VFX Project #2 Report

a10409007 劉健彬

編譯環境

Matlab R2016b + Image Processing Toolbox

實作內容

1. Taking Photos on a tripod

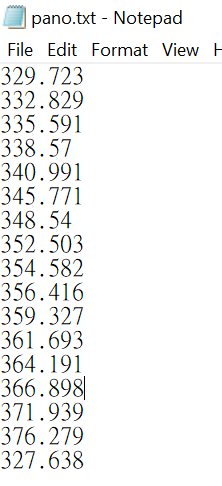
使用設備：Cannon 6D / 腳架

使用腳架固定相機在一個位置，然後利用腳架上的角度參數旋轉相機，每旋轉間隔大約20度，拍一張，總共拍了17張（1824 x 2736）。

1. Warp to cylindrical coordinate
2. 讀取照片

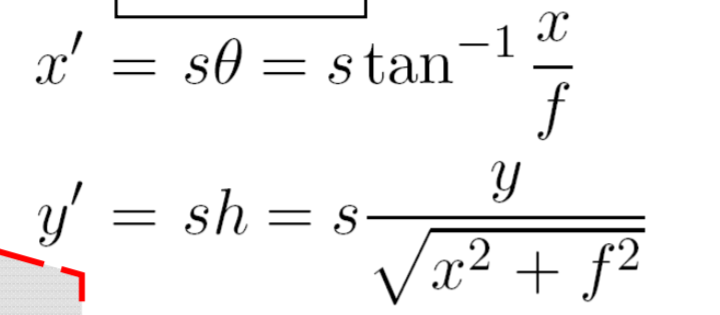
首先讀取一組17張的照片。

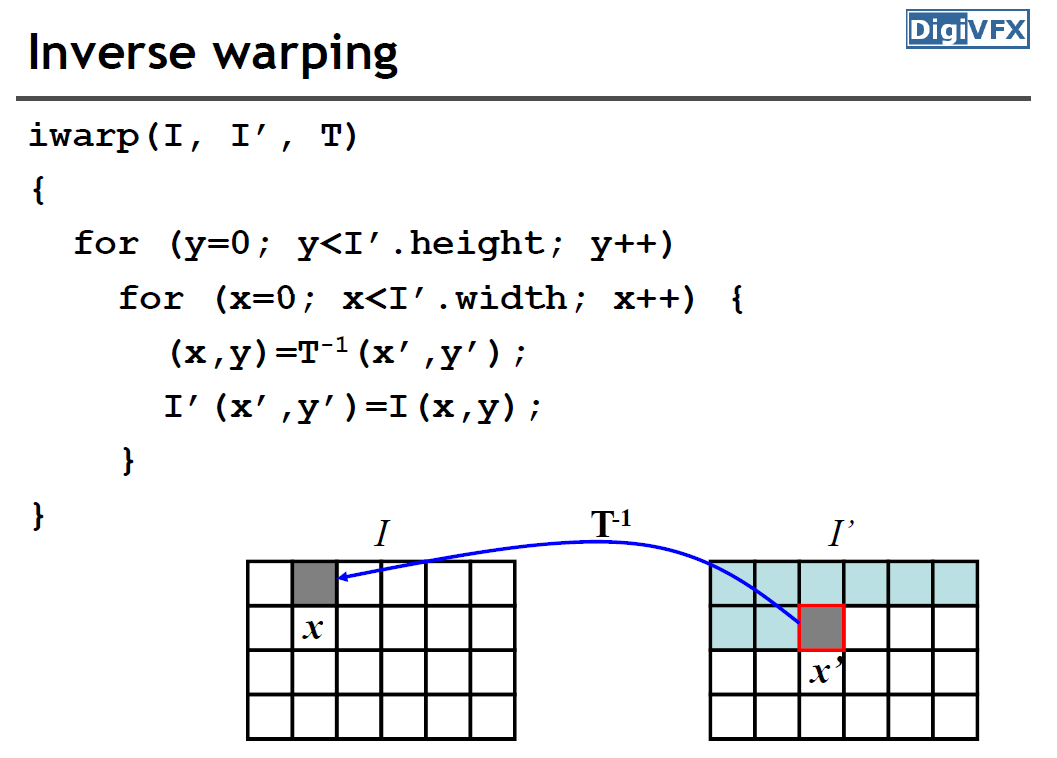
1. 獲取每張圖片的focal值

通過autostitch (32bit windows version)軟體，讀取拍攝的17張照片，得到pano.txt，裡面記錄了估算的每張照片的焦距值。將多餘的訊息刪除，僅留下每張照片的焦距值，並用換行符相隔（如右圖）。

1. 將原來座標各點資訊對應到圓柱座標上

直接用拍攝的照片進行image stitching的結果會是Rectilinear Projection，不是很符合人的習慣，所以要將原本的每一張照片進行Cylindrical projection。通過投影片裡提到的公式以及演算法，將原來照片每個像素的XY座標轉換到圓柱座標上。





1. Feature Detection

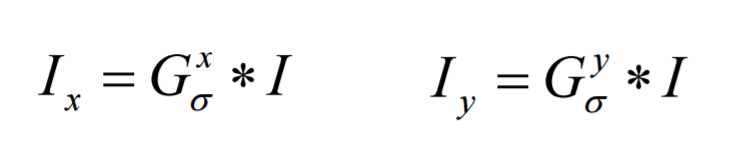
因為拍攝的照片不涉及到會發生scale的情況，所以採用相對其他兩種比較沒那麼複雜的方法，Harris corner detector.

1. 將照片轉成灰階圖

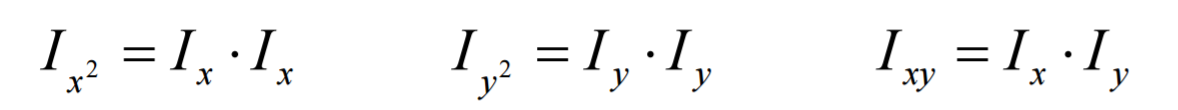
將每張照片都轉為灰階圖。

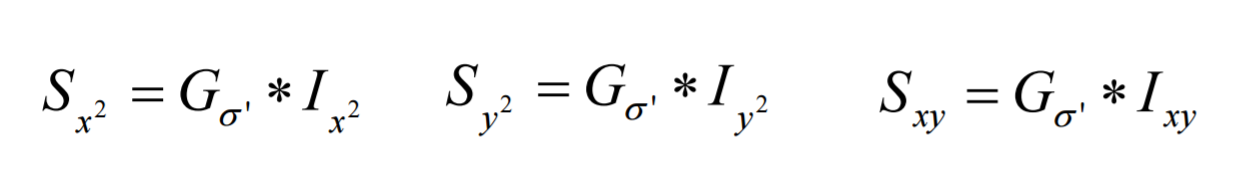
1. 減少noise，計算x和y方向的derivatives

為了減少noise，對每張圖做gaussian filter，並且根據方程式計算x和y方向的derivatives。

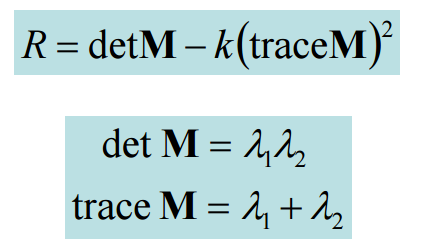


1. 計算每個像素的xy方向的乘積

根據方程式，計算每個像素的x和y方向各自的乘積以及xy的乘積。

1. 根據方程式，對每個值進行gaussian filter
2. 求出R

為了決定怎樣的特徵值λ1和λ2才是夠大的，定義Response R。

k的取值範圍是0.04~0.06，經過測試，在自己拍攝的照片中取0.04效果相對比較好。並且根據方程式及上一步驟中算得的值算出R矩陣。

1. 設定threshold

對R設定一個threshold,超過這個值的pixel就抓取下來，因為這個pixel就有可能對應到corner。

1. Feature Description

通過上一步驟Feature Detection之後，找出了feature在哪裡。為了下一步能夠比較兩兩圖片之間的feature，所以要對feature進行描述。

取了上下左右各2個neighbor，形成8維的vector，作為feature points的Descriptor.

1. Features matching

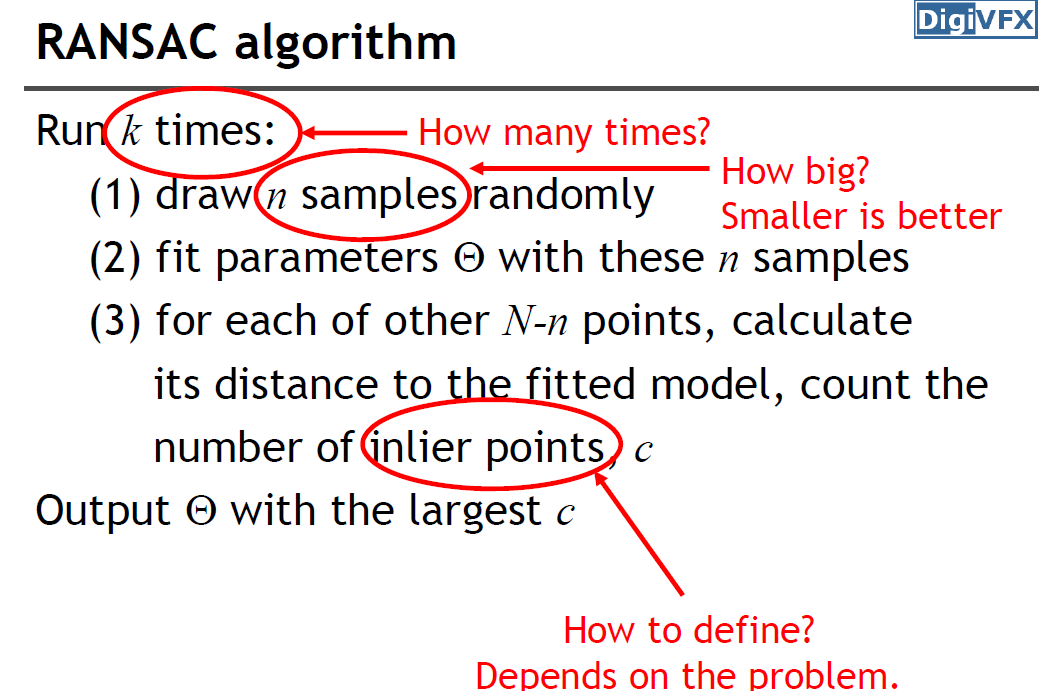
有了上一步驟取得的Descriptor之後，接下來就要比較feature之間的相似程度。

1. 計算兩張照片feature之間的距離

用距離來表示兩張照片之間的相似程度，距離越短則代表相似性越高。

1. RANSAC演算法

找出來相對應的features，不一定都是inliers，如果直接計算的話，如果裡面有一個差異很大的outlier，則算出來的model會被影響。所以在這裡要利用RANSAC演算法去避免Outlier.

根據投影片中的演算法，重複run k次，每次都隨機從裡面取出n個點，利用這n個點估計Θ，接著把剩餘的點（N-n個）再帶入新的model裡算出有多少點是inlier。重複k次之後能讓inlier個數最多的Θ值就是算出的解。

其中涉及到k和n兩個參數值的取值問題。根據關係式，p是真正inlier的機率，而P是演算法跑了k次之後成功的機率。在固定n值的時候，p越大,k則越小。相反固定p時，n越大，所需要的p就越大。而p通常未知，所以n取小一點比較好。所以在程式中將P值設為0.999，n值設為3。

1. Images matching

利用inlier match中的feature對應的座標做image matching. 透過Model Ax = b計算位移量，即圖片1要位移多少才能和圖片2接合。

1. Images blending

將兩張照片拼接的時候往往會因為拍攝的角度或是光源問題導致顏色上有無法，所以要做blending. 程式中使用了線性的方法進行Blending，使兩張圖的接合處不會太過突兀。

處理過程及效果討論



結果圖

**解決的問題**

1. 直接拿拍攝的照片跑程式會非常慢，因為圖片太大，所以後期為了方便測試將所有圖片的尺寸從1824 x 2736轉到300x450。
2. 有些圖的feature非常多，常常集中在某個角落，常會發生在一些例如草地之類的場景，導致feature太多程式跑很慢。所以試著做了一下Non-Maximal Suppression，不過最後效果還不算特別好。

**仍然存在的問題**

1. Drift問題沒有解決，最後拼接的照片會逐漸地往上移。
2. Blending沒有做的很好，圖片交合處仍然很明顯。整體色調和亮度也一致。

References

* 課堂投影片
* M. Brown and D. G. Lowe, Recognising Panorama, ICCV 2003.
* Chris Harris, Mike Stephens, A Combined Corner and Edge Detector, 4th Alvey Vision Conference, 1998