國立交通大學

資訊科學研究所

碩士論文

以 Active Shape Model 為基礎的 3D 物體變形演算法

3D Objects Warping Algorithm Based on the Active Shape Model

研究生：劉方正

指導教授：荊宇泰　副教授

中華民國九十三年六月

以 Active Shape Model 為基礎的 3D 物體變形演算法

學生：劉方正　　指導教授：荊宇泰

國立交通大學資訊科學研究所

摘要

現今，作影像變形的方法有許多，但最主要還是以 Registration 的方法為主，因此在實作方面難免會遭遇到使用複雜性與效率性等等方面的問題；本論文提出根據 **Statistical Models of Appearance 的方法發展出**物體自動化變形系統的方法；我們稱我們的方法為 Active Shape Models for Warping（ASMW）。並將該方法應用於果蠅腦中 mushroom body 及其周圍組織以及神經 warping 的實驗。

關鍵字：Active Shape Models, ASM, Active Appearance Models, AAM.

3D Objects Warping Algorithm Based on the Active Shape Model

Student : Fang-Zhang Liu Advisor : Yu-Tai Ching

Institute of Computer and Information Science

National Chiao Tung University

Abstract

Recently, there are many ways to do image warping. Most of them were based on Registration. The registration based approaches need many user assists and thus are not automatic methods. In this thesis, we developed a 3D warping algorithm based on the Active Shape Model (ASM). We call the proposed method the “Active Shape Models for Warping（ASMW）”. The method was applied to warp between mushroom bodies of fruit fly brain.

Keyword ： Active Shape Models, ASM, Active Appearance Models, AAM.

誌謝

本篇論文能得以完成，首先要感謝我的指導教授 荊宇泰老師的指導。由於他的專業知識傳授及持續的幫助和指導，使的我得以順利完成碩士論文。同時還要感謝實驗室博士班林志陽學長的大力幫忙及實驗室其他同學與學弟妹們的協助與建議。在此特別感謝周樹偉學弟提供我所需要的中軸資料。感謝大家陪我度過這段快樂的時光。

最後感謝我的父母、家人以及朋友們，有你們的支持與鼓勵我才能順利的完成學業。感謝在此變形領域中默默努力研究的每一位學者，由於他們所提供的寶貴研究知識我才得以有今日的成果。感謝大家。

目錄

圖表目錄

第一章　序論

1.0 研究背景：

人類大腦是目前世界上已知自然系統中最複雜的一個東西，其複雜可與最錯綜複雜的社會和經濟結構匹敵，甚至超越。它是科學的新領域，美國國家衛生研究院在一九九○年時宣布這是「大腦的十年」，就像二十世紀前半部被宣稱為物理的時代、後半部被稱為生物的時代一樣。所以，二十一世紀的開始被稱為「大腦與心智科學的時代」（the age of brain mind science）。

大腦的研究是一項非常艱深且不容易探測的學問，例如當人類大腦中的某一塊區域受傷時我們該如何判定；另外當人類受刺激時，腦中的哪些部分的神經發生變化以及對於記憶這件事對於腦的影響等等都是現今我們所有人研究的重點。

為探索人類腦神經網路的作用機制，科學家已經開始利用酵母菌、線蟲、果蠅、及小白鼠 等已經完成基因體定序的少數模式生物進行比較研究，果蠅有學習及記憶的能力，是研究腦功能最簡單的模式動物。腦學習及記憶的基因，多數都是先從果蠅的研究得知，而人腦的各種主要疾病，例如帕金森症、愛滋海默症、杭氏跳舞症等，也已累積果蠅的研究模式系統[13]。因此我們*此次的研究目標以鎖定果蠅腦為主*，在這種研究中需要將功能性的資訊映射 (mapping)到解剖上的位置, 因此建立一個標準腦以及將一個個別的腦變形(warp)到一個標準腦是一個必經的過程. 本研究試圖建立一個標準果蠅腦中的 mushroom body, 並將一個果蠅 mushroom body 變形至一顆標準果蠅 mushroom body ；因為本研究主要針對 Mushroom body 上的神經變化，因此在變形上我們只在乎其周圍的神經變化。

一開始本文主要會針對兩種方法來作討論，其一為 ASM（Active Shape Model）演算法，另外一種為 AAM（Active Appearance Model）演算法；此兩種

方法都是由 Cootes 等人於 2000 年所提出。就作者本文應用方面，ASM 的方法較常用於 Segmentation，而 Image Warping 的方式則以 AAM 為主；在下一章節中將簡單介紹一下兩者的方法與差別，以及為何本文會以 ASM 演算法做為變形系統依據的理由，至於方法細節請參考[1]原作者的文章。

1.1 研究動機

利用 ASM 演算法做變形的好處有下列幾項：

A. 自動化（Automatic）：我們的第一個訴求是希望能發展一套自動變形的系統；現今雖然存在許許多多種變形的演算法，如 Registration，但是很少方法能提供自動化；因此為了以自動化為目標，便以 ASM 演算法為主，因為只需利用中軸（Centerline）就可以完成整個變形的工作；原因在於只需要計算出兩組中軸的變化量，便可以輕易的將影像變形；程式的變形方式，大致分成兩部分，一部分是 Global 變形，另外一種則是 local 變形；Global 變形（稱之為 Transform）方式是利用單純的位移（Translation）、縮放（Scaling）以及旋轉（Rotation）達成目的，而且現今對於 Transform 的計算方式有提出許多種自動計算的方法；而對於 local 方面，ASM 則有提供一種簡單且自動的方式就可達成 local 的調整；因此利用本文的方法，使用者只需要給系統關於中軸以及影像(或 Volume data)等資料（欲測試），經過系統的計算後，便可以在短暫的時間內得到兩者的變化量，而不需要使用者介入設參數等等，提供了一套完整的自動化。

B. 效率性(Efficiency)：因為 ASMW 只需要計算到中軸的變化量，因此將可以節省掉許多時間；在比對的過程中，程式真正花比較多時間的是在將影像資料作變化的步驟；在 2D 實作上，其所花的時間非常的少，但是在 3D 上，因為它是一個三維空間的資料，可以想像它是由幾十張甚

至於上百張的 2D 影像所組成，因此在作變化的過程，系統的負載相對的較重，因此時間上花費較多。

C. 正確性（accuracy）：因為比對的資料是以中軸為主，較具代表性，所以在實驗結果上也較合理，當然正確性較高。

ASM 的方法並非沒有缺點，它的缺點之一在於使用者必須在事前先 Training 出一個 model，之後才能利用此 model 來計算出變化量，以達成變形的最終目的。

1.2 論文架構

本論文共分為下列五個主要章節

1.2.1 序論：主要介紹關於本文的研究背景、研究動機、論文架構。

1.2.2 ASM 與 AAM 之介紹與比較：本章節主要介紹關於 Cootes 等人所提的 ASM 與 AAM 的演算法，並且比較兩種演算法的差異。

1.2.3 ASMW(Active Shape Model for Warping)：本章節以介紹本文如何以 ASM 演算法為基礎來達成變形的目的。

1.2.4 實驗結果與討論分析：本章節主要介紹與分析實驗的結果。

1.2.5 結論與未來研究方向：本章節主要在介紹對於未來的研究方向，以及對於未來程式仍需加強的地方。

第二章 ASM 與 AAM 之介紹與比較

2.0 背景以及動機

近幾年來，ASM 與 AAM 的方法漸漸受到注目，並且應用的範圍也越來越為廣泛，而當初創造它們的動機主要在於解決 Image Understanding 方面的問題；所謂 Image Understanding 是以重建(Recovering)影像並且了解其背後的意義(To know how it represent）為主；在過去的實驗中，model-based 的方法應用於這一類的問題還算成功；但是它仍有一個問題，問題在於對於重建（Recovering）影像的正確性方面，例如當一張影像有遭受到雜訊的干擾或者是一張不完整的影像時，如何使物體或影像模型（model）能正確的重建出那張影像呢？因此為了解決此類問題，本文作者提了利用許多相關的影像來訓練一個模型，並且利用這個模型來作為重建影像的方法，舉例來說以人臉來作 model，事前工作即收集一群人的臉來訓練（training），並且利用這個模型來重建或者辨別一張新的人臉影像；因此本文作者提了 ASM 與 AAM 的方法，利用統計（statistical）的方法來達成詮釋（interpreting）影像以及重建影像的目的。[1]

每個人(或同種動物)的器官、臉等都不盡相同，當要表示一張新的影像(臉或器官等等)的特徵時，變形是一項必要的動作；而至今雖然變形的方法有許多，但是很少有一個方法能夠提供自動化（Automatic）的能力;最大因素在於系統對於物體（Object）的先前知識（Prior Knowledge）的認知有限，因此為了解決這個問題，所以作者提出利用統計的方式創造出 ASM 與 AAM 的方法；其最大目的便是為了提供一套自動化的系統，除了自動化的優點外，關於 ASM 與 AAM 的優點如下所示[1]：

I. 應用範圍廣；它們可以應用在許許多多的範圍，例如人臉、器官等等。

II. 系統可以取得一些影像所代表的意義 。

III. 系統能夠提供一個嚴謹且正確的變形，而不是隨意的變形，造成最後結果的失真。

IV. 系統不需要過多的假設（例如設定參數微調使的邊界看起來較為平滑（Smoothness））

除了以上的優點外，也有一些在使用前應該注意遵守的條件，如以下所述：

I. 一般性（General）：模型（Model）要能變化成任何它要表示的東西，換句話，也就是能夠達到變形的效果。

II. 正確性（Specific）：Model 只允許變化成合理（Legal）的結果，而不是隨便亂變化。

2.1 ASM 演算法

2.1.0 簡介：

一個物體的形狀通常可以定義一些特徵點（Feature Point, Landmark），並且利用這些特徵點來表示出該物體的形狀(Shape)（如下 Fig. 1）；

Fig. 1 人臉的形狀以及定義出的特徵點[1]

利用物體的形狀通常就能夠做許許多多的事情，例如說：掌紋的比對、指紋的比對、辨識系統等等;而應用的領域更是從日常生活到軍事、醫療等等;而 ASM （Active Shape Models algorithm）就是利用統計的方法製造出一個形狀模型（Modeling），以利用這個模型來達到分析一張新的形狀；舉例

來說，某家公司有員工十人，利用這十個人的臉型來訓練（training）出一個關於臉方面的模型；當今日有一個陌生人來的時候，系統就可以利用ASM 的方法將這名人士的臉型抓出來，並且判斷出這名人士是不是屬於該公司的人。底下章節介紹 ASM 的步驟：

Training the Model：

就 Training 的部分，分成下列幾項討論：

I. **Sample的取向**：Sample的取向必須為同一種「物體」(Object)，舉例來說：如果今日ASM系統是針對人臉的話，Sample的取向就必須以人臉為主，而不可以隨便取像牛或馬的臉等，如果隨便取樣則最後在變識的結果必然會失去其正確性；稱所取的所有Sample的集合為Training Set， 假設取了s個人臉的Sample，則Training Set = $T = \{x_1, x_2, ..., x_s\}$，當中每個x代表一個Sample的圖形。

II. **特徵點（Landmark）的設置**：實際上，ASM 並非直接用圖形來當 Sample，而是在圖形上設置一些特徵點（Landmark），且每個 Sample 之間 landmark 取向必須是相對性(Correspondence)的；一般 landmark 的取樣通常是 邊界（boundary）、T 型交界處（'T'junction）以及特徵點與特徵點間，最後一項乃是為了使的之後我們再作變形時，使的結果更平滑(Smoothness)。(如下 Fig. 2 所示)

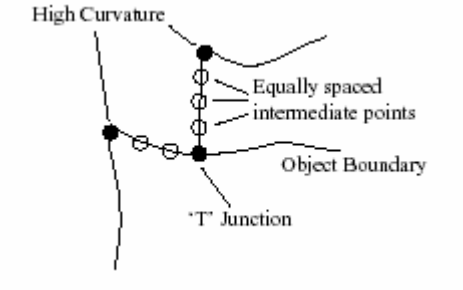


Fig. 2 Landmark 的取向[1]

因此當每個 sample 取了 *n* 個點，則每個 sample 的長相：

2D 表示法：

$$x_i = (X_{i1}, X_{i2}, \ldots, X_{in}, Y_{i1}, Y_{i2}, \ldots, Y_{in})$$

3D 表示法：

$$x_i = (X_{i1}, X_{i2}, \ldots, X_{in}, Y_{i1}, Y_{i2}, \ldots, Y_{in}, Z_{i1}, Z_{i2}, \ldots, Z_{in})$$

當中*i*代表第*i*個Sample，*n*代表著共有*n*個Landmarks，即由每個sample組成一個($d_{(dimension)} X n$)個維度向量。

III. **Aligning the Training Set**：目的是為了降低（minimize）$\sum|x_i - \bar{x}|$，在 training 出 model 前，一定要先作 alignment，這樣才能夠確保 model 的正確性。

IV. *模組化（Modeling）*：模組化即算出 Mean、Training Set 的 Covariance 以及 Covariance 的特徵向量（Eigenvector）與相對應的特徵值（Eigenvalue）。

Mean：

$$\bar{x} = \frac{1}{s}\sum_{i=1}^{s} x_i \qquad -(2.1)$$

Covariance：

$$S = \frac{1}{s-1}\sum_{i=1}^{s}(x_i - \bar{x})(x_i - \bar{x})^T \qquad -(2.2)$$

Eigenvectorφ_i與相對應的Eigenvalue λ_i：

請參略＜[1]的附錄 A 以及[2]＞

只要算出 Mean 與 Eigenvector 以後，ASM 的 Model 的建立就算完成了，Model：

$$x = \bar{x} + \Phi b \qquad -(2.3)$$

當中Φ為Eigenvector所成的集合，即$\Phi = (\varphi_1, \varphi_2, \ldots, \varphi_{nxd})$，且其相對應的特徵值$\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_{nxd})^T$；$b$為可調參數（a set of parameters of a deformable model），即$b = \{b_1, b_2, \ldots, b_{nxd}\}^T$，$b$為

控制 x 的量;請將整個（2.3）看成是一個以 $\bar{x}$ 為原點，Φ為(d x n)-dimension空間的基底所成的集合（稱此空間為Face－Space，如下Fig. 3 所示箭頭處為 $\bar{x}$ ）空間；而每一張新輸入的圖形x都是在此(d x n)-dimension當中的一個向量。

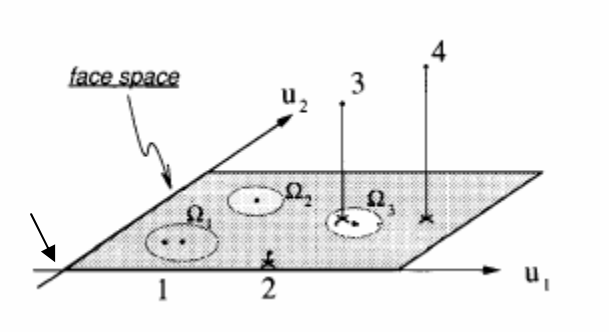


Fig. 3 Face-Space 示意圖[3]

ASM 演算法

當一張新的影像輸入後，如何利用ASM演算法來完成工作（例如辨識、Segmentation等）；第一步驟即是確認該影像的形狀；方法是再設置一些landmark時（Training model前），在每張sample的shape兩邊各取k pixel的灰階值，因此每張sample各有一 2k+1 個pixel所組成的灰階向量g_i（如Fig. 4 所示）；

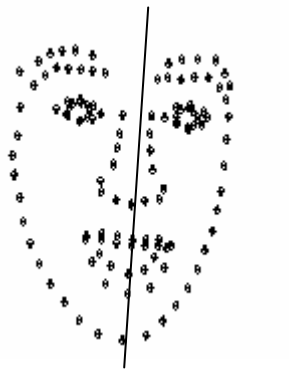


Fig. 4 一張設好 Landmark 的臉，在其每一邊（以中線為分界）各取 k 個 pixel，最後組成一 2k + 1 個 pixel 的灰階向量

並且必須將每個灰階向量作 Normalize，因為每張影像的灰階都

會受到不同光照與週遭環境等因素所影響，因此為了使的在執行 ASM 演算法前能找到較為精確的圖形，需先作 Normalize 使的影響降低；Normalize 的算法如下：

$$g_i = \frac{1}{\sum_i |g_{ij}|} g_i \qquad -(2.4)$$

當中小寫 i 代表著第 i 個sample，j為此灰階向量中的第j項；之後在算出它們的mean $\bar{g}$ 與Covariance S_g，當一張新的影像進來， model同樣從該影像當中取出 $2k+1$ 個pixel, g_s,要計算出其間的差異度如下所示（Mahalanobis Distance）：

$$f(g_s) = (g_s - \bar{g})^T S_g^{-1} (g_s - \bar{g}) \qquad -(2.5)$$

此步驟需要一直（iterative）計算，直到找到最小的$f(g_s)$，則此形狀即為新的影像的圖形。（詳細的說明請看 “[1] 7.2 的部分”）

為了使的 ASM 的結果越正確（因為如果只用上面的方法去找影像的圖形，其正確性與代表性有很大的機率會錯誤），本文作者提出了用 Multi-Resolution 的方法，套用在 ASM 上成為 MRASM（Multi-Resolution Active Shape Models）；MRASM 是先將影像的解析度降低，然後從解析度最低的影像（稱為 Coarser Level）執行 ASM，等到條件滿足了以後在處理較高的解析度，如下頁 Fig. 5 所示，最高的代表解析度較低，影像較小：

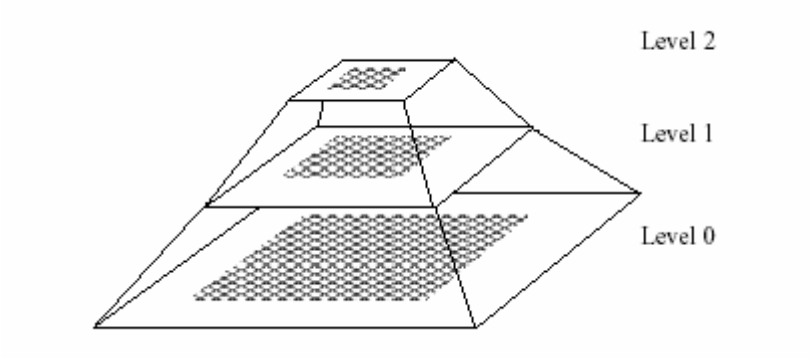


Fig. 5 MRASM 個解析度影像說明圖[1]

底下為 *ASM 演算法變形的整個流程*：([1] Chapter 7 與 Chapter 4）

I. 令$L = L_{max}$

II. 滿足 $L \geq 0$，則執行以下動作

甲、計算在此Level的影像形狀，找尋新的模型位置（New Model Points），X_{sam}（方法如本節第一段所述）

乙、計算出 Pose 與 Shape 參數（本節 2.1.1 的最後一段）

丙、更新 Pose 與 Shape 參數，代入$T_{(x_t,y_t,s,\theta)}(\bar{x}+\Phi b)$令我們的 Mode fit 到新的位置（new position）。

丁、若p_{close}比例的model points沒有match到新的位置或者還iteration次數未到N_{max}時，返回到（II.甲）繼續迴圈。

戊、令 L -> （L - 1）若滿足 $L > 0$.

III. 當所有的執行已經到 Level 0 的時候整個演算法結束.

當中變數說明如表一，

表一　MRASM 變數說明

變數名稱	*敘述*
L_{max}：	coarsest level of gaussian pyramid to search.
N_{max}：	maximum number of iterations allowed at level.
p_{close}：	proportion of points found within ns/2 of the current pos.
L：	目前處理的 Level
Pose 參數：	即 Transform，分 Translation、Scaling 與 Rotation.
Shape 參數：	即 b Vector

在（II.乙）步驟中，並沒有提到該怎麼計算 Pose（T）以及 Shape（b）參數，底下為計算 T 與 b 的方法：

求T與b的目的是為了讓Model經變形後接近於x_{sam}，

即滿足 minimize $|x_{sam} - T_{(x_t,y_t,s,\theta)}(\bar{x} + \Phi b)|$ —（2.6）

演算法如下：

I. 令 b 為一 s-Dimension 的零向量，即 b = （0,0,…,0）

II. 代 b 進（2.3）式，得一結果 x

III. 利用Euclidean或Affine等方法求出位移量、縮放量以及旋轉角度等，使的x 與 x_{sam}間的契合（match）最佳；而此變化量即為Pose 參數 T

IV. 令 $x_{sam}' = T^{-1}(x_{sam})$

V. 將x_{sam}'投影至$\bar{x}$的tangent space上，方法為乘上 $1 / (x_{sam}' \cdot \bar{x})$

VI. 算出 Shape 參數 b，利用 $b = \Phi^T (x'_{sam} - \bar{x})$.　　－（2.7）

VII. 如果算出的 b 參數與 Eigenvalue 滿足：

$$(\sum_{i=1}^{s} \frac{b_i^2}{\lambda^i}) \le M_t \qquad －（2.8）$$

則停止，否則回到II；當中，b_i為b向量之第i項；λ_i為Eigenvalue的第i項；M_t為threshold.

實驗結果：

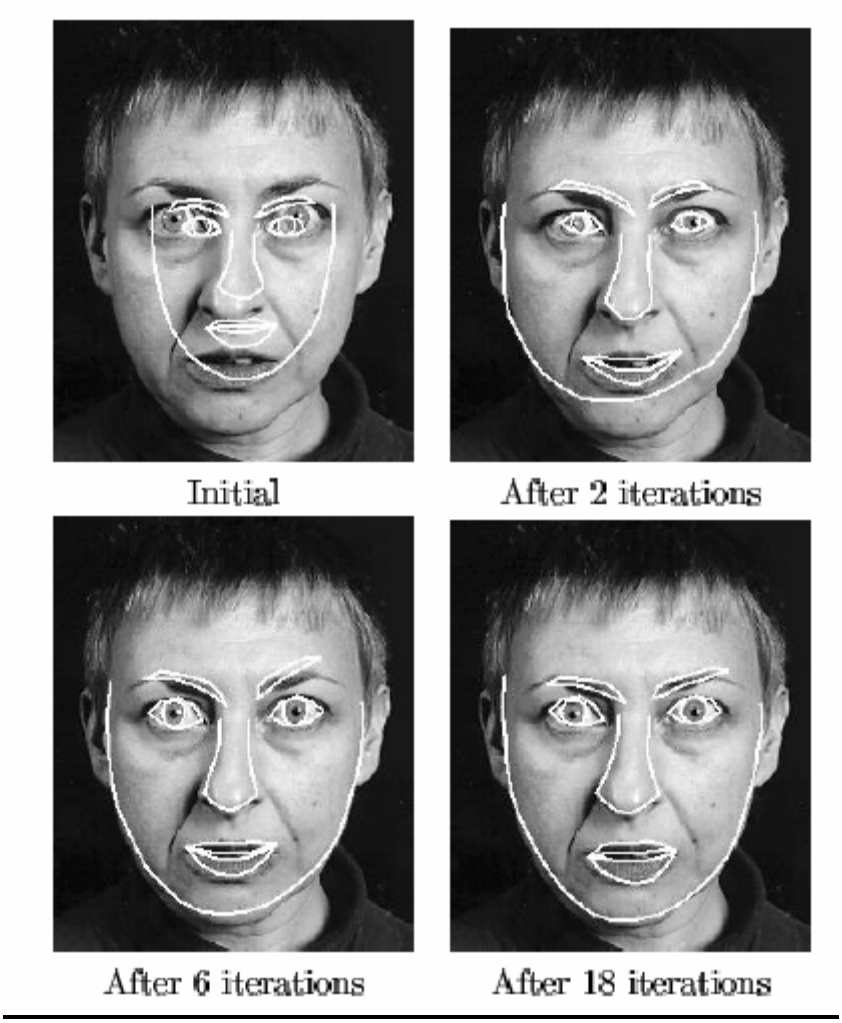


Fig. 6 應用於人臉，一開始的位置取在圖形的中央

上圖 Fig. 6 是利用 MRASM 演算法去定位出一張臉的特徵；Model（白色的部分）一開始是定位在靠近臉中心的位置，搜尋（Search）一開始是從解析度較差的層級（Level）開始；解析度最差的層級是 3（Level 3），因此解析度是降低 1/8。從結果可以很清楚的發現到經過一連串的 Iteration 步驟來搜尋，一些臉的特徵都可以很正確的找到。

下頁圖 Fig. 7 是為了讓使用者知道如果將 Model 定位在很差的地方或

者是距離新的影像過遠，則因為演算法會找到 Local Minimum 的因素，因此結果非常的差。Fig 8.是將 ASM 的方法應用於 MR 影像上，利用 ASM 演算法找出一張新的膝蓋影像的結構。

Fig. 7 當一開始的位置取的不好時，所造成的失真結果

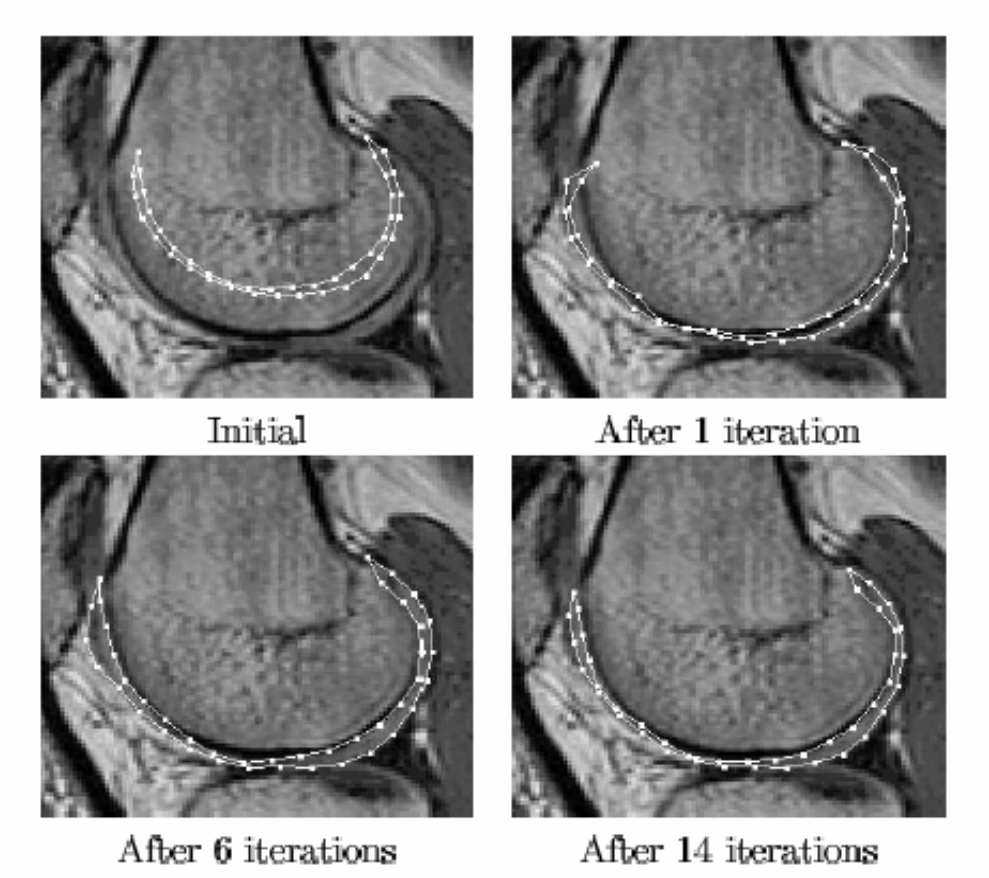


Fig. 8 應用於醫學影像（MR）

2.2 AAM 演算法

2.2.0 簡介：

AAM 其目的在於加強 ASM 對於整張影像作變形的弱點；在前一節中，ASM 它只能處理關於物體的形狀，而這對於一套辨識系統是一個嚴重的問題，以某家公司的 ASM 辨識系統為例，很難保證有個人（A）的臉與整間公司某個員工（B）的臉型是一樣的，但是他們並不是同一人，因為形狀一樣，ASM 系統會產生誤判，導致那 A 先生得以很順利地進入公司的大廈，這對於安全上是一大隱憂；因此 AAM 的創造是為了彌補 ASM 對於處理整張影像的弱點，而 AAM 就為了解決 ASM 的缺點。

2.2.1 Training the Model：

因為 AAM 需要處理影像方面的資訊，因此除了 Shape Model 以外還需要訓練所謂的 Texture Model，Texture Model 主要是再管理影像方面的資訊；因此在 AAM 訓練 Model 方面總共有兩個部分，第一個部分是先訓練 Shape Model，第二個部分則是訓練 Texture Model；對於第一個部分 Shape Model，其建造（Making）方式與 ASM 的方法一樣，因此在這邊就不在重複（請見 2.1.1）；至於 Texture Model 方面，所謂 Texture，在影像上指的就是影像上的灰階值（intensity）或者是色彩（color），因此可以說 Texture Model 為建造一個影像“內容”的模型而 Shape Model 則為“外皮”的模型（請見 Fig. 9 與 Fig. 10）；要建造 Texture Model 前，Shape Model 必須先完成，在這邊假定 Shape

Fig. 9 Set of Points

Fig. 10 Shape Free Patch

Model已經建造完成，因此存在一Mean Shape, $\bar{x}$，接著將所有的Samples的影像（如Fig. 10）Warp到mean shape上，方法可用三角化（triangulation）或者是thin-plate的方法，並將這些經過warping後的intensity資訊向量化，即 $g_i = (g_{i1}, g_{i2}, g_{i3}, \ldots, g_{in})$，$g_i$代表著第$i$個sample的Texture vector；但是因為每個Sample影像的環境不盡相同，因此可能會有光照差異造成影像亮度不一等因素影響，所以需要作一些normalize的動作，本文作者提了兩個係數Scaling α，offset β，利用這兩個係數來降低這些因素的影響，方法如(2.9)

$$g' = (g_i - \beta \cdot 1)/\alpha \qquad -(2.9)$$

如何取得最好的 α 與 β 係數減低環境影響對於影像影響的問題：

I. 先計算出出一個 $\bar{g}$，

II. 然後利用（2.10）與（2.11）算出 α 與 β 係數，再代入（2.9），

$$\alpha = g_i \cdot \bar{g} \qquad -(2.10)$$

$$\beta = (g_i \cdot 1)/n \qquad -(2.11)$$

重複此步驟直到結果不在改變；算出最佳的α與β係數後，將這些係數代入（2.9）算出每個sample影像的new Texture vector，之後在算出mean $\bar{g}$， 與Texture的eigenvector $\Phi_g = \{\varphi_{g1}, \varphi_{g2}, \ldots, \varphi_{gs}\}$以及對應的eigenvalues $\lambda_g = (\lambda_{g1}, \lambda_{g2}, \ldots, \lambda_{gs})^T$，構成Texture Model（2.12）：

$$g = \bar{g} + \phi_g b_g \qquad -(2.12)$$

$$b_g = \Phi_g{}^T (g_{sam} - \bar{g}). \qquad -(2.13);$$

當要復原一張影像g_{im}時，令一u向量，$\mathrm{u} = (u_1, u_2)^T = (\alpha\text{-}1, \beta)^{\mathrm{T}}$，代入(2.13)即可得$g_{im}$。<[1] 附錄E>

$$g_{im} = T_u(\bar{g} + \phi_g b_g) = (1+u_1)(\bar{g} + \phi_g b_g) + u_2 \cdot 1 \qquad -(2.14)$$

完成訓練Texture Model的步驟後，接著就是如何利用Shape Model與Texture Model；因為它們並不是同一個模型，一個專管圖形，另一個則是負責圖形所包含的Intensity即影像內容，因此需要先在它們兩個之間找到一個"橋樑"；在（2.3）與（2.12）中，可以發現到當中的"變數"是b_s以及b_g，因此希望能找到一個共通變數c，首先需將所有的Sample代入（2.7）與（2.13），算出每一個b_s以及b_g，然後利用（2.15）式算出每一個Sample的b向量，

$$b = \begin{pmatrix} W_s b_s \\ b_g \end{pmatrix} = \begin{pmatrix} W_s \Phi_s^T (x - \bar{x}) \\ \Phi_g^T (g - \bar{g}) \end{pmatrix} \qquad -(2.15)$$

當中W_s（Diagonal Matrix,詳細的說明請參考[1] 5.2.1）是為了使的b_s與b_g一致性，因為b_s與b_g在意義上不相同，因此需要W_s當轉換；舉例來說，一個人跑d公尺，花了s秒，則速率上是Speed = d/s，當中速率即為時間與空間的仲介；而W_s的腳色就跟速率的角色是一樣的；算出所有的b向量以後，接著算出它們的Eigenvector Q以及相對應的Eigenvalues λ_c（利用PCA的方法），當中，

$$Q = \begin{pmatrix} Q_s \\ Q_g \end{pmatrix} \qquad -(2.16)$$

之後就可以利用 Q 算出 b，利用（2.17）式：

$$b = Qc \qquad -(2.17)$$

c 參數稱為 Appearance Parameter（或稱 Model Parameter）； 利用（2.17）改寫（2.7）、(2.12)：

$$x = \bar{x} + \phi_s W_s^{-1} Q_s c \qquad -(2.18)$$

$$g = \bar{g} + \phi_g Q_g c \qquad -(2.19)。$$

2.2.2 AAM 演算法：

AAM的演算法主要分成兩大部分，第一個部分是計算出兩張影像（AAM的Model與Training Set當中的任何一個Sample）的灰階差與Pose（即T參數，請見2.1.2）、Texture（即u，請見上一節）以及 Model parameter（即c，請見上一節）之間的關係，第二部分才是介紹如何讓Model變形成目標影像（新的影像）；為何第一個部分要先求出影像的灰階差與此三個參數間的關係式（R_c, R_T, R_u）？因為整個AAM的演算都以彼此間（Model與新的影像）的灰階差來判斷所有的參數應該要調整的量，進而改變Model使的Model更接近目標影像，所以第一步驟就需要先計算灰階差與三個參數間的關係式，底下的演算法Step1 是第一階段的過程，Step2 是第二階段：

Step1：Learning to Correct Model Parameter.

令$u = (u_1, u_2)$，$t = (s_x, s_y, t_x, t_y)$.

I. 在Training Set中挑出一個Sample，已知該Sample的Model parameter c_0, Pose parameter t_0,以及Texture parameter u_0.

II. 隨機產生 Model, Pose 以及 Texture 的變量 $\delta c, \delta t, \delta u$

III. 令 $c = \delta c + c_0$.

IV. 令t為t_0經過 δt 改變後的Pose parameter，$T_t = T_{t0}(T_{\delta t}(X))$（$T_t$的計算方式請見 [1] 附綠D），當中$X$為經過（2.18）式計算後所得.

V. 令u為u_0經過 δu 改變後的Texture parameter，$T_u = T_{u0}(T_{\delta u}(g))$

（T_u的計算方式請見 [1] 附綠D），當中g為經過（2.19）式計算後所得.

VI. 將c代入（2.18）式子後可得X_{new}，在將X_{new}經Pose parameter T_t（X_{new}）可得一新的Shape X'；將c代入（2.19）式後可得一normalized 的texture vector, g_m.

VII. 將所得的X'取其在改Sample上所涵蓋範圍內的影像得一Texture vector, g_{im}.（g_{im}未經過normalized）

VIII. 將g_{im}作normalize得 $g_s = T_u^{-1}(g_{im})$.

IX. 計算δg 利用 $\delta g = g_s - g_m$

接著記錄目前的$\delta c, \delta t, \delta u$ 以及 δg ，之後輸入另一張Sample（從Training Set中）返回II重新開始，等所有的Samples都試完後，利用multi-variate regression的方法計算Model參數與灰階差（δg）的關係R_c, Pose與灰階差的關係R_t以及Texture與灰階差的關係R_u：

$$\delta c = R_c \delta g$$

$$\delta t = R_t \delta g \qquad -（2.20）$$

$$\delta u = R_u \delta g$$

但這種方法（multi-variate regression）只有在變化幅度不大的情況下才有可能達到 linear 的條件，因此作者在最後有提到必須將 Model、Pose 以及 Texture 等參數限定在一定範圍內的條件，詳細的條件請參照（[1] 8.4）。

Step2：Iterative Model Refinement.（[1]）（[4]）

輸入一新Sample I_{new}

I. 給定一組c_0, t_0, u_0參數；代入（2.18）與（2.19）式並經過T_{t0}可得一Shape X與Model所呈現的影像g_m；

II. 利用X取X所涵蓋（在I_{new}）範圍的影像，g_s

III. 計算 $\delta g_0 = g_s - g_m$.

IV. 計算 RMS 令 $E_0 = |\delta g_0|^2$.

V. 利用（2.20）算出 $\delta c, \delta t, \delta u$.

VI. 設 $k = 1$.

VII. 令 $c_1 = c_0 - k\delta c$

$$t_1 = t_0 - k\delta t \qquad \text{—（2.21）}$$

$$u_1 = u_0 - k\delta u$$

VIII. 利用新算出來的參數代入（2.18）與（2.19），可得一Shape x'與產生一新的Model texture vector g_m'.並且利用$T_{t1}(x')$得X'.

IX. 利用新的X'取得（從I_{new}）一未經過Normalized的Texture Vector, g_{im}.

X. 取 $g_s = T_{u1}(g_{im})$，並計算 $\delta g_1 = g_s - g_m'$

XI. 假如 $|\delta g_1|^2 < E_0$，則取代c_0,t_0,u_0以及 δg_0 等舊參數，並且跳回IV繼續執行.

XII. 否則令 $k = 1.5, 0.5, 0.25$ 等等跳回 VII 繼續執行，重複 VII 到 XII 直到 RMS 一直維持不在改變.

實驗結果：

下圖 Fig. 11 利用 AAM 演算法做三張新的人臉影像重建，每一組實驗都是先將模型放置在新的人臉影像的中間（Face Centre），然後再利用 AAM 演算法做影像變形，使的模型變成目標的臉。Fig. 12 是利用 AAM 演算法找尋 Model 與新的影像間最佳的契合（Fit），重建膝蓋骨頭的影像位置。

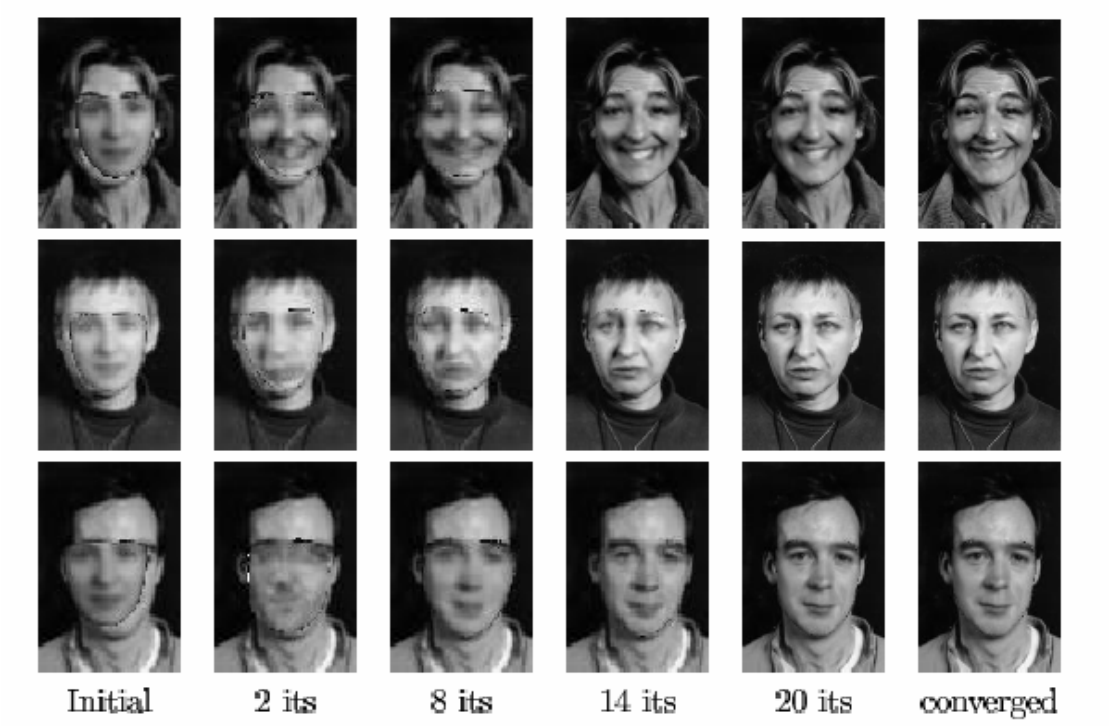


Fig. 11 ：應用於人臉辨識

Fig. 12 : 利用 AAM 尋找膝蓋骨頭的位置

2.3 結論:

前面的章節中，本文介紹了關於 ASM 與 AAM 演算法，可以很清楚的了解到 ASM 與 AAM 的差別以及它們的共同優點，它們共通的優點在於它們提供了一個自動化的效果，這能夠大大的降低使用者在處理資訊上面的麻煩；而它們差別在 ASM 只能處理 Shape 方面的資訊，因此對於影像方面的處理能力非常的薄弱；反觀，AAM 就是為了解決這個問題而發展起來的，它能夠將整個影像作一個適當的變化，因此在做辨識等系統方面，AAM 能夠提供較精確的結果；但是 AAM 並非沒有缺點，它的缺點就是過程太過複雜，且所需要處理的資訊也較多，因此對於整個系統它所需求的容量相對的也較大，負載也較重；反觀 ASM 因為只是很單純的處理圖形方面的資訊，因此它處理的資訊較 AAM 簡單，所以對於系統的負載較輕，所需求的容量也較低；因為我們此次的實驗資料是果蠅腦的中軸，因此在直觀上本文便決定使用 ASM 的方法，但是因為不論是 ASM 或者是 AAM 在 3D 上幾乎少有人實作過（除了少數幾篇 paper 如[4]），因此首要的工作就是如何將 2D 的方法套用到 3D，下面一章將提出如何利用 ASM 來作 3D 變形的演算法。

第三章 ASMW

(Active Shape Model for Warping in 3D)

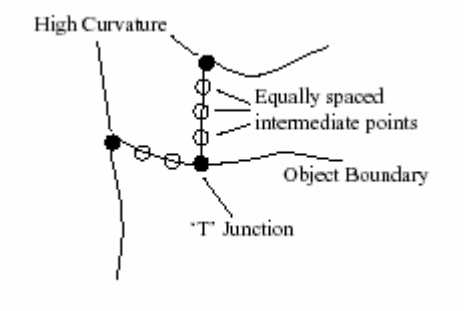
3.0 簡介：

在上一章節中大致介紹有關 ASM 與 AAM 的演算法，本章節將介紹如何將 ASM 作一些改變以符合變形（Warping）的效果；這邊所提到的變形並非像 ASM 一樣只做圖形（Shape）方面的變形，而是對整個物體作變形，但是在前一章中可以很清楚的知道 ASM 只能作圖形方面的變形，除了這個問題以外，是否有注意到在上一章的介紹中不論是 ASM 或者是 AAM 演算法，都只提到如何再 2D 上實作，且在結果中也只能看到 2D 的結果，因此又多了一個問題－3D 是否會成功以及 3D 該怎麼實作？這些問題都是需要解決的地方，除此，之所以會選擇 ASM 演算法也是因為我們所利用的的資訊只有中軸以及影像資料，因此稱我們的方法為 *ASMW（Active Shape Model for Warping）*，目的就在於解決物體變形、3D 化等問題。

3.1 資料型態：

之前在簡介中有介紹到本文選擇 ASM 當 ASMW 的依據其最重要的原因是因為輸入資料是物體的中軸；中軸對於一個條狀物體而言可說是最具代表性的形狀，常常在實驗中以中軸來取代原本的物體，其最主要的目的就是為了方便

使用者觀察其物體的形狀；但在這邊並不是直接利用中軸圖形，而是在中軸上取一些特徵點（Landmarks），而特徵點的取法與 ASM 的取法一樣，1.**邊界（boundary）、2.T 型交界處（'T' junction）**以及 **3.特徵點與特徵點間**，最後一項乃是為了使的之後在作變形時，使的結果能更平滑（Smoothness）。（如下圖）



當所有的 Samples 的特徵點都決定好了以後，接下來的步驟需要將每個每個 Sample 的特徵點組合 1 個 *X-Dimension* 維度的向量，其組成方式如下所示：

2D 表示法：

$$x_i = (X_{i1}, X_{i2}, \ldots, X_{in}, Y_{i1}, Y_{i2}, \ldots, Y_{in}) \quad -(3.1)$$

3D 表示法：

$$x_i = (X_{i1}, X_{i2}, \ldots, X_{in}, Y_{i1}, Y_{i2}, \ldots, Y_{in}, Z_{i1}, Z_{i2}, \ldots, Z_{in}) \quad -(3.2)$$

當中 *i* 代表第 *i* 個 Sample，*n* 代表著共有 *n* 個 Landmarks。

而由許多 Sample 所組成的集合稱之為 Training Set，如（3.3）式所示：

$$T = \{x_1, x_2, \ldots, x_i, \ldots, x_s\} \quad -(3.3)$$

i 代表著第 *i* 個 Sample，*s* 代表共有 *s* 個 Samples。

3.2 ASMW

因為 ASMW 是以 ASM 演算法為基礎，因此這邊分成兩部分來作介紹，第一部分當然就是先 Training 出 model，第二部分則是利用 ASMW 作物體的變形

（Warping）：

3.2.0 Training the Model：

在 Training 這個步驟中，其方式與 ASM 幾乎相同，只有一些地方要稍作改變：

I. **Alignment the Training Set**：

首先須先 Alignment 所有的 Samples，這個目的是為了使的所 training 出的 Model 具有代表性，因為雖然是以同一種物體來 training 出來的 Model，但是如果說每個物體所拍攝的角度有差別或者是所取的解析度不一樣，這些因素都會影響到所 Training 出來的 model，因而會造成最後在作變形時結果的誤差；而 Alignment 的方式是透過位移 Translation、旋轉 Rotation 以及縮放 Scaling 來降低 $\sum|x_i-\bar{x}|$。

II. **Modeling**：

Modeling 的部分與 ASM 一樣，算出 Mean、Training Set 的 Covariance 以及 Covariance 的特徵向量（Eigenvector）與相對應的特徵值（Eigenvalue）。

Mean：

$$\bar{x}=\frac{1}{s}\sum_{i=1}^{s}x_i \qquad -(3.4)$$

Covariance：

$$S=\frac{1}{s-1}\sum_{i=1}^{s}(x_i-\bar{x})(x_i-\bar{x})^T \qquad -(3.5)$$

Eigenvector φ_i與相對應的Eigenvalue λ_i：

請參略＜[1] 的附錄 A 以及[2]＞

只要算出 Mean 與 Eigenvector 以後，ASMW Model 的建立就算完成了，Model：

$$x = \bar{x} + \Phi b \qquad -(3.6)$$

當中Φ為Eigenvector所成的集合，即$\Phi = (\varphi_1, \varphi_2, \ldots, \varphi_{nxd})$，且其相對應的特徵值$\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_{nxd})^T$；$b$為可調參數（a set of parameters of a deformable model），即$b = \{b_1, b_2, \ldots, b_{nxd}\}^T$，$b$為控制$X$的量，稱$b$為Model Parameter（模型參數）;將整個（3.6）看成是一個以$\bar{x}$為原點，Φ為$(d\ X\ n)$-dimension空間的基底所成的集合（稱此空間為Face－Space，如下圖所示）；

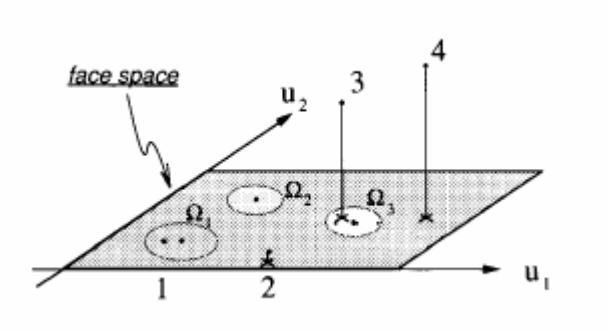


Fig. 13 Face-Space

而每一張新輸入的圖形x都是在此$(d\ X\ n)$-dimension當中的一個向量;以2D空間為例，其原點是$(0, 0)^T$，稱O點，今日有一點A其座標為(a, b)，a、b為任一，則可以用底下的算式來取代(a, b)：

$$\begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} + a\begin{pmatrix} 1 \\ 0 \end{pmatrix} + b\begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\begin{pmatrix} a \\ b \end{pmatrix} \qquad -(3.7)$$

當中$\begin{pmatrix} a \\ b \end{pmatrix}$所構成的向量即是（3.6）式中的Model Parameter b，而可以將陣列$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$當作（3.6）式中的$\Phi$。

雖然計算出 Eigenvalue、Eigenvector、Covariance 以及 Sample 的 mean，但是因為最後真正有用到（在作 Warping 的時候）的部分是 mean 與 eigenvector，這與 ASM 的方法並不相同， ASM 演算法在變形的過程中有利用到 Eigenvalue 與 Model Parameter 來判斷

loop 是否要停止，但為了讓結果能越近似於目標物體，而不是像 ASM 一樣只要滿足在空間上到達某一程度的相似即可（例如：在 ASM 中有提到的$(\sum_{i=1}^{s}\frac{b_i^2}{\lambda^i}) \le M_t$，只要滿足這個條件就可以說目標物體是一個人臉，因為它很接近 Model 所構成的 Face-Space），因此最後只需要紀錄 Mean 與 Eigenvector 向量。

3.2.1 Warping（ASMW）：

前一節提到了如何training出ASMW的model，這一節將介紹 1.如何取得 Transform T以及Model Parameter b（**3.2.1.0**），以及 2.如何利用T與b使的Test Sample, $I_{Testsample}$（欲測者）,變形成Target Sample, I_{target}（目標物體）（**3.2.1.1**）這兩部分；

3.2.1.0 Transform *T* 以及 Model Parameter *b* 的取得：

就這個部分來談，求 T 與 b 的方法與 2.1.2 ASM 演算法的最後一段非常相似（同樣是求 T 與 b），但是因為 ASMW 是一種作變形的方法，因此要讓所求的 T 與 b 能夠使的 Test Sample 能越接近 Target Sample，所以在做法上有些許的改變，以符合要求，底下為求 T 與 b 的演算法：

- 今日有一Sample，令它的中軸為X_{sam}，欲算出它X_{sam}與我們的 Model $\bar{x}$ 間的變化量（Transform T與Model Parameter b）.
- 目的：求 T 與 b 使的（3.8）式 minimized，

$$|x_{sam} - T_{(x_t,y_t,z_t,s,\theta)}(\bar{x} + \Phi b)| \qquad -(3.8)$$

- 演算法中有用到的參數介紹：

T_{temp} : Transform T 的暫存器（temporal buffer）((X_t, Y_t, Z_t, s, θ)，當中X_t, Y_t, Z_t表示X, Y以及Z項作X_t, Y_t, Z_t量的位移).

b_{temp}: Model parameter 的暫存器（temporal buffer）

(屬於 s dimension vector).

b : Model Parameter buffer （屬於 s dimension vector).

T : Transform buffer (X_t, Y_t, Z_t, s, θ).

Cont：計數器，設為 0.

I. 令$b_{temp} = [0]_{s\ x\ 1}$.

II. 代b_{temp}入（3.6）式 $x = \bar{x} + \Phi b_{temp}$. ，得一Shape x.

III. 利用**ICP**等方法算出x與X_{sam}最佳的T_{temp}（利用Scaling，Rotation以及Translation等方式使的x與X_{sam}最接近）。

IV. 利用求得的T_{temp}對X_{sam}作inverse，得x'_{sam}，即 $x'_{sam} = T_{temp}^{-1}(x_{sam})$

V. 利用 $b_{temp} = \Phi^T(x'_{sam} - \bar{x})$.，算$b_{temp}$.

VI. 令 $E1 = | x_{sam} - T(\bar{x} + \Phi b) |$
$E2 = | x_{sam} - T_{temp}(\bar{x} + \Phi b_{temp}) |$ —（3.9）

若滿足$Cont = 0$ 或者是$E1 > E2$ 的情況時，$b = b_{temp}$且$T = T_{temp}$.

VII. 若 *Cont* = Maximum Counter 或者是 b 與 T 不再改變時，結束迴圈。

VIII. 否則跳回第 II 並且 $Cont = Cont + 1$.

這個演算法是否會收斂呢？會，原因在於其實T與b參數兩者是一體的，換句話說，T改變則b才會改變，因此可以將b與T結合在一起看使的$P = [T, b]^T$，則整個演算法就可以看成只是改變P，使的x_{sam}與Model $\bar{x}$能越契合（match），證明方式請參考[12]。

3.2.1.1 變形：

這個章節將介紹如何使的預測者$I_{Testsample}$變形成目標物體I_{target}，但

是暫時先不談整個物體變形的問題，先以中軸的變形流程來做說明，為了方便讀者了解大致的**變形順序**（流程圖如下Fig. 14 所示）；首先計算出預測者與Model間的T與b參數(利用 3.2.1.0)，然後讓預測者的中軸$X_{Testsample}$先變形成Model $\bar{x}$，接著再從Model變形成目標物體的中軸，但是因為整個方法是連續性，而Model與X_{target}在過程中並無法得知Transform T與Model b參數，因此必須先利用 3.2.1.0 節所介紹的方法算出Model與目標物體之間的T與b，如此才能夠讓$X_{Testsample}$從Model這個“中繼站”轉變成X_{target}：

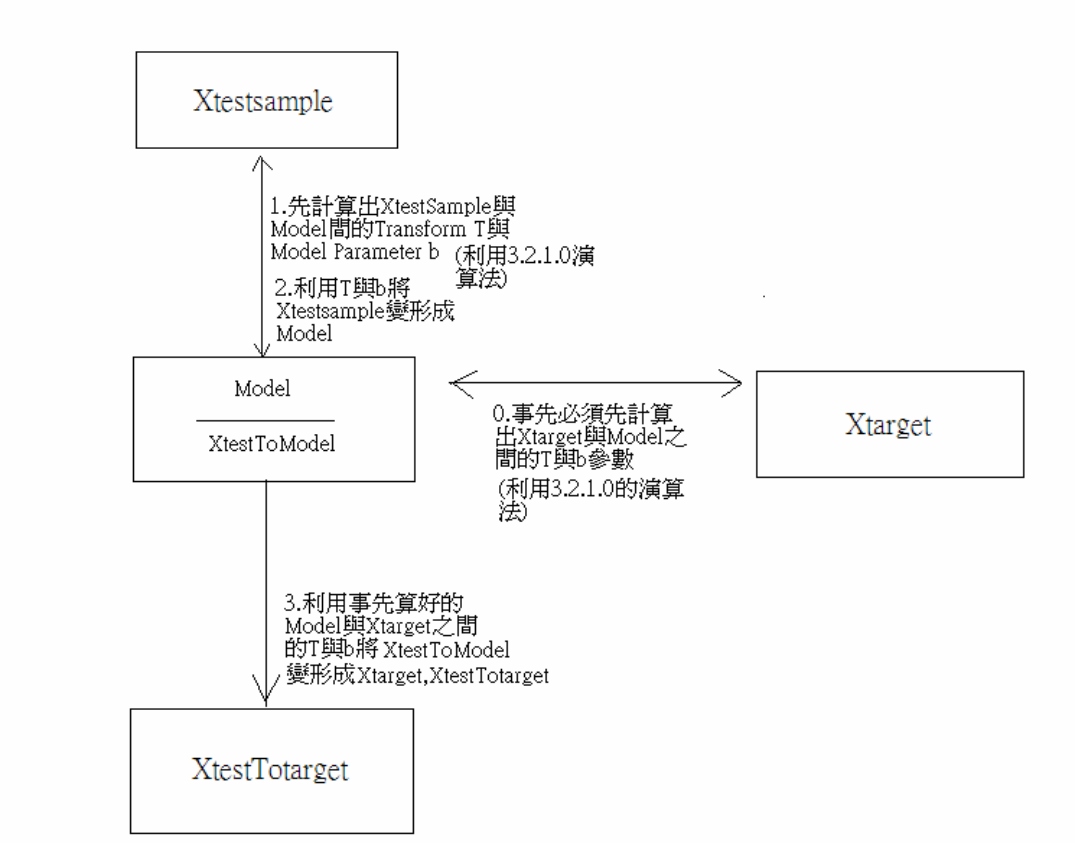


Fig. 14 變形流程

接著，說明關於如何對整個物體（3D - Volume Data，2D - Image）作變形（Warping）；在上一段文章中有提到如何讓預測者的中軸$X_{testsample}$轉變成$X_{testTotarget}$的流程，而尚未提到物體的變形，其實對於物體的變形

在這邊所提供的方法會牽涉到一點簡單的分群概念；但在說明前，必須先讓讀者了解到一件事情，到目前為止所談到的向量不論是$X_{testsample}$、$X_{testTotarget}$或者是$\bar{x}$都看成是一個（d x n）-dimension的向量（d是原本的維度，例如 2D,3D；n是設的特徵點的數量），並非是真實的 2D或 3D維度，但是一旦作物體變形的時候，首先必須要將這些向量轉回原本的Dimension，意思是說如果原本是 3D，則必須將$X_{testsample}$轉回 3D上的n個點，而不能直接用$X_{testsample}$等向量表示法來作變形；**底下為整個變形的演算法的流程**：

- 演算法中有使用到的變數介紹：

  $X_{testsample}$：Test Sample的Centerline。

  $I_{testsample}$：Test Sample的圖檔；例如 2D為影像,3D為Volume Data等。

  X_{target}：目標物體的Centerline。

  $\overline{X}$：Model 的 Centerline。

  $T_{ToModel}$：由Test Sample變成Model的Transform參數。（可逆）

  $b_{ToModel}$：由Test Sample變成Model的Model參數。（可逆）

  $T_{ToTarget}$：由Model變成目標物體形狀的Transform參數。（可逆）

  $b_{ToTarget}$：由Model變成目標物體形狀的Model參數。（可逆）

  Φ：所作出的Model其s個eigenvector所組成的matrix，即$\Phi=(\varphi_1, \varphi_2, \ldots, \varphi_s)$。

- 變形的式子：

  對整個 Shape：

  由 Sample 變到 Model $\bar{x} = T^{-1}(x) - \phi b$ —（3.10）

  由 Model 變到 Sample $x = T(\bar{x} + \phi b),$ —（3.11）

Φb 是微調即 Local warping，T（Transform）是 Global warping。整個變形的方式都是會作一次 Global warping 以及一次 Local warping。

對單點：

由 Sample 變到 Model 的位置上 $x' = T^{-1}(x) - Dist(i)$ —（3.13）

由 Model 到 Sample 的位置 $x = T(x' + Dist(i)),$ —（3.14）

其中，$Dist = \phi b = \left(x_{d1}, x_{d2}, \ldots, x_{dn}, y_{d1}, y_{d2}, \ldots, y_{dn}, z_{d1}, z_{d2}, \ldots, z_{dn}\right)^T$，

Dist (i) = (xdi, ydi, zdi)^T，i 為第 i 個 Landmark 的微調量（我們稱 $Dist$ 為微調）；$x = (x, y, z)^T, \bar{x} = (x', y', z')^T$（以上接為 3D 例子，2D 請自行將 z 值省略即可）。

I. 先計算由Model到Target Sample之間的Transform $T_{ToTarget}$以及$b_{ToTarget}$利用 3.2.1.0 節中所提的方法。

II. 利用 3.2.1.0 的演算法計算Test Sample與Model之間的Transform $T_{ToModel}$以及Model Parameter $b_{ToModel}$，並且利用$T_{ToModel}$與$b_{ToModel}$使的$X_{Testsample}$變成Model（利用 3.10），稱$X_{testToModel}$.

III. $X_{testToModel}$雖然是一個$n \ X \ d$維度的向量，但實際上是一個n個points所組成的向量（如有問題請看 3.1 節），因為此步驟是將$I_{Testsample}$ Warp到Model上，在這必須利用一些分群的方法，所以必須將$X_{testToModel}$的特徵點（Landmarks）分開來看，作法是：

將每個在$X_{testToModel}$上的特徵點，以每個點為中心擴張一個範圍（例如 41 x 41 x 41 的立方體）；接著就是要取得每個範圍內的所有點的顏色，必須計算出點與每個特徵點的距離（並非是直接取該範圍中心特徵點），取最近距離的特徵點的微調，為了讓讀者容易了解請見下頁Fig. 15（以 2D示意圖做說明）：

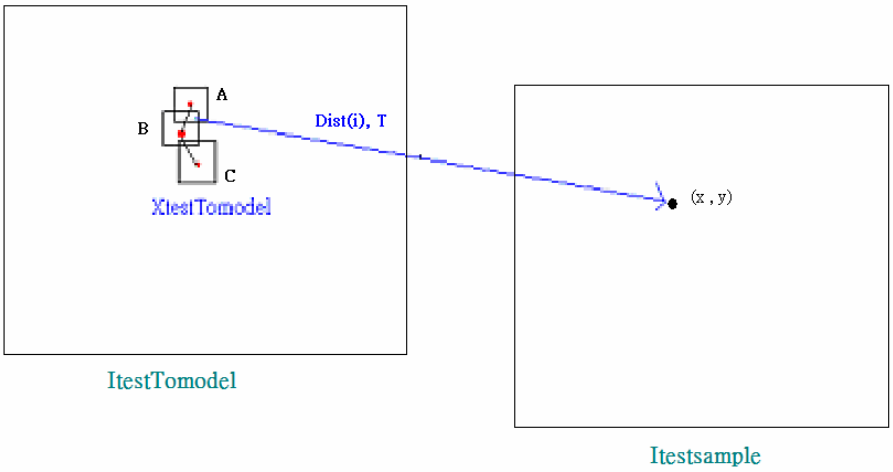


Fig. 15 將點變回原位置之示意圖

右邊大正方形是原本的影像，左邊的正方形是變形到Model上的影像（所求的影像），A,B,C為Landmarks，小正方形是每個Landmark變形到Model上後所框出來的範圍，紅色的點是我們的Landmarks，A,B以及C間的連線是Shape的長相（秀Landmark的順序），可以注意到小正方形都有一些重覆到的區域，這也就是為什麼要用分群法中取最近距離的因素，利用距離的遠近作一次分群，並取最近距離的Landmark的微調參數*Dist (i)*，然後利用（3.14）式，使的該點回到原本在$I_{Testsample}$中的位置（黑點（x, y）的位置）並取出該位置的顏色填回變型到Model上的位置；（之所以取最近距離的微調因子是因為距離最接近的Landmark應與該點的"性質"（其變化的量）最接近，例如讓一個彈簧或繩子之類的東西做彎曲動作，假設在上面劃一紅點，則可以在實驗中發現到當距離紅點越近時其變化的程度與紅點越接近，反之距離紅點越遠時其變化的程度差的越遠；如遇到多個相同距離的Landmark時取它們的微調值的平均，即

$Dis\tan ce(i)+Dis\tan ce(j)+.../t$ ， t為與該點最近距離的Landmark數量）令最後所得的影像為$I_{testTomodel}$。

IV. 接下來利用事先算好的$T_{ToTarget}$與$b_{ToTarget}$帶入（3.10）式，先將$X_{testToModel}$變形成X_{targer}。

V. 最後運用同III步驟一樣的方法，先將變形過後的Centerline $X_{testTotarget}$中每個Landmark點取一個範圍，然後對所有範圍內的所有點作分群，計算出範圍內每個點與所有的Landmark的距離，將距離較近的放在同一群，然後（與III唯一不同的地方）代$T_{ToTarget}$與$Dist(i)$參數入（3.13）式，得原本在Model（$I_{testTomodel}$）上的位置，然後取此位置的顏色值填入變形後的位置；等所有範圍內的所有點都完成後，得一新的圖像 $I_{testTotarget}$。

以上便是整個 ASMW 變形演算法的流程，在第四章中將介紹關於程式的使用流程以及幾個實驗結果的討論。

3.3 結論

底下 Fig. 16，17，18 為一開始作 2D 實驗時的結果，Sample 是取至於心血管。在此作 2D 的實驗為的是求證 ASMW 是否能成功，由實

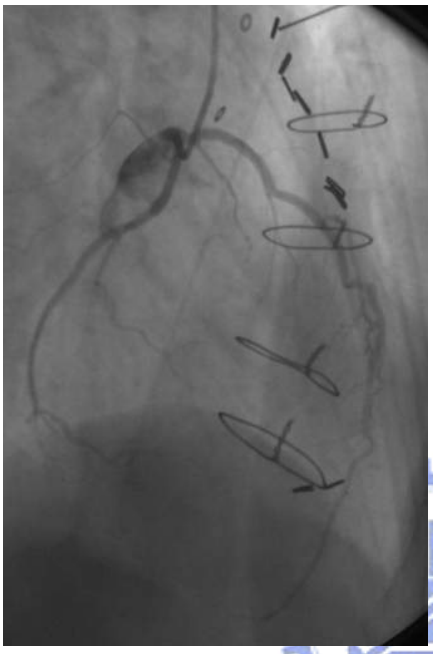


Fig. 16：變形前

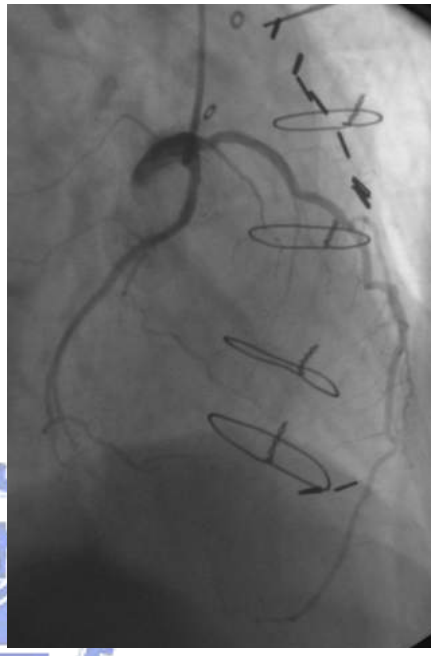


Fig. 17：欲變形的目標

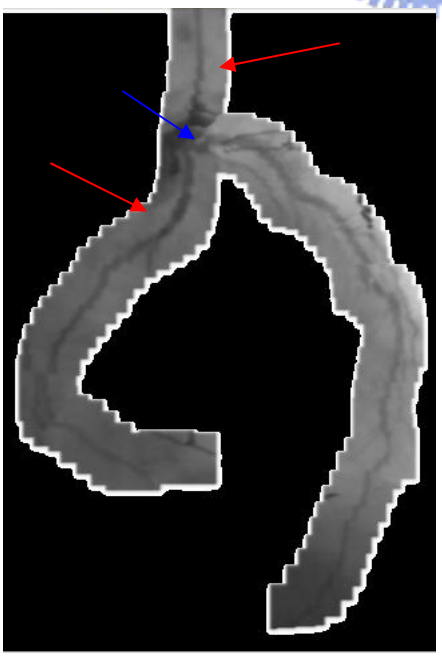
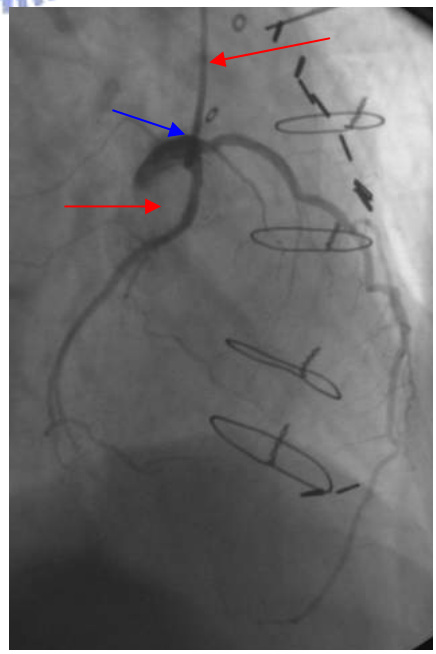


Fig. 18：變形後的結果



（同 Fig. 17 對照用）

驗結果證明 ASMW 應用在 2D 上其結果還不錯，請看紅線的地方並且對

照 Fig. 16（尚未變形的圖），請觀察到 Fig. 16 與 Fig. 17 的斜度與寬度都不一樣，變形後可以很明顯的看出兩者間的形狀都很吻合，藍色箭頭的部分是因為在實驗中那個部分取了一些多餘的點，導致最後在那個部分的結果有些擠壓的效果；2D 的實驗所取的 Landmark 的範圍是 21x21 大小的正方形。（補充：在 2D 的實驗上並非使用 Centerline 而是利用人工設點＜設置在靠近中線的地方＞，因此才會造成如 Fig. 18 藍色箭頭的誤差；在 3D 的實驗中，設置的特徵點則完全取自於中線，找中線的方式是利用 GVF 演算法）

第四章　　程式流程與實驗結果

4.1 實驗結果討論

4.1.0 3D 結果圖：

底下的圖為這次的實驗對象－果蠅的頭腦，為了方便使用者觀看，秀圖的方式是用 Volume Render，Fig. 20~22 為測試物體變形到目標物體（目標果蠅腦），Fig. 23~24 為對照圖：

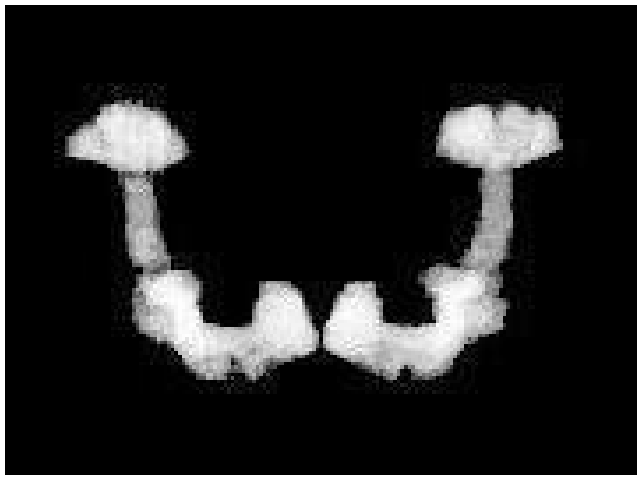


Fig. 19　目標果蠅腦

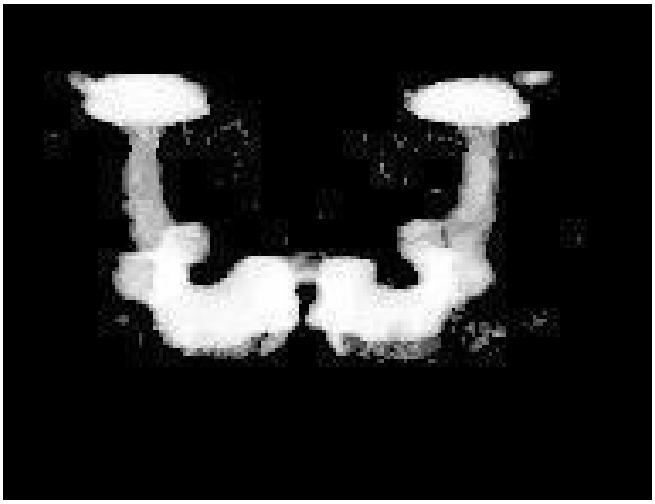


Fig. 20　測試的資料 $I_{testsample}$

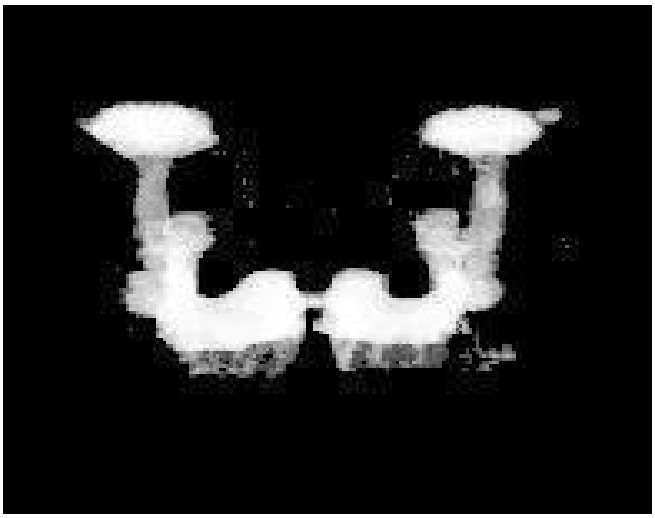


Fig. 21　變形到 Model 上

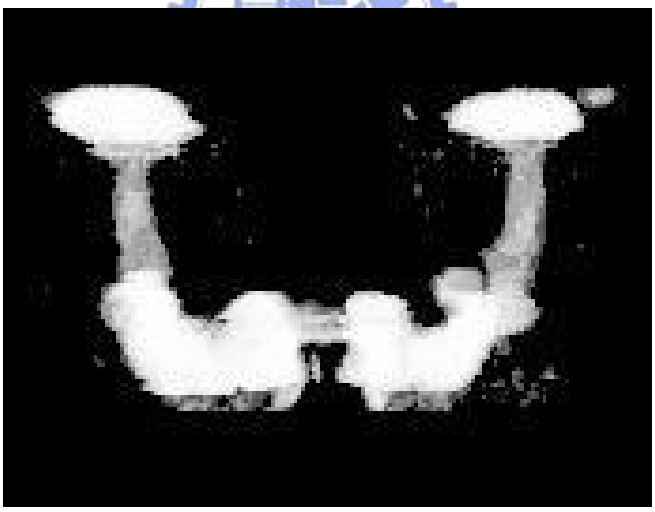


Fig. 22　變形到目標果蠅腦後的結果

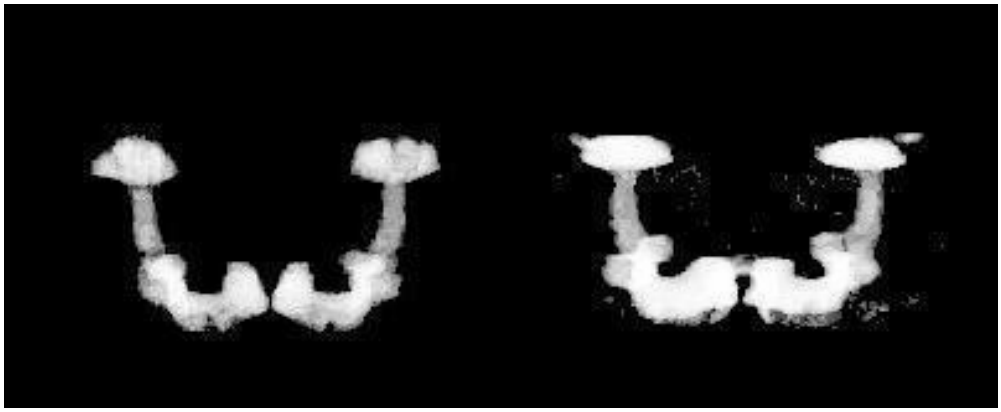


Fig. 23　未變行前的對照圖,左圖為目標果蠅腦,右圖為測試的果蠅腦

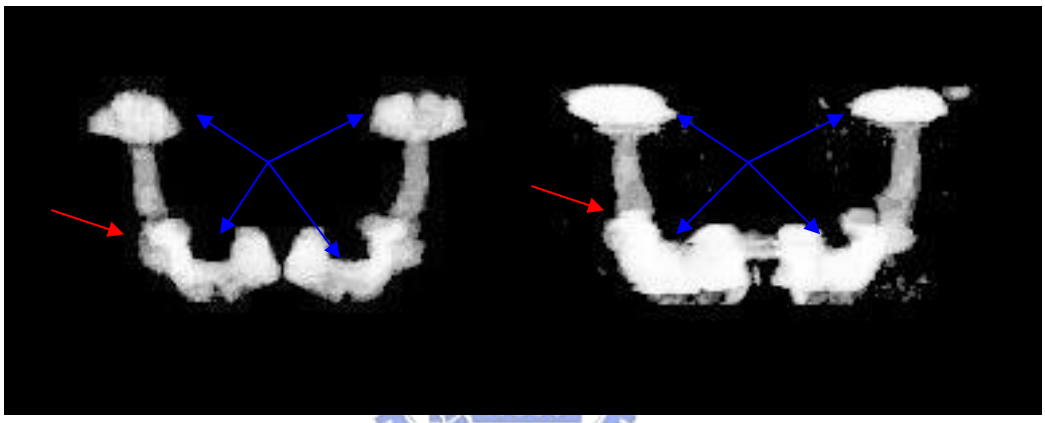


Fig. 24　變形後的測試果蠅腦與目標果蠅腦的對照圖

大家可以發現到到紅色線的部分，在未變行前測試目標紅色箭頭所指的地方非常的突出，經過變形後變的與目標腦非常的像；但是大家一定也發現到藍色的部分仍沒有變的比較像，以下頁 Fig. 25 來做說明，O 點為物體中軸，外邊大圓圈假設是測試目標，因為所輸入資料只有中軸（O）的部分，對於其(中軸)周圍並不加以考慮，如果想讓藍色部分的結果更接近於目標圖形，可以在其中軸特徵點的周圍設置一些點，以下頁 Fig. 25 為例，即在 O 點周圍設置 A,B,C 以及 D 點。

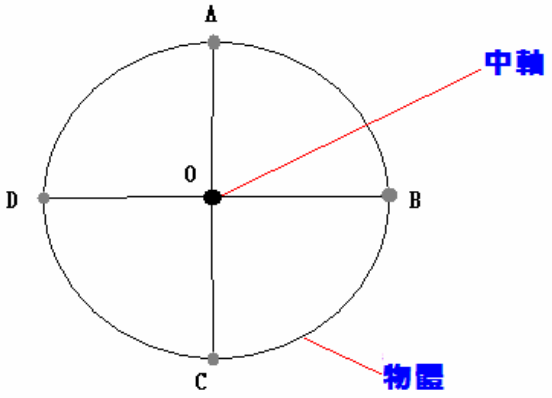


Fig.25　說明圖示意圖

4.1.1 以錯誤的 Model 所造成的誤差：

在前面的章節中，本文有提到關於取樣的部分，做什麼樣的 Model 就要用什麼樣的 Sample，以果蠅為例，做果蠅的 Model 當然就要用果蠅的 Sample 才合理，但如果不是用果蠅做 Model，將會造成錯誤的結果；以這次的實驗中就使用蟑螂以及一顆經過破壞的果蠅腦所做出的 Model 來讓果蠅腦的 Sample 做變形，可以看到在結果上造成的誤差非常的大，如下圖所示:

Fig. 26 左圖未變型前，右圖變形後

因此當要做什麼實驗時，請務必取樣要一樣，不然所做成的 Model 將失去其意義。

第五章　論文總結與未來研究方向

綜觀現今的所有變形方法，很少有一種能提供自動化的效果，在這邊本文提了 ASMW 的目的就是為了提供一套全自動化變形的方法，如此一來就可以節省掉許多麻煩，使用者也不需要手動一個一個去調參數等等，帶來許多便利以及提升效率，使用者只需要將中軸點與影像資料代入系統中，就可以讓系統自動去自算執行變形，使用者還可以趁系統正在處理的空閒時間做其它的事情。

對於未來的研究方向，希望能夠在改進關於找 Transform 的方法，利用 ICP 並不能完全滿足讓實驗結果更精確的需求，因為 ICP 只能提供旋轉與位移的計算，對於縮放並不考慮，因此目前的方式是代入一堆參數（系統內定參數）做縮放，然後從中找到最佳的解，這在正確性方面可能會造成影響，因此對於未來目標是希望能套用一套更好的找 Transform 的方法。

參考文獻

[1]・T.F. Cootes and C.J. Taylor “Statistical Model of Appearance for Computer Vision,” Wolfson Image Analusis Unit, Imaging Science and Biomedical Engineering, University of Manchester, July 10, 2000.

[2]・Zhujie and Y.L. Yu “Face Recognition with Eigenfaces.”

[3]・Matthew A. Turk and Alex P. Pentland “Face Recognition Using Eigenface,” CH2983-5/91/0000/0586, 586 ~ 591,1991 IEEE.

[4]・Steven C. Mitchell, Johan G. Bosch, Boudewijn P. F. Lelieveldt, Rob J. van der Geest, Johan H. C. Reiber, and Milan Sonka “3-D Active Appearance Models: Segmentation of Cardiac MR and Ultrasound Images,” IEEE TRANSACTIONS ON MEDICAL IMAGING, VOL. 21, NO. 9, SEPTEMBER 2002

[5]・Axel Christian Varchmin, Robert Rae and Helge Ritter “Image Based Recognition of Gaze Direction Using Adaptive Methods,” http://citeseer.nj.nec.com/varchmin97image.html

[6]・Le-Tien Yeh “Registration of Drosophila Brain Neural Network in Standard Brain Model.”

[7]・Bahadir K. Gunturk, Aziz U. Batur, Yucel Altunbasak, Monson H. Hayes, III, and Russell M. Mersereau “Eigenface-Domain Super-Resolution for Face

Recognition," IEEE TRANSACTIONS ON IMAGE PROCESSING, VOL. 12, NO. 5, MAY 2003

[8]・T.F. Cootes, G.V. Wheeler, K.N. Walker, C.J. Taylor "View-based active appearance models," Image and Vision Computing 20 (2002) 657–664

[9]・G. J. Edwards, T. F. Cootes and C. J. Taylor "Advances in Active Appearance Models," IEEE International Conference on Computer Vision-Volume 1, September 20 - 25, 1999.

[10]・T.F Cootes, C.J Taylor, D.H Cooper and J. Graham "Active Shape Models-Their Training and Application." Computer Vision and Image Understanding Vol61, No. 1, Jan., 38-59, 1995.(http://www.wiau.man.ac.uk/~bim/Papers/cviu95.pdf)

[11]・Zhengwei Yang and Fernand S. Cohen "Image Registration and Object Recognition Using Affine Invariants and Convex Hulls," IEEE Transactions on Image Processing, Vol. 8, No. 7, July 1999.

[12]・Paul J. Besl and Neil D. McKay "A Method for Registration of 3-D Shapes," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 14, No.2, February 1992.

[13]・擷錄至財團法人國家實驗研究院國家高速網路與計算中心；網址 http://www.nchc.org.tw/Chinese_1/01_information/news_show.php?sn=260&mode=form，新聞內容，標題爲「國網中心協助建立全球第一個果蠅腦部基因表現資料庫 台灣首創腦神經網路虛擬實境全球服務」，日期爲 2004-02-18.

附錄

(Appendix)

A.0 程式流程

底下的 Fig. 27 是本程式的介面，整個程式的介面非常的單純，目的是

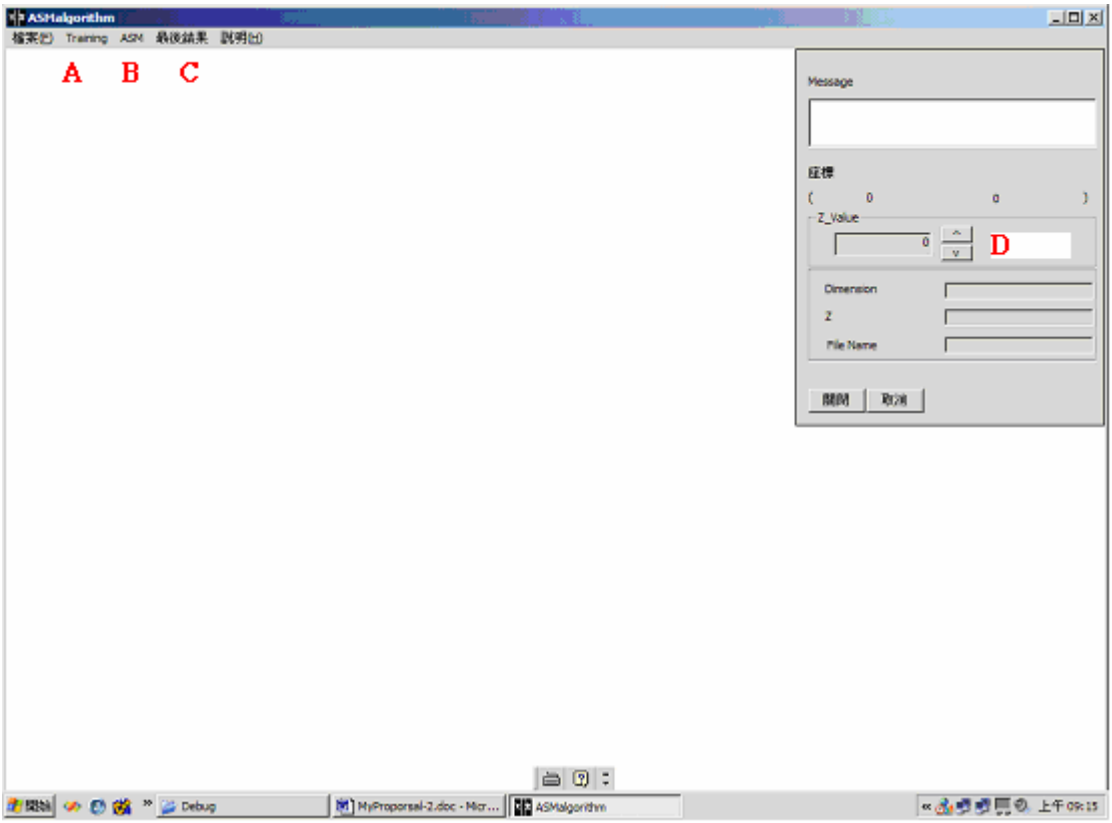


Fig. 27 程式介面

為了讓使用者能夠清楚的使用；整個程式主要分成兩大功能 A 與 B，至於 C 與 D 的功能則是屬於附加性質，底下將為各為解說：

A. Training：為 Training 出 Model 的部分，在 ASMW 中必須先 Training 出 Model 才能夠利用此 Model 來做物體的變形，這個部分又分成三個子細節，

a、 Open：開啟中軸檔案，一定要用兩個以上的中軸才能夠

Training 出 Model，另外檔案的格式是以*.txt 為主，且內容第一個一定要是點的個數，之後才以 ＸＹＺ 寫入，檔案格式如下所示：

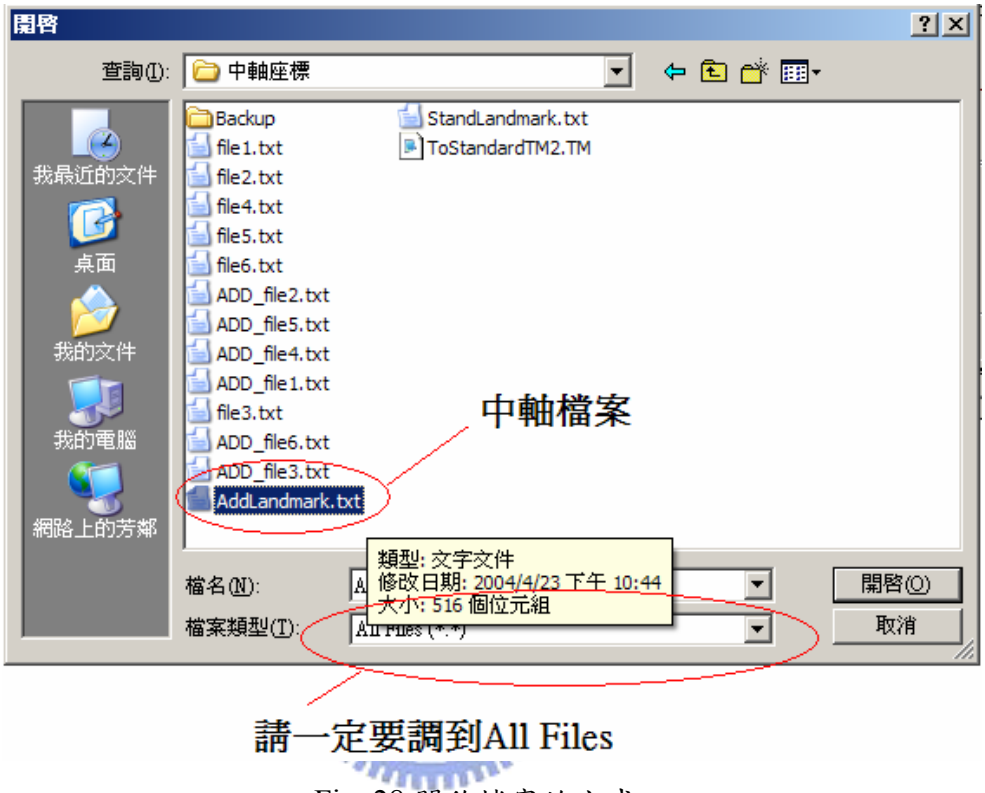


Fig. 28 開啟檔案的方式

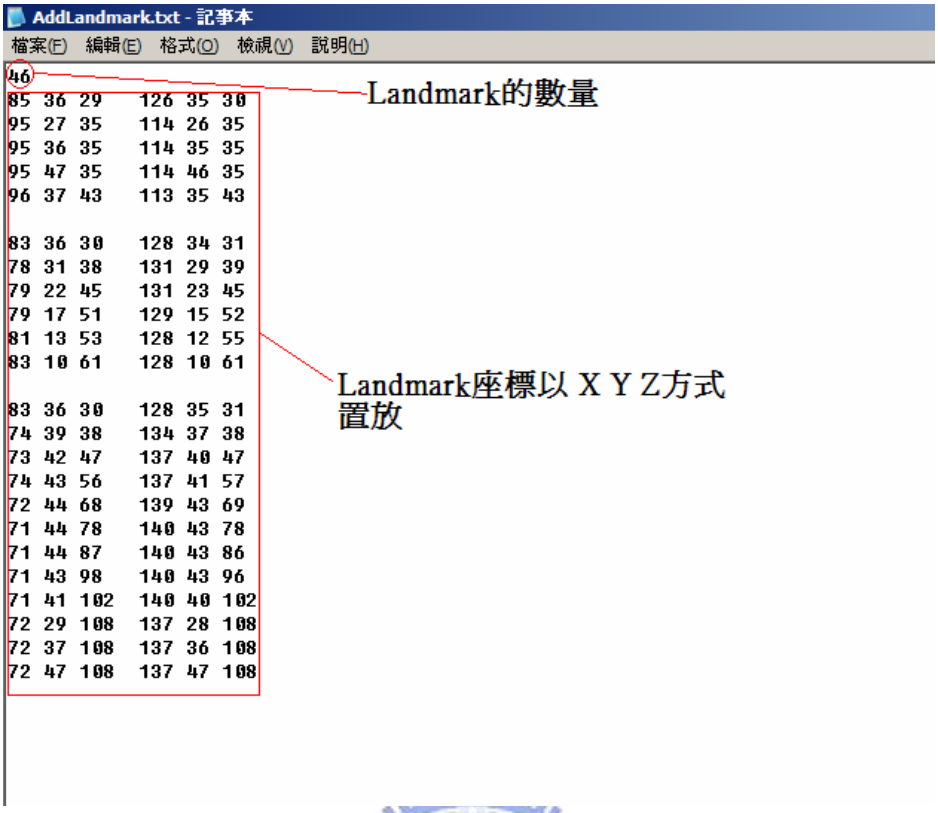


Fig. 29 檔案的格式

在設置點的時候有一點須注意到，每個 Sample 所設 Landmark 都必需相對應（Correspondence，如下圖所示，A,B,C 為 Landmarks），且數量要一致。

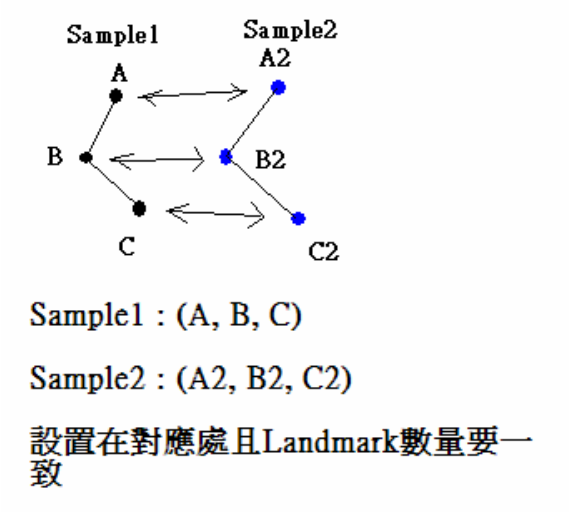


Fig. 30

b、 Save Landmark to Buffer：在 a 項中，程式尚未將資料輸入到 Training Set，這個步驟只是將 Sample 存入 Training Set 中。

c、 Make Model：計算出 Mean 與 Eigenvector 等資訊，並將最後結果存入使用者所指定的檔案，檔案為*.ini 與 *Eigenvector.dat、*Eigenvalue.dat 以及*Mean.dat 等檔案，其中*為使用者所鍵入的檔案名稱；另外後三個檔案因為系統是設在 Reference 的資料夾底下，因此使用前請先在 C 底下創造一個 Reference 的資料夾以方便系統正常運作。

B. ASM：此部分共有五個步驟

a、 Load Data from ini：

此步驟是載入 Model，前一功能中在最後的部分中提到 Make Model 這個步驟，對於 ini 並沒有稍加說明，所謂的 ini 檔就是紀錄 Model 的一些資訊如維度（Dimension X number of Landmarks），Eigenvector 以及 Mean 的資訊位在哪個檔案中，下頁 Fig. 31 即為 ini 檔的長相；Fig. 32 則為執行 Load Data from ini 以後所跳出來的對話方塊，請使用者一定要選擇*.ini 的檔案，否則系統將不會執行任何動作，並且出下錯誤訊息，如 Fig. 33 所示。

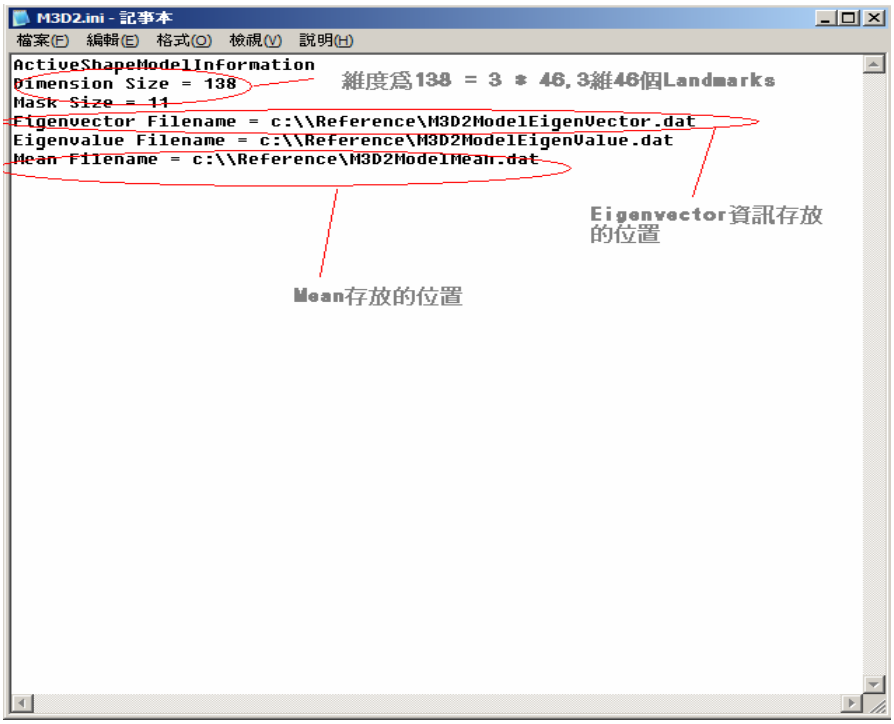


Fig. 31 *.ini 檔案的格式

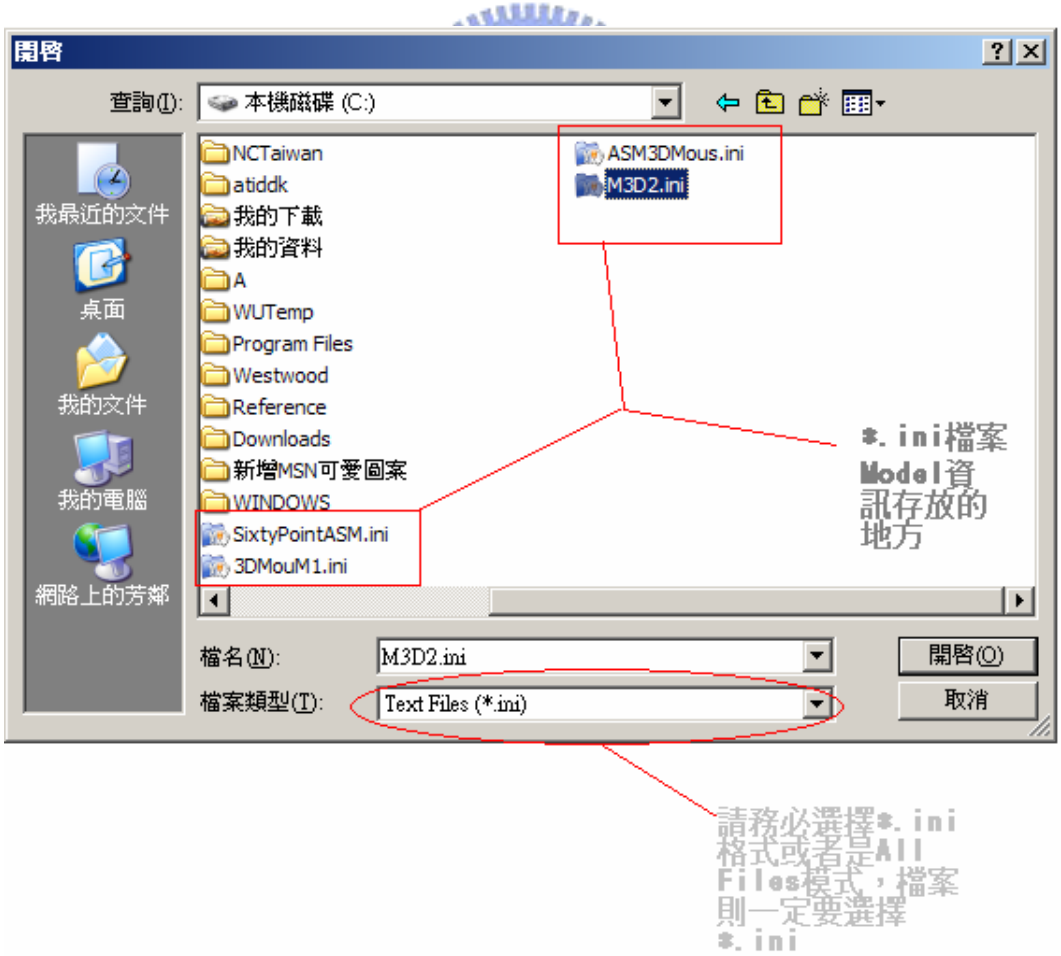


Fig. 32　執行 Load Data from ini 以後的對話盒

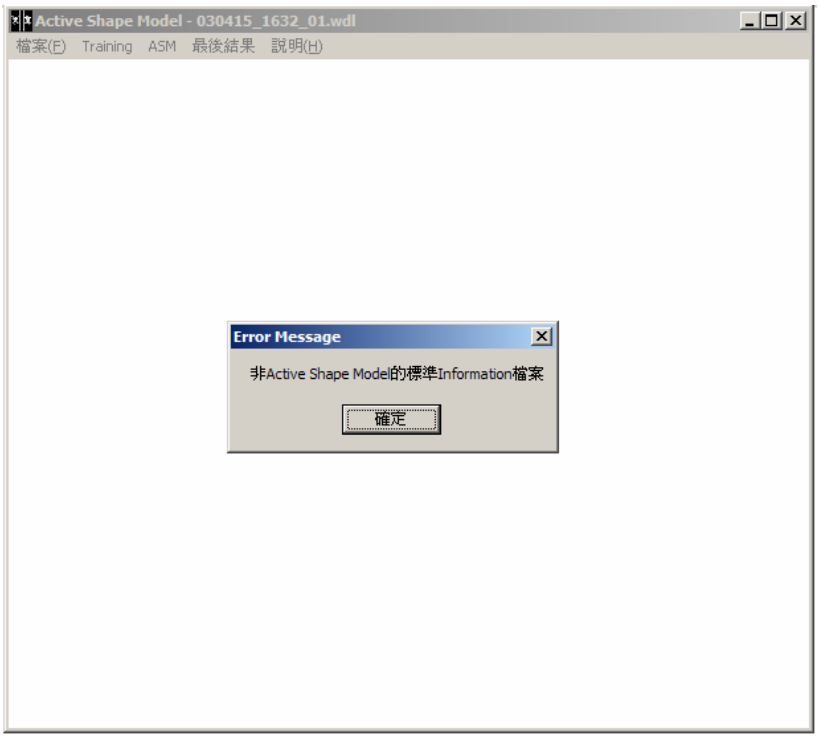


Fig. 33　打開非屬於 ASM 所創造的 ini 檔案的錯誤訊息

b、 Load Test Sample：

此步驟是載入欲測試的目標影像即在第四章所說的 $I_{Testsample}$，檔案格式是讀入 3D的Volume Data，格式為本實驗室（醫學影像實驗室）自創的*.IPT格式；本程式只支援IPT的檔案，對於其它格式一律不支援，下頁Fig. 34 為執行Load Test Sample所出現的對話方塊。

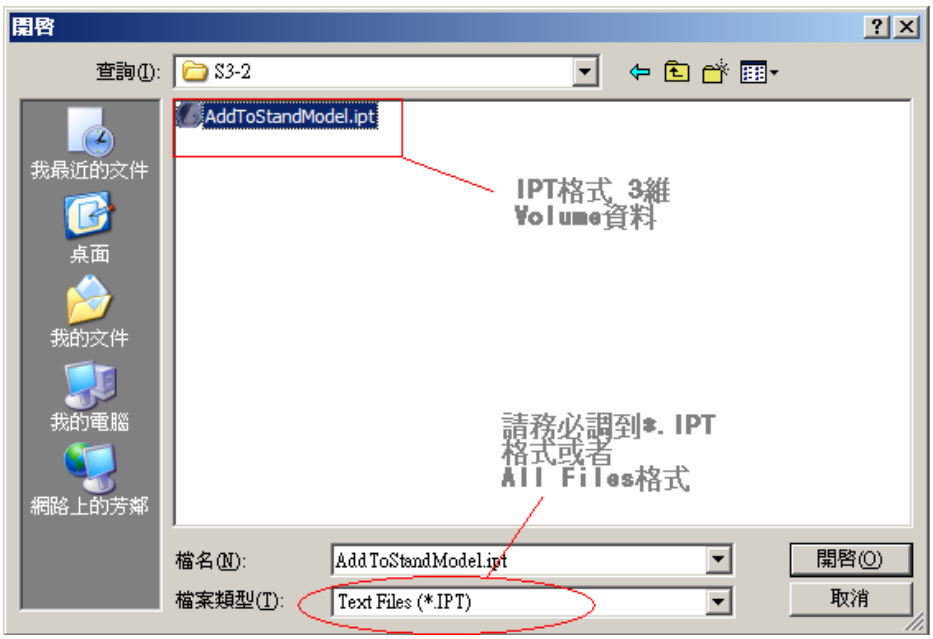


Fig. 34　執行 Load Test Sample 所出現的對話方塊

c、 Load Landmark of the Sample：

此步驟與上一步應屬一體的，此步驟是載入在上一步驟中所載入的影像的中軸檔案，與 Training　Model 時的中軸檔案的格式一樣（請見 Fig. 29）為*.txt 檔案，當中*代表此影像的中軸檔案名稱。

d、 To Model：

此步驟共分成兩部分，第一是先找出 Test Sample 與 Model 間的 Transform T 與 Model parameter b，第二則是利用這些參數做 Test Sample 變形（Warp）到 Model 上（方法請見 3.2.1 節）（此後有一額外步驟，請務必參考 C 部分）。

e、 To Target：

此步驟則是將變形到Model上的Test Sample變形到目標檔案，這裡我們需先載入事先算好的Transform $T_{ToTarget}$以及Model Parameter $b_{ToTarget}$，而$T_{ToTarget}$與$b_{ToTarget}$的資料都存放

在副檔名TM的檔案中（請見C部分），然後利用$T_{ToTarget}$與$b_{ToTarget}$的資料使的變形到Model上的Test Sample變形到目標物體，底下圖 35 為執行To Target後的對話方塊，要求使用者載入目標物體的Transform $T_{ToTarget}$以及Model Parameter $b_{ToTarget}$；當整個變形完成後所呈現的結

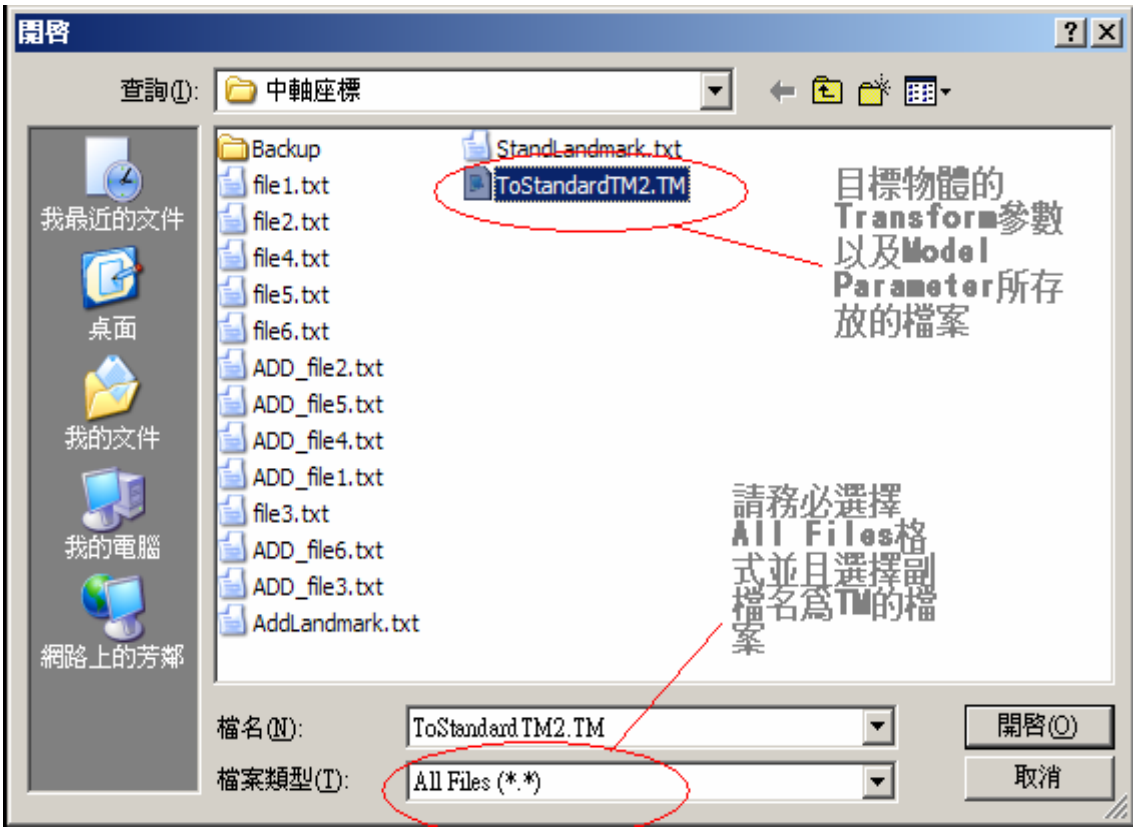


Fig. 35 執行 To Target 後所出現的對話方塊

果會以 2D 的方式畫出每一層的圖片，即是將 3D 影像的高度切成一片片的 2D 的畫面，如下頁圖 36 與 37 所示，Fig. 36 為第 24 層高度的結果，Fig. 37 為 27 層的結果。

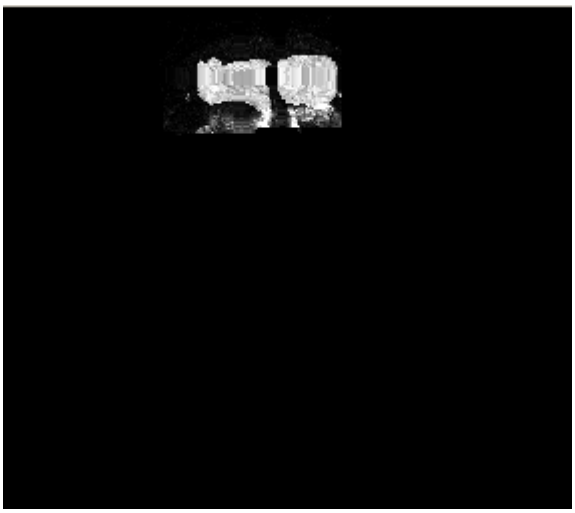


Fig. 36 執行變形後第 24 層的畫面

Fig. 37　第 27 層所呈現的結果

C.　這個部分總共有兩個主要功能，如下所示：

a、 Save Result To Raw：

這部分是將結果（不論是變形到 Model 或者是變形到目標物體上）存成一份固定格式的 Raw Data（Volume），而大小如何將由使用者所輸入為主，下頁圖為執行此步驟所出現的對話方塊。

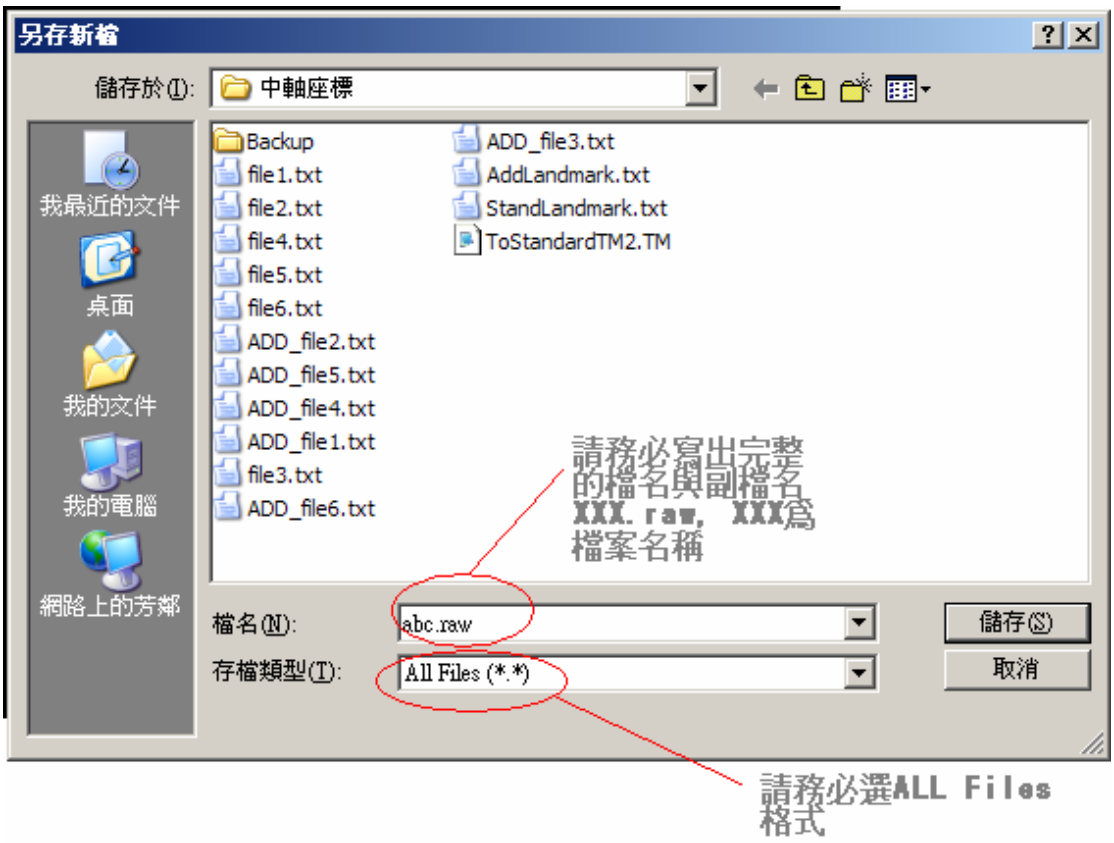


Fig. 38　執行 Save Result To Raw 功能後所出現的對話方塊

b、 Save Transform Magnitude：

這個部分是存入Model與Sample間的Transform參數T與Model Parameter b，因此當程式執行完To ASM以後，如果想將T與 b 參數存起來時，就需執行此步驟，以便將來要將某Sample warp到目標物體時使用，而存入的檔案的副檔名為TM，格式如下頁圖 39 所示，此檔案格式非常的單純，只是將$T_{ToTarget}$與$b_{ToTarget}$組成一個向量即TM = $(b_{ToTarget}{}^{T}, T_{ToTarget}{}^{T})^{T}$然後將它存入*.TM中，當中*為使用者鍵入的檔名（如Fig. 40 所示）。

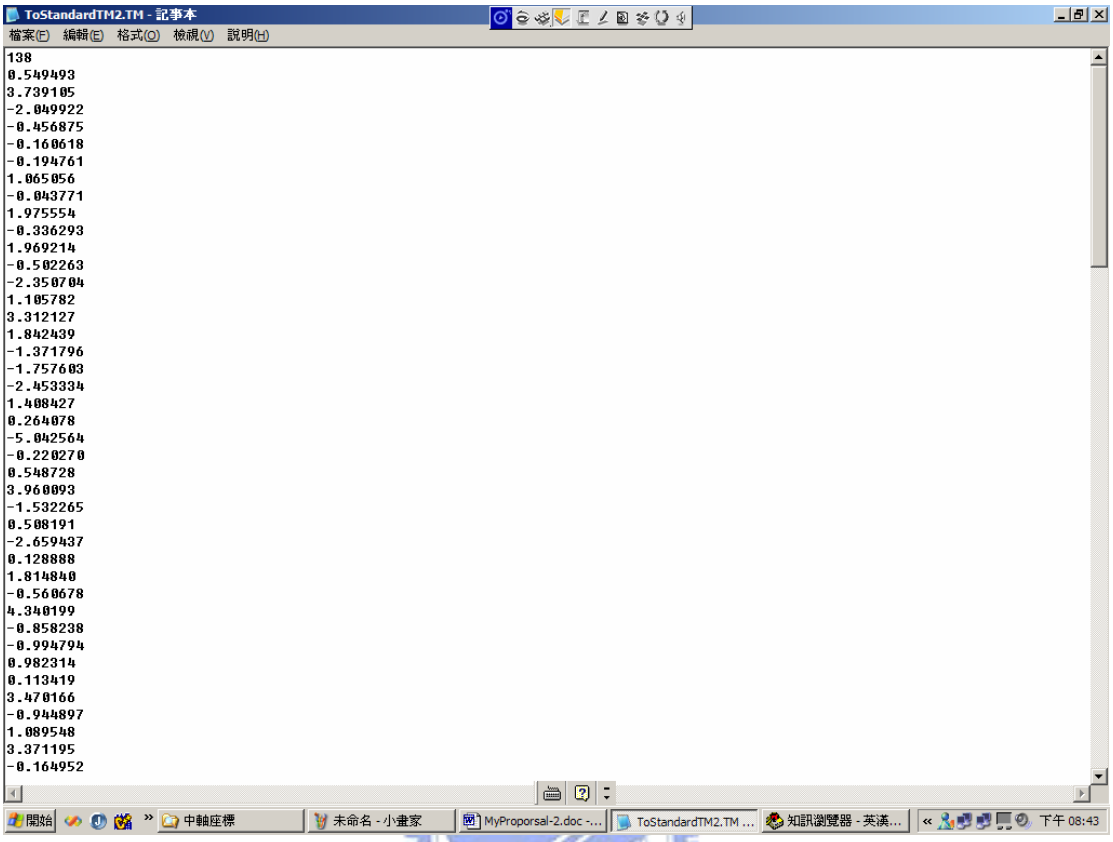


Fig. 39　TM 檔案的格式（僅附部份）

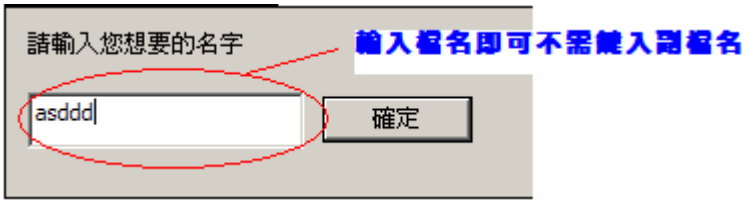


Fig. 40　TM 的對話方塊

上下鍵：此部分的功用僅是在完成變形後，讓使用者觀看變形後的結果，並沒有什麼特別用意，在這邊建議各位使用者，可以先將變形結果輸出到 Raw 檔後利用本實驗室林志陽學長的 IPToolBox 來觀看結果。