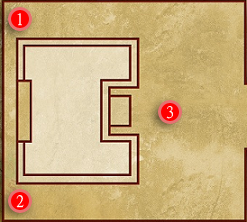
|  |
| --- |
| **Digital Image Processing Final Report** |
| **View transition Effect** |
| Team 10- R01944040 柳成蔭．R01922125 吳儒涵．R01942095 簡家豪 |
|  |



簡介

　　這次的期末專題，我們要做的是遠近場景之間的view transition animation，希望能夠做出較擬真且有行進感的轉場動畫，為了到要達到較真實的效果，我們藉由計算視野前進時場景的相對運動，將畫面中的每個牆面一一分開計算來達到較細緻的效果



設計方式

在解決問題的過程當中，我們試過許多方法，如直接使用morphing以及助教給的參考論文提到的view morphing的做法，但在直接將近景與遠景影像作morphing之後會發現，因為兩張圖片相差太大，有許多無法從遠景對應至近景的地方，會使得原本應該是直線的牆面及牆角有扭曲的情形發生，效果相當不佳

|  |  |
| --- | --- |
| 圖１遠景 | 圖２近景 |



圖３.　圖１與圖２morphing之結果

而view morphing作法則是在深度差異較大的兩個view時結果也不甚理想，不適用於我們想要呈現的效果。

所以最後我們使用的方法為將遠景與近景各個牆面分開計算，以近景和遠景對應的feature points之間的線性轉換關係來計算各牆面在下一個時間點的移動軌跡，並以此轉換關係使用OpenCV函式來算出中間內插出來的frame，製作出30fps的轉場影片。

實作方式

**Step 1:** 找出近景與遠景的對應特徵點，一個場景約找出15～20個特徵點



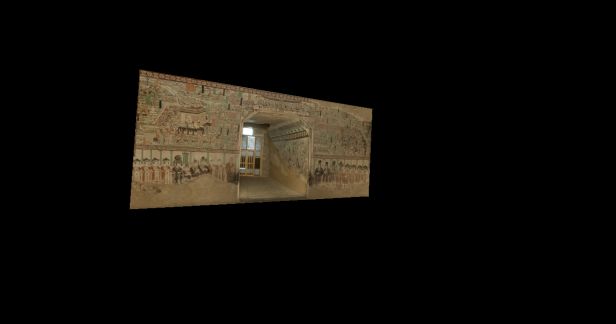
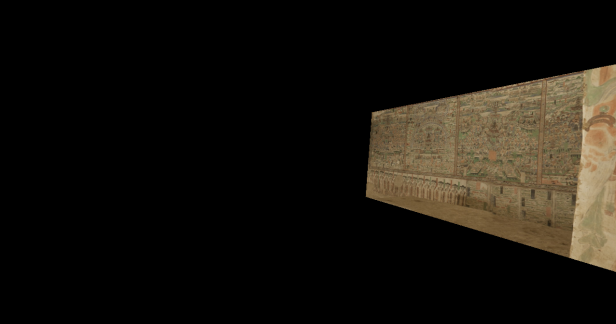
圖４.　遠景

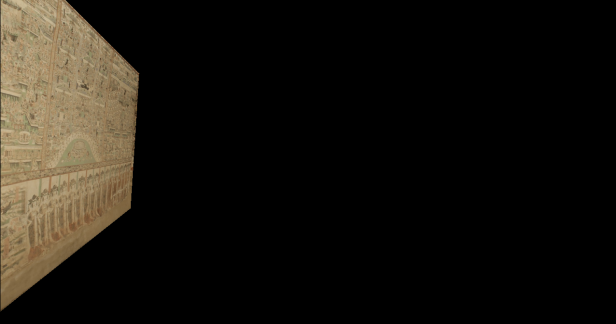
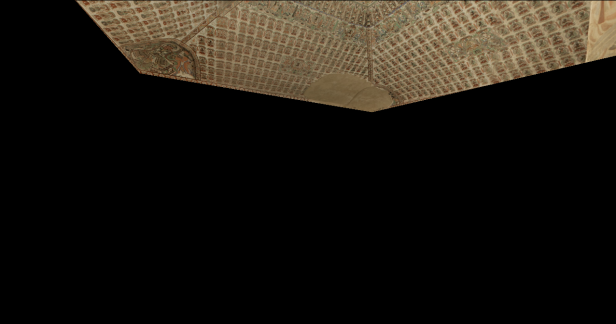


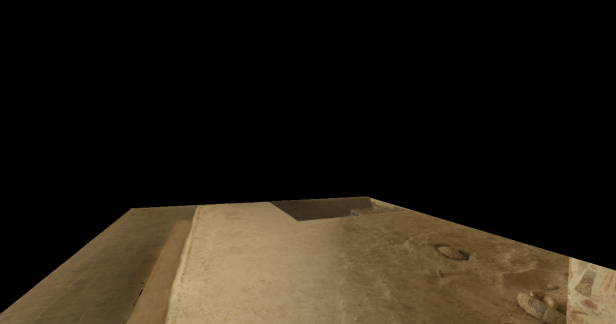
圖５.　近景

**Step 2**: 找出四組適合的對應點利用**getPerspectiveTransform**()算出兩圖之間的perspective transform，兩張圖皆有出現的牆面以對應點來做，但是若近景圖沒有出現的牆面，如圖４的天花板及地板，則要自己估算在近景時對應點的座標

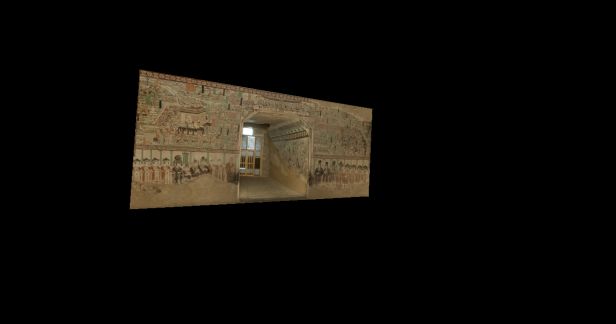
**Step 3**: 將圖片的牆面一一切割

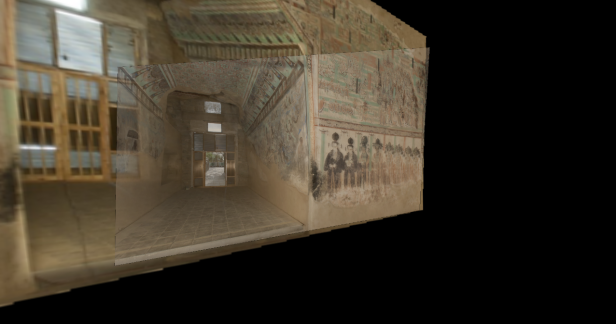


**Step 4:** 將切好的牆面，還有以**getPerspective()**算出的transform matrix，利用**warpPerspective()**做出perspective transformation











**Step 5:** 將每個牆面做好的perspective transformation結果相加，合成完整的場景

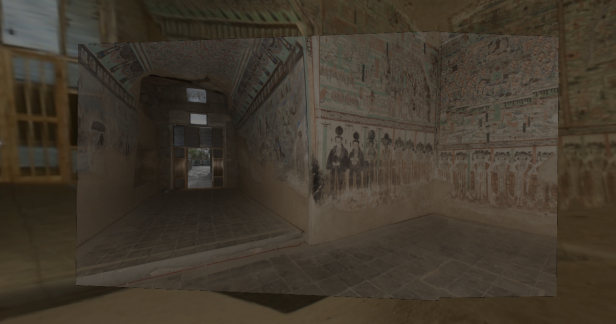
結果













心得

　　這次在實作時遇到了一些困難，主要是因為測資兩張圖之間遠近深度差異很大，要根據兩張圖片就做出真實的中間轉場效果實為不易，在一開始試過許多其他方式，如直接morphing，也考慮過view morphing的方法，但都做不出我們想要的結果，直到最後才使用費工的分牆面做warping，過程中花費許多時間，許多對應點都要自己推估，還有為了整體畫面精緻真實，分牆面一一去做，一個場景可能就要做出４～６面牆再合成，所以最後時間上有些來不及而無法顧慮到美化或是一些特效 ，像是一開始如果能夠先把遠近景兩張圖的色調及亮度調到接近，做出來的效果會更加順暢自然，這是這次結果較可惜的地方。