

Smart AI Cam國產IC開發套 件之應用案例設計與實作

AI骰子點數辨識

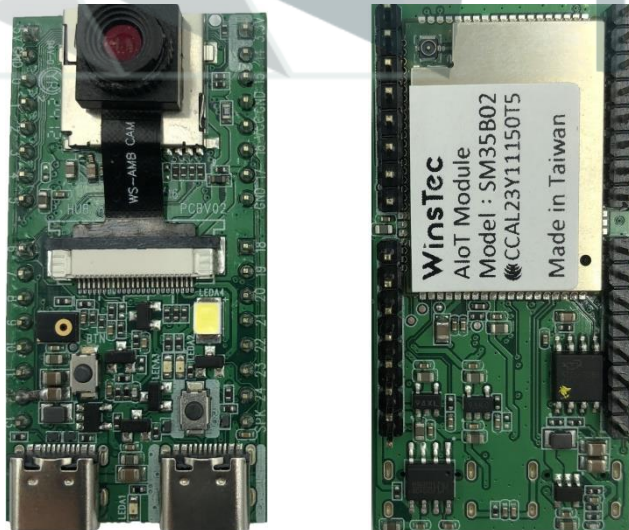


Ideas
Hatch

WinsTouch

開發套件介紹

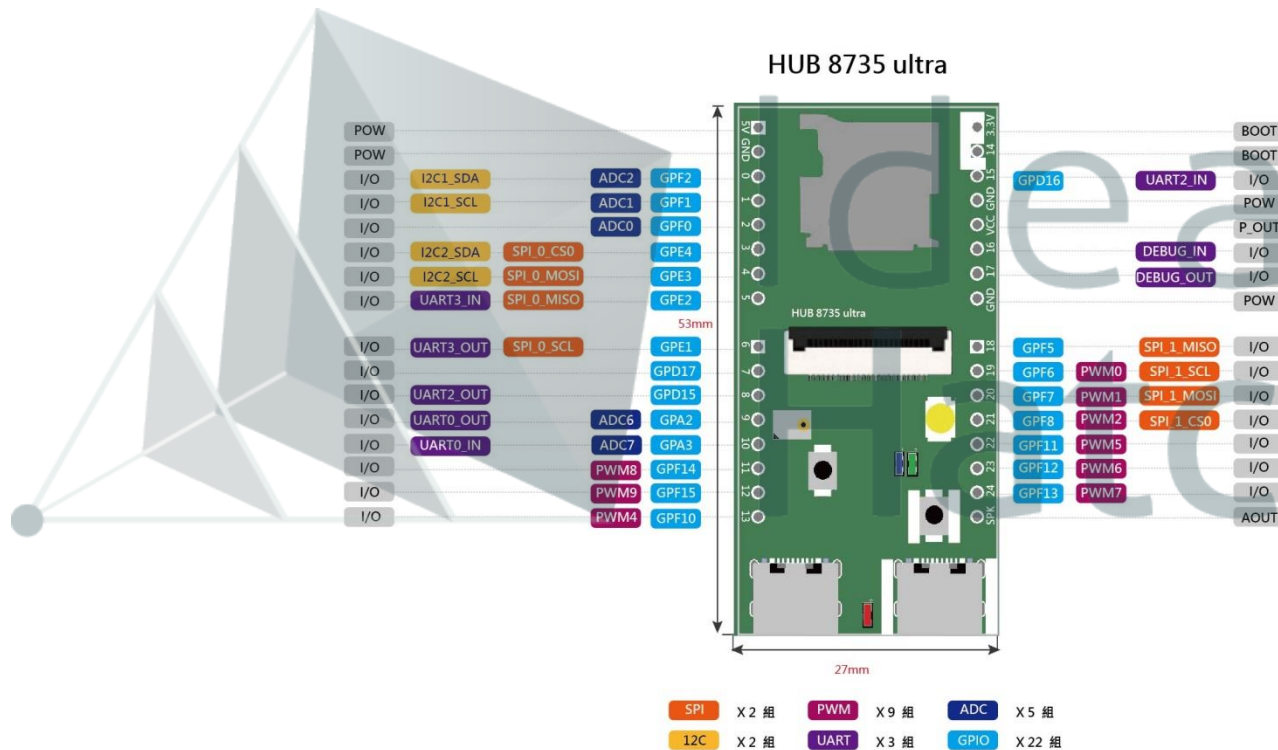
- HUB 8735 ultra
 - HUB 8735 ultra 適合開發各式的感測器或物聯網應用的開發板。除了內建多種AI 模型以及 Wi-Fi, Bluetooth外，上面的介面還有GPIO, I2C, UART, SPI, PWM, ADC。這些介面可以接一些電子元件像是LED燈、開關、壓力計、溫濕度感測器、PM2.5粉塵感測器等等。這些資料可以由內建的Wi-Fi上傳到雲端，搭配手機的App實現物聯網的實作。



WinSTouch

開發套件介紹

- HUB 8735 ultra 腳位圖



開發套件介紹

功能	描述
處理器	RTL8735B AIoT國產晶片
影像輸入	搭配國產Full HD 1080P CMOS感測器
語音輸入	內建MIC語音輸入功能、高感度數位MIC
儲存裝置	支援SD記憶卡
無線連通	Wi-Fi 2.4GHz/5GHz Bluetooth BLE
影像壓縮	H.264/265
AI處理	提供多種pre-trained AI models供快速上手 自己客製AI models
USB-C介面	USB燒錄、debug、USB OTG
LED	補光LED

開發套件及周邊

- HUB 8735 ultra * 1 片
- LCD I602 (I2C) * 1 組
- 麵包版 400孔 * 1 片
- 手機支架 * 1 組
- 骰子 * 1 盒
- USB type-C cable * 1 條
- 杜邦線(公對母) * 4 條



開發環境建置

- 請使用 Arduino IDE 1.8.19 之後版本。
- 本次開發使用 Windows 10 搭配 Arduino IDE 2.3.2 版本



案例介紹

- 功能說明：

- 自動辨識骰子骰出的點數種類，除了在RTSP可以看到標記的骰子外，也同時計算總數顯示在外接LCD屏上。
- 由於骰子可能有不同角度，如何不辨識到側邊的骰子，只取得我們需要的那一面是有一定的困難度，所以需要準備更多骰子的各種情境來做辨識。

案例介紹

- 骰子訓練文件說明：

- dice資料夾：

- 照片，標註檔以及訓練用資料。

- results資料夾：

- 訓練完的權重檔、設定檔以及轉換完成的nb。

- 轉換文件及檔案：

- 三種colab教案，acuity轉換文件檔案。

AI 骰子模型訓練-前言

本範例AI骰子模型訓練在Google Colab 上使用。

網址：<https://colab.research.google.com/>

Colab 有以下特點

- 對本身電腦硬件要求不高
- 可開發及訓練類神經網路
- 方便共用

AI 骰子模型訓練-資料取得

此模型訓練的資料一共有六個點數，全部有**68**張照片，都是透過**HUB 8735 ultra** 進行拍照取得的樣本，以下為部分內容：

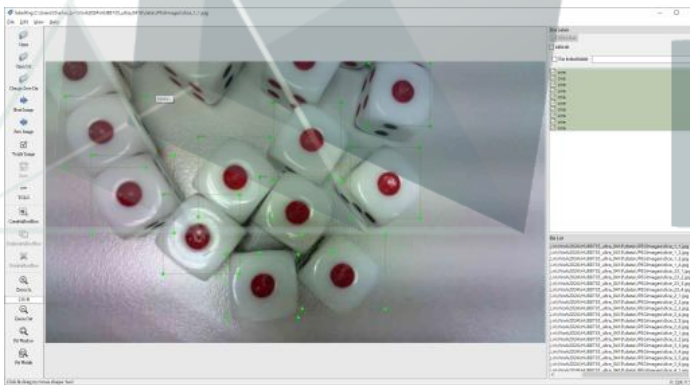


WinsTouch

AI 骰子模型訓練-標註

- 標註：

- 使用Labellmg標註軟體做特徵值標註，此軟體可提供YOLO(*.txt)的標註格式，本範例分別標註不同的點數：
- One、two、three、four、five、six。



AI 骰子模型訓練-訓練準備

- **Darknet 訓練前準備(dice資料夾) :**

- JPEGImages 資料夾內包含68張照片
- Labels資料夾內包含68個標註檔
- classes.txt 為標註種類
- yolov4-tiny-test.cfg 為自定義模型及參數
- test.data 資料集設定檔
- test.name 資料標籤名稱
- train.txt 自定義資料訓練集檔案列表
- val.txt 自定義資料驗證集檔案列表

AI 骰子模型訓練-導入Darknet

yolov4-tiny-test.cfg 為自定義模型及參數。

line 20: max_batches = 12000 #為classes數量 *2000

line 22: steps=7200,9600 #為max_batches的60%,80%

line 212, 263: filters=33 #為 (classes數量+5) *3

line 220, 269: classes=6 #為物件辨識數量(骰子1~6點)

AI 骰子模型訓練-導入Darknet

test.data 資料集設定檔，classes一共六個類別，其餘設定為各資料的路徑。

檔案內容：

```
classes= 6
```

```
train = train.txt
```

```
valid = test.txt
```

```
#valid = data/coco_val_5k.list
```

```
names = test.names
```

```
backup = backup/
```

Ideas
Hatch

WinsTouch

AI 骰子模型訓練-導入Darknet

test.name 資料標籤名稱，設定每一個類別的名稱。

檔案內容：

one

two

three

four

five

six

Ideas
Hatch

WinsTouch

AI 骰子模型訓練-導入Darknet

train.txt 自定義訓練集檔案列表。為照片的檔案路徑與名稱。

部分內容列舉如下：

./JPEGImages/dice_1_1.jpg

./JPEGImages/dice_1_2.jpg

./JPEGImages/dice_1_3.jpg

./JPEGImages/dice_1_4.jpg

./JPEGImages/dice_23_1.jpg

./JPEGImages/dice_23_2.jpg

./JPEGImages/dice_23_3.jpg

./JPEGImages/dice_23_4.jpg

Ideas
Hatch

WinsTouch

AI 骰子模型訓練-導入Darknet

val.txt 自定義驗證集檔案列表。為照片的檔案路徑與名稱。
因為是驗證用檔案，可自行增訂數量。

本範例內容如下：

./JPEGImages/dice_m_3.jpg

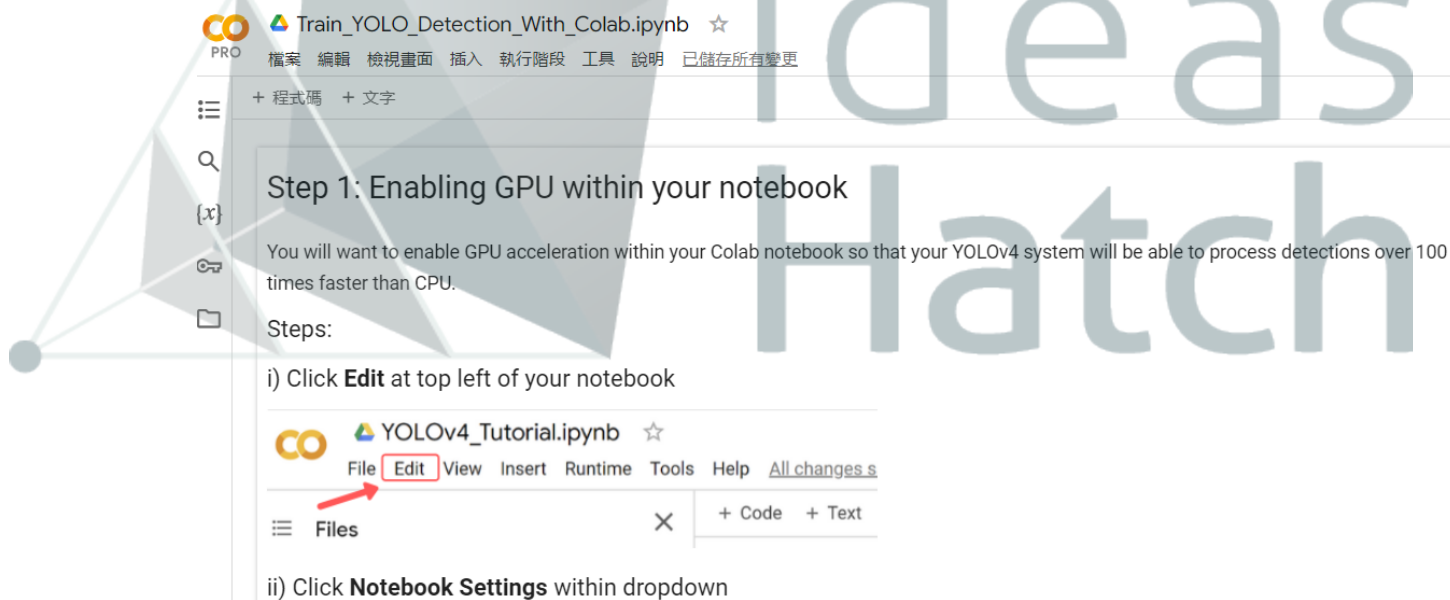
Ideas
Hatch

WinsTouch

AI 骰子模型訓練-訓練前準備

打開Colab開啟Train_YOLO_Detection_With_Colab.ipynb

Step 1: 開啟筆記本上的GPU



Step 1: Enabling GPU within your notebook

You will want to enable GPU acceleration within your Colab notebook so that your YOLOv4 system will be able to process detections over 100 times faster than CPU.

Steps:

- Click **Edit** at top left of your notebook
- Click **Notebook Settings** within dropdown

AI 骰子模型訓練-訓練前準備

Step 2: 從github下載Darknet並且修改makefile，將GPU，CUDNN，CUDNN_HALF，OPENCV設為Enable後，編譯Compiler Darknet。

Train_YOLO_Detection_With_Colab.ipynb ☆
檔案 編輯 檢視畫面 插入 執行階段 工具 說明 已儲存所有變更

+ 程式碼 + 文字

Step 2: Cloning and Building Darknet

The following cells will clone darknet from AlexeyAB's famous repository, adjust the Makefile to enable OPENCV and GPU for darknet and then build darknet.

Do not worry about any warnings when you run the 'make' cell!

```
[ ] # clone darknet repo
!git clone https://github.com/AlexeyAB/darknet
```

```
[ ] # change makefile to have GPU and OPENCV enabled
!cd darknet
!sed -i 's/OPENCV=0/OPENCV=1/' Makefile
!sed -i 's/GPU=0/GPU=1/' Makefile
!sed -i 's/CUDNN=0/CUDNN=1/' Makefile
!sed -i 's/CUDNN_HALF=0/CUDNN_HALF=1/' Makefile
```

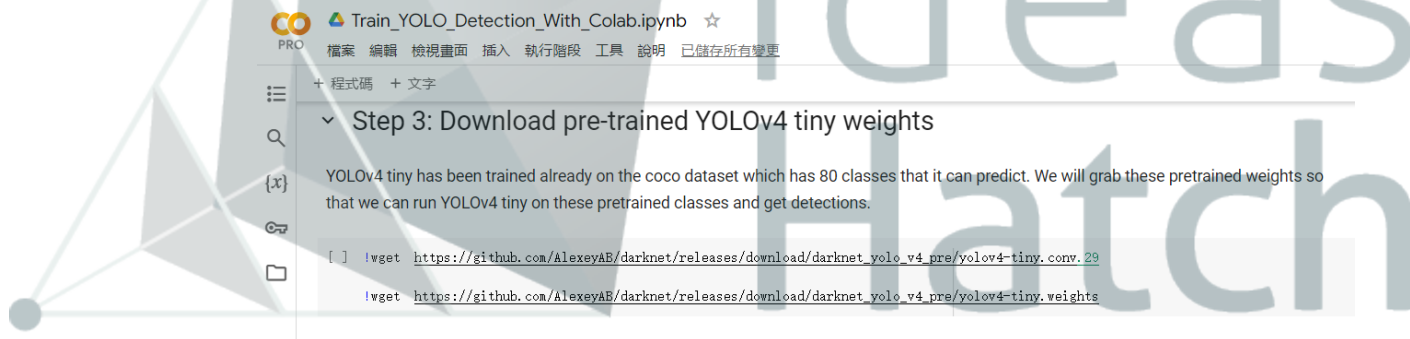
```
[ ] # verify CUDA
!usr/local/cuda/bin/nvcc --version
```

```
[ ] # make darknet (builds darknet so that you can then use the darknet executable file to run or train object detectors)
!make
```

WinsTouch

AI 骰子模型訓練-訓練前準備

Step 3: 從github下載YOLOv4-tiny的預訓練檔(conv.29)以及權重檔(weights)。



Train_YOLO_Detection_With_Colab.ipynb ☆

檔案 編輯 檢視畫面 插入 執行階段 工具 說明 已儲存所有變更

+ 程式碼 + 文字

▽ Step 3: Download pre-trained YOLOv4 tiny weights

YOLOv4 tiny has been trained already on the coco dataset which has 80 classes that it can predict. We will grab these pretrained weights so that we can run YOLOv4 tiny on these pretrained classes and get detections.

```
[ ] !wget https://github.com/AlexeyAB/darknet/releases/download/darknet_yolo_v4_pre/yolov4-tiny.conv.29
!wget https://github.com/AlexeyAB/darknet/releases/download/darknet_yolo_v4_pre/yolov4-tiny.weights
```

AI 骰子模型訓練-訓練前準備

Step 4:執行自定義的幫助函式。

```
# define helper functions
def imshow(path):
    import cv2
    import matplotlib.pyplot as plt
    %matplotlib inline

    image = cv2.imread(path)
    height, width = image.shape[:2]
    resized_image = cv2.resize(image, (3*width, 3*height), interpolation = cv2.INTER_CUBIC)

    fig = plt.gcf()
    fig.set_size_inches(18, 10)
    plt.axis('off')
```

AI 骰子模型訓練-訓練前準備

Step 5: 測試Darknet，使用內建的coco資料集。

Train_YOLO_Detection_With_Colab.ipynb

檔案 編輯 檢視畫面 插入 執行階段 工具 說明 已儲存所有變更

+ 程式碼 + 文字

Step 5: Run Your Detections with Darknet and YOLOv4 tiny!

Darknet is now built and ready to run detections using YOLOv4 tiny in the cloud! You can find out which sorts of classes the pre-trained YOLOv4 tiny weights can detect by clicking here. [COCO CLASSES](#)

The object detector can be run using the following command

```
./darknet detector test <path to .data file> <path to config> <path to weights> <path to image>
```

Darknet comes with a few images already installed in the darknet/data/ folder.

Note: After running detections OpenCV can't open the image instantly in the cloud so we must run:

```
imshow('predictions.jpg')
```

This will output the image with the detections shown. The most recent detections are always saved to 'predictions.jpg'

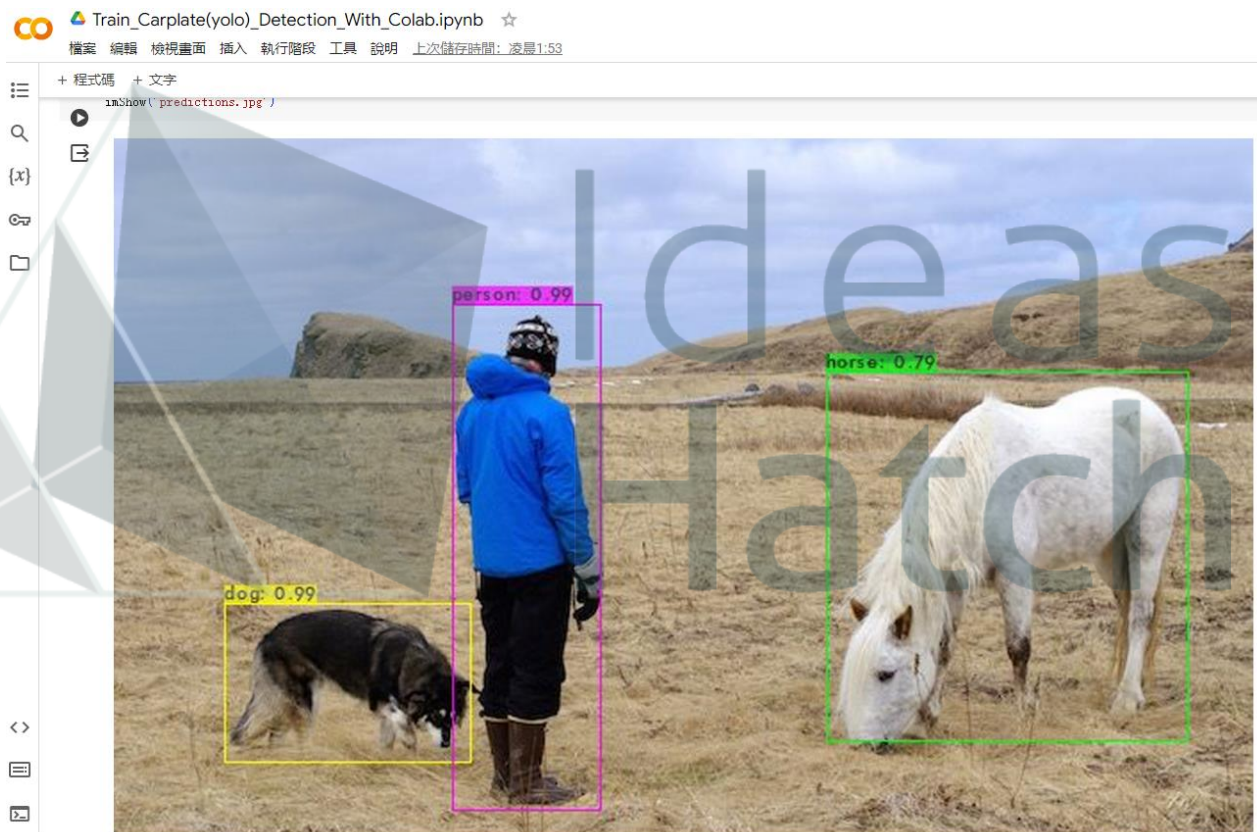
Try out the examples below for yourself!

```
[ ] # run darknet detection on test images
./darknet detector test cfg/coco.data cfg/yolov4-tiny.cfg yolov4-tiny.weights data/person.jpg
```

```
[ ] # show image using our helper function
imshow('predictions.jpg')
```

WinsTouch

AI 骰子模型訓練-訓練前準備



WinsTouch

AI 骰子模型訓練-開始訓練

將之前訓練準備的dice資料夾複製到雲端硬碟內。並透過指令複製到darknet目錄下。

Train_YOLO_Detection_With_Colab.ipynb

+ 程式碼 + 文字

換為自己的AI模型for HUB 8735 ultra

連線雲端硬碟

```
[10] from google.colab import drive
drive.mount('/content/gdrive')
```

Mounted at /content/gdrive

複製雲端硬碟資料夾dice底下的照片及label資料到darknet資料夾

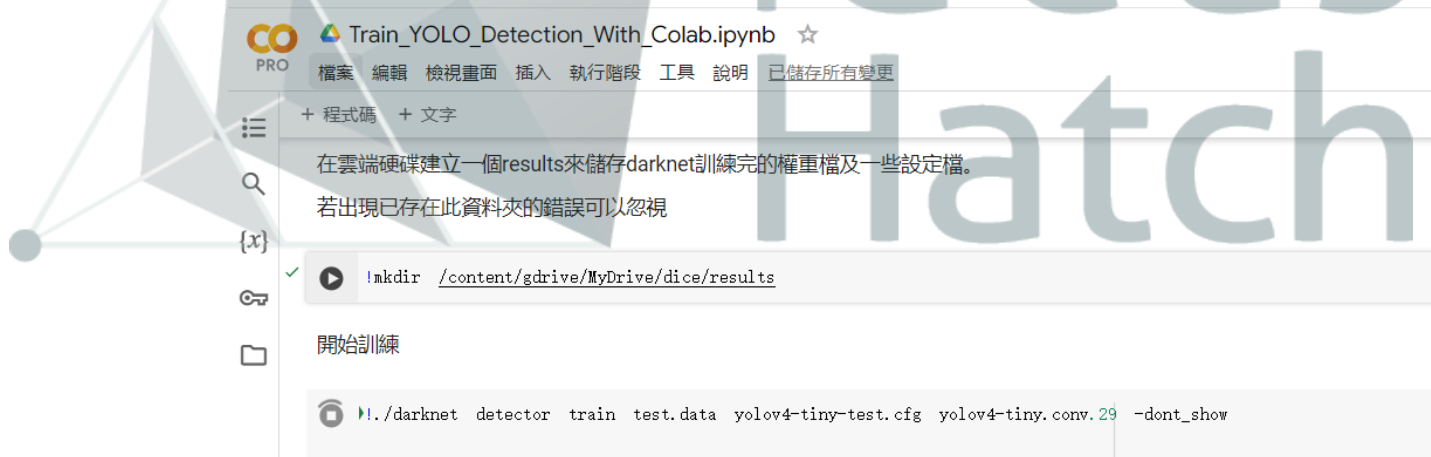
```
[17] !cp -r /content/gdrive/MyDrive/dice/JPEGImages /content/darknet/
!cp -r /content/gdrive/MyDrive/dice/labels /content/darknet/
!cp -r /content/gdrive/MyDrive/dice/classes.txt /content/darknet/
!cp -r /content/gdrive/MyDrive/dice/train.txt /content/darknet/
!cp -r /content/gdrive/MyDrive/dice/val.txt /content/darknet/
#複製dice資料夾底下的yolov4-tiny-test.cfg、test.data、test.names檔案到darknet根目錄下
!cp -r /content/gdrive/MyDrive/dice/yolov4-tiny-test.cfg /content/darknet/yolov4-tiny-test.cfg
!cp -r /content/gdrive/MyDrive/dice/test.names /content/darknet/test.names
!cp -r /content/gdrive/MyDrive/dice/test.data /content/darknet/test.data
```

WinsTouch

AI 骰子模型訓練-開始訓練

準備完成，開始訓練。

訓練時間約1~2小時，訓練結果會在darknet/backup資料夾下生成yolov4-tiny-test_final.weights權重檔。



```
Train_YOLO_Detection_With_Colab.ipynb ☆
PRO 檔案 編輯 檢視畫面 插入 執行階段 工具 說明 已儲存所有變更

+ 程式碼 + 文字

在雲端硬碟建立一個results來儲存darknet訓練完的權重檔及一些設定檔。
若出現已存在此資料夾的錯誤可以忽視

!mkdir /content/gdrive/MyDrive/dice/results

開始訓練

!./darknet detector train test.data yolov4-tiny-test.cfg yolov4-tiny.conv.29 -dont_show
```

AI 骰子模型訓練-訓練結果

驗證模型結果，透過以下指令來辨識JPEGImages/dice_mix_4.jpg
並將最終權重檔跟設定檔複製回雲端硬碟，以便轉檔使用。



```
Train_YOLO_Detection_With_Colab.ipynb ☆
檔案 編輯 檢視畫面 插入 執行階段 工具 說明

+ 程式碼 + 文字

✓ 測試模型

# run darknet detection on test images
!./darknet detector test test.data yolov4-tiny-test.cfg backup/yolov4-tiny-test_last.weights JPEGImages/dice_mix_4.jpg
imshow('predictions.jpg')

將yolov4-tiny_test_last.weight 及 yolov4-tiny-test.cfg 都放到雲端硬碟空間中

!cp -r /content/darknet/yolov4_tiny-test.cfg (ctrl + click) .weights /content/gdrive/MyDrive/results/
!cp -r /content/darknet/yolov4_tiny-test.cfg /content/gdrive/MyDrive/results/
```

AI 骰子模型訓練-訓練結果

從照片可以確認訓練結果是可以清楚辨識出骰子類型的。



AI 骰子模型訓練-轉檔準備

我們需要將權重檔轉檔成HUB 8735 ultra可以接受的格式。

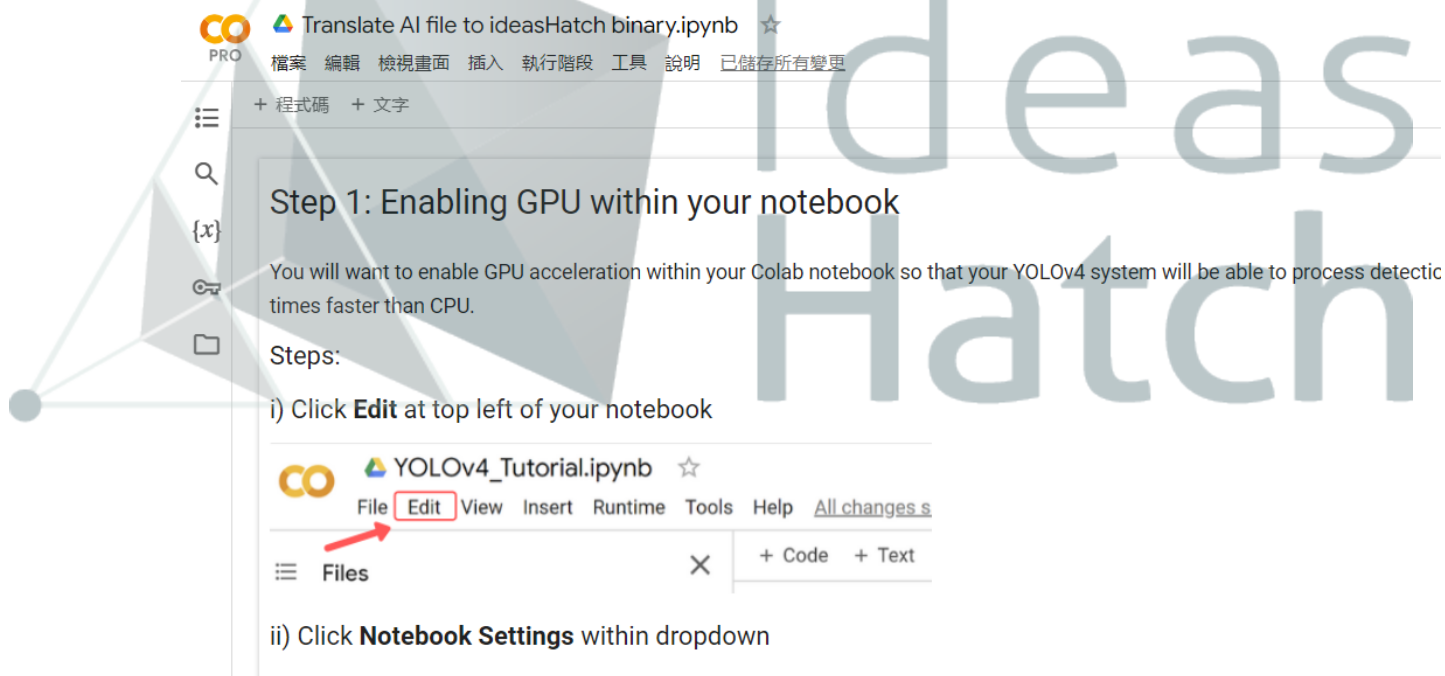
轉檔前準備(translate資料夾)：

- YOLOv4-tiny-test_last.weights 訓練完的權重檔
- YOLOv4-tiny-test.cfg 訓練的設定檔
- 0_import_model_test.sh 導入模型的腳本
- 1_quantize_model_test.sh 量化模型的腳本
- 2_export_case_code_test.sh 轉換模型的腳本
- VIPNANOQI_PID0XAD 認證檔

AI 骰子模型訓練-轉檔準備

打開Colab開啟Translate AI file to ideasHatch binary.ipynb

Step 1: 一樣開啟筆記本上的GPU，否則轉檔過程會有錯誤。



CO PRO Translate AI file to ideasHatch binary.ipynb ☆

檔案 編輯 檢視畫面 插入 執行階段 工具 說明 已儲存所有變更

+ 程式碼 + 文字

Step 1: Enabling GPU within your notebook

You will want to enable GPU acceleration within your Colab notebook so that your YOLOv4 system will be able to process detectic times faster than CPU.

Steps:

i) Click **Edit** at top left of your notebook

CO YOLOv4_Tutorial.ipynb ☆

File **Edit** View Insert Runtime Tools Help [All changes s](#)

Files × + Code + Text

ii) Click **Notebook Settings** within dropdown

AI 骰子模型訓練-開始轉檔

Clone the translation tool



AI 骰子模型訓練-開始轉檔

開啟Google雲端硬碟，並同意所有連線。

進入acuity-toolkit/demo資料夾後建立轉換用的路徑資料夾。



WinsTouch

AI 骰子模型訓練-開始轉檔

將translate資料夾複製到雲端硬碟內。

複製雲端硬碟內 translate跟dice目錄的相關檔案到指令目錄。



```
!cp -r /content/gdrive/MyDrive/translate/VIPNANOQI_PIDOXAD /content/aml_npu_sdk/acuity-toolkit/bin/
!cp -r /content/gdrive/MyDrive/dice/results/* /content/aml_npu_sdk/acuity-toolkit/demo/model_test/
!cp -r /content/gdrive/MyDrive/dice/JPEGImages/* /content/aml_npu_sdk/acuity-toolkit/demo/JPEGImages/
!cp -r /content/gdrive/MyDrive/dice/train.txt /content/aml_npu_sdk/acuity-toolkit/demo/dataset_test.txt

!cp -r /content/gdrive/MyDrive/translate/0_import_model_test.sh /content/aml_npu_sdk/acuity-toolkit/demo/
!cp -r /content/gdrive/MyDrive/translate/1_quantize_model_test.sh /content/aml_npu_sdk/acuity-toolkit/demo/
!cp -r /content/gdrive/MyDrive/translate/2_export_case_code_test.sh /content/aml_npu_sdk/acuity-toolkit/demo/
```

AI 骰子模型訓練-開始轉檔

將translate資料夾複製到雲端硬碟內。

複製雲端硬碟內 translate跟dice目錄的相關檔案到指令目錄。



```
Translate AI file to ideasHatch binary.ipynb ☆
檔案 編輯 檢視畫面 插入 執行階段 工具 說明 已儲存所有變更

+ 程式碼 + 文字

複製轉檔的相關資料

!cp -r /content/gdrive/MyDrive/translate/VIPNANOQI_PIDOXAD /content/aml_npu_sdk/acuity-toolkit/bin/
!cp -r /content/gdrive/MyDrive/dice/results/* /content/aml_npu_sdk/acuity-toolkit/demo/model_test/
!cp -r /content/gdrive/MyDrive/dice/JPEGImages/* /content/aml_npu_sdk/acuity-toolkit/demo/JPEGImages/
!cp -r /content/gdrive/MyDrive/dice/train.txt /content/aml_npu_sdk/acuity-toolkit/demo/dataset_test.txt

!cp -r /content/gdrive/MyDrive/translate/0_import_model_test.sh /content/aml_npu_sdk/acuity-toolkit/demo/
!cp -r /content/gdrive/MyDrive/translate/1_quantize_model_test.sh /content/aml_npu_sdk/acuity-toolkit/demo/
!cp -r /content/gdrive/MyDrive/translate/2_export_case_code_test.sh /content/aml_npu_sdk/acuity-toolkit/demo/
```

AI 骰子模型訓練-開始轉檔

導入模型，執行0_import_model.sh



The screenshot displays the JupyterLab interface. At the top, the title bar reads "Translate AI file to ideasHatch binary.ipynb" with a star icon. Below the title bar, a menu bar includes "檔案", "編輯", "檢視畫面", "插入", "執行階段", "工具", and "說明". The left sidebar contains icons for a file explorer, search, and a list of files. The main area shows a list of steps to follow:

- 依序執行下面步驟
- 1. 導入模型(0_import_model.sh)
- 2. 量化模型(1_quantize_model.sh)
- 3. 生成 Binary (2_export_case_code.sh)

Below the list, a folder icon is visible. The bottom section shows a terminal output for the command `!bash ./0_import_model_test.sh`, which has been executed successfully, indicated by a green checkmark.

AI 骰子模型訓練-開始轉檔

量化模型，執行1_quantize_model.sh



WinsTouch

AI 骰子模型訓練-開始轉檔

生成模型，執行2_quantize_model.sh

並將生成出來的yolov4_test.nb複製到雲端硬碟內。



AI 骰子模型訓練-Arduino程式

開啟檔案總管至

C:\Users\<User>\AppData\Local\Arduino15\packages\ideasHatch\hardware\AmebaPro2\4.0.10-Release\variants\ common_nn_models

將剛剛複製到雲端硬碟的yolov4_test.nb下載並更名替換原本的yolo4_tiny.nb。

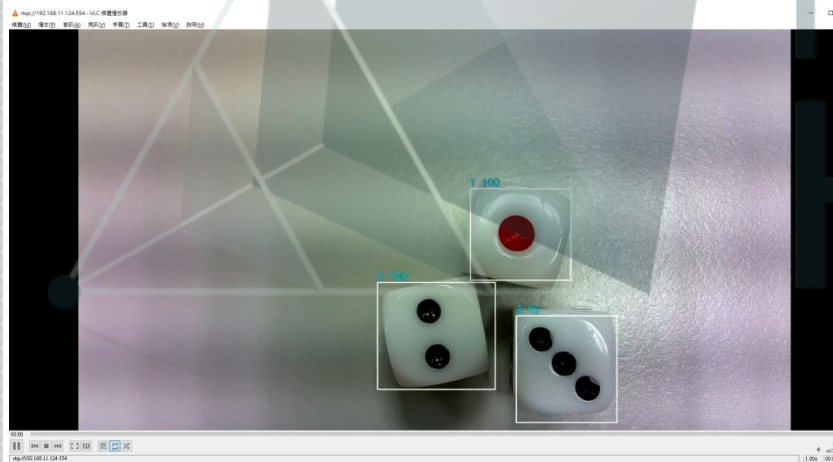
AI 骰子模型訓練-Arduino程式

透過Arduino IDE -> 檔案 -> 開啟... 開啟Arduino SDK\Dice底下的dice.ino。我們已經將物件種類修改成1.2.3.4.5.6在ObjectClassList.h

```
13 ObjectDetectionItem itemList[6] = {  
14     {0, "1", 1},  
15     {1, "2", 1},  
16     {2, "3", 1},  
17     {3, "4", 1},  
18     {4, "5", 1},  
19     {5, "6", 1},  
20 };
```

AI 骰子模型訓練-Arduino程式

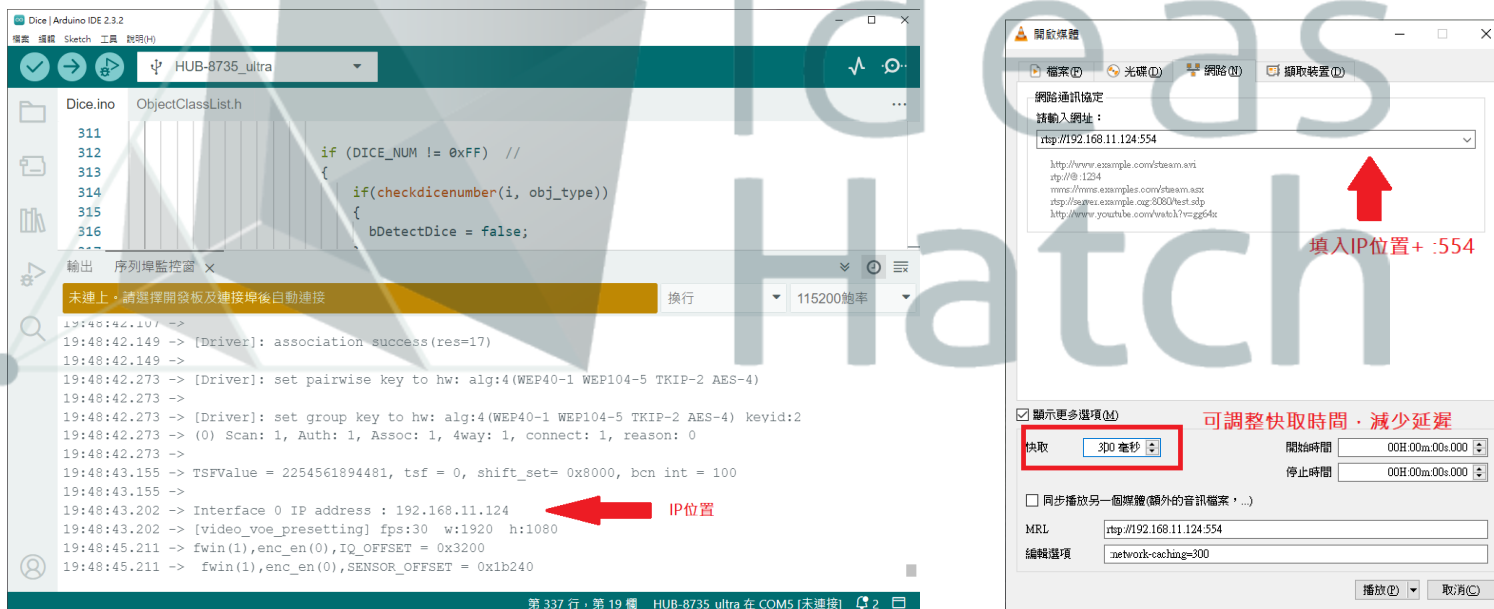
修改完自己的wi-fi帳號密碼後，上傳至HUB 8735 ultra。以下為透過VLC軟體接收video streaming畫面來確認辨識效果。



WinsTouch

AI 骰子模型訓練-VLC使用說明

Arduino 序列埠監控窗內印出的IP位置為video streaming的位置，將位置如右圖方式填入至VLC軟體，rtsp://<ip address>:554



AI 骰子模型訓練-Arduino程式

定義骰子數量跟固定確認時間

```
53 #define DICE_NUM 3
54 #define check_time 10
```

LCD1602 定義I2C位置

```
80 // LCM1602 I2C LCD
81 LiquidCrystal I2C lcd(0x27, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE); // Set the LCD I2C address
```

LCD1602 畫面清空跟填新的值

```
88 void print_dice_rolling()
89 {
90     lcd.clear();
91     lcd.setCursor(0, 0); // go to home
92     lcd.print("GAME Start");
93
94     lcd.setCursor(0, 1); // go to the next line
95     lcd.print("Rolling... ");
96 }
```

AI 骰子模型訓練-Arduino程式

將辨識出來**3**顆骰子的點數以及加總寫到**LCD1602**上。

```
98 void print_dice_result(uint8_t n1, uint8_t n2, uint8_t n3)
99 {
100     lcd.clear();
101     lcd.setCursor(0, 0);    // go to home
102     lcd.print("The dice: ");
103     lcd.print(itemList[n1].objectName);
104     lcd.print(",");
105     lcd.print(itemList[n2].objectName);
106     lcd.print(",");
107     lcd.print(itemList[n3].objectName);
108
109     lcd.setCursor(0, 1);    // go to the next line
110     lcd.print("Total: ");
111     lcd.print(itemList[n1].index + itemList[n2].index + itemList[n3].index + 3);
112     lcd.print(".");
113
114     snprintf(osd_dice_str, sizeof(osd_dice_str), "The dice: %s %s %s ", itemList[n1].objectName,
115             snprintf(osd_dice_str2, sizeof(osd_dice_str2), "Total: %d.", itemList[n1].index + itemList[n2].index + itemList[n3].index + 3);
116     bShowDiceOSDOnce = true;
117 //     OSD.drawText(CHANNEL, xmin, ymin - OSD.getTextHeight(CHANNEL), text_str, OSD_COLOR_CYAN);
118 }
```

AI 骰子模型訓練-Arduino程式

LCD initial code

```
126   lcd.begin(16, 2, LCD_5x8DOTS, Wire);    // initialize the lcd
127   // lcd.begin(16, 2, LCD_5x8DOTS, Wire1);    // Uncomment this line to use Wire1
128
129   lcd.backlight();
130
131   lcd.setCursor(0, 0);    // go to home
132   lcd.print("*** DICE  GAME ***");
133   lcd.setCursor(0, 1);    // go to the next line
134   lcd.print("Press BTN Start");
135
```

判斷三顆骰子的點數，如果點數一致則累計固定時間。

```
215 bool checkdicenumber(uint8_t index, uint8_t cur_index)
216 {
217     if (Dice[index].index == 0xff)
218     {
219         Dice[index].index = cur_index;
220         Dice[index].count++;
221     }
222     else if (Dice[index].index == cur_index)
223     {
224         Dice[index].count++;
225     }
226     else //Dice[i].index !=
227     {
228         cleardice();
229     }
230
231     printf("*****\r\n");
232     printf("Dice[0] = %d, %d\r\n", Dice[0].index, Dice[0].count);

```

AI 骰子模型訓練-Arduino程式

演算骰子行為：若各條件滿足，則**OSD**可印出點數。否則持續刷新辨識

```
263     if ((!bDetectDice) || (DICE_NUM == 0xFF))
264     {
265         OSD.createBitmap(CHANNEL);
266         if (bShowDiceOSDOnce)
267         {
268             OSD.drawText(CHANNEL, 10, 10, osd_dice_str, OSD_COLOR_BLACK);
269             OSD.drawText(CHANNEL, 10, 40, osd_dice_str2, OSD_COLOR_BLACK);
270         }
271         OSD.update(CHANNEL);
272
273         bDetectDice = true;
274     }
275     else
276     {
277         OSD.createBitmap(CHANNEL);
278         OSD.update(CHANNEL);
279         bShowDiceOSDOnce = false;
280     }
281
282     if(bDetectDice) 若辨識數量為三個，才進行演算處理。
283     {
284         printf("Total number of objects detected = %d\r\n", ObjDet.getResultCount());
285
286         if ((ObjDet.getResultCount() == DICE_NUM) || (DICE_NUM == 0xFF))
```

AI 骰子模型訓練-Arduino程式

得知三個骰子的位置、準確度以及種類，標記並顯示出來。

```
287     {
288         OSD.createBitmap(CHANNEL);
289
290         if (ObjDet.getResultCount() > 0) {
291             for (int i = 0; i < ObjDet.getResultCount(); i++) {
292                 int obj_type = results[i].type();
293                 if (itemList[obj_type].filter) { // check if item should be ignored
294
295                     ObjectDetectionResult item = results[i];
296                     // Result coordinates are floats ranging from 0.00 to 1.00
297                     // Multiply with RTSP resolution to get coordinates in pixels
298                     int xmin = (int)(item.xMin() * im_w);
299                     int xmax = (int)(item.xMax() * im_w);
300                     int ymin = (int)(item.yMin() * im_h);
301                     int ymax = (int)(item.yMax() * im_h);
302
303                     // Draw boundary box
304                     printf("Item %d %s:\t%d %d %d %d\n\r", i, itemList[obj_type].objectName,
305                           OSD.drawRect(CHANNEL, xmin, ymin, xmax, ymax, 3, OSD_COLOR_WHITE);
306
307                     // Print identification text
308                     char text_str[20];
309                     snprintf(text_str, sizeof(text_str), "%s %d", itemList[obj_type].objectN
```

AI 骰子模型訓練-Arduino程式

```
310         OSD.drawText(CHANNEL, xmin, ymin - OSD.getTextHeight(CHANNEL), text_str,
311
312         if (DICE_NUM != 0xFF) //
313         {
314             if(checkdicenumber(i, obj_type))
315             {
316                 bDetectDice = false;
317             }
318         }
319     }
320 }
321 }
322 OSD.update(CHANNEL);
323 }
324 else
325 {
326     print_dice_rolling();
327     loop_timeout--;
328     if (loop_timeout == 0)
329     {
330         loop_timeout = 30;
331         bDetectDice = 0;
332     }
333 }
334 // delay to wait for new results
335 //delay(100);
336 }
337 // delay to wait for new results
338 delay(100);
339 }
```

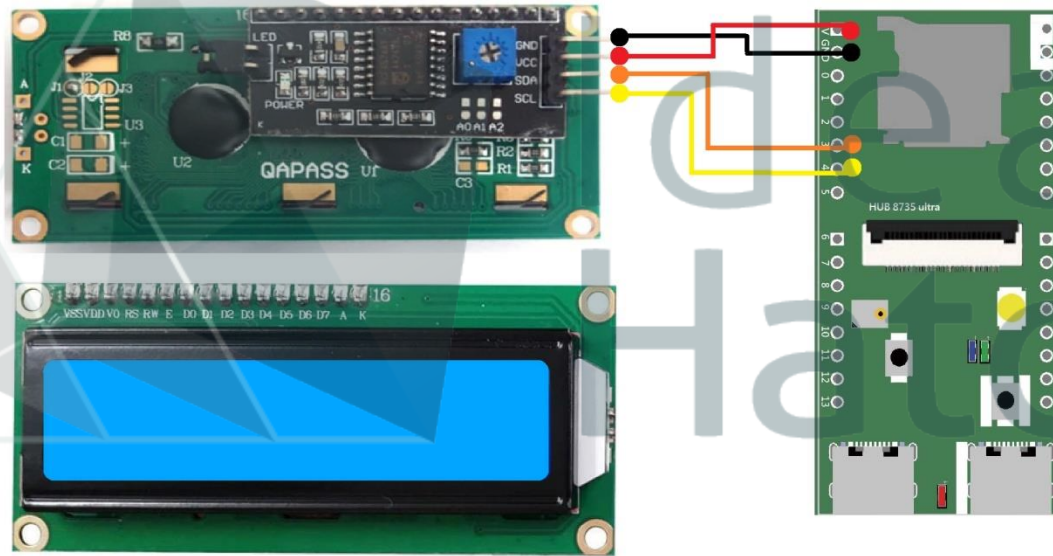
若辨識到的骰子數量不是三個，則準備重新開始。(可能被拿起來準備骰下一回合)

AI 骰子模型訓練-Arduino程式

```
310         OSD.drawText(CHANNEL, xmin, ymin - OSD.getTextHeight(CHANNEL), text_str,
311
312         if (DICE_NUM != 0xFF) //
313         {
314             if(checkdicenumber(i, obj_type))
315             {
316                 bDetectDice = false;
317             }
318         }
319     }
320 }
321 }
322 OSD.update(CHANNEL);
323 }
324 else
325 {
326     print_dice_rolling();
327     loop_timeout--;
328     if (loop_timeout == 0)
329     {
330         loop_timeout = 30;
331         bDetectDice = 0;
332     }
333 }
334 // delay to wait for new results
335 //delay(100);
336 }
337 // delay to wait for new results
338 delay(100);
339 }
```

若辨識到的骰子數量不是三個，則準備重新開始。(可能被拿起來準備骰下一回合)

AI 骰子模型訓練-LCD1602接線



WinsTouch

AI 骰子模型訓練-組合成品

在還沒有骰子的時候判斷 rolling



WinsTouch

AI 骰子模型訓練-組合成品

有辨識到三顆骰子的時候在螢幕上顯示點數以及總和。

