MakeCode Car Tutorial with the AI Vision Module

Preface

Our Company

ACEBOTT STEM Education Tech Co.,Ltd

Founded in China's Silicon Valley in 2013, ACEBOTT is a STEM education solution leader We have a team of 150 individuals, including members from research and development, sales, and logistics. Our goal is to provide high-quality STEM education products and services to our customers. We are working together with STEM education experts and our business partners to produce successful STEM products together Our self-own factory also provides CEM services for our clients including Logo customization on product packaging and PCB.

Our Tutorial

This practical course for beginners aims to introduce students to the world of programming, electronics, and robotics. It's based on the AI vision module. In this course, students will learn the control theory and practical applications of the AI vision module and practice using a smart car equipped with the AI vision module.

With this kit, you can:

1. Master the hardware structure and functional features of the AI vision module.
2. Understand the basic concepts and practical application scenarios of machine vision.
3. Learn the image processing methods of the AI vision module.
4. Deeply understand the implementation principles of the AI module for color recognition, face recognition, Traffic sign recognition, and intelligent line patrol
5. Complete an AI vision car project using the ACEBOTT kit to enhance maker skills.

This ACEBOTT AI vision module learning kit is designed for beginners. Through systematic learning, you will master the basic working principles and

usage of the AI vision module and be able to flexibly apply what you've learned to solve real-world problems.

Customer Service

ACEBOTT is a dynamic and fast-growing STEM education technology company that strives to offer excellent products and quality services that meet your expectations. We value your feedback and encourage you to drop us a line at **support@acebott.com** with any comments or suggestions you may have.

Our experienced engineers are dedicated to promptly addressing any problems or questions you may have about our products. We guarantee a response within 24 hours during business days.

Follow Us

Scan the QR codes to follow us for troubleshooting & the latest news.

We have a very large community that is very helpful for troubleshooting and we also have a support team at the ready to answer any questions.

ACEBOTT FB QR Code

YouTube QR Code

Contents

Lesson 1 Understanding the Vision Module

Ⅰ. Concepts and Applications of Machine Vision

Machine vision is a technology that enables computers to "see" and understand images or videos like humans. It uses sensors such as cameras and lidar to collect visual information, then uses algorithms to analyze objects, colors, shapes, and even motion trajectories in the image to ultimately perform recognition, measurement, or decision-making tasks.

The core processes of machine vision include: 1. Image acquisition (taking photos and recording videos) → 2. Image processing (enhancement and noise reduction) → 3. Feature extraction (finding key information) → 4. Analysis and decision-making (recognizing objects or executing actions).

For example, suppose you want a machine to identify whether an animal in front of it is a cat or a dog. First, you need to take a photo of the animal. After receiving the image, the machine needs to perform some processing, such as enhancing the brightness and cropping out unnecessary background, to facilitate subsequent identification. Then, the machine's algorithm extracts key animal features, such as ears and nose, and finally, based on these features, determines whether it is a cat or a dog.

In general, machine vision uses pattern recognition and algorithms to achieve object detection, classification, or decision-making through image acquisition, image processing, feature extraction, and classification decisions.

Ⅱ. AI Vision Module Application Scenarios

Vision modules have numerous applications in daily life, for example:

1. Smart Retail: Installing a vision module on a traditional shopping scale can automatically detect vegetables and fruits that need to be weighed.

2. Smart Security: Installing cameras on door locks allows facial recognition to replace traditional keys for door opening. Smartphones and laptops can also be unlocked using similar methods.

3. Smart Driving: Cars equipped with multiple cameras can automatically identify traffic elements such as vehicles on the road, road signs, and traffic lights, enabling autonomous driving.

Intelligent Retail Face recognition Autonomous driving

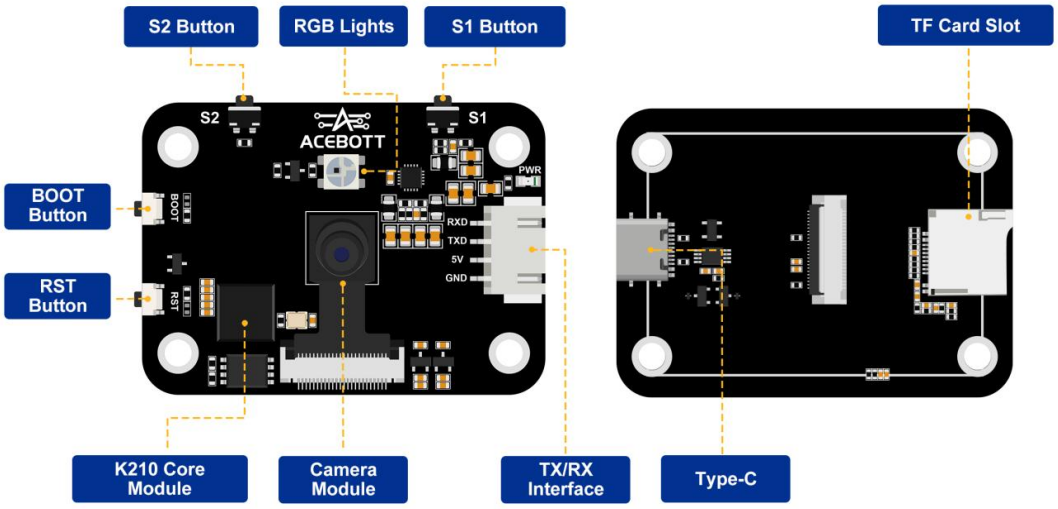
Ⅲ. Introduction to the AI Vision Module

The AI vision module is a powerful AI vision processing module with a built-in neural network hardware accelerator (KPU). It efficiently executes convolutional neural network operations, enabling high-performance machine vision and audio processing. This module supports 15 visual functions, including face recognition, QR code recognition, feature point detection, number recognition, color recognition, license plate recognition, Traffic sign recognition, and visual line patrol.

- **Buttons S1 and S2**: Customizable button functions.
- **RGB light:** Programmable to display red, green, blue, and white. Used for camera recognition and fill light, improving recognition accuracy.
- **TF card slot:** used to insert a TF card, with the gold finger

facing the module, to save programs, models, and other files.

- **Fixed Copper Pillars:** Four fixed copper pillars, M3 size.
- **AI Vision Module Core Module:** Contains the minimum operating system for the K210 chip.
- **Camera:** captures images.
- **RST Button:** Reset button, used to restart the AI vision module.
- **BOOT Button: Loads the startup program.**
- **TX/RX port**: Connects to the mainboard and transmits data.
- **Type-C Port**: Download firmware, serial port debugging, connect to the IDE, etc.



Ⅳ. Checking the AI Vision Module Firmware Version

Connect the AI vision module to your computer using a Type-C cable. The firmware version will be displayed on the startup screen. To improve performance, we recommend upgrading to the latest firmware. If your version is prior to V1.0.2, you need to update to V1.0.2.

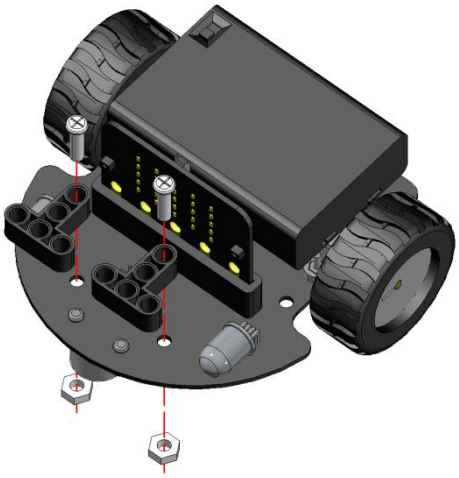
For update methods, please refer to the "0. Must Read Before Use" document in the "English\MakeCode car" directory. The second section explains how to update the firmware.

V. Steps for Building a Small Car

To enable the car to have recognition capabilities, you can combine building blocks to quickly build the basic structure of the car. The following are the building steps:

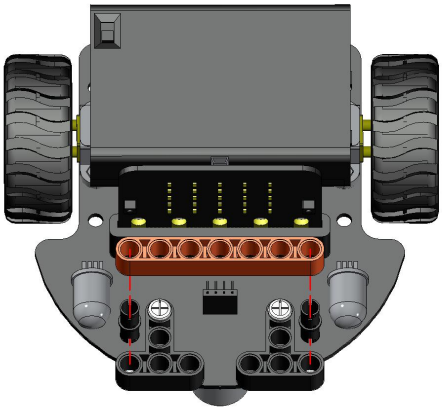
1. Install the T-Beam:

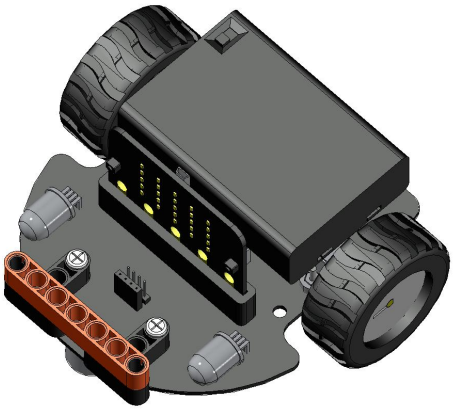
Name	Quantity
T-Beam	2
M4*14mm Round Head Screws	2
M4 Nut	2



2. Installing the Straight Beam:

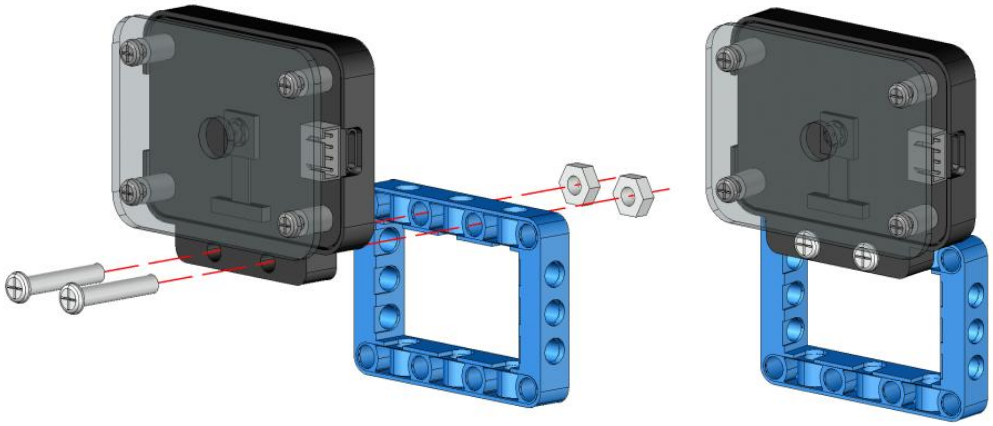
Name	Quantity
7-Hole Straight Beam	1
Friction Pin	2





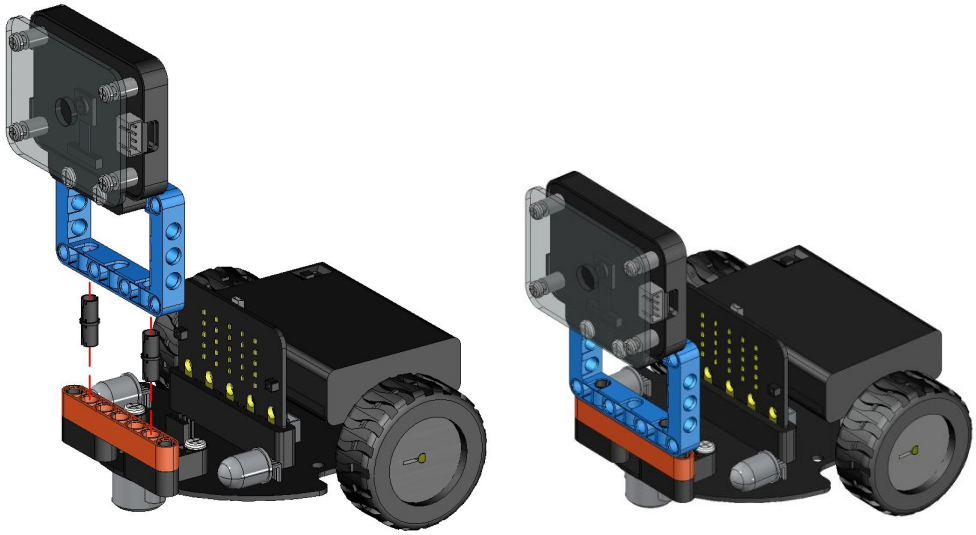
3. Install the AI vision module:

Name	Quantity
5x7 Frame Beam	1
AI Vision Module	1
M4*22mm Round Head Screws	2
M4 Nut	2



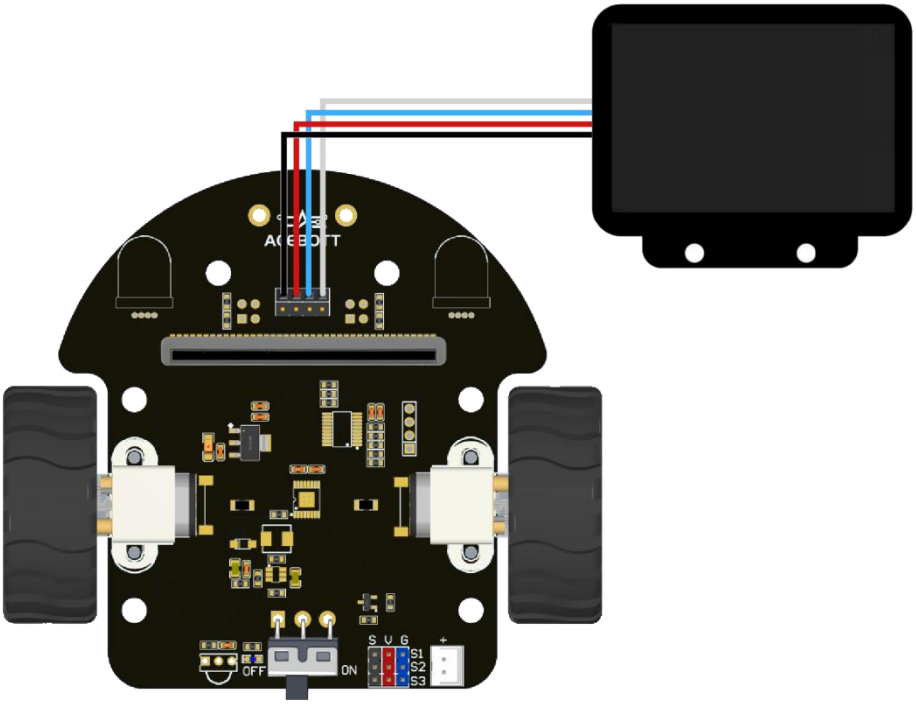
4. Securing the AI vision module:

Name	Quantity
Friction Pin	2



5. Connect the AI vision module to the car as shown in the diagram:

Name	Quantity
4Pin to DuPont Cable	1

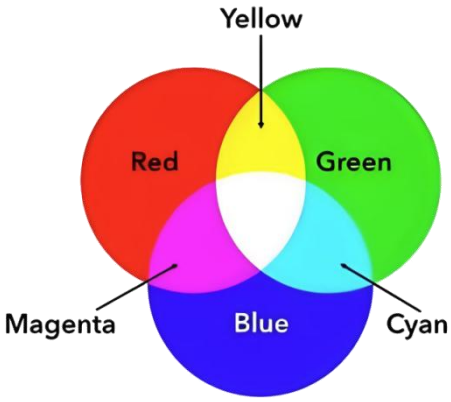

OFF
ON
S V G
S1
S2
S3
+

Lesson 2 Color Recognition

Ⅰ. Principles of Color Recognition

Before learning the principles of color recognition, you need to understand the three primary colors of light.

The three primary colors of light are red, green, and blue. These three colors cannot be further decomposed and can form other colors, so they are called primary colors. These three colors can be combined to form various other colors.



Additive color

For machines, the core of color recognition relies on the optical RGB primary colors (Red, Green, and Blue). A camera receives the RGB values of a color through a photosensitive element, and each element detects the brightness of red, green, and blue respectively. Each color is represented by a number from 0 to 255, representing the intensity of red, green, and blue, allowing for over 16 million color combinations.

For example, if a self-driving car sees a red light, the values it reads are: R=255, G=0, B=0, indicating that the light is red and the car must stop.

If a green light is detected, the values are: R=0, G=255, B=0, indicating that the system is starting.

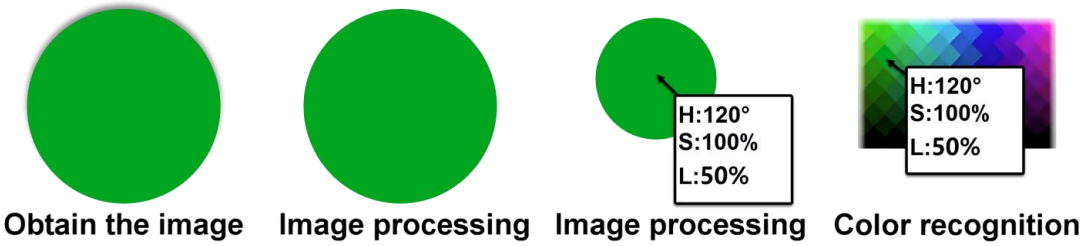
Color recognition by a machine involves multiple steps, which can generally be divided into:

1. Image Acquisition: Use a camera or other image acquisition device to acquire the target image.

2. Image Processing: Adjust the image and convert it into the HSL model (the model that best aligns with human intuition).

3. Feature Extraction: Extract features such as hue, saturation, and brightness from the HSL model.

4. Color Recognition: Identify colors in the image based on extracted features.



Ⅱ. Application Scenarios for Color Recognition

Like vision modules, color sensors play an important role in multiple fields:

1. Industrial Sorting: On automated production lines, color sensors can quickly identify and sort products of different colors, improving production efficiency and quality control.

2. Product Quality Inspection: In industries like electronics, automotive, and pharmaceuticals, color sensors are used to detect product color consistency and ensure that product quality meets standards.

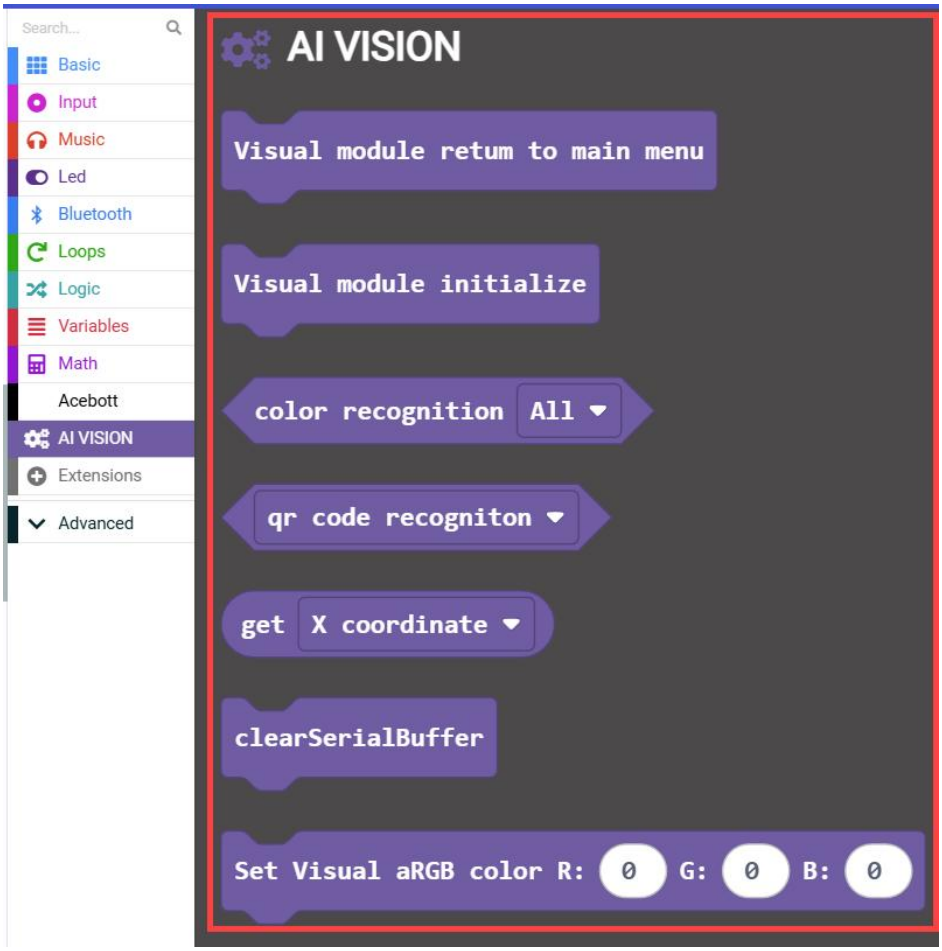
3. Robot Vision: In robotics, color sensors provide robots with visual perception capabilities, enabling them to identify and manipulate objects of different colors.

Ⅲ. AI Module Color Recognition

1. Learning Programming Blocks

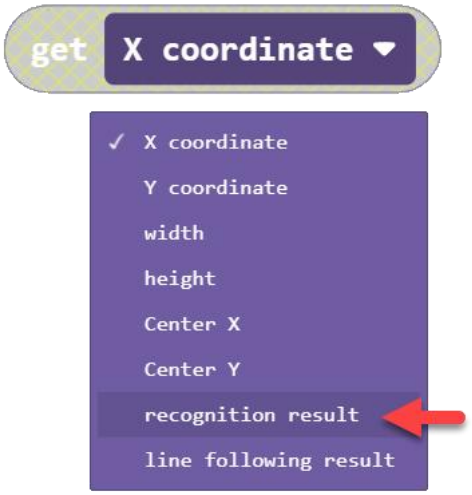
Before programming, familiarize yourself with this lesson's code blocks. First, import the Acebott extension on the MakeCode website. To do this, open the "Method for Adding MakeCode Extensions" file located in the "English\MakeCode car\2.MakeCode Extension Tutorial" folder for reference. After completion, the AI

vision module programming blocks will appear under the "AI VISION" category, as shown in the example image.



There are some blocks you need to learn here. The blocks used in this lesson are as follows:

Blocks	Usage
	Vision module initialization: Before using the various functions of the vision module, you must initialize it.
	Perform color recognition. Currently, only red, green, and blue are supported. Returns a Boolean value of "True" when a color is detected; otherwise, returns "False."

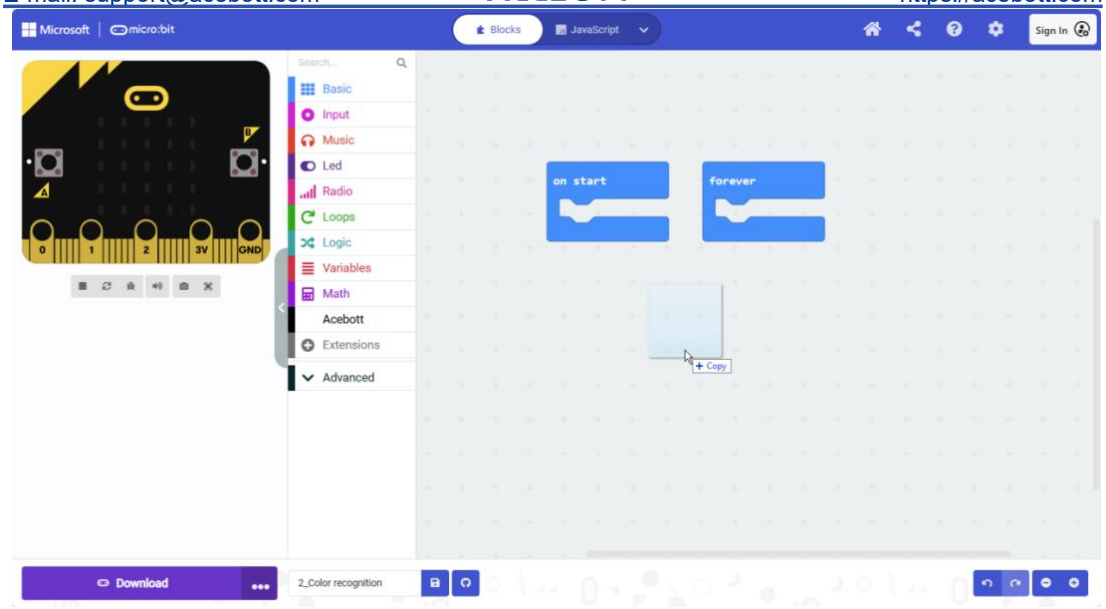
	The "get" block is used to obtain the recognition result. Click the small white triangle to switch to "recognition result."
	Because the vision module returns recognition results via the serial port, various reasons may cause old data to remain in the buffer when the Micro:bit receives module data. This block discards old data and ensures that the latest data is returned.
	Used to turn on the camera's fill light. The three parameters correspond to RGB color values and are generally set to 255 when turned on.

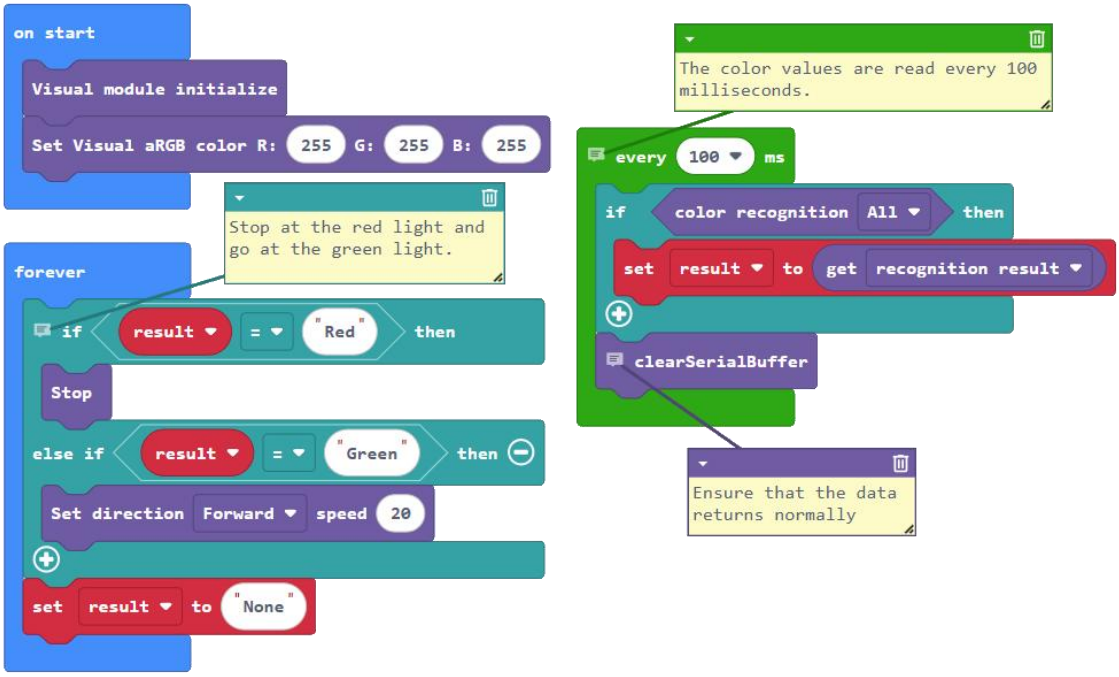
2. Color Recognition Program

This lesson's program enables the module to recognize red and green cards. When green is detected, the car moves forward; when red is detected, the car stops.

To open the program, navigate to the "English\MakeCode car\4.Program" folder, locate the "2.Color_recognition.hex" file, and drag it into the MakeCode programming area.

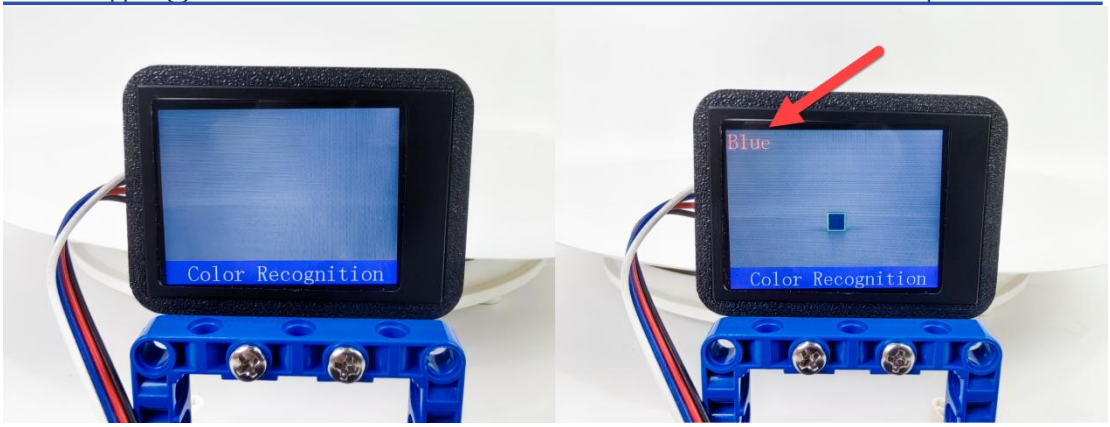
Open the program in the programming area.





3. Program Results

After uploading the program, you'll notice the module interface changes to color recognition mode, with the recognized color appearing in the upper-left corner. Currently, only red, green, and blue are supported.



Note: If there is no response after uploading the program, press the reset button on the Micro:bit V2.

If the module recognizes green, it moves forward, and only stops if it recognizes red.

Lesson 3 Color Tracking

Ⅰ. Color Tracking Process

In addition to detecting colors, the Vision Module can also track colors and perform specific tasks.

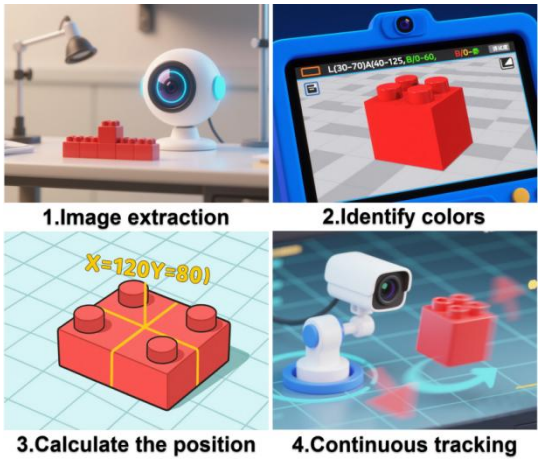
Color tracking primarily involves the following steps:

1. Image Extraction: Use a camera to capture real-time images or read video files, converting the real scene into a digital image that can be processed by a computer. Multiple frames can be captured per second.

2. Color Recognition: Convert the image from RGB to LAB HSV color space (which can be understood as a color space closer to human vision) and filter pixels of the target color based on a preset threshold range.

3. Calculate Position: Perform contour detection on the identified color area and calculate its geometric center coordinates (X, Y) as the tracking coordinate point.

4. Continuous Tracking: By looping through each image frame, dynamically updating the target position, and continuously tracking color using the motion structure.

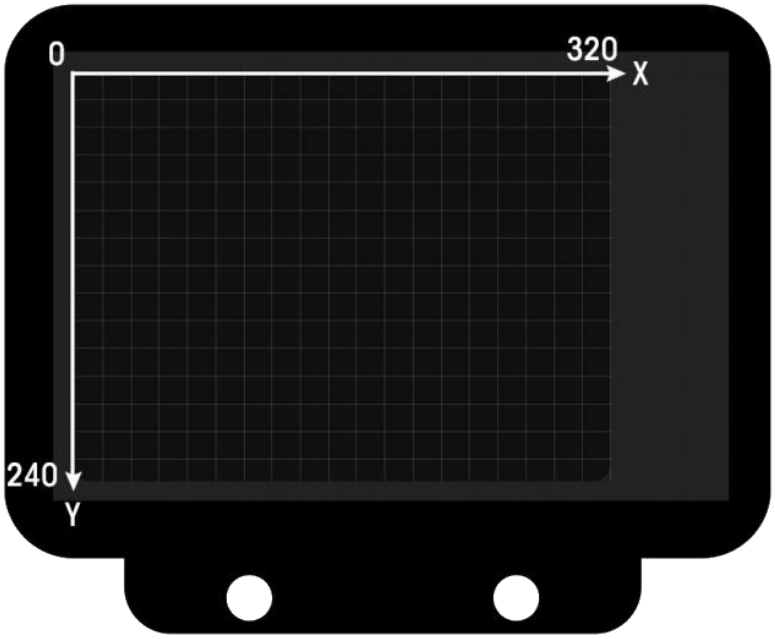


Ⅱ. Color Tracking Principles

1. AI Vision Module Screen Coordinates

The screen coordinates of the AI vision module are based on the upper left corner as the origin (0,0). The horizontal axis is the X axis, with a maximum value of 320, and the vertical axis is the Y axis, with a maximum value of 240.

The AI vision module will select the object to be tracked in real time, and when the coordinates of the object relative to the screen change, it will return the new coordinate values in real time, and the AI car will execute the corresponding motion instructions based on the coordinate changes.



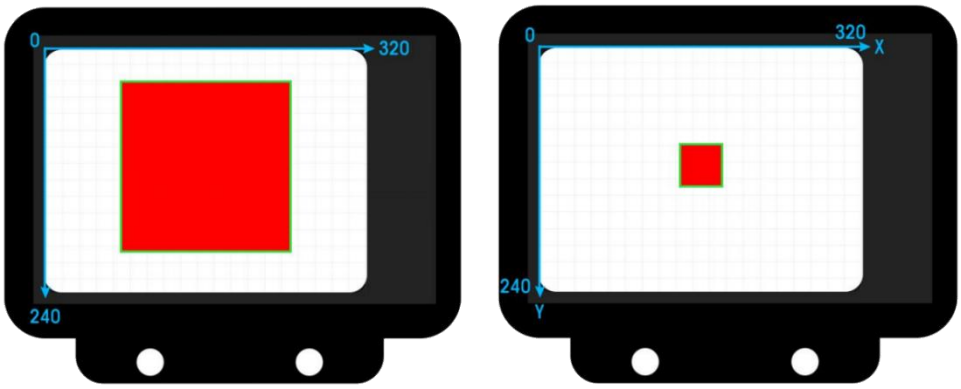
2. AI Car Color Tracking Logic

(1) AI car's forward and backward movement

When an object approaches the AI vision module, the green bounding box surrounding it on the screen will enlarge. At this point,

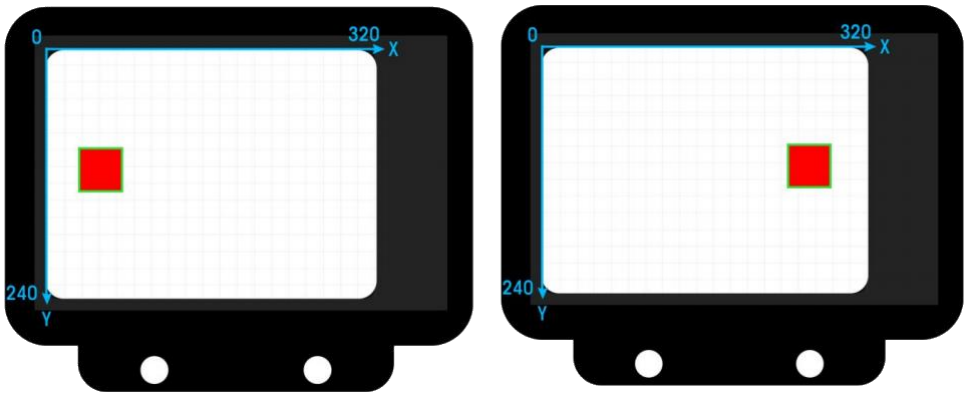
the program will control the AI car to move backward until the bounding box returns to its original size.

When the object moves away from the AI vision module, the green bounding box on the screen will shrink. The program will then control the AI car to move forward until the bounding box is restored to its initial size.



(2) AI car's left and right rotation

When the object shifts to the left on the screen, the X coordinate value of the object's green box will decrease, and the program will control the AI car to rotate to the left until the object's box remains in the center of the screen. When the object shifts to the right on the screen, the X coordinate value of the object's green box will increase, and the program will control the AI car to rotate to the right until the object's box remains in the center of the screen.



Ⅲ. Color Tracking Program

1. Learning Programming Blocks

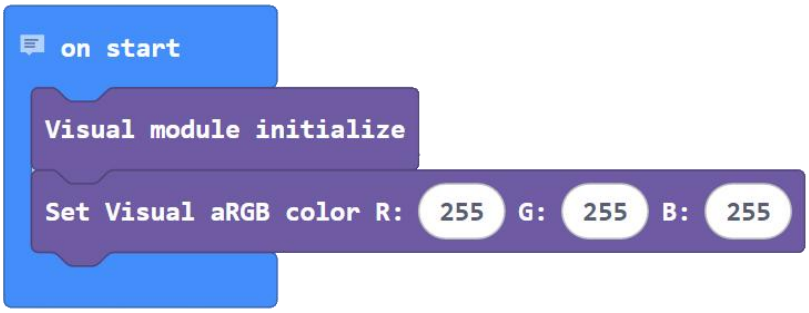
Blocks	Usage
	The "get" type block is used to return the horizontal coordinate value of the recognized object's center. The returned data type is uniformly string type.
	The "get" type block returns the height of the object, with the data type being string.
	The "get" type block returns the width of the object, with the data type being string.

2. Color Tracking Program

This lesson's program allows the module to track a red card, controlling the robot to move forward, turn left or right, and stop based on the card's position and size.

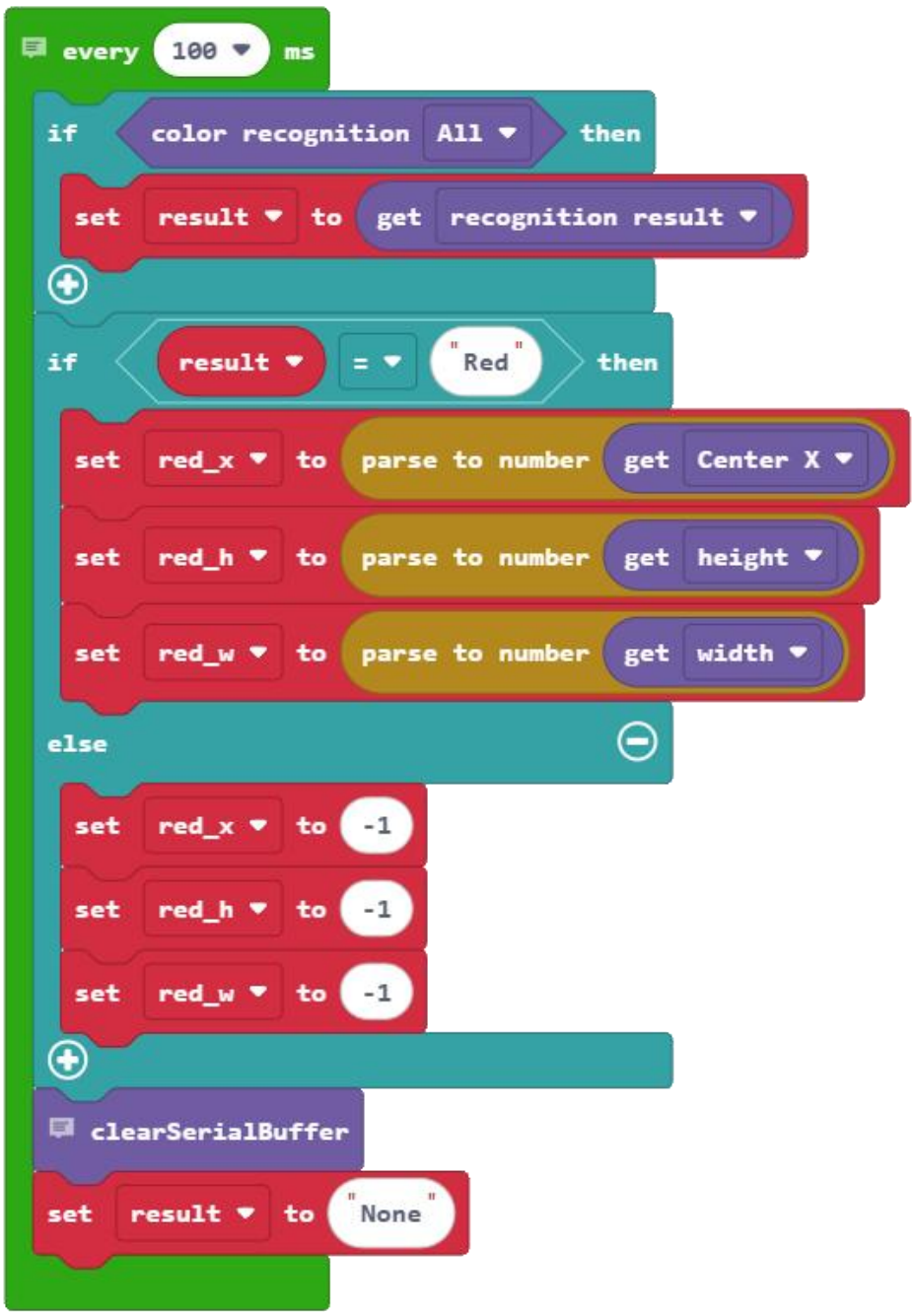
To open the program, navigate to the "English\MakeCode car\4.Program" folder, locate the "3.Color_tracking.hex" file, and drag it into the MakeCode programming area.

Upon opening, the program is divided into three parts:
The first part is the initialization program, which initializes the vision module and turns on the fill light.



Note: Whether to enable the fill light depends on the recognition results and is not always required. If you find that recognition is better without the fill light, turn it off.

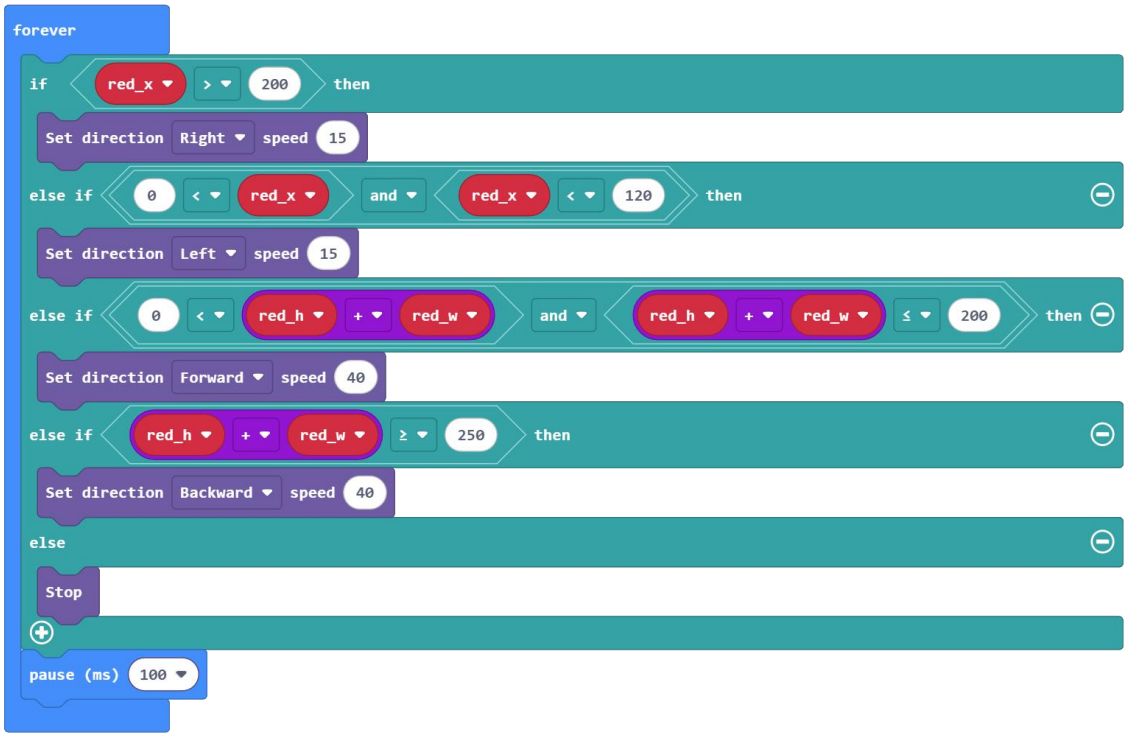
The second part is to obtain color location information. Get the recognition result every 100 milliseconds. If the result is red, get the color's width, height, and x-coordinate. Note that all values are converted to numeric values and stored in corresponding variables. If no color is detected, all related variables are set to -1, and the car will not respond. Finally, clear the serial port buffer and reset the "result" variable.



The third part involves the car performing color tracking based on position information. If the horizontal coordinate is greater than 180, the car continues to turn right; if it is greater than 0 and less than 180, the car turns left; otherwise, the car stops moving.

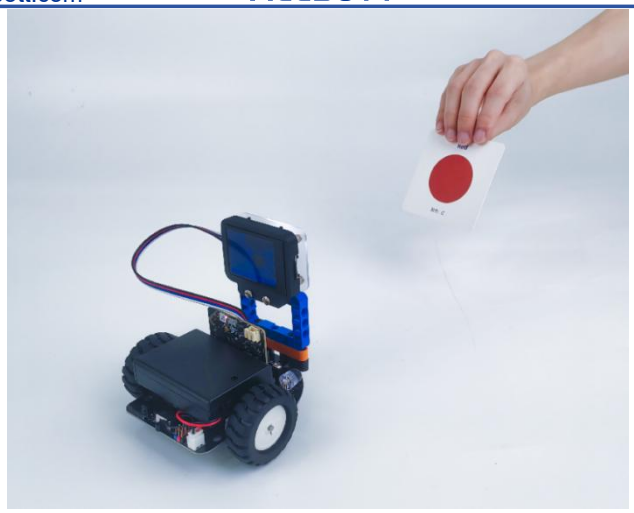
If the cumulative value of the width and height of the object being recognized is between 0 and 250, the car moves forward; if it

exceeds 250, it reverses. If none of the above conditions are met, the car stops.



3. Program Results

After the program upload is complete, place a red card in front of the AI vision module's camera. The screen will display a box around the card's color. When the card moves away from the AI vision module, the car will move forward. When the card moves closer to the AI vision module, the car will move backward. When the card moves to the left, the car will rotate left. When the card moves to the right, the car will rotate right.



Lesson 4 Face Recognition

Ⅰ. Principles of Face Recognition

Face recognition is already very common in our daily lives. It is a biometric technology that automatically identifies people by analyzing facial features (such as the shape and position of the eyes, nose, and mouth). Combining internal data and intelligent algorithms, it can quickly determine whether a face is present in an image and who the person is.

Facial recognition is performed by a machine in five key steps:

1. Image Input: The computer receives a photo or video frame captured by a camera and converts it into a digital signal for processing.

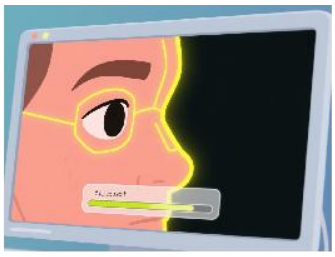
2. Feature Analysis: The algorithm scans the image, detecting features such as changes in brightness and edges, focusing on key areas such as the eyes, nose, and mouth.

3. Pattern Matching: The extracted features are compared with a pre-trained face model, calculating similarity and determining whether they match the facial features.

4. Location Determination: When the matching degree in a certain area exceeds a set threshold, the algorithm marks the location with a rectangular frame, indicating that a face has been detected.

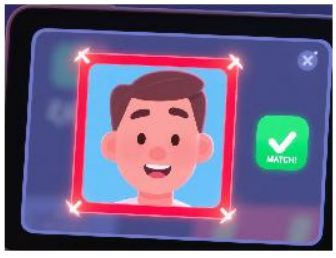
5. Result Output: The coordinates of all faces are ultimately output for subsequent applications (such as face detection and cartoon effects).

1. Image Input



2. Feature Analysis

3. Pattern Matching



4. Position Determination

5. Result Output

Ⅱ. Applications of Face Recognition

1. Phone Unlock: Many phones now support facial recognition, such as iPhone's Face ID or Android phones' 3D facial recognition. It's more convenient than a password; just a glance at your phone unlocks it. It's also highly secure, making it difficult for someone to impersonate you.

2. Payment Verification: Apps like Apple Pay and Google Pay have introduced "face payment" features, allowing you to pay at

supermarkets or restaurants without even pulling out your phone. The system compares your face with the linked account information to ensure it's your real self and prevent fraudulent transactions.

3. Access Control System: Some schools, companies, or communities use facial recognition instead of traditional access cards or fingerprints, making entry and exit more convenient. For example, when students arrive at school, they can use facial recognition to clock in, and the system automatically records their attendance, eliminating the need for teachers to manually call the roll.

4. Entertainment Filters: The funny special effects in apps like TikTok and Snapchat (such as changing to animal faces and adding virtual glasses) are all achieved through facial recognition. It accurately tracks your facial features, allowing the effects to follow your expressions, making video recordings even more fun.

1.Phone Unlocking

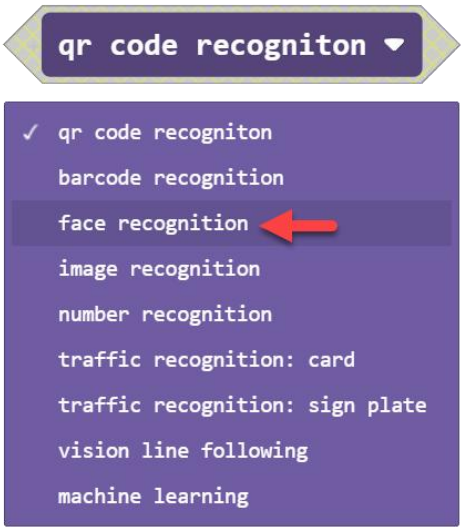
2.Payment Verification

3.Access Control Systems

4.Entertainment Filters

Ⅲ. Using Facial Recognition to Start the AI Car

1. Learning Programming Blocks

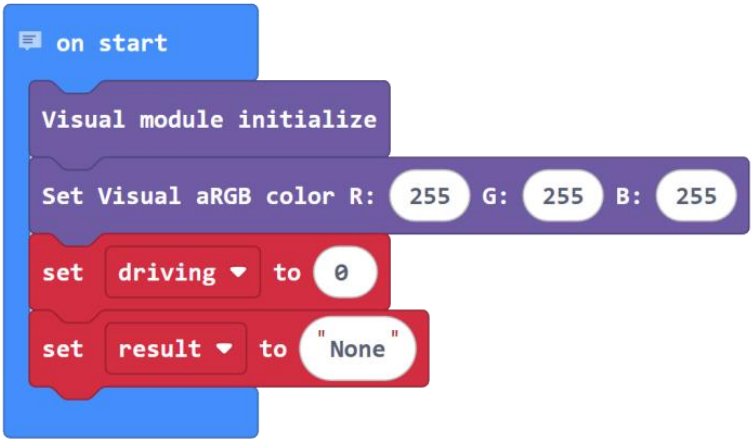
Blocks	Usage
	The "recognition" type block: Click the small triangle on the block to select face recognition mode. Returns "True" if a face is detected, otherwise "False".
	The "get" type block: When used with "face recognition", it returns facial data information. The returned information is displayed in the upper left corner of the vision module screen.

2. Start the Car Program with Face Recognition.

This lesson's program enables the module to recognize face cards. Upon detecting a face, it will drive in a circle and then stop.

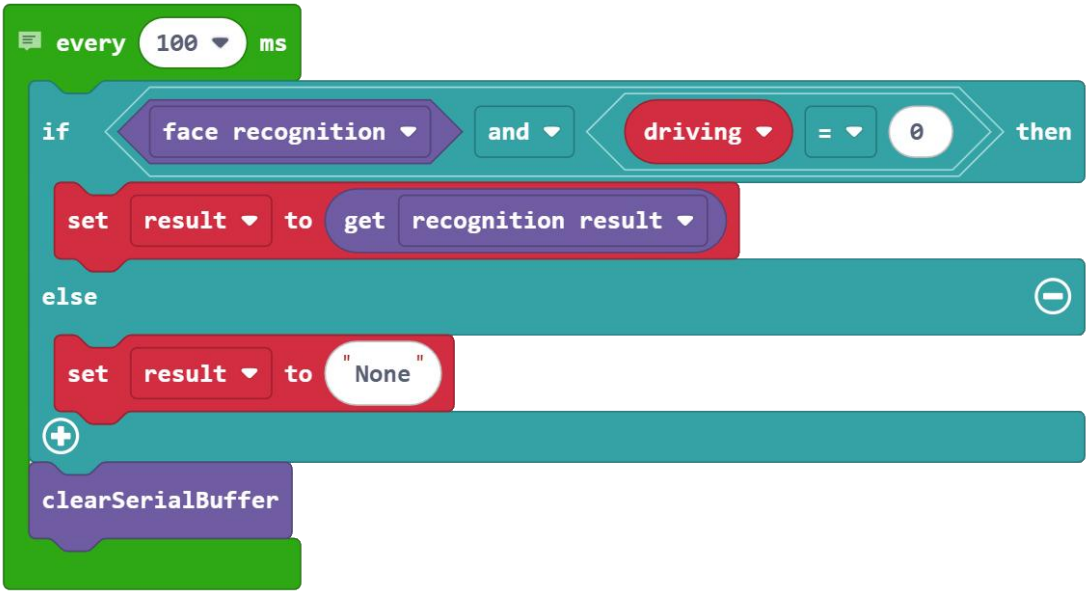
To open the program, navigate to the "English\MakeCode car\4.Program" folder, locate the "4.face_recognition.hex" file, and drag it into the MakeCode programming area.

Upon opening, the program is divided into four parts:
The first part is the initialization program, which sets up modules and variables with initial configurations.

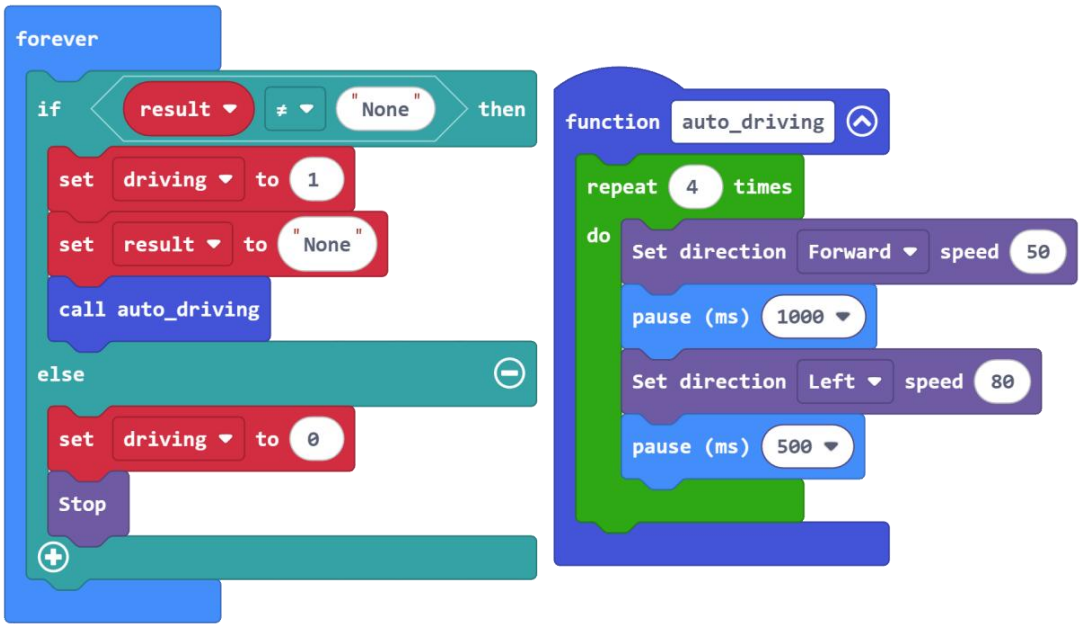


Note: Whether to enable the fill light depends on the recognition results and is not always required. If you find that recognition is better without the fill light, turn it off.

The second part is to obtain face data. If a face is recognized, the data is stored in the "result" variable. Information is only stored when the "driving" variable is equal to 0 to prevent the car from performing face recognition while driving. If no face is detected, the variable data is cleared.



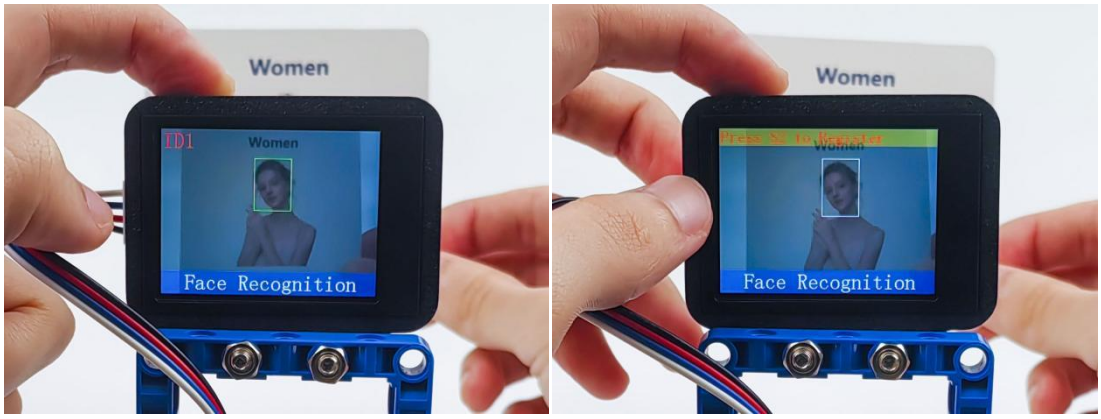
The third and fourth parts are the car's driving program. If the "result" variable is not "None," the car starts driving in a circle; otherwise, it stops.



3. Program Results

After the program is uploaded, you need to first identify specific facial information. For example, hold a "woman" card up to the camera. After recognizing the face, press the S2 button on the vision module to enter the data. After entering the data, the module will start the car program when it recognizes the same face again. You can see that the face is labeled ID1.

To clear the module's facial data, press and hold the S1 key. The screen will freeze slightly, and then the facial data will be cleared.



Lesson 5 Traffic Sign Recognition

Ⅰ. Principles of Traffic Sign Recognition

Traffic sign recognition technology combines computer vision and machine learning techniques and is primarily used in autonomous driving, driver assistance systems, and intelligent transportation. Traffic sign recognition relies on multiple technologies, including image acquisition, preprocessing, sensors, object detection, and deep learning. Deep learning, in particular, plays a crucial role. With technological advancements, the accuracy and real-time performance of Traffic sign recognition systems have continuously improved, making them a crucial component of autonomous driving and intelligent transportation systems.

The main process of artificial intelligence recognizing road signs can be divided into four steps:

1. Image Acquisition: The AI uses cameras (like the car's eyes) to capture photos or videos of the road. These cameras can be forward-facing, rear-facing, or surround-view, capturing environmental information from multiple angles.

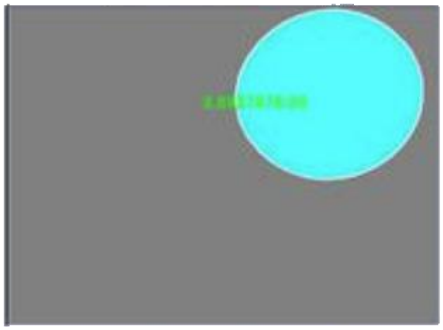
2. Image Preprocessing: The AI "beautifies" the photo by adjusting brightness and removing noise (like wiping the glass in front of you), making road signs more visible.

3. Road Sign Detection: The AI searches for possible road signs in the photo, just like finding distinctive shapes in an image. Common methods include edge detection (finding contours) and morphological operations (enhancing connectivity).

4. Classification and Recognition: The AI compares the found shape with the thousands of road signs it has learned to determine the type of sign.

1.Image Acquisition

2.Image Preprocessing



3.Road Sign Detection

4.Classification and Recognition

Ⅱ. Applications of Traffic Sign Recognition

1. Traffic Sign Recognition (TSR)

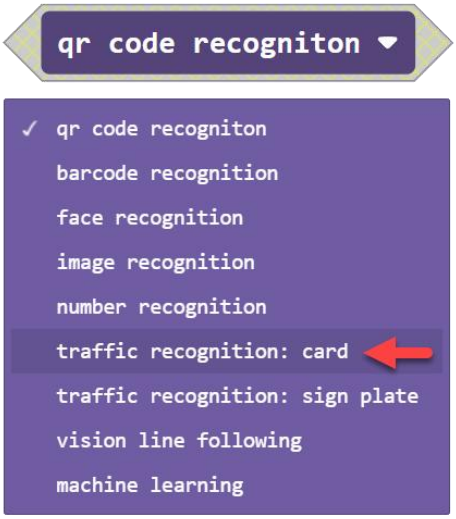
The TSR system uses a forward-facing camera to identify traffic signs on the road, such as No Parking, No Entry, Construction, and No U-Turn. The system can collect and recognize traffic sign information and issue instructions or warnings to the driver, or even directly control the vehicle to ensure smooth traffic flow and prevent accidents. For example, the TSR system can recognize speed limit signs and automatically adjust the vehicle's speed according to the speed limit to ensure that the vehicle does not exceed the speed limit.

2. Autonomous Driving Assistance

Traffic sign recognition is a key technology in autonomous driving. By identifying various road signs, the autonomous driving system can make appropriate driving decisions, such as slowing down, avoiding obstacles, and turning. This greatly improves the safety and reliability of autonomous driving.

Ⅲ. Traffic Sign Recognition Program

1. Learning Programming Blocks

Blocks	Usage
	The "recognition" type block: Click the small triangle on the block to select the card Traffic sign recognition mode. It can recognize up to 5 types of road signs. Returns "True" if a road sign is detected, otherwise "False".

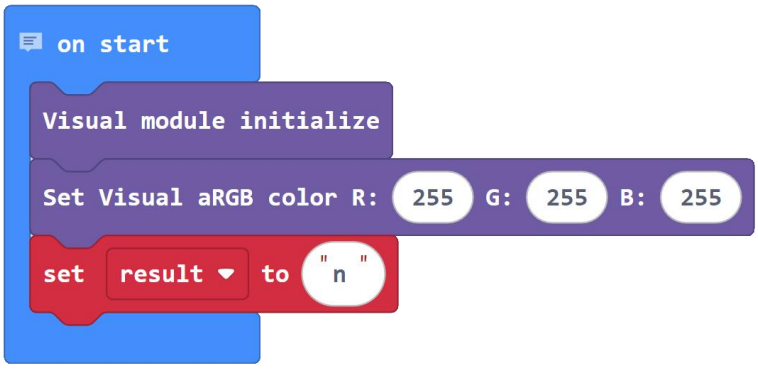
	The "get" type block: After entering the Traffic sign recognition function, use this block to return the road sign result.

2. Traffic Sign Recognition Program

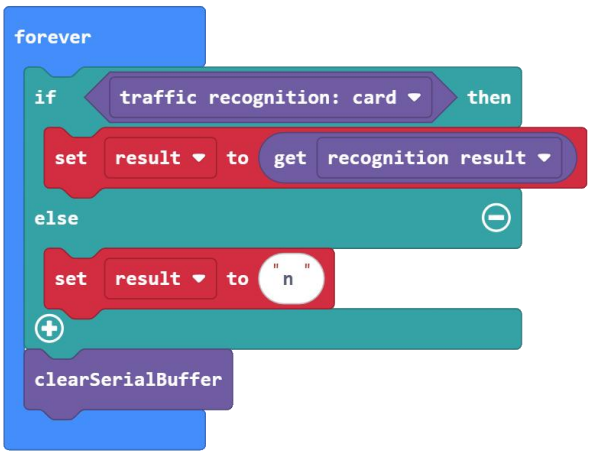
This program allows the module to recognize five road sign cards and perform corresponding actions upon detection.

To open the program, navigate to the "English\MakeCode car\4.Program" folder, locate the "5.1Traffic_sign.hex" file, and drag it into the MakeCode programming area.

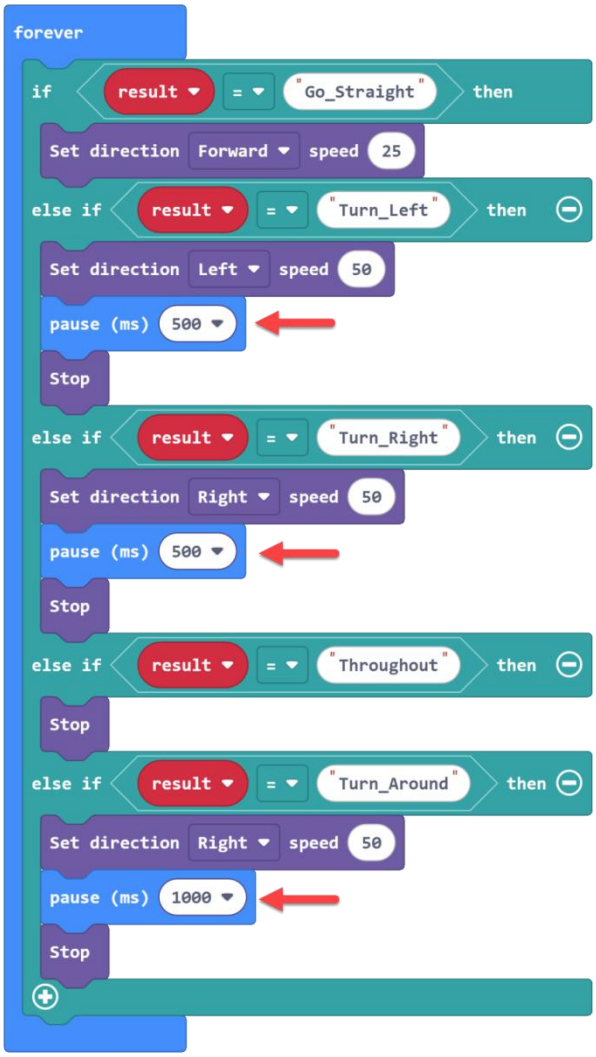
Upon opening, the program is divided into three parts:
The first part is the initialization program, which sets up modules and variables with initial configurations.



The second part is to obtain the card data and store it in a variable. You need to use the "clear serial buffer" block to clear the serial port data. Otherwise, the car will execute the related program twice after each card recognition.



The third part involves the car performing corresponding actions based on the recognition results. The delay of the red arrow controls the rotation time and can be adjusted according to actual conditions.



3. Program Results

Card	Effect
	The car continues moving until it detects another road sign.
	The car stops moving.
	The car turns right on the spot.
	The car turns left on the spot.
	The car makes a U-turn on the spot.

Ⅳ. Line Patrol and Traffic Sign Recognition

Now let's combine the car's line patrol function with the vision module to make an intelligent line patrol car. During patrol, it can turn left or right, limit speed, and honk the horn according to the wooden road signs.

1. Learning Programming Blocks and Road Sign Effects

Blocks	Usage
	The "recognition" block is specifically designed for recognizing wooden road signs. It can identify up to 4 types of road signs. Returns "True" if a road sign is detected, otherwise "False".

The effect after the car recognizes the road sign is as follows:

Road Sign	Effect
	After the vision module detects the right turn sign, the car will turn right at the fork in the road.
	After the vision module detects a left-turn sign, the car will turn left at the fork in the road ahead.
	Honk the horn. Upon recognizing a sign, the car will stop and play a buzzer sound. After the buzzer sound is complete, it will continue patrolling the line.

	Honk the horn. Upon recognizing a sign, the car's forward speed will slow down until it reaches the next intersection, at which point it will return to normal.

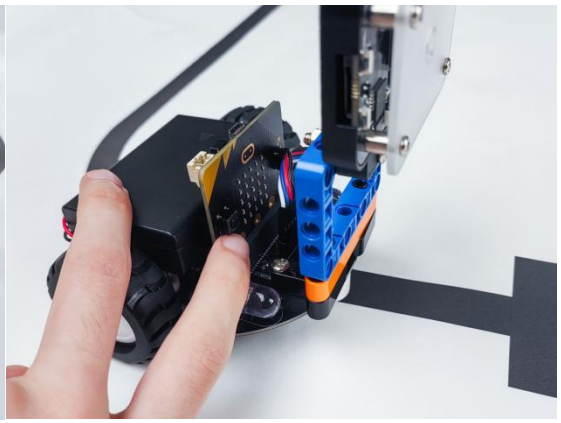
2. Line Patrol and Traffic Sign Recognition Program

To open the program, navigate to the "English\MakeCode car\4.Program" folder, locate the "5.2Traffic_sign_line.hex" file, and drag it into the MakeCode programming area.

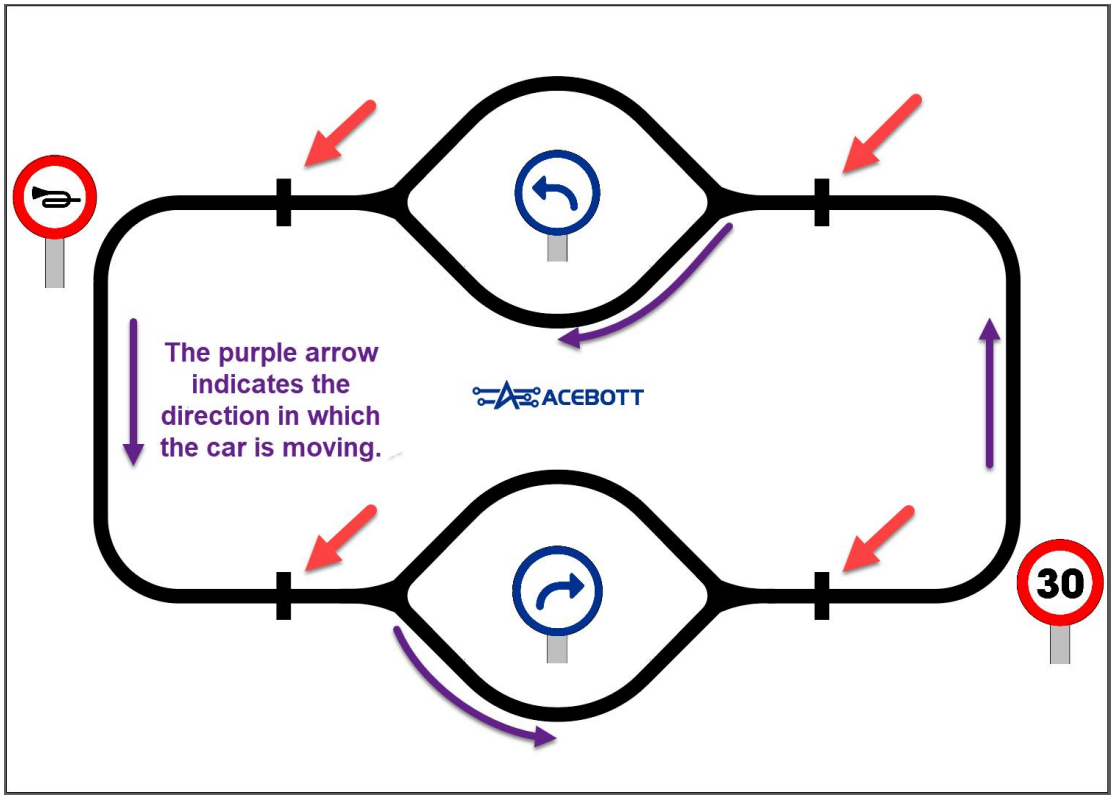
3. Line Patrol and Traffic Sign Recognition Program Functions

First, prepare the line-following map. The map file is located in "English\MakeCode car\3.MakeCode Car Map\Road Sign Patrol Line Map (100 x140cm)". You need to print it according to the specified dimensions.

After uploading the program and turning on the device, place the car on the line-following map. Wait until the AI vision module is fully initialized, then press the A button on the Micro:bit board to start.

The car will begin patrolling the line and recognize signs at intersections (indicated by red arrows in the image). Recognition takes up to 2 seconds. Four signs can be placed on the map. Refer to the illustration for placement.

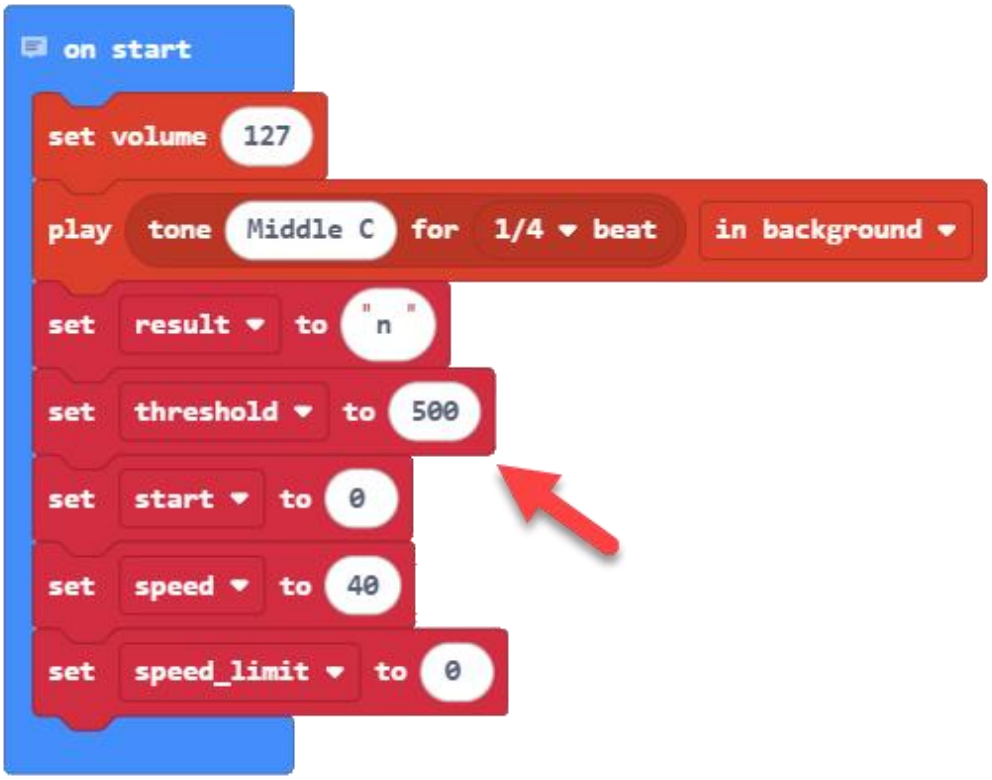


Note: You can use the included black tape to tape out a route for testing.

If the robot fails to move normally during line patrol, it may be due to the following reasons:

1. The robot's battery is dead.

2. Due to interference from ambient light, the sensor at the bottom of the car may mistakenly recognize white areas as black lines. To solve this problem, you can first test the specific values of the line patrol sensor on the white area and the black line, and adjust the value of the threshold variable in the program accordingly to make it closer to the actual measured black line value. The default value is 500.



3. Inaccurate traffic sign recognition may be caused by excessive lighting or dimming. Normal indoor white light is generally sufficient for detection. Alternatively, the road sign may be too far from the module, requiring adjustment of the distance or position. Alternatively, the road sign may not be placed directly in the center of the road, requiring adjustment of its position.

Lesson 6 Visual Line Patrol

Ⅰ. Principles of Visual Line Patrol

Visual line patrol is a technology in which a robot uses a camera to identify lines on the ground and follow them. Its core is to enable the robot to identify the position and direction of lines and make the correct movement decisions.

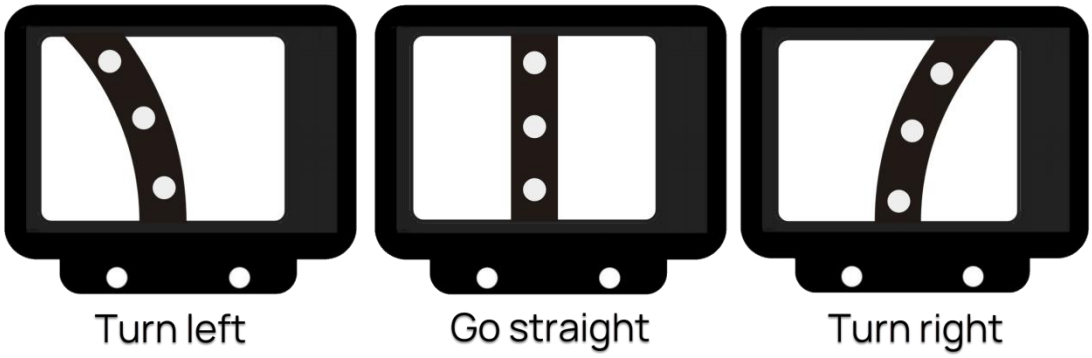
Simply put, machine vision line patrol generally involves four steps:

1. Image Acquisition: The camera acts like the robot's eyes, continuously capturing images of the ground ahead.

2. Color Recognition: The computer analyzes the image to identify lines of a specific color (such as black).

3. Position Determination: Calculate the position of the line in the image and determine its direction by marking the line (such as the dots in the image below).

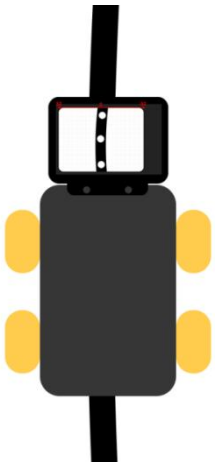
4. Direction Adjustment: Control the wheel steering based on the line's position.



During visual line patrol, when a black line is detected on the screen, the program draws positioning points at the top, center, and bottom of the line. The weights of these three positioning points

decrease from top to bottom. When these three positioning points complete the visual line patrol algorithm, they return a numerical parameter on the screen indicating the current position of the patrol line relative to the robot, ranging from approximately -52 to 52.

For the specific relationship between the numerical range and robot motion, see the table:

State Diagram	State Description	Execution State
	When the values returned by the three patrol points on the screen are between -5 and 5, the robot is located in the center of the patrol line.	The car moves forward.
	When the three line patrol points on the screen return values between -52 and -5, the robot's posture is offset to the right relative to the line patrol direction.	The car rotates right.

	When the three line patrol points on the screen return values between 5 and 52, the robot's posture is offset to the left relative to the line patrol direction.	The car rotates left.

Ⅱ. Visual Line Patrol Algorithm

The visual line patrol algorithm captures track images using a camera and uses image processing techniques to identify track lines, thereby controlling the robot to follow a predetermined route. This algorithm primarily includes the following steps:

1. Image Acquisition and Preprocessing: First, the track image is captured using a camera. The color image is then converted to grayscale for subsequent processing. Grayscale conversion is typically performed using a weighted averaging method.

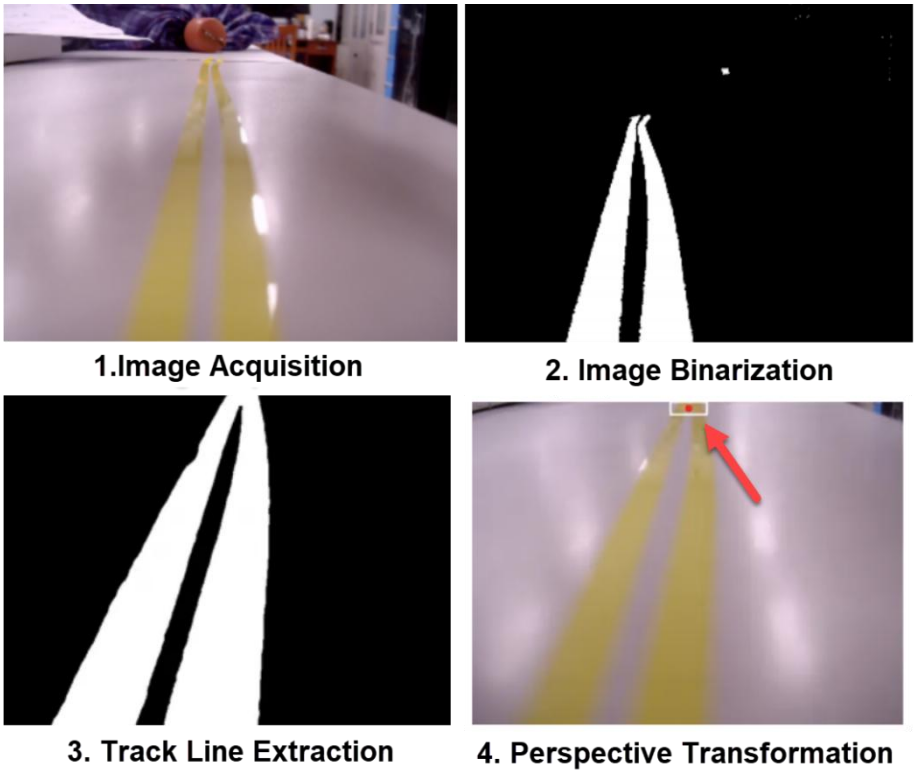
2. Image Binarization: Convert the grayscale image to a binary image by setting a threshold. Pixel values greater than the threshold are set to white, and pixels less than the threshold are set to black.

3. Track Line Extraction: Using methods such as the eight-neighborhood method or the maze method, extract the left and right edges of the track from the binary image. Then, find the longest white column and calculate its average value as the centerline.

4. Perspective Transformation: Because camera images may be distorted, a perspective transformation is required to transform the

area in front of the vehicle into a top-down view for more accurate track line identification.

5. Vehicle Control: Based on the recognition results, the vehicle's motor speed is adjusted and the robot's steering is controlled.



Application scenarios for the visual line-following algorithm primarily include autonomous vehicles and robotics competitions. Through precise image processing and control systems, the visual line-following algorithm enables robots to accurately follow a predetermined route, possessing high practical value and research significance.

Ⅲ. AI Car Visual Line Patrol Program

Now let's use the vision module's line patrol function to patrol the line.

1. Learning Programming Blocks

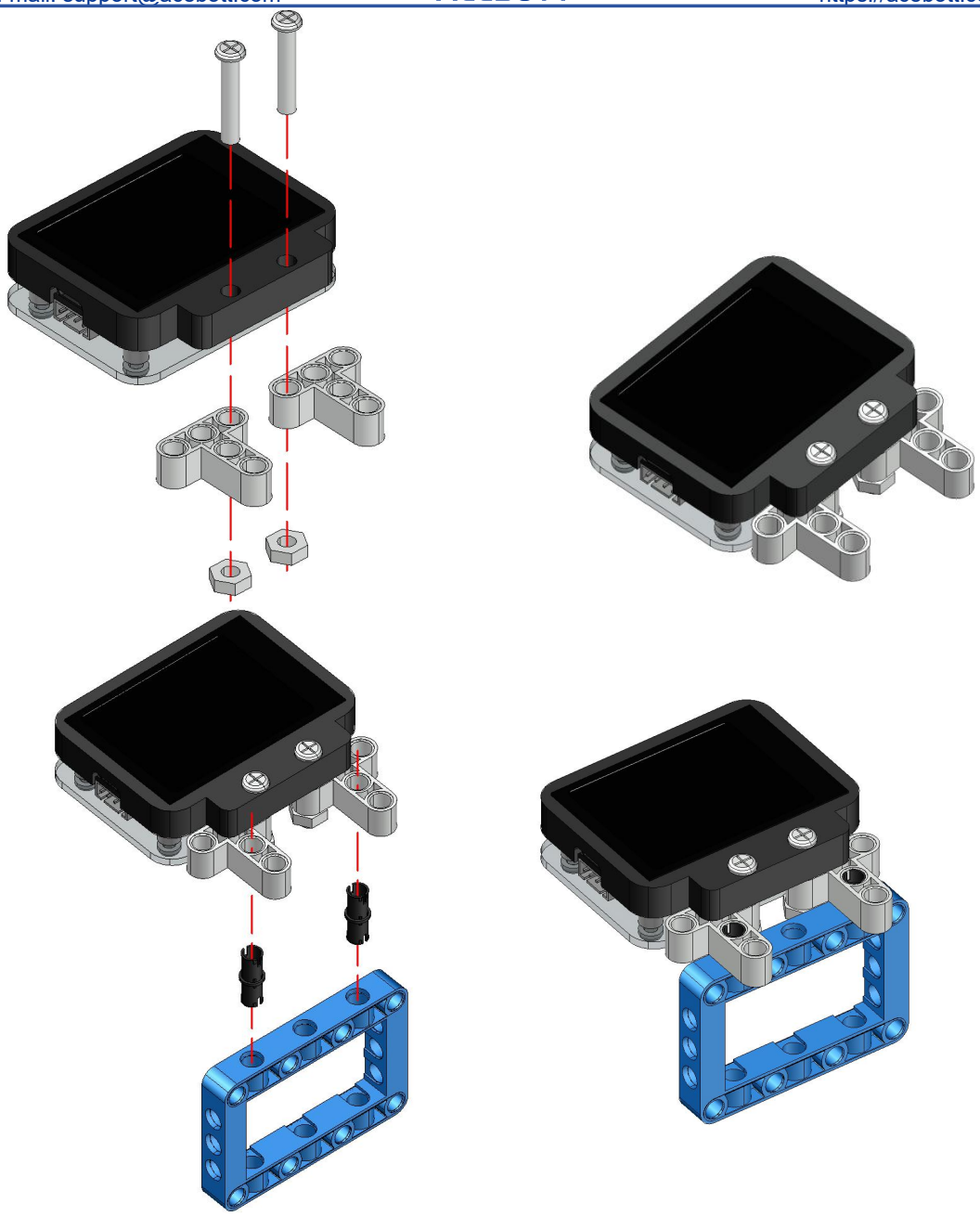
Blocks	Usage
	The "recognition" block is used for visual line patrol recognition. It returns "True" when it recognizes a line.
	The "get" block is used to get the line patrol value, which ranges from approximately -52 to 52. If no line is detected, it returns 53. The data type is string.

2. Visual Line Patrol Model Construction

Since the car's camera can't see the route, you need to use the building blocks in the set to modify the car structure so that the module's camera faces directly downward to detect the route. The construction steps are shown in the figure:

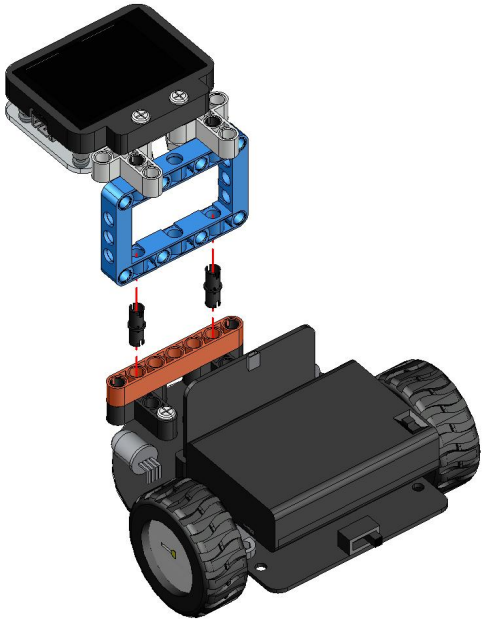
1. Install the AI vision module. Tighten the screws:

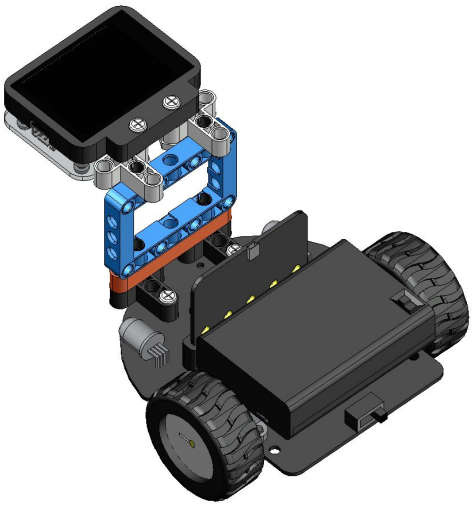
Name	Quantity
T-Frame Beam	1
5x7 frame beam	1
Friction Pin	2
AI Module	1
M4*22mm Round Head Screws	2
M4 Nut	2



2. Securing the Vision Module:

Name	Quantity
Friction Pin	2





3. Line Patrol and Street Sign Recognition Program

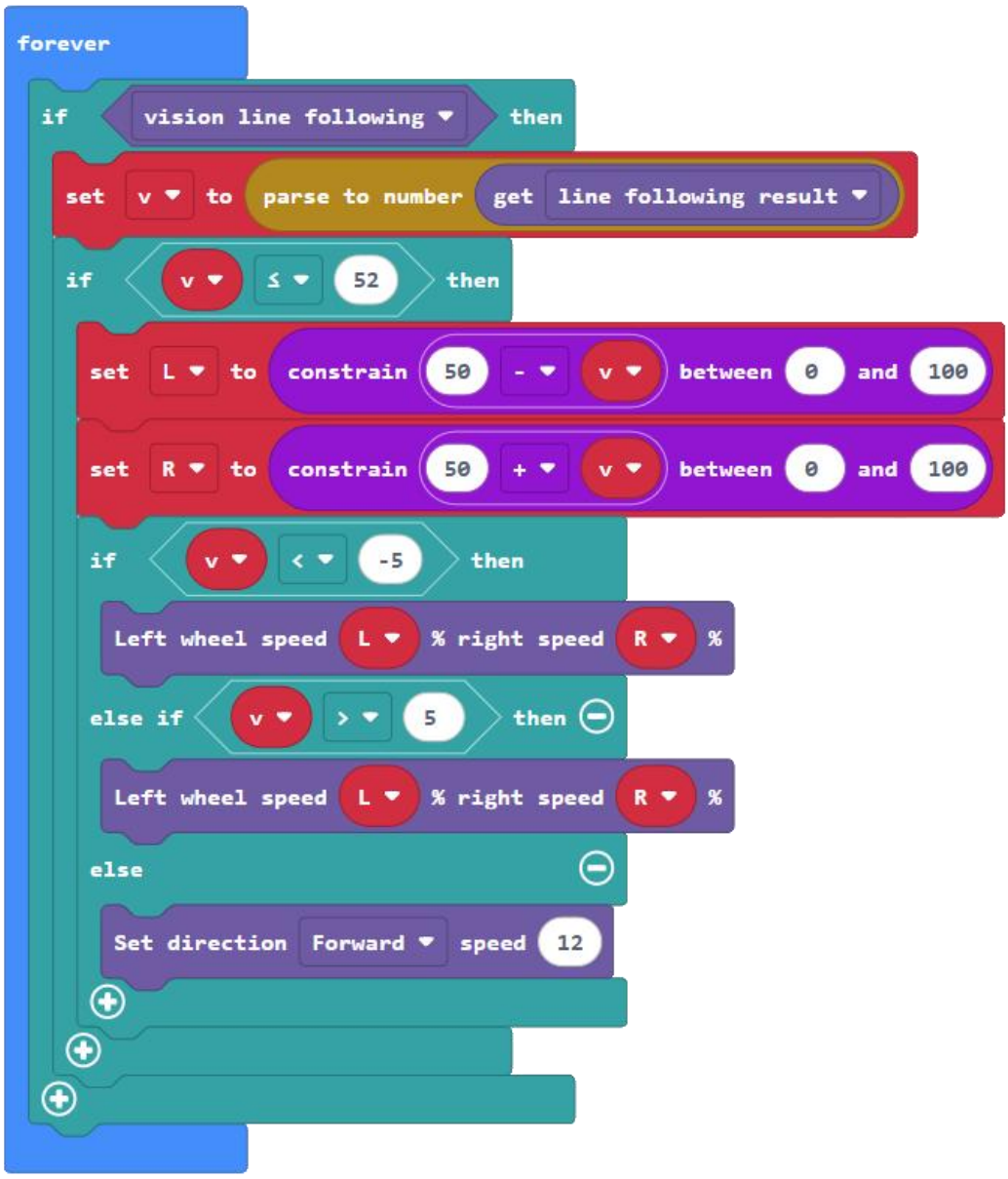
To open the program, navigate to the "English\MakeCode car\4.Program" folder, locate the "6.Visual_patrol.hex" file, and drag it into the MakeCode programming area.

This program has only two parts:

The first part is initialization. The module will only be initialized after pressing the onboard A button on the Micro:bit.

The second part is the visual line patrol. When a route is detected, the returned value is converted from a string to a number and

stored in the variable "v." The car then controls forward or backward based on the variable's value. If the variable is less than -5, the car turns left; if it is greater than 5, it turns right; otherwise, the car moves forward.



Then prepare the line-following map. The map file is located in "English\MakeCode car\3.MakeCode Car Map\Visual Patrol Line Map (80x120cm)". Print it according to the specified dimensions.

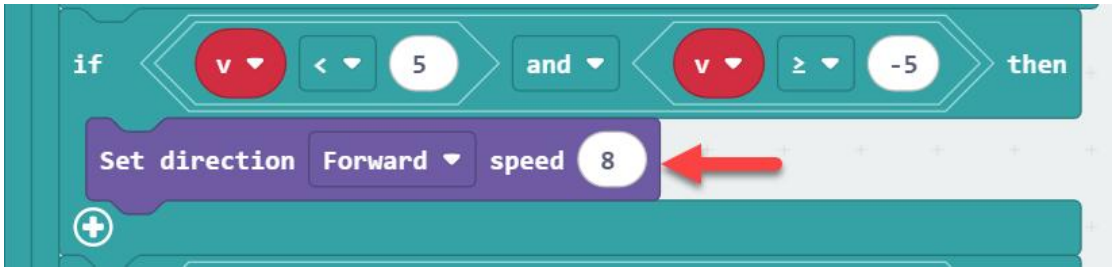
After completing the setup, upload the program. Place the car in the center of the line-following path. Turn on the device and wait until the module can capture images, then press the onboard A button on the Micro:bit to start line tracking.

Note: If you can't patrol the line, or can't turn in time at a bend on the map, it may be due to the following reasons:

1. The site is too bright or too dark. Too bright will cause reflections on the patrol map, causing the vision module's key patrol points to disappear. Try turning on the fill light to improve this. Too dark will prevent the black line from being detected.

2. Low battery power can cause the robot to move slowly, the vision module to malfunction, repeated restarts, and screen freezes.

3. Because the sensitivity of visual line patrol is much lower than that of grayscale sensor line patrol, the speed will be reduced. The recommended speed range is 8-15.



Lesson 7 Machine Learning

Ⅰ. Machine Learning

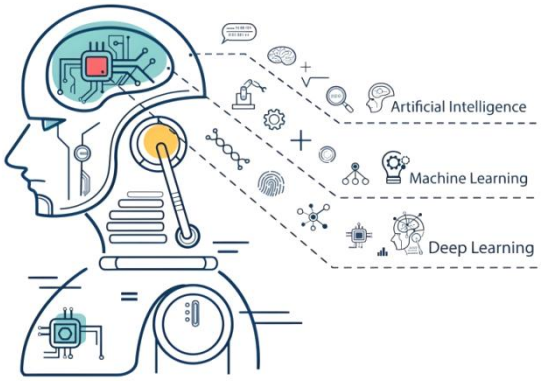
Machine learning is a branch of artificial intelligence that aims to enable computers to acquire knowledge and experience from data through learning and automated reasoning, and use this knowledge and experience to perform pattern recognition, predictions, and decision-making.

The core idea of machine learning is to use data to train computer algorithms, enabling them to automatically learn from data and improve their performance without explicit programming. By analyzing and interpreting large amounts of input data, machine learning algorithms can identify patterns and trends in the data and generate predictive models that can be applied to new data.

For example, after seeing numerous pictures of cats and dogs, a computer can automatically identify characteristics such as "cats have pointy ears and dogs have long noses," allowing it to automatically recognize new images the next time it encounters them. This type of learning doesn't require step-by-step human programming; instead, it's "trained" through data.

Machine learning has widespread applications in various fields, such as image recognition, speech recognition, natural language processing, recommendation systems, and data prediction. It has become a key area of development in modern science and

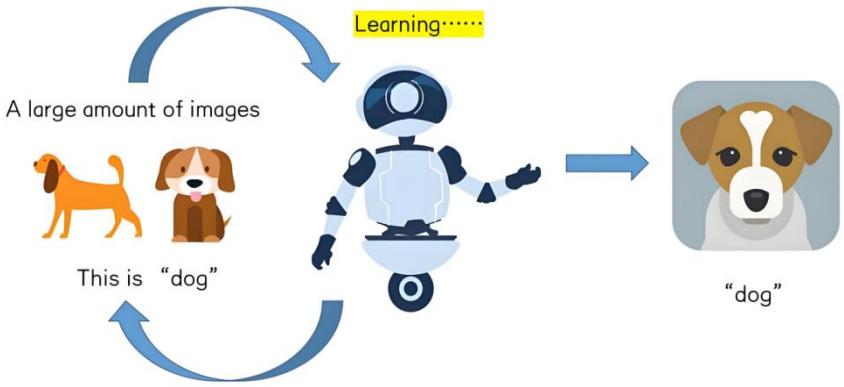
technology and artificial intelligence, providing effective methods and tools for solving many practical problems.



Ⅱ. Image Classification

Image classification is an image processing method that distinguishes objects of different categories based on the distinct features reflected in the image information. Image classification plays a key role in many applications, such as face recognition, object recognition, and autonomous driving.

The basic principle of image classification is to extract image features and compare these features with a pre-trained model to determine the image's category. Common feature extraction methods include traditional hand-crafted features and deep learning methods. Traditional hand-crafted features typically include color, texture, and shape features, but these methods often perform poorly when processing complex images. Deep neural networks constructed through deep learning methods, on the other hand, can automatically learn more discriminative features from images.



Ⅲ. Built-In Image Classification Application

The image classification function of the vision module relies entirely on machine learning. The module supports two types of image classification: one is built-in image classification, which uses pre-trained images for direct recognition; the other involves training new custom images. This tutorial will first explain the usage of the first type.

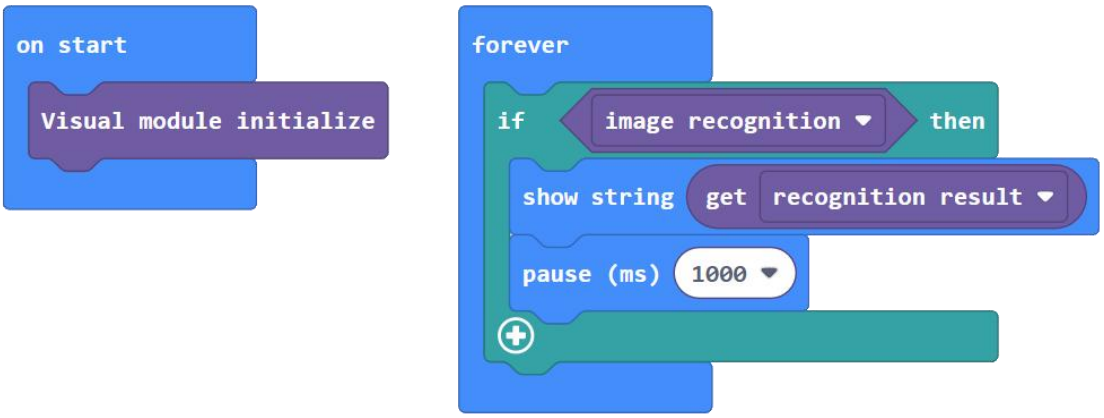
1. Learning Programming Blocks

Blocks	Usage
	The "recognition" block is used to enable the built-in image recognition function. It can recognize objects from four types of cards. Returns "True" if the corresponding card is detected, otherwise returns "False".

2. Machine Learning Program

To open the program, navigate to the "English\MakeCode car\4.Program" folder, locate the "7.1Image_recognition.hex" file, and drag it into the MakeCode programming area.

This program is very simple. After the device is turned on, it initializes the modules and then enables the image recognition function. During use, if the visual module is not switched to the image recognition mode, you need to press the reset button of the Micro:bit.



3. Program Results

After running the program, you'll notice that the module mode changes to Object recognition. It can recognize images of four cards and return their corresponding names, such as "Bike."

Ⅳ. Machine Learning Image Classification

Now let's use machine learning image classification to classify specific images.

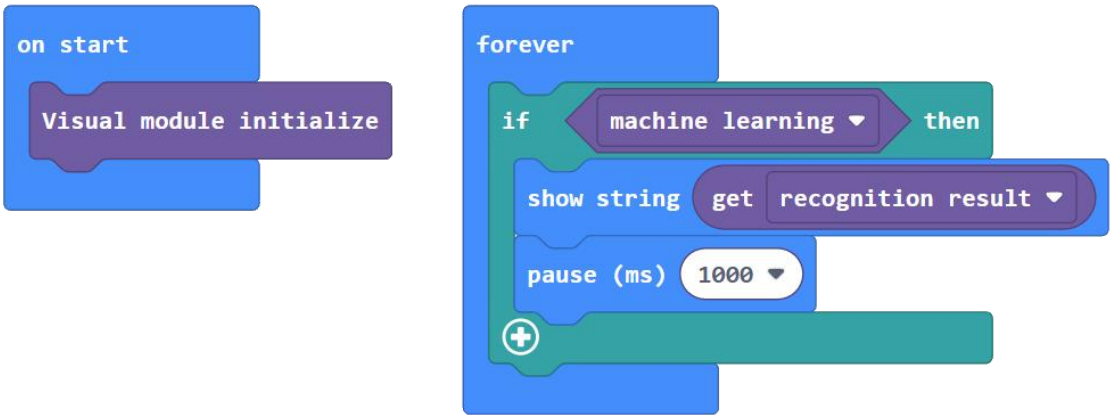
1. Learning Programming Blocks

Blocks	Usage
	The "recognition" block is used to enable the machine learning image classification function. Currently, up to 3 different image classifications can be made.

2. Machine Learning Program

To open the program, navigate to the "English\MakeCode car\4.Program" folder, locate the "7.2Machine_learning.hex" file, and drag it into the MakeCode programming area.

This program is also very simple. After the device is turned on, it performs module initialization and then automatically enables the machine learning function.



3. Program Results

After the program upload is complete, place the card or object you want the AI vision module to learn in front of its camera. Once the

card or object is clearly visible, press the S2 button to record the image from the first angle. Then, switch to the second angle and press the S2 button again to record the image from all four angles. The AI vision module will learn the information about the current card or object and automatically generate a label.

Then change to another card or object and follow the previous steps to train up to 3 different images. Finally, place the learned card or object in front of the AI vision module's camera. Its screen will display the corresponding label of the card or object.


Training...
Machine Learning

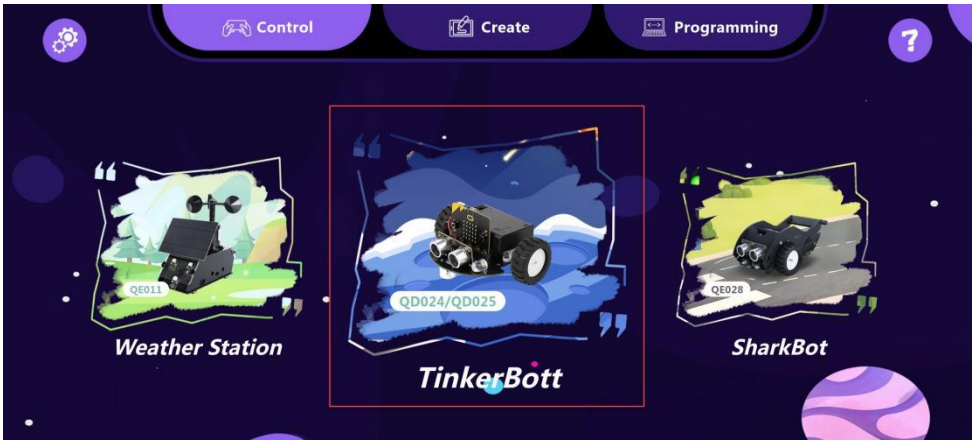

Class2
Machine Learning

Lesson 8 App Control

Ⅰ. App Download

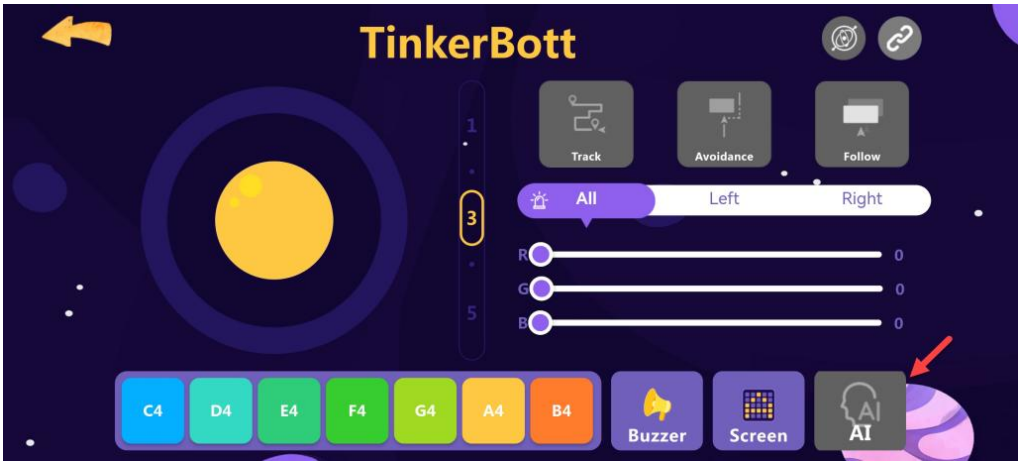
1. If you are using an iOS phone, you need to search for the keyword "ACEBOTT" in the App Store and then download it; if you are using an Android phone, you need to search for the keyword "ACEBOTT" in the Google Play Store and then download it; the icon is shown in the figure below.

2. After entering, swipe left or right to select TinkerBott.



3. Select Control to enter the control interface.

4. Enter the car control interface and see the AI option in the lower corner. If this option does not appear, please update to the latest version (it cannot be controlled directly yet and the program needs to be uploaded):

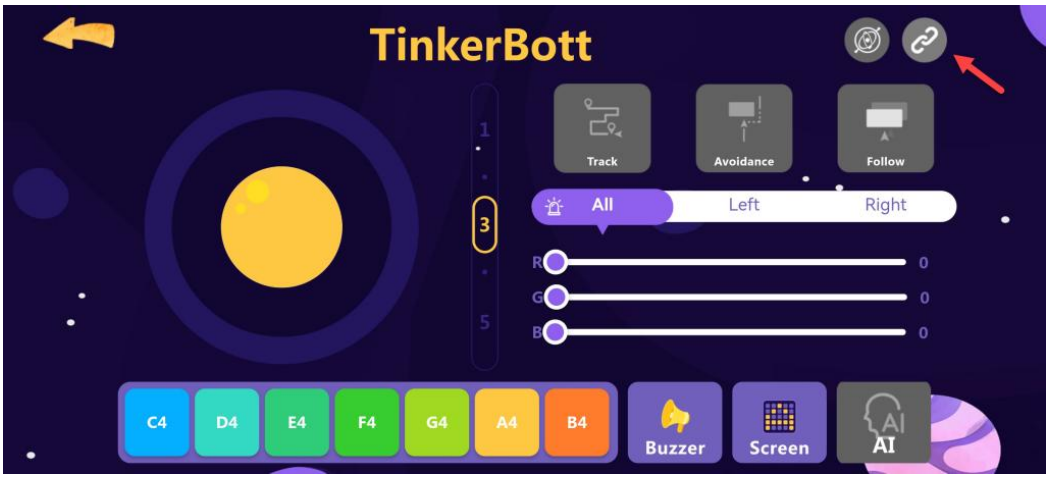


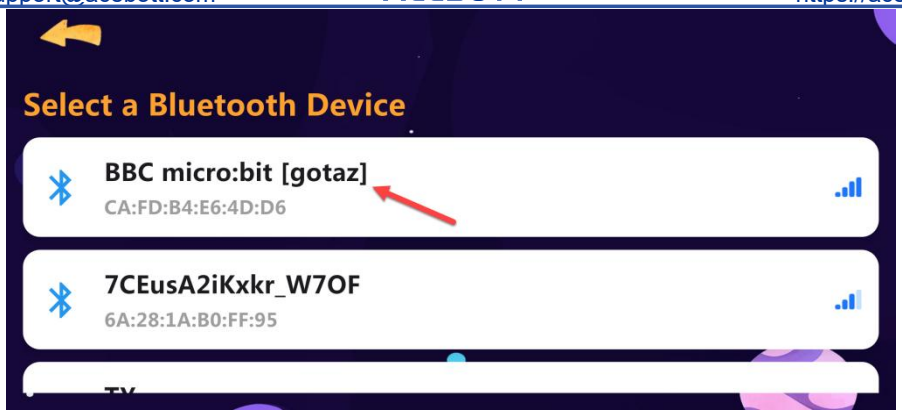
Ⅱ. Program Upload

To open the program, navigate to the "English\MakeCode car\4.Program" folder, locate the "8.App_Control.hex" file, upload it to the Micro:bit.

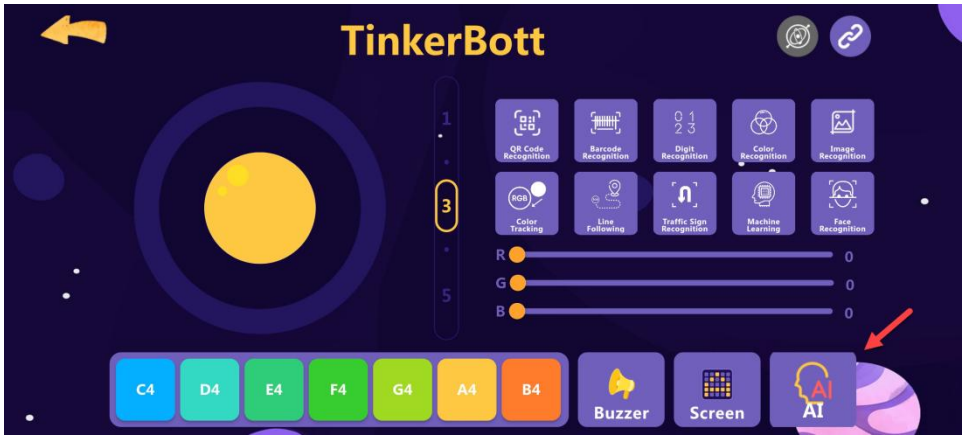
Ⅲ. App Control Instructions

After uploading, run the program, turn on your phone's Bluetooth, and click the icon to connect to Bluetooth.





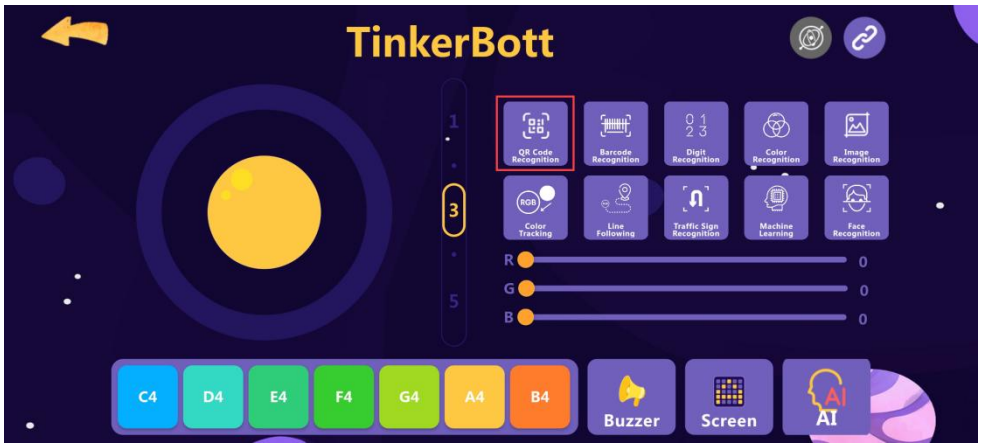
After a successful connection, click the AI function to enable it successfully.



The following is an introduction to the 10 AI functions of the APP:

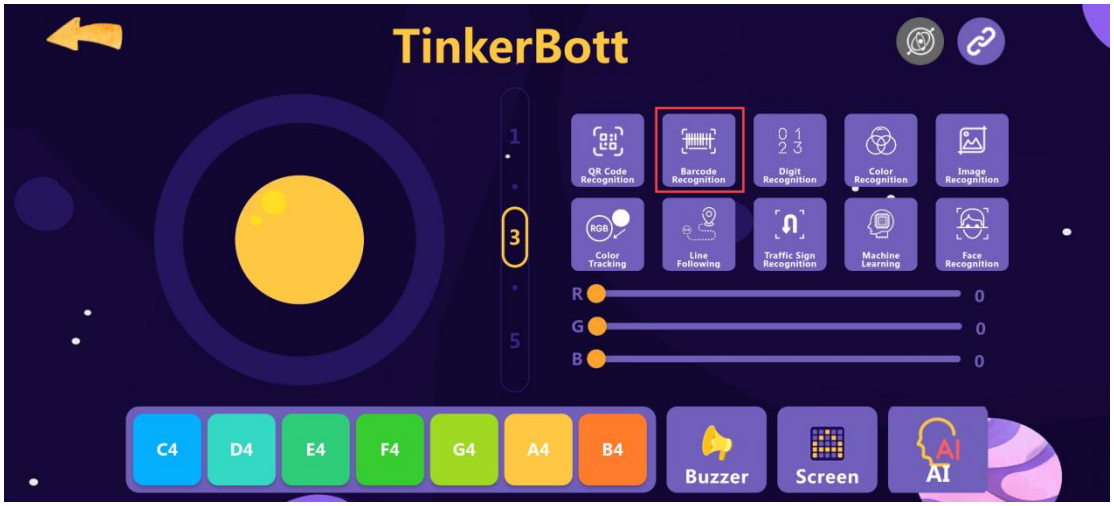
1. QR Code Recognition:

Click the "QR Code Recognition" button. Once enabled, the Vision module switches to the QR code recognition function and returns the text content of the QR code.



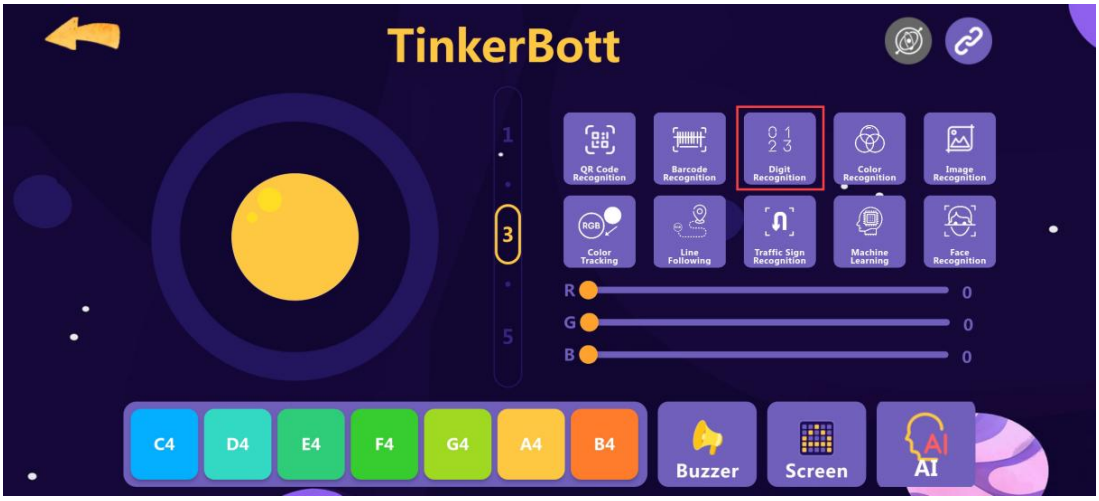
2. Barcode Recognition

After clicking the "Barcode Recognition" button, place a card with a barcode in front of the AI vision module's camera. The screen will recognize the barcode information.



3. Digit Recognition

After clicking the "Digit Recognition" button, place a card with a number in front of the AI vision module's camera. The screen will recognize the number on the card.

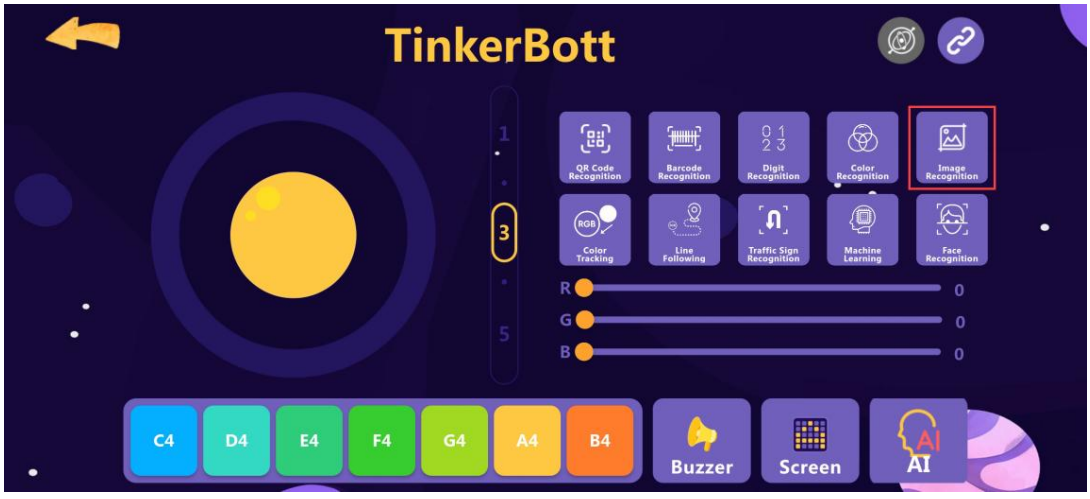


4. Color Recognition

After clicking the "Color Recognition" button, place the colored card in front of the AI vision module's camera. The screen will recognize the color information on the card (red, green, and blue).

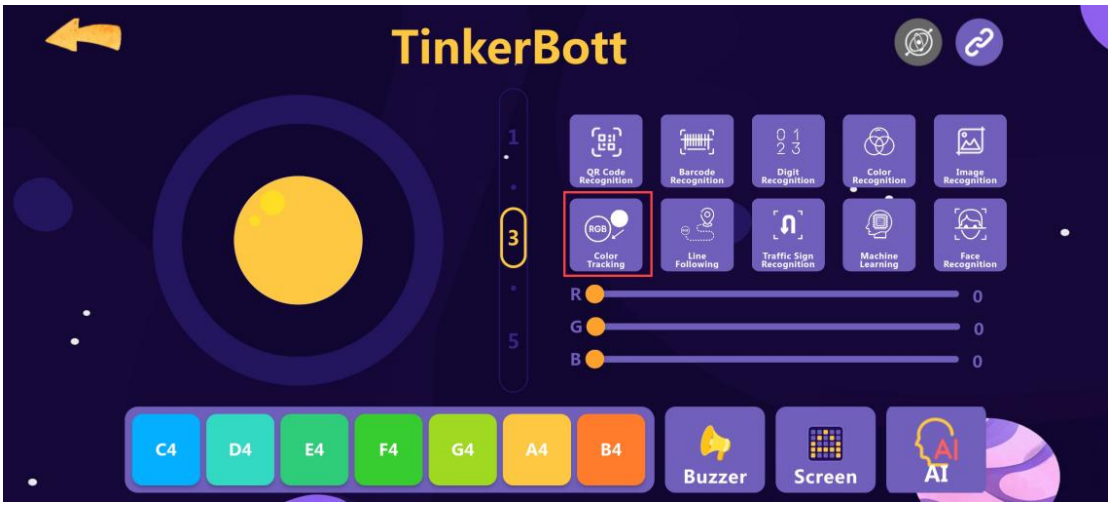
5. Image Recognition

After clicking the "Image Recognition" button, place cards with different object images in front of the AI vision module's camera. The screen will recognize the image information from the card.



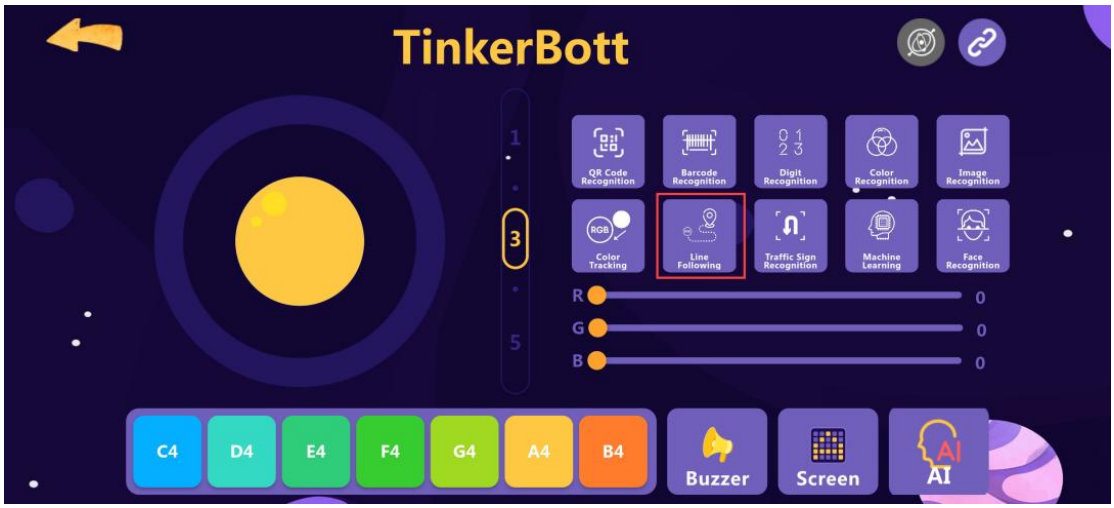
6. Color Tracking

After clicking the "Color Tacking" button, select a color. Place the card with the selected color in front of the AI vision module's camera. The screen will recognize the color information on the card, and the smart car will move according to the color of the card.



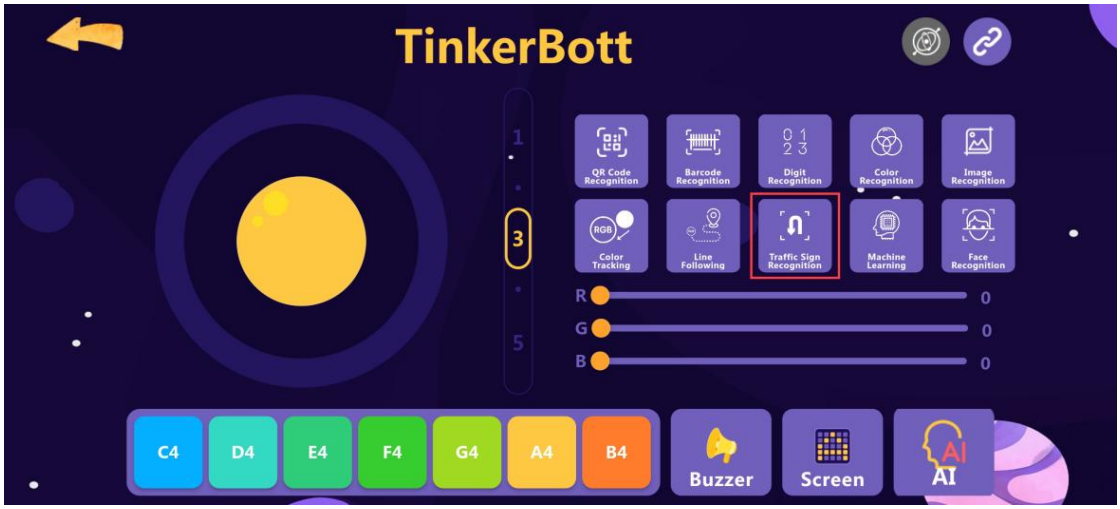
7. Visual Line Patrol

After clicking the "Line Following" button, place the car on the patrol map. The car will move along the black line. Note that the car's structure needs to be adjusted. Refer to Lesson 6 for the construction steps.



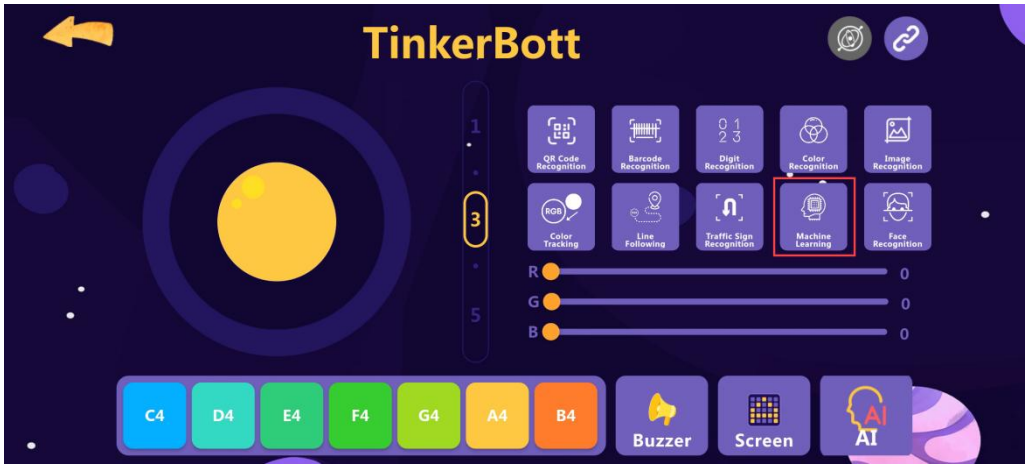
8. Traffic Sign Recognition

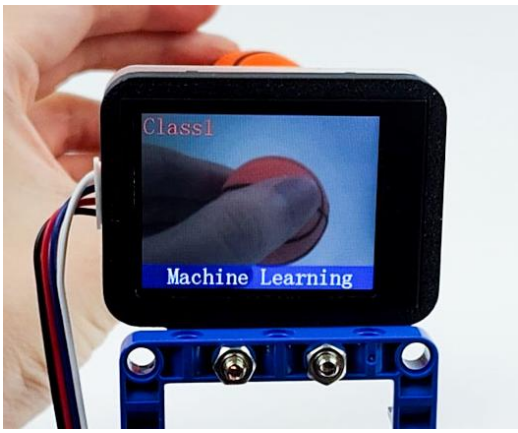
After clicking the "Traffic Sign Recognition" button, place a card with a sign in front of the AI vision module's camera. The screen will recognize the road sign information on the card, and the smart car will follow the movement of the card's road sign.

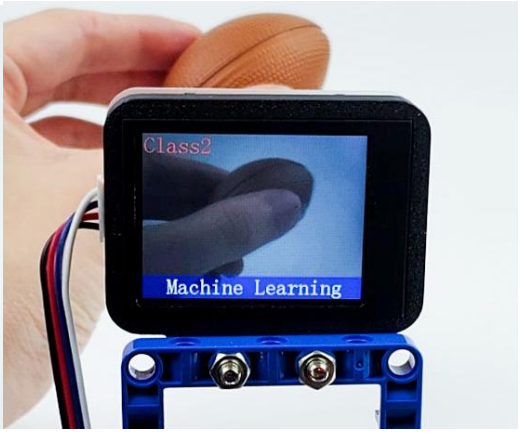


9. Machine Learning

After clicking the "Machine Learning" button, place cards with images of different objects in front of the camera of the AI vision module. The AI vision module will learn the information in the card. When the recognition is completed, when the recognized card is placed in front of the camera again, the AI vision module will display its information.

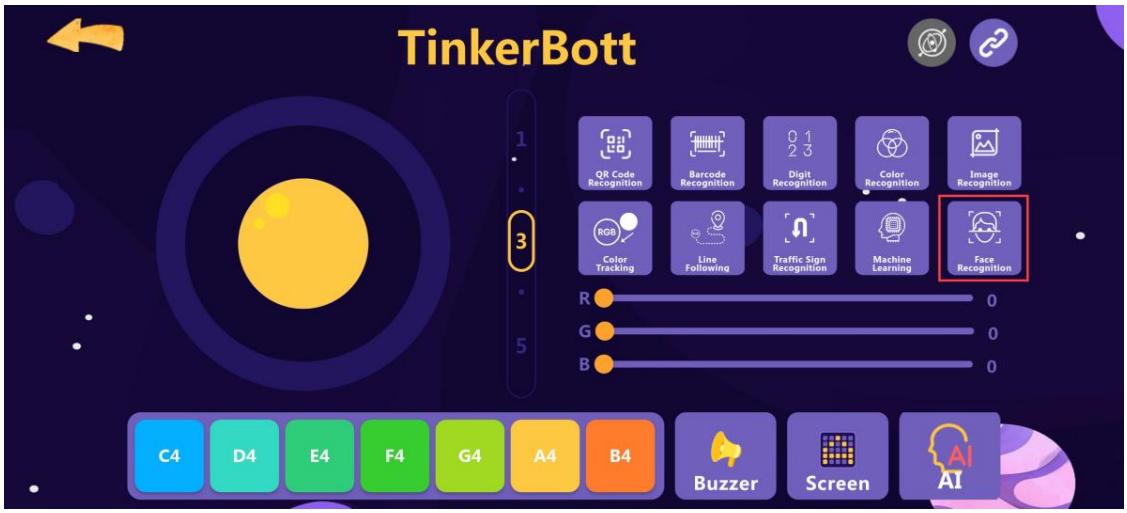






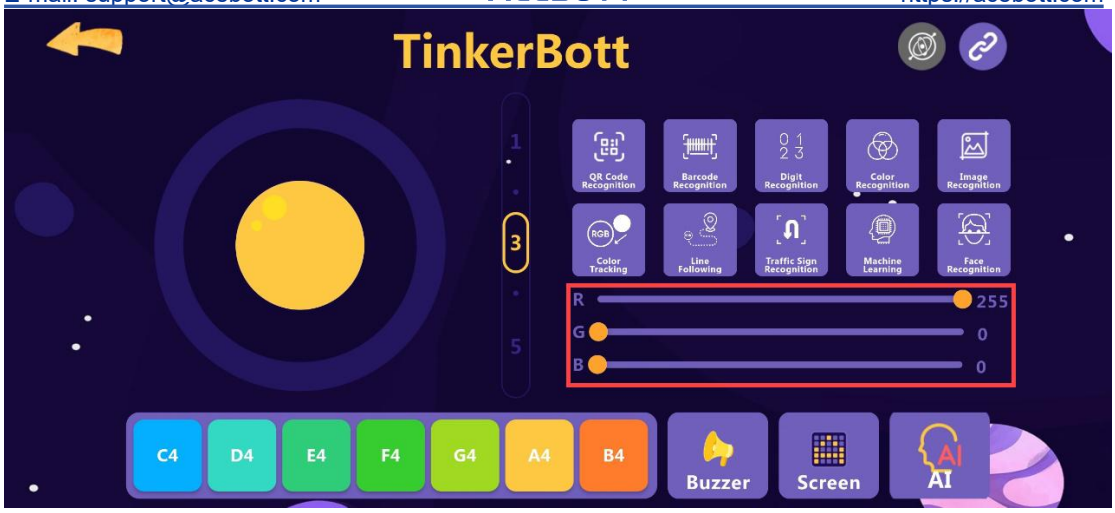
10. Face Recognition

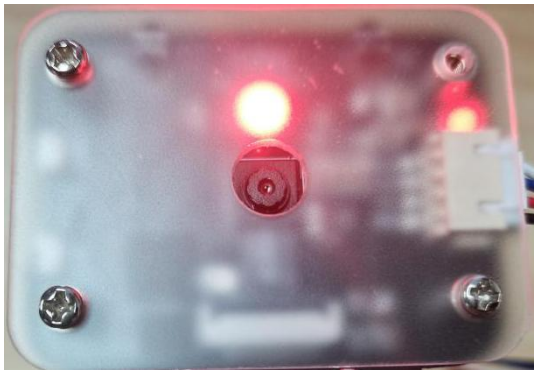
After clicking the "Face Recognition" button, place cards with different faces in front of the AI vision module's camera. The AI vision module will learn the card information. After recognition is complete, if you place the recognized card in front of the camera again, the AI vision module will display its information.



11.RGB Light

By sliding the R, G, and B sliders, you can control the color change of the RGB light on the AI vision module. The value range is: 0~255.





Note: This app program can only use the AI function and mobile control function. Due to the limited storage capacity of Micro: bit, more programs cannot be written, so the buzzer and screen programs cannot be used.

Follow Us

Scan the QR codes to Follow Us for troubleshooting & the latest news.

We have a very large community that is very helpful for troubleshooting and we also have a support team at the ready to answer any questions.

ACEBOTT FB QR Code

YouTube QR Code