Fast Probabilisitic Estimation of Egomotion From Image Intensities

Jamil Draréni Nicolas Martin
Sébastien Roy
Département d'Informatique et recherche opérationnelle, Université de Montréal
CP 6128 succ Centre-Ville, Montreal QC, Canada.
{drarenij,martinic,roys}@iro.umontreal.ca

Abstract

This paper proposes a real-time probabilistic solution to the problem of camera motion estimation in a video sequence. Instead of using explicit tracking of features, it only uses instantaneous image intensity variations without prior estimation of optical flow. We represent the camera motion as a probability density which is constructed from the individual motion densities, estimated from spatio-temporal derivatives, of each pixel of the image. The density is formed by accumulating the contribution of each pixel, making it very robust to local perturbations in the image. A fast algorithm is proposed and experimental results show how real-time motion estimation is possible directly from the image stream with good precision.

1. Introduction

Determining the egomotion of a moving camera relative to its environment from a sequence of images is very useful in passive navigation and in video coding. Methods for egomotion computation fall in two major categories: direct and indirect approaches.

Indirect approaches perform first, a features correspondence or optical flow computation to obtain image velocities. After that, the equations relating these image velocities to the 3D motion parameters are solved yielding an estimate of the camera egomotion [6, 5]. Such methods are time-consuming due to the intermediate stages that must be solved. Moreover, these methods perform badly under small baselines.

Direct approaches consist of computing the egomotion parameters from the normal flow which is fully determined by the spatio-temporal derivatives of the image sequence. these approaches are quoted ”direct methods” because they rely only on image properties [7].

In this paper we propose a method to estimate egomotion parameters of a camera directly from the normal flow without making any assumption on the scene geometry. Typical optical flow and associated normal flow are illustrated in fig.1.

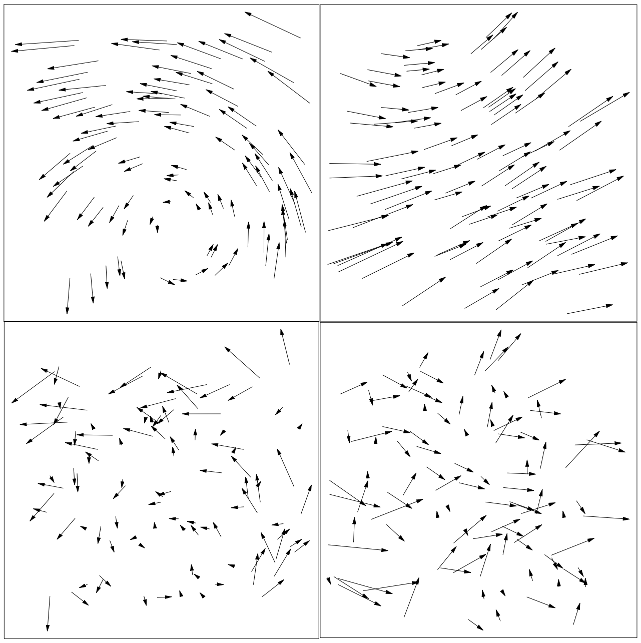

Figure 1. **Optical Flow.** Top) full optical flow induced by typical camera motions. Bottom) Example of normal flows observed for the same camera motions.

We propose to define a probability distribution for the instantaneous camera motion (egomotion) based on the estimation of normal flow [3]. The camera motion, represented by a rotation vector Ω and a translation vector T, is related directly to the spatio-temporal derivatives $\dot{I}$ at an image point p obtained from the image sequence. Therefore, we must estimate the density

$$p(\Omega; T|\dot{I})$$

First, in section 2 we define the camera motion equations and adapt it to normal flow directly instead of optical flow.

Section 3, defines a probabilistic model to relate spatio-temporal derivatives and pixel's motion estimate in the context of optical flow. After that, the probabilistic framework for camera motion estimation will be described, followed by experimental results in section 4. Finally, conclusions will be drawn in section 4 along with future directions.

2. Camera Motion Equations

The 3D motion V of a 3D point P (expressed in the camera coordinate system) under a camera motion (Ω, T) is

$$V = -T - \Omega \times P \qquad (1)$$

where T is the camera translation and Ω is the rotation represented as a vector whose direction is the axis of rotation and the norm is the magnitude of the rotation [10].

The projection of V in the image, v, is obtained by first projecting P onto the image to obtain the image point p (in camera coordinates):

$$p = \frac{P}{P_z} \qquad (2)$$

where P_z is the z coordinate of point P.

Deriving equation (2) leads to the observed motion in the image:

$$v = p' = (\frac{P}{P_z})' = \frac{P_z P' - P P'_z}{P_z^2} = \frac{P_z V - P V_z}{P_z^2}$$

After replacing V from (1) we obtain the motion field v induced by the camera motion (Ω, T)

$$v = \mathbf{M}_\Omega \, \Omega + \frac{1}{P_z} \mathbf{M}_T \, T \qquad (3)$$

Where which features a translational part $\mathbf{M}_T$ that depends on reverse scene structure $(\frac{1}{P_z})$ and a rotation part $\mathbf{M}_\Omega$ that is invariant to scene structure.

2.1. Getting rid of depth

In (3), if the depth P_z of each point is known then it is easy to build a linear equation system using a minimum of 3 points to solve for the 6 unknowns (Ω, T) of the camera motion:

$$v^i = M^i_\Omega \Omega + d^i M^i_T T$$

where d^i is the inverse depth $\frac{1}{P_z}$ of point i.

In practice, depth is usually unknown. However, if the depth is uniformly distributed (ex: urban navigation), we can change the previous system to:

$$v^i = M^i_\Omega \Omega + \overline{d} M^i_T T$$

where $\overline{d}$ is the average scene disparity, defined as

$$\overline{d} = \frac{1}{N} \sum_{i=1}^{N} d^i = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{Z^i}.$$

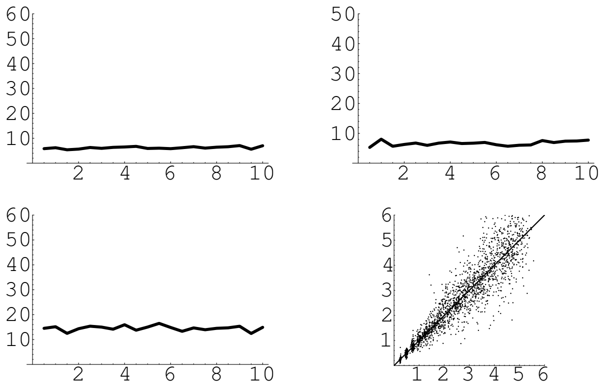

Figure 2. Random camera motions were solved from normal flows while varying the average scene disparity value $\overline{d} = 1/Z$ from $1/0.5$ to $1/10$. A) rotation axis error, expressed as an angle with the true axis, B) rotation magnitude error, expressed as a fraction of true magnitude, C) Translation direction error, expressed as angle with true direction, and D) Ratios of computed translation magnitude to true magnitude compared with ratios of used $\overline{d}$ to true scene average disparity.

We observed empirically that the solution of this new system is close to the same the original system. In fig. 2, we show how varying the constant $\overline{d}$ has no effect on the accuracy of rotation estimation (cf. fig. 2-A,B) or translation direction (cf. fig. 2C), as indicated by the flat error curves obtained. Also, the translation magnitude is scaled from the true magnitude in proportion to the ratio of $\overline{d}$ to the true average scene disparity (cf. fig. 2D). The exact context where this *average scene disparity* property holds remains to be investigated.

If the exact average scene disparity $\overline{d}$ is not known, then an arbitrary value can be used for $\overline{d}$ and the resulting translation will be scaled by the same factor.

Even if the translation is estimated up to an unknown scale factor in each image along a sequence, there are many ways to relate translation magnitude temporally along a sequence. This is usually accomplished by making various assumptions regarding the motion or scene structure, such as, but not limited to, scene rigidity, constant velocity, planar motion and planar world [4]. In our case, we focus on instantaneous motion recovery only, leaving the translation scale determination to other subsequent algorithms. This reduces the need for specific constraints about the motion or the scene and makes the algorithm more versatile.

2.2. From optical flow to normal flow

We can see how (3) relates the optical flow to camera motion. In practice, optical flow must be computed first. Since this is an ill-posed problem, the accuracy of flow field is low and it requires a large amount of time to compute.

An alternative approach is to use the so called *nor-*

mal flow, which provides the motion component along the image-gradient direction. The normal flow field is fully determined by the image intensity derivatives and is thus fast to compute and more accurate than optical flow. We must reformulate (3) in terms of normal flow.

The solution space of the system (3) is represented by a 4D hyperplane in the motion space that satisfies the camera motion equation. Any motion that falls onto the hyperplane is a putative solution while any motion outside the hyperplane is not possible.

A 4D hyperplane is represented in the motion space with two normals: n_1 and n_2. The position s of the hyperplane is set by using a sample solution

$$s = (0, 0, 0, -P_z v_x, -P_z v_y, 0).$$

Intuitively, this solution explains the image motion of a single pixel observed by a simple camera translating parallel to the image plane and in the opposite direction of the observed pixel motion v.

The 4D hyperplane has two normals n_1 and n_2 which are directly extracted from (3) as

$$\begin{aligned} n_1 &= (p_x p_y, -(1+p_x^2), p_y, -1/P_z, 0, p_x/P_z) \\ n_2 &= ((1+p_y^2), -p_x p_y, -p_x, 0, -1/P_z, p_y/P_z) \end{aligned}$$

It then follows that any camera motion $c = (\Omega, T)$ consistent with a single pixel motion (p, v, P_z) is represented by the triplet (s, n_1, n_2) and satisfies

$$(c - s) \cdot n_1 = 0 \qquad \text{and} \qquad (c - s) \cdot n_2 = 0.$$

From the constant brightness assumption (CBA) illustrated in fig.3, we see that the optical flow v is constrained along a line defined by the normal flow v_n as

$$v = v_n + k v_n^{\perp} \qquad (k \in -\infty \dots \infty)$$

Replacing the definition in s yields

$$s = s' + k n_3$$

where $s' = (0, 0, 0, -P_z v_{nx}, -P_z v_{ny}, 0)$ and $n_3 = (0, 0, 0, P_z v_{ny}, -P_z v_{nx}, 0)$. Considering all possible values of k is equivalent to add a dimension to the 4D hyperplane. The new 5D hyperplane has a single normal n_0 which must be a linear combination of n_1 and n_2 and orthogonal to n_3. We have

$$n_0 \cdot n_3 = (\alpha n_1 + \beta n_2) \cdot n_3 = 0$$

This system is easy to solve and yields $(\alpha, \beta) = v_n$ so

$$n_0 = v_{nx} n_1 + v_{ny} n_2.$$

Now we can connect the normal flow to the camera motion

$$n_0 \cdot (c - s') = 0$$

:

$$\begin{pmatrix} v_{nx} P_x P_y + v_{ny}(1 + P_y)^2 \\ -v_{nx}(1 + P_x^2) - v_{ny} P_x P_y \\ v_{nx} P_y - v_{ny} P_x \\ -v_{nx}/P_z \\ -v_{ny}/P_z \\ (v_{nx} P_x + v_{ny} P_y)/P_z \end{pmatrix}^{\perp} \begin{pmatrix} \Omega \\ T \end{pmatrix} = ||v_n||^2 \tag{4}$$

We can now solve directly for camera motion from the normal flow. The scene disparity can be handled in the same way as before by substituting the average scene disparity $\overline{d}$ for each individual disparity $1/P_z$ in (4).

3. Probabilistic Model of Camera Motion

The normal flow used in solving for the camera motion is determined by the image intensity derivatives, which are known to be noisy [9].

If we consider the image derivatives as random variables with some known probability densities, then we can derive a probability density

$$p(\Omega; T | \dot{I})$$

of camera motions conditional on measurements of image derivatives $\dot{I}$ at an image point p.

This distribution is broken in two terms, one expressing how the camera motions explains to normal flow, and one expressing how the normal flow relates to image intensity derivatives. We have that, for each point p:

$$p(\Omega; T | \dot{I}) = \int p(\Omega; T | v_n) p(v_n | \dot{I}) \mathrm{d} v_n \tag{5}$$

These two terms are described in the following subsections.

3.1. Camera Motion from normal flow

We define the probability distribution of the camera motion (Ω, T) for a given normal flow v_n at pixel p as

$$p(\Omega; T | v_n)$$

where the depth of point p is assumed to have been replaced by a constant as in §2.1 so the depth P_z does not appear in the density.

In practice, the squared distance between a motion point $c = (\Omega, T)$ and the 5D hyperplane of (4) will be used to estimate the probability of motion. This distance has the form

$$D(c, p, v_n) = ((c - s') \cdot n_0)^2$$

Note that n_0 is assumed to be normalized.

The probability density is defined as

$$p(\Omega; T | v_n) \propto e^{-D((\Omega, T), p, v_n)^2}$$

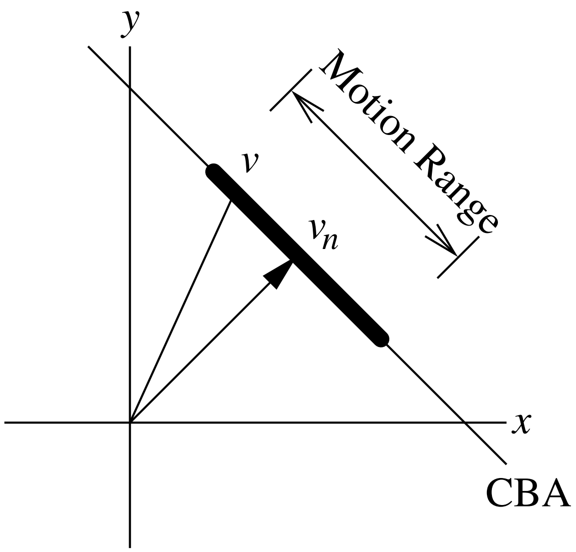

Figure 3. **Constant Brightness assumption.** The normal flow v_n determines a line that defines the possible motions v. We restrict the motion within an interval around v_n.

3.2. Optical flow and image derivatives

As proposed in [9], we define the probability of a true spatio-temporal gradient $\dot{I}^*$ from the measurements $\dot{I}$ as

$$p(\dot{I}^*|\dot{I}) = N(\dot{I}, \Sigma) \tag{6}$$

which is a Gaussian distribution centered around the measurement and covariance Σ.

We easily model the distribution of normal image flow v_n from a given true derivative $\dot{I}^*$ as the density

$$p(v_n|\dot{I}^*) = \delta(-\frac{\dot{I}^*_t}{||\nabla\dot{I}^*||^2}\nabla\dot{I}^*)$$

where $\delta(x)$ is the impulse density function. We can then define the distribution of normal flow for an intensity measurement $\dot{I}$ as

$$\begin{aligned} p(v_n|\dot{I}) &= \int p(v_n|\dot{I}^*)p(\dot{I}^*|\dot{I})\mathrm{d}\dot{I}^* \\ &= \int \delta(-\frac{\dot{I}^*_t}{||\nabla\dot{I}^*||^2}\nabla\dot{I}^*)N(\dot{I},\Sigma)\mathrm{d}\dot{I}^* \end{aligned} \tag{7}$$

Assuming that the full motion v lies within an interval $\pm k$ from the normal flow v_n on the CBA line, we can derive a density $p(v|\dot{I})$ for the full flow by using the relation

$$v = v_n + \lambda v_n^\perp$$

where λ is a random variable with uniform density $U[-k, k]$. The illustrations in fig.4 depict the distributions of v for various levels of spatial texture. Notice how the density gracefully represents the increase of motion ambiguity as texture fades.

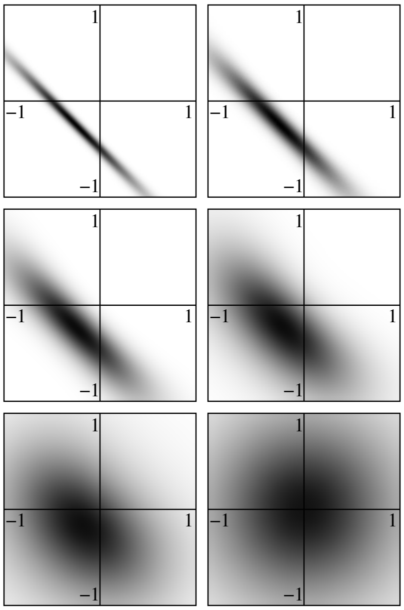

Figure 4. Distribution of v. The spatio-temporal gradient is $\lambda(1, 1, \frac{1}{2})$, with spatial gradients magnitudes set to $\lambda = [16, 8, 4, 2, 1, 0]$ from top-left to bottom-right. The derivative noise is set to a fixed level $\sigma = (1, 1, 1)$.

3.3. Global density of camera motion

The probability of camera motion $p(\Omega; T)$ associated with multiple pixels is simply the sum of individual densities of each image point p_i

$$p(\Omega, T) = \sum_i p(\Omega; T|\dot{I}_i).$$

The sum is used in order to provide robustness against multiple simultaneous motions and errors induced by outliers. To find the most probable camera motion, we can simply maximize $p(\Omega; t)$ using a simple gradient descent. The function maximized is continuous and very smooth, making the search converging very quickly while generally avoiding local minima.

Since the density of camera motion cannot be derived analytically, it is built using samples. For each image point p, we compute the spatio-temporal gradient $\dot{I}$. A number of samples are generated around that gradient according to our model (see (6)). For each sample gradient $\dot{I}$, a hyperplane for (p, v_n) is obtained. The hyperplane samples are cumulated and generate the final density of camera motions.

In practice, (5) can be hard to represent directly. We use samples to approximate the density $p(v_n|\dot{I})$. We have also replaced the Gaussian model $N(\dot{I}, \Sigma)$ of (7) by an impulse density $\delta(\dot{I})$ to represent cases where no image noise is expected.

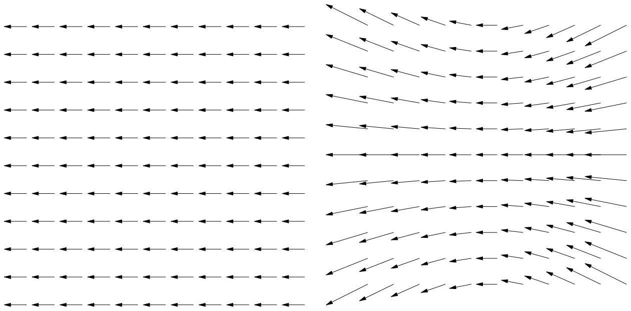

Figure 5. **Translation - Rotation ambiguity.** Left) A motion field for horizontal translation. Right) A field for pure rotation around the y axis.

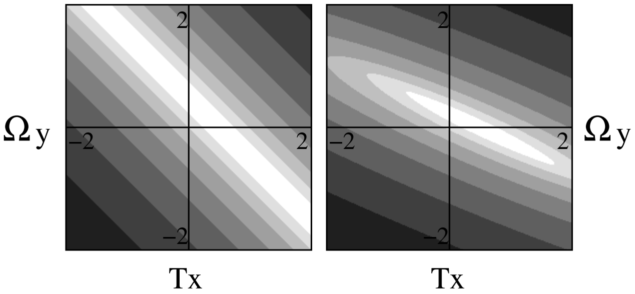

Figure 6. **Translation - Rotation ambiguity.** Densities are shown for T_x and Ω_y. Left) Density for a point in the middle of the image. The minimum is ambiguous. Right) Density for a point in a corner of the image, removing the ambiguity.

3.4. Rotation - Translation Ambiguity

It is well known that horizontal translation (T_x) is hard to distinguish from rotation around the Y axis ($\Omega = (0, \Omega_y, 0)$) [1]. This is illustrated in fig.5. This fact should be reflected in the density of egomotion. To demonstrate this, we have computed the egomotion density for one image point locate in the middle of the image ($p = (0, 0)$) with a horizontal motion ($v = (1, 0)$). The density, shown on the left side of fig.6 is ambiguous. Translation and Rotation cancel can each other perfectly.

However, when the selected image point is located in the upper left corner ($p = (-1, -1)$) with the same horizontal motion, it is possible to resolve the ambiguity, as shown in the density on the right of fig. 6. The minimum in this case is clearly defined and yields the correct minimum.

4. Experiments and results

Our method was tested on synthetic and real image sequences. In all cases, we computed the full density of egomotion, where no *a priori* knowledge is available on the camera motion or scene observed.

In all the experiments presented, images were reduced to a size of 256^2 pixels or less. From the images, a small portion of the image pixels are randomly selected (typically 10%, or 400 points). With such low number of points, the running time is close to real time at 30 frames per second on a 2Ghz PC.

Also, no temporal smoothing of any kind is applied on the camera motion computed at each frame along a sequence. The trajectories presented are the simple accumulation of raw instantaneous motion data.

The 3D camera motion trajectory is illustrated with a vector depicting the z axis of the camera and a vector depicting the y axis.

4.1. Synthetic sequence

The synthetic sequence used for our tests is presented at the top of fig.7. It features a large number of shaded spheres distributed randomly into a large cube. The camera undergoes a circular motion around the spheres, always looking toward the center. Notice that this sequence contains a large amount of occlusions and very smooth textures, thereby making it very difficult to track using feature point detection. It can be considered ”cluttered” as described by Mann et al. [8].

The results, shown in fig.7, show excellent trajectory recovery. In the middle result, the scene depth was known for each point. This allows the algorithm to compute the exact translation magnitude. However, it was slightly over estimated, which explains why the trajectory does overlap itself slightly.

The bottom result of fig.7 was computed using a single depth value along the sequence. The trajectory is now slightly distorted, because the translation magnitude is over estimated when the distribution of sphere depth comes closer to the camera. This happens four times as the camera circles around the corner of the cube of spheres. In this case, we made the assumption that the depth distribution in the scene is fixed, which is not satisfied. A better assumption would have been to impose constant translational velocity.

The fig.8 illustrates on the left how the average depth of the scene varies as the camera circles around the scene. When a corner of the cube of spheres comes closer to the camera (at the diagonal lines) the average depth is closer to the camera. On the right of fig.8, the impact of this average depth variation on the computed translation magnitudes is illustrated. When the average scene depth is over estimated, the translation magnitude is also over estimated.

4.2. Real image sequences

Real image sequences were taken with a consumer video camera. The motion of the camera is only known approximately, since the sequences were taken handheld without a calibrated rig.

The fig.9 shows a forward motion sequence through a corridor. Notice the lack of texture and large amount of

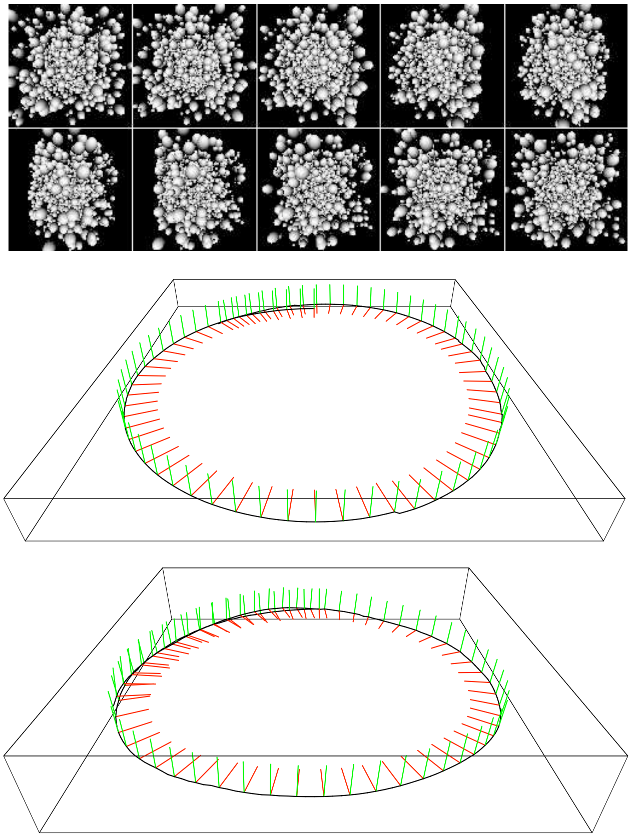

Figure 7. **Synthetic sequence.** Top) The image sequence. Middle) recovered camera motion trajectory for known depth of scene. Bottom) recovered camera motion trajectory computed without knowledge of depth.

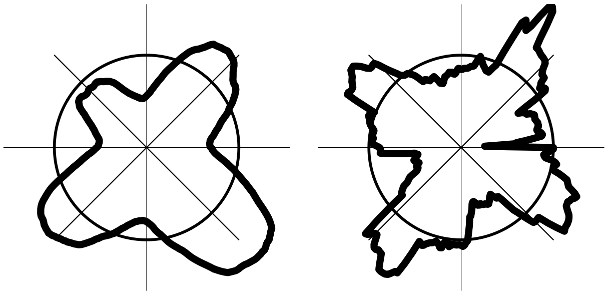

Figure 8. **Translation scale factor.** Left) Average scene disparity as camera turns around. The 4 bumps near diagonals correspond to the scene getting closer to the camera. Right) Estimated translation magnitude for constant scene depth. The magnitude is over estimated when the scene depth is closer than assumed.

specularity on the floor. The results show excellent recovery of the forward trajectory.

The fig.10 illustrates a sideway motion sequence (along the x axis of the camera) taken under very low lighting conditions (the images are histogram equalized for illustration purposes). The result, shown in fig.11, show excellent recovery of the sideway motion.

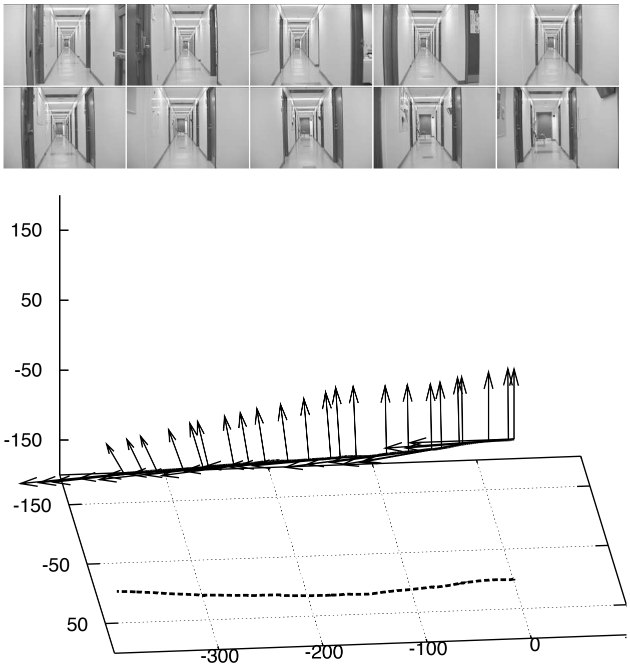

Figure 9. **Corridor sequence and recovered trajectory of the camera.** The 3D trajectory is projected on the X-Z plane and runs along the Z axis.

5. Conclusion

We presented a fast method to express the probability density of camera motion directly from image intensity derivatives, thereby removing any need for full optical flow estimation. Only the normal flow is used.

We proposed to use the average disparity of the scene for all points to alleviate the depth estimation problem. This eases considerably the process of estimating the camera translation. It also makes it easy to eventually express temporal constraints about the evolution of scene depth in time.

The probabilistic approach makes this method very robust to image noise and can naturally express camera motion ambiguity when these are effectively present in a sequence. In this framework, image sequences with low texture, bad illumination conditions, or large amount of occlusions can be handled without difficulty.

References

[1] G. Adiv. Inherent ambiguities in recovering 3-d motion and structure from a noisy flow field. *Transaction in Pattern Analysis and Machine Intelligence*, pages 477–489, 1989. 5

[2] Y. Aloimonos and Z. Duric. Estimating the heading direction using normal flow. *International Journal of Computer Vision*, pages 33–56, 1995.

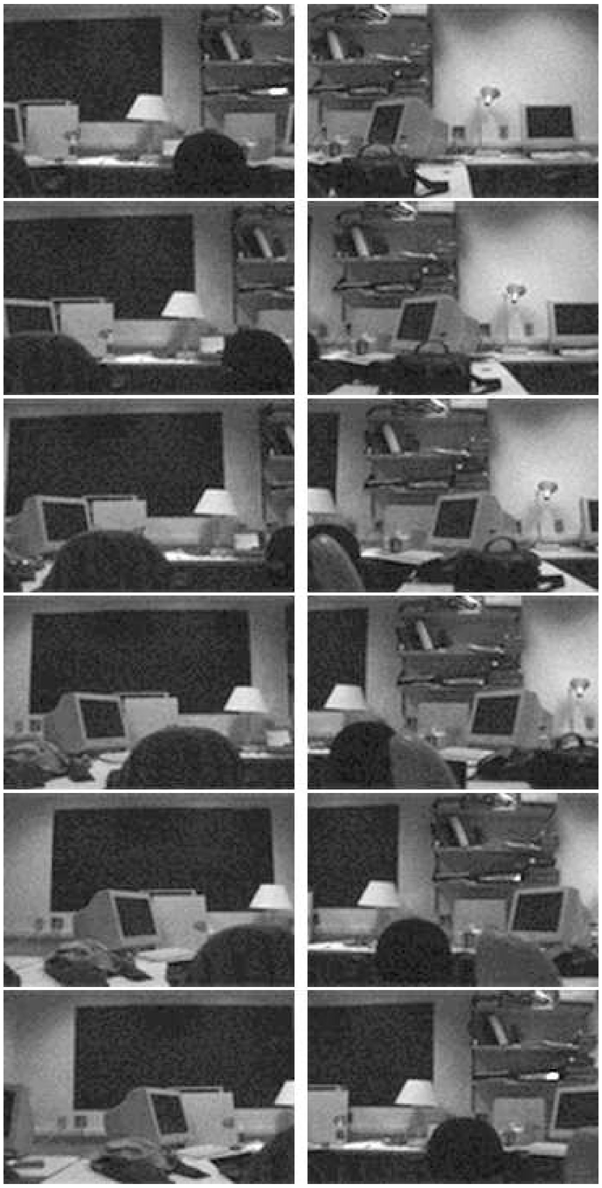

Figure 10. **Sideways motion.** Camera is moving left under low lighting conditions. Notice the amount of pixel noise.

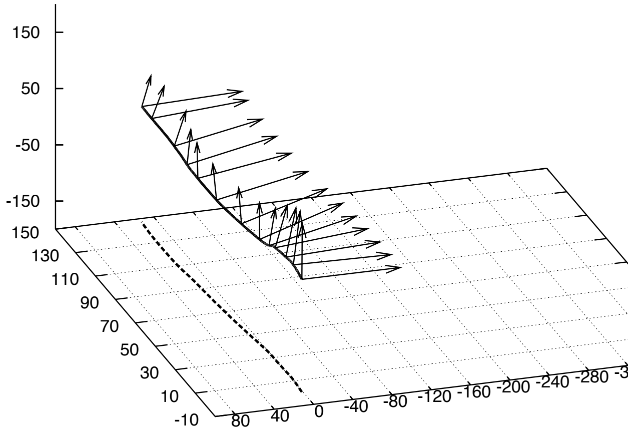

Figure 11. Trajectory for Sideways sequence in low lighting conditions. The trajectory is projected on the X-Z plane and runs along the X axis.

[3] J. L. Barron, D. J. Fleet, and S. S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, pages 43–77, 1994. 1

[4] J.-Y. Bouguet and P. Perona. Visual navigation using a single camera. In *International Conference on Computer Vision*, pages 645–653, 1995. 2

[5] A. Branca, E. Stella, and A. Distante. Mobile robot navigation using egomotion estimates. *International Conference on Intelligent Robots and Systems*, pages 533–537, 1997. 1

[6] W. Burger and B. Bhanu. Estimating 3-d egomotion from perspective image sequences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1040–1058, 1990. 1

[7] C. Fermller and Y. Aloimonos. Qualitative egomotion. *International Journal of Computer Vision*, pages 7–29, 1995. 1

[8] R. Mann and M. S. Langer. Optical snow and the aperture problem. In *International Conference On Pattern Recognition*, pages 264–267, 2002. 5

[9] G. P.Stein, O. Mano, and A. Shashua. A robust method for computing vehicle ego-motion. In *IEEE Intelligent Vehicles Symposium*, pages 362–368, 2000. 3, 4

[10] E. Trucco and A. Verri. *Introductory Techniques for 3-D Computer Vision*. Prentice Hall, New Jersey, 1998. 2