Enhancing GraphCut Algorithm Based Depth Map Generation by Using Moving Objects Detection

Jaeho Lee[1,2], Gun Bang[1], Namho Hur[1], Jinwoong Kim[1] and Changick Kim[2]

[1]Electronics and Telecommunications Research Institute,

161 Gajeong-Dong, Yuseong-Gu, Daejeon, Republic of Korea.

[2]Electrical Engineering, Korea Advanced Institute of Science and Technology,

Mungi-Dong, Yuseong-Gu, Daejeon, Republic of Korea

Abstract

In this paper, we present a novel method for enhancing GraphCut algorithm based depth map generation. The usual GraphCut based depth map generation is conducted in the first frame. For the successive frames, the GraphCut algorithm is applied to foreground (i.e., moving objects) only. To this end, each frame needs to be partitioned into foreground and background by using frame difference. For the background regions, the depth values obtained in the previous frame are used. To reduce error, user interaction can be allowed.

1. Introduction

With the development of recent broadcasting systems, three-dimensional television (3DTV) has become more popular for TV viewers. Since 3DTV gives the sense of depth in contrast to the conventional 2DTV, it helps viewers raised realistic viewing experience.

For the realization of the 3DTV, the different image should be projected to the both eyes of viewers. The images can be respectively obtained by using two cameras arranged in the narrow interval. Recently, however, the DIBR (Depth Image Based Rendering)[1] technique which produces the image of arbitrary view from the color image and corresponding depth map is used for the coding efficiency[2]. Moreover, the natural view change within a scene should be possible in order to give viewers immersive feeling as well as the sense of depth.

For this purpose, it is important to generate a dense depth map in any 3D system. Especially, MVC (Multi-view Video Coding) and acquisition of corresponding depth map have been big issues in the MPEG standardization[3]. They use GraphCut[4] algorithm to get the depth map at each camera in multi-view systems. Although this method shows reasonable results[5], they also have some drawbacks. First of all, since this method conducts energy minimization algorithm at every frame, depth values in the background tend to change continuously. In addition, since GraphCut algorithm uses the relationship between current pixel and its neighbor pixels, it may yield estimation errors on foreground objects.

In this paper, we present a novel method for enhancing depth map estimation by using moving objects detection. For better and fast estimation of foreground depth, we apply GraphCut algorithm to foreground regions only after separating foreground from background. For this purpose, we assume that foreground regions belong to moving objects. And we also make use of the depth values of previous frame in the background region for the prevention of flicker noise.

The computational complexity can be reduced by applying the GraphCut only to the foreground region. And also possible errors can be minimized by separate handling of foreground and background.

The outline of this paper is as follows. Section 2 gives the idea of the proposed method and section 3 shows the experimental results. The discussion and conclusion are presented in section 4.

2. Depth Map Generation

The proposed algorithm is illustrated in Fig. 1. The details are explained in the following subsections.

2.1 Foreground detection

As explained previously, we assume that foreground regions belong to moving objects. For fast detection, we use frame difference to detect the moving objects region.

The process for determining the moving objects pixels at the *n*-th frame is as follows:

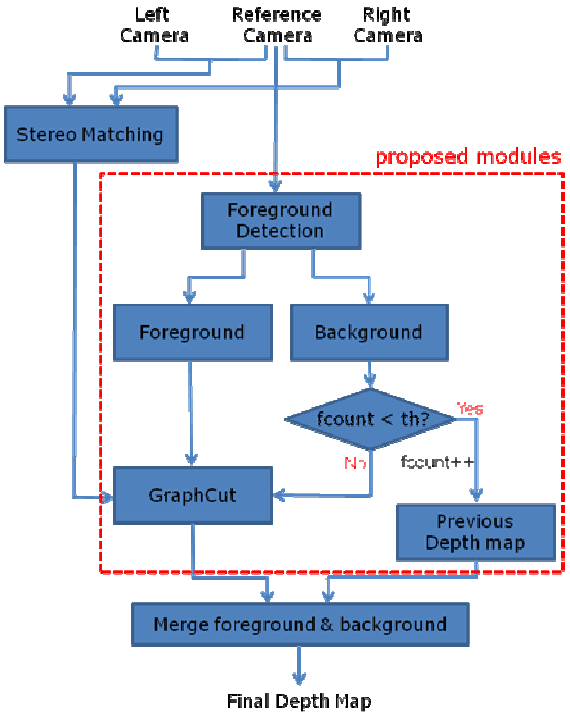


Fig 1. The flow chart of the proposed algorithm

$$F_n(i,j)=\begin{cases}1, & if\ |Y_n(i,j)-Y_{n-1}(i,j)|>th_n\\ 0, & otherwise\end{cases} \quad (1)$$

$Y_n(i, j)$ means the intensity of (i, j) at the frame n. In case of the pixel determined as the background region in the current frame but moving objects region in the previous frame, the depth value should be newly calculated since we cannot use the depth value calculated in the previous frame. Therefore, the expansion version of pixel based moving objects region is needed. We use the block based map in this paper. For the expansion to the block based moving objects region, we use HOS (Higher Order Statistics) to emphasize the moving objects pixels. The second-order moments are calculated for moving pixels in this paper. For instance, the second-order moment at a moving pixel (i, j) is defined as follows:

$$\hat{m}_n^{(2)}(i,j)=\frac{1}{N_\eta}\sum_{(s,t)\in\eta(i,j)}\left(F_n(s,t)-F_n(i,j)\right)^2 \quad (2)$$

$\eta(i,j)$ means the neighbor pixel of (i,j) and N_η means the number of pixels within the ηxη rectangle. In this paper, we define η as 3. In case that the pixels more than 10% within the block are determined as moving objects pixels, this block is defined as a moving block. The

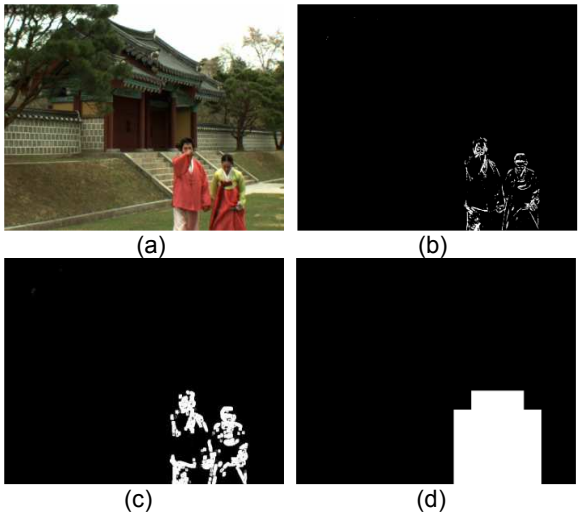


Fig 2. Moving objects extracting procedure
(a) texture map, (b) pixel based moving objects map, (c) emphasized objects map through HOS procedure, (d) converted block based objects map

threshold value used in Eq. (1) is the mean of pixel difference in the whole frame and is defined as follows:

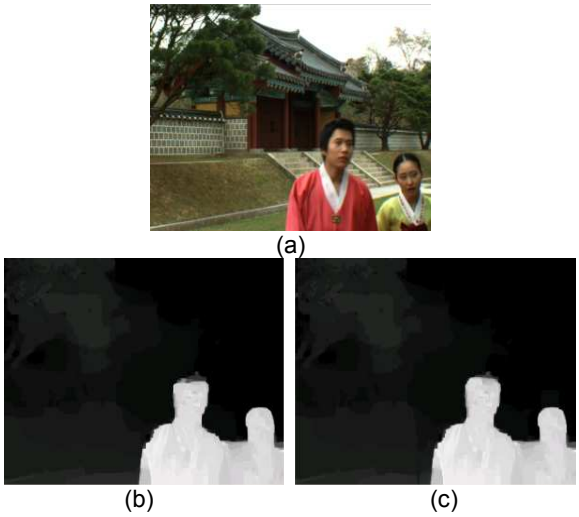
$$th_n=\frac{1}{w\times h}\sum_{i=0}^{w-1}\sum_{j=0}^{h-1}\left(Y_n(i,j)-Y_{n-1}(i,j)\right) \quad (3)$$

w and h mean the width and height of the image. The results at each process of obtaining the block based moving objects region are shown in Fig. 2.

2.2 Foreground depth map generation

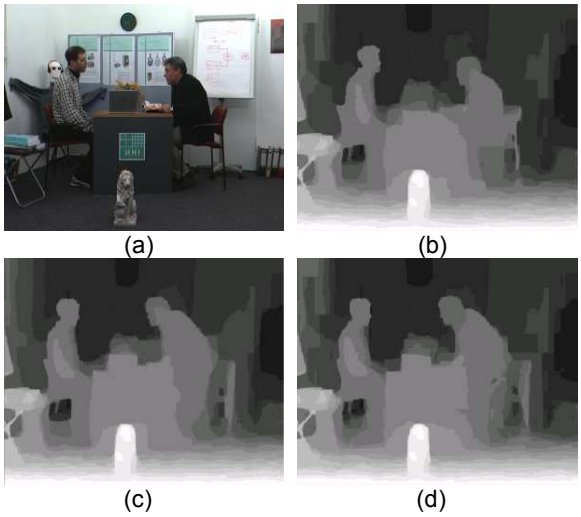
Since the depth value of the moving objects region changes continuously, the depth value should be newly calculated using the GraphCut. Since the GraphCut is only applied to the predicted region that the depth value changes, the computational waste generated by applying the GraphCut to the background region can be reduced. In addition, since the energy minimization process is applied by using a comparison with the adjacent pixels, the depth value of the current pixel is influenced by those pixels.

There are some estimation errors at the head part of walking man due to the initial disparity value of background region (See Fig. 3(a)). But these estimation errors are reduced by removing background part at the GraphCut stage. The depth values of foreground at Fig. 3 are emphasized slightly for better observation.



(a) (b) (c)

Fig 3. The result of foreground region
(a) texture map, (b) the result of using GraphCut at whole image, (c) the result of using GrapgCut at moving objects region only



(a) (b) (c) (d)

Fig 4. The result of background region
(a) texture map (frame #25), (b) depth map (frame #25), (c) depth map of previous method (frame #35) (d) depth map of proposed method (frame #35)

2.3 Background depth map generation

It is noted that there are no depth changes at the background region. But there are some flicker noises at the background region if the GraphCut method is used at every frame. It gives a big fatigue to viewers in watching the 3DTV.

We use temporal information to reduce these flicker noises. In the previous stage, we divide reference frame into foreground and background region. There are no motion changes at the background region. In other words, there are no depth changes at the background region. So it is reasonable that the depth value of the previous frame is used repeatedly at the background region until the end of sequence.

However, since the depth value of a first frame may be mis-estimated and the error is propagated to the next frame, we allow user interaction every predefined period. The variable "fcount" (see Fig. 1) is increased at every frame and return to zero when this value becomes equal to the period. At this moment, the depth value of the background region is re-estimated. User may set up this period in the configuration file.

As shown in Fig. 4(d), there are no depth changes at background region compared with Fig. 4(b), whereas some differences are shown in Fig. 4(c). We can reduce flicker noises at the complex background region such as the chair at the left-bottom in Fig. 4(a).

3. Experimental Results

For testing the performance of the proposed method, we use "Leaving Laptop" and "Lovebird1" sequences provided by MPEG 3DAV group. All sequences have YUV420 format and 1024x768 resolutions. The sequence "Leaving Laptop" is composed of 100 frames at 16 cameras whereas the sequence "Lovebird1" is composed of 300 frames at 12 cameras. Since the depth map corresponding to each image is not separately provided, a performance cannot be evaluated as the generated depth map itself. So we compare the results by generating the intermediate image.

The PSNR results of Y value are shown in Table 1. The results show the PSNR improvements of about 0.4~0.8dB while the complexity of a calculation is reduced. Not only objective results but also subjective results are shown in Fig. 5 and 6. In the result from the previous method as shown in Fig. 5(a), we can see the error of green line within the red circle. This error is removed in Fig 5(c) from the proposed method. Moreover, in the indoor image in which the range of the depth value is

Table 1. PSNR results of proposed method

dB	Leaving Laptop		Lovebird1	
	Cam08	Cam09	Cam06	Cam07
Previous method	37.21	36.20	30.25	30.04
Proposed method	37.63	37.02	30.98	30.47

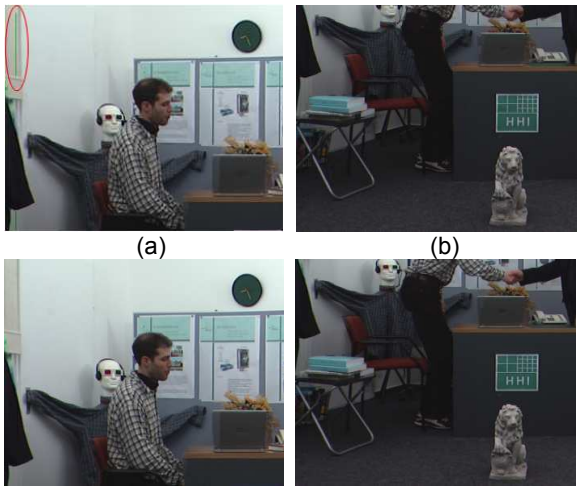


Fig 5. The results of “Leaving Laptop” sequence (a)(b) intermediate view using depth map of the previous method, (c)(d) intermediate view using depth map of the proposed method

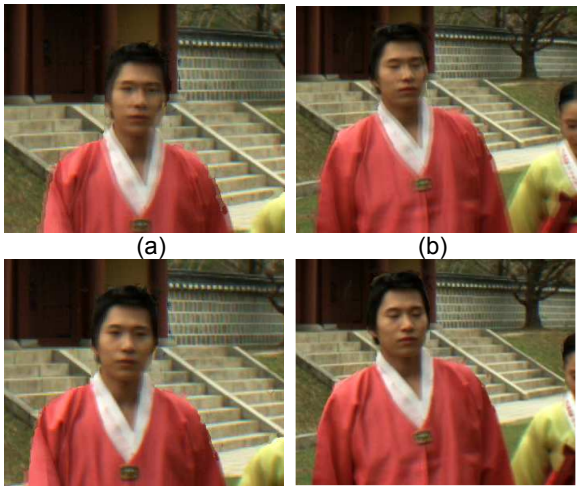


Fig 6. The results of “Lovebird1” sequence (a)(b) intermediate view using depth map of the previous method, (c)(d) intermediate view using depth map of the proposed method

not big like “Leaving Laptop”, the difference of the depth value between the frames is relatively large (see the laptop at Fig. 5(b)). This kind of the difference appears as the noise which seems to swing in the result of generating intermediate view. But we can see the reduction of this noise at the result of the proposed method (see Fig. 5(d)). We also see the improvement in the image quality of an intermediate view of “Lovebird1” sequence. As shown in Fig. 6(a) and 6(b), there are some blurred effects around head and arm region in the previous method. But these noises are reduced by using the proposed method (see Fig. 6(c) and 6(d)).

4. Conclusions

We propose a novel method for enhancing depth map estimation. The proposed method reduces the complexity of calculating the depth value and noisy effects of the image of the intermediate view.

But the block-based approach achieves fast processing time at the expense of possible blocky artifacts in the background. And there is a problem that the initial depth map is required to be dense. The study to resolve these problems is necessary.

Acknowledgement

This paper was supported by the project "Development of Next-Generation DTV Core Technology" through MKE (Ministry of Knowledg e Economy) at ETRI.

References

[1] A. Ignatenko and A. Konushin, “A Framework for Depth Image-Based Modeling and Rendering,” *Proc. Graphicon*, pp. 169-172, 2003.

[2] C. Fehn, “Depth-Image-Based Rendering (DIBR), Compression and Transmission for a New Approach on 3D-TV,” *Proc. SPIE, Stereoscopic Displays and Virtual Reality Systems XI*, vol. 5291, pp. 93-104, 2004.

[3] MPEG 3DAV Group, “Description of Exploration Experiments in 3D Video Coding ,” *ISO/IEC JTC1/SC29/WG11*, Doc. N10173, , Oct. 2008.

[4] Y. Boykov and V. Kolmogorov, “An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision,” *IEEE Trans. on PAMI*, vol. 26, no. 9, pp. 1124-1137, Sept. 2004.

[5] M. Tanimoto, T. Fujii, and K. Suzuki, “Reference Software of Depth Estimation and View Synthesis for FTV/3DV”, *ISO/IEC JTC1/SC29/WG11*, Doc. M15836, Oct. 2008.