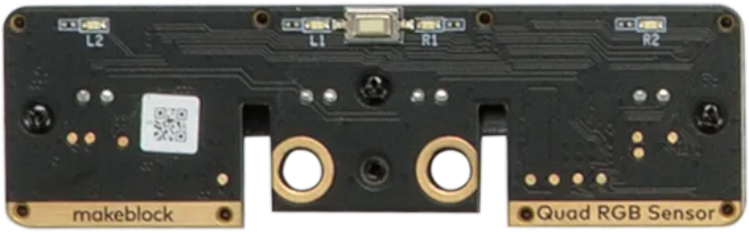
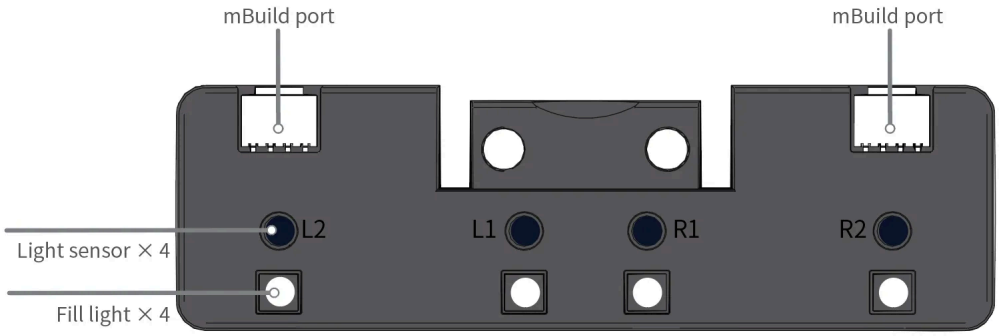
Quad RGB Sensor



The quad RGB sensor uses visible light as fill lights, which significantly reduce the interference of ambient light. In addition, it provides the function for recognizing colors. The new ambient light calibration function also reduces the interference of the ambient light in line following. With four light sensors, it can support more programming scenarios.



	Quad RGB sensor	Dual RGB color sensor
Plastic casing to improve the durability and quality	Yes	No
Line-following sensor	4	2
Color sensor	4 (also serve as line-following sensors)	No
Light sensor	4 (also serve as line-following sensors)	No
Fill light	Visible light	IR light
Ambient light calibration to significantly reduce the interference of ambient light	Yes	No

Specification

Detection range: 5–15 mm from the object to be detected

Button description

- Double-press: When the button is double-pressed, the quad RGB sensor starts to learn the background and line for line following.

  Place the light sensors on the background of the line-following track map and double-press the button. When you see the LEDs indicating the line-following state blink quickly, sway the sensors from side to side above the background and line until the LEDs stop blinking. It takes about 2.5 seconds. The parameter values obtained are automatically stored. If the learning fails, the LEDs blink slowly, and you need to start the learning again.

- Long-press: When the button is long-pressed, the quad RGB sensor switches the color of the fill lights. Generally, you don't need to change the color. The color is set automatically after the learning is complete.

Programming guide

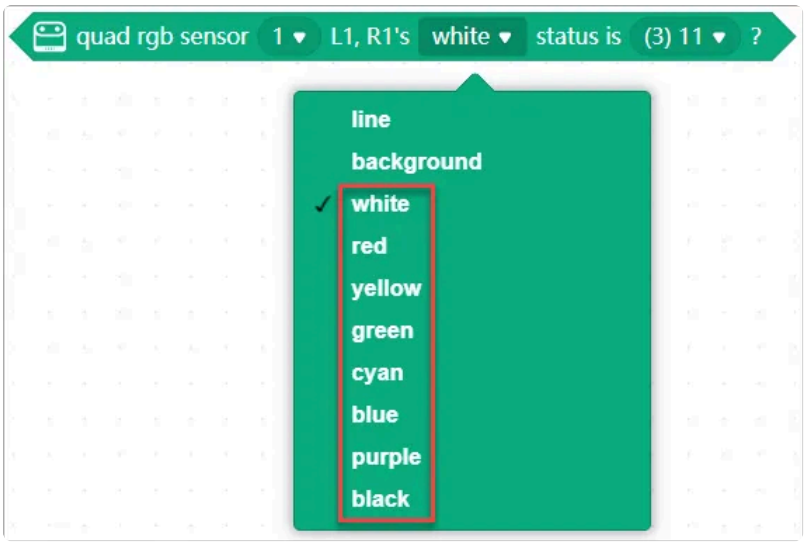
You can use mBlock 5 to program the quad RGB sensor.
Connect the quad RGB sensor to CyberPi, add CyberPi and connect it to mBlock 5, and then add the **Quad RGB Sensor** extension. For details about how to add CyberPi and connect it to mBlock 5 and how to add the extension, see "Add and connect mBot2 <https://www.yuque.com/makeblock-help-center-en/cyberpi/mbot2-start#GdG8U> " and "Add extensions <https://www.yuque.com/makeblock-help-center-en/cyberpi/mbot2-start#ZPpLb> ."

Set the detection mode

The quad RGB sensor can be programmed to detect either of the following:

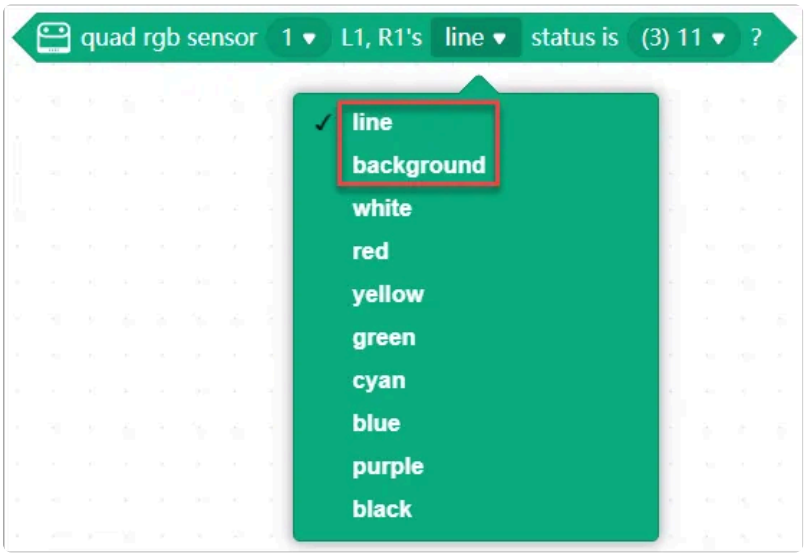
- color
- grayscale

For color detection, the quad RGB sensor measures the light reflected by an object in terms of its Red, Green, and Blue (RGB) values. Eight colors are available.
To detect a color, you need to select a color in the block to be used, for example:



For grayscale detection, the quad RGB sensor measures the intensity of the light reflected by an object regardless of its color.

To detect grayscale, you need to select **line** or **background** in the block to be used, for example:

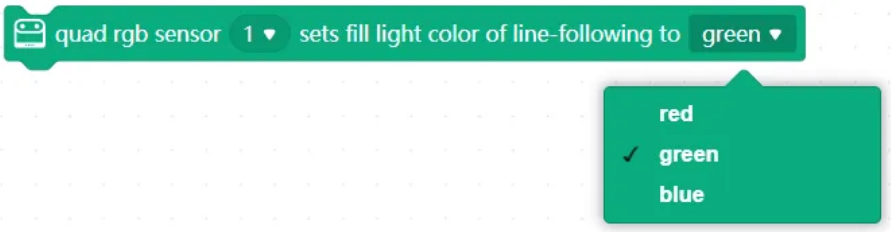


Obtain the output values

The quad RGB sensor can output values based on the color or grayscale it detects, and you can obtain these values by using the reporter blocks, for example:

Change the color of the fill lights

You can change the color of the fill lights between green, red, and blue by long-pressing the button or by using the following block:



Note: The quad RGB sensor can detect colors regardless of what colors the fill lights are.

You need to change the color of the fill lights in some application scenarios. For details, see "Line detection scenarios."

Line detection scenarios

Before you use the quad RGB sensor to detect lines, you need to perform basic calibration on it.

Basic calibration

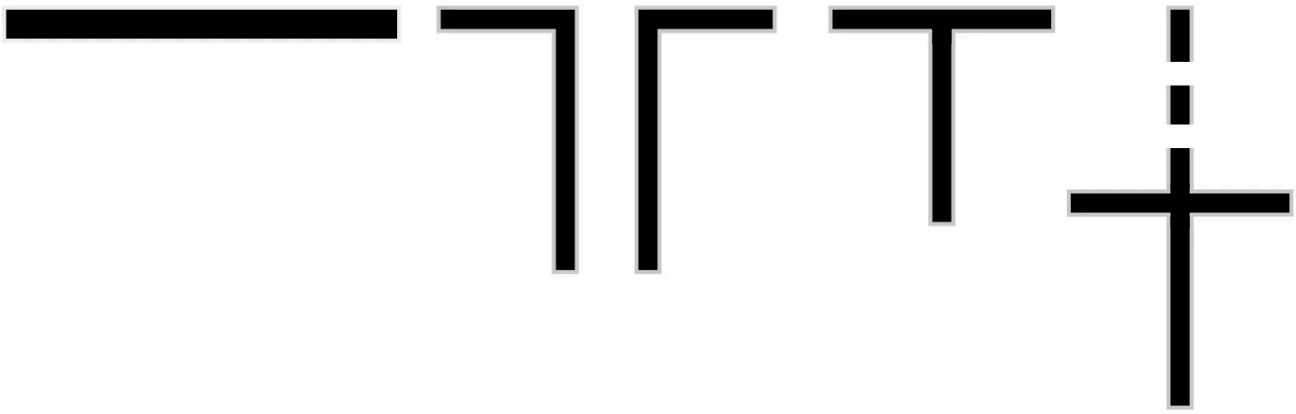
1. Place the four light sensors of the quad RGB sensor on the background and double-press the button on the sensor, and then sway the sensor horizontally over the line and background.
2. Check the two light sensors in the middle (L1 and R1) of the quad RGB sensor. If the robot is on the line, the blue LEDs of the two light sensors are supposed to be off. The two LEDs are turned on and off alternately when you sway the robot over the line.

Tips: The quad RGB sensor can detect dark lines on light backgrounds regardless of the color of the illumination.

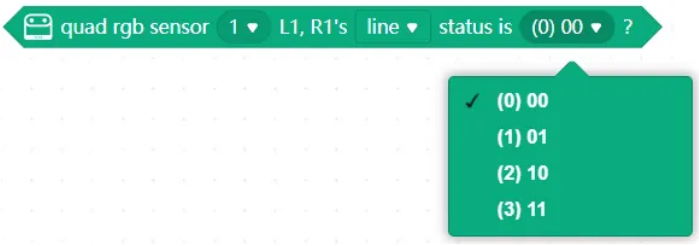
The following describes multiple line detection scenarios, and the purposes and blocks recommended for each scenario.

Scenario 1: Black line + white margin around line + white background

For example:



Perform basic calibration on the quad RGB sensor, so that it can recognize the line and background.

Level	Purpose(s) and block(s) recommended	Remarks
1a: junior	• To determine whether the quad RGB sensor detects the specific line status:  • To obtain the status of the line detected 	
1b: intermediate	 To detect crossings  	The quad RGB sens overlook the line, an it can't determine wh junction is actually a ("+" shape, the actu continues after the T only be checked by straight over the T a detecting if the line

1c: advanced	To implement smooth line following: 	Check the output va block. Switch the color of t to make sure that th value is **0** when the the middle of the lin The color of the fill l have a slight impact output value.

Scenario 2: Black line with colored sections + white margin around line + white background

For example:

The purposes and blocks recommended are the same as those in Scenario 1, but special calibration is required.

Either of the following ways can be used:

- Perform the basic calibration on the lightest color among the colored sections, for example, yellow, so that all darker colors are recognized as black.
- Perform basic calibration on the black line, and then long-press the button on the sensor to change the color of the fill light and see if the sensor recognizes the different colored sections as the line to be followed.
  - red section: use the green or blue/white fill light
  - yellow section: use the blue/white fill light
  - green & blue section: all fill lights work

Important to know:

- If you have set the color of the fill lights on mBlock 5, the calibration resets the fill lights (each fill light cycles through R/G/B illumination fast).
- Colored sections inside the line to be followed may affect the advanced, smooth line following (recognized as the deviation from the line). For example, when the basic calibration is done on the yellow section, the blue section may be reported as a deviation even if it is 100% on the line.

Scenario 3: Single-colored line + white margin around line + white background

For example:

Perform basic calibration on the quad RGB sensor, so that it can recognize the line and background.

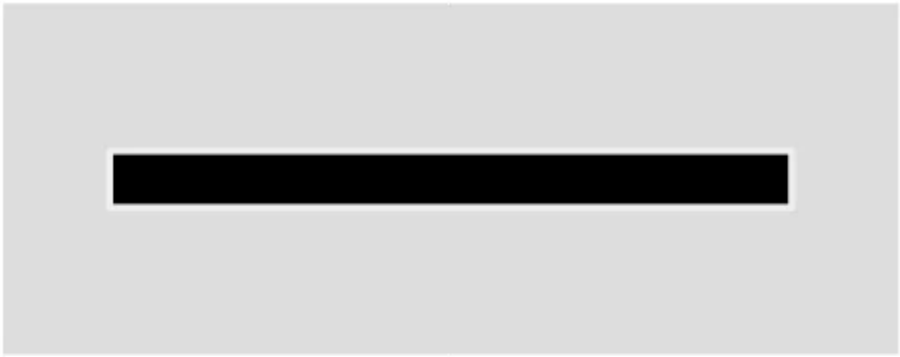
The purposes and blocks recommended are the same as those in Scenario 1.

Notes:

- If you use the deviation reporter block, long-press the button on the quad RGB sensor to switch the color of the fill lights to make sure that the output value is **0** when you put the robot in the middle of the line.
- It is not recommended to use the deviation reporter block when the contrast between the line and background is low.

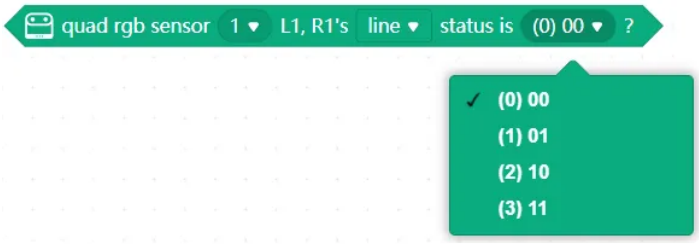
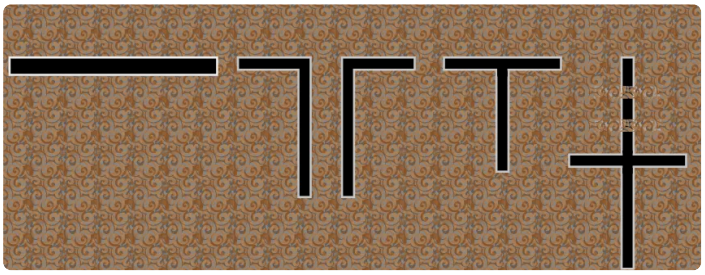
Scenario 4: Black line + white margin around line + colored background/illustrations

For example:



Perform basic calibration on the quad RGB sensor, so that it can recognize the line and background.

Level	Purpose(s) and block(s) recommended	Remarks

4a: junior	• To determine whether the quad RGB sensor detects the specific line status:  • To obtain the status of the line detected 	
4b: intermediate	 To detect crossings: 	The outer areas (bac may be recognized a to be followed, depe the light conditions. When this happens, use only the L1 and sensors to detect lin make sure that the w margin around line is enough to keep the s from detecting the ou background.

4c: advanced	To detect deviation from the line: 	If the colors of the ou are close enough to the line to be followe output value of the d reporter block may b and thus line followir In this case, long-pre button on the quad F to change the color o lights, making sure t difference between t background is recog blue LEDs are turnec alternately when you sensor over the line. If you use the deviat block, switch the col lights to make sure t output value is **0** whe robot is put in the mi line. The color of the fill li have a large impact output value.

Scenario 5: Black line + single-colored margin around line + white background

For example:

Perform basic calibration on the quad RGB sensor, so that it can recognize the line and background.

The purposes and blocks recommended are the same as those in 1a of Scenario 1. Crossing detection and deviation detection are not recommended because the margin around lines may be recognized by L2 or R2 as the line to be followed.

Tips: You can also program the quad RGB sensor to detect light-colored lines on dark or colored backgrounds.

Color detection programming examples

Example 1: Obtain the values of RGB detected by light sensors of the quad RGB sensor

Use the following program to display the values of RGB detected by the light sensors of the quad RGB sensor on the screen of CyberPi:
RGB output <https://drive.google.com/file/d/1AapU7EOUI9ujoXjO-PBbSdktwNIXm0Js/view?usp=sharing>

Example 2: Define a color and its tolerance

- You can define a color with the following RGB ranges:
  - R: 0–255
  - G: 0–255
  - B: 0–255
- Tolerance

  For example, set the block as follows.

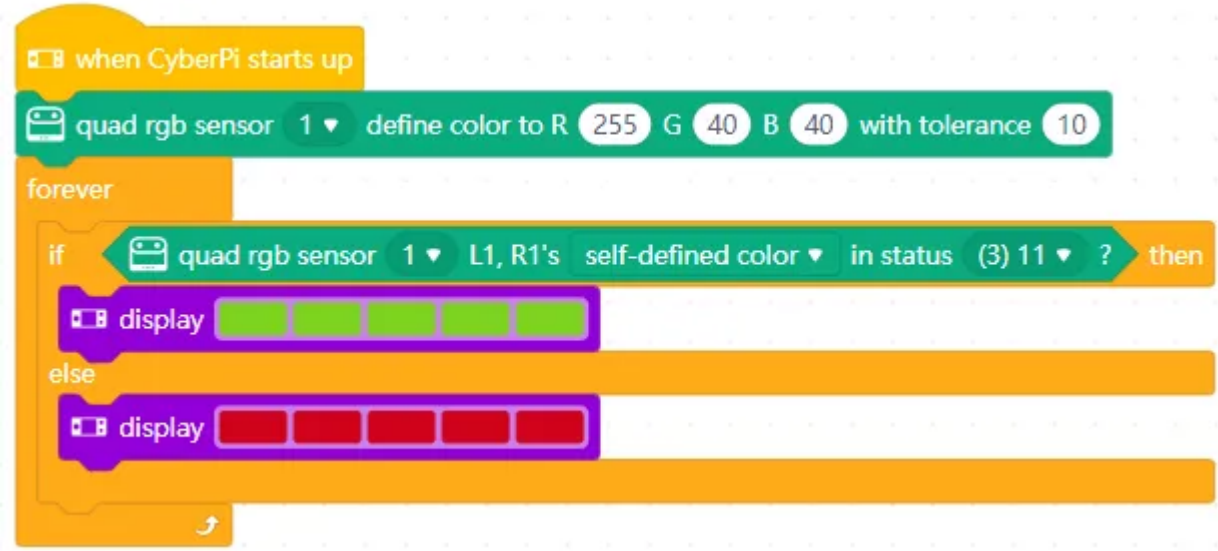
	R	G	B

Initial RGB values	255	40	40
If **tolerance** is set to 10, then:			
Upper limit of RGB values	255	50	50
Lower limit of RGB values	245	30	30

You can set the tolerance based on the ambient light and object color differences.

Example program:

Define a color <https://drive.google.com/file/d/170h5wmayZGL5GwUFiMJqynFnw0NzjRe4/view?usp=sharing>

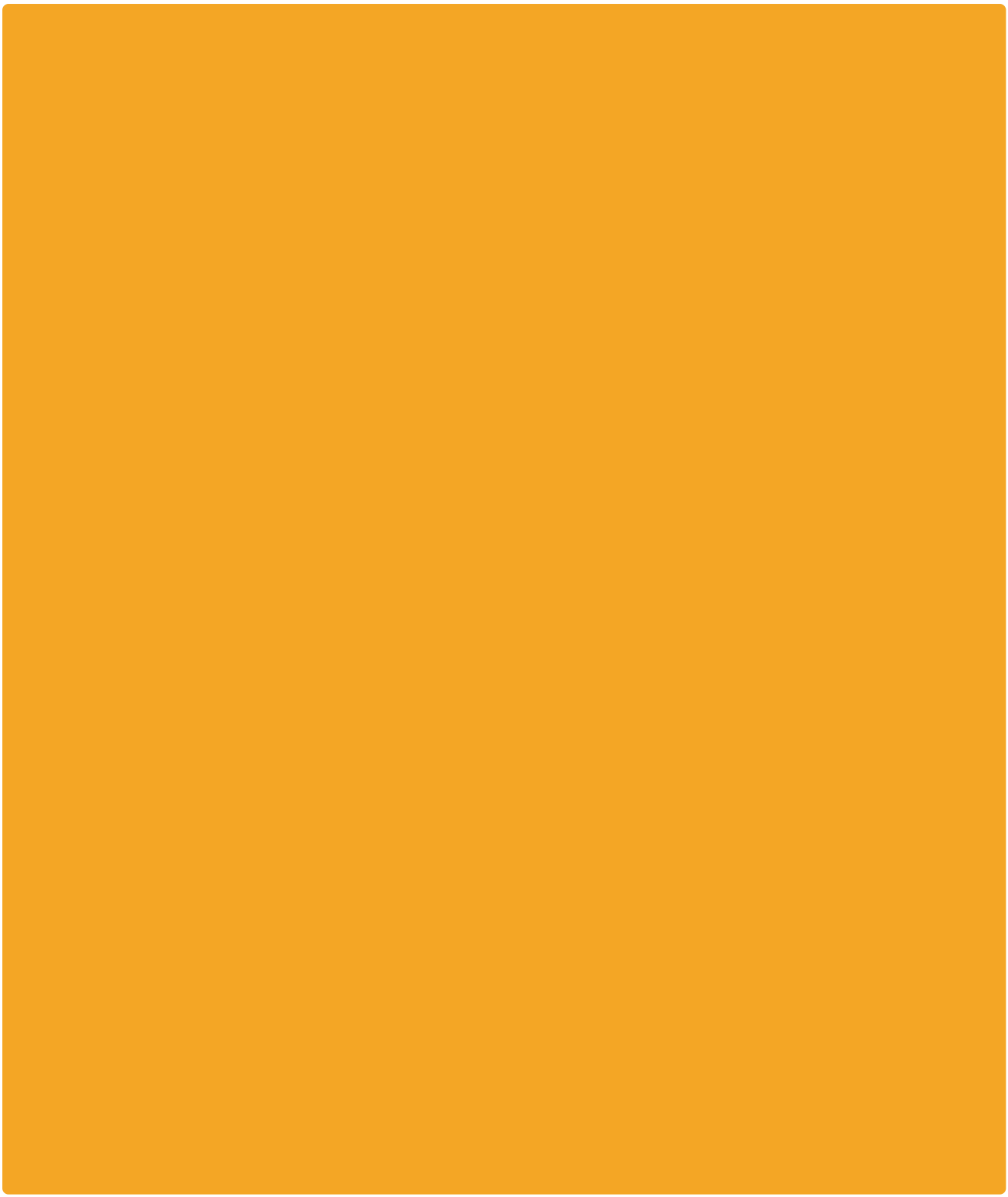


After you start CyberPi, the LEDs on CyberPi light up in green when the color defined by you, red in this example, is detected by both sensors R1 and L1. Otherwise, the LEDs on CyberPi are on in red.

Example 3: Tell the color of an object

Example program:

Tell the color of an object <https://drive.google.com/file/d/1o5K_C3UIF-kpj1ZI9Fl5-llCpFoBcJmI/view?usp=sharing>



(1) Place the quad RGB sensor over the object of which the color is to be recognized.
(2) Press button A on CyberPi. The color of the object is detected and the RGB values are displayed on the screen of CyberPi.
(3) Press button B on CyberPi. The RGB values are cleared from the screen on CyberPi.

Block added for quad RGB sensor calibration

Why do I need to perform calibration on the quad RGB sensor?

The performance of the quad RGB sensor may be significantly affected by the ambient light, map, color card, or the distance between the light sensors and the object to be detected. Therefore, performing calibration on the quad RGB sensor in the environment and conditions where it is to be used can improve the detection accuracy.

Calibration preparation

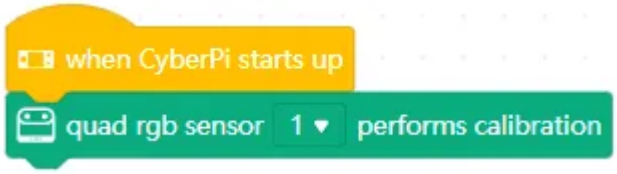
- Update the firmware to version 011, and update the **Quad RGB Sensor** extension to the latest version.
- Assembled mBot2, or CyberPi + mBot2 Shield + quad RGB sensor
- White paper of A6 or larger, or the white area on the mBot2 line-following map
- Find a place with common room light instead of intense light

Calibration steps

(1) Connect the quad RGB sensor to the mBuild port on mBot2 Shield or CyberPi by using an mBuild cable.
(2) Connect CyberPi to your computer by using a USB cable.
(3) Open mBlock 5 and connect CyberPi to mBlock 5.

(4) Add the **Quad RBG Sensor** extension.

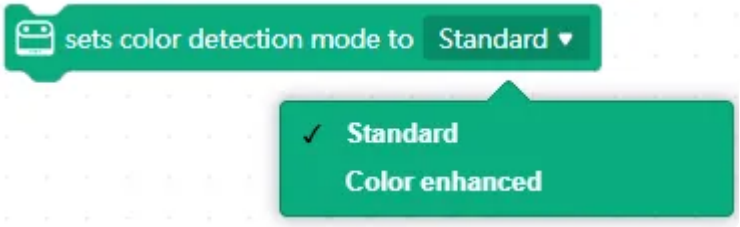
(5) Write the following program and upload it to CyberPi. Alternatively, you can write a program in **Live** mode.



The calibration process is short. Place the quad RGB sensor (**12 mm to 13 mm recommended**) over the white paper before you start the calibration program.

During the calibration, the fill lights are lit up in other colors and then reset to the pre-defined color, which looks like blinking. When the calibration is complete, it says "**Calibration completed**" on the screen of CyberPi. You can execute the program to perform calibration on the quad RGB sensor again.

Block added for switching color detection mode



What does the color detection mode mean?

The algorithm of the quad RGB sensor is performed based on the hue, saturation, value (HSV) color model. The standard mode refers to the threshold used in earlier firmware versions, and the color enhanced mode is added to the latest firmware, in which the threshold is adapted to application scenarios based on the HSV model. You can select a color detection mode as required.

Notes

By default, the standard color detection mode is used when this block is not executed.

To switch back to the standard mode after you have used the color enhanced mode, you can use either of the following two ways:

- Delete the **sets color detection mode to ()** block from your program. But note that you need to power off CyberPi and restart it to make the standard mode take effect.
- Use the **sets color detection mode to ()** block and select the **Standard** mode.