Effective View-dependent Weighting of Multiple Videos for Smooth Virtual View Transition

Ryota Kojima Tadahiro Fujimoto
Graduate School of Engineering
Iwate University
4-3-5, Ueda, Morioka, Iwate, 020-8551, JAPAN
{h22j039, fujimoto} (a) cis.iwate-u.ac.jp

***Abstract*—Plane sweep is a well-known method to create a virtual view and a depth map of an object from multiple videos. In this paper, we propose an effective view-dependent weighting method for camera pixel colors to compute a plausible virtual view pixel color in plane sweep. Our method achieves the smooth transition of a virtual view, in which a sudden change and a blurring effect of colors are avoided, when a virtual viewpoint moves. For each virtual view pixel, two cameras are selected among all cameras according to the angles between the viewing rays of the cameras and the virtual view. The camera which has the smallest angle is used as a primary camera whose pixel color is surely used. The camera which has the second smallest angle is used as a secondary camera whose pixel color is used if its angle is near the angle of the primary camera; in this case, the pixels of the two cameras are given the weights determined by the difference between their angles. This approach results in segmenting the virtual view into two kinds of regions, that is, one-camera regions and two-camera regions. Each pixel in a one-camera region is given a pixel color of a single camera while each pixel in a two-camera region is given a weighted average of pixel colors of two cameras. The boundary of two one-camera regions becomes a two-camera region which achieves the smooth blending of the two regions. Some experiments show an effective performance of our method.**

Keywords—plane sweep; virtual view; depth map; view-dependent weighting; GPU

I. INTRODUCTION

Plane sweep is a well-known method to create a *virtual view* and a *depth map* of an object from multiple videos which capture the object from different camera positions. This method sweeps a plane parallel to the screen of a virtual view at regular depth intervals in its viewing direction. Each depth is evaluated to judge whether the surface of an object is on the plane or not. For the 3D point on the plane hit by the viewing ray of a virtual view pixel, if the colors of the camera pixels projected to the 3D point are similar to each other, which is called *"photo-consistent"*, the 3D point is judged to be an existing point on the surface. Then, the color of the virtual view pixel is computed from the photo-consistent colors of the camera pixels.

In order to compute a plausible color for a virtual view pixel, a well-known way uses only the color of the camera pixel whose ray is the closest to the ray of the virtual view pixel. However, this causes a sudden color change of the virtual view when the closest camera changes to another one by the movement of the virtual viewpoint. In another way, the colors of the camera pixels are averaged with weights determined in an inversely proportional manner to the angles between the rays of the camera pixels and the virtual view pixel. However, such a simple view-dependent weighting tends to blur the virtual view. In this paper, we propose an effective view-dependent weighting method to avoid the above problems. Our method is considered to be a combination of the above two ways by selecting *primary* and *secondary cameras* for each virtual view pixel and blending their pixel colors with view-dependent weights determined by the difference between the angles of their rays for the virtual view ray. This results in segmenting the virtual view into *one-camera regions* with *two-camera boundary regions* to achieve the smooth transition of the virtual view without a sudden change and a blurring effect of colors.

II. RELATED WORK

Many plane sweep methods have been proposed so far after Collins proposed the first one [1]. Yang et al. proposed a real-time method to obtain a virtual view and a depth map by a hardware acceleration scheme using programmable pixel shader on GPU [2]. Their method calculates the RGB mean and the luminance SSD for the photo-consistency test of each virtual view pixel and each depth using a representative reference camera and selects the mean with the smallest SSD. Woetzel et al. also proposed a hardware-accelerated method [3]. Their method uses truncated SSD scores to limit the influence of outliers due to image noise and non-diffuse surfaces and shiftable windows to handle discontinuities. Li et al. proposed a hardware-accelerated method [4] to efficiently render the *photo hull* of an object using its *visual hull* reconstructed by their other method [5]. Geys et al. proposed a two-step method [6] [7]. In their method, after an efficient plane sweep algorithm generates a crude depth map on GPU, the depth map is refined by a graph cut optimization algorithm on CPU. Nozick et al. proposed a method to treat occlusions by effective score computation to remove outliers [8]. Liu et al. proposed an undersampled light field rendering method using a plane sweep approach [9]. Zabulis et al. proposed a sphere sweeping method [10]. McKenney et al. proposed a parallel plane sweep algorithm for multi-core systems [11]. Ja-

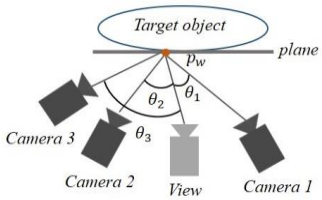


Fig. 1. Angles between camera rays and a virtual view ray. Camera 1 with the smallest angle θ_1 is selected as a primary camera, and camera 2 with the second smallest angle θ_2 is selected as a secondary camera.

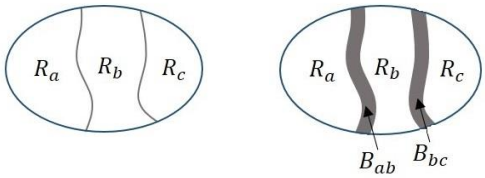


Fig. 2. Segmentation of a virtual view. (a) Segmentation into one-camera regions. (b) Segmentation into one-camera and two-camera regions.

rusirisawad et al. proposed a real-time plane sweep method for uncalibrated cameras by using a projective grid space [12]. Irschara et al. proposed a 3D reconstruction method for aerial images by plane sweep [13]. Hane et al. proposed a plane sweeping stereo method for fisheye images [14].

Matusik et al. proposed a real-time rendering method for a polyhedral visual hull of an object using multiple videos [15]. They used the k-nearest neighbor weighting for camera pixel colors at each vertex of the polyhedral model based on the unstructured lumigraph rendering [16]. This weighting selects the k cameras whose rays to the vertex are closest in angle to the virtual view ray. By using the k-th ray with the largest angle θ_k, the weight W_i for the i-th ray with angle θ_i is defined as follows: $W_i = w_i / \sum_{j=1}^{k} w_j$, $w_i = 1 - \theta_i/\theta_k$. The weight W_k falls to zero while the remaining $k-1$ weights are non-zero. This simple weighting realizes the smooth transition of a virtual view when a virtual viewpoint moves. However, such an weighting approach often blurs the virtual view. We propose a new effective view-dependent weighting method to avoid such a blurring effect as well as a sudden color change caused by using only a single camera.

III. Proposed Method

A. Depth Evaluation by Photo-consistency in Plane Sweep

In plane sweep, the plane at each depth is evaluated to judge whether the surface of an object is on the plane or not. For the 3D point on the plane hit by the ray of each virtual view pixel, if the colors of the camera pixels projected to the 3D point are similar to each other, that is, photo-consistent, the 3D point is judged to be an existing point on the surface. The test of photo-consistency is often done by using the variance of the camera pixel colors; if the variance at the depth is the smallest among the variances at all depths or smaller than a predefined threshold, the camera pixel colors are judged to be photo-consistent. Then, the color of the virtual view pixel is computed from the photo-consistent colors.

B. Selection of Cameras

After obtaining the camera pixels with the photo-consistent colors, we obtain the angle between the ray of each camera pixel that hits the 3D point on the depth plane and the ray of the virtual view pixel, as shown in Fig. 1. Among all cameras, our method selects the two cameras with the smallest angle θ_1 and the second smallest angle θ_2. The camera with θ_1 is used as a primary camera while the camera with θ_2 is used as a secondary camera. Actually, our method compares the angles by their cosines, $\cos\theta_i$, $i = 1, 2, \cdots$, for efficiency by computing each cosine using the inner product of the 3D vectors of two rays; The cameras with the largest and the second largest cosines are selected.

C. Effective View-dependent Weighting

If each virtual view pixel is given only a pixel color of its primary camera, a virtual view is segmented into regions, each of which has pixel colors given by a single camera with clear boundaries, as shown in Fig. 2 (a). Each region, which is called one-camera region, has a high quality with continuous colors that come from a single video while the boundaries with color discontinuity degrade the quality of the whole virtual view. In addition, a sudden color change happens when the virtual viewpoint moves. In order to solve this problem, our method enlarges each boundary, as shown in Fig. 2 (b), and smoothly blends the two one-camera regions adjacent to each other in the enlarged boundary. The enlarged boundary is called two-camera region. The primary camera of a one-camera region R_a becomes the secondary camera of a one-camera region R_b adjacent to R_a. In the two-camera region B_{ab} between R_a and R_b, the angles of the rays of the primary and secondary cameras are close to each other. Thus, our method obtains the primary and secondary cameras and their angles for each virtual view pixel individually and uses them according to the difference between the angles; This process for individual virtual view pixels results in segmenting the whole virtual view into one-camera and two-camera regions appropriately. The angle difference of the primary and secondary cameras is used to determine the weights for the pixel colors of the two cameras. Actually, we use the difference between the cosines of the angles, instead of the difference between the angles, for efficient computation.

For each virtual view pixel p_v, the difference between the cosines of the angles θ_1 and θ_2 of the primary and secondary cameras is represented as follows.

$$D_{12} = \cos\theta_1 - \cos\theta_2 \tag{1}$$

In our method, if the condition of (2) is satisfied, the pixel p_v is treated as a pixel in a one-camera region.

$$D_{12} \geq D_{bnd} \tag{2}$$

The threshold D_{bnd} is defined in advance to control the width of a two-camera region. In this case, the color C_v of the pixel p_v is given the pixel color C_1 of the primary camera.

$$C_v = C_1 \tag{3}$$

Fig. 3. Cameras and a target object.

Inversely, if the condition of (4) is satisfied, the pixel p_v is treated as a pixel in a two-camera region.

$$D_{12} < D_{bnd} \tag{4}$$

In this case, the color C_v of the pixel p_v is given a weighted average of the pixel colors C_1 and C_2 of the primary and secondary cameras by using the difference D_{12} as follows.

$$C_v = \frac{1}{2}\left(1 + \frac{D_{12}}{D_{bnd}}\right) C_1 + \frac{1}{2}\left(1 - \frac{D_{12}}{D_{bnd}}\right) C_2 \tag{5}$$

The above is unified as our view-dependent weighting as follows.

$$C_v = w_1 C_1 + w_2 C_2$$
$$\begin{cases} w_1 = 1 \\ w_2 = 0 \end{cases} \quad (D_{12} \geq D_{bnd}) \tag{6}$$
$$\begin{cases} w_1 = (1/2)(1 + D_{12}/D_{bnd}) \\ w_2 = (1/2)(1 - D_{12}/D_{bnd}) \end{cases} \quad (D_{12} < D_{bnd})$$

This weighting smoothly blends adjacent two one-camera regions in their two-camera region to achieve the smooth transition of a non-blurred virtual view for a moving virtual viewpoint.

D. Implementation for Efficiency

Our software is implemented by using graphic libraries OpenGL and GLSL. In particular, a color of each virtual view pixel is efficiently computed in parallel on a GPU by GLSL fragment shader programming. In each frame time, first, the silhouette of an object is extracted from each video by background subtraction. Then, at each depth in plane sweep, the silhouettes of all the videos are projected on the depth plane to obtain their intersection, which is the interior of a visual hull of the object intersecting with the plane. For efficiency, the photo-consistency tests are done only for virtual view pixels projected within the interior. The depth plane is implemented by using an OpenGL polygon. The 3D point on the plane hit by the ray of each virtual view pixel is obtained efficiently by GLSL rasterizer. Thus, the 3D vectors of the rays of the virtual view pixel and camera pixels to compute their angles are easily obtained.

IV. Experimental Results

We made three experiments to verify the performance of our method. We used ten cameras to capture a target object as shown in Fig. 3. All the cameras were connected to one PC by IEEE1394. The experimental environment is shown in Table 1.

The results of the first experiment are shown in Fig. 4. The images in Fig. 4 are frame images extracted from a resulting

Table 1. Experimental environment.

OS	Windows 8.1 64bit
IDE	Visual Studio 2015
Language	C/C++
Library	OpenGL, GLSL, OpenCV
CPU	Intel(R)Core(TM)i7 2.93GHz
RAM	8GB
GPU	NVIDIA GeForce GTX680
Camera device	PointGreyResearch Firefly MV FFMV-03M2M-CS
Image resolution	640×480

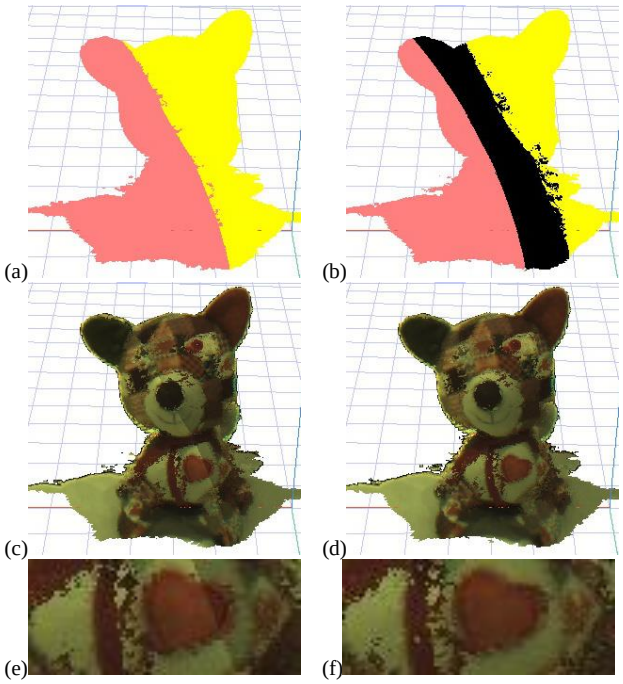


Fig. 4. Removal of color discontinuity caused by boundaries.

virtual view video. The images (a), (c), and (e) show the result by giving each virtual view pixel only a pixel color of its primary camera. The two colors, yellow and pink, in (a) indicate two one-camera regions, each of which is given pixel colors by a single camera. The resulting virtual view image (c) and its magnified image (e) have an artifact of color discontinuity caused by a clear boundary between the two regions. This artifact is noticeable particularly in the resulting video. The images (b), (d), and (f) show the result by our method. The black color in (b) indicate a two-camera region in which the pixel colors of the two one-camera regions are blended by our method. In the resulting virtual view image (d) and its magnified image (f), the artifact in (c) and (e) disappears. The resulting video shows that our smooth blending method works to avoid a sudden color change for a moving virtual viewpoint.

The results of the second experiment are shown in Fig. 5. The images (a), (b), and (c) were made by using a well-known view-dependent weighting approach [15,16]. Every virtual view pixel is given a weighted average of the pixel colors of N cameras, $N = 2$ in (a), $N = 3$ in (b), and $N = 4$ in (c), by us-

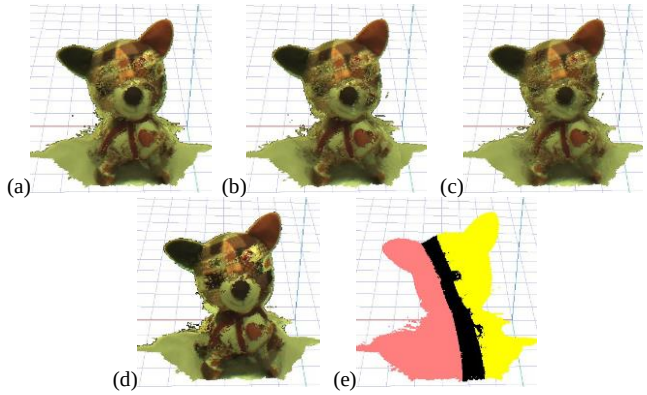


Fig. 5. Comparison of different view-dependent weighting approaches.

ing the angles of their rays to select the cameras and determine the weights. Each of these images has a blur in the whole image although it does not have color discontinuity. The image (d) made by our method does not have a severe blur because it consists of dominant non-blurred one-camera regions and slightly blurred two-camera regions. The image (e) shows one-camera and two-camera regions of (d).

The results of the third experiment are shown in Fig. 6. The images in each of (a) and (b) are a set of successive frame images extracted from a resulting virtual view video. The video of (a) was made by the simplest way; It was made by selecting a camera whose optical axis direction was the closest to that of a virtual view and giving the pixel colors of the selected camera to all virtual view pixels. That is, the virtual view depends on a single camera. In (a), the colors of the whole virtual view suddenly change between the second and third frame images by the change of the selected camera caused by the movement of the virtual viewpoint. In the video of (b) made by our method, such a sudden color change does not happen. Each virtual view pixel is assigned its primary and secondary cameras according to the virtual viewpoint. This results in segmenting the whole virtual view into one-camera and two-camera regions and blending smoothly camera pixel colors by appropriate view-dependent weights.

V. Conclusion

In this paper, we proposed an effective view-dependent weighting method to smoothly blend camera pixel colors for each virtual view pixel in plane sweep. Our method utilizes the difference of the angles of the viewing rays of camera pixels for the viewing ray of a virtual view pixel instead of the angles themselves. As a result, adjacent two one-camera regions given pixel colors of their respective primary cameras are blended in their boundary as a two-camera region given weighted averages of pixel colors of their primary and secondary cameras. Our method avoids a sudden color change and a blurring effect which are serious problems in usual methods.

Currently, we are planning to generalize our method so as to use more than two cameras for appropriate boundary blending to make a high-quality virtual view.

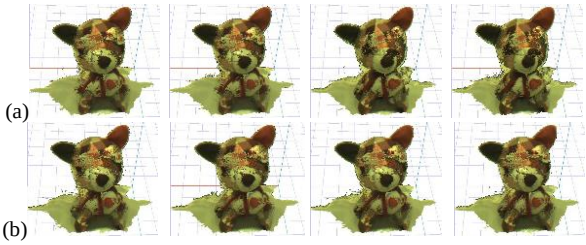


Fig. 6. Successive frame images with virtual viewpoint movement.

References

[1] R.T. Collins, “A Space-sweep Approach to True Multi-image Matching”, Proc. of Conf. on Computer Vision and Pattern Recognition 1996, pp. 358-363, 1996.

[2] R. Yang, G. Welch, and G. Bishop, “Real-time Consensus-based Scene Reconstruction using Commodity Graphics Hardware”, Computer Graphics Forum, vol. 22, no. 2, pp. 207-216, 2003.

[3] J. Woetzel and R. Koch, “Real-time Multi-stereo Depth Estimation on GPU with Approximative Discontinuity Handling”, Proc. of the 1st European Conference on Visual Media Production, pp. 245-254, 2004.

[4] M. Li, M. Magnor, and H.-P. Seidel, “Hardware-accelerated Rendering of Photo Hulls”, Computer Graphics Forum, vol. 23, no. 3, pp. 635-642, 2004.

[5] M. Li, M. Magnor, and H.-P. Seidel, “Hardware-accelerated Visual Hull Reconstruction and Rendering”, Proc. of Graphics Interface 2003, pp. 65-71, 2003.

[6] I. Geys, T.P. Koninckx, and L.V. Gool, “Fast Interpolated Cameras by Combining a GPU based Plane Sweep with a Max-Flow Regularisation Algorithm”, Proc. of the 2nd International Symposium on 3D Data Processing, Visualization, and Transmission, pp. 534-541, 2004.

[7] I. Geys and L.V. Gool, “Extended View Interpolation by Parallel Use of the GPU and the CPU”, Proc. of SPIE 5665, Videometrics VIII, 96, 2005.

[8] V. Nozick, S. Michelin, and D. Arquès, “Real-time Plane-sweep with Local Strategy”, Journal of WSCG, vol. 14, no. 1-3, pp. 121-128, 2006.

[9] Y. Liu, G. Chen, N. Max, C. Hofsetz, and P. McGuinness, “Undersampled Light Field Rendering by a Plane Sweep”, Computer Graphics Forum, vol. 25, no. 2, pp. 225-236, 2006.

[10] X. Zabulis, G. Kordelas, K. Mueller, and A. Smolic, “Increasing the Accuracy of the Space-Sweeping Approach to Stereo Reconstruction Using Spherical Backprojection Surfaces”, IEEE International Conf. on Image Processing 2006, pp. 2965-2968, 2006.

[11] M. McKenney and T. McGuire, “A Parallel Plane Sweep Algorithm for Multi-Core Systems”, Proc. of the 17th ACM SIGSPATIAL International Conf. on Advances in Geographic Information Systems, pp. 392-395, 2009.

[12] S. Jarusirisawad, V. Nozick, and H. Saito, “Real-time Video-based Rendering from Uncalibrated Cameras Using Plane-sweep Algorithm”, Journal of Visual Communication and Image Representation, vol. 21, no. 5-6, pp. 577-585, 2010.

[13] A. Irschara, M. Rumpler, P. Meixner, T. Pock, and H. Bischof, “Efficient and Globally Optimal Multi View Dense Matching for Aerial Images”, ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences, vol. I-3, pp. 227-232, 2012.

[14] C. Hane, L. Heng, G.H. Lee, A. Sizov, and M. Pollefeys, “Real-Time Direct Dense Matching on Fisheye Images Using Plane-Sweeping Stereo”, Proc. of the 2nd International Conf. on 3D Vision, vol. 1, pp. 57-64, 2014.

[15] W. Matusik, C. Buehler, and L. McMillan, “Polyhedral Visual Hulls for Real-Time Rendering”, Eurographics Workshop on Rendering 2001, pp.115-125, 2001.

[16] C. Buehler, M. Bosse, S. Gortler, M. Cohen, and L. McMillan, “Unstructured Lumigraph Rendering”, Proc. of SIGGRAPH 2001, pp.425-432, 2001.

Creating more comprehensive facial expressions by morphable 3D face model:

Transformation of 3D faces through impression transfer vectors defined by SVM

Yudai Arai , Kazuya Horii , Yoshinori Inaba
Graduate School of Science and Engineering
Hosei University
Kajino-cho 3-7-2, Koganei-shi, Tokyo 184-8584 Japan
E-mail: yudai.arai@akamatsu.info

Shigeru Akamatsu
Faculty of Science and Engineering,
Hosei University
Kajino-cho 3-7-2, Koganei-shi, Tokyo 184-8584 Japan
E-mail: akamatsu@hosei.ac.jp

***Abstract*— In this study, based on applications of the Support Vector Machine in discriminative learning, we proposed a new method for making appropriate transformations of the 3D shapes of faces to create more appealing impressions with 3D avatar faces. Through preliminary experiments we made subjective assessments of the impressions conveyed by facial expressions, and the synthesized 3D shapes of the faces generated by our proposed method were recognized more accurately than those by the previous method.**

Keywords— facial expression generation, vector representation of 3D object, principal component analysis, morphable 3D model, dimensionality reduction, support vector machine

I. INTRODUCTION

Facial expressions play an important role in all facets of human communication, even in human-machine communications. Since the facial expressions displayed on an avatar face are interpreted more accurately in 3D than in 2D, we must properly transform the 3D shape of its original neutral face to create a 3D avatar face that displays various expressions [1, 2]. Morphable 3D face models [3, 4] were introduced by learning from many 3D faces that display various expressions and impressions on personal attributes with which each transformation of the 3D shape to change the facial expressions and impressions is described by controlling just a few parameters defined as principal component scores. In our previous works [5, 6], such parameter transformation suitable to generate changes in facial appearance was introduced by impression transfer vector [7], obtained by applying linear discriminant analysis on the learning face data and uniformly defined for any input face as the transformation target. We proposed a new method to individually obtain the transformations of the parameters for each input face on the basis of positional relationship in the parametric space between the point representing the input face and the discrimination boundaries among different categories of impression classes defined by Support Vector Machine learning [8]

II. BUILDING A MORPHABLE 3D FACE MODEL FOR GENERATING FACIAL EXPRESSIONS

To build a morphable 3D face model that represents transformations due to facial expressions, we conducted 3D measurements of total M faces displaying various facial expressions with a commercial range finder (Danae-R, NEC Engineering Ltd.). With a normalization procedure [6] that consisted of feature point extraction, pose normalization, and re-sampling facial of the surface, we represented the shape of the m-th 3D face displaying any expression by N-dimensional vector $\mathbf{X}_m$ whose components indicate the 3D coordinates of the re-sampled points on the facial surface. Let $\mathbf{S}_m$ indicate a vector that describes the differences of the face shape between the m-th face that displays an expression and the neutral face:

$$\boldsymbol{S}_m = \boldsymbol{X}_m - \boldsymbol{X}_{neutral}. \quad (1)$$

We then applied Principal Component Analysis (PCA) to the set of $\mathbf{S}_m$ for $m = 1,2,\cdots,M$.
As a result of the dimensional reduction achieved by PCA, each piece of 3D shape data $\mathbf{S}_m$ was transformed into low-dimensional vector $\boldsymbol{f}_m$, whose k-th component (described as $f_{m,k}$) was obtained as Eq. (2):

$$f_{m,k} = \mathbf{U}_k^t \cdot (\boldsymbol{S}_m - \overline{\boldsymbol{S}}), \quad (2)$$

where $\bar{\mathbf{S}}$ is obtained as the average of $\mathbf{S}_m$ for $m = 1,2,\cdots,M$ and $\mathbf{U}_k$ is the k-th eigenvector of the covariance matrix obtained from $\mathbf{S}_m$ for $m = 1,2,\cdots,M$.
On the contrary, a linear combination of arbitrary parameters $\hat{f}_k$ (k=1,2,…,K) and orthonormal bases $\mathbf{U}_k$ (k=1,2,…,K), shown in Eq. (3), represents N-dimensional vector $\hat{\mathbf{S}}$ that indicates the difference in the 3D shape between the novel facial expression designated by parameters $\hat{f}_k$ (k=1,2,…,K) and the neutral face:

$$\hat{\mathbf{S}} = \bar{\mathbf{S}} + \sum_{k=1}^{K} \hat{f}_k \cdot \mathbf{U}_k. \quad (3)$$

Since an arbitrary change of each component of parameter vector $\hat{\mathbf{f}} = (\hat{f}_1, \hat{f}_2, \cdots, \hat{f}_K)$ generates a variety of 3D facial expressions, we expect that a novel expression design can be approximately represented by this morphable 3D face model.

III. IMPRESSION TRANSFORMATION OF 3D FACES BY IMPRESSION TRANSFER VECTOR

Our previous studies [5, 6, 7] confirmed that the impression transfer vector method effectively transformed the impressions of the input faces. This method was proposed to manipulate the input face either in 2D or 3D in terms of its low-dimensional parameters provided by the morphable face model. Let **e** indicate the impression transfer vector obtained for the transformation of the input face in terms of the specified factor of impressions, such as enhancing the impression of specific facial expressions. For transformations with the given impression factor, low-dimensional parameter **f** of the input face is converted to as

$$\widehat{\mathbf{f}} = \mathbf{f} + \mathcal{Q} \cdot \mathbf{e}, \tag{4}$$

where weighting factor $\mathcal{Q}$ indicates the impression transformation's intensity.

In our previous works [5, 7], impression transfer vector **e** was defined by Fisher's linear discriminant analysis as a unit vector that indicates the direction of the directly leading axis from one cluster of the training samples to another, both of which are in an adversary relationship regarding a specific factor of the perceived impression: a set of faces with a glimpse of a specific expression vs. a set of faces without. We call this approach for impression transformation the Fisher Method.

In this work, we propose a new version of an impression transfer vector defined by Support Vector Machine (SVM) learning, as shown in Eq. (5):

$$\boldsymbol{w} = \frac{1}{m}\sum_{k=1}^{m} \boldsymbol{\zeta}_k - \frac{1}{n}\sum_{k=1}^{n} \boldsymbol{\eta}_k, \tag{5}$$

where $\boldsymbol{\zeta}_k$ for k = 1,2, ⋯, indicate the support vectors in the positive domain of the parametric space that are m-th nearest from the input face, and $\boldsymbol{\eta}_k$ for k = 1,2, ⋯, indicate the support vectors in the negative domain of the parametric space that are n-th nearest from the input face. In the positive domain, face samples exist with the same impression as the input face, but in the negative domain, however, other learning samples have opposite impressions to the input face.

After the **w** being normalized to a unit vector, the weight vector was substituted for the previous impression transfer vector **e** in Eq. (4). We call this approach for impression transformation the SVM Method.

Transformation from the input face **X** with the original facial expression to the synthesized face $\widehat{\mathbf{X}}$ with a more appealing facial expression was conducted in the same framework of impression transformation by the impression transfer vector (Eqs. (6), (7), (8), and (9)):

$$\mathbf{S} = \mathbf{X} - \mathbf{X}_{neutral}, \tag{6}$$

$$\mathbf{f} = (f_1, f_1, \cdots, f_K), \text{ where } f_k = \mathbf{U}_k^t \cdot (\mathbf{S} - \bar{\mathbf{S}}), \tag{7}$$

$$\hat{\mathbf{f}} = \left(\widehat{f_1}, \widehat{f_1}, \cdots, \widehat{f_K}\right), \text{ where } \widehat{\mathbf{f}} = \mathbf{f} + \mathcal{Q} \cdot \boldsymbol{w}, \tag{8}$$

$$\widehat{\mathbf{X}} = \mathbf{X}_{neutral} + \left\{\bar{\mathbf{S}} + \sum_{k=1}^{K} \widehat{f_k} \cdot \mathbf{U}_k\right\}. \tag{9}$$

IV. EXPERIMENTAL RESULTS ON FACIAL EXPRESSION GENERATION

To obtain a set of training data for building a morphable 3D face model, we conducted 3D measurements of faces with 40 people, each of whom provided a neutral face that displayed no expressions and six facial expressions that displayed four utterances and two emotions (Table 1). Although we also measured 3D faces while uttering[e], we omitted these data in this experiment because their mouth openings were much smaller than the other vowels, which complicate the discrimination.

In this experiment, we applied two types of impression transformation methods, the Fisher and SVM Methods, to the averaged faces of the 40 measured 3D faces to create their facial expressions. Fig. 1 shows the set of original 3D faces, denoted as $\mathbf{X}_m$ in Eq. (6), each of which displays non-emphasized facial expressions on the average face. Figs. 2 and 3 show the corresponding set of the impression emphasized 3D faces denoted as $\widehat{\mathbf{X}}$ in Eq. (9), obtained as a result of the impression transformation by the previous Fisher Method and the proposed SVM Method.

In addition, we made more quantitative comparisons between the Fisher and the SVM Methods to determine whether the proposed SVM Method successfully generates more perceptible facial expressions by the 3D shape transformation scheme based on the impression transfer vector. We subjectively assessed the impression transformations of the 3D faces. We built a morphable face model from the 3D faces of 40 people, each of whom provided neutral faces and six facial expressions. By using the k-means clustering method, 40 neutral faces were clustered into four spatially dispersed groups in the parameter space. And we selected the face whose Euclidean distance from the origin in the parametric space obtained by PCA is largest in each of the four groups. from origin in PCA space. These were called Persons A, B, C, and D (Fig. 4)

For the two expression categories, i.e., mouth-closed and mouth-opened smiling, we conducted subjective evaluation tests on the strength of the impressions presented by the facial expressions that were synthesized on these four faces by three different methods: the Fisher Method, the SVM Method, and a non-emphasized version of the expressions. As test subjects, ten university student quantified the intensities of the impressions perceived from the three types of generated facial expressions. We adopted Thurston's paired comparison method for impression ratings. As shown in Figs. 5 and 6, the values individually obtained for each facial expression indicate the merits of each impression transformation method in terms of the intensity of the impression presented by the facial expressions.

V. CONCLUSION

Through these subjective assessments of the impressions presented by the facial expressions, the synthesized expressions made by our proposed method (SVM Method) were found to be more strongly perceptible in many instance than those made by the previous method.

The reason why the SVM method obtained a good mental measure seems to be that the facial expression generated by the SVM method generated facial features such as screwed up eyes, raised corners of the mouth. In addition, the SVM method tends to generate natural facial expressions because facial shapes which are similar to the input novel facial expression are used.

Table 1 Six types of facial expressions investigated in experiment

1	Uttering [a]	2	Uttering [i]
3	Uttering [u]	4	Uttering [o]
5	Mouth-closed smiling	6	Mouth-opened smiling

Fig. 1 Original 3D faces X_m displaying non-emphasized facial expressions

Fig. 2 Impression emphasized 3D faces obtained by impression transformation by conventional Fisher Method

Fig. 3 Impression emphasized 3D faces obtained by impression transformation by proposed SVM Method

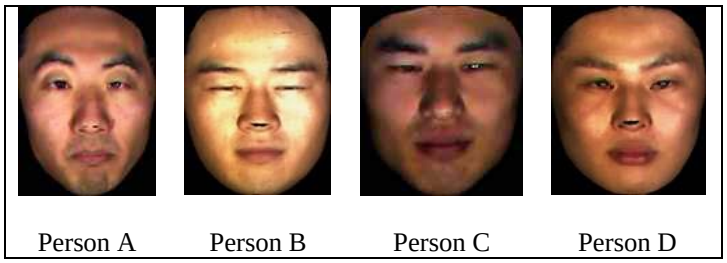


Fig. 4 Four 3D faces used for impression testing

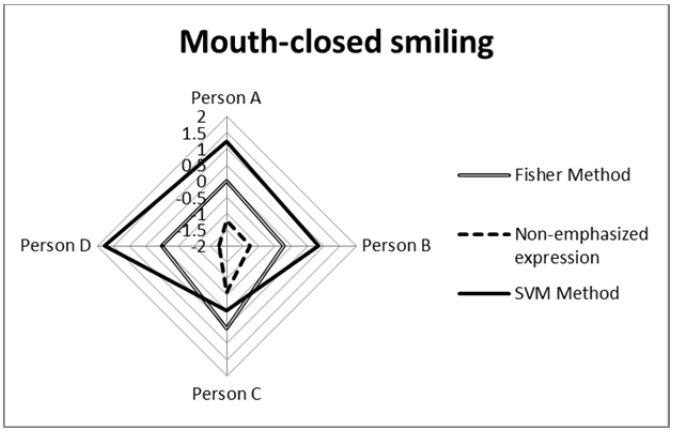


Fig. 5 Results of paired comparison reflecting intensity of perceived impression (mouth-closed smiling)

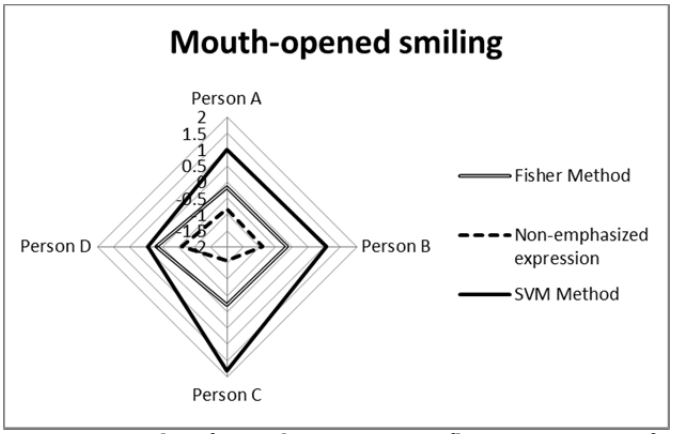


Fig. 6 Results of paired comparison reflecting intensity of perceived impression (mouth-opened smiling)

REFERENCES

[1] T. Masuda, T. Takahashi, and S. Moroi, "Evaluation of the Making of Short Facial Animation Movies based on Facial Action Coding System," Forum on Information Technology, Vol. 6, No. 3, pp. 313-314, August 2007

[2] S. Ahn and S. Ozawa, "Generating another Person Facial Expressions Based on Estimation of Muscular Contraction Parameters from Face Image," The Transactions of the Institute of Electronics, Information and Communication Engineers D-II, Vol. J8-D-II, No. 10, pp. 2081-2088, October 2005 (in Japanese)

[3] V. Blanz and T. Vetter, "A Morphable Model for the Synthesis of 3D Faces," SIGGRAPH'99, 1999

[4] M. Walker and T. Vetter, "Portraits made to measure: manipulating social judgments aboutindividuals with a statistical face model," Journal of Vision, vol. 9, pp. 1-13, 2009

[5] Y. Okada, N. Takeshita, T. Akita, and S. Akamatsu, "Automatic generation of 3D morphable model and application to impression conversion of a face," Journal of Japanese Academy of Facial Studies KAOGAKU, vol. 7, no. 1, pp. 111-120, February 2007 (in Japanese)

[6] Y. Inaba, J. Kochi, H. Ishi, J. Gyoba, and S. Akamatsu, "Impression-driven Design Scheme for a Class of 3D Objects Based on Morphable 3D Shape Model, and its Automatic Buildup by Supplementary Feature Sampling," Proc. of IWAIT2009, Jan. 2009

[7] T. Kobayashi, M. Ohzu, S. Ohtake, and S. Akamatsu, "Impression Transformation of a Face Based on Discriminant Analysis on Separately Coded Representation of Facial Shape and Texture," Proc. of Intl. Conf. Automatic Face and Gesture Recognition (FG2004), pp. 711-716, May 2004

[8] K. Tsuda, "Overview of Support Vector Machine," The Journal of the Institute of Electronics, Information, and Communication Engineers, Vol. 83, No. 6, pp. 460-466, June 2000 (in Japanese)

3D reconstruction from a video taken by low-priced drone quadrotor using outlier-less estimation of corresponding feature points

Takafumi KONNO
Graduate school of Engineering
Iwate University
4-3-5 Ueda, Morioka, Iwate 020-8551 JAPAN
Email: kontaka@mips.cis.iwate-u.ac.jp

Akio KIMURA
Faculty of Engineering
Iwate University
4-3-5 Ueda, Morioka, Iwate 020-8551 JAPAN
Email: kimura@cis.iwate-u.ac.jp

***Abstract*—In this paper, we focus on 3D reconstruction issue using a low-priced drone quadrotor such as AR.Drone manufactured by Parrot Corporation (Fig.1). In this case, there are several problems: the resolution of a camera equipped with the drone is not so high, errors of corresponding feature points between image frames easily occur (called "outlier") because of blurring or shaking the video, and the final accuracy of 3D reconstruction decreases. Therefore we focus on image frames cut out from the drone video in time series and adopt the estimation method with smaller outliers (or outlier-less estimation method) for corresponding feature points using a combination of FAST[1] (Features from Accelerated Segment Test) and an optical flow. Furthermore, we perform some experiments with real images to verify the validity of our method.**

Keywords—3D reconstruction; Drone; Quadrotor; Bundle adjustment; Optical flow;

I. INTRODUCTION

3D reconstruction from 2D images is one of the most important research subjects in computer vision or augmented reality, and thus many studies have been conducted and successful results have been known such as SfM (Structure from Motion), BA (Bundle Adjustment), SLAM (Simultaneous Localization and Mapping), and MVS (Multi-View Stereo). These methods are quite effective when a good photographing condition is ensured, and it might be possible to obtain accurate 3D information comparable to the one obtained by an expensive laser range sensor. However, they are not always valid when we need to process a relatively low-quality video, e.g. the ones taken by a low-priced drone quadrotor (Fig.1). In such a case, there are several problems: the maximum resolution of a drone camera is not so high (1280×720 pixels per frame in the AR. Drone case), errors of corresponding feature points between image frames easily occur (called "outlier") because of blurring or shaking the video, and the final accuracy of 3D reconstruction decreases. Furthermore, proper steering lessons are necessary for the drone operator to make the drone flight stable.

Therefore in this paper, for realizing easier 3D reconstruction using a low-priced drone, we establish a method to effectively reconstruct 3D entities from a relatively low-resolution video taken by beginner drone operators. To be more concrete, we take the following two measures into conventional method using BA. The first measure is to adopt FAST[1] (Features from Accelerated Segment Test) as an extraction method of feature points and the second one is to use optical flows between temporal adjacent images with the FAST results. This means that we examine both estimated point correspondences based on the feature points extracted by FAST and the ones by optical flows, remove outliers and determine outlier-less point correspondences as much as possible. By inputting such outlier-less corresponding points determined by our method to BA, it is expected to achieve efficient 3D reconstruction from a drone video.

In the following sections, we outline for projective geometry and subsequently describe the details of our method.

Fig.1. AR.Drone quadrotor

II. CAMERA MODEL AND PRELIMINARY KNOWLEDGE

A. Notation

A space point in the world XYZ coordinate system is described as $\boldsymbol{X} = [X, Y, Z]^{\mathrm{T}}$ using the capital letters of alphabet. The point in a uv image coordinate system corresponding to the space point is described as $\boldsymbol{m} = [u, v]^{\mathrm{T}}$ using the small letters. The homogenous vector for space point $\boldsymbol{X}$ is described as $\widetilde{\boldsymbol{X}} = [X, Y, Z, 1]^{\mathrm{T}}$ and that for image point $\boldsymbol{m}$ as $\widetilde{\boldsymbol{m}} = [u, v, 1]^{\mathrm{T}}$.

B. Camera projection matrix

The processing for space point $\boldsymbol{X}$ projected to image point $\boldsymbol{m}$ by a camera is modeled by using 3×4 matrix P as follows:

$$\lambda\tilde{\boldsymbol{m}} = P\tilde{\boldsymbol{X}} = K[R, \boldsymbol{t}]\tilde{\boldsymbol{X}}\,, \tag{1}$$

where λ has the depth information of a 3D space. The degree of freedom of projection matrix P is 11 because the matrix can only be defined meaningfully up to a scaling factor. K is called "camera intrinsic matrix" including the parameters unique to the camera to be used (in this paper, K is already known). Matrix $[R, \boldsymbol{t}]$ is called "camera extrinsic matrix" representing transform between the world coordinate system and the camera coordinate system, where matrix R represents rotation and vector $\boldsymbol{t}$ represents translation.

C. Epipolar constraint and fundamental matrix

If two images of the same scene are taken from different viewpoints and $\boldsymbol{m}\,(u, v)$ of the first image and $\boldsymbol{m}'\,(u', v')$ of the second image are the same point in the scene, the following epipolar equation is satisfied:

$$\tilde{\boldsymbol{m}}'^{\mathrm{T}} F\tilde{\boldsymbol{m}} = 0\,, \tag{2}$$

where F is a 3×3 homogenous matrix of rank 2, called "fundamental matrix." As K is known in this paper, $\tilde{\boldsymbol{m}}_k = K^{-1}\tilde{\boldsymbol{m}}$ is obtained by multiplying the image point by the inverse matrix K^{-1}. In this case, we obtain the following projection model in a new image coordinate system from Eq.(1):

$$\lambda\tilde{\boldsymbol{m}}_k = K^{-1}K[R, \boldsymbol{t}]\tilde{\boldsymbol{X}} = [R, \boldsymbol{t}]\tilde{\boldsymbol{X}}\,. \tag{3}$$

Since the epipolar constraint of Eq.(2) is satisfied in the new image coordinate system, we can write the following equation:

$$\begin{aligned}\tilde{\boldsymbol{m}}_k'^{\mathrm{T}} E\tilde{\boldsymbol{m}}_k &= (K^{-1}\tilde{\boldsymbol{m}}')^{\mathrm{T}} E(K^{-1}\tilde{\boldsymbol{m}}) \\ &= \tilde{\boldsymbol{m}}'^{\mathrm{T}}(K^{-\mathrm{T}}EK^{-1})\tilde{\boldsymbol{m}} = 0\,.\end{aligned} \tag{4}$$

Matrix E is a fundamental matrix for the coordinate system normalized by K^{-1}, and it is particularly called "essential matrix." When a set of corresponding points $\tilde{\boldsymbol{m}} \leftrightarrow \tilde{\boldsymbol{m}}'$ is determined for Eq. (2), the following equation can be described:

$$\begin{bmatrix} u' & v' & 1 \end{bmatrix} \begin{bmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = 0\,.$$

When $\boldsymbol{f} = [f_{11} \quad \cdots \quad f_{33}]^{\mathrm{T}}$, the above equation can be converted to $[uu' \;\; vu' \;\; u' \;\; uv' \;\; vv' \;\; v' \;\; u \;\; v \;\; 1]\;\boldsymbol{f} = 0$. If $\tilde{\boldsymbol{m}}_i \leftrightarrow \tilde{\boldsymbol{m}}_i'\;(i = 1, 2, \cdots)$ is determined by using multiple point correspondences, the following equation can be obtained:

$$\begin{bmatrix} u_1u_1' & v_1u_1' & u_1' & u_1u_1' & v_1u_1' & v_1' & u_1 & v_1 & 1 \\ u_2u_2' & v_2u_2' & u_2' & u_2u_2' & v_2u_2' & v_2' & u_2 & v_2 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u_iu_i' & v_iu_i' & u_i' & u_iu_i' & v_iu_i' & v_i' & u_i & v_i & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u_8u_8' & v_8u_8' & u_8' & u_8u_8' & v_8u_8' & v_8' & u_8 & v_8 & 1 \end{bmatrix} \boldsymbol{f} = \boldsymbol{0}\,.$$

Thus, it is possible to estimate F if there are eight pairs of corresponding points and E can also be estimated because $F = K^{-\mathrm{T}}EK^{-1}$.

D. Optical flow

Since moving pictures are images taken in time series, it can be differentiated with respect to time. When the brightness of the image points corresponding to the same point in a three-dimensional space is I and we assume that the brightness does not change between temporal adjacent images, the following spatiotemporal equation is satisfied:

$$\frac{dI}{dt} = I_x u + I_y v + I_t = 0\,,$$

where I_x, I_y, and I_t represent partial differentiations to horizontal, vertical and time axis directions respectively, and $\boldsymbol{u} = [u, v]^{\mathrm{T}} = [\partial x/\partial t\,, \partial y/\partial t]^{\mathrm{T}}$ represents a motion vector in horizontal and vertical directions. $\boldsymbol{u}$ is called "optical flow."

III. 3D RECONSTRUCTION ALGORITHM

In this paper, we perform 3D reconstruction in the following steps:

1. Image frames are cut out in chronological order from a drone video and the feature points are extracted from each of the images by using FAST.
2. The feature point correspondence is estimated by combining the result of FAST and optical flow.
3. 3D reconstruction is performed by inputting the feature point correspondence determined in step 2 to the BA algorithm.

A. Estimation of feature point correspondence

The key of 3D reconstruction using multiple images is the estimation of feature point correspondence. Since the resolution of a video taken by a low-priced drone is not so high, lots of wrong correspondences would be defined if we do not devise anything. So, we focus on the fact that image frames cut out from a drone video are time-series images. It is possible to easily calculate the optical flow between adjacent images in time axis. In addition, we adopt FAST that can extract the feature points very quickly from images and verify the feature point correspondences between the images by the combination. To be more concrete, first, feature points are extracted from the first image using FAST in two temporal adjacent images of the same scene taken from different viewpoints and the optical flow of the second image is calculated for their point group using the Lucas-Kanade method. The corresponding point coordinates on the second image calculated by the optical flow are compared with the feature point coordinates on the second image extracted by FAST, and the correspondence is regarded as an outlier if their distance is greater than a pre-defined threshold.

By the measures described above, it can be expected to define correspondences effectively between adjacent image points even from a video taken in relatively bad photographing condition.

B. Formulation

An image point obtained by photographing 3D point $\boldsymbol{X}_j (j = 0, \cdots, M-1)$ in the i-th camera $P^i = K[R_i, \boldsymbol{t}_i]$ $(i = 0, \cdots, N-1)$ is described as $\boldsymbol{m}_j^i$ and this process is modeled as follows:

$$\lambda_j^i \widetilde{\boldsymbol{m}}_j^i = P^i \widetilde{\boldsymbol{X}}_j \ ,$$

while $P^0 = K[I, \boldsymbol{0}]$ is promised. Here, the goal is to restore projection matrix P^i and 3D point $\boldsymbol{X}_j$ from the image point group $\boldsymbol{m}_j^i$. So, we consider the minimization problem of the following function:

$$\min_{\hat{P}^i, \hat{\widetilde{\boldsymbol{X}}}_j} \sum_{ij} \left\| \hat{P}^i \hat{\widetilde{\boldsymbol{X}}}_j - \widetilde{\boldsymbol{m}}_j^i \right\| , \tag{5}$$

where $\hat{P}^i$ represents the projection matrix and $\hat{\widetilde{\boldsymbol{X}}}_j$ represents the 3D point estimated from the correspondence of image group $\boldsymbol{m}_j^i$. Thus, $\hat{P}^i \hat{\widetilde{\boldsymbol{X}}}_j$ is the re-projection point to the image and Eq. (5) represents the minimization of re-projection errors.

It is well known that optimal solution of Eq. (5) is determined by the non-linear iterative method such as Levenberg-Marquart method if optimal initial values can be selected. This is called "SfM" or "Bundle Adjustment." In this paper, we perform the minimization of Eq. (5) using SSBA[5] (Simple Sparse Bundle Adjustment), which is one of BA. In addition, we show how to obtain the optimal initial values in the following section.

C. Estimation of initial value

Here, we consider estimation of $\hat{P}^1, \hat{\widetilde{\boldsymbol{X}}}_j$ from point correspondence $\widetilde{\boldsymbol{m}}_j^0 \leftrightarrow \widetilde{\boldsymbol{m}}_j^1$ between two images determined by the method of section 3-A. Projection to the image point can be modeled as the following equation:

$$\begin{cases} \lambda_j^0 \widetilde{\boldsymbol{m}}_j^0 = P^0 \hat{\widetilde{\boldsymbol{X}}}_j \\ \lambda_j^1 \widetilde{\boldsymbol{m}}_j^1 = \hat{P}^1 \hat{\widetilde{\boldsymbol{X}}}_j \end{cases} . \tag{6}$$

As camera parameter K is known,

$$\begin{cases} \lambda_j^0 \left(K^{-1} \widetilde{\boldsymbol{m}}_j^0 \right) = [I|\boldsymbol{0}] \hat{\widetilde{\boldsymbol{X}}}_j \\ \lambda_j^1 \left(K^{-1} \widetilde{\boldsymbol{m}}_j^1 \right) = [R_1|\boldsymbol{t}_1] \hat{\widetilde{\boldsymbol{X}}}_j \end{cases} \tag{7}$$

is satisfied and essential matrix E between the 0th and the 1st images can be determined by using the eight-point algorithm[3] and others. As $E = [\boldsymbol{t}]_\times R$, it is possible to estimate $P^1 = K[R_1, \boldsymbol{t}_1]$ using singular value decomposition of E. In addition, the following equation is satisfied by Eq. (6):

$$\begin{bmatrix} P^0 & -\widetilde{\boldsymbol{m}}_j^0 & \boldsymbol{0} \\ \hat{P}^1 & \boldsymbol{0} & -\widetilde{\boldsymbol{m}}_j^1 \end{bmatrix} \begin{bmatrix} \hat{\widetilde{\boldsymbol{X}}}_j \\ \lambda_j^0 \\ \lambda_j^1 \end{bmatrix} = \boldsymbol{0}$$

It is possible to estimate 3D point $\hat{\boldsymbol{X}}_j$ by solving this equation. Since it is possible to estimate $\hat{P}^k, \hat{P}^{k+1}$, and $\hat{\widetilde{\boldsymbol{X}}}_j$ from the correspondence of $\widetilde{\boldsymbol{m}}_j^k \leftrightarrow \widetilde{\boldsymbol{m}}_j^{k+1}$ by the same steps, we can perform the 3D reconstruction by inputting the initial value into the minimization problem of Eq. (5).

IV. Evaluation experiment

To verify the effectiveness of our method, we performed 3D reconstruction from a video of a house photographed by AR. Drone quadrotor. Frist, we manually cut out multiple sets of two image frames that are adjacent in the time axis from the drone video and extracted the feature points from the frames. An example is shown in Fig.2. We calculated the optical flows of the first and second images (Fig.3 (upper)) and showed the result of applying the process of section 3-A (Fig.3 (lower)). Observing these results, it can be confirmed that the wrong flows are eliminated. In addition, we show the result of defining the corresponding points by using only SIFT[4] in Fig.4 (upper) and those by using the method of section 3-A in Fig.4 (lower). Lots of wrong correspondences exist in the result of SIFT, but those are successfully removed in the result of our method. We find that it is possible to estimate the outlier-less feature point correspondence as a whole.

We calculated the initial value by the method of section 3-C using corresponding point groups estimated by our method and minimized the process of section 3-B. The estimated visualization result of 3D point groups is shown in Fig. 5. The exterior wall parts of the house were successfully restored and good results were obtained.

Fig.2. Two temporal adjacent images are cut out from a drone video.

V. Conclusion

In this paper, we estimated outlier-less corresponding feature points from the video taken by low-priced AR. Drone quadrotor and examined the 3D reconstruction method by using the points. Especially in the feature point correspondence between the images, we showed an outlier-less estimation method using combination of FAST and optical flow. We confirmed the effectiveness of our method to some extent by the experiments with real drone videos, but there is also a problem that not so many 3D points are restored by only correspondence between adjacent images. In the future, we apply our method to more frames and need to carry out dense reconstruction experiments.

Fig.3. Calculated optical flow (upper) and removed outliers (lower).

Fig.4. Matching of SIFT (upper) and matching results of our method (lower).

Fig.5. Left: 3D points viewed from the left side, Center: view from the top, and Right: view from the right side.

REFERENCES

[1] E. Rosten and T. Drummond, "Machine Learning for High-Speed Corner Detection," Computer Vision ECCV 2006, pp.430-443, 2006.

[2] B. D. Lucas and T. Kanade, "An Iterative Image Registration Technique with an Application to Stereo Vision," Proc 7th International Joint Conferences on Artificial Intelligence 1981, pp.674-679, 1981.

[3] R. Hartley and A. Zisserman, "Multiple View Geometry in Computer Vision, second edition", Cambridge University Press, 2004.

[4] D. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints", International Journal of Computer Vision, 2004.

[5] http://www.ics.forth.gr/~lourakis/sba/

Upright Detection of In-Plain Rotated Face Images for Setting the EXIF Orientation Flag

Yoshihiro Shima *Hiromitsu Shukuya*
School of Science and Engineering
Meisei University
Hino, Tokyo, Japan
shima@ee.meisei-u.ac.jp

***Abstract*—Digital Cameras and smart-phones with orientation sensors allow auto-rotation of portrait images. Auto-rotation of portrait is done by using the image file's metadata, exchangeable image file format (EXIF). The output of these sensors is used to set the EXIF orientation flag to reflect the positioning of the camera with respect to the ground. Unfortunately, software program support for this feature is not widespread or consistently applied. Our research goal is to create the EXIF orientation flag by detecting the upright direction of face images having no orientation flag and is to apply the software of organizing photos. In this paper, we propose a novel upright detection scheme for face images that relies on generation of rotated images in four direction and part-based face detection with Haar-like features. Inputted images are frontal faces and these images are in-plain rotated in four possible direction. The process of upright detection is that among four possible rotated images, if only one rotated image is accepted in face detection and other three rotated images are rejected, the upright direction is obtained from the accepted direction. Rotation angle of EXIF orientation is, 0 degree, 90 degree clockwise, 90 degree counter-clockwise, or 180 degree. Experimental results on 450 face image samples show that proposed method is very effective in detecting upright of face images with background variations.**

Keywords—upright; face image; facial parts; geometric structure; image rotation; organizing photoss

I. INTRODUCTION

The camera is equipped with an orientation Sensor that detects the orientation of an image shot and automatically rotates it to the correct viewing orientation in the display. The orientation Sensor automatically detects whether a picture is being taken horizontally or vertically. The sensor also automatically orients the image correctly for previewing on the camera's LCD screen or a PC monitor[1]. Exchangeable image file format (EXIF) is a standard that specifies the formats for images, and ancillary tags used by digital cameras (including smart-phones), scanners and other systems handling image files recorded by digital cameras[2]. Many newer digital cameras have a built-in orientation sensor. The output of this sensor is used to set the EXIF orientation flag in the image file's meta data to reflect the positioning of the camera with respect to the ground. The value of the EXIF orientation flag is specified as 1, 3, 6, or 8 ,which presents as horizontal (normal), rotate 180°, rotate 90° clockwise, or rotate 270° clockwise, respectively[2]. It is a 2-axis tilt sensor, allowing 4 possible orientations to be detected. Digital Cameras with orientation sensors allow auto-rotation of portrait images.

However, support for this auto-rotation feature is not widespread or consistently applied[3][4,5]. While there are so many software programs available today that display JPEG images, only a subset of them interpret the EXIF orientation flag. Many programs simply display the JPEG image as it is stored, and completely ignore any extra details stored in the file's metadata. Some software do not support the EXIF orientation flag, and, what is worse, if image rotation is performed by using the rotate clockwise buttons, the EXIF orientation flag may be removed in some cases. Due to the above reasons, our research goal is to create the EXIF orientation flag by detecting the upright direction of face images. Face images are allowed having no orientation flag, and are applied to the software of organizing and viewing photos.

Our approach of detecting the upright direction is based on face detection. Face detection methods usually handle two major types of head rotation: (1)out-of plane (left-right) rotation; (2) in-plane rotation. As the EXIF orientation flag shows the type of in-plane rotation, our face detection method handle the type of in-plane rotation. In recent years, face detection and expression recognition technology has rapidly developed[6] [7]. In particular, face detection has been applied to grouping and selecting the photos in photo organizing software such as accurate face extraction for face expression recognition as well as a number of various areas for the detection and tracking of persons in images. Some of the representative algorithms include the Boost-based algorithm in which faces are detected using learned data [8,9,10]. To handle in-plane rotated faces, the upright face detection framework was proposed with using a new set of ±26.565 ° Haar-like features [11]. The part-based approaches treat a face as a combination of some important facial parts: eyes, mouth and nose. These methods extract the facial parts and detect faces by analyzing the relationships of these parts[12]. However, These methods are not applied to create the EXIF orientation flag, and they cannot be utilized in real photo organizing applications to achieve the auto-rotation of face images.

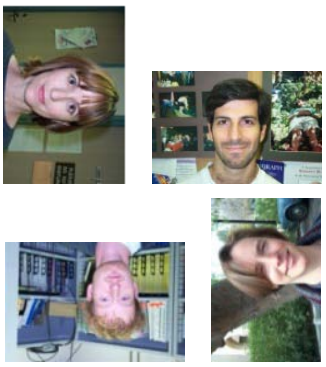
In this paper, we propose a novel upright detection scheme for face images that relies on generation of rotated images in four direction and whole-part-based face detection using Haar-

like features[8]. Inputted images are frontal faces and these images are in-plain rotated in four possible direction. The process of upright detection is that among four possible rotated images, if only one rotated image is accepted in face detection and other three rotated images are rejected, the upright direction is obtained from the accepted direction. Furthermore, experimental results on CALTEC frontal face image samples[13] are shown that the accuracy of upright detection is sufficient to apply photo organizing software.

II. Upright Detection Method for Rotated Face Images

A. *Outline of Upright Detection Methd for Rotated Face Images*

Photo organizing software include customizable ways to organize and view images. Photos are selected and grouped from categories, keywords, tags, color codes and more. It is convenient to have editing tools in photo organizing software. It is also convenient to have options within the software to send photos to social networking sites. Fig. 1 shows the one role of photo organizer, that is auto-rotation for mix of rotated and upright face images. Our research goal is to create the EXIF orientation flag by detecting the upright direction of face images having no orientation flag and is to apply the software of organizing photos. Rotation angle of EXIF orientation is, 0 degree, 90 degree clockwise, 90 degree counter-clockwise, or 180 degree.



(a) Photo images with mix of in-plain rotated and upright

(Before organizing photos)

(b) Upright images (After organizing photos)

Fig. 1. Upright of rotated face images.

Fig. 2 shows the flowchart of upright face images. Inputted images are frontal faces and allowed in four possible rotation. Inputted image is rotated by every 90 degree and four rotated images are made. Part-based face detection process is executed for each rotated images, and the results of accept or reject of face detection are saved with each rotation angle. After that, upright decision process is executed. In upright decision process, among four possible rotated images, if only one rotated image is accepted and other three rotated images are rejected in face detection, the upright direction is obtained from the accepted direction.

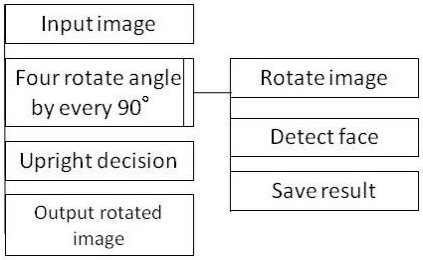
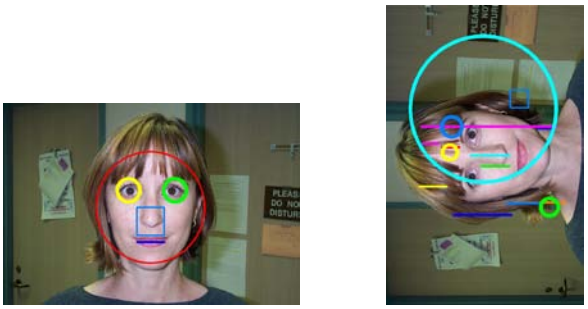


Fig. 2. Flowchart of upright face images.

(a) Inputted Image (rotated)

(b) Rotated by 0 °; reject



(c) Rotated by 90 ° counter-clockwise; accept

(d) Rotated by 180 °; reject

(e) Rotated by 90 ° clockwise; reject

(f) Outputted Image (upright)

Fig. 3. Outline of upright face images by face detection.(The result of part-based face detection shows figures; thin circle:whole face, square:nose, bold circle:eye, bold line:mouth)

Fig. 3 shows the outline of upright face images by face detection. Fig. 3(a) shows inputted image. It is assumed that the photo is taken by holding the camera vertically. Inputted image is rotated by every 90 degree and four rotated images

are made. Fig. 3(b) shows the image rotated by 0 degree. Fig. 3(c) shows the image rotated by 90 degree counter-clockwise. Fig. 3(d) shows the image rotated by 180 degree, and Fig. 3(e) shows the image rotated by 90 degree clockwise. Face detection is accepted for the image rotated by 90 degree counter-clockwise in Fig. 3(c). Other three rotated images are rejected by face detection, which are shown in Fig. 3(b)(d)(e). In upright decision, the rotated image shown in Fig. 3(c) is decided as upright and the image rotated by 90 degree counter-clockwise is outputted.

B. *Face Detection based on Geometric Structure of Facial Parts*

Our face detection process is the combination of whole face detection and facial parts detection. Fig. 4 shows the geometric structure of facial parts. Thin circles show the whole faces detected. Squares, bold circles, and bold lines show the nose, eyes, and mouth, respectively. Extracted features are the center location and the size of these whole face and parts. The geometric constraints between parts and whole figures are presented and by using these constraints the candidates of whole face and parts are narrowed down. Fig. 5 shows the flowchart of face detection based on facial parts structure. Firstly, classifier for whole face is provided and detection of whole face is executed. Secondly, classifier for facial parts such as nose, eyes, and mouth is provided and detection of facial parts is executed. Finally, The geometric constraints between parts and whole figures are applied to the candidates and the accepted whole face and facial parts are extracted.

Fig. 4. Geometric structure of facial parts. (thin circle:whole face, square:nose, bold circle:eye, bold line:mouth)

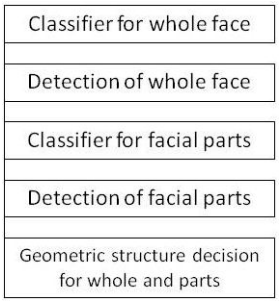


Fig. 5. Flowchart of face detection based on facial parts structure.

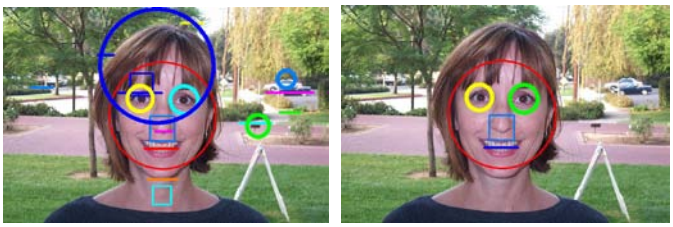
III. Experimental Results of Upright Rotated Images

Frontal face dataset, CALTEC Faces 1999[12], is used for experiment. This dataset contains 450 frontal face color images of 27 or so unique people under with different light, expression, and background. The image size is 896 x 592 pixels and the file is Jpeg format. All sample face images are upright originally, so for upright experiment, sample images are rotated in four direction in advance. Upright experiment program is based on OpenCV2.1 and the sample source named facedetect.cpp is used for face detection process. Target objects are whole face, nose, eyes and mouth. We use haarcascade_frontalface_alt.xml, haarcascade_mcs_nose.xml, etc. , as the object classifier in OpenCV2.1 library[9,10].

A. *Experimental Results of Face Detection based on Geometric Structure of Facial Parts*

Fig. 6 shows examples of candidates narrow-down by geometric structure of facial parts. The plural candidates of whole image and facial parts are detected in Fig. 6(a)(c). In the background, so many wrong whole face and facial parts are detected. Those candidates are error. By using geometric structure, those candidates are narrowed down and single candidate of whole face and nose are detected.

(a) Whole face and parts detection (b) Narrow-down by Geometric structure decision



(c) Whole face and parts detection (d) Narrow-down by Geometric structure decision

Fig. 6. Examples of candidates narrow-down by geometric structure of facial parts.

TABLE I shows the accept rate of face detection in four-direction. In case of rotate angle 0 degree, number of accept samples is 439 and the number of reject samples is 11. Thus accept rate is 97.6% for 450 upright face images. In case of rotate angle 90 degree or 180 degree, the number of accept samples is 5 to 6, respectively. Those accept samples are error in face detection process.

TABLE I. ACCEPT RATE OF FACE DETECTION IN FOUR-DIRECTION

Rotated angle	0 °	90 ° counter-clockwise	180 ° (upside down)	90 ° clockwise
Number of accept samples	439	6	5	6
Number of reject samples	11	444	445	444
Accept rate	97.6%	1.3	1.1	1.3

B. *Experimental Results of Face Image Upright*

TABLE II shows the frequency distribution of accept samples in four-direction. In upright decision, among four possible rotated images, if only one rotated image is accepted and if other three rotated images are rejected in face detection, the upright direction is obtained from the accepted direction. The number of accept samples is 419, and the accept rate of upright decision is 93.1%, for the only one accept.

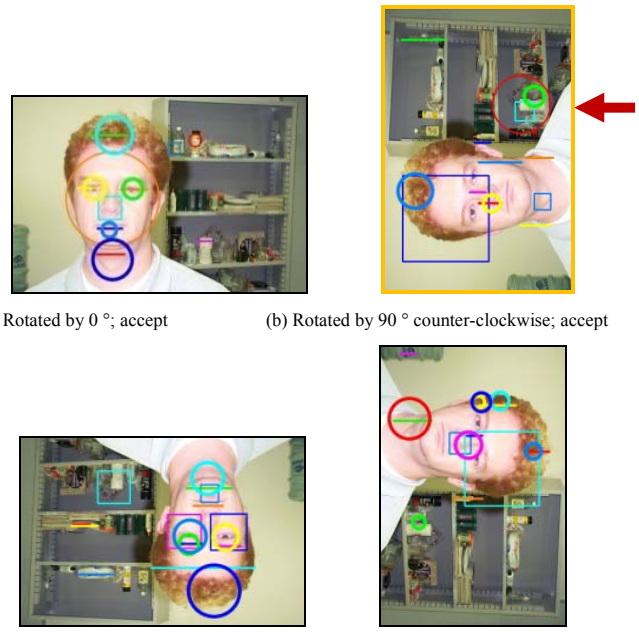
TABLE III shows the success rate of upright detection for 450 sample images. By visual check, the success rate of image is 93.1% and the reject rate is 6.9%. No error is occurred in image upright process.

TABLE II. FREQUENCY DISTRIBUTION OF ACCCEPT SAMPLES IN FOUR-DIRECTION

Number of accept angle in 4-direction	0	1	2	3	4	Total
Number of accept samples	12	419	18	1	0	450
Accept rate	2.7%	93.1	4.0	0.2	0	100

TABLE III. SUCCESS RATE OF UPRIGHT DECISION FOR SAMPLE IMAGES

	Success	Reject	Error	Total
Number of Samples	419	31	0	450
Rate	93.1%	6.9%	0	100%



(a) Rotated by 0 °; accept (b) Rotated by 90 ° counter-clockwise; accept

(c) Rotated by 180 °; reject (d) Rotated by 90 ° clockwise; reject

Fig. 7. Example of sample images for upright reject (two accept cases of face detection; Red arrow shown in (b) denotes the misdetected face.).

C. *Consideration on Reject Sample Images*

Fig. 7 shows the example of sample images for upright reject. The image rotated by 0 degree is accepted in face detection process shown in Fig. 7(a). And the image rotated by 90 degree counter-clockwise is also accepted in face detection shown in Fig. 7(b). At the bookshelf in the background, whole face and nose are detected wrongly, and the geometric structure decision is accepted. Hence the number of accept in face detection is two, upright decision is rejected.

IV. CONCLUSION

In this paper, we propose a novel upright detection scheme for face images that relies on generation of rotated images in four direction and part-based face detection with OpenCV2.1 Haar-like features. Inputted images are frontal faces and these images are in-plain rotated in four possible direction. From the detected upright direction, the EXIF orientation flag can be created easily for photo organizing software. Experimental results on 450 frontal face image samples show that proposed method is very effective in detecting upright of face images with background variations. Future works are on upright detection of natural scenes and on producing intelligent photo organizer software.

REFERENCES

[1] Canon, "Canon Camera Museum, Digital Compact Camera,Intelligent Orientation Sensor," http://www.canon.com/camera-museum/camera/dcc/data/2003-2004/2003_ps-g5.html.

[2] EXIF.org, "Exchangeable image file format for digital still cameras: EXIF 2.2," April 2002, http://www.exif.org/specifications.html.

[3] ImpulseAdventure,"JPEG Rotation and EXIF Orientation; Digital Cameras with Orientation Sensor," http://www.impulseadventure.com/photo/exif-orientation.html.

[4] Google, "Organize, edit and share your photos; Picasa 3.9 – Now with Google+ sharing and tagging ," http://www.google.com/intl/en/picasa/

[5] Microsoft, "Windows Photo Gallery," http://windows.microsoft.com/ja-jp/windows7/photo-album.

[6] Pradipta K. Banerjee, "An approach of eye detection under varying in-plane rotation of face image," in Proc. of 2011 International Conference on Recent Trends in Information Systems, pp.235-240 , Dec 2011.

[7] Shaoyi Du, Nanning Zheng, Qubo You, Yang Wu, Maojun Yuan, Jingjun Wu, "Rotated Haar-Like Features for Face Detection with In-Plane Rotation," Interactive Technologies and Sociotechnical Systems Lecture Notes, in Computer Science, Volume 4270, pp 128-137, 2006.

[8] Viola, P. and Jones., M. J., "Rapid Object Detection using a Boosted Cascade of Simple Features," in Proc. of Computer Vision and Pattern Recognition (CVPR2001), Vol. 1, pp.511–518 (2001)..

[9] OpenCV documentation, "Haar Feature-based Cascade Classifier for Object Detection," http://docs.opencv.org/modules/objdetect/doc/cascade_classification.html.

[10] Adrian Kaehler, Gary Bradski, "Learning OpenCV Computer Vision in C++ with the OpenCV Library," O'Reilly Media, October 2013.

[11] Shaoyi Du,Nanning Zheng, Qubo You, Yang Wu, Maojun Yuan, Jingjun Wu,"Rotated haar-like features for face detection with in-plane rotation," Interactive Technologies and Sociotechnical Systems Lecture Notes in Computer Science Volume 4270, pp 128-137, 2006.

[12] Yi Ji, K. Idrissi,"Facial Expression Recognition by Automatic Facial Parts Position Detection with Boosted-LBP," in Proc. of 2009 Fifth International Conference on Signal-Image Technology & Internet-Based Systems (SITIS), pp.28-35 , Nov. 2009.

[13] California Institute of Technology, "Faces 1999 (Front) ," http://www.vision.caltech.edu/html-files/

Enhancement of Depth Map Post-Processing for 3D Video Coding

Dohoon Lee, Yoonmo Yang, and Byung Tae Oh
School of Electronics and Information Engineering
Korea Aerospace University
Goyang, Korea

Abstract— **Recently developed 3D video codecs are based on multi-view plus depth map format. In the system, it is more general to use depth maps at lower resolution than texture for the coding efficiency and complexity. Accordingly, it is necessary to adjust the reconstructed depth map to the size of texture for view rendering, where the simple dilation filter is widely used for this purpose. In this paper, we first analyze the functionalities of the dilation filter, and then enhance its drawbacks. The simulation results are also given to show the performance improvement of the proposed scheme.**

Keywords – Multi-view plus Depth map, Post processing, Muli-veiw coding, 3D video coding, dilation filter

I. INTRODUCTION

Nowadays, 3D video system is getting more attentions as the next generation video system. Stereoscopic or multi-view systems are well-known examples. Stereoscopic method presents two offset images separately to the left and right eyes of the viewer, and makes viewer feels depth in space. But, it only provides a fixed view-point image to viewers. For solving this problem, multi-view system, that can provide multiple view-point images, is getting attentions. Additionally, the problem about huge data size for expression of multiple view-point images can be solved by multi-view plus depth map (MVD) system with depth-image based rendering (DIBR) technique. Therefore, MVD system mainly relies on the depth map, and is focusing to reduction about redundancy of view-point [1-3].

Depth map has some interesting characteristic in comparison with existing images. Above all, in MVD system, foreground information of depth is more important than background information, and location of edge is more important than each values of depth. So, when depth map is compressed, compression errors around edge cause severe quality degradation in rendered images. Because of this reason, a set of filters for depth map have been continuously proposed [4-6].

In this paper, we first analyze the gray-scale dilation filter [4]. In 3D video coding, methods such as lower resolution depth than texture is sometimes used for the coding efficiency and complexity. But, in this procedure, edge information of depth is more damaged, and the simple dilation filter as post processing is widely used for solving the problem in the rescaling of the reconstructed depth maps. But, the simple dilation filter has some problems, so we propose the modified method for the further coding gain.

The rest of the paper is organized as follows. In Section II, we first present and analyze the functionalities of the dilation filter in 3D video coding, and then introduce the proposed modifications with the additional morphological processing. Experimental results are shown in Section 3. The concluding remarks are given in Section 4.

II. PROPOSED FILTER

A. Dilation filter

As mentioned above, depth map is often distorted by the compression process in 3D video coding. Moreover, lower resolution of depth map than texture is preferred for the coding efficiency and complexity, and therefore, edge profile in depth maps are severely distorted during rescaling. Consequently, they give the unpleasant visual artifacts in synthesized views.

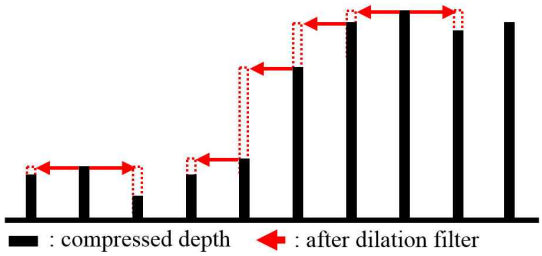
Dozens of studies have been focusing this issue [4, 7-10], and the simple dilation filter takes the maximum value in neighboring pixels as the following equation (1).

$$\tilde{D}_k = \max(D_j), \quad j \in N(k) \tag{1}$$

where $\tilde{D}$ represents the filtered image of D, and $N(k)$ shows the neighboring pixels around k. The dilation operation usually uses a structuring element for expanding the shape of the input image.

B. Modified dilation filter

As mentioned above, dilation filter takes maximum value in neighboring pixels. During the process, however, pixels of background regions take unwanted foreground values. But, in MVD system, location of edge is very important to determine the warping pixel position. Therefore, we propose the background-foreground classification before dilation filtering, and only apply the dilating filter to the foreground regions as shown in Figure 1.



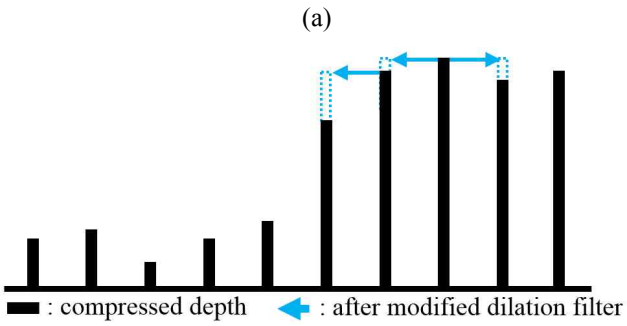


Figure 1. (a) Conventional dilation filter, (b) Modification of dilation filter

C. *Morphological filter*

Modified dilation filter takes maximum value at only foreground regions. But, clustering about foreground regions and background regions is not trivial problem due to compression errors. Simple application of the proposed scheme results in errors as shown in Figure 2.

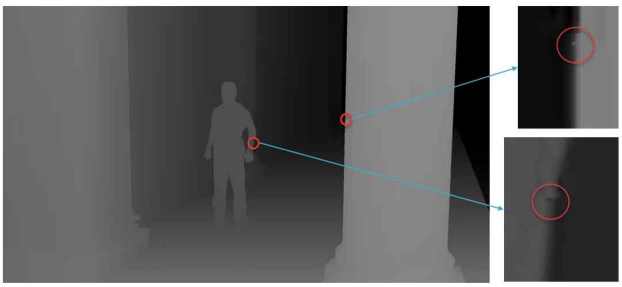


Figure 2. Errors by the modified dilation filtering.

Those errors in binary images are removed by morphological processing. Since depth map has similar characteristics to the binary image, we propose to additionally apply the grey-scale dilation and erosion filter. As a result, most errors by the modified dilation filter are effectively removed as shown in Figure 3.

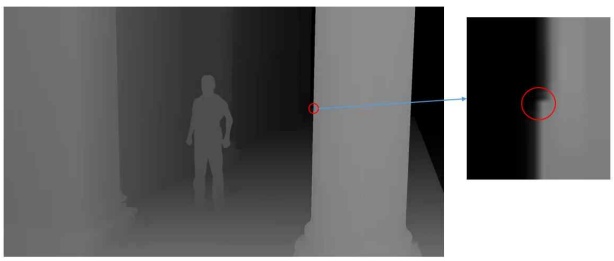


Figure 3. Results of modified dilation filtering + morphological filtering.

III. EXPERIMENTAL RESULTS

In this paper, we do experiment using the test sequences in Common Test Condition (CTC) of JCT-3V for a careful experiment and evaluation about the proposed filter. 3D-ATM v9.0 [11] is used for the Reference S/W, and 1D fast VSRS [12] is for the intermediate view rendering. Details of test sequences and experimental conditions are provided in the CTC text [13].

We set an anchor with CTC setting, and compare the dilation filter, and proposed filters with anchor. The objective comparisons are shown in Table I w.r.t. BDBR.

TABLE Ⅰ. COMPARISONS OF CODING EFFICIENTY

	Dilation filter	***Modified dilation filter***	***Modified dilation filter + morphological filter***
Poznan Hall2	-1.05	-1.29	-1.43
Poznan Street	-3.20	-3.13	-3.59
Undo Dancer	-19.74	-17.77	-18.52
GT Fly	-3.98	-1.82	-2.27
Kendo	0.11	-0.11	-0.48
Balloons	-1.65	-5.81	-6.78
Newpaper	-0.64	-3.00	-3.63
AVG.	-4.31	-4.70	-5.24

As shown in results, the proposed modification of dilation filter improves the view quality compared to the conventional dilation filter, and modified dilation filter + morphological filter even enhances the performance on average w.r.t. the coding efficiency, especially, for *Balloons* and *Newspaper* sequences. On the other hand, there is small quality degradation for *Undo Dancer* sequence, which is caused by the inaccuracy of the segmentation for foreground and background.

Subjectively, some visual errors by the modified dilation filtering are effectively removed by the proposed morphological filter as shown in Figure 3, because characteristic of depth map is similar to the binary image in local region.

IV. CONCLUSIONS

In this paper, we propose the modified dilation filter by enhancing the conventional dilation filter, and morphological filter for removing errors about modified dilation filter. Performance improvement of modified dilation filter and consecutive morphological filter is shown by the experiments.

ACKNOWLEDGEMENTS

This research war supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning (NRF-2013R1A1A1057779).

REFERENCES

[1] Text of ISO/IEC 14496-10:200X/FDAM 1 Multi-View Video Coding, I SO/IEC JTC1/SC29/WG11, Doc. N9978, Hannover, Germany, 2008.

[2] A. Smolic, K. Mueller, N. Stefanoski, J. Ostermann, A. Gotchev, G. B. Akar, G. Triantafyllidis, A. Koz, “Coding algorithms for 3DTV a surve y,” IEEE Trans. Circuits System. Video Technol., vol. 17, no. 11, pp. 1 606-1621, 2007.

[3] P. Merkle, A. Smolic, K. Muller, T. Wiegand, “Multiview video plus de pth representation and coding,” in Proc. IEEE Int. Conf. on Image Proc essing, vol. 1, no., pp. 201–204, 2007.

[4] 3D-CE3.a results on dilation filter for depth post processing, ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, Doc. JCT2-A0038, Stockh olm, 2012.

[5] K.J. Oh, S. Yea, A. Vetro, Y.S. Ho, "Depth reconstruction filter and do

wn/up sampling for depth coding in 3-D video," IEEE Signal Processin g. Lett., vol. 16, no. 9, pp. 747-750, 2009.
[6] K.J. Oh, A. Vetro, Y.S. Ho, “Depth coding using a boundary reconstruc tion filter for 3D video system,” IEEE Trans. Circuits System. Video T echnology, vol. 21, no. 3, pp. 350-359, 2011.
[7] J. Kopf, M. Cohen, D. Lischinski, M. Uyttendaele, “Joint bilateral upsa mpling”, Proc. ACM Siggraph, vol. 26, no. 3, pp. 96, 2007.
[8] D. De Silva, W. Fernando, H. Kodikaraarachchi, S. Worrall, A. Kondoz, “Adaptive sharpening of depth maps for 3D-TV,” Electronics Lett., vol. 46, no. 23, pp. 1546 –1548, 2010.
[9] D. Min, J. Lu, M. N. Do, “Depth video enhancement based on weighted mode filtering,” IEEE Trans. Image Processing, vol. 21, no. 3, pp. 117 6-1190, 2012.
[10] Y.S. Kang, S.B. LEE, YS Ho, "Depth map upsampling using depth loca l features," Electronics Lett., vol. 50, no. 3, pp. 170-171, 2014
[11] 3D-AVC Test Model 5, ITU-T SG16 WP3 ISO/IEC JTC1/SC29/WG11, Doc. JCT3V-C1003, Geneva, 2013.
[12] 3D-HEVC Test Model 3, ITU-T SG16 WP3 ISO/IEC JTC1/SC29/WG1 1, Doc. JCT3V-C1005, Geneva, 2013.
[13] Common test condition of 3DV core experiments, ITU-T SG16 WP3 IS O/IEC JTC1/SC29/WG11, Doc. JCT3V-C1100, Geneva, 2013.