Index

Preface

The Manual introduces how to use Krypton controller and Abilix Chart 2.0 and offers some simple examples for your better understanding to make the use effectively. Please read through the whole manual after you have our product.

Abilix Chart 2.0 is an App for flow chart programming and greatly upgraded on the basis of Abilix chart 1.0. It supports wire and wireless download and makes the interaction more convenient. Software on PC end supports MAC/Windows.

Wish you a great experience!

1st Chapter Installation and First-time Use

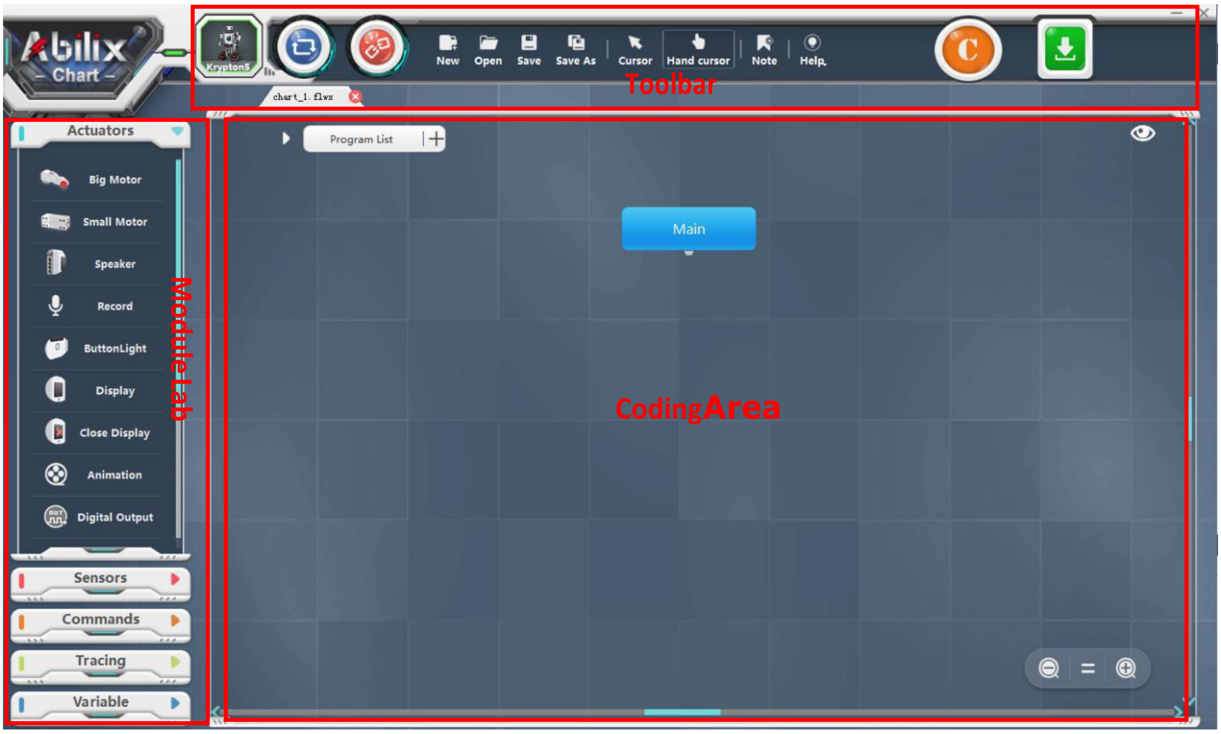
Abilix chart 2.0 is upgraded on the basis of 1.0 version and supports various types of Krypton products. Please select the correct type before use. The full name of Abilix chart is Abilix flowchart programming software. It supports standard flowchart and C language programming and is characterized by being easy for beginners to pick up and challenging for advanced learners to explore its potential functions.

1.1 Download and Installation

You can visit www.abilix.com and download installation package: Abilix chart Setup_X.X.X.X from “Tech Service/Material Download”, decompress the file after downloading and follow the instructions to complete the installation.

Installation details are as below: the default installation directory of Abilix Chart is in the root of C:\. You can alter installation directory according to the instructions during the procedure. A quick-start of “Abilix Chart” will be generated respectively on the Desk and in the Quick Launch after installation. Double tap the quick-start to open Abilix Chart.

1.2 Interface Introduction



Toolbar Icon List：

Icon	Function
🔁	Choose Type
🔗	Choose Controller
📄+	New
📂	Open
💾	Save
🗐	Save as
↖	Cursor
👆	Hand Cursor

⊕	**Zoom in**
⊖	**Zoom out**
✥	**Original**
C	**C editor**
👁	**Thumbnail**
🔖	**Notes**
💡	**Support**
⬇	**Download**

1.3 Connection

2 ways to connect Abilix chart to controller: wire and wireless connection. Proceed data communication and download programs after connecting successfully.

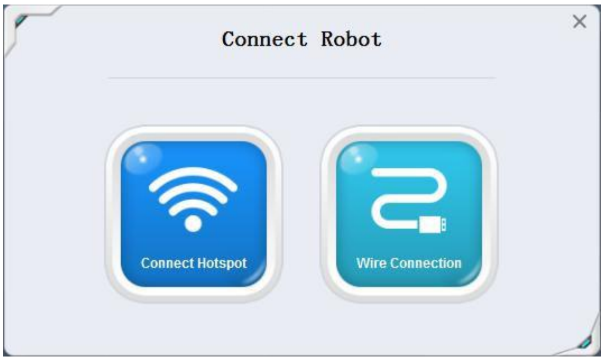
1.3.1 Krypton 0/1/2

Wire connection is applicable to Krypton 0/1/2:

1. **Open Abilix chart, tap** 🔁 **to select correct product type;**

2. **Use cable to connect controller to computer. (One end of the cable connects to a random sensor port of the controller and the other connects to USB port on the computer.)**

3. Tap 🔗 to choose connection method - > choose “wire connection”;

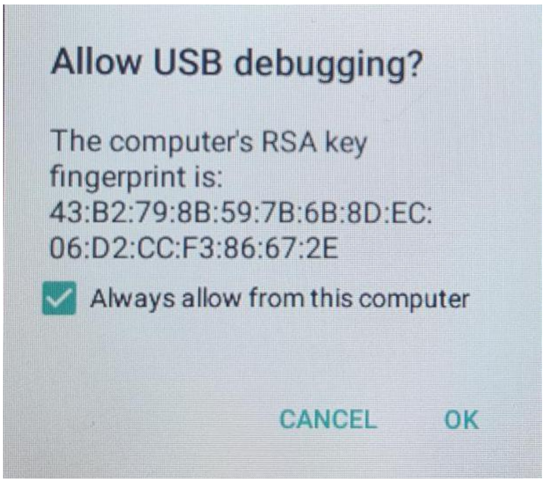


4. A prompt of “Connect Successfully” will pop up and 🔗 will turn 🔗.

1.3.2 Krypton 3 - Krypton 9

1. Wire connection

1. **Open Abilix chart, tap 🔄 to select correct product type;**
2. **Choose “USB debug - On” in controller “Setting” - “About”;**
3. **Use cable to connect controller to computer; (One end of the cable connects to CAM and the other connects to USB port on the computer.)**
4. **The following prompt will pop up at the first-time use, tick “Always allow to use this computer to debug”, press “OK”;**



Allow USB debugging?

The computer's RSA key fingerprint is:
43:B2:79:8B:59:7B:6B:8D:EC:
06:D2:CC:F3:86:67:2E

☑ Always allow from this computer

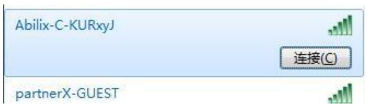
5. Tap 🔗 to choose connection method - > choose ‘‘Wire connection’’;

6. A prompt of ‘‘Connect successfully’’ will pop up and 🔗 will turn 🔗.

2. Wireless connection

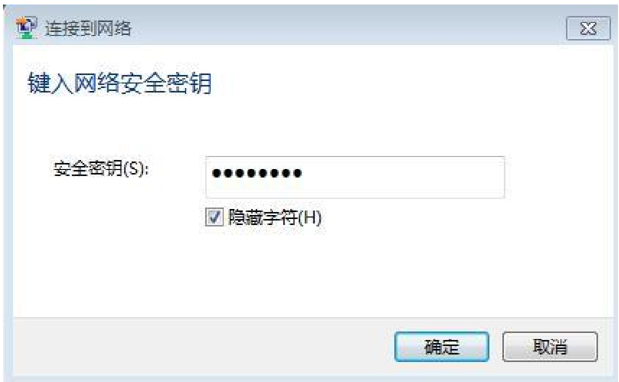
1. Turn on hotspot in controller ‘‘Setting’’ - ‘‘Wifi’’;

2. Turn on Wifi connection to find hotspot’s name and enter the password, tap ‘‘OK’’ to make computer join controller’s hotspot as below:



Abilix-C-KURxyJ

partnerX-GUEST



连接到网络

键入网络安全密钥

安全密钥(S): ●●●●●●●●●

☑ 隐藏字符(H)

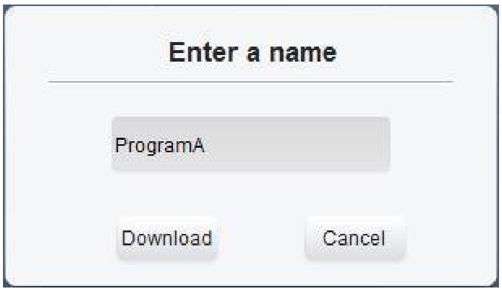
3. Tap 🔗 to choose connection method - > choose “Wireless connection”;

4. A prompt of “Connect successfully” will pop up and 🔗 will turn 🔗.

Notes: If there is a prompt of “Connection Failed”, please check whether robot’s hotspot and networks work normally; you also can turn off firewall on computer end and anti-virus software as well as forbid wire connection (The default of firewall forbids wire connection.) to try again.

1.4 Program Download

1. Abilix chart has connected to Krypton 9 controller successfully;
2. Tap 📥 to see a pop-up interface to modify program name, the default name is Program A;



Enter a name

ProgramA

3. Tap “OK”; if program is downloaded to the controller, a prompt of “Download successful” will appear.

Krypton 9 controller can save multi user programs simultaneously. Customized program name is allowed and requires a combination of 1-10 letters, digits or underlines, or no more than 5 Chinese characters. Corresponding program name will appear on the screen.

2nd Chapter Abilix Chart Module Lab

A robot consists of controller, sensor, actuator and programs. Sensor and actuator are both connected to different ports on the controller while the controller also contains inherent sensors, such as: gyroscope. Programs will run and collect returned value from controller ports. After computing, commands are sent to actuator to facilitate motions through dedicated ports.

Abilix Chart Module Lab is categorized into actuator, sensor, command, tracing and variable module lab.

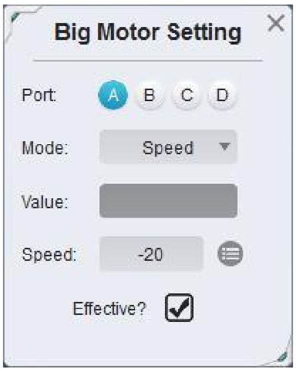
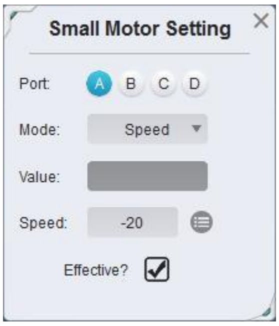
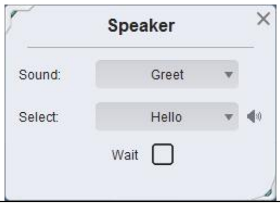
2.1 Main

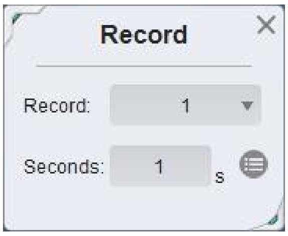
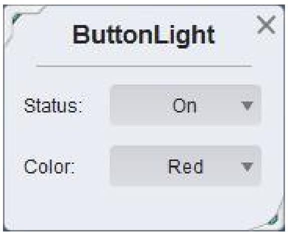
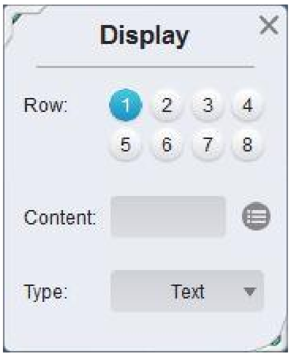
Main **is in editing area and can not be deleted. All modules must be dragged below "Main" to be downloaded to the controller.**

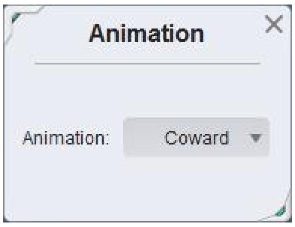
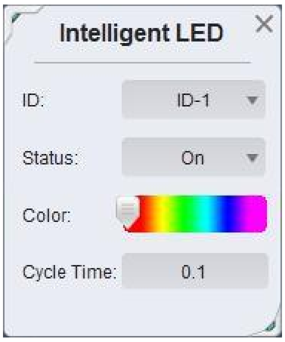
2.2 Actuator modulelab

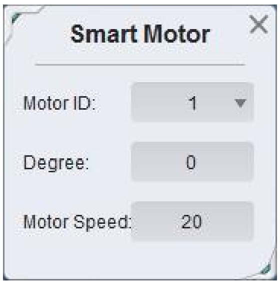
Actuator module lab includes big motor, small motor, speaker, LED, intelligent LED and motor, etc to respond to controller's commands to make corresponding actions. Name, parameter and function of actuator modules are as below:

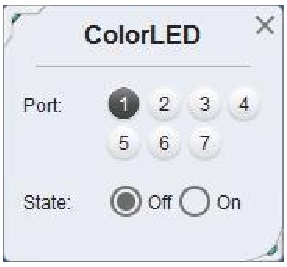
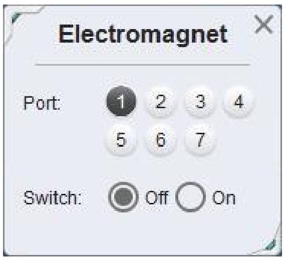
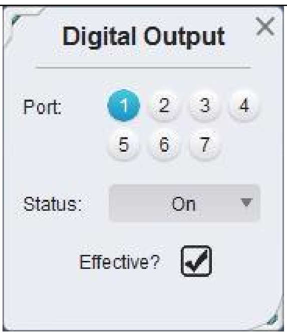
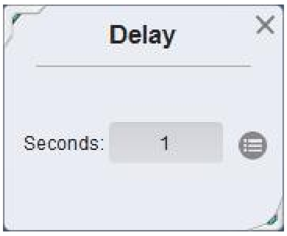
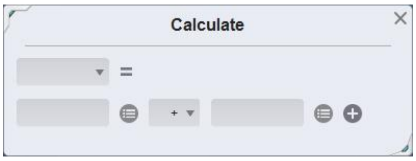
No	Name	Parameter	Function

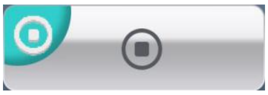
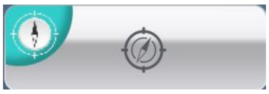
1	Big Motor	 Big Motor Setting Port: A B C D Mode: Speed Value: Speed: -20 Effective? ☑	Port: Connect the motor to a certain port ofthe controller; Mode: Decide speed, degree, loop and time of the motor; Value: Value of degree, loop and time. If user chooses “Loop” in mode, the value refers tothe loop that motor rotates; (When speed is chosen, the value doesn’t need to be set.) Speed: Speed of rotation, ranging in -100~100; Effective: When ticked, the corresponding port will be on; If A and B ports are both ticked, 2 motors are activated simultaneously.
2	Small Motor	 Small Motor Setting Port: A B C D Mode: Speed Value: Speed: -20 Effective? ☑	Port: Connect the motor to a certain port ofthe controller; Mode: Decide speed, degree, loop and time of the motor; Value: Value of degree, loop and time. If user chooses “Loop” in mode, the value refers tothe loop that motor rotates; (When speed is chosen, the value doesn’t need to be set.) Speed: Speed of rotation, ranging in -100~100; Effective: When ticked, the corresponding port will be on; If A and B ports are both ticked, 2 motors are activated simultaneously.
3	Speaker	 Speaker Sound: Greet Select: Hello Wait □	Speaker is used to play music. Sound type: 8 categories include greeting, animal, musical instrument, etc;

			Choose sound: choose different types of sounds; Wait: If "wait" is ticked, the next sentence will be executed after playing the current audio.
4	**Record**	 Record Record: 1 Seconds: 1 s	Turn on "Record". Record: Set the name of the recording; Seconds: Set the time of recording. Tap the corresponding name of the recording to play in the controller "Multi Media" -> "Record" after the recording is finished.
5	**Button LED**	 ButtonLight Status: On Color: Red	Set the button LED of controller. Status: Turn on/off button LED; Color: Set LED color: Red, Blue, Green.
6	**Display**	 Display Row: 1 2 3 4 5 6 7 8 Content: Type: Text	Display text or image on the controller screen. Type: text, image, animation and custom, 4 types in total. When choosing "text", characters or sensor's variables are displayed: Row: It displays in which row the chosen content lies and each row displays a maximum of 20 characters. Content: Manually enter characters or select variables. Tap ⌨ or ☰ on the right side of the enter frame to switch. Variable represents the returned value of sensor or customized value. When choosing "image", shot photos when

			program is running will display; Choose image: Decide an image for display through ID. ID of image is decided by “Camera” and can be checked in controller “Multi Media” interface. When choosing “animation”, emoticon will display. When choosing “custom”, lead in images from local disk. A single image can not exceed 2M, R.P. is recommended in 320*320 and a maximum of 10 images can be led in.
7	Close Display		Close “Display” of controller.
8	Animation	 Animation Animation: Coward	Play inherent animation. Animation: coward, happy, cheer up, lovely, cry, aggrieved, blink.
9	Intelligent LED	 Intelligent LED ID: ID-1 Status: On Color: Cycle Time: 0.1	ID: ID of intelligent LED Status: Status of intelligent LED Color: Choose the color of intelligent LED; Cycle time: Decide the cycle of “Breathe” or “Flash”, ranging in 0.1s - 60s; and only under the status of “Breathe” or “Flash” does the value work.

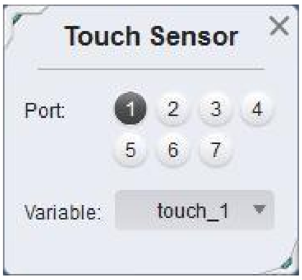
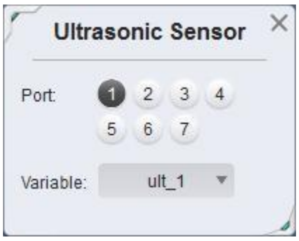
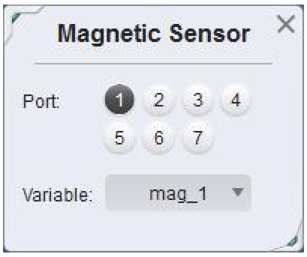
10	Smart Motor	
		Smart Motor
		Motor ID: 1
		Degree: 0
		Motor Speed: 20
			Motor ID: ID of smart motor;
			Degree: Degree of rotation;
			Speed: Speed of rotation.

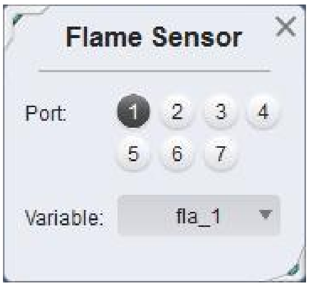
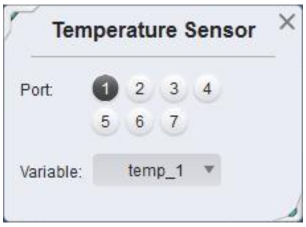
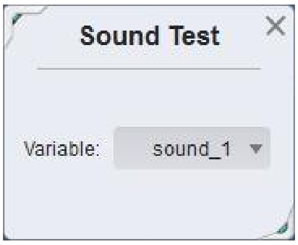
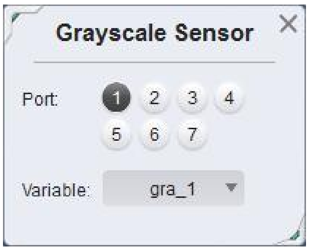
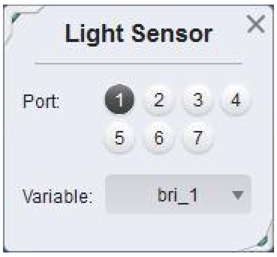
11	**Color LED**	 ColorLED Port: ● 1 2 3 4 5 6 7 State: ◉ Off ○ On	**Port: Port No. of color LED connecting to the controller;** **Switch: Turn on/off port of the controller. Color LED will be lit up when being inserted into the port which is "on" All sensor ports of the controller are "on" as default. Use "Switch" to control "On/Off" of port.**
12	**Electromagnet**	 Electromagnet Port: ● 1 2 3 4 5 6 7 Switch: ◉ Off ○ On	**Port: Choose Port No. of the controller to connect electromagnet;** **Switch: Turn on or off port of the controller. Magnetism will be generated when inserting electromagnet into the port which is "on".**
13	**Digit output**	 Digital Output Port: ● 1 2 3 4 5 6 7 Status: On ▾ Effective? ☑	**DO of corresponding I/O port of the controller, all actuators can be controlled by this module.** **Port: Choose Port No. of the controller to connect sensor;** **Switch: Turn on or off port of the controller.**
14	**Delay**	 Delay Seconds: 1	**Wait for corresponding time and controller does not perform any motions in the duration. Parameter is the time for waiting with a unit of "second" and a range in 0.00-60.00.**
15	**Calculate**	 Calculate ▾ = + ▾	**Support addition, subtraction, multiplication and division.** **You can enter Numbers or select variables to perform operations, and to switch by tapping ☰ or ⌨ on the right side of the enter frame。**

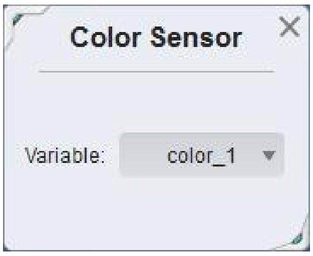
16	**Stop “Play”**		**Stop playing the audio.**
17	**Calibrate compass**		**As instructed, rotate the controller in 8-shaped route to calibrate compass.**

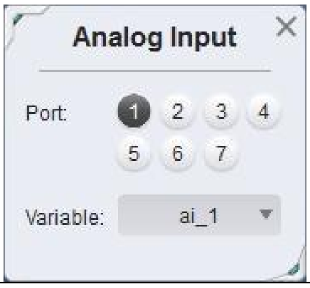
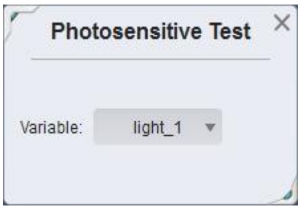
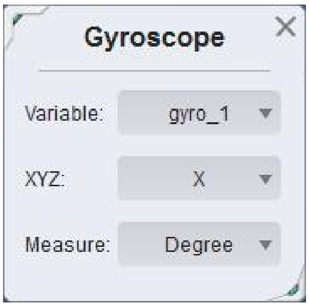
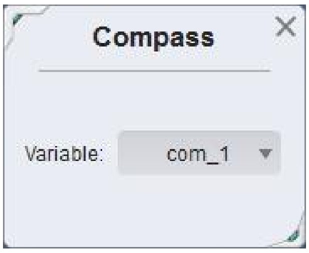
2.3 Sensor Module Lab

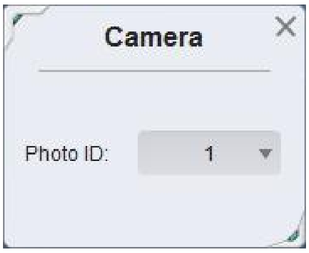
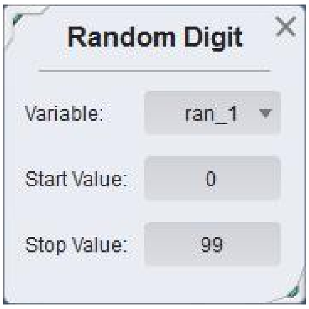
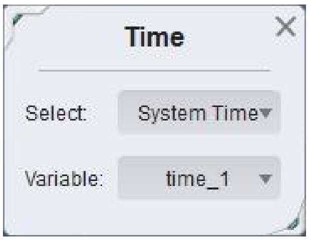
Sensor module lab includes all modules collecting environment data for the controller. Controller can collect environment data by AI, DI and counting functions of port. Name, parameter and function of sensor modules are as below:

No	Name	Parameter	Function
1	**Touch Sensor**	 Touch Sensor Port: 1 2 3 4 5 6 7 Variable: touch_1	**Sensor is used to detect touch and Return the following values: 0:button wasn't pressed; 1:button was pressed;** **Port: Choose port to connect the sensor;** **Variable: Variable of the sensor.** **(In following module, the meanings of "port" and "variable" are the same.)**
2	**Ultrasonic sensor**	 Ultrasonic Sensor Port: 1 2 3 4 5 6 7 Variable: ult_1	**Measure the distance between obstacle and sensor by ultrasonic distance-measuring, ranging from 50mm to 1500mm.** **Returned value of sensor ranges from 50 to 1500. The longer the distance is, the larger the value will be.**
3	**Magnetic sensor**	 Magnetic Sensor Port: 1 2 3 4 5 6 7 Variable: mag_1	**Sensor is used to measure magnetic field, ranging from 0 to 4095. When magnetic sensor approaches magnet (about 1.7cm), the returned value is larger than 1000; when the sensor is far away from magnet, the returned value is less than 10.**

4	Flame sensor	 Flame Sensor Port: ● 1 2 3 4 5 6 7 Variable: fla_1	Sensor is used to measure the intensity of light source (e.g. flame). Returned values ranges from 0 to 4095, the closer the sensor gets to the light source, the larger the returned value will be.
5	Temperature Sensor	 Temperature Sensor Port: ● 1 2 3 4 5 6 7 Variable: temp_1	Sensor is used to measure the temperature, ranging from 0℃ to 70℃ Returned value ranges from 0 to 4095. The higher the temperature is, the larger the value will be; and vice versa.
6	Sound Test	 Sound Test Variable: sound_1	Sensor is used to test sound strength of the surrounding, with a returned value ranging from 0 to 100.
7	Grayscale sensor	 Grayscale Sensor Port: ● 1 2 3 4 5 6 7 Variable: gra_1	The module is used to collect data of grayscale sensor. Returned value ranges from 0 to 4095. Returned value of white is small while the darker the color is, the larger the returned value will be.
8	Photosensitive sensor	 Light Sensor Port: ● 1 2 3 4 5 6 7 Variable: bri_1	It is used to test the light intensity. Returned value ranges from 0 to 4095, the brighter the light is, the larger the value will be.

9	Color sensor	
		Color Sensor
		Variable: color_1
			It is used to collect color data.
			The tested color corresponds to the following returned value:
			Red: 0; Yellow: 1; Green: 2; Blue: 3;
			White: 4;

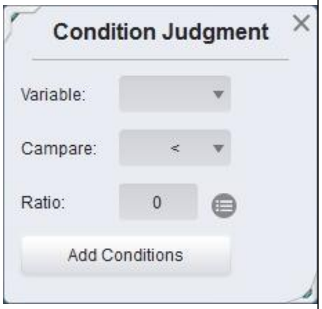
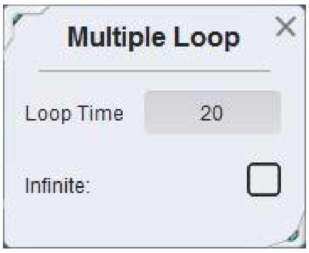
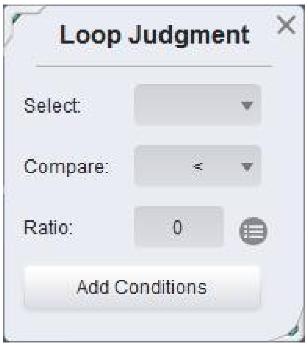
10	Analog input	 Analog Input Port: 1 2 3 4 5 6 7 Variable: ai_1	Read AI of corresponding port of the controller and collect data (Touch sensor, grayscale sensor and ultrasonic sensor are all analog sensors.)
11	Photosensitive test	 Photosensitive Test Variable: light_1	Use color sensor to test light intensity, ranging in 0-255; the brighter the light is, the larger the returned value will be.
12	Gyroscope	 Gyroscope Variable: gyro_1 XYZ: X Measure: Degree	Collect corresponding data from gyroscope in the controller. X: Obtain the degree of looking up and stooping down, ranging in -180~180 degrees; Y: Obtain the degree of tilting left and right, ranging in -90~90 degrees; Z: When controller faces right side up, obtain the degree of rotating clockwise and counter-clockwise, ranging in 0~359 degrees.
13	Compass	 Compass Variable: com_1	It is used to detect the head of controller is pointing to what direction. The returned value ranges from 0 to 359 degrees, 0 degree indicates North; 90, East; 180, South; 270, West.

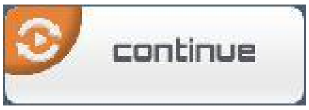
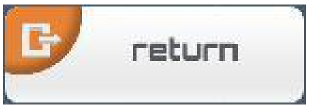
14	Camera	 Camera Photo ID: 1	Take photos through “Camera” and the sensor is connected to CAM port. Photo ID: Set name of the taken photo. When “Camera” module is run, “Camera” is on and all taken photos are saved in controller “Multi Media” - > “Photo”. Long press the photo to delete.
15	Random digit	 Random Digit Variable: ran_1 Start Value: 0 Stop Value: 99	Randomly generate a digit and assign it a variable. “Start value” and “End value” confine the range of random digit.
16	Time	 Time Select: System Time Variable: time_1	System Time: Obtain the time from program running till now. Reset time: Set the time as 0.

2.4 Command Module Lab

To read returned value from all ports is usually used for storage and judgement (in most scenarios). There are 3 judgement methods, including “while” sentence, “if...else” sentence and “for” sentence, in Chart and Command Module Lab. Name, parameter and function of sensor modules are as below:

No	Name	Parameter	Function

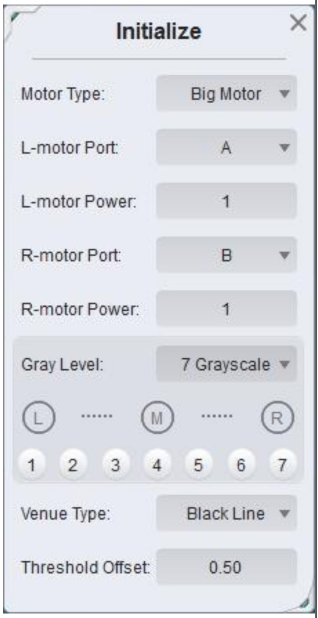
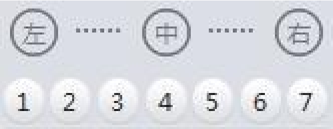
1	Condition Judgement	 Condition Judgment Variable: Campare: < Ratio: 0 Add Conditions	“If...else” sentence in C language. If the condition is met, perform the left branch sentence; otherwise, perform the right branch sentence. Choose variable: Choose different variables. Comparison: >，<，!=，== Comparison value: Input value or choose variable. Tap “Add condition” to add judgement condition and parameter remains the same as previous. Logical relationship between two judgement conditions can be && or ||.
2	Multi loop	 Multiple Loop Loop Time 20 Infinite: □	For sentence of C language. The sentence in loop will run for designated loop. When pressing “Infinite loop”, module will become “While（1）” sentence and the sentence in loop will keep performing repetitively.
3	Loop judgement	 Loop Judgment Select: Compare: < Ratio: 0 Add Conditions	“While” sentence in C language. Parameter is condition set by user by functioning in a situation of performing branch sentence repetitively if condition is met while jumping out of loop to perform the next sentence if condition is not met.
4	Break	 break	“Break” sentence in C language without parameter and within loop. When performing the module, exit the current

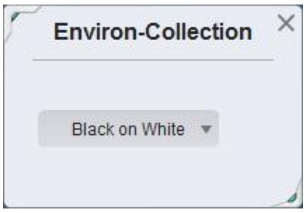
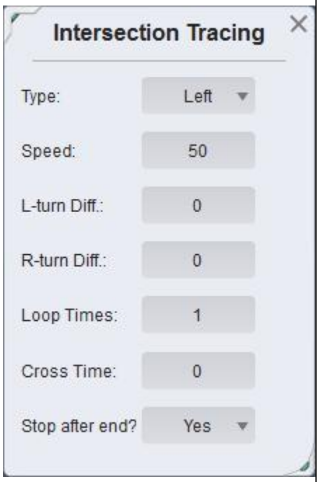
			loop.
5	Continue loop		"Continue" sentence in C language without parameter and within loop. When performing the module, start from the first-row sentence in loop.
6	Return		"Return" sentence in C language without parameter. Go back to the starting point and do not perform the program after "Return".

2.5 Tracing Module Lab

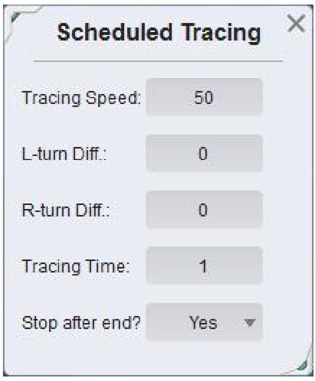
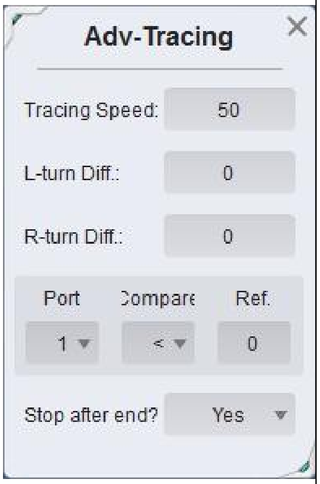
7 tracing modules, including initialization, environment collection, intersection tracing, scheduled tracing, advanced tracing, swerving and engine starting, are used to program when vehicle is doing tracing by ground grayscale. Grayscale sensors (usually 5 or 7) should be installed in the front of the vehicle in a row when using. Name, parameter and function of sensor modules are as below:

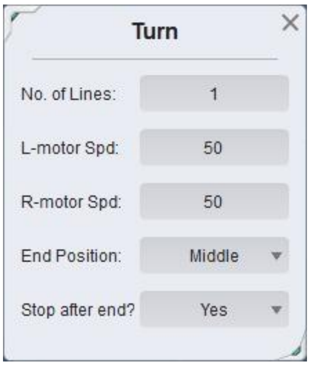
No	Name	Parameter	Function

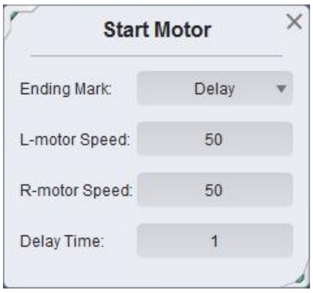
1	Initialize	 Initialize Motor Type: Big Motor L-motor Port: A L-motor Power: 1 R-motor Port: B R-motor Power: 1 Gray Level: 7 Grayscale Venue Type: Black Line Threshold Offset: 0.50	**Motor type: Choose big motor or small motor;** **L/R motor port: A-D are ports of controller, indicating L/R motors are connected to corresponding ports of the controller respectively.** **L/R motor power: Set motor power. The output speed of motor is speed value times power value (ranging in -1~1). if motor rotates clockwise and robot moves backward, set the power value negative.** **Grayscale number: Number of grayscale sensors. If vehicle uses 7 grayscale sensors, please choose “7 grayscale”; if vehicle uses 5 grayscale sensors, please choose “5 grayscale”; if vehicle uses neither 7 nor 5 grayscale sensors, user should write tracing program by him/herself.**  **indicates grayscale sensor should be connected to a certain I/O of controller. If “7 grayscale” is chosen, grayscale sensor should be connected to port 1-7 of controller from left side to right side.** **Venue type: if line on the venue (white) is black, please choose “black line”; if line on the venue (black) is white, please**

			choose “white line”. In here, “white” and “black” refer to grayscale sensor. If glass is transparent, grayscale sensor regards glass as “black”. Threshold deviation: Usually the critical value is (large value + small value)*0.5”, “0.5” is threshold deviation ratio. Under some circumstances, the threshold needs to lean to the smaller value, then, reduce “0.5”. Do not reduce it to a tiny value. For instance: “(2+8)*0.5” = 5, if “0.5” is reduced to “0.1”, the value is meaningless.
2	Environment-collection	 Environ-Collection Black on White	The module is used to collect grayscale threshold. If robot is running on a certain type of venue, choose a corresponding type only. If robot is running on both types of venues, choose 2 types. There must be an initialization before environment-collection. Please make reference to 5-1.
3	Intersection tracing	 Intersection Tracing Type: Left Speed: 50 L-turn Diff.: 0 R-turn Diff.: 0 Loop Times: 1 Cross Time: 0 Stop after end? Yes	Intersection type: The pending-to-cross intersection is on either left side or right side of the vehicle. Choosing intersection type should make reference to following 2 conditions based on priorities: 1) Keep aligned with the next swerve; 2) If there is not a next swerve, choose the side with less intersections.

			Tracing speed: When central grayscale is on the line, the motor with a higher speed (ranging from 10 to 100) out of two will limit the highest speed of the vehicle. **L/wheel differential: Set the vehicle's speed. When central grayscale is on the line, L/wheel speed = (tracing speed - L/wheel differential) x power value.** **R/wheel differential: Set the vehicle's speed. When the central grayscale is on the line, R/wheel differential = (tracing speed - R/wheel differential) x power value. If tracing speed is 100, set L/R wheel differential as 30 respectively, the vehicle's speed is (70 x power value). When L/R wheel differentials are different, adjust L/R wheel differentials to remain the same.** **Loop times: How many intersections are pending to cross.Intersection tracing is to cross an intersection to finish a complete loop, and loop times is to cross how many intersections.** **Time to cross the intersection: How long does the vehicle keep proceeding after detecting the intersection. (The time is decided by the concrete venue condition.)**

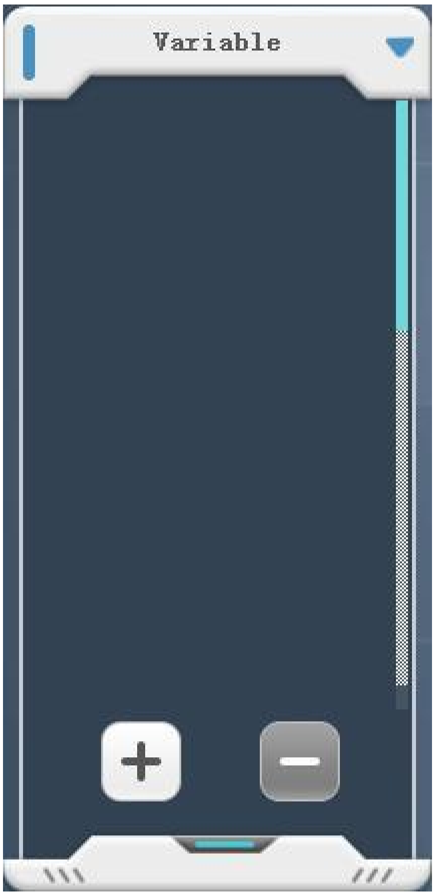
			Stop after end: Choose whether to stop the robot after completing the module.
4	Scheduled tracing	 Scheduled Tracing Tracing Speed: 50 L-turn Diff.: 0 R-turn Diff.: 0 Tracing Time: 1 Stop after end? Yes	When there is no intersection or reference, make use of scheduled tracing to control the proceeding distance by time. The proceeding distance is decided by speed and time. There are various factors impacting the speed, such as: parameter, voltage, resistance, etc, among which some are irresistible. Therefore, the proceeding distance can not be 100% precisely controlled.
5	Advanced tracing	 Adv-Tracing Tracing Speed: 50 L-turn Diff.: 0 R-turn Diff.: 0 Port: 1 Compare: < Ref.: 0 Stop after end? Yes	Ending mark is the returned value reaches the premeditated requirement, for instance: a grayscale sensor connects to a certain I/O port (port 1) and the returned value is 3000 when no object is detected; when the returned value is 500, an object is measured, then the "stop" stipulates: "port 1 < 500". Tracing speed, L/R wheel differential: the same as what has been presented in "intersection tracing". Port: Port No. of the sensor to stop the module. Comparison symbol: operating symbol, which is decided by changes of returned

			value. Reference value: corresponding threshold value to stop tracing module.
6	Turn	 Turn No. of Lines: 1 L-motor Spd: 50 R-motor Spd: 50 End Position: Middle Stop after end? Yes	Motion of the robot is controlled by 2 motors and a swerve is completed by 2 motors' differential. If speed is regarded as a value (reverse speed - negative), the robot will swerve to the side with a smaller speed. Ending mark of a swerve is parts of the grayscale sensor cross a certain line in order. Number of crossed lines: How many lines are crossed when swerving. If a 90-degree turning needs to cross a line at the crossroads, a 180-degree turning must cross 2 lines. L/R motor speed: Set L/R motor speed. Ending position: It decides whether the robot will stop at the "center", "left-of-center", or "right-of-center". Robot has inertia and sometimes overturns, therefore, choose "left-of-center" when turning left at a high speed. Finally, robot is supposed to be in a stable status after completing the swerve.

7	**Start motor**	 Ending Mark: Delay L-motor Speed: 50 R-motor Speed: 50 Delay Time: 1	**The module is used to control the engine, divided into a control by time or acontrol by sensor. If it is controlled by time, robot stops after the module ends; if it is controlled by sensor, choose whether to stop robot after the module ends. The condition to end the module lies in whether the returned value of sensor meets the setting.**

2.6 Variable Module Lab

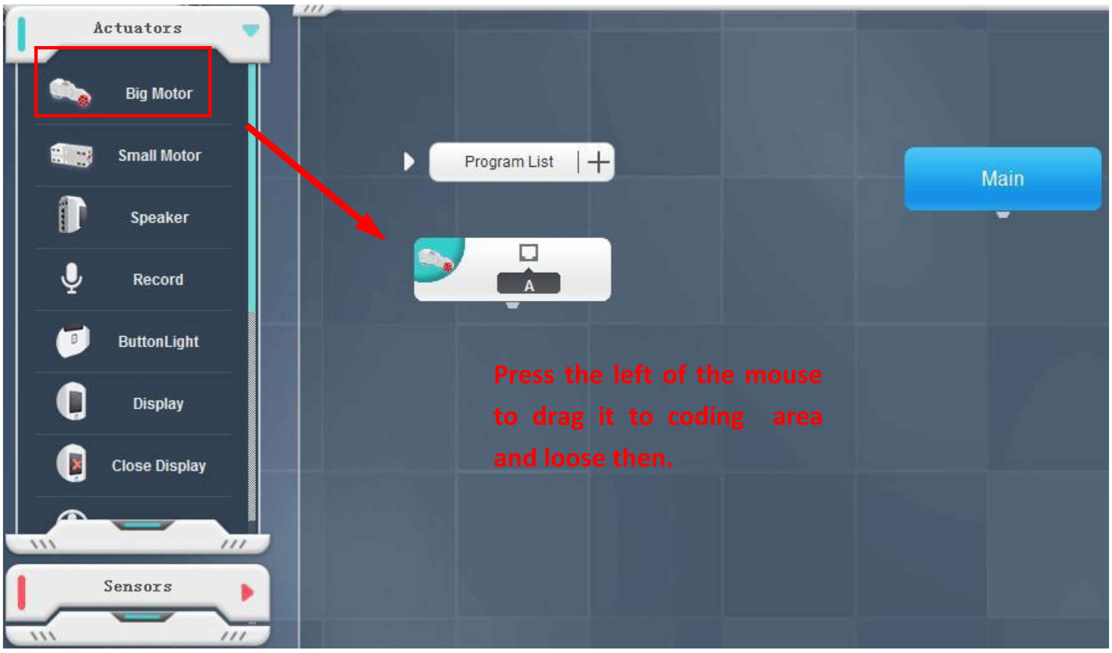
Variable module lab can record generated global variables, or perform a function of customized variables.



3rd Chapter Program Coding

3.1 Move module

Use the left side of cursor to choose an icon on the left side and drag it to the coding area on the right side. The module can be modified parameter to connected to other modules.



Press the left of the mouse to drag it to coding area and loose then.

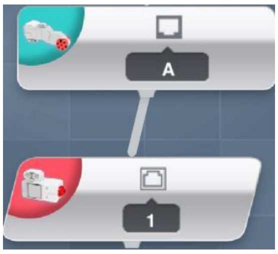
3.1.1 Move module and background

Move module: Place cursor on the module and press the left side of cursor to move around and drag module.

Move background: Place the hand cursor at blank coding area and press the left side of cursor (which has turned to be a fist) to move around and move background. Mouse wheel also can be used to move background.

3.1.2 Connect 2 modules

Drag module to stay close to the connecting point of another module, when a line is generated between 2 connecting points, the 2 modules can be connected. For instance:



2 modules will be connected to each other after loosing the cursor. The line will break if a module is dragged away.

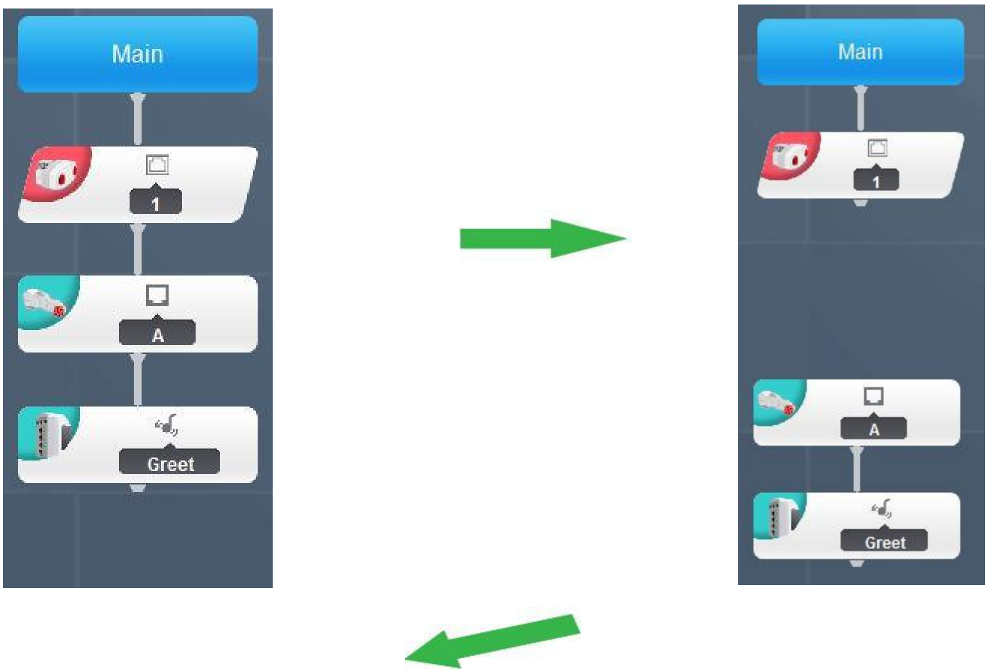
3.1.3 Insert a module

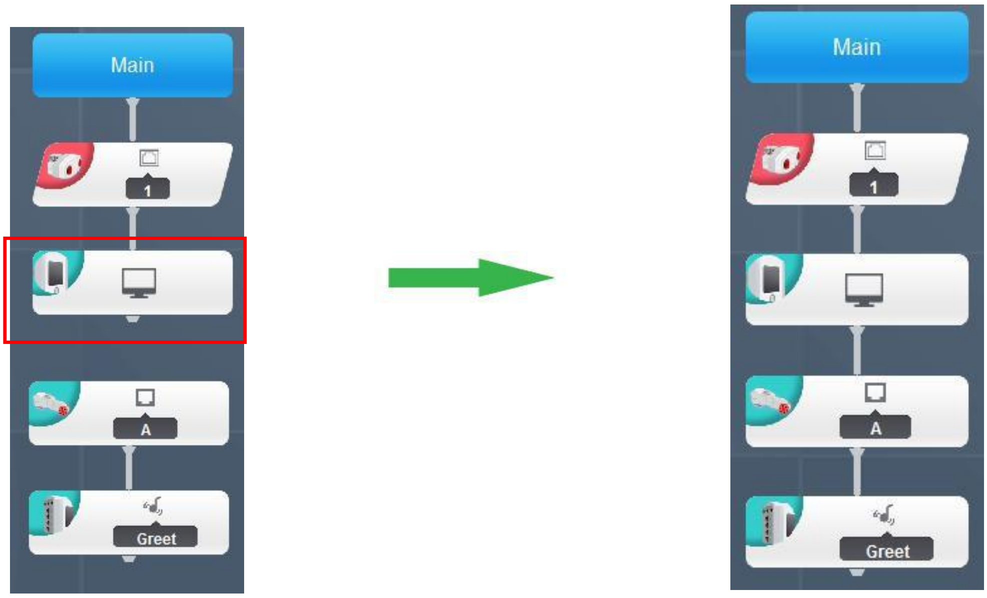
If there is a module in the position pending to insert a new module, drag it away and insert the new module, then, put the dragged away module back. For instance: Insert "Display" module between "Ultrasonic Sensor" and "Big Motor" as instructed below:

1. Drag "Big Motor" away;

2. Connect "Display" to "Ultrasonic Sensor";

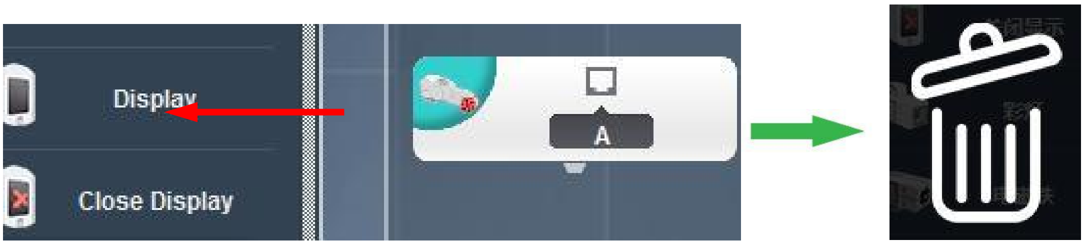
3. Connect "Big Motor" to "Display".





3.2 Delete module

Drag module to the left, when the trash bin appears, loose the left side of cursor and the module will be deleted. For instance:



3.3 Code program

The starting point is “Main”, therefore, user can connect different modules to “Main” based on his/her logic to form a program.

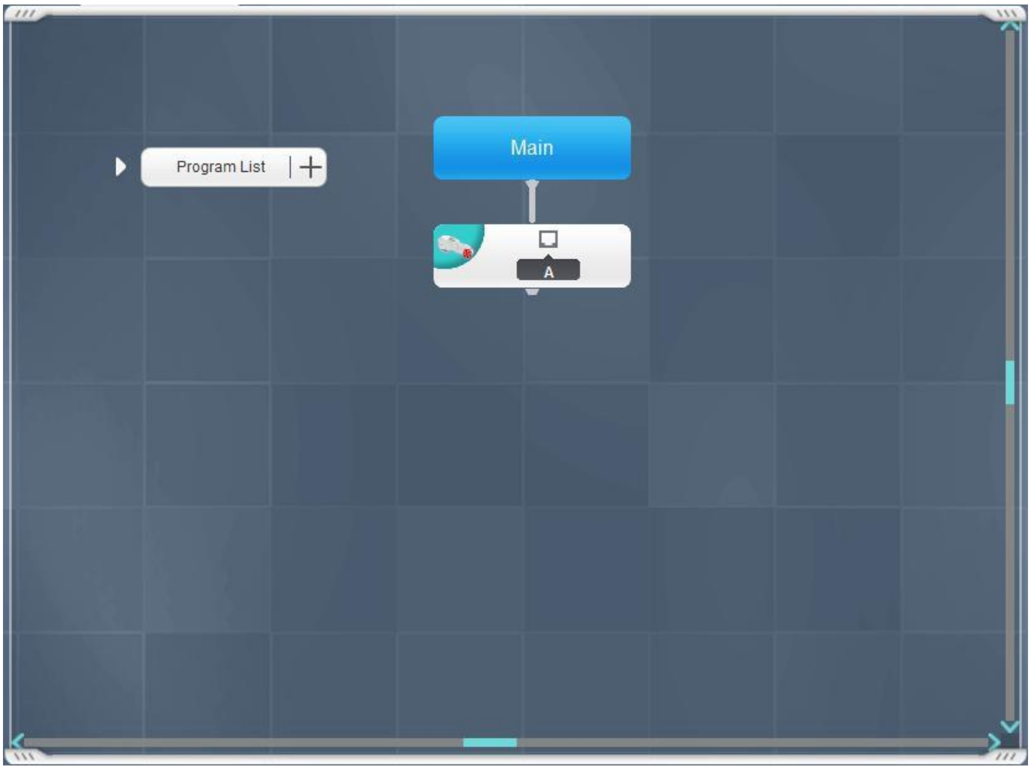
When dragging the module to coding area, it is dissociated and invalid. Only when the module joins the module chain under “Main” can it be valid.

Code program as follows:

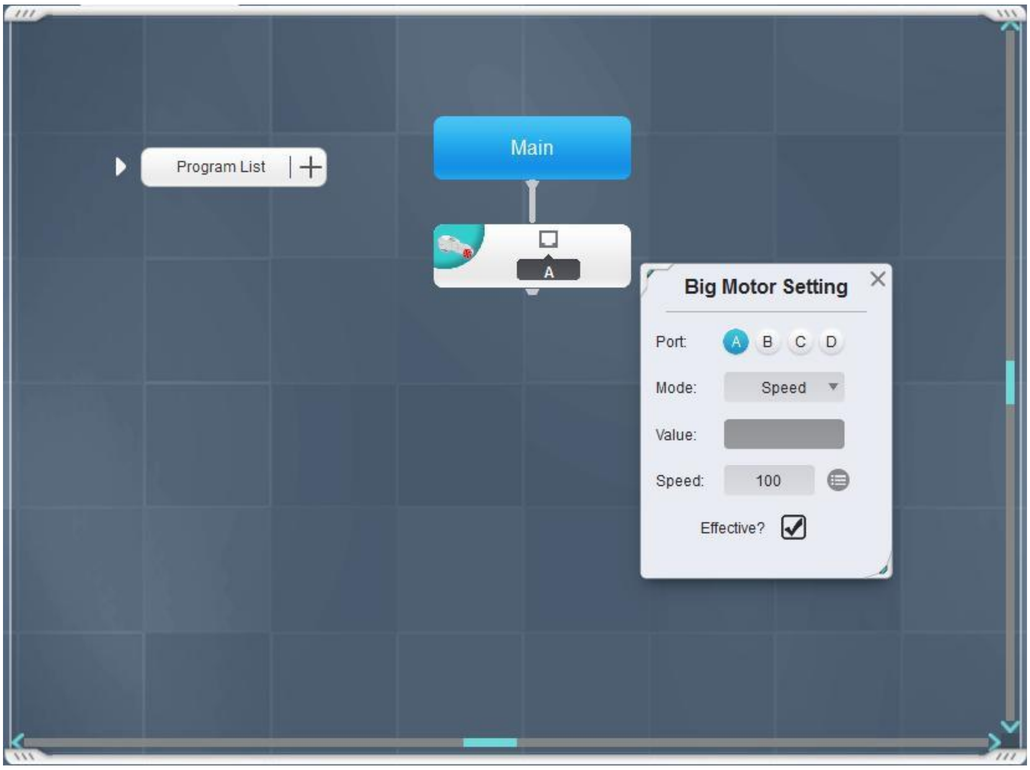
1. **Choose the needed module from icons on the left side and drag it to the coding area on the right side;**
2. **Move the module under "Main" to join module chain;**
3. **Press the module and alter parameter in setting frame;**

If user tends to code a program to let motor on Port A rotate clockwise at a speed 60 for 2 seconds and stop then, follow the instructions as below:

1. **Drag a "Big Motor" from the left side to connect to "Main";**



2. **Press ‘‘Big Motor’’ and parameter setting frame will pop up. Set motor port as ‘‘A’’, mode as ‘‘Speed’’ and speed as ‘‘100’’, tick ‘‘Effective’’;**



Big Motor Setting

Port: A B C D

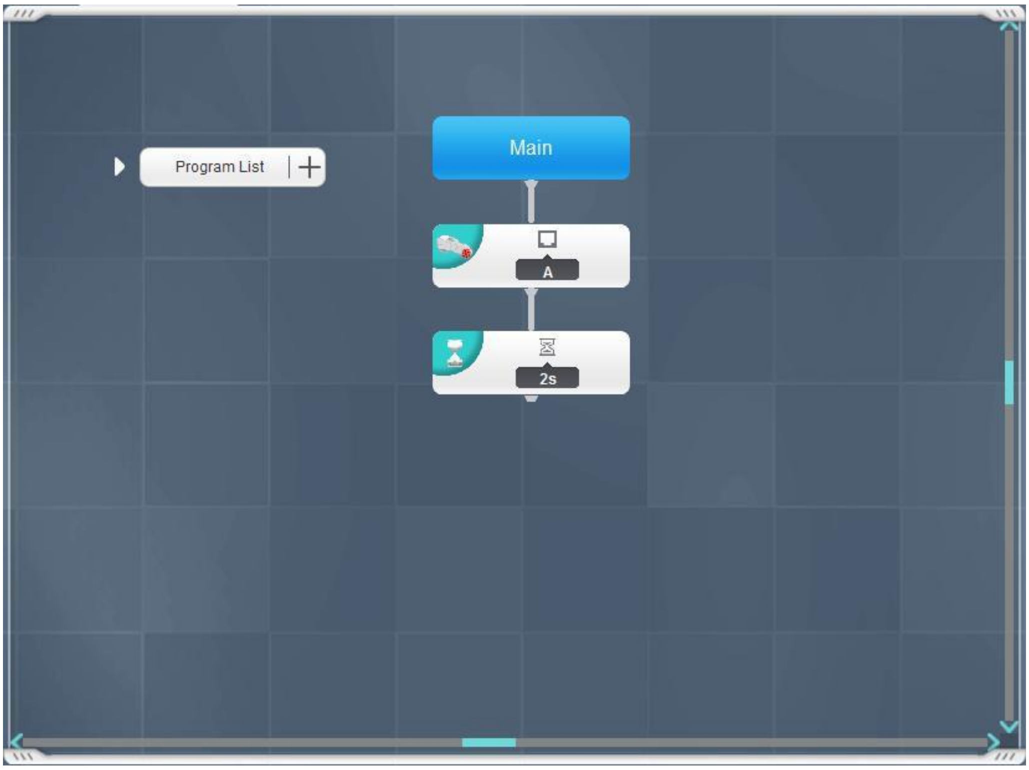
Mode: Speed

Value:

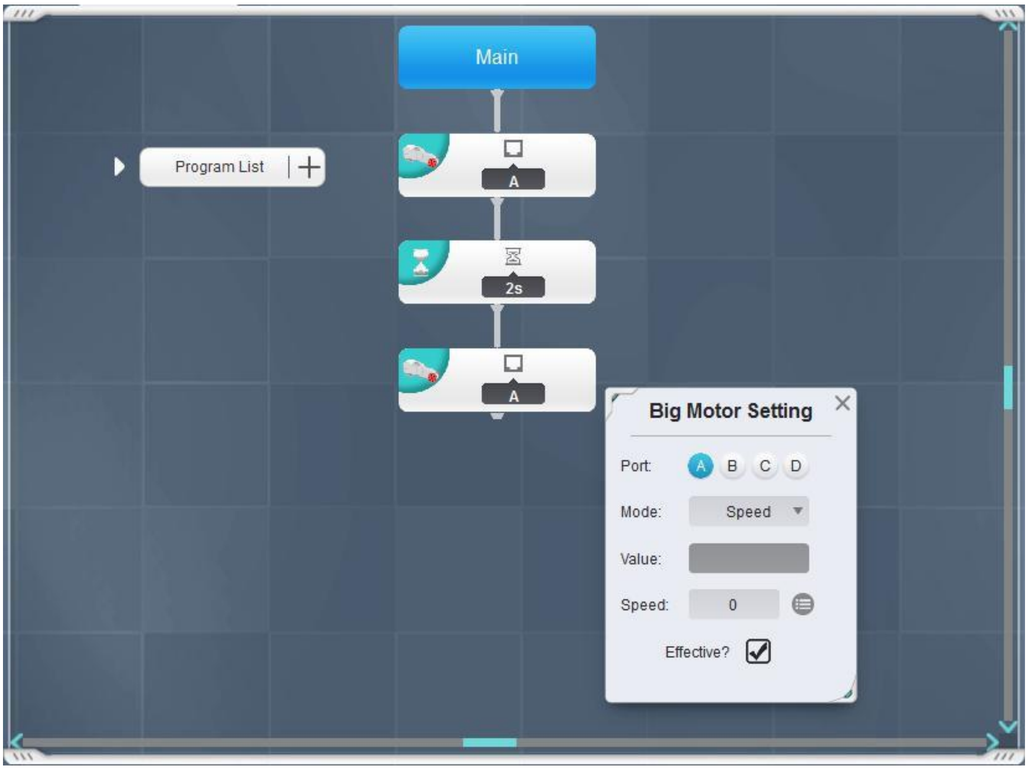
Speed: 100

Effective? ☑

3. Drag a “Delay” below “Big Motor” according to step 1 and step 2, and set the time as “2”;



4. Drag a “Big Motor” below “Delay” and set motor port as “A”, mode as “speed”, speed as “0”, tick “Effective” to complete programming.



Big Motor Setting

Port: [✓ A] B C D

Mode: Speed

Value:

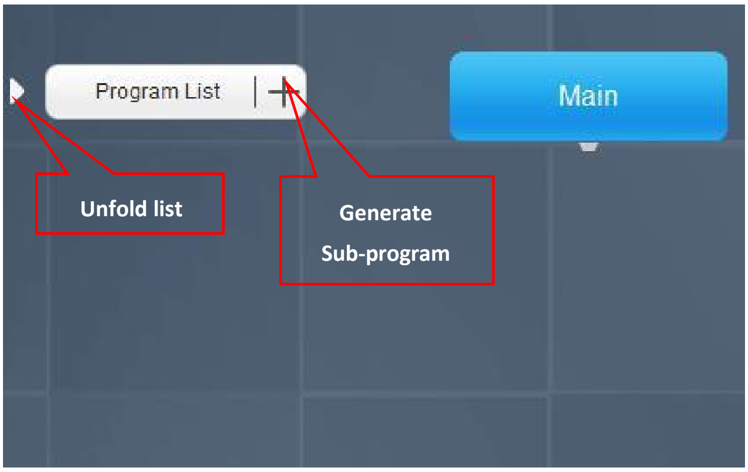
Speed: 0

Effective? ☑

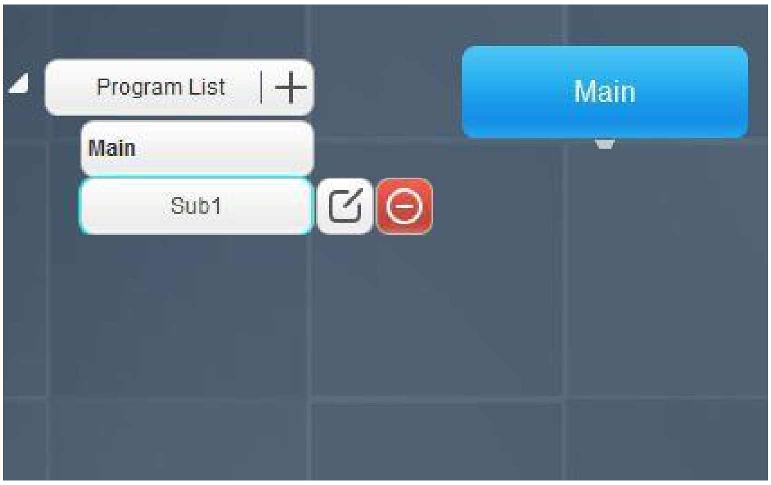
3.4 How to usesub-program

Use sub-program to cut down on code quantity and improve the readability of program. For instance: A complicated program can be divided into different function modules and each function module can be performed in sub-program. Follow the below instructions to set up a sub-program which makes controller light up for 5 seconds and then light off:

1. As instructed below, press + to generate sub-program and press ▶ to unfold program list, press sub-program to switch to corresponding programming interface.



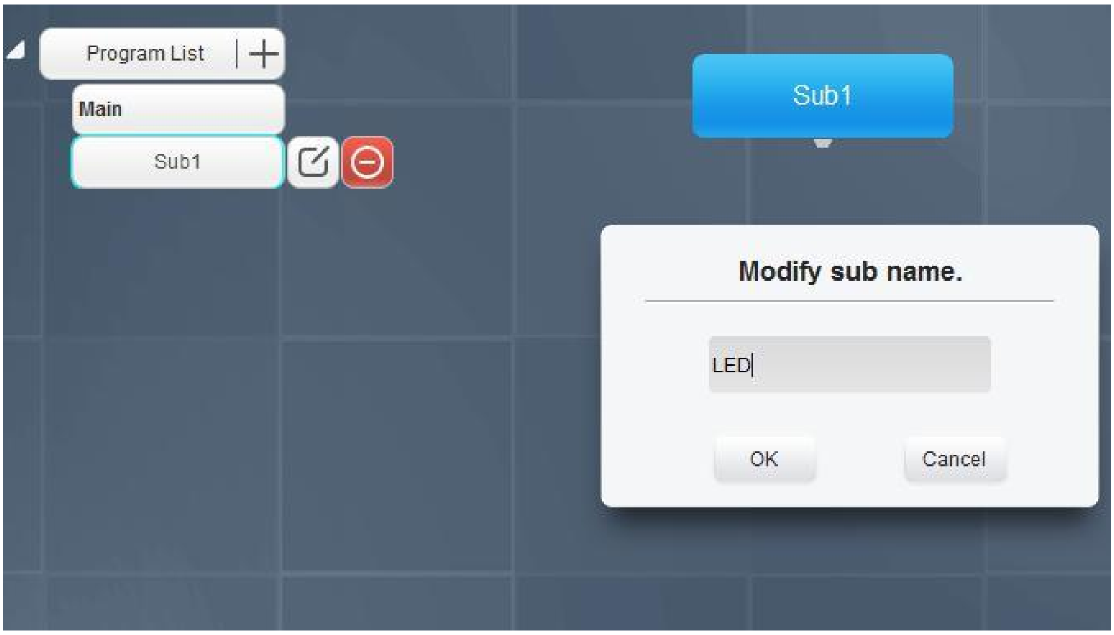
Pic 1



Pic 2

2. Place hand cursor on sub-program and ✎ ⊖ will appear. Press ✎ to rename sub-program, for instance: “LED”, as presented below:

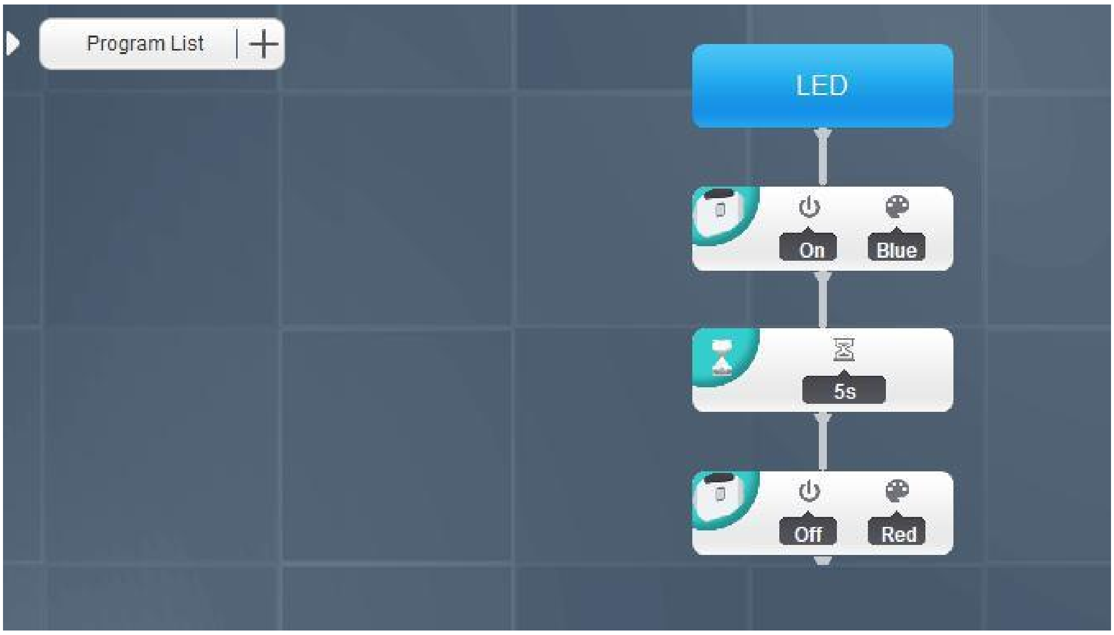
(Note: Press ⊖ to delete sub-program)



Modify sub name.

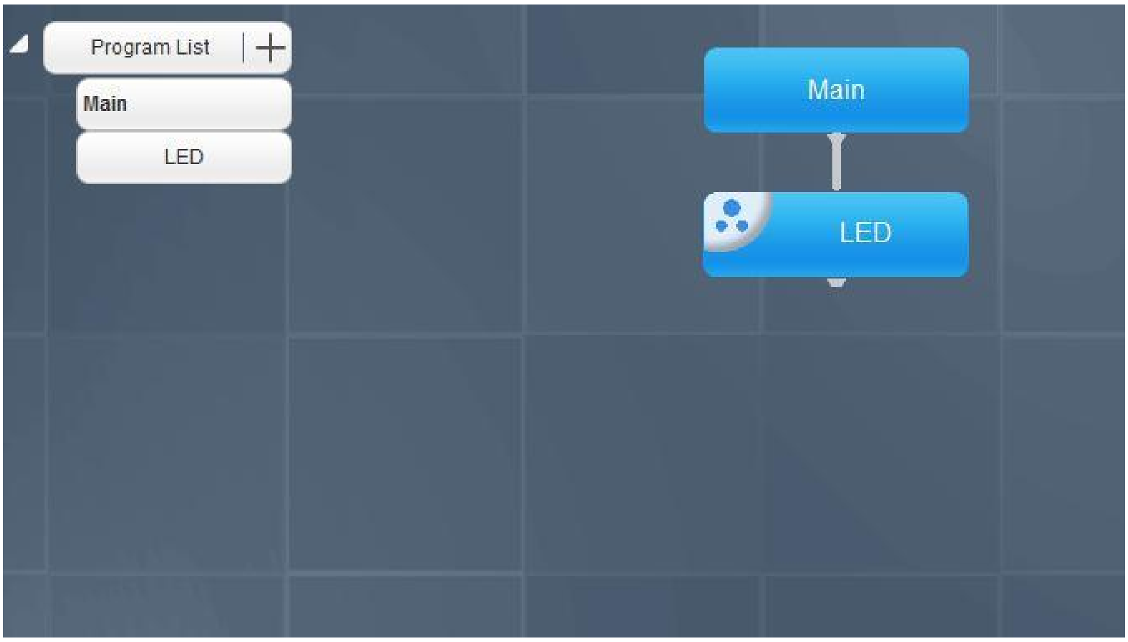
LED

3. Add a “LED” in sub-program and set the status as “On”, color as “blue”; add a “Delay” and set the time as 5 seconds; and then add another “LED” and set the status as “Off”, as instructed below:



4. Use the sub-program in “Main”

Press “Main” to enter main interface and drag sub-program “LED” to the right side to connect to “Main”, as presented below:



Download and execute the program, controller will flash blue for 5 seconds and then light off.

3.5 How to usevariables

Use variables to program can make the control flexible. After setting up a variable, it can be used after being assigned a value or operating.

New variable: In variable module lab, when pressing ＋, a customized variable will be generated with a default name “cus_1”; user also can press blue area to re-name as presented in pic 1.

Drag sensor to coding area and the variable name will appear on the interface. For example: drag grayscale sensor to coding area, the variable name “gra_1’’ will appear on the interface.

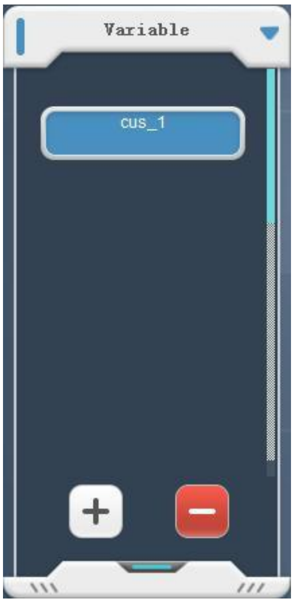


图1

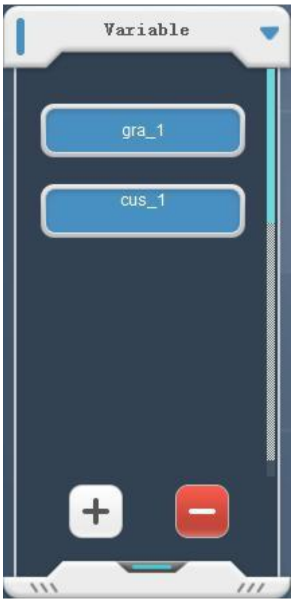


图2

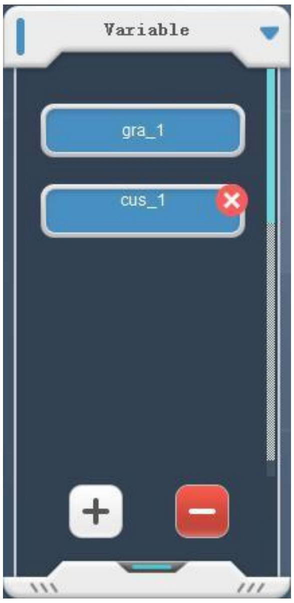


图3

Delete variable

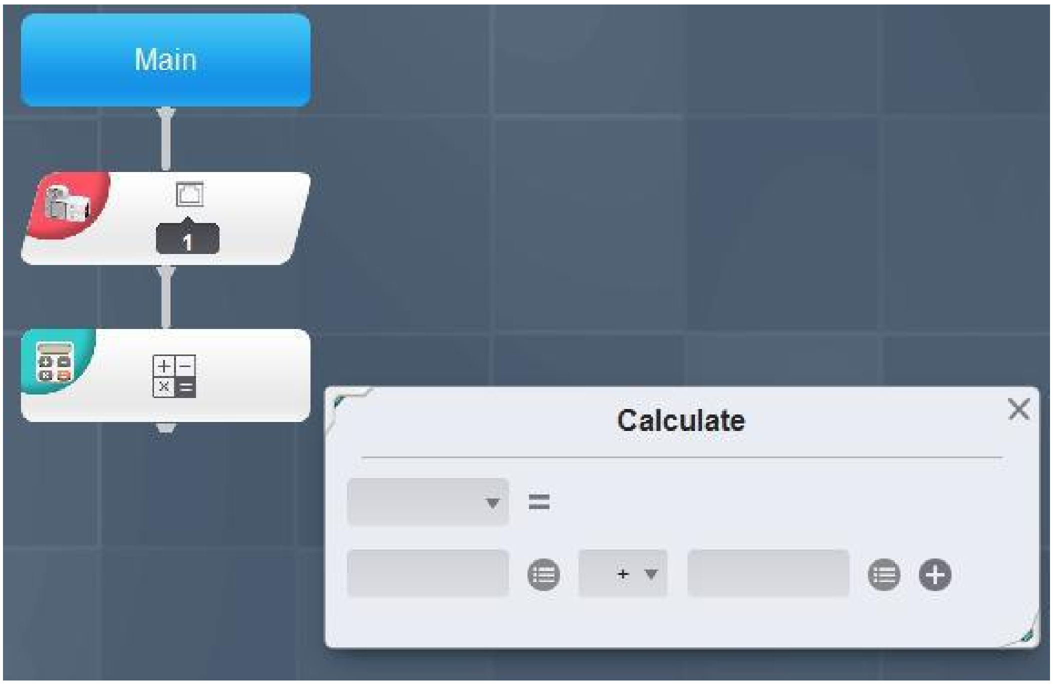
Press ⊖ as presented in pic 3:

Gra_1: generated by system (Grayscale sensor) and can not be deleted.

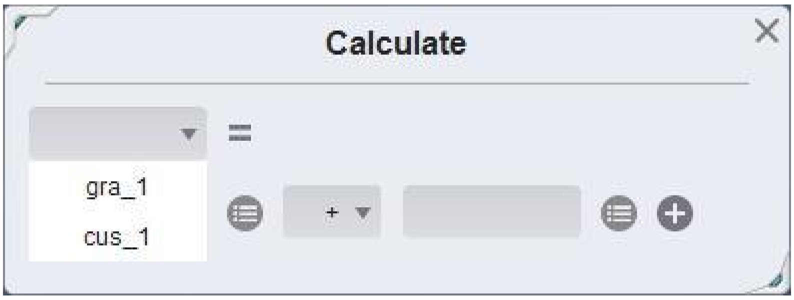
Cus_1: customized variable and can be deleted. Press ⊗ on the top right corner to delete.

Variable:

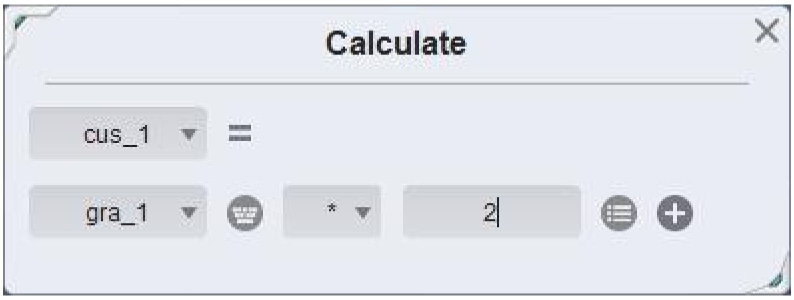
Assign a value to customized variable cus_1. Drag grayscale sensor and calculate module:



Follow the instruction as below:



Choose customized variable cus_1 in the first enter frame

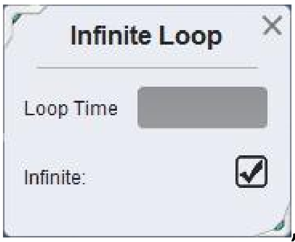


Choose variable, operating symbol, enter digit, the formula is: cus_1 = gra_1*2

3.6 Judgement module

There should be object and reference for judgement. Object is usually the returned value or updated variable from the sensor. Look at the “Speaker” program to see how to use judgement module.

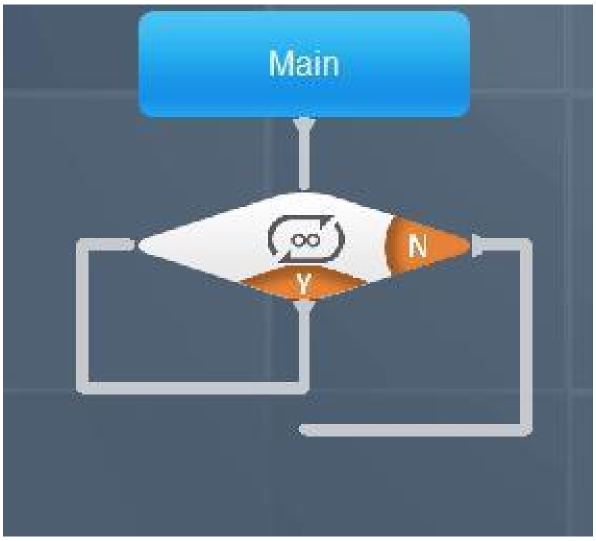
Drag a “Multi Loop” module , tick “Infinite Loop” in parameter setting frame, for example , and connect to “Main” as below:



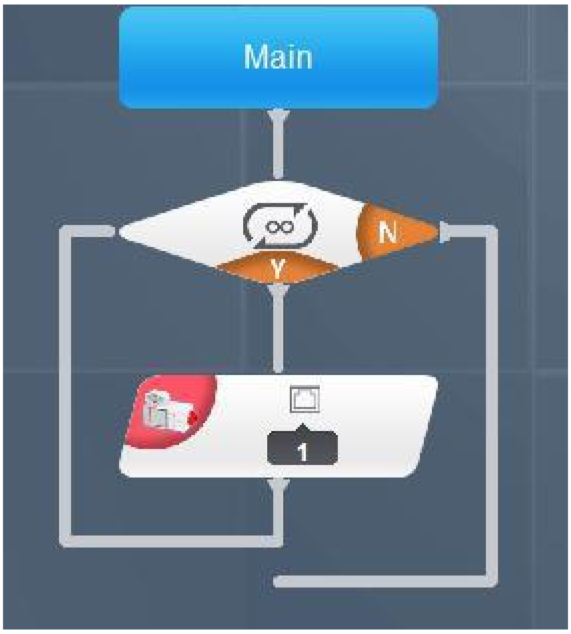
Infinite Loop

Loop Time

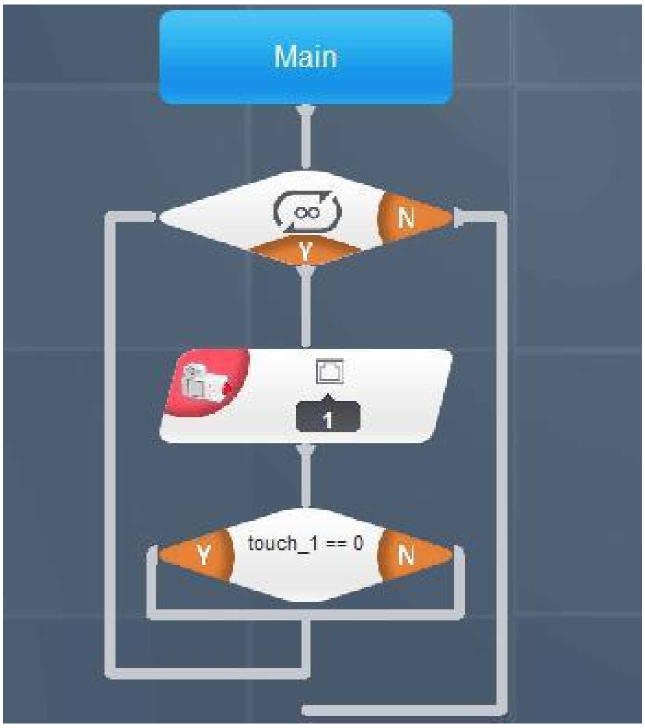
Infinite: ☑



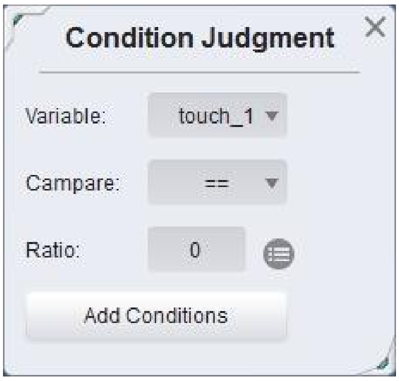
1. **Drag touch sensor , connect to “Infinite Loop” as below. Choose Port 1 and a variable air_1 will be generated automatically in variable module lab.**



2. Drag "Condition Judgement" to connect to "Touch Sensor" as below:



3. Set parameter of judgement module as below: If touch sensor is not held down, execute the program on the left side of the module, otherwise, execute the program on the right side.

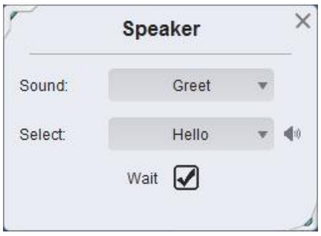


Condition Judgment

Variable: touch_1

Campare: ==

Ratio: 0



Speaker

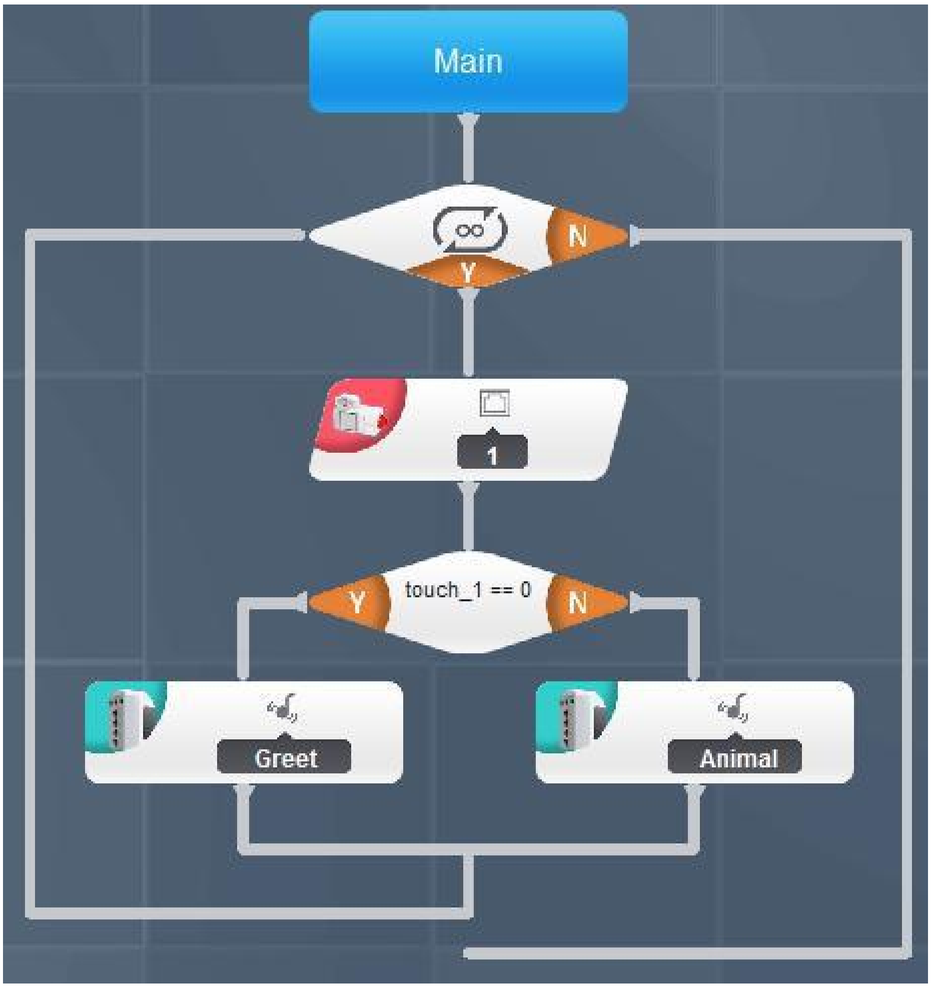
Sound: Greet

Select: Hello

Wait ☑

1. Drag “Speaker” module . In parameter setting frame , choose sound type as “Greet”, sound as “Hello”, tick “Waiting”; connecting the module to the left side of condition judgement.

2. Drag “Speaker” module, choose sound type as “Animal”, sound as “Seal”; connecting the module to the right side of condition judgement. Please see picture as below:



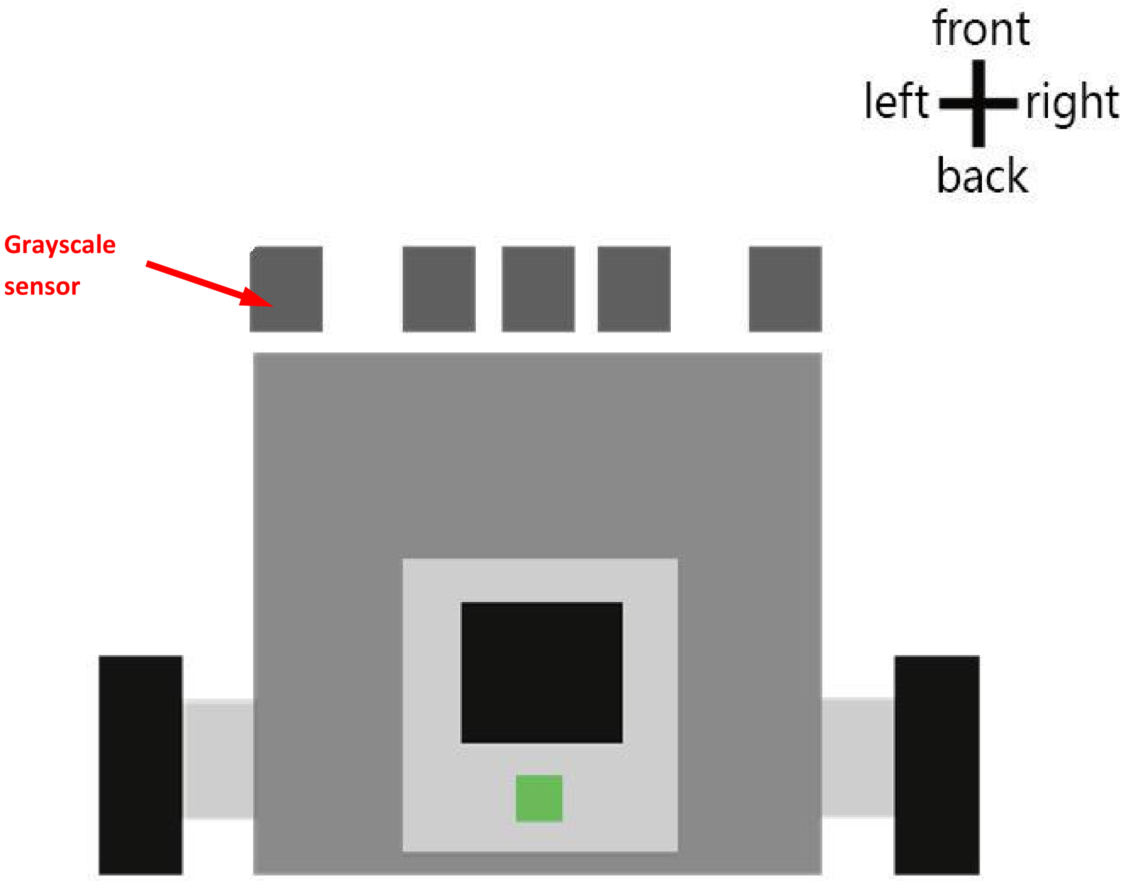
3. “Speaker’’ program is written. Connect touch sensor to the corresponding port of controller (Port 1), download the program to the controller and press the program to turn on “Speaker’’.

If touch sensor is not pressed, “Speaker’’ will play “Hello’’; if touch sensor is pressed, “Speaker’’ will play the sound of “Seal’’.

3.7 Tracing program

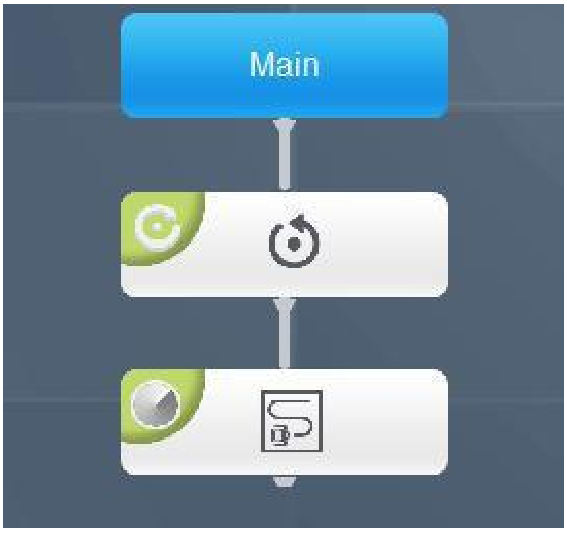
Prepare a tracing vehicle with 5 grayscale sensors, which are respectively inserted into port 1, 3,

4, 5, 7 of controller from left to right. Left and right motors are inserted into port A and D of controller as below:

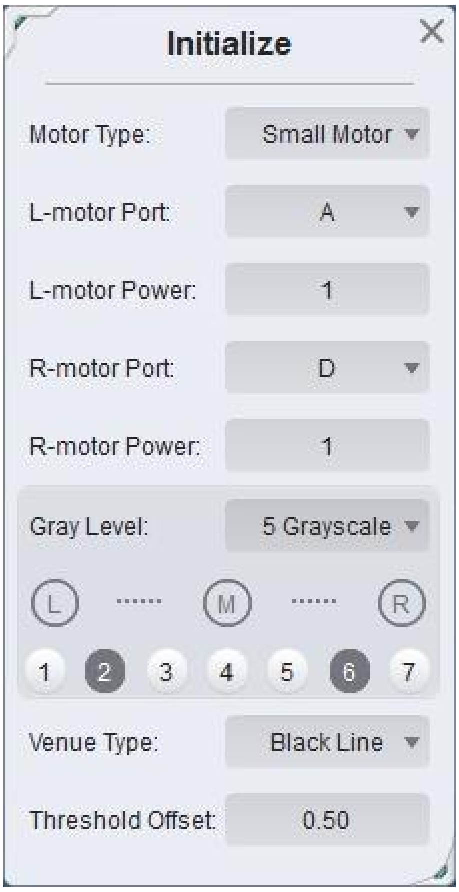


Collect environment data before programming, drag "Initialization"

and "Environment-collection" into coding area as below:



Parameter should be set as below:



Initialize

Motor Type: Small Motor

L-motor Port: A

L-motor Power: 1

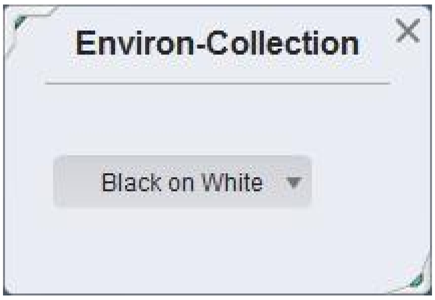
R-motor Port: D

R-motor Power: 1

Gray Level: 5 Grayscale

Venue Type: Black Line

Threshold Offset: 0.50



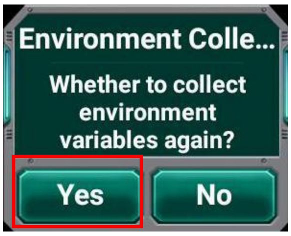
Environ-Collection

Black on White

Collect data:

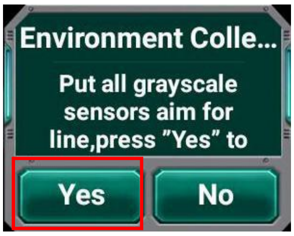
Download program and press “On’’, press “OK’’ according to prompt. Align grayscale sensors with lines and press “OK’’ to collect data; then, align grayscale sensors with background, press “OK’’ to collect data (The height is the distance from the grayscale sensor on the vehicle to the ground); there will be 2 rows of data on controller screen, black data and background data respectively; press blank area to finish collection.

Controller will save data automatically as below:


Environment Colle...

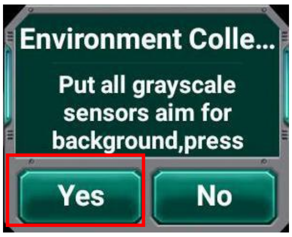
Whether to collect environment variables again?

（1）


Environment Colle...

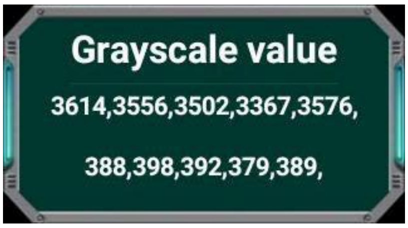
Put all grayscale sensors aim for line,press ”Yes” to

（2）


Environment Colle...

Put all grayscale sensors aim for background,press

（3）


Grayscale value

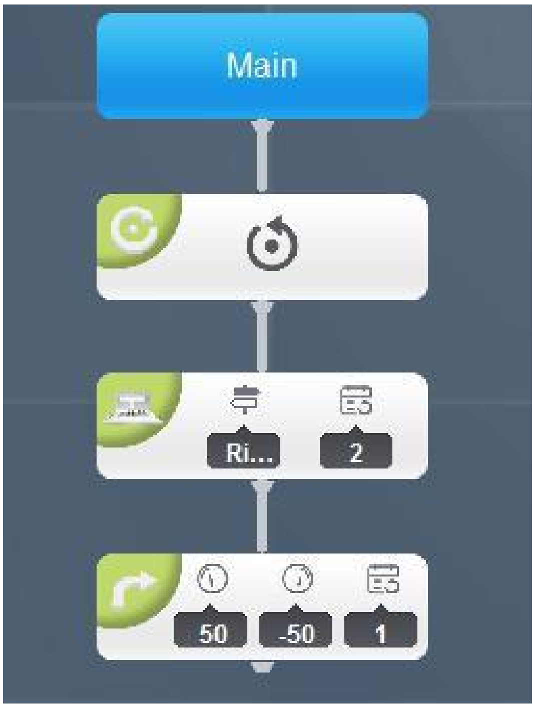
3614,3556,3502,3367,3576,

388,398,392,379,389,

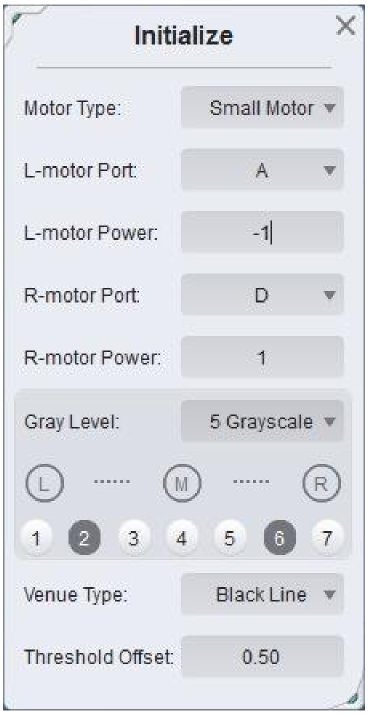
（4）

Vehicle tracing

After completing environment data collection, drag “Initialization’’, “Intersection tracing’’ and “Swerving’’ into coding area as below:

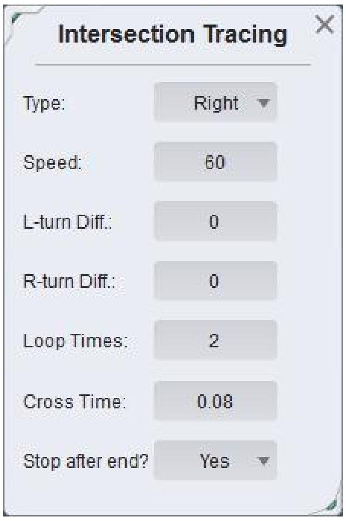


Set parameter as below (For reference only, data may differ based on types of vehicles):



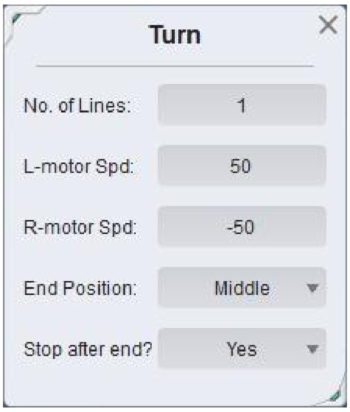
Initialize

Motor Type:	Small Motor
L-motor Port:	A
L-motor Power:	-1
R-motor Port:	D
R-motor Power:	1
Gray Level:	5 Grayscale
Venue Type:	Black Line
Threshold Offset:	0.50



Intersection Tracing

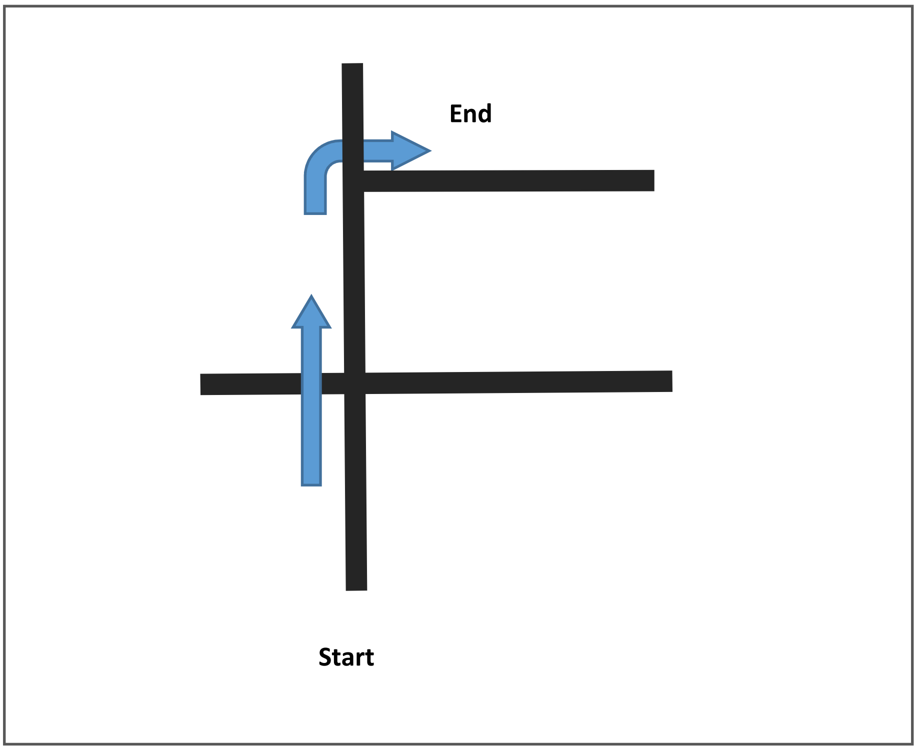
Type:	Right
Speed:	60
L-turn Diff.:	0
R-turn Diff.:	0
Loop Times:	2
Cross Time:	0.08
Stop after end?	Yes



Turn

No. of Lines:	1
L-motor Spd:	50
R-motor Spd:	-50
End Position:	Middle
Stop after end?	Yes

Press "On", as for a venue with white background and black lines, the vehicle will move along the curve and cross 2 intersections, then, stop after turning right at the 2nd intersection.


End
Start

4th Chapter C language programming

Click C in toolbar to enter C language programming. Controller of Krypton 9 supports standard C language format. Function accesses of modules such as: motor, LED, speaker and sensor are provided. User can code C language programs to download to controller.

Click Help to gain Abilix Chart Manual, user can look up detailed function accesses and how they are used. Drag different kinds of modules to coding area to generate corresponding functions in C editor.

Function access:

Editor module C function list:

Name	Function	Parameter
Initialization	void Abilix_C_Init(void)	Initialize C environment
Quit C environment	void Abilix_C_Exit(void)	Analysis is done, quit C environment
Turn on closed-loop motor	void SmallMotor (char port, int speed)	Port: Controller motor port, A-E Speed: 0-100
Turn on closed-loop motor (Loop control)	void SmallMotor_circle (char port, int speed, int circle)	Port: Controller motor port, A-E Speed: 0-100 Circle: Rotated circles
Turn on closed-loop motor (Angle control)	void SmallMotor_angle (char port, int speed, int angle)	Port: Controller motor port, A-E Speed: 0-100 Angle: Rotated degree
Turn on closed-loop motor	void SmallMotor_time (char port, int speed, int time)	Port: Controller motor port, A-E Speed: 0-100

(Time control)		Time: Rotated time
Turn on closed-loop motor	void BigMotor (char port, int speed)	Port: Controller motor port, A-E Speed: 0-100
Turn on closed-loop motor (Loop control)	void BigMotor_circle (char port, int speed, int circle)	Port: Controller motor port, A-E Speed: 0-100 Circle: Rotated circles
Turn on closed-loop motor (Angle control)	void BigMotor_angle (char port, int speed, int angle)	Port: Controller motor port, A-E Speed: 0-100 Angle: Rotated degree
Turn on closed-loop motor (Time control)	void BigMotor_time (char port, int speed, int time)	Port: Controller motor port, A-E Speed: 0-100 Time: Rotated time
Speaker	void playSound (char*fileName, int isWait)	File Name: Audio name, see attachment for reference. For instance: “Greeting” in Chinese: dazhaohu_c0 isWait： to wait for the speaker to play or not?
Stop “Play”	void stopPlay (void)	Stop playing sounds
Record	void record (int name, int time)	Use microphone to record. Name: Name of recording, 1-10; Time: Record for 0-60seconds
Button LED	void setLED (int switch, int color)	Switch: Turn on or off button LED, 0 off, 1 on. Color: 1: Red, 2; Green, 3: Blue.
Display	void setDisplay ()	setDisplay() Function is formatted output function. setDisplay(“<formatted character sting>”, <reference>). The use method is the same as that of C

		language printf().
Display Image	void setDisplayImage (int type, char* name)	Type: 0 displays taken photos, 1 displays customized photos; Name: Photo ID or name.
Close Display	void closeDisplay (void)	Close display on robot end.
Animation	void playAnimation (int ID)	Play inbuilt animation. ID: 1. coward, 2. happy, 3. cheer up, 4. lovely, 5 cry, 6. grieved, 7. blink

Smart LED	void smartLED (int mode, int R, int G, int B, float rate)	Mode: LED mode, 0-on, 1-off, 2-breathe;
		R/G/B: Set RGB value, 0-225;
		Rate: Breath rate: 0.1-60.

Smart Motor	void smartMotor (int ID, int angle, int speed)	ID: Servo ID, 1-3;
		Angle: Servo rotated degree, -90-90;
		Speed: Servo rotated speed, 200-1023.

Color LED	void DO (int port, int switch)	Port: Controller Port. No.; Mode: 0-off, 1-on.
Electromagnet	void DO (int port, int switch)	Port: Controller Port. No.; Mode: 0-off, 1-on.
Digit Output	void digitalOutput (int isDo, int status)	isDo: Choose port, Port7~Port 1 sequenced from high to low and controlled by binary form, 1-choose, 0-not choose. Parameters need to be converted into decimal. Status: whether to open the port, port 7~ port 1 sequenced from high to low and controlled by binary form, 1- on, 0- off. Parameters need to be converted into decimal.

Delay	void delay (float time)	Time: Delayed time, unit: second.
Compass Calibrate	void initCompass (void)	No returned value.

Sensor module C function list:

Analog output	int AI (int port)	Read returned value (grayscale, ultrasonic, touch, photosensitive, magnetic, flame and temperature sensor). Port: controller port No. 1-7; Returned value range: 0-4095.
Color sensor	int getColor (void)	Read returned value. Returned value: 0: Red, 1: Yellow, 2. Green, 3. Blue, 4. White, 5. Black, 7: Abnormal;
Sound test	float soundDetection (void)	Test the sound volume of surroundings.
Photosensitive detection	int lightDetection (void)	Use color sensor to detect the strength of rays; returned value ranges from 0 to 255.
Gyroscope	float getGyro (int axial, int angle)	Collect corresponding value of gyroscope. Axial: 0: X, 1: Y, 2: Z; Angle: Degree, 0
Read Compass	float getCompass (void)	Detect to which orientation the head of controller points. Returned value ranges from 0 to 359, 0 - North, 90 - East, 180 - South, 270 - West.
Camera	void camera(int pic_id)	Use camera to take photos Pic_id: ID of saved photos, ID 1-10
Time Reset	void resetTime (void)	No returned value.
System time	float getTime (void)	Obtain the time from program starting till the present.
Random digit	int getRand (int min,int max)	A random digit will be generated between min and max (Range: 0-9999)

In a situation that a function facing a motor with various ports, set as below:

void bigMotor(MotorInfo* p_motorRunInfo);

void smallMotor(MotorInfo* p_motorRunInfo);

The function allows a setting of motor with various ports to achieve a random combination of ports turned on simultaneously. (Random combination: A+B, A+C, A+B+C, ...)

The parameter of function is a MotorInfo, including a structure with ABCD ports. The structure also includes attributes of a certain port, for example: port is set opened: 0-not now, 1-OK; motor mode: 0-closed loop, 1-open loop; value type: 0-speed, 1-degree, 2-loop, 3-time; value: value type: degree/loop/time, effective; speed: -100-100.

4.1 附录

音频文件名对照表（中文）：

音频类型	音频名称	音频调用名称
打招呼	你好	dazhaohu_c0
	再见	dazhaohu_c1
	反对	dazhaohu_c2
	欢迎	dazhaohu_c3
	请多关照	dazhaohu_c4
表情	生气	biaoqing_c0
	傲慢	biaoqing_c1
	哭泣	biaoqing_c2
	激动	biaoqing_c3
	惊吓	biaoqing_c4
	委屈	biaoqing_c5
	高兴	biaoqing_c6
	可爱	biaoqing_c7
	大笑	biaoqing_c8
	悲伤	biaoqing_c9
	愤怒	biaoqing_c10
	调皮	biaoqing_c11
动作	打寒颤	dongzuo_c0
	卖萌	dongzuo_c1
	赞成	dongzuo_c2
	求抱抱	dongzuo_c3
	打哈欠	dongzuo_c4
	加油	dongzuo_c5
	睡觉	dongzuo_c6
	休闲	dongzuo_c7

	鬼鬼祟祟	dongzuo_c8

音频文件名对照表（中英文共用）：

音频类型	音频名称	音频调用名称
动物	海豹	dongwu_p0
	鹦鹉	dongwu_p1
	青蛙	dongwu_p2
	羊	dongwu_p3
	鸡	dongwu_p4
	鸭子	dongwu_p5
	狗	dongwu_p6
	天鹅	dongwu_p7
	蜜蜂	dongwu_p8
交通	摩托车	jiaotong_p0
	赛车	jiaotong_p1
	火车	jiaotong_p2
	汽车	jiaotong_p3
	飞机	jiaotong_p4
	轮船	jiaotong_p5
	自行车	jiaotong_p6
	直升机	jiaotong_p7
钢琴	1	gangqin_p0
	2	gangqin_p1
	3	gangqin_p2
	4	gangqin_p3
	5	gangqin_p4
	6	gangqin_p5
	7	gangqin_p6
	8	gangqin_p7

小提琴	1	xiaotiqin_p0
	2	xiaotiqin_p1
	3	xiaotiqin_p2
	4	xiaotiqin_p3
	5	xiaotiqin_p4
	6	xiaotiqin_p5
	7	xiaotiqin_p6
	8	xiaotiqin_p7
架子鼓	低音大鼓	jiazigu_p0
	军鼓	jiazigu_p1
	落地鼓	jiazigu_p2
	嗵嗵鼓 1	jiazigu_p3
	嗵嗵鼓 2	jiazigu_p4
	踩镲	jiazigu_p5
	吊镲	jiazigu_p6
	节奏镲	jiazigu_p7
	镲片	jiazigu_p8

音频文件名对照表（英文）：

音频类型	音频名称	音频调用名称
Greet	Hello	dazhaohu_e0
	Goodbye	dazhaohu_e1
	Reject	dazhaohu_e2
	Welcome	dazhaohu_e3
	Nice to meet you	dazhaohu_e4
Expression	Angry	biaoqing_e0
	Arrogant	biaoqing_e1
	Wheep	biaoqing_e2
	Excited	biaoqing_e3
	Frightened	biaoqing_e4
	Wronged	biaoqing_e5
	Happy	biaoqing_e6
	Lovely	biaoqing_e7
	Laugh	biaoqing_e8
	Sad	biaoqing_e9
	Wrathful	biaoqing_e10
	Tricky	biaoqing_e11
Action	Tremble	dongzuo_e0
	Act cute	dongzuo_e1
	Agree	dongzuo_e2
	Hug	dongzuo_e3
	Yawn	dongzuo_e4
	Cheer up	dongzuo_e5
	Sleep	dongzuo_e6
	Rest	dongzuo_e7
	Sneak into	dongzuo_e8