Researh and Implementation on 3D Reconstruction with Single View

Xiaochen Zhou
zhouxiaochen@wustl.edu

Abstract

3D model reconstruction is one of the most popular direction in computer vision, and single view reconstruction of piecewise swept scenes, especially in outdoor envrionment, is an interesting and challenging problem. In this paper, I recurrent effective methods for camera calibration from vanishing points and pixel-wise 3D model reconstruction throngh one 2D single-view picture for architecture in outdoor environment. Then some optimizations and predigestions on reconstruction for models with regular shape in outdoor environment are implemented to solve the reconstruction tasks more efficiently.

1 Introduction

The reconstruction of 3D models from 2D images is a long history and interesting problem in computer vision. This technology can be used in several fields, like video game modeling, auto-driving, architecture reconstrution and visualization. The single view reconstruction, as a challenging problem in reconstruction fields, is highly anticipated owing to the efficiency, low redundancy and high precision. How to find effective and precise method to achieve camera calibration from single view and reconstruct the 3D model through one single 2D projected image are still hot topics in recent few years.

One effective method for single-view reconstruction is [1], where the authors present an approach to solve the reconstruction problem by computing the dense orientation map. As the prerequisite, camera calibration is required for the valid projection from 3D to 2D and for the generation of the orientation map. An effective method for camera calibration is necessary for the rapid reconstruction. While methods in [2] can handle the problem, the algorithm in [4] is much suitable for this task, where the parameters of camera calibration are calculated through vanishing points which are also utilized in reconstruction work.

To solve the reconstruction problem, normals for all the pixels in the image are required, which is solved by sequential methods where normals for faces are calculated through directions of lines and normals for lines are solved via normals of known faces and the faces formed through the center of camera and the lines in 3D space. Then we can get the normals for all pixels in the single image. This orientation map is utilized to get a depth map for all pixels, which is similar to photometric shapes from shading techniques like [5] and [6]. Considering that the implemented method is used for architecture with regular shape in outdoor environment, some optimizations and simplifications are conducted in the algorithm. This algorithm will be explained in Section 3.

An overview of the following sections is as follows. In section 2, we will introduce the background and related work in 3D model reconstruction. In section 3, alogrithms for camera calibration and reconstruction will be explained in detail. Then section 4 will show the result of the experiment and analysis the result. The last section is the conclusion.

2 Background & Related Work

In recent years, several researchers paid attention to 3D reconstructions with single view. Compared with 3D reconstruction with multi-views, like Furukawa et.al [12], single-view reconstruction satisfies the requirements where only one side of 3D model is necessary, such as city planning, video game modeling for line-based scenes and coarse panorama for film industry. Instead of generating all points in 3D space, only reconstructing required points greatly lower the computation and redundancy, making the real-time generation possible. Also, single-view reconstruction is specially suitable for architecture designers and graphic designers to get an easy access to 3D scene of their designs.

For reconstruction from line drawings, as is mentioned in [1], Kanade [7] reconstructed composites of shells and sheets as belonging to the special class of ”origami world”. Huffman [8] used line intersections between concave, convex and occluding intersections to detect objects. Horry et.al [9] exerted method to build piecewise planar reconstution system for paintings and photographs. These methods all require tremendous information marked by users, which is restricted and not effective.

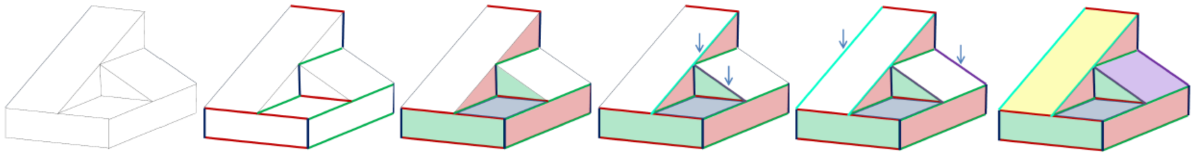

Figure 1: Steps for reconstruction. (1). cluster the lines in orthogonal directions and compute the directions. (2). compute the normal for faces bounded by clustered lines. (3). compute the directions of lines on faces with computed normals. (4). cluster the lines with the same direction of the line with new computed directions. (5). repeat step (3) to compute normal for faces

Several researches have been conducted on reconstruction with point cloud analysis. Vesselman et.al [10] took advantages of laser scanners to build point clouds for hard detected structure, such as roof etc. Carr et.al [11] use polyharmonic Radial Basis Functions to reconstruct smooth, manifold surfaces from point-cloud data and to repair incomplete meshes. These methods can relatively improve the precision of reconstruction. However, the time consuming and equipments required for these methods made the reconstruction complicated.

Inspired by reconstruction through line drawing, this method use less information from user marks and build a automatic system to extend the depth and normal information from lines to faces and faces to lines. Proper simplification and approximation lower the computation without sacrificing the accuracy of reconstruction.

3 Proposed Approach

3.1 Camera Calibration

In this section, we will introduce the algorithm of camera parameters approaching. Vanishing points corresponding to three mutually orthogonal directions will be used to figure out the right parameters:

- the camera calibration matrix, denoted as *K*. Assume that calibration matrix has zero skew.
- the rotation matrix *R*.
- the direction of translation *T*. Considering that *T* is related to the position of camera in 3D space, *T* is set as zero vector, meaning that the center of camera is set in the original point for the convenience of reconstruction.
- the projecting matrix *P*.

Considering the points at infinity corresponding to the direction of three orthogonal axis, we can get the equation as follows:

$$\begin{bmatrix} \lambda_1 u_1 & \lambda_2 u_2 & \lambda_3 u_3 \\ \lambda_1 v_1 & \lambda_2 v_2 & \lambda_3 v_3 \\ \lambda_1 & \lambda_2 & \lambda_3 \end{bmatrix} = P \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

where $[u_i, v_i]$ denotes the vanishing point x_i of the ith direction. Considering that *T* is set as zero vector, the equation can be written as:

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

where $[u_0, v_0]$ is denoted as x_i. Owing to the orthonormality of *R*, *f* can be recovered by x_1, x_2, x_3 and x_0 as:

$$(x_1 - x_0)(x_2 - x_0) + f^2 = 0$$
$$(x_2 - x_0)(x_3 - x_0) + f^2 = 0$$
$$(x_1 - x_0)(x_3 - x_0) + f^2 = 0$$

Through the equation above, the relationship between vanishing points can be expressed as:

$$(x_1 - x_0)(x_2 - x_3) = 0$$

Then according to Caprile and Torre [3], row normality for *R* must be satisfied owing to the geometric interpretation. Thus, the following equation can be achieved:

$$\lambda_1^2(u_1 - u_0) + \lambda_2^2(u_2 - u_0) + \lambda_3^2(u_3 - u_0) = 0$$
$$\lambda_1^2(u_1 - u_0) + \lambda_2^2(u_2 - u_0) + \lambda_3^2(u_3 - u_0) = 0$$
$$\lambda_1^2 + \lambda_2^2 + \lambda_3^2 = 1$$

Then the parameter for each λ can be calculated as:

$$\lambda_1^2 = \frac{(v_0 - v_3)(u_2 - u_3) - (u_0 - u_3)(v_2 - v_3)}{(v_1 - v_3)(u_2 - u_3) - (u_1 - u_3)(v_2 - v_3)}$$

λ_2 and λ_3 can be approached as the same method. Then all the parameters acquired can be calculated.

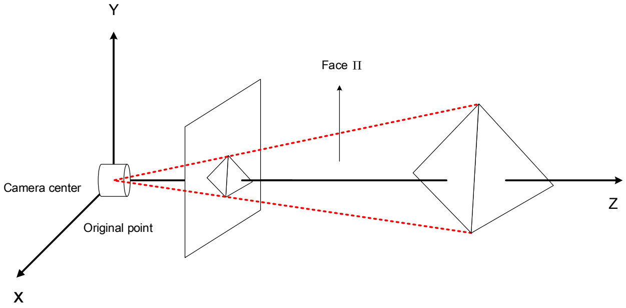

Figure 2: Visualization of computing line direction.

3.2 Single View Orientation Mapping

To clarify the process of reconstruction, Figure 1 shows the steps of the reconstruction. The 2D view provides the information of faces and lines in the scene which should be recorded by users. Faces refer to a close region bounded by several lines, denoted as $f_i \in F$ where F means the set of faces, and lines can be expressed as the intersection of faces or the connection of two points. For the optimization, different from [1], lines formed through two faces are denoted as $l* \in L$ where L means the set of lines, and other lines denoted as $l \in L$. Normals for lines and faces are denoted as d_l, d_{l*} and d_f.

To calculate the camera calibration, the vanishing points of orthonogal directions should have been achieved by users. In this case, the direction for these lines pass through the vanishing points can be calculated as:

$$d_l = R^{-1}K^{-1}vp_i$$

where vp_i is denoted as the ith vanishig point. In this case, d_l for three orthogonal directions have been calculated. Considering for the faces bounded by the lines with the known direction, the normals for these faces can be obtained as:

$$d_f = d_{l_1} \times d_{l_2}$$

Thus, the normals for faces bounded via l can be calculated. Given the orientation of face normals, lines $l*$ bounding the faces with known normal can be computed. To make a clear clarifying, Figure 3.2 shows how this step works. First, the projected view should be set in the 3D space, where the center of the camera, lines formed by two points in projected view, called l_π, and the 3D object in 3D space are in the same plate, denoted as f_π. Owing that l_π can be formed as the intersection of f_π and another face with the known normal, the direction of l_π can be expressed as:

$$d_{l_\pi} = d_{f_\pi} \times d_f$$

So the mission is to compute d_{f_π}. Considering that points in 2D view is shaped like $z_i[u_i, v_i, 1]$, where z_i is a free variable, it is unable to get the accurate position for this point in 3D space after the rotation. However, all the possible positions should be in a line pass through the camera and the center of the camera is set to original point, which means that the direction of this line is the same as the point. In this case, after the normalization, the direction of two lines bounding f_π can be achieved. Then the normal of f_π can be computed. Then the directions for $l*$ can be calculated.

The algorithm should be functioned until all the normals for lines and faces are achieved.

3.3 Reconstruction

Given the normal of all pixels, next step is to compute the depth of each pixels. In [1], the authors provided a non-convex method and then several optimizations are applied to ease the method. In this section, an easier applied method will be given. Considering that all the pixels in 2D images are connected, if the depth of one pixel is set, all other points can be calculated through this point and breadth-first search (BFS) can be exerted to achieve the depth of all the pixels. The following three situations should be taken into consideration.

- *Face to face* The depth-known pixel belongs to the same face as the target pixel. In this case, normal of these two points should be the same. Considering the normal of the face and the direction of these two points should be perpendicular, the following equation can be written:

  $$d_f(P_1 - P_2) = 0$$

  where P_1 and P_2 are the two points in 3D space.

- *Face to line* The target pixel is in a line $l*$ belonging to both two faces. In this case, the normal of this pixel will be considered as the depth-known pixel, which means that the direction of these two points in 3D space should be perpendicular with the normal of face where the depth-known pixel is set.

- *Line to face* The target pixel is in a face and the depth-known pixel is in a line. This method is the same as face to line method. The only difference is the normal of face should be that the target pixel belongs to.

To implement the reconstruction, the influence of the free variable z should be taken into consideration. Considering that one 3D point is rotated by matrix *R* and vector *T*, denoted as $[x', y'z']$, then the relationship between the projection pixel and rotated points can be written as follows:

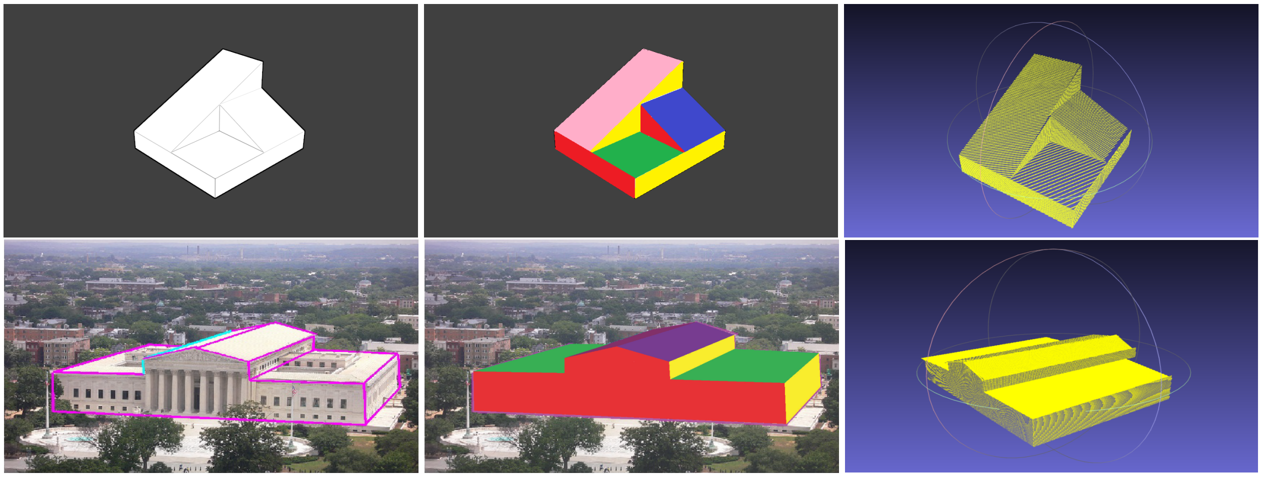

Figure 3: Result of reconstruction. The first image is the single-view image. Then the second image shows the normal of all pixels in the object, where the color for each pixel is normalized through normal vector sphere. The third image is the reconstructed 3D object.

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} f & 0 & u_0 \\ 0 & f & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix}$$

We can transform this equation as follows:

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = z' \begin{bmatrix} \frac{x-u_0}{f} \\ \frac{y-v_0}{f} \\ 1 \end{bmatrix}$$

Thus, the reconstruction period can be expressed as:

$$P = z' R^I p$$

where p means the position of pixels in the image and P means the 3D point reprojected through p. In this case, taking the reconstruction equation into account, we can achieve the relationship between two pixels in 3D space:

$$d_f(z_1' R^I p_1 - z_2' R^I p_2) = 0$$

$$z_1' = z_2' \frac{d_f R^I p_2}{d_f R^I p_1}$$

where p_1 is the target pixel and z_1' is the depth that we need to compute, and p_2' is the depth-known pixel and z_2' is the calculated depth. In this case, we can calculate the depth of all the pixels in the image.

4 Experimental Results

4.1 Implementation Details

To successfully reconstruct the 3D model, several parameters are required from users. First, we need users to mark the outlines and faces of the model. Then, cluster the lines in 3 orthogonal directions. After obtaining the necessary information, we can use the clustered lines to generate corresponding vanishing points and calculate camera calibration. To minimize the deviation, RANSAC method is used to generate vanishing points. Then, marked lines and faces will be used to check which face or line one pixel should belong to. Next, we randomly select one pixel in one face, reproject it into 3D space and initialize the depth to 100. After that, all the pixels near around this pixel will be computed. When all the points belonging to a line or a face have been calculated, the process will stop.

4.2 Experiment Result and Analysis

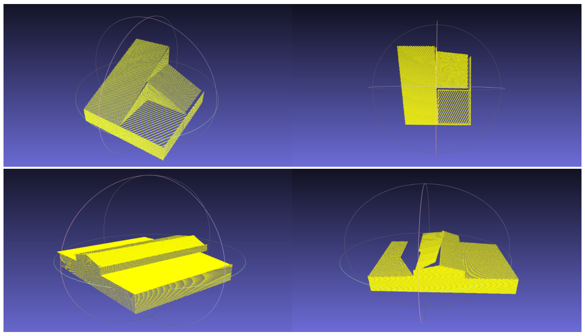

Figure 4: Rotation of the reconstructed 3D model. (1). Deviations from users' mark and pixel-wise method lead to the margine and skew of some shapes. (2). It is reasonable that points occluded by the model, which is invisible in the single view, can not be learned and reconstructed

The result can be found in Figure 3.2. As can be seen, all the faces with the same color share the same normal, and the reconstructed 3D objects present precise architecture of the models. Besides, the time-consuming is acceptable, where reconstructing one model with 256076 pixels takes around 72 seconds with Intel i5 2.3 GHz CPU.

Though the performance of reconstruction is satisfying and effective, however, some deviations still occur. The rotated model of the 3D model can be seen in Figure 4.2. There are several reasons for these errors. First, deviations from hand-craft marks to some extent bring errors to the computation, which leads to the skew of some shapes. Besides, the uniform sampling in the image plane grid will cause a non-uniform sampling in 3D space.

4.3 Further Work

In the future, we will pay more attention to solving the diviation from hand-craft marks and margins from pixelwise method. First, machine learning and convolutional neural network would be used to develop the end-to-end model for framework extraction and reconstruction. Next, we will utilize verticle and lines to reconstruct the 3D model, where we can compute the mean value of intersected points to avoid margins between shapes. Then we will map the texture to the 3D model and evaluate the performance of texture-mapped models.

5 Conclusion

In this paper, we implement a 3D model reconstruction method through single view and do some optimizations on this method. Vanishing points are used to compute the camera calibration parameters and 3 orthogonal directions. Then we use the calculated directions from lines and normals faces to compute the unknown directions for lines and normals for faces. Next, we initialize the depth for one point and use breadth-first search method to compute all the depth of pixels belonging to lines or faces.

For future work, we will use machine learning and deep learning method to optimize the deviations from hand-craft marks and margins between shapes. Besides, an end-to-end model will be trained for framework extraction and model reconstruction.

References

[1] A. Kushal and S. M. Seitz. Single view reconstruction of piecewise swept surfaces. In *3D Vision-3DV 2013, 2013 International Conference on*, pages 239–246. IEEE, 2013.

[2] R. Szeliski. *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.

[3] B. Caprile and V. Torre. Using vanishing points for camera calibration. *International journal of computer vision*, 4(2):127–139, 1990.

[4] R. Cipolla, T. Drummond, and D. P. Robertson. Camera calibration from vanishing points in image of architectural scenes. In *BMVC*, volume 99, pages 382–391, 1999.

[5] P. Kovesi. Shapelets correlated with surface normals produce surfaces. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 2, pages 994–1001. IEEE, 2005.

[6] E. Prados and O. D. Faugeras. Perspective shape from shading and viscosity solutions. In *ICCV*, volume 3, page 826, 2003.

[7] T. Kanade. A theory of origami world. *Artificial intelligence*, 13(3):279–311, 1980.

[8] D. A. Huffman. Impossible object as nonsense sentences. *Machine intelligence*, 6:295–324, 1971.

[9] Y. Horry, K.-I. Anjyo, and K. Arai. Tour into the picture: using a spidery mesh interface to make animation from a single image. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 225–232. ACM Press/Addison-Wesley Publishing Co., 1997.

[10] G. Vosselman, S. Dijkman, et al. 3d building model reconstruction from point clouds and ground plans. *International archives of photogrammetry remote sensing and spatial information sciences*, 34(3/W4):37–44, 2001.

[11] J. C. Carr, R. K. Beatson, J. B. Cherrie, T. J. Mitchell, W. R. Fright, B. C. McCallum, and T. R. Evans. Reconstruction and representation of 3d objects with radial basis functions. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 67–76. ACM, 2001.

[12] Y. Furukawa, C. Hernández, et al. Multi-view stereo: A tutorial. *Foundations and Trends® in Computer Graphics and Vision*, 9(1-2):1–148, 2015.