國立交通大學

多媒體工程研究所

碩 士 論 文

以粒子群最佳化技術重建稠密式物體三維模型

Dense 3D Reconstruction with Particle Swam Optimization

研 究 生：宋秉一

指導教授：陳 稔 教授

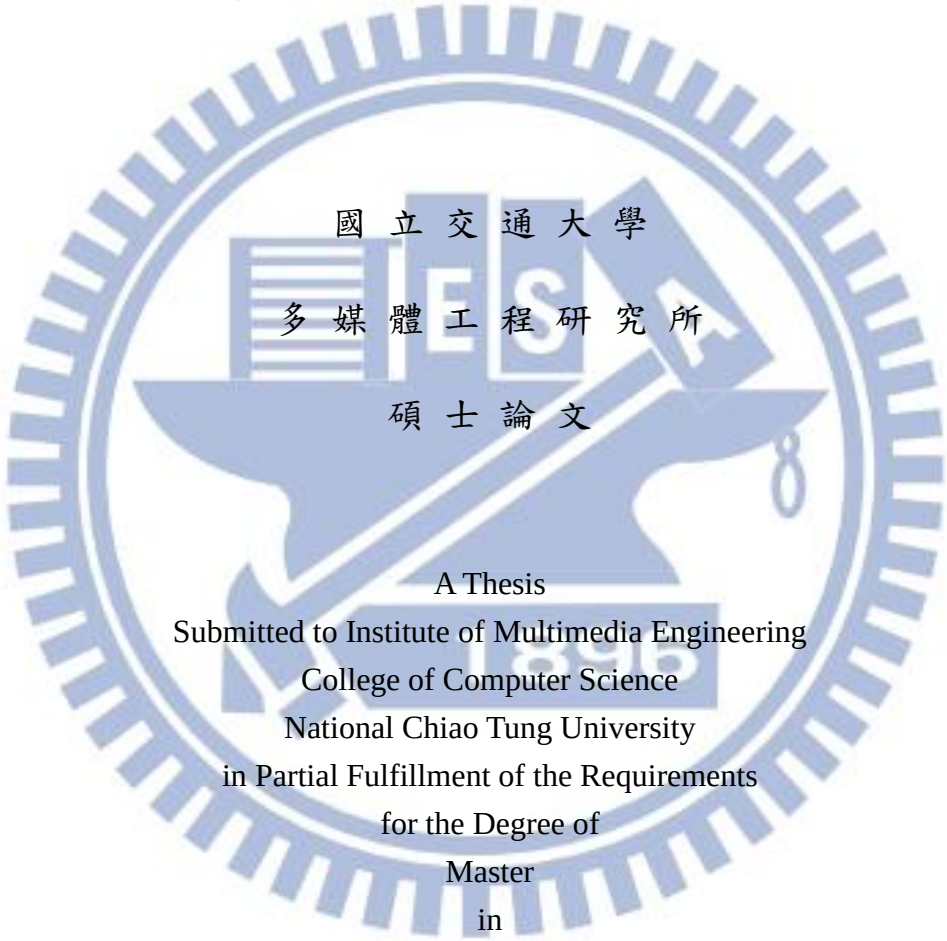
中 華 民 國 一〇一 年 七 月

以粒子群最佳化技術重建稠密式物體三維模型

Dense 3D Reconstruction with Particle Swam Optimization

研 究 生：宋秉一　　Student：Ping-Yi Sung

指導教授：陳　稔　　Advisor：Zen Chen



國 立 交 通 大 學

多 媒 體 工 程 研 究 所

碩 士 論 文

A Thesis
Submitted to Institute of Multimedia Engineering
College of Computer Science
National Chiao Tung University
in Partial Fulfillment of the Requirements
for the Degree of
Master
in
Computer Science

July 2012

Hsinchu, Taiwan, Republic of China

中華民國一〇一年七月

以粒子群最佳化技術重建稠密式物體三維模型

研 究 生：宋秉一　　指導教授：陳　稔

國 立 交 通 大 學

多 媒 體 工 程 研 究 所

摘要

本論文旨在利用 Particle Swan Optimization 的方式重建 Multi-view Stereo 多視角三維模型。本論文主要延伸 patch based 的 expansion 重建方式，分為幾大部分，分別為 camera calibration、seed patch optimization、patch expansion、patch filtering，並且透過 PSO 來提供一個 derivative free 的解法。

本論文特色之一是延伸 patch based 的概念並改進在較 sparse 的 camera view 之下提供完整重建，主要使用 geometric 與 texture matching 兩個限制來定義較多且可靠的 visible camera，並利用 GLN-PSO local search 的特性來進行有效率 patch optimization 與 expansion。此外對於 textureless 的物體，我們使用 multi-scale 的 pyramid image，相較於直接擴增 patch size 能夠提供更快的收斂速度。而對於非自然物的重建，我們則是利用 adaptive fitness weighting 來進行 patch optimization，以獲得預期的銳利轉折處。

Dense 3D Reconstruction with Particle Swam Optimization

Student： Ping-Yi Sung Advisor：Zen Chen

Institute of Multimedia Engineering

College of Computer Science

National Chiao Tung University

Abstract

This paper presents a stochastic optimization based multi-view stereo (MVS) approach for 3D dense reconstruction. We propose to apply adaptive weighted stereo matching functions to achieve more accurate optimization result. On the other hand, the reconstruction completeness falls short of the lack of enough visible views. We advocate allowing the child patch to borrow the parent visible view when needed even though the parent view is not in the specified viewing angle range. In addition, we shall adopt a GLN-PSO stochastic patch optimization method to avoid the local traps of a derivative based numerical optimization method. To improve the reconstruction quality we propose a patch priority queue to select the best patch to search for the next patch for the patch expansion process.

誌謝

首先誠摯的感謝指導教授陳稔博士，老師悉心的教導使我得以一窺電腦視覺與影像處理領域的深奧，不時的討論並指點我正確的方向，使我在這些年中獲益匪淺。老師對學問的嚴謹更是我輩學習的典範。

兩年裡的日子，實驗室裡共同的生活點滴，學術上的討論、言不及義的閒扯、讓人又愛又怕的宵夜、趕作業的革命情感、因為睡太晚而遮遮掩掩閃進實驗室.....，感謝眾位學長姐、同學、學弟的共同砥礪(墮落?)，你們的陪伴讓兩年的研究生活變得絢麗多彩。

感謝嘉峻、文昭、東哥學長們不厭其煩的指出我研究中的缺失，且總能在我迷惘時為我解惑，也感謝文威、蔡銘同學的幫忙，恭喜我們順利走過這兩年。實驗室的詠聖學弟當然也不能忘記，你的幫忙及搞笑我銘感在心。

女朋友小花在背後的默默支持更是我前進的動力，沒有小花的體諒、包容，相信這兩年的生活將是很不一樣的光景。

最後，謹以此文獻給我摯愛的家人。

目錄

圖目錄

表目錄

第一章 緒論 (Introduction)

1.1 研究動機 (Motivation)

近年來隨著電腦圖學與電腦視覺兩大技術的進步，透過如迪士尼 3D 動畫、Hollywood 商業電影特效等，一般民眾已開始大量接觸 3D 影像資訊。然而現階段的 3D 影像內容(content)皆需依賴 3D 影像專業人員以及專業建模軟體(如 3DStudio, Maya)製作產生，進入門檻相對地較高。若想將 3D 影像技術打入一般人的生活當中，勢必要讓 3D 內容能被輕易的製作。因此，利用民眾隨手可得的 2D 平面影像來建置 3D 影像資料即是我們近年來努力發展的方向，期望能用最平易近人的方式，得到更豐富的 3D 影像內容，有效增加三維模型的使用機會及應用情境。本研究則是使用發展相機從不同角度拍攝的多張影像進行三維物體模型重建的技術，透過相關演算法的改進以得到較準確的物體重建結果。

1.2 相關研究 (Related Work)

在最近幾年，從多張不同角度的平面影像，利用 Multi-View Stereo (MVS) algorithms 來重建 3D 模型的技術有大幅度的發展。Seitz 等人更設立了一 public website 提供 data set 進行 benchmark 評量 [1] [2]。MVS algorithms 主要分成幾大類：patch-based, volume-based…等。由於 MVS 的計算量很大，所以在近兩年有多位學者利用 parallel hardware 來加速計算 MVS [3] [4]，甚至用於 real-time 的應用 [5]。

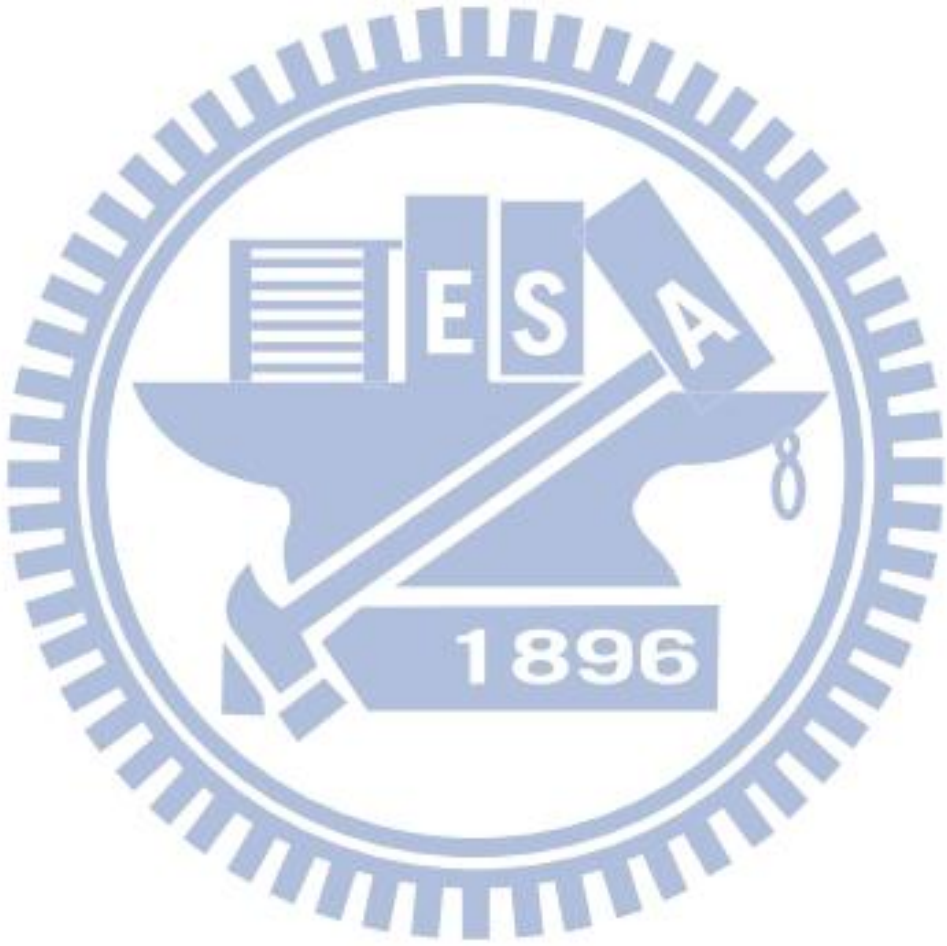
目前絕大部份的 MVS 技術都是針對 general purpose 來設計 [6] [7] [8]，其中最為知名的是 Furukawa et al. 的方法，以 patch 為單位，估測出於空間中的 3D 位置及其 normal 方向，透過多次的 expansion 及 filtering 來處理 noisy 的重建結果。此方法在 Middlebury 的 benchmark data set 當中，得到了相當不錯的結果。另一方面，Habbecke et al.則是使用 region growing 的方式，假設物體表面為

smooth 的變化，穩定地從較正確的 region 向外 expansion 並重建，而無 Furukawa et al.的 filtering 動作。前述的方法皆適用於 well controlled 的拍攝環境，如 [2] 中的 data set，其相機不管數目多寡，皆 uniform 地分佈在空間中，對於實際拍攝物體時，較難有如此控制良好的環境。目前的 MVS algorithms 皆聲稱非針對 uniform 相機擺設而設計，然而，在本論文的實際測試發現，如 Furukawa et al. 的方法，在有較完善的相機擺設會有較佳的重建結果，若在一些資訊較缺乏(如 texture-less, specular)的物體表面無較多的影像來參考時，容易重建出 noisy 的 3D model，或是 completeness 較不足的結果。本論文則是針對 non-uniform sparse camera arrangement 問題，透過較穩定的 feature detection and matching，以及 novel photo consistency 計算方法，可以有效地解決此問題。

在利用 MVS algorithms 重建物體時，多會有 smoothness constraint [9]，因此在重建如四方體這種有 sharp edge 時，會因為在計算 patch 投影至影像上 window 的 photo consistency 時，因為 window 包含了不同物體平面的 planar patch，重建出的結果多會是 smooth 的表面，無法正確重現物體邊緣 sharp 的變化。本論文亦提出了 adaptive weighted window algorithm，可以解決 mixture of planar patches 的問題。

透過最大化 photo consistency 的值來重建物體表面的 MVS algorithm，其核心技術皆是利用 optimization 來求得某一局部表面 patch 的最佳空間位置及 normal 方向，或是透過如 graph cut [10] [11] [12] [13], level-set [14] [15] [16] [17] 或 deformable models [18] [19]的 global energy minimization approaches。然而在設計要最佳化的 equation 時，往往會面臨到幾個問題：(1)使用 numerical optimization 時，容易進入 local trap，造成計算出的結果並非是最佳解，也就是會建出 noisy 的 data，還需要其他方法(如 filtering)來處理。(2)無法設計出複雜的最佳化算式，造成欲最佳化的算式無法微分，將使得此方面的技術無法往更精確的方向發展。本論文則是採用 stochastic optimization 演算法中的 Particle Swarm Optimization

(PSO)來避開 local trap，此外也使用更適用於 image-based 3D modeling 的 GLN-PSO 演算法，GLN-PSO 演算法是標準 PSO 演算法的延伸技術，同時使用了 global、local、以及附近鄰近粒子的最佳位置來更新每一粒子的速度。此外，在做如 Furukawa 的 expansion 時，我們考慮不同的因素，如 photo consistency value, texture correlation, number of visible cameras, level of detail 來評斷 expansion 的 priority，如此可以重建出更正確的結果。



1.3 論文架構 (Thesis Organization)

圖 1 為本論文的整體流程架構圖，第二章首先說明特徵點(feature point)的擷取與多視角影像之間的對應，並分成已校正(calibrated)及未校正(un- calibrated)的情況，在未校正的情況下，先採用 structure from motion 演算法取得相機空間位置與特徵點的相對關係。當多視角的特徵對應點決定後做為種子點(seed point)。第三章主要說明如何用 patch 來描述物體表面，以及如何利用各相關資訊來提昇各 patch 的深度(depth)及法向量(normal)之準確度。利用 PSO 進行最佳化的過程則在第四章中說明。第五章描述如何由種子點往外擴展，並提出了一種新的擴展順序(Patch Priority)以保持物體的準確性。在建出初始模型後，則利用第六章的過濾演算法去除 3D 雜訊點。相關的實驗結果及正確性驗證於第七章中所呈現，最後第八章提出本研究的結論與未來展望。

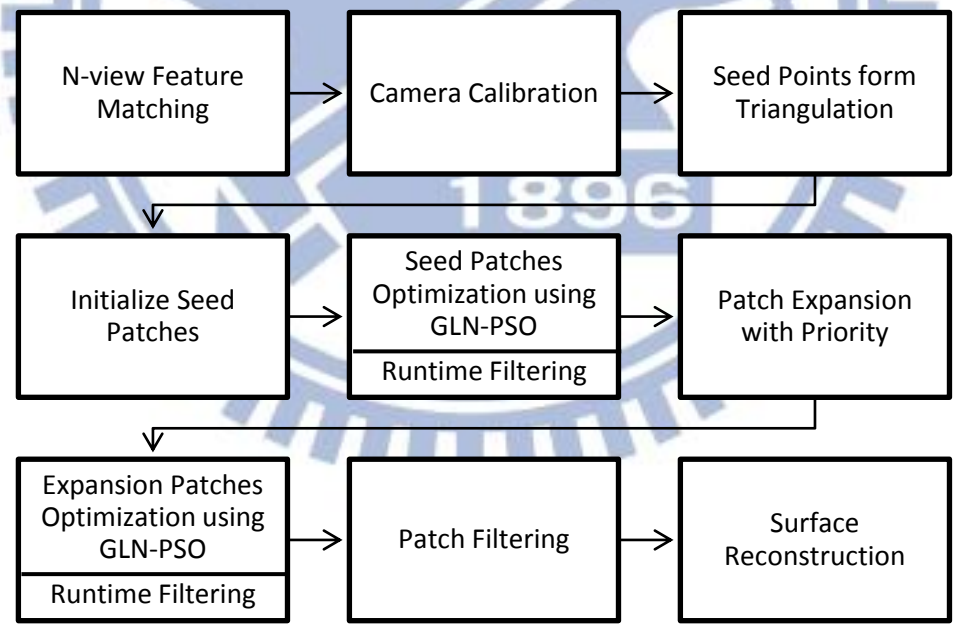


圖 1. 論文架構圖

第二章 相機校正與特徵點偵測 (Camera Calibration and Feature Detection)

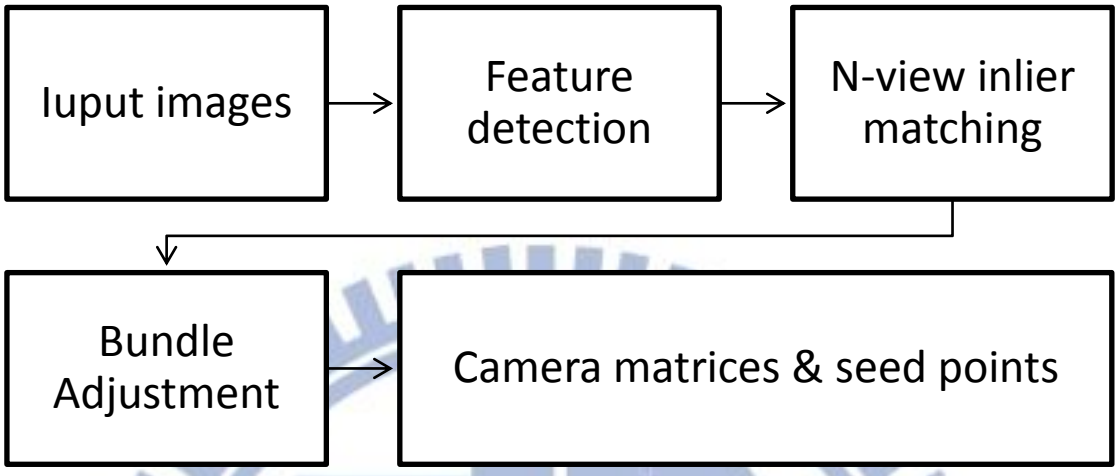


圖 2. 相機校正與特徵點偵測流程圖

對於 un-calibrated 的場景，輸入該場景的多張影像，首先使用 Scale-invariant feature transform (SIFT)分別對各影像偵測 feature point，並使用 Random Sample Consensus (RANSAC)與 Epipolar line constrain 做 n-view inlier matching 找出影像間 robust 的 inlier feature correspondence。利用 inlier feature correspondence 的對應，使用 bundle adjustment [20]估測各影像的相機參數與 inlier feature 的 3D point，本論文將這些 3D point 作為 seed point，並在之後的步驟以 seed patch 做 expansion。

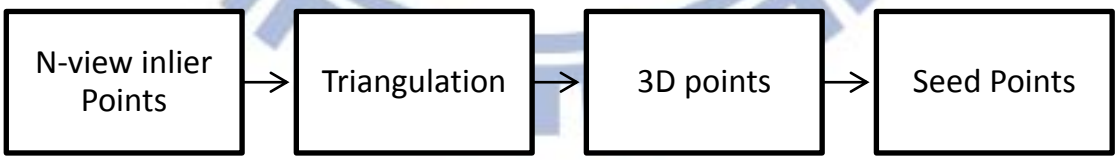


圖 3.Seed point 建立流程圖

並且我們的方法也可以相容 calibrated 的場景，對 calibrated 的場景，我們仍然使用 n-view inlier matching，並使用 calibrated camera matrices 對應 linier feature 的 3D point。我們會保留 n-view inlier matching 的結果作為幫助後面 visible camera selection 的依據。

第三章 平面最佳化 (Patch Optimization)

3.1 平面中心點與平面(Patch Center and Patch)

本論文假設每一個 3D point 都可擴張出一塊以該點為中心的 patch p，一個 p主要有三個參數構成，分別為 patch center $c(p)$、patch normal $n(p)$ 與 patch extent u，我們的目標就是精準地在真實物體的表面上佈滿 dense 的 patch。並在最後階段使用 Poisson surface reconstruction 或 Delaunay triangulation 將封閉的完整模型建出。我們的 seed patch 是由第二章求得的 seed point 而來，除了已知的 patch center $c(p)$，我們另外定義了 patch normal $n(p)$ 與 patch extent u，使 seed point 延伸為 seed patch。

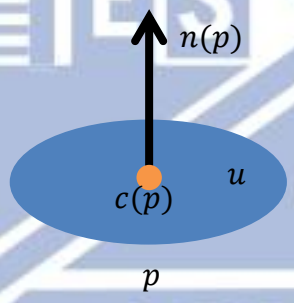


圖 4. Patch 示意圖

我們將重建過程中的 patch 分為兩大類，分別為 seed patch 與 expansion patch；其中 seed patch 是由 seed point 延伸而來的穩定 patch，而 expansion patch 則是由 seed patch 或是其他 optimized expansion patch 向外擴增產生。

3.2 平面法向量與其範圍 (Patch Normal and Normal Range)

對 seed patch 使用已知的 visible camera $V(p)$，定義 $c(p)$ 到相機中心 $O_i(p)$ 連心射線之平均和向量為 $n(p)$，其中 $|V(p)|$ 表 $V(p)$ 集合的數量。而 expansion patch 的 $n(p)$ 則是繼承 parent patch 的 optimized $n(p)$，爾後再進行 optimization 調整。其中 parent patch 是由 priority queue 中取出，可以是 seed patch 或其他 expansion patch，將在第五章詳細敘述。

$$n(p) = \frac{1}{|V(p)|} \sum_{i \in V(p)} \frac{O_i(p) - c(p)}{\|O_i(p) - c(p)\|}$$

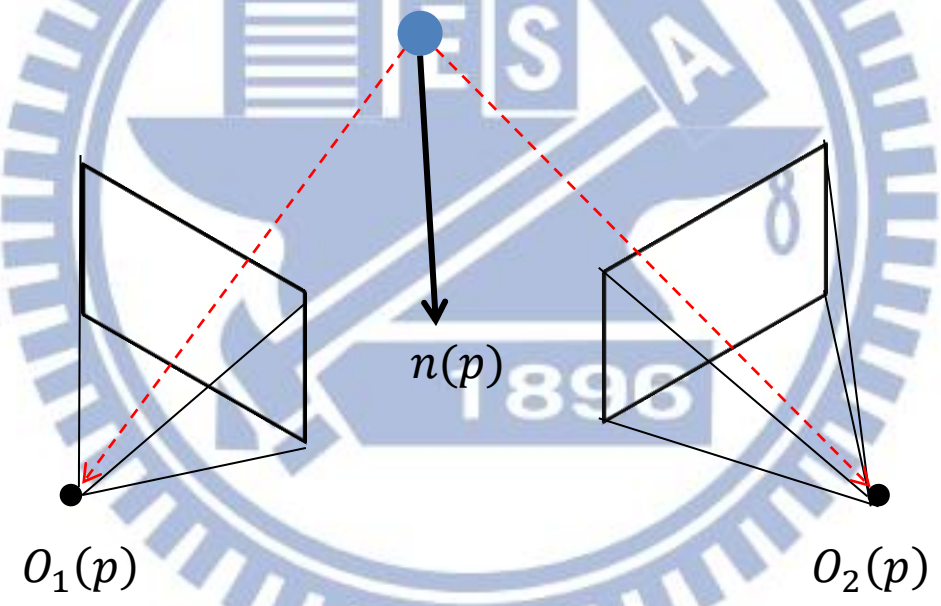


圖 5. Initial patch normal 示意圖

PSO 較不易受到初始 initial guess 而左右其收斂結果，但需要指定參數範圍來做 optimization search。本論文將 normal 以 spherical coordinate 表示，normal 表示由 3 維降至 2 維。定義 θ 與 φ 分別為 spherical coordinate 的垂直與水平夾角；根據 back-face culling 與 front-face intersection 的概念，不可能存在 $n(p)$ 背對相機的 patch，我們可以再簡化 normal 的搜尋範圍，φ 範圍定義在所有 $V(p)$

光軸的正面 180 度交集區。

$$\varphi_{range} = \bigcap_{i \in V(p)} \varphi_i(p)$$

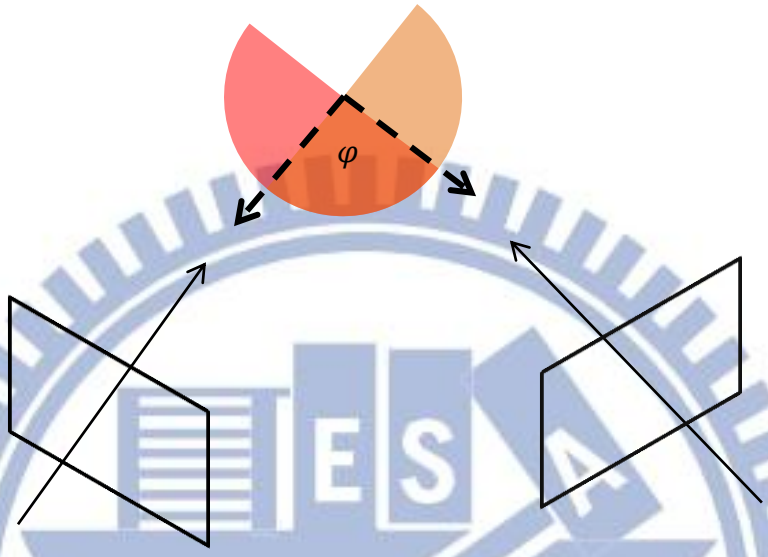


圖 6. φ 範圍交集區域示意圖

3.3 平面延伸範圍 (Patch Extent)

Patch extent Ω 以 p 的 reference image $I_{R(p)}$ 決定，p 投影在 $I_{R(p)}$ 上為一個半徑 u pixel 的圓形 region (圖 7)。實作上我們在 $I_{R(p)}$ 上的投影點取正負 u pixel 的距離定義一個邊長 $2u + 1$ pixel 的 extent window，而圓型的 region 範圍則是利用 distance weighting 來 mask 去除圓型範圍外的影響力，這部分將在 3.9 節詳細敘述。由於投影點與 extent window 不一定為整數值，本論文皆使用 bilinear interpolation 來內插非整數 pixel 的 intensity。

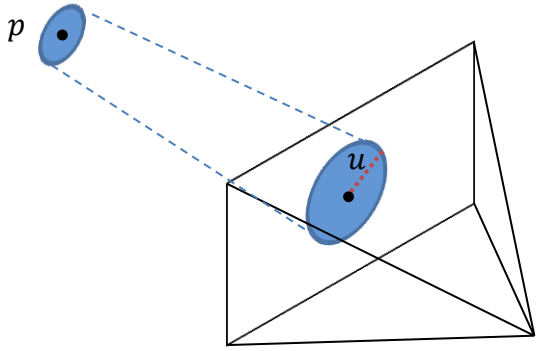


圖 7. Patch extent 示意圖

3.4 平面可視相機與參考相機 (Patch Visible Cameras and Reference Camera)

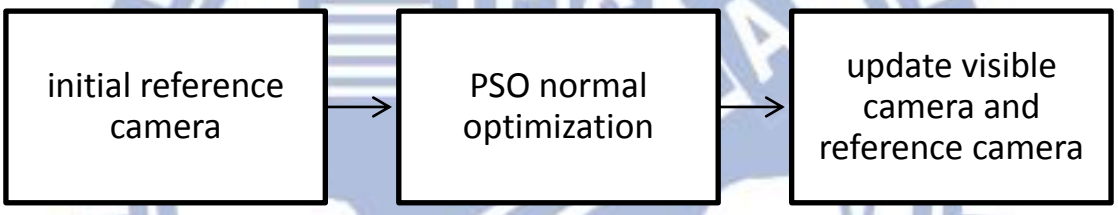


圖 8. Reference camera 建立與更新之流程圖

對 seed patch，我們以 n-view inlier matching 的 linier camera 作為 visible camera $V(p)$，並以光軸與 $n(p)$ 夾角最小的 $V(p)$ 作為 reference camera $R(p)$。

對 expansion patch，我們預設 $V(p)$ 為光軸與 $n(p)$ 夾角 correlation 大於 ω 的 camera view，但在較為 un-uniform 的 camera view 分布時，在 ω 限定的範圍中可能不足 V_{min} 個 camera view 時，這時則繼承 parent patch 的 $V(p)$ 與 $R(p)$ 資訊，再執行 invisible camera filtering。

在每次完成 patch optimization 之後，我們會利用 optimized patch texture correlation $\gamma(p)$ 來檢測是否有 invisible camera 並刪除之，避免 expansion 擴增的 $V(p)$ 中有 invisible camera，同時更新光軸與 optimized $n(p)$ 夾角最小的 $V(p)$ 作為 $R(p)$。

3.5 平面材質相關性 (Patch Texture Correlation)

對於每個 patch，我們可以衡量該 patch 在 $V(p)$ 影像中的 texture correlation，藉此判斷 patch 對應的好壞，以便我們在 expansion 時決定 patch priority。將 patch window 經 homography 投影取樣的 intensity 轉為 unit vector t，並計算 t 的 correlation 之平均 $\gamma(p)$。

$$\gamma(p) = \frac{\sum_{i,j \in V(p), i \neq j} t_i \cdot t_j}{|\{i, j \in V(p), i \neq j\}|} = \frac{\sum_{i,j \in V(p), i \neq j} t_i \cdot t_j}{|V(p)|^2 - |V(p)|}$$

3.6 不可視相機過濾 (Invisible Camera Filtering)

對於 $V(p)$ 的選擇，若在 optimization 時含括了 invisible camera，則收斂的 patch 結果會受到影響失去意義，所以我們建立的 patch 都必須確保所有 visible camera 都是有效的。我們這裡使用 patch 在兩影像間的 correlation 來決定是否該要刪去 invisible camera。首先我們先挑選 correlation 和最高的 camera view 作為 reference correlation camera $V_\gamma(p)$，並刪除 $V(p)$ 中與 $V_\gamma(p)$ 的 correlation 小於 α 的 camera view。在 camera filtering 後，若 $|V(p)|$ 有所改變，我們就會再重新執行 optimization 與 camera filtering 直到 $V(p)$ 收斂，若發現 $|V(p)|$ 小於 V_{min} 則丟棄這個 patch。

$$V_\gamma(p) = arg \max_{i \in V(p)} \sum_{j \in V(p), i \neq j} t_i \cdot t_j$$

$$V(p) \leftarrow \{i \in V(p), t_i \cdot t_{V_\gamma(p)} > \alpha\}$$

3.7 平面深度及深度範圍 (Patch Depth and Depth Range)

我們將 patch center 的位置以一個 depth 參數調整，參考由 reference camera center $O_{R(p)}$ 射出到 $c(p)$ 的單位向量 $r(p)$，並乘上一個 depth $d(p)$ 作為 scalar，這樣可以將 $c(p)$ 的座標由 3 維簡化成 1 維。

$$d(p) = \left\| c(p) - O_{R(p)} \right\|$$

$$r(p) = \frac{c(p) - O_{R(p)}}{d(p)}$$

而 depth 調整範圍則是利用非 $R(p)$ 的其他 visible camera 來限制，我們限制 $d(p)$ 調整投影後最大不會在任何 $V(p)$ 影像中與原本的調整前的位置相差 β 個 pixel。我們將 $V(p)$ 影像的 pixel 轉換為在 $r(p)$ 射線上的真實世界單位 Δ^{-1}，其中 Δ 表在$r(p)$射線上一世界單位投影在影像上的距離，而 β 則與 cell size τ 有關，原則上 τ 越大則 β 也越大，因為相鄰 cell 中的 $d(p)$ 也越不連續，需要更大的搜尋範圍。若 $V(p)$ 中有相機與 $R(p)$ 相當接近，我們稱為 narrow view case，則 Δ 會得到一個趨近 0 的數字，取倒數的真實距離範圍會變為正負無窮大，在這麼廣的範圍中就無法有效的搜尋到最佳解，這時的深度範圍就不會考慮該 view，在我們的實驗中則是跳過了 Δ 小於 0.01 pixel 的 camera view。

$$d_{range}(p) = d(p) \pm \max_{i \in V(p), i \neq R(p), \Delta \geq 1} \beta \Delta^{-1}, where$$

$$\Delta = \left\| P_i\big(c(p) + r(p)\big) - P_i\big(c(p)\big) \right\|$$

3.8 細節層次 (Level of Detail)

對於 textureless 的 surface，由於 textureless 沒有鑑別力，我們可以將其視為 repeated pattern 的極端情況，故容易收斂到錯誤的解，重建出錯誤的 surface；最直覺的改善方法就是增加 window 的大小，使其包含有 texture 的影像資訊。Furukawa 的方法中使用 9x9 或 7x7 的小 window，其優點是計算有效率，但對於 textureless 的 surface 就會錯誤，他使用多次 iterative 的 expansion 與 filtering，並遞減放鬆每一次 iterative filtering 的 threshold 標準使其收斂，利用由穩定 parent patch 的資訊向外 expansion 以達到覆蓋整個 model surface 的效果。Habbecke 的方法中設定了一個 window size 的上下限，預設使用最小的 window size，並統計 window 內的 intensity variation，當 variation 不足時則增加 window size；這個方

法能夠抵抗 textureless 的 surface，但相對較大的 window 也較沒有效率。

本論文的方法受 Furukawa 與 Habbecke 的啟發，目的是要利用小 window 的計算效率與大 window 較豐富的 texture 資訊，希望在效率與準確度中取得更好的平衡，我們事前建立各影像 scale ratio 為 ε 的 pyramid，並使用固定的小 window，並在 intensity variance Δv 不足時切換取樣較小的 texture。其中 Ω 表 patch extent 在 $R(p)$ 影像的取樣點集合 (point set)，$|\Omega|$ 為取樣點總數。

$$\Delta v = |\Omega|^{-1} \sum_{(x,y)\epsilon\Omega} \left(I_{R(p)}(x,y) - \overline{\Delta v}\right)^2 \text{ , where}$$

$$\overline{\Delta v} = |\Omega|^{-1} \sum_{(x,y)\epsilon\Omega} I_{R(p)}(x,y)$$

$$l(p) = arg\min_{l} \Delta v > v$$

由於現今相機的高畫素影響，平均一張照片都有千萬畫素以上，對於要重建高品質的模型，早期增加 window size 的方法產生的 overhead 影響會更大(而 middlebury 與 Habbecke 的解析度約為 30~80 萬畫素)，而對這些情形我們使用 level of detail 的方式建立 image pyramid。雖然需要更多記憶體來儲存影像，但對於 optimization 速度有顯著的改善。

3.9 自適應適合度權重 (Adaptive Fitness Weighting)

由於 optimization 的 patch size 是基於 reference image 上的 window 求 homography 投影到其他 visible image 中，而 window 在 image 上很有可能同時包含了非預期的其他相鄰平面，若將各 pixel 的 weighting 視為相同，會使 optimization 收斂到一個 average difference 的結果，而在物體銳角邊緣交界處就會產生非預期的重建結果。同時為了要能在 textureless 物體上正確比對 patch 參數，必須要削減 uniform texture 的 pixel 之影響力，並加強比對 edge 處及 intensity 不同處的結果。要改善這些情形，我們認為 window 中每一個 pixel 應給予不同的 weighting。

我們目的是重建 $c(p)$ 所在的 patch，所以我們希望在 optimization 同時將 homography difference 集中到真實的物體平面範圍外，這裡我們考慮與 distance to center、homography difference 與 edge gradient magnitude 三個因素做 weighting。在這裡我們定義 $(x,y) \in \Omega$ 為 p 投影至 $R(p)$ 影像上投影點的座標，Ω 表 p 的 patch extent 範圍。

Homography difference weighting $h(x,y)$ 可使 optimize 的 pixel 重分配，其中 $\Delta T(x,y)$ 是影像 intensity 之 difference，而 distance weighting $g(x,y)$ 則保證了要 optimize 接近 patch center 的範圍，也就是我們期望 fitting 的 surface。其中 (x_0, y_0) 為 $c(p)$ 投影點位置。

$$h(x,y) = e^{-\frac{|\Delta T(x,y)|^2}{\sigma_h}}, (x,y) \in \Omega$$

$$g(x,y) = \frac{e^{-(\frac{(x-x_0)^2}{2\sigma_g} + \frac{(y-y_0)^2}{2\sigma_g})}}{\sum_{u,v \in \Omega} e^{-(\frac{(u-x_0)^2}{2\sigma_g} + \frac{(v-y_0)^2}{2\sigma_g})}}, (x,y) \in \Omega$$

最後 edge gradient magnitude weighting $k(x,y)$ 目的是要抵抗 textureless 的影響，避免 optimization 掉入 local trap，對於位在 edge 的 pixel 給予較大的 weighting，而位於 uniform region 的 pixel 則給予較小的 weighting，在本論文使用 3x3 的 Sobel operator 來偵測 edge gradient，並且將整個影像的 edge gradient normalize 至 [0,1] 之間作為 weighting。其中 $*$ 表示 convolution operator，G_x, G_y 分別表橫向與縱向的 Sobel operator。

$$k(x,y) = \exp\left(\left(\frac{\left\|I(x,y) * G_x + I(x,y) * G_y\right\|\sigma_k}{max\left(\left\|I(\Omega) * G_x + I(\Omega) * G_y\right\|\right)}\right)^{-1}\right), (x,y) \in \Omega$$

最後我們將這三個 weighting 相乘並 normalize 後作為每個取樣 pixel 的 adaptive fitness weighting $w(x,y)$；相關的實驗結果會在 7.2 節中討論。

$$w(x,y) = \frac{g(x,y)k(x,y)h(x,y)}{\sum_{u,v \in p} g(u,v)k(x,y)h(u,v)}, (x,y) \in \Omega$$

3.10 Homography 投影與平面最佳 (Homography Projection and Optimization)

若已知 patch 參數 $n(p)$ 與 $d(p)$，Ω 為 patch extent 在 $I_{R(p)}$ 影像的座標，我們最小化 Ω 投影到各 $I_{V(p)}$ 的 texture difference，藉此來最佳化 $n(p)$ 與 $d(p)$ 參數。計算 $O_{R(p)}$ 到 $(x,y) \in \Omega$ 射線與 patch plane 的交點，令這些交點為 patch 之取樣點，並投影到所有 $I_{V(p)}$ 上取得 texture intensity。

這個取樣可分為由 $I_{R(p)}$ 投影至 patch plane 與投影至 $I_{V(p)}$ 兩個步驟，並且同時考慮 level of detail 在不同 scale 的影像取樣，故我們需將相機內部參數乘上一個 intrinsic scalar $L(p)$，我們稍微修改 Habbecke 的 homography matrix，將這兩個步驟合併為一個由 $I_{R(p)}$ 影像到其他 $V(p)$ 間的 homography $H_i(p), i \in V(p)$。

$$L(p) = \begin{bmatrix} \varepsilon^{-l(p)} & 0 & 0 \\ 0 & \varepsilon^{-l(p)} & 0 \\ 0 & 0 & 1 \end{bmatrix}; \varepsilon = 0.8, l(p) = 0,1,2,\dots$$

$$P_i = L(p)K_i[R_i\ T_i] = [M_i\ m_i]$$

$$D(p) = \frac{-n(p)^T c(p)}{\|n(p)\|}$$

$$H_i(p) = (D(p)M_i - m_i n(p)^T)(D(p)M_{R(p)} - m_{R(p)} n(p)^T)^{-1}$$

Texture difference 理想上是使用 normalized cross correlation (NCC)來計算；這裡為了加速 fitness 的計算過程，我們計算 Ω 中各 pixel 在不同 view 之 intensity 與 mean intensity 的 absolute difference $\Delta T(x,y)$，如下

$$\bar{T}(x,y) = |V(p)|^{-1} \sum_{j \in V(p)} I_j \left(H_j \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \right)$$

$$\Delta T(x,y) = |V(p)|^{-1} \sum_{i \in V(p)} \left| I_i \left(H_i \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \right) - \bar{T}(x,y) \right|$$

並且對 $\Delta T(x,y)$ 加入 weighting 的作為 fitness $\xi(p)$。

$$\big(n(p),d(p)\big)=arg\min_{n(p),d(p)}\xi(p),where\ \xi(p)=\sum_{(x,y)\in\Omega}w(x,y)\Delta T(x,y)$$

3.11 最佳化策略 (Optimization Strategy)

在 seed patch 的 optimization 時，由於還未有 normal 資訊，為避免掉入 local minimum，在這裡 PSO 使用 uniform distribution 散佈 particle 位置，並插入 initial normal 與 depth 作為其中一個 initial particle。

對 expansion patch，先假設其與 parent patch 同平面，定義 initial normal 與 parent patch 相同，且延伸 parent patch 的平面範圍作為 initial depth。由於 expansion patch 僅需要微調參數，這裡的 PSO optimization 為了加速收斂，同樣插入由 parent patch 求得的 initial particle。

在 expansion 的過程中，我們加入了夾角相近的 camera 作為 visible camera，但由於物體本身遮擋的關係，加入的 camera 可能看不到該 patch，這會讓 optimization 的過程受到其他 texture 影響 patch 參數的收斂結果，對於 invisible camera 造成的影響，我們會在所有 patch optimization 之後利用 correlation 檢查並刪除 invisible camera，並重新執行 optimization 直到 visible camera 收斂，或是當 visible camera 不足時則刪除該 patch。

第四章 粒子群最佳化 (Particle Swam Optimization)

4.1 粒子群最佳化概述 (Introduction to Particle Swam Optimization)

為了要達到精準的重建結果，我們的 patch optimization 含括了 adaptive weighting 與有可能改變的 visible camera 及 level of detail，fitness function 將變成一個難以微分的 non-linear optimization problem，因此我們這裡使用 Particle Swam Optimization (PSO) [21]來求解；相較於 Furukawa 使用 conjugate gradient 之類的 numerical optimization method，PSO 對於 initial guess 的依賴性低，並且也不易因為 initial guess 的好壞掉入 local trap。

使用 PSO 來求解 non-linear problem，其優點是不需要推導參數矩陣的一次微分(Jacobian)及二次微分(Hessian)，提供一個 derivative free 的演算法，僅需寫出每個獨立 particle 的 fitness function，並且受惠於現今電腦多核心的平行架構，相較於標準的循序的 Levenberg – Marquardt Algorithm (LMA)解法仍然有不錯的收斂速度。

PSO 的目的就是找到一組 D 維的參數 x 來最小化 fitness function $\xi(x)$，在這個 D 維度的參數空間散布 I 個 particle 來進行取樣，並根據本次的取樣結果作為下一次取樣的參考，並紀錄每個 iteration 的最佳 particle，iterative 執行直到滿足收斂或停止條件。而最後一個 iteration 的最佳 particle 就是我們所求的最佳解。

$$\arg\min_{x} \xi(x)$$

4.2 GLN-PSO

GLN-PSO [22]是 PSO 的一種改進演算法，目的在於透過更多的參考點來加速 particle 的收斂與 local search 的效率及準度。與傳統 PSO 相同，GLN-PSO 的 particle 移動也參考了 global best (gbest)、personal best (pbest)以及 inertia 方向，GLN-PSO 則是在這些之外加入了 local best (lbest)及 near neighbor best (nbest) 的參考點。

Local Best 概念上就是尋找鄰近最好的 pbest；將目前該代每個 particle 的 pbest 與目前該 particle 計算距離，取前 k 近中 fitness 最小的 pbest 即為 lbest。Near neighbor best 概念上是分別估測每個維度較好的座標。實作上，尋找所有其他 pbest 在各維度的座標，與目前點的差值對 fitness 比。並以滿足 fitness-distance-ratio (FDR)比值最大的 pbest 維度座標作為 nbest 在該維的座標。

$$n_{id} = \arg\max_{p_{jd}} \left\{ FDR = \frac{\xi(X_i) - \xi(P_j)}{|x_{id} - p_{jd}|} which\ i \neq j \right\}$$

GLN-PSO 的移動向量計算如下式，其中 v_{id} 表示第 i 個 particle 的第 d 維移動向量；p_{id}、g_{id}、l_{id}、n_{id} 及 x_{id} 各表示 pbest、gbest、lbest、nbest 及 current 在第 i 個 particle 的 d 維之位置；w、c_p、c_g、c_l、c_n 則各表示 inertia、pbest、gbest、lbest 及 nbest 移動權重；u 則是一個 uniform distribution 介於 $[0,1)$ 的隨機實數。

$$v_{id} = wv_{id} + c_p u(p_{id} - x_{id}) + c_g u(g_{id} - x_{id}) + c_l u(l_{id} - x_{id}) + c_n u(n_{id} - x_{id})$$

4.2.1 加速常數 (Acceleration Constant)

對於 GLN-PSO 中各參考向量的權重選擇，w、c_p、c_g、c_l、c_n 各表示 inertia、pbest、gbest、lbest 及 nbest 移動權重，因為乘上一個 uniform distribution 介於 $[0,1)$ 的隨機實數，權重即表示各向量的權重最大值。對於每個 particle，w 與 v_{id} 控

制該 particle 的慣性移動，並且由於 v_{id} 初始範圍涵蓋整個參數空間，在第一次 iteration 時可使各 particle 跳躍移動到空間任意位置，對於超過參數範圍的 particle 則強制限制其位置在範圍內；並且考慮到收斂性，我們使 w 隨著 iteration 呈線性遞減到一個不為 0 的最小值 w_L。由於不同的參考向量有可能互相削弱其方向強度，為了要確保讓 particle 往 global minimum 移動，我們會使 c_g 的大小略高於其它 acceleration constant 來加速收斂。

4.2.2 粒子數與迭代次數 (Particle Number and Iteration Number)

PSO 演算法需指定一個最大的 iteration 次數，程式在執行 particle number * iteration 次數的 fitness 計算就會停止。Particle number 的數量必須要至少能有意義的覆蓋到參數空間中；舉例來說，同樣是 1000 次的 fitness 計算，可有 20*50 或 2*500 兩種組合，但明顯 20 個 particle 跑 50 iteration 才有 exploration 的意義。並且我們可以藉由 convergence condition 來提早結束已經收斂的最佳解，這部分將在下一節提到。

4.2.3 收斂性分析 (Convergence Analysis)

PSO 類型演算法的收斂主要需同時滿足 dispersion index $\bar{\delta}$ 與 velocity index $\bar{v}$ 之 threshold，利用簡單直覺的物理限制來判斷是否收斂，其概念是對已收斂的 iteration 而言，所有 particle 應與 gbest 距離相近，且所有粒子的 velocity 應接近 0；即所有 particle 應相近且不再移動。Dispersion index $\bar{\delta}$ 與 velocity index $\bar{v}$ 計算公式如下所示，其中 I 代表 particle 總數，D 代表維度。

$$\bar{\delta} = \frac{\sum_{i=1}^{I}\sum_{d=1}^{D}|x_{id} - p_{gd}|}{I \cdot D}$$

$$\bar{v} = \frac{\sum_{i=1}^{I}\sum_{d=1}^{D}|v_{id}|}{I \cdot D}$$

4.3 GLN-PSO 優勢分析 (Comparison between GLN-PSO and PSO)

然而 GLN-PSO 的 local best 計算牽涉到 k-nearest neighbor 的 indexing 問題，與傳統 PSO 的計算相比，若是在 high dimension 或 large number of particles 時，勢必會產生一些 overhead，但由於我們將問題空間縮小到 3 維，相對需要的 particle number 也較少；再者 fitness 計算在需要進行大量 pixel 的 homography 投影與 bilinear interpolation 取得在其他影像的 intensity，因此 indexing 的成本遠低於任何一次 fitness 計算；並且在 patch expansion 的過程中，我們已知 parent patch 得來的 initial guess，僅需做些許調整就可收斂，而 GLN-PSO 加強的 local search 特性能幫助 expansion 加速，故使用 GLN-PSO 可減少收斂代數且有效地加速整體的計算效率與準度。

第五章 平面擴增 (Patch Expansion)

5.1 平面優先權 (Patch Priority)

從實驗結果我們發現，patch 的 expansion 若是考慮了 priority 則會有較佳的連續性與收斂結果；在 Furukawa 的 PMVS 中，他們實作了高度平行化的 optimization process，並且每個 patch 彼此間都互為 independent，缺點是 patch 間需要經過多次的 filtering 與 re-expansion 才能有較佳的連續性。而 Habbecke 的方法中同時在 growing 的 region 僅有一個，僅在 growing region 無法再向外 expansion 時才注入一個新的 seed region，filtering 的過程利用與 parent patch 的關係在 runtime 就已決定，因此擁有較佳的 surface 連續性。基於平行化的結構與考慮 surface 連續性，我們認為所有 patch 都有其 priority，並且要由 priority 較高的 patch 先行 expansion。我們經實驗觀察發現 priority 較高的 patch 其 expansion patch 的 priority 也較高，因此利用 priority 來決定先後順序可較有意義的填補穩定的 surface，並且也先確定有信心的 patch 參數作為下一個 expansion 的 parent patch。

每個 patch 的 priority 定義為 $q(p)$，數值越小則 priority 越高，考慮下列幾點分別是 fitness、correlation、visible camera number 與 level of detail。我們認為 texture 對應較佳以及 visible camera 較多的 patch 相對穩定，而 level of detail 較低的 patch 也含有較多的 texture，故給予較高的 priority。

如下式，$\xi(p)$ 表該 patch 經 adaptive weighting 的 fitness、$\gamma(p)$ 表該 patch 與 visible images 投影的 correlation、$|V|$ 與 $|V(p)|$ 則分別表系統中的 camera number 與 visible camera number；$l(p)$ 表以 0 開始的 level of detail。其中a, b表兩個可調的 weighting 參數 (預設皆為 1)。

$$q(p) = (l(p)+1)\xi(p)\exp\left(-\frac{\gamma(p)}{a} - \frac{|V(p)|}{b|V|}\right)$$

首先，我們將 seed patch 放入 priority queue 中，並依照 patch priority 由高至低取出作為 parent patch 進行 expansion，而 expansion 產生的 child patch 也會放

入 priority patch 中進行排序，重複直到 priority queue 為空時整個重建則結束。

5.2 影像網格 (Image Cell Map)

本論文的最終目的是在物體表面上佈滿 dense 的 patch，為了要能夠衡量重建的完整性以及 expansion 鄰近的新 patch，我們將影像切割為 cell size τ pixel 的 cell map Q_i；τ 定義了 patch 在空間中的密度，數值越小則密度越高，描述物體也越精細，但也相對需要更多時間來重建整個模型，在有效的取樣密度下，τ 的最小值可以是 1 pixel。當所有 priority queue 取出的 parent patch 都無法再向鄰近 cell expansion 時程式就結束。而可能有 object image regions 與其四周有陡峭牆壁，而自身又無 seed points，未被掃到的地方就為空缺不重建。

將 p 投影到所有 visible image 上的 cell $C_i(p), i \in V(p)$，並紀錄該 cell 包含哪些 patch。Expansion 的過程嘗試使每一個 cell 都至少有一個 patch，但為了在 patch filtering 步驟確認 cell 中 patch 的正確性，我們允許一個 cell 能夠有複數個 patch。為了計算效率問題，我們在實作上仍然加入了一個 cell 所能容納的最大 patch 數量 C_{max}，當大於或等於 C_{max} 則該 cell 就算完成 expansion。

5.3 擴增相鄰網格之平面 (Expansion of Neighboring Cells)

我們嘗試在 $C_i(p), i \in V(p)$ 的 one-ring neighbor cells $N_i(p)$ 共 8 個方向做 expansion，但不包含三種情況，第一是若 neighbor cell 中已有一個已經長好的 patch 且與目前 patch 相近時，第二是 neighbor cell 已有一個不相鄰但可信度很高的 patch 時，第三則是當這個 neighbor cell 已經達到 patch 數量上限 C_{max} 時。

在第一種情況時，由於 neighbor cell 已經有一個經過最佳化後仍可能會調整到相近參數的 patch，我們可以直覺的省略該 neighbor cell 的 expansion。我們投影兩個 patch 中心點到對方平面，並計算平均是否小於 ρ，若實為共平面，則平均距離會相當接近 0，反之則會越大。如下式，其中$|(c(p_1) - c(p_2)) \cdot n(p_1)|$為

$c(p_2)$到p_1之距離，相對的$|(c(p_1)-c(p_2))\cdot n(p_2)|$則是$c(p_1)$到$p_2$的距離。

$$\mathrm{n}(p_1,p_2)=\frac{|(c(p_1)-c(p_2))\cdot n(p_1)|+|(c(p_1)-c(p_2))\cdot n(p_2)|}{2}<\rho$$

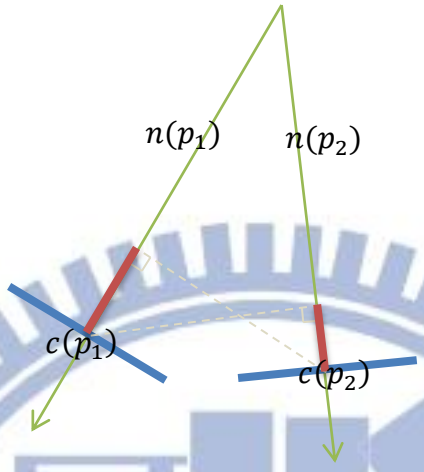


圖 9. 相鄰 patch 距離示意圖，計算兩個紅色線段的平均長度作為判斷基準

第二種情況則常發生在 depth 不連續的地方，雖然兩個 patch 距離很遠，但由於該處也有一個很穩定的patch，實作上我們判斷若 $\gamma(p)>\alpha$，則不expansion。

第三種情況則是為了要加速 expansion 過程，理論上我們希望可以從同一個 cell 中建立多個 patch，來幫助我們判斷 patch 是否需要 filtering，若 $|V(p)|$ 很大則會造成一些 overhead，所以我們這裡僅取到足夠的 patch 數量 C_{max} 即可。

第六章 平面過濾 (Patch Filtering)

6.1 執行階段之平面過濾 (Runtime Patch Filtering)

在 seed patch refinement 及 expansion 的過程中，可能會有收斂到不合理的 patch，根據下面幾個原則，我們會直接丟棄這些 patch，並且也不加入 priority queue 中。

首先，因為我們在 patch optimization 後進行 invisible camera filtering，若 $|V(p)| < V_{min}$，則丟棄 p。第二，由於在 expansion 可能加入了 invisible camera，故有機會造成 texture trap，經實驗結果發現，這些 trap 的 view，patch homography 投影的面積都相對細長，與我們定義方型的 window size 差異甚多。我們計算 homography 後的 window 最短邊與 patch size 之比率，若比率小於 λ 則刪除該 view，並且同時檢查前述的條件是否符合。第三，若影像有提供 silhouette 資訊，我們也會進行 visibility test，並刪除投影在 silhouette 之外的 patch。

6.2 後處理平面過濾 (Post-Processing Patch Filtering)

在 expansion 的過程中，我們嘗試在每一個 cell 中至少建立一個以上的 patch，並且在 priority queue 的 patch 都被檢查過後結束整個 expansion 過程，但並無法保證每個 patch 都是正確的，除了一些能在執行時 filter out 的限制，其他 filtering 需要在整個 model 重建結束後，這時已獲得整體模型的大致輪廓後再執行後處理。

6.2.1 深度測試過濾 (Depth Test Filtering)

我們的第一個 filter 來自於 cell 的 depth test，我們假設同一個 cell 的 patch 都表示同一個模型平面，根據 z-buffer test 的概念，我們刪去 p 在同 cell 中 $d(p)$ 非最小的 camera view，並在其後檢查是否 $|V(p)| < V_{min}$ 來決定是否捨去 p。

6.2.2 平面相關性過濾 (Patch Correlation Filtering)

第二個 filter 是根據 normalized correlation $\gamma(p)$ 來決定，若 patch 的 $|V(p)|$ 與 $\gamma(p)$ 相對比其他同 cell 的 patch 還小，則將 p 視為 outlier 刪除，下面的判斷式成立則刪除 p。

$$|V(p)|\gamma(p) < \sum_{p' \in C_i(p), i \in V(p)} \gamma(p')$$

6.2.3 相鄰網格過濾 (Neighboring Cells Filtering)

第三個 filter 再另外考慮相鄰 cell 中的 patch 相關性，檢查 p 所在及相鄰 cell 中的所有 patch，一個穩定的 patch 應有足夠的 neighbor patch 支持。利用 5.3 提到的 $\mathrm{n}(p_1, p_2)$ 確認是否為 neighbor patch，並計算是否有超過 $\boldsymbol{\Lambda}$ 的相鄰 cell 中之 patch 為 p 的 neighbor，下面的判斷式成立則刪除 p。

$$\frac{\left|\mathrm{n}(p, p_j) < \rho; p_j \in N_i(p)\right|}{\sum_{i \in V(p)} |N_i(p)|} < \boldsymbol{\Lambda}$$

第七章 實驗結果 (Experiments)

在本章中的實驗，我們以 Intel i7 2600 3.4Ghz 的四核心電腦進行重建，在計算時由於加入了 level of detail 的 pyramid 架構，最大消耗的記憶體有 6GB (Dino dataset 共 363 張影像)，所以須使用 64bit 的作業系統方能定址大容量記憶體。

7.1 GLN-PSO 與 PSO 比較分析 (Analysis of GLN-PSO and PSO)

我們比較在同樣參數下，GLN-PSO 與 PSO 的收斂品質，棋子模型的 10266 個 seed patch 做實驗，為 $w = 0.8$、$c_p = 1.2$、$c_g = 1.5$、$c_l = 1$、$c_n = 1$與 k=5，particle number 為 15 進行 60 次 iteration。

為了要加速收斂速度與品質，我們在初代時加入了 initial 參數作為其中一個 particle。實驗目的是驗證 initial 參數是否在初代就提供了較佳的預估解，並且比較有無加入 initial 參數之結果。實驗以 seed patch 的 optimization 來觀察並紀錄最後的 fitness 與收斂代數。

Type	Average Fitness
GLN-PSO w/o initial particle	5.0581
GLN-PSO with initial particle	**4.1242**
PSO w/o initial particle	5.5116
PSO with initial particle	4.3865

表 1. GLN-PSO 與 PSO 之比較表

Type	Average Fitness	Average Iteration
GLN-PSO with initial particle	**4.1242**	41.3885
GLN-PSO w/o initial particle	5.1300	40.2145

表 2. GLN-PSO 有無加入 initial particle 之比較(convergence threshold 0.01)

實驗結果顯示，在相同的參數下，GLN-PSO 提供了較佳的收斂結果，並且無論在 GLN-PSO 或 PSO 加入 initial particle 的情況下，收斂結果都有改善；而在使用 convergence threshold 提早結束收斂的情況下，GLN-PSO 可以提早在 41 代達到與 60 代相當的解；故我們的演算法中採用 GLN-PSO 作為 optimization 方法，並且加入 initial particle 與 convergence threshold。

7.2 自適應適合度權重分析 (Analysis of Adaptive Fitness Weighting)

這部分我們以一個 synthetic concave cube 來進行一個小實驗，我們給予一個較大的 window size (61x61)使其容易含括其他平面範圍，藉此來模擬比較有無使用 adaptive weighting (distance、difference、gradient) 的差別。實驗結果發現，若是含括其他平面的範圍遠小於 window 範圍的 1/2 時，兩種方法的差異並不會太大；但若是含括了多個平面或是範圍接近甚至大於 1/2 時，有使用 adaptive weighting 的收斂結果明顯較好。

而在 real dataset 的實驗中，由於我們參考了 edge gradient magnitude 作為 weighting，同時也增強了對於 textureless 的抵抗性，發現有使用 adapted weighting 的 patch 較不易因為 major uniform texture 而掉入 local trap。

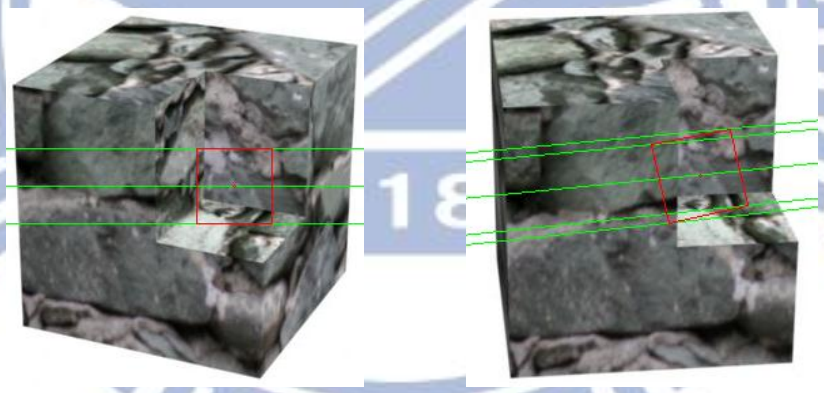


圖 10. 一個跨平面 window 的 example，左圖為 reference image，右圖為其中一個 visible image，紅色中心點標示 patch center 的投影位置，紅框表示 homography 的投影範圍

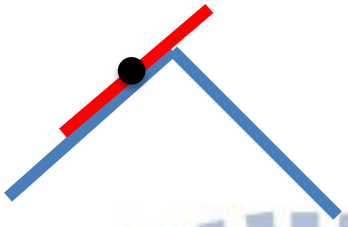
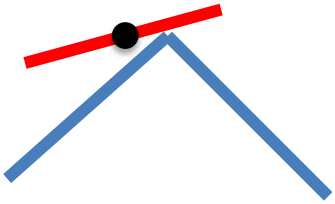
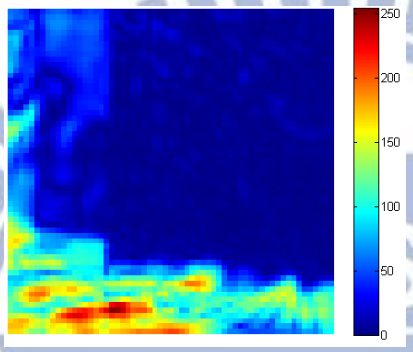
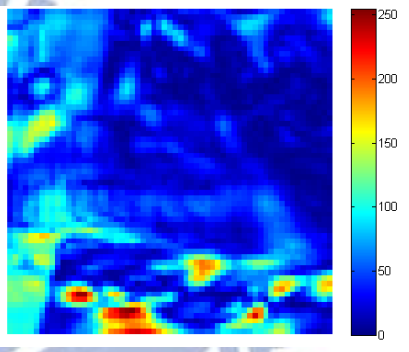
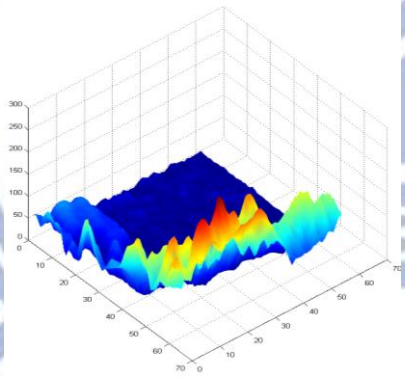
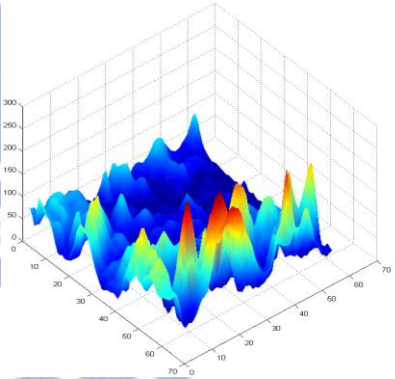
	With adaptive weighting (ours)	Without adaptive weighting (Furukawa's)
重建結果示意圖		
normalized pixel-wised sum of mean difference		
		

表 3. 有無使用 Adaptive Fitness Weighting 比較表

7.3 使用合成影像重建三維模型 (Reconstruction of Synthesis Images)

7.3.1 建立合成影像棋子模型 (Reconstruction of Synthesis Pawn Model)

本實驗以 24 張解析度 500x800 的棋子合成影像 (圖 11) 進行重建，已知相機內外部參數，經由 SIFT 做 n-view matching 產生的 seed point 共有 10266 個，經過 runtime filtering 後的 seed patch 剩餘 6490 個。我們以這些 seed patch 開始重建 dense patches。由於合成影像的 silhouette 是可得的，故在重建模型時我們引入了 silhouette 資訊。

在相同的 cell size 參數與 patch 密度下 (cell size 皆為 2)，與 Furukawa 的重建結果比較，我們的方法提供了相對較完整的 patch 分布 (圖 14)，並且在經過 Poisson surface reconstruction 的重建後還能完整的表示模型輪廓 (圖 15)。

圖 11. Pawn dataset 棋子影像由左至右為第 1、10、20 張影像

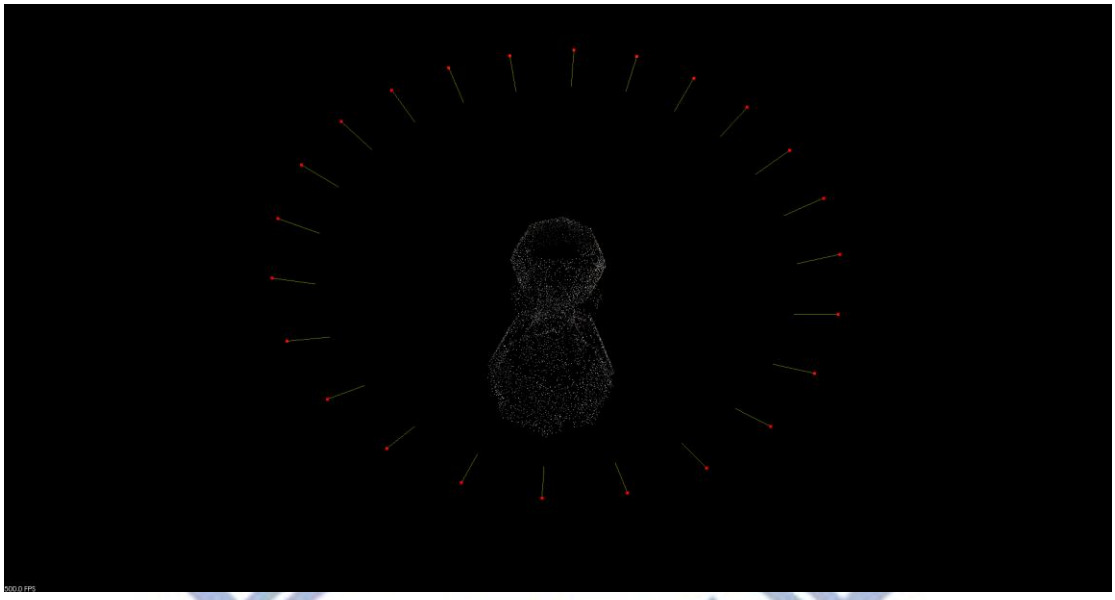


圖 12. Pawn dataset 中 6490 個 seed patch 在空間中之分布，其中紅點位置表相機中心，黃線為相機光軸方向

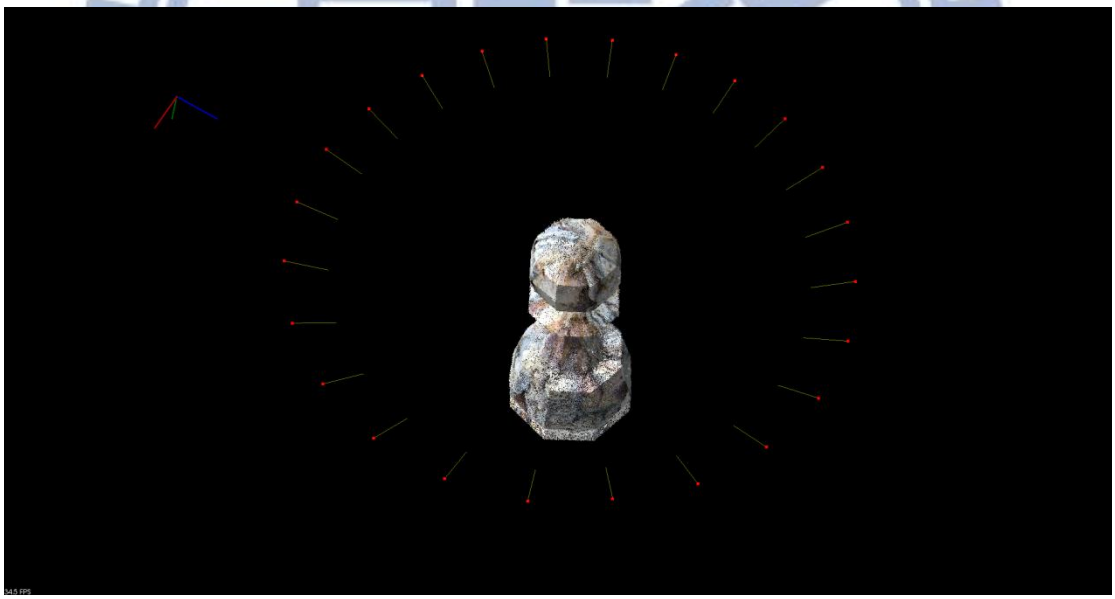


圖 13. Pawn dataset 重建後的空間模型與相機分布，其中紅點位置表相機中心，黃線為相機光軸方向

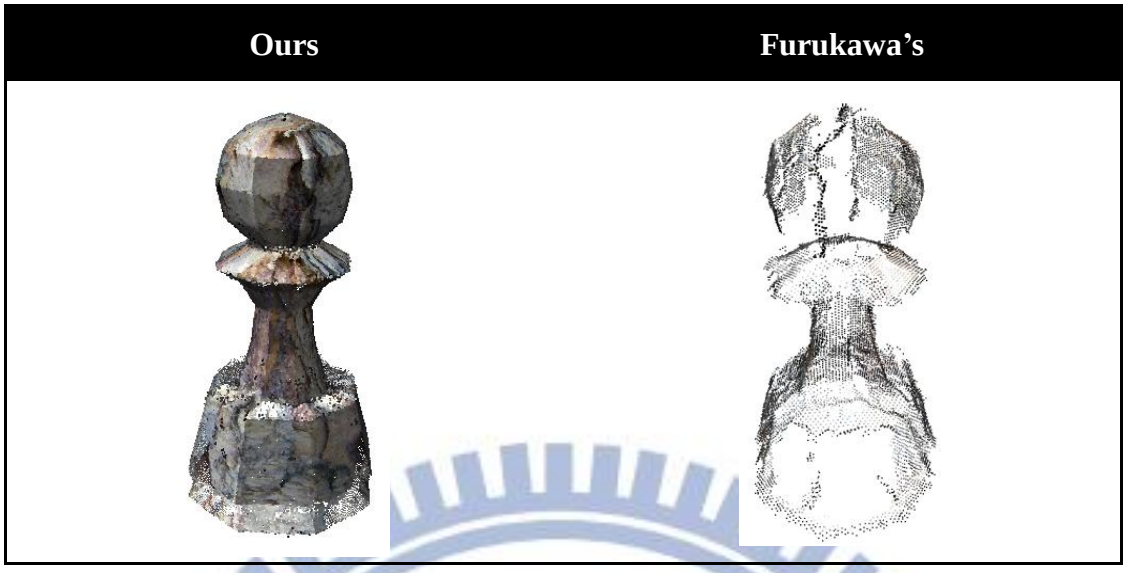


圖 14. Pawn dataset 的 patch 的重建結果比較

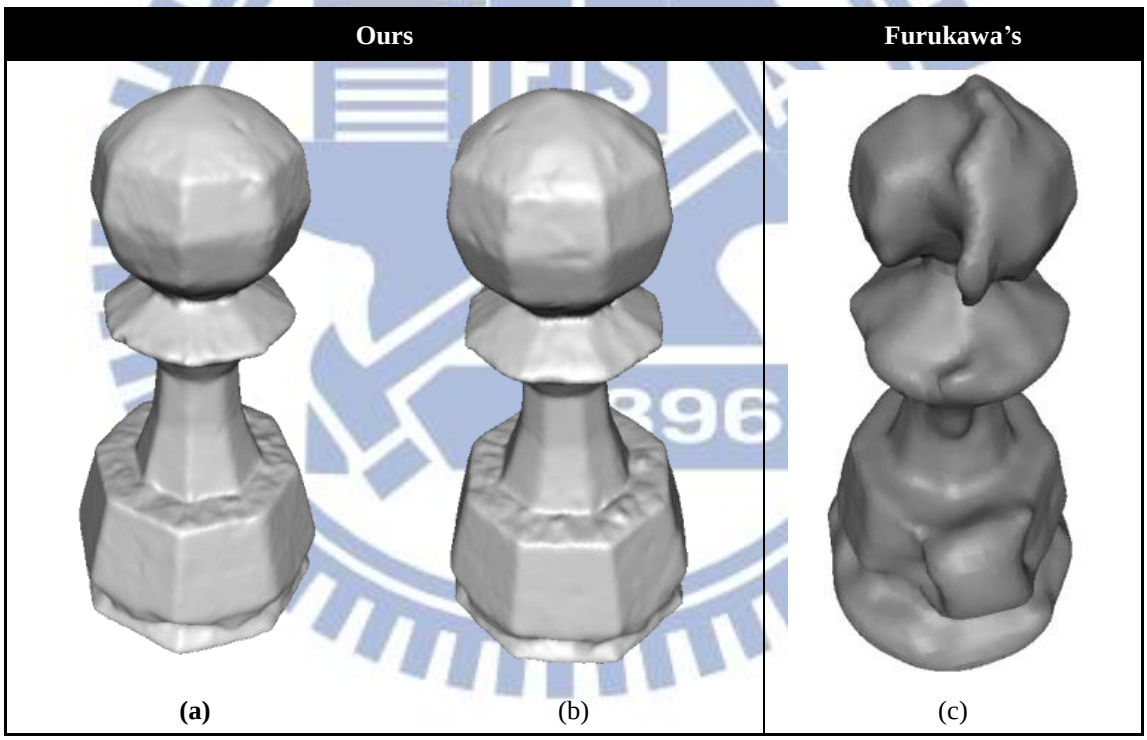


圖 15. Pawn dataset 重建結果比較，左二圖為我們方法的正面(a)與背面(b)重建結果，(c)為 Furukawa (PMVS)的重建結果

7.4 使用真實影像重建三維模型 (Reconstruction of Real Images)

在本節的實驗中，我們對實拍影像進行重建，7.4.1 與 7.4.2 使用 middlebury 的 dataset，並且套用已知的相機參數；7.4.3 中的 face dataset 我們使用由 bundle adjustment 的相機參數進行重建。

7.4.1 建立實拍 Middlebury 恐龍模型 (Reconstruction of Middlebury Dinosaur Model)

在這個實驗中，我們使用 Middlebury [2]的影像資料庫 Dino dataset，在一個固定光源的攝影棚拍攝，共有 363 張影像從半球型狀向球心的角度拍攝如圖 17，每張影像大小為 640x480，並且已知相機內部參數與外部參數。首先我們利用 SIFT 進行 n-view matching，並使用三角定位求出 seed point，再依本論文的演算法重建 dense patches。

圖 16. Dino dataset 其中三張影像，由左至右分別為正面、背面、頂面三張影像

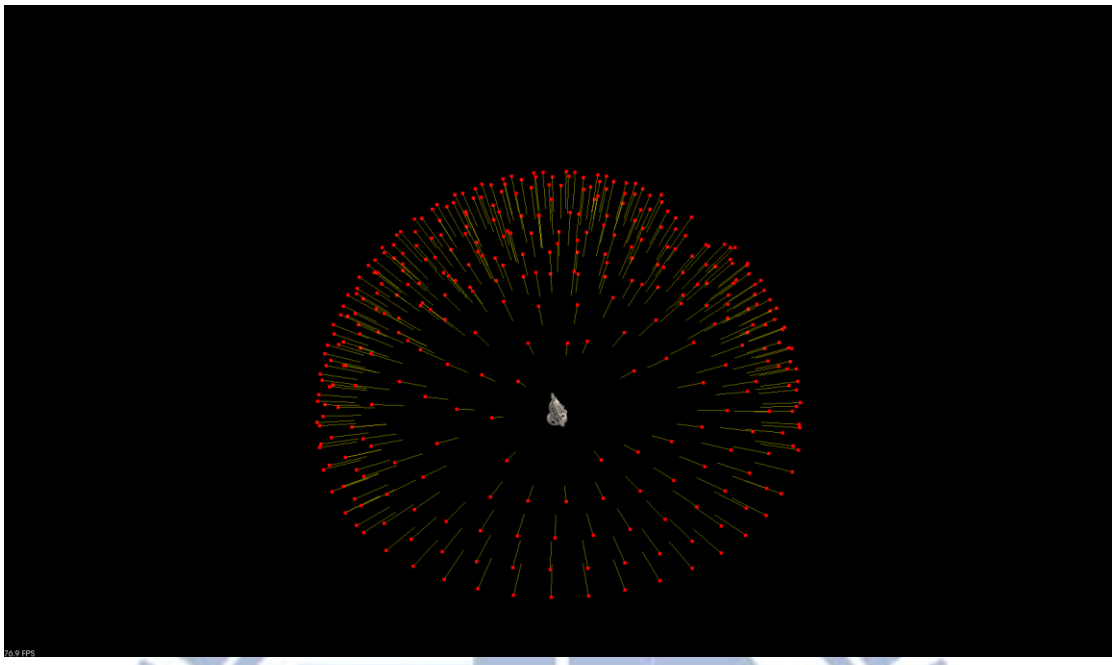


圖 17. Dino dataset 空間中的模型與相機分布，其中紅點位置表相機中心，黃線為相機光軸方向，球心部分維模型所區域。

由於這是一個相對較容易控制的拍攝環境，我們使用簡單的方法取得 silhouette，首先將每張影像的 intensity 分別 normalize 到 [0,1]，我們設定 threshold 僅取 intensity 大於 0.15 的 pixel，再依序進行 kernel size 10 的 dilate 與 erode 作為 silhouette 影像。

首先我們仍使用 SIFT 做 n-view feature matching，並確認作為 seed point 的 3D 點，在實驗中 Dino dataset 共有 6057 個 initial seed point，經過 runtime filtering 後的 seed patch 共 1872 個。

這個實驗由於 camera view 的數量相當豐富，為了增加 expansion 速度，故我們這裡不對每個 visible cell map 進行 expansion，而僅對每個 patch 的 reference cell map 進行 expansion。在這個實驗中我們嘗試了不同 cell size 的重建結果，當 cell size 為 4 的模型明顯比 cell size 2 的模型粗略許多，但獲得了明顯較少的重建時間僅 30 分鐘，而 cell size 2 的模型則需要 10 小時。

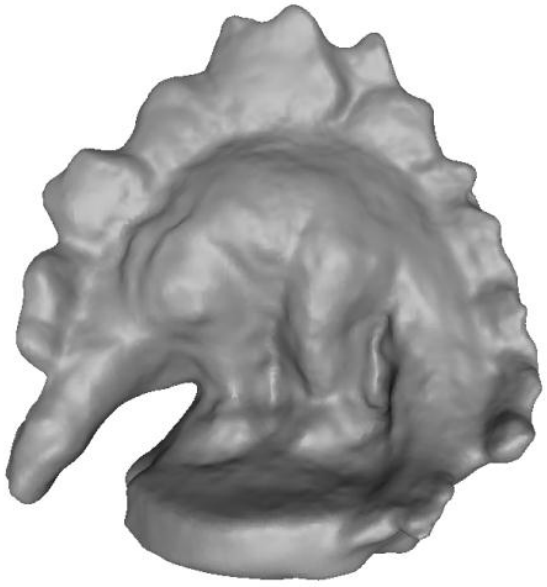


圖 18. Dino dataset cell size 4 之重建結果

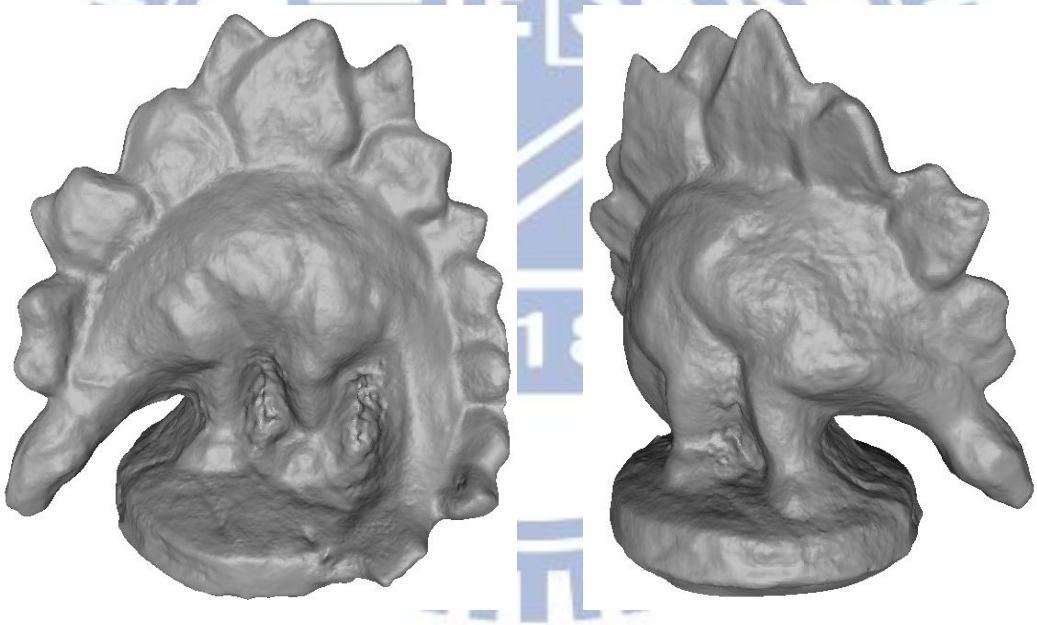


圖 19. Dino dataset cell size 2 之重建結果

7.4.2 建立實拍 Middlebury 神殿模型 (Reconstruction of Middlebury Temple Model)

由於 temple dataset 與 dino dataset 的環境相同，我們與 7.4.1 節採用相同的前處理方式與特徵對應，並且也使用相同的方法取得 silhouette 影像，再執行我們的演算法嘗試建立 dense patch 與模型。

圖 20. Temple dataset 其中三張影像，由左至右分別為正面、背面、頂面三張影像

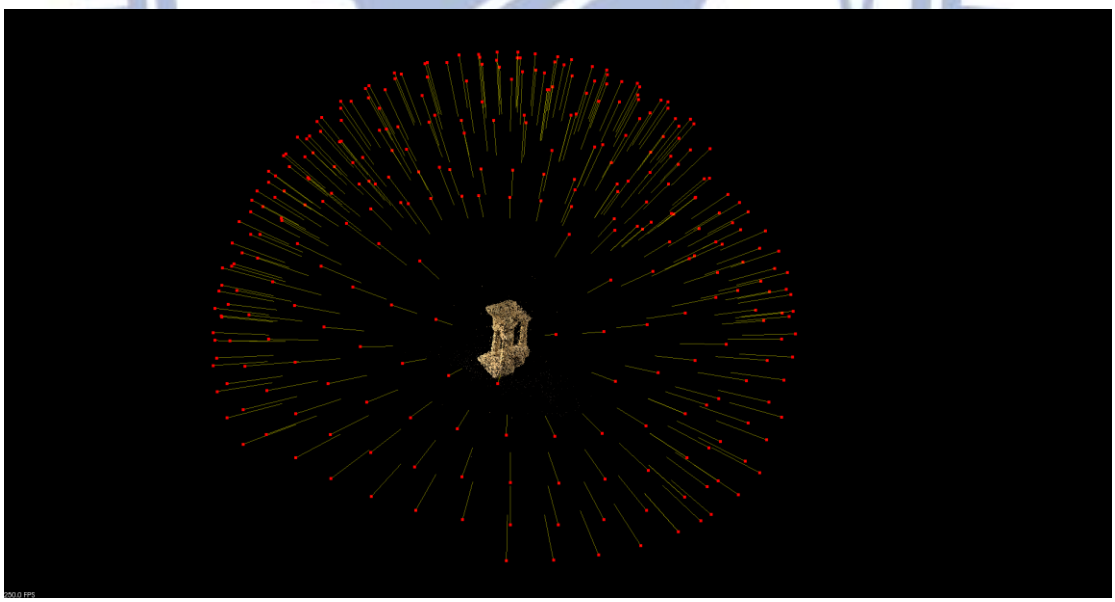


圖 21. Temple dataset 空間中的模型與相機分布，其中紅點位置表相機中心，黃線為相機光軸方向，球心部分維模型所區域。

圖 22. Temple dataset cell size 4 之重建結果

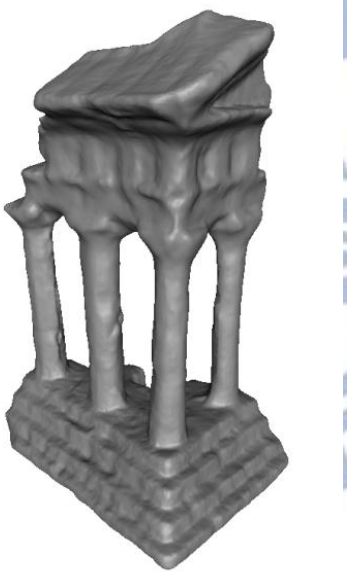


圖 23. Temple dataset cell size 2 之重建結果

7.4.3 建立人臉模型 (Reconstruction of Human Face Model)

我們從 [23]中的人臉 dataset 嘗試建立模型，拍攝環境同樣也是固定光源的攝影棚，影像共 7 張千萬畫素的不同相機拍攝 (圖 24)，這組 dataset 的主要困難點在於相機之間的角度相對較大，並且分布也不是 uniform，對於 expansion 階段時的 visible camera 選取較有挑戰性。與 Furukawa 的結果相比，由於側臉與正臉的相機分布夾角差異較大，故 Furukawa 的重建結果並沒有側臉資訊，而我們的方法因為參考了 parent patch 的 camera view，故較能選取到側臉部位的相機，重建完整度較高 (圖 25)。

第二的困難點是影像的大小約 4000x3000，並且在側臉及額頭部分有明顯的 textureless，而要在原圖影像用 31x31 的 window 含括有效的 texture 範圍機率較低，對於較為 textureless 的區域我們是使用 level of detail 的縮圖方式增加每個的 pixel 的描述力，而非增加 window size，對於 optimization 的加速有明顯的改善。設定上我們令 $\varepsilon = 0.8$，最大 LOD 為 10，也就是同樣在 31x31 的 window 內我們最多可以獲得約 288x288 的影像描述力。與使用固定 61x61 windows size 的計算相比，實際重建時間可由 11 個小時縮減為約 4~5 小時。

圖 24. Face dataset 中正臉影像(左圖)與相機分布(右圖)

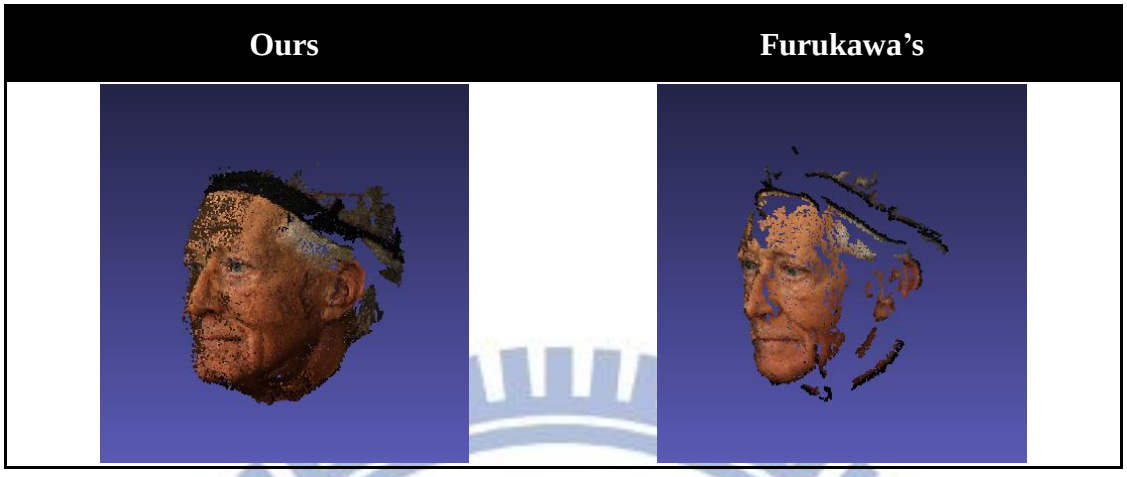


圖 25. Face dataset 與 Furukawa 的重建完整度比較

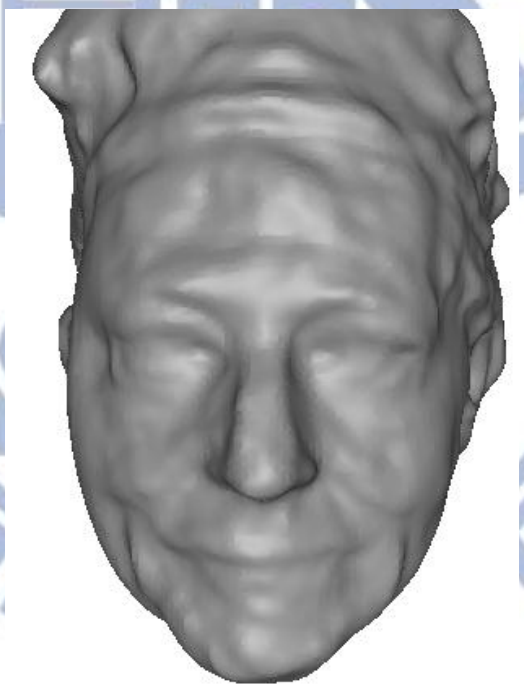


圖 26. Face dataset cell size 2 之重建結果

第八章 結論與未來發展 (Conclusion and Future Work)

8.1 結論 (Conclusion)

本論文利用 inlier feature matching 的 camera view 提供可靠的 visible camera，可以有效解決在 un-uniform 分布的 camera view 造成 visible camera 不足之問題，進而提升整體模型的完整性，在合成影像與實拍影像的重建實驗中，我們的方法對於 camera view 的擺設有較低的依賴性，並且與 Furukawa 的重建相比提供相對完整的模型描述。此外，對於 textureless 物體或 high resolution image 的 sampling，我們採用了 level of detail 的 pyramid sampling，可以利用記憶體換取更有效率的收斂速度及準確度。而在物體邊緣交界處或 textureless 物體，我們使用了 adaptive fitness weighting 來重分配要 optimize 的 pixel 範圍，使其符合我們希望的銳利交界平面及收斂結果。而我們在 expansion 時考慮了 patch 不同的 priority，使其能夠從相對穩定的平面區塊向外擴增。最後我們採用 GLN-PSO 來進行整個複雜的 optimization，並且在提供 initial particle 的幫助下達到穩定且快速的收斂結果。

8.2 未來發展與研究方向 (Future work)

本論文的方法需依賴一開始的 n-view image feature matching 產生 seed point，而 SIFT 雖然提供大量的 linier matching，但其中仍然有不少錯誤；必須要在 runtime filtering 就先行刪除，否則其錯誤的 seed point 可能會 expansion 到意外的地方，而大多數的錯誤對應可藉由簡單的 silhouette 影像進行快速的 visibility test 刪除之。並且 SIFT 對於 perspective 的 invariant 抗性仍有限，在些某些 wide-baseline 情況因無法對應而難以發揮 inlier feature matching 提供可靠的 visible camera 之特性。未來可考慮使用 Maximally Stable Extremal Regions (MSER) 的 perspective invariant 特性搭配其他 descriptor 來描述 (如 Zernike moment)，將更有效地發揮我們演算法的精神。

由於 PSO 提供了相對全面的參數搜尋 (15 particle x 40 iteration)，雖然較難掉入 local-trap，但相較於 Furukawa 使用的 conjugate gradient 僅迭代 3 次收斂一個 patch 仍有速度上的巨大落差，而現有 CPU 核心數難以發揮 PSO 可平行計算 fitness 的特性，若未來轉換到 GPGPU 平台如 CUDA 上就有機會獲得更快的 optimization 速度。

整個模型的產生最後是由 Poisson surface reconstruction 建立，而該方法本身難以用曲線方程式來描述交界銳角處，我們能做的就是盡量描述更 dense 的 patch 在交界處來綁定這個銳角的特性，故儘管數值上我們獲得了一個相對銳利的平面 patch，最後仍會受到 PSR 的 smooth 影響，造成結果沒有理想中銳利。未來應可繼續朝新的 surface reconstruction 方法發展，如利用 plane equation fitting 的方式交界出邊緣銳角處進行重建，以發揮原本 patch 的完整描述力。

參考文獻 References

[1] S. Seitz, B. Curless, J. Diebel, D. Scharstein and R. Szeliski, "A comparison and evaluation of multi-view stereo reconstruction algorithms," *Computer Vision and Pattern Recognition*, vol. 1, pp. 519-528, 2006.

[2] S. Seitz, B. Curless, J. Diebel, D. Scharstein and R. Szeliski, "Multi-view stereo evaluation web page," [Online]. Available: http://vision.middlebury.edu/mview/.

[3] M. Beljan, R. Klowsky and M. Goesele, "A multi-view stereo implementation on massively parallel hardware," *Technical Report*, 2011.

[4] J. Y. Chang, H. Park, I. K. Park, K. M. Lee and S. U. Lee, "GPU-friendly multi-view stereo reconstruction using surfel representation and graph cuts," *Computer Vision and Image Understanding*, vol. 115, pp. 620-634, May. 2011.

[5] G. Vogiatzis and C. Hernández, "Video-based, real-time multi-view stereo," *Image and Vision Computing*, vol. 29, pp. 434-441, June. 2011.

[6] M. Habbecke and L. Kobbelt, "Iterative Multi-View Plane Fitting," *VISION, MODELING, AND VISUALIZATION*, pp. 73-80, Nov. 2006.

[7] M. Habbecke and L. Kobbelt, "A Surface-Growing Approach to Multi-View Stereo Reconstruction," *Computer Vision and Pattern Recognition*, pp. 1-8, 2007.

[8] Y. Furukawa and J. Ponce, "Accurate, Dense, and Robust Multi-View Stereopsis," *Pattern Analysis and Machine Intelligence*, vol. 32, pp. 1362-1376, Aug. 2010.

[9] Y. Furukawa, B. Curless, S. Seitz and R. Szeliski, "Manhattan-world Stereo," *Computer Vision and Pattern Recognition*, pp. 1422-1429, June. 2009.

[10] G. Vogiatzis, C. H. Esteban, P. H. S. Torr and R. Cipolla, "Multi-view stereo via

volumetric graph-cuts and occlusion robust photo-consistency," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 12, pp. 2241-2246, Dec. 2007.

[11] M. Goesele, B. Curless and S. Seitz., "Multi-view stereo revisited," *Computer Vision and Pattern Recognition*, vol. 2, pp. 2402-2409, 2006.

[12] A. Hornung and L. Kobbelt, "Hierarchical Volumetric Multi-view Stereo Reconstruction of Manifold Surfaces based on Dual Graph Embedding," *Computer Vision and Pattern Recognition*, vol. 1, pp. 503-510, June. 2006.

[13] G. Vogiatzis, P. Torr and R. Cipolla., "Multi-view stereo via volumetric graph-cuts," *Computer Vision and Pattern Recognition*, vol. 2, pp. 391-398, June. 2005.

[14] H. Jin, S. Soatto and A. J. Yezzi, "Multi-View Stereo Reconstruction of Dense Shape and Complex Appearance," *International Journal of Computer Vision*, vol. 63, pp. 175-189, July. 2005.

[15] O. Faugeras and R. Keriven, "Variational principles, surface evolution, PDEs, level set methods, and the stereo problem," *IEEE Transactions on Image Processing*, vol. 7, pp. 336-344, Mar. 1998.

[16] S. Soatto, A. J. Yezzi and H. Jin, "Tales of Shape and Radiance in Multi-view Stereo," *International Conference on Computer Vision*, vol. 2, pp. 974-981, 2003.

[17] H. Jin, S. Soatto and A. Yezzi, "Multi-view stereo beyond lambert," *Computer Vision and Pattern Recognition*, vol. 1, pp. I-171- I-178, June. 2003.

[18] Y. Duan, L. Yang, H. Qin and D. Samaras, "Shape Reconstruction from 3D and 2D Data Using PDE-Based Deformable Surfaces," *European Conference on Computer Vision*, pp. 238-251, 2004.

[19] C. Hernandez Esteban and F. Schmitt, "Multi-stereo 3D object reconstruction," *Proceedings of 3D Data Processing Visualization and Transmission,* pp. 159-166, 2002.

[20] C. Wu, "VisualSFM : A Visual Structure from Motion System," 2012. [Online]. Available: http://www.cs.washington.edu/homes/ccwu/vsfm/.

[21] J. Kennedy and R. Eberhart, "Particle Swarm Optimization," *IEEE International Conference on Neural Networks.*, vol. 4, pp. 1942-1948, 1995.

[22] P. Pongchairerks and V. Kachitvichyanukul, "A NON-HOMOGENOUS PARTICLE SWARM OPTIMIZATION WITH MULTIPLE SOCIAL STRUCTURES," *International Conference on Simulation and Modeling*, 2005.

[23] T. Beeler, B. Bickel, P. Beardsley, B. Sumner and M. Gross, "High-Quality Single-Shot Capture of Facial Geometry," *ACM Transactions on. Graphics (Proc. SIGGRAPH)*, vol. 29, no. 3, pp. 40:1--40:9, 2010.

[24] M. Goesele, N. Snavely, B. Curless, H. Hoppe and S. Seitz, "Multi-view stereo for community photo collections," *International Conference on Computer Vision*, pp. 1-8, Oct. 2007.

附錄 1. 參數符號表 (Symbol Table)

Symbol	Descriptions	default value
$\boldsymbol{V}$	A set of cameras used to shoot the object	
$\boldsymbol{I_i}$	Image associated with camera # i in V	
$\boldsymbol{O_i}$	The *i*-th camera center	
$\boldsymbol{P_i}$	Projection matrix of camera # i	
$\boldsymbol{P_i(x)}$	Projection of 3D point *x* onto image plane I_i	
$\boldsymbol{p}$	Patch identity symbol	
$\boldsymbol{c(p)}$	Patch center of p	
$\boldsymbol{n(p)}$	Patch normal of p	
$\boldsymbol{V(p)}$	The set of cameras seeing patch p $V(p) \in V$	
$\boldsymbol{R(p)}$	Patch reference camera $R(p) \in V(p)$	
$\boldsymbol{r(p)}$	Unit ray vector from $O_{R(p)}$ to $c(p)$ $\|r(p)\| = 1$	
$\boldsymbol{d(p)}$	Depth of patch p $c(p) = d(p)r(p)$	
$\boldsymbol{D(p)}$	Distance from patch plane to world origin	
$\boldsymbol{q(p)}$	Patch priority value of patch p	
$\boldsymbol{l(p)}$	The level of detail for patch p $l(p) = 0$ stands for the original patch level	
$\boldsymbol{L(p)}$	The shrinkage matrix for the image at l–th level of detail	
$\boldsymbol{\Omega_l(p)}$	Patch extent at the l–th level of detail (given by a	

	set of image pixels)	
$\boldsymbol{\gamma(p)}$	Normalized texture correlation of patch p	
$\boldsymbol{V_\gamma(p)}$	Reference correlation camera (See section 3.6)	
$\boldsymbol{C_i}$	Cell map generated from I_i	
$\boldsymbol{C_i(p)}$	The cell of patch p in cell map $\boldsymbol{C_i}$	
$\boldsymbol{N_i(p)}$	A group of unexpanded neighbor cells of $C_i(p)$	
$\boldsymbol{\tau}$	Cell size (in 2D image pixel)	4
$\boldsymbol{C_{max}}$	Maximum patch number in a cell	5
$\boldsymbol{V_{min}}$	Minimum visible camera number	3
$\boldsymbol{u}$	Patch extent radius in $I_{R(p)}$ (in 2D image pixel)	15
$\boldsymbol{\varepsilon}$	Level of detail scalar ratio	0.8
$\boldsymbol{v}$	Intensity variance threshold	36
$\boldsymbol{\alpha}$	Normalized texture correlation threshold	0.95
$\boldsymbol{\omega}$	Visible camera normal correlation threshold	0.87
$\boldsymbol{\lambda}$	Homography window ratio threshold	0.55
$\boldsymbol{\Lambda}$	Neighbor support ratio	0.25
$\boldsymbol{\beta}$	Depth range scalar (pixel)	0.25τ
$\boldsymbol{\sigma_g}$	Patch distance weighting	$\frac{(u*2+1)}{3}$
$\boldsymbol{\sigma_h}$	Patch difference weighting	128*128
$\boldsymbol{\sigma_k}$	Patch gradient weighting	10.0

表 4. 參數符號表

附錄 2. 虛擬程式碼 (Pseudocode)

Input: **multiple images**

Output: **a set of dense patches**

I. **Compute SIFT features and do n-view matching for all camera images.**

II. **For each feature matching**

1. **Compute $c(p)$ from triangulation.**
2. **Initialize $V(p), R(p), n(p), d(p), r(p)$**
3. **Optimize patch p using GLN-PSO algorithm.**
4. **Compute $q(p)$ and push p into priority queue.**

III. **While priority queue is not empty**

1. **Pop the top priority patch p form priority queue**
2. **Initialize $C_i(p), N_i(p)$ $where$ $i \in V(p)$**
3. **For each p' of $N_i(p)$**

   i. **Initialize $V(p'), R(p'), n(p'), c(p'), d(p'), r(p')$**

   ii. **Optimize p' using GLN-PSO algorithm.**

   iii. **Filter out invisible views $V(p') \leftarrow \{i \in V(p'), t_i \cdot t_{V_\gamma(p')} > \alpha\}$**

   iv. **If $|V(p')| < V_{min}$, drop p'**

   v. **If all $|C_i(p')| \geq C_{max}, i \in V(p')$, drop p'**

   vi. **Compute $q(p')$**

   vii. **Push p' into visible cell maps $C_i(p'), i \in V(p')$ and priority queue.**

IV. **3-Step Patch filtering**

表 5. 演算法虛擬程式碼