Recursive Flow Based Structure from Parallax with Automatic Rescaling

Marco Zucchelli   Henrik I. Christensen
CVAP & CAS, Royal Institute of Technology, Stockholm, Sweden S100 44

Abstract

In this paper we propose a technique to recursively estimate structure and motion from a sequence of images acquired by an hand held camera. The algorithm is based on estimation of instantaneous optical flow for each frame of the sequence and on the use of the differential epipolar constraint to compute motion and structure. An alternative procedure to manage scale ambiguity across frames is also proposed. Neither user interaction nor any kind of scene knowledge is required and the algorithm, given its low complexity, is adaptable to real time applications.

1 Introduction

The different approaches to the Structure from Motion problem can be roughly classified into *discrete* and *differential* schemes. In the first case snapshots are taken from different point of views and, in general, no motion smoothness is involved. Epipolar geometry is used to constrain the matching of features and recover the shape of the scene. In the second case the camera moves while it captures a continuous stream of images. Consecutive images are very close and the displacement of features does approximate pretty well optical flow: the differential version of the epipolar constraint, can be used to compute camera velocity and depths. The algorithm we propose in this paper belongs to the second category: we use sequences acquired at 30 Hz by a hand held moving camera. We track sparse features over the sequence and then use the optical flow to recover scene shape. The procedure has a layered structure: for each frame, first egomotion is estimated, then the structure is computed. The structures relative to each frame of the sequence are recursively integrated to provide an optimal estimate. In this framework we propose an alternative approach to handle scale ambiguity across the sequence and compare it to the more classic approach of fixing one feature. We tested our algorithm both on simulated data and real sequences and show the results we obtained. The algorithm is completely automatic and, since it is mostly built on linear optimization, it can successfully be used for real time applications. The literature available is vast, so we report here the papers that most inspired our work. A description of the qualitative properties of the optical flow can be found in [1] while a comparison of different techniques to compute it has been reported by Fleet and Barron [2]. For sparse flow estimation we refer to the Tomasi and Kanade classical tracker [3]. Tomasi and Heeger [4] compared a number of the best known linear and non linear techniques for egomotion estimation from sparse flow fields. MacLean [5] proposed a whitening method to reduce the bias that applies in the case of the Jepson-Heeger Linear Subspace [6] method. Another recent and interesting closed form method for egomotion was presented by Ma, Sastry and Kosecka [7].

BMVC 2001 doi:10.5244/C.15.20

For reconstruction algorithms based on optical flow we remind the work of Szeliski [8] who presented a non linear least square technique to reconstruct simultaneously structure and motion from an image sequence and Barron Jepson and Tsotsos [9] propose a Kalman filter based approach to reconstruct structure from lateral displacements. Soatto and Perona worked extensively on Kalman filter formulations of this problem [10, 11]. Pentland [12] first proposed to handle the scale problem fixing a feature and to estimate focal length simultaneously with structure and motion. Comparison between differential and discrete approaches can be found in [13]; A very comprehensive review of multiview point stereo can be found in the book by Zisserman and Hartley [14] and in the D. Nister Ph.D thesis [15]. Good references for a general overview of the problem are [16, 17].

2 3D Motion and Optical Flow

We denote with $\mathbf{X} = (X, Y, Z)$ the position of a scene point with respect to a camera centered coordinate system, and with $\mathbf{x} = (\frac{X}{Z}, \frac{Y}{Z}, 1)$ the projection of such a point on the focal plane of the camera. The camera relative position and orientation is given by the couple $(R, \mathbf{T})$ where R is a rotation matrix and $\mathbf{T}$ the translation vector. The couple $(\mathbf{\Omega}, \mathbf{V})$ represents the rotational and linear velocity of the camera.

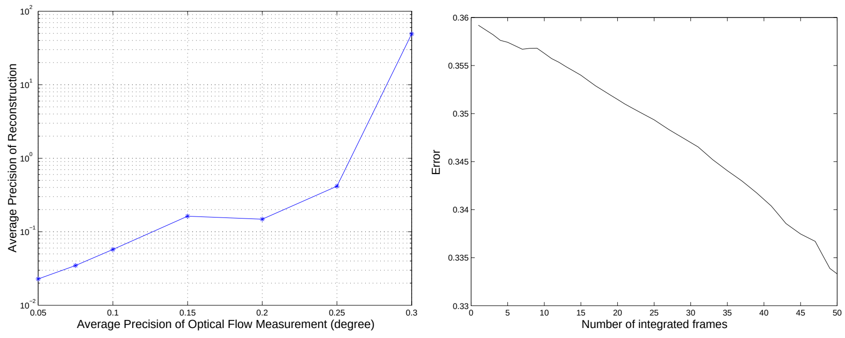

Figure 1: (**a**) Performance of the $LLSE$ depth estimation for single frame. Motion is computed with the 2 steps process as described. Error is in pixels and motion is approximately 0.6 pixels/frame. (**b**) Global performance of the algorithm as a function of the number of frames integrated. Error is 0.20 pixels and motion is approximately 0.6 pixels/frame.

The velocity of a point $\mathbf{X}$ seen from a moving camera is given en by:

$$\dot{\mathbf{X}} = -\mathbf{\Omega} \wedge \mathbf{X} - \mathbf{V} \tag{1}$$

We call velocity field $\mathbf{v}(x, y, t)$ the velocities of the projected points on the focal plane at time t.

3 Algorithm

The process of reconstruction is divided into three different stages: first sparse features are extracted and tracked over time using the $Kanade - Lucas - Tomasi$ tracker. Then egomotion is computed for each frame using the linear subspace method [6] and refined by non linear optimization. For the reconstruction of the frame by frame 3D position of scene points we propose a linear least squares routine that estimates not only the depths but also their first time derivative. This is the basis for a procedure to automatically rescale all the views to the same scale, result that traditionally is achieved using some kind of knowledge of the scene or motion. Finally the information retrieved for each single view is time integrated into a 3D model. Results on both synthetic and real sequences are shown to demonstrate the performance of this method.

3.1 Simulation Benchmarks

We extensively tested the algorithm using synthetically generated flow fields. For homogeneity and simplicity we used the same experimental conditions and benchmarks as in [4]. The focal length of the simulated camera is set to 1 and the focal plane dimensions to 512×512 pixels. The field of view is 90 degrees. Random cloud of 100 points are generated in a depth range of 2-8 focal lengths. The motion is a combination of rotations and translations. The rotational speed magnitude was constant and chosen to be 0.23 degrees per frame. The magnitude of the linear velocity was chosen to fixate the point at the center of the random cloud. Zero-mean gaussian noise was added to the components of the velocity and the standard deviation of such noise chosen to be constant. Sensitivity in the depth is measured as the standard deviation of the distribution of the relative distance of the reconstructed depths from their their real values. In formula:

$$\sigma_Z = \sqrt{E([\frac{Z_i - \hat{Z}_i}{Z_i}]^2)} \tag{2}$$

Where the '^' generally indicates measured quantities and Z the generated depth. The index i runs over the features.

The sensitivity on translation is measured as:

$$\sigma_{\mathbf{T}} = \sqrt{\frac{1}{M-1}\sum_{j=1}^{M}[\cos^{-1}\bar{\mathbf{V}}\hat{\mathbf{V}}_j]^2} \tag{3}$$

where $\bar{\mathbf{V}}$ is the average of the reconstructed velocities which was chosen to minimize $\sum_{j=1}^{M}\cos^{-1}\hat{\mathbf{V}}_j\bar{\mathbf{V}}$ subject to $||\bar{\mathbf{V}}|| = 1$. M is the number of trials.

The average rotation matrix $\bar{R}$ is computed from the estimates and the rotation sensitivity is computed as the standard deviation of the difference angle ϕ between the average rotation $\bar{R}$ and the measured rotations for each trial sample $\hat{R}$. In formula this reads :

$$\sigma_\phi = \sqrt{\frac{1}{M-1}\sum_{j=1}^{M}\phi_j^2}\ ;\ \phi_j = \cos^{-1}[\frac{Tr(\hat{R}_j\bar{R}) - 1}{2}] \tag{4}$$

3.2 Egomotion Estimation

Egomotion estimation is a 2 step process. We first estimate linear velocity with the subspace method corrected for bias reduction [5]. This algorithm, even if not among the most precise (see [4]), has a closed form solution which is advantageous for automatic reconstruction. A non-linear refinement of the solution is done in the same fashion as in [18]. By applying different algebraic manipulation the differential epipolar constraint can be written as follows:

$$\mathbf{V}^T(\mathbf{x} \times \mathbf{v}(\mathbf{x}, t)) + (\mathbf{V} \times \mathbf{x})^T(\mathbf{x} \times \mathbf{\Omega}) = 0 \tag{5}$$

From the constraint a least square estimate of the rotational velocity can be obtained as a function of $\mathbf{V}$. Substituting this rotation back a non-linear constraint on $\mathbf{V}$ is obtained. Non-linear optimization is then used to obtain the linear velocity. If no convergence is reached within a certain number of steps the estimate for the linear velocity we got from linear subspaces is used. The performance of this 2 steps technique has been extensively tested. For different motions we found convergence rate of the non-linear refinement to be approximately 100% and that usually, with exception of a few cases, the convergence is to the global minima. Local minima can be easily detected since they lie approximately at 90 degrees respect to the real value of the linear velocity [4].

3.3 $LLSE$ Structure from Motion

Structure and linear velocity can be estimated up to a scale. Let $\zeta = \alpha Z$ and $\vartheta = \alpha ||\mathbf{V}||$ we can write:

$$\mathbf{X} = \frac{\zeta}{\alpha}(\frac{X}{Z}, \frac{Y}{Z}, 1) \equiv \zeta \frac{\mathbf{x}}{\alpha} \; ; \; \mathbf{V} = \frac{\vartheta}{\alpha}\hat{\mathbf{V}} \tag{6}$$

From eq. (1) we get:

$$\dot{\zeta}\mathbf{x} + \zeta\dot{\mathbf{x}} = -\zeta(\mathbf{\Omega} \wedge \mathbf{x}) - \vartheta\hat{\mathbf{V}} \tag{7}$$

$$\tag{8}$$

Given N feature points we can rewrite the equation (8) in matrix form where the scale quantities are the unknown $\zeta = (\zeta_1, \dot{\zeta}_1 ...., \zeta_N, \dot{\zeta}_N, \vartheta)$:

$$S\zeta = 0 \tag{9}$$

where the structure matrix S is:

$$S = \begin{pmatrix} x^{(1)} & \dot{x}^{(1)} + (\mathbf{\Omega} \wedge \mathbf{x})_x^{(1)} & 0 & 0 & \dots & \hat{\mathbf{V}}_x \\ y^{(1)} & \dot{y}^{(1)} + (\mathbf{\Omega} \wedge \mathbf{x})_y^{(1)} & 0 & 0 & \dots & \hat{\mathbf{V}}_y \\ 1 & 0 + (\mathbf{\Omega} \wedge \mathbf{x})_z^{(1)} & 0 & 0 & \dots & \hat{\mathbf{V}}_z \\ 0 & 0 & x^{(2)} & \dot{x}^{(2)} + (\mathbf{\Omega} \wedge \mathbf{x})_x^{(2)} & \dots & \hat{\mathbf{V}}_x \\ 0 & 0 & y^{(2)} & \dot{y}^{(2)} + (\mathbf{\Omega} \wedge \mathbf{x})_y^{(2)} & \dots & \hat{\mathbf{V}}_y \\ 0 & 0 & 1 & 0 + (\mathbf{\Omega} \wedge \mathbf{x})_z^{(2)} & \dots & \hat{\mathbf{V}}_z \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \end{pmatrix}$$

and find the depths solving the $LLSE$ problem:

$$\begin{gathered} min\|S\zeta\|^2 \\ \|\zeta\| = 1 \end{gathered} \tag{10}$$

This is known to be the eigenvector of $M^T M$ corresponding to the smallest eigenvalue. We did extensive tests of the performance of the $LLSE$ estimate of feature depths. In figure (1) (b) we report the sensitivity on Z as a function of the error on the optical flow. It can be noticed that, even if going quickly over 100%, the error is small for limited amounts of noise. Usually, as shown in figure (3), using just two frames, good reconstructions can already be achieved. A major point is that time integration reduces drastically the initial error. This argument is discussed in the final simulations presented later.

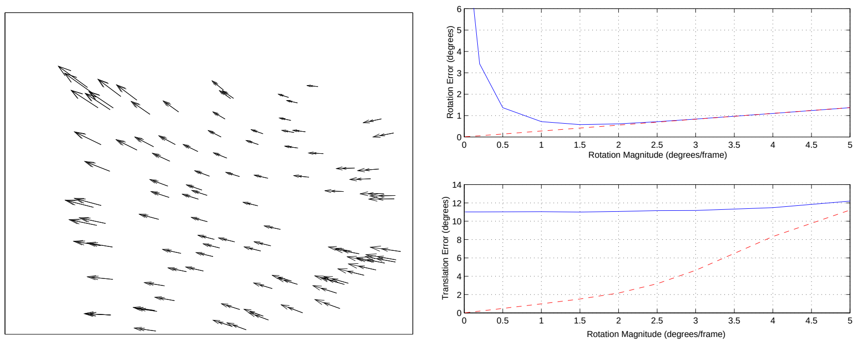

Figure 2: (**a**) Optical flow and displacement flow. Depth from point to point vary randomly between 2 and 8 focal lengths. The FOV is 90 degrees and the motion is a rotation of 0.23 degrees per frame and a translation of 0.01 focal lengths per frame. (**b**) Angular error due to the displacement approximation (dashed line) and due to displacement approximation and tracking error (continuous line). Camera internal parameters are the same as in the left figure. Translational velocity is 0.02 focal lengths per frame in the direction z. Results are similar for different motions.

3.4 Automatic Rescaling

The $LLSE$ algorithm returns depth of features with respect to the the current camera position up to a scale factor:

$$\zeta_i(t) = \alpha(t) Z_i(t) \tag{11}$$

where the time dependence expresses the fact that depths are measured with respect to different coordinate systems at each time t. Time integration requires the α to be identical at all the times. Such rescaling usually requires some kind of scene or motion knowledge: for example the modulus of the linear velocity, the distance of some point to the camera or the relative distance of two points can be used for this purpose. In the framework of the $LLSE$ algorithm rescaling can be achieved without the use of any external information.

We observe that the ratio $f(t) = \frac{\dot{\zeta}}{\zeta}$ is scale independent; if we write $f(t) = \frac{dln(\zeta)}{dt}$ and let $u = ln(\zeta)$ we get :

$$\dot{u} = f(t) \tag{12}$$
$$u(t_0) = ln(\zeta(t_0)) \tag{13}$$

This ODE can be solved to rescale all the views to the t_0 scale. We use the notation $\zeta^r(t)$ for the rescaled depths (notice that $\zeta^r(t_0) \equiv \zeta(t_0)$). We get

$$\zeta^r(t) = \zeta(t_0)e^{u(t)} = \alpha(t_0)Z(0)e^{\int_{t_0}^{t} f(s)ds} = \alpha(t_0)Z(t) \tag{14}$$

Approximating the integral by a discrete sum we can write the integral equation (14) as:

$$\zeta^r(t_j) = \zeta(t_0)\prod_{l=1}^{j} e^{f(t_l)\Delta t_l} = (\zeta(t_0)\prod_{l=1}^{j-1} e^{f(t_l)\Delta t_l})e^{f(t_j)\Delta t_j} = \zeta^r(t_{j-1})e^{f(t_j)\Delta t_j} \tag{15}$$

This is the basic equation to recursively rescale all the views to the same scaling constant: at each time t the ratio $f(t) = \frac{\dot{\zeta}}{\zeta}$ can be estimated and using eq.(14) the up-to-a-scale depth $\zeta(t)$ can be updated. Note that the global scale is still unknown and that such indeterminancy can be resolved just by scene or motion knowledge.

Partial tracks can be easily managed in this framework. Suppose that new feature, indicated by $N + 1$, is tracked at time $t = \bar{t}$ and its unrescaled depth measured as $\zeta_{N+1}(\bar{t}) = \alpha(\bar{t})Z_{N+1}(\bar{t})$. Rescaling cannot be done using equation (15) since the previous history of the track is not known. Anyway the ratio $\frac{\alpha(t_0)}{\alpha(\bar{t})}$ and the rescaled depth can be estimated as:

$$\rho = E[\frac{\zeta_i^r(\bar{t})}{\zeta_i(\bar{t})}]|_{i \neq N+1} \; \Rightarrow \; \zeta_{N+1}^r(\bar{t}) = \rho\zeta_{N+1}(\bar{t}) \tag{16}$$

At time $t > \bar{t}$ rescaling can be done also for this track using equation (15). Note that the integration of the differential equation produces a further error of the order $o((\Delta t)^{-1})$. For $30Hz$ sequences this error is pretty small compared to the measurement errors and can be trascurated. A more classical autorescaling procedure consists of fixing one of the points in the initial frame and using it to rescale all the other points [12]. The position of the fixed point in the following frames can be estimated by transforming according to the estimated inter-frame motion. The main drawback common to both these techniques is error accumulation. A big advantage of the autorescaling we propose is the fact that features are rescaled independently of each other. Fixing a particularly noisy feature can drastically reduce the effectiveness of the algorithm. Moreover the rescaling feature can be lost during then motion due to tracking inefficiencies or occlusions.

3.5 Time Integration

The easiest way is to integrate the different structure observations is to transform them in the initial frame of reference, like:

$$\hat{\mathbf{X}}^i_{t_j}(t_0) = \hat{R}^T(t_j, t_0)\hat{\mathbf{X}}^i(t_j) + \hat{\mathbf{T}}(t_j, t_0) \tag{17}$$

Where i and j are the respectively feature and frame indexes. The best spatial position of a feature is estimated from the M observations taking the $median$ of these:

$$\hat{\mathbf{X}}^i = \underset{t_j}{median} \quad (\hat{\mathbf{X}}^i_{t_j}(t_0)) \tag{18}$$

$\mathbf{T}$ and R are defined by compounding the partial motions:

$$\hat{\mathbf{T}}(t_n, t_0) = \sum_{j=0}^{n} \hat{\mathbf{T}}(t_j, t_0) \ ; \ \hat{R}(t_n, t_0) = \prod_{j=0}^{n} \hat{R}(t_j, t_0) \tag{19}$$

The overall performance of the algorithm was tested on simulated data as shown in figures (1) (b) and (c). Figure (1) (b) reports the sensitivity for reconstruction from a single frame and shows that for enough precise measurements of the optical flow the error on depth estimation can be kept small. In figure (3) we show that reasonable reconstructions can be obtained using a single flow field. Figure (1) (c) shows how time integration improves the estimates. The slope of the curve is small due to the correlation between measurements at different times produced by rescaling. Anyway, due to the high acquisition rate, hundreds of frames can be integrated reaching very high precisions. In figure (3) we show the improvement we get time integrating 10 consecutive frames of the sequence.

3.6 Instantaneous Approximation

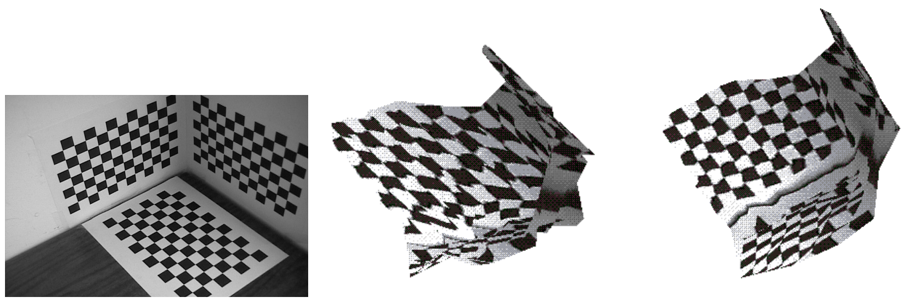

Figure 3: Reconstructed model of the calibration grid using 2 frames (left) and 10 frames (right). The model is based on 237 features and the average disparity between frames is about 1 pixel

Optical flow is approximated with the displacement of features between two consecutive frames. It is interesting to see how such approximation is good for human motion and when otherwise it does lead to serious errors. The displacement field is generated by the discrete motion equation :

$$\mathbf{X}(t+1) = -R(\mathbf{\Omega}\Delta t)\mathbf{X}(t) - \mathbf{T} \tag{20}$$

Where $R(\mathbf{\Omega}\Delta t) = e^{\mathbf{\Omega}\Delta t}$. For small Δt this can be rewritten as:

$$\mathbf{X}(t+1) \cong -(I + \mathbf{\Omega}\Delta t)\mathbf{X}(t) - \mathbf{V}\Delta t \tag{21}$$

The displacement field is easily found to be :

$$\frac{\mathbf{x}(t+1) - \mathbf{x}(t)}{\Delta t} = \frac{\mathbf{v}(x,y,t)}{1 + Z^{-1}(\Omega_x Y - \Omega_y X)\Delta t + Z^{-1}V_z\Delta t} \tag{22}$$

Equation (22) states that even in the case of smooth motion the displacement field can be quite different from the optical flow. To have a satisfactory approximation we must impose that the denominator of eq.(22) is close to 1. Note that the two terms $Z^{-1}(\Omega_x Y - \Omega_y X)\Delta t$ and $Z^{-1}V_z\Delta t$ represent the motion of the camera along the optical axis, generated respectively by rotation and translation, relative to the distance of the scene from the camera. We finally get the following set of conditions to be satisfied:

$$\dot{\mathbf{\Omega}}\Delta^2 t \ll \mathbf{\Omega}\Delta t \,;\, \dot{\mathbf{V}}\Delta^2 t \ll \mathbf{V}\Delta t \tag{23}$$

$$\Omega_x X\Delta t \ll Z \,;\, \Omega_y Y\Delta t \ll Z \,;\, V_z\Delta t \ll Z \tag{24}$$

The first two implies that the motion must be smooth, so rapid changes in the velocities are not allowed. The last three express the fact that the motion along Z must be small compared to the distance of the object from the camera. Figure (2) shows how egomotion estimation behaves when the instantaneous approximation breaks down. Synthetic fields were generated and linear and rotational velocities estimated with the technique described above. Both the case of noise-free and noisy tracking are shown. Clearly, when noise is present, a too dense time sampling produces huge errors due to the fact that the features displacement becomes smaller while the tracking error in general is lower bounded. Observe that a rotation of 90 degrees per second (pretty fast for a video amateur exploring some new environment!) corresponds, at $30Hz$, to $\mathbf{\Omega}\Delta t \cong 5\%$. So we can conclude that approximating the optical flow with the displacement field is safe in the case of human motion.

4 Experiments

We performed extensive tests over real sequences acquired by a hand held digital camera. The camera was previously calibrated using different views of a planar grid (one of the planes in figure (3) was used). Sparse features were tracked using the Kanade-Lucas-Tomasi tracker [3] and optical flow approximated with features displacement. In figures (3),(4) and (5) we show examples of the reconstructions obtained with the algorithm presented in the paper. We would like to stress that precise calibrations can be obtained with the technique we used even with a small number of images and that if the focal length is not changed calibration is well preserved over time: we used the camera for about one month recalibrating regularly and checked that the camera parameters were always consistent with the initial calibration. This is an important observation since, in the differential framework, full auto calibration of the camera cannot be performed, so at least partial initial knowledge of the camera parameters is necessary [7].

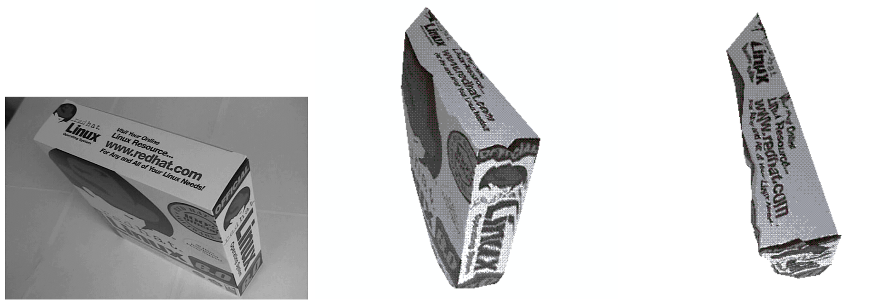

Figure 4: Reconstructed model of a Linux box based on 11 frames and 220 tracks. The average disparity is about 0.9 pixels per frame

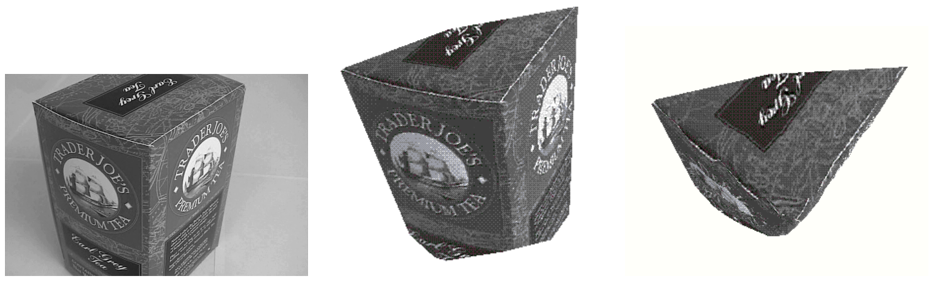

Figure 5: Reconstructed model of a teabox based on 7 frames and 250 tracks. The average disparity is about 1.1 pixels per frame.

5 Conclusions and Future Work

In this paper a recursive algorithm to estimate the structure of a scene from a calibrated video sequence was presented. The algorithm is based on the extraction of optical flow for each frame and the sequential computation of motion and structure. Moreover an alternative new way to automatically rescale the 3D information from different views is presented and tested. The two stages of egomotion and structure computation are approached with linear procedures : this makes our algorithm fast enough to be run in real time. Results on simulated data and real images are presented to validate the effectiveness of the approach.
A major weakness of the algorithm is the time integration that at the moment is performed with very rough procedure. For the near future we intend to embed the one-view structure algorithm in a Kalman filtering framework. Constraints over non consecutive views are another issue to be explored to reduce drift and get a better overall consistence of the reconstruction.

Acknowledgments

The sequences used for this paper have been shot during my visit at George Mason University, Fairfax, VA. I would like to thank Jana Kosecka for hosting me, making the equipment available and for the numerous and interesting suggestions and the foundation Ludovisi-Blanceflor-Bomcompagni for the economical support. I would like also to thank the EC for supporting my Ph.D through the TMR project $CAMERA$ (Project ERB FMRX-CT97-0127)).

References

[1] Poggio T. and Verri A. Motion field and optical flow:qualitative properties. *PAMI*, 2(5):490–498, 1989.

[2] Fleet D. J. Barron J. L. and Beuchemein S. S. Performance of optical flow techniques. *IJCV*, 12(1):43–77, 1994.

[3] Tomasi C. and Kanade T. Detection and tracking of point features. Technical Report TR CMU CS 91 132, CMU, 1991.

[4] Tomasi C. and Heeger D. J. Comparison of approaches to egomotion computation. In *CVPR*, pages 315–320, 1994.

[5] MacLean W. J. Removal of translational bias when using subspace methods. In *ICCV*, pages 753–758, 1999.

[6] Heeger D. J. and Jepson A. D. Subspace methods for recovering rigid motion. *IJCV*, 7(2):95–117, 1992.

[7] Kosecka J. Ma Y. and Sastry S. Linear differential algorithm for motion recovery:a geometric approach. *IJCV*, 36(1):71–89, 2000.

[8] Szeliski R. and Kang S.B. Recovering 3d shape and motion from image streams using non-linear least squares. *JVCIR*, 5(1):10–28, 1994.

[9] Jepson A. D Barron J. L and Tsotsos J. K. The feasibility of motion and structure from noisy time varying image velocity information. *IJCV*, 5(3):239–269, 1990.

[10] Soatto S. and Perona P. Reducing “structure from motion” part 1: modeling. *PAMI*, 20(9):933–942, 1998.

[11] Soatto S. and Perona P. Reducing “structure from motion” part 2: experimental evaluation. *PAMI*, 20(9):943–961, 1998.

[12] Azarbayejani A. and Pentland A.P. Recursive estimation of motion, structure, and focal length. *PAMI*, 17(6):562–575, 1995.

[13] Kosecka J. Ma Y. and Sastry S.S. Motion recovery from image sequences: Discrete viewpoint vs. differential viewpoint. In *ECCV*, 1998.

[14] Harteley R. and Zisserman A. *Multiple View Geometry in Computer Vision*. Cambridge, 2000.

[15] Nister D. *Automatic Dense Reconstruction from Uncalibrated Video Sequences.* PhD thesis, Royal Institute of Technology, 2001.

[16] Trucco E. and Verri A. *Introductory Techniques for 3D Computer Vision*. Prentice Hall, 1998.

[17] Kanatani K. *Geometric Computation for Machine Vision*. Clarendon Press, 1993.

[18] Bruss A.R. and Horn B.K.P. Passive navigation. *CVGIP*, 21:3–20, 1983.