



# MU Vision Sensor III Programming Guide

For Mixly



V0.8.0

2019/07/03

## Table of Contents

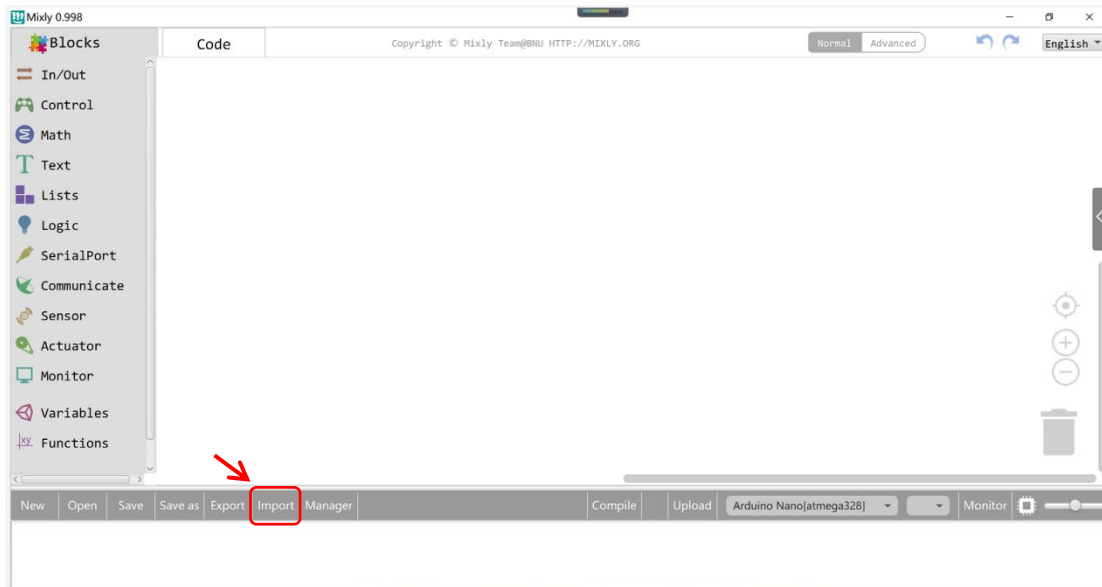
Disclaimer & Copyright.....	2
1. Import Mixly Library for MU Vision Sensor.....	3
2. Connect to Arduino.....	4
2.1 I2C Mode (recommended).....	5
2.2 Serial Mode*.....	5
3 Instructions of the Programming Blocks.....	5
3.1 Initialization.....	5
3.2 Enable Vision Algorithms.....	6
3.3 Optional Settings.....	6
3.4 Get Detection Results.....	7
Example 1: Colors Recognition.....	11
Example 2: Balls Detection.....	12
Example 3: Cards Detection.....	13
FAQ.....	14
Technical Support.....	16
Revision.....	17

## Disclaimer & Copyright

- The information in this manual applies to the MU Vision Sensor III produced by Morpx Inc. Please check the Morpx Inc's website <http://www.morpx.com> for the latest version of the firmware and library functions. Updates are subject to change without notice.
- Please read this manual carefully before using MU Vision Sensor and make sure you understand it, incorrect operation may cause the device to stop working , worse detection results, or even damaging the device.
- Morpx Inc will not warrant the damage caused by unauthorized repair or modification of electronic components on the product.
- The technical solution, vision algorithms, communication protocol mentioned in this manual is developed by Morpx and protected by intellectual property rights. No organization or individual may copy or plagiarize the technical achievements of Morpx Inc. In case of any infringement, Morpx will take legal actions to protect its rights.
- Morpx is the registered trademark of Morpx.Inc., and MU is the registered trademark of MU Vision Sensor. All trademarks (names and patterns) presented here in the text or figures belong to the holders of the marks.

# 1. Import Mixly Library for MU Vision Sensor

(1) Open Mixly, click 'Import'

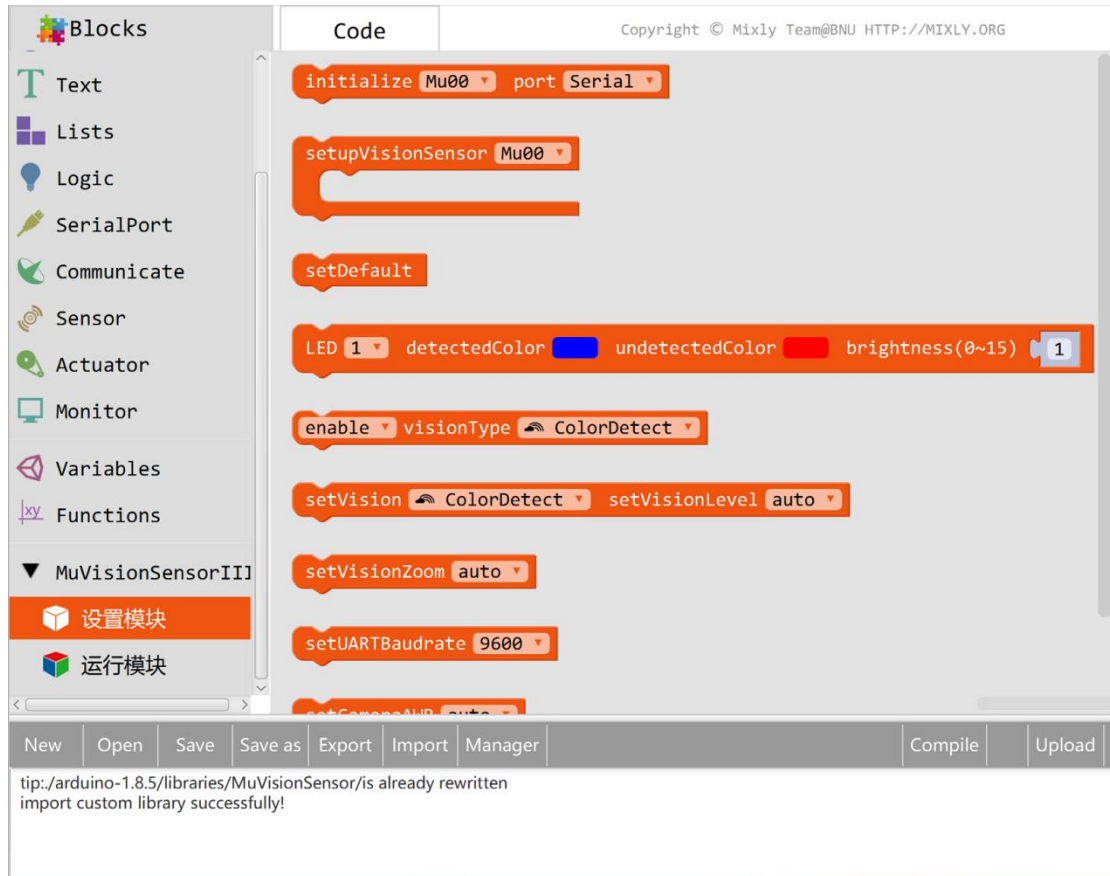


(2) Locate the 'MuVisionSensorIII.xml' file , select and open it.

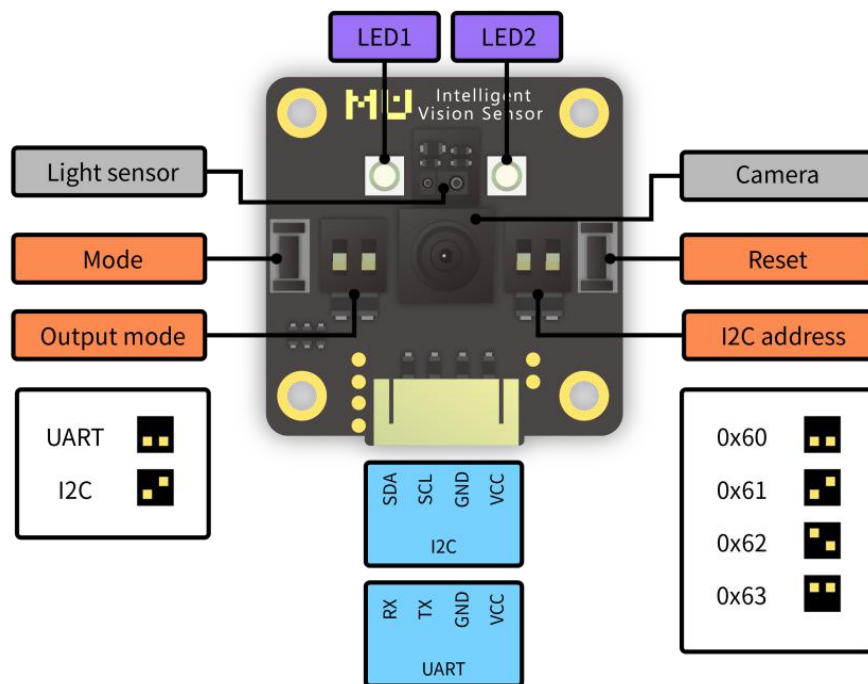
block	2019/7/2 18:23
generator	2019/7/2 18:23
language	2019/7/2 18:23
MuVisionSensor	2019/7/2 18:23
MuVisionSensorIII.xml	2019/7/2 18:23



(3) Then the nether information window prompts " import custom Library successful", and you will find 'MuVisionSensor' library from the left side 'Blocks' window.



## 2. Connect to Arduino



## 2.1 I2C Mode (recommended)

(1) Output Mode DIP Switch: set switch 1 downwards and switch 2 upwards

(2) Connect the output Pin1(SDA) to the SDA pin of Arduino, and Pin2(SCL) to SCL pin of Arduino

(3) Choose the I2C address of MU sensor by resetting Address DIP Switch. By default, both switches are downward and the address is set to 0x60. (Changing this setting is not recommended)

## 2.2 Serial Mode\*

(1) Output Mode DIP Switch: both switches downward

(2) Connect the output Pin1(RX) to TX pin of Arduino and Pin2(TX) to RX Pin of Arduino

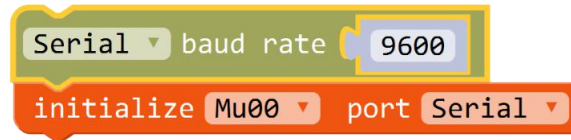
(3) Change the UART address of MU sensor by resetting Address DIP Switches. By default, both switches are downward and the address is 0x60. (Changing this setting is not recommended)

*\* Arduino cannot send messages to PC when MuVisionSensor is running in serial mode, due to a communication conflict.*

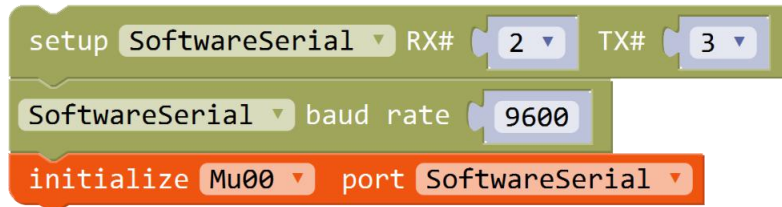
# 3 Instructions of the Programming Blocks

## 3.1 Initialization

(1) Hardware Serial Port Mode



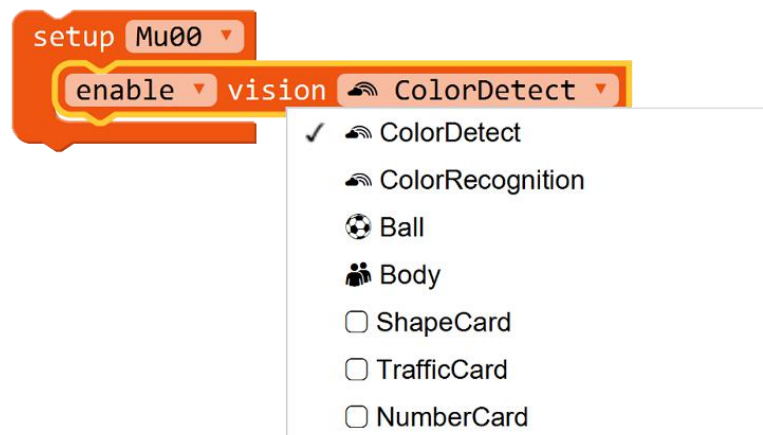
(2) Software Serial Port Mode



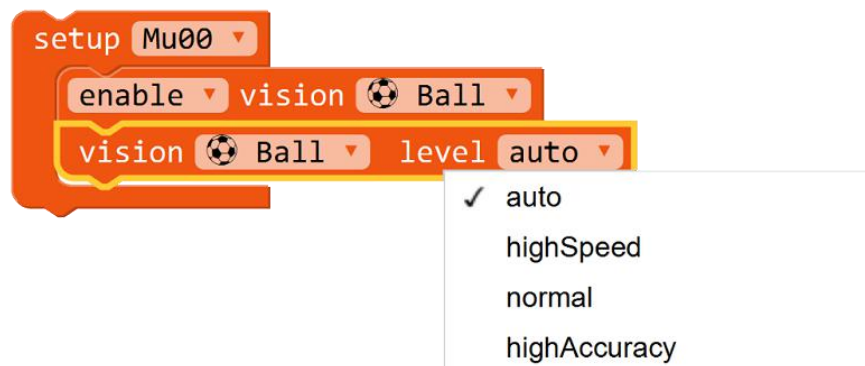
(3) Hardware I2C Mode



### 3.2 Enable Vision Algorithms



### 3.3 Optional Settings



Enable/Disable the High FPS Mode



Set White Balance Mode



Restore Default Settings

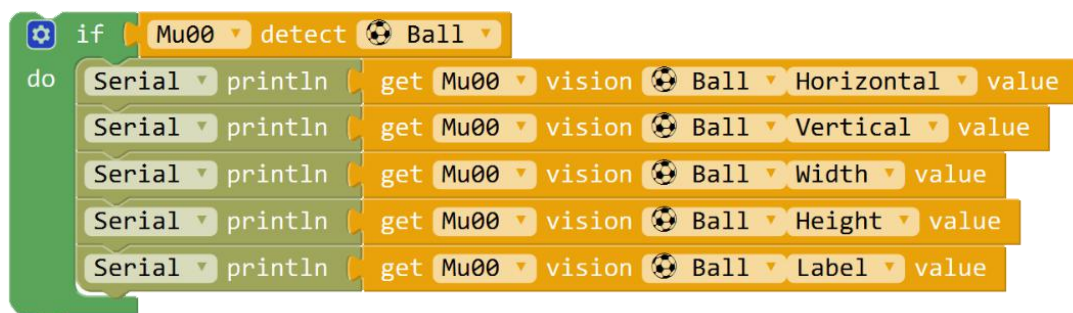


LED Settings



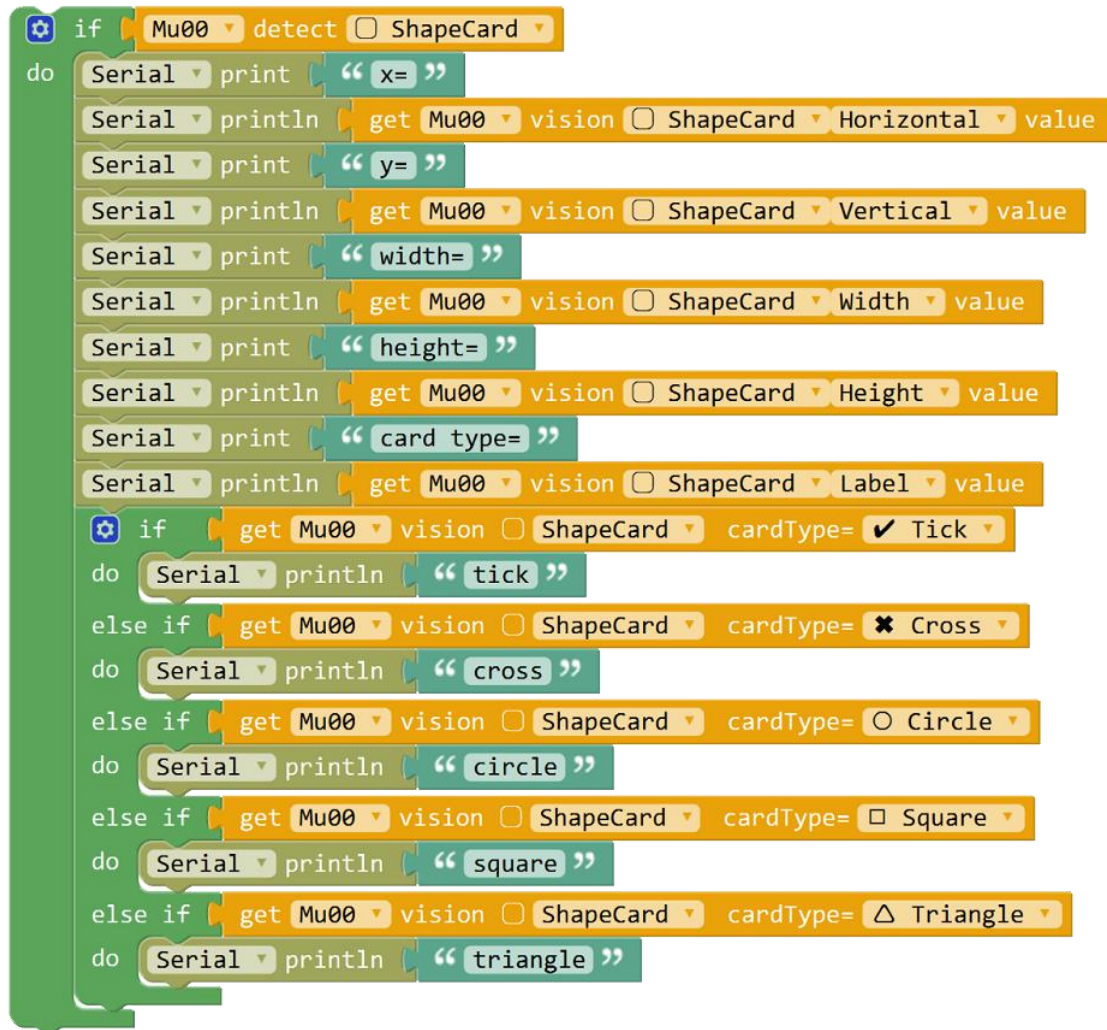
### 3.4 Get Detection Results

(1) Ball/Body Detection

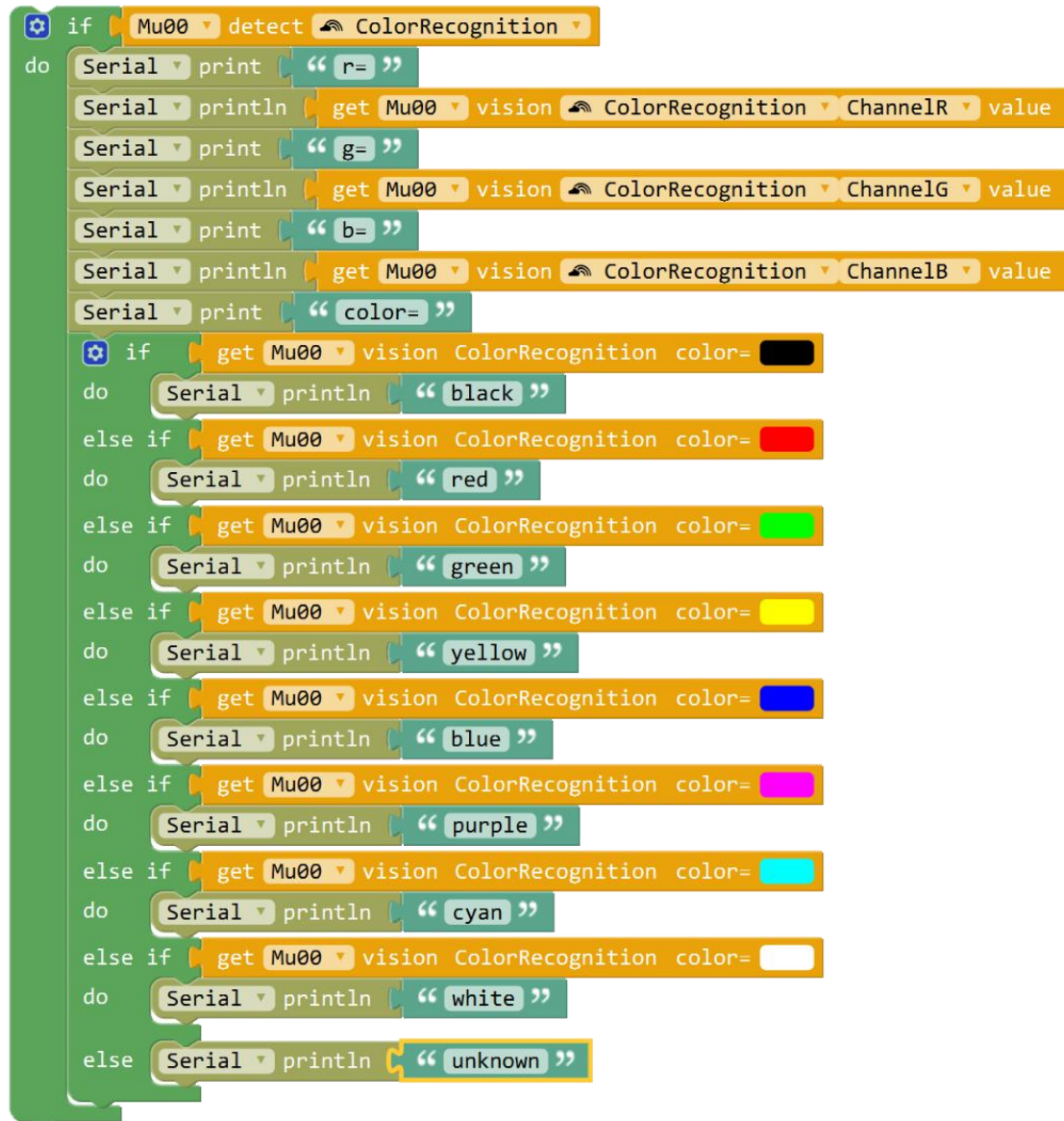




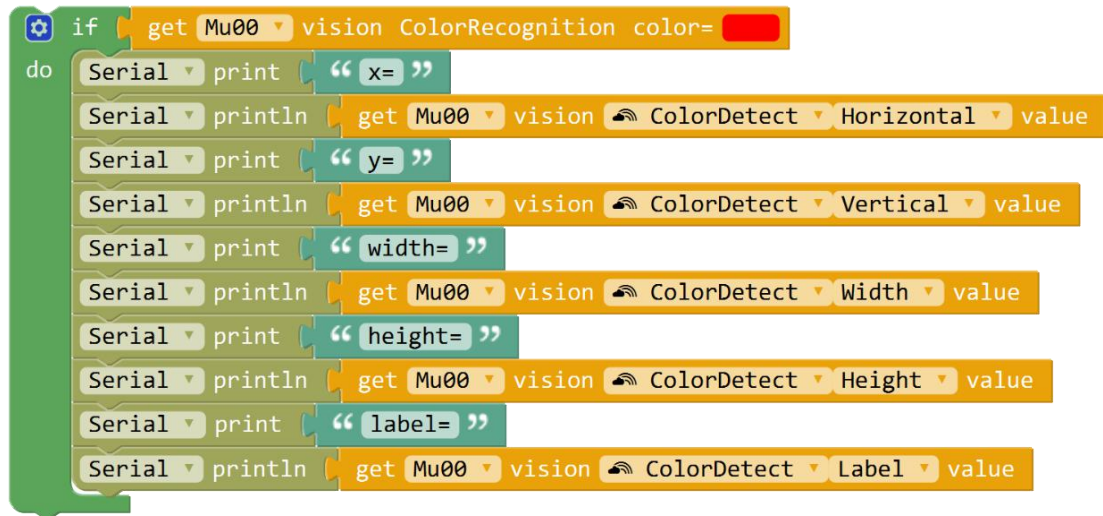
## (2) Cards Detection



### (3) Colors Recognition



#### (4) Colors Detection



## Example 1: Colors Recognition

The code is written in a Scratch-like block-based language. It starts with a 'setup' block containing several initialization steps: setting the serial baud rate to 9600, initializing the 'Mu00' module to I2C mode, enabling the 'ColorRecognition' vision algorithm, and locking the camera's white balance (AWB). Comments explain these steps: '//Initialize module to I2C mode', '//Enable color recognition algorithm', and '//Lock camera white balance to prevent image bias'. The main logic is in a 'do' loop triggered by an 'if' condition: 'if Mu00 detect ColorRecognition x= 50 y= 50'. A comment explains this: '//Recognize the color of some point of the image, in this example (50,50)'. Inside the 'do' loop, the code prints the RGB values: 'r=', 'g=', and 'b=' using 'Serial print' and 'Serial println' blocks. It then gets the 'ChannelR', 'ChannelG', and 'ChannelB' values from the 'Mu00' module's 'vision ColorRecognition' block. After printing the individual channels, it prints the full 'color=' and enters another 'if' block to identify the color. This second 'if' block has ten conditions, each with a color swatch and a corresponding text label: black, red, green, yellow, blue, purple, cyan, and white. If none of these match, it prints 'unknown'. Comments for this section read: '//Get the results of colors recognition'.

```
setup
  Serial ▾ baud rate 9600
  initialize Mu00 ▾ port I2C ▾ //Initialize module to I2C mode
  setup Mu00 ▾
    enable ▾ vision ColorRecognition ▾ //Enable color recognition algorithm
    cameraAWB LockAWB ▾ //Lock camera white balance to prevent image bias

  if Mu00 ▾ detect ColorRecognition x= 50 y= 50 //Recognize the color of some point of the image, in this example (50,50)
  do
    Serial ▾ print "r="
    Serial ▾ println get Mu00 ▾ vision ColorRecognition ▾ ChannelR ▾ value
    Serial ▾ print "g="
    Serial ▾ println get Mu00 ▾ vision ColorRecognition ▾ ChannelG ▾ value
    Serial ▾ print "b="
    Serial ▾ println get Mu00 ▾ vision ColorRecognition ▾ ChannelB ▾ value
    Serial ▾ print "color="

    if get Mu00 ▾ vision ColorRecognition color= [black] //Get the results of colors recognition
    do
      Serial ▾ println "black"
    else if get Mu00 ▾ vision ColorRecognition color= [red]
    do
      Serial ▾ println "red"
    else if get Mu00 ▾ vision ColorRecognition color= [green]
    do
      Serial ▾ println "green"
    else if get Mu00 ▾ vision ColorRecognition color= [yellow]
    do
      Serial ▾ println "yellow"
    else if get Mu00 ▾ vision ColorRecognition color= [blue]
    do
      Serial ▾ println "blue"
    else if get Mu00 ▾ vision ColorRecognition color= [purple]
    do
      Serial ▾ println "purple"
    else if get Mu00 ▾ vision ColorRecognition color= [cyan]
    do
      Serial ▾ println "cyan"
    else if get Mu00 ▾ vision ColorRecognition color= [white]
    do
      Serial ▾ println "white"
    else
      Serial ▾ println "unknown"
```



## Example 2: Balls Detection

```

setup
  Serial ▾ baud rate 9600
  initialize Mu00 ▾ port I2C ▾ //Initialize module to I2C mode
  setup Mu00 ▾
    enable ▾ vision ⚽ Ball ▾ //Enable shape ball recognition algorithm

if Mu00 ▾ detect ⚽ Ball ▾ //When the balls were detected
do
  Serial ▾ println " Ball is detected, results as follows: "
  Serial ▾ print " x= "
  Serial ▾ println get Mu00 ▾ vision ⚽ Ball ▾ Horizontal ▾ value
  Serial ▾ print " y= "
  Serial ▾ println get Mu00 ▾ vision ⚽ Ball ▾ Vertical ▾ value
  Serial ▾ print " width= "
  Serial ▾ println get Mu00 ▾ vision ⚽ Ball ▾ Width ▾ value
  Serial ▾ print " height= "
  Serial ▾ println get Mu00 ▾ vision ⚽ Ball ▾ Height ▾ value
  Serial ▾ print " label= "
  Serial ▾ println get Mu00 ▾ vision ⚽ Ball ▾ Label ▾ value
else
  Serial ▾ println " undetected "
```

### Example 3: Cards Detection

```
setup
  Serial ▾ baud rate 9600
  initialize Mu00 ▾ port I2C ▾ //Initialize module to I2C mode
  setup Mu00 ▾
    enable ▾ vision ☐ ShapeCard ▾ //Enable shape card recognition algorithm

  if Mu00 ▾ detect ☐ ShapeCard ▾ //When the shape cards were detected
  do
    Serial ▾ println "Card is detected, results as follows: "
    Serial ▾ print "x= "
    Serial ▾ println get Mu00 ▾ vision ☐ ShapeCard ▾ Horizontal ▾ value
    Serial ▾ print "y= "
    Serial ▾ println get Mu00 ▾ vision ☐ ShapeCard ▾ Vertical ▾ value
    Serial ▾ print "width= "
    Serial ▾ println get Mu00 ▾ vision ☐ ShapeCard ▾ Width ▾ value
    Serial ▾ print "height= "
    Serial ▾ println get Mu00 ▾ vision ☐ ShapeCard ▾ Height ▾ value
    Serial ▾ print "card type "
    if get Mu00 ▾ vision ☐ ShapeCard ▾ cardType= ☒ Tick ▾
    do
      Serial ▾ println "tick "
    else if get Mu00 ▾ vision ☐ ShapeCard ▾ cardType= ☒ Cross ▾
    do
      Serial ▾ println "cross "
    else if get Mu00 ▾ vision ☐ ShapeCard ▾ cardType= ☐ Circle ▾
    do
      Serial ▾ println "circle "
    else if get Mu00 ▾ vision ☐ ShapeCard ▾ cardType= ☐ Square ▾
    do
      Serial ▾ println "square "
    else if get Mu00 ▾ vision ☐ ShapeCard ▾ cardType= ☐ Triangle ▾
    do
      Serial ▾ println "triangle "
    else
      Serial ▾ println "undetected "
```

## FAQ

1. Q: What should I do if I can't open programming blocks or blocks are black boxes ?

A: Please download the latest Mixly program and import the library again.

2. Q: What should I do when I import the library and download the examples correctly, but the vision sensor doesn't respond and the serial port output nothing?

A:

- 1) Check whether the cable is correctly and tightly connected .
- 2) Check whether the white light on the back side is on. If this light is not on ,this means something wrong with power supply.
- 3) Check whether the output mode switch and address switch are correctly setted.
- 4) It takes MU sensor a period of time to initialize after power-on , we suggested to add a delay of not less than 500 ms before calling 'setup' block in your program.
- 5) After reset, the two LEDs on the front side of MU vision sensor will flicker once. Red light indicates that the current mode is serial mode, green light indicates that the current mode is I2C mode. If the color does not go with the setting of output mode switch, please reset this switch.

3. What should I do if I download the program and the serial port output correctly, but the LED does not light?

- 1) When the color recognition algorithm runs, the LED lights will be shut down in

order not to interfere with the recognition results.

2) Calling the LED setting block and alter the brightness parameter with non-zero number.

4. Why there are some functions which can be found in the datasheet while are not listed in Mixly library?

In order to make the library easy to understand and operate, some uncommon functions are removed from the Mixly library, and some parameter setting methods are simplified. If these functions are needed, please email to [support@morpx.com](mailto:support@morpx.com).

5. Why the previously downloaded recognition algorithms have impact on the current running program? For example download ball detection algorithm after running color recognition algorithm ,you will find the LEDs are still on even if the ball is not recognized, but this will not occur when only ball recognition algorithm was downloaded.

Because the previous algorithm is not terminated even if the program ends, you can add 'setDefault' block when calling 'setup' block or restart MU vision sensor.



## Technical Support

Thanks for purchasing MU Vision Sensor III, and we would like to provide continuous updating service, please check to our website: [www.morpx.com](http://www.morpx.com) regularly. Updates are subject to change without notice. You can get the latest technical information from the following websites:

GitHub: <https://github.com/mu-opensource/>

Official Website: <http://mai.morpx.com/page.php?a=sensor-support>

If there is any problem, you can contact us through telephone +86-571-81958588 or by email: [support@morpx.com](mailto:support@morpx.com).

Official Technical Support

Wechat ID



Official Technical Support

QQ ID



## Revision

Date	Version	Release Notes
2019-7-3	V0.8.0	Released