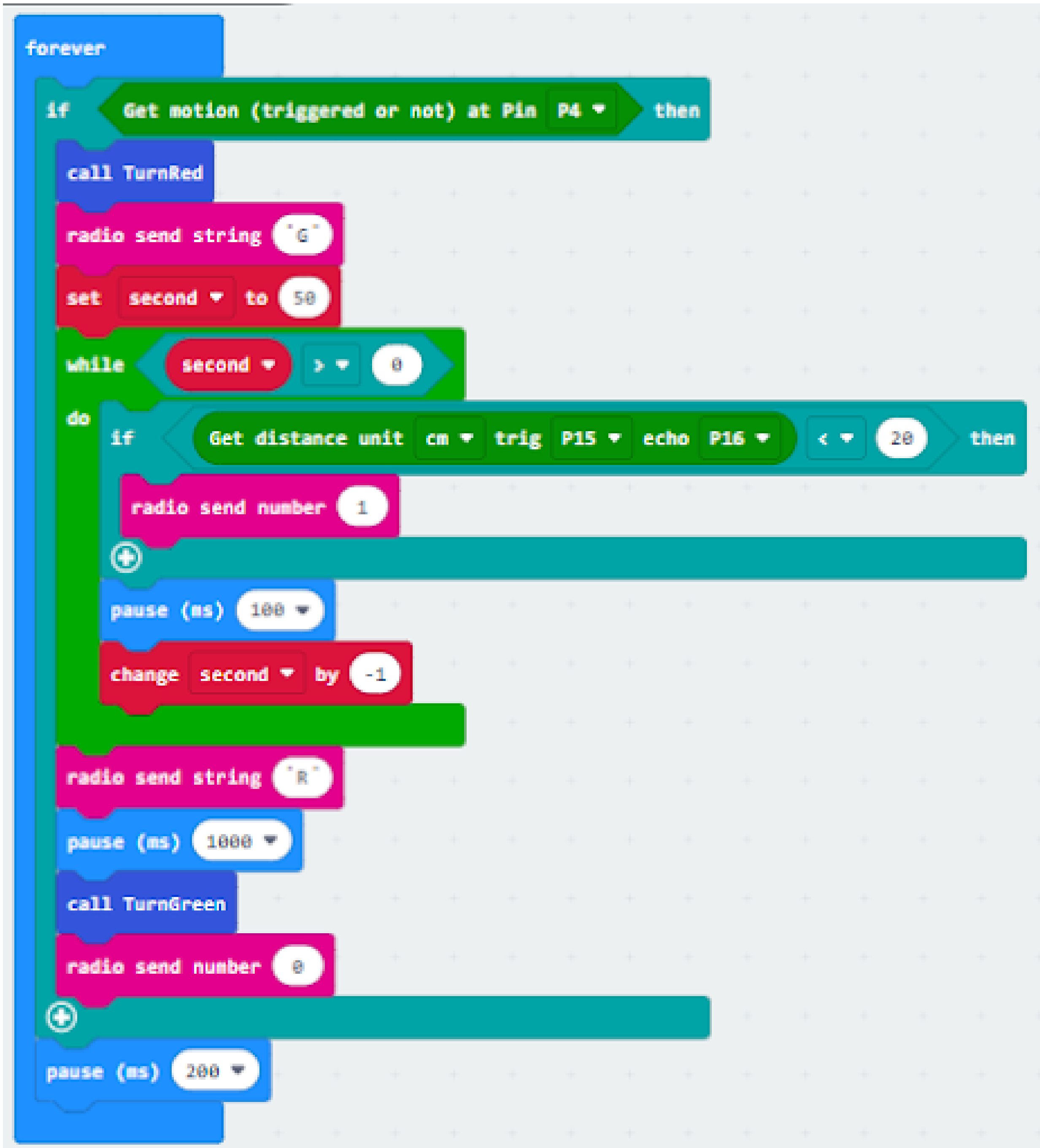
- Drag set variable trafficLight to Traffic light pin setting Red P0 Yellow P1 Green P2 to on start
- Control traffic light green on
- Drag radio set group 6 to on start
- In forever, snap function TurnRed into if get motion (triggered or not) at pin P4 case
- Drag function TurnGreen after the while loop





Step 4. Control traffic light 2 by sending radio string

- Drag radio send string “R” before TurnGreen
- Drag radio send string “G” after TurnRed

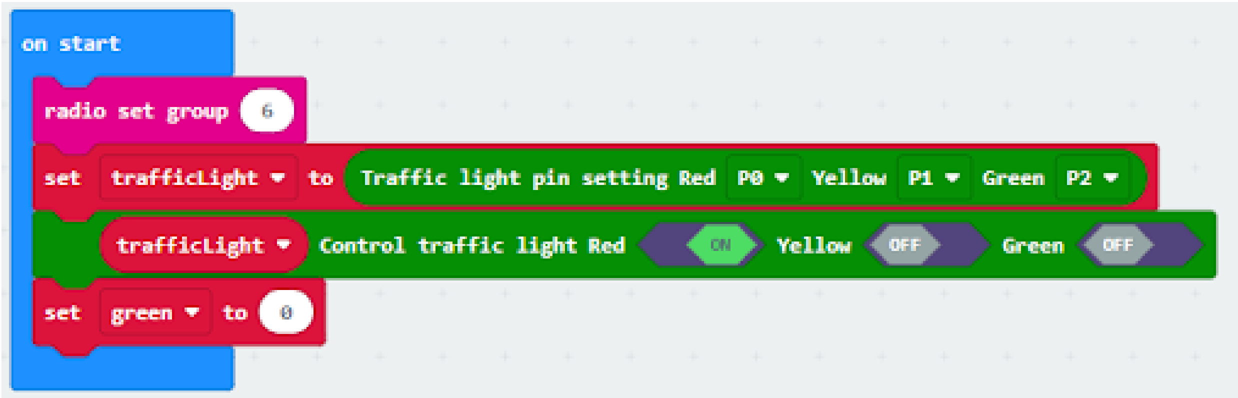

forever
if
Get motion (triggered or not) at Pin P4
then
call TurnRed
radio send string "G"
set second to 50
while second > 0
do
if
Get distance unit cm trig P15 echo P16
< 20
then
radio send number 1
pause (ms) 100
change second by -1
radio send string "R"
pause (ms) 1000
call TurnGreen
radio send number 0
pause (ms) 200

Traffic light 2

Step 1. Set up new functions

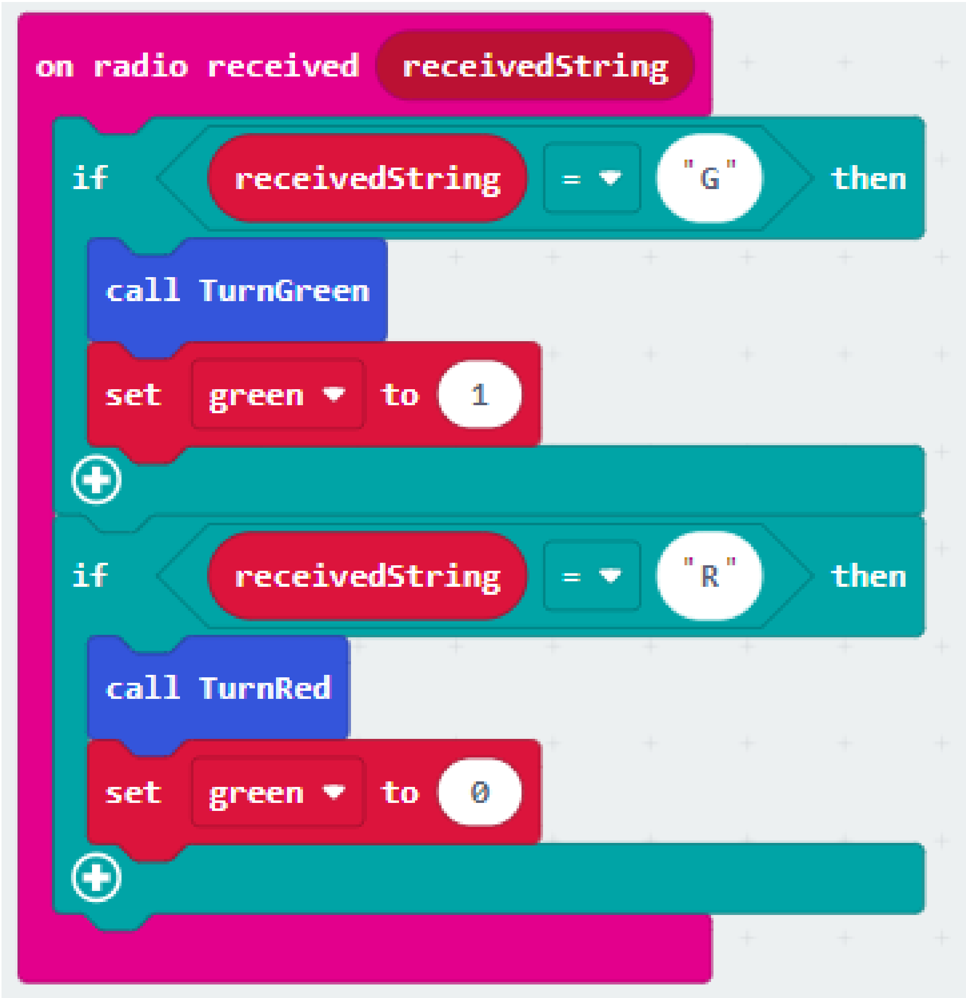
Step 2. Initialize the program

- Drag set variable trafficLight to Traffic light pin setting Red P0 Yellow P1 Green P2 to on start
- Drag radio set group 6 to on start
- Control traffic light green on
- Set a variable green=0



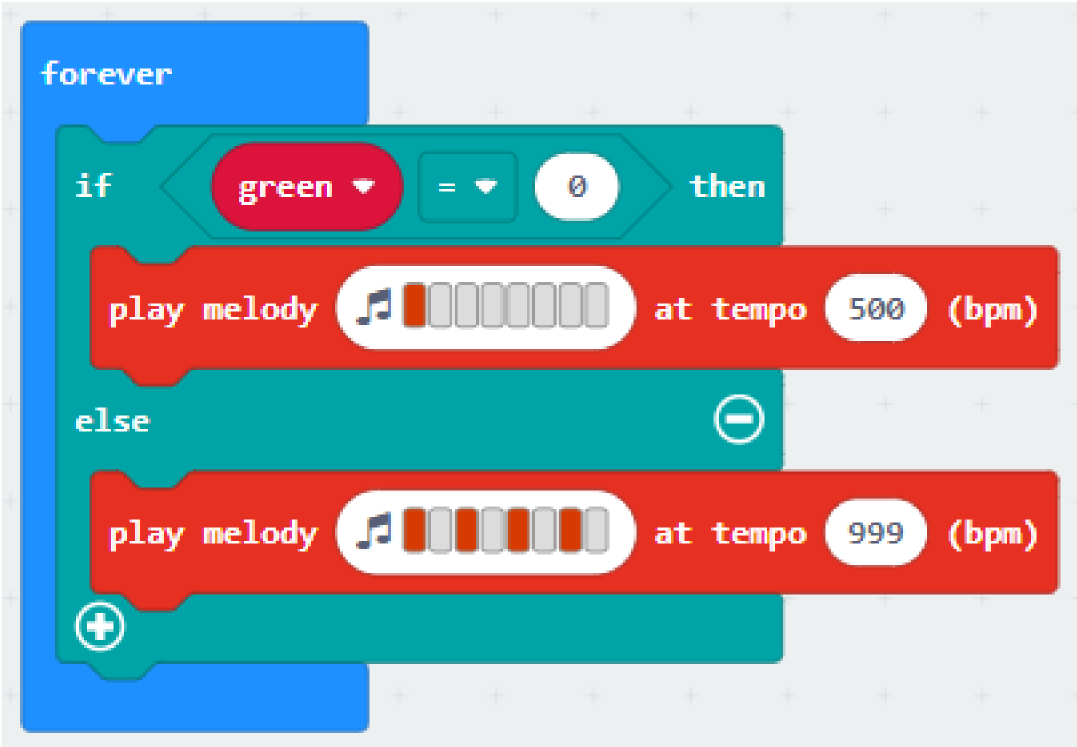
Step 3. Control traffic light by receiving different number

- Snap if statement into on radio received receivedString
- Set receivedString =“R” and call TurnRed
- Set receivedString =“G” and call TurnGreen
- Change variable green depend on the light



Step 4. Play sound effect depend on the light status

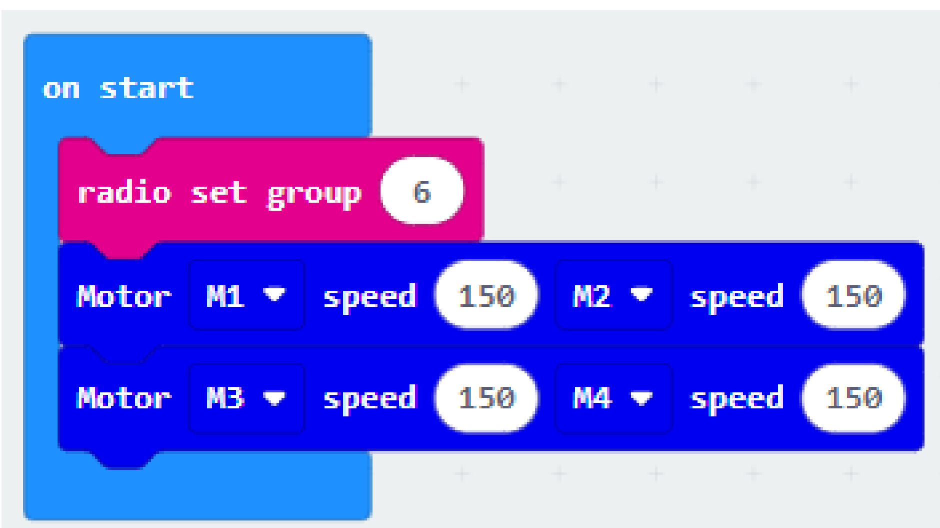
- Snap if statement into forever
- Play melody with different tempo



Receiver

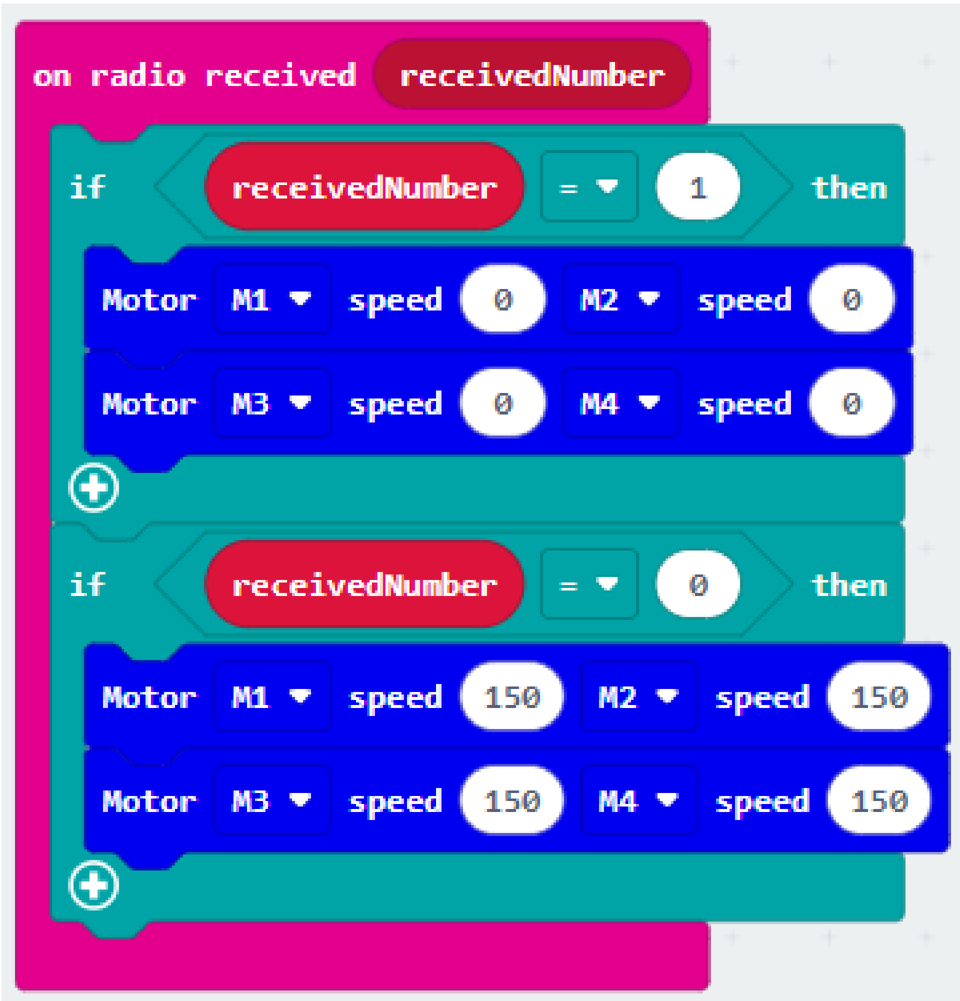
Step 1. Set radio set group at start position

- Drag radio set group 6 to on start
- Initially, the car moves forward by default



Step 2. Control car by receiving different number

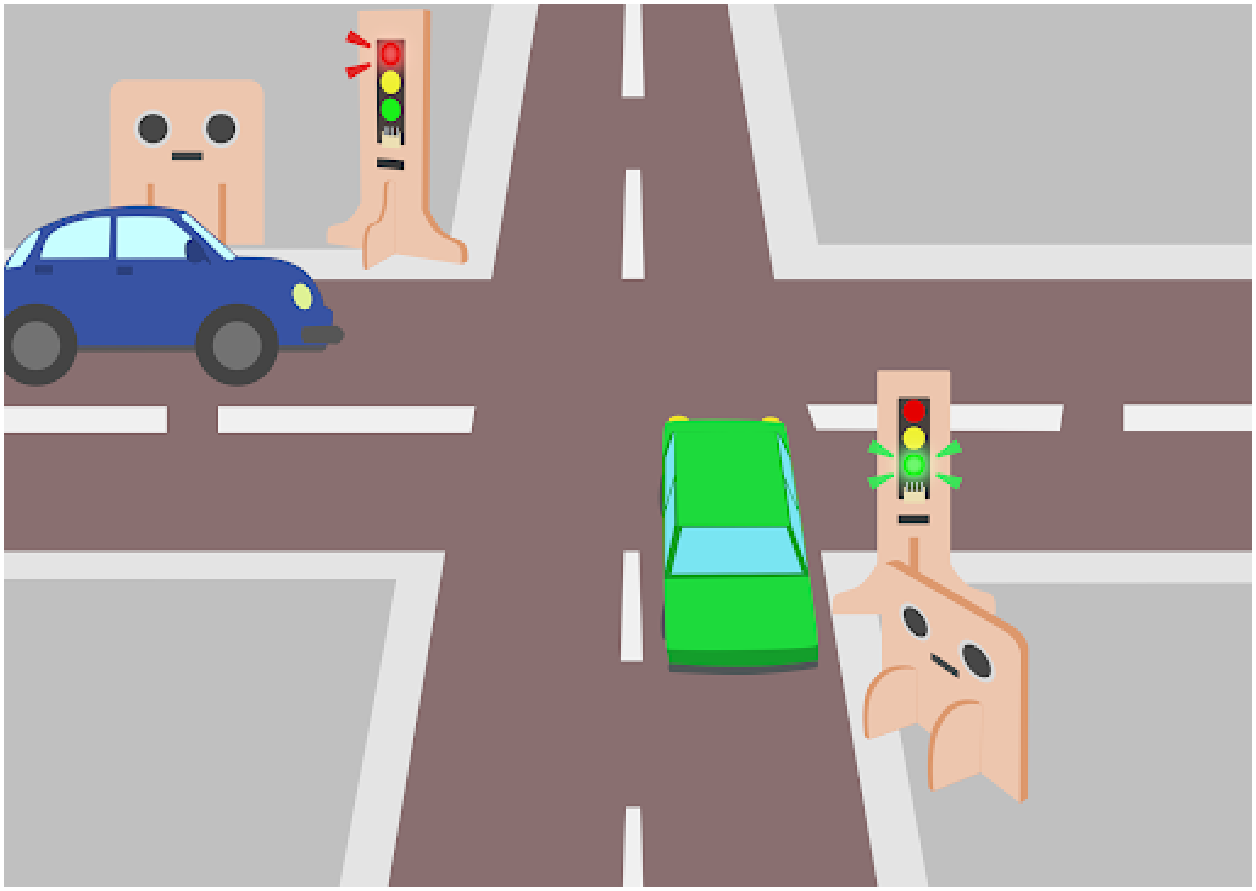
- Snap if statement into on radio received receivedNumber
- Set receivedNumber =1 and make the car stop
- Set receivedNumber=0 and make the car move forward



Result

Think

1.2.11 Intersection

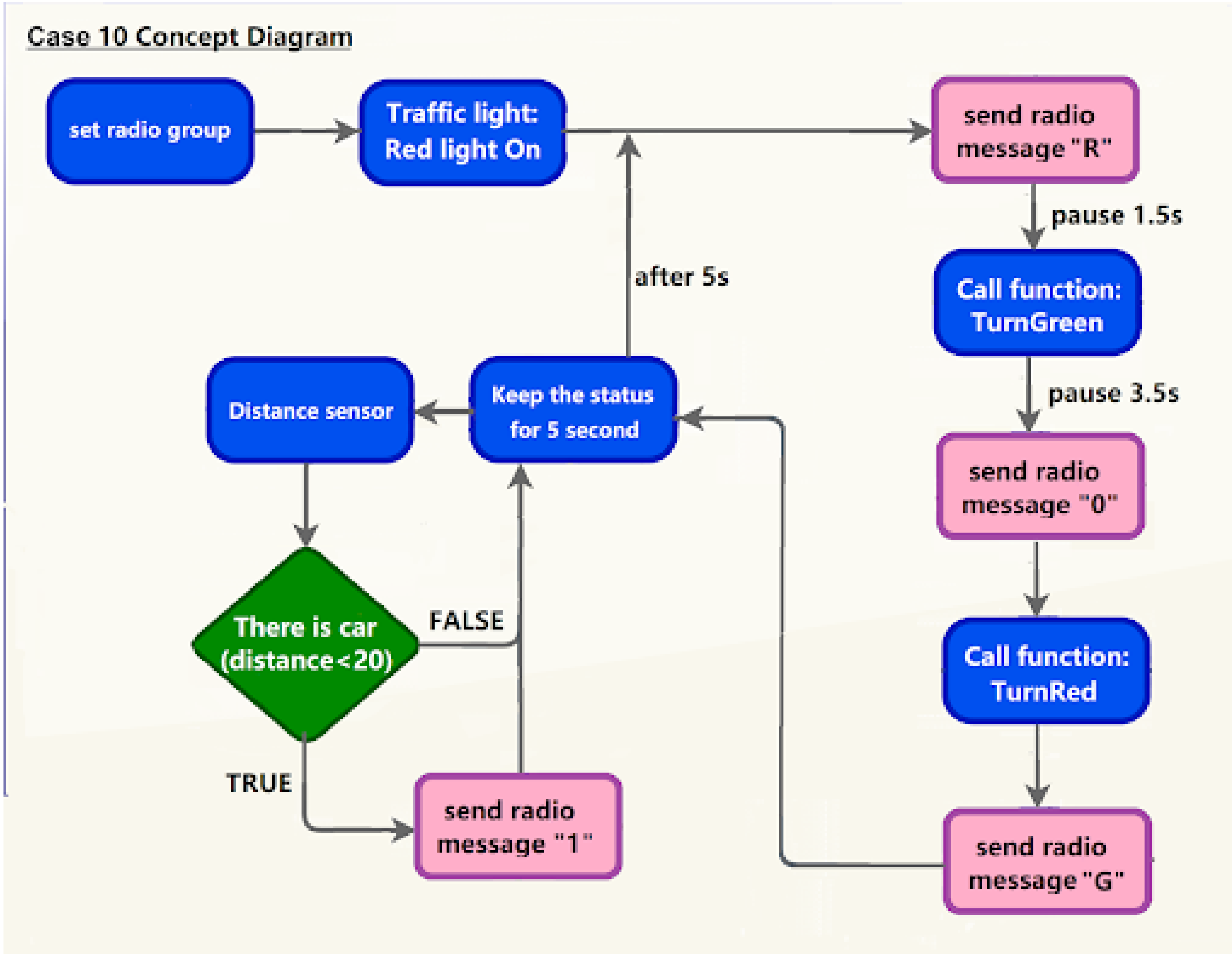


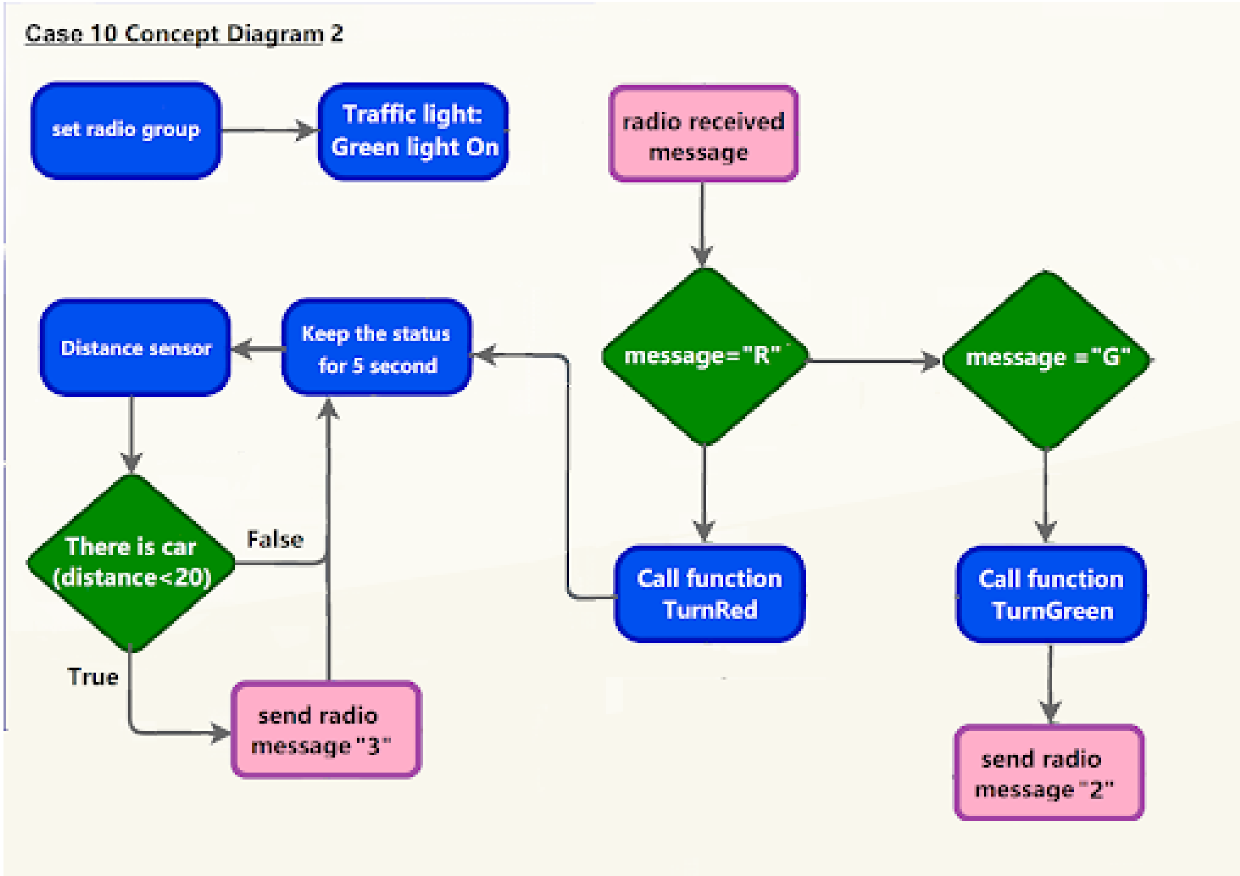
Goal

Background

What is a smart traffic light?

Smart traffic light operation

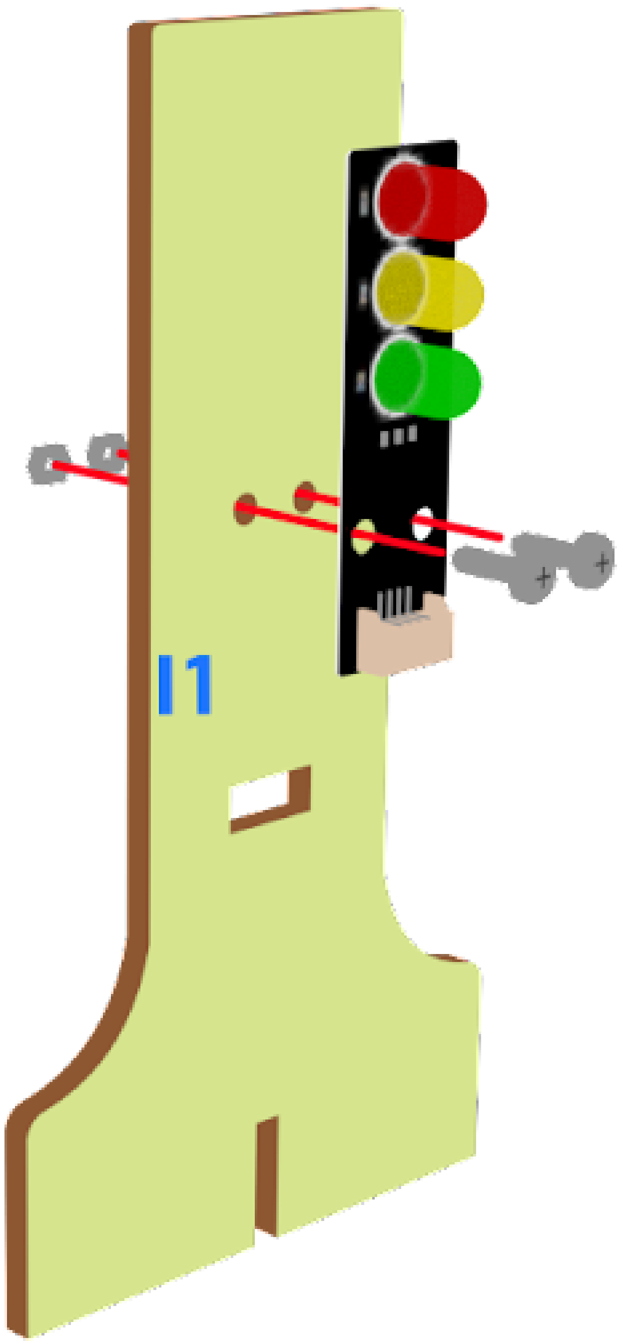




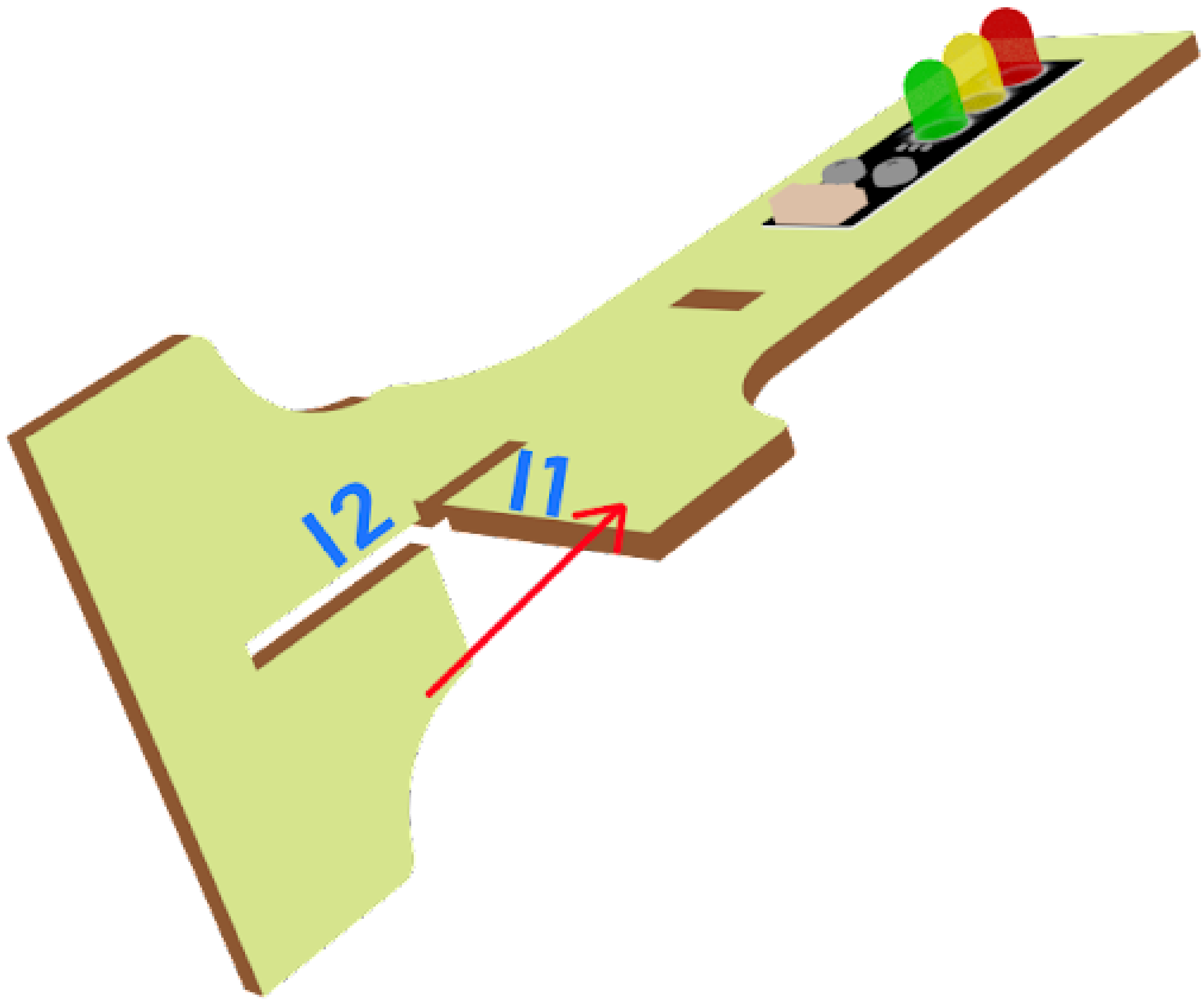
Part List

Assembly step

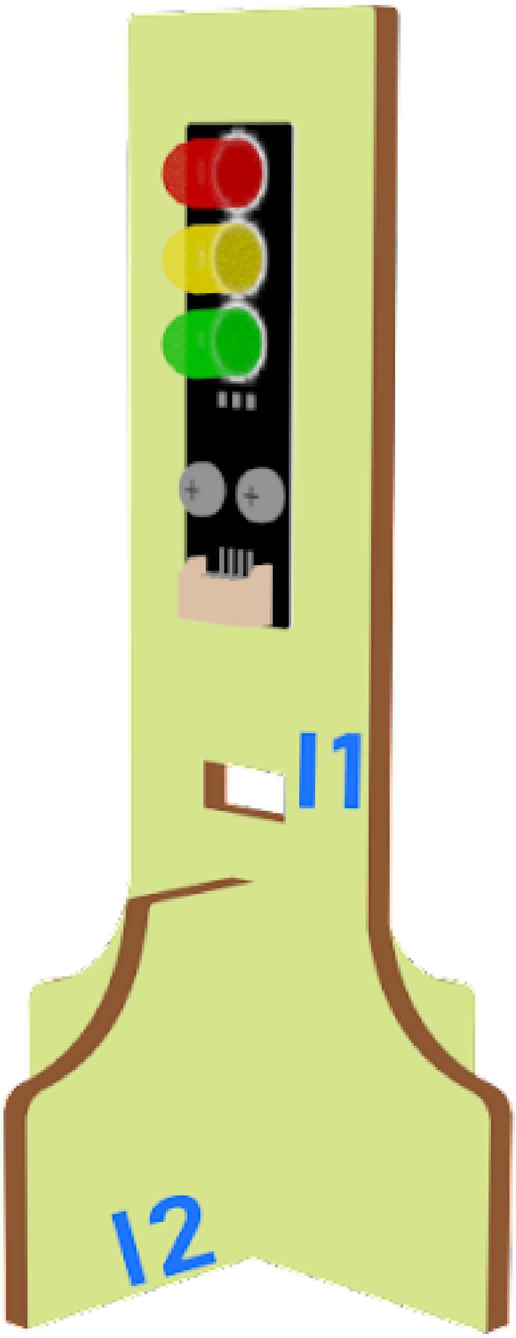
Step 1



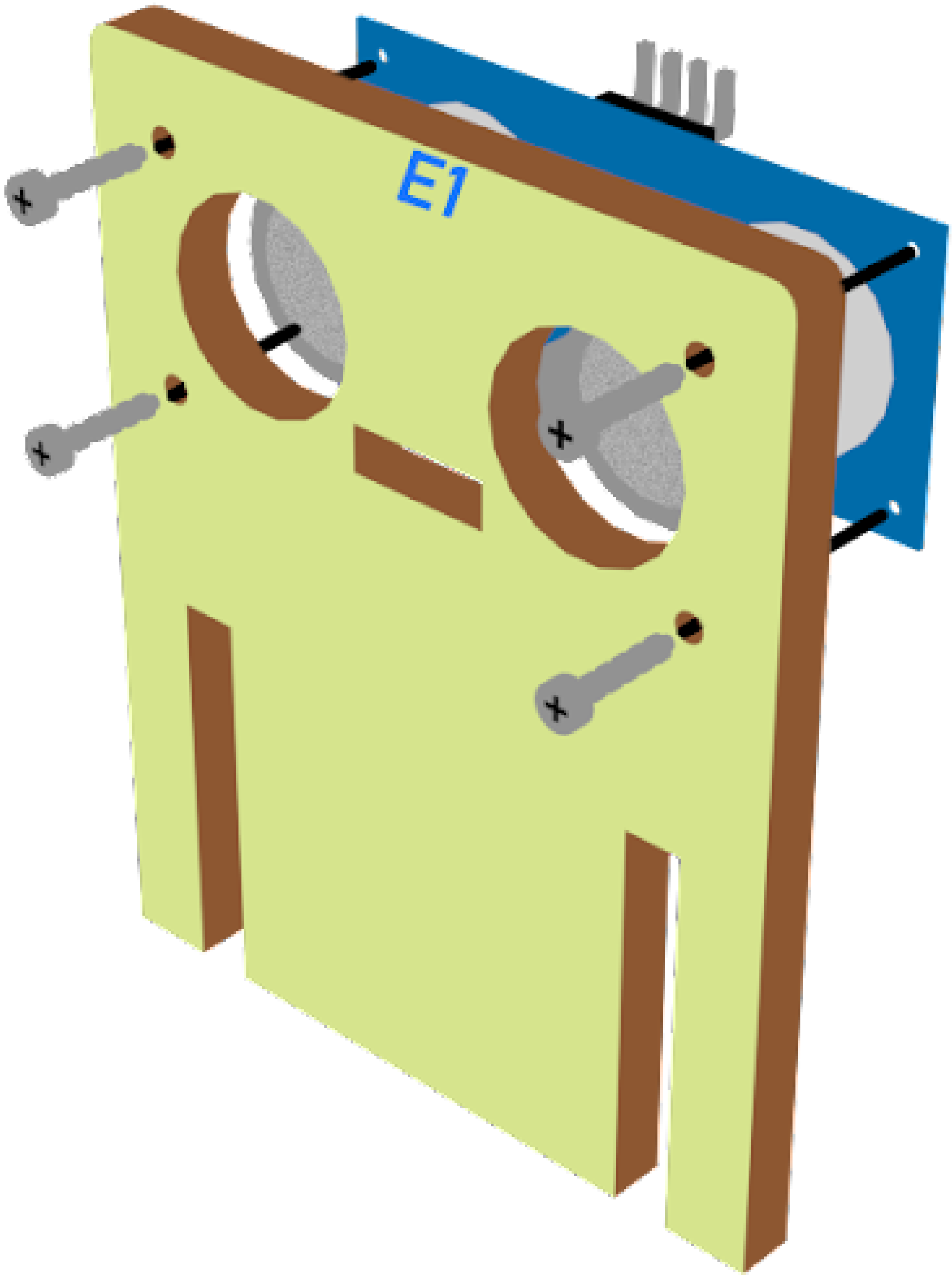
Step 2



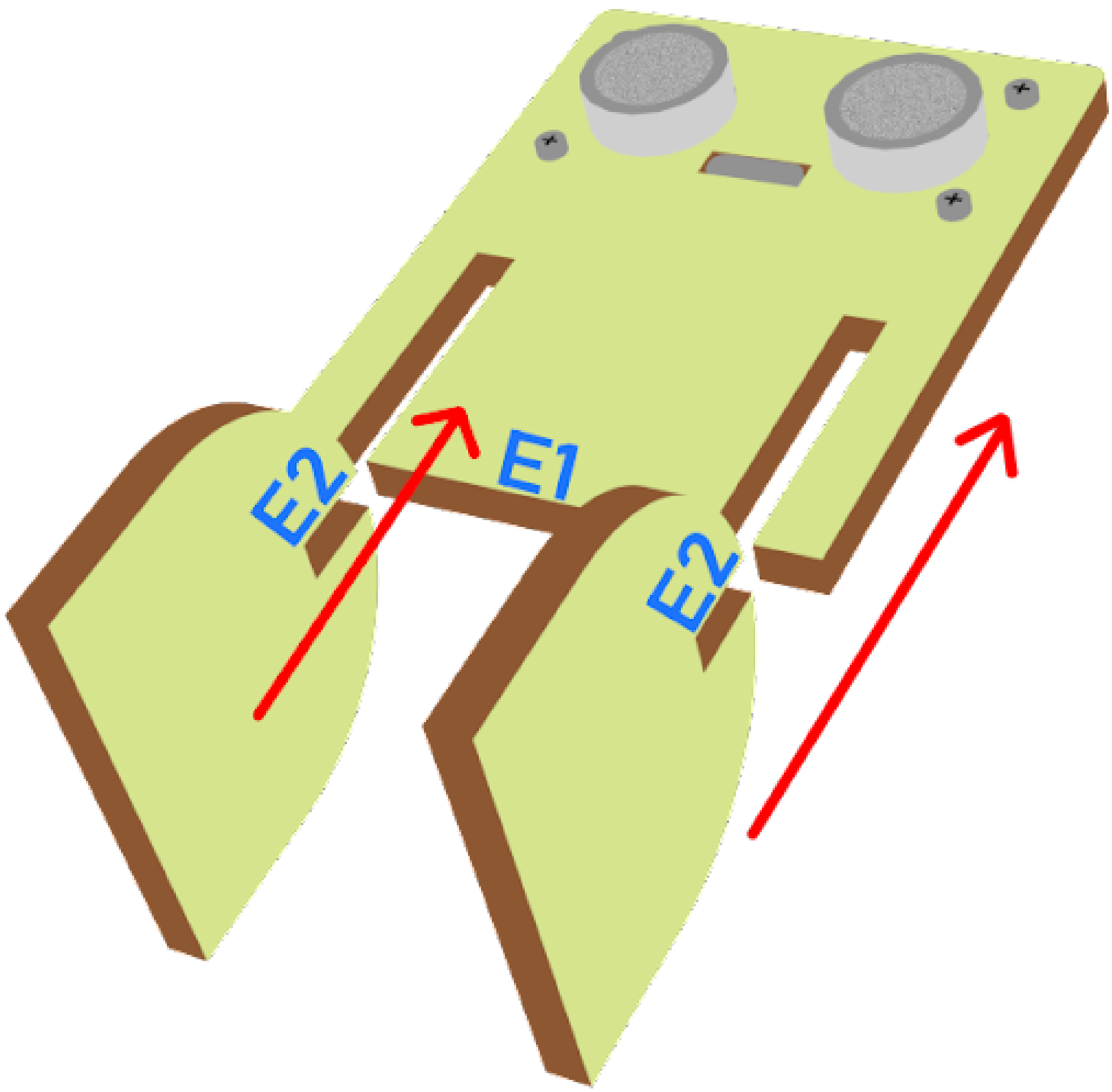
Step 3



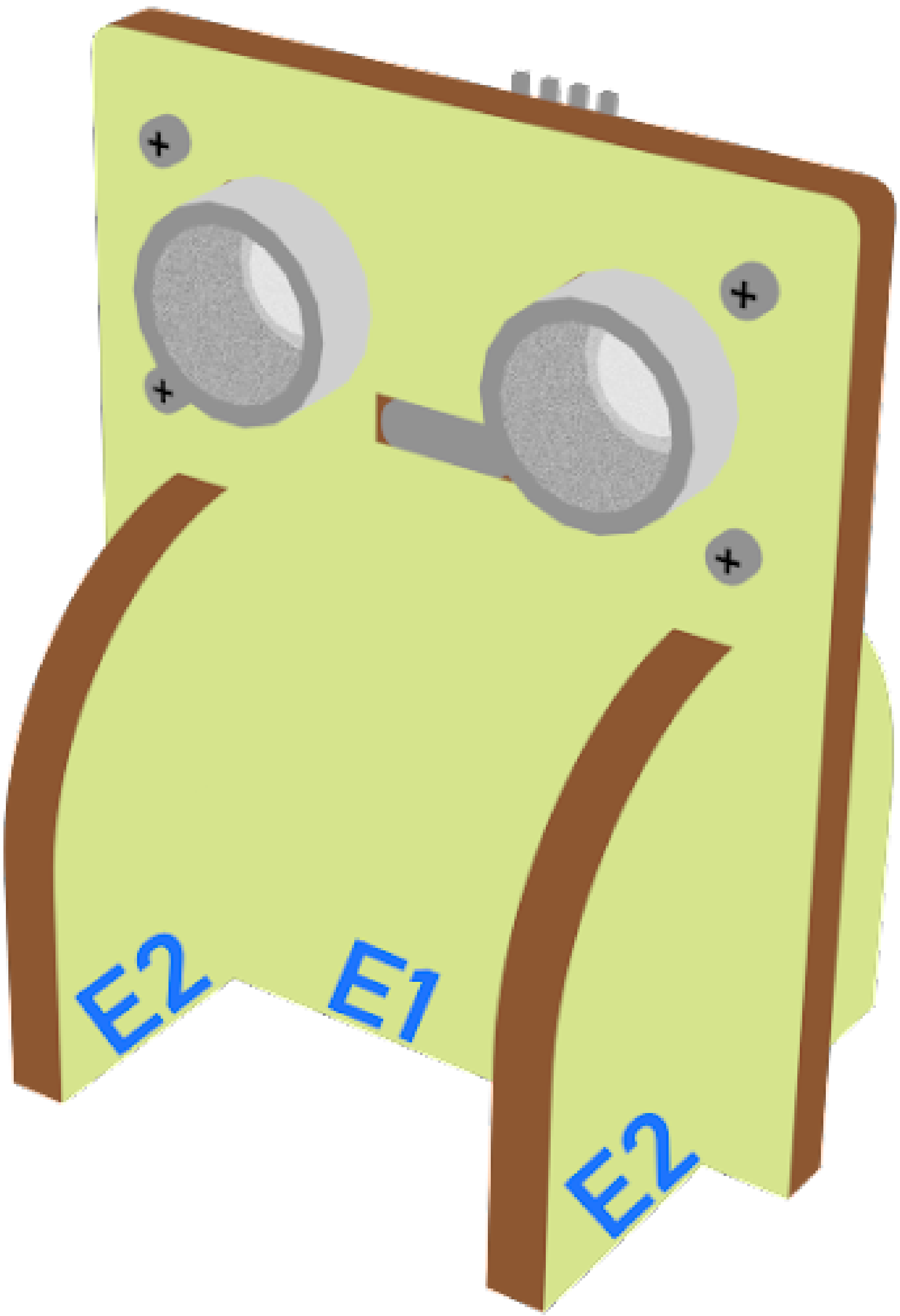
Step 4



Step 5



Step 6



Hardware connect

Programming (MakeCode)

Traffic light 1

Step 1. Set up a new function (TurnRed)

- Snap pause to wait 1 second
- Control traffic light yellow on
- Snap pause to wait 1 second
- Control traffic light red on

Step 2. Set up a new function (TurnGreen)

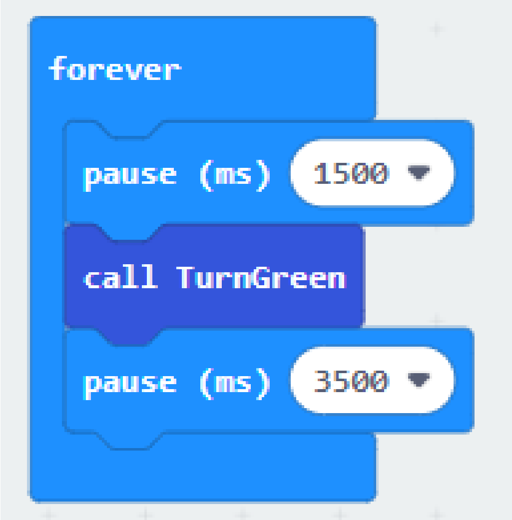
- Snap pause to wait 1 second
- Control traffic light yellow on
- Snap pause to wait 1 second
- Control traffic light green on

Step 3. Initialize the program

- Drag set variable trafficLight to Traffic light pin setting Red P0 Yellow P1 Green P2 to on start
- Drag radio set group 10 to on start
- Control traffic light red on
- Pause for 5s

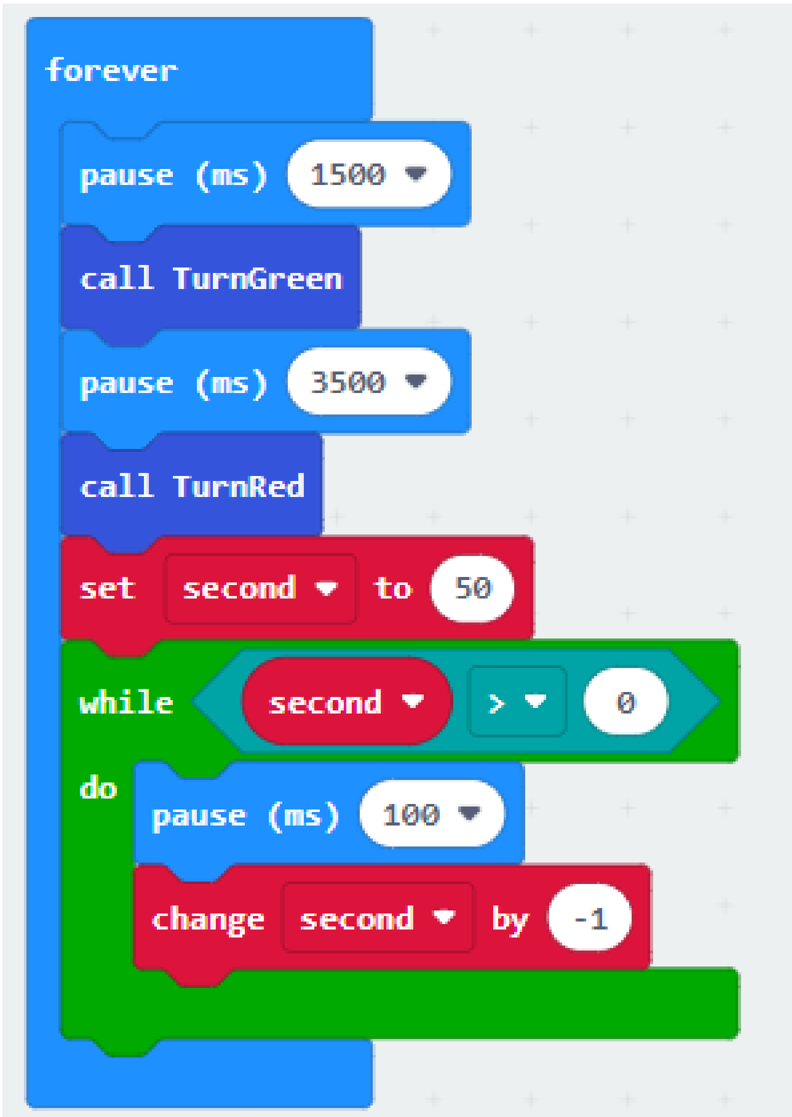
Step 4. Keep the green light status for 5 second

- Call function TurnGreen
- Pause 1.5s before TurnGreen
- Pause 3.5s after TurnGreen



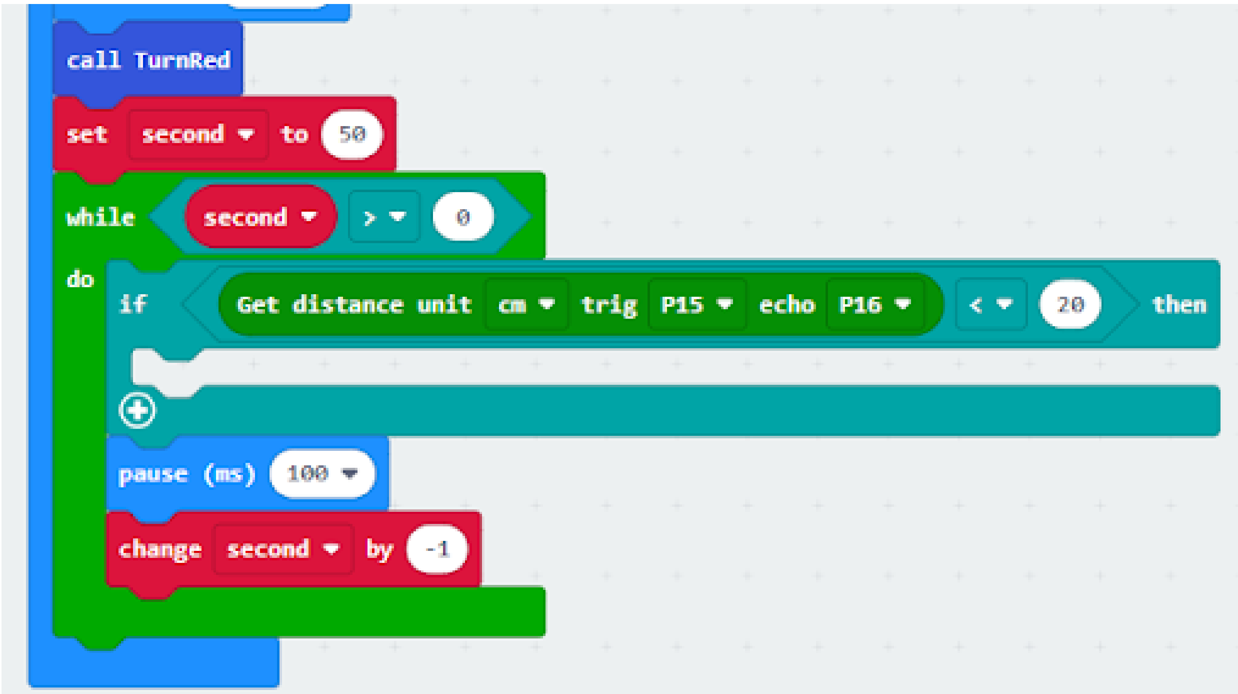
Step 5. Keep the red light status for 5 second

- Call function TurnRed
- set variable second to 50
- While second > 0, snap pause to 0.1 second and change second by -1.



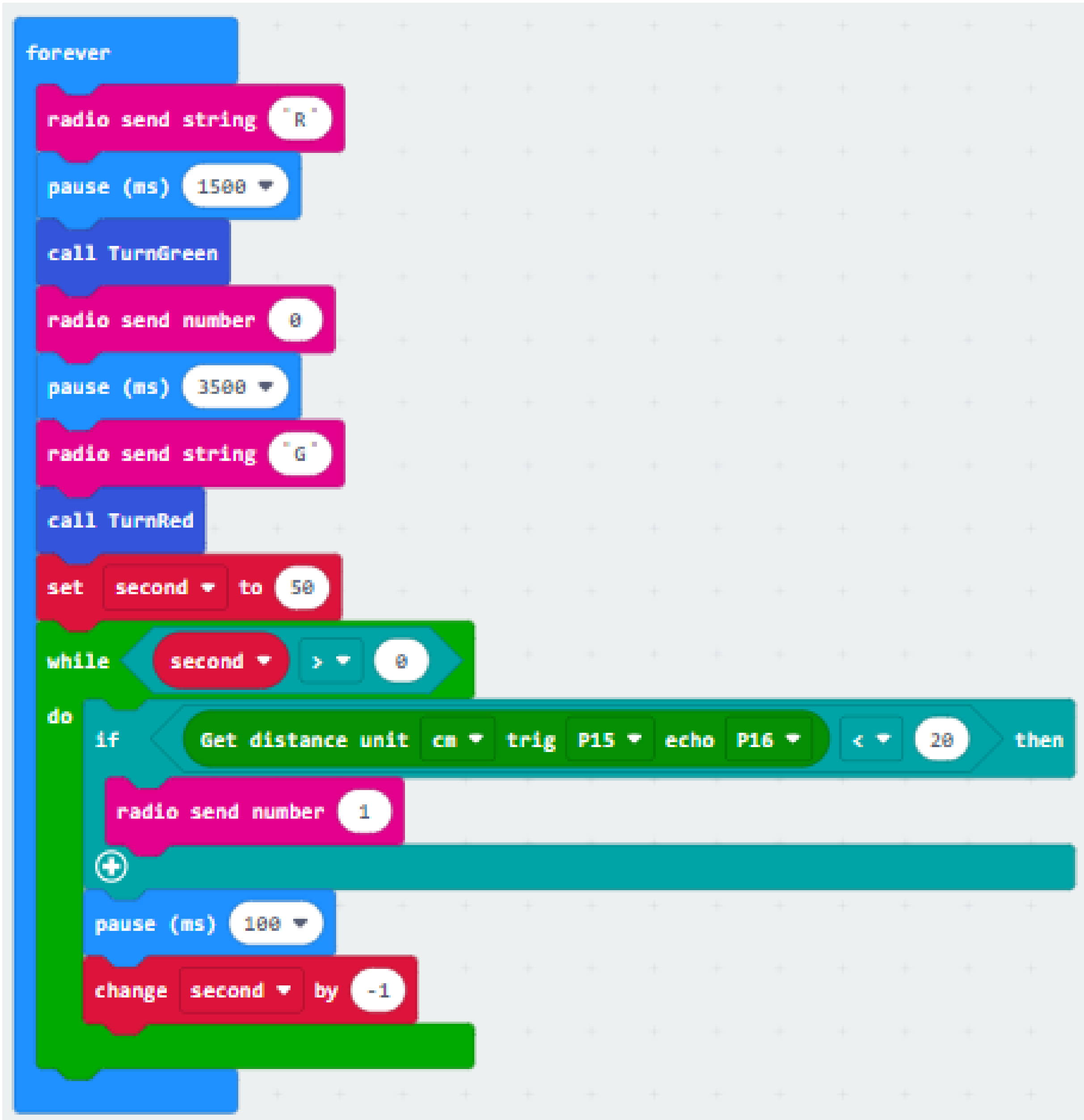
Step 6. Get distance value

- Snap if statement into while loop, set get distance unit cm trig P15 echo P16 < 20



Step 7. Control traffic light 2 and car by sending radio number

- Drag radio send number 1 into if
- Drag radio send number 0 after TurnGreen
- Drag radio send string "R" before TurnGreen
- Drag radio send string "G" before TurnRed

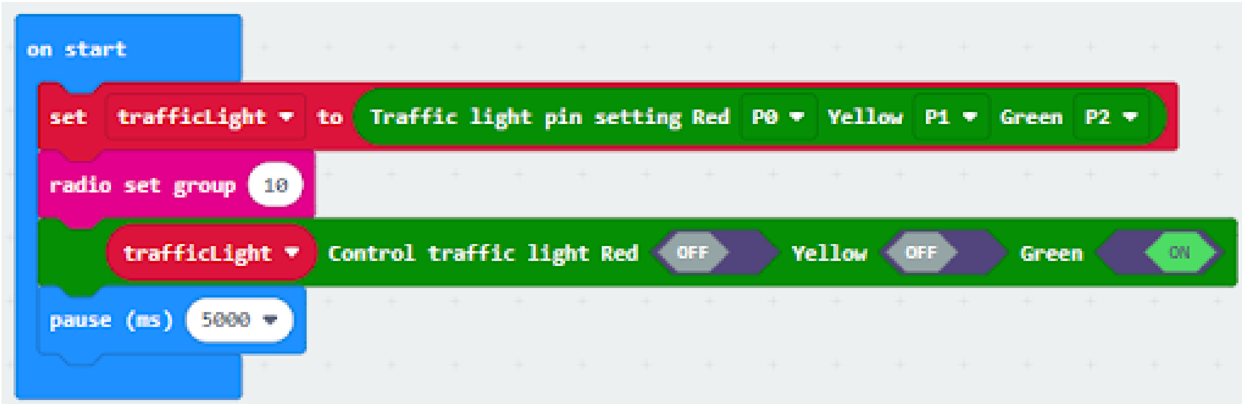


Traffic light 2

Step 1. Set up new functions

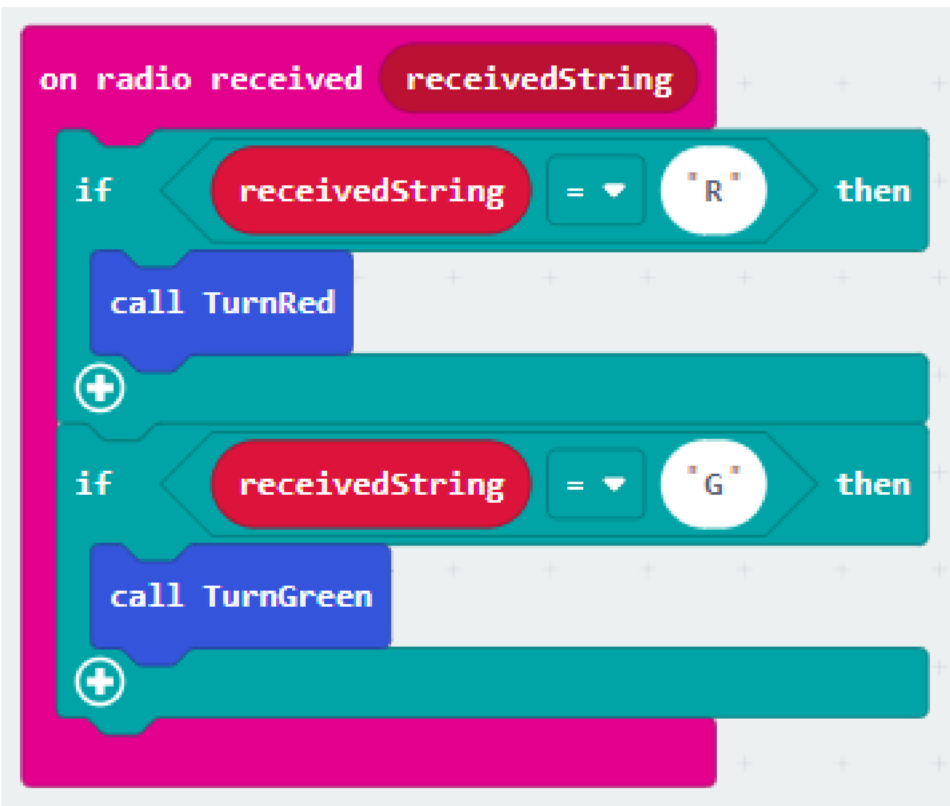
Step 2. Initialize the program

- Drag set variable trafficLight to Traffic light pin setting Red P0 Yellow P1 Green P2 to on start
- Drag radio set group 10 to on start
- Control traffic light green on
- Pause for 5s



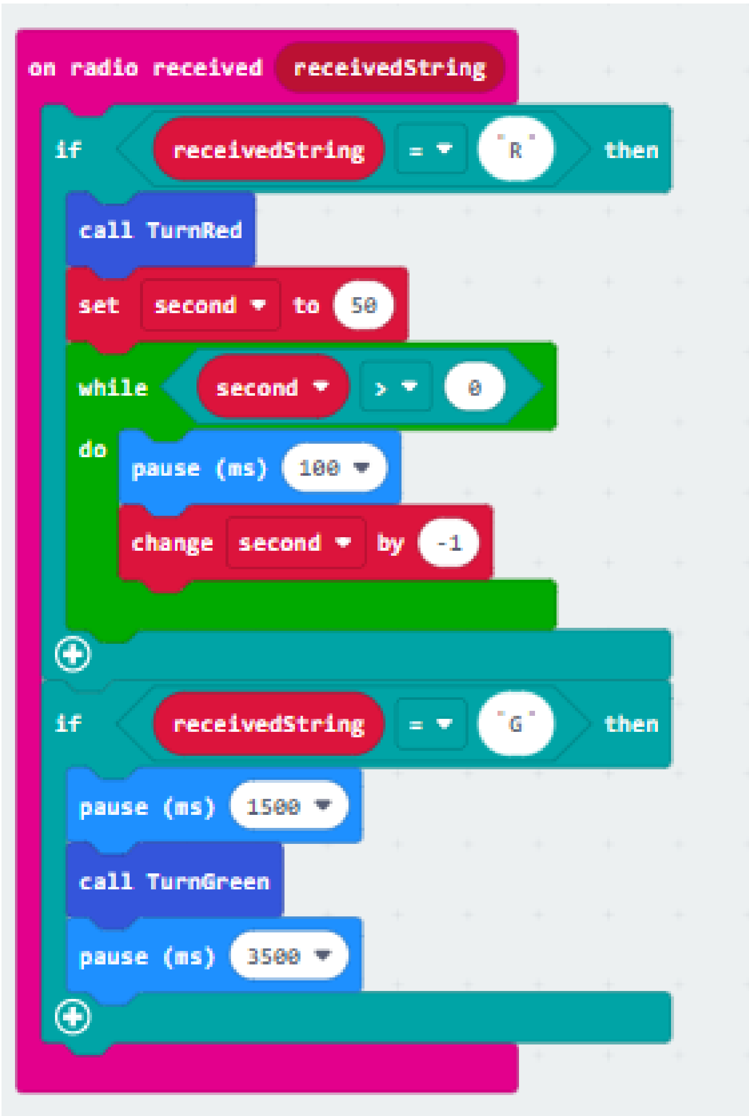
Step 3. Control traffic light and car by receiving different number

- Snap if statement into on radio received receivedString
- Set receivedString =“R” and call TurnRed
- Set receivedString =“G” and call TurnGreen



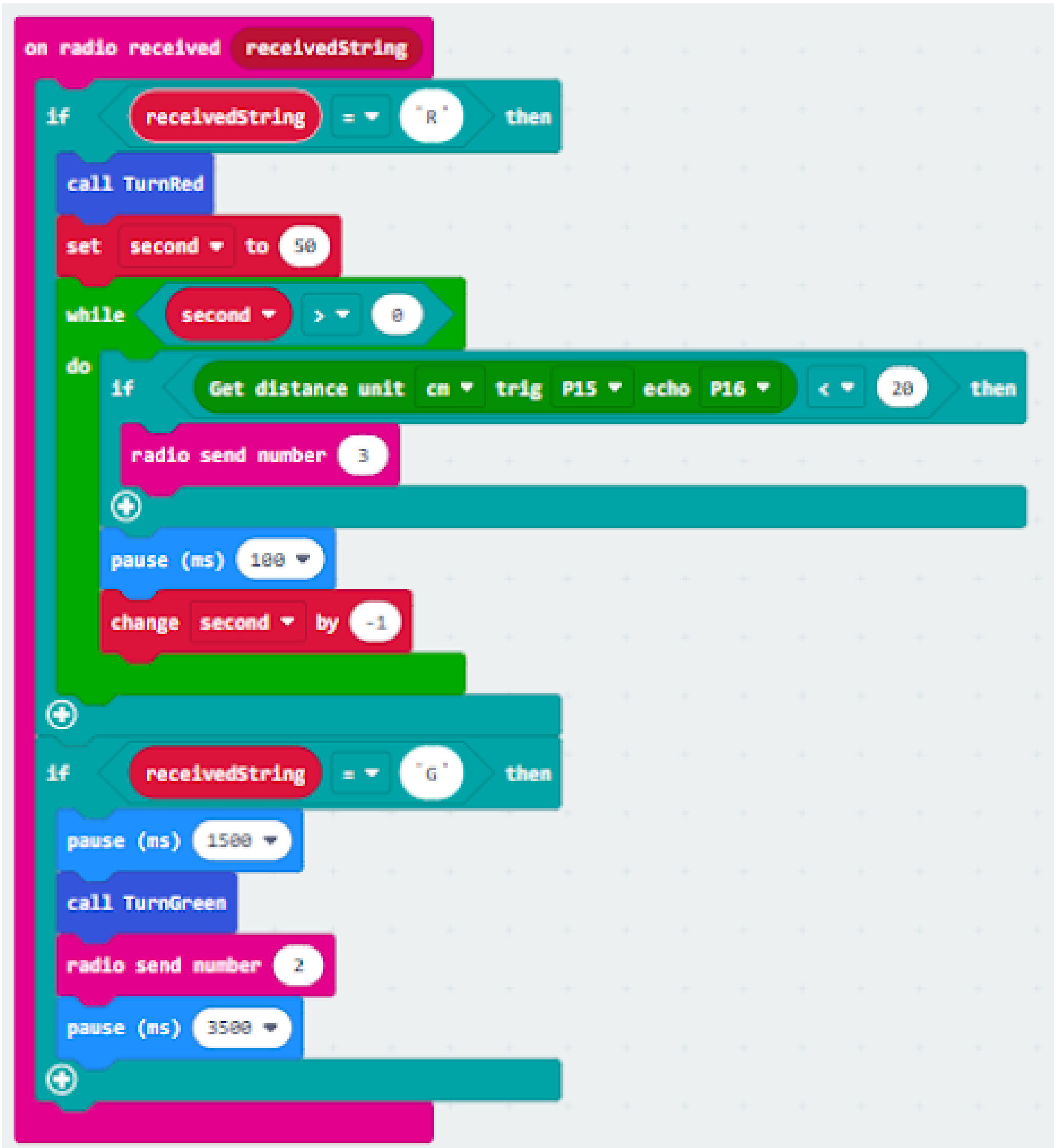
Step 4. Keep the red light status for 5 second

- After TurnRed, set variable second to 50
- While second > 0, snap pause to 0.1 second and change second by -1.
- Before TurnGreen, pause 1.5s
- After TurnGreen, pause 3.5s



Step 5. Get distance value and control the car

- Snap if statement into while loop, set get distance unit cm trig P15 echo P16 < 20
- Drag radio send number 3 into if 1 if



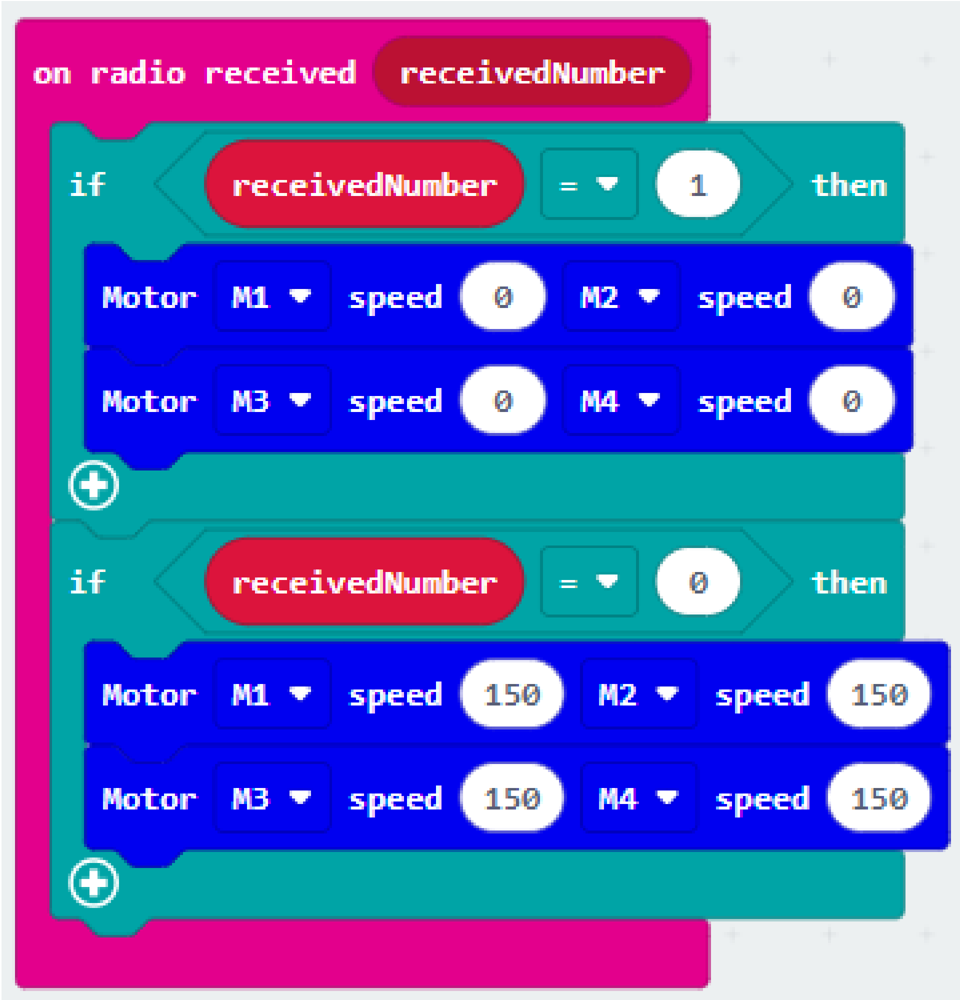
Car 1:

Step 1. Set radio set group at start position

- Drag radio set group 10 to on start

Step 2. Control car by receiving different number

- Snap if statement into on radio received receivedNumber
- Set receivedNumber =1 and make the car stop
- Set receivedNumber=0 and make the car move forward



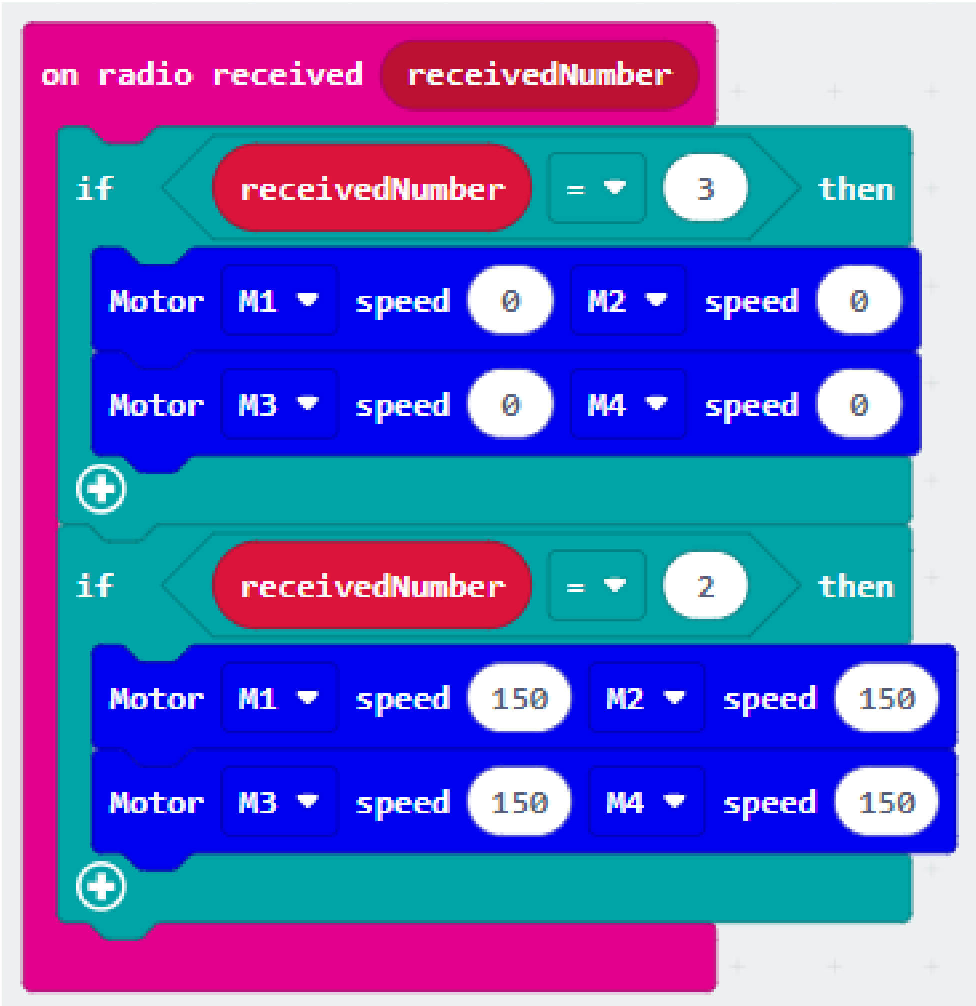
Car 2:

Step 1. Set radio set group at start position

- Drag radio set group 10 to on start

Step 2. Control car by receiving different number

- Snap if statement into on radio received receivedNumber
- Set receivedNumber =3 and make the car stop
- Set receivedNumber=2 and make the car move forward



Result

Think

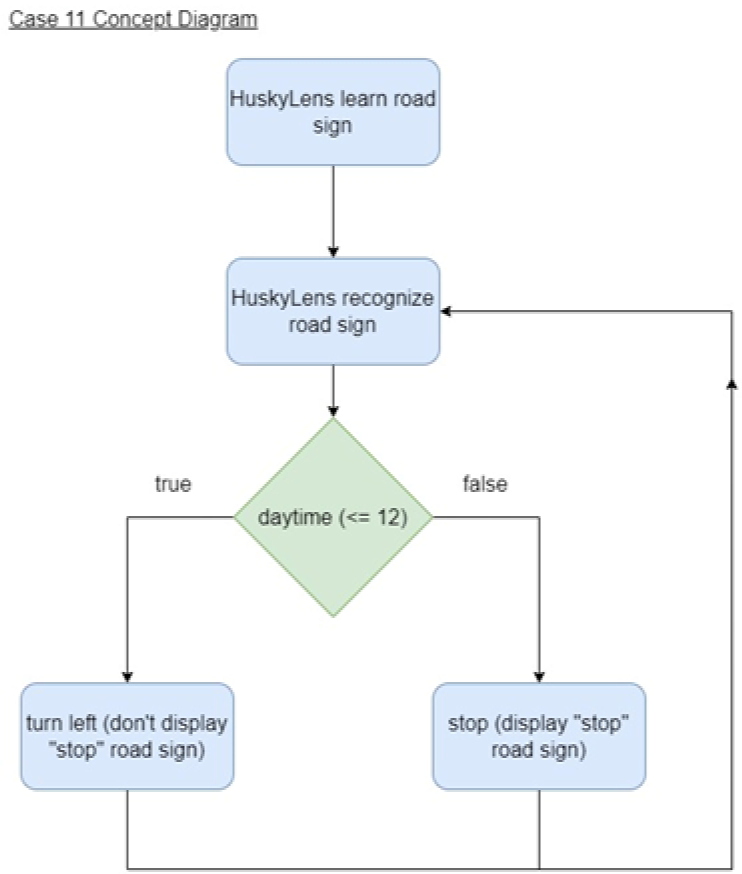
1.2.12 Intelligent Road Signs

Goal

Background

What is intelligent road sign?

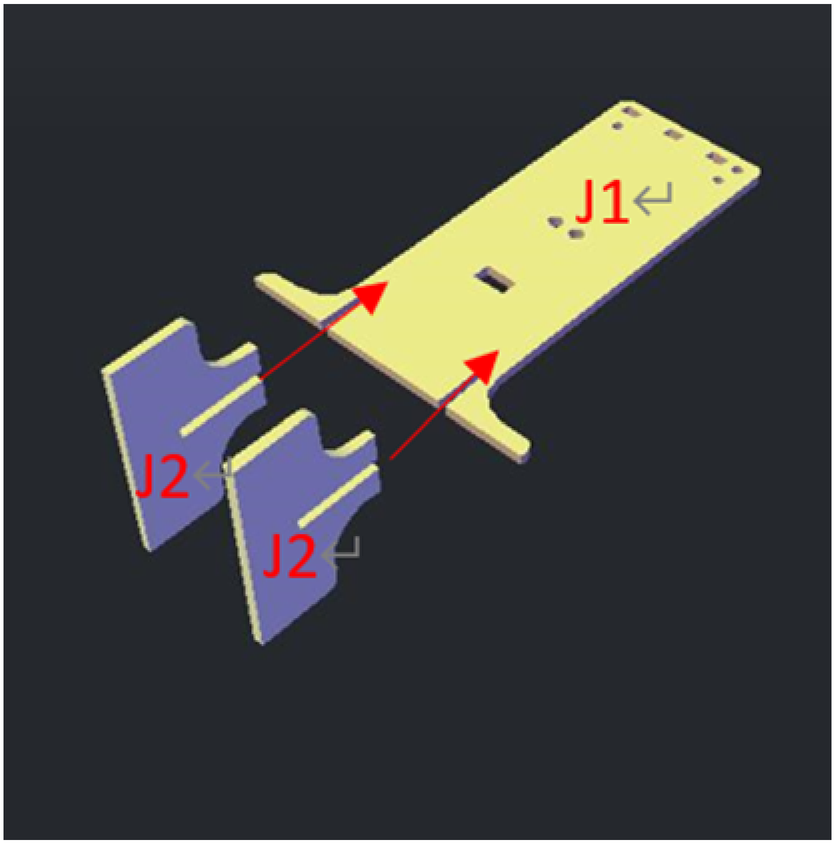
Intelligent road sign operation



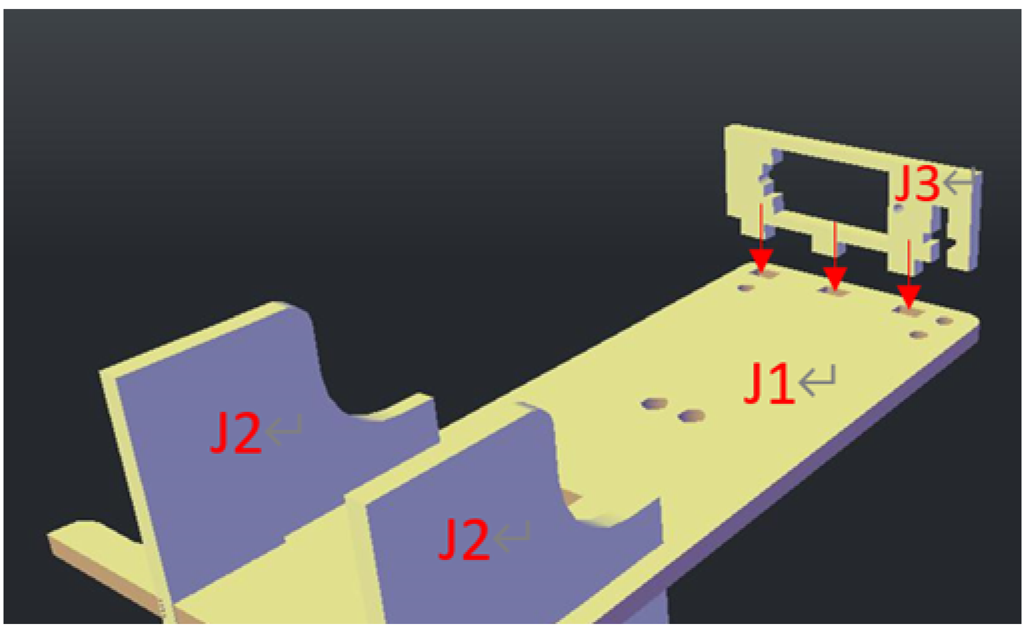
Part List

Assembly step

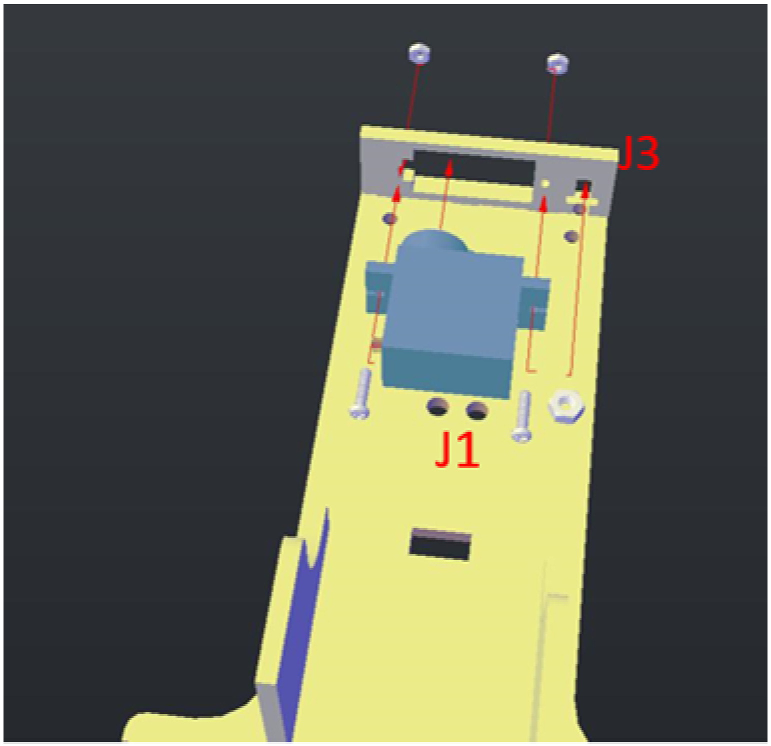
Step 1



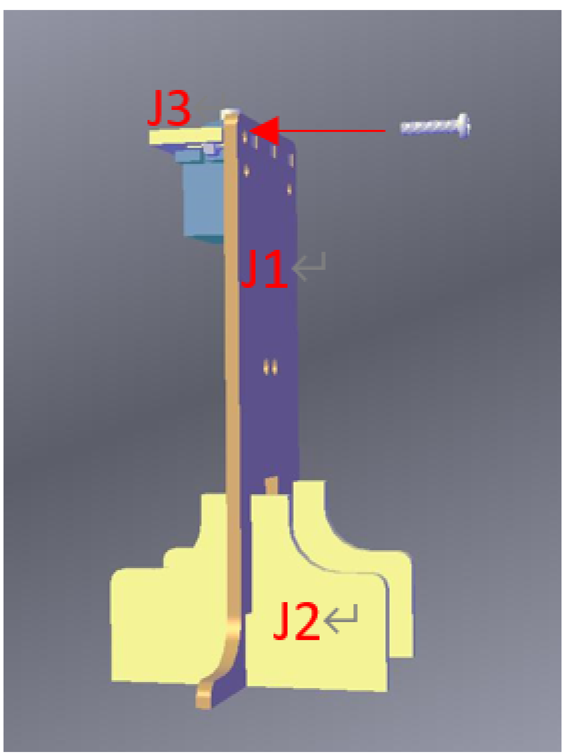
Step 2



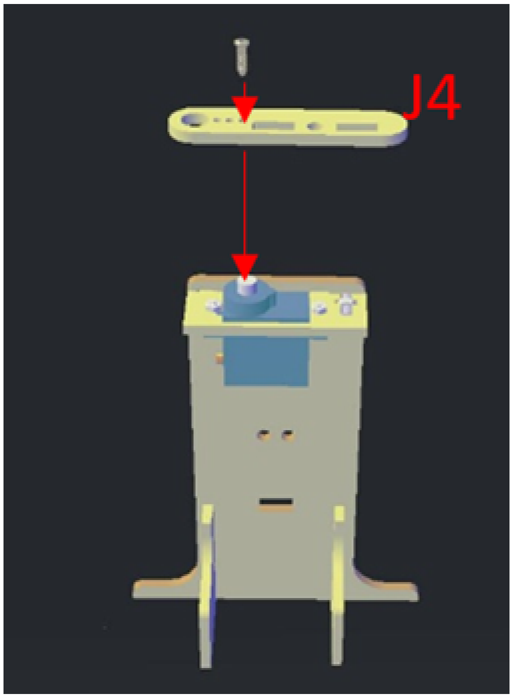
Step 3



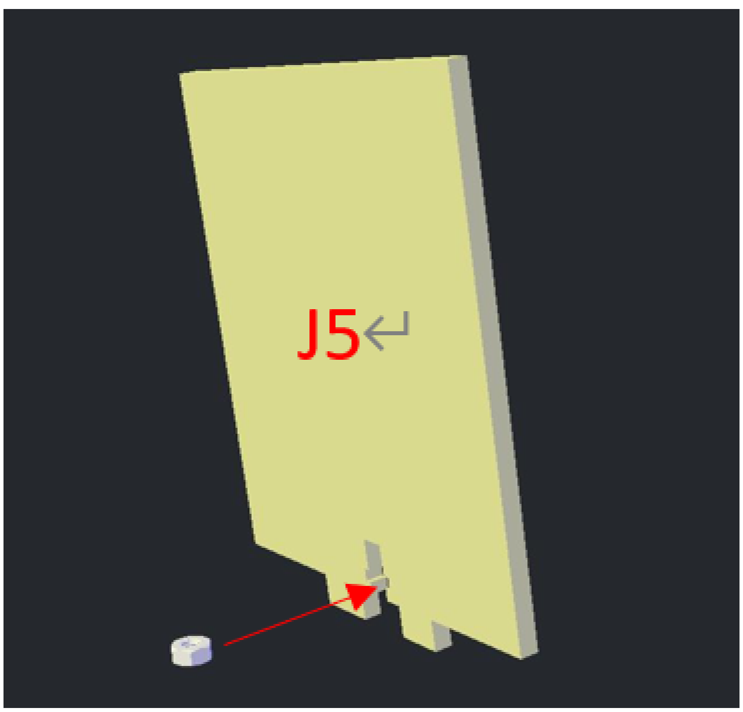
Step 4



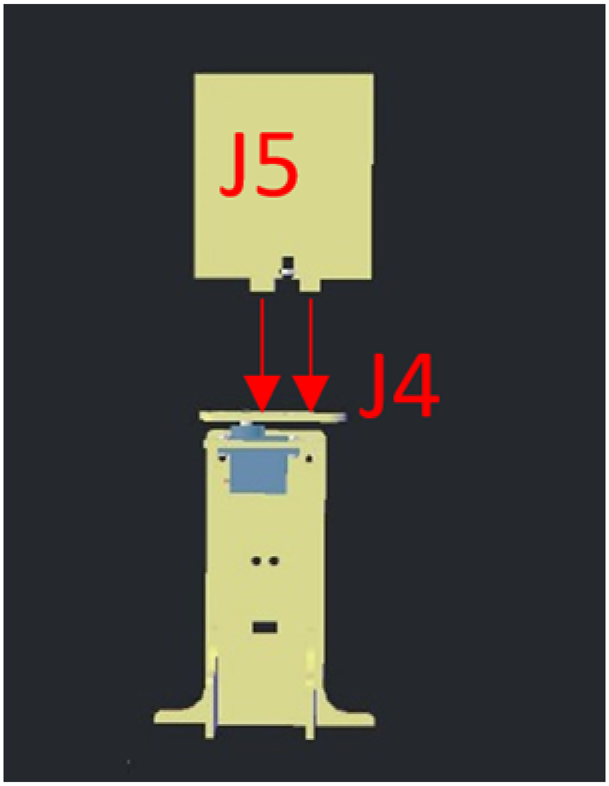
Step 5



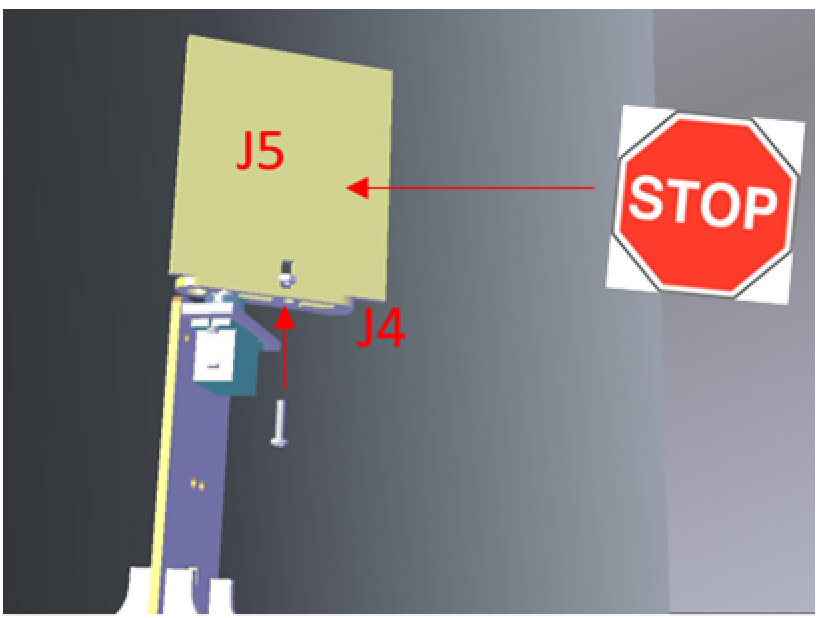
Step 6



Step 7



Step 8



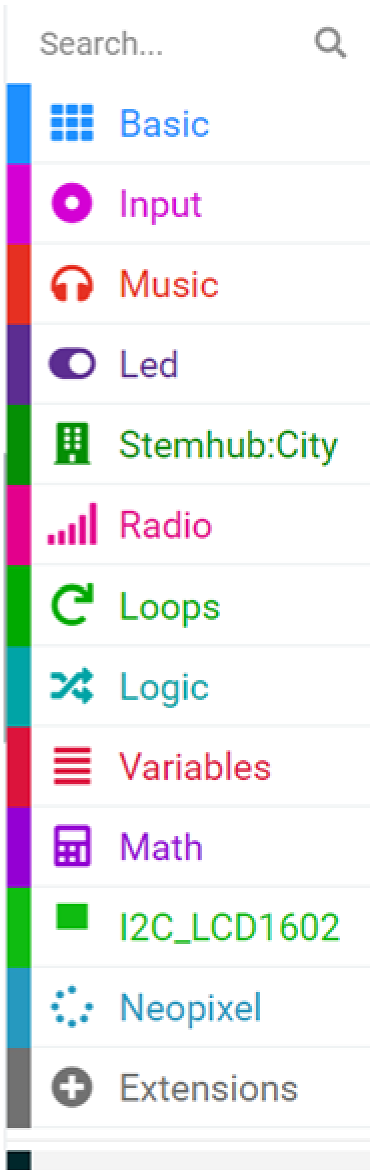
Step 9

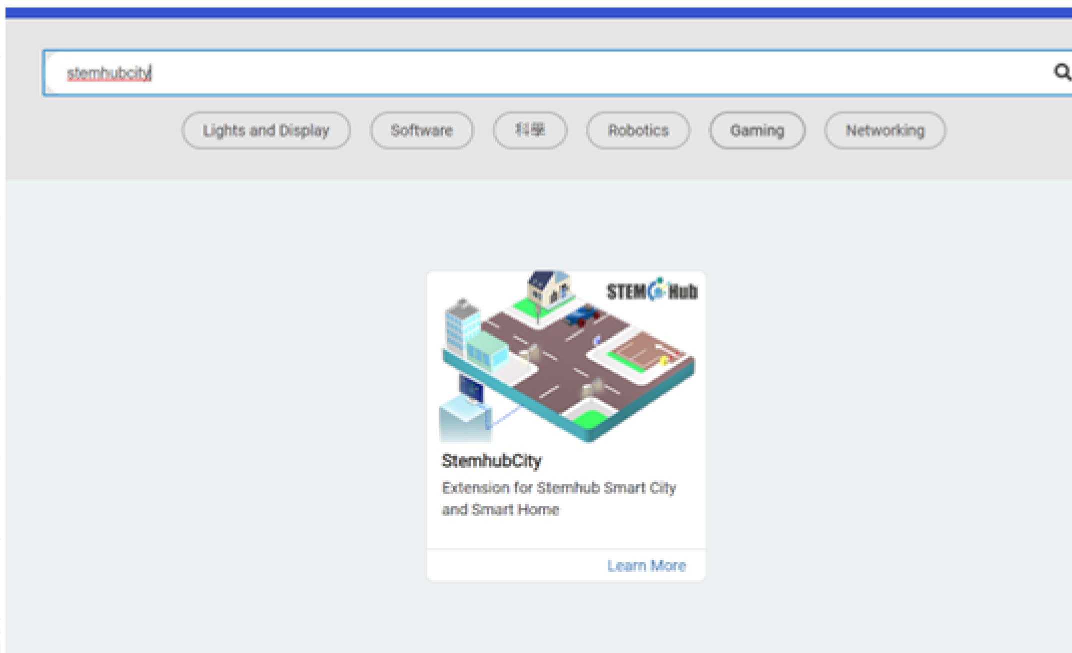
Hardware connect

Programming (MakeCode)

Install library

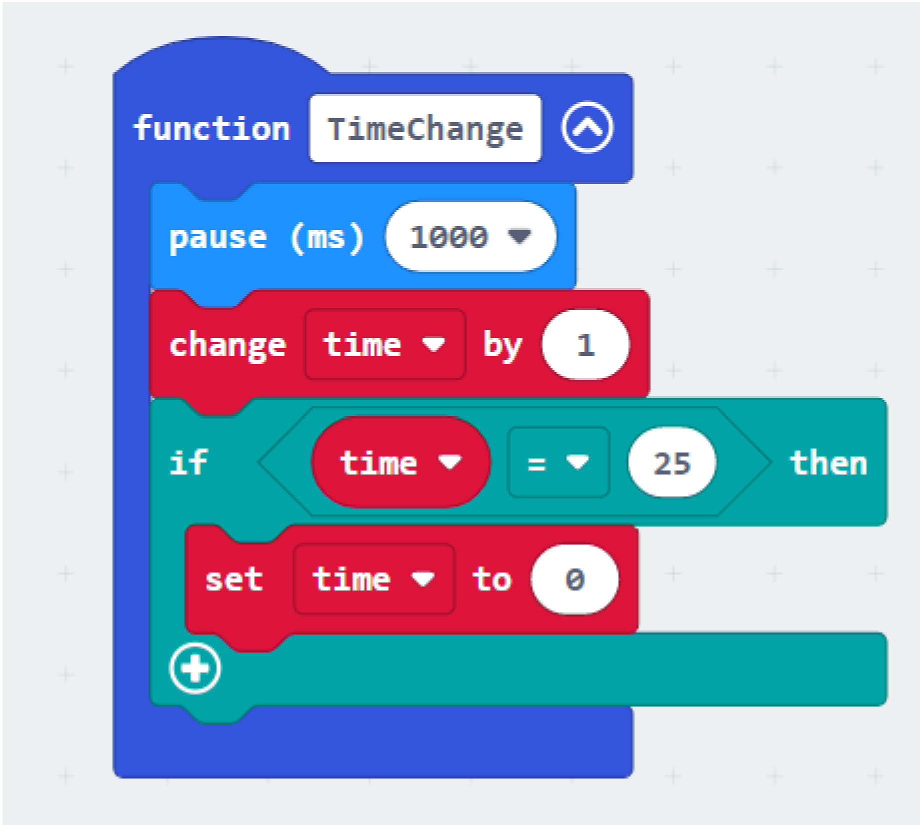
- Click extensions
- Enter StemhubCity





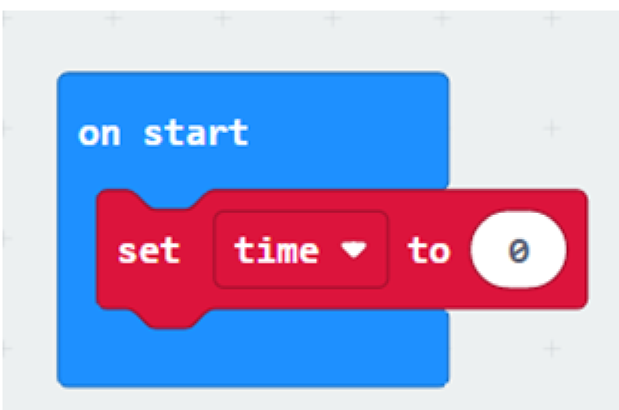
Update Time

- Pause 1 second
- Then increase time by 1
- Set time to 0 when time is equal to 25



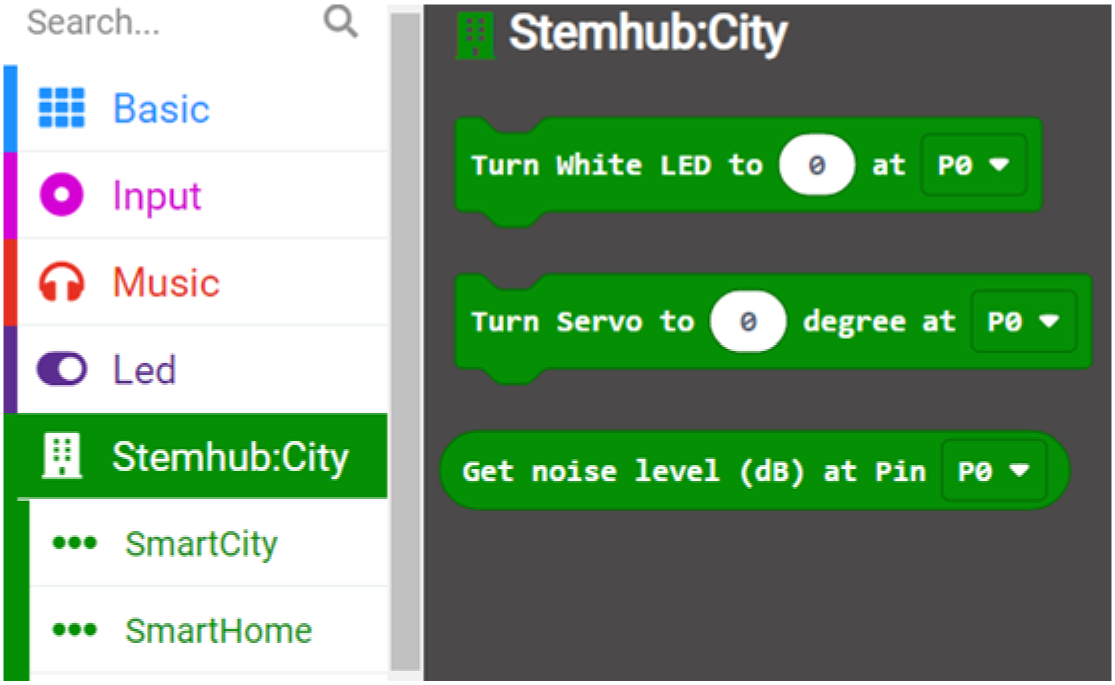
Initialize Time

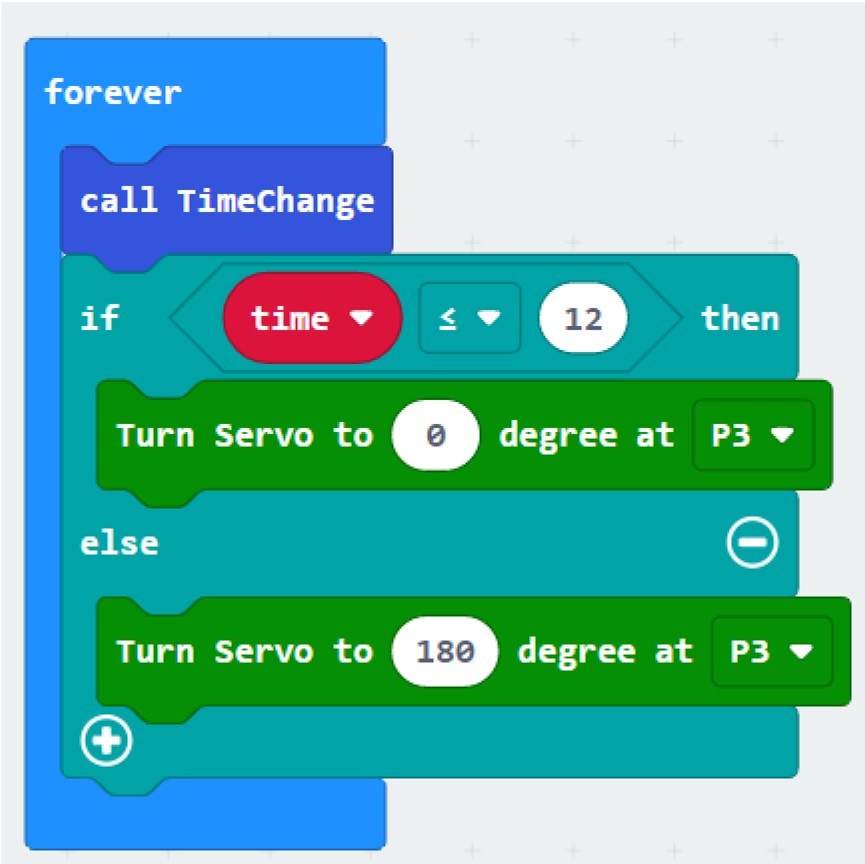
- set time to 0



Checking time and changing road signs

- Call function TimeChange to update the time
- Drag out two turn servo to 0 degree at P0 from the stemhubcity library
- Then place them inside the if condition and else condition
- Change turn servo to 0 degree at P0 as turn servo to 0 degree at P3 and turn servo to 180 degree at P3




forever
call TimeChange
if time ≤ 12 then
Turn Servo to 0 degree at P3
else
Turn Servo to 180 degree at P3

Result

Think

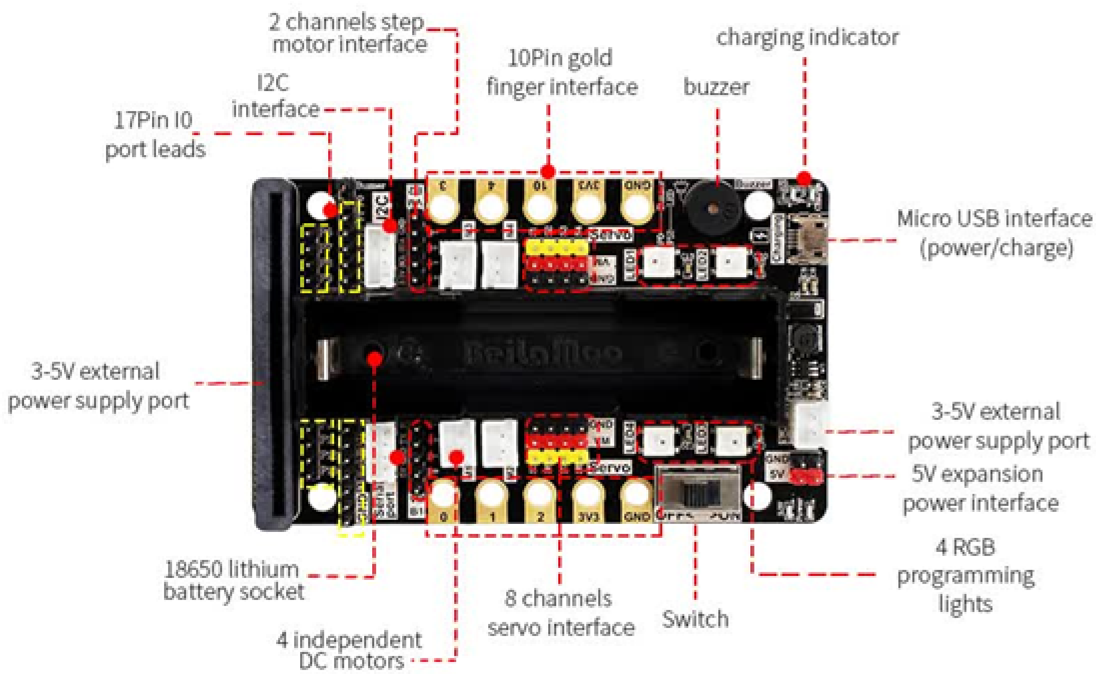
1.3 Micro: bit M1 SMART CAR_Beginner

1.3.1 Lesson 1

Introduction

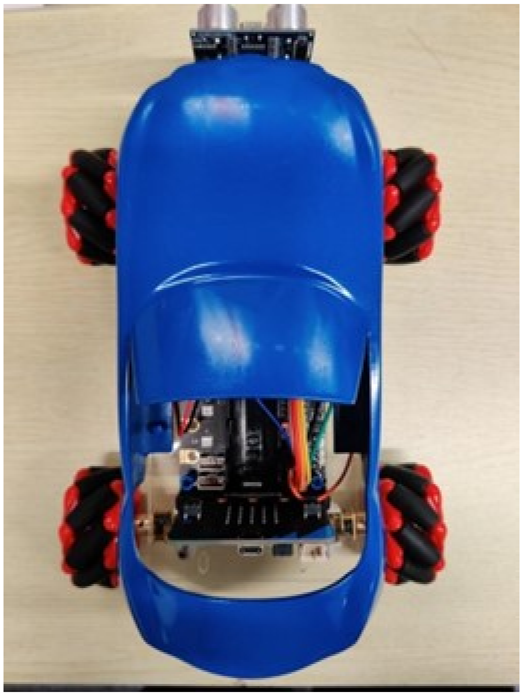
Teaching Objectives

Understand micro: bit expansion Circuit Board

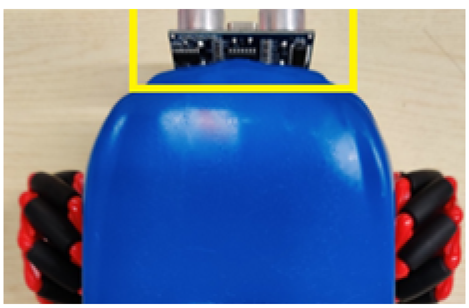


Understand the Mecanum Wheel Car

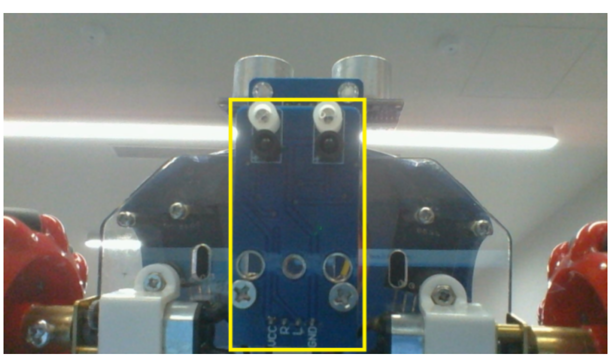
- 4 Motors
- 4 Mecanum Wheels
- Micro bit
- Micro bit Expansion Circuit Board



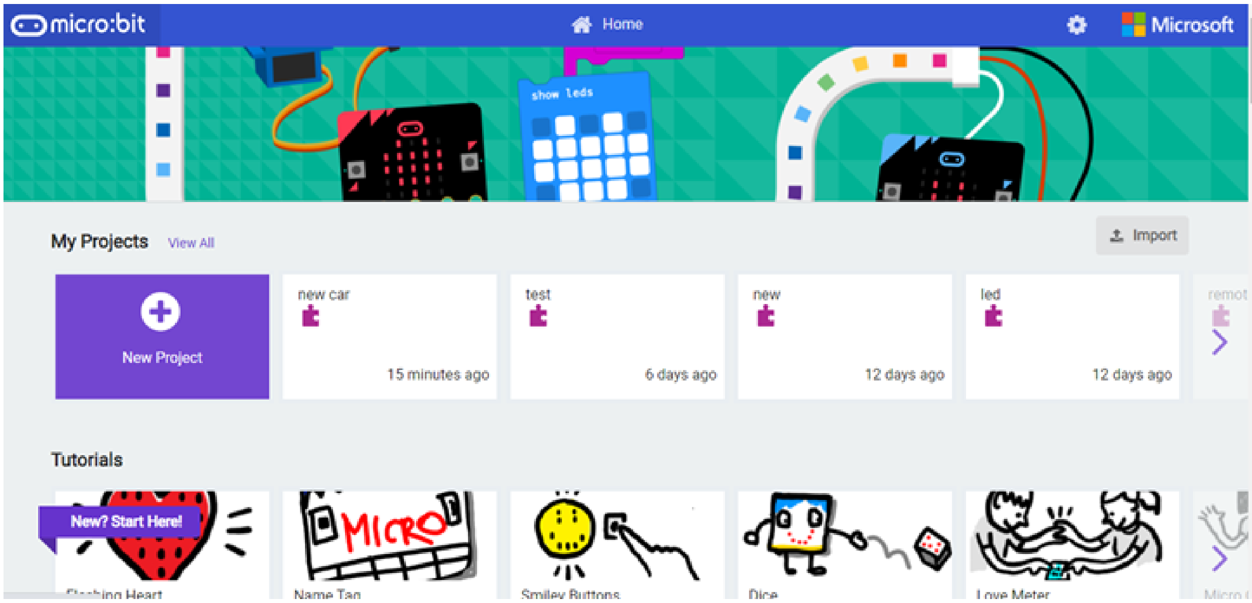
- Ultrasonic Sensor

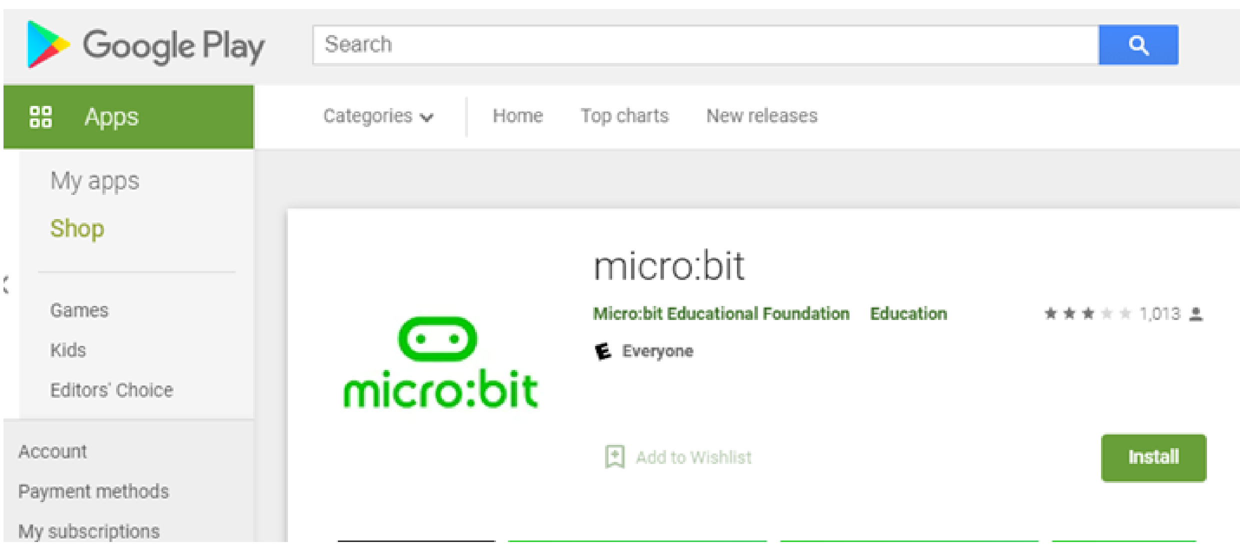


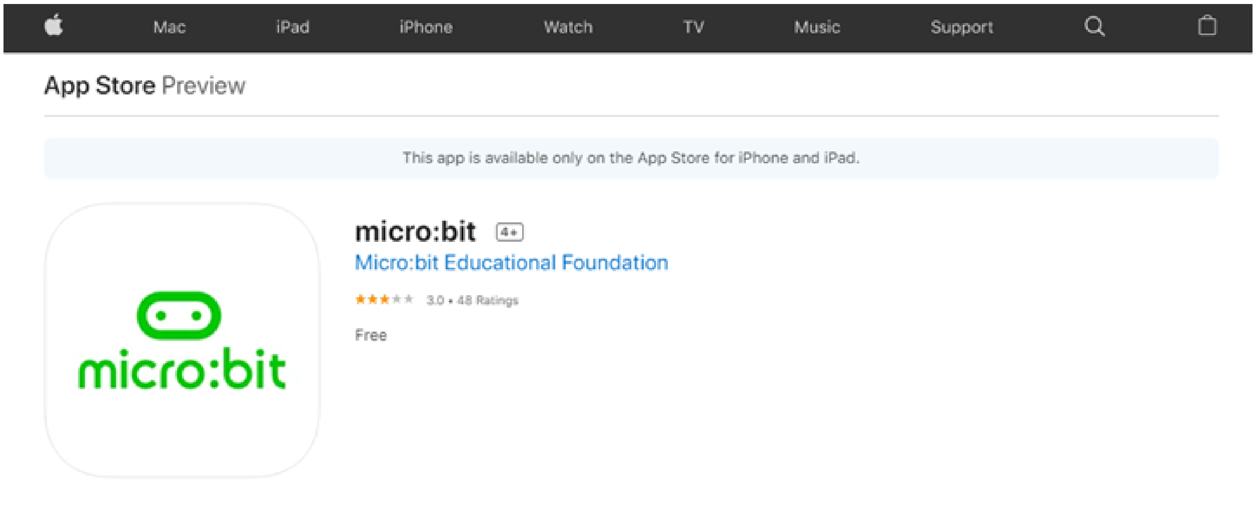

- Infrared Line Follower



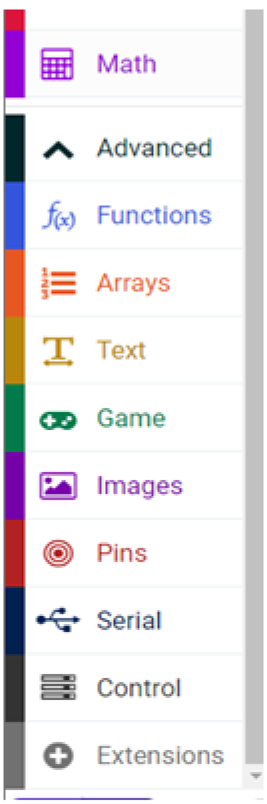

Preparing Micro: bit Programming Software Makecode







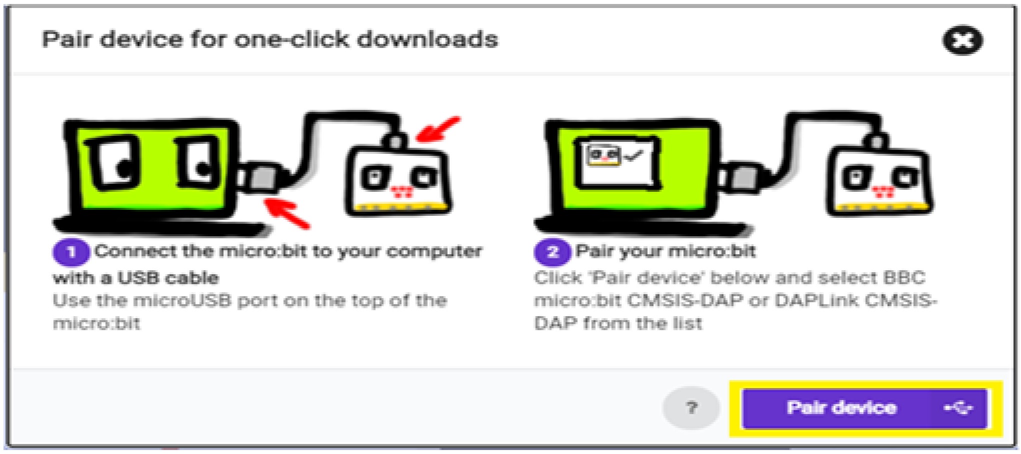
New Extensions

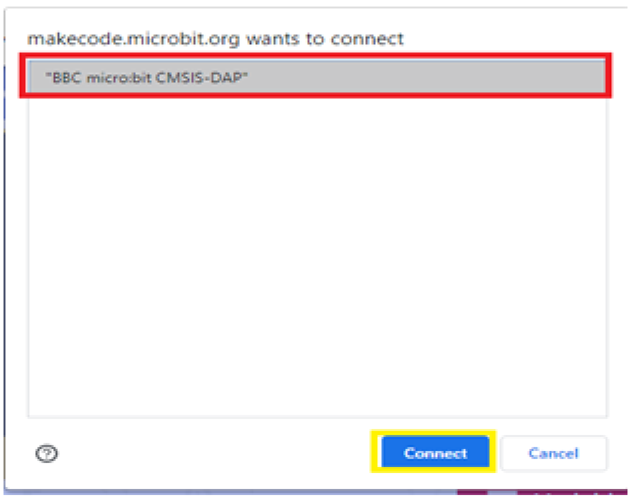


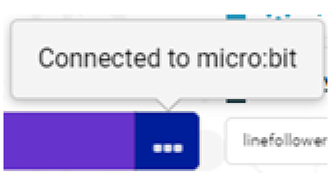

返回
擴展
?
bluetooth
Bluetooth services
radio-broadcast
Adds new blocks for message communication in the radio
servo
A micro-servo library
neopixel
AdaFruit NeoPixel driver


返回
擴展
?
https://github.com/stemhub/pxt-Stemhubbit
stemhubbit
Extension for Stemhub Car

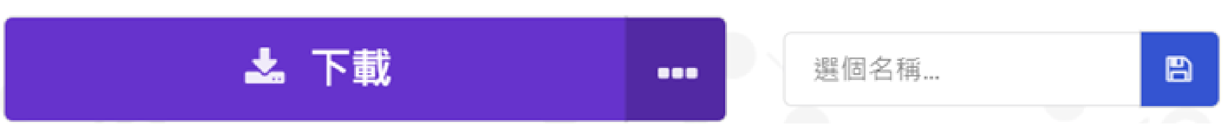
Connect micro: bit to the computer





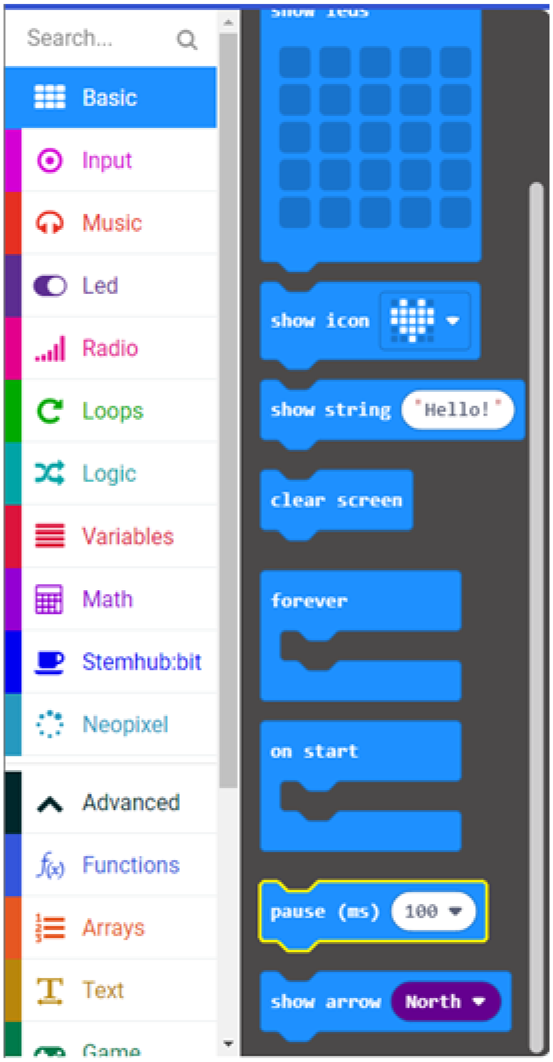


Save program file.hex) to micro: bit

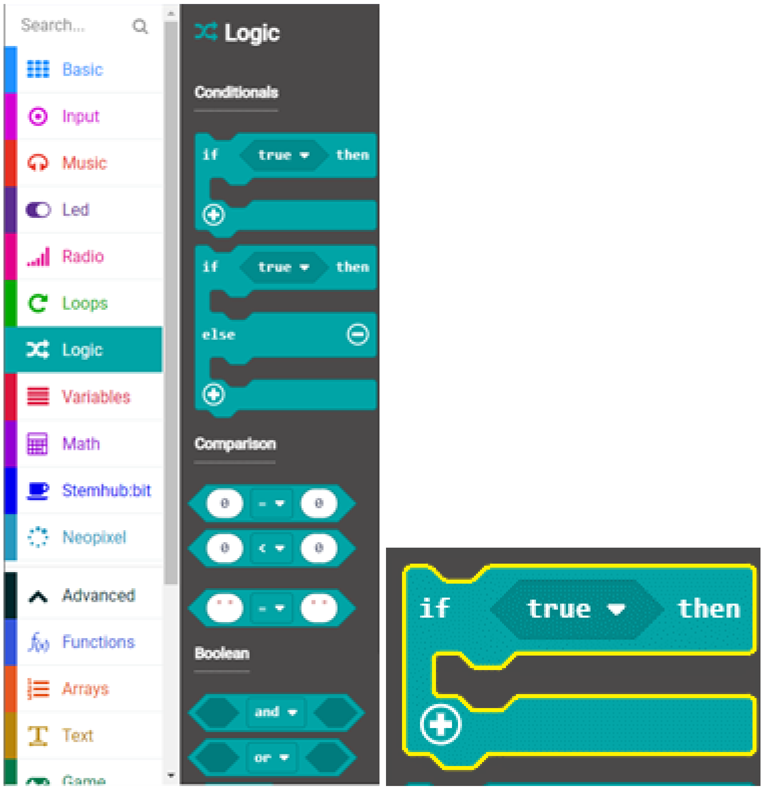


understand Block Programming

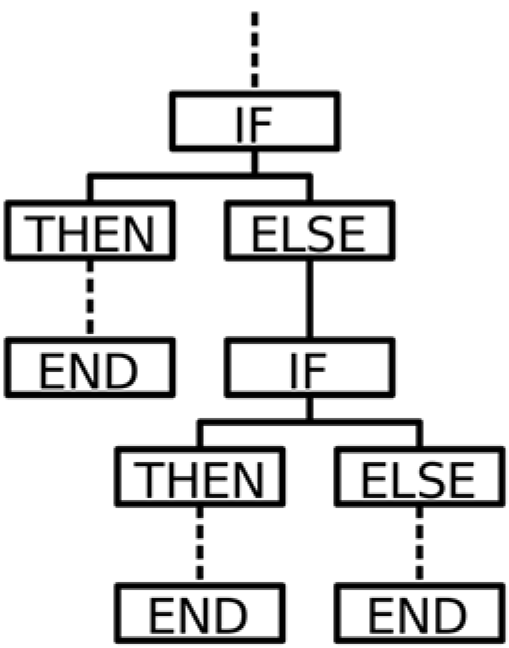
PAUSE Block module:



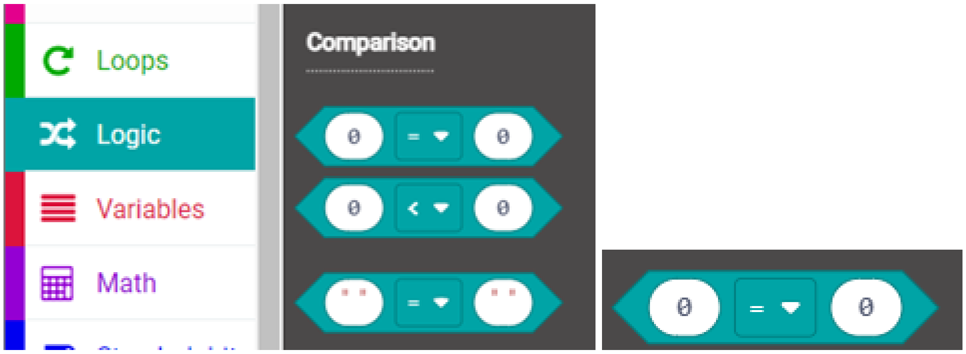
IF Logic Block Module:



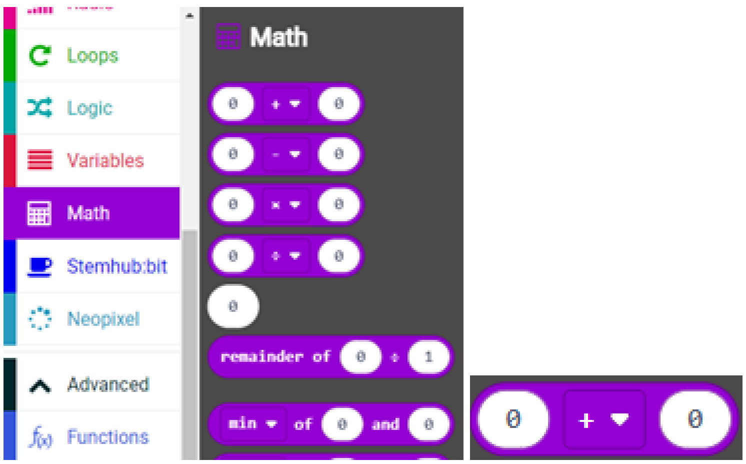
- “else if” can be used to set additional condition
- “else“ is used to set the action to be taken when condition of both “if” and “else if” did not fulfilled.



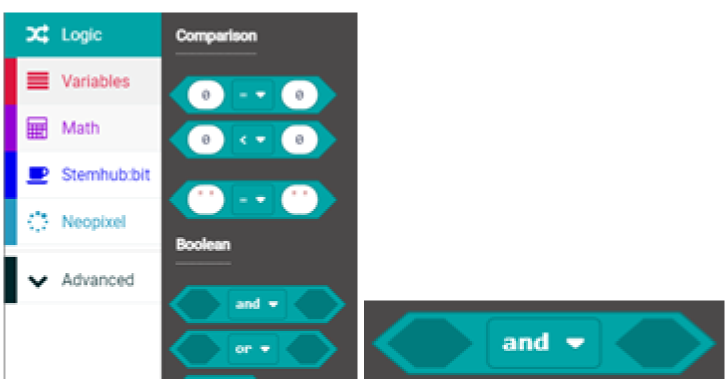
Comparison block module:



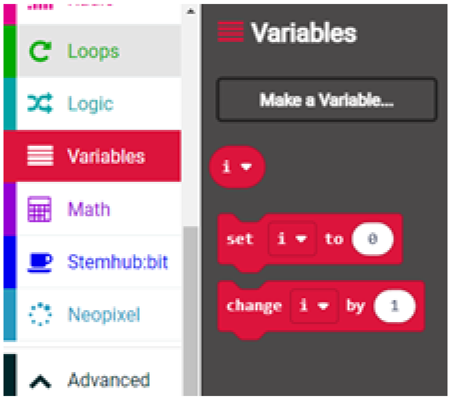
Mathematical operation block module:



"AND,OR" block module

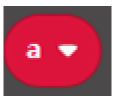


” Variable”


Loops
Logic
Variables
Math
Stemhub:bit
Neopixel
Advanced
Variables
Make a Variable...
i
set i to 0
change i by 1


Make a Variable...


New variable name:
Ok


a


set i to 0

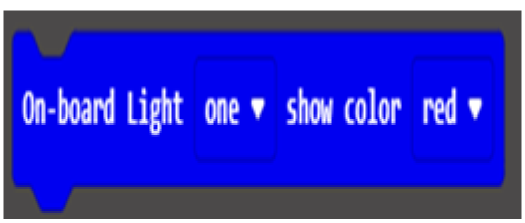

change i by 1

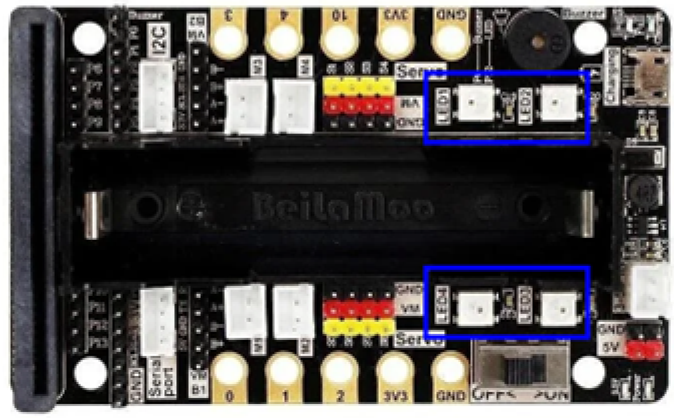
1.3.2 Lesson 2

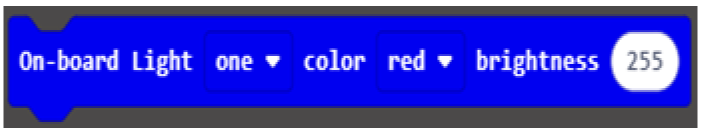
Introduction

Teaching Objectives

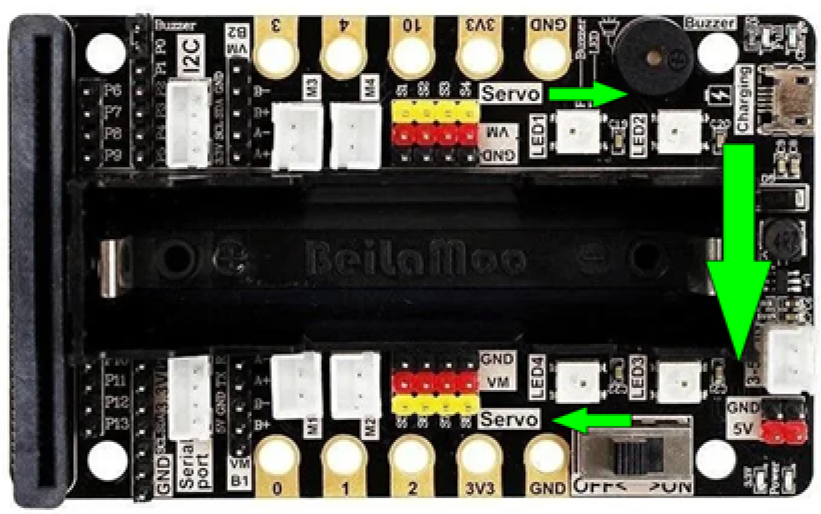
Control of RGB light (LED) on expansion board



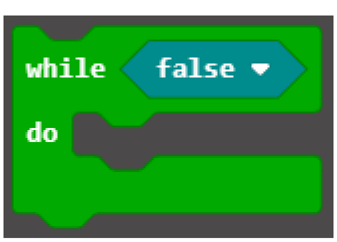


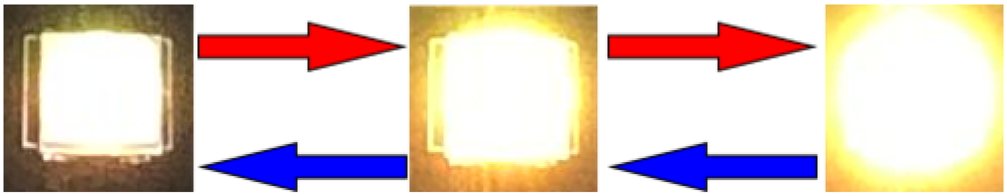


Exercise 1: design a program to flash the LED on the expansion board in order.

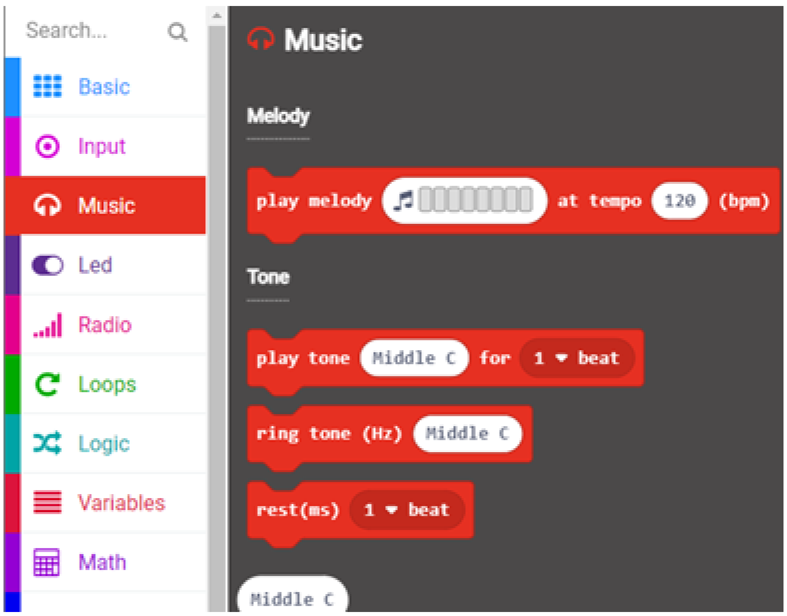


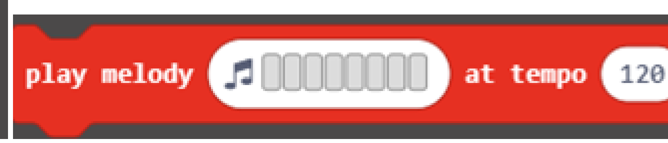
Exercise 2: design a program to change the brightness of the LED on expansion board start from dim to bright and then bright to dim. Breathing light effect





Buzzer

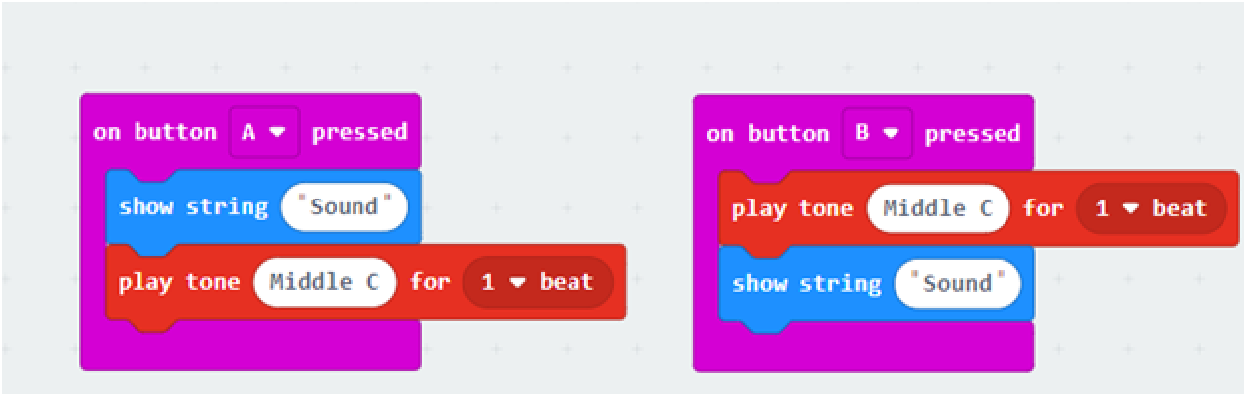

Search...
Basic
Input
Music
Led
Radio
Loops
Logic
Variables
Math
Music
Melody
play melody at tempo 120 (bpm)
Tone
play tone Middle C for 1 beat
ring tone (Hz) Middle C
rest(ms) 1 beat
Middle C

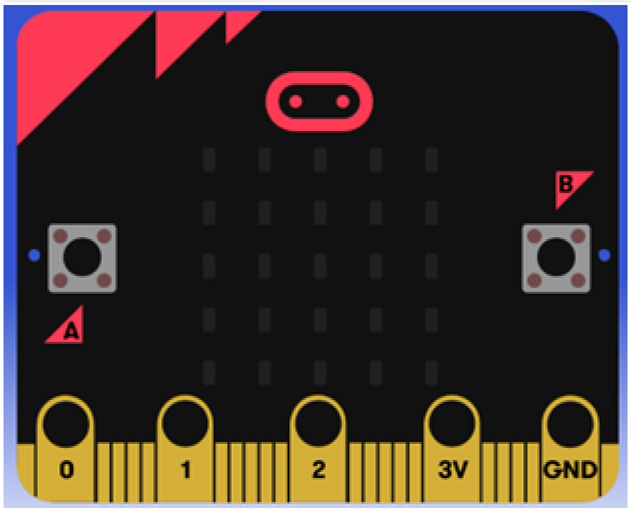

play melody at tempo 120


play tone Middle C for 1 beat


set volume 127

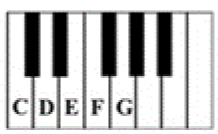
Delay of the word display





Exercise: Play a song, design a program to play the song below by your car

(rh):	1	1	5	5	5	5	5
	G	G	D	D	E	E	D
(lh):	5	5	1	1	1	1	1
	Twin -	kle	twin -	kle	lit -	tle	star
	4	4	3	3	2	2	1
	C	C	B	B	A	A	G
	2	2	3	3	4	4	5
	how	I	won -	der	what	you	are
	5	5	4	4	3	3	2
	D	D	C	C	B	B	A
	1	1	2	2	3	3	4
	up	a -	bove	the	world	so	high
	5	5	4	4	3	3	2
	D	D	C	C	B	B	A
	1	1	2	2	3	3	4
	like	a	dia -	mond	in	the	sky
	1	1	5	5	5	5	5
	G	G	D	D	E	E	D
	5	5	1	1	1	1	1
	Twin -	kle	twin -	kle	lit -	tle	star



Jingle Bells

James Pierpont
Arranged by Julie A. Lind

Right hand only

E E E - E E E - E G C D E - - -
Jin- gle bells, jin - gle bells, jin - gle all the way.

F F F F F E E EE
Oh what fun it is to ride in a

E D D E D - G -
one horse op - en sleigh

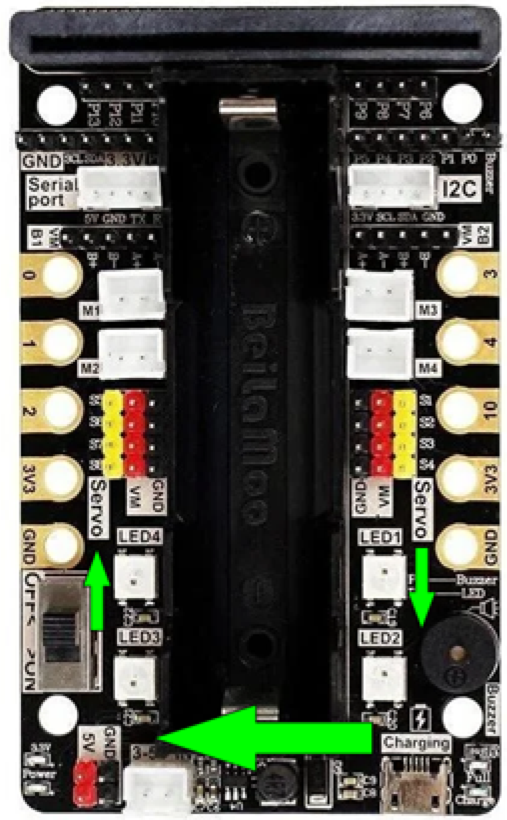
E E E - E E E - E G C D E - - -
Jin- gle bells, jin - gle bells, jin - gle all the way.

F F F F F E E EE
Oh what fun it is to ride in a

G G F D C
one horse op - en sleigh!

Answers

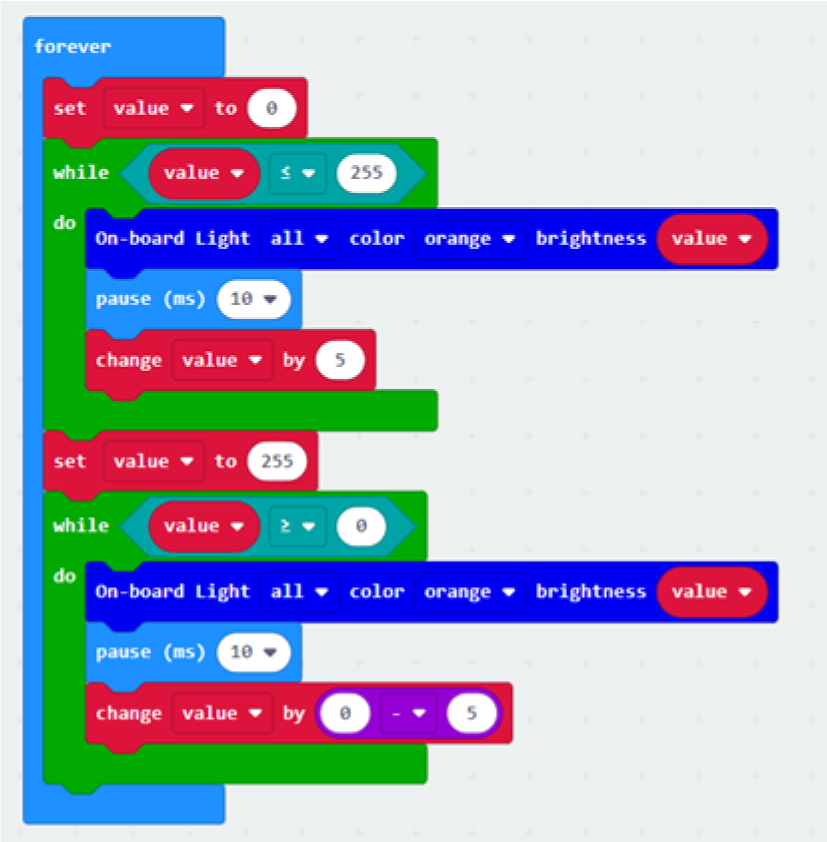
Exercise 1:



Exercise 2: Breathing light

```
a.        Changing the color of all the light into orangebrightness of the light will␣
↪change as ‘value’ change.
b.        Stop for 0.01 second
c.        ‘value’ ADD  5
```

```
a.        Changing the color of all the light into orangebrightness of the light will␣
↪change as’ value’ change.
b.        stop for 0.01 second
c.        ‘value’ SUBTRACT 5
d.        Program will end when’ value’ drop to 0.
```

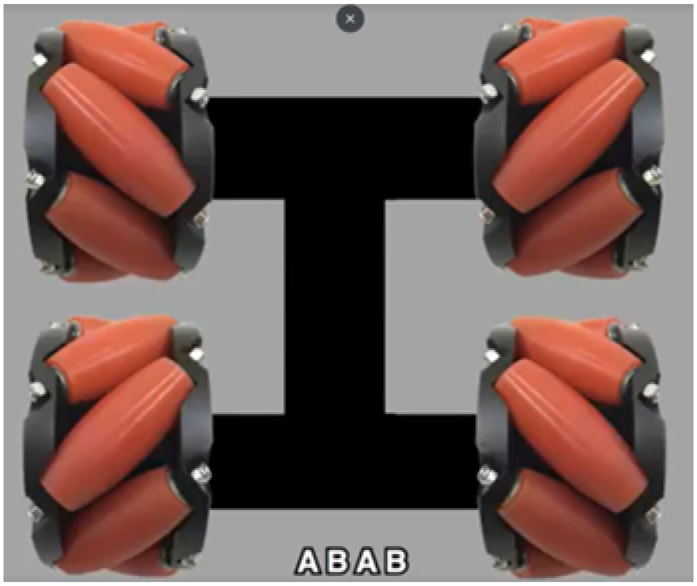

forever
set value to 0
while value ≤ 255
do
On-board Light all color orange brightness value
pause (ms) 10
change value by 5
set value to 255
while value ≥ 0
do
On-board Light all color orange brightness value
pause (ms) 10
change value by 0 - 5

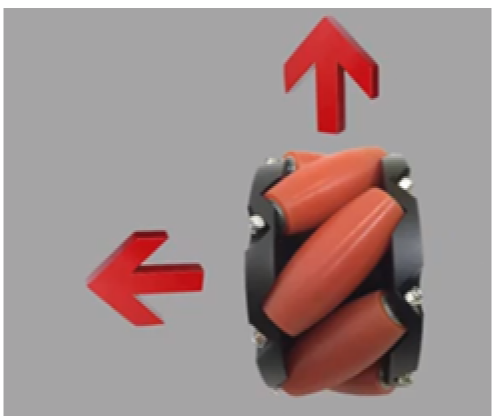
1.3.3 Lesson 3

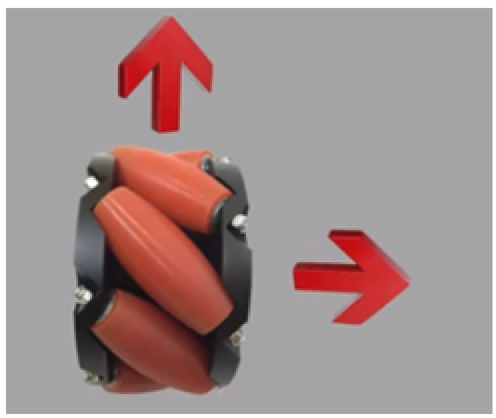
Introduction

Teaching Objectives

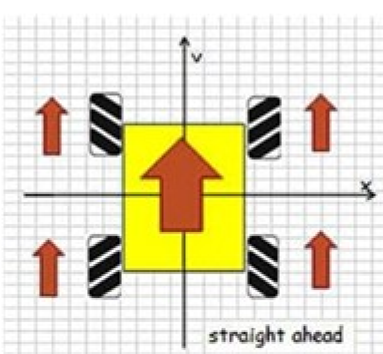
Understanding the Mecanum wheels

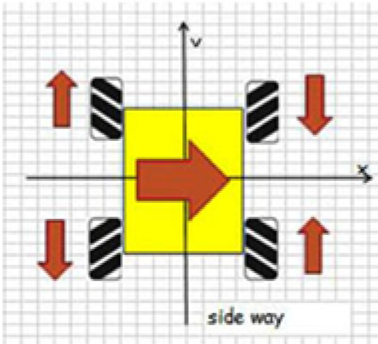


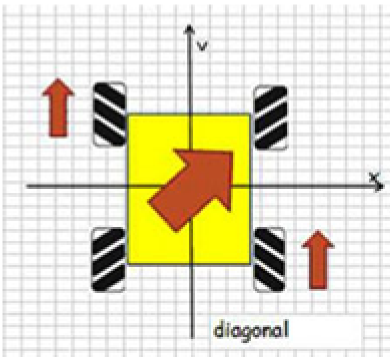


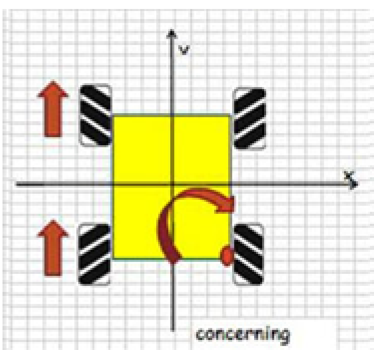


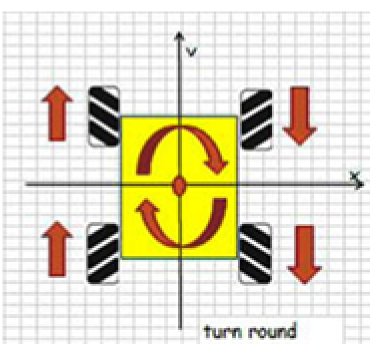
Principal of the wheels

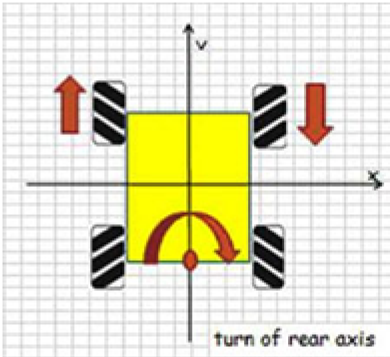



y
x
side way


y
x
diagonal

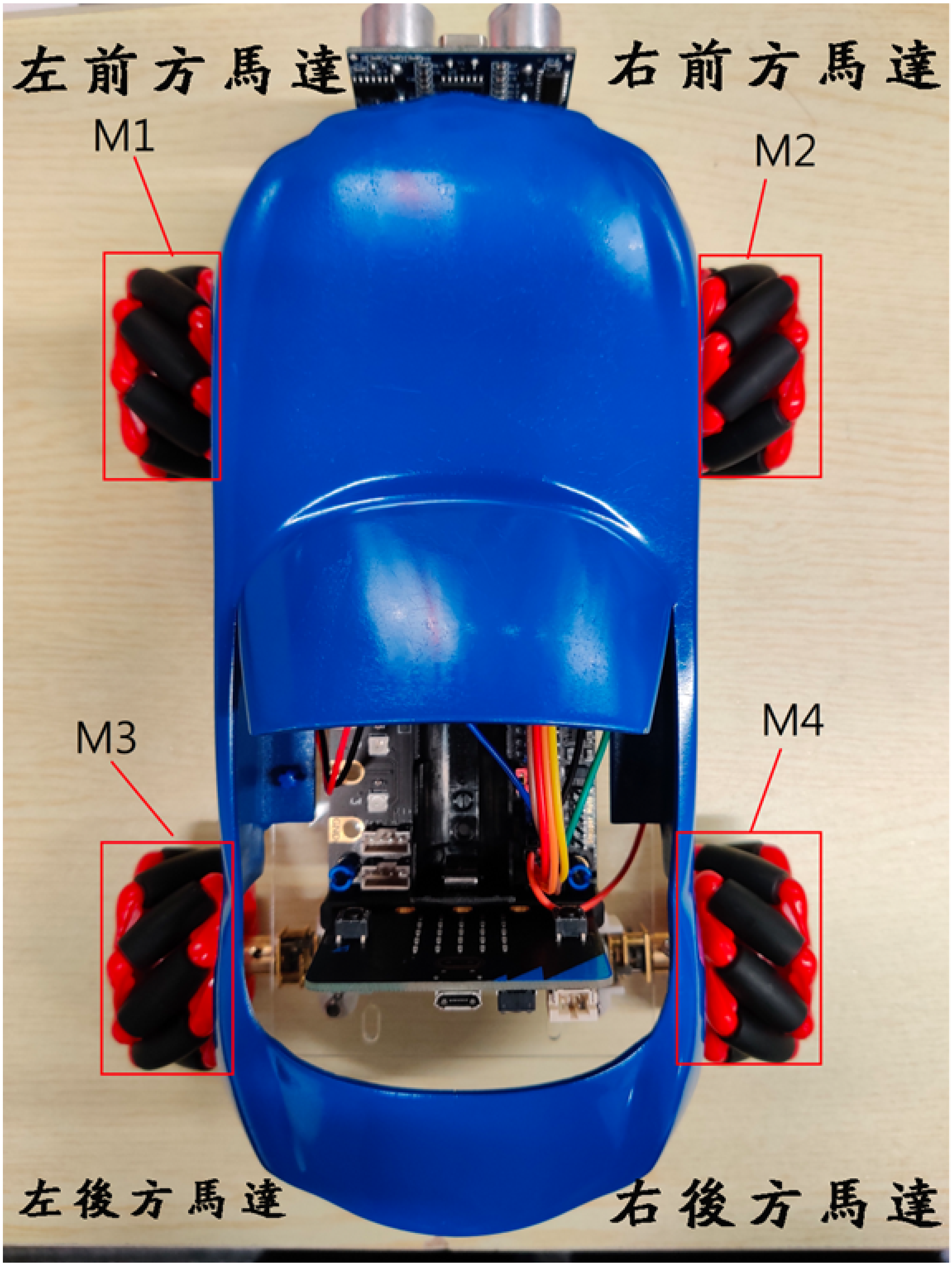

y
x
concerning


y
x
turn round

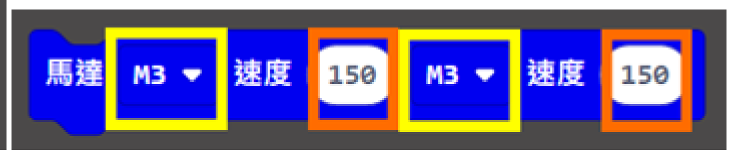


1.3.4 Lesson 4

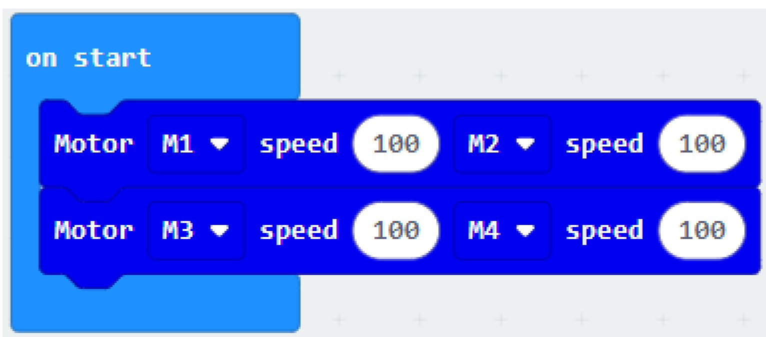
Movement of Car



Block module that control motor

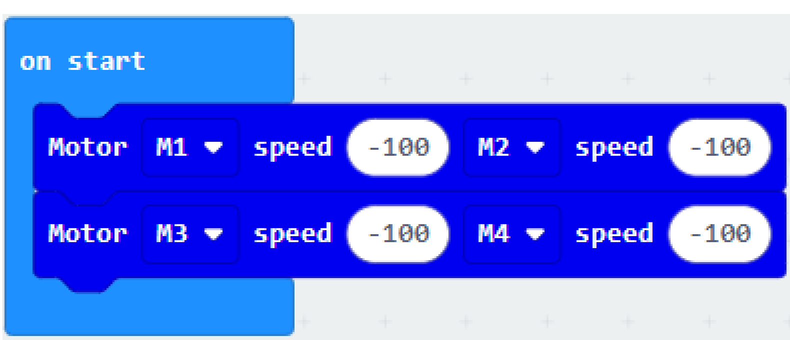


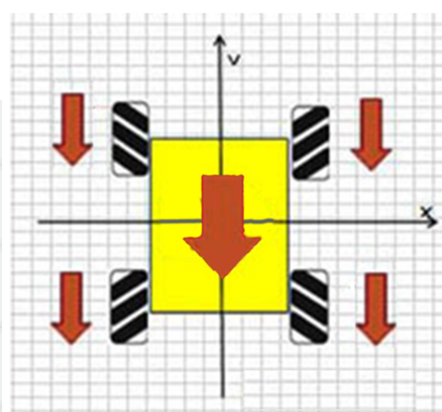
Forward



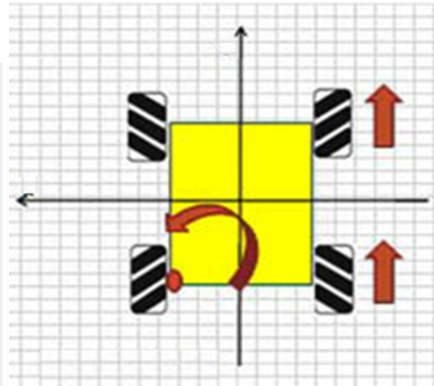
microbitcar/pic/4/4_5.png

Backward

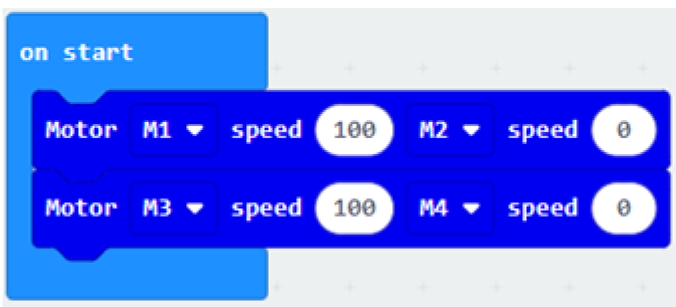


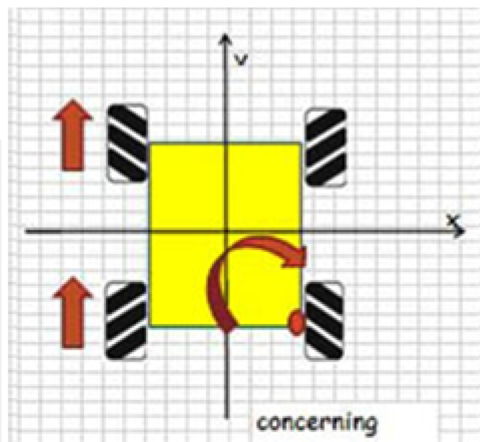


Turn to the left

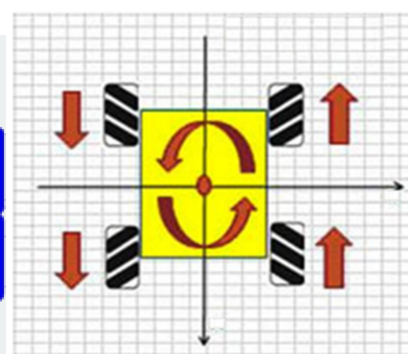


Turn to the right

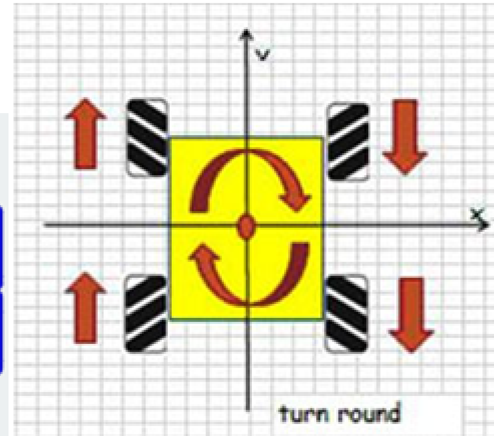




Spinning in anti-clockwise direction

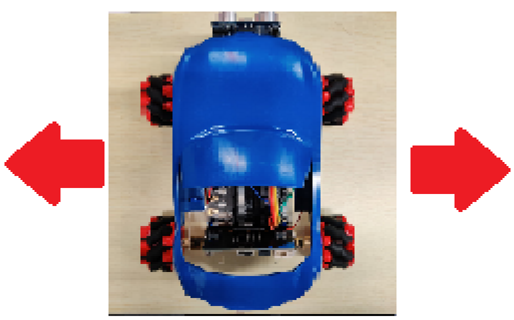


Spinning in clockwise direction

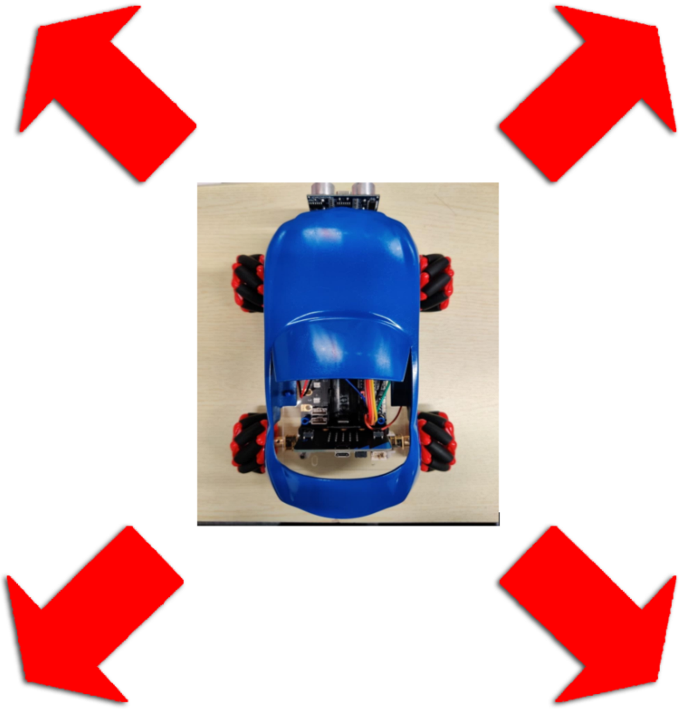


Conclusion

Exercise1: Horizontal movement of the car

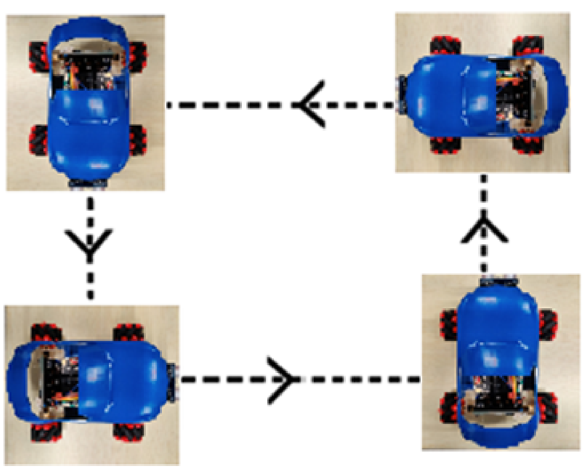


Exercise 2: Diagonal movement of the car

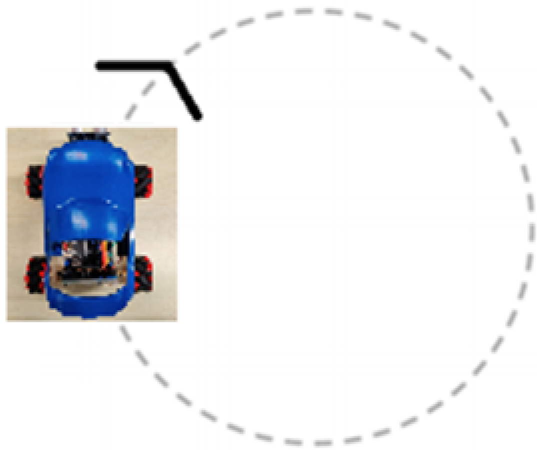


Exercise 3: Different ways to drive the car

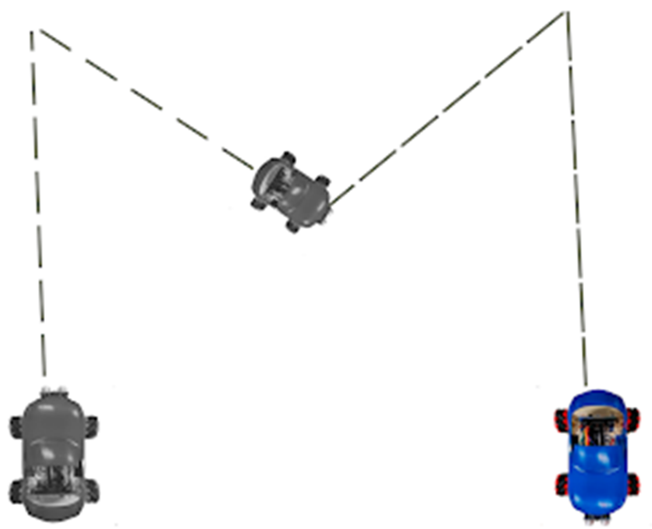
Moving around in a square:



Moving around in a circle:

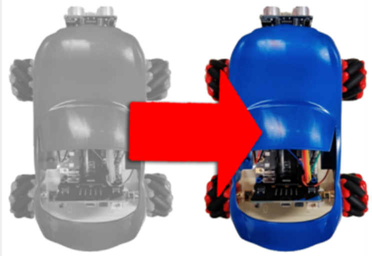


M shape route driving



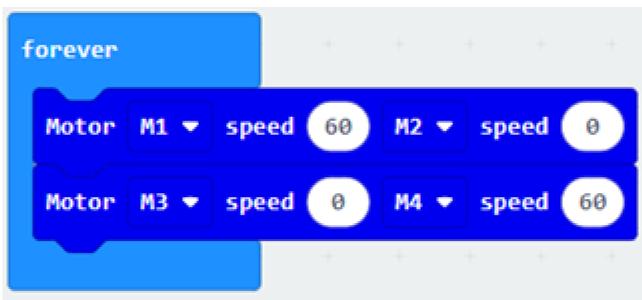
Answer

Exercise 1:

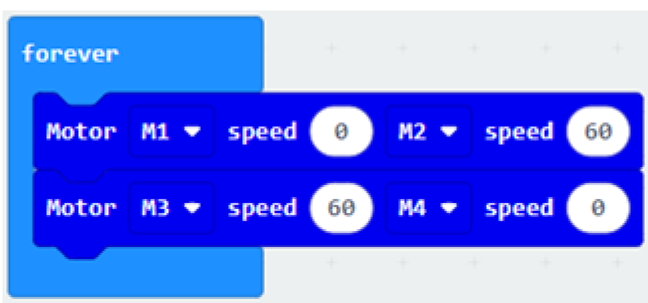


Exercise 2:

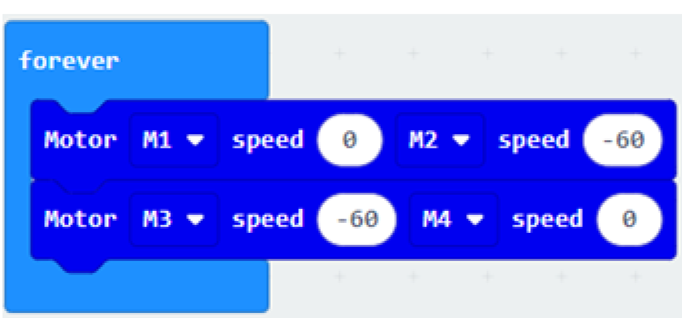
Moving toward right upper corner:



Moving toward left upper corner:



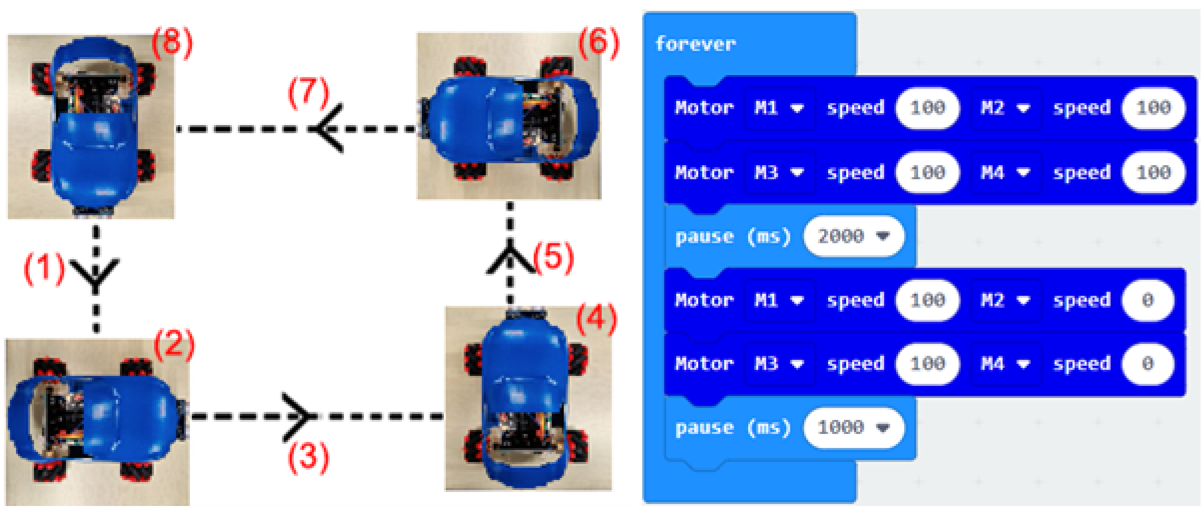
Moving toward right lower corner:



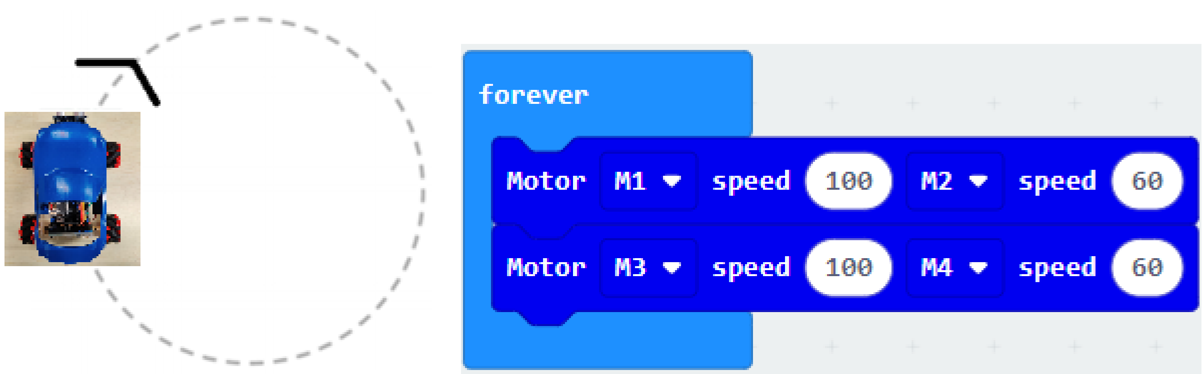
Moving toward left lower corner:

Exercise 3:

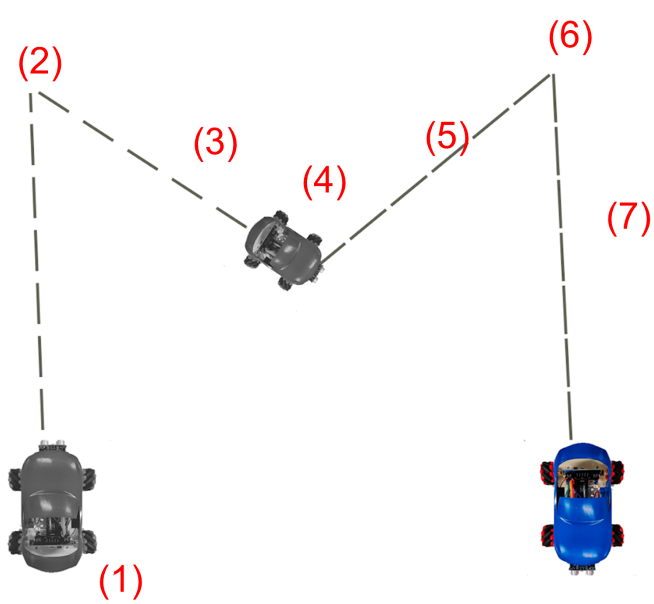
Moving around in a square:



Moving around in a circle:



M shape route driving

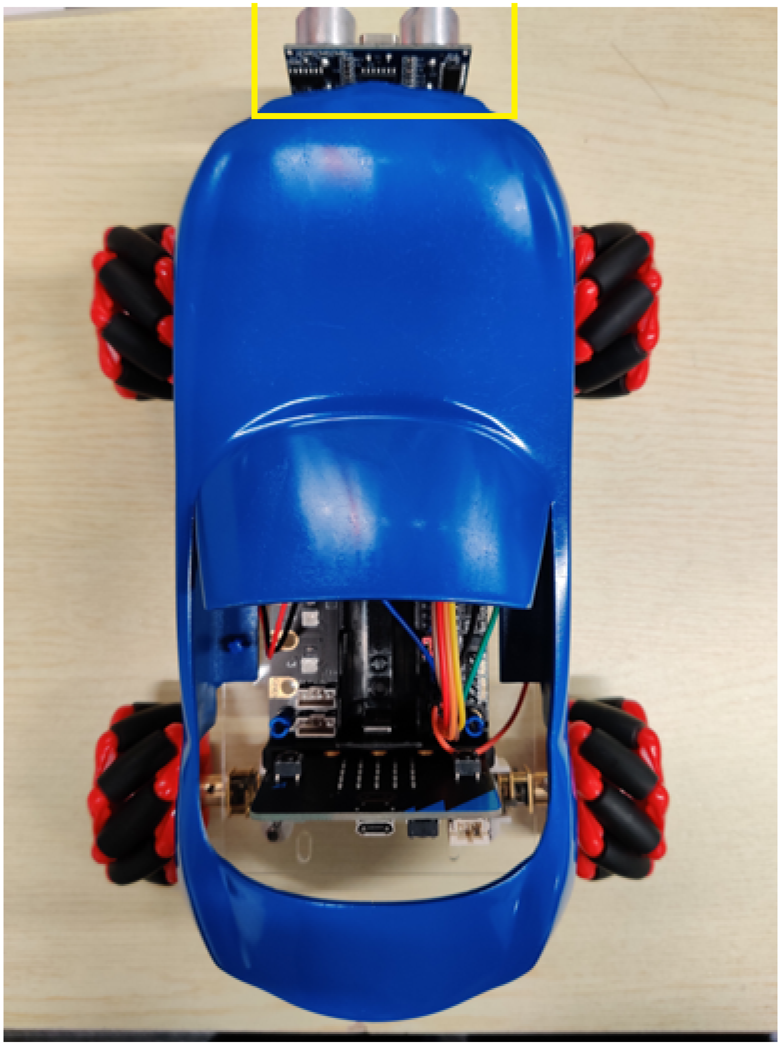


1.3.5 Lesson 5 and Lesson 6

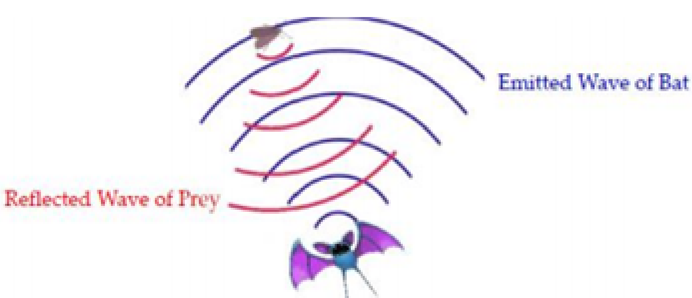
Introduction

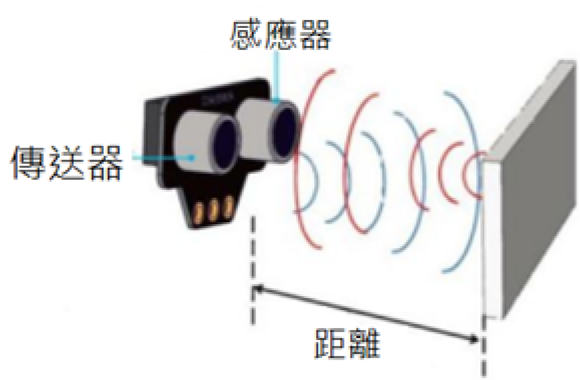
Teaching Objectives

Ultrasonic Obstacle Avoidance

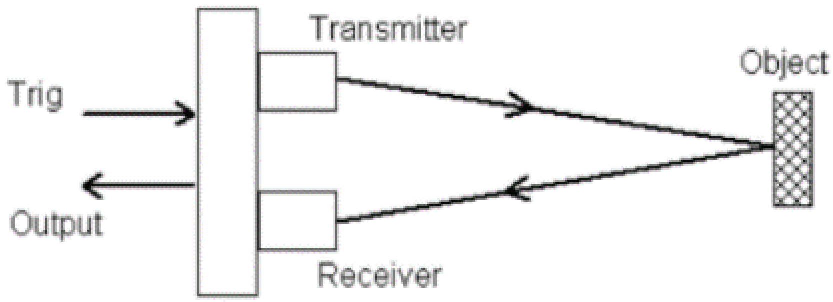


Principal of ultrasonic module

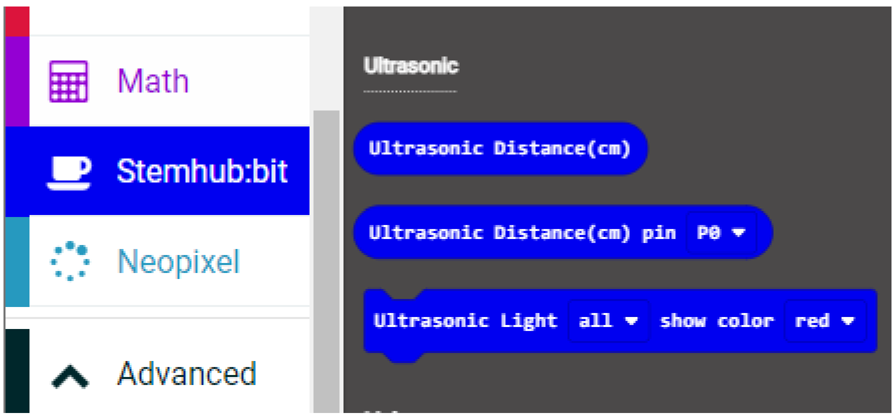


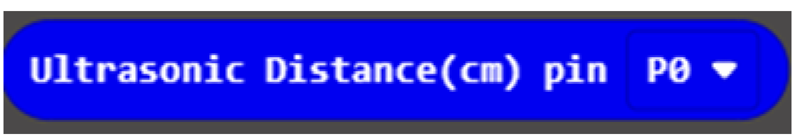


$$\frac{Time \times 0.0346(cm/\mu s)}{2}$$



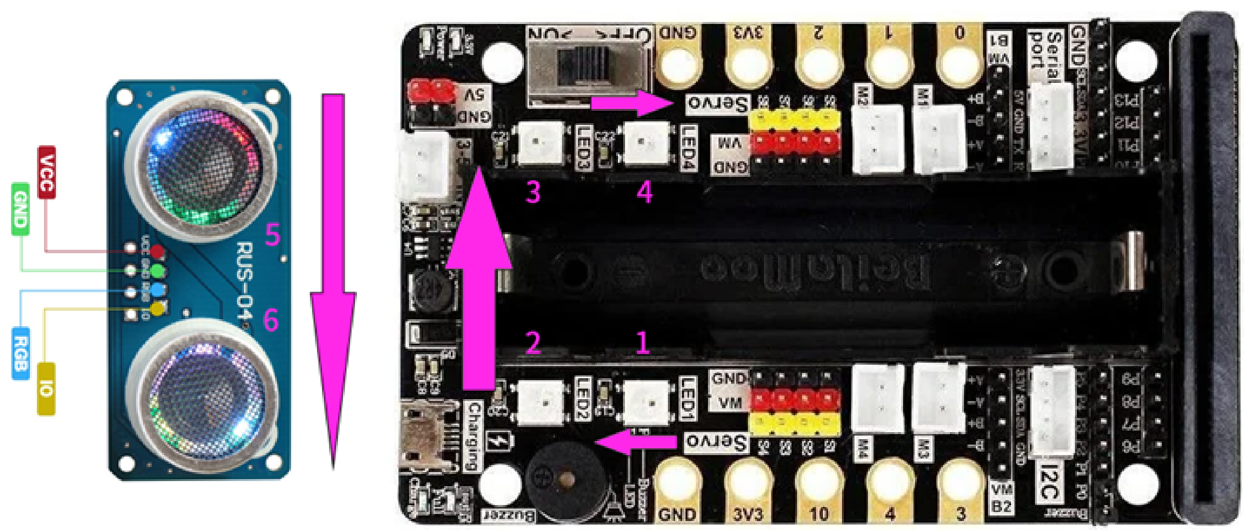
Stemhub: bit Ultrasonic





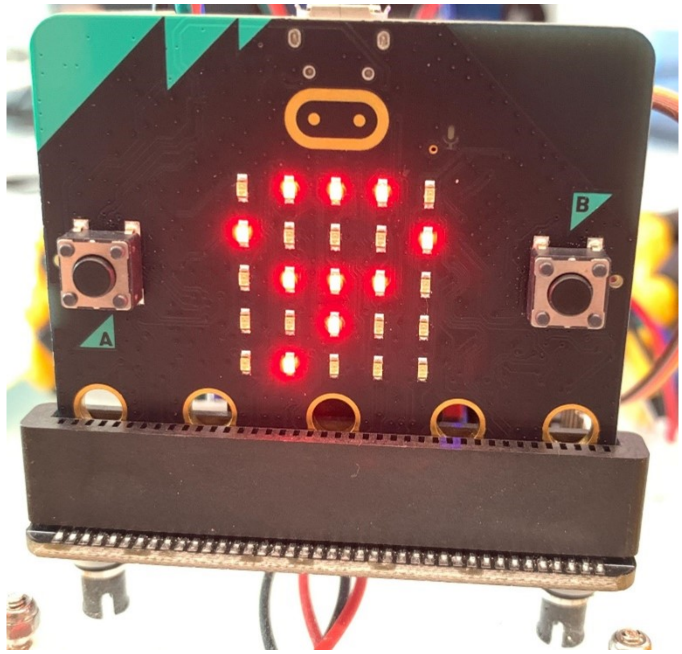

左
右
RUS-04
VCC GND RGB IO
VCC
GND
RGB
IO

Exercise 1


VCC
GND
RGB
IO
RUS-04
5
6
1
2
3
4
LED1
LED2
LED3
LED4
Servo
GND
3V3
10
4
3
Serial port
I2C
Charging
Buzzer

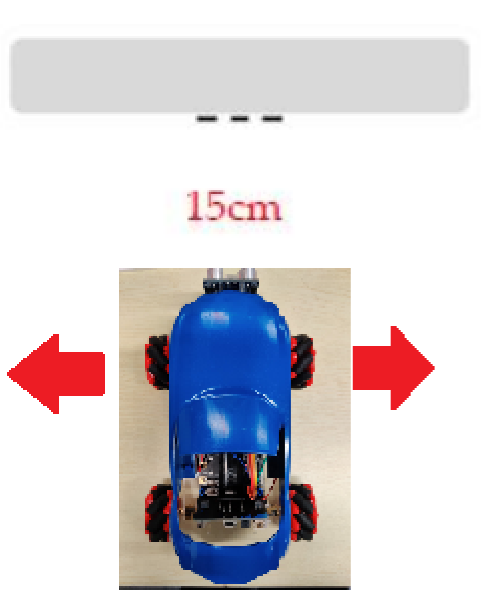

On-board Light one color red brightness 255

Exercise 2

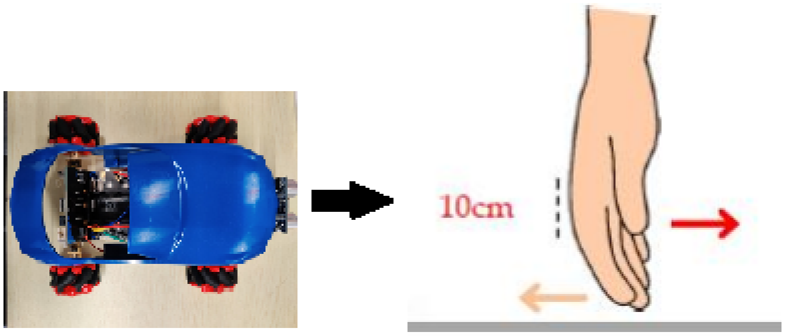


Exercise 3

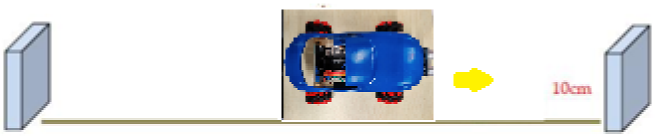
Exercise 4



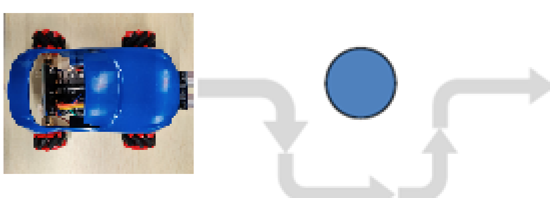
Exercise 5



Exercise 6

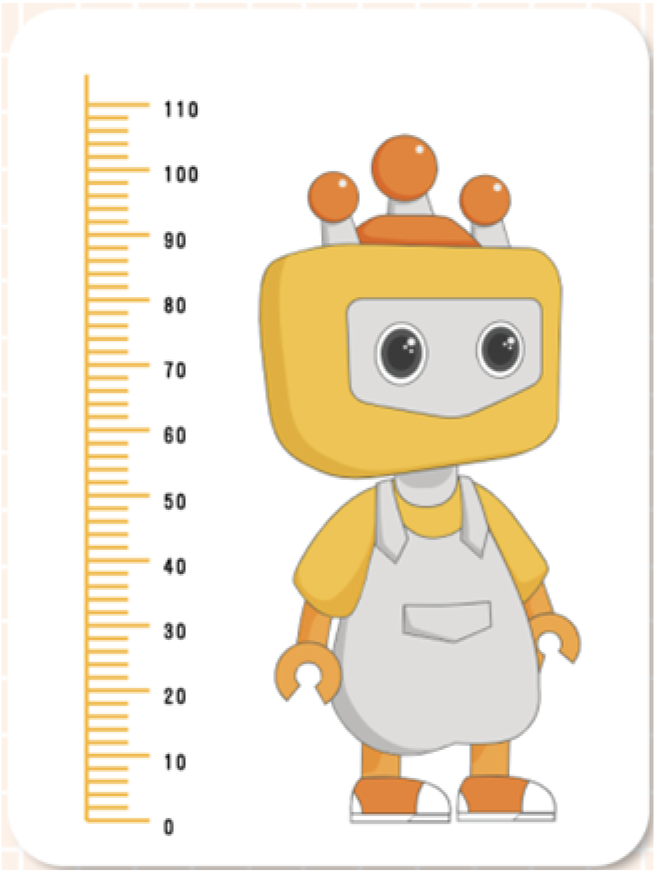


Exercise 7



Exercise 8

Think about it

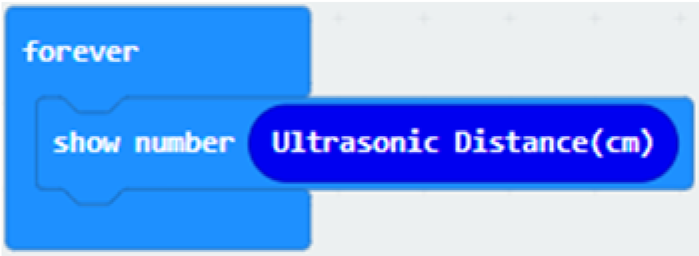


Answer

Exercise 1

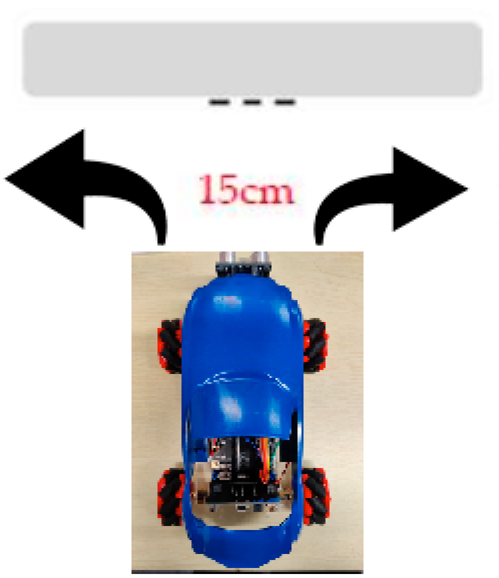

forever
On-board Light one color blue brightness 255
pause (ms) 100
On-board Light one color blue brightness 0
On-board Light two color blue brightness 255
pause (ms) 100
On-board Light two color blue brightness 0
On-board Light three color blue brightness 255
pause (ms) 100
On-board Light three color blue brightness 0
On-board Light four color blue brightness 255
pause (ms) 100
On-board Light four color blue brightness 0
Ultrasonic Light left show color blue
pause (ms) 100
Ultrasonic Light left show color black
Ultrasonic Light right show color blue
pause (ms) 100
Ultrasonic Light right show color black

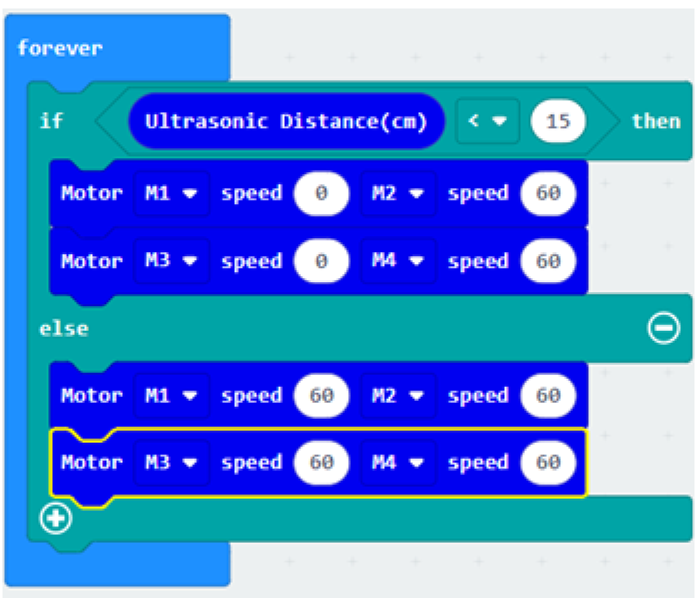
Exercise 2

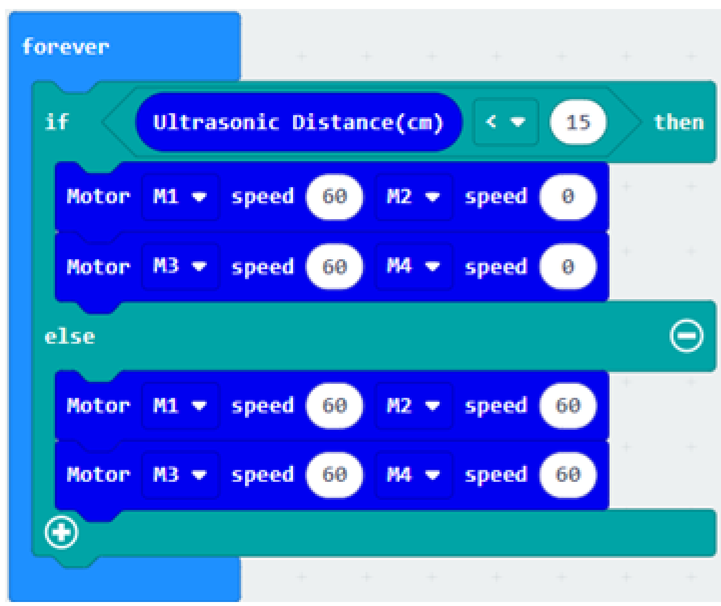


Exercise 3

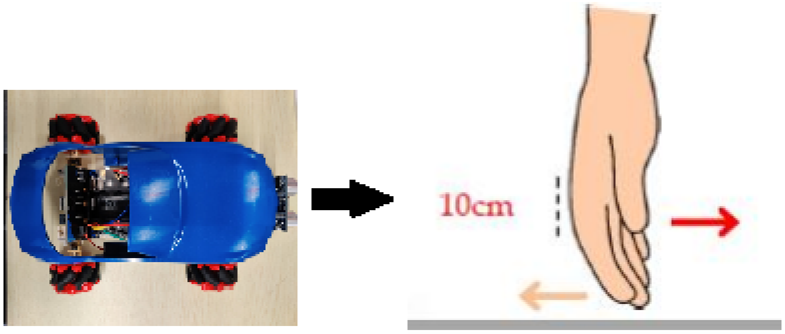
Exercise 4


15cm

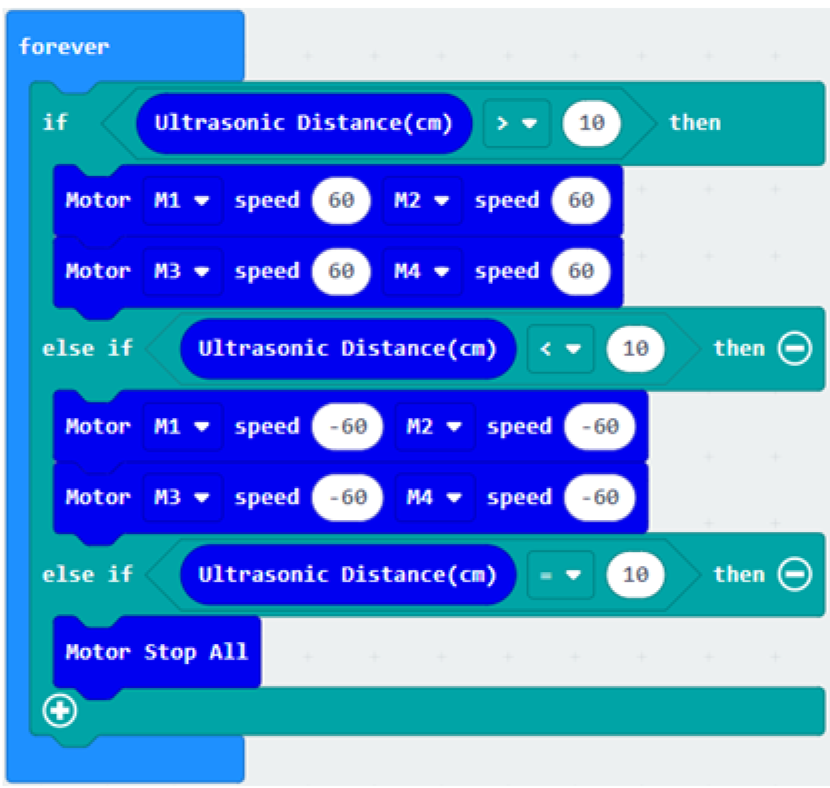

forever
if Ultrasonic Distance(cm) < 15 then
Motor M1 speed 0 M2 speed 60
Motor M3 speed 0 M4 speed 60
else
Motor M1 speed 60 M2 speed 60
Motor M3 speed 60 M4 speed 60


forever
if Ultrasonic Distance(cm) < 15 then
Motor M1 speed 60 M2 speed 0
Motor M3 speed 60 M4 speed 0
else
Motor M1 speed 60 M2 speed 60
Motor M3 speed 60 M4 speed 60

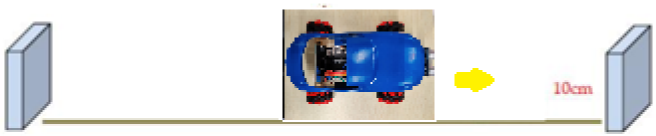
Exercise 5

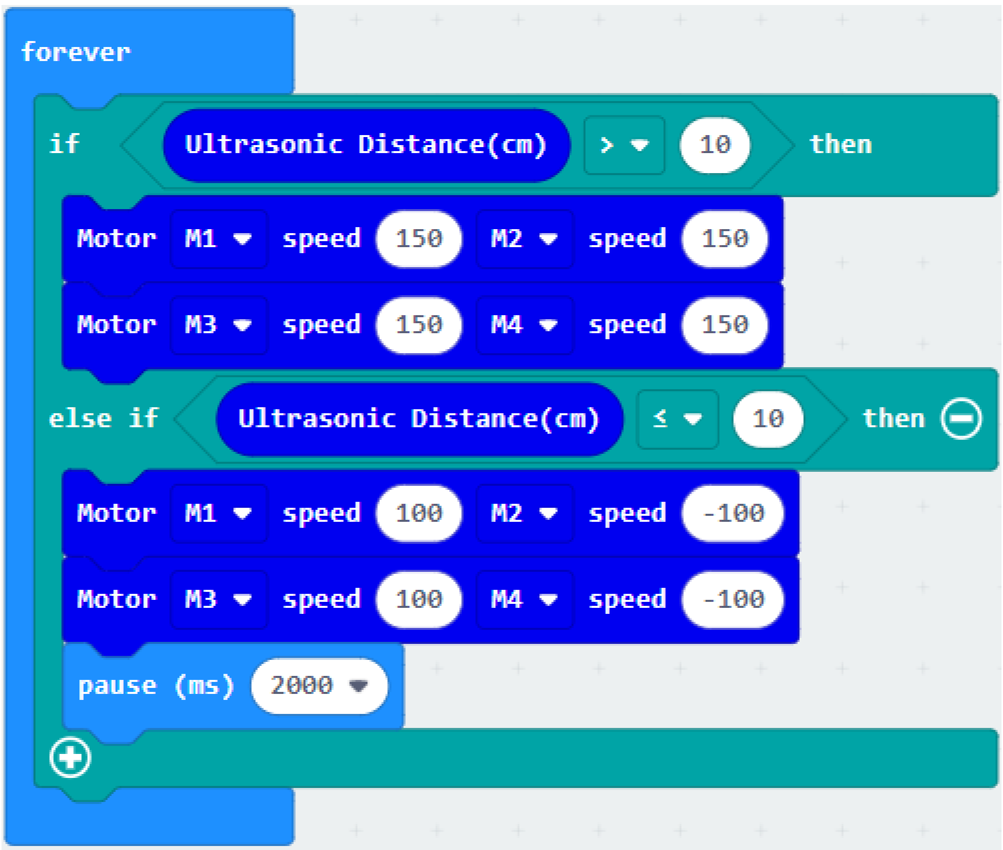



```
When obstaclehandand the car distanced 10cm above, the car will move forward
When obstaclehandand the car distanced 10cm less, the car will move backward
When obstaclehandand the car distanced 10cmthe car will stop there to avoid distancing furthermore.
```

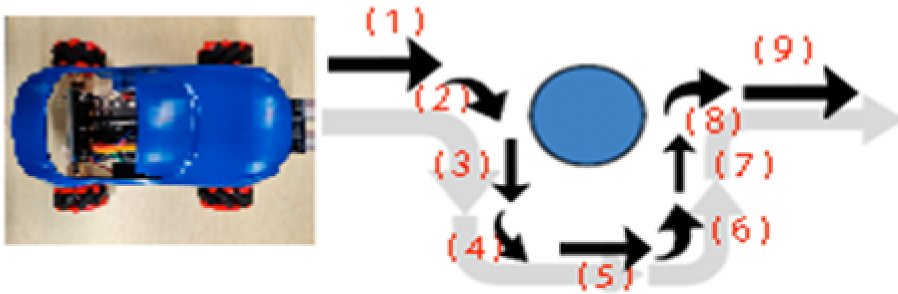


Exercise 6

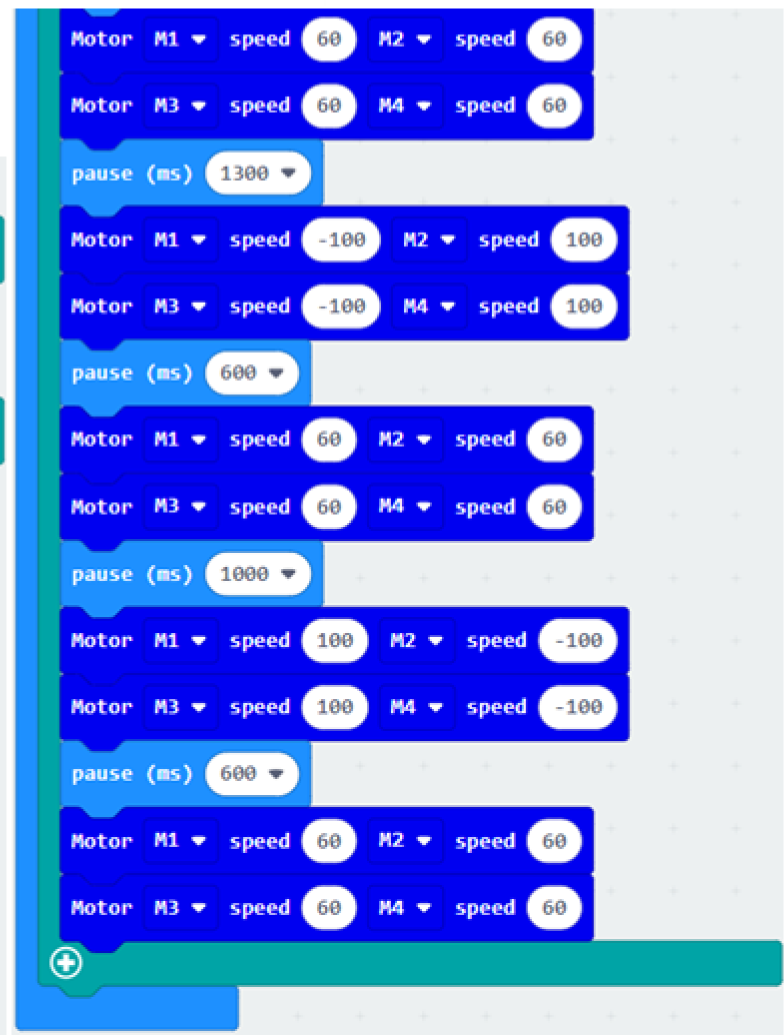

10cm


forever
if Ultrasonic Distance(cm) > 10 then
Motor M1 speed 150 M2 speed 150
Motor M3 speed 150 M4 speed 150
else if Ultrasonic Distance(cm) ≤ 10 then
Motor M1 speed 100 M2 speed -100
Motor M3 speed 100 M4 speed -100
pause (ms) 2000

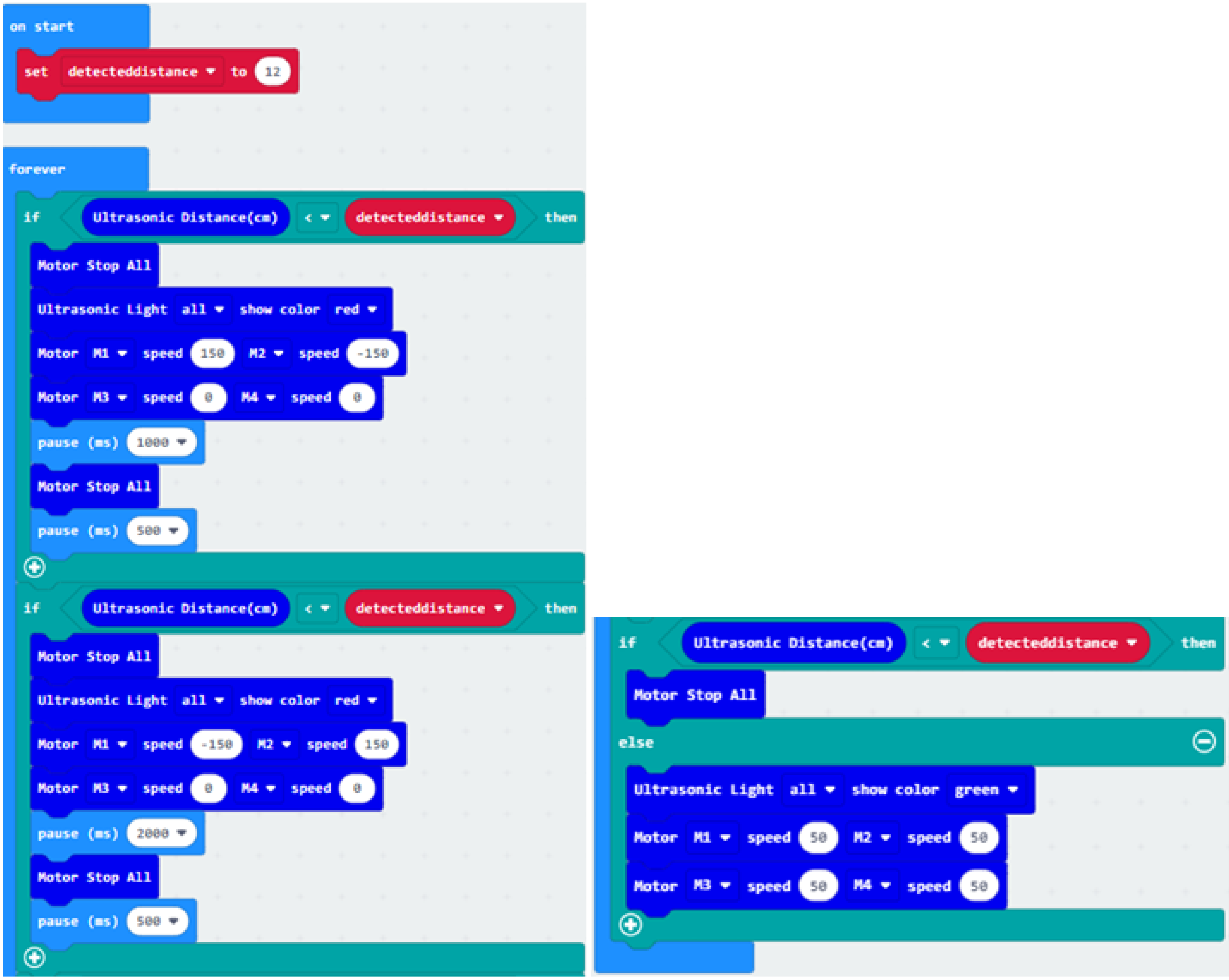
Exercise 7


(1)
(2)
(3)
(4)
(5)
(6)
(7)
(8)
(9)

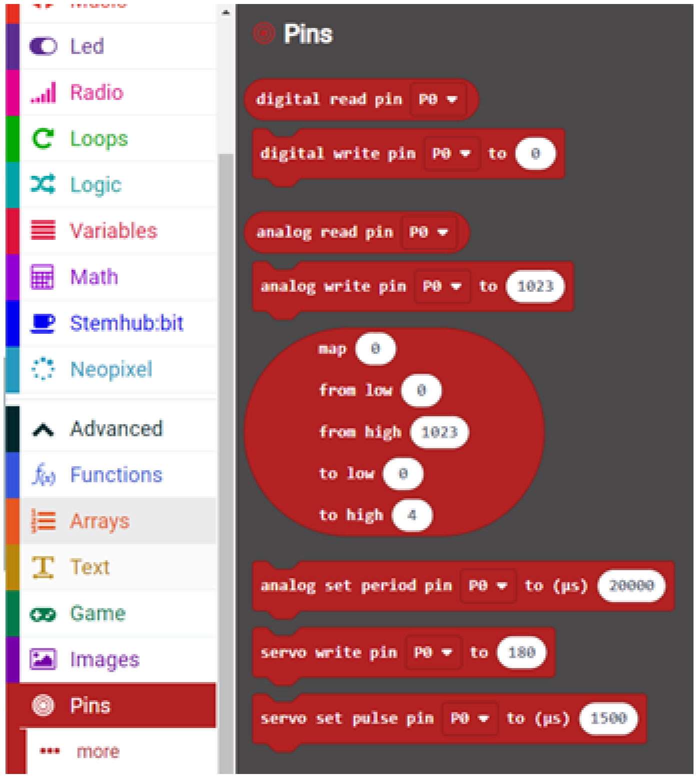

forever
if Ultrasonic Distance(cm) > 12 then
Motor M1 speed 60 M2 speed 60
Motor M3 speed 60 M4 speed 60
else if Ultrasonic Distance(cm) ≤ 12 then
Motor Stop All
Motor M1 speed 100 M2 speed -100
Motor M3 speed 100 M4 speed -100
pause (ms) 600
Motor M1 speed 60 M2 speed 60
Motor M3 speed 60 M4 speed 60
pause (ms) 1000
Motor M1 speed -100 M2 speed 100
Motor M3 speed -100 M4 speed 100
pause (ms) 600


Motor M1 speed 60 M2 speed 60
Motor M3 speed 60 M4 speed 60
pause (ms) 1300
Motor M1 speed -100 M2 speed 100
Motor M3 speed -100 M4 speed 100
pause (ms) 600
Motor M1 speed 60 M2 speed 60
Motor M3 speed 60 M4 speed 60
pause (ms) 1000
Motor M1 speed 100 M2 speed -100
Motor M3 speed 100 M4 speed -100
pause (ms) 600
Motor M1 speed 60 M2 speed 60
Motor M3 speed 60 M4 speed 60

Exercise 8


on start
set detecteddistance to 12
forever
if Ultrasonic Distance(cm) < detecteddistance then
Motor Stop All
Ultrasonic Light all show color red
Motor M1 speed 150 M2 speed -150
Motor M3 speed 0 M4 speed 0
pause (ms) 1000
Motor Stop All
pause (ms) 500
if Ultrasonic Distance(cm) < detecteddistance then
Motor Stop All
Ultrasonic Light all show color red
Motor M1 speed -150 M2 speed 150
Motor M3 speed 0 M4 speed 0
pause (ms) 2000
Motor Stop All
pause (ms) 500
if Ultrasonic Distance(cm) < detecteddistance then
Motor Stop All
else
Ultrasonic Light all show color green
Motor M1 speed 50 M2 speed 50
Motor M3 speed 50 M4 speed 50

Appendix


Led
Radio
Loops
Logic
Variables
Math
Stemhub:bit
Neopixel
Advanced
Functions
Arrays
Text
Game
Images
Pins
more
Pins
digital read pin P0
digital write pin P0 to 0
analog read pin P0
analog write pin P0 to 1023
map 0
from low 0
from high 1023
to low 0
to high 4
analog set period pin P0 to (µs) 20000
servo write pin P0 to 180
servo set pulse pin P0 to (µs) 1500


digital write pin P0 to 0


Led
Radio
Loops
Logic
Variables
Math
Stemhub:bit
Neopixel
Advanced
Functions
Arrays
Text
Game
Images
Pins
more
more
on pin P0 pulsed high
pulse duration (µs)
pulse in (µs) pin P0 pulsed high
i2c read number at address 0 of format Int8LE repeated false
i2c write number
at address 0
with value 0
of format Int8LE
repeated false
spi write 0
spi frequency 1000000


pulse in (µs) pin P0 pulsed high

Programming for Ultrasonic Sensor

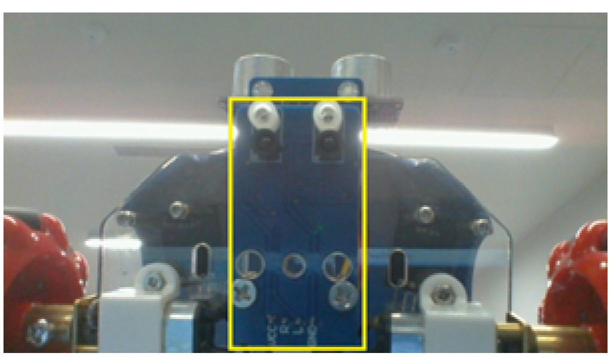
1.3.6 Lesson 7 and Lesson 8

Introduction

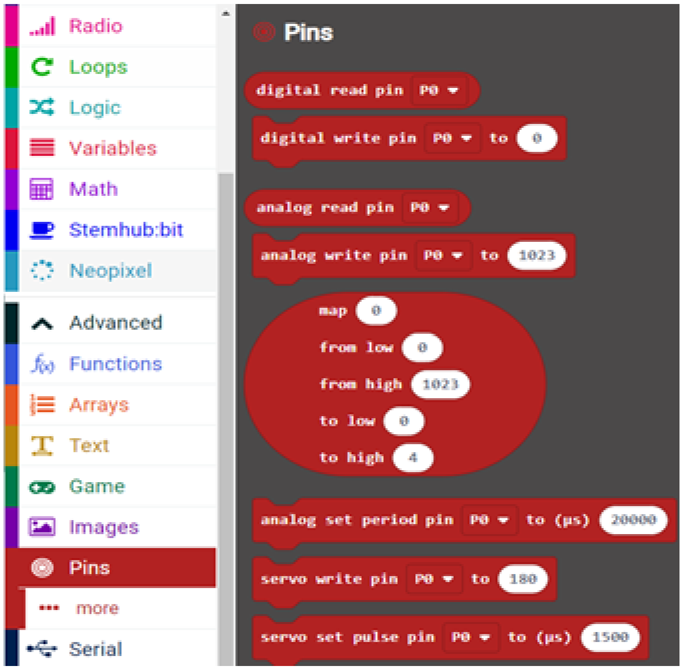
Teaching Objectives

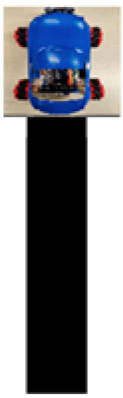
Movement through line tracking

- Infrared Line Tracking sensor

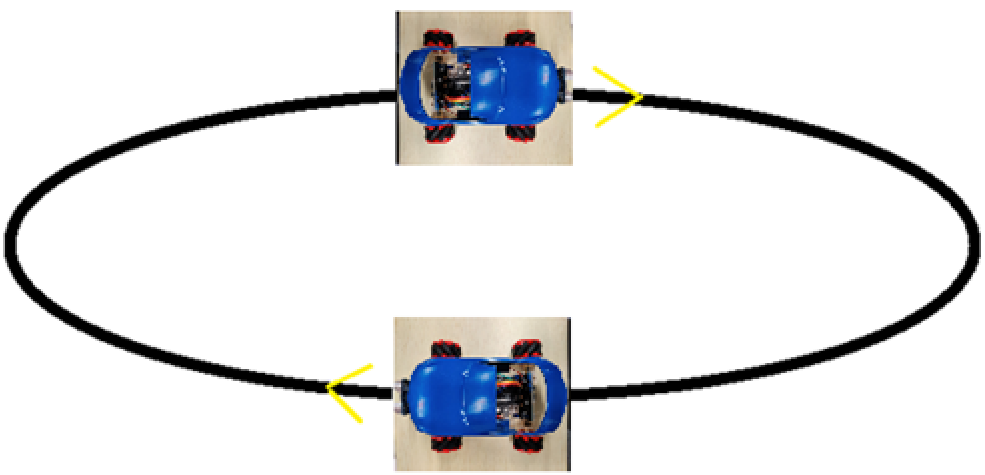



- Line following Basic Module

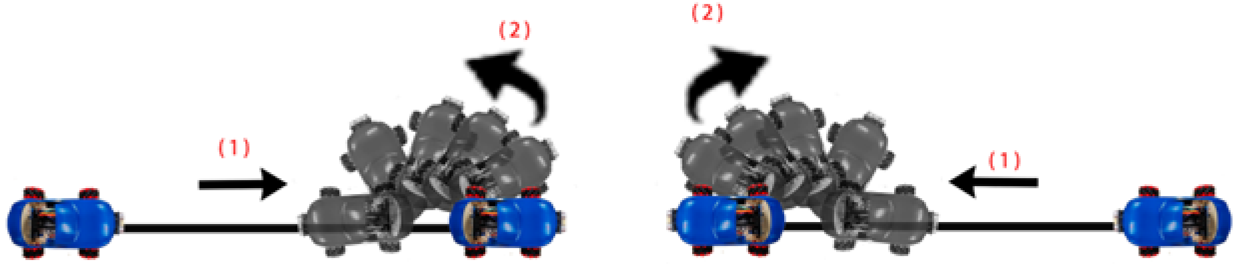




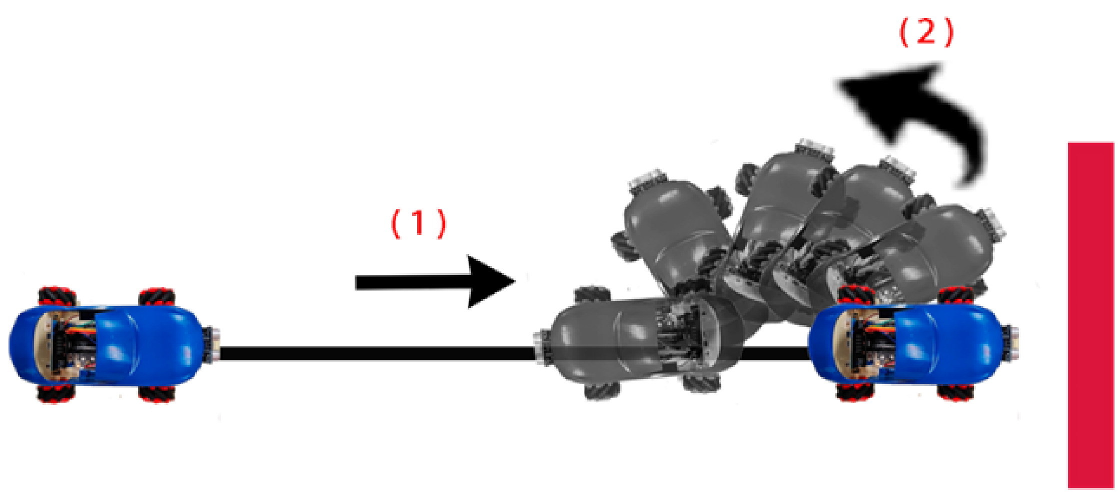
Exercise 1



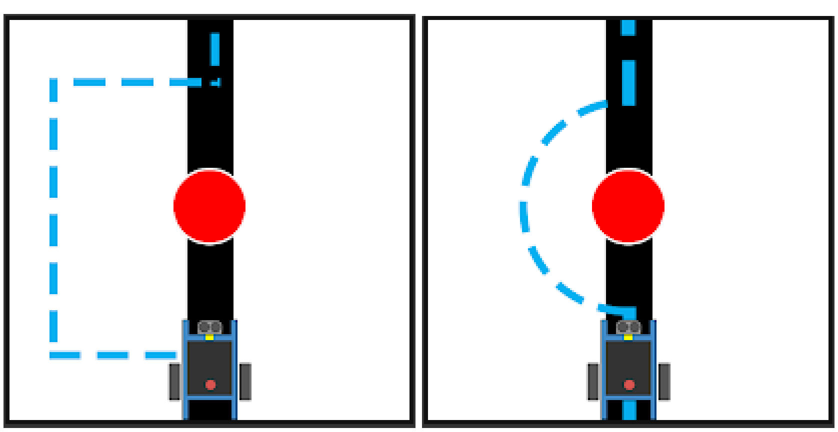
Exercise 2



Exercise 3



Exercise 4Maze

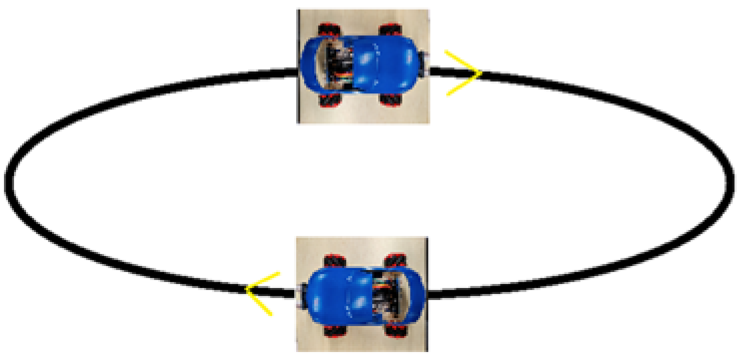


Challenge

- Build Variable "BW_value", think about how to store both digital signal of both left and right sensor at the same time.
- Build Variable "last_value" to store the signal when it is out of bounds and decide the turning direction when out of bounds.

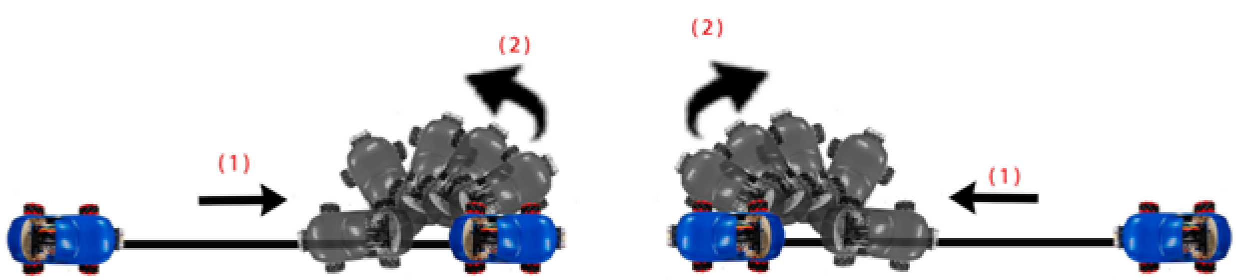
Answer

Exercise 1: Design a program to let the car move following the black line



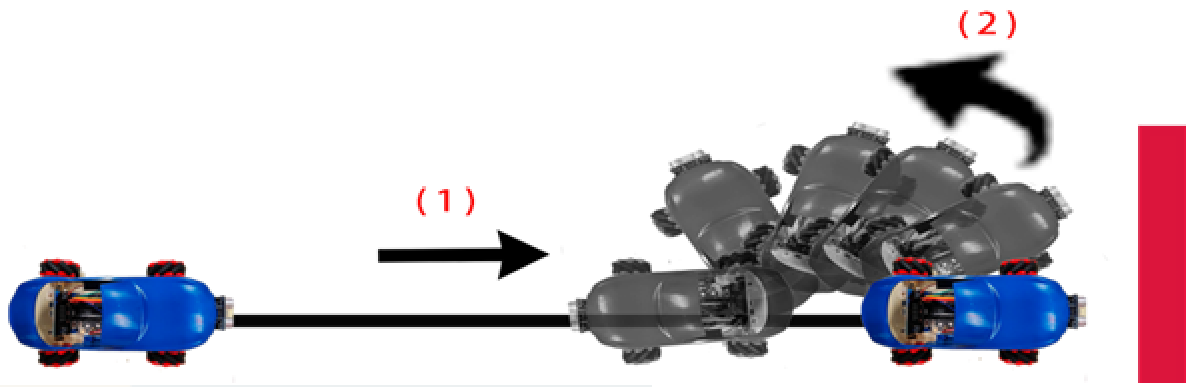

forever
set L to digital read pin P13
set R to digital read pin P14
if L = 0 and R = 0 then
Motor M1 speed 60 M2 speed 60
Motor M3 speed 60 M4 speed 60
else if L = 1 and R = 0 then
Motor M1 speed 80 M2 speed 0
Motor M3 speed 80 M4 speed 0
else if L = 0 and R = 1 then
Motor M1 speed 0 M2 speed 80
Motor M3 speed 0 M4 speed 80
else if L = 1 and R = 1 then
Motor M1 speed -60 M2 speed -60
Motor M3 speed -60 M4 speed -60

Exercise 2:

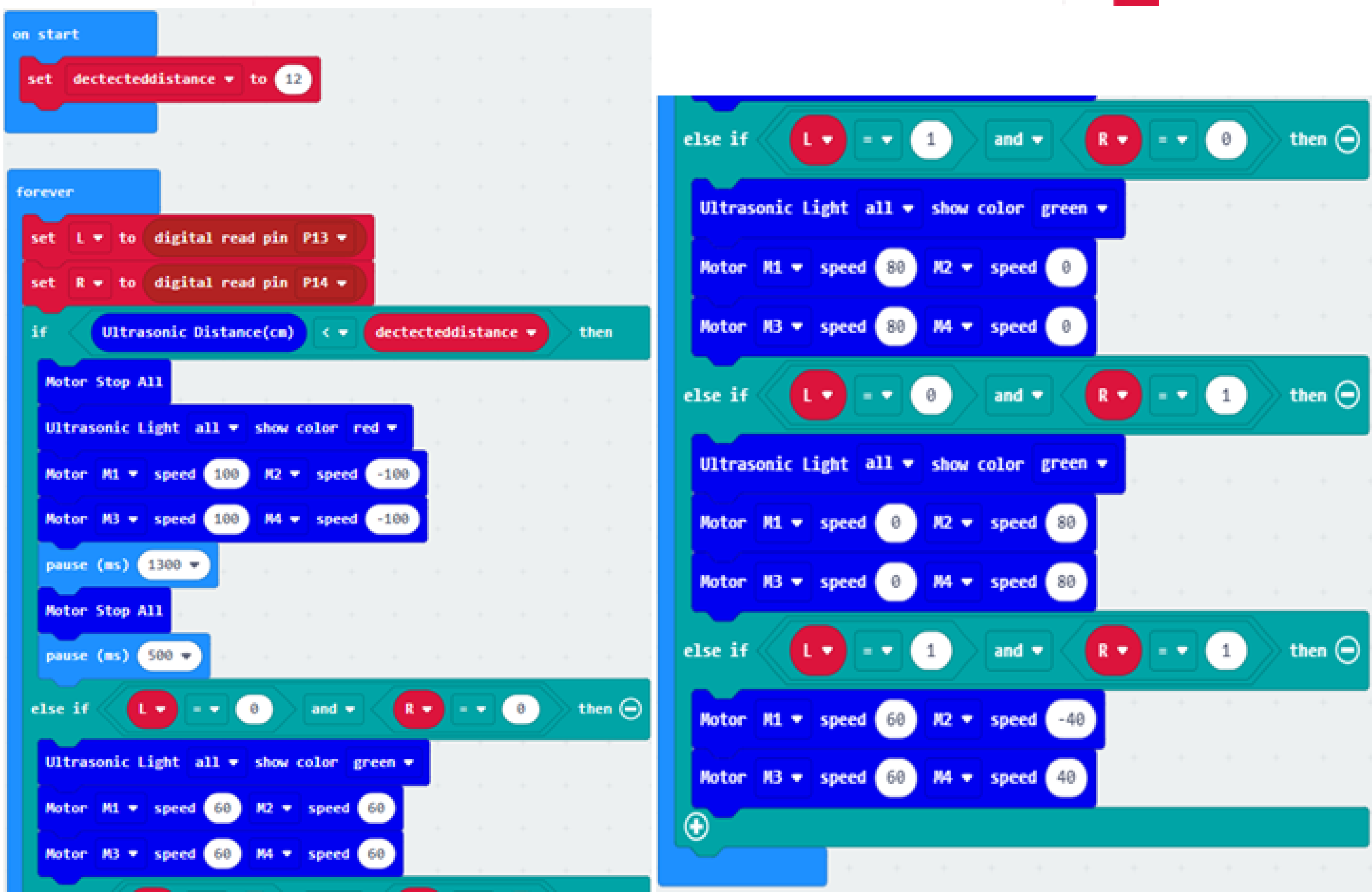

(1)
(2)
(2)
(1)


forever
set L to digital read pin P13
set R to digital read pin P14
if L = 0 and R = 0 then
Motor M1 speed 60 M2 speed 60
Motor M3 speed 60 M4 speed 60
else if L = 1 and R = 0 then
Motor M1 speed 80 M2 speed 0
Motor M3 speed 80 M4 speed 0
else if L = 0 and R = 1 then
Motor M1 speed 0 M2 speed 80
Motor M3 speed 0 M4 speed 80
else if L = 1 and R = 1 then
Motor M1 speed -60 M2 speed -60
Motor M3 speed -60 M4 speed -60

Exercise 3

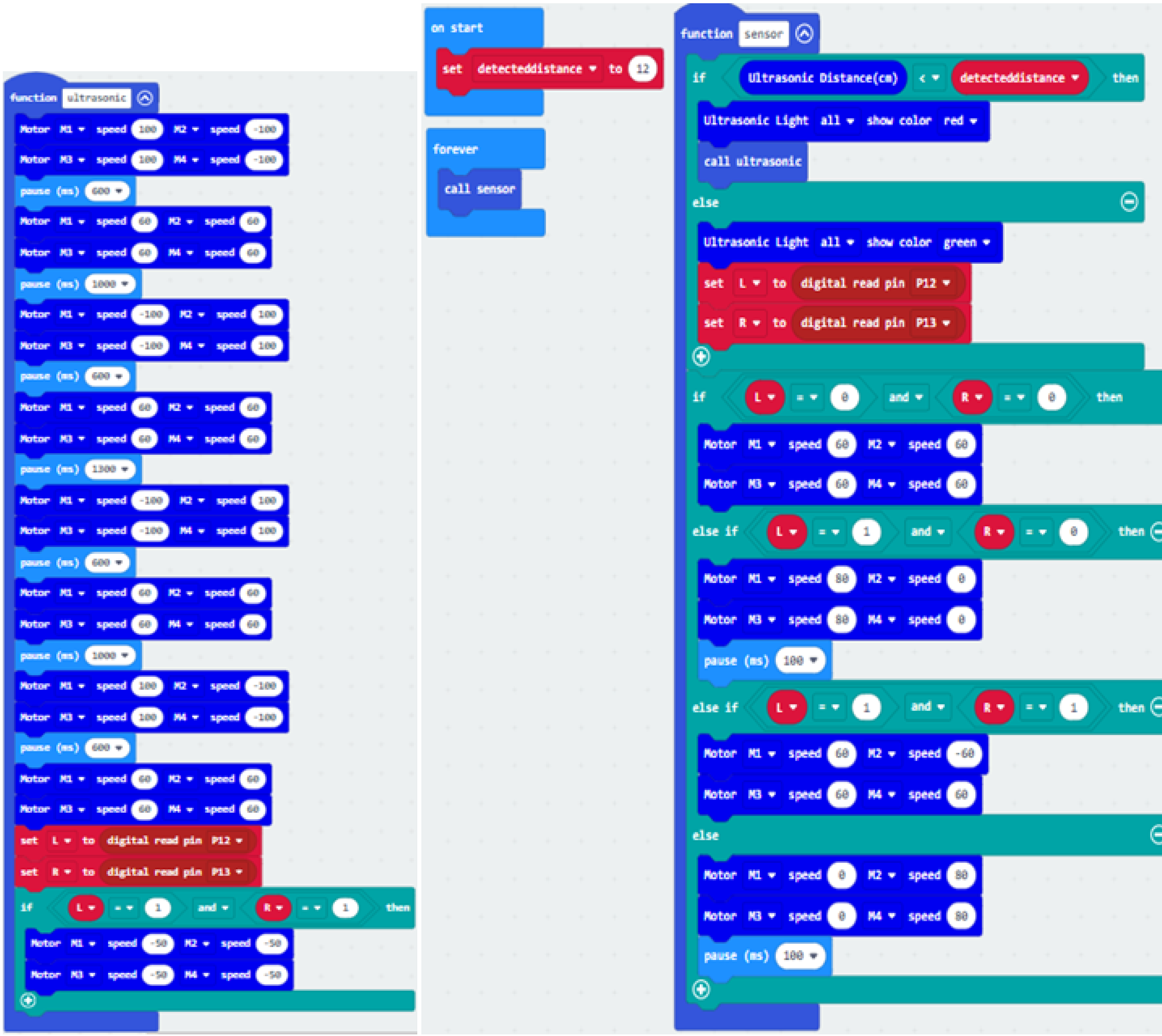


(1)
(2)

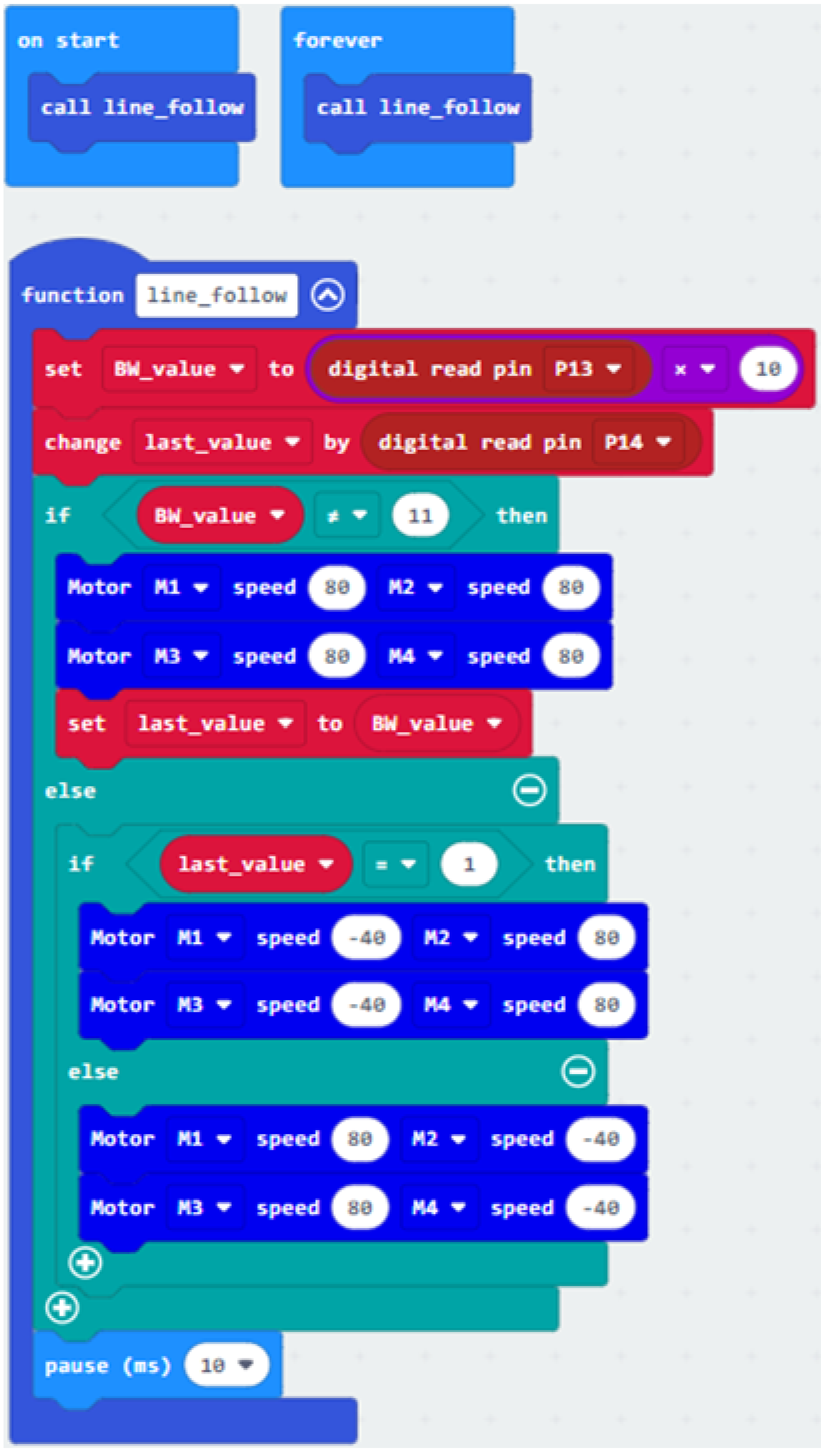


on start
set dectecteddistance to 12
forever
set L to digital read pin P13
set R to digital read pin P14
if Ultrasonic Distance(cm) < dectecteddistance then
Motor Stop All
Ultrasonic Light all show color red
Motor M1 speed 100 M2 speed -100
Motor M3 speed 100 M4 speed -100
pause (ms) 1300
Motor Stop All
pause (ms) 500
else if L = 0 and R = 0 then
Ultrasonic Light all show color green
Motor M1 speed 60 M2 speed 60
Motor M3 speed 60 M4 speed 60
else if L = 1 and R = 0 then
Ultrasonic Light all show color green
Motor M1 speed 80 M2 speed 0
Motor M3 speed 80 M4 speed 0
else if L = 0 and R = 1 then
Ultrasonic Light all show color green
Motor M1 speed 0 M2 speed 80
Motor M3 speed 0 M4 speed 80
else if L = 1 and R = 1 then
Motor M1 speed 60 M2 speed -40
Motor M3 speed 60 M4 speed 40

Exercise 4


function ultrasonic
Motor M1 speed 100 M2 speed -100
Motor M3 speed 100 M4 speed -100
pause (ms) 600
Motor M1 speed 60 M2 speed 60
Motor M3 speed 60 M4 speed 60
pause (ms) 1000
Motor M1 speed -100 M2 speed 100
Motor M3 speed -100 M4 speed 100
pause (ms) 600
Motor M1 speed 60 M2 speed 60
Motor M3 speed 60 M4 speed 60
pause (ms) 1300
Motor M1 speed -100 M2 speed 100
Motor M3 speed -100 M4 speed 100
pause (ms) 600
Motor M1 speed 60 M2 speed 60
Motor M3 speed 60 M4 speed 60
pause (ms) 1000
Motor M1 speed 100 M2 speed -100
Motor M3 speed 100 M4 speed -100
pause (ms) 600
Motor M1 speed 60 M2 speed 60
Motor M3 speed 60 M4 speed 60
set L to digital read pin P12
set R to digital read pin P13
if L = 1 and R = 1 then
Motor M1 speed -50 M2 speed -50
Motor M3 speed -50 M4 speed -50
on start
set detecteddistance to 12
forever
call sensor
function sensor
if Ultrasonic Distance(cm) < detecteddistance then
Ultrasonic Light all show color red
call ultrasonic
else
Ultrasonic Light all show color green
set L to digital read pin P12
set R to digital read pin P13
if L = 0 and R = 0 then
Motor M1 speed 60 M2 speed 60
Motor M3 speed 60 M4 speed 60
else if L = 1 and R = 0 then
Motor M1 speed 80 M2 speed 0
Motor M3 speed 80 M4 speed 0
pause (ms) 100
else if L = 1 and R = 1 then
Motor M1 speed 60 M2 speed -60
Motor M3 speed 60 M4 speed 60
else
Motor M1 speed 0 M2 speed 80
Motor M3 speed 0 M4 speed 80
pause (ms) 100

Challenge

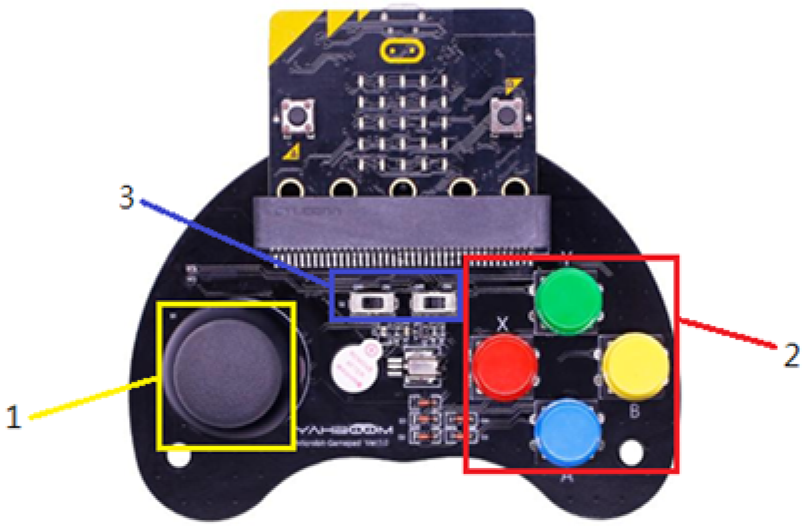

on start
call line_follow
forever
call line_follow
function line_follow
set BW_value to digital read pin P13 × 10
change last_value by digital read pin P14
if BW_value ≠ 11 then
Motor M1 speed 80 M2 speed 80
Motor M3 speed 80 M4 speed 80
set last_value to BW_value
else
if last_value = 1 then
Motor M1 speed -40 M2 speed 80
Motor M3 speed -40 M4 speed 80
else
Motor M1 speed 80 M2 speed -40
Motor M3 speed 80 M4 speed -40
pause (ms) 10

1.3.7 Lesson 9 and Lesson 10

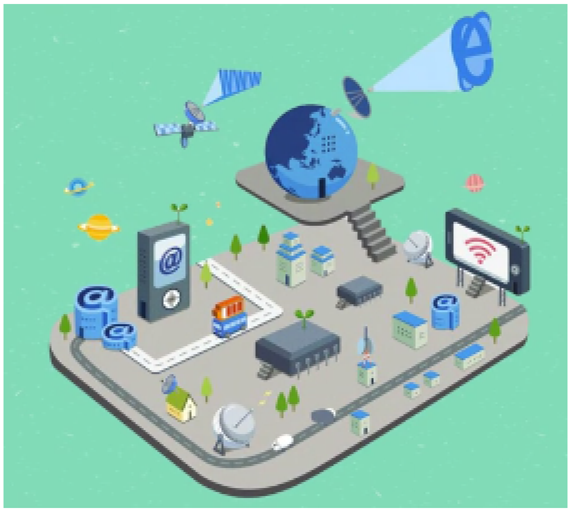
Introduction

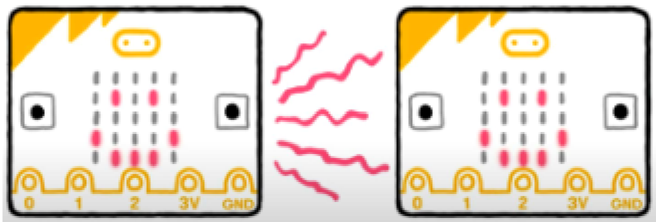
Teaching Objectives

Controller

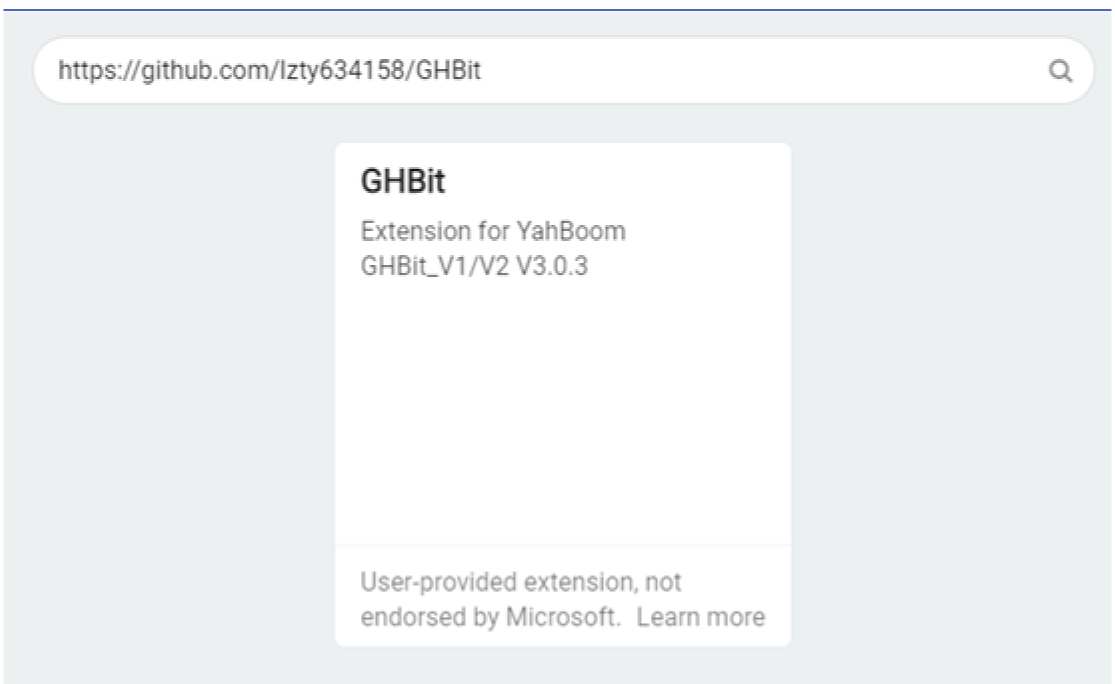


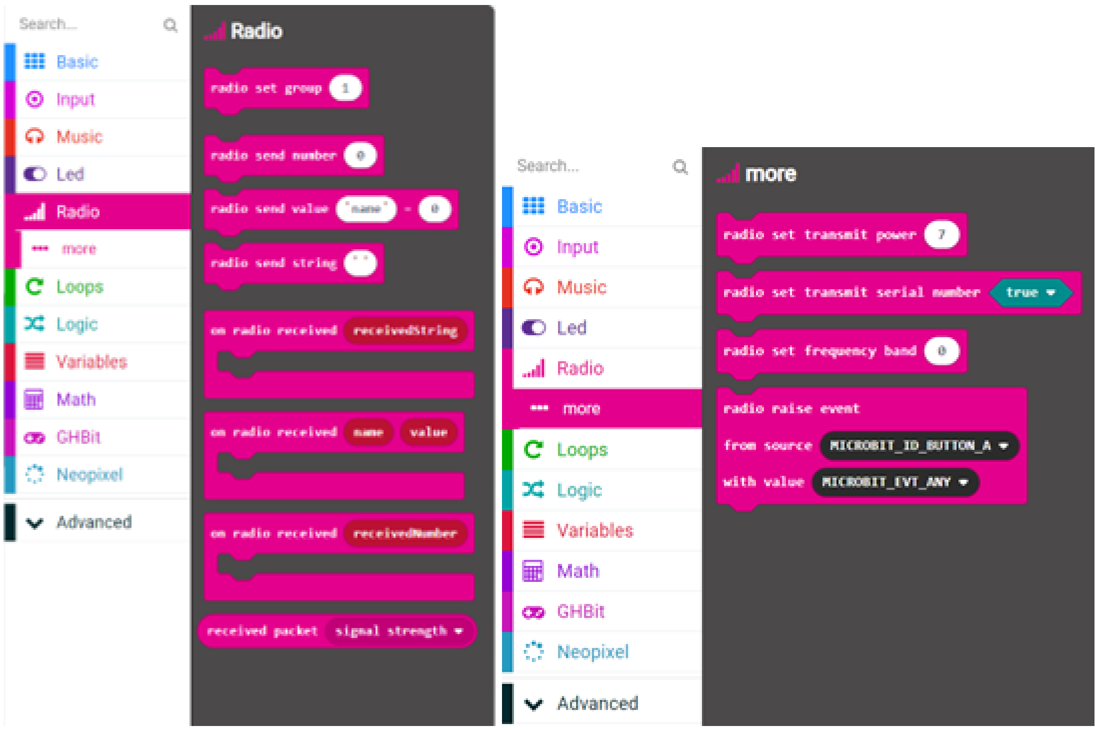
Radio and our daily life



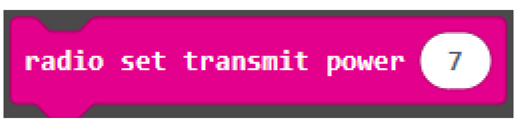


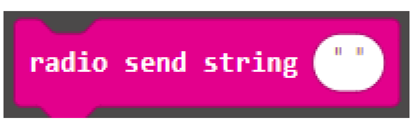
Radio programming module




Search...
Basic
Input
Music
Led
Radio
more
Loops
Logic
Variables
Math
GHBit
Neopixel
Advanced
Radio
radio set group 1
radio send number 0
radio send value "name" = 0
radio send string " "
on radio received receivedString
on radio received name value
on radio received receivedNumber
received packet signal strength
Search...
Basic
Input
Music
Led
Radio
more
Loops
Logic
Variables
Math
GHBit
Neopixel
Advanced
more
radio set transmit power 7
radio set transmit serial number true
radio set frequency band 0
radio raise event
from source MICROBIT_ID_BUTTON_A
with value MICROBIT_EVT_ANY

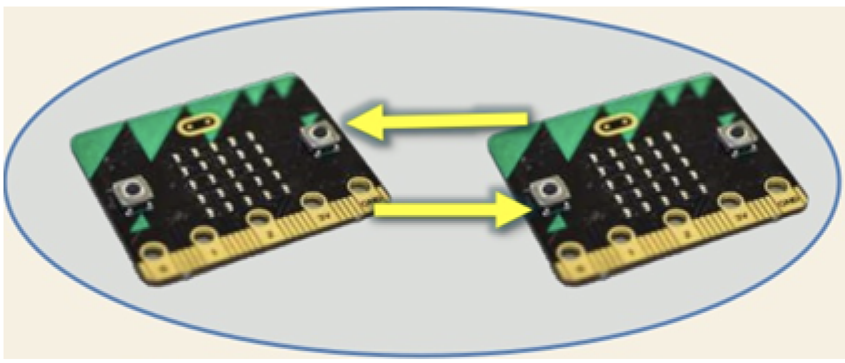

radio set group 1

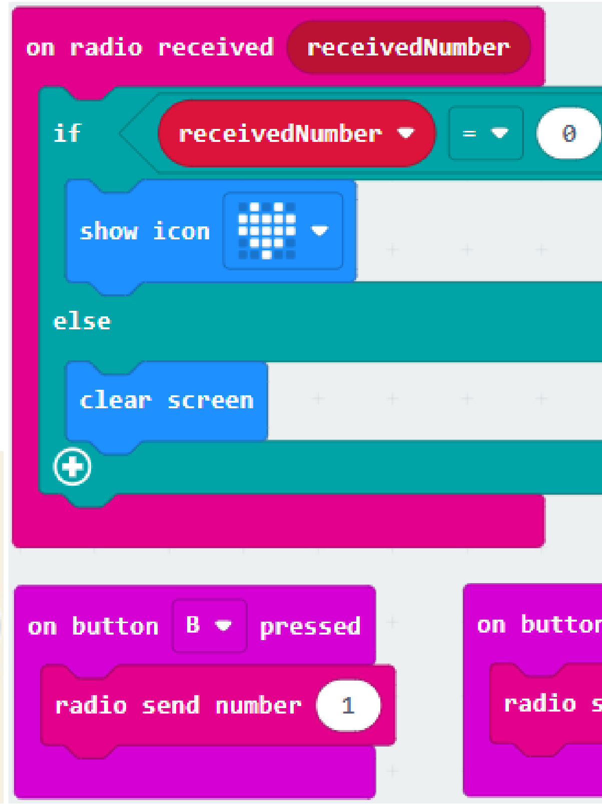

radio set transmit power 7


radio send string " "


on radio received receivedString

Example 1: Two-way radio

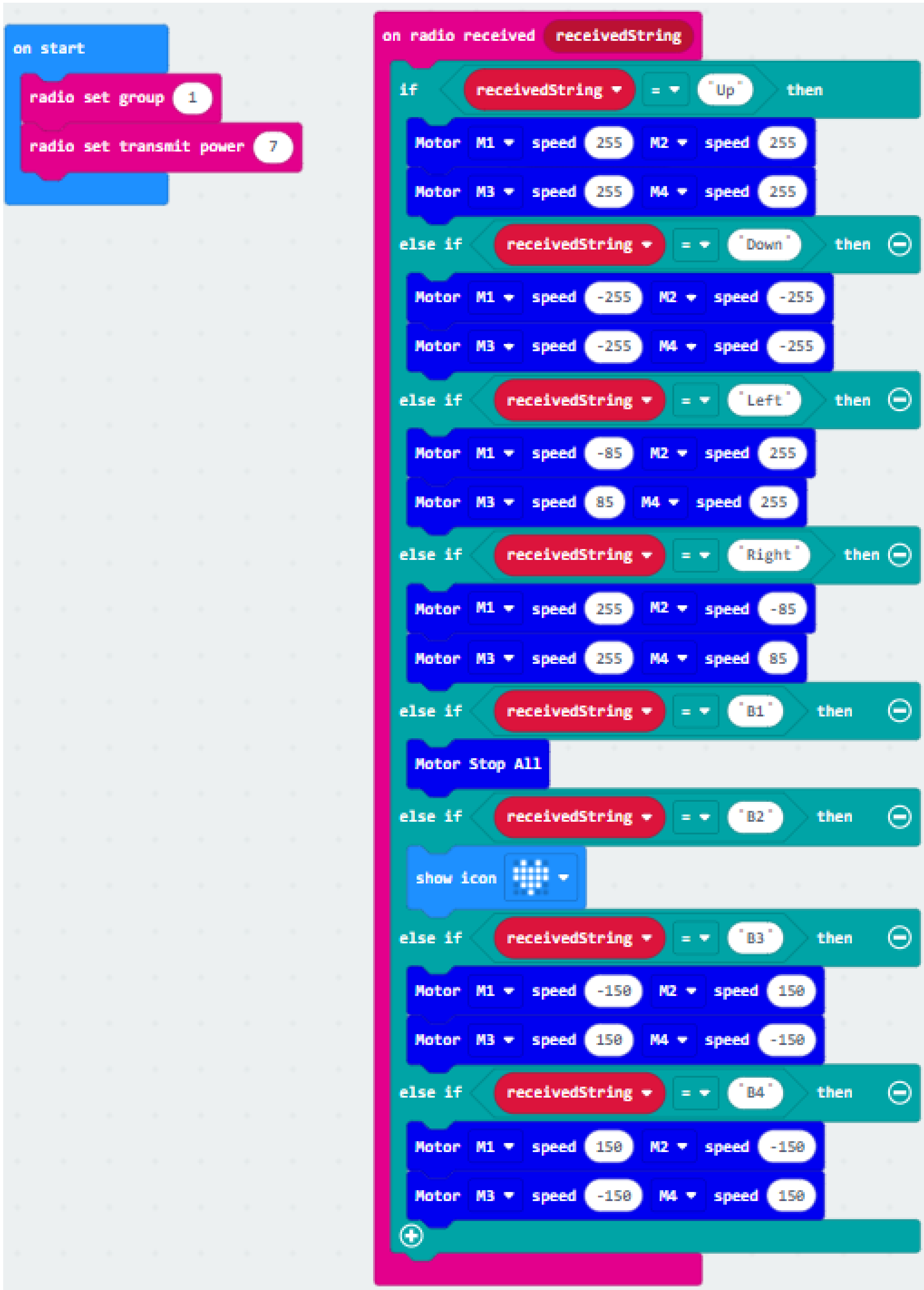




Program module for controller and the car

Programming for controller:(GHBit)


```
B1: stop moving
B2: Display a heart shape
B3: The car moves toward left
B4: The car move toward right
```

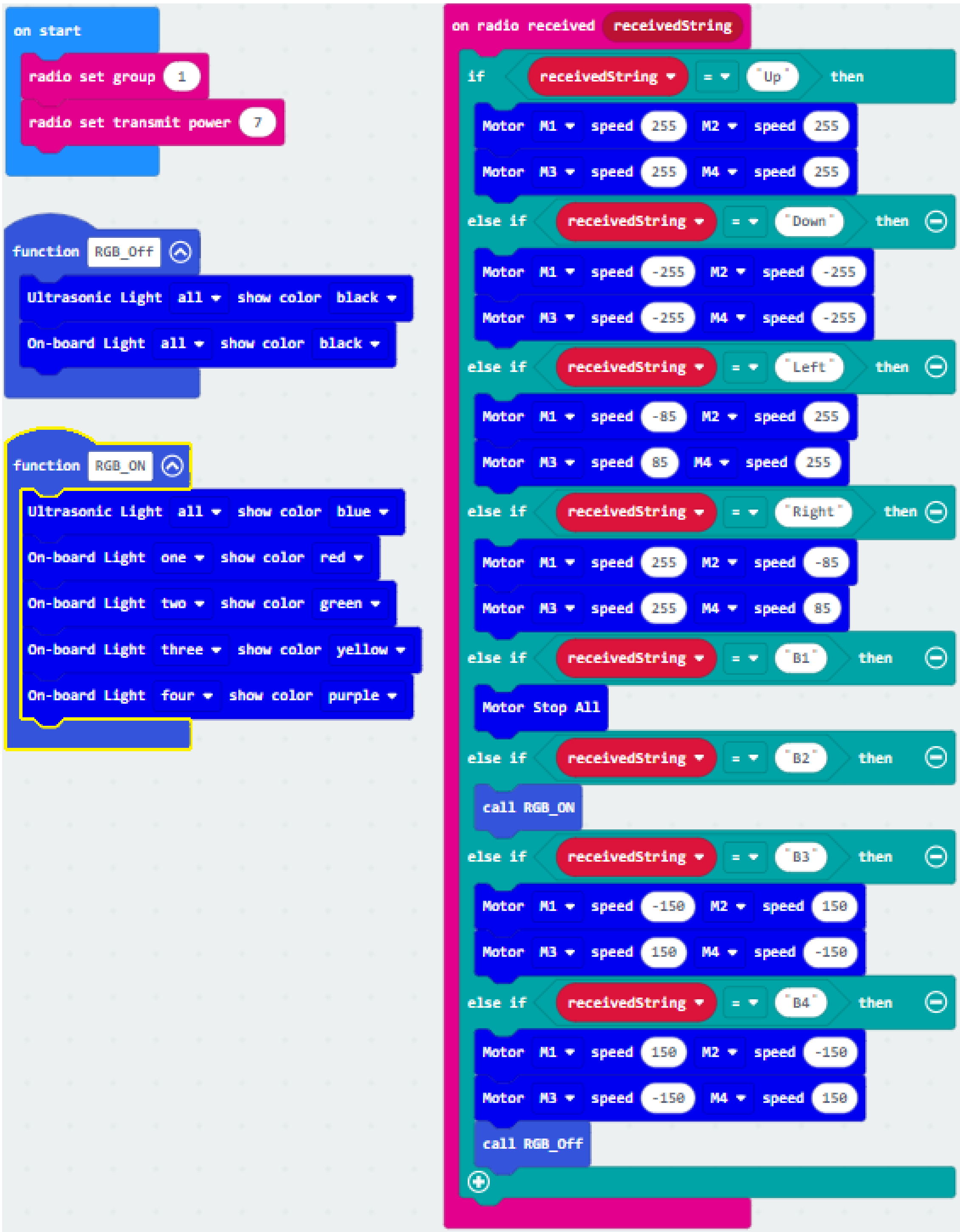

on start
radio set group 1
radio set transmit power 7
on radio received receivedString
if receivedString = "Up" then
Motor M1 speed 255 M2 speed 255
Motor M3 speed 255 M4 speed 255
else if receivedString = "Down" then
Motor M1 speed -255 M2 speed -255
Motor M3 speed -255 M4 speed -255
else if receivedString = "Left" then
Motor M1 speed -85 M2 speed 255
Motor M3 speed 85 M4 speed 255
else if receivedString = "Right" then
Motor M1 speed 255 M2 speed -85
Motor M3 speed 255 M4 speed 85
else if receivedString = "B1" then
Motor Stop All
else if receivedString = "B2" then
show icon
else if receivedString = "B3" then
Motor M1 speed -150 M2 speed 150
Motor M3 speed 150 M4 speed -150
else if receivedString = "B4" then
Motor M1 speed 150 M2 speed -150
Motor M3 speed -150 M4 speed 150

Exercise 1

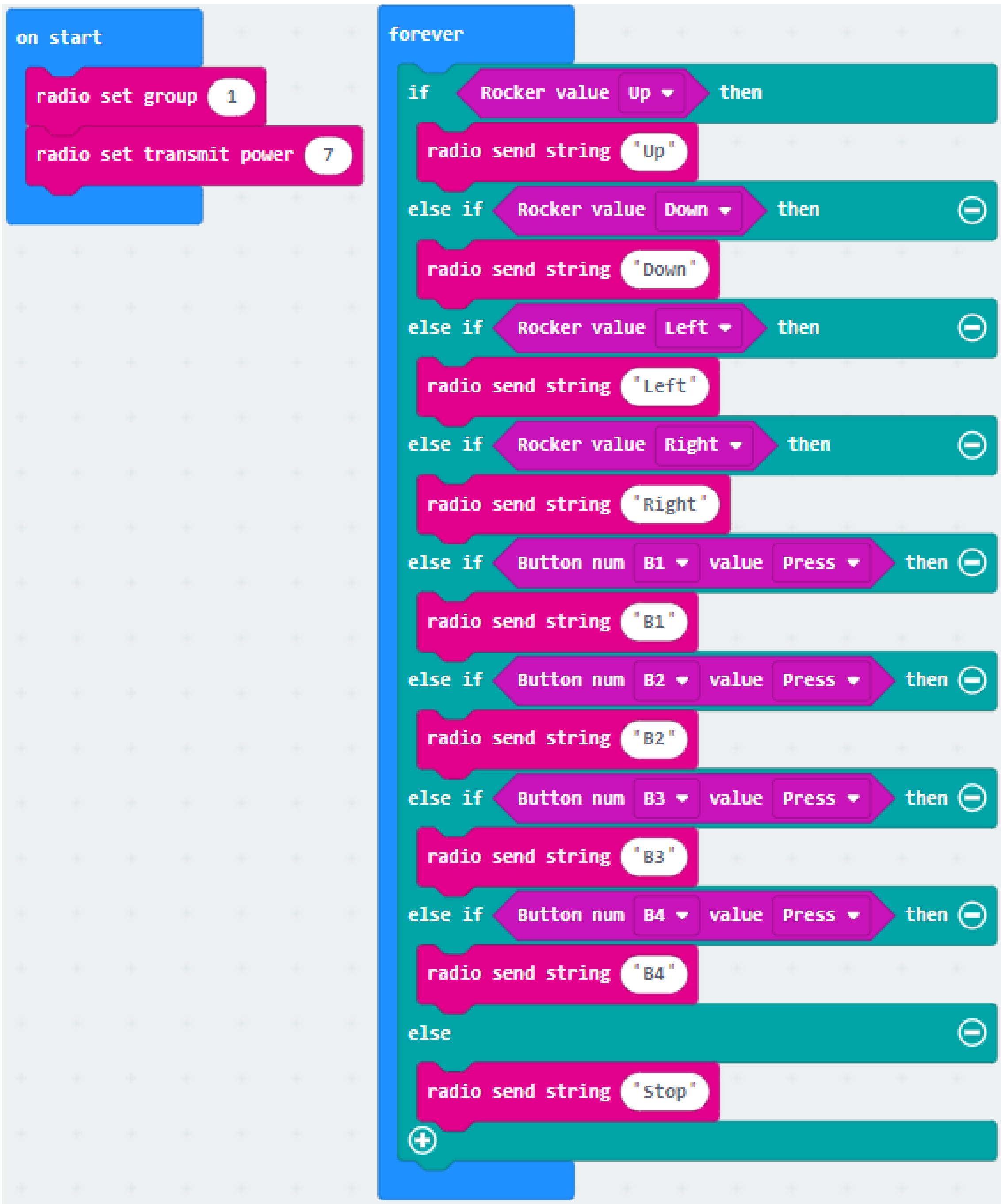
Exercise 2

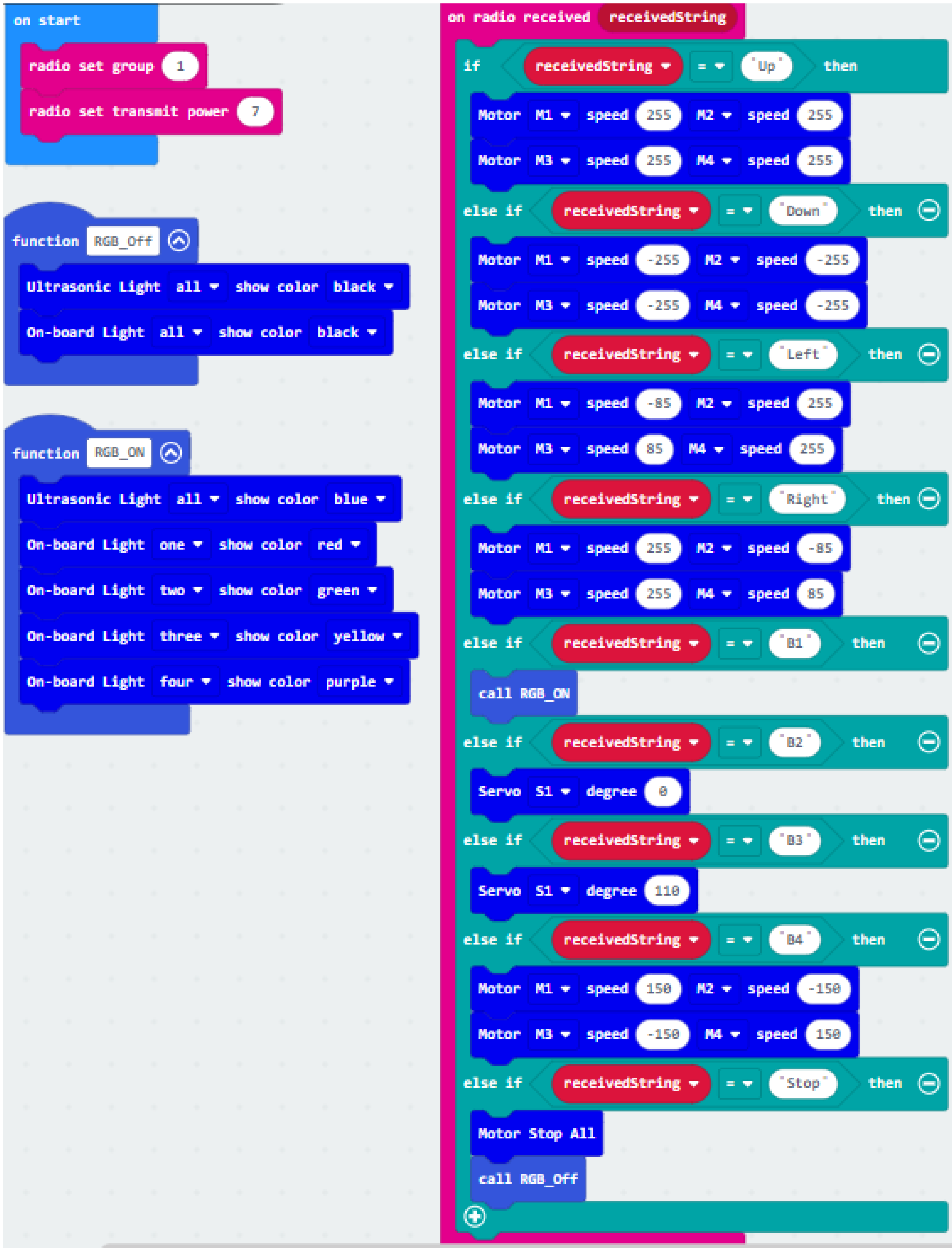
Answers

Exercise 1


on start
radio set group 1
radio set transmit power 7
function RGB_Off
Ultrasonic Light all show color black
On-board Light all show color black
function RGB_ON
Ultrasonic Light all show color blue
On-board Light one show color red
On-board Light two show color green
On-board Light three show color yellow
On-board Light four show color purple
on radio received receivedString
if receivedString = "Up" then
Motor M1 speed 255 M2 speed 255
Motor M3 speed 255 M4 speed 255
else if receivedString = "Down" then
Motor M1 speed -255 M2 speed -255
Motor M3 speed -255 M4 speed -255
else if receivedString = "Left" then
Motor M1 speed -85 M2 speed 255
Motor M3 speed 85 M4 speed 255
else if receivedString = "Right" then
Motor M1 speed 255 M2 speed -85
Motor M3 speed 255 M4 speed 85
else if receivedString = "B1" then
Motor Stop All
else if receivedString = "B2" then
call RGB_ON
else if receivedString = "B3" then
Motor M1 speed -150 M2 speed 150
Motor M3 speed 150 M4 speed -150
else if receivedString = "B4" then
Motor M1 speed 150 M2 speed -150
Motor M3 speed -150 M4 speed 150
call RGB_Off

Exercise 2


on start
radio set group 1
radio set transmit power 7
forever
if Rocker value Up then
radio send string "Up"
else if Rocker value Down then
radio send string "Down"
else if Rocker value Left then
radio send string "Left"
else if Rocker value Right then
radio send string "Right"
else if Button num B1 value Press then
radio send string "B1"
else if Button num B2 value Press then
radio send string "B2"
else if Button num B3 value Press then
radio send string "B3"
else if Button num B4 value Press then
radio send string "B4"
else
radio send string "Stop"


on start
radio set group 1
radio set transmit power 7
function RGB_Off
Ultrasonic Light all show color black
On-board Light all show color black
function RGB_ON
Ultrasonic Light all show color blue
On-board Light one show color red
On-board Light two show color green
On-board Light three show color yellow
On-board Light four show color purple
on radio received receivedString
if receivedString = "Up" then
Motor M1 speed 255 M2 speed 255
Motor M3 speed 255 M4 speed 255
else if receivedString = "Down" then
Motor M1 speed -255 M2 speed -255
Motor M3 speed -255 M4 speed -255
else if receivedString = "Left" then
Motor M1 speed -85 M2 speed 255
Motor M3 speed 85 M4 speed 255
else if receivedString = "Right" then
Motor M1 speed 255 M2 speed -85
Motor M3 speed 255 M4 speed 85
else if receivedString = "B1" then
call RGB_ON
else if receivedString = "B2" then
Servo S1 degree 0
else if receivedString = "B3" then
Servo S1 degree 110
else if receivedString = "B4" then
Motor M1 speed 150 M2 speed -150
Motor M3 speed -150 M4 speed 150
else if receivedString = "Stop" then
Motor Stop All
call RGB_Off

1.4 Micro:bit M1 SMART CAR_Intermediate

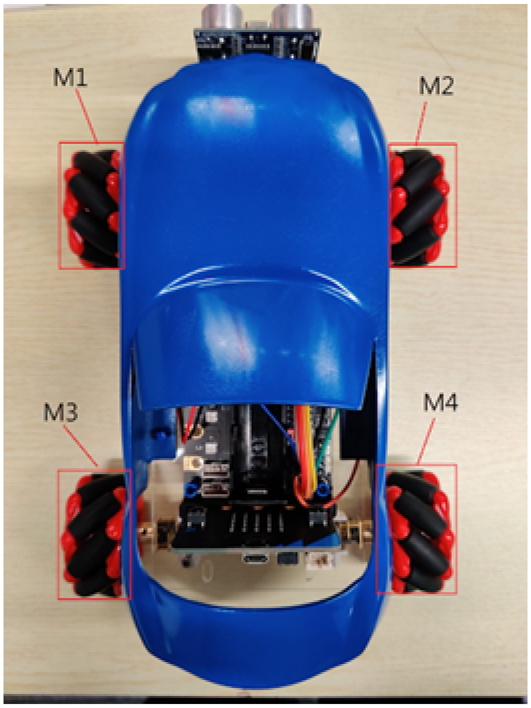
1.4.1 Lesson 1

Introduction

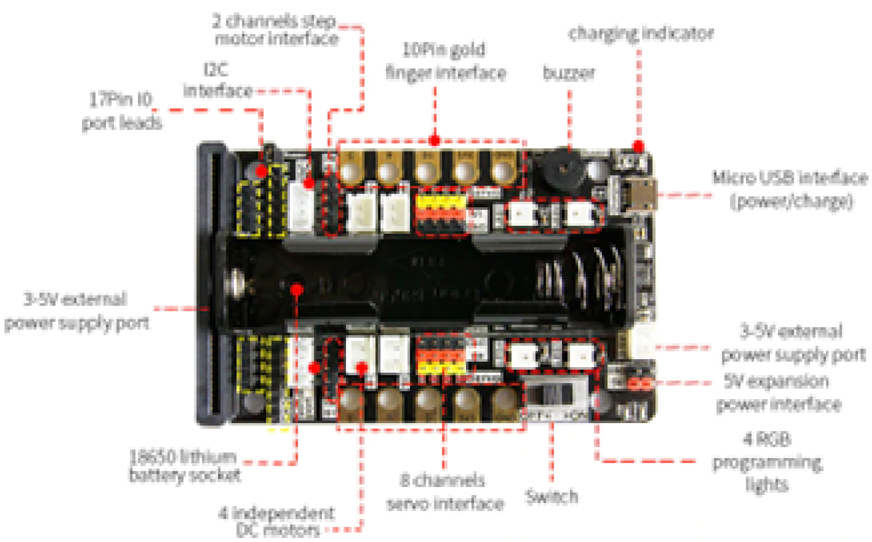
Learning Target

Reviewing the structure of Mecanum Wheel Car

- 4 Motor(M1, M2, M3, M4)
- 4 Mecanum Wheel
- Micro bit
- Micro bit expanding board



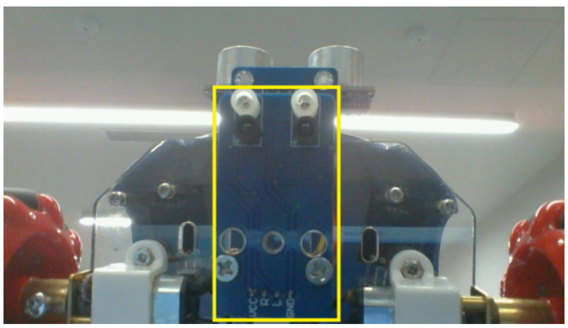
micro:bit expansion board



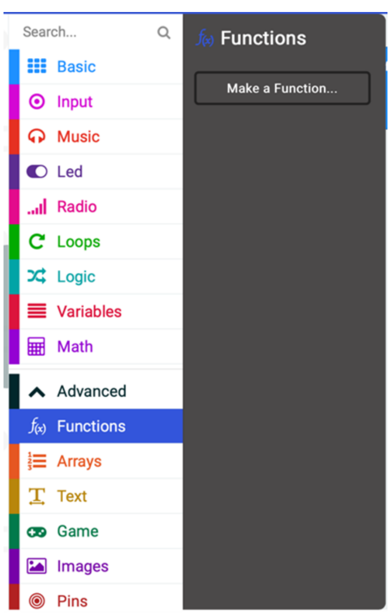
Sensor

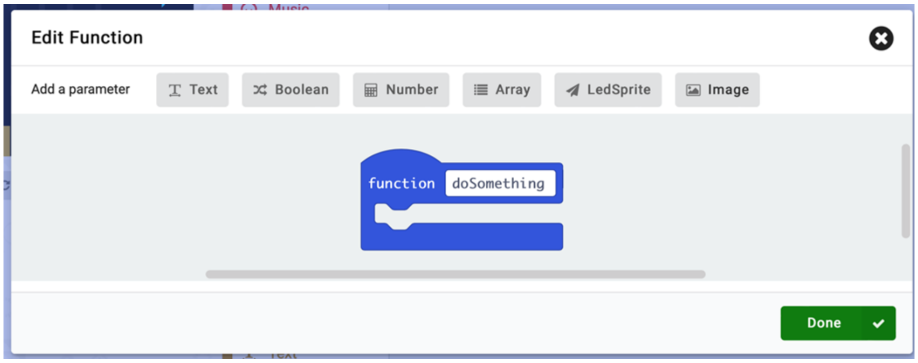
- Ultrasonic Sensor


- Infrared Receiver and Sender



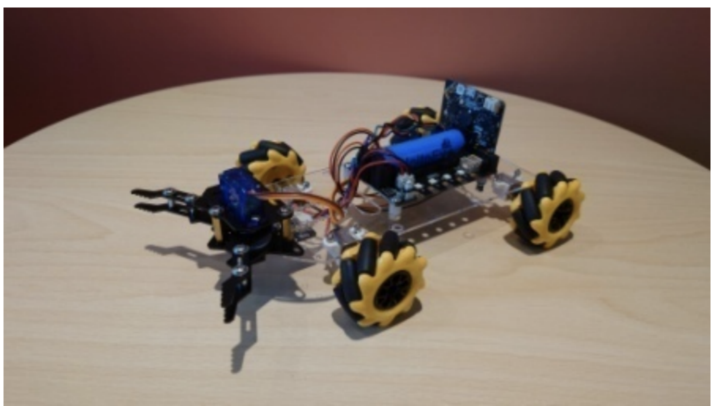
Reviewing of advanced block



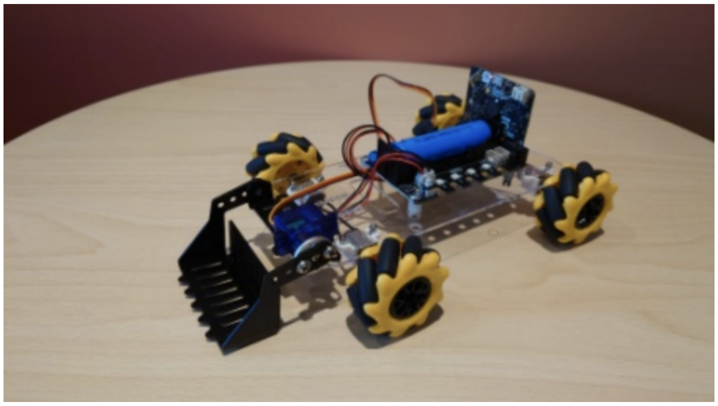


Meet the Mechanical extension tools

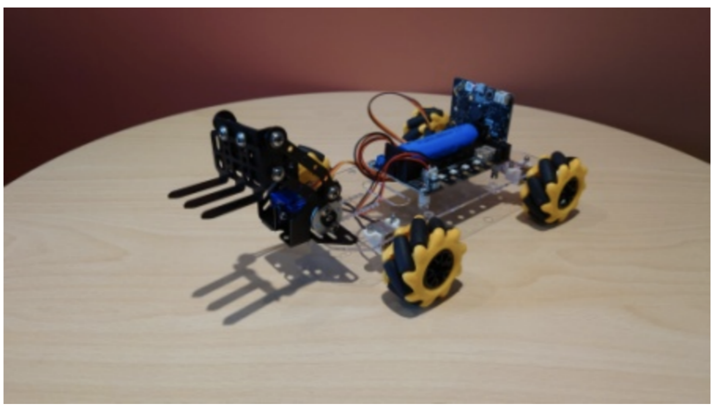
Beetle



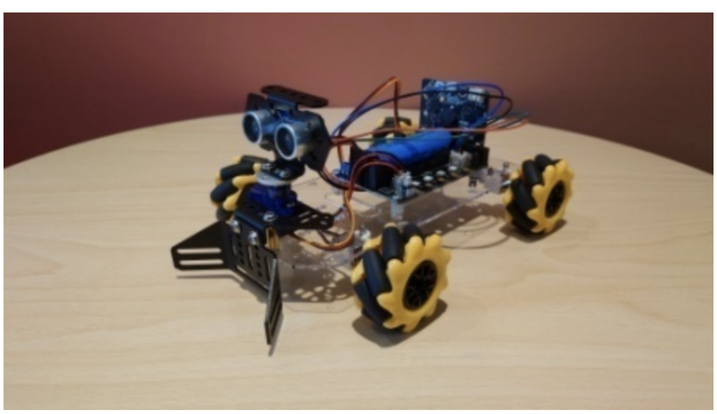
Loader



Forklift

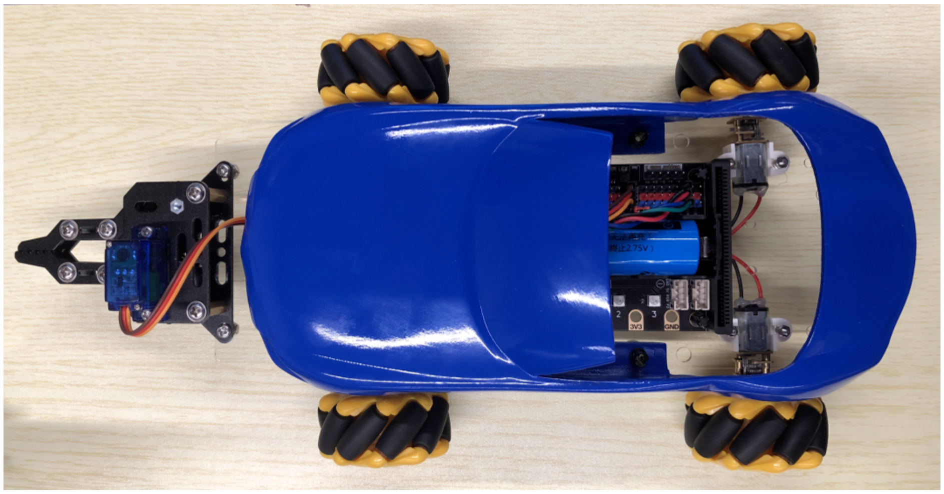


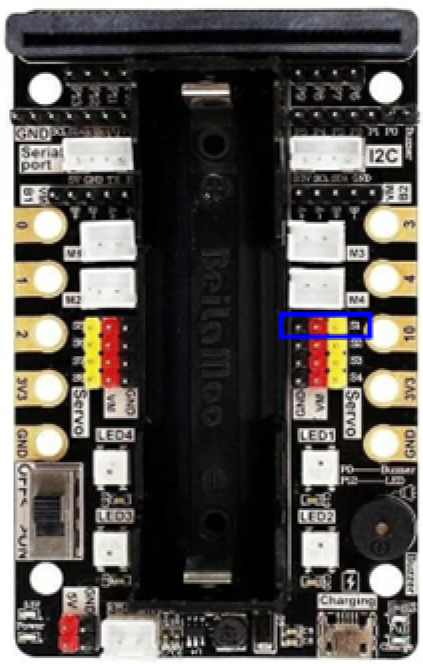
Bulldozer with ultrasonic



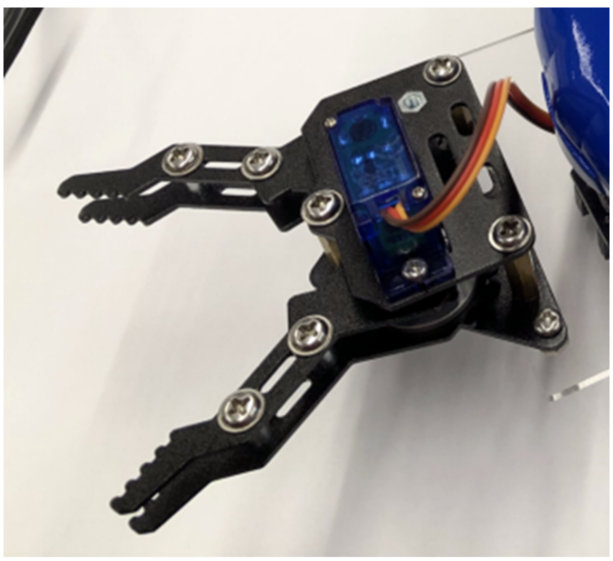
Components in extension package

Meet the Micro:bit extension tools – beetle

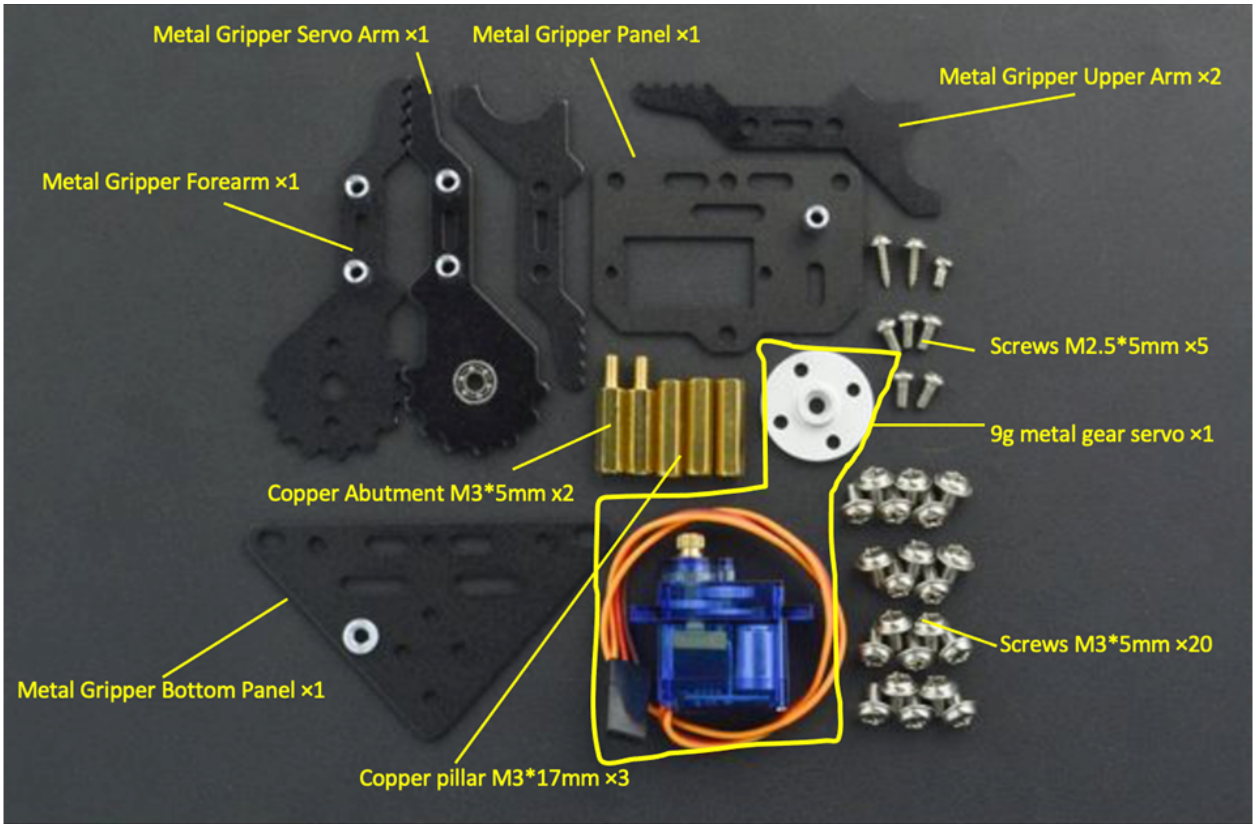


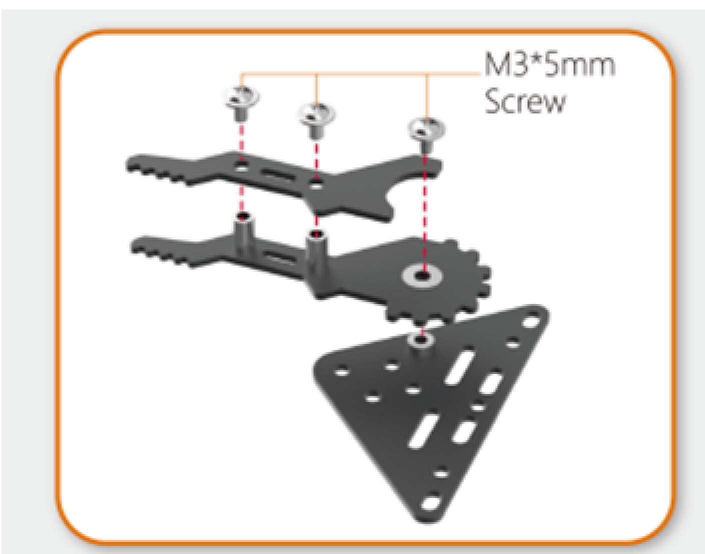


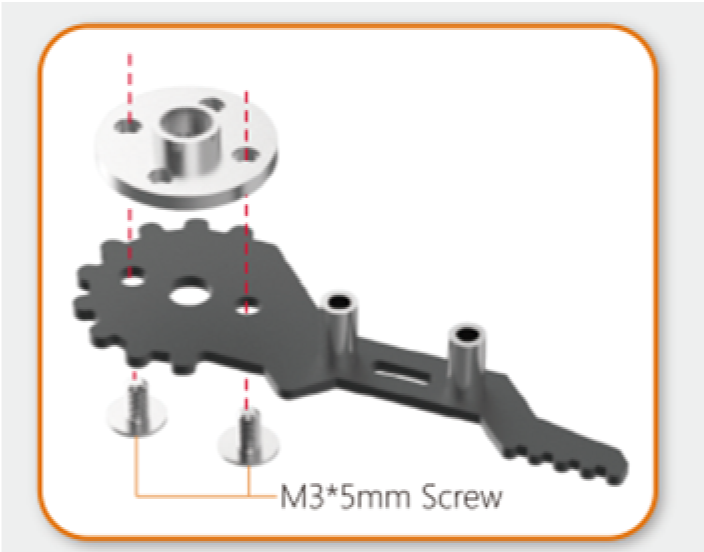
Principle and function of Mechanical beetle

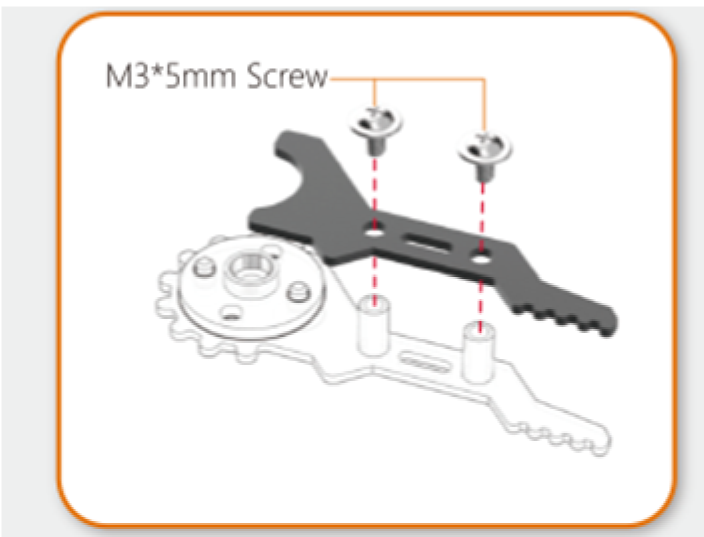


Install of beetle

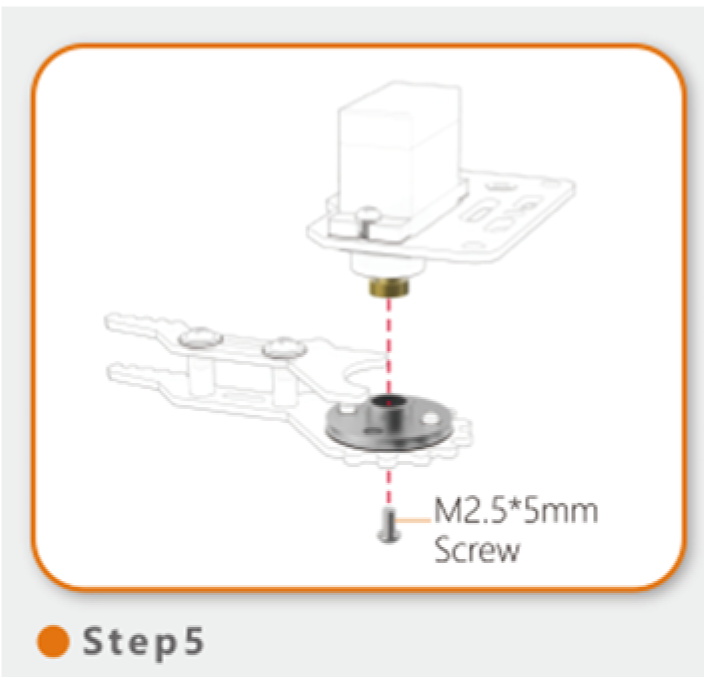


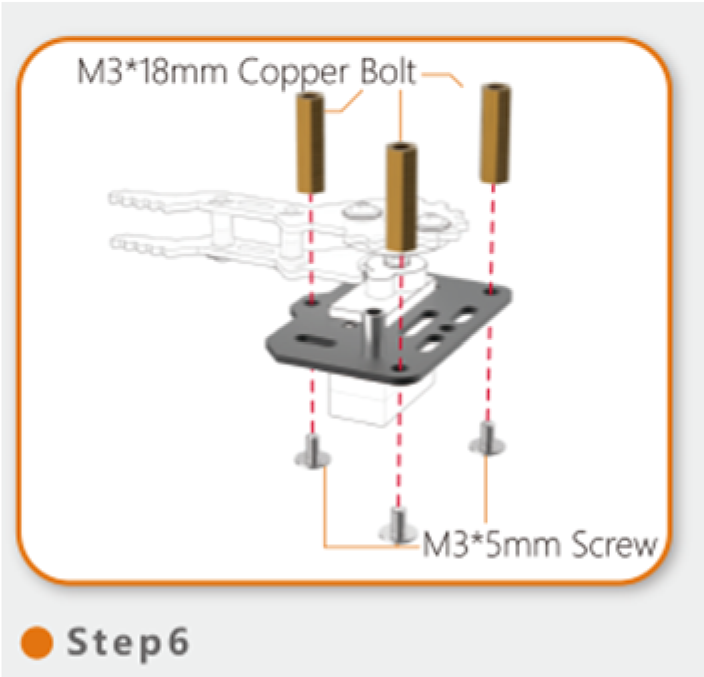


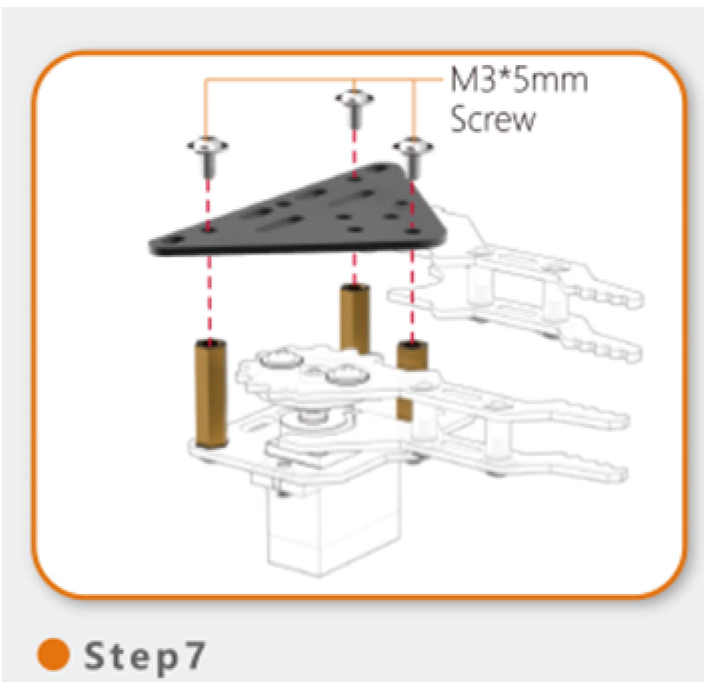

M3*5mm Screw

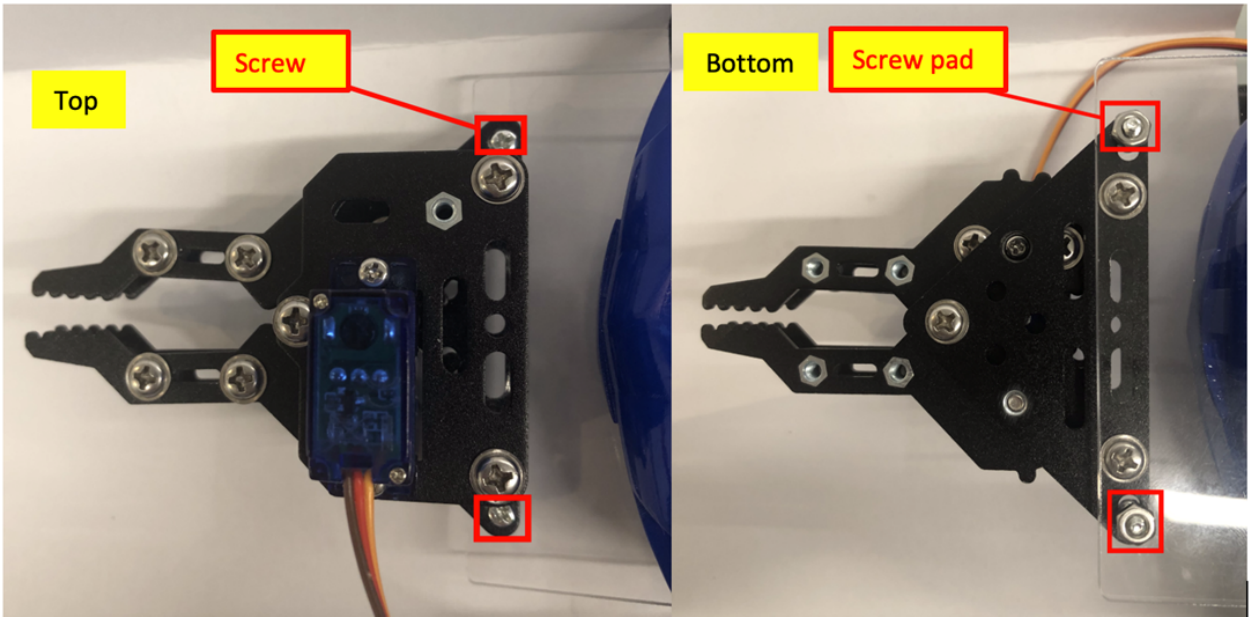

M3*5mm Screw


M2.5*5mm
Screw
Step4

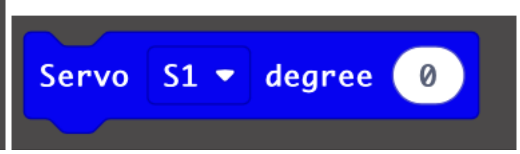

M2.5*5mm
Screw
Step5


M3*18mm Copper Bolt
M3*5mm Screw
Step6

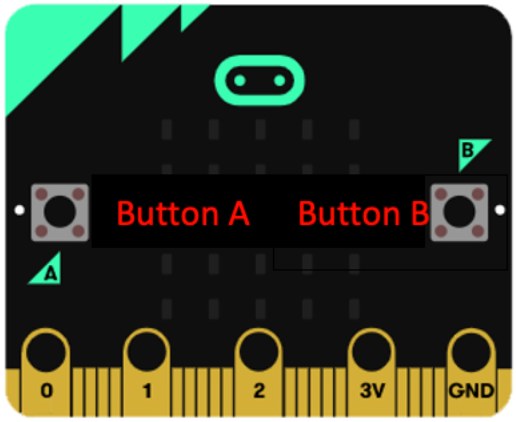

M3*5mm
Screw
Step7

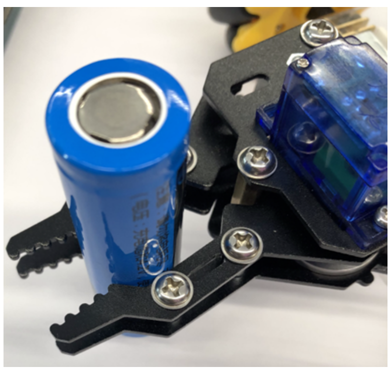
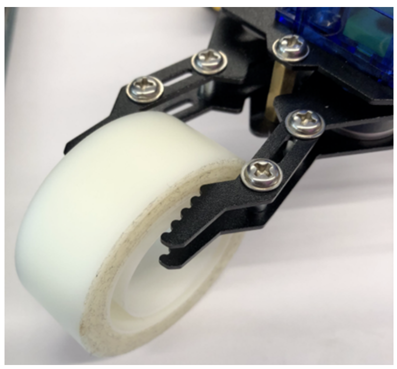


Stemhub:bit - Servo Motor block module

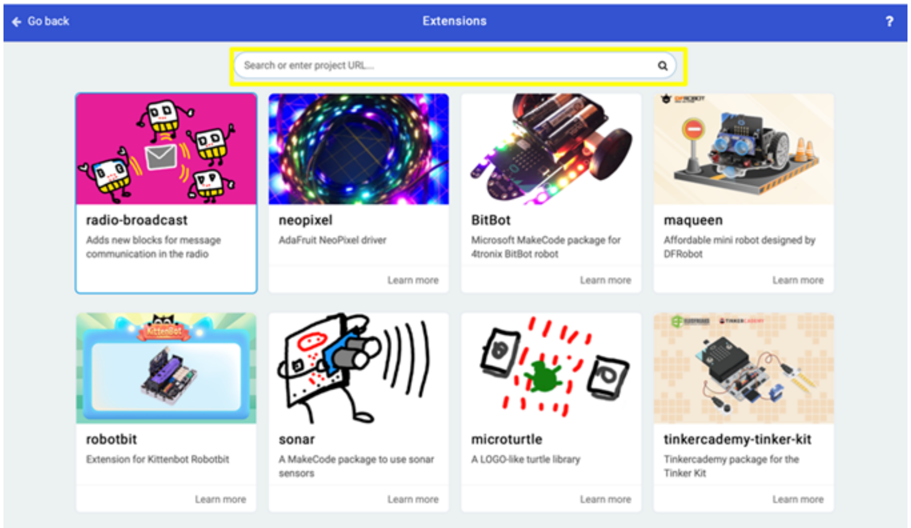


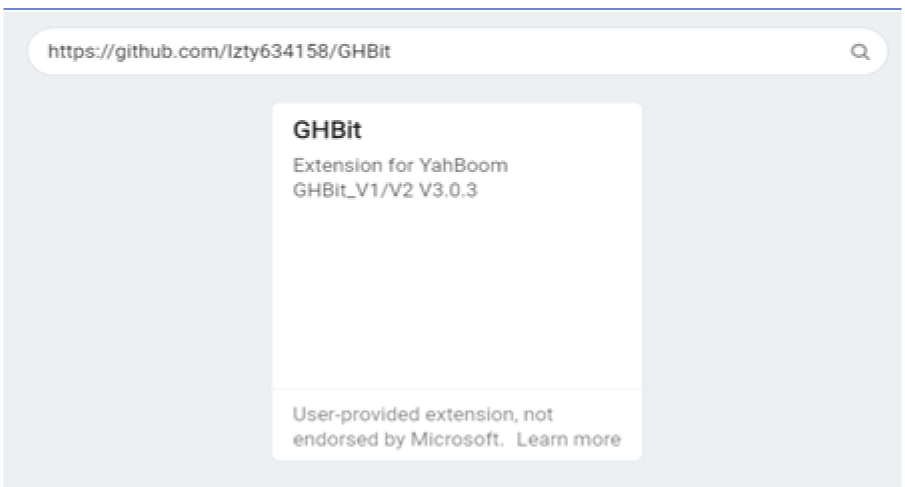
Exercise 1

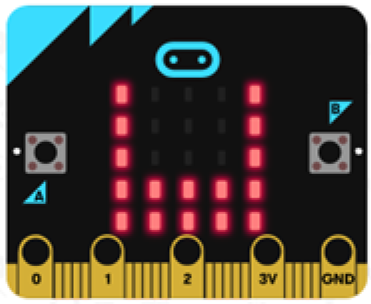


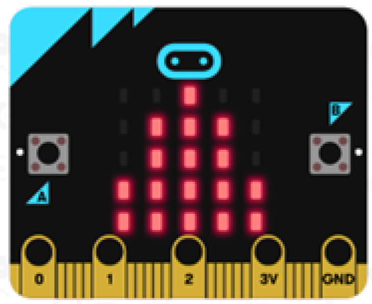



Exercise 2


Go back
Extensions
?
Search or enter project URL...
radio-broadcast
Adds new blocks for message communication in the radio
neopixel
AdaFruit NeoPixel driver
Learn more
BitBot
Microsoft MakeCode package for 4tronix BitBot robot
Learn more
maqueen
Affordable mini robot designed by DFRobot
Learn more
robotbit
Extension for Kittenbot Robotbit
Learn more
sonar
A MakeCode package to use sonar sensors
Learn more
microturtle
A LOGO-like turtle library
Learn more
tinkercademy-tinker-kit
Tinkercademy package for the Tinker Kit
Learn more


https://github.com/lzty634158/GHBit
GHBit
Extension for YahBoom GHBit_V1/V2 V3.0.3
User-provided extension, not endorsed by Microsoft. Learn more

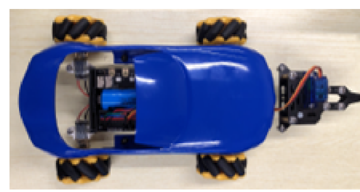
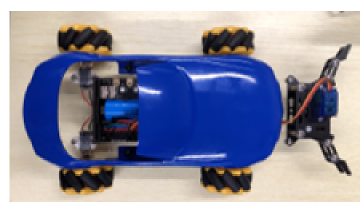


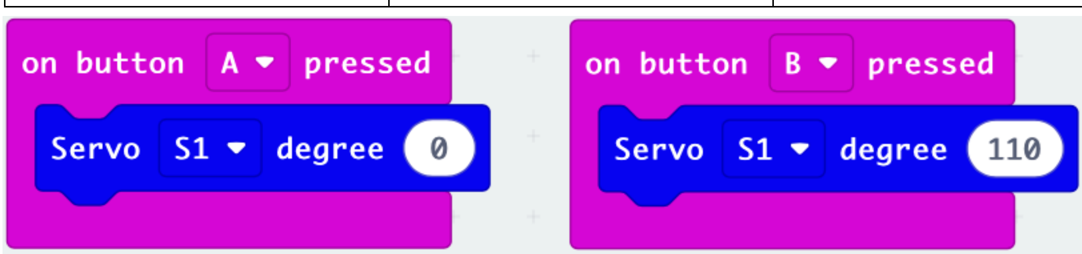


- Refer to Beginner, Lesson 10, receive text as operation condition
- Apply basic >> Show of indicate light using block module

Answer

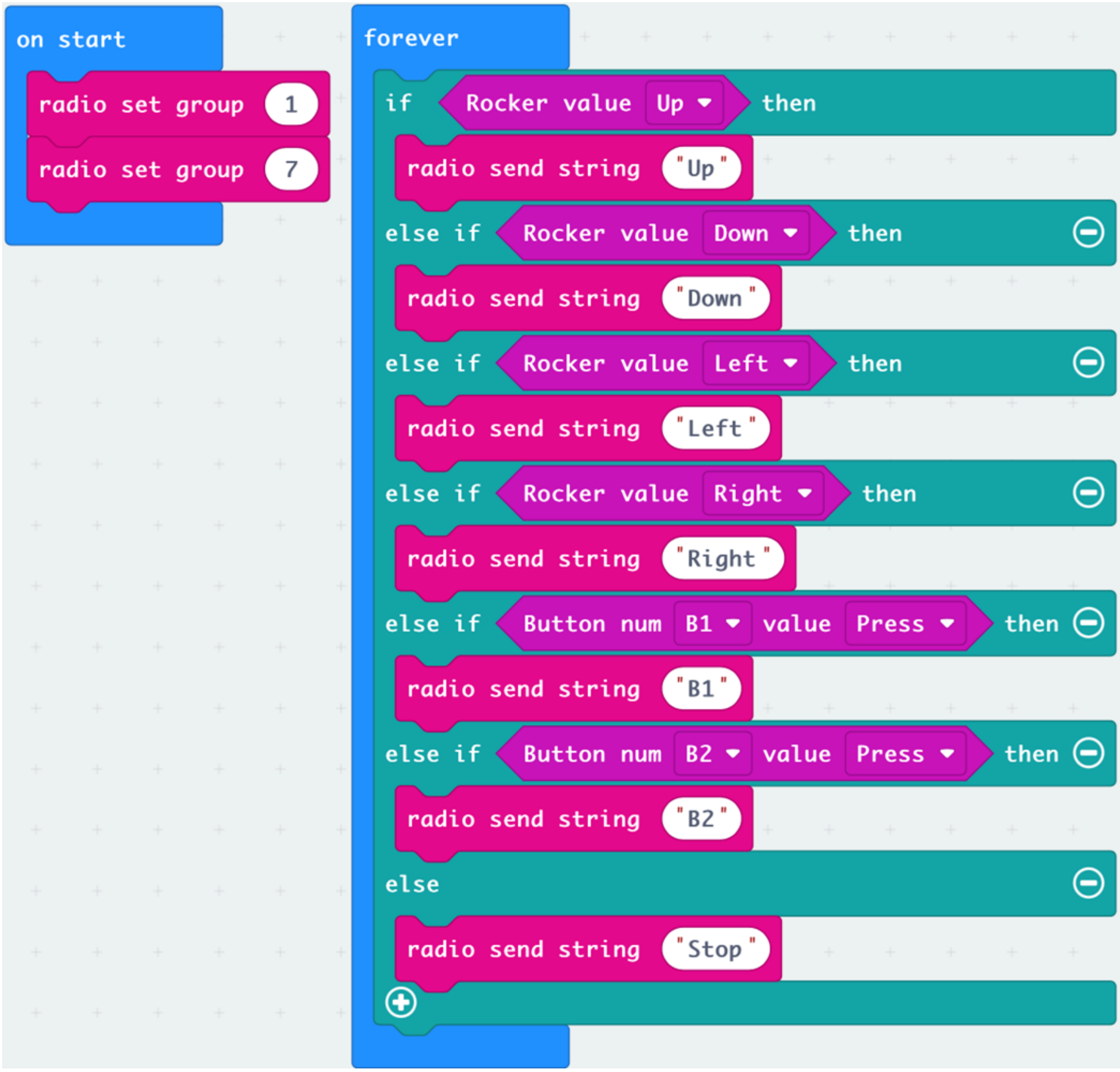
Exercise 1

Angle value	Beetle movement	
Increase	Close	
Decrease	Open	

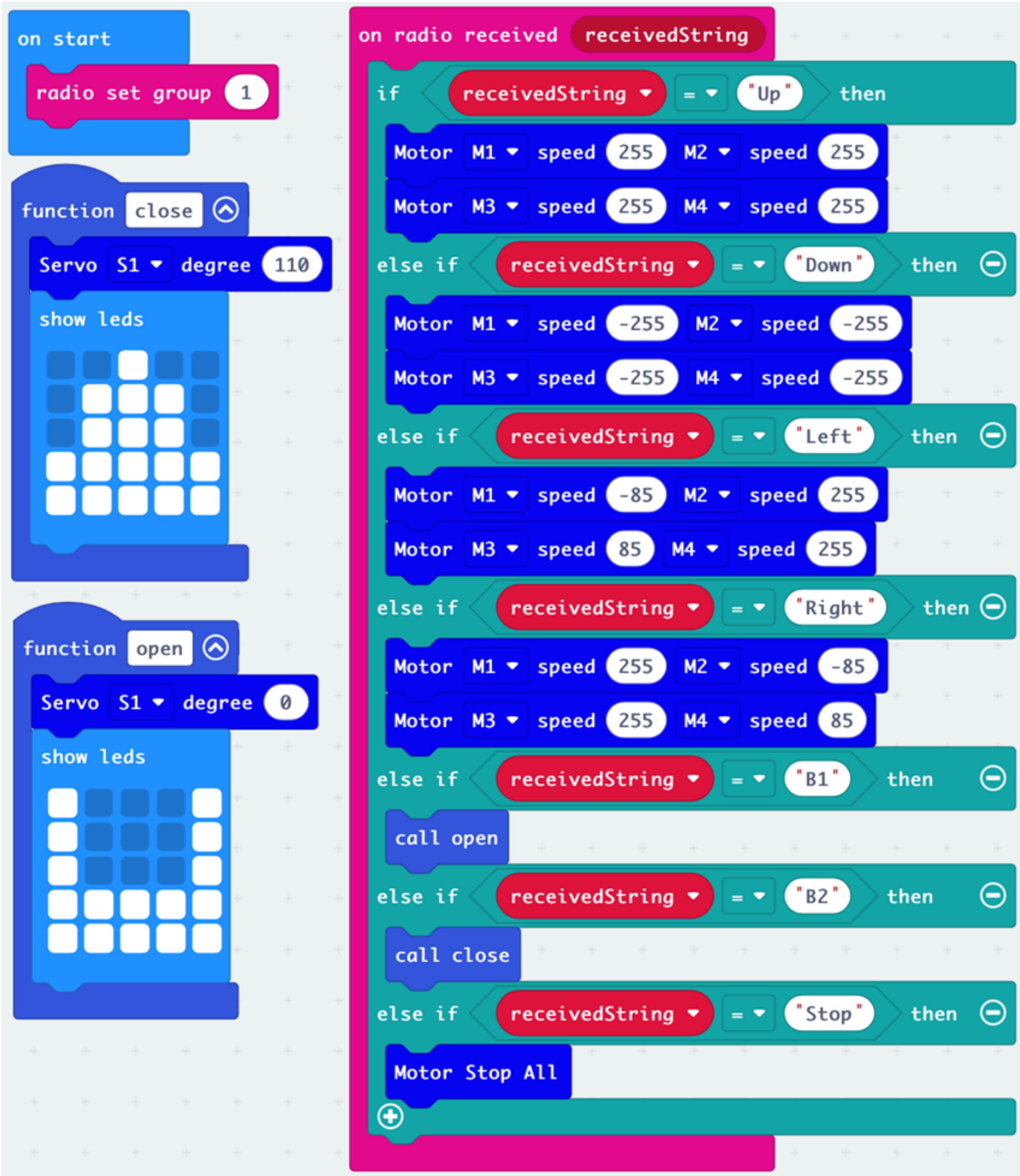


Exercise 2

Program of Remote



Program of the Car

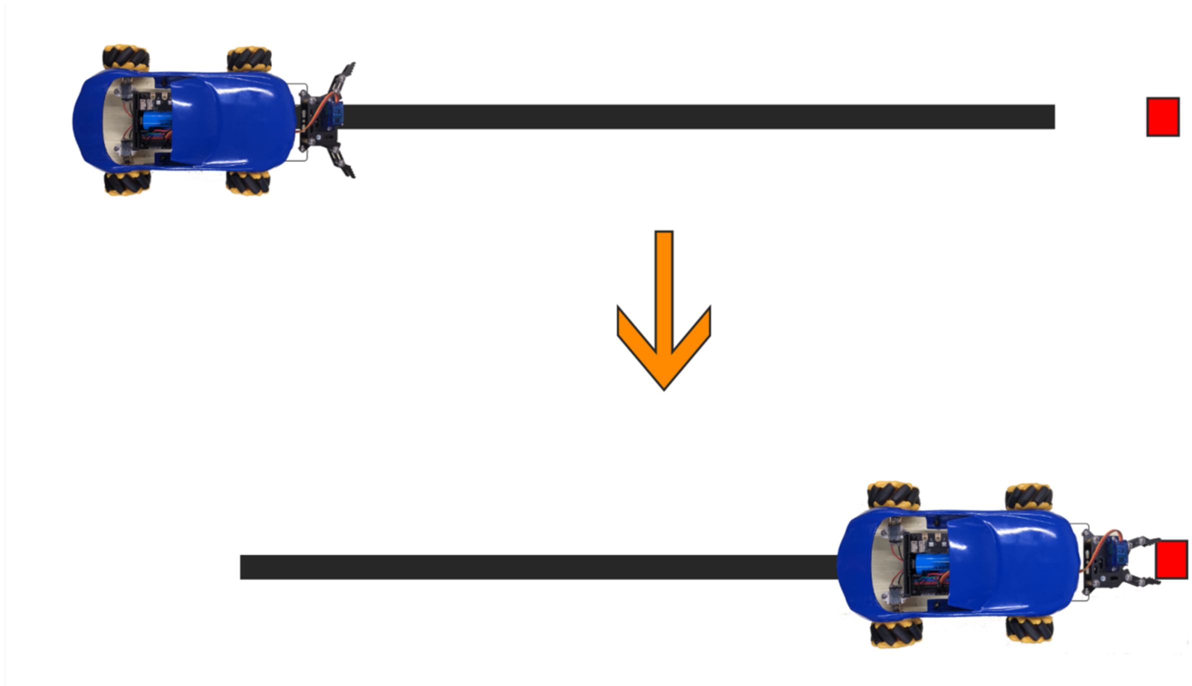


1.4.2 Lesson 2

Introduction

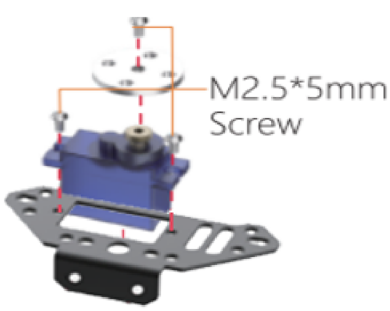
Learning Tutorial

Exercise 1

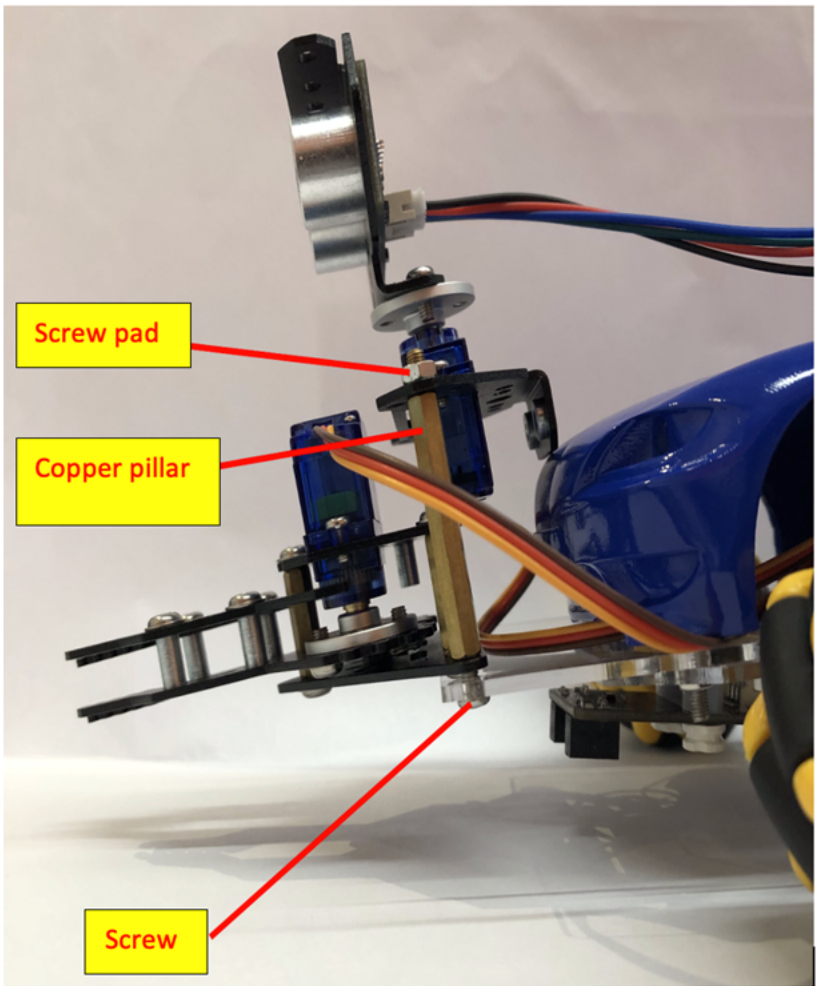


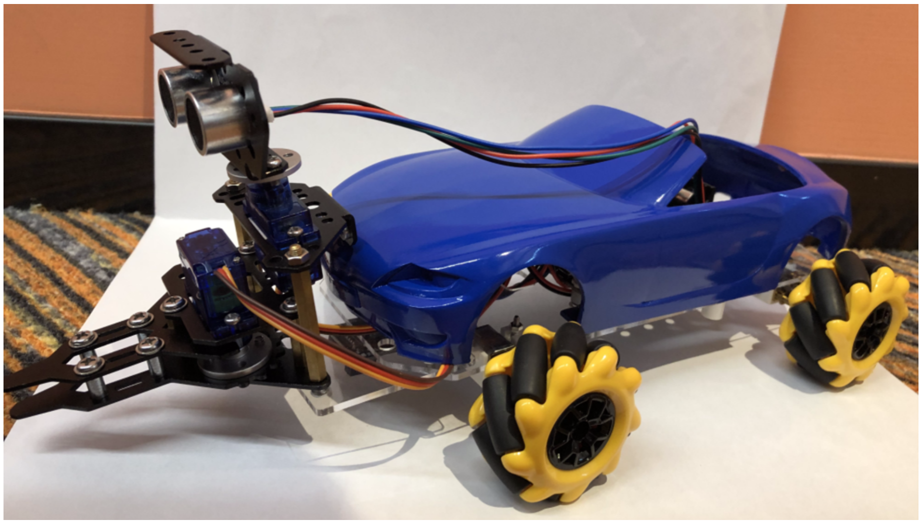
Install of Ultrasonic sensor:

	
universal expansion mounting plate	pan-tilt-zoom mount plate

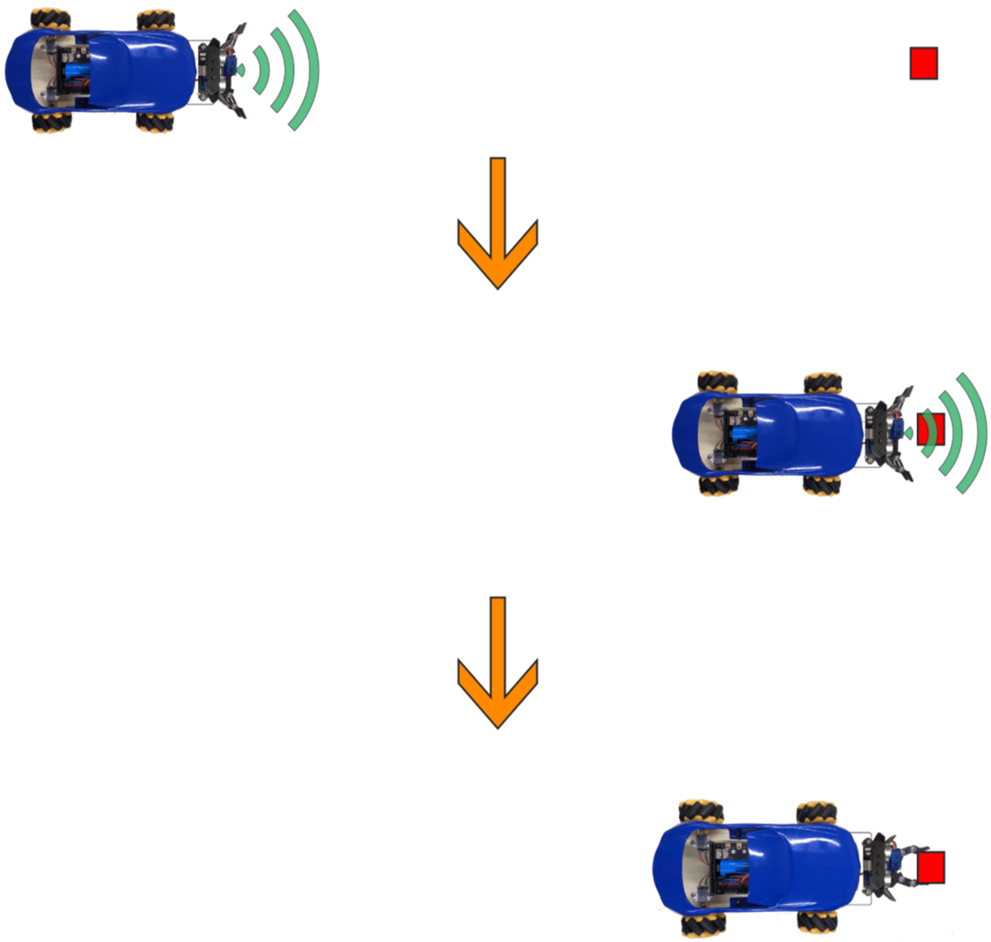



M3*5mm
Screw

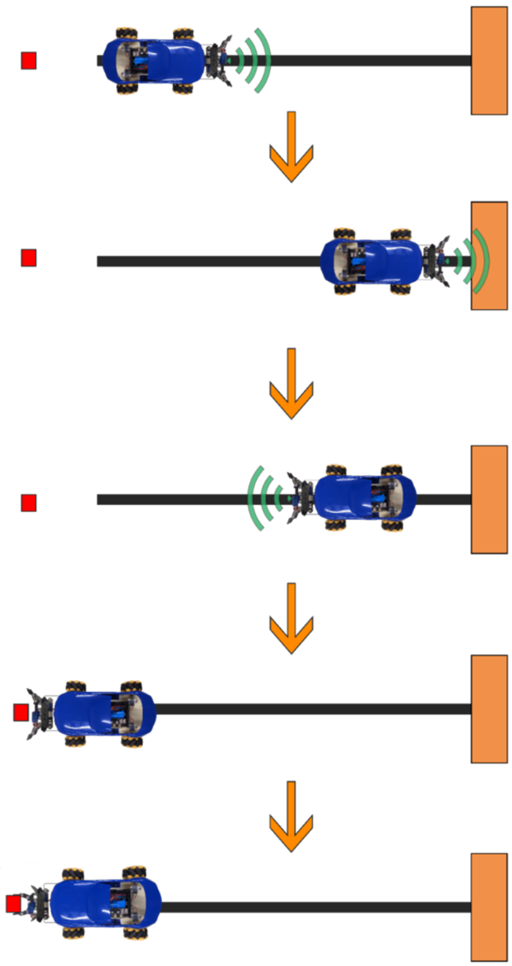

Screw pad
Copper pillar
Screw



Exercise 2



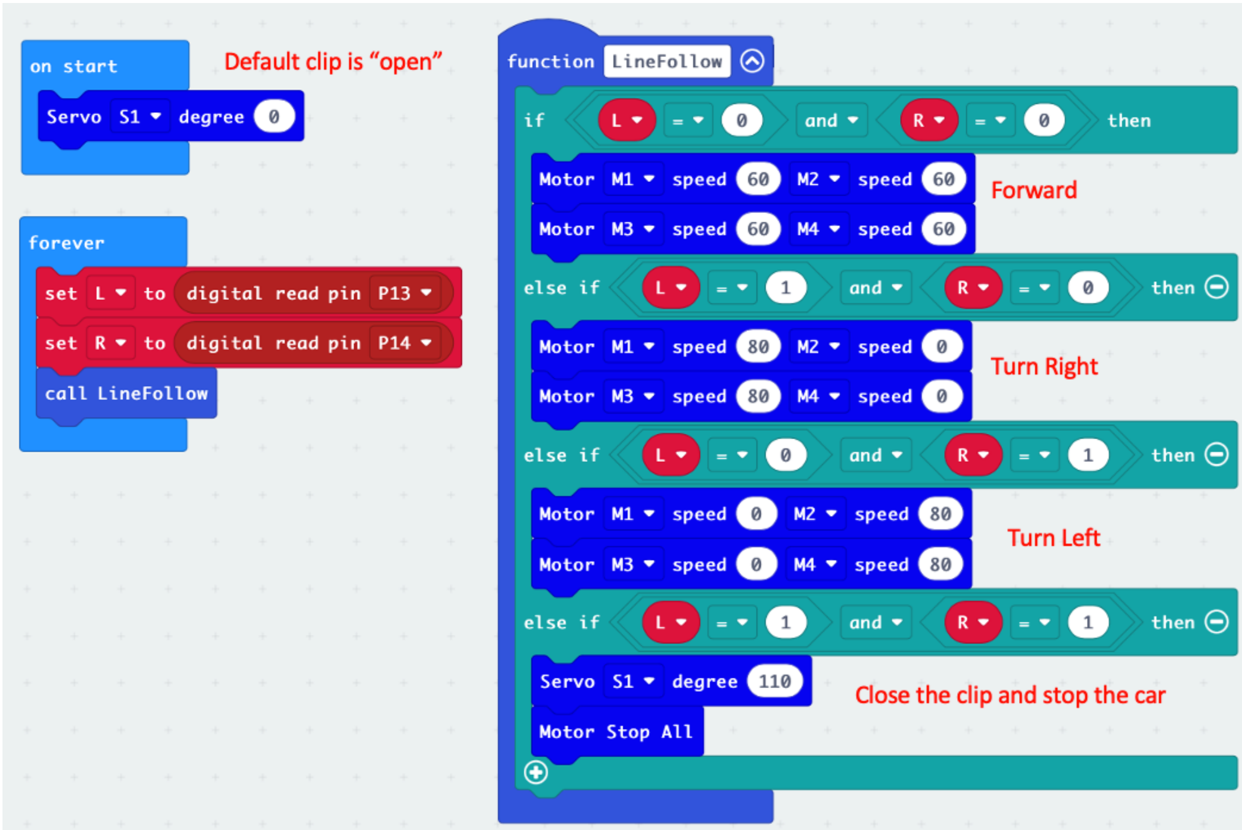
Exercise 3



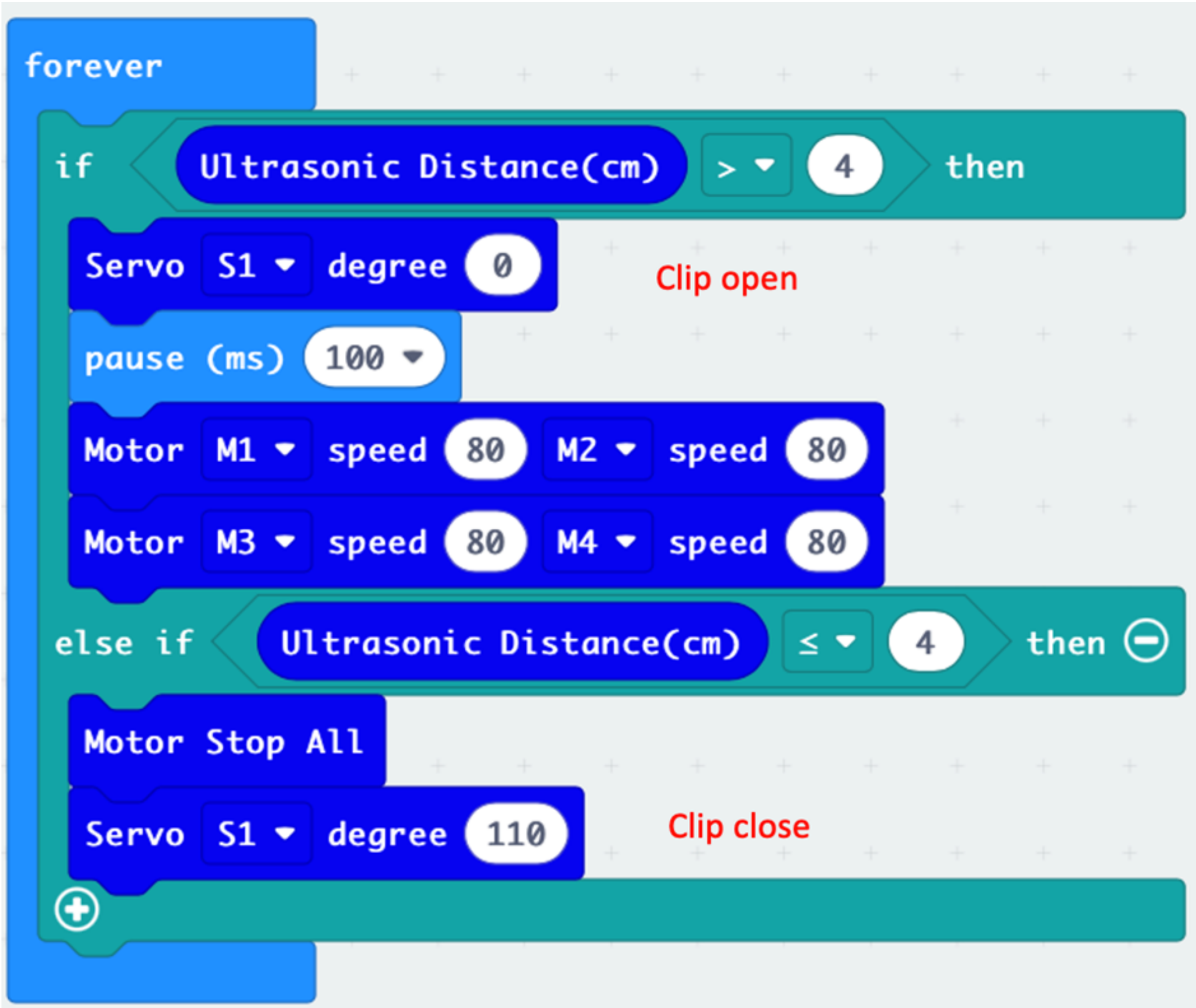
- Built variable “goal” to mark the destination, and use it in Conditional blockUse text “start” and “end” indicate start point and end point
- Beware of the sequence of the condition of ultrasonic, line patrol and “goal” condition

Answer

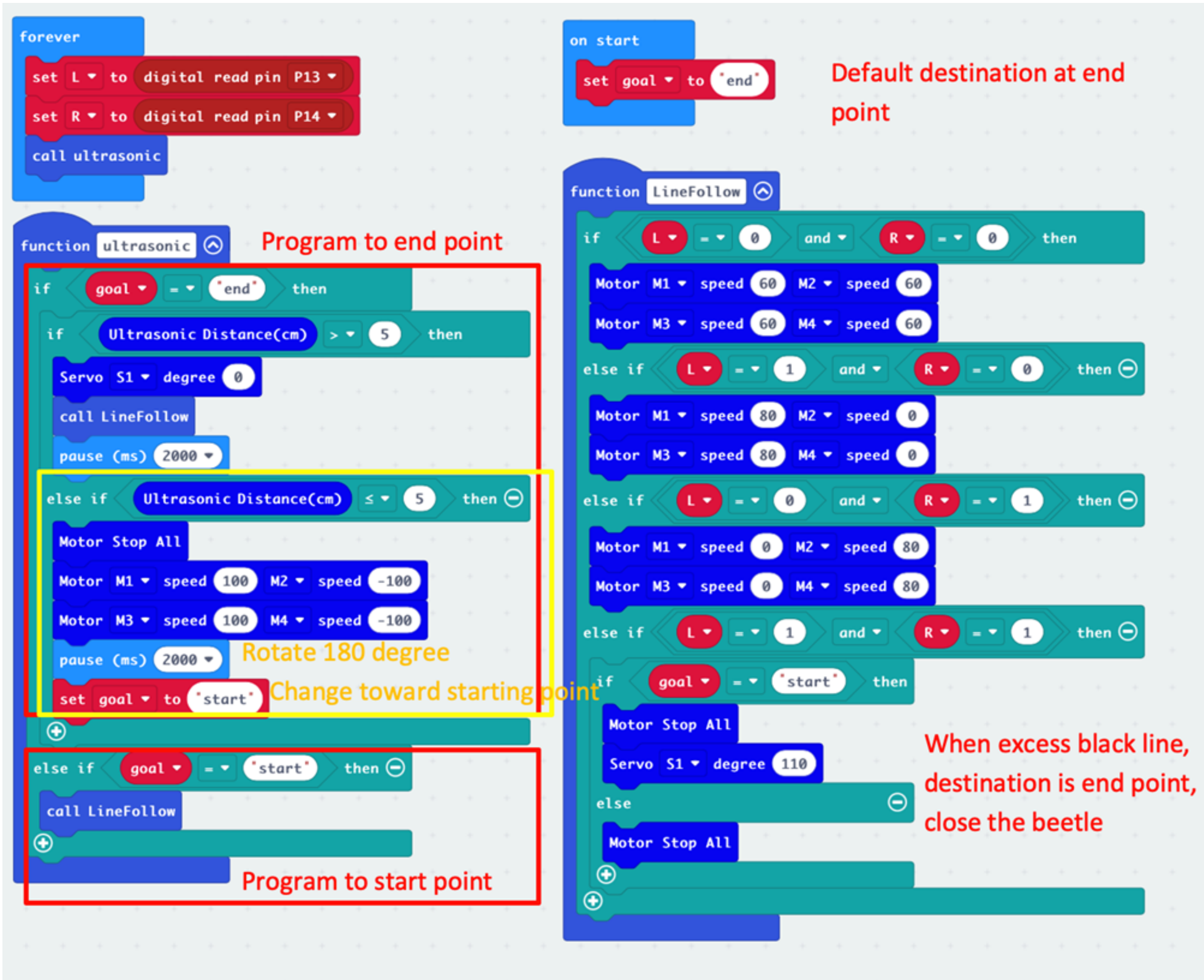
Exercise 1


on start
Servo S1 degree 0
Default clip is "open"
forever
set L to digital read pin P13
set R to digital read pin P14
call LineFollow
function LineFollow
if L = 0 and R = 0 then
Motor M1 speed 60 M2 speed 60
Motor M3 speed 60 M4 speed 60
Forward
else if L = 1 and R = 0 then
Motor M1 speed 80 M2 speed 0
Motor M3 speed 80 M4 speed 0
Turn Right
else if L = 0 and R = 1 then
Motor M1 speed 0 M2 speed 80
Motor M3 speed 0 M4 speed 80
Turn Left
else if L = 1 and R = 1 then
Servo S1 degree 110
Motor Stop All
Close the clip and stop the car

Exercise 2


forever
if
Ultrasonic Distance(cm)
>
4
then
Servo
S1
degree
0
Clip open
pause (ms)
100
Motor
M1
speed
80
M2
speed
80
Motor
M3
speed
80
M4
speed
80
else if
Ultrasonic Distance(cm)
≤
4
then
Motor Stop All
Servo
S1
degree
110
Clip close

Exercise 3

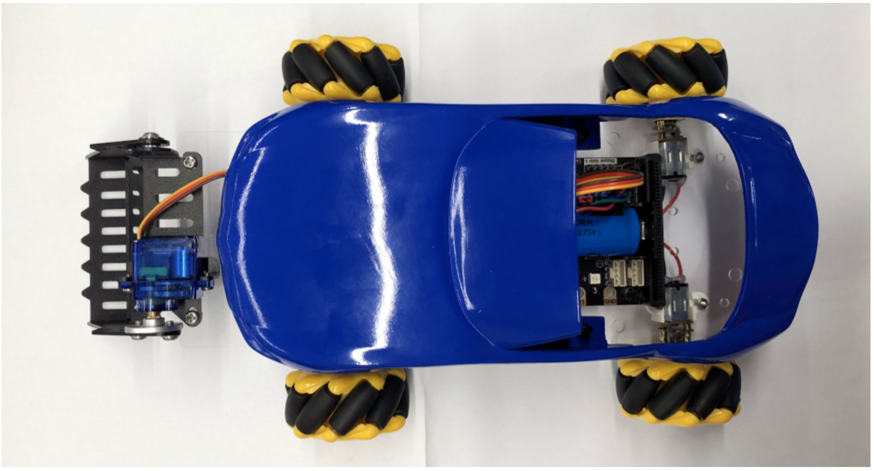

forever
set L to digital read pin P13
set R to digital read pin P14
call ultrasonic
function ultrasonic
Program to end point
if goal = "end" then
if Ultrasonic Distance(cm) > 5 then
Servo S1 degree 0
call LineFollow
pause (ms) 2000
else if Ultrasonic Distance(cm) ≤ 5 then
Motor Stop All
Motor M1 speed 100 M2 speed -100
Motor M3 speed 100 M4 speed -100
pause (ms) 2000
Rotate 180 degree
set goal to "start"
Change toward starting point
else if goal = "start" then
call LineFollow
Program to start point
on start
set goal to "end"
Default destination at end point
function LineFollow
if L = 0 and R = 0 then
Motor M1 speed 60 M2 speed 60
Motor M3 speed 60 M4 speed 60
else if L = 1 and R = 0 then
Motor M1 speed 80 M2 speed 0
Motor M3 speed 80 M4 speed 0
else if L = 0 and R = 1 then
Motor M1 speed 0 M2 speed 80
Motor M3 speed 0 M4 speed 80
else if L = 1 and R = 1 then
if goal = "start" then
Motor Stop All
Servo S1 degree 110
else
Motor Stop All
When excess black line, destination is end point, close the beetle

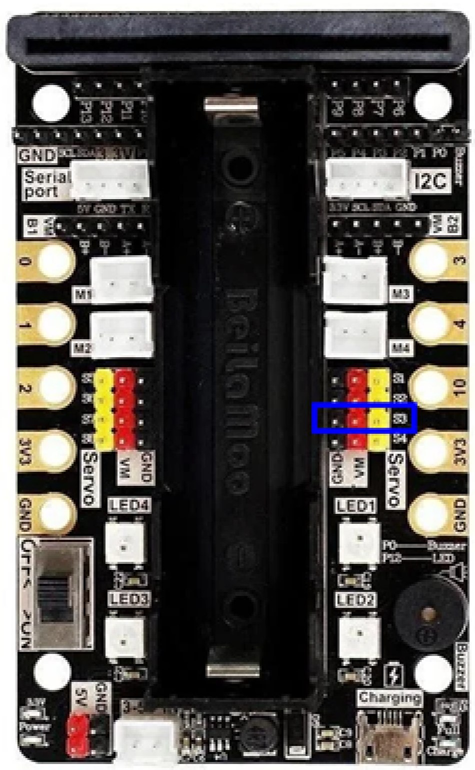
1.4.3 Lesson 3

Introduction

Learning Target

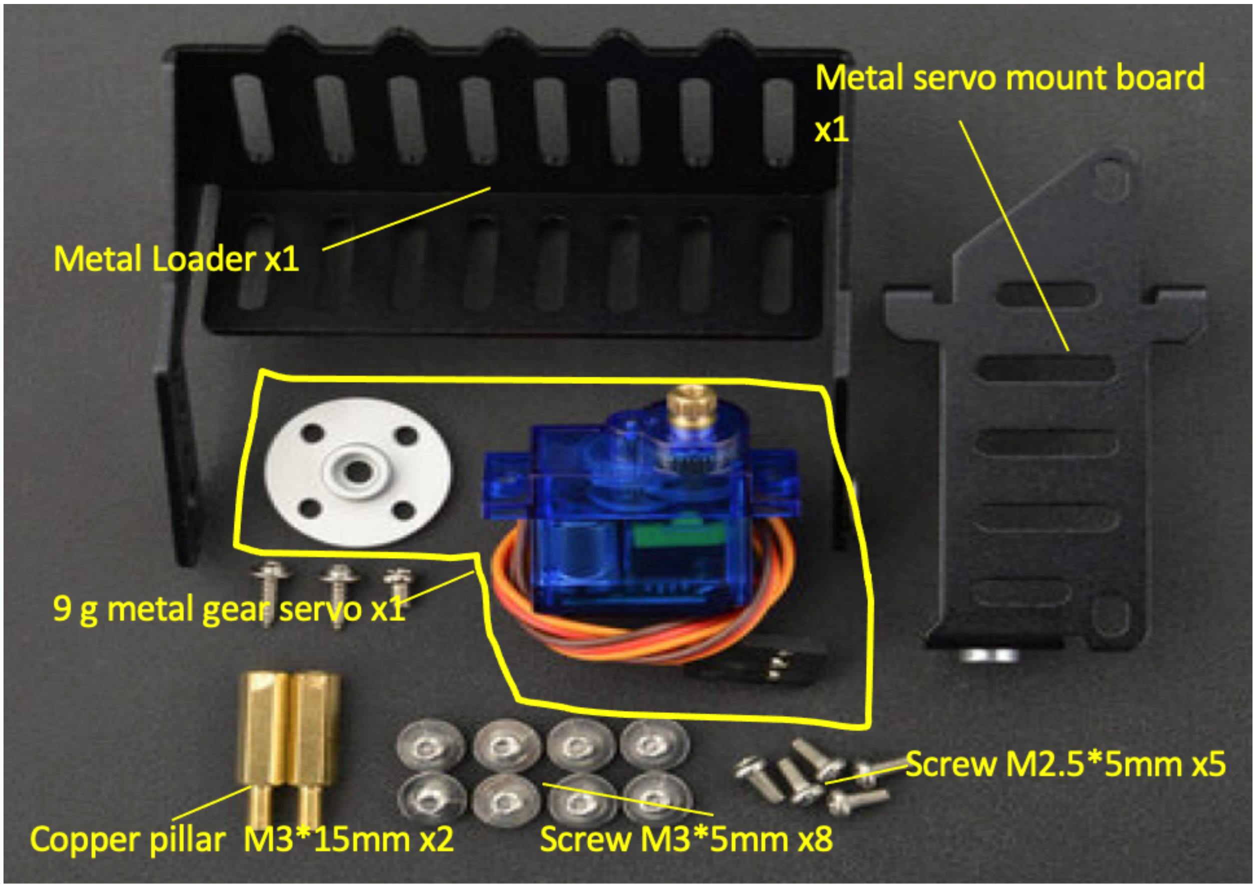
Understand Micro:bit expansion tool – Loader



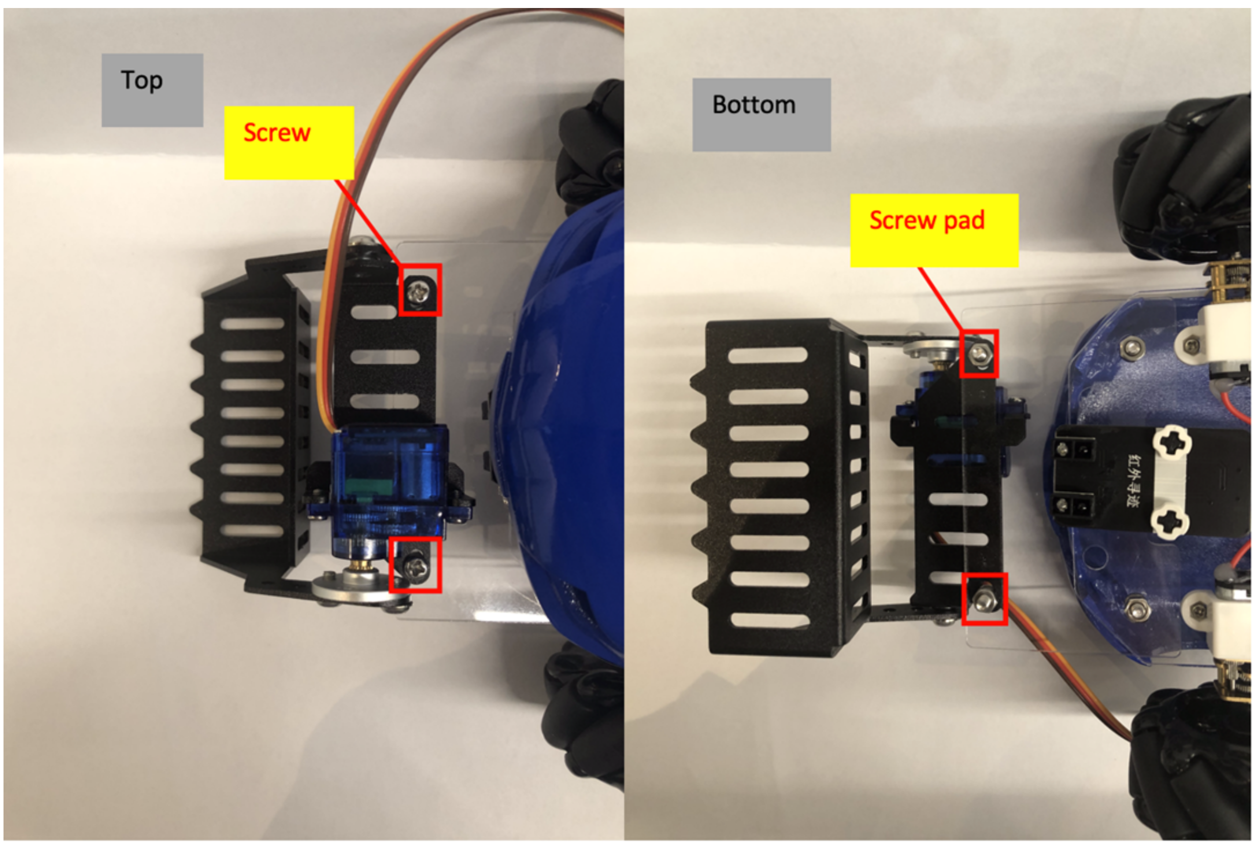


Principle and function of loader

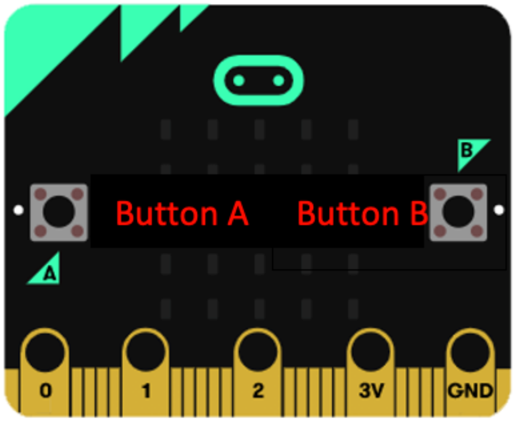
Install of Loader



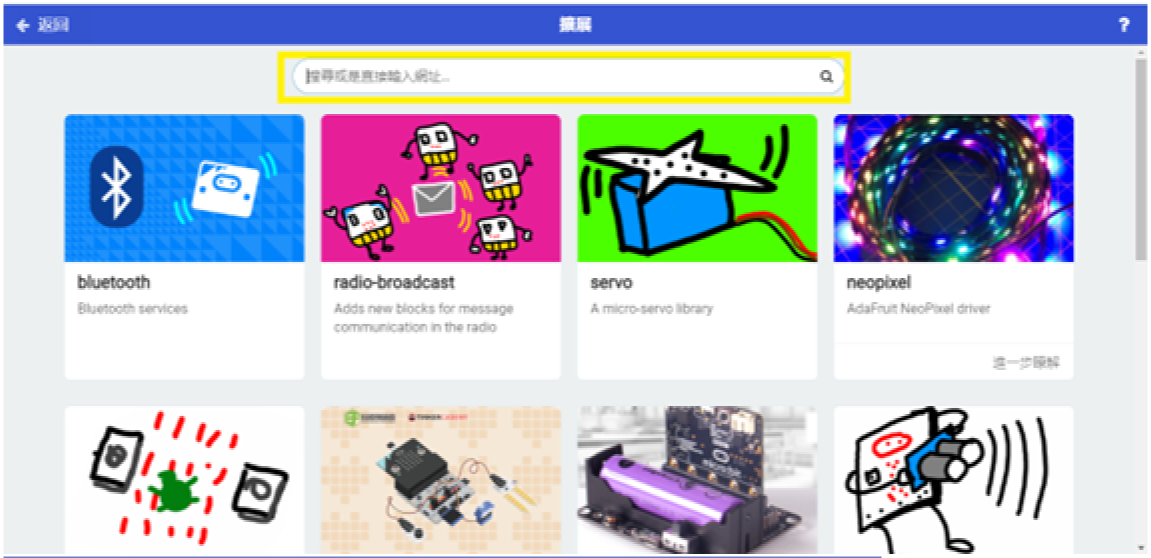

M3*5mm Screw
M3*5mm Screw
Step2

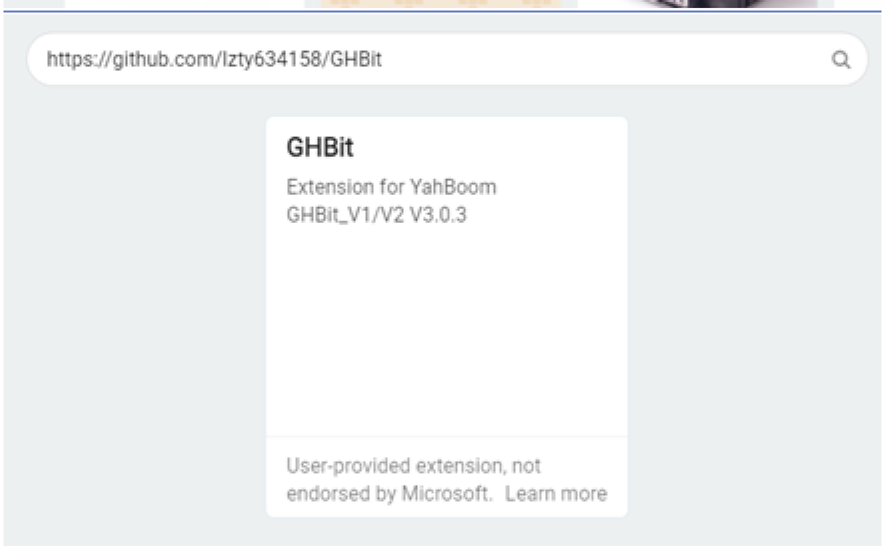

Top
Screw
Bottom
Screw pad

Exercise 1



Exercise 2



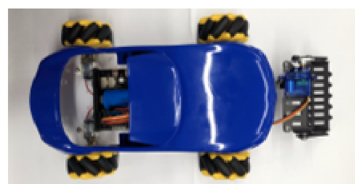
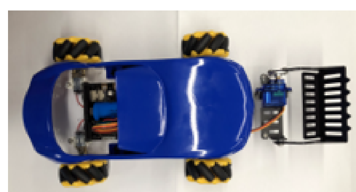


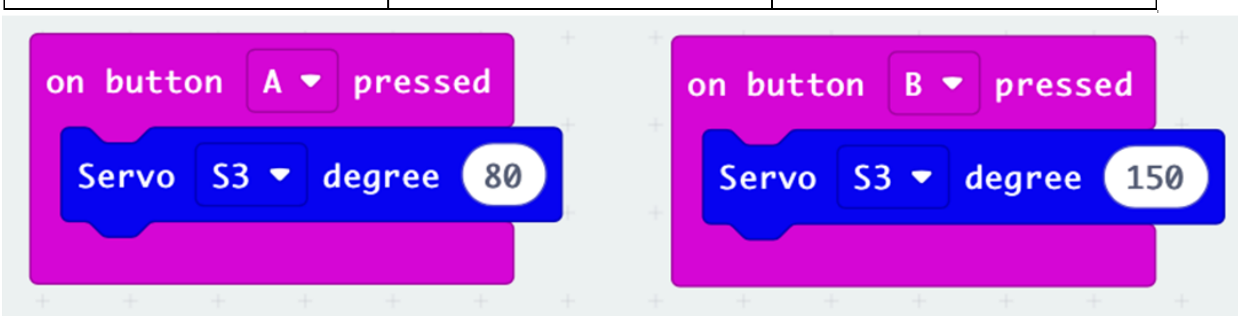
- B1 and B4 bottom control the horizontal movement of left and right
- B2 and B3 bottom control the load or unload of loader
- Control rods control the back and forward and turning left or right
- Car stops when pending of remote
- When load or unload, DoReMi and SoFaMi will play respectively (or any other melody

- Refer to Beginner, Lesson 10, Receive text as operation condition
- Use of sound effect >> perform melody speed (bpm) in block module

Answer

Exercise 1

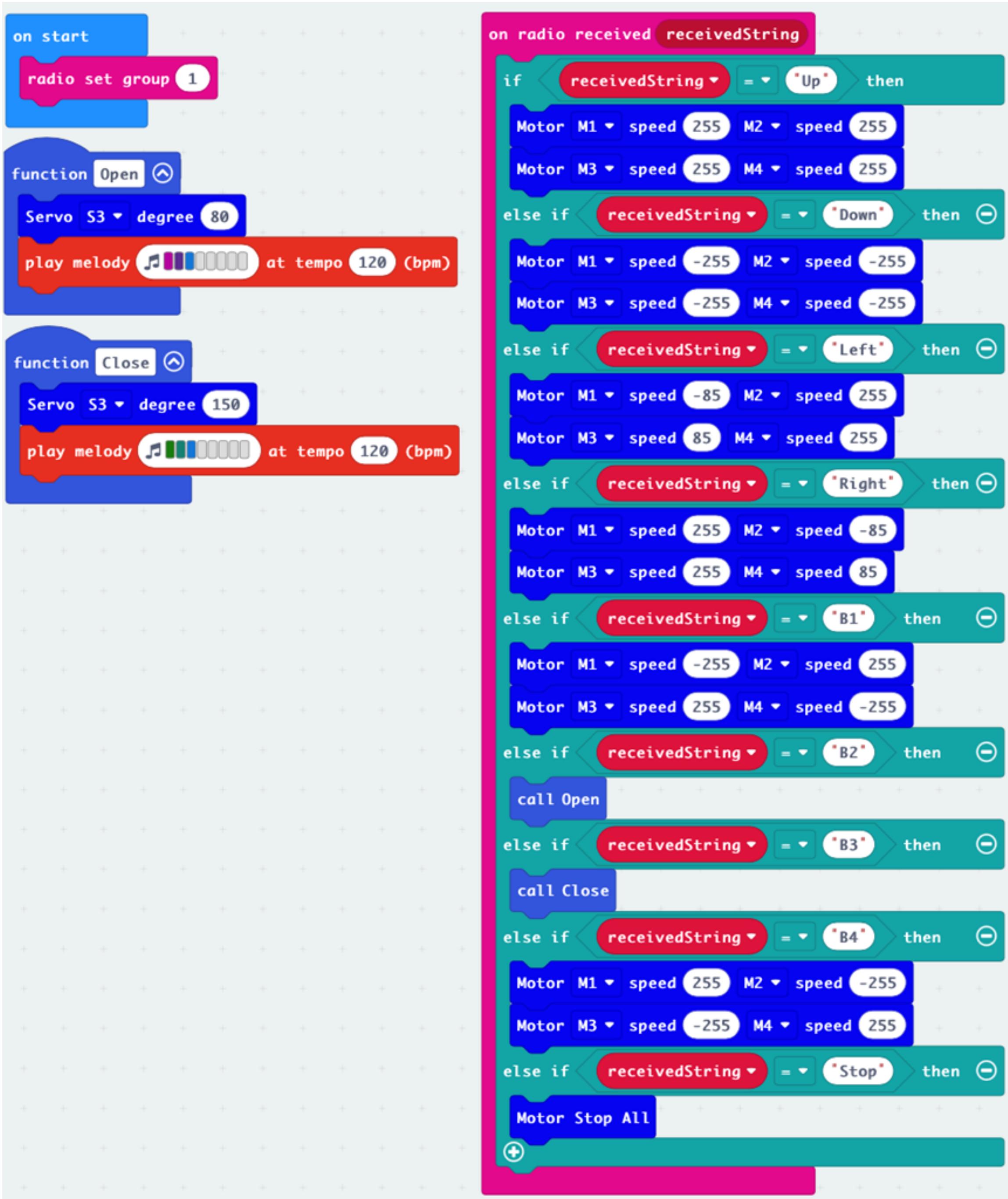
Angle value	Loader movement	
Increase	Unload	
Decrease	Load	



Exercise 2

Program of remote

Program of Car

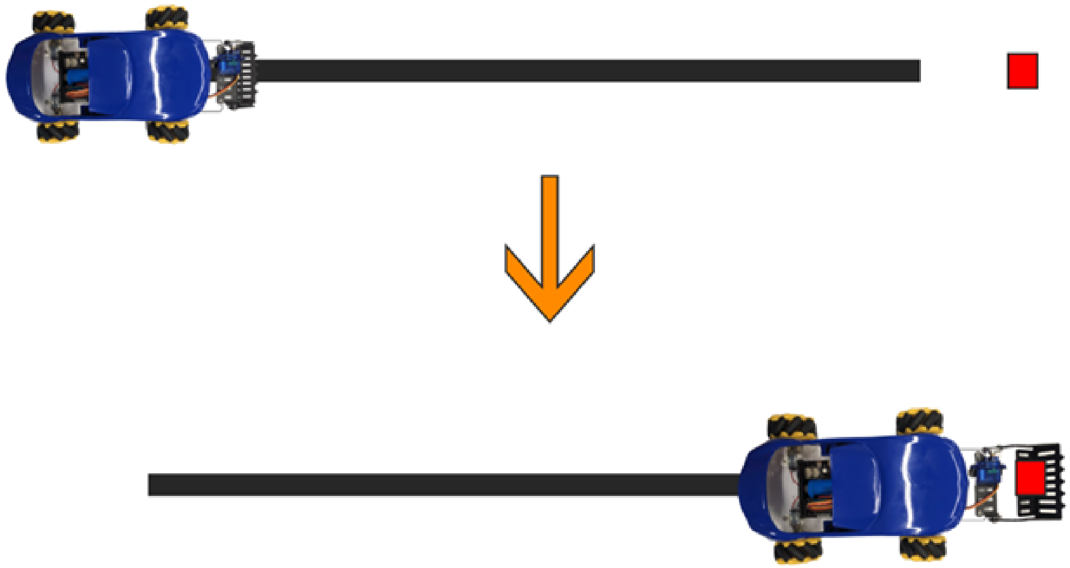


1.4.4 Lesson 4

Introduction

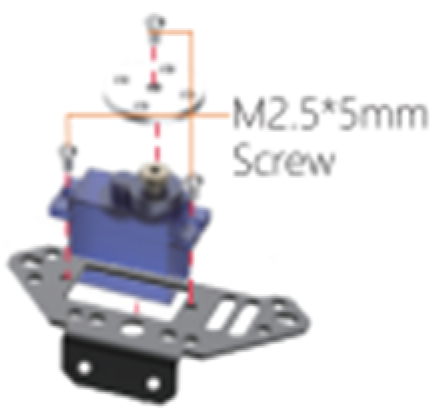
Learning Target

Exercise 1

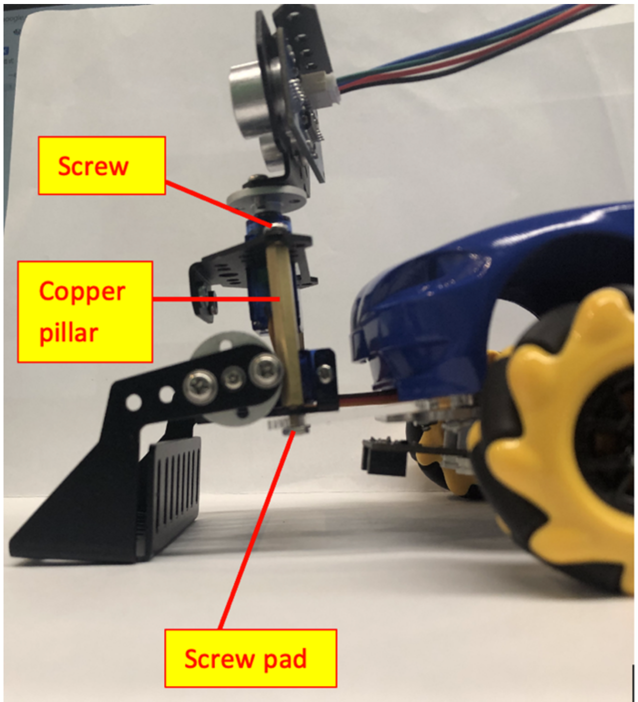


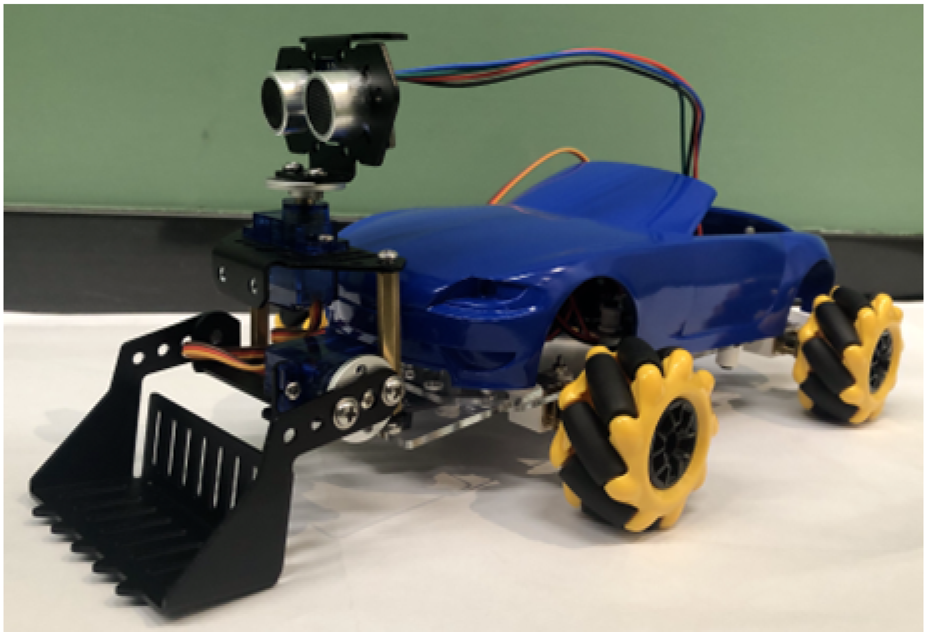
To install the ultrasonic sensor

	
universal expansion mounting plate	pan-tilt-zoom mount plate

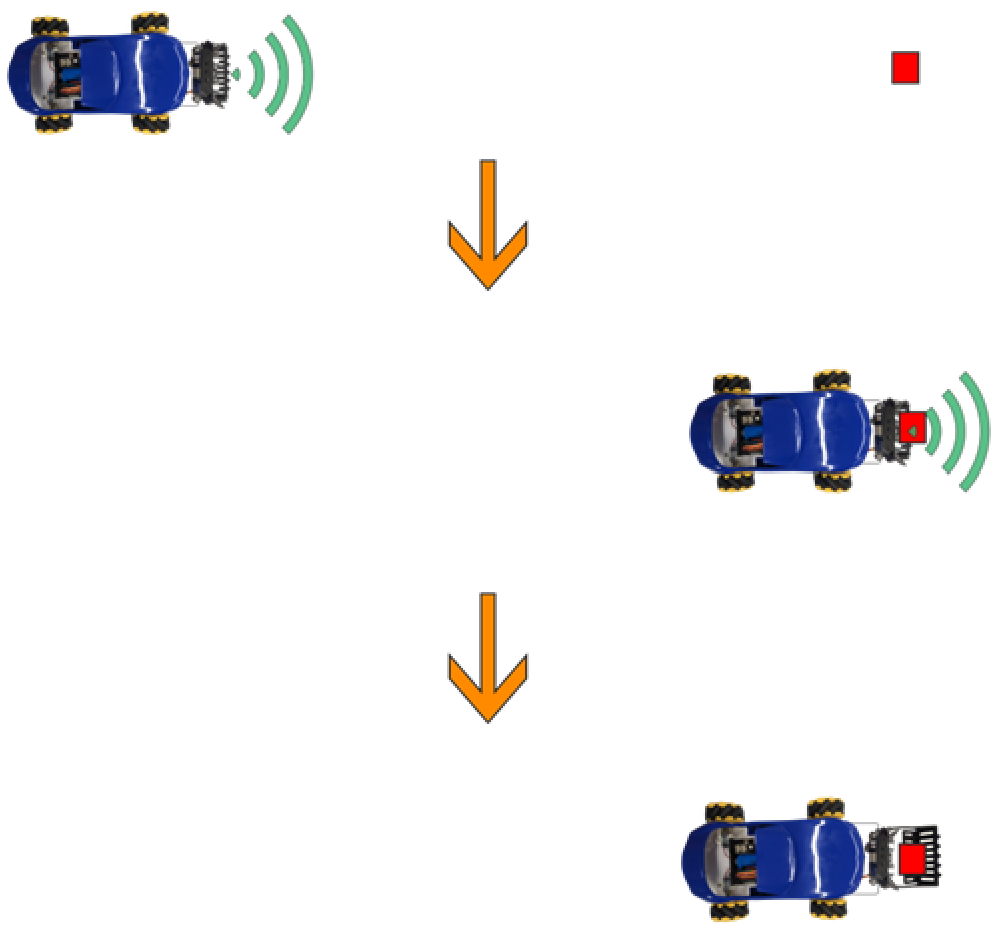



M3*5mm
Screw

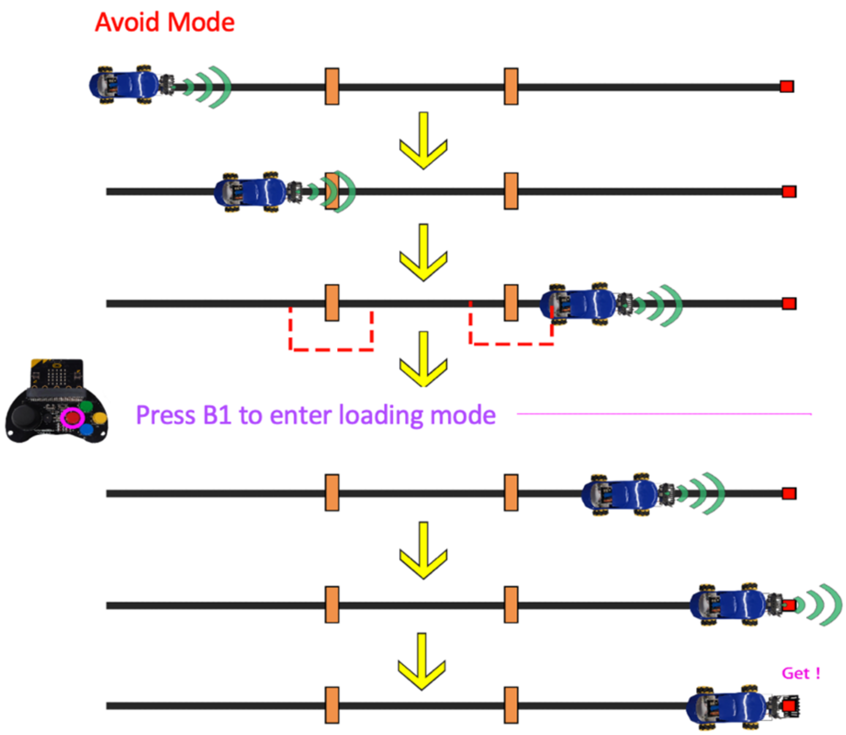

Screw
Copper
pillar
Screw pad



Exercise 2

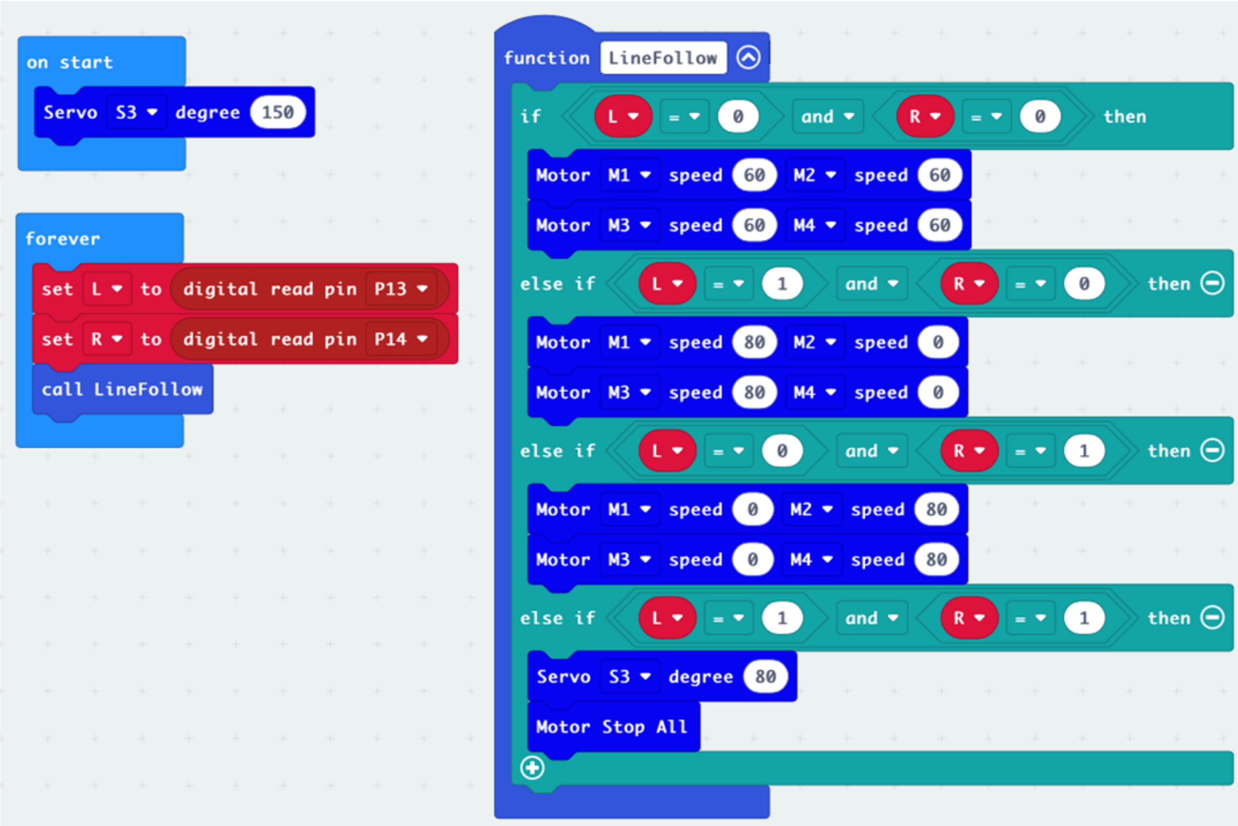


Exercise 3

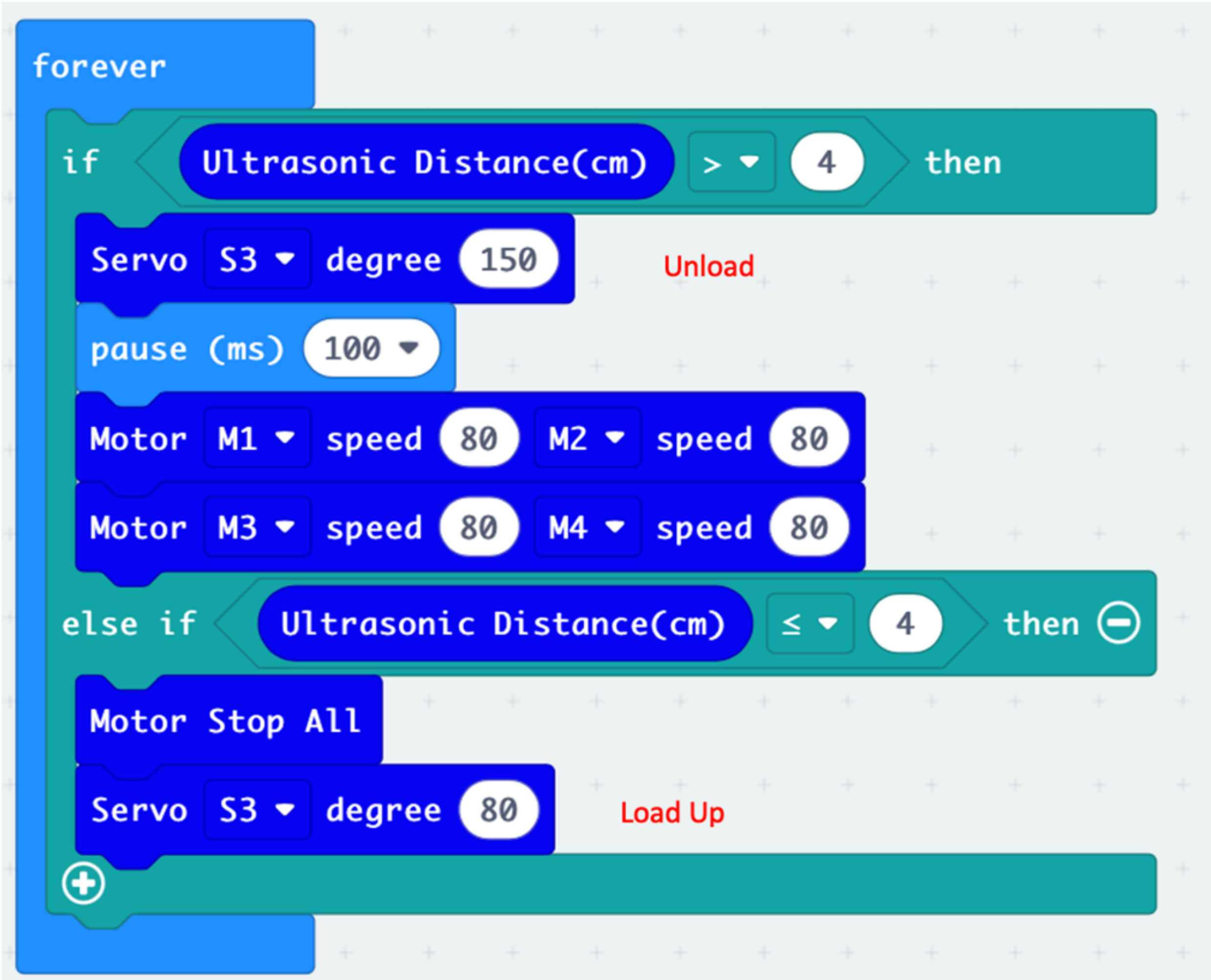


Answer

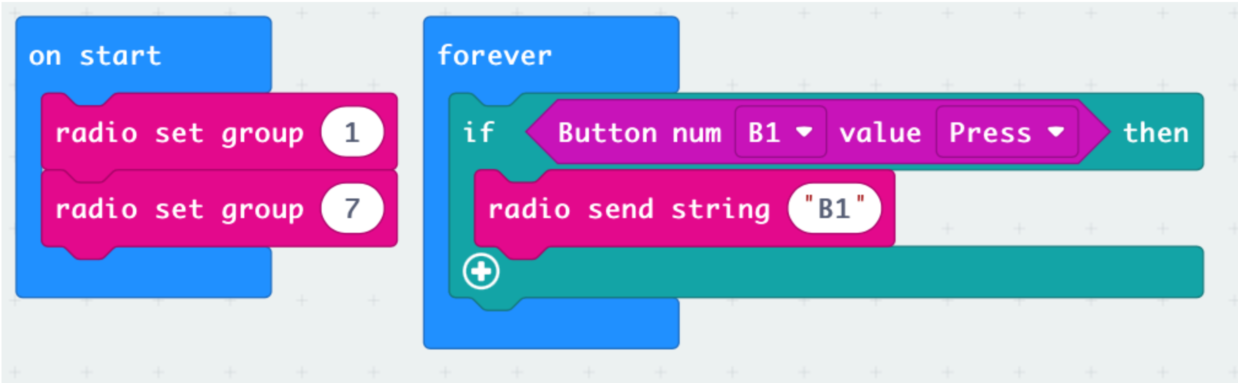
Exercise 1

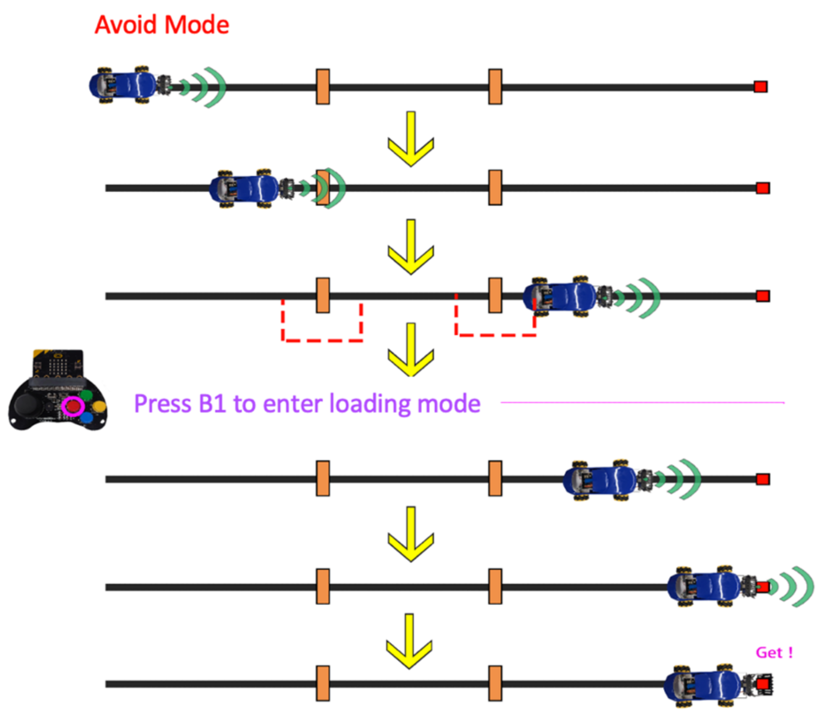

on start
Servo S3 degree 150
forever
set L to digital read pin P13
set R to digital read pin P14
call LineFollow
function LineFollow
if L = 0 and R = 0 then
Motor M1 speed 60 M2 speed 60
Motor M3 speed 60 M4 speed 60
else if L = 1 and R = 0 then
Motor M1 speed 80 M2 speed 0
Motor M3 speed 80 M4 speed 0
else if L = 0 and R = 1 then
Motor M1 speed 0 M2 speed 80
Motor M3 speed 0 M4 speed 80
else if L = 1 and R = 1 then
Servo S3 degree 80
Motor Stop All

Exercise 2

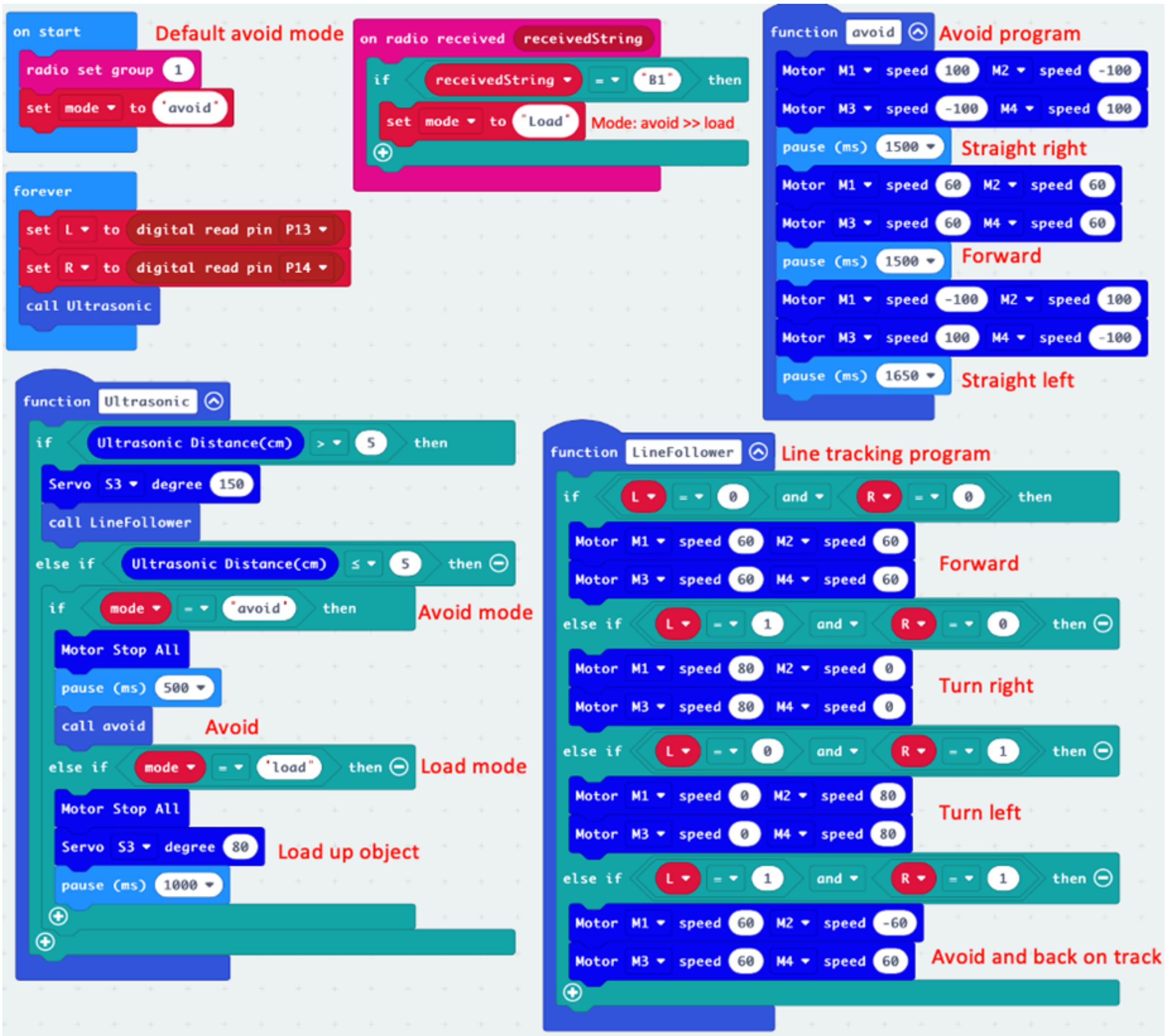

forever
if
Ultrasonic Distance(cm)
>
4
then
Servo
S3
degree
150
Unload
pause (ms)
100
Motor
M1
speed
80
M2
speed
80
Motor
M3
speed
80
M4
speed
80
else if
Ultrasonic Distance(cm)
≤
4
then
Motor Stop All
Servo
S3
degree
80
Load Up

Exercise 3


on start
radio set group 1
radio set group 7
forever
if Button num B1 value Press then
radio send string "B1"


Avoid Mode
Press B1 to enter loading mode
Get !

The program of the car

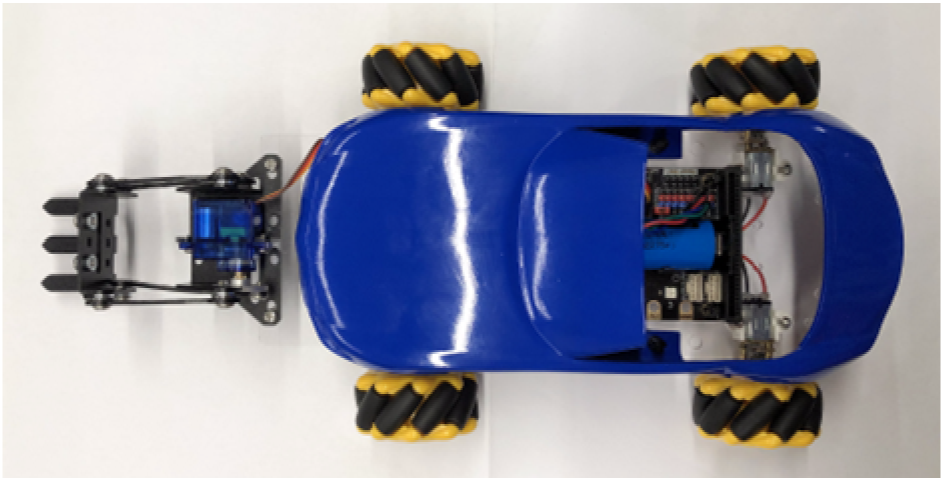


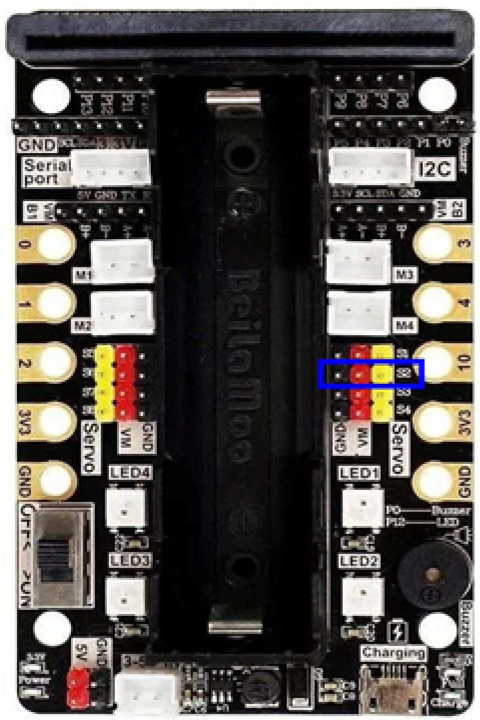
1.4.5 Lesson 5

Introduction

Learning Target

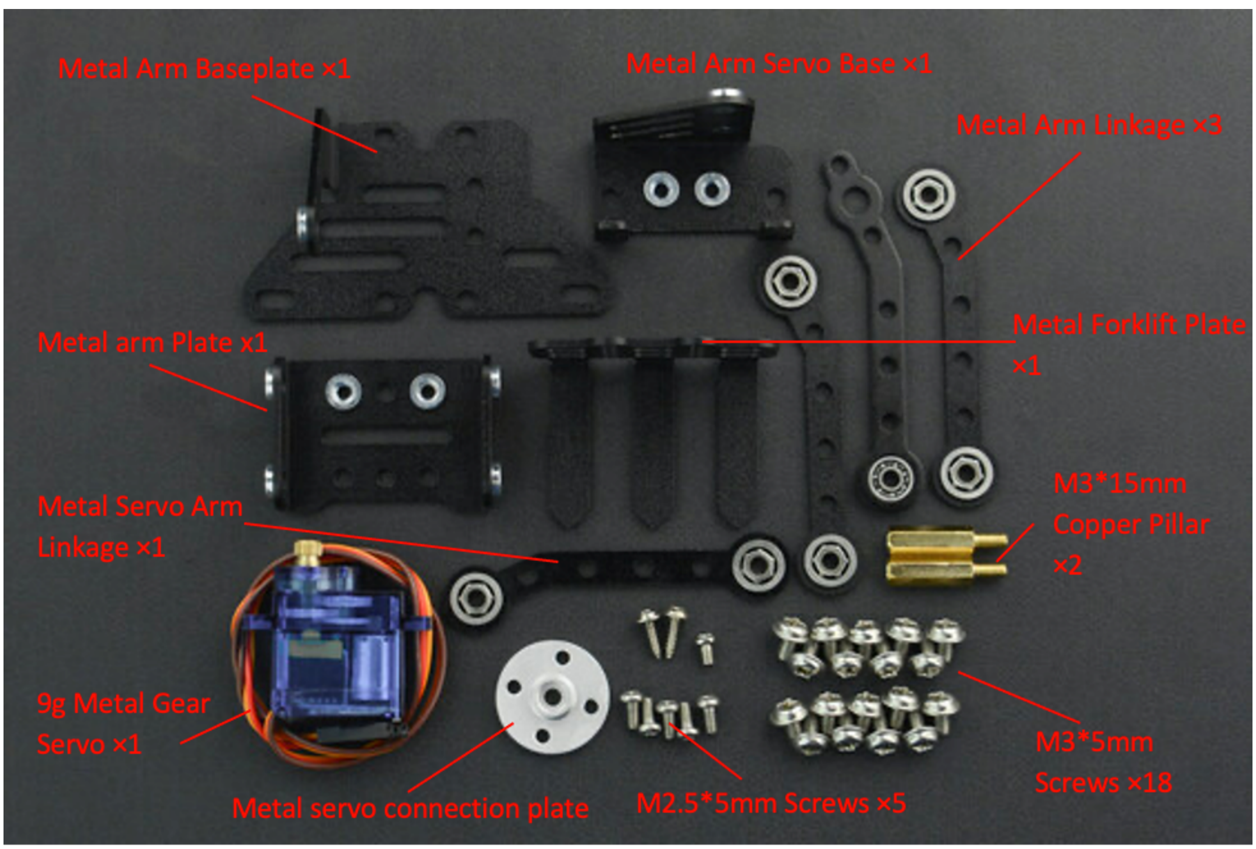
Meet the Micro:bit Expansion Tool - Forklift

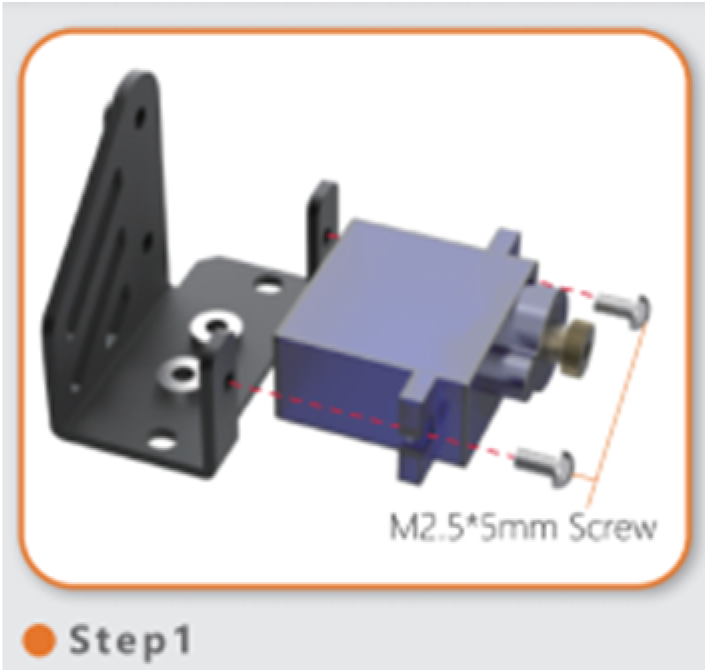


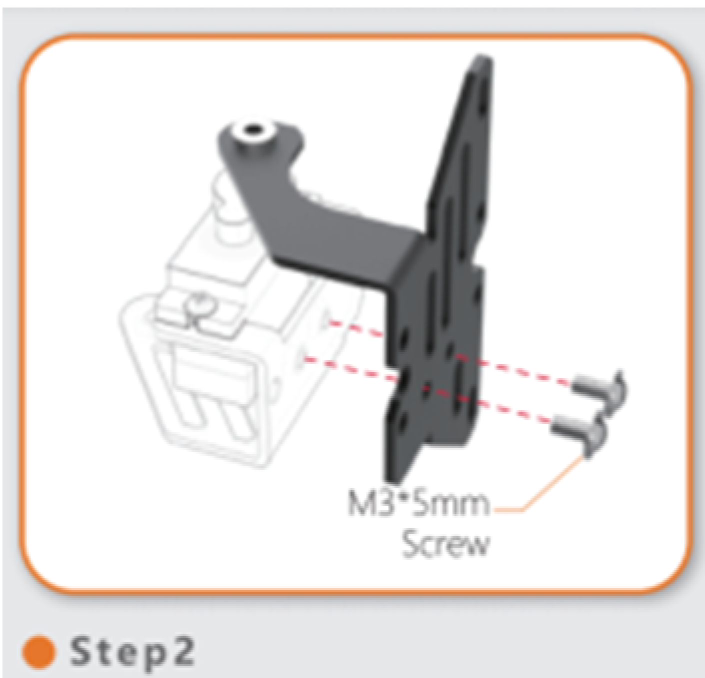


The principle and function of forklift

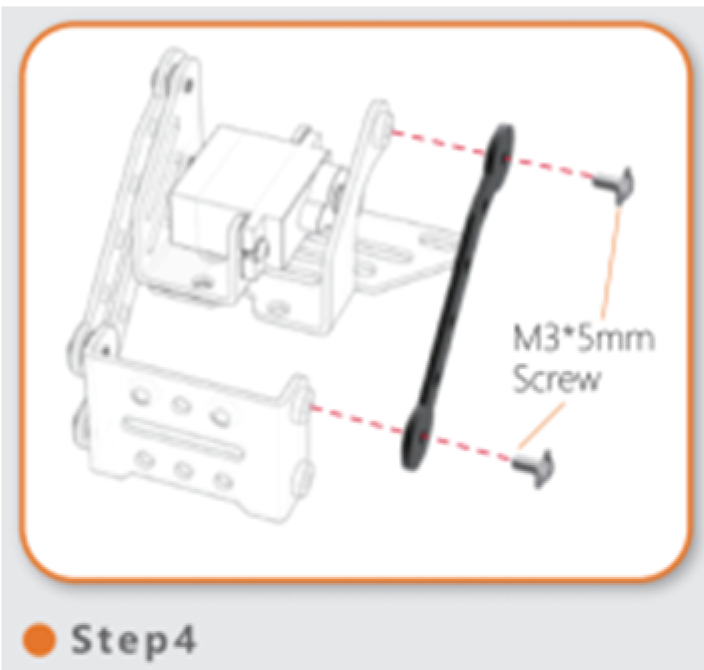
Install the forklift

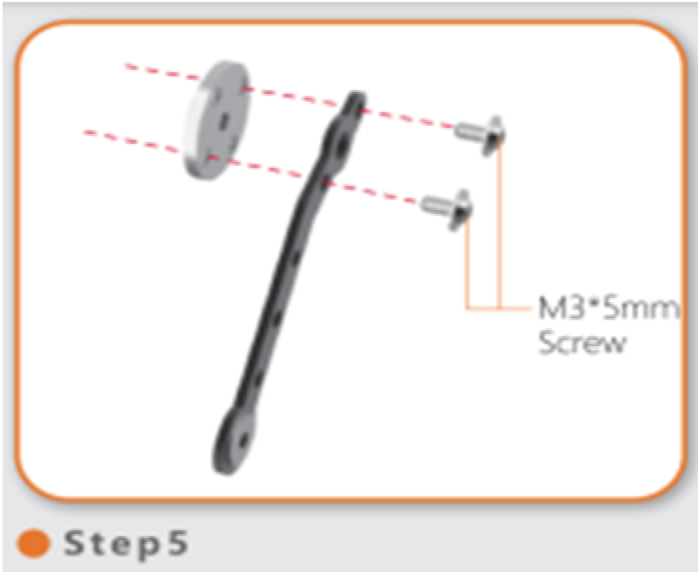


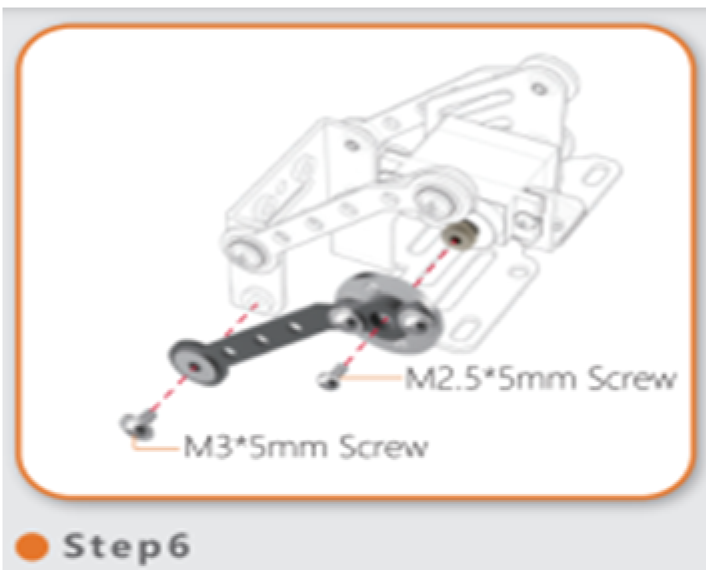

M2.5*5mm Screw
Step1


M3*5mm
Screw
Step2

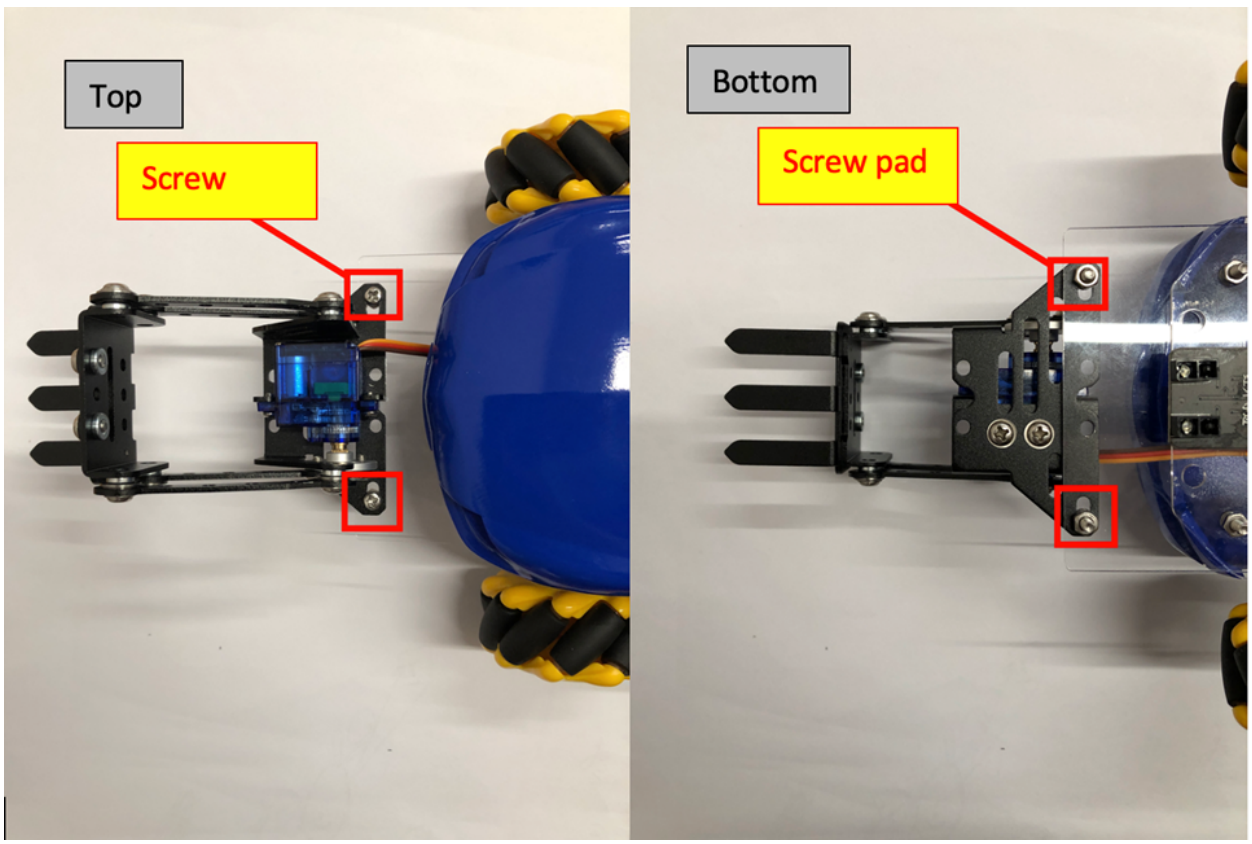

M3*5mm
Screw
Step3


M3*5mm
Screw
Step4

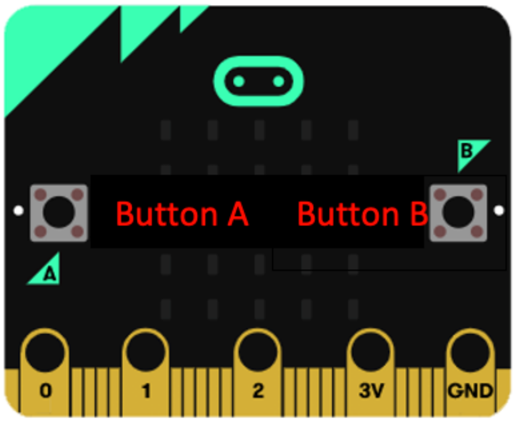

M3*5mm
Screw
Step5


M2.5*5mm Screw
M3*5mm Screw
Step6


M3*5mm Screw
Step7


Top
Screw
Bottom
Screw pad

Exercise 1

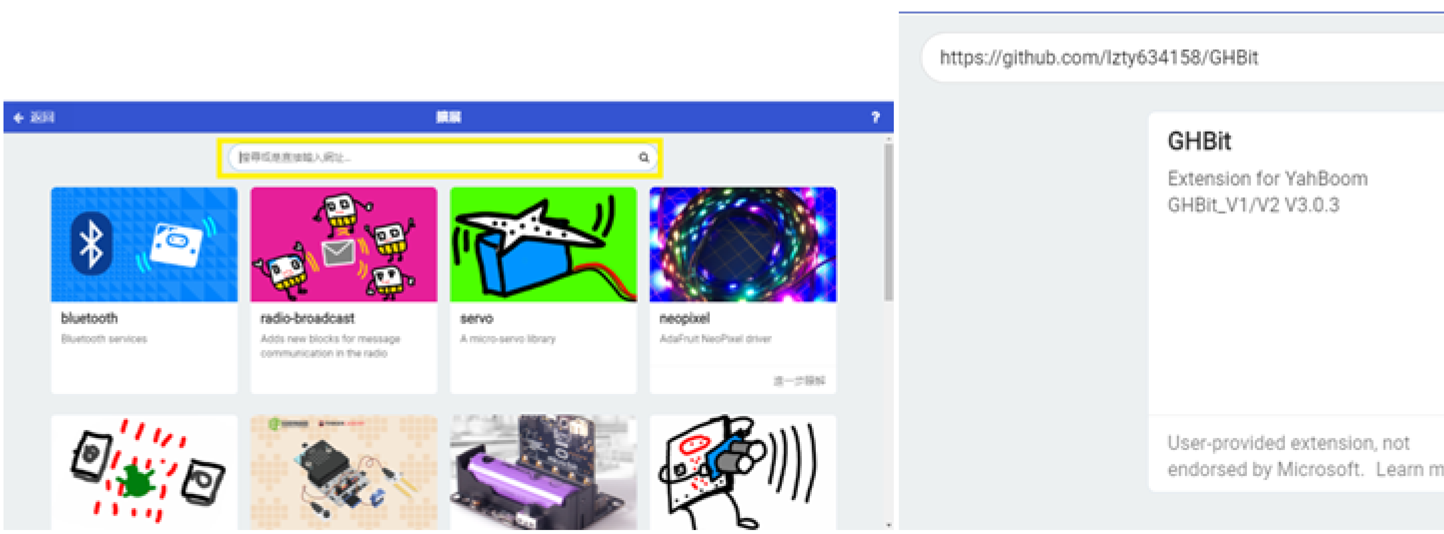


Exercise 2

Find items suitable for forklift handling and calculate their angles

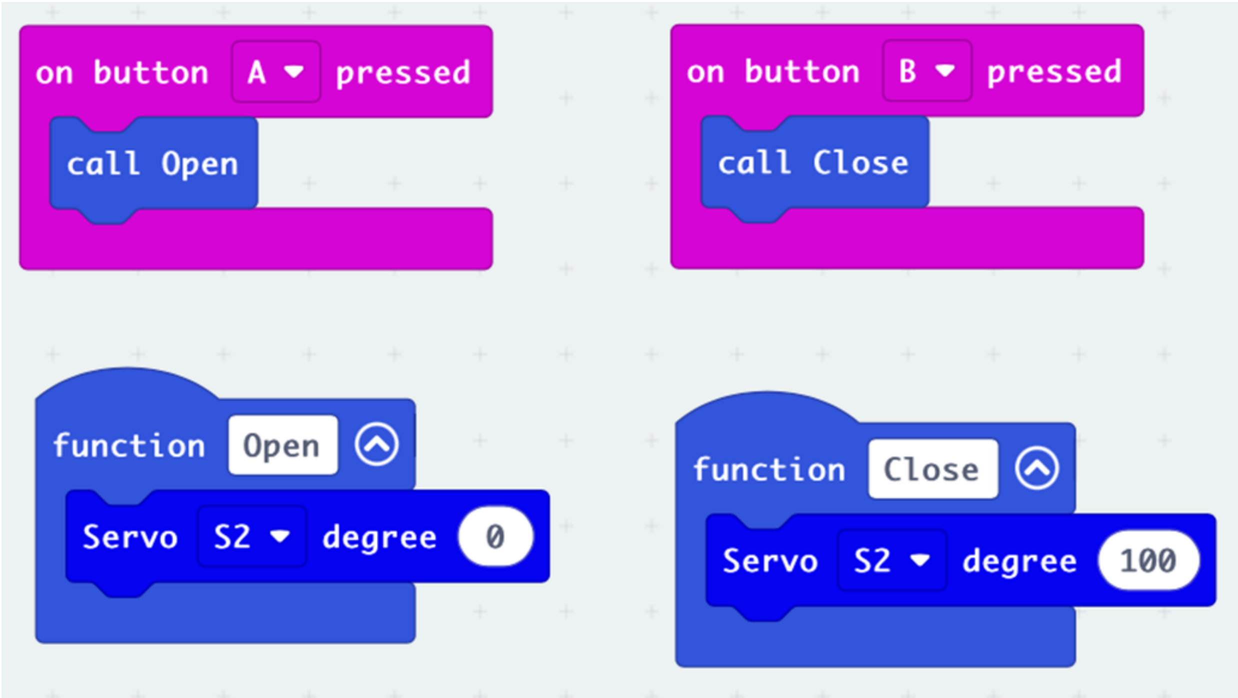
(**The highest height that the forklift can lift is 6cm, pay attention to the size of the object selected**)

Exercise 3



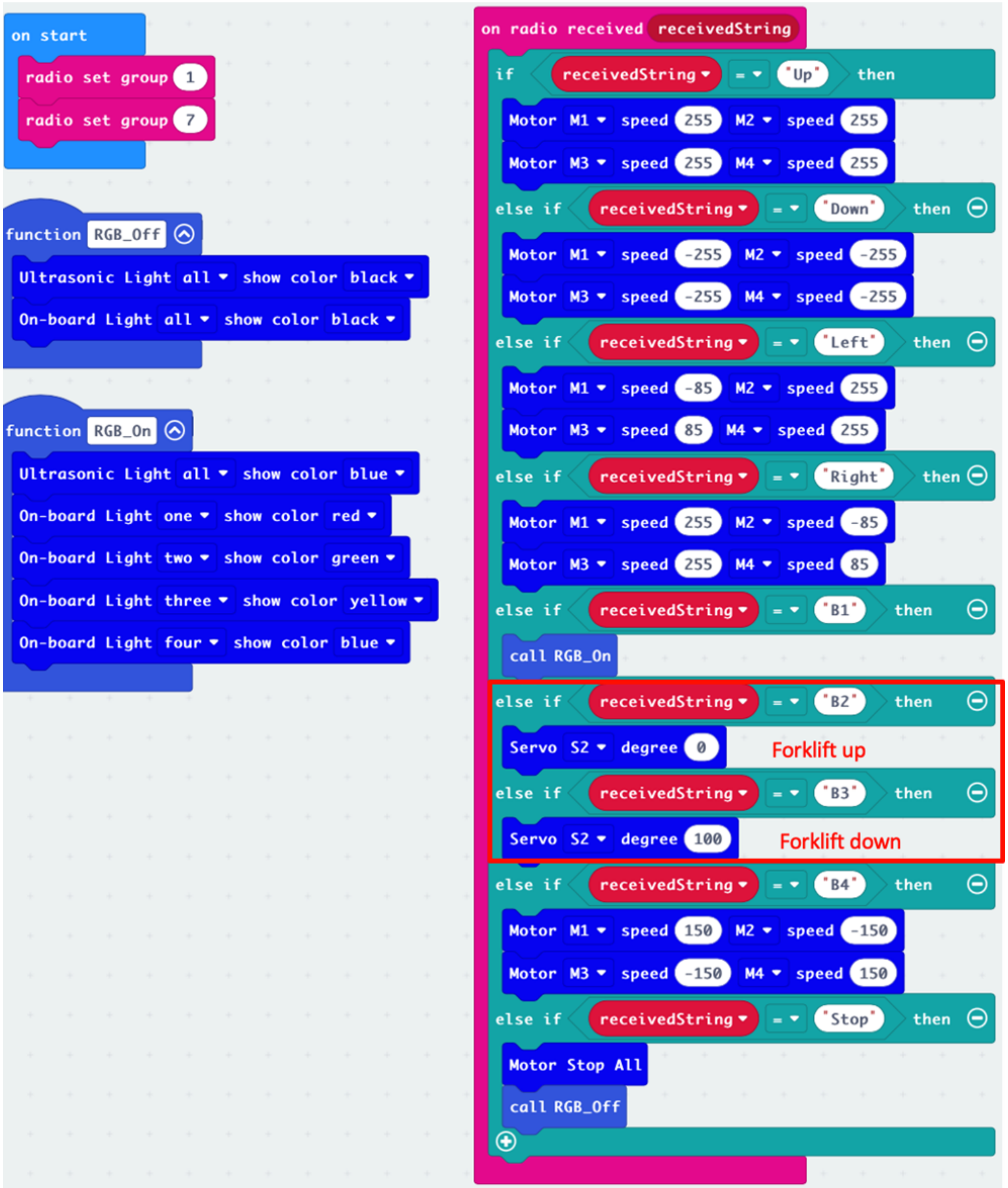
Answer

Exercise 1:



Exercise 3:

Car program


on start
radio set group 1
radio set group 7
function RGB_Off
Ultrasonic Light all show color black
On-board Light all show color black
function RGB_On
Ultrasonic Light all show color blue
On-board Light one show color red
On-board Light two show color green
On-board Light three show color yellow
On-board Light four show color blue
on radio received receivedString
if receivedString = "Up" then
Motor M1 speed 255 M2 speed 255
Motor M3 speed 255 M4 speed 255
else if receivedString = "Down" then
Motor M1 speed -255 M2 speed -255
Motor M3 speed -255 M4 speed -255
else if receivedString = "Left" then
Motor M1 speed -85 M2 speed 255
Motor M3 speed 85 M4 speed 255
else if receivedString = "Right" then
Motor M1 speed 255 M2 speed -85
Motor M3 speed 255 M4 speed 85
else if receivedString = "B1" then
call RGB_On
else if receivedString = "B2" then
Servo S2 degree 0
Forklift up
else if receivedString = "B3" then
Servo S2 degree 100
Forklift down
else if receivedString = "B4" then
Motor M1 speed 150 M2 speed -150
Motor M3 speed -150 M4 speed 150
else if receivedString = "Stop" then
Motor Stop All
call RGB_Off

Remote programm

1.4.6 Lesson 6

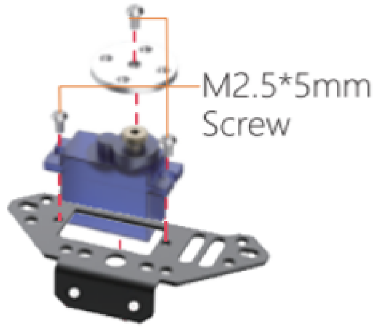
Introduction

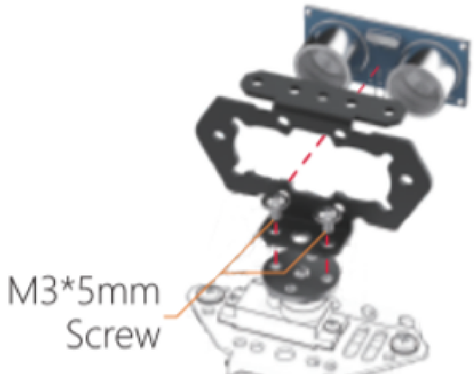
Learning Target

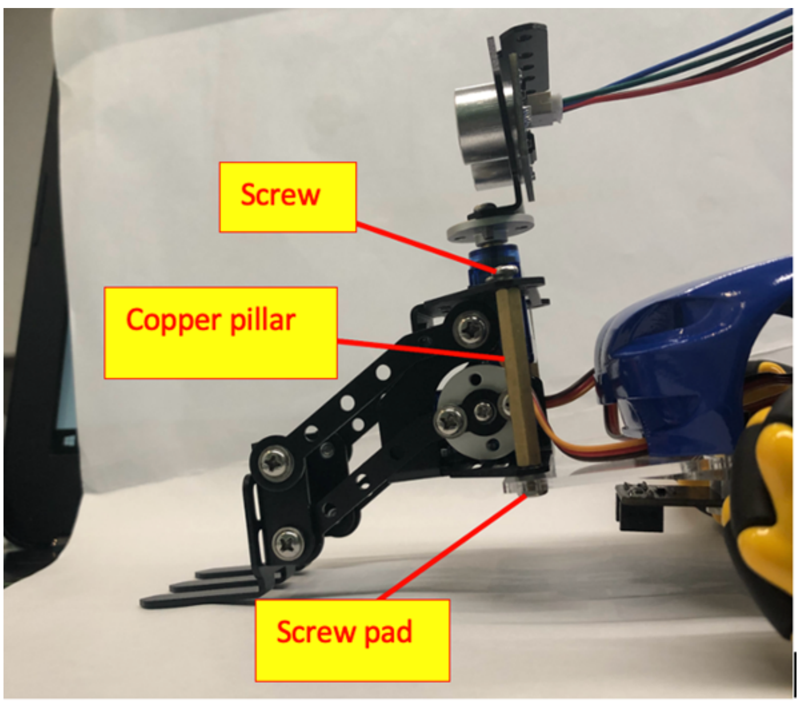
Use of forklift tools

To install the ultrasonic sensor

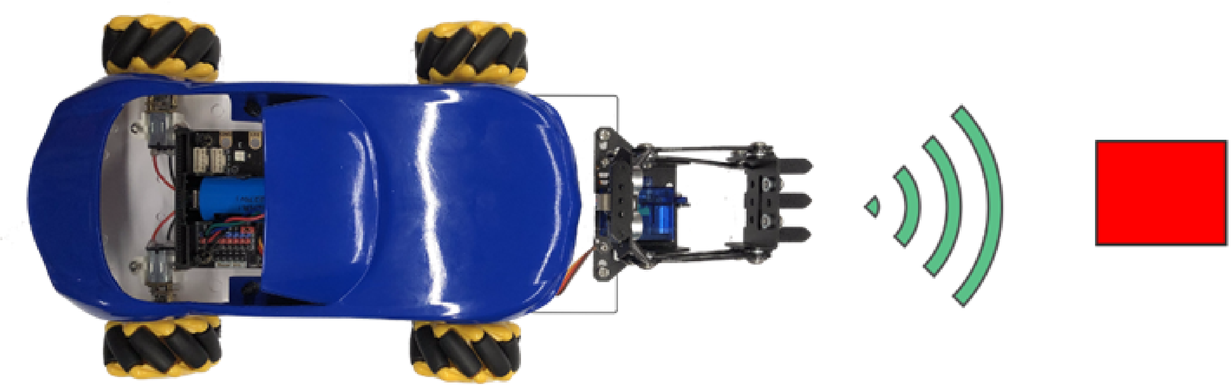
	
universal expansion mounting plate	pan-tilt-zoom mount plate


M2.5*5mm
Screw


M3*5mm
Screw


Screw
Copper pillar
Screw pad

Exercise 1

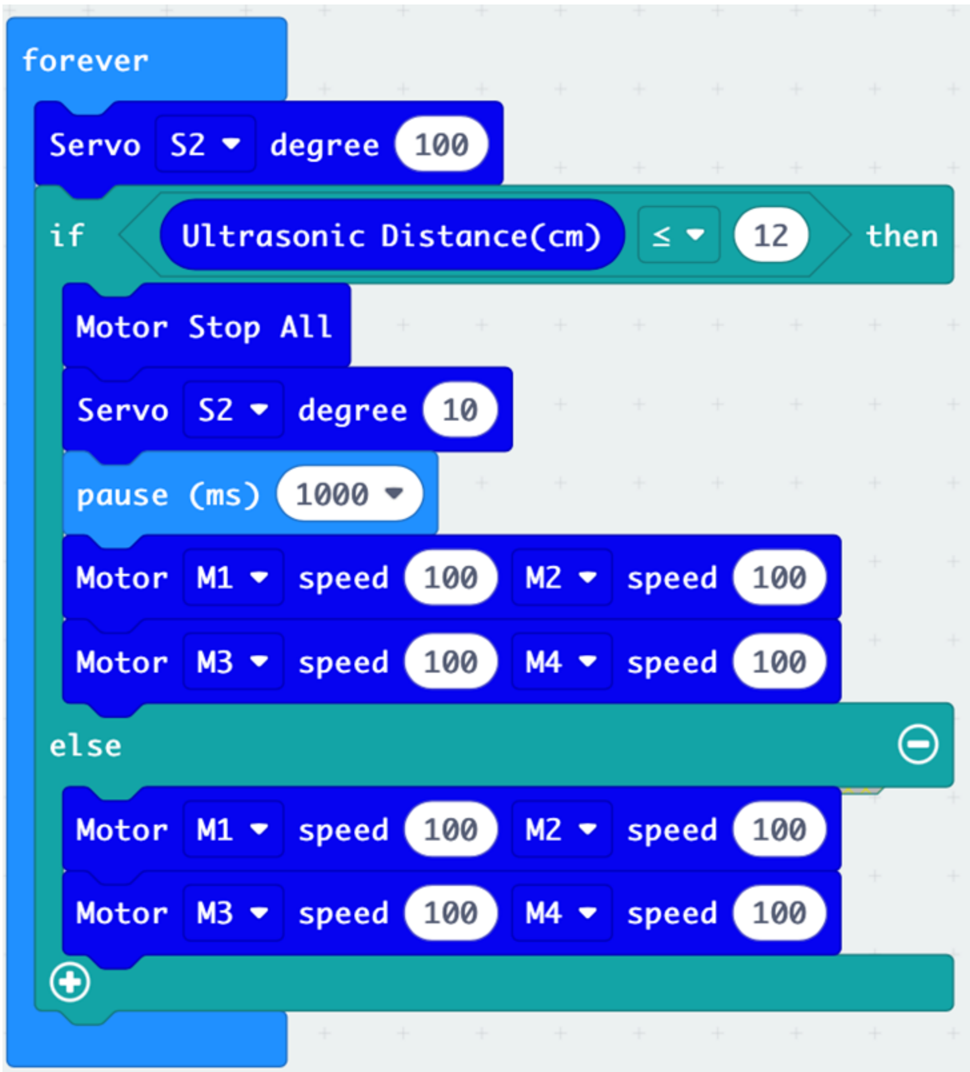


Exercise 2

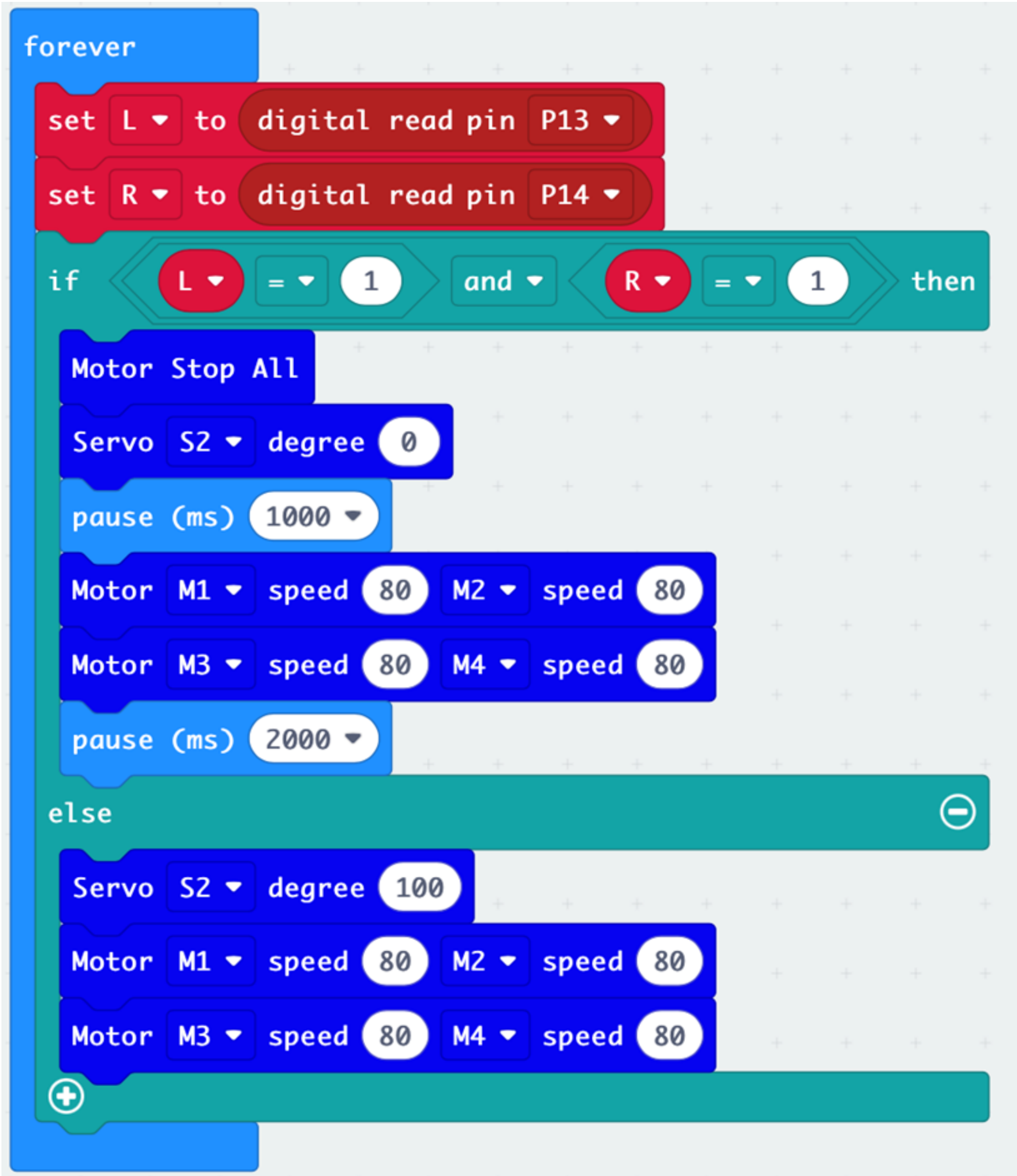
Exercise 3

Answer

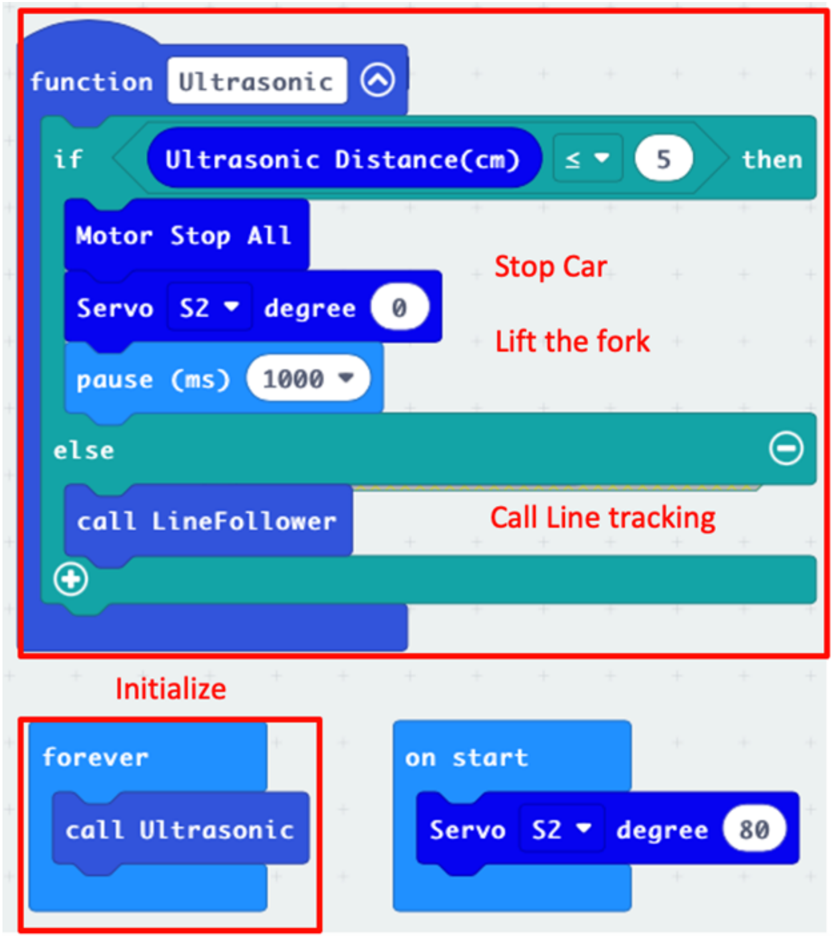
Exercise 1



Exercise 2


forever
set L to digital read pin P13
set R to digital read pin P14
if L = 1 and R = 1 then
Motor Stop All
Servo S2 degree 0
pause (ms) 1000
Motor M1 speed 80 M2 speed 80
Motor M3 speed 80 M4 speed 80
pause (ms) 2000
else
Servo S2 degree 100
Motor M1 speed 80 M2 speed 80
Motor M3 speed 80 M4 speed 80

Exercise 3


function Ultrasonic
if Ultrasonic Distance(cm) ≤ 5 then
Motor Stop All
Stop Car
Servo S2 degree 0
Lift the fork
pause (ms) 1000
else
call LineFollower
Call Line tracking
Initialize
forever
call Ultrasonic
on start
Servo S2 degree 80

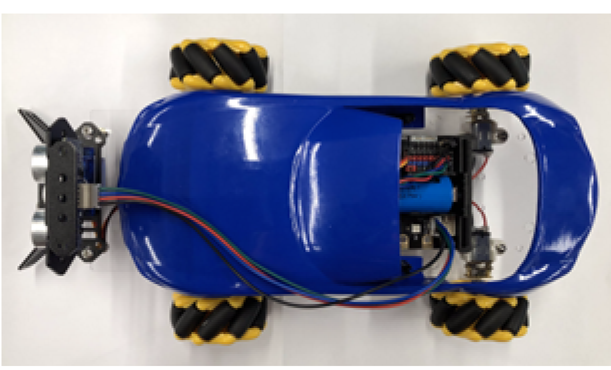
Line tracking module

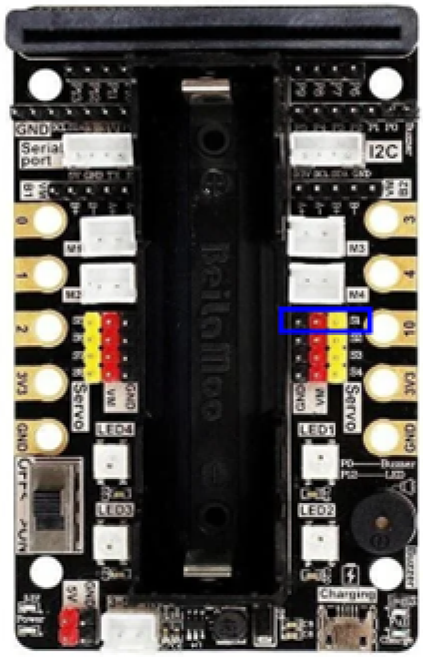
1.4.7 Lesson 7

Introduction

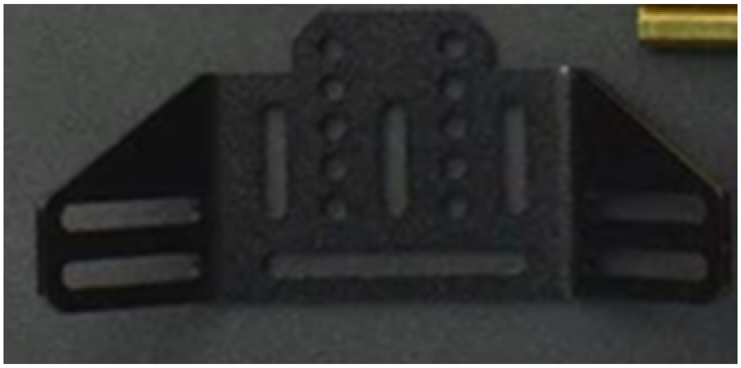
Learning Target

Meet the Micro:bit Expansion Tool - Bulldozer

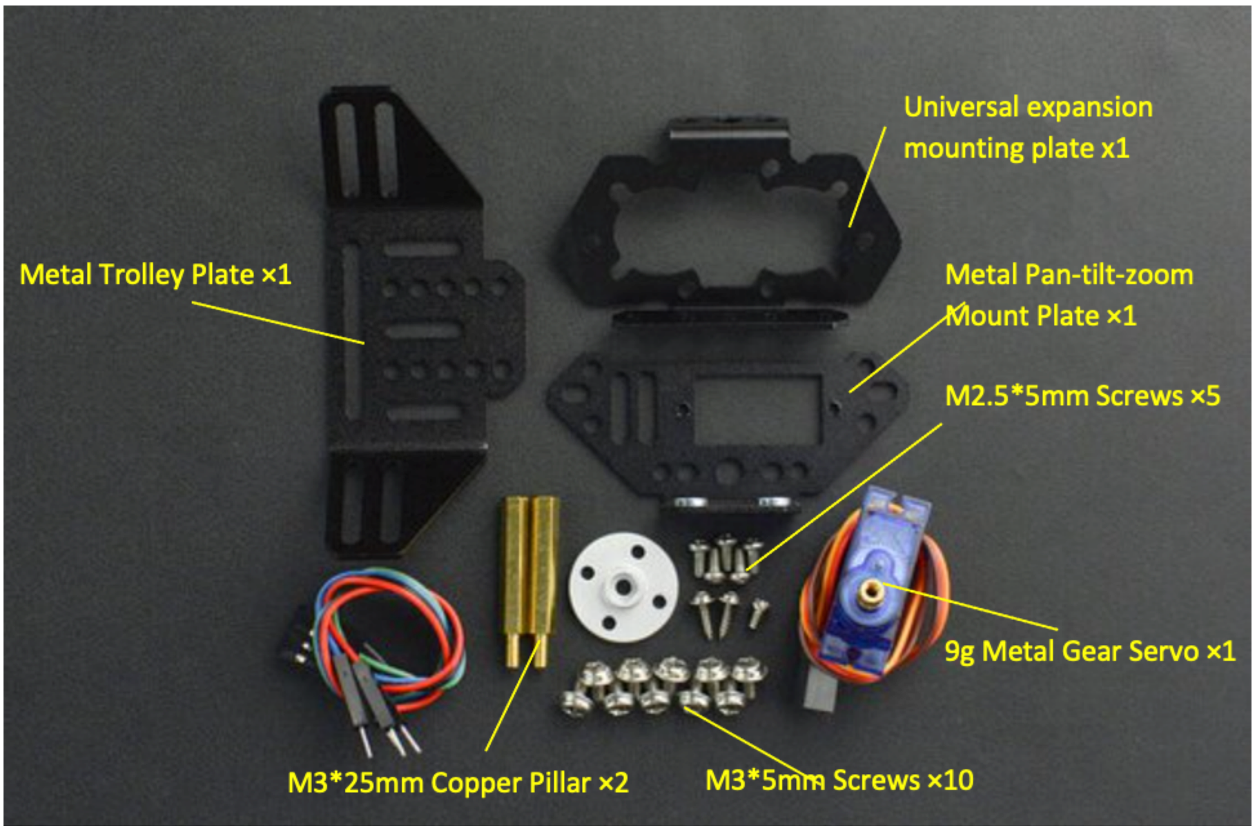




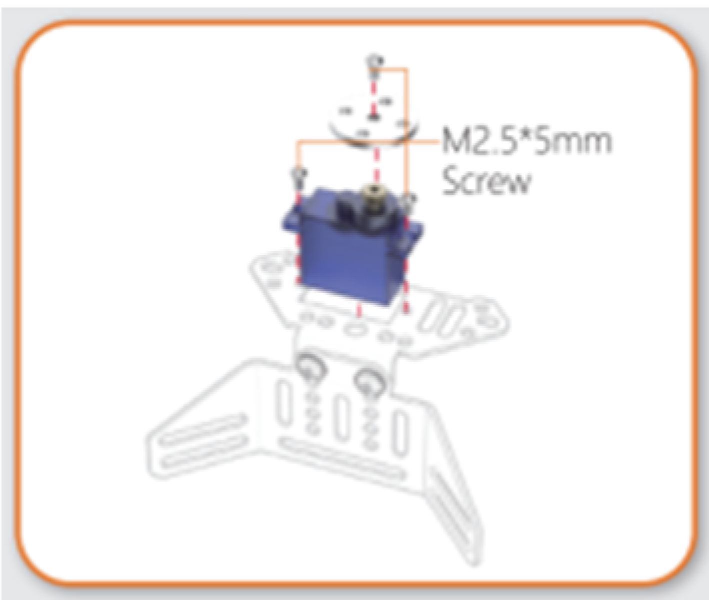
Principle and function of mechanical bulldozer



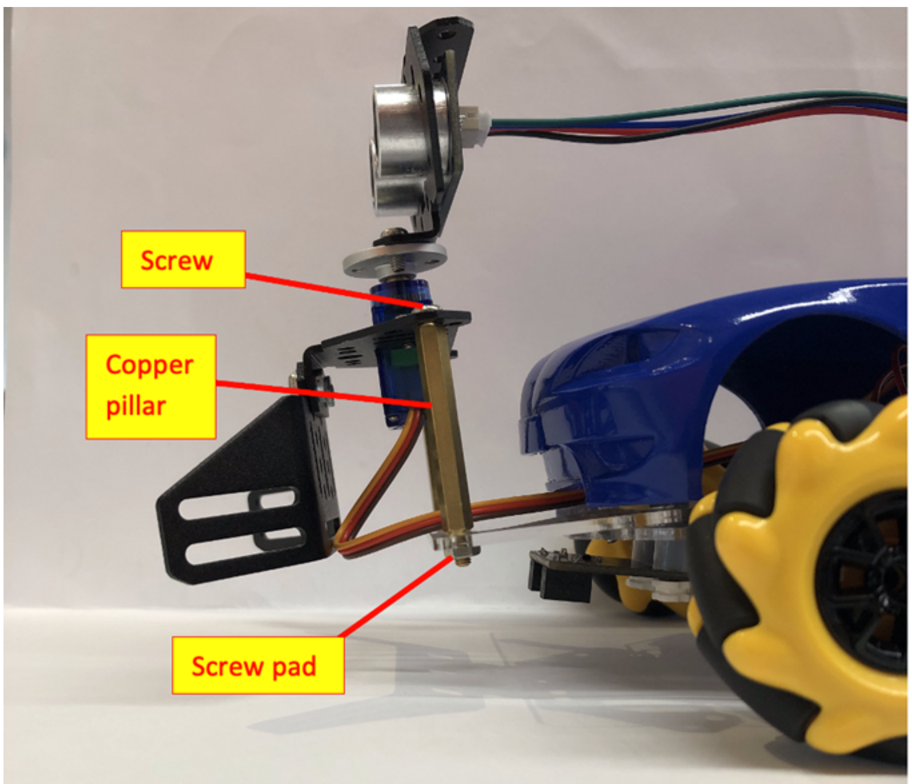
To install the bulldozer

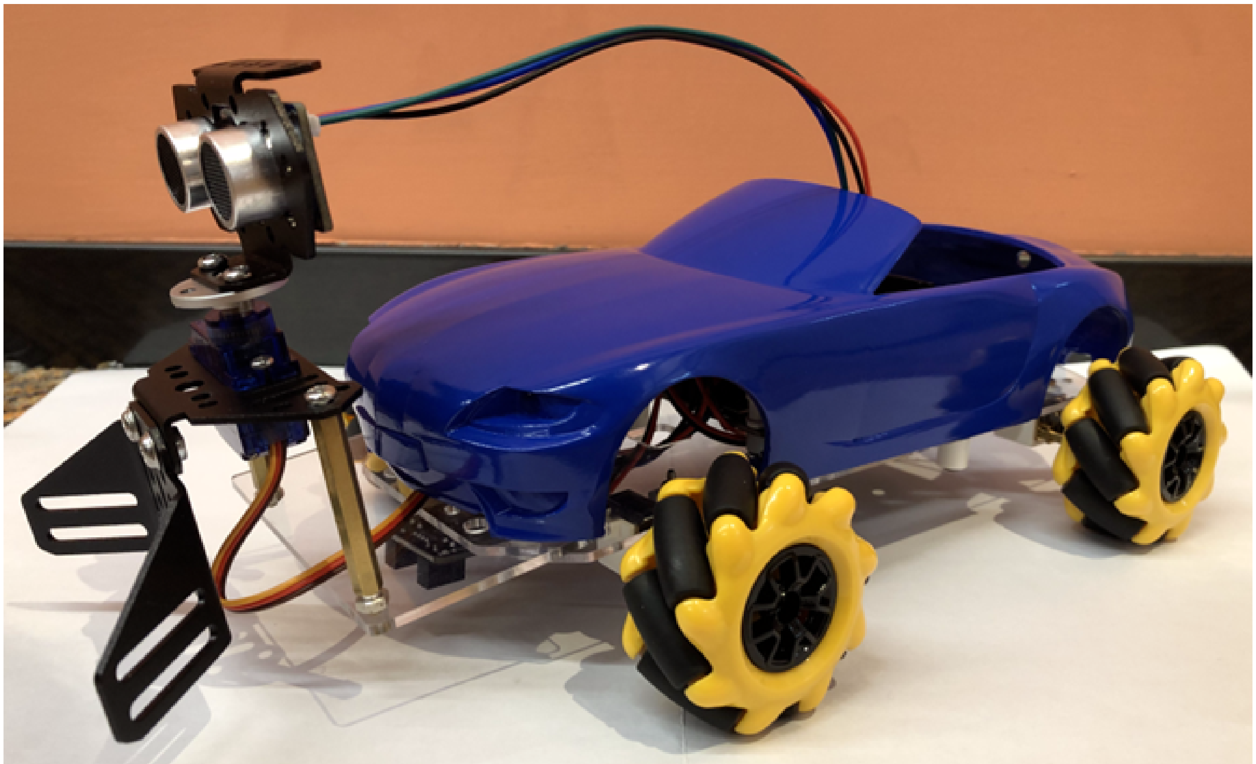


Components – with an ultrasonic sensor:

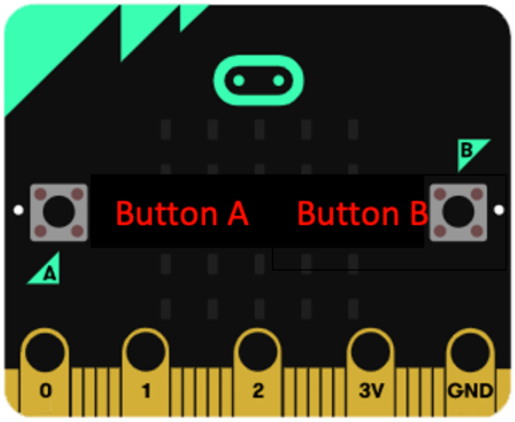



M3*5mm
Screw


Screw
Copper
pillar
Screw pad

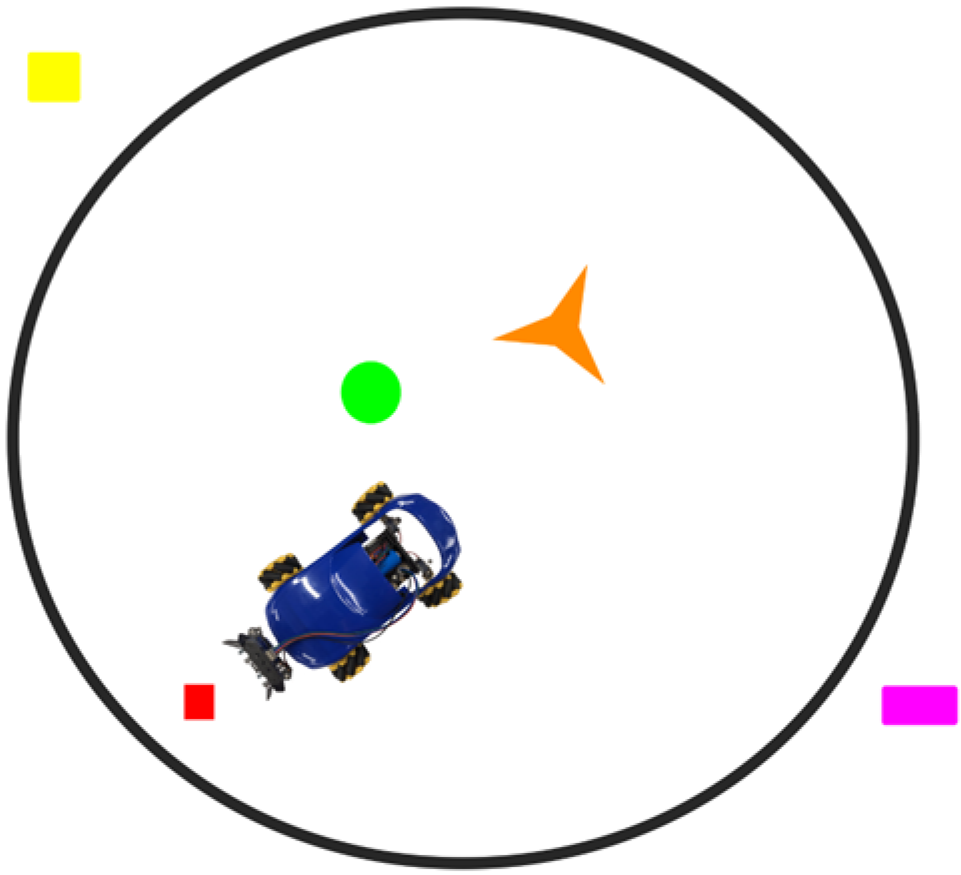


Exercise 1



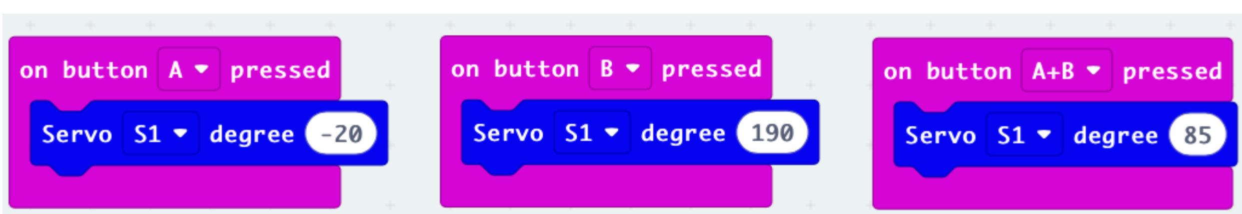
- First adjust the installation angle of the servo so that the angle value of the servo module is 85 as the front
- Only using the servo module 0-180 degrees cannot make the servo ultrasonic sensor look to the left and right
- Negative numbers and values above 180 degrees must be tried

Exercise 2

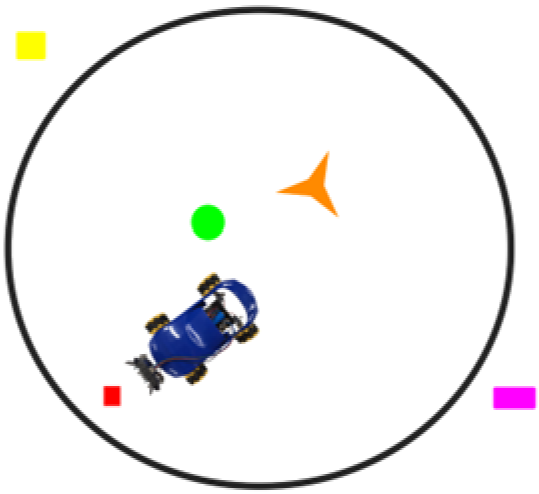


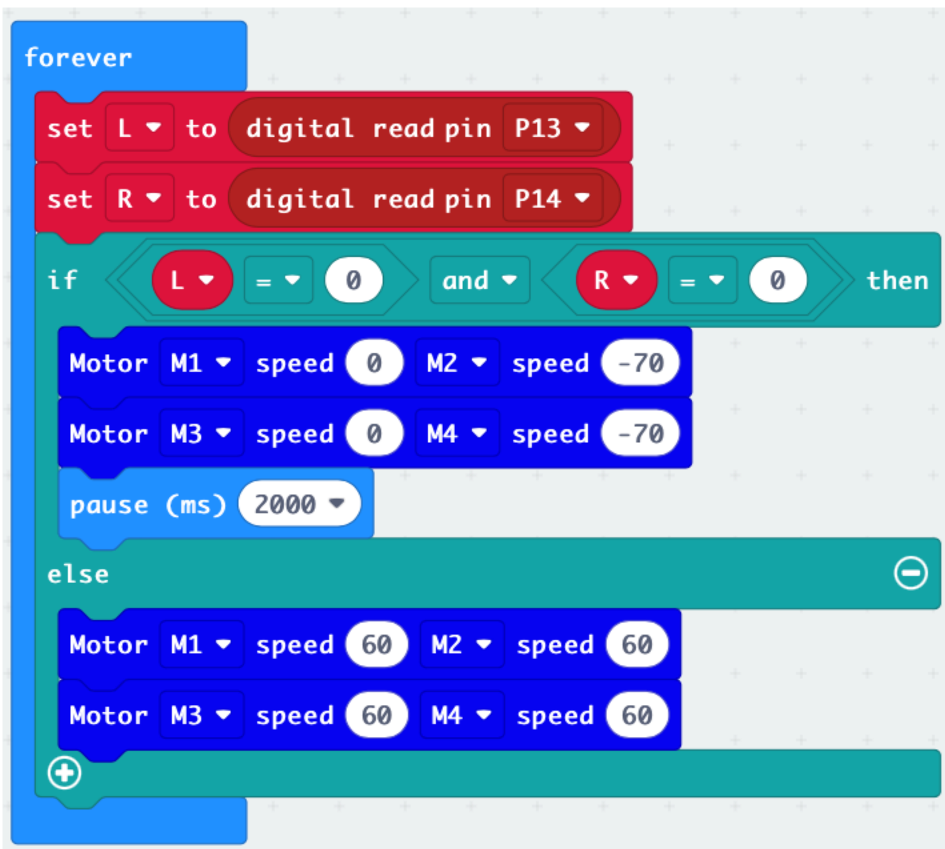
Answer

Exercise 1



Exercise 2




forever
set L to digital read pin P13
set R to digital read pin P14
if L = 0 and R = 0 then
Motor M1 speed 0 M2 speed -70
Motor M3 speed 0 M4 speed -70
pause (ms) 2000
else
Motor M1 speed 60 M2 speed 60
Motor M3 speed 60 M4 speed 60

1.4.8 Lesson 8

Introduction

Learning Target

Use of bulldozer tools

Exercise 1

Shoot the gantry

Football match

```
1. The car or extension tool cannot exceed the line in front of the gantry
2. The speed cannot exceed 150
3. Do not touch the car during the race
```

Answer

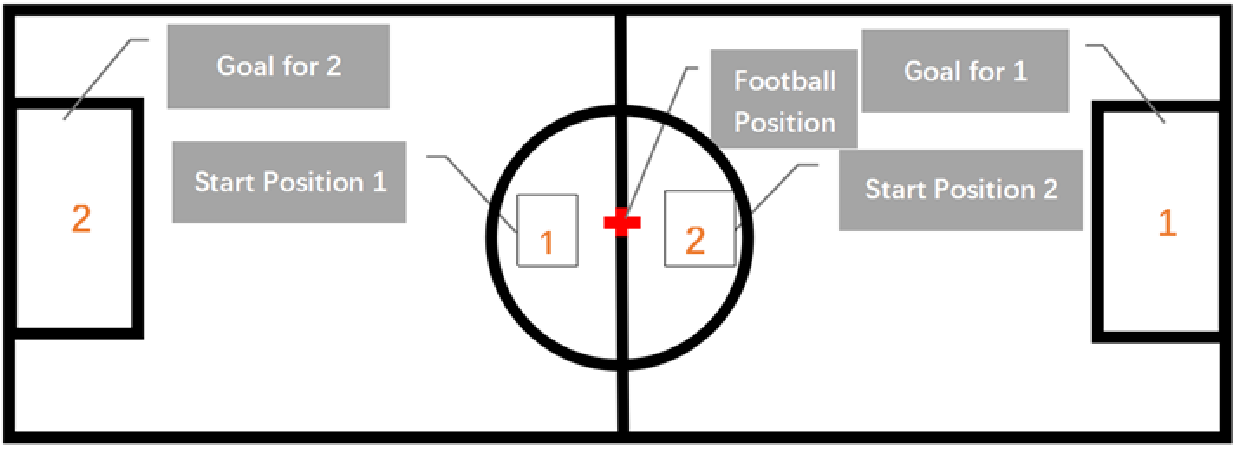
Exercise 1:

Car program


on start
radio set group 1
radio set group 7
on radio received receivedString
if receivedString = "Up" then
Motor M1 speed 100 M2 speed 100
Motor M3 speed 100 M4 speed 100
else if receivedString = "Down" then
Motor M1 speed -100 M2 speed -100
Motor M3 speed -100 M4 speed -100
else if receivedString = "Left" then
Motor M1 speed -85 M2 speed 100
Motor M3 speed 100 M4 speed 100
else if receivedString = "Right" then
Motor M1 speed 100 M2 speed -85
Motor M3 speed 100 M4 speed 85
else if receivedString = "B1" then
Motor M1 speed 150 M2 speed 150
Motor M3 speed 150 M4 speed 150
High speed forward
else if receivedString = "B2" then
Motor M1 speed 80 M2 speed 80
Motor M3 speed 80 M4 speed 80
Low speed forward
else if receivedString = "B3" then
Motor M1 speed -150 M2 speed 150
Motor M3 speed 150 M4 speed -150
Straight Left
else if receivedString = "B4" then
Motor M1 speed 150 M2 speed -150
Motor M3 speed -150 M4 speed 150
Straight Right
else if receivedString = "Stop" then
Motor Stop All
Stop

Remote program

Appendix



1.4.9 Lesson 9

Introduction

Learning Target

Liftable clip

To install the lift bracket

To attach the clip to the lift bracket

Exercise 1

Exercise 2

- The length of the clip will increase after adding the bracket, sufficient distance is needed when arranging the site.
- Pay attention to whether it is necessary to pause when operating every tool.
- Create a variable "destination" to record the destination, and use the destination to determine the action when the car goes out of bounds

Muti – function engineering car (Clips and Forks)

Exercise 3

- The forklift is in front of the car and the clip is behind the car
- Directly adjust the program in exercise 1
- Create variables to record the usage status of the two tools, and use this status to determine the next action of the tool

Answer

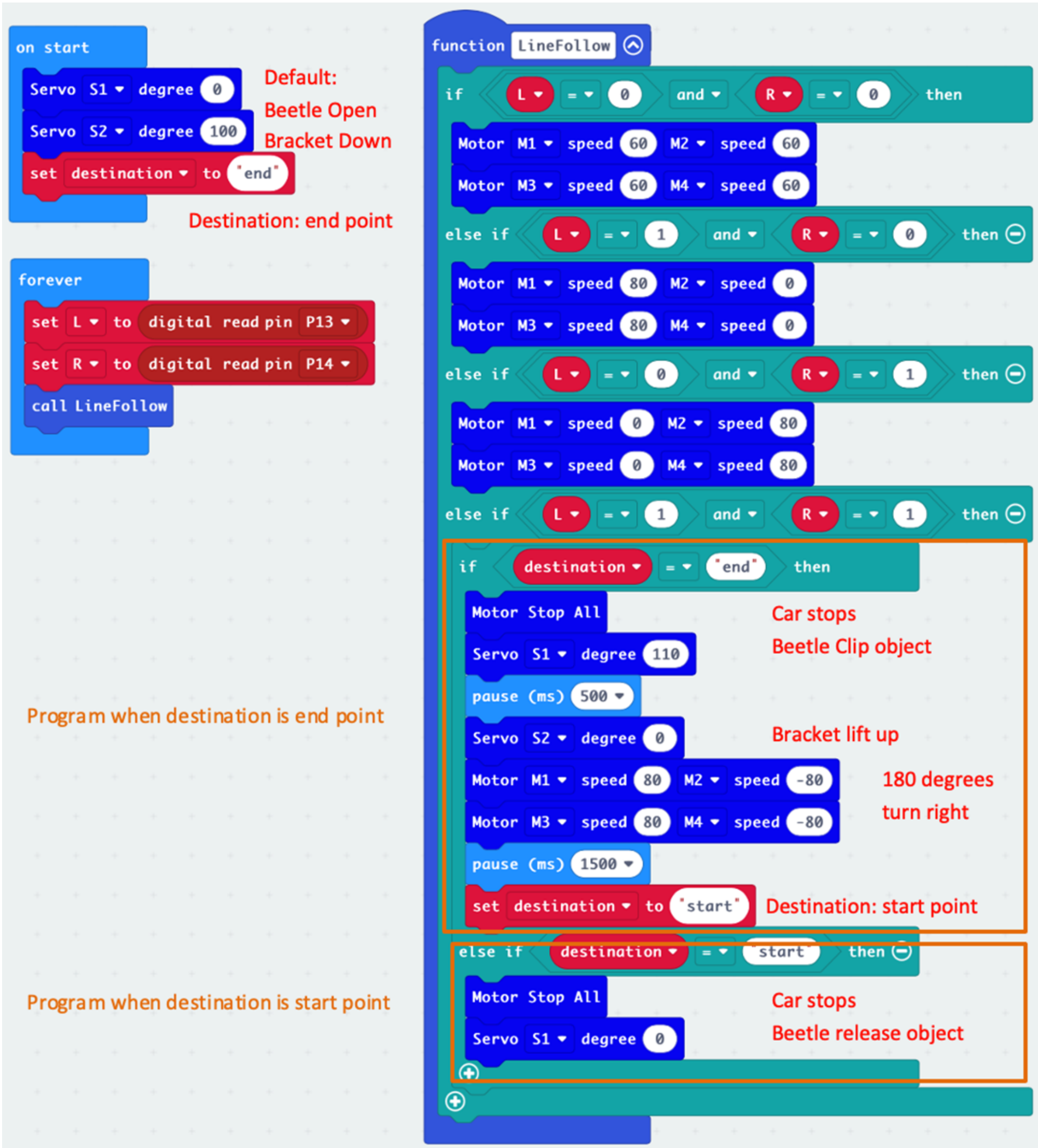
Exercise 1

Remote program

Car program

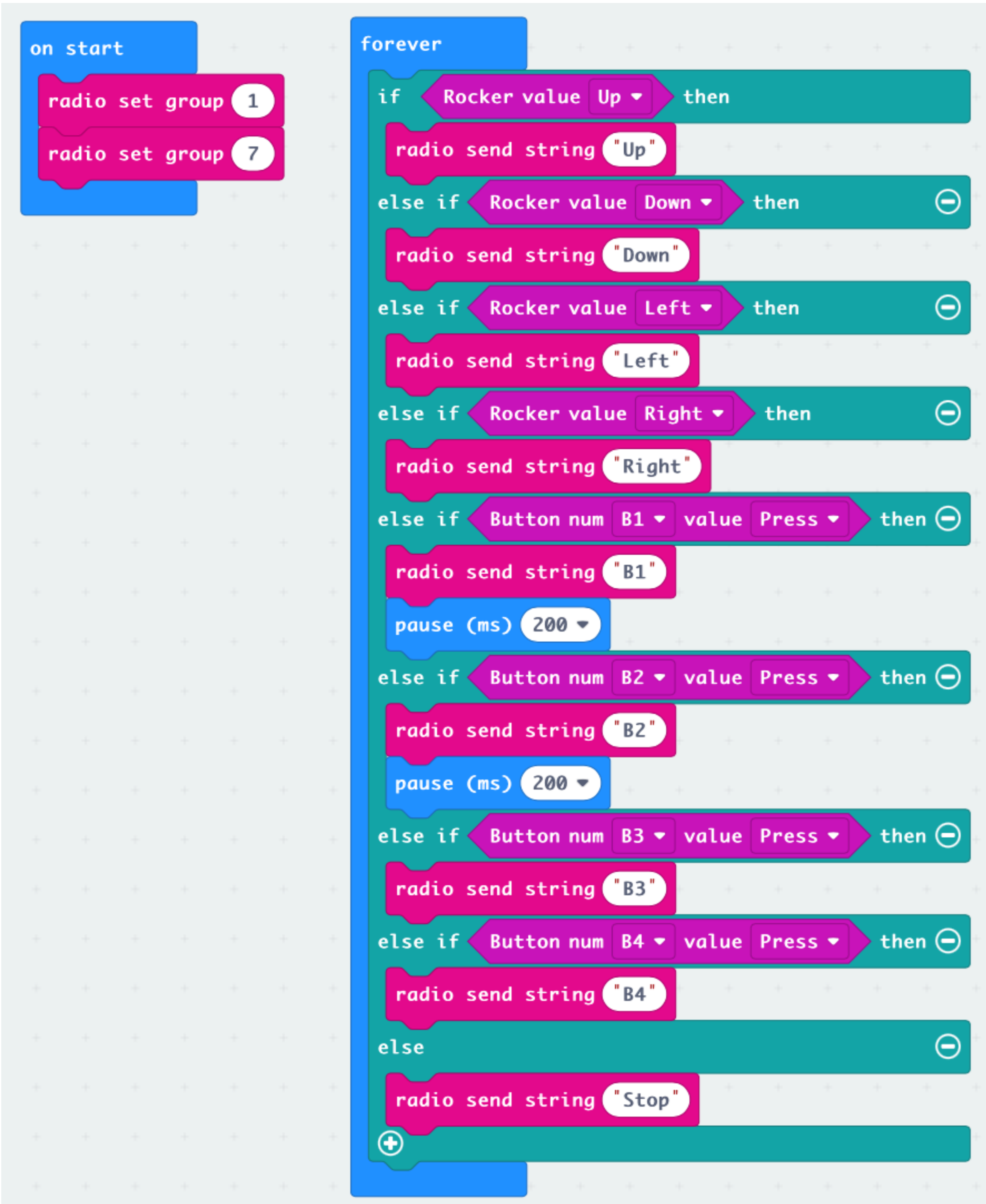

on start
radio set group 1
Servo S1 degree 110
Servo S2 degree 100
Default:
Beetle Close
Bracket down
on radio received receivedString
if receivedString = "Up" then
Motor M1 speed 255 M2 speed 255
Motor M3 speed 255 M4 speed 255
else if receivedString = "Down" then
Motor M1 speed -255 M2 speed -255
Motor M3 speed -255 M4 speed -255
else if receivedString = "Left" then
Motor M1 speed -85 M2 speed 255
Motor M3 speed 85 M4 speed 255
else if receivedString = "Right" then
Motor M1 speed 255 M2 speed -85
Motor M3 speed 255 M4 speed 85
else if receivedString = "B1" then
Servo S1 degree 0
else if receivedString = "B2" then
Servo S1 degree 110
else if receivedString = "B3" then
Servo S2 degree 0
else if receivedString = "B4" then
Servo S2 degree 100
else if receivedString = "Stop" then
Motor Stop All

Exercise 2


on start
Servo S1 degree 0
Servo S2 degree 100
set destination to "end"
Default:
Beetle Open
Bracket Down
Destination: end point
forever
set L to digital read pin P13
set R to digital read pin P14
call LineFollow
Program when destination is end point
Program when destination is start point
function LineFollow
if L = 0 and R = 0 then
Motor M1 speed 60 M2 speed 60
Motor M3 speed 60 M4 speed 60
else if L = 1 and R = 0 then
Motor M1 speed 80 M2 speed 0
Motor M3 speed 80 M4 speed 0
else if L = 0 and R = 1 then
Motor M1 speed 0 M2 speed 80
Motor M3 speed 0 M4 speed 80
else if L = 1 and R = 1 then
if destination = "end" then
Motor Stop All
Car stops
Servo S1 degree 110
Beetle Clip object
pause (ms) 500
Servo S2 degree 0
Bracket lift up
Motor M1 speed 80 M2 speed -80
Motor M3 speed 80 M4 speed -80
180 degrees
turn right
pause (ms) 1500
set destination to "start"
Destination: start point
else if destination = "start" then
Motor Stop All
Car stops
Servo S1 degree 0
Beetle release object

Exercise 3

Remote Program



Car program


on start
radio set group 1
Servo S1 degree 110
Servo S2 degree 100
set beetle_state to "close"
set forklift_state to "down"
Default:
Beetle close
Bracket down
Beetle state: close
Bracket state: down
function beetle_control
Beetle program
if beetle_state = "close" then
Servo S1 degree 0
Beetle open
set beetle_state to "open"
State: open
else if beetle_state = "close" then
Servo S1 degree 110
Beetle close
set beetle_state to "close"
State: close
function forklift_control
Bracket program
if forklift_state = "down" then
Servo S2 degree 0
Bracket lift up
set beetle_state to "up"
State: lift up
else if forklift_state = "up" then
Servo S1 degree 100
Bracket lift down
set beetle_state to "down"
State: lift down
on radio received receivedString
if receivedString = "Up" then
Motor M1 speed 255 M2 speed 255
Motor M3 speed 255 M4 speed 255
else if receivedString = "Down" then
Motor M1 speed -255 M2 speed -255
Motor M3 speed -255 M4 speed -255
else if receivedString = "Left" then
Motor M1 speed -85 M2 speed 255
Motor M3 speed 85 M4 speed 255
else if receivedString = "Right" then
Motor M1 speed 255 M2 speed -85
Motor M3 speed 255 M4 speed 85
else if receivedString = "B1" then
call beetle_control
else if receivedString = "B2" then
call forklift_control
else if receivedString = "B3" then
Motor M1 speed 80 M2 speed -80
Motor M3 speed 80 M4 speed -80
Turn Right
else if receivedString = "B4" then
Motor M1 speed -80 M2 speed 80
Motor M3 speed -80 M4 speed 80
Turn Left
else if receivedString = "Stop" then
Motor Stop All

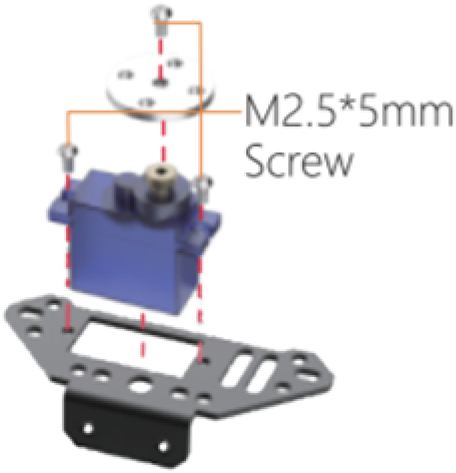
1.4.10 Lesson 10

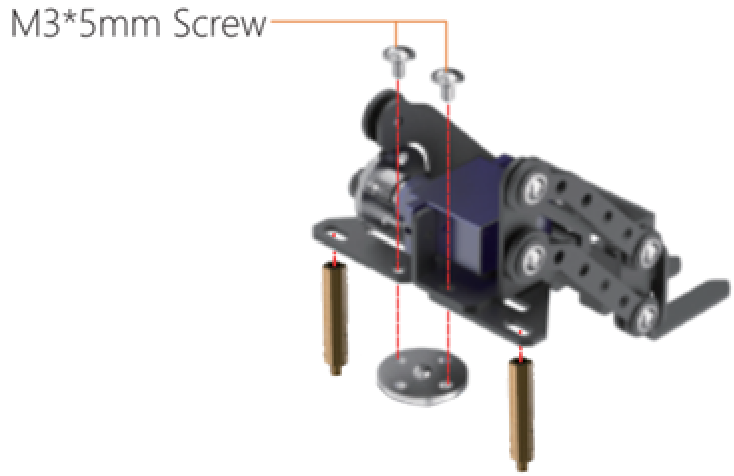
Introduction

Learning Target

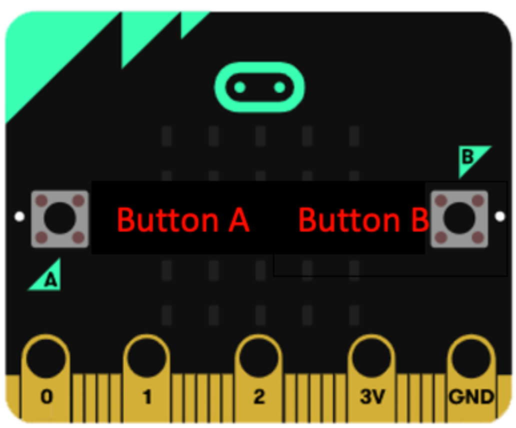
Two-way movable fork

Installation steps:





Exercise 1



Task

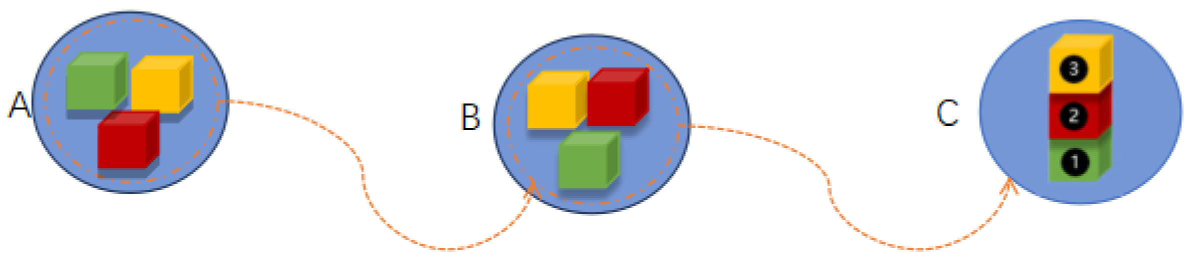
Control the car to transport the objects in the left to the middle, then transport the objects in the right to the left, and finally transport the objects in the middle to the right.

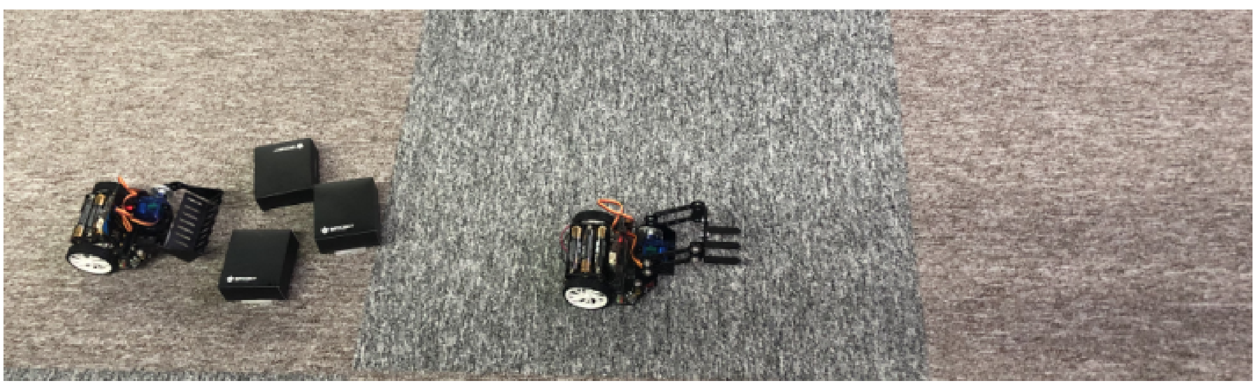
Car relay race

Introduce

Game rules

```
The relay race is divided into two stages, Stage 1: AB (task: transport all three␣
↪objects from point A to point B), stage 2: BC (task: transport three objects from␣
↪point B to point C, and Overlap these objects in the order of 1, 2, 3). After␣
↪completing the second stage, only two cars reach point C (point A is the starting␣
↪point, point B is the relay point, and point C is the end point) to be considered␣
↪successful.
During the game, players cannot touch the car.
Whoever completes all tasks in the shortest time wins.
```

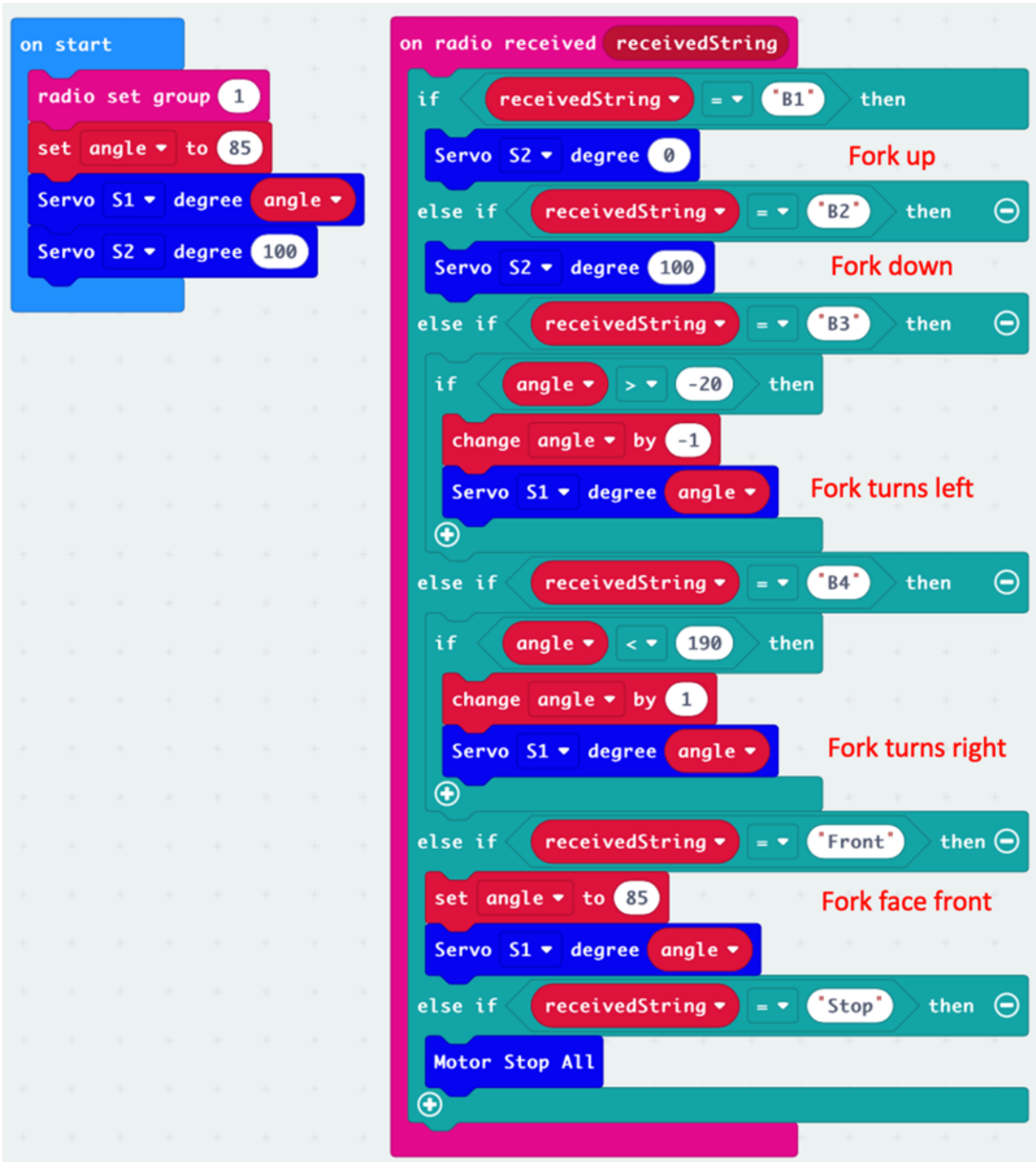




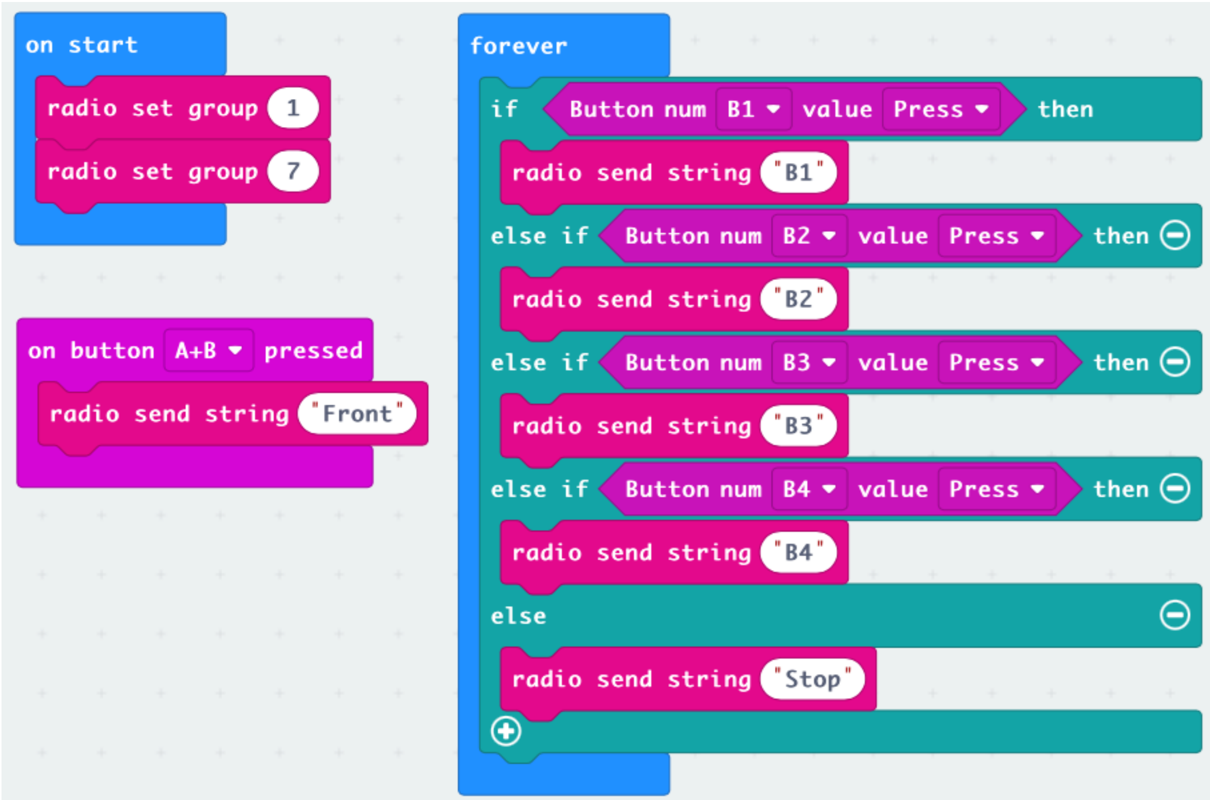
Answer

Exercise 1

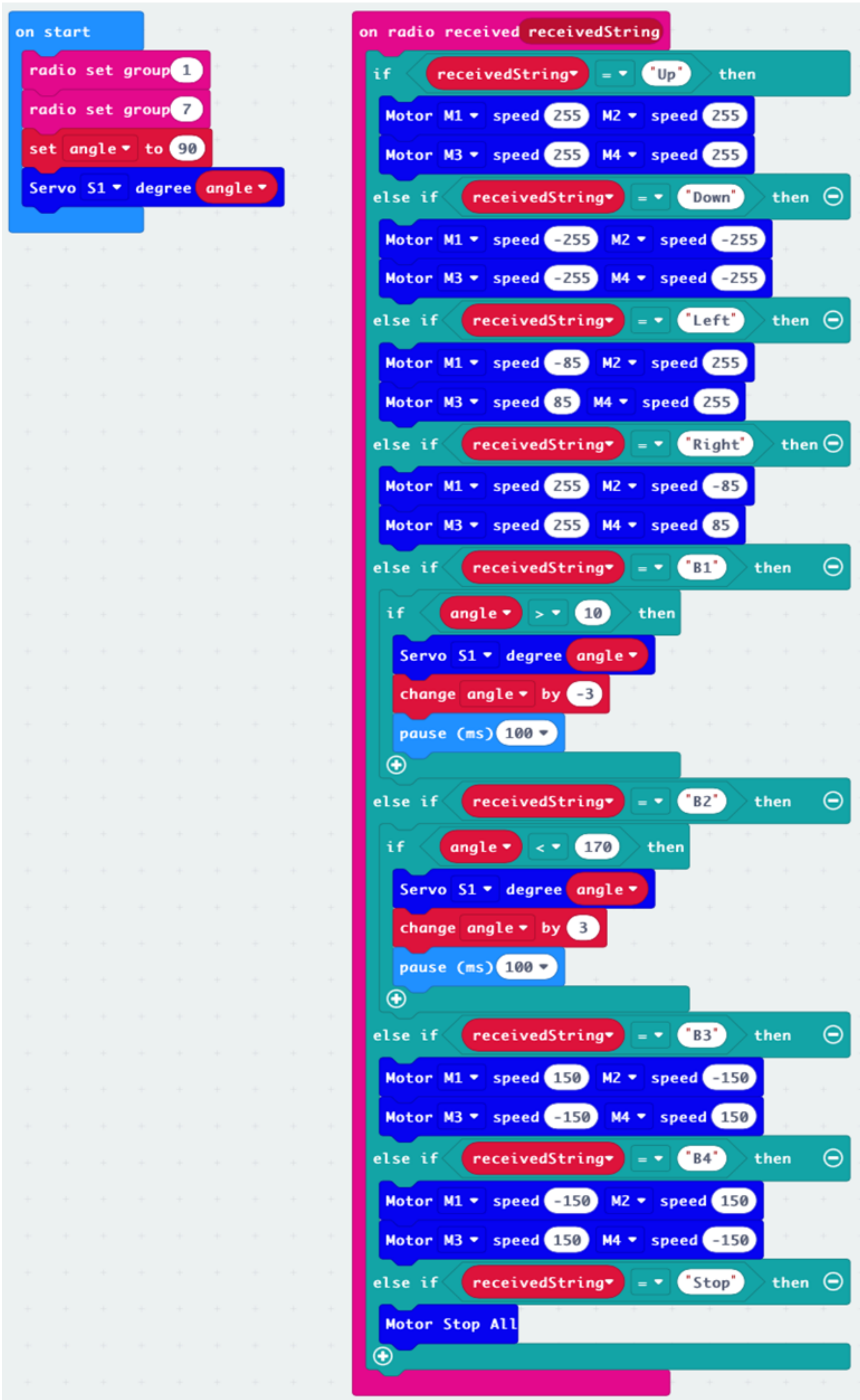
Car program (the horizontal moving part of the fork)

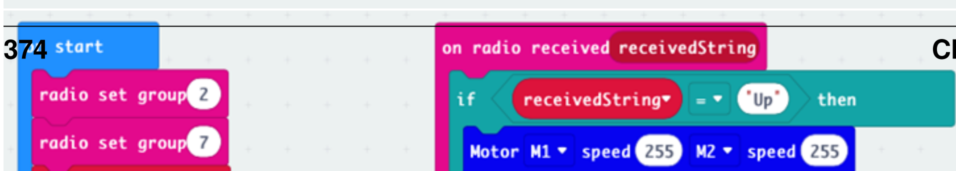


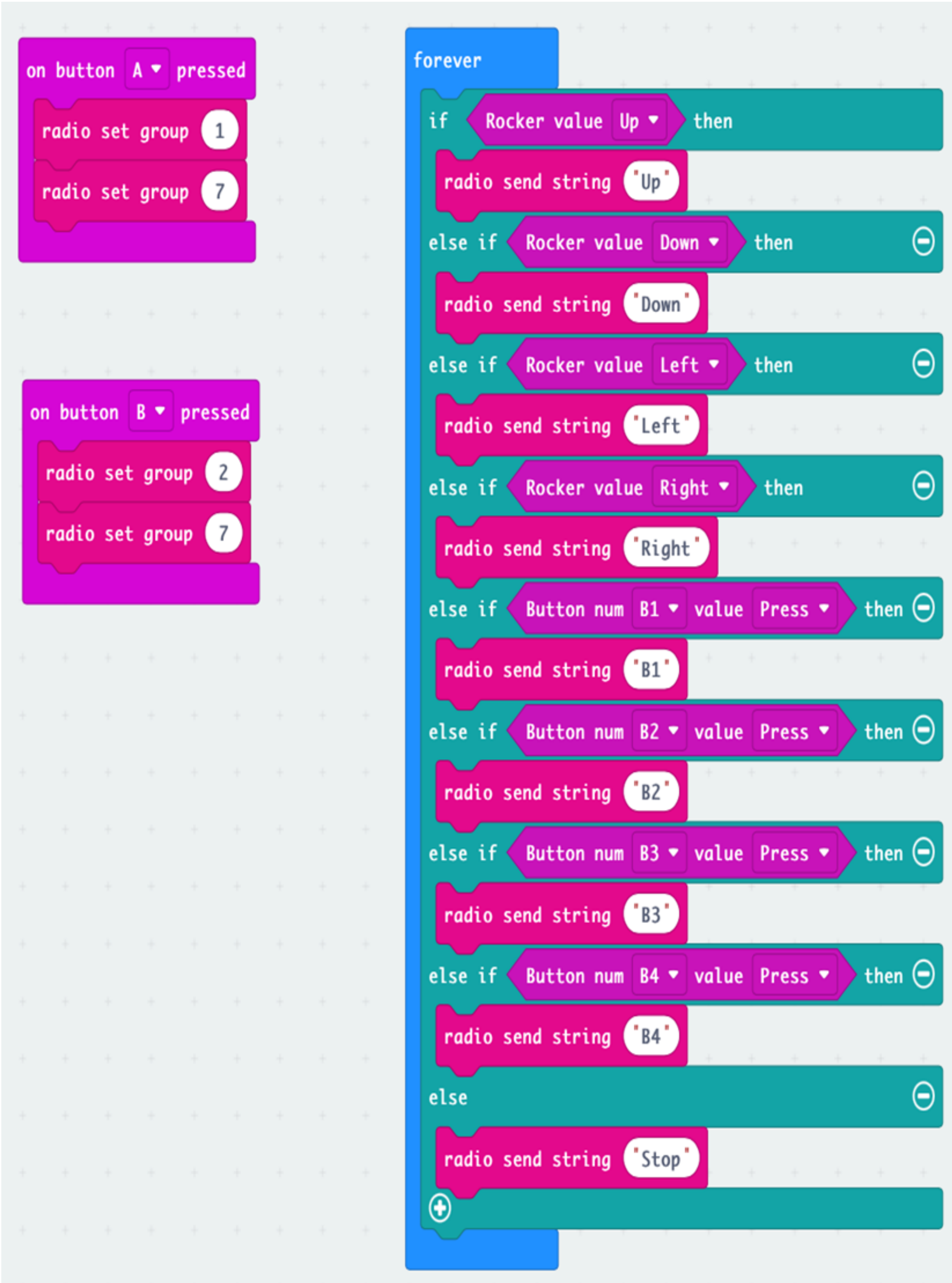
Remote Program



Car relay race






on button A pressed
radio set group 1
radio set group 7
on button B pressed
radio set group 2
radio set group 7
forever
if Rocker value Up then
radio send string "Up"
else if Rocker value Down then
radio send string "Down"
else if Rocker value Left then
radio send string "Left"
else if Rocker value Right then
radio send string "Right"
else if Button num B1 value Press then
radio send string "B1"
else if Button num B2 value Press then
radio send string "B2"
else if Button num B3 value Press then
radio send string "B3"
else if Button num B4 value Press then
radio send string "B4"
else
radio send string "Stop"

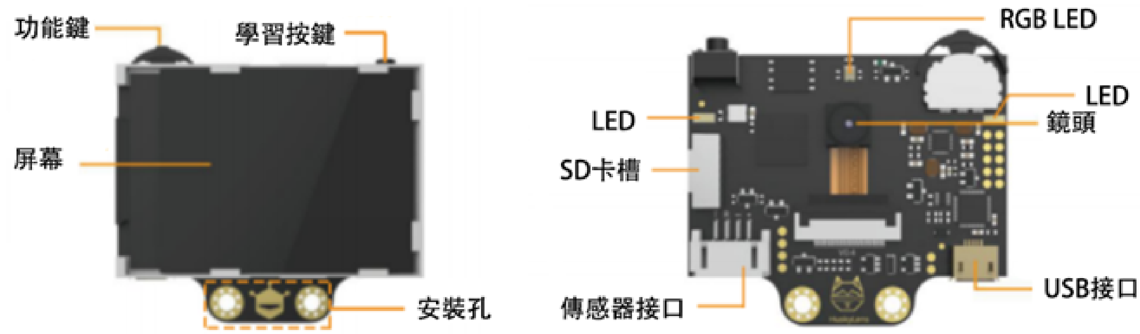
1.5 Micro:bit M1 SMART CAR_Advanced

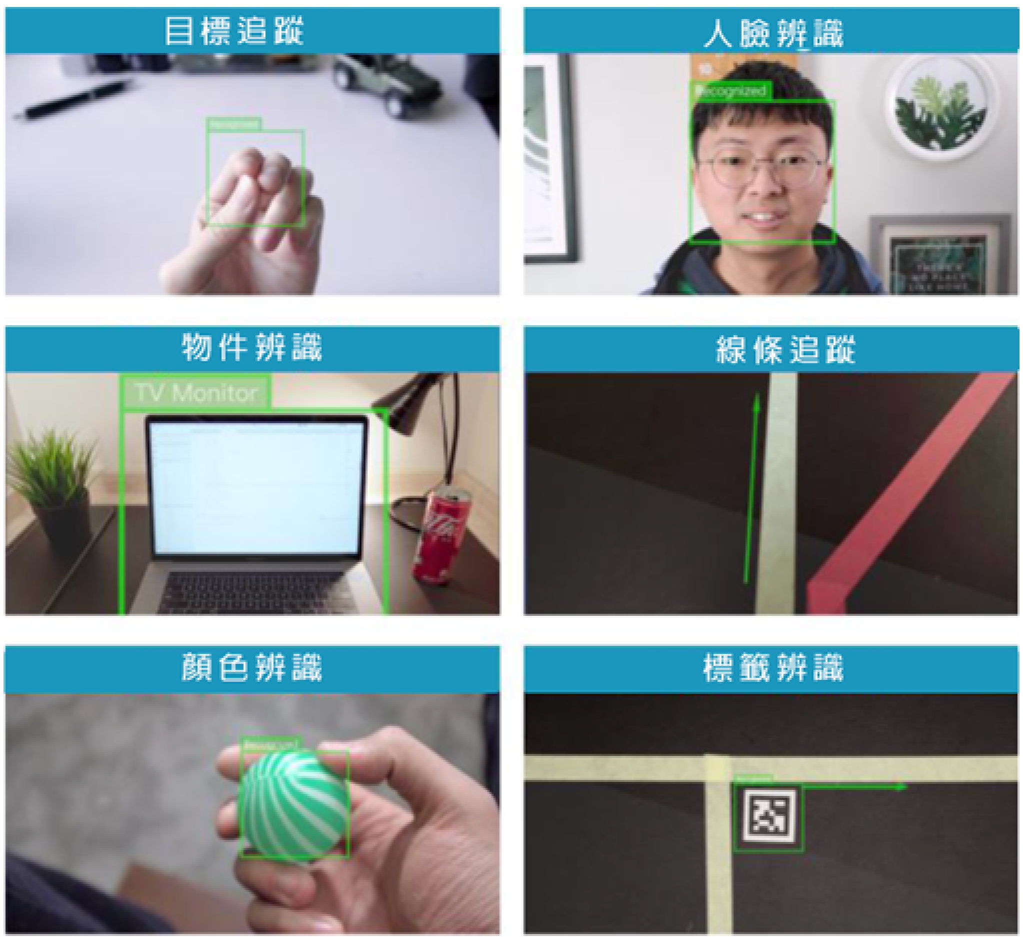
1.5.1 Lesson 1

Introduction

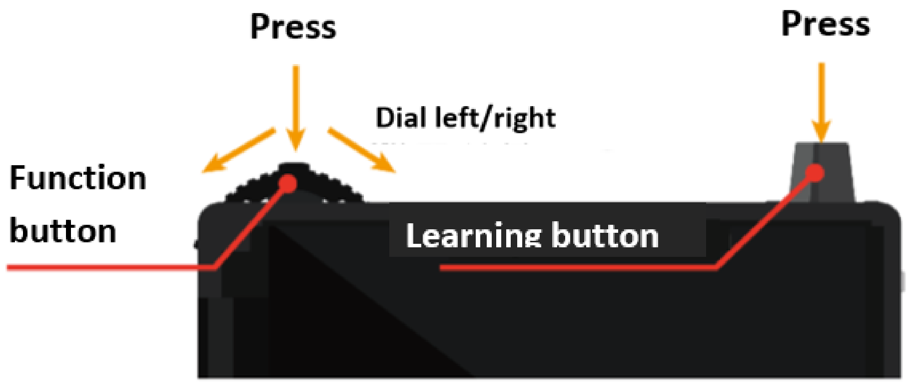
Objective

Introduction of HuskyLens



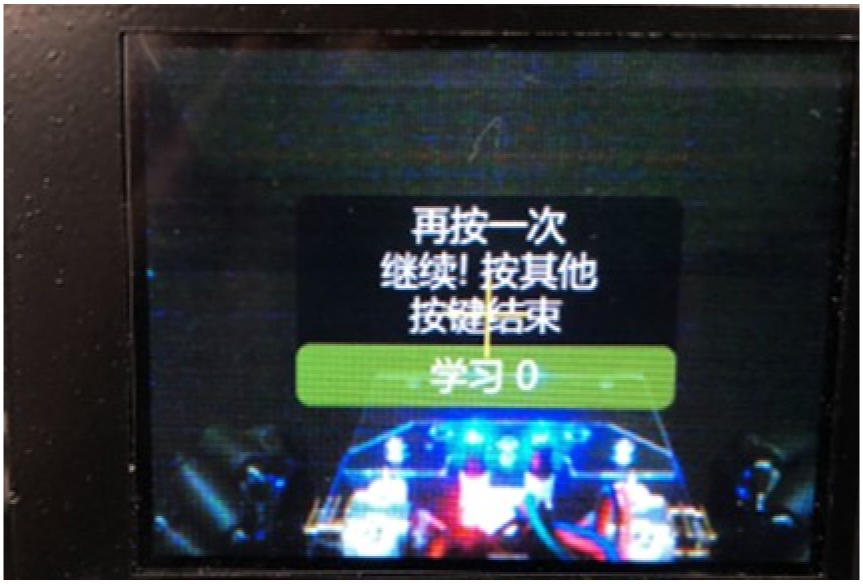


Method



- Dialing the function button to the left/right to switch functions/modes
- Click on the function button to call out the function menu
- After long pressing the function button, you can set whether to learn multiple targets.
- Click the function button again to drag the scroll bar to the right to enable “Learn More”.

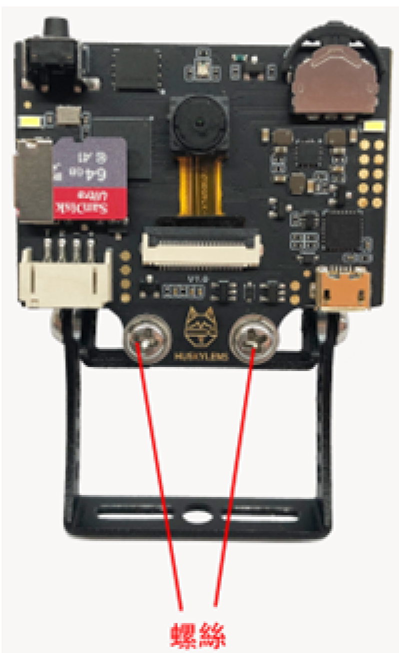
- Click the Learn button to learn a new object, and the object ID will be displayed on the screen.
- Press and hold the learning button to learn to recognize new object from different angle, and distance.
- After studying the new object, it is required to press the learning button again for further study within 4s.Otherwise, stop studying after 4 seconds, you must first forget the target and then re-study

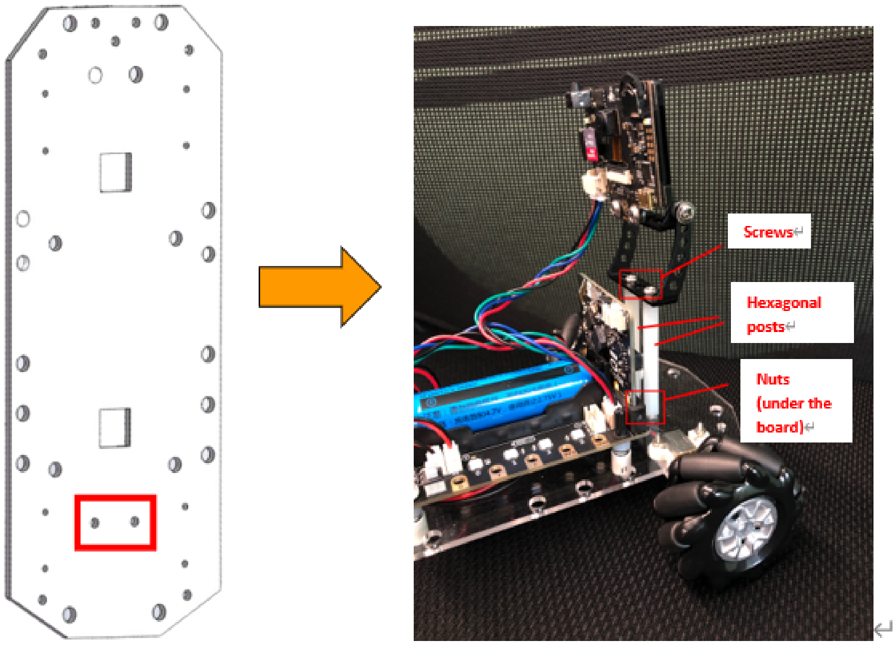


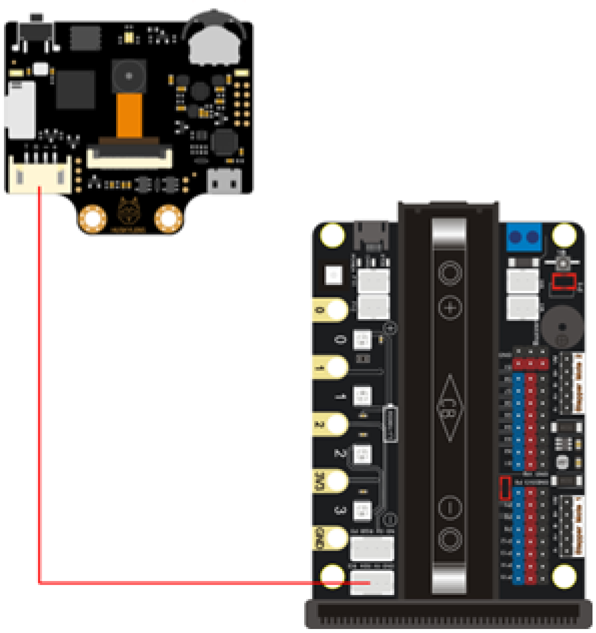
- After terminating the study, press learning button twice for forgetting the learned objects.

Installing HuskyLens in Smart Car

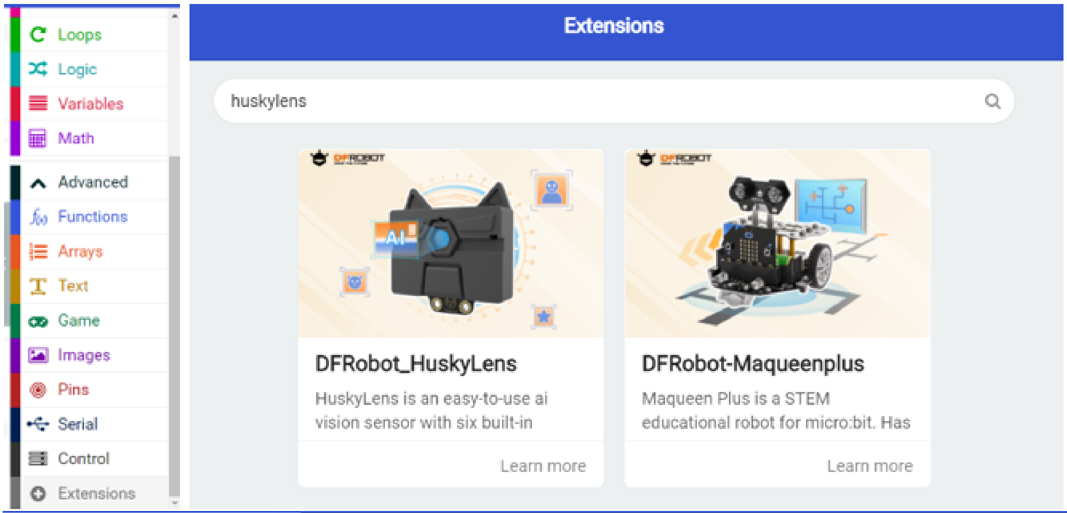

螺絲

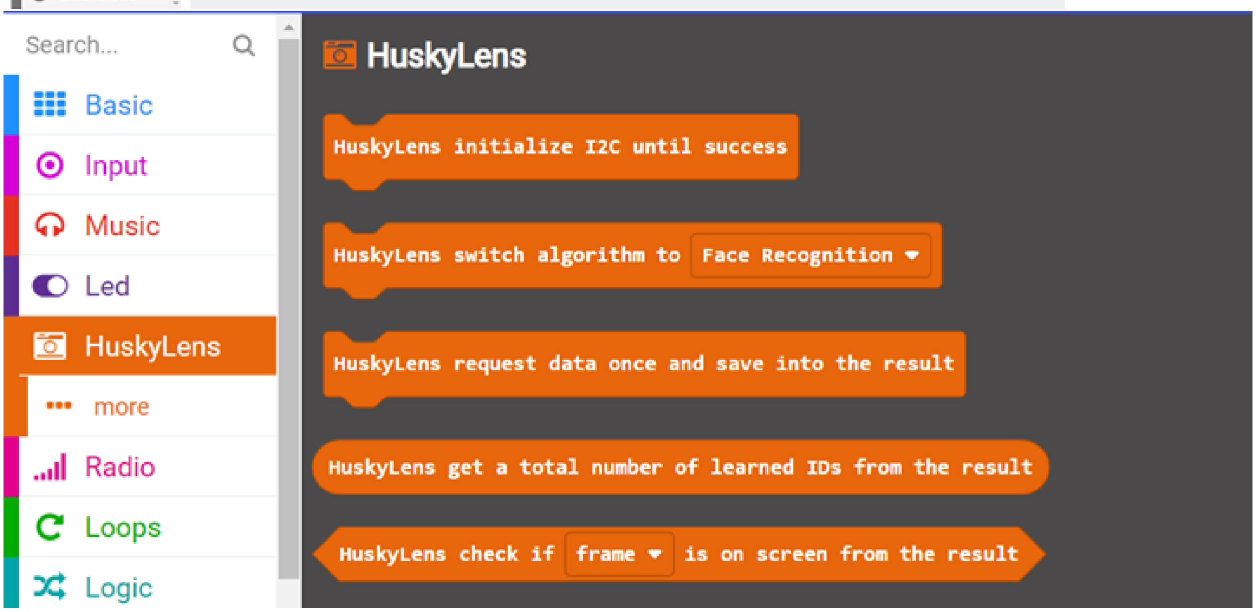

螺絲


Screws
Hexagonal posts
Nuts (under the board)



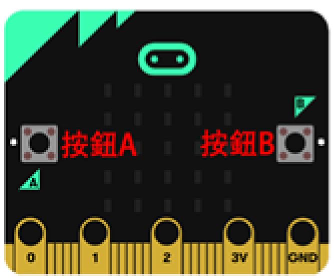

HuskyLens expansion pack





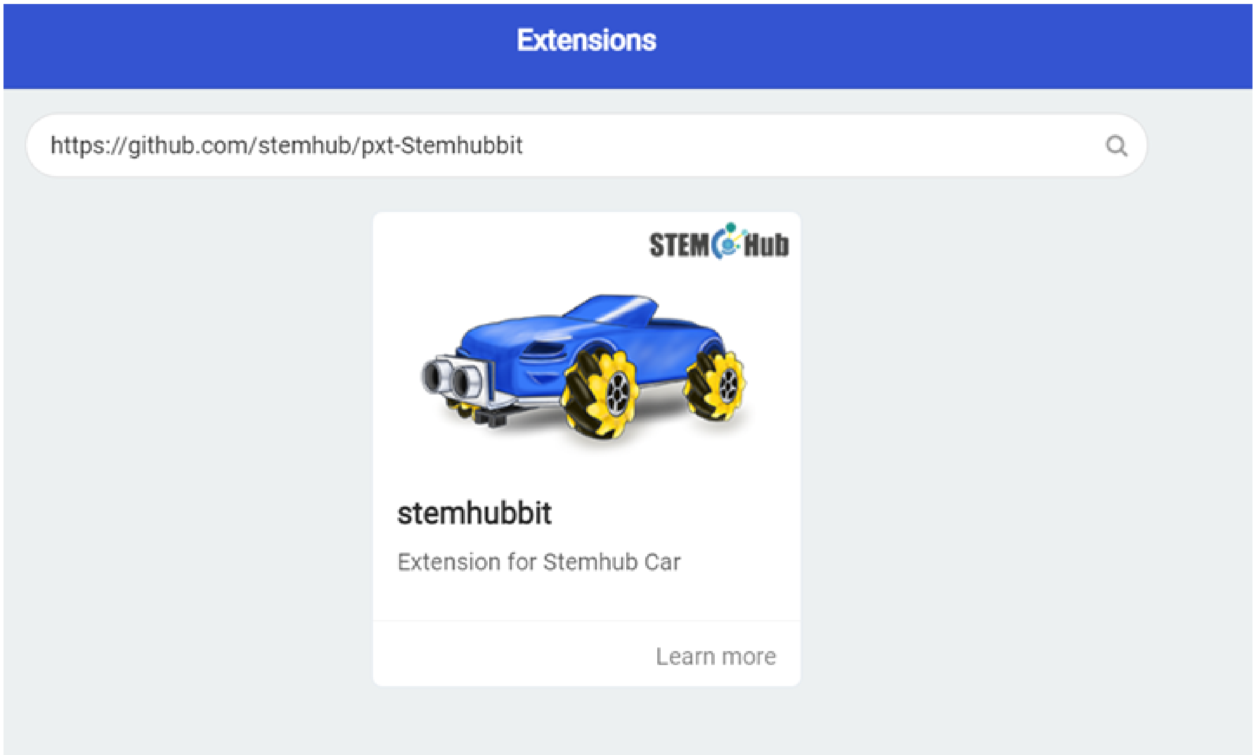
HuskyLens building block module

Exercise 1




```
1When the program starts, it automatically switches to the object classification mode.
2Use HuskyLens to learn 5 kinds of objects (ID)
3Press the button A: display the number of learning objects - (a)
4Press the button B: clear learning object data
5Press the button A again to display the number of learning objects - (b)
```

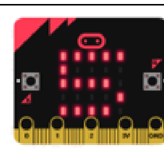
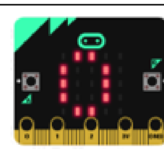
Exercise 2:

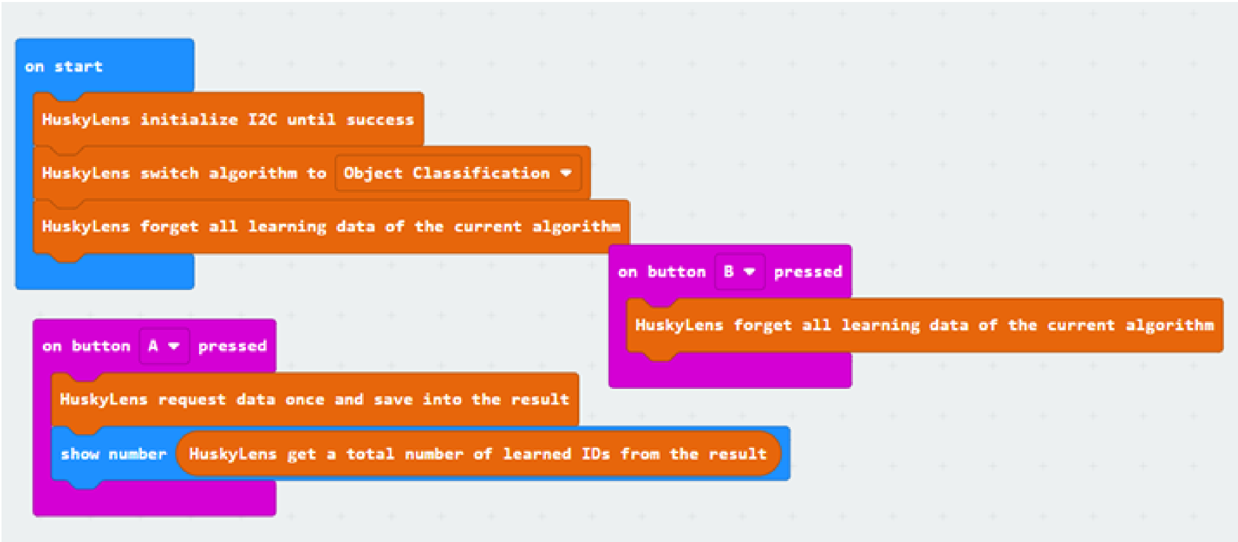



```
1When the program starts, it automatically switches to the object classification mode.
2Use HuskyLens to learn multiple objects.
3If there is a learned object within the screen, the Micro:bit board displays the object ID closest to the center of the screen and lights all onboard LEDs green, otherwise lights up red.
```

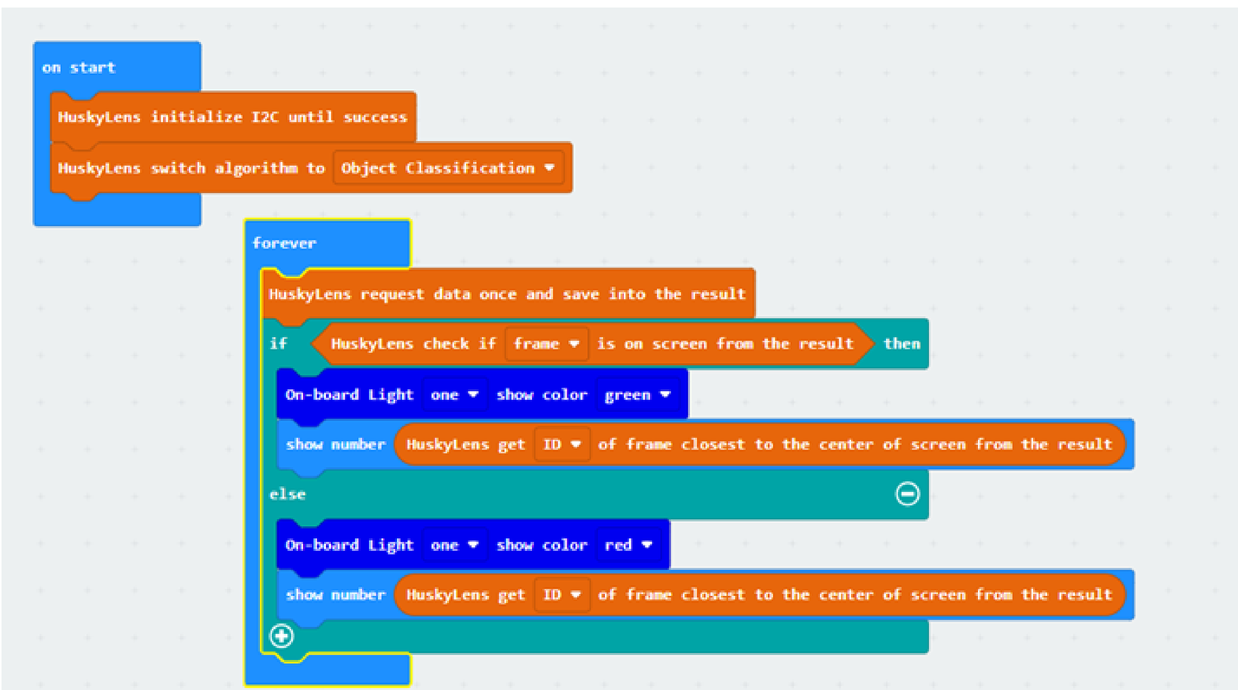
Answer

Exercise 1

Location↵	Number↵
(a)↵	↵
(b)↵	↵


on start
HuskyLens initialize I2C until success
HuskyLens switch algorithm to Object Classification
HuskyLens forget all learning data of the current algorithm
on button B pressed
HuskyLens forget all learning data of the current algorithm
on button A pressed
HuskyLens request data once and save into the result
show number HuskyLens get a total number of learned IDs from the result

Exercise 2


on start
HuskyLens initialize I2C until success
HuskyLens switch algorithm to Object Classification
forever
HuskyLens request data once and save into the result
if HuskyLens check if frame is on screen from the result then
On-board Light one show color green
show number HuskyLens get ID of frame closest to the center of screen from the result
else
On-board Light one show color red
show number HuskyLens get ID of frame closest to the center of screen from the result

1.5.2 Lesson 2

Introduction

Objective

HuskyLens facial recognition function

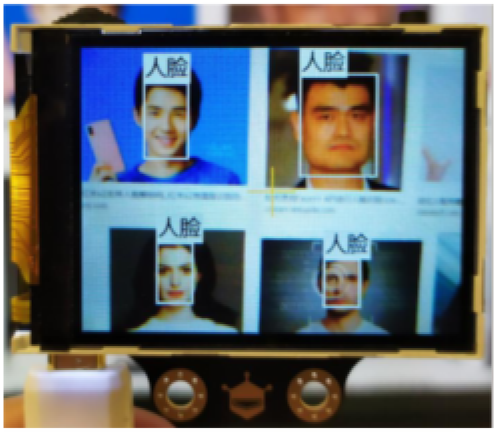
What is facial recognition?

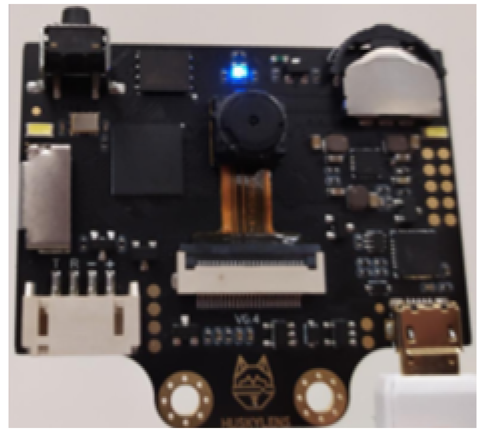
Applications of facial recognition

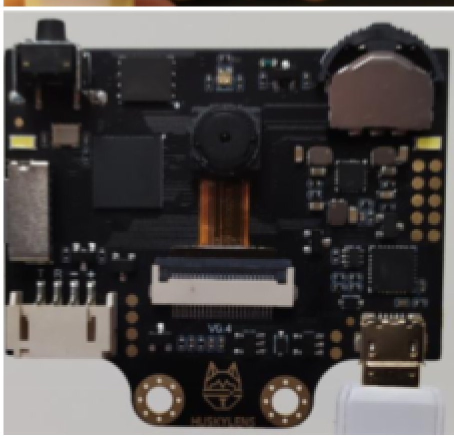
- Access right system: assess the right for entering protected areas such as prisons, detention centers, communities, schools, and residences by using facial recognition.
- Surveillance systemIt can be used to monitor crowds in public places, such as banks, airports, stadiums, shopping malls and supermarkets.
- Internet application: use face recognition to assist online payment, prevent others from stealing credit cards and protect society.

facial recognition with Husky lens

Face Detection:

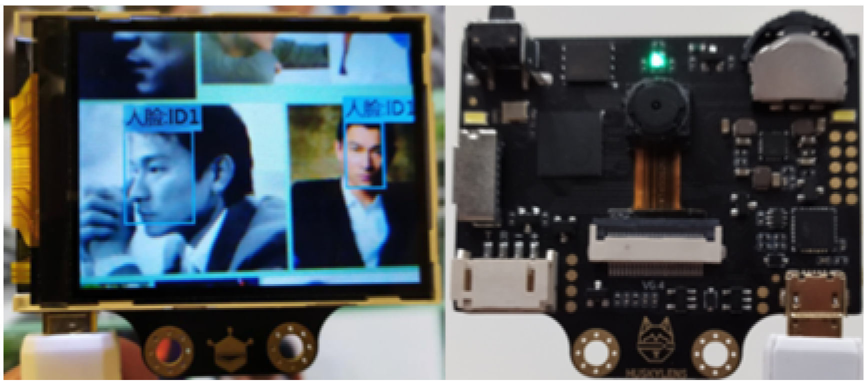






face learning:

facial recognition:



Exercise 1

- In HuskyLens >> More. There are blocks that change name by ID

- If you don't have an idea for giving a name, try the following names: Peter, Sam, Mary
- Since it takes time for HuskyLens to change objects, changing the names of multiple objects at one time may fail
- Therefore use "forever"

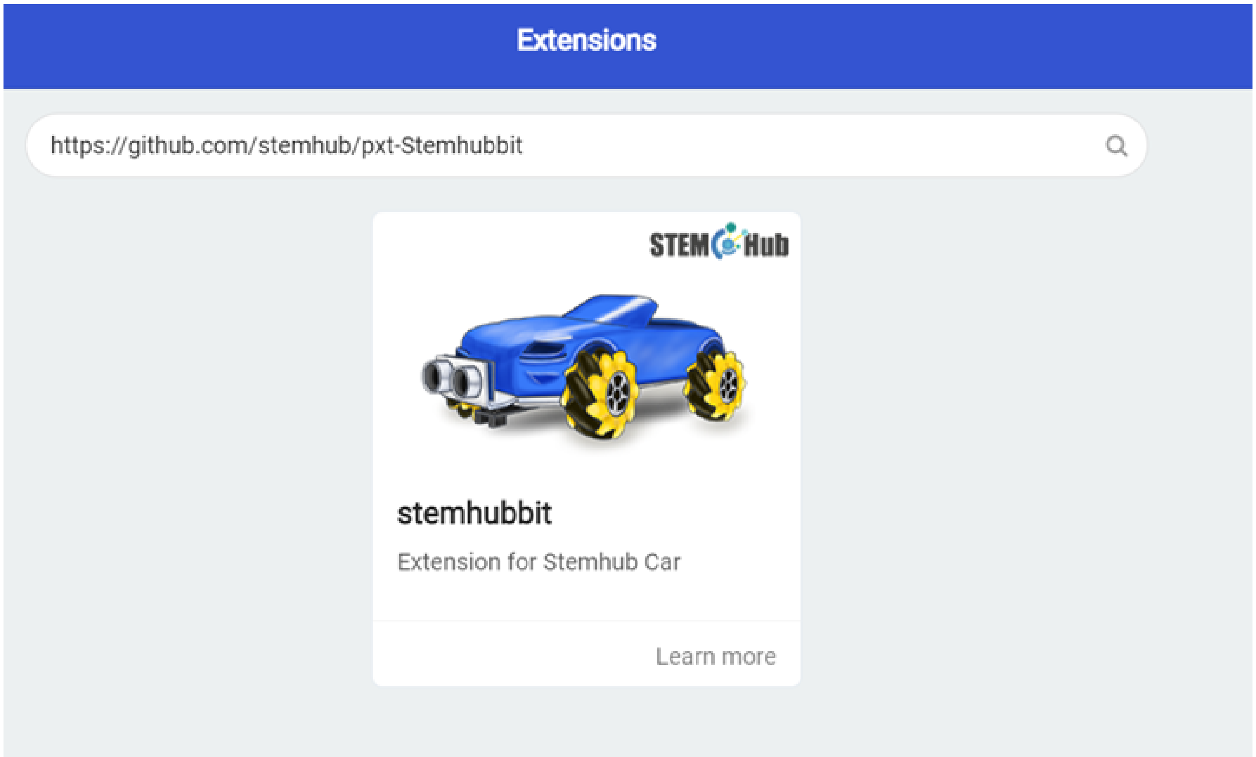
Restore changed object names

Exercise 2

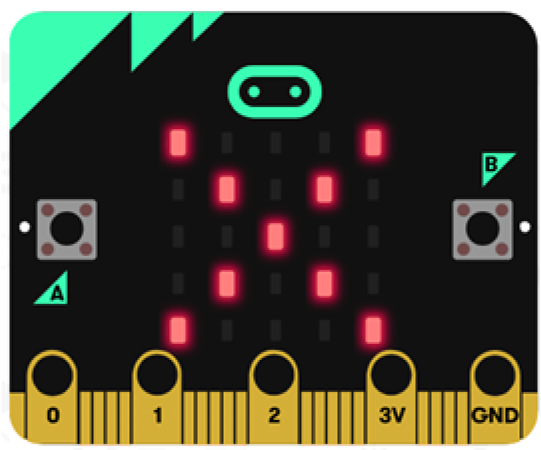
- Create variable "ID" to store each face ID
- Use Loop >> Repeat Judgment… Execute Blocks to change face names one by one

Exercise 3

- Assume HuskyLens sees only one face at a time
- Since the angle of the lens cannot be adjusted in real time, it is better to place different photos in front of the car in turn, or hold the car and place the lens on the face.
- The blocks for controlling the car's motor are in the Stemhub:bit expansion pack:https://github.com/stemhub/pxt-Stemhubbit

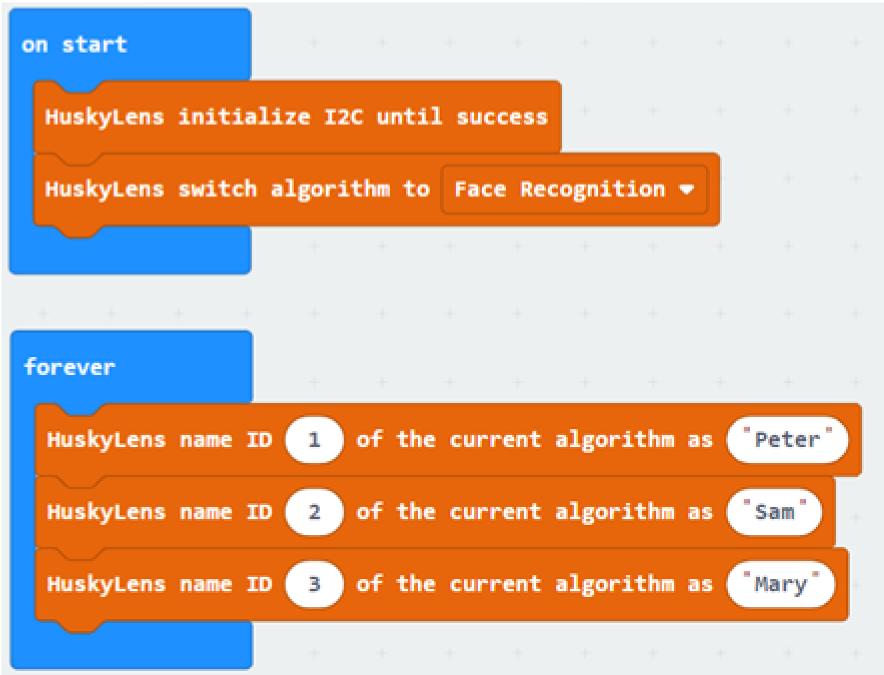

Extensions
https://github.com/stemhub/pxt-Stemhubbit
STEM Hub
stemhubbit
Extension for Stemhub Car
Learn more

Exercise 4

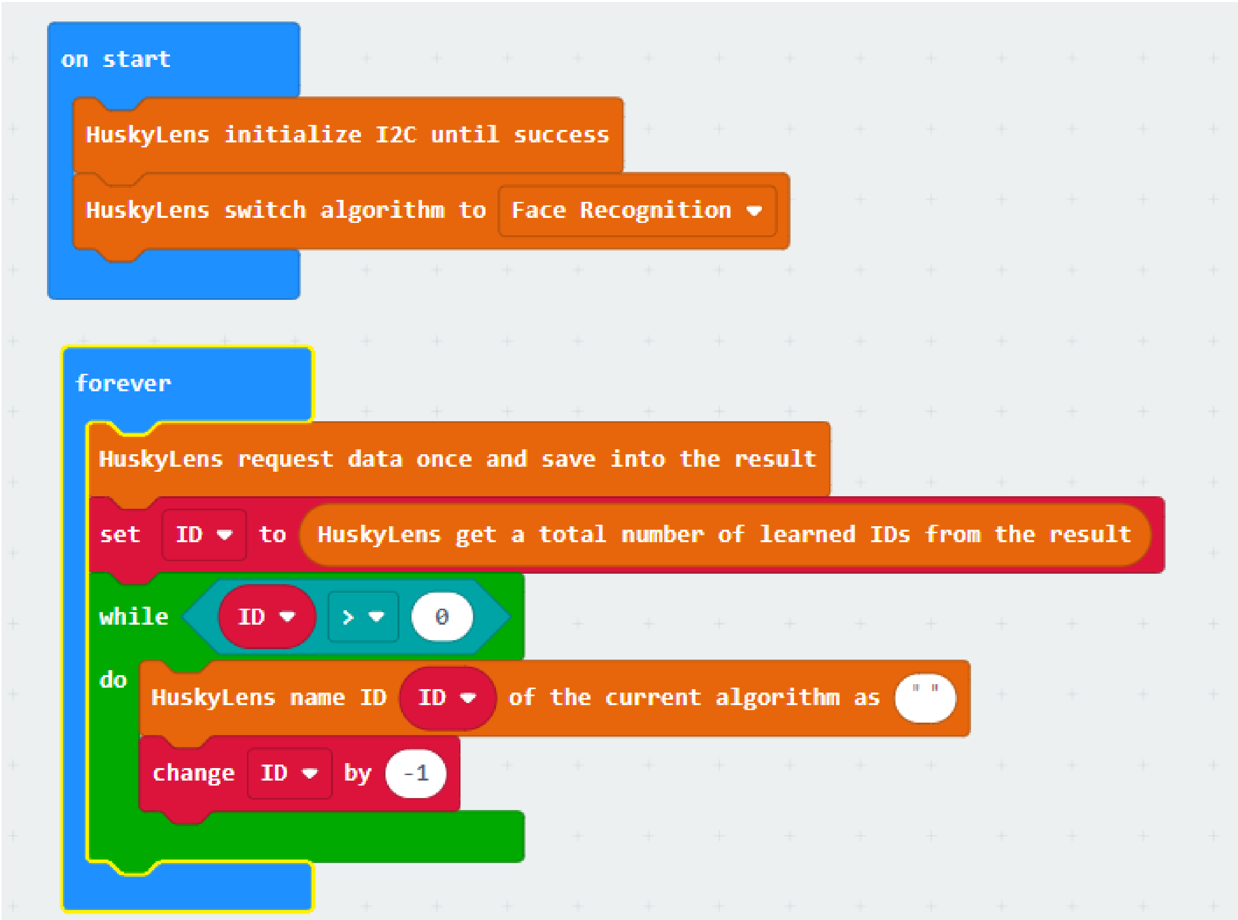

B
A
0
1
2
3V
GND

Answer

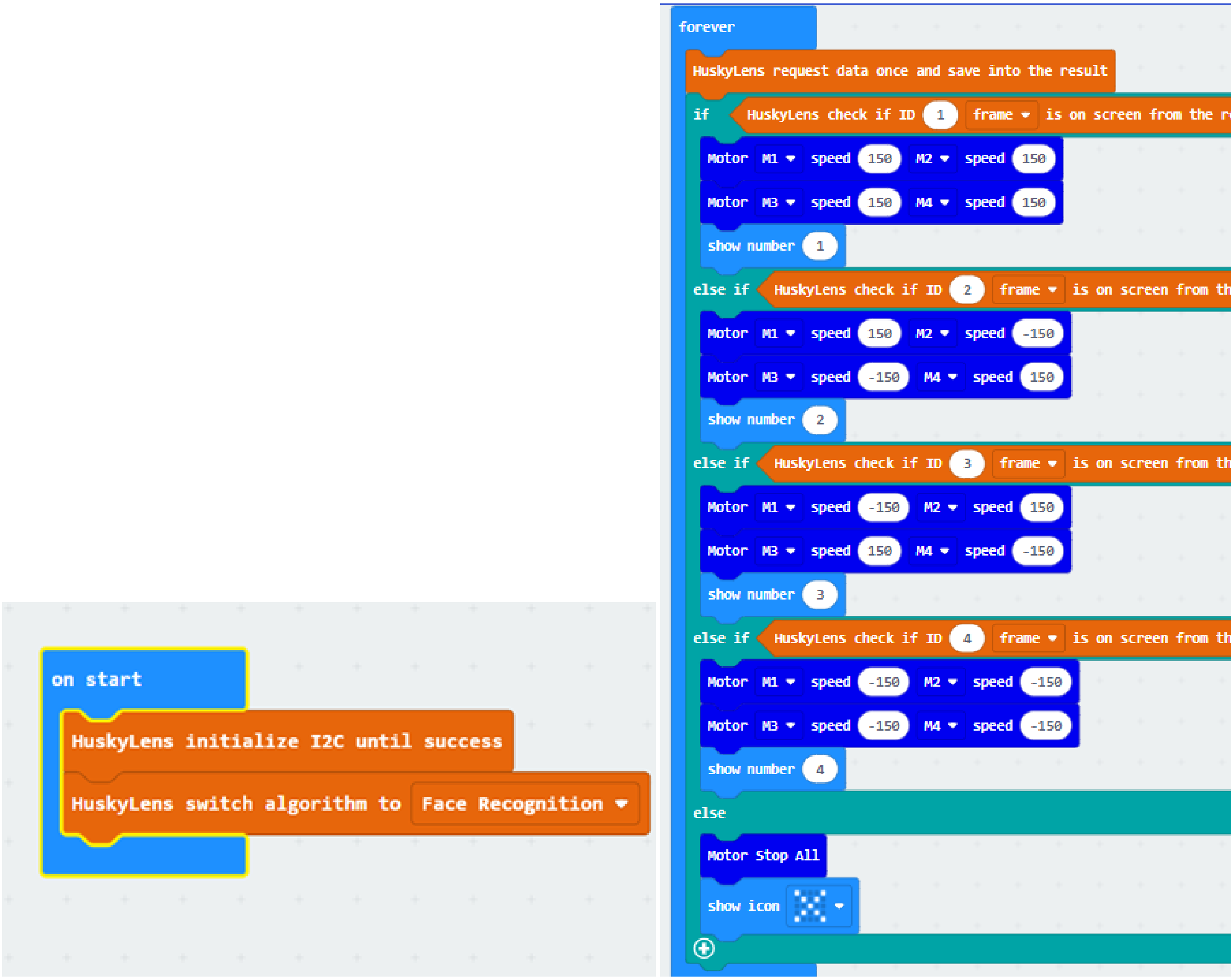
Exercise 1



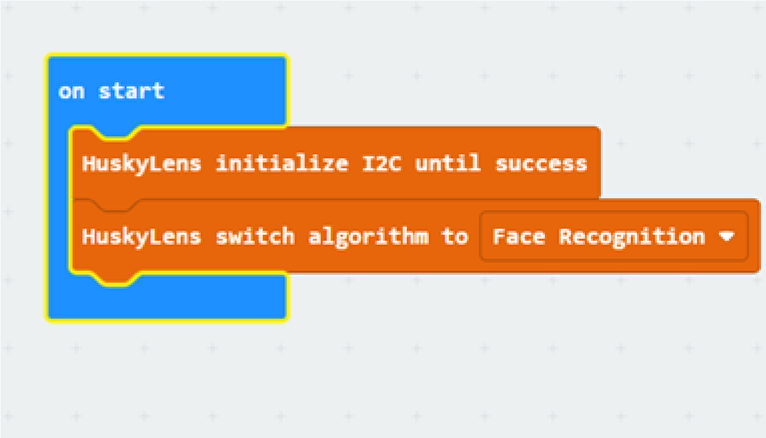
Exercise 2

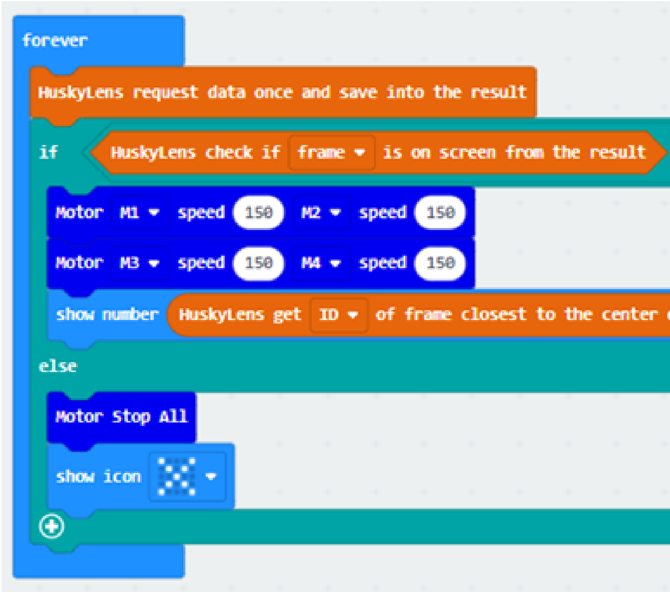

on start
HuskyLens initialize I2C until success
HuskyLens switch algorithm to Face Recognition
forever
HuskyLens request data once and save into the result
set ID to HuskyLens get a total number of learned IDs from the result
while ID > 0
do
HuskyLens name ID ID of the current algorithm as " "
change ID by -1

Exercise 3


on start
HuskyLens initialize I2C until success
HuskyLens switch algorithm to Face Recognition
forever
HuskyLens request data once and save into the result
if HuskyLens check if ID 1 frame is on screen from the r
Motor M1 speed 150 M2 speed 150
Motor M3 speed 150 M4 speed 150
show number 1
else if HuskyLens check if ID 2 frame is on screen from th
Motor M1 speed 150 M2 speed -150
Motor M3 speed -150 M4 speed 150
show number 2
else if HuskyLens check if ID 3 frame is on screen from th
Motor M1 speed -150 M2 speed 150
Motor M3 speed 150 M4 speed -150
show number 3
else if HuskyLens check if ID 4 frame is on screen from th
Motor M1 speed -150 M2 speed -150
Motor M3 speed -150 M4 speed -150
show number 4
else
Motor Stop All
show icon

Exercise 4


on start
HuskyLens initialize I2C until success
HuskyLens switch algorithm to Face Recognition


forever
HuskyLens request data once and save into the result
if
HuskyLens check if frame is on screen from the result
Motor M1 speed 150 M2 speed 150
Motor M3 speed 150 M4 speed 150
show number HuskyLens get ID of frame closest to the center
else
Motor Stop All
show icon

1.5.3 Lesson 3

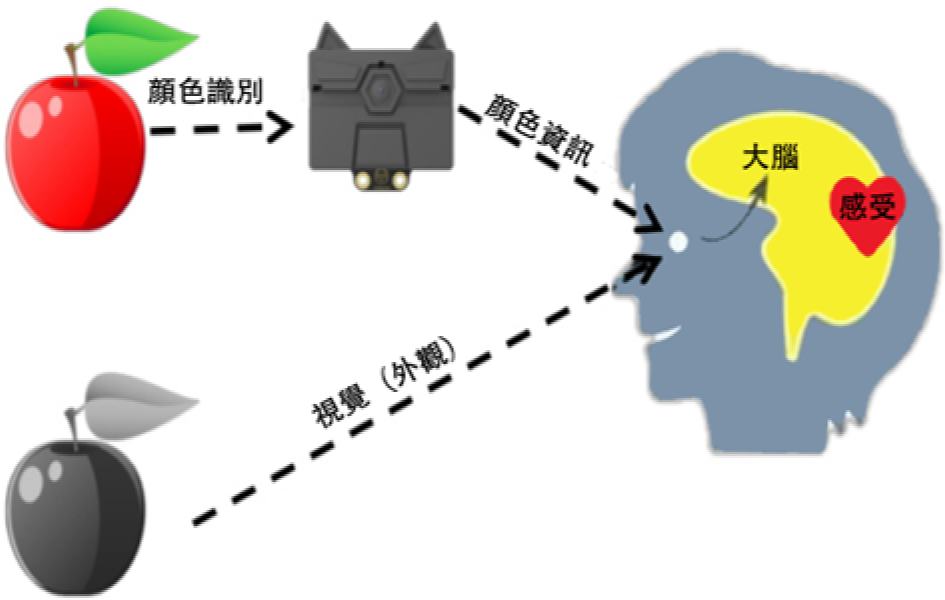
Introduction

Objective

HuskyLens Color Recognition Function

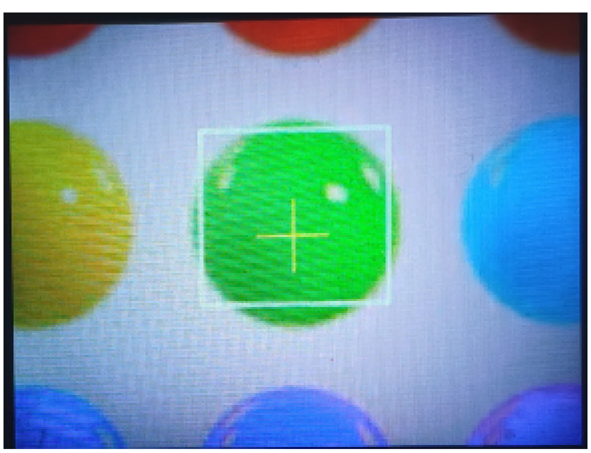
What is Color Recognition

Applications of Color Recognition

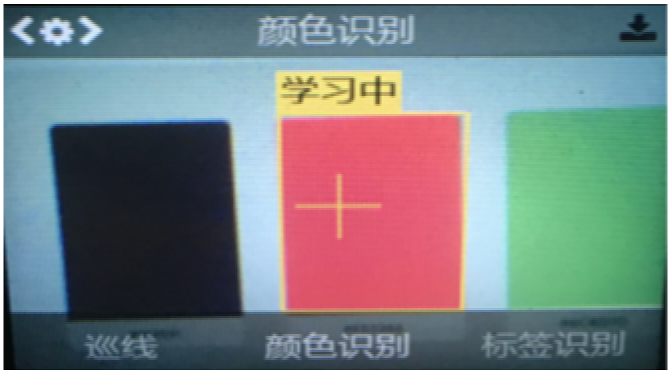


Color Recognition of Huskylens

Detecting color

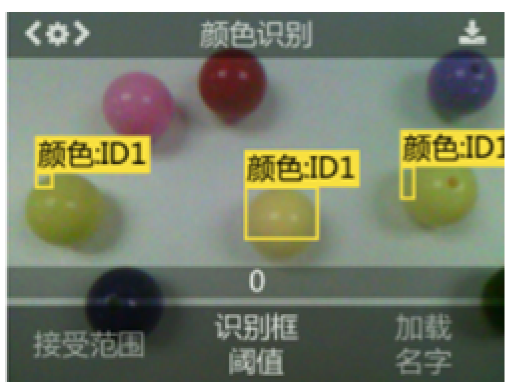


Learning color

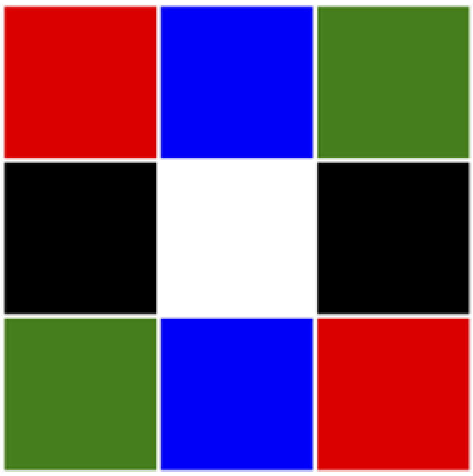


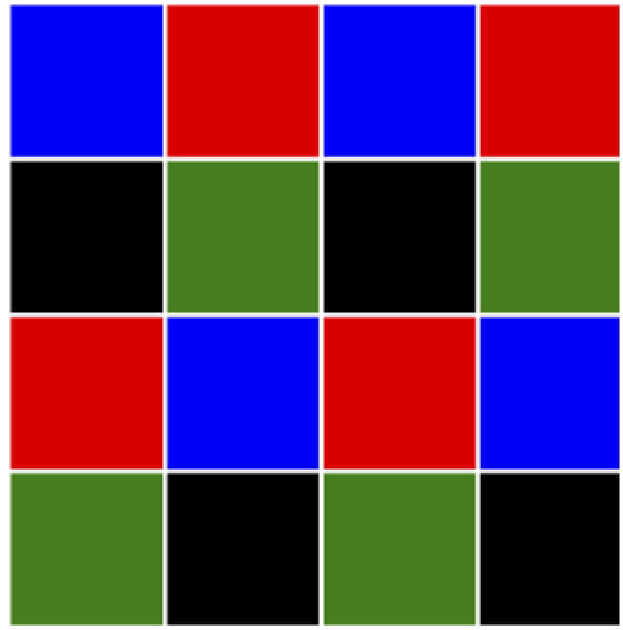
Identifying color

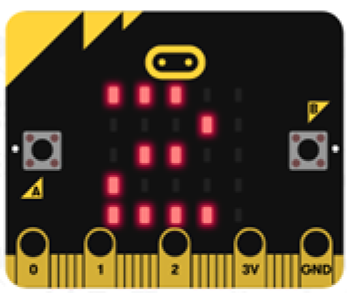
Adjusting the threshold of the recognition border



Exercise 1

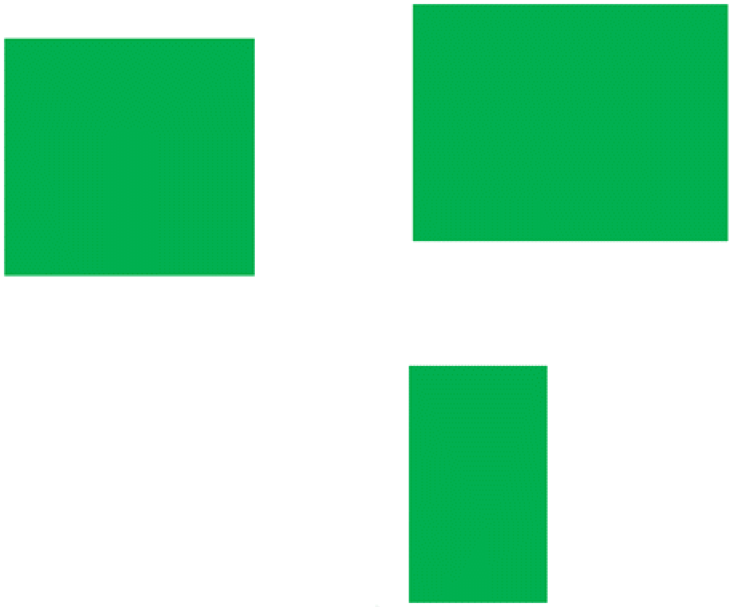





0
1
2
3V
GND


HuskyLens get a total number of ID 1 frame from the result

Exercise 2

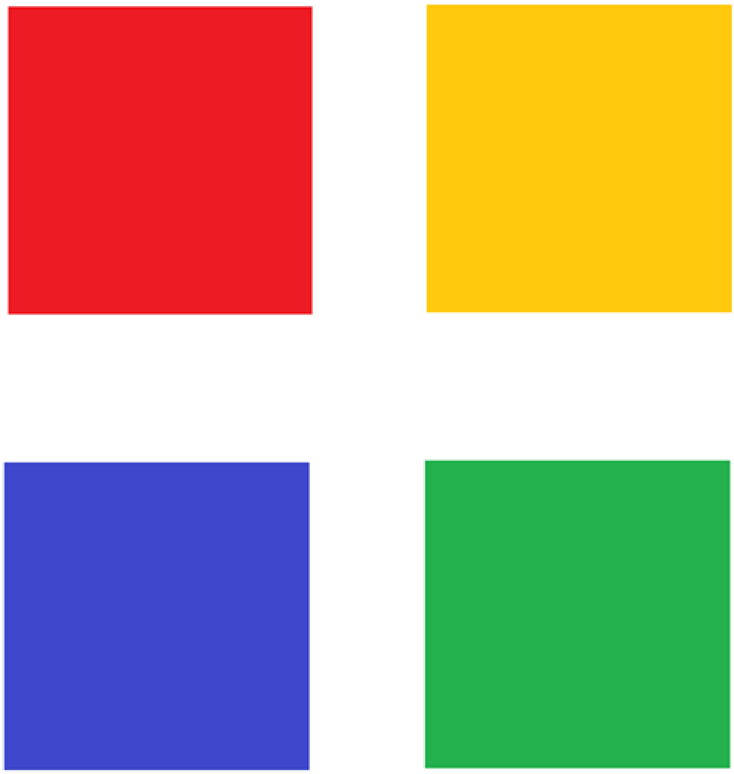


- Use the following Huskylens blocks to get the height and width of each color block. Area = (length) height × width

- Build "number" store "HuskyLens get atotal number of ID 1 frame from the result"
- Build "area". Store the increased area and use the loop to increase the area of the color block gradually

Exercise 3

- ID 1 light up red
- ID 2 light up yellow
- ID 3 light up blue
- ID 4 light up green



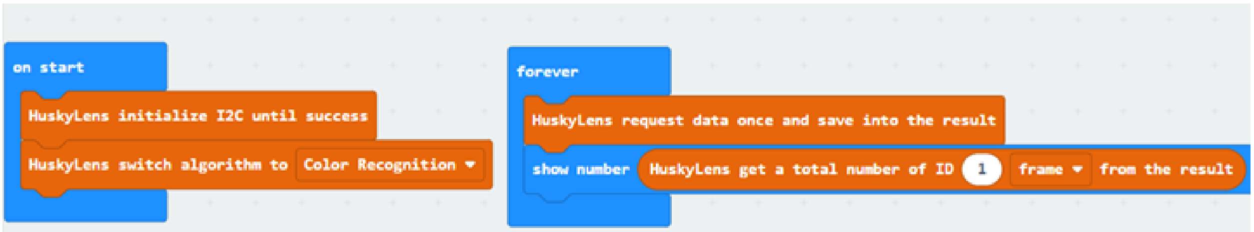
Exercise 4

```
1When Huskylens detects that red color is closest to the center of the screen, the car starts to move forward
2Until Huskylens detects that the purple squre is closest to the center of the screen, the car stops briefly and starts to reverse
3Keep backing up until Huskylens detects the red square closest to the center of the screen again and the car stops.
```

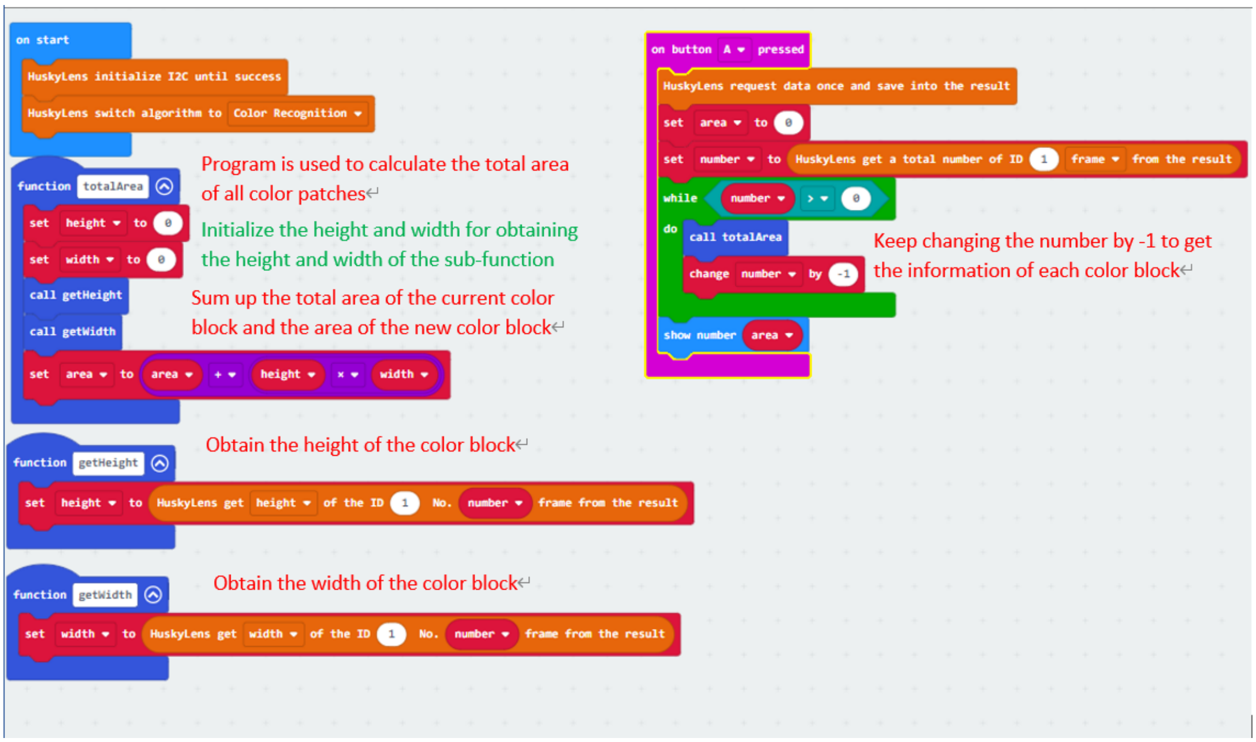
- First, adjust the angle of Huskylens and look at the red square
- Create variables “direction” Record the moving direction of the car to determine the action of the car
- Create variables “finsih” Determine if the car has returned to red square
- if returned, play mid-range C for one second, and set finish as “true”
- Stop playing the tone when finish is “true”

Answer

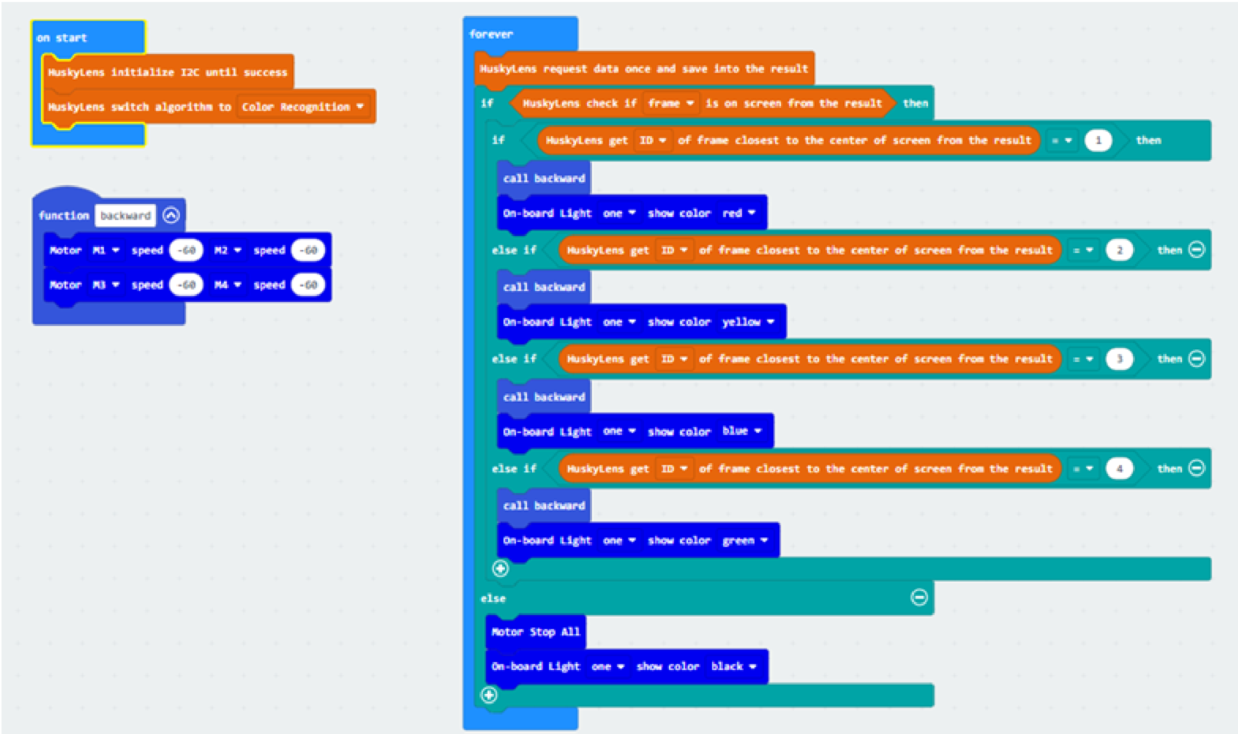
Exercise 1



Exercise 2

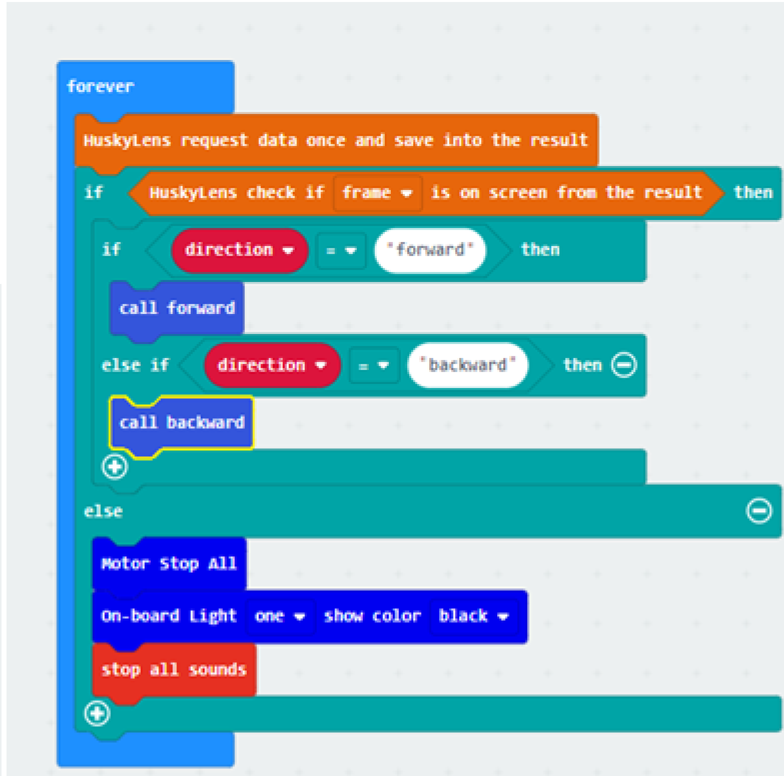


Exercise 3


on start
HuskyLens initialize I2C until success
HuskyLens switch algorithm to Color Recognition
function backward
Motor M1 speed -60 M2 speed -60
Motor M3 speed -60 M4 speed -60
forever
HuskyLens request data once and save into the result
if HuskyLens check if frame is on screen from the result then
if HuskyLens get ID of frame closest to the center of screen from the result = 1 then
call backward
On-board Light one show color red
else if HuskyLens get ID of frame closest to the center of screen from the result = 2 then
call backward
On-board Light one show color yellow
else if HuskyLens get ID of frame closest to the center of screen from the result = 3 then
call backward
On-board Light one show color blue
else if HuskyLens get ID of frame closest to the center of screen from the result = 4 then
call backward
On-board Light one show color green
else
Motor Stop All
On-board Light one show color black

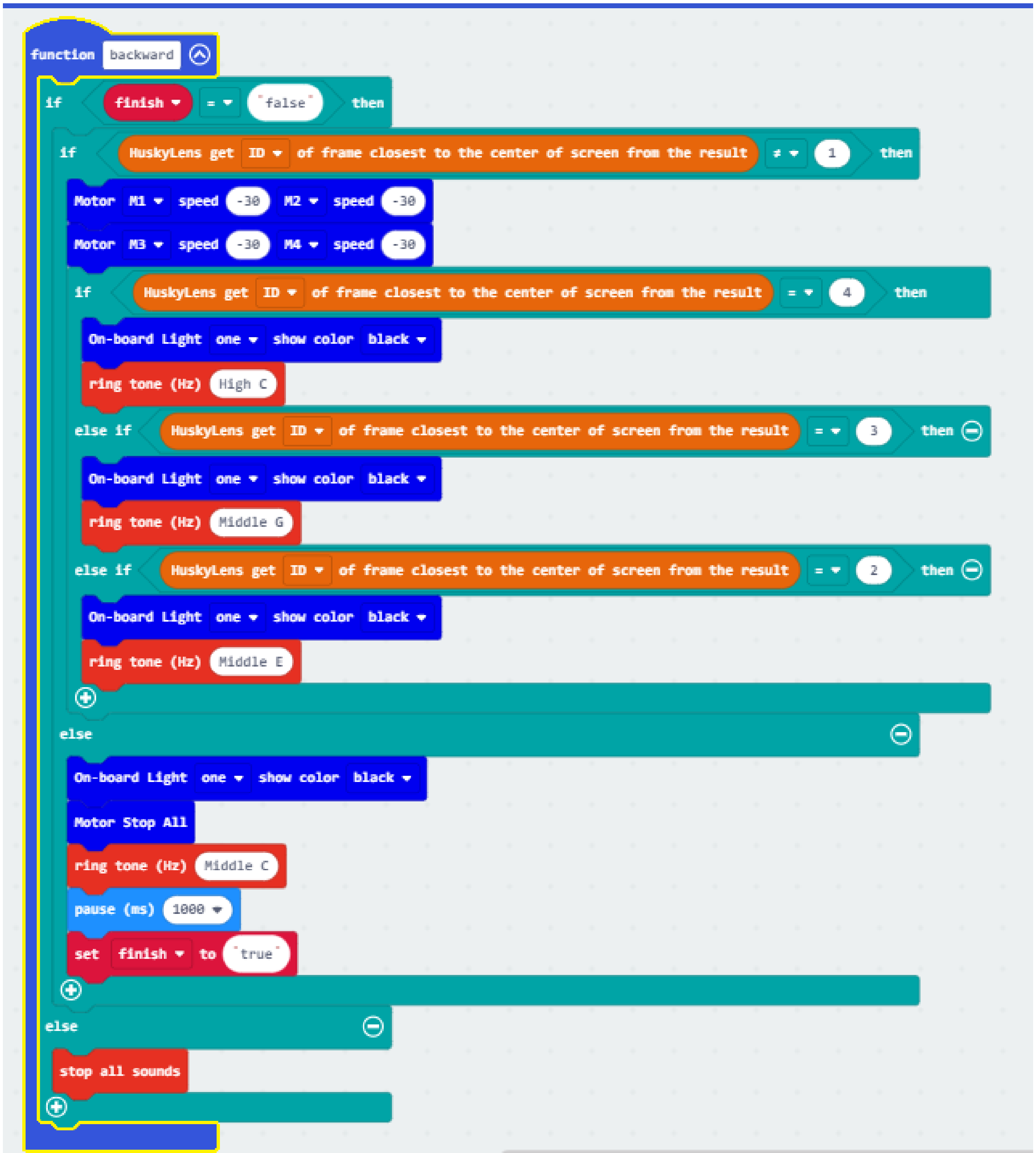
Exercise 4


on start
stop all sounds
On-board Light one show color black
HuskyLens initialize I2C until success
HuskyLens switch algorithm to Color Recognition
set direction to "forward"
set finish to "false"


forever
HuskyLens request data once and save into the result
if HuskyLens check if frame is on screen from the result then
if direction = "forward" then
call forward
else if direction = "backward" then
call backward
else
Motor Stop All
On-board Light one show color black
stop all sounds

Forward function

Backward function



1.5.4 Lesson 4

Introduction

Objective

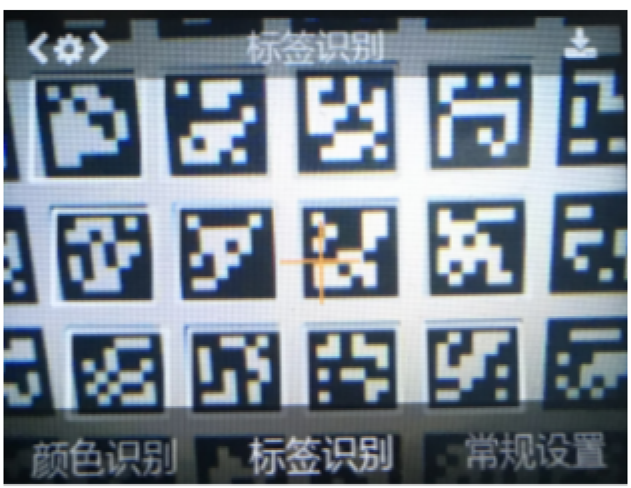
HuskyLens tag recognition function

what is tag recognition

Applications of tag recognition

Huskylens tag recognition

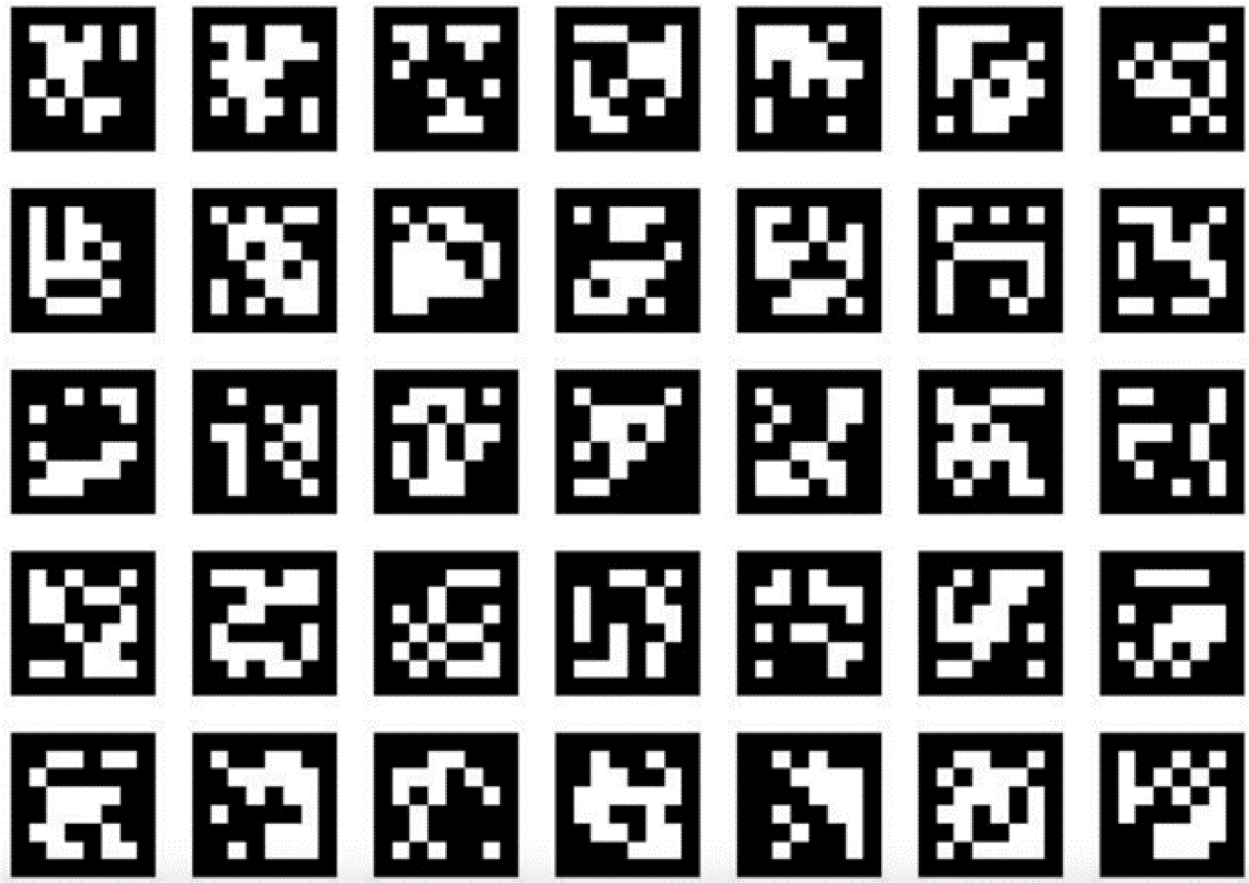
Detecting tag



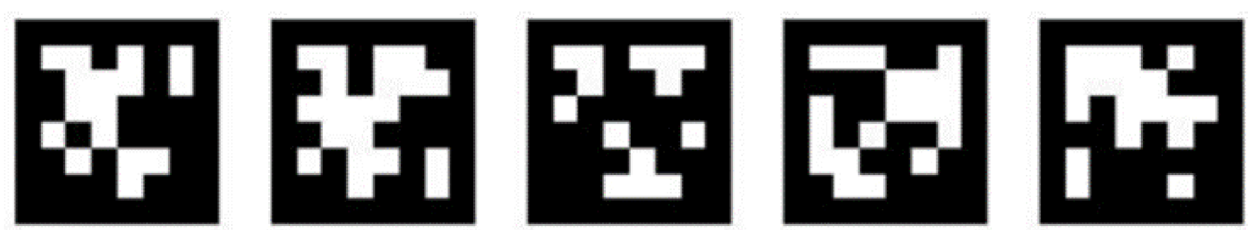
Learning tag

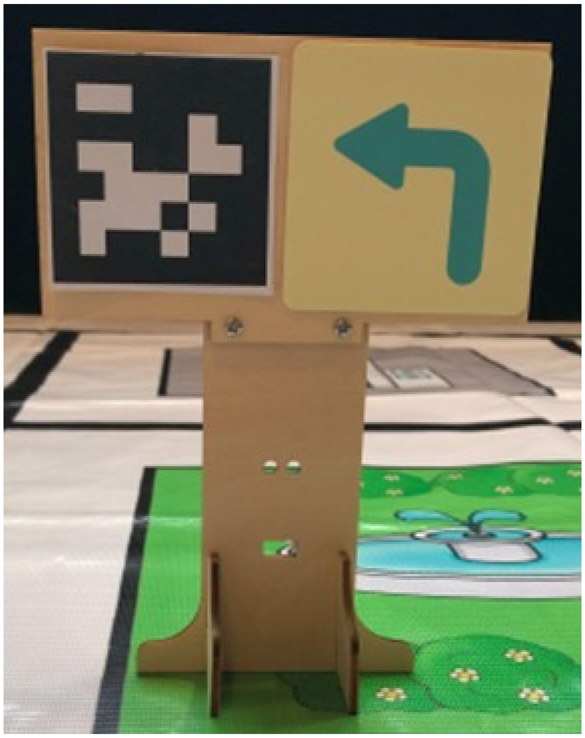
Identifying tag

AprilTag

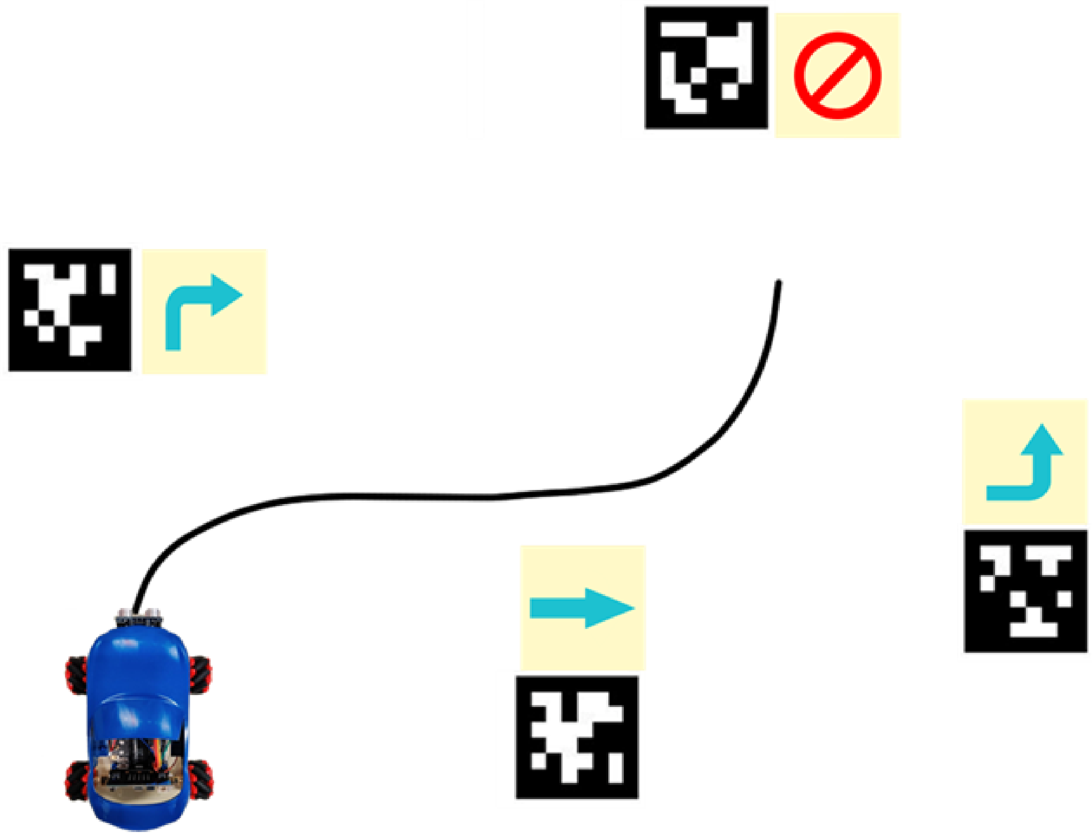


Exercise 1





Exercise 2

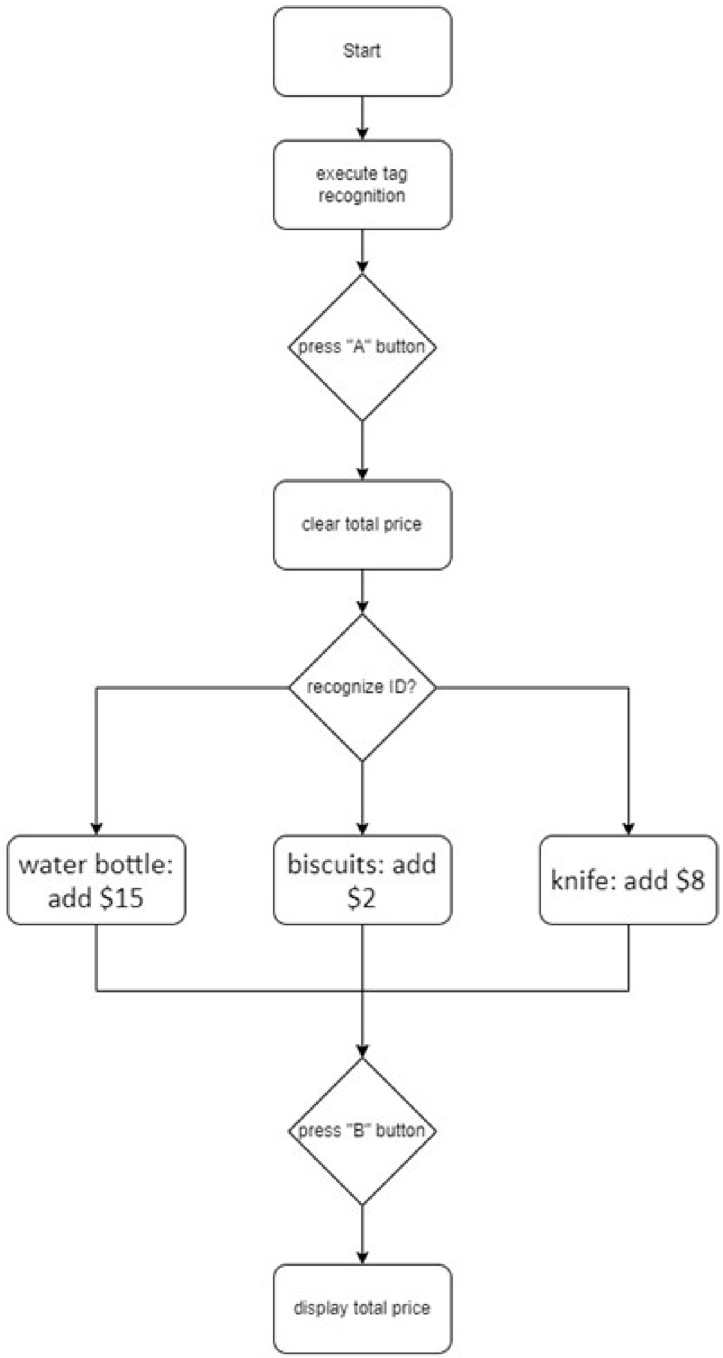


Exercise 3: Self-service supermarket cash register

Mission 1Identify goods

Mission 2Start and end scan

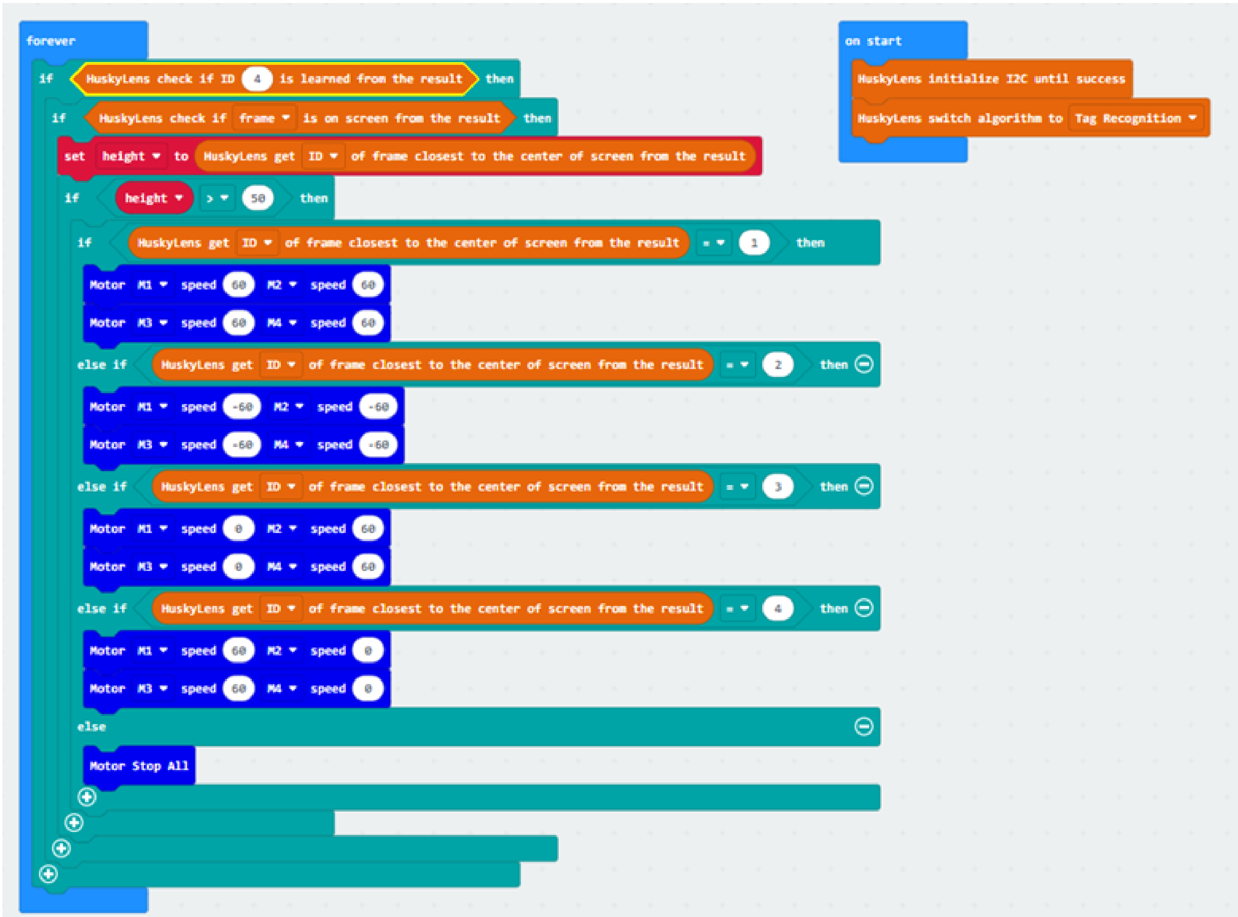
Mission 3goods settlement



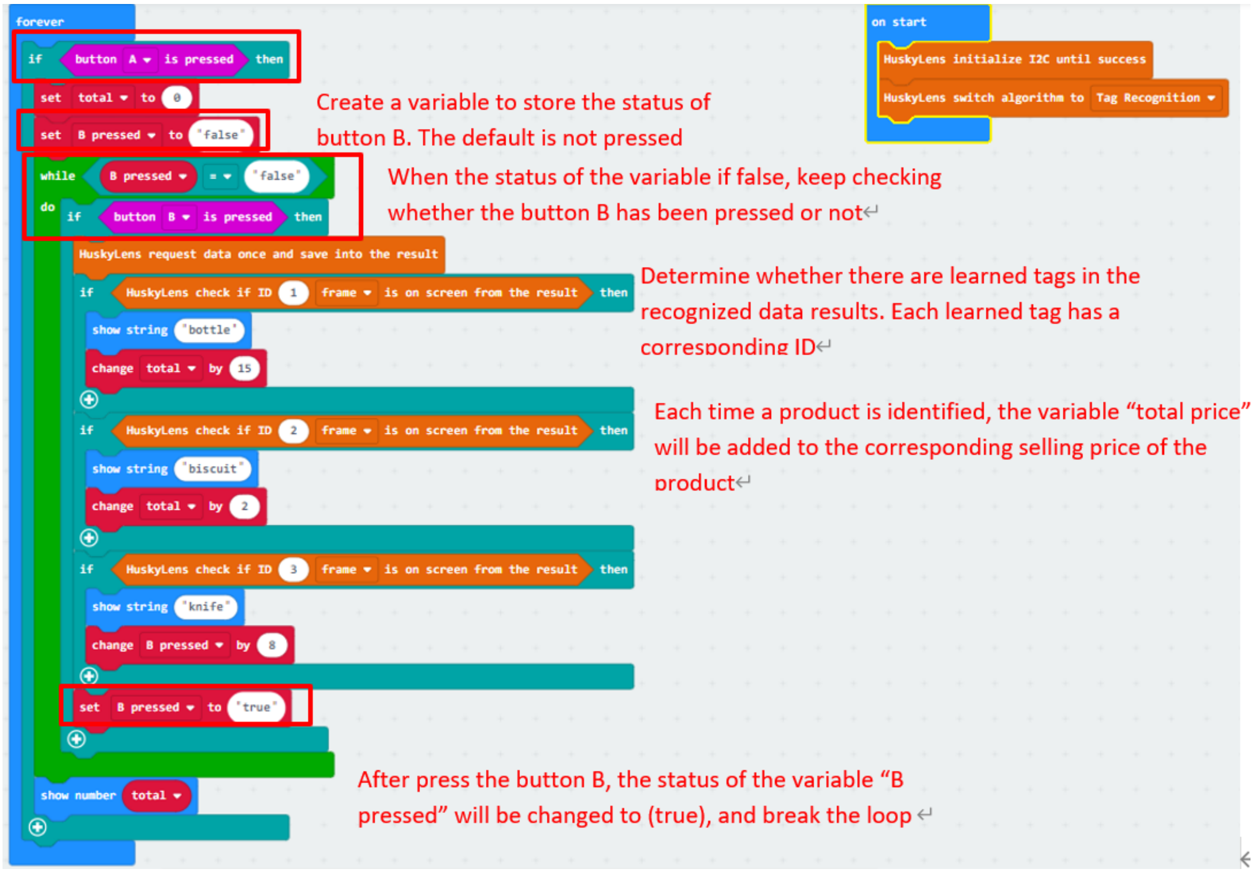
Answer

Exercise 1

Exercise 2


forever
if HuskyLens check if ID 4 is learned from the result then
if HuskyLens check if frame is on screen from the result then
set height to HuskyLens get ID of frame closest to the center of screen from the result
if height > 50 then
if HuskyLens get ID of frame closest to the center of screen from the result = 1 then
Motor M1 speed 60 M2 speed 60
Motor M3 speed 60 M4 speed 60
else if HuskyLens get ID of frame closest to the center of screen from the result = 2 then
Motor M1 speed -60 M2 speed -60
Motor M3 speed -60 M4 speed -60
else if HuskyLens get ID of frame closest to the center of screen from the result = 3 then
Motor M1 speed 0 M2 speed 60
Motor M3 speed 0 M4 speed 60
else if HuskyLens get ID of frame closest to the center of screen from the result = 4 then
Motor M1 speed 60 M2 speed 0
Motor M3 speed 60 M4 speed 0
else
Motor Stop All
on start
HuskyLens initialize I2C until success
HuskyLens switch algorithm to Tag Recognition

Exercise 3


forever
if button A is pressed then
set total to 0
set B pressed to "false"
while B pressed = "false"
do if button B is pressed then
HuskyLens request data once and save into the result
if HuskyLens check if ID 1 frame is on screen from the result then
show string "bottle"
change total by 15
if HuskyLens check if ID 2 frame is on screen from the result then
show string "biscuit"
change total by 2
if HuskyLens check if ID 3 frame is on screen from the result then
show string "knife"
change B pressed by 8
set B pressed to "true"
show number total
on start
HuskyLens initialize I2C until success
HuskyLens switch algorithm to Tag Recognition
Create a variable to store the status of button B. The default is not pressed
When the status of the variable if false, keep checking whether the button B has been pressed or not
Determine whether there are learned tags in the recognized data results. Each learned tag has a corresponding ID
Each time a product is identified, the variable "total price" will be added to the corresponding selling price of the product
After press the button B, the status of the variable "B pressed" will be changed to (true), and break the loop

1.5.5 Lesson 5

Introduction

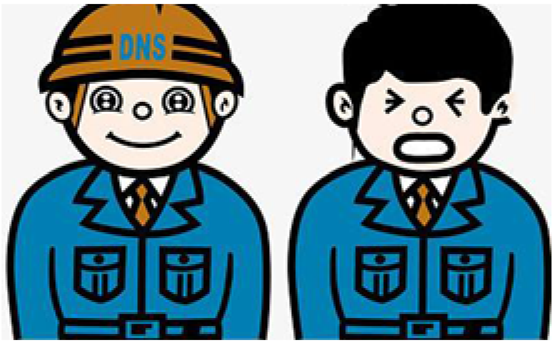
Objective

HuskyLens object classification

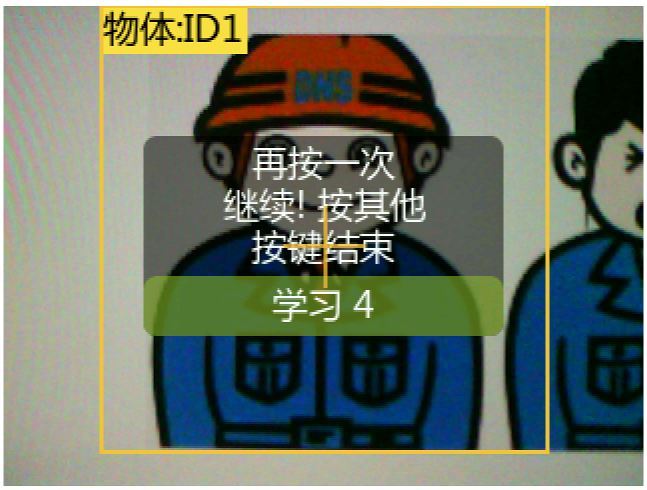
what is object classification

HuskyLensDifferences between object classification and other functions

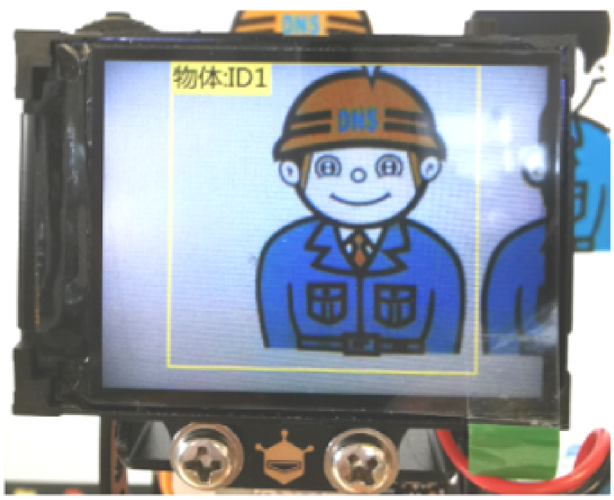
Huskylens' object classification



learning objects

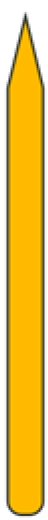


Identification of labeled objects



Exercise 1Symbol classification

Exercise 2Location Recognition

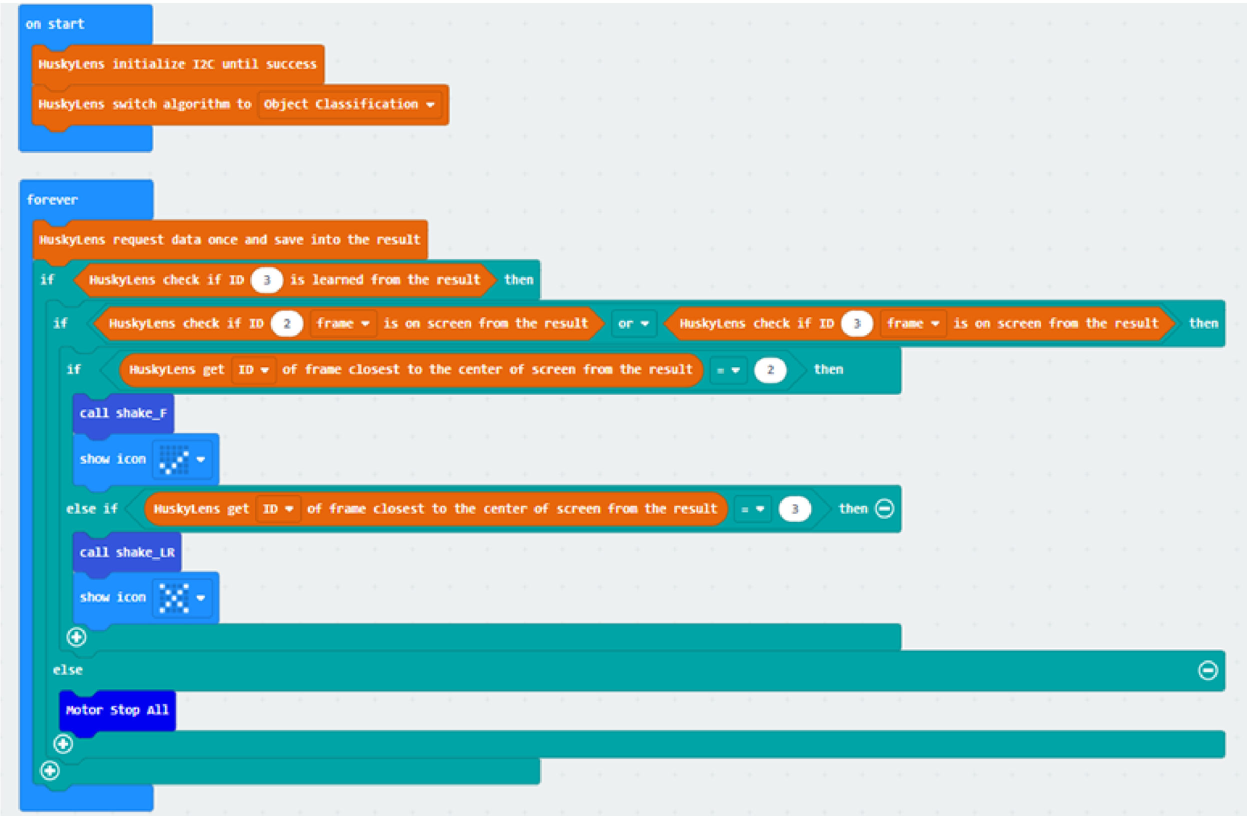


Exercise 3Line tracking

Answer

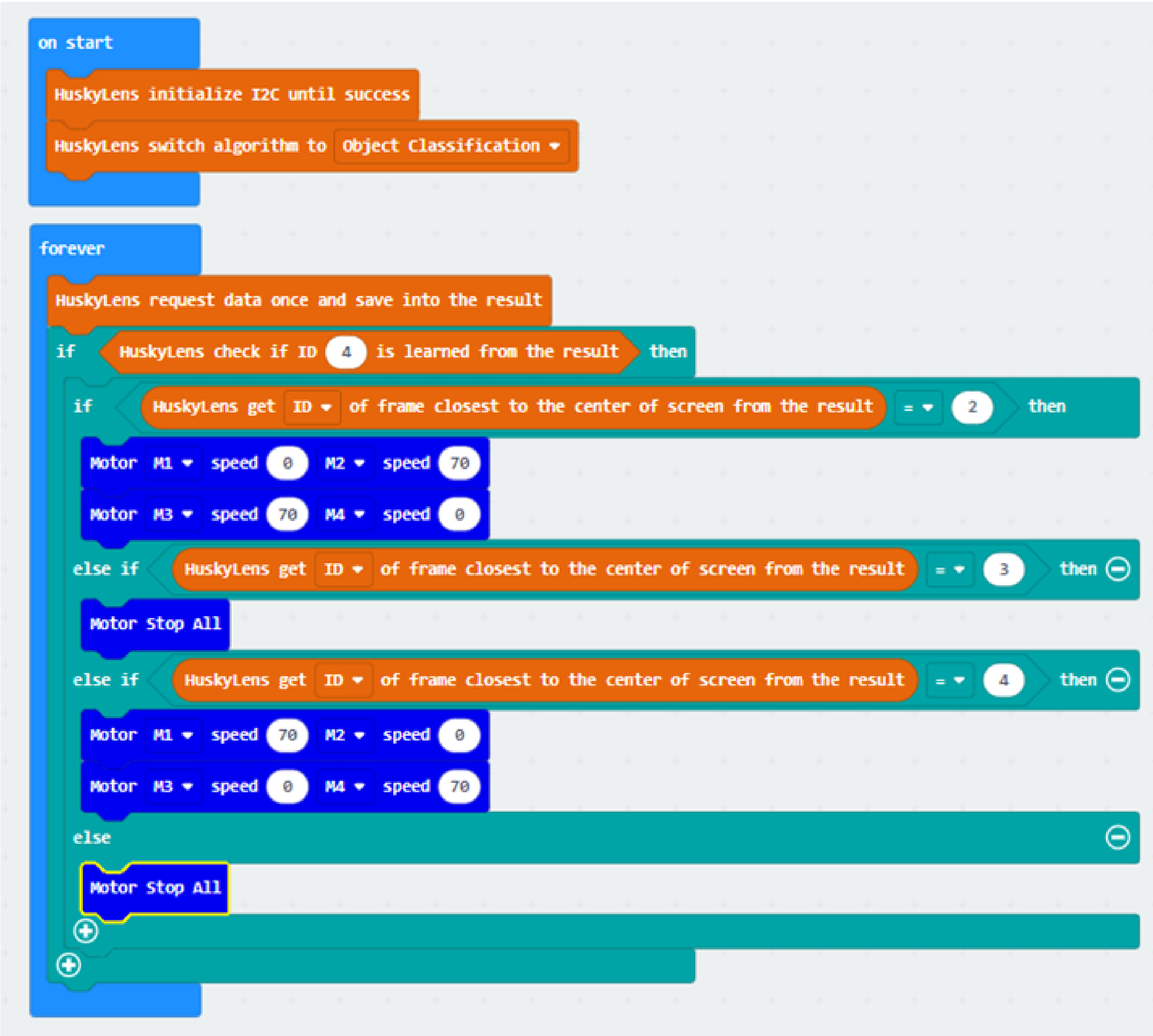
Exercise 1

The main body of the program



The program for shaking the car to the left and right and to the front and back

Exercise 2


on start
HuskyLens initialize I2C until success
HuskyLens switch algorithm to Object Classification
forever
HuskyLens request data once and save into the result
if HuskyLens check if ID 4 is learned from the result then
if HuskyLens get ID of frame closest to the center of screen from the result = 2 then
Motor M1 speed 0 M2 speed 70
Motor M3 speed 70 M4 speed 0
else if HuskyLens get ID of frame closest to the center of screen from the result = 3 then
Motor Stop All
else if HuskyLens get ID of frame closest to the center of screen from the result = 4 then
Motor M1 speed 70 M2 speed 0
Motor M3 speed 0 M4 speed 70
else
Motor Stop All

Exercise 3


on start
HuskyLens initialize I2C until success
HuskyLens switch algorithm to Object Classification
forever
HuskyLens request data once and save into the result
if HuskyLens check if ID 4 is learned from the result then
Line is straight
if HuskyLens get ID of frame closest to the center of screen from the result = 2 then
Motor M1 speed 70 M2 speed 70
Motor M3 speed 70 M4 speed 70
Move forward
Line change to top left
else if HuskyLens get ID of frame closest to the center of screen from the result = 3 then
Motor M1 speed -40 M2 speed 60
Motor M3 speed 40 M4 speed 60
Move top left
Line change to top right
else if HuskyLens get ID of frame closest to the center of screen from the result = 4 then
Motor M1 speed 60 M2 speed -40
Motor M3 speed 60 M4 speed 40
Move top right
else
Motor Stop All
pause (ms) 10

1.5.6 Lesson 6

Introduction

Objective

HuskyLens line tracking function

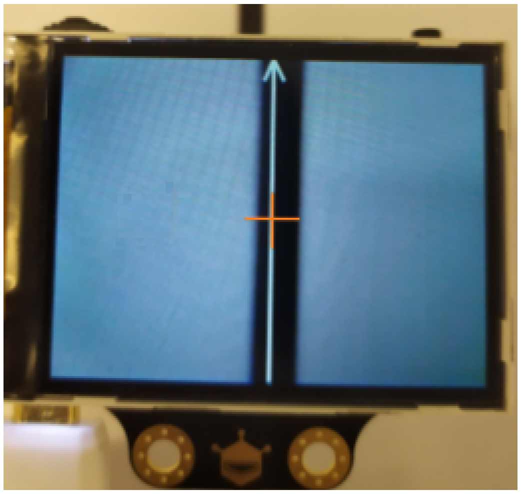
what is line tracking function

Application of line tracking

- For navigation in public places such as shopping malls or museums
- For industrial transportation, save a lot of space compared to traditional conveyor belts
- For restaurants, deliver food to guests

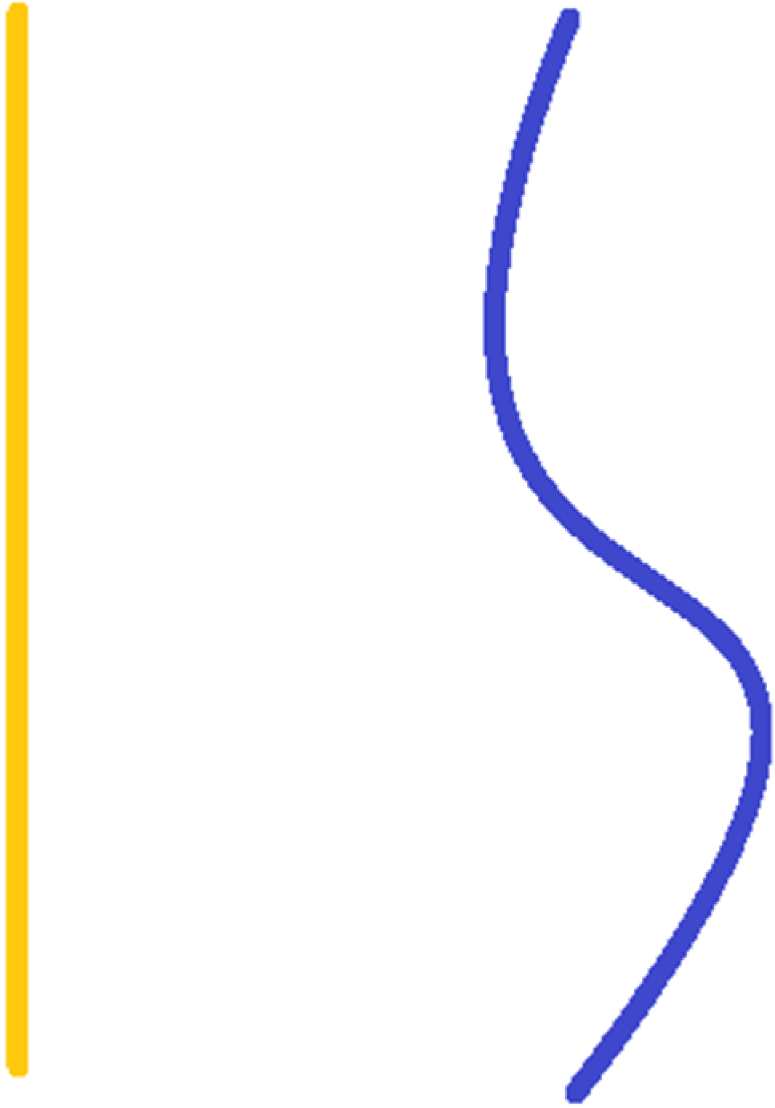
Huskylens' line tracking function

learning

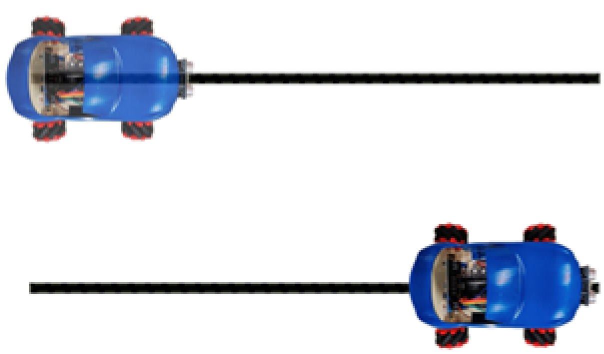


Line tracking

Exercise 1

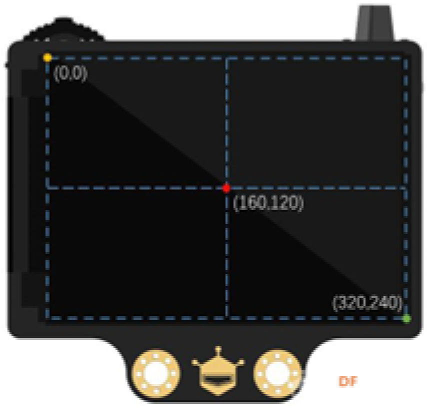


Exercise 2

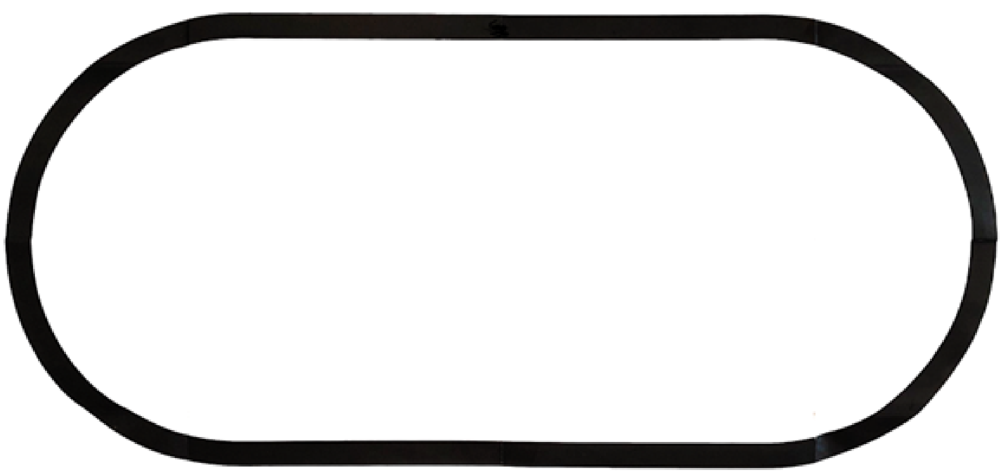


- Use different X beginning to decide whether the car should go forward, turn left or turn right

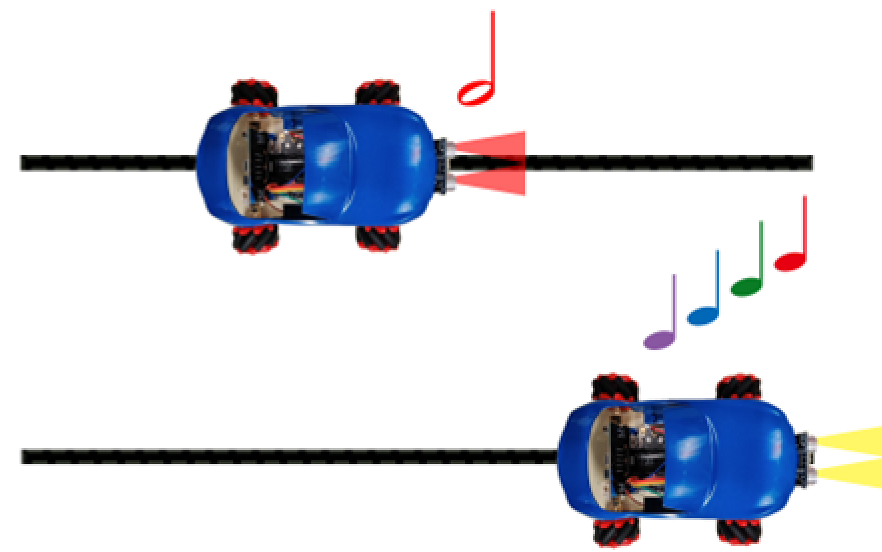
- X coordinate range0 - 320
- Y coordinate range0 - 240



Exercise 3

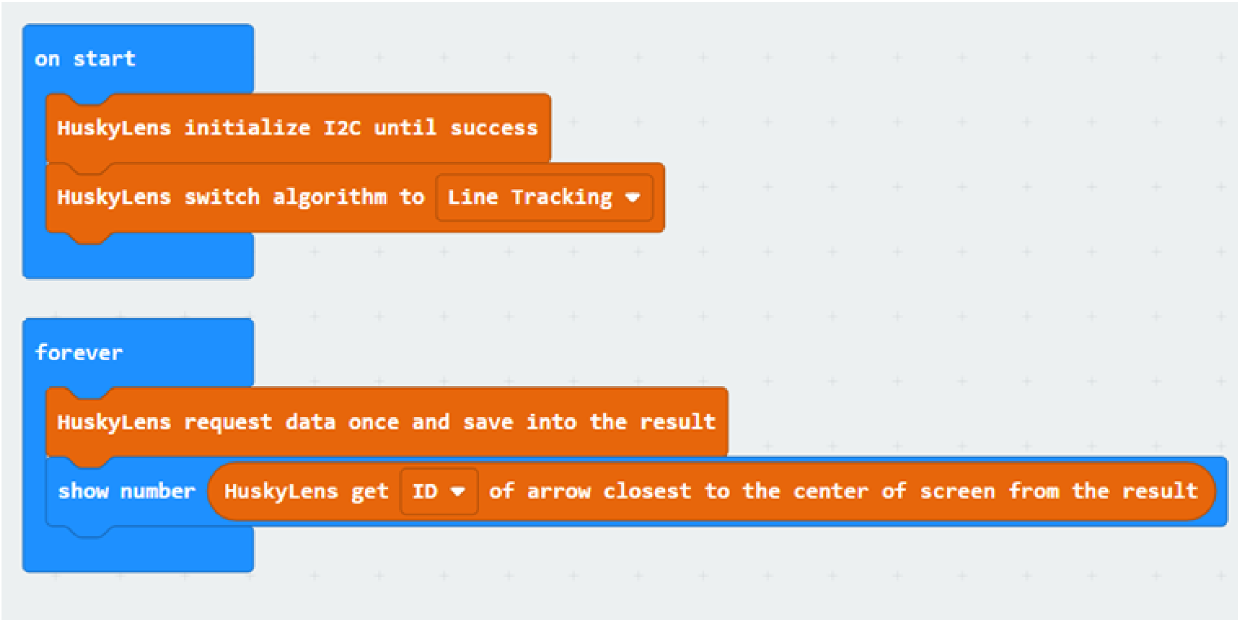


Exercise 4Autonomous line tracking cars:



Answer

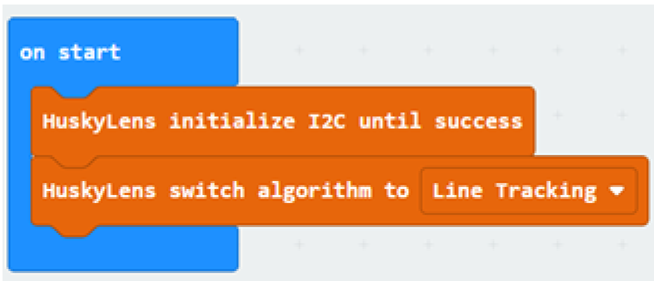
Exercise 1

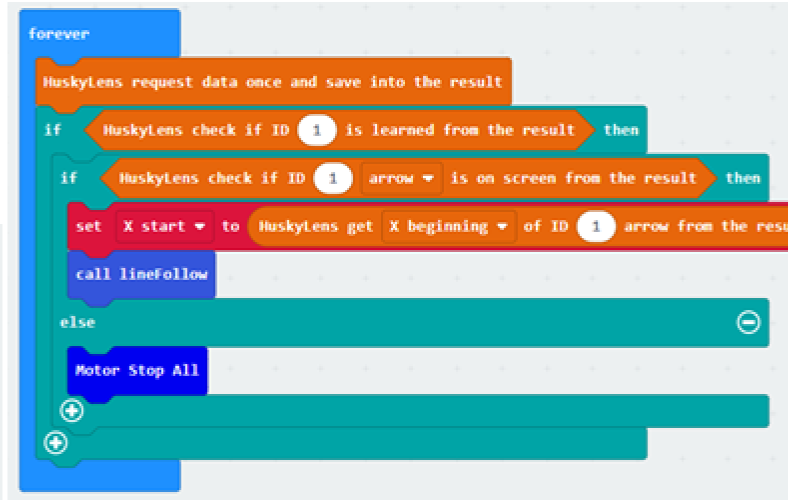


Exercise 2

Exercise 3

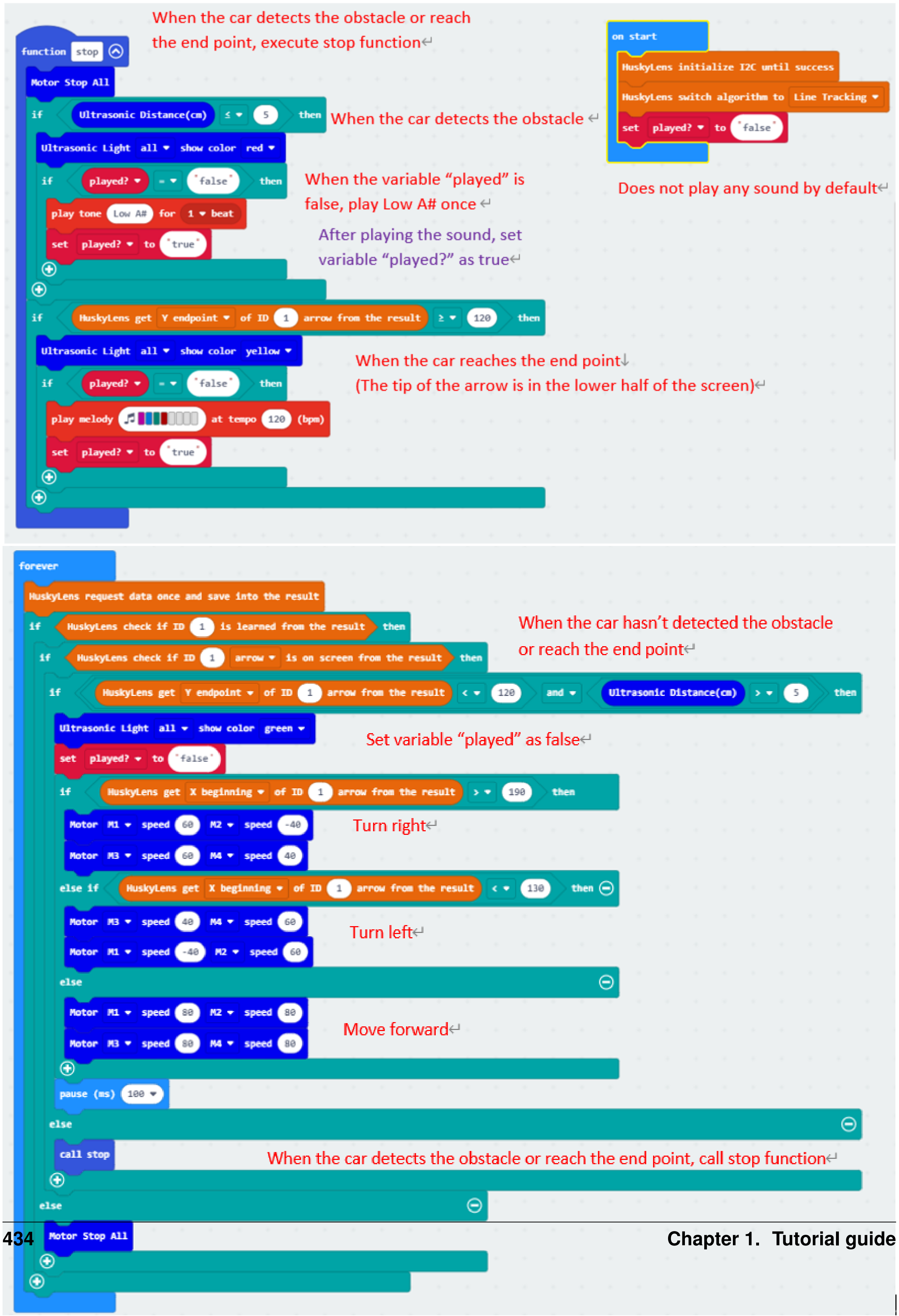
Program


on start
HuskyLens initialize I2C until success
HuskyLens switch algorithm to Line Tracking


forever
HuskyLens request data once and save into the result
if HuskyLens check if ID 1 is learned from the result then
if HuskyLens check if ID 1 arrow is on screen from the result then
set X start to HuskyLens get X beginning of ID 1 arrow from the resu
call lineFollow
else
Motor Stop All


function lineFollow
if HuskyLens get Y endpoint of ID 1 arrow from the result < 120 then
if X start ≥ 170 and X start ≤ 190 then
Motor M1 speed 50 M2 speed -30
Motor M3 speed 50 M4 speed 30
When X beginning is skewed to the right of the screen
Slowly turn right
else if X start > 190 then
Motor M1 speed 60 M2 speed -40
Motor M3 speed 60 M4 speed 40
When X is right of the screen
turn right
else if X start ≥ 130 and X start ≤ 150 then
Motor M1 speed -30 M2 speed 50
Motor M3 speed 30 M4 speed 50
When X beginning is skewed to the left of the screen
Slowly turn left
else if X start < 130 then
Motor M1 speed -40 M2 speed 60
Motor M3 speed 40 M4 speed 60
When X is left of the screen
turn left
else
Motor M1 speed 80 M2 speed 80
Motor M3 speed 80 M4 speed 80
When X is middle of the screen
forward
else
Motor Stop All
pause (ms) 100

Exercise 4


When the car detects the obstacle or reach the end point, execute stop function
function stop
Motor Stop All
if Ultrasonic Distance(cm) ≤ 5 then
When the car detects the obstacle
Ultrasonic Light all show color red
if played? = "false" then
When the variable "played" is false, play Low A# once
play tone Low A# for 1 beat
set played? to "true"
After playing the sound, set variable "played?" as true
if HuskyLens get Y endpoint of ID 1 arrow from the result ≥ 120 then
Ultrasonic Light all show color yellow
When the car reaches the end point
(The tip of the arrow is in the lower half of the screen)
if played? = "false" then
play melody at tempo 120 (bpm)
set played? to "true"
on start
HuskyLens initialize I2C until success
HuskyLens switch algorithm to Line Tracking
set played? to "false"
Does not play any sound by default
forever
HuskyLens request data once and save into the result
if HuskyLens check if ID 1 is learned from the result then
if HuskyLens check if ID 1 arrow is on screen from the result then
When the car hasn't detected the obstacle or reach the end point
if HuskyLens get Y endpoint of ID 1 arrow from the result < 120 and Ultrasonic Distance(cm) > 5 then
Ultrasonic Light all show color green
set played? to "false"
Set variable "played" as false
if HuskyLens get X beginning of ID 1 arrow from the result > 190 then
Motor M1 speed 60 M2 speed -40
Motor M3 speed 60 M4 speed 40
Turn right
else if HuskyLens get X beginning of ID 1 arrow from the result < 130 then
Motor M3 speed 40 M4 speed 60
Motor M1 speed -40 M2 speed 60
Turn left
else
Motor M1 speed 80 M2 speed 80
Motor M3 speed 80 M4 speed 80
Move forward
pause (ms) 100
else
call stop
When the car detects the obstacle or reach the end point, call stop function
else
Motor Stop All

1.5.7 Lesson 7

Introduction

Objective

HuskyLens object tracking function

What is object tracking function

Application of object tracking

Event detection

Directing traffic

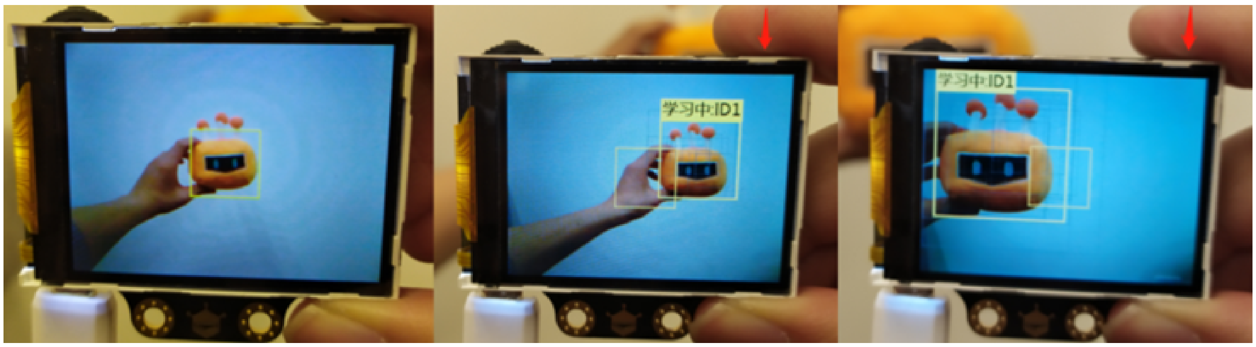
Human-computer interaction

Virtual Reality

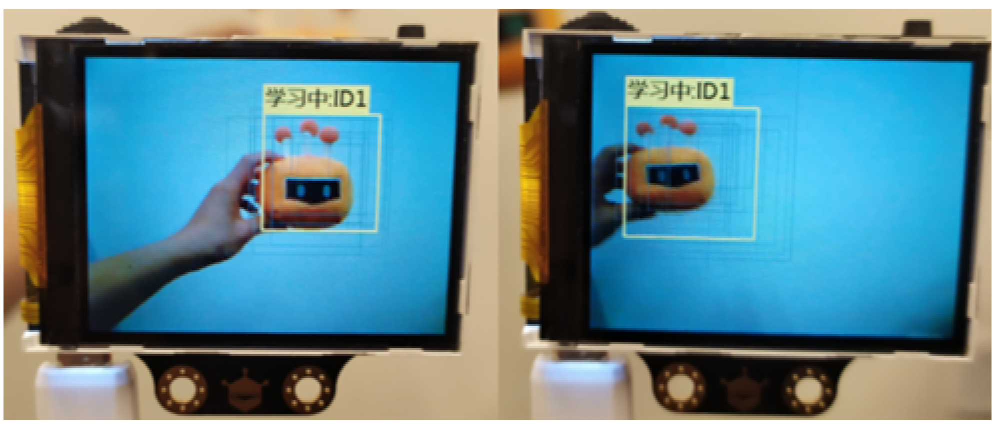
Huskylens' object tracking

HuskyLens's object tracking function can be divided into two parts: learning objects and tracking objects.

Learning objects

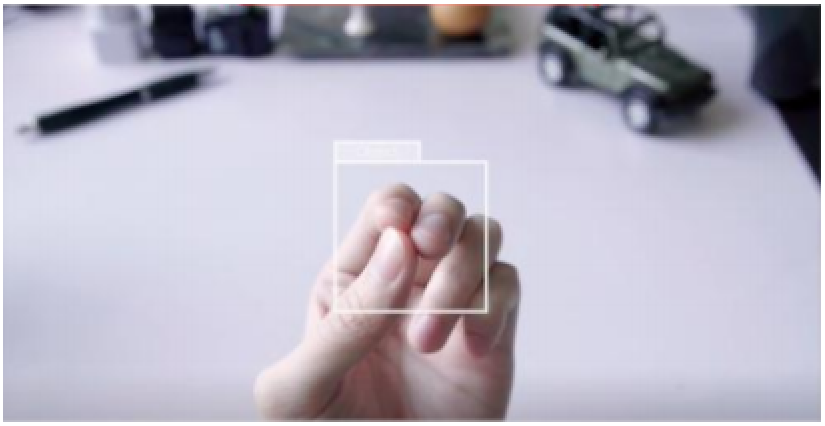


Tracking objects

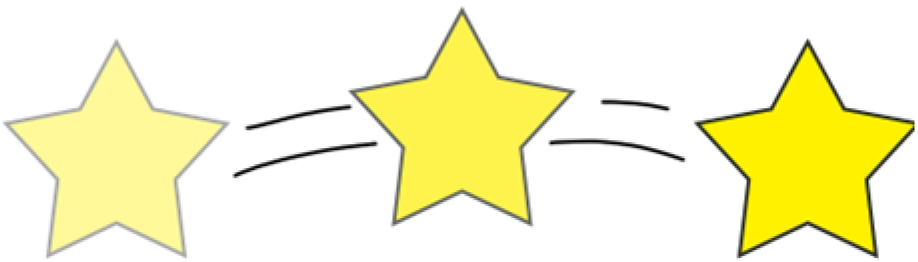


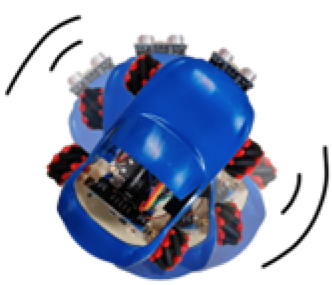
The method of turning off tracking and learning at the same time

Exercise 1

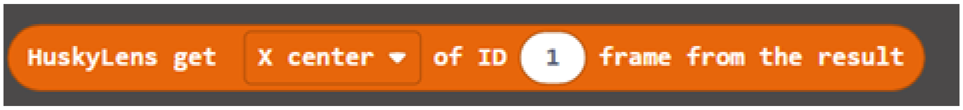


Exercise 2





- Use "pause" blocks to control how long the car takes to turn in each direction
- Use the X center to represent the object position



Exercise 3

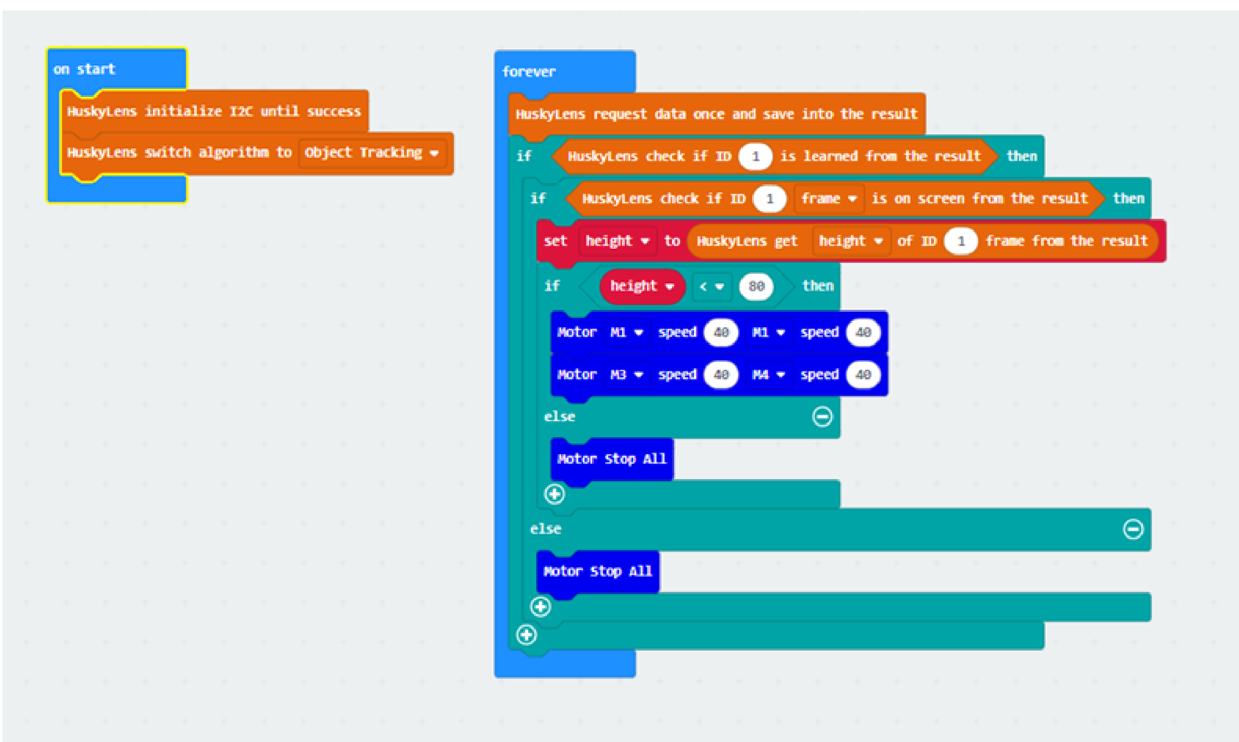
- How do we usually know if we are near an object by our eyes?
- The learning frame of HuskyLens object tracking is preset to 70*70 square units.
- Learning objects at longer distances is more effective

Exercise 4

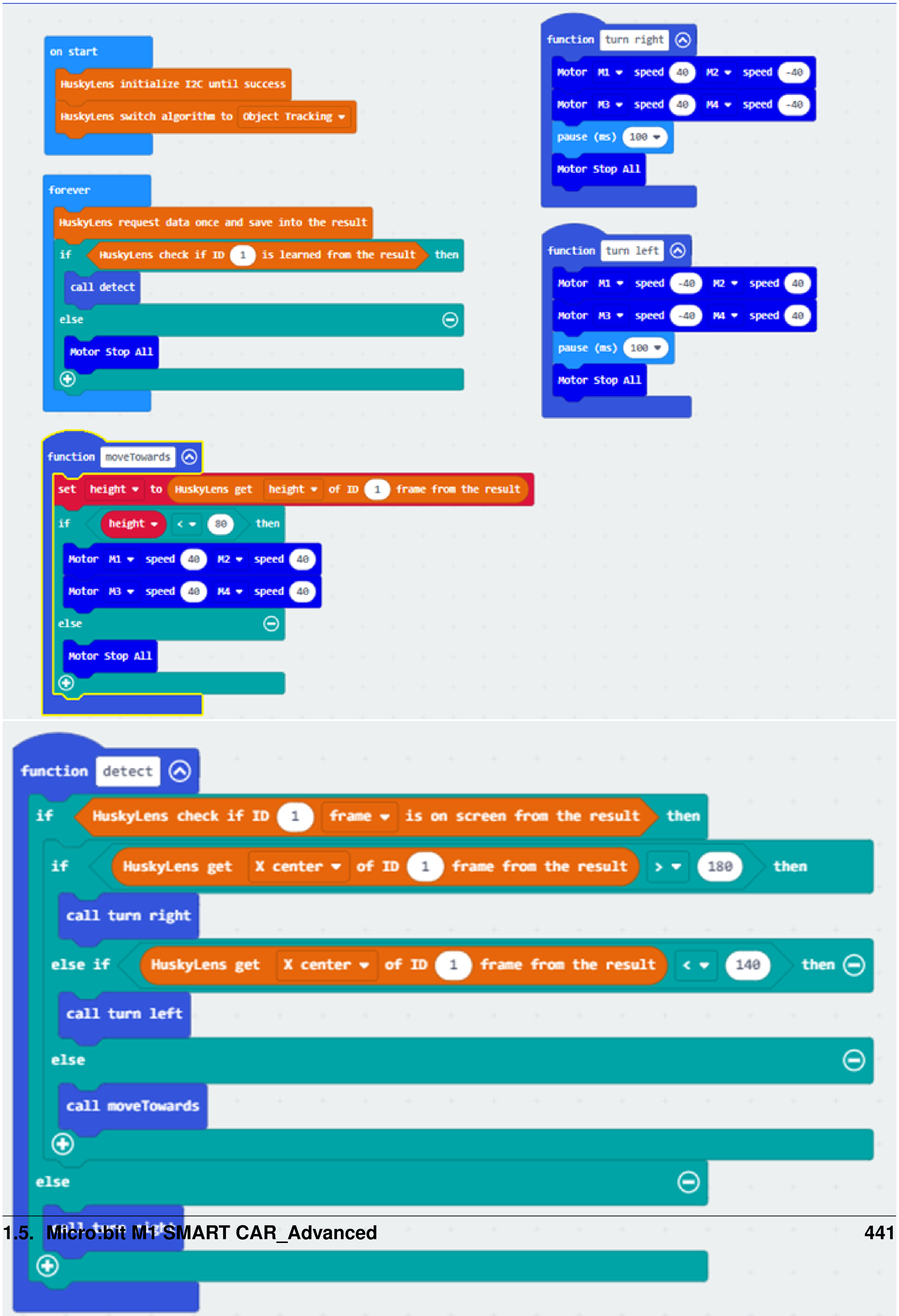
Answer

Exercise 2

Exercise 3



Exercise 4


on start
HuskyLens initialize I2C until success
HuskyLens switch algorithm to Object Tracking
forever
HuskyLens request data once and save into the result
if HuskyLens check if ID 1 is learned from the result then
call detect
else
Motor Stop All
function turn right
Motor M1 speed 40 M2 speed -40
Motor M3 speed 40 M4 speed -40
pause (ms) 100
Motor Stop All
function turn left
Motor M1 speed -40 M2 speed 40
Motor M3 speed -40 M4 speed 40
pause (ms) 100
Motor Stop All
function moveTowards
set height to HuskyLens get height of ID 1 frame from the result
if height < 80 then
Motor M1 speed 40 M2 speed 40
Motor M3 speed 40 M4 speed 40
else
Motor Stop All
function detect
if HuskyLens check if ID 1 frame is on screen from the result then
if HuskyLens get X center of ID 1 frame from the result > 180 then
call turn right
else if HuskyLens get X center of ID 1 frame from the result < 140 then
call turn left
else
call moveTowards
else

1.5.8 Lesson 8

Introduction

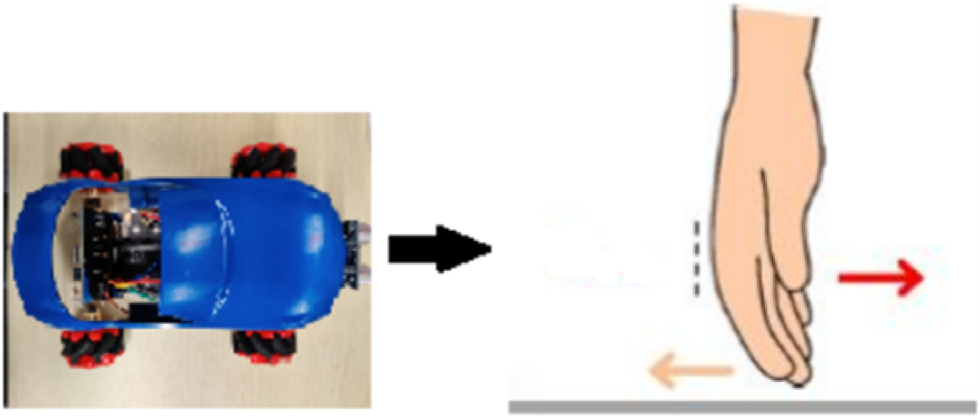
Objective

HuskyLens Object tracking function

HuskyLens Advanced Application of Object Tracking

Obstacle detection: HuskyLens object tracking v.s. ultrasonic sensor

Exercise 1



Exercise 2

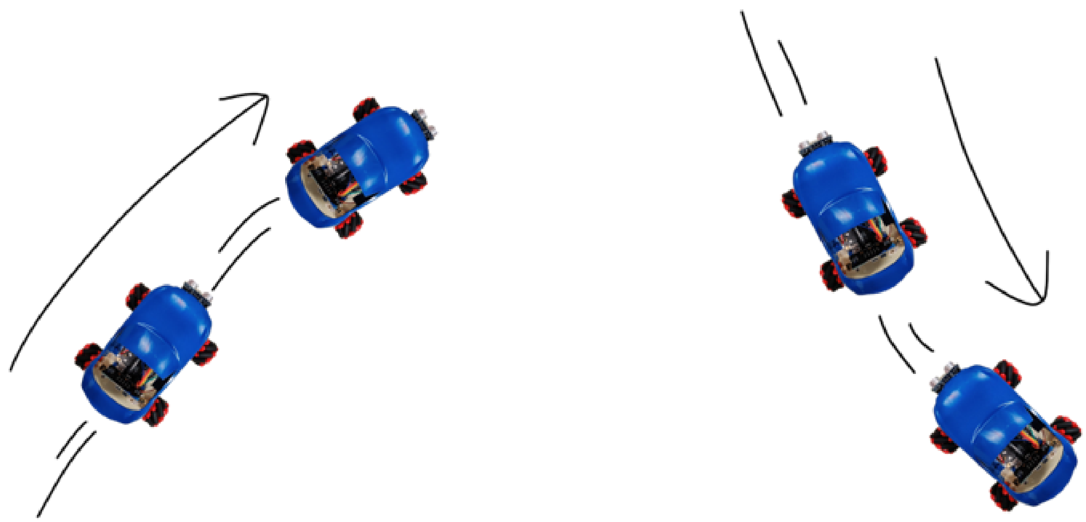
- You can divide the screen into three areas by referring to Exercise 2 in Lesson 6.
- Adjust the direction of the car first, and then continue to move forward.

Exercise 3

Car's action when objects are in different positions

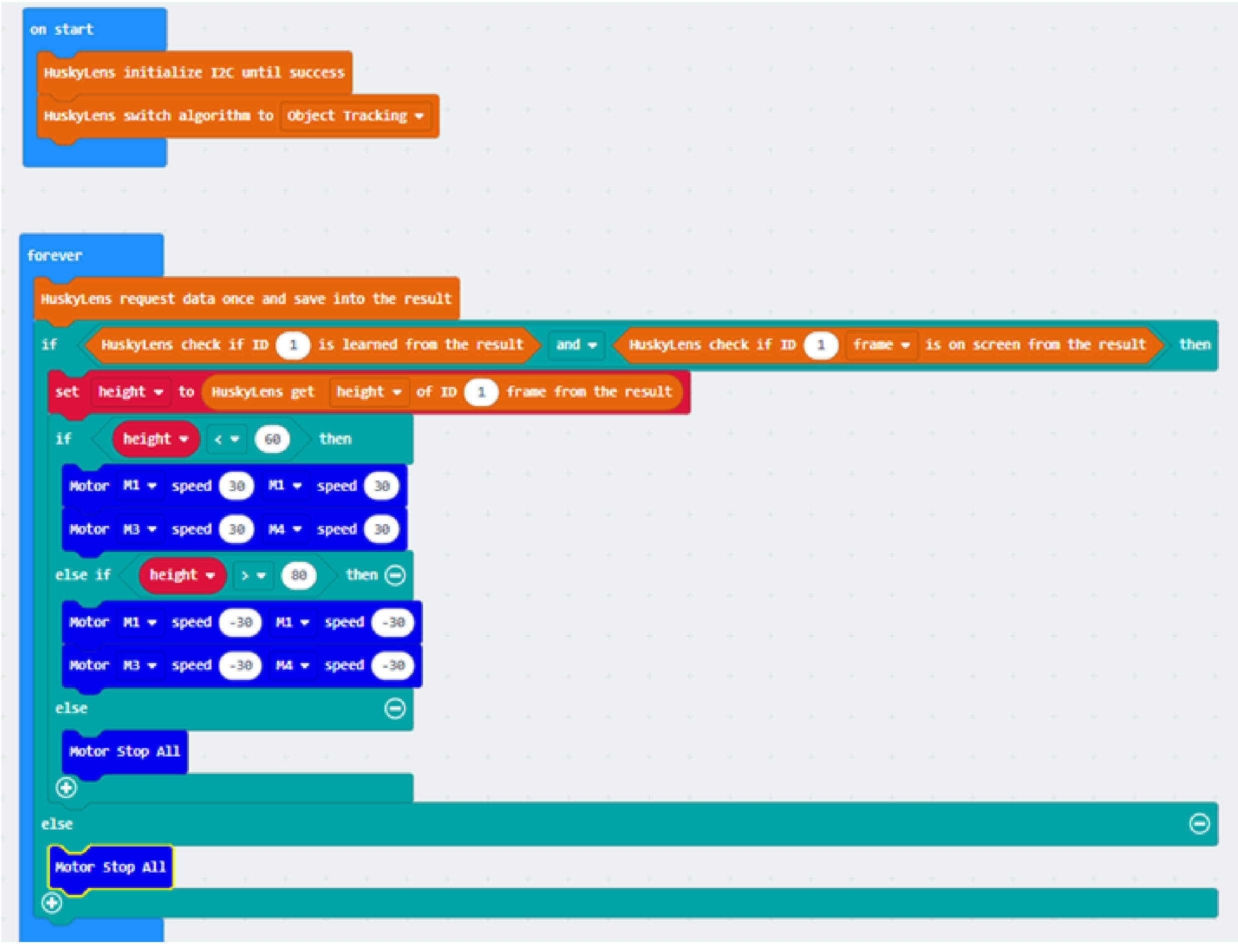
Object distance	Object in screen position			Color ofuUltrasonic light
	left	middle	right	
Far away				Green
Moderate				Yellow
Close				Red

Challenge

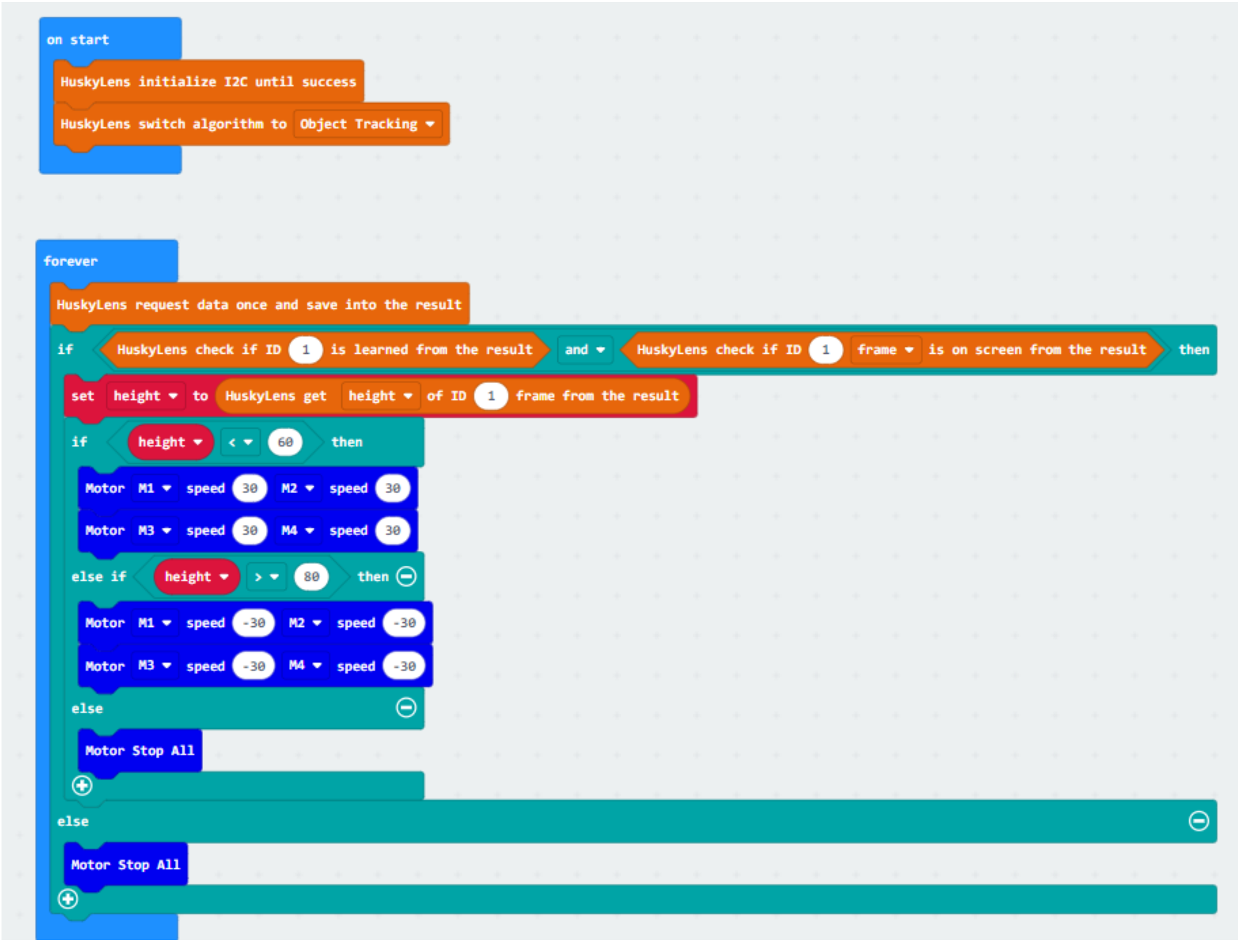


Answer

Exercise 1


on start
HuskyLens initialize I2C until success
HuskyLens switch algorithm to Object Tracking
forever
HuskyLens request data once and save into the result
if HuskyLens check if ID 1 is learned from the result and HuskyLens check if ID 1 frame is on screen from the result then
set height to HuskyLens get height of ID 1 frame from the result
if height < 60 then
Motor M1 speed 30 M1 speed 30
Motor M3 speed 30 M4 speed 30
else if height > 80 then
Motor M1 speed -30 M1 speed -30
Motor M3 speed -30 M4 speed -30
else
Motor Stop All
else
Motor Stop All

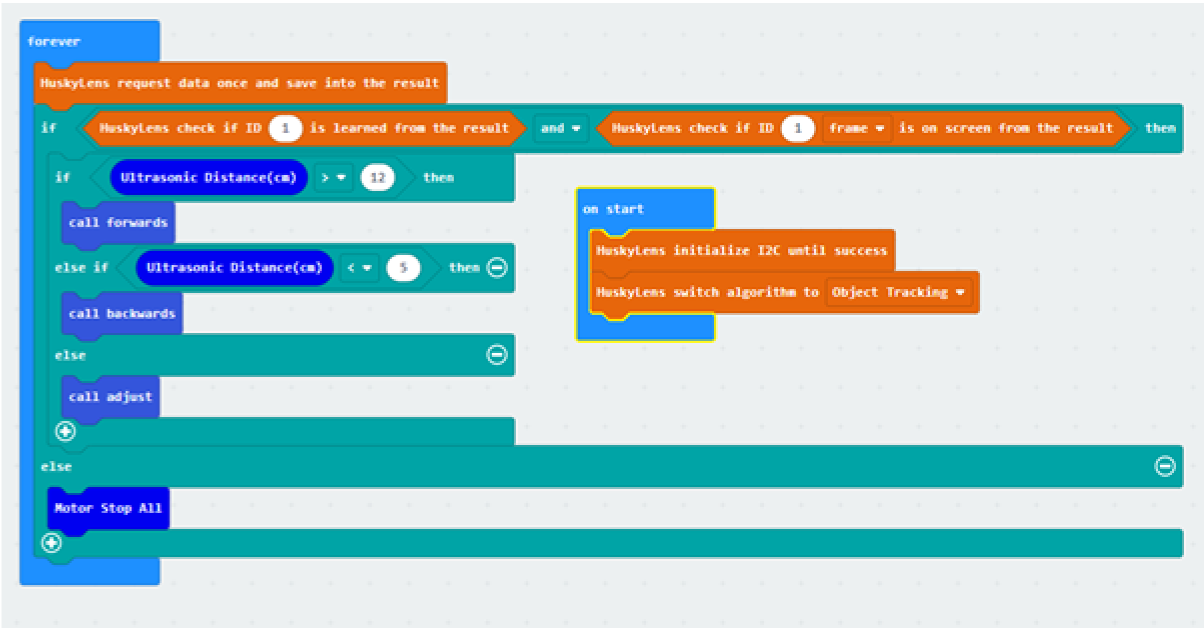
Exercise 2



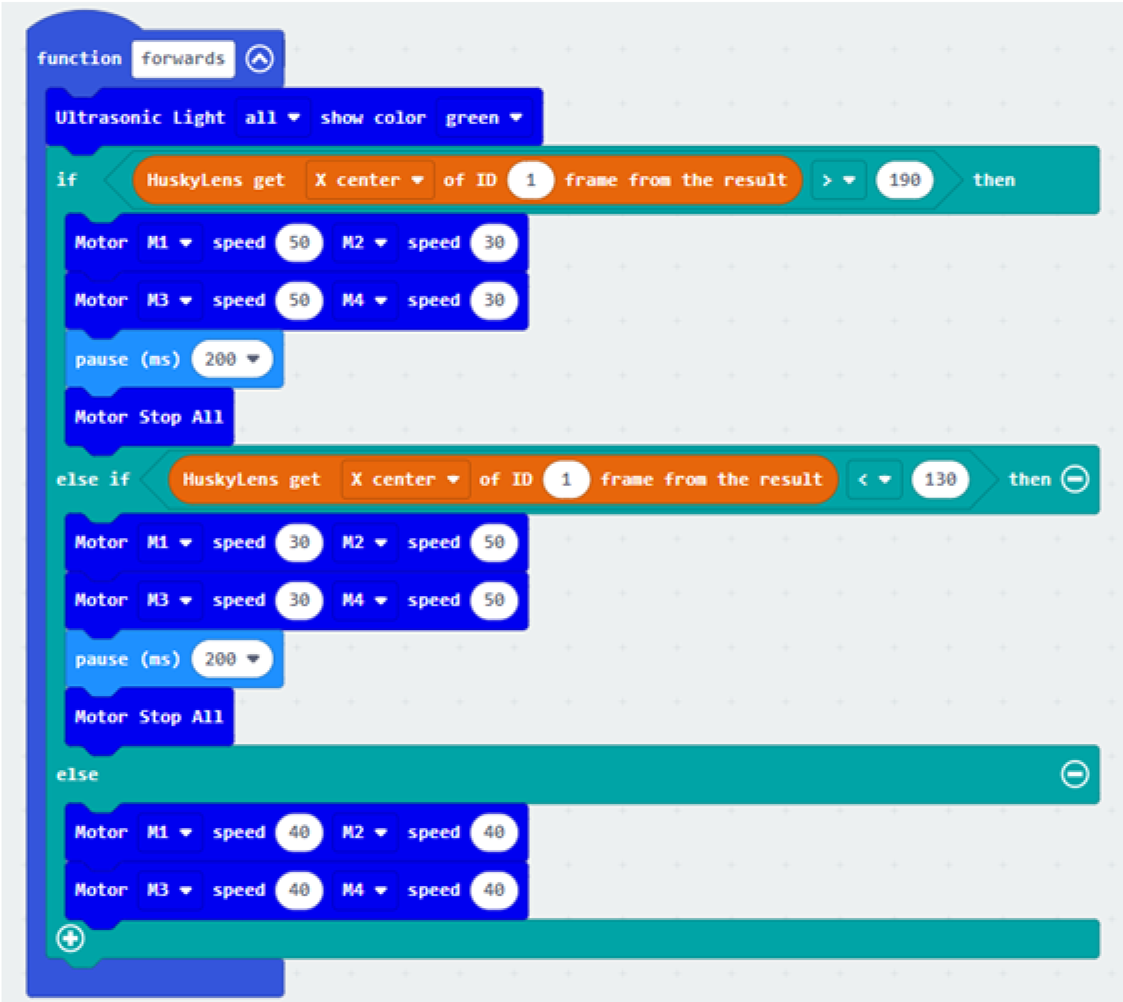
Exercise 3

- The object is large and only one feature of the object is learned.Ensure that the sensor receives the bounce back signal
- It is important to include enough angles and distances when learning with HuskyLens.Since the angle of the lens on the mount is fixed, and there are differences in the appearance of objects seen at different angles and distances, it is important to ensure that HuskyLens can recognize objects in different positions.

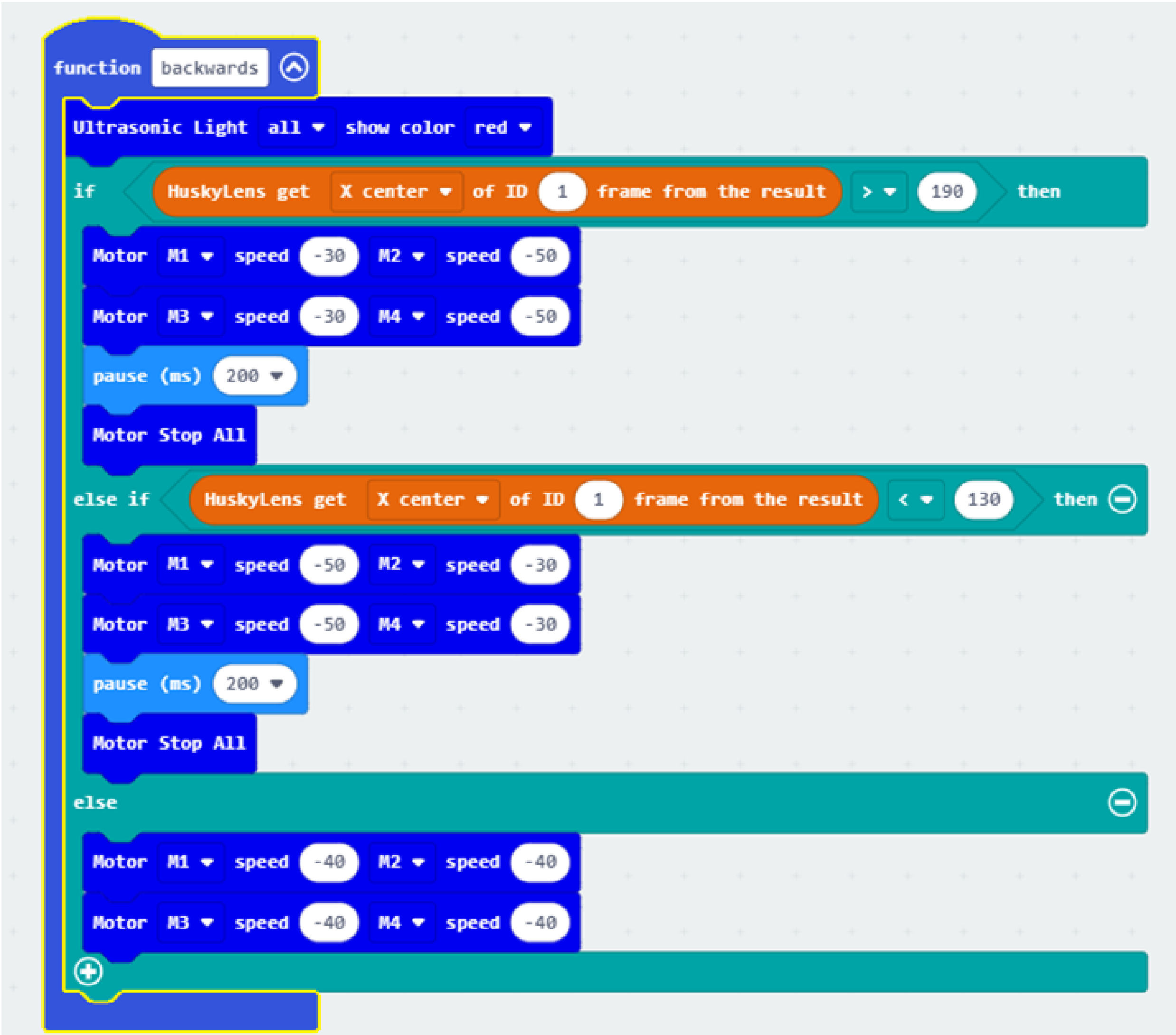
Unlimited execution of the program



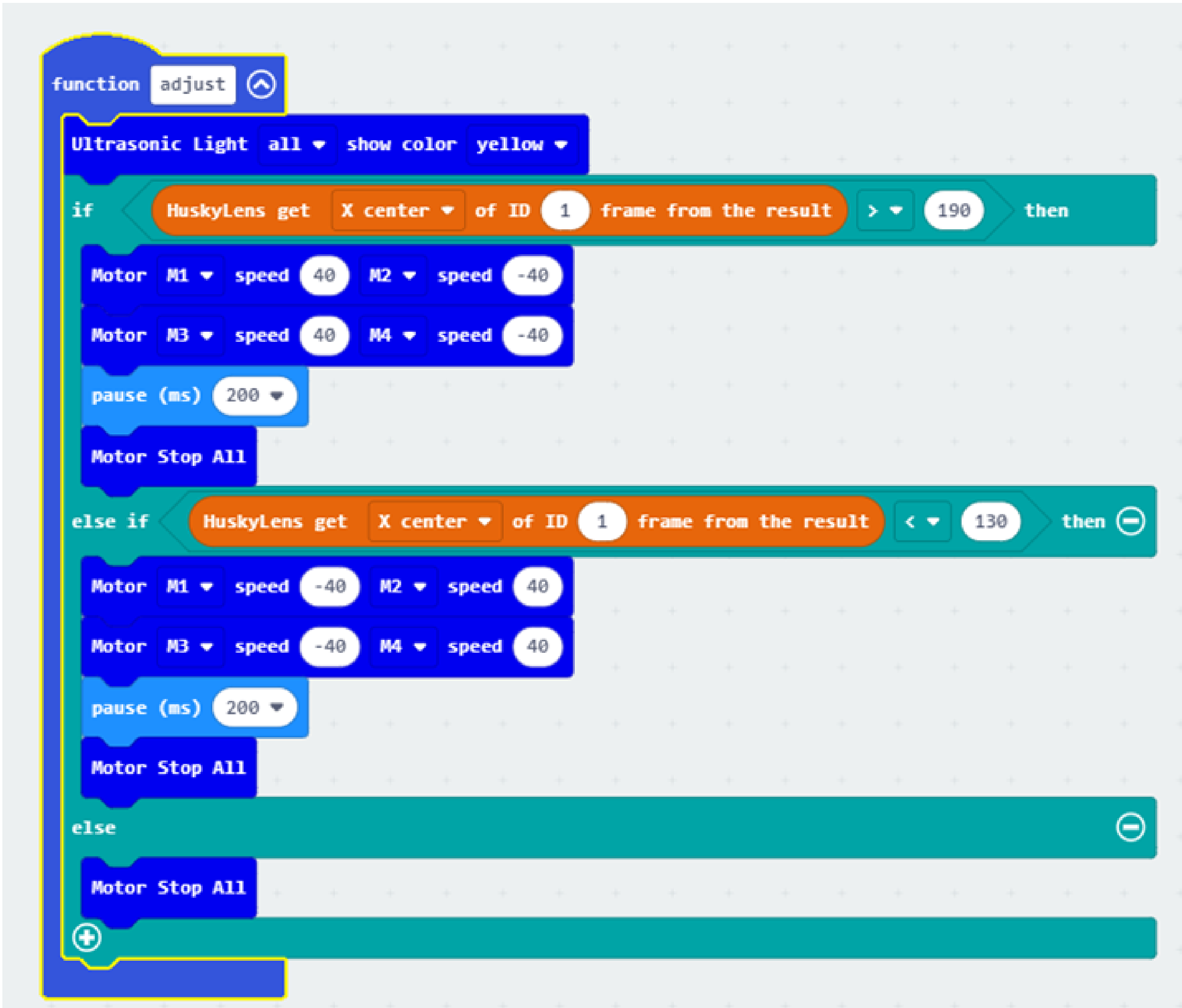
forward


function forwards
Ultrasonic Light all show color green
if HuskyLens get X center of ID 1 frame from the result > 190 then
Motor M1 speed 50 M2 speed 30
Motor M3 speed 50 M4 speed 30
pause (ms) 200
Motor Stop All
else if HuskyLens get X center of ID 1 frame from the result < 130 then
Motor M1 speed 30 M2 speed 50
Motor M3 speed 30 M4 speed 50
pause (ms) 200
Motor Stop All
else
Motor M1 speed 40 M2 speed 40
Motor M3 speed 40 M4 speed 40

backward


function backwards
Ultrasonic Light all show color red
if HuskyLens get X center of ID 1 frame from the result > 190 then
Motor M1 speed -30 M2 speed -50
Motor M3 speed -30 M4 speed -50
pause (ms) 200
Motor Stop All
else if HuskyLens get X center of ID 1 frame from the result < 130 then
Motor M1 speed -50 M2 speed -30
Motor M3 speed -50 M4 speed -30
pause (ms) 200
Motor Stop All
else
Motor M1 speed -40 M2 speed -40
Motor M3 speed -40 M4 speed -40

Move in Place



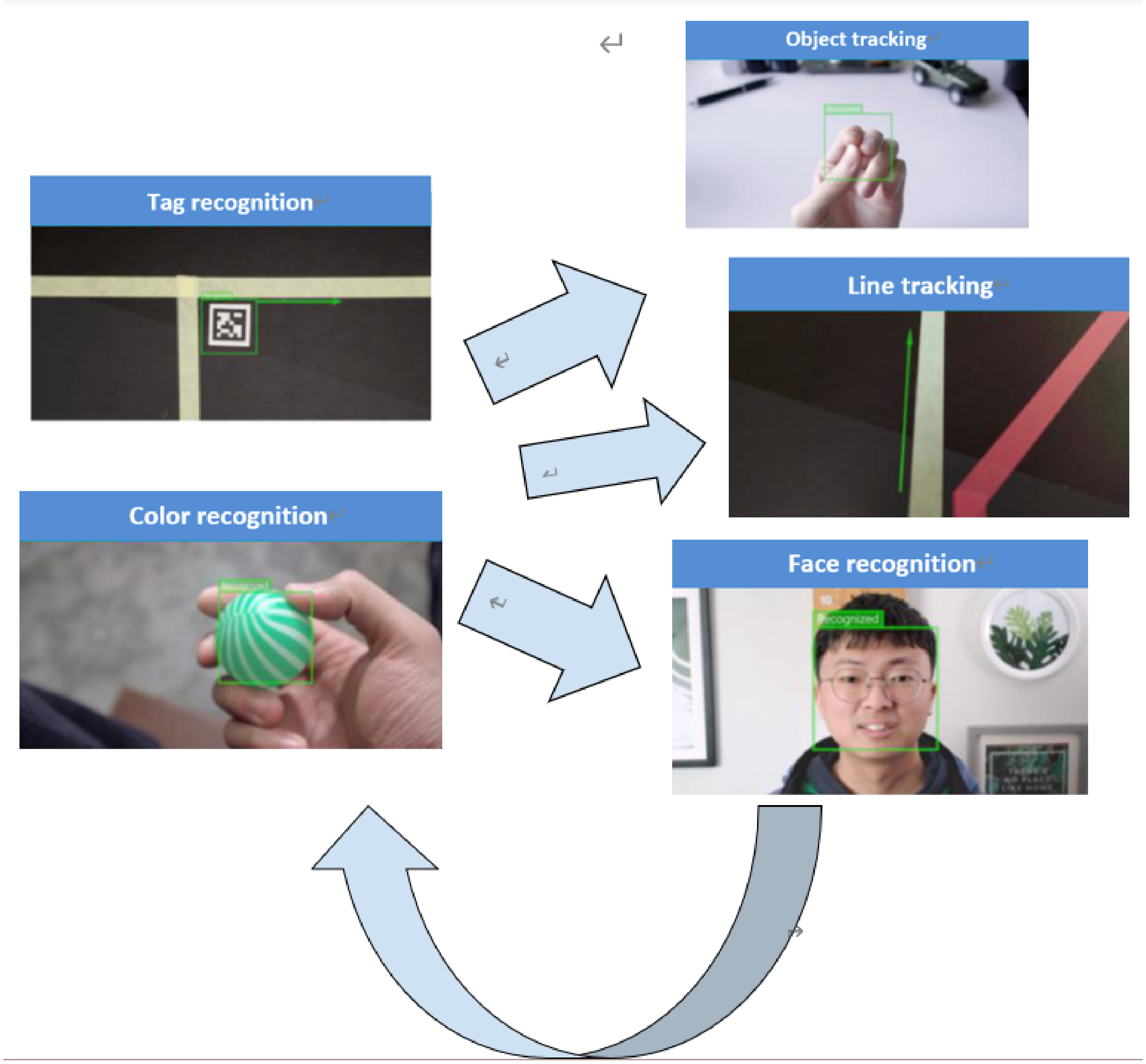
1.5.9 Lesson 9

Introduction

Objective

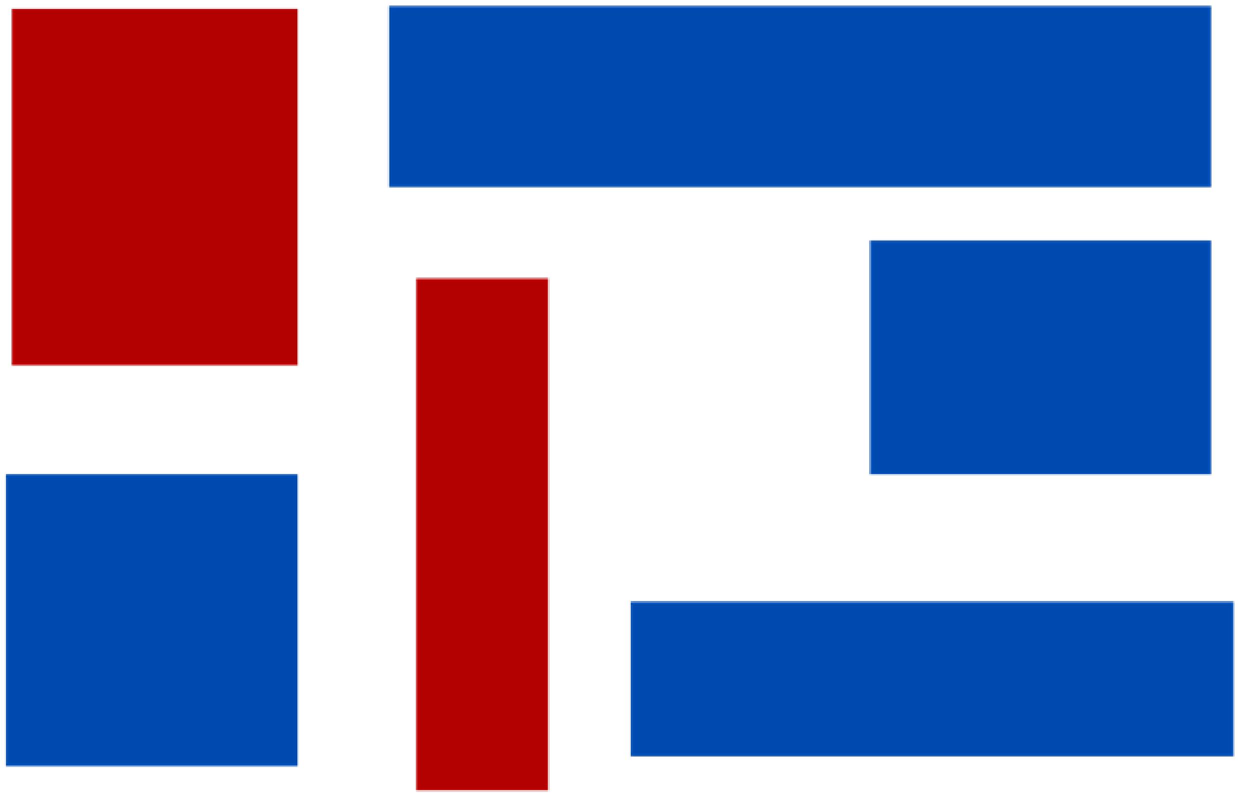
Micro:bit AI Smart Car

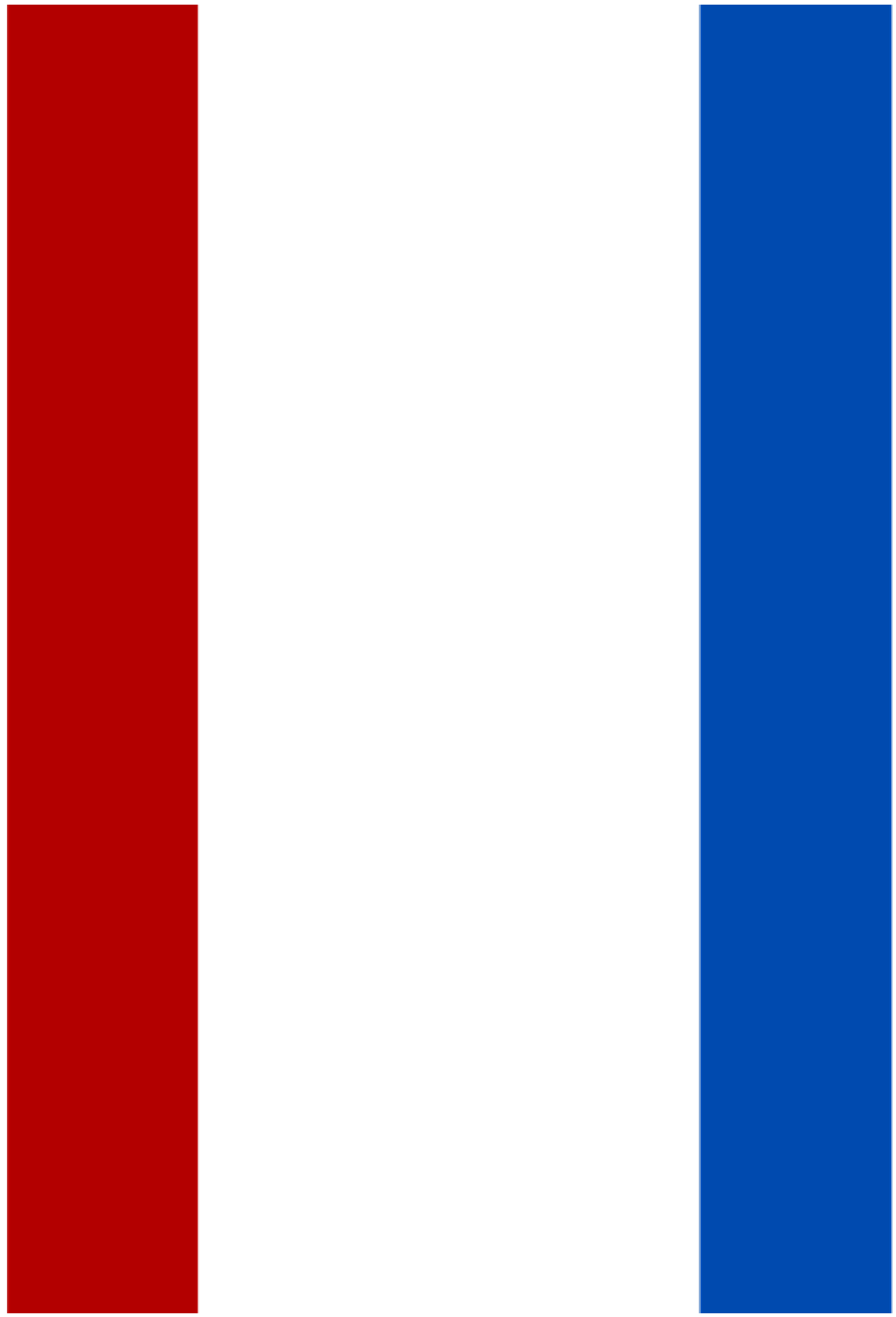
HuskyLens Advanced Applications



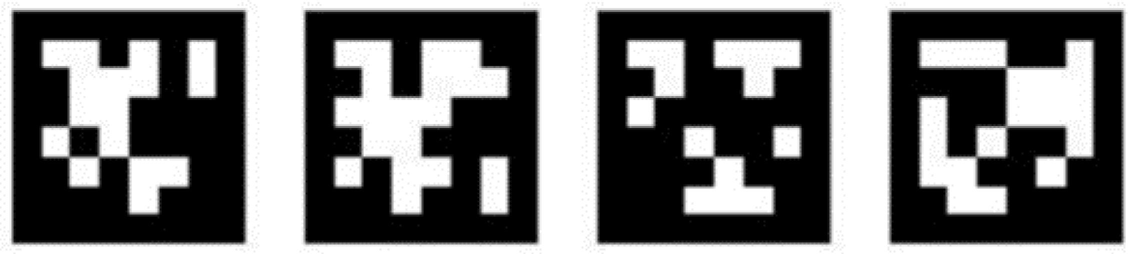
Exercise 1Function combination: color recognition + line tracking

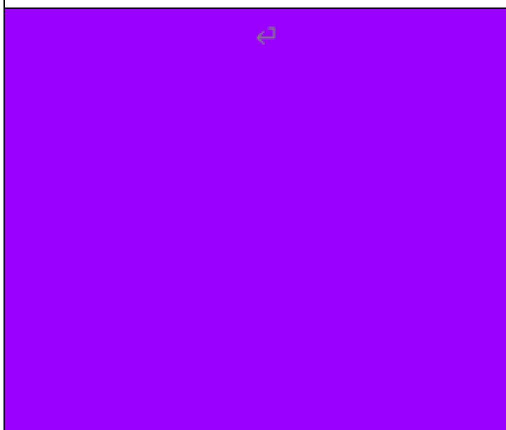
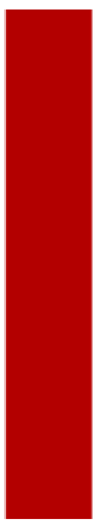
- If the tracking line is easy to go out of bounds, you can use the pause blocks to make the car brake after each movement.
- After pressing the A button to switch to color recognition mode, send two requests for data storage results.
- Initialize the data for the first time and update the data to the new learning result for the second time
- Create a variable id to store the color ID with the highest number of squares.
- Create variable mode to save the current mode of HuskyLens.



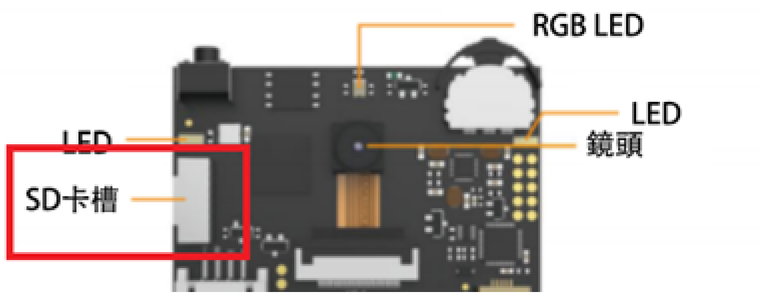


Exercise 2Use different labels to represent different HuskyLens modes

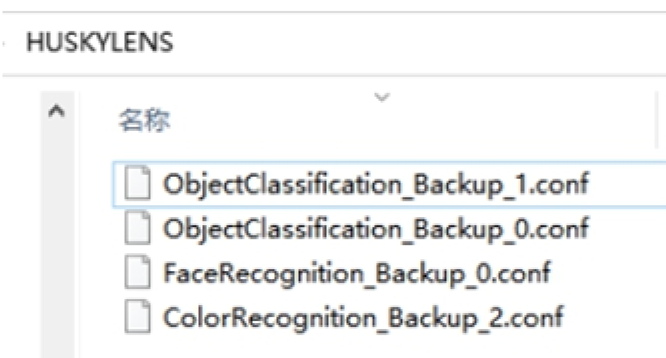


Color	Face
	
Line	Object
	

SD card save/load model



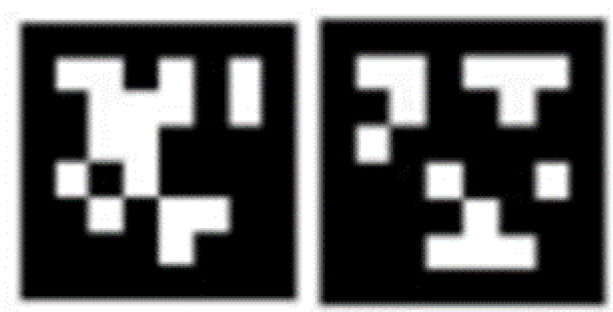
Method 1 : Manual operation in the second-level menu of each algorithm function



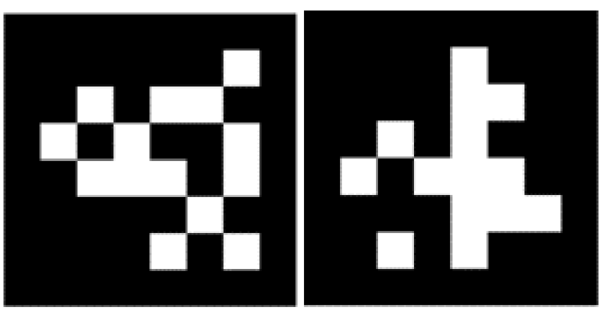
Method two : use the block module operation

Take photos, take screenshots and save to SD card

Exercise 3

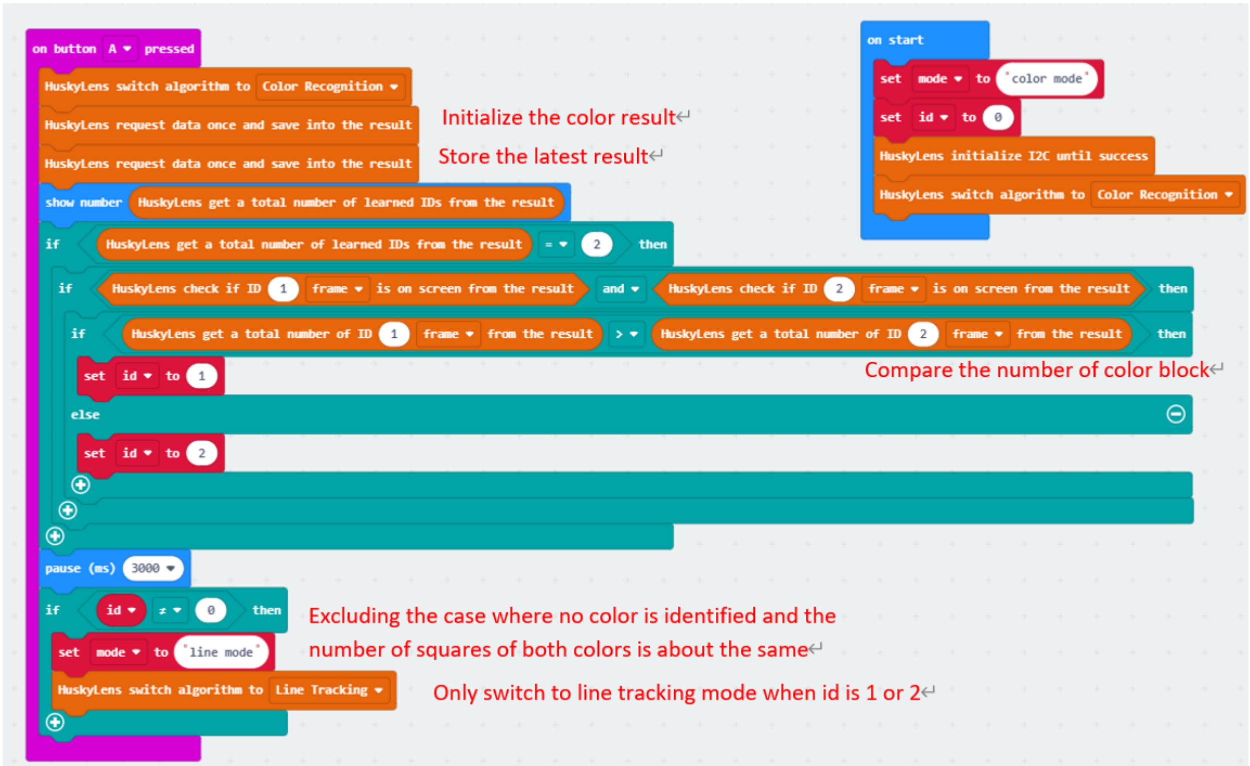


Exercise 4Sort the objects first, then switch to the corresponding mode + screenshot.



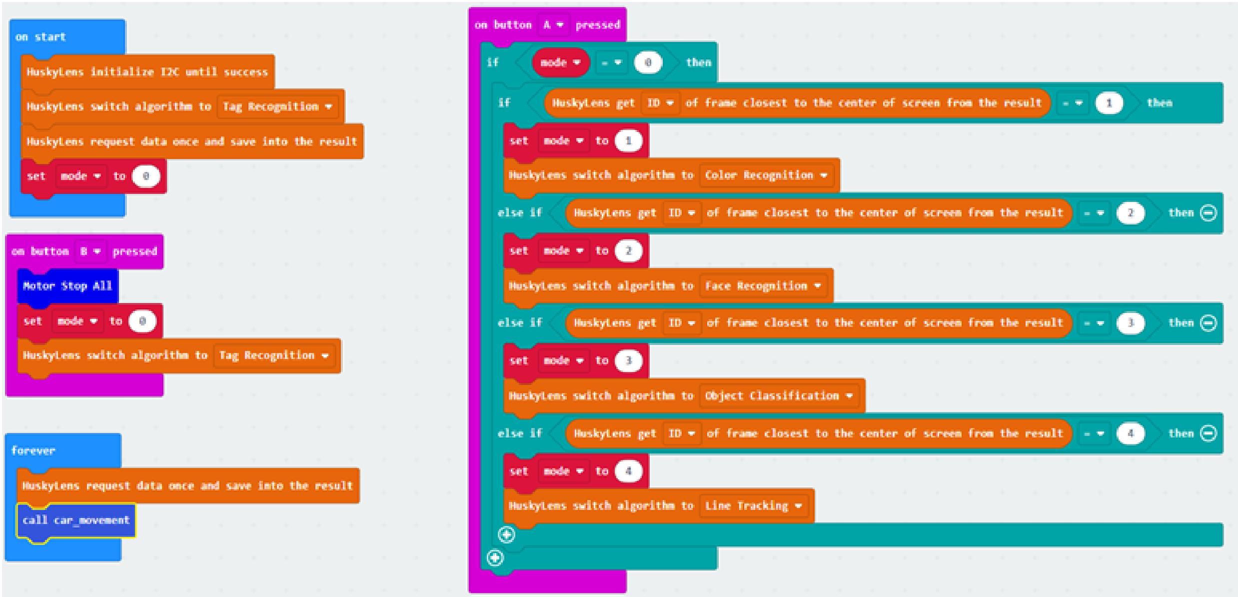
Answer

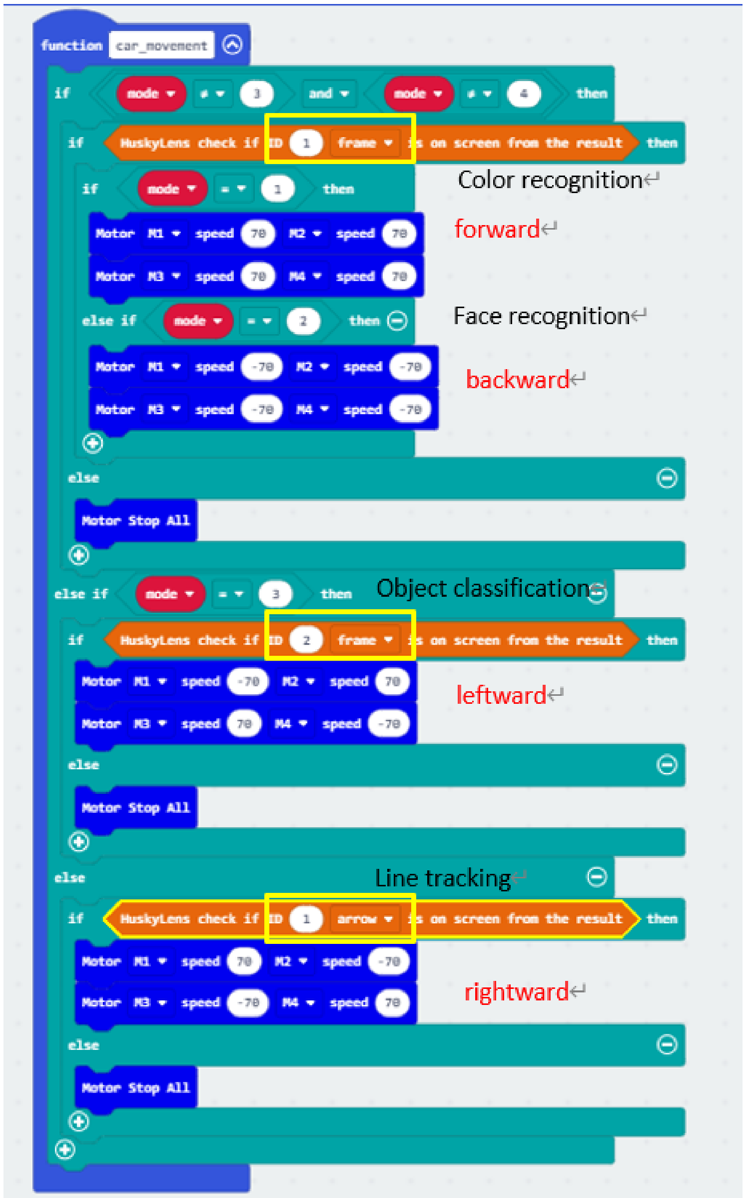
Exercise 1


on button A pressed
HuskyLens switch algorithm to Color Recognition
HuskyLens request data once and save into the result
Initialize the color result
HuskyLens request data once and save into the result
Store the latest result
show number HuskyLens get a total number of learned IDs from the result
if HuskyLens get a total number of learned IDs from the result = 2 then
if HuskyLens check if ID 1 frame is on screen from the result and HuskyLens check if ID 2 frame is on screen from the result then
if HuskyLens get a total number of ID 1 frame from the result > HuskyLens get a total number of ID 2 frame from the result then
set id to 1
Compare the number of color block
else
set id to 2
pause (ms) 3000
if id ≠ 0 then
set mode to "line mode"
HuskyLens switch algorithm to Line Tracking
Excluding the case where no color is identified and the number of squares of both colors is about the same
Only switch to line tracking mode when id is 1 or 2
on start
set mode to "color mode"
set id to 0
HuskyLens initialize I2C until success
HuskyLens switch algorithm to Color Recognition

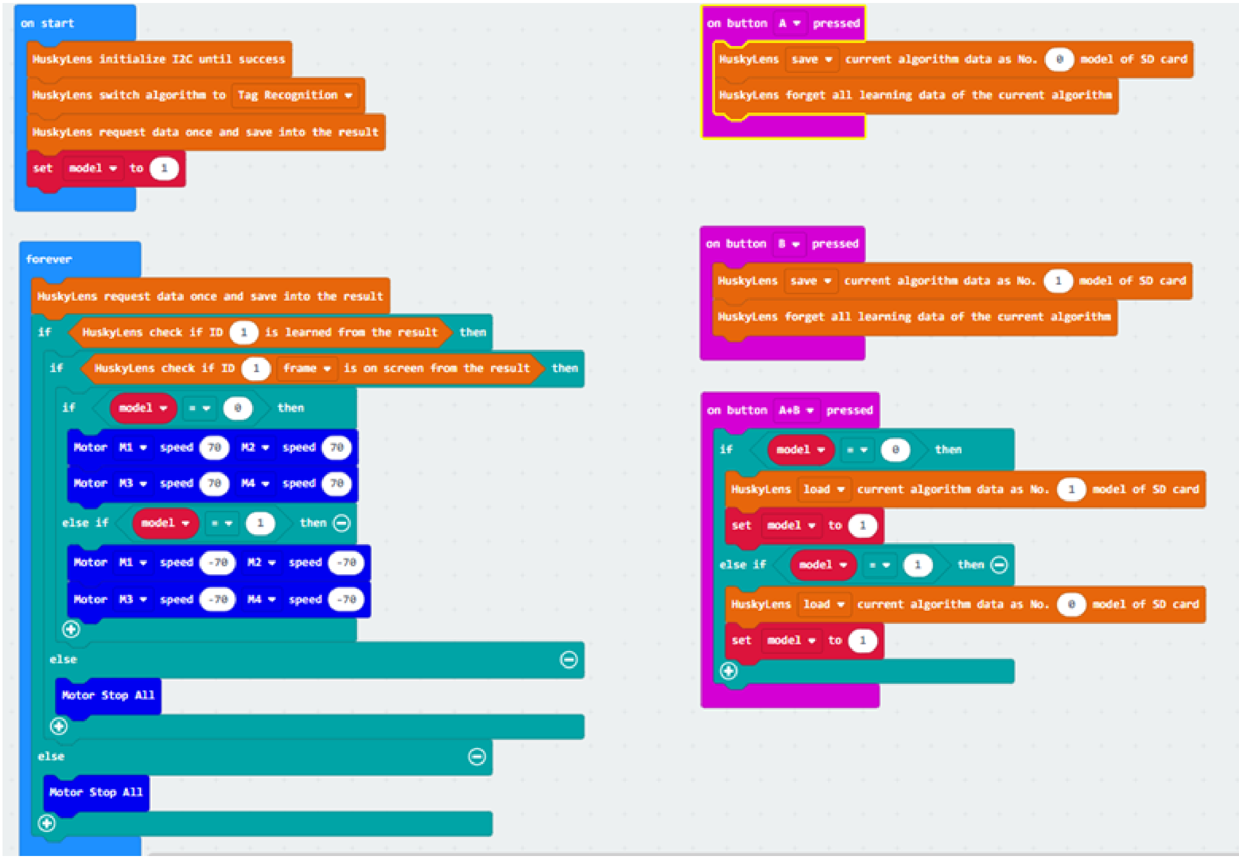

forever
if mode = "line mode" then
Execute this function only in line tracking mode
HuskyLens request data once and save into the result
if HuskyLens get Y endpoint of the No. id arrow from the result < 120 then
if HuskyLens get X beginning of the No. id arrow from the result < 130 then
Motor M1 speed 40 M2 speed 60
Motor M3 speed -40 M4 speed 60
else if HuskyLens get X beginning of the No. id arrow from the result > 190 then
Motor M1 speed 60 M2 speed 40
Motor M3 speed 60 M4 speed -40
else
Motor M1 speed 40 M4 speed 40
Motor M3 speed 40 M4 speed 40
else
Motor Stop All

Exercise 2

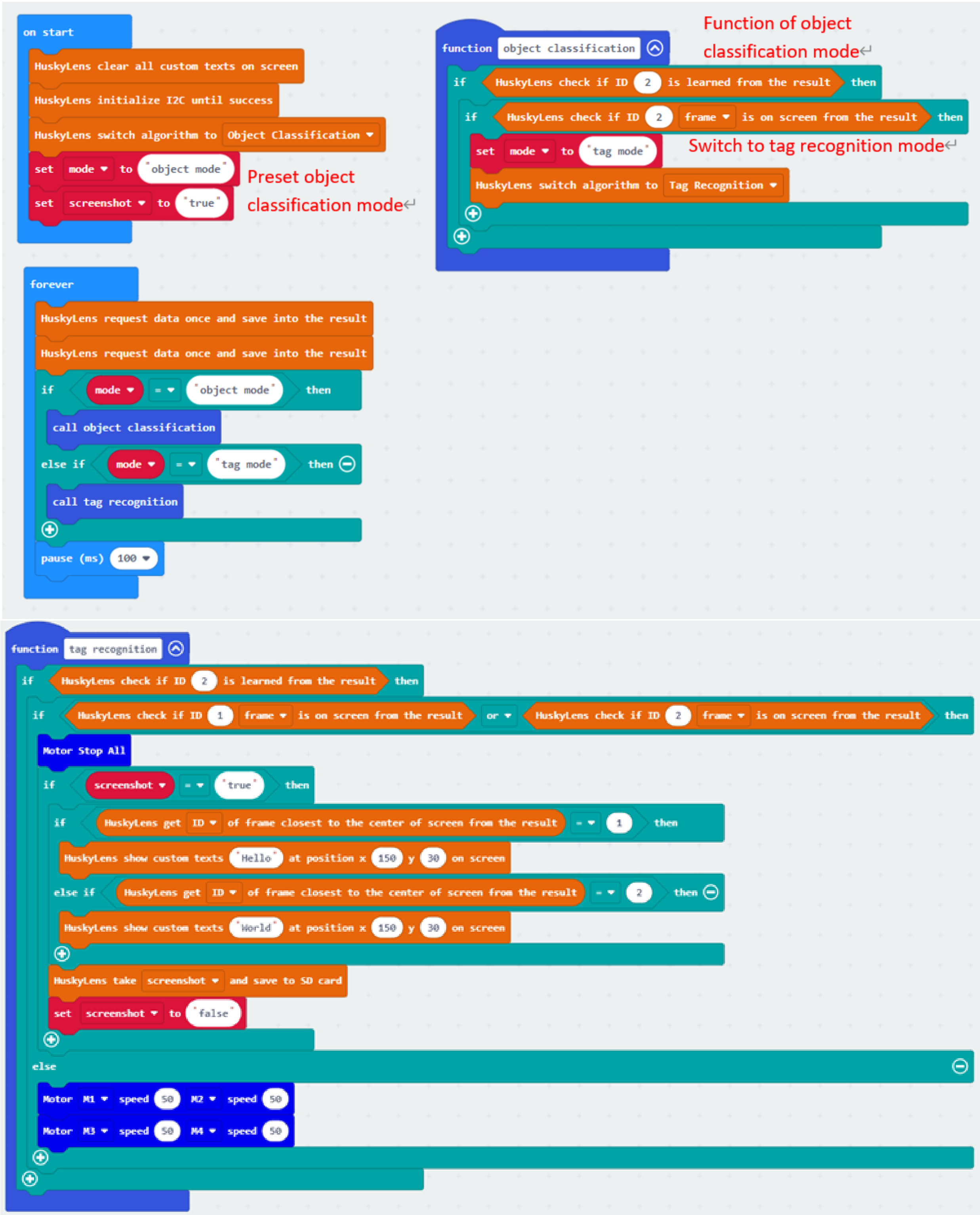

on start
HuskyLens initialize I2C until success
HuskyLens switch algorithm to Tag Recognition
HuskyLens request data once and save into the result
set mode to 0
on button B pressed
Motor Stop All
set mode to 0
HuskyLens switch algorithm to Tag Recognition
forever
HuskyLens request data once and save into the result
call car_movement
on button A pressed
if mode = 0 then
if HuskyLens get ID of frame closest to the center of screen from the result = 1 then
set mode to 1
HuskyLens switch algorithm to Color Recognition
else if HuskyLens get ID of frame closest to the center of screen from the result = 2 then
set mode to 2
HuskyLens switch algorithm to Face Recognition
else if HuskyLens get ID of frame closest to the center of screen from the result = 3 then
set mode to 3
HuskyLens switch algorithm to Object Classification
else if HuskyLens get ID of frame closest to the center of screen from the result = 4 then
set mode to 4
HuskyLens switch algorithm to Line Tracking


function car_movement
if mode ≠ 3 and mode ≠ 4 then
if HuskyLens check if ID 1 frame is on screen from the result then
Color recognition
if mode = 1 then
Motor M1 speed 70 M2 speed 70
Motor M3 speed 70 M4 speed 70
forward
Face recognition
else if mode = 2 then
Motor M1 speed -70 M2 speed -70
Motor M3 speed -70 M4 speed -70
backward
else
Motor Stop All
else if mode = 3 then
Object classification
if HuskyLens check if ID 2 frame is on screen from the result then
Motor M1 speed -70 M2 speed 70
Motor M3 speed 70 M4 speed -70
leftward
else
Motor Stop All
else
Line tracking
if HuskyLens check if ID 1 arrow is on screen from the result then
Motor M1 speed 70 M2 speed -70
Motor M3 speed -70 M4 speed 70
rightward
else
Motor Stop All

Exercise 3


on start
HuskyLens initialize I2C until success
HuskyLens switch algorithm to Tag Recognition
HuskyLens request data once and save into the result
set model to 1
forever
HuskyLens request data once and save into the result
if HuskyLens check if ID 1 is learned from the result then
if HuskyLens check if ID 1 frame is on screen from the result then
if model = 0 then
Motor M1 speed 70 M2 speed 70
Motor M3 speed 70 M4 speed 70
else if model = 1 then
Motor M1 speed -70 M2 speed -70
Motor M3 speed -70 M4 speed -70
else
Motor Stop All
else
Motor Stop All
on button A pressed
HuskyLens save current algorithm data as No. 0 model of SD card
HuskyLens forget all learning data of the current algorithm
on button B pressed
HuskyLens save current algorithm data as No. 1 model of SD card
HuskyLens forget all learning data of the current algorithm
on button A+B pressed
if model = 0 then
HuskyLens load current algorithm data as No. 1 model of SD card
set model to 1
else if model = 1 then
HuskyLens load current algorithm data as No. 0 model of SD card
set model to 1

Exercise 4



on start
HuskyLens clear all custom texts on screen
HuskyLens initialize I2C until success
HuskyLens switch algorithm to Object Classification
set mode to "object mode"
set screenshot to "true"
Preset object classification mode
function object classification
Function of object classification mode
if HuskyLens check if ID 2 is learned from the result then
if HuskyLens check if ID 2 frame is on screen from the result then
set mode to "tag mode"
Switch to tag recognition mode
HuskyLens switch algorithm to Tag Recognition
forever
HuskyLens request data once and save into the result
HuskyLens request data once and save into the result
if mode = "object mode" then
call object classification
else if mode = "tag mode" then
call tag recognition
pause (ms) 100
function tag recognition
if HuskyLens check if ID 2 is learned from the result then
if HuskyLens check if ID 1 frame is on screen from the result or HuskyLens check if ID 2 frame is on screen from the result then
Motor Stop All
if screenshot = "true" then
if HuskyLens get ID of frame closest to the center of screen from the result = 1 then
HuskyLens show custom texts "Hello" at position x 150 y 30 on screen
else if HuskyLens get ID of frame closest to the center of screen from the result = 2 then
HuskyLens show custom texts "World" at position x 150 y 30 on screen
HuskyLens take screenshot and save to SD card
set screenshot to "false"
else
Motor M1 speed 50 M2 speed 50
Motor M3 speed 50 M4 speed 50

1.5.10 Lesson 10

Introduction

Objective

Conclusion Micro:bit AI Smart Cart

Competition 1

- Face: micro:bit board showing smile symbols
- Color: Lights up one RGB on-board light or ultrasonic light (six in total, each a different color)

Rules

```
1.        The competition is a score system, there are two rounds, the highest score␣
↪of the round will be the final score
2.        when the car successfully identify the face, +5 points
3.        Whenever the car successfully identifies a color, +2 points
4.        If all RGB on-board lights and ultrasound lights are successfully lit, +6␣
↪point
5.        If the car does not go to the end point within the time limit, the score␣
↪will be reduced by half
```

Competition 2

Rules

```
1.        The competition is a score system, there are two rounds, the highest score␣
↪of the round will be the final score
2.        Whenever the car successfully identifies a tag, +2 points
3.        Each time a single word in the HuskyLens screen is successfully linked into␣
↪a phrase, an additional +5 points will be awarded.
4.        If the car does not go to the end point within the time limit, the score␣
↪will be reduced by half. The fastest group to complete will +6 points, the second␣
↪group will +4 points, the third group will +2 points.
```