

Interactive Indoor Scene Reconstruction From Image Mosaics Using Cuboid Structure

Bo Hu Christopher Brown

The University of Rochester
Computer Science Department
Rochester, New York 14627

Technical Report 787

August 2002

Abstract

We describe a method for reconstructing indoor scenes from image mosaics using prior knowledge of the cuboid structure of the environment. The method is inspired by traditional approaches to the camera pose estimation problems known as the **PnL** and **PnA** problems. We show that a cuboid can be reconstructed from the images of three of its corners. The necessary camera intrinsic parameters are obtained by self-calibration in the image mosaicking process. The major advantages of this method over methods such as single view metrology are 1) it can do metric reconstruction; 2) it is a closed form solution so it is numerically stable; 3) it requires only minimal user interaction.

Keywords: scene reconstruction, image mosaicking, pose estimation

1 Introduction

This report tackles the problem of reconstructing indoor walls and lighting models from image mosaics of the scene. The image mosaic is created by stitching together images taken by a stationary camera panning around its principal point. Since there is no parallax between any images, it is essentially a single-view image of the scene, which means traditional triangulation methods for depth or structure reconstruction do not apply. On the other hand, indoor scenes have desirable regularities, such as parallel structures and orthogonal planes, which facilitate the use of methods like single view metrology [Criminisi *et al.*, 2000]. An image mosaic is a single *wide*-view picture of the scene. One can recover a large portion of a scene at once from an image mosaic using the single view metrology method, but there are some disadvantages of this approach:

- It requires substantial human interaction. One has to identify a reference plane and do every single measurement by hand. To reconstruct a whole scene, several reference planes have to be given and the process can be quite laborious.
- The method heavily relies on accurate estimation of vanishing points, which is numerically unstable.
- Theoretically, single view metrology methods can only do affine reconstruction. While in our application, a metric reconstruction is required.

Besides the above mentioned disadvantages, methods using straight single view metrology do not fully employ the scene regularities, such as orthogonality. It is easily noticed that indoor scenes have abundant cuboid structures. Reconstruction using cuboid structures directly utilizes the scene regularities and saves a great deal of human interaction. Chen *et al* [Chen *et al.*, 1999] pointed out when 6 points of a cuboid are known, the camera parameters and the structure of the cuboid can be recovered. Wilczkowiak *et al* [Wilczkowiak *et al.*, 2001] extended Chen’s work by revealing the duality between a parallelepiped and the camera parameters. In this report, we present a method to reconstruct indoor scenes from image mosaics using cuboid structure. The camera parameters are recovered during the creation of the image mosaic. Once the camera parameters are known, two opposite points plus an arbitrary point of a cuboid are enough to obtain the structure of the cuboid. Our method is closely related to the pose estimation problem of **PnL** (*perspective n lines*) [Liu *et al.*, 1990] and **PnA** (*perspective n angles*) [Kanatani, 1988; Wu *et al.*, 1994]. The novelty of our method lies in

1. The algorithm combines the approaches of **PnL** and **PnA** methods, which enables us to recover both the orientation and translation of a corner.
2. Our presentation is geometrically clearer than those in [Kanatani, 1988; Wu *et al.*, 1994]. Our method eliminates some unnecessary complications brought by angle measurement and use of multi-valued functions.

The contribution of this work is that we extend the cuboid-based reconstruction methods by exploiting the specialty of image mosaics, namely, that the images are taken by a stationary camera. We only need three points instead of six as in [Chen *et al.*, 1999; Wilczkowiak *et al.*, 2001]. The human interaction is significantly reduced comparing with that of single view metrology methods and a metric reconstruction is achieved.

This report is organized as follows. First we introduce some background and notation. We then briefly describe how we create the image mosaic and calibrate the camera. In Section 5, we give the details of our reconstruction method. Finally, some results are shown.

2 Background and notations

We model imaging as a standard pin-hole camera with the origin at the center of projection, z -axis aligned to the view axis and the plane $z = 1$ being the retina plane. A space point $\mathbf{P} = (X, Y, Z)$ in the camera frame is projected to a retina point $\mathbf{p} = (x, y, z)$ by the following simple equation:

$$\begin{cases} x &= X/Z \\ y &= Y/Z \end{cases} \quad (1)$$

If \mathbf{P} is in a world coordinate system, it is first transformed into the camera frame by a rigid body transformation:

$$\mathbf{P}' = R\mathbf{P} + \mathbf{t} \quad (2)$$

where R is an orthonormal rotation matrix and \mathbf{t} is a 3D translation vector.

The relationship between a retina point $\mathbf{p} = (x, y, z)$ and its pixel coordinates is defined by:

$$\begin{cases} x &= (u - u_0)/f_u \\ y &= (v - v_0)/f_v \\ z &= 1 \end{cases}$$

where (u, v) are the pixel coordinates of \mathbf{p} and f_u and f_v are the scale factors of the u , v axes, respectively. The point (u_0, v_0) is the image of the center of projection.

The imaging process can be written in matrix form as:

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = K \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} E \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \quad (3)$$

where

$$K = \begin{pmatrix} f_u & 0 & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{pmatrix} \quad (4)$$

is the camera intrinsic matrix and

$$E = \begin{pmatrix} R & \mathbf{t} \\ \mathbf{0}^T & 1 \end{pmatrix}$$

is the rigid body transformation.

In the above equation, matrices are in capital letters and vectors are in bold type. We also use capital letters for 3D points. The meaning should be clear from context. We use \overline{AB} for a directed segment from point A to point B and \overleftrightarrow{AB} for a directed infinite line passing through point A and then through B .

3 Image mosaicking

When images are taken by a camera rotating around its center of projection, it is possible to stitch the images together to form a wide-view image of the scene. The process is called *image mosaicking* [Szeliski, 1996; Shum and Szeliski, 2000]. Setting the origin of the world coordinate system at the rotating point, we have $\mathbf{t} = 0$ in Equation (2). Equation (3) becomes

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = K \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} R & \mathbf{0} \\ \mathbf{0}^T & 1 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = KR \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}. \quad (5)$$

Let's consider a point \mathbf{x} and its two projections \mathbf{u}_i and \mathbf{u}_j in two images taken by a rotating camera. We have $\mathbf{u}_i = KR_i\mathbf{x}$ and $\mathbf{u}_j = KR_j\mathbf{x}$. It follows that

$$\mathbf{u}_j = KR_jR_i^{-1}K^{-1}\mathbf{u}_i = KR_{ij}K^{-1}\mathbf{u}_i = H_{ij}\mathbf{u}_i \quad (6)$$

where $R_{ij} = R_jR_i^{-1}$ is the rotation between image i and image j and $H_{ij} = KR_{ij}K^{-1}$ is the *homography* between the two images.

Selecting a reference image I_0 and computing H_{0j} for each image I_j , we can stitch other images to the reference image. See [Hu *et al.*, 2001] for details. Figure 6(c) shows an image mosaic of an office scene and eight images from the set of images we used to create the mosaic.

4 Camera calibration

Our reconstruction method is based on the pose estimation algorithm which requires the intrinsic parameters of the camera. The intrinsic parameters can be obtained by an offline calibration process. However, the image set used for mosaicking is usually big and this redundancy makes camera self calibration possible [Hartley, 1997; de Agapito *et al.*, 1999].

We know from Equation (6) that a homography H_{0j} is similar to an orthonormal matrix

$$K^{-1}H_{0j}K = R_{0j}.$$

Multiplying both side with the transposes, we have $K^{-1}H_{0j}KK^TH_{0j}^TK^{-T} = I$, where I is a 3×3 identity matrix. It follows that

$$H_{0j}CH_{0j} = C \quad (7)$$

where $C = KK^T$ is a positive definite symmetric matrix. Equation (7) provides six linear equations of entries in C . For reasons given in [Hartley, 1997], at least two homographies (three views) are needed to determine C uniquely. After C is computed, the intrinsic matrix K can be obtained by Cholesky decomposition of C . Two implementation details are worth to point out.

1. The matrix C solved from Equation (7) may be negative definite. However, since the solution is up to a scale factor, we can just negate the solution and get a positive definite C .
2. The common presentation of Cholesky decomposition is to factor a positive definite symmetric matrix to the product of a lower triangle and its transpose. Consequently, the Cholesky decomposition routines in common software packages, such as MATLAB, LAPACK and Numerical Recipes, output a lower triangle matrix L such that $C = LL^T$. Note that we cannot just let $K = L^T$, because matrix multiplication is not commutative. We have to write our own Cholesky decomposition routine.

5 Scene reconstruction using cuboid structure

5.1 Pose from a corner

The problem is formulated as follows. Given an image (o, x, y, z) of a rectangular corner ¹ (O, X, Y, Z) , determine the pose of the camera (Figure 1).

¹The following derivation can be extended to a general non-orthogonal corner if we know the three angles of the corner.

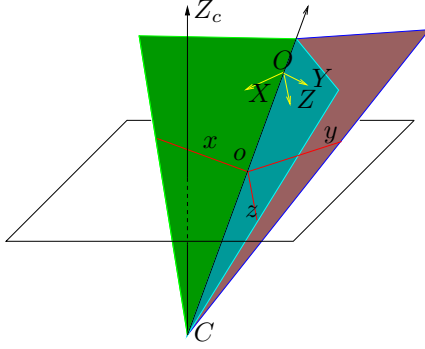


Figure 1: (o, x, y, z) is an image of a 3D corner (O, X, Y, Z) . The fact that a 3D line OX is in the plane defined by the center of projection (C) and an image line ox gives the constraints to solve the pose of the corner (see text).

Let the space line \overrightarrow{OX} be

$$\overrightarrow{OX} = \mathbf{v}t + \mathbf{x}_0$$

where \mathbf{v} is the direction of the line and \mathbf{x}_0 is any point (e.g. O) on the line. Expressed in the camera coordinate system, the line is

$$\begin{aligned} \mathbf{v}' &= R\mathbf{v} \\ \mathbf{x}'_0 &= R\mathbf{x}_0 + \mathbf{t} \end{aligned}$$

Let the corresponding image line \overrightarrow{ox} be

$$Ax + By + C = 0$$

where x and y are in the image coordinates.

Substituting Equation (1) into the above equation, we have

$$AX + BY + CZ = 0,$$

or

$$\mathbf{n} \cdot \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = 0 \quad (8)$$

where $\mathbf{n} = (A, B, C)^T$ and X, Y and Z are in the space coordinates. Equation (8) is a plane with normal \mathbf{n} passing the center of projection and the lines \overrightarrow{ox} and \overrightarrow{OX} (Figure 1). The fact can be written as

$$\mathbf{n} \cdot R\mathbf{v} = 0 \quad (9)$$

$$\mathbf{n} \cdot (R\mathbf{x}_0 + \mathbf{t}) = 0. \quad (10)$$

So each line correspondence provides two constraints. The pose can be solved if three or more line correspondences are available [Liu *et al.*, 1990]. This problem is usually called the **P3L** (*Perspective 3 Lines*) problem. When the three lines intersect, i.e. forming a corner, the orientation of the camera can be determined from the angles between the three lines. This problem is called the **P3A** (*Perspective 3 Angles*) problem [Wu *et al.*, 1994]. The solution of the **P3A** problem has been well established by [Kanatani, 1988; Wu *et al.*, 1994]. Here we combine approaches of **P3L** and **P3A** and give a new geometrical presentation. By using lines instead of angles directly, our method eliminates

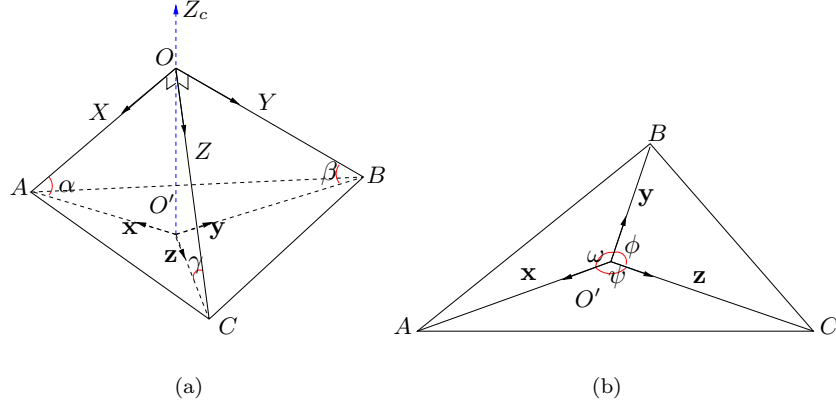


Figure 2: Solving the orientation of a corner (see text). (a) is a 3D view of a rectangular corner (O, X, Y, Z) and its image (o, x, y, z). The bottom triangle of the tetrahedron $OABC$ is shown in (b).

some unnecessary complications in [Kanatani, 1988] caused by angle measurement and use of multi-valued functions. Also, our method can solve the translation up to a scale factor, whereas methods in [Kanatani, 1988; Wu *et al.*, 1994] only find out the orientation. We use this method later to reconstruct a cuboid.

As Kanatani did in [Kanatani, 1988], we first rotate the image plane by a rotation matrix R_0^T to align the view axis $\overrightarrow{CZ_c}$ to $\overrightarrow{C\hat{o}}$ (Figure 1). The rotation matrix R_0 can be obtained by

$$\mathbf{rotate}(\overrightarrow{CZ_c} \times \overrightarrow{C\hat{o}}, \arccos(\frac{\overrightarrow{CZ_c} \cdot \overrightarrow{C\hat{o}}}{|\overrightarrow{CZ_c}| |\overrightarrow{C\hat{o}}|})).$$

where $\mathbf{rotate}(\mathbf{r}, \theta)$ means rotating angle θ around axis \mathbf{r} . Details can be found in [Kanatani, 1988]. Figure 2(a) shows an image of a rectangular corner after the above rotation is applied. Note that the construction is invariant to the length $|\overrightarrow{OO'}|$. If O moves along $\overrightarrow{OO'}$, the image of the corner will not change. This motion along $\overrightarrow{OO'}$ is in fact the scale factor of the translation we cannot decide. Setting $|\overrightarrow{OO'}| = 1$, we have

$$\begin{aligned} |\overrightarrow{O'A}| &= \cot \alpha \\ |\overrightarrow{O'B}| &= \cot \beta \\ |\overrightarrow{O'C}| &= \cot \gamma. \end{aligned}$$

Consider an edge, e.g. \overrightarrow{AC} , of the triangle $\triangle ABC$. In triangle $\triangle O'AC$ (Figure 2(b)), we have

$$|\overrightarrow{AC}| = \cot^2 \alpha + \cot^2 \gamma - 2 \cot \alpha \cot \gamma \cos \psi.$$

And in triangle $\triangle OAC$ (Figure 2(a)), we have

$$|\overrightarrow{AC}| = \sec^2 \alpha + \sec^2 \gamma,$$

Combining the two equations, we obtain

$$\sec^2 \alpha + \sec^2 \gamma = \cot^2 \alpha + \cot^2 \gamma - 2 \cot \alpha \cot \gamma \cos \psi.$$

Or

$$\tan \alpha \tan \gamma = -\cos \psi.$$

Similarly,

$$\begin{aligned}\tan \alpha \tan \beta &= -\cos \omega \\ \tan \beta \tan \gamma &= -\cos \phi\end{aligned}$$

Multiplying the last three equations together, we have

$$\tan^2 \alpha \tan^2 \beta \tan^2 \gamma = -\cos \phi \cos \psi \cos \omega \quad (11)$$

The cosine values can be computed from dot products of the three image lines. Because $\phi + \psi + \omega = 2\pi$, the right hand side of Equation (11) is non-negative. We can then solve for $\tan \alpha, \tan \beta$ and $\tan \gamma$. Let $0 < \alpha, \beta, \gamma < \pi/2$, we have

$$\begin{aligned}\tan \alpha &= \frac{\sqrt{-\cos \phi \cos \psi \cos \omega}}{\tan \beta \tan \gamma} \\ \tan \beta &= \frac{\sqrt{-\cos \phi \cos \psi \cos \omega}}{\tan \alpha \tan \gamma} \\ \tan \gamma &= \frac{\sqrt{-\cos \phi \cos \psi \cos \omega}}{\tan \alpha \tan \beta}\end{aligned} \quad (12)$$

Equation (12) is similar to that in [Kanatani, 1988]. But in our solution, all the values are computed directly from image measurements. We are not facing the problems of choosing the right sign of an angle and the right value from a multi-valued function such as the arctangent.

The image (O', A, B, C) also admits a mirror solution, which is the corner with O at the other side of O' . The mirror solution can be obtained by negating α, β and γ .

After $\tan \alpha, \tan \beta$ and $\tan \gamma$ are computed, we are ready to compute the orientations of \overrightarrow{OX} , \overrightarrow{OY} and \overrightarrow{OZ} , which are orientations of the X, Y and Z axes in the rotated camera frame.

$$\begin{aligned}\overrightarrow{OA} &= |\overrightarrow{O'A}| \mathbf{x} - (0, 0, 1)^T = \cot \alpha \mathbf{x} - (0, 0, 1)^T \\ \overrightarrow{OB} &= |\overrightarrow{O'B}| \mathbf{y} - (0, 0, 1)^T = \cot \beta \mathbf{y} - (0, 0, 1)^T \\ \overrightarrow{OC} &= |\overrightarrow{O'C}| \mathbf{z} - (0, 0, 1)^T = \cot \gamma \mathbf{z} - (0, 0, 1)^T\end{aligned}$$

and

$$\begin{aligned}\overrightarrow{OX} &= \overrightarrow{OA} / |\overrightarrow{OA}| \\ \overrightarrow{OY} &= \overrightarrow{OB} / |\overrightarrow{OB}| \\ \overrightarrow{OZ} &= \overrightarrow{OC} / |\overrightarrow{OC}|\end{aligned}$$

The orientations of the X, Y and Z axes in the original camera coordinates are:

$$\begin{aligned}\mathbf{X} &= R_0 \overrightarrow{OX} \\ \mathbf{Y} &= R_0 \overrightarrow{OY} \\ \mathbf{Z} &= R_0 \overrightarrow{OZ}\end{aligned}$$

Once the orientations of the axes are known, the rotation matrix R is immediate:

$$R = (\mathbf{X} \ \mathbf{Y} \ \mathbf{Z}). \quad (13)$$

Since point O is on all three lines \overrightarrow{OX} , \overrightarrow{OY} and \overrightarrow{OZ} , without loss of generality we can let O be the origin of the world coordinate system, Equation (10) can then be written

$$N \mathbf{t} = \begin{pmatrix} \mathbf{n}_x^T \\ \mathbf{n}_y^T \\ \mathbf{n}_z^T \end{pmatrix} \mathbf{t} = \mathbf{0} \quad (14)$$

where $\mathbf{n}_x, \mathbf{n}_y$ and \mathbf{n}_z are the normals of the planes passing through \overrightarrow{OX} , \overrightarrow{OY} and \overrightarrow{OZ} . Because $\mathbf{n}_x, \mathbf{n}_y$ and \mathbf{n}_z are three planes intersecting at a line, N has a rank of two. Thus, \mathbf{t} is in the 1-D zero space of N . In other words, \mathbf{t} is determined up to a scale factor.

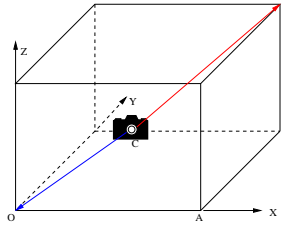


Figure 3: Given a pair of opposite corners (O and P) and an arbitrary third corner (A), we can reconstruct the cuboid (see text).

5.2 Reconstruction of an indoor scene using cuboid structure

A key observation is that the mosaicking images are taken by a stationary camera, which means the translation vector \mathbf{t} is the same for all images. By a counting argument, we know that at least three corners can reconstruct a cuboid: solving each of the three corners recovers five DoF (degrees of freedom). Three corresponding rotations have nine DoF. A single translation has three DoF and the structure of a cuboid has three DoF (length, width and height). These add up to fifteen DoF which is the same as three corners provide. More specifically, we will show that a pair of opposite corners and an arbitrary third corner will reconstruct the cuboid.

Without loss of generality, let's assume we know the image of $O(0, 0, 0)$, $P(l, w, h)$ and $A(l, 0, 0)$, where l, w and h are the length, width and height of the cuboid, respectively (Figure 3). Let $(R_O, \mathbf{t}), (R_P, \mathbf{t})$ and (R_A, \mathbf{t}) be the transformations that transform O, P, A into the camera's coordinate system. We can compute R_O, R_P and R_A using the method described in Section 5.1. From Equation (10), we also have the following linear equations for each of the three points:

$$N_p \mathbf{p} = \begin{pmatrix} \mathbf{n}_x^T R \\ \mathbf{n}_y^T R \\ \mathbf{n}_z^T R \end{pmatrix} \mathbf{p} = - \begin{pmatrix} \mathbf{n}_x \cdot \mathbf{t} \\ \mathbf{n}_y \cdot \mathbf{t} \\ \mathbf{n}_z \cdot \mathbf{t} \end{pmatrix}, \quad (15)$$

where \mathbf{p} can be O, P and A . When $\mathbf{p} = O(0, 0, 0)$, Equation (15) becomes Equation (14), from which we can solve $\mathbf{t} = \zeta \mathbf{t}_0$, where \mathbf{t}_0 is a unit vector in the zero space of N_O and ζ is an unknown constant. For points P and A , we have the following equations:

$$\begin{aligned} N_P \begin{pmatrix} l \\ w \\ h \end{pmatrix} &= -\zeta \begin{pmatrix} \mathbf{n}_x^P \cdot \mathbf{t}_0 \\ \mathbf{n}_y^P \cdot \mathbf{t}_0 \\ \mathbf{n}_z^P \cdot \mathbf{t}_0 \end{pmatrix} \\ N_A \begin{pmatrix} l \\ 0 \\ 0 \end{pmatrix} &= -\zeta \begin{pmatrix} \mathbf{n}_x^A \cdot \mathbf{t}_0 \\ \mathbf{n}_y^A \cdot \mathbf{t}_0 \\ \mathbf{n}_z^A \cdot \mathbf{t}_0 \end{pmatrix}. \end{aligned}$$

There are four unknowns (l, w, h and ζ) and four independent equations (recall that N_P and N_A are of rank two), we can then solve the equations and reconstruct the cuboid.

The same solution can be reached from another perspective. If we consider O, P and A as origins of three world coordinate systems, we can use the method from Section 5.1 to compute the translations of the camera according to these world coordinate systems. Let the translations be $\zeta \mathbf{t}_O,$

$\eta\mathbf{t}_P$ and $\xi\mathbf{t}_A$, in which $\mathbf{t}_O, \mathbf{t}_P$ and \mathbf{t}_A are unit vectors, we have

$$\begin{aligned} \zeta\mathbf{t}_O + \eta\mathbf{t}_P &= \begin{pmatrix} l \\ w \\ h \end{pmatrix} \\ \zeta\mathbf{t}_O + \xi\mathbf{t}_A &= \begin{pmatrix} l \\ 0 \\ 0 \end{pmatrix} \\ \xi\mathbf{t}_A + \eta\mathbf{t}_P &= \begin{pmatrix} 0 \\ w \\ h \end{pmatrix}. \end{aligned}$$

Note that these \mathbf{t} 's are the positions of the camera in each of the coordinate systems and have different meanings from those in Equation (2). There are six unknowns and six independent equations and so we can solve for the unknowns. Note that ζ , η and ξ are the distances from the camera to the corners. As we have pointed out in Section 5.1, these distances cannot be recovered from a single corner.

5.3 Texture generation

The primary goal of this work is to obtain the locations and sizes of light sources, such as ceiling lights and windows, in an indoor scene. This is done by creating the texture of each plane (wall) of a cuboid. In general, the pixel values of light sources are uniformly bigger than those of other part of scene. Thus we can use a simple image segmentation algorithm for each textured plane to extract the light sources. Because each plane has been calibrated (we have reconstructed the cuboid), we immediately know the positions and shapes of the extracted light sources.

In more details, we first choose a corner to act as the origin of the world coordinate system and let the image containing this corner be the reference image I_0 . We then compute the pose of the camera at this position and compute the homographies H_{0i} for each image I_i . A plane of the cuboid is projected toward the set of images using the pose information and the homographies. The texture of each plane is thus obtained from the pixel values of correspondent image points. For instance, let's consider the wall with a window in Figure 4(a). We already know its height h and width w , since we have reconstructed the room. We *digitize* this world plane into an $H \times W$ -point array, of which H is any whole number (*e.g.* 256) and W is $\lfloor H \times w/h \rfloor$. We also know the world coordinates of the corners \mathbf{O} , \mathbf{A} and \mathbf{B} of the plane from the reconstruction and the camera pose information. Thus we can write down the world coordinates of a point (i, j) of the array:

$$\mathbf{O} + \frac{i}{H}h\mathbf{i} + \frac{j}{W}w\mathbf{j}$$

where $\mathbf{i} = (\mathbf{A} - \mathbf{O})/|\mathbf{A} - \mathbf{O}|$ and $\mathbf{j} = (\mathbf{B} - \mathbf{O})/|\mathbf{B} - \mathbf{O}|$. This point is projected to image I_0 using Equation (3) and then image I_i using H_{0i} . If the projection falls into an image, bilinear interpolation is used to retrieve the pixel value from this image. A weighted average of all these pixel values gives the texture value of point (i, j) (Figure 4(b)).

6 A working example

Figure 5 shows the block diagram of the reconstruction system. We here demonstrate how to create a lighting model from a set of images of an office using cuboid structure.

We took ninety pictures of the office using a Sony pan-tilt camera. Figure 6(a) shows eight of them. The image mosaicking algorithm takes the image set as input and automatically computes

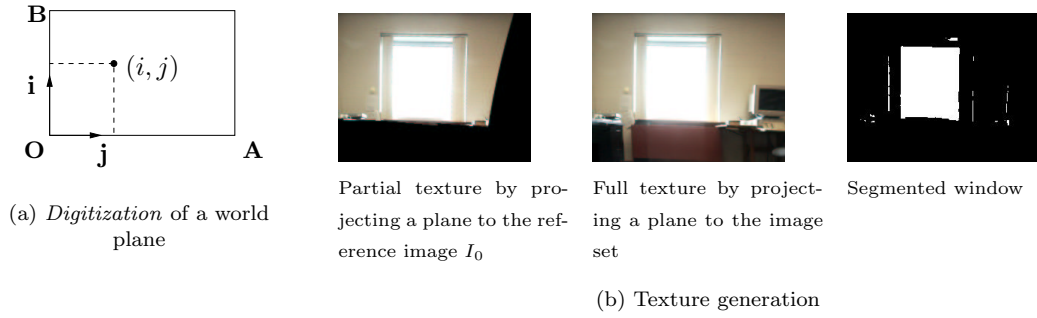


Figure 4: Texture generation

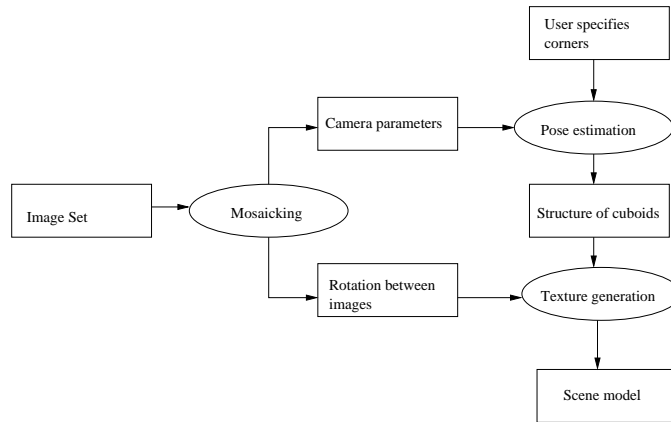
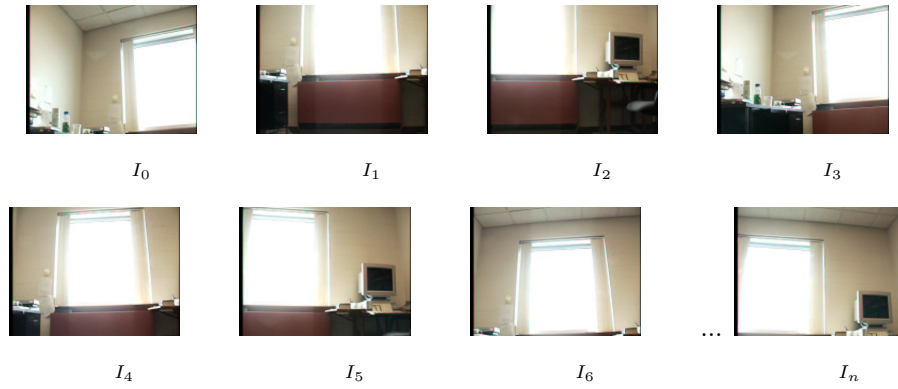


Figure 5: Block diagram of the interactive reconstruction process.

the homographies between images and calibrates the camera. The mosaicking process is to compute the homographies between images and it is not necessary to create the image mosaic explicitly. However, an image mosaic is a good way to visually evaluate the homographies. Figure 6(c) shows an image mosaic created from Figure 6(a). The user picks three images that contain three corners of the room and specifies the three corners through a GUI program (Figure 6(b)). The structure of the office is recovered from these three corners. Texture of walls of interest (in this case, the wall with the window and the ceiling with two lights) is generated from the structure and the set of images (Figure 4(b)). The sizes and locations of light sources are extracted from the texture. A graphic model can then be built.

7 Conclusion

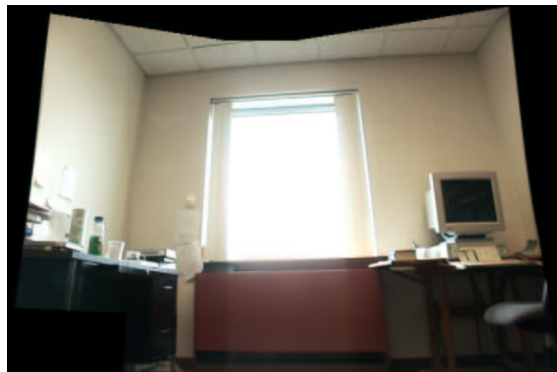
In this report, we showed a method for reconstructing indoor scenes from image mosaics. The method exploits indoor scene regularities by using cuboid structures. It uses a new algorithm to compute the camera pose and cuboid structure from rectangular corners. The advantage of the method is that it only requires minimum human interaction. Future work includes extending the method to more complex scenes, specifically, looking for methods for breaking a scene into cuboids easily and combining the reconstruction of individual cuboids together.



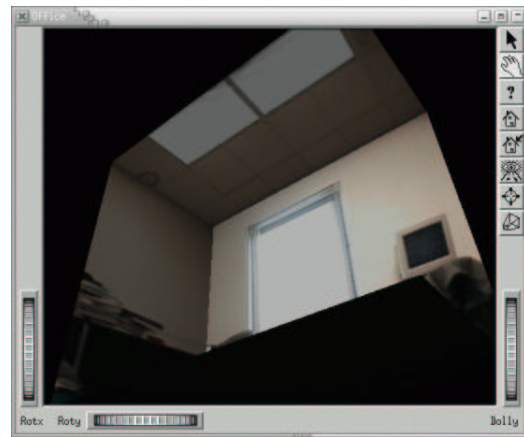
(a) Some of the images used to create the mosaics



(b) Specifying three corners of an office



(c) An image mosaic of an office scene



(d) A part of the reconstructed 3D model

Figure 6: Reconstruction of a room scene

References

- [Chen *et al.*, 1999] Chu-Song Chen, Chi-Kuo Yu, and Yi-Ping Hung, “New Calibration-free Approach for Augmented Reality BAsed on Parameterized Cuboid Structure,” In *ICCV'99*, pages 30–37, 1999.
- [Criminisi *et al.*, 2000] A. Criminisi, I. Reid, and Z. Zisserman, “Single View Metrology,” *International Journal of Computer Vision*, 40(2):123–148, 2000.
- [de Agapito *et al.*, 1999] Lourdes de Agapito, Richard I. Hartley, and Eric Hayman, “Linear calibration of a rotating and zooming camera,” In *CVPR'99*, pages 15–21, 1999.
- [Hartley, 1997] Richard Hartley, “Self-calibration of Stationary Cameras,” *International Journal of Computer Vision*, 22(1):5–23, February 1997.
- [Hu *et al.*, 2001] Bo Hu, Chris Brown, and A. Choi, “Acquiring An Environment Map Through Image Mosaicking,” Technical report, Computer Science Department, University of Rochester, 2001.
- [Kanatani, 1988] Ken-Ichi Kanatani, “Constraints on Length and Angle,” *CVGIP*, 41:28–42, 1988.
- [Liu *et al.*, 1990] Yuncai Liu, Thomas S. Huang, and O. D. Faugeras, “Determination of Camera Location From 2D To 3D Line and Point Correspondences,” *IEEE Transactions on PAMI*, 12(1):28–37, January 1990.
- [Shum and Szeliski, 2000] Heung-Yeung Shum and Richard Szeliski, “Systems and experiment paper: construction of panoramic image mosaics with global and local alignment,” *International Journal of Computer Vision*, 36(2):101–130, 2000.
- [Szeliski, 1996] R. Szeliski, “Video Mosaics for Virtual Environments,” *IEEE Computer Graphics and Applications*, 16, March 1996.
- [Wilczkowiak *et al.*, 2001] Marta Wilczkowiak, Edmond Boyer, and Peter Sturm, “Camera Calibration and 3D Reconstruction from Single Images Using Parallelepipeds,” In *ICCV'2001*, volume 1, pages 142–148, 2001.
- [Wu *et al.*, 1994] Yuyan Wu, S. Sitharama Iyengar, Ramesh Jain, and Santanu Bose, “A New Generalized Computational Framework for Finding Object Orientation Using Perspective Trihedral Angle Constraint,” *IEEE Transactions on PAMI*, 16(10):961–975, October 1994.