Image Montage for Constructing Photorealistic Virtual World from Different Real Scene Images

Mizuki Tachibana
Iwate University
Graduate School of Engineering
Morioka, Japan
h24j092@eecs.iwate-u.ac.jp

Tadahiro Fujimoto
Iwate University
Graduate School of Engineering
Morioka, Japan
fujimoto@cis.iwate-u.ac.jp

***Abstract*—An image montage technique creates a photorealistic output image without artifacts by synthesizing multiple input images. We propose an image montage method to create a photorealistic virtual scene that seems as if it exists in a real world by synthesizing real scene images captured in different places. The images to synthesize are appropriately selected from a large database of different scene images. We suppose that images in the database were captured in the same place by moving a camera to translate its 3D position and rotate its viewing direction. Our method first obtains lots of similar image pairs such that two images in each pair have similar regions as if they are the same scene part that shifts its position in respective images by a camera motion. Then, from the pairs, appropriate successive images are selected and arranged in a 3D virtual space such that every image is similar to both adjacent images in their overlapping regions to fake a camera motion of a walk-through in the space, which results in constructing a photorealistic virtual world. The similarity in the overlapping region is evaluated using gist and color histogram. The selection of the successive images is done by solving an optimization problem using a graph of images to search for appropriate closed paths, each of which satisfies a restriction of a 3D space. A final virtual scene image is synthesized by blending the overlapping regions of adjacent images in a desired closed path.**

Keywords—image edit; image montage; virtual scene; gist; color histogram;

I. Introduction

In computer graphics and vision, image montage is one of active research topics and lots of methods have been proposed so far. The method proposed by Sivic et al. [1] creates a photorealistic virtual world by treating real scene images captured at different places in a large database as images captured by moving a camera at the same place. Images to use are sequentially selected and moved in left/right, forward, and horizontally rotational directions to arrange in a 3D virtual space so as to make adjacent images partially overlap each other. The adjacent images are selected by evaluating the similarity in their overlapping region using gist [2] and color layout vectors. This sequential selection of images results in making a chain of images in the 3D space. Then, a serious problem can be caused by inconsistently assigning different images to the same 3D position and direction by different parts of the chain. In this research, we propose a method to cope with this inconsistency. Our method uniquely assigns images to their own suitable 3D positions and directions as a solution of an optimization problem using a graph of images. The graph is searched for closed paths that satisfy 3D restrictions to give each image in the paths consistency for its 3D position and direction.

II. Related Work

Various image montage methods have been proposed. The interactive tool of [3] automatically determines the optimal boundaries of the desired regions from multiple input images to synthesize a natural output image. In the system of [4], an output image is synthesized from suitable input images automatically selected from a database by putting category labels on a canvas. Graphcut texture synthesis of [5] can stitch two input images along an automatically determined boundary. The image summarization method of [6] summarizes multiple input images into an output image by bidirectional similarity. Image melding of [7] synthesizes an output texture to change between two textures gradually. The methods of [8] and [9] create a patchwork in which important parts cut out from respective input images are arranged as compactly as possible.

Panorama synthesis from input images is an important topic in image montage. QuickTime VR of [10] and Panoramic image mosaics of [11] are pioneering works. The method of [12] reduces the distortion of a synthesized panorama image. The method of [13] makes seams between input images inconspicuous. The methods of [14] and [15] synthesize lots of input images along a street into a horizontally-long panorama image by a multi-perspective technique. The method of [16] generates a panorama image with depth. The methods of [17] and [18] synthesize a panorama image from a video.

The methods of [19] and [20] treat a large number of images captured in the same place, such as a tourist spot, from various camera positions by different persons. The images are arranged in a 3D space and related to each other by exactly estimating their 3D positions. The method of [21] allows a user to move his/her viewpoint smoothly between videos.

III. Similarity Evaluation

The *gist* of an image is a feature to evaluate the global structure of the image [2]. For example, it is often used for scene classification. It is obtained as a M-dimensional vector by applying multi-scale and multi-orientation Gabor filters to an image, where M is the number of the filters. The similarity of two gists is evaluated using their dissimilarity, d_g, defined by the sad (sum of absolute difference) of their vectors. The smaller d_g is, the more similar the gists are.

The *color histogram* of an image evaluates the color distribution of the image by voting for L bins defined by dividing the range of pixel colors. It is obtained as a L-dimensional vector. One way to evaluate the similarity of two histograms is histogram intersection [22], which gives a similarity value s_h, $0 \le s_h \le 1$. As s_h gets closer to 1, the histograms are more similar. We define the dissimilarity $d_h = 1 - s_h$.

Our method evaluates the similarity of two images in their overlapping region by a dissimilarity d_{gh} defined by averaging the two dissimilarities using a weight w_{gh}, $0 \le w_{gh} \le 1$.

$$d_{gh} = w_{gh}\, d_g + \left(1 - w_{gh}\right) d_h \tag{1}$$

IV. Proposed Method

A. Outline

Our database has real scene images of different places, and the images are treated as images captured by moving a camera in the same place to synthesize virtual scene images. First, our method obtains lots of similar image pairs such that two images in each pair have similar regions as if they are the same scene part that shifts its position in the respective images by a camera motion. In order to obtain such pairs, the similarity of two images in every pair is evaluated for their overlapping region formed by applying some camera motions: t_{max} translations and r_{max} rotations. Next, from the similar image pairs, appropriate successive images to arrange in a 3D virtual space are determined such that every image is similar to both adjacent images in their overlapping regions to fake a walk-through camera motion. In order to arrange unique images for respective 3D positions and directions without inconsistency, the successive images are determined as a solution of an optimization problem using an image montage graph constructed by using the similar image pairs as edges and their images as nodes. The graph is searched for closed paths that satisfy 3D restrictions to give each image in the paths consistency for its 3D position and direction. Finally, a virtual scene image is synthesized by blending the images in a desired closed path.

B. Virtual Camera Condition

Images in the database were actually captured in different places by different cameras using different parameters such as pixel resolutions, viewing angles, and focal lengths. In order to treat these images as images captured in the same place by moving a common camera, a *virtual camera condition* is given as follows. The 3D shapes and positions of objects in each image are not known. Thus, we suppose that all the objects are on a 3D plane that was positioned in parallel to the image plane of the common camera when the image was captured. The 3D plane is called *object plane*. Each image is mapped on its object plane. The local 3D camera coordinate system of each image is defined by a left-hand system; the position at which the

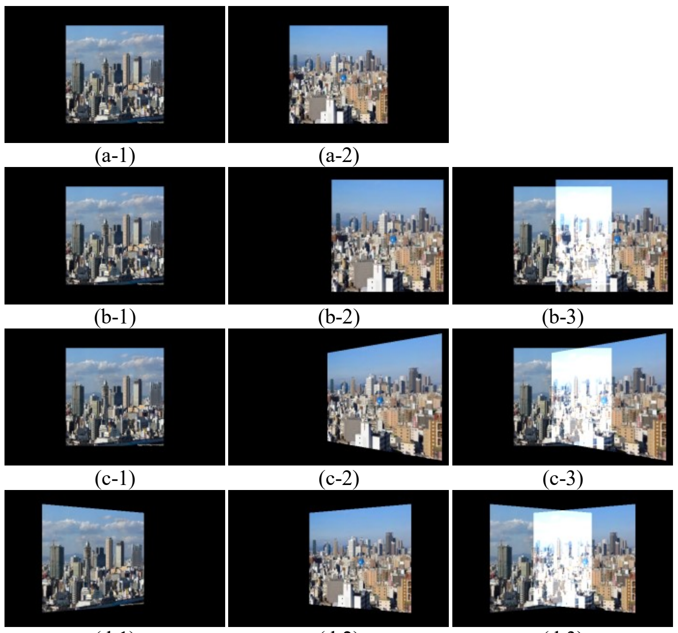

Fig. 1. Examples of overlapping regions.

camera was when the image was captured is its origin, the viewing direction of the camera is *z* axis, the right direction of the viewing direction is *x* axis, and the upper vertical direction is *y* axis. The distance from the origin to the object plane in the *z* direction is a parameter to determine. The scaling of the object plane in the *x* and *y* directions between the camera coordinates and the pixel coordinates of the image is also a parameter. For example, one simple parameterization is to use the number of pixels of the image as the length of the object plane in each of the *x* and *y* directions and give the average of the two lengths to the distance from the origin in the *z* direction.

C. Overlapping Region of Two Images

Two images are arbitrarily picked out from the database as an image pair and partially overlapped by a camera motion chosen from t_{max} translations T_t, $t = 1, \cdots, t_{max}$, and r_{max} rotations R_r, $r = 1, \cdots, r_{max}$. Then, the similarity of their overlapping region is evaluated. In Fig. 1, two images are mapped on their object planes respectively as shown in (a-1) and (a-2). In each case of (b) and (c), the object plane of (a-2) is moved by a 3D coordinate transformation from the camera coordinate system of (a-1) to that of (a-2) to simulate a camera motion as shown in (b/c-2), while the object plane of (a-1) remains the same as a reference plane as shown in (b/c-1). This results in forming their overlapping region as the bright part shown in (b/c-3), which is made by simply adding the pixel colors of (b/c-1) and (b/c-2). The virtual camera condition for these cases is given the simple parameterization described in section IV B. In the case of (b), the camera motion of (b-2) is a translation by a 3D vector $\boldsymbol{t} = (t_x, 0, t_z)$, which moves the object plane of (a-2) to the right in the *x* direction and near to the origin in the *z* direction. The overlapping region forms a rectangle in (b-3). In the case of (c), the camera motion of (c-2) is a rotation by an angle θ_y, which rotates the object plane of (a-2) around the *y* axis to the right. The overlapping region forms a hexagon in (c-3). This asymmetrical hexagon can generally cause a different result in similarity evaluation when the rotation is applied reversely by the same angle. In order to avoid this problem, our method rotates not only the object plane of (a-2) but also that of (a-1) in mutually reversed directions by $\theta_y/2$ to form a symmetrical hexagon as shown in the case of (d).

D. Selection of Similar Image Pairs

The similarity of each pair of two images I_i and I_j, $i \neq$ j, is evaluated for their overlapping region by applying t_{max} translations T_t , $t = 1, \cdots, t_{max}$, and r_{max} rotations R_r , $r = 1, \cdots, r_{max}$. From among all combinations of $(I_i,\ I_j,\ T_t,\ R_r)$, the e_{max} most similar pairs E_e, $e = 1, \cdots, e_{max}$, are selected as *similar image pairs*. Each similar image pair E_e has a camera motion, $T_e \in \{T_t, t = 1, \cdots, t_{max}\}$ or $R_e \in \{R_r, r = 1, \cdots, r_{max}\}$, to apply to its two images.

The similarity evaluation is done using the dissimilarity d_{gh} of Eq. (1) between two square patches S_i and S_j that come from the images I_i and I_j respectively. The patches have the same size, $w \times w$ pixels, and exactly overlap each other. The position of the patch pair is slided at intervals of some pixels within the overlapping region. The dissimilarity of the whole overlapping region is given the average of the dissimilarity values at all the positions. Each similar image pair E_e is given a dissimilarity d_e calculated for its overlapping region.

E. Search for Closed Paths of Similar Images

Using the e_{max} similar image pairs E_e, appropriate successive images to arrange in a 3D virtual space are determined such that every image is similar to both adjacent images in their overlapping regions to fake a walk-through camera motion. In order to arrange unique images for respective 3D positions and directions without inconsistency, the successive images are determined as a solution of an optimization problem as follows.

First, n_{max} *candidate images* I_n, $n = 1, \cdots, n_{max}$, are obtained by enumerating all the images from the e_{max} similar image pairs E_e. Then, an *image montage graph* is constructed using the candidate images I_n as nodes and similar image pairs E_e as edges. Each edge E_e has a camera motion, T_e or R_e, to apply to the two images whose nodes are connected by the edge. The edge E_e also has a dissimilarity d_e that indicates the similarity for the overlapping region of the two images. Each node can be connected to arbitrary number of other nodes. Two nodes can be connected by more than one edge having a different camera motion. Then, the graph is searched for *closed paths* such that each closed path satisfies the *3D restriction for positional and directional consistency*, by which the accumulation of the camera motions of all the edges along the closed path restores the 3D position and viewing direction of the camera to those at the first node after visiting all the nodes along the path. This gives each image in the closed paths consistency for its 3D position and direction.

The search for the closed paths in the graph is done by a recursive depth-first search algorithm as follows.

Step 1 : If all the nodes in the graph have been visited at least once, this algorithm is finished. Otherwise, one node is selected from among the nodes that have not been visited yet. The selected node becomes the current node N_c, and a new search starts from this node. In the depth-first search, the nodes that are visited in the depth direction are stored in a node list N_m , $m = 1, \cdots, m_{max}$, where m indicates the depth level. The number of stored nodes, m_{max}, is initialized by 0.

Step 2 : The current node N_c is checked whether it is stored in the current list $N_1, \cdots, N_{m_{max}}$.

Step 2-1 : If the current node N_c is not stored, this means that a closed path does not exist in the list. Then, m_{max} is increased by one, and the node N_c is stored to $N_{m_{max}}$. The search proceeds from this node $N_{m_{max}}$ to all the adjacent nodes except for the previous node $N_{m_{max}-1}$. The search to each adjacent node recursively goes to step 2, in which the adjacent node becomes the current node N_c.

Step 2-2 : If the current node N_c is stored, this means that a closed path exists in the list. When the m_s-th node N_{m_s} is the node N_c, the closed path consists of nodes $N_{m_s}, \cdots, N_{m_{max}}$. In this case, the search stops without proceeding to adjacent nodes. If the closed path satisfies the 3D restriction above, it is reserved as a candidate closed path. Then, this recursive search returns backward by one depth level.

Step 3 : If recursive search of step 2 is finished, go to step 1.

This algorithm repeats the above steps by changing a starting node. This copes with the case in which the whole graph consists of more than one isolated subgraph. One node can be visited more than once, which means that each node is allowed to be in more than one closed path. The recursive searches to the respective adjacent nodes in step 2-1 are executed individually. That is, the increment of m_{max} and the store of nodes to the list N_m for the adjacent nodes are done independently of each other after the search has branched to them.

After executing the above algorithm, desired closed paths are selected from candidate closed paths reserved in step 2-2. The average of the dissimilarities of all edges in a closed path is one criterion to determine whether the virtual scene synthesized by blending adjacent images of the path is photorealistic.

F. Image Synthesis of Virtual Scene

A final virtual scene image is synthesized by blending the overlapping regions of adjacent images of a desired closed path. One simple way of the blending is feathering [11] that uses a weight determined by the distances from the boundaries of two images. Poisson image editing [23] achieves better results.

V. Experimental Result

We used the following PC condition: OS: Windows10 Pro, CPU: Intel(R) Core(TM) i7-4790K 4.00GHz, RAM: 8.00GB. We developed and executed our software using Visual Studio 2015. The software to calculate gists was obtained from [24].

Fig. 2 and 3 are panorama images obtained by our method. The images in our database were obtained from [25]. The resolutions of all images were 256×256 pixels. In order to obtain similar image pairs, three rotations R_1, R_2, and R_3, which had

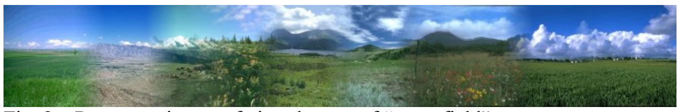
Fig. 2. Panorama image of virtual scene of “grass field”.

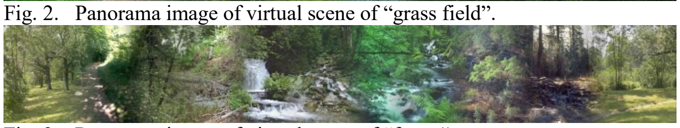
Fig. 3. Panorama image of virtual scene of “forest”.

rotation angles 20, 30, and 40, were applied to every image pair. For gists, Gabor filters of three scales were used, and the numbers of orientations in the scales were 8, 8, 4. The gists were calculated for R, G, and B respectively. As a result, one gist was given a 60-d vector. For color histograms, a and b values in Lab color system were used. Each color range was divided by 8. This results in 64 bins, and one histogram was given a 64-d vector. The weight w_{gh} of Eq. (1) was given 0.5. A patch of 40×40 pixels was slided at intervals of 30 pixels within an overlapping region. Each panorama image of Fig. 2 and 3 was obtained by the closed path with the smallest dissimilarity. In this experiment, the 3D restriction for candidate closed paths was that the accumulation of rotation angles along all the edges of a closed path was 360 degrees. The overlapping regions of adjacent images were blended by feathering. Other experimental data for Fig. 2 and 3 are shown in Table I.

In Fig. 2 and 3, some parts are given good results of natural montage. However, as a whole, photorealistic virtual scenes with high quality are not obtained. We consider that this was caused mainly by the shortage of database images. Besides, the small number of similar image pairs is also a serious reason.

VI. Conclusion

We proposed a method to create a virtual scene that seems as if it exists in a real world by synthesizing real scene images captured in different places. Virtual scenes synthesized by our method partially have photorealistic appearance. However, as a whole, high quality results have not been obtained yet. Although the increase of database images and similar image pairs is one solution, it causes the explosive increase of computation time for the current algorithm. The improvement of the algorithm is necessary as a future work.

References

[1] J. Sivic, B. Kaneva, A. Torralba, S. Avidan and W. T. Freeman, Creating and Exploring a Large Photorealistic Virtual Space, Proc. of IEEE Computer Vision and Pattern Recognition Workshops 2008, pp. 1-8, 2008.

[2] A. Oliva and A. Torralba, Building the Gist of a Scene: the Role of Global Image Features in Recognition, Progress in Brain Research, Vol. 155, pp. 23-36, 2006.

[3] A. Agarwala, M. Dontcheva, M. Agrawala, S. Drucker, A. Colburn, B. Curless, D. Salesin and M. Cohen, Interactive Digital Photomontage, Proc. of ACM SIGGRAPH 2004, pp. 294-302, 2004.

[4] M. Johnson, G. J. Brostow, J. Shotton, O. Arandjelovic, V. Kwatra and R. Cipolla, Semantic Photo Synthesis, Proc. of EUROGRAPHICS 2006, pp. 407-413, 2006.

[5] V. Kwatra, A. Schodl, I. Essa, G. Turk and A. Bobick, Graphcut Textures: Image and Video Synthesis Using Graph Cuts, Proc. of ACM SIGGRAPH 2003, pp. 277-286, 2003.

[6] D. Simakov, Y. Caspi, E. Shechtman and M. Irani, Summarizing Visual Data Using Bidirectional Similarity, Proc. of IEEE Computer Vision and Pattern Recognition 2008, pp. 1-8, 2008.

[7] S. Darabi, E. Shechtman, C. Barnes, D. B Goldman and P. Sen, Image Melding: Combining Inconsistent Images using Patch-based Synthesis, Proc. of ACM SIGGRAPH 2012, pp. 82(1)-(10), 2012.

[8] C. Rother, S. Kumar, V. Kolmogorov and A. Blake, Digital Tapestry, Proc. of IEEE Computer Vision and Pattern Recognition 2005, vol. 1, pp. 589-596, 2005.

[9] C. Rother, L. Bordeaux, Y. Hamadi and A. Blake, AutoCollage, Proc. of ACM SIGGRAPH 2006, pp. 847-852, 2006.

[10] S. E. Chen, QuickTime VR - An Image-Based Approach to Virtual Environment Navigation, Proc. of ACM SIGGRAPH 1995, pp. 29-38, 1995.

[11] R. Szeliski and H.-Y. Shum, Creating Full View Panoramic Image Mosaics and Environment Maps, Proc. of ACM SIGGRAPH 1997, pp. 251-258, 1997.

[12] J. Kopf, D. Lischinski, O. Deussen, D. Cohen-Or and M. Cohen, Locally Adapted Projections to Reduce Panorama Distortions, Computer Graphics Forum, vol. 28, no. 4, pp. 1083-1089, 2009.

[13] B. Summa, J. Tierny and V. Pascucci, Panorama Weaving: Fast and Flexible Seam Processing, Proc. of ACM SIGGRAPH 2012, pp. 83(1)-(11), 2012.

[14] A. Agarwala, M. Agrawala, M. Cohen, D. Salesin and R. Szeliski, Photographing Long Scenes with Multi-Viewpoint Panoramas, Proc. of ACM SIGGRAPH 2006, pp. 853-861, 2006.

[15] J. Kopf, B. Chen, R. Szeliski and M. Cohen, Street Slide: Browsing Street Level Imagery, Proc. of ACM SIGGRAPH 2010, pp. 96(1)-(8), 2010.

[16] G. Bahmutov, V. Popescu, M. Mudure and E. Sacks, Depth Enhanced Panoramas, Proc. of Conf. on Visualization 2004, 2004.

[17] A. Agarwala, K. C. Zheng, C. Pal, M. Agrawala and M. Cohen, Panoramic Video Textures, Proc. of ACM SIGGRAPH 2005, pp. 821-827, 2005.

[18] C. Hermans, C. Vanaken, T. Mertens, F. Van Reeth and P. Bekaert, Augmented Panoramic Video, Computer Graphics Forum, vol. 27, no. 2, pp. 281-290, 2008.

[19] N. Snavely, S. Seitz and R. Szeliski, Photo Tourism: Exploring Photo Collections in 3D, Proc. of ACM SIGGRAPH 2006, pp. 835-846, 2006.

[20] N. Snavely, R. Garg, S. Seitz and R. Szeliski, Finding Paths through the World’s Photos, Proc. of ACM SIGGRAPH 2008, pp. 15(1)-(11), 2008.

[21] L. Ballan, G. Brostow, J. Puwein and M. Pollefeys, Unstructured Video-Based Rendering: Interactive Exploration of Casually Captured Videos, Proc. of ACM SIGGRAPH 2010, pp. 87(1)-(11), 2010.

[22] M. J. Swain and D. H. Ballard, Color Indexing, International Journal of Computer Vision, vol. 7, no. 1, pp. 11-32, 1991.

[23] P. Perez, M. Gangnet and A. Blake, Poisson Image Editing, Proc. of ACM SIGGRAPH 2003, pp. 313-318, 2003.

[24] http://lear.inrialpes.fr/software

[25] http://people.csail.mit.edu/torralba/code/spatialenvelope/

TABLE I. Experimental Data of Fig. 2 and 3

Fig.	Size of data	Num. of sim. pairs	Num. of images	Rotation angles (degree)	Dissimilarity	Time for similar pairs (min.)	Time for closed paths (min.)
2	30	190	10	40, 40, 20, 40, 40, 40, 20, 40, 40, 40	1.237	5.943	2.133
3	40	220	11	30, 40, 20, 40, 30, 40, 20, 40, 40, 40, 20	1.502	9.371	25.010

“Size of data” : the number of images in the database
“Num. of sim. pairs” : the number of selected similar image pairs
“Num. of images” : the number of images in the closed path
“Rotation angles” : the rotation angles of edges along the closed path
“Dissimilarity” : the averaged dissimilarity of the closed path
“Time for similar pairs” : the computation time to select similar image pairs
“Time for closed paths” : the computation time to search for all closed paths