[1] Semiautomatic 3-D model extraction from uncalibrated 2-D camera views

Shawn Becker and V. Michael Bove, Jr.

{sbeck, vmb}@media.mit.edu

MIT Media Laboratory

20 Ames Street Cambridge MA 02139 USA

ABSTRACT

Scenes that contain every-day man-made objects often possess sets of parallel lines and orthogonal planes, the projective features of which possess enough structural information to constrain possible scene element geometries as well as a camera's intrinsic and extrinsic parameters.

In particular, in a scene with three mutually orthogonal sets of parallel lines, detection of the corresponding three vanishing points of the imaged lines allows us to determine the camera's image-relative principal point and effective focal length. In this paper we introduce a new technique to solve for radial and decentering lens distortion directly from the results of vanishing point estimation, thus precluding the need for special calibration templates. This is accomplished by using an iterative method to solve for the parameters that minimize vanishing point dispersion. Dispersion here is measured as covariance of vanishing point estimation error projected on the Gaussian sphere whose origin is the estimated center of projection.

Having found a complete model for each camera's intrinsic parameters, corresponding points are used in the relative orientation technique to determine the camera's extrinsic parameters as well as point-wise structure. Surfaces inherit planar geometry and extent from manually identified coplanar lines and points. View independent textures are created for each surface by finding the 2-D homographic texture transformation which corrects for planar perspective foreshortening. We utilize the local Jacobian of this transformation in two important ways: to prevent aliasing in the plane's texture space and to merge correctly texture data arising from varying sampling resolutions in multiple views.

1 Introduction

As real-time scene-rendering capabilities become more affordable and widespread, it becomes reasonable to consider the use of three-dimensional models of real scenes in lieu of two-dimensional signal-based image repre-

[1] as seen in *Proceedings SPIE Visual Data Exploration and Analysis II*, Vol. 2410, pp. 447-461, San Jose, California, February 8-10, 1995.

sentations. Handling digital video in such a fashion can provide both compression advantages and the ability to support interactive or personalized display of visual information.[1] The eventual goal of the research described in this paper is to develop a coding method for digital video based on the extraction of 3-D models where appropriate and possible.

As a special case, we consider extracting a textured polygon model of an enclosed space given a small number of partially overlapping snapshots from one or more ordinary handheld cameras.

The fact that we would like to be able to handle snapshots from random locations rather than a continuous video stream rules out direct spatiotemporal[7] and optical flow[8] approaches which assume infinitesimally small changes in the camera's extrinsic parameters (*i.e.* the six terms describing the camera's scene-relative rotation and translation). For the same reasons, recursive computational frameworks which use extended or iterated extended Kalman filters (EKF or IEKF)[10,11] cannot be used. Also, the fact that we are working with indoor scenes rules out *factorization*[9] methods specialized for the orthographic case.

Because we want to use ordinary handheld cameras, it would be beneficial to use a framework that either avoids the necessity of knowing the camera's intrinsic parameters or (even more desirable) allows us to estimate these parameters directly. In this paper we introduce a new approach that solves all of the camera's intrinsic parameters including lens distortion without requiring any special calibration patterns, just a scene with three-point perspective. This is done using a variation of the analytical plumb-line method introduced by Brown,[15] where instead of straightening points in a curve, the algorithm strives to make image lines of a set of parallel scene edges intersect at a single point.

Camera and point feature positions are determined using Horn's relative orientation method[4] given point correspondences in overlapping images. Coplanar lines and points are then manually identified and used to fit planar geometry and polygonal extent for planar scene surfaces. View independent textures are created for each surface by finding the 2-D texture transformation which corrects for planar perspective foreshortening. We utilize the local Jacobian of this transformation in two important ways: to prevent aliasing in the plane's texture space and to merge correctly texture data arising from varying sampling resolutions in multiple views.

The rest of the paper is as follows. Section 2 introduces our new camera calibration method and reviews how these calibration results can be used to unwarp geometrically distorted images. Section 3 describes minimum squared error estimation of scene-relative camera pose and point-wise position given point correspondences. Section 4 discusses texture rectification using an 8-parameter homographic transform. Here we also show how the Jacobian of this transform is used to avoid aliasing and to correctly merge texture samples from multiple perspectives and resolutions. Various results using real images are then shown in section 5.

2 Camera calibration

In this section we first review the projection equation which transforms world points into image pixels. Issues of line representation are then discussed, followed by the introduction of a new method of lens distortion correction that aims to minimize vanishing point dispersion. To complete intrinsic camera calibration, we then discuss how the vanishing points of three mutually orthogonal sets of lines are used to estimate center of projection.

2.1 Camera model

The transformation from world coordinates $[X_w, Y_w, Z_w]^T$ to measured image coordinates $[x_m, y_m]^T$ can be treated more simply as a composition of several transformations. The *rigid body transformation* brings world

coordinates into camera-relative coordinates $[X_c, Y_c, Z_c]^T$

$$\left[\ X_c, Y_c, Z_c, 1\ \right]^T = \mathbf{B}\left[\ X_w, Y_w, Z_w, 1\ \right]^T \tag{1}$$

where $\mathbf{B}$ is a 4×4 matrix made up of the orthonormal 3×3 rotation matrix $\mathbf{R}$ and the 3×1 translation vector $\mathbf{t}$.

The *perspective transformation* is the non-linear mapping, $\mathcal{P}$, which uses the camera-relative center of projection, $[C_x C_y C_z]$ with positive C_z toward the viewer, to project camera-relative scene position into homogeneous camera-relative image coordinates $[x_p, y_p, w_p]^T$. It uses the 4×4 matrix,

$$\begin{bmatrix} 1 & 0 & \beta C_x & 0 \\ 0 & 1 & \beta C_y & 0 \\ 0 & 0 & \beta & 1 \end{bmatrix} \tag{2}$$

where inverse focal length is $\beta = -1/C_z$, followed by normalization by the homogeneous term.

The *view transformation* uses the 3×3 matrix $\mathbf{V}$ which maps projected camera-relative coordinates $[x_p, y_p, 1]^T$ into ideal undistorted pixel coordinates $[x_u, y_u, 1]^T$

The *lens distortion equation* describes the non-linear relationship between ideal undistorted pixel coordinates $[x_u, y_u]^T$ and actually measured pixel coordinates $[x_m, y_m]^T$

$$\begin{aligned} x_u &= x_m + \bar{x}\left(K_1 r^2 + K_2 r^4 + K_3 r^6\right) + \left[P_1\left(r^2 + 2\bar{x}^2\right) + 2P_2\bar{x}\bar{y}\right]\left[1 + P_3 r^2\right] \\ y_u &= y_m + \bar{y}\left(K_1 r^2 + K_2 r^4 + K_3 r^6\right) + \left[P_2\left(r^2 + 2\bar{y}^2\right) + 2P_1\bar{x}\bar{y}\right]\left[1 + P_3 r^2\right] \end{aligned} \tag{3}$$

where

$$\begin{aligned} \bar{x} &= x_m - PP_x \\ \bar{y} &= y_m - PP_y \\ r &= \sqrt{\bar{x}^2 - \bar{y}^2} \\ \begin{bmatrix} Q_x \\ Q_y \\ 1 \end{bmatrix} &= \mathbf{V}\begin{bmatrix} C_x \\ C_y \\ 1 \end{bmatrix} \end{aligned} \tag{4}$$

and where K_1, K_2 and K_3 are radial distortion coefficients, P_1, P_2 and P_3 are the decentering distortion coefficients and $[Q_x, Q_y]^T$ are the pixel coordinates of the camera's principal point. Since the equations (3) output an undistorted point for a measured point already subjected to lens distortion, we might describe this as an *undistortion* mapping which is functionally of the form

$$\mathbf{p}_u = \mathcal{U}(\mathbf{p}_m) \tag{5}$$

where $\mathbf{p}_u = [x_u, y_u]^T$ and $\mathbf{p}_m = [x_m, y_m]^T$. The inverse mapping which would be used to synthesize point-wise lens distortion is functionally of the form

$$\mathbf{p}_m = \mathcal{U}^{-1}(\mathbf{p}_u) = \mathcal{D}(\mathbf{p}_u) \tag{6}$$

where the distorted coordinate $\mathbf{p}_m$ for a particular undistorted coordinate $\mathbf{p}_u$ is given. The entire *projection equation* can now be described as

$$\mathbf{p}_m = \mathcal{D}\mathbf{V}\mathcal{P}\mathbf{B}\mathbf{p}_w \tag{7}$$

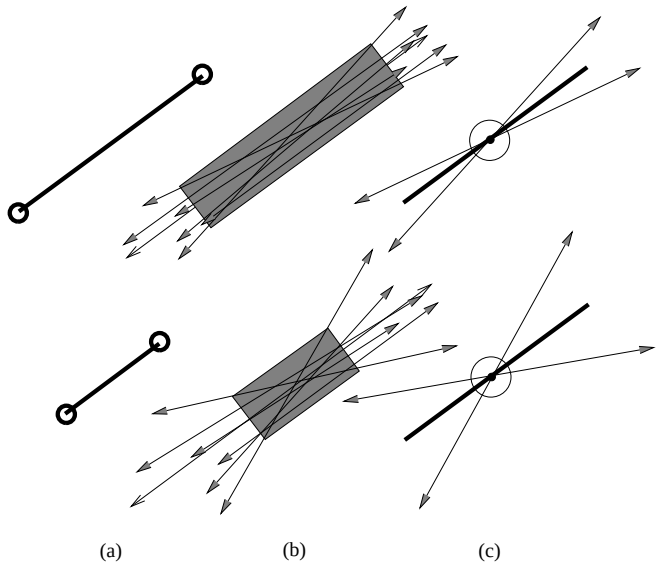

Figure 1: Inverse relationship between segment length and angle variance: (a) results of segment detection, (b) a set of true lines, any one of which could have given rise the segment observed under noisy conditions, (c) confidence intervals of a probablistic representation that decouples uncertainty in center and angle.

2.2 Probabilistic representations for uncertain geometry

2.2.1 Image line representation

An image is produced by applying the projection equation (7) to a scene containing three-dimensional edges to get an image of line segments. We represent line segments in the three-parameter form

$$y = (x - e_x)\tan(e_\theta) + e_y \tag{8}$$

where e_θ, e_x and e_y describe the segment's direction angle and center point. Since line detection is noisy, it is appropriate to model the parameters of the line segment probabilistically as the random vector $\mathbf{e} = [e_\theta, e_x, e_y]^T$ which has a normal distribution whose mean value is obtained from the observed line segment and whose diagonal covariance is $\Lambda_e = diag(\sigma_\theta^2, \sigma_x^2, \sigma_y^2)$. As illustrated in Figure 1 angle variance, σ_θ^2 is inversely proportional to segment length. Center point variances σ_x^2 and σ_y^2 are set inversely proportional to the center point's distance from an approximated principal point, since pixels at the principal point project a larger solid angle than those at the periphery.

2.2.2 Sphere point representation

To do this we must find a representation for line segments and line intersections which is insensitive to the camera-relative orientation of scene edges. Referring to Figure 2, let $E_{i,j}$ denote the j-th edge belonging to the i'th set of parallel scene edges whose scene relative direction is $\hat{\mathbf{d}}_i$. Also let $\mathbf{e}_{i,j}$ be its observed image segment with mean parameters $[e_{\theta_{i,j}}, e_{x_{i,j}}, e_{y_{i,j}}]$. If $\mathbf{Q} = [Q_x, Q_y, Q_z]^T$ is an arbitrary image-relative center of projection, then under conditions of noiseless line detection and zero lens distortion, $E_{i,j}$ will lie in the plane created by $\mathbf{e}_{i,j}$

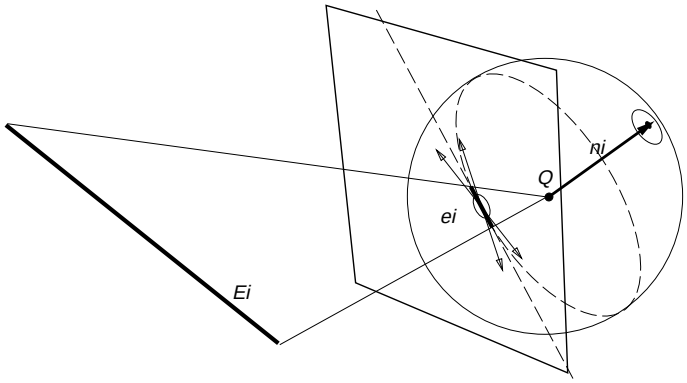

Figure 2: Using the sphere-point to represent a probabilistic image segment

and **C**. This plane contains **C** and has unit normal

$$\hat{\mathbf{n}}_{i,j} = \frac{\mathbf{n}_{i,j}}{\|\mathbf{n}_{i,j}\|} \tag{9}$$

where

$$\mathbf{n}_{i,j} = \begin{bmatrix} e_{x_{i,j}} - Q_x \\ e_{y_{i,j}} - Q_y \\ -Q_z \end{bmatrix} \times \begin{bmatrix} \cos e_{\theta_{i,j}} \\ \sin e_{\theta_{i,j}} \\ 0 \end{bmatrix} \tag{10}$$

If we consider a unit Gaussian sphere whose origin is the arbitrary center of projection, the segment $\mathbf{e}_{i,j}$ can be represented as a *sphere point* whose mean coordinates are defined by $\hat{\mathbf{n}}_{i,j}$.

To find the covariance of the 3×1 random vector $\hat{\mathbf{n}}_{i,j}$ we use equations (9) and (10) to describe the function g where $\mathbf{n} = g(\mathbf{e})$. We now use the fact[13] that for any sufficiently smooth function g where $\mathbf{n} = g(\mathbf{e})$, the covariance Λ_n of random vector $\mathbf{n}$ at $E[\mathbf{n}] = E[g(\mathbf{e})] = g(E[\mathbf{e}])$ can be approximated as $\Lambda_n = J\Lambda_e J^T$ where J is the 3×3 Jacobian matrix of the form $j_{m,n} = \partial n_m / \partial e_n$ evaluated at $\mathbf{n}$.

2.2.3 Intersecting a pair of lines

This probabilistic representation for image segments is particularly useful for describing intersections between a pair of lines. Suppose we have two scene edges $E_{i,1}$ and $E_{i,2}$ with the same unit scene direction $\hat{\mathbf{d}}_i$ and corresponding image segments $\mathbf{e}_{i,1}$ and $\mathbf{e}_{i,2}$ and subsequent sphere points $\hat{\mathbf{n}}_{i,1}$ and $\hat{\mathbf{n}}_{i,2}$ (see Figure 3). We see here that the plane which contains both $\hat{\mathbf{n}}_{i,1}$ and $\hat{\mathbf{n}}_{i,2}$ has the normal vector whose direction is $\hat{\mathbf{d}}_i$, the image-relative direction of both scene lines. Also note that the vector between point V_i, the vanishing point for the i-th set of parallel lines and **Q**, the center of projection, has the edge direction $\hat{\mathbf{d}}_i$.

The important strength of the sphere point representation is that line intersections have no preferred (or unpreferred) direction. In particular, if a pair of image segments are parallel but not collinear, no image point exists that describes the intersection. In the Gaussian sphere domain, however, this poses no special problem, and thus is well suited for vanishing point estimation under perspective as well as orthographic projection where parallel lines always project to parallel lines.

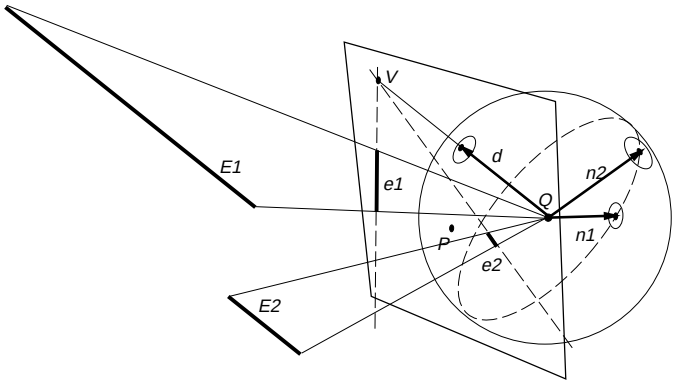

Figure 3: Using sphere-points to represent the intersection between a pair of lines

2.2.4 Estimating vanishing point direction from many lines

Now suppose there are a total of N_i line segments belonging to the i-th direction. If no $\mathbf{e}_{i,j}$ are collinear as N_i increases, the subsequent set of positive and negative pairs of sphere points $\pm\mathbf{n}_{i,j}$ will create a symmetric great circle on the Gaussian sphere. The plane that minimizes the square error fit to the set of sphere points will have normal vector $\hat{\mathbf{d}}_i$ and will contain the center of projection C.

Thus the vanishing point direction $\hat{\mathbf{d}}_i$ can be estimated from the aggregate distribution of probabilistic sphere points with 3-D means $\hat{\mathbf{n}}_{i,j}$ and covariances $\Lambda_{n_{i,j}}$. Since the mean of this set is known to be the center of projection the covariance of the aggregate distribution can be approximated as the weighted average[13] of $\hat{\mathbf{n}}_{i,j}\hat{\mathbf{n}}_{i,j}^T$

$$\Lambda_{d_i} = \sum_{j=1}^{N_i} \hat{\mathbf{n}}_{i,j}\Lambda_{n_{i,j}}^{-1}\hat{\mathbf{n}}_{i,j}^T \left[\sum_{i=1}^{N}\Lambda_{n_i}^{-1}\right]^{-1} \tag{11}$$

The desired vanishing point direction, $\hat{\mathbf{d}}_i$, is estimated as

$$\hat{\mathbf{d}}_i = \phi_{i,3} \tag{12}$$

where $\phi_{i,3}$ is the eigenvector associated with the minimum eigenvalue, $\lambda_{i,3}$, of Λ_{d_i}. Proof: if all sphere points $\hat{\mathbf{n}}_{i,j}$ for $j = [1, N_i]$ lie exactly in some plane (i.e. no line detection noise and no lens distortion) and are not colinear then the rank of the sample covariance matrix Λ_{d_i} will be 2 and the smallest eigenvalue will be zero. The positive and negative senses of the associated eigenvector is the plane normal since the first two eigenvectors (associated with the non-zero eigenvalues) lie in the plane and because of the condition of orthogonality of the eigenvectors.

2.3 Vanishing points and lens distortion

As explained by Brown,[15] in the absence of distortion, the central projection of a straight line is itself a straight line. Most approaches to lens distortion attempt to do so by minimizing curvature along curves or among points which should be straight. We instead make the observation that as shown in Figure 4(a) in the absence of distortion, the central projection of a set of straight parallel lines is itself a set of straight lines that intersect at a distinct vanishing point. A set of N non-overlapping segments results in $K = C_2^N$ possible intersection points. Without distortion, all K candidate vanishing points will be identical. Under lens distortion, however, the projection of this set of lines yields a set of curved lines that if broken into piecewise linear segments (as done by some line detection techniques), the extension of these segments will not result in a common intersection point. Instead, the set of K candidate vanishing points will be dispersed as shown in Figure 4(b).

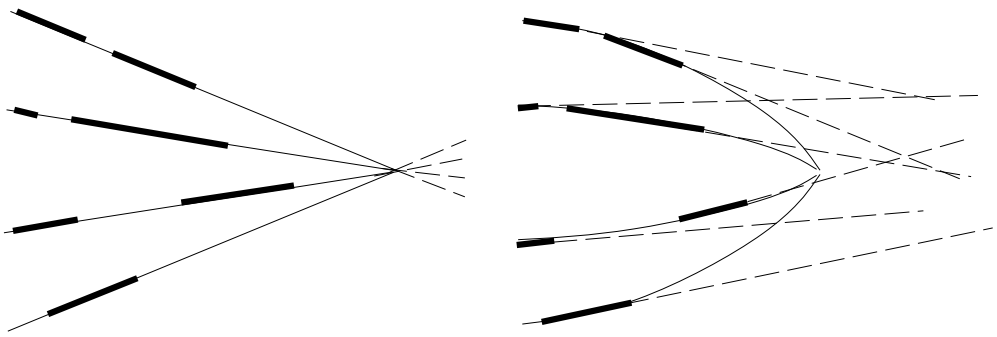

Figure 4: Vanishing point of a set of parallel scene lines (a) without lens distortion is compact and distinct (b) with lens distortion is dispersed and "confused".

Resuming the Section 2.2.4 discussion of estimating the vanishing direction of the i-th set of lines, as line detection noise and lens distortion are introduced, the set of sphere points become less constrained to a plane and the minimum eigenvalue, $\lambda_{i,3}$, becomes non-zero. Each non-overlapping pair of segments contributes a candidate vanishing point. Similarly as shown in Figure 3 each pair of sphere points contributes a candidate vanishing point direction. Just as noise and distortion confuses or disperses the image vanishing point, it also disperses the candidate vanishing point directions and consequently widens the minimum thickness of the aggregate distribution. Thus $\lambda_{i,3}$ can be interpreted as a measure of vanishing point dispersion.

This suggests that $\lambda_{i,3}$ can be used as a measure of the amount of distortion introduced by the projection equation (7). To correct for lens distortion, our goal then, is to find the distortion parameters that minimize $\lambda_{i,3}$. Using equations (5), (9) and (12) we define the observation function

$$f(\hat{\mathbf{e}}_{i,j}, \hat{\mathbf{d}}_i; K_1, K_2, K_3, P_1, P_2, P_3) = 0 \tag{13}$$

If M denotes the total number of parallel sets of lines and N_i denotes the number of lines detected in the i-th set, the total number of equations of the form (13) will amount to $N = \sum_{i=1}^{M} N_i$. Letting $\mathbf{x} = [K_1, K_2, K_3, P_1, P_2, P_3]$ our task is to simultaneously solve N non-linear systems of 6 variables. That is to find the $\mathbf{x}$ for which

$$f_n(\mathbf{x}) = 0 \tag{14}$$

is true for all $n = [1, N]$. This is typically done with a series of linear regression equations using a least squares method to refine some initial estimate. The Taylor series expansion of equation (14) is

$$f_n(\mathbf{x} + \boldsymbol{\Delta}_x) = f_n(\mathbf{x}) + \sum_{k=1}^{6} \frac{\partial f_n}{\partial x_k} \Delta_{x_k} + \mathcal{O}\left(\boldsymbol{\Delta}_x^2\right) + \cdots \tag{15}$$

By ignoring 2nd and higher order terms and asserting that the solution exists at $\mathbf{x} + \boldsymbol{\Delta}_x$ we have

$$f_n(\mathbf{x} + \boldsymbol{\Delta}_x) = 0 \tag{16}$$

then (15) becomes

$$-f_n(\mathbf{x}) = \sum_{k=1}^{6} \frac{\partial f_n}{\partial x_k} \Delta_{x_k} \tag{17}$$

$$\mathbf{e} = \mathbf{A}\boldsymbol{\Delta}_x \tag{18}$$

In order to solve for $\mathbf{\Delta}_x$ we use singular value decomposition (SVD) to do linear least squares fitting of parametric data.[12] The solution is then substituted into (16) to adjust the parameter vector $\mathbf{x}$

$$\mathbf{x}^{new} = \mathbf{x}^{old} + \mathbf{\Delta}_x \tag{19}$$

The new parameter vector $\mathbf{x}$ is then used to update all $\hat{\mathbf{d}}_n$ for $n = [1, N]$ and the $N \times 1$ dispersion vector whose elements are $\lambda_{n,3}$. This process is then iterated until the dispersion vector reaches convergence.

2.3.1 Applying distortion estimation results to an image

Once the distortion parameters have been estimated, equation (5) can be used to correct point-wise distortions for all image segments. However, correcting the entire distorted image given fixed distortion parameters requires inverting this equation to find distorted image coordinates for a desired pixel in the undistorted output image.

Here we again use the iterative root solving method of simultaneous non-linear equations. To get the initial estimate for the distortion coordinates $\mathbf{p}_m$ given an undistorted point $\mathbf{p}_u$ we rewrite equation (5) as a displacement function

$$\begin{aligned} \mathbf{p}_u &= \mathbf{p}_m + (\mathcal{U}(\mathbf{p}_m) - \mathbf{p}_m) \\ &= \mathbf{p}_m + u(\mathbf{p}_m) \end{aligned} \tag{20}$$

if we assume that the displacement function u is locally smooth then we can say that

$$u(\mathbf{p}_u) \approx u(\mathbf{p}_m) \tag{21}$$

therefore we can make the approximation

$$\mathbf{p}_m \approx \mathbf{p}_u - u(\mathbf{p}_u) \tag{22}$$

Now to invert (5) for a particular undistorted point $\mathbf{p}_u$ we define the error function

$$f(\mathbf{p}_m) = \mathcal{U}(\mathbf{p}_m) - \mathbf{p}_u = 0 \tag{23}$$

initialize $\hat{\mathbf{p}}_m$ to (22) and use linear regression to iteratively converge on a solution.

2.3.2 Stochastic image unwarping

Let $\mathbf{R_u}$ denote the square image region covered by one pixel in the undistorted image with center $\mathbf{p}_u$. Using the iterative approach shown in section 2.3.1, the corresponding center coordinate $\mathbf{p}_m$ of the region $\mathbf{R_m}$ in the distorted image that maps onto the $\mathbf{R_u}$ region in the undistorted image. Letting Λ_u and Λ_m denote the covariances of these two regions and linearizing the mapping at the final estimate of we find that $\mathbf{\Lambda}_m = \mathbf{J}^{-1}\mathbf{\Lambda}_u\mathbf{J}^{-T}$ where $\mathbf{J}$ is the 2×2 Jacobian matrix computed at each iteration in the linear regression used to solve (23) and is of the form $j_{a,b} = \partial p_{u,a}/\partial p_{m,b}$ evaluated at $\mathbf{p}_u$.

The pixel area of the two regions can be approximated as $A_u = det(\mathbf{\Lambda}_u)^{1/2}$ and $A_m = det(\mathbf{\Lambda}_m)^{1/2}$. When $A_m > A_u$, the undistorted pixel maps from a distorted region of relatively greater coverage. Thus, to avoid aliasing we average a set of $\lceil A_m/A_u \rceil$ bilinearly interpolated normally distributed intensity samples in $\mathbf{R_m}$. When $A_m < A_u$ a single bilinearly interpolated intensity at position $\hat{\mathbf{p}}_m$ is sufficient.

2.4 Estimating center of projection

The center of projection is completely described by the principal point, the point in the image plane nearest to the center of projection, and the principal distance (or effective focal length) which is that minimum distance.

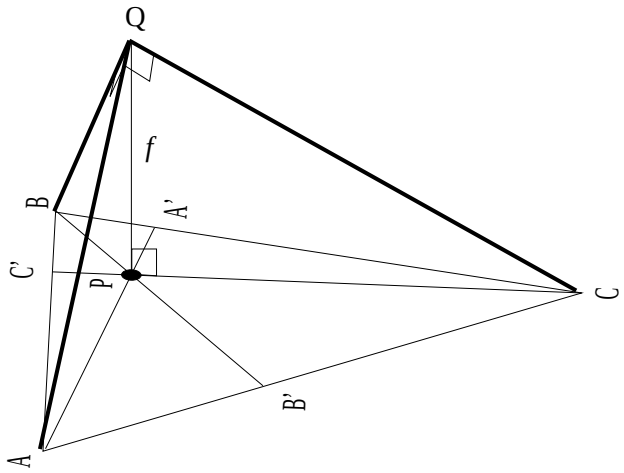

Figure 5: Estimating principal point and principal distance from three vanishing points of mutually orthogonal parallel lines

Letting **A**, **B** and **C** denote the image-relative coordinates of three vanishing points formed by three mutually orthogonal sets of parallel lines, the principal point **P** is the orthocenter of the triangle **ABC** (see Figure 5). The principal distance (or effective focal length) f is the distance from the principal point, **P**, to the center of projection, **Q** that makes the 4 points **ABCQ** into a right tetrahedron, where the lines **AQ**, **BQ** and **CQ** are mutually orthogonal and where **PQ** is orthogonal to the image plane containing **A**,**B**,**C** and **P**.

3 Structure and Motion

We now discuss estimation of scene structure and camera motion given the calibration information already obtained in section 2. For the more general problem of solving structure and motion given only point correspondences with unknown rotation refer to Horn's method of relative orientation.[4] To relate camera-relative to scene-relative structure and motion refer to closed form solutions of absolute orientation.[5,6]

3.1 Absolute rotation

Suppose we have already estimated three image-relative vanishing point unit directions $\hat{\mathbf{d}}_A$, $\hat{\mathbf{d}}_B$, $\hat{\mathbf{d}}_C$, lens distortion and the center of projection **Q** as outlined in section 2. We now define the matrix $\mathbf{L}_C$ to be the camera-relative coordinate frame of lines whose columns are the three vanishing point unit directions. Similarly, we use the known world-relative unit line directions to define the world-relative coordinate frame $\mathbf{L}_W$. The world-relative camera rotation matrix $\mathbf{C}_W$ that brings $\mathbf{L}_C$ and $\mathbf{L}_W$ into correspondence is

$$\mathbf{C}_W = \mathbf{L}_W \mathbf{L}_C^{-1} = \mathbf{R}^{-1} \tag{24}$$

As shown, $\mathbf{C}_W$ is also the inverse of that camera's rigid-body rotation matrix **R** from Equation 1.

3.2 Absolute translation and structure

Given known world-relative camera rotation of each view we now solve world-relative positions of each camera and of any corresponding point features visible in more than 1 frame. This is done using the known rotation case of Horn's relative orientation approach[4] to find the best-fit unit baseline vector between multiple cameras given $L \geq 2$ corresponding image points and known rotation. Once the baselines are found, the known world-relative positions of $K \geq 2$ points where $K \subseteq L$ are used to find the best-fit baseline magnitude for each camera. This is then used to bring all point and camera positions into world-relative scale thus solving for absolute camera motion and feature structure.

Here, we briefly discuss the two view case where $L \geq 2$ correlated features in the left and right cameras are respectively represented as the camera-relative unit vectors $\mathbf{r}_r$ and $\mathbf{r}_l$. Let $\mathbf{r'}_l$ be the right camera-relative direction of the left ray. This can be found since world-relative rotations for the left and right cameras (denoted by $\mathbf{C}_{W,l}$ and $\mathbf{C}_{W,r}$) already known.

$$\mathbf{r'}_l = \mathbf{C}_{W,r}^{-1}\mathbf{C}_{W,l}\mathbf{r}_l \tag{25}$$

Let unit vector $\mathbf{b}$ denote the right camera-relative direction of the difference between the left and right cameras' centers of projection. The best fit for the baseline is the eigenvector associated with the minimum eigenvector of the 3×3 matrix $\mathbf{C}$ where

$$\mathbf{C} = \sum_{i=1}^{L} w_i \mathbf{c}_i \mathbf{c}_i^T \tag{26}$$

where w_i is a scalar confidence measure for each point initially set to unity and $\mathbf{c}_i = \mathbf{r'}_{l,i} \times \mathbf{r}_{r,i}$. The baseline vector $\mathbf{b}$ may be iteratively refined by using the previous estimation to adjust the confidence measure for each sample using the equation

$$w_i = \frac{1}{\|\mathbf{c}_i\|^2(\gamma_i^2\sigma_{l,i}^2 + \alpha_i^2\sigma_{r,i}^2)} \tag{27}$$

where $\sigma_{l,i}^2$ and $\sigma_{r,i}^2$ are known positional uncertainties of the i-th feature in the left and right views. The values γ_i and α_i represent unscaled camera-relative distances to the scene features from the left and right cameras respectively and are defined as

$$\gamma_i = (\mathbf{b} \times \mathbf{r}_{r,i}) \cdot \mathbf{c}_i / \|\mathbf{c}_i\|^2 \quad \alpha_i = (\mathbf{b} \times \mathbf{r'}_{l,i}) \cdot \mathbf{c}_i / \|\mathbf{c}_i\|^2 \tag{28}$$

Note that the calculations for γ_i and α_i require division by $\|\mathbf{c}_i\|^2$. If this value is zero, then the two cameras may share the same position and the relative translation vector will be zero. This problem can be averted by checking the eigenvalues from the initial unit weighted calculation of $\mathbf{C}$. Zero magnitude translation between cameras can be identified by $\mathbf{C}$ having only rank 2 (that is having only 2 eigenvalues which are non-zero within machine precision).

Values γ_i and α_i, describe unscaled camera-relative distances for all L features visible in both images. These distance are *unscaled* in the sense that their values are based on $\mathbf{b}$ being a unit baseline vector. To bring the baseline and camera distances into correct world-relative scale, we find the best fit scale factor S that brings all features within proper distance of each other.

$$S = \sum_{i=2}^{K} \frac{\|\mathbf{p}_i - \mathbf{p}_{i-1}\|}{\|\alpha_i \mathbf{r}_{r,i} - \alpha \mathbf{r}_{r,i}\|} \tag{29}$$

where $\mathbf{p}_i$ are known world-relative positions of the i-th feature out of $K \geq 2$ points that also belong to the set of L point correspondences. The right camera-relative position of the left camera is now $S\mathbf{b}$. Similarly, right camera-relative point positions are now defined as non-unit vectors $S\alpha_i\mathbf{r}_{r,i}$.

The least squares estimate of the right camera's world-relative position, $\mathbf{o}_w$ is

$$\mathbf{o}_w = \mathbf{m}_w - \mathbf{C}_{W,r}\mathbf{m}_r \tag{30}$$

where $\mathbf{C}_{W,r}$ is the world-relative coordinate frame of the right camera found from equation (24) and

$$\mathbf{m}_w = \frac{1}{K}\sum_{i=1}^{K}\mathbf{p}_i \quad \mathbf{m}_r = \frac{1}{K}\sum_{i=1}^{K} S\mathbf{r}_{r,i} \tag{31}$$

where $\mathbf{m}_w$ and $\mathbf{m}_r$ are the world- and right camera-relative means of the K fixed features. Known world-relative position and rotation of the right camera can now be used to fix positions for the left camera and all the remaining L features.

This process is successively repeated for all C_2^N pairs of frames, where camera and feature positions are probablistically refined by updating covariance measures of uncertainty at each step. The final result provides the best fit solutions for the rigid-body transformation for each camera as well as world-relative positions of all features.

4 Textured planar surfaces

The scene model is a collection of textured planar surfaces each defined as a membership list of coplanar point and segment elements. Each surface thus inherits structure from member elements manually or automatically assigned to it. Geometric structure is found by using world-relative positions and directions of member points and segments to make a best fit plane. Texture structure is found by first "shrink wrapping" a closed polygon around all elements belonging to a particular surface which are visible in any view. These image regions are then sampled and merged into a coherent view independent surface texture representation. The following sections describe the sampling and merging of new texture data from multiple views with their differing resolutions due to distance and foreshortening and differing extents due to occlusions and framing.

4.1 View independent texture sampling

To preserve the highest resolution of textural information available in all views we rectify planar texture, thus effectively undoing perspective, while retaining a measure of confidence for each sample so that textures from multiple perspectives can be merged by weighted averaging. As described by Mann and Becker[2] the non-linear 2-D from-warping that embodies the 8 *pure parameters* of a planar patch under perspective projection[21] is of the form

$$\mathbf{f} = \frac{\mathbf{A}\mathbf{t} + \mathbf{b}}{\mathbf{c}^T\mathbf{t} + 1} \tag{32}$$

where $\mathbf{f}$ and $\mathbf{t}$ are 2-D pixel coordinates in the perspective warped film image and perspective corrected texture image. The 8 pure parameters are described in the 2×2 matrix $\mathbf{A}$, and the 2×1 vectors $\mathbf{b}$ and $\mathbf{c}$. This mapping can be used to warp any planar surface under one perspective view into any other perspective view. This solution comes from solving 4 pairs of linear equations of the form

$$f_x = \begin{bmatrix} t_x & t_y & 0 & 0 & 1 & 0 & -t_x f_x & -t_y f_x \end{bmatrix} \cdot \mathbf{P} \tag{33}$$

$$f_y = \begin{bmatrix} 0 & 0 & t_x & t_y & 0 & 1 & -t_x f_y & -t_y f_y \end{bmatrix} \cdot \mathbf{P} \tag{34}$$

where the 8 pure parameters are contained in $\mathbf{P} = [Axx, Axy, Ayx, Ayy, bx, by, cx, cy]^T$.

To rectify the texture which contributes to a given surface from a particular camera, we first find a set of four parametric texture coordinates $\mathbf{t} = [u, v]$ which bound the texture space of the surface with corresponding world-relative coordinates $\mathbf{w}$. These coordinates are then projected into image coordinates $\mathbf{f}$ using Equation 7. These 4 pairs of coordinates are then used in equations (33) and (34) to solve for the 8 warping parameters. Positions of all texture coordinates may now be found efficiently by separately doing linear interpolation of the 2 numerator terms and single denominator terms of (32) and then dividing.

One advantage of this approach is that the warping function allows us to directly calculate the Jacobian matrix $\mathbf{J} = \partial\mathbf{f}/\partial\mathbf{t}$ which is used to define the covariance of the region for stochastic sampling where, assuming unit Λ_t, $\Lambda_f = \mathbf{J}\mathbf{J}^T$. As shown above for lens distortion correction in section 2.3.2 this lets us avoid aliasing in the undistorted texture image.

The above approach is also an important aid in merging data from multiple sources. Consider the portion of a surface whose textures in the image appear shrunken due to perspective distortion. When using stochastic sampling, intensities in the corresponding portion of the texture image are obtained by interpolation. In this case, interpolation can be modeled as zero padded up sampling followed by convolution with a Gaussian filter whose covariance is Λ_f^{-1} in the texture image domain (recall that the Fourier transform of a 2-D Gaussian is itself a 2-D Gaussian with inverse covariance[22]). Thus Λ_f^{-1} can be thought of as the error covariance $\mathbf{\Lambda}_a$ of a measurement of intensity $\mathbf{a}$. Large $\mathbf{\Lambda}_a$ indicates lower frequencies in texture space and thus has higher uncertainty in the intensity measurement.

Now suppose we have N different views of the same surface and we use the above approach to unwarp portions of the N images into N texture images. Further suppose, that at each position in a texture image we store intensity $\mathbf{a}$ as well as the 3 unique terms of Λ_a (the 2×2 covariance matrix is symmetric). If we have N different measurements of the same phenomenon (i.e. intensity, $\mathbf{a}$), where each measurement $\mathbf{a}_i, i = [1, N]$ has been subject to additive zero mean Gaussian white noise with covariance $\mathbf{\Lambda}_{a_i}$ the maximum likelihood estimate of the phenomenon is a Gaussian distribution with error covariance

$$\mathbf{\Lambda}_a = \left[\sum_{i=1}^{N} {\mathbf{\Lambda}_{a_i}}^{-1}\right]^{-1} \tag{35}$$

and mean

$$\hat{\mathbf{a}} = \mathbf{\Lambda}_a \sum_{i=1}^{N} \mathbf{a}_i \mathbf{\Lambda}_a^{-1} \tag{36}$$

Since both (35) and (36) use $\mathbf{\Lambda}_a^{-1}$ it makes sense instead to save the three unique terms of $\mathbf{\Lambda}_f$. To handle untextured regions caused by viewport framing and occlusions, the intensity measurement $\mathbf{a}$ actually is a 4×1 vector describing color and alpha.

5 Results

To detect image line segments we use the RobotVis image processing software developed by INRIA[19] which uses the Canny edge detection algorithm.[18] This software takes a user defined gradient magnitude threshold and fits line segments subject to a breaking angle and minimum edge length.

In each image line segments are first manually grouped into sets of parallel and sets of coplanar segments. Corresponding line directions and planes among views are specified. Three mutually orthogonal line directions are then selected and the distance between at least one pair of lines is provided.

The algorithm as described in sections 2, 3 and 4 are then used to solve internal camera parameters, camera motion, scene structure, and surface textures.

Figures 6 and 7 show the results from a single view from an uncalibrated camera with severe lens distortion, while Figure 8 shows the actual model extracted.

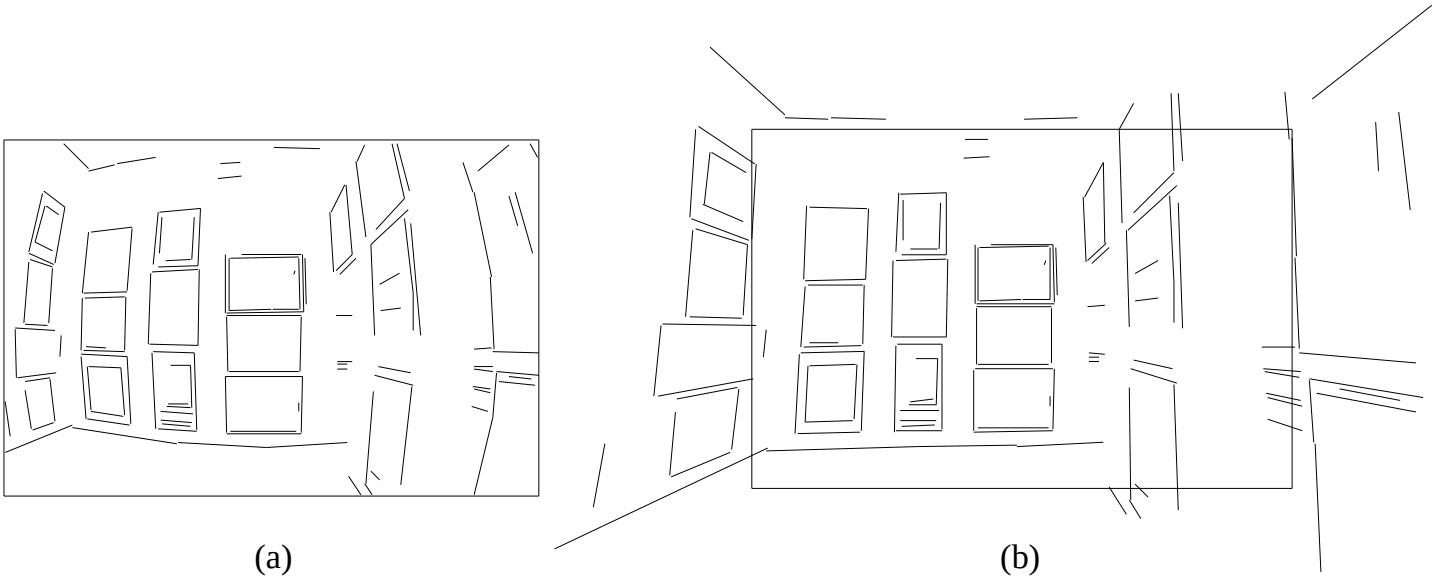

Figure 6: Camera calibration: (a) original view with lens distortion (b) corrected result

6 Conclusion

Our approach is to exploit the proven effectiveness of vision techniques where possible and human guidance where necessary to create detailed scene descriptions from random sets of uncalibrated views. Special care has been taken to rely upon only a modicum of user assistance in the form of grouping relevant features.

The further development of this work looks toward eventual complete automation of the model-creation process, application of other methods for analyzing non-planar objects after extracting the calibration information by the method here described, and the use of the resulting description to produce a compact model for real, moving scenes.

7 Acknowledgments

Special thanks to sponsors of the Television of Tomorrow Consortium whose financial and technical contributions have supported this research.

8 REFERENCES

[1] V. M. Bove, Jr., B. D. Granger, and J. A. Watlington, Real-Time Decoding and Display of Structured Video, *Proceedings IEEE ICMCS '94*, Boston MA, (1994), pp. 456-462.

[2] Steve Mann and Shawn Becker, Computation of some projective-chirplet-transform and metaplectic-chirplet-transform subspaces, with applications in image processing, in *Proceedings DSP World Symposium*, Boston, Massachusetts, November, (1992).

[3] Amnon Shashua and Nassir Navab, Relative affine structure: theory and application to 3D reconstruction

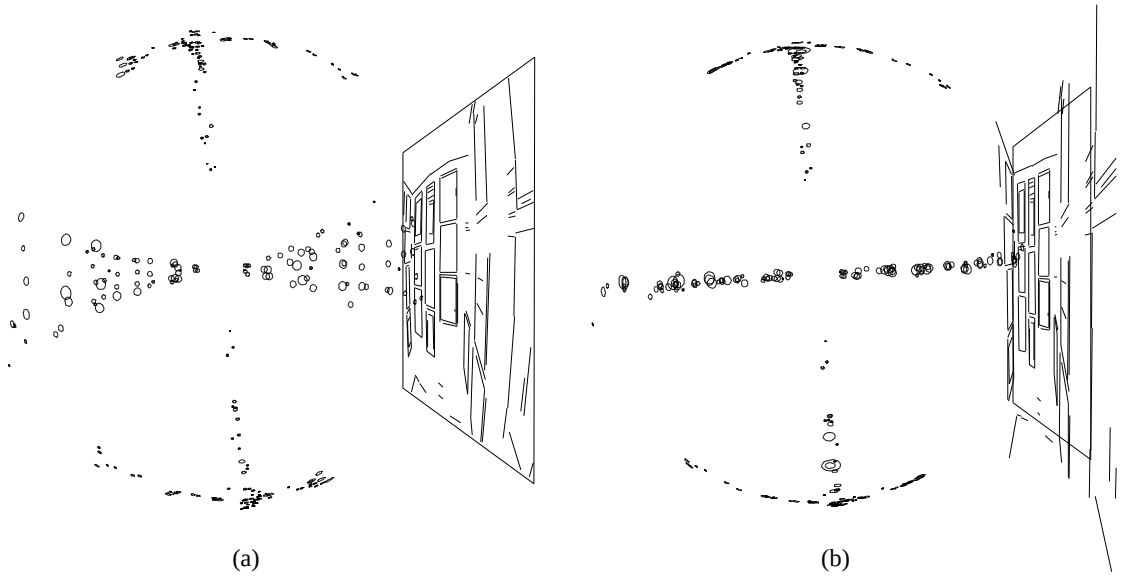

Figure 7: Results of camera calibration on distributions of sphere points: (a) detected edges and sphere points of a severely distorted image (b) edges and sphere points after distortion correction

from perspective views, In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, Seattle, Washington, (1994).

[4] Berthold K.P. Horn, Relative orientation, *International Journal of Computer Vision*, 4, 59-78 (1990).

[5] Berthold K.P. Horn, Closed-form of absolute orientation using unit quaternions, *International Journal of Computer Vision*, 4, 629-641 (1987).

[6] Neil D. McKay, Using Quaternions for least-squares fitting of rotation transformations, *General Motors Research Laboratory Research Report*, CS-518, November 3, (1986).

[7] Berthold K.P. Horn and E.J. Weldon Jr., Direct methods for recovering motion, *International Journal of Computer Vision*, 2, 51-76 (1988).

[8] David Heeger and Allan Jepson, Simple method for computing 3D motion and depth, *Proceedings 3rd IEEE International Conference on Computer Vision, 96-100*, (1990).

[9] Carlo Tomasi and Takeo Kanade, Shape and motion from image streams: a factorization method; full report on the orthographic case, *International Journal of Computer Vision*, 9(2):127-154, November (1992).

[10] A. Azarbayejani and A. Pentland, Recursive estimation of motion, structure, and focal length, (in press) *IEEE Pattern Analysis and Machine Intelligence*, April (1994).

[11] T. J. Broida, S. Chandrashekhar, and R. Chellappa, Recursive 3-D motion estimation from a monocular image sequence, *IEEE Transactions on aerospace and electronic systems*, Vol. 16, No. 4, July (1990).

[12] W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling, *Numerical recipes in C*, Cambridge University Press, Cambridge, UK, (1988).

[13] Athanasios Papoulis, *Probability, Random Variables, and Stochastic Processes*, Third ed., McGraw Hill, New York, page 156, (1991).

[14] B. K. Horn, Robot Vision, MIT Press, (1987).

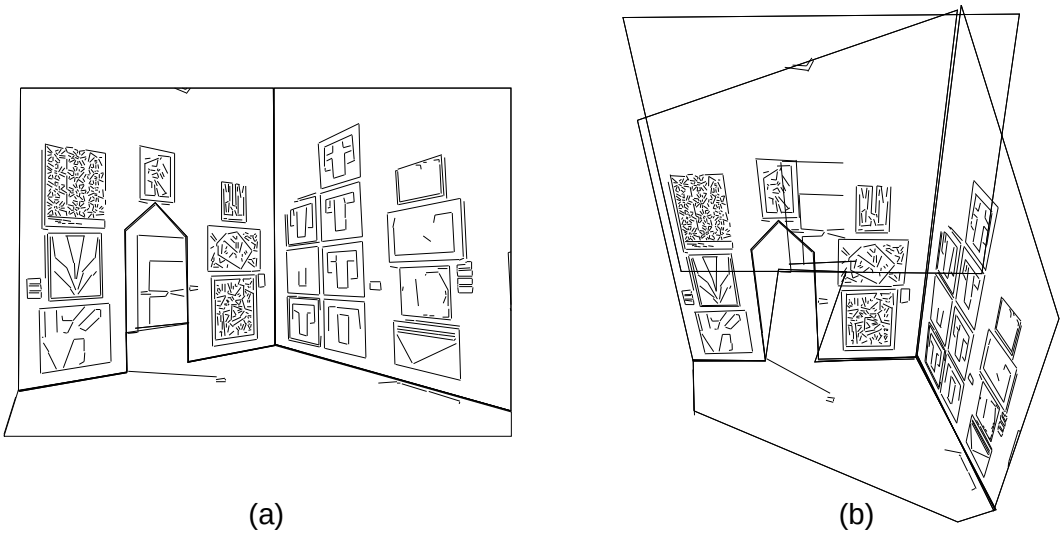

Figure 8: Results of semiautomatic 3-D model extraction from uncalibrated 2-D camera view: (a) detected edges from the original uncalibrated view (b) extracted model as seen from a new synthetic camera position

[15] Duane C. Brown, Close-range camera calibration, Presented at the *Symposium on close-range photogrammetry*, Urbana, Illinois, January (1971).

[16] Kenichi Kanatani, Statistical analysis of focal-length calibration using vanishing points, *IEEE Transactions on Robotics and Automation*, Vol. 8, No. 6, December (1992).

[17] Jim Z.C. Lai, On the sensitivity of camera calibration, *SPIE*, Vol. 1822, (1992).

[18] J. Canny, A Computational Approach to Edge Detection, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-8, No. 6. Nov. (1986), pp. 679-698.

[19] RobotVis Team, Project 4, INRIA, http://zenon.inria.fr:8003/Equipes/ROBOTVIS-eng.html

[20] H. F. Durrant-Whyte, Uncertain Geometry, chapter of *Geometric Reasoning*, MIT Press, ed. D. Kapur and J. L. Munday, pp. 447-481, (1989).

[21] R. Y. Tsai and T. S. Huang, Estimating Three-Dimensional Motion Parameters of a Rigid Planar Patch, *IEEE Transactions on Acoustics, Speech and Signal Processing*, Vol. ASSP-29, No. 6, Dec. (1981), pp. 1147-1152.

[22] Alan V. Oppenheim and Ronald W. Schafer, *Discrete-time signal processing*, Prentice Hall, Englewood Cliffs, New Jersey, (1989).