Camera Motion Estimation Using Monocular and Stereo-Vision

Conference Paper · September 2008

DOI: 10.1109/ICCP.2008.4648385 · Source: IEEE Xplore

2 authors:

Silviu Bota
Google Switzerland GmbH
17 PUBLICATIONS **257** CITATIONS
SEE PROFILE

Sergiu Nedevschi
Universitatea Tehnica Cluj-Napoca
356 PUBLICATIONS **3,449** CITATIONS
SEE PROFILE

Camera Motion Estimation Using Monocular and Stereo-Vision

Silviu Bota and Sergiu Nedevschi
Technical University of Cluj-Napoca
{Silviu.Bota, Sergiu.Nedevschi}cs.utcluj.ro

Abstract

A preliminary step for many computer vision algorithms is the estimation of camera motion. In this paper we describe and compare the results of three algorithms for camera motion estimation, which require various degrees of knowledge about the camera(s) used and the scenario. The first algorithm uses a single uncalibrated camera and it is useful for applications such as content based image retrieval. This method provides only qualitative results. The second algorithm uses a single calibrated camera in order to obtain quantitative results. The third approach uses a calibrated stereo camera pair, together with a hardware stereo reconstruction system.

1. Introduction

This paper concentrates on two problems which require camera and object motion computation. The first problem is in the domain of content based image retrieval, specifically in automated motion picture annotation. The requirements are to detect various camera motions which are specific to motion picture filming techniques, such as panning, zooming, new shot etc. The requirements for this problem are to qualitatively determine the camera motion. Because digitally encoded movies are used, we have no knowledge about camera parameters or about the scene, so the solution for this problem must be restricted to using only 2D information.

The second problem is in the domain of driving assistance systems. It is often important to determine the ego vehicle motion relative to the fixed scene. Usually, there are ways to obtain this information directly, from odometry and yaw rate sensors. However, odometry cannot be always accurate in case some of the ego vehicle's wheels are slipping. Accurate yaw rate sensors tend to be expensive. Using a mono or stereo camera system, which are passive sensors provides the means to measure the ego vehicle motion without any additional sensors . The ego vehicle motion information is useful for many algorithms involved in driving assistance systems, such as object tracking, lane detection and tracking.

In all of the algorithms described here, we use the optical flow algorithm described in [1] which provides a sparse, 2D optical flow field. The optical flow is computed only on corner points, which provide the most accurate results. From this 2D field we obtain our 3D results, using the information available in each case.

2. Camera Model

Our vehicle and camera reference frames have the x axis pointing to the left, the y axis pointing down, and the z axis pointing away from the viewer. The origin of the vehicle reference frame is a point on the road, below the middle of the front bumper of the ego vehicle. The origin of the camera reference frame is the optical center of the camera. The image reference frame has its origin in the top left corner, the x axis pointing left and the y axis pointing down.

3. The Uncalibrated Mono Camera Case

In the current implementation we chose to estimate the following camera motion types: pan left, pan right, pan up, pan down (all by the rotation of the camera around the optical center) and zoom in and zoom out (by altering the focal length). Because we don't really know any of the camera parameters, we make the following simplifying assumptions: we consider the principal point to be in the image center ($c_x = \text{width}/2$ and $c_y = \text{height}/2$ and the pixels square ($f_x = f_y$), because cameras are usually manufactured in such a way as to ensure these. To simplify the notations, we make the substitutions $x \leftarrow x - c_x$ and $y \rightarrow y - c_y$. We therefore have:

$$x = \frac{fX}{Z} \; ; \; y = \frac{fY}{Z} \tag{1}$$

with $P_c = (\ X \quad Y \quad Z\ )^T$. It is convenient at this time to express the 3D points in the camera's reference frame in polar coordinates. This is because projection happens along optical rays, and all the points on a given ray are projected onto the same image point. The following equations describe this transform:

$$X = R\cos\beta\sin\alpha \; ; \; Y = R\sin\beta \; ; \; Z = R\cos\beta\cos\alpha \tag{2}$$

In the above equations R represents the 3D point's distance from the camera's optical center, α represents the horizontal angle made by the optical ray with the yOz plane and β the angle made by the optical ray with the xOz plane. With these transforms, the projection equations become:

$$x = \frac{fR\cos\beta\sin\alpha}{R\cos\beta\cos\alpha} = f\tan\alpha \tag{3}$$

$$y = \frac{fR\sin\beta}{R\cos\beta\cos\alpha} = f\frac{\tan\beta}{\cos\alpha} \tag{4}$$

The camera motion detection algorithm is based on the results obtained by optical flow computation. The optical flow estimates a number of correspondent corner point pairs, in the previous and next frames. Let's call these points $\begin{pmatrix} x_0 & y_0 \end{pmatrix}^T$ and $\begin{pmatrix} x_1 & y_1 \end{pmatrix}^T$. We have:

$$x_1 = x_0 + \Delta x\,;\, y_1 = y_0 + \Delta y \tag{5}$$

We also consider that the displacement of each point is caused by the motion of the camera (the point is a background point, or an object which is fixed relative to the world's (scene's) coordinates system. We have the following parameters which capture the camera's motion: $\Delta\alpha = \alpha_1 - \alpha_0$ is the horizontal panning angle (between 2 subsequent frames) $\Delta\beta = \beta_1 - \beta_0$ is the vertical panning angle and $\Delta f = f_1 - f_0$ is the focal length increase or decrease.For the points in the second frame we have:

$$x_0 + \Delta x = (f_0 + \Delta f)\tan(\alpha_0 + \Delta\alpha) \tag{6}$$

$$y_0 + \Delta y = (f_0 + \Delta f)\frac{\tan(\beta_0 + \Delta\beta)}{\cos(\alpha_0 + \Delta\alpha)} \tag{7}$$

These equations are obviously non-linear, and thus are difficult to solve. Therefore, we approximate the trigonometric functions using a first-order Taylor expansion (around the old angles and focal length values). This approximation is valid, as we expect small displacements (the frame-rate in motion pictures is usually higher than 24 fps). We arbitrarily take f_0 to correspond to a typical field of view of 40 degrees.

We use the RANSAC [9] method. We randomly select 10 points at each iteration and solve the equations using least squares. We repeat this until we find a model with a high percentage of inliers (60 percent in our tests) or until we exceed a maximum number of iteration (50 in our tests). If no model can be found, then we assume that a new camera shot has begun and signal this.

If a model is found, then we determine $\Delta\alpha$, $\Delta\beta$ and Δf. We filter these values across multiple frames to obtain more stable results. Some experimental results are presented in section 6

4. The Calibrated Mono Camera Case

In this section we describe an algorithm that is useful in the context of vision based driving assistance systems. We start by applying the method described in the previous section, but, this time, c_x, c_y, f_x and f_y are know. Therefore we can recover $\Delta\alpha$ (the instant yaw angle) and $\Delta\beta$ (the instant pitch angle) exactly (within the limits of the first order approximation we made). This time we know that Δf is zero, because the focal length is fixed. However, we allow this free parameter, as a approximation for the zoom-in effect caused by the ego vehicle motion.

The next task is to determine the speed of the ego vehicle. For this we make a few assumptions. First, we discard the translation vector from the world to camera transform equations. This is, of course, an approximation, because the speed is not the same in this new coordinate system, but for all practical purposes, this approximation is a very good one.

Second, we assume that the ego vehicle moves only along the Z axis. This is also a good approximation. Our camera may have its optical axis non-parallel with the Z axis (usually, camera is positioned in such a way as to provide optimal results for specific applications, so we cannot assume a default configuration). To compensate for this, we transform the points for which we compute the optical flow by rotating and translating them using the camera's parameters

Unfortunately, it is still impossible to determine the speed on the Z axis, because we lack 3D information. This is why, some previous knowledge about the scene must be incorporated in the algorithm. This could either be the distance to some object or the dimension of some object. In the spirit of the monocular camera approach, we provided only the 3D distance between pairs of two points (such as the height of a pole, or tree, or the width of a stationary vehicle. We manually provided these distances, but they could be provided as a result of a monocular lane detection algorithm or object recognition algorithm. In either case, there are uncertainties associated with these distances, because 3D information is not available. In section 6 we provide some results of this algorithm.

5. The Calibrated Stereo Camera Case

In this section we describe an algorithm for ego vehicle motion estimation, useful in the context of stereo vision based driving assistance systems, which is able to provide highly accurate results for the pitch, yaw and roll angles variation and the speed of the ego vehicle. The algorithm makes full usage of the available 3D data, as supplied by a dense stereo reconstruction hardware system (described in [10]). The algorithm is integrated into a complete stereo

vision based driving assistance system, described in [11] and uses the results of the lane detection and object detection modules. The stereo reconstruction hardware provides 3D coordinates for some 2D image pixels, allowing us to perform all the computations in 3D. The lane and object detection algorithms provide flags for 3D points, such as "below road", "above road", "road point", "beyond driving area", "object point" etc. This allows us to select only points which belong to fixed scene elements ("road points", "beyond driving area")

3D points belonging to a fixed elements in the scene undergo a rigid 3D transformation from one frame to the next, caused by the motion of the ego vehicle. The general transformation is given by:

$$P_1 = RP_0 + T \tag{8}$$

Because we are determining the motion of a vehicle, we restrict ourselves to the case where the translation occurs only along the Z axis, so that $T = (\ 0 \quad 0 \quad \Delta Z\ )^T$. With this restrictions the results become much better. However, because of errors in 3D reconstruction, caused by erroneous stereo matching, and amplified by the least squares procedure, the results for the translation component along the z axis (ΔZ) are still extremely large.

Therefore, we took a two step approach. In the first step we computed ΔZ as the average displacement along the z axis for all the 3D point pairs:

$$\Delta Z = \frac{1}{N}\sum_{i=1}^{N} Z_1^i - Z_0^i \tag{9}$$

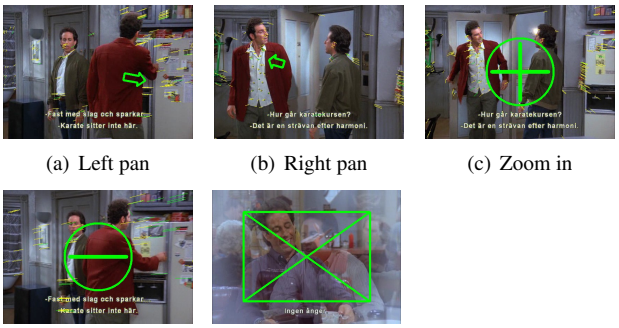
This estimation for ΔZ is much less sensitive to noise, because it uses differences instead of squared differences. In the second step we solved (using least squares) the systems:

$$X_1^i = r_1 X_0^i\,;\; Y_1^i = r_2 Y_0^i\,;\; i = \overline{1,N} \tag{10}$$

where r_1 and r_2 are the first two rows of the rotation matrix. Because the rows of a rotation matrix are orthonormal vectors, we normalized these to unit length and obtained $r_3 = r_1 \times r_2$. From the rotation matrix we obtain the pitch, yaw and roll angles.

6. Experimental Results

The first algorithm, for obtaining camera motion from uncalibrated mono camera, we tested our system using manually labeled digitally encoded motion pictures, with a total length of roughly 3 hours. We found that the accuracy rate for shot detection is 91%, while the accuracy for qualitative motion estimation was 88%. Figure 1 presents some results.



(a) Left pan (b) Right pan (c) Zoom in

(d) Zoom out (e) New shot with cross fade

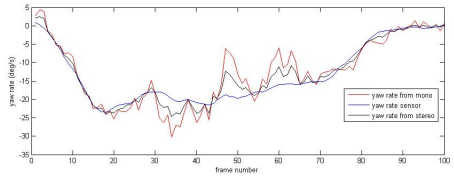
Figure 1. Various camera motions detected by our system. Optical flow vectors are represented with lines

The second algorithm (motion from calibrated camera) and the third algorithm (motion from calibrated stereo cameras) where tested against data provided by speed and yaw rate sensors (see figure 2). By integrating these we obtained the trajectory of the vehicle, shown in figure 3. This was compared visually with aerial photography images.

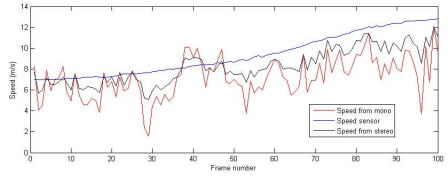
The computation for the first two, mono algorithms is around 100ms per frame, on a 2.6 GHz Intel Core 2 Duo processor. For the stereo case, the computation lasts only around 7ms per frame, which allows the system to run in real time at more than 20 fps.

7. Conclusions and Future Work

In this paper we have presented three algorithms for camera motion estimation, based on different assumption. The first algorithm works with uncalibrated mono camera, and thus can provide only qualitative results. The second algorithm uses a calibrated mono camera, and is able to recover the yaw and pitch rate exactly. Assuming that the distance between some pairs of points is known, the algorithm can also compute the speed of the vehicle on which the camera is mounted. The third algorithms works with a calibrated stereo camera pair. This algorithm provided robust data, for yaw, pitch, and roll rates and vehicle speed, in complex scenarios. We tested our algorithms against ground truth data, and found that they are correct, and produce small errors. The third algorithm is extremely fast, and can run in real-time. In the future we will concentrate on improving the third algorithm, especially by choosing better points for optical flow computation and providing supplementary checks on the 3D optical flow vectors (such as the amount of translation along z).



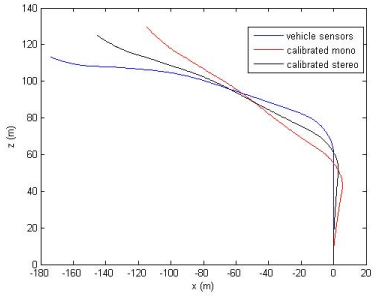
(a) Yaw rate



(b) Speed

Figure 2. Comparison of speed and yaw rate results from the calibrated mono and stereo algorithms and those supplied by the sensors

(a) Aerial photography



(b) Trajectories supplied by sensors and our algorithms

(c) Image at the beginning of the sequence

(d) Image from the middle of the sequence

(e) Image near the end of the sequence

Figure 3. Trajectory computed from yaw rate and speed for calibrated mono and stereo algorithms

References

[1] J.-Y. Bouguet. (2000) Pyramidal implementation of the Lucas Kanade feature tracker. [Online]. Available: http://mrl.nyu.edu/ bregler/classes/vision_spring06/bouget00.pdf

[2] L.-Y. Duan, J. Wang, Y.-T. Zheng, C. Xu, Q. Tian, J. S. Jin, and H. Lu, "Shot-level camera motion estimation based on a parametric model," in *TRECVID*, 2005.

[3] A. Mallet, S. Lacroix, and L. Gallo, "Position estimation in outdoor environments using pixel trackingand stereovision," in *Proceedings of IEEE International Conference of Robotics and Automation (ICRA)*, vol. 4, 2000, pp. 3519–3524.

[4] H. Hirschmuller, P. R. Innocent, and J. M. Garibaldi, "Fast, unconstrained camera motion estimation from stereo without tracking and robust statistics," in *Proceedings of the 7th International Control, Automation, Robotics and Vision Conference (ICARCV)*, vol. 2, December 2002, pp. 1099–1104.

[5] N. Molton and M. Brady, "Practical structure and motion from stereo when motion is unconstrained," *International Journal of Computer Vision*, vol. 39, pp. 5–23, August 2000.

[6] P. Saeedi, P. Lawrence, and D. Lowe, "3d motion tracking of a mobile robot in a natural environment," in *Proceedings of the IEEE International Conference on Robotics and Automation*, vol. 2, April 2000, pp. 1682–1687.

[7] T. Marita, F. Oniga, S. Nedevschi, T. Graf, and R. Schmidt, "Camera calibration method for far range stereovision sensors used in vehicles," in *Proceedings of IEEE Intelligent Vehicles Symposium, (IV2006), Tokyo, Japan*, June 2006, pp. 356–363.

[8] J.-Y. Bouguet. Camera calibration toolbox for matlab. [Online]. Available: www.vision.caltech.edu/bouguetj

[9] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Comm. of the ACM 24*, pp. 381–395, June 1981.

[10] J. I. Woodfill, G. Gordon, and R. Buck, "Tyzx deepsea high speed stereo vision system," in *Proceedings of the IEEE Computer Society Workshop on Real Time 3-D Sensors and Their Use, Conference on Computer Vision and Pattern Recognition*, jun 2004. [Online]. Available: http://www.tyzx.com/pubs/CVPR3DWorkshop2004.pdf

[11] S. Nedevschi, R. Danescu, T. Marita, F. Oniga, C. Pocol, S. Sobol, C. Tomiuc, C. Vancea, M. M. Meinecke, T. Graf, T. B. To, and M. A. Obojski, "A sensor for urban driving assistance systems based on dense stereovision," in *Proceedings of Intelligent Vehicles 2007, Istanbul*, June 2007.