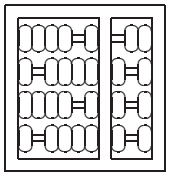


INSTITUTO DE COMPUTAÇÃO
UNIVERSIDADE ESTADUAL DE CAMPINAS

Robust Estimation of Camera Motion using Local Invariant Features

Jurandy Almeida *Rodrigo Minetto*
Tiago A. Almeida *Ricardo da S. Torres*
Neucimar J. Leite

Technical Report - IC-09-12 - Relatório Técnico

April - 2009 - Abril

Robust Estimation of Camera Motion using Local Invariant Features

Jurandy Almeida[1] Rodrigo Minetto[1] Tiago A. Almeida[2]
Ricardo da S. Torres[1] Neucimar J. Leite[1]

[1] Institute of Computing
University of Campinas
13083-970 – Campinas – SP – Brazil
{jurandy.almeida,rodrigo.minetto,rtorres,neucimar}@ic.unicamp.br

[2]School of Electrical and Computer Engineering
University of Campinas
13083-970 – Campinas – SP – Brazil
tiago@dt.fee.unicamp.br

April 16, 2009

Abstract

Most of existing techniques to estimate camera motion are based on analysis of the optical flow. However, the estimation of the optical flow supports only a limited amount of scene motion. In this report, we present a novel approach to estimate camera motion based on analysis of local invariant features. Such features are robust across a substantial range of affine distortion. Experiments on synthesized video clips with a fully controlled environment show that our technique is more effective than the optical flow-based approaches for estimating camera motion with a large amount of scene motion.

1 Introduction

The estimation of camera motion is one of the most important aspects for video processing, analysis, indexing, and retrieval. Most of existing techniques to estimate camera motion are based on analysis of the optical flow [1, 2, 3]. However, the estimation of the optical flow, which is usually based on gradient or block matching methods, supports only a limited amount of scene motion.

To address this problem, we present a novel approach for the estimation of camera motion with a large amount of scene motion. Our technique relies on analysis of local invariant features obtained from extrema in the scale space rather than on analysis of the optical flow. Such features are robust across a substantial range of affine distortion.

In order to validate our approach, we use synthetic videos sequences based on POV-Ray scenes including all kinds of camera motion and many of their possible combinations. The main advantage of such a synthetic test set is that the camera motion parameters can be fully controlled. Further, we have conducted several experiments to show that our technique is more effective than the optical flow-based ones for estimating camera motion with a large amount of scene motion.

The remainder of the report is organized as follows. Section 2 presents our approach for the estimation of camera motion. The experimental settings and results are discussed in Section 3. Finally, Section 4 presents conclusions and directions for future work.

2 Our Approach

In presence of a substantial range of affine distortion, the methods for estimating camera motion by analysis of the optical flow can fail [3]. To address this problem, we present a novel approach for the estimation of camera motion based on the analysis of the local invariant features. It consists of three main steps: (1) feature matching; (2) motion model fitting; and (3) robust estimation of the camera parameters.

2.1 Feature Matching

The first step for estimating camera motion in video sequences is to extract and match features between consecutive frames. Here, we use a framework to detect and describe local invariant features in images, called *Scale Invariant Features Transform* (SIFT) [4]. This approach is composed by four major stages: (1) scale-space peak selection; (2) keypoint localization; (3) orientation assignment; and (4) keypoint description.

The scale-invariant features are efficiently identified by using a staged filtering approach. The first stage identifies key locations in scale space by looking for locations that are maxima or minima of a difference-of-Gaussian function. Next, for each candidate keypoint, interpolation of nearby data is used to accurately determine its position. Moreover, this information allows to reject candidate keypoints that have low contrast or are poorly localized along an edge. Thereafter, it identifies the dominant orientations for each keypoint using local image gradient directions. Finally, the method builds a local descriptor for each keypoint based on the image gradients in its local neighborhood. To match keypoints from two images, we use the Euclidean distance between the local descriptors. To ensure correct match, the ratio of the distance for the best match and the second best match must be less than 0.6 [4].

2.2 Motion Model Fitting

A camera projects a 3D world point into a 2D image point. The motion of the camera may be limited to a single motion such as rotation, translation, or zoom, or some combination of these three motions. Such camera motion can be well categorized by few parameters.

In our case, we use a two-dimensional affine model to estimate a parametric form for describing the displacement of the video frame content from the correspondence between

local invariant features. The affine model was employed in the following considerations. First, the affine model is more resilient to noisy data. In addition, it can represent all of the basic camera motions often used in video indexing.

If we denote the position in the first image by (x, y) and the corresponding position in the second image by $(\hat{x}, \hat{y})$, we can formulate the two-dimensional affine motion model as

$$\begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} = \begin{bmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix},$$

where $\{a_{ik}\}$, t_x, and t_y are the motion parameters.

The parameter-estimation problem consists in finding a good estimate of the six parameters $(\{a_{ik}\}, t_x, t_y)$ based on a set of measured point correspondences. We denote a set of points in the first image as $\{(x_i, y_i)\}$ and their corresponding points in the second image as $\{(\hat{x}_i, \hat{y}_i)\}$. Since the point measurements are not exact, we cannot assume that they will all fit perfectly to the motion model. Hence, the best solution is to compute a least-squares fit to the data. We consequently define the model error E as the sum of squared distances between the measured positions $(\hat{x}_i, \hat{y}_i)$ and the positions obtained from the motion model:

$$E = \sum_i ((a_{00}x_i + a_{01}y_i + t_x) - \hat{x}_i)^2 + ((a_{10}x_i + a_{11}y_i + t_y) - \hat{y}_i)^2. \tag{1}$$

To minimize the model error E, we can take its partial derivatives with respect to the model parameters $(\{a_{ik}\}, t_x, t_y)$ and set them to zero. This gives the equation system

$$\begin{bmatrix} \sum_i x_i^2 & \sum_i x_i y_i & \sum_i x_i & 0 & 0 & 0 \\ \sum_i x_i y_i & \sum_i y_i^2 & \sum_i y_i & 0 & 0 & 0 \\ \sum_i x_i & \sum_i y_i & \sum_i 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sum_i x_i^2 & \sum_i x_i y_i & \sum_i x_i \\ 0 & 0 & 0 & \sum_i x_i y_i & \sum_i y_i^2 & \sum_i y_i \\ 0 & 0 & 0 & \sum_i x_i & \sum_i y_i & \sum_i 1 \end{bmatrix} \begin{pmatrix} a_{00} \\ a_{01} \\ t_x \\ a_{10} \\ a_{11} \\ t_y \end{pmatrix} = \begin{pmatrix} \sum_i \hat{x}_i x_i \\ \sum_i \hat{x}_i y_i \\ \sum_i \hat{x}_i \\ \sum_i \hat{y}_i x_i \\ \sum_i \hat{y}_i y_i \\ \sum_i \hat{y}_i \end{pmatrix} \tag{2}$$

which can be solved more easily by splitting the equation system into two independent systems.

Finally, we can express the estimated parameters in another form more directly related to the physically meaningful camera motion, as follows:

$$pan = t_x, tilt = t_y, div = \frac{1}{2}(a_{00} + a_{11}), rot = \frac{1}{2}(a_{10} - a_{01}),$$

where the terms *pan*, *tilt*, *div*, and *rot* represent the motion induced by the camera operations of panning (or tracking), tilting (or booming), zooming (or dollying), and rolling, respectively.

2.3 Robust Estimation of the Camera Parameters

The direct least-squares approach for parameter estimation works well for a small number of outliers that do not deviate too much from the correct motion. However, the result is

significantly distorted when the number of outliers is larger, or the motion is very different from the correct camera motion. Especially if the video sequence shows independent object motions, a least-squares fit to the complete data would try to include all visible object motions into a single motion model.

To reduce the influence of outliers, we apply a well-known robust estimation technique called RANSAC (RANdom SAmple Consensus) [5]. The idea is to repeatedly guess a set of model parameters using small subsets of data that are drawn randomly from the input. The hope is to draw a subset with samples that are part of the same motion model. After each subset draw, the motion parameters for this subset are determined and the amount of input data that is consistent with these parameters is counted. The set of model parameters with the largest support of input data is considered the most dominant motion model visible in the image.

3 Experiments and Results

In order to evaluate our approach, we create a synthetic test set with four MPEG-4 video clips[1] (640 × 480 pixels of resolution) based on well textured POV-Ray scenes of a realistic office model (Figure 1) including all kinds of camera motion and many of their possible combinations. The main advantage is that the camera motion parameters can be fully controlled which allows us to verify the estimation quality in a reliable way.

(a) (b) (c)

Figure 1: The POV-Ray scenes of a realistic office model used in our synthetic test set.

The first step to create the synthetic videos is to define the camera's position and orientation in relation to the scene. The world-to-camera mapping is a rigid transformation that takes scene coordinates $p_{\mathrm{w}} = (x_{\mathrm{w}}, y_{\mathrm{w}}, z_{\mathrm{w}})$ of a point to its camera coordinates $p_{\mathrm{c}} = (x_{\mathrm{c}}, y_{\mathrm{c}}, z_{\mathrm{c}})$. This mapping is given by [6]

$$p_{\mathrm{c}} = Rp_{\mathrm{w}} + T, \tag{3}$$

where R is a 3×3 rotation matrix that defines the camera's orientation, and T defines the camera's position.

[1] All video clips and ground truth data of our synthetic test set are available at `http://www.liv.ic.unicamp.br/~minetto/videos/`.

The rotation matrix R is formed by a composition of three special orthogonal matrices (known as *rotation matrices*)

$$R_x = \begin{bmatrix} \cos(\alpha) & 0 & -\sin(\alpha) \\ 0 & 1 & 0 \\ \sin(\alpha) & 0 & \cos(\alpha) \end{bmatrix} R_y = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\beta) & \sin(\beta) \\ 0 & -\sin(\beta) & \cos(\beta) \end{bmatrix} R_z = \begin{bmatrix} \cos(\gamma) & \sin(\gamma) & 0 \\ -\sin(\gamma) & \cos(\gamma) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

where α, β, γ are the angles of the rotations.

We consider the motion of a continuously moving camera as a trajectory where the matrices R and T change according to the time t, in homogeneous representation,

$$\begin{bmatrix} p_{\mathrm{c}} \\ 1 \end{bmatrix} = \begin{bmatrix} R(t) & T(t) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} p_{\mathrm{w}} \\ 1 \end{bmatrix}. \tag{4}$$

Thus, to perform camera motions such as tilting (gradual changes in R_x), panning (gradual changes in R_y), rolling (gradual changes in R_z), and zooming (gradual changes in focal distance f), we define a function $F(t)$ which returns the parameters α, β, γ, and f used to move the camera at the time t. We use a smooth and cyclical function

$$F(t) = \mathcal{M} * \frac{1 - \cos(2\pi t/\mathcal{T})(0.5 - t/\mathcal{T})}{0.263}, \tag{5}$$

where $\mathcal{M}$ is the maximum motion factor and $\mathcal{T}$ is the duration of camera motion in units of time. We create all video clips using the maximum motion factor $\mathcal{M}$ equals to 3^o for tilting (α), 8^o for panning (β), 90^o for rolling (γ), and 1.5 for zooming (f).

Figure 2 shows the main characteristics of each resulting video sequence M_i. The terms P, T, R, Z stand for the motion induced by the camera operations of panning, tilting, zooming, and rolling, respectively. The videos M_3 and M_4 have combinations of two or three types of camera motions. To represent a more realistic scenario, we modify the videos M_2 and M_4 to have occlusions due to object motion.

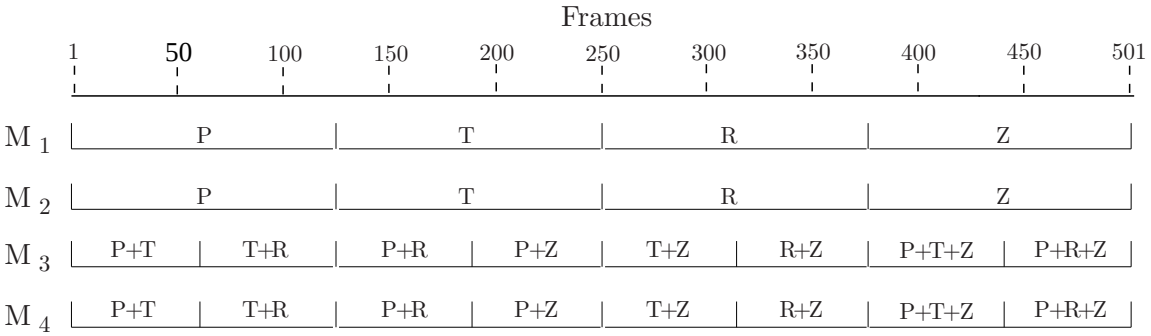


Figure 2: The main characteristics of each video sequence (M_i) in our synthetic test set.

We assess the effectiveness of the proposed method using the well-known Zero-mean Normalized Cross Correlation (ZNCC) metric [7], defined by

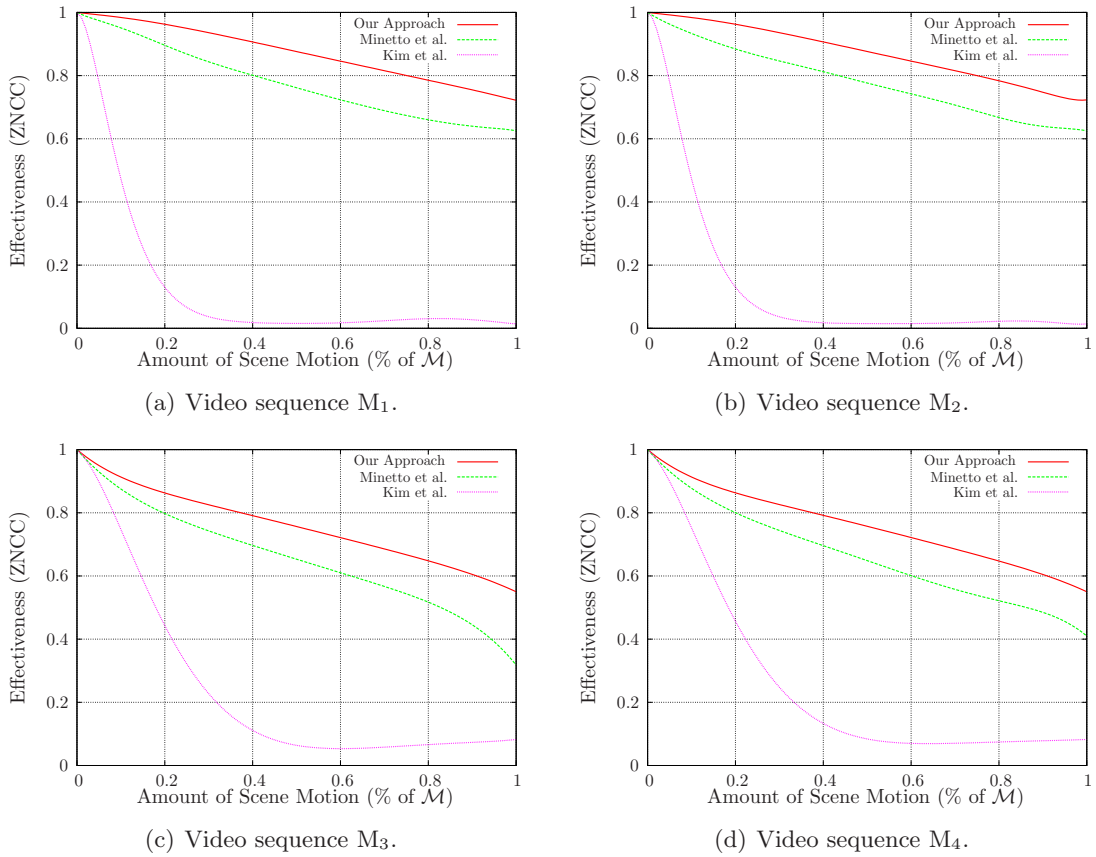
$$\mathrm{ZNCC}(\mathcal{F}, \mathcal{G}) = \frac{\sum_t (\mathcal{F}(t) - \bar{\mathcal{F}})(\mathcal{G}(t) - \bar{\mathcal{G}})}{\sqrt{\sum_t (\mathcal{F}(t) - \bar{\mathcal{F}})^2 \sum_t (\mathcal{F}(t) - \bar{\mathcal{G}})^2}} \tag{6}$$

where $\mathcal{F}(t)$ and $\mathcal{G}(t)$ are the estimate and the real camera parameters, respectively, at the time t. It returns a real value between -1 and $+1$. A value equals to $+1$ indicates a perfect estimation; and -1, an inverse estimation.

We compare our approach with the techniques proposed by Kim et al. [8] and Minetto et al. [3]. The former estimates camera motion by using a least-squares fit to the motion vectors extrated from MPEG bitstream. The latter uses a weighted least-squares fit to the optical flow computed by using the well-known Kanade-Lucas-Tomasi (KLT) algorithm [9].

The purpose of our experiments is to evaluate the effectiveness of different approaches in estimating camera motion on a substantial range of affine distortion. We can vary the amount of scene motion by using a sampling rate. Thus, we estimate camera motion between temporally sparse frames.

Figure 3 shows the effectiveness achieved by all approaches on varying the amount of scene motion by a proportion of the maximum motion factor $\mathcal{M}$. Table 1 presents the average time spent to estimate camera motion between two video frames. We performed all experiments on Intel Core 2 Quad Q6600 (four cores running at 2.4 GHz), 2GB memory DDR3.



(a) Video sequence M_1. (b) Video sequence M_2.

(c) Video sequence M_3. (d) Video sequence M_4.

Figure 3: Effectiveness achieved by all approaches on varying the amount of scene motion.

Table 1: Average time spent to estimate camera motion between two video frames.

Method	Our Approach	Minetto et al.	Kim et al.
Average Time (s)	0.234	0.423	0.006

In fact, the use of local invariant features for estimating camera motion with a substantial range of affine distortion is more effective than the optical flow-based approaches. Moreover, our approach is almost two times faster than the KLT-based method [3] to estimate camera motion between two video frames.

Despite of the high computational efficiency presented by techniques based on MPEG motion vectors, they support only a very small amount of scene motion. In addition, they cannot be applied on all video formats neither for estimating camera motion in real-time applications.

In order to show that our technique is suitable for real-time applications, we implement a video player able to characterize different types of camera motions at the playing time[2]. Figure 4 presents screenshots of our video player. The showed frames belong to video recordings of a meeting of the Board of Trustees of the University of Campinas. On the top left corner of each frame, there is a tag indicating which kind of camera motion is being identified.

4 Conclusions

In this report, we present a novel approach to estimate camera motion based on analysis of local invariant features. Such features are robust across a substantial range of affine distortion.

We have provided several experiments showing that our technique is more effective than two baselines (one based on the KLT algorithm [3] and other based on MPEG motion vectors [8]) for estimating camera motion with a large amount of scene motion. Furthermore, we show that our approach is suitable for real-time applications.

Future work includes the evaluation of other interest point detectors, motion models, and robust estimation techniques. In addition, we want to investigate the effects of embedding the proposed method into video recording devices for real-time applications.

Acknowledgments

The authors thank the financial support of Microsoft EScience Project, CAPES/COFECUB Project (Grant 592/08), and Brazilian agencies FAPESP (Grants 07/54201-6 and 08/50837-6), CNPq (Grant 142466/2006-9), and CAPES (Grant 01P-05866/2007).

[2]The compiled binaries for Linux on Intel compatible processors are available at `http://www.liv.ic.unicamp.br/~jurandy/pub/mcplayer.tar.gz`.

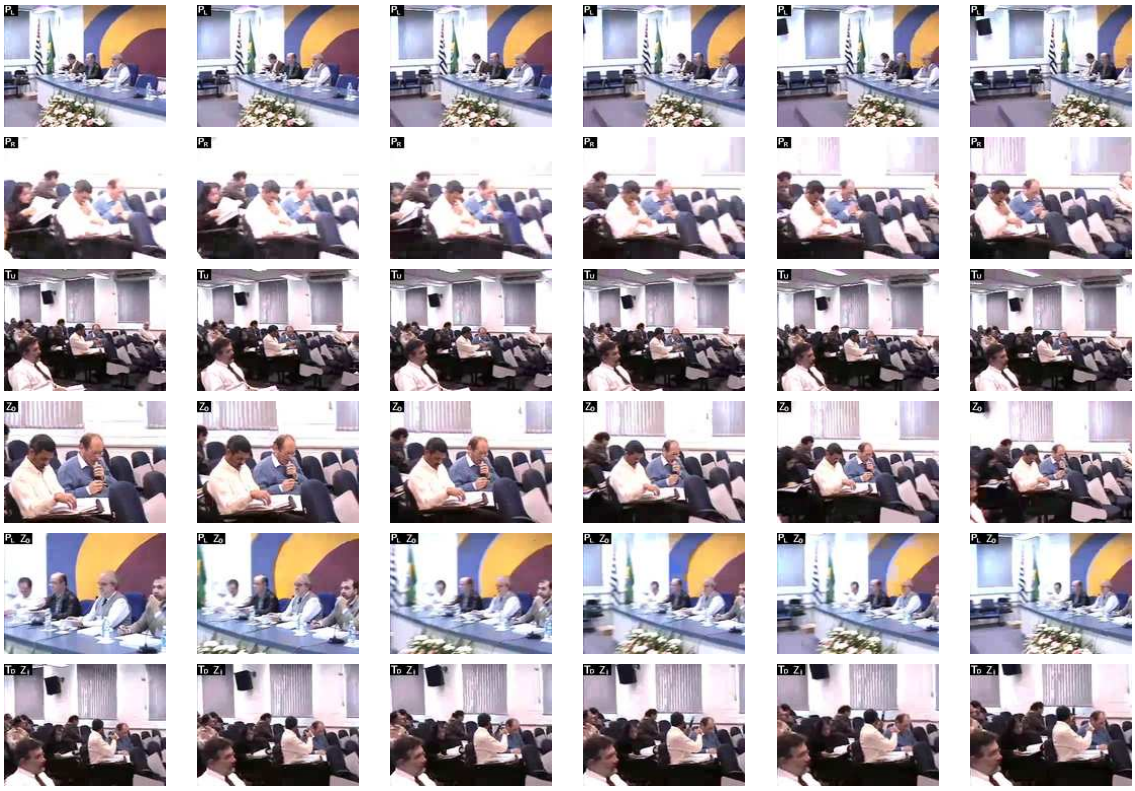


Figure 4: Screenshots of our video player. Each row is associated to a kind of camera motion: from top to bottom, we have panning to left, panning to right, tilting up, zooming out, panning to right with zooming out, and tilting down with zooming in.

References

[1] Srinivasan, M.V., Venkatesh, S., Hosie, R.: Qualitative estimation of camera motion parameters from video sequences. Pattern Recognition **30**(4) (1997) 593–606

[2] Park, S.C., Lee, H.S., Lee, S.W.: Qualitative estimation of camera motion parameters from the linear composition of optical flow. Pattern Recognition **37**(4) (2004) 767–779

[3] Minetto, R., Leite, N.J., Stolfi, J.: Reliable detection of camera motion based on weighted optical flow fitting. In: VISAPP (2007) 435–440

[4] Lowe, D.G.: Distinctive image features from scale-invariant keypoints. IJCV **60**(2) (2004) 91–110

[5] Fischler, M.A., Bolles, R.C.: Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM **24**(6) (1981) 381–395

[6] Ma, Y., Soatto, S., Kosecka, J., Sastry, S.S.: An Invitation to 3-D Vision: From Images to Geometric Models. Springer-Verlag (2003)

[7] Martin, J., Crowley, J.L.: Experimental comparison of correlation techniques. In: Int. Conf. on Intelligent Autonomous Systems (1995)

[8] Kim, J.G., Chang, H.S., Kim, J., Kim, H.M.: Efficient camera motion characterization for mpeg video indexing. In: ICME (2000) 1171–1174

[9] Shi, J., Tomasi, C.: Good features to track. In: CVPR (1994) 593–600