Comparison of Stereovision Odometry Approaches*

Niko Suenderhauf
Department of electrical engeineering and information technology
Chemnitz University of Technology
Germany
niko@hrz.tu-chemnitz.de

Kurt Konolidge
AI Center
SRI International
333 Ravenswood Avenue
Menlo Park, CA 94025, USA
konolige@ai.sri.com

Thomas Lemaire, Simon Lacroix
LAAS / CNRS
7, Ave du Colonel Roche
F-31077 TOULOUSE Cedex 4
France
firstname.name@laas.fr

I. Introduction

A precise localization estimate is essential for planetary rovers, in order to make sure that the environment models built by the rover is spatially consistent, or that the planned motions are properly executed, be they determined autonomously by the rover or uploaded from Earth. Odometry is prone to be mislead by slippages, and inertial sensors have a very low signal/noise ratio for rovers evolving slowly on non-flat terrains: an additional mean to estimate the rover displacements is required to ensure a precise position estimate.

Estimating the rover motions by tracking pixels in consecutive stereo frames, now often referred to as "stereovisual odometry", has proven to be a good solution. Various approaches have been proposed in the literature [8], [7], [10], [1][1], and such an approach has been successfully applied several times with the rover Opportunity. The main advantage of such an approach is that it is totally independent of the rover motions: provided the environment is textured enough (which is the case in all the planetary environments from which ground images have been captured up to now), the algorithms are able to estimate the rover displacements with a sometimes surprisingly good precision.

Stereovisual odometry calls for three distinct steps:

1) establishment of matches in consecutive frames;
2) determination of the 3D coordinates of the matched pixels for the two considered frames - along with an estimate of their covariances if possible (stereovision);
3) and finally numerical estimation of the 6 displacement parameters - along with the corresponding covariances if possible.

Given a solution for the first two steps (and the literature is plenty of approaches for that purpose (to obtain the results presented in this paper, we used the algorithm presented in [4] for feature selection and matching), the third step is the one that determines the precision of the motion estimates. For that purpose, several optimization techniques are adapted, and thorough experimental comparisons are required to determine their precision. In this paper, we discuss various approaches for this issue, and experimentally compare some of them. The paper describes on-going work, and only some preliminary experimental evaluations are provided.

Section II presents the Maximum Likelihood Estimation proposed by [10] to estimate the motion between two consecutive stereo frames. Section III presents two solutions based on bundle adjustment, and experimentally compares them. Section IV presents a way to estimate the motion using a Kalman filter based Simultaneous Localization And Mapping approach, and some conclusions are sketched at the end of the paper.

II. Maximum Likelihood Estimation for Two Frames Motion

Whenever we triangulate a pair of matched image points $\mathbf{x}$ and $\mathbf{x}'$ to a 3D world point $\mathbf{X}$, we introduce an uncertainty on the coordinates of $\mathbf{X}$. This uncertainty arises from the discrete nature of our sensor (each image consists of single, discrete pixels, see figure 1) and uncertainties in calibration (length of the baseline etc.) and camera alignment. The resulting uncertainty is commonly modeled as zero-mean Gaussian, although this is just an approximation [9], [5]. Algorithms for Visual Motion Estimation can either simply ignore this error model or exploit it to produce more reliable results. How to achieve that is shown in the rest of this section.

Let $\mathbf{X}_i$ and $\mathbf{X}'_i$ be the observed (triangulated) world coordinates of the i-th landmark *before* and *after* a robot motion. Thus we can write:

$$\mathbf{X}'_i = R\mathbf{X}_i + T + e_i \tag{1}$$

where T is the true translation of the robot motion and R is the true rotation matrix. e_i is a zero-mean Gaussian error vector with covariance matrix Σ_i that models all the different uncertainties that may arise. Σ_i has to be obtained by the measurement (triangulation) process.
The conditional probability for the observations $\mathbf{X}'_i$ given the motion parameters R and T can be written as

$$P_{(\mathbf{X}'_1,\mathbf{X}'_2,\ldots,\mathbf{X}'_n|R,T)} \propto e^{-\frac{1}{2}\sum_{i=0}^{n} r_i^T \Sigma_i^{-1} r_i} \tag{2}$$

*Most of the work presented here has been made during the stay of Niko Suenderhauf and Kurt Konolidge at LAAS/CNRS

[1]This list is far from being exhaustive

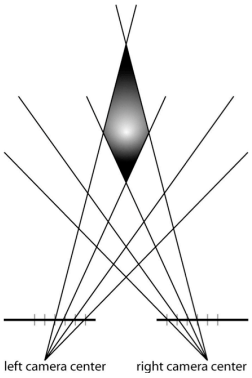

Fig. 1. The triangulation error under central projection arising from the discrete nature of imaging devices. The triangulated world point might be anywhere inside the shaded diamond area.

which is a Gaussian distribution. Here $r_i = \mathbf{X}'_i - R\mathbf{X}_i - T$. Obviously, minimizing the exponent will result in a maximized probability. Thus one has to solve

$$\min_{R,T} \sum_{i=0}^{n} r_i^T \Sigma_i^{-1} r_i \tag{3}$$

for the maximum-likelihood estimate of R and T. Because of the involved rotation this is a nonlinear minimization problem. We linearize it by taking the first order Taylor expansion with respect to the rotation. We assume Θ_0 to be the initial estimate on the rotation angles and R_0 the corresponding rotation matrix. Deriving from (1) we develop:

$$\mathbf{X}'_i \approx R_0\mathbf{X}_i + J_i(\Theta - \Theta_0) + T + \tilde{e_i} \tag{4}$$

J_i is the Jacobian of the rotation evaluated at Θ_0. The error vector $\tilde{e_i}$ now has covariance $\tilde{\Sigma}_i = \Sigma'_i + R_0\Sigma_i R_0^T$ where Σ'_i is the measurement covariance *after* the motion and Σ_i *before* the motion respectively.

After this linearization we can solve (3) using a linear method. We now use $r_i = \mathbf{X}'_i - R_0\mathbf{X}_i - J_i(\Theta - \Theta_0) - T$ and $\tilde{\Sigma}_i$ for the covariance.

The authors tested their approach in practice and observed a super-linear error growth [11]. They pointed out that the position error grows with the square root of the traveled distance while the orientation error grows as $O(d^{\frac{3}{2}})$ where d is the traveled distance. Therefore the error in orientation dominates the position error. This behavior makes the use of an absolute orientation sensor critical for long distance navigation.

The authors further describe their method for robust estimation, including an optimized feature selection (based on the Förstner interest operator [2]) and several techniques for outlier rejection. In combination with these methods the proposed estimation algorithm yields good results: After traversing approximately 20 meters and taking images every 10 cm the estimated robot pose was off by 1.2% from the ground truth provided by GPS.

III. Bundle Adjustment

Bundle Adjustment provides a solution to the following problem: Consider a set of world points $\mathbf{X}_j$ is viewed from a set of cameras with camera matrices P_i. Each camera projects $\mathbf{X}_j$ to $\mathbf{x}_{ij} = P_i\mathbf{X}_j$, so that $\mathbf{x}_{ij}$ are the image coordinates of the j-th world point in the i-th image.

What are the "optimal" projection matrices P_i and world coordinates $\mathbf{X}_j$ so that the summed squared *reprojection error* is minimal?

Thus we want to solve

$$\min_{P_i,\mathbf{X}_j} \sum_{ij} d(P_i\mathbf{X}_j,\ \mathbf{x}_{ij})^2 \tag{5}$$

where $d(\mathbf{x}, \mathbf{y})$ is the Euclidean distance between image points **x** and **y**.
Equation (5) can be extended to a weighted least-squares formulation, such as

$$\min_{P_i,\mathbf{X}_j} \sum_{ij} w_{ij}\ d(P_i\mathbf{X}_j,\ \mathbf{x}_{ij})^2 \tag{6}$$

where the weights w_{ij} may be chosen according to the variances σ_{ij}^2 of the measured image coordinates $\mathbf{x}_{ij}$.

As we see from the equations above, Bundle Adjustment is a non-linear minimization problem providing a maximum likelihood estimate for both cameras and structure parameters if the measurement noise is considered to be zero-mean Gaussian. It can be solved using iterative non-linear least squares methods such as Levenberg-Marquardt.

Bundle Adjustment (BA) is usually used as a final optimization step after good initial estimates for P_i and $\mathbf{X}_j$ have been obtained by other methods. It is, however, possible to use BA as the only algorithm for camera and structure reconstruction if attention is payed to certain issues.

A short reflection about the complexity and computational costs instantly reveals a distinct problem of the Bundle Adjustment approach: Each of the world points $\mathbf{X}_j$ has 3 degrees of freedom that have to be estimated. Each of the projection matrices P_i has 11 degrees of freedom in general and still 6 DOF when the camera calibration is known. Thus, a reconstruction over n world points and m cameras requires a minimization over $3n+6m$ parameters, which can become practically intractable very quickly with growing n and m.

A. Sparse Bundle Adjustment

However, an efficient solution to the problem has been proposed by [3] and implemented by [6].

Solving (5) with Levenberg-Marquardt involves iterative solving of normal equations of the form

$$\mathbf{J}^T\mathbf{J}\delta = \mathbf{J}^T\epsilon \tag{7}$$

where $\mathbf{J}$ is the Jacobian of the *reprojection function* $f_{(\mathbf{a},\mathbf{b})} = \tilde{\mathbf{x}}$. f takes $\mathbf{a} = (\mathbf{a}_1^T, \mathbf{a}_2^T, \dots \mathbf{a}_m^T)^T$ and $\mathbf{b} = (\mathbf{b}_1^T, \mathbf{b}_2^T, \dots \mathbf{b}_n^T)^T$ as parameters and returns $\tilde{\mathbf{x}} = (\tilde{\mathbf{x}}_{11}^T, \tilde{\mathbf{x}}_{12}^T, \dots \tilde{\mathbf{x}}_{mn}^T)^T$. Here $\mathbf{a}_i$ is the 6-Vector of the currently estimated parameters of the i-th camera, $\mathbf{b}_j$ is the 3-vector with the parameters of the j-th world point respectively. The projected image coordinates of world point j in the i-th image (according to $\mathbf{a}_i$ and $\mathbf{b}_j$) are given by $\tilde{\mathbf{x}}_{ij}$.

The Jacobian $\mathbf{J}$ of f is made up of entries $\partial\tilde{\mathbf{x}}_{ij}/\partial a_k$ and $\partial\tilde{\mathbf{x}}_{ij}/\partial b_k$. One may notice, that $\partial\tilde{\mathbf{x}}_{ij}/\partial a_k = 0$ unless $i = k$ and similar $\partial\tilde{\mathbf{x}}_{ij}/\partial b_k = 0$ unless $j = k$. This is simply because the projected coordinates of world point j in the i-th image are not dependent on any camera's parameters but the i-th and they neither depend on any other world point but the j-th. Given this, one verifies that $\mathbf{J}$ contains large blocks with 0-entries. In other words, $\mathbf{J}$ has a sparse structure. The *Sparse Bundle Adjustment* (SBA) implementation as presented by [6] takes advantage of that very structure and thus enables SBA to solve huge minimization problems over many thousands of variables within seconds on a standard PC. The C source code is freely available (under the terms of the GNU General Public License) on the author's website `http://www.ics.forth.gr/~lourakis/sba`.

B. Visual Motion Estimation using SBA

The SBA implementation solves the minimization problem as stated in (5), not considering any uncertainty of whatever kind, assuming calibrated cameras and monocular image information.

In that way, the algorithm is dependent on good initial estimates of both structure parameters (the 3D positions of the features points) and camera poses. Without these information SBA still estimates consistent values for structure and camera parameters that are accurate up to a scale factor. That means, the *proportions* of movement and 3D structure will be as in reality, but not the absolute values.

We extended the original SBA implementation in a way that it can handle stereo data input directly. We therefore had to change some internal algorithms and data structures, so that the routines take 4-vectors $\mathbf{x}_{ij}$ instead of 2-vectors. The additional two entries to these vectors are the image coordinates in the (right) stereo image. This way, we do not necessarily have to provide proper initial estimates for the world points $\mathbf{X}_j$ or the camera poses P_i. Although this impressively demonstrates the power of Bundle Adjustment, it is of course not an reasonable approach and should not be used in any seriously motivated application. Instead, we should provide BA with initial estimates for the camera pose acquired by odometry or other sensor systems like inertial navigation systems, compass, or GPS and initialize the 3D coordinates of the feature points by triangulation. Starting with reasonable initial estimates speeds up the optimization process significantly and may detain the optimization from stepping into a false local minimum.

Besides the extension to stereo input, we also implemented a simple iterative outlier rejection method. After SBA finished its minimization loop with the solutions $\mathbf{P}^*$ and $\mathbf{X}^*$, there is still an *residual error* $\epsilon_{ij} = d(P_i^*\mathbf{X}_j^*,\ \mathbf{x}_{ij})^2$ left for every $\mathbf{x}_{ij}$. A simple outlier rejection is to discard all $\mathbf{x}_{ij}$ where $|\epsilon_{ij} - \bar{\epsilon}| > k\sigma_\epsilon$ for any $k > 0$ where $\bar{\epsilon}$ is the mean of all residual errors and σ_ϵ is the corresponding standard deviation. $k = 1.5$ was chosen empirically. After the so determined outliers have been removed from the input data, SBA is restarted using $\mathbf{P}^*$ and $\mathbf{X}^*$ as initial estimates. The procedure is repeated for a fixed number of iterations or until no more outliers are found. This process helps decreasing the mean residual error and is in many cases sufficient to discard outliers arising from false or bad matches. However, as the experiments showed, it is a fairly naive approach, computationally expansive, may fail sometimes, and should therefore not be used in practical applications. Instead, we favor a more robust outlier rejection method, based on RANSAC.

After all these modifications, were able to use SBA for visual motion estimation in two different ways:

1) Sliding Window SBA
2) full SBA

1) Sliding Window SBA: The simplest and fastest estimation method is estimating structure and motion parameters between two consecutive stereo frames only. The overall motion is obtained by simple concatenation or 'chaining' of the single estimates. Intuitively one will expect this to be fairly inaccurate, as possible small errors will accumulate quickly.

To avoid the problems of simple chaining, we implemented a sliding window SBA approach. Instead of optimizing for two consecutive images only, we choose a *n-window*, e.g. a subset of n images which we perform SBA upon. The pose and structure parameters estimated in this way are used as initial estimates in the next run, where we slide the window further one frame.

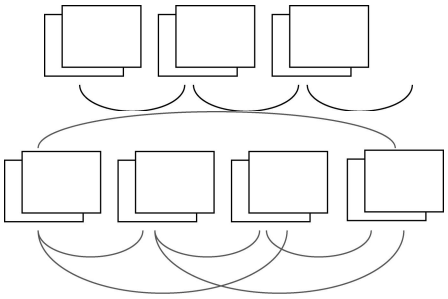

Fig. 2. Simple chaining of consecutive stereo frames and bundle adjusting a window of size 4.

With that basic sliding window approach, we would bundle adjust every consecutive n-window in the sequence of the obtained images, even if the robot has not or only very little moved while the images of the window were taken. Therefore another idea is to only include those images into the window, which pose are more than a certain threshold away from the pose of their respective predecessor in the window. The final algorithm can be summarized as follows:

1) Starting from image I_i find the closest image I_{i+k} so that the motion between I_i and I_{i+k} exceeds a certain threshold. This can be determined by pairwise SBA between I_i and I_{i+k} or, of course, using odometry data.
2) Add I_{i+k} to the window
3) Set $i = i + k$ and repeat from 1. until there are sufficient many (n) images in the window
4) bundle adjust the window using the poses obtained in step 1 as initial estimates

In this way a window size of two corresponds to the simple chaining approach.

2) Full SBA: Full SBA optimizes the whole bundle of obtained images at once. It determines camera poses and structure parameters for all recorded frames in one big optimization loop. Although this should intuitively yield the best results, it is, due to its complexity, an off line (batch) method not usable to continuously update the robot's position as he moves along.

C. Experimental Results

During all experiments we used the triangulated world coordinates as initial estimates for the world points, but did not initialize the camera poses. Instead, we assumed the cameras did not move at all. The outlier rejection was enabled and limited to 10 iterations.

1) Indoor Data: We use here an indoor sequence of 39 images. The trajectory was approximately 3 meters long and moved around a corner in a corridor. The baseline of the stereo bench is 8cm.

No ground truth was available for this dataset. We therefore have to compare the different sliding window approaches against the full SBA solution which can serve as a pseudo ground truth. 662 feature points were identified and tracked during the 39 stereo frames. The average point was visible for 5 consecutive images.

Table I summarizes the position errors in cm ordered by window size and motion threshold. The error is the distance between the estimated position at the end of the run and the ground truth (full SBA) end position.

	movement threshold in cm								
window size	0	10	15	17	20	25	30	35	40
2	3.44	2.65	0.75	1.81	1.04	2.14	1.69	0.77	1.81
3	2.65	2.78	0.72	2.29	1.13	1.74	1.14	0.95	1.78
4	2.64	2.33	0.61	2.06	0.56	1.99	1.06	0.73	2.56
5	2.22	2.62	0.48	2.36	0.84	1.02	1.00	1.12	3.60
6	2.28	2.35	0.49	2.05	0.56	1.64	1.18	1.19	3.68
7	1.77	2.26	0.55	1.91	0.87	3.23	1.42	2.24	3.06
8	1.95	1.69	0.78	2.24	0.64	2.80	1.36	1.06	3.05
9	1.04	1.49	0.92	1.37	0.29	3.19	1.42	1.06	2.92

TABLE I

POSITION ERRORS IN CM FOR INDOOR DATA COMPARED FOR WINDOW SIZES AND MOTION THRESHOLDS

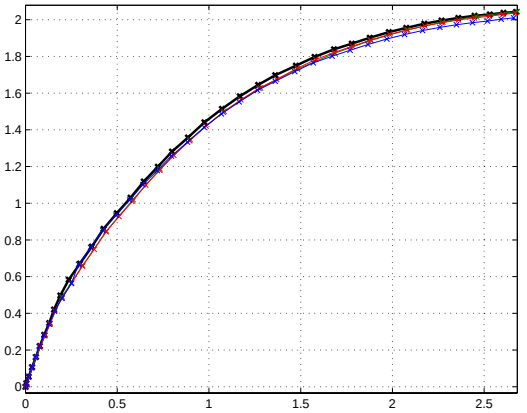

Fig. 3. Several trajectories for the indoor dataset. The thick black line is the full SBA that served as ground truth.

The robot moved approximately 3 meters around a corner of a corridor. We can observe the highest position errors to be approximately 1% of the traveled distance, most are well below the 1% level. The differences between the single results are not significant.

As a rule of thumb you see that the most simple approaches (window size 2 and zero motion threshold) yield the worst results. It is reasonable to expect bigger windows to give better results, as long as enough points are visible through *all* of the window's images. The same can be said about the motion threshold. It should be chosen as

big as possible, taking care that enough points can still be matched between the images in the window. The window size is further limited by available computation time. If the algorithm should run online (what it is supposed to do), window sizes above 5 or 6 may become intractable, depending on your hardware and the overall processor load.

2) Outdoor Data: The outdoor dataset consisted of 80 images[2], taken with a stereovision bench made of two 640x480 greyscale Point Grey Dragonfly camera with 2.8mm Rainbow lenses, and a baseline-length of approximately 8 cm. Ground truth data was extracted from using a Leica "total station" surveying instrument that measured the position of 4 reflective prisms attached to the rover. The robot moved approximately 10 Meters, turning left and right during the motion but following a relatively straight trajectory. Between 18 and 308 feature points were visible in each image, with 106 in average. Each point was visible in only 4 consecutive images in average.

a) Sliding Window SBA: The outdoor dataset was tested with window sizes between 2 and 6 and the same motion thresholds as above. As the results between the different window sizes do not differ much, we just give the minimum, maximum and mean error ordered by motion threshold in table II.

motion threshold	min.	max.	mean error
0	22.30	22.61	22.37
10	22.37	23.35	22.67
15	22.03	22.33	22.18
17	22.49	22.97	22.68
20	22.07	23.01	22.53
25	22.48	24.05	23.11
30	23.19	24.37	23.76
35	23.04	24.27	23.29
40	23.57	24.76	24.32

TABLE II

DISTANCE FROM THE GROUND-TRUTH END POSE IN CM FOR SLIDING WINDOW SBA

The average deviation from the ground truth position for all tested methods was approximately 23 cm (2.3 % of traveled distance). As we see, the error tends to increase slightly with increasing motion threshold above 20 cm because fewer feature points can be matched between two consecutive images and the quality of the matches drops with increasing movement between the images. However, the differences between the different parameter settings are not significant. They are far below 1% of the traveled distance. The camera was mounted very close to the ground and was tilted down, so only few feature points were identified (and successfully tracked) in a feasible distance from the robot to compare the different window sizes and motion thresholds.

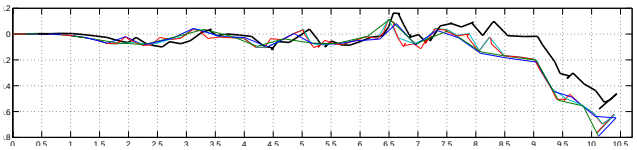

Fig. 4. Several trajectories for the outdoor dataset. The thick black line is the full SBA that served as ground truth.

b) Full SBA: The position estimated by full SBA was 20.48 cm away from the ground truth position which is slightly better than the above results, as we expected. The estimation involved 2129 3D points and 8462 image projections. 6947 parameters had to be estimated (3 for each 3D point and 7 for every camera pose). Our implementation took 6 iterations to detect and remove outliers. On a 1GHz P3 machine running Linux, the algorithm finished in 4.5 minutes. This rather long time is caused by the computationally expansive simple outlier rejection method and the stereo-extension forcing us to use SBA in the inefficient 'simple driver' mode. The required runtime may be reduced if more care is taken for runtime efficiency.

3) Conclusion: Both tested datasets did not show significant differences among the different parameter settings. However, as a rule of thumb one should not use simple chaining (resp. window size 2) as the error tends to be higher than with window sizes 3 or 4. Window sizes above 4 or 5 do in general not help to improve the result, as feature points may in average not be tracked for more than 5 images. This, of course, is highly dependent on your matching and tracking algorithm, image quality and the environment. Larger windows increase computation time drastically, so a tradeoff between computational costs and accuracy has to be considered.

The error, 2.3 % of traveled distance for the outdoor data, proves SBA to be a feasible method for visual motion estimation. In further work, the methods should be refined to make them more robust against outliers arising from false matches. This should help decreasing the error and, if implemented efficiently, can even decrease overall computation time. A possible way to achieve this is to use SBA in a RANSAC approach to estimate hypothesizes and refine the final solution on the resulting inlier set. The techniques we presented here (sliding window, motion threshold) may of course be used further in such an approach.

IV. STEREOVISUAL ODOMETRY USING EKF

Extended Kalman Filter is a well known optimization framework which we apply to Visual Motion Estimate problem. The goal is to estimate current robot pose X_r using a set of observations $z_i = [x, y, z]^t$ from landmarks

[2]We would like to acknowledge Max Bajracharya from JPL for providing us with that data

identified in the current stereo frame. Estimate X_{l_i} of landmark i must be included in the filter state in order to apply update using observation z_i. Either the landmark is already in the filter, and update can be directly applied, or the landmark is firstly observed, and in this case it must be added to the filter.

Similarity with the EKF based solution to SLAM problem (Simultaneous Localization And Mapping) is not an accident. However some differences can be underlined:

1) In the stereovisual odometry problem, we are not concerned neither with building a large map of the environment nor closing loops, which is a hard issue in SLAM.
2) Stereovisual odometry targets outdoor robot navigation and must run without any odometry measures.

As a consequence of *(1)* the number of landmarks estimated in the filter can be kept constant so as to guarantee a constant time complexity of the algorithm. The state of a landmark can be safely removed from the filter, along with its covariance and cross covariances, when this landmark is lost by the feature tracking algorithm.

Because of *(2)*, usual EKF prediction based on odometry used in SLAM does not hold. Instead a prediction model of the robot must used:

- Models such as "constant pose", or "constant speed" are very simple and can really meet our needs.
- Also a physical model of the robot can be defined, which would use some inputs such that voltage of the motors. This kind of model must be studied on a per robot basis, and they often require a tedious preliminary parameters identification. We do not favor this solution for a rover.

V. Discussion

Some points of comparison between SBA and EKF solution to stereovisual odometry are given, as well as a simple extension to deal with linearization problem which will arise with the EKF.

EKF based approach has the benefit of not requiring the tuning of the size of a sliding window, EKF optimization process takes into account all information available with the current features, whereas SBA is limited by the sliding window. Within the tested experimental setup, it was demonstrated that windows larger than 3 frames do not increase quality of the result. This number is intuitively influenced by the uncertainty of the measure, which appear in the weights of the reprojection error, and by the parameters of the camera (resolution, field of view) which appear in the projection matrix P_i. When measures are less accurate, sliding window should be increased.

On the one side EKF gives an incremental solution and is well suited to real-time implementation, on the other side linearizations must be handled with care to prevent the filter from diverging.

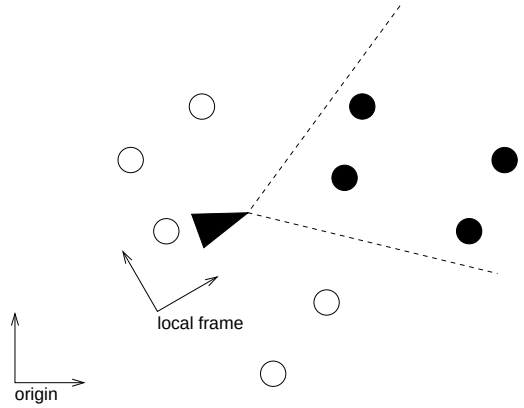

Fig. 5. Robot and landmarks estimated in a local frame, empty circles landmarks are not visible any more and have been taken off the local map.

While the robot is moving, uncertainty of the estimated pose is growing since no global pose estimator is available. Large uncertainty on estimated variables leads to EKF divergence because of linearizations computed at poorly estimated variable mean value. To cope with this problem, estimated pose of the robot can be "reset" from time to time, when uncertainty is getting too large. The "reset" operation consists in changing the reference frame of the current estimated variables (robot pose and current landmarks) to the frame defined by the current robot pose. As a result, the new pose of the robot is $[0]$ with null uncertainty, also uncertainty on the landmarks poses in the new frame is decreased thanks to the correlations with robot pose. In the meantime, variable T_0^{lf} is keeping track of the transformation from origin to current local frame, along with its covariance $P_{T_0^{lf}}$, so that robot pose and uncertainty can be computed in the reference frame. $(T_0^{lf}, P_{T_0^{lf}})$ is updated at each "reset", T_0^{lf} is composed with the old robot pose, and $P_{T_0^{lf}}$ is computed consistently. Fig 5 illustrates the local map and the different frames.

Finally, let's note that the stereovision bench setup has a big impact on the precision of the motion estimates provided, whatever the numerical solution is chosen. For instance, experimental results not presented here confirm the intuitive fact that a bench oriented perpendicularly to the rover direction of motion (looking sidewards, or even downwards) yields better motion estimates than a bench looking forward. Image resolution and focal length are also parameters of interest to determine.

References

[1] P. Corke. An inertial and visual sensing system for a small autonomous helicopter. In *11th International Conference on Advanced Robotics, Coimbra, Portugal*, July 2003.

[2] W. Förstner and E. Gülch. A fast operator for detection and precise localisation of distinct points, corners, and centres of cicular features. In *Proceedings of the Intercommission Conference on Fast Processing of Photogrammetric Data*, pages 281–3058, 1987.

[3] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004.

[4] I-K. Jung and S. Lacroix. A robust interest point matching algorithm. In *8th International Conference on Computer Vision, Vancouver (Canada)*, July 2001.

[5] I-K. Jung and S. Lacroix. High resolution terrain mapping using low altitude aerial stereo imagery. In *International Conference on Computer Vision, Nice (France))*, Oct 2003.

[6] M.I.A. Lourakis and A.A. Argyros. The design and implementation of a generic sparse bundle adjustment software package based on the levenberg-marquardt algorithm. Technical Report 340, Institute of Computer Science - FORTH, Heraklion, Crete, Greece, Aug. 2004. Available from `http://www.ics.forth.gr/~lourakis/sba`.

[7] A. Mallet, S. Lacroix, and L. Gallo. Position estimation in outdoor environments using pixel tracking and stereovision. In *IEEE International Conference on Robotics and Automation, San Francisco, Ca (USA)*, pages 3519–3524, April 2000.

[8] R. Mandelbaum, G. Salgian, and H. Swaney. Correlation-based estimation of ego-motion and structure from motion and stereo. In *7th International Conference on Computer Vision,*, pages 544–550, 1999.

[9] L. Matthies. Toward stochastic modeling of obstacle detectability in passive stereo range imagery. In *IEEE Conference on Computer Vision and Pattern Recognition, Champaign, Illinois (USA)*, pages 765–768, 1992.

[10] C. Olson, L. Matthies, M. Schoppers, and M. Maimone. Robust stereo ego-motion for long distance navigation. In *IEEE Conference on Computer Vision and Pattern Recognition, Hilton Head Island, SC (USA)*. JPL, June 2000.

[11] C. Olson, L. Matthies, M. Schoppers, and Maimone Maimone. Robust stereo ego-motion for long distance navigation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR-00)*, pages 453–458, Los Alamitos, June 13–15 2000. IEEE.