Symmetric Architecture Modeling with a Single Image

Nianjuan Jiang Ping Tan Loong-Fah Cheong

Department of Electrical & Computer Engineering, National University of Singapore

(a) (b) (c)

Figure 1: A traditional Chinese architecture, the Pavilion of Manifest Benevolence (also known as TiRen Ge) in the Forbidden City, is modeled from a single input image. (a) is the input image overlaid with user-drawn strokes. (b) is the rendering of the recovered model from the same viewpoint as the input image for validation. (c) shows the rendering from a novel viewpoint.

Abstract

We present a method to recover a 3D texture-mapped architecture model from a single image. Both single image based modeling and architecture modeling are challenging problems. We handle these difficulties by employing constraints derived from shape symmetries, which are prevalent in architecture. We first present a novel algorithm to calibrate the camera from a single image by exploiting symmetry. Then a set of 3D points is recovered according to the calibration and the underlying symmetry. With these reconstructed points, the user interactively marks out components of the architecture structure, whose shapes and positions are automatically determined according to the 3D points. Lastly, we texture the 3D model according to the input image, and we enhance the texture quality at those foreshortened and occluded regions according to their symmetric counterparts. The modeling process requires only a few minutes interaction. Multiple examples are provided to demonstrate the presented method.

Keywords: Architecture modeling, 3D reconstruction, symmetry

1 Introduction

Creating high quality 3D architecture models is important for many applications including digital heritage, games, movies, etc. Many methods [Debevec et al. 1996; Liebowitz et al. 1996; Müller et al. 2007; Xiao et al. 2008; Sinha et al. 2008] have been proposed for this purpose. Most of them focus on piecewise planar architectures and take multiple images as input. Planar structures induce strong shape constraint and simplify the 3D reconstruction. However, many traditional and more artistic architectures have intricate geometric structure and curved roofs, which are highly non-planar and cannot be modeled well by existing methods. Yet, these buildings are often landmarks that are particularly worthy of being modeled. Furthermore, multiple images of the same building are not always available. Thus, it is practically important to build a modeling system that works on the basis of a single input image. Single image based modeling is difficult. First, it is difficult to calibrate the camera (i.e. recovering both intrinsic and extrinsic camera parameters), which is necessary to relate the image to the 3D model. Second, a single image often does not provide enough texture information due to foreshortening and occlusion.

This paper addresses the problem of modeling complex architecture from a single image. Instead of relying on planar structures and multiple images, we advocate exploiting symmetries for 3D reconstruction. As Magdolna and Hargittai [1994] have commented, symmetry is 'a unifying concept' in architecture. A single image of a symmetric building effectively provides observations from multiple symmetric viewpoints [Zhang and Tsui 1998; Francois et al. 2002; Hong et al. 2004]. In other words, shape symmetry effectively upgrades a single input image to multiple images. To exploit this property, we first propose a method to calibrate the camera from a single image according to the presented symmetry. This calibration allows our system to handle images with completely unknown camera information (e.g. internet downloaded pictures and archive pictures). Then, a virtual camera is duplicated at the position symmetric to the real camera, and its observed image is derived from the input image. A stereo algorithm follows to recover a set of 3D points from the real and virtual camera pair. After that, the user interactively organizes the reconstructed 3D points into a high quality mesh model. To keep the interaction simple, the user only manipulates in the image space to mark out various architecture components such as walls and roofs, whose shapes and positions in 3D are automatically computed. Symmetric counterparts of each marked component are generated automatically to reduce the user interaction. Thanks to the strong symmetry, the modeling process typically takes less than 5 minutes of interaction. Lastly, the model is textured according to the single input image. We use symmetry again to enhance the texture quality at those foreshortened and occluded regions.

The contribution of this paper is a systematic architecture modeling method building upon ubiquitous architecture symmetries. We build a novel camera calibration algorithm (sec. 3.1), an efficient interactive architecture modeling interface (sec. 4.1) and a practi-

cal texture enhancement method (sec. 4.2). All these components prove architecture modeling can be made very efficient by making appropriate usage of symmetry-based constraints.

Figure 2 shows the pipeline of our system. We first calibrate the camera from a single image. Then we reconstruct a set of 3D points according to the calibration and architecture symmetry. Next, the user interactively marks out structural components such as roofs and walls to build an initial 3D model. The user can also add in more geometric detail, such as roof tiles and handrails, or insert predefined primitives, such as pillars and staircases. At last, the recovered model is textured according to the input image. Texture synthesis is used to improve texture quality at the foreshortened and occluded regions.

2 Related work

3D reconstruction and architecture modeling have received a lot of research interest, with a large spectrum of modeling systems developed to build realistic 3D models. Here we only review those works related to symmetry and architecture modeling. We categorize them according to their methodologies.

3D reconstruction from symmetries: It is well known that symmetry provides additional constraint for 3D reconstruction. Rothwell et al. [1993] and Francois et al. [2002] studied the 3D reconstruction of bilaterally symmetric objects. Zhang and Tsui [1998] extended it to handle arbitrary shape by inserting a mirror into the scene. Hong et al. [2004] provided a comprehensive study of reconstruction from various symmetries. Most of these works focus on bilateral symmetry and study the resulting multi-view geometric structures such as the special configurations of the fundamental matrix and epipoles. These works often assume the camera is precalibrated with known focal length and/or pose (position and orientation) for 3D reconstruction. Although Hong et al. [2004] studied the camera calibration problem from symmetries, their results are limited, e.g. the camera can be calibrated from the vanishing points of three mutually orthogonal axis of bilateral symmetry. In comparison, we focus on the application of architecture modeling and study both bilateral and rotational symmetries. Since we are more specific about the object to be modeled, we obtained stronger results both on camera calibration and texture creation, which lead to a complete system for high quality modeling with a single uncalibrated image.

Procedural architecture modeling: Procedural methods build 3D architecture models from rules and shape grammars. They can generate highly detailed models at the scale of both an individual building and a whole city [Parish and Müller 2001; Müller et al. 2006]. A disadvantage of these methods is that it takes expertise for its effective usage. It is also hard to specify rules to model a particular building.

Interactive architecture modeling Façade [Debevec et al. 1996] fitted a parametric building model to the single (or multiple) input image(s) according to the user marked geometric primitives. High quality results can be achieved. There are also commercial modeling systems like Google SketchUp, where the user sketches freely to create a 3D building model from scratch or according to an image. The major limitation of these two systems is the large amount of user interaction involved. In Google SketchUp, all the shape details have to be sketched manually. As reported in Debevec's PhD thesis [1996], for the relatively simple Berkeley Campanile example (see Figure 13), about one hundred edges need to be manually marked and corresponded which takes 2 hours. Our method involves much less interaction, because the 3D information is explicitly recovered *before* the interactive façade decomposition and reconstruction. With our system, the user draws less than 20 lines and it takes only 9 minutes (for a novice) to model the Berkeley Campanile building. Another limitation of the Façade system is that it requires the camera to be pre-calibrated with known intrinsic parameters. To handle uncalibrated cameras, the vanishing points of three mutually orthogonal directions need to be detected from the image [Debevec 1996], which is often impossible (e.g. for buildings in Figure 10–Figure 12) and numerical unstable [Wilczkowiak et al. 2005]. In comparison, our novel auto-calibration algorithm is more robust and can handle more general data, which is a critical feature for a desktop modeling toolkit.

Single image based architecture modeling: Images provide very useful information to assist modeling. Even a single image can guide the modeling quite effectively. Hoiem et al. [2005] obtained a rough 3D model by recognizing 'ground', 'sky' and 'vertical' objects in the image. Liebowitz et al. [1996] created a 3D model by exploiting parallelism and orthogonality from a single image. Oh et al. [2001] manually assigned a depth with a painting interface to create 3D model. Such a procedure is tedious and labor intensive. Müller et al. [2007] derived shape grammars from a single image of a façade plane. These single image based methods are limited to simple buildings. While our method also takes a single image as input, we explicitly reconstruct 3D points from the input image, which helps both to simplify the user interaction and to model more complicated buildings.

Multiple images based architecture modeling: Multiple images from different viewpoints provide strong geometric constraints on 3D structure. Dick et al. [2004] built statistical model to infer building structure from multiple images. However, such inference is unreliable for complex buildings. The multi-view stereo algorithm developed in the computer vision community can generate cloud of 3D scene points from multiple images, which lead to more robust reconstruction. Sinha et al. [2008] used an unordered collection of pictures to assist interactive building reconstruction. Xiao et al. [2008] took pictures along streets and built 3D models of the whole street. Pollefeys et al. [2008] developed a real-time system for urban modeling from video data. Our method is inspired by the work of Sinha et al. [2008] and Xiao et al. [2008], where reconstructed 3D points can be used to guide user for efficient interaction. Specifically, if 3D points are reconstructed, tedious manual correspondence as in [Debevec et al. 1996] can be avoided. The user only needs to mark out structural components, whose shape and position can then be determined from the reconstructed 3D points. A disadvantage of these multiple image based methods is their need for multiple images of the same building as input, which is not always available. In contrast, our method requires only a single image.

Aerial images based architecture modeling: There are also methods [Zebedin et al. 2006] which used aerial images to reconstruct buildings. Some of them [Früh and Zakhor 2003] combine aerial images with ground-level images for the modeling. The focus of these methods is on how to efficiently model very large set of data. As such, the quality of each individual building could be sacrificed for modeling efficiency. In comparison, our focus is on how to create a high quality model for a single building.

Our method combines the strength of both interactive modeling and image based modeling. We take a single image as input and reconstruct explicit 3D information by leveraging on prevalent architectural symmetry. The reconstructed 3D information helps us to design a more efficient interface than previous interactive methods and single image based methods. Compared with those methods with multiple images, our system is more flexible since it requires much less data.

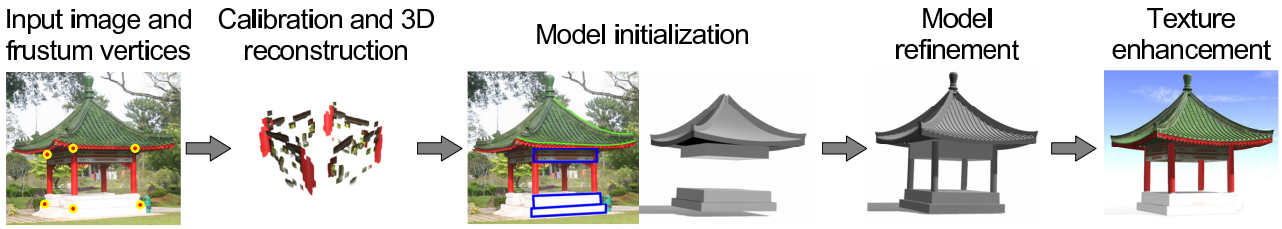

Figure 2: The modeling pipeline. We first calibrate the camera according to the user specified frustum vertices and reconstruct a set of 3D points. The architecture components (i.e. walls and roofs) are then interactively decomposed and modeled. Shape details can be added if necessary. Lastly, the final model is textured with our texture enhancement technique.

3 3D Reconstruction by Symmetry

In this section, we reconstruct the camera pose and a set of 3D points from a single image by exploiting architectural symmetries, including both bilateral and rotational symmetry. We first calibrate the camera from an observed pyramid frustum. Then we duplicate a virtual camera according to the calibration and the observed symmetry. 3D points are computed by a stereo algorithm from the real and virtual cameras.

3.1 Symmetry-based calibration

Cameras need to be calibrated for undistorted 3D reconstruction. The calibration accuracy is important as the image is related to the 3D model according to the calibration. 3D reconstruction is simplified when the camera is pre-calibrated offline as in [Debevec et al. 1996]. However, the requirement of pre-calibration also limits the images can be processed. We propose a novel auto-calibration algorithm to give our system the flexibility of working on images with completely unknown camera information, e.g. internet downloaded pictures and historical pictures.

Camera can be calibrated from the vanishing points of three mutually orthogonal directions in a single image [Hartley and Zisserman 2001], which is applied for façade modeling in [Debevec 1996]. However, many images, e.g. Figures 10– 12, do not have three such vanishing points. Furthermore, the vanishing point based approach is often numerically unstable [Wilczkowiak et al. 2005]. Naturally embedding the constraints from three vanishing points, an parallelepiped in a single image can be used to calibrate the camera [Wilczkowiak et al. 2001; Wilczkowiak et al. 2005]. This approach is stable and accurate and is applied for architecture modeling. Parallelepiped, however, is not the most suitable geometric primitive for architecture. A degree of freedom is redundant for architecture since the horizontal shearing of a parallelepiped is not present in real buildings. On the other hand, the horizontal size of real buildings often gradually shrink when the height increase. This feature is common in architectures, as illustrated in Figure 1 and Figures 10–12, but it cannot be represented by a parallelepiped. A better geometric primitive is the pyramid frustum, which does not introduce the redundant degree of freedom and can model real buildings well.

A pyramidal frustum is a truncated pyramid as illustrated in Figure 3. Here, we use a frustum with a rectangular base as an example for discussion, though our results are valid for frustums with different bases. We parameterize a pyramid frustum by $\alpha, \theta, l_1, l_2, l_3$, as illustrated in Figure 3. α is ≤ 1 and controls the shrinking of the pyramid. If $\alpha = 1$, the pyramid frustum degenerates to a right prism, a parallelepiped with zero horizontal shearing. θ is the angle between the two adjacent horizontal edges of the frustum base. $l_i, 1 \leq i \leq 3$, are the three independent lengths of the structure. For modeling applications, the absolute position and size of the structure is not important. Hence, without loss of generality, we can let the height $l_3 = 1$ and consider the origin of the world coordinate system to be at the bottom face of the frustum, with the z-axis passing through the apex of the pyramid, and the y-axis parallel to one of the edges. From a single image of a building, part of a pyramid frustum can often be seen (the highlighted vertices and edges in Figure 3). The corresponding points are highlighted in the Figure 1 (a).

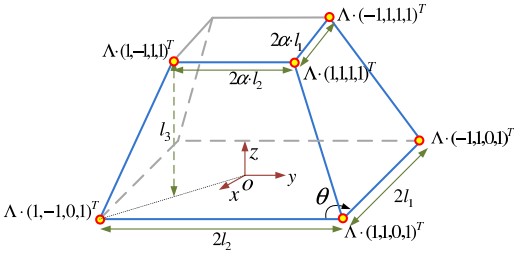

Figure 3: A pyramid frustum is a truncated pyramid. It shape is defined by 5 parameters. $l_i, 1 \leq i \leq 3$ defines the length of its edges. α controls the shrinking in vertical direction. θ is the angle between the two horizontal edges. Blue edges and red vertices are the parts of a frustum that are often visible in architecture images.

The homogeneous coordinates of a frustum vertex can be represented as $\mathbf{P}_i = \Lambda \cdot \hat{\mathbf{P}}_i^\top$, where $\hat{\mathbf{P}}_i = (x_i, y_i, z_i, 1)$, and $x_i, y_i \in \{1, -1\}, z_i \in \{0, 1\}$ (see Figure 3). Here,

$$\Lambda = \begin{pmatrix} l_1 & l_2 c & 0 & 0 \\ 0 & l_2 s & 0 & 0 \\ 0 & 0 & \beta l_3 & 0 \\ 0 & 0 & \beta - 1 & 1 \end{pmatrix},$$

where $\beta = 1/\alpha, s = \sin\theta$ and $c = \cos\theta$. As Λ contains all the shape parameters of the pyramid frustum, the 3D reconstruction of the frustum amounts to the estimation of Λ. Frustum vertices are projected into image coordinate $\mathbf{p}_i = (u_i, v_i, w_i)$ by the projective transformation $\mathbf{M}$, i.e. $\mathbf{p}_i \simeq \mathbf{M}\mathbf{P}_i = \mathbf{M}\Lambda\hat{\mathbf{P}}_i = \hat{\mathbf{M}}\hat{\mathbf{P}}_i$, where $\simeq$ means equality up to a scale. $\mathbf{M} = \mathbf{K} \cdot [\mathbf{R}|\mathbf{t}]$ is the 3×4 camera matrix, where $\mathbf{K}$ encodes the camera intrinsic parameters, $\mathbf{R}$ and $\mathbf{t}$ represent relative rotation and translation between the camera and the world coordinate system. If six or more frustum vertices can be observed from the image, $\hat{\mathbf{M}}$ can be computed by a linear algorithm [Hartley and Zisserman 2001]. Camera calibration and 3D reconstruction of the pyramid frustum then amounts to the factorization of $\hat{\mathbf{M}}$ as

$$\hat{\mathbf{M}} = \mathbf{K} \cdot [\mathbf{R}|\mathbf{t}] \cdot \Lambda.$$

A general camera intrinsic matrix $\mathbf{K}$ contains 5 unknowns. $\mathbf{R}, \mathbf{t}$ each contains 3 unknowns. Λ has another 4 unknowns (considering $l_3 = 1$), making a total of 15 unknowns. The 12 components of the 3×4 projective matrix $\hat{\mathbf{M}}$ provide only 11 independent constraints. This factorization is impossible without further assumption about the camera parameters and the scene structure. The assumption involves the tradeoff between the generality of the camera model and the frustum structures. To model a larger variety of buildings, we assume the simplest camera model where only the focal length is unknown[1]. With this simplification, all the 11 unknowns can be

[1]The other known camera parameters are the principal point, the pixel aspect ratio, and the camera skew.

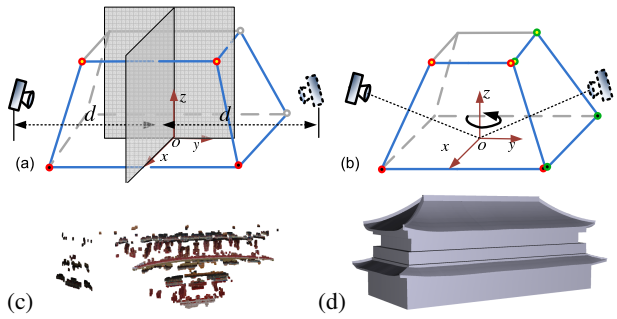

Figure 4: Representing architecture symmetry by pyramid frustum. (a) Bilateral symmetry is characterized by the symmetry plane, i.e. x-z plane. (b) Rotational symmetry is characterized by the rotation axis, i.e. z-axis. With the calibration of the real camera, a virtual camera can be duplicated according to the underlying symmetry. (c) Stereo algorithms can be applied to the real and virtual camera pair to recover a set of 3D points. (d) With a few strokes to delineate the key parts, the user can build an initial model from these 3D points.

computed from the 11 constraints with a general non-linear optimization method. If further information is known about the architecture structure as a prior, such as the value of the angle θ or the length ratio between l_1 and l_2, we can handle more general camera matrix with unknown pixel aspect ratio or principal point.

Quadratic initialization A good initialization is critical for the success of the above non-linear optimization. In this subsection we describe a method to initialize the estimation by solving a quadratic equation. We observe that

$$\hat{\mathbf{M}}^\top \mathbf{K}^{-\top} \cdot \mathbf{K}^{-1} \hat{\mathbf{M}} = \begin{pmatrix} l_1^2 & l_1 l_2 c & * & * \\ l_1 l_2 c & l_2^2 & * & * \\ * & * & * & * \\ * & * & * & * \end{pmatrix}.$$

Here, $\mathbf{K}^{-\top} \cdot \mathbf{K}^{-1} = \omega$ is the matrix representing the image of the absolute conic (IAC). Hence, we have the following equations,

$$\hat{\mathbf{m}}_1^\top \omega \hat{\mathbf{m}}_1 = l_1^2; \qquad \hat{\mathbf{m}}_1^\top \omega \hat{\mathbf{m}}_2 = l_1 l_2 c; \qquad \hat{\mathbf{m}}_2^\top \omega \hat{\mathbf{m}}_2 = l_2^2. \quad (1)$$

Here, $\hat{\mathbf{m}}_1, \hat{\mathbf{m}}_2$ are the first two columns of $\hat{\mathbf{M}}$. Assuming the simplest camera model, ω depends only on the focal length f. Equation (1) provides 3 equations for 4 unknowns l_1, l_2, θ, f. From a single image, very often we can either tell the value of θ or the length ratio of l_1 and l_2, which reduces one unknowns from Equation (1) and enables the recovery of the other threes. This provides the initialization of l_1, l_2, θ, and f. Next, we initialize the other unknowns, i.e. $\mathbf{R}, \mathbf{t}$, and β.

Once f is determined, $\mathbf{K}$ is known and we can compute

$$\begin{aligned} [\mathbf{R}|\mathbf{t}]\Lambda &= \mathbf{K}^{-1}\hat{\mathbf{M}} = [\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3, \mathbf{t}]\Lambda \\ &= [l_1\mathbf{r}_1, l_2c_2\mathbf{r}_1 + l_2s_2\mathbf{r}_2, \beta\mathbf{r}_3 + (\beta-1)\mathbf{t}, \mathbf{t}]. \end{aligned}$$

Here, $\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3$ are the three columns of $\mathbf{R}$. Hence, $\mathbf{r}_1$ can be obtained by normalizing the first column of $\mathbf{K}^{-1}\hat{\mathbf{M}}$. $\mathbf{r}_2$ is generated by projecting the second column to the plane perpendicular to $\mathbf{r}_1$, and $\mathbf{r}_3$ is simply $\mathbf{r}_1 \times \mathbf{r}_2$. $\mathbf{t}$ is the last column of $\mathbf{K}^{-1}\hat{\mathbf{M}}$. β can be obtained from the magnitude of the third column. This gives a complete initialization to the camera calibration and frustum reconstruction procedure.

3.2 Symmetry-based triangulation

Many architectures exhibit symmetry. The two commonest symmetries are bilateral symmetry and rotational symmetry. Both of them can be represented by the pyramid frustum as illustrated in Figure 4 (a) and (b) respectively. Bilateral symmetry is characterized by the symmetry plane, i.e. the x-z plane. (Some buildings exhibit further symmetry across the y-z plane.) Rotational symmetry is characterized by the rotation axis, i.e. the z-axis. Once the frustum shape, i.e. Λ, is computed, the type of symmetry can then be automatically determined from the frustum shape.

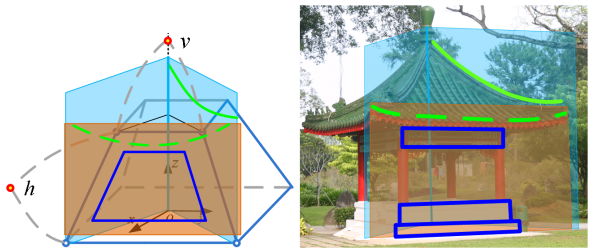

Figure 5: The user interactively draw a few strokes to build an initial model. The blue quadrilateral is the user-drawn wall structure. Green strokes are the user-drawn roof boundaries.

With the camera calibration, we can virtually duplicate another camera according to the underlying symmetry of the building. The virtual camera is generated by flipping the real camera across the x-z plane in the case of bilateral symmetry, or by rotating the real camera around the z-axis for an angle $\pi - \theta$ in the case of rotational symmetry. The observed image from the virtual camera can be derived from the input image. It is the input image with a horizontal flipping in the case of bilateral symmetry, or is exactly the input image in the case of rotational symmetry [Hong et al. 2004]. Some previous works [Zhang and Tsui 1998; Francois et al. 2002] demonstrate that stereo algorithms can be applied to the real and virtual camera pair with manually specified correspondences.

In practice, we find that the recent progress in feature detection and matching [Lowe 2004] has made the automatic stereo algorithm feasible on the real and virtual image pair without manually established correspondences. To facilitate matching, we take the frustum's side face as an initial estimation of the building façade, which induces a homography between the real and virtual image [Hartley and Zisserman 2001]. We only consider matches consistent with this homography. To reduce projective distortion, the image region enclosed in the frustum side face is further 'rectified' by mapping it to a rectangle. Image features are computed in this 'rectified' rectangle. Obtained matches are further propagated according to the method described in [Lhuillier and Quan 2002]. Then all matched features are reconstructed by triangulation [Hartley and Zisserman 2001], which generates a set of 3D points on the building façade. An example of this reconstruction is shown in Figure 4 (c). More examples of 3D reconstructions are included in the supplementary video.

4 Surface Modeling

After the symmetry-based triangulation, we obtain a set of 3D points on the building façade. Then the user interactively builds a surface model according to these points and the image information. The user first marks out large architectural components, such as walls and roofs, with a few strokes. The shapes of these components are automatically determined from the recovered 3D information. If further shape details are required, the user can also add roof tiles and handrails by a few additional strokes. The whole model is then textured according to the input image. In the input image, part of the model is viewed from slanted angle, which causes texture distortion. We use texture on fronto-parallel faces to correct this distortion. Texture on the invisible surfaces are synthesized by taking the weathering pattern into consideration.

4.1 Geometry modeling

Model initialization Multi-view stereo automatically reconstructs a set of 3D points. User interaction often follows to build

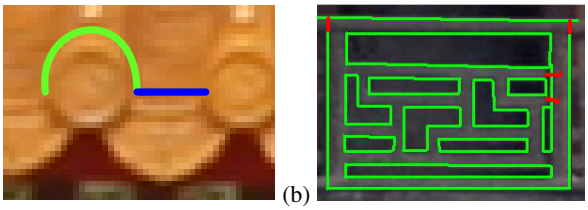

Figure 6: Model refinement. (a) The user marks out the tile shape from a rectified view of the roof's front edge. This tile is applied to all roofs. (b) Cut pattern is automatically extracted. The user can also refine some incorrect line segments as illustrated in red.

surface model according to these points. This leveraging of automatic vision technique together with user interaction is employed in several previous systems, such as [van den Hengel et al. 2007; Sinha et al. 2008; Xiao et al. 2008]. Similarly, we interactively identify architectural components such as walls and roofs from the image plane, and then compute their 3D shape and position according to the reconstructed 3D information. Müller et al. [2007] and Xiao et al. [2008] propose an automatic method to partition building façades into rectangular components. However, these methods cannot handle our complex façade data such as those shown in Figure 1 and Figure 11. Thus, we rely on user interaction to mark out architectural structures and leave the automatic partition for future research. We mark out two kinds of structures, namely planar walls and curved roofs.

The user interactively marks out a planar structure in the image. Then its position is determined according to the enclosed 3D points. Its symmetric counterparts are also automatically generated. Each planar structure should pass through two additional points, which are the intersection of the two pairs of edges of the frustum side face. In the left of Figure 5, these two points are illustrated as h and v. h is the vanishing point of the horizontal direction. v can be a finite or infinite point, depending on the parameter α of the frustum. Similar plane fitting approach is used in [Sinha et al. 2008]. These vanishing points serve the purpose of maintaining architecture shape regularities, such as the angle between two adjacent wall planes. Of course the user can also choose to discard these constraints to have larger modeling flexibility. In Figure 5, the blue quadrilateral is the user-drawn planar structure. With the two constraint points h, v, one reconstructed 3D point in the enclosed region can uniquely determine a planar structure. If multiple points are available, we apply RANSAC to obtain a robust fitting.

Generally, there are not enough reconstructed 3D points on the roof (see Figure 4 (c)) to determine its shape and position. The user needs to mark out roof boundaries in 3D to model the roof. To simplify the user interaction, we introduce a set of auxiliary planes. The user draws 2D curves within these planes to decide 3D roof boundaries. Once the roof modeling function is enabled, multiple blue auxiliary planes are overlaid on the input image as illustrated in Figure 5. [2] The user marks out the *back edge* of the roof, i.e. the solid green curve. Its 3D position is determined by projecting the drawn curve to the auxiliary plane according to the calibrated pinhole camera. All the symmetric counterparts of this *back edge* are generated according to the symmetry automatically. The roof hip, the beam along the *back edge*, is modeled by raising the 3D edge for a constant distance. Then the user draws the *front edge*, i.e. the dashed green curve in Figure 5. Similarly, its 3D position is obtained by projecting the drawn curve to the brown auxiliary plane, which is parallel to the z-axis and passing through the end points of the *back edge*. The curved roof is interpolated according to these surrounding edges. More details of the user interaction

[2]The auxiliary planes of bilaterally symmetric buildings are illustrated in the supplementary video.

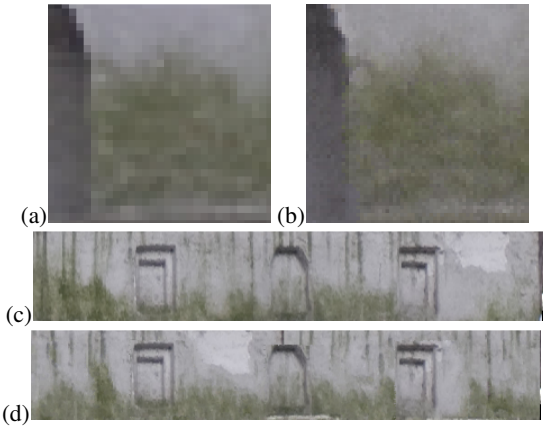

Figure 7: Examples of texture enhancement. The texture in (a) has low resolution because of the foreshortening in the input image. This texture is enhanced to (b) to reduce artifacts. Texture (d) is synthesized according to the sample from texture (c). Similar weathering pattern is maintained.

can be found in the supplementary video. With these strokes to mark out walls and roofs, an initial 3D model of the building can be obtained as shown in Figure 4 (d). Corresponding strokes are shown in Figure 1 (a).

Model refinement Many architectures contain intricate geometric ornaments, which are hard to reconstruct from stereo triangulation. We describe an efficient way to model these details. We deal with two types of shape detail here, roof tiles and carved handrails. In addition, we also have predefined geometric primitives such as spheres, pillars and steps, which can be directly inserted into the building model. The user can also model revolved surface by specifying the revolving boundary.

From the initial model, we extract a rectified view of the roof *front edge* as shown in Figure 6 (a). The user marks out one tile in this rectified view. The tile's shape and interval are determined according to these two strokes. The number of tiles is calculated by dividing the whole *front edge* length by the tile's size and interval. Then the tiling is applied to the whole roof surface. Each tile is textured by a predefined generic texture. Handrails have intricate cut patterns. We also extract a rectified view of the handrail as shown in Figure 6 (b). We apply Canny edge detection to extract edge pixels, which are then traced to form segments. We discard short segments and constrain remaining ones to have a few predefined directions, such as vertical or horizontal. In Figure 6 (b), the green edges are automatically detected, while the red strokes are user-input to refine incorrect edges. The user can either add or delete edges. The 3D shape of the handrail is created from the 2D pattern with a constant depth.

4.2 Texture enhancement

One inevitable problem in single image modeling is the lack of texture samples. Some parts of the building are viewed from a slanted view angle or occluded in the input image. Texturing by back-projecting the image to the 3D model will cause large texture distortion. This distortion is systematically studied in [Tai et al. 2008], where the texture map is segmented and synthesized with consideration of orientation and scale changes. We also apply synthesis techniques to improve the texture quality with two novel features. First, we require the final texture to be consistent with the foreshortened image, which contains partial information of the underlying texture. Second, we require the synthesized texture to have consistent weathering patterns.

We enhance texture maps by applying patch based synthesis [Kwa-

Figure 8: A pavilion example. On the left is the single input image. On the right is the recovered model rendered from the same viewpoint as the input image.

tra et al. 2003]. We first generate an initial texture map by back projection. This texture map is marked automatically as regions free of distortion, with distortion, or occluded, by thresholding the ratio between the size of the mesh triangle and that of its image projection. Larger ratio indicates larger texture distortion. Texture in the distortion free regions is used as samples to enhance that of other regions. We treat the enhancement of foreshortened region as a 'super-resolution' problem [Freeman et al. 2000]. This enhancement runs by iterations. At each iteration, we overwrite a foreshortened texture patch by a distortion free patch that is most similar (in the sense of SSD) to it. This new patch is stitched to the texture map by a graph-cut optimization as in [Kwatra et al. 2003]. A result of this super-resolution is shown in Figure 7. The texture in (a) has limited resolution due to foreshortening in the input image. Our method can enhance it to one with a similar resolution to the fronto-parallel view (Figure 7(b)).

The simplest way to texture the occluded regions is to repeat the same texture as those of their symmetric counterparts. However, this simple repeating makes the model look artificial. Instead, we synthesize texture in the occluded regions according to those texture found in the distortion free region. Another difficulty lies in maintaining consistent weathering patterns. Architecture surfaces often have strong weathering patterns as exemplified in Figure 10. Wang et al. [Wang et al. 2006] propose to extract the surface appearance manifold and weathering degree map from image samples to generate physically correct result. Here, we seek a simple solution that yields plausible results. We apply similar idea to [Ramanarayanan and Bala 2007], where a constraint map is used to guild texture synthesis. We observe that the weathering degree on buildings is generally inversely proportional to the height. Hence, we take the height as indicative of the weathering degree, which is used to guide the patch-based synthesis to generate the missing texture. We iteratively copy a patch from a distortion free region with consistent boundary and similar weathering degree to the occluded surface. The copied patch is also stitched with graph-cut optimization. To maintain semantic texture structures, such as doors and windows, we first copy them from the symmetric counterpart and keep them uncovered during synthesis. An example of this synthesis is shown in Figure 7 where (c) depicts the distortion free texture, and (d) depicts the texture synthesized according to (c).

5 Results

We first evaluate our symmetry based 3D reconstruction with a synthetic frustum image. We manually mark out 6 visible frustum vertices to reconstruct the frustum shape. In our experiments, the user clicks often deviate from the true vertex position by 1.3 pixels (in an image of resolution of 1200x800). The error of focal length is 4.3% of the true value, and the mean error of vertex position is 0.1% of the distance between the camera and pyramid frustum center. The reconstructed frustum (shown in red wireframe) is verified from a novel viewpoint as shown in the right of Figure 9. Then we verified the accuracy of estimated focal length in real photo of Figure 8. The true focal length calculated from photos of a checkerboard is 2864.8 pixels (50 mm). In 10 trials, our estimated focal length

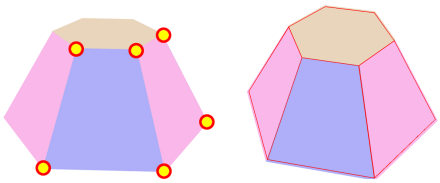

Figure 9: Validation of symmetry based reconstruction on synthetic data. On the left is the input image (overlaid with user clicked 6 vertices). On the right is a rendering from a novel viewpoint. The reconstructed shape is overlayed on the image (drawn as red wireframe).

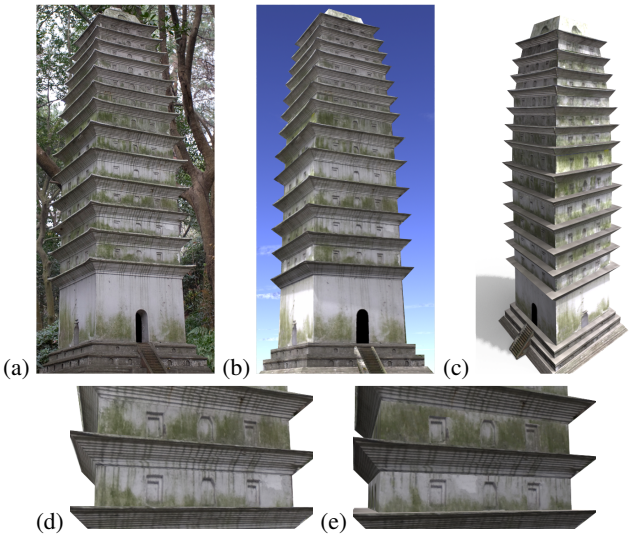

Figure 10: A pagoda example. (a) is the single input image. (b) is the recovered model rendered from the same viewpoint as the input image. (c) is the rendering from a novel viewpoint. (d) and (e) are two different façades at the same height. The façade in (d) is textured from the input image; the texture in (e) is synthesized by our method. Our texture synthesis generates more vivid texture than simple repetition.

varies from 2582.4 pixels to 3069.8 pixels. In comparison, we also implemented the calibration method discussed in [Debevec 1996] which generates results vary from 2029.4 pixels to 3968.8 pixels.

We test our method on several examples with different level of complexity. Our symmetry-based triangulation takes 4-5 minutes on a PC with 2.83GHz CPU and 4GB memory. With the recovered 3D points, the user draws strokes to build an initial model (strokes for each example are provided in the supplementary video). We measure the user effort by the interaction time, because the other parts are automatic. It takes 2 – 10 minutes user interaction to generate a result. We report the user interaction time and the number of reconstructed 3D points for each example in the Table 1. [3]

As no ground truth 3D model is available, here we only evaluate the results visually against the input image. Figure 1 shows an example with bilateral symmetry (the TiRen Ge in the Forbidden City). Figure 1 (a) shows the user-drawn strokes. The highlighted six points are as the corners of the pyramid frustum. Feature matching is most effective on this example, with 12,000 3D points recovered as shown in Figure 4 (c). Figure 1 (b) shows the rendering of the recovered model from the same viewpoint as the input im-

[3]This interaction time is measured for an experienced user. We also let a novice try our system. After watching a 10 minutes instruction of the system (with the pavilion example), he spends 9 minutes in total to model the Berkeley Campanile (5 minutes for the automatic triangulation and 4 minutes for user interaction; no texture synthesis is applied for this example).

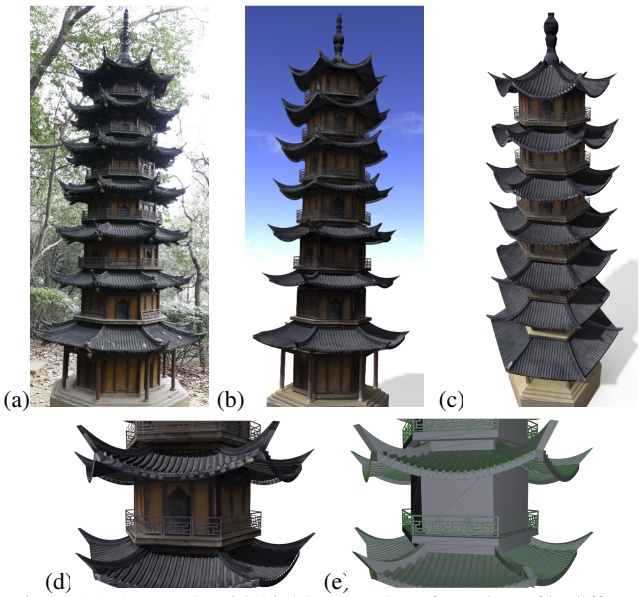

Figure 11: A pagoda with highly curved roof. Each roof is different from the others. (a) is the single input image. (b) is the recovered model rendered from the same view as the input image. (c) is the rendering from a novel viewpoint. (d) shows a closeup view of the building. (e) is the shaded wireframe to highlight the geometry. (Please zoom in the electronic version.)

age. A novel viewpoint rendering is provided in Figure 1 (c) for validation. A simple pavilion example is shown in Figure 8, where (a) is the input image and (b) shows the model rendered from the same viewpoint for validation. A ridge on the top is missing, which is caused by the inaccuracy in the interactive modeling. The roof hip is a little higher than it should be, and thus occludes part of the second ridge. Figure 10 shows an example with multiple floors. We first model the first floor and apply the modeled result to the other floors with only one stroke to compute a scaling and vertical translation (please refer to the supplementary video). This example highlights our texture enhancements. Figures 10 (d) and (e) show two different building façades at the same height; while the façade in (d) is textured according to the input image, the texture in (e) is synthesized using our method. Our synthesized texture produces consistent weathering pattern. Figure 11 shows a complex pagoda with rotational symmetry. It has highly curved roofs, which are different at each floor. We draw 3 strokes to model each roof (1 additional stroke for the *back edge* to model the shrinking of the hip). Furthermore, its handrail has intricate cut patterns. It takes about 10 minutes of user interaction to model this finely detailed example. Figure 11 (d) shows a closeup view. Further geometrical details are highlighted by the wireframe rendering in Figure 11 (e).

Our method is also applicable to the modeling of western buildings, as the various principles invoked in this paper are generally true for most architectural forms. The Eiffel Tower in Figure 12 and the Berkeley Campanile in Figure 13 show two western buildings modeled using our method. We model the curved surface of the Eiffel Tower according to its curved silhouettes. One of the cut patterns is modeled in the same way as the handrail. The Berkeley is the easiest as there is no curved roof and cut patterns. It is modeled with less than 2 minutes interaction.

6 Discussion

Architecture modeling has been an active research field for many years. Existing systems, such as [Debevec et al. 1996; Sinha et al. 2008; Xiao et al. 2008], create highly realistic results from multiple images. In comparison, we seek to provide an alternative solution for architecture modeling when only a single image is available. Hence, we do not require complicated geometric and photometric image alignments. Our method can be a standalone toolkit for artists to create 3D architecture models from online pictures or archive pictures. It can also be integrated into previous multi-view based systems to provide further shape constraints.

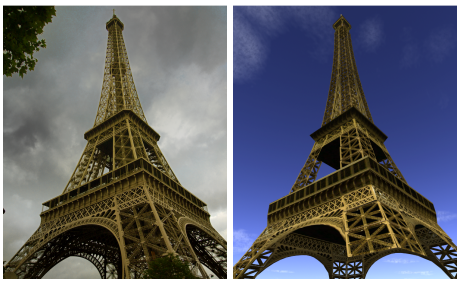

Figure 12: The Eiffel Tower example. On the left is the single input image. On the right is the recovered model rendered from the same viewpoint as the input image.

Figure 13: The Berkeley Campanile example. From left to right, they are: the single input image, the rendering from the same viewpoint as the input image and the rendering from a novel viewpoint.

Limitations The current method has several limitations. First, like most image based modeling systems, our method prefers the input images to be free of shadow effect. Otherwise, shadow could be mistaken as texture and cause artifacts in the rendered models. Second, in the case that no symmetry is present (though a rare case), we cannot model the building from a single image. If the building can be decomposed into multiple symmetric parts, we might still model it part by part. If multiple images are available, our interactive system for decomposing and modeling architecture components can still be applied. However, we cannot reduce the interactions by symmetry. In that case, our method will be a regular multi-view interactive modeling system like [Sinha et al. 2008]. Third, our camera calibration relies on the quadratic initialization, which requires the principle point to be close to the image center unless both the length ratios and the frustum angle in Equation (1) are known. Last, large lens distortions could be a problem, as our current algorithm for calibration does not consider it.

Examples:	TiRen Ge	pavilion	P1	P2	ET	BC
# 3D pts ($\times 10^3$):	12	5	7	11	0.5	0.7
IT: (mins)	5	2	2	10	5	2

Table 1: Modeling statistics. We show the number of reconstructed 3D points and the user interaction time for each example in the paper. Most of our examples require less than 5 minutes of user interaction. Note: IT = user interaction time, P1 = the pagoda in Figure 10, P2 = the pagoda in Figure 11, ET = the Eiffel Tower, BC = the Berkeley Campanile.

7 Conclusion and Future Work

We present an efficient method to build high quality architecture models from a single image by exploiting constraints derived from symmetry. To achieve this, a novel method is designed to calibrate the camera and recover 3D scene points from a single image. The recovered 3D information helps to reduce the amount of user interaction by avoiding tedious manual correspondence. We also enhance the texture quality to improve single view modeling.

There are several ways to improve our current system. For example, in the current system, the roof tile is manually marked out from the rectified view. This part can be automated by image analysis and shape template matching. Image processing techniques can also be applied to snap user strokes to image edges to make the interface more efficient.

8 Acknowledgement

The authors thank Li Na, Tian Fang, Andrew J Zuckerman and Nathan Hall for their permission to use their pictures in this paper. This work is supported by Singapore Grant R-263-000-477-112, R-705-000-018-279 and the Microsoft Research Asia eHeritage Theme grant.

References

DEBEVEC, P. E., TAYLOR, C. J., AND MALIK, J. 1996. Modeling and rendering architecture from photographs: a hybrid geometry and image-based approach. *ACM SIGGRAPH 1996*, 11–20.

DEBEVEC, P. E. 1996. *Modeling and Rendering Architecture from Photographs*. PhD thesis, University of California at Berkeley, Computer Science Division, Berkeley CA.

DICK, A. R., TORR, P. H. S., AND CIPOLLA, R. 2004. Modelling and interpretation of architecture from several images. *Int. J. Comput. Vision 60*, 2, 111–134.

FITZGIBBON, A. W., CROSS, G., AND ZISSERMAN, A. 1998. Automatic 3d model construction for turn-table sequences. In *Proc. of SMILE Workshop on Structure from Multiple Images in Large Scale Environments*, 155–170.

FRANCOIS, A., MEDIONI, G., AND WAUPOTITSCH, R. 2002. Reconstructing mirror symmetric scenes from a single view using 2-view stereo geometry. In *Proc. of ICPR 2002*.

FREEMAN, W., PASZTOR, E., AND CARMICHAEL, O. 2000. Learning low-level vision. *Int. J. of Comput. Vision 40*, 2000.

FRÜH, C., AND ZAKHOR, A. 2003. Constructing 3d city models by merging ground-based and airborne views. In *Proc. of CVPR*.

HARGITTAI, I., AND HARGITTAI, M. 1994. *Symmetry: A Unifying Concept*. Shelter Publications.

HARTLEY, R., AND ZISSERMAN, A. 2001. *Multiple View Geometry in Computer Vision*. Cambridge University Press.

HOIEM, D., EFROS, A. A., AND HEBERT, M. 2005. Automatic photo pop-up. *ACM SIGGRAPH 2005 24*, 3, 577–584.

HONG, W., YANG, A. Y., HUANG, K., AND MA, Y. 2004. On symmetry and multiple-view geometry: Structure, pose, and calibration from a single image. *Int. J. Comput. Vision 60*, 241–265.

KWATRA, V., SCHÖDL, A., ESSA, I., TURK, G., AND BOBICK, A. 2003. Graphcut textures: image and video synthesis using graph cuts. *ACM SIGGRAPH 2003 22*, 3, 277–286.

LHUILLIER, M., AND QUAN, L. 2002. Match propagation for image-based modeling and rendering. *IEEE Trans. Pattern Anal. Mach. Intell. 24*, 8, 1140–1146.

LIEBOWITZ, D., CRIMINISI, A., AND ZISSERMAN, A. 1996. Creating architectural models from images. *Computer Graphics Forum* (September), 39–50.

LOWE, D. G. 2004. Distinctive image features from scale-invariant keypoints. *Int. J. of Comput. Vision 60*, 2, 91–110.

MÜLLER, P., WONKA, P., HAEGLER, S., ULMER, A., AND VAN GOOL, L. 2006. Procedural modeling of buildings. *ACM Trans. Graph. (SIGGRAPH 2006) 25*, 3, 614–623.

MÜLLER, P., ZENG, G., WONKA, P., AND VAN GOOL, L. 2007. Image-based procedural modeling of facades. *ACM Trans. Graph. (SIGGRAPH 2007) 26*, 3, 85.

OH, B. M., CHEN, M., DORSEY, J., AND DURAND, F. 2001. Image-based modeling and photo editing. In *ACM SIGGRAPH 2001*.

PARISH, Y. I. H., AND MÜLLER, P. 2001. Procedural modeling of cities. In *ACM SIGGRAPH 2001*.

POLLEFEYS, M., AND ET AL. 2008. Detailed real-time urban 3d reconstruction from video. *Int. J. Comput. Vision 78*, 143–167.

RAMANARAYANAN, G., AND BALA, K. 2007. Constrainted texture synthesis via engergy minimization. *IEEE Trans. Visualization and Computer Graphics 13*, 1, 167–178.

ROTHWELL, C. A., FORSYTH, D. A., ZISSERMAN, A., AND MUNDY, J. L. 1993. Extracting projective structure from single perspective views of 3d point sets. In *Proc. of ICCV1993*.

SINHA, S. N., STEEDLY, D., SZELISKI, R., AGRAWALA, M., AND POLLEFEYS, M. 2008. Interactive 3d architectural modeling from unordered photo collections. *ACM Trans. Graph. (SIGGRAPH Asia 2008) 27*, 5, 1–10.

TAI, Y.-W., BROWN, M. S., TANG, C.-K., AND SHUM, H.-Y. 2008. Texture amendment: reducing texture distortion in constrained parameterization. *ACM Trans. Graph. (Siggraph Asia 2008) 27*, 5, 1–6.

VAN DEN HENGEL, A., DICK, A., THORMÄHLEN, T., WARD, B., AND TORR, P. H. S. 2007. Videotrace: rapid interactive scene modelling from video. 86.

WANG, J., TONG, X., LIN, S., PAN, M., WANG, C., BAO, H., GUO, B., AND SHUM, H.-Y. 2006. Appearance manifolds for modeling time-variant appearance of materials. *ACM Trans. on Graph. (SIGGRAPH 2006)*, 754–761.

WILCZKOWIAK, M., BOYER, E., AND STURM, P. 2001. Camera calibration and 3d reconstruction from single images using parallelepipeds. In *Proc. of ICCV*, 142–148.

WILCZKOWIAK, M., STURM, P., AND BOYER, E. 2005. Using geometric constraints through parallelepipeds for calibration and 3d modeling. *IEEE Trans. Pattern Anal. Mach. Intell. 27*, 2.

XIAO, J., FANG, T., TAN, P., ZHAO, P., OFEK, E., AND QUAN, L. 2008. Image-based façade modeling. *ACM Trans. Graph. (SIGGRAPH Asia 2008) 27*, 5, 1–10.

ZEBEDIN, L., KLAUS, A., GRUBER-GEYMAYER, B., AND KARNER, K. 2006. Towards 3d map generation from digital aerial images. *Journal of photogrammetry and remote sensing 60*, 6, 413–427.

ZHANG, Z. Y., AND TSUI, H. T. 1998. 3d reconstruction from a single view of an object and its image in a plane mirror. In *Proc. of ICPR 1998*.