

**ROBUST RECONSTRUCTION OF TEXTURED  
SURFACES FROM 3D POINT CLOUDS**

ZHU Chen

Department of Computer Science

School of Computing

National University of Singapore

January 2013

**ROBUST RECONSTRUCTION OF TEXTURED  
SURFACES FROM 3D POINT CLOUDS**

ZHU Chen

( B.Comp (Honours), NUS )

THESIS SUBMITTED

FOR THE DEGREE OF MASTER OF SCIENCE

Department of Computer Science

School of Computing

National University of Singapore

January 2013

Supervisor: Associate Professor Leow Wee Kheng

In Proceedings of 30th Computer Graphics International (2013).

## **Declaration**

I hereby declare that this thesis is my original work and it has been written by me in its entirety.

I have duly acknowledged all the sources of information which have been used in the thesis.

This thesis has also not been submitted for any degree in any university previously.

Zhu Chen

23 January 2013

## **Acknowledgments**

I want to express my sincere gratitude to my supervisor Associate Professor Leow Wee Kheng for guiding me to embark on 3D reconstruction research on August 2011. Ever since then, I have learned not only academic knowledge from his course, but also ways of logical thinking to tackle complex problems by using simplified but innovative approach. Each time after discussion, his advice and reasoning have helped me to stay clear about my research objective and to keep on progressing throughout my research study.

Also, I want to thank Associate Professor Michael S. Brown, who has given me valuable recommendations on literature and insights on texturing mapping.

Last but not least, I want to thank my friend Cheng Yuan in Computer Vision Lab for providing me help and logistics.

## Table of Contents

|   |    |
|---|----|
| <b>Abstract</b> .....                       | 1  |
| <b>Acknowledgment</b> .....                 | 2  |
| <b>1. Introduction</b> .....                | 6  |
| 1.1 Motivation.....                         | 6  |
| 1.2 Thesis Objective.....                   | 7  |
| 1.2 Thesis Organization.....                | 7  |
| <b>2. Literature Review</b> .....           | 8  |
| <b>3. Algorithms</b> .....                  | 11 |
| 3.1 Overview of Algorithm.....              | 11 |
| 3.2 3D Point Cloud Recovered by PMVS.....   | 12 |
| 3.3 Robust Surface Fitting Algorithm.....   | 14 |
| 3.3.1 Plane Fitting.....                    | 14 |
| 3.3.2 Curve Surface Fitting.....            | 16 |
| 3.3.3 Robust Algorithm.....                 | 17 |
| 3.4 Robust Surface Splitting Algorithm..... | 21 |
| 3.4.1 Robust Surface Splitting.....         | 21 |

|           |  |           |
|-----------|--|-----------|
| 3.4.2     | Alignment of Surfaces .....                    | 24        |
| 3.5       | Surface Resampling .....                       | 27        |
| 3.5.1     | Resampling of Surface Points .....             | 27        |
| 3.5.2     | Color Retrieval from Frontal Image.....        | 30        |
| 3.5.3     | Removal of Distortion and Color Blending ..... | 31        |
| 3.6       | Construction of 3D Mesh Model .....            | 34        |
| <b>4.</b> | <b>Test Results &amp; Discussions</b> .....    | <b>36</b> |
| 4.1       | Test Data .....                                | 36        |
| 4.2       | Reconstructed 3D Models .....                  | 38        |
| <b>5.</b> | <b>Conclusion</b> .....                        | <b>45</b> |
| 5.1       | Summary .....                                  | 45        |
| 5.2       | Future work .....                              | 45        |
|           | <b>Bibliography</b> .....                      | <b>46</b> |

## Abstract

3D building models have a wide range of applications. A reconstructed 3D building model can be used for virtual reality, games, films and city planning. There are three main approaches for 3D building reconstruction from images: Laser Scanning, Structure-from-Motion (SfM) and Multi-View Stereopsis (MVS). Laser Scanning is accurate, but expensive and limited by the range of the laser. Both SfM and MVS recover 3D buildings from multiple images. SfM recovers 3D data as well as camera positions and orientations from a sequence of input images. MVS methods, especially patch-based MVS (PMVS), apply various geometric constraints to determine point correspondence, and thus can achieve higher accuracy and often recover denser 3D point cloud than do SfM methods. Complex algorithms need to be applied to the point cloud to construct mesh surfaces. But, the recovered point cloud can have missing data points in regions that lack features for matching, making recovery of complete surface difficult.

This thesis presents a robust reconstruction of textured surfaces from 3D point clouds given by patch-based MVS method. To a good first approximation, a building's surface can be modeled by either a flat plane or a curve surface, when the scene depth is much smaller than the distance of the building to the camera. Therefore, simple surfaces can be fitted into the point cloud. Then, 3D points are resampled on the fitted surfaces in an orderly pattern, whose colors are retrieved from the input images. This approach is thus simple, inexpensive and effective for

recovering textured surfaces of buildings. Test results show that such 3D mesh models are sufficiently accurate and realistic for 3D visualization in many applications.

**Keywords:**

3D Building Reconstruction, Multi-View Stereo, Surface Fitting, Mesh Surface Reconstruction

## List of Figures

|   |    |
|---|----|
| Fig. 1. Projection of 3D patches in multiple views.....                 | 4  |
| Fig. 2. Point cloud recovered by PMVS .....                             | 5  |
| Fig. 3. Point cloud of Old Stamford House recovered by PMVS.....        | 7  |
| Fig. 4. Fitted plane of Old Stamford House.....                         | 9  |
| Fig. 5. Fitted curve surface of Old Stamford House.....                 | 11 |
| Fig. 6. Convergence curve of robust surface fitting .....               | 13 |
| Fig. 7. 3D points distribution near the curve surface.....              | 14 |
| Fig. 8. Point cloud in the world coordinate .....                       | 15 |
| Fig. 9. Splitting of point cloud to reconstruct multiple surfaces ..... | 18 |
| Fig. 10. Alignment of surfaces .....                                    | 21 |
| Fig. 11. Resampled surface points .....                                 | 24 |
| Fig. 12. Resampled 3D points with colors.....                           | 26 |
| Fig. 13. Removal of distorted color texture .....                       | 28 |
| Fig. 14. Selected region for color blending .....                       | 29 |
| Fig. 15. Color blending.....  | 30 |
| Fig. 16. 3D Mesh model of Old Stamford House.....                       | 32 |

|  |    |
|--|----|
| Fig. 17. Input images of SMRT Headquarters from different views.....   | 33 |
| Fig. 18. Input images of Old Stamford House from different views ..... | 34 |
| Fig. 19. 3D point cloud of SMRT Headquarters recovered by PMVS .....   | 35 |
| Fig. 20. 3D point cloud of Old Stamford House recovered by PMVS .....  | 36 |
| Fig. 21. Refined surface of SMRT Headquarters .....                    | 37 |
| Fig. 22. Resampled 3D points on surface of SMRT Headquarters .....     | 38 |
| Fig. 23. Resampled 3D points on surface of Old Stamford House.....     | 39 |
| Fig. 24. Reconstructed mesh model of SMRT Headquarters .....           | 40 |
| Fig. 25. Reconstructed mesh model of Old Stamford House .....          | 41 |

# 1. Introduction

## 1.1 Motivation

3D building models have a wide range of applications. A reconstructed 3D building model can be used for virtual reality, games, films and city planning. For heritage preservation, reconstructed 3D model of a historical building can be stored in a digital library.

There are three main approaches for 3D building reconstruction: namely laser scanning, Structure-from-Motion (SfM) and Multi-View Stereopsis (MVS). Laser scanning [1,4,8,9,14,36] is very accurate and efficient, but requires an expensive laser scanner, and is limited by the range of the laser. Both SfM and MVS recover 3D buildings from multiple images. SfM recovers 3D data as well as camera positions and orientations from a sequence of input images [2, 6, 7, 17, 18, 28, 29, 33, 34, 37]. The required camera parameters can be calibrated explicitly or computed by self-calibration methods. MVS methods apply various geometric constraints to determine point correspondence for triangulation of 3D points given known camera parameters [13, 20, 26]. MVS methods, especially patch-based MVS (PMVS) [12, 13, 15, 20, 23, 24], can achieve higher accuracy [30] and often recover denser 3D point cloud than do SfM methods.

This thesis presents a robust reconstruction of textured surfaces from 3D point clouds given by patch-based MVS method. Building's surfaces are modeled by either flat planes or curve surfaces. This is a good first approximation when the scene depth is

much smaller than the distance of the building to the camera, which holds true for many contemporary architectural design. Therefore, simple surfaces can be fitted into the point cloud, and then 3D points are resampled on the fitted surfaces in an orderly pattern, whose colors are retrieved from the input images. This approach is thus simple, inexpensive and effective for recovering parametric textured surfaces of buildings. Test results show that such 3D mesh models are sufficiently accurate and realistic for 3D visualization in many applications.

## **1.2 Thesis Objective**

The main objective of this thesis is to develop a pipeline for 3D reconstruction from multiple views for buildings with curved surfaces, with the following criteria

- 1) The reconstructed surfaces should be complete with colour texture without holes or missing texture.
- 2) Multiple surfaces of a building should be accurately reconstructed.

## **1.3 Thesis Organization**

Next, literature review is discussed in chapter 2. Then, the three main stages of our 3D reconstruction pipeline are elaborated in chapter 3. Next, reconstruction results including point cloud, meshes and 3D model with reconstructed surface will be displayed in chapter 4. Finally, I will conclude this paper in chapter 5.

## 2. Literature Review

Various methods have been used for the acquisition of 3D shapes of objects in the past. There are two main categories, namely active methods and passive methods. Active methods use specially controlled light sources to illuminate the scene and acquire 3D data from the illuminated patterns. These methods include time-of-flight, shape-from-shading, structured light, active stereo, and photometric stereo [27]. They are computationally less demanding than do passive methods, since special controlled light is used to simplify the 3D acquisition process. Laser scanning and structure light are already used in commercial products. But they are limited to scenes, where special controlled illumination can be properly applied.

Passive methods acquire 3D data only from input images, without using controlled light. Based on the numbers of view points, passive methods can be further categorized into two groups. Methods such as shape-from-texture, shape-from-occlusion, shape-from-defocus, and shape-from-contour acquire 3D data from single view point [27]. Due to the nature of the features used, these methods are restricted to scenes that are rich in the required features, and they tend to be less accurate than active methods. Methods such as passive stereo, multi-view stereo, structure-from-motion, shape-from-silhouettes acquire 3D data using images captured from two or more view points [27].

Among the above methods, laser scanning, structure-from-motion, and multi-view -stereo have been successfully demonstrated for 3D reconstruction of buildings and street scenes. Laser scanning is very accurate and efficient, but requires an expensive

laser scanner, and is limited by the range of the laser. In reconstructing large historic sites, laser scanning is often used together with other data such as engineering drawing, aerial images, ground-plane images, etc. [1,4,8,9,14,36]

Structure-from-motion (SfM) methods recover 3D data as well as camera positions and orientations from a sequence of input images [2, 6, 7, 17, 18, 28, 29, 33, 34, 37]. The required camera parameters can be calibrated explicitly or computed by self-calibration methods [16]. MVS methods apply various geometric constraints to determine point correspondence for triangulation of 3D points given known camera parameters [13, 20, 26]. MVS methods, especially patch-based MVS (PMVS) [12, 13, 15, 20, 23, 24], can achieve higher accuracy [30] and often recover denser 3D point cloud than do SfM methods. PMVS can produce dense oriented rectangular patches with center vectors and normal vectors. Each patch can be projected into multiple images viewing it (See Fig. 1).

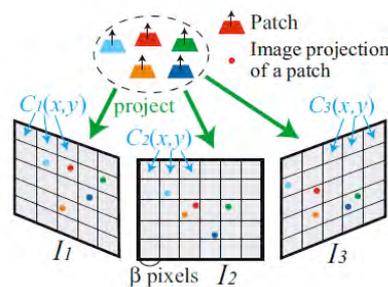


Fig. 1. Projection of 3D patches in multiple views.

An initial point cloud can be established through feature matching with Harris and Difference of Gaussians operators. Then, after procedures of patch initialization in neighboring image cell, patch optimization with respect to center vector and normal

vector, patch expansion and filtering, the point cloud gradually becomes dense. Since PMVS uses a weak form of regularization during patch expansion and filtering, it can recover a dense 3D point cloud, but also contain many outliers. Moreover, point cloud recovered by PMVS has large empty regions, which are not successfully reconstructed (see Fig. 2). Hence, PMVS is suitable to recover an initial point cloud from multiple views. Then, further techniques are needed to refine the point cloud and recover the missing region on the surface of building.



Fig. 2. Point cloud recovered by PMVS. Empty regions are not successfully reconstructed.

SfM and MVS may be used together in a pipeline with MVS serving as the refinement stage [6, 25]. Sophisticated algorithms are required to reconstruct mesh surfaces from point cloud. For example, [10,13] use Poisson surface reconstruction method [21], and [20,22] employ graph cut algorithm with geometric constraints.

## 3. Algorithms

### 3.1 Overview of Algorithm

This proposed algorithm continues from where PMVS ends. The pipeline for 3D reconstruction consists of three main stages as below:

- Stage 1: Recover a 3D point cloud of a building using PMVS.
- Stage 2: Reconstruct main surfaces from 3D point cloud by Robust Surface Fitting (chapter 3.3) and Splitting (chapter 3.4).
- Stage 3: Resample 3D points to refine surfaces (chapter 3.5).

First, the Bundler algorithm [31] is applied to the input images to extract matching feature points and camera parameters. Then, we apply PMVS to recover a 3D point cloud in stage 1. The reason to choose PMVS is that it can output a dense point cloud, which describes the overall structure of the building. In stage 2, I will perform robust surface fitting algorithm to obtain the parametric equation of the underlying surface. A robust surface splitting algorithm is also introduced to split the input 3D point cloud into multiple partitions for fitting different surfaces. In stage 3, 3D points are resampled on the fitted surfaces, and their colors are retrieved from frontal images.

For a large building, a single point cloud cannot cover the full extent of the building. In this case, multiple point clouds can be recovered for different parts of the building, and our algorithm can be applied to reconstruct different parts of the surfaces, which can then be aligned and merged together.

## 3.2 3D Point Cloud Recovered by PMVS

First, the Bundler algorithm [31] is applied to the input images to extract matching feature points and camera parameters. Next, PMVS algorithm [11, 12, 29, 32] makes use of the matched feature points and projection matrix of each view to recover an initial 3D point cloud with color information. Then after several iterations of patch optimization and patch expansion, the point cloud gradually becomes dense.

Even though PMVS can generate dense point cloud, there still remain large empty regions, which are not successfully reconstructed. Also, points do not lie on the same plane due to inaccuracy on their 3D coordinates. (See Fig. 3)

Hence, in order to overcome these defects, a robust surface fitting algorithm will be introduced in the next chapter to compute an accurate surface parametric equation of the building.

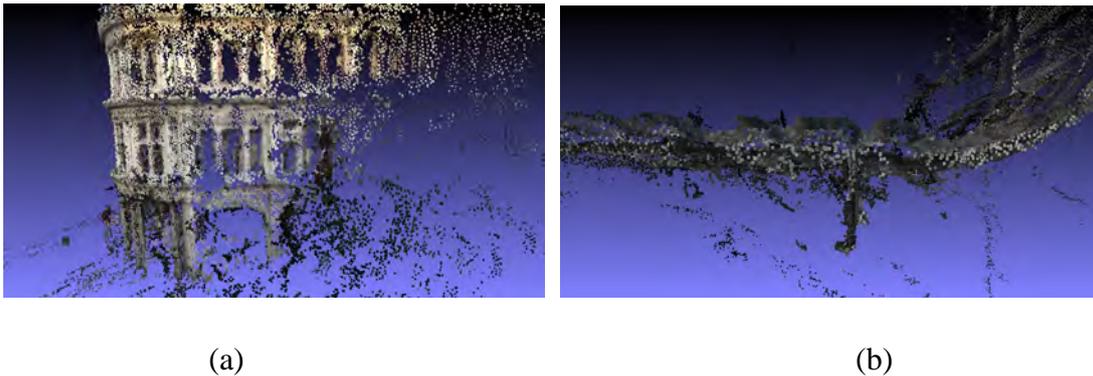


Fig. 3. Point cloud of Old Stamford House recovered by PMVS.

(a) Right plane of the building. Many regions of the plane are empty.

(b) Top-down view. Points don't lie on the plane

### 3.3 Robust Surface Fitting Algorithm

#### 3.3.1 Plane Fitting

To a first approximation, buildings consist of planes. So, it is natural to fit a plane into the point cloud and then reconstruct the whole plane of a building. But, as we zoom into the point cloud, we find that 3D points generally do not reside in the same plane. Although these points can provide some depth information about the structure, this depth information is usually quite small, and even not reliable at the very first place. Hence, there is a trade-off between the overall structure and the small depth information of a building. In our work, the geometrical accuracy of the overall structure determines the entire reconstruction process and is thus much more important than the small depth information which is negligible. Therefore, a plane will be used to cover one facet of a building. Initially, we will use an estimated line to split the point cloud given by PMVS, and use those 3D points near the target plane for plane fitting.

A plane in 3D space can be defined as

$$z = S(x, y) = b_1 x + b_2 y + b_3 \quad (1)$$

Then 3D points  $(x_i, y_i, z_i)$  around the facet of building can be fitted into this plane, and be written into a system of  $n$  linear equations as below

$$\begin{bmatrix} z_1 \\ \vdots \\ z_n \end{bmatrix} = \begin{bmatrix} x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots \\ x_n & y_n & 1 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}, \quad (2)$$

which can also be written in a matrix form:

$$\mathbf{Z} = \mathbf{M} \mathbf{B} \quad (3)$$

The plane surface parameters  $\mathbf{B}$  can be solved using Least Square method.

Next, we need to refine this plane by discarding points with large distance to the plane (see Fig. 4). And then, we use this newly filtered point set to solve equation system (2) again, and get more accurate plane parameters  $\mathbf{B}$  until the sum of square errors of projection distance to the plane is smaller than a threshold. I will illustrate more about this iterative refinement process in chapter 3.3.3 Robust algorithms.

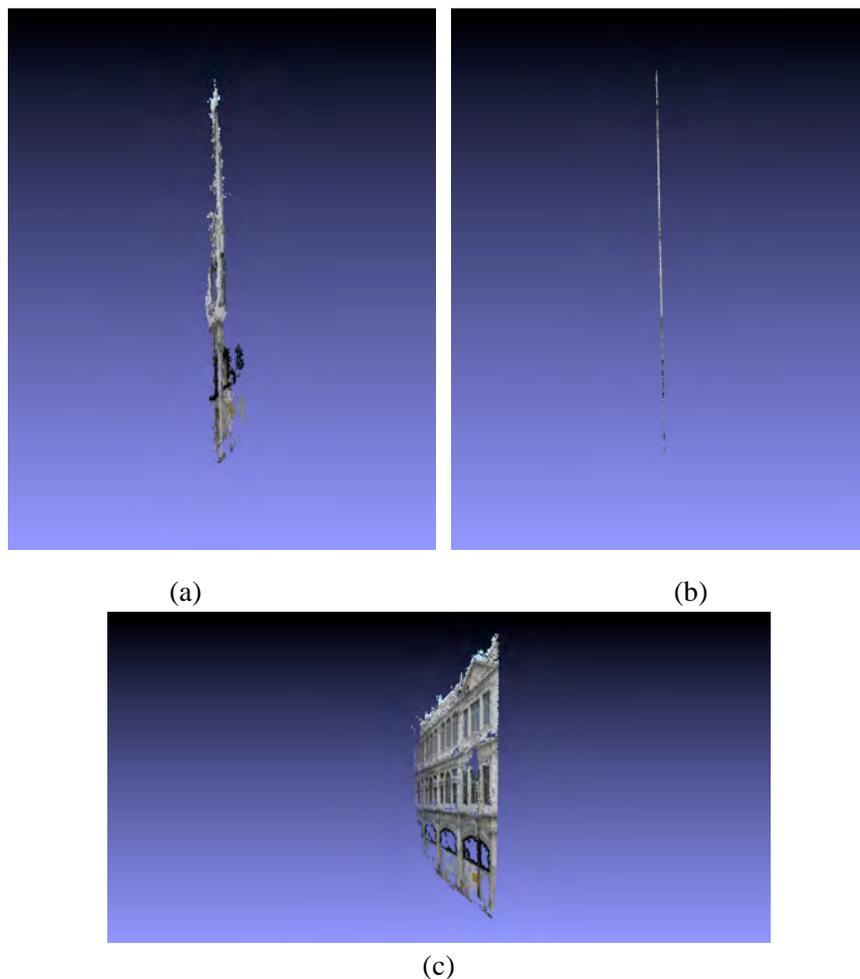


Fig. 4. Fitted plane of Old Stamford House. (a) point cloud with outliers before plane fitting (b) fitted points lie on a single plane (c) right view of the plane

### 3.3.2 Curve Surface Fitting

Curved surface can have various shapes and it is thus more complex to reconstruct than a plane. Since most buildings with curved surface exist in cylinder or parabola form, we can use second order polynomial functions to determine its shape. If the curved surface is more complex and varied, we can define it with a spline function and split it into different parts, of which each can be estimated using second order polynomial function. Or, we can use higher order polynomial function for more complex shapes in the same manner. Our current implementation restricts to planar and quadratic surfaces which are sufficient for the test cases. The type of surface is specified by the user.

A second order polynomial function can be defined as:

$$z = S(x, y) = c_{20}x^2 + c_{02}y^2 + c_{11}xy + c_{10}x + c_{01}y + c_{00} \quad (4)$$

All the points  $(x_i, y_i, z_i)$  belong to the curved surface region can be fitted into function (4), and be organized into a system of equations as below

$$\begin{bmatrix} z_1 \\ \vdots \\ z_n \end{bmatrix} = \begin{bmatrix} x_1^2 & y_1^2 & x_1y_1 & x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n^2 & y_n^2 & x_ny_n & x_n & y_n & 1 \end{bmatrix} \begin{bmatrix} c_{20} \\ \vdots \\ c_{00} \end{bmatrix}, \quad (5)$$

which can also be written into a matrix form as

$$\mathbf{Z} = \mathbf{N} \mathbf{C} \quad (6)$$

$(x_i, y_i, z_i)$  is known from the point cloud, and the curve surface parameters  $\mathbf{C}$  can be solved using least square method. With  $\mathbf{C}$  determined, we will have a explicit parametric equation (Eq. (4) ) of the curve surface (See Fig. 5).

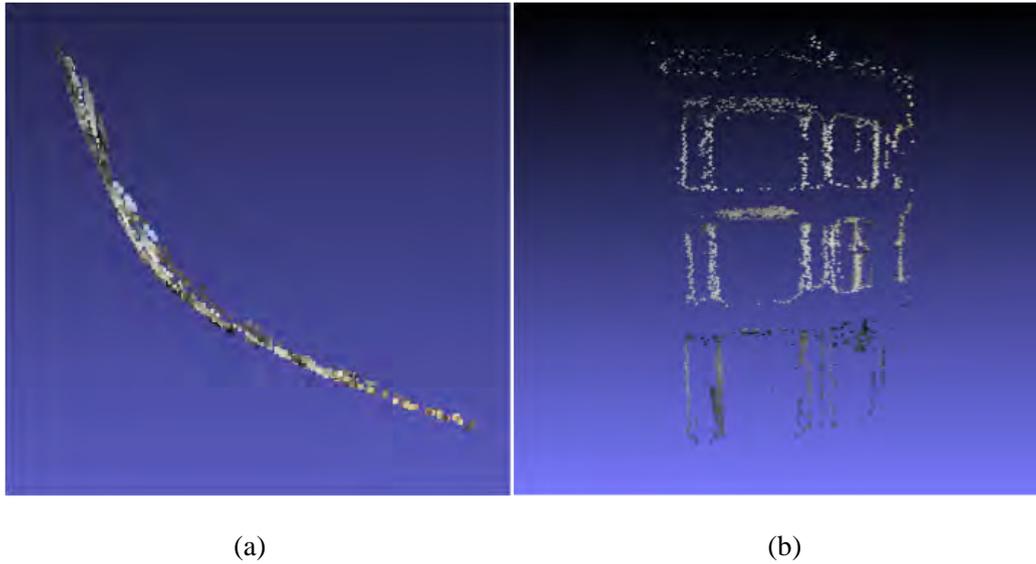


Fig. 5. Fitted curve surface of Old Stamford House. 3D points lie on the curve surface given by Curve Surface Fitting algorithm. (a) top view of the curve surface. (b) frontal view of the curve surface.

### 3.3.3 Robust Algorithm

If the surface fitting algorithm is only performed once, many outliers which don't stay on the surface, will be included into the point set for surface fitting. As a result, the surface parameters are not accurate. In order to refine surface parameters, we need to discard these outliers by iteratively running surface fitting algorithm.

The error term can be defined as

$$e_i = (z_i - S(x, y))^2 \quad (7)$$

Then, we compute the median of squared error, and denote it as  $E$

$$E = \text{median}_i(e_i) \quad (8)$$

The simplified robust algorithm is summarized as below:

$P^*$  : 3D point set for surface fitting     $\tau$  : minimum error

$k$  : minimum number of inliers

Step1: Initialization:  $P^* \leftarrow$  Original 3D points ,  $E \leftarrow \infty$

Step2: Repeat while (  $E > \tau$  and  $|P^*| > k$  ) :

- (1) Fit a surface  $S$  to the point set  $P^*$
- (2) Compute robust error  $E$  of  $S$  on  $P^*$
- (3) Remove points with residual  $e_i \geq E$  from  $P^*$

Empirical tests show that a threshold of  $\tau = 5 \times 10^{-7}$  gives good results. Figure 6 shows that the algorithm can converge within a small number of iterations and the solution has very small error. Although standard robust algorithm such as RANSAC may be used, the above algorithm is much more efficient than RANSAC. It is stable and accurate because of large number of inliers available in the point cloud for accurate fitting.

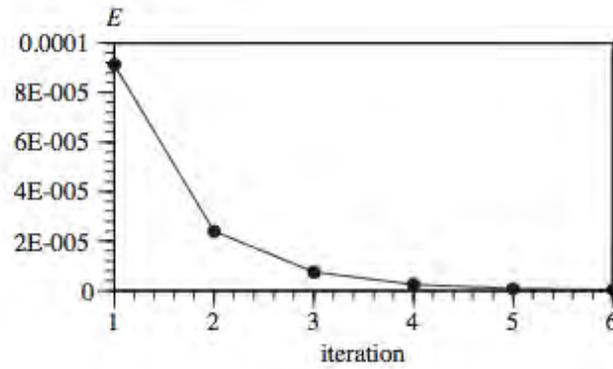


Fig. 6. Convergence curve of robust surface fitting. Median error  $E$  drops after each surface fitting iteration.

After each refinement, points with relatively large error terms will be regarded as outlier and get rejected, and thus set  $P^*$  becomes finer. With each newly filtered set  $P^*$ , the surface parameter  $C$  gets refined.

If we only perform one iteration of surface fitting, sort  $(e_i)$  in ascending order and take  $n$  points with smallest  $e_i$  to calculate surface parameter  $C$ , this would result in large inaccuracy of curve surface parameter  $C$ . This can be observed from Fig. 6, where the median error  $E$  in 1<sup>st</sup> iteration is significantly larger than median error in 2<sup>nd</sup> iteration and afterwards. That's because the 3D point set  $P$  for 1<sup>st</sup> iteration of surface fitting, contains many outliers. Therefore, it is necessary to reject outliers gradually, based on a more and more accurate surface parameter  $C$ , which gets refined after each iteration of surface fitting.

The accuracy of parameter  $C$  is critical to determine the shape of curved surface. It helps to avoid seams when joining a curved surface and a plane together, which will be elaborated in the following chapter 3.4 . Moreover, the accuracy of parameter  $C$  is

crucial to establish a correct correspondence between a 3D point on the curved surface and a 2D point in its viewing image.

Below is an example of a curved surface region of old Stamford House. Before the curved surface refinement, there are 34673 points including outliers, e.g. Traffic lights ahead of a building, and background noise (see Fig. 7a). After refinement, the shape of the curved surface is much clearer and finer. Only 7462 points are used to compute the curved surface parameter. Outliers and other non critical points have been rejected (See Fig. 7b).

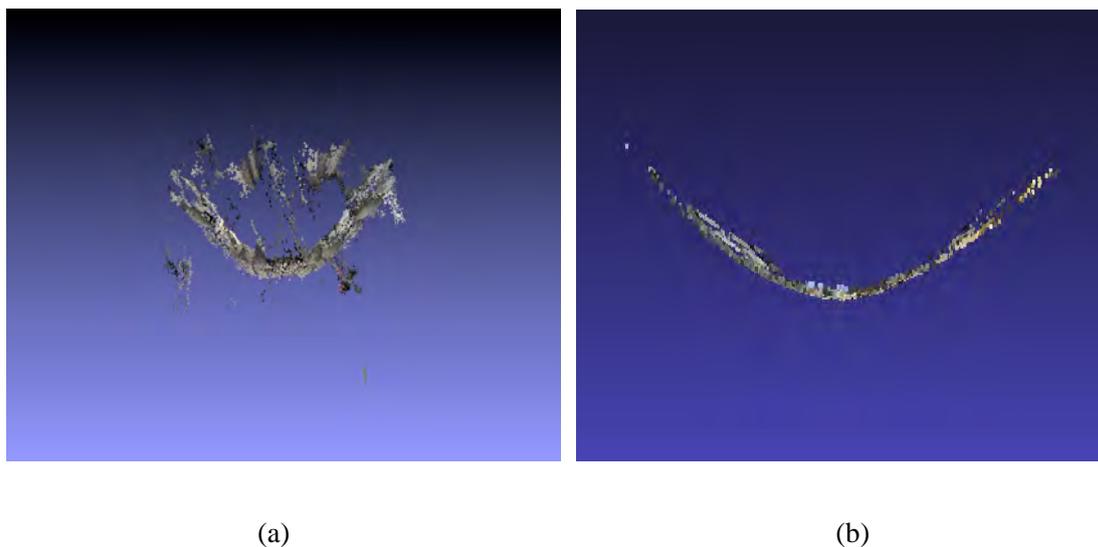


Fig 7. 3D points near the curve surface. Fig. 7 shows a bottom view of the curve surface region of Old Stamford House. (a) 3D points around a curved surface. Many outliers exist before the robust fitting algorithm. (b) Inliers identified by the robust fitting algorithm, lie very close to the fitted surface.

## 3.4 Robust Surface Splitting Algorithm

### 3.4.1 Robust Surface Splitting

Robust surface fitting algorithm in chapter 3.3 fits one single surface to a 3D point cloud. Now, I will show how to split input 3D point cloud into multiple partitions for fitting different surfaces independently.

Two non-parallel surfaces of the following forms:

$$\text{surface 1: } z = S_1(x, y)$$

$$\text{surface 2: } z = S_2(x, y)$$

The multiple views of the building are taken in such a way that the depth direction of the building is always on  $z$  direction. (See Fig. 8.)

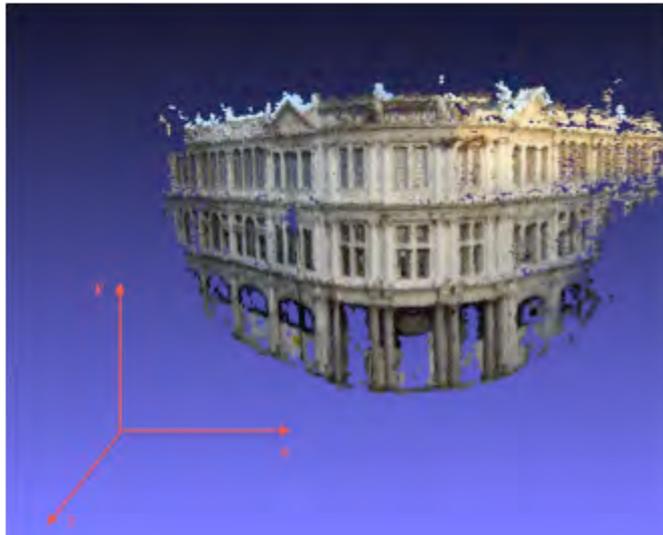


Fig. 8. Point cloud in the world coordinate.  $z$  is the depth direction.

Define function  $D(x, y, z)$  as the absolute difference between the above two surfaces

$$D(x, y, z) = |S_1(x, y) - S_2(x, y)|, \quad (9)$$

Then, the equation  $D(x, y) = 0$  denotes the intersection of the two surfaces.  $D(x, y)$  measures the distance of any surface point  $\mathbf{X}=(x, y, z)$  to the intersection of the two surfaces.

We can use a line  $l(\mathbf{X})$  to denote the intersection between two surfaces. We project the 3D data points onto the  $x$ - $y$  plane along  $z$ -axis, and use  $l(\mathbf{X})$  to split the 3D points into two different partitions for fitting two different surfaces.

The intersection line  $l(\mathbf{X})$  on  $x$ - $y$  plane is defined as follows:

$$l(\mathbf{X}) = b_1 x + b_2 y + b_3 = 0$$

Then, points  $\mathbf{X}_i$  not lying on the line have non-zero values  $l(\mathbf{X}_i)$ . So, a set of 3D points  $\mathbf{X}_i$  can be split into two subsets according to the sign of  $l(\mathbf{X}_i)$ . After splitting, a surface can be fitted to each of the two subsets using the robust surface fitting algorithm. The intersection of these surfaces induces a change of  $l(\mathbf{X})$ , and the whole process can be iterated to obtain the optimal splitting line.

The robust surface splitting algorithm is summarized as follows:

Step1: Let  $l$  be the initial intersection,  $P$  be the 3D point cloud to be split

Step2: Repeat:

- (1) Split  $P$  into  $P_1$  and  $P_2$  according to the sign of  $l(\mathbf{X}_i)$ , for all points  $\mathbf{X}_i \in P$
- (2) Perform robust fitting of surface  $S_1$  on  $P_1$  and  $S_2$  on  $P_2$
- (3) Compute  $D(\mathbf{X}_i)$  for each point  $\mathbf{X}_i$  and select a subset  $Q$  of points with the smallest  $D(\mathbf{X}_i)$ , which are near the intersection.
- (4) Fit  $l$  to the points in  $Q$  using linear least square method.

This algorithm is iterated for a fixed number of times to obtain the optimal intersection to split the input point cloud into different independent partitions and fit surfaces respectively (see Fig.9). In the current implementation, the size of  $Q$  is 50, and empirical tests show that 2 iterations are sufficient. The above surface splitting algorithm is repeatedly applied to different parts of the input point cloud to robustly split and recover multiple surfaces of a building. To define the boundaries of a quadrilateral surface, 4 lines are needed.

After performing the above algorithm, we can get not only the intersection line  $l$ , but also the points in subset  $Q$  on the intersection, which can be used as corresponding points to remove seam between two surfaces in chapter 3.4.2.

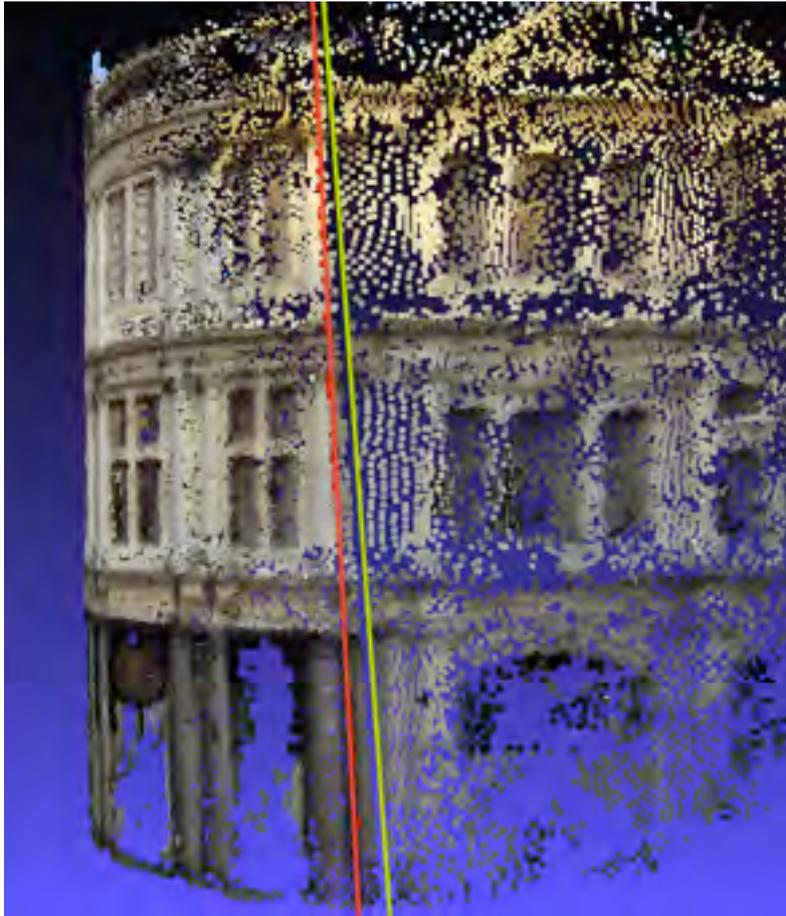


Fig. 9. Splitting of point cloud to reconstruct multiple surfaces. Initial intersection (Green).

Refined intersection (Red).

### 3.4.2 Alignment of Surfaces

Seam could be observed around the intersection where two surfaces are split (see Fig. 10(a)). Without changing the parameters of the plane or the curved surface, similarity transformation can be used to align two adjacent surfaces to remove the seam or gap.

The points on the intersection  $l$  obtained from the above algorithm can be used to get the corresponding points  $\mathbf{X}_i$  and  $\mathbf{X}_i'$  on surface  $S_1$  and  $S_2$  respectively as follows:

Recall that  $Q$  is the subset containing the points on the intersection  $l$

$$\forall (X_i, Y_i, Z_i) \in Q,$$

Let  $\mathbf{X}_i = (x_i, y_i, z_i)^T$  on surface  $S_1$  and  $\mathbf{X}_i' = (x_i', y_i', z_i')^T$  on surface  $S_2$ , where

$$x_i = x_i' = X_i$$

$$y_i = y_i' = Y_i$$

$$z_i = S_1(X_i, Y_i)$$

$$z_i' = S_2(X_i, Y_i)$$

Thus  $\mathbf{X} = (\mathbf{X}_1, \dots, \mathbf{X}_N)$  are the  $N$  points on surface  $S_1$ ,  $\mathbf{X}' = (\mathbf{X}_1', \dots, \mathbf{X}_N')$  are their corresponding points on surface  $S_2$ , then we can compute the best similarity transformation to align the two surfaces using well-known algorithms such as [19, 35]. Here, I present the algorithm of [19] for completeness.

we will transform  $\mathbf{X}$  to align with  $\mathbf{X}'$  by the following equation

$$\mathbf{X}' = s\mathbf{R}\mathbf{X} + \mathbf{T} \quad (10)$$

(1) Centralize each point by subtracting its mean vector:

$$\mathbf{r}_i = \mathbf{X}_i - \bar{\mathbf{X}}, \quad \mathbf{r}'_i = \mathbf{X}'_i - \bar{\mathbf{X}}' \quad (11)$$

$$\text{where } \bar{\mathbf{X}} = \frac{1}{n} \sum_{i=1}^n \mathbf{X}_i, \quad \bar{\mathbf{X}}' = \frac{1}{n} \sum_{i=1}^n \mathbf{X}'_i \quad (12)$$

(2) Determine the scaling factor by comparing variance:

$$s^2 = \frac{\sum_{i=1}^n \|\mathbf{r}'_i\|^2}{\sum_{i=1}^n \|\mathbf{r}_i\|^2} \quad (13)$$

(3) Compute rotation matrix as follows:

Form matrix  $\mathbf{M}$  from sum of outer product

$$\mathbf{M} = \sum_{i=1}^n \mathbf{r}'_i \mathbf{r}_i^T \quad (14)$$

Then rotation matrix is given by

$$\mathbf{R} = \mathbf{M}\mathbf{Q}^{-1/2} \quad (15)$$

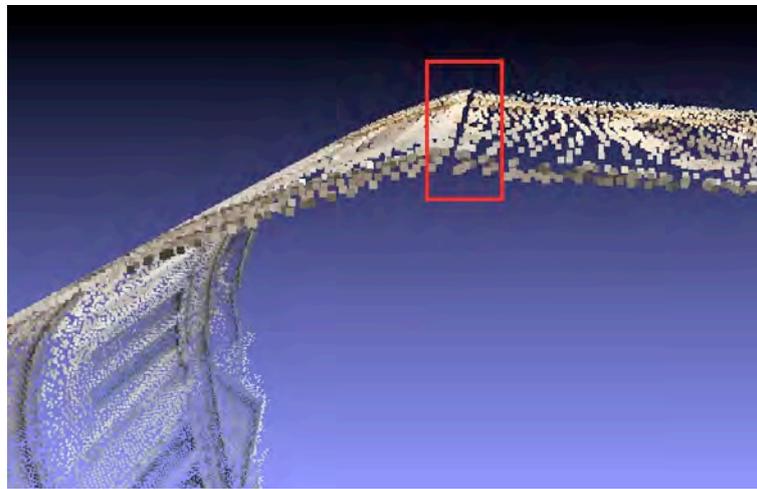
$$\text{where } \mathbf{Q} = \mathbf{M}^T \mathbf{M} \quad (16)$$

(4) Compute translation matrix by

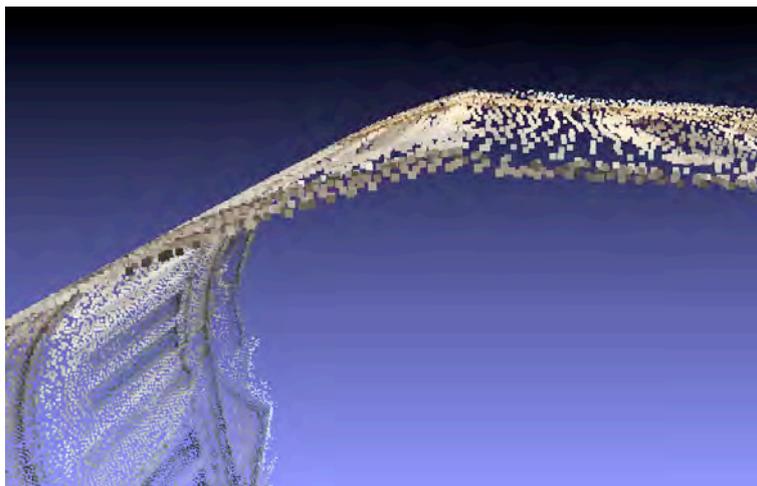
$$\mathbf{T} = \bar{\mathbf{X}}' - s\mathbf{R}\bar{\mathbf{X}} \quad (17)$$

For a degenerate case, when  $\mathbf{X}$  and  $\mathbf{X}'$  lie on the same plane, we can use  $\text{real}(\mathbf{Q}^{-1/2})$  to approximate  $\mathbf{Q}^{-1/2}$ , since coefficients of its complex part is close to 0. Another approach is to use singular value decomposition.

With  $s$ ,  $\mathbf{R}$  and  $\mathbf{T}$  obtained, we can transform one surface based on Eq. (10) to join the other adjacent surface. By doing so, seam between two surfaces can be removed.(see Fig. 10.)



(a)



(b)

Fig. 10. Alignment of surfaces. (a) Before alignment, seam (red line) exists between two surfaces. (b) After alignment, seam is removed.

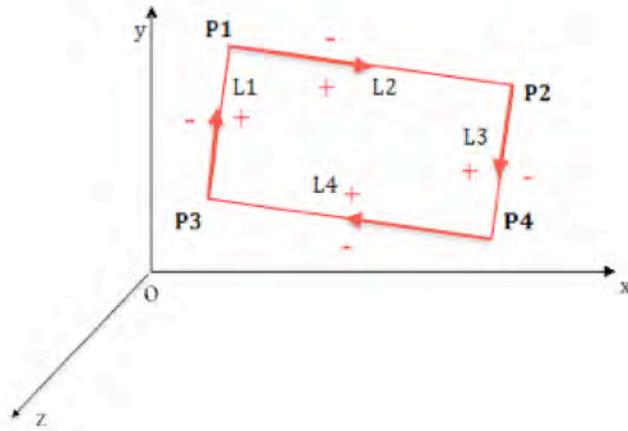
## 3.5 Surface Resampling

### 3.5.1 Resampling of Surface Points

The 3D point cloud recovered by PMVS in the 1<sup>st</sup> stage usually does not cover a surface completely with some missing data on the surface. In order to obtain the complete texture for a surface, it is necessary to resample 3D point on the reconstructed surfaces given the parametric equations of surfaces obtained in the 2<sup>nd</sup> stage. Then, the color information can be retrieved from frontal input images.

To resample 3D points on a surface, we can first project the surface on  $x$ - $y$  plane, grid sample  $(x_i, y_i)$  of each 3D point in the projected region, and compute  $z_i$  of each 3D point based on parametric equation of the surface. Then we can resample the 3D point  $(x_i, y_i, z_i)$  on the reconstructed surface.

Suppose polygon  $P_1P_2P_4P_3$  is the projection of a surface onto  $x$ - $y$  plane, we need to grid sample points  $(x_i, y_i)$  inside the polygon.  $P_1, P_2, P_3, P_4$  are four corner points manually marked on the  $x$ - $y$  plane. A line segment  $L(\mathbf{X})$  is computed between two connected corners such that  $L(\mathbf{X})$  is positive for points inside the resampling region and negative for points outside. This property is true for all convex regions and is easy to determine whether a point is inside the resampling region.



We add a direction for each edge, such that the orientation is clockwise. Then for a point to be sampled inside the polygon, it must be always on the positive side for all four directed edges.

Suppose  $\mathbf{P}_1 = (x_1, y_1)$ ,  $\mathbf{P}_2 = (x_2, y_2)$

Line  $\mathbf{P}_1\mathbf{P}_2$  is given by

$$a_{11} x + a_{21} y + a_{31} = 0$$

Then,

$$a_{11} x_1 + a_{21} y_1 + a_{31} = 0$$

$$a_{11} x_2 + a_{21} y_2 + a_{31} = 0$$

Therefore,

$$a_{11}(x_1 - x_2) + a_{21}(y_1 - y_2) = 0 \quad (18)$$

For a convex polygon, choose the center point  $\mathbf{P}_0 = (\mathbf{P}_1 + \mathbf{P}_2 + \mathbf{P}_3 + \mathbf{P}_4) / 4 = (x_0, y_0)$ ,

$\mathbf{P}_0 = (x_0, y_0)$  will be always inside the polygon.

Choose  $(a_{11}, a_{21})$  s.t. Eq. (18) is true, and

$$a_{11}(x_0 - x_1) + a_{21}(y_0 - y_1) \geq 0 .$$

$\forall \mathbf{P}=(x, y)$ , if  $\mathbf{P}$  is on the positive side of Line  $\mathbf{P}_1\mathbf{P}_2$  , then

$$a_{11}(x - x_1) + a_{21}(y - y_1) \geq 0 , \text{ must be satisfied.}$$

Similarly, if  $\mathbf{P}$  lies on the positive side of all four directed edges, then

$$a_{1i}(x - x_i) + a_{2i}(y - y_i) \geq 0 , \text{ for } i= 1, 2, 3, 4, \text{ must be satisfied. (19)}$$

After recovery the parameters of the four lines, we can sample points  $(x, y)$  inside the region by satisfying the condition:

$$a_{1i}x + a_{2i}y + a_{3i} \geq 0, \text{ for } i= 1, 2, 3, 4.$$

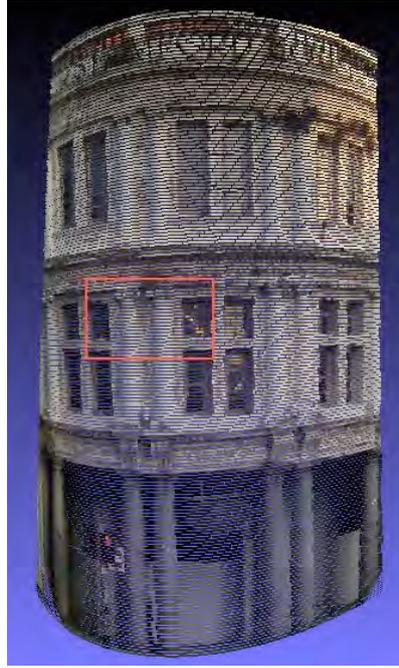
Since surface is given,  $z$  can be computed with  $(x, y)$  using the equation of surface (Eq.1, Eq.4).

$$z = \mathbf{S}(x, y)$$

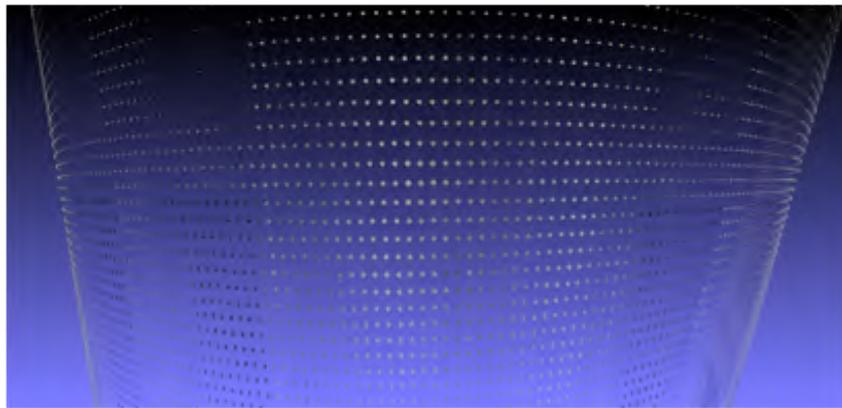
Then we can resample each 3D point  $(x, y, z)$  on the reconstructed surface (See Fig. 11) at regular spacing. Note that the resampling rate is determined by the user according to the resolution required and it is independent of the sampling density of the point cloud.



(a)



(b)



(c)

Fig. 11. Resampled surface points. (a) 3D points in initial point cloud is sparse and missing data points in some regions (b) Resampled 3D points cover the surface completely (c) Zoomed-in view of resampled 3D points inside the red box.

### 3.5.2 Color Retrieval from Frontal Image

After all points are resampled in 3D space, next we want to display them with color. We will choose a default color for the roof, the bottom and the back of a building. But for main structures reconstructed from multiple images, we can retrieve color pixel for each 3D point from its viewing image.

Let  $(x, y, z, 1)^T$  denotes a homogeneous coordinate of a resampled point in 3D space and  $(u, v, 1)^T$  denotes its homogeneous 2D image coordinate. They are related by the following equation

$$d \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \mathbf{P} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}, \quad (20)$$

where  $\mathbf{P} = \mathbf{K}(\mathbf{R}|\mathbf{T})$ .  $\mathbf{R}$  is the Rotation Matrix,  $\mathbf{T}$  is Translation vector and  $\mathbf{K}$  is the projection matrix of the camera.

Eq. (20) can also be written as

$$d \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{bmatrix} \mathbf{P}_1^T \\ \mathbf{P}_2^T \\ \mathbf{P}_3^T \end{bmatrix} \mathbf{X}, \quad (21)$$

where  $\mathbf{X} = (x, y, z, 1)^T$

Compute  $d$  first,

$$d = \mathbf{P}_3^T \mathbf{X}$$

Then we have

$$u = \frac{\mathbf{P}_1^T \mathbf{X}}{\mathbf{P}_3^T \mathbf{X}}, \quad v = \frac{\mathbf{P}_2^T \mathbf{X}}{\mathbf{P}_3^T \mathbf{X}} \quad (22)$$

If we choose image  $I$  as the viewing image for resampled 3D points, then the color pixel of each point  $X$  can be found at  $I(v,u)$ . Finally, the whole structure is determined and resampled accordingly by 3D points with their colors for display purpose (see Fig. 12).



Fig. 12. Resampled 3D points with colors

### 3.5.3 Removal of Distortion and Color Blending

After we reconstruct the whole structure and display with color pixels, some part of the building might be occluded by trees, traffic lights or may have severe image distortion due to oblique views (see Fig. 13(a)).

One way to avoid such distortion is to reconstruct that blocked part of building from another better view, for instance a frontal view. So, for the distorted plane, instead of using image  $I_1$ (see Fig. 13(c)), we will manually choose another image  $I_2$  with frontal view for reconstruction (see Fig. 13(d)). And the color pixel for each 3D points on the distorted plane will be  $I_2(v', u')$  from Eq. (22).

By using a frontal view for reconstruction, image distortion and occlusion can be reduced quite efficiently. However, since color pixels of 3D points come from different images, there might be a color seam observed at the intersection line of curved surface and plane (see Fig. 14).



(a)

(b)

(c)

Fig. 13. Removal of distorted color texture. (a) Color texture with gross distortion. (b) Color retrieved from a frontal view image. Distortion is removed, but a visible seam is induced

(c) Color blending removes the seam.



Fig. 14. Selected region for color blending (right rectangular box in red)

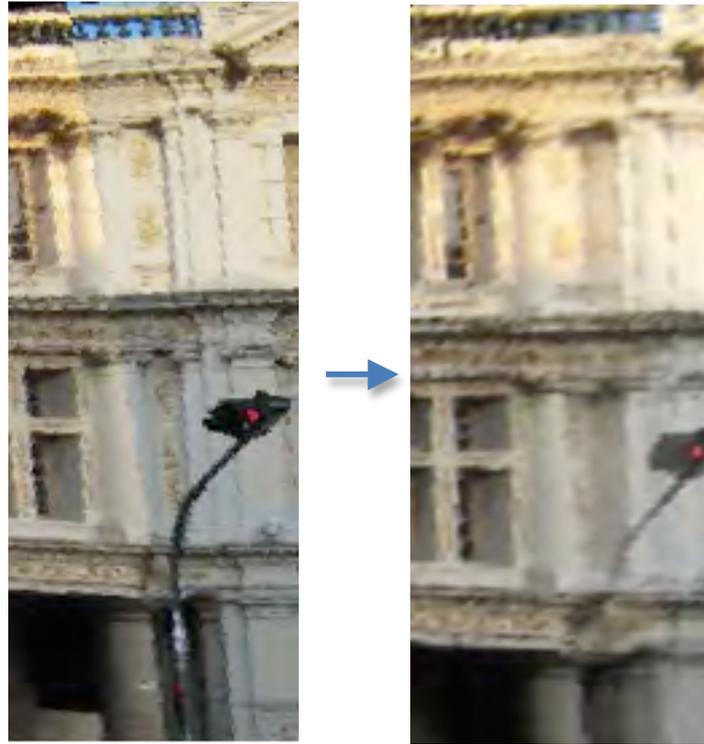
On the left of the color seam (red dotted line), color is retrieved from  $I_1$  (See Fig. 13(c)). On the right of the seam, color is retrieved from  $I_2$  (See Fig. 13(d)). We can blend colors of  $I_2$  on the right of the seam with color of  $I_1$  in the same location to reduce this seam.

$w$  : an effective width to define the region for color blending

$d$  : distance to color seam

$$\text{Blended Color} = (1 - d/w) * I_1(v,u) + (d/w) * I_2(v',u'), \text{ for } d \in [0,w]$$

After blending two colors from  $I_1$  and  $I_2$ , we can see that the color seam can be reduced quite effectively (see Fig. 15).



(a)

(b)

Fig. 15. Color blending. (a) color seam exists before blending (b) Color seam is removed after blending.

### **3.6 Construction of 3D Mesh Model**

Since the 3D points in chapter 3.5 are sampled in such a way that the unit distance between any two neighboring points are similar, it will be easier to construct a 3D mesh model with color on vertex based on the resampled 3D points, to cover up the empty space between two adjacent 3D points. Here, we adopt ball pivoting algorithm [3] to reconstruct surface. The reconstruction process starts with a seed triangle, and pivot a ball of a given radius around the edges of the seed triangle to touch another point and hence create a new triangle. After the reconstruction process, meshes are created in the point cloud. From its mechanism, we can deduce and observe that ball pivoting algorithm works well on our uniformly sampled points (see Fig 16).



(a)



(b)

Fig. 16. 3D Mesh model of Old Stamford House. (a) Frontal view. (b) Zoomed-in view of the red box in (a).

## 4. Test Results & Discussions

### 4.1 Test Data

To reconstruct 3D model of SMRT Headquarter Building, 27 images from different views have been taken. The SMRT Headquarter Building locates in City Hall District, 251 North Bridge Road, Singapore 179102.

Some examples are shown below. (See Fig. 17.)



Fig. 17. Input images of SMRT Headquarters from different views.

To reconstruct 3D model of Old Stamford House, 31 images from different views have been taken. The Old Stamford House locates at the corner of the junction of Stamford Road and Hill Street, in the Downtown Core of Singapore. Some examples are shown below. (See Fig. 18.)



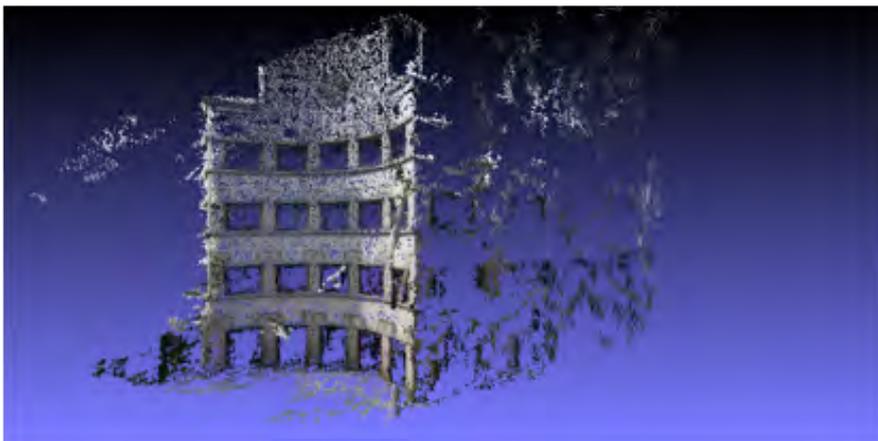
Fig. 18. Input images of Old Stamford House from various views.

## 4.2 Reconstructed 3D Models

### 4.2.1 Results of Stage 1: 3D Point Cloud given by PMVS



(a)



(b)

Fig. 19. 3D point cloud of SMRT Headquarters recovered by PMVS. (a) left view

(b) right view

From Fig. 19(b), we can see very clearly most part of the right plane is empty and hence it will be rather difficult to reconstruct surface directly on the right plane. But with plane fitting algorithm, we can figure out the accurate parametric equation of the plane, resample points on it and therefore reconstruct the plane.



(a)



(b)



(c)

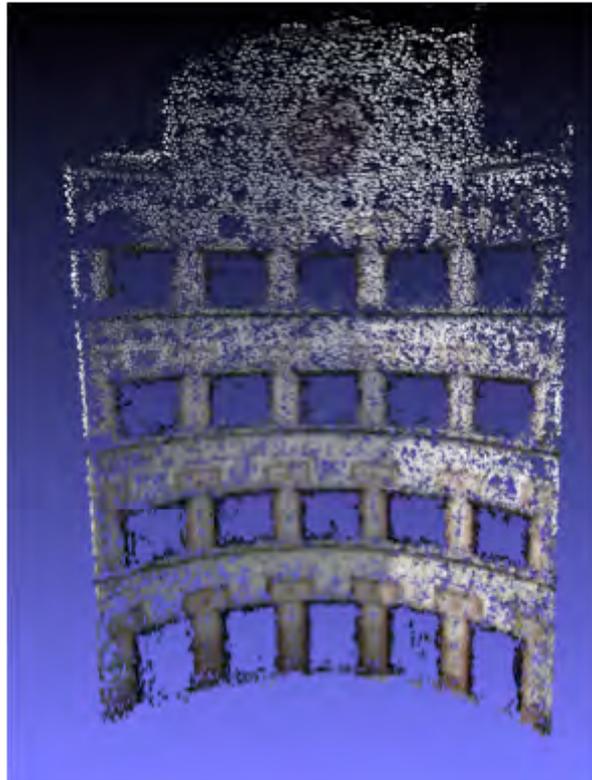
Fig. 20. 3D point cloud of Old Stamford House recovered by PMVS. (a)-(b) First point cloud. (c) Second point cloud.

From Fig. 20(a), we can see there are many outliers around the surface. Moreover, large region of empty space can be observed on the right plane of the building.

Therefore, we need a robust surface fitting algorithm to filter those outliers and get the parametric equation of the surface. Then resample 3D points on the surface to cover those empty space on the right plane.

#### 4.2.2 Results of Stage 2: Refined Surface given by Robust Surface Fitting

##### Algorithm



(a)

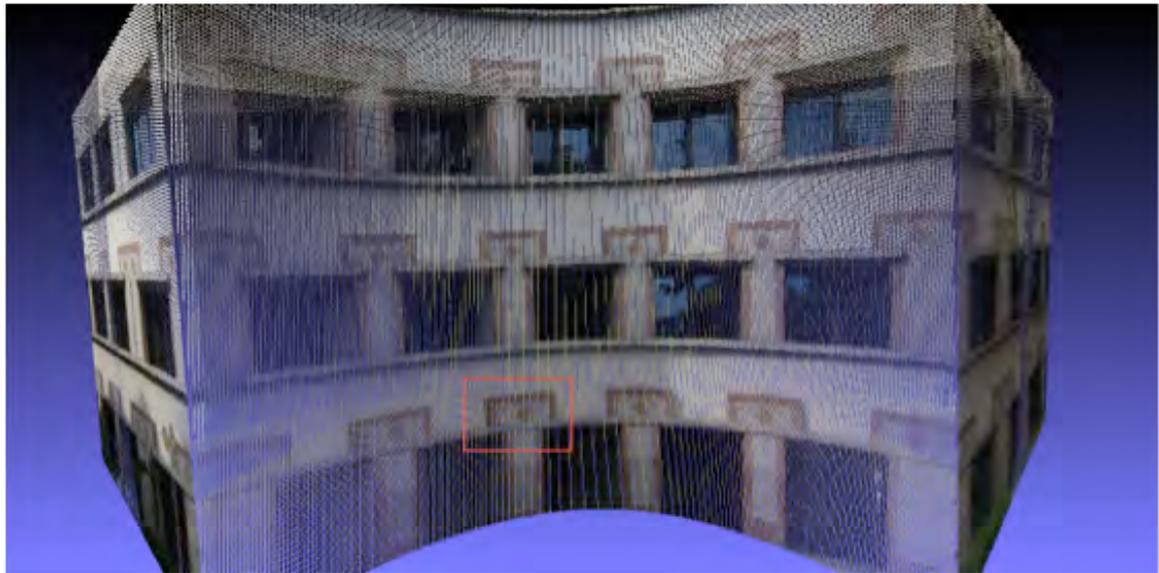


(b)

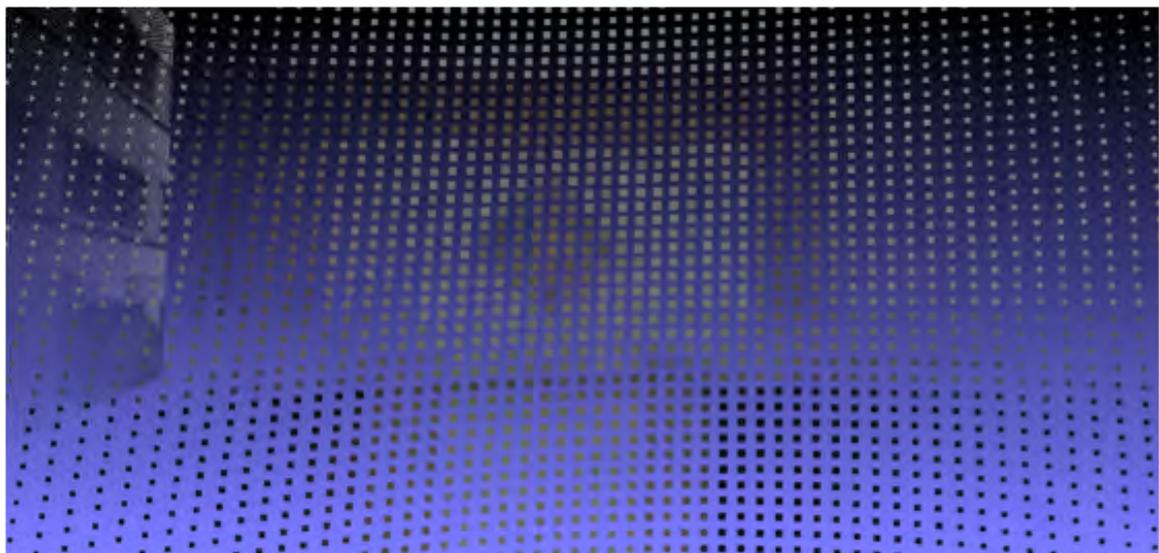
Fig. 21. Refined surface of SMRT Headquarters. (a) frontal view (b) bottom view

28370 points lie on the curve surface. Outliers are discarded. The median squared error is smaller than  $5 \cdot 10^{-7}$ .

### 4.2.3 Results of Stage 3: Resampled 3D Points on Reconstructed Surfaces



(a)



(b)

Fig. 22. Resampled 3D points on surface of SMRT Headquarters.

(a) complete view of resampled 3D points on surface.

(b) zoomed in view of the red box in (a).

151944 points are resampled on the surface of SMRT Headquarters building with color retrieved from frontal images. This point cloud is more dense than original point cloud given by PMVS. Moreover, these resampled points have covered the entire surface of the building, i.e. left plane, right plane, and curve surface in the middle, leaving no missing regions on the surface.



(a)



(b)

Fig. 23. Resampled 3D points on surface of Old Stamford House.

(a) complete view of resampled 3D points on surface.

(b) zoomed in view of the red box in (a).

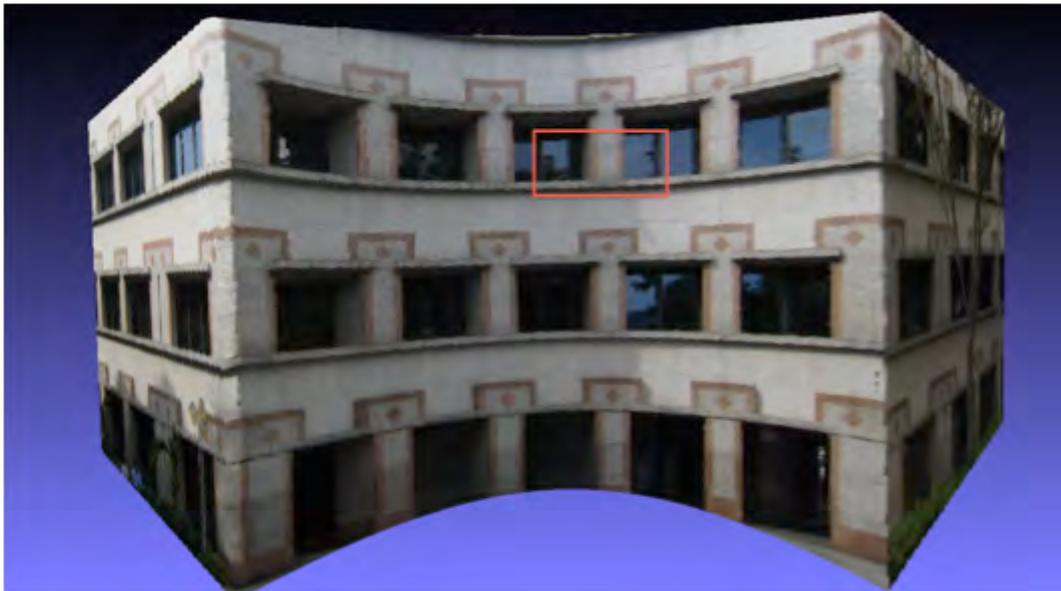
106225 points are resampled on the surface of Old Stamford House. The resampled points with colors can effectively reduce distortion or misalignment, which often occurs, when we directly texture map an 2D image onto a curve surface or a plane.

#### 4.2.4 Final 3D Mesh Model



(a)

(b)



(c)



(d)

Fig. 24. Reconstructed mesh model of SMRT Headquarters. (a) Left view. (b) Right view. (c) Frontal view. (d) Zoomed-in mesh of the red box in (c).



Fig. 25. Reconstructed mesh model of Old Stamford House. (Row 1-2) Mesh model from various views. (Row 3) Zoom-in view of the region in the red box above.

**Table 1.** Results of mesh surface reconstruction. Execution times are measured in minutes, excluding manual inputs.

|                   | No. of images | No. of point clouds | No. of surfaces | Bundler run time (mins) | PMVS run time (mins) | My algo run time (mins) | Total run time (mins) |
|-------------------|---------------|---------------------|-----------------|-------------------------|----------------------|-------------------------|-----------------------|
| Stamford House    | 105           | 2                   | 3               | 79                      | 47                   | 14                      | 140                   |
| SMRT Headquarters | 27            | 1                   | 3               | 20                      | 12                   | 6                       | 38                    |

Two large buildings with curve surfaces were used as the test cases: one with a convex curved surface, the other with a concave curved surface. Multiple images of the buildings were taken from various viewing angles. For each test case, PMVS algorithm was executed on the multiple views to recover 3D point clouds, and our complete surface reconstruction algorithm was executed to reconstruct the surfaces.

Table 1 shows the results of applying the algorithms on the test cases. Building 1 had an extended wall and required two separate point clouds that capture the various walls. Bundler and PMVS took majority of the time to compute the matching feature points, camera parameters, and point cloud. Fig17- Fig25 show samples inputs, recovered point clouds, and reconstruction results of the buildings. Notice that the point clouds are sparse in some parts of the surfaces. Nevertheless, our algorithm can resample the color textures in those areas from the input images and reconstruct complete mesh of the surfaces.

## **5. Conclusion**

### **5.1 Summary**

This thesis presented a simple, inexpensive, and effective method for reconstructing textured mesh surfaces of large buildings with curved surfaces. It applies PMVS algorithm to recover point cloud from multiple input images. Then, robust surface fitting and splitting algorithms are applied to fit multiple surfaces to different parts of the point cloud. These surfaces are then aligned, merged and color blended to produce a single mesh model of the building, completed with color texture. The mesh models can be used directly in various applications. Test results show that the 3D models reconstructed by the algorithm are sufficiently accurate and realistic for 3D visualization in various applications.

### **5.2 Future Work**

This thesis demonstrated the reconstruction of surfaces from a single point cloud. For a large building, multiple point clouds need to be recovered to cover different parts of the building. Our algorithm can be applied to the point clouds to reconstruct the surfaces separately, and then align and merge them into a single mesh model. Reconstruction of a large building that is close in proximity to other buildings or constructions is a very challenging task. The challenge is to devise a scheme that can reconstruct the entire building with the least number of input images and point clouds.

# Bibliography

1. Allen, P.K., Troccoli, A., Smith, B., Murray, S., Stamos, I., Leordeanu, M.: New methods for digital modeling of historic sites. *IEEE Computer Graphics and Applications* 23(6), 32–41 (2003)
2. Beardsley, P., Torr, P., Zisserman, A.: 3D model acquisition from extended image sequences. In: *Proc. ECCV*, pp. II:683–695 (1996)
3. Bernardini, F., Mittleman, J., Rushmeier, H., Silva, C., Taubin, G.: The ball-pivoting algorithm for surface reconstruction. *IEEE Trans. Visualization and Computer Graphics* 5(4), 349–359 (1999)
4. Blais, F.: Review of 20 years of range sensor development. *J. Electronic Imaging* 13(1), 232–240 (2004)
5. Cornelis, N., Cornelis, K., Van Gool, L.: Fast compact city modeling for navigation pre-visualization. In: *Proc. CVPR* (2006)
6. Cornelis, N., Leibe, B., Cornelis, K., van Gool, L.J.: 3D urban scene modeling integrating recognition and reconstruction. *Int. J. Computer Vision* 78(2–3), 121–141 (2008)
7. Dellaert, F., Seitz, S.M., Thorpe, C.E., Thrun, S.: Structure from motion without correspondence. In: *Proc. IEEE CVPR* (2000)
8. El-Hakim, S., Whiting, E., Gonzo, L., Girardi, S.: 3-D reconstruction of complex architectures from multiple data. In: *Proc. ISPRS Int. Workshop on 3D Virtual Reconstruction and Visualization of of Complex Architectures* (2005)

9. Frueh, C., Zakhor, A.: Constructing 3D city models by merging aerial and ground views. *IEEE Computer Graphics and Applications* 23(6), 52–61 (2003)
10. Furukawa, Y., Curless, B., Seitz, S., Szeliski, R.: Towards internet-scale multi-view stereo. In: *Proc. CVPR* (2010)
11. Furukawa, Y., Ponce, J.: Patch-based multi-view stereo software. <http://www.di.ens.fr/pmvs/>
12. Furukawa, Y., Ponce, J.: Accurate, dense, and robust multi-view stereopsis. In: *Proc. CVPR* (2007)
13. Furukawa, Y., Ponce, J.: Accurate, dense, and robust multi-view stereopsis. *IEEE Trans. PAMI* 32(8), 1362–1376 (2010)
14. Guidi, G., Micoli, L., Russo, M., Frischer, B., De Simone, M., Spinetti, A., Carosso, L.: 3D digitization of a large model of imperial rome. In: *Proc. Int. Conf. 3D Imaging and Modeling*, pp. 565–572 (2005)
15. Habbecke, M., Kobbelt, L.: Iterative multi-view plane fitting. In: *Proc. Fall Workshop on Vision, Modeling, and Visualization* (2006)
16. Hartley, R., Zisserman, A.: *Multiple View Geometry in Computer Vision*. Cambridge University Press (2000)
17. Havlena, M., Torii, A., Knopp, J., Pajdla, T.: Randomized structure from motion based on atomic 3D models from camera triplets. In: *Proc. CVPR* (2009)
18. Havlena, M., Torii, A., Pajdla, T.: Efficient structure from motion by graph optimization. In: *Proc. ECCV* (2010)
19. Horn, B.K.P., Hilden, H.M., Negahdaripour, S.: Closed- form solution of absolute

- orientation using orthonormal matrices. *J. Optical Society of America A* 5(7), 1127–1135 (1988)
20. Jancosek, M., Pajdla, T.: Multi-view reconstruction preserving weakly-supported surfaces. In: *Proc. CVPR* (2011)
21. Kazhdan, M., Bolitho, M., Hoppe, H.: Poisson surface reconstruction. In: *Proc. Int. Symp. Geometry Processing*, pp. 61–70 (2006)
22. Labatut, P., Pons, J., Keriven, R.: Robust and efficient surface reconstruction from range data. *Computer Graphics Forum* (2009)
23. Labatut, P., Pons, J.P., Keriven, R.: Efficient multi-view reconstruction of large-scale scenes using interest points, delaunay triangulation and graph cuts. In: *Proc. CVPR* (2007)
24. Lhuillier, M., Quan, L.: A quasi-dense approach to surface reconstruction from uncalibrated images. *IEEE Trans. PAMI* 27(3), 418–433 (2005)
25. Liang, Y., Sheng, Y., Song, H., Wang, Y.: 3D reconstruction of architecture from image sequences
26. Micusik, B., Kosecká, J.: Piecewise planar city 3D modeling from street view panoramic sequences. In: *Proc. CVPR* (2009)
27. Moons, T., Van Gool, L., Vergauwen, M.: 3D reconstruction from multiple images part 1: Principles. *Foundations and Trends in Computer Graphics and Vision* 4(4), 287–398 (2008)
28. Morris, D., Kanade, T.: A unified factorization algorithm for points, line segments and planes with uncertainty models. In: *Proc. ICCV*, pp. 696–702 (1998)

29. Poelman, C., Kanade, T.: A paraperspective factorization method for shape and motion recovery. *IEEE Trans. PAMI* 19(3), 206–218 (1997)
30. Seitz, S.M., Curless, B., Diebel, J., Scharstein, D., Szeliski, R.: A comparison and evaluation of multi-view stereo reconstruction algorithms. In: *Proc. CVPR* (2006)
31. Snavely, N.: Bundler: Structure from motion (SfM) for unordered image collections. <http://phototour.cs.washington.edu/bundler/>
32. Open Source Photogrammetry, <http://opensourcephotogrammetry.blogspot.com/2010/12/newpackage-available-for-windows.html>
33. Snavely, N., Seitz, S.M., Szeliski, R.: Photo tourism: Exploring image collections in 3D. In: *Proc. SIGGRAPH* (2006)
34. Torii, A., Havlena, M., Pajdla, T.: From google street view to 3D city models. In: *Proc. Workshop on Omni-directional Vision, Camera Networks and Non-classical Cameras* (2009)
35. Umeyama, S.: Least-square estimation of transformation parameters between two point patterns. *IEEE Trans. PAMI* 12(4), 376–380 (1991)
36. Vosselman, G., Dijkman, S.: 3D building model reconstruction from point clouds and ground plans. In: *Int. Archives of Photogrammetry and Remote Sensing*, vol. XXXIV-3/W4 (2001)
37. Xiao, J., Fang, T., Zhao, P., Lhuillier, M., Quan, L.: Image-based street-side city modeling. In: *Proc. SIGGRAPH Asia* (2009)