16

Point Cloud Streaming for 3D Avatar Communication

Masaharu Kajitani, Shinichiro Takahashi and Masahiro Okuda
Faculty of Environmental Engineering, The University of Kitakyushu
Japan

1. Introduction

Real-time 3D imaging has gained special attention in many fields such as computer vision, tele-imaging, and virtual reality. Especially real-time 3D modeling is one of emerging topics. During the last years, many systems that reconstruct 3D images using image sequences obtained from several viewpoints have been proposed. In the work (Saito et al., 1999), the 3D image sequences are acquired by multiple images of about 50 cameras. They accomplish high quality 3D images. In the method, a modeling step is done offline. In (Wu & Matsuyama, 2003), (Ueda et al., 2004), (Würmlin et al., 2002), and (Würmlin et al., 2004), the 3D modeling is employed in real-time using parallel processing with PC clusters. All of them use Silhouette Volume Intersection (SVI) method to approximate a shape by a visual hull. In order to make the surface finer and map textures on it, some extra tasks are required, which are in principle time-consuming.
On the other hand, some methods have been established to generate range images in realtime, for example (Kanade et al., 1996). Some of them have already been commercialized (Pointgrey). These methods achieve range images at 5-30 fps by the stereo matching algorithm.
In this paper, we propose a client-server system in which the server reconstructs, encodes, and transmits the 3D images to client in real-time. We do not apply the polygonal modeling but transmit a point cloud of multiple range images and render it directly, due to the following two reasons:

1. To reconstruct mesh model, a triangulation is required. Moreover, one needs to extract textures from colour images, and map them onto the meshes, which are complicated and time-consuming. The computational cost of the processes depends on complexity of the object shapes. The point rendering does not need these.
2. In general, the polygonal meshes have connectivity information as well as the 3D coordinates of the vertices. The connectivity has to be encoded without any loss, which consumes many bits. Moreover, only one bit error on the connectivity causes catastrophic damage on its quality. This is not suitable for transmission over wireless network or UDP.

The point cloud can easily be divided and packetized to individual blocks.
The client receives and decodes a bit-stream, and then renders the point cloud quickly by using a strip-based rendering. As a rendering method for the point cloud, Point-Sampled-Geometry (Rusinkiewicz & Levoy, 2000) is well known, which is used to represent high

Source: Stereo Vision, Book edited by: Dr. Asim Bhatti,
ISBN 978-953-7619-22-0, pp. 372, November 2008, I-Tech, Vienna, Austria

resolution mesh data. Our rendering scheme is similar to the method. The client divides point cloud obtained from range image into strips, and renders it with triangle strips.
In certain situations at the modeling stage, it is hard to capture the 3D shape perfectly. Failure to obtain precise camera-calibration or Region of Interest (ROI) extraction results in overlaps and annoying artifacts between different range images. To address these problems, we apply a cylindrical method to each frame of 3D sequence before rendering stage. The cylindrical method maps each point of the Point Cloud on the inner wall of a virtual cylinder, which is then warped to a 2D plane. Mapping of the 3D points on a plane facilitates handling of the 3D Point Cloud efficiently, and makes the process work in realtime.
In the next section, we review conventional real-time 3D imaging methods. In Section 3, we introduce our client-server system that reconstructs, encodes and displays the 3D images. In Section 4, we verify the validity of our method with some experimental results. And then we conclude the chapter in Section 5.

2. Past work

In this section, we review conventional methods that reconstruct 3D mesh models and explain the drawback and advantage with comparison to our method.
Some papers (Wu & Matsuyama, 2003), (Ueda et al., 2004), (Würmlin et al., 2002), (Würmlin et al., 2004), have already presented systems reconstructing 3D images in real-time. In many of these methods, first object silhouettes are extracted from multiple images obtained from several viewpoints. Next, these silhouettes are re-projected to a 3D space, and the intersectional space is assumed as Visual Hull of the object. The procedure is called Silhouette Volume Intersection (SVI). Fig. 1 illustrates the SVI method. This method successfully reconstructs the rough shape of an object without holes, while the stereo method often fails to make a continuous surface due to mismatch of corresponding points. Thus the shape without holes can be reconstructed robustly if the image is divided exactly into regions and the silhouettes are faithfully extracted. However, SVI method has a drawback that it is difficult to reconstruct curved or concave surfaces in principle. In (Wu & Matsuyama, 2003), to overcome this drawback, the Visual Hull is divided into voxels and the 3D mesh is estimated by Marching Cube method, finally, high-resolution shape is obtained by relocating vertices of 3D mesh appropriately.
On the contrary, our method reconstructs a 3D point cloud using depth images obtained from several viewpoints by the stereo method. In general, reconstructing the 3D mesh model by the stereo method requires the following five processes: (1) taking depth image, (2) integration of several depth images, (3) polygonal meshing of a point cloud, (4) simplification of meshes, (5) generating texture and mapping it to the 3D mesh. As mentioned above, the computational cost of the stereo method is basically high. The feature of our method is to treat point cloud with the 3D coordinates and the colour information directly without any 3D meshes. So, calculation is reduced drastically because above (3), (4) and (5) processes are not required. Moreover, each depth image can be processed separately. Thus it can be implemented with simple parallel processing. In (Waschbüsch, et al., 2007), a 3D imaging system based on the stereo matching method has been proposed for real-time application.
In the stereo method, estimating depth is unstable because of the mismatch of corresponding points. The precision of corresponding points depends on lighting condition

or complexity of colour, etc. And it is impossible to reconstruct flat colour surface with low reflectance and specular reflection. And, many points are required to represent the colour information of an object surface precisely because the colour is assigned to each point. And, additional process is necessary before rendering point cloud because the overlapping of the range images is caused.

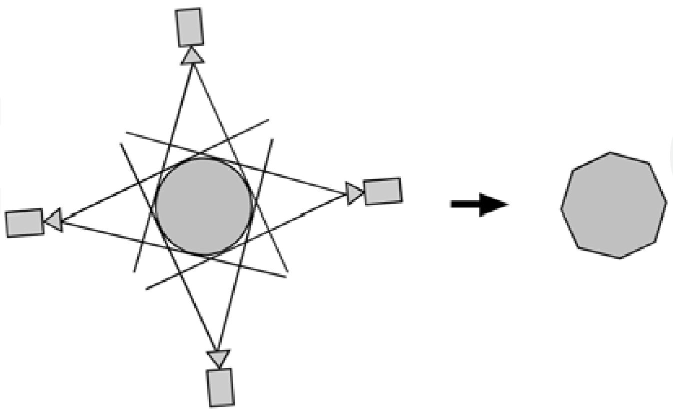


Fig. 1. Silhouette Volume Intersection

3. Client-server system

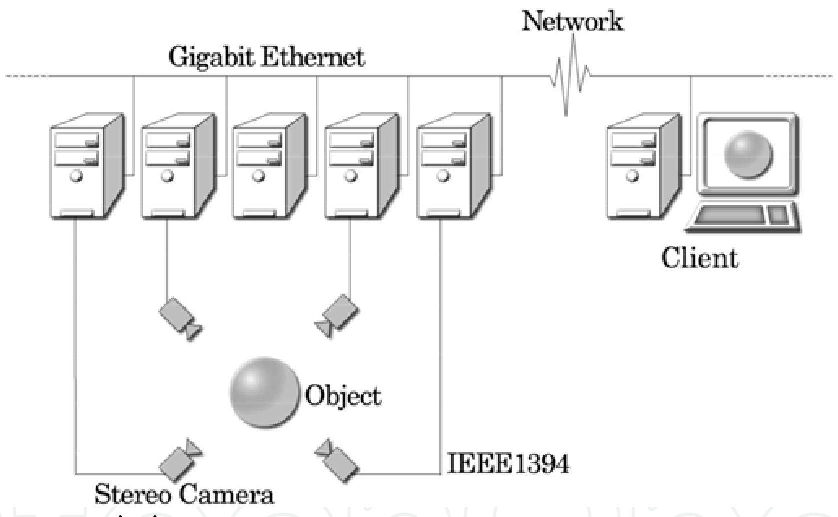


Fig. 2. 3D image transmission system

3.1 Outline

Our Client-Server system is illustrated in Fig.2. The server consists of N+1 Personal Computers (PCs) and N stereo cameras. The PCs in the server are connected through Gigabit switch. We use the commercial stereo cameras [10]. This camera consists of three CCD cameras and transfers three RGB colour images at the same time. The maximum frame rate is 15 fps. Each stereo camera is allocated to surround the object and connected with a PC through IEEE 1394. The PC takes a depth image and encodes it. Thus the entire server takes N depth images. The encoded bitstream is transmitted to the remaining PC, which is responsible for integration of data, packetization, transmission to the client, and sending out a command of taking images to each PC. We call the PC *ParentPC*. The relative position of each stereo camera is calculated by calibration in advance and informed to the client. The stereo cameras are synchronized with each other by special synchronization equipments. *ParentPC* generates synchronized signal, so each PC can take an image at the same time. Hereinafter, we explain the details of the server and the client.

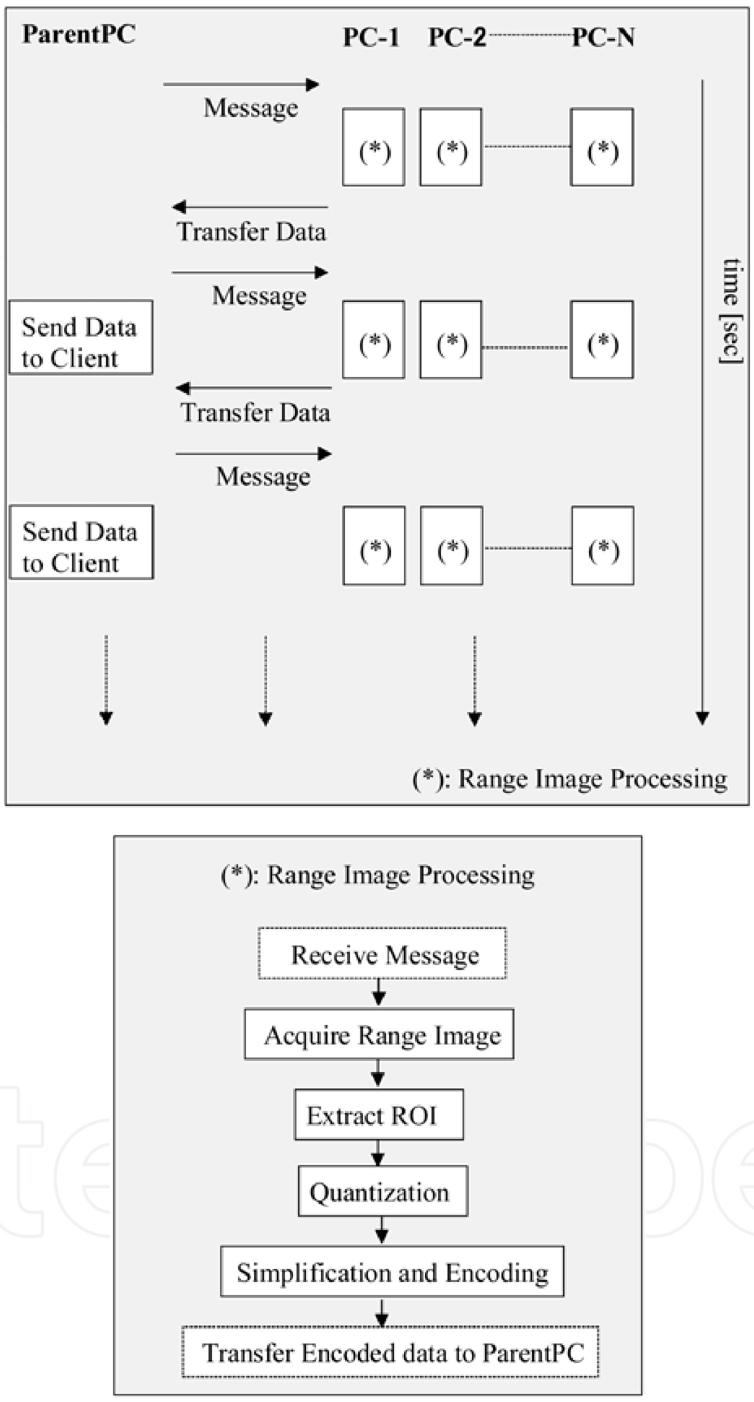


Fig. 3. Processing of depth image for each PC

3.2 Tasks in server

Flowchart of tasks in the server is illustrated in Fig.3. *ParentPC* tells the rest of PCs to take colour images. The *N* PCs does same process in parallel. Here, we explain about the process of the PCs.

Range Image Acquisition, Extraction of ROI, and Quantization

Each PC acquires colour images from its connected stereo camera. Each camera has three CCD units and can capture three colour images at the same time. All PCs are synchronized based on a signal sent from *ParentPC*. At the second step, range images are generated using these colour images by the stereo matching method. In this paper, we do not refer the detail of method of acquiring depth image and we use a conventional method. The next step is ROI extraction based on background subtraction. Here, a background image is taken previously without any object in front of the cameras. This method removes the pixels corresponding to the background based on colour and depth values of the obtained images. In our case, the ROI is a bounding box including remaining pixels and we only transmit the ROIs. Each pixel in ROI is quantized by linear quantization.

Simplification and Encoding

For streaming 3D images obtained in real-time, the simplification and the compression of the 3D images are essential. In general, a 3D mesh model includes coordinate, colour and connectivity information of vertexes. Since this connectivity information must be encoded losslessly and it is represented by a set of integers, it makes coding efficiency relatively low. Most of conventional 3D mesh simplification methods, for example (Hoppe, 1996) and compression methods such as (Khodakovsky et al., 2000) have high computational complexity and it can hardly be executed in real-time.

There exist many compression methods for equally sampled signals and data, such as DPCM (Gersho & Grey, 1992) and an integer type orthogonal transformation (Pratt et al., 1969). These encoding methods can be implemented with low computational complexity and can reduce its entropy after quantization by removing correlation with surrounding pixels. However since in the range images, some pixels may be blank due to failure in finding depth value, the above methods, which basically handles regularly sampled signals, cannot be applied directly. Moreover some extra computation is required for the entropy coding. Our simplification proposed here is simple. After all the vertices are quantized, one first finds the difference to neighbours. And then if the difference is small, completely replace the pixel value with it. The method does not need the entropy coding. We explain the detail of the algorithm.

First, the server scans the pixels in ROI of the range image along the horizontal line from upper left to lower right. For the depth values and the colour of RGB, we calculate the differences between the current pixel and each of four neighbours, the left, and upper left, upper, and upper right pixels. And then, if the absolute value of the difference between the current value and one of the four pixels is smaller than a threshold, the pixel value is replaced with the neighbor's. Fig. 4 illustrates the example. The left pixel is given priority if there were more than one pixel that can be replaced, in order to improve the rendering efficiency with triangle strip, the detail of which is explained later. Note that the pixel values of four compared neighbors are quantized in advance to enable to decode at the client. Four codes are prepared to identify pixels in the neighbourhood. If a pixel is replaced, then one of the four codes is allocated to the pixel. After all, the depth and RGB of the pixel is described by one code. The simplification procedure is applied with the valid pixels only. So, the valid pixels that have only invalid neighbours are not simplified. Fig. 4 (upper right) is the range image after the simplification. The arrows show that the pixel values are replaced with other neighbours indicated by the arrows.

As a result, the encoded bit stream is divided into packets and transferred to the decoder. The packets are well aligned according to a rule so that the client can determine which range images the packets come from.

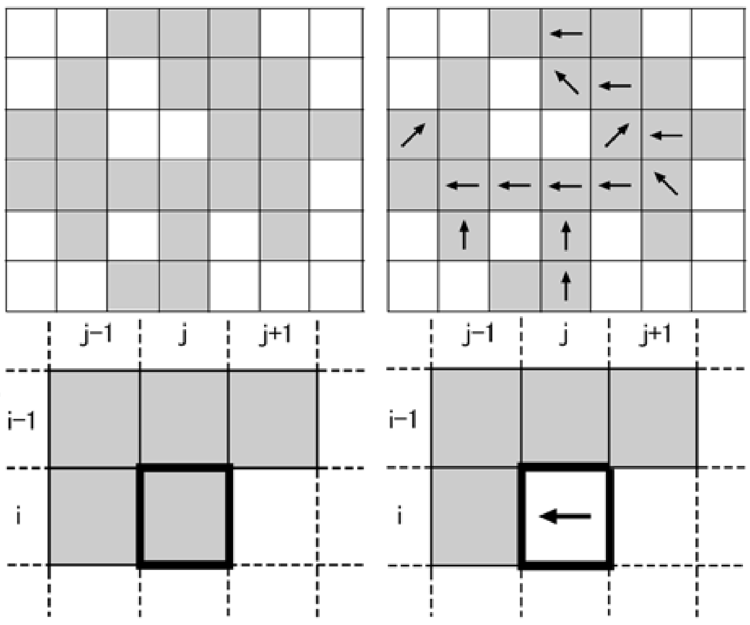


Fig. 4. The range image before (upper left) and after (upper right) the simplification. Current pixel (i, j) is compared with its for neighbors (lower left) and if the difference between the current pixel value and left pixel is smaller than a threshold, the pixel value is replaced with the left (lower right). A special code is allocated to the (i, j) pixel.

3.3 Tasks in client

The client receives the bit-stream of the point cloud from *ParentPC*. The received bit-stream consists of the spatial coordinates and colour information of the points. The 3D shape is represented by the point based geometry. At this time, overlaps and annoying artefacts are caused due to failure of camera calibration or the ROI extraction. In order to remove the invalid points, we apply the cylindrical method to the 3D model before rendering. The detail is explained below.

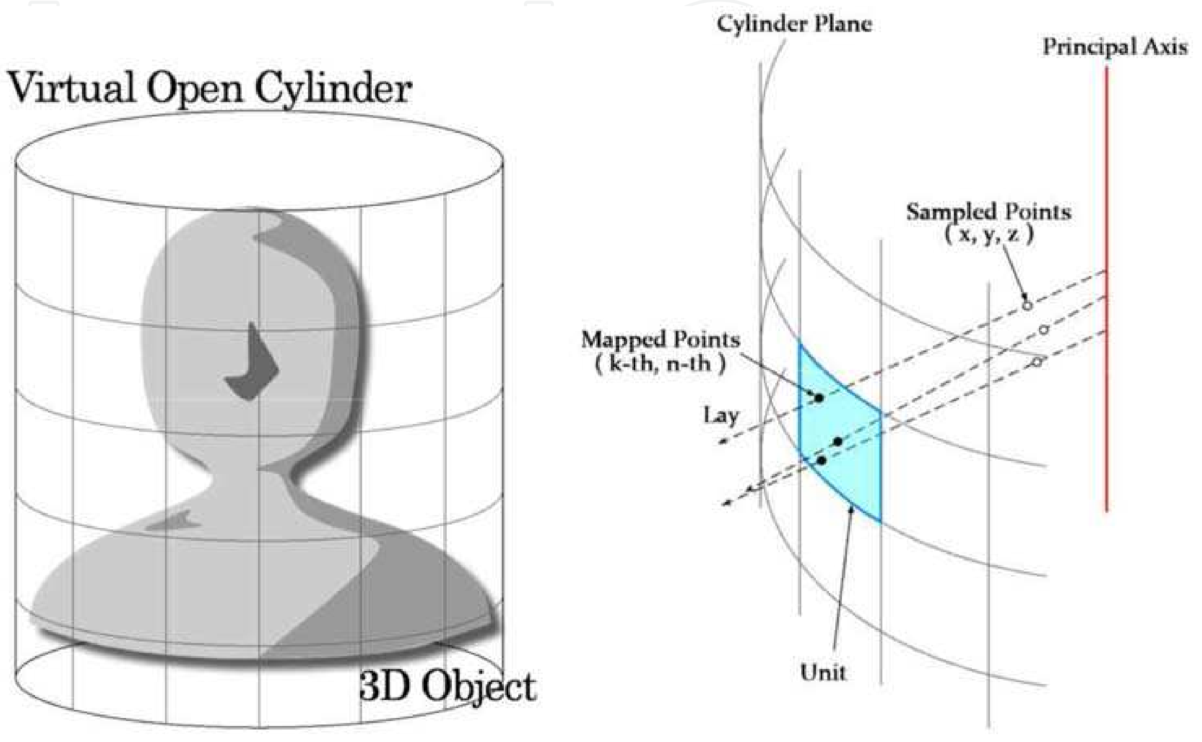
Post Processing

In the 3D image, annoying artifacts often appears due to overlaps between different range images. Fig.13 (a) shows an example of the 3D image with the overlaps. Due to the influence of the shadow of the object itself, the annoying artifacts are often left around the boundary at the ROI extraction stage. That results in image degradation. Since the 3D point cloud does not have connectivity information, it is not straightworward to decide the relationships among points. This makes the determination of the unnecessary points more difficult. To handle this, we employ a method which projects the vertices of the point cloud onto the inner wall of the cylinder. It can treat whole the points in a single 2D plane. Note that the cylindrical method is used to determine the unnecessary points, and that does not actually transform the coordinates of points for rendering.

We assume that the 3D object is surrounded by a virtual open cylinder (see Fig.5). The axis of the cylinder lies along the principal axis of the object. The cylinder and the object are cut into thin horizontal slices. In each slice a 2D point set are mapped from the surface of the object surrounded by a circle corresponding to the circumference of the cylinder (see Fig.6 (a)). In each slice, each point of the 2D point set is mapped on an intersection of a ray and the circle. This ray is emitted from a center of the circle and passes through that point.

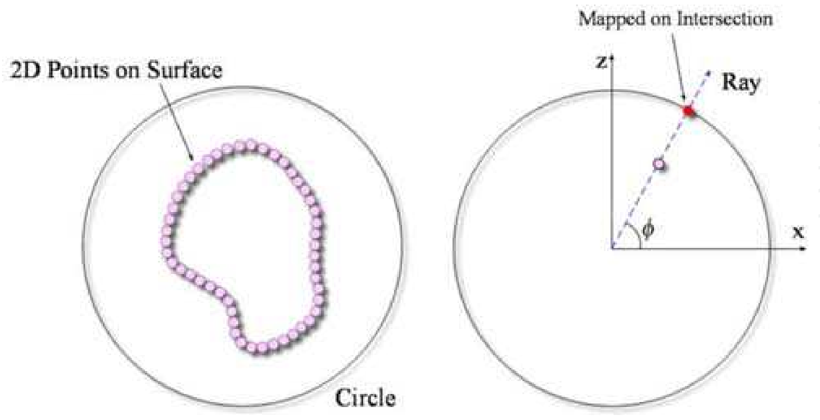
Therefore, a point can be represented by magnitude r and an angle ϕ based on coordinates x and z of that point (see Fig.6 (b)). The values of r and ϕ are calculated by following simple mathematical formula.

$$r = \sqrt{x^2 + z^2}, \qquad \phi = \tan^{-1}(z/x) \tag{1}$$



(a) (b)

Fig. 5. Mapping on the wall of the Cylinder: (a) 3D object with virtual open cylinder. (b) Image of mapping on the inner wall of the cylinder. Sampled points in 3D space are converted to mapped points in 2D plane.



(a) (b)

Fig. 6. Mapping based on angle: (a) One of horizontal slice. 2D point set are surrounded by a circle corresponding to circumference of cylinder. (b) Image of mapping based on angle. Each point of 2D point set is mapped on intersection of ray and circle. Ray is decided angle ϕ calculated by x and z components of that point.

By repeating this process for all points in a slice, all of 2D point set is projected on the circle. When this procedure has been done for all the slices, the entire surface of the object is mapped on the inner wall of the cylinder.
After the mapping process, the cylinder is opened and warped to a plane (Fig.7). Then this plane is divided by a regular grid based on the angle ϕ and the y components. These divided small domains are named "Units". Here, one of the rows corresponds to the slice. If the plane has M columns, points stored in the k-th unit have angle ϕ satisfying the following condition.

$$\frac{2\pi(k-1)}{M} \leq \phi < \frac{2\pi k}{M} \tag{2}$$

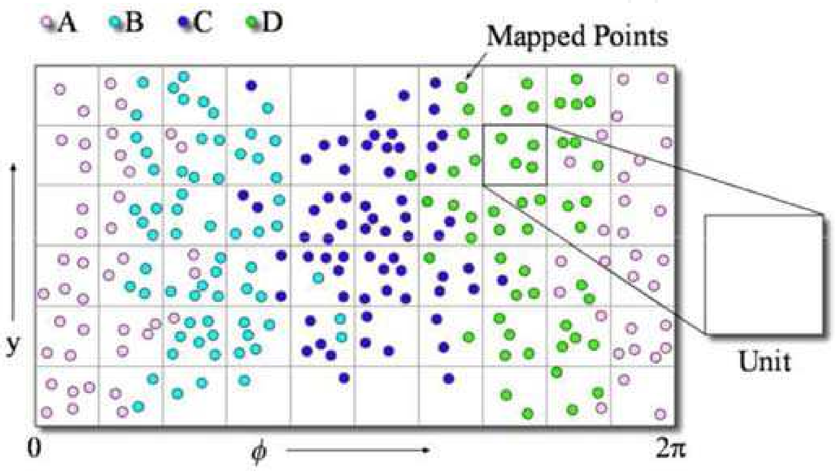


Fig. 7. Unit Plane: Unit Plane warped from the cylinder. This plane is divided by a regular grid. These divided small domains are called "Unit". The units store the mapped points.

We consider the points assigned to a unit as one segment. We can control quality and speed of processing by changing the size of the grid.
When there are points from more than one camera in a segment, we take it as the overlap. Fig.8 shows the image observed from the vertical direction. In Fig.8 (lower left), one unit stores the points from only one camera. However in (lower right), the points acquired from two cameras are stored in the unit. In this case, one of these surfaces should be removed. First, we find a tangent vector of each surface by calculating the differences of each point from the same camera. In the case of Fig.9, we have two vectors. These vectors are normalized. Secondly, the weighted average of each tangent vector is computed using number of points from each camera. Thirdly, the inner products of the average tangent vector and view axis of each camera are calculated. In case of Fig.9 (b), we have two inner products. Finally, the points coming form a camera with the minimum inner product are left, and others are removed (see Fig.9 (c)).
Rendering
The client receives only the XYZ coordinates and its colour of each point. Although the polygonal representation reconstructs a contiguous surface, computational complexity for the triangulation is relatively high. Our method renders the point cloud without reconstructing polygonal meshes to enhance the rendering speed. In our method, the texturing is not needed, since the colour is provided as attributes of the points. To reduce the load of rendering and to make up for the drawback of the stereo method, we adopt the strip-based rendering. First, we explain about a simple point-based rendering method using

a rectangle primitive. Next, we explain about strip-based rendering using triangle strip as primitive, and then we compare the two rendering methods.

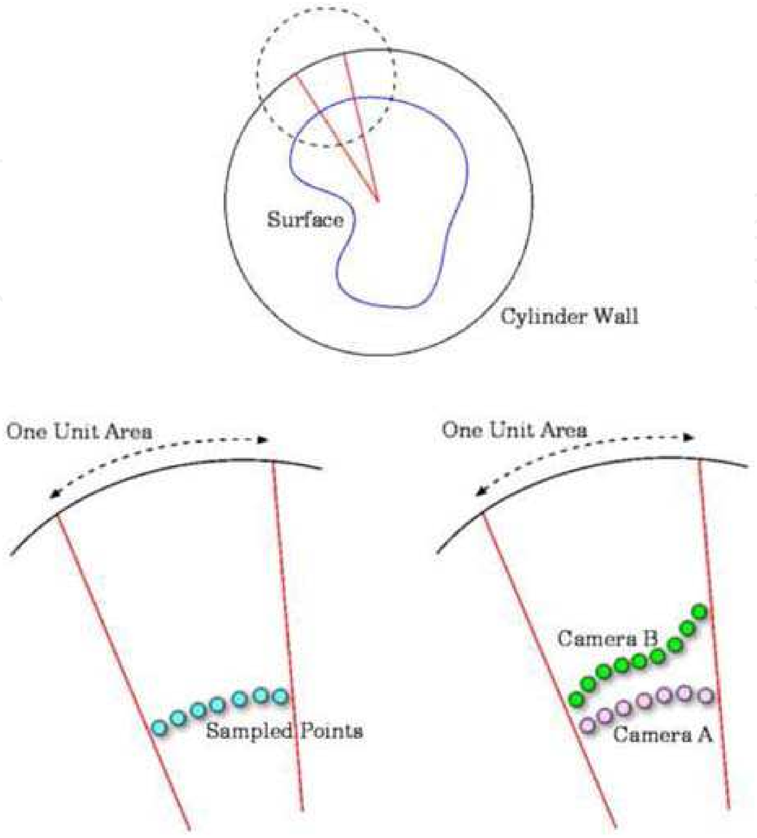


Fig. 8. (top) Overlap in One Unit: These are images observed from y direction. (lower left) One unit stores points that came from only one camera. (lower right) There are points came from two cameras.

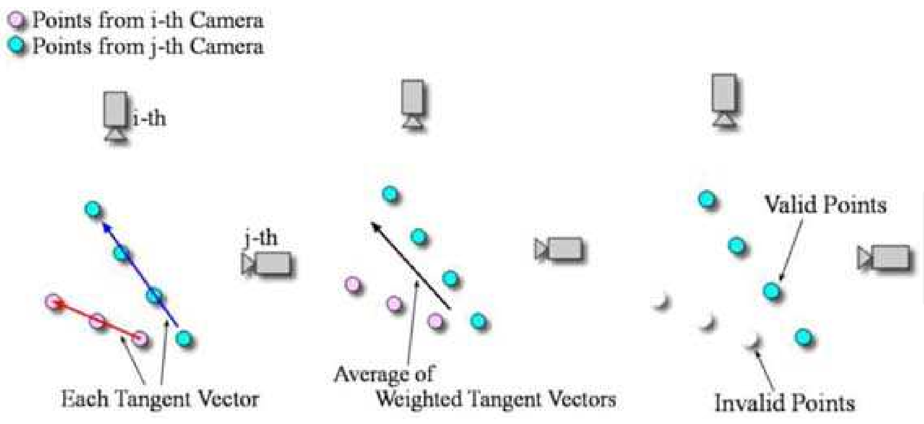


Fig. 9. Procedure of Unit Process: Blue points come from i-th camera and pink points from j-th one. (a) Each tangent vector is acquired. (b) The average of weighted tangent vectors is estimated. After that, inner products are calculated. (c) Points came form camera with minimum inner product are left.

1. Point-Based Rendering

The point-based geometry is proposed for example in (Rusinkiewicz & Levoy, 2000). In our implementation, the client renders the received point could with the rectangle (see Fig.

10(a)). The client estimates the normal vector using the neighbors. By using the estimation of the normal vector and the viewpoint of a user, the size and direction are changed adaptively with respect to rotation by the user. Note that although in the conventional point rendering method it is possible to control the size of the rectangles and avoid holes and it is efficient especially for large scale meshes, in our framework the client avoids the time-consuming step and linearly changes the size of the rectangles based on its depth.

2. Strip-Based Rendering.

Since high accuracy is not required in geometry and computational complexity in the client should be low, we employ the strip-based rendering. The client treats the successive valid pixels in the horizontal line as a band and expresses the pixels by a set of the bands (see Fig.10(b)). The client renders it with the triangle strip. Since the normal vector of each triangle is determined automatically in the steps, we can skip the step of estimating the normals. To enhance the efficiency of rendering, if the valid pixel is replaced by the left pixel in the above simplification algorithm, the pixel is integrated with the left pixel and the pixels are rendered by one strip (Fig.10(c)). And, when scanning the range image in the horizontal direction, if there are invalid pixels, the client calculate the distance of its left and right valid pixels. If the distance is within the threshold, the client fills the blank with its valid pixel. The isolated valid point is rendered by a rectangle. For example, if the pixel is Fig.4(upper left), the rendering results in the strips shown in Fig.10(d).

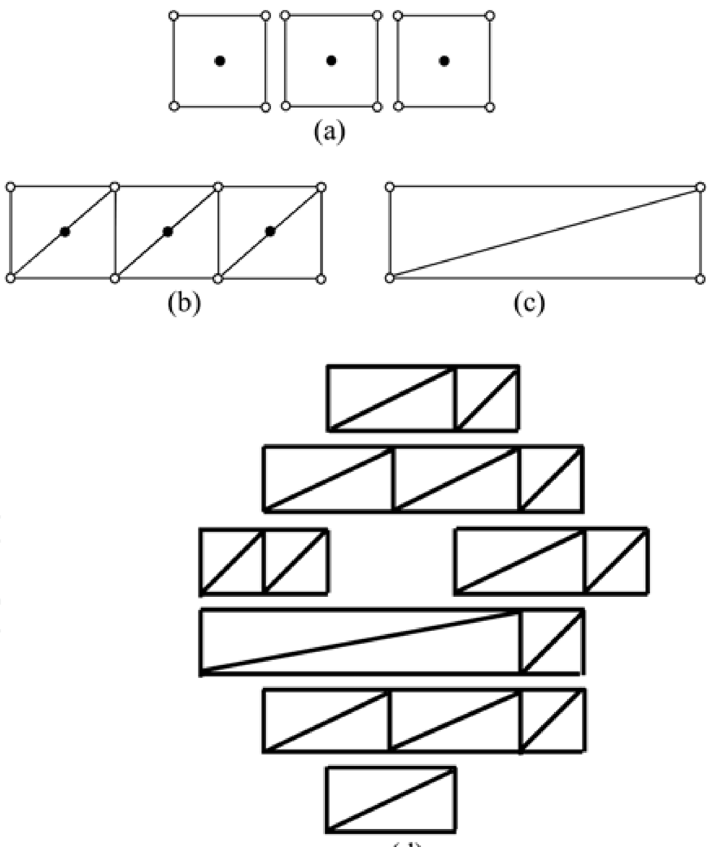


Fig. 10. (a) Rectangle as primitive. The number of vertices is four times as many as the number of the points. (b)Triangle strip as primitive. The number of vertices is about two times as many as the number of the points. (c) Example of merged triangle strip. (d) The example of rendering Fig.4 (upper left).

4. Experimental results

The server consists of five PCs (Pentium IV) and four stereo cameras. We use the commercial stereo cameras (PointGrey). The resolution of colour image and depth image obtained from the camera is 240 x 320. The PCs in the server are connected through Gigabit switch. The server acquires four range images and *ParantPC* integrates them and transmits them to the client. For the client PC, we tested several types such as high end PCs to laptops. The client is located in the same LAN via a router. The example of a man's upper body obtained under these experiment conditions is presented in Fig.11. The system worked at about 9-12fps(frame per sec).

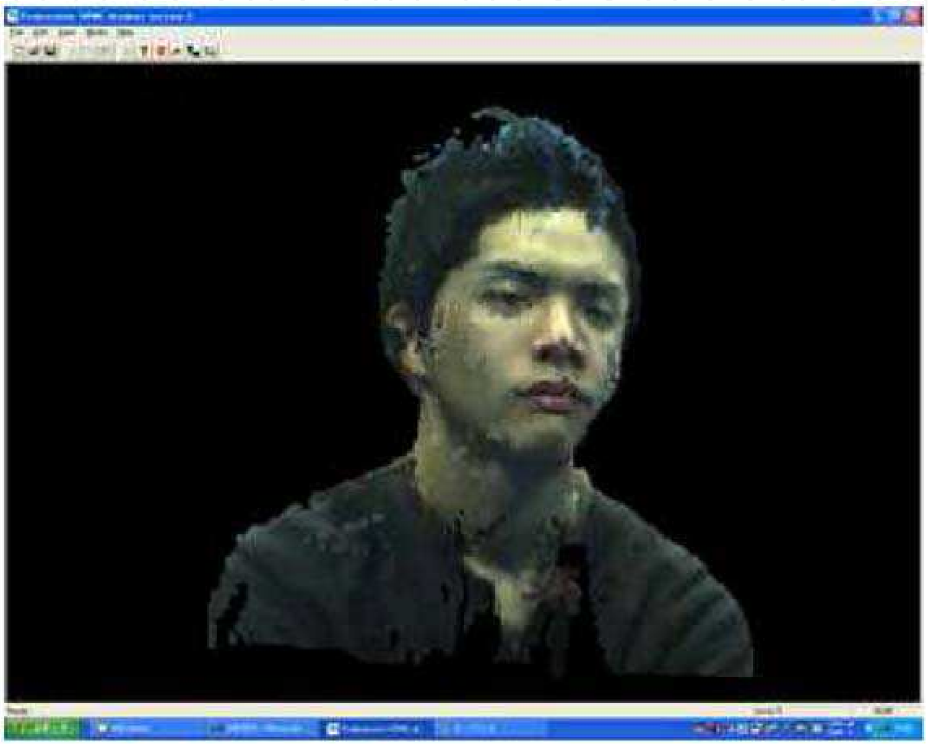


Fig. 11. 3D Viewer

Fig.12 shows the comparison of the point based and strip based rendering. In this experiment, the total data size of 100 frames of the human face was 3.86M bytes without the proposed simplification. On the contrary, our method achieves 1.46M bytes. The data is decreased by 38 %. The average rendering time for one frame was 0.036 sec by the rectangle rendering. On the contrary, with the triangle strip, the rendering time was 0.022 sec. It can be seen from the Fig. 12 that our method performs well without much degradation.
Fig.13 (a) illustrates the 3D model reconstructed simply by integrating four range images. On the other hand, Fig.13 (b) is the model obtained by the integration with our cylindrical method. Blue points from right and left range images are the annoying artifacts on the front of the face in (a). However in (b), these annoying artifacts are removed and the facial expression is finer than (a). The validity of cylindrical method is shown in Fig.14. These pictures are horizontal slice of the face. Each line represents the surface acquired from a camera. Fig.14 (b) is the result by our cylindrical method and (a) is the result of the conventional method. One can see that some overlaps are removed by the cylindrical method in (b). By applying the cylindrical method, the number of points is reduced by 25%.

The computation time to apply the cylindrical method was less than 0.1 second and it did not almost affect the sequences visually.

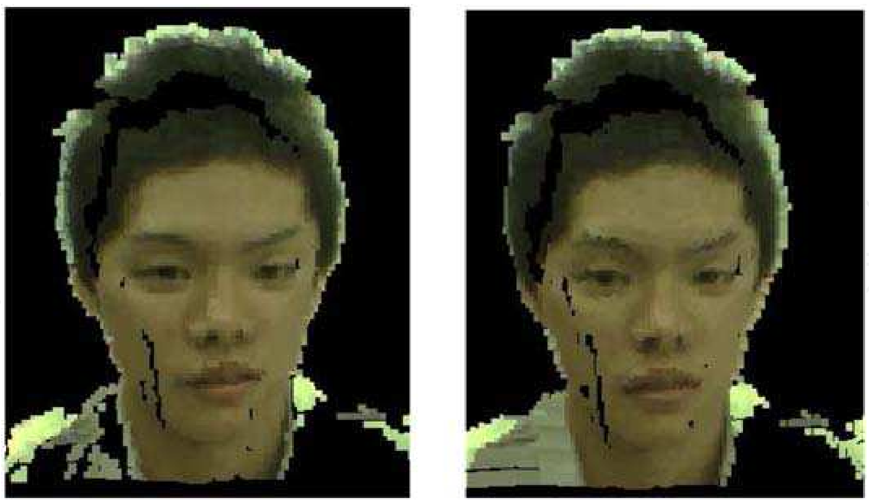
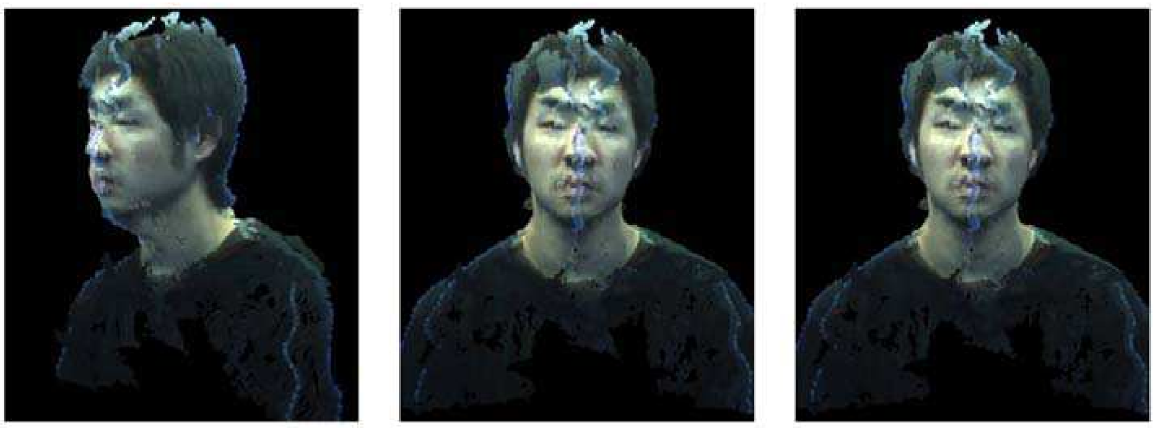
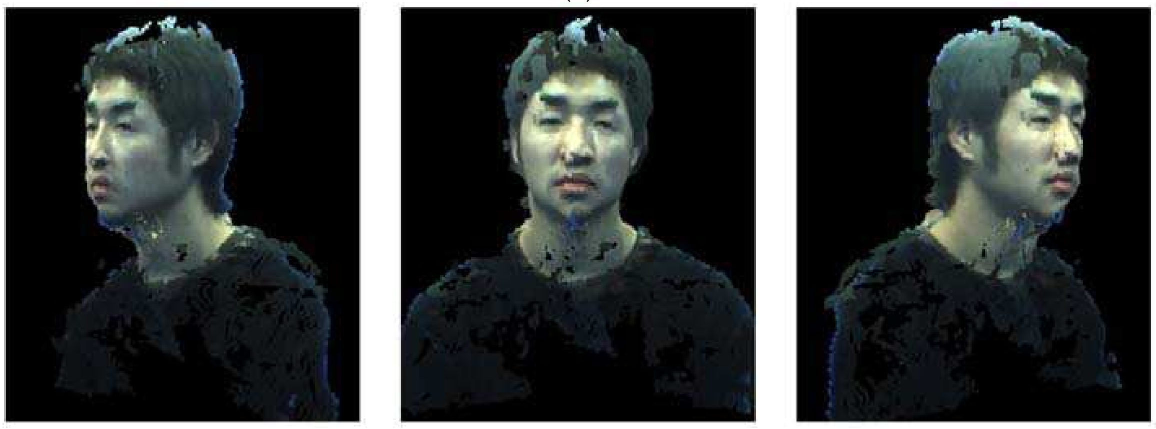


Fig. 12. Rendering example of rectangle as primitive (left). Rendering example of triangle strip as primitive using proposal simplification (right).



(a)



(a)

Fig. 13. Result of Cylindrical Method: These show one frame of the 3D sequence. (a) Integrated four range images. (b) After applying the cylindrical method. Annoying artifacts on the front of face are removed in (b).

Fig. 14. Horizontal Cut Surface of Nose: (left) without the cylindrical method. Overlapping area cannot be in sight like blue line. However overlaps are removed especially front side of face in (right).

5. Conclusion

In this paper, we introduced a real-time 3D imaging system by handling the 3D shape and the colour information as Point Cloud. It is an advantage of our system that it enhances the transmission efficiency by eliminating the connectivity information. Moreover, by applying the cylindrical method to the real-time 3D imaging system, the overlaps and the annoying artifacts are removed efficiently in real-time.

6. References

Gersho, A. & Gray, R. M. (1992) “Vector Quantization and Signal Compression”, Kluwer Academic Publishers.

Hoppe, H. (1996) “Progressive Meshes.”, SIGGRAPH ’96, pp.99-198.

Kanade, T.; Yoshida, A.; Oda, K.; Kano H. & Tanaka, M. (1996) “A Stereo Machine for Video-rate Dense Depth Mapping and Its New Application”, the 15th Computer Vision and Pattern Recognition Conference, pp. 196-202.

Khodakovsky, A., Schröder, P., and Sweldens, W., (2000) “Progressive Geometry Compression.”, SIGGRAPH ’00, pp.271-278.

PointGrey Research, Digiclops, http://www.ptgrey.com/products/stereo.html

Pratt, W. K.; Andrews, H. C. & Kane, J. (1969) “Hadamard Transform Image Coding”, Proc. IEEE, Vol.57, pp.58-68, Jan, 1969.

Rusinkiewicz, S., & Levoy, M., (2000) “Qsplat: A Multiresolution Point Rendering System for Large Meshes”, SIGGRAPH ’00, pp.343-352.

Saito, H.; Baba, S.; Kimura, M.; Vedula, S. & Kaneda, T. (1999) “Appearance-Based Virtual View Generation of Temporally-Varying Events from Multi-Camera Image in the 3D Room”, Second International Conference on 3-D Digital Imaging and Modeling (3DIM99), pp.516-525.

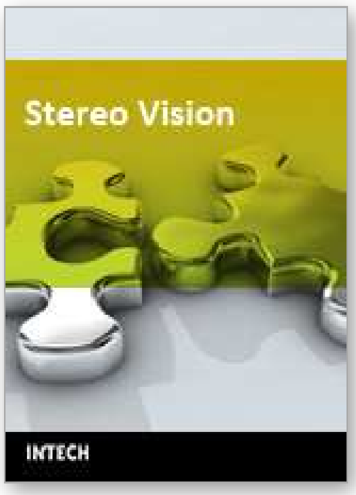
Ueda M.; Arita, D. & Taniguchi, R. (2004) “Real-time Free-viewpoint Video Generation Using Multiple Cameras and a PC-cluster”, Pacific-Rim Conference on Multimedia, pp.418-425.

Wu, X. & Matsuyama, T, (2003) “Real-time Active 3D Shape Reconstruction for 3D Video”, Proc. of 3rd International Symposium on Image and Signal Processing and Analysis, September, pp.186-191.

Waschbüsch, M.; Würmlin, S.; Cotting D., & Gross, M. (2007) “Point-sampled 3D video of real-world scenes”, Journal of Signal Processing: Image Communication, vol. 22, no. 2, 2007, pp. 203-216

Würmlin, S.; Lamboray, E.; Staadt, O. G. & Gross, M. H. (2002) “3D video recorder.”, Proceedings of Pacific Graphics ’02, pages 325-334. IEEE Computer Society Press.

Würmlin, S.; Lamboray, E.; Gross, M. (2004) “3D Video Fragments: Dynamic Point Samples for Real-time Free-Viewpoint Video”, Computers & Graphics, Special Issue on Coding, Compression and Streaming Techniques for 3D and Multimedia Data.



Stereo Vision
Edited by Asim Bhatti

ISBN 978-953-7619-22-0
Hard cover, 372 pages
Publisher InTech
Published online 01, November, 2008
Published in print edition November, 2008