A Survey of Motion-Parallax-Based 3-D Reconstruction Algorithms

Ye Lu, Jason Z. Zhang, *Member, IEEE*, Q. M. Jonathan Wu, *Member, IEEE*, and Ze-Nian Li, *Member, IEEE*

***Abstract*—The task of recovering three-dimensional (3-D) geometry from two-dimensional views of a scene is called 3-D reconstruction. It is an extremely active research area in computer vision. There is a large body of 3-D reconstruction algorithms available in the literature. These algorithms are often designed to provide different tradeoffs between speed, accuracy, and practicality. In addition, even the output of various algorithms can be quite different. For example, some algorithms only produce a sparse 3-D reconstruction while others are able to output a dense reconstruction. The selection of the appropriate 3-D reconstruction algorithm relies heavily on the intended application as well as the available resources. The goal of this paper is to review some of the commonly used motion-parallax-based 3-D reconstruction techniques and make clear the assumptions under which they are designed. To do so efficiently, we classify the reviewed reconstruction algorithms into two large categories depending on whether a prior calibration of the camera is required. Under each category, related algorithms are further grouped according to the common properties they share.**

***Index Terms*—Camera self calibration, motion parallax, stereo vision, three-dimensional reconstruction, triangulation.**

I. Introduction

ANALYSIS and interpretation of visual information are at the heart of all computer vision research. The human visual system perceives the three-dimensional (3-D) world as retinal images in our eyes through a process called *projection*. The availability of both physiological and psychological cues gives the human visual system the ability to perceive depth. Some examples of these cues include binocular parallax, monocular movement parallax, accommodation, convergence, linear perspective, shades and shadows, and so on [1]. Computer vision researchers exploit these visual cues to design algorithms to emulate the way our eyes perceive depth. Binocular parallax is the most important depth cue in our visual system. It refers to the slightly different images sensed by the left and right eyes because of their slight difference in location. The biological visual system combines these two images to reconstruct a 3-D description of the world that it sees. In a similar manner, our visual system exploits the monocular motion parallax by fusing together slightly different images taken at slightly different locations in the world. Even though these two cues are physiologically different, we can however regard binocular parallax as a special case of monocular motion parallax from the computational perspective. Therefore, we will loosely refer to both as motion parallax in this paper.

In this paper, we shall restrict ourselves to motion-parallax-based reconstruction algorithms. Even with this restriction, there is still an incredible amount of recent research results. It is therefore essential for both new comers as well as veteran researchers to arm themselves with the state-of-the-art developments. One of the important goals of this paper is to offer precisely this up to date information to other researchers. In addition, we believe that a proper classification of the available algorithms in the literature will not only pave the way for a systematic study of 3-D reconstruction, but also help in developing insights into many of the core issues within the underlying problem.

We will begin this paper by proposing a classification of the available methods. Then, after a brief review of projective geometry and associated notations used in this paper, we will closely examine the algorithms within each category. Concluding remarks will be given in Section VI.

II. Classification of Reconstruction Algorithms

An important step towards emulating the human visual system is the ability to compute 3-D properties of the world from two or more two-dimensional (2-D) images. Algorithms that hope to fulfill this purpose need to operate on the measurements of light in free space. The complete set of all such measurements is known as the *plenoptic function* $P(V_x, V_y, V_z, \theta, \phi, \lambda, t)$, which represents the radiance in space as a function of viewing position V_x, V_y, and V_z, the angles θ and ϕ in which the light rays pass through the pupil, the wave length λ, and time t [2]. Motion parallax is captured within the plenoptic function by translating the observer through a range of viewing positions. Since the information needed for reconstruction algorithms can all be found within the plenoptic function, it is natural to wonder whether the plenoptic function itself is enough to unambiguously describe the 3-D structure of the scene. This important question is answered by Baker *et al.* [3]. They found that this problem is unambiguous if and only if there is no extended region in the scene that is radiating a constant intensity, color, and polarization. From here on, we shall assume the scene is unambiguous in that it does not contain extended constant intensity regions. Armed with this knowledge, we can effectively discuss 3-D reconstruction in the context of unambiguous scenes and develop a meaningful classification of the available algorithms.

Various algorithms have been proposed to solve the 3-D reconstruction problem. Even though these algorithms may po-

Manuscript received August 13, 2003; revised January 18, 2004 and March 29, 2004. This paper was recommended by Associate Editor D. Zhang.

Y. Lu and Z.-N. Li are with Simon Fraser University, Vancouver, BC V5A 1S6, Canada (e-mail: yel@cs.sfu.ca; li@cs.sfu.ca).

J. Z. Zhang and Q. M. J. Wu are with the National Research Council of Canada, Vancouver, BC V6T 1WJ, Canada (e-mail: jason.zhang@nrc.ca; jonathan.wu@nrc.ca).

Digital Object Identifier 10.1109/TSMCC.2004.829300

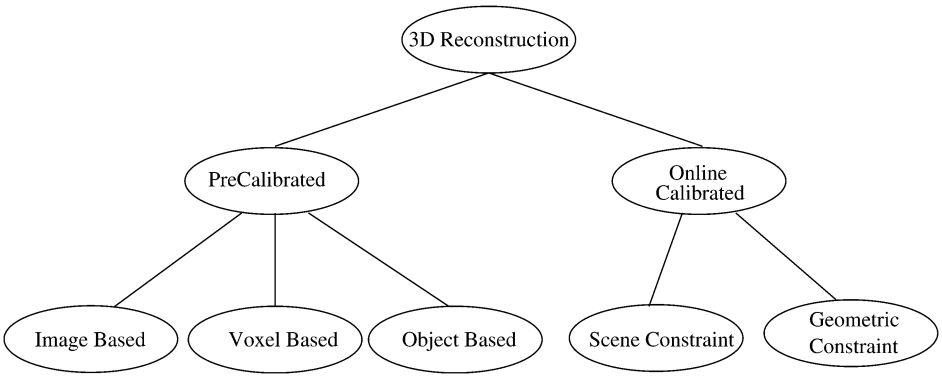


Fig. 1. Classification of reconstruction algorithms.

tentially differ in their initial assumptions and sparsity of input and output, the ultimate goal they have in common is to produce a 3-D description of the underlying scene. Because of the vast amount of algorithms available, it is helpful to group them into classes and study each class individually. Here, we propose such a classification shown in Fig. 1.

Accurate camera calibration is vitally important for any 3-D reconstruction task. Therefore, instead of grouping together algorithms according to the sparsity of their input and output (e.g., the reconstruction of extracted image features versus the recovery of dense depth maps) or the number of frames they need to compute the reconstruction, we feel that it is more natural to place emphasis on the methods in which camera calibration is obtained in the algorithms.

In our classification, we differentiate between algorithms that assume a prior calibration of the camera (pre-calibrated) and those that can obtain the calibration at run time (online calibrated). This distinction is significant in practical settings as well. For example, when the goal is to obtain a 3-D model as accurately as possible and we have access to the camera such that a full calibration of it can be performed, it is much more beneficial to select one of the pre-calibrated reconstruction algorithms. However, if we wish to reconstruct a scene from some previously recorded video sequence, we would then have to use an online calibrated reconstruction method. Algorithms within both the pre-calibrated and online calibrated categories are further divided according to the common principles in which the algorithms are designed. In the pre-calibrated case, we have the classes "image-based algorithms", "voxel-based algorithms", and "object-based algorithms". The first class "image-based algorithms" contains algorithms that compute reconstructions from either sparsely matched image features or dense stereo correspondences. The algorithms in the class "voxel-based algorithms" are those that works by discretizing the scene space into a set of voxels. The "object-based algorithms" use variational principles to deform an initial set of surfaces toward the objects to be detected and reconstructed. On the online calibrated side, algorithms are classified according to how the calibration of the camera is computed. At present, there are two classes: "scene constraint" and "geometric constraint". Algorithms that fall within the class "scene constraint" are those that take advantage of special scene structures such as parallel lines within the scene and compute the vanishing points in each principle direction to determine the cameras's intrinsic and extrinsic parameters. By contrast, the algorithms in the second class do not assume any prior knowledge of scene structure. They estimate the camera parameters using the projected image of an abstract geometric object called the *absolute conic*. Apparently, the five categories in our classification cover only a majority of the most popular algorithms. As the need arises, new classes can be defined and added easily into our classification.

III. Review of Projective Geometry

The use of Euclidean geometry is common place when describing 3-D entities. The intuitive notion of parallelism, angle between lines, ratio of lengths, and so on are readily quantified using Euclidean geometry. However, when considering the imaging process, Euclidean geometry is inadequate. This is because the perspective projection process performed by the camera does not preserve any of the entities mentioned before. Therefore, we need a more general form of geometry to be able to describe the image formation process.

Projective geometry includes Euclidean geometry as a special case. The group of transformations associated with projective geometry subsumes that of Euclidean geometry which is completely depicted with rotation and translation. In particular, it includes the perspective projection. Furthermore, working within the projective framework often yields much simpler formulas, reduces the need to handle many special cases, and most important of all, creates a natural concept of duality. As an example of projective duality, a 2-D point with coordinate $(a/c, b/c)$ and a line $ax + by + c = 0$ are both represented as the three vector (a, b, c). Thus, for any theorems proven for points, there is a corresponding dual proof for lines. Under projective geometry, the action of a perspective camera becomes a linear operation.

A. Pinhole Camera Model

A camera is a device that maps points in 3-D space onto a plane called the *image plane*. The point $\mathbf{X} = (X, Y, Z)^T$ in 3-D is mapped to a 2-D point on the image plane where the line from the point $\mathbf{X}$ to the *center of projection* $\mathbf{C}$ intersects the image plane. Using similar triangles, it is easy to show that the point $\mathbf{X}$ projects to the 2-D point $(fX/Z, fY/Z)^T$, where f is the *focal length* of the camera. The line from the camera center perpendicular to the image plane is called the *principal ray* and the

principal ray intersects the image plane at the *principal point*. Using homogeneous coordinates, we can express this projection operation in matrix form as

$$\begin{pmatrix} fX \\ fY \\ Z \end{pmatrix} = \begin{bmatrix} f & & & 0 \\ & f & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}. \tag{1}$$

If we call the above transformation matrix P, we can write the above equation in algebraic form as

$$\mathbf{x} = P\mathbf{X}. \tag{2}$$

The above formulation assumes that the camera coordinate system coincides with the world coordinate system. Generally, the principal point will be at the location (p_x, p_y), and the mapping from a 3-D location $(X, Y, Z)^T$ to the 2-D image plane is

$$(X, Y, Z)^T \rightarrow \left(\frac{fX}{Z} + p_x, \frac{fY}{Z} + p_y \right)^T. \tag{3}$$

If we let K be the matrix

$$K = \begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix} \tag{4}$$

then we can rewrite the projection equation as $\mathbf{x} = K[I|0]\mathbf{X}$, and we have $P = K[I|0]$. The matrix K is called the *camera calibration matrix*. Notice that we still have the assumption that the camera coordinate system coincides with the world coordinate system. To remove this assumption, we need to translate and rotate the 3-D input point into the camera coordinate system from the world coordinate system. This is done in the following equation:

$$\mathbf{X}_{\text{cam}} = \begin{bmatrix} R & -R\mathbf{C} \\ 0 & 1 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{5}$$

where R is the 3×3 rotation matrix that rotates the world coordinate system into the camera coordinate system and $\mathbf{C}$ is the three vector representing the position of the camera center in the world coordinate system. Writing everything algebraically, we have

$$\mathbf{x} = KR[I| - \mathbf{C}]\mathbf{X}. \tag{6}$$

For additional generality, the camera calibration matrix may be written in the form

$$K = \begin{bmatrix} \alpha_x & s & x_0 \\ & \alpha_y & y_0 \\ & & 1 \end{bmatrix} \tag{7}$$

where the added parameter s is referred to as the *skew* of the camera. The skew parameter is normally 0, but it may take on nonzero values in some unusual circumstances. A nonzero skew value would imply that the x and y axes of the image plane are not perpendicular. This is very unlikely to happen in practice. However, when taking an image of an image or enlarging an image, the skew parameter of the resulting image can be nonzero. Equation (7) also handles the case where the camera produces nonsquare pixels. In that case, we have $\alpha_x = fm_x$ and $\alpha_y = fm_y$, where m_x and m_y represent the pixel dimensions in the x and y directions, respectively. In terms of pixels coordinates, we have $x_0 = m_x p_x$ and $y_0 = m_y p_y$.

B. Epipolar Geometry

The projective geometry between two views of a scene is completely described by the *epipolar geometry*. In other words, the epipolar geometry is the intrinsic geometry of two views. It has important applications in stereo matching as it limits the search for correspondence into a one-dimensional search space.

Consider a point $\mathbf{X}$ in 3-D that is imaged in two views. In the first view, its image is point $\mathbf{x}$, while in the second view, its image is $\mathbf{x}'$. For a typical stereo matching problem, we have the point $\mathbf{x}$ in one image and wish to find its correspondence $\mathbf{x}'$ in another image. We observe that both camera centers $\mathbf{C}$ and $\mathbf{C}'$, the points $\mathbf{x}$, $\mathbf{x}'$, and $\mathbf{X}$ are coplanar. This plane is the *epipolar plane* π. The line that connects the two camera centers is called the *baseline*. The points $\mathbf{e}$ and $\mathbf{e}'$ where the baseline intersects the two views are called *epipoles*. The lines connecting $\mathbf{x}$, $\mathbf{e}$ and $\mathbf{x}'$, $\mathbf{e}'$ are the *epipolar lines*. From the definition of perspective projection, we know that the points $\mathbf{C}$, $\mathbf{x}$, and $\mathbf{X}$ are collinear and that any point on this line between $\mathbf{x}$ and $\mathbf{X}$ projects as $\mathbf{x}$ in the first image. Therefore, we see that the correspondence of $\mathbf{x}$ must lie on the projection of the line from $\mathbf{x}$ to $\mathbf{X}$ in the second image. An illustration of epipolar geometry is shown in Fig. 2.

C. Fundamental Matrix

The fundamental matrix is the algebraic representation of epipolar geometry. It forms the mapping between a point $\mathbf{x}$ in one image and the epipolar line that contains its correspondence in the second image. Derivations of the fundamental matrix can be found in [4] and [5]. Since the dot product of a point and the line on which it belongs to is zero, we arrive at the *epipolar constraint*

$$\mathbf{x}'^T F\mathbf{x} = 0 \tag{8}$$

where F is the fundamental matrix. Some properties of the fundamental matrix include [4] the following.

- If F is the fundamental matrix of the pair of cameras (P, P'), then F^T is the fundamental matrix for (P', P).
- For the point $\mathbf{x}$ in the first image, its epipolar line is given by $\mathbf{l}' = F\mathbf{x}$. Similarly, for $\mathbf{x}'$ in the second image, its epipolar line is $\mathbf{l} = F^T\mathbf{x}'$.
- The epipolar line $\mathbf{l}' = F\mathbf{x}$ contains the *epipole* $\mathbf{e}'$, where an epipole is the projection of the other camera center onto the current image plane.
- F satisfies the constraint $det(F) = 0$, which means that F is not of full rank.
- F is a projective map that takes a point to a line. If $\mathbf{l}$ and $\mathbf{l}'$ are corresponding epipolar lines, then any point on $\mathbf{l}$ maps to the same line $\mathbf{l}'$. Thus, there is no inverse mapping.

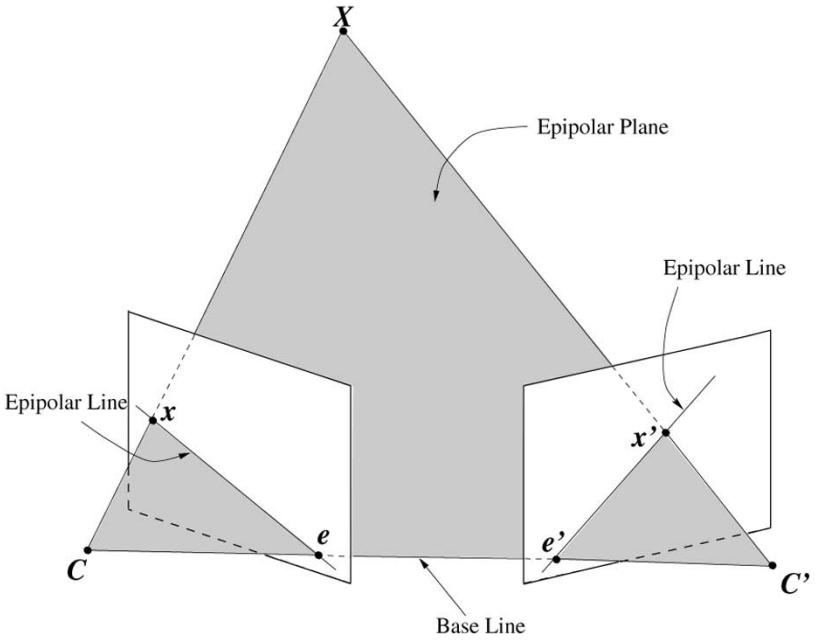


Fig. 2. Illustration of epipolar geometry.

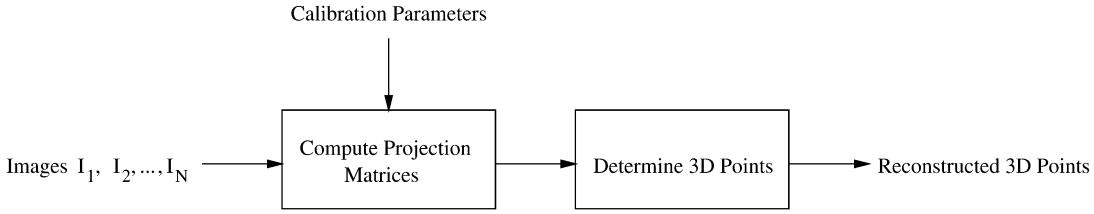


Fig. 3. Block diagram for calibrated reconstruction.

The eight-point algorithm can be used to estimate the fundamental matrix between two views. With appropriate normalization of point coordinates, this simple algorithm is able to yield very good results [6].

D. Infinite Homography

A plane π relates the projection of points on it in one image with their correspondences in the second image. This is a projective relationship and we call it the *homography* between two views induced by the plane π. The homography induced by the *plane at infinity* is particularly important. We call it the *infinite homography* and denote it as H_∞. The form of the infinite homography may be derived as a limiting process from the homographies induced by planes of increasing distance away from the first camera on a calibrated stereo rig. From the derivation given in [4], we have

$$H_\infty = K' R K^{-1} \tag{9}$$

where K and K' are the calibration matrices of the first and second camera respectively, and R is the orientation of the second camera with respect to the first camera.

By definition, H_∞ maps points on the plane at infinity π_∞. Thus, it maps the vanishing points from one image to the other. As we shall see in Sections IV and V, the infinite homography is important for online calibrated reconstruction algorithms as it propagates calibration information from one view to the next.

IV. Pre-Calibrated Reconstruction

Pre-calibrated reconstruction algorithms are those that require an accurate prior calibration of the cameras. In other words, both the camera's intrinsic and extrinsic parameters need to be computed. One of the most popular camera calibration technique was developed by Tsai [7]. This method relies on the availability of a 3-D calibration object with special markers on it. In addition, it is required that the markers are not all coplanar. This calibration object provides correspondence between points on the image and 3-D points in space. A more practical method was recently proposed by Zhang [8]. This method only requires the camera to observe a planar pattern in at least two different orientations. The camera calibration toolbox [9] is a efficient implementation of calibration algorithms. It can be used both as a stand alone application in Matlab or as part of the Intel Computer Vision Library [10]. Using the calibration parameters, the projection matrices can be trivially computed using the expression for P in (6). The block diagram for calibrated reconstruction algorithms is shown in Fig. 3.

Because the camera calibration provides metric information about the scene, the resulting reconstruction is called a metric reconstruction. Because the absolute translation, rotation, and scale of the world coordinate system is unknown, a metric reconstruction only estimates the shape up to a similarity transform. These inherent indeterminacies are called *gauge freedoms*. Morris *et*

al. [11] showed that the selection of the unknown scale factor can significantly affect the accuracy of the estimated shape and proposed a way to determine what measurement should be made to maximize the shape accuracy.

Algorithms belonging to different classes under the calibrated reconstruction category generally differ in the last step "Determine 3-D Points". Each class of algorithms will be discussed in detail in Sections IV-A–IV-D.

A. Image-Based Reconstruction

Using sparse image features or dense pixel matchings are both considered image-based methods. Reconstruction of selected image features or dense matchings between images follow a very similar path. The only difference is the amount of data processed. The method used to determine the 3-D point from pairs of matched image pixels (the last step in Fig. 3) is usually triangulation. In the absence of noise, triangulation is trivial. However, with the presence of noise, the triangulation problem is much more complicated as the back projected rays from the two images will not generally meet in 3-D space. We therefore need to find a suitable point of "intersection". Because of the known camera calibration, our reconstruction is metric. Hence the concept of distance and perpendicularity is clearly defined. Therefore, the simplest method is to estimate the required 3-D location as the midpoint of the common perpendicular between the two back projected rays. In addition, with an assumed Gaussian noise model, a provably optimal triangulation method is available. From a pair of point correspondences $\mathbf{x}$ and $\mathbf{x}'$, this algorithm seeks an alternate pair $\hat{\mathbf{x}}$ and $\hat{\mathbf{x}}'$ such that the sum of squared distances to the original pair of points is minimized subject to the epipolar constraint. Thus, the optimal points to select are those that lie on a pair of corresponding epipolar lines closest to the original point correspondence. This pair of epipolar lines, $\mathbf{l}$ and $\mathbf{l}'$, can be found the minimizing the distance between them and the original point correspondence. Furthermore, by parameterizing the pencil of epipolar lines in the first image by a suitable parameter t, this minimization problem can be reduced to finding the real roots of a polynomial of degree 6. The complete algorithm can be found in [12].

1) Feature Detection and Correspondence: For feature-based algorithms, the crucial steps to an accurate reconstruction are feature detection and feature correspondence. The pioneer work on feature detection is done by Moravec in [13], [14]. The features, or "points of interest", are defined as locations where large intensity variations in every direction occur. The un-normalized local autocorrelation in four directions are computed and the lowest result is taken as the measure of interest. These measurement values are then thresholded with nonmaximum suppression applied. Harris and Stephens [15] built on the idea of the interest operator but used the first order image derivative to estimate local autocorrelation. The variation of the autocorrelation over different orientations is found by calculating functions related to the principle curvatures of the local autocorrelation. This algorithm gives robust detection but poor localization accuracy at certain junction types. Other corner-detection methods based on the product of gradient magnitude and edge contour curvature were independently proposed by Beaudet [16], Kitchen and Rosenfeld [17], Dreschler and Nagel [18], [19], and Zuniga and Haralick [20]. In their work, a corner is defined as the point of maximum planar curvature for the line of steepest slope.

More recent works on corner detection include the Smallest Univalue Segment Assimilating Nucleus (SUSAN) detector proposed by Smith and Brady [21] and the curvature scale space approach proposed by Mokhtarian and Suomela [22]. The SUSAN approach works by observing that each image point has a local area of similar brightness associated with it. The corner location is found using only the size, centroid, and second moments of the local area or USAN. In contrast, the curvature scale space (CSS) approach works by finding locations of maximum absolute curvature of edge contours and tracking these locations through multiple scales to improve localization. Both of these methods are robust with respect to image noise.

To successfully solve a structure from motion problem, we not only need to be able to detect features in one frame, but also locate the corresponding features in other frames as well. This problem is known as the stereo correspondence problem. There are number of elements that may affect the performance of any stereo correspondence algorithm such as change in light condition, foreshortening effects, uncorrelated image noise, and occlusion.

Two different approaches exist for solving the stereo correspondence problem. The first approach finds a set of features in one frame and attempts to track them through subsequent frames. In contrast, the second method tries to locate a set of features in each frame and finds the correspondence between the detected features. The first approach is often characterized as feature tracking and is exemplified by the work of Tomasi and Kanade [23]. The second approach, often referred to as feature correspondence, can be implemented using several methods. When computational resources are limited or that real-time performance is required, the correspondence problem can be solve using the correlation-based methods [24]. Correlation scores are computed by comparing a fixed window in the first image and a moving window in the second image. The pixel in the second image that received the highest correlation score is chosen to be the match. Marr and Poggio [25] proposed a relaxation-based method for stereo correspondence. The relaxation method works by propagating constraints to reorganize potential matches. Pilu [26] proposed an elegant SVD-based matching method. The SVD method builds a correlation weighted proximity matrix and then factors it using the SVD algorithm. A new matrix is recomposed by replacing the elements of the diagonal matrix with 1. If the ijth element in the new matrix is the greatest element in both its row and column, then the ith feature in the first image is in correspondence with the jth feature in the second image. The stereo correspondence problem can also be cast into an optimization problem. Baker and Binford [27], Ohta and Kanade [28] employed dynamic programming to match edge-delimited intervals. Li [29] also used dynamic programming to solve the correspondence problem. In this work, the figural continuity constraint is enforce by matching lines in Hough space.

2) Dense Stereo Matching: Similar to feature-based algorithms, the success of dense reconstruction algorithms relies heavily on the accuracy of the densely matched pixels. When the sampling along the time axis is also dense, the pixel displacements between consecutive frames can be approximated by optical flow [30]. Assuming that the camera is a rigid body, the motion field of the projected points on the optical plane satisfies the differential equation

$$\dot{\mathbf{X}}_{cam} = \boldsymbol{\omega} \times \mathbf{X}_{cam} + \mathbf{v} \tag{10}$$

where ω and $\mathbf{v}$ are angular and linear velocities of the camera respectively. It can be shown that the relationship between the image plane motion field $\mathbf{u}(\mathbf{x}) = [u_x, u_y]^T$ and the motion of the camera can be expressed as

$$\mathbf{u}(\mathbf{x}) = \frac{1}{Z} A(x)\mathbf{v} + B(x)\boldsymbol{\omega} \tag{11}$$

where Z is the depth and $A(\mathbf{x})$ and $B(\mathbf{x})$ are as defined in [31]. Thus, we see that the optical flow field has two components generated by the angular velocity and the linear translation respectively. Also, it is clear that no structure information is contained within the angular component of optical flow. This confirms our intuition that scene structure cannot be computed from images taken by a rotating camera.

A 3-D structure can be triangulated using the optical flow approximation since we have pixel to pixel matches between the two frames. This method has the advantage that the scene structure can be computed from two or more frames from a single calibrated camera. There are various methods to approximate the optical flow. A good review of a large number of methods is given in [30]. In practice, under the assumption of dense temporal sampling, the optical flow can be efficiently approximated by sparse feature displacements [32]. However, even if we can discount the effect of occlusions, lighting and shading changes on the accuracy of the estimated optical flow, the accuracy of the 3-D reconstruction is still very much constrained by the fact that the baseline between consecutive frames is usually very small. Thus, more reliable ways to compute the 3-D scene structure need to offer a large enough baseline.

Two calibrated cameras can be setup so that the epipolar lines between the two views are horizontal and that the corresponding epipolar lines lie on the same scan line on both images [33]. With this setup, many traditional stereo correspondence algorithms can be applied to compute the *horizontal disparity* between the two views. The correspondence problem can be effectively solved within the Markov random field (MRF) framework. In particular, if we assume that the images resulting from the two views of the scene are realizations of MRFs, then because of the Markov–Gibbs equivalence [34], the set of random variables $F = \{F_1, F_2, \ldots, F_m\}$ defined over the set of pixel matchings (or disparities) follow the Gibbs distribution of the form

$$P(F = f) = \frac{e^{-(1/T)U(f)}}{\sum_{f \in F} e^{-(1/T)U(f)}} \tag{12}$$

where T is a constant called *temperature*, and $U(f)$ is the energy function

$$U(f) = \sum_{c \in C} V_c(f). \tag{13}$$

The energy function is summed over all the clique potential $V_c(f)$ over all possible cliques C. If we consider the stereo correspondence problem as a labeling problem (the disparity value being the label on each pixel), then we can formulate the solution as being the labeling that maximizes the posterior probability. Using the prior model $P(f)$, the posterior probability $P(f|d)$ with the observation d depends on the likelihood energy $U(f|d) = U(d|f) + U(f)$. Thus, finding the*maximum a posteriori* (MAP) estimate is equivalent to finding the solution to

$$f^* = \arg\min_f U(f|d). \tag{14}$$

Since the MAP-MRF framework requires the solution of a nonlinear optimization problem, locally optimal solutions can be readily obtained using gradient-based methods. Recently, globally optimal solutions to such problems using graph theoretic algorithms have been proposed [35]–[37]. In their work, Kolmogorov *et al.* casted this optimization problem as a minimum cut problem in graph theory. In their formulation, the energy function has the form

$$U(f|d) = \sum_{p \in L} D_p(f_p) + \sum_{p\, q \in N} V_{p,q} \cdot T(f_p \neq f_q) \tag{15}$$

where L is the set of pixels, N is an appropriate neighborhood system, and the function T is 0 when $f_p = f_q$ and 1 otherwise. The first term assigns a cost to each pixels with disparity f_p and the second term penalizes neighboring pixels having different disparities. This optimization problem can be solved using a Max-flow/Min-cut algorithm. For the case of binary labels, the graph $G = (V, E)$ is constructed as described in [38] and the global minimum can be obtained. However, for general disparity values, a local minimum in a strong sense can be found using the graph cut algorithm [35]. The result of their algorithm is shown in Fig. 4. Scharstein and Szeliski [39] provided a taxonomy of the various stereo correspondence algorithms, as well as an up to date review and comparison of the popular algorithms.

B. Voxel-Based Reconstruction

Because of the rapid growth of computational storage and processing power, volumetric representation of scene structure becomes practical. Various approaches to recover volumetric scene structure from a sequences of images have been proposed. Earlier attempts to solve this problem approximate the *visual hull* of the image objects. The visual hull of an object is defined as the maximal shape that gives the same silhouette as the actual object for all views outside the convex hull of the object. Methods that approximate the visual hull are referred to as *volume intersection* or *silhouette intersection*.

Similar to the convex hull, the visual hull is an approximation of the actual shape of the object. However, the size of the visual hull decreases monotonically with the number of 2-D images [41]. A necessary pre-processing step to compute the visual

(a) (b) (c) (d)

Fig. 4. Matching result from the graph cut algorithm [40]: (a) left image, (b) right image, (c) ground truth disparity, and (d) computed disparity.

hull is the segmentation of each 2-D image into object foreground and background. The segmented 2-D silhouettes from each image are then back projected and intersected to yield a volume segment representation which can be further processed into a surface description. This is the basic approach taken by Martin *et al.* [42]. This work is extended into octree representations by a number of other researchers [43]–[46].

A method related to volume intersection is the voxel occupancy algorithm proposed by Snow *et al.* [47]. In voxel occupancy, the scene is represented by a set of voxels and the algorithm labels each voxel as being either filled or empty. Thus, this is a labeling problem and can be solved using the graph cut algorithm as outlined in Section IV-A-II. However, the voxel occupancy method generalizes on the volume intersection method in the sense that it provides the notion of spatial smoothness as a term within the energy function that it tries to minimize.

An alternative volumetric reconstruction method utilizes *color consistency* to identify surface points in the scene. Color consistency requires that the projection of all surface point in different views to have consistent color. Specifically, if a nonoccluded point belongs to the surface of the object, then its projection onto different views should have approximately the same color. Conversely, the projections of a point not on the surface usually do not have a consistent color. Color consistency can be mathematically defined as the standard deviation, the L_1, L_2, or L_∞ norm of the colors of the pixels that a particular voxel projects to in each view. Algorithms that exploit color consistency are all variants of the *voxel coloring* approach. In practice, voxel coloring starts by initializing the scene with only opaque voxels. As the algorithm iterates, voxels are tested for color consistency. Only consistent voxels are kept while inconsistent voxels are carved out. The algorithm stops when all voxels are consistent. As voxels often occlude each other, it is vitally important for voxel coloring algorithms to determine the *visibility* of a particular voxel before performing the consistency test.

Determining the visibility (the set of pixels that a particular voxel can be seen) of a voxel is relatively difficult as the set of occluding voxels changes frequently as voxels are constantly being carved. To simplify this task, Seitz and Dyer [48], [49] imposed the *ordinal visibility constraint* on the camera locations. This constraint requires that all cameras be located on one side of the scene in a way that all voxels can be visited in a near to far order relative to every camera. To determine visibility, an occlusion bit map is used that consists of 1 bit for every camera pixel in the input. From the near to far order, every voxel is tested for consistency. If a voxel is found to be consistent, then all occlusion bits along all back projected rays from each view are marked as occluded. Then, the visibility of a voxel is simply the pixels in its projections such that the occlusion bit is clear. Note that voxel coloring algorithms based on the ordinal visibility constraint only needs to traverse each voxel once. Reconstruction results from Seitz's algorithm is shown in Fig. 5. However, further performance improvements of this algorithm is still possible. Prock [50] proposed a multiresolution voxel coloring scheme which reduced the running time by up to 40 times. Culbertson [51] *et al.* proposed the generalized voxel coloring methods which computes visibility exactly and hence yields a color-consistent model.

Compared with the image-based methods which need to solve the difficult correspondence problem, the voxel-based methods have the advantage allowing geometric reasoning to be performed in 3-D. Thus, no explicit correspondence is needed. Furthermore, the occlusion problem is handled much more elegantly. These reasons account for the success of the voxel-based algorithms. One of the main design considerations of voxel-based algorithm is that memory consumption. For a moderate scene having 100 voxels in each dimension, the entire scene would require a total of 10^6 voxels to represent. Therefore, memory constraints would often lead to coarse reconstructions. Moreover, the ordinal visibility constraint imposes a very tight constraint on the camera locations. In particular, camera configurations that surround the scene is not allowed. In Section IV-C, we will explore object-based reconstruction techniques that will alleviate this restriction.

C. Object-Based Reconstruction

While voxel-based reconstruction algorithms fill the scene with voxels and the reconstruction algorithm determines visibility of each voxel, object-based reconstruction algorithms aim at recovering a surface description of the objects in the scene. The level-set reconstruction method proposed by Faugeras *et al.* [52] is the first object centered 3-D reconstruction technique from image sequences. Applying variational principles used in their previous work for dense depth recovery [53], [54], Faugeras *et al.* reformulated the reconstruction problem into a surface evolution problem that can be solved using the level-set technique.

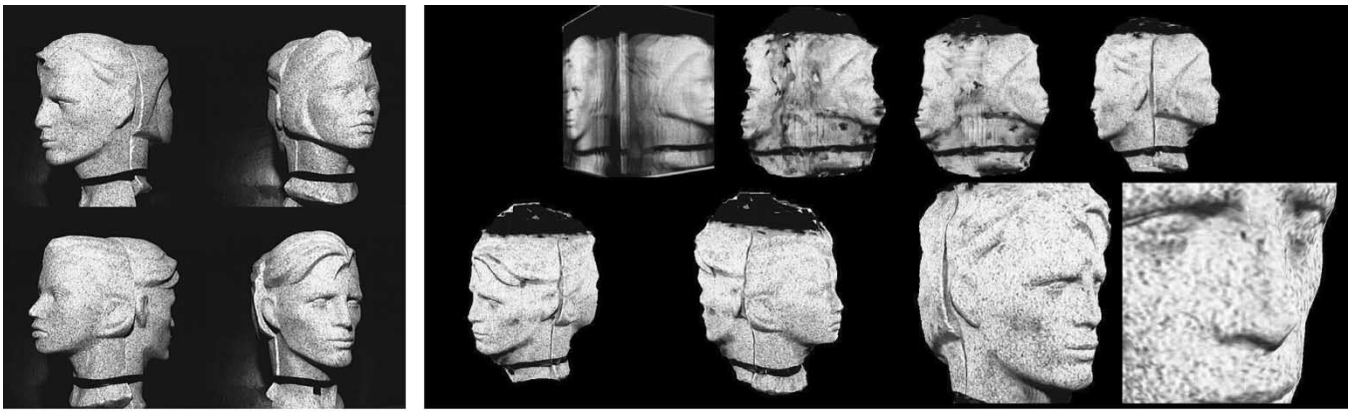
The level-set approach starts by choosing a surface that minimizes some energy functional

$$E(S) = \int_S \Phi(\mathbf{X}) dA. \tag{16}$$

Therefore, we need to choose a $\Phi(\mathbf{X})$ such that it is small at good matching locations and large otherwise. Once $\Phi(\mathbf{X})$ is

(a) (b) (c)

Fig. 5. Voxel coloring results [49]: (a) input image, (b) image from reconstruction, and (c) a novel view.



(a) (b)

Fig. 6. Reconstruction using the level-set method [52]: (a) input images and (b) evolving reconstruction results.

chosen, a surface S which minimizes (16) can be found using gradient descent. A simple choice of $\Phi(\mathbf{X})$ can be the summed square error of the matching pixels

$$\Phi(\mathbf{X}) = \frac{1}{n}\sum_{i \neq j} \Phi_{ij}(\mathbf{X}) \tag{17}$$

$$\Phi_{ij}(\mathbf{X}) = (I_i(\mathbf{x}) - I_j(\mathbf{x}'))^2. \tag{18}$$

However, this choice of Φ is very sensitive to noise and local texture. Therefore, Faugeras *et al.* suggested comparing neighborhoods around the matching points instead.

Essentially, the level-set reconstruction method evolves a time varying surface toward the objects in the scene. This is accomplished by moving the surface along the direction of its inwardly pointing normal. The velocity in which the surface moves depends on the cross correlation measure across views. When the evolving surface is far from the true surface of the object, the velocity is high because of the poor cross correlation. When the evolving surface moves near the true surface, the velocity decreases as the cross correlation becomes significantly better. An interesting property of the level-set method is that it can handle arbitrary topological changes as the zero level set can break apart and merge if necessary. Some reconstruction results are shown in Fig. 6.

The level-set approach proposed by Faugeras *et al.* assumes perfectly Lambertian surfaces. In particular, it does not address stereo matching problems caused by specularities present in the scene. An improved level-set approach that handles specular surfaces is proposed by Jin *et al.* [55]. A further improvement that handles specular, as well as translucent surfaces, is given in [56].

D. Comparison of Pre-Calibrated Reconstruction Algorithms

In this section, we compare the pre-calibrated reconstruction algorithms that were reviewed in Sections IV-A–IV-C. A quantitative comparison of these algorithms in terms of reconstruction accuracy cannot be meaningfully performed since the output from the various classes of algorithms are quite different. For image-based algorithms, the reconstruction is composed of a set of sparse 3-D points, voxel-based algorithms mark a set of visible voxels from each view, and the object-based algorithms produce a description of the surface enclosing the objects in the scene. Thus, we provide a qualitative comparison of these algorithms in terms of occlusion handing, the need for explicit feature tracking and matching, the ability to place cameras arbitrarily, and the need to remove background before applying the algorithm. The comparison is shown in Table I.

In order to select a pre-calibrated reconstruction algorithm for any application, it is essential to know the format of the required

TABLE I
COMPARISON OF PRE-CALIBRATED RECONSTRUCTION ALGORITHMS

	Occlusion Handling	Explicit Feature Correspondence	Camera Placement	Background Removal
Feature Tracking	No	No	Arbitrary	No
Feature Detection + Matching	No	Yes	Arbitrary	No
Dense Stereo Algorithms	Yes	No	Arbitrary	No
Voxel Occupancy	No	No	Arbitrary	Yes
Voxel Coloring	Yes	No	Constrained	Yes
Level-set	Yes	No	Arbitrary	Yes

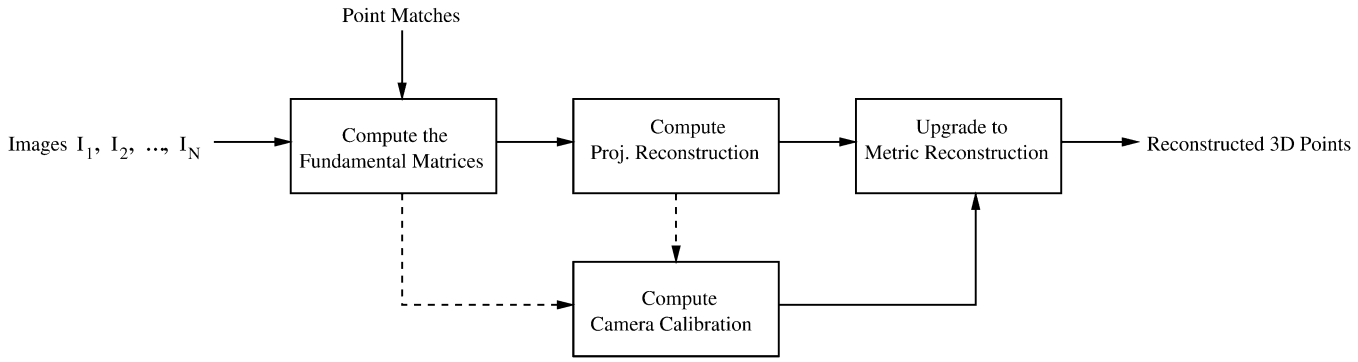


Fig. 7. Block diagram for online calibrated reconstruction.

output. In cases where only sparse 3-D output is needed, performing the reconstruction task using either feature tracking or feature detection and matching, with triangulation may be more appropriate. If succinct features are present on the objects of interest, these features can be detected and tracked (or matched) very reliably, yielding good reconstructed results. Another advantage of these algorithms is that no explicit background removal is necessary as the desired features are not present in the background. This property no only simplifies preprocessing, it also enables the algorithm to be robust against background noise.

However, if the eventual goal of the application is view synthesis, then voxel-based methods may be good choices. Because voxels are explicitly marked in 3-D, they can be easily projected onto any 2-D view given the desired movements of the camera. This class of algorithms performs best when there is an abundance of memory which allows the scene to be approximated by a large number of small voxels. The advantage of the voxel-based approach is that the output is in simple voxel form which is a commonly used format in computer graphics. Thus, any further processing can be easily performed with this representation. In addition, the voxel coloring algorithm is fairly efficient as each voxel only needs to be visited once. However, because color consistency is heavily enforced in the voxel-based approach, background removal should be performed prior to reconstruction to prevent mismatch with background pixels that just happen to have similar colors as the foreground pixels. Another disadvantage of voxel coloring is that cameras cannot be arbitrarily placed because of the ordinal visibility constraint.

The level-set algorithm offers an alternative to the voxel-based approach, but without the limitation on camera placements. This method works well when there is a number of dominant objects in the scene with unknown topology. However, similar to the voxel-based algorithms, background removal is necessary in order to achieve good reconstruction results.

V. ONLINE CALIBRATED RECONSTRUCTION

Under various circumstances, we may not have the luxury of performing the calibration task using a pre-made calibration object. It is a very realistic scenario for a number of applications. For example, in a video indexing application, we are only given the final video data without even knowing what type of video camera these videos were taken from, not to mention getting the same camera to take a video sequence of the calibration object. Furthermore, the intrinsic parameters of the camera may be changing during the acquisition of the video sequence because of focusing and zooming. Therefore, 3-D reconstruction tasks under these scenarios would have to be performed using online calibrated reconstruction methods.

The key difference between online calibrated reconstruction methods is the way the camera's parameters are estimated. This on-the-fly estimation of camera parameters is often referred to as camera self calibration or autocalibration. Autocalibration methods can be divided into two classes: those that exploit scene constraints and those that exploit geometric constraints. Hence, we divide the online calibrated reconstruction methods using these two classes. A block diagram for typical online calibrated reconstruction algorithms is shown in Fig. 7.

Algorithms that exploit scene constraints do not require the fundamental matrices and the initial projective reconstruction to determine the calibration parameters. Thus, only algorithms that apply geometric constraints follow the dashed lines in the diagram.

A. Projective Reconstruction

Using the estimated fundamental matrix, we can compute a projective reconstruction of the 3-D scene. A projective reconstruction consists of a set of 3-D points $\{\mathbf{X}_i\}$ and a set of camera projection matrices $\{P_i\}$. However, being a projective reconstruction means that the reconstruction is determined only up to a projective transform. Thus, for any projective transformation H, $\{P_i H^{-1}\}$ and $\{H\mathbf{X}_i\}$ yield an equally valid reconstruction.

From the estimated fundamental matrix between two view, a pair of projection matrices can always be retrieved [57]. The two camera matrices are chosen to be $P = [I|0]$ and $P' = [[\mathbf{e}']_\times F|\mathbf{e}']$. Using these projection matrices, we can compute a projective reconstruction of the detected features with a simple linear triangulation method. Since $\mathbf{x} = P\mathbf{X}$ and $\mathbf{x}' = P'\mathbf{X}$, we can eliminate the scale factor by taking the cross product of the left hand side with the right-hand side. Doing so produces three linear equations for each point, out of which two are linearly independent. Stacking the equations for the corresponding points and writing them in matrix form, we have

$$A\mathbf{X} = \begin{bmatrix} xp^{3T} - p^{1T} \\ yp^{3T} - p^{2T} \\ x'p'^{3T} - p'^{1T} \\ y'p'^{3T} - p'^{2T} \end{bmatrix} \mathbf{X} = \mathbf{0} \tag{19}$$

where p^{iT} are the rows of the projection matrix P. The solution of (19) is the singular vector corresponding to the smallest singular value of the SVD of A. Methods for optimal triangulation are discussed in [12].

When the image sequence contains more than two frames, it may be more efficient to solve the projective reconstruction problem for all feature correspondences and all views at once. A method for doing that is presented by Tomasi and Kanade [58] for projective reconstruction of orthographic cameras. The full perspective method is proposed by Sturm and Triggs [59]. An important assumption of all factorization-based methods is that all feature correspondences are visible in all views. That is, there are no occluded points. This may seem to be an unreasonable assumption for large sequences, but we can always get around it by breaking large sequences into a number of smaller sequences. In addition, Tomasi and Kanade [58] proposed a modification to the usual factorization algorithm to recover a small set of missing points. The way the perspective version of the factorization algorithm works is to first estimate a set of *projective depths*, λ_{ip}, for feature p in view i and then construct a measurement matrix by stacking together the product of the features in each view and the estimated projective depths in all views. After that, the measurement matrix can be factored using the SVD into the projection matrices and the 3-D locations of each feature. Putting the above discussion into mathematical form, we have

$$W = \begin{bmatrix} \lambda_{11}\mathbf{x}_{11} & \lambda_{12}\mathbf{x}_{12} & \dots & \lambda_{1n}\mathbf{x}_{1n} \\ \lambda_{21}\mathbf{x}_{11} & \lambda_{22}\mathbf{x}_{12} & \dots & \lambda_{2n}\mathbf{x}_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ \lambda_{m1}\mathbf{x}_{11} & \lambda_{m2}\mathbf{x}_{12} & \dots & \lambda_{mn}\mathbf{x}_{1n} \end{bmatrix} = \begin{bmatrix} P_1 \\ P_2 \\ \vdots \\ P_m \end{bmatrix} [\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_n]. \tag{20}$$

The projective depths are recovered using a re-statement of the epipolar constraint

$$(F_{ij}\mathbf{x}_{ip})\lambda_{jp} = (\mathbf{e}_{ij} \times \mathbf{x}_{ip})\lambda_{ip} \tag{21}$$

where F_{ij} is the fundamental matrix that maps points in view i into epipolar lines in view j.

Using (21), we can find the ratio between the projective depths in consecutive frames as

$$\lambda_{jp} = \frac{\|\mathbf{e}_{ij} \times \mathbf{x}_{jp}\|^2}{(\mathbf{e}_{ij} \times \mathbf{x}_{ip}) \cdot (F_{ij}\mathbf{x}_{jp})}\lambda_{ip}. \tag{22}$$

Therefore, we can chain together this relationship and estimate the complete set of projective depths for a feature $\mathbf{x}$ from some arbitrary initial value such as $\lambda_{1p} = 1$.

Of course, to ensure good numerical conditioning, some normalization needs to be done. First, we need to normalize the image coordinates of features as described in [6]. In addition, we need to condition the $m \times n$ matrix $[\lambda_{ip}]$ by repeatedly rescaling the rows and columns such that the Frobenius norm of the row or column is 1 after each rescaling. This conditioning step terminates when the entries of the matrix do not change significantly. Experiments indicate that their algorithm performs quite well under real imaging conditions. However, the shortcoming of this algorithm is that it is quite sensitive to correspondence errors. Therefore, it is always a good idea to reject the outlier matchings using the epipolar constraint before invoking this algorithm.

B. Calibration Using Scene Constraints

If images are taken within constrained environments, the camera self calibration step can usually be simplified considerably. One example of such constrained environment is the architectural or man-made scene. The most noticeable characteristic within this environment is that it contain a large number of parallel lines. Parallel lines in each direction intersect at a point on the plane at infinity. The projection of these intersection points are the *vanishing points*. The knowledge of the position of the vanishing points in three dominant directions in an image greatly simplifies the determination of the intrinsic parameters of the camera used to produce this image. In fact, we can obtain closed form solutions for these parameters as a function of the vanishing points [60].

Before we can take advantage of the parallel structures within the scene for self calibration, the vanishing points have to be computed first.

1) Computing the Vanishing Points: The estimation of vanishing points from detected line segments can be divided into two steps: the *accumulation step* and the *search* step. The goal of the accumulation step is to use the detected line segments to vote for some location in the accumulation space which could potentially share the same vanishing point. The search step searches the accumulation space for cells that possess a large number of votes.

Various types of accumulation spaces are developed. Barnard [61] proposed the use of the Gaussian sphere centered on the optical center of the camera as the accumulation space. The advantage of this method is that the computational complexity is

low as the unbounded space of R^2 is mapped into a bounded space. Further improvements on this method are suggested by Quan and Mohr [62], Lutton [63], and Magee and Aggarwal [64]. However, as pointed out by Rother [65], the main drawback of this method is that distances between points and lines on the image plane are not invariant to translation and rotation. This problem can be avoided if the line segments are not transformed into a bounded space. This is exactly the approach taken by Rother [65] in which he considered the intersection of all pairs of noncollinear line segments and used the entire image plane as the accumulation space.

After the accumulation process, the search step is performed to find the vanishing points. The search method employed in [62], [64] is very straight forward. It repeatedly searches for the most dominant cell and removes all line segments corresponding to it until the maximum vote drops below a certain threshold. A modified version of this method that explicitly enforces the orthogonality condition is developed by van den Heuvel in [66].

2) Computing the Camera Calibration: Having detected the vanishing points in an image, the intrinsic parameters of the camera can now be estimated. When a point $\mathbf{X}$ is perspectively projected to the point $\mathbf{x}$ on the image plane, we have the usual projection equation $\mathbf{x} = P\mathbf{X}$, where $P = K[R|-\mathbf{t}]$ is the projection matrix. The matrix R and the vector $\mathbf{t}$ represent the rotation and translation of the camera relative to the world coordinate system respectively, while the matrix K is the calibration matrix containing the intrinsic parameters.

Caprile and Torre [60] presented a geometric derivation which establishes that the orthocenter of the triangle formed by the vanishing points in three mutually orthogonal directions is the principal point. Hartley [4] computes the focal length using the calibrating conic centered on the orthocenter of the triangle formed by the vanishing points. In his formulation, when the aspect ratio is unity and the skew is zero, the reflection $\mathbf{v}'$ of one vanishing point, say $\mathbf{v}_1$, about the orthocenter is the pole of the line $\mathbf{l}$ joining the other two vanishing points $\mathbf{v}_2$ and $\mathbf{v}_3$ with respect to the calibrating conic. This relationship can be written mathematically as

$$\mathbf{l} = C\mathbf{v}' \tag{23}$$

where C is the quadratic form of the calibrating conic. Thus, the focal length can be computed as the radius of the calibrating conic. In the work of Liebowitz and Zisserman [67], they define constraints on the image of the absolute conic in terms of the estimated locations of the vanishing points. The image of the absolute conic $\boldsymbol{\omega}$ is then estimated as the singular vector corresponding to the smallest singular value of the SVD of the constraint matrix. Since $\boldsymbol{\omega} = K^{-T}K^{-1}$, it can then be factored using Cholesky factorization to obtain the calibration matrix K.

C. Calibration Using Geometric Constraints

The more flexible class of algorithms can perform self calibration using only geometric constraints that are inherently available in the image sequences themselves. In this section, we shall examine the underlying principle that allows us to perform self calibration. Several different methods will be presented and analyzed.

1) The Dual Image of the Absolute Quadric Method (DIAC): In order to perform self calibration, we need a projective entity that conveys the calibration information from frame to frame. In addition, we make the observation that if all objects in the scene are rigid, moving around in it with a video camera is equivalent to applying *similarity transforms* to the entire scene and taking snapshots of the transformed scene. With this in mind, we find that the conic represented by the identity matrix I on the plane at infinity is invariant under similarity transforms. Since the only class of transformation that does not move the plane at infinity is the affine transform and that any affine transform can be written in the form

$$H_A = \begin{bmatrix} A & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \tag{24}$$

we have $A^{-T}IA^{-1} = I$, which implies that $AA^T = I$. This means that A is orthogonal and hence H_A must be a similarity transform. We call this conic the *absolute conic*. The dual of the absolute conic is the *absolute dual quadric* Q_∞^*. Its representation in matrix form is

$$Q_\infty^* = \begin{bmatrix} I_{3\times 3} & \mathbf{0} \\ \mathbf{0}^T & 0 \end{bmatrix}. \tag{25}$$

The absolute dual quadric has the same property that it is invariant under similarity transforms.

Any point on the plane at infinity can be written as $\mathbf{X}_\infty = (\mathbf{d}^T, 0)^T$. When imaged by a general camera $P = KR[I|-\mathbf{C}]$, we have

$$\mathbf{x} = P\mathbf{X}_\infty = KR\mathbf{d}. \tag{26}$$

Thus, the planar homography between the plane at infinity and the image plane is $H = KR$. With this, we see that the *image of the absolute conic* (IAC) is given by

$$\omega = (KR)^{-T}I(KR)^{-1} = K^{-T}RR^{-1}K^{-1} = (KK^T)^{-1}. \tag{27}$$

Therefore, it is obvious that the image of the absolute conic is invariant under translation and rotation of the camera and that it also conveys the intrinsic parameters of the camera. It is often more convenient to work with the *dual image of the absolute conic* (DIAC) $\omega^* = KK^T$ since it has a simpler form and that it is the projected image of the dual absolute quadric.

The idea behind DIAC-based self calibration is to transfer constraints on ω^* to constraints on Q_∞^* using the known camera matrices from the projective reconstruction of the scene. The method of self calibration using Q_∞^* was introduced by Triggs in [68]. If we make some simplifying assumptions about the calibration matrix K, it is trivial to translate it into constraints on the absolute dual quadric using the equation

$$\omega_i^* = K^i K^{iT} = P^i Q_\infty^* P^{iT}. \tag{28}$$

For example, if we assume that the principal point is located at the origin (we can always translate it to the origin if it is not),

the aspect ratio is unity, and that the skew is zero, we can obtain four linear constraints

$$\begin{aligned}\left(P^i Q_\infty^* P^{iT}\right)_{12} &= 0\\ \left(P^i Q_\infty^* P^{iT}\right)_{13} &= 0\\ \left(P^i Q_\infty^* P^{iT}\right)_{23} &= 0\\ \left(P^i Q_\infty^* P^{iT}\right)_{11} - \left(P^i Q_\infty^* P^{iT}\right)_{22} &= 0\end{aligned} \tag{29}$$

on the entries of Q_∞^* using the known form of ω_i^*, as follows:

$$\omega_i^* = \begin{bmatrix} f^2 & 0 & 0\\ 0 & f^2 & 0\\ 0 & 0 & 1\end{bmatrix}. \tag{30}$$

When there are more than three views in the sequence, a unique solution exists.

An iterative formulation is also possible. We can define the cost function as

$$C(\mathcal{X}) = \sum_i \left\| K^i K^{iT} - P^i Q_\infty^* P^{iT} \right\|^2 \tag{31}$$

where $\mathcal{X}$ is the vector that contains all the parameters within K^i, and Q_∞^* and both $K^i K^{iT}$ and $P^i Q_\infty^* P^{iT}$ are normalized to have unit Frobenius norm. This problem can be solved using the Levenberg-Marquardt algorithm [4].

2) The Kruppa Equations: The first self-calibration method introduced to the computer vision community by Faugeras, Maybank, and Luong [69], [70] employs the *Kruppa equations*. The original derivation of the Kruppa equations uses the dual image of the absolute conic and assumes the equality of the DIACs in both views. The Kruppa's equation in its original form given in [71] is

$$[\mathbf{e}']_\times \omega^* [\mathbf{e}']_\times = F \omega^* F^T. \tag{32}$$

The form of the Kruppa's equation given in (32) is not easily applied in practice. To address this problem, Hartley [72] gives an alternate derivation of the Kruppa equations using the fundamental matrix. In this work, the Kruppa equations are stated as

$$\frac{\mathbf{u}_2^T \omega^{*\prime} \mathbf{u}_2}{\sigma_1^2 \mathbf{v}_1^T \omega^* \mathbf{v}_1} = -\frac{\mathbf{u}_1^T \omega^{*\prime} \mathbf{u}_2}{\sigma_1 \sigma_2 \mathbf{v}_1^T \omega^* \mathbf{v}_2} = \frac{\mathbf{u}_1^T \omega^{*\prime} \mathbf{u}_1}{\sigma_2^2 \mathbf{v}_2^T \omega^* \mathbf{v}_2} \tag{33}$$

where $\mathbf{u}_i$ and $\mathbf{v}_i$ are the column vectors of corresponding matrices in the SVD of $F = UDV^T$ and σ_i are the singular values. Note that DIACs in the two views are not assumed to be identical in the formulation given in (33). Cross multiplying yields two quadratic equations in the parameters of ω^* and $\omega^{*\prime}$. Therefore, with the equality assumption of the DIACs, we have six quadratic equations in the five unknown parameters of ω^* for three views. We can solve this system of quadratic equations using the method of homotopy continuation.

The advantage of using Kruppa equations is that the calibration process does not require a prior projective reconstruction as in the case for methods that estimate the absolute dual quadric. In addition, for two views, the Kruppa equations provide the only constraint available on ω^*. However, it is often observed in practice that the performance of this approach is inferior compared to the absolute dual quadric formulation when applied in multiple views. This is mainly because that the Kruppa equations do not explicitly enforce the degeneracy of the dual quadric and the fact that there is a common supporting plane for the absolute conic over multiple views. Furthermore, when the motion of the camera is purely translational, the Kruppa equations reduce to a tautology. The ambiguities of calibration by Kruppa equations are discussed in [73].

3) Stratified Self Calibration: In order to upgrade from a projective reconstruction to a metric reconstruction, we need to estimate both the calibration matrix and the position of the plane at infinity or the infinite homography. The calibration approach using the dual absolute quadric estimates both quantities at the same time: the DIAC contains the calibration information and the null space of Q_∞^* represents transformed position of the plane at infinity. The advantage of such approach is obvious as it can directly upgrade from a projective reconstruction to a metric reconstruction. This convenience, however, comes at a price. As we have seen in the discussion of calibration by DIAC, quite a number of parameters need to be estimated at the same time. Doing so will no doubt lead to inaccuracies and numerical instabilities. An alternative would be to go one step at a time, from projective to affine and then to metric. This is the intuition behind the stratified calibration approach.

The most difficult part of stratified approach is the estimation of position of π_∞. Having the transformed position of π_∞ is equivalent to having an affine reconstruction since applying the projective transform

$$H = \begin{bmatrix} I|\mathbf{0} \\ \pi_\infty^T \end{bmatrix} \tag{34}$$

to the projective reconstruction correctly places π_∞ at (0, 0, 0, 1). We know that the only class of projective transform that fixes the position of π_∞ is the class of affine transformations.

There are several methods for finding the transformed position of π_∞. Many of them require prior knowledge of the scene structure such as parallel lines and vanishing points. Some methods that do not have this requirement are those that employ the *modulus constraint* [74]. The modulus constraint is a constraint on the position of π_∞ in the form of a polynomial equation. Assuming constant internal parameters, a projection matrix $P = [A|\mathbf{a}]$ in the projective reconstruction relates to its metric reconstruction $P_M = K[R|-RC]$ by

$$A - \mathbf{a}\mathbf{p}^T = \mu K R K^{-1} \tag{35}$$

where $\mathbf{p}$ is the position of π_∞. We see that the right-hand side is conjugate to a rotation matrix which has eigenvalues $(1, e^{i\theta}, e^{-i\theta})$. Then, the left hand side must have eigenvalues $(\mu, \mu e^{i\theta}, \mu e^{-i\theta})$ and the characteristic polynomial is

$$\begin{aligned} det(\lambda I - A + \mathbf{a}\mathbf{p}^T) &= (\lambda - \lambda_1)(\lambda - \lambda_2)(\lambda - \lambda_3) && (36)\\ &= \lambda^3 - f_1\lambda^2 + f_2\lambda - f_3 && (37)\end{aligned}$$

where λ_i are the eigenvalues and

$$f_1 = \lambda_1 + \lambda_2 + \lambda_3 = \mu(1 + 2\cos\theta) \tag{38}$$

$$f_2 = \lambda_1\lambda_2 + \lambda_1\lambda_3 + \lambda_2\lambda_3 = \mu^2(1 + 2\cos\theta) \tag{39}$$

$$f_3 = \lambda_1\lambda_2\lambda_3 = \mu^3. \tag{40}$$

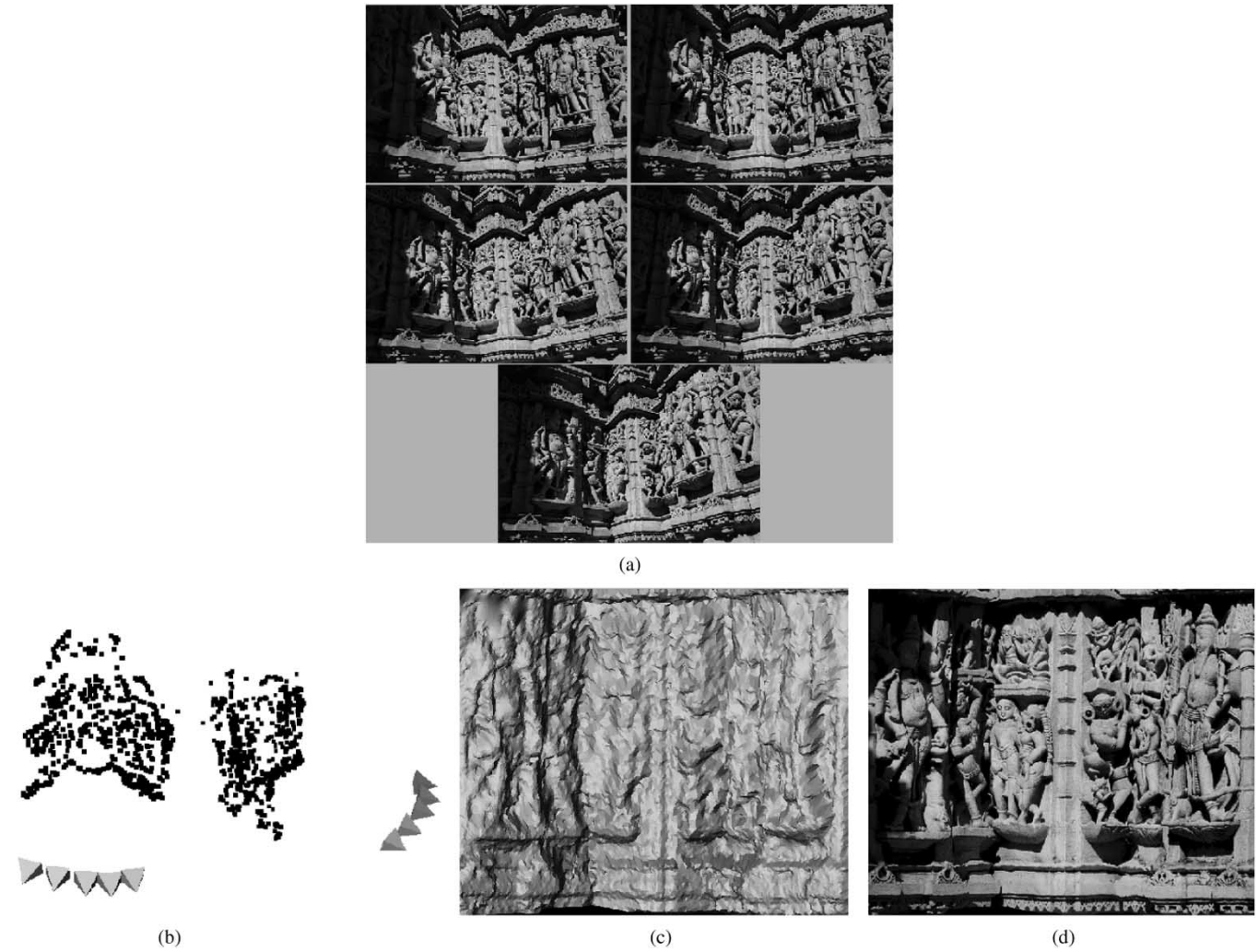


Fig. 8. Reconstruction results using stratified self calibration [74]: (a) input images, (b) reconstructed points and cameras, (c) shaded reconstruction, and (d) textured reconstruction.

Eliminating θ, we arrive at the modulus constraint

$$f_3 {f_1}^3 = {f_2}^3. \tag{41}$$

The elements of $\mathbf{p}$ appear only linearly in (41). Thus, the modulus constraint can be written as a quartic polynomial in the elements of $\mathbf{p}$. Upon solving this polynomial, we have the transformed coordinate of π_∞.

Having computed $\mathbf{p}$, a linear algorithm now exists to compute the calibration matrix K. As evident in (35), the infinite homography from a camera $[I|\mathbf{0}]$ and the camera $[A|a]$ is given by

$$H_\infty = A - \mathbf{a}\mathbf{p}^T. \tag{42}$$

Since the absolute conic lies on the plane at infinity and its image is invariant between different views, we have

$$\omega^* = H_\infty \omega^* H_\infty^T. \tag{43}$$

From (43), we can write six linear equation in the independent elements of the symmetric matrix ω^* and obtain the homogeneous linear form

$$A\mathbf{c} = 0 \tag{44}$$

where A is a 6×6 matrix formed by the elements of H_∞ and $\mathbf{c}$ is the dual conic ω^* written in vector form. Since A has rank at most four, we need more than two views to uniquely compute $\mathbf{c}$. Reconstruction results from a system implemented by Pollefeys *et al.* using the stratified self-calibration method is shown in Fig. 8.

4) Degenerate Camera Configurations: The theoretical derivation of the self-calibration equations naturally opens up questions the question of whether self calibration and metric reconstruction can be performed on any video sequence. Unfortunately, the answer to this question is no. There are distinct classes of motion sequences for which self calibration is ambiguous. We refer to these motion sequences as *critical motion sequences*. Furthermore, if the selected feature points lie on a surface of certain ruled quadric and the camera is at a certain location with respect to this surface, then 3-D reconstruction is ambiguous. We call such surfaces *critical surfaces*. It should be made clear that these two concepts are not equivalent: the inability to uniquely reconstruct a scene do not imply that an accurate and unique calibration of the camera cannot be obtained.

The probability of having all the detected features lying on a critical surface is so small that this case can usually be safely

TABLE II
COMPARISON OF ONLINE CALIBRATED RECONSTRUCTION ALGORITHMS

	Required Scene Constraint	Camera Placement	Min. number of views	Require Proj. Recon.
Vanishing Point	Yes	Arbitrary	2	No
DIAC	No	Degenerate Config.	3	Yes
Kruppa Equations	No	Degenerate Config.	2	No
Stratified Calibration	No	Degenerate Config.	3	Yes

ignored in practice [75]. There are, however, many critical motion sequences that naturally arise from various modeling and image acquisition applications and thus a careful study of their properties is warranted. The basic observation that made self calibration possible is that the image of the absolute conic is invariant under camera motion given that the intrinsic parameters of the camera are constant. Situations in which critical sequences occur are exactly those when at least one other conic, besides the absolute conic, have the same projected image in all frames. To continue the discussion, we need to give some definitions. A *virtual quadric* is one in which there are no real points on it. A *proper quadric* is a quadric whose matrix has nonzero determinant. Formally, we say that a sequence S is critical if there exists a proper virtual conic Φ, distinct from Ω, such that its projection is the same in all frames of S. A sequence S critical for metric reconstruction does not imply that it is critical for affine reconstructions. A sequence S is critical for affine reconstructions if S is critical for a proper virtual conic whose supporting plane is not π_∞.

A complete catalog of critical motion sequences is given in [76]. Self-calibration algorithms for special critical motion sequences are given in [77]–[79]. A method of affine reconstruction of scenes with purely translating cameras is given in [80].

D. Upgrading to Metric Reconstruction

A metric reconstruction differs from the projective reconstruction by a homography $\mathbf{H}$. The form of the homography is given in [4] as

$$H = \begin{bmatrix} K^1 & \mathbf{0} \\ -\mathbf{p}^T K^1 1 & \end{bmatrix} \tag{45}$$

where K^1 is the calibration matrix of the first image and $\mathbf{p}$ is the position of the plane at infinity in the projective reconstruction.

For calibration algorithms that employ the vanishing points, the vector $\mathbf{p}$ can be computed from vanishing point correspondence in the two views. Let the vanishing points in the first view be $\{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3\}$ and those in the second view be $\{\mathbf{v}'_1, \mathbf{v}'_2, \mathbf{v}'_3\}$. The correspondence problem can be solved using the SVD method suggested by Pilu [26]. Then, the corresponding vanishing points can be triangulated to obtain its 3-D coordinate in this projective reconstruction. Hartley [4] describes a variety of triangulation algorithms that are suitable for this purpose. Let the triangulated position of the vanishing points be $\{\mathbf{V}_1, \mathbf{V}_2, \mathbf{V}_3\}$. The normal $\mathbf{N}$ to the plane at infinity in this projective reconstruction is

$$\mathbf{N} = (\mathbf{V}_2 - \mathbf{V}_1) \times (\mathbf{V}_3 - \mathbf{V}_1). \tag{46}$$

The offset of this plane is then $w = \mathbf{N} \cdot \mathbf{V}_1$ and $\mathbf{p} = \mathbf{N}/w$. The components of the homograhy can now be assembled to obtain H. An Euclidean reconstruction can be obtained as $\{P_i H, H^{-1}\mathbf{X}_j\}$.

For algorithms that compute the dual image of the absolute quadric, the homography H needed to upgrade the projective reconstruction to a metric reconstruction can be obtained by decomposing Q^*_∞ using eigenvalue decomposition into $H\tilde{I}H^T$, where $\tilde{I}$ is the diagonal matrix $\mathrm{diag}(1,1,1,0)$. Then, apply H to the cameras and H^{-1} to the points to obtain the metric reconstruction. A similar procedure can be used for the stratified calibration methods, where the required homography H can be assembled using the computed π_∞ and K. For Kruppa equation-based methods, however, no prior projective reconstruction is computed. Therefore, after estimating the calibration matrix K, we can follow the steps in the calibrated reconstruction block diagram to obtain a metric reconstruction of the scene.

E. Comparison of Online Calibrated Reconstruction Algorithms

For the same reason stated in the comparison of calibrated reconstruction algorithms, quantitative comparison of online calibrated algorithms in terms of reconstruction accuracy cannot be easily performed. In this section, we again provide qualitative comparisons of online calibrated algorithms in terms of the need for special scene constraints, camera placement, minimum number of views required for reconstruction, and the need for an initial projective reconstruction. However, since the online calibrated algorithms differs mainly in their calibration techniques, this is essentially a comparison of the various online camera calibration methods. The comparison is shown in Table II.

The vanishing-point-based calibration method is best applied when parallel lines exist in all three dominant directions in a 3-D scene. These methods offer closed form expressions for the internal camera parameters. They have been successfully applied to many architectural reconstruction applications where the 3-D shapes of buildings are computed. However, this class of methods is very sensitive to the estimated location of the vanishing points. Automatic estimation of vanishing point is usually not accurate enough and often leads to incorrect camera calibrations.

The other online calibration approaches take advantage of the geometric constraints available from the image sequence. They offer a more flexible framework than the vanishing-point-based methods. The DIAC method should be used when a subset of the internal parameters is known and we only wish to solve for the remaining ones. For example, if we know that the location

of the principal point, the aspect ratio is unity, and that the skew is 0, then four linear constraints are available on Q^*_∞ from each view. We can find a unique solution if there are three or more views available. It can also be applied when we have no knowledge about the individual internal parameters, but only know that they are constant throughout all views. In that case, a total of ten equations in the entries of Q^*_∞ results from three views for which a unique solution can be found. The disadvantage of this method is that it assumes that Q^*_∞ is positive semi-definite. This condition can often be violated because of noisy data and thus leading to spurious calibrations.

The approach using Kruppa equations are two view constraints that require only the fundamental matrix F to be known. This method should be used when the internal camera parameters are constant across views. However, if the motion between the two views are purely translational, the Kruppa equations provide no constraint on ω^*. In addition, since the performance of this approach is usually inferior to those of the DIAC method when applied to more than two views, it should only be used when exactly two views are available.

The stratified calibration method improves upon the DIAC method by going one step at a time from projective to affine and then to metric reconstruction. This method is slightly more complex than the DIAC method, but it often offers better quality of solutions. However, this method is ambiguous when the set of camera motions are all about a single axis, the solution is a one parameter family of calibration matrices.

VI. Conclusion

In this paper, we have surveyed a number of 3-D reconstruction algorithms that exploit the motion parallax. We divided the algorithm into two large categories depending on whether a prior calibration of the camera is required. Under the pre-calibrated reconstruction category, we discussed image-based algorithms that rely on feature correspondence or dense stereo matching to compute the reconstruction of the scene, voxel-based algorithms that project the scene filling voxels back onto each view to determine its visibility, and object-based algorithms that formulates the 3-D reconstruction algorithm as a level-set evolution problem in which a system of partial differential equations gradually converges to the object to be reconstructed. On the online calibrated side, we discussed algorithms that take advantage of parallel structures in the scene to compute the vanishing points which aid greatly in computing closed form solutions of the camera's calibration parameters. In addition, we also reviewed more flexible self-calibration methods that do not rely on specific prior scene structure, but only on inherent geometric constraints such as the position of the absolute conic. Furthermore, camera configurations in which the absolute conic-based self calibration breaks down are also surveyed and presented.

There are still a number of open issues in performing motion-parallax-based 3-D reconstruction. On the pre-calibrated side, feature correspondence and dense stereo matching has always been open research topics for image-based methods. The application of global optimization techniques to these problems are currently being investigated by an increasing number of researchers. For voxel and level-set-based approaches, efficient data representations and the ability to handle non-Lambertian surfaces are interesting research topics. On the online calibrated side, much effort is devoted to finding numerically stable solutions of camera parameters. Also, research on methods that detect degenerate camera configurations is extremely useful in practice in order to avoid spurious calibrations caused by these configurations.

Because of the vast amount of literature available on this topic, the algorithms presented in this survey is by no means exhaustive. However, we believe that the reviews of the algorithms included in this paper provides a good starting point for researcher entering this field in computer vision as well as giving the veteran researchers a handy reference to the subject.

References

[1] D. Marr, *Vision*. New York: W. H. Freeman, 1982.

[2] E. Adelson and J. Bergen, "The plenoptic fucntion and the elements of early vision," in *Computational Models of Visual Processing*, M. Landy and J. A. Movshon, Eds. Cambridge, MA: MIT Press, 1991, pp. 3–20.

[3] S. Baker, T. Sim, and T. Kanade, "When is the shape of a scene unique given its lightfield: A fundamental theorem of 3D vision?," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 25, pp. 100–109, Jan. 2003.

[4] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge, U.K.: Cambridge Univ. Press, 2000.

[5] O. Faugeras, Q. T. Luong, and T. Papadopoulo, *The Geometry of Multiple Images*: MIT Press, 2001.

[6] R. I. Hartley, "In defense of the eight-point algorithm," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 19, pp. 580–593, Oct. 1997.

[7] R. Tsai, "A versatile camera calibration technique for high accuracy 3D machine vision metrology using off the shelf TV cameras and lenses," *IEEE J. Robot. Automat.*, vol. RA-3, no. 4, pp. 323–344, 1987.

[8] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 22, pp. 1330–1334, Nov. 2000.

[9] J.-Y. Bouguet. (2004) Camera Calibration Toolbox for Matlab. [Online]. Available: http://www.vision.caltech.edu/bouguet/calib_doc/

[10] Intel Corporation. (2004) Intel Open Source Computer Vision Library. [Online]. Available: http://www.intel.com/research/mrl/research/opencv/

[11] D. D. Morris, K. Kanatani, and T. Kanade, "Gauge fixing for accurate 3D estimation," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2001, pp. 343–350.

[12] R. Hartley and P. Sturm, "Triangulation," *Comput. Vis. Image Understanding*, vol. 68, no. 2, pp. 146–157, 1997.

[13] H. P. Moravec, "Toward automatic visual obstacle avoidance," in *Proc. Int. Joint Conf. Artificial Intelligence*, 1977, pp. 584–584.

[14] ——, "Visual mapping by a robot rover," in *Proc. 6th Int. Joint Conf. Artificial Intelligence*, 1977, pp. 598–600.

[15] C. G. Harris and M. Stephens, "A combined corner and edge detector," in *Proc. 4th Alvey Vision Conf.*, 1988, pp. 147–151.

[16] P. R. Beaudet, "Rotational invariant image operator," in *Proc. Int. Conf. Pattern Recognition*, 1978, pp. 579–583.

[17] L. Kitchen and A. Rosenfeld, "Gray-level corner detection," *Pattern Recognit. Lett.*, pp. 95–102, 1982.

[18] L. Dreschler and H. H. Nagel, "Volumetric model and 3D trajectory of a moving car derived from monocular tv-frame sequence of a street scene," *Comput. Vis., Graph., Image Processing*, vol. 20, no. 3, pp. 199–228, 1981.

[19] H. H. Nagel, "Principles of (low level) computer vision," in *Fundamentals in Computer Understanding: Speech and Vision*, J. P. Haton, Ed. Cambridge, U.K.: Cambridge Univ. Press, 1987, pp. 113–139.

[20] O. A. Zuniga and R. M. Haralick, "Corner detection using the facet model," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 1983, pp. 30–37.

[21] S. M. Smith and J. M. Brady, "Susan—A new approach to low level image processing," *Int. J. Comput. Vis.*, vol. 23, no. 1, pp. 45–78, 1997.

[22] F. Mokhtarian and R. Suomela, "Robust image corner detection through curvature scale space," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 20, pp. 1376–1381, Dec. 1998.

[23] C. Tomasi and T. Kanade, "Detection and tracking of point features,", Tech. Rep. CMU-CS-91-I32, 1991.

[24] Z. Zhang, R. Deriche, O. Faugeras, and Q.-T. Luong, "A robust technique for matching two uncalibrated images through the recovery of unknown epipolar geometry," *Artific. Intell. J.*, vol. 78, pp. 87–119, 1995.

[25] D. Marr and T. Poggio, "Cooperative computation of stereo disparity," *Science*, vol. 194, pp. 283–287, 1976.

[26] M. Pilu, "A direct method for stereo correspondence based on singular value decomposition," in *Proc. Int. Conf. Computer Vision and Pattern Recognition*, San Juan, Puerto Rico, 1997, pp. 261–266.

[27] H. H. Baker and T. O. Binford, "Depth from edge and intensity based stereo," in *Proc. 7th Int. Joint Conf. Artificial Intelligence*, 1981, pp. 631–636.

[28] Y. Ohta and T. Kanade, "Stereo by intra- and inter-scanline search using dynamic porgramming," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-7, pp. 139–145, Feb. 1985.

[29] Z. N. Li, "Stereo correspondence based on line matching in Hough space using dynamic programming," *IEEE Trans. Syst., Man, Cybern.*, vol. 24, pp. 144–152, 1994.

[30] J. L. Barron, D. J. Fleet, and S. S. Beuchemein, "Performance of optical flow techniques," *Int. J. Comput. Vis.*, vol. 12, no. 1, pp. 43–77, 1994.

[31] D. J. Heeger and A. D. Jepson, "Subspace methods for recovering rigid motion i: Algorithm and implementation," *Int. J. Comput. Vis.*, vol. 7, no. 2, pp. 95–117, 1992.

[32] M. Zucchelli, "Optical flow based structure from motion," Ph.D. thesis, Royal Institute of Technology, 2002.

[33] O. Faugeras, *Three-Dimensional Computer Vision: A Geometric Viewpoint*. Cambridge, MA: MIT Press, 1993.

[34] S. Z. Li, *Markov Random Field Modeling in Computer Vision*. New York: Springer, 1995.

[35] V. Kolmogorov and R. Zabih, "Computing visual correspondence with occlusions using graph cuts," in *Proc. Int. Conf. Computer Vision*, Vancouver, BC, Canada, 2001, pp. 508–515.

[36] Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max-flow algorithms for energy minimization in computer vision," in *Proc. 3rd Int. Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*, Sophie Antipolis, France, 2001, pp. 359–374.

[37] V. Kolmogorov and R. Zabih, "What energy functions can be minimized via graph cuts?," in *Proc. European Conf. Computer Vision*, vol. III, Copenhagen, Denmark, 2002, pp. 65–82.

[38] D. Greig, B. Porteous, and A. Seheult, "Exact maximum a posterior estimation for binary images," *J. Roy. Statist. Soc.*, vol. 51, no. 2, pp. 271–279, 1989.

[39] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *Int. J. Comput. Vis.*, vol. 47, no. 1–3, pp. 7–42, 2002.

[40] V. Kolmogorov and R. Zabih, "Multi-camera scene reconstruction via graph cuts," in *Proc. European Conf. Computer Vision*, 2002.

[41] G. Slabaugh, B. Culbertson, T. Malzbender, and R. Schafer, "A survey of methods for volumetric scene reconstruction from photographs," in *Int. Workshop Volume Graphics*, Stony Brook, NY, 2001.

[42] W. Martin and J. K. Aggarwal, "Volumetric description of objects from multiple views," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-5, pp. 150–158, Feb. 1983.

[43] C. H. Chien and J. K. Aggarwal, "A volume/surface representation," in *Proc. Int. Conf. Pattern Recognition*, 1984.

[44] ——, "Volume/surface octrees for the representation of three dimensional objects," *Comput. Vis., Graph., Image Process.*, vol. 36, no. 1, pp. 100–113, 1986.

[45] M. Potmesil, "Generating octree models of 3D objects from their silhouettes in a sequence of images," *Comput. Vis., Graph., Image Process.*, vol. 40, no. 1, pp. 1–29, 1987.

[46] R. Szeliski, "Rapid octree representation from image sequences," *Comput. Vis., Graph., Image Process.*, vol. 58, no. 1, pp. 23–32, 1993.

[47] D. Snow, P. Viola, and R. Zabih, "Exact voxel occupancy with graph cuts," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2000.

[48] S. Seitz and C. Dyer, "Photorealistic scene reconstruction by voxel coloring," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, San Juan, Puerto Rico, 1997, pp. 1067–1073.

[49] ——, "Photorealistic scene reconstruction by voxel coloring," *Int. J. Comput. Vis.*, vol. 35, no. 2, pp. 151–173, 1999.

[50] A. Prock and C. Dyer, "Toward real-time voxel coloring," in *Proc. DARPA Image Understanding Workshop*, Monterrey, CA, 1998, pp. 315–321.

[51] W. B. Culbertson, T. Malzbender, and G. Slabaugh, "Generalized voxel coloring," in *Proc. ICCV Workshop, Vision Algorithms Theory and Practice*, Corfu, Greece, 1999, pp. 67–74.

[52] O. Faugeras and R. Keriven, "Variational principles, surface evolution, PDE's, level set methods, and the stereo problem," *IEEE Trans. Image Processing*, vol. 7, pp. 336–344, 1998.

[53] L. Robert and R. Deriche, "Dense depth map reconstruction: A minimization and regularization approach which perserves discontinuities," in *Proc. European Conf. Computer Vision*, Cambridge, U.K., 1996, pp. 439–451.

[54] L. Robert, R. Deriche, and O. Faugeras, "Dense depth recovery from stereo images," in *Proc. European Conf. Artificial Intelligence*, Vienna, Austria, 1992, pp. 821–823.

[55] H. Jin, S. Soatto, and A. J. Yezzi, "Variational multiframe stereo in the presence of specular reflections," in *Proc. 1st Int. Symp. 3D Data Processing Visualization and Transmission*, Padova, Italy, 2002, pp. 626–630.

[56] ——, "Multi-view stereo beyond lambert," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, Madison, WI, 2002, pp. 171–179.

[57] Q. T. Luong and T. Vieville, "Canonical representations for the geometries of multiple projective views," *Comput. Vis. Image Understanding*, vol. 64, no. 2, pp. 193–229, Sept. 1996.

[58] C. Tomasi and T. Kanade, "Shape and motion from image streams under orthography: A factorization approach," *Int. J. Comput. Vis.*, vol. 9, no. 2, pp. 137–154, Nov. 1992.

[59] P. Sturm and W. Triggs, "A factorization based algorithm for multi-image projective structure and motion," in *Proc. European Conf. Computer Vision*, 1996, pp. 709–720.

[60] B. Caprile and V. Torre, "Using vanishing points for camera calibration," *Int. J. Comput. Vis.*, vol. 4, no. 2, pp. 127–139, 1986.

[61] S. T. Barnard, "Interpreting perspective images," *Artificial Intelligence*, vol. 21, pp. 435–462, 1983.

[62] L. Quan and R. Mohr, "Determining perspective structures using hierarchical hough transform," *Pattern Recognit. Lett.*, vol. 9, pp. 279–286, 1989.

[63] E. Lutton, H. Maitre, and J. Lopez-Krahe, "Contribution to the determination of vanishing points using Hough transform," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 16, pp. 430–438, Sept. 1994.

[64] M. J. Magee and J. K. Aggarwal, "Determining vanishing points from perspective images," *Comput. Vis., Graph., Image Process.*, vol. 26, pp. 256–267, Sept. 1984.

[65] C. Rother, "A new approach for vanishing point detection in architectural environments," in *Proc. British Machine Vision Conf.*, Bristol, U.K., 2000, pp. 382–391.

[66] F. A. van den Heuvel, "Vanishing point detection for architectural photogrammetry," *Int. Arch. Photogramm. Remote Sensing*, vol. XXXII, no. 5, 1998.

[67] D. Liebowitz and A. Zisserman, "Combining scene and auto-calibration constraints," in *Proc. Int. Conf. Computer Vision*, Corfu, Greece, 1999, pp. 293–300.

[68] W. Triggs, "Auto-calibration and the absolute quadric," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 1997, pp. 609–614.

[69] O. D. Faugeras and Q. T. Luong, "Camera self-calibration: Theory and experiments," in *Computer Vision—ECCV '92*, ser. LNCS-Series. New York: Springer-Verlag, 1992, vol. 588, pp. 321–334.

[70] S. J. Maybank and O. D. Faugeras, "A theory of self-calibration of a moving camera," *Int. J. Comput. Vis.*, vol. 8, no. 2, pp. 123–151, 1992.

[71] T. Vièville and D. Lingrand, "Using singular displacements for uncalibrated monocular vision systems," INRIA, Tech. Rep. 2678, 1995.

[72] R. I. Hartley, "Kruppa's equations derived from the fundamental matrix," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 19, pp. 133–135, Feb. 1997.

[73] P. Sturm, "A case against Kruppa's equations for camera self-calibration," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 22, pp. 1199–1204, Oct. 2000.

[74] M. Pollefeys and L. van Gool, "Stratified self-calibration with the modulus constraint," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 21, pp. 707–724, Aug. 1999.

[75] B. K. P. Horn, "Motion field are hardly ambiguous," *Int. J. Comput. Vis.*, pp. 259–274, 1987.

[76] P. Sturm, "Critical motion sequences for monocular self-calibration and uncalibrated euclidean reconstruction," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, 1997, pp. 1100–1105.

[77] M. Armstrong, A. Zisserman, and R. Hartley, "Self-calibration from image triplets," in *Proc. European Conf. Computer Vision*, 1996, pp. 3–16.

[78] R. I. Hartley, "Self-calibration from multiple views with a rotating camera," in *Proc. European Conf. Computer Vision*, 1994, pp. 471–478.

[79] C. Wiles and M. Brady, "Ground plane motion camera models," in *Proc. European Conf. Computer Vision*, 1996, pp. 238–247.

[80] T. Moons, L. van Gool, M. van Diest, and E. Pauwels, "Affine reconstruction from perspective image pairs," in *Darpa-Esprit Workshop on Applications of Invariants in Computer Vision*, 1993, pp. 249–266.

Ye Lu received the B.A.Sc. degree in computer science and the M.A.Sc. degree in electrical engineering from Simon Fraser University, Burnaby, BC, Canada, in 1995 and 1997, respectively, where he is currently working toward the Ph.D. degree in computer science, specializing in computer vision and multimedia information processing.

His research interests include multiple view geometry, active vision, and video object extraction and processing.

Jason Z. Zhang (M'97) received the B.Sc. degree in electronic engineering from Nanjing University of Posts and Telecom, Nanjing, China, in 1984, the M.Sc. degree in communications and electronic systems from Beijing University of Posts and Telecom, Beijing, China, in 1989, and the Ph.D. degree in signal and information processing from Beijing Jiaotong University, Beijing, China, in 1994.

He is a Research Officer with the Institute for Fuel Cell Innovation of National Research Council (NRC) of Canada. Previously, he was a NSERC Visiting Fellow with NRC Vancouver Innovation Center from 1999 to 2001. He was also with Nanyang Technological University, Singapore, and The Chinese University of Hong Kong, China, before joining the NRC. His research interests include multiview environment modeling and understanding, active vision for surveillance and robot navigation, video object extraction and representation, real-time multivideo-stream compression and communication, and imaging and analysis of materials microstructure. He has published many refereed papers in journals and conference proceedings.

Q. M. Jonathan Wu (M'92) received the Ph.D. degree in electrical engineering from the University of Wales, Swansea, U.K., in 1990.

He is a Research Officer and Group Leader at the Institute for Fuel Cell Innovation, National Research Council of Canada, Vancouver, BC. He is also an Adjunct Professor in the School of Engineering Science, Simon Fraser University, Burnaby, BC, Canada. From 1982 to 1984, he was a Lecturer at Shandong University. From 1992 to 1994, he was with the University of British Columbia as a Senior Research Scientist. His research interests include pattern recognition, image analysis, neural networks, robotics, intelligent control, and computer vision systems.

Dr. Wu was a recipient of the Visiting Fellowship of the BC Advanced Systems Institute. In 1990, he received one of the major awards in the control and automation field—the Best Control Applications Paper Prize, awarded at the 11th IFAC World Congress, held in Tallinn.

Ze-Nian Li (M'86) received the Bachelor's degree in electrical engineering from the University of Science and Technology of China and the Master's and Ph.D. degrees in computer sciences from the University of Wisconsin–Madison.

He is a Professor in the School of Computing Science, Simon Fraser University, Vancouver, BC, Canada. He has also been the Director of the School of Computing Science since 2001 and is the Director of the Vision and Media Laboratory. Previously, he was an Electronic Engineer in charge of design of digital and analogical systems. His current research interests include multimedia, computer vision, pattern recognition, image processing, and artificial intelligence. He has published over 80 refereed papers in journals and conference proceedings.