Kinect Quality Enhancement for Triangular Mesh Reconstruction with a Medical Image Application

A. Khongma, M. Ruchanurucks, T. Koanantakool, T. Phatrapornnant, Y. Koike and P. Rakprayoon

Abstract This chapter presents a method to estimate proportion between skin burn area and body surface using computer vision techniques. The data are derived using Microsoft Kinect. We first show a comprehensive Kinect calibration method. Then the color image is segmented into 3 sections, background, burn area, and body area. The segmentation method developed is based on watershed algorithm and Chebyshev's inequality. The segmented color image is to be mapped with the depth image to generate triangular mesh. We discover that reconstructing the 3D mesh, using marching cube algorithm, directly from Kinect depth information is erroneous. Hence, we propose to filter the depth image first. After the enhanced mesh is derived, the proportion between 3D meshes of burn area and 3D meshes of body surface can be found using Heron's formula. Finally, our paradigm is tested on real burn patients.

Keywords Microsoft kinect · Burn patient · Triangular mesh reconstruction

A. Khongma (✉)
TAIST Tokyo Tech, ICTES Program, Electrical Engineering,
Kasetsart University, Bangkok, Thailand
e-mail: tourain_p@hotmail.com

M. Ruchanurucks · P. Rakprayoon
Kasetsart Signal and Image Processing Laboratory, Electrical Engineering,
Kasetsart University, Bangkok, Thailand
e-mail: fengmtr@ku.ac.th

T. Koanantakool
Chest Disease Institute, Nonthaburi, Thailand
e-mail: tawetong@gmail.com

T. Phatrapornnant
National Electronics and Computer Technology Center (NECTEC), Bangkok, Thailand
e-mail: teera.phatrapornnant@nectec.or.th

Y. Koike
Tokyo Institute of Technology, Yokohama, Japan
e-mail: koike@pi.titech.ac.jp

S. Patnaik and B. Zhong (eds.), *Soft Computing Techniques in Engineering Applications*, Studies in Computational Intelligence 543,
DOI: 10.1007/978-3-319-04693-8_2,

1 Introduction

Currently, three-dimensional technology is widely used and is a popular topic among developers working on computer vision/3D graphics applications. Such technology can be used in conjunction with several applications as follows:

- Surveillance systems use it to detect people, for object indentifying, etc.
- Robotic applications utilize it for advanced robots such as SLAM, mobile robot, etc.
- Entertainment facilities apply it to develop graphic of an animation or for video-gaming and three-dimensional film.
- Medical fields use it for diagnosis, surgery, and rehabilitation. The 3D scheme provides some advantage over traditional 2D image processing.

For input of 3D-technology, one conventional method is using stereo cameras. The principle of method is a trigonometry technique. This allows the camera to simulate human binocular vision, and therefore gives it the ability to capture three-dimensional images, a process known as stereo photography [1]. Many researches showed that it still has some error and instability in spite of that it requires several techniques in computer vision for stereo camera. Thus, stereo technique is imperfect. Partly, because the nature of stereo system that requires the use of multiple cameras. In this sense, it would be beneficial to use a depth camera instead of stereo cameras. We choose a new device called "Microsoft Kinect".

Microsoft Kinect has many pros; one is inexpensiveness. On the contrary, its cons are among precision. This research addresses the problem of increasing precision of Kinect in 3D mesh reconstruction. We will show that using a simple spatial filtering technique will greatly help improve the precision of Kinect.

What are we going to talk about Kinect? The topics are among, first, its characteristics, second, its calibration (camera model, color camera calibration, depth camera calibration, calibration between color and depth cameras). However, for our main objective of precision improvement we have to talk about other topics beforehand as will be stated below.

The motivation that leads to the main objective is our application. In this research, Burn Care in Nopparat Rajathanee Hospital and Computer Vision Laboratory in Kasetsart University (Thailand) aim to apply Kinect and computer vision theories for detecting and estimating burn area of human subjects. The hospital requires a good system to calculate ratio between burn area and whole body surface precisely. Statistics shows that death rate for patients with 50 % burn area are 52 %. The death rate will decrease if a patient gets the correct treatment in primary period. One crucial treatment procedure is giving a correct does of water to the patient. The dose value depends on burn area ration and can be computed by Eq. (1):

$$Doses = 4cc \times \%Burn \times BodyWeight. \tag{1}$$

As can be observed, to estimate the dose of water, physician must know the proportion between burn area and body surface. Nowadays, to measure the proportion, physician use eye estimation with a body chart called rule of nine to determine the percentage of burn area for each major section of body [2]. However, the approximation of burn area is not accurate.

In order to improve the precision of this medical estimation, we focus on generating 3D mesh using point cloud from Kinect. Why? After generating 3D mesh we can find the ratio between burn area and body area. To generate the mesh, we use marching cube algorithm. However, using Kinect data directly, the mesh of simple one layer object surprisingly results in multiple layer of mesh. Hence we cannot find the ration of burn area from this mesh. Our research will show that applying median filter to the point cloud prior to marching cube helps reduce the problem of multiple wrong mesh layers.

This document is organized as follow. Next section shows what are the methods in research? It consists of system overview, a flow chart, and so on. It also identifies the materials to work in our system and explains the initial methods to develop in our system. It presents characteristics of Microsoft Kinect, basis of a comprehensive Kinect calibration method, and criterion of complete mapping 3D into 2D. It also describes the tri-state segmentation using watershed algorithm and Chebyshev's inequality. The 3D reconstruction methods and Heron's formula also are proposed in Sect. 5. Section 6 shows more results and discussion. Finally, conclusion is in Sect. 7. You will understand details more…

2 Methods

2.1 System Overview

Hardware system consists of a Microsoft Kinect, as a color and depth sensor, and a computer. It will be used to capture images of burn patients. Depth and color images can be obtained from Kinect. We will show how to calibrate each sensors as well as how to calibrate extrinsic parameters between sensors.

Software system includes color image segmentation, color and depth images mapping, depth image filtering, and 3D mesh reconstruction. These steps are essential for generating burn area surface and normal skin surface. Accurate ratio between such two areas can help doctors decide correct doses of liquid for patients.

Among the mention algorithms, we need color image segmentation is important because captured images still compose of an environment and the burn patient. Thus we need to segment surface of environment and patient. Furthermore, we need to segment surface of burn area and normal skin as well. This is done using tri-state segmentation.

Color and depth images mapping is necessary because we perform tri-state segmentation in color images nevertheless do 3D reconstruction using depth

images. It is quite straightforward to segment the burn area in color images. However calculating the burn area in 3D images is more accurate than doing so in color images. To achieve this, both images must be in the same views. This is done using extrinsic parameters from calibration state.

Depth image filtering is essential as Kinect's accuracy is low. We have tested that generating 3D mesh directly from Kinect data is errorness. The resulting surface of even a simple plane turns out into multiple layer of mesh. We propose to correct this using average filter.

Finally, 3D mesh reconstruction is indispensible because the ratio of burn areas is to be calculated base on such mesh. Then we sum up the areas of many mesh to represent the burn/skin areas. To generate the mesh, we use a well-known Marching cube algorithm.

This thesis is organized as followed. First a background of Microsoft Kinect is discussed and characteristics of Microsoft Kinect are explained. We will talk about physical components of Kinect, resolution of depth sensor, and the comprehensive Kinect calibration method. Calibration parameters are then used to map depth information to color image [3]. And after that the tri-state segmentation of color image is performed to detect burn image area. The segmentation uses Watershed algorithm and Chebyshev's inequality [4]. After that we map depth information to color images, of only burn area and body surface. Next step is filtering and then triangular mesh reconstruction. The reconstruction algorithm used in this work is marching cube [5]. Finally, Heron's formula is used to calculate triangular area [6]. The overall process can be illustrated as below figure. It is divided into three states composing of pre-processing state, processing state and post-processing state (Figs. 1, 2).

2.2 A Background and the Characteristics of Microsoft Kinect

Since November 4, 2010, Microsoft has released a new device that shows in North America Market. This devices is Kinect for Xbox 360, is a motion sensing input device for the Xbox 360 video game console [7]. Kinect was developed in cooperation with PrimeSense and has a divergent concept than the Wii Remote and PlayStation Move: it creates a depth image [8]. Kinect is a new alternative gaming device that is capable to capture color and a depth image at the same time (Fig. 3). It consists of RGB camera, 3D depth sensor: depth camera and IR projector, multi-array microphone, motorized tilt (Fig. 2). It has been determined that the Kinect sensor outputs video at a frame rate of 30 Hz. It also has the image size around 640 × 480 pixels. It consist of RGB camera (Color CMOS- VNA38209015 [10]), 3D depth sensor: IR camera (IR CMOS-Microsoft/X853750001/VCA379C7130 [10]) and IR projector (IR Projector-OG12/0956/D306/JG05A [10]), multi-array microphone (4 microphones), motorized tilt [11]. The RGB video stream uses 8-bit

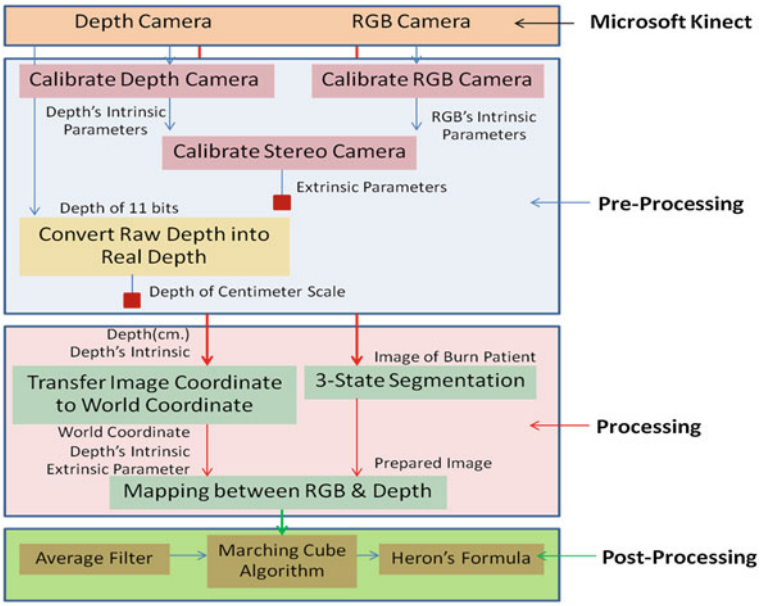

Fig. 1 Flowchart of our system

Fig. 2 Microsoft Kinect

VGA resolution with a Bayer color filter, while the monochrome depth sensing video stream is in VGA resolution with 11-bit depth, which provides 2,048 levels of sensitivity [7].

The depth measurement process is based on a triangulation process [8, 12]. From a technical perspective, the depth sensor uses a static Laser pattern projector and a CMOS camera to triangulate a dense depth map (Fig. 4).

Illumination intensity, baseline, depth of field and speed are tuned to cover a depth range that depends on several open source. The sensor can be use in typical indoor environments for the purpose of segmenting and 3D motion recognition [10].

The resulting RGB value and point cloud can be loaded to a computer using open source libraries which enable Kinect to be operated with Windows, Linux or Mac. Data connection to the computer is through USB interface.

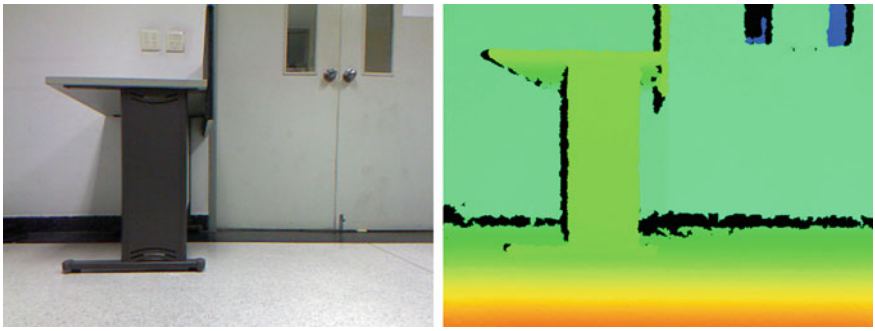

Fig. 3 Color and depth images captured from Kinect

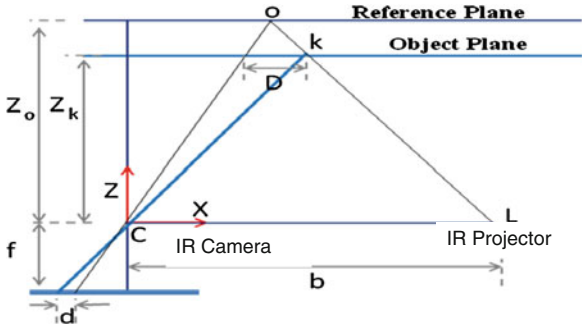

Fig. 4 Triangulation methods (with structured light)

Kinect's mechanism is similar to 3D scanner that IR projector of Kinect is used for structured light scanning [13, 14]. The principle of depth sensor is triangulation method (reference plane) which are reflected by objects and recaptured by the IR camera as follows:

$$\frac{D}{d} = \frac{(z_o - z_k)}{z_0} \tag{2}$$

$$\frac{d}{f} = \frac{D}{z_k} \tag{3}$$

where, Eq. (2) is a ratio of disparity and depth distance between object plane and depth sensor. Equation (3) is a ratio of intrinsic parameters and depth parameters. D is a disparity of object's position between reference and object plane. d is a disparity of object's position from the optic axis of intrinsic parameters. z_o is a depth of reference plane. z_k is a depth of object plane. Last, f is a focal length of IR camera.

However, limitation of depth sensor is still importance factor to modify 3D informations. Thus the error and imperfection of the Kinect possibly originated from two main issues:

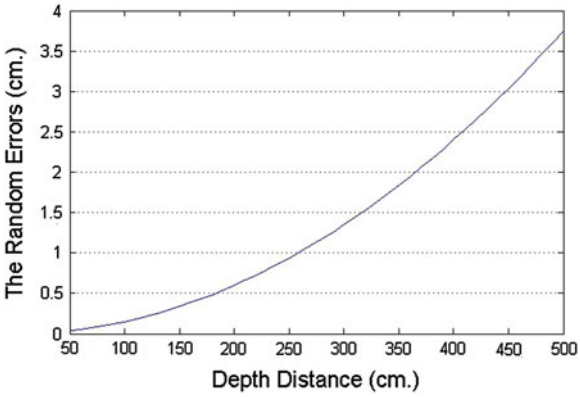

Fig. 5 Shows the calculated standard deviations plotted with the depth distance from the plane to the depth sensor

- The depth sensor: the random error is occurred from estimation of the calibration parameters.
- The measurement setup: the IR laser speckle appears in low contrast in the infrared image at strong light.

Furthermore, as the depth sensor of Kinect used infrared method, some object surface does not reflect IR as they are absorption, transparent, translucent, or silky.

For precision issue, Kinect cannot match with off-the-shelf laser scanner because Kinect's precision range is millimeter, not micrometer. Figure 5 shows the calculated standard deviations plotted with the depth distance from the plane to the depth sensor [15]. The random errors can be increase from a few millimeters at 0.5 m distance up to a few centimeters at the faraway range of the depth sensor. In fact, the random errors increase with the square distance from the depth sensor as follows:

$$F(Z) = 1.5 \times 10^{-5} \times Z^2 \tag{4}$$

where, Z is a depth in real world coordinate. $F(Z)$ is the error value of each point in the same plane. For example, at 50 cm distance, the random error is around ± 0.375 mm of each point in the same plane [15] (Fig. 6).

3 A Comprehensive Kinect Calibration Method

With an active scanning technique, an inexpensive Kinect is used in order to acquire RGB and depth information (Fig. 7a, b). Alignment and mapping between RGB and depth image is applied to calculate the burn surface area of patient. Prior to that, we must calibrate the equipment which can be called pre-processing state. The calibration consists of calibration of RGB camera (intrinsic parameters),

Fig. 6 The depth image consists of a chessboard with manually detected corners shown as *red points*

calibration of depth camera (raw depth to real depth conversion, intrinsic parameters), and calibration between RGB and depth camera (extrinsic parameters). However, the basis of camera model is added to understand more.

3.1 Calibration of RGB/Depth Camera

In this research, we assume that lens distortion is so small and negligible. The method and process for camera calibration is utilized to calibrate the RGB and depth camera of Kinect. For RGB camera, the chessboard is captured into color image to detect its corners (in Fig. 8a). To estimate the intrinsic parameters of the RGB camera, the points of chessboard corners in world and color image coordinate are input into *cvCalibratecamera2()* function of OpenCV.

For depth camera, we test to calibrate of the depth image first. We added input of the depth image to estimate intrinsic parameters. The same chessboard is captured into depth image to detect its corner (in Fig. 8b). It is necessary to fix the four corners of the chessboard in depth image (in Fig. 6). Then, the points of chessboard corners in world and depth image coordinate are input into *cvCalibratecamera2()* to calculate the intrinsic parameters of the depth camera. It is not perfectly when we manually configure the four corners of chessboard as shown Fig. 6. Thus, the IR images are captured and are used to calibrate function and stereo function.

For mapping between RGB and depth images, we must use intrinsic and extrinsic parameters of the two cameras. In order to do so, we also calibrated the extrinsic parameters between the RGB and depth cameras. A known object, chessboard, is captured into RGB and depth images simultaneously. Then, corners of the chessboard in the RGB and IR images are detected to estimate the intrinsic and extrinsic parameters using *cvStereoCalibrate()* function with OpenCV.

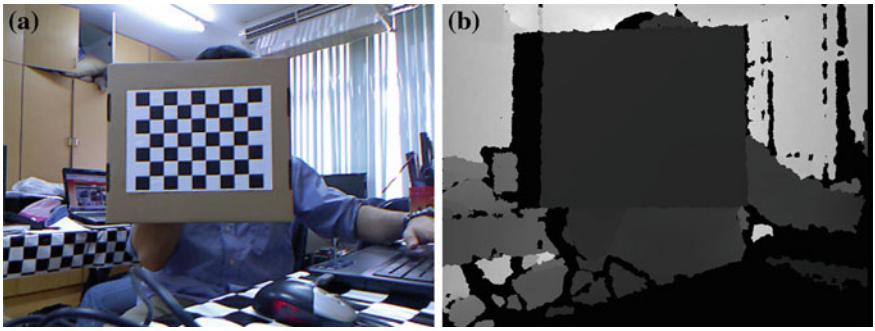

Fig. 7 The output images from RGB and depth cameras from Kinect. **a** Color image. **b** Depth image

3.2 Calibration of Depth Sensor

Since, Kinect's depth camera returns the raw depth data. For acquiring real depth data, depth calibration is applied using line regression technique. Then, raw depth data must be transformed into the depth data (in centimeter) using Eqs. (5) and (6):

$$Depth = \frac{1.0}{f(d)} \tag{5}$$

$$f(d) = -0.00307110156374373d + 3.33094951605675 \tag{6}$$

where;

d is the depth data which measure from depth sensor

$f(d)$ is a linear function which can be estimated from regression techniques (Fig. 9) [15].

Using the intrinsic parameters of depth camera, the 3D point (X, Y, Z) can be approximated from the depth image (x, y, Z) in Eqs. (7) and (8).

$$X = \frac{Z(x - c_x)}{f_x} \tag{7}$$

$$Y = \frac{Z(y - c_y)}{f_y} \tag{8}$$

$$Z = depth \tag{9}$$

where;

x, y are pixel of depth image

Z is a distance between object and Kinect

f_x, f_y, c_x, c_y are the elements of intrinsic parameter[M] of depth camera.

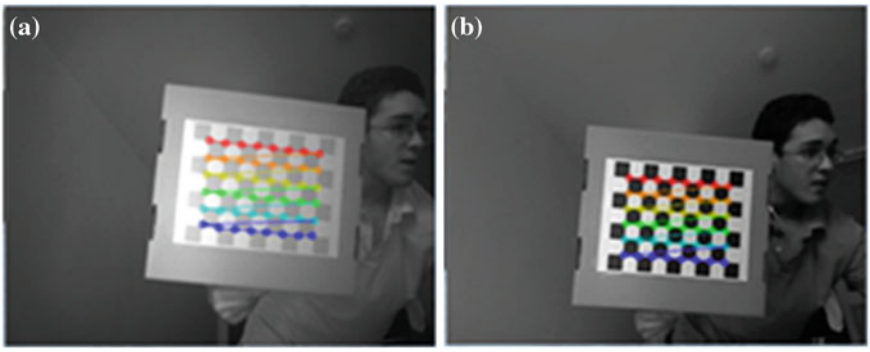

Fig. 8 Show the result of the experimental image. **a** IR image. **b** RGB image

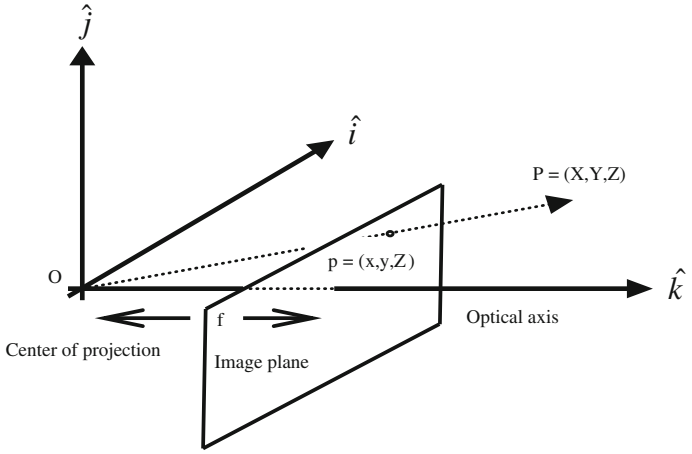

Fig. 9 Displays the image coordinates (xyZ) are projected into the world coordinates (XYZ)

3.3 Mapping Between RGB and Depth Image

Using the extrinsic parameters [R|T], the Eq. (10) is used to project each 3D point (XYZ) in depth camera coordinate onto the RGB camera coordinate:

$$P3D' = R * P3D + T \tag{10}$$

$P3D$ is a matrix consists of X, Y, Z in depth camera coordinate. The rotation and translation matrices are transferred in RGB camera coordinate ($P3D'$) as shown Fig. 10.

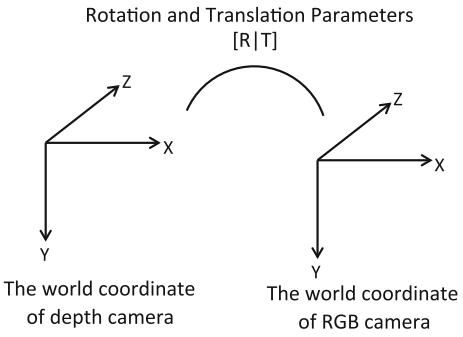

Fig. 10 Transfering the depth coordinate into the RGB coordinate

After that, the intrinsic parameters of RGB camera are processed with 3D point on the color image. We can then map the 3D depth images onto 2D color images using Eqs. (11) and (12):

$$P2D.x = \left(\frac{P3D'.x * f_x}{P3D'.z}\right) + c_x \tag{11}$$

$$P2D.y = \left(\frac{P3D'.y * f_y}{P3D'.z}\right) + c_y \tag{12}$$

where, f_x, f_y, c_x, c_y are the elements of intrinsic parameter [M] of RGB camera. The *P2D* is a position of 2D color image as shown Fig. 3. Do not confuse the mapped image with the RGB image. The mapped image consists of RGB plus depth information, shown in 2D coordinate. In other words, we can also show the mapped image in 3D coordinate as well, using OpenGL.

4 The Tri-States Segmentation

To achieve our goal of burn area proportion estimation, we must divide the depth information into three areas, namely, the burn area, the body area and the background area.

Before burn segmentation cases, we design to threshold the area of sample objects. It also uses the same procedure that consists of sub-sample area, sample area, and environment area to compare accuracy of the surface area resulting with known ground truth object in Fig. 11. Luckily we already mapped the RGB image and depth image. Hence, the segmentation can be done based on color and edge information in RGB image. Then the result can be reflected onto the same area in the depth image. The segmentation program is based on a locally segmentation method called watershed algorithm and a statistical threshold called Chebyshev's inequality. The detail of algorithm can be found from [4].

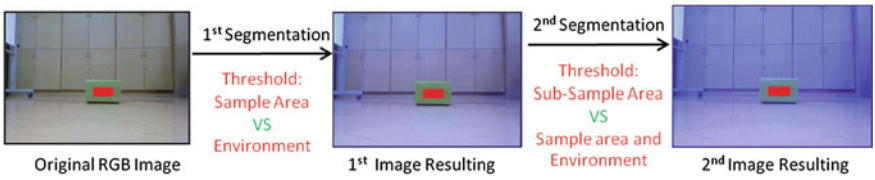

Fig. 11 The image resulting of segmentation program. There are 3-states segmentation: sub-sample area, sample area, and environment

5 Triangular Mesh Reconstruction

5.1 Average Filter

Average filter normalizes noisy surface into an enhanced one.

Figure 12 shows a point cloud of a planar surface. One may notice that it is not in the same plane, even it should be so. This is due to the fact that Kinect's resolution is coarse. So we normalize it using the average filter only for depth information (Z axis) of 3D-world coordinate (Fig. 13).

$$\overline{Z} = \frac{1}{n}\sum_{i=1}^{n} z_i \tag{13}$$

where, z_i is a raw depth data of each point cloud. $\overline{Z}$ is a normalize depth data of average values. n is a number of average set.

5.2 Marching Cubes Algorithm

For point cloud to 3D triangular mesh reconstruction, there are many methods, e.g. intrinsic property driven (IPD) method [16]. We will focus on marching cubes as it is a well-known technique and, currently, many variants of it are still being developed.

Marching cubes algorithm is one of the latest algorithms of surface construction used for viewing 3D data. In 1987 Lorensen and Cline (GE) described the marching cubes algorithm. It has been published at the SIGGRAPH convention. It is a high resolution 3D surface construction algorithm [17]. This algorithm generates a triangular mesh estimates iso-surface. It calculates the normal vector of the surface at each vertex of the triangle (Fig. 14).

The principle of marching cubes algorithm locates the surface of point cloud in a cube of eight vertexes. The surface intersects edge of this cube where one vertex is outside and the other inside the surface. There are 256 cases that the surface may intersect this cube. It reduced to 15 patterns using symmetries reduces. After that, it calculates normal vectors and marches to the next cube.

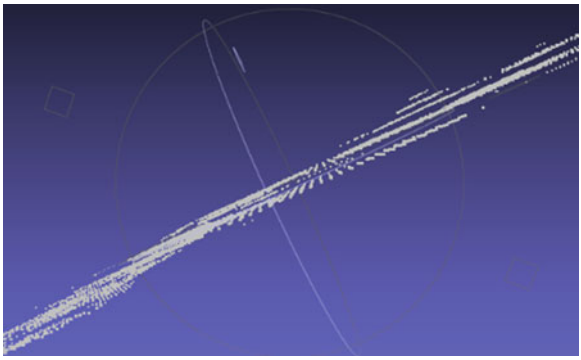

Fig. 12 A set of point cloud of the planar object that is shown in *top* view

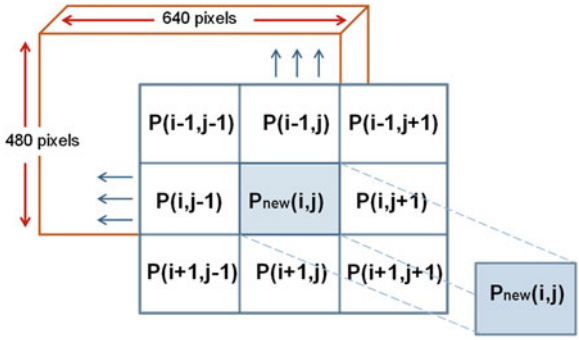

Fig. 13 Searching windows size used 3×3 of the average filter using mean equation

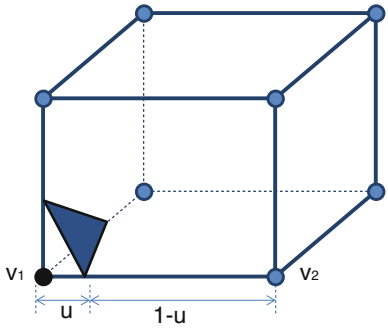

Fig. 14 The cube of eight vertexes is represented with surface intersection by triangular mesh

Interpolate surface intersection along each edge:

$$v_i = v_i * (1 - u) + v_2 \tag{14}$$

$$u = \frac{(v_1 - v_i)}{(v_1 - v_2)}. \tag{15}$$

Calculate normal for each cube vertex:

$$G_x(i,j,k) = \frac{(D(i+1,j,k) - D(i-1,j,k))}{\Delta x} \tag{16}$$

$$G_y(i,j,k) = \frac{(D(i,j+1,k) - D(i,j-1,k))}{\Delta y} \tag{17}$$

$$G_z(i,j,k) = \frac{(D(i,j,k+1) - D(i,j,k-1))}{\Delta z}. \tag{18}$$

The central differences with the three coordinate axes are used to estimate the gradient at cube vertex (i,j,k).

Interpolate the normals at the vertices of the triangles:

$$\overrightarrow{n_1} = \overrightarrow{ug_2} + (1-u)(\overrightarrow{g_1}). \tag{19}$$

5.3 Area Calculation

From the triangular mesh, we use the Heron's formula to calculate each triangle's area [18].

Heron's formula is a method for calculating the area of a triangle when you know the lengths of all three sides. The formula is credited to Heron (or Hero) of Alexandria, and a proof can be found in his book, Metrica, written c. A.D. 60.

The area T of a triangle whose sides have lengths a, b, and c is

$$T = \sqrt{(s(s-a)(s-b)(s-c))} \tag{20}$$

$$s = \frac{(a+b+c)}{2} \tag{21}$$

where;

T	is a triangular area of each triangular mesh
s	is the semiperimeter of the triangle
a, b, c	are lengths of each sides.

6 Experimental Result

First experiment, we design to test sample planar objects which must be known area as shown Table 1. Then we apply to test the burn surface area per a view. It is captured by depth and color cameras in the same time.

To efficiently segmentation program, we consider color image only. Thus, the color image is divided using Watershed Algorithm and Chebyshev's Inequality

Table 1 Comparison results of ground truth and average filter techniques

Surface area distances (in front of views)		Result of surface area (cm^2)			% Error of surface area	
Width (cm)	Length (cm)	Actual area	Marching cubes	Marching cubes (normalized)	Marching cubes	Marching cubes (normalized)
5.11713	8.28747	42.40806136	68.1104	45.3756	60.6072002	6.997581459
4.66838	7.91935	36.97053515	60.3286	39.4187	63.1802184	6.621935108
4.76699	9.06871	43.23044988	52.9238	46.8473	22.4225058	8.366441078

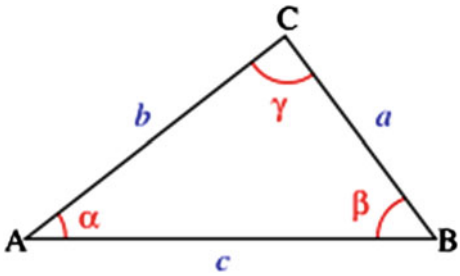

Fig. 15 A triangle with sides *a*, *b*, and *c*

into 3 areas, there are 2 steps. First, segment between the skin and the background. Second, further segment the skin into normal skin and the burn areas. Now we are ready to prepare image for reconstruction (Fig. 15).

Experiment is divided into two parts. First is the average filter's evaluation. Second is burn area estimation. First, we collect point cloud of planar objects and compare the resulting area of our method with and without average filter. We calculate the resulting area of sample rectangular between the ground truth and our programs. In this experiment, a set of point cloud is captured in range between 0.5 and 1 m as shown in Fig. 16. The result is shown in this table, it is clear that average filter enhances the correctness of surfaces dramatically.

From the Table 1, the sizes of sample rectangular are used to estimate actual surface area for comparing with the surface area that is calculated from a set of point cloud. Using marching cubes algorithm, triangular meshes of object is generated to calculate the surface area that is shown in marching cubes table. Also, an average filter is used to normalize the result of surface area that is shown in marching cubes (Normalized) table. According to error of surface area, the resulting area with average filter is more accurate than without average filter and is similar to the actual surface area. It is clear that average filter (normalization) enhances the correctness of surfaces dramatically (Fig. 17).

Second, we segment the burn area and the skin area as shown in Fig. 18. An example of such area's point cloud is shown in Fig. 19.

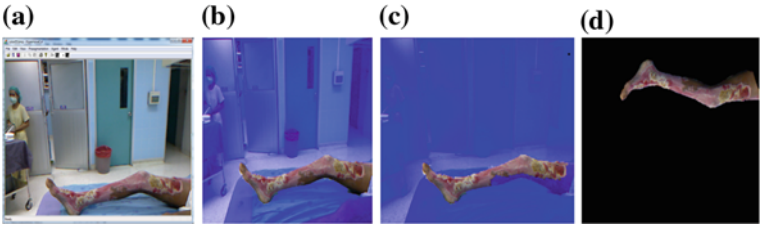

Fig. 16 The result images of segmentation program (**a**) Original image (**b**). The skin area is separated between skin area of patient and environment. **c** The burn area is separated between burn area and skin area. **d** Prepare image for mapping program

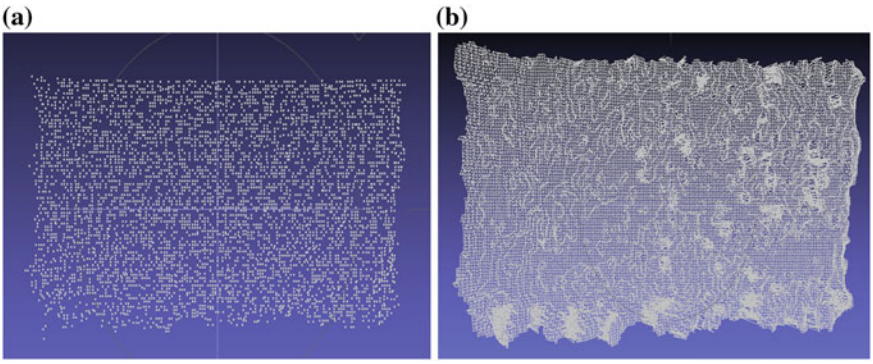

Fig. 17 **a** Point cloud of sample object is normalized using average filter. **b** Triangular mesh area of planar object is generated using marching cubes algorithm

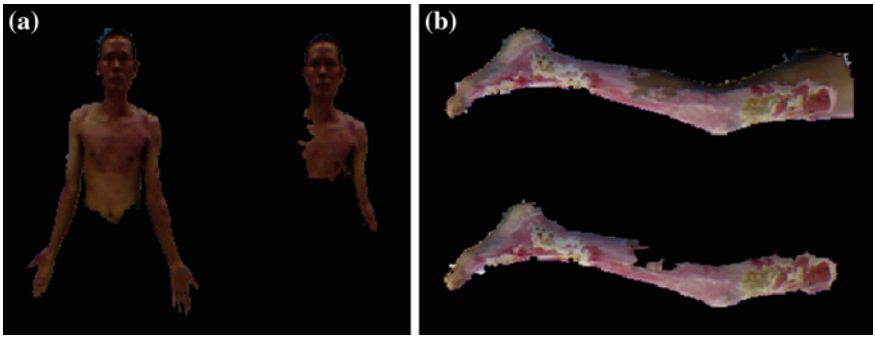

Fig. 18 **a**–**b** Show the segmentation of burn area and the skin area

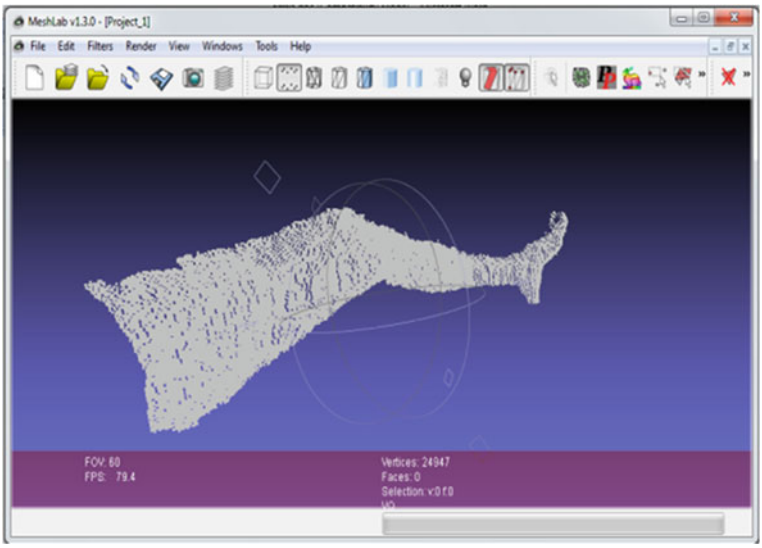

Fig. 19 The rest point cloud is deleted that shows in MeshLab program

7 Conclusion

Using marching cube algorithm to generate triangular mesh from Kinect data is erroneous. The error of Kinect increases in proportion with the distance from the device [8]. A simple method that can filter such point cloud to be more accurate is proposed. Experiment is done with the target objects in the range between 0.5 and 1 m. Error rate of triangular mesh area reduces to a great deal. The remaining error, in Table 1, is assumed to be that due to inaccuracy of Kinect itself as mentioned in [8].

References

1. Wikipedia, Stereo camera (2011), http://en.wikipedia.org/wiki/Stereo_camera. Accessed 7 Sept 2011
2. B. des Bouvrie, Improving RGBD Indoor Mapping, The Netherlands (2011)
3. K. Conley, Wikipedia (2011), http://www.ros.org/wiki/kinect_node. Accessed May 2011
4. K. Ogawara, K. Ikeuchi, M. Ruchanurucks, Integrating region growing and classification for segmentation and matting, Tokyo (2008)
5. H.E. Cline, W.E. Lorensen, Marching cubes: a high resolution 3D surface construction algorithm, in *Proceeding SIGGRAPH '87 Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques* (1987)
6. J.W. Wilson, Problem solving with heron's formula, in *The Northwest Mathematics Conference* (1986)
7. Wikipedia, Kinect (2011), http://en.wikipedia.org/wiki/Kinect. Accessed 3 Sept 2011

8. K. Khoshelham, Accuracy analysis of Kinect depth data (ITC Faculty of Geo-information Science and Earth Observation, University of Twente, 2011)
9. A. Khongma, M. Ruchanurucks, T. Phatrapornnant, Y. Koike, P. Rakprayoon, Kinect calibration and quality improvement for triangular mesh reconstruction, Bangkok (2012)
10. Teardown, Microsoft Kinect Teardown (2010), http://www.ifixit.com/Teardown/Microsoft-Kinect-Teardown/4066/1. Accessed 4 Nov 2010
11. Wikipedia, File: KinectTechnologiesE3.jpg (2011), http://en.wikipedia.org/wiki/File:Kinect TechnologiesE3.jpg. Accessed 14 Jan 2011
12. M. Zlatka, Triangulation methods (2009), pp. 9–10
13. A. Shpunt, Y. Arieli, B. Freedman, Distance-varying illumination and imaging techniques for depth mapping, United States Patent Application (2010)
14. P. Cignoni, C. Montani, P. Pingi, R. Scopigno, C. Rocchini, A low cost 3D scanner based on structured light, The Eurographics Association and Blackwell Publisher, vol. 20 (2001)
15. K. Khoshelham, Accuracy analysis of kinect depth data (International Society for Photogrammetry and Remote Sensing, 2011), p. 6
16. Z. Jianhui, Y. Zhiyong, D. Yihua, Z. Yuanyuan L. Chengjiang, Improvements on IPD algorithm for triangular mesh reconstruction, in *International Conference on Multimedia Information Networking and Security* (2009)
17. H.E. Cline, W.E. Lorensen, Marching cubes: a high resolution 3D surface construction algorithm, in *Proceeding SIGGRAPH '87 Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques* (1987)
18. J.W. Wilson, Problem solving with Heron's formula, in *The Northwest Mathematics Conference* (1986)

http://www.springer.com/978-3-319-04692-1

Soft Computing Techniques in Engineering Applications
Patnaik, S.; Zhong, B. (Eds.)
2014, VI, 206 p. 134 illus., 57 illus. in color., Hardcover
ISBN: 978-3-319-04692-1