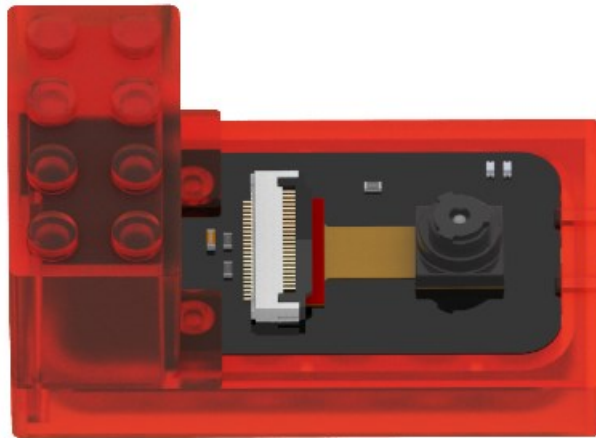


Image Recognition Sensor



www.weeemake.com

Introduce

The image recognition sensor is a small-sized, low-power primary level visual recognition sensor. With built-in recognition program, all recognition algorithms are processed locally, no need for network.

The image recognition sensor has the functions of **color recognition**, **size recognition**, **face tracking**, **shape blur recognition**, etc. It can be used to track small balls, human faces, identify azimuth distance, follow the line, drive automatically, sort automatically, and other cases.

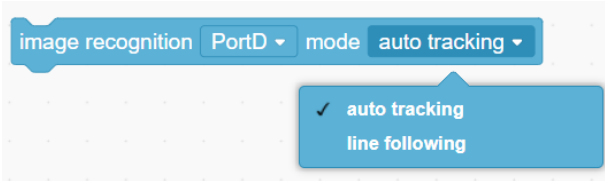
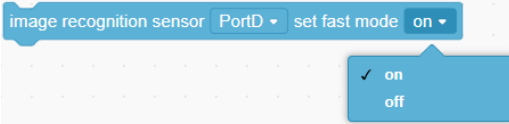
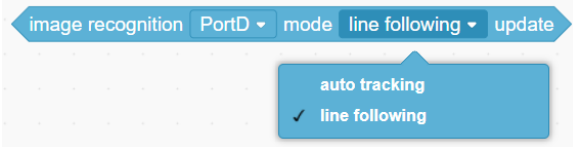
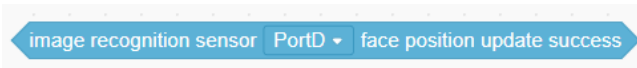
Users can freely input the color information they want to identify, communicate with any Arduino, micro:bit platform main controller through Weeemake 1-wire protocol, and can be programmed with Scratch, Python, Arduino C++, etc., which is convenient and fast.

This module contains a machine vision algorithm that feeds back the processed results back to the master but cannot transfer pictures or videos. It has a variety of output methods to meet different application needs, and supports parameter input and modification. Has the ability to automatically track the target in front. There are 2

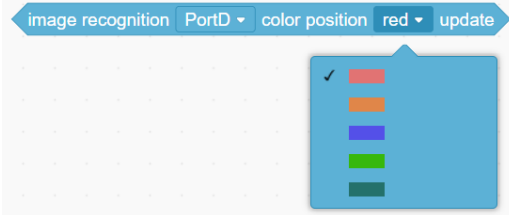
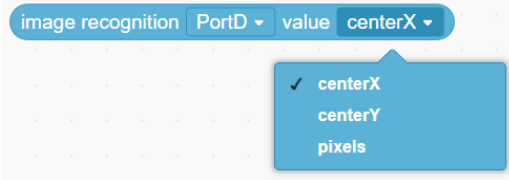

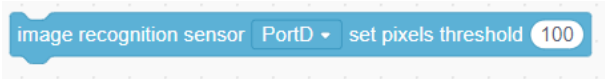
Specification

Parameter	Value/Description
Operating voltage	DC 5V(built-in 3.3V level shifting)
Wiring Port	RJ11
Communication	WM 1-wire
Processor	STM32 high performance processor, 180MHz,ARM Cortex M4 core
Camera	30W pixel camera, angle of view 66°, maximum frame rate 60fps, high sensitivity, lens loss compensation, automatic saturation adjustment, etc.
Indicator LED	When the camera is working, the blue LED flashes; it recognizes once and flashes once; when it is not working, it goes out.
Reset Button	When the module is not working properly, press the reset button shortly.
Power Consumption	120mA~200mA
Working Temperature	-20℃ ~70℃
Size	55mm*24mm*18.5mm(L*W*H)
Recognition Color Gamut	LAB color space
Preset Color Data	1-Red, 2-Yellow, 3-Blue, 4-Grass Green, 5-Dark Green
Color Recognition Rate	Up to 25 ms
Line-following Mode Output	Angle mode / Coordinate mode

WeeeCode 3.0 Guide

	<p>This block defines the working mode of image recognition sensor.</p> <ul style="list-style-type: none"> -Auto tracking: select this mode to track object set by yourself. When run this program, camera will automatically capture the object in front as the tracking target and only capture once. (Color of this object should be simple, and the background should be clean to identify the object.) -Line following: use this mode to read angle value and do line-following.
	<p>Use this block to turn on/off the fast mode. On default mode, the sensor will read value every 100ms. On fast mode the sensor will read the value quicker, please don't add any lag time under this mode or bug would happen.</p>
	<p>Use the block to check if the object or patrol line is in the camera capturing range, and if so, program keep running, otherwise 0 is returned and program stop running. (Boolean value)</p>
	<p>Use the block to check if a human face is in the camera capturing range, and if so, program keep running and the X/Y axis of face center position will be updated, otherwise 0 is returned and program stop running. (Boolean value)</p>

WeeCode 3.0 Guide

	<p>Use the block to check if the color you want to track is in the camera capturing range, and if so, return 1 and program keep running, otherwise 0 is returned and program stop running. (Boolean value)</p> <p>This block is mostly use to detect the ball in AI advanced kit.</p>
	<p>This block refers to the position axis of target object in camera. The axis X (0-320) and Y (0-240) value can tell the object's moving direction, the pixels value can tell the size in image and indicate the distance between the object to camera.</p>
	<p>This block refers to the angle value of image recognition sensor while working under line-following mode.</p>
	<p>Sets the minimum number of pixels to identify the object. Use this block, sensor will only recognize the object with pixels more than the set number, otherwise it will be neglected. This function can be used to filter out interference or to identify larger objects.</p>

WeeeCode 3.0 Guide

image recognition sensor PortD ▼ set traffic signs mode

Set the working mode of image recognition sensor to traffic signs mode, so it can recognize the preset traffic signs.

image recognition sensor PortD ▼ identify traffic signs Forward ▼

Use the block to check if the traffic sign you want to track is in the camera capturing range, and if so, return 1 and program keep running, otherwise 0 is returned and program stop running. (Boolean value)

✓ Forward
Backward
Left
Right
Park

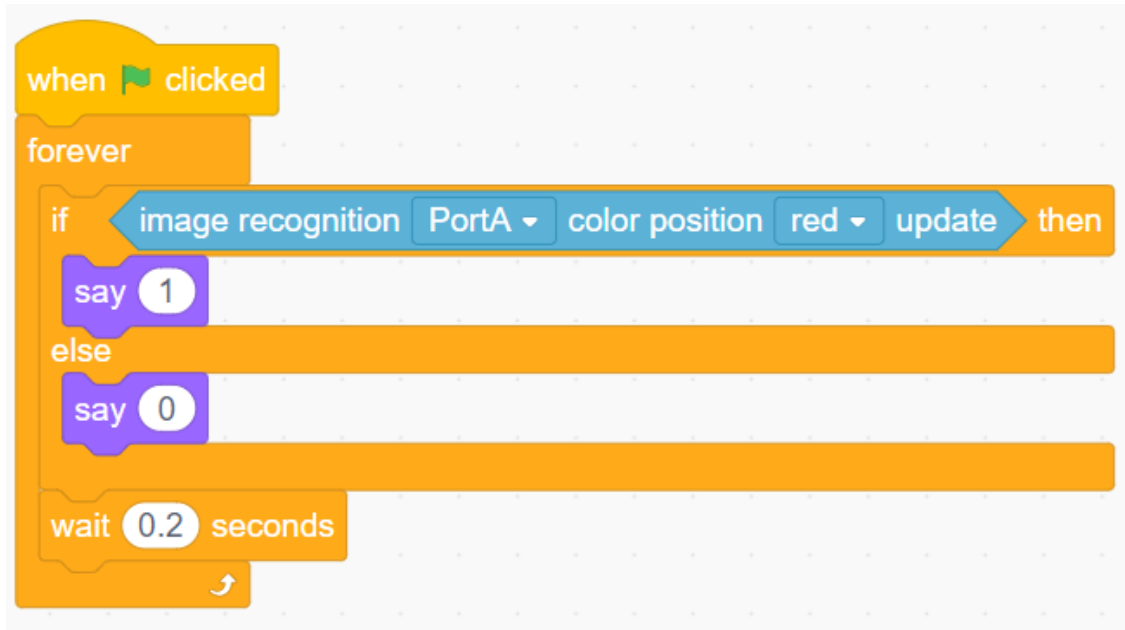


image recognition sensor PortD ▼ set LAB color 1 ▼ L min 0 L max 0 A min 0 A max 0 B min 0 B max 0

Sets the user-specified color information to identify the object. With the LAB color gamut, six values represent a range. The wider the range, the easier it is to identify, the better the environmental adaptability, but the possibility of misjudgment is also improved.

WeeeCode 3.0 Guide

Sample Program:



Arduino Guide

Arduino Library	Description
<code>bool getLab(void)</code>	Feedback LAB value in front of the camera. The updated data is [minL, maxL, minA, maxA, minB, maxB], which are the maximum and minimum values of the three components, respectively. This mode has an automatic capture function that locks the output only when the colors in the center of the camera are basically the same. Users can make appropriate modifications based on the LAB range of feedback to meet the identification in different environments.
<code>void setAutoTrackingMode(void)</code>	Sets the automatic acquisition mode. This mode first searches for the target object information in the front center of the camera. When the color of the target center area is basically the same, it locks and then tracks the object. This mode is similar to the way the puppy picks up the object. It first places the object in front of the eye and then looks for it. The program is simple and easy to control.
<code>bool getAutoPosition(void)</code>	Gets the object information in automatic mode. This function needs to be run in [setAutoTrackingMode] mode to get the X, Y axis and pixel size of the target object in front of the camera. When the camera captures the target object and returns to 1, no capture returns 0.
<code>bool getColorPosition(uint8_t mun)</code>	Gets the object information of the built-in color [mun]. This function captures one of the five built-in colors and appears in front of the camera. [mun] fills in 1-5, which corresponds to 1-red, 2-yellow, 3-blue, 4-grass green, and 5-dark green. When the camera captures the target object and returns to 1, no capture returns 0. When it returns 1, the X, Y axis and pixel size of the object are updated.

Arduino Guide

Arduino Library	Description
<code>void setLabColor1(int8_t minL, int8_t maxL, int8_t minA, int8_t maxA, int8_t minB, int8_t maxB)</code>	Sets the user-specified color information. With the LAB color gamut, six values represent a range. The wider the range, the easier it is to identify, the better the environmental adaptability, but the possibility of misjudgment is also improved. This function sets the color information of number 1. Similarly, [setLabColor2], [setLabColor3], and [setLabColor4] correspond to the color information of 2, 3, and 4 respectively.
<code>bool getAppColorPosition(uint8_t mun)</code>	Gets the object information of the user-specified color [mun]. This function is to capture the user-defined color, [mun] fills in 1-4, corresponding to the X serial number in setLabColorX. When the camera captures the target object and returns to 1, no capture returns 0. When it returns 1, the X, Y axis and pixel size of
<code>void setLineFollowerMode(void)</code>	Sets the line-following mode. In this mode, the camera will obtain grayscale information of 160*120 pixels, which is used to separate the black line and the white line.
<code>void setLineFollowerThreshold(uint8_t min,uint8_t max)</code>	Sets the grayscale threshold at which the line needs to be captured. If the background is white, the line is black, generally takes the value [0,100], the minimum value is 0, the maximum is 255, the pure black is 0, and the pure white is 255.
<code>bool getLineFollowerAngle(void)</code>	Get the line angle information. The obtained angle information is a positive or negative angle value, according to which the car is controlled to rotate left and right, and the value is within $\pm 53^\circ$. When the camera captures the line, it returns 1 and no capture returns 0. Note that this function needs to be used under the [setLineFollowerMode] function.

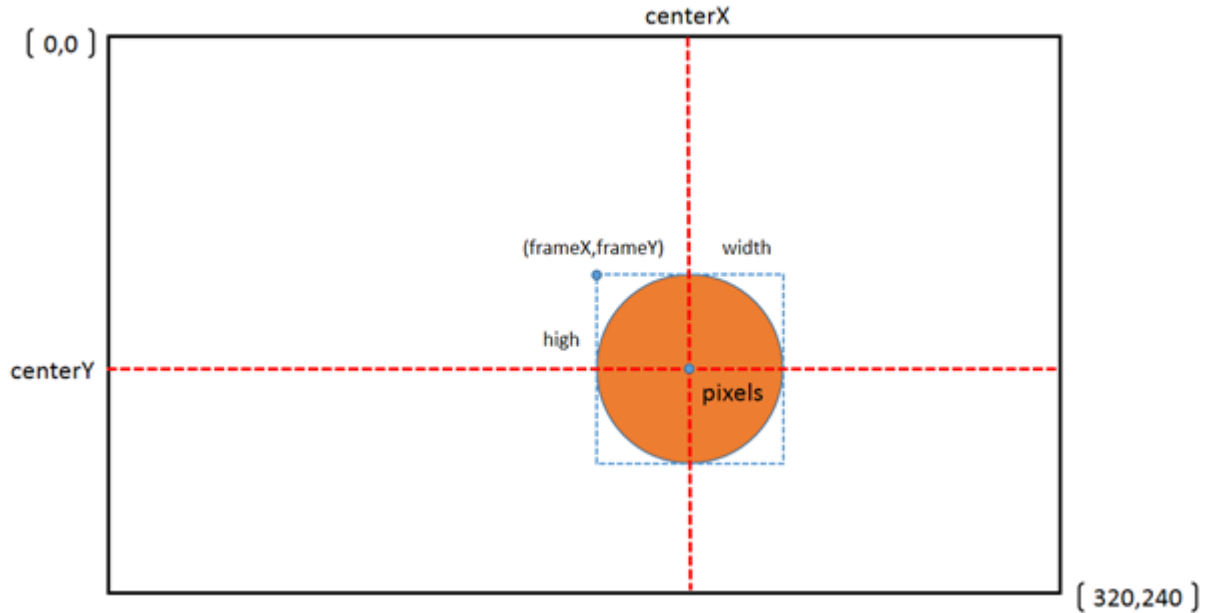
Arduino Guide

Arduino Library	Description
<code>bool getLineFollowerAxis(void)</code>	Get the line-following area information. This function updates five values representing the center X axis of the five regions arranged in order from far to near before the camera. These five regions represent the black line region. When the camera captures the line, it returns 1 and no capture returns 0. Note that this function needs to be used under the [setLineFollowerMode] function.
<code>void setPixelsThreshold(uint8_t mun)</code>	Sets the minimum number of pixels to identify the object. When the number of pixels identifying the object exceeds [mun], the feedback information will be received and recognized, otherwise it will be neglected. This function can be used to filter out interference or to identify larger objects. The default is 10.
<code>bool getColorAllPosition(uint8_t mun)</code>	Get all the information of the built-in color [mun] object. Similar to [getColorPosition], but with more information, including 8 items such as centerX, centerY, pixels, frameX, frameY, high, width, rotation, etc., representing the X axis and Y axis of the object center, the number of pixels, the X axis and Y axis of the object frame, the frame height, frame width, and object rotation angle. Due to the large amount of data, all recognition rates are slower than
<code>bool getAppColorAllPosition(uint8_t mun)</code>	Get all the information of the user-specified color [mun] object. Similar to getAppColorPosition, but with more information, including 8 items such as centerX, centerY, pixels, frameX, frameY, high, width, rotation, etc., representing X axis and Y axis of the object center, the number of pixels, the X axis and Y axis of the object frame, the frame height, frame width, and object rotation angle. Due to the large amount of data, all recognition rates are slower than getAppColor-

Arduino Guide

Arduino Library	Description
<code>void resetColorMode(uint8_t time)</code>	Reinitializes object recognition (not line-following). The parameter [time] is the exposure time. The shorter the time, the darker the picked-up image is, which is suitable for high-brightness environment. The longer the time, the brighter the picked-up image is, suitable for a darker environment. Time is in milliseconds and defaults to set 2000, which is suitable for dark environment.
<code>bool getFacePositon(void)</code>	Gets the face location information. When the camera captures a face, it returns 1 and no capture returns 0. When it returns 1, the X and Y coordinate values of the face center point are updated.

Data Information



Note:

When an identified object is not a regular shape, the center point focuses on the center of gravity.

LAB Color Space

The **CIELAB color space** (also known as **CIE $L^*a^*b^*$** or sometimes abbreviated as simply "Lab" color space) is a color space defined by the International Commission on Illumination (CIE) in 1976. It expresses color as three values: L^* for the lightness from black (0) to white (100), a^* from green (-) to gray to red (+), and b^* from blue (-) to gray to yellow (+). CIELAB was designed so that the same amount of numerical change in these values corresponds to roughly the same amount of visually perceived change.

Value range:

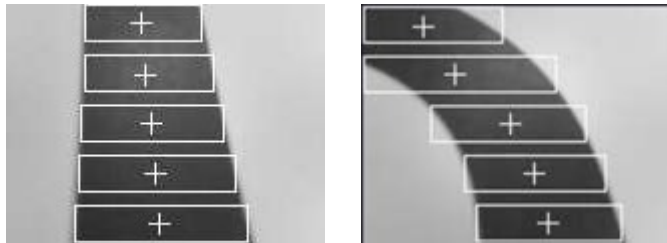
L [0,100]

A [-128, 128]

Line-following

The image recognition sensor has 2 ways to follow the line.

1. Give the angle value directly, angle range is **-53 to 53** degrees, the user can control the car rotation according to this angle.
2. Give 5 values, as shown in the picture below. On the grayscale map, divide in 5 regions equally, each area will look for the center point of the black line, return the X axis of the five center points. Linex1 to Linex5 represent those 5 center points from top to bottom, that is, from far to near. According to these values, it can be judged whether the map is a straight line or a curve. If the posture of the car is consistent, a weighting formula can be written to determine whether a turn is needed and the turn size. By setting the threshold value in the [setLineFollowerThreshold] function, you can change the gray value of the black line to meet the needs of different colors of the line map.



Range X[0,160], Y[0,120]

In this mode, a visual recognition line-following car or a simulated driverless car can be made. It should be noted that the speed of image recognition for the line-following is not fast, and the judgment is as fast as 50ms. So compared with the traditional line-following sensor, the image recognition sensor has no advantage in speed. According to the map, the

Application Skill

1. Calibration field of view

Since this module does not have the function of outputting images and videos, in actual application, the field of view of the camera is not known, and it is not convenient to debug. At this time, the field of view needs to be measured.

Measuring method: using a small ball of a predetermined color, slowly approaching the center from 4 corners, when the ball is captured at the edge position, and after the data is output, it is the boundary of the horizon. After adjusting the proper field of view, fix the camera, which will facilitate further debugging in the future.

2. A maximum of 2 objects of the same color can be recognized at the same time.

In the camera range, the sensor can recognize up to 2 objects with same color at one time. If there is a lot of fragmented color interference in the background, you can set the [setPixelsThreshold] to identify the minimum number of pixels in the object, so you can directly filter out some small color block interference.

3. Face recognition

Face recognition uses a grayscale image of 240*160 size, so the face tracking effect is better when the face from the camera is 20cm or more. The module can only recognize one face at the same time. The face needs to be facing the camera. Side-faced, or squatting, will influence the recognition effect. This algorithm only supports the recognition of the entire face, and does not support the recognition of half or partial faces. The fastest recognition speed is 70ms at a time.

Tips

1. Changes in the environment will have different effects on the results of image recognition. When using the sensor, you should pay attention to:

- a. Avoid environments that are too dark, too bright, or strong backlighting;
- b. Avoid reflection of strong light or sunshine from target object;
- c. Avoid colored lights or convertible lights. Stable and uniform white is the best light source;
- d. Keep the camera from facing the light source;
- e. Avoid complicated background patterns, or the target object and background color are similar;

2. Boot time. It takes about 3s for the image recognition sensor to receive commands from power-on to start receiving. Therefore, it is necessary to delay the camera parameters for more than 3s when setting the camera parameters on the main program, to ensure that the camera module enters the working state, or reset the main controller after power-on.

3. The image recognition sensor is generally continuously called and used in a non-stop working state. If it is off for a long time, it may be feed the last unreceived data in the data cache back. In this case, it is best to run more than 2 times to eliminate residual interference.

4. If the face recognition program or the line recognition program is turned on and you want to open the program for