CPSC601.58 PROJECT
3D RECONSTRUCTION

ERIKA HARRISON

1. Introduction

To-date, much of our understanding of the world around us comes from two dimensional data sources: geographical maps, medical MRI and CT scans, architectural plans and many more. To be able to truly analyze and evaluate the structures and creations in our life, it is important to not simplify them to 2D representations, but instead capture their fully 3D forms. Specifically, three dimensional models of high-precision accuracy can benefit numerous fields including military, manufacturing and scientific.

In capturing the three dimensional models, one must blend the highly successful retrieval aspect of computer vision with the manipulation and visualization techniques of computer graphics. While motion capture systems and high resolution scanners are also good methods for constructing models of real-world objects, their expense and extensive calibration requirements restrict their general use. Advancement of computer vision techniques for highly detailed model reconstruction lends itself to widespread adoption and employment.

From computer vision, feature detection and tracking techniques [3, 21, 37] enable reverse calibration of camera positions. From employment of numerous images or cameras, three dimensional coordinates can be determined, leading to expansive 3D reconstructions of statues, bridges and even small towns [1, 10]. As these approaches require extensive time (a full day) or processing power, the exploitation of computer graphics techniques lends itself to reconstruction improvement. For example, there is the potential to improve the surface quality of these representations, and through multiresolution techniques, speed up the construction process.

The goal is to explore the current research within 3D reconstruction and locate potential areas for improvement. The main objectives are to create a coarse reconstruction via three stages.

(1) Gather 3D Data: Construct 3D data from height map extraction and point cloud-based reconstruction.
(2) Align Data: Merge the 3D data of an additional viewpoint of the same environment with the initial mesh.
(3) Construct Surface: Stitch the 3D data together and construct a 3D mesh of the result.

Within this work, 3D data refers to 3D point cloud data, and the additional viewpoint must include overlap with the initial input. Finally, to merge the data, an ability to uniquely discern corresponding points between viewpoints is required. Therefore the views must have sufficient detail and texture to readily correspond the scenes. This is discussed in more detail in the following sections.

The approach taken is described in greater detail in Section 3 with results presented in Section 4. The paper concludes in Section 5 with a discussion on an overall summary, limitations of the approach, and potential future directions. Beforehand, a general overview of the current techniques and a discussion on important approaches are presented in Section 2.

Date: April 19, 2013.

2. Background

Modern approaches to 3D reconstruction are multi-layered, and span numerous research areas.[1] Initial image capture is followed by camera calibration and point identification (features/keypoints). Points are then transformed into a global coordinate space either individually, or as a disparity (and consequent depth) map. From here, a surface is generated by transforming a planar mesh similar to a height map or by remeshing techniques on the point cloud, comparable to skinning. All this, and simply from one image or pair of images. From this initial mesh, subsequent images require the merger of the resulting surfaces, or alternatively additional remeshing based on the new points in the point cloud. Combined this generates an improved representation of the 3D object. Extensive research has covered much of these topics, with many having implementations either directly from the research, or offered through open source libraries.

Image capture may come from a variety of locations. While single-image processing and reconstruction is explored in computer vision [12, 17], these may require known camera configurations, or extensive assumptions on illumination conditions. As such, multi-image input is considered for the research herein.

Stereo vision offers an easy computation for disparity maps. As explored in CPSC635, stereo images involve two cameras usually offset by a small displacement along a single axis. This small discrepancy enables displacement calculations. For a point located in the two images of the stereo pair, the difference in distance indicates the camera offset and displacement. Initial camera configuration can be computed by reading in numerous known points. From the camera calibration, the disparity of subsequent images may be readily calculated. As was noticed in the assignment for CPSC635, the resulting disparity maps - reliant on Gaussian blurring with a sigma parameter and identification of a disparity level count - may appear inconsistent with an expected depth map. The open source library, OpenCV [5] implements camera calibration and correspondence from stereo images, which employs Birchfield's depth algorithm [4].

The Microsoft Kinect [25] is another multi-image technology. It captures traditional RGB data and uses an infrared laser projector and associated sensor to determine disparity, and consequently depth information about an environment. One can consider the projector and sensor as a partially inverted stereo vision setup. In this case, a known pattern is emitted into the scene, and the sensor detects the offset from its own expected positioning. While this is a highly calibrated system, it is considered a viable input device as it is relatively inexpensive and therefore accessible to researchers. An open source library, Open Natural Interaction (OpenNI) [26] simplifies reading and extracting disparity information from the Kinect, while the Point Cloud Library (PCL) [29] has some examples on working with the resulting data.

Multi-image input refers to one of either single images of a given scene taken from different viewing angles, or alternatively a video input stream where the camera may change location and orientation. In either case, camera calibration may be more challenging to deduce (especially in the case of an arbitrary image collection). The former case has been well explored in the recent papers of Agarwal et al. [1] and later Frahm et al.'s single processor variant [10]. In their research, arbitrarily organized image collections are processed to determine the resulting point cloud scene. Pollefey et al. [27] use video input, but simplify the problem by also taking advantage of GPS for coordinate determination. Han at al. [11], Li et al. [20] and Bregler et al. [6] present examples which use video input for their 3D reconstruction approaches.

To calibrate the cameras, features are often identified within the input images and used to determine offsets between input images. Sufficient numbers of features across the given frames can characterize the corresponding cameras. The most common approach is Lowe's Scale Invariant Feature Transform [21]. Subsequent feature detection schemes and descriptors have been explored,

[1] Text taken from proposal, as there has not been significant change in the related works.

including Bay et al.'s [3] Speeded Up Robust Features, which is also available through the open source computer vision library, OpenCV [5]. With sufficient number of common feature points between images, cameras may be calibrated by constructing the world-to-screen transformation matrix - referred to as a homography within computer vision. A simplified homography assumes an affine projection, while additional feature points could deduce a more general projection. These are well discussed in CPSC635, with OpenCV offering helper functions for this calculation.

From construction of the associated camera transformations, it is then desired to transform the 2D points of the image into 3D world coordinates. This involves point identification between cameras to solve a system of equations for the third coordinate. Again, OpenCV offers helper methods for this calculation. In the case of stereo input, a resulting disparity map can be generated, and can be used to estimate an associated height map for the surface mesh. Schneider et al. [30] present an example of a comparable surface mesh, with an underlying single-orientation assumption. In the case of the PCL Kinect example, a 3D point cloud is generated, and uses its developed mesh construction technique for surface generation such as Kazhdan et al.'s [15] Poisson surface reconstruction.

At this point, a mesh has been constructed from a single direction. To achieve a water-tight, or wholesome model, two steps are required: feature registration within shape identification, and additive mesh construction. Feature registration isn't quite as simple as merely using a SIFT feature for example. If a 3D model is being generated, the view of a given feature from one direction may not correspond to the keypoint, let alone the same feature, from another direction. Agarwal et al. [1] consider feature tracking across images to assist in this. To build a water-tight mesh from different input images one can continue to build a 3D point cloud, using identified keypoints as frames of reference for point placement, and then apply remeshing techniques. Alternatively, Li et al. [20] demonstrate how shape completion techniques may be applied to assist in temporally coherent mesh construction.

Numerous other techniques have been applied to 3D reconstruction. Improving mesh resolution enables improved detail identification [30]. Volumetric approaches are considered [8, 24] wherein reconstruction relies on carving away invalid voxels of data from the silhouettes or contours of the shape. Symmetry has been considered for object completion [2, 14, 33, 36]. Speed improvements have been explored on both the hardware and software [31, 35, 38].

3. Methodology

As alluded to in the introduction, there are three main stages in the reconstruction of a coarse mesh from real world data.

(1) Gathering 3D Data
(2) Aligning Data
(3) Surface Construction

The approaches taken are discussed in more detail in the subsequent subsections.

3.1. **Gathering Data.**

A Microsoft Kinect is used as the method for data acquisition in this work. Despite its recent debut in 2010, it has been the source of much depth-based research [13,16,18] and is increasingly well supported by existing frameworks such as the Open Natural Interface (OpenNI) Library [26], the Point Cloud Library (PCL) [29] and the Robot Operating System [23]. The infrared technology inconjunction with the hardware support, enables a quick determination of a disparity map of the viewed environment, which can be converted into a depth approximation. In connection with the standard RGB-style camera, the captured input can have depth approximately associated with the colour of the scene.

In contrast, alternative input mechanisms include the use of a stereoscopic camera, a collection of monocular cameras, or a single, but moving, monocular camera. Each of these methods provide sufficient information to discern disparity information about the scene by exploiting the parallax effect. For monocular cameras, numerous methods are being explored for 3D reconstruction [10, 11, 34], however their implementations are often highly sophisticated, or rely on learning through a training database - either drastically expanding the time required for implementation. While a stereoscopic camera is more easily calibrated for disparity calculations and therefore point cloud generation, initial trials resulted in poor alignment with undesired artifacting and would require additional exploration.

Capture of Kinect data is easily managed through existing libraries. Through a collection of calls, the PCL library can be used, in conjunction with the OpenNI library, to easily stream data from the Kinect and convert it into a point cloud. Based on the depth and the 2D information, points can be discerned and displayed in a 3D viewer. Figure 1 demonstrates two views of the same point cloud captured from a Kinect.

FIGURE 1. Kinect Point Cloud From Two Views

3.2. **Aligning Data.** Within computer vision, the alignment of data is also referred to as registration. It involves finding the 'same' or corresponding points between images, and determining how they have changed or been transformed. When a projective transformation is involved, as is often the case for images from cameras or eyes, then an homography may be computed to attempt to align one image with the other.

The key problems within this alignment stage is three fold: generating corresponding points (correspondence problem), determining the transformation, and ensuring accurate alignment.

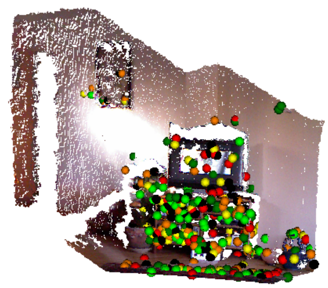
In generating corresponding points, within 2D images the concept of keypoints and associated descriptors are widely used. The idea is to find points (2D coordinates) within the image which are sufficiently distinct as to not be confused with a similar point. The use of gradient, binning for rotational invariance and multiresolution for scale invariance often play a role. These attributes are then used to describe the given keypoint so that if it is found within another image, the difference between the descriptions is small. In an implementation, the descriptions become numerical descriptors for the given keypoint, and a distance metric is defined.

Several methods for finding such keypoints are currently in existence. The Scale Invariant Feature Transform (SIFT) [21] is the most widely used mechanism for keypoint finding as well as descriptor generation. The resulting matching is highly reliable and may be repeatedly applied for successful associations. The Speeded-Up Robust Features [3] is faster at finding keypoints, more stable than SIFT at illumination changes, but isn't as strong at rotational invariance. Other approaches include Binary Robust Independent Elementary Features (BRIEF) [7], Oriented BRIEF (ORB) [28] or Binary Robust Invariant Scalable Keypoints (BRISK) [19] but these often look to reduce the size of the descriptors rather than improving the keypoint detection.

For 3D registration, it is possible to explore surfaces and curves as possible descriptors for the 3D points. However, the limited existing (and readily available) techniques. One such approach is the Normal Aligned Radial Features [32] which relies on an input range (depth/disparity) image. As range images often exhibit jitter reliable keypoints are challenging to discern, especially within scenes where the depth does not vary substantially. More exploration may be required.

Through trials on the sample scenes, it has been determined that for this application, it is more effective to use SURF features over SIFT. Since the Kinect gives both the approximate depth mapping along with the 2D RGB image, 3D keypoints are extracted by first finding the 2D SURF keypoints, and then determining the position of the keypoint as it has been projected into the 3D point cloud. This is readily performed since PCL organizes Kinect point cloud data for efficient association. Figure 2 demonstrates the 2D and 3D keypoints from a given image.

(a) 2D Intensity Image



(b) 3D Point Cloud

Figure 2. SURF Keypoints

For 2D registration and alignment, once corresponding keypoints have been determined, they can then be used to establish the homography transform between the images. In the case of the 3D, there is a significant room for possible errors. For example, the resolution of the depth map is low (480×640), exhibits jitter, and is directly aligned with the 2D image (since there is an offset with the camera) although the hardware attempts to minimize this. Consequently, the resulting keypoints often result in poor matching. Even employing Fastest Library for Approximate Nearest Neighbour from the OpenCV library results in mismatching, as illustrated in Figure 3. Notice how the blue keypoint of the rug on the right matches to a point on the TV screen on the left.

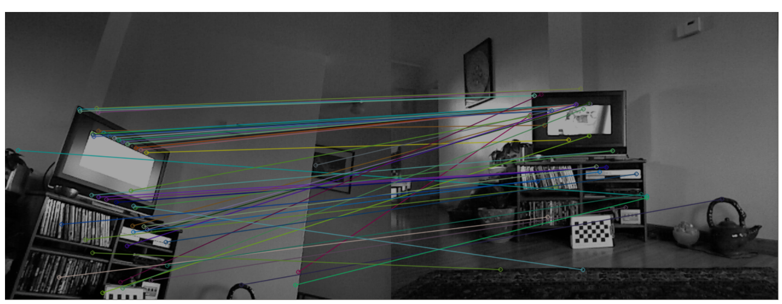


Figure 3. Bad Keypoint Matches

All that's needed, however, to determine the transformation between the two point clouds is the best possible matches. Consequently, a homography is computed that employs Random Sample Consensus (RANSAC) [9] to eliminate poor matches. Since it is not the 2D homography that is desired, we extract out the correspondences that were classified as being significantly better based on the homography transform assumption. While it doesn't eliminate all poor matches as illustrated in Figure 4, it does reduce the match samples to a more plausible collection.

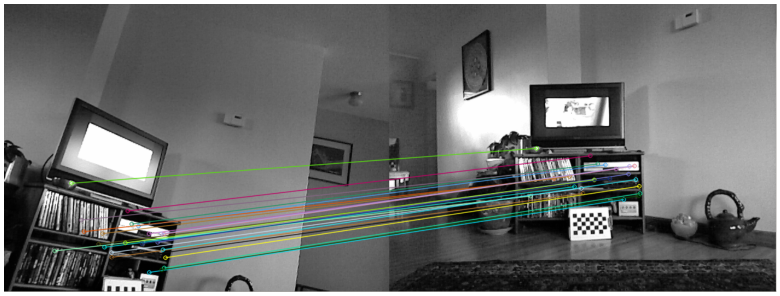


Figure 4. Improved Keypoints Matches via RANSAC

Given these improved correspondences, a 3D transformation can be determined between the two point clouds. A least squares solution is used to resolve the over determined system, which may result in decent, but not quite accurately aligned clouds, as illustrated in Figure 5. Notice how the walls of the two point clouds don't align correctly.

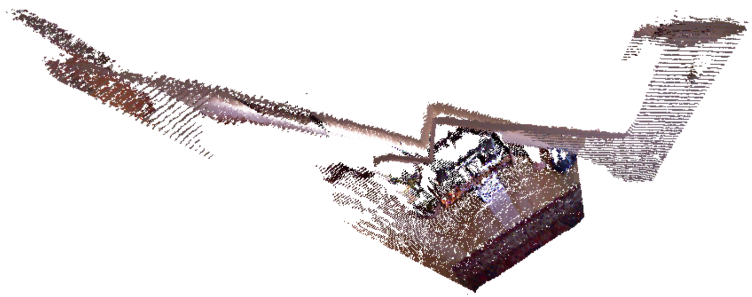


Figure 5. Bad Keypoint Alignment (Top-Down View)

To resolve this poor alignment, existing techniques are explored. The Iterative Closest Point approach was briefly explored, but found to require a strong initial alignment estimate. Instead the Normal Distribution Transform (NDT), as described by Magnusson [22]. While slow to converge requiring several seconds, it creates a cohesive point cloud as illustrated in Figure 6.

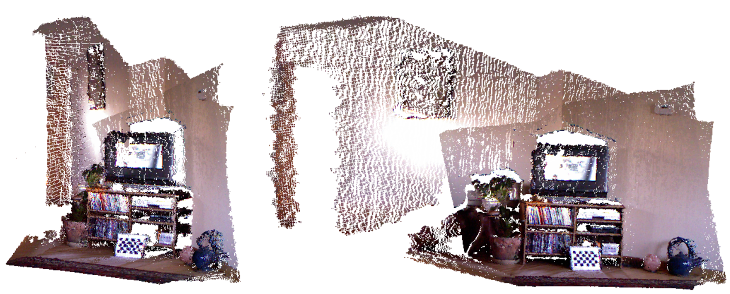


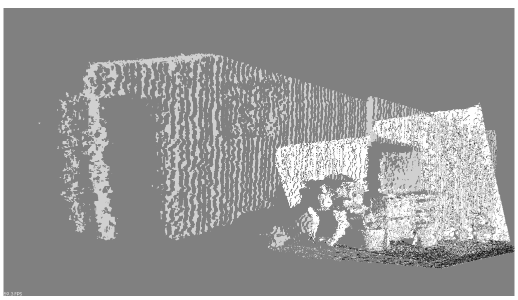
Figure 6. Merged Point Clouds from Different Views

It should be noticed that if NDT is used without prior initialization of the transformation from the correspondence points, another poor alignment results (Figure 7).

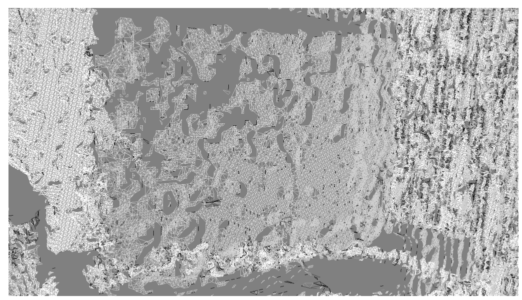
3.3. **Surface Construction.** Finally, the 3D point cloud is converted into a mesh. This is perfomed through simple nearest neighbour triangulation. While the colours and texture of the mesh

Figure 7. Bad NDT Alignment

are not yet captured, the results are visualized in Figure 8. Notice how the jitter from the depth computation of the Kinect results in a bumpy surface.



(a) Overall



(b) Close Up of Screen

Figure 8. Resulting Mesh

4. Results

A collection of different scenes were explored for 3D reconstruction.

Figure 9 illustrates basic alignment between two views from similar perspectives. Notice in the matching how the highly detailed environment results in many positive features matches. This results in an unnecessarily longer time to converge within the least squares transformation finding. Prunning down the number of successful matches is a good way to reduce this bottleneck.

Notice also, how the background for the final merged point cloud has large holes missing. This is due to the depth information being lost from the occlusion of the individuals in the scene. The infrared capture stops at them, and misses the details behind. Merging an alternative viewpoint of the Kinect camera would help to capture this occluded information.

Figure 10 illustrates how bad matching can result in poor convergence for the NDT alignment. Notice that the valid matches cluster around points that will be in a similar location with 3D space, and the final point erroneously matches the couch to the vase. Consequently this creates a poor estimated transformation and poorly aligns the mesh.

Figure 11, on the other hand, doesn't require many matches, but as they are all valid, and not clustered around the same 3D location, able to better estimate the transformation between the two spaces. Notice in the final point cloud how the arm of the chair which is partially occluded in each picture is merged into a more cohesive unit.

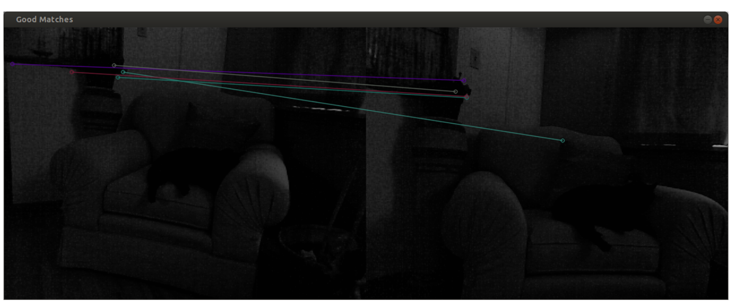
(a) Initial Point Clouds

(b) Keypoint Matches

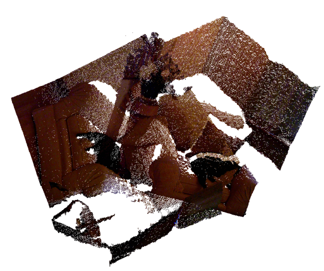
(c) Merged Point Cloud

FIGURE 9. Example 1 - Office Environment

(a) Initial Point Clouds



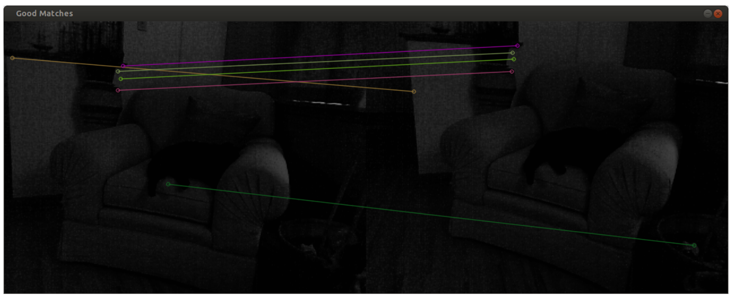
(b) Keypoint Matches



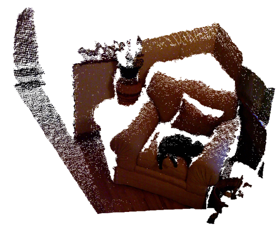
(c) Merged Point Cloud

FIGURE 10. Example 2 - Bad Matching

(a) Initial Point Clouds



(b) Keypoint Matches



(c) Merged Point Cloud

Figure 11. Example 3 - Reduced Matching

5. Conclusion

The works presented haved explored and attempted to over come the challenges faced in 3D reconstruction. Initial data extraction is achieved through the use of the Microsoft Kinect. Data alignment requires an initial transformation estimate through 3D correspondence finding. This is then refined through the iterative alignment process of the Normal Distribution Transform. Finally, the resulting merged point cloud is converted into a mesh using nearest neighbour triangulation.

Numerous limitations are faced by the existing approach. For example, by employing the Kinect, one is limited to an indoor environment with relatively low depth resolution. The finer details and bevels within a scene are lost, alongside the standard occlusion resulting from depth shadowing. Additionally, due to the iterative alignment process, a good close estimate of the point clouds must be determined during the initial data extraction. Applying the same process using a stereoscopic camera or monocular video feed will require supplemental processing for this 'good' initial input. Finally, if there are insufficient matches between the different input images, due to low detail, poor matches, or clustering of match points, then a poor initial transformation is generated, and often results in a poor resulting surface. Identifying when this is the case maybe useful to avoid such unhelpful input data.

Immediate areas of improvement include the construction of a reliable initial transformation by improving matching. For example, if a 3D transformation is explored rather than the 2D homography, perhaps RANSAC will generate better results. Furthermore, the simultaneous alignment of streamed input, and not simply two point clouds, is beneficial for whole-environment reconstruction. For the alignment of streamed input, the keypoints and their descriptors must be stored in 3D to help in the alignment of new data. Efficiency measures include reducing or eliminating redundant keypoints, and ignoring input frames which do not significantly contribute to the overal scene construction. Finally, for this approach to be beneficial for overall surface construction, immediate or near-realtime generation is important. Consequently the iterative approaches for mesh alignment will need to be explored for speed improvements.

More fundamental extensions include the ability to identify and extract out independent objects for subsequent analysis. Being able to determine an object, or entity's plausible movement mechanisms enable an animation-based modeling of the scene and can aid in a more expansive analysis of the resulting environment.

This lays the fundamental building blocks for improving and expanding on high resolution reconstruction techniques. Highly detailed and accurate reconstruction techniques could enable inspectors of structures such as pipelines, bridges and buildings to identify surface-based weaknesses or deteriorations due to environmental or mechanical stressors. Medical examinations could also benefit from such approaches by capturing and comparing surface changes to patients in the identification of growths, tumours, moles and other deformations. As the computer can store such information across many years, this vastly enhances the visual inspections, rough notes, and human memory. Expanding upon existing knowledge and approaches for 3D reconstruction will broaden our understanding of the world, as we facilitate the analysis of increasingly larger datasets and overall environments.

References

[1] Sameer Agarwal, Yasutaka Furukawa, Noah Snavely, Ian Simon, Brian Curless, Steven M. Seitz, and Richard Szeliski. Building Rome in a Day. *Commun. ACM*, 54:105–112, Oct. 2011.

[2] V. Avagyan. *Shape Matching and Symmetric Reconstruction of 3D Models for Manufacturing Applications*. University of Michigan-Dearborn, 2005.

[3] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-Up Robust Features (SURF). *Comput. Vis. Image Underst.*, 110:346–359, June 2008.

[4] S. Birchfield and C. Tomasi. Depth Discontinuities by Pixel-to-Pixel Stereo. In *Computer Vision, 1998. Sixth International Conference on*, pages 1073 –1080, Jan 1998.

[5] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[6] Christoph Bregler, Aaron Hertzmann, and Henning Biermann. Recovering Non-Rigid 3D Shape from Image Streams. pages 690–696, 2000.

[7] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. BRIEF: Binary Robust Independent Elementary Features. In *Proceedings of the 11th European conference on Computer vision: Part IV*, ECCV'10, pages 778–792, Berlin, Heidelberg, 2010. Springer-Verlag.

[8] Brian Curless and Marc Levoy. A Volumetric Method for Building Complex Models from Range Images. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, SIGGRAPH '96, pages 303–312, New York, NY, USA, 1996. ACM.

[9] Martin A. Fischler and Robert C. Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Commun. ACM*, 24(6):381–395, June 1981.

[10] Jan-Michael Frahm, Pierre Fite-Georgel, David Gallup, Tim Johnson, Rahul Raguram, Changchang Wu, Yi-Hung Jen, Enrique Dunn, Brian Clipp, Svetlana Lazebnik, and Marc Pollefeys. Building Rome on a Cloudless Day. In *Proceedings of the 11th European conference on Computer vision: Part IV*, ECCV'10, pages 368–381, Berlin, Heidelberg, 2010. Springer-Verlag.

[11] Bing Han, Christopher Paulson, and Dapeng Wu. 3D Dense Reconstruction from 2D Video Sequence via 3D Geometric Segmentation. *J. Vis. Comun. Image Represent.*, 22:421–431, July 2011.

[12] D. Hoiem, A.A. Efros, and M. Hebert. Geometric Context from a Single Image. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 1, pages 654–661, Oct. 2005.

[13] Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard A. Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew J. Davison, and Andrew W. Fitzgibbon. KinectFusion: Real-Time 3D Reconstruction and Interaction Using a Moving Depth Camera. In *UIST'11*, pages 559–568, 2011.

[14] Nianjuan Jiang, Ping Tan, and Loong-Fah Cheong. Symmetric Architecture Modeling with a Single Image. *ACM Trans. Graph.*, 28:1–8, December 2009.

[15] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson Surface Reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, SGP '06, pages 61–70, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association.

[16] K Khoshelham. Accuracy Analysis of Kinect Depth Data. volume 38, pages 1–6, 2010.

[17] Kevin Köser, Christopher Zach, and Marc Pollefeys. Dense 3D Reconstruction of Symmetric Scenes from a Single Image. In *Proceedings of the 33rd International Conference on Pattern Recognition*, DAGM'11, pages 266–275, Berlin, Heidelberg, 2011. Springer-Verlag.

[18] C. Kuster, T. Popa, C. Zach, C. Gotsman, and M. Gross. FreeCam: A Hybrid Camera System for Interactive Free-Viewpoint Video. In *Proceedings of Vision, Modeling, and Visualization (VMV)*, 2011.

[19] Stefan Leutenegger, Margarita Chli, and Roland Y Siegwart. BRISK: Binary Robust Invariant Scalable Keypoints. In *International Conference on Computer Vision (ICCV)*. IEEE, 2011.

[20] Hao Li, Linjie Luo, Daniel Vlasic, Pieter Peers, Jovan Popović, Mark Pauly, and Szymon Rusinkiewicz. Temporally Coherent Completion of Dynamic Shapes. *ACM Transactions on Graphics*, 31(1), January 2012.

[21] D.G. Lowe. Object Recognition from Local Scale-Invariant Features. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 2, pages 1150–1157, 1999.

[22] M. Magnusson. *The Three-Dimensional Normal-Distributions Transform - an Efficient Representation for Registration, Surface Analysis, and Loop Detection.* Dec 2009.

[23] M. Quigley and B. Gerkey and K. Conley and J. Faust and T. Foote and J. Leibs and E. Berger and R. Wheeler and A. Y. Ng. ROS: An Open-source Robot Operating System. *Proc. Open-Source Software Workshop of the International Conference on Robotics and Automation (ICRA)*, 2009.

[24] A.Y. Mulayim, U. Yilmaz, and V. Atalay. Silhouette-Based 3-D Model Reconstruction From Multiple Images. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 33(4):582–591, Aug. 2003.

[25] Richard A. Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J. Davison, Pushmeet Kohli, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. KinectFusion: Real-time Dense Surface Mapping and Tracking. In *Proceedings of the 2011 10th IEEE International Symposium on Mixed and Augmented Reality*, ISMAR '11, pages 127–136, Washington, DC, USA, 2011. IEEE Computer Society.

[26] OpenNI organization. *OpenNI User Guide*, November 2010. Last viewed 19-01-2011 11:32.

[27] Pollefeys, M. and Nistér, D. and Frahm, J. -M. and Akbarzadeh, A. and Mordohai, P. and Clipp, B. and Engels, C. and Gallup, D. and Kim, S. -J. and Merrell, P. and Salmi, C. and Sinha, S. and Talton, B. and Wang, L. and Yang, Q. and Stewénius, H. and Yang, R. and Welch, G. and Towles, H. Detailed Real-Time Urban 3D Reconstruction from Video. *Int. J. Comput. Vision*, 78(2-3):143–167, July 2008.

[28] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. ORB: An Efficient Alternative to SIFT or SURF. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2564–2571, 2011.

[29] Radu Bogdan Rusu and Steve Cousins. 3D is Here: Point Cloud Library (PCL). In *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, May 9-13 2011.

[30] D.C. Schneider, M. Kettern, A. Hilsmann, and P. Eisert. A Global Optimization Approach to High-Detail Reconstruction of the Head. pages 9–15, Oct 2011.

[31] Jonathan Starck and Adrian Hilton. Surface Capture for Performance-Based Animation. *IEEE Comput. Graph. Appl.*, 27:21–31, May 2007.

[32] B. Steder, R. B. Rusu, K. Konolige, and W. Burgard. NARF: 3D Range Image Features for Object Recognition. In *Workshop on Defining and Solving Realistic Perception Problems in Personal Robotics at the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2010.

[33] Demetri Terzopoulos, Andrew Witkin, and Michael Kass. Symmetry-Seeking Models and 3D Object Reconstruction. *International Journal of Computer Vision*, 1:211–221, 1987.

[34] C. Wefelscheid, R. Hnsch, and Olaf Hellwich. Three-Dimensional Building Reconstruction Using Images Obtained by Unmanned Aerial Vehicles. September 2011.

[35] Ruigang Yang, G. Welch, and G. Bishop. Real-time Consensus-Based Scene Reconstruction Using Commodity Graphics Hardware. In *Computer Graphics and Applications, 2002. Proceedings. 10th Pacific Conference on*, pages 225–234, 2002.

[36] Lilit Yenokyan, Armen Zakarian, Pravansu Mohanty, and Vahram Avagyan. Symmetric Reconstruction Algorithms for Incomplete 3D Models. *IJCAT*, 34(2):143–151, 2009.

[37] Alper Yilmaz, Omar Javed, and Mubarak Shah. Object Tracking: A Survey. *ACM Comput. Surv.*, 38, December 2006.

[38] Kun Zhou, Minmin Gong, Xin Huang, and Baining Guo. Data-Parallel Octrees for Surface Reconstruction. *Visualization and Computer Graphics, IEEE Transactions on*, 17(5):669–681, May 2011.