L_063

Multiple-channel Streaming Delivery for HyperOmni Vision System

Yang Haoran† Wang Xiaoqian† Yoshio Iwai† Hajime Nagahara † Masahiko Yachida † Toshiya Suzuki ‡

Abstract— HyperOmni Vision (HOVI) is an imaging system that can capture a surrounding image in whole direction by using a hyperbolic mirror and a conventional CCD camera. We propose a novel method for transferring the image data of HOVI by network more efficiently and more controllable for clients, the proposed method uses multiple channels to deliver multiple streams synchronously. Through this way, the system enables a client to interactively view a different part of HOVI image and also supports the function to change the vision area during the playback. This technique can be widely used in the travel pilot, long distance teaching, long distance medical treatment and so on.

1. INTRODUCTION

With the rapid development of sensor manufacturing techniques, the spatial resolution of image sensors has been increased and an advanced image system called HyperOmni Vision(HOVI)[1] has been invented to capture surrounding images in the whole direction. If HOVI equips a high-resolution camera, we can get excellent-quality images of wide-range view. By watching such images, people can achieve the wide perspective vision of the scene. This creates much more lifelike feeling to the local scene[3].

Streaming media is also a welcome technique that enables the users to play a movie immediately instead of waiting for the entire file to download. The streaming technology in progress makes networks friendly to media and media friendly to networks[2].

This paper proposes a method for delivering these HOVI images by multiple-channel streams in high frame rate through the network. The goal of this research is to develop a data delivery system which is more flexible to transfer the request data and more friendly to the people under the limit resource of network bandwidth.

2. PROBLEM SETTING

The common streaming server system is not suitable for HOVI images transmission. One disadvantage is that even though the people only want to watch a small part of the whole image, the conventional streaming server can not but transfer the whole image. Moreover, delivery of the whole HOVI images with high frame rate will cost a great bandwidth resource especially when HOVI equips a high-resolution camera. Another problem is that the conventional streaming server provides only " "STOP" and " "CONTINUE" function during the period of playing a movie. It is not able to switch from a movie to another movie smoothly. And in current application, we hope the people can view different parts during the whole playing period without any pause.

This paper discusses a method to setup a streaming server which is able to solve above two problems. By using this server, the clients can choose a part of the image that they want to view. Moreover, the clients can interactively change their view to the new part during the playback. the playback.

†Department of System Innovation, Graduate School of Engineering Science, Osaka University.

‡Eizoh Co., LTD, 2-1-10 Nanko-Kita, Suminoe, Osaka 559-0034

3. DIVISION OF IMAGE

The approach proposed by this paper is to divide the whole image to a lot of small ones, and then build these small temporal consecutive images into respective movie files. By this method, the users can choose only a part of the whole image what they want to view.

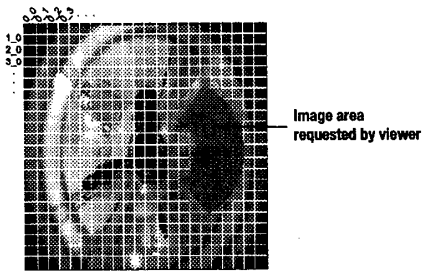

Figure 1: Divided image and requested area

Fig. 1 shows a large HOVI image which is averagely divided into small ones, the green part on it is the part what the people want to view.

There are two ways to divide a HOVI image. One is No Layer Division Method which makes average division of a image. Fig. 1 is an example. Another is Layer Division Method that divides an image into four different size layers and the division size of external layer is four times greater than a neighbor internal layer– as shown in Fig. 2. The reason why Layer Division Method is introduced will be explained in the next session in detail.

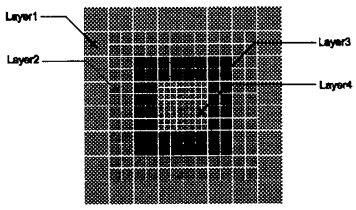

Figure 2: Layer Division Method

We choose MPEG4 as a compressed format to build these divided images into respective movies to prepare for future delivery.

4. SELECTION OF MOVIE FILES BY SERVER

People can decide a field of view by following four parameters as shown in Fig. 3:

$\alpha 1$: horizontal angle of the vision center,

$\alpha 2$: vertical angle of the vision center,

$\alpha 3$: horizontal angle of the field of vision,

$\alpha 4$: vertical angle of the field of vision.

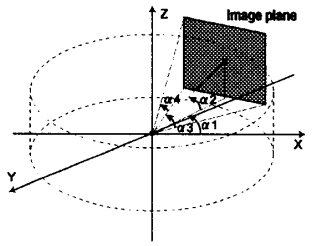

Figure 3: 4 Parameters of view field

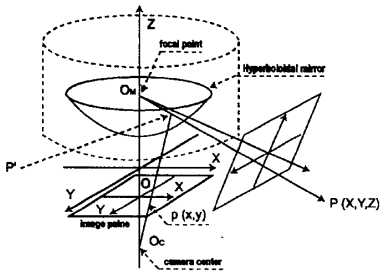

Figure 4: Geometry of HyperOmni Vision

After the request for the field of view is decided by these four parameters, the server should find out all the movie files which contain the view field. However, a camera of HOVI are different from the common camera.

HOVI projects a scene onto the image plane through a hyperboloidal mirror and a lens. We call it mapping hyperboloidal projection.

As shown in Fig. 4, a ray going from the point $P(X, Y, Z)$ in a space toward the focal point of the mirror O_M is reflected by the mirror and passes through the other focal point O_c which intersects the image plane at a point $p(x, y)$. We can use the following equations to transform $P(X, Y, Z)$ to $p(x, y)$:

$$x = X \times f \times \frac{(b^2 - c^2)}{(b^2 + c^2)Z - 2bc\sqrt{X^2 + Y^2 + Z^2}}, \quad (1)$$

$$y = Y \times f \times \frac{(b^2 - c^2)}{(b^2 + c^2)Z - 2bc\sqrt{X^2 + Y^2 + Z^2}}. \quad (2)$$

By using these two equations, the server can find out the movie files that contain the image data of the requested view field.

One feature of the HOVI mapping is that the mapping is not equiangular. The image size of a scene is changed in proportion to the vertical angle of the view field. We can easily understand this feature from Figs. 1 and 5. Therefore, we propose Layer Division Method in this paper.

The clients have received the movie data from the streaming server, conversion should be done in order to change the HOVI images to the common ones before the movies are finally shown to the viewers.

5. MULTIPLE-CHANNEL STREAMING DELIVERY

5.1 Multiple-channel Streaming Technique

Unlike downloading — where you save a complete file locally before you are able to play it — streaming allows you to play a file as it is being downloaded. In this application, we use streaming technique to transfer the multimedia data in real time.

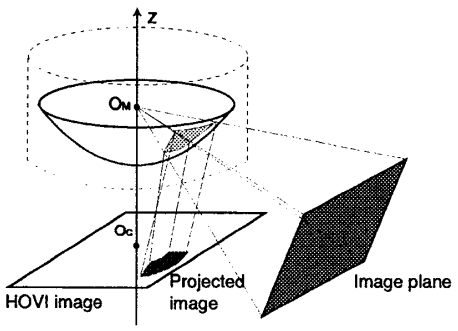

Figure 5: HOVI Projection

In our system, the data of the area requested by the client have been already divided and saved as respective movie files. Therefore, when an client chooses an area of a scene, in fact, he/she chooses a lot of small movies that make up the area jointly. Therefore, multiple streams should be delivered by the server in order to play these movies synchronously. We call such a synchronous multiple streams delivery as Multiple-Channel streaming technique.

5.2 Internet Protocol

The data delivery through the Internet must rely on the Internet protocols.

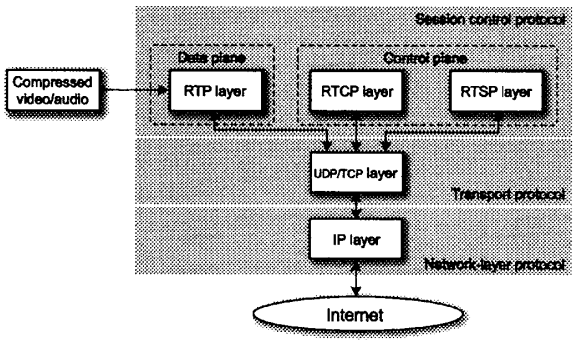

Figure 6: Internet protocols

Let us see Fig. 6. RTSP protocol, plugged onto TCP, is mandated to manage the streaming session[4]. This is the protocol that allows a client to know available streams at a location, get stream parameters and most of all, and allows the client to play, pause and stop a movie: just what you generally expect from a remote control. RTP is a protocol used to transport a multimedia stream to a client[5]. RTP is build upon UDP. Using UDP, RTP sends a packet to the network but does not guarantee to transfer the packets to its destination. RTCP is used to monitor the session. It is mainly used to give the streaming server the reception statistics from the client.

In our system, a client should be able to play multiple movies and change them or stop them whenever he/she wants.

Fig. 7 is the basic RTSP flow. Usually a conventional system that supports to play one stream often uses this RTSP flow to control a whole session between the server and the client.

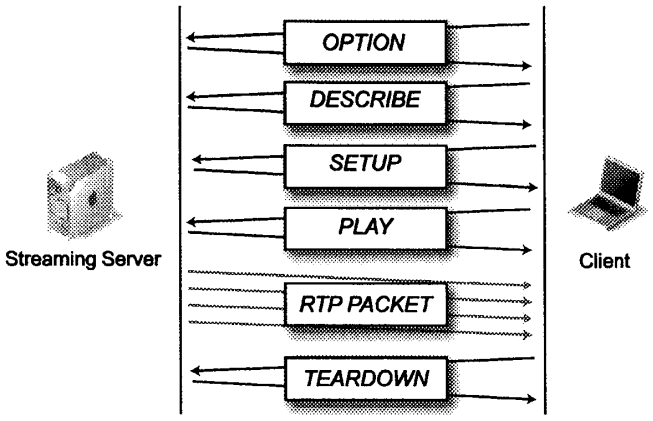

Figure 7: RTSP protocol flow

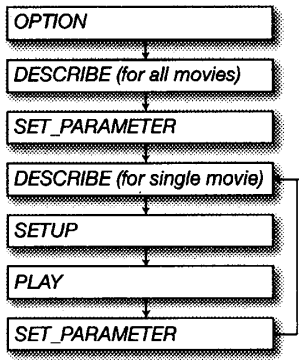

Figure 8: Improved RTSP flow

OPTION is a RTSP method to inquire about methods supported by the server. The DESCRIBE method retrieves the description of a presentation or media object identified by the request URL from a server. The SETUP request for a URL specifies the transport mechanism to be used for the streamed media. PLAY method starts data transmission on a stream allocated via SETUP. The client sends the TEARDOWN method to the server to stop media data delivery. When the TEARDOWN method has been received, the ports associated with this session can now be deallocated.

But this basic RTSP flow is not enough for our system to setup multiple streams in the session. One problem is that– the clients must be able to choose different movie files at any time they want. Therefore, another new method is introduced. This new method is call SET_PARAMETER. In the content of SET_PARAMETER method, it includes the value of four parameters $\alpha 1, \alpha 2, \alpha 3$ and $\alpha 4$ which is requested by the client. This parameter can be sent by the client at any time during the period between DESCRIBE method and TEARDOWN method to play the different movie files. After the server has received the SET_PARAMETER method, it will stop all the current movie streams and make preparation to send the newly requested movies. Another problem is that clients must know about the whole list of divided movie files before they choose the movie files to play. This information will be added into our DESCRIBE method, therefore, now the DESCRIBE method can give description of both the movie list and the single movie. The new RTSP flow is shown in Fig. 8.

We send multiple streams in turn by using RTP protocol after the RTSP method PLAY has been received by the server. We choose Darwin Streaming Server(DSS)[6] as the basic server for developing a platform and add a new module to realize all these functions.

6. EXPERIMENT RESULTS

We do some tests in order to validate the proposed method. And we also want to know the system capability to support the clients. We want let the server to support the clients as many as possible. The size of HOVI image used in the experiment is 2560 × 2560 pixel, we use both Lay Division Method and No Layer Division Method to divide the images. In the No Layer Division Method, images are divided into 1024(32 × 32) small ones. In the Layer Division Method, images are divided as Fig. 2. In each experiment, the four parameters $\alpha 1, \alpha 2, \alpha 3$ and $\alpha 4$ are given in advance. The playback time of movies is 60 seconds.

Experiment 1:

$\alpha 1 = 0°$ $\alpha 2 = 30°$
$\alpha 3 = 30°$ $\alpha 4 = 30°$

No Layer Division Method
Frame rate is 30 frames/sec.
The number of streams of each client is 63.

# of clients	# of total streams	Delay Time
1	63	0.25
4	252	0.35
10	630	0.333
14	882	0.349
17	1071	0.331
18	Too much streams	

Experiment 2:

$\alpha 1 = 0°$ $\alpha 2 = 0°$
$\alpha 3 = 30°$ $\alpha 4 = 30°$

No Layer Division Method
Frame rate is 30 frames/sec.
The number of streams of each client is 22.

# of clients	# of total streams	Delay Time
1	22	0.51
4	88	0.265
10	220	0.235
30	660	0.224
50	1100	0.225
51	Too much streams	

Experiment 3:

$\alpha 1 = 0°$ $\alpha 2 = -30°$
$\alpha 3 = 30°$ $\alpha 4 = 30°$

No Layer Division Method
Frame rate is 30 frames/sec.
The number of streams of each client is 10.

# of clients	# of total streams	Delay Time
1	10	0.082
20	200	0.18
40	400	0.175
80	800	0.184
105	1050	0.187
106	Too much streams	

Experiment 4:

$\alpha 1 = 0°$ $\alpha 2 = 30°$
$\alpha 3 = 30°$ $\alpha 4 = 30°$

Layer Division Method
Frame rate is 30 frames/sec.
The number of streams of each client is 15

# of clients	# of total streams	Delay Time
1	15	0.048
16	240	0.071
36	540	0.085
64	960	0.083
70	1050	0.089
71	Too much streams	

Experiment 5:

$\alpha 1 = 0°$ $\alpha 2 = 0°$
$\alpha 3 = 30°$ $\alpha 4 = 30°$

Layer Division Method
Frame rate is 30 frames/sec.
The number of streams of each client is 8.

# of clients	# of total streams	Delay Time
1	8	0.036
20	160	0.053
40	320	0.057
80	640	0.059
126	1008	0.062
127	Too much streams	

Experiment 6:

$\alpha 1 = 0°$ $\alpha 2 = -30°$
$\alpha 3 = 30°$ $\alpha 4 = 30°$

Layer Division Method
Frame rate is 30 frames/sec.
The number of streams of each client is 8.

# of clients	# of total streams	Delay Time
1	8	0.033
20	160	0.049
40	320	0.059
80	640	0.051
129	1024	0.054
130	Too much streams	

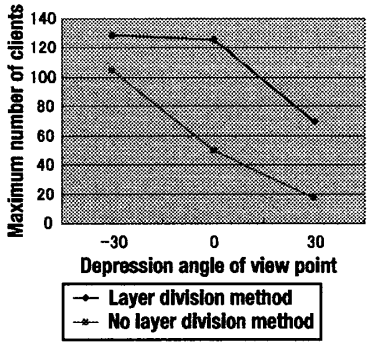

Figure 9: Comparison of supported client number

6.1 Analysis of Experiment

From these experiments, the clients are able to choose the area of the HOVI image. The system using multiple-channel streaming delivery can transfer multiple streams synchronously. The clients can also change the area during the whole playback by using SET_PARAMETER RTSP method.

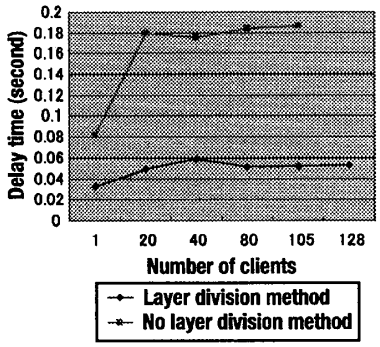

Figure 10: Comparison of delay time

We find out that the maximum number of streams that our streaming server supports to play is around 1050, and comparing between two Division Methods, the delay time of streaming delivery of two division method is nearly same, as shown in Fig. 10. But when the system delivers the same part of the HOVI image, the maximum supported number of clients of the Layer Division Method is much larger than that of No Layer Division Method, as shown in Fig. 9. Therefore, Layer Division Method of the HOVI image will be proposed in order to make the streaming server support more clients.

7. CONCLUSION

This paper has proposed a method to construct a streaming server to deliver multiple-channel streams to support high frame rate transmission of HOVI images. This system also enables clients to interactively choose the area of the image whenever they want during the playback of the whole movie.

Acknowledgment

A part of this research is supported by "Key Technology Research Promotion Program" of the National Institute of Information and Communication Technology.

References

[1] K. Yamazawa, Y. Yagi and M. Yachida: "New real-time omnidirectional image sensor with hyperboloidal mirror," *Proc. The 8th Scandinavian Conf. on Image Analysis*, Vol. 2, pp. 1381–1387, 1993.

[2] Y. Wang, J. Ostermann, Y.-Q. Zhang, "VIDEO PROCESSING AND COMMUNICATIONS," pp. 519–546, Prentice Hall, New Jersoy, 1999.

[3] T. Yamamoto and M. Doi, " Panovl: Panoramic movie system for real-time network transmission," *In IEEE Workshop on Multimedia Signal Processing,* pp. 389–394, 2001.

[4] H. Schulzrinne, A. Rao, and R. Lanphier, "Real Time Streaming Protocol(RTSP)," RFC 2326, Internet Engineering Task Force, April 1998.

[5] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson, "RTP: A Transport Protocol for RealTime Applications," RFC 1889, Internet Engineering Task Force, January 1996.

[6] Apple Computer, "QuickTime Streaming Server & Darwin Streaming Server Administrator's Guide," Apple Computer, Inc., November 2002.