Direct Recovery of Planar-Parallax from Multiple Frames

Michal Irani[1], P. Anandan[2], and Meir Cohen[1]

[1] Dept. of Computer Science and Applied Math, The Weizmann Inst. of Science, Rehovot, Israel,
irani@wisdom.weizmann.ac.il
[2] Microsoft Research, One Microsoft Way, Redmond, WA 98052, USA,
anandan@microsoft.com

Abstract. In this paper we present an algorithm that estimates dense planar-parallax motion from *multiple uncalibrated* views of a 3D scene. This generalizes the "plane + parallax" recovery methods to more than two frames. The parallax motion of pixels across multiple frames (relative to a planar surface) is related to the 3D scene structure and the camera epipoles. The parallax field, the epipoles, and the 3D scene structure are estimated directly from image brightness variations across multiple frames, without pre-computing correspondences.

1 Introduction

The recovery of the 3D structure of a scene and the camera epipolar-geometries (or camera motion) from multiple views has been a topic of considerable research. The large majority of the work on structure-from-motion (SFM) has assumed that correspondences between image features (typically a sparse set of image points) is given, and focused on the problem of recovering SFM based on this input. Another class of methods has focused on recovering dense 3D structure from a set of dense correspondences or an optical flow field. While these have the advantage of recovering *dense* 3D structure, they require that the correspondences are known. However, correspondence (or flow) estimation is a notoriously difficult problem.

A small set of techniques have attempted to combine the correspondence estimation step together with SFM recovery. These methods obtain dense correspondences while *simultaneously* estimating the 3D structure and the camera geometries (or motion) [3,11,13,16,15]. By inter-weaving the two processes, the local correspondence estimation process is constrained by the current estimate of (global) epipolar geometry (or camera motion), and vice-versa. These techniques minimize the violation of the brightness gradient constraint with respect to the unknown structure and motion parameters. Typically this leads to a significant improvement in the estimated correspondences (and the attendant 3D structure) and some improvement in the recovered camera geometries (or motion). These methods are sometimes referred to as "direct methods" [3], since they directly

B. Triggs, A. Zisserman, R. Szeliski (Eds.): Vision Algorithms'99, LNCS 1883, pp. 85–99, 2000.

use image brightness information to recover 3D structure and motion, without explicitly computing correspondences as an intermediate step.

While [3,16,15] recover 3D information relative to a *camera-centered* coordinate system, an alternative approach has been proposed for recovering 3D structure in a *scene-centered* coordinate system. In particular, the "Plane+Parallax" approach [14,11,13,7,9,8], which analyzes the parallax displacements of points relative to a (real or virtual) physical planar surface in the scene (the "reference plane"). The underlying concept is that after the alignment of the reference plane, the residual image motion is due only to the *translational* motion of the camera and to the *deviations* of the scene structure from the planar surface. All effects of camera rotation or changes in camera calibration are eliminated by the plane stabilization. Hence, the residual image motion (the planar-parallax displacements) form a *radial flow field* centered at the epipole.

The "Plane+Parallax" representation has several benefits over the traditional camera-centered representation, which make it an attractive framework for correspondence estimation and for 3D shape recovery:

1. *Reduced search space:* By parametrically aligning a visible image structure (which usually corresponds to a planar surface in the scene), the search space of unknowns is significantly reduced. Globally, all effects of unknown rotation and calibration parameters are folded into the homographies used for patch alignment. The only remaining unknown global camera parameters which need to be estimated are the epipoles (i.e., 3 global unknowns per frame; gauge ambiguity is reduced to a single global scale factor for all epipoles across all frames). Locally, because after plane alignment the unknown displacements are constrained to lie along radial lines emerging from the epipoles, local correspondence estimation reduces from a 2-D search problem into a simpler 1-D search problem at each pixel. The 1-D search problem has the additional benefit that it can uniquely resolve correspondences, even for pixels which suffer from the aperture problem (i.e., pixels which lie on line structures).
2. *Provides shape relative to a plane in the scene:* In many applications, distances from the camera are not as useful information as fluctuations with respect to a plane in the scene. For example, in robot navigation, heights of scene points from the ground plane can be immediately translated into obstacles or holes, and can be used for obstacle avoidance, as opposed to distances from the camera.
3. *A compact representation:* By removing the mutual global component (the plane homography), the residual parallax displacements are usually very small, and hence require significantly fewer bits to encode the shape fluctuations relative to the number of bits required to encode distances from the camera. This is therefore a compact representation, which also supports progressive encoding and a high resolution display of the data.
4. *A stratified 2D-3D representation:* Work on motion analysis can be roughly classified into two classes of techniques: 2D algorithms which handle cases with no 3D parallax (e.g., estimating homographies, 2D affine transforma-

tions, etc), and 3D algorithms which handle cases with dense 3D parallax (e.g., estimating fundamental matrices, trifocal tensors, 3D shape, etc). Prior *model selection* [17] is usually required to decide which set of algorithms to apply, depending on the underlying scenario. The Plane+Parallax representation provides a *unified* approach to 2D and 3D scene analysis, with a strategy to gracefully bridge the gap between those two extremes [10]. Within the Plane+Parallax framework, the analysis always starts with 2D estimation (i.e., the homography estimation). When that is all the information available in the image sequence, that is where the analysis stops. The 3D analysis then gradually builds *on top* of the 2D analysis, with the gradual increase in 3D information (in the form of planar-parallax displacements and shape-fluctuations w.r.t. the planar surface).

[11,13] used the Plane+Parallax framework to recover dense structure relative to the reference plane from *two* uncalibrated views. While their algorithm linearly solves for the structure directly from brightness measurements in two frames, it does not naturally extend to multiple frames. In this paper we show how dense planar-parallax displacements and relative structure can be recovered directly from brightness measurements in *multiple* frames. Furthermore, we show that many of the ambiguities existing in the two-frame case of [11,13] are resolved by extending the analysis to multiple frames. Our algorithm assumes as input a sequence of images in which a planar surface has been previously aligned with respect to a reference image (e.g., via one of the 2D parametric estimation techniques, such as [1,6]). We do *not* assume that the camera calibration information is known. The output of the algorithm is: (i) the epipoles for all the images with respect to the reference image, (ii) dense 3D structure of the scene relative to a planar surface, and (iii) the correspondences of all the pixels across all the frames, which must be consistent with (i) and (ii). The estimation process uses the *exact* equations (as opposed to *instantaneous* equations, such as in [4,15]) relating the residual parallax motion of pixels across *multiple* frames to the relative 3D structure and the camera epipoles. The 3D scene structure and the camera epipoles are computed directly from image measurements by minimizing the variation of image brightness across the views without pre-computing a correspondence map.

The current implementation of our technique relies on the prior alignment of the video frames with respect to a planar surface (similar to other plane+parallax methods). This requires that a real physical plane exists in the scene and is visible in all the video frames. However, this approach can be extended to arbitrary scenes by folding in the plane homography computation also into the simultaneous estimation of camera motion, scene structure, and image displacements (as was done by [11] for the case of *two* frames).

The remainder of the paper describes the algorithm and shows its performance on real and synthetic data. Section 2 shows how the 3D structure relates to the 2D image displacement under the plane+parallax decomposition. Section 3 outlines the major steps of our algorithm. The benefits of applying the algorithm to multiple frames (as opposed to two frames) are discussed in Sec-

tion 4. Section 5 shows some results of applying the algorithm to real data. Section 6 concludes the paper.

2 The Plane+Parallax Decomposition

The induced 2D image motion of a 3D scene point between two images can be decomposed into two components [9,7,10,11,13,14,8,2]: (i) the image motion of a reference planar surface Π (i.e., a homography), and (ii) the residual image motion, known as "planar parallax". This decomposition is described below.

To set the stage for the algorithm described in this paper, we begin with the derivation of the plane+parallax motion equations shown in [10]. Let $\boldsymbol{p} = (x, y, 1)$ denote the image location (in homogeneous coordinates) of a point in one view (called the "reference view"), and let $\boldsymbol{p'} = (x', y', 1)$ be its coordinates in another view. Let $\mathbf{B}$ denote the homography of the plane Π between the two views. Let $\mathbf{B}^{-1}$ denote its inverse homography, and $\mathbf{B}^{-1}{}_3$ be the third row of B^{-1}. Let $\boldsymbol{p_w} = (x_w, y_w, 1) = \frac{\mathbf{B}^{-1}\boldsymbol{p'}}{\mathbf{B}^{-1}{}_3\boldsymbol{p'}}$, namely, when the second image is warped towards the first image using the inverse homography B^{-1}, the point $\boldsymbol{p'}$ will move to the point $\boldsymbol{p_w}$ in the *warped image*. For 3D points on the plane Π, $\boldsymbol{p_w} = \boldsymbol{p}$, while for 3D points which are not on the plane, $\boldsymbol{p_w} \neq \boldsymbol{p}$. It was shown in [10] that[1]:

$$\boldsymbol{p'} - \boldsymbol{p} = (\boldsymbol{p'} - \boldsymbol{p_w}) + (\boldsymbol{p_w} - \boldsymbol{p})$$

and

$$\boldsymbol{p_w} - \boldsymbol{p} = -\gamma(t_3\boldsymbol{p_w} - \boldsymbol{t}) \tag{1}$$

where $\gamma = H/Z$ represents the 3D structure of the point $\boldsymbol{p}$, where H is the perpendicular distance (or "height") of the point from the reference plane Π, and Z is its depth with respect to the reference camera. All *unknown* calibration parameters are folded into the terms in the parenthesis, where t denotes the epipole in projective coordinates and t_3 denotes its third component: $\boldsymbol{t} = (t_1, t_2, t_3)$.

In its current form, the above expression cannot be directly used for estimating the unknown correspondence $\boldsymbol{p_w}$ for a given pixel $\boldsymbol{p}$ in the reference image, since $\boldsymbol{p_w}$ appears on both sides of the expression. However, $\boldsymbol{p_w}$ can be eliminated from the right hand side of the expression, to obtain the following expression:

$$\boldsymbol{p_w} - \boldsymbol{p} = -\frac{\gamma}{1 + \gamma t_3}(t_3\boldsymbol{p} - \boldsymbol{t}). \tag{2}$$

This last expression will be used in our direct estimation algorithm.

3 Multi-frame Parallax Estimation

Let $\{\Phi_j\}_{j=0}^{l}$ be $l+1$ images of a rigid scene, taken using cameras with unknown calibration parameters. Without loss of generality, we choose Φ_0 as a reference

[1] The notation we use here is slightly different than the one used in [10]. The change to projective notation is used to unify the two separate expressions provided in [10], one for the case of a finite epipole, and the other for the case of an infinite epipole.

frame. (In practice, this is usually the middle frame of the sequence). Let Π be a plane in the scene that is visible in all l images (the "reference plane"). Using a technique similar to [1,6], we estimate the image motion (homography) of Π between the reference frame Φ_0 and each of the other frames Φ_j $(j = 1, \ldots, l)$. Warping the images by those homographies $\{\mathbf{B}_j\}_{j=1}^{l}$ yields a new sequence of l images, $\{I_j\}_{j=1}^{l}$, where the image of Π is aligned across all frames. Also, for the sake of notational simplicity, let us rename the reference image to be I, i.e., $I = \Phi_0$. The only residual image motion between reference frame I and the warped images, $\{I_j\}_{j=1}^{l}$, is the residual planar-parallax displacement $\boldsymbol{p}_{\boldsymbol{w}}^{\boldsymbol{j}} - \boldsymbol{p}$ $(j = 1..l)$ due to 3D scene points that are *not* located on the reference plane Π. This residual planar parallax motion is what remains to be estimated.

Let $\boldsymbol{u}^{\boldsymbol{j}} = (u^j, v^j)$ denote the first two coordinates of $\boldsymbol{p}_{\boldsymbol{w}}^{\boldsymbol{j}} - \boldsymbol{p}$ (the third coordinate is 0). From Eq. (2) we know that the residual parallax is:

$$\boldsymbol{u}^{\boldsymbol{j}} = \begin{bmatrix} u^j \\ v^j \end{bmatrix} = -\frac{\gamma}{1 + \gamma t_3^j} \begin{bmatrix} t_3^j x - t_1^j \\ t_3^j y - t_2^j \end{bmatrix}, \tag{3}$$

where the superscripts j denote the parameters associated with the jth frame.

In the *two*-frame case, one can define $\alpha = \frac{\gamma}{1+\gamma t_3}$, and then the problem posed in Eq. (3) becomes a bilinear problem in α and in $\boldsymbol{t} = (t_1, t_2, t_3)$. This can be solved using a standard iterative method. Once α and $\boldsymbol{t}$ are known, γ can be recovered. A similar approach was used in [11] for shape recovery from two-frames. However, this approach does not extend to multiple (> 2) frames, because α is *not* a shape invariant (as it depends on t_3), and hence varies from frame to frame. In contrast, γ *is* a shape invariant, which is shared by all image frames. Our multi-frame process directly recovers γ from *multi-frame* brightness quantities.

The basic idea behind our direct estimation algorithm is that rather than estimating l separate $\boldsymbol{u}^{\boldsymbol{j}}$ vectors (corresponding to each frame) for each pixel, we can simply estimate a single γ (the shape parameter), which for a particular pixel, is common over all the frames, and a single $\boldsymbol{t}^{\boldsymbol{j}} = (t_1, t_2, t_3)$ which for each frame I_j is common to all image pixels. There are two advantages in doing this:

1. For n pixels over l frames we reduce the number of unknowns from $2nl$ to $n + 3l$.
2. More importantly, the recovered flow vector is constrained to satisfy the epipolar structure implicitly captured in Eq. (2). This can be expected to significantly improve the quality of the recovered parallax flow vectors.

Our direct estimation algorithm follows the same computational framework outlined in [1] for the *quasi-parametric* class of models. The basic components of this framework are: (i) pyramid construction, (ii) iterative estimation of global (motion) and local (structure) parameters, and (iii) coarse-to-fine refinement. The overall control loop of our algorithm is therefore as follows:

1. Construct pyramids from each of the images I_j and the reference frame I.
2. Initialize the structure parameter γ for each pixel, and motion parameter $\boldsymbol{t^j}$ for each frame (usually we start with $\gamma = 0$ for all pixels, and $\boldsymbol{t^j} = (\,0, 0, 1\,)^T$ for all frames).
3. Starting with the coarsest pyramid level, at each level, refine the structure and motion using the method outlined in Section 3.1.
4. Repeat this step several times (usually about 4 or 5 times per level).
5. Project the final value of the structure parameter to the next finer pyramid level. Propagate the motion parameters also to the next level. Use these as initial estimates for processing the next level.
6. The final output is the structure and the motion parameters at the finest pyramid level (which corresponds to the resolution of the input images) and the residual parallax flow field synthesized from these.

Of the various steps outline above, the pyramid construction and the projection of parameters are common to many techniques for motion estimation (e.g., see [1]), hence we omit the description of these steps. On the other hand, the refinement step is specific to our current problem. This is described next.

3.1 The Estimation Process

The inner loop of the estimation process involves refining the current values of the structure parameters γ (one per pixel) and the motion parameters $\boldsymbol{t^j}$ (3 parameters per frame). Let us denote the "true" (but unknown) values of these parameters by $\gamma(x, y)$ (at location (x, y) in the reference frame) and $\boldsymbol{t^j}$. Let $\boldsymbol{u^j}(x, y) = (u^j, v^j)$ denote the corresponding unknown true parallax flow vector. Let $\gamma_c, \boldsymbol{t_c^j}, \boldsymbol{u_c^j}$ denote the *current estimates* of these quantities. Let $\delta\gamma = \gamma - \gamma_c$, $\delta\boldsymbol{t^j} = (\delta t_1^j, \delta t_2^j, \delta t_3^j) = \boldsymbol{t^j} - \boldsymbol{t_c^j}$, and $\delta\boldsymbol{u^j} = (\delta u^j, \delta v^j) = \boldsymbol{u^j} - \boldsymbol{u_c^j}$. These δ quantities are the refinements that are estimated during each iteration.

Assuming brightness constancy (namely, that corresponding image points across all frames have a similar brightness value)[2], we have:

$$I(x, y) \approx I_j(x^j, y^j) = I_j(x + u^j, y + v^j) = I_j(x + u_c^j + \delta u^j, y + v_c^j + \delta v^j)$$

For small $\boldsymbol{\delta u^j}$ we make a further approximation:

$$I(x - \delta u^j, y - \delta v^j) \approx I_j(x + u_c^j, y + v_c^j).$$

Expanding I to its first order Taylor series around (x, y) :

$$I(x - \delta u^j, y - \delta v^j) \approx I(x, y) - I_x \delta u^j - I_y \delta v^j$$

[2] Note that over multiple frames the brightness will change somewhat, at least due to global illumination variation. We can handle this by using the Laplacian pyramid (as opposed to the Gaussian pyramid), or otherwise pre-filtering the images (e.g., normalize to remove global mean and contrast changes), and applying the brightness constraint to the filtered images.

where I_x, I_y denote the image intensity derivatives for the reference image (at pixel location (x, y)). From here we get the brightness constraint equation:

$$I_j(x + u_c^j, y + v_c^j) \approx I(x, y) - I_x \delta u^j - I_y \delta v^j$$

Or:

$$I_j(x + u_c^j, y + v_c^j) - I(x, y) + I_x \delta u^j + I_y \delta v^j \approx 0$$

Substituting $\boldsymbol{\delta u^j = u^j - u_c^j}$ yields:

$$I_j(x + u_c^j, y + v_c^j) - I(x, y) + I_x(u^j - u_c^j) + I_y(v^j - v_c^j) \approx 0$$

Or, more compactly:

$$I_j^\tau(x, y) + I_x u^j + I_y v^j \approx 0 \tag{4}$$

where

$$I_j^\tau(x, y) \stackrel{\text{def}}{=} I_j(x + u_c^j, y + v_c^j) - I(x, y) - I_x u_c^j - I_y v_c^j$$

If we now substitute the expression for the local parallax flow vector $\boldsymbol{u^j}$ given in Eq. (3), we obtain the following equation that relates the structure and motion parameters directly to image brightness information:

$$I_j^\tau(x, y) \ + \frac{\gamma(x,y)}{1+\gamma(x,y)t_3^j} \left(I_x(t_3^j x - t_1^j) + I_y(t_3^j y - t_2^j) \right) \approx 0 \tag{5}$$

We refer to the above equation as the "epipolar brightness constraint".

Each pixel and each frame contributes one such equation, where the unknowns are: the relative scene structure $\gamma = \gamma(x, y)$ for each pixel (x, y), and the epipoles $\boldsymbol{t^j}$ for each frame $(j = 1, 2, \ldots, l)$. Those unknowns are computed in two phases. In the first phase, the "Local Phase", the relative scene structure, γ, is estimated separately for each pixel via least squares minimization over multiple frames simultaneously. This is followed by the "Global Phase", where all the epipoles $\boldsymbol{t^j}$ are estimated between the reference frame and each of the other frames, using least squares minimization over all pixels. These two phases are described in more detail below.

Local Phase In the local phase we assume all the epipoles are given (e.g., from the previous iteration), and we estimate the unknown scene structure γ from all the images. γ is a local quantity, but is common to all the images at a point. When the epipoles are known (e.g., from the previous iteration), each frame I_j provides one constraint of Eq. (5) on γ. Therefore, theoretically, there is sufficient geometric information for solving for γ. However, for increased numerical stability, we locally assume each γ is constant over a small window around each pixel in the reference frame. In our experiments we used a 5×5 window. For each pixel (x, y), we use the error function:

$$Err(\gamma) \stackrel{\text{def}}{=} \sum_j \sum_{(\tilde{x},\tilde{y}) \in \text{Win}(x,y)} \left(\tilde{I}_j^\tau (1 + \gamma t_3^j) + \gamma \left(\tilde{I}_x(t_3^j \tilde{x} - t_1^j) + \tilde{I}_y(t_3^j \tilde{y} - t_2^j) \right) \right)^2 \tag{6}$$

where $\gamma = \gamma(x,y)$, $\tilde{I}_j^\tau = I_j^\tau(\tilde{x},\tilde{y})$, $\tilde{I}_x = I_x(\tilde{x},\tilde{y})$, $\tilde{I}_y = I_y(\tilde{x},\tilde{y})$, and $\mathrm{Win}(x,y)$ is a 5×5 window around (x,y). Differentiating $Err(\gamma)$ with respect to γ and equating it to zero yields a single linear equation that can be solved to estimate $\gamma(x,y)$. The error term $Err(\gamma)$ was obtained by multiplying Eq. (5) by the denominator $(1+\gamma t_3^j)$ to yield a linear expression in γ. Note that without multiplying by the denominator, the local estimation process (after differentiation) would require solving a polynomial equation in γ whose order increases with l (the number of frames). Minimizing $Err(\gamma)$ is in practice equivalent to applying *weighted* least squares minimization on the collection of original Eqs. (5), with weights equal to the denominators. We could apply *normalization* weights $\frac{1}{1+\gamma_c t_3^j}$ (where γ_c is the estimate of the shape at pixel (x,y) from the previous iteration) to the linearized expression, in order to assure minimization of meaningful quantities (as is done in [18]), but in practice, for the examples we used, we found it was not necessary to do so during the local phase. However, such a normalization weight was important during the global phase (see below).

Global Phase In the global phase we assume the structure γ is given (e.g., from previous iteration), and we estimate for each image I_j the position of its epipole $\boldsymbol{t^j}$ with respect to the reference frame. We estimate the set of epipoles $\{\boldsymbol{t^j}\}$ by minimizing the following error with respect each of the epipoles:

$$Err(\boldsymbol{t^j}) \stackrel{\text{def}}{=} \sum_{(x,y)} \Big(W_j(x,y) \Big[I_j^\tau(1+\gamma t_3^j) + \gamma \Big(I_x(t_3^j x - t_1^j) + I_y(t_3^j y - t_2^j) \Big) \Big] \Big)^2 \tag{7}$$

where $I_x = I_x(x,y), I_y = I_y(x,y), I_j^\tau = I_j^\tau(x,y), \gamma = \gamma(x,y)$. Note that, when $\gamma(x,y)$ are fixed, this minimization problem decouples into a set of separate individual minimization problems, each a function of one epipole $\boldsymbol{t^j}$ for the jth frame. The inside portion of this error term is similar to the one we used above for the local phase, with the addition of a scalar weight $W_j(x,y)$. The scalar weight is used to serve two purposes. First, if Eq. (7) did not contain the weights $W_j(x,y)$, it would be equivalent to a *weighted* least squares minimization of Eq. (5), with weights equal to the denominators $(1+\gamma(x,y)t_3^j)$. While this provides a convenient linear expression in the unknown $\boldsymbol{t^j}$, these weights are not physically meaningful, and tend to skew the estimate of the recovered epipole. Therefore, in a fashion similar to [18], we choose the weights $W_j(x,y)$ to be $(1+\gamma(x,y)t_{3,c}^j)^{-1}$, where the γ is the updated estimate from the local phase, whereas the $t_{3,c}^j$ is based on the current estimate of $\boldsymbol{t^j}$ (from the previous iteration).

The scalar weight also provides us an easy way to introduce additional robustness to the estimation process in order to reduce the contribution of pixels that are potentially outliers. For example, we can use weights based on residual misalignment of the kind used in [6].

4 Multi-frame vs. Two-Frame Estimation

The algorithm described in Section 3 extends the plane+parallax estimation to multiple frames. The most obvious benefit of multi-frame processing is the improved signal-to-noise performance that is obtained due to having a larger set of independent samples. However, there are two additional benefits to multi-frame estimation: (i) overcoming the *aperture problem*, from which the two-frame estimation often suffers, and (ii) resolving the singularity of shape recovery in the vicinity of the epipole (we refer to this as the *epipole singularity*).

4.1 Eliminating the Aperture Problem

When only two images are used as in [11,13], there exists only one epipole. The residual parallax lies along epipolar lines (centered at the epipole, see Eq. (3)). The epipolar field provides one line constraint on each parallax displacement, and the Brightness Constancy constraint forms another line constraint (Eq. (4)). When those lines are not parallel, their intersection uniquely defines the parallax displacement. However, if the image gradient at an image point is parallel to the epipolar line passing through that point, then its parallax displacement (and hence its structure) can not be uniquely determined. However, when multiple images with multiple epipoles are used, then this ambiguity is resolved, because the image gradient at a point can be parallel to at most one of the epipolar lines associated with it. This observation was also made by [4,15].

To demonstrate this, we used a sequence composed of 9 images (105×105 pixels) of 4 squares (30×30 pixels) moving over a stationary textured background (which plays the role of the aligned reference plane). The 4 squares have the same motion: first they were all shifted to the right (one pixel per frame) to generate the first 5 images, and then they were all shifted down (one pixel per frame) to generate the next 4 images. The width of the stripes on the squares is 5 pixels. A sample frame is shown in Fig. 1.a (the fifth frame).

The epipoles that correspond to this motion are at infinity, the horizontal motion has an epipole at $(\infty, 52.5]$, and the vertical motion has an epipole at $[52.5, \infty)$. The texture on the squares was selected so that the spatial gradients of one square are parallel to the direction of the horizontal motion, another square has spatial gradients parallel to the direction of the vertical motion, and the two other squares have spatial gradients in multiple directions. We have tested the algorithm on three cases: (i) pure vertical motion, (ii) pure horizontal motion, and (iii) mixed motions.

Fig. 1.b is a typical depth map that results from applying the algorithm to sequences with purely vertical motion. (Dark grey corresponds to the reference plane, and light grey corresponds to elevated scene parts, i.e., the squares). The structure for the square with vertical bars is not estimated well as expected, because the epipolar constraints are parallel to those bars. This is true even when the algorithm is applied to multiple frames with the same epipole.

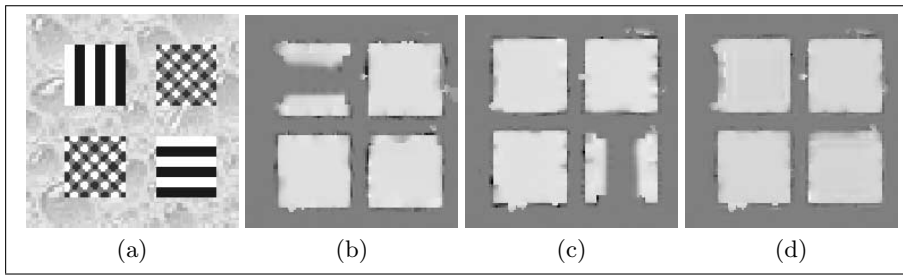


Fig. 1. Resolving aperture problem: (a) A sample image, (b) Shape recovery for pure vertical motion. Ambiguity along vertical bars, (c) Shape recovery for pure horizontal motion. Ambiguity along horizontal bars, (d) Shape recovery for a sequence with mixed motions. No ambiguity.

Fig. 1.c is a typical depth map that results from applying the algorithm to sequences with purely horizontal motion. Note that the structure for the square with horizontal bars is not estimated well.

Fig. 1.d is a typical depth map that results from applying the algorithm to multiple images with mixed motions (i.e., more than one distinct epipole). Note that now the shape recovery does not suffer from the aperture problem.

4.2 Epipole Singularity

From the planar parallax Eq. (3), it is clear that the structure γ cannot be determined at the epipole, because at the epipole: $t_3^j x - t_1^j = 0$ and $t_3^j y - t_2^j = 0$. For the same reason, the recovered structure at the *vicinity* of the epipole is highly sensitive to noise and unreliable. However, when there are multiple epipoles, this ambiguity disappears. The singularity at one epipole is resolved by information from another epipole.

To test this behavior, we compared the results for the case with only one epipole (i.e., two-frames) to cases with multiple epipoles at different locations. Results are shown in Fig. 2. The sequence that we used was composed of images of a square that is elevated from a reference plane and the simulated motion (after plane alignment) was a looming motion (i.e., forward motion). Fig. 2.a,b,c show three sample images from the sequence. Fig. 2.d shows singularity around the epipole in the two-frame case. Figs. 2.e,h,i,j show that the singularity at the epipoles is *eliminated* when there is more than one epipole. Using more images also increases the signal to noise ratio and further improves the shape reconstruction.

5 Real World Examples

This section provides experimental results of applying our algorithm to real world sequences. Fig. 3 shows an example of shape recovery from an indoor sequence

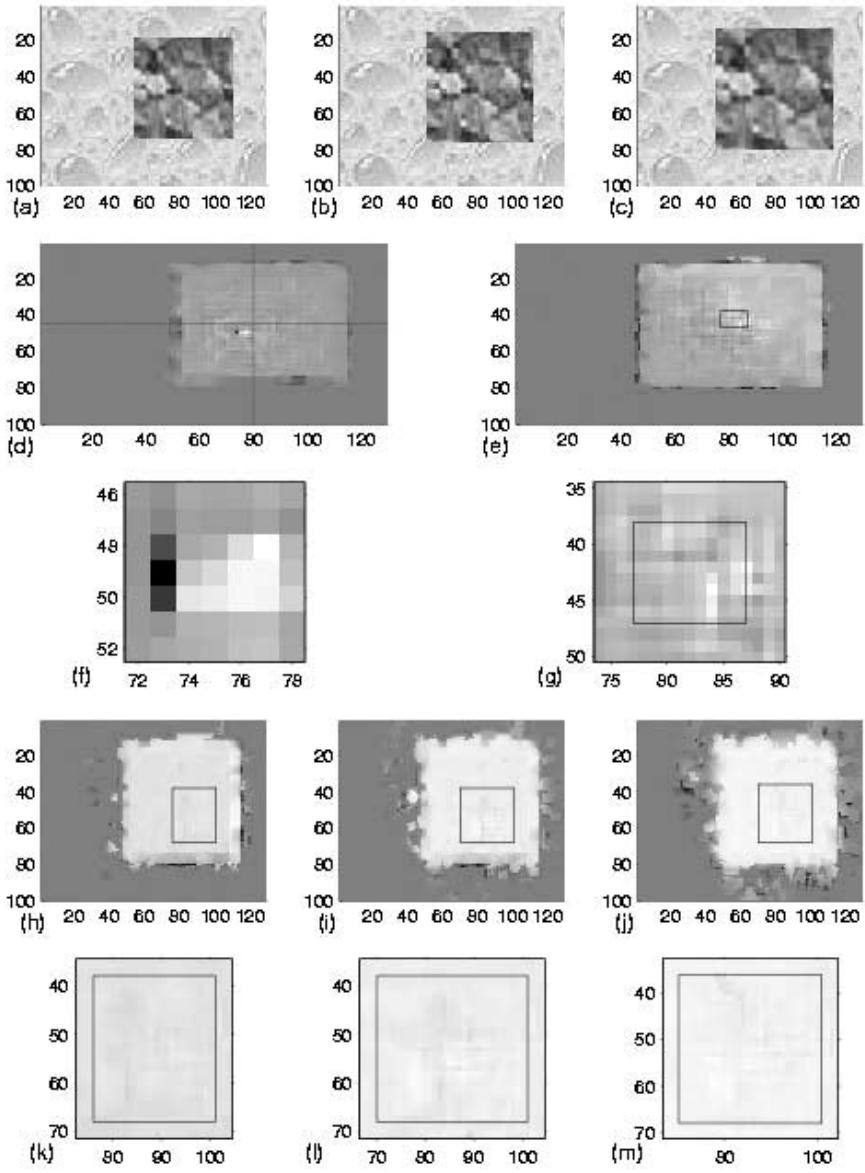
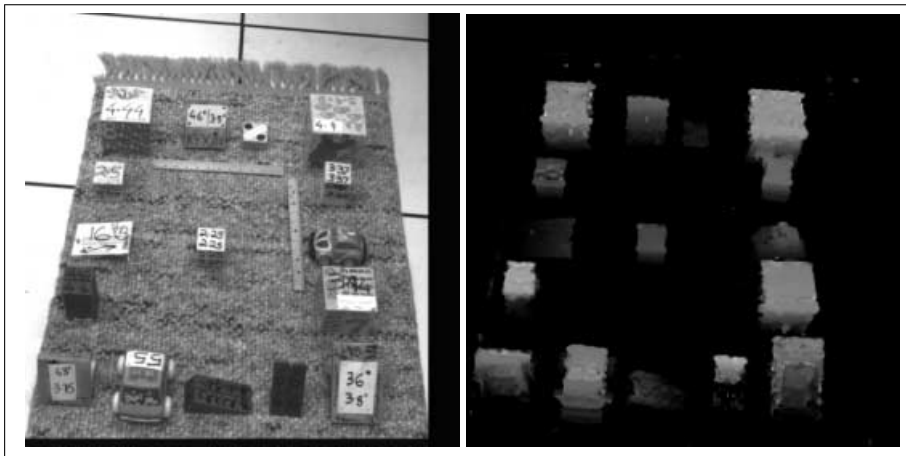


Fig. 2. Resolving epipole singularity in case of multiple epipoles. (a-c) sample images from a 9-frame sequence with multiple epipoles, (d,f) shape recovery using 2 images (epipole singularity exist in this case), (e,g) using 3 images with 2 different epipoles, (h,k) using 5 images with multiple epipoles, (i,l) using 7 images with multiple epipoles, (j,m) using 9 images with multiple epipoles. Note that epipole singularity disappears once multiple epipoles exist. (f,g,k,l,m) show an enlarge view of the depth image at the vicinity of the epipoles. The box shows the region where the epipoles are. For visibility purposes, different images are shown at different scales. For reference, coordinate rulers are attached to each image.



(a) (b)

Fig. 3. Blocks sequence. (a) one frame from the sequence. (b) The recovered shape (relative to the carpet). Brighter values correspond to taller points.

(a) (b)

Fig. 4. Flower-garden sequence. (a) one frame from the sequence. (b) The recovered shape (relative to the facade of the house). Brighter values correspond to points farther from the house.

(the "block" sequence from [11]). The reference plane is the carpet. Fig. 3.a shows one frame from the sequence. Fig. 3.b shows the recovered structure. Brighter grey levels correspond to taller points relative to the carpet. Note the fine structure of the toys on the carpet.

Fig. 4 shows an example of shape recovery for a sequence of five frames (part of the flower garden sequence). The reference plane is the house. Fig. 4.a shows the reference frame from the sequence. Fig. 4.b shows the recovered structure. Note the gradual change of depth in the field.

(a) (b)

Fig. 5. Stairs sequence. (a) one frame from the sequence. (b) The recovered shape (relative to the ground surface just in front of the building). Brighter values correspond to points above the ground surface, while darker values correspond to points below the ground surface.

Fig. 5 shows an example of shape recovery for a sequence of 5 frames. The reference plane is the flat region in front of the building. Fig. 5.a show one frame from the sequence. Fig. 5.b shows the recovered structure. The brightness reflects the magnitude of the structure parameter γ (brighter values correspond to scene points above the reference plane and darker values correspond to scene points below the reference plane). Note the fine structure of the stairs and the lamp-pole. The shape of the building wall is not fully recovered because of lack of texture in that region.

6 Conclusion

We presented an algorithm for estimating dense planar-parallax displacements from multiple uncalibrated views. The image displacements, the 3D structure, and the camera epipoles, are estimated *directly* from image brightness variations across multiple frames. This algorithm extends the two-frames plane+parallax estimation algorithm of [11,13] to multiple frames. The current algorithm relies on prior plane alignment. A natural extension of this algorithm would be to fold the homography estimation into the simultaneous estimation of image displacements, scene structure, and camera motion (as was done by [11] for *two* frames).

References

1. Bergen J. R., Anandan P., Hanna K. J., Hingorani R., *Hierarchical Model-Based Motion Estimation*, In European Conference on Computer Vision, pages 237-252, Santa Margarita Ligure, May 1992.
2. Criminisi C., Reid I., Zisserman Z., *Duality, Rigidity, and Planar Parallax*, In European Conference on Computer Vision, vol.II, 1998.
3. Hanna K. J., *Direct Multi-Resolution Estimation of Ego-Motion and Structure From Motion*, Workshop on Visual Motion, pp. 156-162, Princeton, NJ, Oct. 1991.
4. Hanna K. J. and Okamoto N. E., *Combining Stereo and Motion for Direct Estimation of Scene Structure*, International Conference on Computer Vision, 357-365, 1993.
5. Hartley R. I., *In Defense of the Eight-Point Algorithm*, In IEEE Trans. on Pattern Analysis and Machine Intelligence, 19(6):580-593, June 1997.
6. Irani M., Rousso B., and Peleg S., *Computing Occluding and Transparent Motions*, In International Journal of Computer Vision 12(1):5-16, Jan. 1994. (also in ECCV-92).
7. Irani M. and Anandan P., *Parallax Geometry of Pairs of Points for 3D Scene Analysis*, In European Conference on Computer Vision, A, pages 17-30, Cambridge, UK, April 1996.
8. Irani M., Rousso B. and peleg P., *Recovery of Ego-Motion Using Region Alignment*, In IEEE Trans. on Pattern Analysis and Machine Intelligence, 19(3), pp.268-272, March 1997. (also in CVPR-94).
9. Irani M., Anandan P., Weinshall D., *From Reference Frames to Reference Planes: Multi-View Parallax Geometry and Applications*, In European Conference on Computer Vision, vol.II, pp.829-845, 1998.
10. Irani M. and P. Anandan, *A Unified Approach to Moving Object Detection in 2D and 3D Scenes*, In IEEE Trans. on Pattern Analysis and Machine Intelligence, 20(6), pp. 577-589, June 1998.
11. Kumar R., Anandan P. and Hanna K., *Direct Recovery of shape From Multiple Views: a Parallax Based Approach*, International Conference on Pattern Recognition pp. 685-688, Oct. 1994.
12. Longuet-Higgins H.C., and Prazdny K., *The Interpretation of a Moving Retinal Image*, Proceedings of the Royal Society of London B, 208:385–397, 1980.
13. Sawhney H. S., *3D Geometry From Planar Parallax*, In IEEE Conference on Computer Vision and Pattern Recognition, pages 929-934, June 1994.
14. Shashua A. and Navab N., *Relative affine Structure: Theory and Application to 3D Reconstruction From Perspective Views*, In IEEE Conference on Computer Vision and Pattern Recognition, pages 483-489, 1994.
15. Stein G. P. and Shashua A., *Model-based Brightness constraints: On Direct Estimation of Structure and Motion*, In IEEE Conference on Computer Vision and Pattern Recognition, pages 400-406, 1997.
16. Szeliski R. and Kang S.B., *Direct Methods for Visual Scene Reconstruction*, In Workshop on Representations of Visual Scenes, 1995.
17. Torr P.H.S., *Geometric motion segmentation and model selection*, Proceedings of The Royal Society of London A, 356:1321–1340, 1998.
18. Zhang Z., *Determining the Epipolar Geometry and its Uncertainty: A Review*, IJCV, 27(2):161-195, 1997.

Discussion

Rick Szeliski: How do you initialize? – You have a sort of back and forth, two phase method for solving a bilinear problem. But when you have your first plane stabilized in a set of images, how do you guess the initial epipole or depth-map?

Michal Irani: We start with a zero depth map, and for the epipoles we try five different positions, one in the centre and one in each of the four quadrants. This does provide a good enough initialization — even in the case where the epipoles are at infinity we converge to the correct solution.

Bill Triggs: Just a comment on a comment you made about linearized methods being stabler than fully nonlinear ones. Assuming that the nonlinear method optimizes the statistically correct error model, its stability is by definition the true stability of the problem. If a linear method appears stabler it must be because it's either estimating a simplified model, or biased. So linear methods are not intrinsically stabler, they're just more often allowed to give wrong results.

Michal Irani: Well, linear algorithms are much simpler, and when their approximations are valid, they *don't* give the wrong results. So that's exactly the question — when *are* they valid, because when they are you would like to use them. The case I was talking about — the intermediate approximation where the global component of the homography is exact and only the local component is approximated — turns out to be valid in many, many cases. That's what we're checking right now. It has the potential to produce very simple algorithms without making any severe assumptions, whereas the original Longuet-Higgins approximation was very restrictive.

P. Anandan: I want to make a comment on Bill's comment. I think you see a similar thing about nonlinear methods being unstable in the work on encoding epipolar geometry. I recall Adiv's work, where at each iteration you normalize by the current depth to make the flow look more like the exact equation. So in some sense, by using weights based on the current estimate, you reduce the error introduced by the linear approximation during the iterative process. I'm not sure whether linear methods with varying weights should count as linear or nonlinear for stability. The same issues come up in correspondence based methods for structure-from-motion as well.

Michal Irani: I'm not sure whether this is what you meant Anandan, but generally when you solve a nonlinear iteration step you make some approximations that may not be correct. It's better to start with valid approximations than to start with bad ones. So, when you're assuming linear models, at least you know which approximations you're making.