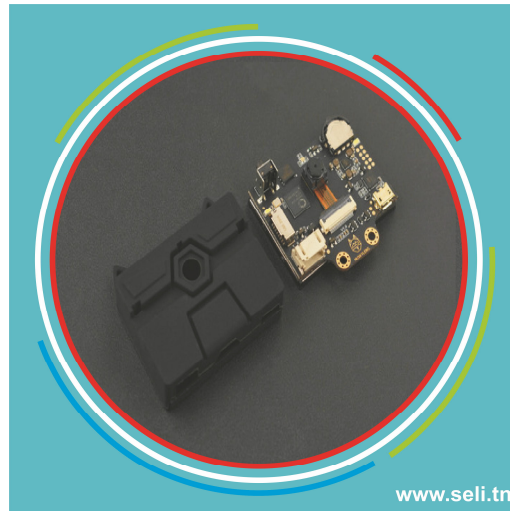


HUSKYLENS+C

MODULE INTELLIGENCE ARTIFICIEL HUSKYLENS COMPATIBLE AVEC ARDUINO MICROBIT RASPBERRY... + PROTECTEUR EN SILICONE

SKU:SEN0305



1. Introduction

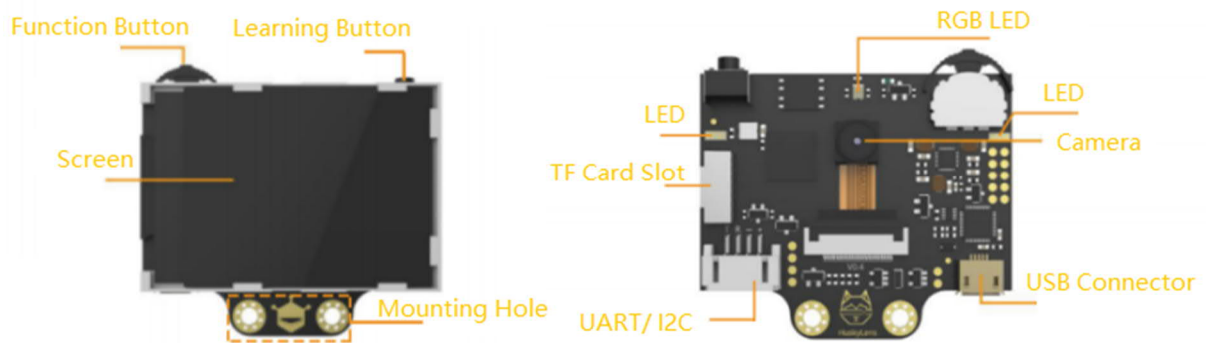
HuskyLens is an easy-to-use AI machine vision sensor with 7 built-in functions: face recognition, object tracking, object recognition, line tracking, color recognition, tag recognition and object classification.

Through the UART / I2C port, HuskyLens can connect to Arduino and micro:bit to help you make very creative projects without playing with complex algorithms.

2. Specification

- Processor: Kendryte K210
- Image Sensor:
 - SEN0305 HuskyLens: OV2640 (2.0Megapixel Camera)
 - SEN0336 HuskyLens PRO: OV5640 (5.0MegaPixel Camera)
- Supply Voltage: 3.3~5.0V
- Current Consumption(TYP): [320mA@3.3V](#) , [230mA@5.0V](#) (face recognition mode; 80% backlight brightness; fill light off)
- Communication Port: UART; I2C
- Display: 2.0-inch IPS screen with 320*240 resolution
- Built-in Algorithms: Face Recognition, Object Tracking, Object Recognition, Line Tracking, Color Recognition, Tag Recognition, Object Classification
- Dimension: 52mm x 44.5mm (2.05*1.75 inch)

3. Board Overview



3.1 Connectors

- USB Connector: power supply for HuskyLens; connect to the computer to upgrade the firmware
- 4pin Connector in UART Mode

Num	Label	Pin Function	Description
1	T	TX	TX pin of HuskyLens
2	R	RX	RX pin of HuskyLens
3	-	GND	Negative (0V)
4	+	VCC	Positive (3.3~5.0V)

- 4pin Connector in I2C Mode

Num	Label	Pin Function	Description
1	T	SDA	Serial clock line
2	R	SCL	Serial data line
3	-	GND	Negative (0V)
4	+	VCC	Positive (3.3~5.0V)

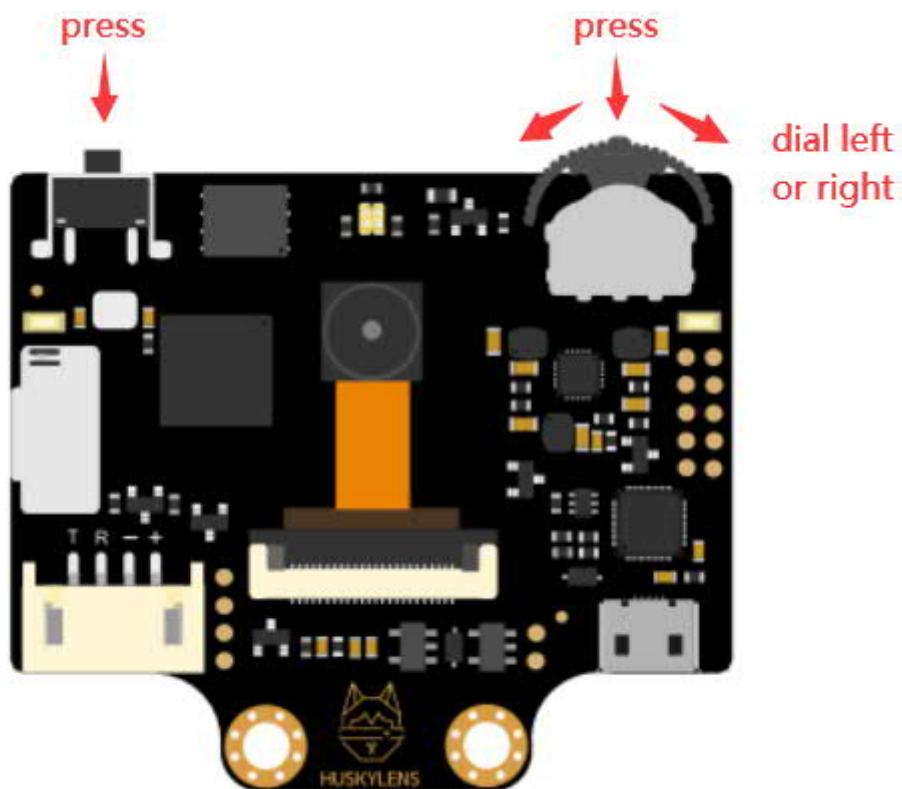
Tips:

- You can connect either USB port or 4pin port to power HuskyLens.
- Due to the onboard power supply automatic switching circuit, the USB port and the 4pin port can be connected to the power supply at the same time, and the power supply on the USB port is used first.
- Please ensure that the supply voltage and rate are sufficient to prevent HuskyLens from working abnormally.

3.2 Buttons

There are two buttons on the HuskyLens, the function button and the learning button. The basic operations of these two buttons are shown as follows:

- Dial the "function button" to left or right to switch different functions.
- Short press the "Learning button" to learn the specified object; long press the "Learning button" to continuously learn the specified object from different angles and distances; if HuskyLens has learned the object before, short press the "Learn button" to make it forget.
- Long press the "function button" to enter into the second-level menu(parameter setting) in the current function. Dial left, right or short press the "function button" to set related parameters.



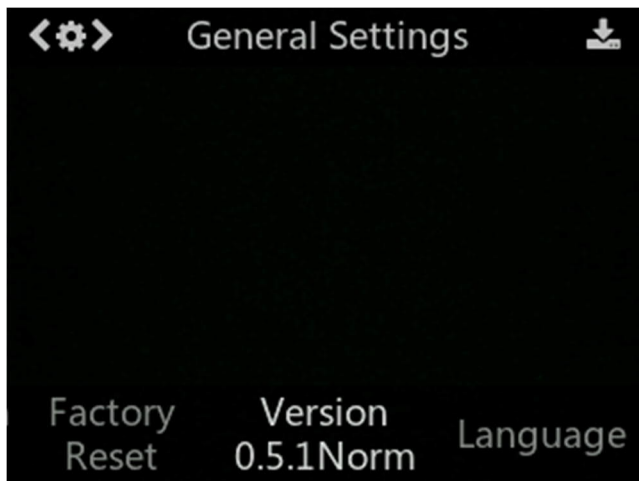
4. Upgrade Firmware

Before using this product, it is strongly recommended to upgrade HuskyLens' firmware to the latest version to get the latest functions and the most stable experience. If your HuskyLens already has the latest firmware, no update is required. We recommend to upload the firmware on windows 10 using the HuskyLens Uploader since it features a GUI, and easy-to-use.

The firmware version required for this tutorial is not lower than V0.5.1a. This version integrates all functions (including object classification).

How to check the firmware version?

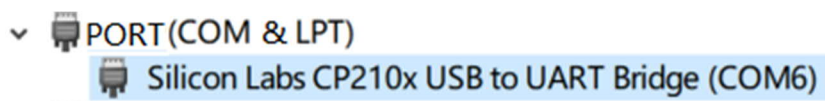
1. Dial the function button to the right until the last option "General Settings", short press the function button to enter the secondary menu.
2. Dial the function button to the right until you reach the "version" option, you can see the version number. As shown in the figure below, the version number is: 0.5.1Norm.



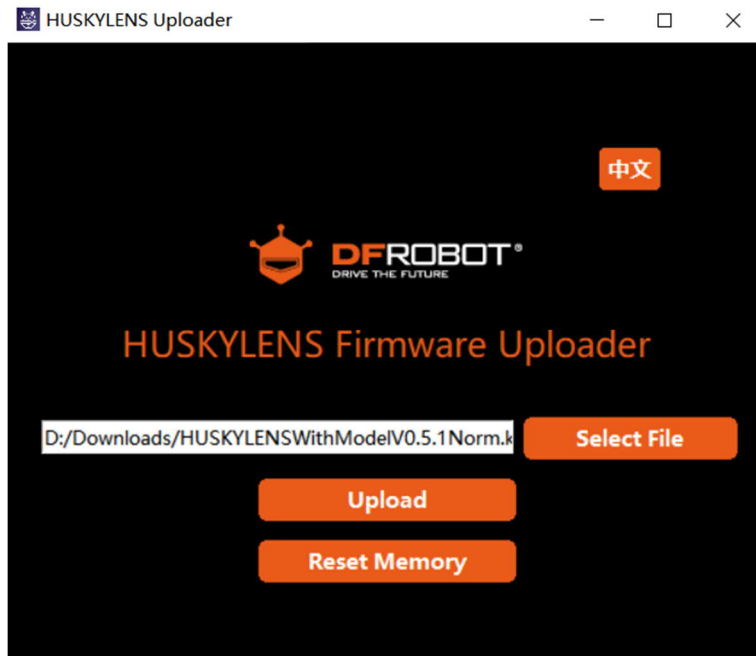
4.1 In Windows

We recommend to upload the firmware on windows 10 using the HuskyLens Uploader. It is easy and convenient. The steps are shown below:

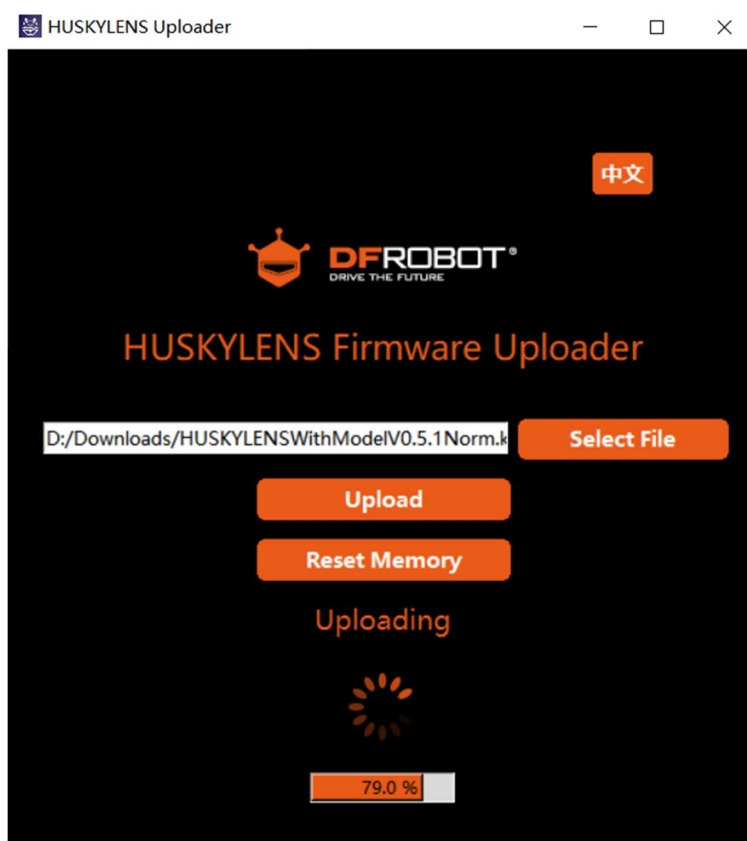
1. Download the HuskyLens Uploader. [Click here](#) to download it.
2. Download the USB to UART driver, and install it. [Click here](#) to download it.
HuskyLens adopts the CP2102N chip to implement the USB to serial port function. After the driver is installed, you can use the microUSB cable to connect the USB port of HUSKYLENS to the computer's USB port. At this time, open the device manager and there should be a COM port whose name starts with "Silicon Labs CP210x". As shown below:



3. Download the latest firmware. [Click here](#) to download the V0.5.1a. [Click here](#) to check the firmwares of all versions.
4. Run the HuskyLens Uploader, a small black cmd window will pop up first, and after a while, the interface window will appear, then click the "Select File" button to load the firmware.



5. Click the "Upload" button. Wait about 5 minutes to complete the uploading. The firmware file is large, so it may take a little bit time. After the uploading is completed, the prompt text "Uploading" will disappear and the HuskyLens screen will light up. Please do not close the interface window and the small black cmd window during uploading.



Tips:

- If you are prompted to enter Huskylens' COM port Manually, you need to manually enter the HuskyLens' COM port mapped on your computer. The COM port number can be found in the device manager.
- If the firmware uploading fails or the HuskyLens' screen does not light up, or other strange phenomenas, such as invalid buttons, HuskyLens can not learn, etc. you can try to click the "Reset Memory" button. After a while, the two fill lights of HUSKYLENS will light up, and HuskyLens has been reset. Please refer to the above steps and re-upload the firmware.

4.2 In Linux or Mac

In this section, we take ubuntu 18.04.4 as an example to show you how to upgrade Huskylens firmware on Linux or Mac. These steps are shown as follows:

1. Download the USB to UART driver, and install it. [Click here](#) to download it.

HuskyLesn adopts the CP2102N chip to implement the USB to serial port function.

In Ubuntu 18.04.4, the USB serial port of HuskyLens can be directly identified when plugged in, so the driver is not required to be installed.

2. Download the latest firmware and kflash.py script. [Click here](#) to check them.

You can clone the entire repository of "HuskyLens / HUSKYLENSUploader" to your computer by git command.

3. Install `pip3` first if you do not have it in your OS.

```
sudo apt install python3-pip
```

```
user@ubuntu:~$ sudo apt install python3-pip
[sudo] password for user:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  build-essential dh-python dpkg-dev fakeroot g++ g++-7 gcc gcc-7
  libalgorithm-diff-perl libalgorithm-diff-xs-perl libalgorithm-merge-perl
  libasan4 libatomic1 libc-dev-bin libc6-dev libcilkrts5 libexpat1-dev
  libfakeroot libgcc-7-dev libitm1 liblsan0 libmpx2 libpython3-dev
  libpython3.6-dev libquadmath0 libstdc++-7-dev libtsan0 libubsan0
  linux-libc-dev make manpages-dev python-pip-whl python3-dev
  python3-distutils python3-lib2to3 python3-setuptools python3-wheel
  python3.6-dev
Suggested packages:
  debian-keyring g++-multilib g++-7-multilib gcc-7-doc libstdc++6-7-dbg
  gcc-multilib autoconf automake libtool flex bison gcc-doc gcc-7-multilib
  gcc-7-locales libgcc1-dbg libgomp1-dbg libitm1-dbg libatomic1-dbg
```

Install `pip3` on MAC

```
/bin/bash -c "$(curl -fsSL
https://raw.githubusercontent.com/Homebrew/install/master/install.sh)"
brew install python3
```

4. Run the following script to install `pyserial`.

```
sudo pip3 install pyserial
```

```
user@ubuntu:~$ sudo pip3 install pyserial
The directory '/home/user/.cache/pip/http' or its parent directory is not owned
by the current user and the cache has been disabled. Please check the permission
s and owner of that directory. If executing pip with sudo, you may want sudo's -H
flag.
The directory '/home/user/.cache/pip' or its parent directory is not owned by th
e current user and caching wheels has been disabled. check the permissions and o
wner of that directory. If executing pip with sudo, you may want sudo's -H flag.
Collecting pyserial
  Downloading https://files.pythonhosted.org/packages/0d/e4/2a744dd9e3be04a0c090
7414e2a01a7c88bb3915cbe3c8cc06e209f59c30/pyserial-3.4-py2.py3-none-any.whl (193k
B)
    100% |████████████████████████████████████████| 194kB 1.1MB/s
Installing collected packages: pyserial
Successfully installed pyserial-3.4
```

5. Go to the `HUSKYLENSUploader` folder.

```
cd HUSKYLENSUploader
```



```
user@ubuntu:~/Downloads$ cd HUSKYLENSUploader-master
```

6. According to the actual firmware version, enter the full name of the firmware in the following script. Take V0.5.1 version firmware as an example, run the following script to upload the firmware:

```
sudo python3 kflash.py -b 2000000 HUSKYLENSWithModel V0.5.1Norm.kfpkg
```

```
user@ubuntu:~/Downloads/HUSKYLENSUploader-master$ sudo python3 kflash.py -b 2000000 HUSKYLENSWithModelV0.4.6Stable.kfpkg
[sudo] password for user:
[INFO] COM Port Auto Detected, Selected /dev/ttyUSB0
[39][INFO] Default baudrate is 115200 , later it may be changed to the value you set.
[39][INFO] Trying to Enter the ISP Mode...
[39]_
[INFO] Greeting Message Detected, Start Downloading ISP
Downloading ISP: |-----| 2.9% Complete
Downloading ISP: |-----| 5.9% Complete
Downloading ISP: |-----| 8.8% Complete
Downloading ISP: |-----| 11.8% Complete
Downloading ISP: |-----| 14.7% Complete
Downloading ISP: |-----| 17.6% Complete
Downloading ISP: |-----| 20.6% Complete
Downloading ISP: |-----| 23.5% Complete
Downloading ISP: |-----| 26.5% Complete
Downloading ISP: |-----| 29.4% Complete
Downloading ISP: |-----| 32.4% Complete
Downloading ISP: |-----| 35.3% Complete
Downloading ISP: |-----| 38.2% Complete
```

7. Wait about 5 minutes to complete the uploading.

```
[INFO] Writing clearFlash.bin into 0x00d90000
Downloading: |-----| 100.0% Complete
[INFO] Writing clearFlash.bin into 0x00da0000
Downloading: |-----| 100.0% Complete
[INFO] Writing clearFlash.bin into 0x00db0000
Downloading: |-----| 100.0% Complete
[INFO] Writing detect.kmodel into 0x00600000
Downloading: |-----| 100.0% Complete
[INFO] Writing key point.kmodel into 0x0065f000
Downloading: |-----| 100.0% Complete
[INFO] Writing feature.kmodel into 0x00680000
Downloading: |-----| 100.0% Complete
[INFO] Writing mobilenetv1 1.0.kmodel into 0x001bb000
Downloading: |-----| 100.0% Complete
[INFO] Writing object_detect.bin into 0x009d4000
Downloading: |-----| 100.0% Complete
[INFO] Rebooting...
```

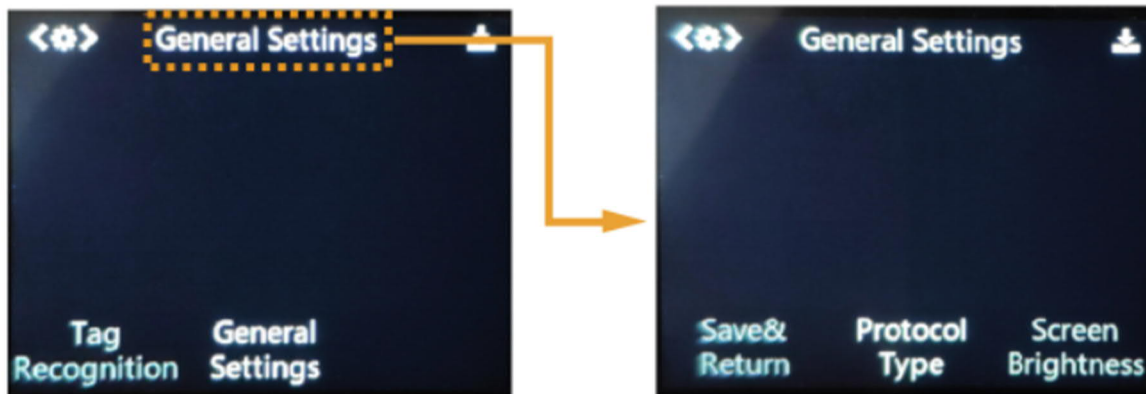
8. Upgrade has been completed now. Enjoy it.

5. General Settings

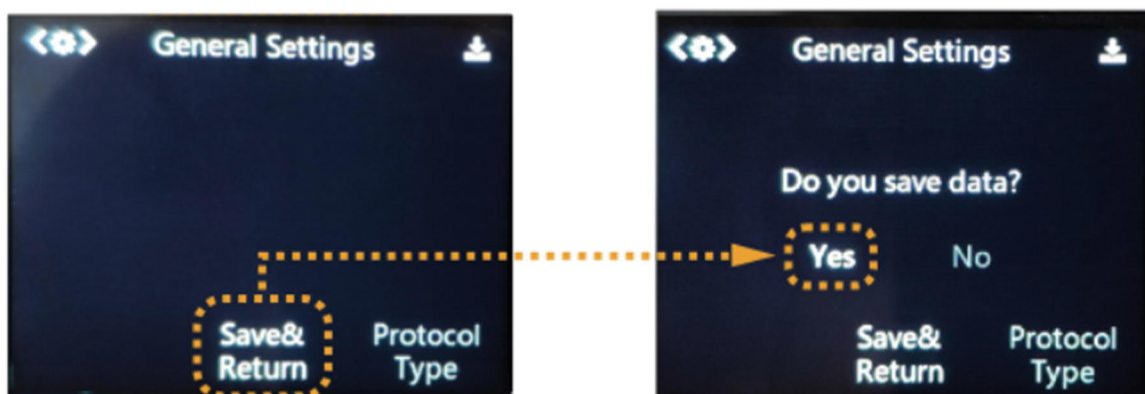
5.1 Basic Operation

The basic operation is shown as follows:

1. Select the general settings: Dial the function button right until the words "General Settings" is displayed at the top of the screen, then select it.
2. Enter the general setting mode: Short press the function button to enter it.



3. Select and adjust the parameter: Dial the function button left or right to select different parameter, then short press the function button to enter the parameter. Dial the function button left or right to adjust the parameter. Then short press the function button again to confirm the parameter.
4. Save the settings: After adjusting the parameters, dial the function button left to select "Save & Return", then short press the function button. A message "Do you save data?" will appear. The default selection is "Yes". At this time, short press the function button to save and exit.



Tips:

- The parameters in the general settings are all selected and adjusted by dialing "function button" left or right, and pressing the "function button".
- After setting the parameters, be sure to select "save and return" to save the set parameters.

5.2 Parameters Introduction

There are 10+ different parameter settings in the general settings.

- **Protocol Type**

Huskylens supports three UART baud rates (9600, 115200, 1000000), and I2C protocol. In addition, it supports auto-detection of the protocols, that is, Huskylens will automatically switch between UART and I2C. We recommend to use the auto detection protocol, which is simple and convenient. The default value is auto-detection.

- **Screen Brightness**

The screen supports the brightness from 1 ~ 100. The default value is 80.

- **Menu Auto-hide**

When you don't operate the Huskylens for a period of time, the menu on the screen will automatically hide. This duration time can be adjusted from 1-100 seconds. The default value is 10 seconds.

- **LED Light**

There are two LED lights on the front of the Huskylens. You can set it ON or OFF. The default value is OFF.

- **LED Brightness**

The brightness of these two LED lights ranges from 1 to 100. The default value is 50.

- **RGB Light**

There is also an RGB light on the front of the Huskylens. You can set it ON or OFF. The default value is ON.

- **RGB Brightness**

The brightness range of this RGB light is 1 to 100. The default is 20.

- **Factory Reset**

Huskylens can be reseted to factory settings via this function.

- **Version**

The current version of the built-in firmware.

- **Language**

Huskylens supports Chinese and English.

6. Color and Coordinate

6.1 Color Instructions

In each function, the color definitions of the frame and the symbol "+" in the center of the screen are all the same, which helps you know the current status of HuskyLens.

Color	Status
From orange to yellow, then from yellow to orange	Have not learned the object yet but ready to learn
Yellow	Learning the new object
Blue	Have learned the object and recognized it

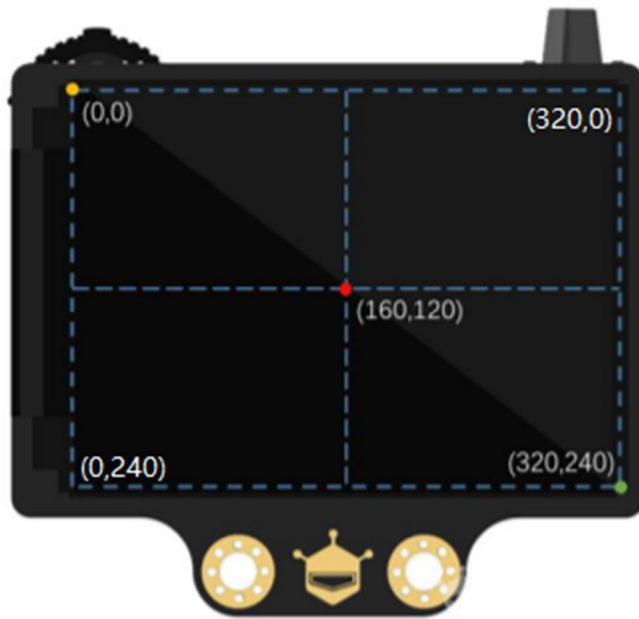
The RGB LED indicator is used to indicate the status of the face recognition function. Its colors are defined as follows.

Color	Status
Blue	Have not learned the face yet, but detected the face
Yellow	Learning the new face
Green	Have learned the face and recognized it

6.2 Coordinate System

When HuskyLens is detecting a object, the target will be automatically selected by a color frame on the screen. The coordinates of the color frame position x and y are assigned according to the following coordinate system. After getting the coordinates from the UART / I2C port , you can know the position of the object.

Format: (x,y)



7. Functions Introduction

7.1 Face Recognition

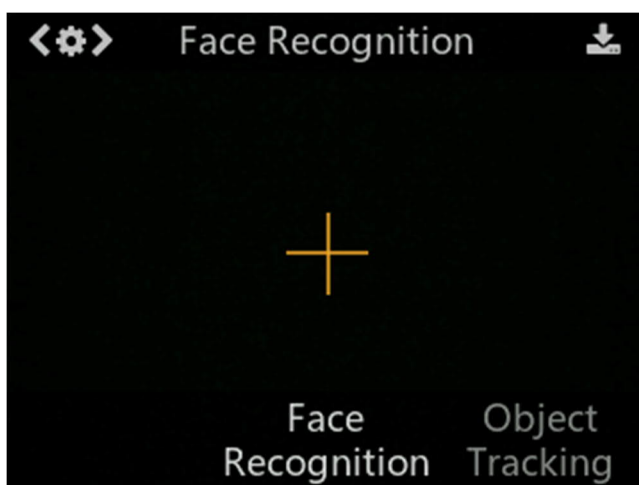
This function can detect any face contour, recognize and track the learned face.

7.1.1. Recognize one face

The default setting is to learn and recognize a single face.

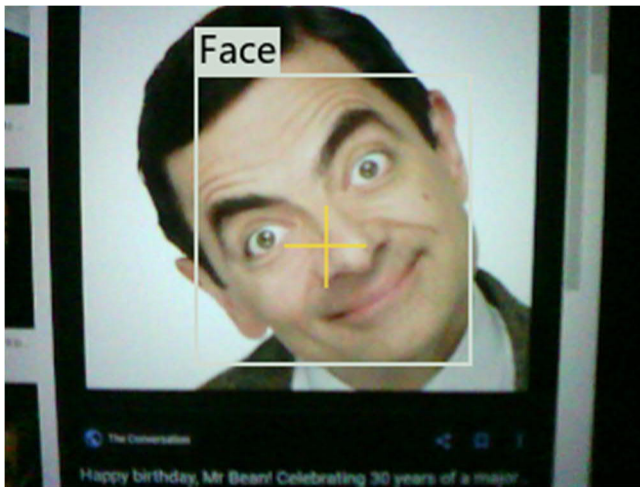
Operation and Setting

Dial the function button to the left until the word "Face recognition" is displayed at the top of the screen.



Learning and Detection

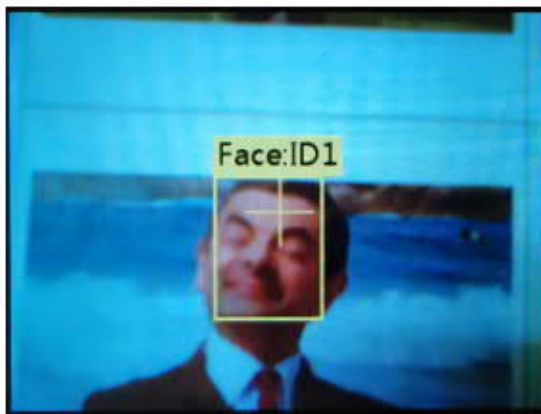
1. Face Detection: Point the HuskyLens at any faces. When a face is detected, it will be automatically selected by a white frame with words "Face" on the screen.



Tips:

If you want HuskyLens to learn or recognize your face, that is, take a selfie, you can't see the screen at this time, you can determine the status according to the different colors of the RGB indicator.

2. Face Learning: Point the "+" symbol at a face, short press the "learning button" to learn the face. If the same face is detected by HuskyLens, a blue frame with words "Face: ID1" will be displayed on the screen, which indicates that HuskyLens has learned the face and can recognize it now.

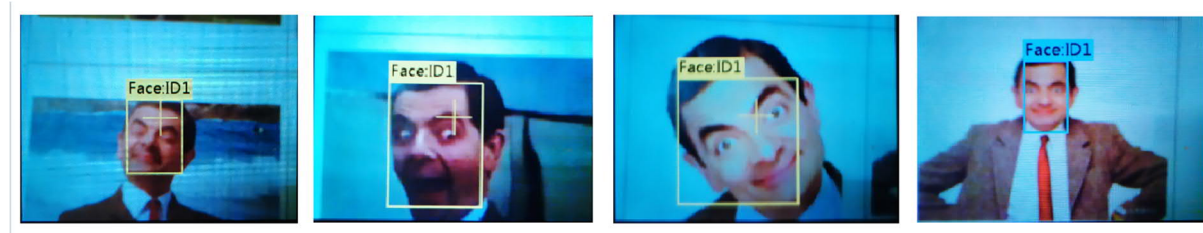


However, HuskyLens only learned one plane (one-dimensional) of the face after the above operation, while human face is three-dimensional. If the angle of the face has been changed, HuskyLens may not recognize it. So you need to let HuskyLens learn a face from its different angles.

The operation shows as follows:(Before HuskyLens learning news thing, please let it forget the former things first.)

Keep pressing the “learning button”, point HuskyLens' "+" symbol at different angles of the face. During this process, a yellow frame with words "Face: ID1" will be displayed on the screen, which indicates HuskyLens is learning the face . Please point the yellow frame at different angles of the same person's face, such as front face and side face (or multiple photos of the same person), to enter all angles of this person's face.

Then you can release the "learning button" to finish the learning. When Huskylens detected the learned face, a blue frame with words "Face: ID1" will be displayed, now HuskyLens can recognize the face from different angles.



Tips:

If there is no “+” symbol in the center of the screen, it means that the HuskyLens has already learned the face in the current function, now HuskyLens is detecting it. If you want to let HuskyLens learn a new face, you have to make it forget the learned face first.

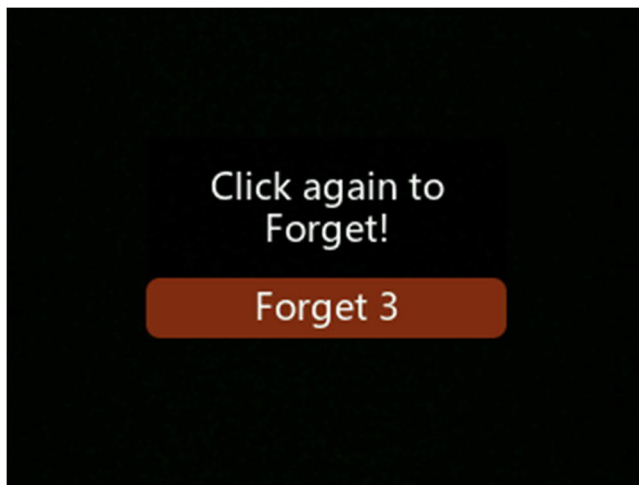
3. Face Recognition: The learned face information will be automatically saved. When HuskyLens detects the learned face, this face will be selected by a blue frame and identified as face: ID1. The size of the frame will change with the size of the face , and the face will be tracked automatically.



4. Forget the Learned Face: If there is no “+” symbol in the center of the screen, it means that the HuskyLens has already learned the face in the current function. If you want to recognize another face, or re-enter face information, you need to delete the current face information.

When HuskyLens is in the face recognition mode, short press the "learning button", the screen will display "click again to forget". Before the countdown ends, short press the "learning button" again to delete the learned face information, then the yellow "+" symbol is displayed. Now you can let HuskyLens learn a new face.

The operation of forgetting is totally the same in other functions. Therefore, this operation will not be repeated in subsequent chapters.

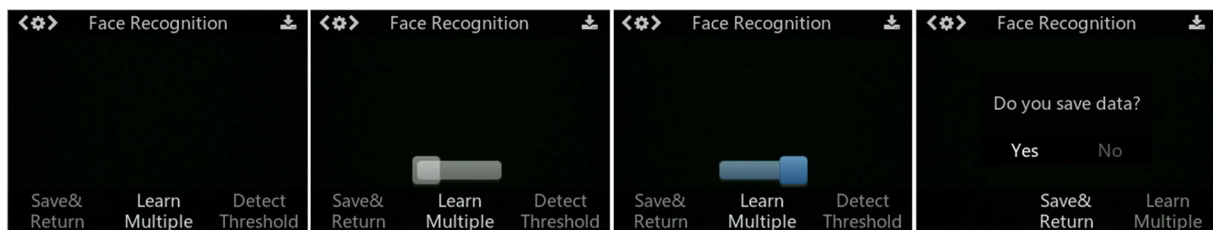


7.1.2. Recognize multiple faces

The default setting is to learn a single face. In order to learn multiple faces, we need to enable "Learn Multiple" of face recognition.

Operation and Setting

1. Dial the function button to the left until the word "Face recognition" is displayed at the top of the screen.
2. Long press the function button to enter the parameter setting of the face recognition function.
3. Dial the function button until "Learn Multiple" is displayed, then short press the function button, and dial to the right to turn on the "Learn Multiple" switch, that is, progress bar turns blue and the square icon on the progress bar moves to the right. Then short press the function button to confirm this parameter.
4. Dial the function button to the left until "Save & Return" shows. And the screen prompts "Do you want to save the parameters?" Select "Yes" in default, now short-press the function button to save the parameters and return automatically .

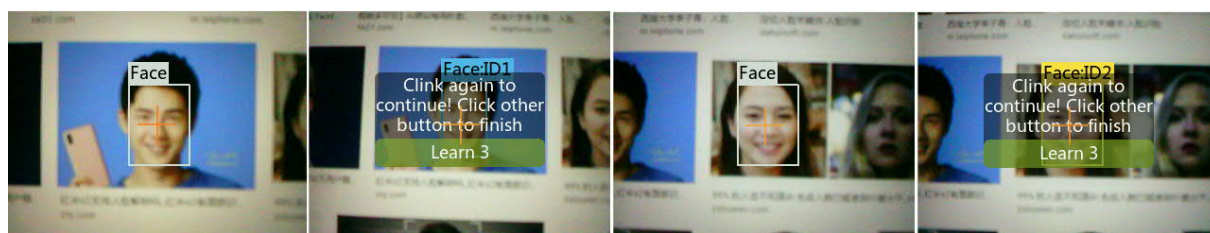


Learning and Detection

1. Multiple Faces Learning: Point the “+” symbol at the face, long press the "learning button" to learn the face of the first person. Then release the "learning button", a blue frame

with words "Face: ID1" will be detected if HuskyLens the same face, meanwhile, a message "Click again to continue! Click other button to finish" will be displayed. Please short press the "learning button" before the countdown ends if you want to learn the face of other person. If not, short press the "function button" before the countdown ends, or do not press any button to let the countdown ends.

In this chapter, we will learn the next face continuously. So we need to short press the "learning button" before the countdown ends. Then we can let HuskyLens learn the face of the second person. The same as the steps to recognize the first face, point the "+" symbol at the second face, long press the "learning button" to learn the face of the second person. Then release the "learning button", a blue frame with words "Face: ID2" will be displayed if HuskyLens detects the same face.



The face ID is the same as the order of entering the face, that is: the learned faces will be marked as "face: ID1", "face: ID2", "face: ID3" in order, and different face IDs correspond to different color frames.

Tips:

If there is no "+" symbol in the center of the screen before learning, it means that the HuskyLens has already learned, now HuskyLens is detecting face. If you want to let HuskyLens learn new face, you need to let HuskyLens forget the learned face first.

Please turn to the [7.1.1. Learn one face](#) to check the way to forget the learned face.

2. Multiple Faces Recognition: The learned face information will be saved automatically. When HuskyLens detects the learned face from multiple faces, the face will be selected with a frame and identified by the message face: IDx. For example, when HuskyLens detects the learned face of the first person, it will be selected with a blue frame and identify face: ID1; when HuskyLens detects the learned face of the second person, it will be selected with a yellow frame and identify face: ID2; and so on.

The color of the frame corresponding to different face IDs is also different, and the frame size will change with the size of the face, and the face will be automatically tracked.

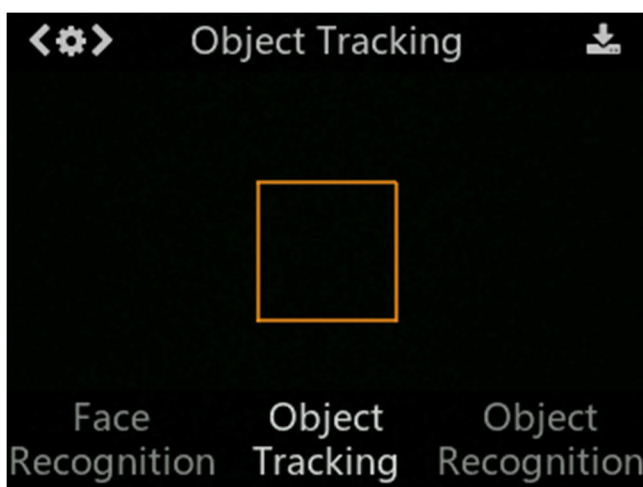


7.2 Object Tracking

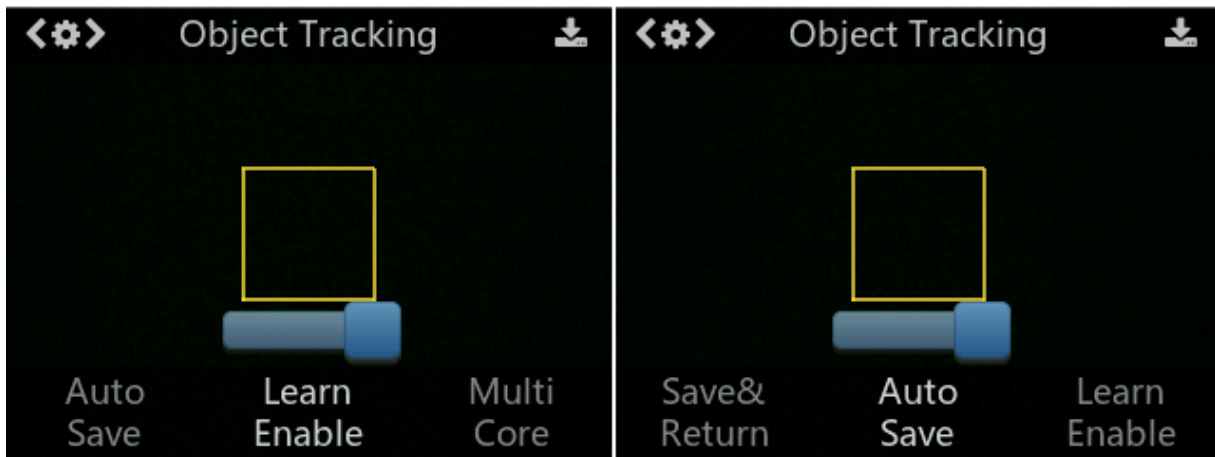
This function can learn and track a specified object. Only one object can be tracked, and multiple objects are not supported.

Operation and Setting

1. Dial the function button to the left or right until the word "Object Tracking" is displayed at the top of the screen.



2. Long press the function button to enter the parameter setting of the object tracking function.
3. Dial the function button to the right to select "Learn Enable", then short press the function button, and dial it to the right to turn the "Learn Enable" ON, that is, the square icon on the progress bar is turned to the right. Then short press the function button to confirm this parameter.
4. The method to turn on the switch of saving models automatically is the same as before. According to the steps above to switch "Auto Save" ON.



5. You can also adjust the size of the frame by setting "Frame Ratio" and "Frame Size" to match the shape of the object.
6. Dial the function button to the left to select "Save & Return", and short press the function button to save the parameters and return automatically.

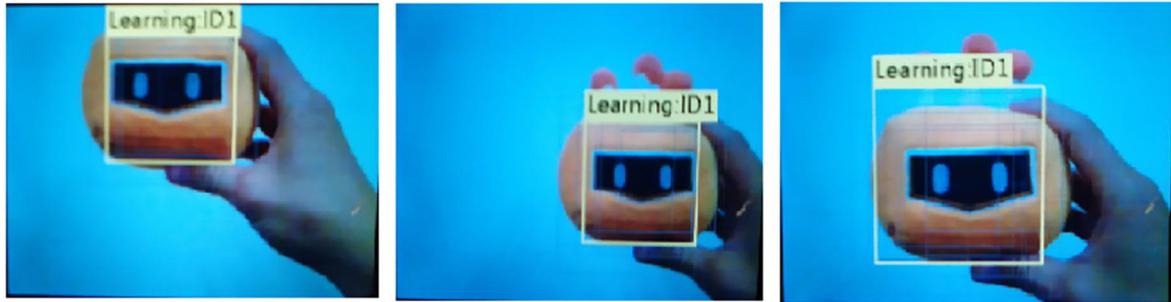
Learning and Detection

1. Object Learning: Point HuskyLens to the target object, adjusting the distance and until the object is included in the yellow frame of the center of the screen. Then long press "learning button" to learn the object from various angles and distances. During the learning process, the yellow frame with words "Learning: ID1" will be displayed on the screen.



When HuskyLens can track the object at different angles and distances, you can release the "learning button" to complete the learning.

2. Object Tracking: Move the HuskyLens or the target, the frame will track the target automatically. When tracking the object, the yellow words "Learning: ID1" will be displayed, indicating that HuskyLens is tracking the object while learning. This setting improves the object tracking ability.



When the recognition result meets the requirements, you can turn off "Learn Enable".

Tips:

Only one object can be tracked at a time. It can be any object with a clear outline, even various gestures.

7.3 Obejct Recognition

This function can recognize what it is, and track it.

HuskyLens can recognize 20 built-in objects. They are aeroplane, bicycle, bird, boat, bottle, bus, car, cat, chair, cow, dining-able, dog, horse, motorbike, person, potted lant, sheep, sofa, train, TV.

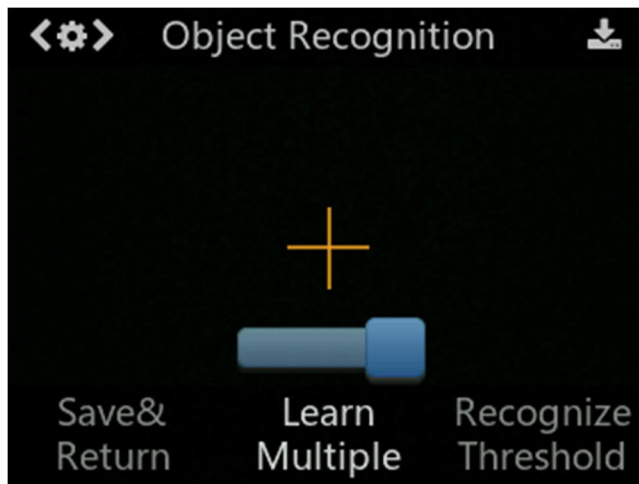
Tips:

Object recognition cannot distinguish the difference between objects of the same type. For example, it can only recognize that this is a cat, but it cannot recognize what kind of cat it is. Different from face recognition, people are one type, but different faces can be distinguished.

The default setting is to recognize a single object. This chapter uses marking and recognizing multiple objects as an example to demonstrate.

Operation and Setting

1. Dial the function button to the left or right until the word "Obejct Recognition" is displayed at the top of the screen.
2. Long press the function button to enter the parameter setting of the object recognition function.
3. Dial the function button until "Learn Multiple" is displayed, then short press the function button, and dial to the right to turn on the "Learn Multiple" switch, that is, progress bar turns blue and the square icon on the progress bar moves to the right. Then short press the function button to confirm this parameter.



4. Dial the function button to the left until "Save & Return" shows. And the screen prompts "Do you want to save the parameters?" Select "Yes" in default, now short-press the function button to save the parameters and return automatically .

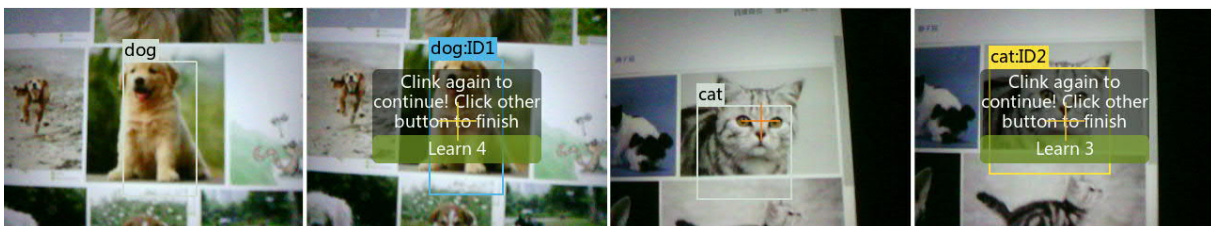
Learning and Detection

1. Object Detection: When detecting objects, HuskyLens will automatically recognize it, and the object will be displayed by the white frame with its name on the screen.

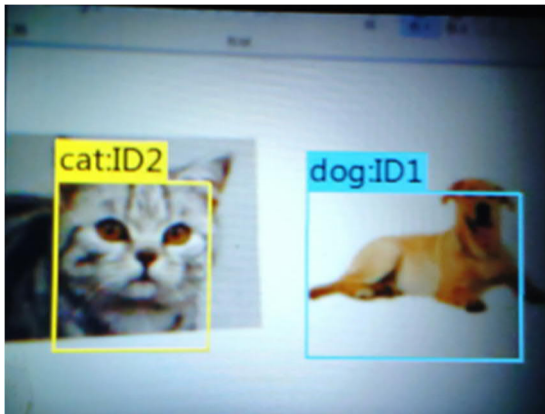
At present, only 20 built-in objects can be recognized, and the remaining objects cannot be recognized temporarily.



2. Object Mark: Point the “+” symbol at the object, then short press the “learning button”, the color of the frame changes from white to blue, and the name of the object and its ID1 will appear on the screen, meanwhile, a message "Click again to continue! Click other button to finish" will be displayed. Please short press the "learning button" before the countdown ends if you want to mark the next object. If not, short press the "function button" before the countdown ends, or do not press any button to let the countdown ends.



3. Object Recognition: When encountering the learned objects, they will be selected by the color frame, and the name and ID number will be displayed. When encountering new ones, the selection frame is white.



The ID number is related to the order of marking objects. For example, if a dog is marked for the first time and a cat is marked for the second time, when the dog is recognized, the words "dog: ID1" will be displayed on the screen; and when the cat is recognized, the words "cat: ID2" will be displayed on the screen.

This can be used as a simple filter to find out what you need from a bunch of objects.

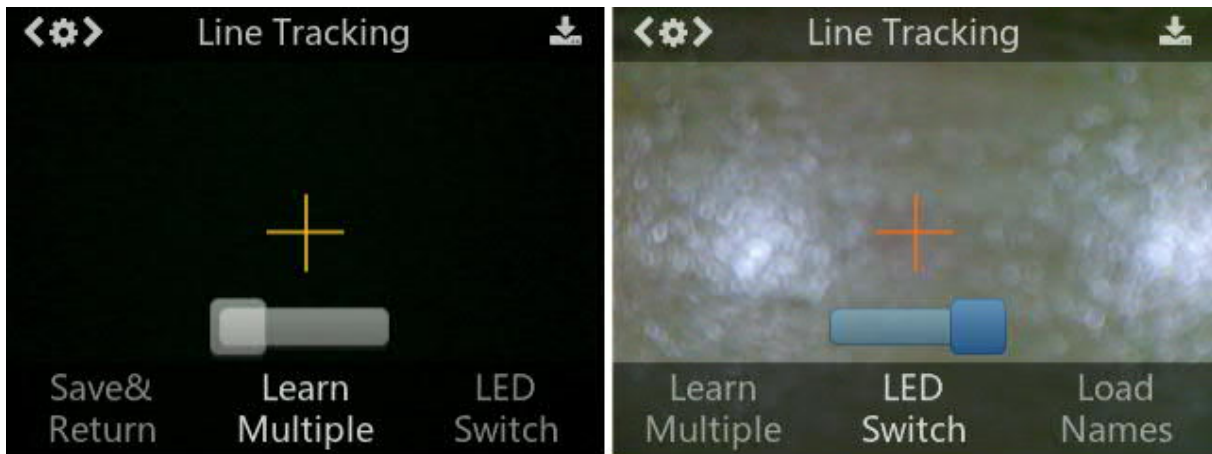
7.4 Line Tracking

This function can track specified color lines and make path predictions. The default setting is to track only one color line.

In this chapter, we use tracking one color line as an example to demonstrate.

Operation and Setting

1. Dial the function button to the left or right until the word "Line Tracking" is displayed at the top of the screen.
2. Long press the function button to enter the parameter setting of the line tracking function.
3. Dial the function button right or left until "Learn Multiple" is selected, then short press the function button, and dial it to the left to turn off the "Learn Multiple" switch, that is, the square icon on the progress bar is turned to the left. Then short press the function button to complete this parameter.



4. You can also turn on the LEDs by setting "LED Switch". This is very useful in the dark environment.
5. Dial the function button to the left until "Save & Return" is selected, and short press the function button to save the parameters and it will return automatically.

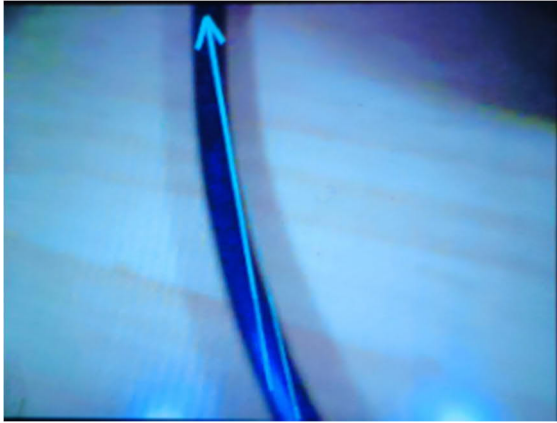
Learning and Detection

1. Line Learning: It is recommended that within the view field of HuskyLens, just remain line to learn and no any cross lines. Point the "+" symbol at the line. Then HuskyLens will automatically detect the line and a white arrow will appear on the screen.

At that time, short press the "learning button" to complete the learning process. A blue route direction arrow will appear on the screen.



2. Line Prediction: When HuskyLens detects the line which has been learned, a blue arrow will appear automatically on the screen. The direction of the arrow indicates the predicted direction of the line.



Tips:

- When learning the line, we need to adjust the position of HuskyLens to be parallel to the line.
- HuskyLens can learn multiple lines according to the color of lines, but these lines must be monochrome lines with obvious color that different from the background.
- In most cases, the color of tracking line is only one. Therefore, in order to ensure stability, we recommend to track the single color line.
- The color of the lines has a lot to do with the ambient light, so please try to keep the ambient light as stable as possible.

7.5 Color Recognition

This function can learn, recognize, and track the specified color.

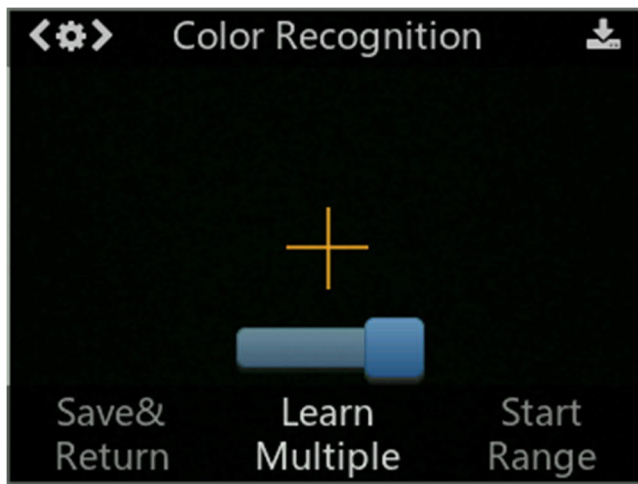
The default setting is to learn, recognize, and track a single color. This chapter uses learning, recognizing, and tracking multiple colors as an example to demonstrate.

Tips:

Color recognition is greatly affected by ambient light. Sometimes HuskyLens may misidentify similar colors. Please try to keep the ambient light unchanged.

Operation and Setting

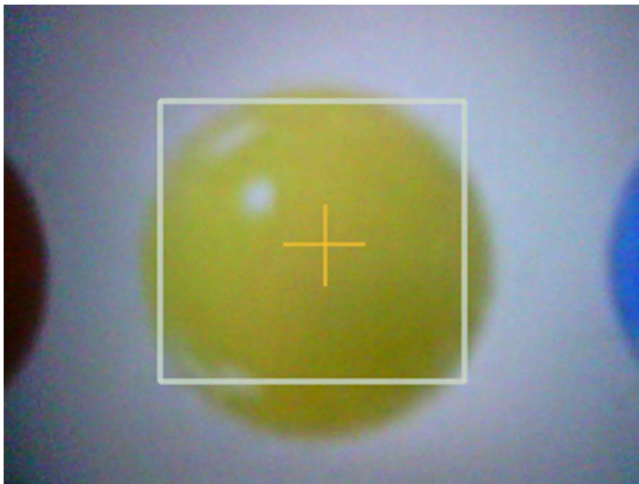
1. Dial the function button to the right or left until the word "Color Recognition" is displayed at the top of the screen.
2. Long press the function button to enter the parameter setting of the color recognition function.
3. Dial the function button until "Learn Multiple" is displayed, then short press the function button, and dial to the right to turn on the "Learn Multiple" switch, that is, progress bar turns blue and the square icon on the progress bar moves to the right. Then short press the function button to confirm this parameter.



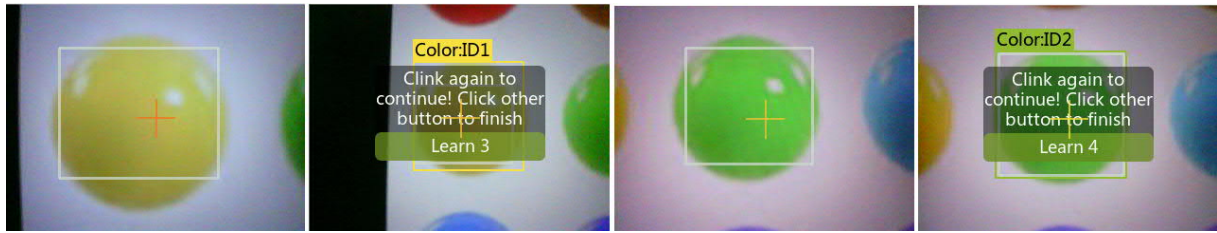
4. Dial the function button to the left until "Save & Return" shows. And the screen prompts "Do you want to save the parameters?" Select "Yes" in default, now short-press the function button to save the parameters and return automatically .

Learning and Detection

1. Color Detection: Point the icon "+" in the center of the HuskyLens screen to the target color block, and a white frame will appear on the screen, which selects the target color block automatically. Adjust the angle and distance of the HuskyLens to the color block so that the white frame can include the entire target color block as far as possible.

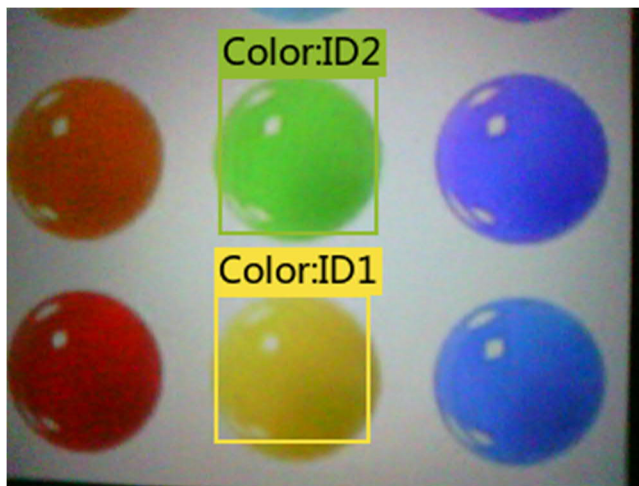


2. Color Learning: Point the "+" symbol at the first color block, and long press the "learning button". A yellow frame will be displayed on the screen, indicating that HuskyLens is learning the color. At this time, adjust the distance and angle between HuskyLens and the color block, to let HuskyLens learn the color block in various distances and angles. Then, release the "learning button" to complete learning the first color block, meanwhile, a message "Click again to continue! Click other button to finish" will be displayed. Please short press the "learning button" before the countdown ends if you want to learn other color blocks. If not, short press the "function button" before the countdown ends, or do not press any button to let the countdown ends.



3. Color Recognition: When encountering the same or similar color blocks, some color frames with IDs will be automatically displayed on the screen, and the size of the frames are same as the size of the color blocks.

The ID number is related to the order of learned color. For example, if a yellow block is marked for the first time and a green block is marked for the second time, when the yellow block is recognized, the words "Color: ID1" will be displayed on the screen, and when the green block is recognized, the words "Color: ID2" will be displayed on the screen.



In the firmware below V0.5.1, when there are multiple same or similar color blocks appear at the same time, the other color blocks cannot be selected, that is, only one color block can be recognized at each time.

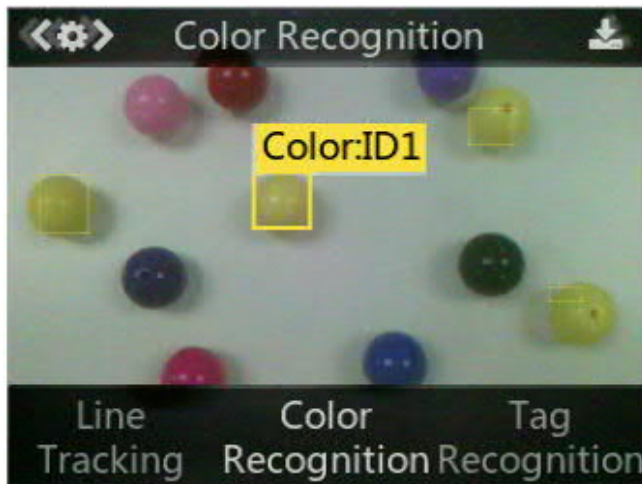
In firmware version V0.5.1 and above, this function is optimized. when there are multiple same or similar color blocks appear at the same time, these color blocks can be recognized and selected at the same time. This function can be used to count the color blocks.

Count Color blocks

In firmware V0.5.1 and above, HUSKYLENS can count the color blocks in the color recognition function, that is, calculate the number of color blocks in the HUSKYLENS screen. The following is an example of recognizing multiple balloons of similar colors.

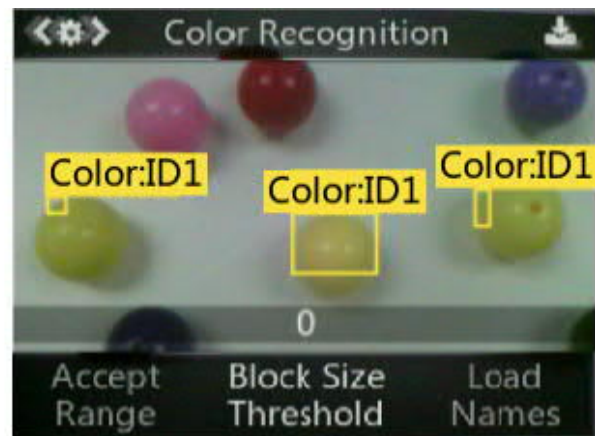
1. Learn and Recognize Colors

Point the "+" symbol at the target balloon, then press the "learning button" to learn the color of the balloon, and then release the "learning button" to complete the learning. You can see that the color of the balloon can be recognized, but for balloons with similar colors, it may not be able to identify them.



2. Adjust the Threshold

When recognizing color blocks of similar colors, the recognition accuracy can be adjusted by setting the threshold. For example, in the figure above, some yellow balloons with similar colors are not recognized, and the threshold may be set higher. In the parameter setting of the color recognition function, there is the "Block Sized Threshold" parameter. The lower the value of this parameter, the lower the accuracy, but the more similar color blocks can be recognized. As shown in the figure below, when the threshold is 20, only one yellow ball can be recognized, and when the threshold is 0, all three yellow balls can be recognized. Please adjust the threshold according to the actual effect, so that the recognition accuracy is within your acceptable range.



With this function, you can easily obtain the number of yellow balls on the screen in real time.

7.6 Tag Recognition

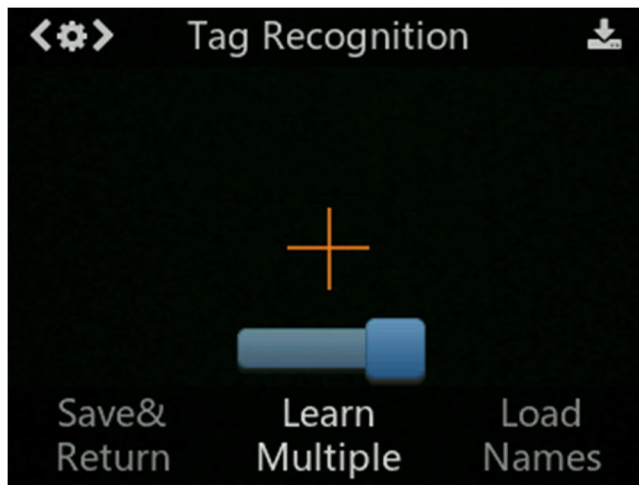
This function can detect tags, and learn, recognize, track specified tags. These tags are called April Tags. [Click here](#) to download more April Tags.

Tips: Only April Tags can be regonized.

The default setting is to learn a single tag. This chapter uses learning, recognizing, and tracking multiple tags as an example to demonstrate.

Operation and Setting

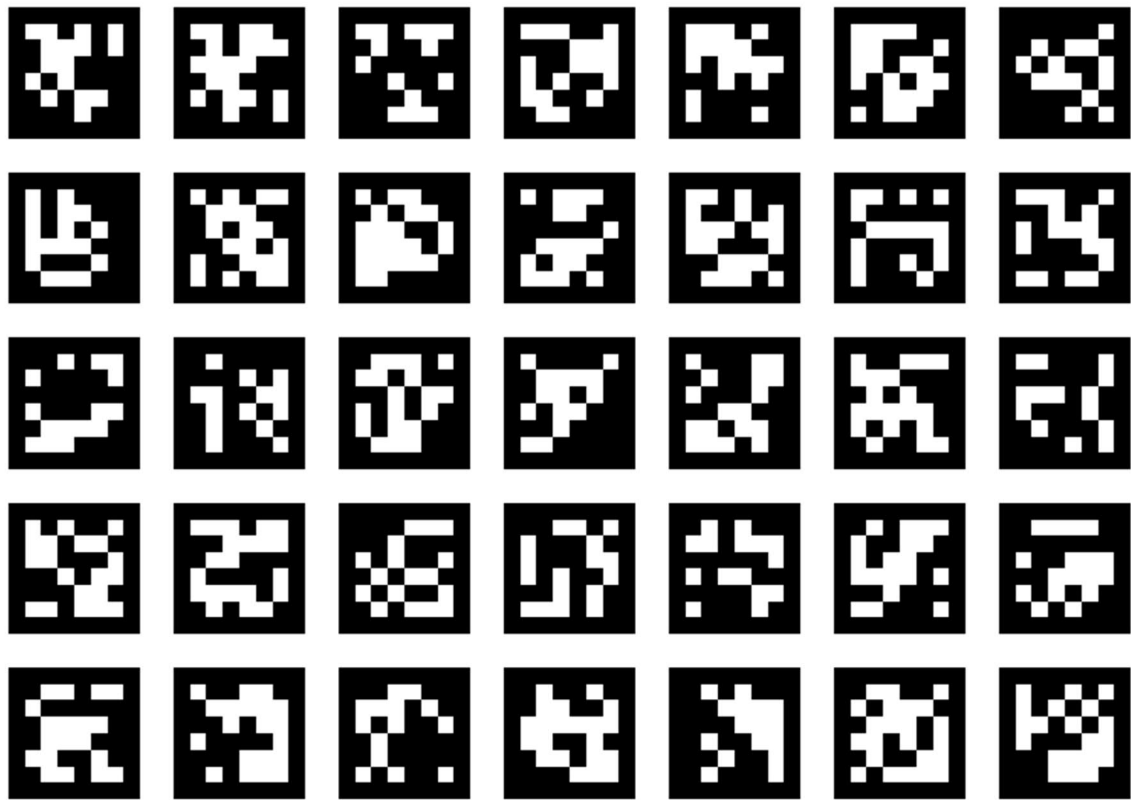
1. Dial the function button to the right or left until the words "Tag Recognition" is displayed at the top of the screen.
2. Long press the function button to enter the parameter setting of the tag recognition function.
3. Dial the function button until "Learn Multiple" is displayed, then short press the function button, and dial to the right to turn on the "Learn Multiple" switch, that is, progress bar turns blue and the square icon on the progress bar moves to the right. Then short press the function button to confirm this parameter.



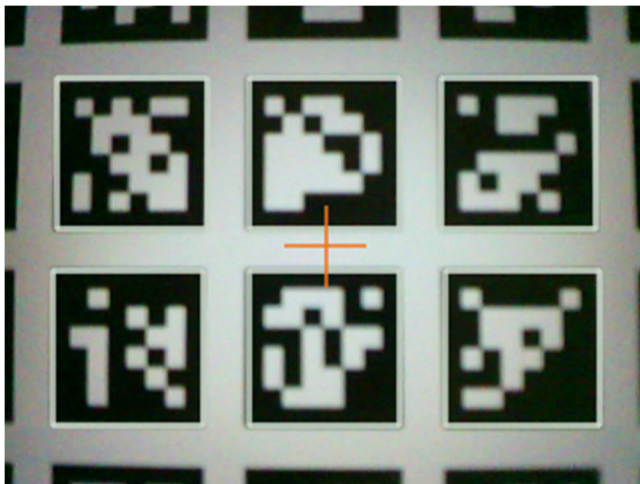
4. Dial the function button to the left until "Save & Return" shows. And the screen prompts "Do you want to save the parameters?" Select "Yes" in default, now short-press the function button to save the parameters and return automatically .

Learning and Detection

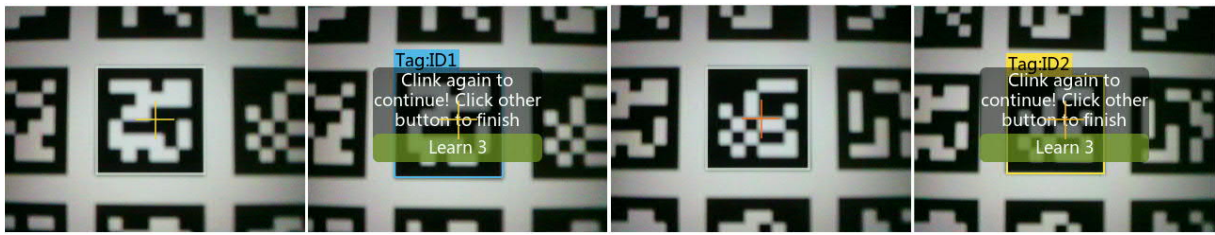
You can use the following tags to test this function.



1. Tag Detection: When Huskylens detects the tag, the tag will be automatically selected by the white frame on the screen.



2. Tag Learning: Point the “+” symbol at the first tag, and press the “learning button”. A yellow frame with words "Tag:ID1" will be displayed on the screen, indicating that HuskyLens is learning the tag now. Then, release the "learning button" to complete learning the first tag, meanwhile, a message "Click again to continue! Click other button to finish" will be displayed. Please short press the "learning button" before the countdown ends if you want to learn other tags. If not, short press the "function button" before the countdown ends, or do not press any button to let the countdown ends.



3. Tag Recognition When encountering the learned tag, some color frames with IDs will be automatically displayed on the screen. The size of the frames changes with the size of the tags, and the frames automatically track these tags.



7.7 Object Classification

This function can learn multiple photos of different objects, and then use the built-in machine learning algorithm for training. After the training is completed, when the learned objects appear again in the HuskyLens' camera, HuskyLens can recognize them and display their ID numbers. The more HuskyLens learns the photos of the same object, the more accurate the recognition can be.

The default setting is to learn multiple objects. This chapter uses recognizing whether a worker wears a helmet as an example to demonstrate.

Operation and Setting

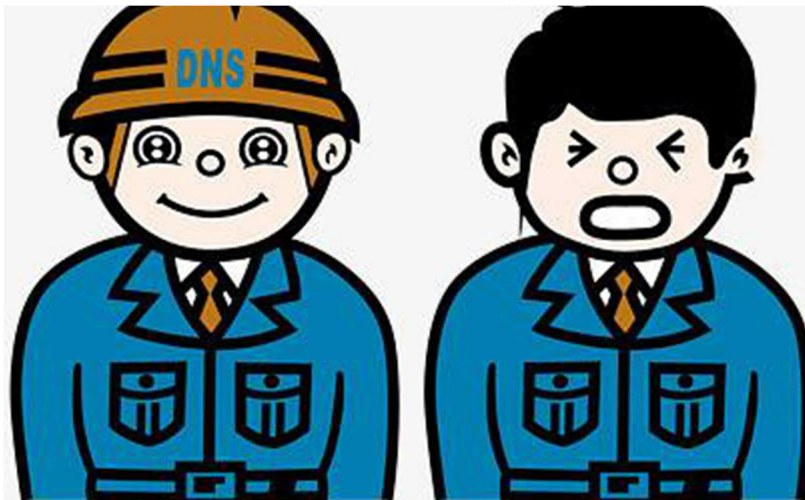
1. Dial the function button to the right or left until the words "Object Classification" is displayed at the top of the screen.
2. Long press the function button to enter the parameter setting of the object classification function.
3. Dial the function button until "Learn Multiple" is displayed, then short press the function button, and dial to the right to turn on the "Learn Multiple" switch, that is, progress bar turns blue and the square icon on the progress bar moves to the right. Then short press the function button to confirm this parameter.



4. Dial the function button to the left until "Save & Return" shows. And the screen prompts "Do you want to save the parameters?" Select "Yes" in default, now short-press the function button to save the parameters and return automatically .

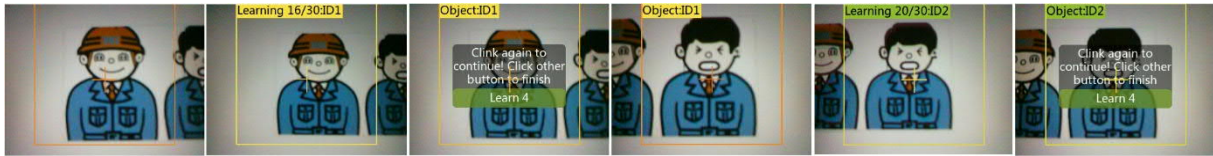
Learning and Detection

You can use the following picture to test.



1. Object Learning:

Point the large frame at the first target object (the worker with a helmet on the left in the picture above), and long press the "learning button", a yellow frame with words "Learning XX/30 ID:1" will be displayed on the screen, indicating that HuskyLens is learning the object now. Adjust the distance and angle, let HuskyLens learn the object in various distances and angles. Then, release the "learning button" to complete learning the first object, meanwhile, a message "Click again to continue! Click other button to finish" will be displayed. Please short press the "learning button" before the countdown ends if you want to learn other objects. If not, short press the "function button" before the countdown ends, or do not press any button to let the countdown ends.

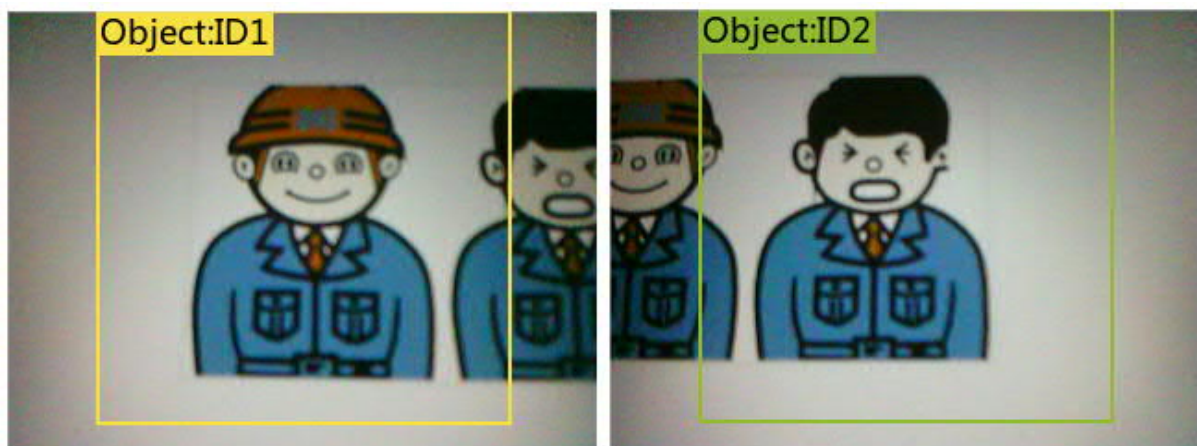


In this chapter, you need to continue to learn the next object (the worker without a helmet on the right in the above picture), so press the "learning button" before the countdown ends, and then point the large frame at the second target object, long press the "learning button" to complete the learning of the second object. And so on.

The order of the object ID and the learned object is the same, that is: the learned objects will be marked as "object: ID1", "Object: ID2", "Object: ID3", and so on, and the color of the frame corresponding to the object is also different.

2. Object Recognition:

When HuskyLens encounters the learned object again, its ID number will be displayed on the screen. As shown in the figure below, when HuskyLens recognizes that the worker is wearing a helmet, the screen displays ID1, and if there is no helmet, it displays ID2.



More interesting ideas based on object

classification: <https://community.dfrobot.com/makelog-308212.html>

FAQ

- **Can the object classification function give the relative position of the object?**

Answer: No. In the object classification function, the position of the frame is fixed, and its x and y center coordinates on the screen remain unchanged, so it cannot give the relative position of the object on the screen. But you can learn different positions of objects as different IDs, and judge the position by ID. For example, in automatic vehicles, learn the ID 1, 2, and 3 as on the left, middle, and right sides of the road. By judging the ID, you can know the position of the automatic vehicles relative to the road.

- **How to improve the accuracy of recognition under the object classification algorithm?**

Answer: Long press the "learning button" without releasing it, you can let HuskyLens learn the target photos from multiple angles and distances, to improve the recognition accuracy.

7.8 Auxiliary Function

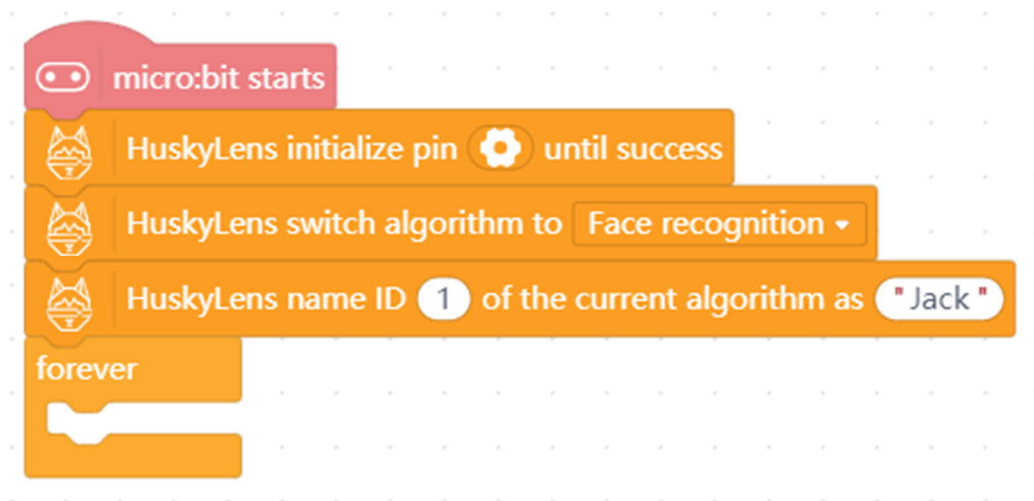
In firmware V0.5.1 and above, more auxiliary functions are added to help you use HUSKYLENS to set up projects better.

7.8.1 Customize the ID Name

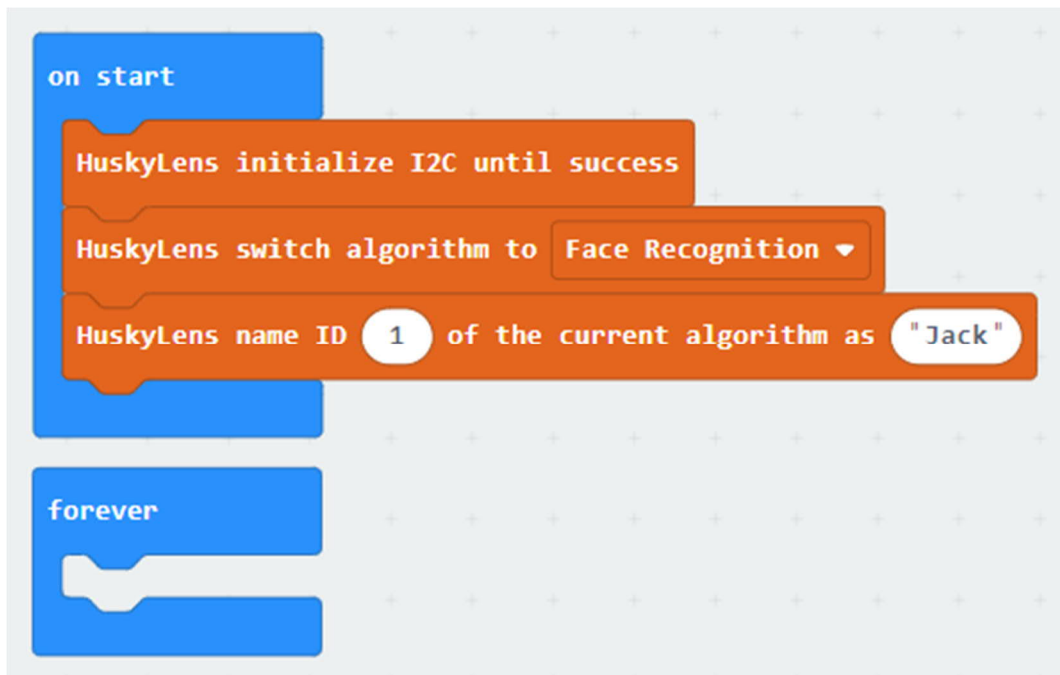
You can change the corresponding name of the ID, for example: change “Face:ID1” to “Jack:ID1”, and “Color:ID1” to “Red:ID1”. This function is used to name a people or object, and make the result of recognition more recognizable. All 6 algorithms support customizing the ID name except line tracking. And only English names are supported.

Demonstration: Change the name of ID1 to Jack in face recognition

Mind+ Sample Program:



MakeCode Sample Program:



Arduino Sample Program:

```
#include "HUSKYLENS.h"

HUSKYLENS huskyLens;

void setup()
{
    Serial.begin(115200);
    Wire.begin();
    while (!huskyLens.begin(Wire))
    {
        Serial.println(F("Begin failed!"));
        delay(100);
    }
    while (!huskyLens.setCustomName("Jack", 1)) // bool setCustomName(String
name,uint8_t id)
    {
        Serial.println(F("ID1 customname failed!"));
        delay(100);
    }
}

void loop()
{
}
```

Result

"Jack:ID1" will be shown on screen after HUSKYLENS learning the first face ID1. As shown below:

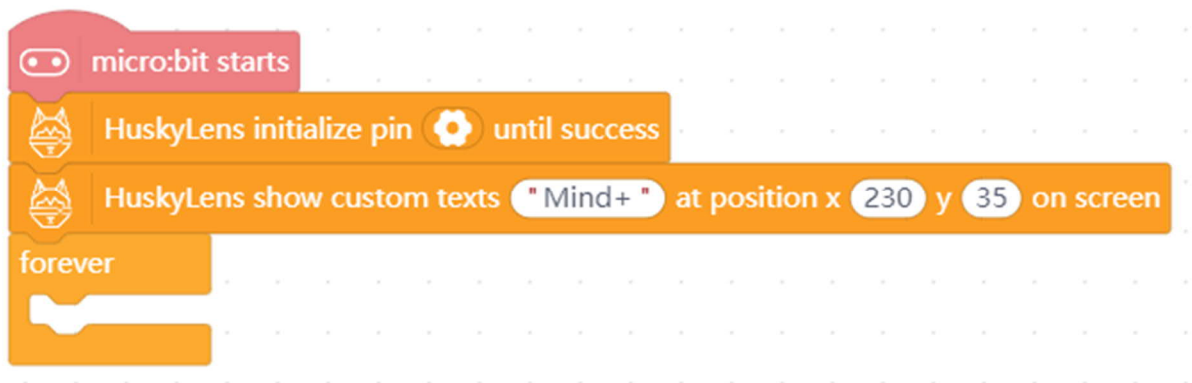


7.8.2 Displaying Customized Text on the Screen

The onboard screen can be used to display texts anywhere on itself. It supports English characters, numbers and symbols. The recognition results and data from sensors can be directly displayed on the screen. All 7 algorithms support the customized text.

Demonstration: Displaying "Mind+" on the upper right corner at coordinates of (230,35)

Mind+ Sample Program:



MakeCode Sample Program:



Arduino Sample Program:

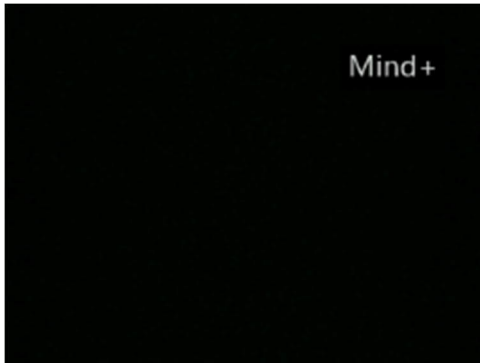
```
#include "HUSKYLENS.h"

HUSKYLENS huskyLens;

void setup()
{
    Serial.begin(115200);
    Wire.begin();
    while (!huskyLens.begin(Wire))
    {
        Serial.println(F("Begin failed!"));
        delay(100);
    }
    while (!huskyLens.customText("mind+", 230, 35)) // bool customText(String
text, uint16_t x, uint8_t y)
    {
        Serial.println(F("custom text failed!"));
        delay(100);
    }
}

void loop()
{
}
```

Result



7.8.3 Saving the Photos or Screenshots into SD Card

Like a digital camera, HUSKYLENS can take photos or screenshots, and save them on an SD card. With an onboard SD card slot on HUSKYLENS, you can just plug in the SD card and use it. The screenshots contain the texts, frames displayed on the screen, while the photos contain only the image.

All 7 algorithms support this function. An SD card is required when using this function.

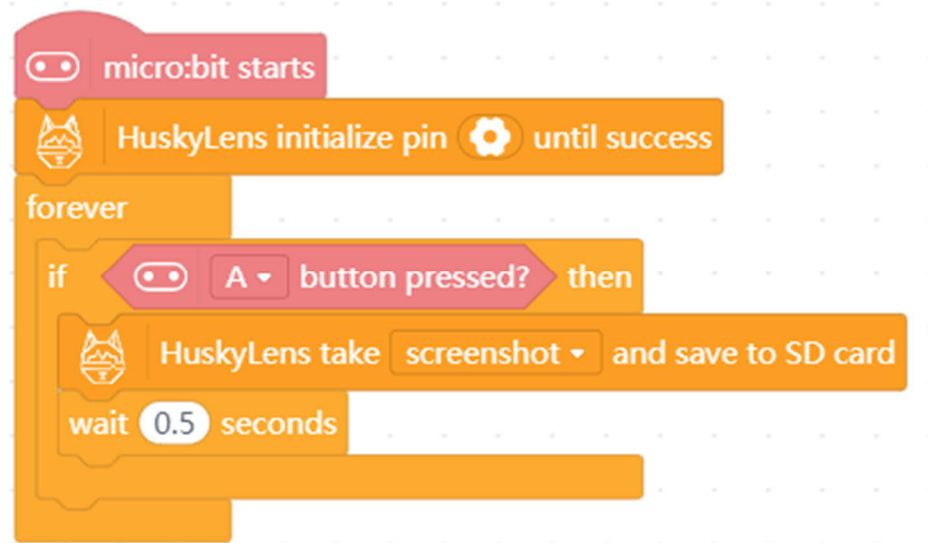
Tips:

- First-line brands, such as SanDisk, Toshiba, Samsung, and Kingston are recommended. Incompatibility problems may exist between SD cards of small brands.
- The images saved by HUSKYLENS is with resolution ratio of 320*240. And it is unchangeable.
- Please format the SD card as FAT32. Taking photos and screenshots, and saving them costs some time. So, it is recommended to use this function at least 0.5 seconds apart.
- The SD card slot is facing inwards, so, the SD card should be inserted outwards as shown below:



Demonstration: Press button A of micro:bit or pull down the Arduino Pin A0, then HUSKYLENS will take a screenshot and save it in SD card

Mind+ Sample Program:



MakeCode Sample Program:



Arduino Sample Program:


```

#include "HUSKYLENS.h"

HUSKYLENS huskylens;

void setup()
{
    Serial.begin(115200);
    pinMode(A0, INPUT_PULLUP);
    Wire.begin();
    while (!huskylens.begin(Wire))
    {
        Serial.println(F("Begin failed!"));
        delay(100);
    }
}

void loop()
{
    if(digitalRead(A0) == 0)
    {
        while (!huskylens.saveScreenshotToSDCard()) // bool saveScreenshotToSDCard()
or bool savePictureToSDCard()
        {
            Serial.println(F("save screenshot to SD card failed!"));
            delay(100);
        }
        Serial.println(F("saving screenshot to SD card..."));
        delay(500); // The interval between calling this function is not less than 0.5
seconds.
    }
}

```

Result

Press button A of micro:bit or pull down the Arduino Pin A0, then HUSKYLENS will take a screenshot and save it in SD card. Take out the SD card and insert it into a computer, then you can view the screenshot taken by HUSKYLENS.

- The image shown on the screen of HUSKYLENS



- Read the image file(.bmp) on the SD card on a computer

 1.bmp	226 KB
 2.bmp	226 KB

- Open the image file



7.8.4 SD Card saving/loading the models

HUSKYLENS supports taking multiple objects learned from the same algorithm as one data model. The SD card can be used to save the model. The model can be reloaded even if all the data from the current algorithm is deleted, allowing HUSKYLENS to automatically learn the data in the model. By saving the model, HUSKYLENS can be used for multiple scenes in one algorithm. For example, in the object classification algorithm, learn rock-paper-scissors as one model and garbage classification as one model. Then, through model switching, the required functions can be quickly realized to avoid repeating learning.

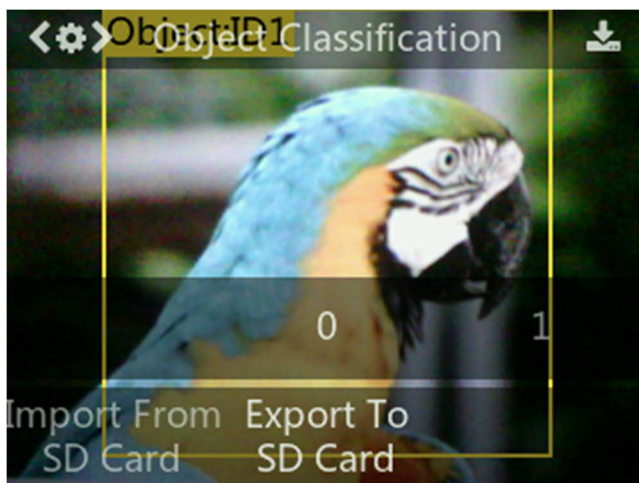
This function requires an SD card, which needs to be formatted as FAT32. All 7 algorithms, each of which can save 5 models, support SD card saving and loading models.

Method 1: Operate manually under the secondary menu of each algorithm

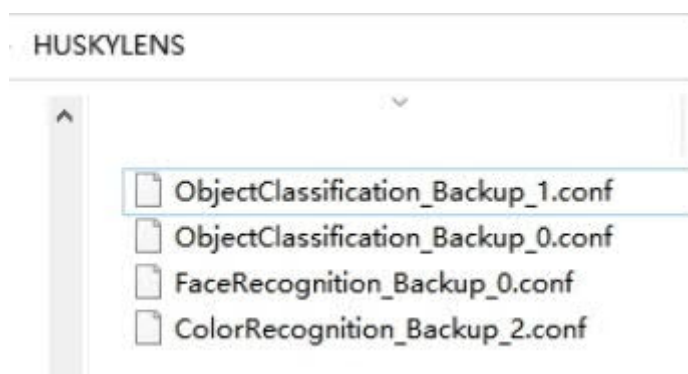
Here we take the object classification function as a demonstration. The usage in other algorithms are the same.

1. Learn new objects : Let HUSKYLENS learn several new objects. For example: in the object classification function, learn several animals with different ID numbers in turn.

2. Export model : long press "Function Button" to enter parameter settings interface of the secondary menu in the classification function, and then dial "Function Button" to the right until the option of "export to SD card" (that is, save the model to the SD card). Short press "Function Button", dial "Function Button", select anyone from 0 to 4(equivalent to select the save location, making it easy to distinguish the different models), then short press "Function Button" to save the model.



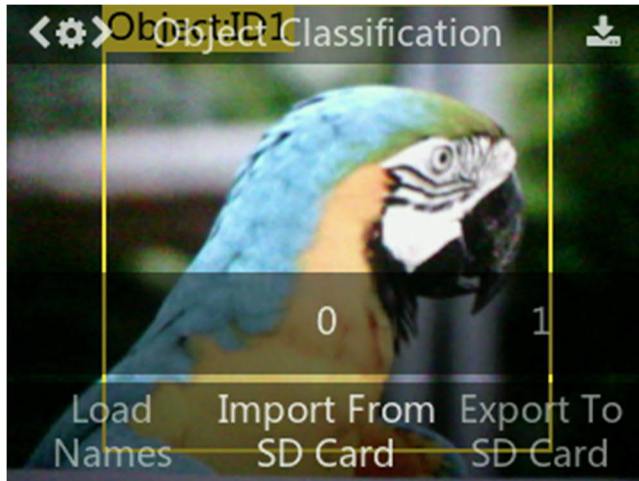
After the model is exported to the SD card, read the SD card with the computer, and you can see the model data file with suffix .conf in the "HUSKYLENS" folder. The file names saved by different algorithms are different. Shown as below :



Tip: According to the names of the files, we can know which algorithm function the exported model data file belongs to. The pictures learned in each model cannot be viewed.

3. Import model : When the model data file is already in the SD card, we can directly import it into HUSKYLENS. For example, in the object classification function, if the model data file of object classification is already in the SD card, then , just select "Import from SD Card" in parameter setting interface of secondary menu of object classification function, and select anyone from 0 to 4 to load the model into HUSKYLENS.

After importing the model, HUSKYLENS can realize recognition using the current model.

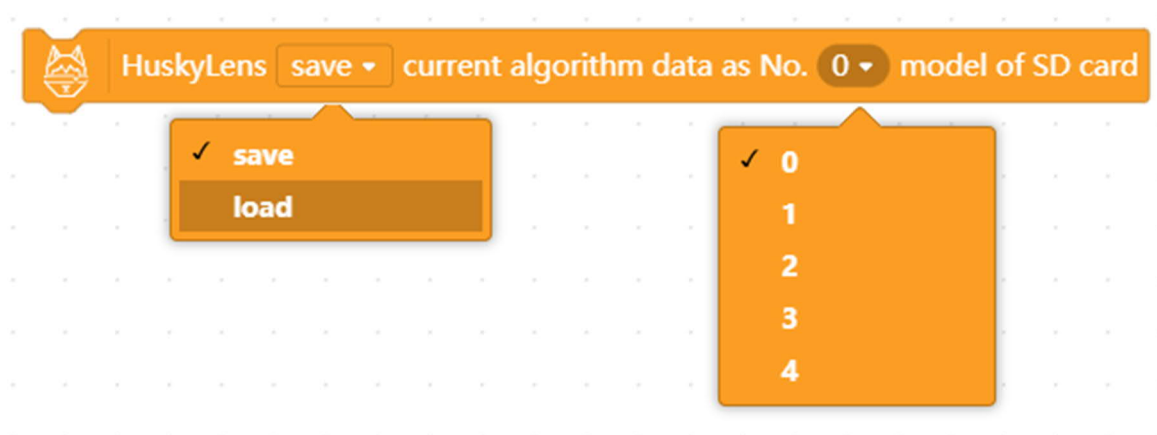


Tips: When importing models, only models with the same algorithm are supported.

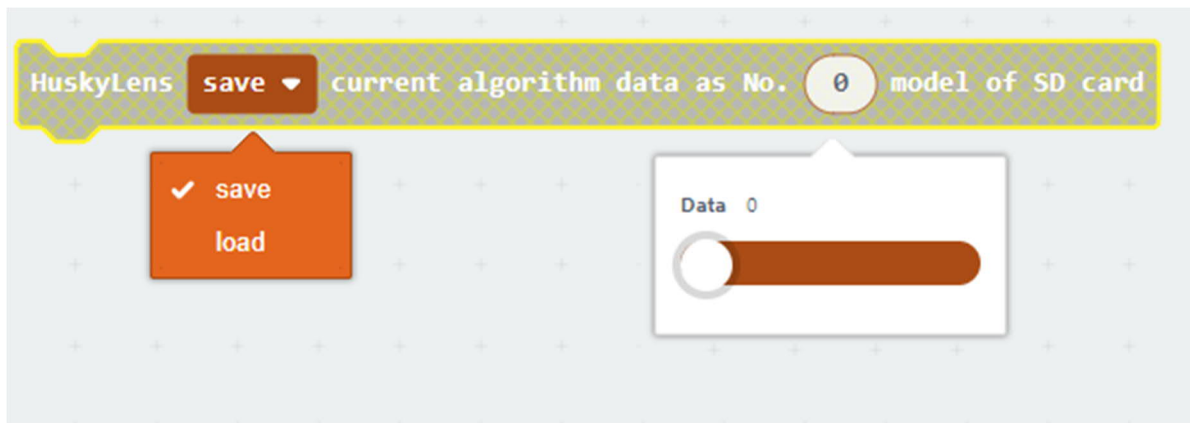
Method 2: Make A Program to Trigger Saving/Loading Models

In this way, micro:bit, Arduino, and other mainboards can be used to automatically trigger the saving or loading of model files. The following code block can be used to achieve this function.

Mind+ Block:



MakeCode Block:



Arduino function:

```
bool saveModelToSDCard(int fileNum);
bool loadModelFromSDCard(int fileNum);
```

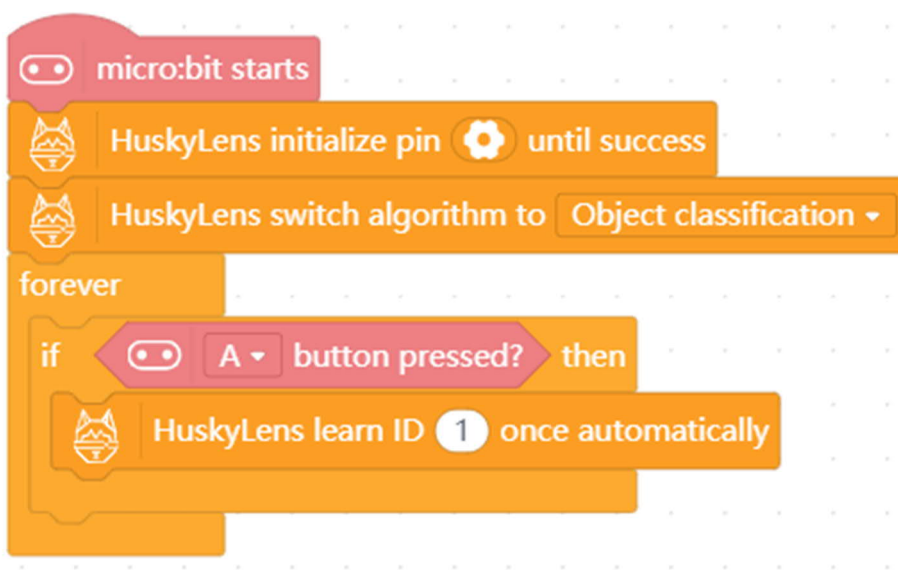
7.8.5 Make A Program to Trigger Learning Function

In addition to manually pressing the "learn button" to learn the target object, it can also be triggered programmatically. This function enables micro: Bit, Arduino, and other mainboards to automatically control the learning of HUSKYLENS.

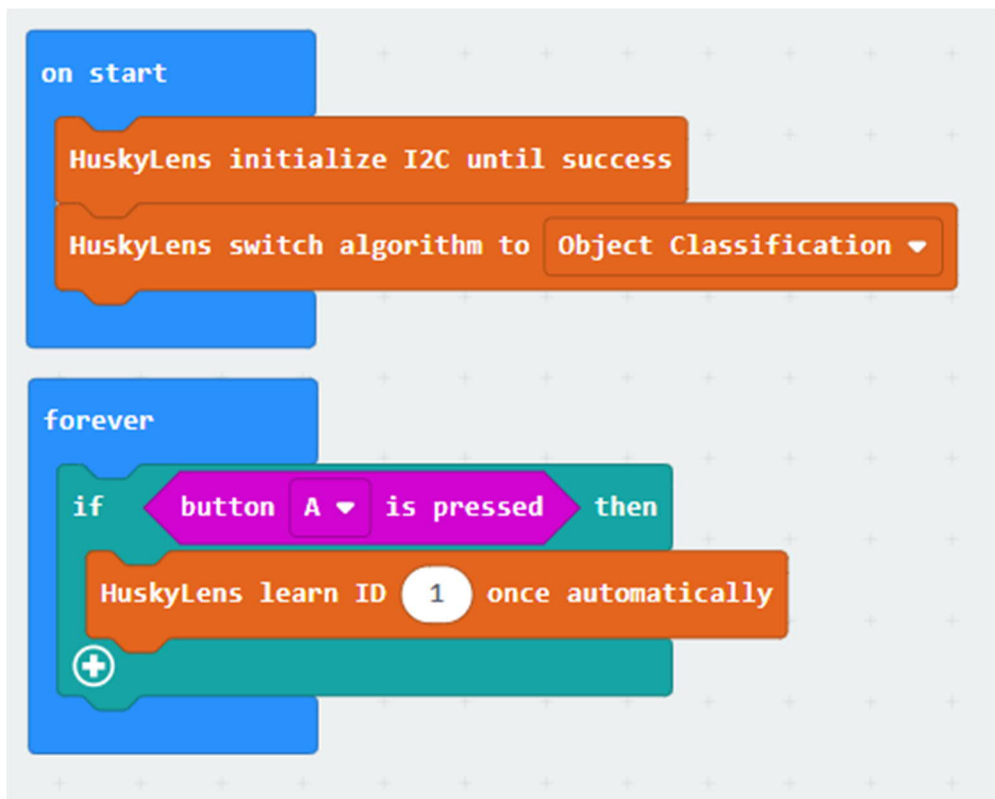
All 7 algorithms of HUSKYLENS support this function.

Demonstration: Press button A on the mainboard of micro:bit or pull down the Arduino Pin A0 to let HUSKYLENS learn the object once. The object's ID is 1.

Mind+ Sample Program:



MakeCode Sample Program:



Arduino Sample Program:

```
#include "HUSKYLENS.h"

HUSKYLENS huskyLens;



void setup()
{
    Serial.begin(115200);
    pinMode(A0, INPUT_PULLUP);
    Wire.begin();
    while (!huskyLens.begin(Wire))
    {
        Serial.println(F("Begin failed!"));
        delay(100);
    }
}

void loop()
{
    if(digitalRead(A0) == 0)
    {
        while (!huskyLens.writeLearn(1)) // bool writeLearn(int ID)
        {
            Serial.println(F("Learn object ID1 failed!"));
            delay(100);
        }
        Serial.println(F("Learn object ID1 success"));
    }
}
```

```
}  
}
```

Result

In object classification mode, aim HUSKYLENS at the following three images in turn. Press button A of micro:bit or pull down the Arduino Pin A0, then HUSKYLENS will learn these three images as objects:ID1 in turn. When HUSKYLENS recognizes any of these three images again, it displays object:ID1 on the screen, indicating that the three workers are wearing safety hats.

1	2	3
		

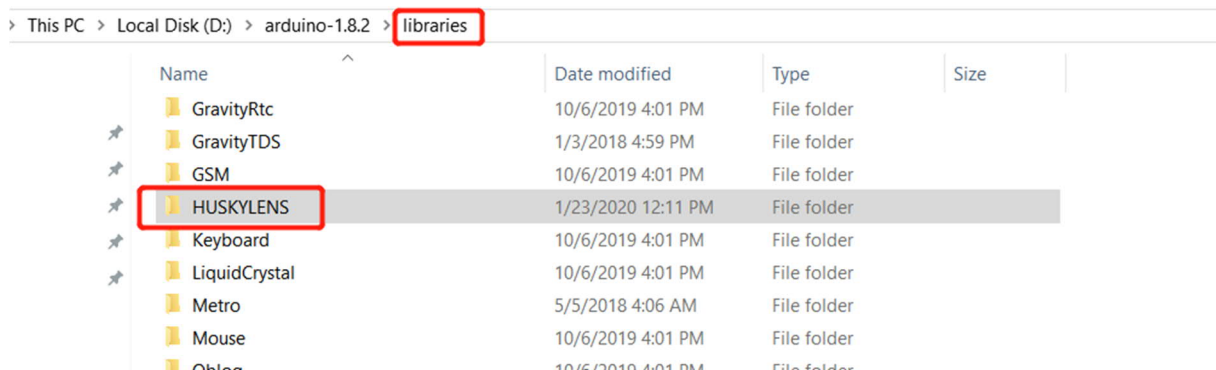
Here you can customize the name of ID1, such as safe:ID1; You can also customize the text on the screen, overlaying relevant information to indicate that the status is with safety hat. With the information, the recognition results are easy to understand.

8. Arduino Tutorial

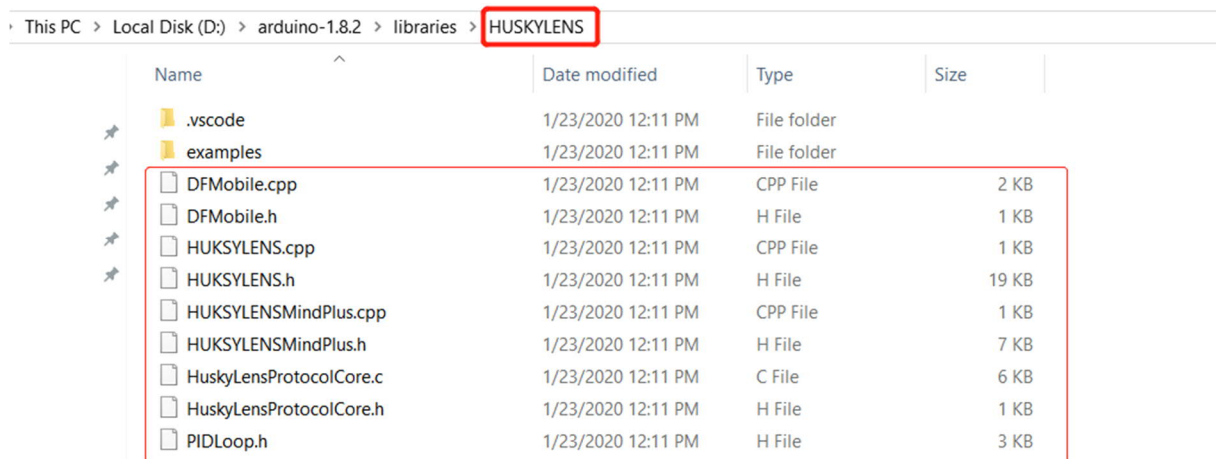
Please download and install the [HUSKYLENS Library](#) first.

8.1 Install the Library

1. Unzip the file, then copy the folder to the "libraries" folder of the Arduino IDE. Then check whether the folder name is "HUSKYLENS". If not, please change it as "HUSKYLENS". Please note that the library file name must be HUSKYLENS.



2. All .h files and .cpp files must in the root directory of the "HUSKYLENS" folder.



Name	Date modified	Type	Size
.vscode	1/23/2020 12:11 PM	File folder	
examples	1/23/2020 12:11 PM	File folder	
DFMobile.cpp	1/23/2020 12:11 PM	CPP File	2 KB
DFMobile.h	1/23/2020 12:11 PM	H File	1 KB
HUSKYLENS.cpp	1/23/2020 12:11 PM	CPP File	1 KB
HUSKYLENS.h	1/23/2020 12:11 PM	H File	19 KB
HUSKYLENSMindPlus.cpp	1/23/2020 12:11 PM	CPP File	1 KB
HUSKYLENSMindPlus.h	1/23/2020 12:11 PM	H File	7 KB
HuskyLensProtocolCore.c	1/23/2020 12:11 PM	C File	6 KB
HuskyLensProtocolCore.h	1/23/2020 12:11 PM	H File	1 KB
PIDLoop.h	1/23/2020 12:11 PM	H File	3 KB

8.2 Project 1: Read Position Data

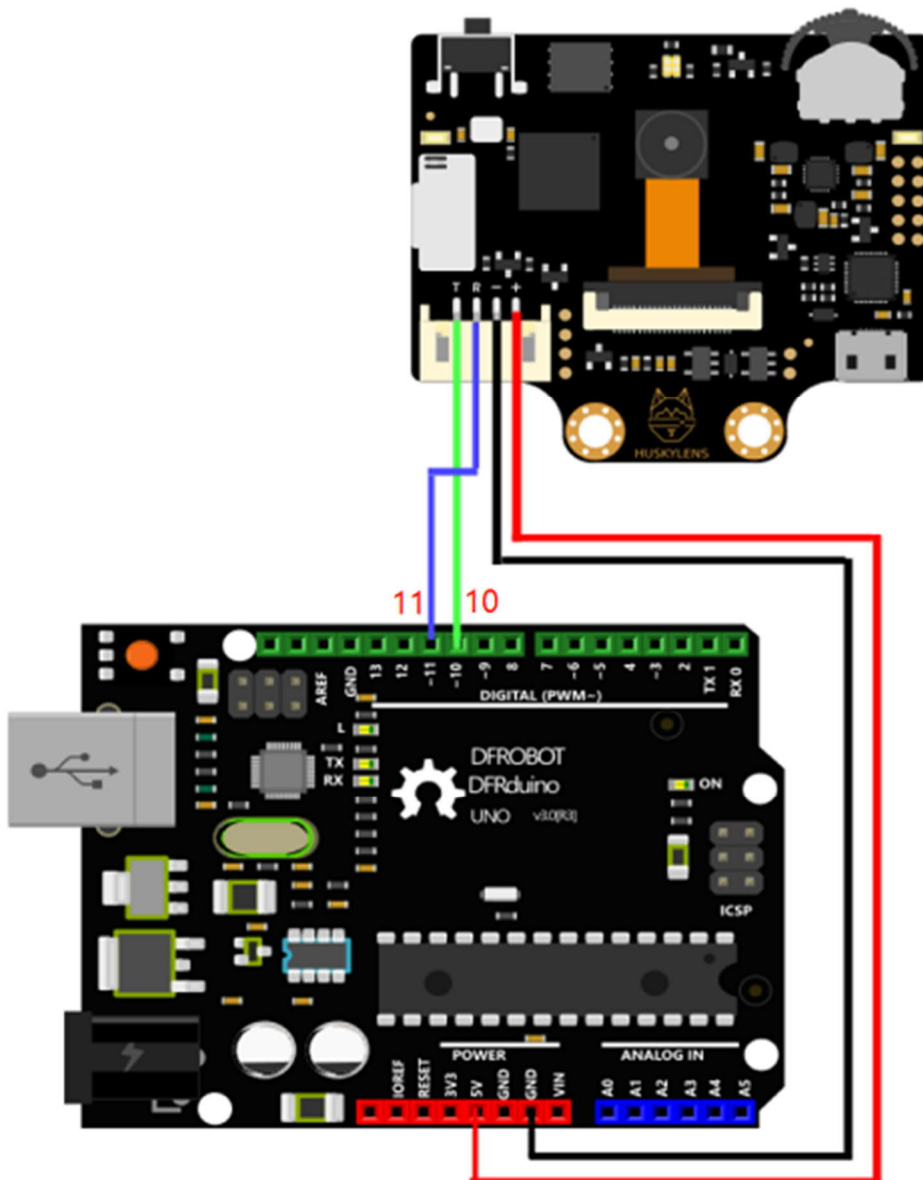
In this project, HuskyLens will be connected to Arduino mainboard. And Arduino Uno will read position data of the object from HuskyLens. Then the serial port monitor will print the data. So that, you can read the position of the object in real time.

Requirements

- **Hardware**
 - [DFRduino UNO R3](#) (or similar) x 1
 - [HUSKYLENS](#) x 1
 - MM/FM/FF Jumper wires
- **Software**
 - [Arduino IDE](#) (version 1.8.x is recommended)
 - Download and install the **HUSKYLENS Library** ([About how to install the library?](#))

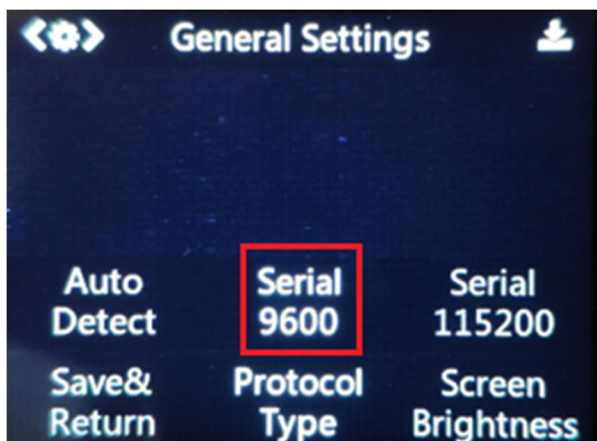
UART Mode(SoftwareSerial)

Connection Diagram



HuskyLens Protocol Setting

You need to set the protocol type of HuskyLens. The protocol should be 'Serial 9600'. Of course, you can adopt the Auto Detect protocol, which is easy-to-use and convenient.



Sample Code

```
#include "HUSKYLENS.h"
#include "SoftwareSerial.h"

HUSKYLENS huskylens;
SoftwareSerial mySerial(10, 11); // RX, TX
//HUSKYLENS green line >> Pin 10; blue line >> Pin 11
void printResult(HUSKYLENSResult result);

void setup() {
    Serial.begin(115200);
    mySerial.begin(9600);
    while (!huskylens.begin(mySerial))
    {
        Serial.println(F("Begin failed!"));
        Serial.println(F("1.Please recheck the \"Protocol Type\" in HUSKYLENS (General Settings>>Protocol Type>>Serial 9600)"));
        Serial.println(F("2.Please recheck the connection."));
        delay(100);
    }
}

void loop() {
    if (!huskylens.request()) Serial.println(F("Fail to request data from HUSKYLENS, recheck the connection!"));
    else if (!huskylens.isLearned()) Serial.println(F("Nothing learned, press learn button on HUSKYLENS to learn one!"));
    else if (!huskylens.available()) Serial.println(F("No block or arrow appears on the screen!"));
    else
    {
        Serial.println(F("#####"));
        while (huskylens.available())
        {
            HUSKYLENSResult result = huskylens.read();
            printResult(result);
        }
    }
}

void printResult(HUSKYLENSResult result){
    if (result.command == COMMAND_RETURN_BLOCK){

        Serial.println(String()+F("Block: xCenter=")+result.xCenter+F(", yCenter=")+result.yCenter+F(", width=")+result.width+F(", height=")+result.height+F(", ID=")+result.ID);
    }
    else if (result.command == COMMAND_RETURN_ARROW){

        Serial.println(String()+F("Arrow: xOrigin=")+result.xOrigin+F(", yOrigin=")+result.yOrigin+F(", xTarget=")+result.xTarget+F(", yTarget=")+result.yTarget+F(", ID=")+result.ID);
    }
    else{
        Serial.println("Object unknown!");
    }
}
```


Copy

Operations and Expected Results

1. Upload the above codes to your Arduino board.
2. Let your HuskyLens learn a new thing first. You can refer to the previous chapters of this tutorial.
3. Open the serial monitor of Arduino IDE, then you will get the position data of the object.

If HuskyLens is in the face recognition, object tracking, object recognition, color recognition, tag recognition mode, you will get the results like follows:

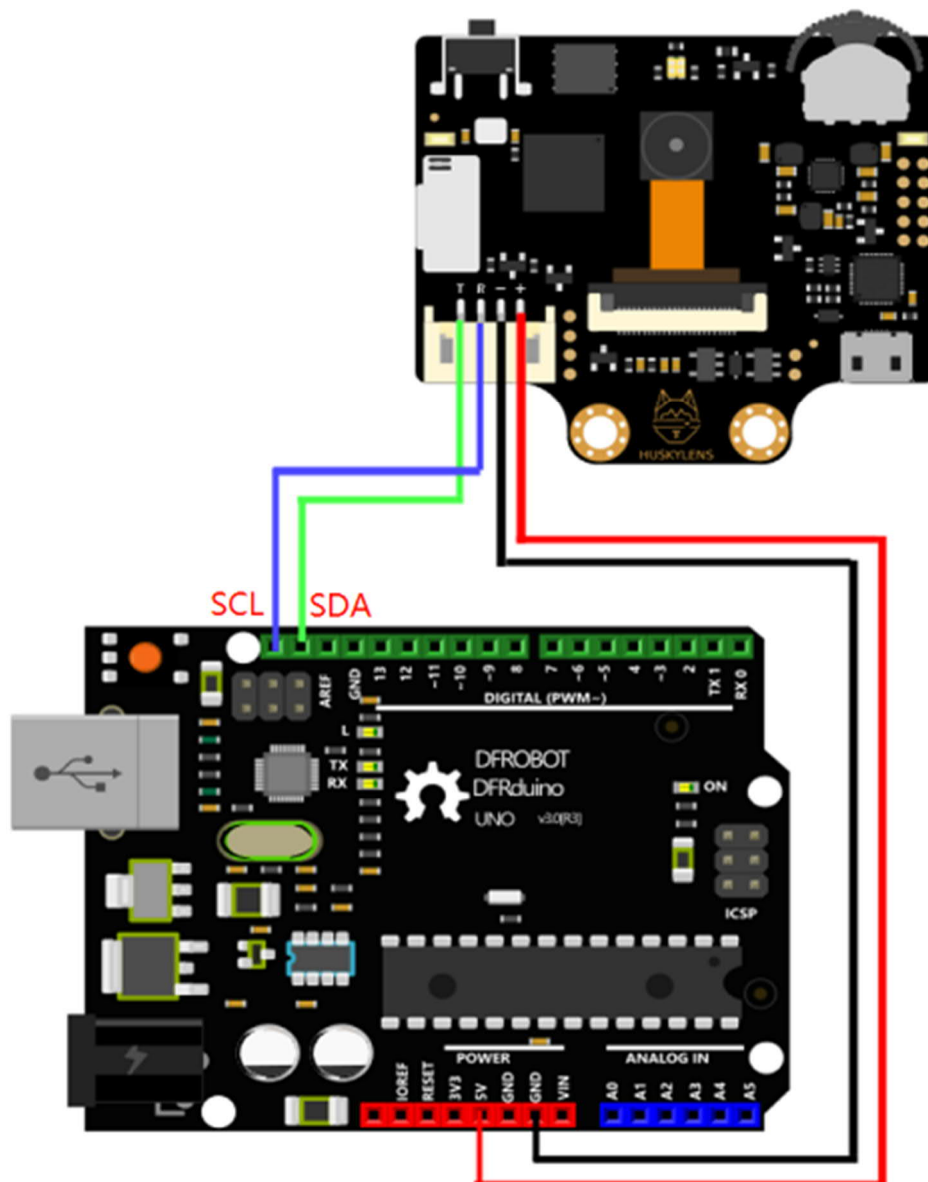
```
COM4
Block:xCenter=162,yCenter=135,width=138,height=146
Block:xCenter=163,yCenter=134,width=138,height=146
Block:xCenter=163,yCenter=134,width=138,height=146
Block:xCenter=162,yCenter=137,width=108,height=145
Block:xCenter=162,yCenter=135,width=108,height=146
Block:xCenter=162,yCenter=135,width=108,height=146
Block:xCenter=163,yCenter=135,width=138,height=146
Block:xCenter=163,yCenter=136,width=109,height=146
Block:xCenter=163,yCenter=136,width=109,height=146
Block:xCenter=162,yCenter=137,width=138,height=145
Block:xCenter=162,yCenter=136,width=138,height=146
Block:xCenter=162,yCenter=136,width=138,height=146
Block:xCenter=162,yCenter=137,width=108,height=145
Block:xCenter=162,yCenter=137,width=108,height=145
Block:xCenter=163,yCenter=135,width=109,height=146
Block:xCenter=163,yCenter=137,width=109,height=145
```

If HuskyLens is in the line tracking mode, you will get the results like follows:

```
COM4
Arrow:xOrigin=304,yOrigin=158,xTarget=216,yTarget=82
Arrow:xOrigin=312,yOrigin=164,xTarget=224,yTarget=82
Arrow:xOrigin=312,yOrigin=164,xTarget=216,yTarget=86
Arrow:xOrigin=304,yOrigin=164,xTarget=216,yTarget=86
Arrow:xOrigin=304,yOrigin=166,xTarget=216,yTarget=86
Arrow:xOrigin=304,yOrigin=164,xTarget=208,yTarget=86
Arrow:xOrigin=304,yOrigin=164,xTarget=216,yTarget=84
Arrow:xOrigin=312,yOrigin=164,xTarget=216,yTarget=84
Arrow:xOrigin=312,yOrigin=164,xTarget=216,yTarget=84
Arrow:xOrigin=304,yOrigin=164,xTarget=216,yTarget=82
Arrow:xOrigin=312,yOrigin=164,xTarget=216,yTarget=84
Arrow:xOrigin=312,yOrigin=162,xTarget=216,yTarget=82
Arrow:xOrigin=312,yOrigin=162,xTarget=216,yTarget=82
Arrow:xOrigin=304,yOrigin=162,xTarget=216,yTarget=82
Arrow:xOrigin=312,yOrigin=162,xTarget=216,yTarget=80
Arrow:xOrigin=312,yOrigin=164,xTarget=216,yTarget=82
```

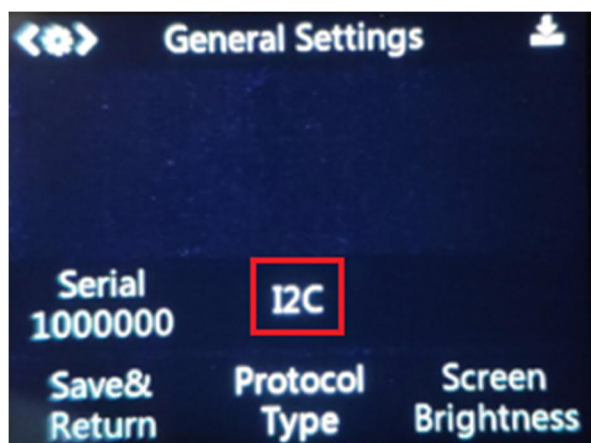
I2C Mode

Connection Diagram



HuskyLens Protocol Setting

You need to set the protocol type of HuskyLens. The protocol should be 'I2C'. Of course, you can adopt the auto detect protocol, which is easy-to-use and convenient.



Sample Code

```
#include "HUSKYLENS.h"
#include "SoftwareSerial.h"

HUSKYLENS huskylens;
//HUSKYLENS green line >> SDA; blue line >> SCL
void printResult(HUSKYLENSResult result);

void setup() {
    Serial.begin(115200);
    Wire.begin();
    while (!huskylens.begin(Wire))
    {
        Serial.println(F("Begin failed!"));
        Serial.println(F("1. Please recheck the \"Protocol Type\" in HUSKYLENS (General Settings>>Protocol Type>>I2C)"));
        Serial.println(F("2. Please recheck the connection."));
        delay(100);
    }
}

void loop() {
    if (!huskylens.request()) Serial.println(F("Fail to request data from HUSKYLENS, recheck the connection!"));
    else if (!huskylens.isLearned()) Serial.println(F("Nothing learned, press learn button on HUSKYLENS to learn one!"));
    else if (!huskylens.available()) Serial.println(F("No block or arrow appears on the screen!"));
    else
    {
        Serial.println(F("#####"));
        while (huskylens.available())
        {
            HUSKYLENSResult result = huskylens.read();
            printResult(result);
        }
    }
}

void printResult(HUSKYLENSResult result){
    if (result.command == COMMAND_RETURN_BLOCK){

        Serial.println(String()+F("Block: xCenter=")+result.xCenter+F(", yCenter=")+result.yCenter+F(", width=")+result.width+F(", height=")+result.height+F(", ID=")+result.ID);
    }
    else if (result.command == COMMAND_RETURN_ARROW){

        Serial.println(String()+F("Arrow: xOrigin=")+result.xOrigin+F(", yOrigin=")+result.yOrigin+F(", xTarget=")+result.xTarget+F(", yTarget=")+result.yTarget+F(", ID=")+result.ID);
    }
    else{
        Serial.println("Object unknown!");
    }
}
```

Copy

Operations and Expected Results

1. Upload the above codes to your Arduino board.
2. Let your HuskyLens learn a new thing first. You can refer to the previous chapters of this tutorial.
3. Open the serial monitor of Arduino IDE, then you will get the position data of the object, same as the results in UART mode. Please refer to the previous chapter, which will not be repeated here.

8.3 Project 2: DIY Line Tracking Robot

In this part, we will use the Devastator Tank Mobile Robot, HuskyLens and Romeo board to make a line tracking robot.

Please refer [this post](#) to get the demonstration.

8.4 Arduino API Introduction

Please [click here](#) to view the API introduction documents.

10. micro:bit Tutorial

In this chapter, we will use micro:bit board to read data from the HuskyLens. The communication protocol is I2C. We adopt Mind+ and MakeCode, then combine with several project cases to demonstrate the usage.

Now we adopt Mind+ for demonstration first. Later we adopt MakeCode to demonstrate.

10.1 Mind+ Introduction

Mind+ is a Scratch 3.0-based programming tool, which allows you to build a program by dragging and snapping coding blocks. With tons of tutorials, sample projects and a large community, it is one of the best tools for you to learn programming from absolutely zero!

Mind+ supports a wide range of hardware including Arduino, [micro:bit](#) or even a series of ESP32-based educational microcontrollers.

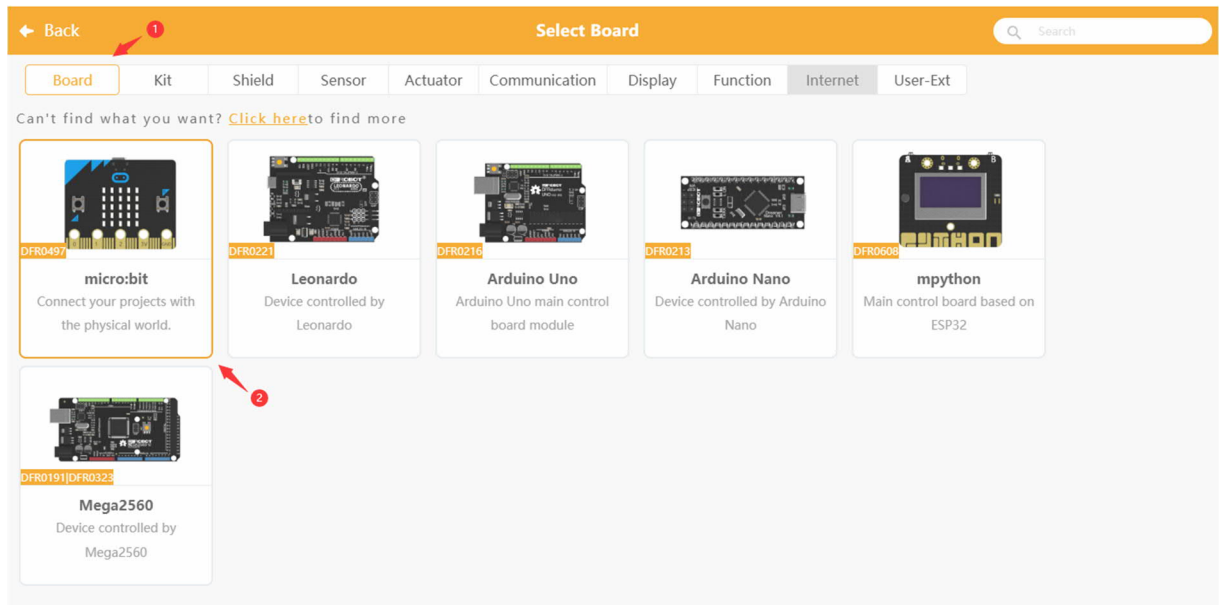
Please view Mind+ official site <http://mindplus.cc> to download the latest Mind+.

10.2 Load HuskyLens Extension

1. In the upper right corner of the Mind+ window, dial the switch to the offline mode.

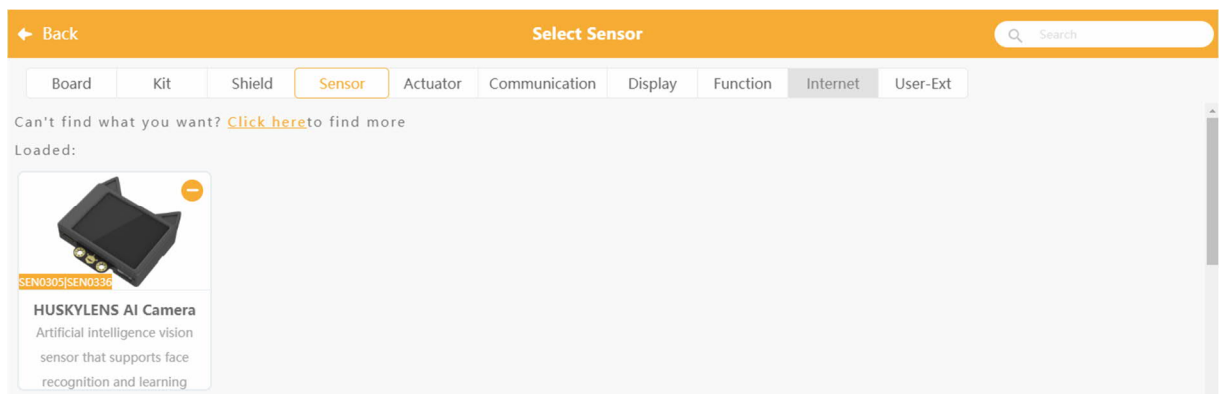


2. Click the **Extensions** button in the lower left corner to view the extensions window.
3. Select a main control board. Here we select the micro:bit.

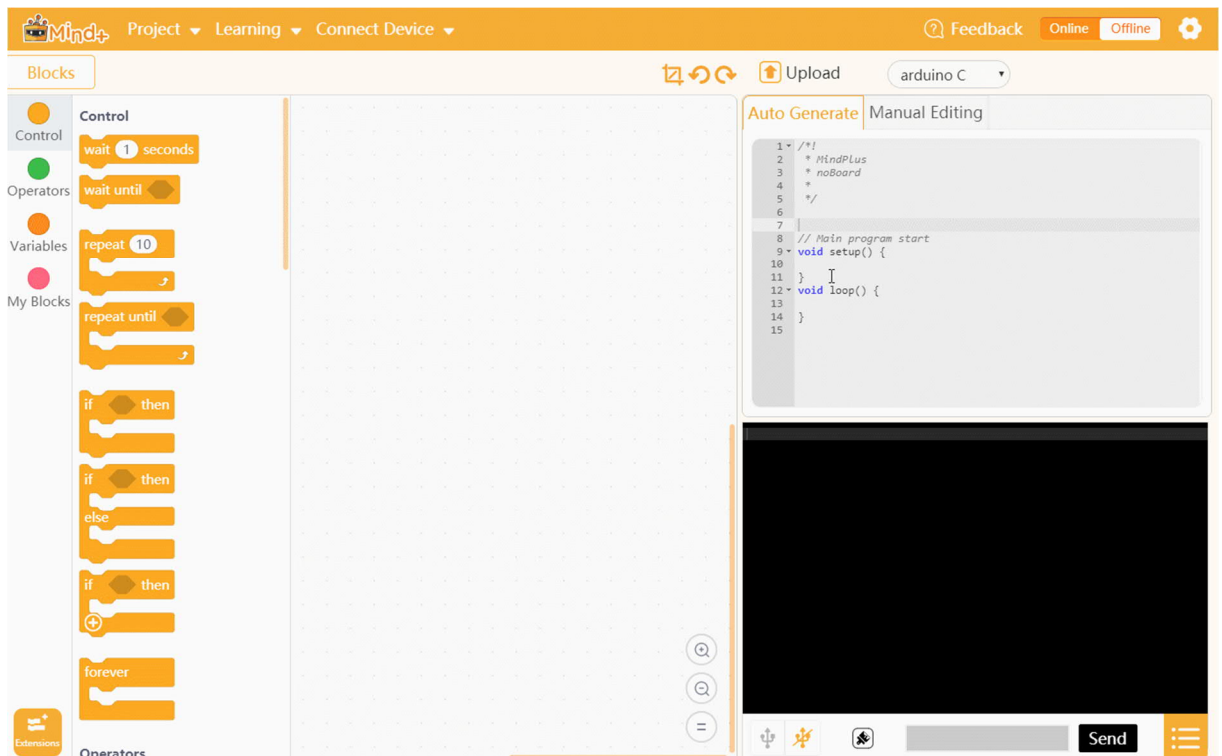


4. After selecting the main control board, the corresponding extension labels become selectable. Then click **Sensor** label, find HuskyLens. Click it to load.

You can enter the sensor's name in the search bar in upper right corner when there are many sensors listed in the menu.



5. After selecting the main control board and extensions, click **Back** button in the upper left corner to return to the programming window.



10.3 Mind+ Project 1: Face Recognition

This chapter demonstrates how to connect HuskyLens to the micro: bit board, then the micro: bit board reads the face recognition results from HuskyLens. If HuskyLens recognizes you (the learned face), the dot-matrix screen of the micro: bit displays a smiling face, otherwise it displays a crying face.

The communication protocol between HuskyLens and micro: bit is I2C.

Requirements

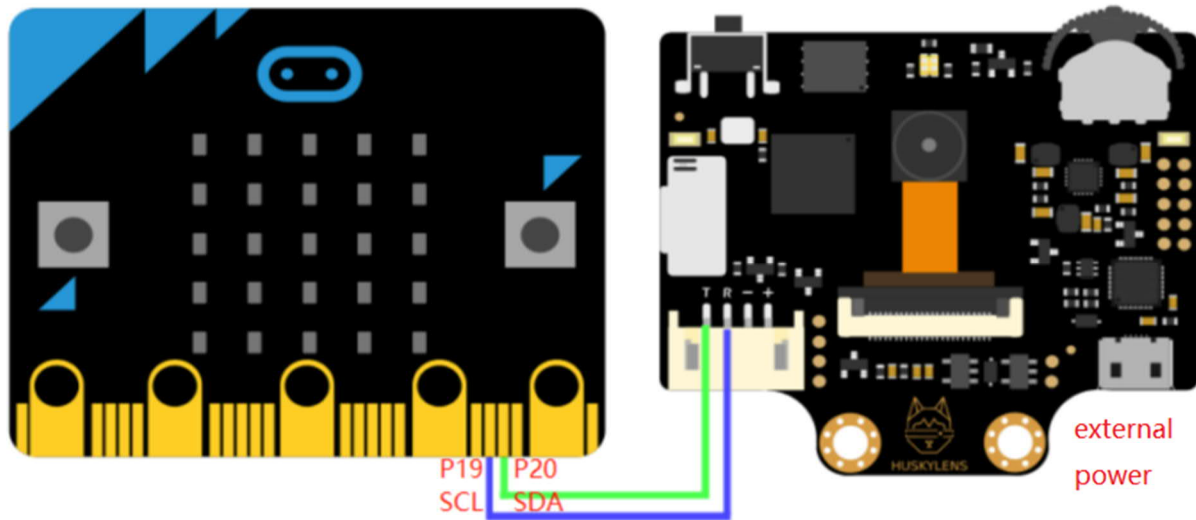
- Hardware
 - [micro:bit board](#) x 1
 - [micro:bit expansion board]([https://www.dfrobot.com/search-micro:bit expansion.html](https://www.dfrobot.com/search-micro:bit%20expansion.html)) x 1
 - [HUSKYLENS](#) x 1
 - MM/FM/FF Jumper wires
- Software
 - [Mind+](#)
 - HUSKYLENS Extension: Mind+ built-in

Connection Diagram

The following picture is only for reference when wiring. The R and T pins of HuskyLens (their functions are SCL and SDA here) are connected to the SCL (P19) and SDA (P20) pins

of the micro: bit respectively. The communication protocol between HuskyLens and micro: bit is I2C.

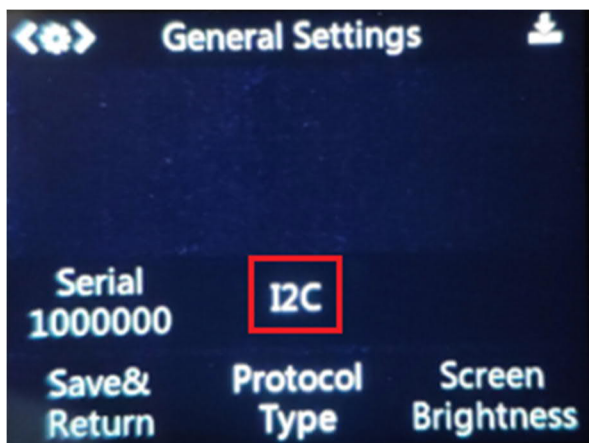
A micro: bit expansion board is recommended for simplify wiring.



Tips: HuskyLens consumes heavy current, up to 3.3V 320mA or more. The micro: bit board is not enough to supply power. Therefore, external power supply is required. You can connect the external power supply to the external power connector of the micro : bit expansion board or the USB connector of HuskyLens.

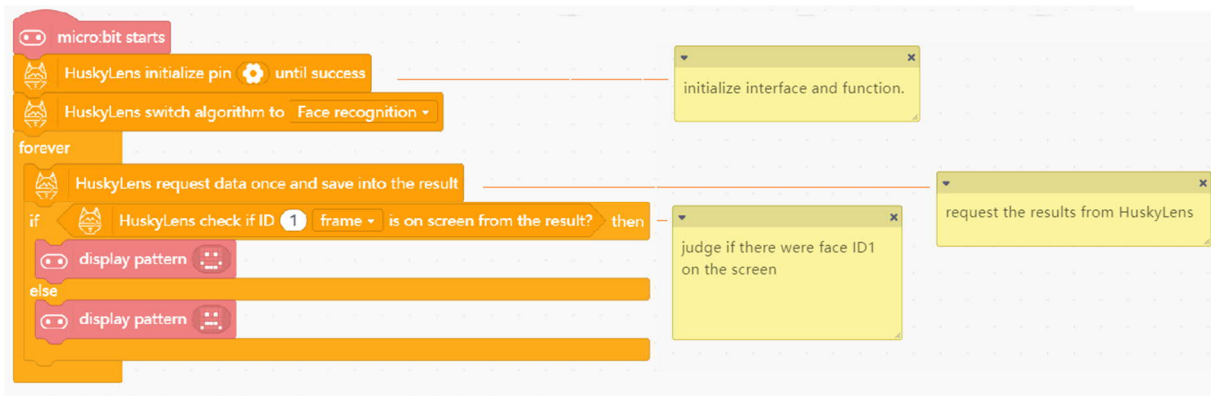
HuskyLens Protocol Setting

You need to set the protocol type of HuskyLens. The protocol should be I2C. Of course, your can adopt the auto detect protocol, which is easy-to-use and convenient.



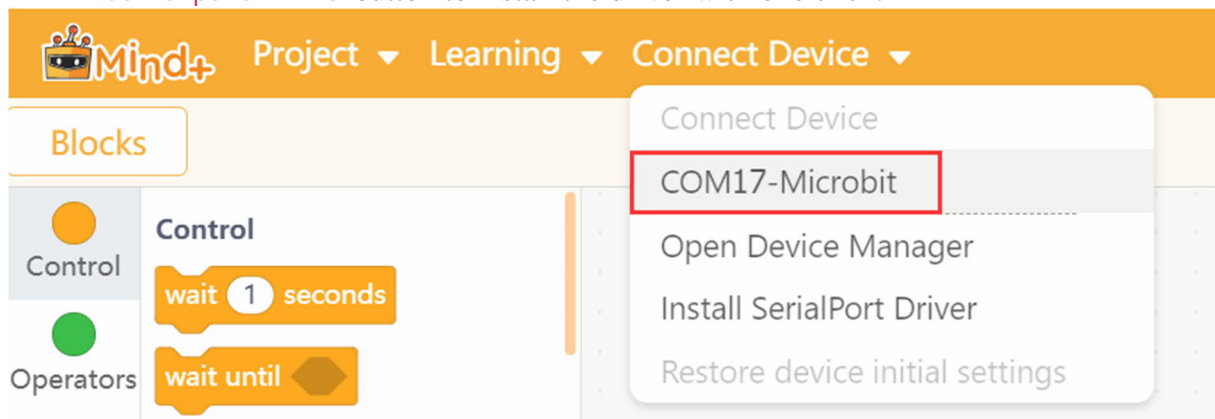
Sample Code

Drag and snap the coding blocks to program, and make a simple face recognition projects. The sample code is shown below.

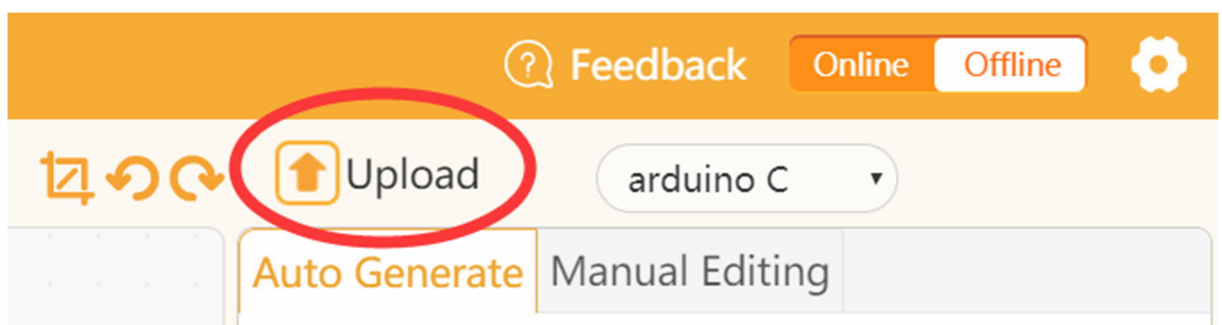


Operations and Expected Results

1. Click the **Connect Device** button, then select the COM port corresponding to the micro:bit.
If no COM port is found or you use micro:bit for the first time, please click the **Install Serial Port Driver** button to install the driver with one click.



2. Click the **Upload** button to upload the sample code to your micro:bit board.



3. Let your HuskyLens learn your face first. You can refer to the chapter 7.1 of this tutorial.
4. When HuskyLens recognizes your face, the dot-matrix screen on the micro: bit board will show a smiling face. If it were not your face, or no face appeared, it would display a crying face.

10.4 Coding Block Introduction

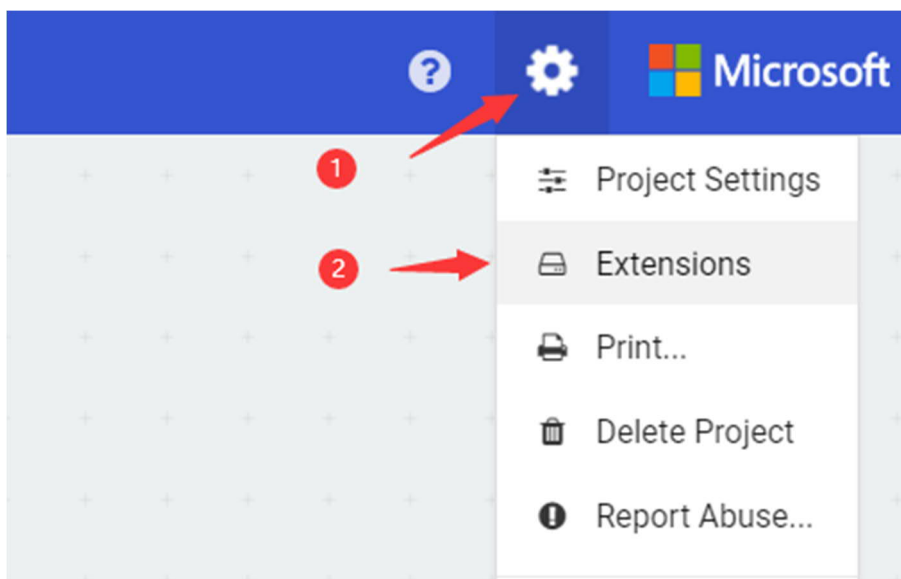
Below, we adopt MakeCode for demonstration below.

10.5 MakeCode Introduction

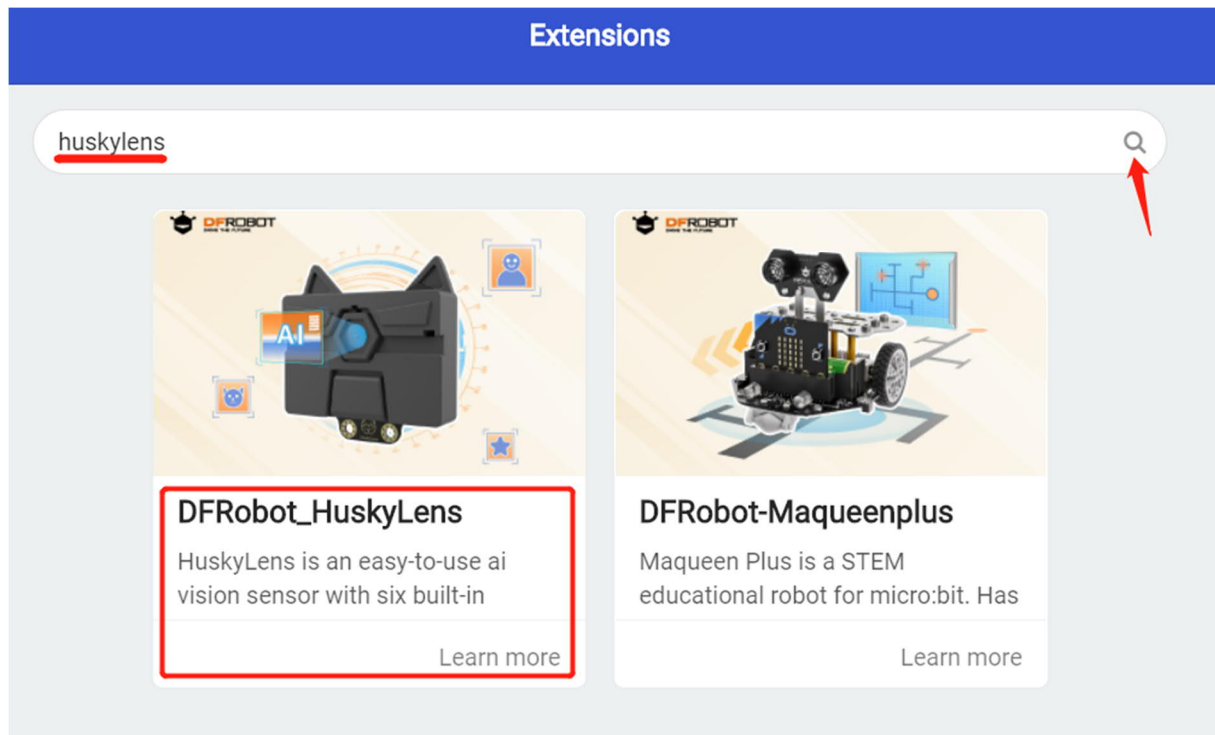
Microsoft MakeCode is a free, open source platform for creating engaging computer science learning experiences that support a progression path into real-world programming. [Click here](#) to view the MakeCode for micro:bit.

10.6 Load HuskyLens Extension

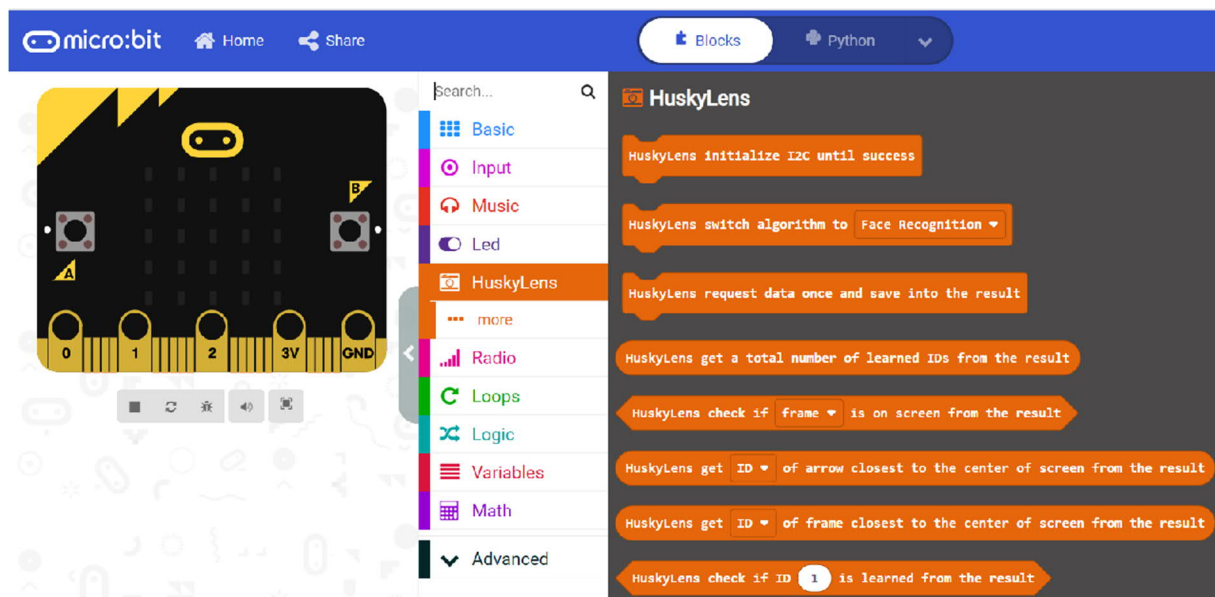
1. Create a new project in [MakeCode](#) web version, and then click the "More..." button (gear icon) at the top right and select "Extensions" from its drop-down menu to open the extension page.



2. Enter `huskylens` in the search bar, then click the search button (the magnifying glass button on the right of the search bar), you will see the HuskyLens extensions. Then click it to load the HuskyLens extension into the MakeCode.



3. In the programming page, you can see the Huskylens module.



10.7 Project 1: Face Recognition

This chapter demonstrates how to connect HuskyLens to the micro: bit board, then the micro: bit board reads the face recognition results from HuskyLens. If HuskyLens recognizes you (the learned face), the dot-matrix screen of the micro: bit displays a smiling face, otherwise it displays a crying face.

The communication protocol between HuskyLens and micro: bit is I2C.

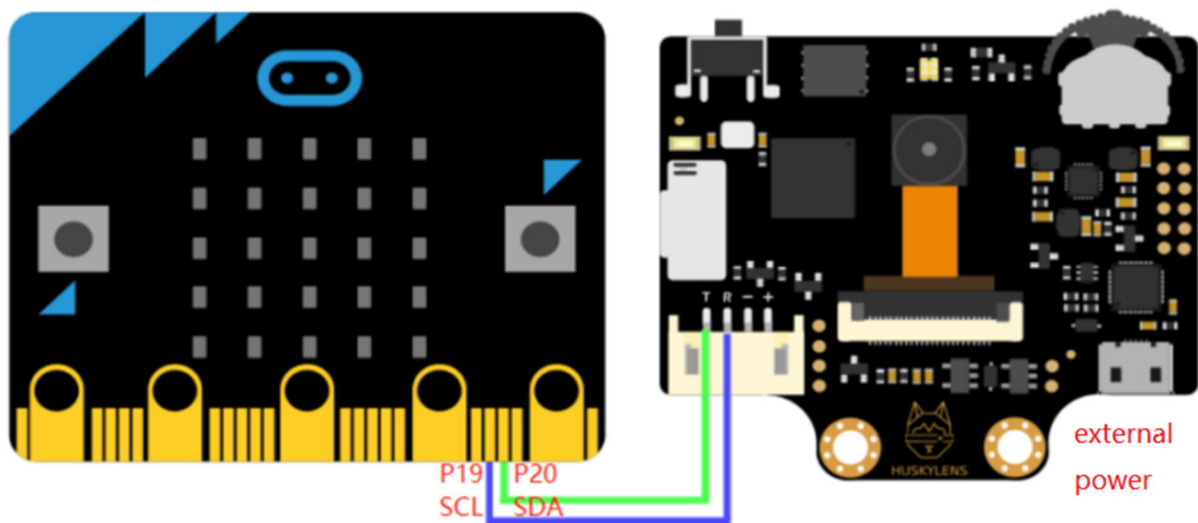
Requirements

- **Hardware**
 - [micro:bit board](#) x 1
 - [micro:bit expansion board](#) x 1
 - [HUSKYLENS](#) x 1
 - MM/FM/FF Jumper wires
- **Software**
 - [Microsoft MakeCode for micro:bit](#)
 - [HUSKYLENS MakeCode Extension](#)

Connection Diagram

The following picture is only for reference when wiring. The R and T pins of HuskyLens (their functions are SCL and SDA here) are connected to the SCL (P19) and SDA (P20) pins of the micro: bit respectively. The communication protocol between HuskyLens and micro: bit is I2C.

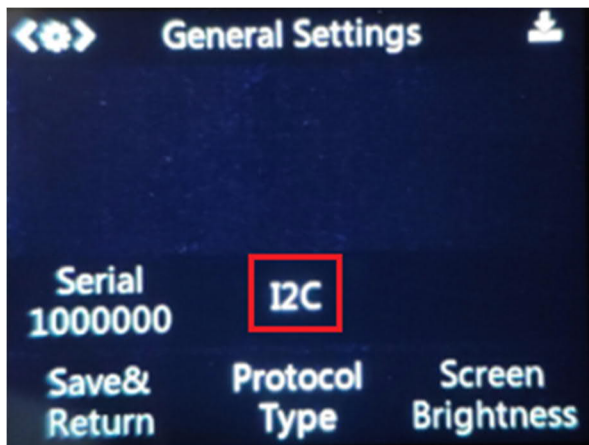
A micro: bit expansion board is recommended for simplify wiring.



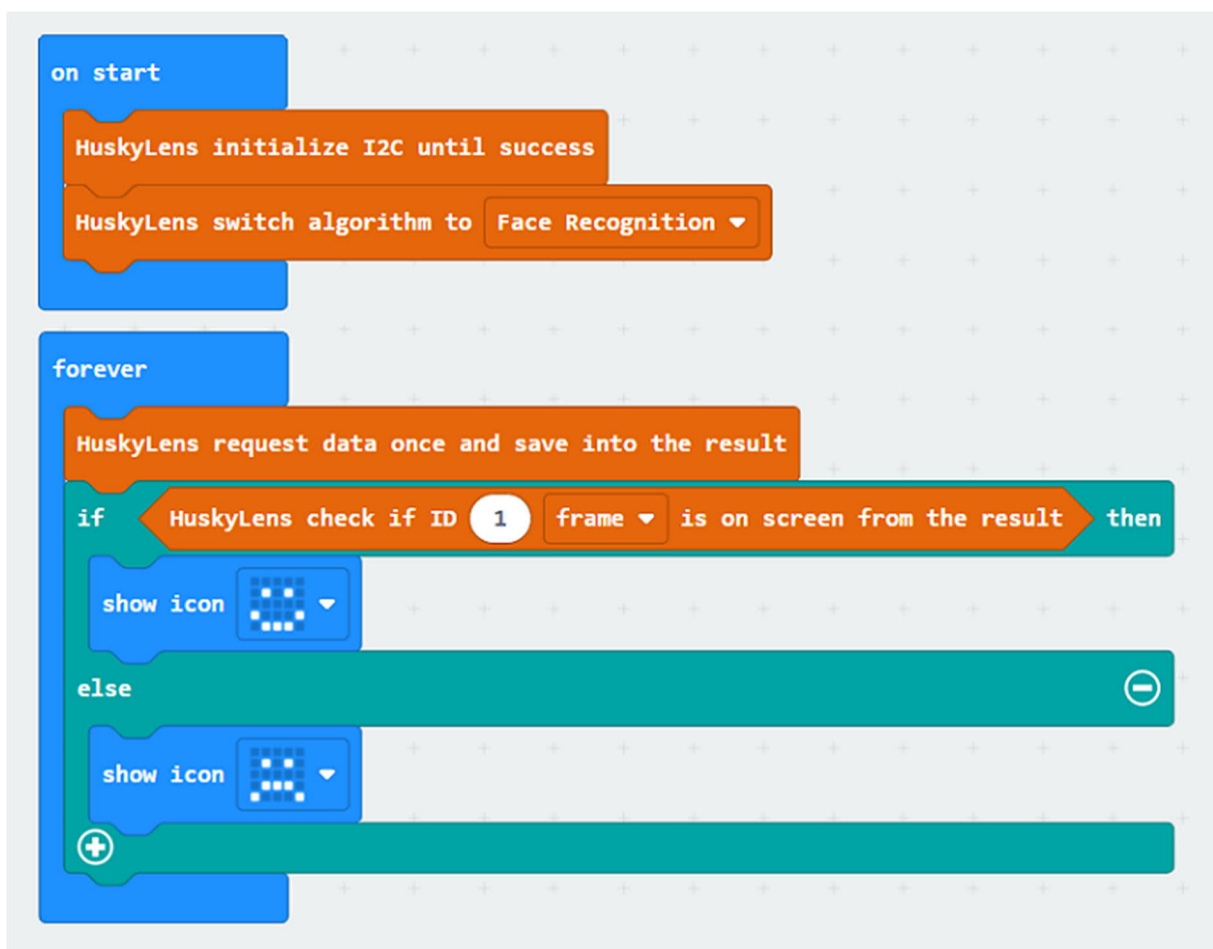
Tips: HuskyLens consumes heavy current, up to 3.3V 320mA or more. The micro: bit board is not enough to supply power. Therefore, external power supply is required. You can connect the external power supply to the external power connector of the micro : bit expansion board, or the HuskyLens USB connector.

HuskyLens Protocol Setting

You need to set the protocol type of HuskyLens. The protocol should be I2C. Of course, you can adopt the auto detect protocol, which is easy-to-use and convenient.



Sample Code



Operations and Expected Results

1. Upload the above codes to the micro: bit board.
2. Refer to the previous chapter which explaining the face recognition function(chapter 7.1), let your HuskyLens learn a face, such as your face.

3. When HuskyLens recognizes your face, the dot-matrix screen on the micro: bit board will show a smiling face. If it were not your face, or no face appeared, it would display a crying face.

10.8 Coding Block Introduction

11. FAQ

Q. When I upload the firmware by the k-flash, but I get the "end of central directory record could not be found" error message. How can I fix this?

A. The firmware you downloaded is not correct, that is, The file is incomplete and wrong. (eg The correct firmware is up to 9MB, but the firmware you download is only 66KB.) It may caused by the unstable network or other matters . Please download it from the github again. <https://github.com/HuskyLens/HUSKYLENSUploader>

12. More Documents

- Latest Firmware: [V0.5.1a](#)
- [History Version Firmware\(github\)](#)
- [Arduino Library\(github\)](#)
- [micro:bit Makecode Library\(github\)](#)
- [Protocol Document](#)
- [Tag Pictures\(More than 500\)](#)
- [Color Block Pictures](#)
- [3D model file\(.stp\)](#)
- [K-Flash Software](#)
- [Download .NET Framework 4.7.1 for K-Flash](#)
- [Download WIKI Page\(PDF\)](#)
- [Forum](#)
- [Community](#)