計算機圖形 Final Project：
Image Morphing

B05902059 廖威仲 B05902011 梁振寧 B06902062 陳法熏

目錄

題目敘述

影像變形（Image Morphing），是由一張圖像流暢地變成另一張圖像的視覺效果。最著名的例子就是 Michael Jackson 的 Black or White 中所使用到的技術。而這個效果說穿了概念就是將兩張照片相對應的點 Match 起來，並且轉換過去，並不複雜，但是根據使用的演算法不同可以達到非常不同的成效。而我們這次期末作業想基於此視覺效果上，開發 GUI 套件，來方便我們進行多張照片的 Image Morphing，並且將這些照片變形的過程組合成小短片。

使用方法

一、紀錄特徵點：

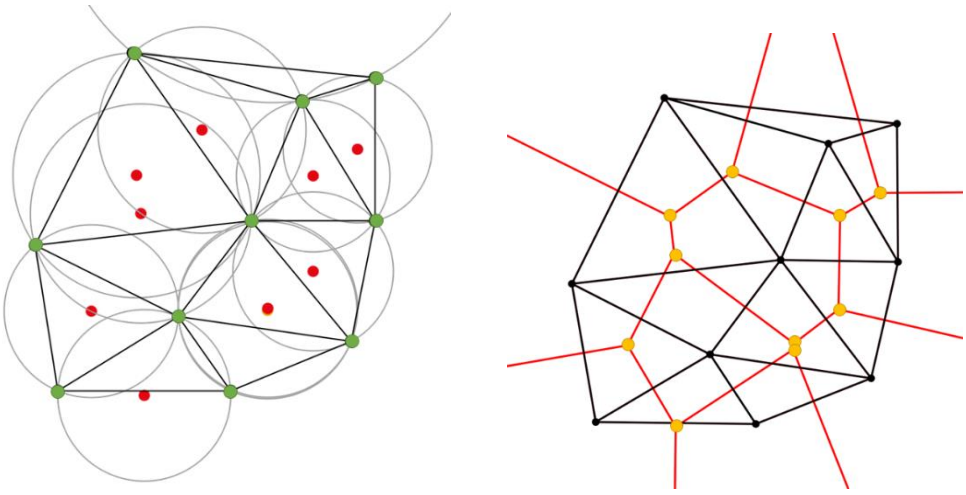
這次我們使用的程式語言及套件為 Python 以及 OpenCv2、scipy 和 Numpy。我們原先考慮使用 CV 的特徵點偵測來找出兩張圖片的特徵點，再去做後續的操作，但是後來發現兩張圖片的特徵點可能沒辦法那麼 Match，因此我們讓使用者自己去點特徵點。並且可以讓使用者自己決定說哪兩個點要互相 Match。點完相對應的兩個特徵點之後，程式就會將這對特徵點存起來。

點完特徵點後，我們會將圖片根據這些特徵點去組成許多三角形。 然後再用兩張圖片的三角形去做轉換。

圖一、示意圖

二、由特徵點組成三角形：

我們使用了 scipy 套件包來實作 Delaunay triangulation（德勞內三角化）。 這個方法能最大化三角形中的最小角角度，避免出現極瘦的三角形破壞 morphing 的過程。德勞內三角與 Voronoi Diagram（沃羅諾伊圖）互相是對偶關係，其特點是這些線段所構成任一三角形的外接圓，其圓內沒有任何其它的點在裡面。



圖二、Delaunay triangulation 以及 Voronoi Diagram

首先特徵點的資料會以兩個 List of points 的資料型態讀取進來，兩個 List — ListA 以及 ListB，分別對應起始圖 A 以及終點圖 B。

> List_of_points = [[X_i, Y_i] for i from 0 to N]

再來將 ListA 進行 Delaunay triangulation，並將每個三角形中的頂點所對應到其在 List 中的 index 值記錄下來

```
> Triangles = Delaunay(ListA)
> Index = []
> for Tri in Triangles:
>  List = []
>  for point in Tri.points:
>        List.append(ListA.find(point))
>  Index.append(List)
```

最後依據其 Index 順序，重新排列、組合 ListA 以及 ListB，將兩個 List 傳到程式下一個階段。

```
> ListA = ListA[Index]
> ListB = ListB[Index]
```

理論上最合適的三角形組合的做法，應該是依據圖中物件的性質來做三角形配對，這個目標可以利用 Machine Learning 來進行實作。這次的 project 我們就沒有將重點放在這邊了。

三、三角形的轉換：

有了兩張圖初始的三角形表後，首先要依欲產生圖的數量，計算出各個階段的三角形表。這邊採用的是線性變換的方式(式一)。在每個階段中，對於各對應的三角形對 (a, b, c)-(d, e, f)，平面上的任意點都能用頂點 a 與 ab、ac 向量的線性組合表達(式二)。

式一: (0, 0, 0) - (1, 2, 5) - (2, 4, 10) - (3, 6, 15) - (4, 8, 20)

式二: (x, y) = (a.x, a.y) + u*(b - a) + v*(c - a)

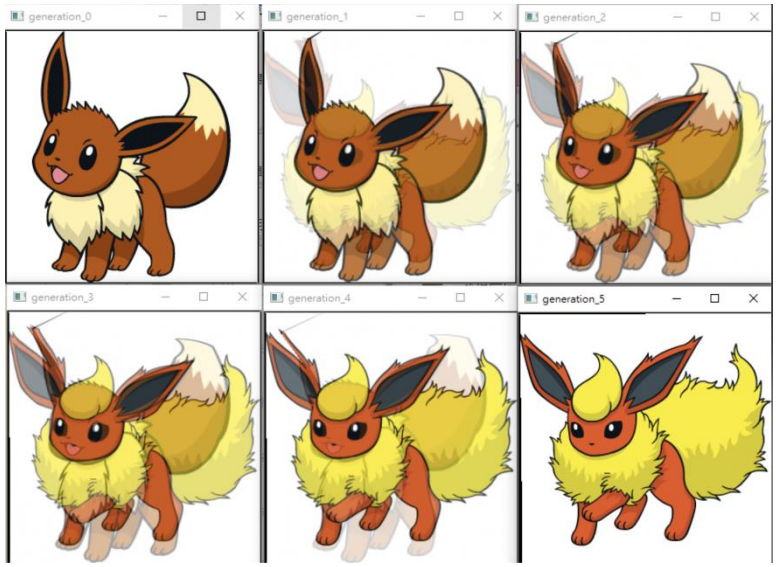
計算出係數 u, v 後，不僅能判斷該點是否在此三角形內，亦可算出在另一張圖的對應點位置(式三)。由於 (x', y') 不一定是整數，我們使用線性內插的方式，計算對應的距離比再由四個格子點取值。

式三: (x', y') = (d.x, d.y) + u*(e - d) + v*(f - a)

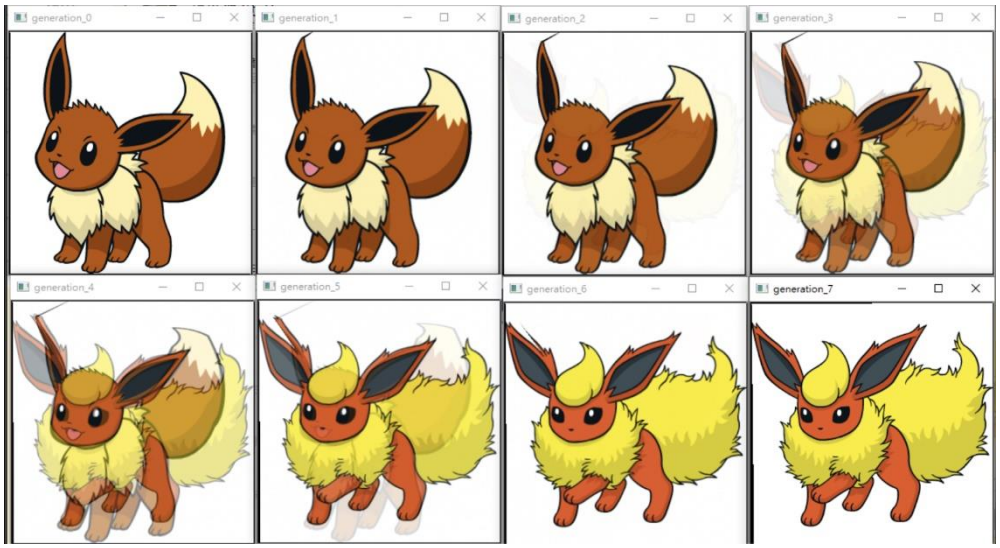
計算出各個階段的變形圖片後，最後一步是找出適合的方式將他們組合起來。一開始使用線性內插的方式但效果不佳，有超過一半的階段會有明顯的鬼影。於是我們改用常態分布的累積分布函數 (CDF) 來當作組合圖片的參數，大幅減少會出現的鬼影的階段數。

成果及討論

在呈現上，我們有實作許多版本不同的成果（i.e. 由 morphing 製作的動畫短片）。在報告內僅挑選靜態結果來呈現：

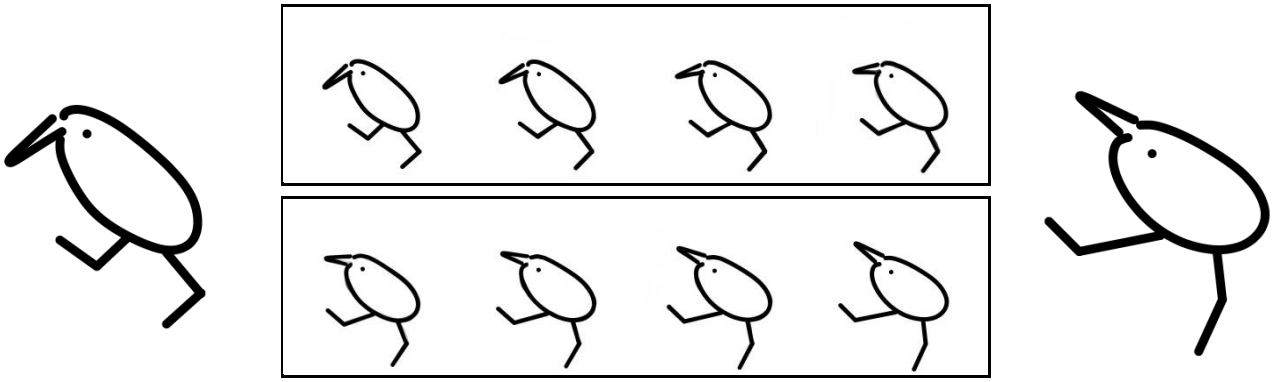


圖三、使用現性內插進行混合照片

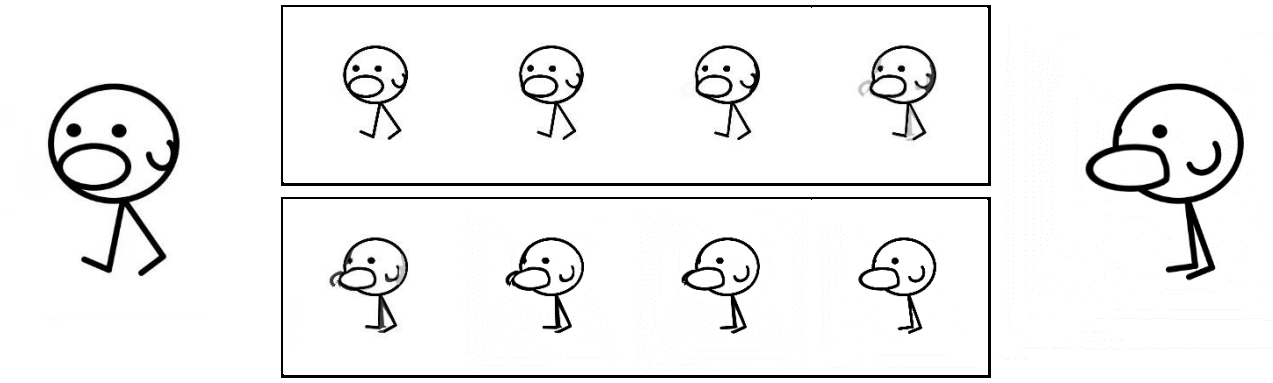


圖四、使用常態分佈的 CDF 進行混合照片

比對以上兩張圖的成果，最初使用線性內差時除了兩張原始照片外的每個階段都有鬼影；在改為使用常態分佈後只有中間階段有明顯鬼影產生。



圖五、使用了簡單的圖像進行混合照片



圖六、嘗試對較複雜動作的圖像進行混合照片

我們嘗試自己繪製更簡單的圖像來進行。從簡單的圖片著手我們能製作出幾乎沒有鬼影的連續圖片，然而在嘗試加入更為複雜的動作以後（併攏的雙腳以及將臉轉向），這些較為複雜的部分會無可避免的出現破碎的圖像及殘影。

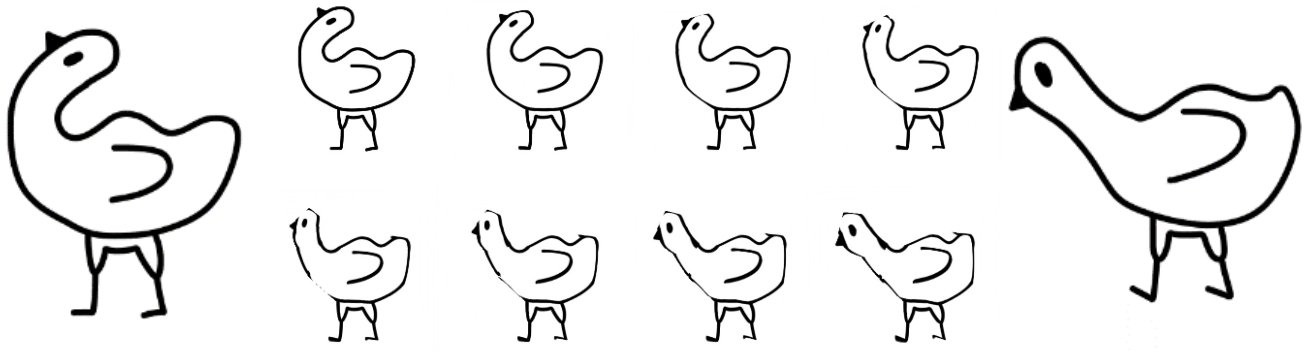
經過前面的嘗試，我們知道使用滑鼠游標來取到的特徵點十分容易破壞圖像（e.g. 破碎的伊布耳朵、破開的鳥喙）；此外，當圖像有大量特徵點擠在一起時，對使用者來說在視覺上會產生障礙，會非常困難用滑鼠游標來點選。

延伸

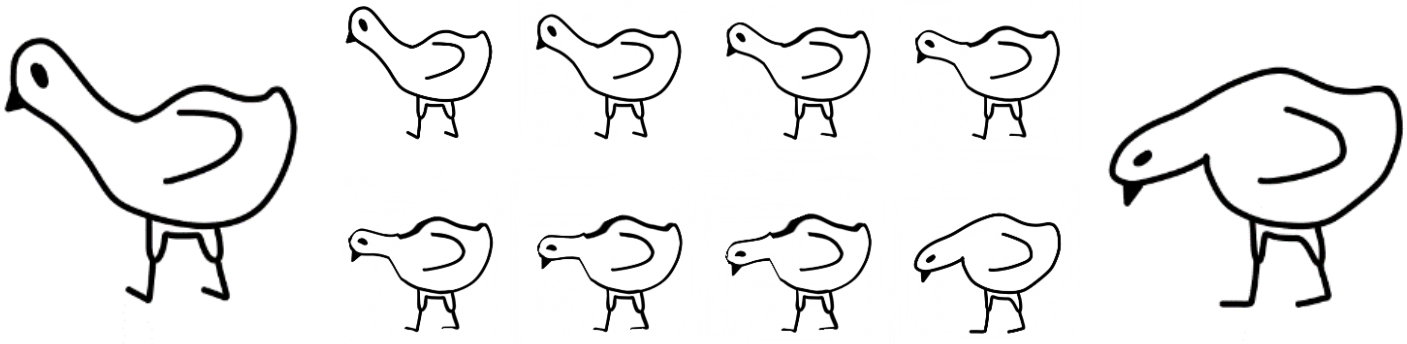
我們想要用另一種向量法來結合現在所使用的特徵點法，希望透過混和兩種計算方式，消除各自缺點的同時，保留各自的優點。

實際做法如下：在三角形的部份跟原先僅特徵點的作法一樣，但是需要將向量經過的三角形移除，空出的區域再交由向量方法來計算，希望透過這樣的方式達到原先三角形無法達到的扭轉效果。

由於時間緊湊，成果不到十分滿意的結果，僅挑選一些如下所示：



圖七、第一次嘗試，僅使用一個向量來描述脖子的轉動，效果比三角形時好



圖八、使用向量描述脖子轉動，可以看到接合處較為不平滑

雖然加上向量後處理彎曲的效果有提升，但現在的方法只是單純的混和，而非讓兩個方法交互影像, i.e. 向量的部份由向量和三角形來共同決定，依權重分配決定的程度。希望以後能實作出以特徵點為主，又能用向量來表示轉動的演算法。