International Journal of Computer Vision 37(3), 231–258, 2000

Structure from Motion: Beyond the Epipolar Constraint

TOMÁŠ BRODSKÝ*, CORNELIA FERMÜLLER AND YIANNIS ALOIMONOS
Computer Vision Laboratory, Center for Automation Research, University of Maryland, College Park, MD 20742-3275, USA
tbr@philabs.research.philips.com
fer@cfar.umd.edu
yiannis@cfar.umd.edu

Abstract. The classic approach to structure from motion entails a clear separation between motion estimation and structure estimation and between two-dimensional (2D) and three-dimensional (3D) information. For the recovery of the rigid transformation between different views only 2D image measurements are used. To have available enough information, most existing techniques are based on the intermediate computation of optical flow which, however, poses a problem at the locations of depth discontinuities. If we knew where depth discontinuities were, we could (using a multitude of approaches based on smoothness constraints) accurately estimate flow values for image patches corresponding to smooth scene patches; but to know the discontinuities requires solving the structure from motion problem first. This paper introduces a novel approach to structure from motion which addresses the processes of smoothing, 3D motion and structure estimation in a synergistic manner. It provides an algorithm for estimating the transformation between two views obtained by either a calibrated or uncalibrated camera. The results of the estimation are then utilized to perform a reconstruction of the scene from a short sequence of images.

The technique is based on constraints on image derivatives which involve the 3D motion and shape of the scene, leading to a geometric and statistical estimation problem. The interaction between 3D motion and shape allows us to estimate the 3D motion while at the same time segmenting the scene. If we use a wrong 3D motion estimate to compute depth, we obtain a distorted version of the depth function. The distortion, however, is such that the worse the motion estimate, the more likely we are to obtain depth estimates that vary locally more than the correct ones. Since local variability of depth is due either to the existence of a discontinuity or to a wrong 3D motion estimate, being able to differentiate between these two cases provides the correct motion, which yields the "least varying" estimated depth as well as the image locations of scene discontinuities. We analyze the new constraints, show their relationship to the minimization of the epipolar constraint, and present experimental results using real image sequences that indicate the robustness of the method.

Keywords: 3D motion estimation, scene reconstruction, smoothing and discontinuity detection, depth variability constraint

1. Introduction

One of the biggest challenges of contemporary computer vision is to create robust and automatic procedures for recovering the structure of a scene given multiple views. This is the well-known problem of structure from motion (SFM) (Faugeras, 1992; Koenderink and van Doorn, 1991). Here the problem is treated in the differential sense, that is, assuming that a camera moving in an unrestricted rigid manner in a static environment continuously acquires images. In all existing approaches to this problem, the solution proceeds in two steps: first, the rigid motion between the

*Present address: Philips Research, 345 Scarborough Road, Briarcliff Manor, NY 10510.

views is recovered, and second, the motion estimate is used to recover the scene structure.

Traditionally, the problem has been treated by first finding the correspondence or optical flow and then optimizing an error criterion based on the epipolar constraint. Although considerable progress has been made in minimizing deviation from the epipolar constraint (Luong and Faugeras, 1996; Maybank, 1986, 1987), the approach is based on the values of flow, whose estimation is an ill-posed problem.

The values of flow are obtained by applying some sort of smoothing to the locally computed image derivatives. When smoothing is done in an image patch corresponding to a smooth scene patch, accurate flow values are obtained. When, however, the patch corresponds to a scene patch containing a depth discontinuity, the smoothing leads to erroneous flow estimates there. This can be avoided only if a priori knowledge about the locations of depth discontinuities is available. Thus, flow values close to discontinuities often contain errors (and these affect the flow values elsewhere), and when the estimated 3D motion (containing errors) is used to recover depth, it is unavoidable that an erroneous scene structure will be computed. The situation presents itself as a chicken-and-egg problem. If we had information about the locations of the discontinuities, we would be able to compute accurate flow and subsequently accurate 3D motion. Accurate 3D motion implies, in turn, accurate location of the discontinuities and estimation of scene structure. Thus 3D motion and scene discontinuities are inherently related through the values of image flow, and each needs the other in order to be better estimated. Researchers avoid this problem by attempting to first estimate flow using sophisticated optimization procedures that could account for discontinuities, and although such techniques provide better estimates, their performance often depends on the scene in view, they are in general very slow, and they require extensive resources (Geman and Geman, 1984; Marroquin, 1985; Mumford and Shah, 1985).

In this paper, instead of attempting to estimate flow at all costs before proceeding with structure from motion, we ask a different question: Would it be possible to utilize local image motion information, such as normal flow for example, to obtain knowledge about scene discontinuities which would allow better estimation of 3D motion? Or, equivalently, would it be possible to devise a procedure that estimates scene discontinuities while at the same time estimating 3D motion? We show here that this is the case and we present a novel algorithm for 3D motion estimation. The idea behind our approach is based on the interaction between 3D motion and scene structure that only recently has been formalized (Cheong et al., 1998). If we have a 3D motion estimate which is wrong and we use it to estimate depth, then we obtain a distorted version of the depth function. Not only do incorrect estimates of motion parameters lead to incorrect depth estimates, but the distortion is such that the worse the motion estimate, the more likely we are to obtain depth estimates that locally vary much more than the correct ones. The correct motion then yields the "least varying" estimated depth and we can define a measure whose minimization yields the correct egomotion parameters. The measure can be computed from normal flow only, so the computation of optical flow is not needed by the algorithm. Intuitively, the proposed algorithm proceeds as follows: first, the image is divided into small patches and a search for the 3D motion, which as explained in Section 3 takes place in the 2D space of translations, is performed. For each candidate 3D motion, using the local normal flow measurements in each patch, the depth of the scene corresponding to the patch is computed. If the variation of depth for all patches is small, then the candidate 3D motion is close to the correct one. If, however, there is a significant variation of depth in a patch, this is either because the candidate 3D motion is inaccurate or because there is a discontinuity in the patch. The second situation can be distinguished from the first by the fact that the distribution of the depth values inside the patch is bimodal with the two classes of values spatially separated. In such a case the patch is subdivided into two new ones and the process is repeated. When the depth values computed in each patch are smooth functions, the corresponding motion is the correct one and the procedure has at the same time given rise to the locations of a number of discontinuities. The estimates can then be used to compute the structure of the scene. However, to obtain a good reconstruction, information from successive flow fields has to be combined and various smoothing, optimization and model-building procedures have to be utilized. The rest of the paper formalizes these ideas and presents experimental results. Preliminary results have previously been published in Brodský et al. (1998c, 1999).

1.1. Organization of the Paper

Section 2 defines the imaging model and describes the equations of the motion field induced by rigid motion;

it also makes explicit the relationship between distortion of depth and errors in 3D motion. Section 3 is devoted to an outline of the approach taken here and the description of the algorithm. It also analyzes the constraints that are introduced and formalizes the relationship of the approach to algorithms utilizing the epipolar constraint. Section 4 generalizes the algorithm to the case of uncalibrated imaging systems. Section 5 illustrates the construction of scene models on the basis of the motion estimates using a short video sequence, and Section 6 describes a number of experimental results with real image sequences.

2. Preliminaries

We consider an observer moving rigidly in a static environment. The camera is a standard calibrated pinhole with focal length f and the coordinate system $OXYZ$ is attached to the camera, with Z being the optical axis.

Image points are represented as vectors $\mathbf{r} = [x, y, f]^T$, where x and y are the image coordinates of the point and f is the focal length in pixels. A scene point $\mathbf{R}$ is projected onto the image point

$$\mathbf{r} = f\,\frac{\mathbf{R}}{\mathbf{R}\cdot\hat{\mathbf{z}}} \tag{1}$$

where $\hat{\mathbf{z}}$ is the unit vector in the direction of the Z axis.

Let the camera move in a static environment with instantaneous translation $\mathbf{t}$ and instantaneous rotation $\boldsymbol{\omega}$ (measured in the coordinate system $OXYZ$). Then a scene point $\mathbf{R}$ moves with velocity (relative to the camera)

$$\dot{\mathbf{R}} = -\mathbf{t} - \boldsymbol{\omega}\times\mathbf{R} \tag{2}$$

The image motion field is then the usual (Horn and Weldon, Jr., 1988)

$$\begin{aligned}\dot{\mathbf{r}} &= -\frac{1}{(\mathbf{R}\cdot\hat{\mathbf{z}})}(\hat{\mathbf{z}}\times(\mathbf{t}\times\mathbf{r})) + \frac{1}{f}\,\hat{\mathbf{z}}\times(\mathbf{r}\times(\boldsymbol{\omega}\times\mathbf{r}))\\ &= \frac{1}{\mathbf{Z}}\,\mathbf{u}_{\mathrm{tr}}(\mathbf{t}) + \mathbf{u}_{\mathrm{rot}}(\boldsymbol{\omega})\end{aligned} \tag{3}$$

where Z is used to denote the scene depth $(\mathbf{R}\cdot\hat{\mathbf{z}})$, and $\mathbf{u}_{\mathrm{tr}}, \mathbf{u}_{\mathrm{rot}}$ are the directions of the translational and rotational flow respectively. Due to the scaling ambiguity, only the direction of translation (focus of expansion, FOE, or focus of contraction, FOC, depending on whether the observer is approaching or moving away from the scene) and the three rotational parameters can be estimated from monocular image sequences.

The next subsection introduces the main concept underlying the approach taken in this paper, the estimation of distorted structure from local image measurements on the basis of erroneous 3D motion estimates.

2.1. Depth Estimation from Motion Fields

The structure of the scene, i.e., the computed depth, can be expressed as a function of the estimated translation $\hat{\mathbf{t}}$ and the estimated rotation $\hat{\boldsymbol{\omega}}$. At an image point $\mathbf{r}$ where the normal flow direction is $\mathbf{n}$, the inverse scene depth can be estimated from (3) as

$$\frac{1}{\hat{Z}} = \frac{\dot{\mathbf{r}}\cdot\mathbf{n} - \mathbf{u}_{\mathrm{rot}}(\hat{\boldsymbol{\omega}})\cdot\mathbf{n}}{\mathbf{u}_{\mathrm{tr}}(\hat{\mathbf{t}})\cdot\mathbf{n}} \tag{4}$$

where $\mathbf{u}_{\mathrm{rot}}(\hat{\boldsymbol{\omega}}), \mathbf{u}_{\mathrm{tr}}(\hat{\mathbf{t}})$ refer to the estimated rotational and translational flow respectively.

Substituting into (4) from (3), we obtain

$$\frac{1}{\hat{Z}} = \frac{\frac{1}{Z}\mathbf{u}_{\mathrm{tr}}(\mathbf{t})\cdot\mathbf{n} - \mathbf{u}_{\mathrm{rot}}(\delta\boldsymbol{\omega})\cdot\mathbf{n}}{\mathbf{u}_{\mathrm{tr}}(\hat{\mathbf{t}})\cdot\mathbf{n}} \quad \text{or}$$

$$\frac{1}{\hat{Z}} = \frac{1}{Z}\,\frac{\mathbf{u}_{\mathrm{tr}}(\mathbf{t})\cdot\mathbf{n} - Z\mathbf{u}_{\mathrm{rot}}(\delta\boldsymbol{\omega})\cdot\mathbf{n}}{\mathbf{u}_{\mathrm{tr}}(\hat{\mathbf{t}})\cdot\mathbf{n}}$$

where $\mathbf{u}_{\mathrm{rot}}(\delta\boldsymbol{\omega})$ is the rotational flow due to the rotational error $\delta\boldsymbol{\omega} = (\hat{\boldsymbol{\omega}} - \boldsymbol{\omega})$. To make clear the relationship between actual and estimated depth we write

$$\hat{Z} = Z\cdot D \tag{5}$$

with

$$D = \frac{\mathbf{u}_{\mathrm{tr}}(\hat{\mathbf{t}})\cdot\mathbf{n}}{(\mathbf{u}_{\mathrm{tr}}(\mathbf{t}) - Z\mathbf{u}_{\mathrm{rot}}(\delta\boldsymbol{\omega}))\cdot\mathbf{n}}$$

hereafter termed the distortion factor. Equation (5) shows how wrong depth estimates are produced due to inaccurate 3D motion values. The distortion factor for any direction $\mathbf{n}$ corresponds to the ratio of the projections of the two vectors $\mathbf{u}_{\mathrm{tr}}(\hat{\mathbf{t}})$ and $\mathbf{u}_{\mathrm{tr}}(\mathbf{t}) - Z\mathbf{u}_{\mathrm{rot}}(\delta\boldsymbol{\omega})$ on $\mathbf{n}$. The larger the angle between these two vectors is, the more the distortion will be spread out over the different directions. Thus, considering a patch of a smooth surface in space and assuming that normal flow measurements are taken along many directions, a rugged (i.e., unsmooth) surface will be computed on the basis of wrong 3D motion estimates.

To give an example, we show the estimated depth for a sequence taken with a hand-held camera in our lab, which we will refer to throughout the paper as "the lab sequence"; one frame is shown in Fig. 1(a). For two different translations we estimate the rotation from the vectors perpendicular to the respective translational vectors, as explained in Section 3.1, and plot the estimated values of (4). Notice the reasonably smooth depth estimates for the correct FOE in Fig. 1(b), and compare with the sharp changes in the depth map (neighboring black and white regions) in Fig. 1(c).

The above observation constitutes the main idea behind our algorithm. For a candidate 3D motion estimate we evaluate the variability of estimated depth within each image patch. If the image patch corresponds to a smooth 3D scene patch, the correct 3D motion will certainly give rise to the overall smallest variability in the image patch. To obtain such a situation we attempt a segmentation of the scene on the basis of the estimated depth while at the same time testing the candidate 3D motions. In contrast to traditional methods that utilize optical flow, all computations are based on normal flow and we thus have available the full statistics of the raw data, so that a good segmentation is generally possible.

3. 3D Motion Estimation from One Flow Field for Calibrated Camera

There exists a lot of structure in the world and almost any scene can be thought of as a collection of smooth surface patches separated by abrupt discontinuities. Here the term "smoothness" is not used to mean

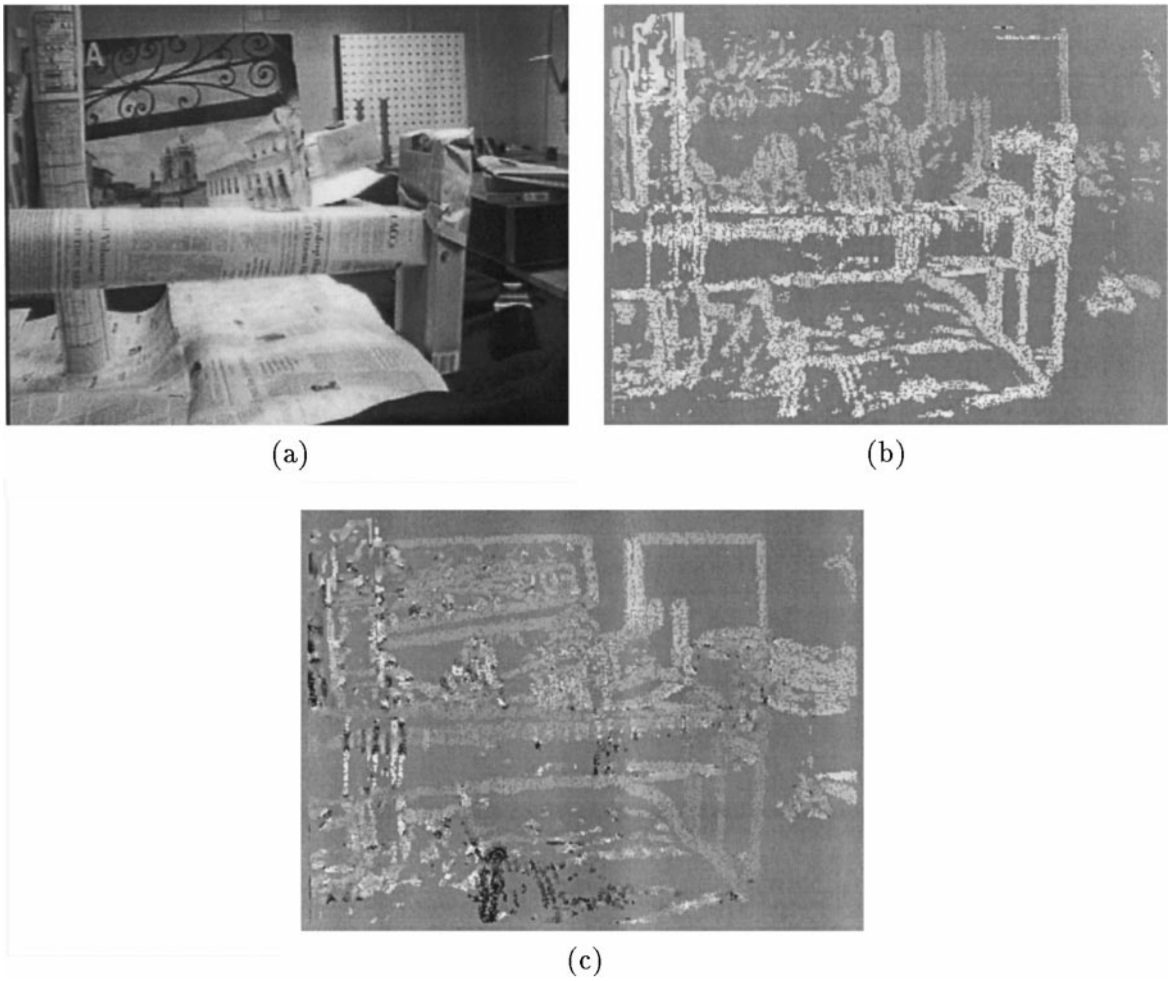

Figure 1. (a) One frame of the lab sequence. (b) Inverse depth estimated for the correct FOE ((397, −115) pixels from the image center). (c) Inverse depth for an incorrect FOE ((−80, 115) pixels from the image center). The gray-level value represents inverse estimated depth with mid-level gray shown in places where no information was available, white representing positive $1/\hat{Z}$ and black representing negative $1/\hat{Z}$.

differentiability, but rather to describe small depth changes within the individual surface patches.

Many previous approaches have used the assumption of locally smooth (constant, linear, or smoothly varying) scene depth. Without the detection of depth discontinuities, however, this assumption is not valid everywhere in the image. Explicit consideration of the depth discontinuities leads to a fundamental difference. If (and only if) we are able to detect the depth boundaries between surface patches, we no longer need to make smoothness *assumptions*; we are merely utilizing a property of the world which in the sequel we call the "patch variability constraint" or "depth variability constraint."

The significance of incorporating the discontinuities has long been understood, and in the past various efforts have been made to estimate smooth flow fields while at the same time detecting discontinuities (Heitz and Bouthemy, 1993; Murray and Buxton, 1987; Schunck, 1989; Spoerri and Ullman, 1987; Thompson et al., 1985). Previous work, however, is based on 2D image information only. Here, we attempt to bring in information about the 3D world, in particular the 3D motion and the depth of the scene, and to utilize it together with image measurements for segmentation.

In classical approaches the process of optical flow estimation, which involves smoothing, is separated from the process of 3D motion estimation and structure computation. After optical flow has been fitted to the image data, that is the normal flow, the information about the goodness of the fit is discarded and not considered in the later processes. By combining the processes of smoothing, 3D motion and structure estimation, we utilize this information in all the processes. The estimation of structure and motion thus becomes a geometrical and statistical problem. The challenge lies in understanding how the statistics of the input relate to the geometry of the image information and how to combine the constraints in an efficient way in the development of algorithms. Practical studies show that, because of the large number of unknowns, the computations cannot be carried out in a strictly bottom-up fashion, but have to be performed in a feedback loop. These considerations led to the development of the proposed algorithm.

The basic approach of the algorithm is quite simple. For a given candidate translation, we perform the following steps: estimate the rotation, perform depth segmentation, and finally evaluate a measure of depth variation taking into account the segmentation. A search in the space of translations for a minimum of the smoothness measure then yields the best 3D motion.

There have been some previous studies which use constraints on the 3D surfaces to relate the spatiotemporal image derivatives directly to 3D motion. Horn (1986) discusses a 3D motion estimation scheme which minimizes a measure consisting globally of departure from the optical flow constraint and locally of departure from scene smoothness, in particular the square Laplacian of inverse depth. Horn and Weldon, Jr. (1988) suggest an iterative algorithm for the case of pure translation which solves in alternating steps for translation and depth, and which estimates the translation by minimizing the error in depth. Bergen et al. (1992) describe an iterative algorithm for general rigid motion. Given an estimate of translation and rotation, they solve patch-wise for the inverse depth minimizing an error measure defined on the flow and then globally for the motion parameters, and they keep reiterating these two steps. The problem with these iteration approaches is that most often it is not possible to recover from initial errors in motion or depth. In the proposed algorithm there is no such iteration, and thus no convergence problem, as the depth is computed for every translation candidate. Another characteristic of our approach is the detection of depth discontinuities which have not been dealt with in other approaches. Finally, the study (Mendelsohn et al., 1997) needs to be mentioned, which also employs a patch-wise minimization with respect to inverse depth. However, in this paper the depth estimation is used not for motion estimation, but in an iterative multi-scale algorithm to estimate optical flow for an uncalibrated camera in rigid motion.

Next we explain in detail the different computations performed in the algorithm. First we describe a fast technique to estimate the rotation, which is used only to narrow the space of possible 3D motion estimates.

3.1. Projections of Motion Fields and Estimation of Rotation

The search for candidate 3D motions is achieved by searching for the translational component, i.e., the FOE. Given a candidate FOE, we need to estimate the rotation that best fits the image data together with that translation. One possibility is to examine normal flow vectors that are perpendicular to lines passing through the FOE, since these flow values do not contain any translational part. For this we need to formalize properties of motion fields projected onto particular directions. Indeed, by projecting the motion field vectors $\dot{\mathbf{r}}$ onto certain directions, it is possible

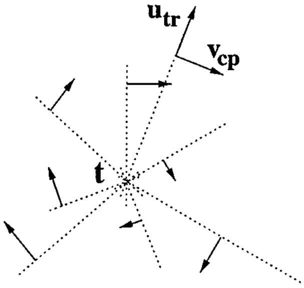

Figure 2. The copoint directions corresponding to **t**.

to gain some very useful insights (Brodský et al., 1998a; Fermüller, 1993; Fermüller and Aloimonas, 1995; Horn, 1987). Of particular interest are the *copoint* projections (Fermüller, 1993), where the flow vectors are projected onto directions perpendicular to a certain translational flow field (see Fig. 2). Let point $\hat{\mathbf{t}}$ be the FOE in the image plane of this translational field and consider the vectors emanating from $\hat{\mathbf{t}}$. Vectors perpendicular to such vectors are $\mathbf{v}_{\text{cp}}(\mathbf{r}) = \hat{\mathbf{z}} \times \mathbf{u}_{\text{tr}}(\hat{\mathbf{t}}) = \hat{\mathbf{z}} \times (\hat{\mathbf{z}} \times (\hat{\mathbf{t}} \times \mathbf{r}))$.

Let the camera motion be $(\mathbf{t}, \boldsymbol{\omega})$. The projection of the flow (3) onto $\mathbf{v}_{\text{cp}}$ is

$$\dot{\mathbf{r}} \cdot \frac{\mathbf{v}_{\text{cp}}}{\|\mathbf{v}_{\text{cp}}\|} = \frac{1}{\|\mathbf{v}_{\text{cp}}\|}\left(\frac{1}{Z} f\,(\mathbf{t} \times \hat{\mathbf{t}}) \cdot \mathbf{r} + (\boldsymbol{\omega} \times \mathbf{r}) \cdot (\hat{\mathbf{t}} \times \mathbf{r})\right) \tag{6}$$

In particular, if we let $\hat{\mathbf{t}} = \mathbf{t}$, the translational component of the copoint projection becomes zero and (6) simplifies into

$$\dot{\mathbf{r}} \cdot \frac{\mathbf{v}_{\text{cp}}}{\|\mathbf{v}_{\text{cp}}\|} = \frac{1}{\|\mathbf{v}_{\text{cp}}\|}\,(\boldsymbol{\omega} \times \mathbf{r}) \cdot (\hat{\mathbf{t}} \times \mathbf{r}) \tag{7}$$

Equation (7) can serve as a basis for estimating the rotation. Assume that the translation **t** is known. As long as there are some normal flow measurements in the direction of the appropriate copoint vectors, we can set up a linear least squares minimization to estimate $\boldsymbol{\omega}$. Specifically, at points with suitable normal flow directions we have equations

$$\dot{\mathbf{r}} \cdot \frac{\mathbf{v}_{\text{cp}}}{\|\mathbf{v}_{\text{cp}}\|} = \left(\frac{1}{\|\mathbf{v}_{\text{cp}}\|}((\mathbf{t} \times \mathbf{r}) \times \mathbf{r})^T\right)\boldsymbol{\omega}$$

that are linear in $\boldsymbol{\omega}$. Thus, theoretically the 3D motion can be estimated by fitting, for every candidate translation, the best rotation and checking the size of the residual of the fit.

It is easy to compute rotation from copoint vectors, but the dependence on the existence of suitable image measurements is crucial in practice, since such measurements may not be available. While a complete lack of measurements is usually not a problem, the number of copoint measurements for different candidate translations varies wildly and so does the reliability of the estimated rotation. Consequently, we have found in our experiments that fluctuations of the estimated rotation caused considerable difficulties not only for finding an accurate solution, but also for methods that could be used to speed up the computation. As the goodness of the rotational fit doesn't change smoothly between neighboring FOE candidates, it becomes difficult to speed up the search for the correct 3D motion with gradient descent methods, which is possible for the method based on the smoothness measure described next.

Nevertheless, the computation is simple and fast enough and the residual of the least squares estimate (scaled to account for the varying number of measurements) is usually sufficient to approximately locate the region of interest that most probably contains the correct translation. We thus use this measure first to narrow the space of solutions and then apply the more sophisticated criterion, explained in the next section, to only a limited region of candidate translations.

3.2. *The Criterion*

Consider a small image region $\mathcal{R}$ that contains a set of measurements $\dot{\mathbf{r}}_{\mathbf{i}}$ with directions $\mathbf{n}_i$. Given candidate motion parameters, we can estimate the inverse depth from (4) up to the overall scale ambiguity. To treat different patches equally, we normalize the estimated translation $\mathbf{u}_{\text{tr}}(\hat{\mathbf{t}})$ to be a unit vector in the middle of the region.

One possible measure of depth variation is the variance of the depth values, or, rather, the sum of squared differences of the depth values from a mean $1/\bar{Z}$:

$$\sum_i \left(\frac{\dot{\mathbf{r}}_i \cdot \mathbf{n}_i - \mathbf{u}_{\text{rot}}(\hat{\boldsymbol{\omega}}) \cdot \mathbf{n}_i}{\mathbf{u}_{\text{tr}}(\hat{\mathbf{t}}) \cdot \mathbf{n}_i} - \frac{1}{\bar{Z}}\right)^2 \tag{8}$$

Approaches that directly evaluate variations of estimated depth (or inverse depth) include (Brodský et al., 1988b, Horn and Weldon, Jr., 1988). However, depth estimates may present a numerical problem, since for many measurements the depth estimate is unreliable due to division by a small $\mathbf{u}_{\text{tr}} \cdot \mathbf{n}$. Thus we can either ignore many measurements where the depth estimate is unreliable, making comparisons between different translations difficult, or, alternatively, we have to deal with numerical instabilities. We choose a third possibility, defining a whole family of depth smoothness measures that includes the variance of estimated depth as well as many other measures.

In region $\mathcal{R}$ we compute

$$\Theta_0(\hat{\mathbf{t}}, \hat{\boldsymbol{\omega}}, \mathcal{R}) = \sum_i W_i \Bigg(\dot{\mathbf{r}}_i \cdot \mathbf{n}_i - \mathbf{u}_{\text{rot}}(\hat{\boldsymbol{\omega}}) \cdot \mathbf{n}_i - \left(\frac{1}{\hat{Z}}\right)(\mathbf{u}_{\text{tr}}(\hat{\mathbf{t}}) \cdot \mathbf{n}_i) \Bigg)^2 \tag{9}$$

where $1/\hat{Z}$ is the depth estimate minimizing the measure, i.e., not necessarily the mean $1/\bar{Z}$.

By setting $W_i = 1/(\mathbf{u}_{\text{tr}}(\hat{\mathbf{t}}) \cdot \mathbf{n}_i)^2$ we obtain (8). Another natural choice is $W_i = 1$. Then Θ_0 becomes the sum of squared differences between the normal flow measurements and the corresponding projections of the best flow obtained from the motion parameters. This measure has been used in Mendelsohn et al. (1997).

With different choices of W_i we can give greater emphasis either to the contributions from the copoint vectors (that is, the vectors perpendicular to the translational component), which are independent of depth, or to the vectors parallel to the translation, which are most strongly influenced by the depth. As long as we keep W_i bounded, criterion (9) nicely combines the contribution of the two perpendicular components. In our algorithm we use two sets of weights to achieve different numerical properties for the estimation of different parameters, as will be discussed later in the paper.

We first minimize Θ_0 with respect to $1/\hat{Z}$. The best inverse depth is

$$\frac{1}{\hat{Z}} = \frac{\sum_i W_i (\dot{\mathbf{r}}_i \cdot \mathbf{n}_i - \mathbf{u}_{\text{rot}}(\hat{\boldsymbol{\omega}}) \cdot \mathbf{n}_i)(\mathbf{u}_{\text{tr}}(\hat{\mathbf{t}}) \cdot \mathbf{n}_i)}{\sum_i W_i (\mathbf{u}_{\text{tr}}(\hat{\mathbf{t}}) \cdot \mathbf{n}_i)^2} \tag{10}$$

If more precision is required (in our experiments with small patches, the constant approximation worked quite well), we can model the scene patch by a general plane and use a linear approximation $1/\hat{Z} = \mathbf{z} \cdot \mathbf{r}$ (note that the third component of $\mathbf{r}$ is a constant f, so $\mathbf{z} \cdot \mathbf{r}$ is a general linear function in the image coordinates). Then we have

$$\frac{\partial \Theta_0(\hat{\mathbf{t}}, \hat{\boldsymbol{\omega}}, \mathcal{R})}{\partial \mathbf{z}} = \sum_i W_i (\mathbf{z} \cdot \mathbf{r_i})(\mathbf{u}_{\text{tr}}(\hat{\mathbf{t}}) \cdot \mathbf{n}_i)^2 \mathbf{r_i} - \sum_{\mathbf{i}} \mathbf{W_i} (\dot{\mathbf{r}}_i \cdot \mathbf{n} - \mathbf{u}_{\text{rot}}(\hat{\boldsymbol{\omega}}) \cdot \mathbf{n_i}) \times (\mathbf{u}_{\text{tr}}(\hat{\mathbf{t}}) \cdot \mathbf{n_i})\, \mathbf{r_i} = \mathbf{0} \tag{11}$$

which is a set of three linear equations for the three elements of $\mathbf{z}$.

Substituting (10) (or the solution of (11)) into (9), we obtain $\Theta_1(\hat{\mathbf{t}}, \hat{\boldsymbol{\omega}}, \mathcal{R})$, a second-order function of $\hat{\boldsymbol{\omega}}$. Notice that the computation can be performed symbolically even when $\hat{\boldsymbol{\omega}}$ is not known. This allows us to use the same equations to obtain both the rotation and a measure of depth variation.

To estimate $\hat{\boldsymbol{\omega}}$, we sum up all the local functions and obtain a global function:

$$\Theta_2(\hat{\mathbf{t}}, \hat{\boldsymbol{\omega}}) = \sum_{\mathcal{R}} \Theta_1(\hat{\mathbf{t}}, \hat{\boldsymbol{\omega}}, \mathcal{R}) \tag{12}$$

Finally, global minimization yields the best rotation $\hat{\boldsymbol{\omega}}$ and also a measure of depth variation for the apparent translation $\hat{\mathbf{t}}$:

$$\Phi(\hat{\mathbf{t}}) = \min_{\hat{\omega}} \Theta_2(\hat{\mathbf{t}}, \hat{\boldsymbol{\omega}}) \tag{13}$$

The computation of $\Phi(\hat{\mathbf{t}})$ involves two separate steps. First we estimate the best rotation $\hat{\boldsymbol{\omega}}$ and then we evaluate the global variability measure for the motion $(\hat{\mathbf{t}}, \hat{\boldsymbol{\omega}})$. In these two steps we choose different weights W_i in the function Θ_0.

To estimate the rotation, we use one set of weights W_i' defining Θ_0' and subsequently Θ_1' and Θ_2'. The rotation is computed as

$$\hat{\boldsymbol{\omega}}_0 = \arg\min_{\hat{\omega}} \Theta_2'(\hat{\mathbf{t}}, \hat{\boldsymbol{\omega}})$$

which amounts to solving an over-determined linear system in the unknown $\hat{\boldsymbol{\omega}}_0$. We also define Θ_0 using a different set of weights W_i. Functions Θ_1 and Θ_2 are derived from Θ_0 and the global depth variability function $\Phi(\hat{\mathbf{t}})$ becomes

$$\Phi(\hat{\mathbf{t}}) = \Theta_2(\hat{\mathbf{t}}, \hat{\boldsymbol{\omega}}_0) = \Theta_2\Big(\hat{\mathbf{t}}, \arg\min_{\hat{\omega}} \Theta_2'(\hat{\mathbf{t}}, \hat{\boldsymbol{\omega}})\Big) \tag{14}$$

Now we need to describe our choices of the weights. The best sources of information about $\hat{\boldsymbol{\omega}}$ are the copoint measurements, as they are independent of depth. Consequently, the copoint vectors should have more influence on Θ_0. However, the use of only these vectors would amount to direct evaluation of the depth variance, which means that in (8) the weighting factor for the copoint vectors tends toward infinity.

To prevent numerical instability, the weights W_i' should certainly be bounded. For the rotation estimation part, we use

$$W_i' = \frac{1}{\cos^2 \psi_i + \lambda} \tag{15}$$

where ψ_i is the angle between $\mathbf{u}_{\text{tr}}(\hat{\mathbf{t}})$ and $\mathbf{n}_i$, and λ is a small positive number.

After we use the weights W_i' to obtain the best rotation from Θ_2', we need to evaluate a global depth variation function to obtain $\Phi(\hat{\mathbf{t}})$. As we need to compare $\Phi(\hat{\mathbf{t}})$ values for different directions of $\hat{\mathbf{t}}$, we choose constant weights

$$W_i = 1 \tag{16}$$

Then the contribution to $\Phi(\hat{\mathbf{t}})$ of a single normal flow measurement is

$$\left(\dot{\mathbf{r}}_i \cdot \mathbf{n}_i - \mathbf{u}_{\text{rot}}(\hat{\omega}) \cdot \mathbf{n}_i - \left(\frac{1}{\hat{Z}}\right)(\mathbf{u}_{\text{tr}}(\hat{\mathbf{t}}) \cdot \mathbf{n}_i)\right)^2$$

and has a clear geometrical meaning; it is the squared difference between the normal flow and the corresponding projection of the best flow obtained from the motion parameters. More importantly, such squared errors can be easily compared for different directions of $\hat{\mathbf{t}}$.

3.3. Algorithm Description

The translation is found by localizing the minimum of function $\Phi(\hat{\mathbf{t}})$ described in (14). To obtain $\Phi(\hat{\mathbf{t}})$:

1. Partition the image into small regions, in each region compute $\Theta_0'(\hat{\mathbf{t}}, \hat{\omega}, \mathcal{R})$ using (15), and perform local minimization of $\hat{Z}$ (the computation is symbolic in the unknown elements of $\hat{\omega}$). After substitution, the function becomes $\Theta_1'(\hat{\mathbf{t}}, \hat{\omega}, \mathcal{R})$. At the same time, compute Θ_0 and Θ_1 using (16).
2. Add all the local functions $\Theta_1'(\hat{\mathbf{t}}, \hat{\omega}, \mathcal{R})$ and minimize the resulting $\Theta_2'(\hat{\mathbf{t}}, \hat{\omega})$ to obtain $\hat{\omega}_0$. Also add $\Theta_1(\hat{\mathbf{t}}, \hat{\omega}, \mathcal{R})$ to obtain $\Theta_2(\hat{\mathbf{t}}, \hat{\omega})$.
3. Estimate depth using $\hat{\mathbf{t}}$, $\hat{\omega}_0$ and perform patch segmentation.
4. Taking the segmentation into account, update both Θ_2 and Θ_2', use Θ_2' to compute a better rotational estimate, and the updated Θ_2 then provides $\Phi(\hat{\mathbf{t}})$.

After the segmentation, we recompute the error measure by enforcing low depth variability only within image regions that do not contain depth discontinuities. However, it is not necessary to re-derive Θ_1 for all the image regions as we need to compute the change of Θ_1 only for the regions that are segmented.

To find the minimum of Φ and thus the apparent translation, we perform a hierarchical search over the 2D space of epipole positions. In practice, the function Φ is quite smooth, that is small changes in $\hat{\mathbf{t}}$ give rise to only small changes in Φ. One of the reasons for this is that for any $\hat{\mathbf{t}}$, the value of $\Phi(\hat{\mathbf{t}})$ is influenced by all the normal flow measurements and not only by a small subset.

Furthermore, as explained before, $\Phi(\hat{\mathbf{t}})$ is computed only when the residual of fitting the copoint vectors (7) is small enough. A comparison of the copoint residual with $\Phi(\hat{\mathbf{t}})$ is given in Fig. 3. The copoint residuals give a very imprecise solution ((222, −85) in image coordinates) when compared to $\Phi(\hat{\mathbf{t}})$ (minimum at (397, −115)), but we can use them to quickly determine a smaller region (or a small number of candidate regions) that most probably contain the solution.

For most motion sequences, the motion of the camera does not change abruptly. Thus the translation does not change much between frames and a complete search has to be performed only for the first flow field. In the successive flow fields, we can search only in a smaller area centered around the previously estimated translation.

3.4. Patch Segmentation

Given a translation candidate $\hat{\mathbf{t}}$, minimization of the function $\Theta_2'(\hat{\mathbf{t}}, \hat{\omega})$ (without any segmentation) provides an initial estimate of the rotation $\hat{\omega}$. Subsequently, we

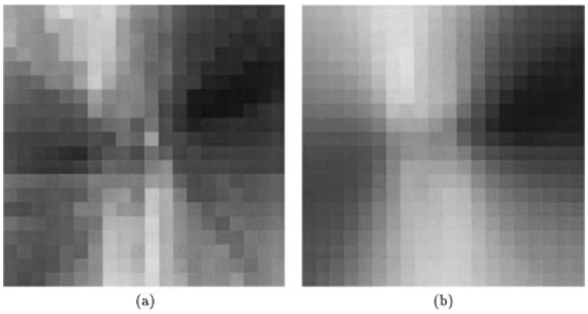

Figure 3. Two translation evaluation functions for the lab sequence. (a) Residual of copoint measurement fitting (notice the ruggedness of the function). (b) Function $\Phi(\hat{\mathbf{t}})$ with no depth segmentation applied. For display purposes both functions are shown after logarithmic scaling.

compute inverse scene depth using (4). The problem of finding depth discontinuities can also be formulated as the problem of finding edges in the depth image. However, we cannot directly use standard edge detection algorithms, because the data is sparse (we only have useful normal flow measurements at points with sufficiently high brightness gradient in the initial image) and the reliability of the depth estimates varies with the angle ψ between the normal flow direction, **n**, and the direction of estimated translational flow, $\mathbf{u}_{\mathrm{tr}}(\hat{\mathbf{t}})$. We thus use $|\cos\psi|$ as the reliability measure.

To better deal with sparse data of varying reliability, we apply a weighted median filter (Cormen et al., 1989) to the estimated inverse depth image. At each point we either have no data, or a depth estimate $1/\hat{Z}$ and a weight $|\cos\psi|$.

Consider a small neighborhood of a point containing measurements $1/\hat{Z}_j$ with weights w_j. The result of the weighted median filter is the depth $1/\hat{Z}_k$ such that

$$\sum_{1/\hat{Z}_j<1/\hat{Z}_k} w_j \le \frac{1}{2}\sum w_j \quad \text{and}$$

$$\sum_{1/\hat{Z}_j>1/\hat{Z}_k} w_j \le \frac{1}{2}\sum w_j$$

After median filtering, we apply a morphological growing operation. If there is no data available at a point, but there are some measurements (or data generated by the median filter) in its neighborhood, we store the weighted average of such measurements for that point. Note that we only add new data but do not change any previously available measurements.

A modified Canny edge detector is used to detect depth boundaries. After Gaussian smoothing and gradient estimation, we set to zero the gradient magnitude at all points where the smoothing or gradient operation involved a point with no available data. This prevents detection of edges between areas with data and areas with no data, as such edges are obviously not useful.

After the removal of spurious gradients, the most important depth edges are obtained by the usual thresholding and non-maximum suppression. The edge detection results for the lab sequence are shown in Fig. 4 for the correct translation and in Fig. 5 for an incorrect translation.

Individual patches are segmented based on the edge image. By segmenting a patch, we decrease its contribution to $\Phi(\hat{\mathbf{t}})$. Ideally, an image patch should be segmented if it contains two smooth scene surfaces separated by a depth discontinuity and the depth is estimated using the correct 3D motion. It would be the

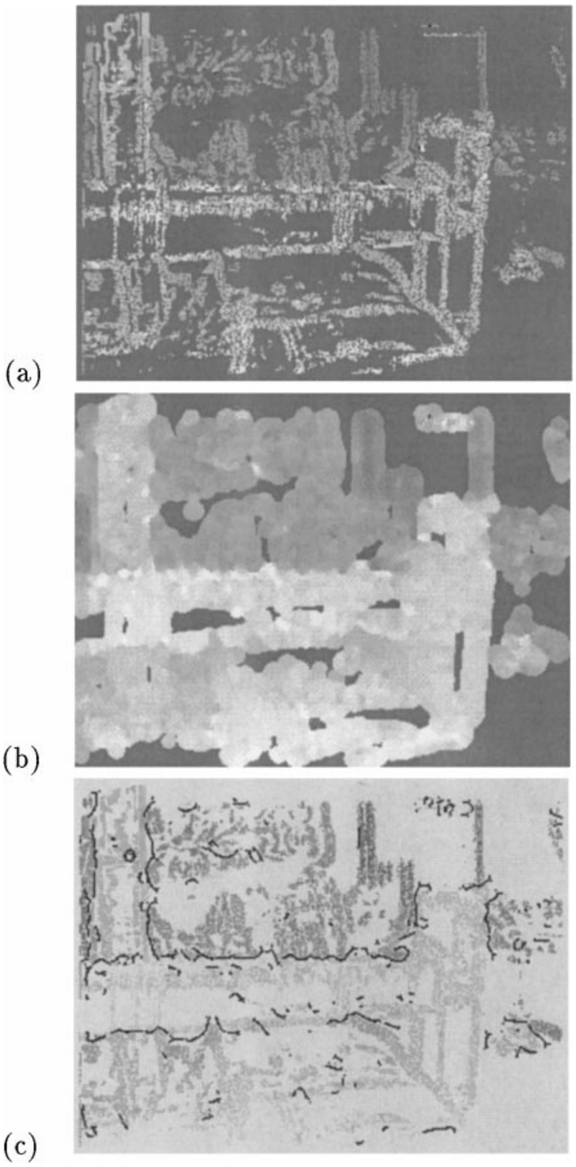

Figure 4. Patch segmentation for one frame of the lab sequence. (a) Estimated inverse depth using the best motion estimates before segmentation. The gray-level value represents inverse estimated depth with mid-level gray shown in places where no information was available, white representing positive $1/\hat{Z}$ and black representing negative $1/\hat{Z}$. (b) Depth image after median filtering and morphological growing. (c) Depth edges (drawn in black) found for the correct translation, overlaid on the computed depth. Increasing gray-level brightness represents increasing estimated depth. White represents areas where no information is available.

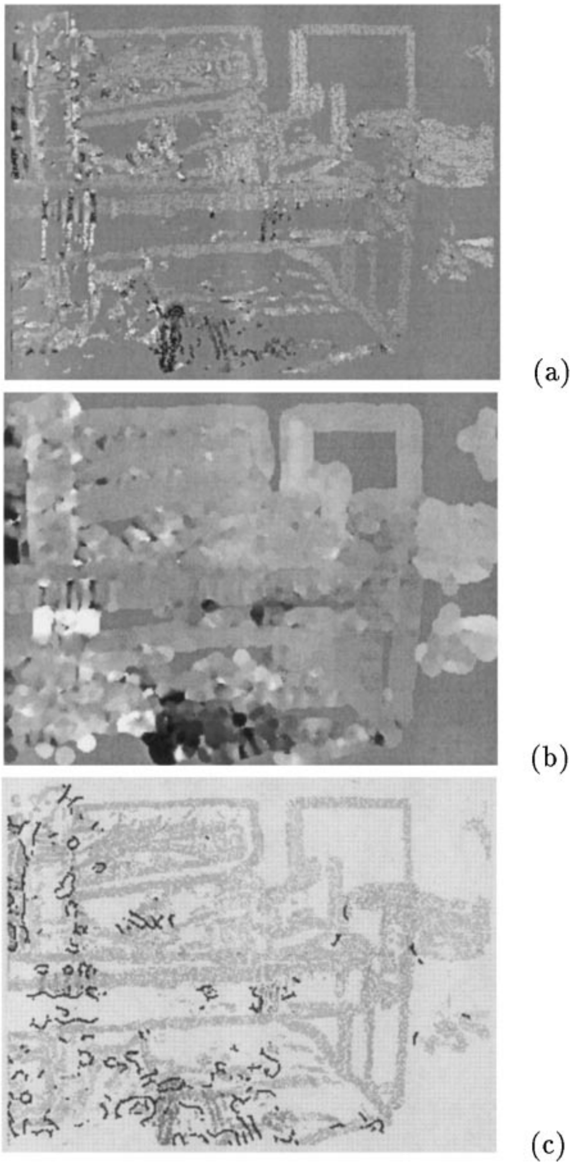

Figure 5. Patch segmentation for the same incorrect translation as in Fig. 1(c). (b) Depth image after median filtering and morphological growing. (c) Depth edges.

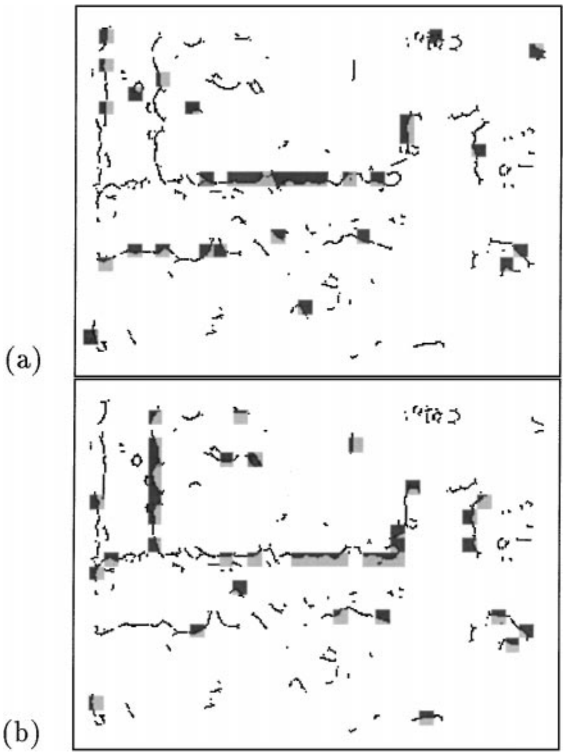

Figure 6. Detected depth edges (from Fig. 4) and all the image regions that were segmented by our algorithm. Each segmented patch is shown as a square; different gray-level values represent different parts of the segmented region. (a) and (b) show the results for the two overlapping sets of initial image regions.

best not to segment any patches that have no depth discontinuities, but this, of course, cannot be guaranteed.

We use a simple segmentation criterion. If the edge detection algorithm finds an edge that divides an image region into two coherent subregions, we split that region. On the other hand, we do not split regions containing more complicated edge structure (as may be expected for a distorted depth estimate, since the distortion depends on the normal flow direction).

In our implementation the initial image regions are 10×10 pixel squares. We use two sets of overlapping regions, one set shifted by 5 pixels in both the x and y directions with respect to the other. Figure 6 shows image regions that were actually segmented based on the edges computed in Fig. 4. Compare the results with Fig. 7, showing segmented patches for the incorrect translation.

This strategy can be expected to yield good patch segmentation. When the motion estimate is correct, the depth estimates are also correct (except for noise), and patches with large amounts of depth variation contain depth discontinuities.

For incorrect motions, the distortion factor depends on the direction of normal flow. While for any patch a splitting of the depth estimates into two groups decreases the error measure, it is highly unlikely that the two groups of measurements define two spatially

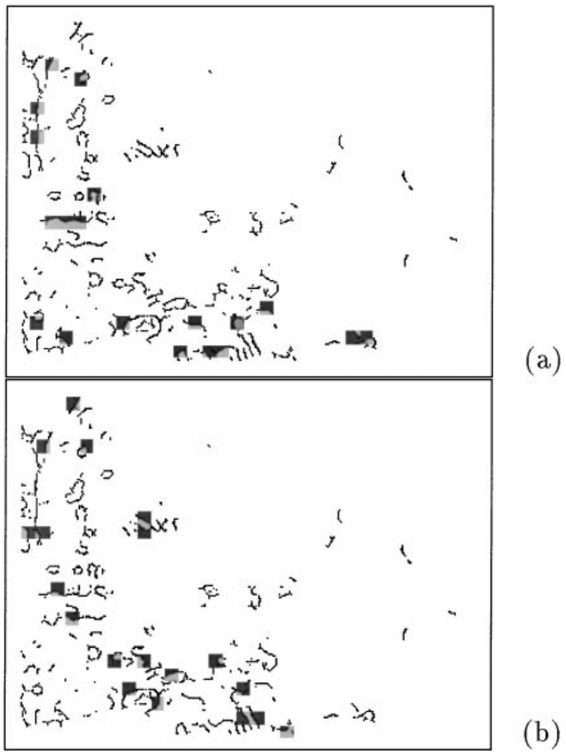

Figure 7. Depth edges and segmented image regions for the incorrect translation shown in Fig. 5.

coherent separate subregions. Consequently, several edges can be expected in such a patch and it will not be split by the algorithm.

The local results are used in a global measure and occasional segmentation errors are unlikely to change the overall results. For the segmentation to cause an incorrect motion to yield the smallest $\Phi(\hat{\mathbf{t}})$, special normal flow configurations would have to occur in many patches of the image.

One may ask whether depth segmentation using an incorrect estimate of rotation (due to incorrect information from regions with depth discontinuities) can actually improve the 3D motion estimation. The answer is almost always "yes."

We are primarily interested in the segmentations for translations that are close to the true solution. A good segmentation for such a candidate translation should lead to an improved solution. On the other hand, for incorrect translations we do not really care about the segmentation as long as the patches are not split indiscriminately. The segmentation criterion enforces spatial coherence to achieve this.

Now consider the important case of a translation $\hat{\mathbf{t}}$ that is close to the true solution. Computation of $\hat{\omega}$ uses information from the whole image and when we ignore depth discontinuities in the first stage of the algorithm, we may bias the solution. However, the bias is limited by giving more weight to the copoint measurements (these are independent of scene depth), and for many scenes most of the image regions correspond to smooth scene patches, so the bias should not be large. Given approximately correct motion parameters, most of the depth estimates are also approximately correct, with the exception of some small regions (whose positions are determined by the true and the estimated translation). Since the depth estimates are approximately correct we should be able to detect significant depth discontinuities, and these are the ones we are interested in.

3.5. Algorithm Analysis

In the following analysis the depth variability measure is compared to the measure used in epipolar constraint minimization. In particular, we analyze the depth variability measure for a single image region and show that it can be decomposed into two components, one component which measures the deviation of the patch from a smooth scene patch (that is, a fronto-parallel plane in our analysis), and a second component which constitutes a multiple of the epipolar constraint.

Consider the function Θ_0 in a small image region $\mathcal{R}$. The vectors $\mathbf{u}_{\text{tr}}(\hat{\mathbf{t}})$ and $\mathbf{u}_{\text{rot}}(\hat{\omega})$ are polynomial functions of image position $\mathbf{r}$ and can usually be approximated by constants within the region. We use a local coordinate system where $\mathbf{u}_{\text{tr}}(\hat{\mathbf{t}})$ is parallel to $[1, 0, 0]^T$. Without loss of generality we can write (in that coordinate system)

$$\begin{aligned} \mathbf{u}_{\text{tr}}(\hat{\mathbf{t}}) &= [1, 0, 0]^T \\ \mathbf{u}_{\text{rot}}(\hat{\omega}) &= [u_{\text{rx}}, u_{\text{ry}}, 0]^T \\ \mathbf{n}_i &= [\cos\psi_i, \sin\psi_i, 0]^T \\ u_{\text{n}_i} &= \dot{\mathbf{r}}_i \cdot \mathbf{n}_i \end{aligned} \tag{17}$$

First, let us consider the problem of fitting the best constant optical flow $(u_{\text{x}}, u_{\text{y}})$ to the measurements in $\mathcal{R}$ using the weights W_i, i.e., minimizing

$$\begin{aligned} &\sum_i W_i \left(u_{\text{n}_i} - (u_{\text{x}}, u_{\text{y}}) \cdot \mathbf{n}_i\right)^2 \\ &\quad = \sum_i W_i \left(u_{\text{n}_i} - u_{\text{x}} \cos\psi_i - u_{\text{y}} \sin\psi_i\right)^2 \end{aligned} \tag{18}$$

To simplify the notation, we define

$$\begin{aligned} S_{\text{cc}} &= \sum W_i \cos^2\psi_i & S_{\text{uc}} &= \sum W_i\, u_{\text{n}_i} \cos\psi_i \\ S_{\text{cs}} &= \sum W_i \cos\psi_i \sin\psi_i & S_{\text{us}} &= \sum W_i\, u_{\text{n}_i} \sin\psi_i \\ S_{\text{ss}} &= \sum W_i \sin^2\psi_i & S_{\text{uu}} &= \sum W_i\, u_{\text{n}_i}^2 \end{aligned} \tag{19}$$

The vector $(u_{\text{x}}, u_{\text{y}})$ minimizing (18) is obtained by differentiating (18) and solving the following linear system:

$$\begin{pmatrix} S_{\text{cc}} & S_{\text{cs}} \\ S_{\text{cs}} & S_{\text{ss}} \end{pmatrix}\begin{pmatrix} u_{\text{x}} \\ u_{\text{y}} \end{pmatrix} = \begin{pmatrix} S_{\text{uc}} \\ S_{\text{us}} \end{pmatrix}$$

We obtain

$$\begin{pmatrix} u_{\text{x}} \\ u_{\text{y}} \end{pmatrix} = \frac{1}{S_{\text{cc}}S_{\text{ss}} - S_{\text{cs}}^2}\begin{pmatrix} S_{\text{ss}} & -S_{\text{cs}} \\ -S_{\text{cs}} & S_{\text{cc}} \end{pmatrix}\begin{pmatrix} S_{\text{uc}} \\ S_{\text{us}} \end{pmatrix} \tag{20}$$

and the minimum error is

$$E_{\text{F}} = S_{\text{uu}} - \frac{1}{S_{\text{cc}}S_{\text{ss}} - S_{\text{cs}}^2}\left(S_{\text{uc}}^2 S_{\text{cc}} + S_{\text{us}}^2 S_{\text{ss}} - 2S_{\text{uc}}S_{\text{us}}S_{\text{cs}}\right) \tag{21}$$

Using the notation (17) we obtain by substituting into (9)

$$\Theta_0 = \sum_i W_i \left(u_{\text{n}_i} - u_{\text{rx}}\cos\psi_i - u_{\text{ry}}\sin\psi_i - \frac{1}{\hat{Z}}\cos\psi_i\right)^2 \tag{22}$$

It can be verified that u_{rx} only shifts the best $1/\hat{Z}$, but does not influence the final measure. Thus we can set u_{rx} to zero without loss of generality and expand Θ_0 to

$$\begin{aligned} \Theta_0 = {} & S_{\text{uu}} + u_{\text{ry}}^2 S_{\text{ss}} + \left(\frac{1}{\hat{Z}}\right)^2 S_{\text{cc}} - 2u_{\text{ry}}S_{\text{us}} \\ & - 2\frac{1}{\hat{Z}}S_{\text{uc}} + 2\frac{1}{\hat{Z}}u_{\text{ry}}S_{\text{cs}} \end{aligned}$$

Minimization of Θ_0 yields

$$\frac{1}{\hat{Z}} = \frac{S_{\text{uc}} - u_{\text{ry}}S_{\text{cs}}}{S_{\text{cc}}} \tag{23}$$

Let

$$u_{\text{ry}} = u_{\text{y}} + \delta u_{\text{ry}} \tag{24}$$

Measure Θ_1 is obtained by substituting (23) into Θ_0. Using (21) it can be written as

$$\Theta_1 = \frac{S_{\text{cc}}S_{\text{ss}} - S_{\text{cs}}^2}{S_{\text{cc}}}\,\delta u_{\text{ry}}^2 + E_{\text{F}} \tag{25}$$

As we show in the next section, if the optical flow in the patch were estimated based on minimization of (18), then δu_{ry} would represent the distance of the estimated flow from the epipolar line.

Thus the first component in (25) is related to the epipolar constraint and it depends on the 3D motion estimate, as well as on the gradient distribution in the patch. The second component in (25), E_{F}, represents how well the scene is approximated by a plane and it is independent of the 3D motion estimate. In classic approaches, after optical flow is computed, the term E_{F} is not considered any further and the estimation of 3D motion parameters is based only on the distance from the epipolar line. Here we keep this term and utilize it for segmentation.

In addition, in contrast with other approaches, in our measure the distance to the epipolar line appears in combination with a function describing the gradient distribution. Thus by studying the statistics of this function insight can be gained which might be exploited in algorithms for 3D motion estimation and segmentation. However, in this paper we have not attempted such an analysis; we only qualitatively describe this function and mention that it is closely related to the bias one would obtain when estimating flow on the basis of image derivatives (Fermüller et al., 2000).

Consider the expression

$$\begin{aligned} & \frac{S_{\text{cc}}S_{\text{ss}} - S_{\text{cs}}^2}{S_{\text{cc}}} \\ & \quad = \frac{\left(\sum_i \cos^2\psi_i\right)\left(\sum_i \sin^2\psi_i\right) - \left(\sum_i \cos\psi_i \sin\psi_i\right)^2}{\sum_i \cos^2\psi_i} \end{aligned} \tag{26}$$

Applying the Cauchy-Schwartz inequality to the numerator, we see that expression (26) is non-negative and can be zero only if the sequences $\sin\psi_1, \ldots, \sin\psi_n$ and $\cos\psi_1, \ldots, \cos\psi_n$ are proportional. Clearly, that can happen only if all the normal flow directions are parallel. Also, expression (26) cannot be arbitrarily large, even though we divide by S_{cc}. Since $S_{\text{cs}}^2 \geq 0$, we have

$$0 \leq \frac{S_{\text{cc}}S_{\text{ss}} - S_{\text{cs}}^2}{S_{\text{cc}}} \leq S_{\text{ss}} \leq N_{\mathcal{R}}$$

where $N_{\mathcal{R}}$ is the number of measurements in region $\mathcal{R}$.

In fact, expression (26) measures the range of normal flow directions within the region. If a region contains only a small range of directions, it may not provide reliable information for all candidate translations and (26) will be small for such a region. On the other hand, (26) will be large if the region contains a large range of measurement directions.

Compared to the epipolar constraint, the depth variability measure for smooth patches emphasizes regions with larger variation of normal flow directions and can thus be expected to yield better results for noisy data.

Finally, let us examine the behavior of Θ_1 for a smooth scene patch. For a fronto-parallel patch, ignoring noise, we get $E_{\mathrm{F}} = 0$ and the estimated $(u_{\mathrm{x}}, u_{\mathrm{y}})$ is equal to the true motion field vector. For any translation, we can make Θ_1 zero by choosing a rotation that yields $\delta u_{\mathrm{ry}} = 0$. But the rotation is not determined locally!

For the correct translation, we should estimate the correct rotation and obtain zero Θ_1 for all the smooth patches. Now consider an incorrect translation candidate. It is easy to find a rotation that makes Θ_1 zero for one or several smooth patches. But if we are able to find a rotation that yields zero Θ_1 for many different patches, this means we can obtain exactly the same motion field for two different 3D motions, and the scene (or large parts of it) has to be close to an ambiguous surface (Brodský et al., 1998a; Horn, 1987). Thus except for ambiguous surfaces we should obtain the correct 3D motion if most of the regions used correspond to smooth patches.

3.6. *The Epipolar Constraint*

The depth variability measure is closely related to the traditional epipolar constraint and we examine their relationship here. In the instantaneous form the epipolar constraint can be written as

$$(\hat{\mathbf{z}} \times \mathbf{u}_{\mathrm{tr}}(\hat{\mathbf{t}})) \cdot (\dot{\mathbf{r}} - \mathbf{u}_{\mathrm{rot}}(\hat{\boldsymbol{\omega}})) = 0 \qquad (27)$$

Usually, the distance of the flow vector $\dot{\mathbf{r}}$ from the epipolar line (determined by $\mathbf{u}_{\mathrm{tr}}(\hat{\mathbf{t}})$ and $\mathbf{u}_{\mathrm{rot}}(\hat{\boldsymbol{\omega}})$) is computed, and the sum of the squared distances, i.e.,

$$\sum ((\hat{\mathbf{z}} \times \mathbf{u}_{\mathrm{tr}}(\hat{\mathbf{t}})) \cdot (\dot{\mathbf{r}} - \mathbf{u}_{\mathrm{rot}}(\hat{\boldsymbol{\omega}})))^2 \qquad (28)$$

is minimized.

Methods based on (28) suffer from bias (Daniilidis and Spetsakis, 1997), however, and a scaled epipolar constraint has been used to give an unbiased solution:

$$\sum \frac{((\hat{\mathbf{z}} \times \mathbf{u}_{\mathrm{tr}}(\hat{\mathbf{t}})) \cdot (\dot{\mathbf{r}} - \mathbf{u}_{\mathrm{rot}}(\hat{\boldsymbol{\omega}})))^2}{\|\mathbf{u}_{\mathrm{tr}}(\hat{\mathbf{t}})\|^2} \qquad (29)$$

Again we use the coordinate system and notation of (17). Suppose that the flow vector $\dot{\mathbf{r}}$ has been obtained by minimization of (18); write it as $(u_{\mathrm{x}}, u_{\mathrm{y}})$. Substituting into (29) we obtain

$$\frac{((\hat{\mathbf{z}} \times \mathbf{u}_{\mathrm{tr}}(\hat{\mathbf{t}})) \cdot (\dot{\mathbf{r}} - \mathbf{u}_{\mathrm{rot}}(\hat{\boldsymbol{\omega}})))^2}{\|\mathbf{u}_{\mathrm{tr}}(\hat{\mathbf{t}})\|^2} = (u_{\mathrm{y}} - u_{\mathrm{ry}})^2 = \delta u_{\mathrm{ry}}^2 \qquad (30)$$

Equations (30) and (25) illustrate the relationship between the epipolar constraint (for the general case using non-standard weights to estimate flow) and the smoothness measure Θ_1.

To summarize, for the function Θ_1 that we minimized, we showed that at an image patch $\Theta_1 = C\,\delta u_{\mathrm{ry}}^2 + E_{\mathrm{F}}$, with E_{F} the error of the least squares fit to the image measurements of an optical flow corresponding to a linear scene depth, δu_{ry} the deviation from the epipolar constraint and C a factor that depends only on the positions and the directions of the normal flow measurements in the region. The larger the variation in the normal flow directions, the larger C is, thus giving more weight to regions where more information is available. In a smooth patch (i.e., an image patch corresponding to a smooth scene patch), $E_{\mathrm{F}} = 0$, making the minimization of Θ_1 equivalent to weighted epipolar minimization, although with weights different from the ones used in the literature; but for a non-smooth patch where the optical flow computation is unreliable, $E_{\mathrm{F}} \neq 0$ and this statistical information was used here to find depth boundaries.

4. Uncalibrated Camera and Self-Calibration

Throughout the paper we have assumed a calibrated imaging system, but the results and algorithms are also applicable when the camera calibration is not known. In this section we review the influence of the intrinsic camera parameters and discuss the modifications that have to be made to the algorithm to account for the additional unknowns.

4.1. *The Depth Variability Criterion for an Uncalibrated Camera*

In this section we consider a standard uncalibrated pinhole camera with internal calibration parameters described by the matrix

$$\mathbf{K} = \begin{pmatrix} f_x & s & \Delta_x \\ 0 & f_y & \Delta_y \\ 0 & 0 & F \end{pmatrix}$$

The coordinate system $OXYZ$ is attached to the camera, with Z being the optical axis, and F a constant we can choose and use as the third coordinate of the image point vectors $[x, y, F]^T$. Note that F is not the focal length of the camera; the real focal length and the aspect ratio are encoded in f_x, f_y.

A scene point $\mathbf{R}$ is projected onto the image point

$$\mathbf{r} = \frac{\mathbf{KR}}{\mathbf{R} \cdot \hat{\mathbf{z}}} \tag{31}$$

The camera motion is again (2). The image motion field is then (Brodský et al., 1998b)

$$\begin{aligned} \dot{\mathbf{r}} &= -\frac{1}{F(\mathbf{R} \cdot \hat{\mathbf{z}})}(\hat{\mathbf{z}} \times (\mathbf{Kt} \times \mathbf{r})) \\ &\quad + \frac{1}{F}(\hat{\mathbf{z}} \times (\mathbf{r} \times (\mathbf{K}[\boldsymbol{\omega}]_\times \mathbf{K}^{-1}\mathbf{r}))) \\ &= \frac{1}{\mathbf{Z}}\mathbf{u}_{\mathrm{tr}}(\mathbf{Kt}) + \mathbf{u}'_{\mathrm{rot}}(\mathbf{K}[\boldsymbol{\omega}]_\times \mathbf{K}^{-1}) \end{aligned} \tag{32}$$

where Z is used to denote the scene depth $(\mathbf{R} \cdot \hat{\mathbf{z}})$ and $[\boldsymbol{\omega}]_\times$ is the skew-symmetric matrix corresponding to the cross product with vector $\boldsymbol{\omega} = [\alpha, \beta, \gamma]^T$:

$$[\boldsymbol{\omega}]_\times = \begin{pmatrix} 0 & -\gamma & \beta \\ \gamma & 0 & -\alpha \\ -\beta & \alpha & 0 \end{pmatrix}$$

The translational component of the field is identical to a calibrated translational field with translation $\mathbf{Kt}$. The rotational component $\mathbf{u}'_{\mathrm{rot}}$ is slightly more complicated in the uncalibrated case (compare (32) and (3)).

The flow $\mathbf{u}'_{\mathrm{rot}}$ is determined by the matrix $\mathbf{A} = \mathbf{K}[\boldsymbol{\omega}]_\times \mathbf{K}^{-1}$ with seven degrees of freedom. As shown in (Brodský et al., 1998b), for a given translation $\tilde{\mathbf{t}}$, matrix $\mathbf{A}$ can be decomposed into

$$\mathbf{A} = \mathbf{A}_{\mathrm{c}} + \mathbf{A}_{\mathrm{t}} = \mathbf{A}_{\mathrm{c}} + F\,\tilde{\mathbf{t}}\mathbf{w}^T + w_0\mathbf{I} \tag{33}$$

Matrix $\mathbf{A}_{\mathrm{c}}$ (also called the copoint matrix) depends on five independent parameters and is the component of $\mathbf{A}$ that can be estimated (together with the direction of $\mathbf{Kt}$) from a single flow field. The vector $\mathbf{w}$ determines the plane at infinity and it cannot be obtained from a single flow field. Finally, w_0 can be computed from the condition trace $\mathbf{A} = 0$.

Also, due to linearity, we have

$$\begin{aligned} \mathbf{u}'_{\mathrm{rot}}(\mathbf{A}) &= \mathbf{u}'_{\mathrm{rot}}(\mathbf{A}_{\mathrm{c}}) + \mathbf{u}'_{\mathrm{rot}}(\tilde{\mathbf{t}}\mathbf{w}^T) + w_0\mathbf{u}'_{\mathrm{rot}}(\mathbf{I}) \\ &= \mathbf{u}'_{\mathrm{rot}}(\mathbf{A}_{\mathrm{c}}) + \frac{1}{F}(\mathbf{w} \cdot \mathbf{r})\mathbf{u}_{\mathrm{tr}}(\tilde{\mathbf{t}}) \end{aligned}$$

i.e., the rotational flow due to $\mathbf{A}_{\mathrm{t}}$ is equal to the translational flow of a certain scene plane.

In the sequel, $\hat{\mathbf{t}}$ is used to denote an estimate of the apparent translation $\mathbf{Kt}$ and $\hat{\mathbf{A}}_{\mathrm{c}}$ denotes an estimate of the copoint matrix $\mathbf{A}_{\mathrm{c}}$. We use $\mathbf{u}'_{\mathrm{rot}}(\hat{\mathbf{A}}_{\mathrm{c}})$ instead of $\mathbf{u}_{\mathrm{rot}}(\hat{\boldsymbol{\omega}})$ in (9) to obtain

$$\begin{aligned} &\Theta_0(\hat{\mathbf{t}}, \hat{\mathbf{A}}_{\mathrm{c}}, \mathcal{R}) \\ &= \sum_i W_i\left(\dot{\mathbf{r}}_{\mathbf{i}} \cdot \mathbf{n}_i - \mathbf{u}'_{\mathrm{rot}}(\hat{\mathbf{A}}_{\mathrm{c}}) \cdot \mathbf{n}_i - \left(\frac{1}{\hat{Z}}\right)(\mathbf{u}_{\mathrm{tr}}(\hat{\mathbf{t}}) \cdot \mathbf{n}_i)\right)^2 \end{aligned} \tag{34}$$

In the calibrated case, the rotational component of $\dot{\mathbf{r}}$ can be compensated for completely by the correct $\hat{\boldsymbol{\omega}}$. Here, the unknown parameters in $\hat{\mathbf{A}}$ introduce a component of the image normal flow, in the form of $(\mathbf{w} \cdot \mathbf{r})\mathbf{u}_{\mathrm{tr}}(\hat{\mathbf{t}})$, that cannot be expressed as part of $\mathbf{u}'_{\mathrm{rot}}(\hat{\mathbf{A}}_{\mathrm{c}})$. We therefore have to use a linear fit to the scene inverse depth $1/\hat{Z} = (\mathbf{z} \cdot \mathbf{r})$, i.e., define Θ_1 using the solution of (11). Then $\mathbf{z}$ incorporates the unknown rotational parameters $\mathbf{w}$ and the resulting depth variation criterion is independent of the rotation.

Minimization of Θ_0 with respect to $\mathbf{z}$ yields linear equations analogous to (11). Also, the functions Θ_1, Θ_2, and Φ can be defined exactly as in the calibrated case. All the computations can still be performed symbolically; instead of three components of $\hat{\boldsymbol{\omega}}$ we need to work with five components of $\hat{\mathbf{A}}_{\mathrm{c}}$.

The algorithm in Section 3.3 can be used with only a minor modification. In the patch segmentation part of the algorithm (see Section 3.4), we need to take the unknown linear component of depth into account. This term cannot be estimated, but to improve the edge detection process, we add a linear function to the inverse estimated depth so that the average depth in different parts of the image is approximately the same.

4.2. The Relationship to the Epipolar Constraint

We show that the relationship (25) can be generalized to the case of uncalibrated cameras. The analysis is almost identical; we only need to assume that instead of a constant function, a linear function is used as the inverse depth of an image region.

We again use the notation (17), the only change being $\mathbf{u}_{\mathrm{rot}}(\hat{\mathbf{A}}_{\mathrm{c}}) = [u_{\mathrm{rx}}, u_{\mathrm{ry}}, 0]^T$. We can then rewrite (9) as

$$\begin{aligned}\Theta_0 = \sum_i W_i \big(u_{\mathrm{n}_i} - (\mathbf{z} \cdot \mathbf{r_i}) \cos \psi_{\mathbf{i}} \\ - u_{\mathrm{rx}} \cos \psi_{\mathbf{i}} - u_{\mathrm{ry}} \sin \psi_{\mathbf{i}}\big)^2 \qquad (35)\end{aligned}$$

Note that u_{rx} can be incorporated into $\mathbf{z}$ (writing $\mathbf{z}' = \mathbf{z} + [0, 0, u_{\mathrm{rx}}/F]^T$) and we thus obtain the same minimum for the simplified expression:

$$\Theta_0 = \sum_i W_i \big(u_{\mathrm{n}_i} - (\mathbf{z} \cdot \mathbf{r_i}) \cos \psi_{\mathbf{i}} - u_{\mathrm{ry}} \sin \psi_{\mathbf{i}}\big)^2 \qquad (36)$$

Now consider the least squares estimation of optical flow in the region using the weights W_i. Allowing linear depth changes, in the local coordinate system we fit flow $(\mathbf{u}_{\mathrm{x}} \cdot \mathbf{r}, u_{\mathrm{y}})$, i.e., a linear function along the direction of $\mathbf{u}_{\mathrm{tr}}(\hat{\mathbf{t}})$ and a constant in the perpendicular direction. We minimize

$$\sum_i W_i \big(u_{\mathrm{n}_i} - (\mathbf{u}_{\mathrm{x}} \cdot \mathbf{r_i}) \cos \psi_{\mathbf{i}} - u_{\mathrm{y}} \sin \psi_{\mathbf{i}}\big)^2 \qquad (37)$$

Expressions (36) and (37) are almost identical, but there is one important difference. The optical flow minimization (37) is strictly local, using only measurements from the region. On the other hand, in (36), the rotational flow $(u_{\mathrm{rx}}, u_{\mathrm{ry}})$ is determined by the global motion parameters.

Let us denote the least squares solution of (37) by $(\hat{\mathbf{u}}_{\mathrm{x}}, \hat{u}_{\mathrm{y}})$ and the residual by E_{F}. After some vector and matrix manipulation we obtain

$$\Theta_1 = \big(m_{\mathrm{ss}} - \mathbf{m}_{\mathrm{cs}}^T \mathbf{M}_{\mathrm{cc}}^{-1} \mathbf{m}_{\mathrm{cs}}\big)\, \delta u_{\mathrm{ry}}^2 + E_{\mathrm{F}} = C\, \delta u_{\mathrm{ry}}^2 + E_{\mathrm{F}} \qquad (38)$$

where

$$m_{\mathrm{ss}} = \sum_i W_i \sin^2 \psi_i, \quad \mathbf{m}_{\mathrm{cs}} = \sum_i W_i \cos \psi_i \sin \psi_i \mathbf{r_i},$$

$$\mathbf{M}_{\mathrm{cc}} = \sum_{\mathbf{i}} \mathbf{W_i} \cos^2 \psi_{\mathbf{i}} \mathbf{r_i} \mathbf{r_i}^T$$

and $\delta u_{\mathrm{ry}} = u_{\mathrm{ry}} - \hat{u}_{\mathrm{y}}$ is the difference between the globally determined rotational component u_{ry} and the best local optical flow component $\hat{u}_{\mathrm{y}}$. Both of the components are in the direction perpendicular to the translational flow and δu_{ry} is therefore the epipolar distance.

Thus we have a relationship analogous to (25). The depth variation measure is the sum of a component E_{F} that evaluates whether the depth in the region is smooth, and a scaled epipolar distance $C\, \delta u_{\mathrm{ry}}^2$.

Again, factor C in (38) depends only on the geometric configuration of the measurements within the region. The complete analysis is complicated by the fact that C also depends on the point positions. One simple observation is the boundedness of C: matrix $\mathbf{M}_{\mathrm{cc}}$ is positive definite, so $0 \le C \le m_{\mathrm{ss}} \le \sum_i W_i$.

To see that (38) is indeed a direct generalization of (25), assume that all the measurements are taken at a single point. Then we can only solve for one component of the vector $\mathbf{z}$ and consequently $\mathbf{M}_{\mathrm{cc}}$ simplifies into S_{cc} and $\mathbf{m}_{\mathrm{cs}}$ becomes S_{cs}.

4.3. Self-Calibration

So far we have shown how to estimate the apparent translation $\hat{\mathbf{t}}$ and the copoint matrix $\hat{\mathbf{A}}_{\mathrm{c}}$. To perform camera self-calibration, and thus subsequently derive structure, we can use the method developed in (Brodský et al., 1998b), combining the partial information assuming that the internal camera parameters are constant throughout the image sequence.

According to (32), the rotational component of the motion field is determined by matrix $\mathbf{A} = \mathbf{K}[\boldsymbol{\omega}]_\times \mathbf{K}^{-1}$. Matrix $[\boldsymbol{\omega}]_\times$ is skew-symmetric, i.e., $[\boldsymbol{\omega}]_\times + [\boldsymbol{\omega}]_\times^T = 0$. This is the constraint that we use, expressed in terms of $\mathbf{K}$ and $\mathbf{A}$:

$$\mathbf{K}^{-1}\mathbf{A}\mathbf{K} + (\mathbf{K}^{-1}\mathbf{A}\mathbf{K})^T = 0 \qquad (39)$$

Suppose we have a set of copoint matrices $\hat{\mathbf{A}}_{\mathrm{c}i}$ and based on (33) we write

$$\hat{\mathbf{A}}_i = \hat{\mathbf{A}}_{\mathrm{c}i} + F\, \hat{\mathbf{t}}_i \mathbf{w}_i^T + w_{0i} \mathbf{I}$$

We solve for w_{0i} using trace $\mathbf{A} = 0$ and obtain

$$\hat{\mathbf{A}}_i = \hat{\mathbf{A}}_{\mathrm{c}i} + F\, \hat{\mathbf{t}}_i \mathbf{w}_i^T - \frac{1}{3} \text{ trace } \big(\hat{\mathbf{A}}_{\mathrm{c}i} + F\, \hat{\mathbf{t}}_i \mathbf{w}_i^T\big)\mathbf{I}$$

Also vectors $\mathbf{w}_i$ are unknown and appear only to the first order in $\hat{\mathbf{A}}_i$. Thus we can minimize

$$\|\mathbf{K}^{-1}(\hat{\mathbf{A}}_i)\mathbf{K} + (\mathbf{K}^{-1}(\hat{\mathbf{A}}_i)\mathbf{K})^T\|^2$$

with respect to $\mathbf{w}_i$ (solving a linear system) and after substitution obtain an error measure with $\mathbf{K}$ as the only unknown. The final error function is the sum of partial errors:

$$\mathcal{E}(\mathbf{K}) = \sum_i \|\mathbf{K}^{-1}(\hat{\mathbf{A}}_i)\mathbf{K} + (\mathbf{K}^{-1}(\hat{\mathbf{A}}_i)\mathbf{K})^T\|^2 \quad (40)$$

Levenberg-Marquardt minimization is used to minimize $\mathcal{E}(\mathbf{K})$ and to obtain the calibration parameters. For details, the reader is referred to (Brodský et al., 1998b).

5. Construction of Scene Models from Multiple Frames

Given a video sequence, one can obtain a set of instantaneous camera motion estimates as well as scene structure estimates. Considering a single flow field, approximate information about the shape of the scene can be recovered, allowing us, for example, to distinguish between close and far scene patches. To obtain accurate models of the scene, it is necessary to combine information from many flow fields so that sufficiently different scene views can be related and utilized. Essentially, this is a form of the multi-camera stereo problem. We have many views of a static scene and we would like to recover the scene structure.

In this particular situation, however, there are some important differences compared to traditional stereo. First, we have a dense sequence and image features move very little between successive frames, requiring sub-pixel accuracy for the disparity estimates. This could be alleviated by skipping frames and thus considering views that are further apart. Second, in traditional stereo algorithms the camera positions are assumed to be known. Here, the camera positions are obtained by integrating the instantaneous velocity estimates, leading to unavoidable errors. Fortunately, it turns out that a lot of redundant information is available and it is possible to not only estimate the scene structure, but also to correct the camera position estimates to obtain a more consistent solution.

The general approach taken here is not concerned with perception, i.e., we are not studying representations that could be extracted in real time by a moving human observer. In fact, it is known and can be verified experimentally (Foley, 1980; Tittle et al., 1995) that human observers do not obtain Euclidean models of their surroundings. Rather, the motivation comes from the need to construct models for use in Computer Graphics or Virtual Reality. As a computational problem, Euclidean model construction is feasible, if we allow batch processing techniques and provide sufficient computing power.

The image measurements we make are based on the brightness constancy assumption—the brightnesses of the projections of the scene points do not change over time. If we extend the brightness constancy assumption to many frames, we obtain a simple idea that has been utilized in several different forms (Faugeras and Keriven, 1998; Seitz and Dyer, 1997; Szeliski and Golland, 1998). Considering a point in the scene and its projections onto all the camera image planes (for the cameras that see that particular point), all the brightness values should be the same. This provides a constraint linking the camera positions and the scene point positions.

More reliable results can be obtained by considering scene surface patches instead of individual scene points. The brightness constancy constraint is then closely related to the usual cross-correlation approaches used in stereo algorithms. Utilizing brightness constancy (for points and/or patches), we can view the task of model construction as a numerical minimization problem over many variables—the camera motions and the 3D scene structure. Clearly, the computational requirements would be extremely high for an “everything-at-once” approach that would try to utilize all the available information. The problem is further complicated by issues such as occlusion and changing lighting conditions for widely disparate camera positions. In addition, integration of the motion parameters over many frames may lead to significant camera position errors.

More feasible is a hierarchical approach, where short sub-sequences are considered first and the partial results are later combined to construct a complete scene model. This section studies the problem of “linking” a short sequence of frames, usually between 20 and 40 frames long. We construct a model of the part of the scene visible in the sub-sequence and we also correct the estimated camera motions.

We need to choose enough frames so that sufficiently different views of the scene are available, thus making the problem more stable. On the other hand, not too many frames should be used due to computational requirements. In addition, for a short subsequence we do not have to worry about occlusion and significant illumination changes. For example, a scene model can be constructed that only represents scene surfaces visible from the middle frame of the sub-sequence.

The method presented in Section 5.4 can be thought of as a differential equivalent of the often-used bundle adjustment (Hartley, 1994), a numerical optimization that minimizes the distances between the estimated feature points and the re-projections of the reconstructed points. In the current literature (Beardsley et al., 1996; Fitzgibbon and Zisserman, 1998; Pollefeys et al., 1998) dealing with motion and scene estimation based on point correspondences and the discrete motion model, most algorithms use bundle adjustment either as the final step, or intertwined with other steps of the algorithm.

5.1. *Coordinate Systems*

We assume that the scene is static and that the intrinsic camera parameters are known, either beforehand, or from self-calibration. The motion estimation method provides the instantaneous camera motion for each successive pair of frames, plus a partial scene model (in the form of depth estimates at some pixels) obtained from normal flow measurements.

During motion estimation, we use a coordinate system attached to the camera, and both the camera motion and the recovered scene information are computed with respect to this coordinate frame. In order to build the scene model, we need to represent all the information in a common coordinate system. Initially, we compute all the camera positions with respect to the first camera. Once all the relative camera positions are known, we can easily transform the data between coordinate systems. To build the model of the scene, we use the coordinate system (called the model coordinate frame below) of the middle camera position. We choose the middle frame rather than the first frame, because the accumulated errors between the middle frame and the first and last frames are smaller than the accumulated errors between the first and last frames.

Any camera position in the sequence can be related to the starting camera position by a (discrete) rigid transformation. In this section, motion composed of a rotation $\mathbf{R}$ and a translation $\mathbf{T}$ transforms scene point $\mathbf{M}$ into point $\mathbf{RM} + \mathbf{T}$. It is well known that for the case of a moving camera with calibration matrix $\mathbf{K}$, the 3×4 projection matrix can be written as

$$\mathbf{P} = \mathbf{K}\mathbf{R}^T(\,\mathbf{I} \mid -\mathbf{T}) \tag{41}$$

where $\mathbf{I}$ is a unit 3×3 matrix, $\mathbf{R}$ is the rotation matrix and $\mathbf{T}$ is the translation vector that transforms the model frame to the camera coordinate frame.

We denote the successive camera positions corresponding to the input images by $\mathbf{P}_i = \mathbf{K}\mathbf{R}_i^T(\,\mathbf{I} \mid -\mathbf{T}_i)$, where $\mathbf{R}_0 = \mathbf{I}$ and $\mathbf{T}_0 = \mathbf{0}$, i.e., the first camera position is at the origin of the model frame.

How are the projection matrices related to the instantaneous motion estimates? Consider a camera displaced by $(\mathbf{R}_i, \mathbf{T}_i)$ and the velocity estimate $(\mathbf{t}_{i+1}, \boldsymbol{\omega}_{i+1})$ for that camera.

The rotational velocity $\boldsymbol{\omega}_{i+1}$ represents the rotation matrix $e^{[\boldsymbol{\omega}_{i+1}]_\times} = \mathbf{I} + [\boldsymbol{\omega}_{i+1}]_\times + \frac{1}{2}[\boldsymbol{\omega}_{i+1}]_\times^2 + \cdots \approx \mathbf{I} + [\boldsymbol{\omega}_{i+1}]_\times$. Since our method works with dense video sequences, the rotationalvelocity is small and we can omit higher-order terms in the expansion. Then

$$\mathbf{R}_{i+1} = (\mathbf{I} + [\boldsymbol{\omega}_{i+1}]_\times)\mathbf{R}_i \tag{42}$$

The translational velocity $\mathbf{t}_{i+1}$ is measured in the coordinate system rotated by matrix $\mathbf{R}_i$ and can be recovered only up to scale. Therefore

$$\mathbf{T}_{i+1} = \mathbf{T}_i + s_i\mathbf{R}_i\mathbf{t}_{i+1} \tag{43}$$

for some unknown scale factor s_i. The scale factor can, as discussed in Section 5.2, be recovered by assuming that all the cameras view the same scene.

5.2. *Finding the Translation Scaling*

The camera translational velocity can be recovered only up to scale from two frames, due to the inherent ambiguity between the size of the translation and the scale of the scene. That is, when the translation is multiplied by a constant, all the depth estimates are multiplied by the same constant and the image measurements remain unchanged. Two successive flow fields in a dense video sequence correspond to two very similar views of the scene. Thus, from the depth estimates of both flow fields, the relative size of the two translations can be computed, leaving the scale as the only unrecoverable variable.

To improve robustness, we do not compare only successive flow fields, but incrementally build an image-based depth map of the scene, i.e., an estimate of scene depth at image pixels where measurements are available. A motion estimate $(\hat{\mathbf{t}}, \hat{\omega})$, together with a scene depth estimate $\hat{Z}$ at an image pixel, allow us to estimate the full flow at that pixel as well as the change in depth between successive frames. The depth map can thus be transferred to provide a prediction of the depth for the next image frame.

The following algorithm uses as input the instantaneous camera motions between successive frames, that is, the rotation and the direction of translation for each pair of successive frames, and it recovers the relative sizes of the translation vectors. A very accurate solution is not required; we only need a reasonable initial solution that can be corrected by the iterative adjustment algorithm. Because a single number is estimated from many image measurements at each step of the algorithm, its performance is very good.

```
1. Fix the overall scale by setting
   $\|\mathbf{t}_1\| = 1$.
2. Compute instantaneous depth for motion
   $(\mathbf{t}_1, \boldsymbol{\omega}_1)$ and use it to initialize the
   depth map.
3. Set $i = 1$.
4. Transfer the depth map using the
   motion $(\mathbf{t}_i, \boldsymbol{\omega}_i)$ to obtain a predicted
   depth map.
5. Increment $i$.
6. Compute instantaneous depth for motion
   $(\mathbf{t}_i, \boldsymbol{\omega}_i)$.
7. Each pixel where both the predicted
   and the current depth estimates are
   available provides one linear equation
   for the translation scaling factor.
   Solve all the equations by least
   squares (throwing out outliers).
8. Update the depth map using
   $Z_{\text{new}} = \alpha\, Z_{\text{predicted}} + (1-\alpha)\, Z_{\text{curr}}$ with $\alpha < 1$.
9. Go to step 4, unless all the frames
   have been processed.
```

Once all the translation scaling factors are computed we have the camera projection matrices $\mathbf{P}_i$ and the problem can be treated as a multi-camera stereo problem. To improve the solution and to correct accumulated errors in the camera positions, it is possible to first run an iterative improvement stage.

5.3. Multi-Camera Brightness Constancy

In this section we consider the brightness constancy assumption generalized to many frames in a sequence. We first consider a single scene point $\mathbf{M}$ projected onto image point $\mathbf{m}$ by camera $\mathbf{P}_j$. Denote the brightness of point $\mathbf{m}$ by E_j. The 3D position of point $\mathbf{M}$ can be expressed as a function of $\mathbf{m}$, the camera $\mathbf{P}_j$ and the depth Z as

$$\mathbf{M} = \mathbf{R}_j \mathbf{K}^{-1} Z\, \mathbf{m} + \mathbf{T}_j.$$

If we project $\mathbf{M}$ onto the other cameras $\mathbf{P}_k$, we obtain points

$$\mathbf{m}_k = \frac{F\, \mathbf{K}\mathbf{R}_k^T (\mathbf{M} - \mathbf{T}_k)}{\big((\mathbf{K}\mathbf{R}_k^T (\mathbf{M} - \mathbf{T}_k)) \cdot \hat{\mathbf{z}}\big)}.$$

Denoting the brightness of $\mathbf{m}_k$ by E_k, we can express the brightness constancy error as

$$E = \sum_k (E_j - E_k)^2 \tag{44}$$

Of course, we do not have to limit ourselves to single points; we can instead estimate the deviation patch-wise. In the simplest case, the error for a patch is the sum of the point-wise errors (44) for all the points in it.

When minimizing the brightness constancy error to find the depths of scene points, the values of E_j and E_k in (44) should at least be close even if there are some errors in the estimated camera geometry. However, due to occlusions, the tested point may not be visible in all image frames and consequently the error function may become large even for the correct scene point. Instead of straightforwardly computing the sum of squares, we also choose a threshold T for the maximum brightness difference and larger brightness differences in the sum (44) are replaced by the square of the threshold:

$$E = \sum_k \min\left((E_j - E_k)^2, T^2\right) \tag{45}$$

In this error function the influence of occluded points is reduced. This is of primary interest when the initial depth map is being constructed. Once the scene model is built, the occlusions can be explicitly taken into account.

5.4. Iterative Adjustment

The error (44) is a function of the scene point $\mathbf{M}$ and the cameras $\mathbf{P}_i$, or, alternatively, a function of the instantaneous velocity parameters $\mathbf{t}_i$, $\boldsymbol{\omega}_i$ and the scene depth Z of the point.

If the camera positions were known exactly, we could find the depth of point **M** by minimizing (44) with respect to Z. However, we can think of (44) as a function of many parameters, namely, all the translation and rotation parameters, and one depth parameter. Then we can also adjust the recovered motion parameters to improve the solution.

This is the basis of the method presented here. To lower the computational requirements, only a small number of scene points (we used 300) are considered together with all the camera positions. The scene points are expressed by image coordinates in the middle frame of the sequence plus a scene depth with respect to that frame. Therefore each point brings in one unknown (the scene depth). We choose points where the brightness gradient is high, since such points provide the most reliable information, and we also try to distribute the chosen points uniformly over the input image.

First-order derivatives of the error function can be computed analytically. The input images are smoothed by a Gaussian filter and we use bilinear interpolation to obtain the brightness values at non-integer image positions. Levenberg-Marquardt minimization can be used to minimize the error function iteratively.

5.5. *Obtaining a Dense Scene Model*

Once the camera positions have been corrected, one can use many different algorithms to obtain the scene model, including multi-frame stereo (Cox et al., 1996; Koch et al., 1998; Okutomi and Kanade, 1993; Robert and Deriche, 1996) and the level set approach (Faugeras and Keriven, 1998). As explained above, we construct a model with respect to the middle frame of the sequence. What we are interested in is estimating, for every point in the middle image, the distance of the corresponding scene point from the middle camera.

Conceptually the simplest solution would consider each pixel of the middle frame separately and try to find the best depth for that pixel, yielding a cloud of scene points. Such a solution tends to be quite close to the actual scene shape overall, but individual points are not very reliable; many errors can occur due to varying object textures combined with small camera position errors, for example.

For most scenes, rather than computing a cloud of scene points, we would like to obtain the object surfaces, for instance approximated by a triangular mesh. Thus we could choose a set of triangles and perform the stereo search not for single points, but for the triangular patches. The texture in a triangle certainly contains more information than the brightness of a single point and the recovered depth is thus much more reliable. However, this approach would only work if the triangles did not cross depth boundaries; otherwise, erroneous results would be obtained.

Depth boundaries can be estimated together with the 3D motion; see for example Fig. 4. However, for the purpose of motion estimation, we only had to find boundaries that separated measurements on both sides of the discontinuity. In many cases, any boundary within a band a few pixels wide would be sufficient. In addition, those boundaries are based on instantaneous flow, are not as reliable as necessary, and some depth boundaries may not be detected at all because for that particular instantaneous motion, the depth at those points is difficult to estimate. These problems, while not crucial for motion estimation, manifest themselves once we decide to create an accurate model of the scene.

In the sequel we describe our approach to model construction and provide illustrations of the various steps in the approach. All the intermediate results are shown for the "flower scene" sequence (see Fig. 8). We use a combination of point-based and triangle-based approaches, and while the models could certainly be improved by employing more sophisticated algorithms, the current results are very promising.

5.5.1. Initial Depth Map. For the purpose of model construction, the depth boundaries cannot be recovered reliably enough from instantaneous flow fields. Since we do not want to make smoothness assumptions (the areas where these assumptions are violated tend to be clearly visible in the resulting model), we start the process by considering each pixel of the middle frame separately.

Each pixel defines a ray in the scene and we search for the scene depth (determining the position of the scene point along the ray) that minimizes the thresholded brightness constancy error (45). To find the depth

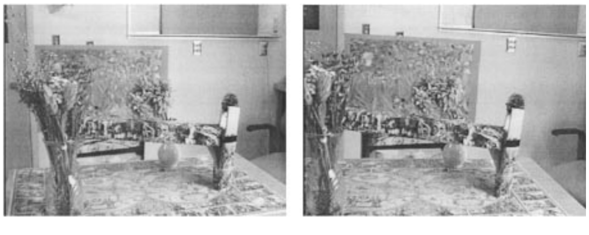

Figure 8. Two input frames from the "flower scene" sequence.

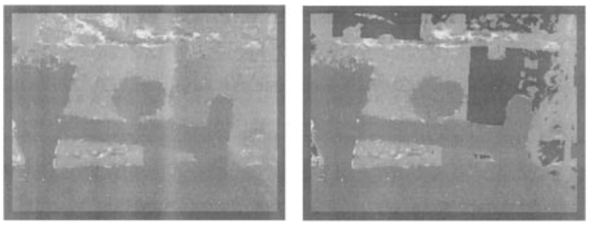

Figure 9. (a) An initial depth map obtained for the "flower scene" sequence. The gray-level value represents the scene depth, with white denoting large values (distant points) and mid-gray representing close points in front of the camera. Black would represent points behind the camera. (b) The same depth map thresholded to remove points with low variation in texture.

interval that needs to be searched for the given scene, we utilize the approximate sparse scene model obtained in Section 5.2, where we computed the relative sizes of the translation velocities. While this model itself is not very good, we can certainly extract from it the range of depth values occurring in the scene.

The initial depth map for the "flower scene" sequence is shown in Fig. 9(a). Notice that the shape of the scene is recovered quite well, except in regions of very low texture (white walls, for example). As the next step, we therefore evaluate the variation of image brightness in a small neighborhood around each pixel. Scene points corresponding to uniform intensity regions are then discarded from the model, yielding a pruned depth map in Fig. 9(b). To recover the shape of a white wall with no texture, we can obtain the depths on the boundaries and then, if a better model is required, interpolate the depth measurements within uniform regions. This step is not performed here.

5.5.2. Depth Boundaries and the Triangle Model. Since we are interested in approximating the recovered scene model by a triangular mesh, we need to find the depth boundaries first. We apply a median filter to the pruned depth map, obtaining Fig. 10(a) and then detect edges with the Canny edge detector modified to handle missing data. A description of the edge detector can be found in Section 3.4. The resulting boundaries shown in Fig. 10(b) are then utilized to correctly split the triangles.

The triangular mesh is also computed with respect to the middle frame. We cover the image with a regular triangular mesh as illustrated in Fig. 11. The triangle vertices can be represented by two image coordinates and one depth parameter. The depth of each vertex is set to the average of the depths of the points in a small neighborhood of the vertex.

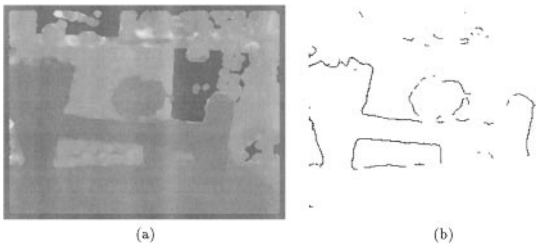

Figure 10. (a) Median filtered depth map. (b) Edges computed by a modified Canny edge detector.

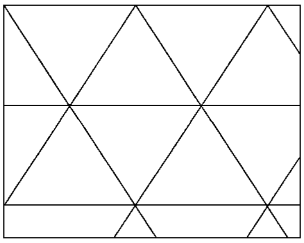

Figure 11. The initial triangular mesh covering the middle frame.

The conversion from a depth map to a triangular mesh yields a single surface that does not respect the depth boundaries. We therefore allow each triangle to include a mask denoting the pixels that are valid in that triangle. For each triangle that is divided by depth boundaries found in the previous section, we make two copies of the triangle, "cloning" the vertices and setting the masks on the two copies to represent the two sides of the depth boundary. Each new, cloned vertex is assigned a depth compatible with the triangle it is associated with.

The splitting is illustrated in Fig. 12. The original (scene) triangle ABC crossing an edge boundary in Fig. 12(a) gets divided into two new triangles ABC′ and

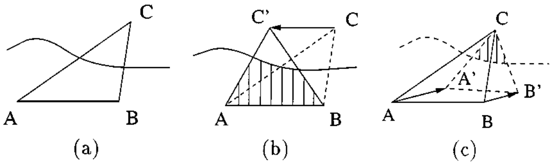

Figure 12. (a) The initial (scene) triangle ABC crosses an edge boundary. (b) Vertex C is "cloned" and the new triangle ABC′ is assigned a mask (shown as the shaded area). Only the shaded points are valid in the new triangle. (c) Vertices A and B are "cloned" and the new triangle A′B′C is assigned a mask.

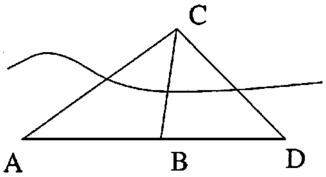

Figure 13. When triangles ABC and BDC are split, the cloned vertex C′ is shared by triangles ABC′ and BDC′ and the cloned vertex B′ is shared by triangles A′B′C and B′D′C.

A′B′C. Both new triangles include a mask that denotes the valid points in the triangle.

With a little care, when a depth boundary splits a band of neighboring triangles, the triangle splitting can be done so that the surfaces on each side of the boundary are still connected. For example, in Fig. 13, the adjacent triangle BDC with vertices B, D on the same side of the boundary would be split into BDC′ (i.e., vertex C′ would be shared by triangles ABC′ and BDC′) and another triangle B′D′C (i.e., vertex B′ would be shared by triangles A′B′C and B′D′C).

The depth map obtained from the triangular mesh, as shown in Fig. 14(a), nicely separates objects at different distances from the camera. Some parts of the depth map that are erroneous (due to lighting changes, for example) usually correspond to sudden variations in scene depth. If the model is to be combined with other models obtained from different subsequences, we can further improve it by removing the most slanted triangles, yielding the depth map in Fig. 14(b). By removing the slanted triangles, we remove model parts that can probably be recovered more reliably from another viewpoint. This can be advantageous when multiple short subsequences are combined to create a more complete scene model.

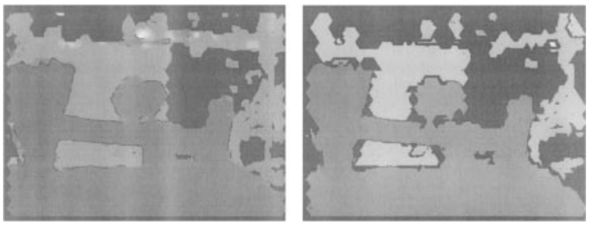

Figure 14. (a) The scene depth obtained from the triangle model. (b) The scene depth after the removal of excessively slanted triangles. Because the range of the depth values changed after the removal, the scaling of this depth map for display purposes was different from the scaling of the previous depth map.

6. Experimental Results

We present experiments testing different aspects of our method. In particular, the ability of our technique to extract depth edges is demonstrated and its performance in 3D motion estimation is shown. To allow the reader to judge the accuracy of the results, a comparison between the depth variability measure and the epipolar constraint is given. The results on the estimation of the calibration parameters are shown, and 3D model reconstruction for a number of sequences is performed.

To demonstrate the estimation of the motion and calibration parameters two sequences are used: the lab sequence, as shown in Fig. 1(a), and the Yosemite fly-through sequence (one frame is shown in Fig. 15). Since the images of the latter sequence also show independently moving clouds, the image frames were clipped to contain only the mountain range. Three more sequences are used for 3D model reconstruction, the “flower scene” (Fig. 8), the “headshot” sequence (Fig. 19), and the “pooh” sequence (Fig. 26).

The normal flow fields computed throughout and the optical flow fields needed in the comparison were derived using the method of Lucas and Kanade (1984), as implemented in Barron et al. (1994) with a temporal support of fifteen frames for one normal flow field. In more detail the following steps are performed in the estimation of normal flow: (1) The image sequence is filtered with a spatiotemporal Gaussian filter, typically with standard deviation sigma $= 1.5$ and kernel size $11 \times 11 \times 11$ pixels. (2) The spatial and temporal

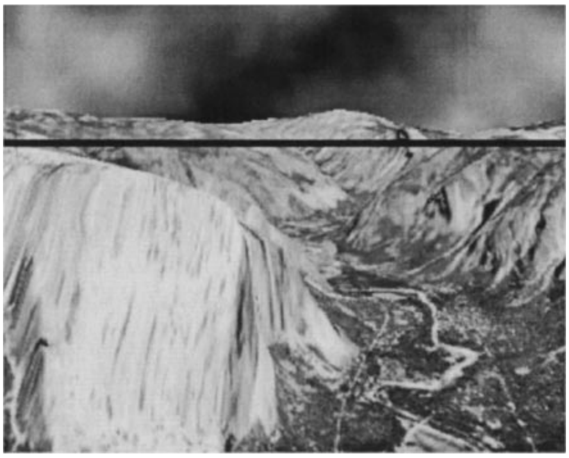

Figure 15. One frame of the Yosemite sequence. Only the bottom part of the image was used.

Table 1. Estimated epipole locations for the Yosemite sequence.

Method	Epipole
Ground truth	$(0.0, -100.0)$
Epipolar minimization	$(0.5, -98.8)$
$\Phi(\hat{\mathbf{t}})$ (No segmentation)	$(2.4, -96.7)$
$\Phi(\hat{\mathbf{t}})$ (With segmentation)	$(0.0, -103.6)$

derivatives are estimated with a 5 point symmetric kernel $1/12 \times (-1, 8, 0, 8, -1)$ applied to the blurred images. (3) Normal flow values are then computed directly from the derivatives, but only at points with high enough brightness gradients.

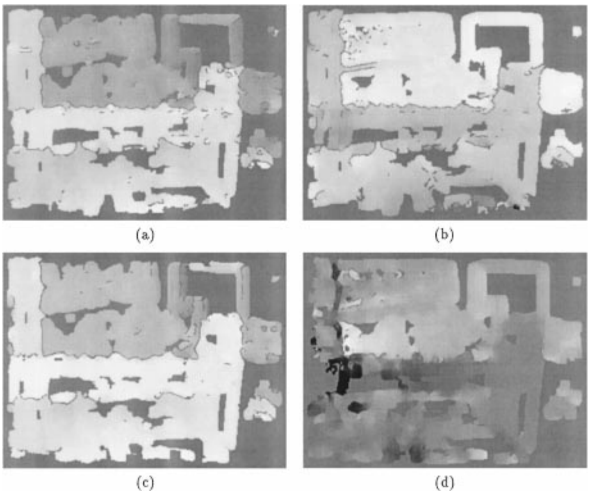

Figure 16. Comparison of recovered inverse depth for the lab sequence using the epipole positions as estimated with the proposed algorithm and with epipolar minimization. (a), (b) Frame 134. (a) Depth variation, epipole: $(377, -125)$; (b) Epipolar minimization, epipole: $(-612, 256)$. (c), (d) Frame 142. (a) Depth variation, epipole: $(483, -123)$; (b) Epipolar minimization, epipole: $(-153, 18)$.

Experiment 1. In this experiment we evaluated the accuracy of 3D motion estimation. In particular, we compared our method using minimization based on function $\Phi(\hat{\mathbf{t}})$ both with and without segmentation and minimization based on the epipolar constraint.

For the Yosemite sequence, both our method and the epipolar minimization perform quite well. The known epipole location in the image plane was $(0, -100)$. The estimated epipole locations for the different methods are summarized in Table 1.

Experiment 2. The lab sequence contains several significant depth discontinuities. For the majority of frames our method performed better than epipolar minimization. No ground truth was available, but we visually inspected the instantaneous scene depth recovered. Out of a ninety-frame subsequence analyzed with both methods, epipolar minimization yielded 25 frames with clearly incorrect depths (i.e., many negative depth estimates, or reversed depth order in large parts of the scene). The performance of our method was significantly better; only seven frames yielded clearly incorrect depths.

Some failures of epipolar minimization are shown in Fig. 16. For some frames, the recovered depth was reversed, i.e., the background was closer than the foreground, as in Fig. 16(b). In other frames, some parts of the scene had negative recovered depth, as in Fig. 16(d), where the black regions correspond to negative depth.

To further demonstrate the superior performance of our algorithm, we plotted the recovered epipole positions in the image as they evolved over time. The results of the proposed algorithm are shown in Fig. 17(a). The computation is stable and the epipole position changes slowly for most frames. The exception is a short subsequence that corresponds to a sudden change in camera motion. For some of the problematic frames the camera motion is predominantly rotational, making scene depth estimation difficult. In comparison, the epipolar minimization results in Fig. 17(b) are clearly less stable and many erroneous solutions are found.

Experiment 3. The lab sequence was taken by a hand-held Panasonic D5000 camera with a zoom setting of approximately 12 mm. Unfortunately, the effective focal length of the pinhole camera model was also influenced by the focus setting and we thus knew the

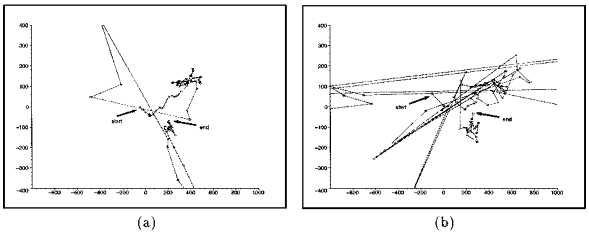

Figure 17. The evolution of recovered epipole positions over ninety frames. (a) The proposed algorithm. (b) Epipolar minimization. Both graphs show an identical part of the image plane (x between -1000 and 1000, y between -400 and 400). The image size was 320×240 pixels.

Table 2. Self-calibration results for the lab sequence.

Frames	f_x	f_y	Δ_x	Δ_y	s
001–300	536	522	16	26	3
001–100	541	543	-33	6	-25
101–200	544	475	26	-38	14
201–300	548	513	-11	8	6

intrinsic parameters only approximately. The internal parameters were fixed and were approximately $f_x = f_y = 450$, $\Delta_x = \Delta_y = s = 0$. The focal lengths were slightly overestimated, but consistent for different frames of the sequence. The calibration results are summarized in Table 2.

Experiment 4. The self-calibration results can be used to build Euclidean models of the scene, but further research is needed to link the individual frames and reliably combine the partial depth estimates in order to create a volumetric model of the scene. Here we present a reconstruction that shows the depth values obtained from a small number of frames.

No effort was made to find the exact depth boundary positions; the depth boundaries are the ones found during the motion estimation process. Two views of the 3D reconstruction are shown in Fig. 18.

Experiment 5. Complete 3D model construction was performed on three color video sequences acquired in our lab: the "flower scene" sequence (see Fig. 8), a video of a person's head, referred to as the "headshot" sequence (see Fig. 19), and a sequence of a child's toy ("pooh" sequence, Fig. 26). In all the sequences the intrinsic settings of the digital video camera were unknown. First, the viewing geometry was computed with the motion estimation and self-calibration algorithms. Examples of the recovered instantaneous inverse depth maps for the first two sequences are shown in Fig. 20. The multi-camera stereo algorithm described in Section 5 was then applied. The scene depth is estimated with respect to the middle frame of each sequence and the middle view also provides texture for the 3D model. The recovered depth in Fig. 21 shows the final depth values of the triangle patches in the model. Two views of the recovered 3D model with and without mapped texture are shown in Figs. 22 and 23 for the "flower scene" sequence and in Figs. 24 and 25 for the "headshot" sequence. Reconstructions for the "pooh" sequence are shown in Fig. 26.

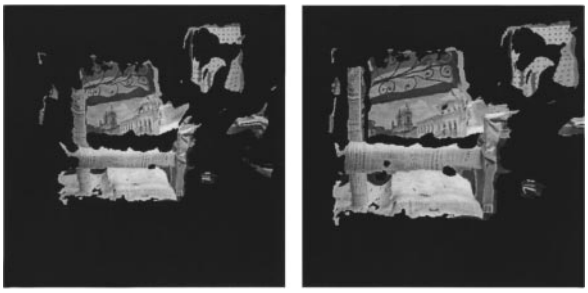

Figure 18. Two views of a 3D reconstruction of the recovered depth combined from fifteen image frames.

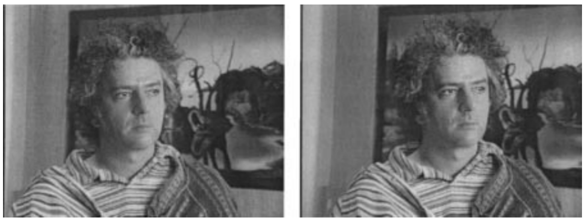

Figure 19. Two input frames from the "headshot" sequence.

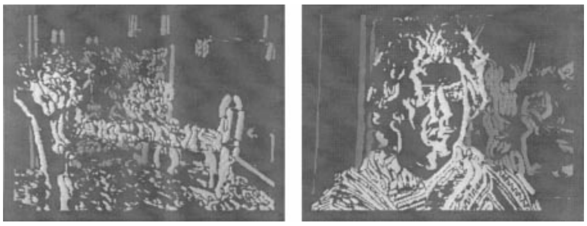

Figure 20. Examples of the recovered instantaneous inverse depths.

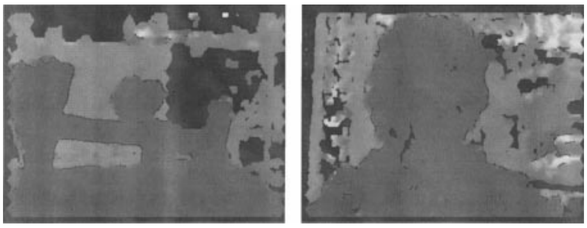

Figure 21. Scene depth recovered by multi-camera stereo.

A further reconstruction was computed for the Yosemite sequence and compared to the ground truth data. Since this sequence is only fifteen frames long, normal flow was computed with a modified algorithm, using only two frames to estimate the temporal derivatives. We computed the global scaling factor

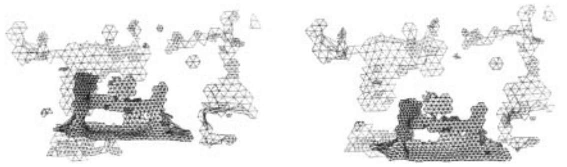

Figure 22. Two views of mesh model of the "flower scene."

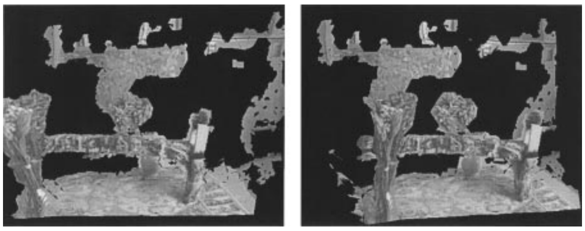

Figure 23. Two views of the "flower scene" 3D model.

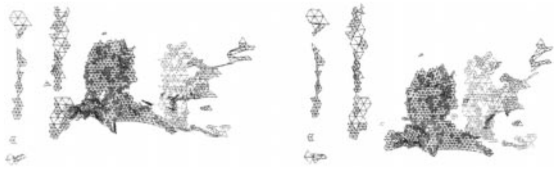

Figure 24. Two views of mesh model of the "headshot" scene.

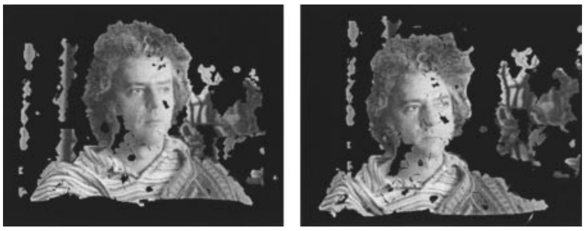

Figure 25. Two views of the "headshot" 3D model.

which minimizes the sum of squared errors between the reconstructed depth using the multi-camera reconstruction algorithm with triangular meshes and the ground truth. Then we estimated the relative error E_i at every point, that is, if Z_i is the actual depth value and $\hat{Z}_i$ the estimated depth value at point i, $E_i = \frac{\hat{Z}_i - Z_i}{Z_i}$.

Figures 27(a) and (b) show the reconstruction with and without mapped texture. Fig. 27(c) shows the relative error values, E_i, coded as gray values, where zero error corresponds to gray value 127, underestimates are shown darker, and overestimates lighter, and Fig. 27(d) shows the same data after histogram equalization. The statistical data of the relative error is as follows: $\text{mean}(E) = 0.002 = 0.2\%$, $\text{st.dev.}(E) = 0.108 = 10.8\%$, and $\text{skewness}(E) = 2.07$.

These and other experiments can be found on the World Wide Web. `http://www.cfar.umd.edu/~brodsky/yardstick.html` (the lab sequence) contains the original lab sequence, which was taken by a hand-held camera (calibration data unknown), the estimated location of the epipole for both our technique and weighted epipolar minimization, scene reconstruction, and de-rotation (subtracting the rotation from the original sequence). `http://www.cfar.umd.edu/~brodsky/reconst/reconst.html` shows a reconstruction of the original image on the basis of the reconstructed scene for the lab sequence. `http://www.cfar.umd.edu/~brodsky/models.html` shows reconstructions of the "flower scene" and "headshot" sequences, `http://www.cfar.umd.edu/~brodsky/pooh-models.html` shows the "pooh" sequence and `http://www.cfar.umd.edu/~brodsky/yosemite` shows the Yosemite fly-through sequence.

As can be verified, the depth estimation in all the sequences is of very high accuracy. However, the reconstructions are not yet perfect. To further improve the reconstructed models will require that one employs various techniques from computer graphics. For example, the depth maps are not dense everywhere as we compute normal flow only at locations with significant spatial gradient. To obtain depth values at these locations where there is no texture one would have to interpolate from the neighboring values.

There are also some errors in the depth estimation which arise from erroneous motion estimates. The instantaneous 3D motion estimates are generally very good, but when combining a sequence of frames small errors can compound and cause errors in depth. To address this problem one could use the estimated depth to compute better correspondence between features in views that are far apart and then use the results to recompute the motion and depth.

Also, one should keep in mind that only one short sequence (up to thirty-five frames) is used to build the depth maps. To compute a full, dense model will require that the information from multiple sequences is combined.

Some errors in depth estimation are due to unavoidable errors in the estimation of normal flow. For

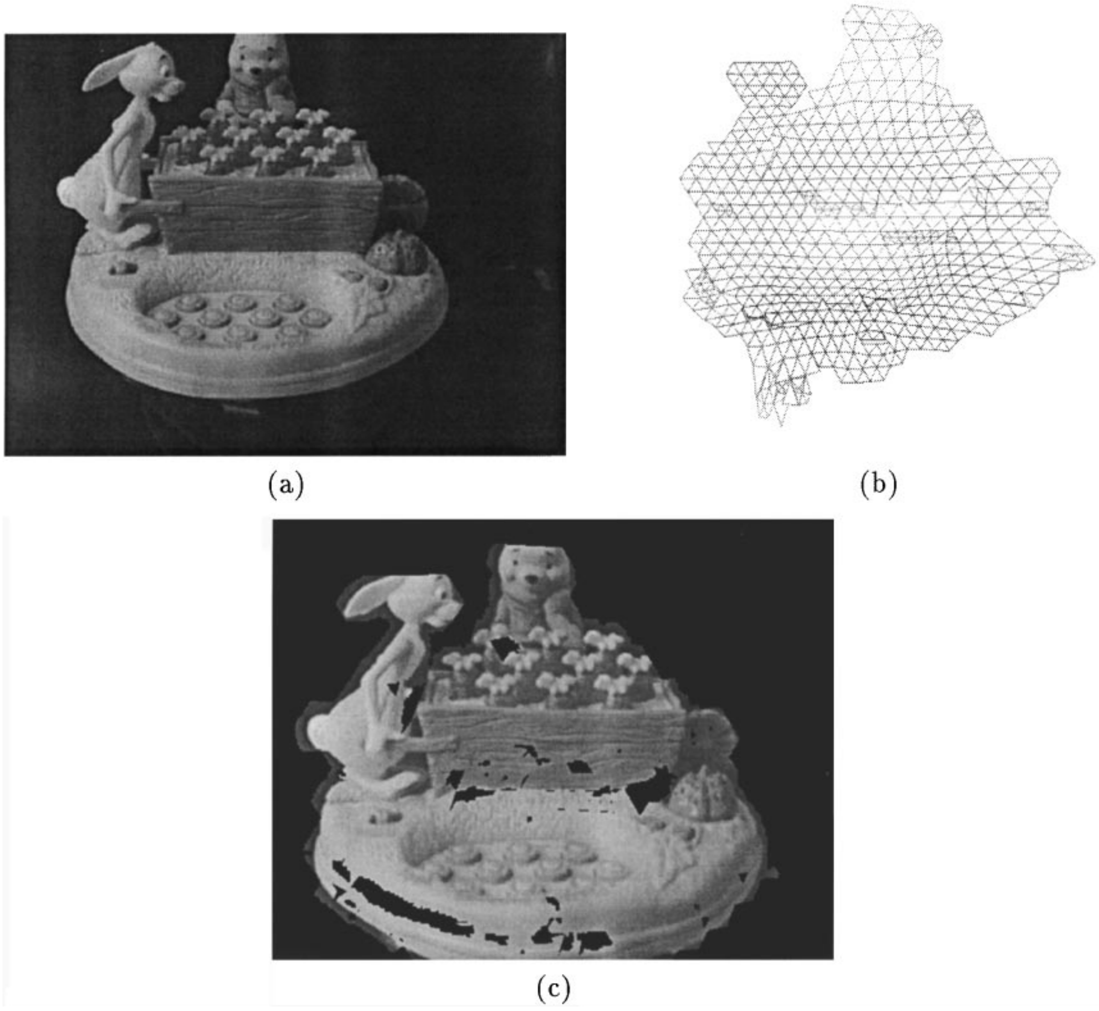

Figure 26. Reconstruction of “pooh” sequence: (a) one original frame, (b) mesh reconstruction; (c) 3D reconstruction with texture mapping.

example, in the “headshot” sequence, a part of the neck appears to be farther away than it should be. This is due to the shadows in the back where the brightness constancy assumption is violated. It will require combining many more frames from different views to account for this error.

Another source of error is the depth segmentation, which has not been optimized yet. By employing multiresolution strategies in the edge detection and other techniques from computational geometry it should be possible to improve upon the segmentation. Efforts along the lines discussed above are currently being undertaken.

Finally, for the interested reader, a note regarding the computational time: Most experiments used color images, 320 × 240 pixels. The images were converted to gray-scale for flow computation, and color was only used for the 3D model. Typical times for a single processor Ultra SPARC II running at 250 MHz are as follows: For the motion estimation stage, a full hierarchical search in the translation space requires approximately five minutes per flow field. A subsequent local search requires approximately forty seconds per flow field. The construction of 3D models is computationally expensive. In a typical case (for example, the “flower scene”), we perform stereo search over thirty-five input images. The initial search for all pixels in the image runs in about one hour and twenty minutes. The construction of triangles, filtering and rendering of the 3D model are much faster, requiring

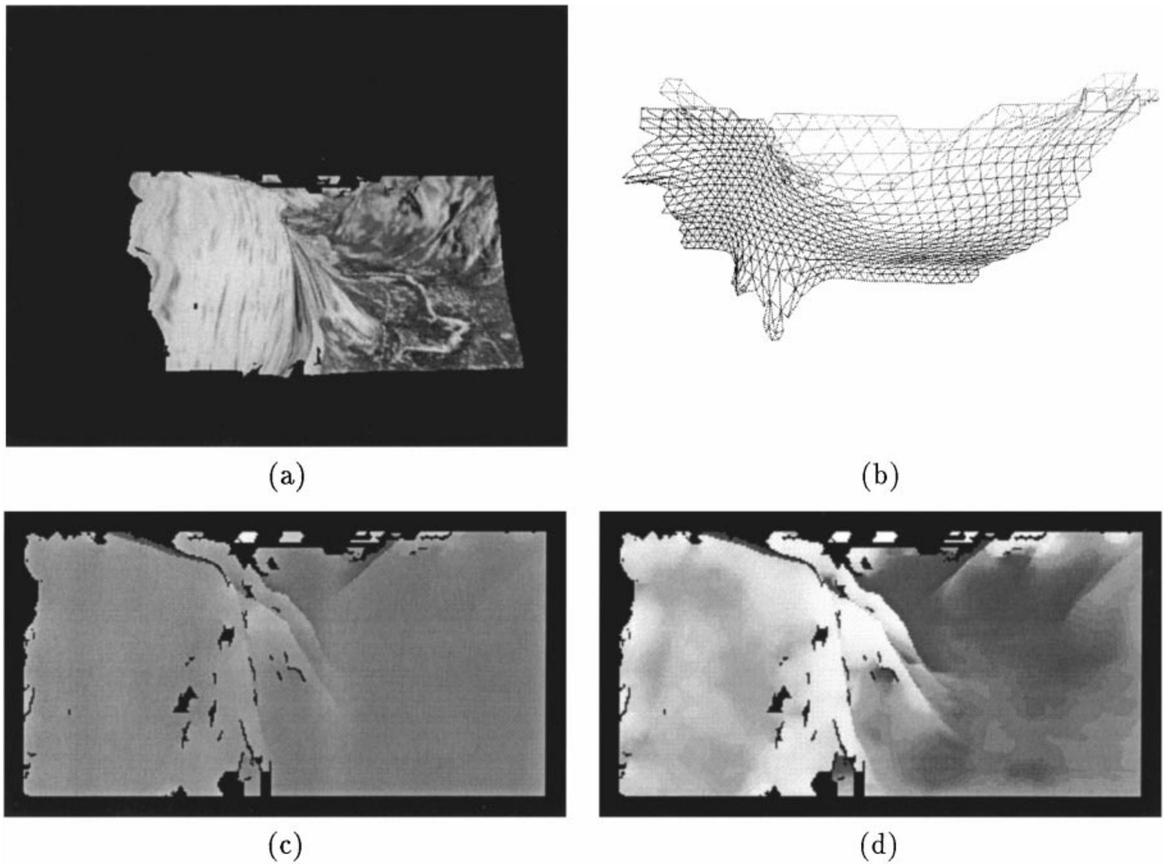

Figure 27. Yosemite sequence: (a) Reconstruction with texture mapping. (b) Mesh reconstruction. (c) Relative error map. (d) Histogram equalization of (c).

approximately five minutes for a seventy-frame fly-through sequence.

7. Conclusions

The structure of almost any scene can be described as a collection of smooth surface patches separated by abrupt discontinuities. This observation has been exploited in the algorithm described in this paper, which estimates the 3D viewing geometry while at the same time recovering scene discontinuities. The results of the algorithm have been applied to obtain scene reconstructions from short video sequences.

The technique is based on constraints which relate normal flow directly to 3D motion and structure. The constraints result from an understanding of the interaction between 3D shape and motion. Wrong 3D motion estimates give rise to estimates of smooth scene patches with depth values that vary locally more than those based on the correct motion estimates. In classical approaches to visual motion analysis the processes of smoothing, 3D motion estimation, and structure estimation are separated from each other. Statistical information obtained from the raw data in early processes (image motion estimation) is not utilized in later ones (3D motion and structure estimation). By combining the different processes one can utilize this information throughout. The constraints are thus of a geometrical and statistical nature, and provide the potential of solving the problem of structure from motion without making assumptions about the scene structure throughout the computations. The constraints have been analyzed and their relationship to minimization of the epipolar constraint has been established.

Future research will include further studies of the constraints, especially with regard to their statistical nature, for the purpose of designing efficient algorithms for the more general structure from motion problem dealing with multiple independently moving objects.

References

Barron, J.L., Fleet, D.J., and Beauchemin, S.S. 1994. Performance of optical flow techniques. *International Journal of Computer Vision*, 12:43–77.

Beardsley, P.A., Torr, P., and Zisserman, A. 1996. 3D model acquisition from extended image sequences. In *Proc. European Conference on Computer Vision*, Vol. 2, Cambridge, England, pp. 683–695.

Bergen, J.R., Anandan, P., Hanna, K.J., and Hingorani, R. 1992. Hierarchical model-based motion estimation. In *Proc. European Conference on Computer Vision*, pp. 237–248.

Brodský, T., Fermüller, C., and Aloimonos, Y. 1998a. Directions of motion fields are hardly ever ambiguous. *International Journal of Computer Vision*, 26:5–24.

Brodský, T., Fermüller, C., and Aloimonos, Y. 1998b. Self-calibration from image derivatives. In *Proc. International Conference on Computer Vision*, pp. 83–89.

Brodský, T., Fermüller, C., and Aloimonos, Y. 1998c. Shape from video: Beyond the epipolar constraint. In *Proc. DARPA Image Understanding Workshop*, pp. 1003–1012.

Brodský, T., Fermüller, C., and Aloimonos, Y. 1998d. Simultaneous estimation of viewing geometry and structure. In *Proc. European Conference on Computer Vision*, pp. 342–358.

Brodský, T., Fermüller, C., and Aloimonos, Y. 1999. Shape from video. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pp. 146–151.

Cheong, L., Fermüller, C., and Aloimonos, Y. 1998. Effects of errors in the viewing geometry on shape estimation. *Computer Vision and Image Understanding*, 71:356–372.

Cormen, T., Leiserson, C., and Rivest, R. 1989. *Introduction to Algorithms*. MIT Press.

Cox, I.J., Hingorani, S., Rao, S., and Maggs, B. 1996. A maximum-likelihood stereo algorithm. *Computer Vision and Image Understanding*, 63:542–567.

Daniilidis, K. and Spetsakis, M.E. 1997. Understanding noise sensitivity in structure from motion. In *Visual Navigation: From Biological Systems to Unmanned Ground Vehicles*, Y. Aloimonos, (Ed.)., Lawrence Erlbaum Associates: Mahwah, NJ, Advances in Computer Vision, Ch. 4.

Faugeras, O.D. 1992. *Three-Dimensional Computer Vision*. MIT Press: Cambridge, MA.

Faugeras, O.D. and Keriven, R. 1998. Complete dense stereo vision using level set methods. In *Proc. European Conference on Computer Vision*, Vol. I, pp. 379–393.

Fermüller, C. 1993. Navigational preliminaries. In *Active Perception*, Y. Aloimonos, (Ed.), Lawrence Erlbaum Associates: Hillsdale, NJ, Advances in Computer Vision, Ch. 3.

Fermüller, C. and Aloimonos, Y. 1995. Direct perception of three-dimensional motion from patterns of visual motion. *Science*, 270:1973–1976.

Fermüller, C., Pless, R., and Aloimonos, Y. 2000. The Ouchi illusion as an artifact of biased flow estimation. *Vision Research*, 40:77–96.

Fitzgibbon, A. and Zisserman, A. 1998. Automatic camera recovery for closed and open image sequences. In *Proc. European Conference on Computer Vision*, Vol. 1, Freiburg, Germany, pp. 311–326.

Foley, J.M. 1980. Binocular distance perception. *Psychological Review*, 87:411–434.

Geman, S. and Geman, D. 1984. Stochastic relaxation, Gibbs distribution and Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741.

Hartley, R.I. 1994. An algorithm for self calibration from several views. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pp. 908–912.

Heitz, F. and Bouthemy, P. 1993. Multimodal estimation of discontinuous optical flow using Markov random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15:1217–1232.

Horn, B.K.P. 1986. *Robot Vision*. McGraw Hill: New York.

Horn, B.K.P. 1987. Motion fields are hardly ever ambiguous. *International Journal of Computer Vision*, 1:259–274.

Horn, B.K.P. and Weldon, E.J., Jr. 1988. Direct methods for recovering motion. *International Journal of Computer Vision*, 2:51–76.

Koch, R., Pollefeys, M., and Gool, L.V. 1998. Multi viewpoint stereo from uncalibrated video sequences. In *Proc. European Conference on Computer Vision*, Vol. I, pp. 55–71.

Koenderink, J.J. and van Doorn, A.J. 1991. Affine structure from motion. *Journal of the Optical Society of America*, 8:377–385.

Lucas, B.D. 1984. *Generalized image matching by the method of differences*. Ph.D. Thesis, Dept. of Computer Science, Carnegie-Mellon University.

Luong, Q.-T. and Faugeras, O.D. 1996. The fundamental matrix: Theory, algorithms, and stability analysis. *International Journal of Computer Vision*, 17:43–75.

Marroquin, J. 1985. *Probabilistic solution of inverse problems*. Ph.D. Thesis, Institute of Technology, Massachusetts.

Maybank, S.J. 1986. Algorithm for analysing optical flow based on the least-squares method. *Image and Vision Computing*, 4:38–42.

Maybank, S.J. 1987. *A theoretical study of optical flow*. Ph.D. Thesis, University of London.

Mendelsohn, J., Simoncelli, E., and Bajcsy, R. 1997. Discrete-time rigidity constrained optical flow. In *Proc. International Conference on Computer Analysis of Images and Patterns*, Springer, Berlin, pp. 255–262.

Mumford, D. and Shah, J. 1985. Boundary detection by minimizing functionals. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pp. 22–25.

Murray, D.W. and Buxton, B.F. 1987. Scene segmentation from visual motion using global optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9:220–228.

Okutomi, M. and Kanade, T. 1993. A multiple-baseline stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15: 353–363.

Pollefeys, M., Koch, R., and Van Gool, L. 1998. Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters. In *Proc. International Conference on Computer Vision*, pp. 90–95.

Robert, L. and Deriche, R. 1996. Dense depth map reconstruction: A minimization and regularization approach which preserves discontinuities. In *Proc. European Conference on Computer Vision*, Vol. I, pp. 439–451.

Schunck, B.G. 1989. Image flow segmentation and estimation by constraint line clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11:1010–1027.

Seitz, S.M. and Dyer, C. 1997. Photorealistic scene reconstruction by voxel coloring. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1067–1073.

Spoerri, A. and Ullman, S. 1987. The early detection of motion boundaries. In *Proc. International Conference on Computer Vision*, pp. 209–218.

Szeliski, R. and Golland, P. 1998. Stereo matching with transparency and matting. In *Proc. International Conference on Computer Vision*, pp. 517–524.

Thompson, W.B., Mutch, K.M., and Berzins, V.A. 1985. Dynamic occlusion analysis in optical flow fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 7:374–383.

Tittle, J.S., Todd, J.T., Perotti, V.J., and Norman, J.F. 1995. Systematic distortion of perceived three-dimensional structure from motion and binocular stereopsis. *Journal of Experimental Psychology: Human Perception and Performance*, 21:663–678.