Panoramic Video Coding Using Affine Motion Compensated Prediction

Zheng Jiali, Zhang Yongdong, Shen Yanfei, and Ni Guangnan

Key Laboratory of Intelligent Information Processing
Institute of Computing Technology, Chinese Academy of Sciences, Beijing, 100080, China
{zhengjiali, zhyd, syf}@ict.ac.cn, ngn@public.bta.net.cn

Abstract. This paper proposes an affine motion compensated prediction (AMCP) method to predict the complex changes between the successive frames in panoramic video coding. A panoramic video is an image-based rendering (IBR) technique [1] which provides users with a large field of view (e.g. 360 degree) on surrounding dynamic scenes. It includes not only the translational motions but also the non-translational motions, such as zooming and rotation etc. However, the traditional motion compensated prediction is a translational motion compensated prediction (TMCP) which cannot predict non-translational changes between panoramic images accurately. The AMCP can model the non-translational motion effects of panoramic video accurately by using six motion coefficients which are estimated by Gauss Newton iterative minimization algorithm [2]. Simulated results show that the gain of coding performance is up to about 1.3 dB when using AMCP compared with TMCP in panoramic video coding.

Keywords: Affine Motion Compensated Prediction (AMCP), Translational Motion Compensated Prediction (TMCP), panoramic video.

1 Introduction

With the increased demand for users' better experience in interactive applications such as virtual walkthrough, tele-presence and autonomous vehicles, virtual reality techniques are becoming popular. Image-based modeling and image-based rendering techniques have been received more attention as powerful alternatives to traditional geometry-based techniques for image synthesis. They rely on the characteristics of the plenoptic function [3] and use images rather than geometry as primitiveness for rendering novel views.

A panoramic video is the representation of such techniques that visualize the photorealistic and interactive environment. A panoramic mosaic is a high-resolution image obtained by projecting a series of images (registering and stitching), taken on a plane when a camera is moving along a given axis, on a cylindrical or spherical surface. [4]-[10] As the following equations [4] show, a cylindrical panorama can be obtained by mapping the world coordinates $P = (x, y, z)$ into the cylindrical screen coordinates (θ, v):

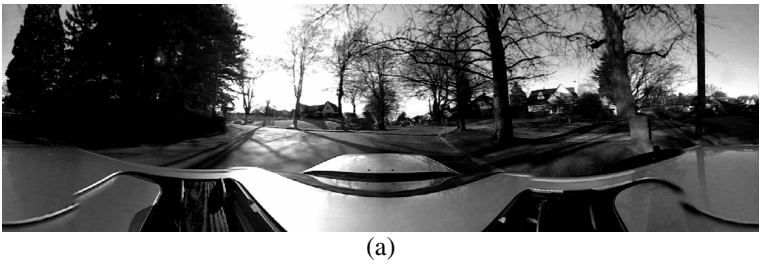
N. Sebe, Y. Liu, and Y. Zhuang (Eds.): MCAM 2007, LNCS 4577, pp. 112–121, 2007.

$$\begin{aligned} \theta &= \tan^{-1}(x/z) \\ v &= y/(\sqrt{x^2+z^2}) \end{aligned} . \tag{1}$$

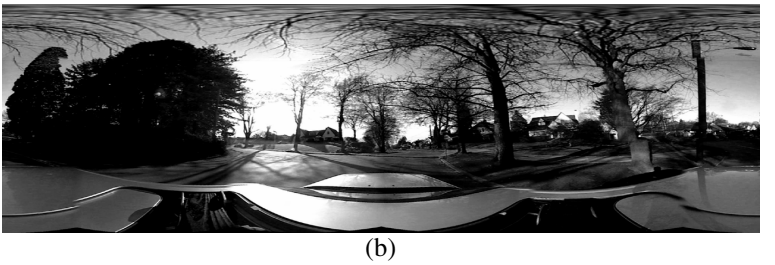
where θ is the panning angle and v is the scan-line. Similarly, a spherical panorama can be obtained by mapping the world coordinates into the spherical screen coordinates (θ, Φ) using the equations [4] below:

$$\begin{aligned} \theta &= \tan^{-1}(x/z) \\ \Phi &= \tan^{-1}(y/\sqrt{x^2+z^2}) \end{aligned} . \tag{2}$$

Fig 1 shows the samples of the cylindrical panorama and the spherical panorama.



(a)



(b)

Fig. 1. Part of a panoramic mosaic mapped into (a) the cylindrical screen coordinates and (b) the spherical screen coordinates

Because panorama is obtained by stitching several images together, its resolution is usually very high (e.g. 1920×352). Considering such a panoramic video sequence with 25 frame/sec would occupy about 48M/s of transmission bandwidth, it is unacceptable in the interactive applications. How to compress the huge amount of panoramic video data efficiently while maintain the satisfying image quality, is a key point to improve the users' smooth interactive experience.

Moreover, a serious problem existing in panoramic video coding is that the encoder has to deal with a large amount of non-translational motions. As discussed above, to display a large field of view (e.g. 360 degree) and reduce the overlapped redundancy of the stitched images, a panoramic mosaic is mapped into a cylindrical or spherical surface. Correspondingly, a portion of horizontal straight lines in

cylindrical panorama or spherical panorama would become curved according to (1) and (2), which results in the warping changes between panoramas. Meanwhile, the panoramic images are digitally warped on-the-fly to simulate camera panning and zooming [5]. This is another important factor contributing to the increasing of the complex motions occurring in panoramic video.

To compress the huge amount of motion-video data effectively, a well known Inter-Frame predictive coding scheme named motion compensated prediction which exploiting the similarities between frames of a video signal, is widely used in existing video compression standards, such as ITU-T Recommendations H.261, H.263, H.264 and ISO MPEG-1, MPEG-2, MPEG-4. However, this motion compensated prediction scheme is based on a translational motion model which assumes the object motions as the rigid motions without any warping. That is to say, this translation motion compensated prediction (TMCP) cannot model accurately the warping motions mentioned above in panoramic video coding.

To overcome this problem, an affine motion compensated prediction (AMCP) is proposed in this paper. The AMCP is based on the principle of the affine motion vector field [11]-[17]. Here, we propose a six-parameter affine motion model to describe the motion effects of translation, zooming and rotation correspondingly in panoramic video coding. By using Gauss Newton iterative minimization algorithm to minimize the mean square prediction error, the low prediction error can be achieved and the values of motion coefficients for each block in current frame can be estimated.

The rest of this paper is organized as following: section 2 describes affine motion compensated prediction scheme in detail. The simulated results for evaluating the quality of the prediction frames and the coding performance by using proposed AMCP compared with TMCP in panoramic video coding, are presented in section 3. Section 4 gives conclusions and future works.

2 Affine Motion Compensated Prediction

The operating principle of motion compensated video coders is to minimize the prediction error $E_n(x, y)$, that is, the difference between the nth frame being coded $I_n(x, y)$, called the current frame, and a prediction frame $P_n(x, y)$:

$$E_n(x, y) = I_n(x, y) - p_n(x, y) \ . \tag{3}$$

The prediction frame $P_n(x, y)$ is built using pixel values of a reference frame denoted by $R_n(x, y)$ and the motion vectors of pixels between the current frame and the reference frame using the equation:

$$p_n(x, y) = R_n[x + \Delta x(x, y), y + \Delta y(x, y)] \ . \tag{4}$$

where the pair of values $[\Delta x(x, y), \Delta y(x, y)]$ represent the motion vector of the pixel in location (x, y) in the current frame. By combining the equation (1) and (2), the prediction error $E_n(x, y)$ can be calculated as the equation below shows:

$$E_n(x,y) = I_n(x,y) - R_n[x+\Delta x(x,y), y+\Delta y(x,y)] \ . \tag{5}$$

In the international standards, such as ISO MPEG-1, MPEG-2, MPEG-4 and ITU-T Recommendations H.26L, the used motion compensated prediction is TMCP which is based on a translational motion model. It requires only two coefficients a_0, b_0 to describe horizontal and vertical translational motion. The motion vector [$\Delta x(x, y)$, $\Delta y(x, y)$] are given by the following equations:

$$\begin{aligned} \Delta x(x, y) &= a_0 \\ \Delta y(x, y) &= b_0 \end{aligned} \ . \tag{6}$$

Since this traditional motion model only take into account the translational motion, it cannot describe accurately the complex motions in panoramic videos which contain translational motions and non-translational motions. To deal with this problem, this paper proposes an AMCP based on an affine motion model which is given by the following equations:

$$\begin{aligned} \Delta x(x, y) &= a_0 + a_1 x + a_2 y \\ \Delta y(x, y) &= b_0 + b_1 x + b_2 y \end{aligned} \ . \tag{7}$$

where the function of a_0, b_0 model the translational effect, a_1x, b_2y model the zooming effect and a_2y, b_1x model the rotation effect. As discussed above, these are the main motion effects existing in the panoramic video.

As (7) shows, to obtain the values of motion vector [$\Delta x(x, y)$, $\Delta y(x, y)$] in affine motion model, the key point is to estimate the values of motion coefficients (a_0,a_1,a_2,b_0,b_1,b_2). The rest of this section gives the calculation of motion coefficients in detail.

Let C = (c_0,c_1,c_2,c_3,c_4,c_5) = (a_0,a,a_2,b_0,b_1,b_2), then C can be calculated by minimizing the mean square prediction error which is given below:

$$\begin{aligned} &\sum_{x\in m, y\in m} DPD^2(x,y,c) = \sum_{x\in m, y\in m} E_n^{\,2}(x,y,c) = \\ &\sum_{x\in m, y\in m} (I_n(x,y) - R_n[x+\Delta x(x,y,c), y+\Delta y(x,y,c)])^2 \end{aligned} \ . \tag{8}$$

where m denotes the coded block in current frame.

For performing the full search of all possible coefficients combinations is very computational, an alternative method which uses Gauss Newton iterative minimization algorithm to obtain a set of motion coefficients for the coded block, is summarized in the following:

$$G\delta = g \ . \tag{9}$$

where $\delta = C^{new} - C^{old}$, C^{old} denote the motion coefficients input of Gauss Newton iteration and C^{new} denote the motion coefficients output of Gauss Newton iteration. G is a 6×6 positively semi-definite symmetric matrix which estimates second derivative

of the $DPD^2(x, y, c)$ with respect to the coefficients C. G has elements as following equations show:

$$G(c)=\frac{DPD^2(x,y,c)}{\partial c_i \partial c_j}=\frac{DPD(x,y,c)}{\partial c_i}\cdot\frac{DPD(x,y,c)}{\partial c_j} \quad . \tag{10}$$

I= 0~2 and j = 0~2
i.e.

$$G(c)=\begin{bmatrix} \frac{\partial^2 DPD(x,y,c)}{\partial c_0 \partial c_0} & \frac{\partial^2 DPD(x,y,c)}{\partial c_0 \partial c_1} & \frac{\partial^2 DPD(x,y,c)}{\partial c_0 \partial c_2} & \frac{\partial^2 DPD(x,y,c)}{\partial c_0 \partial c_3} & \frac{\partial^2 DPD(x,y,c)}{\partial c_0 \partial c_4} & \frac{\partial^2 DPD(x,y,c)}{\partial c_0 \partial c_5} \\ \frac{\partial^2 DPD(x,y,c)}{\partial c_1 \partial c_0} & \frac{\partial^2 DPD(x,y,c)}{\partial c_1 \partial c_1} & \frac{\partial^2 DPD(x,y,c)}{\partial c_1 \partial c_2} & \frac{\partial^2 DPD(x,y,c)}{\partial c_1 \partial c_3} & \frac{\partial^2 DPD(x,y,c)}{\partial c_1 \partial c_4} & \frac{\partial^2 DPD(x,y,c)}{\partial c_1 \partial c_5} \\ \frac{\partial^2 DPD(x,y,c)}{\partial c_2 \partial c_0} & \frac{\partial^2 DPD(x,y,c)}{\partial c_2 \partial c_1} & \frac{\partial^2 DPD(x,y,c)}{\partial c_2 \partial c_2} & \frac{\partial^2 DPD(x,y,c)}{\partial c_2 \partial c_3} & \frac{\partial^2 DPD(x,y,c)}{\partial c_2 \partial c_4} & \frac{\partial^2 DPD(x,y,c)}{\partial c_2 \partial c_5} \\ \frac{\partial^2 DPD(x,y,c)}{\partial c_3 \partial c_0} & \frac{\partial^2 DPD(x,y,c)}{\partial c_3 \partial c_1} & \frac{\partial^2 DPD(x,y,c)}{\partial c_3 \partial c_2} & \frac{\partial^2 DPD(x,y,c)}{\partial c_3 \partial c_3} & \frac{\partial^2 DPD(x,y,c)}{\partial c_3 \partial c_4} & \frac{\partial^2 DPD(x,y,c)}{\partial c_3 \partial c_5} \\ \frac{\partial^2 DPD(x,y,c)}{\partial c_4 \partial c_0} & \frac{\partial^2 DPD(x,y,c)}{\partial c_4 \partial c_1} & \frac{\partial^2 DPD(x,y,c)}{\partial c_4 \partial c_2} & \frac{\partial^2 DPD(x,y,c)}{\partial c_4 \partial c_3} & \frac{\partial^2 DPD(x,y,c)}{\partial c_4 \partial c_4} & \frac{\partial^2 DPD(x,y,c)}{\partial c_4 \partial c_5} \\ \frac{\partial^2 DPD(x,y,c)}{\partial c_5 \partial c_0} & \frac{\partial^2 DPD(x,y,c)}{\partial c_5 \partial c_1} & \frac{\partial^2 DPD(x,y,c)}{\partial c_5 \partial c_2} & \frac{\partial^2 DPD(x,y,c)}{\partial c_5 \partial c_3} & \frac{\partial^2 DPD(x,y,c)}{\partial c_5 \partial c_4} & \frac{\partial^2 DPD(x,y,c)}{\partial c_5 \partial c_5} \end{bmatrix} . \tag{11}$$

Corresponding the elements $\Delta x, \Delta y$ in the prediction error function $DPD(x, y, c)$, there are two equations to calculate $\frac{\partial DPD(x, y, c)}{\partial c_i}$:

$$\frac{\partial DPD(x, y, c)}{\partial c_i}=\frac{\partial DPD(x, y, c)}{\partial \Delta x}\cdot\frac{\partial \Delta x}{\partial c_i} \quad . \tag{12}$$

$$\frac{\partial DPD(x, y, c)}{\partial c_i}=\frac{\partial DPD(x, y, c)}{\partial \Delta y}\cdot\frac{\partial \Delta y}{\partial c_i} \quad . \tag{13}$$

As defined above, $C = (c_0,c_1,c_2,c_3,c_4,c_5) = (a_0,a_1,a_2,b_0,b_1,b_2)$, combined with (7), the values of $\frac{\partial \Delta x}{\partial c_i}, \frac{\partial \Delta y}{\partial c_i}$ can be calculated as shown in the following equations:

$$\frac{\partial \Delta x}{\partial c_0}=\frac{\partial \Delta y}{\partial c_3}=1, \frac{\partial \Delta x}{\partial c_1}=\frac{\partial \Delta y}{\partial c_4}=x, \frac{\partial \Delta x}{\partial c_2}=\frac{\partial \Delta y}{\partial c_5}=y \quad . \tag{14}$$

Let *dx* denote the $\frac{\partial DPD(x,y,c)}{\partial \Delta x}$ and *dy* denote the $\frac{\partial DPD(x,y,c)}{\partial \Delta y}$, following 6×6 positively semi-definite symmetric matrix can be obtained by substituting (12), (13) and (14) into (11):

$$G(c)=\begin{bmatrix} dx^2 & dx^2\cdot x & dx^2\cdot y & dx\cdot dy & dx\cdot dy\cdot x & dx\cdot dy\cdot y \\ dx^2\cdot x & dx^2\cdot x^2 & dx^2\cdot x\cdot y & dx\cdot dy\cdot x & dx\cdot dy\cdot x^2 & dx\cdot dy\cdot x\cdot y \\ dx^2\cdot y & dx^2\cdot x\cdot y & dx^2\cdot y^2 & dx\cdot dy\cdot y & dx\cdot dy\cdot x\cdot y & dx\cdot dy\cdot y^2 \\ dy\cdot dx & dy\cdot dx\cdot x & dy\cdot dx\cdot y & dy^2 & dy^2\cdot x & dy^2\cdot y \\ dy\cdot dx\cdot x & dy\cdot dx\cdot x^2 & dy\cdot dx\cdot x\cdot y & dy^2\cdot x & dy^2\cdot x^2 & dy^2\cdot x\cdot y \\ dy\cdot dx\cdot y & dy\cdot dx\cdot x\cdot y & dy\cdot dx\cdot y^2 & dy^2\cdot y & dy^2\cdot x\cdot y & dy^2y^2 \end{bmatrix}. \quad (15)$$

In (9), g is a vector of size 6 with elements as the following equation show:

$$g(c)=DPD(x,y,c)\cdot\frac{\partial DPD(x,y,c)}{\partial c_i}, i=0\sim 5\ . \quad (16)$$

Similarly, Let *dx* denote $\frac{\partial DPD(x,y,c)}{\partial \Delta x}$, *dy* denote $\frac{\partial DPD(x,y,c)}{\partial \Delta y}$ and E_n denote *DPD(x, y, c)*, (16) can be transformed into the 6×1 matrix below by combining (12), (13) and (14):

$$g(c)=\begin{bmatrix} dx\cdot E_n \\ dx\cdot x\cdot E_n \\ dx\cdot y\cdot E_n \\ dy\cdot E_n \\ dy\cdot x\cdot E_n \\ dy\cdot y\cdot E_n \end{bmatrix}. \quad (17)$$

For δ in (9), it is a 6×1 matrix which can be represented by the following matrix:

$$\delta=\begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \end{bmatrix}. \quad (18)$$

Finally, (9) can be solved by substitute (15), (17) and (18) into (9) and δ is outputted as a result. For $\delta = C^{new}-C^{old}$, a new set of motion coefficients C^{new} are given by the following equation:

$$C^{new} = \delta + C^{old} . \tag{19}$$

When a Gauss-Newton iteration step has been done, the method deciding whether the outputted motion coefficients are the final results, is to evaluate the prediction error. If the difference between mean square prediction error after and before the step is smaller than a threshold, stop the Gauss-Newton iteration and output C^{new} as the final result. Where, the threshold is set as 0.1. If not, look C^{new} as C^{old}, input it to the next Gauss-Newton iteration step to obtain new set of motion coefficients. When final $C^{new} = (c_0,c_1,c_2,c_3,c_4,c_5) = (a_0,a_1,a_2,b_0,b_1,b_2)$ have been obtained, substitute it to (7) to calculate the values of motion vector $[\Delta x(x, y), \Delta y(x, y)]$ for each pixel in the block m.

3 Simulated Results

In the simulated tests, we integrate the proposed technique into the H.264/AVC reference software JM90 [18]. The test sequences include panoramic video sequences [19] Village and Hall. Both sequences are 1920×352 format with 25 fps and coded with GOP structure "IPPPPP…". The Peak Signal to Noise Ratios (PSNR) presented in the results is calculated by the following equation:

$$SNR = 10\log_{10}\frac{M \cdot N \cdot 255^2}{\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}(I_n(x, y)-\tilde{I}_n(x, y))^2} . \tag{20}$$

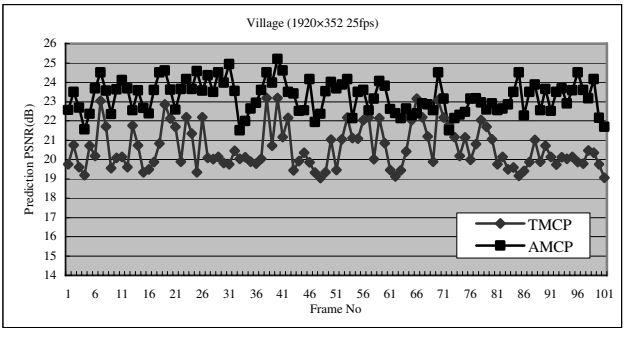
where, $M \times N$ is the image size, $I_n(x, y)$ is the original frame luminance function and $\tilde{I}_n(x, y)$ is the prediction frame luminance function when calculates the prediction PSNR which is presented in Fig 2. The prediction PSNR gives an indication of how similar the prediction is to the original frame. The higher the prediction PSNR is, the better is the prediction. In Fig 3, the reconstruction PSNR presented in rate-distortion curves, also is calculated by using (20), where $\tilde{I}_n(x, y)$ is the reconstruction frame luminance function.

It is clearly observed from Fig 2 that the prediction images PSNR are improved highly when panoramic video coding using AMCP compared with TMCP. Table 1 shows the allocation of bits between coding motion vectors and coding prediction error, as well as average prediction PSNR. Although the affine motion model using more motion coefficients than the translational motion model, which results in a slight increase of coding bits for motion vectors, the motion compensated prediction is more accurate and the prediction PSNR is better. To evaluate the coding performance of proposed technique compared with traditional technique, the rate-distortion curves are

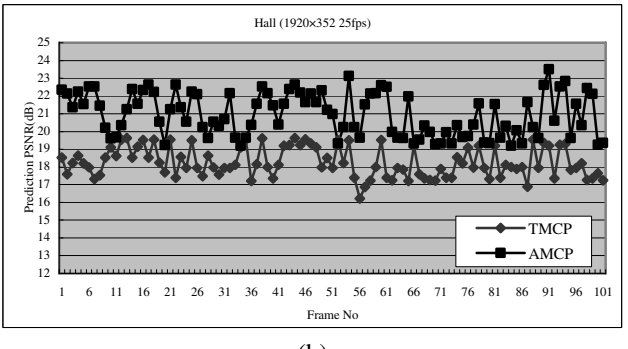
presented in Fig 3. The average gain with the proposed AMCP is up to 1.36 dB in Village sequence and 1.31 dB in Hall sequence.

4 Conclusion

The proposed affine motion model is a six-parameter motion model which more accurately approximates the changes between panoramic frame by predicting not only translational motions, but also non-translational motions, such as zooming and rotation. When integrated with AMCP, the better quality of prediction frame can be obtained and the panoramic videos are compressed more effectively than using traditional TMCP. Considering the affine motion model achieves more accurate motion compensated prediction at the cost of more motion coefficients needed to transmit to decoder, how to decrease the number of these motion coefficients and quantize them effectively according to the motion characteristics of the panoramic video, while remain the coding performance without degenerating highly, will be the issue being worth studying in future work.



(a)

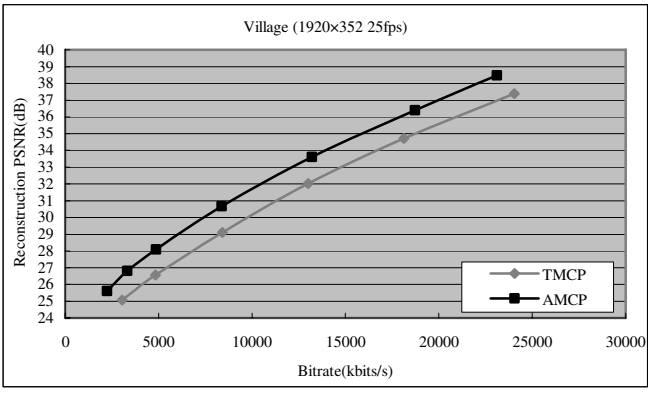


(b)

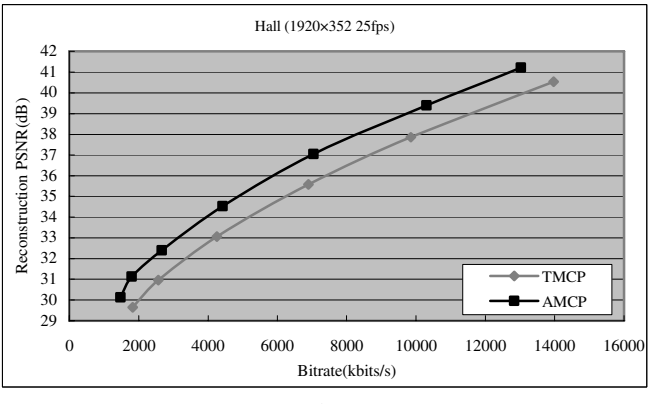
Fig. 2. Prediction PSNR for each P frame in coding panoramic video sequences (a) "Village" and (b) "Hall" by using AMCP compared with TMCP

Table 1. Average bits spent on coding motion vectors and prediction error and average prediction PSNR of each P frame by using AMCP compared with TMCP in panoramic video coding

Sequence	Motion compensated prediction method	Motion Vector (kbits)	Prediction error (kbits)	Prediction PSNR (dB)
Village	TMCP	25.62	611.25	21.5
	AMCP	58.09	582.36	23.86
Hall	TMCP	14.6	183.38	19.86
	AMCP	31.3	171.6	21.83



(a)



(b)

Fig. 3. Rate-distortion curve of coding panoramic video (a) "Village" and (b) "Hall" by using AMCP compared with TMCP

Acknowledgments

This work is partially supported by the National Nature Science Foundation of China under the contract No.60302028, the Beijing Science and Technology Planning Program of China under the contract No.D0106008040291.

References

1. Shum, H.Y., Kang, S.B., Chan, S.C.: Survey of image-based representations and compression techniques. IEEE Transaction on Circuit and Systems for Video Technology 13(11), 1020–1037 (2003)
2. Fletcher, R.: Practical Methods for Optimization, 2nd edn. chapter 3 and chapter 6, John Wiley & Sons, Chichester, (1987)
3. Wong, T.T., Fu, C.W., Heng, P.A., Leung, C.-S.: The plenoptic illumination function. IEEE Transaction on Multimedia 4(3), 361–371 (2002)
4. Szeliski, R., Shum, H.Y.: Creating full view panoramic image mosaics and texture-mapped models. Computer Graphics (SIGGRAPH'97), pp. 251–258 (August 1997)
5. Chen, S.E.: Quick Time VR – an image-based approach to virtual environment navigation. Computer Graphics (SIG-GRAPH'95) , pp. 29–38 (August 1995)
6. Ng, K.T., Chan, S.C., Shum, H.Y.: On the data compression and transmission aspects of panoramic video. IEEE Transaction on Circuit and Systems for Video Technology 15(1), 82–95 (2005)
7. Li, Y., Shum, H.Y., Tang, C.K., Szeliski, R.: Stereo reconstruction from multiperspective panorama. IEEE Transaction on Pattern Analysis and Machine Intelligence 26, 45–62 (2004)
8. Yamada, K., Ichikawa, T., Naemura, T., Aizawa, K., Saito, T.: Generation of a disparity panorama using a 3-camera capturing system. IEEE Transaction on Image Process 2, 772–775 (2000)
9. Tang, W.K., Wong, T.T., Heng, P.A.: A system for real-time panorama generation and display in tele-immersive applications. IEEE Transaction on Multimedia 7(2), 280–292 (2005)
10. Schechner, Y.Y., Nayar, S.K.: Generalized mosaicing: polarization panorama. IEEE Transaction on Pattern Analysis and Machine Intelligence 27(4), 631–636 (2005)
11. Wiegand, T., Steinbach, E., Girod, B.: Affine multipicture motion-compensated prediction. IEEE Transaction on Circuit and Systems for Video Technology 15(2), 197–209 (2005)
12. Karczewicz, M., Nieweglowski, J., Haavisto, P.: Video coding using motion compensation with polynomial motion vector fields. Signal Process: Image Commun 10, 63–91 (1997)
13. Utgikar, A., Badawy, W., Seetharaman, G., Bayoumi, M.: Affine scheme in mesh-based video motion compensation.In: Proc. IEEE Workshop SIPS 2003, pp. 159–164 (August 2003)
14. Huang, J.C., Hsieh, W.S.: Automatic feature-based global motion estimation in video sequences. IEEE Transaction on Consum Electron 50(3), 911–915 (2004)
15. Servais, M., Vlachos, T., Davies, T.: Affine motion compensation using a content-based mesh. In: Proc. IEEE Vis. Image Signal Process, vol. 152(4), pp. 415–423 (August 2005)
16. Keller, Y., Averbuch, A.: Fast motion estimation using bidirectional gradient methods. IEEE Transaction on Image Process, 13(8), 1042–1054 (2004)
17. Gahlots, A., Arya, S., Ghosh, D.: Object-based affine motion estimation. In: Proc. Conf. TENCON 2003, vol. 13, pp. 1343–1347 (October 2003)
18. http://iphome.hhi.de/suehring/tml/download
19. ftp://ftp.tnt.uni-hannover.de