High-Definition 3D Reconstruction in Real-Time from a Moving Depth Sensor

Xu Hu, Yuanyuan Yu and Zhaozhong Wang
Image Processing Center, Beihang University, Beijing 100191, China
{doubt0325, quanquan}@sa.buaa.edu.cn, zwang@buaa.edu.cn

Abstract - This paper proposes a method to reconstruct 3D point clouds of a static scene in real-time by a moving depth sensor. Based on a base-line 3D reconstruction algorithm which fuses depth maps from the moving camera, a depth map enhancement module is embedded in the depth fusion stage to improve the level of details of reconstructed 3D models. Depth enhancements using the Sobel operator and the Laplacian operator are applied for speed and quality considerations. Experiments for real-time reconstruction of 3D scenes, especially the reconstruction of 3D human faces are provided to validate the proposed method. The reconstruction results are inspiring in their high definition compared with the original base-line algorithm.

Index Terms - 3D reconstruction, Real-time reconstruction, Depth enhancement, High-definition, Depth sensor.

I . Introduction

High-quality 3D polygonal models reconstructed and rendered from a real world scene are required in different applications, like video-games, movies, virtual reality, ecommerce and other graphics applications. However the generation of a 3D model with precise surface from unorganized point clouds derived from laser scanner data [5] or photogrammetric image measurements [4,6] is a difficult problem. For the reconstruction of a person's face more level of details and capturing time are required to get data, but it feels uncomfortable for people to keep a pose too long; small movements are inevitable.

A number of methods have been developed for 3D reconstructions using data from different deceives. There are methods based on triangulations, such as using laser light [9], structured light [12], coded light [13], and image measurements [6, 7, 8]. There are also methods do not require triangulations and directly estimate surface normal vectors, such as shape form texture [10] and shape from 2D edge gradients [11].

The methods based on 3D active sensors (mainly laser scanners) provide the highest quality of reconstruction at present. However, the 3D active sensors are quite expensive, which are mainly for industry applications. Passive image-based methods using projective geometry [6] are very portable and the sensors are not expensive. But the accuracy of result is low and it cannot satisfy real-time applications.

The Kinect Fusion system [1] developed by Microsoft Research uses low-cost depth sensor (the Kinect) and commodity graphics hardware for accurate and real-time 3D reconstruction. Depth data from Kinect is used to track the 3D pose of the sensor and reconstruct 3D models of the physical scene [2]. The advantage of Kinect Fusion is its speed for permitting direct feedback and user interaction. But it is still required to improve the resolution of 3D reconstruction in Kinect Fusion since the hardware performance is generally limited.

In this paper, we shall use a depth map enhancement module to improve the quality of the base-line algorithm of Kinect Fusion. We shall apply image enhancement approaches for depth map enhancement. We implement the 3D reconstruction of face models in real-time and achieved inspiring results. Compared with the original implementation of Kinect Fusion, the level of details of 3D reconstruction can be improved.

In Section 2 we introduce the 3D reconstruction system. Section 3 gives more detailed description of processing steps. Section 4 proposes the improvement of the algorithm using depth enhancement. Section 5 gives experimental results, then we draw the conclusion.

II. System Overview

A. Prerequisite Hardware and Library

To implement the algorithm, there are some prerequisite hardware and libraries. A depth sensor (e.g. the Kinect) and a GPU-enabled computer are required. A typical low-cost depth sensor like the Kinect is equipped with an infrared projector and an infrared receiver which provides a depth map of the scene (640x480 pixels at 30 FPS). Internally, the Kinect has a processor for data acquisition, manipulation and transmission [17]. The GPU processor is required to be of high power to achieve a full rate of reconstruction at 30 frames per second.

In addition, the point cloud library (PCL) is used to support our algorithm. The PCL is a stand-alone, large scale, open project for 2D/3D image and point cloud processing. The library can be effectively applied to work at a high level with point clouds. The PCL is divided in several separate libraries, each with its own focus, such as the filters library allowing developer to perform filtering, the feature library to extract feature.

B. Overview of the 3D Reconstruction

An overview of the 3D reconstruction system is given in Figure 1. We grab depth maps continuously from a depth camera. Then in the pre-processing step, the depth map is filtered to compute a vertex map and a normal map. Next by aligning the vertex map and normal map with those from the ray casting we can track the camera pose. After get the 6

© 2013. The authors - Published by Atlantis Press

degree of freedom of camera pose, the raw depth map is enhanced and computed to generate a truncated signed distance function (TSDF), which is fused from the depth data with the previous model. At last, ray casting algorithm renders the surface of model through the model's TSDFs and generate a current vertex map and normal map for the alignment of next frame.

Most of the components in the system is similar to the Kinect Fusion method [1, 2]. The key novelty is the depth map enhancement module, which makes our system to generate 3D reconstruction with higher definition.

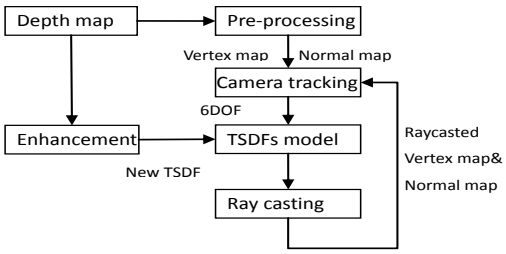

Fig. 1 Flowchart of the real-time 3D reconstruction.

III . Main Processing Steps

A. Obtain the depth data

The input is a temporal sequence of depth maps and no color information. The depth sensor computes the depth maps by emitting a non-uniform infrared pattern on the scene through its infrared projector and then acquiring the reflected pattern through an infrared receiver[17]. So lighting condition does ·not interfere with the captured depth data and the algorithm can work in complete darkness environment.

B. Pre-Processing of the Depth Data

In the pre-processing step, in order to smooth the depth data and remove the noise, both the neighbour average of values and the distance from the target pixel to its neighbours are taken into condition. Using an intrinsic calibration matrix K of the sensor, the filtered depth data is projected in the camera's coordinate space as a vertex map. For each pixel $u = (x, y)$, we have

$$v_i(u) = D_i(u)K^{-1}[u, 1] \quad (1)$$

where $D_i(u)$ is the depth map and $v_i(u)$ is the vertex map. Then using each vertex of the vertex map, a normal vector map is computed from neighbouring re-projected points:

$$N_i(u) = ((v_i(x+1, y) - v_i(x, y)) \times (v_i(x, y+1) - v_i(x, y)) \quad (2)$$

where $N_i(u)$ is the normal map. To speed up the algorithm, both vertex map and normal map are computed in parallel in a three-level multi-resolution pyramid on the GPU.

C. Camera Tracking

Camera tracking needs to estimate a single transform with 6 degrees of freedom (DOF); the transform aligns points of latest frame with those of previous frame. The iterative closed point (ICP) algorithm is used for 3D shape alignment, before this an important step is to find correspondences between two data maps. In this system, the projective data association [2] is used to find the correspondences. Then the global camera pose T_i is initialized with previous pose T_{i-1} and is updated with an incremental transform calculate per iteration of ICP [1] until errors between the two points cloud reaches minimum. If $D_i(u)$ is depth data, then

$$v(globe) = T_i^{-1} v_i(u) \quad (3)$$

where *globe* means global coordinates. From the vertex map $v_i(u)$ we can compute $N_i(u)$ and global normal

$$n(globe) = T_i^{-1} N_i(u) \quad (4)$$

if

$$\left\| v(globe) - v_{i-1} \right\| < distancethreshold \quad (5)$$

and

$$abs(n(globe) \bullet n_{i-1}) < normalthreshold \quad (6)$$

Then the point correspondence can be established.

D. Fuse New Depth Data with Previous Model

Once the alignment transform is found, the new depth data can be merged with the current model. A truncated signed distance function (TSDF) [1] is used for this step. This function extracts the surface and assigns negative numbers to voxels that we have not measured, positive numbers to those in front of the surface [2]. Each new depth map can create a TSDF, then through a running average we merge the new TSDF to the current model's TSDF:

$$TSDF(avg) = \frac{TSDF_{i-1} + TSDF_i}{2} \quad (7)$$

where $TSDF_{i-1}$ is the previous model's TSDF and $TSDF_i$ is the new depth map's TSDF.

E. Add Enhancement Module in the Fusion

In the original fusion step of Kinect Fusion, the raw depth data is used to compute a TSDF. As the depth map obtained by depth sensor is typically with low resolution, it is necessary to add an enhancement module to deal with raw depth data before merging the new TSDF, which can improve the quality of the result of 3D reconstruction. Detailed description of the enhancement module will be given in Section 4. The merged model's TSDF will be used to generate new vertex map and normal map for the camera tracking in the later frame.

F. Generate the 3D Model with Surface

To render the model, a GPU-based ray casting [1] is implemented to generate views of the implicit surface within the globe volume. It is also used for tracking. A ray is a

traveling line, it starts from the view point and travels in a direction until intersect the surface. Then the pixel it corresponds to in the image space of the camera is assigned to the 3D intersection point.

IV . The Depth Enhancement Module

This section focuses on the key improvement to the base-line algorithm. We add a filter module in the depth fusion step to enhance the raw depth map. Then a new TSDF is computed from the filtered data, which is used for merging current depth data with previous model in the camera's coordinate space. In the added filter module, we apply approaches of image enhancement for depth map enhancement. The results of the 3D reconstruction depend on the filter used.

First we review some image enhancement approaches. There are a number of image filters for enhancement purposes. For example, the classical Lanczos filter [14] is the windowed form of sinc filter. It's the "best compromise in terms of reduction of aliasing, sharpness, and minimal ringing", and superior to many other filters in maintaining local contrast. The novel *warp resize* filter [15] shows good performance in real-time video enhancement. Bilateral filter [16] smoothes image while preserving edges, by means of a nonlinear combination of nearby pixel value with the cost of computational load. These filters may be applicable for depth enhancement, but their computational loads may affect the running speed of the entire reconstruction system.

In our system, we test the depth enhancement based on the Sobel and Laplacian operators to improve the levels of details of 3D reconstruction. These operators are computational efficient. The Sobel operator is widely used in image processing for edge detection purpose. The Laplacian operator is a 2D isotropic measure of the 2nd spatial derivative of an image. The Laplacian of an image highlights regions of rapid intensity change and is therefore also used for edge detection. The two operators are as follows:

$$A=\begin{bmatrix}1 & 1 & 0\\ 1 & 0 & -1\\ 0 & -1 & -1\end{bmatrix},\qquad B=\begin{bmatrix}\frac{1}{6} & \frac{2}{3} & \frac{1}{6}\\ \frac{2}{3} & \frac{-10}{3} & \frac{2}{3}\\ \frac{1}{6} & \frac{2}{3} & \frac{1}{6}\end{bmatrix}$$

where A is the Sobel operator, and B is the Laplacian operator. The depth map can be enhanced by adding the convoluted depth with the above operators on the depth map itself. Let $D_i(x,y)$ be the raw depth map, the enhanced depth map can be expressed as

$$D_i'(x,y)=D_i(x,y)+\sum\sum D_i(m,n)H(x-m,y-n) \qquad (8)$$

where H is the cycle extension of the Sobel kernel A or Laplacian kernel B . The enhanced depth map $D_i'(u)$ maintains all the raw information and improves the level of details at depth edges. Applying filter operator on GPU we can reconstruct 3D models in real-time.

V . Experiments

In this section we perform experiments to validate the algorithm. First we demonstrate the effect of Sobel and Laplacian operators on depth map. In Figure 2 and 3 we have two depth maps which are filtered by the Sobel and Laplacian, respectively. We can see that the depth edge are highlighted. By adding these filtered depths to the original depth maps, the reconstruction results can be enhanced.

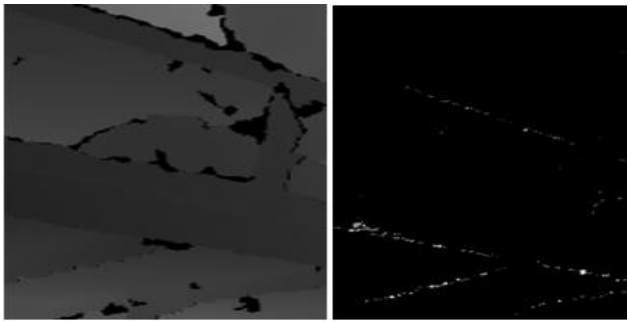

Fig. 2 Applying the Sobel operator to the depth map. Left: the original depth map. Right: the Sobel filtered depth map.

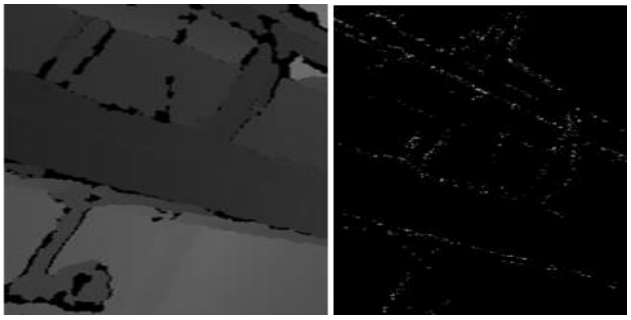

Fig. 3 Applying the Laplacian to the depth map. Left: the original depth map. Right: the Laplacian filtered depth map.

In the original base-line algorithm of Kinect Fusion, the new grabbed raw depth map is directly computed for a TSDF. As for the reconstruction of 3D scene with a low power GPU, it's a problem to discriminate detailed 3D scene, see the left-most images in Figure 4 and 5 for illustration. In these images, the details of faces are not clear since the edge of the eyes and mouths are blurred.

In order to improve the levels of details, we adopt the proposed approach to enhance the edge of the models. First, we apply the Sobel operator to depth data, as shown in the middle of Figure 4. Sobel operator enhances the edge of the mouth, while it increases the glitches around the edge of 3D model. Then we changed the Sobel to Laplacian for depth enhancement. We can see from Figure 5 that, the edge of glasses has been distinguished from the model of face. Similar to the Sobel operator, the Laplacian also increase the glitches around the mode edges. Compared with the Sobel, the Laplacian has less glitches around edges and is faster in computation.

Though both of them slightly slow down the entire algorithm, the reconstruction is still in real-time. Table 1 provides a comparison of running speeds using different operators; here we show the data of relative speed, as the absolute speed depends on the performance of GPU and on the operation method of depth sensor during data capturing. Figure 6 provides more results of 3D reconstruction using the proposed algorithm.

Fig. 4 Comparison of experimental results. Left: the original result of Kinect Fusion. Middle: the enhancement based on the Sobel operator. Right: an enlarged part at mouth of the middle image.

Fig. 5 Comparison of experimental results. Left: the original result of Kinect Fusion. Middle: the enhancement based on the Laplacian and minifying the size of volume, Right: an enlarged part at eye of the middle image.

Fig. 6 More 3D reconstruction results using the proposed algorithm.

TABLE 1 The Relative Running Speed of Different Depth Enhancment Methods.

Filter module	Non	Sobel operator	Laplacian operator
Relative speed	100%	95.4%	97.3%

VI . Conclusion

We implement a high - definition 3D reconstruction algorithm based on the base-line algorithm of Kinect Fusion. We add a depth enhancement module in the depth fusion step. Compared with the non-enhanced results, the details of 3D models are improved. In the future we shall apply some special filters designed for depth maps to improve the definition further.

Acknowledgment

This work was supported by the National Natural Science Foundation of China under Grant 60803071.

References

[1] R. A. Newcombe, S. Izadi, O. Hilliges, et al., “Kinect Fusion: Real-Time Dense Surface Mapping and Tracking,” 10th IEEE International Symposium on Mixed and Augmented Reality (ISMAR), 2011 pages:127 -136.

[2] S. Izadi, D. Kim, O. Hilliges, et al., “Kinect Fusion: Real-time 3D reconstruction and interaction using a moving depth camera,”, Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology , 2011, Page 559-568

[3] R. Fabio, “From Point Cloud To Surface: The Modeling And Visualization Problem,”, International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, 2003, Vol. XXXIV-5/W10.

[4] R. Mv, G. Somanath, D. Norris, J. Gutierrez, and C. Kambhamettu, “A camera flash projector-based reconstruction system for digital preservation of artifacts,” Journal on Computing and Cultural Heritage (JOCCH), March 2013, Volume 6, Issue 1.

[5] R. Mencl,, “Reconstruction of Surfaces from Unorganized 3D Points Clouds,” 2001, PhD Thesis, Dortmund University, Germany.

[6] Remondino, “3D Reconstruction of Static Human Body with a Digital Camera,” Videometrics VII, SPIE Proc., 2003, Vol. 5013, pp. 38-45.

[7] J. Frahm et al. “Building Rome on a cloudless day,” In Proc. Europ. Conf. on Computer Vision (ECCV). 2010.

[8] R. A. Newcombe, S. Lovegrove, and A. J. Davison, “DTAM: Dense tracking and mapping in real-time,”In Proc. of the Int. Conf. on Computer Vision (ICCV), 2011.

[9] V. Sequeira, K. Ng, et al. “Automated reconstruction of 3D models from real environments,” ISPRS Journal for Photogrammetry and Remote Sensing, 1999, 54(1), pp. 1-22.

[10] J. R. Kender, “Shape from Texture,” Proc. DARPA IU Workshop, 1978.

[11] S. Winkelbach, F. M. Wahl, “Shape from 2D Edge Gradient,” Pattern Recognition, Lecture Notes in Computer Science 2191, Springer, 2001.

[12] R. Sablatnig, C. Menard, “3D Reconstruction of Archaeological Pottery using Profile Primitives,” Proc. of International Workshop on Synthetic-Natural Hybrid Coding and Three-Dimensional Imaging, 1997, pp. 93-96.

[13] F. M. Wahl, “A Coded Light Approach for 3-Dimensional Vision,” IBM Research Report, RZ 1452, 1984.

[14] “Lanczos resampling,” virtualdub.org, July 2012.

[15] “Introducing ‘warp resize’”, virtualdub.org, Dec 2005.

[16] C. Tomasi, “Bilateral filtering for gray and color images,” 1998, International Conference on Computer Vision.

[17] Zhang Z, “Microsoft kinect sensor and its effect,” Multimedia, IEEE, 2012, 19(2): 4-10.