NASHENBOT

Nashenbot Training

--- MPBOT S1/S2

Introduction

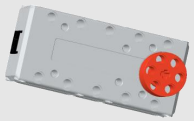
Mpbot S1&S2: Designed for beginners and educators, this compact package MPBOT S1 includes the core MPBOT S robot chassis with pre-installed Micro-brain firmware. The drag-and-drop Micro-brain interface enables instant programming via physical buttons and onboard sensors, eliminating the need for external devices. Perfect for classroom STEM projects and introductory robotics courses, it teaches fundamental concepts like loop control and obstacle avoidance through interactive challenges. The kit comes with modular expansion ports for future upgrades, bridging block-based learning to advanced coding seamlessly.

1. Core parts parameter

Core parts parameter		
Part	MPBOTS1	MPBOTS2
MPC controller	1	1
Motor	2	2
Servo motor	1	1
Infrared Sensor	1	
Sound Sensor	1	
Integrated 5-Grayscales Sensor	1	1
Full Color Light	2	1
Open-loop motor	1	1
Color sensor	1	1

1. Core parts parameter

Controller



Closed-loop motor

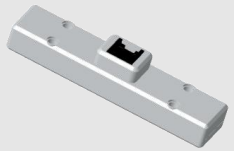
Open-loop motor

Servo motor

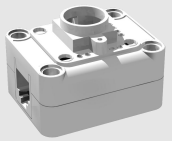
Full color light

Infrared sensor

Sound sensor

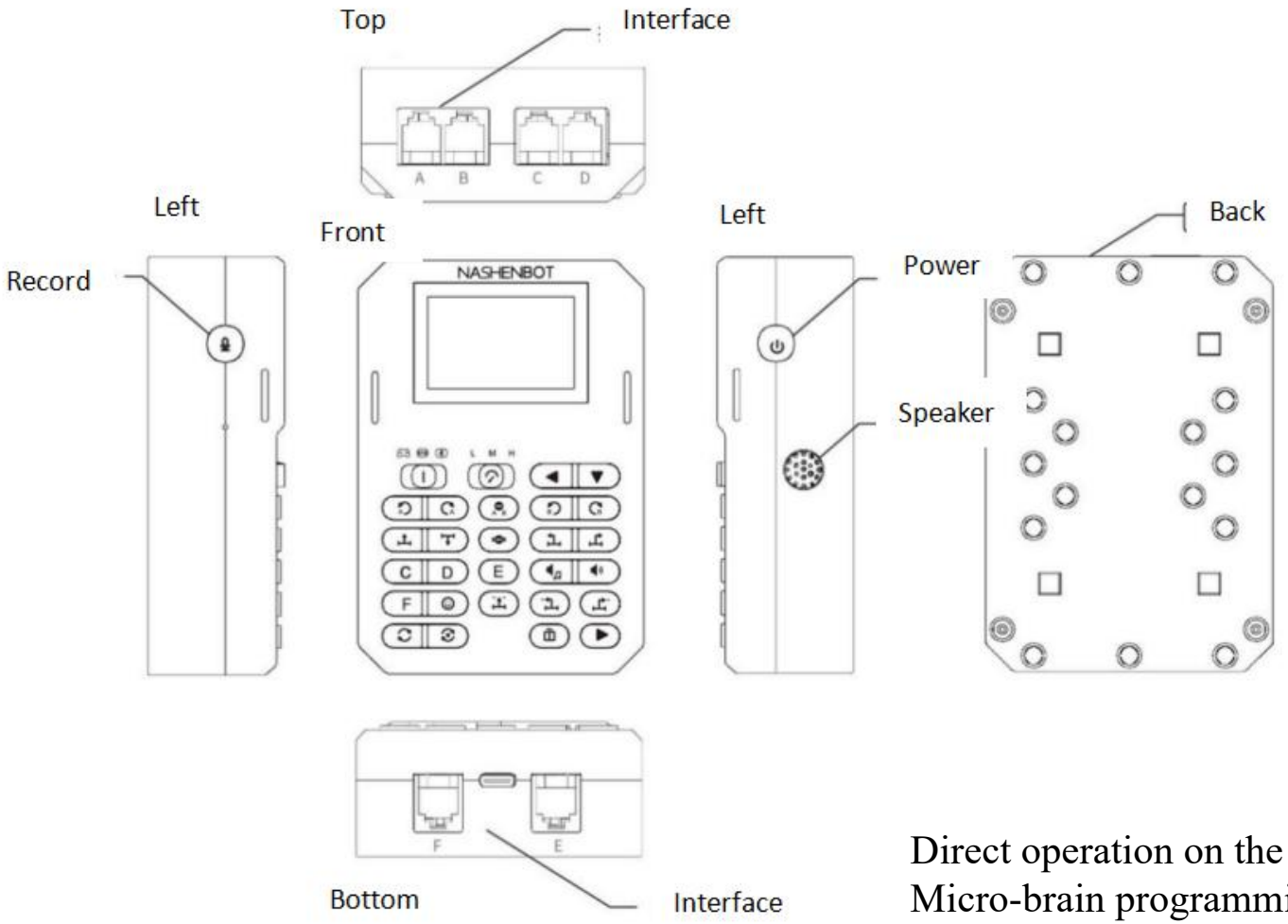


Integrated 5-Grayscales



Color sensor

2.1 MPC controller

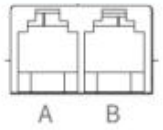
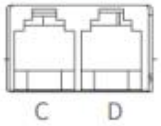


Direct operation on the main control unit is called micro-brain programming. Micro-brain programming is divided into real-time control mode, key-screen programming normal mode, key-screen programming advanced mode, and also supports programming on a computer and then downloading the code.

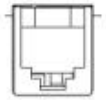
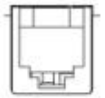
2.1 MPC controller

NASHENBOT

2.1 Top

Icon		
Name	Motor interface	Sensor interface
Parameter	RJ11, 6P6C Support control motor	RJ11. 6P6C Support Infrared obstacle avoidance sensor、Acoustic intensity detection, Magnetic sensitivity detection, Impact switch, Ultrasonic testing, Photosensitive detection、Five-Grayscale sensor and other sensor input control; Support Illumination, Electric fan, Steering engine and other output control

2.2 Bottom

Icon			5V
Name	Sensor interface (common for input and output)	Sensor interface (common for input and output)	Charging interface
Parameter	RJ11, 6P6C; Support Infrared obstacle avoidance sensor, Acoustic intensity detection, Magnetic sensitivity detection, Impact switch, Ultrasonic testing, Photosensitive detection, Five-Grayscale sensor and other sensor input control; Support Illumination, Electric fan, Steering engine and other output control	RJ11, 6P6C; Support Infrared obstacle avoidance sensor, Acoustic intensity detection, Magnetic sensitivity detection, Impact switch, Ultrasonic testing, Photosensitive detection, Five-Grayscale sensor and other sensor input control; Support Illumination, Electric fan, Steering engine and other output control	USB Type C interface; Support 5V 1A charging

2.1 MPC controller

NASHENBOT

2.3 Left

Icon		
Name	Sound recording	
Parameter	Short press to start recording, Only valid in  real-time control mode	

2.4 Right

Icon		
Name	Power switch	Trumpet
Parameter	Press 1 second long to turn on or off	Play Sound

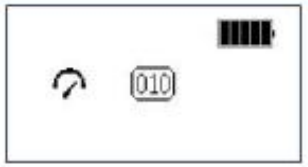
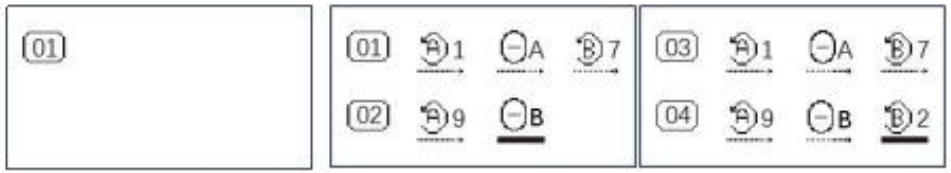
2.2 Micro-brain programming mode

After powering on, you need to select the programming mode first.

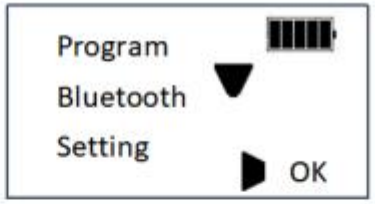
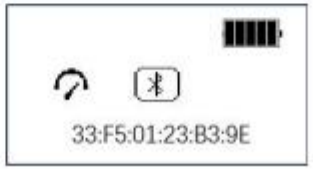
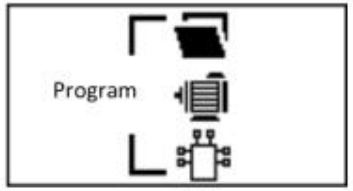
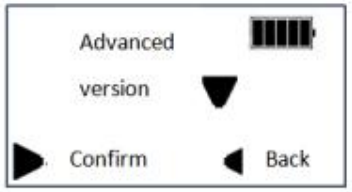
Mode is single choice :

Real-time control	The above keys can take effect; program Displays the currently pressed key in the program area.

2.2 Micro-brain programming mode

010 Key-screen programming	 The first level interface of this mode will automatically enter the second level interface (i.e. key screen programming status) after one second of display; All buttons on the front can take effect; Examples of procedures are as follows:  01 The number here represents the number of program lines, and the program capacity of different lines is supported according to the robot model; 1 The number here represents the program's parameters (in seconds or times); A The letters here represent the program's parameters (ports); B The horizontal bar here indicates that the program is selected (you can delete it or insert a new program backward).

2.2 Micro-brain programming mode

Advanced Function	 This is the homepage of advanced functions, which is divided into three modes: program, Bluetooth connection and settings.
	This is the Bluetooth connection mode, which supports the interconnection of two MPC controllers (pay attention to the official account of Nashen Robot at the end of the article for detailed usage). 
	This is the program mode. This mode is divided into three functions: program, motor and port. The program function can directly execute the program downloaded from the computer. The motor function can view the speed and code values of the two motor ports. The port can view the sensors connected to each sensor port and control the sensor execution commands. 
	This is the setting mode. There is advanced mode and version information in this mode. Advanced mode can be turned on or not. The version displays the current firmware version information. 

Electric quantity is relative value: 0~100% corresponding voltage is 3.5~4.2V±0.2V.
When the battery is low, the buzzer will give a "tick tick" low power alarm prompt, and will automatically stop the action of the motor and other program modules, and will automatically shut down after 5 seconds. At this time, it is necessary to use the standard charging line of the product for compliance charging (5V1A).
During charging, the charging status is red.

2.2 Bluetooh mode

Bluetooh mode: how to connect between two controllers?

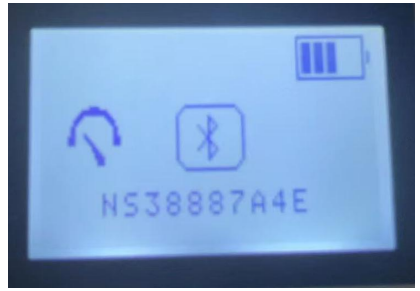
Before connection

After connection

No.1 controller: Remotely controlled controller

No.2 controller: as the remote control

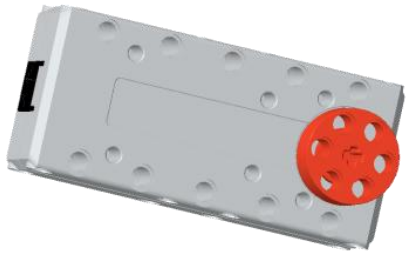
it needs to press for twice

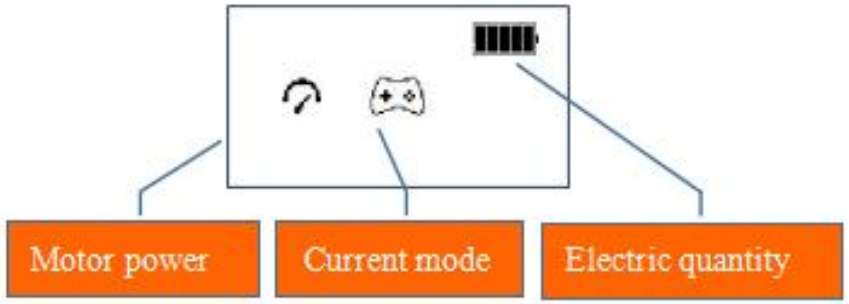


Before connection

After connection

2.3 Power switch





The motor power is a global command, which is divided into three types: high speed, medium speed and low speed.This can be used to control the rotation power of A and B motors. At the beginning of debugging, low speed is usually used to facilitate the observation of running conditions.

Power switch

Three types of power are supported:

1. High speed
2. Medium speed
3. Low speed

All three modes are valid

2.4 Button operation

◀	▼
Insert character/Return	Flip/Move
Support the insertion of program module in 010 key-screen programming mode	Support page down in 010 key-screen programming mode
Support return in advanced function mode	Mobile support in advanced function mode

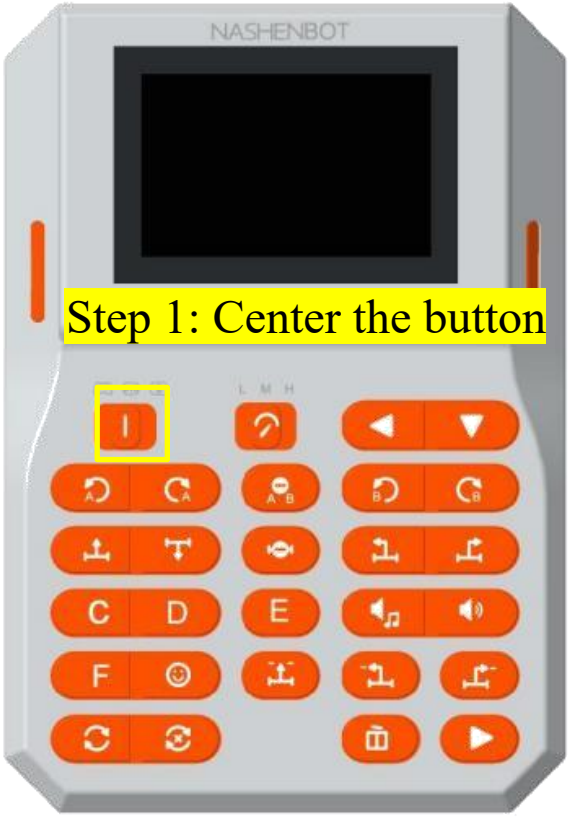
In the key screen programming mode, clicking “**Run**” will execute the completed programming statements, but the program will not automatically exit when it finishes running. If you want to change the program, you need to clic ▶ **ɔp**” again before making changes.

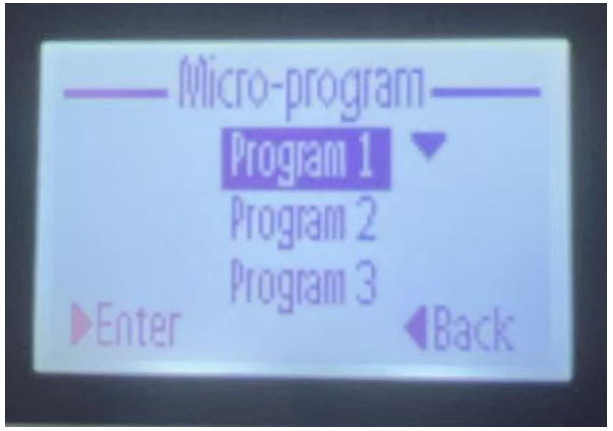
2.4 Button operation

Remove program	Run program/Define
Short press to delete the currently selected program	Short press to run or stop the program in the 010 key-screen programming mode
ning mode	In advanced function mode, the function is determined

In the key screen programming mode, clicking  **Run"** will execute the completed programming statements, but the program will not automatically exit when it finishes running. If you want to change the program, you need to click  "**Stop"** again before making changes.

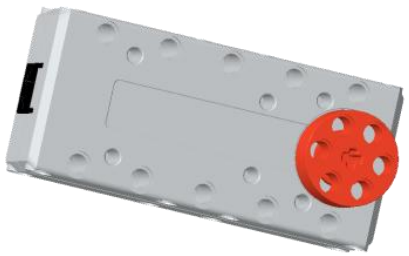
2.5 Key-screen programming





There are ten programs in total. First, select the mode. You can use the down arrow key to select the corresponding program to run. The last run program will be saved automatically, but programs that have been written but not run will not be saved.

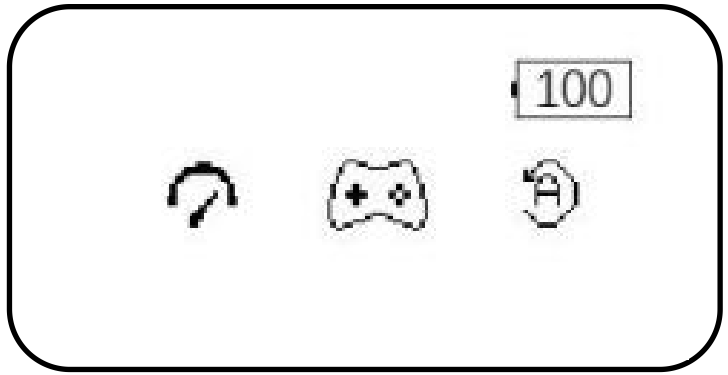
2.6 Single motor control programming



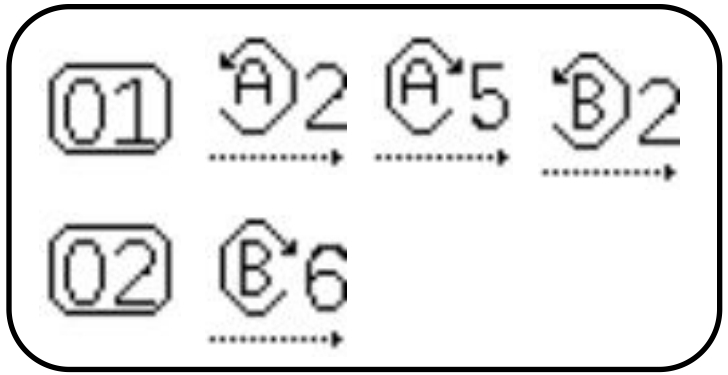
There are two types of individual motors. The buttons here mainly correspond to closed-loop motors, which are also coded motors. They have higher speeds and more precise movements, and are mainly connected to ports A and B. Pay attention to the one-to-one correspondence of the motor control interface. Generally, infinite motion is used in conjunction with sensors.

Icon	A	A	A B	B	B
Name	A Motor reverse	A motor forward rotation	A or B motor stop	B Motor reverse	B motor forward rotation
Parameter	Control the reverse rotation of motor A	Control the forward rotation of motor A	Control A or B motor to stop rotating	Control the reverse rotation of motor B	Control the forward rotation of motor B
	010 In normal mode of key-screen programming, short press can set 1~9 seconds and unlimited time, and long press can set 0.1~9.9 seconds in advanced mode		010 Key-screen programming Normal mode short press to switch A or B motor, advanced mode long press to set the stop time	010 In normal mode of key-screen programming, short press can set 1~9 seconds and unlimited time, and long press can set 0.1~9.9 seconds in advanced mode	
	In real-time control mode, long press will continue to execute			In real-time control mode, long press will continue to execute	

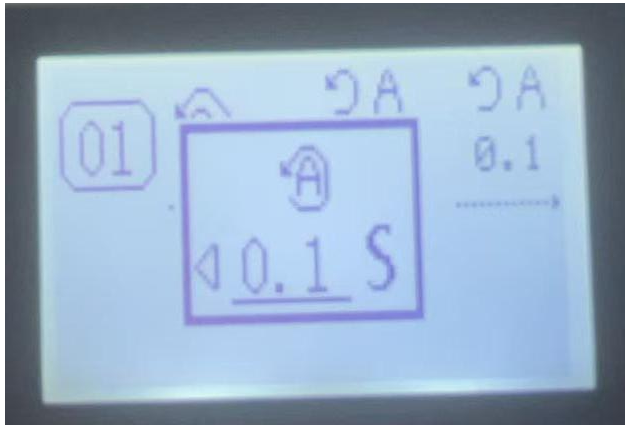
2.6 Single motor control programming example



Press to turn, release to stop turning

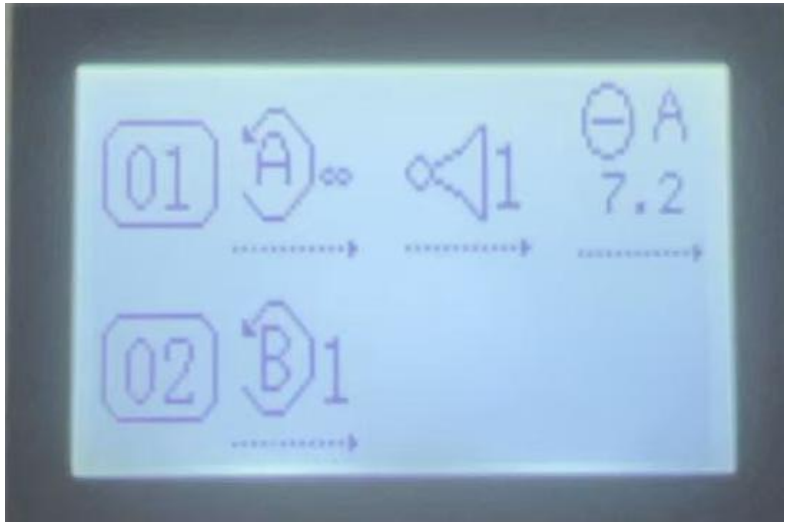


The two motors rotate alternately



In advanced mode, a long press allows for more precise time settings. The left arrow key increases the time by 0.1 seconds per press, while a long press increases the time rapidly. The time cannot be decreased. If the time setting is too long, it can be deleted and rewritten.

2.6 Single motor control programming example

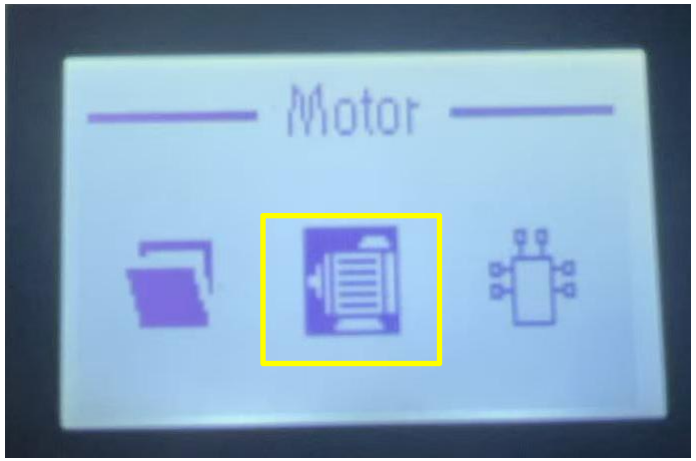


A short press allows you to choose whether motor A or motor B stops immediately, with no waiting time. A long press allows you to select the motor to stop and also set the stop wait time.

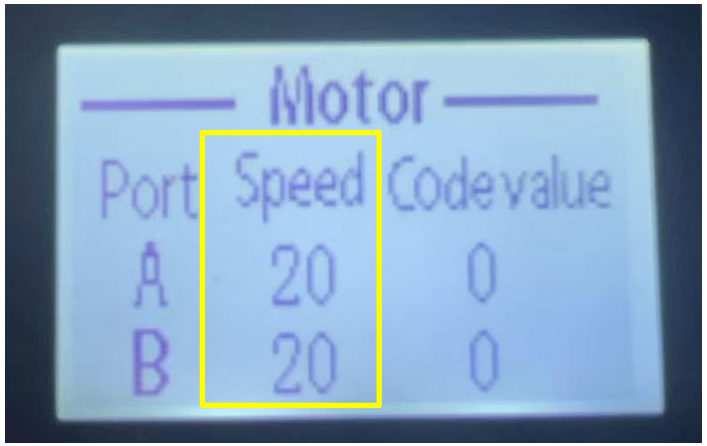
2.6 Single motor control programming example

Closed-loop encoder motors can be directly tested for speed adjustment.

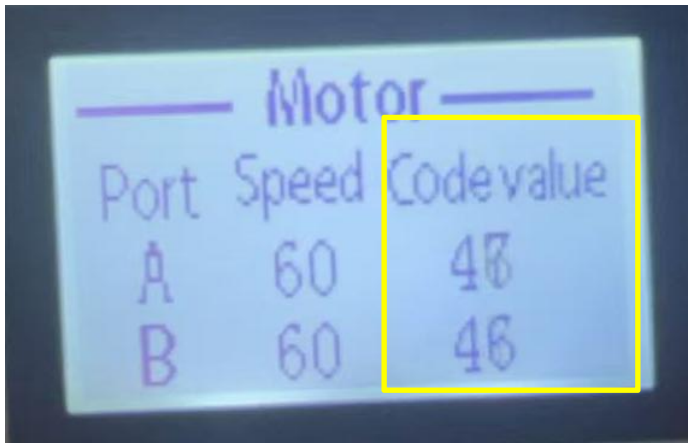
Step 2：



Step 3：



Step 4：



Step 5：

The page down button changes the speed. The closer the code values are, the stronger the consistency between the two motors, making it suitable for building structures like cars that require coordinated movement.

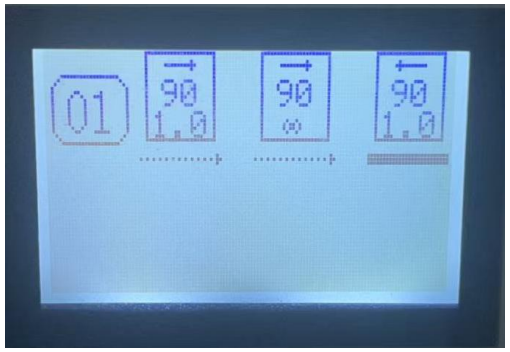
2.6 Single motor control programming

Icon	C	D	E	F
Name	C interface sensor detection	D interface sensor detection	E interface sensor detection	F interface sensor detection
Parameter	Automatically identify 9 sensors such as infrared obstacle avoidance or sound intensity detection for logic judgment; Only valid in 010 key screen programming mode			Automatically identify 9 sensors such as infrared obstacle avoidance or sound intensity detection for logic judgment; Only valid in 010 key screen programming mode

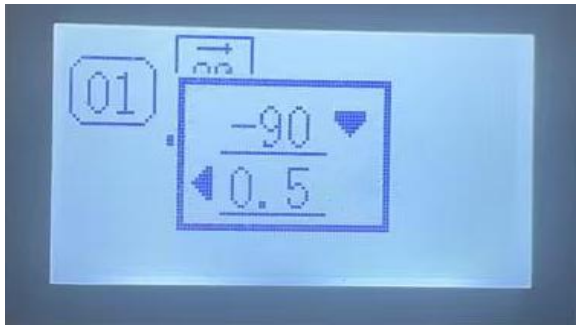
There are two types of individual motors. Note that the open-loop motor and the servo motor look the same, but their working principles are different. The open-loop motor mainly drives the motor to rotate, while the servo motor automatically stops after rotating to a fixed angle. They can be distinguished by the English markings on the casing. When assembling, pay close attention to the location of the English markings on the casing in the diagram to avoid incorrect assembly that could lead to different results with the same programming, causing confusion for students.

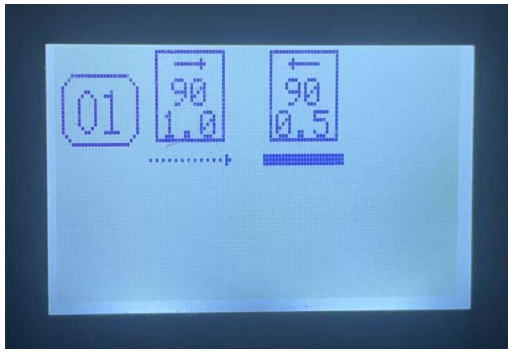
2.6 Single motor control programming

1. Although open loop motor is also a motor, it cannot move in real-time control mode.



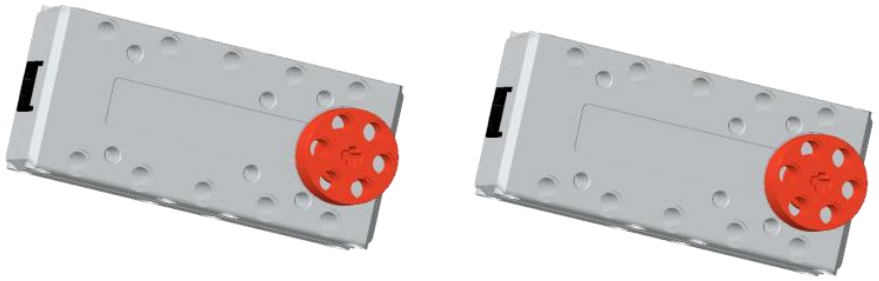
2. The open-loop motor rotates clockwise by default. Clicking ten times will keep it rotating clockwise, and clicking eleven times will start it rotating counterclockwise.





3. In advanced mode, a long press allows you to set a more precise time. Use the left-hand arrow key to increase the time. A short press increases the time by 0.1 seconds each time, while a long press increases it quickly. You cannot decrease the time. If the time setting is too long, you can delete it and rewrite it. The down arrow key can set the direction. A positive speed indicates forward rotation, while a negative speed indicates reverse rotation.

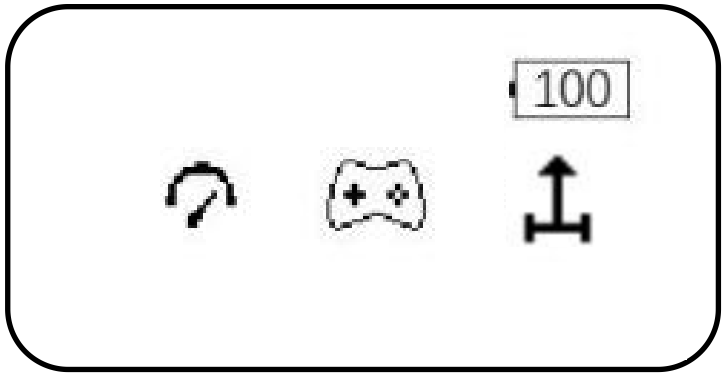
2.7 Dual motors control programming



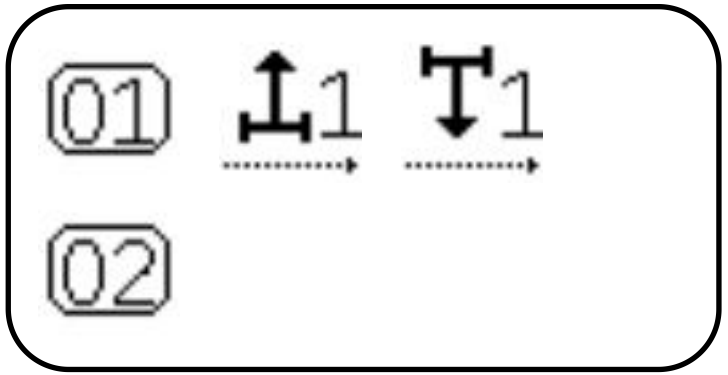
The dual motors are sometimes used separately, but more often they are used to control the car's shape and movement. This can vary depending on the build and wiring. If the forward button causes the car to move backward, swap the motor connectors or change the direction of the car's front.

Icon					
Name	Robot forward	Robot back	Robot stop	Robot turns left	Robot turns right
Parameter	At the same time, control the reverse rotation of motor A and the forward rotation of motor B	At the same time, control forward rotation of motor A and reverse rotation of motor B	At the same time, control A and B motors to stop	At the same time, control the forward rotation of motor A and motor B	At the same time, control the reverse rotation of motor A and motor B
	010 In normal mode of key-screen programming, short press can set 1~9 seconds and unlimited time, and long press can set 0.1~9.9 seconds in advanced mode		010 Key-screen programming Normal mode short press to switch A or B motor, advanced mode long press to set the stop time	010 In normal mode of key-screen programming, short press can set 1~9 seconds and unlimited time, and long press can set 0.1~9.9 seconds in advanced mode	
	In real-time control mode, long press will continue to execute			In real-time control mode, long press will continue to execute	

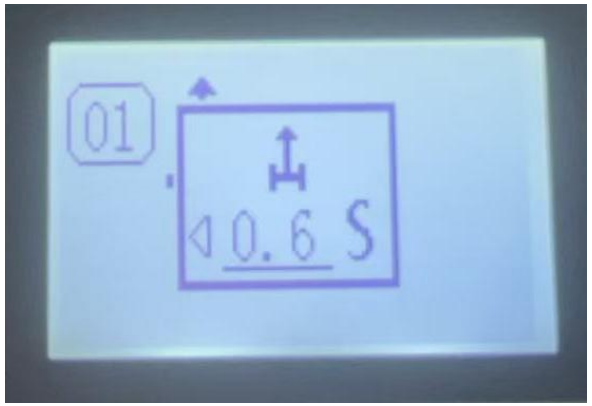
2.7 Dual motors control programming example



Press to go forward, release to stop

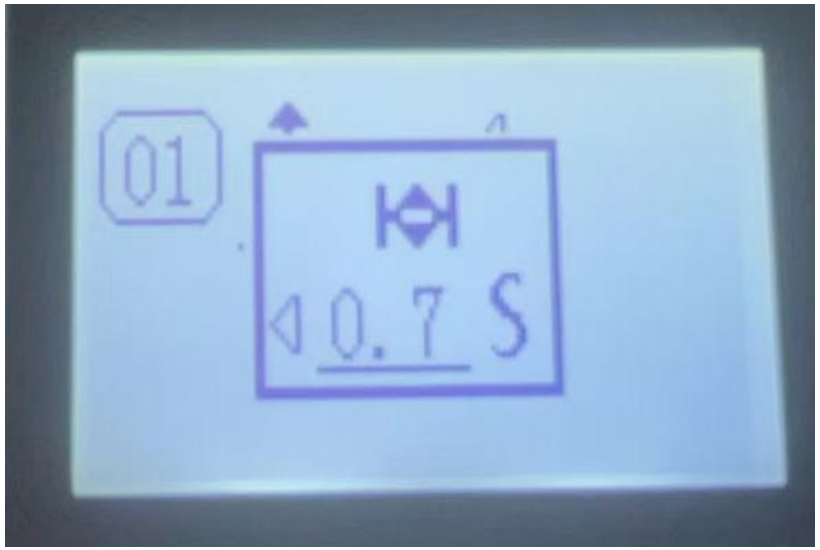


Move forward one second and then stop



In advanced mode, a long press allows for more precise time settings. The left arrow key increases the time by 0.1 seconds per press, while a long press increases the time rapidly. The time cannot be decreased. If the time setting is too long, it can be deleted and rewritten.

2.8 Dual motor control programming example

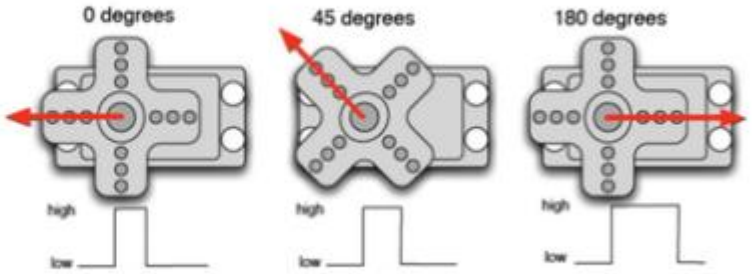


A short press will immediately stop both motors without any waiting time. A long press allows you to set the stop waiting time.

2.9 Servo motor control programming example

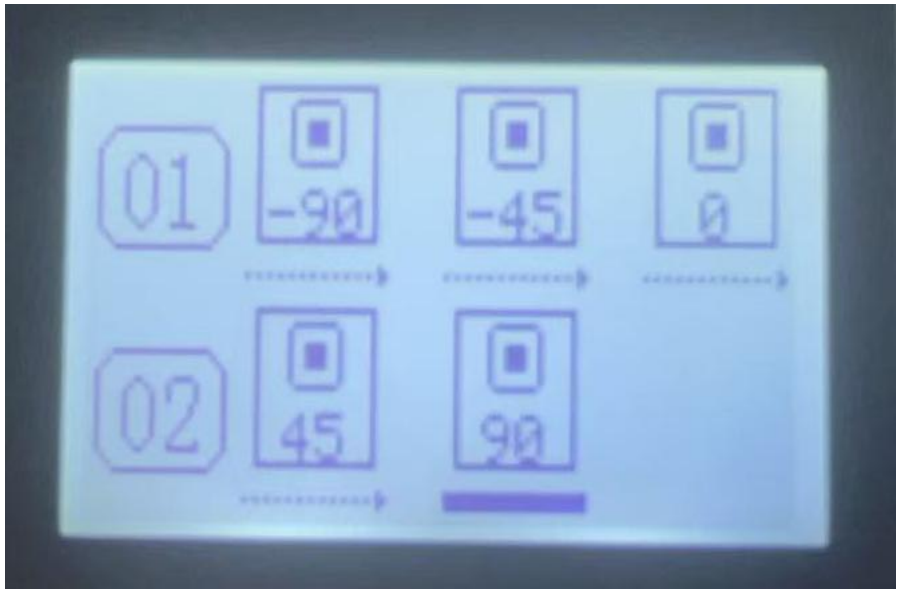
Icon	C	D	E
Name	C interface sensor detection	D interface sensor detection	E interface sensor detection
Parameter	Automatically identify 9 sensors such as infrared obstacle avoidance or sound intensity detection for logic judgment; Only valid in 010 key screen programming mode		

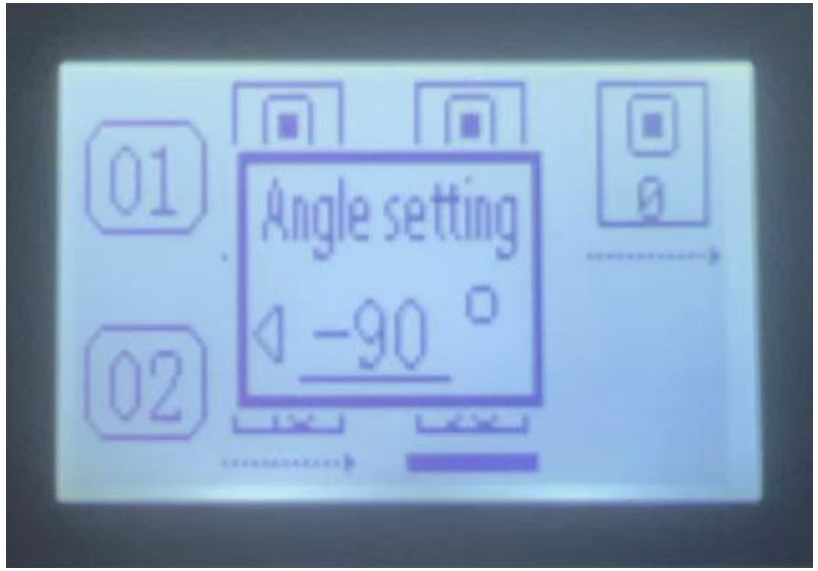
F
F interface sensor detection
Automatically identify 9 sensors such as infrared obstacle avoidance or sound intensity detection for logic judgment;
Only valid in 010 key screen programming mode



A servo motor is an electronic component that moves by controlling the rotation of its output shaft by a specified angle. Unlike a motor, it can only rotate within a certain angle range and cannot rotate a full circle. It does not have a fixed direction of rotation; its movement depends primarily on the relative position of the previous and next angles. For example, 0 to 90 degrees is clockwise, and 90 to -45 degrees is counterclockwise. Therefore, when assembling a servo motor, attention must be paid to the initial position. During installation, the angle is usually adjusted to a certain angle, such as zero degrees, and the program is run before installation.

2.9 Servo motor control programming example





Generally, short presses of 1-5 times correspond to -90, -45, 0, 45, and 90 respectively.
Long presses increment by one from -90, -89, -88, and so on.

2.10 Sensor programming

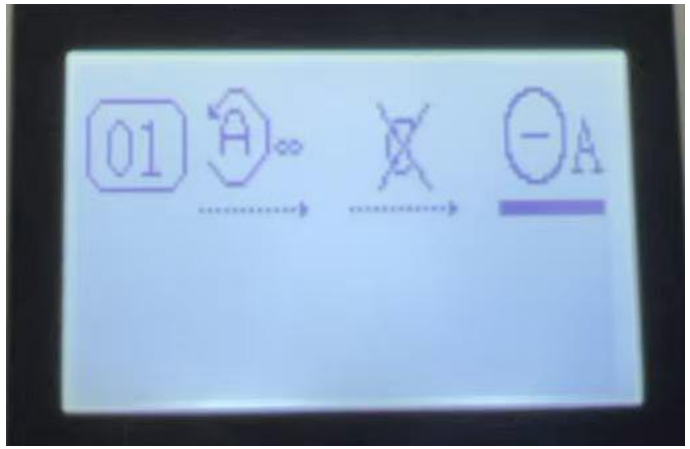
Icon	C	D	E
Name	C interface sensor detection	D interface sensor detection	E interface sensor detection
Parameter	Automatically identify 9 sensors such as infrared obstacle avoidance or sound intensity detection for logic judgment; Only valid in 010 key screen programming mode		

F
F interface sensor detection
Automatically identify 9 sensors such as infrared obstacle avoidance or sound intensity detection for logic judgment;
Only valid in 010 key screen programming mode

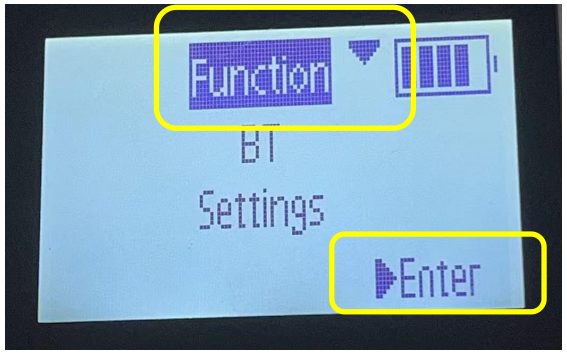
	**Infrared Sensor** Detect whether there are obstacles in front The working voltage is 5v, and the detection range is effective within 10cm (white surface reflection).
	**Sound Sensor** Detect whether there is sound in front The working voltage is 5v, and the detection range is 20~90dB.
	**Integrated 5-Grayscales Sensor** Detect whether there is trace under The working voltage is 5v and the detection height is 1.6cm.
	**Color Sensor** Can recognize 5 colors: black, red, green, blue, and yellow

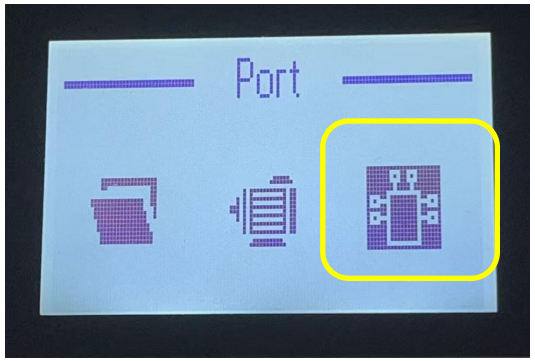
2.10 Sensor programming

Sensor identification is based on the wiring interface. The program will wait until the sensor is identified before continuing. If a sensor is marked with a cross, it could mean the wrong sensor was selected or the wiring is not securely connected. Placing the sensor trigger in different locations will produce different effects.



A cross will be displayed if there is a mismatch.

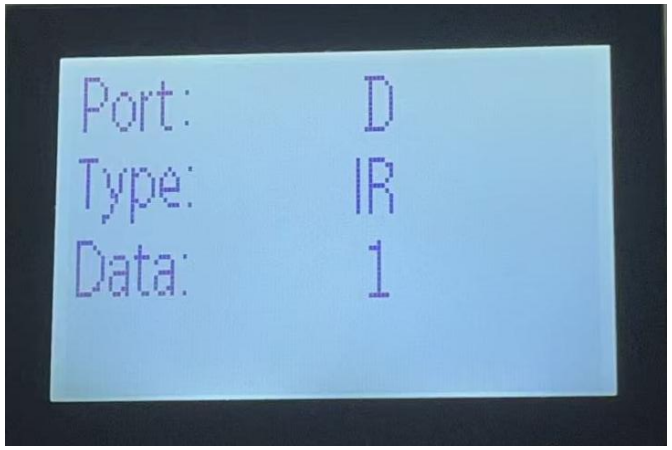




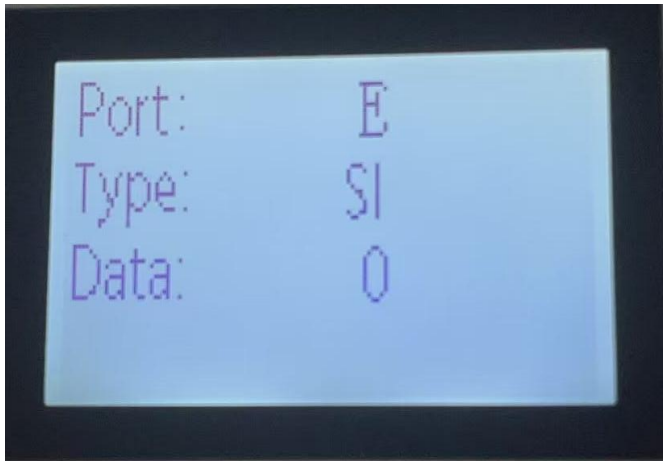
One way to check is to look directly at each wire, and another way is to check directly on the screen. First, push the mode selection to the right, select the first option, click run, scroll down two options, select port, and you can view the sensor and real-time values corresponding to each port. Start from port C, scroll down to port D, and so on.

2.10 Sensor programming

Color recognition requires close contact with the device. The recognized color is 0 for no operation or pure white, 1 for black, 2 for red, 3 for green, and 4 for yellow.



Infrared obstacle avoidance: 0 indicates no obstacle, 1 indicates an obstacle.

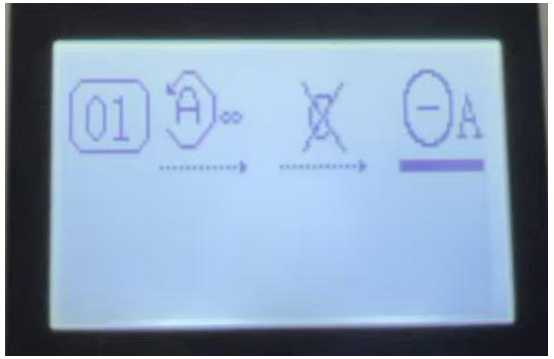


A sound intensity of 0 means there is no loud sound, while 1 means there is.

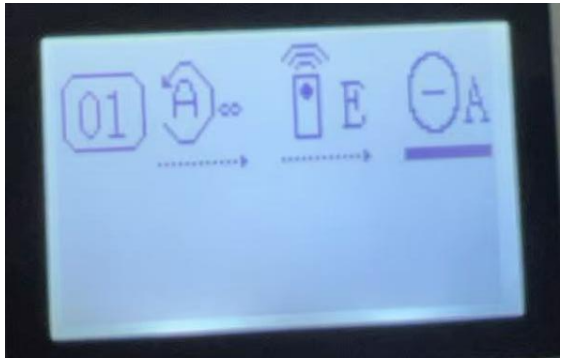
The integrated five-grayscale system assigns a value of 0 to light colors close to white and a value of 1 to dark colors close to black.

2.10 Sensor programming

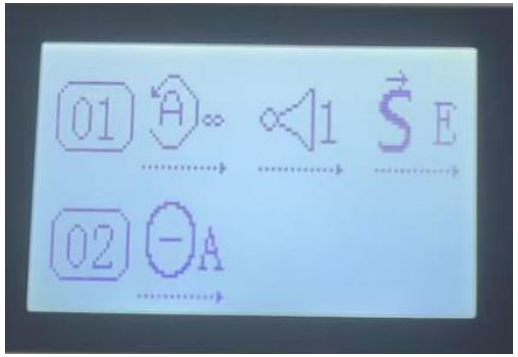
Sensor identification is based on the wiring interface. The program will wait until the sensor is identified before continuing. If a sensor is marked with a cross, it could mean the wrong sensor was selected or the wiring is not securely connected. Placing the sensor trigger in different locations will produce different effects.



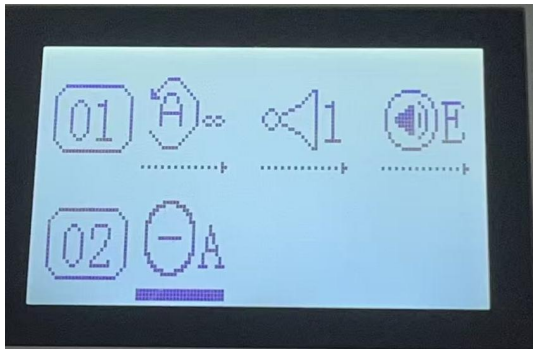
No sensor detected at the corresponding port



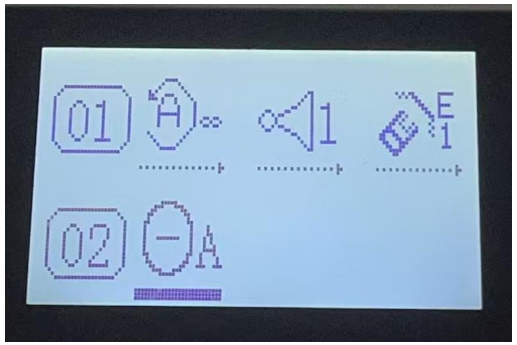
Infrared sensor



Integrated five-grayscale sensors are generally not used in this way; they are used in conjunction with motion line following.



Sound sensor



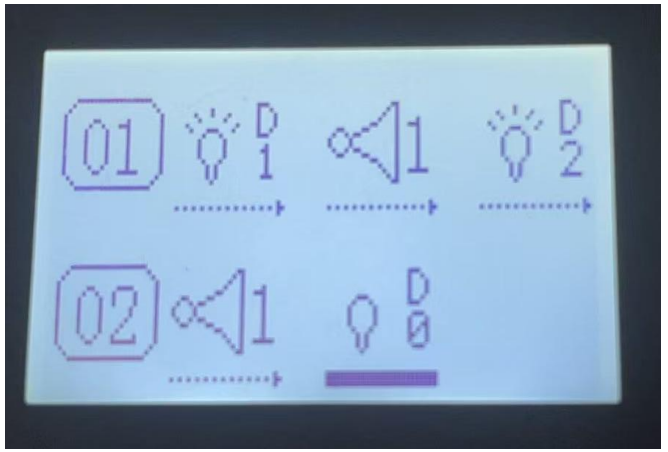
Color sensor

2.11 Lights

Icon	C	D	E
Name	C interface sensor detection	D interface sensor detection	E interface sensor detection
Parameter	Automatically identify 9 sensors such as infrared obstacle avoidance or sound intensity detection for logic judgment; Only valid in 010 key screen programming mode		

F
F interface sensor detection
Automatically identify 9 sensors such as infrared obstacle avoidance or sound intensity detection for logic judgment;
Only valid in 010 key screen programming mode

The light control is activated by clicking once (1), and subsequent clicks increase the value to 5, with the sixth click resetting the value to 0. These correspond to red, green, blue, white, orange, and off, respectively, cycling through the colors.

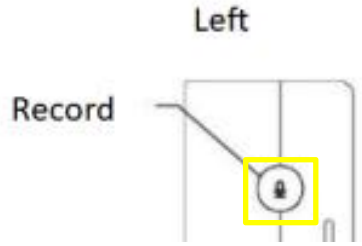


2.12 Sound playback

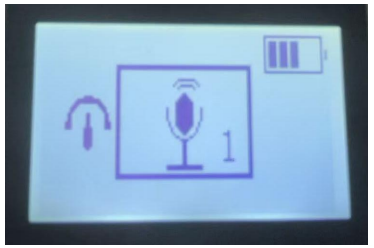
NASHENBOT

sound playback

010 In normal mode of key-screen programming: short press can play 8 audio and 1 recording; in advanced mode, long press can play 19 audio and 9 recording



Step 2
Long-press



When this message appears on the screen, it means recording is in progress. Release the button after you finish speaking, and the recording will continue. If the video automatically stops, it means the recording time is up, and you can start recording the next segment again. Normal mode has only one segment, while advanced mode has nine segments. Resuming recording will start from the first segment and overwrite the previous ones sequentially.

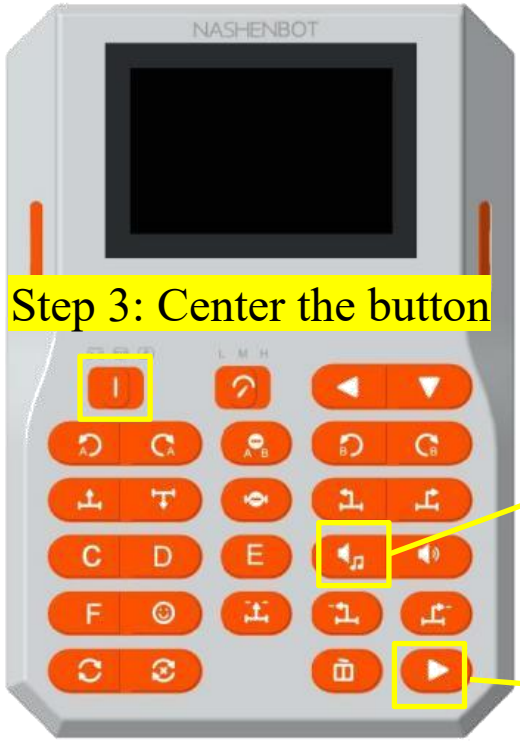
Playing recordings must be used in conjunction with recording. The recording function is only available in real-time mode; first, select real-time mode.

2.12 Sound playback

NASHENBOT

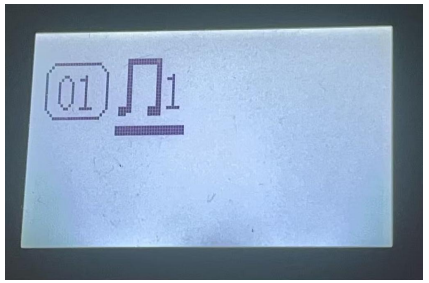
sound playback

010 In normal mode of key-screen programming: short press can play 8 audio and 1 recording; in advanced mode, long press can play 19 audio and 9 recording

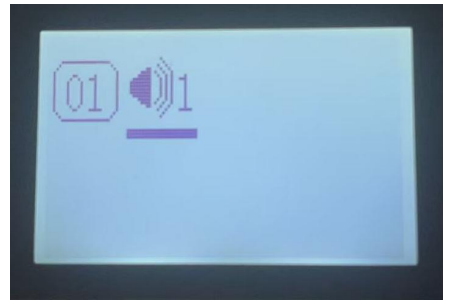


Step 3: Center the button

Step 4 Select program

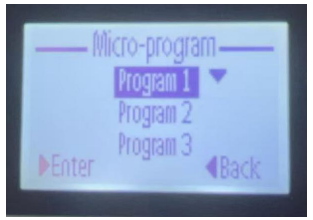


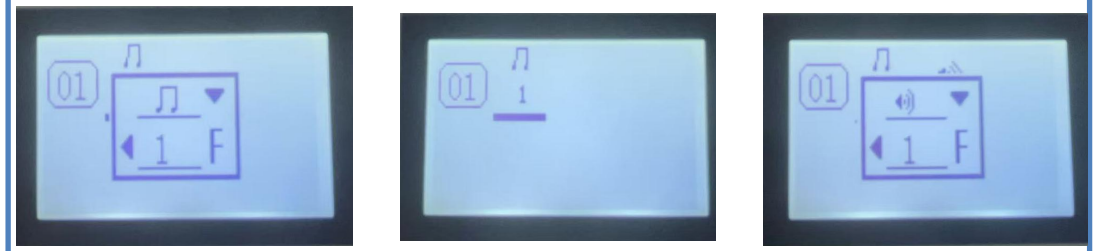
Quickly clicking 1-8 times corresponds to the following phrases: cat meow, dog bark, car sound, motorcycle sound, applause, "You're awesome," "Please pay attention," "That's so fun."



The 9th time was to play the recording.

Step 5 Start programming



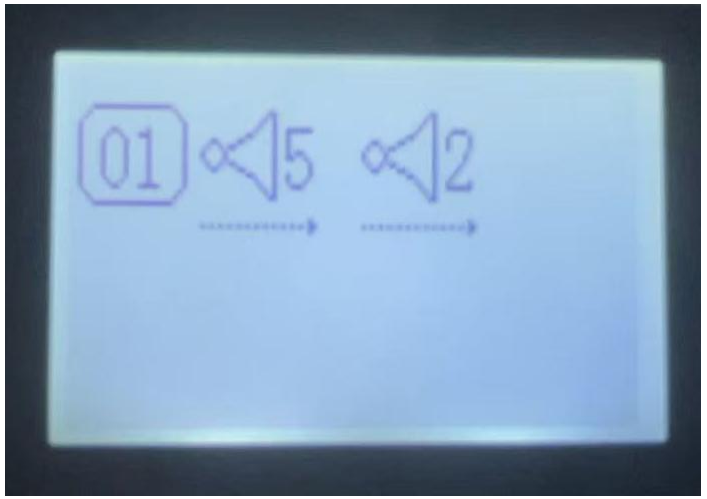


Long-pressing the advanced mode unlocks 19 sounds: cat meow, dog bark, car sound, motorcycle sound, applause, "You're awesome!", "Attention!", "So fun!", "I love you so much!", bird chirp, tiger roar, brakes, laughter, upbeat music, soothing music, energetic music, system startup, zero, one, two.

Use the page up/down keys to select and switch between recordings, up to 9 segments.

2.13 Buzzer

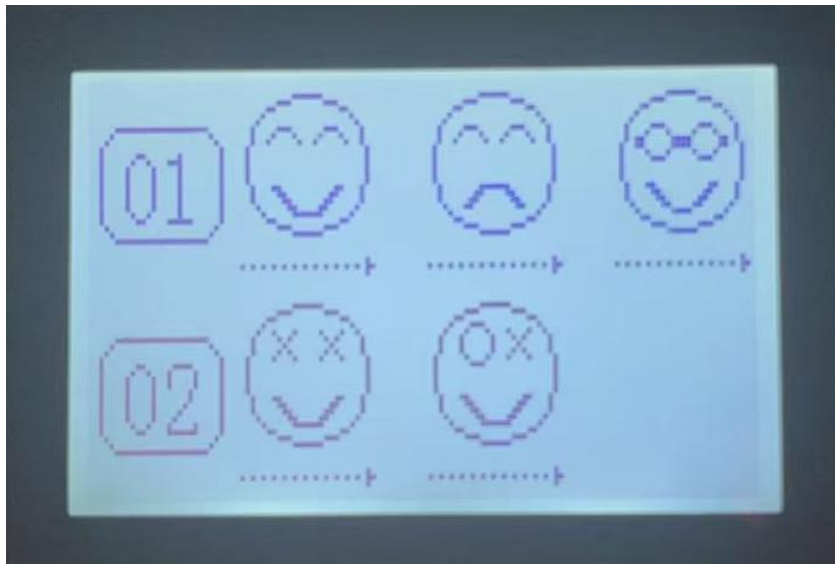

Turn on the buzzer
010 In key-screen programming mode: press briefly to set 1~7 syllables;
In real-time control mode, press once to play sound



Short press to select notes 1-7

2.14 Emojis display

Expression
010 When programming the normal mode of the key screen: short press can switch 5 expressions, while long press can set the switch of 5 expressions and set the display time of the expressions for 0.1~9.9 seconds



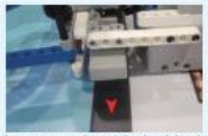
Clicking 1-5 corresponds to the above emojis, and the emojis will be enlarged to the entire screen when running.

2.15 Patrol - Black and white calibration

NASHENBOT

Line inspection

010 In normal mode of key screen programming: short press to select intersection line patrol and time line patrol; in advanced mode, long press to set the time of line patrol for 0.1~9.9 seconds

Long press switch	NASHENBOT
Mode selection far right	Function BT Settings ▶Enter
Move down twice X2 Click OK X1	Function BT Settings ▶Enter
Move down once X1 Click OK X1	Advanced Calibration Versions ▶Enter
"Place the machine on the map, and align the grayscale with the black line (please put all five holes of grayscale on the black line). Click OK X1	Please place tracking sensor directly above the black trace line ▶Enter  Align the grayscale with the black line

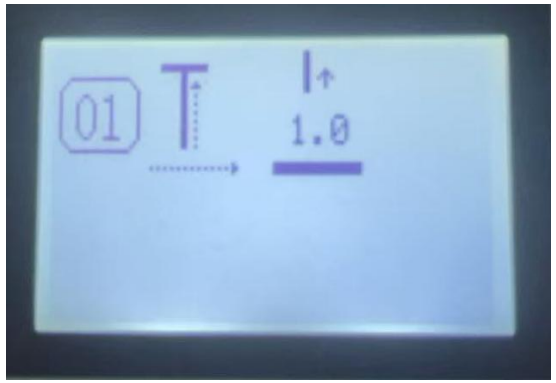
"Place the machine on the map, and aim the grayscale at the light-colored area (please place all five holes of grayscale in the light-colored area). Click OK X1	Please place tracking sensor directly above the light white ground ▶Enter  Align grayscale to light areas
If the calibration is successful, it is considered a success.	Calibration Pass ▶Enter
If the calibration fails, the above operations need to be repeated.	Calibration Fail ▶Enter

Before using the line-following module for line following, and whenever the map is changed or the environment and lighting conditions change significantly, a black-and-white calibration is required. If calibration consistently fails, try adjusting the grayscale position up or down by 4 millimeters (half a unit) and then try again. It is also crucial to ensure that the grayscale values are installed from left to right (1-5), and not backwards.

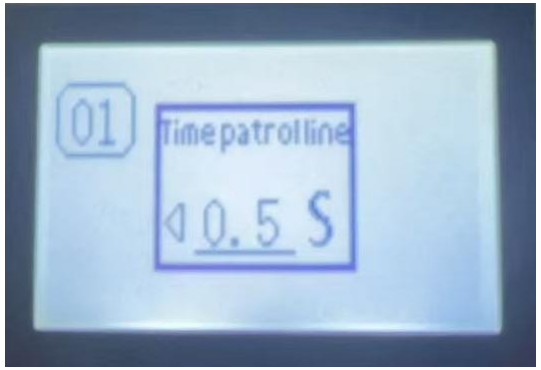
2.15 Patrol

NASHENBOT

Line inspection
010 In normal mode of key screen programming: short press to select intersection line patrol and time line patrol; in advanced mode, long press to set the time of line patrol for 0.1~9.9 seconds



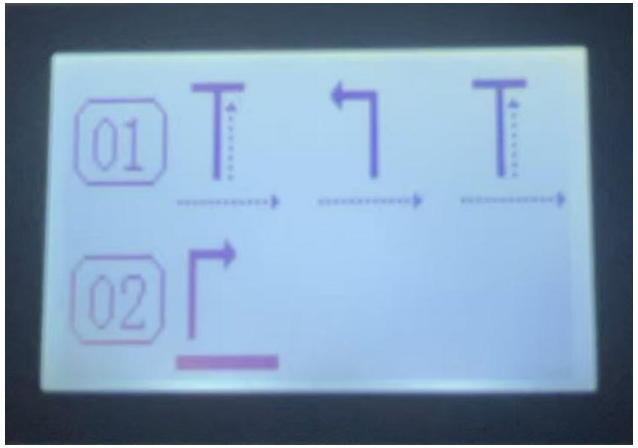
In normal mode, clicking once will guide the line until it encounters an intersection, which includes left intersections, right intersections, crossroads, T-junctions, etc. Clicking twice quickly will guide the line for 1 second and then stop.



In advanced mode, a long press controls the timing of the line patrol. The left-hand arrow key increases the time by 0.1 seconds with each click, and a long press will increase it rapidly, up to a maximum of 9.9 seconds before cycling back to 0.1 seconds.

2.16 Patrol - turn to next line

	
Turn left at the intersection	Turn right at the intersection
Turn left at the intersection in a short time	Turn right at the intersection in a short time
Only valid in 010 key screen programming mode	Only valid in 010 key screen programming mode



The behavior is the same in normal and advanced modes. When turning left or right to the next line, half of the car body will move forward first, then rotate while searching for the line, and stop after finding the line. It is often used in conjunction with intersection line inspection.

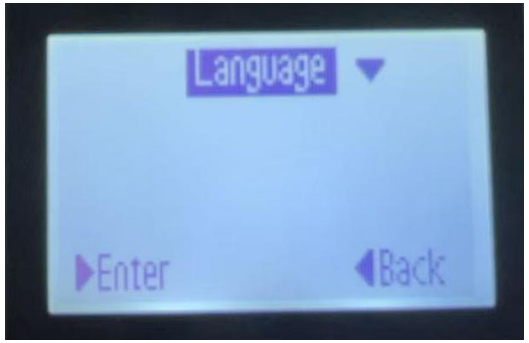
2.17 Loop

Icon		
Name	Start cycle	End cycle
Parameter	Short press to set infinite and 2-9 cycles	-

Loops are usually added before a program to be executed repeatedly. It's a good idea to pair the start and end loops. This won't affect a single loop, but you need to be careful when your program has multiple loops.

2.18 Other operations

Version number lookup: If some functions differ from the reference examples, it may be due to a version update. You can check the version number here and connect to your computer to update the firmware on AICODE.



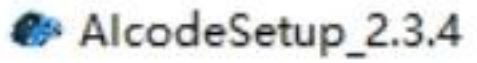
Language Change: You can change the interface display language here.

3.1 Programming on PC - Software installation

NASHENBOT

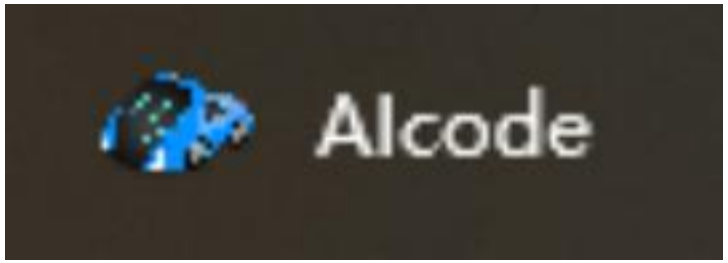
Software installation: First, you need to install the AICODE programming software on your computer.

Please follow these steps to install the software by opening the attached file.
Please note the following during installation:

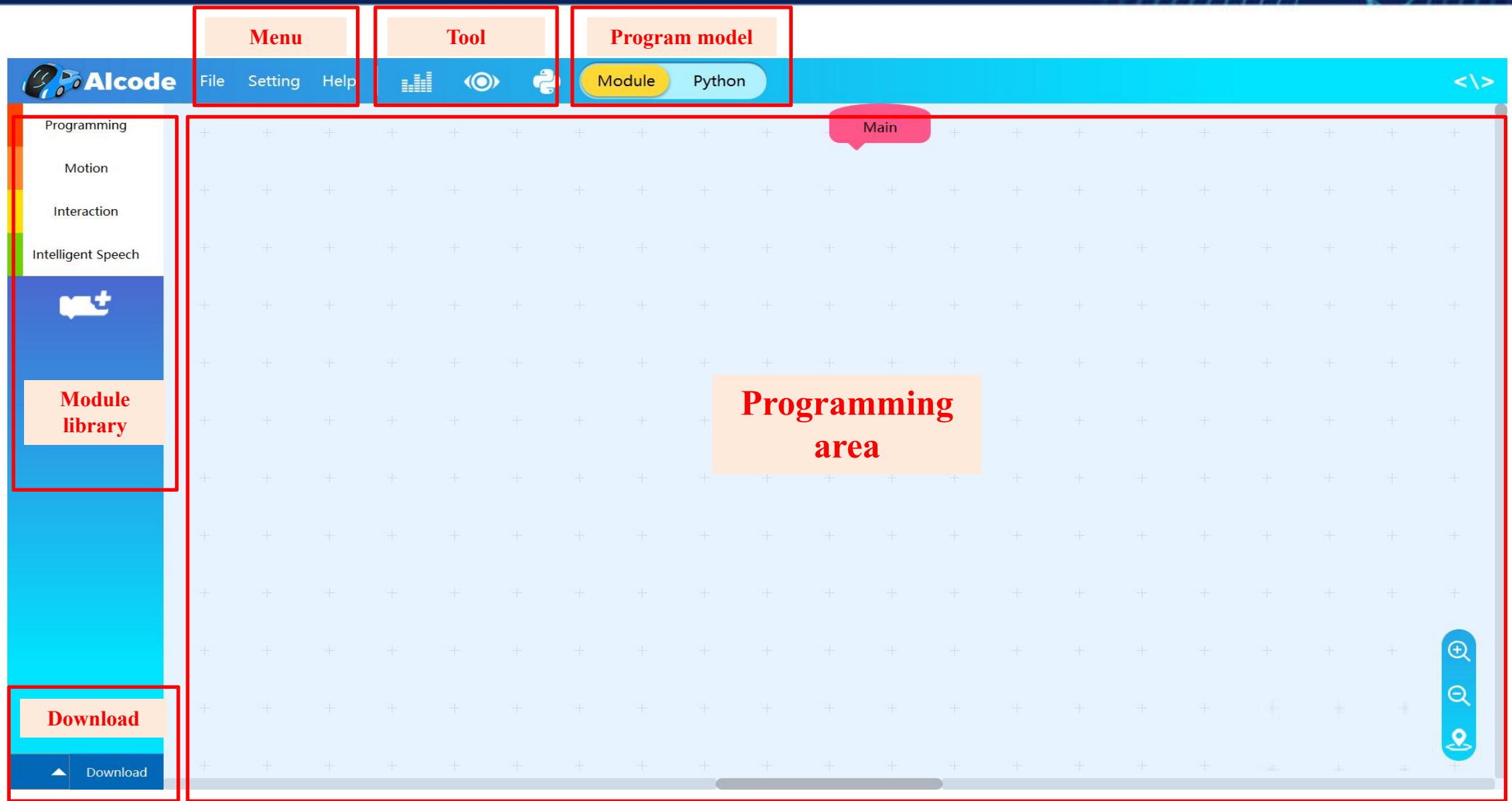


1. This software is compatible with Windows 7 and later operating systems, but not with macOS.
2. Please close all antivirus software before installation to ensure complete installation of necessary plugins, which may affect usability.
3. If a pop-up window appears during installation, please click "OK". If a black pop-up window appears, please wait for it to load; it will close automatically upon completion.

After installation, double-click the icon to open the software.

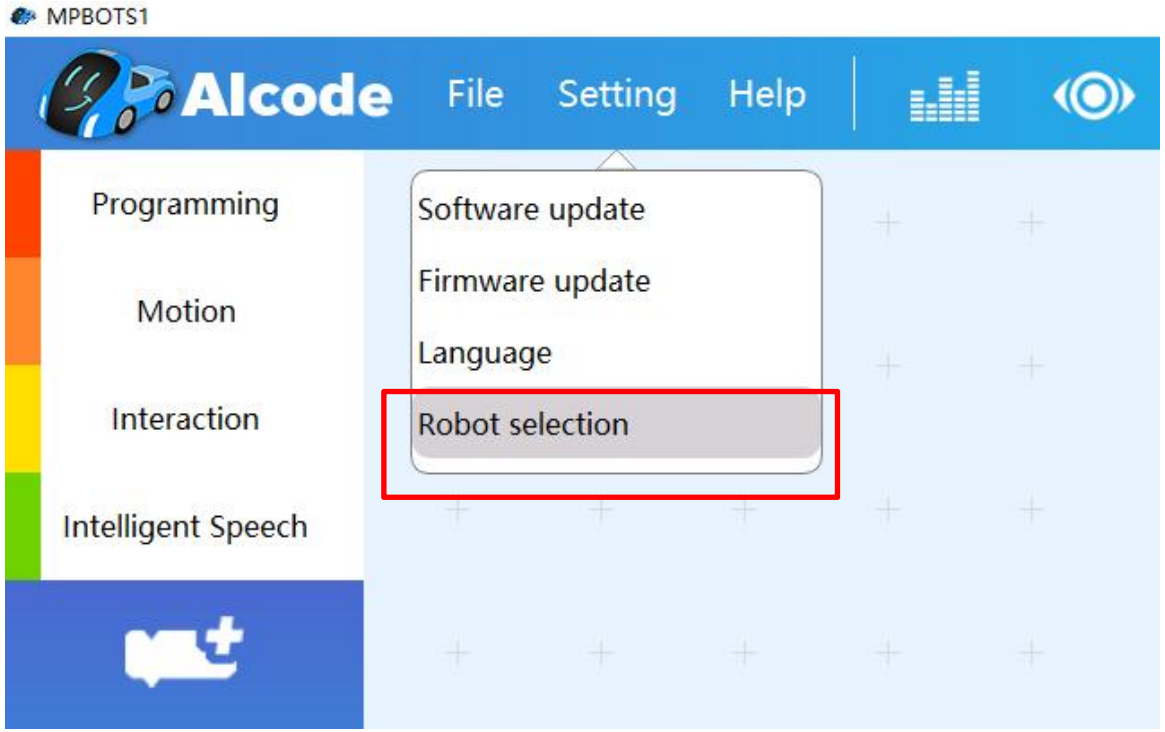


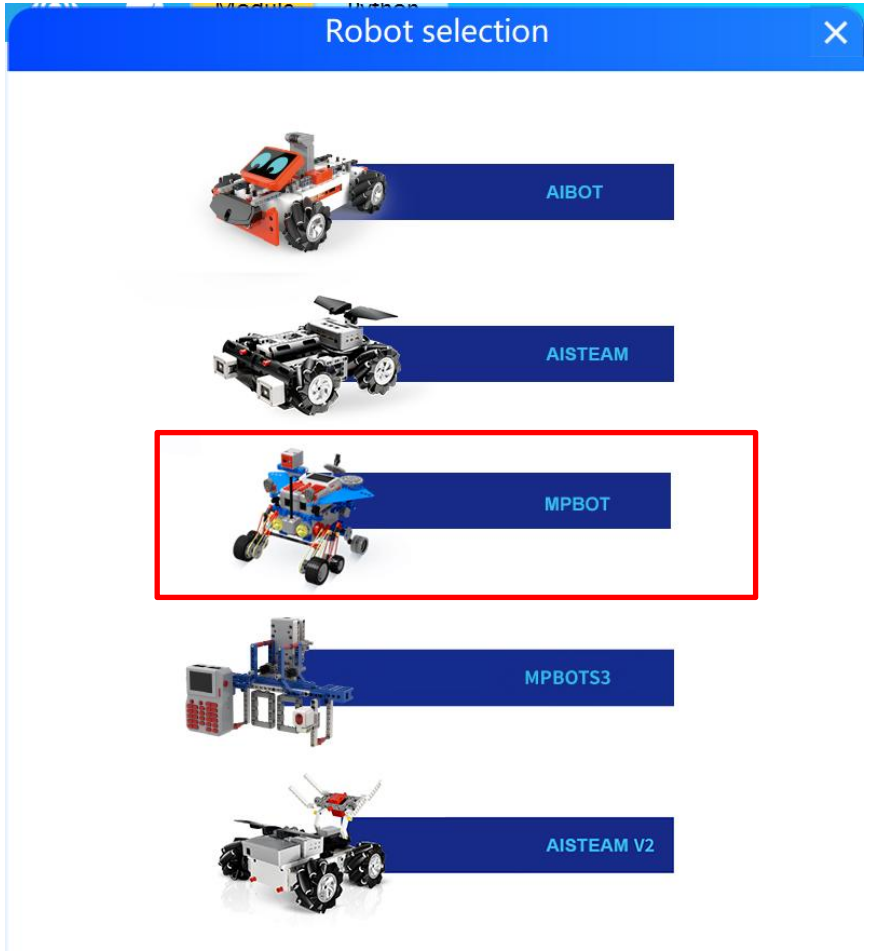
3.1 Programming on PC - Interface



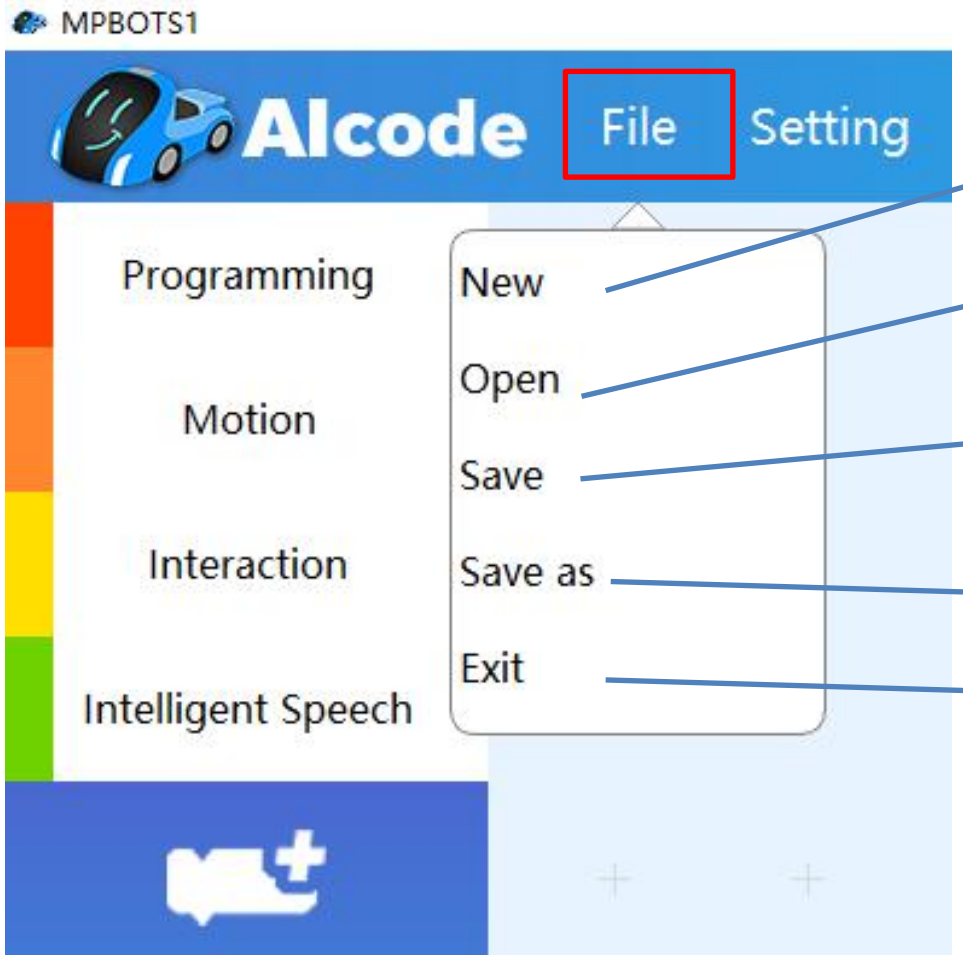
3.2 AIcode Interface -- Robot selection

This programming software supports multiple machines; first, you need to select the corresponding robot.





3.2 AIcode Interface -- File menu



New file

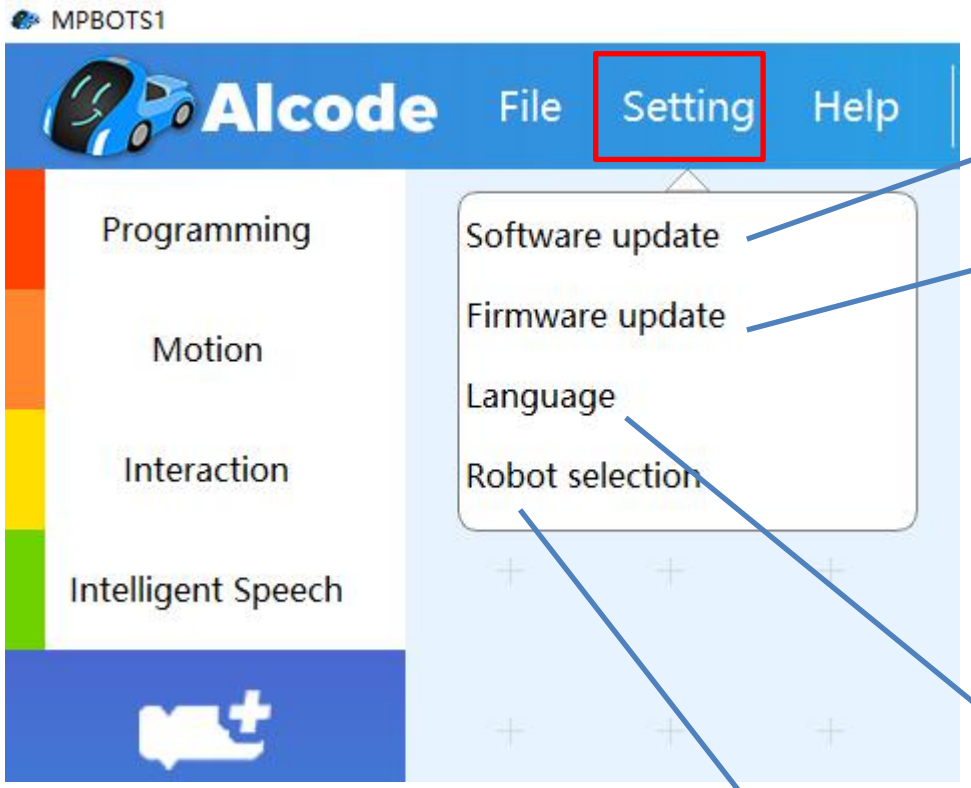
Open the file, but make sure to open it through the software; simply double-clicking an .nsh format program file will not open it.

Save your files frequently as the program does not automatically back them up.

Save file as

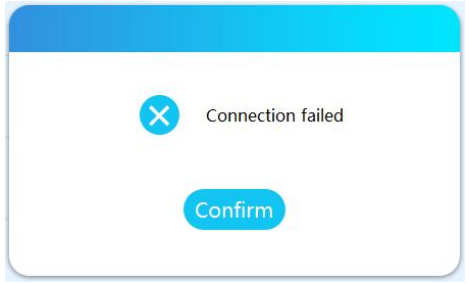
Exit

3.2 AIcode Interface -- Setting



Software updates are generally sufficient for stable normal use and do not require updates.

For firmware updates, you must first power on the device and connect it to your computer via USB cable.



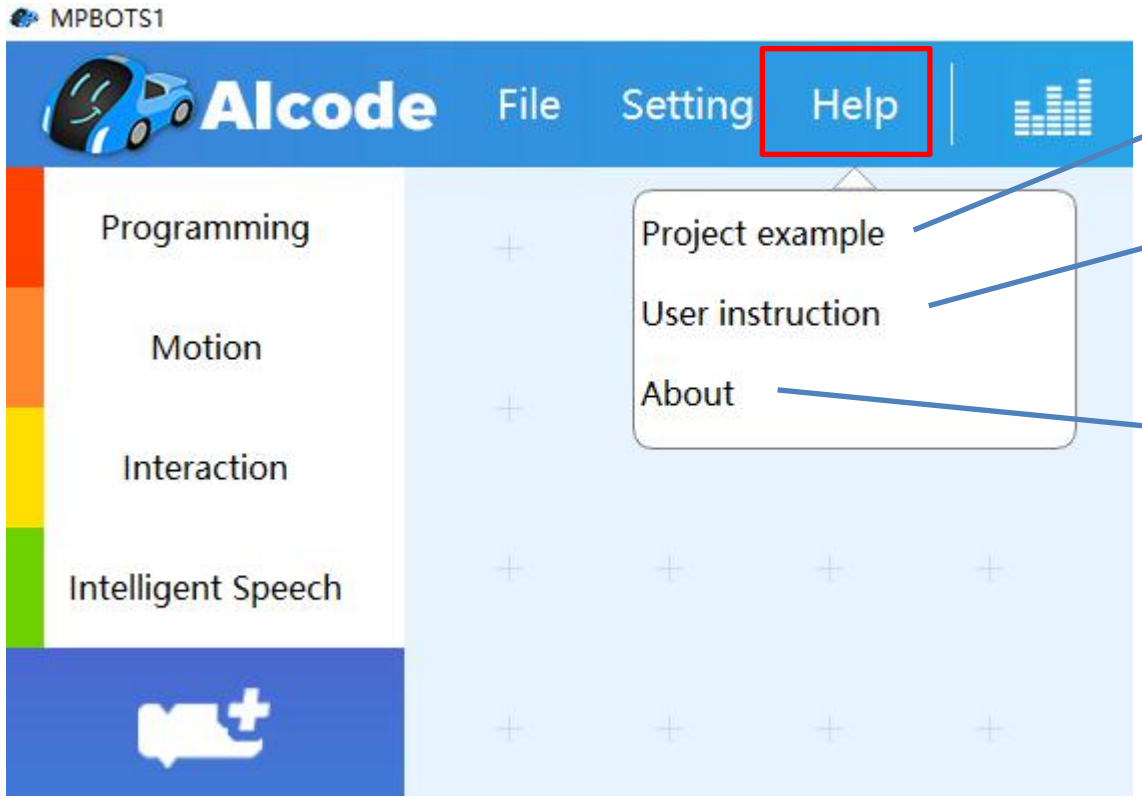
Connection not established

Switch language

Switch robot

Blue indicates items that are updatable, gray indicates the latest version.

3.2 AIcode Interface -- Help

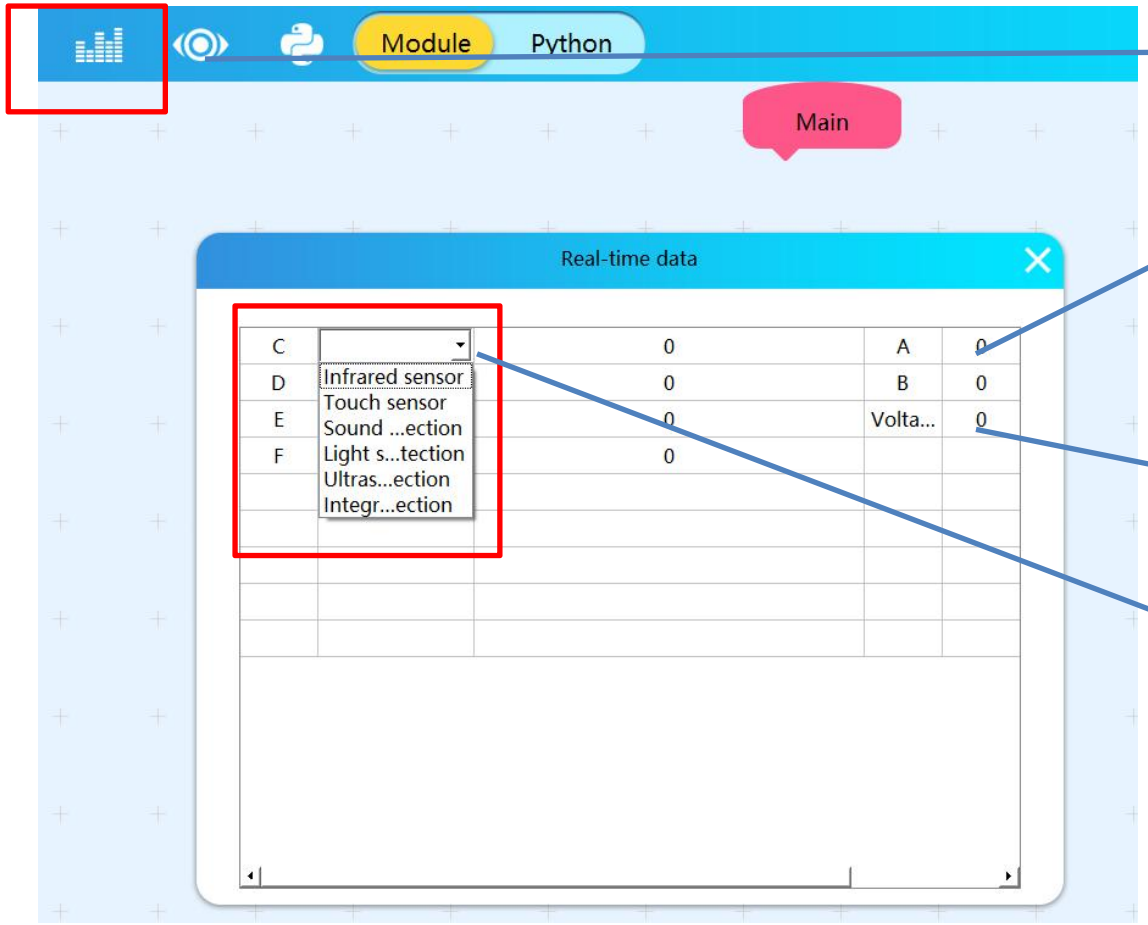


Project Example: Please refer to the setup diagram

User Manual: View the detailed user manual

You can view company information
and software version number.

3.2 AIcode Interface -- Real-time data checking



Clicking will bring up a real-time data box. Data is only available when the machine is connected to the computer via USB cable.

Displays real-time closed-loop encoder motor encoding value.

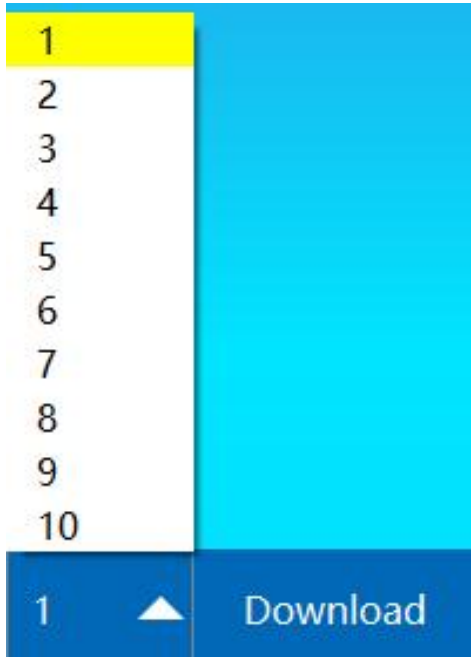
Display voltage

Selecting the corresponding sensor from the drop-down menu will display real-time data.

3.2 AIcode Interface -- Download program

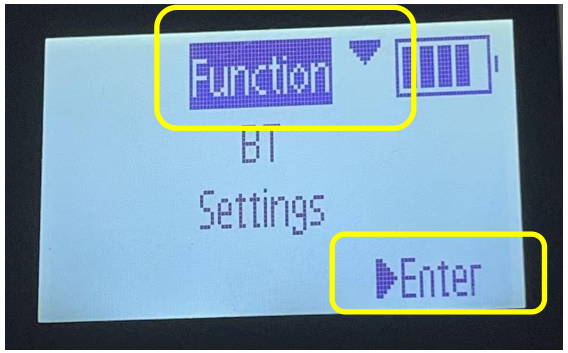
Unable to download due to lack of connection

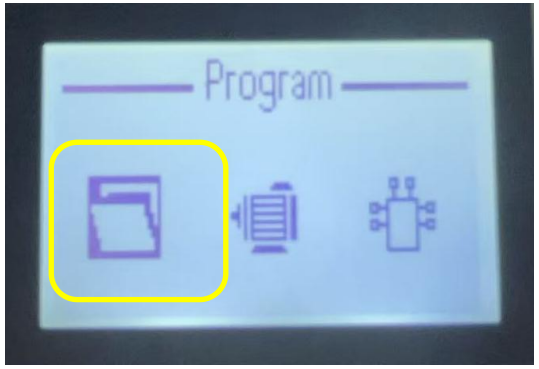
Downloads are now available

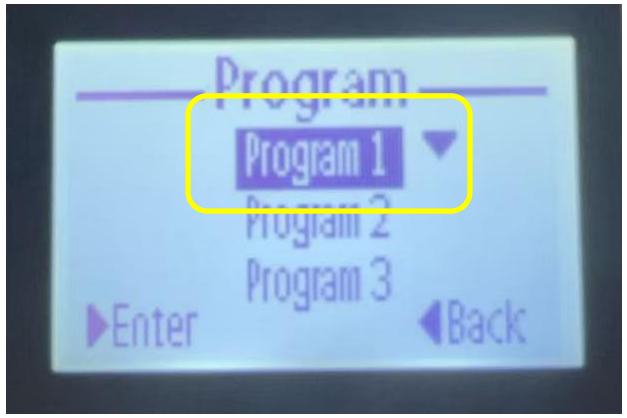


A total of 10 programs can be downloaded.

3.2 AIcode Interface -- Run program

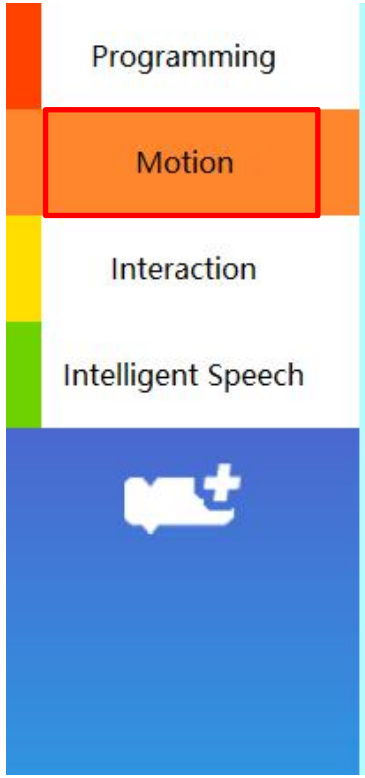


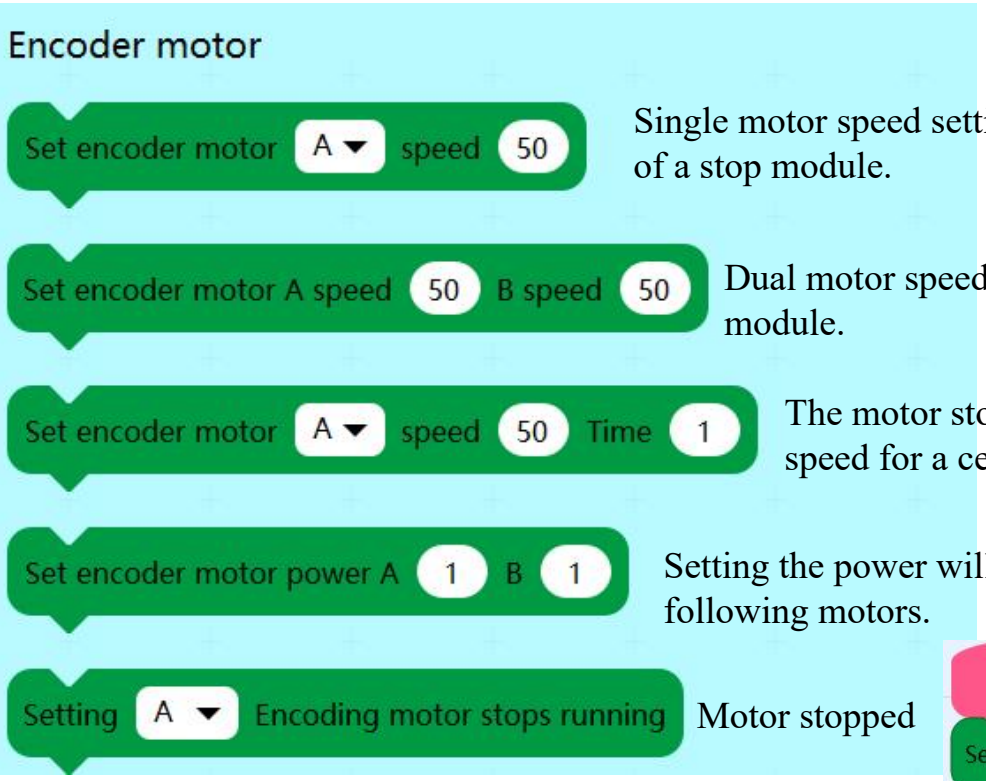




3.3.1 Programming Module -- Encoding Motor Motion

NASHENBOT





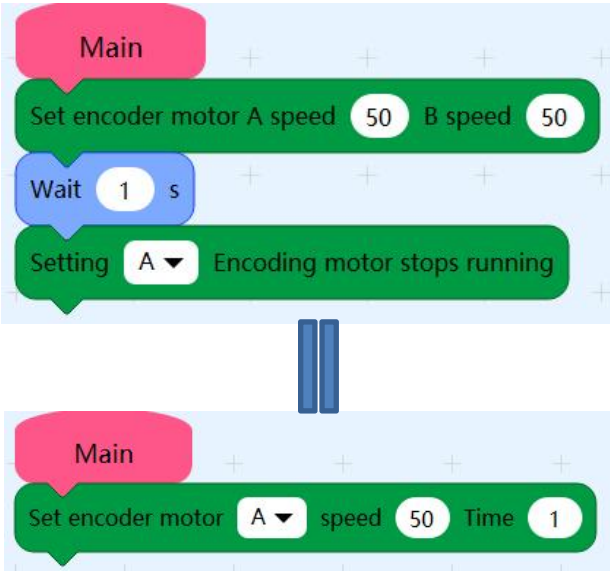
Single motor speed setting generally requires the use of a stop module.

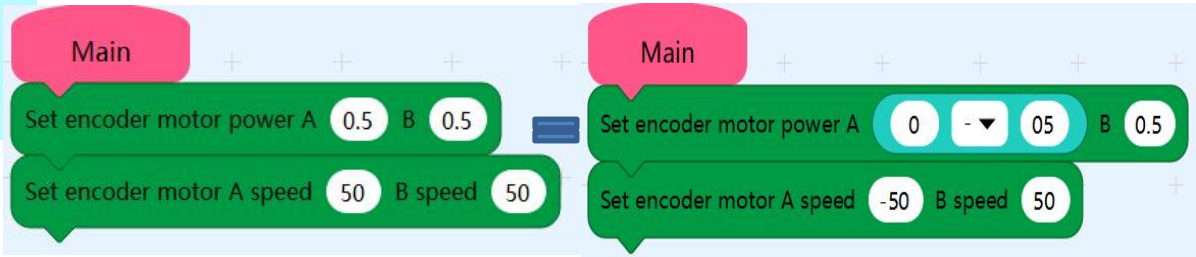
Dual motor speeds generally require a stop module.

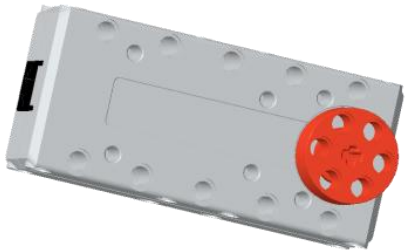
The motor stops after moving at a certain speed for a certain period of time.

Setting the power will affect the speed of all the following motors.

Motor stopped



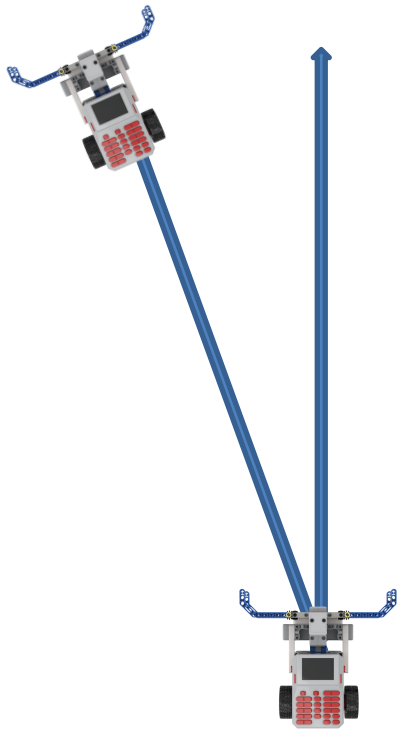


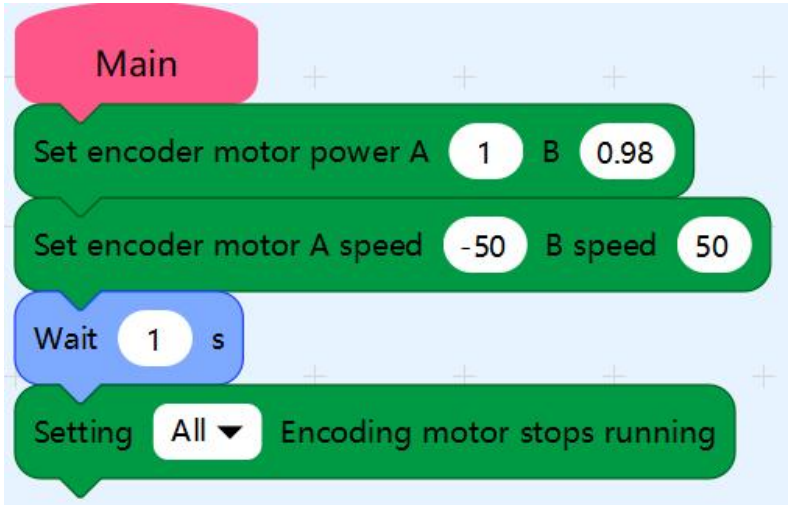


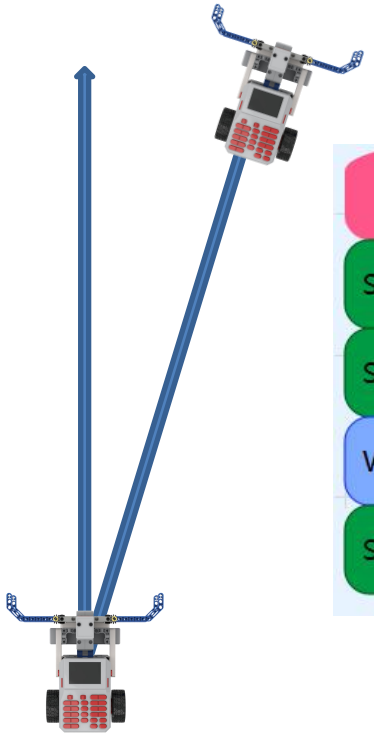
Set the motor speed; the speed range for the encoded motor is -100 to 100. A positive speed corresponds to forward rotation, and a negative speed corresponds to reverse rotation.

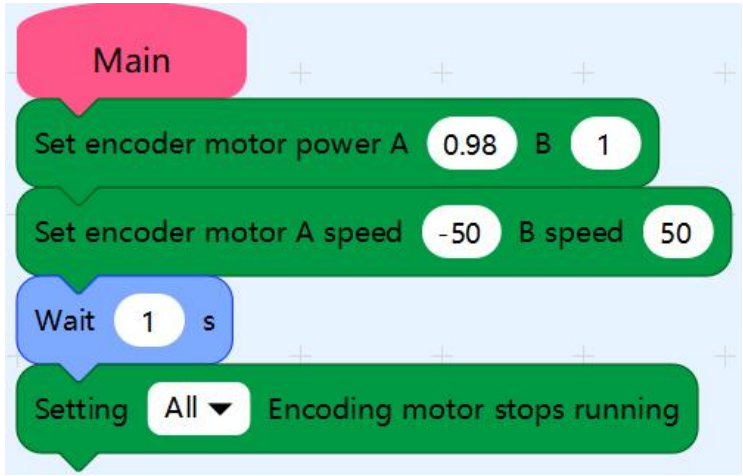
With power adjustment, the actual speed is 25, commonly used for speed regulation. If the two motors are not very synchronized, the power can be adjusted to 0.9-0.99, but the power should not exceed 1.

3.3.1 Programming Module – Encoding Motor Motion Power Adjustment



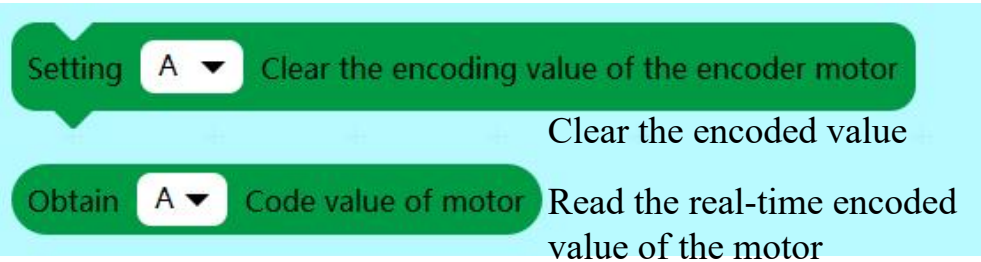






The motors used to build the cart are arranged in opposite directions, so pay attention to their orientation.
If the cart, which was originally moving straight, veers to the left, it means the right motor (B) is rotating too fast; the power of motor B needs to be set to less than 1.
If the cart, which was originally moving straight, veers to the right, it means the left motor (A) is rotating too fast; the power of motor A needs to be set to less than 1.

3.3.1 Programming Module -- Encoded Motor Moves Based on Encoded Values

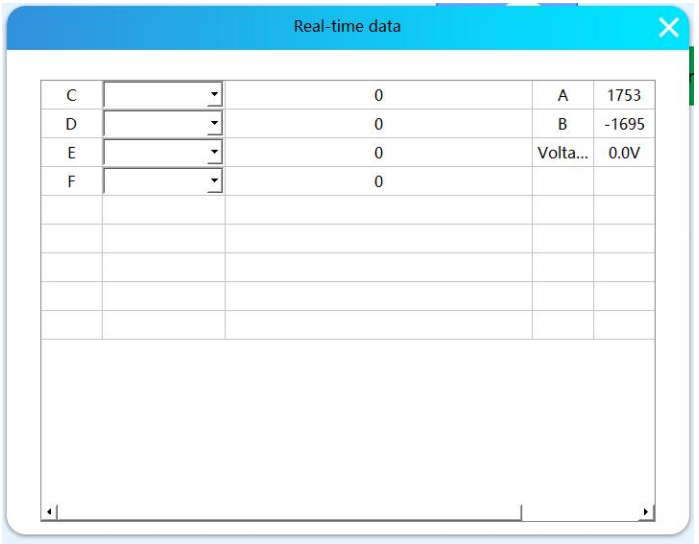


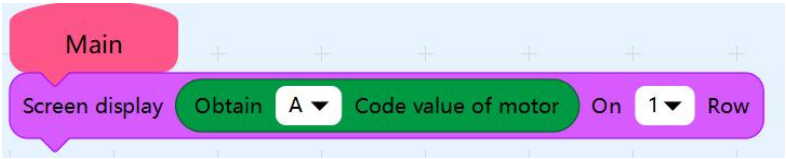
The difference between an encoder motor and a regular motor lies in its built-in encoder value monitoring device, which provides real-time encoder value feedback. However, like a pedometer, this value accumulates continuously—increasing with forward rotation and decreasing with reverse rotation. It generally needs frequent resetting to reduce computational complexity during use.

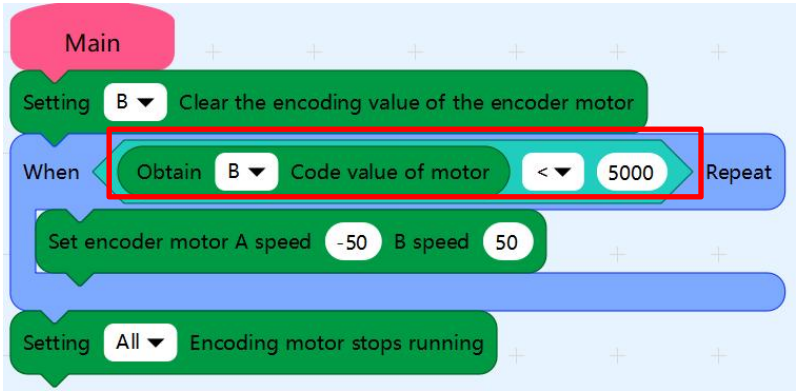
The value can be read from real-time data or displayed on the screen.

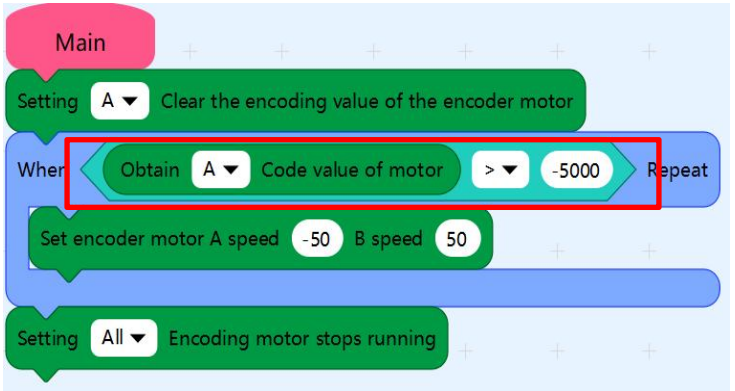
To determine or compare motor encoder values, select one motor; it's recommended to choose the motor that rotates forward.

For example: Motor A displays an encoder value of 1000 after startup. To control it to rotate forward to 5000 encoder values, you could let it move until the encoder value reaches 6000 and then stop. However, if the initial encoder value is different next time, the program needs to be modified. A more convenient approach is to reset the encoder value first, and then each time you check the value, let it move until the encoder value reaches 5000 and then stop.

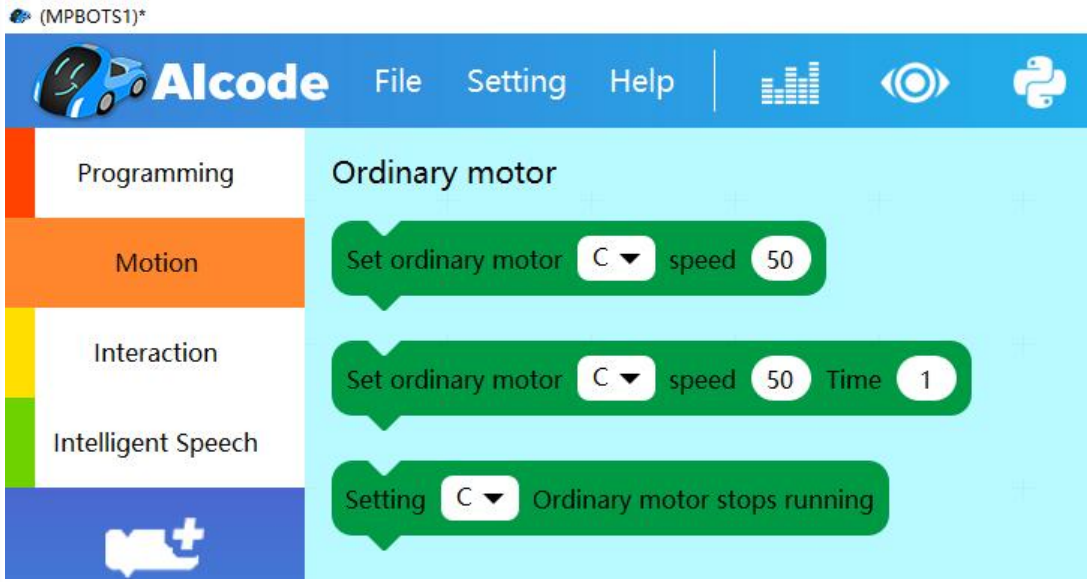






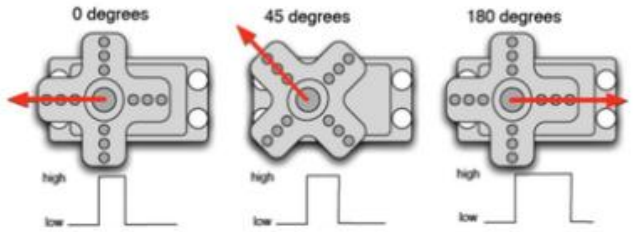


3.3.2 Programming Module -- Open-Loop Motor Motion

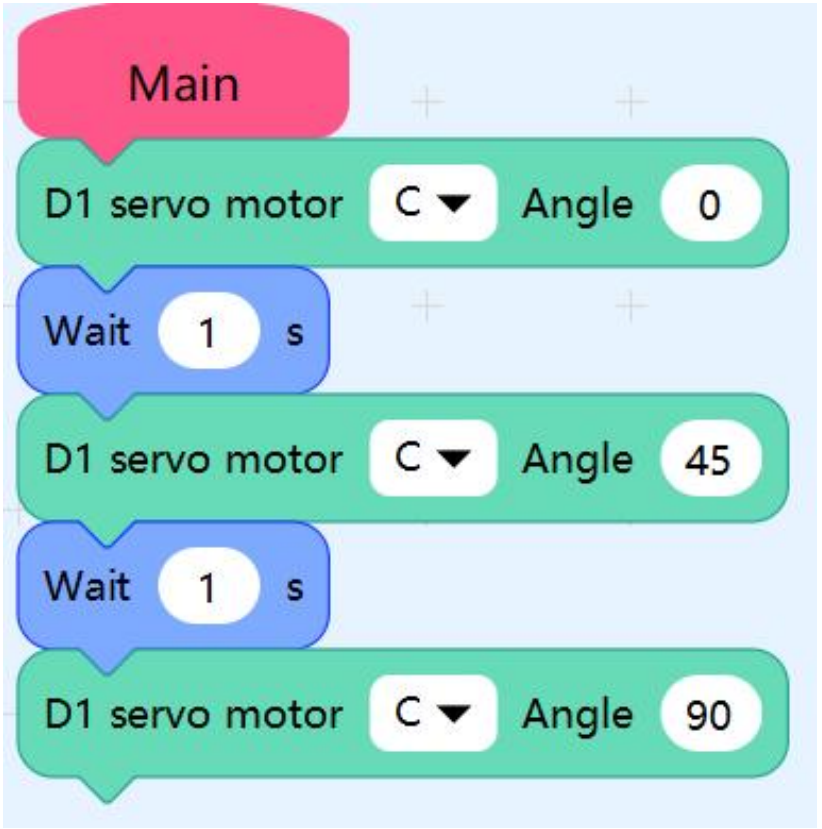


Open-loop motors are also used by setting individual speeds, stops, or timers.
However, if an open-loop motor seems to rotate in the opposite direction despite the same program, it's highly likely that the orientation of the components is incorrect. You need to check the orientation of the text on the side of the components during assembly.

3.3.3 Programming Module -- Servo Motion

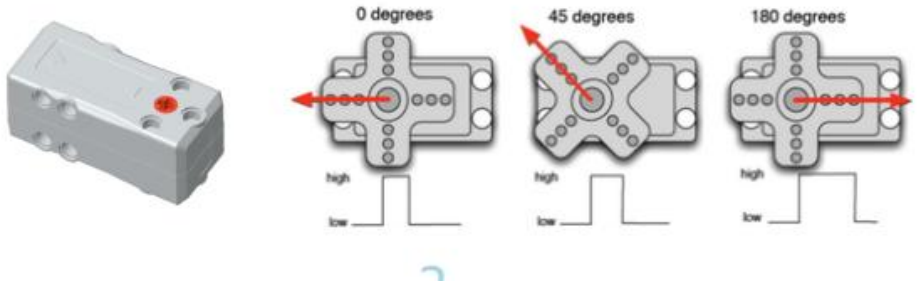


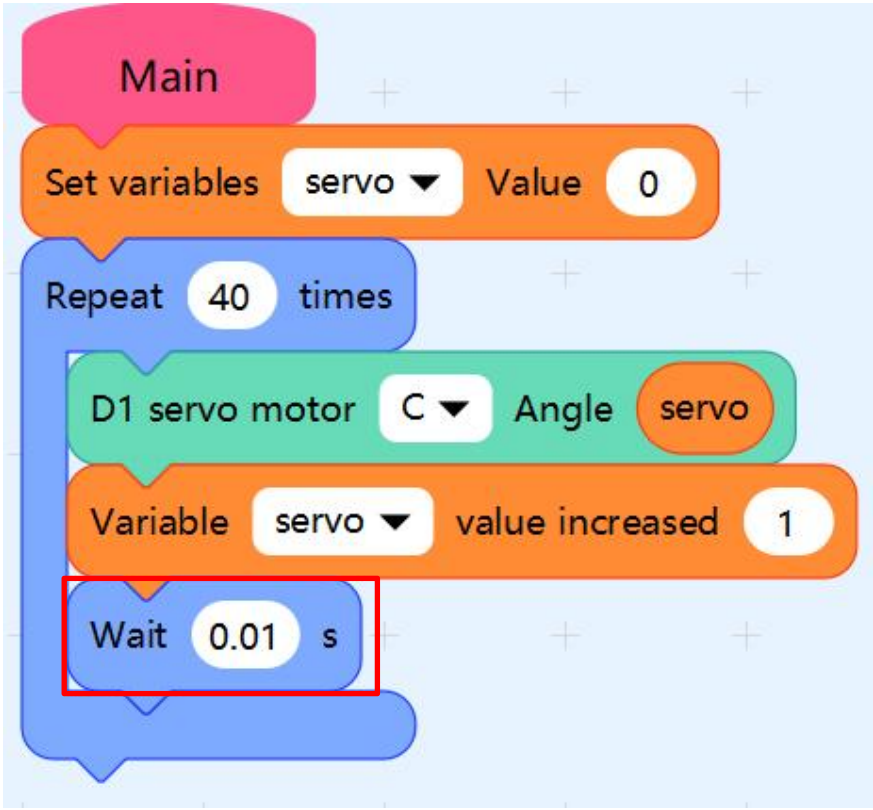
A servo motor is simply a state angle; its directional control depends on the previous angle state. When using a servo motor, it's generally advisable to first adjust it to zero degrees before connecting it to structural components. It's common to set several different angles to represent steering, and usually, an initial servo motor value needs to be set.



3.3.3 Programming Module -- Servo Motion

The servo motor's rotation speed is fixed, but sometimes too high a speed makes it unstable for certain tasks. This can be addressed by programming to reduce the speed by setting variables. The larger the waiting time value, the slower the speed.

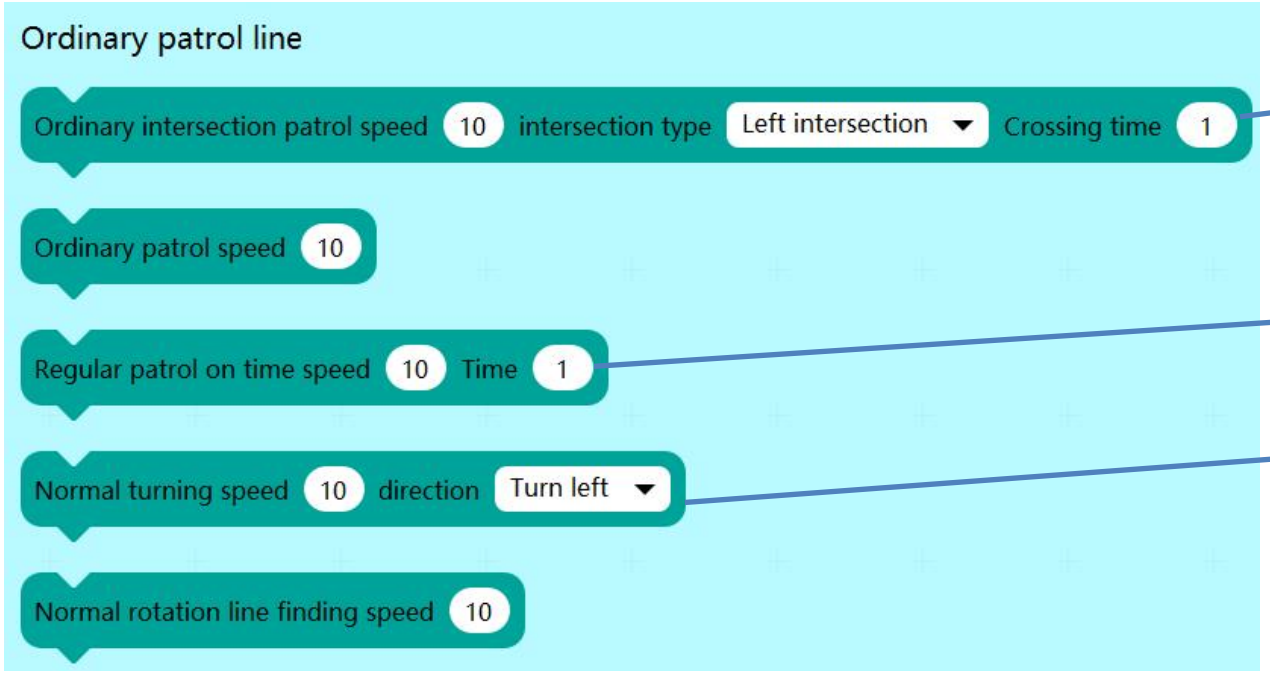




3.3.4 Programming Module -- Closed-Loop Encoding Motor Patrol Module

NASHENBOT

This line-following mode offers more detailed parameter settings and greater flexibility compared to the micro-programming mode. However, it still requires prior black-and-white calibration on the machine before use.



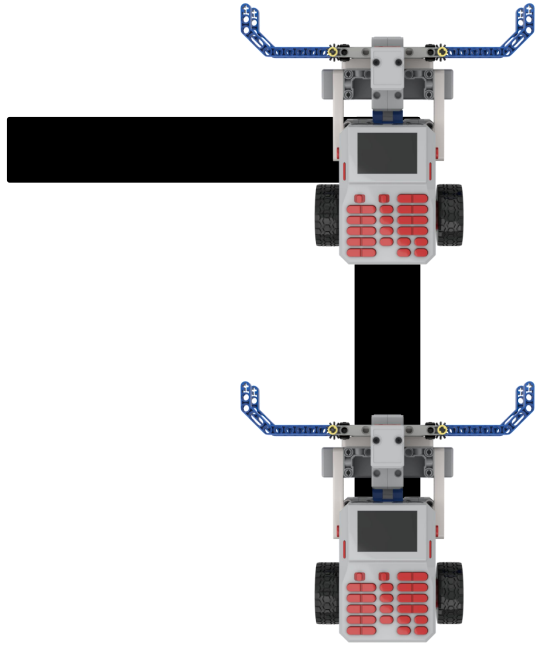
The speed of the line patrol can be set, and intersections are further divided into left and right intersections. The judgment for a left intersection is based on the detection of the line by sensors 1 and 2, while the judgment for a right intersection is based on the detection of the line by sensors 4 and 5. If it is a cross shape, it can patrol either left or right.

Regular scheduled line patrol

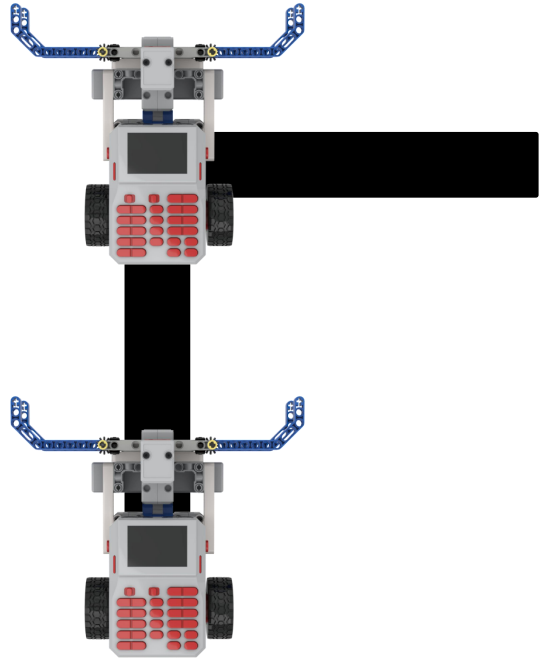
Turn left to the next line, note that the speed is positive; turn right to the next line, note that the speed is negative. This module is usually already on a line; first disconnect from the line, then find a new line.

Turn left to the next line, note that the speed is positive; turn right to the next line, note that the speed is negative. These modules are generally not on the same line; use the turning direction to find the next line.

3.3.4 Programming Module -- Closed-Loop Encoding Motor Patrol Module



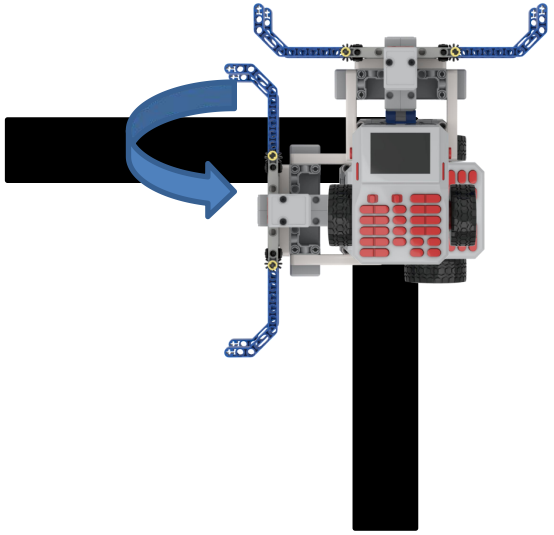
Left intersection

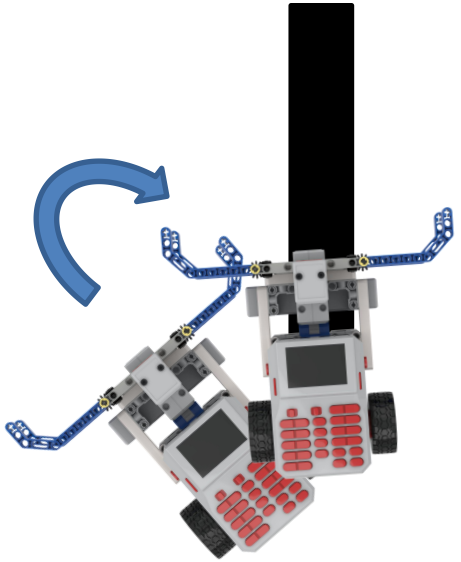


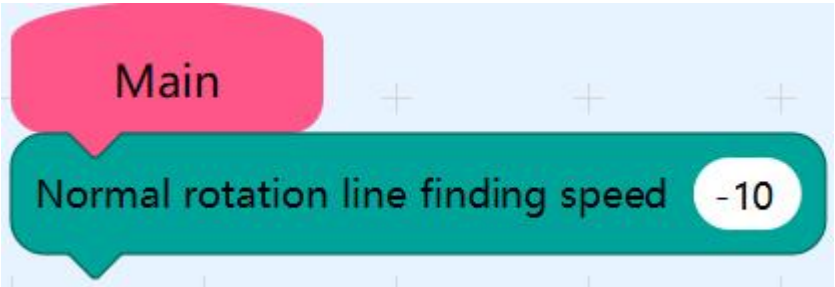
Right intersection

When the time is 0, the grayscale is just right. If you need to turn, you generally need to move the car forward a little more, so you need to set this time.

3.3.4 Programming Module -- Closed-Loop Encoding Motor Patrol Module

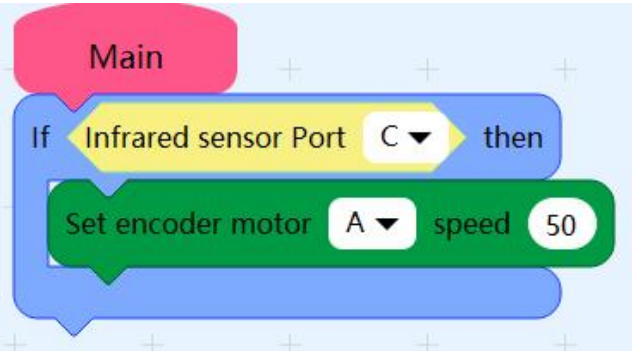




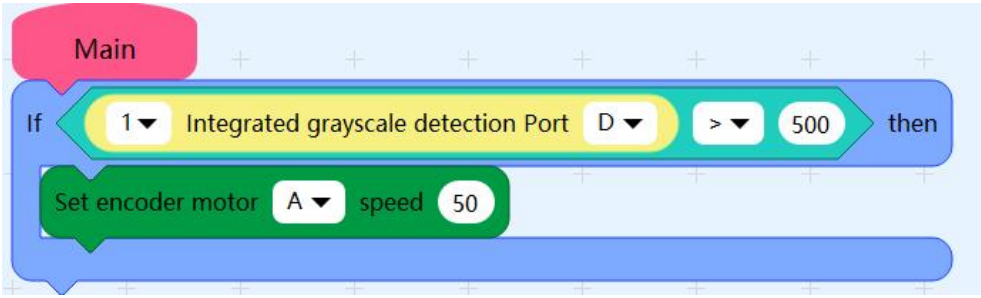


3.4 Programming Module -- Sensors

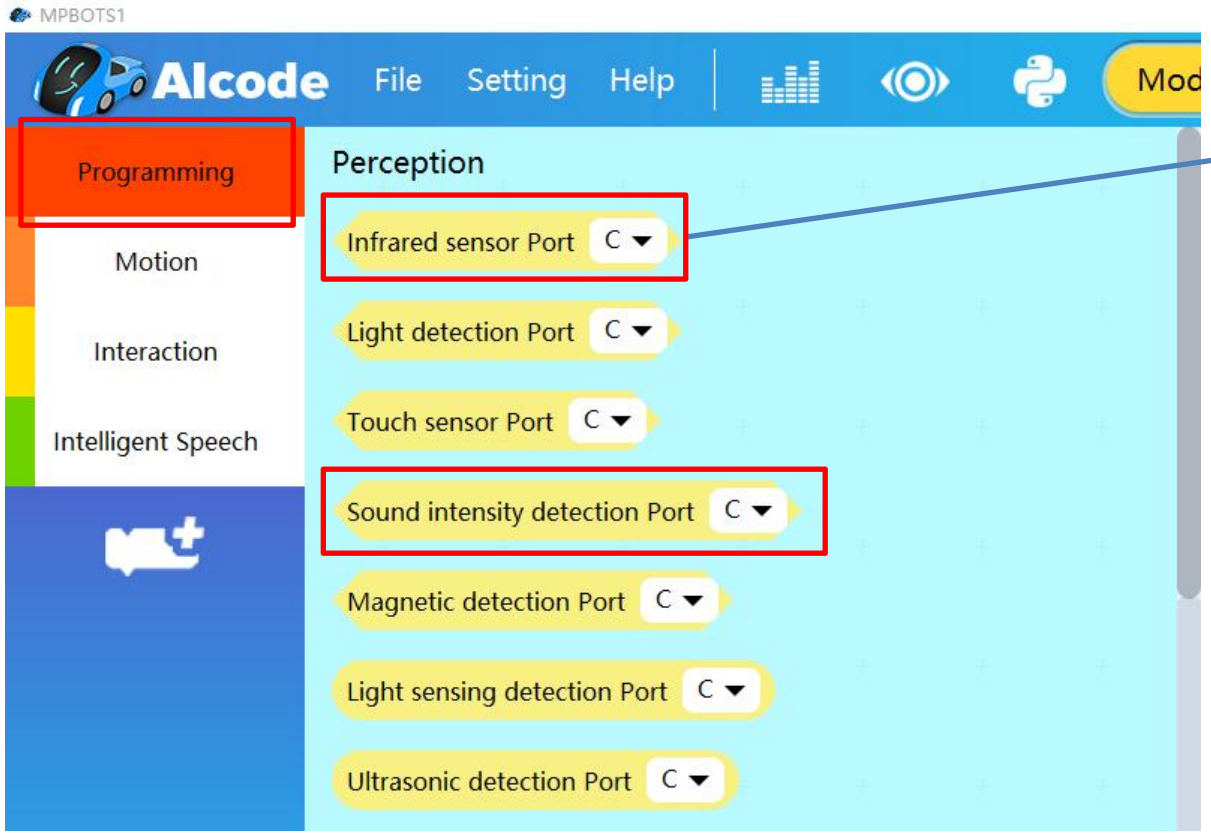
The hexagonal sensor icon is generally used to determine whether it has been triggered. There are usually only two states: 0 indicates that it has not been triggered, and 1 indicates that it has been triggered.



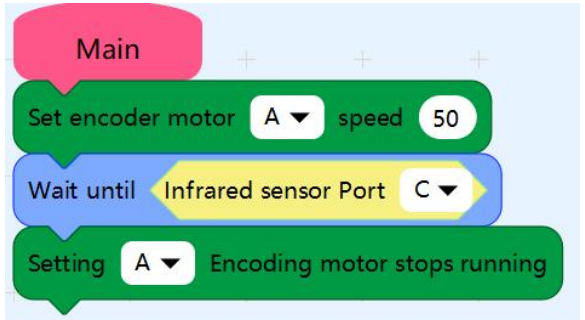
Circular sensor icons typically represent numerical values, which often need to be calculated or compared before being used for judgment, and may contain multiple values.

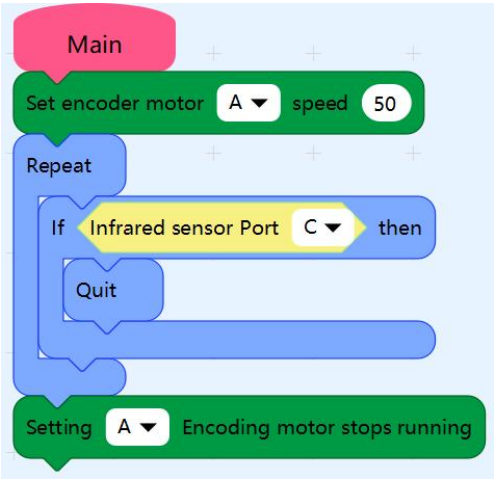


3.4.1 Programming Module -- Infrared & Sound sensor

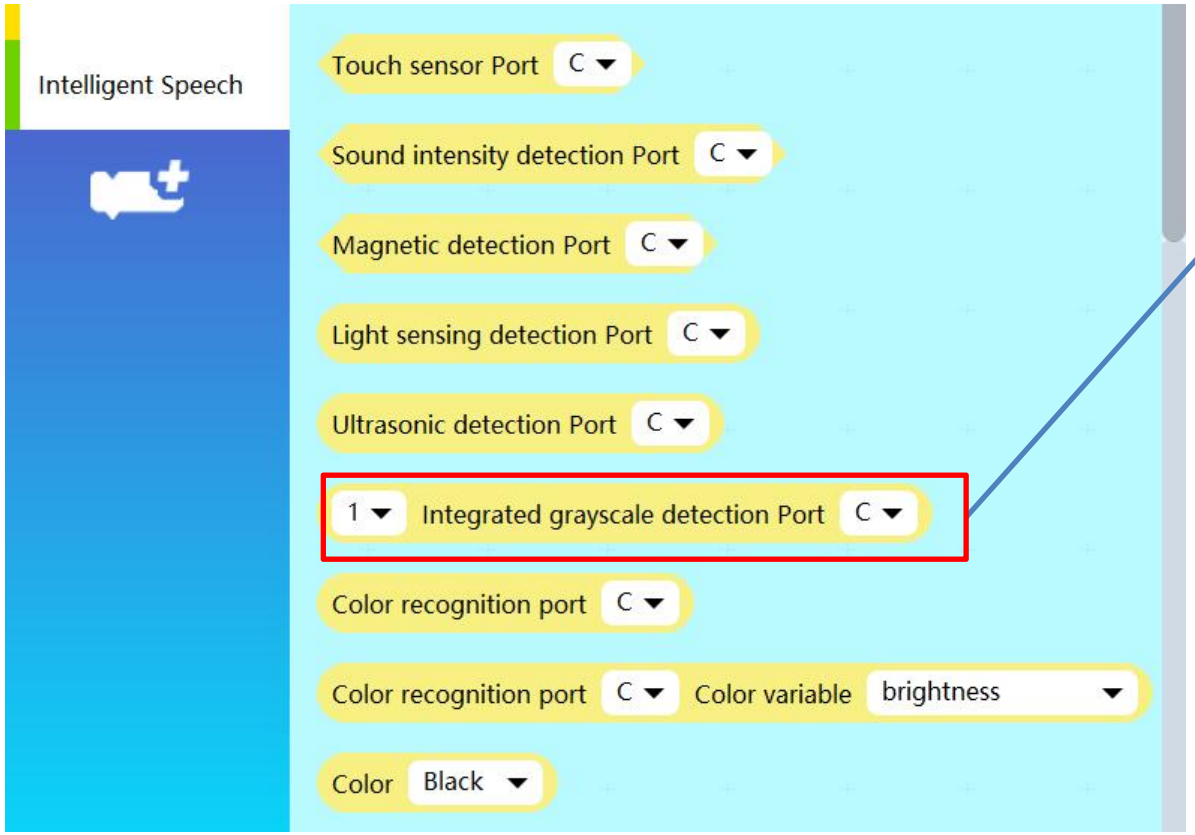


Example: Motor A moves, and stops when the infrared sensor detects an obstacle. This is usually done in conjunction with repeated actions.

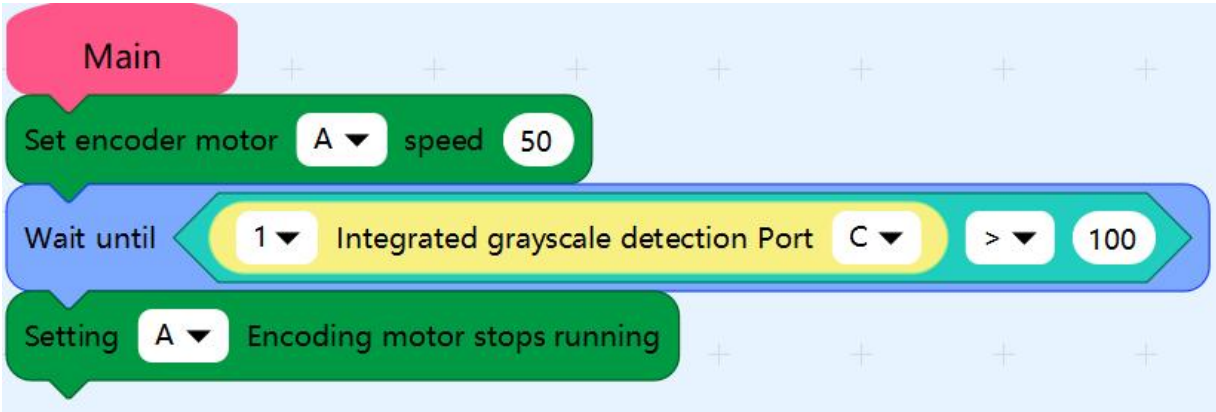




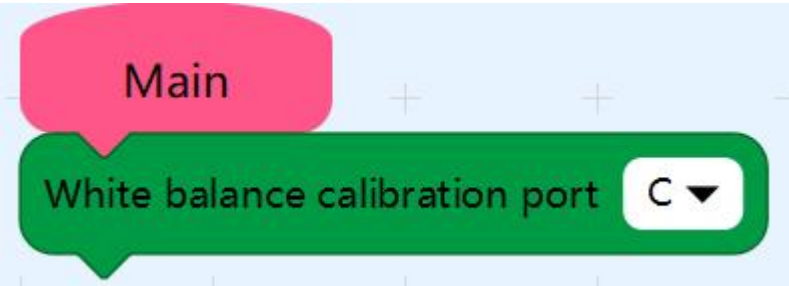
3.4.2 Programming Module -- Integrated five-grayscale sensor

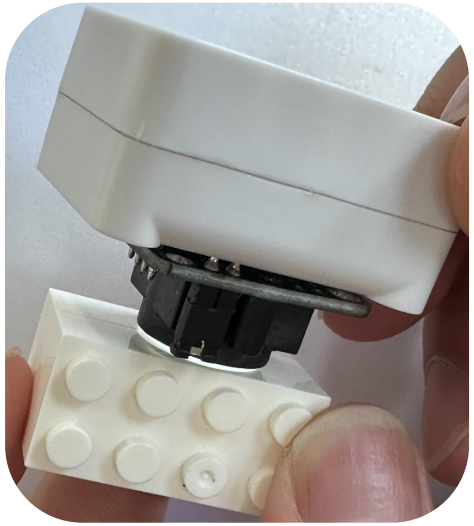


Compare grayscale values. For example, motor A keeps moving until the grayscale value of motor A is greater than 100, which means it stops when it perceives a darker color.



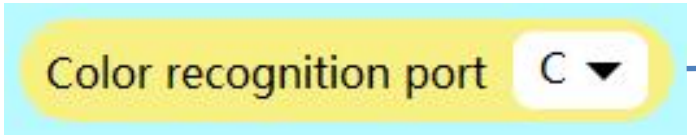
3.4.3 Programming Module -- Color sensor





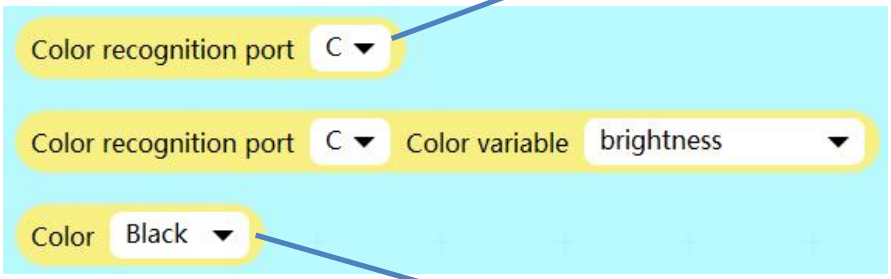
Calibration is necessary to improve its accuracy and stability. When using it for the first time, the color sensor needs to be calibrated for white balance. (Note: Once calibrated, it does not need to be calibrated again.) Connect the program to the main program, download the program, and then place the color sensor's detection head firmly against the white module to perform calibration.

3.4.3 Programming Module -- Color sensor

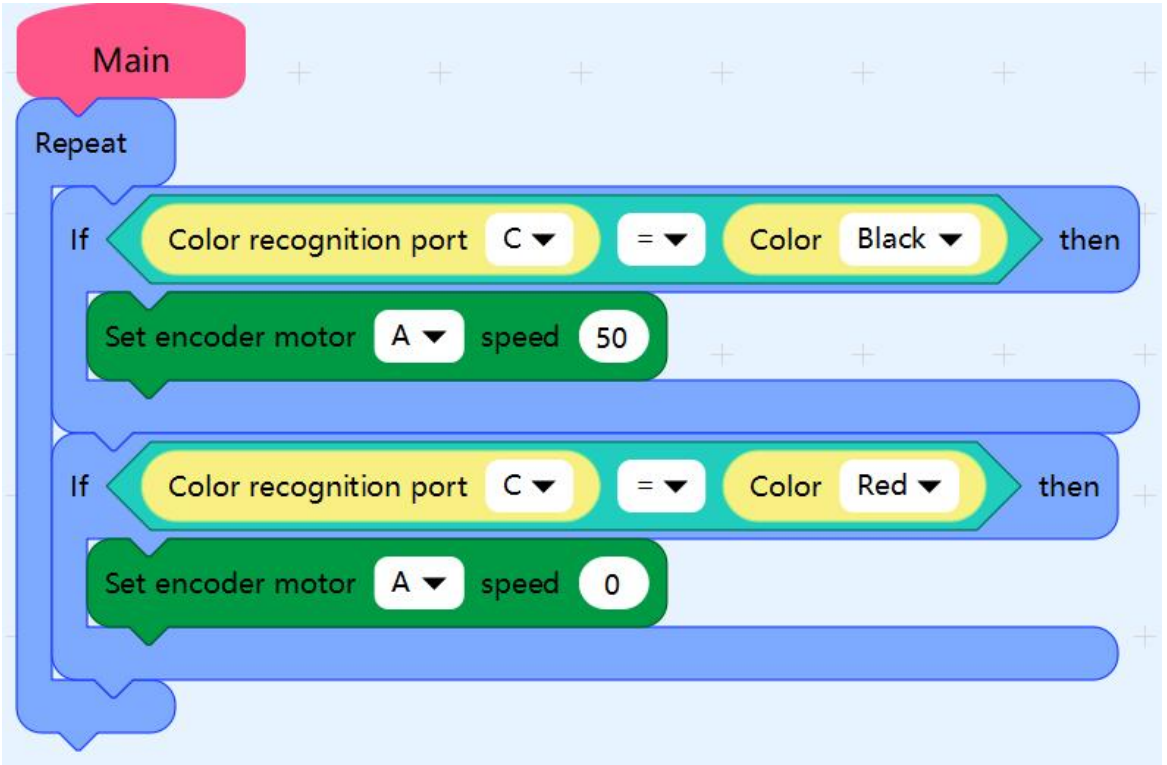


Generally used for display and judgment

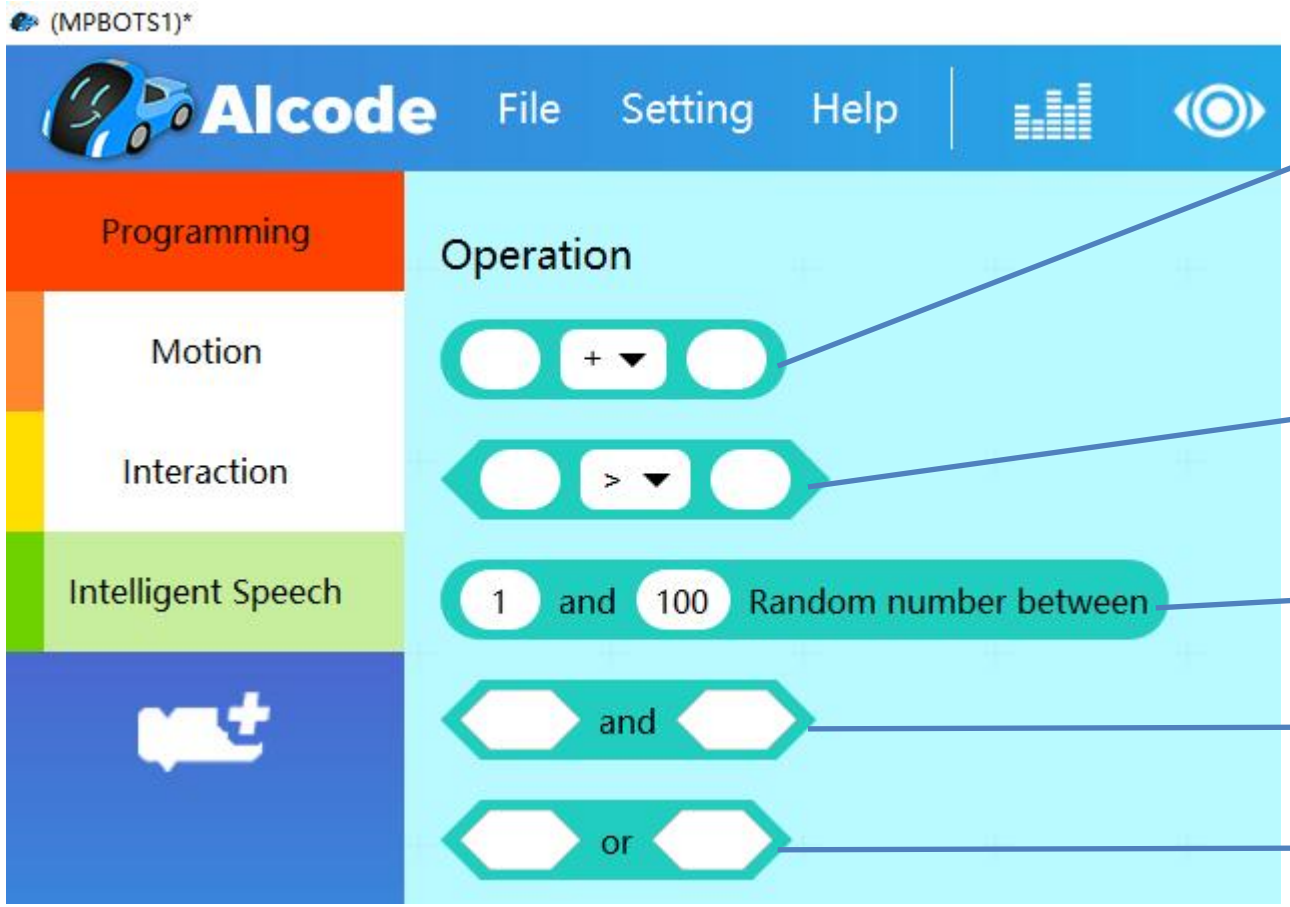
Port settings



Color correspondence



3.5 Programming Module -- Operations



Performing calculations on the data, such as addition, subtraction, multiplication, and division, is commonly used for numerical values from circular sensors.

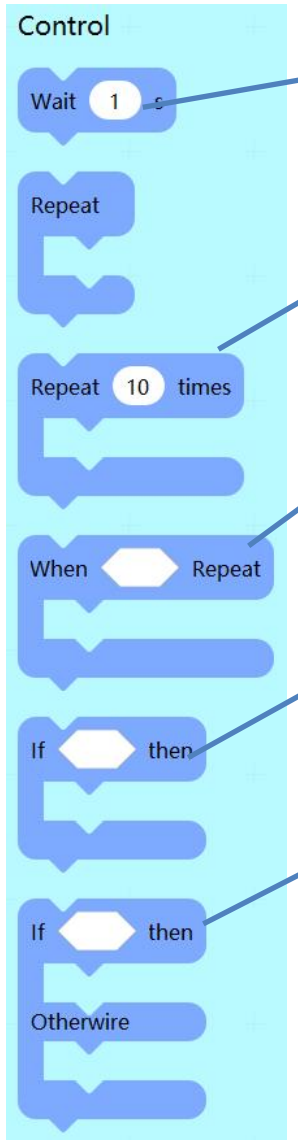
Compare data, comparing greater than, less than, equal to, and not equal to.

Random numbers

Both conditions must be met.

Either condition one or condition two are met.

3.6 Programming Module -- Conditions and Loops



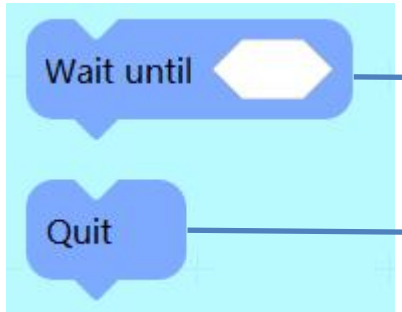
Infinitely repeated execution is typically paired with conditional statements and loop exit modules.

The loop iterates a finite number of times, then automatically exits after the maximum number of iterations, and continues execution of the next module.

A conditional loop, for example, loops only when the grayscale value is less than 100, but automatically exits the loop whenever the grayscale value is greater than 100.

If you make that judgment, note that it should only be performed once. When using it with a sensor, a loop is generally required.

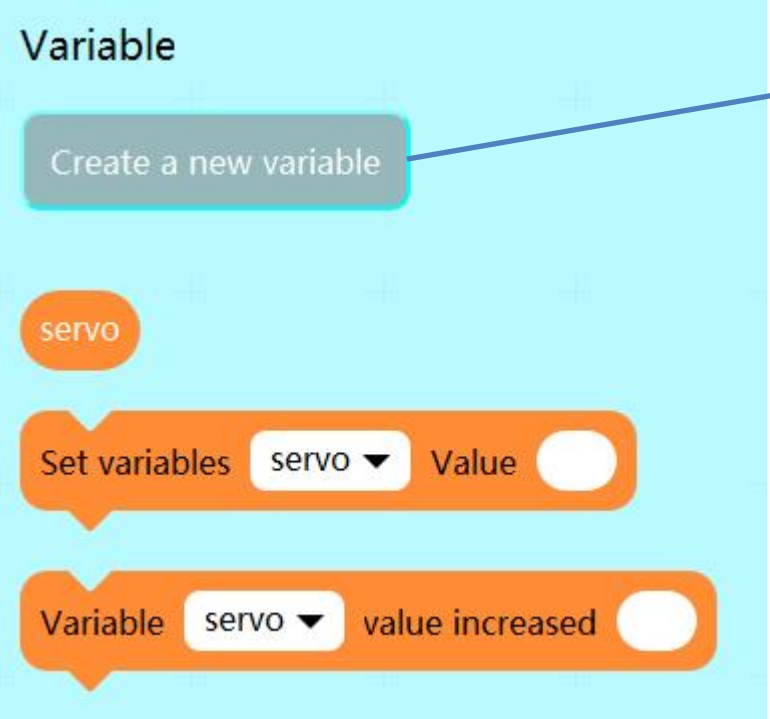
If so, then otherwise, there is an additional branching scenario.



Wait until, maintain the previous state until the condition is met before proceeding to the next step.

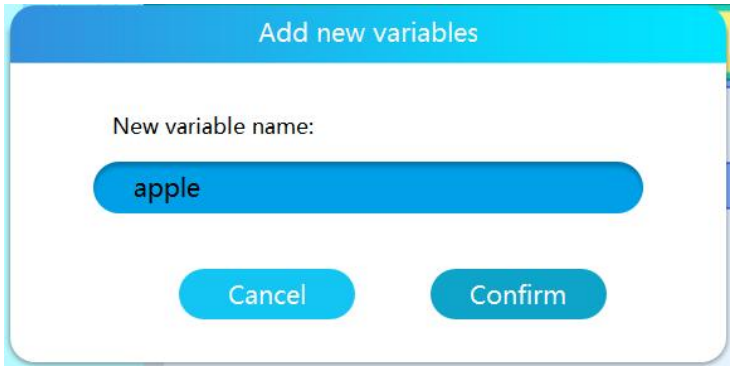
Exiting a loop usually occurs in conjunction with infinite repetition, and requires particular attention when there are nested loops.

3.7 Programming Module -- Variables

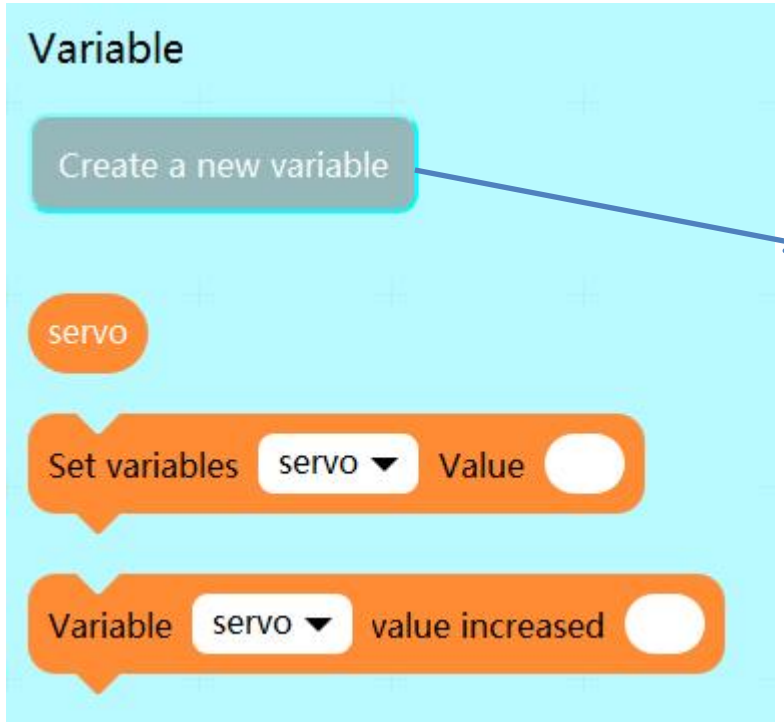


A variable can be viewed as an empty box. We need to name it to distinguish different variables, put things into it (assigning a value), and add or remove items from the box (modifying the variable's value).

For example, creating a variable called "apple" is like putting apples into the empty box and labeling it with the name "apple." Initially, we put in five apples, setting the variable's initial value to 5. Adding two more apples increases the variable's value by 2. The box now contains seven apples, and the variable's value is 7.

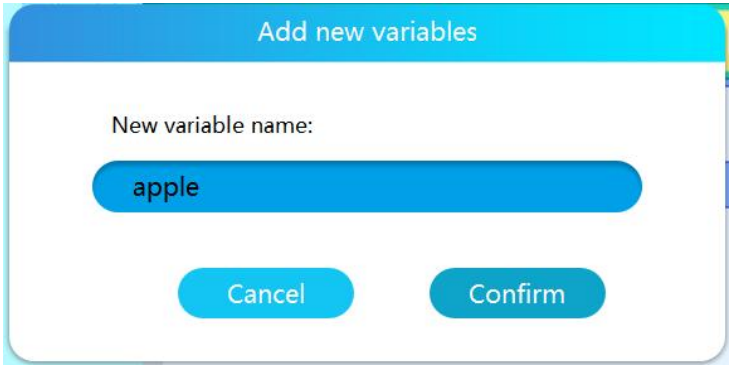


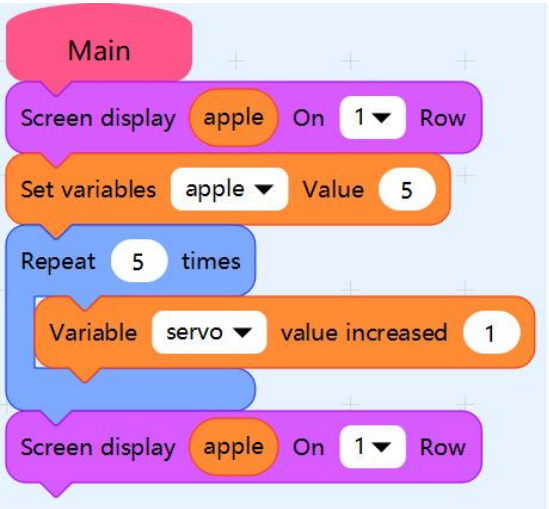
3.7 Programming Module -- Variables

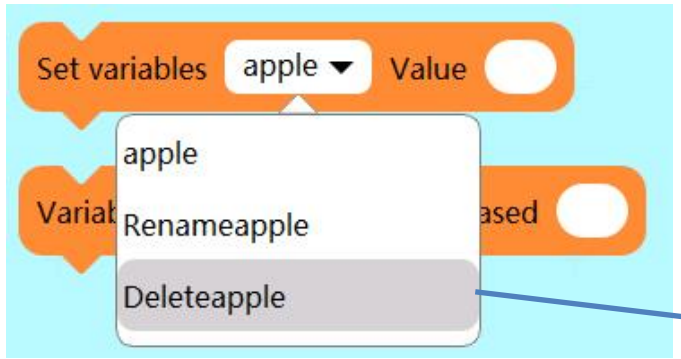


A variable can be viewed as an empty box. We need to name it to distinguish different variables, put things into it (assigning a value), and add or remove items from the box (modifying the variable's value).

For example, creating a variable called "apple" is like putting apples into an empty box and labeling them with the apple's name. Initially, we put five apples in, setting the variable's initial value to 5. Each time we add one more apple, we increase the variable's value by 1. After repeating this five times, the box now has 10 apples, and the variable's value is 10. We can also set values like servo motor angles and motor speeds as variables.

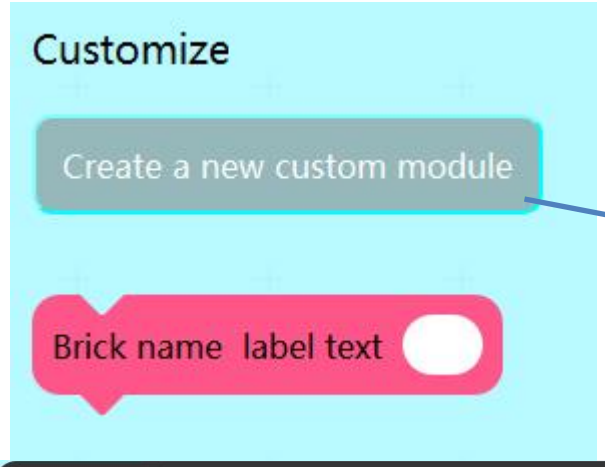




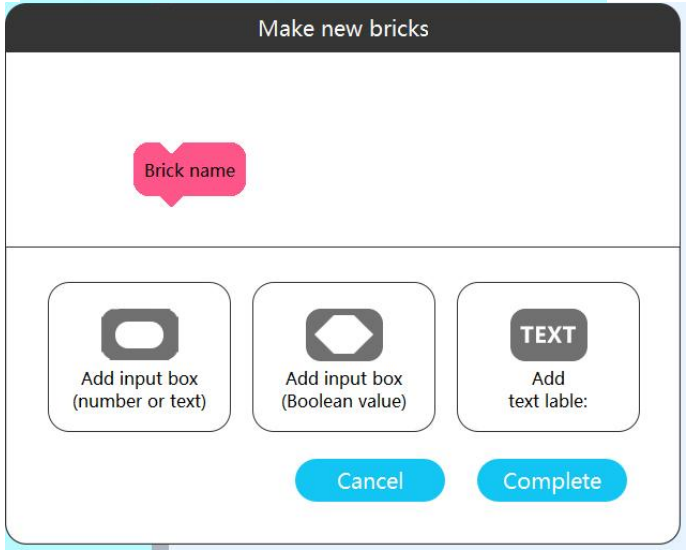


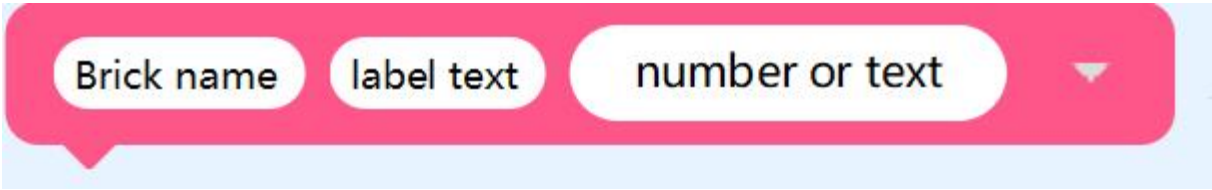
The name of a variable can be modified or deleted.

3.8 Programming Module -- Custom Blocks



If the existing modules don't have what you need, you can create your own, encapsulate a program written using other programming modules, or use it when breaking down a long program into segments.



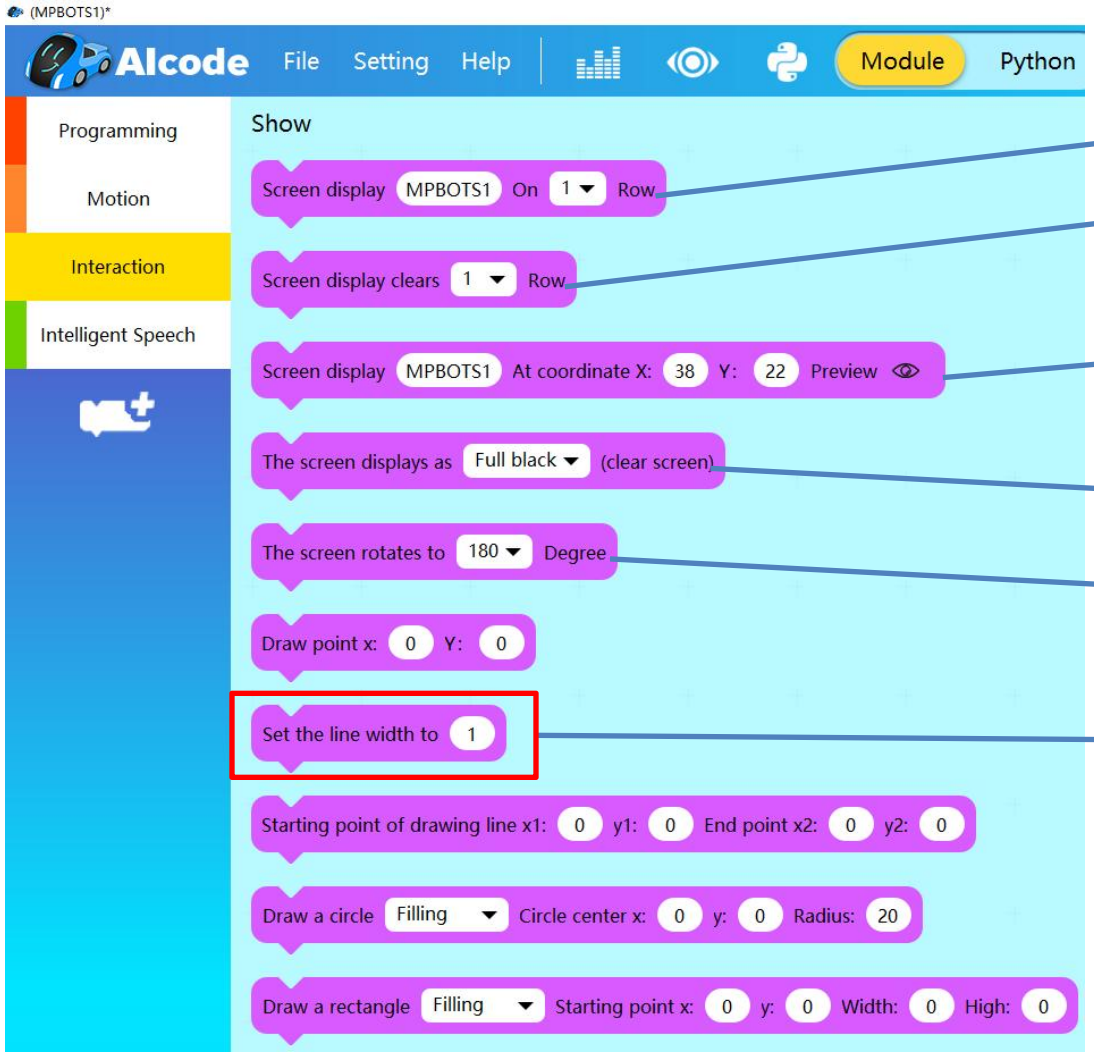


Here you can configure the program.

It can include module names, parameter names, parameter variables, etc., which can be freely combined.

It can be called after the main program.

3.9 Programming Module -- Display



The screen displays data, and you can select the number of rows.

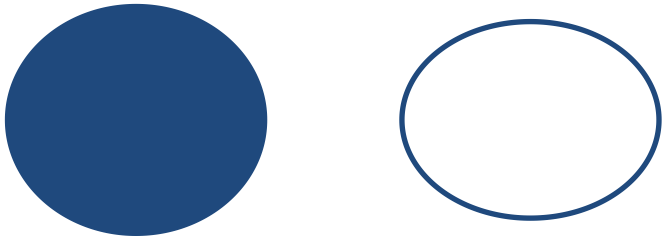
Clearing a line of the screen is crucial; otherwise, overlapping lines can occur, affecting the display quality.

The data is displayed on the screen according to coordinates, with a maximum horizontal coordinate of 128 and a maximum vertical coordinate of 64, divided into 5 rows.

Clear the entire screen; otherwise, overlapping images may appear, affecting the display quality.

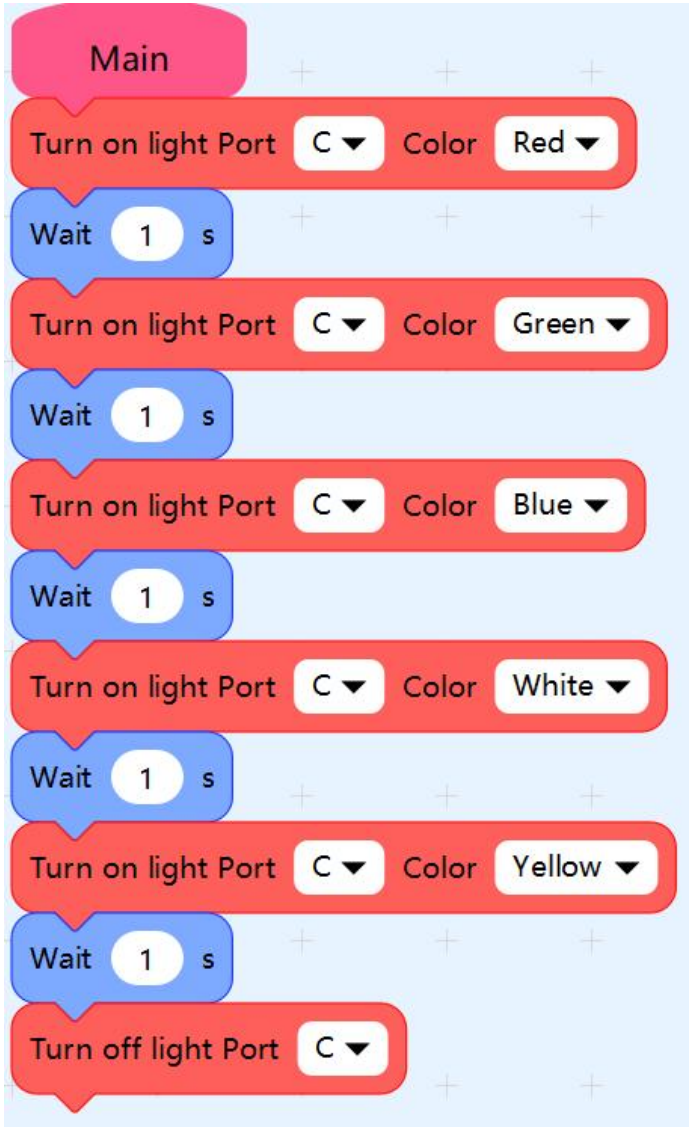
After setting this, all subsequent displays will rotate.

When drawing on the screen, be sure to set the brush thickness first; otherwise, you won't be able to distinguish between filling and not filling effects later.

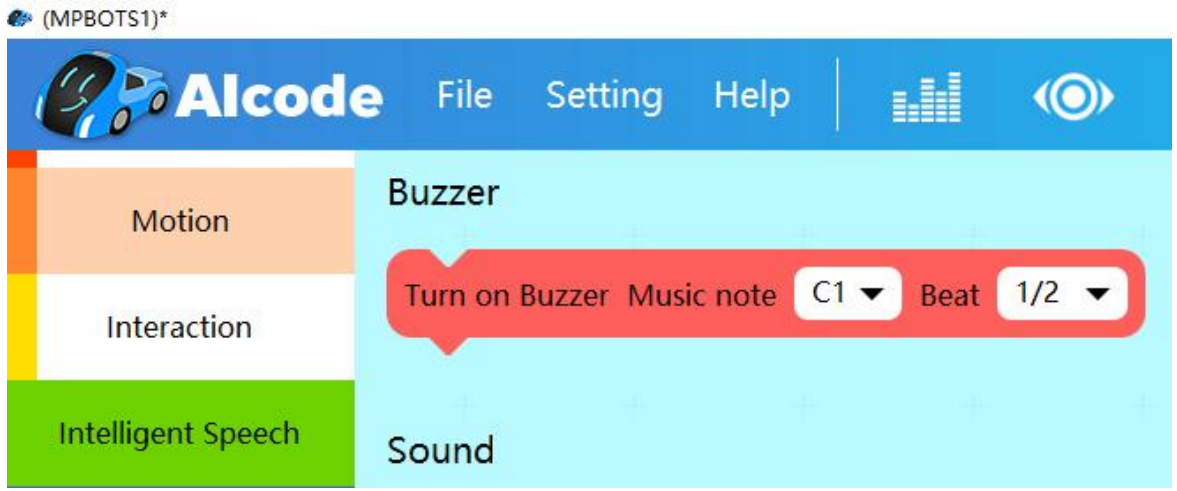


3.10 Programming Module -- Light

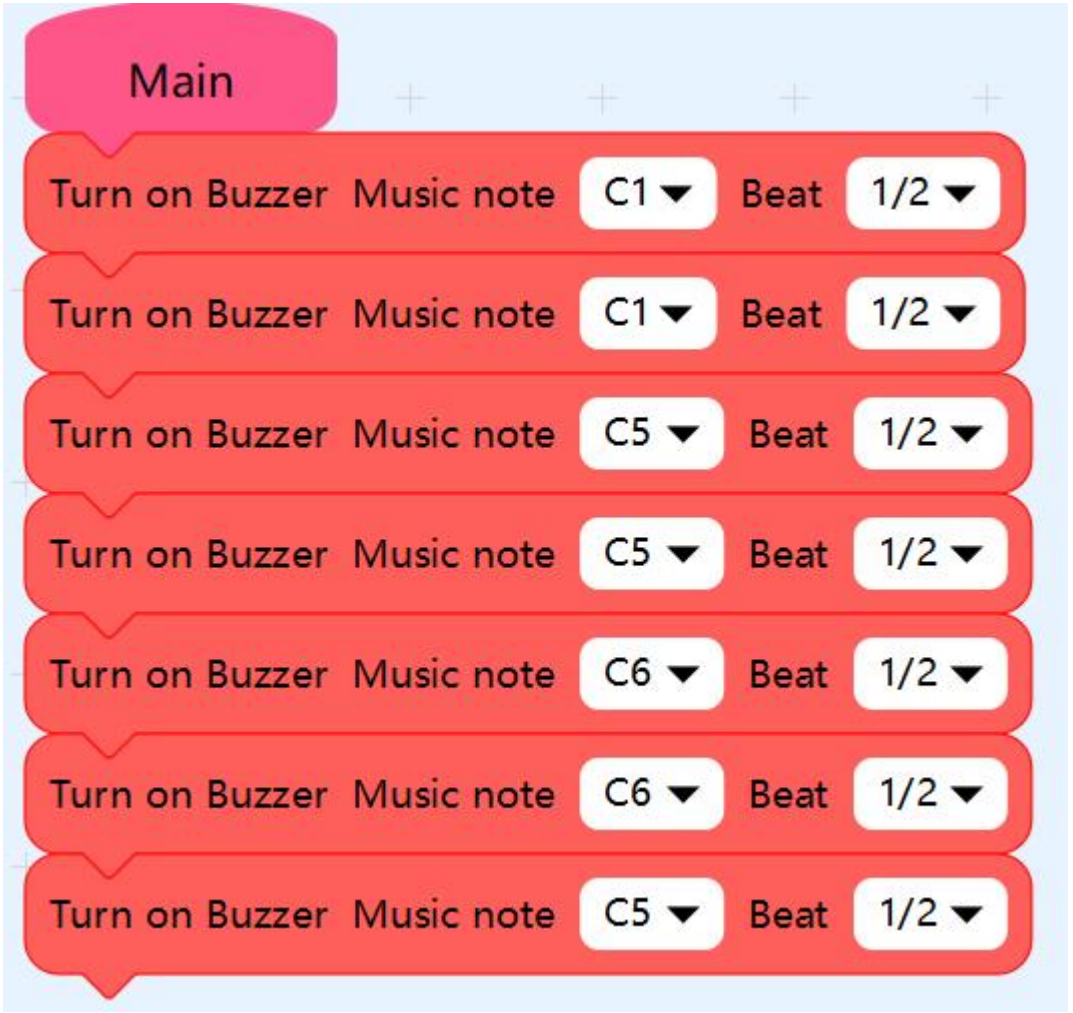
Different light colors can be set.



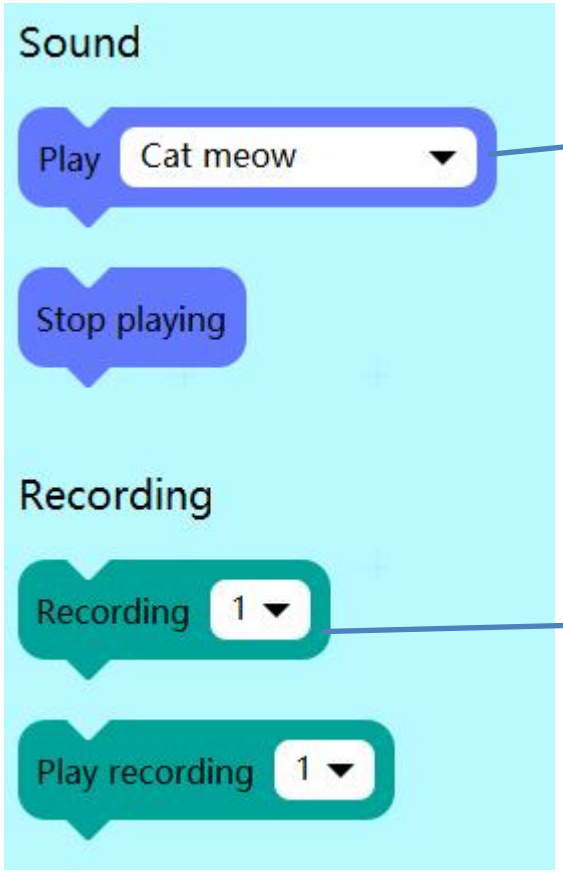
3.11 Programming Module - Buzzer



Different tones and rhythms can be written.

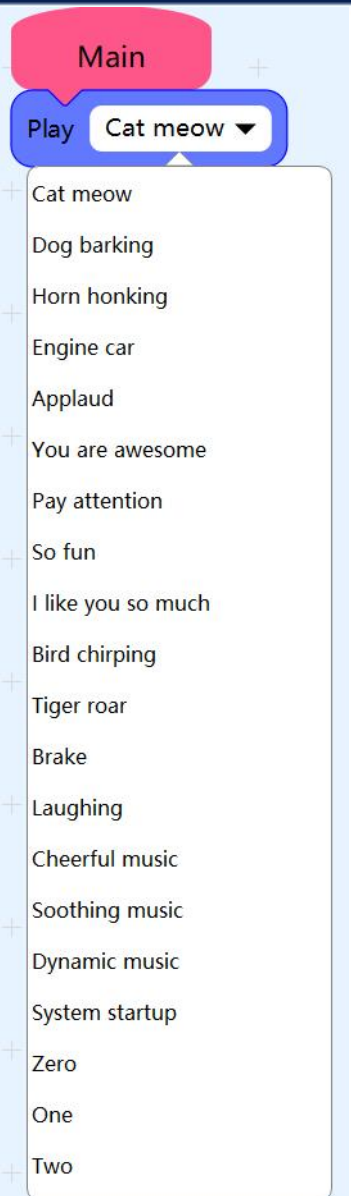


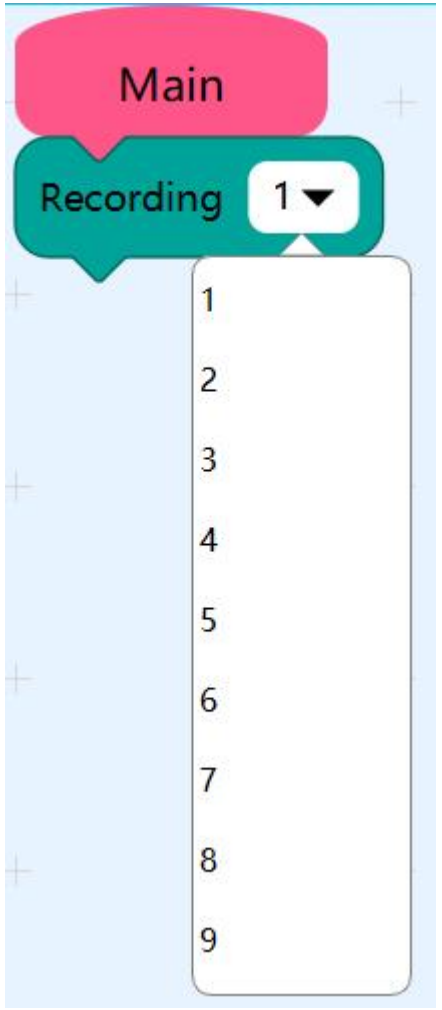
3.12 Programming Module -- Recording Module



Play the system's built-in sound library

Playing recorded sound library





The 5th Generation Educational Robot

— Artificial Intelligence Programming Robot

Thanks !