

# HW2 Report

---

B09902043 沈竑文 B09902040 洪郁凱

- 描述作業內容
  - 讀取多張相同位置・不同方向的照片・拼接組合成更廣角的全景照片。
- 實作演算法
  - Cylindrical Warping
  - Harris Corner Detection
  - SIFT Feature Description
  - Feature Matching
  - Image Matching
  - Image Blending
- 實作細節
  - Cylindrical Warping
    - 參考上課投影片內容的式子・由原圖座標 $(x, y)$ 得到投影後新的座標 $(x', y')$
    - 會做適當平移使得原圖中心得到的 $(x', y')$ 為 $(0, 0)$ ・這樣可以使得左右兩側的變形少一些、均勻一些
    - 將原圖的每個pixel value寫到cylindrical image的對應位置上
  - Harris Corner Detection
    - 生成灰階圖片並用Gaussian模糊去除雜訊
    - 對於x,y方向取Gradient
      - 利用x,y方向的梯度圖片・生出對應的 $I_{x^2}, I_{xy}, I_{y^2}$ 並加上高斯模糊產生對應 $G_{x^2}, G_{xy}, G_{y^2}$
      - 計算圖中每個Pixel位置的eigen value  $\lambda_1, \lambda_2$ ・並取得R值
    - 判斷該點是否為Feature
      - 在鄰近 $3*3$ 的Window中為Local Maxima
      - 且大於R最大值的0.1 (自訂的Threshold)即被認為是Feature
  - SIFT Feature Description
    - 根據公式・計算每個Pixel中的magnitude/orientation
    - 利用Feature鄰近區域的magnitude/orientation生成descriptor
      - 將附近 $16*16$ pixel區域切成 $4*4$ 個window
      - 每個window都將16組magnitude根據orientation分成10 bins加總
      - 將16window中的10維向量串接在一起・變成總共160維度的descriptor
    - 利用Descriptor・選出最長的方向及其長度當作畫圖用的方向
  - Feature Matching
    - 我們預設相機是由右往左拍・所以用前一張左半邊的Feature和右半邊的Feature比對
    - 利用160維度的descriptor作為matching的依據
      - 計算左右兩部分Feature點兩兩的歐氏距離
      - 根據距離排序・兩點距離越靠近越有可能是匹配的Feature

- 在幫某點match時，額外檢查其第一、二靠近的兩點距離比值要大於8:10
  - 若兩距離太接近，則不能斷定哪兩點是matching關係
- Image Matching
  - 計算Alignments時使用RANSAC演算法，避免錯誤的feature pairs對計算結果影響太大
  - 假設只有平移，因此每輪的參數便直接設為其中一組feature pair之間的位移
  - 使用inliner最多的一組位移當作最終的結果
  - 由左而右拼接圖片，放完一張後會將當前拼接起點加上當前這張照片與下一張照片之間的位移，確保圖片皆由正確位置開始拼接
- Image Blending
  - Linear Blending: 若是pixel位於非重疊區域，則直接使用原照片本身的value；反之，則將兩張照片該pixel value依x軸方向距離交界邊界的比例進行加權平均
  - Poisson Blending (bonus): 我的理解是，假設res為結果圖，在重疊位置 $res(i, j)$ 這個點對應到該位置的原圖(取其中一張)  $source(u, v)$ 。  
 $(4 * res(i, j) - res(i - 1, j) - res(i + 1, j) - res(i, j - 1) - res(i, j + 1))$  得到的這個gradient應該要與  
 $(4 * source(u, v) - source(u - 1, v) - source(u + 1, v) - source(u, v - 1) - source(u, v + 1))$  越接近越好 (若是鄰居超出合法範圍我則取自己的value當做一個padding)。利用這條式子去做 iterative optimization 得到一個比較好的 $res(i, j)$

- 實作結果

- 合成圖片：

- Linear Blending:

- Parrington:



- Ours:



可見雖然圖片有順利接合起來且還算自然，但linear blending使得部分區域有鬼影問題產生。此外也能成品觀察到一路向右下斜去，drift問題嚴重。

- Poisson Blending:

- Parrington:



實作上可能有些小細節還沒處理好，因此邊界效果還沒有很好，但可以見到其他中間重疊區域變得清晰，比起linear blending的效果更好。