# Automatic Pose Recovery for High-Quality Textures Generation

Jinhui Hu, Suya You, Ulrich Neumann
*University of Southern California*
*{jinhuihu, suyay,uneumann}@graphics.usc.edu*

## Abstract

*This paper proposes new techniques to generate high quality textures for urban building models by automatic camera calibration and pose recovery. The camera pose is decomposed into an orientation and a translation, an edge error model and knowledge-based filters are used to estimate correct vanishing points with heavy trees occlusion, and the vanishing points are used for the camera calibration and orientation estimation. We propose new techniques to estimate the camera orientation with infinite vanishing points and translation with under-constraints. The final textures are generated using color calibration and blending with the recovered pose. A number of textures for outdoor buildings are automatically generated, which shows the effectiveness of our algorithms.*

## 1. Introduction

High quality texture is a crucial element in today's vision and graphics applications. The generation of textures requires automatic camera calibration and pose recovery, which is a challenging task for outdoor building images. The camera orientation can be estimated using vanishing points [2], however, heavy trees occlusion of outdoor urban buildings (Figure 3(a)) causes line clustering and vanishing points extraction a difficult problem. Furthermore, given a camera's orientation, its translation recovery is often an under-constrained problem using a single image due to lack of features in rough building models (Figure 3(g)) and the narrow field of view of a camera (Figure 3 (c)). Lastly, it is very hard to capture an image that covers a whole building due to the small field of view a camera and the physical barriers of narrow streets. Multiple images are often necessary to generate textures for a whole building, which causes other problems such as illumination variation and parallax in different images. The goal of this paper is to generate high quality textures for given urban building models (especially rough models) by automatic camera calibration and pose recovery.

Previous work [4] fix the image senor with the range sensor to get the texture data, and the camera pose is recovered by a simple calibration relative to the range sensor. This technique has the advantage of simple calibration, but lack of flexibility. Stamos and Allen [10] use a freely moving camera to capture the image data. The camera pose is computed by fitting rectangular features from dense 3D range data, which is not applicable for rough urban models. Sunchun et al. [6] propose a system to register ground images to urban models using vanishing points to estimate the orientation and 2D to 3D feature correspondences to estimate the translation. The under-constrained translation problem is solved by manually infer more 3D points from registered images, which is not applicable when all images are under-constrained.

This paper presents new techniques to solve the challenges in generating textures for urban models. We decompose the camera's external matrix into orientation and translation. The orientation of the camera is estimated using vanishing points extracted by automatic line clustering, the translation is computed using 3D to 2D corner correspondences, and multiple images are used if one image does not provide enough correspondences. A global registration algorithm is used to refine the pose.

## 2. Pose Estimation

### 2.1. Orientation estimation

*Vanishing points extraction*

Vanishing points are used to estimate the camera's orientation due to lack of features in 3D urban models. Many methods have been presented using vanishing points for camera calibration and pose recovery [1,6,9]. We opt to use an automatic line clustering method in image space rather than Gaussian sphere because of several reasons. Gaussian sphere method is a global feature extraction method, which is sensitive to spurious maxima. The accuracy of Gaussian sphere method is limited to the discretizing accuracy, which is hard to achieve the precision that an image can offer.

Edges are detected using Canny edge detector and lines are extracted using Hough Transform as a preprocess step for vanishing points extraction. The intersections of all pairs of line segments are then computed to find all possible vanishing points for line grouping. We need a grouping criterion to assign lines to different clusters (vanishing points). Many methods [9] use a hard threshold of distance or angle, which is sensitive to noise. We use a method that is based on an edge error model without any hard thresholds.

The edge error model is as following. Consider the representation of a line segment using two endpoints, and assume the two end points have one pixel precision, then two fan regions with a rectangular region in the middle can be formed by moving the two end points freely in the pixel squares (Figure 1(left)). Since a true vanishing point cannot lie in the image line segment, the rectangular region has no effect on the intersection of edge regions. We simply take the middle point of the edge, and form two fan regions with the two end points. Furthermore, a true vanishing point can only lie in one direction of the edge, so we just take one of the fan region (Figure 1(middle)). Shufelt [9] uses a constant value of one pixel as the noise magnitude for the two end points, while we model the noise magnitude as $\varepsilon = c/\sqrt{l}$, where $l$ is the length of the edge, and c is set to 3.5 pixels. This model shows a better result than setting the noise magnitude as a constant value for all lines.

The grouping method is consistent with the edge error model. For each cluster of two lines, we find the intersection region $A$ of the edges, and a test edge is assigned to this cluster when its edge region overlaps with region $A$. Furthermore, we use a strong clustering constraint. A test edge is assigned to a cluster only if its edge region covers the intersection point of the cluster (Figure 1(right)). This guarantees the intersection region of the edge regions in each cluster is not empty, thus the vanishing point is not empty. The normalized length of each edge is accumulated in its assigned cluster, and the maximum clusters are chosen to compute potential vanishing points using the vanishing hull theory [1].

*Filtering spurious vanishing points*

Most of our testing images are outdoor building images with heavy occlusion by trees (Figure 3(a)), which causes many spurious vanishing points. Knowledge of the image and vanishing points are used to filter spurious vanishing points. We first roughly classify the extracted lines into $x$ and $y$ groups according to the line orientation, then filter vanishing points using the following three filters.

1) Line length. According to the edge error model, longer lines are more reliable, however, we would like also to keep shorter lines. So we first filter the lines with a large length threshold, then estimate the possible vanishing points, and these points are used to find more line supporters according to the grouping method.

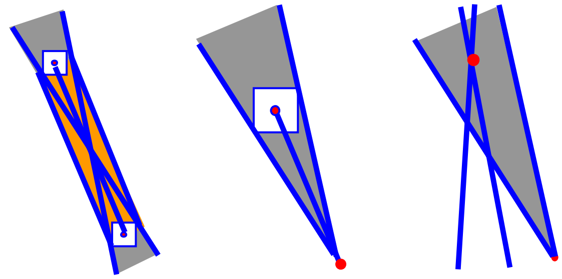

**Figure 1. The edge error model and grouping method.**

2) Covering area. Another observation of the image is that edges of trees only cover a small part of the image region, so the ratio of the covering area against the image area is also used to filter spurious vanishing points.

3) Valid vanishing point. Vanishing points are the intersection of image lines that correspond to parallel lines in 3D, so a valid vanishing point will not lie on the image segment in the image space. This filter is very effective in reducing spurious clusters.

Finding clusters and grouping lines using all pairs of line segments is computational expensive. Even though we classify lines into two directions to reduce the line number and use filters to reject spurious clusters, the number of clusters may still be large. The RANSAC algorithm is used to find the maximum cluster of lines for $x$-$y$ direction rather than testing each pair of line segments. The vanishing point of $z$ direction is estimated using the orthogonal property of the three directions, and supporting lines are found using our grouping method to refine the position of the vanishing point.

*Rotation estimation*

The rotation matrix and camera's focal length and principle point can be estimated given three orthogonal vanishing points. Cipolla et al. [2] derive the following equations:

$$\begin{bmatrix} \lambda_1 u_1 & \lambda_2 u_2 & \lambda_3 u_3 \\ \lambda_1 v_1 & \lambda_2 v_2 & \lambda_3 v_3 \\ \lambda_1 & \lambda_2 & \lambda_3 \end{bmatrix} = \begin{bmatrix} \alpha & 0 & u_0 \\ 0 & \alpha & v_0 \\ 0 & 0 & 1 \end{bmatrix} R \tag{1}$$

$$R = \begin{bmatrix} \lambda_1(u_1 - u_0)/\alpha & \lambda_2(u_2 - u_0)/\alpha & \lambda_3(u_3 - u_0)/\alpha \\ \lambda_1(v_1 - v_0)/\alpha & \lambda_2(v_2 - v_0)/\alpha & \lambda_3(v_3 - v_0)/\alpha \\ \lambda_1 & \lambda_2 & \lambda_3 \end{bmatrix} \tag{2}$$

Where $\lambda_1$, $\lambda_2$ and $\lambda_3$ are scale factors, $(u_0, v_0)$ is the principle point, $(u_i, v_i)$ $i = 1,2,3$ are the three orthogonal vanishing points, and $\alpha$ is the focal length. When only two vanishing points are available, the third vanishing point can be inferred by assuming that the principle point is at the camera center [6].

Equation (2) gives the solution of the rotation matrix for finite vanishing points, however, when the vanishing points are at infinity (lines are parallel in the image space), the solution becomes unclear.

We derive the equations to compute the rotation matrix for infinite vanishing points using Euler angles. A rotation matrix can be represented using three Euler angles (We ignore singularity cases of 90 degrees, which can be treated specifically).

$$R = \begin{bmatrix} cycz & -cysz & sy \\ sxsycz + cxsz & -sxsysz + cxcz & -sxcy \\ -sycxcz + sxsz & sycxsz + sxcz & cxcy \end{bmatrix} \quad (3)$$

Where *x*, *y*, z are the three Euler angles, *cx* stands for *cosx*, and *sx* stands for *sinx*. We classify the situation into two cases according to the number of infinite vanishing points, and derive the solutions respectively.

1. One infinite vanishing point, two finite vanishing points. Suppose the *x* direction vanishing point is at infinite, thus the *y* direction rotation is zero, so:

$$R = \begin{bmatrix} cz & -sz & 0 \\ cxsz & cxcz & -sx \\ sxsz & sxcz & cx \end{bmatrix} \quad (4)$$

Substitute Equation (4) into Equation (1), we have:

$$A = \begin{bmatrix} \lambda_1 u_1 & \lambda_2 u_2 & \lambda_3 u_3 \\ \lambda_1 v_1 & \lambda_2 v_2 & \lambda_3 v_3 \\ \lambda_1 & \lambda_2 & \lambda_3 \end{bmatrix} = \begin{bmatrix} \alpha & 0 & u_0 \\ 0 & \alpha & v_0 \\ 0 & 0 & 1 \end{bmatrix} R$$
$$= \begin{bmatrix} \alpha cz + u_0 sxsz & -\alpha sz + u_0 sxcz & u_0 cx \\ \alpha cxsz + v_0 sxsz & \alpha cxcz + v_0 sxcz & -\alpha sx + v_0 cx \\ sxsz & sxcz & cx \end{bmatrix} \quad (5)$$

Assume $cz \neq 0$ (the rotation around z axis is not 90 degrees, otherwise we have just two variables of *x* and $\alpha$, which is trivial to solve), from certain operations on both sides, we have:

$$\begin{aligned} A_{11} / A_{21} &= u_1 / v_1 = (\alpha ctgz + u_0 sx) / (\alpha cx + v_0 sx) \\ A_{12} / A_{32} &= u_2 = (-\alpha tgz + u_0 sx) / sx \\ A_{22} / A_{32} &= v_2 = \alpha ctgx + v_0 \end{aligned} \quad (6)$$

Without loss of generality, we assume the principle points are at the image center, and $u_0 = 0, v_0 = 0$. Let $k_1 = u_1 / v_1$ (although $u_1$ $v_1$ are at infinity, the slope of the image lines is still finite since *z* rotation is not 90 degrees), $k_2 = u_2 / v_2$, it is easy to find the solution of Equation (6) as:

$$\begin{aligned} x &= \cos^{-1}[-1/(k_1 k_2)]^{1/2} \\ \alpha &= (v_2 - v_0) tgx \\ z &= tg^{-1}[(u_0 sx - u_2 sx)/\alpha] \end{aligned} \quad (7)$$

We can derive the solution in a similar way when the *y* or *z* direction vanishing point is at infinite.

2. Two infinite vanishing points, one finite vanishing points. Suppose *x*, *y* direction vanishing points are at infinite, then the *x*, *y* direction rotation is zero, so we have:

$$\begin{bmatrix} \lambda_1 u_1 & \lambda_2 u_2 & \lambda_3 u_3 \\ \lambda_1 v_1 & \lambda_2 v_2 & \lambda_3 v_3 \\ \lambda_1 & \lambda_2 & \lambda_3 \end{bmatrix} = \begin{bmatrix} \alpha cz & -\alpha sz & 0 \\ \alpha sz & \alpha cz & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (8)$$

Thus, $z = ctg^{-1}(u_1 / v_1)$, where $u_1 / v_1$ is the slope of image lines that corresponds to the *x* direction vanishing point. The focal length cannot be recovered in this case. The other tow cases when the *z*-*y* or *x*-*y* direction vanishing points are at infinite can be derived in a similar way.

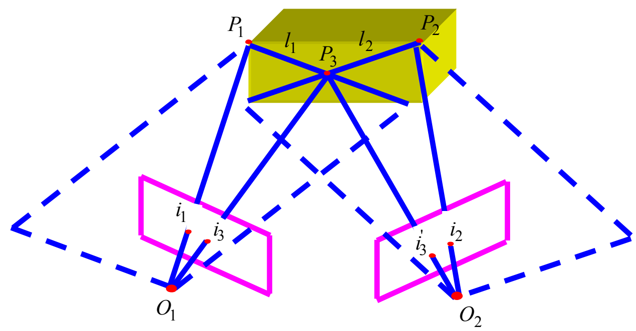


**Figure 2. Translation estimation.**

## 2.2. Translation estimation

Given the orientation of the camera, the translation relative to the 3D models can be computed using two 2D to 3D point correspondences [6]. However, due to the physical limitations (such as a camera's small field of view and narrow streets), sometimes only one 3D corner or none corners are visible for a single image (Figure 3(c)), which makes the translation estimation problem an under-constrained problem.

Multiple images are used to compute the camera's translation when each single image does not provide enough constraints. Let's first consider two images, each of which covers only one 3D model corner (Figure 2). Given only one 2D to 3D point correspondence, the position of the camera's projection center has ambiguity; it is along the line of the 3D point and 2D image point (Figure 2 line $P_1 i_1$). The ambiguity can be fixed for both images given one extra 2D image point correspondence between the two images. We give a geometrical explanation for this, and the exact analytical solution is not shown due to limited space.

As shown in Figure 2, $O_1$ $O_2$ are the projection centers of two images, $P_1$ $P_2$ are two 3D model vertices, with $i_1$ $i_2$ as their image points respectively, and $(i_3, i_3')$ is a given image correspondence pair. The

3D position of $O_1$ is along the line $P_1 i_1$. Since the orientation is fixed, the line $O_1 i_3$ forms a plane while its end points $O_1$ moves along the line $P_1 i_1$. The plane $P_1 O_1 i_3$ intersects with the model plane at a line $l_1$ (or a curve for a curved model surface). Similarly, $O_2 i_3^{'}$ forms a plane while its end points $O_2$ moves along the line $P_2 i_2$, and the plane $P_2 O_2 i_3^{'}$ intersects with the model plane at a line $l_2$. So the 3D model point $P_3$ is uniquely determined by the intersection of line $l_1$ and $l_2$. Hence the 3D position of $O_1$ is fixed by the intersection of line $P_1 i_1$ and $P_3 i_3$. Similarly, the 3D position of $O_2$ is fixed by the intersection of line $P_2 i_2$ and $P_3 i_3^{'}$. Thus we can compute the 3D translation for both images.

### 2.3. Global registration

The pose computed for a single image using several point correspondences is not robust, so a global registration process is employed to refine the pose. 2D image corners are extracted for each image, and they are matched automatically using the estimated pose, then the matched corners are used to find corresponding 3D model points, and a bundle adjustment process is used to refine poses by minimizing the overall projection errors.

## 3. Texture generation

A base buffer [5] is allocated for each façade of the building model to store the final textures. Each image is warped to the base buffer, and multiple images are blended.

Due to the illumination variation in different images, the textures have different colors, which cause visual inconsistence. A color rectification process is used to solve the problem. We first chose an image as the base color image, then pixels with constant colors in a window with user defined size (we use size 5 by 5 in our implementation) are extracted, and these pixels are automatically matched with other images using the estimated pose, finally each image is color rectified with an affine color model [8].

$$\begin{bmatrix} r^{'} \\ g^{'} \\ b^{'} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{23} & a_{23} & a_{23} \\ a_{33} & a_{33} & a_{33} \end{bmatrix} \begin{bmatrix} r \\ g \\ b \end{bmatrix} + \begin{bmatrix} O_r \\ O_g \\ O_b \end{bmatrix} \qquad (9)$$

Images taken at different viewpoints cause a parallax problem. Blending all the overlapping area will create ghost effects. We solve the problem by automatically finding the best blending area based on the histogram of the 2D matching corners between images. The size of the area is a user-defined value, which is set to a small value for strong parallax images. This method helps to reduce visual defects caused by parallax for highly structured textures such as building bricks and windows. However, a more complicated algorithm using dynamical programming to find the best cut is necessary for unstructured textures [3].

## 4. Results

Several different textures are generated using the described method. The 3D models are generated using LiDAR data with a semi-automatic modeling system [12]. Pictures are taken with an un-calibrated camera at different time, and the focal length and illumination varies from image to image.

Figure 3(a) and (b) demonstrate the effectiveness of the vanishing points extraction method using filters described in algorithm 1. Many false vanishing points (yellow points) are detected before the filtering due to the heavy occlusion of trees and small-scale textures (Figure 3(a)). These spurious vanishing points are filtered using our algorithm, and only correct dominant vanishing points are identified (Figure 3(b)). The $x$ direction vanishing point for the image in Figure 3(a) is at infinite, and its orientation is recovered using Equation 7. The final texture created from four separated images is shown in Figure 3(i).

Two of the four original images used to create the texture for another building are shown in Figure 3(c) and (d). Each of the four images only covers one corner of the building, so the translation is an under-constrained problem. We first compute the orientation for each of the image using automatically estimated vanishing points [1], then combine the four images to compute the translation, and the final pose are refined using bundle adjustment. The four images are color corrected, and warped to the base buffer using the refined pose. The best blending area are automatically detected (Figure 3(e)) to reduce parallax effects, and the final texture is generated using blending techniques (Figure 3(f)). More textures are generated using the described technique (Figure 3 (h), (i)), and integrated into a 3D environment (Figure 3 (g)).

## 5. Conclusion

This paper presents new techniques to address the challenges in generating textures for urban models. We decompose the camera pose into orientation and translation, and use an edge error model and knowledge-based filters to estimate correct vanishing points. We derive equations to compute the orientation with infinite vanishing points, estimate the translation by combining information from multiple images when each single image does not provide enough constraints, and generate final textures with color calibration and blending. Future work includes using the registered images to correct 3D models and model façade details.

## 6. References


[1] J. Hu, S. You, U. Neumann. Vanishing hull, to appear in *3DPVT*, 2006, UNC.

[2] R. Cipolla, T. Drummond and D.P. Robertson. Camera calibration from vanishing points in images of architectural scenes. *In Proceedings of British Machine Vision Conference*, pp.382–391, 1999.

[3] A. Efros, and W. T. Freeman. Image quilting for texture synthesis and transfer. *Proceedings of SIGGRAPH 2001*, pp. 341–346, 2001.

[4] C. Fruh and a. Zakhor. Constructing 3D city models by merging aerial and ground views. *CGA*, 23(6): 52-61, 2003.

[5] J. Hu, S. You, and U. Neumann. Texture painting from video. *Journal of WSCG, ISSN 1213–6972, Volume 13,*pp. 119–125, 2005.

[6] S.C. Lee, S. K. Jung, and R. Nevatia. Integrating ground and aerial views for urban site modeling. *ICPR*, 2002.

[7] D. Liebowitz and A. Zisserman. Metric rectification for perspective images of planes. *In CVPR*, pp.482–488, 1998.

[8] F. Mindru, L. V. Gool and T. Moons. Model estimation for photometric changes of outdoor planar color surfaces cuased by changes in illumination and viewpoint. *ICPR*, 2002.

[9] J. Shufelt. Performance evaluation and analysis of vanishing point detection techniques. *PAMI*, 21(3):282–288, 1999.

[10] I. Stamos and P. Allen. Automatic registration of 2D with 3D imagery in urban environments. *ICCV*, pp.731–736, 2001.

[11] X. Wang et al. Recovering façade texture and microstructure from real world images. *In Proceedings 2nd International Workshop on Texture Analysis and Synthesis at ECCV*, pp. 145–149, 2002.

[12] S. You, J. Hu, U. Neumann, and P. Fox. Urban Site Modeling From LiDAR, *Second International Workshop on Computer Graphics and Geometric Modeling*, 2003.


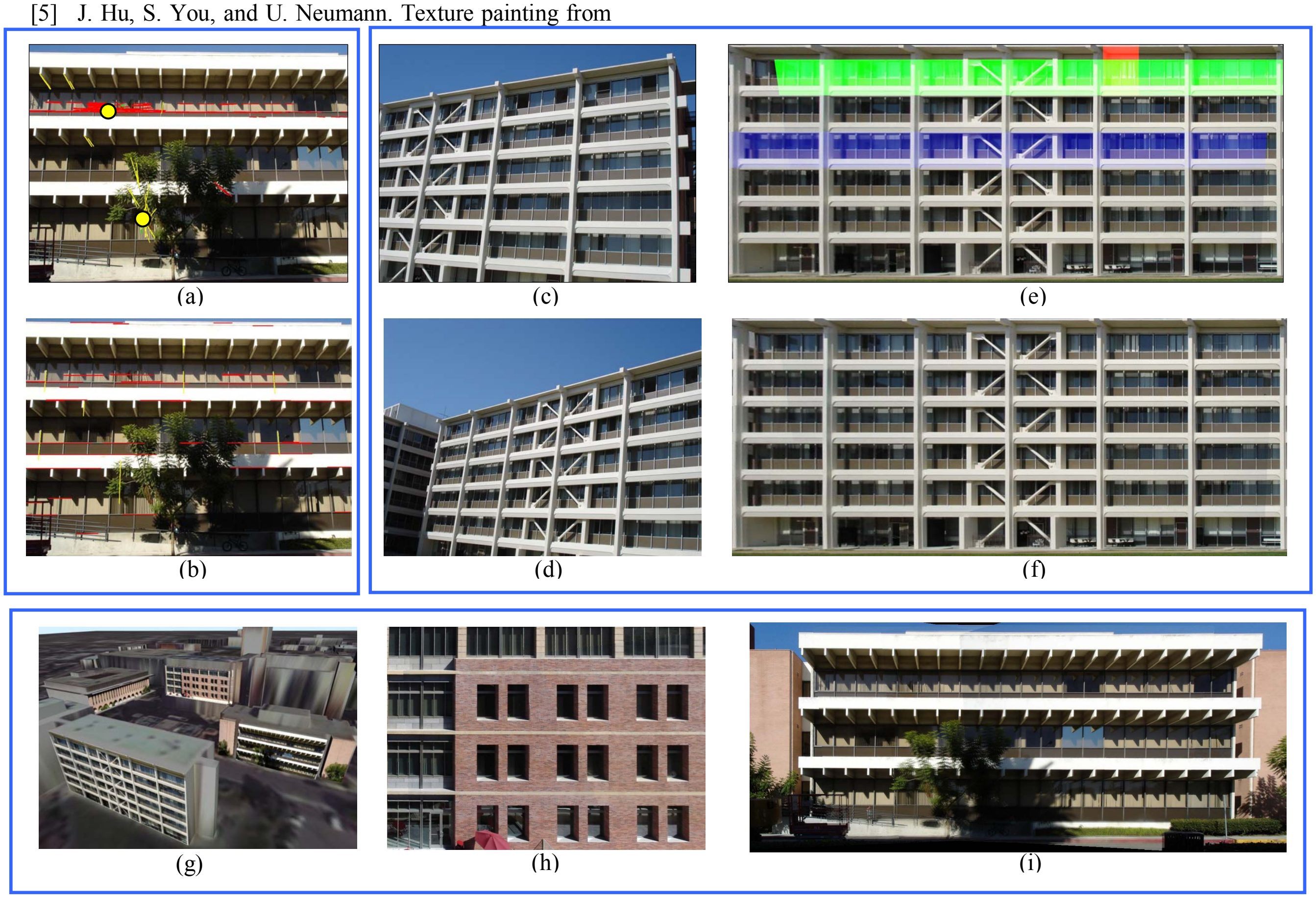


**Figure 3. Results of generated textures.**