Texture-based Depth Frame Interpolation for Precise 2D-to-3D Conversion

Kuan-Ting Lee, En-Shi Shih, and Jar-Ferr Yang

Institute of Computer and Communication Engineering, Department of Electrical Engineering,
National Cheng Kung Univ., Tainan 701, Taiwan

Keywords: 2D-to-3D video, Depth Estimation, Depth Interpolation

ABSTRACT

In general, if the 3D videos are represented by texture and its depth frames, the 3D multiview contents can be effectively produced by depth-image-based rendering (DIBR). In recent years, many researches are proposed to deal with the estimation of the depth map by using stereo images. For many 2D movies, the traditional depth cue methods are limited by specified scenery and achieve poor depth quality. In this paper, we proposed a precise depth map interpolation algorithm to estimate depth maps from two known depth maps as the depth keyframes and the color texture frames. After proper computation of superpixels, the proposed depth frame interpolation system contains texture-based depth estimation, error compensation, noise elimination, and forward/backward depth map merging steps. Simulations show that the proposed system can obtain high quality depth frame interpolation.

1 INTRODUCTION

In recent years, with vigorous development of 3D naked-eyes displays, 3D visualization has been widely used in the fields of entertainment games, film and television, medical applications. If the 3D information is with texture and depth information, the 3D contents can be produced by depth-image-based rendering (DIBR) [1]. The DIBR generates multi-view images for effective 3D visualization. In addition to 2D color texture images, the corresponding depth map which presents the distance information of each pixel is necessary for the DIBR algorithm. The depth map and the texture image are used to interpolate/extrapolate other views. We are going to generate the depth map before performing DIBR for the 2D image frame.

In general, the depth generation methods can be classified into three types: manual, semi-automatic, and automatic. For manual depth generation methods, a large amount of human resources is required to synthesize the depth map through segmentation of objects from color images. Such high labor cost work not only consumes time but also cannot meet the large demand of the 3D content industry. For automatic methods, it is divided into two parts. One is simple and intuitive, but can only handle a specific condition or scene. The other needs more complex calculations, but the results may not be satisfactory. Up to now, many methods, such as image classification [2], relative height cue [3], vanishing lines and vanishing point [4] have been proposed to generate the depth map required for 2D-3D conversion. These methods are intuitive and low computation. Nevertheless, each method has its own limitation to suit a certain situation.

For estimating the depth of frames, it is important to precisely segment the moving objects. In order to meet this requirement, in this paper, we adopt the superpixel algorithm, called simple linear iterative clustering (SLIC) [5]. The SLIC adapts the k-means clustering method to efficiently generate superpixels. Superpixel makes up with adjacent and homogeneous region of the image. Segment the image makes the pixels meaningful and each superpixel consists of many pixels with similar colors and textures. It can reduce megapixels to hundreds of superpixels. Moreover, each superpixel in similar the color or texture is more meaningful than a single pixel. Since the depth map is the most important thing for 3D techniques, this paper proposed a semi-automatic depth generation method. Combining the above advantages, this method is not only intuitive but also high quality. The rest of this paper is organized as follows. The proposed system based on color texture information will be introduced in Section Two. Experimental results for video sequences will be demonstrated in Section Three. Finally, conclusions are drawn in Section Four.

2 THE PROPOSED SYSTEM

In the proposed system, the generation of the depth maps of a video sequence needs all color images frames and a few known depth maps of the keyframes. The video sequence is divided into different scene cuts manually where the frames in the same cut are highly correlated. For about every 5–20 frames, a color texture frame is selected as the keyframe. Then, the depth keyframe is generated by an existed depth manually generation software.

The proposed texture-based depth interpolation system is shown in Fig. 1. Assume that we have n adjacent color texture frames (F_1, F_2, …, F_n) and two depth keyframes (D_1, D_n) as

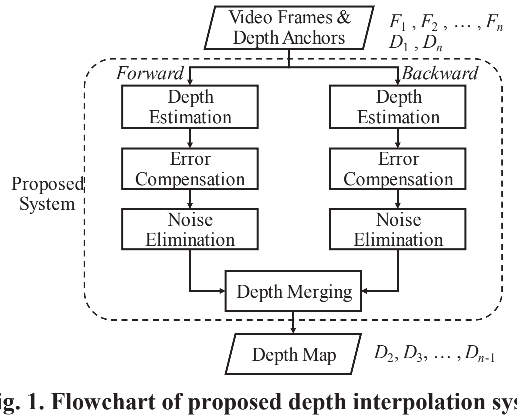

Fig. 1. Flowchart of proposed depth interpolation system

the inputs. The proposed system is designed to interpolate the precise depth maps between two depth keyframes automatically. From the first and last depth keyframes, we could be obtained two estimated depth frames, which are generated from forward and backward propagation flows individually. Finally, we use a depth merge unit to get the final depth maps. In the following three subsections, we will describe them in details.

2.1 Depth Estimation Unit

The reference image $F_{t'}$ with its corresponding reference depth frame $D_{t'}$ are known for depth estimation, where t'=1 or n. For the frame F_t, we need to estimate the corresponding depth frame D_t from $F_{t'}$ and $D_{t'}$, where $1 < t < n$. Fig. 2 shows the flowchart of the depth estimation unit, which is further composed of *similar pixel search* and *candidate pixel chosen* kernels.

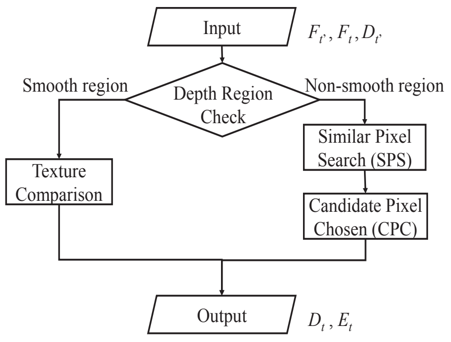

Fig. 2. The flowchart of depth estimation unit

The target pixel p is at the position (x, y) on F_t. A window that should cover the displacement of p is defined centered at the position (x, y) on $F_{t'}$. By limiting the depth and texture values, we first check whether the window is a smooth depth region and the similar pixel could be found in the window, as

$$S(p) = \begin{cases} 1, & \text{if } d_{\max}^{\Omega} - d_{\min}^{\Omega} < 5 \\ 0, & \text{else} \end{cases} \tag{1}$$

and

$$T(p) = \begin{cases} 1, & \text{if } S(p) = 1 \text{ and } \min \left\| I_p - I_q \right\| < \tau,\ q \in \Omega \\ 0, & \text{else} \end{cases}, \tag{2}$$

where $d_{\max}^{\Omega}$ and $d_{\min}^{\Omega}$ denote the maximum and minimum depth values in the window respectively, I_p is the intensity of p, q denotes the pixel in the window, I_q is the intensity of each pixel q, and τ is a pre-defined threshold. If $S(p)$ equals to 0, the window denotes a non-smooth depth region, and “similar pixel search” function will be executed to find a similar pixel. If $T(p)$ equals to 1, the window is a smooth depth region and a similar pixel could be found in the window. The depth value of p would be assigned as the average depth of the window. the equation can be represented as

$$d_p^t = \frac{1}{n} \sum_{1}^{n} d_q^{t'}, \text{ if } T(p) = 1, \tag{3}$$

where n is the pixel number in the window. Another situation is that the window is the smooth region but we cannot find any pixel similar to p. It means that p may move too fast to stay in the window. The missing pixel p is classified as an error denoted as

$$E_t(p) = \begin{cases} 1, & \text{if } S(p) = 1 \text{ and } \min \left\| I_p - I_q \right\| \geq \tau,\ q \in \Omega \\ 0, & \text{else} \end{cases}. \tag{4}$$

2.1.1 Similar Pixel Searching

To find the most similar pixel corresponding for the pixel p on the target frame, we first calculate the color difference between p and each q in the window on the Ft'. The color absolute difference can be expressed by

$$D_q^i = \left| I_p^i - I_q^i \right|,\ i \in \{R, G, B, a, b\}, \tag{5}$$

where i is the index of the color component, which could be red, green or blue from RGB space and a or b from CIELab space. I_p^i is the intensity value of the ith color component of p. I_q^i is the intensity value of the ith color component of the compared pixels q on the reference frame.

Through the depth information of $D_{t'}$ and hue of $F_{t'}$, the window region could be separated into several color clusters. Since using different color space to compute cost, the weights are also classified as five types. Furthermore, the weight is assigned from different regions which are clustered. In other words, each region has its unique weight value. According to different color features of each region, the weight can be set as the inverse of the standard deviation defined as

$$w_q^i = \exp(-\sigma_q^i),\ i \in \{R, G, B, a, b\}, \tag{6}$$

$$\sigma_q^i = \sqrt{\frac{1}{n-1} \sum_{q=1}^{n} \left\| I_q - \bar{I} \right\|^2},\ q \in Z, \tag{7}$$

where σ_q^i denotes the standard deviation of each q, and Z represents the different clusters. When the standard deviations of color components are smaller, the weight becomes bigger. It means that the color of this region with a small standard deviation is more similar.

To search the similar pixel, it compares target pixel p with every q in the window. The cost equation can be represented by

$$C_q^{RGB} = \frac{1}{w_q^R + w_q^G + w_q^B} \cdot \left(w_q^R \cdot D_q^R + w_q^G \cdot D_q^G + w_q^B \cdot D_q^B \right), \tag{8}$$

$$C_q^{ab} = \frac{1}{w_q^a + w_q^b} \cdot \left(w_q^a \cdot D_q^a + w_q^b \cdot D_q^b \right), \tag{9}$$

$$C_q^{Total} = \alpha \cdot C_q^{RGB} + (1 - \alpha) \cdot C_q^{ab}, \tag{10}$$

where C_q^{RGB} and C_q^{ab} denote as the costs of q in RGB and ab color domain, respectively, and α is a weight to adjust the ratio of two color spaces. According to the experiment, the weight is set as 0.2.

The pixel with the minimum cost will be the most similar pixel to the target pixel p. For the microstructure, the pixel-wise searching method is not reliable enough. Moreover, the minimum cost pixel may not be single. We assign these pixels as the candidate pixels. In order to increase the credibility of the result, we further compare the neighbor pixels of the candidate pixels and target pixel p. It can help to confirm the result is the most similar pixel, not just a noise. The prerequisite to becoming the candidate pixel is that the cost is smaller than a threshold. If the minimum cost of the whole window is greater than the threshold, *p* is considered as a missing pixel and the depth of *p* is set as an error. The set of the candidate pixels is expressed as

$$\mathrm{X} = \left\{ q \;\middle|\; \arg\min C_q^{Total} \wedge C_q^{Total} < \tau \right\}, \tag{11}$$

where τ is the threshold depicted in (4). The error map of *p* is updated as

$$E_t(p) = 1, \text{ if } \min C_q^{Total} \geq \tau. \tag{12}$$

2.1.2 Candidate Pixel Chosen

The block cost is calculated by referring to the neighbor eight pixels of the target pixel *p* and the candidate pixels for improving accuracy. It is processed by using *k*-by-*k* block for computation. The block size *k* is an adaptive value referring to the clusters of the window. The value *k* is set to 3 when the clusters in the window are less than 3, otherwise, *k* is set to 5. The block cost of candidate pixels can be expressed as

$$C_{cdd} = \sum \left| \mathbf{b}_p^i - \mathbf{b}_{cdd}^i \right|, \; i = R,\ G,\ B,\ cdd \in \mathrm{X}, \tag{13}$$

where *cdd* denotes the position of the candidate pixel, and **b** denotes the *k*-by-*k* block centered at *p* or *cdd*.

The initial depth assignment adopts the winner takes all method. The block with minimum cost would be picked, and average depth would assign to the target pixel *p*. If there is more than one pixel being the minimum cost, the averaged depth of them are assigned to the target pixel *p*. The depth value of *p* is calculated as

$$d_p^t = \frac{1}{m} \sum \arg\min_d C_{cdd}, \text{ if } S(p) = 0, \tag{14}$$

where *m* is the number of candidate pixels whose block cost is the minimum color difference of the candidate blocks.

2.2 Error Compensation Unit

According to the error map, an error is filled by referring to its adjacent eight pixels because the color intensities between the errors and the neighbor pixels are similar. When the error pixels are on the texture edge, they may cause the inconsistent color with the neighbor pixels. The following two cases are proposed: If the error pixel is on the edge, the most similar pixel of eight neighbors is selected to fill. The other is that we could fill the error pixel by using the average of the neighbors which are not on the edge. Of course, it also needs to exclude the neighbors which are label as errors.

2.3 Noise Elimination Unit

After error compensation, we can get an initial depth map. However, the results may still exist some noises which look redundant and not clean due to computational mistakes. In order to remove them, we further proposed a method to make the depth smoother.

It is noted that we have executed SLIC to segment similar depth region in computation of the initial depth map. The SLIC can group similar pixels together based on color similarity and spatial similarity. The initial depth map is divided into several groups, as called superpixels. Each group tightly connects pixel by pixel, but some noises are classified in a smooth superpixel because of spatiality. The noises are detected by referring to the normal distribution and standard deviation. If the value falling out region of the mean with ±2 standard deviations, it would be treated as noise. The new depth value would replace by calculating from the reliable pixels in four directions, as

$$d_p^t = \begin{cases} \dfrac{\sum \gamma_j d_j}{\sum \gamma_j}, & \text{if } \left| d_p^t - d_{mean}^t \right| > 2\rho,\ d_j \in \Psi \\ d_p^t, & \text{else} \end{cases}, \tag{15}$$

where *j* denotes the reliable pixels, γ_j is the weight differing from the inverse of the distance to the target pixel, Ψ represents the current superpixel, d_{mean}^t is the average depth value of the current superpixel, and ρ denotes the standard deviation of the current superpixel.

2.4 Depth Merging Unit

With the above depth estimation processes, we will obtain two depth maps for each non-keyframe by interpolating from forward and backward propagations. The depth maps generated from different propagated directions must have different results. For the same propagated direction, the best results must be the closest frame from the keyframe. However, one of the two depth maps may have some information that the other losses. Instead of selecting the better one from the two depth maps, we can refer to both of them. To achieve better performance, we would like to merge the two depth maps generated by different directions. This paper adopts a simple merged method mentioned as

$$D_t = \beta \cdot D_t^{fw} + (1 - \beta) \cdot D_t^{bw}, \tag{16}$$

where β is inversely related to the distance from the front keyframe to the target frame.

3 EXPERIMENTAL RESULTS

The sequences of S02 Poznan Street, S03 Undo Dancer, S04 GT Fly, and S10 Shark are utilized in the experiments to demonstrate the performance of the system. The frames of the sequences are randomly selected during testing. Due to multiple steps of this system, we will show the estimated depth map of each step individually. Firstly, the improvements of errors compensation and noise elimination are shown. Secondly, the overall results achieved by the proposed system are presented with the ground truth of the keyframes. The objective quality of the results is measured in terms of PSNR and SSIM performances.

For depth estimation of the video sequence, the accuracy of color matching is very important. Besides, the edge alignment is essential to improve the quality. There are aligned edges accurately estimated by out proposed system as shown in Fig. 3. The depth maps obtained from forward and backward propagation are fused into the finally depth map. For Fig.4, (a) and (b) show the depth maps of frame 11 of estimated forward and backward propagation result. We can see that the merged depth map in Fig. 4(c) is tiny difference between forward and backward depth maps.

The depth maps are generated by interpolation from forward and backward propagations. For forward propagation, the quality of the first frame is better than that of the last frame because of error extension. Since there are two depth maps obtained in two directional propagations for each frame, we make use of their advantages to merge them together. We choose a set from each video sequence. The peak signal-to-noise ratio (PSNR) and structural similarity (SSIM) are utilized as the criterion measurements to analyze the difference among the results, which obtained by forward, backward and merging estimation, as shown in Table 1.

4 CONCLUSIONS

This paper proposed a texture-based depth interpolation system, which can interpolate all unknown depth frames between two depth keyframes by checking texture similarity of all color texture frames through depth estimation, error compensation, noise elimination, and depth map merging steps. The depth region check for smooth regions is purposed to reduce the computational complexity. As for non-smooth regions, it takes the multi-level consideration to handle complex situations by synthesizing different color spaces information in similarity pixel search (SPS) step and it is rigorous to find out the similar pixels in the candidate pixel chosen (CPC) step. The experimental results confirmed that the errors in the estimated depth maps are fewer such that we can simply use the proposed error compensation method to achieve good results. To make the depth smoother, the orphaned pixels are eliminated based on the concept of superpixels. The bi-directional propagation can not only overcome the occlusion of the object but also handle the zoom in/out circumstance. From experimental results, it can be apparently observed that the depth maps are generated successfully. The processed results from depth estimation, error compensation to noise elimination show their own effectiveness and better in step-by-step. Final results show that the edges of the objects between the texture and the depth are well aligned. Generally, the proposed 3D depth interpolation system can produce high-quality 3D videos from 2D videos and depth keyframes, which can be usually obtained from software package.

REFERENCES

[1] Fehn, Christoph. "Depth-image-based rendering (DIBR), compression, and transmission for a new approach on 3D-TV." Stereoscopic Displays and Virtual Reality Systems XI. Vol. 5291. International Society for Optics and Photonics, 2004.

[2] Dong, Hao, et al. "An automatic depth map generation method by image classification." Consumer Electronics (ICCE), 2015 IEEE International Conference on. IEEE, 2015.

[3] Jung, Yong Ju, et al. "A novel 2D-to-3D conversion technique based on relative height-depth cue." Stereoscopic Displays and Applications XX. Vol. 7237. International Society for Optics and Photonics, 2009.

[4] Chou, Chien-Hsing, Yu-Xiang Zhao, and Hsien-Pang Tai. "Vanishing-Point Detection Based on a Fuzzy Clustering Algorithm and New Clustering Validity Measure." 淡江理工學刊18.2 (2015): 105-116.

[5] Achanta, Radhakrishna, et al. "SLIC superpixels compared to state-of-the-art superpixel methods." IEEE transactions on pattern analysis and machine intelligence 34.11 (2012): 2274-2282.

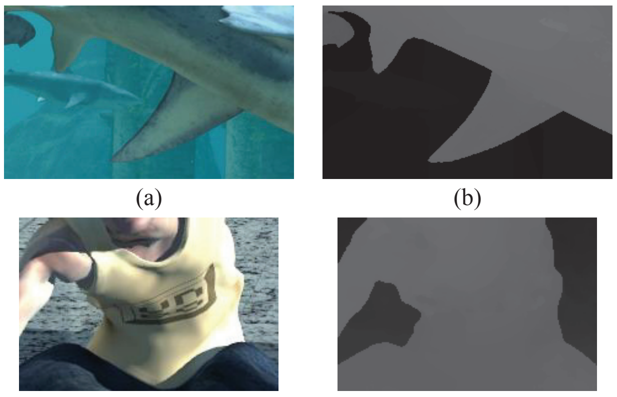
(a) (b) (c) (d)

Fig. 3. Frame #221 of S10 "Shark": (a) texture image and (b) estimated depth map; Frame #111 of S03 "Undo Dancer": (c) texture image and (d) estimated depth map

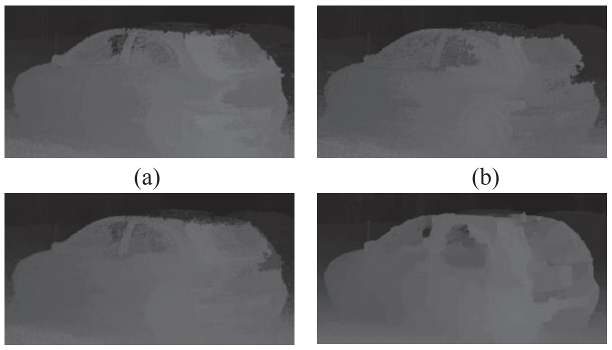
(a) (b) (c) (d)

Fig. 4. The depth maps of "Poznan Street" at frame #11: (a) forward propagation result; (b) backward propagation result; (c) merging result and (d) ground truth

Table 1. PSNR and SSIM results of "Poznan Street"

	PSNR (dB)		
Frame	FW	BW	MG
103	35.9526	33.0686	36.3431
104	34.6240	33.4985	35.4313
105	33.8662	33.8996	35.1049
106	33.7258	34.9447	35.7723
107	32.9865	35.6910	36.0038
	SSIM		
103	0.9118	0.8763	0.9229
104	0.8929	0.8803	0.9164
105	0.8839	0.8852	0.9161
106	0.8789	0.8940	0.9171
107	0.8746	0.9107	0.9219