Interactive Dense 3D Modeling of Indoor Environments

Hao Du[1] Peter Henry[1] Xiaofeng Ren[2] Dieter Fox[1,2] Dan B Goldman[3] Steven M. Seitz[1]
{duhao,peter,fox,seitz}@cs.washinton.edu xiaofeng.ren@intel.com dgoldman@adobe.com
[1]University of Washington [2]Intel Labs Seattle [3]Adobe Systems

Abstract

The arrival of cheap consumer depth cameras, led by Microsoft's Kinect system, presents a huge opportunity for 3D modeling of personal spaces. While 3D indoor mapping techniques are becoming increasingly robust, they are still too brittle to enable non-technical users to build consistent and complete maps of indoor environments. This is due to technical challenges such as limited lighting, occlusion, and lack of texture, and to the fact that novice users lack a deep understanding of the underlying algorithms and their limitations. In this research, we use a prototype affordable RGB-D camera, which provides both color and depth, to build a real-time interactive system that assists and guides a user through the modeling process. Color and depth are jointly utilized to achieve robust 3D alignment. The system offers online feedback and guidance, tolerates user errors and alignment failures, and enables novice users to capture complete and dense 3D models. We evaluate our system and algorithms with extensive experiments.

1. Introduction

Building 3D models of indoor environments has great potentials and interesting usages. For example, having access to an accurate, photorealistic model of one's home can enable many scenarios such as virtual remodeling or online furniture shopping. Such a model can also provide rich context information for smart home applications.

Indoor 3D modeling is also a hard problem for many reasons such as limited lighting, occlusion, limited field of view, and lack of texture. There has been a lot of work and progress on 3D modeling and reconstruction of environments. State-of-the-art research systems can build 3D models at a city scale [22, 18, 1]. On the other hand, building a complete model of a room, say a small room with textureless walls, remains a challenge.

Many recent works addressed the robustness and completeness issues in indoor modeling and searched for a solution to solve or bypass them. Sinha et al [21] built an interactive system to enable a user to mark planar surfaces.

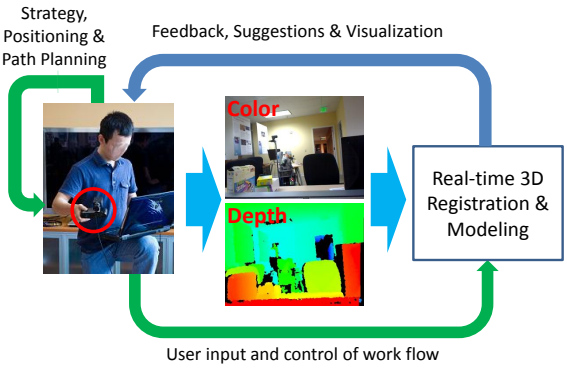

Figure 1. Interactive 3D mapping: The depth and color frames collected by the user are aligned and globally registered in real time. The system alerts the user if the current data cannot be aligned, and provides guidance on where more data needs to be collected. The user can track the model quality and "rewind" data or introduce additional constraints to improve the global consistency of the model.

Furukawa et al [7] used the Manhattan World assumption to automatically find such planes. In both cases, photos have to be carefully taken and registered, and geometric details are sacrificed for the sake of large surface textures and appealing visualization.

Our objective is to enable a non-technical user to build dense and complete models for his/her personal environments. One technology that makes it feasible is the wide availability of consumer depth cameras, such as those deployed in the Microsoft Kinect system [19]. These cameras directly provide dense color and depth information. However, their field of view is limited (about 60°) and the data is rather noisy and low resolution (640×480). Henry et al [11] showed that such cameras are suitable for dense 3D modeling, but much was left to be desired, such as robustness for use by non-experts, or complete coverage of the environment including featureless or low-light areas.

The key idea behind our work is to take advantage of online user interaction and guidance in order to solve many of the issues in 3D environment modeling. We design and implement an interactive 3D modeling system so that the

user holds a depth camera to freely scan an environment and enjoys real-time feedback. Our approach has several advantages:

Robust: We compute 3D alignments of depth frames on-the-fly, so that the system can detect failures (many reasons such as fast motions or featureless areas) and prompt the user to "rewind" and resume scanning. The success of 3D registration of consecutive frames is thus "guaranteed".

Complete: A 3D environment model is constructed on-the-fly. The user can check the model in 3D at any time for coverage and quality. The system also automatically provides suggestions where the map may yet be incomplete.

Dense: Largely due to the nature of the depth sensor, the model constructed by our system is dense without assuming planar surfaces or a "box" model of a room. A dense model reveals details of the environment and can have many uses such as recognizing architectural elements, robot motion planning, telepresence or visualization.

In addition to developing an interactive mapping system, we introduce a variant of RANSAC for frame-to-frame matching that combines the strengths of color and depth cues provided by our camera. In contrast to the standard inlier count to rank matches, our approach learns a classifier that takes additional features such as visibility consistency into account. The learned classifier results in more robust frame-to-frame alignments and provides an improved criterion for detection alignment failures, which is important for our real time user feedback.

This paper is organized as follows. After discussing related work, Section 3 gives an overview of our mapping system. The frame alignment approach is introduced in Section 4, followed by a description of the interactive mapping technique. Section 6 provides experimental results. We conclude in Section 7.

2. Related Works

Modeling and reconstructing the world in 3D is a problem of central importance in computer vision. Various techniques have been developed for the alignment of multiple views, such as pairwise matching of sparse [12] or dense point clouds [2, 3], two-view and multi-view geometries [10] and joint optimization of camera poses and 3D features through bundle adjustment [24].

3D vision techniques, combined with local feature extraction [15], have led to exciting results in 3D modeling. PhotoTourism [22] is an example where sparse 3D models are constructed from web photos. There has been a lot of work on multi-view stereo techniques [20]. The patch-based framework [9], which has been most successful on object modeling, has also been applied to environment modeling. The work of Furukawa et al [8] built on these works to obtain dense indoor models using the Manhattan world assumption.

There have been many successful efforts to build real-time systems for 3D structure recovery. Davison *et.al.* built real-time SLAM (simultaneous localization and mapping) systems using monocular cameras [5]. The Parallel Tracking and Modeling system (PTAM) [13] is a closely related system applying SLAM techniques. Another example of real-time sparse 3D modeling can be found in [16]. One recent development is the dense 3D modeling work of [17] which uses PTAM and flow techniques to compute dense depths. Many real-time systems are limited in the scale they can handle.

Due to the difficulties of indoor modeling, such as lighting and lack of texture, interactive approaches have been proposed to utilize human input. [6] was an early example showing very impressive facade models and visualizations with manual labeling. [23] used interactions to extract planes from a single image. [21] is a recent example combining user input with vanishing line analysis and multi-view stereo to recover polygonal structures. Our work is different and novel, as we enable online user interaction, utilizing user input on-the-fly for both capturing data and extracting geometric primitives.

Recently, there have been many efforts to push the limits of 3D modeling to a large scale. One example is the city-scale, or "Rome"-scale, sparse 3D reconstruction [1]. Another example is the real-time urban street reconstruction work of Pollefeys et al [18]. In comparison, indoor modeling has not taken off beyond a few small-scale results.

This may soon change with the arrival of mass-produced depth cameras. We believe there are great opportunities to make use of these cameras for 3D modeling. The work of Henry et al [11] is most relevant to this work. They showed how to use both color and depth for sequential alignment of depth frames and carried out experimental studies of various alignment algorithms and their combinations. Our work aims at making such a depth-camera-based modeling system online, incorporating various aspects of user interaction to make 3D modeling robust, easy to use, and capable of producing dense, complete models of personal spaces.

3. System Overview

Figure 2 gives an overview of our interactive mapping system. The system is based on the well established structure of online mapping approaches, where each data frame is matched against the most recent frame to provide visual odometry information, and against a subset of previous frames to detect "loop closures" [14, 11, 4]. While visual odometry results in local consistency, loop closures provide constraints used to globally optimize all camera poses.

The globally aligned map is visualized in real time, as shown in Figure 3. The user can assess the quality of frame alignment via a bar shown in the visualizer. In order to avoid capturing data that can not be aligned consecutively, the sys-

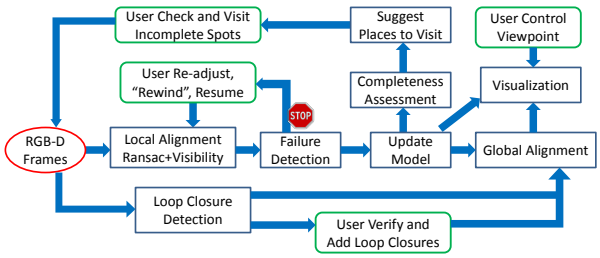

Figure 2. Detailed system overview: Frame alignment, loop closure detection, and global alignment are performed in real time. Green boxes represent user interactions. The user is alerted if alignment fails, notified of suggested place visits, and can verify and improve model quality via manual loop closure insertion.

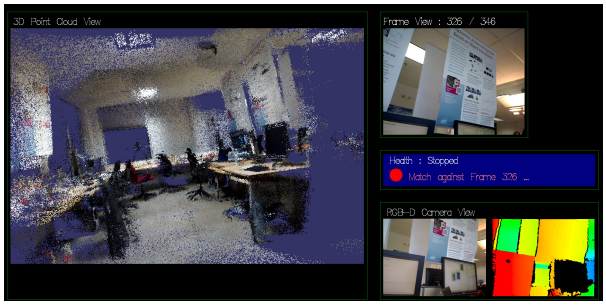

Figure 3. Real time visualization of the mapping process: The left panel provides a viewer of the globally aligned 3D map. The health bar in the center right panel indicates the current quality of frame alignment. In a failure case, as is shown, the user is guided to re-locate the camera with respect to a specific frame contained in the map. The upper right panel shows the target frame, and lower right panel indicates the current camera view.

tem alerts the user whenever local alignment fails. In this case, the user has to re-locate the camera with respect to the global model. To do so, the live camera data is matched against a frame that is globally aligned within the map. Per default, this frame is the most recently matched frame, but it can also be any map frame chosen by the user. Once the camera is re-localized, the mapping process proceeds as before. All data collected between failure detection and re-localization is discarded, and the constraint graph used for global alignment is updated appropriately.

To enable building complete maps, the system continuously checks the current model for completeness. This analysis provides visual feedback about incomplete areas and guides the user to locations that provide the necessary view points. In addition to automatic loop closure detection, which cannot be expected to work in every case, the user can check the model for inconsistencies and add loop closure constraints between pairs of frames chosen from the map.

4. Color and Depth RANSAC

In this section we describe our real-time matching algorithm for visual odometry. The added depth information associated with image pixels enables us to use *3-Point* matching, yielding more robust estimation of the relative camera pose between frame pairs. The proposed *Visibility Criteria* evaluates the quality of a relative camera pose transform. Using a combination of *Visibility Criteria* and RANSAC significantly improves matching accuracy.

4.1. RANSAC and 3-Point Matching Algorithm

Initial feature matches are established with feature descriptors, such as SIFT or Calondar, applied to interest points in 2D images. These matched image feature points are associated with their 3D locations. Instead of traditional methods that use the *7-Point* or *8-Point* algorithm to estimate fundamental matrices (or essential matrices when camera intrinsics are known) [10]), we directly estimate the full camera pose transform using *3-Point* algorithm using 3D locations. The full camera pose transform, as compared with the essential matrix, provides the added information of translational scale. As there are outliers in the initial feature matches, RANSAC is applied to determine the feature match inliers, and thereby the underlying camera pose transform.

Consider N pairs of initial feature matches between Frame F_1 and F_2, represented by 3D coordinates (X_1^i, X_2^i), $(i = 1, 2, ..., N)$ in their respective reference systems. The problem is to find a relative transform (R, T) (rotation and translation) that best complies with the initial matches while being robust to outliers. A typical RANSAC approach samples the solution space to get a candidate (R, T), estimating its fitness by counting the number of inliers, f_0,

$$f_0(F_1, F_2, R, T) = \sum_i^N L(X_1^i, X_2^i, R, T), \qquad (1)$$

where,

$$L(X_1^i, X_2^i, R, T) = \begin{cases} 1, & e = \|RX_1^i + T - X_2^i\| < \epsilon \\ 0, & otherwise \end{cases} \qquad (2)$$

and ϵ is the threshold beneath which a feature match (X_1^i, X_2^i) is determined to be inlier with respect to the particular (R, T). RANSAC chooses the transform consistent with the largest number of inlier matches.

4.2. Visibility Criteria

Owing to the depth maps captured by the RGB-D camera, the quality of a camera pose transform can be indicated by laying out the point clouds in 3D space and performing a visibility check, termed *Visibility Criteria*. The Visibility

Criteria can help obtain more accurate relative camera poses (Sec. 4.3). It also provides cues for the suggestion of loop closure candidates (Sec. 5.3).

Consider the 2D example shown in Figure 4 (left), the scene is a horizontal line shown in black, and is captured by a pair of cameras. The circles and stars are the depth maps sampled at the camera pixels. When (R, T) is the genuine relative transformation, there is no visibility conflict. When (R^*, T^*) is a wrong relative transformation, shown in Figure 4 (right), overlaying the point clouds from both cameras, it is possible to see visibility conflicts – when a camera captures a point in 3D, the space along its viewing line should be completely empty; if there exists points from the other camera in between, there is a conflict.

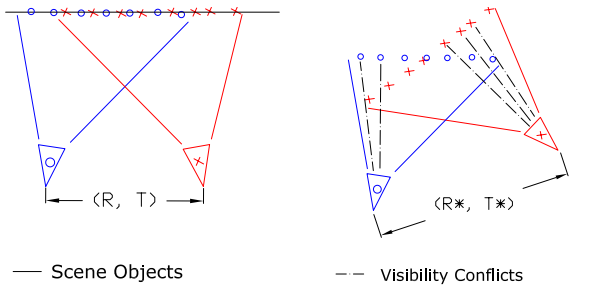

Figure 4. Visibility conflicts.

Practically, we count the visibility conflicts by projecting the point cloud C_1 from frame F_1 onto the image plane of F_2, and check its depth in F_2's camera coordinate system. If any such depth appears smaller than the depth of the F_2's pixel at the corresponding location (ignoring the errors by setting a tolerance error equal to the depth accuracy), it is counted as a visibility conflict. The same approach is reversely applied by projecting C_2 onto the image plane of F_1. Several criteria can be derived from the visibility check. We utilize the following ones: number of visibility conflicts (f_1); average squared distance of points with visibility conflicts (f_2); number of visibility inliers (f_3) by counting those pixels where no visibility conflicts both ways of projections.

A good candidate transform R, T ideally gives $f_1 = 0$ and $f_2 = 0$. f_3 indicates the size of the overlapped area measured by pixels between a pair of frames.

4.3. Decision Function for RANSAC

We now not only have the number of RANSAC inliers, f_0, but multiple features $f_i, (i = 1, 2, 3, ..., m)$ for the RANSAC algorithm to pick up the final transform. Given frame pair F_1, F_2 and candidate transform R, T, a decision function is needed to figure out how likely the candidate R, T would be a good solution. The general form of the decision function is, $g(f_1, f_2, ..., f_m)$. We define g to be the linear function of f_i, i.e.

$$g(F_1, F_2, R, T) = \sum_{i=0}^{m} \alpha_i, \tag{3}$$

and estimation the weights α_i through linear regression.

We demonstrate the effectiveness of incorporating the added visibility criteria in Section 6.1.

5. User Interaction

Our system incorporates user interaction in three ways: failure detection and rewind/resume in matching, completeness guidance, and user-assisted loop closure.

5.1. Rewind and Resume

In the case of interior scene reconstruction, each captured frame usually covers a small area of the entire scene. Thus, the connections (relative camera poses) between neighboring frames are critical for a successful reconstruction. Our system, through online feedback to the user, guarantees that only frames that can be successfully aligned to at least one of the previous frames are added to the map. If a newly captured frame cannot be aligned (for example, because the user moved the camera too quickly or moved to an area with insufficient features) the system stops recording frames until it receives a frame that can be successfully aligned.

Using what we call *Rewind and Resume*, users can capture new frames by selecting any existing frame as the target frame to align against. The related *Undo* operation is straightforward but extremely useful. If the system accepts frames that the user does not want (e.g., they may be blurry or not very well aligned), the user can manually 'undo' to the nearest desired frame, and continue from there, recapturing the scene from a different viewing angle. This also gives the user the ability to remove frames that inadvertently contain moving objects, such as people walking in front of the camera.

5.2. Completeness

Capturing a complete 3D model of the scene is desired, because large missing areas in an incomplete 3D model significantly lower the visual quality. A missing area exists in the scene either because the area has never been captured or the frames that did contain the area did not get depth values, for reasons such as range or relative angle.

We consider the completeness in a user-defined manner. Using a passive capturing system, it can be very difficult for the user to be aware of which parts of the scene have been captured. With an online system, the user can view the current reconstruction in real time, view the up-to-date 3D model, and directly see which areas are missing.

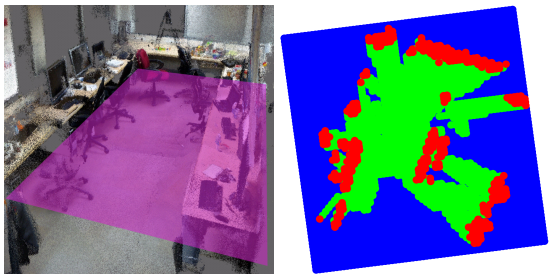

Figure 5. Completeness Guide. Our system displays the classification of voxels from a user specified 2D slice in the 3D point cloud. Green: the voxel is guaranteed to be empty; Red: it is "occupied"; Blue: unknown area.

In order to further assist users in finding uncaptured areas, our system is able to estimate completeness. Consider a bounding box that contains the currently reconstructed point cloud. The inside of the bounding box can be represented by 3D grid voxels. Each grid voxel is classified into one of the three categories: (1) there is at least a scene point in that voxel; (2) there must be no scene point in the voxel; (3) none of the above. All voxels are initialized in Category (3). A voxel is determined as Category (1) when there exists a scene point. A voxel is determined as Category (2) when it is not (1) and the voxel has been seen through by any of the existing camera viewing line.

Figure 5 shows the classification of voxels from a user specified 2D slice in the 3D point cloud. Green: the voxel is guaranteed to be empty; Red: it is "occupied"; Blue: unknown area. The user's goal is then to paint all areas in either green or red by exploring the 3D space.

5.3. Interactive Loop Closure

When frames are aligned with reliable relative poses (as is the case with our system), a small number of loop closure constraints can achieve global consistency. Our system can obtain loop closure constraints through both RANSAC matching and using ICP. While RANSAC matching can be done automatically, this is computationally expensive and only works if the inconsistent areas of the map have matching views. Also, performing RANSAC matching against all previously recorded frames is computationally prohibitive. The Visibility Criteria are used to suggest frames that are likely inconsistent. The user can select any pair of frames to perform a RANSAC or ICP based alignment. The user then inspects the resulting map and decides to accept or reject the change.

6. Experiment Results

In this section, we present our evaluation and experiment results using our system. We show that 3-Point RANSAC based on feature points with 3D locations performs significantly better than 7-Point RANSAC based on feature points with 2D locations only (that is, ignoring depth data). We also demonstrate the benefits of additionally incorporating visibility criteria information via our learned RANSAC classifier. The advantage of the interactive system for visual odometry is evaluated through an experiment requesting five persons to model a meeting room.

In addition to these experiments, we also tried to run bundler [22] and PTAM [13] on parts of our data. PTAM is designed for small environments only and did not scale to the large scale maps we build with our system. Bundler was not able to generate consistent maps of any of our test environments. To ensure that these failures were not only due to the low quality of the images collected by our depth camera, we additionally collected data in one of our test environments with a high resolution video camera. Bundler failed even on this high quality data set, generating inconsistent matches for the majority of data.

6.1. RANSAC for Visual Odometry

We compare the performance of 3-Point RANSAC for full transform estimation on the captured RGB-D data and 7-Point RANSAC for essential matrix estimation on the RGB portion of the same data. The image resolution provided by our camera is 640×480. The effective depth-range we take is from $0.5m$ to $5m$. The depth-accuracy is calculated from the camera specifications ($7.5cm$ baseline, 570 pixel focal length and 640×480 resolution), approximately $0.03cm$ at $0.3m$ range and $7cm$ at $5m$ range. For the initial feature match, we use Calondar features. We generate feature pairs via bidirectional matching, where (X_1^i, X_2^i) is considered a matched feature pair if and only if X_1^i is the best match in Frame 1 for Feature X_2^i and vice versa. To enable good performance of the 7-Point RANSAC based on the features' 2D pixel locations, we include the best feature match if and only if the second best match has a significantly higher distance in feature descriptor space (1.25 times further away than the best distance).

2D versus 3D RANSAC

To generate ground truth data in a realistic scenario, we collected a data sequence in a lab and used our system to generate the globally optimized map shown in the bottom right panel of Fig. 11. The consistency of that map indicates that the camera poses for the individual frames are sufficiently accurate to serve as ground truth for our experiment.

We randomly sampled frame pairs from this data set and determined their transformation using 7 point RANSAC on 2D pixel features and 3 point RANSAC on 3D feature points. Out of all pairs, 7 point RANSAC found a transformation for 15,149 pairs ($\geq$ 7 inliers), with an average reprojection error of 0.69m. 3D RANSAC determined transformations for 15,693 pairs ($\geq$ 3 inliers) with an error of

0.31m. Figure 6 provides a more detailed view of this result. It shows the reprojection error versus the number of inliers found between two consecutive frames. As can be seen, using depth information within RANSAC (3D RANSAC) significantly reduces reprojection errors, generating good results even for small numbers of inliers.

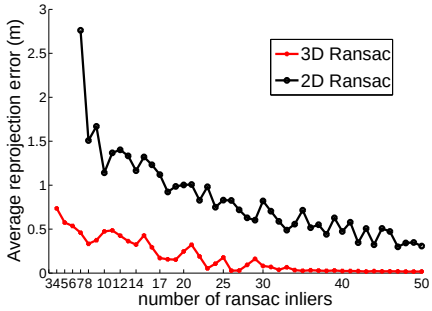

Figure 6. Alignment error of 7 point 2D RANSAC and 3 point 3D RANSAC versus number of inliers.

RANSAC with Visibility Features

We also compare the performance of 3-Point RANSAC with and without incorporating the visibility criteria features for the RANSAC objective. To collect the data needed to train the linear regression model we placed the depth camera at 12 different locations, measured their ground truth distances, and collected 100 depth frames at each location. We then randomly picked pairs of camera frames, ran the RANSAC algorithm, and recorded all estimated transforms along with the number of inliers and the visibility features associated with these transforms (we only used pairs of frames that had at least one transform within 1m of the ground truth distance for the entire RANSAC run). We randomly split the data into a training and evaluation set. The training set was used to estimate the linear regression model, which was then used to re-rank the RANSAC transforms in the evaluation set.

Figure 7 shows that "RANSAC + Visibility" produces more accurate camera pose translations, indicating more accurate camera pose transforms. An example result is given in Figure 8. The top row shows a pair of frames to be matched; bottom is the projection of the point cloud from the top left frame onto the camera pose of the top right frame using the estimated camera transform without (left) and with (right) using the visibility criteria. The pixels with pink color indicate visibility conflicts. As can be seen, our RANSAC version chooses a transformation with less inliers but an overall improved alignment. This experiment shows that the visibility criteria helps RANSAC find a solution that is closer to the groundtruth camera pose transform.

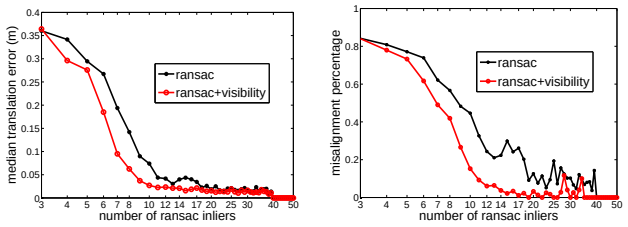

Figure 7. Accuracy of 3D RANSAC with and without visibility features. Left: alignment error vs. number of inliers. Right: percentage of misaligned frames (misalignment threshold 0.1).

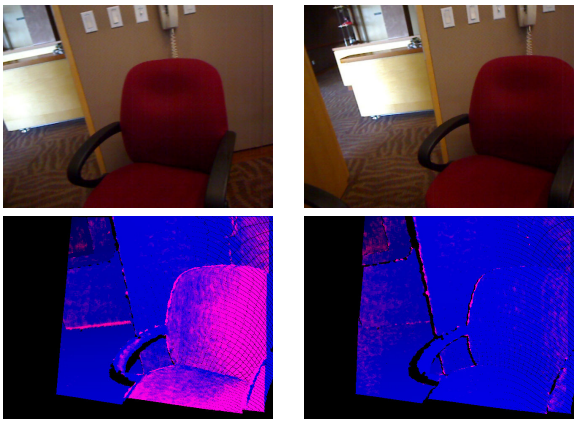

Figure 8. Visibility RANSAC: Top row, image pair. Bottom row, the aligned point cloud. Red: visibility conflicts. Bottom(left), using transform obtained from regular RANSAC. (right) using transform obtained from visibility RANSAC. Original RANSAC #inlier=26, avgdis=1.64. Visibility RANSAC #inlier=23, avgdis = 0.48

6.2. Interactive Mapping

To evaluate the capability of our interactive system to generate improved visual odometry data, we performed a small study in which five persons were tasked to collect data for a map of a small meeting room. A 3D map generated with our system is shown in Fig. 9. For each of the five people, we determined if (s)he was able to collect data that can be consecutively aligned for visual odometry. Three of the people were "expert users" who had substantial experience in using the depth camera for mapping purposes. Two persons were "novice users" who had not previously collected mapping data. The different mapping runs contained between 357 and 781 frames, with roughly 3 frames processed per second.

When using the interactive system, every person was able to collect a data set that covered the entire room and for which all consecutive frames could be aligned. Two of the expert users were able to do so without any intervention by our system. The other three users required on average 16 interventions, that is, the failure detection module in Fig. 2 triggered 16 camera re-localizations, which typically took only 1-2 seconds to achieve. Without the interactive sys-

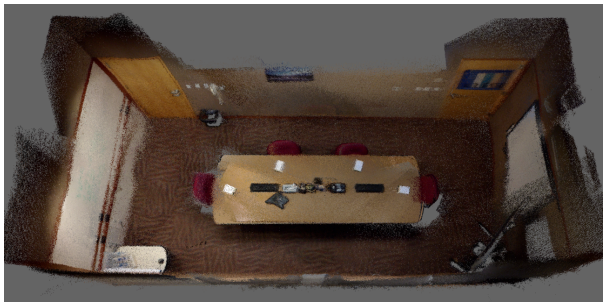

Figure 9. Top down view of 3D map of meeting room used to evaluate the benefits of interactive mapping.

tem, none of these three users was able to successfully capture a sequence without visual odometry failures. The mean time between tracking failures was 20.5 frames. This experiment shows that it is difficult to collect good mapping data without interaction, and that our interactive system makes it easier by overcoming frame-to-frame failures.

6.3. Comparison to PMVS

To demonstrate the advantage of cheap depth cameras over standard cameras for dense 3D modeling, we collected a depth camera sequence and a collection of high-res camera images of a wall and a textured white board standing in front of it. Fig. 10 shows a zoom into the white board part of the reconstruction achieved by our system (left) and by PMVS [8] using the high-res images. As can be seen, while PMVS is not able to generate a dense 3D reconstruction, our system generates a full reconstruction of the whiteboard.

Figure 10. 3D reconstruction achieved by our system (left) and by PMVS using high-res camera images (right). The blue parts in the right image indicate areas without any depth reconstruction.

6.4. Large Scale Mapping

Fig. 11 shows examples of maps we built with our interactive system. These results were achieved by collecting good visual odometry data using the failure detection and re-localization process along with the interactive system for adding pairwise loop closure constraints. For instance, the globally consistent map shown at the top was generated with 25 loop closure constraints originated from the interactive loop closure suggestion strategy.

7. Discussions

We presented an interactive system for building dense 3D reconstructions of indoor environments using cheap depth cameras. Such cameras are extremely promising for enabling people to build maps of their personal spaces. Meanwhile, the solution is far from trivial partly due to high noise and limited field of view.

To generate robust visual odometry estimates, we introduce a RANSAC variant that takes full advantage of both depth and color information. The RANSAC criterion is based on a linear regression function that incorporates additional visibility constraints extracted from the depth data. We demonstrate that our RANSAC approach significantly outperforms existing counterparts.

To ensure that the collected data can be aligned into a globally consistent map, our system continuously checks the quality of frame alignment and alerts the user in case of alignment errors. Such errors can occur, for instance, if the user swipes the camera very quickly or gets too close to a featureless wall. Our experiments indicate that alignment errors are extremely difficult to overcome otherwise, even for expert users. Furthermore, we demonstrate that our system is able to generate consistent 3D maps of large scale indoor environments. In future work, we intend to extract structural information about walls, furniture, and other objects from these 3D maps.

References

[1] S. Agarwal, N. Snavely, I. Simon, S. Seitz, and R. Szeliski. Building Rome in a day. In *CVPR*, pages 72–79, 2010. 1, 2

[2] P. J. Besl and N. D. McKay. A method for registration of 3-d shapes. *IEEE Trans. PAMI*, 14(2), 1992. 2

[3] Y. Chen and G. Medioni. Object modeling by registration of multiple range images. *Image Vision Comput.*, 10(3):145–155, 1992. 2

[4] L. Clemente, A. Davison, I. Reid, J. Neira, and J. Tardós. Mapping large loops with a single hand-held camera. 2007. 2

[5] A. Davison, I. Reid, N. Molton, and O. Stasse. MonoSLAM: Real-time single camera SLAM. *IEEE Trans. PAMI*, pages 1052–1067, 2007. 2

[6] P. Debevec, C. J. Taylor, and J. Malik. Modeling and rendering architecture from photographs: A hybrid geometryand image-based approach. In *SIGGRAPH*, 1996. 2

[7] Y. Furukawa, B. Curless, S. Seitz, and R. Szeliski. Manhattan-world stereo. 2009. 1

[8] Y. Furukawa, B. Curless, S. M. Seitz, and R. Szeliski. Reconstructing building interiors from images. In *ICCV*, 2009. 2, 7

[9] Y. Furukawa and J. Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE Trans. PAMI*, 2009. 2

[10] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge University Press, 2003. 2, 3

Figure 11. (top) 3D map of an office environment built with our system. (bottom) Closeup view and additional test environment.

[11] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox. Rgb-d mapping: Using depth cameras for dense 3d modeling of indoor environments. In *International Symposium on Experimental Robotics*, 2010. 1, 2

[12] B. K. P. Horn. Closed-form solution of absolute orientation using unit quaternions. *J. Opt. Soc. Am. A*, 4(4):629–642, 1987. 2

[13] G. Klein and D. W. Murray. Parallel tracking and mapping for small ar workspaces. In *Int'l. Symp. on Mixed and Augmented Reality*, 2007. 2, 5

[14] K. Konolige, J. Bowman, J. D. Chen, P. Mihelich, M. Calonder, V. Lepetit, and P. Fua. View-based maps. *International Journal of Robotics Research (IJRR)*, 29(10), 2010. 2

[15] D. Lowe. Discriminative image features from scale-invariant keypoints. *Int'l. J. Comp. Vision*, 60(2), 2004. 2

[16] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, and P. Sayd. Real time localization and 3d reconstruction. In *CVPR*, volume 1, pages 363–370, 2006. 2

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

[18] 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. *Int'l. J. Comp. Vision*, 72(2):143–67, 2008. 1, 2

[19] PrimeSense. http://www.primesense.com/. 1

[20] S. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *CVPR*, volume 1, pages 519–528, 2006. 2

[21] S. Sinha, D. Steedly, R. Szeliski, M. Agrawala, and M. Pollefeys. Interactive 3D architectural modeling from unordered photo collections. *ACM Transactions on Graphics (TOG)*, 27(5):1–10, 2008. 1, 2

[22] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: exploring photo collections in 3d. In *SIGGRAPH*, 2006. 1, 2, 5

[23] P. Sturm and S. Maybank. A method for interactive 3d reconstruction of piecewise planar objects from single images. In *BMVC*, pages 265–274, 1999. 2

[24] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle adjustmenta modern synthesis. *Vision algorithms: theory and practice*, pages 153–177, 2000. 2