HawkEye: Automatic Stitching of Hand-Held LIDAR Scans using Photogrammetry and Structure-from-Motion

Shawn Recker, Rob Baltrusch, Christiaan Gribble, and Mark Butkiewicz
SURVICE Engineering Company
Applied Technology Operation
4694 Millennium Drive, Suite 500, Belcamp, MD 21017
530-564-8763
{shawn.recker, rob.baltrusch, christiaan.gribble, mark.butkiewicz}@survice.com

Word Count: 3301
Images Submitted: 4

Abstract

The reconstruction of scene from multiple images or video streams has become an essential component in many modern applications. This paper presents a hand-held consumer-grade optical scanner and supporting application suite, called HawkEye. The fully automatic reconstruction system is based upon advancements in hybrid photogrammetry/structure-from-motion technologies and produces a globally aligned 3D model. Analysis of HawkEye results demonstrate visually accurate scene reconstructions.

1.0 Introduction

Numerous applications—from aligning mirrors on the latest telescopes to accurate body analysis of burn patients—depend on easy and accurate measurements. The field of metrology specializes in approaches to solve the precise measurement problem. Photogrammetry, in particular, uses targets (either non-coded or coded) placed on or around an object to compute the target positions in three-dimensional (3D) space from two-dimensional (2D) photographs. This time intensive process, although accurate, traditionally generates only 3D geometry for target positions[1].

Other metrology techniques, such as Light Detection and Ranging (LIDAR), provide dense 3D geometry of the imaged scene. Broadly, LIDAR creates distance measurements by illuminating a target with laser light and timing the return. Unfortunately, such systems are expensive, cumbersome, and time-consuming to use. Moreover, globally stitching LIDAR scans together is a semi-manual process that relies on accurate user-selected matches or careful calibration of equipment, or that employs high-precision motorized controls to set device poses.

To combat the issues and costs associated with current approaches and processes, we developed a hand-held consumer-grade optical scanner and supporting application suite, called HawkEye™. Our system utilizes recent advancements in hybrid photogrammetry/structure-from-motion technologies[2] to provide a fully automated stitching process that is inexpensive and provides an accurate and dense 3D reconstruction.

2.0 Background

The following section provides relevant information to key technologies related to our HawkEye system. The first section reviews 3D reconstruction algorithms, and the second section details relevant depth camera technology and algorithms.

2.1 Structure-from-Motion Algorithms

A variety of techniques extract 3D information from 2D images. The traditional solution is typically referred to as multi-view scene reconstruction, but the general algorithm is also known as multi-view stereo or structure-from-motion (SfM). While many different algorithms for multi-view reconstruction are available, the following stages are necessary to achieve a 3D reconstruction:

- Typically, feature matches (between consecutive pairwise views) and tracks (concatenating across multiple views) are first obtained. Such matching and tracking can be sparse or dense, and consists of computing and linking the pixel coordinates in all images for each particular scene point, wherever it is visible. While numerous feature tracking algorithms exist in the literature, the general process involves computing high-dimensional feature vectors from local image patterns that are invariant to different transformations, followed by finding vectors that are close geometrically[3,4,5].

- Next, camera intrinsic calibration is performed by a process known as self-calibration, which aims to recover the cameras' intrinsic parameters, for example, focal length. In many cases, such information is already known from specification sheets, and this process is not necessary. For an overview of self-calibration, the reader is referred to Hartley et al[6].

- Given an initial image pair, the 'epipolar geometry' can be estimated from the feature matches. Epipolar geometry mathematically encapsulates the intrinsic projective geometry between the pair of views. This geometry is encapsulated in the 'essential matrix', E, which is then used to extract the relative camera pose parameters between the two views[6].

- Once initial poses have been estimated, computation of the scene's 3D structure is achieved by triangulation methods. In standard linear triangulation, solving for the best-fit 3D scene point involves setting up a $2n \times 4$ data matrix and performing SVD or eigen analysis to obtain a solution of a system in the form $AX = 0$ for each 3D position[6].

- With the initial 3D structure estimated, additional camera poses are solved via a camera resectioning algorithm, the most common of which is, perhaps, the Efficient Perspective N-Point (EPnP) algorithm[7]. The algorithm uses 3D structure and corresponding 2D image points to infer the camera pose Its solution is expressed as the weighted sum of the null eigenvectors of a matrix M of size $2n \times 12$ or $2n \times 9$. The correct linear combination is the one that yields 3D camera coordinates. Any new poses are used to compute additional 3D scene structure through triangulation methods. This process continues until all images have been processed.

- Finally, because errors in the above steps influence accuracy of the computed structure, bundle adjustment is performed to optimize camera and structure parameters. Typically, the Levenberg-Marquardt algorithm is used to minimize 'reprojection error' of all computed structure points across all cameras[8].

Numerous algorithms for scene reconstruction that employ all or a number of the previously mentioned steps currently exist in the literature. Comprehensive overviews and comparisons of different methods are presented by Strecha et al[9]. Additionally, software packages such as Bundler[10] and VisualSFM[11] can estimate all calibration and structure parameters from a set of images. Additional examples of scene reconstruction algorithms include: Akbarzadeh et al.[12] introduced a method for dense reconstruction of urban areas from a video stream; Pollefeys et al.[13] used a similar approach for real-time urban reconstruction. Goesele et al.[14] presented a reconstruction pipeline for large, unstructured collections of online photographs of a scene, based on an adaptive view selection technique that robustly computes depth maps along with 3D models of the scene. Snavely et al.[10] presented a system for interactively browsing and exploring large unstructured collections of photographs of a scene. This system used an image-based modeling front-end that automatically computes the viewpoint of each photograph, as well as a sparse 3D model of the scene.

2.2 RGB-Depth Cameras and Algorithms

RGB-D cameras, commonly known as depth cameras, have become very popular over the past few years, particularly with the advent of the Microsoft Kinect. This device has brought quality, low-cost, and real-time depth sensing to consumers, including researchers and enthusiasts. The first generation Kinect uses structured light to generate real-time depth maps containing discrete range measurements of the physical scene[15]. The second generation Kinect, while producing the same output, is based on a time-of-flight system. This data can be reprojected as a set of discrete 3D points or *point cloud*. The main issue with acquired depth data is noise. Depth measurements often fluctuate and depth maps contain 'holes,' or regions in which no readings are obtained. To generate 3D models for use in applications such as gaming, physics, or CAD, higher-level surface geometry must be inferred from the noisy data[15].

Perhaps the most popular application of the Kinect specifically, and of RGB-D cameras generally, is the highly-successful Kinect Fusion algorithm by Izadi et al.[15]. Kinect Fusion enables a user holding and moving a standard Kinect camera to rapidly create detailed 3D reconstructions of an indoor scene. In this case, only the depth data from Kinect is used to track the 3D pose of the sensor and reconstruct a single and accurate 3D model of the physical scene in real-time, such that a user can move the Kinect within any indoor space and reconstruct a 3D model of the scene in seconds. The system continuously tracks the 6 degrees-of-freedom (DOF) pose of the camera and fuses new viewpoints of the scene into a global surface-based representation. A novel GPU pipeline enables accurate camera tracking and surface reconstruction at real-time rates.

The memory required when construction and updating the global model comprise one issue for the original Kinect Fusion algorithm. To address this issue, a number of more recent algorithms deal with efficient model creation. Quiroga et al.[16] proposed a scene flow approach that exploits the local and piece-wise rigidity of real world scenes. By modeling motion as a field of twists, this method encourages piece-wise smooth solutions of rigid body motions. A general formulation is provided to solve for local and global rigid motions by jointly using intensity and depth data. Another approach to computing dense scene flow between a pair of consecutive RGB-D frames is presented by Hornacek et al.[17] The availability of depth data is exploited by seeking correspondences with respect to patches specified not as the pixels inside square windows, but as 3D points that are the inliers of spheres in world space. Steinbrucker et al.[18] introduced a method to generate highly detailed, textured 3D models of large environments from RGB-D sequences. This system runs in real-time on a standard desktop PC. To reduce memory consumption, the acquired depth maps and colors are fused in a multi-scale octree representation of a signed distance function. To estimate camera poses, a pose graph is constructed and dense image alignment determines the relative pose between pairs of frames. Thomas and Sugimoto[19] describe a new 3D scene representation using a set of planes that is memory efficient and achieves accurate reconstruction of indoor scenes from RGB-D image sequences. Here, projecting the scene onto different planes reduces significantly the size of the scene representation and allows generation of a global textured 3D model with lower memory requirements while both maintaining accuracy and enabling straightforward updates with live RGB-D measurements. Raposo et al.[20] presented a novel approach for estimating relative motion between successive RGB-D frames using plane-primitives instead of point features. Planes in the scene are extracted, and motion estimation is cast as a plane-to-plane registration problem with a closed-form solution. The algorithm by Bylow et al.[21] can also reconstruct large-scale 3D scenes involving many planar surfaces.

3.0 Methodology

As mentioned previously, we have developed a hand-held consumer-grade optical scanner and supporting application suite, named HawkEye. Our system is based on a hybrid photogrammetry/structure-from-motion 3D reconstruction system[2]. The end-to-end system captures RGB-D images and automatically stitches the entire collection together to provide an accurate and dense reconstruction. Figure 1 demonstrates the system pipeline.

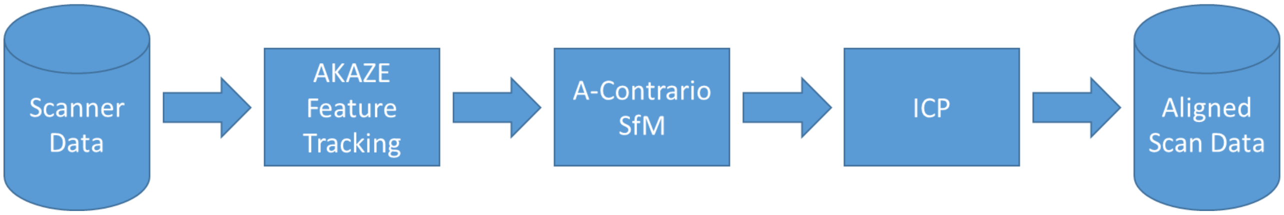
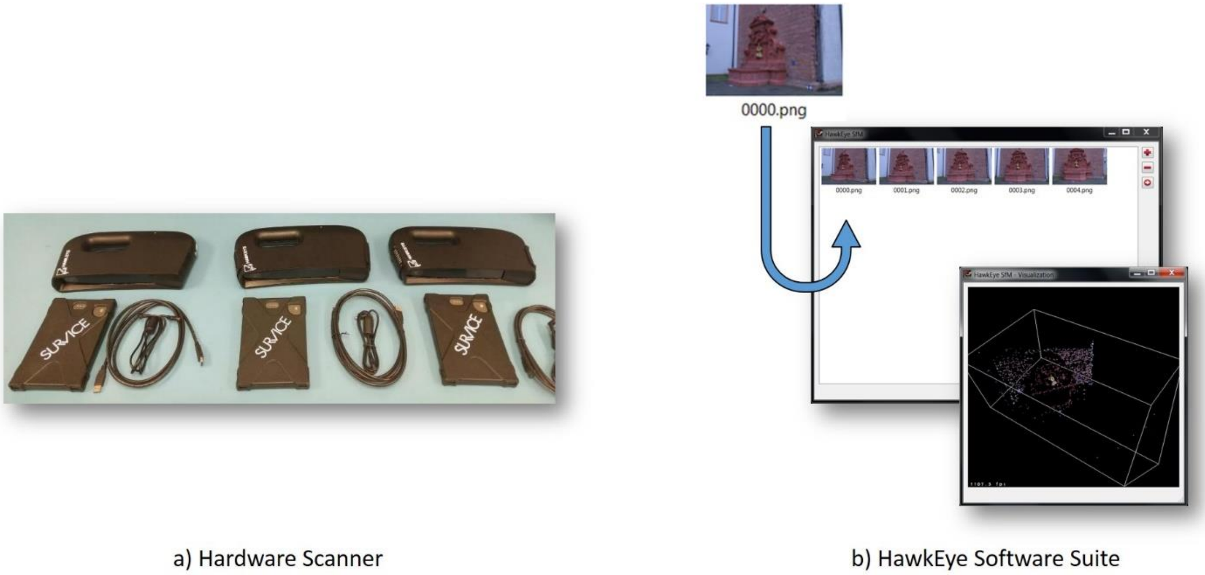


Figure 1: HawkEye Reconstruction Pipeline. The pipeline begins with the capture of data from our scanner. The scanner image is composed of both an RGB image and point cloud (computed from depth information). This collection of scanner data then has visual features computed from the AKAZE feature tracking algorithm and is used to estimate the pose information for each scanner image. Poses are computed using an a-contrario structure-from-motion (SfM) algorithm. The initial poses are used to align the point clouds, which are further refined via Iterative Closest Point (ICP). The aligned point cloud is then saved to disk.

The scanner utilized in this pipeline is based on the Microsoft Kinect v2 optics[15], which includes both a time-of-flight depth sensor (512 × 424 resolution, 30 Hz, 0.5 – 4.5 meter range) and an optical 1080p RGB camera (30 Hz). Scanner data is packed into a *ply* format that includes both the RGB image and the point cloud generated from the depth sensor. Data collection is achieved by executing a standalone application that interfaces with the device to store a collection of *ply* data 'snapshots'. The snapshots should be views of the scene to be reconstructed containing photogrammetry targets. The targets are used to derive metric distances in the reconstruction. Ultimately, these snapshots are globally aligned, in the reconstruction application suite, to produce the final 3D scan. Figure 2 depicts the hardware scanner and reconstruction application suite.



a) Hardware Scanner

b) HawkEye Software Suite

Figure 2: HawkEye Scanner Hardware and Application Software. The HawkEye pipeline utilizes the portable hand-held scanner shown in part (a). The scanner is based upon the Microsoft Kinect v2 optics[15] package, including an HD RGB camera and time-of-flight IR sensor. The output of the scanner is provided to a standalone reconstruction suite capable of globally aligning the individual scan snapshots.

Once scaning is complete, the data is provided to reconstruction software based on a hybrid photogrammetry/structure-from-motion system[2]. Specifically, we employ AKAZE feature tracking to compute feature tracks across the RGB image data for use in an A-Contrario Structure-from-Motion (AC-

SfM) algorithm[22]. SfM results provide initial estimates of the relative scanner poses for each snapshot in the collection. These initial poses are further refined and upgraded to a metric reconstruction based on the photogrammetry target information also present in the scan.

3.1 A-Contrario Structure-from-Motion

As previously mentioned, using AC-SfM to obtain initial pose estimates from the scanner data collection is a core component of the HawkEye system. While Moulon et al. provide a detailed summary of the *a-contrario* model and its application to SfM[22], we summarize the method here for completeness.

The incremental SfM pipeline is a growing reconstruction process in which each iteration contributes additional 3D point cloud information, after computing the new view pose. Due to the incremental nature, successive steps of non-linear refinement, such as bundle adjustment, are used to minimize accumulated error. Traditional SfM pipelines employ RANSAC to estimate inliers from random noise present in 3D reconstruction parameters[2]. Unfortunately, RANSAC depends on a critical parameter, namely, the threshold value that separates inliers from noise, and requires advanced knowledge of the SfM process to choose correctly. The approach implemented by Moulon et al. utilizes *a-contrario* model estimation to avoid this undesirable threshold[22].

The a-contrario (AC) methodology is based on the Helmholtz principle: “an observed strong deviation from the background model is relevant information.” When applied to model estimation, this principle asks, “Does this model arise by chance?” The corresponding statistical criterion is data-specific and avoids threshold values, ultimately providing a parameter-free version of RANSAC, called AC-RANSAC[22]. AC-RANSAC looks for a consensus set that includes a controlled Number of False Alarms (NFA). A false alarm is a model that occurs by chance, and the NFA is described by the following formula:

$$NFA(M,k) = N_{out}(n - N_{sample})\binom{n}{k}\binom{k}{N_{sample}}\left(e_k(M)^d\alpha_0\right)^{k-N_{sample}}. \tag{1}$$

Here, k is the number of hypothesized inlier correspondences, n is the total number of correspondences, N_{sample} is the cardinal of a RANSAC sample (the size), N_{out} is the number of models that can be estimated from a RANSAC sample of N_{sample} correspondences, $e_k(M)$ is the k-th lowest error to the model M among all n correspondences, α_0 is the probability of a random correspondence has a specified error value (in Moulon et al.’s case a reprojection error of value one), and d is the error dimension. Model M is considered valid if:

$$NFA(M) = \min_{k= N_{sample}+1\dots n} NFA(M,k) \le \epsilon, \tag{2}$$

where ϵ is the only parameter and its value is set to one. The AC model estimate requires finding the solution to:

$$\arg\min_{M} NFA(M). \tag{3}$$

In AC-SfM, model estimations require the fundamental matrix, homography, essential matrix, and pose estimation algorithms. Moulon et al. defined $\alpha_0 = \frac{\pi}{A}$ (the ratio of the radius 1 disk area to image area A) and $d = 2$ for homography and camera pose estimation. For essential and fundamental matrix estimation, $\alpha_0 = \frac{2D}{A}$, where D is the image diameter and $d = 1$. The a-contrario approach thus results in adaptive thresholds for all components of the SfM pipeline that require robust model estimation[22].

3.2 Hybrid Photogrammetry/Structure-from-Motion

At its core, the HawkEye system utilizes photogrammetry-enhanced structure-from-motion to perform an initial 3D reconstruction and align the collected scanner data. Specifically, we modify Recker et al.'s hybrid photogrammetry/structure-from-motion pipeline[2], substituting the AC-SfM pipeline described in the previous section.

In the hybrid photogrammetry/structure-from-motion (photo-SfM) pipeline, photogrammetry target positions are placed in the scene and imaged by the scanner. Target positions are tracked, along with AKAZE features, and serve as input to the AC-SfM pipeline. The output from the AC-SfM reconstruction pipeline is a 3D point cloud, corresponding to the tracked features and photogrammetry targets, along with the scanner pose information for each snapshot. At this point, the reconstruction is only relative: ratios of distances between scene points are valid, but the distance values themselves are meaningless. The photogrammetry targets are used to 'upgrade' the relative reconstruction to a metric reconstruction, where distances between scene points correspond to physical units.

To promote the relative reconstruction, two photogrammetry targets for which the measured distance, d_{real}, between them is known, are selected in the application. Relative distance between targets, $d_{relative}$, in the reconstruction is computed, and the relative reconstruction is scaled by $\frac{d_{real}}{d_{relative}}$. At this point, point cloud data from each scanner snapshot (obtained from the depth sensor) is transformed into the global coordinate system from the estimated scanner snapshot pose. The result is a globally aligned version of the collected scanner data.

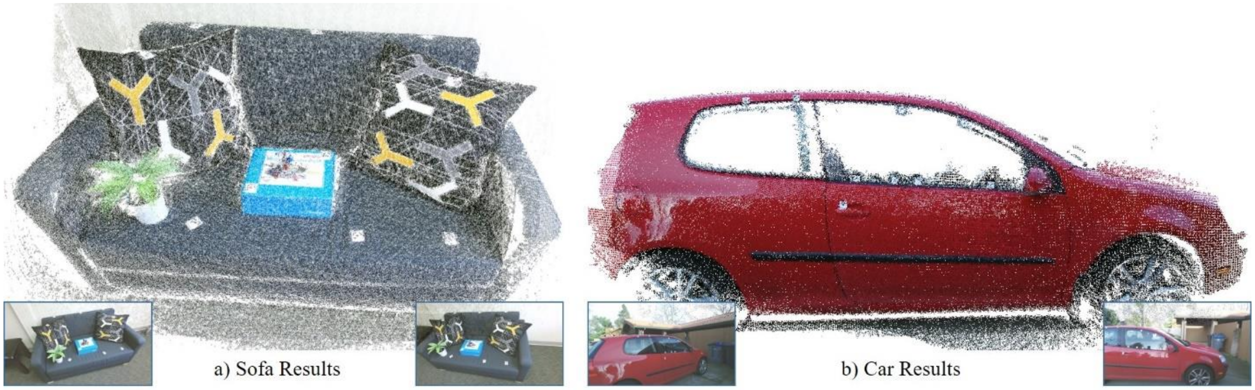
3.3 Iterative Closest Point

As with any reconstruction process, small estimation errors ultimately result in slightly misaligned point cloud data. To minimize the impact of these small errors, we utilize the Iterative Closest Point (ICP) algorithm to further refine the initial alignment. In traditional ICP, a point cloud known as the *reference* or *target* is kept fixed, while another point cloud, the *source*, is transformed to best match the reference. The algorithm is iterative, and the essential steps involve finding the closest point in the source for each point in the reference. From these correspondences, ICP estimates the rigid transformation (translation and rotation) using a mean squared error cost function. For our system, we utilize the ICP version proposed by Zhang[23] and implemented in the Point Cloud Library (PCL).

4.0 Results

To verify the capabilities of the HawkEye system, we developed tests to explore the algorithm's general behavior and accuracy. The implementation of both the data collection application and reconstruction suite were written in C++. Results were obtained on a Desktop PC running Ubuntu 14.04 with an Intel Core i5 processor with 24 GB of RAM with an NVIDIA Tesla K40c accelerator.

Initial experiments investigated results generated the HawkEye pipeline visually. Toward this end, we captured a simple indoor scene of a sofa, as well as an outdoor scene of the side of a car. The resulting point clouds are displayed in Figure 3. From the figure, it is easily observed that the alignment process produces visually accurate results, thereby demonstrating an effective pipeline.

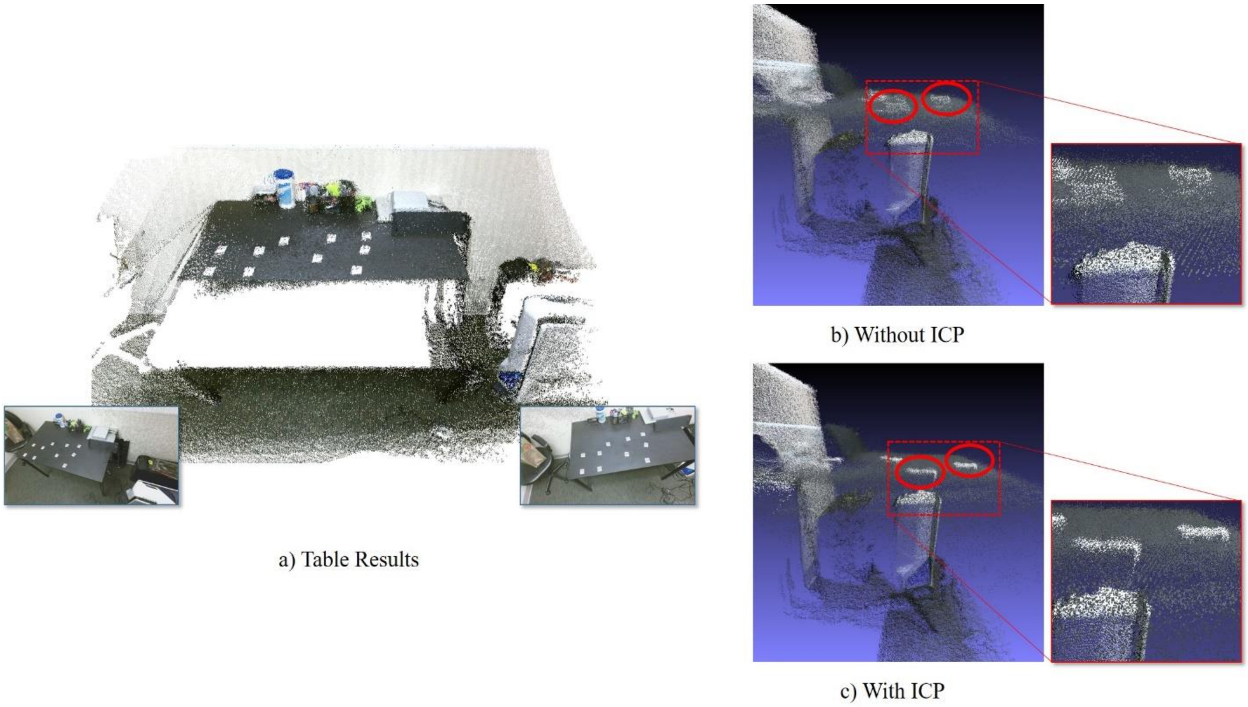


a) Sofa Results

b) Car Results

Figure 3: Initial HawkEye Pipeline Results. Initial experiments investigated the effectiveness of the HawkEye reconstruction pipeline using two datasets: an indoor scene of a sofa (8 snapshots) and an outdoor scene of the side of a vehicle (6 snapshots). The pipeline clearly aligns the scanner data to produce visually accurate results.

The second experiment focused on the importance of the ICP refinement step. For this experiment, seven snapshots of a were processed to produce both initial alignment results and the results after ICP refinement. Figure 4 depicts the results. As shown in the figure, ICP refinement rectifies minor misalignments, ultimately leading to more accurate reconstructions.



a) Table Results

b) Without ICP

c) With ICP

Figure 4: ICP Refinement Results. For this experiment, seven snapshots of a table were captured and processed both with and without the final ICP refinement step. The overall result is displayed in part (a). In part (b), minor misalignments are easily seen in the 'blurring' of the photogrammetry targets. However, with ICP, the targets are properly aligned, producing the expected flat result, as shown in part (c).

5.0 Discussion and Conclusions

The HawkEye hand-held reconstruction scanner and software suite provides a portable solution for creating 3D point clouds. The scanner hardware is based on the Microsoft Kinect v2, a consumer-grade optical sensor package. In combination with hybrid photogrammetry/structure-from-motion software, the HawkEye scanner provides a fast, accurate solution to global alignment of individual snapshots. Specifically, the reconstruction software utilizes the a-contrario structure-from-motion pipeline along with photogrammetry target information to first perform a relative 3D reconstruction. Ultimately, photogrammetry targets provide metric distances and globally align the 3D scanner data.

The HawkEye pipeline was tested for its accuracy and behavior in both indoor and outdoor scenes. Initial results demonstrate accuracy to within the underlying hardware limits, and HawkEye clearly generates visually accurate results. Moreover, ICP refinement increases accuracy, by rectifying minor misalignments in the initial 3D reconstruction.

In the future, the HawkEye pipeline can integrate different—and more accurate—data acquisition devices to further increase reconstruction accuracy. Additionally, the SfM and ICP algorithms on which HawkEye is based can be further enhanced by optimizing their implementation for particular computing architectures, for example, a heterogeneous platform with both multicore CPUs and one or more many core GPUs.

Acknowledgement

This work was supported in part by the US Marine Corps SBIR program, and SURVICE Engineering's Internal R&D program. The authors thank their colleagues in the Applied Technology Operation at SURVICE Engineering for their support.

References

[1] T. Dodson, R. Ellis, C. Priniski, S. Raftopoulos, D. Stevens, and M. Viola, "Advantages of high tolerance measurements in fusion environments applying photogrammetry," in Fusion Engineering, 2009. SOFE 2009. 23rd IEEE/NPSS Symposium on, pp. 1–4, June 2009.

[2] S. Recker, M. M. Shashkov, M. Hess-Flores, C. Gribble, R. Baltrusch, M. A. Butkiewicz, K. I. Joy, "Hybrid Photogrammetry Structure-from-Motion Systems for Scene Measurement and Analysis." *Coordinate Metrology Systems Conference*, 2014.

[3] D. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal On Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[4] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *Computer Vision – ECCV 2006*, ser. Lecture Notes in Computer Science, A. Leonardis, H. Bischof, and A. Pinz, Eds. Springer Berlin / Heidelberg, 2006, vol. 3951, pp. 404–417, 10.1007/11744023.32.

[5] S. S. Beauchemin and J. L. Barron, "The computation of optical flow," *ACM Comput. Surv.*, vol. 27, pp. 433–466, September 1995. http://doi.acm.org/10.1145/212094.212141

[6] R. I. Hartley and A. Zisserman, "Multiple View Geometry in Computer Vision", 2nd ed. Cambridge University Press, 2004.

[7] V. Lepetit, F.Moreno-Noguer, and P.Fua, "Epnp: An accurate o(n) solution to the pnp problem," *International Journal Computer Vision*, vol. 81, no. 2, 2009

[8] M. Lourakis and A. Argyros, "The design and implementation of a generic sparse bundle adjustment software package based on the Levenberg-Marquardt algorithm," Institute of Computer Science - FORTH, Heraklion, Crete, Greece, Tech. Rep. 340, August 2000.

[9] C. Strecha, W. von Hansen, L. J. V. Gool, P. Fua, and U. Thoennessen, “On benchmarking camera calibration and multi-view stereo for high resolution imagery,” in *CVPR’08,* 2008.

[10] N. Snavely, S. M. Seitz, and R. Szeliski, “Photo tourism: exploring photo collections in 3D,” in *SIGGRAPH ’06: ACM SIGGRAPH 2006 Papers*. New York, NY, USA: ACM, 2006, pp. 835–846.

[11] Changchang Wu, “VisualSfM: A visual structure from motion system,” 2011. Available: http://homes.cs.washington.edu/ ccwu/vsfm/

[12] A. Akbarzadeh, J.-M. Frahm, P. Mordohai, B. Clipp, C. Engels, D. Gallup, P. Merrell, M. Phelps, S. Sinha, B. Talton, L. Wang, Q. Yang, H. Stewenius, R. Yang, G. Welch, H. Towles, D. Nister, and M. Pollefeys, “Towards urban 3d reconstruction from video,” in *3D Data Processing, Visualization, and Transmission, Third International Symposium on*, june 2006, pp. 1 –8.

[13] 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,” *International Journal of Computer Vision*, vol. 78, pp. 143–167, 2008, 10.1007/s11263-007-0086-4. Available: http://dx.doi.org/10.1007/s11263-007-0086-4

[14] M. Goesele, N. Snavely, C. Curless, H. Hoppe, and S. M. Seitz, “Multiview stereo for community photo collections,” in *Proceedings of ICCV 2007*, 2007.

[15] 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.” ACM Symposium on User Interface Software and Technology, October 2011. Available: http://research.microsoft.com/apps/pubs/default.aspx?id=155416

[16] J. Quiroga, T. Brox, F. Devernay, and J. Crowley, “Dense semi-rigid scene flow estimation from RGBD images,” in *Computer Vision ECCV 2014*, ser. Lecture Notes in Computer Science, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Springer International Publishing, 2014, vol. 8695, pp. 567–582.

[17] M. Hornacek, A. Fitzgibbon, and C. Rother, “Sphereflow: 6 DoF scene flow from RGB-D pairs,” in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, June 2014, pp. 3526–3533.

[18] F. Steinbrucker, C. Kerl, and D. Cremers, “Large-scale multi-resolution ¨ surface reconstruction from RGB-D sequences,” in ICCV’13, 2013, pp. 3264–3271.

[19] D. Thomas and A. Sugimoto, “A flexible scene representation for 3D reconstruction using an RGB-D camera,” in *Computer Vision (ICCV), 2013 IEEE International Conference on*, Dec 2013, pp. 2800–2807.

[20] C. Raposo, M. Lourenco, M. Antunes, and J. Barreto, “Plane-based odometry using an RGB-D camera,” in British Machine Vision Conference 2013, 2013.

[21] E. Bylow, C. Olsson, and F. Kahl, “Robust camera tracking by combining color and depth measurements,” in *International Conference on Pattern Recognition*, 2014.

[22] P. Moulon, M. Pascal, and R. Marlet. “Adaptive Structure from Motion with A-Contrario Model Estimation.” *Asian Conference on Computer Vision*, 2012.

[23] Z. Zhang “Iterative Point Matching for Registration of Free-Form Curves and Surfaces.” *International Journal of Computer Vision*, 2012.