Fast and Accurate Structure and Motion Estimation

Johan Hedborg, Per-Erik Forssén, and Michael Felsberg

Department of Electrical Engineering
Linköping University, Sweden
hedborg@isy.liu.se

Abstract. This paper describes a system for structure-and-motion estimation for real-time navigation and obstacle avoidance. We demonstrate a technique to increase the efficiency of the 5-point solution to the relative pose problem. This is achieved by a novel sampling scheme, where we add a distance constraint on the sampled points inside the RANSAC loop, before calculating the 5-point solution. Our setup uses the KLT tracker to establish point correspondences across time in live video. We also demonstrate how an early outlier rejection in the tracker improves performance in scenes with plenty of occlusions. This outlier rejection scheme is well suited to implementation on graphics hardware. We evaluate the proposed algorithms using real camera sequences with fine-tuned bundle adjusted data as ground truth. To strenghten our results we also evaluate using sequences generated by a state-of-the-art rendering software. On average we are able to reduce the number of RANSAC iterations by half and thereby double the speed.

Structure and motion (SaM) estimation from video sequences is a well explored subject [1,2,3]. The underlying mathematics is well understood, see e.g. [1], and commercial systems, such as Boujou by 2d3 [4], are used in the movie industry on a regular basis. Current research challenges involve making such systems faster, more accurate, and more robust, see e.g. [2,3]. These issues are far from solved, as is illustrated by the 2007 DARPA urban challenge [5]. In the end, none of the finalists chose to use the vision parts of their systems, instead they relied soley on LIDAR to obtain 3D structure. Clearly there is still work to be done in the field.

This paper aims to increase the speed and accuracy in structure-and-motion estimation for an autonomous system with a forward looking camera, see figure 1. Although on a smaller scale, this platform has the same basic geometry and motion patterns as the DARPA contenders, and as the vision based collision warning systems developed for automotive applications. In such systems, estimated 3D structure can be used to detect obstacles and navigable surfaces.

When dealing with forward motion there are a number of problems that must be adressed. The effective baseline is on average much smaller than for the sideways motion case, resulting in a more noise sensitive structure estimation. A tracked point feature near the camera often has a short lifespan because it quickly

G. Bebis et al. (Eds.): ISVC 2009, Part I, LNCS 5875, pp. 211–222, 2009.

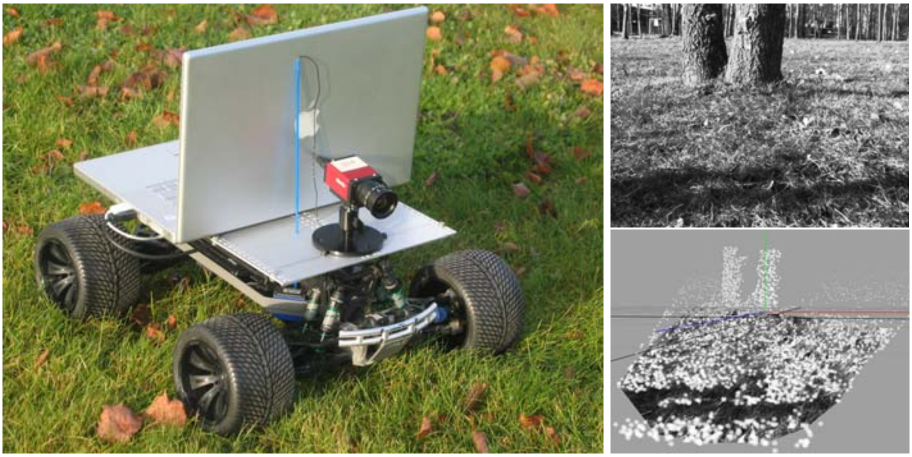


Fig. 1. Left: Robotic car platform. Upper right: Frame from a forward motion sequence. Lower right: Estimated structure using the proposed algorithm.

moves out of the the visual field. Unfortunately, such points also contain most of the structural information [6]. Forward motion also produces large scale changes in some parts of the image, and this can be a problem for some trackers.

This paper studies the calibrated SaM formulation, which has several advantages over the uncalibrated formulation. In calibrated SaM, estimated cameras and structure will be in Euclidean space instead of a projective space, and we can use more constrained problem formulations [1]. Planar-dominant scenes are not an issue when doing calibrated five point pose estimation [7], which turns out to be a very desirable property when doing autonomous navigation, as these kinds of scenes are quite common. Note also that in autonomous navigation the camera is often fixed, which makes calibration of the camera straightforward.

We should also note that monocular SaM has an inherent scale ambiguity [1]. Despite this, the estimated structure can still be effectively used for obstacle avoidance if *time-to-collision* is used as the metric [8].

The main contributions of this paper are:

1. Introduction of a distance constraint that significantly reduces the number of RANSAC iterations in the five point algorithm, while retaining the pose accuracy. Even when all correspondences in a sample are inliers, their distribution in space makes a big difference. This idea is very easy incorporate. Somewhat surprisingly it does not appear to be described elsewhere.
2. Experimental evaluation of a recently introduced outlier rejection technique for the KLT tracker [9], in the SaM setting. This technique adds an outlier rejection step already in the tracking algorithm. With respect to performance this is used to move calculations from the CPU to a GPU.

We also demonstrate how sophisticated rendering techniques can be used for controlled evaluation of the system.

1 Previous Work

The use of calibrated epipolar geometry in computer vision was pioneered by Longuet-Higgins in his seminal work [10]. The minimal number of point correspondences in the calibrated case is five, and the current state-of-the-art five-point method is the one introduced by Nistér [7]. The exact solution involves cubic constraints, which result in a polynomial with 13 roots. Nistér reduced the number of roots to 10 and provided very efficient solutions to each step of the algorithm. His paper also describes how to add a third view, by solving the perspective-three-point problem [11]. We use this complete three view method in our paper, and refer to it as the Nistér three view method.

Ever since the original RANSAC algorithm was introduced [11], many modifications to the algorithm have appeared. Some methods assume prior information of which points are likely to be inliers, and use this to bias the sampling, e.g. PROSAC. Others estimate the point inlier likelihoods as they go [12], for instance using the point residual distributions [13]. Others discard samples (i.e. groups of points), before scoring them against a model, by comparing the sample points against the model. R-RANSAC [14] and preemptive RANSAC [15], are examples of this. In our setup model estimation is relatively expensive, so it would be better if we could discard a sample even before computing the model. This is exactly what our constraint does, and in this respect, it is similar to NAPSAC [16], which selects points that lie close together when estimating hyperplanes. But, as we will show, for our problem it is on the contrary better to select points that are far apart.

Wu et al.[17] have shown how the KLT tracker can be improved by simultaneously tracking both forwards and backwards in time. Another approach is to simply run the tracker again, backwards in time, and reject trajectories that do not end up at the starting point [9]. We will use the latter approach, and demonstrate its effect in the experiment section.

Rendered 3D scenes as synthetic ground truth has a long history in the field of motion estimation, e.g., the famous Yosemite sequence [18]. This was at the time a very complex scene as it had real 3D structure. Baker et al. argued in [9] that the Yosemite sequence is outdated and they introduce a new set of ground truth data. These new datasets use modern rendering software that can accurately model effects such as shadows, indirect lighting and motion blur. We will use similar data of our own design, in the experiment section.

2 Method

2.1 Overview

The real-time SaM method that we are using consists of two steps:

1. Point correspondences are maintained over time using the Kanade-Lucas-Tomasi (KLT) tracking framework [19,20].

2. The relative pose is estimated for 3 cameras, according to the method described in [7]. In this approach, the relative pose is first found between 2 cameras. Then, triangulation and the perspective-3-point algorithm [11] is used to incorporate the third camera. All of this is done for the minimal 5-point case, inside a RANSAC [11] loop.

We will describe these two steps in detail below, as well as the modifications we have added to each step.

2.2 Tracking

We use the KLT-tracker [20] to maintain point correspondences across time. The KLT-tracker is basically a least-squares matching of rectangular patches that obtains sub-pixel accuracy through gradient search. We use a translation-only model between neighbouring frames. Tracking between neighbouring frames instead of across larger temporal windows improves the stability, especially since it helps us to deal with the scale variations that are present in forward motion. When tracking frame-by-frame, the changes in viewing angle and scale are sufficiently low for tracking to work well.

A simple way to increase the quality of the point correspondences is to add an early outlier rejection step. We do this by running the tracker backwards from the current frame and position and checking if it ends up at its initial position in the previous frame. We will call this procedure *track-retrack* from now on. Adding the track-retrack step doubles the amount of computations. However, since our tracker is running on the GPU, which has cycles to spare, this does not affect the overall performance of the rest of our system.

The KLT tracker has successfully been implemented on a GPU by several authors [21,22,23]. Reference [22] shows a speed increase of more than 20x, and can track thousands of patches in real time. Such a large amount of tracked points is not necessary to estimate the camera motion, and we can thus easily afford to run the tracker a second time. In order to fully utilize the GPU, the number of threads of an implementation must be high. While a CPU can efficiently run 2 threads on a 2 core system, the GPU's core is a simpler version of a processor with very high memory latencies, little or no cache and with many SIMD characteristics. To achieve maximum performance from such a design we need many more threads than processors, and as the current high-end hardware has 240 processors, one often needs more than 5000 threads [24].

2.3 Five Point Pose Estimation

The minimal case for relative pose estimation in the calibrated case is five corresponding points seen in two cameras. Currently, the fastest algorithm for this problem is given by Nistér in [7]. It runs in real-time, and thus we have chosen it as our starting point. In this method, the relative pose estimation is extended to three cameras by doing the following within the RANSAC loop: The essential matrix is estimated from five point correspondences between two cameras.

The relative camera position and rotation are then extracted from the essential matrix. From these, the camera projection matrices are created and used to triangulate the five 3D points. The perspective-3-point algorithm [11] is then used to calculate the third camera from three 3D points and their respective projections onto this camera.

One advantage with using the minimal case is that it imposes all available constraints on the estimation. This is especially important when handling more complicated cases like the forward motion case. If one plane is dominant, the uncalibrated case has several solutions. In the calibrated case, the number of solutions is reduced to two, where one can be discarded (as it has the cameras below ground). It is thus not necessary to switch between the homographic and the full epipolar geometry model. We use the Nistér method here, but note that in principle any five point solver would benefit from the improvements we suggest in this paper.

2.4 Distance Constraint

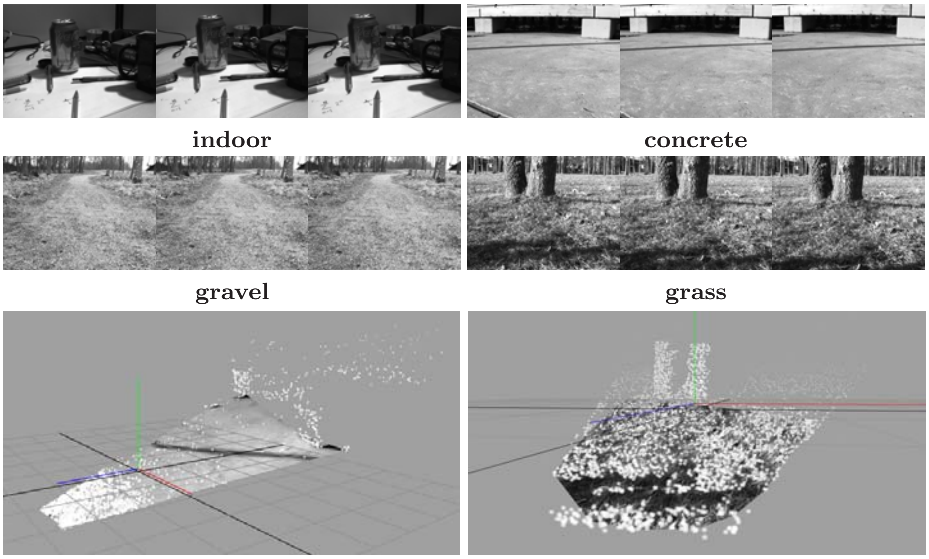
Forward motion in structure from motion is a notoriously difficult case, because of the much smaller *equivalent baseline*[1] created between two cameras than with other types of motions. The forward motion also gives rise to large scale differences in the point correspondence estimation. The point tracking becomes less accurate under these scale transformations. Most of the time we will also have a singular point (the epipole) lying in the image (in the motion direction). At this point the equivalent baseline is zero, and it increases towards the edges of the frame.

Computation of the relation between two cameras by estimating the essential matrix is quite sensitive to the actual 3D positions of the used correspondences. This is demonstrated by a recent discovery by Martinec and Pajdla [3]. In their paper, they show that bundle adjustment using only four carefully chosen correspondences between each pair of views can be almost as accurate (and much faster) as using all correspondences. These four points are chosen to be maximally distant in the 3D space with metric determined by the data covariance. However, for a direct solution of SaM this is too expensive as it requires triangulated 3D points. Instead, our proposal is to look at the projections in the image plane. The rationale is that points that are distant in the image plane are *likely* to be distant also in 3D space.

The standard approach when solving structure and motion with [7] is to place the minimal case five point solver inside a RANSAC loop. Our proposal is to add a simple distance test inside the loop, before the minimal case solver. With this test we put a minimum distance constraint on the randomly chosen image points, $\mathbf{x}$. Only sets of five points $\mathbf{x}_1,\ldots,\mathbf{x}_5$, that satisfy:

$$||\mathbf{x}_i - \mathbf{x}_j||_2 > T,\ \forall\{(i,j) : i,j \in [1..5], j > i\}\ , \tag{1}$$

[1] By equivalent baseline, we informally mean the distance between the camera centers when projected onto the image plane.



indoor

concrete

gravel

grass

reconstruction of *concrete* sequence

reconstruction of *grass* sequence

Fig. 2. The top four subplots show the used frames from the real world sequences. The indoor sequence (top left) is chosen as it has plenty of occlusions. The three outdoor scenes are captured while driving the robotic car forward on different terrains: concrete, gravel and grass. The two lower images are the three-view reconstructions of two of the sequences (a navigable planar surface is also estimated and textured for illustration).

will be used for pose estimation. For other sets the sampling is run again. We use the threshold $T = 0.1$ on distances in normalized image coordinates. This corresponds to approximately 150 pixels in our sequences (or about 10% of the image diagonal). This value gave a reasonable compromise between the number of resamplings, and the precision obtained. We have not done any extensive tests on the exact value to use.

The computational load of the point pre-selection procedure is small compared to the 5 point solver. It consists of 5 `rand()` function calls, 10 conditional instructions and some simple arithmetical operations. If necessary this can be further optimized by different gridding methods. The average time to compute one sample on our platform (in one thread on an Intel 2.83GHz Q9550 CPU) in standard C++ is 0.1 microseconds. On our datasets this is done on average 2-4 times, for $T = 0.1$. The three-view five-point method is reported to take 140 microseconds in 2004 [7]. Accounting for CPU speedups gives us about 35 microseconds, or 87-175x more than our sampling step.

3 Evaluation

Evaluation was carried out on four real world sequences captured with a PointGrey Flea2 camera (1280 × 960 at 15 Hz) with pure forward motion, see top

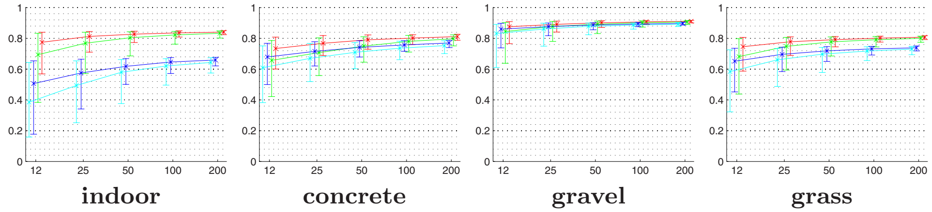


Fig. 3. Statistics of inlier/outlier ratios as a function of number of RANSAC iterations (12, 25, ...). Each graph shows results for one sequence. Each experiment is repeated 500 times, and the median, and 5% and 95% quantiles are plotted. The four curves (in left-right order within each group) show results without distance constraint and without track-retrack (CYAN), with distance constraint and without track-retrack (BLUE) without distance constraint and with track-retrack (GREEN), with distance constraint and with track-retrack (RED).

of figure 2. OpenCV's software library is used to calibrate the camera from a checkerboard pattern [25]. We have also chosen to use OpenCV's KLT implementation in the experiments to make it easier for others to reproduce our results.

Additionally we have generated two synthetic sequences, shown in figure 4. We have used a rendering software called Mental ray that has a wide variety of modelling capabilities such as complex geometry, soft shadows, specular highlights and motion blur. These are effects that impact the performance of the SaM, and we would like to further investigate this in the future. For now we have just used them with settings that give footage similar to the real camera. Besides being used in many movies, Mental ray was also used in [9] to generate image sequences and ground truth for optical flow.

We will use the real-world sequences together with the synthetic sequences to evaluate the efficiency of the distance constraint. We use two measures in the experiments:

- **Inlier Frequency.** We count the number of inliers in the best model found by RANSAC. This is the criterion that RANSAC itself tries to maximise, and thus it demonstrates how much our modifications have assisted RANSAC.
- **Model Precision.** We evaluate the scale normalised position of the third camera. On real sequences, this is done by comparing our estimate against the output of the bundle-adjustment algorithm described in [26]. On the synthetic sequences, we know the exact locations of each camera, and use that as ground truth.

3.1 Inlier Frequency Evaluation

In each sequence, we have used 3 images to compute SaM, and this has been done for 12, 25, 50, 100 and 200 RANSAC iterations. This procedure is run 500 times, and from this we calculate the median and the 5 and 95 percentiles to show where 90% of the estimates lie. This kind of evaluation is used for all graphs in the paper. To evaluate the performance of the distance constraint the same setting is run both with, and without the constraint.

In figure 3 we give graphs of the expected inlier frequency in the best model found by RANSAC, as a function of the number of RANSAC iterations. These graphs clearly show an inlier increase when the distance constraint is used, and this holds both with, and without the track-retrack step. Almost everywhere, the curves without the distance constraint need more than double the amount of iterations to reach the same inlier frequency. In most of the real sequences we could reduce the number of RANSAC iterations by half and still have the same inlier count as when the distance constraint was not used.

The indoors sequence was chosen to demonstrate an important aspect of the track-retrack scheme: As can clearly be seen in the graphs, the improvement caused by track-retrack is much bigger in the indoor sequence than in the others. The reason for this is that track-retrack is very effective at detecting outliers caused by occlusions.

The same test is run for the synthetic test data, see figure 4. The behaviour here is nearly identical to the evaluation with real images, and we can also observe that we can reduce the number of RANSAC steps by approximately half with maintained inlier/outlier ratio when adding the distance constraint.

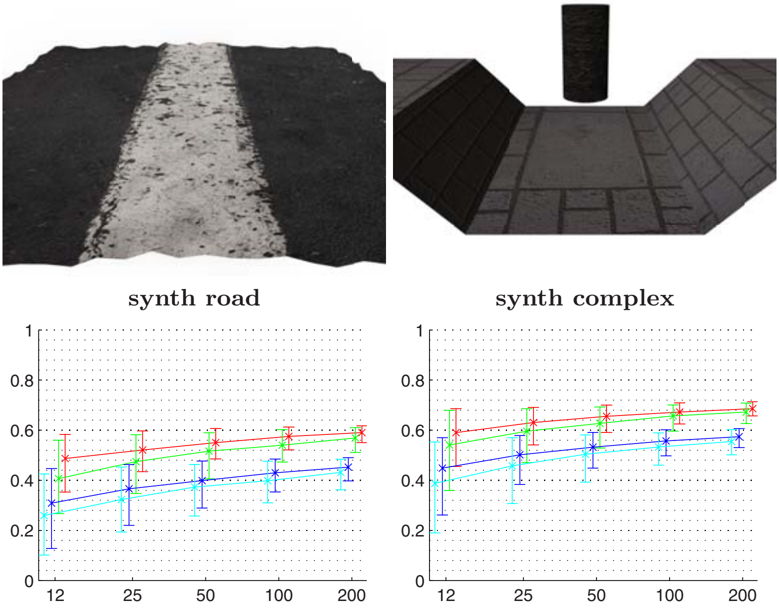


Fig. 4. The synthetic sequences and their inlier/outlier ratios, as function of number of RANSAC iterations. Same legend as figure 3.

The first scene consists of a forward motion on a planar road and the second is also forward motion but on a more complex scene. For each synthetic sequence, we have also generated images of calibration patterns. This allows us to process the synthetic sequences in exactly the same manner as the real ones.

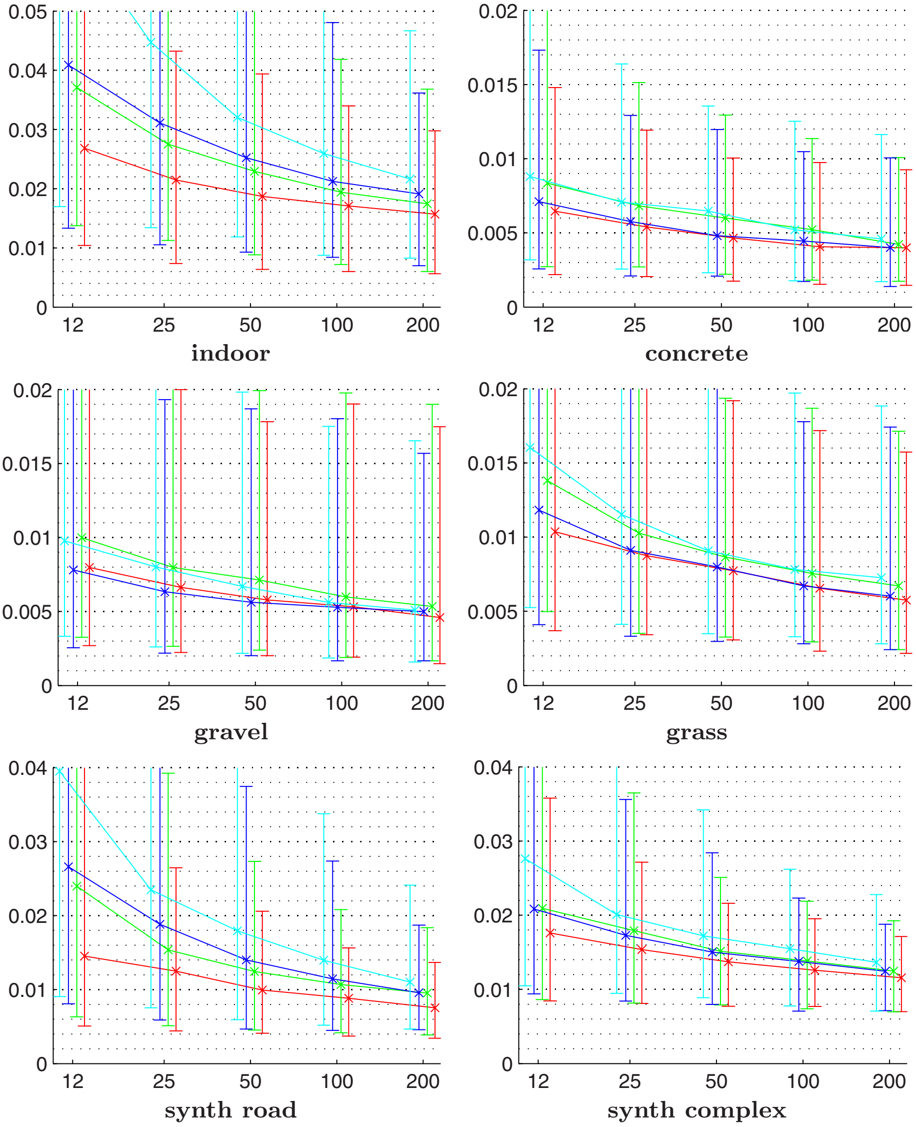


Fig. 5. Position errors on real and synthetic sequences as function of number of RANSAC iterations. The sequences are evaluated against bundle adjustment output on the same data set. The synthetic sequences are evaluated against their ground truth. The four methods are coloured as in figure 3.

3.2 Precision Evaluation

As calibrated monocular reconstructions are only defined up to scale, accuracy evaluation is rather problematic. We have chosen to evaluate the accuracy of the obtained SaM solutions using the position error of the third camera. For this to be possible, we need first to adjust the distance of the second camera to be the same as in the ground truth (here we set this distance to be 1). Only after this normalisation are we able to compare positions of the third camera.

In the absence of real ground truth on the real sequences, we have used the output of the bundle-adjustment (BA) algorithm described in [26]. Bundle-adjustment is the maximum likelihood estimate of SaM, and has been shown to greatly improve the results [27]. On the synthetic sequences we simply save the camera locations used for generating the frames. Note that we get similar results on both real and synthetic sequences, which supports the use of BA for evaluation purposes.

Figure 5 shows the absolute position error of the third camera in each triplet. Here we can see that the position error follows the same trend as the inlier/outlier ratio. If anything the improvement is even more pronounced here.

We have summarised the results of the precision experiments in table 1 and 2. Table 1 shows the speed increase (i.e. reduction in number of RANSAC samplings needed to obtain the same precision) from the distance constraint, when the track-retrack step is disabled, and table 2 shows the speed increase with track-retrack enabled.

The percentages are obtained as follows: For each number of iterations, we look up the precision with the distance constraint enabled. We then estimate how many iterations it would take to archieve the same precision when the constraint is not used. The estimate is found through linear interpolation (note although

Table 1. Speed increase from the distance constraint, with track-retrack disabled (i.e. CYAN curve vs. BLUE)

#iterations	synt. r.	synt. c.	indoor	concrete	gravel	grass
12	180%	190%	230%	198%	216%	193%
25	184%	198%	215%	256%	234%	199%
50	199%	225%	216%	265%	199%	187%
100	185%	194%	>200%	>200%	162%	>200%

Table 2. Speed increase from the distance constraint, with track-retrack enabled (i.e. GREEN curve vs. RED)

#iterations	synt. r.	synt. c.	indoor	concrete	gravel	grass
12	229%	211%	215%	243%	201%	197%
25	198%	191%	242%	277%	244%	195%
50	264%	209%	235%	258%	232%	183%
100	>200%	191%	>200%	>200%	>200%	>200%

the graphs are in log scale, the interpolation is done on a linear scale). The speed increase is now the ratio of the two iteration counts. There are some cases where the intersection point lies after the 200 iteration value, these values could have been extrapolated, but we have instead chosen to just show them as >200.

Note that a speed increase computed in this way does not take the extra overhead of the sampling into account. But, as shown in section 2.4 this overhead is 0.6-1.2% of the total time, and as mentioned, there are ways to reduce this even further.

4 Conclusions

We have introduced a method that significantly speeds up the current state of the art SaM algorithm for calibrated cameras. On average we are able to reduce the number of RANSAC iterations by half and thereby double the speed. The improvement is achieved by adding a distance constraint to the point selection inside the RANSAC loop. We have also added an outlier rejection step (which we call track-retrack) to the KLT tracker. For scenes with high level of occlusion the track-retrack scheme also gives a similar improvement in performance. For scenes with little or no occlusion, however, the difference is negligible. Note also that the two improvements are independent, for scenes where the track-retrack scheme gives and improvement, we will get further improvement by adding the distance constraint.

Acknowledgments

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement no 215078, DIPLECS.

References

1. Hartley, R., Zisserman, A.: Multiple View Geometry in Computer Vision, 2nd edn. Cambridge University Press, Cambridge (2003)
2. Pollefeys, M., et al.: Detailed real-time urban 3D reconstruction from video. International Journal of Computer Vision 78, 143–167 (2008)
3. Martinec, D., Pajdla, T.: Robust rotation and translation estimation in multiview reconstruction. In: IEEE CVPR 2007 (2007)
4. 2d3 Ltd., B. (2009), http://www.2d3.com
5. Various: Special issues on the 2007 DARPA urban challenge, parts I- III. In: Buehler, M., Iagnemma, K., Singh, S. (eds.) Journal of Field Robotics, vol. 25. Wiley, Blackwell (2008)
6. Vedaldi, A., Guidi, G., Soatto, S.: Moving forward in structure from motion. In: IEEE International Conference on Computer Vision (2007)
7. Nistér, D.: An efficient solution to the five-point relative pose problem. IEEE TPAMI 6, 756–770 (2004)

8. Källhammer, J.-E., et al.: Near zone pedestrian detection using a low-resolution FIR sensor. In: Intelligent Vehicles Symposium, IV 2007 (2007)
9. Baker, S., Scharstein, D., Lewis, J.P., Roth, S., Black, M.J., Szeliski, R.: A database and evaluation methodology for optical flow. In: IEEE ICCV (2007)
10. Longuet-Higgins, H.: A computer algorithm for reconstructing a scene from two projections. Nature 293, 133–135 (1981)
11. Fischler, M., Bolles, R.: Random sample consensus: a paradigm for model fitting, with applications to image analysis and automated cartography. Communications of the ACM 24, 381–395 (1981)
12. Tordoff, B., Murray, D.: Guided sampling and consensus for motion estimation. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002. LNCS, vol. 2350, pp. 82–96. Springer, Heidelberg (2002)
13. Zhang, W., Kosecka, J.: A new inlier identification scheme for robust estimation problems. In: Robotics Science and Systems RSS02 (2006)
14. Chum, O., Matas, J.: Randomized RANSAC with t(d, d) test. In: BMVC 2002, pp. 448–457 (2002)
15. Nistér, D.: Preemptive RANSAC for live structure and motion estimation. Machine Vision and Applications 16, 321–329 (2005)
16. Myatt, D., et al.: NAPSAC: High noise, high dimensional model parametrisation - it's in the bag. In: BMVC 2002 (2002)
17. Wu, H., Chellappa, R., Sankaranarayanan, A.C., Zhou, S.K.: Robust visual tracking using the time-reversibility constraint. In: IEEE ICCV (2007)
18. Barron, J.L., Fleet, D.J., Beauchemin, S.S.: Performance of optical flow techniques. International Journal of Computer Vision 12, 43–77 (1994)
19. Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: IJCAI 1981, pp. 674–679 (1981)
20. Shi, J., Tomasi, C.: Good features to track. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 1994, Seattle (1994)
21. Ringaby, E.: Optical flow computation on compute unified device architecture. In: Proceedings of SSBA 2009 (2009)
22. Sinha, S.N., Frahm, J.M., Pollefeys, M., Genc, Y.: GPU-based video feature tracking and matching. Technical report, UNC Chapel Hill (2006)
23. Hedborg, J., Skoglund, J., Felsberg, M.: KLT tracking implementation on the GPU. In: Proceedings SSBA 2007 (2007)
24. Kirk, D., Hwu, W.-m.W.: The CUDA programming model (2007)
25. Open source computer vision software library, O.A. (2009), `http://opencv.willowgarage.com/`
26. Lourakis, M., Argyros, A.: 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 (2004)
27. Engels, C., Stewénius, H., Nistér, D.: Bundle adjustment rules. In: Photogrammetric Computer Vision, PCV (2006)