

Project 2 (Image Stitching) Report

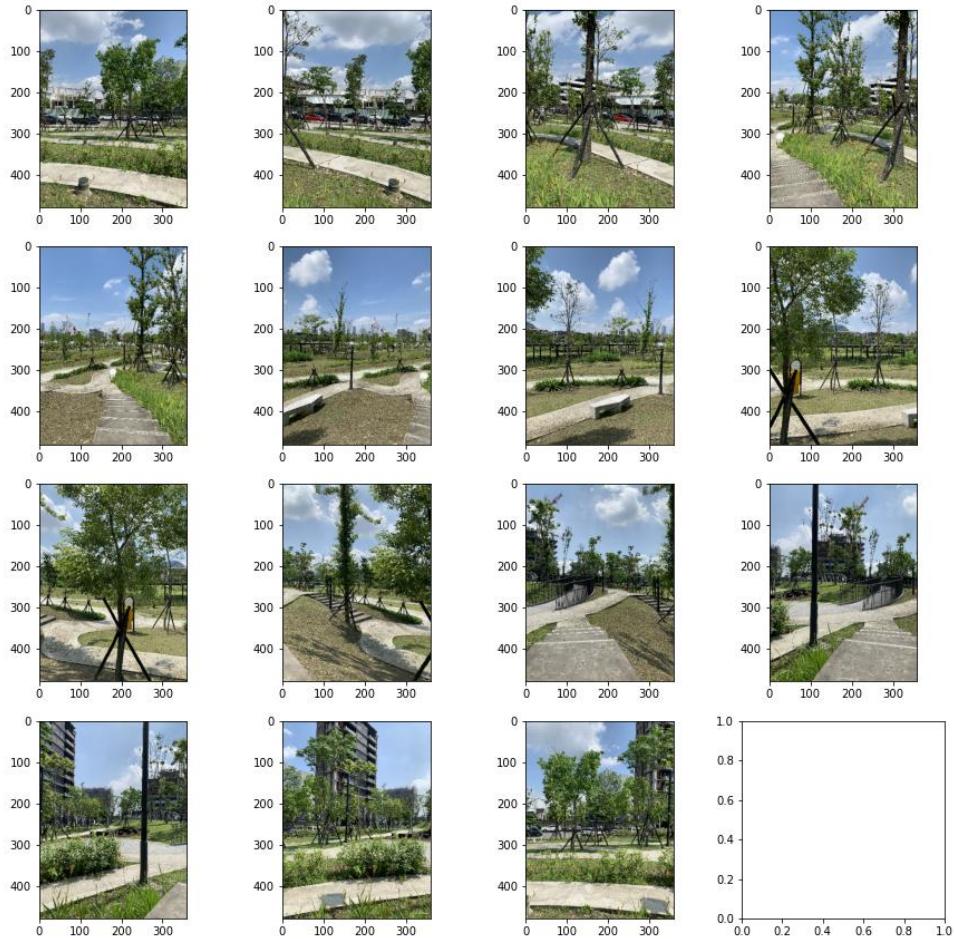
R10922124 林家毅 (Group 46)

大綱

- Capture photos & preprocessing
- Cylinder Warping
- Harris corner detector
- SIFT feature descriptor (orientation assignment & key-point descriptor)
- Feature matching
- RANSAC
- Linear blending
- Bundle adjustment (**Bonus**)
- References

Capture photos & preprocessing

這次 Image Stitching 是利用 iPhone XS MAX 拍攝的 15 張照片作為原始圖片，檔案格式為 .jpg format，如下圖。



為了方便後續 Image Stitching 流程，在讀取照片的時候先 resize 成高度 480 pixels 的同比例照片，並以 RGB 格式讀入。

Focal Length 的部分原本檔案就有儲存以公釐為單位的值，經下式可轉換成 pixel 單位

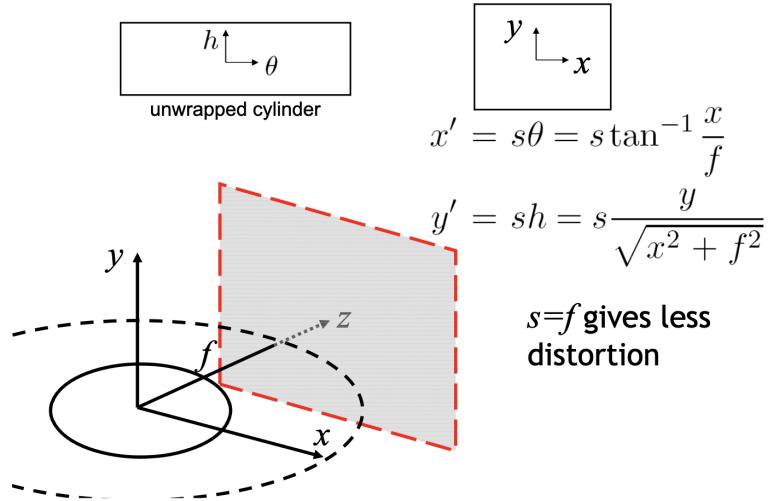
$$focal\ length(pixel) = \frac{focal\ length(mm) * image\ width(pixel)}{sensor\ width(mm)}$$

Cylinder Warping

對於這種要將全景圖組合在一起的任務中，如果先把所有照片投影到圓柱上，再把圓柱攤開來，雖然會有些微扭曲，但是每個視角都可以得到最好的資訊，因為圓柱上的視角跟真實世界的視角是很接近的。實作上是參考下圖，

Cylindrical projection

DigiVFX



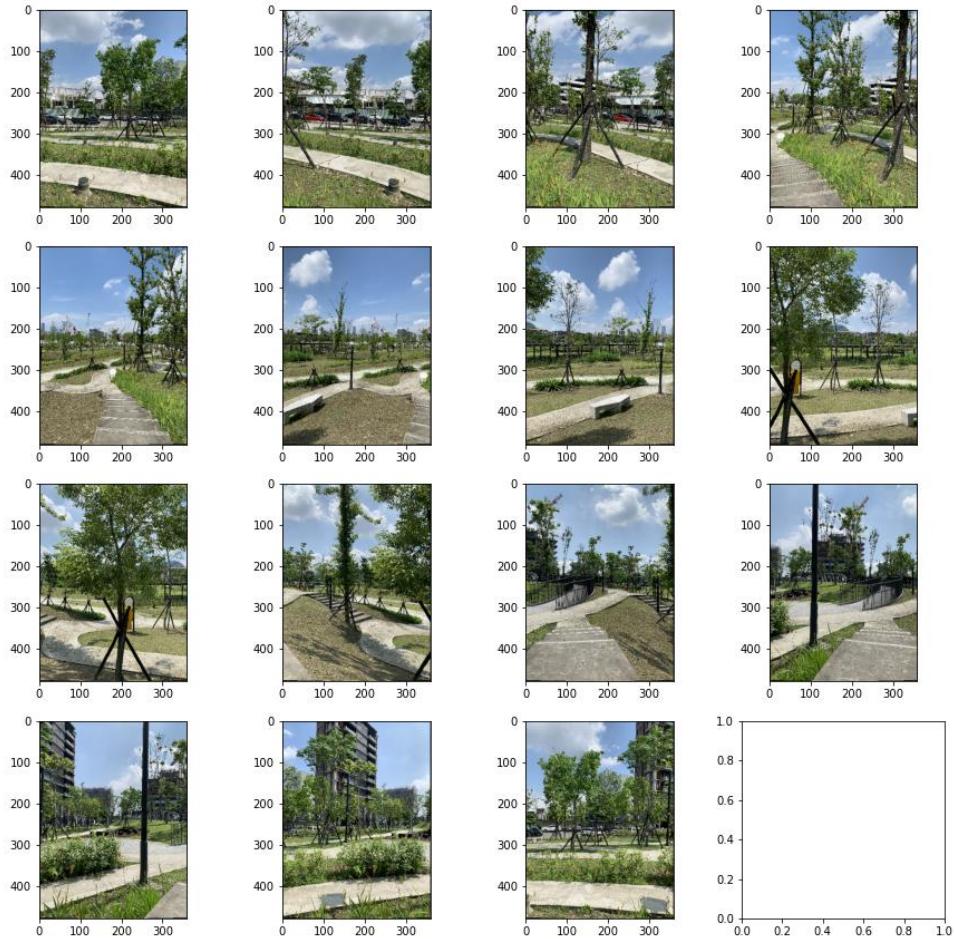
從圖中式子可知，假設對齊的中心點座標為 $(0, 0)$ ，則圓柱上每個點 (x', y') 會依據

$$x = f * \tan \frac{x'}{s}$$

$$y = \frac{y' \sqrt{x^2 + f^2}}{s}$$

從原圖的 (x, y) 位置取得 RGB 值放在圓柱上 (如果 (x, y) 對應的是無效的位置就都取 0)。

經過 warping 之後得到的結果如下圖



Harris corner detector

首先，對原圖 I 做一個 σ 值很小的 Gaussian Convolution 來 smooth noise，然後分別取 x, y 方向的 Gradient 得到 I_x, I_y ，接著，以 element-wise product 的方式算出

$$I_{x^2} = I_x \cdot I_x, I_{y^2} = I_y \cdot I_y, I_{xy} = I_x \cdot I_y$$

然後再分別對每個 pixel 計算 Gaussian kernel 範圍內的 weighted sum，

$$S_{x^2} = G_{\sigma'} * I_{x^2}, S_{y^2} = G_{\sigma'} * I_{y^2}, S_{xy} = G_{\sigma'} * I_{xy}$$

接著對於每個 pixel 可算出 corner response $R(x, y)$ ，公式如下

$$R(x, y) = S_{x^2}(x, y) * S_{y^2}(x, y) - (S_{xy}(x, y))^2 - 0.04(S_{x^2}(x, y) + S_{y^2}(x, y))^2$$

最後，對於每個 pixel，如果它的 corner response > maximum corner response * 0.06，且比周圍 8 個 pixels 的 corner response 都來得大的話，該點就認定為一個 corner，經過這個流程找到的 corners 即是 feature points。

SIFT feature descriptor (orientation assignment & key-point descriptor)

- Orientation assignment

為了找到每個特徵點的主要旋轉方向，會先算出一個 M 矩陣代表每個 pixel Gradient 的長度

$$M = \sqrt{I_x^2 + I_y^2}$$

以及 Θ 矩陣代表角度

$$\Theta = \arctan\left(\frac{I_y}{I_x}\right)$$

然後對於每個特徵點，會用一個 Gaussian kernel 約該 pixel 周圍的點的 M 值做加權，然後根據 Θ 選擇 36 個方向的其中一個做投票，值最大的方向以及值次大的方向(且值至少為最大值的 0.8 倍)都認定為 feature point 的 major orientation。

- key-point descriptor

為了找到每個 feature point 的 128 維 descriptor，會在 feature point 周圍的 16*16 範圍內將每個 4*4 block 利用像 orientation assignment 的投票機制，不過這次投票只會將所有角度分成八個 bins，再將這 8 個 bins 的總值經過 normalization、clipping、normalization 得到 8 維 sub-vector，最後對 16 個 4*4 block 都做相同的運算後即可得到 feature point 的 128 維 descriptor。

Feature matching

要對兩張照片的 feature points 做 matching 時，會計算不同照片所有 feature descriptors 之間的 Euclidean distance，若能滿足最近的 descriptor 小於次近的 0.8 倍，則代表成功對兩點做 matching。

RANSAC

我們希望用上個步驟的 matching 結果求出一張照片要接上另一張照片所需要的 global shift，所以會進行一個 RANSAC 演算法。這個演算法每次會抽 n 個樣本，然後算出 n 個樣本在不同圖片的位置差異平均值，作為假定的 global shift，然後在把這個 shift 套用在所有 match 的點上後，統計有多少點沒有完全吻合，透過多次執行這個流程 k 次後，找出誤差最小的 global shift 即可。

Linear blending

每當要把一張圖片加進 panoramas 的時候，可以先利用 panoramas 中已經有值的部分創造零一矩陣，新的圖片也創造零一矩陣，進而取出重疊部分，對於重疊部分，可以用 0 到 1 或 1 到 0 的 linear space 作為加權，以將 image 和 panoramas 的重疊部分做 blending，其餘沒有重疊到的地方就依照屬於 panoramas 或 image 的部分取值。

Bundle adjustment (Bonus)

將所有圖片連接在一起之後，會因為些微扭曲使得結果不是長方形，因此可以利用 `getPerspectiveTransform` 這個 cv2 的函數將原本的四角座標轉換到長方形的四角，其餘部分會順著做扭曲，最後即可完成 Image stitching，結果如下圖



References

1. [lec06_feature](#)
2. [lec07_stitching](#)
3. [Recognising Panoramas](#)
2. <https://github.com/qhan1028/Image-Stitching/tree/ae88b023ead6c86cb56cc216bd353fe7f9b260a0>
3. https://github.com/susan31213/VFX_Stitching