Recovery of Ego-Motion Using Region Alignment

Michal Irani Benny Rousso Shmuel Peleg

Abstract—

A method for computing the 3D camera motion (the *ego-motion*) in a static scene is introduced, which is based on initially computing the 2D image motion of an image region. The computed dominant 2D parametric motion between two frames is used to register the images so that the corresponding image region appears perfectly aligned between the two registered frames. Such 2D parametric registration removes all effects of camera rotation, even for the misaligned image regions. The resulting *residual* parallax displacement field between the two region-aligned images is an *epipolar field* centered at the FOE (Focus-of-Expansion). The 3D camera translation is recovered from the epipolar field. The 3D camera rotation is recovered from the computed 3D translation and the detected 2D parametric motion.

The decomposition of image motion into a *2D parametric motion* and *residual epipolar parallax displacements* avoids many of the inherent ambiguities and instabilities associated with decomposing the image motion into its *rotational* and *translational* components, and hence robustifies ego-motion or 3D structure estimation.

I. Introduction

The motion observed in an image sequence can be caused by camera motion (ego-motion) and by motions of objects moving in the scene. In this paper we address the case of a camera moving in a static scene. Complete 3D motion estimation is difficult since the image motion at every pixel depends, in addition to the six parameters of the camera motion, on the depth at the corresponding scene point. To overcome this difficulty, additional constraints are usually added to the motion model or to the environment model.

3D motion is often estimated from the optical or normal flow derived between two frames [1], [12], [26], or from the correspondence of distinguished features (points, lines, contours) extracted from successive frames [27], [13], [8]. Both approaches depend on the accuracy of the feature detection, which can not always be assured. Methods for computing the ego-motion *directly* from image intensities were also suggested [11], [15].

Camera rotations and translations can induce similar image motions [2], [9] causing ambiguities in their interpretation. The problem of recovering the 3D camera motion from a flow field is therefore an ill-conditioned problem, since small errors in the 2D flow field usually result in large perturbations in the 3D motion [2]. At depth discontinuities, however, it is much easier to distinguish between the effects of camera rotations and camera translations, as the image motion of neighboring pixels at different depths will have similar rotational components, but different translational components. Motion parallax methods use this effect to obtain the 3D camera motion [22], [21], [8]. Other methods use motion parallax for shape representation and analysis [29], [7], [10]. However, accurate flow estimation at depth discontinuities is difficult.

In this paper a method is introduced for computing the ego-motion based on a decomposition of the image motion into a 2D parametric transformation and a residual parallax displacement field. This decomposition can be obtained more robustly, and avoids many of the inherent ambiguities and instabilities associated with decomposing a flow field into its rotational and translational components.

We introduce the following scheme: We use previously developed methods [17], [18], [4] to detect an image region with a 2D parametric motion between two image frames. The two frames are then registered according to the computed 2D parametric transformation. This step removes all effects of the camera rotation, even for the misaligned image regions. The residual parallax displacement field between the 2D region-aligned images is an *epipolar* field centered at the FOE of the camera. The FOE is then estimated from the epipolar field. When calibration information is provided, the 3D camera translation is recovered. The 3D rotation is estimated by solving a small set of *linear* equations, which depend on the computed 3D translation and the detected 2D parametric motion.

As opposed to other methods which use motion parallax for 3D estimation [22], [23], [21], [8], our method does *not* rely on parallax information at depth discontinuities (where flow computation is likely to be inaccurate). The residual displacements after 2D alignment provide a *denser* and more reliable parallax field.

The advantage of this technique is in its simplicity and in its robustness. No prior detection and matching are assumed, it requires solving only small sets of linear equations, and each computational step is stated as an overdetermined highly constrained problem which is numerically stable.

This paper is an updated version of our [19] paper. Since the paper was submitted to the journal, other publications with similar approaches have appeared [28], [20], [30], [16]. These techniques are often referred to by the name "plane-plus-parallax", since the estimated 2D parametric transformation frequently corresponds to the induced homography of a 3D planar surface in the scene.

II. Ego-Motion from 2D Image Motion

A. Basic Model and Notations

Let (X, Y, Z) denote the Cartesian coordinates of a scene point with respect to the camera (see Fig. 1), and let (x, y)

This research has been sponsored by the U.S. Office of Naval Research under Grant N00014-93-1-1202, R&T Project Code 4424341—01.

M. Irani is now with David Sarnoff Research Center, Princeton, NJ, U.S.A

B. Rousso, S. Peleg are with the Institute of Computer Science, The Hebrew University of Jerusalem, 91904 Jerusalem, ISRAEL

This research has been sponsored by ARPA through the U.S. Office of Naval Research under Grant N00014-93-1-1202, R&T Project Code 4424341—01, and by the Israeli Ministry of Science and Technology.

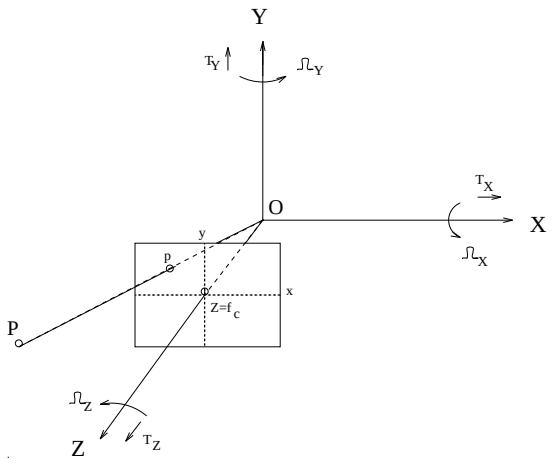

Fig. 1. The coordinate system.
The coordinate system (X, Y, Z) is attached to the camera, and the corresponding image coordinates (x, y) on the image plane are located at $Z = f_c$. A point $P = (X, Y, Z)^t$ in the world is projected onto an image point $p = (x, y)^t$. $T = (T_X, T_Y, T_Z)^t$ and $\Omega = (\Omega_X, \Omega_Y, \Omega_Z)^t$ represent the relative translation and rotation of the camera in the scene.

denote the corresponding coordinates in the image plane. The image plane is located at the focal length: $Z = f_c$. The perspective projection of a scene point $P = (X, Y, Z)^t$ on the image plane at a point $p = (x, y)^t$ is expressed by:

$$p = \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \frac{X}{Z} f_c \\ \frac{Y}{Z} f_c \end{bmatrix} \tag{1}$$

The camera motion has two components: a translation $T = (T_X, T_Y, T_Z)^t$ and a rotation $\Omega = (\Omega_X, \Omega_Y, \Omega_Z)^t$. Due to the camera motion the scene point $P = (X, Y, Z)^t$ appears to be moving relative to the camera with rotation $-\Omega$ and translation $-T$, and is therefore observed at new world coordinates $P' = (X', Y', Z')^t$, expressed by:

$$P' = M_{-\Omega} \cdot P - T, \tag{2}$$

where $M_{-\Omega}$ is the matrix corresponding to a rotation by $-\Omega$.

When the field of view is not very large and the camera motion has a relatively small rotation [1], the 2D displacement (u, v) of an image point (x, y) in the image plane can be expressed by [24], [4]:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} -f_c(\frac{T_X}{Z} + \Omega_Y) + x\frac{T_Z}{Z} + y\Omega_Z - x^2\frac{\Omega_Y}{f_c} + xy\frac{\Omega_X}{f_c} \\ -f_c(\frac{T_Y}{Z} - \Omega_X) - x\Omega_Z + y\frac{T_Z}{Z} - xy\frac{\Omega_Y}{f_c} + y^2\frac{\Omega_X}{f_c} \end{bmatrix} \tag{3}$$

All points (X, Y, Z) of a planar surface in the 3D scene satisfy a plane equation $Z = A + B \cdot X + C \cdot Y$, which can be expressed in terms of image coordinates by using Eq. (1) as:

$$\frac{1}{Z} = \alpha + \beta \cdot x + \gamma \cdot y. \tag{4}$$

In a similar manipulation to that in [1], substituting Eq. (4) in Eq. (3) yields the *2D quadratic transformation*:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} a + b \cdot x + c \cdot y + g \cdot x^2 + h \cdot xy \\ d + e \cdot x + f \cdot y + g \cdot xy + h \cdot y^2 \end{bmatrix} \tag{5}$$

where:

$$\begin{array}{ll} a = -f_c\alpha T_X - f_c\Omega_Y & e = -\Omega_Z - f_c\beta T_Y \\ b = \alpha T_Z - f_c\beta T_X & f = \alpha T_Z - f_c\gamma T_Y \\ c = \Omega_Z - f_c\gamma T_X & g = -\frac{\Omega_Y}{f_c} + \beta T_Z \\ d = -f_c\alpha T_Y + f_c\Omega_X & h = \frac{\Omega_X}{f_c} + \gamma T_Z \end{array} \tag{6}$$

Eq. (5), expressed by eight parameters (a, b, c, d, e, f, g, h), describes the 2D parametric image motion of a 3D planar surface. The quadratic transformation (5) is a good approximation to the 2D projective transformation assuming a small field of view and a small rotation.

Besides being an exact description of the instantaneous motion field of a planar surface, the quadratic transformation also describes well the 2D image motion of an *arbitrary* 3D scene undergoing camera rotations, zooms, and small camera translations. It also approximates well the 2D image motion under larger camera translations, when the overall 3D range (Z) to the scene is much greater than the variation of the range within the scene (ΔZ).

B. General Framework of the Algorithm

In this section we present a scheme which utilizes the robustness of the 2D motion computation for computing 3D motion between two consecutive frames:

1. A single image region with a 2D *parametric* image motion is automatically detected (Sec. III). As mentioned in Sec. II-A, this image region typically corresponds to a planar surface in the scene, or to a remote part of the scene.
2. The two frames are registered according to the computed 2D parametric motion of the detected image region. This image region alignment cancels the rotational component of the camera motion for the *entire* scene (Sec. II-C), and the FOE (focus-of-expansion) and the camera translation can now be computed from the residual epipolar displacement field between the two 2D *registered* frames (Sec. II-D).
3. The 3D rotation of the camera is now computed (Sec. II-E) from the 2D motion parameters of the detected image region and the 3D translation of the camera.

C. Cancelling Camera Rotation by 2D Region Alignment

At this stage we assume that a single image region with a parametric 2D image motion has been detected, and that the 2D image motion of that region has been computed. The automatic detection and computation of such a 2D transformation is briefly described in Sec. III.

Let $(u(x, y), v(x, y))$ denote the 2D image motion of the entire scene from frame f_1 to frame f_2, and let $(u_s(x, y), v_s(x, y))$ denote the 2D image motion of a single image region (the *detected* image region) between the two frames. Let S denote the 3D surface corresponding to the detected image region, with depths $Z_s(x, y)$. As mentioned in Sec. II-A, (u_s, v_s) can be expressed by a 2D parametric transformation (Eq. (5)) if S satisfies one of the following conditions: (i) S is a *planar* surface in the 3D scene, (ii)

S is an *arbitrary* 3D scene undergoing camera rotations, zooms, and small camera translations, or (iii) S is a portion of the scene that is distant enough from the camera (i.e., its overall 3D range (Z_s) is much greater than the range variations within it (ΔZ_s)).

Assuming the existence of such a surface S in the scene is not a severe restriction, as most indoor scenes contain a planar surface (e.g., walls, floor, pictures, windows, etc.), and in outdoor scenes the ground or any distant object can serve as such a surface. Note also that only the 2D motion parameters $(u_s(x,y), v_s(x,y))$ of the 3D surface S are estimated. The 3D structure or motion parameters of S are not estimated at this point.

Let f_1^R denote the frame obtained by warping the entire frame f_1 towards frame f_2 according to the 2D parametric transformation (u_s, v_s) *extended* to the entire frame. This warping will bring the image region corresponding to the detected surface S into perfect alignment between f_1^R and f_2. In the warping process, each pixel (x,y) in f_1 is displaced by $(u_s(x,y), v_s(x,y))$ to form f_1^R. Points that are not located on the parametric surface S (i.e., $Z(x,y) \neq Z_s(x,y)$) will *not* be in registration between f_1^R and f_2. We will now show that the residual 2D image displacements between the two registered frames (f_1^R and f_2) forms an epipolar field centered at the FOE, i.e., affected only by the camera translation T.

Let $P_1 = (X_1, Y_1, Z_1)^t$ denote the 3D scene point projected onto $p_1 = (x_1, y_1)^t$ in f_1. According to Eq. (1): $P_1 = (x_1 \frac{Z_1}{f_c}, y_1 \frac{Z_1}{f_c}, Z_1)^t$. Due to the camera motion (Ω, T) from frame f_1 to frame f_2, the point P_1 will be observed in frame f_2 at $p_2 = (x_2, y_2)^t$, which corresponds to the 3D scene point $P_2 = (X_2, Y_2, Z_2)^t$. According to Eq. (2):

$$P_2 = M_{-\Omega} \cdot P_1 - T. \tag{7}$$

The warping of f_1 by (u_s, v_s) to form f_1^R is equivalent to applying the camera motion (Ω, T) to the 3D points as though they are all located on the surface S (i.e., with depths $Z_s(x,y)$). Let P_s denote the 3D point on the surface S which corresponds to the pixel (x,y) with depth $Z_s(x,y)$. Then:

$$P_s = \begin{bmatrix} x_1 \frac{Z_s}{f_c} \\ y_1 \frac{Z_s}{f_c} \\ Z_s \end{bmatrix} = \frac{Z_s}{Z_1} \cdot \begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \end{bmatrix} = \frac{Z_s}{Z_1} \cdot P_1. \tag{8}$$

After the image warping, P_s is observed in f_1^R at $p^R = (x^R, y^R)^t$, which corresponds to a 3D scene point P^R. Therefore, according to Eq. (2) and Eq. (8): $P^R = M_{-\Omega} \cdot P_s - T = \frac{Z_s}{Z_1} \cdot M_{-\Omega} \cdot P_1 - T$, and therefore:

$$P_1 = \frac{Z_1}{Z_s} \cdot {M_{-\Omega}}^{-1} \cdot (P^R + T). \tag{9}$$

By substituting (9) in (7), P^R can be expressed as:

$$P^R = \frac{Z_s}{Z_1} \cdot P_2 + (1 - \frac{Z_s}{Z_1}) \cdot (-T). \tag{10}$$

Eq. (10) shows that P^R is independent of the camera rotation Ω. Moreover, P^R is on the straight line passing through P_2 and $-T$. Therefore, the projection of P^R on the image plane (p^R) is on the straight line passing through the projection of P_2 (i.e., p_2) and the projection of $-T$ (i.e., the FOE). This means that p^R is found on the radial line emerging from the FOE towards p_2. In other words, the residual image displacements between the registered frames f_1^R and f_2 (i.e., $p^R - p_2$) form an epipolar field centered at the FOE. (Note, that the *magnitudes* of the residual displacements depend on the scene structure, $\frac{Z_s}{Z_1}$, however their *directions* do not).

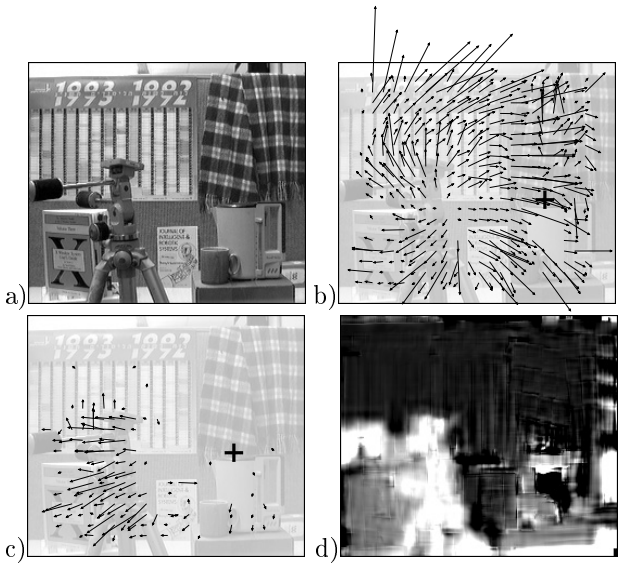

Fig. 2. The effect of 2D region alignment. The real FOE is marked by +.
a) One of the frames.
b) The optical flow between two adjacent frames (before registration), overlayed on Fig. 2.a (for display purposes only).
c) The optical flow after (automatic) 2D alignment of the wall. The flow is induced by pure camera translation (after the camera rotation was canceled), and points now to the correct FOE.
d) The computed depth map. Bright regions correspond to close objects.

In Fig. 2, the optical flow is displayed before and after registration of two frames according to the computed 2D motion parameters of the image region (which happened to correspond in this case with the wall at the back of the scene). The optical flow is given for *display* purposes only, and was *not* used in the registration. After registration, the rotational component of the optical flow was canceled for the *entire* scene, and all flow vectors point towards the real FOE (Fig. 2.c). Before registration (Fig. 2.b) the FOE mistakenly appears to be located elsewhere (in the middle of the frame). This is due to the ambiguity caused by the rotation around the Y-axis, which visually appears as a translation along the X-axis. This ambiguity is resolved by the 2D registration.

In Section III we briefly explain why the interpretation of image motion in terms of a *2D parametric transformation* and a residual (epipolar) *parallax displacement field* is less ambiguous and numerically more stable than the interpretation the image motion in terms of its induced *rotational* and *translational* image displacements.

D. Computing Camera Translation

Once the rotation is canceled by the 2D alignment of the detected image region, the ambiguity between image motion induced by 3D rotation and that induced by 3D translation no longer exists (see Sec. II-C). Having cancelled effects of camera rotation, the residual displacement field is directed towards, or away from, the FOE. The computation of the FOE therefore becomes overdetermined and numerically stable, as there are only two unknowns to the problem: the 2D coordinates of the center of the epipolar field (i.e, FOE) in the image plane.

To locate the FOE, the optical flow between the 2D *registered* frames is computed, and the FOE is located using a search method similar to that described in [22]. Candidates for the FOE are sampled over a half sphere and projected onto the image plane. For each such candidate, a global error measure is computed from local deviations of the flow field from the radial lines emerging from the candidate FOE. The search process is repeated by refining the sampling (on the sphere) around good FOE candidates. After a few refinement iterations, the FOE is taken to be the candidate with the smallest error.

Since the problem of locating the FOE in a *purely translational* (epipolar) flow field is a highly overdetermined problem, the computed flow field need *not* be accurate. This is opposed to most methods which try to compute the ego-motion from the flow field, and require an *accurate* flow field in order to resolve the rotation-translation ambiguity [2]. Given the FOE and camera calibration information, the 3D camera translation is recovered.

E. Computing Camera Rotation

Let (a, b, c, d, e, f, g, h) be the 2D motion parameters of a an image region as expressed by Eq. (5). Given these 2D motion parameters and the 3D translation parameters of the camera (T_X, T_Y, T_Z), the 3D rotation parameters of the camera $(\Omega_X, \Omega_Y, \Omega_Z)$ (as well as the surface parameters when a plane (α, β, γ)) can be obtained by solving Eq. (6), which is a set of *eight* linear equations in *six unknowns*.

From our experience, the parameters g and h in the quadratic transformation, computed by the method described in Sec. III, are not as reliable as the other six parameters (a, b, c, d, e, f), as g and h are second order terms in Eq. (5). Therefore, whenever possible (when the set of Eq. (6) is numerically overdetermined), we avoid using the last two equations (for g and h), and use only the first six. This yields more accurate results.

As a matter of fact, the only case in which all eight equations of (6) must be used to recover the camera rotation is the case when the camera translation is parallel to the image plane (i.e., $\vec{T} \neq 0$ and $T_Z = 0$). In that case, only Ω_Z can be recovered purely from the first six equations of (6) (i.e., using only the reliable parameters a, b, c, d, e, f, and disregarding the unreliable ones, g and h). In order to recover the two other rotation parameters, Ω_X and Ω_Y, the second order terms g and h must be used. Therefore, in the case of $\vec{T} \neq 0$ and $T_Z = 0$, the translation parameters (T_X, T_Y, T_Z) and one rotation parameter, Ω_Z (the rotation around the optical axis) can be recovered accurately, while the other two rotation parameters, Ω_X and Ω_Y, can only be approximated. In other configurations of camera motion the camera rotation can be reliably recovered.

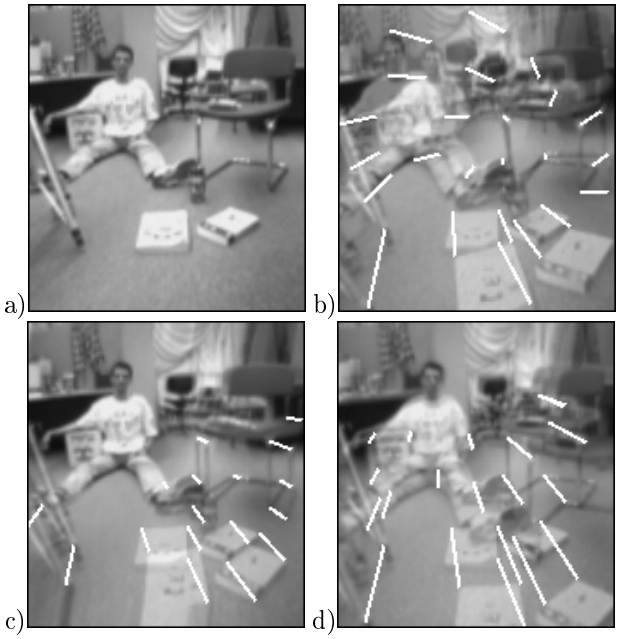

Fig. 3. Camera Stabilization.
a) One of the frames in the sequence.
b) The average of two frames, having both rotation and translation. The white lines display the image motion.
c) The average of the two frames after (automatic) 2D alignment of the shirt. Only effects of camera translation remain.
d) The average of the two frames after recovering the ego-motion, and canceling the camera rotation. This results in a 3D-stabilized pair of images (i.e., no camera jitter).

F. Experimental Results

The camera motion between the two frames in Fig. 2 was: $(T_X, T_Y, T_Z) = (1.7_{cm}, 0.4_{cm}, 12_{cm})$ and $(\Omega_X, \Omega_Y, \Omega_Z) = (0°, -1.8°, -3°)$. The computation of the 3D motion parameters of the camera (after calibrating T_Z to 12_{cm}, as $\vec{T}$ can only be determined up to a scale factor [**REF?**]), yielded: $(T_X, T_Y, T_Z) = (1.68_{cm}, 0.16_{cm}, 12_{cm})$ and $(\Omega_X, \Omega_Y, \Omega_Z) = (-0.05°, -1.7°, -3.25°)$.

Once the 3D motion parameters of the camera are computed, the 3D scene structure can be reconstructed using a scheme similar to that suggested in [11]. Correspondences between small image patches (currently 5×5 pixels) are computed only along the radial lines emerging from the FOE (taking the rotations into account). The depth map is computed from the magnitude of these displacements. In Fig. 2.d, the computed inverse depth map of the scene $(\frac{1}{Z(x,y)})$ is displayed. Similar approaches to 3D shape recovery have since been suggested by [28], [20], [30], [16].

Fig. 3 shows an example where the ego-motion estimation was used to electronically stabilize (i.e., remove camera jitter) from a sequence obtained by a hand held camera.

III. Computing a 2D Parametric Motion

We use previously developed methods [17], [18], [4] in order to detect a 2D parametric transformation of an image region. In this section we briefly describe these methods. For more details see [18]. Other for computing a 2D parametric region motion have also been suggested and can be used [6], [3] equally.

Let R be an image region that has a single 2D parametric transformation $(u(x,y), v(x,y))$ between two frames, $I(x,y,t)$ and $I(x,y,t+1)$. (u,v) is a quadratic transformation expressed by eight parameters $\mathbf{q} = (a,b,c,d,e,f,g,h)$ (see Eq. (5)). To solve for these parameters, the following SSD error measure is minimized:

$$\begin{aligned} Err^{(t)}(\mathbf{q}) &= \sum_{(x,y)\in R} (I(x,y,t) - I(x+u, y+v, t+1))^2 \\ &\approx \sum_{(x,y)\in R} (uI_x + vI_y + I_t)^2. \end{aligned} \quad (11)$$

The objective function Err is minimized via the Gauss-Newton optimization technique. Let $\mathbf{q}_i$ denote the current estimate of the quadratic parameters. After warping the inspection image $(I(x,y,t+1))$ towards the reference image $(I(x,y,t))$ by applying the parametric transformation $\mathbf{q}_i$ to it, an incremental estimate $\delta\mathbf{q}$ can be determined. After iterating certain number of times within a pyramid level, the process continues at the next finer level [5], [4], [18].

When the above technique is applied to a region R, the reference and the inspection images are registered so that the desired image region is aligned. However, a region of support R of an image segment with a single 2D parametric motion is not known a-priori. To allow for automatic detection and locking onto a single 2D parametric image motion, a robust version of the above technique is applied [18]. The robust version of the algorithms introduces two additional mechanisms to the above described technique:

1. *Outlier Rejection:* A truncated function of the local misalignments at each iteration provides weights for the weighted-least-squares regression process of the next iteration.
2. *Progressive Model Complexity:* The complexity of the 2D parametric motion model used in the regression process is gradually increased with the progression of the iterative process and the outlier rejection. Initially a simple 2D translation (2 parameters) is used, and is gradually refined to a 2D affine (6 parameters) and further to a 2D quadratic (8 parameters). The progressive complexity scheme focuses first on the most stable constant terms (a and d), then further refines them along with the less stable linear terms (b,c,e,f), and finally refines all parameters along with the least stable quadratic terms (h and g). This provides the algorithm with increased stability and locking capabilities, and prevents it from converging into local minima.

For more details see [18]. Other robust methods for locking onto a dominant 2D parametric transformation have also been suggested [6], [3].

We would like to stress a few important points:

- Assuming the existence of a significant image region with a single 2D parametric transformation is realistic in a wide range of scenarios: Indoor scenes contain planar surfaces (e.g., walls, floor, pictures, windows, etc.), and in outdoor scenes the ground, a boulevard of trees, or any distant large object will induce a 2D parametric image motion.
- The interpretation of image motion in terms of a *2D parametric transformation* and a residual (epipolar) *parallax displacement field* (as presented in Section II-C) is less ambiguous and numerically more stable than the interpretation the image motion in terms of its induced *rotational* and *translational* image displacements. This is due to the increased robustness and accuracy of 2D parametric motion estimation in comparison to optical flow estimation.
  When flow is computed, the support region for each estimated flow vector is very small [14] (typically 5×5 or 7×7 windows), as larger windows will violate the simple flow equations. Such small windows, however, frequently suffer from aperture effects, hence accurate *non-constrained* flow estimation is known to be an ill-conditioned problem. Noisy flow, however, introduces ambiguities in the interpretation of flow in terms of its induced rotational and translational components [2]. Constraining the local flow estimation is difficult, as flow vectors depend on the unknown depth Z of the corresponding scene point.
  A 2D parametric transformation (e.g., Eq. (5)), however, is expressed in terms of few parameters (e.g., 8), yet has a substantially larger region of support in the image plane. Therefore, the "flow" estimation of a 2D parametric motion is highly constrained and well conditioned. For example, textured areas within the region of support provide accurate image motion estimation for the non-textured area in that region. Using one of the robust estimatio techniques [18], [6], [3] provides the ability to accurately estimate a 2D parametric image motion of an image region. 2D alignment using the computed 2D parametric transformation was shown (see Section II-C) to disambiguate camera rotation and translation. Furthermore, the *residual* epipolar flow field can be also estimated more accurately than general flow, as it is constrained to lie on an epipolar field.

IV. Concluding Remarks

A method for computing ego-motion in static scenes was introduced. At first, an image region with a dominant 2D parametric transformation is detected, and its 2D motion parameters between successive frames are computed. The 2D transformation is then used for image warping, which cancels the rotational component of the 3D camera motion for the *entire* image, and reduces the problem to that of a pure 3D translation. The FOE and the 3D camera translation are computed from the 2D registered frames, and then the 3D rotation is computed by solving a small set of linear

equations.

The advantage of the presented technique is in its simplicity, and in the robustness and stability of each computational step. The interpretation of the image motion in terms of a 2D parametric transformation and a residual (epipolar) parallax displacement field, can be obtained more robustly, and avoids many of the inherent ambiguities and instabilities associated with decomposing a flow field into its rotational and translational components. Hence, the proposed method provides increased numerical stability and computational efficiency. There are no severe restrictions on the camera motion or on the 3D structure of the environment. Most steps use only image intensities, and the optical flow is used only for extracting the FOE in the case of pure epipolar field, which is an overdetermined problem and hence does not require accurate optical flow. The inherent problems associated with optical flow or with feature matching are therefore avoided.

Acknowledgment

The authors wish to thank R. Kumar for pointing out an error in the first version of this paper.

References

[1] G. Adiv. Determining three-dimensional motion and structure from optical flow generated by several moving objects. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 7(4):384–401, July 1985.

[2] G. Adiv. Inherent ambiguities in recovering 3D motion and structure from a noisy flow field. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 11:477–489, May 1989.

[3] S. Ayer and H. Sawhney. Layered representation of motion video using robust maximum-likelihood estimation of mixture models and mdl encoding. In *International Conference on Computer Vision*, pages 777–784, Cambridge, MA, June 1995.

[4] J.R. Bergen, P. Anandan, K.J. Hanna, and R. Hingorani. Hierarchical model-based motion estimation. In *European Conference on Computer Vision*, pages 237–252, Santa Margarita Ligure, May 1992.

[5] J.R. Bergen, P.J. Burt, R. Hingorani, and S. Peleg. A three-frame algorithm for estimating two-component image motion. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 14:886–895, September 1992.

[6] M.J. Black and P. Anandan. The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields. *CVGIP: Image Understanding*, 63(1):75–104, January 1996.

[7] S. Carlsson and J.O. Eklundh. Object detection using model based prediction and motion parallax. In *European Conference on Computer Vision*, pages 297–306, April 1990.

[8] R. Chipolla, Y. Okamoto, and Y. Kuno. Robust structure from motion using motion paralax. In *International Conference on Computer Vision*, pages 374–382, Berlin, May 1993.

[9] K. Daniilidis and H.-H. Nagel. The coupling of rotation and translation in motion estimation of planar surfaces. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 188–193, June 1993.

[10] W. Enkelmann. Obstacle detection by evaluation of optical flow fields from image sequences. In O. Faugeras, editor, *European Conference on Computer Vision*, pages 134–138, 1990.

[11] K. Hanna. Direct multi-resolution estimation of ego-motion and structure from motion. In *IEEE Workshop on Visual Motion*, pages 156–162, Princeton, NJ, October 1991.

[12] D.J. Heeger and A. Jepson. Simple method for computing 3d motion and depth. In *International Conference on Computer Vision*, pages 96–100, 1990.

[13] B.K.P. Horn. Relative orientation. *International Journal of Computer Vision*, 4(1):58–78, June 1990.

[14] B.K.P. Horn and B.G. Schunck. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981.

[15] B.K.P. Horn and E.J. Weldon. Direct methods for recovering motion. *International Journal of Computer Vision*, 2(1):51–76, June 1988.

[16] M. Irani and P. Anandan. Parallax geometry of pairs of points for 3D scene analysis. In *European Conference on Computer Vision*, pages I:17–30, Cambridge, UK, April 1996.

[17] M. Irani, B. Rousso, and S. Peleg. Detecting and tracking multiple moving objects using temporal integration. In *European Conference on Computer Vision*, pages 282–287, Santa Margarita Ligure, May 1992.

[18] M. Irani, B. Rousso, and S. Peleg. Computing occluding and transparent motions. *International Journal of Computer Vision*, 12(1):5–16, January 1994.

[19] M. Irani, B. Rousso, and S. Peleg. Recovery of ego-motion using image stabilization. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 454–460, Seattle, June 1994.

[20] R. Kumar, P. Anandan, and K. Hanna. Direct recovery of shape from multiple views: a parallax based approach. In *Proc 12th ICPR*, 1994.

[21] J. Lawn and R. Chipolla. Epipole estimation using affine motion parallax. Technical Report CUED/F-INFENG/TR-138, Cambridge, July 1993.

[22] D.T. Lawton and J.H. Rieger. The use of difference fields in processing sensor motion. In *ARPA IU Workshop*, pages 78–83, June 1983.

[23] C.H. Lee. Structure and motion from two perspective views via planar patch. In *International Conference on Computer Vision*, pages 158–164, 1988.

[24] H.C. Longuet-Higgins. Visual ambiguity of a moving plane. *Proceedings of The Royal Society of London B*, 223:165–175, 1984.

[25] F. Meyer and P. Bouthemy. Estimation of time-to-collision maps from first order motion models and normal flows. In *International Conference on Pattern Recognition*, pages 78–82, The Hague, 1992.

[26] S. Negahdaripour and S. Lee. Motion recovery from image sequences using first-order optical flow information. In *IEEE Workshop on Visual Motion*, pages 132–139, Princeton, NJ, October 1991.

[27] F. Lustman O.D. Faugeras and G. Toscani. Motion and structure from motion from point and line matching. In *Proc. 1st International Conference on Computer Vision*, pages 25–34, London, 1987.

[28] Harpreet Sawhney. 3d geometry from planar parallax. In *IEEE Conference on Computer Vision and Pattern Recognition*, June 1994.

[29] A. Shashua. Projective depth: a geometric invariant for 3d reconstruction from two perspective/orthographic views and for visual recognition. In *International Conference on Computer Vision*, pages 583–590, Berlin, May 1993.

[30] A. Shashua and N. Navab. Relative affine structure: Theory and application to 3d reconstruction from perspective views. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 483–489, Seattle, Wa., June 1994.