程式碼 https://github.com/ZH8000/FlyCamera.git

混料

相關檔案：

CommonFlySDK.hpp
CommonFlySDK.cpp
CameraProp.hpp
OpenCVProp.hpp
AlexMatch_MultiCamera_v2.hpp
AlexMatch_MultiCamera_v2.cpp

執行方法：

開啟 CMakeLists.txt 檔案，add_executable 部份只留下以下指令。

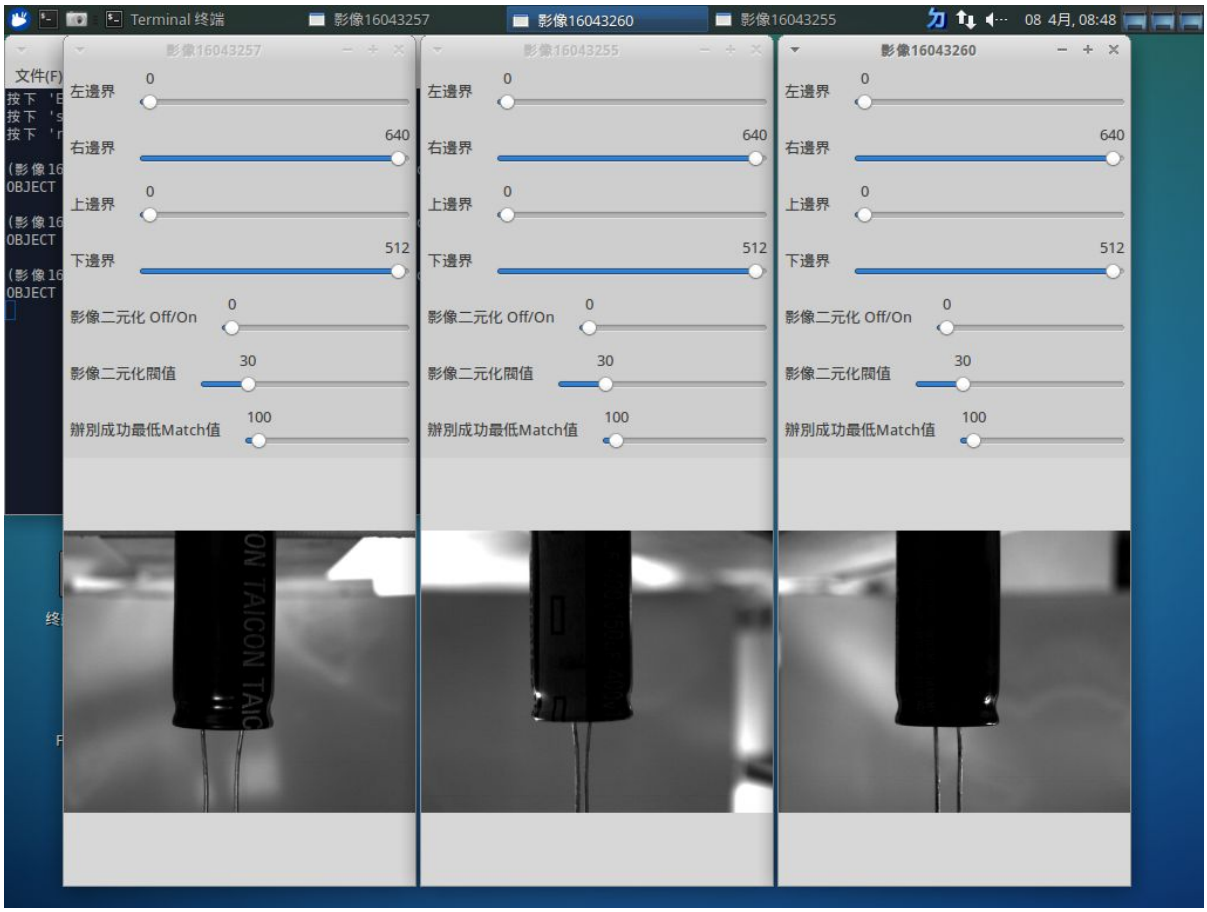
```
add_executable( app AlexMatch_MultiCamera_v2.cpp CommonFlySDK.cpp )
```

在 FlyCamera 底下再開一資料夾 release，如下：
mkdir release
cd release
cmake ..
make

會在 release 資料夾底下產生 app 檔案，執行即可。
./app

操作方法：

先取樣幾個正常電容之樣本，之後在針對每個需要檢測的樣本取樣與正常樣本比較差異度，若差異大於某個臨界值則表示並不屬於同一種型號。



按鍵 s：

1. 擷取作為樣本的影像。可以在 AlexMatch_MultiCamera_v2.cpp 中設定樣本數 (sampleImagesSize)。
2. 如果已經擷取到所設定的樣本數，在按下 s 會開始比較影像。

按鍵 r：清除已經擷取的樣本影像。

按鍵 Esc：離開程式。

a. 『左，右，上，下邊界』：調整畫面範圍。
b. 『影像二元化 Off/On』：影像處理。

c. 『影像二元化閥值』：影像處理程度。
d. 『辨別成功最低Match值』：判別影像的門檻值。

頭折（短）

相關檔案：

2_Circles_v3_head.cpp

執行方法：

開啟 CMakeLists.txt 檔案，add_executable 部份只留下以下指令。

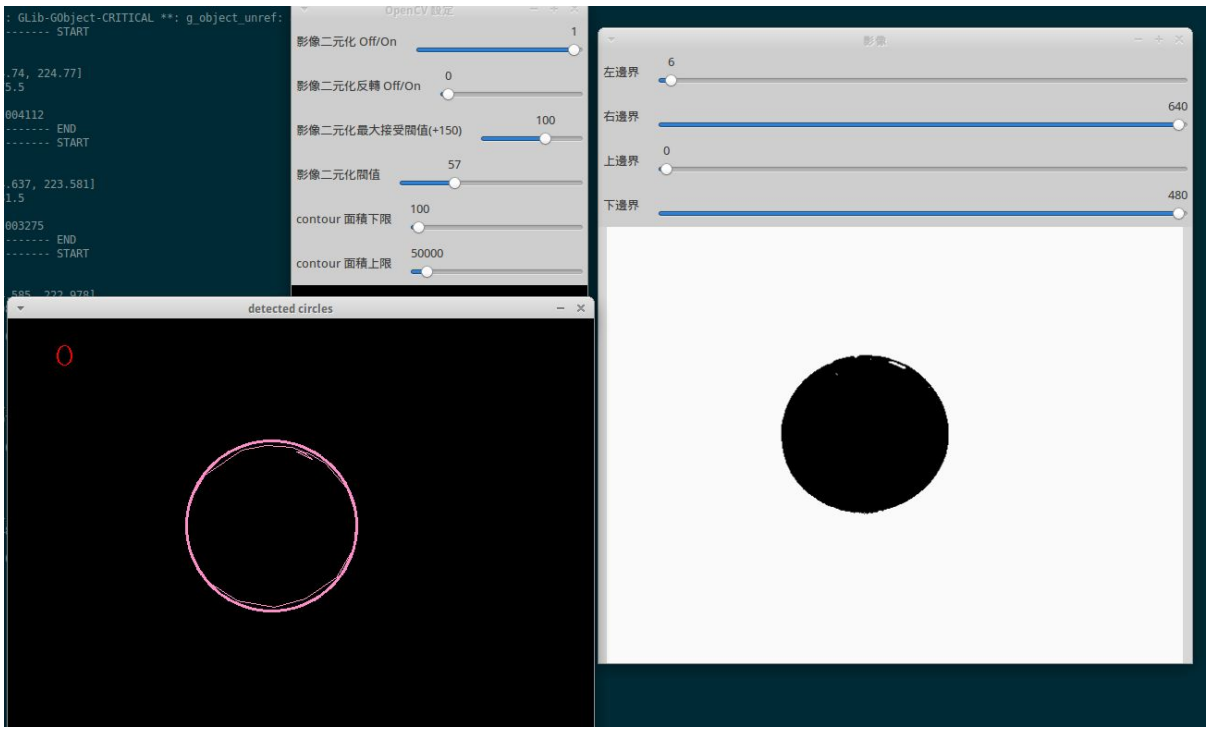
```
add_executable( app 2_Circles_v3_head.cpp )
```

在 FlyCamera 底下再開一資料夾 release，如下：
mkdir release
cd release
cmake ..
make

會在 release 資料夾底下產生 app 檔案，執行即可。
./app

操作方法：

如果電容頭部正常完整，則不應該露出任何金屬部位，故檢測畫面中影像若是只有一個完整的電容輪廓則為正常電容。



按鍵 s：檢測電容。
按鍵 q：離開程式。

e. 『左，右，上，下邊界』：調整畫面範圍。
f. 『影像二元化 Off/On』：影像處理。
g. 『影像二元化閥值』：影像處理程度。
h. 『辨別成功最低Match值』：判別影像的門檻值。

尾頭折（短）

相關檔案：

2_Circles_v3.cpp

執行方法：

開啟 CMakeLists.txt 檔案，add_executable 部份只留下以下指令。

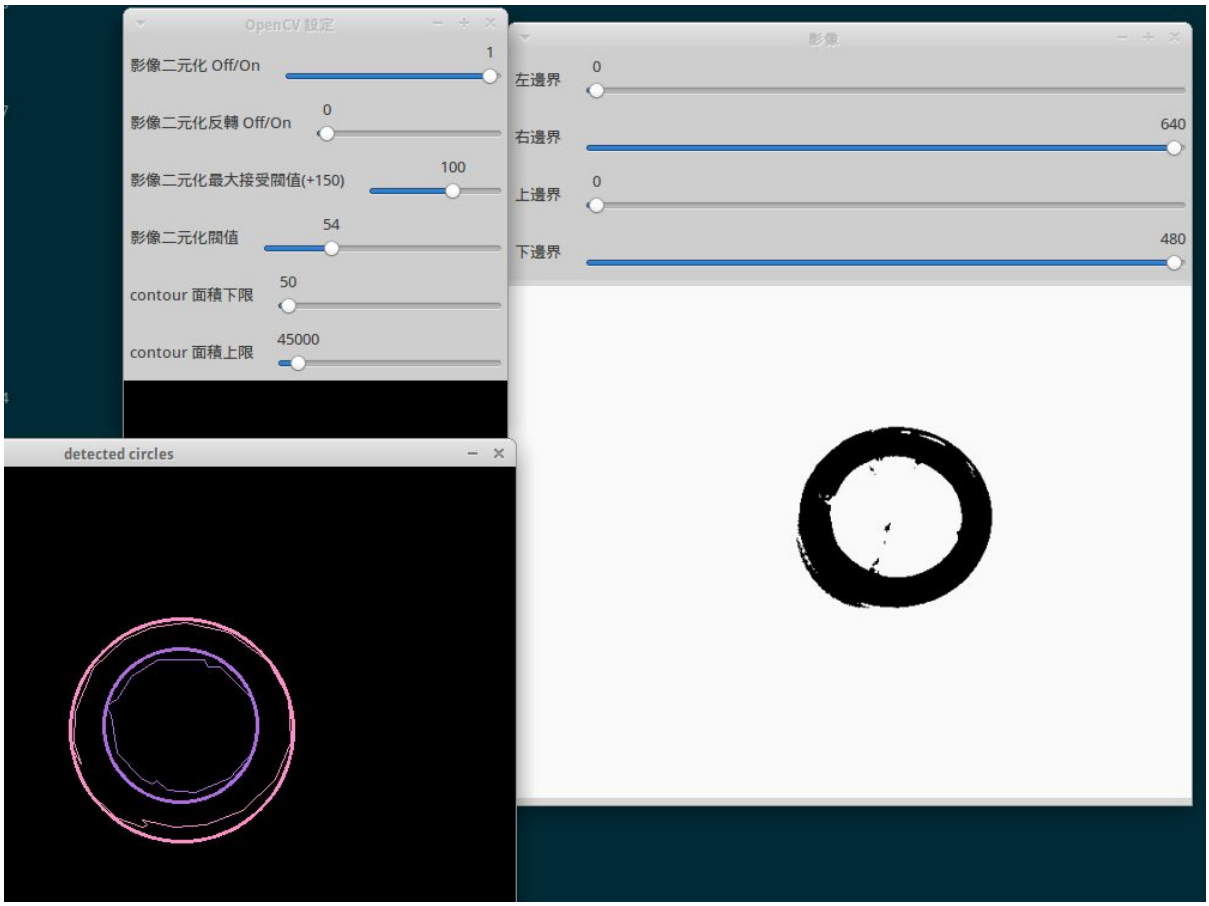
```
add_executable( app AlexMatch_MultiCamera_v2.cpp CommonFlySDK.cpp )
```

在 FlyCamera 底下再開一資料夾 release，如下：
mkdir release
cd release
cmake ..
make

會在 release 資料夾底下產生 app 檔案，執行即可。
./app

操作方法：

如果電容尾部正常完整，則不應該露出任何金屬部位，故檢測畫面中影像若是只有一個完整的電容輪廓則為正常電容。



按鍵 s：檢測電容。
按鍵 q：離開程式。

i. 『左，右，上，下邊界』：調整畫面範圍。
j. 『影像二元化 Off/On』：影像處理。
k. 『影像二元化閥值』：影像處理程度。
l. 『辨別成功最低Match值』：判別影像的門檻值。