Robust Recovery of Ego-Motion*

Michal Irani Benny Rousso Shmuel Peleg

Institute of Computer Science
The Hebrew University of Jerusalem
91904 Jerusalem, ISRAEL
Email: {michalb, rousso, peleg}@cs.huji.ac.il

Abstract

A direct method is introduced for computing the camera motion (the *ego-motion*) in a static scene. The method is based on detecting two planar surfaces in the scene and computing their 2D motion in the image plane. *Theoretically*, the 3D camera motion can sometimes be computed from the 2D image motion of a single planar surface. In practice, however, such computation is complicated, as it requires solving a set of high-order nonlinear equations, and is ill-conditioned. When using the 2D motion of *two* planar surfaces, however, the 3D camera motion can be computed by simple solutions of linear equations.

The presented method uses image intensities, and the inherent problems of computing the optical flow and of feature matching are avoided. This result is a step forward in our effort to robustly compute motion and 3D structure from image sequences, using image intensities and not assuming any prior detection or correspondence. There are no severe restrictions on the ego-motion or on the structure of the environment.

*This research was supported by grants from the Austrian Friends of the Hebrew University and the Israel Academy of Sciences.

1 Introduction

The motion observed in an image sequence can be caused by camera motion (ego-motion) or by motions of independently moving objects 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 it is ill-conditioned, due to the very large number of unknowns (the depth at each scene point and the 6 motion parameters) in comparison to the amount of available input data. To overcome this difficulty, additional constraints are usually added to the motion model or to the environment model, e.g. introducing a regularization term [HS81], assuming a limited model of the world [Adi85], restricting the range of possible motions [HW88], or assuming some type of temporal motion constancy over a longer sequence [DF90].

3D motion is often estimated from the optical or normal flow field derived between two frames [Adi85, LR83, GU91, JH91, Sun91, NL91, HA91, TS91, AD92], or from the correspondence of distinguished features (points, lines, contours) previously extracted from the two frames [OFT87, Hor90]. Both approaches usually suffer from numerical instabilities in case of noisy data. Feature matching is also very sensitive to occlusions.

Increasing the temporal region to more than two frames improves the accuracy of the computed motion. Methods for estimating local image velocities with large temporal regions have been introduced using a combined spatio-temporal analysis [FJ90, Hee88, SM90]. These methods assume motion constancy in the temporal regions, i.e. motion should remain uniform in the analyzed sequence. In feature based methods, features are tracked over a larger time interval using recursive temporal filtering [DF90]. The issue of extracting good features and overcoming occlusions still remains.

Methods for computing the ego-motion *directly* from image intensities were also suggested [Han91, HW88, Taa92]. Horn and Weldon [HW88] deal only with restricted cases (pure translation,pure rotation, or general motion with known depth). Hanna [Han91] suggests an iterative method for the general case, but relies strongly on a reasonably accurate initial guess. Taalebinezhaad [Taa92] has a complete mathematical formulation, but relies on computations at a single image point, which is very unreliable.

In this work a direct method for computing the ego-motion from image intensities is introduced. It is a coarse-to-fine technique in the sense that at first only robust 2D data is extracted, and only later is it fused to obtain the relevant 3D information.

We use existing methods [IRP92] to extract two planar surfaces in the scene with their 2D projective transformation. Theoretically it is enough to locate one planar surface in a static scene and use its 2D projective motion parameters in order to compute the 3D motion parameters of the camera. However, this requires solving a set of high-order nonlinear equations. To avoid this difficulty, we use *two* different planar surfaces in the scene. The difference between their 2D projective transformations can then be used to extract the translation of the camera by solving a set of *linear* equations. Once the translation is known, it can be used together with one of the 2D projective transformations to compute the rotation of the camera by solving another set of linear equations.

The advantage of this technique is in its simplicity (solving small sets of linear equations), and in the robustness and stability of the 2D algorithm. The choice of an initial 2D motion model enables efficient motion computation, and is numerically stable because the 2D problem is overdetermined, as opposed to directly approach the problem of 3D motion computation which may become underdetermined in its general form.

This method does not restrict the camera motion or the environment structure. It only requires the existence of two planar surfaces in the scene (but the scene needs *not* be piecewise linear, as in [Adi85]). Most indoor scenes have the required two planar surfaces, and in outdoor scenes the ground can serve as one planar surface, and only a single additional planar surface is required.

In Section 2 the technique for computing the ego-motion is described, given two planar surfaces with their 2D projective transformations.

In Section 3 the method for extracting the two planar surfaces and their 2D motion parameters is described.

2 Ego-Motion from Two 2D Motions

In this section we describe the technique for computing the ego-motion given two planar surfaces with their 2D projective transformations in the image plane. In Section 3 the method for extracting the two planar surfaces and their 2D motion is described.

2.1 Basic Model and Notations

The notations describing the 3D motion of the camera and the corresponding 2D motion of the planar surfaces in the image plane are introduced in this section. Let (X, Y, Z) denote the Cartesian coordinates of a scene point with respect to the camera (see Figure 1), and let (x, y) 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 (X, Y, Z) on the image plane at a point (x, y) is expressed by:

$$x = \frac{X}{Z} f_c \quad , \qquad y = \frac{Y}{Z} f_c \tag{1}$$

The motion of the camera has two components: a translation (T_X, T_Y, T_Z) and a rotation $(\Omega_X, \Omega_Y, \Omega_Z)$.

When the field of view is not very large and the rotation is relatively small [Adi85], a 3D motion of the camera between two image frames creates a 2D displacement (u, v) of an image point (x, y) in the image plane, which can be expressed by [BAHH92]:

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

Assuming (X, Y, Z) lies on a planar surface in the scene, represented by the equation:

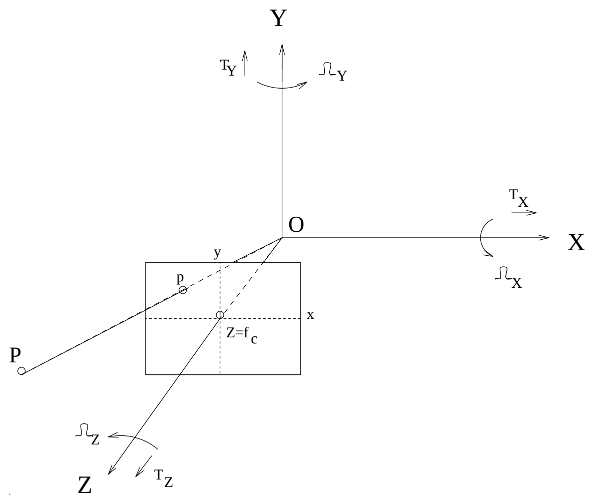

Figure 1: The coordinate system.
The coordinate system (X, Y, Z) is attached to the camera, and the corresponding image coordinates (x, y) on the image plane are located at $Z = f_c$. A point $P = (X, Y, Z)$ in the world is projected onto an image point $p = (x, y)$. $T = (T_X, T_Y, T_Z)$ and $\Omega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

$$Z = A + B \cdot X + C \cdot Y,$$

then, dividing by $A \cdot Z$ and rearranging the terms, we get:

$$\frac{1}{Z} = \frac{1}{A} - \frac{B}{A}\frac{X}{Z} - \frac{C}{A}\frac{Y}{Z}$$

which can be rewritten using Equation (1) as:

$$\frac{1}{Z} = \alpha + \beta \cdot x + \gamma \cdot y \tag{3}$$

where: $\alpha = \frac{1}{A}$, $\quad \beta = -\frac{B}{f_c A}$, $\quad \gamma = -\frac{C}{f_c A}$.

In a similar manipulation to that in [Adi85], substituting (3) in (2) yields:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} a & + & b \cdot x & + & c \cdot y & + & g \cdot x^2 & + & h \cdot xy \\ d & + & e \cdot x & + & f \cdot y & + & g \cdot xy & + & h \cdot y^2 \end{bmatrix} \tag{4}$$

where:

$$\left\{\begin{array}{lclcl} a & = & -f_c\alpha T_X & - & f_c\Omega_Y \\ b & = & \alpha T_Z & - & f_c\beta T_X \\ c & = & \Omega_Z & - & f_c\gamma T_X \\ d & = & -f_c\alpha T_Y & + & f_c\Omega_X \\ e & = & -\Omega_Z & - & f_c\beta T_Y \\ f & = & \alpha T_Z & - & f_c\gamma T_Y \\ g & = & -\frac{\Omega_Y}{f_c} & + & \beta T_Z \\ h & = & \frac{\Omega_X}{f_c} & + & \gamma T_Z \end{array}\right. \qquad (5)$$

Equation (4) describes the 2D motion in the image plane, expressed by eight parameters (a, b, c, d, e, f, g, h), which corresponds to a general 3D motion of a planar surface in the scene assuming a small field of view and little rotation. Actually, Equation (4) is a good approximation to the 2D projective transformation of that image segment between two frames, as long as the change in angle between the planar surface and the camera in this time interval was negligible.

We call this 2D transformation (Equation (4)) a *pseudo 2D projective* transformation. A method for computing the pseudo projective parameters of an image segment is described in Sections 3.2 and 3.3.

2.2 Computing the Ego-Motion

Theoretically it is enough to locate one planar surface in a static scene and use its pseudo 2D projective parameters (Equations (4) and (5)) in order to compute the 3D motion parameters of the camera. However, approaching the problem in this way is numerically unstable, in many cases underdetermined, and requires solving a set of high-order nonlinear equations. To avoid these difficulties, we use *two* different planar surfaces in the scene. The difference between their pseudo 2D projective transformations can then be used to extract the translation of the camera by solving a set of *linear* equations. Once the translation is computed, it can be used together with one of the pseudo 2D projective transformations to compute the rotation of the camera by solving yet another set of linear equations. This method is described in details in this section.

Let $(a_1, b_1, c_1, d_1, e_1, f_1, g_1, h_1)$ and $(a_2, b_2, c_2, d_2, e_2, f_2, g_2, h_2)$ be the pseudo 2D projective parameters of two different planar surfaces in the scene, computed as described in Sections 3.2 and 3.3. Using Equation (5) we get:

$$
\begin{cases}
a_1 - a_2 &= -f_c(\alpha_1 - \alpha_2)T_X \\
b_1 - b_2 &= (\alpha_1 - \alpha_2)T_Z - f_c(\beta_1 - \beta_2)T_X \\
c_1 - c_2 &= -f_c(\gamma_1 - \gamma_2)T_X \\
d_1 - d_2 &= -f_c(\alpha_1 - \alpha_2)T_Y \\
e_1 - e_2 &= -f_c(\beta_1 - \beta_2)T_Y \\
f_1 - f_2 &= (\alpha_1 - \alpha_2)T_Z - f_c(\gamma_1 - \gamma_2)T_Y \\
g_1 - g_2 &= (\beta_1 - \beta_2)T_Z \\
h_1 - h_2 &= (\gamma_1 - \gamma_2)T_Z
\end{cases}
\tag{6}
$$

where, as mentioned before, (T_X, T_Y, T_Z) are the camera translation parameters, f_c is the camera focal length, and α, β, γ are the plane parameters.

Note that by taking a difference, the rotation parameters $(\Omega_X, \Omega_Y, \Omega_Z)$ were discarded in Equation (6), leaving only the translation parameters (T_X, T_Y, T_Z). This can be easily understood from Equation (2), as the contribution of the camera *rotation* to the displacement of an image point is *independent* of the depth Z of the scene point, while the contribution of the camera *translation* to the displacement of an image point *does* depend on the depth Z of the scene point.

Following is an analysis of the ego-motion computation case by case, addressing all possible motion configurations. For each case there is also an experimental example. More details regarding the experimental results can be found in the Appendix.

Case 1: Camera motion without Translation (Pure Rotation).

In this case $T_X = T_Y = T_Z = 0$, therefore: $a_1 = a_2, b_1 = b_2, c_1 = c_2, d_1 = d_2, e_1 = e_2, f_1 = f_2, g_1 = g_2, h_1 = h_2$. The 2D detection algorithm described in Section 3 will identify the entire image as a projected region of a *single* planar surface. In this special case, however, the rotation of the camera can be computed from a single pseudo 2D projective transformation $(a_1, b_1, c_1, d_1, e_1, f_1, g_1, h_1)$, as follows. Setting $T_X = T_Y = T_Z = 0$ in Equation (5) yields

$$
\begin{cases}
a_1 &= -f_c\Omega_Y \\
b_1 &= 0 \\
c_1 &= \Omega_Z \\
d_1 &= f_c\Omega_X \\
e_1 &= -\Omega_Z \\
f_1 &= 0 \\
g_1 &= -\frac{\Omega_Y}{f_c} \\
h_1 &= \frac{\Omega_X}{f_c}
\end{cases}
\tag{7}
$$

The ego-motion in this case is therefore:

$$T_X = T_Y = T_Z = 0 \tag{8}$$

$$\Omega_X = \frac{d_1}{f_c}, \quad \Omega_Y = -\frac{a_1}{f_c}, \quad \Omega_Z = \frac{c_1 - e_1}{2} \tag{9}$$

a)  b)  c) 

Figure 2: Pure rotation (Case 1).

a) The first image.
b) The second image, taken after rotating the camera by 2° around its Y-axis.
c) The difference image taken after registering the two images according to the computed pseudo 2D projective motion. Dark regions correspond to large intensity differences. The computed 2D motion corresponds to the *entire* image (except for new areas in the image boundary), and the difference image is actually nulled almost everywhere.

The case of pure rotation can be detected by observing that $b_1 \approx f_1 \approx 0$ and $c_1 \approx -e_1$.

Remark: The focal length f_c can theoretically be recovered from g_1 and h_1. However, from our experience, the parameters g and h in the pseudo 2D projective transformation, computed by the method described in Section 3, are not as reliable as the other six parameters (a, b, c, d, e, f), as g and h are second order terms in Equation (4). We therefore avoid using those two parameters throughout this work, and use the known focal length of the camera. However, developing an improved method for obtaining more reliable pseudo 2D projective parameters would enable us to obtain additional information.

Experiment 1: In our experiments a $16mm$ camera was used, where the focal length was found to be approximately 877 pixels (i.e., $f_c = 877$). The scene in Figure 2 contains three different planar surfaces: two walls meeting in a corner, and a row of books whose backs form a single plane. The camera motion between Figure 2.a and Figure 2.b was: $(T_X, T_Y, T_Z) = (0, 0, 0)$ and $(\Omega_X, \Omega_Y, \Omega_Z) = (0°, 2°, 0°)$. Using Equations (8) and (9) with the the computed 2D motion parameters yields the following ego-motion: $(T_X, T_Y, T_Z) = (0, 0, 0)$ and $(\Omega_X, \Omega_Y, \Omega_Z) = (0.05°, 1.93°, 0.1°)$.

Case 2: Camera motion with Translation ($T_Z \neq 0$).

Without loss of generality it can be assumed that $T_Z = f_c$, as there is a degree of freedom in the magnitude of translation and depth (this can be easily observed in Equation (2), as the ratios $\frac{T_X}{Z}$, $\frac{T_Y}{Z}$, and $\frac{T_Z}{Z}$ remain unchanged if both (T_X, T_Y, T_Z) and Z are multiplied by the same scaling factor). Substituting $T_Z = f_c$ in the first *six* equations of (6) (see remark at the end of "Case 1"), we get:

$$\left\{\begin{array}{rclcl} a_1 - a_2 & = & -f_c(\alpha_1 - \alpha_2)T_X & & \\ b_1 - b_2 & = & f_c(\alpha_1 - \alpha_2) & - & f_c(\beta_1 - \beta_2)T_X \\ c_1 - c_2 & = & -f_c(\gamma_1 - \gamma_2)T_X & & \\ d_1 - d_2 & = & -f_c(\alpha_1 - \alpha_2)T_Y & & \\ e_1 - e_2 & = & -f_c(\beta_1 - \beta_2)T_Y & & \\ f_1 - f_2 & = & f_c(\alpha_1 - \alpha_2) & - & f_c(\gamma_1 - \gamma_2)T_Y \end{array}\right. \quad \begin{array}{r} (10.1) \\ (10.2) \\ (10.3) \\ (10.4) \\ (10.5) \\ (10.6) \end{array}$$

From Equations (10.1), (10.3), and (10.6) we get the equation:

$$(a_1 - a_2) \quad = \quad -(f_1 - f_2)T_X \quad + \quad (c_1 - c_2)T_Y \tag{11.1}$$

From Equations (10.2), (10.4), and (10.5) we get the equation:

$$(d_1 - d_2) \quad = \quad (e_1 - e_2)T_X \quad - \quad (b_1 - b_2)T_Y \tag{11.2}$$

Equations (11.1) and (11.2) form a set of two linear equations in the two unknown translation parameters: T_X and T_Y ($T_Z = f_c$). Solving those two equations recovers the camera's translation parameters.

In order to recover the camera's rotation parameters, the first six equations of (5) are used, together with the 2D parameters of just one of the planar surfaces, say the first $(a_1, b_1, c_1, d_1, e_1, f_1)$, and with the already computed translation parameters of the camera (T_X, T_Y, T_Z). This yields the following set of six linear equations in six unknowns $(\alpha_1, \beta_1, \gamma_1, \Omega_X, \Omega_Y, \Omega_Z)$:

$$\begin{bmatrix} a_1 \\ b_1 \\ c_1 \\ d_1 \\ e_1 \\ f_1 \end{bmatrix} = \begin{bmatrix} -f_cT_X & 0 & 0 & 0 & -f_c & 0 \\ f_c & -f_cT_X & 0 & 0 & 0 & 0 \\ 0 & 0 & -f_cT_X & 0 & 0 & 1 \\ -f_cT_Y & 0 & 0 & f_c & 0 & 0 \\ 0 & -f_cT_Y & 0 & 0 & 0 & -1 \\ f_c & 0 & -f_cT_Y & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \beta_1 \\ \gamma_1 \\ \Omega_X \\ \Omega_Y \\ \Omega_Z \end{bmatrix} \tag{12}$$

Solving this linear set of equations yields the camera's rotation parameters $\Omega_X, \Omega_Y, \Omega_Z$ (as well as the plane parameters $\alpha_1, \beta_1, \gamma_1$, if not both T_X and T_Y are 0).

Experiment 2: Between Figure 3.a and Figure 3.b the camera was translated by $(T_X, T_Y, T_Z) = (-6cm, -0.5cm, 5cm)$ and was rotated by $(\Omega_X, \Omega_Y, \Omega_Z) = (0°, 2.25°, 0°)$. The 2D detection algorithm described in Section 3 detected the two walls (see Figures 3.c and 3.d). Setting the computed 2D motion parameters in the linear sets of equations (11) and (12) and solving them yields the following

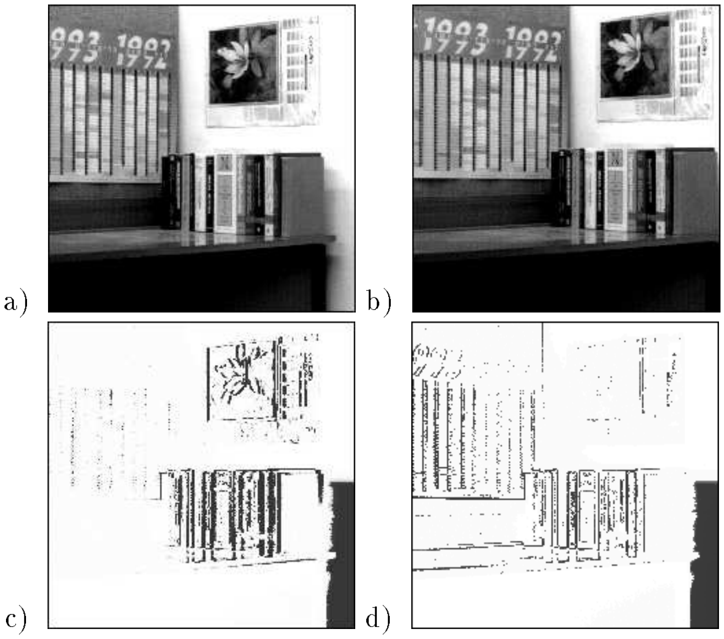

Figure 3: General ego-motion with $T_Z \neq 0$ (Case 2).

a) The first image.

b) The second image, taken after translating the camera by $(T_X, T_Y, T_Z) = (-6cm, -0.5cm, 5cm)$ and rotating it by $(\Omega_X, \Omega_Y, \Omega_Z) = (0°, 2.25°, 0°)$.

c) The difference image taken after registering the two images according to the first computed pseudo 2D projective motion. Dark regions correspond to large intensity differences. It is clear that this motion corresponds to the left wall, as it is nulled in the difference image.

d) The difference image taken after registering the two images according to the second computed pseudo 2D projective motion. It is clear that this motion corresponds to the right wall, as it is nulled in the difference image.

ego-motion: $(T_X, T_Y, T_Z) = (-1049.3_{pixels}, -97.9_{pixels}, f_c)$ and $(\Omega_X, \Omega_Y, \Omega_Z) = (-0.22°, 2.38°, 0.39°)$, where $(f_c = 877_{pixels})$. The translation magnitude cannot be determined, but when setting T_Z to be the correct size ($5cm$), the resulting translation is then: $(T_X, T_Y, T_Z) = (-5.98cm, -0.56cm, 5cm)$.

Case 3: Camera motion with Translation ($T_Z = 0$).

Since there exists a translation and $T_Z = 0$, then either $T_X \neq 0$ or $T_Y \neq 0$. Substituting $T_Z = 0$ in the first *six* equations of (6) (see remark at the end of "Case 1"), we get:

$$\left\{ \begin{array}{lcl} a_1 - a_2 & = & -f_c(\alpha_1 - \alpha_2)T_X \\ b_1 - b_2 & = & -f_c(\beta_1 - \beta_2)T_X \\ c_1 - c_2 & = & -f_c(\gamma_1 - \gamma_2)T_X \\ d_1 - d_2 & = & -f_c(\alpha_1 - \alpha_2)T_Y \\ e_1 - e_2 & = & -f_c(\beta_1 - \beta_2)T_Y \\ f_1 - f_2 & = & -f_c(\gamma_1 - \gamma_2)T_Y \end{array} \right.$$

In this case, only the ratio $\frac{T_Y}{T_X}$ can be recovered. This corresponds to the previous remark that only the direction of the translation can be recovered, and not its magnitude:

$$\frac{T_Y}{T_X} = \frac{d_1 - d_2}{a_1 - a_2} = \frac{e_1 - e_2}{b_1 - b_2} = \frac{f_1 - f_2}{c_1 - c_2}$$

This case can be detected from the computed 2D motions by observing that: $\frac{d_1-d_2}{a_1-a_2} \approx \frac{e_1-e_2}{b_1-b_2} \approx \frac{f_1-f_2}{c_1-c_2}$.
If $T_X \geq T_Y$ (i.e., $\frac{T_Y}{T_X} \leq 1$), then without loss of generality it can be assumed that $T_X = 1$, so that the obtained camera's translation parameters are:

$$T_X = 1, \quad T_Y = \frac{(d_1 - d_2) + (e_1 - e_2) + (f_1 - f_2)}{(a_1 - a_2) + (b_1 - b_2) + (c_1 - c_2)}, \quad T_Z = 0. \tag{13}$$

If, on the other hand, $T_X < T_Y$ (i.e., $\frac{T_Y}{T_X} > 1$), then without loss of generality it can be assumed that $T_Y = 1$, so that the obtained camera's translation parameters are:

$$T_X = \frac{(a_1 - a_2) + (b_1 - b_2) + (c_1 - c_2)}{(d_1 - d_2) + (e_1 - e_2) + (f_1 - f_2)}, \quad T_Y = 1, \quad T_Z = 0. \tag{14}$$

In order to recover the camera's rotation parameters, Equation (5) is used, together with the 2D parameters of just one of the planar surfaces, say the first $(a_1, b_1, c_1, d_1, e_1, f_1, g_1, h_1)$, and with the already computed translation parameters of the camera (T_X, T_Y, T_Z). This yields the following overdetermined set of eight linear equations in six unknowns $(\alpha_1, \beta_1, \gamma_1, \Omega_X, \Omega_Y, \Omega_Z)$:

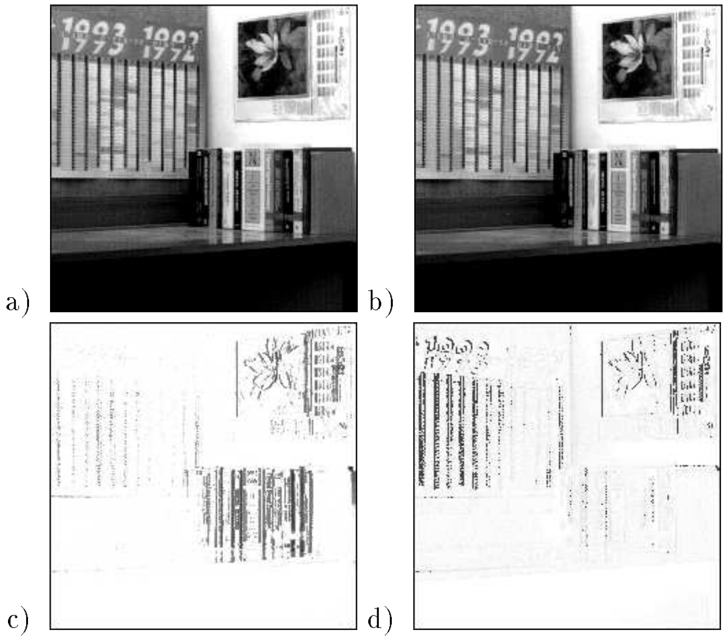

Figure 4: General ego-motion with translation, $T_Z = 0$ (Case 3).
a) The first image.
b) The second image, taken after translating the camera by $(T_X, T_Y, T_Z) = (4.5cm, 0cm, 0cm)$ and rotating it by $(\Omega_X, \Omega_Y, \Omega_Z) = (0°, -0.65°, 0°)$.
c) The difference image taken after registering the two images according to the first computed pseudo 2D projective motion. Dark regions correspond to large intensity differences. It is clear that this motion corresponds to the left wall, as it is nulled in the difference image.
d) The difference image taken after registering the two images according to the second computed pseudo 2D projective motion. It is clear that this motion corresponds to the plane consisting from the back of the books, as this region is nulled in the difference image.

$$
\begin{bmatrix} a_1 \\ b_1 \\ c_1 \\ d_1 \\ e_1 \\ f_1 \\ g_1 \\ h_1 \end{bmatrix} = \begin{bmatrix} -f_c T_X & 0 & 0 & 0 & -f_c & 0 \\ 0 & -f_c T_X & 0 & 0 & 0 & 0 \\ 0 & 0 & -f_c T_X & 0 & 0 & 1 \\ -f_c T_Y & 0 & 0 & f_c & 0 & 0 \\ 0 & -f_c T_Y & 0 & 0 & 0 & -1 \\ 0 & 0 & -f_c T_Y & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{1}{f_c} & 0 \\ 0 & 0 & 0 & \frac{1}{f_c} & 0 & 0 \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \beta_1 \\ \gamma_1 \\ \Omega_X \\ \Omega_Y \\ \Omega_Z \end{bmatrix} \tag{15}
$$

If only the first six equations of (15) are used (i.e., using only the reliable parameters $a_1, b_1, c_1, d_1, e_1, f_1$, and disregarding the unreliable ones, g_1 and h_1), then only Ω_Z can be recovered in this case. In order to recover the two other rotation parameters, Ω_X and Ω_Y, the second order terms g_1 and h_1 must be used. This means that for the case of an existing translation with $T_Z = 0$, only 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. The other two rotation parameters, Ω_X and Ω_Y, can only be approximated.

Experiment 3: Between Figure 4.a and Figure 4.b the camera was translated by $(T_X, T_Y, T_Z) = (4.5cm, 0cm, 0cm)$ and was rotated by $(\Omega_X, \Omega_Y, \Omega_Z) = (0°, -0.65°, 0°)$. The 2D detection algorithm described in Section 3 detected two planar surfaces (the left wall and the row of books; see Figures 4.c and 4.d). Solving Equations (13) and (15) with the computed 2D motion parameters yields the following ego-motion: $(T_X, T_Y, T_Z) = (1,\ 0.106,\ 0)$ and $(\Omega_X, \Omega_Y, \Omega_Z) = (0°, -0.8°, -0.025°)$. Note that in this case, only (T_X, T_Y, T_Z) and Ω_Z are reliable. Ω_X and Ω_Y are only *estimations*, since they were computed using the unreliable parameters g_1 and h_1. The translation magnitude cannot be determined, but when setting T_X to be the correct size ($4.5cm$), the resulting translation is then: $(T_X, T_Y, T_Z) = (4.5cm, 0.47cm, 0cm)$.

3 Detecting Planes and Their 2D Motions

We use existing methods [IRP92] in order to extract two planar surfaces in the scene with their 2D projective transformation. Those methods dealt with analysis of dynamic scenes, in which there were assumed to be multiple moving planar objects with 2D parametric motions. The image plane was segmented into the differently moving objects, and their 2D motion parameters were computed.

In this work we use the 2D detection algorithm in order to detect two different planes and their 2D motion parameters. Due to camera translation planes at different depths will have different 2D motions in the image plane, and will therefore be identified as differently moving planar objects.

This section describes very briefly the technique for detecting differently moving planar objects and their 2D motion parameters. For more details refer to [IRP92].

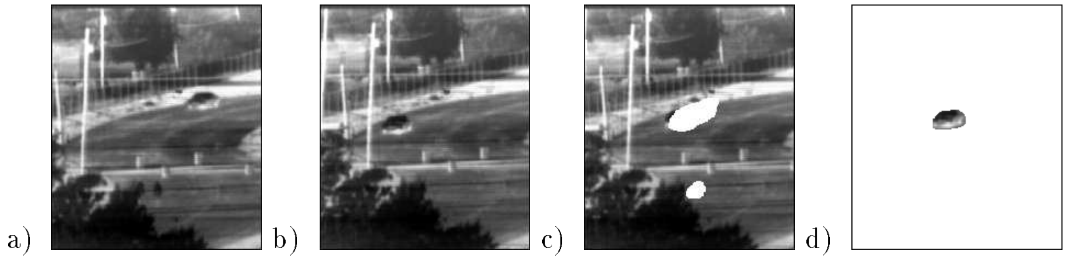

Figure 5: Detecting the first two planar objects in an infrared image sequence with several moving objects.
a-b) The first and last frames in the sequence. The background and the car move differently.
c) The segmented dominant object (the background) after 5 frames, using a 2D affine motion model. White regions are those *excluded* from the detected region.
d) The segmented second dominant object (the car) after 5 frames, using a 2D affine motion model. White regions are those *excluded* from the detected region.

3.1 Detecting Multiple Moving Planar Objects

2D motion estimation is difficult when the region of support of an object in the image is not known, which is the common case. It was shown in [BHK91] that the motion parameters of a single *translating* object in the image plane can be recovered accurately by applying the 2D motion computation framework described in Section 3.3 to the *entire* image, using a 2D *translation* motion model (see Section 3.2). This can be done even in the presence of other differently moving objects in the region of analysis, and with no prior knowledge of their regions of support. This object is called the *dominant object*, and its 2D motion the *dominant 2D motion*. A thorough analysis of hierarchical 2D translation estimation is found in [BHK91]. In [IRP92] this method was extended to compute *higher order* 2D motions (2D affine, 2D projective) of a single planar object among differently moving objects. A segmentation step was added to the process, which segments out the region corresponding to the computed dominant 2D motion. This is the region of the dominant planar object in the image.

After detecting and segmenting the dominant planar object between two image frames, the dominant planar object is excluded from the region of analysis, and the detection process is repeated on the remaining parts of the image to find other planar objects and their 2D motions parameters. More details are found in [IRP92].

Temporal integration over longer sequences was used in order to improve the segmentation and the accuracy of the 2D motion parameters [IRP92]. Figure 5 shows the detection of the first and second dominant planar objects in an image sequence. In this sequence, taken by an infrared camera, the background moves due to camera motion, while the car has another motion. The results are presented after several frames, when the segmentation has improved with temporal integration.

3.2 The 2D Motion Models

2D parametric transformations are used to approximate the projected 3D motions of planar objects on the image plane. The choice of a 2D motion model enables efficient motion computation, and is numerically stable because the problem is overdetermined, as opposed to the general problem of 3D motion computation.

Given two grey level images of an object, $I(x,y,t)$ and $I(x,y,t+1)$, it is assumed that:

$$I(x+p(x,y,t), y+q(x,y,t), t+1) = I(x,y,t), \tag{16}$$

where $(p(x,y,t), q(x,y,t))$ is the displacement induced on pixel (x,y) by the motion of the planar object between frames t and $t+1$. Expanding the left hand side of Equation (16) to its first order Taylor expansion around (x,y,t) and neglecting all nonlinear terms yields:

$$I(x+p, y+q, t+1) = I(x,y,t) + pI_x + qI_y + I_t, \tag{17}$$

where

$$I_x = \frac{\partial I(x,y,t)}{\partial x}, \quad I_y = \frac{\partial I(x,y,t)}{\partial y}, \quad I_t = \frac{\partial I(x,y,t)}{\partial t}, \quad p = p(x,y,t), \quad q = q(x,y,t).$$

Equations (16) and (17) yield the well-known constraint [HS81]:

$$pI_x + qI_y + I_t = 0. \tag{18}$$

The desired 2D motion (p,q) minimizes the error function at Frame t in the region of analysis R:

$$Err^{(t)}(p,q) = \sum_{(x,y)\in R} (pI_x + qI_y + I_t)^2. \tag{19}$$

The error minimization is performed over the parameters of one of the following 2D motion models:

1. **Translation:** 2 parameters, $p(x,y,t) = a, \quad q(x,y,t) = d$. In order to minimize $Err^{(t)}(p,q)$, its derivatives with respect to a and d are set to zero. This yields two linear equations in the two unknowns, a and d. Those are the two well-known optical flow equations [HS81], where every small window is assumed to have a single translation in the image plane. In this translation model, the entire *object* is assumed to have a single translation in the image plane.

2. **Affine**: 6 parameters, $p(x,y,t) = a + bx + cy, \quad q(x,y,t) = d + ex + fy$. Deriving $Err^{(t)}(p,q)$ with respect to the motion parameters and setting to zero yields six linear equations in the six unknowns: a, b, c, d, e, f [BBH+91].

3. **A Moving planar surface** (a pseudo 2D projective transformation): 8 parameters [Adi85] (see Section 2.1), $p(x, y, t) = a + bx + cy + gx^2 + hxy$, $q(x, y, t) = d + ex + fy + gxy + hy^2$. Deriving $Err^{(t)}(p, q)$ with respect to the motion parameters and setting to zero, yields eight linear equations in the eight unknowns: a, b, c, d, e, f, g, h. Our experience shows that the computed parameters g and h are not as reliable as the other six parameters (a, b, c, d, e, f), as g and h are second order terms.

3.3 The Motion Computation Framework

A multiresolution gradient-based iterative framework [BA87, BBH+91, BBHP90] is used to compute the 2D motion parameters. The basic components of this framework are:

- Construction of a Gaussian pyramid [Ros84], where the images are represented in multiple resolutions.
- Starting at the lowest resolution level:
  1. Motion parameters are estimated by solving the set of linear equations to minimize $Err^{(t)}(p, q)$ (Equation (19)) according to the appropriate 2D motion model (Section 3.2). When the region of support of the planar object is known, minimization is done only over that region. Otherwise, it is performed over the entire image.
  2. The two images are registered by warping according to the computed 2D motion parameters.
     Steps 1 and 2 are iterated at each resolution level for further refinement.
  3. The 2D motion parameters are interpolated to the next resolution level, and are refined by using the higher resolution images.

4 Concluding Remarks

In this paper a method was introduced for computing ego-motion in static scenes directly from intensity images. At first, two planar surfaces in the scene are detected, and their pseudo 2D projective transformations between successive frames are computed. The 2D transformations are then used to construct two linear sets of equations: one in the camera's translation parameters, the other in the camera's rotation parameters *in term* of the translation parameters. Solving those two linear sets of equations results in the camera motion.

It was shown that the ego-motion can be recovered reliably in all cases, except for the case of an ego-motion with a translation in the x-y plane only. In such a case, it was shown that only the translation parameters of the camera and the rotation around its optical axis can be recovered accurately. The panning parameters (rotation around the x and y axes) can only be roughly estimated in this special case.

The advantage of the presented technique is in its simplicity (solving small sets of linear equations), and in the robustness and stability of the 2D algorithm. The choice of an initial 2D motion model enables efficient motion computation and numerical stability. There are no severe restrictions on the ego-motion or on the structure of the environment. Using only image intensities, the inherent problems of computing optical flow and of feature matching are avoided.

APPENDIX

In this appendix the experimental results along with their intermediate 2D motion parameters are given. In our experiments a $16mm$ camera was used, where the focal length was found to be approximately 877 pixels (i.e., $f_c = 877$).

Experiment 1:

The camera motion was:

$$\begin{array}{rcl}(T_X, T_Y, T_Z) & = & (0, 0, 0) \\ (\Omega_X, \Omega_Y, \Omega_Z) & = & (0^{rad}, 0.035^{rad}, 0^{rad}) \quad = \quad (0°, 2°, 0°)\end{array}$$

The computed pseudo 2D projective parameters were:

$$(a_1, b_1, c_1, d_1, e_1, f_1, g_1, h_1) = \\ (-29.5322, -0.0004, 0.0028, 0.715, -0.0008, 0.0029, -0.00002, -0.000014)$$

Substituting the values of $(a_1, b_1, c_1, d_1, e_1, f_1, g_1, h_1)$ in Equations (8) and (9) yields the ego-motion:

$$\begin{array}{rcl}(T_X, T_Y, T_Z) & = & (0, 0, 0) \\ (\Omega_X, \Omega_Y, \Omega_Z) & = & (0.0008^{rad}, 0.0337^{rad}, 0.0018245^{rad}) \quad = \quad (0.05°, 1.93°, 0.1°).\end{array}$$

Experiment 2:

The camera motion was:

$$\begin{array}{rcl}(T_X, T_Y, T_Z) & = & (-6cm, -0.5cm, 5cm) \\ (\Omega_X, \Omega_Y, \Omega_Z) & = & (0^{rad}, 0.04^{rad}, 0^{rad}) \quad = \quad (0°, 2.25°, 0°)\end{array}$$

The computed pseudo 2D projective parameters were:

$$(a_1, b_1, c_1, d_1, e_1, f_1, g_1, h_1) = \\ (-20.275, 0.038, 0.0038, 3.344, -0.00645, 0.0345, 0.000057, -0.000075) \\ (a_2, b_2, c_2, d_2, e_2, f_2, g_2, h_2) = \\ (-19.3947, 0.02, 0.0015, 3.0068, -0.00778, 0.0351, 0.000015, -0.00001)$$

Substituting the values of $(a_1, b_1, c_1, d_1, e_1, f_1, g_1, h_1)$ and $(a_2, b_2, c_2, d_2, e_2, f_2, g_2, h_2)$ in the linear sets of equations (11) and (12) yields the following ego-motion:

$$\begin{array}{rcl}(T_X, T_Y, T_Z) & = & (-1049.3_{pixels}, -97.9_{pixels}, \ f_c) \\ (\Omega_X, \Omega_Y, \Omega_Z) & = & (-0.00388^{rad}, 0.0416^{rad}, 0.006775^{rad}) \quad = \quad (-0.22°, 2.38°, 0.39°),\end{array}$$

where ($f_c = 877_{pixels}$). The translation magnitude cannot be determined, setting T_Z to be the correct size ($5cm$), the resulting translation is then: $(T_X, T_Y, T_Z) = (-5.98cm, -0.56cm, 5cm)$.

Experiment 3:

The camera motion was:

$$\begin{array}{rcl} (T_X, T_Y, T_Z) & = & (4.5cm, 0cm, 0cm) \\ (\Omega_X, \Omega_Y, \Omega_Z) & = & (0^{rad}, -0.011^{rad}, 0^{rad}) \quad = \quad (0°, -0.65°, 0°) \end{array}$$

The computed pseudo 2D projective parameters were:

$$\begin{array}{l} (a_1, b_1, c_1, d_1, e_1, f_1, g_1, h_1) = \\ \qquad (0.39, -0.012, -0.0008, -0.799, 0.000844, 0.000027, 0.000016, 0) \\ (a_2, b_2, c_2, d_2, e_2, f_2, g_2, h_2) = \\ \qquad (-0.326, 0.013, 0.00147, -0.875, -0.001166, -0.00045, -0.000044, 0.000001) \end{array}$$

Substituting the values of $(a_1, b_1, c_1, d_1, e_1, f_1, g_1, h_1)$ and $(a_2, b_2, c_2, d_2, e_2, f_2, g_2, h_2)$ in Equations (13) and (15) yields the following ego-motion:

$$\begin{array}{rcl} (T_X, T_Y, T_Z) & = & (1,\ 0.106,\ 0) \\ (\Omega_X, \Omega_Y, \Omega_Z) & = & (0^{rad}, -0.014^{rad}, -0.000435^{rad}) \quad = \quad (0°, -0.8°, -0.025°). \end{array}$$

Note that in this case, only (T_X, T_Y, T_Z) and Ω_Z are reliable. Ω_X and Ω_Y are only *estimations*, since they were computed using the unreliable parameters g_1 and h_1. The translation magnitude cannot be determined, but when setting T_X to be the correct size ($4.5cm$), the resulting translation is then: $(T_X, T_Y, T_Z) = (4.5cm, 0.47cm, 0cm)$.

References

[AD92] Y. Aloimonos and Z. Duric. Active egomotion estimation: A qualitative approach. In *European Conference on Computer Vision*, pages 497–510, Santa Margarita Ligure, May 1992.

[Adi85] 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.

[BA87] J.R. Bergen and E.H. Adelson. Hierarchical, computationally efficient motion estimation algorithm. *J. Opt. Soc. Am. A.*, 4:35, 1987.

[BAHH92] 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.

[BBH+91] J.R. Bergen, P.J. Burt, K. Hanna, R. Hingorani, P. Jeanne, and S. Peleg. Dynamic multiple-motion computation. In Y.A. Feldman and A. Bruckstein, editors, *Artificial Intelligence and Computer Vision: Proceedings of the Israeli Conference*, pages 147–156. Elsevier, 1991.

[BBHP90] J.R. Bergen, P.J. Burt, R. Hingorani, and S. Peleg. Computing two motions from three frames. In *International Conference on Computer Vision*, pages 27–32, Osaka, Japan, December 1990.

[BHK91] P.J. Burt, R. Hingorani, and R.J. Kolczynski. Mechanisms for isolating component patterns in the sequential analysis of multiple motion. In *IEEE Workshop on Visual Motion*, pages 187–193, Princeton, New Jersey, October 1991.

[DF90] R. Deriche and O. Faugeras. Tracking line segments. In *Proc. 1st European Conference on Computer Vision*, pages 259–268, Antibes, April 1990.

[FJ90] D.J. Fleet and A.D. Jepson. Computation of component image velocity from local phase information. *International Journal of Computer Vision*, 5(1):77–104, 1990.

[GU91] R. Guissin and S. Ullman. Direct computation of the focus of expansion from velocity field measurements. In *IEEE Workshop on Visual Motion*, pages 146–155, Princeton, NJ, October 1991.

[HA91] L. Huang and Y. Aloimonos. Relative depth from motion using normal flow: An active and purposive solution. In *IEEE Workshop on Visual Motion*, pages 196–204, Princeton, NJ, October 1991.

[Han91] 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.

[Hee88] D.J. Heeger. Optical flow using spatiotemporal filters. *International Journal of Computer Vision*, 1:279–302, 1988.

[Hor90] B.K.P. Horn. Relative orientation. *International Journal of Computer Vision*, 4(1):58–78, June 1990.

[HS81] B.K.P. Horn and B.G. Schunck. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981.

[HW88] B.K.P. Horn and E.J. Weldon. Direct methods for recovering motion. *International Journal of Computer Vision*, 2(1):51–76, June 1988.

[IRP92] 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.

[JH91] A.D. Jepson and D.J. Heeger. A fast subspace algorithm for recovering rigid motion. In *IEEE Workshop on Visual Motion*, pages 124–131, Princeton, NJ, October 1991.

[LR83] D.T. Lawton and J.H. Rieger. The use of difference fields in processing sensor motion. In *DARPA IUWorkshop*, pages 78–83, June 1983.

[NL91] 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.

[OFT87] 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.

[Ros84] A. Rosenfeld, editor. *Multiresolution Image Processing and Analysis.* Springer-Verlag, 1984.

[SM90] M. Shizawa and K. Mase. Simultaneous multiple optical flow estimation. In *International Conference on Pattern Recognition*, pages 274–278, Atlantic City, New Jersey, June 1990.

[Sun91] V. Sundareswaran. Egomotion from global flow field data. In *IEEE Workshop on Visual Motion*, pages 140–145, Princeton, NJ, October 1991.

[Taa92] M.A. Taalebinezhaad. Direct recovery of motion and shape in the general case by fixation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 14:847–853, August 1992.

[TS91] M. Tistarelli and G. Sandini. Direct estimation of time-to-impact from optical flow. In *IEEE Workshop on Visual Motion*, pages 226–233, Princeton, NJ, October 1991.