|  |
| --- |
| 機器視覺作業報告 |
| 擴增實境遊戲 |
|  |
|  |
| 姓名: 王心妤  學號: 00657127  日期: 2019/06/09 |
|  |

|  |
| --- |
|  |

# 方法

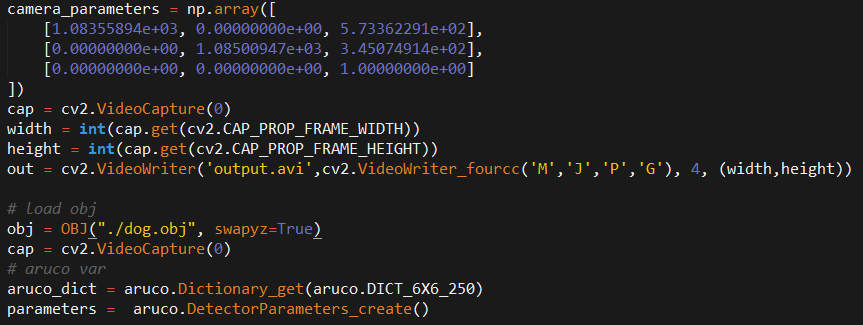
程式碼: <https://github.com/creek0810/2019_mv_hw2>

在這次的作業中，我將步驟分為main、calc\_projection及render三個部分。

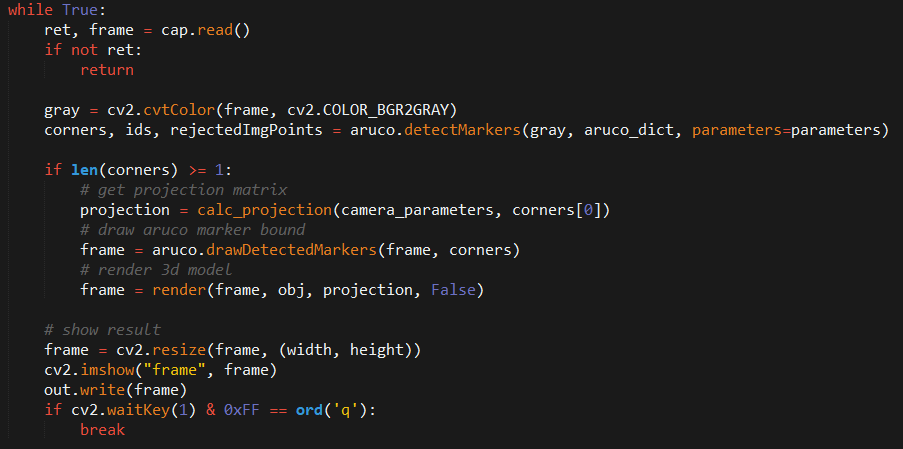
1. main

此function主要用來讀取當前的視訊frame、進行位置偵測、呼叫calc\_projection 與 render，及結果的顯示。

位置偵測的部分，我使用opencv中的aruco模組，在偵測到aruco marker後，將這些corner傳入calc\_projection找出包含z軸的transformation matrix。再利用此transformation matrix進行圖像的render。



將校正好的相機參數填上，並宣告之後要用到的變數

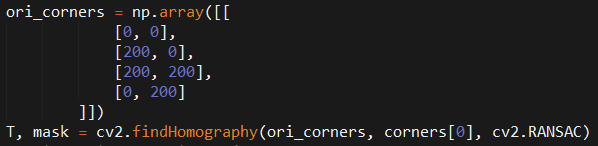


讀取視訊畫面，並進行偵測，若偵測到aruco marker則計算projection matrix，並進行render

1. calc\_projection

此function主要利用內部參數矩陣及homograpy，進行反推，得到真正的具備z軸的transformation matrix。

我們先自己定義原始的aruco marker分別的位置，再算出，由原始位置，轉換到目前圖片上所經過的matrix(T)



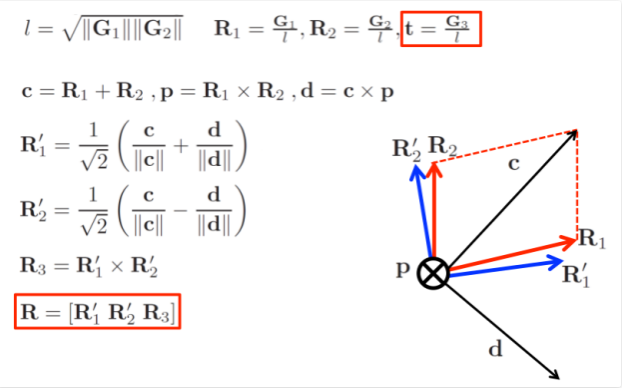
而我們知道T為內部參數矩陣與外部參數矩陣進行cross的結果，為了求得外部矩陣，所以我們可以將式子寫為

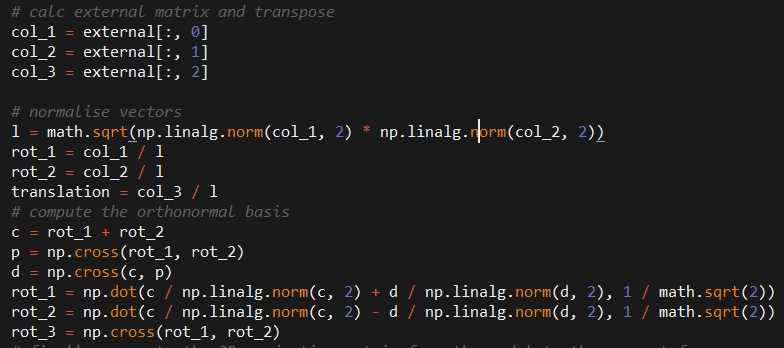


而由於我們找出T的方法是從一平面，轉到另一平面，所以我們找到的T僅能知道x及y的訊息，缺乏z軸。

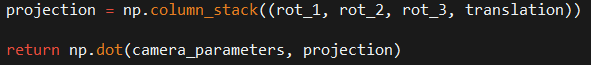
已知原始的rotation matrix為homogenous transform，所以rotation matrix的三個向量將互相正交，之後我們將利用此特性找出缺乏的第三個方向向量。

又因為我們找出的external為估計出來的，所以不能保證rotaion matrix的第一個向量(R1)正交於第二個向量(R2)，所以我們將對此進行一些處理，如下圖所示





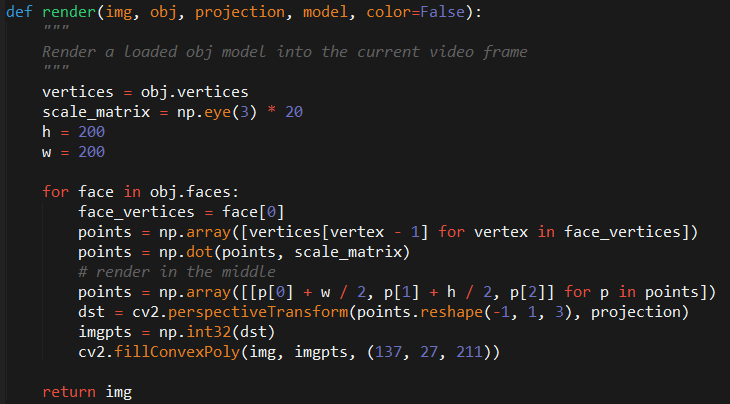
對R1及R2進行完處理後，算出R3。我們把rotation matrix(處理過後的R1、R2及R3)與tranlation matrix組合起來。將camera\_parameters與我們算出的transformation matrix做dot，得到projection matrix。



1. render

此function主要利用算好的projection matrix，將3d model貼上原圖。

我們先將points乘上scale\_matrix，對該model進行適當的縮放，並利用perspectiveTransform及projection matrix對這些points進行坐標轉換(貼到原圖時的位置)，最後，將算出的點圍出的區域著色(fillConvexPoly)，貼上原圖。



# 結果

<https://www.youtube.com/watch?v=WiWQGJ9rgZA&feature=youtu.be>



# 結論

透過實作擴增實境的遊戲，我對相機模型，及各種坐標系的轉換有了更深的理解，同時，也理解到了，線性代數的重要性(對於各個矩陣的名稱還是有點混亂，如果有寫錯的地方，還請老師指正)。

一開始有了做這個遊戲的想法後，真的很茫然，完全不清楚該怎麼實現，幸好網路上有很多的資源可以參考，才得以完成此次作品。

比較可惜的是，由於時間不夠(我自己時間規劃不夠完善)，導致沒能完成原先預期的彩色圖片功能。

# 參考文獻

<http://www.pygame.org/wiki/OBJFileLoader>

<https://home.gamer.com.tw/creationDetail.php?sn=4093935>

<https://bitesofcode.wordpress.com/2017/09/12/augmented-reality-with-python-and-opencv-part-1/>

<https://bitesofcode.wordpress.com/2018/09/16/augmented-reality-with-python-and-opencv-part-2/>