Parallel, Real-Time Visual SLAM

Brian Clipp[1], Jongwoo Lim[2], Jan-Michael Frahm[1] and Marc Pollefeys[1,3]

Department of Computer Science[1]
The University of North Carolina
Chapel Hill, NC, USA
{bclipp,jmf}@cs.unc.edu

Honda Research Institute USA, Inc.[2]
Mountain View, CA, USA
jlim@honda-ri.com

Department of Computer Science[3]
ETH Zurich, Switzerland
marc.pollefeys@inf.ethz.ch

Abstract— **In this paper we present a novel system for real-time, six degree of freedom visual simultaneous localization and mapping using a stereo camera as the only sensor. The system makes extensive use of parallelism both on the graphics processor and through multiple CPU threads. Working together these threads achieve real-time feature tracking, visual odometry, loop detection and global map correction using bundle adjustment. The resulting corrections are fed back into to the visual odometry system to limit its drift over long sequences. We demonstrate our system on a series videos from challenging indoor environments with moving occluders, visually homogenous regions with few features, scene parts with large changes in lighting and fast camera motion. The total system performs its task of global map building in real time including loop detection and bundle adjustment on typical office building scale scenes.**

I. Introduction

In recent years the visual simultaneous localization and mapping (VSLAM) problem has become a focus of the robotics and vision communities. This effort has been made possible by advances in camera hardware and the computational power available in a personal computer. In this paper we introduce a novel, real-time system for six degree of freedom visual simultaneous localization and mapping. It operates in real-time at more than 15 frames per second by leveraging a combination of data parallel algorithms on the GPU, parallel execution of compute intensive operations and producer/consumer thread relationships that effectively use modern multi-core CPU architectures.

A particular problem for visual navigation is posed by indoor environments due to their lack of salient features. A combination of local tracking and global location recognition enables our system to robustly operate in these environments. The system is demonstrated on two challenging indoor sequences that include sections with very few salient features to track because of large textureless regions. To overcome inherent drift problems from local feature tracking the system detects loops once it re-enters an area it has mapped before using SIFT [11] features. The loop closing mechanism additionally enables re-initialization into the global model after local tracking failure. We demonstrate the improvement in the maps after loop detection and loop completion in comparison to using only visual odometry, which does not detect loops.

The remainder of this paper begins with a discussion of related work in section II. Section III introduces the system from an algorithmic perspective while section IV focuses on implementation details. Experimental results are given in section V followed by our conclusions.

II. Background

The robotics and vision communities have made great strides toward solving the visual SLAM problem in recent years. Davison first presented a real-time VSLAM system in 2003 based on an Extended Kalman filter [4]. That system performed very well in cubicle scale environments but in larger environments the cubic scaling complexity of the extended Kalman filter prevented real time operation.

A particle filter approach to VSLAM was presented by Eade and Drummond [6]. Their system could also map small office scale environments but the small number of particles that could be processed in real time limited their map size.

More recent approaches to real-time VSLAM have focused on dealing with the complexity of updating a globally consistent map. Clemente et al. [2] proposed a sub-map approach where the total map is made up of a set of smaller metric maps connected by transformations. Since each metric map is limited in size it can be updated in real-time. A major drawback of this approach is that global correction is done by fixing the sub-maps and varying the transformations between them. This forces all of the error accumulated in each of the sub-maps into the joints between the maps.

Mei et al. in [12] presented a system using a relative map representation. Each pair of camera poses that see a 3D feature in common is connected by a transformation in a graph structure. The 3D features are attached to the coordinate frame of the first camera that measures them. With this structure a globally topological but locally metric map is obtained. Since it does not perform global map updates, maps of arbitrarily large environments can be computed in real-time.

In [10] Konolige and Agrawal presented a real-time VSLAM method including visual odometry and global map correction. They addressed scaling issues in global map correction by limiting the number of key-frames in the map and converting constraints from measurements into probability distributions over inter-camera transformations. Key-frame poses are updated using these distributions as constraints in a non-linear minimization. While addressing scaling issues, converting measurements to distributions on transformations trades accuracy for speed.

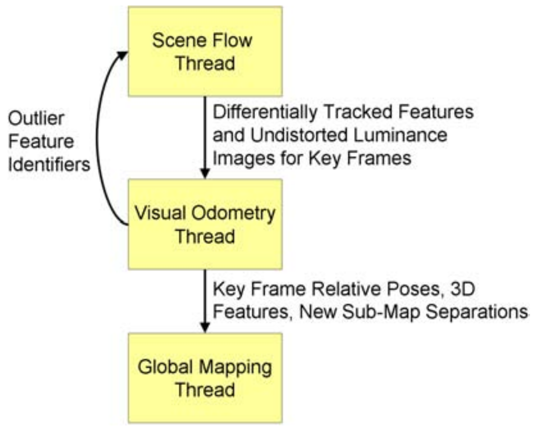
Scene Flow Thread
Outlier Feature Identifiers
Differentially Tracked Features and Undistorted Luminance Images for Key Frames
Visual Odometry Thread
Key Frame Relative Poses, 3D Features, New Sub-Map Separations
Global Mapping Thread

Fig. 1. The main threads of the system architecture.

The most closely related work to ours is Klein and Murray's parallel tracking and mapping [9] (PTAM). Their system is designed for use in augmented reality in a small working volume such as a desk or in front of a building. In PTAM one thread estimates the camera pose with respect to an existing map in real time while a second thread updates the underlying map which is a reduced set of key frames and 3D features. Our application is different in that we explore office building scale environments and map them in real time. However, we do not perform a full 6DOF pose estimate for every frame of the video as they do.

III. System Description

Our parallel, real-time VSLAM system is composed of three primary modules: Scene Flow (SF), Visual Odometry (VO) and Global-SLAM (GS) as shown in Figure 1. The Scene Flow module calculates the sparse optical flow and selects key-frames based on the magnitude of the average flow. It then passes the key-frames and the tracks to the Visual Odometry module, which calculates the inter-keyframe motion and passes this motion as well as the 3D features to the Global-SLAM module. The Global-Slam module then performs loop detection, and global error correction of the map based on the detected loops. The final result of our method is a globally consistent sparse 3D model of the environment made up of 3D feature points and camera poses for the key-frames.

A. Scene Flow Module

To determine the local motion of the camera we track features from frame to frame using multi-camera scene flow proposed by Devernay et al. in [5], which is an extension of differential KLT tracking into three dimensions. To meet the real-time goal our system uses an efficient GPU based implementation. In multi-camera scene flow features are first extracted using the approach of Tomasi and Shi [16] and then matched from left to right image enforcing both the epipolar constraint and cross-validating feature matches to eliminate outliers. After the features are matched they are triangulated to establish 3D positions for the inlier features. At this point features are tracked as small 3D planar surface patches in front of the cameras. The feature motion in 3D is determined through the temporal image flow of the 2D features in the stereo cameras. Using this parametrization the epipolar constraint is enforced without resorting to stereo matching a feature in each stereo image.

Given the varying temporal redundancy in the video which is mainly due to camera motion, we adaptively select key-frames through a threshold on the minimum average optical flow of features since the last key-frame. To minimize costly feature detection, detection is only performed in the selected key-frames with the additional constraint that if too few features are tracked another key-frame occurs. Hence images with small camera motion are not taken as key-frames. The 2D feature tracks and the triangulated 3D points are then passed to the Visual-Odometry module.

B. Visual Odometry Module

The stereo camera system enables estimation of the 3D points in the Scene Flow module. Therefore, we use an approach along the lines of Nistér [14] to determine the camera motion. Our method uses the three point perspective pose method [8] in a RANSAC [1] framework to determine the camera pose using tracks from the Scene Flow module. While this method is sufficient for tracking the differential camera motion it accumulates small inaccuracies over time, which theoretically lead to an unbounded drift.

To counter the drift our system detects camera path intersections using SIFT features [11]. SIFT features can be matched over large changes in viewpoint, in contrast to differentially tracked KLT-features. To boost performance we use a CUDA based GPU implementation [17]. In addition to using the SIFT features for loop detection we also use them in refining the incremental motion estimation. This refinement is performed using a windowed bundle adjustment [7] delivering refined camera poses and more accurate 3D points than delivered by the scene flow from Section III-A. In our windowed bundle adjustment a window spanning the the last n key-frame poses is optimized. The oldest two key-frame poses are held fixed while the youngest $n-2$ key-frame poses are varied along with all of the 3D features, both SIFT and KLT. The bundle adjustment uses a robust cost function so that outliers have a limited influence on the result.

Combining the refined camera motion estimate based on KLT feature tracks with the 3D position of the SIFT features we can predict where the SIFT features should project into the current key-frame. We use this prediction to our advantage by limiting the candidate matches to close by SIFT features in the current key-frame. The benefits of this are twofold. We are less prone to problems caused by repetitive structures and given the smaller number of potentially matching features we can reduce the number of SIFT-descriptor comparisons. Further, we empirically found that this prediction allows us to relax Lowe's SIFT matching uniqueness criteria [11] but still be robust to repetitive structures in the scene.

Following predictive SIFT matching we match the remaining unmatched SIFT features from left to right images in the current key-frame using the stereo camera's calibration to constrain the search for matches along the epipolar lines. These matches are then triangulated and un-matched SIFT features are discarded.

At this point the newest key-frame has been completely incorporated into the local map. It will be considered until it leaves the bundle adjustment window or the visual odometry fails and a new sub-map is started. Please note that as soon as a frame has an initial pose in the visual odometry module its 3D pose w.r.t the global map can be found. This pose will be locally accurate and will be refined through the windowed bundle adjustment. The pose may be changed when loops are detected in the global SLAM module but this should not affect tasks such as obstacle avoidance. After exiting the bundle adjustment window key-frames are processed by the Global SLAM module.

C. Global-SLAM Module

The Global-SLAM module ensures global consistency in our VSLAM system. It incorporates the information of all currently available key-frame poses, feature measurements and initial 3D feature estimates from the Visual Odometry module. The final result is a set of globally consistent, metric sub-maps each of which has its own global coordinate frame. The sub-maps are disjoint, meaning that they cover separate areas in the environment or cannot be linked by common 3D features due to limitations of wide-baseline feature matching.

The key element to improve the incremental motion estimation provided by the Visual-Odometry module is the detection of loop completions. Loop completions provide additional constraints to the local constraints found in the VO module. Our system uses the vocabulary tree [15] based approach to detect loops. Please note, any alternative approach like the Fab-Map [3] approach of Cummins and Newman could be used instead. In our approach SIFT feature descriptors are quantized into visual words using a K-d tree over a descriptor space which is pre-computed. The visual words seen in an image are then organized so that one can find out quickly, which images a visual word is seen in. Finding similar images to a query image is then as simple as computing a vote to determine in what other images a query image's visual words are found. In the vote higher weight is given to the more discriminative visual words that are found less frequently.

The Global SLAM module can operate in one of two modes. When exploring new areas the system operates in *loop seeking mode* while in previously mapped regions the system operates in *known location mode*.

1) Loop Seeking Mode: Loop seeking mode performs loop detection for each new key frame and after a successful loop identification a global refinement is computed through bundle adjustment. Loop detection begins by using the vocabulary tree to find a list of the most similar images to the current key-frame sorted by similarity. Images from recent key-frames are removed from the list so that loops are only found to older sections of the map. Images in the list are tested in order of similarity until a matching image is found or the similarity score of the next best match is too low.

Rather than simply match SIFT features from the query image to those visible in the next most similar image we use the putative matching image to find a region of local 3D scene structure and match the query image to this structure. This can be seen as a form of query expansion based on 3D geometry. The expansion is done by finding the images near the next most similar image and including all of the 3D features visible in all of these images in the SIFT matching and geometric verification. The SIFT matching is then performed from the image to the 3D structure. SIFT descriptor matching is performed from the descriptors of the features in the current key-frame to the 3D features' descriptors. We only try to match SIFT descriptors with the same associated visual word, which reduces the number of descriptor dot products performed. A RANSAC process using the three-point perspective pose method is then used to find the pose of the current camera and the pose is non-linearly optimized afterwards.

If the above method finds a solution supported by enough inlier matches it is considered a loop. The features associated with the inlier measurements to the RANSAC are linked so that they are treated as a single feature in bundle adjustment. Using 3D feature to 2D projection matching with geometric verification makes false positive loop detections much less likely than using an image to image matching approach. Still truly repetitive 3D structures can cause incorrect loops to be detected. Dealing with repetitive structures remains an open research problem.

If no loop has been detected then the next key-frame is tested for a potential loop closing. If a loop was detected the system performs a global correction to the current sub-map incorporating the newly detected loop. Since the newly detected loop features have high reprojection errors in the current key-frame they would be deemed invalid by our bundle adjustment which uses a robust cost function. Hence they would not influence the error mitigation process. To overcome this effect we re-distribute the error before bundle adjustment. This initializes the bundle adjustment much closer to the global minimum of its cost function, increasing its convergence rate and decreasing the chance of converging to a local minimum.

We re-distribute the accumulated error by starting with the difference in the current key-frame pose and the current key-frame's pose calculated w.r.t. the old features. This gives us the amount of drift that the system has accumulated since it left the last known location in the sub-map. This last known location is either the first frame in the sequence if no loops have been found so far or the last place the system was operating in known location mode. The system is operating in known location mode when it has reacquired features it has mapped before and is tracking with respect to that known map. The system linearly distributes the error correction for the cameras back to the point it was operating in known location mode. Spherical linear interpolation of the rotation error quaternion is used to interpolate the rotation error. Feature points are similarly corrected by moving them along with the camera that first views them. A global bundle adjustment of the map is then performed. After bundle adjustment outlier measurements are removed as well as features visible in fewer than two key-frames. These features give little information about the scene structure and are more likely to be incorrect since they do not match the camera's motion. After successfully detecting the loop and correcting the accumulated error the Global-SLAM module enters known location mode.

2) Known Location Mode: After successfully identifying a loop this mode continuously verifies that the robot is still moving in the previously mapped environment. Verification is done by linking the current 3D SIFT features to previously seen 3D SIFT features in the environment surrounding the current location. These matches are added to a windowed bundle adjustment in the GS module which keeps the camera path consistent with the older previously computed parts of the map.

In the known location mode SIFT feature matching between the current key-frame and the old 3D SIFT features is done using the predictive approach described in the visual odometry module (see Section III-B). Older features can be linked to the features visible in the current frame by projecting all of the 3D SIFT features seen in the previous key-frame and it's neighboring images (two key-frames are neighbors if they see the same 3D feature) and comparing descriptors. If no matching older SIFT features are found then the robot has left the previously observed parts of the environment and the system reenters the "Loop Seeking" mode.

The windowed bundle adjustment in GS is much the same as the one performed in the Visual Odometry module. The only difference in this case is that the older key-frames are also included in the bundle but fixed. This ensures that the new camera poses stay consistent with the existing map. Fixing the older cameras is also justified since they have already been globally bundle adjusted and so are probably more accurate than the more recent key-frames. After the windowed bundle adjustment processing begins on the next key-frame.

IV. Implementation Details

A key to the performance our system is that each of the three modules Scene Flow (SF), Visual Odometry (VO) and Global-SLAM (GS) operates independently and in parallel. To ensure that all captured information is used only the Scene Flow module has to operate at frame-rate. The timing constraints on the visual odometry are dynamic and only depend on the frequency of key-frames. This module can lag behind by a few frames. The Global SLAM module is less time constrained since its corrections can be incorporated into the local tracking when they are available. The system's modules operate in separate threads that each adhere to the individual module timing requirements.

A. Scene Flow Module

The scene flow module begins by taking raw, bayer pattern images off of the stereo cameras. These images must be converted to luminance images and radially undistorted before the sparse scene flow can be measured. We use color cameras so that the video we record can later be used for dense stereo estimation and 3D modeling. While tracking could be performed on radially distorted images, we remove the radial distortion from the images so that later SIFT feature extraction in the Visual Odometry module can be done on undistorted images. Using undistorted images helps in SIFT matching when using cameras with a large amount of radial distortion.

De-mosaicing, radial undistortion and sparse scene flow are all calculated on the graphics processing unit (GPU) using CUDA. To increase performance we minimize data transfer between CPU to GPU by downloading the raw image to GPU for each frame, performing all computations in GPU memory and then only uploading undistorted images to the CPU for the key frames as well as the tracked feature positions.

After each key-frame the feature tracks (2D position and feature identifier) and the undistorted images are passed to the Visual Odometry module. While the Visual Odometry module processes the key frame the Scene Flow thread can track ahead of it, buffering new key frames until the Visual Odometry module is able to process them. Hence the speed of Visual Odometry does constrain the Scene Flow module's real-time performance. This is just one example of how parallelism adds robustness to our system.

B. Visual Odometry Module

In this module we perform the incremental motion estimation from the KLT-features tracks and the detection of SIFT features in parallel. For efficiency we use one thread for each of the two stereo images. After the SIFT detection we release the image buffers to save memory.

As described in Section III-B the Visual Odometry module's outputs are the relative camera motion and the new 3D points. These outputs are stored in a queue and are removed from Visual Odometry's local storage. Using a queue decouples processing in the VO and GS module threads. Whenever tracking fails all the VO module's internal data (key-frame poses and 3D features) is queued for processing by the Global SLAM module.

V. Experimental Results

In order to demonstrate the speed, accuracy and long term stability of our VSLAM system we present results from two video sequences of two indoor environments with different characteristics. The first sequence was taken in an office environment which has a large, open floor plan. The second hallway sequence was shot in a building with long, but relatively narrow (1.7m) hallways. The closed floor plan does not allow features to be tracked for long periods of time since they quickly leave the stereo camera's field of view, yet the system successfully maps the halls accurately with an error of less than 30cm over the 51.2 m length of the longest hall shown in Figure 9. This is an error of less than 0.6%.

Our setup uses a calibrated stereo camera pair consisting of two Point-Grey Grasshopper cameras with 1224×1024 pixel resolution color CCD sensors delivering video at fifteen frames (stereo pairs) per second. The system's 7cm baseline is comparable to the median human inter-pupil distance. The cameras are mounted on a rolling platform with the computer. Using a rolling platform the planarity of the camera path can be used to evaluate the quality of the reconstruction results. However, the full six degrees of freedom are estimated for the camera's motion. While performing real-time VSLAM the system also records the imagery to disk for debugging or archival purposes.

The office sequence includes transparent glass walls and other reflective surfaces that make tracking more challenging (please see Figure 2 for example frames). It also has a hallway with relatively low texture which our system successfully maps, showing it is robust to areas without a large amount of structure. In one section of the video a person moves in front of the camera, partially occluding some of the tracked features (see Figure 2 middle lower panes). Even in this case the system is able to reject the moving person's feature tracks as outliers and continue tracking correctly.

Figure 3 shows the difference between operating only using visual odometry and performing the full VSLAM with loop detection and global map correction. In the left of Figure 3 the map is shown using only visual odometry where the relative motion from frame to frame is accumulated to form the camera path. In visual odometry no loop detection of global map correction is performed hence the system drifts over time. In this scene VO accumulated drift of approximately 3m over an approximately 150 meter path. In the right pane the results of our global SLAM module are shown. Clearly, the long term drift of visual odometry is eliminated by loop detection and the succeeding error mitigation through bundle adjustment.

Additionally, in Figure 4 we show the vertical drift using only visual odometry of approximately 2 meters over a traveled distance of 70 meters. Figure 4 shows two side views of the map without (left) and with (right) loop detection and global map correction. In the left pane the accumulated vertical error of 2.0 meters is clearly visible while in the right pane it is eliminated. Note the regular pattern on the ground in the right pane that reflects the repetitive pattern in the carpet there. We have overlaid the results of our system with loop detection and correction over an architectural layout of the office as illustrated in Figure 5. This figure demonstrates the accuracy of our system. The results shown here were processed from 8304 stereo frames of video at fifteen frames per second including multiple loop detections and global bundle adjustments, one for each time a loop was detected.

We use the hallway sequence to demonstrate our system working in a less open environment where feature tracks are typically shorter and fewer in number. As shown in the panes of Figure 7 some of the hallways have large amounts of texture while others are largely textureless. The system robustly performs camera tracking in either case, at times tracking with fewer than ten features.

Figure 9 shows an architectural drawing of the building's hallways with dimensions. From the overlay of our model on top of the floor plan it is clear that our system creates accurate maps. Taking the difference between the measured center to center distance of the longest mapped hallways on the figure and the distances between comparable camera centers in our reconstructed map we find an error of 30cm in the horizontal direction and 40cm in the vertical direction in the figure. These errors equate to 0.6% error in the figure's horizontal direction and 1.4% in the vertical.

The three modules' timing results on this sequence are shown in Figure 6. Note the four spikes in the global SLAM module's processing time. These are times when loops were detected and the map was bundle adjusted. In the Scene flow graph the spikes in processing time occur when new features are detected. In a typical frame 200-500 scene flow features are tracked. In the global SLAM module there are typically 60-120 scene flow features and 100-200 SIFT features which are inliers to the motion model. Reprojection errors in the Global SLAM module's results average approximately 0.6 throughout the sequence.

Another example of loop completion is illustrated in Figure 8 showing the map before loop detection and correction on the left pane and the right pane gives the result after loop detection. In these panes the camera returned to the known area from the right. Note how the accumulated error in the camera poses and features is eliminated by the loop detection and global map correction. Finally a side view of the hallway map is shown in figure 10. The side view demonstrates the planarity of the map which matches the flat floor of the halls. The accompanying video shows the operation of our system on the hallway data set.

VI. Conclusion

In this paper we introduced a VSLAM system that fully exploits the recent parallelism gains in consumer computer hardware. The system uses imagery from a stereo camera as its only input. We implemented a two-view consistent 2D tracking module on the GPU to find sparse optical flow. We also used the GPU to extract wide-baseline features for use in loop detection. Our system exploits modern multi-core processors using multiple concurrent threads to perform sparse scene flow, visual odometry and global mapping in parallel. This parallelism allows us to perform the full metric map reconstruction in real time.

While our system operates in real time on office building scale scenes, extension to larger environments remains a challenge. In the future we will investigate hierarchical bundle adjustment methods like [13] to address scaling issues. This would allow us to optimize several sub-problems that are mutually very weakly dependent on each other in parallel. After that a global combination step could create a globally consistent map.

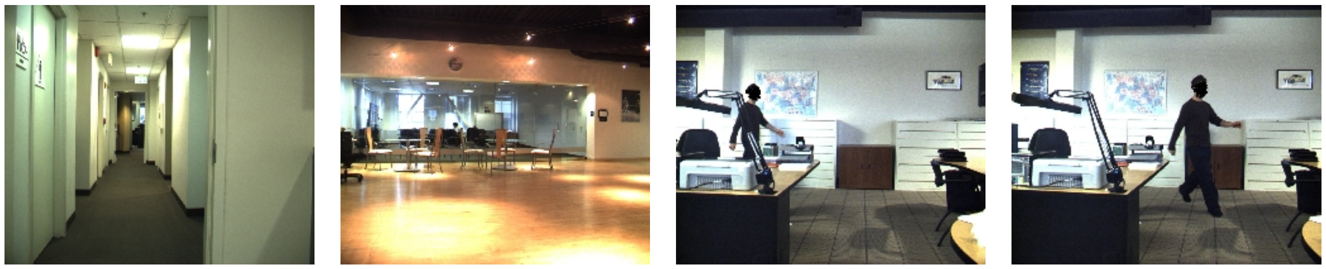

Fig. 2. Sample frames from the left camera of the stereo pair for the office sequence. Note the reflective glass walls, textureless regions and moving person in the images.

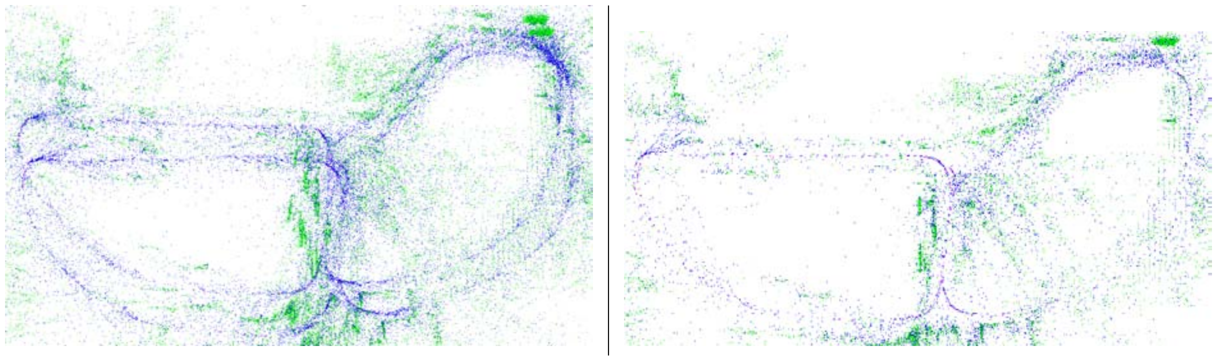

Fig. 3. Results of office sequence top view. Left: Camera path calculated using visual odometry only. Right: Camera path calculated loop detection and correction global slam module.

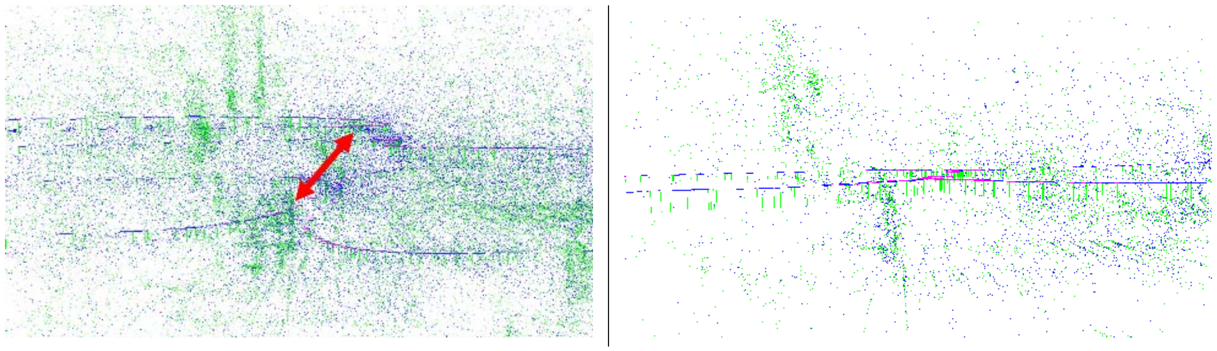

Fig. 4. Results of office sequence side view. Left: Camera path calculated using visual odometry only. Right: Camera path calculated loop detection and correction global slam module. Note that the path processed through the global slam module is planar where visual odometry has large vertical error accumulation as shown by the red arrow.

References

[1] R. Bolles and M. Fischler. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981.

[2] L. Clemente, A. Davison, I. Reid, J. Neira, and J. Tardos. Mapping large loops with a single hand-held camera. In *Robotics: Science and Systems*, June 2007.

[3] M. Cummins and P. Newman. FAB-MAP: Probabilistic Localization and Mapping in the Space of Appearance. *The International Journal of Robotics Research*, 27(6):647–665, 2008.

[4] A. Davison. Real-time simultaneous localisation and mapping with a single camera. In *ICCV*, volume 2, pages 1403–1410, Oct. 2003.

[5] F. Devernay, D. Mateus, and M. Guilbert. Multi-camera scene flow by tracking 3-d points and surfels. In *CVPR*, 2006.

[6] E. Eade and T. Drummond. Scalable monocular slam. In *CVPR*, 2006.

[7] C. Engels, H. Stewenius, and D. Nister. Bundle adjustment rules. In *Photogrammetric Computer Vision*, September 2006.

[8] R. Haralick, C. Lee, K. Ottenberg, and M. Nolle. Review and analysis of solutions of the three point perspective pose estimation problem. *IJCV*, 13(3):331–356, December 1994.

[9] G. Klein and D. Murray. Improving the agility of keyframe-based slam. In *ECCV*, 2008.

[10] K. Konolige and M. Agrawal. Frameslam: From bundle adjustment to real-time visual mapping. *IEEE Transactions on Robotics*, 2008.

[11] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, November 2004.

[12] C. Mei, G. Sibley, M. Cummins, P. Newman, and I. Reid. A constant time efficient stereo slam system. In *BMVC*, 2009.

[13] K. Ni, D. Steedly, and F. Dellaert. Out-of-core bundle adjustment for large-scale 3d reconstruction. In *ICCV*, pages 1–8, 2007.

[14] D. Nistér, O. Naroditsky, and J. Bergen. Visual odometry. In *CVPR*, 2004.

[15] D. Nistér and H. Stewénius. Scalable recognition with a vocabulary tree. In *CVPR*, pages 2161–2168, 2006.

[16] C. Tomasi and J. Shi. Good features to track. In *CVPR*, 1994.

[17] C. Wu. SiftGPU: A GPU implementation of scale invariant feature transform (SIFT). `http://cs.unc.edu/~ccwu/siftgpu`, 2007.

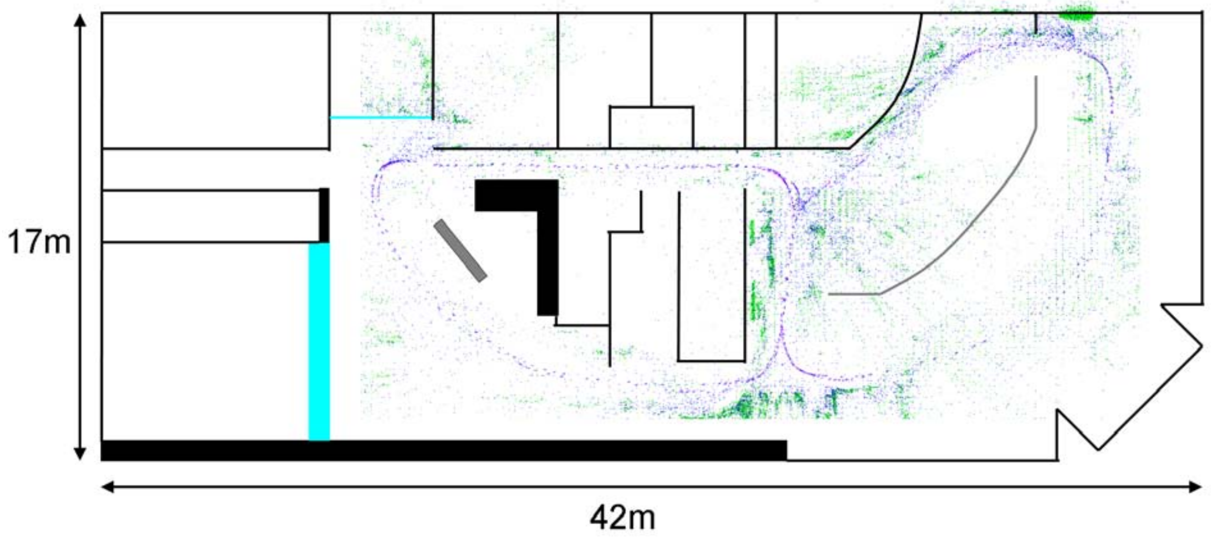

Fig. 5. Results of office sequence. Camera path from global SLAM module using bundle adjustment for loop correction. The camera made two complete passes around both loops. Blue walls are glass. Grey walls are half ceiling height partitions.

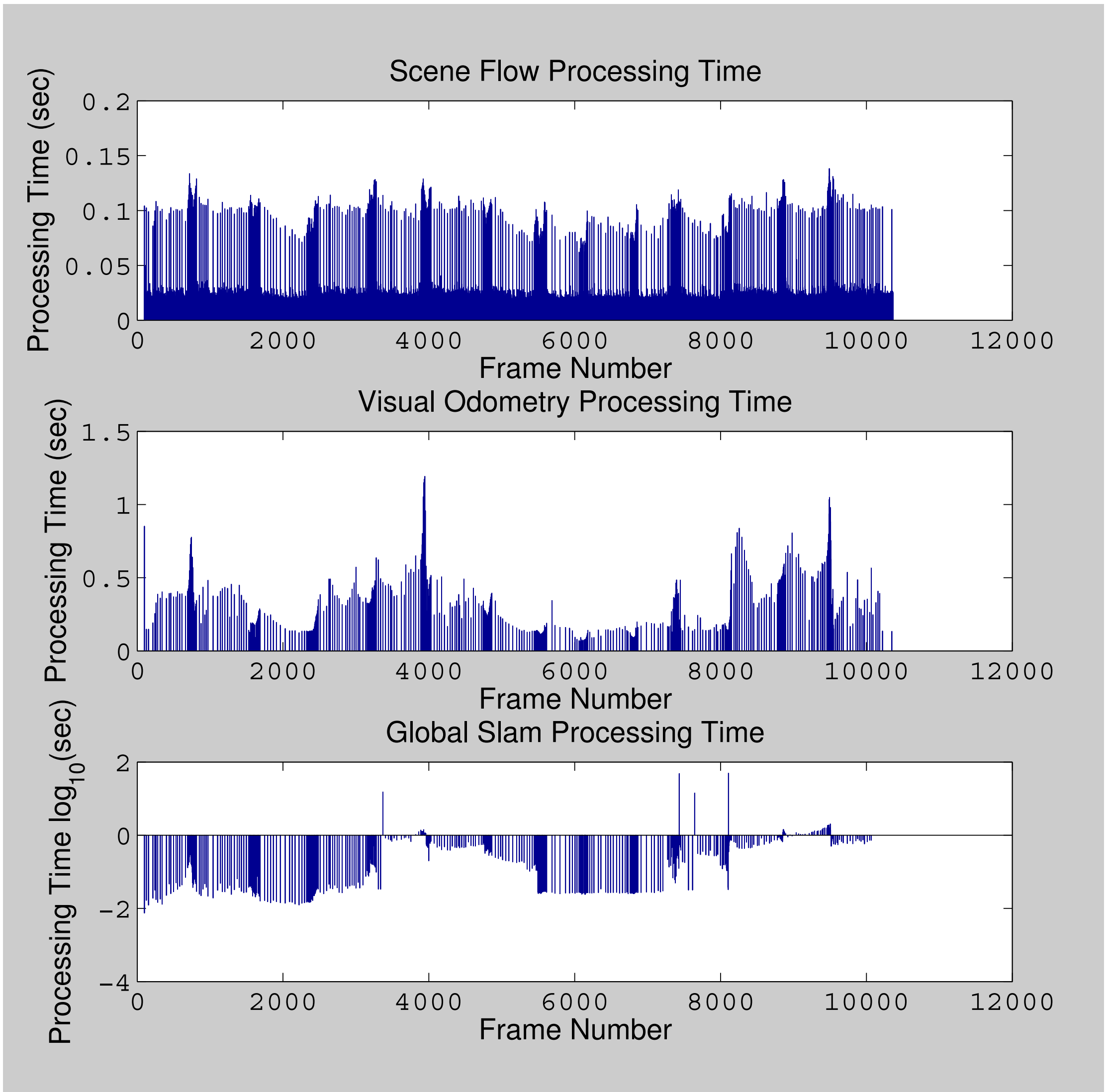

Fig. 6. Timing results on the hallway sequence. Note the processing time spikes in the scene flow module when new features are extracted. The four large spikes in Global SLAM processing time are caused by bundle adjustments after loop completions. The average scene flow processing rate is 37.1 frames per second (fps), visual odometry is 61.0 fps and global SLAM is 33.5 fps. The VO and Global SLAM processing rates are calculated as total processing time/video frames (not key-frames).

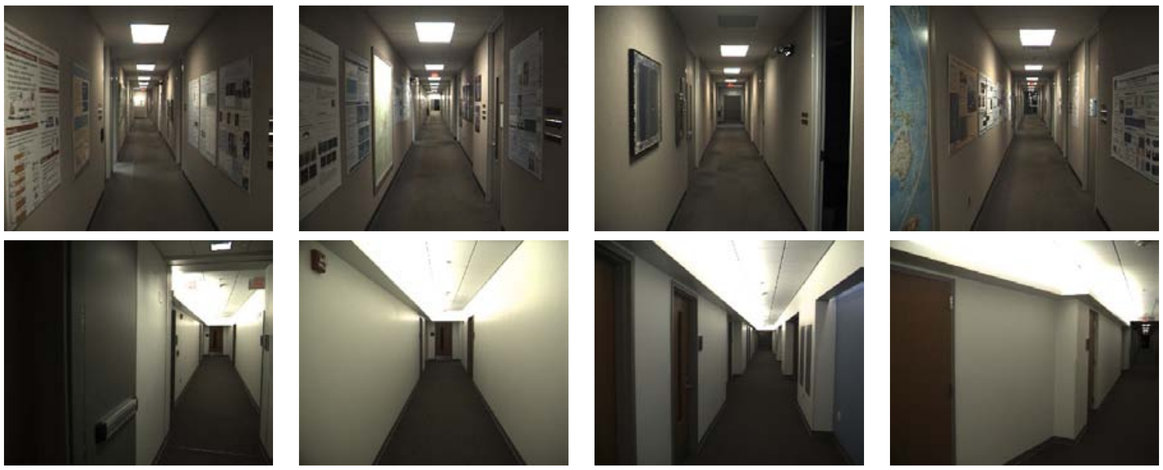

Fig. 7. Sample frames from the left camera of the stereo pair for the hallway sequence. Note the lack of texture in some images and the forward motion which makes visual odometry more challenging than if the camera was pointing to the side.

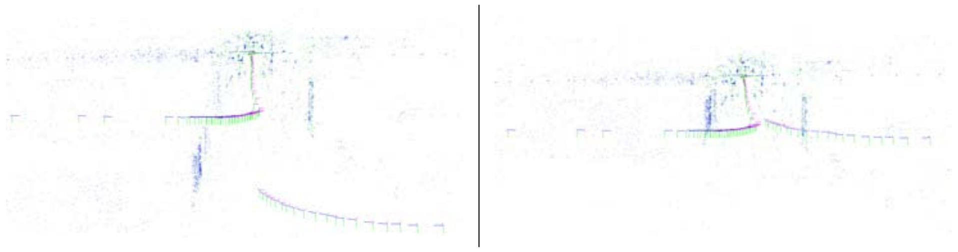

Fig. 8. Results of large building sequence. Left: Camera path intersection before loop detection and global map correction. Right: Corrected camera path. Note that the paths are now in the same plane.

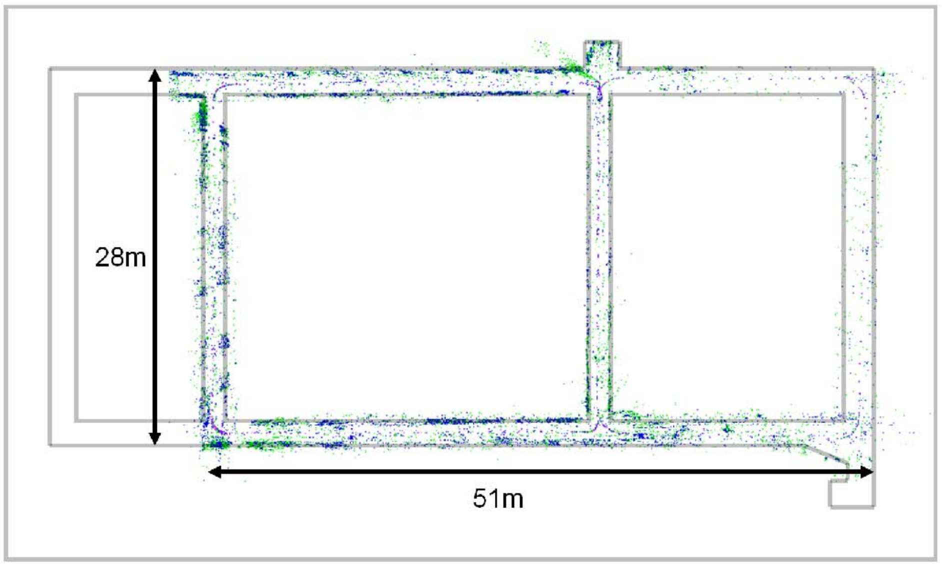

Fig. 9. Results of large building sequence. The camera made three rounds of the left loop and one of the right.

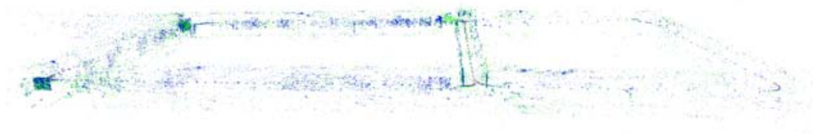

Fig. 10. The final hallway sequence model viewed from the side. Note that the model is planar matching the planar structure of the building.