A Stitching Algorithm of Still Pictures with Camera Translation

Yang Linhong and Masahito Hirakawa
Graduate School of Engineering, Hiroshima University
1-4-1 Kagamiyama, Higashi-Hiroshima 739-8527, Japan
{yanglh, hirakawa}@isl.hiroshima-u.ac.jp

Abstract

We propose a method to stitch still images taken by a camera with translation and generate a panoramic image automatically. In such situation, perspective parallax may arise at overlapping regions of two neighboring picture images, and this makes the stitching difficult if we use the existing algorithm in panoramic systems. To solve this problem, we introduce a technique of pattern detection and point-based matching in this paper.

Pattern detection is carried out in two stages. The first one is line detection, and the second one is image segmentation. By this technique, the system extracts where the overlapping region is in each pair of neighboring images, and whether the perspective parallax exists. If this pattern detection can not extract similar segment from two neighboring images, point-based matching is applied to complete panoramic image generation. Several points with significant feature are sampled from one image, and their corresponding points are searched in the other image by point pattern matching.

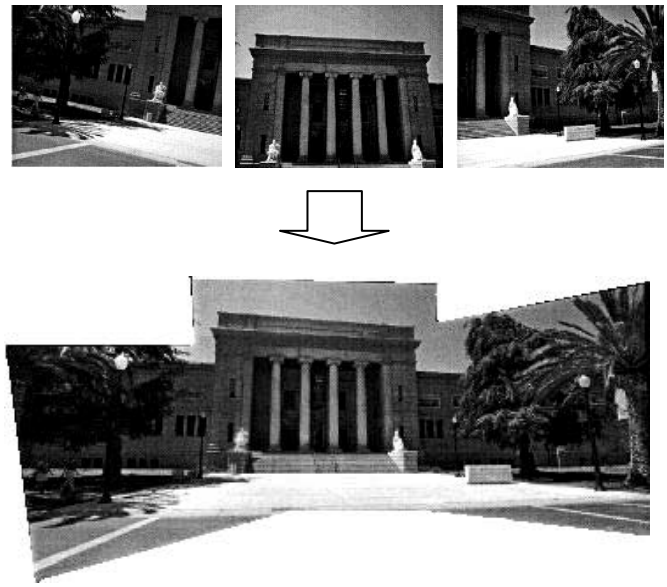

Figure 1 A mosaic image

1. Introduction

Mosaicing is a technique to join/merge a sequence of still pictures into one blended picture [1]. As digital cameras become popular, software products allowing a user to make a mosaic (panoramic) image are now on the market.

The basic process in generating a mosaic image can be described as follows [1]:

(1) find or ask for camera parameters
(2) estimate the motion model according to those parameters and warp images
(3) stitch images
(4) adjust and blend color intensities

Images are mapped into cylindrical or spherical coordinates in response to the camera's motion. Figure 1 shows an example of the resultant mosaic image which is generated from three original pictures. Here, most of the existing mosaicing software assume that images are taken without changing camera's position [2] and only panning motion is allowed as illustrated in Fig.2; that is, camera's optical axis moves just around an optical center of the camera.

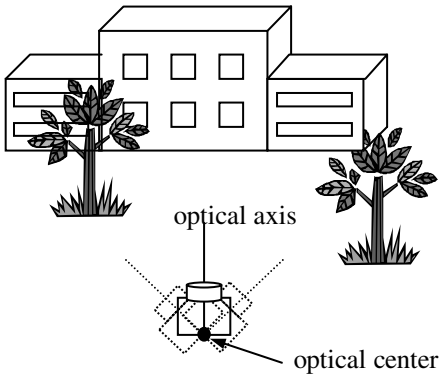

Figure 2 Camera movement

* Ms. Yang is now for SRA Nishinihon Inc., and Dr. Hirakawa is for Shimane University (hirakawa@cis.shimane-u.ac.jp).

Proceedings of the First International Symposium on Cyber Worlds (CW'02)
0-7695-1862-1/02 $17.00 © 2002 IEEE

However suppose, for example, when someone wants to take a street-wide flat view picture. In such case, the person or camera moves along the street as in Fig.3. This causes perspective parallax as in Fig.4 and, thus, existing technique is not adequate. For example, Fig 5 is an example of the output image generated by Photoshop. As you can see, perspective parallax makes an unnatural part in the resultant image. Furthermore, it should be noted that the image in Fig. 5 can not be generated automatically and the positioning of input images was carried out manually.

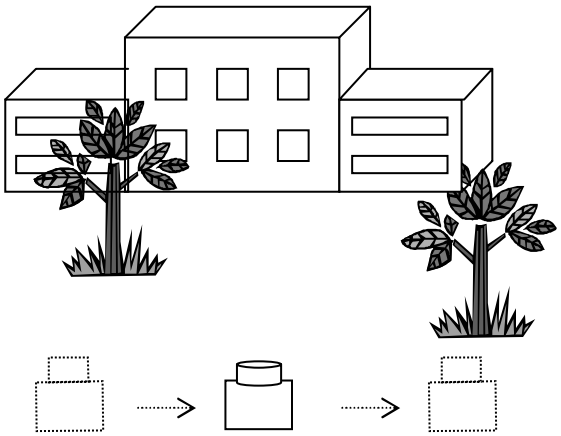

Figure 3 Camera movement in parallel to target object(s)

In this paper, we present a stitching algorithm for still images which may be taken with camera translation. To do this, we first need to detect the position of perspective parallax and find out the overlapping regions between two neighboring images. We solve this problem by performing pattern detection. Pattern detection is to find out one object pattern in the input images and estimate the perspective parallax. If pattern detection does not work properly, point-based matching is used. This is based on the techniques which are called base-point sampling and point pattern matching.

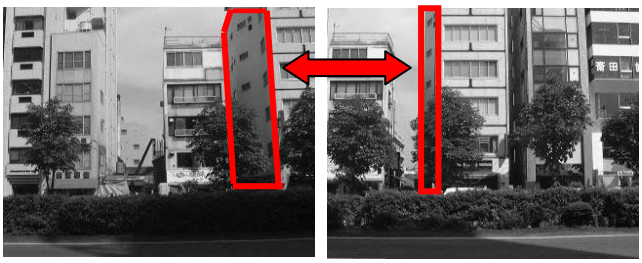

Figure 4 Two pictures captured with camera translation

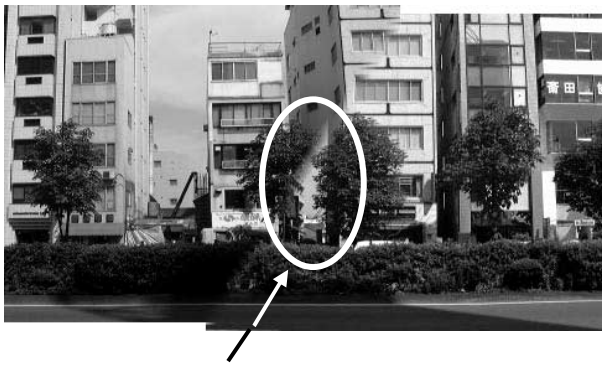

Figure 5 A panoramic image generated by the existing software

In the following, in Section 2, related work is described. Section 3 gives an overview of the proposed algorithm. Pattern detection is explained in Section 4. Section 5 presents point-based matching. We evaluated the performance of the algorithm and the result is presented in Section 6. Finally, a conclusion is given in Section 7.

2. Motivation and Related Work

Mosaicing is an established technique in many disciplines, including computer vision, image processing, computer graphics, and photogrammetry. It helps to generate wide-view (panoramic) images – provided that some overlapping exists among neighboring images [1], [2]. The image mosaicing software has now been accepted as a tool for the people who use digital cameras. Commercial products such as Photoshop provides a function of panoramic image generation.

In addition, there has been a growing interest in the use of mosaic images to represent the information contained in videos [3], [4]. The mosaicing technique is interesting and work effectively in, for example, video indexing, search, and manipulation [5].

Here, most of the existing mosaicing systems assume that images are taken by a camera which is placed at a fixed position. Panning, tilting, and zooming are the possible camera operations. However, this assumption is not convenient in some cases.

As a fact, a person specialized in sign design, who is a friend of one of the authors, is interested in the color distribution in the scenery of a town. In such application, the user may move along the street to take pictures, since the buildings, shops, trees, and others organizing the scenery spread out before him/her. In the case of taking pictures with translation, perspective parallax arises, resulting in the difficulty in stitching pictures. A new

technique is thus necessary to solve this problem.

3. Algorithm Overview

In the existing mosaicing systems, the most important task is to find out the overlapping region in two neighboring images. In the proposed algorithm, each pair of images is first examined whether they have significant overlap - for example, more than a quarter of the image size. Here we have an assumption: perspective parallax may arise in the region where horizontal edges can not be detected, that is, the region indicates the position of, for example, a building's side. The other region containing sets of horizontal edges would be a building's face and is considered a possible overlapping region.

Figure 6 shows a flow of the stitching algorithm proposed in the paper. Firstly, the pattern detection is carried out, which includes line detection and image segmentation to extract outlines of different regions in an image; i.e., to divide the image into regions which are made up of pixels which have something in common [6] - for example, attributes (brightness, color, etc.) are similar, indicating that they belong to the same facet of an object.

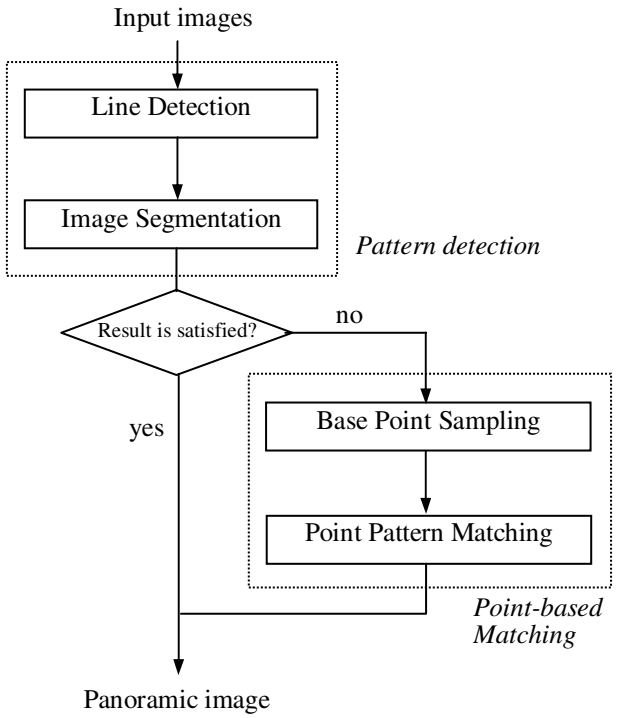

Figure 6 Stitching algorithm

If the result of the pattern detection is not satisfactory, the point-based matching is invoked. At this stage, we firstly choose a significant point as a base point for constructing a point-pattern which is presented for image matching. For this task, the sequential similarity detection algorithm (SSDA) [7], [8] is used, though just few points are selectively applied for comparison.

4. Pattern Detection

Perspective parallax exists at overlapping regions if pictures are captured with camera translation. So, in the algorithm, the most important task is to perceive the perspective parallax.

From Fig.4, we notice that, in the case of taking street-wide view pictures, perspective parallax always exists on the plane which is not parallel to camera translation, such as the building's side. On the other hand regions which could be overlapped without any serious consideration appear in the parallel plane, such as the building's face. This phenomenon calls a rule. The building's face often aligns numbers of windows, balconies, etc. They consist of horizontal lines and vertical lines if we apply image processing upon those pictures. On the contrary, horizontal lines become slant when they lie on the building's side where the perspective parallax arises.

4.1 Line detection

We first detect the edges in an image pair. The purpose of the edge detection is to extract information from an image in such a way that the output image contains much less information than the original one, but the little information it contains is much more convenient than the discarded information for comparison in generating a panoramic image.

We apply the Laplacian Mask (L-Mask) for edges detection.

$$\text{L-Mask} = \begin{pmatrix} l_{11} & l_{21} & l_{31} \\ l_{12} & l_{22} & l_{32} \\ l_{13} & l_{23} & l_{33} \end{pmatrix}$$

Let I be the original image, and i_{mn} the pixel at position (m, n).

$$I = \begin{pmatrix} i_{11} & i_{21} & \cdots & i_{m1} \\ i_{12} & i_{22} & \cdots & i_{m2} \\ \cdots & & & \\ i_{1n} & i_{2n} & \cdots & i_{mn} \end{pmatrix}$$

Edge detection will yield an output image I' whose

pixel is denoted by $i'_{m,n}$.

$$i'_{mn} = \begin{array}{l} i_{m-d,\,n-d} \times l_{11} + i_{m,\,n-d} \times l_{21} + i_{m+d,\,n-d} \times l_{31} + \\ i_{m-d,\,n} \times l_{12} + i_{m,\,n} \times l_{22} + i_{m+d,\,n} \times l_{32} + \\ i_{m-d,\,n+d} \times l_{13} + i_{m,\,n+d} \times l_{23} + i_{m+d,\,n+d} \times l_{33} \end{array}$$

where d is the distance between the point (m,n) and it's associated neighbors (for edge detection, d is set to 1). If $i'_{mn} > 0$, point (m, n) is considered on an edge.

Next, horizontal and vertical lines are extracted in the following way:

Let dx be a parameter to determine whether line segments are combined. For a certain point $i'_{m,n}$, if $i'_{m-dx,\,n} = i'_{m,n} = i'_{m+dx,n}$, the points set [(m-dx,n), (m-dx+1,n),…,(m-1,n), (m,n), (m+1,n) ...(m+dx,n)] will construct a horizontal line.

In the same way, given a parameter dy for vertical line generation, if $i'_{m,\,n-dy} = i'_{m,\,n} = i'_{m,\,n+dy}$, the points set [(m,n-dy), (m,n-dy+1), .., (m,n-1), (m,n), (m,n+1) … (m,n+dy)] will construct a vertical line.

Fig.7 shows a result after the line detection is applied to the input image. Here, in order to remove the noise from original images, we use the median filter before application of the line detection.

4.2 Image segmentation for similarity detection

Line detection generates an output image I' with horizontal lines and vertical lines. We thus specify the image I' in the form as given below:

$$I' = \{H_1, H_2, \ldots, H_i, \ldots H_j\} \cup \{V_1, V_2, \ldots V_i, \ldots V_k\}$$

Here H_i is a horizontal line and denoted by:

$$H_i = ((m_i, n_i), length_i)$$

where (m_i, n_i) is the start-point of H_i and $length_i$ is the length of H_i. The origin is supposed to be at the upper left corner.

Similarly, V_i is a vertical line and denoted by:

$$V_i = ((m_i, n_i), length_i)$$

where (m_i, n_i) is the start-point of V_i and $length_i$ is the length of V_i.

Image segmentation is then carried by checking the horizontal and vertical lines. In our approach, a segment is an area which covers a group of horizontal and vertical lines (see Fig. 8) and managed by:

$$Seg_i = ((m_i, n_i), width_i, height_i)$$

where (m_i, n_i) is the lower-left corner of the segment Seg_i, $width_i$ is the width of Seg_i, and $height_i$ is its height.

The value of $height_i$ is first determined by the following function.

$$height_i = E_v + \sigma_v$$

Original pictures

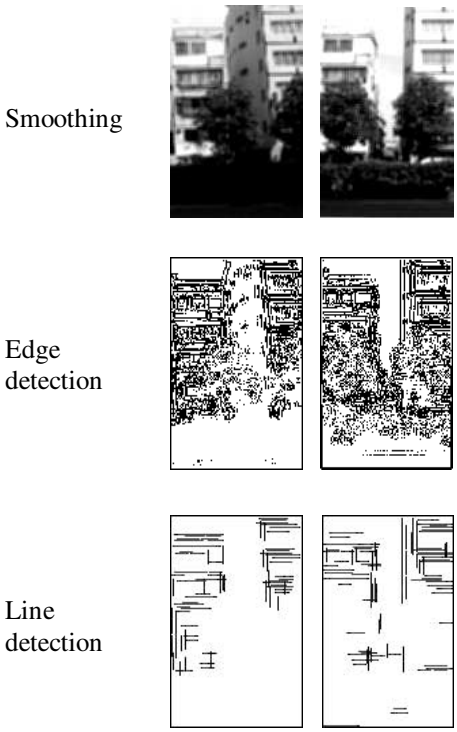

Figure 7 Line detection

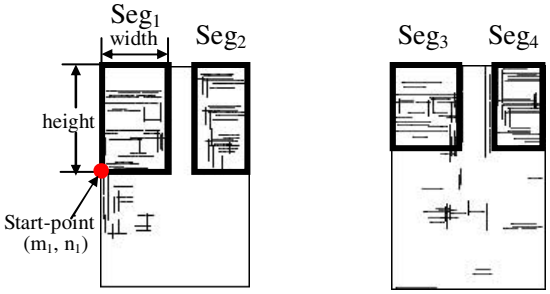

Figure 8 Image segmentation

where E_v is an average of y-coordinate values of the lower end points of all vertical lines and σ_v is a distribution of vertical lines position, as given below.

$$E_v = \sum_{i=1}^{k} (V_i.n_i + V_i.length_i) / k$$

$$\sigma_v^2 = \sum_{i=1}^{k} (V_i.n_i + V_i.length_i - E_v)^2 / k$$

$Vi.n_i$ and $Vi.length_i$ represent the y-coordinate value and the length of Vi, respectively. Here all segments take the same height since a purpose of the segmentation is just to identify the corresponding region in neighboring images. In addition, we consider the images which are taken on the street and, in such situation, the lower part of those images may contain trees, cars, and some other "noisy" objects. Such area should be removed in comparison.

The width of a segment is then determined so that there is no rift in the horizontal lines; i.e., for any x-coordinate point in the segment, there is at least one horizontal line which covers the point. As the result, we get the segments as shown in Fig. 8.

A segment Seg_i is characterized for comparison by three features: HL_i, E_i, and σ_i. HL_i is the y-coordinate value of the horizontal line located at the top. E_i is an average of y-coordinate values of the horizontal lines in Seg_i, and σ_i is a distribution of the horizontal lines.

A similarity SIM between two segments, Seg_i and Seg_j, is scored by the following function:

$$SIM = (|HL_i - HL_j| + |E_i - E_j| + |\sigma_i - \sigma_j|) / 3$$

5. Point –based Matching

In the process of pattern detection, if no similar segments can be detected, just like in Fig.9, the point-based matching is applied to find out the stitching position of two images.

5.1 Base-point sampling

In the point-based matching, we first decide the most significant point from one original image as a base point. This point is determined by applying the Laplacian mask, where d in the function in 4.1 is set to the pattern's width (in our trial, d = image's width/20). Then the difference of $|i_{mn} - i'_{mn}|$ is calculated, and the point which takes the maximum value is considered as the base point.

5.2 Point pattern matching

A point pattern, $Ptn_i^{m,n}$, is formed by the base-point (m,n) and several other points in the image (we set the number at 3 in our trial; that is, a point pattern is organized by four points). Those points are placed at the corner of the rectangle whose size is MxN, as illustrated in Fig.10. In our experiment, we assumed that M = (image's width) / 20 and N = (image's height) / 10.

Figure 9 Two original pictures and their detected vertical lines and horizontal lines

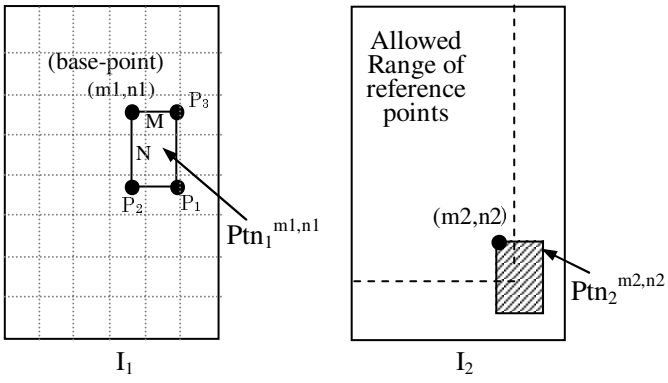

Figure 10 Point pattern matching

Four corner points of $Ptn_1^{m1,n1}$ in the image I_1 are compared with target points in the other image I_2 repetitively by sliding the position of matching in I_2. Suppose $Ptn_2^{m2,n2}$ keeps the minimum error with $Ptn_1^{m1,n1}$. (m_2,n_2) is then determined as the corresponding point to (m_1,n_1). Here the error scoring function is given by:

$$E = \sum_{k=0}^{3} |CV_1(P_k) - CV_2(P_k)|$$

where $CV_i(P_k)$ denotes the color value of the point P_k in image I_i.

Fig.11 shows a resultant image generated by point pattern matching

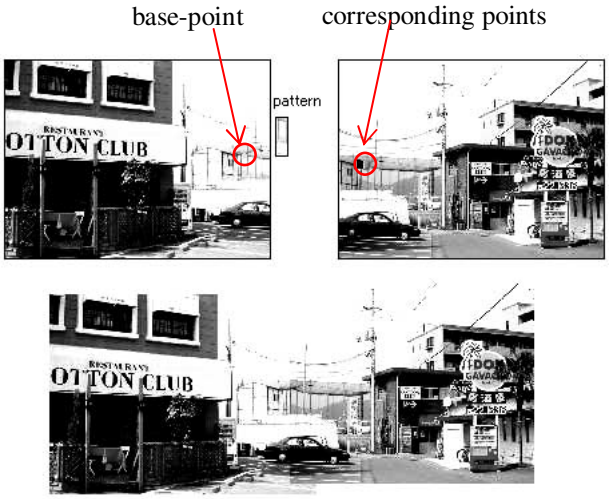

Figure 11 Result by point pattern matching

6. Experiment

We have tested the performance of the proposed method on a number of real image pairs where two neighboring images have about 30% to 50% overlap and focus length does not change so much. We used 65 sample image sets for experiment.

A result of the experiment is summarized in Table 1. One stitching result is given as an example in Fig. 12 which is generated by the pattern detection method.

Table 1 Result of experiment

Image pair	Success in pattern detection	Success in point-pattern matching
65 pairs	46 pairs (71%)	61 pairs (94%)

As given in the table, though the pattern detection method could find the appropriate stitching position for 46 image pairs (71%), application of the point-pattern matching increases the performance up to 94%.

The proposed algorithm is rather simple, but the experiment showed that it gives reasonable performance.

7. Conclusion and Future Works

In this paper, we have presented a technique for constructing a panoramic image from a sequence of images with camera translation. We proposed pattern detection to reduce the negative influence caused by the perspective parallax and estimate stitching position between two images. In addition, we introduced the

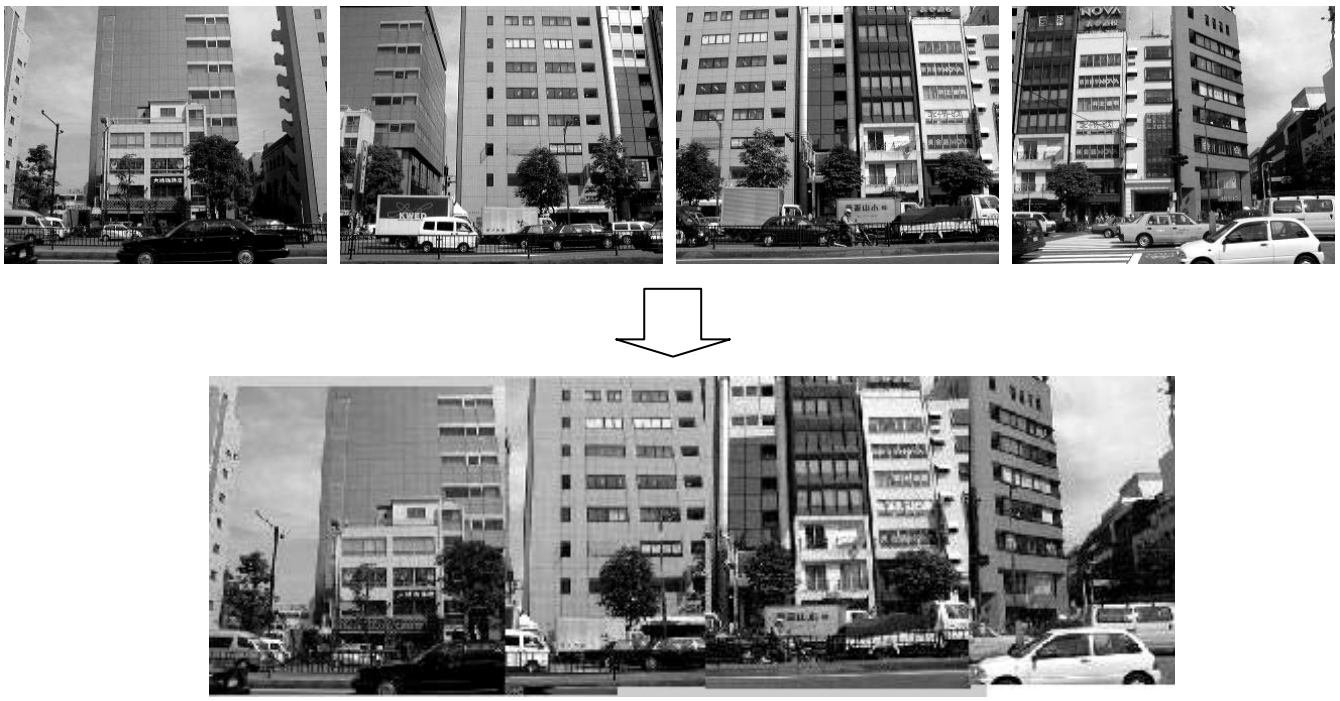

Figure 12 An example of the stitching result

point-based matching method to find out the corresponding points between two images, in the case when the pattern detection fails.

Future work includes the following: In pattern detection, if the similarities between each pair of segments are close, it is necessary to give a mechanism for identifying the right pair of segments. As to the point pattern matching, when perspective parallax exists and a point pattern is taken out from a parallax region, we fail to find the corresponding points. So we need an extension to optimize base-point sampling.

References

[1] F. Nielsen, "Randomized Adaptive Algorithms for Mosaicing Systems", IEICE Trans. of Inf. & Syst., Vol.E83-D, No.7, July 2000.

[2] H. Y. Shum and R. Szeliski, "Panoramic Image Mosaics", Technical Report, Microsoft Research, 1998

[3] M. Massey and W. Bender, "Salient Stills: Process and Practice", IBM Systems Journal, Vol.35, No.3&4, 1996.

[4] M. Irani, P. Anandan, J. Bergen, R. Kumar, and S. Hsu, "Mosaic Representations of Video Sequences and Their Applications", Signal Processing: Image Communication, Special Issue on Image and Video Semantics: Processing, Analysis, and Applications, Vol.8, No.4, 1996.

[5] J. Assfalg, A. Del Bimbo, and M. Hirakawa, "A Mosaic-based Query Language for Video Databases", Proc., IEEE Symposium on Visual Languages, 2000.

[6] M. Petrou and P. Bosdogianni, "Image Processing The Fundamental", John Wiley & Sons, 1999.

[7] J. Hasegawa, "Image Processing on Personal Computer", Technical Remark Press, 1986

[8] D. I. Barnea and H. F. Silverman, "A Class of Algorithms for Fast Digital Image Registration", IEEE Trans. on Computers, Vol.C-21, No.2, 1972.