A Robust Approach for Ego-Motion Estimation Using a Mobile Stereo Platform

Hernán Badino

DaimlerChrysler AG, Research and Technology
D-70546 Stuttgart, Germany
Hernan.Badino@DaimlerChrysler.com

Abstract. We propose a robust approach to the problem of ego-motion estimation using a mobile stereo platform. Stereo is computed for every frame obtaining 3D points of the environment. In addition optical flow establishes correspondences between points in successive frames. A smoothness motion constraint is applied in order to detect flow vectors which are inconsistent with the current ego-motion. The optimal rotation and translation between the resulting clouds of static points is then computed using a closed form solution based on unit quaternions, providing the motion between the current and previous frame. Stabilization and a better estimation are achieved computing the observed motion between the current frame and many frames in the past in a multi-frame fashion. The integration of successive movements allows the reconstruction of the travelled path. Experimental results with sequences covering more than 1.5 kilometers of travelled distance are presented and compared with GPS and odometry.

1 Introduction

The extraction of the observed motion of a camera has been an active area of research over the last decades. Ego-motion computation is motivated by applications like autonomous navigation, self-localization, obstacle-detection and scene reconstruction. Ego-motion is also needed by other applications which require the relative orientation of the cameras with respect to a reference frame. Our interest lies in the computation of the six degrees of freedom of the movement of a vehicle in typical traffic situations. For that purpose, a binocular platform has been mounted in the vehicle, which provides the main input to the ego-motion algorithm. Our method is a passive one, in the sense that no additional information is required, i.e. ego-motion is computed only analyzing the images provided by the cameras and the required calibration parameters.

Many approaches have been proposed with monocular and multi-ocular platforms. When using more than one camera ([9] [4] [3] [6] [7] [8] [10] [14]) the scene structure can be directly recovered through triangulation providing 3D points of the environment. Monocular approaches, instead, do not compute the scene structure ([12] [1] [11]) or they do it at the cost of integrating measurements of the image points (and possible also of other sensors) over a long time, until a reliable structure is obtained ([7] [13] [2]). Therefore multi-ocular approaches perform better in most of the cases.

Computing ego-motion from an image sequence means obtaining the camera movement with respect to a static scene, i.e. the motion is relative to something which can be

considered static. The scenarios we are interested in are typical traffic situations. Such an environment presents many participants with self-motion which can make our estimation fail if they are considered static. Also the incorrect computation of a 3D position or the incorrect tracking of features can introduce errors in the computation. Therefore an effective rejection rule must be applied in order to identify which image points are not showing a coherent movement. We propose a very effective constraint which we call smoothness motion constraint (SMC).

Mallet et al [6] present an approach very similar to the one presented here, where they focus on the reduction of errors produced by the stereo and tracking algorithms. The reduction of errors is of course important to improve accuracy, but a robust method should still be able to work well with noisy measurements, since errors are unavoidable. The computation of ego-motion is also normally computed considering only the current and previous state, which provides the current relative motion. The complete motion is then obtained concatenating the individual estimations. This may lead to poor results because of destabilization. A more stable estimation can be achieved if considering not only the last two frames, but also many frames back in the time (multi-frame estimation).

In the following section we propose a robust method for the computation of ego-motion and present the first experimental results with two sequences, summing both sequences together a travelled distance of more than 1.5 kilometers.

1.1 Organization of the Paper

In section two we present a block diagram of the method and describe shortly the stereo and tracking algorithm. Section three summarizes the least-square approach used to compute the 6 degrees of freedom of motion. Section four introduces the smoothness motion constraint. In section five we describe the multi-frame motion estimation procedure. Experimental results are shown in section six. Conclusions comprise the last section.

2 A General Description of the Approach

Figure 1 shows a block diagram of the method. The inputs are given by the left and right images from a stereo imaging system. We assume that the calibration parameters are known and that the provided images are rectified. Optical flow is computed using the current and previous left image. Disparities between the left and right image are only computed for those image positions where the flow algorithm was successful. Triangulation is performed and a list with the tracked points for the current frame is generated. The list is added to a table where the last m lists of tracked 3D points are stored. This will allow the integration of a multi-frame motion estimation, i.e. motion is not only obtained based on the last observed movement, but also between m frames in the past and the current frame.

The six motion parameters (three components for translation and three for rotation) are then computed as the optimal translation and rotation found between the current and previous list of 3D points using a least squares closed-form solution based on rotation

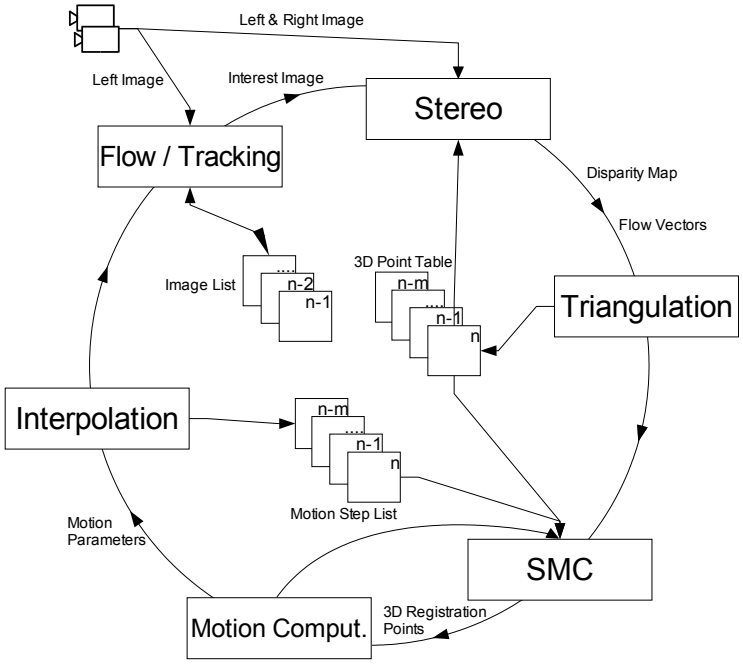

Fig. 1. Block Diagram of the Approach

quaternions as shown in the next section. In order to avoid the introduction of erroneous data in the least square computation, a smoothness motion constraint is applied, rejecting all pairs of points which represent an incoherent movement with respect to the current ego-motion (section 4). These two steps are repeated but using the list of tracked points between the current frame and m frames in the past. The two motion hypotheses are then interpolated obtaining our final ego-motion estimation, and updating the list of motion steps (section 5). The whole process is then repeated for every new input data stream.

The flow and stereo algorithms to be used are not constraint to a specific implementation. In fact, our approach was tested with different algorithms obtaining almost identical results. Nevertheless, we describe shortly the stereo and optical flow algorithms used in the experimental results of section 6. The stereo algorithm works based on a coarse-to-fine scheme in which a gaussian pyramid for left and right images is constructed with a sampling factor of two. The search for the best disparity is only performed at the top level of the pyramid and then a translation of the disparity map is made to the next level, where a correction is done within an interval ± 1 of the calculated disparity. We use the sum of squared differences as the default correlation function. Different filters and constraints are applied between pyramid translations. The zero-mean

normalized cross-correlation (ZNCC) is used in order to check the confidence of the match. A match is considered reliable if the ZNCC coefficient is larger than a predefined threshold. Dynamic programming can also be applied between pyramid translations in order to eliminate matches which invalidate the ordering constraint. Finally a sub-pixel disparity map is computed as the last step in the pyramid. This is achieved by fitting a second degree curve to the best match and its neighbors and finding the sub-pixel disparity where the slope is zero in the quadratic function.

The tracking algorithm we use for the computation of optical flow is the Kanade / Lucas / Tomasi (KLT) tracker. An extended description of the algorithm can be found in [13] and therefore we skip the description of this method here. Our experience with different tracker algorithms has shown that the KLT tracker can track feature points with a small error over tens of frames.

3 Obtaining the Relative Orientation

Let $P = \vec{p}_i$ be the set of static points of the previous time and $X = \vec{x}_i$ the sets of static points observed at the current time, where $\vec{p}_i \leftrightarrow \vec{x}_i$, i.e. $\vec{x}_i$ is the transformed version at time t_n of the point $\vec{p}_i$ at time t_{n-1}. If the measurement of point positions were free of noise, only three non-collinear points of each set would be enough to obtain the exact translation and rotation between both sets, which at the same time corresponds to the inverse movement of the camera. Therefore the optimal rotation and translation between two noisy sets must be found minimizing some error function. We use the closed form solution based on quaternions presented by Horn [5]. The main steps of the closed form are presented here. For more details see the above reference.

The error function to be minimized can be expressed as the sum of the weighted residual errors between the rotated and translated data set P with data set X, i.e.:

$$e = \sum_{i=1}^{n} w_i \|e_i\|^2 = \sum_{i=1}^{n} w_i \|\vec{p}_i - R(\vec{q})\vec{x}_i - \vec{t}\|^2 \tag{1}$$

where n is the amount of points in the sets, R is a rotation matrix obtained as a function of the unit rotation quaternion $\vec{q}$, $\vec{t}$ is a translation vector and w_i are individual weights representing the expected error in the measurement of the points.

We compute first the cross-covariance matrix of both sets:

$$\begin{aligned} \Sigma_{XP} &= \frac{\sum_{i=1}^{n} [w_i (\vec{x}_i - \vec{u}_X)(\vec{p}_i - \vec{u}_P)^t]}{\sum_{i=1}^{n} w_i} \\ &= \frac{\sum_{i=1}^{n} [w_i \vec{x}_i \vec{p}_i^t]}{\sum_{i=1}^{n} w_i} - \vec{u}_X \vec{u}_P^t \end{aligned}$$

where

$$\vec{u}_X = \frac{\sum_{i=1}^{n} w_i \vec{x}_i}{\sum_{i=1}^{n} w_i} \qquad \vec{u}_P = \frac{\sum_{i=1}^{n} w_i \vec{p}_i}{\sum_{i=1}^{n} w_i} \tag{2}$$

correspond to the weighted centroids of each set. Through the definition of the anti-symmetric matrix:

$$A = (\Sigma_{XP} - \Sigma_{XP}^t)$$

and a vector consisting of the elements of A:

$$\Delta = [\, A_{2,3}\, A_{3,1}\, A_{1,2} \,]$$

the optimal rotation quaternion $\vec{q}$ is found as the unit eigenvector corresponding to the maximum eigenvalue of the matrix:

$$Q(\Sigma_{XP}) = \begin{bmatrix} tr(\Sigma_{XP}) & \Delta^t \\ \Delta & \Sigma_{XP} + \Sigma_{XP}^t - tr(\Sigma_{XP}) I_3 \end{bmatrix}$$

where I_3 is the 3×3 identity matrix and $tr(M)$ expresses the trace of a matrix M. The rotation matrix $R(\vec{q})$ of equation 1 is obtained from the lower right hand 3×3 sub-matrix of:

$$\overline{Q}^T Q = \begin{bmatrix} \vec{q} \cdot \vec{q} & 0 & 0 & 0 \\ 0 & (q_0^2 + q_x^2 - q_y^2 - q_z^2) & 2(q_x q_y - q_0 q_z) & 2(q_x q_z + q_0 q_y) \\ 0 & 2(q_y q_x + q_0 q_z) & (q_0^2 - q_x^2 + q_y^2 - q_z^2) & 2(q_y q_z - q_0 q_x) \\ 0 & 2(q_z q_x + q_0 q_y) & 2(q_z q_y - q_0 q_x) & (q_0^2 - q_x^2 - q_y^2 + q_z^2) \end{bmatrix}$$

where $\overline{Q}^T$ and Q are the 4×4 orthogonal matrices corresponding to the quaternion $\vec{q}$ (see [5]). The translation can now be obtained as the difference of the rotated and translated centroid of the current set with the centroid of the previous set:

$$\vec{t} = \vec{u}_P - R(\vec{q})\vec{u}_X$$

3.1 Motion Representation with Matrices

The computed motion of the camera at time t_n (i.e. the motion observed from frame $n-1$ to frame n, which we call single step estimation) is represented by the matrix M'_n where:

$$M'_n = \begin{bmatrix} R(\vec{q}_n) & \vec{t}_n \\ 0 & 1 \end{bmatrix}$$

The complete motion of the camera since initialization can be obtained as the products of individual motion matrices:

$$M_n = \prod_{i=1}^{n} M'_i \tag{3}$$

Being the initial position of the camera $\vec{p}_0 = \begin{bmatrix} x_0 & y_0 & z_0 & 1 \end{bmatrix}$ the current ego-position $\vec{p}_n = \begin{bmatrix} x_n & y_n & z_n & 1 \end{bmatrix}$ can be obtained by multiplying the complete motion matrix with p_0, i.e. $\vec{p}_n = M_n \vec{p}_0$. A sub-chain of movement from time t_j to time t_k can also be obtained as:

$$M_{j,k} = M_j^{-1} M_k = \prod_{i=j+1}^{k} M'_i \tag{4}$$

Figure 2 shows an example of motion integration with matrices. As it will be seen in section 5, equation 4 will help in the integration of the motion produced between two non-consecutive frames (multi-frame estimation) in order to stabilize the computed ego-motion parameters.

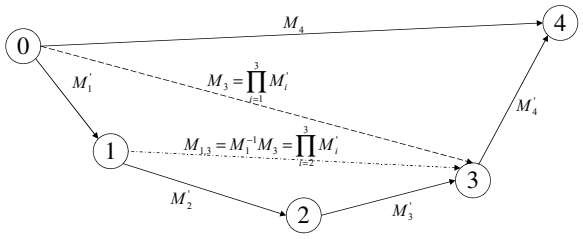

Fig. 2. The integration of single-step estimations can be obtained by just multiplying the individual motion matrices. Every circle denotes the state (position and orientation) of the camera between time t_0 and time t_4. Lines indicate motion in 3D-space.

4 Smoothness Motion Constraint

Optical flow and/or stereo can deliver false information about 3D position or image point correspondence between image frames, or some of the points can correspond to a self-moving object. Typical traffic situations present many actors with self-motion such as other vehicles and pedestrians. The situation where more than 50% of the image area is composed of moving objects is not very unusual. A robust method should still be able to give accurate results in front of such situations. If the frame rate is high enough to obtain a smooth motion between consecutive frames, then the current motion to be estimated should be similar to the immediate previous motion. Therefore, before including the pair of points $\vec{p}_i$ and $\vec{x}_i$ into their corresponding data sets P and X we

evaluate if the vector $\vec{v}_i = \overrightarrow{x_i p_i}$ indicates a coherent movement. Let us define $\vec{d} = \begin{bmatrix} \dot{x}_{max} & \dot{y}_{max} & \dot{z}_{max} & 1 \end{bmatrix}$ as the maximal accepted error of the position of 3D point with respect to a predicted position. Based on our previous estimation of motion at time t_{n-1} we evaluate the movement coherence of the vector $\vec{v}_i$ as:

$$\vec{c}_i = M'_{n-1}\vec{x}_i - \vec{p}_i \tag{5}$$

i.e., the error of the point position with respect to our prediction. If the absolute value of any component of c_i is larger than

$$\vec{d'} = \frac{\Delta t_n}{\Delta t_{n-1}}\vec{d} \tag{6}$$

where a Δt_k corresponds to the time between time t_{k-1} and t_k, then the pair of points are discarded and not included in the data sets for the posterior computation of relative orientation. Otherwise, we weight the pair of points as the ratio of change with respect to the last motion:

$$w_i = 1 - \frac{\|\vec{c}_i\|^2}{\|\vec{d'}\|^2} \tag{7}$$

which later is used as indicated in section 3. Equations (5) to (7) define the smoothness motion constraint. In figure 3 an example of the effectiveness of the SMC is shown. The left image shows all the flow vectors computed with the KLT tracker. The right image shows only those vectors which were selected by the SMC. Comparing both images it can be seen how the SMC has eliminated most of the incorrect vectors and has left only those which provide the most reliable information.

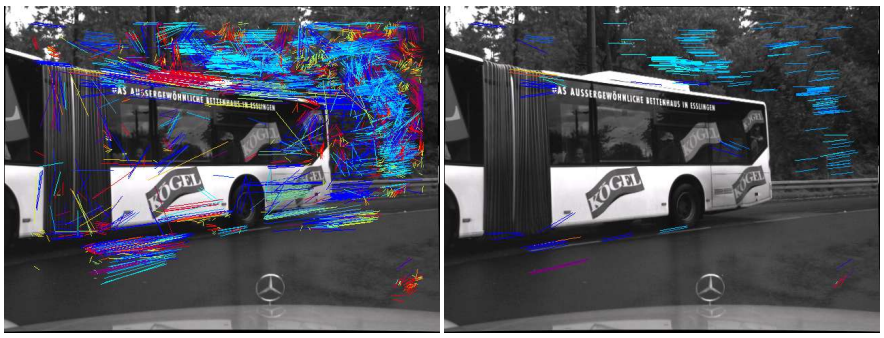

Fig. 3. Example of SMC. The images show the flow vectors before (left image) and after (right image) the SMC.

5 Multi-Frame Estimation

Single step estimation, i.e. the estimation of the motion parameters from the current and previous frame is the standard case in most approaches. If we were able to compute the motion also between non-consecutive frames, the estimation would be improved thanks to the integration of more measurements. Also robustness is increased, since when one of the estimations fail, the motion will be still provided by some of the others. Another problem is the propagation of errors. An error produced in one step will propagate superlinearly in the future ego-position estimations [10]. For example, if the yaw rate is incorrectly estimated in one step, it is implausible that this will be corrected later. A stabilization process is then also required.

We propose a simple approach in order to provide stabilization to the estimation process. If we are able to track points in m frames, then we can also compute the motion produced between the current and the m previous frames. The estimation of motion between frame m and current frame n ($m < n - 1$) follows exactly the same procedure as explained above. Only when applying the SMC, a small change takes place, since the prediction of the position for m frames is not the same as for a single step. In other words, the matrix M'_{n-1} of equation (5) is not valid any more. If the single step estimation matrix $\tilde{M}_n$ for the current frame was already computed, then equation (5) becomes:

$$\vec{c}_i = M_{n-m}^{-1} M_{n-1} \tilde{M}_n \vec{x}_i - \vec{p}_i \tag{8}$$

i.e. the estimated motion between times t_{n-m} and t_{n-1}, updated with the current simple step estimation of time t_n. This allows the SMC to be even more precise, since the uncertainty in the movement is now centered around an updated prediction, while for the single step estimation, the uncertainty is centered around a position defined by the last motion.

Once the camera motion matrix $\tilde{M}_{m,n}$ between times t_{n-m} and t_n is obtained, it must be integrated with the single step estimation. This is performed by interpolation. The interpolation of matrices makes sense if they are estimations of the same motion. This is not the case since the single step motion matrix is referred as the motion between the last two frames and the multi-frame motion matrix as the motion between m frames in the past to the current one. Therefore, the matrices to be interpolated are $\tilde{M}_n$ and $M_{m,n-1}^{-1} \tilde{M}_{m,n}$ (see figure 4). The corresponding rotation matrices are converted to quaternions in order to apply a spherical linear interpolation. The interpolated quaternion is converted to the final rotation matrix R_n. Translation vectors are linearly interpolated, obtaining the new translation vector $\vec{t}_n$. The factors of interpolation are given by the weighted sum of quadratic deviations obtained when computing the relative motion of equation 1.

6 Experimental Results

The method was tested with two sequences taken from a vehicle while driving in typical traffic situations. Aerial views[1] of the tested route of each sequence can be seen in figure

[1] Courtesy of the Municipality of Esslingen am Neckar. Source: www.esslingen.de.

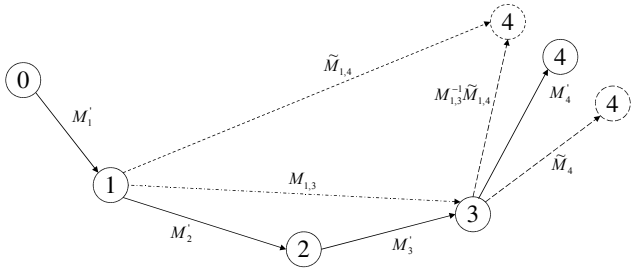

Fig. 4. Multi-frame approach for ego-motion estimation. The motion between times t_3 and t_4 is obtained as the interpolation of two motion matrices.

5 where the arrows indicate the start and the end of the sequence. The frame rate was selected to 10 frames per second. The baseline of the stereo camera is 0.35 meters and the images have a standard VGA resolution (640×480 pixels). The velocity of the vehicle varies between 18 and $55\,km/h$. The first sequence, which we call *Curves*, covers a travelled distance of around 900 meters, while the second sequence, called *Ring* covers a distance of 650 meters.

Fig. 5. Aerial images of the tested routes for the sequence *Curves* (left image) and sequence *Ring* (right image). The arrows indicates the start and end of the tested routes.

The computed ego-position for the *Curves* sequence is shown in figure 6 (a). Comparing the curve with GPS information, one observes that the error in the ego-position is relatively small. In fact, the estimation is better than using odometry, which overestimates the travelled distance. This can be seen more clearly when comparing the velocity estimations of figure 6 (b), which shows the computed velocity curves in time. Observe how the curve of our approach fits very good to the one obtained with GPS while odometry estimates the velocity too high. The estimation of the travelled distance for GPS

was 696.30 meters, for our approach 664.55 meters, and using odometry 715.99 meters. The GPS distance was expected to be larger than the distance computed with our approach, since the GPS position oscillates around the true position. In figures 6(c) is shown the change in the vehicle orientation since initialization, where the curves corresponding to odometry and ego-motion are smooth in comparison to the GPS estimation. Finally, the yaw-rate estimation for odometry and ego-motion is shown in 6(d) where almost a complete overlap of both curves is observed.

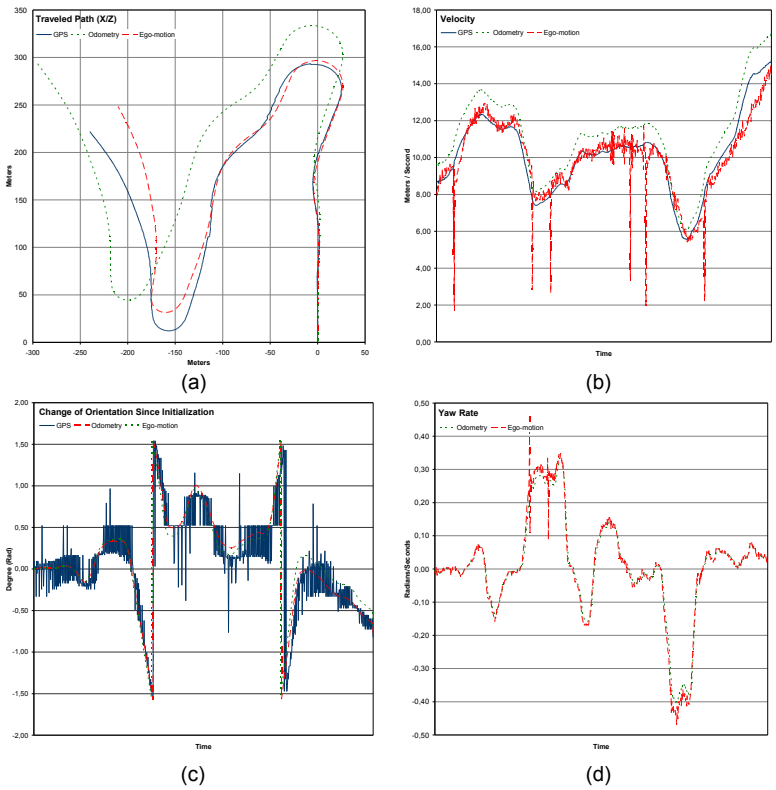

Fig. 6. Results for the sequence *Curves*

The ego-position estimations for the sequence *Ring* can be seen in figure 7 (a). Our purpose here was to observe the behavior of each method when driving 360°. The ego-motion estimation could close the ring, while odometry failed to do so. The distance between start position and end position for odometry at the point where the ring should have been closed is about 37.63 meters, for GPS is 1.73 meters and using our method 16.97 meters. The yaw rate estimation for odometry and ego-motion is shown in figure 7 (b), where once again both curves overlap most of the time.

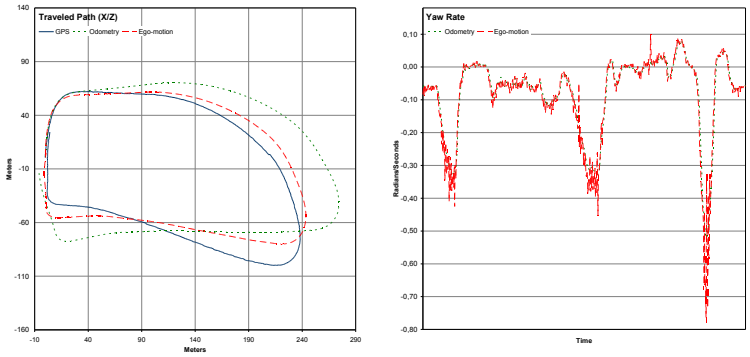

Fig. 7. Results for the sequence *Ring*

7 Conclusion

We have presented an approach for the computation of ego-motion which is very robust against moving objects and stereo/flow failures. The 6 degrees of freedom of motion can be accurately extracted with our method by computing the optimal rotation and translation between the clouds of 3D points of multiple frames. Robustness is given by an effective constraint which eliminates all flow vectors which are inconsistent with the current ego-motion. Motion can also be better estimated if the motion is computed not only based on the current and last frame, but also between the current frame and many frames in the past, providing motion stability.

The method has demonstrated to be better than odometry and yields more accurate results than GPS for short travelled distances since the position obtained with GPS are rough estimations around the true position.

References

1. Anna R. Bruss and Berthold K.P. Horn. Passive navigation. In *Computer Vision, Graphics, and Image Processing*, volume 21(1), pages 3–20, January 1983.
2. S. Carlsson. Recursive estimation of ego-motion and scene structure from a moving platform. In *Proc. 7:th Scand. Conf. on Image Analysis*, pages 958–965, Aalborg, 1991.
3. D. Demirdjian and T. Darrel. Motion estimation from disparity images. In *Technical Report AI Memo 2001-009, MIT Artificial Intelligence Laboratory*, May 2001.
4. D. Demirdjian and R. Horaud. Motion-egomotion discrimination and motion segmentation from image pair streams. In *Computer Vision and Image Understanding*, volume 78(1), pages 53–68, April 2000.
5. Berthold K. P. Horn. Closed-form solution of absolute orientation using unit quaternions. In *Journal of the Optical Society of America A*, volume 4(4), pages 629–642, April 1987.

6. Anthony Mallet, Simon Lacroix, and Laurent Gallo. Position estimation in outdoor environments using pixel tracking and stereovision. In *Proceedings of the 2000 IEEE International Conference on Robotics and Automation, ICRA 2000*, pages 3519–3524, San Francisco, April 2000.
7. R. Mandelbaum, G. Slagian, and H. Sawhney. Correlation-based estimation of ego-motion and structure from motion and stereo. In *Proceedings of the Internation Conference on Computer Vision (ICCV'98)*, pages 544–550, Kerkyra, Greece, September 1999.
8. L. Matthies and S. A. Shafer. Error modeling in stereo navigation. In *IEEE Journal of Robotics and Automation*, volume RA-3(3), pages 239–248, June 1987.
9. Louis-Philippe Morency and Trevor Darrell. Stereo tracking using icp and normal flow constraint. In *Proceedings of International Conference on Pattern Recognition*, Quebec, Canada, August 2002.
10. Clark F. Olson, Larry H. Matthies, Marcel Schoppers, and Mark W. Maimone. Rover navigation using stereo ego-motion. In *Robotics and Autonomous Systems*, volume 43(4), pages 215–229, June 2003.
11. G. P. Stein, O. Mano, and A. Shashua. A robust method for computing vehicle ego-motion. In *IEEE Intelligent Vehicles Symposium (IV2000), Dearborn, MI*, October 2000.
12. T. Suzuki and T. Kanade. Measurements of vehicle motion using optical flow. In *IEEE Conference on Intelligent Transportation Systems, Tokyo, Japan*, October 1999.
13. C. Tomasi and T. Kanade. Shape and motion from image streams: a factorization method – 3. detection and tracking of point features. Technical Report CMU-CS-91-132, School of CS – CMU, April 1991.
14. Wannes van der Mark, Daniel Fontijne, Leo Dorst, and Frans C.A. Groen. Vehicle ego-motion estimation with geometric algebra. In *Proceedings IEEE Intelligent Vehicle Symposium, Versialles, France, May 18-20 2002*, May 2002.