Real-time multi-sensor RGB-D image stitching and noise removal

Connor Monahan
cmonahan@wustl.edu

Abstract

Low-cost, consumer-oriented depth sensors such as the Microsoft Kinect provide an accessible way to get quality metric depth data, contrasted with more accurate but more expensive LIDAR or time-of-flight based sensors. However, these devices by themselves suffer from problems such as sparse information spaces and limited field of view. This project addresses these problems by applying previous work in hole filling and image stitching to allow for depth maps with wide field of view based on multiple depth sensors in different orientations. Experimental results demonstrate that with some incorporated improvements, smooth depth maps can be generated from multiple source images with minimal tuning.

1 Introduction

A normal color image can be augmented with depth/distance information at every pixel, forming what's known as an RGB-D image where depth is an additional channel. This additional spatial information has been increasingly useful in applications of computer vision, including fields such as biometric authentication, autonomous mobile robotics, and self-driving cars.There exist many methods of computing a depth map from a scene. Structured light sensors, laser scanners, LIDAR systems, time-of-flight sensors, and binocular stereo vision cameras produce this data tailored to a variety of environments at a variety of price points. This paper focuses on the Microsoft Kinect sensor, a speckle imaging device with small field of view oriented at consumer use in close range [5]. Despite its limitations, it remains an attractive device for amateur or small scale applications due to its low cost on the order of $100 during production and $10 on the resale market, especially due to the wide variety of possible indoor applications.

While some limitations cannot be avoided such as the restriction to indoor use, we know that we can overcome limited fields of view in color cameras by applying image stitching algorithms to create a panoramic photo [1], thus it should be possible to improve the Kinect depth map by stitching multiple views together as well. Solving this problem can thus enable the use of cheaper sensors in more complicated vision applications. In this project, I will implement noise filtering and depth map stitching. I will test my implementation using images taken on my Kinect sensor, visually comparing the resulting panorama to the real environment as well as to characteristics of the results shown in [2].

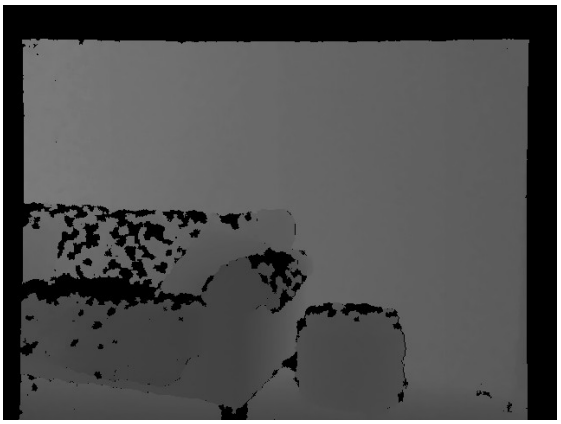

Figure 1: Registered Kinect metric depth map

2 Background & Related Work

The algorithm presented in this paper is an adaptation of the work in [2], which investigated depth map preprocessing and alignment, color image registration, and compositing to solve this problem based on pairs of Kinects. That paper in turn first built upon the work in [4] who worked on noise reduction of Kinect depth maps based on a guided anisotropic diffusion method.

For the second step of stitching together the processed images, they applied the work in [1], which solved the stitching problem in color images by computing features in both images using the SIFT algorithm [3], performing a dual image keypoint matching step using a nearest-neighbor algorithm, and finally stitching all the images together using multi-band blending based on source images in spherical coordi-

nates. To compensate for the issue that SIFT was not designed for depth maps and that the depth information might have fewer features than the color space, they used the registered color images to perform the feature extraction and mapping and then used the resulting homography to transform and blend the depth information.

3 Proposed Approach

In this paper, I apply an adaptation of the algorithm presented in [2] to preprocess and then stitch together depth maps into one panorama. This paper will only consider the task of stitching two images together, but it could be easily extended to stitch n images in the same way as the original paper, forming up to a full 360° panorama. This paper will also only consider concentric cameras, which corresponds with diagram e in Figure 2 in the original paper, which was shown to produce results with the least interference in their testing.

3.1 Single image registration

By default, the Kinect produces an RGB-D image where the depth channel is not aligned with the color image. This is due to the fact that the cameras responsible for each are offset from each other by a few centimeters, creating a stereo effect. In [2], the registration problem was solved by calibrating the depth camera and the color camera, using the resulting information to relate both by a homography and then applying that to map the depth information into the space of the color information. The approach in this paper does not explicitly rely on a manual camera calibration, as all the necessary parameters are calculated at the factory and stored on the sensor. The problem of depth map registration is thus shifted to dataset generation time, and is realized by setting parameters in the device driver provided by OpenKinect. The result of this step is the original color image and the shifted depth image, with larger invalid regions on three sides, as shown in Fig. 1.

3.2 Hole filling

When the Kinect cannot calculate the distance to a location with certainty, it reports a distance of 0 to signal that it should be treated as invalid. This occurs in a variety of circumstances, such as due to multiple reflections, shiny surfaces, occlusion, or interference from other sources of infrared light. Removing this noise is a useful preprocessing step as it will provide a depth map with increased detail.

Previous solutions to this problem involved applying anisotropic diffusion, an edge-preserving smoothing method, to the depth image in either a color image guided or unguided method. During initial research, I was unable to reproduce the results in [2] of applying this method directly to the depth map. The method of anisotropic diffusion, presented in Eq. 1 as

$$I_t = c(x, y, t)\Delta I + \nabla c \nabla I \tag{1}$$

is thus highly dependent on the setting of the parameters c, the conduction coefficients, λ, the diffusion rate in the discrete implementation, and the number of iterations, which were not provided in previous work.

To solve the hole filling problem in this application, I applied a median filter

$$Y[n] = \text{median}_{n'}\, X[n - n'] \tag{2}$$

on each 5×5 neighborhood in the source image. This procedure was repeated for 10 iterations. The resulting image had most small holes filled by this method with a handful of remaining larger holes, which produced acceptable results for this application.

3.3 Paired color image registration

The final goal of the stitching process is to warp the depth maps to lie on the same plane. To achieve this, we need to first find the relative positioning of each of the images with respect to one another. We can define the intrinsic camera projection matrix of the Kinect color camera using the pinhole camera model as

$$K = \begin{bmatrix} f_x & 0 & W/2 \\ 0 & f_y & H/2 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 524 & 0 & 320 \\ 0 & 524 & 240 \\ 0 & 0 & 1 \end{bmatrix} \tag{3}$$

where the field of view in pixels is taken from [5], and I simplify by assuming zero skew. Given images 1 and 2, we want to project image 1 into the plane of image 2. The projection of image 2 is given by

$$I_2 = K_2 R_2 [X, Y, Z]^T \tag{4}$$

where R_2 is the 3×3 identity matrix, and $K_2 = K$ as defined before. The projection of image 1 is similarly given by

$$I_1 = K_1 R_1 [X, Y, Z]^T \tag{5}$$

If we further reduce this problem to pairs of images taken from the same camera under rotation, the justification for which is presented in the Dataset Generation section below, then we can observe that $K_1 = K_2 = K$ and the only unknown parameters

relating I_1 and I_2 is the unknown matrix R_1. We will estimate R_1 by estimating a homography between points in the associated color images.

As in the original paper, the SIFT algorithm is used off-the-shelf to extract features from each of the color images in the form of keypoints and descriptors [3]. The descriptors in both images are then matched using a nearest-neighbors algorithm. These matches are then filtered based on a ratio test as described in [1] Eq. 13, using the provided threshold of 0.7. The keypoints of the remaining matches in both images We then use the RANSAC method to estimate a homography M between the keypoints of the remaining matches from I_1 to I_2, with the maximum allowed inliner pair error set to 5. This allows us to then solve for R_1 in Eq. 5.

3.4 Spherical image projection

As in the original paper, each of the depth maps for images 1 and 2 are then projected into the spherical coordinates given by the equation

$$f * (\sin\theta\cos\phi, \sin\phi, \cos\theta\cos\phi) = (X, Y, Z) \quad (6)$$

to take account for the fact that the images come from a rotation in the same scene, where f is the focal length we used earlier. Image 1 is further warped according to the R_1 we found in the previous step to bring it into the same plane as image 2.

3.5 Image blending

The final step in building the panorama is combining the warped depth images together. [2] used the multi-band blender algorithm for this purpose. However, testing that algorithm in this project lead to poor results, in that the two images would appear to have hard seams near their borders in the resulting panorama. This issue may have been caused by either unknown parameters used in the original study or a preprocessing difference.

For this project, a simpler technique was instead chosen where the two images were averaged at their borders. This was shown to produce an acceptable result as surfaces known to be smooth in the real world also had smooth depth values.

4 Experimental Results

4.1 Dataset generation

All data used to test this process was collected using an Microsoft Kinect V1 in an indoor environment using the OpenKinect Linux drivers. See Fig. 2. To save time and money, only one Kinect V1 sensor was used for this process. Registered RGB-D images were collected from a static scene during a period of a few seconds. The sensor was then rotated around its center to offsets of 15 and 30 degrees counterclockwise from its original position. Natural sources of infrared light were blocked to prevent interference of the sensor by closing all curtains and blinds.

Figure 2: Test apparatus

4.2 Results at 15 degrees

Results of the algorithm were first tested across a rotation of 15 degrees between datasets. This corresponds to an increased depth map field of view of 72° assuming a Kinect field of view of 57°. Original images are shown in Fig. 3 and the stitched depth map is shown in Fig. 4. As can be observed from the figures, the algorithm produced a satisfactory panorama preserving smoothness of different objects.

4.3 Results at 30 degrees

The algorithm was then tested at an increased separation of 30°, corresponding to a total depth map field of view of 87°. Original images are shown in Fig. 5 and the stitched depth map is shown in Fig. 6. Similarly to the figures for the smaller rotation, a smooth depth map was produced without any additional significant issues. This means that a larger field of view can be produced from fewer cameras without any sac-

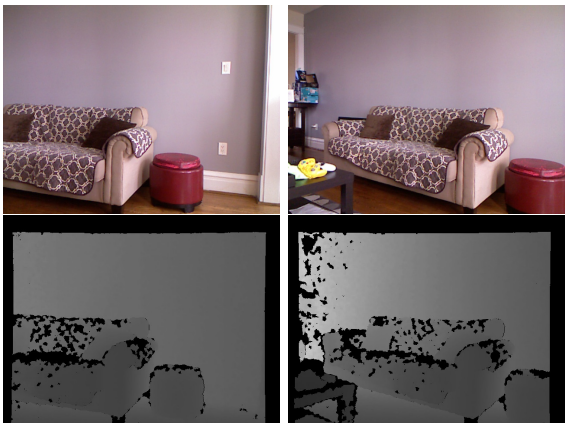

Figure 3: Registered RGB-D images, 15° rotation

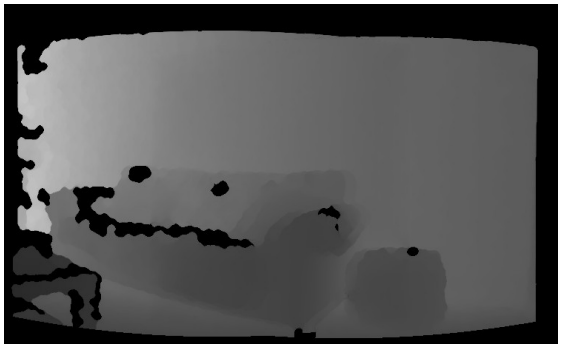

Figure 4: Stitched depth map with hole filling, 15°

Figure 5: Registered RGB-D images, 30° rotation

rifice of accuracy. It is noted however that one of the source depth maps in 5 contains a significant amount of undetectable depth pixels. This error is attributed to distance to the far wall in my apartment (>10m) which falls out of the spec for the Kinect depth sensor.

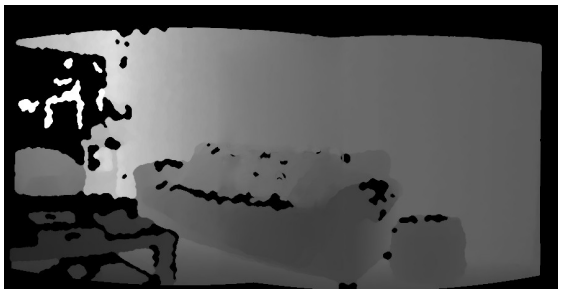

Figure 6: Stitched depth map with hole filling, 30°

5 Conclusion

This paper demonstrates that an array of inexpensive sensors can provide comparable detail to that produced by a single larger sensor such that it preserves all necessary detail from the environment. It also demonstrates how some simplifications to previous methods of depth map stitching can produce comparable results, which may lead to decreased execution time. This algorithm has potential applications to a variety of small indoor spaces, such as robotics for household purposes or those in relatively confined spaces such as grocery store robots, enabling a variety of additional depth-based methods such as localization, mapping, and segmentation.

Further research on this topic might include modifying the preprocessing step to use another type of edge-preserving filter, such as a bilateral filter. It could also include trying a different type of feature extractor, such as convolution layers pretrained on ImageNet, or a feature extractor known to work well on depth maps in order to bypass the entire color image processing and registration step.

Acknowledgments

Thanks to the OpenCV project for the detailed documentation of their library and examples, including the provided implementations for SIFT and nearest neighbors which allowed this project to stay focused. Additionally, thanks to the OpenKinect project for the reliable Kinect for Linux driver and depth registration support.

References

[1] M. Brown and D. G. Lowe. Automatic panoramic image stitching using invariant features. International journal of computer vision, 74(1):59–73, 2007.

[2] H. Li, H. Liu, N. Cao, Y. Peng, S. Xie, J. Luo, and Y. Sun. Real-time rgb-d image stitching us-

ing multiple kinects for improved field of view. International Journal of Advanced Robotic Systems, 14(2):1729881417695560, 2017.

[3] D. G. Lowe. Distinctive image features from scale-invariant keypoints. International journal of computer vision, 60(2):91–110, 2004.

[4] K. R. Vijayanagar, M. Loghman, and J. Kim. Real-time refinement of kinect depth maps using multi-resolution anisotropic diffusion. Mobile Networks and Applications, 19(3):414–425, 2014.

[5] X. Zhou. A study of microsoft kinect calibration. Dept. of Computer Science, George Mason University, Fairfax, 2012.