REAL-TIME ENHANCEMENT OF RGB-D POINT CLOUDS USING PIECEWISE PLANE FITTING

Kazuki Matsumoto, Francois de Sorbier and Hideo Saito

Keio University
Graduate School of Science and Technology
3-14-1 Hiyoshi, Kohoku-ku, Yokohama, Kanagawa, Japan

ABSTRACT

In this paper, we propose an efficient framework for reducing noise and holes in depth map captured with an RGB-D camera. This is performed by applying plane fitting to the groups of points assimilable to planar structures and filtering the curved surface points. We present a new method for finding global planar structures in a 3D scene by combining superpixel segmentation and graph component labeling. The superpixel segmentation is based on not only color information but also depth and normal maps. The labeling process is carried out by considering each normal in given superpixel's clusters. We evaluate the reliability of each plane structure and apply the plane fitting only to true planar surfaces. As a result, our system can reduce the noise of the depth map especially on planar area while preserving curved surfaces. The process is done in real-time thanks to GPGPU acceleration via CUDA architecture.

Index Terms— Plane Fitting, Superpixel, RGB-D camera, GPU, Noise Reduction

1. INTRODUCTION

Due to the development of devices for range data, such as stereo vision cameras, time of flight sensors and structured light 3D scanners, it has become easier to capture videos with high frame rate getting not only the color information, but also the 3-dimensional geometrical data of the real world. These systems are increasingly used in many on-going computer vision research areas including 3D reconstruction, object recognition and augmented reality. Like recent 3D sensors, RGB-D cameras, such as the Microsoft Kinect, have drawn considerable attention among computer vision researchers for their ease of usability and low-cost. However these RGB-D cameras do not have the capability to satisfy the accuracy of the range data required to develop rigorous 3D application because the raw depth data suffer from a significant amount of noise.

One of the main reasons for this problem is that infrared light emitted by RGB-D cameras is affected by specular surfaces of the objects in a scene. Moreover, the artifacts along object boundaries and the structural noise that results from the average noise as well as spontaneous occlusions of the projected pattern to the physical environment also call for the use of hole filling methods, interpolating algorithms and some degree of post-processing to increase the accuracy of the data.

In this paper, we present a framework that reduces the noise and fill the holes in RGB-D data with a piecewise planar fitting approach based on normal adaptive segmentation. As our plane detection is combining normal adaptive superpixel segmentation and graph component labeling, the point clouds are precisely divided into planar surface clusters. Our plane fitting algorithm discriminates between planar surfaces and curved surfaces based on the dependability of estimated local planar surface structure and applies plane fitting to truly planar distributed area. As a result, we can get smooth point clouds retaining the shape of uneven surfaces. This full pipeline can enhance the range data from RGB-D camera while maintaining high frame rate by utilizing highly parallel processing capabilities of modern commodity GPUs. In the following, we will discuss related works in Section 2. After describing the detail of our system in Section 3, Section 4 will show the result of experiments and discussion about them. We finally conclude the paper in Section 5.

2. RELATED WORKS

In order to enhance the depth data captured by an RGB-D camera, several approaches have been proposed and can be divided into two groups. The first one deals with the instability of depth measurements provided by the RGB-D camera over different distances [1, 2]. These approaches use several depth images for reducing variations over each pixel value. Color images captured at the same time than depth images are utilized for filling holes in depth image. But these method can't cope with huge motion of objects in captured scenes.

The second group of methods applies denoising methods on only one pair of depth and color images for reducing structural noise. Among these, Joint Bilateral Filter, a modified version of bilateral filter, is an edge preserving smoothing filter applied on the depth map that adaptively changes

EUVIP 2014, Dec. 10-12, 2014, Paris, France.
978-1-4799-4572-6/14/$31.00 ©2014 IEEE.

the spatial kernel according to the intensity differences from the color image [3]. Another popular method is an optimization for depth map denoising based on Markov Random Field that maximizes a posterior probability of each pixel value [4]. These methods assume that the edges of color images and depth images are highly correlated and both of them are aligned precisely, however depth data near the object boundaries are not reliable and edges in the depth image do not coincide with color image edges because of the drawbacks mentioned before.

Region Growing Bilateral Filter [5] solved this problem by ignoring the depth values around the edges in the corresponding color image and by applying region growing. This method also modifies the joint bilateral filtering to deal with the error of depth values which increase as a quadratic function of distance based on the theory of [6].

Milani et al.[7] proposed a method for depth map interpolation using local plane model. They applied Sobel operators for collecting misalignment between color and range data. Next they interpolated depth values based on coefficients calculated from polynomial regression of pertinent depth pixels. In order to estimate pixels used for interpolation, they divided depth image into segments where the structure of depth map represents a local plane by adopting the segmentation technique from [8] and by discriminating reliable depth values. This approach can apply appropriate interpolation according to local structure so that it can smooth the planar surface and fill the holes in depth map, maintaining the curved surface of complex objects. However, their method is computationally expensive since they employ global graph-cut algorithm for initial segmentation and then use k-means in local segment for post-processing.

Compared with these previous works, we propose a new noise reduction and hole filling algorithm based on locally estimated planar structure in real-time.

3. PROPOSED METHOD

3.1. System Overview

Our method is divided in several steps. We first apply joint bilateral filter to the depth image to reduce the noise while preserving the edge boundaries of objects. Assuming that the camera ' s intrinsic parameters are known, depth data can be converted into a 3D vertex map defined in the camera ' s coordinate space. After that, we calculate the normal map by applying the method proposed by Holzer et al. [9] . Then, we apply normal-adaptive superpixel segmentation, the modified version of depth-adaptive superpixels[10], to divide the 3D point cloud into clusters so that the 3D points in each cluster represent a planar structure. In order to merge similar clusters, graph component labeling is applied to segmented image by comparing the normal of each cluster. Then, we compute normal vector and center point in each cluster to estimate its plane equation. Finally, we project the depth map onto estimated planar surfaces and optimize them based on the reliability of each plane parameter for generating smooth point clouds retaining the shape of uneven surface.

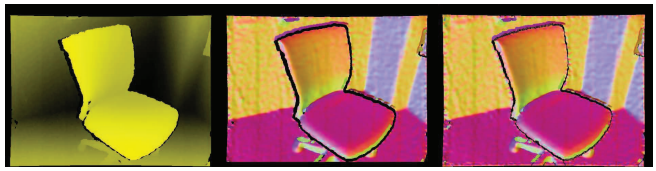

Fig. 1. Result of normal estimation. Left: corresponds to the depth image. Center: normal map estimated with method [9]. Right: normal map calculated from our proposed method.

3.2. Normal Estimation

3.2.1. Joint Bilateral Filter

Joint Bilateral Filter[3] uses not only depth data but also color image of one scene for depth map filtering. The smoothed depth value D_{f_p} at the pixel p is computed as

$$D_{f_p} = \frac{\sum_{q\in\Omega} g_s(p-q)g_c(C_p - C_q)g_d(D_p - D_q)D_q}{\sum_{q\in\Omega} g_s(p-q)g_c(C_p - C_q)g_d(D_p - D_q)} \quad (1)$$

where Ω is the neighborhood of p. g_s, g_c, g_d are Gaussian functions controlled by the standard deviation parameters σ_s, σ_c, σ_d respectively. $p-q$ represents the spatial distance, $C_p - C_q$ is color similarity and $D_p - D_q$ is the depth similarity.

The smoothed depth map is then back-projected into the 3D space using the intrinsic parameters of RGB-D camera.

3.2.2. Real-Time Normal Estimation

After back-projection, we apply the normal estimation technique from [9] on the 3D points for calculating the normal map. This method uses integral images to speed-up the computation of the normal for a specific point from the covariance matrix of its local neighborhood. It then generate a smooth and accurate normal map at a high frame rate. However, this method cannot estimate normals in the pixels around the object boundaries if there is a large difference between the depth value and its neighborhood. Therefore, we search for two close points around these invalid pixel vertices V_p and if we can find those close points $V(p_{n_1})$ and $V(p_{n_2})$, we estimate the normal $n(p)$ as follows.

$$n(p) = (V(p_{n_1}) - V(p)) \times (V(p_{n_2}) - V(p)) \quad (2)$$

In that case, after normalization, we obtain a normal map as shown in figure 1. As it can be seen, a smooth normal map is generated without omitting normal information on object boundaries.

3.3. Normal Adaptive Segmentation

3.3.1. Normal Adaptive Superpixels

Weikersdorfer et al.[10] proposed a novel oversegmentation technique for RGB-D images so that the 3D geometry surface is partitioned into uniformly distributed and equally sized planar patches. Firstly, this method samples points while guaranteeing the blue-noise spectrum property[11] so that the density of the points around sampled points can be equally distributed. Then, a clustering algorithm assigns points to superpixels and improves their centers using iterative k-means algorithms with a distance calculated from the color distance, the depth value and the direction of normal vector. We also apply similar technique as depth-adaptive segmentation for getting granular planar regions in real-time with RGB-D data and the normal map calculated by our method in 3.2.2 as illustrated in Algorithm 1

Algorithm 1 Normal Adaptive Superpixels

for all p_k in all 3D points at each cluster **do in parallel**
  Move its center C_k to adjacent positions with lowest normal gradient position
end for
for all p_i in all 3D points **do in parallel**
  label $l_i \leftarrow -1$
  distance $dist_i \leftarrow \infty$
end for
for all C_k **do in parallel**
  for all p_i in 2S x 2S region around C_k **do in parallel**
    calculate distance $dist_k(p_i)$ between C_k and p_i
    if $dist_k(p_i) < dist_i$ **then**
      $dist_i \leftarrow dist_k(p_i)$
      $l_i \leftarrow k$
    end if
  end for
end for

where the center C_k is defined as the average of the 3D points in the cluster k. This algorithm is based on gSLIC, real-time implementation of Simple Linear Iterative Clustering (SLIC), proposed by [12]. The distance $dist_k(p_i)$ is calculated as follows

$$dist_k(p_i) = \frac{\sum_j w_j dist_{k_j}(p_i)}{\sum_j w_j} \tag{3}$$

with the subscript j consecutively representing the spatial(s), color(c), depth(d) and normal(n) terms. w_s, w_c, w_d and w_n are empirically defined weights of spatial, color, depth and normal distances, respectively represented as $dist_{k_s}(p_i)$, $dist_{k_c}(p_i)$, $dist_{k_d}(p_i)$ and $dist_{k_n}(p_i)$.

3.3.2. Representation of Planar Structure

The result of the normal adaptive segmentation gives for each cluster its center $C_k(X_c, Y_c, Z_c)$ and its representative normal $n_k(a, b, c)$. Considering a cluster as a locally planar surface, each of its point $V_{k_p}(X_{k_p}, Y_{k_p}, Z_{k_p})$ can be defined as follows

$$aX_{k_p} + bY_{k_p} + cZ_{k_p} = d_k \tag{4}$$

where d_k is the distance between the plane and the origin. Considering that C_k is located on the planar surface, we compute d_k as follows.

$$d_k = aX_c + bY_c + cZ_c \tag{5}$$

3.4. Merging Superpixels

The result of normal adaptive superpixels is illustrated in figure 2, where the scene is divided into homogeneous regions in terms of color, depth and normal vector. Due to the oversegmentaion procedure, post-processing is required to find the global planar structures. Therefore, we discriminate whether cluster l is in the same planar structure than adjacent clusters k or not, by calculating θ_{kl} and d_{kl} as follows

$$d_{kl} = |d_k - d_l| \tag{6}$$

$$\theta_{kl} = \arccos(n_k \times n_l) \tag{7}$$

and if $d_{kl} < \alpha$ and $\theta_{kl} < \beta$, cluster l is merged into cluster k via the graph component labeling proposed by Hawick et al. [13]. They present several algorithms for performing graph component labelling with GPUs and CUDA. Among their proposed algorithms, we chose the "Equivalence list" algorithm because it does not require a large equivalence matrix which uses too much memory. This results in a chain of pixel labels similar to the conventional linked list structure, but this chain is quickly refined thanks to parallel processing.

Finally, the center and the representative normal vector of each cluster are calculated again by taking the average of normals and 3D coordinates of the points in each cluster.

Fig. 2. Result of normal adaptive segmentation and connected component labeling. Left: corresponding RGB image. Center: Normal Adaptive Superpixels. Right: Result of connected component labeling.

3.5. Plane Based Projection and Optimization

By using the equation(4), the relationship between normalized image coordinates $u_n(x_n, y_n)$ and the corresponding 3D coordinates $V_{k_p}(X_{k_p}, Y_{k_p}, Z_{k_p})$ is represented as follows

$$Z_{k_p} = \frac{d_k}{ax_n + by_n + c}, X_{k_p} = x_n Z_{k_p}, Y_{k_p} = y_n Z_{k_p} \quad (8)$$

Since some clusters may not necessarily contain a planar structure, we detect which are planar ones by evaluating the reliability of the plane model calculated during the previous step. By using the 3D point V_{f_p} computed from the Joint Bilateral Filter in section 3.2.1 and the variance of normal vectors ψ_k obtained in section 3.4, we can get the optimized point V_{o_p} as follows

$$V_{o_p} = \begin{cases} V_{f_p} & (|V_{f_p} - V_{k_p}| > \gamma V_{k_p} \textbf{ or } \psi_k > \delta) \\ V_{k_p} \cos\psi_k + V_{f_p}(1.0 - \cos\psi_k) & (\textbf{otherwise}) \end{cases} \quad (9)$$

where γ and δ are the adaptively changing threshold specifically chosen for a given scene for rejecting unreliable plane model. After this optimization, we apply ordinary bilateral filter to V_{o_p} for smoothing the artifacts around boundaries.

4. EXPERIMENTS

In order to evaluate the performance of our method, we applied it on two different scenes and compared our result(**PROPOSED**) with previous works, Joint Bilateral Filter(**JBF**), Markov Random Field(**MRF**) and Region Growing Bilateral Filter(**RGBF**) in terms of runtime and accuracy. Moreover, we calculated an average depth data by accumulating 1,000 frames of data and taking the average for comparing our method with frame accumulation. To evaluate the accuracy of our method, we generate the groundtruth depth data with a scene generated via OpenGL and obtained depth data by adding noise to the groundtruth data based on the RGB-D camera noise model from [6]. Table 3 illustrates the root-mean-square-error(RMSE) between ground-truth depth 3D points and processed depth 3D points. Our full pipeline is implemented on a system equipped with Intel Core i7-4770K, NVIDIA GeForce GTX 780, and 16.0GB of memory. We used OpenCV for trivial visualizations of color and depth images as well as data manipulations, and PointCloudLibrary for 3-dimensional visualization. Microsoft Kinect was accessed via Prime Sense drivers. All GPGPU implementations were done with CUDA version 5.0.

4.1. Discussion

Table 1 shows the parameters for each experiment. For instance, we adjust the number of clusters for the superpixel segmentation so that we can obtain a smooth point clouds. If the number of clusters is too big, the clusters will start to suffer from the noise of point clouds or if it is too small, we will loose geometrical details. As shown in figure 4, our system can reproduce smooth depth data from bumpy raw depth data from Kinect(**INPUT**), especially in planar structure area. **MRF**, **JBF** and **RGBF** suffer from noisy data because these methods estimate a pixel depth value from its neighborhood. However, remodeling based on the planar structures is independent of the input depth readings once the planar model has been found, we can completely replace the inevitably discretized representations of smooth flat surface with true flat representation as shown in figures 5 and 8. Even in a scene which contains curved surface objects, we can get smooth 3D points while preserving complex structure of input data as shown in figure 7 and figure 8, since our system checks the reliability of each plane structure and applies optimization as we discussed in section 3.5. Moreover, as our algorithm can interpolate the depth map according to the plane structures, our method can fill the holes of the point clouds. If the size of the hole is over the filter size, **MRF**, **JBF** and **RGBF** can't cope with the holes and **AVERAGE** can't fill the hole which is observed frequently in several frames.

Apart from visual assessments, we also conducted numerical analysis of the results. Table 2 shows that our method is slower than **MRF** and **JBF** but it can still work in real-time because of parallel computation capability of GPU. Table 3 shows results of our pipeline close from the ground-truth because our method totally replaces the input points in plane areas with plane fitted points and applies joint bilateral filter to curved surface area. Since other methods use the original pixel value for estimating the filtered pixel value, the estimated value is contaminated with noise of the data. As figure 9 shows, the proposed method can reduce the noise of input point clouds and produce smooth point clouds especially in planar area.

Table 2. Runtime

Method	**Corner** (msec)	**Complex** (msec)
JBF	3.0	3.0
MRF	5.0	5.0
RGBF	45.0	45.0
PROPOSED	21.0	21.0

5. CONCLUSIONS

In this paper, we proposed an efficient pipeline that reduces noise and interpolates depth values by fitting the points to estimated planar structures. In order to detect planar structures, we combined normal adaptive superpixels and graph component labeling by using color image, depth data and normal map simultaneously. As a result, we can produce smooth depth map form bumpy and noisy raw depth data while pre-

Table 1. parameters for experiment

Method	Parameters	**Corner**	**Complex**	**Accuracy Evaluation**
Joint Bilateral Filter	$\sigma_s, \sigma_c, \sigma_d$	70, 50 , 20	70, 50 , 20	70, 50 , 20
Superpixel Segmentation	w_s, w_c, w_d, w_n	100, 30 , 50, 150	80, 30 , 70, 150	10, 50 , 50, 150
	$iteration, clusters$	1, 300	1, 300	1, 300
Merging Superpixels	α, β	$140mm, \pi/8$	$30mm, \pi/12$	$150mm, \pi/8$
Optimization	γ, δ	$0.1, \pi/8$	$0.01, \pi/8$	$0.01, \pi/8$

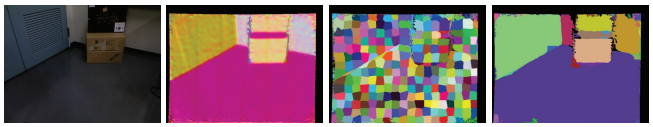

Fig. 3. RGB: normals: superpixels: merging superpixels

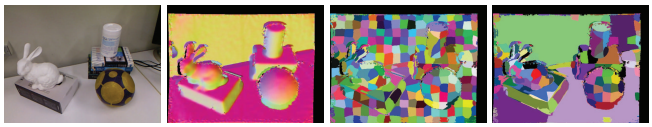

Fig. 6. RGB: normals: superpixels: merging superpixels

Fig. 4. **Corner**

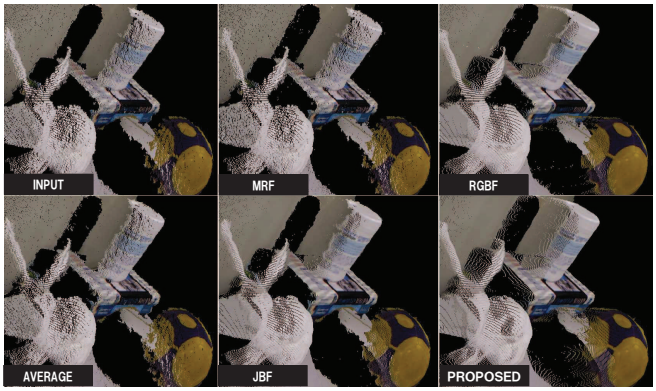

Fig. 7. **Complex**

Fig. 5. **Corner(zoom)**

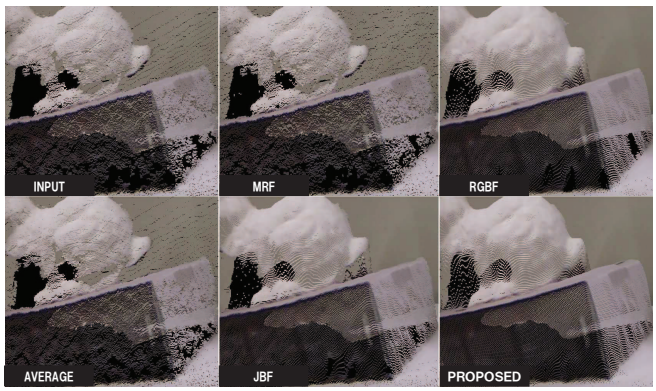

Fig. 8. **Complex(zoom)**

serving curved objects surfaces compared with other state-of-the-art algorithms. Since the whole procedure can be parallelized, our system is implemented in GPU and maintains high frame rate.

Our system can be further enhanced by using new methods. For example, using superpixel segmentation and labeling for detecting planar area sometimes fails to merge small plane clusters into the large plane which these clusters locate. This

can be avoided by merging these small clusters into adjacent large cluster after the labeling. Another possibility is to use curved surface fitting to non-planar area so that our method can not only reduce noise but fill holes in curved surface areas.

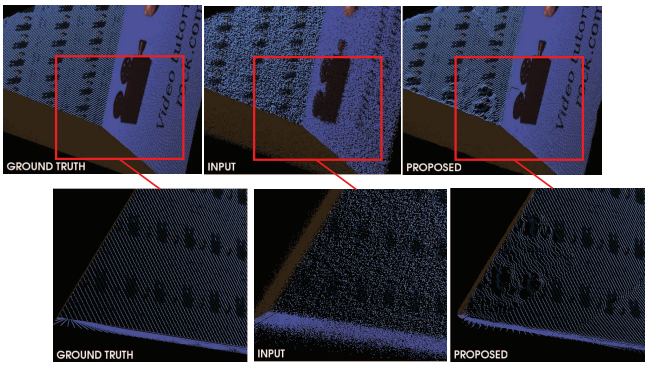

Fig. 9. Result of accuracy evaluation. left : Ground Truth from OpenGL. center : point cloud with noise. right : proposed method.

Table 3. Accuracy

Method	RMSE (mm)
point cloud with noise	38.93
JBF	12.16
MRF	18.04
RGBF	12.09
PROPOSED	10.98

ACKNOWLEDGEMENTS

This work is partially supported by National Institute of Information and Communications Technology (NICT), Japan.

6. REFERENCES

[1] Sergey Matyunin, Dmitriy Vatolin, Yury Berdnikov, and Michail Smirnov, "Temporal filtering for depth maps generated by kinect depth camera," in *3DTV Conference: The True Vision-Capture, Transmission and Display of 3D Video (3DTV-CON), 2011*. IEEE, 2011, pp. 1–4.

[2] Massimo Camplani and Luis Salgado, "Efficient spatio-temporal hole filling strategy for kinect depth maps," in *Proceedings of SPIE*, 2012, vol. 8920.

[3] Georg Petschnigg, Richard Szeliski, Maneesh Agrawala, Michael Cohen, Hugues Hoppe, and Kentaro Toyama, "Digital photography with flash and no-flash image pairs," in *ACM transactions on graphics (TOG)*. ACM, 2004, vol. 23, pp. 664–672.

[4] James Diebel and Sebastian Thrun, "An application of markov random fields to range sensing," in *NIPS*, 2005, vol. 5, pp. 291–298.

[5] Li Chen, Hui Lin, and Shutao Li, "Depth image enhancement for kinect using region growing and bilateral filter," in *2012 21st International Conference on Pattern Recognition (ICPR)*. IEEE, 2012, pp. 3070–3073.

[6] Kourosh Khoshelham and Sander Oude Elberink, "Accuracy and resolution of kinect depth data for indoor mapping applications," *Sensors*, vol. 12, no. 2, pp. 1437–1454, 2012.

[7] Simone Milani and Giancarlo Calvagno, "Joint denoising and interpolation of depth maps for ms kinect sensors," in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2012, pp. 797–800.

[8] Pedro F Felzenszwalb and Daniel P Huttenlocher, "Efficient graph-based image segmentation," *International Journal of Computer Vision*, vol. 59, no. 2, pp. 167–181, 2004.

[9] Stefan Holzer, Radu Bogdan Rusu, M Dixon, Suat Gedikli, and Nassir Navab, "Adaptive neighborhood selection for real-time surface normal estimation from organized point cloud data using integral images," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2012, pp. 2684–2689.

[10] David Weikersdorfer, David Gossow, and Michael Beetz, "Depth-adaptive superpixels," in *2012 21st International Conference on Pattern Recognition (ICPR)*. IEEE, 2012, pp. 2087–2090.

[11] Raanan Fattal, "Blue-noise point sampling using kernel density model," in *ACM Transactions on Graphics (TOG)*. ACM, 2011, vol. 30, p. 48.

[12] Carl Yuheng Ren and Ian Reid, "gslic: a real-time implementation of slic superpixel segmentation," *University of Oxford, Department of Engineering, Technical Report*, 2011.

[13] Kenneth A Hawick, Arno Leist, and Daniel P Playne, "Parallel graph component labelling with gpus and cuda," *Parallel Computing*, vol. 36, no. 12, pp. 655–678, 2010.