Study on 3 DoF Image and Video Stitching Using Sensed Data

Minwoo Kim[1], Jonghoon Chun[2], Sang-Kyun Kim[3]
Department of Computer Engineering, Myongji University
116 Myongji-ro Cheoin-gu Yongin-si, S.Korea
[e-mail: [1]minwoo1989@gmail.com, [2]jonghoonchun@gmail.com, [3]goldmunt@gmail.com]
*Corresponding author: Sang-Kyun Kim

Received November 22, 2016; revised April 18, 2017; accepted May 25, 2017; published September 30, 2017

Abstract

This paper proposes a method to generate panoramic images by combining conventional feature extraction algorithms (e.g., SIFT, SURF, MPEG-7 CDVS) with sensed data from inertia sensors to enhance the stitching results. The challenge of image stitching increases when the images are taken from two different mobile phones with no posture calibration. Using inertia sensor data obtained by the mobile phone, images with different yaw, pitch, and roll angles are preprocessed and adjusted before performing stitching process. Performance of stitching (e.g., feature extraction time, inlier point numbers, stitching accuracy) between conventional feature extraction algorithms is reported along with the stitching performance with/without using the inertia sensor data. In addition, the stitching accuracy of video data was improved using the same sensed data, with discrete calculation of homograph matrix. The experimental results for stitching accuracies and speed using sensed data are presented in this paper.

Keywords: Image stitching, video stitching, panoramic video, ultra wide viewing video, sensor-based stitching

A preliminary version of this paper appeared in 2nd EEECS 2016, August 11-14, Qingdao, China. This version includes the detailed experimental results of stitching speed and accuracy using sensed data along with a method of video stitching. This work was supported by the ICT R&D program of MSIP/IITP [B0126-15-1013, Development of generation and consumption of Jigsaw-liked Ultra-Wide Viewing Spacial Media].

https://doi.org/10.3837/tiis.2017.09.019 **ISSN : 1976-7277**

1. Introduction

As large-size screens and combinations of multiple displays have become common recently, panoramic contents are applied more widely than before. For instance, panoramic contents are used in various areas of life including high-resolution point panoramic views of specific places such as streets, tourists' spots, accommodations, restaurants, and so forth [1] as well as many industrial sectors including advertisement and education.

The global sensor market is expected to grow from 79.5 billion dollars in 2014 to 116.1 billion dollars by 2019 at the rate of 7.9% per year. This growth rate is similar to that of the TV market [2]. As the Internet of Things advances lately, the importance of sensor data is more emphasized than before [3].

To overcome the disadvantage of existing processes of producing panoramic contents, which need to maintain posture calibrations between cameras, this study proposes a way of generating panoramic images at 3 different degrees of freedom (DoF) by means of sensor data. In addition to commonly used algorithms such as SIFT and SURF, the feature extraction algorithm of MPEG-7 CDVS (Compact Descriptor for Visual Search) that applies MPEG standard is adopted for image feature extraction and stitching. The feature extraction time, number of extracted features, and number of true points are compared among algorithms, and the stitching accuracy is determined to find out the optimal solution.

2. Related Work

D. G. Lowe proposed the SIFT (Scale-Invariant Feature Transform) algorithm that extracts from image features that are robust to such factors as scale, rotation, and noise [4]. However, the SIFT algorithm is disadvantageous in that its calculation rate is low. H. Bay, etal. proposed the SURF (Speeded Up Robust Features) algorithm that is robust to environmental changes in terms of scale, lighting, point of time, etc. [5][6] The SURF algorithm calculates at higher rates than the SIFT with a similar level of performance, but it has limitations in that this algorithm cannot utilize various useful features in a colorful space as it uses information only in a gray space. P. M. Panchal, etal. compared the number of features and extraction rates of existing SIFT / SURF algorithms. L. Juan and O. Gwun compared the number of features, extraction rates, scale and rotation, etc. of SIFT, PCA-SIFT, and SURF algorithms [7] [8]. MPEG too has developed its MPEG-7 CDVS standard for image feature extraction [9][10]. This enhances the image searching performance and speed drastically as it compares, not all, but certain parts of two images [11]. For image stitching, features appropriate for matching need to be selected among various extracted features. To choose features for matching, there needs to be a proper method to remove outliers. The RANSAC (RANdom SAmple Concensus) algorithm, suggested by M. A. Fischler and R. C. Bolles, selects sample data sets at random and then chooses specific ones of them that have formed the maximum concensus [12]. The homography estimation method, proposed by E. Dubrofsky, is used for 2D conversion in image stitching [13]. As the homography method utilizes the projective transformation of a plane from one image to another, it is an important parameter used for image stitching, 3D restoration, camera calibration, and so forth [14].

The interface standardization for sensors and actuators bridging between the virtual and real world has been performed in ISO/IEC 23005 (MPEG-V) [15]. The content authoring for sensorial effects was researched in [16][17][18][19], and the interaction between the real and virtual worlds using sensor data was presented in [20].

3. Image Preprocessing by Means of Sensor Data

3.1 Collection of Images and Sensor Data

For image preprocessing by means of sensor data, this study utilizes an Android smart-phone's acceleration sensor and magnetic field sensor. The acceleration sensor is used to measure the acceleration in the x/y/z axes of the smart-phone. **Fig. 1** shows the x, y, and z axes of the Android smart-phone.

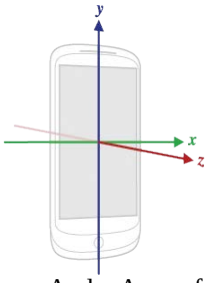

Fig. 1. The x, y, And z Axes of the Android Smartphone

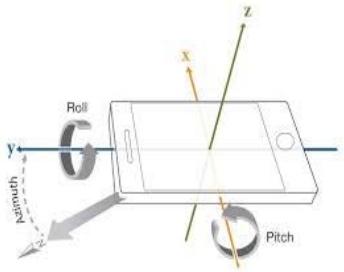

Fig. 2. Rotation Angle of the 3 Axes

One can obtain the rotation matrix by inputting the measured acceleration sensor data and magnetic sensor data into the getRotationMatrix() method of the Android SensorManager class. One can use the obtained rotation matrix and the getOrientation method to obtain the rotation angles corresponding to the three axes. The image can be rotated using the obtained rotation angle. Equation (4) in the subclause 3.2 is the rotation matrix generated by the average of the obtained rotation angles. The rotation angle (pitch) on the x-axis indicates the gradient of the smart-phone; the rotation angle (roll) on the y-axis indicates the left and right side rotation angle; and the rotation angle (azimuth) on the z-axis indicates the azimuth. **Fig. 2** shows the rotation angle on each axis.

3.2 Image Preprocessing in Reference to Sensor Data

For preprocessing of an original image in reference to the rotation angles on the x/y/z axes that are calculated by means of the application, the x/y/z axes of the smart-phone are matched with those of the image. **Fig. 3** shows the x/y/z axes of the smart-phone and image.

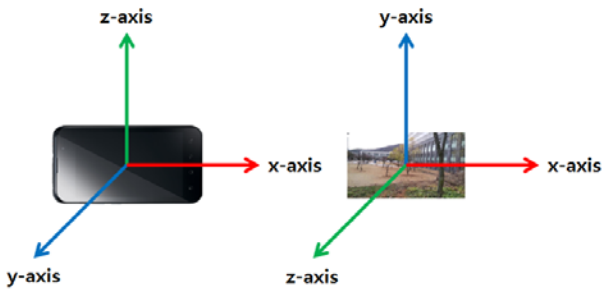

Fig. 3. 3-axis of the Smartphone and Image

In **Fig. 3**, the smart-phone and image have the same x-axis angle with the y and z-axes at the opposite side. Except the x-axis, the inverse values of the rotation angles on the y and z-axes are applied. The rotation

angles used for preprocessing are as follows:

$$\begin{aligned} \Delta pitch &= image2.pitch - image1.pitch \\ \Delta roll &= image2.roll - image1.roll \\ \Delta azimuth &= image2.azimuth - image1.azimuth \end{aligned} \tag{1}$$

Image stitching involves motions of entering or restricting images manually in order [21]. To enter images in this study, two images are classified manually with the left-side image (image 1) entered first and then the right-side image (image 2) entered later. To rotate each image, the average difference between the sensor data sets is calculated as follows:

$$\begin{aligned} avg_{pitch} &= \Delta pitch/NUMBER\_OF\_IMAGES \\ avg_{roll} &= \Delta roll/NUMBER\_OF\_IMAGES \\ avg_{azimuth} &= \Delta azimuth/NUMBER\_OF\_IMAGES \end{aligned} \tag{2}$$

For 3D rotation of images, a 3D rotation matrix is used. To this end, 2D images are converted into 3D images for the corresponding projection matrix [22].

$$P_{2d\to 3d} = \begin{bmatrix} 1 & 0 & -image.width/2 \\ 0 & 1 & -image.height/2 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3}$$

To rotate 3D images, the rotation matrix for each axis is calculated.

$$\begin{aligned} \mathrm{R}_x(avg_{pitch}) &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(avg_{pitch}) & -\sin(avg_{pitch}) & 0 \\ 0 & \sin(avg_{pitch}) & \cos(avg_{pitch}) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ \mathrm{R}_y(avg_{azimuth}) &= \begin{bmatrix} \cos(avg_{azimuth}) & 0 & \sin(avg_{azimuth}) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(avg_{azimuth}) & 0 & \cos(avg_{azimuth}) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ \mathrm{R}_z(avg_{roll}) &= \begin{bmatrix} \cos(avg_{roll}) & -\sin(avg_{roll}) & 0 & 0 \\ \sin(avg_{roll}) & \cos(avg_{roll}) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned} \tag{4}$$

The rotation matrixes are integrated by multiplying the 3 matrixes.

$$R_{xyz} = R_x(avg_{pitch}) * R_y(avg_{azimuth}) * R_z(avg_{roll}) \tag{5}$$

Finally, 3D images are converted back to 2D images for the corresponding projection matrix.

$$P_{3d\to 2d} = \begin{bmatrix} 1 & 0 & image.width/2 & 0 \\ 0 & 1 & image.height/2 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{6}$$

The 3D projection matrix is multiplied by the rotation matrix and 2D projection matrix to calculate the transformation matrix.

$$M_{trans} = P_{3d \to 2d} * (R_{xyz} * P_{2d \to 3d}) \tag{7}$$

The resulting transformation matrix is used for each image's warping. Warping is a way to distort images by relocating points on the original image. **Fig. 4** shows the sensor data and images before rotation. The differences between sensor data for each direction are described in **Fig. 4**. Using the equation (1), we can find how much the first image is distorted compared to the second image. If we rotate the first image in Fig. 4 with each amount of change, we can see that it is almost similar to the second image. If the amount of change is a positive number, it rotates in a clockwise direction, and if it is a negative number, it rotates in a counterclockwise direction.

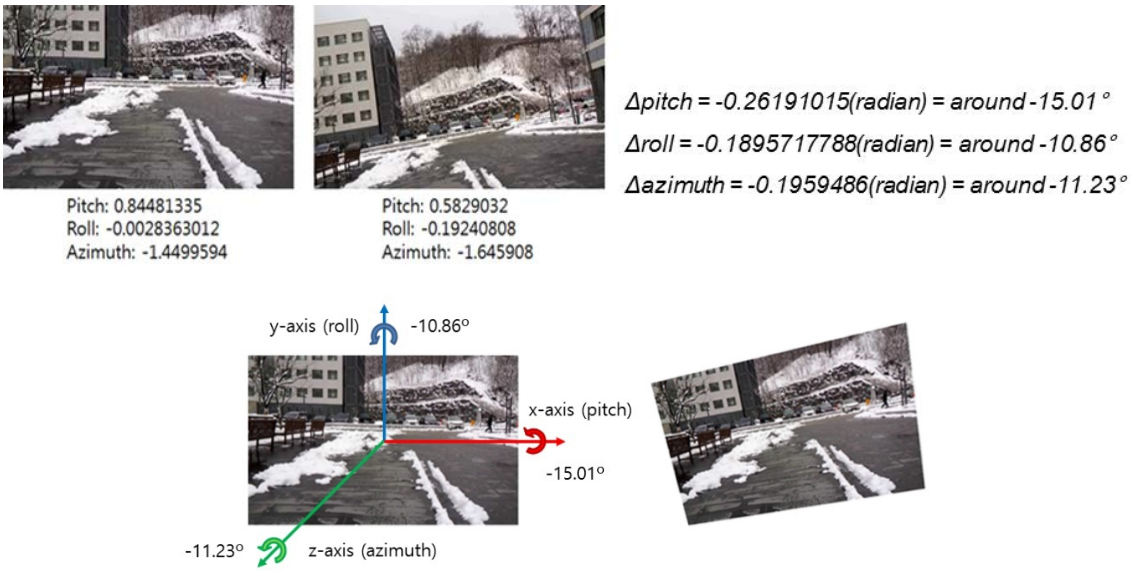

Fig. 4. Images and sensed data before 3-dimensional rotation

With each image rotated as much as about 7.505° on the x-axis, about 5.615° on the y-axis, and about 5.43° on the z-axis, the result is as in **Fig. 5,** where the two images aligned on the almost same horizontal line.

Fig. 5. Images after 3-dimensional rotation

4. Image Stitching

Fig. 6 shows a flow chart of image stitching steps to generate a panoramic image by stitching two similar images.

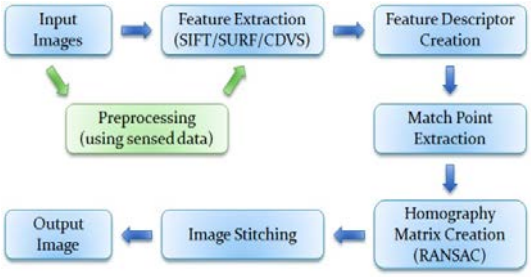

Fig. 6. Image stitching flow chart

The preprocessing step that utilizes sensor data proposed in this study is taken between the step of entering an image and the step of feature extraction.

4.1 Image Feature Extraction

For object recognizing, tracking, and matching in an image, one of the common ways is to extract visual features.

Fig. 7. Easily identified point *A* and not easily identified point *B*

When points in the right-side image that correspond to *A* and *B* in the left-side image in **Fig. 7**, *A* is easily identified while *B* is not. It is advantageous to find a spot that is easily recognizable, just like *A*, for matching. The reason why the point *A* is more advantageous than the point *B* is that it can be easily identified even if the shape, size, position, and viewpoint of the illumination camera in the image change. In this study, features are extracted by means of SIFT, SURF, and MPEG-7 CDVS algorithms to extract points that meet the same condition of *A*.

4.2 Image Stitching

For stitching of two images, image features need to be extracted by using a feature extraction algorithm. Extracted features contain outliers unnecessary for matching as shown in **Fig. 8**. Outliers are values that fall far out of the scope of data distribution. To remove these outliers, RANSAC algorithm suggested by M. A. Fischler and R. C. Bolles is used [12].

Fig. 8 . Extracted CDVS interest point(inlier: red, outlier: blue)

Fig. 9. Extracted CDVS interest point(removed outlier) and line prediction

Fig. 9 shows straight lines estimated based on true points of CDVS features that were extracted by means of RANSAC algorithm. The homography matrix is calculated based on true features of each image that were calculated by means of RANSAC algorithm, which is followed by image warping. Another image is added to the image warped by means of the homography matrix for image stitching. **Fig. 10** shows the result of image stitching. As indicated by red circles in the picture, stitching is not smooth. This results from inaccurate estimation of the homography. As the image went through the preprocessing of 3D rotation before feature extraction in application of the proposed method, the result of image stitching is presented in **Fig. 11**.

Fig. 10. Resualt of image stitching

Fig. 11. Resualt of image stitching after preprocessing

5. Video Stitching

A video is a set of sequential images. For video stitching, images need to be extracted from each frame of the video. After stitching of extracted images by means of the image stitching method stated in chapter 4 above, stitched images are combined as one video. **Fig. 12** shows the process of video stitching in application of the image stitching method.

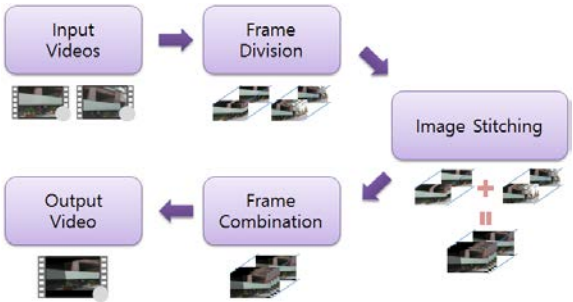

Fig. 12. Video stitching flow chart

Two videos may be stitched in the following way of video stitching:
(1) For stitching, two videos with overlapping parts are entered.
(2) Frame images are extracted from each video.
(3) All the extracted frame images are stitched by means of the image stitching method stated in chapter 4.
(4) Stitched frames are reconstituted as one video.
(5) After stitching, the produced video is saved.

Prerequisites for video stitching in this study are as follows:
(1) Videos that were taken from a similar distance and in a similar location are to be used.
(2) No scale change such as zoom-in/out is applied.
(3) Attention needs to be paid to preventing any blur from occurring due to a sudden change of the photographing angle.
(4) Sequential videos with no shot transition.

In general, a video's frame rate is 29.97fps (frame per second). To stitch a 100-second long video whose frame rate is 29.97fps, image stitching needs to be performed 2997 times, which takes a long time. The results of the video stitching time comparison are shown in Table 6, and the results of the stitching accuracy comparison in the video stitching using the proposed method are shown in Table 7. In order to shorten the video stitching time, this chapter proposes a way of using sensor data to generate a homography matrix and to improve the accuracy.

5.1 Discrete Generation of Homography Matrix

Fig. 13 shows the suggested way of dicretely generating homography matrices.

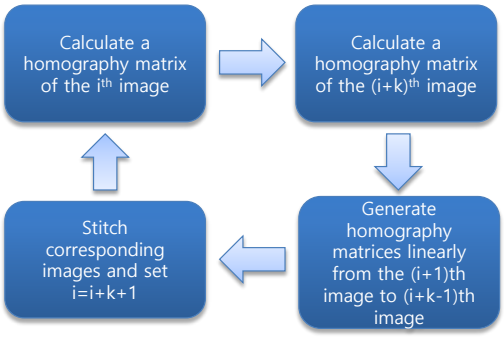

Fig. 13. A way of generating a homography matrix

To shorten the time of feature extraction and homography matrix estimation for image stitching, features of the first and N^{th} frame images are extracted, and then the homography matrix is calculated. N is a random number, and it is smaller than the number of the entire frame images.

$$H_1 = \begin{bmatrix} h_1^1 & h_2^1 & h_3^1 \\ h_4^1 & h_5^1 & h_6^1 \\ h_7^1 & h_8^1 & h_9^1 \end{bmatrix} \qquad H_N = \begin{bmatrix} h_1^N & h_2^N & h_3^N \\ h_4^N & h_5^N & h_6^N \\ h_7^N & h_8^N & h_9^N \end{bmatrix} \tag{8}$$

In homography matrix H_1 and H_N, elements h_1, h_2, h_3, …, h_9 are divided by N to calculate elements of H_2, H_3, …, H_{N-1}.

$$h^n = h^1 + \frac{h^N - h^1}{N-1} * (n-1) \tag{9}$$

h is the n^{th} element of the homography matrix. By using the expression above, homography matrix H_2 can be calculated as follows:

$$H_2 = \begin{bmatrix} h_1^1 + \frac{h_1^N - h_1^1}{N-1} * 1 & h_2^1 + \frac{h_2^N - h_2^1}{N-1} * 1 & h_3^1 + \frac{h_3^N - h_3^1}{N-1} * 1 \\ h_4^1 + \frac{h_4^N - h_4^1}{N-1} * 1 & h_5^1 + \frac{h_5^N - h_5^1}{N-1} * 1 & h_6^1 + \frac{h_6^N - h_6^1}{N-1} * 1 \\ h_7^1 + \frac{h_7^N - h_7^1}{N-1} * 1 & h_8^1 + \frac{h_8^N - h_8^1}{N-1} * 1 & 1 \end{bmatrix} \tag{10}$$

The above calculation continues up to the point of homography matrix H_{N-1}, and then image stitching is performed with the produced homography matrixes. Homography matrix element h_9 is not modified since it has been normalized. **Fig. 14** shows H_2, …, H_9 that were produced by means of H_1 and H_{10} after homography matrixes H_1 and H_{10} were calculated in the condition that *N=10*.

$$H_1 = \begin{bmatrix} 0.316439 & 0.180669 & 703.026 \\ -0.22353 & 0.900518 & -57.2087 \\ -0.00047269 & 0.000114387 & 1 \end{bmatrix} \quad H_2 = \begin{bmatrix} 0.316003 & 0.179586 & 702.058 \\ -0.223281 & 0.899005 & -55.5413 \\ -0.000472575 & 0.000113566 & 1 \end{bmatrix}$$

$$H_3 = \begin{bmatrix} 0.315568 & 0.178502 & 701.089 \\ -0.223032 & 0.897492 & -53.8739 \\ -0.00047246 & 0.000112746 & 1 \end{bmatrix} \quad H_4 = \begin{bmatrix} 0.315132 & 0.177419 & 700.121 \\ -0.222783 & 0.89598 & -52.2066 \\ -0.000472345 & 0.000111926 & 1 \end{bmatrix}$$

$$H_5 = \begin{bmatrix} 0.314697 & 0.176336 & 699.152 \\ -0.222534 & 0.894467 & -50.5392 \\ -0.00047223 & 0.000111106 & 1 \end{bmatrix} \quad H_6 = \begin{bmatrix} 0.314261 & 0.175252 & 698.184 \\ -0.222285 & 0.892955 & -48.8718 \\ -0.000472114 & 0.000110286 & 1 \end{bmatrix}$$

$$H_7 = \begin{bmatrix} 0.313826 & 0.174169 & 697.215 \\ -0.222035 & 0.891442 & -47.2044 \\ -0.000471999 & 0.000109466 & 1 \end{bmatrix} \quad H_8 = \begin{bmatrix} 0.31339 & 0.173085 & 696.246 \\ -0.221786 & 0.889929 & -45.5371 \\ -0.000471884 & 0.000108645 & 1 \end{bmatrix}$$

$$H_9 = \begin{bmatrix} 0.312955 & 0.172002 & 695.278 \\ -0.221537 & 0.888417 & -43.8697 \\ -0.000471769 & 0.000107825 & 1 \end{bmatrix} \quad H_{10} = \begin{bmatrix} 0.312519 & 0.170919 & 694.309 \\ -0.221288 & 0.886904 & -42.2023 \\ -0.000471654 & 0.000107005 & 1 \end{bmatrix}$$

Fig. 14. Homography matrixes $\boldsymbol{H_1}$, $\boldsymbol{H_{10}}$ in the condition that *N=10* and the produced homography matrixes $\boldsymbol{H_2}$, …, $\boldsymbol{H_9}$

Fig. 15 shows the result of image stitching by means of homography matrixes H_2, …, H_9 that were generated in application of calculated homography matrixes H_1, H_{10} and the suggested method. The stitching result is improper since homography matrixes that were generated at random were used.

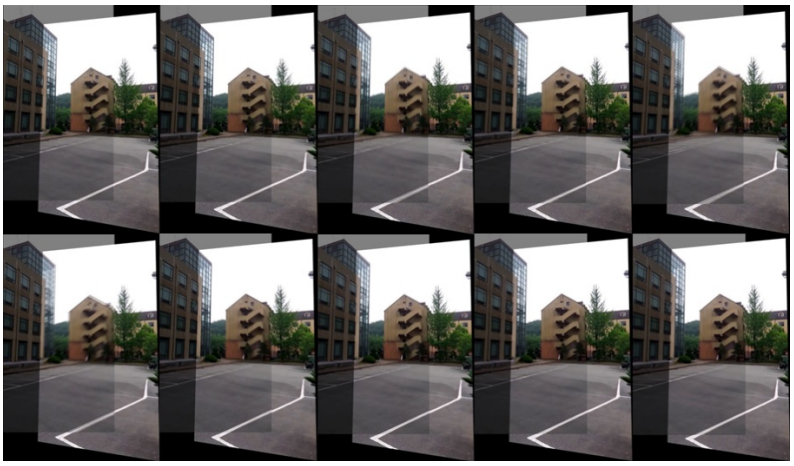

Fig. 15. The result of image stitching by means of homography matrixes $\boldsymbol{H_1}$, $\boldsymbol{H_{10}}$ and homography matrixes $\boldsymbol{H_2}$, …, $\boldsymbol{H_9}$ that were generated

5.2 Use of Sensor Data for Stitching Accuracy

The accuracy is likely to decrease when such homography matrixes in 5.1 above are used. To enhance the stitching accuracy, this section suggests a way of utilizing sensor data. **Fig. 16** shows a flow chart where the way to improve stitching accuracy by means of sensor data is added to **Fig. 13**.

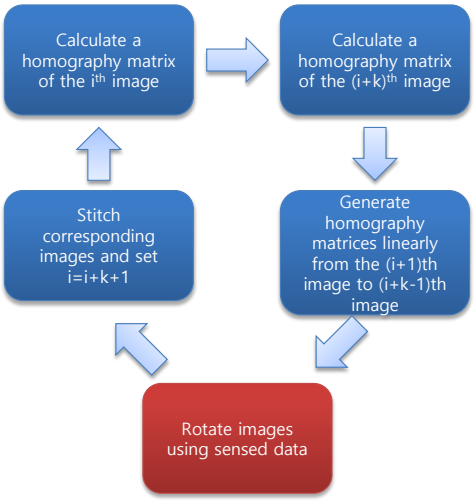

Fig. 16. A flow chart of the accuracy improvement of stitching using sensor data

Fig. 17. Stitching (left) by means of discretely generated homography matrixes; stitching (right) after image preprocessing with sensor data

In reference to rotation angles on the x/y/z axes calculated by means of the image, video, and sensor data collecting application in chapter 3.1 above, the preprocessing prior to frame image stitching improves the accuracy. **Fig. 17** shows the result of stitching by means of randomly generated homography matrixes and the result of stitching after image preprocessing with sensor data. As for the image that used randomly generated homography matrixes, stitching of the apartment in front is inaccurate. In contrast, the accuracy is outstandingly improved when sensor data was used for image rotation although it was is perfect.

6. Experimental Result

6.1 Experiment Environment

For image stitching with sensor data utilized, the experiment environment was created as shown in **Table 1**:

Table 1. Experimental environment of image stitching using sensed data

Experimental environment of image stitching	
CPU	Intel Core i7-4770 3.4GHz
RAM	8GB
OS	Windows 7 Enterprise K 64-bit
IDE	Visual Studio Professional 2013
Library	OpenCV 2.4.11
	MPEG-7 CDVS evaluation framework

6.2 Collection of Sensor Data

Experiment data was produced according to the following rules:
Images were taken by using the left-side smart-phone in the following condition: horizontal angle: 0°; vertical angle: 0°; left-right rotation: 0°.
Images were then taken by using the right-side smart-phone in the following condition: horizontal angle: 0~12°; vertical angle: 0~12°; left-right rotation: 0~12° with 4-degree adjustment at a time.

6.3 Comparison of Image Feature Extraction Algorithm Performance

As the number of features reaches a certain level, the data-processing time increases. As the feature extraction speed is fast, the time for image stitching reduces accordingly. In this study, 10,000 images of 640×360 are extracted by using each algorithm.

Table 2. Interest point extraction algorithms comparison of SIFT, SURF, CDVS

Class	SIFT	SURF	CDVS
Avg. extraction time(s)	1.726	0.347	1.111
Avg. number of interest points(n)	1776.11	1553.95	300
Avg. number of inliers(n)	155.66	251.68	39.2
The ratio of inliers per extracted interest points(%)	8.76	16.2	13.06

Table 2 compares the extraction results of SIFT, SURF, and CDVS feature extraction algorithms. SIFT algorithm extracted the largest number of features among the three algorithms, and its ratio of true points per feature was the lowest down to 8.76%. SURF algorithm extracted a number of features within a short time. Its ratio of true points per feature was 16.2%, which is the highest among the three. CDVS algorithm extracted only 300 features as restricted by the standard. Its ratio of true points per feature was 13.06%, which is the medium level among the three.

6.4 Comparison of Image Stitching Performance

One of the important elements when it comes to comparison of image stitching performance is how long the stitching process takes. **Table 3** shows the time that the entire image stitching process took when SIFT, SURF, and CDVS algorithms were used respectively. As it did for

feature extraction, SIFT algorithm took the longest time for stitching too up to 4.53 seconds. SURF algorithm took the shortest time for stitching, 1.116 seconds shorter than that of SIFT algorithm.

Table 3. Image stitching time comparison of SIFT, SURF, CDVS

Class	SIFT	SURF	CDVS
Avg. stitching time(s)	4.530	1.116	2.477

Image stitching performance may be evaluated, not only based on stitching time, but also based on accuracy although there is no appropriate way of stitching accuracy evaluation. In this study, the stitching result on the screen is analyzed as it is. To assess the stitching accuracy of each algorithm, when two images are perfectly matched with each other, the level is “high”; when there are afterimage effects due to some discordance, the level is “middle”; if images are discordant, the level is “low”; if an abnormal homography matrix is produced or the stitching fails, the level is “failure.” **Table 4** shows the table of image stitching accuracy assessment. According to the experiment results, for fast featre extraction and accurate stitching, SURF algorithm is recommended.

Table 4. Assessment of image stitching accuracy (High: 3; Middle: 2; Low: 1; Failure: 0) – When SIFT, SURF, and CDVS images are all in the “high” level, the data is removed.

Index	Description	SIFT	SURF	CDVS
1	Reference image	-	-	-
3	Horizontally 8° tilted image	3	3	2
7	Horizontally 8°, vertically 4° tilted image	3	2	2
8	Horizontally 12°, vertically 4° tilted image	3	2	2
9	Vertically 8° tilted image	3	3	2
10	Horizontally 4°, vertically 8° tilted image	2	3	1
11	Horizontally 8°, vertically 8° tilted image	2	3	2
13	Vertically 12° tilted image	3	3	1
14	Horizontally 4°, vertically 12° tilted image	1	3	1
15	Horizontally 8°, vertically 12° tilted image	3	1	0
16	Horizontally 12°, vertically 12° tilted image	0	1	1
18	Horizontally 4°, rightwards 4° tilted image	3	3	1
19	Horizontally 8°, rightwards 4° tilted image	3	3	1
20	Horizontally 12°, rightwards 4° tilted image	1	2	1
22	Horizontally 4°, vertically 4°, rightwards 4° tilted image	3	3	2
23	Horizontally 8°, vertically 4°, rightwards 4° tilted image	3	2	3
24	Horizontally 12°, vertically 4°, rightwards 4° tilted image	3	2	1
27	Horizontally 8°, vertically 8°, rightwards 4° tilted image	2	3	1
28	Horizontally 12°, vertically 8°, rightwards 4° tilted image	3	3	1
29	Vertically 12°, rightwards 4° tilted image	2	3	1
30	Horizontally 4°, vertically 12°, rightwards 4° tilted image	2	2	1
31	Horizontally 8°, vertically 12°, rightwards 4° tilted image	2	2	2
32	Horizontally 12°, vertically 12°, rightwards 4° tilted image	0	3	3

33	Rightwards 8° tilted image	0	3	3
34	Horizontally 4°, rightwards 8° tilted image	3	2	3
35	Horizontally 8°, rightwards 8° tilted image	3	3	1
36	Horizontally 12°, rightwards 8° tilted image	2	3	1
38	Horizontally 4°, vertically 4°, rightwards 8° tilted image	3	3	2
39	Horizontally 8°, vertically 4°, rightwards 8° tilted image	3	3	2
40	Horizontally 12°, vertically 4°, rightwards 8° tilted image	3	3	2
41	Vertically 8°, rightwards 8° tilted image	0	3	2
42	Horizontally 4°, vertically 8°, rightwards 8° tilted image	2	3	1
43	Horizontally 8°, vertically 8°, rightwards 8° tilted image	2	2	3
44	Horizontally 12°, vertically 8°, rightwards 8° tilted image	2	3	3
45	Vertically 12°, rightwards 8° tilted image	2	3	1
46	Horizontally 4°, vertically 12°, rightwards 8° tilted image	1	3	1
47	Horizontally 8°, vertically 12°, rightwards 8° tilted image	2	3	2
48	Horizontally 12°, vertically 12°, rightwards 8° tilted image	1	1	1
49	Rightwards 12° tilted image	3	3	2
50	Horizontally 4°, rightwards 12° tilted image	3	2	3
52	Horizontally 12°, rightwards 12° tilted image	3	2	3
55	Horizontally 8°, vertically 4°, rightwards 12° tilted image	2	3	1
57	Vertically 8°, rightwards 12° tilted image	3	3	2
58	Horizontally 4°, vertically 8°, rightwards 12° tilted image	2	3	2
59	Horizontally 8°, vertically 8°, rightwards 12° tilted image	3	1	3
60	Horizontally 12°, vertically 8°, rightwards 12° tilted image	1	3	1
61	Vertically 12°, rightwards 12° tilted image	2	3	1
62	Horizontally 4°, vertically 12°, rightwards 12° tilted image	2	3	1
63	Horizontally 8°, vertically 12°, rightwards 12° tilted image	0	1	3
64	Horizontally 12°, vertically 12°, rightwards 12° tilted image	2	2	1
Average		2.33	2.65	1.98

In this experiment, a preprocessing step with sensor data is included for image stitching robust to 3 different degrees of freedom. Stitching results after preprocessing are presented in **Table 5**. The preprocessing subject is the experiment data that was evaluated as “middle” in the stitching evaluation as shown in **Table 4**. According to the experiment results, the preprocessing step with sensor data can enhance the stitching accuracy.

Table 5. Stitching results after preprocessing

Index	Description	SIFT	SURF	CDVS
10	Horizontally 4°, vertically 8° tilted image	2->3	3	1->3
13	Vertically 12° tilted image	3	3	1->1
14	Horizontally 4°, vertically 12° tilted image	1->1	3	1->1
15	Horizontally 8°, vertically 12° tilted image	3	1->2	0->1
16	Horizontally 12°, vertically 12° tilted image	0->3	1->0	1->0

18	Horizontally 4°, rightwards 4° tilted image	3	3	1->2
19	Horizontally 8°, rightwards 4° tilted image	3	3	1->2
20	Horizontally 12°, rightwards 4° tilted image	1->3	2->2	1->3
24	Horizontally 12°, vertically 4°, rightwards 4° tilted image	3	2->3	1->2
27	Horizontally 8°, vertically 8°, rightwards 4° tilted image	2->2	3	1->2
28	Horizontally 12°, vertically 8°, rightwards 4° tilted image	3	3	1->3
29	Vertically 12°, rightwards 4° tilted image	2	3	1->2
30	Horizontally 4°, vertically 12°, rightwards 4° tilted image	2->2	3	1->2
31	Horizontally 8°, vertically 12°, rightwards 4° tilted image	2->1	2->3	2->3
32	Horizontally 12°, vertically 12°, rightwards 4° tilted image	0->2	3	3
33	Rightwards 8° tilted image	0->3	3	3
35	Horizontally 8°, rightwards 8° tilted image	3	3	1->3
36	Horizontally 12°, rightwards 8° tilted image	2->3	3	1->2
41	Vertically 8°, rightwards 8° tilted image	0->3	3	2->2
42	Horizontally 4°, vertically 8°, rightwards 8° tilted image	2->3	3	1->3
45	Vertically 12°, rightwards 8° tilted image	2->2	3	1->2
46	Horizontally 4°, vertically 12°, rightwards 8° tilted image	1->2	3	1->2
48	Horizontally 12°, vertically 12°, rightwards 8° tilted image	1->3	1->3	1->2
55	Horizontally 8°, vertically 4°, rightwards 12° tilted image	2->3	3	1->2
59	Horizontally 8°, vertically 8°, rightwards 12° tilted image	3	1->3	3
60	Horizontally 12°, vertically 8°, rightwards 12° tilted image	1->3	3	1->2
61	Vertically 12°, rightwards 12° tilted image	2->2	3	1->3
62	Horizontally 4°, vertically 12°, rightwards 12° tilted image	2->2	3	1->2
63	Horizontally 8°, vertically 12°, rightwards 12° tilted image	0->2	1->3	3
64	Horizontally 12°, vertically 12°, rightwards 12° tilted image	2->3	2->3	1->2
Average		1.77 -> 2.57	2.53 -> 2.83	1.30 -> 2.20

6.5 Comparison of Video Stitching Performance

The stitching time is an important element in comparison of video stitching performance as well. The video stitching experiment in this study was conducted using SURF algorithm, which was the fastest and most accurate among three tested algorithms as shown in 6.4. **Table 6** shows the time of stitching all the frame images by means of 10,000 pairs of frame image data, the time of stiching with discretely generated homography matrices, and the time of stitching with the generated homography matrix calibrated in reference to sensor data. Since the image I/O speed is a major factor in video stitching, the image I/O time was included in the experiment unlike the calculation of algorithm performance. In general, video frame rates are about 30fps. When stitching was performed for every 5 frames, there was little advantage in terms of speed. If a homography matrix was randomly generated for 15 frames, the accuracy was far inferior. Hence, a homography matrix was generated linearly for the middle 8 image pairs out of 10 pairs during the experiment.

Table 6. The time that the stitching step took when all the frame images were used, when a homography matrix was generated, and when sensor data was used for calibration

Class	Every frame	Discrete generation of homography matrix	Sensor data calibration w/ discrete homography matrix
Stitching(s) of 10,000 pairs	25951.83	10463.64	11043.28
Average time (s) of stitching each pair	2.60	1.05	1.10

As for stitching of every frame image, it took about 2.6 seconds, which is the performance rate of SURF algorithm. As for stitching by means of the homography matrix generation method stated in chapter 5.1, the average stitching time for a pair of frame images was about 1.05 seconds, which is about 40.5% faster than the way of stitching every frame. It was expected that stitching for every 10 images would accelerate the process as much as 90%, but the result fell short of the expectation probably due to the linear calculation of a homography matrix and the operation delay for homography matrix storage in the C++ STL vector. As for stitching with frame images calibrated in reference to sensor data, it took about 1.1 seconds including about 0.5 second for image rotation.

Accuracy is another important factor when it comes to video stitching performance. **Fig. 18** shows frame images when all frame images were used, when the homography matrix was generated, and when images were calibrated in reference to sensor data.

Fig. 18. Frame images when every frame image was used, when homography matrices was generated, and when sensor data was used for calibration

As shown in **Fig. 18**, the result of stitching all frame images is quite inaccurate because SURF feature extraction algorithm was not executed properly. The image generated with a homography matrix is inaccurate too, but after calibration with sensor data, the result was improved to a large degree.

Table 7 compares the accuracy when all the 100 frame images extracted from three pairs of video data were stitched, when the homography matrix was discretely generated for stitching, and when the image was calibrated by means of sensor data. The accuracy score when all the frame images were stitched was 2.36 on average. That for stitching with the discretely generated homography matrix was decreased to 2.03 on average, which is lower than that of

every frame image stitching. The accuracy score when the image was calibrated by means of sensor data was improved to 2.16 on average. Although this is lower than that of every frame image stitching, the accuracy was improved in comparison with when the discretely generated homography matrix was used for stitching.

Table 7. Video stitching evaluation (E1: every frame image stitching, E2: discrete generation of the homography matrix and then stitching, E3: sensor data based preprocessing before stitching)

Index	Test video set 1				Test video set 2				Test video set 3			
	E1	E2	E3	Comp.	E1	E2	E3	Comp.	E1	E2	E3	Comp.
1	3	3	3	Equal	1	1	1	Equal	2	2	2	Equal
2	3	3	2	Worse	2	1	1	Equal	2	2	3	Better
3	3	3	3	Equal	1	1	1	Equal	2	2	2	Equal
4	3	3	3	Equal	1	1	1	Equal	2	2	2	Equal
5	3	2	2	Equal	1	1	1	Equal	2	2	3	Better
6	3	2	2	Equal	1	1	1	Equal	2	2	2	Equal
7	3	3	3	Equal	1	1	1	Equal	2	2	2	Equal
8	3	3	3	Equal	2	1	1	Equal	2	2	2	Equal
9	3	3	3	Equal	2	1	1	Equal	2	2	2	Equal
10	2	2	2	Equal	2	2	2	Equal	2	2	2	Equal
11	3	3	3	Equal	2	2	2	Equal	2	2	2	Equal
12	3	3	2	Worse	1	2	2	Equal	2	2	3	Better
13	3	2	2	Equal	1	2	2	Equal	2	2	3	Better
14	3	2	2	Equal	2	2	2	Equal	2	2	3	Better
15	3	2	1	Worse	2	2	2	Equal	2	2	3	Better
16	3	2	2	Equal	2	2	2	Equal	2	2	3	Better
17	3	2	1	Worse	2	2	2	Equal	2	2	2	Equal
18	3	2	1	Worse	2	2	2	Equal	2	2	3	Better
19	3	3	2	Worse	2	2	2	Equal	2	2	3	Better
20	3	3	3	Equal	2	2	2	Equal	2	2	2	Equal
21	3	3	3	Equal	1	1	1	Equal	2	2	2	Equal
22	3	3	2	Worse	1	1	1	Equal	2	2	2	Equal
23	3	2	3	Better	1	1	1	Equal	2	2	2	Equal
24	3	2	2	Equal	1	1	1	Equal	2	2	2	Equal
25	3	2	2	Equal	3	1	1	Equal	2	2	2	Equal
26	2	3	2	Worse	1	1	1	Equal	2	2	3	Better
27	3	3	2	Worse	2	1	1	Equal	2	2	3	Better
28	3	3	3	Equal	2	1	1	Equal	2	2	3	Better
29	3	3	3	Equal	2	1	1	Equal	2	2	3	Better
30	3	3	3	Equal	1	1	1	Equal	2	2	2	Equal
31	3	3	3	Equal	2	2	2	Equal	2	2	2	Equal
32	3	3	3	Equal	2	2	2	Equal	2	2	3	Better
33	3	2	2	Equal	2	2	2	Equal	2	2	2	Equal
34	3	1	1	Equal	3	2	2	Equal	2	2	3	Better

35	3	1	1	Equal	3	2	2	Equal	2	2	2	Equal
36	3	2	2	Equal	2	2	2	Equal	2	2	2	Equal
37	3	2	2	Equal	2	2	2	Equal	2	2	2	Equal
38	3	3	3	Equal	1	2	2	Equal	2	2	3	Better
39	3	3	3	Equal	3	2	2	Equal	2	2	2	Equal
40	3	3	3	Equal	2	2	2	Equal	2	2	2	Equal
41	3	3	3	Equal	2	2	2	Equal	2	2	2	Equal
42	3	2	3	Better	3	2	2	Equal	2	2	2	Equal
43	3	2	3	Better	2	2	2	Equal	2	2	3	Better
44	3	1	2	Better	2	2	2	Equal	2	2	3	Better
45	3	1	1	Equal	2	2	2	Equal	2	2	2	Equal
46	3	1	1	Equal	2	1	2	Better	2	2	3	Better
47	3	1	1	Equal	1	1	1	Equal	2	2	3	Better
48	3	1	1	Equal	2	1	1	Equal	2	2	2	Equal
49	3	2	2	Equal	2	1	1	Equal	2	2	2	Equal
50	3	3	3	Equal	1	1	1	Equal	2	2	2	Equal
51	3	3	3	Equal	1	1	1	Equal	2	2	2	Equal
52	3	2	2	Equal	2	1	1	Equal	2	2	3	Better
53	3	2	2	Equal	2	1	1	Equal	2	2	3	Better
54	3	2	2	Equal	3	1	1	Equal	2	2	3	Better
55	3	2	1	Worse	2	1	1	Equal	2	2	3	Better
56	3	1	1	Equal	3	1	1	Equal	2	2	2	Equal
57	3	1	1	Equal	1	1	1	Equal	2	2	2	Equal
58	3	2	1	Worse	2	1	1	Equal	2	2	2	Equal
59	3	2	1	Worse	3	1	1	Equal	2	2	2	Equal
60	3	3	3	Equal	2	2	1	Worse	2	2	2	Equal
61	3	3	3	Equal	2	2	2	Equal	2	2	2	Equal
62	3	3	2	Worse	2	3	3	Equal	2	2	2	Equal
63	3	2	2	Equal	3	2	2	Equal	2	2	3	Better
64	3	2	2	Equal	3	2	2	Equal	2	2	2	Equal
65	3	2	2	Equal	1	2	2	Equal	2	2	2	Equal
66	3	2	1	Worse	2	2	3	Better	2	2	2	Equal
67	3	2	2	Equal	2	2	3	Better	2	2	2	Equal
68	3	3	3	Equal	2	2	3	Better	2	2	2	Equal
69	3	3	3	Equal	2	2	3	Better	2	2	2	Equal
70	3	3	3	Equal	3	3	3	Equal	2	2	2	Equal
71	3	3	3	Equal	2	2	2	Equal	2	2	2	Equal
72	3	3	3	Equal	3	2	2	Equal	2	2	2	Equal
73	3	2	2	Equal	3	2	2	Equal	2	2	2	Equal
74	3	1	2	Better	1	2	2	Equal	2	2	2	Equal
75	3	2	3	Better	3	2	2	Equal	2	2	2	Equal
76	3	2	3	Better	3	2	2	Equal	2	2	2	Equal

77	3	3	3	Equal	2	2	3	Better	2	2	2	Equal
78	3	3	3	Equal	2	3	3	Equal	2	2	2	Equal
79	3	3	3	Equal	2	3	3	Equal	2	2	2	Equal
80	3	3	3	Equal	3	3	3	Equal	2	2	2	Equal
81	3	3	3	Equal	3	3	3	Equal	2	2	2	Equal
82	3	2	3	Better	3	2	3	Better	2	2	2	Equal
83	3	2	3	Better	3	2	3	Better	2	2	2	Equal
84	3	2	2	Equal	3	2	2	Equal	2	2	2	Equal
85	3	2	2	Equal	3	2	2	Equal	2	2	2	Equal
86	3	2	3	Better	3	2	2	Equal	2	2	2	Equal
87	3	1	2	Better	3	3	3	Equal	2	2	2	Equal
88	3	2	3	Better	2	3	3	Equal	2	2	3	Better
89	3	3	3	Equal	3	3	3	Equal	2	2	2	Equal
90	3	3	3	Equal	3	3	3	Equal	2	2	2	Equal
91	3	3	3	Equal	3	3	3	Equal	2	2	2	Equal
92	3	2	3	Better	3	2	2	Equal	2	2	3	Better
93	3	2	2	Equal	3	2	2	Equal	2	2	3	Better
94	3	2	2	Equal	2	2	2	Equal	2	2	2	Equal
95	3	2	2	Equal	3	2	2	Equal	2	2	2	Equal
96	3	2	2	Equal	2	2	2	Equal	2	2	2	Equal
97	3	2	3	Better	3	2	2	Equal	2	2	2	Equal
98	3	2	3	Better	3	2	3	Better	2	2	2	Equal
99	3	2	3	Better	3	2	3	Better	2	2	2	Equal
100	3	3	3	Equal	2	2	2	Equal	2	2	2	Equal
Avg.	2.98	2.32	2.34	-	2.11	1.78	1.87	-	2.00	2.00	2.28	-
Equal	70				89				72			
Better	16				10				28			
Worse	14				1				0			

Five pairs of video data were added to the experiment result shown in **Table 7**, and 100 frame images were extracted by using eight pairs of video data in total. The result of stitching with the discretely generated homography matrix was then compared with the result of calibration in reference to sensor data as shown in **Table 8**. The improvement rate of stitching with sensor data was 26.13% on average, and the deterioration rate was 11.63% on average. The rate of no change despite calibration was 62.25%. Thus, the percentage of stitching accuracy improvement is higher than that of deterioration.

Table 8. Result of calibration in reference to sensor data

Class	Equal (%)	Better (%)	Worse (%)
Video set 1	70	16	14
Video set 2	89	10	1
Video set 3	72	28	0
Video set 4	58	15	27
Video set 5	28	72	0

Video set 6	56	22	22
Video set 7	63	18	19
Video set 8	62	28	10
Average	62.25	26.13	11.63

7. Conclusion

Important elements in the performance evaluation of algorithms for image feature extraction include the time of extraction and stitching, ratio of true points per extracted feature, and matching accuracy. Under constraint described in chapter 5, SURF algorithm shows the shortest extraction and stitching time, and the highest ratio of true points per extracted features. The level of matching accuracy of SURF was the highest among three compared algorithms.

This study intends to improve stitching accuracy through 3D rotation of each image in reference to sensor data prior to extracting image features in the existing way of image stitching. To verify stitching accuracy improvement, the preprocessing step was applied to data whose stitching accuracy was "middle" or lower. Although the reliability was not high when the accuracy was judged visually, accuracy improvement was significant compared to the time that the suggested method took, and this way of image stitching was thus proved effective.

The homography matrix was discretely calculated to shorten the time that video stitching would take. To improve the accuracy, image preprocessing was performed prior to video frame stitching. When the proposed method was applied, stitching was faster than that of the existing one. In addition, the use of sensor data along with the discrete generation of homography matrices enhanced the stitching accuracy further. The stitching results are similar to that of stitching every frame while the stitching speed decreased about 40%.

References

[1] Panorama VR, from http://www.uok3d.com/bussiness/panorama.php
[2] K. M. Park, W. H. Seok, and K. H. Lee, "Sensor Industry and Major Promising Sensor Market and Technology Trend," *Electronics and Telecommunications Research Institute*, February, 2015. Article (CrossRef Link)
[3] S. H. Lee, "State and the Issues of IOT," *Institute for Information & communications Technology Promotion*, April, 2014. Article (CrossRef Link)
[4] D. G. Lowe, "Distinctive Image Features from Scale-invariant Keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91-110, May, 2004. Article (CrossRef Link)
[5] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool, "Speeded-Up Robust Features (SURF)," *Similarity Matching in Computer Vision and Multimedia*, vol. 110, no. 3, pp. 346-359, June, 2008. Article (CrossRef Link)
[6] H. Bay, T. Tuytelaars, and L. V. Gool, "SURF: Speeded Up Robust Features," in *Proc. of 9th European Conference on Computer Vision*, 404-417, May, 2006. Article (CrossRef Link)
[7] P. M. Panchal, S. R. Panchal, and S. K. Shah, "A Comparison of SIFT and SURF," *International Journal of Innovative Research in Computer and Communication Engineering*, vol. 1, no. 2, pp. 323-327, April, 2013. Article (CrossRef Link)
[8] L. Juan and O. Gwun, "A Comparison of SIFT, PCA-SIFT and SURF," *International Journal of Image Processing (IJIP)*, vol. 3, no. 4, pp. 143-152, 2009. Article (CrossRef Link)
[9] CDVS1, "Call for Proposals for Compact Descriptors for Visual Search," *N12201*, Turin, Italy, ISO/IEC JTC1/SC29/WG11, 2011. Article (CrossRef Link)

[10] L. Y. Duan, F. Gao, J. Chen, J. Lin, and T. Huang, "Compact Descriptors for Mobile Visual Search and MPEG CDVS Standardization," in *Proc. of IEEE International Symposium on Circuits and Systems*, pp. 885-888, May, 2013. Article (CrossRef Link)
[11] H. K. Kim, "MPEG-7 CDVS Standardization Media Searching under Mobile Environment," *Telecommunications Technology Association*, vol. 144, pp. 35-38, 2012. Article (CrossRef Link)
[12] M. A. Fischler and R. C. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381-395, June, 1981. Article (CrossRef Link)
[13] E. Dubrofsky, "Homography Estimation," *University of British Columbia*, 2009. Article (CrossRef Link)
[14] Y. H. Park and O. S. Kwon, "Multiple Homographies Estimation using a Guided Sequential RANSAC," *The Korea Contents Association*, vol. 10, no. 7, pp. 10-22, July, 2010. Article (CrossRef Link)
[15] Kyoungro Yoon, Sang-Kyun Kim, Jae Joon Han, Seungju Han, Marius Preda, "MPEG-V: Bridging the virtual and real world," *Elsevier*, ISBN: 978-0-12-420140-8.
[16] Yong-Soo Joo, Sang-Kyun Kim, "Sensory Effect Authoring Tool for Sensible Media," *Journal of Broadcast Engineering*, Vol. 16, No. 5, pp. 693-893, Sept. 2011 (in Korean). Article (CrossRef Link)
[17] Sang-Kyun Kim, "Authoring Multisensorial Content," *Signal Processing: Image Communication*, vol. 28, Issue 2, pp. 162-167, Feb. 2013. Article (CrossRef Link)
[18] Sang-Kyun Kim, Yong-Soo Joo, YongMi Lee, "Sensible Media Simulation in an Automobile Application and Human Responses to Sensory Effects," *ETRI Journal*, Vol. 35, No. 6, pp. 1001-1010, Dec. 2013. Article (CrossRef Link)
[19] Sang-Kyun Kim, Seung-Jun Yang, Chung Hyun AHN, Yong Soo Joo, "Sensorial Information Extraction and Mapping to Generate Temperature Sensory Effects," *ETRI Journal*, Vol. 36, No. 2, pp. 224-231, Apr. 2014. Article (CrossRef Link)
[20] Sang-Kyun Kim, Jae Joon Han, Seungju Han, Yong Soo Joo, "Virtual world control system using sensed information and adaptation engine," *Signal Processing: Image Communication*, vol. 28, Issue 2, pp. 87-96, Feb. 2013. Article (CrossRef Link)
[21] M. Brown and D. G. Lowe, "Automatic Panoramic Image Stitching using Invariant Features," *International Journal of Computer Vision*, vol. 74, no. 1, pp. 59-73, December, 2007. Article (CrossRef Link)
[22] J. Heikkilä and O. Silvén, "A Four-step Camera Calibration Procedure with Implicit Image Correction," in *Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1106-1112, Jun. 1997. Article (CrossRef Link)

Minwoo Kim

- 2016 : Computer Eng. Dept. BS(2014), MS
- 2016.11 ~ : CTO in MintPot Co.
- ORCID: http://orcid.org/0000-0002-9405-6069
- Major interest: image processing, pattern recognition, 4D media, sensors and actuators, VR, and Internet of Things

Jonghoon Chun

- 1986: Computer Science in U of Denver), BS
- 1992: Computer Engineering in Northwestern University, MS(1988), PhD
- 1992.9 ~ 1995.6: Assistant prof. in U of Central Oklahoma
- 1995.9 ~ 2016.3: Professor of Computer Eng. Dept. in Myongji University
- 2016.3 ~ : Professor of Convergence Software College in Myongji University
- 2011.4 ~ : CEO of Prompt Technology Co.
- Major interest: Dababase, Big Data, Smart Signage, Data Mining and Recommender Systems, Information Retrieval, eBusiness Solutions, Medical Informatics, and Supplier Relationship Management

Sang-Kyun Kim

- 1997 : Computer Science in U of Iowa, BS(1991), MS((1995), PhD
- 1997.3~ 2007.2 : Principal researcher of Multimedia Lab in Samsung Advanced Institute of Technology
- 2007. 3 ~ 2017.2 : Professor of Computer Eng. Dept. in Myongji University
- 2017. 3 ~ : Professor of Convergence Software College in Myongji University
- Chairperson and project editor of MPEG International Standard Organization (MPEG-7, MPEG-A, MPEG-V, MPEG IoMT)
- ORCID : http://orcid.org/0000-0002-2359-8709
- Major Interest : digital content (image, video, and music) analysis and management, fast image search and indexing, color adaptation, 4D media, sensors and actuators, VR, Internet of Things, and multimedia standardization