DEVELOPMENT OF A SIMPLE FREE VIEWPOINT VIDEO SYSTEM

Seokhwan Jo, Dohyun Lee, Yoonseob Kim, and Chang D. Yoo

Div. of EE, School of EECS, KAIST,
373-1 Guseong-dong, Yuseong-gu, Daejeon, 305-701, Korea
{antiland00, ichigoichie, netstorm}@kaist.ac.kr, cdyoo@ee.kaist.ac.kr

ABSTRACT

A simple free viewpoint video system which is able not only to display user-specified views at arbitrary angle but also to efficiently stream the necessary video over a network is described. Virtual view synthesis based on projection theory was used to obtain intermediate views between multiple cameras. The depth information, which is required for the virtual view synthesis, was obtained using the segmentation-based stereo matching algorithm. For real-time rendering, our system was optimized using Single Instructions, Multiple Data (SIMD) technology. For efficient streaming, a novel method of combining the video with the depth information is proposed.

Index Terms— free viewpoint video, stereo matching, view synthesis, SIMD, VLC

1. INTRODUCTION

With current technology, a single scene can be recorded with multiple videos and displayed in various fashion. Compared to a single-view video, multi-view videos can provide more flexibility and freedom to users [1]. Promising applications of multi-view videos are free viewpoint video, three dimensional display, panoramic display, etc. However, due to the large bandwidth requirement for storage and transmission, most services offer only fixed views. In response to emerging applications for multi-view video, the MPEG has recently presented a Call for Proposals on multi-view video coding.

In this paper, a simple free viewpoint video system which is able not only to offer free viewpoint video service but also to efficiently stream the necessary video over a network is described. To provide videos from an arbitrary viewpoint, virtual view synthesis technique which was used to generate intermediate views between multiple cameras is developed. While image-based view synthesis rendering has been researched for a long time and achieved high perceptual quality of synthesized results [2], [3], video-based view synthesis rendering in real-time still has problems in terms of both computation time and perceptual quality of synthesized views. This paper describes a virtual view synthesis using projection to overcome aforementioned problems. When synthesizing virtual view using projection, depth information is required, and a disparity map, which is directly related to the depth information, is obtained using a segmentation-based stereo matching algorithm. An efficient way of refining the estimated disparity map is proposed. For encoding, decoding, and streaming of the video, modified VideoLan Client (VLC) [4] program which supports various video codecs and streaming protocols is used. For real-time rendering, the Single Instruction, Multiple Data (SIMD) technology is used in the virtual view synthesis.

The rest of the paper is organized as follows. Section 2 presents the overall system. Section 3 provides the simulation results, and Section 4 concludes the paper.

2. FREE VIEWPOINT VIDEO SYSTEM

Fig. 1 illustrates the block diagram of the overall system. At the encoder side, first, disparity maps of two video clips from left and right cameras are extracted using a stereo matching algorithm. Then, the video clips and their disparity maps are encoded and streamed using the VLC. At the decoder side, video clips and their disparity maps are decoded from the received streaming bits. When a user selects viewpoint, the video clip that corresponds to the selected viewpoint is synthesized using the virtual view synthesis technique and shown to the user using the VLC player.

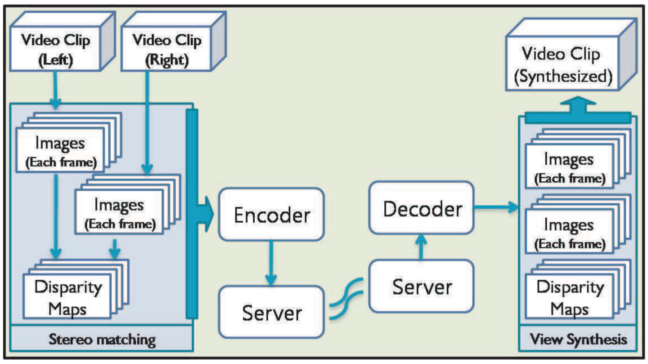

Fig. 1. Block diagram of system.

2.1. Stereo Matching

Stereo matching aims to estimate a disparity map from which we can reconstruct 3D scene structure of a pair of images obtained by multiple cameras [5]. For our application, we used a segmentation-based stereo matching algorithm with belief propagation (BP). When using segmentation-based algorithm like [6] rather than pixel-based one [7], it produces good results not only at depth discontinuities but also in textureless or occlusion regions generally.

2.1.1. MRFs construction

In a conventional method, MRFs are created with the pixels as nodes and edges between 4-connected neighbors [7]. However, in our application, MRFs for each image are constructed as in [6] where each segment is considered as node, and an edge is generated between

This work was supported by grant No. R01-2007-000-20949-0 from the Basic Research Program of the Korea Science and Engineering Foundation.

two segments when their boundaries adjoin each other. Color segmentation is performed to divide given images into regions that are likely to have similar colors which corresponds to similar disparity values [6]. Consequently the number of neighbors for each node will vary unlike the conventional MRF-based approaches.

For each image I_i in the set of all images $\mathcal{I}$, there is a node corresponding to each segment $s_k \in S_i$ and an edge between all neighboring segments $s_l \in N(s_k)$ in each disparity field D_i, where S_i and $N(s_k)$ are the set of all segments in image I_i and the set of neighboring segments to s_k respectively. Then the number of states for each node is equal to the number of possible disparity levels. Our goal is to estimate the most probable D_i, given $\mathcal{I}$, $P(D_i|\mathcal{I})$. This posterior probability can be rearranged by using Bayes' rule as:

$$P(D_i|\mathcal{I}) \propto P(\mathcal{I}|D_i)P(D_i). \tag{1}$$

And this posterior $P(D_i|\mathcal{I})$ can be again factorized as:

$$P(D_i|\mathcal{I}) \propto \prod_{j\in N(i)} \prod_{s_k \in S_i} \psi_{ik}(d_{ik}; I_j) \prod_{s_k \in S_i} \prod_{s_l \in N_i(s_k)} \psi_{ikl}(d_{ik}, d_{il}) \tag{2}$$

where $N(i)$ is the set of neighboring images to I_i, $N_i(s_k)$ is the set of neighbors of segment s_k in image I_i, d_{ik} is disparity value for segment s_k in image I_i, $\psi_{ik}(d_{ik}; I_j)$ and $\psi_{ikl}(d_{ik}, d_{il})$ are the compatibility functions between a latent node and an evident one and between latent nodes of the MRFs. We choose the compatibility functions, $\psi_{ik}(d_{ik}; I_j)$ and $\psi_{ikl}(d_{ik}, d_{il})$ as in [6].

$$\psi_{ik}(d_{ik}; I_j) \propto \left(\frac{h^*_{ik}(d)}{max_{\bar{d}} h^*_{ik}(\bar{d})} \right)^{\nu}, \tag{3}$$

$$\psi_{ikl}(d_{ik}, d_{il}) = \varphi_{kl} N(d_{ik}; d_{il}, \sigma_s^2) + (1 - \varphi_{kl})U. \tag{4}$$

In (3), $\psi_{ik}(d_{ik}; I_j)$ represents the likelihood $P(\mathcal{I}|D_i)$ in (1) and it means match probability that indicates how many pixels in segment s_k of image I_i agree with pixels in image I_j when the disparity value of segment s_k is d_{ik}. To compute $\psi_{ik}(d_{ik}; I_j)$, each pixel in segment s_k of image I_i is projected onto image I_j and find the difference in color at first. Then each difference is added to a histogram with bins of a certain range. If we denote the largest bin in the histogram for a certain disparity value d as $h^*_{ik}(d)$, the value of $h^*_{ik}(d)$ is scaled by the maximum over all possible disparities like (3), and ν is used to get more reliable match probability and can be set to any reasonable value.

$\psi_{ikl}(d_{ik}, d_{il})$ in (4) denotes the prior $P(D_i)$ in (1), and it describes the distribution of the disparities for all segments. To constrain the disparities to vary smoothly between neighboring segments as well as over inside of the segments, $\psi_{ikl}(d_{ik}, d_{il})$ is formulated by using normal distribution with mean d_{il} and variance σ_s^2 and using uniform distribution over a range of possible disparities. These two distributions are controlled by φ_{kl} which is formulated as follows:

$$\varphi_{kl} = \eta exp\{-||m_k - m_l||^2 / 2\sigma_m^2\} + \rho \tag{5}$$

where η has a value less than one not to enforce the smoothness assumption between nodes strictly, and ρ has a value greater than zero to ensure that each state of neighboring nodes has some influence. The parameters m_k and m_l are the mean color of segments s_k and s_l.

Now we can compute the disparities by using belief propagation techniques such as sum-product/max-product method. The overall algorithm, similar to [6], is given as follows:

1. Initialize

   i) Calculate $\psi_{ik}(d_{ik}; I_j)$ for every segment and disparity for each image I_i.

   ii) Assign uniform distribution to all messages between nodes in each MRFs.

2. Calculate the beliefs for each node.
3. Update all messages between nodes by using belief propagation.
4. Update $\psi_{ik}(d_{ik}; I_j)$.
5. Iterate until converging.

2.1.2. *Refinement of disparity maps*

There might be erroneous region in estimated disparity map due to a scene structure. Most of them occurs in occlusion regions. We propose an efficient way of detecting and handling the occlusion, which is appropriate to our application and maybe helpful for another applications. To get a reliable disparity map, left-right consistency check method was used. In this step, if the difference between the disparity value of a pixel in the left image and that of the corresponding pixel in the right image is greater than 2 levels, then the pixel is labelled as an invalid pixel. After the left-right consistency check, each invalid pixel is classified into occluded and mismatched pixel. To decide whether the pixel is occluded or mismatched, at first, each pixel in the left image is projected onto the right image according to their disparity value. So, let's say, if pixel 'p' is projected onto the right image with it's disparity d_p and the corresponding pixel in the right image is 'q' with disparity d_q then the magnitude of d_p and d_q is compared. If d_q has a greater value than d_p, then pixel 'q' is projected onto the left image and find the matching pixel of 'q', $\hat{q}$, in the left image. Finally, if d_q and $d_{\hat{q}}$ has same disparity, then the pixel 'p' is occluded. Otherwise, 'p' is mismatched. If the invalid pixel is occluded, then the lowest disparity value is taken among its 8-neighborhood's disparities since the occluded pixel must have a lower disparity than a occluding pixel. And if the invalid pixel is mismatched, then the median of its 8-neighborhood's disparities is chosen for the disparity of the mismatched pixel as illustrated in Fig. 2. Using this simple and efficient refinement method, we could get a plausible disparity map for our application. In Fig. 3, result of the left-right consistency check is shown for 'Dolls' image. Both the leftmost region in the left image and the rightmost region in the right image are treated as the occlusion.

2.2. View Synthesis

View synthesis technique can improve the performance of both encoding and decoding of multi-view video system. In encoding process, view synthesis can be used to reduce spatial redundancy between views from adjacent cameras. And in decoding process, view synthesis technique is used to show views at an angle between multiple cameras. The following sections present the view synthesis algorithm based on the projection theory.

2.2.1. *Projection Theory*

Projection theory can provide information that which points in the world coordinate are matched to points in the image coordinate. A procedure to find projected points of a virtual view is described as follows. First, we obtain a depth value from the same location in the disparity map. The disparity value is inversely proportional to the

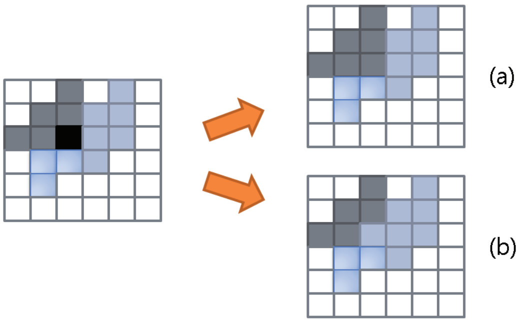

Fig. 2. Illustration of the refinement process. Black-colored pixel could be occluded or mismatched. If (a) the pixel is occluded then the lowest disparity value is taken among its 8-neighborhood's disparities. Otherwise, (b) the pixel is mismatched and the median of its 8-neighborhood's disparities is chosen for the disparity of the pixel.

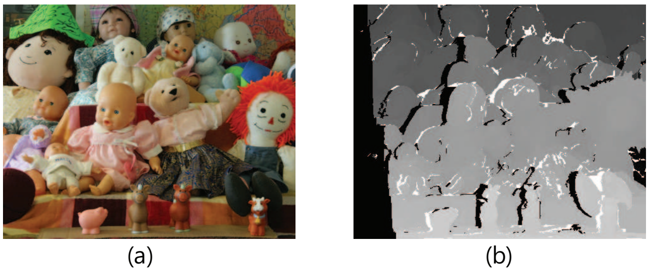

Fig. 3. (a) Left image for 'Dolls' (b) Result of the left-right consistency check. The occluded regions are filled with black and the mismatched regions are filled with white.

depth, thus it should be properly converted. Let $D[c, t, u, v]$ denote the depth at camera c, frame t, and at location $[u, v]$ in the image. Then we can find a world point $[x, y, z]$ related to a point $[u, v]$ on image coordinate using inverse procedure of projection as follows:

$$[x, y, z] = \{R(c)A^{-1}(c)[u, v, 1]\}D[c, t, u, v] + T(c) \qquad (6)$$

where $A(c)$, $R(c)$, and $T(c)$ are an intrinsic matrix, a rotation matrix, and a translation vector of camera c, respectively. Then, the world coordinates are mapped into the target coordinates $[u'', v'', w]$ of the frame in camera c' which we wish to predict from via

$$[u'', v'', w] = A(c') \cdot R^{-1}(c) \cdot \{[x, y, z] - T(c')\}. \qquad (7)$$

Finally, to obtain a pixel location, the target coordinates are converted to homogenous form $[u''/w, v''/w]$ and the intensity for pixel located at $[u, v]$ in the synthesized frame is $I[c', t, u, v] = I[c, t, u''/w, v''/w]$. The $A(c)$, $R(c)$, and $T(c)$ are estimated by using the camera calibration algorithm described in [8]. Fig. 4 (a) and (b) illustrate the projected virtual view images using images from left and right cameras, respectively.

2.2.2. *View synthesis*

Virtual view synthesis based on the projection theory is performed by using the two projected images shown in Fig. 4. For the left image, the black regions reside in the right side of the objects, while it reside in the left side of the objects for the right image as shown in Fig. 4 (a) and (b), respectively. Thus, to synthesize a virtual view using the projection theory, the black regions of the left image is filled with the pixel values of the right image.

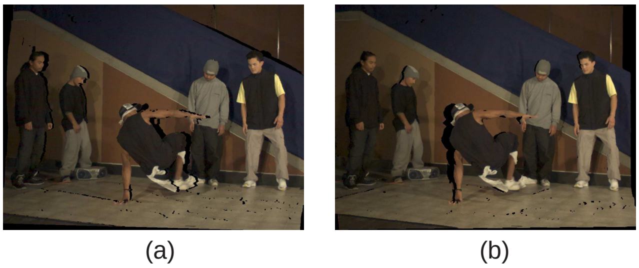

Fig. 4. Projected views (a) projected image using image from left camera (b) projected image using image from right camera.

2.2.3. *Virtual projection matrix for virtual view*

To synthesize a virtual view using projection theory, the intrinsic matrix, the rotation matrix and the translation vector for the virtual view should be prepared. By interpolating the intrinsic matrices of two outer cameras that can be thought of as the left and the right camera of the virtual view, the intrinsic matrix for the virtual view is obtained. In the same way, the translation vector is calculated. The rotation matrix for the virtual view is estimated as follows:

i) The rotation vectors of the two outer cameras are obtained using inverse Rodrigue's formula.

ii) The rotation vector of the virtual view is obtained from interpolating the rotation vectors of the two outer cameras.

iii) The rotation matrix of the virtual view is obtained using Rodrigue's formula.

2.2.4. *Optimization*

The processing speed should be faster than the video frame rate in order to perform view synthesis in real-time. The SIMD technology is beneficial to reducing the computation time of applications where the same operations are performed for many data points, which is appropriate for our situation at hand. We convert 3 functions of projection to SIMD operations using C intrinsic functions. These are the function of calculating $D[c, t, u, v]$, the function of inverse procedure of projection and the function of projection. Consequently, we were able to reduce processing time of view synthesis.

2.3. H.264 codec and streaming

To stream, play, encode and decode the video clip, we used a modified version of VLC [4] which is a highly portable multimedia player supporting many video codecs and various streaming protocols. Among many free codecs that VLC uses, we use H.264/AVC. User datagram protocol(UDP) unicast is used for streaming.

To encode two video clips and disparity maps together, the encoder module of VLC is modified. We split video clips into frames using FFmpeg module in VLC. From each frame of the video, we obtained two frame images from two video clips and two disparity maps from the stereo matching explained above. Then we concatenated these four images into one for every frame to avoid any synchronizing problem.

The decoder module of VLC is modified so that streamed video clip is decoded to images at the client PC for the view synthesis.

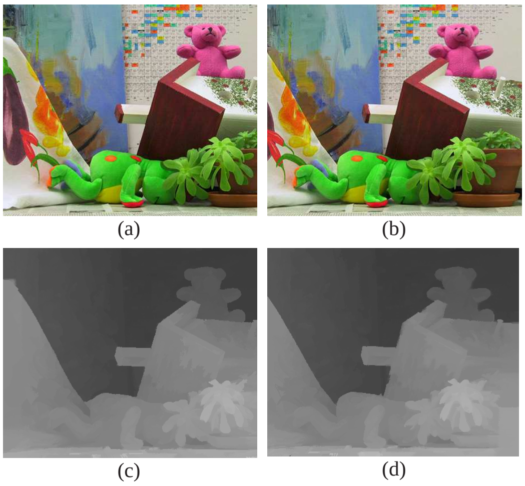

Fig. 5. Stereo matching results (a) left color image (b) right color image (c) left disparity map (d) right disparity map

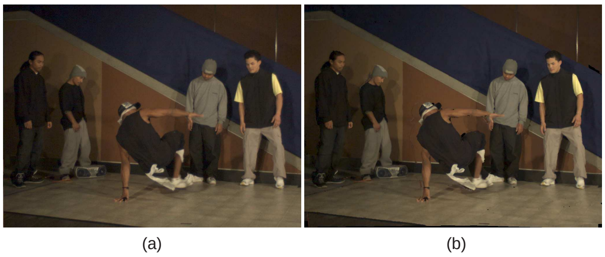

Fig. 6. Comparison synthesis view with original view (a) original cam4 image (b) synthetic cam4 image

And when users select the viewpoint - any position between two cameras - using keyboard, the selected view is synthesized based on the algorithm described in 2.2.

3. SYSTEM EVALUATION

Performance of the stereo matching and the view synthesis were evaluated using Middlebury database [9] and Microsoft (MS) multi-view video database [1], respectively.

For stereo matching algorithm, the parameters of algorithm were set to the following values: $\nu = 4$, $\sigma_s^2 = 2$, $\sigma_m = 18$, $\eta = 0.9$ and $\rho = 0.001$. In Fig. 5, left and right reference images and estimated disparity maps for each image are shown. Comparing estimated disparity map with ground truth disparity map, it is successful to get a disparity map with high quality. In [9], the performance of the stereo matching algorithm can evaluate in terms of percentage of bad matching pixels(PBM). When error threshold is equal to 1 pixel, the PBM of the algorithm described in 2.1 is 7.84 on average. This is not the best in rank of [9], but it is not a problem to synthesize the virtual view using these disparity maps estimated by the algorithm described in 2.1.

For the evaluation of the view synthesis algorithm, cam3 and cam5 images from MS database are resized to 640×480 and used to perform virtual view synthesis. Fig. 6 shows an original cam4 image and a synthetic cam4 image obtained from cam3 and cam5 images using the view synthesis algorithm. As shown in Fig. 6, two images are almost the same except the boundaries of objects. Since the boundaries of depth and color images are different, the boundaries of synthesis image are different from those of original image. Peak signal-to-noise ratio (PSNR) of synthesized image is about 35dB.

We were able to reduce a computational time of virtual view synthesis using SIMD. While one virtual image was synthesized for 0.069 sec before SIMD optimization, it was synthesized for 0.047 sec after SIMD optimization. Thus, the synthesized frame did not drop even at 20fps. We note that the system is implemented on an Intel Core2Quad Q6600 with 2G RAM.

4. CONCLUSION

In this paper, a simple free viewpoint video system which is able not only to display user-specified views at arbitrary angle but also to efficiently stream these signals over a network is described. Virtual view synthesis technique needs a depth information of snapped video clips. Thus, we implemented segmentation-based stereo matching algorithm to estimate a depth information. And an efficient way of refining the estimated disparity map was proposed. The projection theory is used for virtual view synthesis. It is important on the video-based rendering to reduce the processing time. Then, we optimize virtual view synthesis algorithm using SIMD to achieve fast processing speed. And we modify VLC program to encode and decode multi-view video, to offer streaming service, and to play video. This system can be applied to Internet streaming service for movie which is snapped by multiple cameras and rebroadcast of sports games obtained by several cameras.

5. REFERENCES

[1] Matthew Uyttendaele Simon Winder C. Lawrence Zitnick, Sing Bing Kang and Richard Szeliski, “High-quality video view interpolation using a layered representation,” *SIGGRAPH ’04: ACM SIGGRAPH 2004 Papers*, pp. 600–608, August 2004.

[2] Marc Levoy and Pat Hanrahan, “Light field rendering,” *SIGGRAPH ’96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pp. 31–42, August 1996.

[3] Leonard McMillan Steven Gortler Chris Buehler, Michael Bosse and Michael Cohen, “Unstructured lumigraph rendering,” *SIGGRAPH ’01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pp. 425–432, August 2001.

[4] See http://www.videolan.org.

[5] Darius Burschka Myron Z. Brown and Gregory D. Hager, “Advances in computational stereo,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 8, pp. 993–1008, August 2003.

[6] C.L. Zitnick and S.B. Kang, “Stereo for image-based rendering using image over-segmentation,” *International Journal of Computer Vision*, vol. 75, no. 1, pp. 49–65, October 2007.

[7] N.N. Zheng J. Sun and H.Y. Shum, “Stereo matching using belief propagation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 7, pp. 787–800, July 2003.

[8] Zhengyou Zhang, “A flexible new technique for camera calibration,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330–1334, November 2000.

[9] See http://vision.middlebury.edu/stereo/.