Automatically Smoothing Camera Pose Using Cross Validation for Sequential Vision-Based 3D Mapping

M. Farenzena, A. Bartoli and Y. Mezouar
LASMEA, UMR6602 CNRS - Université Blaise Pascal
Clermont, France
Michela.Farenzena@univ-bpclermont.fr

***Abstract*— Building an accurate three dimensional map is an important task for autonomous localisation and navigation. In a sequential approach to reconstruction from video streams, we show how adding prior knowledge about camera motion improves reconstruction accuracy, obtaining a more precise trajectory estimation and preventing failures over time. We add a smoothing penalty on camera trajectory and the smoothing parameter, usually fixed by trial and error, is automatically estimated using Cross-Validation. The method is substantiated by experimental results on synthetic and real data. They show that it improves accuracy and stability in the reconstruction process, preventing several failure cases.**

I. INTRODUCTION

Three dimensional (3D) reconstruction from video streams plays an important role in robotics, for example in the context of autonomous localisation and navigation [16]. A GPS receiver, the most popular localisation sensor, is accurate only if enough satellites are visible from the receiver. This is not the case in many urban areas, or indoor. The use of vision is in that really attractive because it provides a low-cost, complementary sensor to the GPS.

We consider video sequences provided by some calibrated cameras, e.g. a handheld camera, exploring new environments. These videos are challenging because they usually contain forward motions, approximations around the camera centre and shaky movements.

The sequential approach to Structure-from-Motion (SfM) [8], [10], [15] entails starting from a seed reconstruction, then adding a new view at a time, updating the structure accordingly. Some approaches for visual SLAM assume a model for camera motion [3], [2] and they can work in real time. These methods showed really promising results, but still in indoor, quite restricted areas. In our case we are interested in outdoor settings, covering wider areas.

The strategy that is usually adopted to robustly calculate a new camera pose is to use the already estimated three-dimensional (3D) points to solve a resection problem [8], [4], [7] within RANSAC [5], a robust estimator widely used in Computer Vision.

In our experience however, this does not guarantee a good initialisation for bundle adjustment and does not prevent the reconstruction process from failing. Resection indeed uses only local information; it is prone to drifting and local instabilities.

It is commonly admitted that using prior knowledge improves the quality of an estimate. In video sequences, it is reasonable to add a continuity or smoothing prior on the camera trajectory, encouraging each camera to lie close to the previous ones. This prior is not too restrictive, and reasonable for every trajectory. We minimise a cost function which is the sum of the reprojection error and the smoothing penalty, whose strength is regulated by a smoothing parameter. Usually the smoothing term is implemented as the distance between camera matrices, but this formalisation is numerically unstable. In this respect our contribution is a more stable formalisation, in terms of distance between feature points.

The smoothing parameter is commonly tuned by trial and error, and is kept constant in the whole sequence. In this work we also show that accuracy can be enhanced by choosing this parameter automatically, customising the problem for each pose. The idea is to estimate the *most predictive camera pose*, in the sense that it can "explain" the whole image as well as possible given a restricted set of data points. This is a typical machine learning problem. A Cross-Validation (CV) technique is used.

The approach is validated by experimental results on synthetic and real data. The tests are performed in different contexts and they show the versatility of the approach, preventing several failure cases.

II. RECONSTRUCTION PIPELINE

We overview the reconstruction pipeline, as summarised in Table I. We track features points in the original video stream, extracting a set of keyframes. The 3D map is initialised using the first three selected views. Then, keyframes are sequentially added, calculating the pose using the previously estimated 3D points and upgrading the 3D map with the information conveyed by the new view.

If two images are very similar the computation of the epipolar geometry is ill conditioned. Therefore in case of a video sequence a subset of images (keyframes) is usually selected. Many ways to choose these keyframes have been proposed in literature [10], [15], [18]. It should be a compromise between the distance among views, so that triangulation is well conditioned, and the number of features in common. For the calibrated case the issue is less delicate, though. Degenerate cases for epipolar geometry, such as pure rotations or planar scenes, can be handled using proper algorithms. In an exploratory setting, both camera motion

1) Track feature points on the sequence.
2) Extract keyframes and refine matches.
3) Using the first three keyframes:
   a) Estimate the relative pose using the 5-point algorithm and RANSAC;
   b) Bundle adjustment.
4) For every new keyframe extracted:
   a) Initialise camera pose with Fiore's algorithm and RANSAC;
   b) Nonlinear refinement of camera pose with smoothing penalty;
   c) Upgrade of the 3D structure;
   d) Local bundle adjustment.

TABLE I
OVERVIEW OF THE RECONSTRUCTION PIPELINE

and scene can change a lot, and we found that the number of point correspondences among multiple keyframes is the most important factor.

We used the KLT tracker [17] to detect and track features among the sequence. Similarly to [16], the first frame is chosen as the first keyframe I_1. I_2 is chosen so that there are as many frames as possible between I_1 and I_2 with at least N feature points in common. The frame I_n is selected as a keyframe if:

1) there are as many frames as possible between I_n and I_{n-1};
2) there are at least N point correspondences between I_{n-1} and I_n;
3) there are at least M point correspondences between I_{n-2} and I_n.

This criterion assures that there are matches in common at least in three consecutive views. Once a new keyframe is selected, putative matches coming from tracking are pruned by estimating the fundamental matrix with RANSAC followed by outlier rejection with the X84 rejection rule [6] on the Sampson error. Afterwards, matches are connected into tracks, keeping only those composed by at least three feature points. In our experiments we used $N = 300$ and $M = 200$.

For the first image triplet, the computation of camera motion is computed as described in [13]. It involves computing the essential matrix between the first and the third view using the 5 points algorithm [14], while the pose of the remaining camera is calculated with exterior orientation [12] using the 5 3D points triangulated from the two other views. This process is coupled with RANSAC, in order to have a robust estimation, and the final solution is further refined with bundle adjustment [11].

Afterwards, each time a new keyframe is selected, its pose is calculated referring to the 3D map already computed. This is an exterior orientation problem and we employ Fiore's algorithm [4], again within RANSAC in order to assure a robust estimation. In order to refine this initial estimation a common strategy is to minimise a geometric error, i.e. the reprojection error, and the problem is formalised as a least square minimisation of the mean of squared residuals (MSR):

$$\mathcal{E}_d^2(\mathsf{P}) = \frac{1}{n}\sum_{i=1}^{n} \parallel \Psi(\mathsf{P}, \mathbf{Q}_i) - \mathbf{q}_i \parallel_2^2 , \qquad (1)$$

where P is the projective matrix and $\mathbf{Q}_i$ is the 3D position of the image point $\mathbf{q}_i$. The function $\Psi(\mathsf{P}, \mathbf{Q}_i)$ is the reprojection of $\mathbf{Q}_i$ through P, in Cartesian coordinates. In Section III we show how to improve this refinement to increase stability.

Subsequently, the 3D map is updated. The 3D points are obtained by triangulation considering all image points of the visible tracks up to the current keyframe. A reconstructed point is considered an inlier if a) its computation is well conditioned – we set a threshold on the condition number of the matrix in the linear system that computes the 3D point – and b) if it projects sufficiently close, say by a distance of one pixel, to all associated image points. This requires to refine the initial estimation of a 3D point based on all observations, including the last. Therefore each time a new keyframe is added the tracks visible in it are checked and the list of inliers updated. When the track is no more visible it is labelled as definitely accepted or rejected.

Both structure and motion are finally adjusted using bundle adjustment. The aim is to find the parameters for the cameras and the 3D points for which the mean squared distances between the observed image points and the reprojected image points is minimised. For the first 10 views a full bundle adjustment, using all keyframes and all points, is performed. After that the computation becomes increasingly expensive, even if the sparseness inherent in the problem is exploited [19]. So we perform a local bundle adjustment, i.e. only a subset of keyframe poses are adjusted. Similarly to [9], we choose the last 5 keyframes, while the frames beyond these are locked and not moved. All 3D points visible in the last keyframes are considered, together with all measurements ever made of these points. That is, the reprojection errors are accumulated for the entire track lengths backwards in time, regardless of whether the views where the reprojections reside are locked.

III. NONLINEAR CAMERA POSE REFINEMENT

After a new camera pose is estimated, bundle adjustment is performed in order to refine both structure and motion. Though that has been proved to be the essential step to achieve a good accuracy and to prevent failures [9], the initial estimate must be sufficiently close to the optimal solution, otherwise the minimisation converges to a wrong position or diverges.

Fig. 1 shows an example of reconstruction failure, from a real sequence taken by a handheld camera. At the 47th

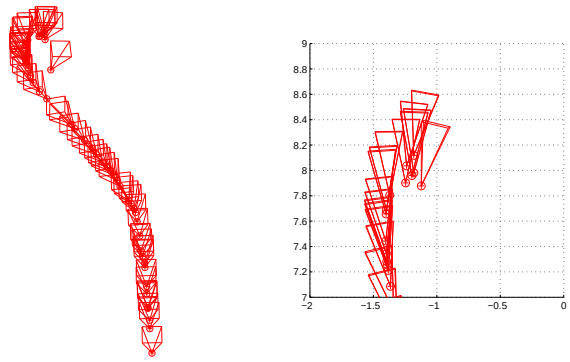

Fig. 1. Reconstruction failure in case of non refinement of camera pose. On the left a 3D view of the recovered keyframes and on the right the top view of the last keyframes.

keyframe the computation stops because there are not enough points to estimate the new camera pose (less than 6 points), meaning that all points seen in the last views have been rejected as outliers. At the moment of failure the camera was rotating. Even if the rotation is not around the camera optical centre, this is a delicate situation, where the field of view varies rapidly and the reconstruction accuracy is crucial. It is evident from Fig. 1 that the last 5 keyframes were wrongly estimated, and that bundle adjustment could not fix the problem.

As the keyframes come from a video sequence, it is reasonable to add a smoothing penalty on the camera trajectory, saying that the position of one keyframe should not differ too much from that of the previous one. This increases stability in the camera trajectory estimation.

The problem is formalised as the minimisation of a cost function, which is the sum of the reprojection error as in Eq. 1 (data term) and a smoothing term:

$$\mathcal{E}^2(\mathsf{P},\lambda) = (1-\lambda)^2\mathcal{E}_d^2(\mathsf{P}) + \lambda^2\mathcal{E}_s^2(\mathsf{P}) \qquad (2)$$

where λ is the smoothing parameter and $\mathcal{E}_s$ is the smoothing function.

The following measure is usually employed:

$$|| \mathsf{P} - \mathsf{P}_p||^2 \qquad (3)$$

with P_p the projection matrix of the previous keyframe. However, this function has a merely algebraic meaning, and its behaviour is not always as expected. In Fig. 2 (left column) two examples of pose refinement are shown using Eq. (3) as smoothing function and varying the smoothing parameter from 0 to 1. It is evident that there are numerical instability problems.

We propose here a different measure: it is the mean of squared residuals between reprojected points in the current and previous keyframes. We are essentially saying that if the two cameras are close to each other then their reprojected points should be close as well:

$$\mathcal{E}_s^2(\mathsf{P}) = \frac{1}{n}\sum_{i=1}^{n} \| \Psi(\mathsf{P},\mathbf{Q}_i) - \Psi(\mathsf{P}_p,\mathbf{Q}_i) \|_2^2 \ . \qquad (4)$$

This function is actually the finite difference approximation to the first derivatives of the predicted tracks. We could include derivatives of higher order, involving more than two views, and the resulting trajectory would be smoother, up to a straight line. As is, this is a continuity measure, reasonable for any sequential trajectory.

Our choice shows to be sensible, as depicted in Fig. 2 (right column): varying the smoothing parameter from 0 to 1 the estimated camera gradually moves toward P_p, as desired.

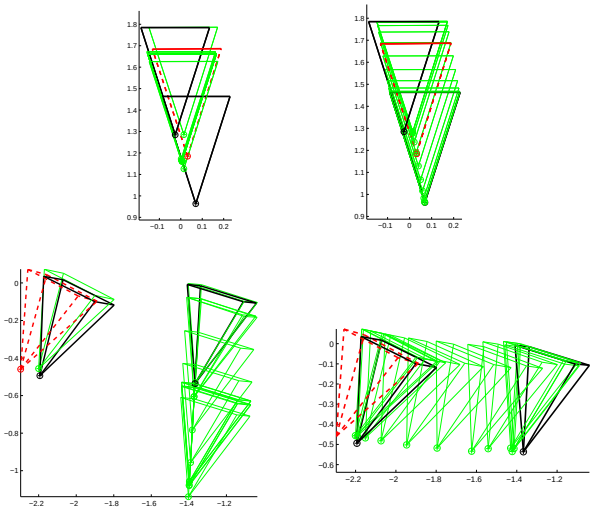

Fig. 2. Examples of camera refinement using Eq. (3) (left column) or Eq. (4) (right column) as smoothing functions. In black thick line the ground truth for the previous and current pose, in thick dashed red line the pose estimated before refinement and in thin green line after refinement, considering λ varying from 0 to 1.

The cost function (2) depends on a smoothing parameter that must be estimated. It is common to use trial and error to manually set an acceptable value, but an automatic data-driven method is obviously desirable.

A. Smoothing Parameter Estimation by Cross-Validation

We propose to estimate both the camera pose *and* the smoothing parameter. The idea is to find a camera pose as general as possible, in the sense that it can explain the whole image, given a restricted set of correspondences 3D positions – 2D points. This concept derives from the machine learning paradigm of supervised learning from examples.

The approach we follow is to split the data points in a training and a test set, and select the smoothing parameter for which the trained model minimises the test error. A well-known method, widely applied in machine learning [1], is Cross-Validation, firstly introduced in [20]. Considering that the number of samples is small, this technique recycles the test set, averaging the test error over several different partitions of the whole data set.

There are different kinds of Cross-Validations, like leave-one-out (CV_{loo}), k-fold (CV_{kf}) and generalised cross validation. CV_{loo} gives more accurate results, but it is computationally too expensive. We chose CV_{10f} as a compromise between accuracy and efficiency.

The method works as follows. The dataset S is divided into 10 partitions $\{S_j\}_{j=1..10}$. The test error, or CV_{10f} *score*, is defined as a function of the parameter λ:

$$\mathcal{E}_g^2(\lambda) = \frac{1}{10}\sum_{j=1}^{10}\sum_{i\in S_j} ||\Psi(\hat{\mathsf{P}}_{(S_j)}(\lambda), \mathbf{Q}_i) - \mathbf{q}_i||_2^2. \quad (5)$$

$\hat{\mathsf{P}}_{(S_j)}(\lambda)$ is the camera pose estimated with all but the correspondences in the j-th partition:

$$\hat{\mathsf{P}}_{(S_j)}(\lambda) = \arg\min_{\mathsf{P}} \ \mathcal{E}^2_{(S_j)}(\mathsf{P}, \lambda). \quad (6)$$

Fixing a value for λ, each time 9 of the 10 partitions are used to estimate the camera pose, and the reprojection error is calculated on the unused partition. The CV_{10f} score is the mean of these errors.

The most predictive camera pose $\hat{\mathsf{P}}$ is obtained by solving the following nested optimisation problem:

$$\hat{\mathsf{P}} = \arg\min_{\mathsf{P}} \ \mathcal{E}^2(\mathsf{P}, \arg\min_{\lambda} \ \mathcal{E}_g^2(\lambda)). \quad (7)$$

This means that the optimal $\hat{\lambda}$ – the one with the lowest CV_{10f} score – is firstly selected, and afterwards the final camera pose is obtained by minimising $\mathcal{E}(\mathsf{P}, \hat{\lambda})$.

As is, this Cross-Validation method is not robust, in the sense that it does not cope with mismatched correspondences. Therefore, we use the robust RANSAC estimation as initial solution for determine $\hat{\mathsf{P}}_{(S_j)}(\lambda)$ in Eq. (6), and the data set is restricted to only correspondences classified as inliers after RANSAC. Moreover, the calculus of $\hat{\lambda}$ is carried out by sampling, estimating Eq. (5) at steps of 0.01 from 0 to 1. Here 0.01 was experimentally derived as a sufficient order of approximation.

Fig. 3 shows the final camera pose obtained by the proposed nonlinear refinement for the cases depicted in Fig. 2. The optimal values of λ calculated considering CV_{10f} score are respectively 0.05 (left) and 0.03 (right).

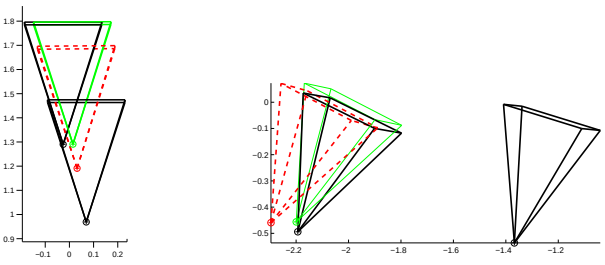

Fig. 3. Final pose estimation (green, thin line) for the examples in Fig. 2, with λ obtained automatically by CV_{10f}. In black thick line the ground truth for the previous and current pose, in red dashed line the initial pose estimated by RANSAC.

IV. EXPERIMENTAL RESULTS

We show the effectiveness of our method on synthetic critical sequences. The dataset consisted of 100 points randomly scattered in a sphere of radius 1 meter, centred at the origin.

We considered three different scenarios. In the first setting, views were generated by placing cameras along a line in z-direction, at a distance from the origin of 5.5 up to 7 meters approximately. In the second setting, in order to simulate more unstable cases, the rectilinear trajectory was perturbed along x-direction, applying a Gaussian noise of standard deviation 0.8. In the third setting the trajectory was perturbed in the three directions, with the same noise. In the three cases the number of views was fixed to 10, and a Gaussian noise with standard deviation 0.5 was added to the image points.

For each scenario we compared results in terms of distance of the estimated cameras to the ground truth, considering four cases: a) without using nonlinear refinement of camera pose, b) using nonlinear refinement, but without the smoothing term, c) with the smoothing term and λ carefully set by hand and kept constant for the whole sequence, and finally d) with λ estimated by Cross-Validation, as proposed in this paper. For each scenario 50 independent trials were carried out.

Results are shown in Table II. The distances between the camera centres of the estimated cameras and the ground truth are reported. It is interesting to note that on some cases b) and c) produce worst results wrt the non refined a). This shows that our data-driven method, that estimates λ for each case independently, is the right way to face the problem. In the third scenario without nonlinear pose refinement the computation stopped before estimating all cameras in 20% of cases. The proposed method clearly improves stability and accuracy in pose estimation.

Fig. 4. Three consecutive keyframes for the *Campus1* sequence (top), *Campus2* (middle) and *Laboratory* (bottom).

For real sequences, we first show the results from a video, *Campus1*, taken by a calibrated handheld camera (see Fig. 4). The trajectory executed was an initial rotation,

	Setting 1				Setting 2				Setting 3			
	a	b	c	d	a	b	c	d	a*	b	c	d
mean	0.1704	0.0916	0.0793	0.0566	0.0766	0.110	0.091	0.044	0.052	0.072	0.090	0.067
min	0.001	0.005	0.004	0.002	0.005	0.004	0.007	0.001	0.003	0.003	0.005	0.004
max	1.018	1.021	0.301	0.383	0.582	0.707	0.614	0.543	0.396	0.479	0.545	0.465

TABLE II

RESULTS ON SYNTHETIC SCENES. MEAN, MINIMUM AND MAXIMUM DISTANCE OF ESTIMATED OPTICAL CENTRES FROM THE GROUND TRUTH (IN METERS) FOR THE THREE SYNTHETIC SETTINGS AND THE CASES A) WITHOUT USING NONLINEAR REFINEMENT, B) USING NONLINEAR REFINEMENT, BUT WITHOUT THE SMOOTHING TERM, C) WITH THE SMOOTHING TERM AND λ CAREFULLY SET BY HAND AND D) WITH λ ESTIMATED BY CROSS-VALIDATION. (*) IN THIS SETTING THE COMPUTATION FAILS IN 20% OF CASES.

then a rectilinear part and finally another small rotation, without caring too much about shaking. From 1608 frames of resolution 784×516, without nonlinear refinement the reconstruction process stops at the 47th keyframe, as already displayed in Fig. 1. Using the proposed method, instead, 135 keyframes are extracted, with 5000 points reconstructed in a maximum reprojection error of 1.0 pixel, as expected. The 3D map produced, with the estimated camera trajectory, can be seen in Fig. 5.

The second video, *Campus2*, was taken with a camera mounted on a car-like robot (see Fig. 4). In this case camera trajectory was rectilinear with a sharp rotation of 90° at the end. From 306 frames of resolution 768×1024, 74 keyframes were extracted and successfully estimated with the proposed method (Fig. 5). The final 3D map is composed by 3438 3D points in a maximum reprojection error of 1.0 pixel, as expected. Even if in this case camera motion is really stable, without refinement the reconstruction process stops after 35 keyframes, with the last keyframe really far away from the correct position, as shown in Fig. 6.

The third video, *Laboratory*, was taken with a camera mounted on a Unmanned Autonomous Vehicle (UAV), in an indoor setting. It was made up of 929 frames of resolution 576×784 (see Fig. 4). 79 keyframes were calibrated, and the final 3D map is composed by 4421 points (Fig. 5) in a maximum reprojection error of 1.0 pixel, as expected. Without refinement the reconstruction process stopped in the last rotation (Fig. 6).

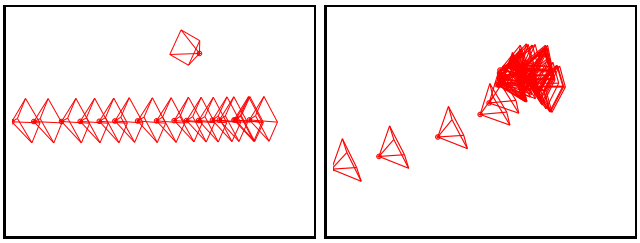

Fig. 6. Failures in camera motion recovery without camera pose refinement for *Campus2* (left) and *Laboratory* (right) sequences.

V. CONCLUSIONS AND FUTURE WORKS

In this paper we proposed to refine camera pose adding a smoothing penalty on camera trajectory and to automatically estimate the smoothing parameter, usually manually fixed, using Cross-Validation. The experimental results show that the method is effective in improving accuracy and stability in the reconstruction process. It is also versatile, in the sense that it has been successfully applied in different contexts, with camera mounted on different kind of vehicles.

Future work will include the investigation of approximations for the Cross-Validation score, in order to reduce the computational cost, which is in fact the main drawback of the approach. In that way our method could be embedded in real time SfM pipelines.

VI. ACKNOWLEDGMENTS

This research was founded by the EU-Project FP6 IST μDrones, FP6-2005-IST-6-045248.

REFERENCES

[1] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

[2] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "Monoslam: Real-time single camera slam," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 1052–1067, June 2007.

[3] E. Eade and T. Drummond, "Scalable monocular vision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2006, pp. 469–476.

[4] P. D. Fiore, "Efficient linear solution of exterior orientation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 2, pp. 140–148, 2001.

[5] M. A. Fischler and R. C. Bolles, "Random Sample Consensus: a paradigm model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, June 1981.

[6] F. Hampel, P. Rousseeuw, E. Ronchetti, and W. Stahel, *Robust Statistics: the Approach Based on Influence Functions*, ser. Wiley Series in probability and mathematical statistics. John Wiley & Sons, 1986.

[7] R. Haralick, C. Lee, K. Ottenberg, and M. Nolle, "Review and analysis of solutions of the three point perspective pose estimation problem," *International Journal of Computer Vision*, vol. 13, no. 3, pp. 331–356, 1994.

[8] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*, 2nd ed. Cambridge University Press, 2003.

[9] C. Hengels, H. Stewénius, and D. Nistér, "Bundle adjustment rules," in *Photogrammetric Computer Vision*, September 2006.

[10] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *Proc. Sixth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'07)*, Nara, Japan, November 2007.

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

[12] F. Moreno-Noguer, V. Lepetit, and P. Fua, "Accurate non-iterative O(n) solution to the PnP problem," in *Proceedings of the IEEE International Conference on Computer Vision*, 2007.

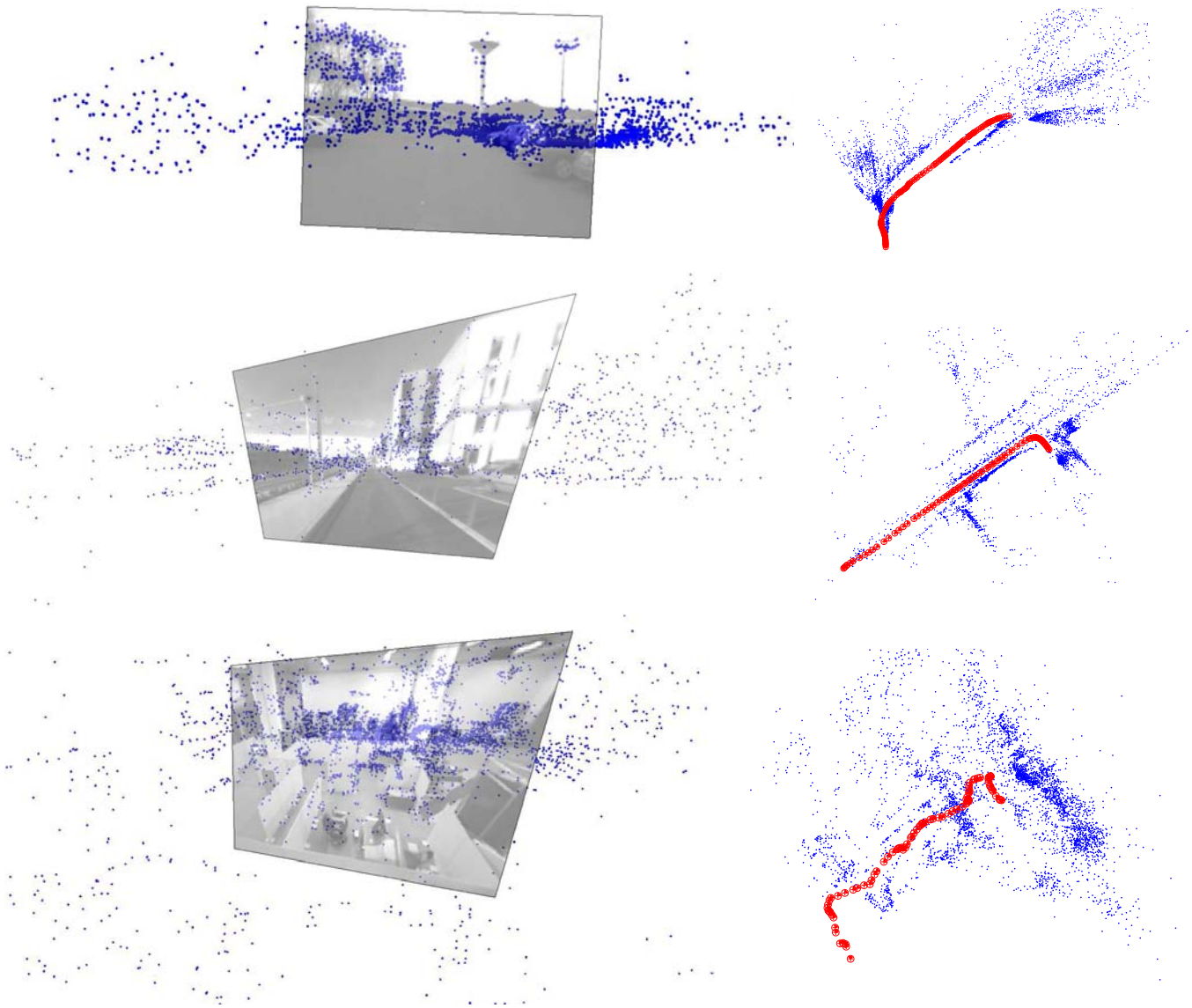

Fig. 5. 3D maps of (from top to bottom) *Campus1*, *Campus2* and *Laboratory* sequences. The left column shows a perspective view of the maps, where for visualisation understanding one keyframe of the sequence is superimposed. The right column shows a top view of the maps, with in red the estimated cameras.

[13] D. Nistér, O. Naroditsky, and J. Bergen, "Visual odometry," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, 2004, pp. 652–659.

[14] D. Nistér, "An efficient solution to the five-point relative pose problem," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 6, pp. 756–777, 2004.

[15] M. Pollefeys, L. V. Gool, M. Vergauwen, F. Verbiest, K. Cornelis, J. Tops, and R. Koch, "Visual modeling with a hand-held camera," *International Journal of Computer Vision*, vol. 59, no. 3, pp. 207–232, 2004.

[16] E. Royer, M. Lhuillier, M. Dhome, and J. Lavest, "Monocular vision for mobile robot localization and autonomous navogation," *International Journal of Computer Vision*, vol. 74, no. 3, pp. 237–260, 2007.

[17] J. Shi and C. Tomasi, "Good features to track," Department of Computer Science, Cornell University, Ithaca, NY 14853-7501, Technical Report 93-1399, November 1993.

[18] P. H. S. Torr, "Bayesian model estimation and selection for epipolar geometry and generic manifold fitting," *International Journal of Computer Vision*, vol. 50, no. 1, pp. 35–61, 2002.

[19] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, "Bundle adjustment - a modern synthesis," in *ICCV '99: Proceedings of the International Workshop on Vision Algorithms*, 2000, pp. 298–372.

[20] G. Wahba and S. Wold, "A completely automatic french curve: Fitting spline functions by Cross-Validation," *Communications in Statistics*, vol. 4, pp. 1–17, 1975.