Dense Reconstruction On–the–Fly

Andreas Wendel, Michael Maurer, Gottfried Graber, Thomas Pock, Horst Bischof
Institute for Computer Graphics and Vision
Graz University of Technology, Austria
{wendel, maurer, graber, pock, bischof}@icg.tugraz.at

Abstract

We present a novel system that is capable of generating live dense volumetric reconstructions based on input from a micro aerial vehicle. The distributed reconstruction pipeline is based on state-of-the-art approaches to visual SLAM and variational depth map fusion, and is designed to exploit the individual capabilities of the system components. Results are visualized in real-time on a tablet interface, which gives the user the opportunity to interact. We demonstrate the performance of our approach by capturing several indoor and outdoor scenes on-the-fly and by evaluating our results with respect to a ground-truth model.

1. Introduction

A variety of mobile, integrated image acquisition tools such as micro aerial vehicles (MAVs) or tablet computers have recently become available and affordable. The possible applications for 3D reconstruction in this context are manifold; they reach from photogrammetric mapping of cultural heritage sites over industrial inspection of danger zones to thermal 3D imaging of buildings.

While powerful and accurate Structure–from–Motion pipelines [16, 15] have been available for several years, three major problems prohibited a wide-spread use of 3D reconstruction on mobile devices: First, many approaches are not suitable for devices with low computational capabilities. Second, most systems lack real–time capabilities which means that users cannot directly interact with the system. Finally, typical representations in form of sparse 3D point clouds are adequate for further processing, but do not reflect the expectations of the potential users of such reconstruction applications.

In an attempt to fulfill these requirements, we present the first system that is capable of generating live dense volumetric reconstructions based on input provided by a remote platform. Image acquisition and tracking is performed directly on a mobile device with restricted computational abilities, and only selected keyframes are sent to a powerful server for sparse and dense mapping. The resulting triangulated feature points are returned to the tracker, whereas the dense reconstructions are sent to a tablet computer for live visualization. The user can additionally monitor the tracking quality using a live preview and interact with the system on a touch–screen (see Fig. 1). In contrast to existing 3D reconstruction pipelines,

Figure 1. Our system is able to reconstruct a scene on-the-fly using a micro aerial vehicle. After automatic marker-based initialization, a live preview (left) and a live dense reconstruction (right) are streamed to a tablet for visualization. Best viewed in color.

- we propose a distributed reconstruction pipeline which exploits the individual capabilities and requirements of the system components,
- we can obtain dense reconstructions on-the-fly, so the typical acquisition time for outdoor scenes including real–time processing is below 10 minutes, and
- we provide an interface which allows the user to interact with the reconstruction process.

Our system exploits several state-of-the-art algorithms for tracking, mapping, and dense reconstruction [11, 9, 24, 23]. We demonstrate the performance of our approach using a quad–rotor MAV in indoor and outdoor scenes, and evaluate the quality of reconstructions using a publicly available paper model as ground–truth.

978-1-4673-1228-8/12/$31.00 ©2012 IEEE

2. Related Work

A system capable of live, dense reconstruction of an environment, running on a remote mobile platform, requires three major components: Simultaneous localization and mapping (SLAM) for determining the pose of the camera, a dense reconstruction approach to estimate the geometry of the scene, and a communication framework for distributing the workload to the most suitable entities in the system. The following paragraphs give an overview of selected work from these very active research areas.

Simultaneous localization and mapping addresses the problem of exploring a previously unknown environment. This requires to determine the sensor pose given an estimated map, and to optimize the map given the sensor information at the estimated pose; two tasks which highly depend on each other and thus have to be solved simultaneously. Early SLAM systems using a single camera as the main perceptual sensor were based on filtering approaches. Davison [5] fused measurements from a sequence of images by updating probability distributions over features and extrinsic camera parameters using an Extended Kalman Filter (EKF). This requires tracking and mapping to be closely linked, as the camera pose and feature locations are updated together at every single frame. In contrast, Klein and Murray [11, 9] presented a real-time visual SLAM system based on keyframes named Parallel Tracking and Mapping. PTAM employs two separate threads, one for tracking the camera pose through every single frame, and another for mapping the environment by applying bundle adjustment to a set of spatially distributed keyframes. The major drawback of filtering approaches, namely the limited number of features that can be updated in between two frames, is circumvented in PTAM by leaving more time to the parallel mapping process. Strasdat et al. [20] experimentally showed that localization using a large amount of features measured at low temporal frequency is in general superior to a small amount of features measured at every frame, thus keyframe-based methods typically outperform filtering approaches.

SLAM approaches are often not suitable for providing input to dense reconstruction, as they maintain a map of sparse feature points and do not store full frame information. Newcombe and Davison [14] presented a method which is able to generate a rough mesh based on sparse visual SLAM features. The mesh is then successively refined by depth information obtained using variational optical flow between tracked frames, which leads to physically meaningful dense reconstructions. However, the topology of possible reconstructions is limited by the initial mesh and does not allow modeling of concave scenes containing holes or extrusions. Stühmer et al. [21] presented a similar real-time system which robustly generates accurate depthmaps based on variational optical flow, but their work lacks depthmap fusion and is thus restricted to 2.5D geometries. More recently, Graber et al. [6] presented a system based on variational depthmap fusion [24, 23] which is able to overcome the aforementioned limitations and works with a set of keyframes instead of all tracked frames. They propagate visibility information in a volumetric representation and extract the surface as zero level-set of a global convex energy, which leads to smooth reconstructions with arbitrary topology. All currently available live dense reconstruction systems exploit general-purpose graphics processing units (GPGPUs) to achieve real-time performance and are therefore not directly applicable in a mobile system.

Mobile visual localization and mapping is mainly used in augmented reality and robotics, where the computational resources are often much lower than in a desktop setting. A common approach to circumvent this problem is to use model–based [18] or extensible tracking [3], where prior knowledge about the scene is exploited to aid localization. While this has been shown to help in indoor and outdoor environments, we focus on an approach where no prior knowledge is necessary. Klein and Murray extended the original PTAM approach to work with mobile phone cameras [10]. To compensate for the lack of processing power they had to introduce several simplifications: First, they require corners to be detected over multiple scales; this leads to less but more stable features. Second, measurements are thinned rather than densified as in original PTAM to limit the map size. As a result, the pose estimate is far less robust and a considerable amount of texture is required in the scene for the tracking to work at all. In robotics, PTAM has recently been used for providing position estimates for visual control in indoor and outdoor settings [1]. Again, simplifications in tracking and bundle adjustment were necessary for the approach to run on a quad–rotor helicopter. Typically, such approaches do not allow to send high–quality imagery to the ground for real-time visualization or dense reconstruction because this would require all available processing power. In contrast, Reitinger et al. [17] presented an interactive 3D reconstruction system which uses a server–based reconstruction pipeline for urban modeling. A human scout selects suitable views and transmits the image and GPS positioning data to the server, which returns a dense point cloud after some seconds. Lee et al. [12] showed a similar concept on a mobile phone, featuring server–based pose estimation and Shape–from–Silhouette reconstruction. In comparison, our system performs pose estimation directly on the device but outsources sparse and dense reconstruction to the server. This allows on the one hand to process considerably more input images and to continuously obtain pose estimates, on the other hand the data transfer is restricted to high–quality keyframes.

Several other state–of–the–art approaches exists, but cannot be applied to our problem. First, stereo imaging has

been successfully used in real–time, outdoor SLAM systems [13]. However, on the one hand the baseline available to most mobile devices is too small for effective outdoor usage, and on the other hand the required processing power is not available. Active sensors could be used as an alternative; real-time 3D reconstruction using a Microsoft Kinect system has already been demonstrated [8]. Unluckily, the structured light pattern projected by Kinect is only suitable for indoor usage. Second, Pollefeys et al. [16] presented an impressive reconstruction pipeline which relies on huge amounts of imagery, additional GPS/INS data, and a lot of processing power. In contrast, we would like to reconstruct scenes with as little data as possible. Finally, Newcombe et al. [15] recently introduced a system which is capable of dense tracking and mapping, meaning that their system does not rely on sparse feature tracking anymore. Their approach is highly sophisticated and delivers accurate depth maps based on all captured frames; however, this is exactly the reason why it cannot be used in our system as it would require to transfer every single frame from client to server.

3. Live Reconstruction System

We present a distributed system that is capable of generating dense volumetric reconstructions based on input from a mobile platform. In contrast to other approaches using offline reconstruction, the user gets a live preview of the result on a tablet computer and can thus interact with the process. Interaction can circumvent the deficiencies of todays tracking and mapping algorithms by giving the user the chance to supervise and (if necessary) correct the result.

3.1. System Overview

Matching our goal of aerial outdoor reconstruction we choose a micro aerial vehicle with an on–board computer for image acquisition; however, our approach is as well perfectly suited for current tablet computers. The mobile device is used to capture live video and to track the pose of the camera in the environment. Having both capturing and tracking in one place allows to make use of the full frame rate delivered by the camera, and provides localization without delay and the need for persistent connectivity. This is for instance important when using a visual servoing approach in robotics [1].

We further employ a server equipped with high–performance hardware, including a state–of–the–art graphics processing unit (GPU), for sparse and dense mapping. The work of Klein and Murray [11] has shown that mapping can be done using keyframes only, which is the principle we rely on in this work. Keyframes are selected by the mobile device based on the coverage of the current map and transmitted in full quality to the server. Given the additional pose estimate by the tracker, they are directly integrated into the map and bundle adjustment is performed. Within seconds, the updated map is sent back to the mobile device and tracking continues on the updated map.

Additionally, we run a second thread on the server for dense reconstruction. Starting at the minimum of three keyframes, we estimate a depth map for every new keyframe using multi–view plane sweep [4]. The depth maps are then refined with variational denoising and integrated into a volumetric representation using variational depth map fusion [24, 23]. The resulting smooth three-dimensional surface is rendered on the GPU and transmitted to the user if new data is added to the volume or if the user triggers viewpoint changes.

Finally, the live tracking preview and the dense reconstruction are visualized on the mobile device; for aerial acquisition, this is a tablet computer on the ground. The user can on the one hand control image acquisition and on the other hand rotate and zoom the live dense reconstruction. An example of the user interface can be seen in Fig. 1 and in the accompanying video.

We want to stress that our system as described above and depicted in Fig. 2 is very flexible in terms of hardware and algorithms. The coupling between individual parts is very loose and based on the robotic operating system (ROS)[1] and a wireless link. Our work as described in the following paragraphs is based on the work of Klein and Murray [11, 9] as well as Graber et al. [6], with some major changes relating to the distributed operation. However, all parts can easily be adjusted to reflect the application at hand or the future developments in tracking and mapping.

3.2. Tracking

The tracking part of our system runs on the mobile device and has two important tasks: It has to deliver pose estimates for every input frame and it has to select keyframes based on the the scene coverage of the map. Processing the data directly avoids tracking on the ground, thus pose updates are not delayed and the low transmission bandwidth required enables operation over Wifi and 3G.

For every acquired frame, our tracking system extracts FAST-10 [19] features on a four-level image pyramid. We follow a two–stage tracking procedure which first searches for the 50 largest features on coarse levels of the image pyramid based on the prior pose estimate, updates the pose accordingly, and then performs accurate tracking by searching for 1000 features in finer levels. Due to computational constraints, we use very simple 8×8-pixel patch descriptors and match them to mapped descriptors at FAST corner location within a circular search region. This is especially useful when updating the map, because features do not need to be stored separately but can be indexed directly by the triplet $(id, s, \mathbf{x}_{2d})$, where id denotes the unique keyframe index, s denotes the pyramid scale, and $\mathbf{x}_{2d}$ the patch position in

[1] http://www.ros.org

Figure 2. Overview of the distributed live dense reconstruction system, which exploits the individual capabilities and requirements of the system entities. The tracker is run on the quad–rotor, sparse and dense mapping on the server, and visualization on the tablet.

the image. Undistortion is only applied on a per–feature basis to reduce computational effort. A detailed description of the tracking and pose estimation process can be found in the original PTAM work [11].

Keyframe selection on the mobile device is important to avoid streaming too much data to the ground. To start the tracking process, an initial map consisting of two keyframes is necessary. PTAM requires the operator to select these two frames manually, which has several issues: It requires streaming the data to the ground, it requires some experience in moving the camera, and it results in maps with arbitrary scale and origin. In contrast, we employ a single artificial marker with known dimensions to initialize the map directly on the mobile device. We use ARToolkitPlus [22] to obtain an accurate, metrical localization relative to the marker. Once the marker is visible in the image, candidates for the first keyframe are stored and SURF features [2] are extracted. Candidates for the second keyframe are stored when a baseline of more than b_{init} has been robustly measured (*i.e.* over 10 frames). Once more than f_{init} SURF features can be successfully matched between any pair of candidates fulfilling the required baseline, the two keyframes are transmitted to the server-based mapper. Given the two poses obtained by ARToolkitPlus, the essential matrix is estimated. The initial map is triangulated and refined through bundle adjustment. The resulting map with metrical scaling and a defined origin in the center of the marker is finally returned to the tracker. While marker-less initialization works as well, the proposed process is more comfortable and typically takes less than 5 seconds.

A proper initialization to metrical scale also aids further keyframe selection for expanding the map. We transmit a keyframe to the server if the distance d to the nearest keyframe in the map exceeds a minimum baseline b_{map}, or if $d > 0.25 b_{map}$ and the rotation angle α relative to the nearest keyframe exceeds a minimum angle α_{map}. Additionally, keyframes are only sent if the tracking quality is good and if a certain time t_{map} has passed since the last keyframe has been added. In comparison to PTAM, α_{map} ensures that keyframes are also added when the mobile device rotates, and t_{map} compensates for the latency between requesting a map update and actually receiving it, while at the same time ensuring that the tracker does not freeze while waiting for a map update.

Every time the tracker is lost or the received map contains considerable changes, re-localization is necessary. We employ a fuzzy image re-localization approach [9] supported by GPS and compass data (if available) for outdoor usage. We store the initial GPS/compass pose of the mobile device and insert all further keyframes relative to that. The set of blurry, scale-reduced keyframes is ranked based on the distance to the current GPS/compass pose and then matched as in the original approach. As a result, our approach can re-localize in difficult outdoor scenes with repetitive features.

3.3. Mapping

The mapping part of our system runs on the server and is responsible for providing a sparse three-dimensional structure which can be used for pose estimation by the tracker. The mapper gets keyframes with an estimated pose and a unique identifier id in irregular intervals from the remote tracker. These images are transmitted uncompressed and in full–resolution; an important detail that allows the mapper to detect the same features as the tracker. New map points are added by triangulation with neighboring views and refined by local bundle adjustment as described in [11]. Global bundle adjustment is applied whenever the mapper is idle, but interrupted if a new keyframe should be added.

Once local bundle adjustment has converged, the refined map is ready to be sent back to the tracker. All keyframes available to the mapper are already stored on the tracker, so it is sufficient to transmit the vector $(id, s, \mathbf{x}_{2d}, \mathbf{x}_{3d})$ for every new point, and for every point that was moved by bundle adjustment by

$$\|\mathbf{x}_{2d} - \mathbf{x}_{2d,old}\|_2 > \epsilon_{2d} \text{ or } \|\mathbf{x}_{3d} - \mathbf{x}_{3d,old}\|_2 > \epsilon_{3d}. \quad (1)$$

A typical map update can therefore be performed very quickly and does not require high bandwidth. If large maps need to be refined by bundle adjustment, we propose to iteratively update the map starting with points in the vicinity of the current pose and proceeding with points further away. When the map update is successful, the tracker automatically resets and continues to track using the new map.

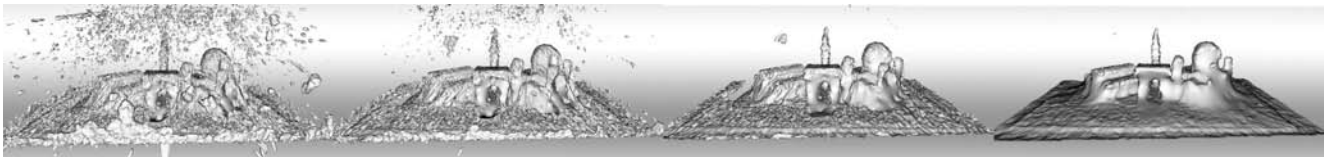


Figure 3. Variational noise removal on the dense surface helps to cope with inaccurate or incomplete depth-maps. From left to right, $\lambda = \{0.5; 0.3; 0.1; 0.01\}$ decreases, i.e., smoothing is increased. The images show a scene where considerable amounts of data are missing; however, the fusion algorithm can successfully gap the holes in the reconstruction.

3.4. Depth Map Creation

Quasi-dense depth-maps are computed based on the keyframes stored by the mapper using a GPU-accelerated multi-view version of the planesweep algorithm [4]. Keyframes might exhibit different lighting conditions, therefore normalized cross correlation is used as robust similarity measure for planesweep. Additionally, we discard depth values with a correlation value below a threshold c_{min} in order to get the most reliable depth hypotheses only. The output of planesweep is typically noisy and contains outliers as well as areas with missing data. To improve the subsequent fusion, raw depth-maps are smoothed using a total variation (TV-L1) based image denoising model denoted by

$$\min_{u_d} \left\{ \int_{\Omega_d} |\nabla u_d| dx + \int_{\Omega_d} \lambda_d |u_d - f_d| dx \right\}. \quad (2)$$

Here, u_d is the unknown denoised image, f_d is the input and $\Omega_d \subseteq \mathbb{R}^2$ is the image domain. Using total variation as regularizer has the desired property of preserving edges while smoothing the flat regions. The parameter λ_d controls the amount of smoothing.

3.5. Depth Map Fusion

We employ a volumetric representation of geometry using a truncated signed distance function. In contrast to mesh based representations, volumetric approaches allow solving for arbitrary 3D geometry, *i.e.* there is no constraint concerning the genus of the surface topology. The reconstruction algorithm is based on the method of [24, 23]. Using the signed distance formulation, the 3D surface is represented implicitly as the zero level-set of a function $u : \Omega \rightarrow [-1, 1]$, where Ω is a subset of $\mathbb{R}^3$.

Depth-maps are converted to truncated signed distance fields f and for memory reasons compressed into a histogram representation. Every voxel x has an associated histogram $h(x)$ which approximates the probability density function of the truncated signed distance function. $h(x, i)$ denotes the histogram count of bin i, *i.e.* how often the value d_i occurred in the distance fields f at voxel x.

Individual depth maps are fused together by minimizing the convex energy functional

$$\min_{u} \left\{ \int_{\Omega} |\nabla u| dx + \lambda \sum_{i=1}^{N} \int_{\Omega} h(x, i) |u(x) - d_i| dx \right\}. \quad (3)$$

The regularization term $\int_\Omega |\nabla u| dx$ measures the total variation of the function u. It minimizes the surface area of the level sets and hence effectively removes noise caused by the outliers of the depth maps while simultaneously computing a minimal surface approximation in areas with missing depth data (see Fig. 3). The data term measures the ℓ_1 distance of the solution to the individual distance fields. For minimization, we employ a globally optimal primal–dual approach as proposed by [6].

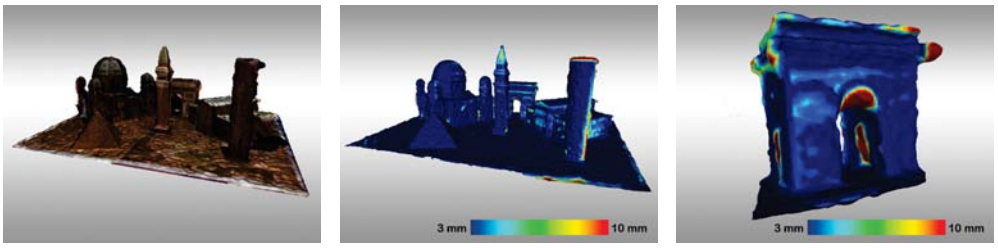
3.6. Visualization

Visualization is done through a GPU-accelerated raycaster which is capable of rendering iso-levels of u. The advantage of this approach is that no additional data structures are necessary since the raycaster operates directly on the implicit representation of geometry. Moreover, the result of raycasting is a single image which can be efficiently transmitted, *i.e.* the mobile device used for supervision of the reconstruction process does not require any special computational capabilities.

4. Experiments

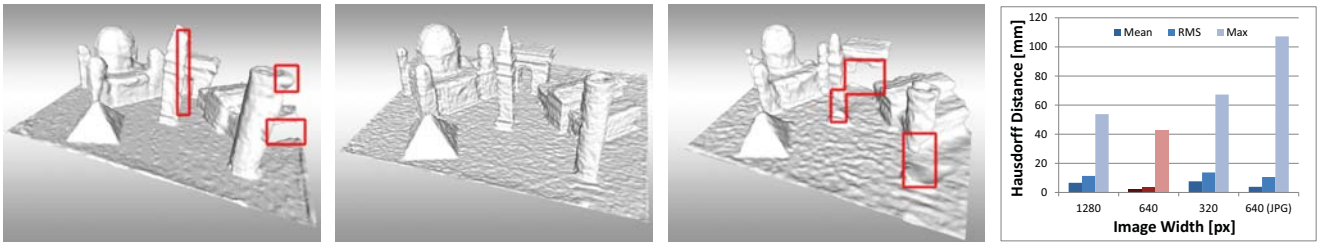
We conducted several indoor and outdoor experiments using a "Pelican" quad–rotor MAV by Ascending Technologies and a single industrial camera (IDS UI-1240SE) with a maximum resolution of 1280×1024px, global shutter, and an $8mm$ lens. The camera is connected via USB to a 1.6 GHz Intel Atom on–board computer with 1 GB RAM and a wireless 802.11g link. Our server is equipped with a 2.54 GHz Quad-Core CPU and an NVIDIA GTX480 GPU which is used for highly parallelized dense reconstruction. The user interface is implemented on an Android-based NVIDIA Tegra3 prototype tablet, which is again connected by a wireless link.

A typical live reconstruction process starts with the creation of an initial map, which happens automatically if the user has placed an artificial ARToolkitPlus marker [22] into the scene. Then, the desired volume of interest for dense



(a) Textured reconstruction (b) Difference to ground–truth (c) Maximum error

Figure 4. Reconstruction of the *City of Sights* model using an input resolution of 640×512px. The mean Hausdorff distance to ground–truth is 2.4mm, the RMS distance is 3.9mm, and the maximum distance underneath the side gates of the Arc de Triomphe is 42.8mm.



(a) 1280×1024px (b) 640×512px (c) 320×256px (d)

Figure 5. Qualitative and quantitative evaluation of the input resolution. (a) to (c) show the reconstructed geometry for different input resolutions with erroneous regions marked in red. The diagram (d) shows mean, RMS, and maximum Hausdorff distance compared to ground–truth. The best result is obtained using an uncompressed keyframe resolution of 640×512px.

reconstruction, relative to the center and orientation of the marker, has to be defined by the user. For outdoor experiments, this process can be done on the ground or while the aerial vehicle is airborne. Once the system is initialized, reconstruction is fully automated and starts as soon as a minimum of three keyframes is available.

In the following paragraphs we experimentally evaluate the effect of different keyframe resolutions on the reconstruction. All other parameters can be interactively tuned to obtain good reconstructions in different environments. However, for the evaluation we used the following set of parameters: for initialization $b_{init} = 0.3m$ and $f_{init} = 20$, for keyframe selection $b_{map} = 0.1m$ (indoors) and $b_{map} = 0.4m$ (outdoors), $\alpha_{map} = 20°$, and $t_{map} = 3$s, for map updates $\epsilon_{2d} = \sqrt{2}$px, $\epsilon_{3d} = 0.001m$ (indoors), and $\epsilon_{3d} = 0.01m$ (outdoors), for dense mapping $N = 4$, $\lambda = 0.4$, $\lambda_d = 0.5$, and $c_{min} = 0.65$.

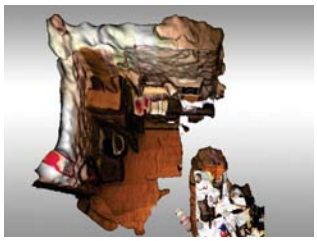
4.1. Model Evaluation

Typical evaluation datasets providing dense ground–truth are not designed for SLAM, and thus tracking typically fails. We recorded our own evaluation data based on standardized geometry and texture using the *City of Sights* paper model [7], which allows us to demonstrate the quality of our reconstructions by comparing to ground–truth. Digital blueprints and a virtual model of the scene are provided online[2]. As proposed in the original paper, we use the Iterative Closest Points method [25] to align the reconstructed volume to the ground–truth mesh. We finally measure the Hausdorff distance from reconstructed to ground–truth points and visualize the residual in pseudo colors. Gruber et al. [7] state that typical paper models have a mean deviation of 3 mm compared to ground–truth, thus we denote our reconstruction as correct within this limit. Errors of more than 10 mm occur only in regions which are either seen by a small number of views or which are hard to triangulate due to missing texture. In the *City of Sights* model, this applies to the top of the Irish Round Tower and the inner part of the Arc de Triomphe. A standard result for an input resolution of 640×512px, as well as a pseudo–colored distance image are shown in Fig. 4.

Our system can capture and process the full image resolution of 1280×1024px with a frame rate of $5fps$ on–board the quadrotor helicopter. Frame rates increase to $10fps$ when lowering the image resolution to 640×512px, and $20fps$ for 320×256px. We compare the different resolutions and frame rates by recording uncompressed video with full resolution and $20fps$ on a desktop computer. The data is then ported to a laptop, resampled to the correct resolution and frame rate, and used as input to the tracker which sends keyframes over WiFi to the server. The rather complicated setup perfectly simulates our system; however, re-

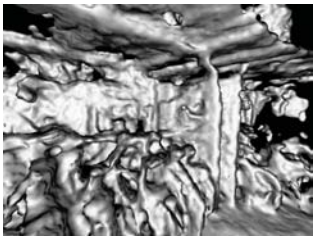
[2] http://cityofsights.icg.tugraz.at

(a) Scene photo



(b) Textured reconstruction

Figure 6. Office scene (indoor) with a reconstruction rotated to an overhead view. Note that even weakly textured regions are reconstructed. Volume size $320 \times 320 \times 320$ voxels.



(a) Untextured reconstruction

(b) Textured reconstruction

Figure 7. Dense reconstruction of complex geometry. Thanks to our volumetric reconstruction approach, we obtain smooth surfaces and do not depend on topological constraints.

peated reconstructions might still differ due to network and CPU workloads. Fig. 5 shows a qualitative and quantitative comparison of reconstructions with different keyframe resolutions based on the Hausdorff distance. While the full resolution provides the smoothest result as expected, tracking often fails due to the low frame rate and some holes remain. Medium resolution performs best, whereas the low resolution of 320×256px cannot connect all extrusions in the model.

Directly streaming the imagery to the server has also been evaluated, and is possible with the same frame rate for compressed medium resolution (80% JPG) and uncompressed low resolution images. While low image resolution has already been shown to be undesirable, compression also affects the reconstruction quality (see Fig. 5(d)). Additionally, the benefit of having latency-free pose updates on the MAV is omitted.

Finally, when comparing to the non–distributed system we did not see significant deviations in reconstruction quality. A difference is only apparent if the tracker has to quickly update the map several times in a row, which results in a short-term loss of the map because of the network latency. However, this is in general not an issue when operating outdoors because the distance to the scene is larger and less frequent map updates are required.

4.2. Real–World Evaluation

Our system has also proven to deliver very good results in real–world scenes. We have successfully reconstructed indoor office scenes by moving the micro aerial vehicle (Fig. 6) and outdoor scenes by flying it (Fig. 8). The typical acquisition time is below 10 minutes for all the scenes shown, and dense reconstruction is computed in real-time, with visualization on the tablet. Our reconstruction approach is not tied to a specific topology but can visualize very complex scenes as shown in Fig. 7. We can also handle different input such as thermal imaging data, where colored texture is ultimately required, and reconstruct the dense 3D scene on–the–fly. More details can be found online[3] in the accompanying video.

5. Conclusion

We have presented the first system that is capable of generating live dense volumetric reconstructions based on input provided by a mobile platform. We showed how existing state-of-the-art approaches to tracking, mapping, and dense reconstruction can be adapted to work in a distributed environment with modern technologies such as micro aerial vehicles for image acquisition and tablet computers for visualization. Our work focuses on exploiting the individual capabilities of the devices, which leads to an interactive system that is able to tolerate mistakes of the individual parts.

In future work we will further extend the capabilities of our system in outdoor reconstruction. Our current results could be further improved by adapting the tracking approach. Currently, re–localization needs to be run on many frames to cope with repetitive patches. Additionally, we propose to research multiple volumes which are indexed by GPS, compass, and depth data to cope with the large spaces in outdoor scenes.

Acknowledgments. This work has been supported by the Austrian Research Promotion Agency (FFG) project FIT-IT Pegasus (825841/10397) and the Austrian Science Fund (FWF) under grant P22492-N23. We thank NVIDIA for providing a Tegra3 prototype tablet and Ascending Technologies for providing thermal imagery.

References

[1] M. Achtelik, M. Achtelik, S. Weiss, and R. Siegwart. Onboard IMU and monocular vision based control for MAVs in unknown in- and outdoor environments. In *Proc. ICRA*, 2011. 2, 3

[2] H. Bay, T. Tuytelaars, and L. Van Gool. SURF: Speeded up robust features. In *Proc. ECCV*, 2006. 4

[3] http://aerial.icg.tugraz.at

(a) (b) (c) (d)

Figure 8. Airborne outdoor results. In (a) to (c), various outdoor scenes are depicted where the top row shows an image of the scene and the bottom row depicts the reconstruction using a volume of $320 \times 320 \times 320$ voxels. (d) shows the reconstruction of a car park using thermal images as input.

[3] G. Bleser, H. Wuest, and D. Stricker. Online camera pose estimation in partially known and dynamic scenes. In *Proc. ISMAR*, 2006. 2

[4] R. Collins. A space-sweep approach to true multi-image matching. In *Proc. CVPR*, 1996. 3, 5

[5] A. J. Davison. Real-time simultaneous localisation and mapping with a single camera. In *Proc. ICCV*, 2003. 2

[6] G. Graber, T. Pock, and H. Bischof. Online 3d reconstruction using convex optimization. In *Proc. ICCV Workshops*, 2011. 2, 3, 5

[7] L. Gruber, S. Gauglitz, J. Ventura, S. Zollmann, M. Huber, M. Schlegel, G. Klinker, D. Schmalstieg, and T. Höllerer. The city of sights: Design, construction, and measurement of an augmented reality stage set. In *Proc. ISMAR*, 2010. 6

[8] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, and A. Fitzgibbon. Kinectfusion: Real-time 3d reconstruction and interaction using a moving depth camera. In *ACM Symposium on User Interface Software and Technology (UIST)*, 2011. 3

[9] G. Klein and D. Murray. Improving the agility of keyframe-based SLAM. In *Proc. ECCV*, 2008. 1, 2, 3, 4

[10] G. Klein and D. Murray. Parallel tracking and mapping on a camera phone. In *Proc. ISMAR*, 2009. 2

[11] G. Klein and D. W. Murray. Parallel tracking and mapping for small AR workspaces. In *Proc. ISMAR*, 2007. 1, 2, 3, 4

[12] W. Lee, K. Kim, and W. Woo. Mobile phone-based 3d modeling framework for instant interaction. In *Proc. ICCV Workshops*, 2009. 2

[13] C. Mei, G. Sibley, M. Cummins, P. Newman, and I. Reid. A constant–time efficient stereo slam system. In *Proc. BMVC*, 2009. 3

[14] R. A. Newcombe and A. J. Davison. Live dense reconstruction with a single moving camera. In *Proc. CVPR*, 2010. 2

[15] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison. DTAM: Dense tracking and mapping in real-time. In *Proc. ICCV*, 2011. 1, 3

[16] M. Pollefeys, D. Nister, J. M. Frahm, A. Akbarzadeh, P. Mordohai, B. Clipp, C. Engels, D. Gallup, S. J. Kim, P. Merrell, C. Salmi, S. Sinha, B. Talton, L. Wang, Q. Yang, H. Stewenius, R. Yang, G. Welch, and H. Towles. Detailed real-time urban 3d reconstruction from video. *IJCV*, 78(2-3), 2008. 1, 3

[17] B. Reitinger, C. Zach, and D. Schmalstieg. Augmented reality scouting for interactive 3d reconstruction. In *IEEE Virtual Reality Conference*, 2007. 2

[18] G. Reitmayr and T. Drummond. Going out: robust model-based tracking for outdoor augmented reality. In *Proc. ISMAR*, 2006. 2

[19] E. Rosten and T. Drummond. Machine learning for high–speed corner detection. In *Proc. ECCV*, 2006. 3

[20] H. Strasdat, J. M. M. Montiel, and A. J. Davison. Real-time monocular SLAM: Why filter? In *Proc. ICRA*, 2010. 2

[21] J. Stühmer, S. Gumhold, and D. Cremers. Real-time dense geometry from a handheld camera. In *Proc. DAGM, LNCS 6376*, 2010. 2

[22] D. Wagner and D. Schmalstieg. ARToolKitPlus for pose tracking on mobile devices. In *Proceedings of Computer Vision Winter Workshop (CVWW)*, 2007. 4, 5

[23] C. Zach. Fast and high quality fusion of depth maps. In *Proc. International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT)*, 2008. 1, 2, 3, 5

[24] C. Zach, T. Pock, and H. Bischof. A globally optimal algorithm for robust TV-L1 range image integration. In *Proc. ICCV*, 2007. 1, 2, 3, 5

[25] Z. Zhang. Iterative point matching for registration of free-form curves and surfaces. *IJCV*, 13(2), 1994. 6