Constructing interactive multi-view videos based on image-based rendering

Shih-Ming Chang*, Joseph C. Tsai and Shwu-Huey Yen

Department of Computer Science and Information Engineering,
Tamkang University,
New Taipei City, 25137, Taiwan
Email: rest306@hotmail.com
Email: kkiceman@gmail.com
Email: 105390@mail.tku.edu.tw
*Corresponding author

Timothy K. Shih

Department of Computer Science and Information Engineering,
National Central University,
Taoyuan County, 30201, Taiwan
Email: timothykshih@gmail.com

Abstract: In this paper, we use image-based rendering (IBR) to develop a scene rotation mechanism. We shot several images in the same scene and computed the angles between images. A video is then composed, allowing users to select viewing angles when the video is playing. We made three kinds of assumptions that may affect the resulting video, and proved our assumptions by a series of experiments. Finally, we use video of realistic scenario and produce interactive video by the proposed method. The contribution also includes techniques to compute geometric parameters of the scene from one or more images.

Keywords: SIFT algorithm; mean-shift algorithm; homography matrix; image-based rendering; IBR.

Reference to this paper should be made as follows: Chang, S-M., Tsai, J.C., Yen, S-H. and Shih, T.K. (2015) 'Constructing interactive multi-view videos based on image-based rendering', *Int. J. Computational Science and Engineering*, Vol. 10, No. 4, pp.402–414.

Biographical notes: Shih-Ming Chang is currently with the Department of Computer Science and Information Engineering, Tamkang University, Taiwan. He is a PhD student in the Department of Computer Science and Information Engineering at Tamkang University, Taiwan. He received his BS and MS from the St. John's University, Taiwan and Tamkang University, Taiwan in 2007 and 2009, respectively. His research interests include computer vision, image and video processing, and their applications.

Joseph C. Tsai is currently with the Department of Computer Science and Information Engineering, Tamkang University, Taiwan. He is a PhD student in the Department of Computer Science and Information Engineering at Tamkang University, Taiwan. He received his BS and MS from the same university in 2006 and 2008, respectively. His research interests include computer vision, image and video processing, and their applications.

Shwu-Huey Yen received her BS in Applied Mathematics from Fu-Jen Catholic University, Taipei, Taiwan in 1980 and MS and PhD in Mathematics from Northeastern University, Boston, USA in 1982 and 1986, respectively. She is currently an Associate Professor at the Department of Computer Science and Information Engineering of Tamkang University, Taipei, Taiwan. Her research interest is in multimedia processing including security, feature matching and information retrieval.

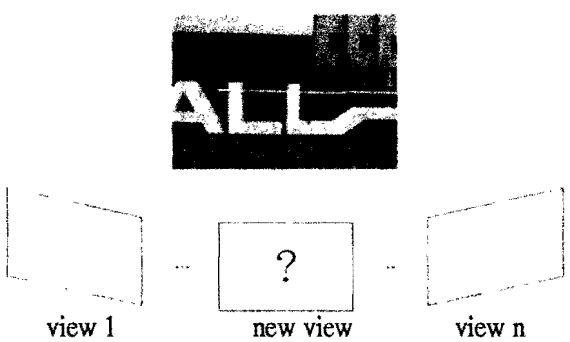
Timothy K. Shih is a Professor of the Department of Computer Science and Information Engineering, National Central University, Taiwan. He was a Department Chair of the CSIE Department at Tamkang University, Taiwan. He is a fellow of the Institution of Engineering and Technology (IET). In addition, he is a senior member of ACM and a senior member of IEEE. He also joined the Educational Activities Board of the Computer Society.

1 Introduction

3D movies, TV and stereo games are emerging and advertent technology. The famous 3D movie, Avatar, brings a different visual effect to audiences. Recently, 3D televisions have 3D stereo effects that can be viewed with 3D glasses. And, the game machine N3DS also presents 3D stereo effects. The technology called 'auto-stereoscopy' is becoming a trend in entertainments. We can say that the 3D's era is coming. People are not just satisfied with 2D videos. 3D stereo videos bring the visual impact; and this kind of viewing effect lets spectators feel the scene in a super realistic way.

To realise 3D video, traditional 2D videos can be used to reconstruct 3D videos. But, it is a complex process of modelling and calculation, which has drawbacks such as the slow speed of execution and wasting a lot of time on modelling. Another drawback is that 3D scenes were created with 3D models. These 3D scenes may lead to unnatural or unrealistic effects owing to the use of 2D texture in 3D. Therefore, directly using scenes in videos, the visual effects are more realistic and the performance can be reasonable in general (i.e., the model of 3D scenes and recovery of surface texture can be omitted).

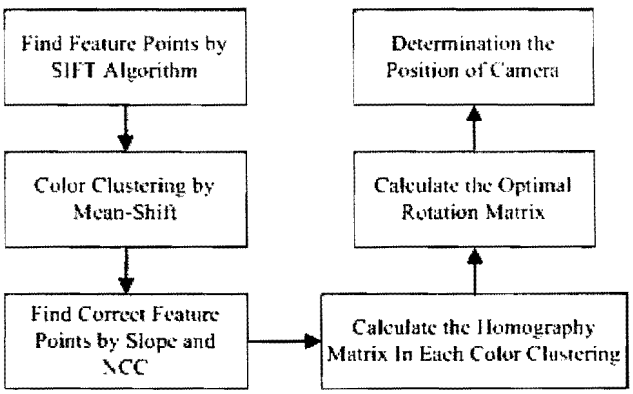
Figure 1 The concept of using IBR from different views (see online version for colours)



In order to achieve the goal of viewing changes in 3D scenes from 2D videos, in the literature, there are some effective and useful methods. Image-based rendering (IBR) technology like McMillan and Bishop's (1995) method, Debevec et al.'s (1998) method and Fitzgibbon et al.'s (2003) method are good technology for constructing 3D graphics. The performance is reasonable and can be implemented in real-time. The concept of IBR is to capture some geometric information from images, such as geometric proxies, epipole consistency and minimal angular deviation in Fitzgibbon et al.'s (2003) method and Buehler et al.'s (2001) method, and to generate 3D images at all viewing positions. Our conceptual diagram is shown in Figure 1, where a new view can be re-constructed from multiple views, and Figure 2 is the flow chart of proposed method.

This concept is very intuitive. If we have a source video of view 1 and view n, we can use IBR to create an image or a video of new views and to compute the angle between videos. Another technology of rendering method is called video-based rendering (VBR). In Zitnick et al.'s (2004) method transforms 2D coordinate in image to world coordinate by direct linear transform (DLT). The method distinguishes foreground and background in image by K-nearest-neighbour (KNN) and mean-shift tracking, and uses the information of foreground and background in previous frames to create a new view image in McMillan and Bishop's (1995) method and Ballan et al.'s (2010) method.

Figure 2 Flow chart of proposed method



View-dependent texture mapping (VDTM) technology like Debevec et al.'s (1996, 1998) method and Blinn and Newell's (1976) method is an effective IBR method. It can establish a basic 3D model by detecting the edge of a building in a photo and optimise the model by using the geometric constraints. Then, the texture is pasted to the model and new views are generated. VDTM is simple, quick, and only requires a small number of photos to synthesis realistic 3D images. But the disadvantage is that it must be used in ordinary buildings or objects with edges very easy to be computed. Another method is called light field/lumigraph in Levoy and Hanrahan's (1996) method and Gortler et al.'s (1996) method. Instead of using a plenoptic function $(x, y, z, \theta, \varphi)$, Adelson and Bergen's (1991) method, Heigl's (1999) method and Cheng's (1995) method propose a new 4D function (s, t, u, v) to represent the flow of light. One can think of two parallel planes in the space. Point (s, t) is on a 'focal plane'; and (u, v) is on other plane called 'camera plane'. Therefore, by using four variables we can describe a ray completely. This method needs to take lots of photos to ensure that there are rays going through every direction. It spends a lot of time in data processing due to the large number of rays. But the result is generally better than VDTM in rendering images. View interpolation/view morphing technology like Chen and Williams' (1993) method, Seitz and Dyer's (1996) method and Narayanan's (1995) method refers to the images information near the new viewpoints and the camera position must be on the baseline (e.g., baseline is a straight line connecting two cameras). The main idea uses the geometric properties to determine pixels in new viewpoints from left or right photos in a sequence. This method is easy and requires a small number of photos. But the scope of new vision is limited and cannot produce the correct picture if there are no reference images.

IBR has some applications such as computer games, virtual travel, TV advertising, etc. In this paper, we propose a new application of IBR. We focus on the interaction between users and films. We can determine the angles of pictures taken, which are very important in texture mapping from a variety of IBR approaches. By calculating the angles of each frame pair, users can move around to select the desired angle of rotation for the current scene during playback. According to the calculation of angles by our method, the system will choose the most suitable frame to continue playback. Our goal is to provide a platform for users, who can watch scene from any arbitrary viewpoint like in the real world. The main challenge of our system is the accuracy of homography matrix. It is very sensitive. A small deviation would seriously affect our result. We use an efficient algorithm to solve this problem and ensure the correctness of the homography matrix.

In Section 2, we explain how to select the feature points and cluster those feature points. In Section 3, we explain how to calculate the correct homography matrix. Section 3.1 explains the relationship of plane parallax. Section 3.2 explains how to decompose a homography matrix to obtain the angle of rotation. In Section 3.3, we explain how to find the optimal rotation matrix and rotation angle. In Section 3.4, we explain how to determination position of camera. Finally, Sections 4 and 5 are the results and conclusions.

2 Feature points matching and clustering

In our method, we obtain the basic technique from Heigl's (1999) method and from clustering the data by mean-shift algorithm. In this section, we describe our method of feature point matching by clustering feature points in the background. Assume there are at least two images of the same scene available and SIFT feature points/descriptors are adopted for matching. Then, we use mean-shift algorithm to cluster feature points of the background image according to their colour and spatial information.

2.1 Feature points selection

In order to find the effective feature points, we consider two methods – SIFT algorithm like Lowe's (2004) method and SURF algorithm like Bay et al.'s (2008) method. In our experiments of SIFT algorithm and SURF algorithm, we found the computing time of SURF algorithm is less than that of SIFT algorithm. But, the number of mismatch feature points in SURF algorithm is more than that in SIFT algorithm. Since we want to obtain both effectiveness and correctness of feature points, we use SIFT algorithm to extract corresponding feature points in two frames and build connections between these feature points. Nevertheless, mismatching the feature points cannot be avoided in most of cases. In order to improve the results, it is necessary to remove the mismatching feature points. The matching error condition can be divided into two types, X and Y directions. By observing the lines from connecting the corresponding matching pairs, we find that these lines are consistent in direction. The lines connecting two matching feature points with slopes far away from the majority slopes of the lines could be the result of mismatching. Because the majority slopes can be represented the major move location of two images. Thus, we can remove the mismatched feature points that are wrong slope in connecting the corresponding matching pairs. In addition, the normalised cross correlation (NCC) in X direction is adopted to further remove the mismatching. NCC is used to determine the degree of correlation between two feature points and in this system we use a 5 * 5 mask for calculation. Note that, our two frames are in different camera positions. And each pixel contains the information about rotation and translation. It is possible that two correctly matching feature points results a small value in NCC. The setting of the appropriate threshold is very important. If we set the value too large, it will reduce the numbers of matching feature points and is not conducive to solving the homography matrix. Therefore, we must ensure that there will be enough matching pairs after removing the mismatching feature points. This can be automatically checked by point numbers necessary for the matrix. Although it cannot completely remove the mismatching error, a small amount of error points will not have too much influence for solving the optimal homography matrix.

2.2 Feature points clustering

The mean-shift algorithm can be used in image smoothing, image segmentation, image clustering and object tracking. In our method, we use the mean-shift algorithm like Heigl's (1999) method and Cheng's (1995) method to cluster regions in the background image and to find feature points in different regions. This approach can reduce the errors in feature points matching. In basic mean-shift algorithm, the $\{x_i \mid i = 1, \ldots, n\}$ represent data points. The equation of mean-shift can be represented by:

$$M_h(x) = \frac{1}{k}\sum_{x_i \in S_h}(x_i - x) \tag{1}$$

$$S_h(x) = \left\{y : (y-x)^T (y-x) \le h^2\right\} \tag{2}$$

where S_h is a circular area, and the radius is h. The parameter k represents number of S_h in x_i. Equation (1) can calculate the average of the displacement vectors between each point in S_h and the centre x.

In our method, we use a kernel function and weight in mean-shift algorithm, since the points may have different impacts on a centre point x. But, the impact on centre point x is the same in the original mean-shift algorithm. Our revised mean-shift algorithm can be represented by:

$$M_h(x) = \frac{\sum_{i=1}^{n} G_H(x_i - x)\, w(x_i)(x_i - x)}{\sum_{i=1}^{n} G_H(x_i - x)\, w(x_i)} \tag{3}$$

$$G_H(x_i - x) = |H|^{-1/2} G\left(H^{-1/2}(x_i - x)\right) \tag{4}$$

$G(x)$ represents the kernel function [proposed in equation (8)], and $w(x_i)$ represents weights of x_i. H represents a symmetric matrix of $d \times d$. In symmetric matrix H, we often define $H = diag[h_1^2, \ldots h_d^2]$ where h_i is the bandwidth respect to the i-dimension. For computation simplicity, it is common to use the same bandwidth for all dimension, i.e., $H = h_I^2$, Therefore, $H = h_I^2$ is used in mean-shift, and the equation can be represented by:

$$M_h(x) = \frac{\sum_{i=1}^{n} G\left(\frac{x_i - x}{h}\right) w(x_i)(x_i - x)}{\sum_{i=1}^{n} G\left(\frac{x_i - x}{h}\right) w(x_i)} \tag{5}$$

Note that, if the following conditions are satisfied, equation (5) will be the same as equation (1) (basic mean-shift method):

1 $w(x_i) = 1$

2 $G(x) = 1$, if $||x|| < 1$

$G(x) = 0$, if $||x|| >= 1$.

We can rewrite equation (5) by:

$$M_h(x) = \frac{\sum_{i=1}^{n} G\left(\frac{x_i - x}{h}\right) w(x_i) x_i}{\sum_{i=1}^{n} G\left(\frac{x_i - x}{h}\right) w(x_i)} - x \tag{6}$$

$$M_h(x) = \frac{\sum_{i=1}^{n} G\left(\frac{x_i - x}{h}\right) w(x_i) x_i}{\sum_{i=1}^{n} G\left(\frac{x_i - x}{h}\right) w(x_i)} \tag{7}$$

And we can obtain the relationship of $M_h(x) = m_h(x) - x$.

Therefore, the steps for our algorithm are:

1 Defining convergence conditions. Starting from x, giving the kernel function $G(x)$, and setting the threshold ε of error.

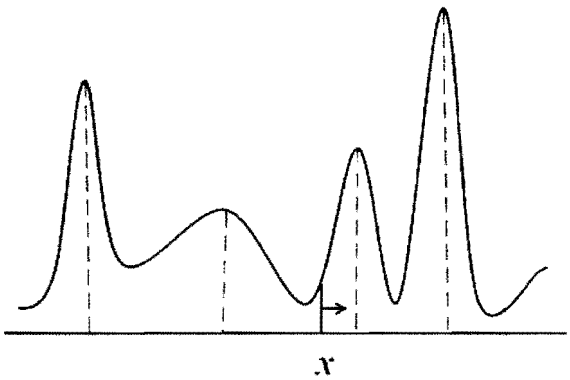
2 Computing $m_h(x)$.

3 Moving centre point to new position.

4 If $|| m_h(x) - x|| < \varepsilon$, the result is convergence. If the result is not convergence, let $x = m_h(x)$ and back to step 2 and compute step (2)~(4) until result is convergence.

The above steps compute the mean-shift algorithm. These steps represent that the new point will be the sum of start point and average displacement vector (the relationship of $m_h(x) = M_h(x) + x$). When the new point position and the old start point are at the same location, the centre point will not move in those steps and represent convergence. The concept is illustrated in Figure 3.

Figure 3 The concept of mean-shift algorithm (see online version for colours)



Next, we describe how to apply mean-shift algorithm in image clustering. We define d as the dimension of the image. When $d = 1$, it represents grey image; and $d = 3$ represents colour image. And, $x = (x^s, x^r)$ represent information of pixel x. Note that x^s is a coordinate in image, x^r is the dimension of pixel (grey image or colour image). The relationship of kernel function in our method can be represented by:

$$K(x) = K\left(\left(x^s, x^r\right)\right) = \frac{c}{h_s^2 h_r^d} k\left(\left\|\frac{x}{h}\right\|^2\right) k\left(\left\|\frac{x}{h}\right\|^2\right) \tag{8}$$

where h_s and h_r represent control variables of clustering, h_s is the size of search region of window and h_r is resolution; C is a corresponding normalisation constant. Then, our steps of image clustering are:

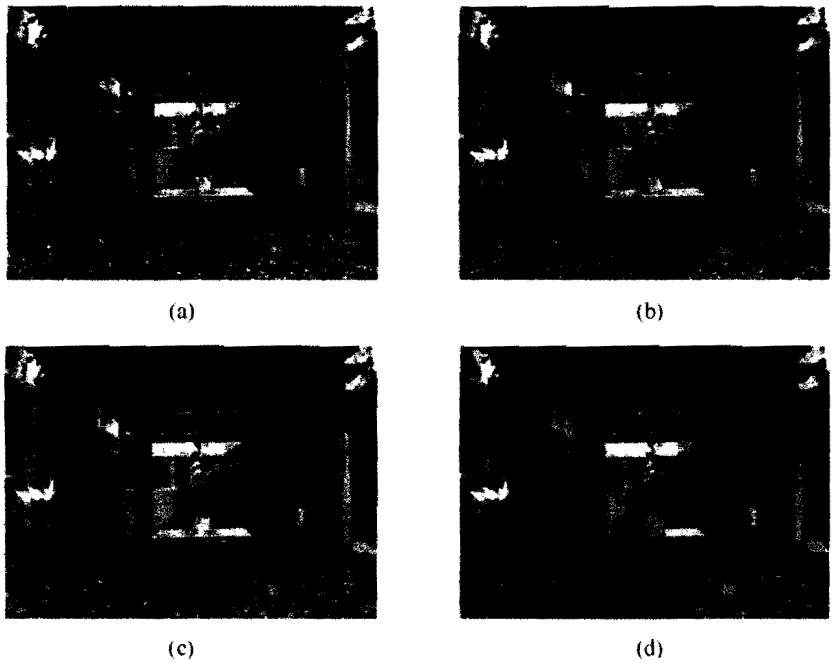
1 start from $y_{i,1} = x_{i,j} = 1$

2 apply the mean-shift algorithm from $y_{i,j+1}$ until the result is converge; and the value of convergence is saved as $y_{i,c}$

3 record $z_i = (x_i^s, y_{i,c}^r)$.

After the above steps, we can obtain the clustering image and x_i, z_i represent the pixels of the original image and the result of clustering, respectively. The image value of the pixel at the location x_i^s is assigned to be the convergence image value $y_{i,c}^r$ after the implementation of mean-shift algorithm. Consequently, pixels belonging to the same cluster are of the same image value, i.e., the same colour value. Note that, h_s and h_r will affect the clustering results. In Figure 4, the h_s and h_r are set from small to large [from Figures 4(a) to 4(d)]. In our algorithm, colour clustering is to improve the accuracy in feature matching. Thus, under clustering is preferred. Thus, we used large values in h_s and h_r which give better results.

Figure 4 Result of mean-shift, (a) $h_s = 5$, $h_r = 5$ (b) $h_s = 10$, $h_r = 10$ (c) $h_s = 15$, $h_r = 15$ (d) $h_s = 20$, $h_r = 20$ (see online version for colours)



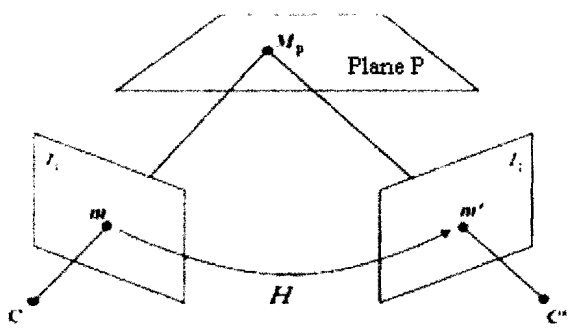
(a) (b) (c) (d)

After the mean-shift algorithm, we obtained the colour clustering image, and we also obtained the information of feature points by SIFT algorithm (in Section 2.1). Therefore, we can use feature points to find the most similar region of colour clustering in two images, and calculate the homography matrix of each region via the coordinates of feature points. We will describe how to calculate the homography matrix and find the optimal rotation matrix in Section 3.

3 Computing angle of background by homography matrix

In this section, we will discuss how to compute the angle of rotation with respect to background images. We compute the angle of rotation according to the geometric principle of camera motion. We use the property of the homography matrix and a schematic diagram shown in Figure 5.

Figure 5 The relationship between different planes



In Figure 5, M_P is a point in the three-dimensional space. Assuming the following, the position of two cameras are C and C', and the projections of M_P on two planes I_1 and I_2 are m and m'. Therefore, the geometric relationship of M_P, m and m' can be represented by:

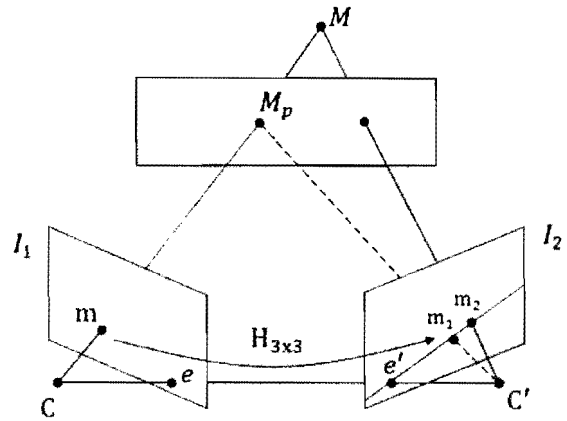
$$sm' = Hm \tag{9}$$

where s is a scale of the zoom. $m = (x, y, 1)$ and $m' = (x', y', 1)$ is a pair of corresponding points in plane I_1 and I_2. H is the homography matrix, which contains information of rotation and translation. Therefore, the angle of rotation can be obtained by matrix decomposition of the homography matrix H.

3.1 The plane parallax

Since we use 2D image to compute the angle of rotation, it is very important to ensure that the corresponding points are on the same plane in the space. But, in most situations, corresponding points will not be on the same plane. This is due to artificial errors when we obtained source video from a camera. So, equation (9) in the last section cannot be used when the corresponding points are not on the same plane. In order to solve this problem, we describe the assumptions and methods in the following. Assume M_P and M are points in 3D space and M_P is based on plane P. In plane I_1, M_P and M are projected onto the same point m, but in plane I_2, M_P and M are projected onto m_1 and m_2 respectively. After computing the homography matrix, M has the projection in m_2. Therefore, in this situation, only M_P can satisfy equation (9). The method is illustrated in Figure 6.

Figure 6 Schematic diagram of plane parallax



Therefore, we have to rewrite equation (9) and use ρ to represent parameter of parallax. The new equation is:

$$sm_2 = Hm + \rho e' \tag{10}$$

3.2 *Solution to the homography matrix*

In the above step, we use $m = (x, y, 1)$ and $m' = (x', y', 1)$, which is a pair of corresponding points in plane I_1 and I_2. Therefore, the equation can be written as:

$$\begin{bmatrix} sx \\ sy \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{11}$$

In order to solve equation (11), we can rewrite the equation as:

$$\begin{aligned} x'(h_{31}x + h_{32}y + h_{33}) &= h_{11}x + h_{12}y + h_{13} \\ y'(h_{31}x + h_{32}y + h_{33}) &= h_{21}x + h_{22}y + h_{23} \end{aligned} \tag{12}$$

Equation (12) has eight parameters (i.e., the scale of H is variable, and h_{33} is usually normalised to 1. Therefore, we need four pairs of corresponding points to solve eight parameters. We can expand equation (12) to:

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1'x_1 & -x_1'y_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -y_1'x_1 & -y_1'y_1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -x_2'x_2 & -x_2'y_2 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -y_2'x_2 & -y_2'y_2 \\ x_3 & y_3 & 1 & 0 & 0 & 0 & -x_3'x_3 & -x_3'y_3 \\ 0 & 0 & 0 & x_3 & y_3 & 1 & -y_3'x_3 & -y_3'y_3 \\ x_4 & y_4 & 1 & 0 & 0 & 0 & -x_4'x_4 & -x_4'y_4 \\ 0 & 0 & 0 & x_4 & y_4 & 1 & -y_4'x_4 & -y_4'y_4 \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \end{bmatrix} = \begin{bmatrix} x_1' \\ y_1' \\ x_2' \\ y_2' \\ x_3' \\ y_3' \\ x_4' \\ y_4' \end{bmatrix} \tag{13}$$

Moreover, according to the characteristic of homography matrix, these points in the three-dimensional space must be on the same plane. Thus, the following algorithm will cluster all feature points and calculate the best homography matrix:

1. Use the mean-shift algorithm to cluster feature points according to colour features.
   - 1.1. Transform the colour space into CIELuv.
   - 1.2. Make the L and U dimension into the 2D array, arrayLU.
   - 1.3. According to the arrayLU, perform the clustering process.
   - 1.4. Eliminate small regions by merging with neighbour regions.
2. For each group, calculate the homography matrix by using the feature points within the group. Solve at least four pair of corresponding points. Thus, if there is note sufficient number of points, the group is rejected.
3. The homography matrix of each group is put into equation (9) to calculate the value of Hm, and compare the deviation between the actual m' and the calculated Hm, and n is the number of matching feature pairs.

$$d = \frac{\sum_{i=0}^{n} (m_i' - Hm_i)}{n} \tag{14}$$

4. The optimal homography matrix is the one with the minimum deviation, that is the one with the smallest d.

3.3 *Optimal rotation matrix and rotation angle*

Owing to the homography matrix property, which includes rotation and translation information of the camera, through the decomposition of the homography matrix, the angle of rotation can be extracted. We assume:

$$K^{-1} * H * K \approx [r_1, r_2, T] \tag{15}$$

where K is the camera's intrinsic parameter, T is the translation matrix, and r_1, r_2 are the two row vectors of the rotation matrix R. We can use camera calibration on a chessboard to get intrinsic parameters K. Different cameras have different intrinsic parameters. The experiments of our paper use a camera with the same settings when we collect experimental videos. Therefore, K is a fixed matrix. From the above, we have

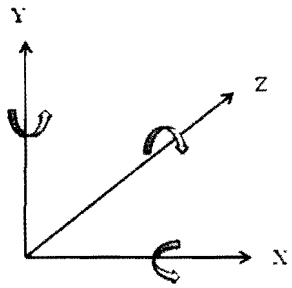
$$r_3 = r_1 \times r_2 \tag{16}$$

$$R = [r_1, r_2, r_3] \tag{17}$$

where r_3 is the third row vector of R. It can be obtained by the cross product of r_1, r_2. Calculating the three row vectors, we get a complete rotation matrix R. In the same method, we record the rotation matrix of each clustering and find the optimal rotation matrix by compare all rotation matrices.

Using the rotation of three-dimension coordinates, we can represent the difference before and after rotation by a rotation matrix. The schematic diagram of rotation of three-dimension coordinates is shown in Figure 7.

Figure 7 The rotation of three-dimension coordinate



The rotation matrix will be changed when the direction of rotation is different along different axis. We show rotation matrix of three cases in the following:

When object rotate θ along Z axis:

$$R_\theta = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{18}$$

When object rotate α along X axis:

$$R_\alpha = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix} \tag{19}$$

When object rotate β along Y axis:

$$R_\beta = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \tag{20}$$

Assuming the real world is in Cartesian coordinates, we can have rotation along the Y axis when we obtained sources in difference view. Therefore we make the normalisation of rotation matrix R and compute the error with rotation matrix along the Y axis. After this step, we can obtain the optimal rotation matrix that has the minimum error d. The equations of this step can be represented by:

$$R = [r_1, r_2, r_3] = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & 1 & r_{23} \\ r_{31} & r_{31} & r_{33} \end{bmatrix} \tag{21}$$

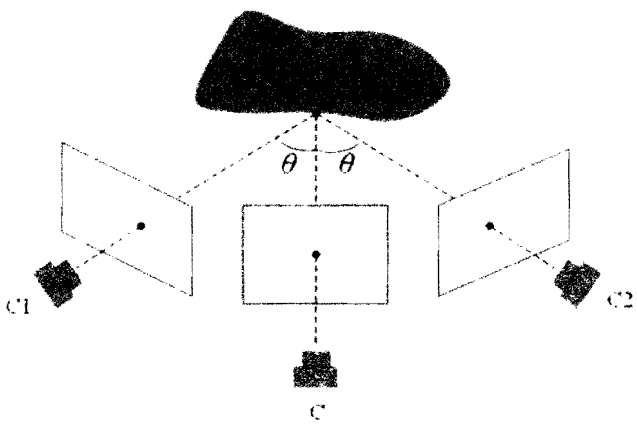
$$d = \|r_{11}| - |r_{33}\| + \|r_{13}| - |r_{31}\| \tag{22}$$

In equation (21), we can obtain the rotation matrix between two frames (along the Y axis). However, owing to possible mismatching, the values in the rotation matrix may not be consistent with the matrix in equation (20). To optimise the result, we use equation (22) and the optimal rotation matrix is the one which give the minimal value of d. After obtaining the optimal rotation matrix, the angle β can be compute by the average angles obtained by $1/4(\cos^{-1}(r_{11}) + \sin^{-1}(r_{13}) + \sin^{-1}(-r_{31}) + \cos^{-1}(r_{33}))$.

3.4 *Determination the position of camera*

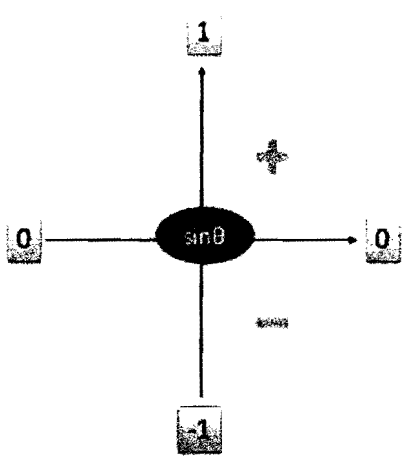
After we compute the angle of rotation β, this angle is the camera rotation angle between two frames. Thus, the position of view will be from left to right when we want to rotate the scene. This situation also affects our result after rotation. Because, we can not only use the rotation angle to determine the correct frame (left frame or right frame). The schematic diagram is depicted in Figure 8 which shows there are two cases for camera C to rotate angle. Therefore, we have to know the position of camera and to find the correct frame.

Figure 8 Position of camer



To determine the position of the camera, we use the trigonometry of the rotation matrix. The schematic diagram is shown in Figure 9. In Figure 9, if $\sin\theta$ is a positive number, the camera moves to the left (object moves to the right). If $\sin\theta$ is a negative number, the camera moves to the right (object moves to the left). In the model, the angle of rotation β is between 0° and 90°.

Figure 9 Schematic diagram of camera direction (see online version for colours)



4 Experiments

Our experimental system is divided into three parts. The first part is for video pre-processing and for calculating the angle of each frame. The second part is to show some assumptions that may affect the performance in our method, and to prove assumptions are true by experiment. We also show results of our method in the second part. The third part is comparing the advantages and disadvantages in proposed method.

4.1 Video pre-processing and for calculating the angle

First, a user can select a video to start playing, and press the left mouse button to pause the screen when an interesting scene is encountered. The user can move the mouse around to select the desired rotation angle. Our system will load the data of current frame and select the most suitable frame from the record. After releasing the left mouse button, the screen will jump to the selected frame and continue to play video (see Figure 10).

The first part costs most of the CPU time in our system, owing to the number of frames needed to be handled. Fortunately, this part is only executed once. If the user watches the same videos, the intermediate data have been produced. The second part is completely proceeded in real-time, similar to the Google Map's street view service, fast and efficient scene change allowing users to have a better interaction with films. Figure 11 is the method of collecting videos in our experiments. Users can shoot many videos in any position. If we provide more samples, the results will be more accurate.

4.2 Analysis the impact of external factors in proposed method

In this section, we consider three assumptions that may affect the performance of our method. Assumptions are made on distance (depth), complexity, and shape of object. We justify our assumptions by implementing a comprehensive experiment.

Experiment I The impact of the depth between objects and camera.

Assumption I Short distance image performs better than long distance image.

In our method, we find the feature points by SIFT algorithm, and the number of feature points will affect the results. Our method can compute the optimal rotation matrix with a large number of feature points. In our assumption, we believe that there will be more feature points detected if images are captured in a short distance than those detected in a long distance between object and camera. We set up experiment and comparison as the following. The source images of the CD box are acquired in difference distances (24 cm and 16 cm). The multi-views images are taken for every ten degrees from 0°~30°. Figures 12 and 13 show the source images.

Figure 10 User input the angle of rotation and result (see online version for colours)

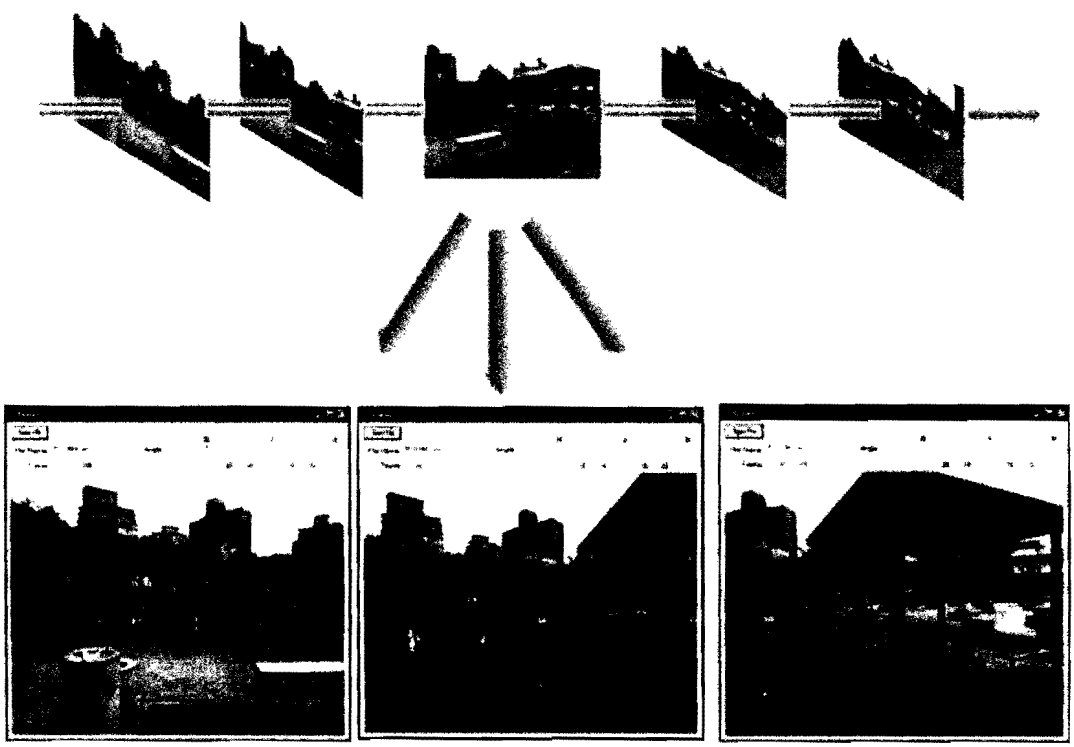


Figure 11 The camera positions of collecting videos

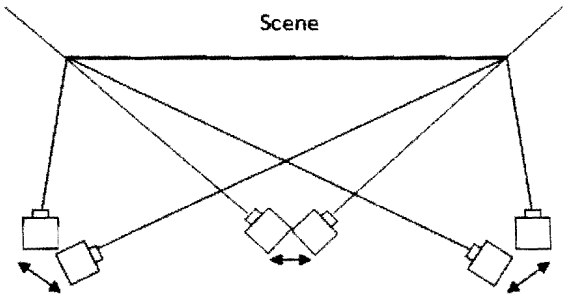


Figure 12 Different angles of the object in image of Experiment I from a long camera distance (see online version for colours)

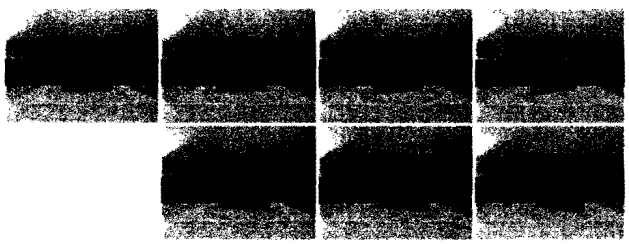


Figure 13 Different angles of the object in image of Experiment I from a short camera distance (see online version for colours)

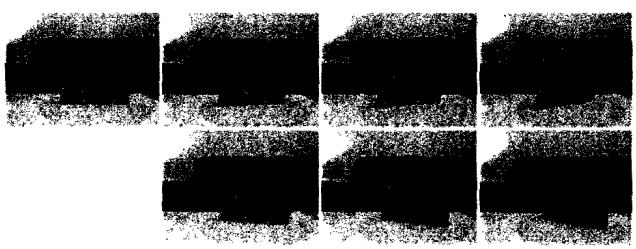


Figure 14 Result of experiment and comparison in Experiment I (see online version for colours)

Experiment I: Depth between Objects and Camera		
Angle	**Experiment (24cm)**	**Comparison (16cm)**
L0°↔L10°	11°	9°
L0°↔L20°	16°	19°
L0°↔L30°	31°	30°
L0°↔R10°	8°	9°
L0°↔R20°	15°	21°
L0°↔R30°	12°	29°
L10°↔L20°	7°	10°
L10°↔L30°	21°	21°
L10°↔R10°	18°	19°
L10°↔R20°	28°	30°
L20°↔L30°	10°	10°
L20°↔R10°	28°	24°
R10°↔R20°	11°	9°
R10°↔R30°	16°	16°
R20°↔R30°	12°	10°

To test the assumption, the rotation angle between two images is estimated by the proposed method. Then estimation errors are compared between these two distances. Figure 14 shows the estimated angle value of the experiment. For example, L10° ↔ R20° represents two rotations experiments; one is made from left 10° of the image 1 to right 20° of the image 2 and the other one is made from right 20° of the image 2 to left 10° of the image 1, so the true rotation angle should be 30° in both cases. In our method, the estimated angle would be the same no matter is obtained from image 1 to image 2 or vice versa. The figure marked in red in Figure 14 represents the error is more than ten degrees from the true rotation angle.

Table 1 summarises the result in Figure 13 which covers 30 rotation tests. In Table 1, we compute the value of average errors with experiment and comparison. Note that, the number of error is 2 since it happened on rotations of L0° ↔ R30° and R30° ↔ L0°. From Table 1, our assumption on 'short distance image performs better than long distance image' is confirmed since the average error is 1.2° compared to 3.2°.

Experiment II The impact of the complexity of object colours.

Assumption II Simple colour image performs better than complex colour image.

Table 1 Result of analysis in Experiment I (angle of rotation is 0°~30° in image)

	Sum number	*True number*	*Error*	*Average error*
Experiment (Z = 24 cm)	30 images	28 images	2 images	3.2°
Comparison (Z = 16 cm)	30 images	30 images	30 images	1.2°

In our method, mean-shift algorithm is adopted for colour-spatial clustering followed by cluster wise feature matching. Therefore, those feature points lying on the same plane but with different colours will belong to different clusters. This situation may cause problems. One is the insufficient matching pairs in computing the homography matrix for some clusters. In addition, if there are sufficient matching pairs, the computed homography matrix may not be the optimal one. Therefore, we make the Assumption II. To set up the experiment, the complex images are taken from a CD box with a colourful design as in Figure 15 and the simple images are as in Figure 13. In our source images, the background is the same (the camera is fixed) and images are acquired for every ten degrees as before. Therefore, with the only difference in CD box, we ensure that images in Figure 15 are more complicated than those in Figure 13.

We test the assumption as before. Figure 16 shows the result of experiment The 'NULL' in Figure 16 means there is no estimated angle obtained.

Figure 15 Different angles of object in image of experiment in Experiment II (see online version for colours)

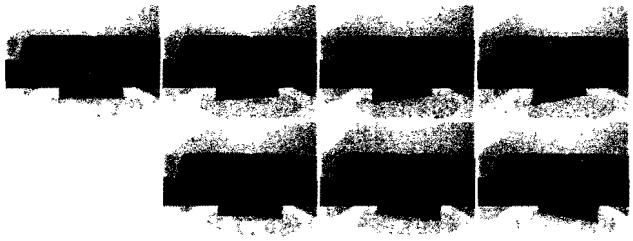


Figure 16 Result of experiment and comparison in Experiment II (see online version for colours)

Experiment II: Complexity of Object Colors		
Angle	**Experiment (complicated)**	**Comparison (simple)**
L0°↔L10°	18°	9°
L0°↔L20°	NULL	19°
L0°↔L30°	32°	30°
L0°↔R10°	15°	9°
L0°↔R20°	21°	21°
L0°↔R30°	47°	29°
L10°↔L20°	15°	10°
L10°↔L30°	42°	21°
L10°↔R10°	24°	19°
L10°↔R20°	57°	30°
L20°↔L30°	9°	10°
L20°↔R10°	20°	24°
R10°↔R20°	10°	9°
R10°↔R30°	20°	16°
R20°↔R30°	13°	10°

Table 2 Result of analysis in Experiment II (angle of rotation is 0°~30° in image)

	Sum number	*True number*	*Error*	*Average error*
Experiment (complicacy)	30 images	22 images	8 images	7.5°
Comparison (simple)	30 images	30 images	0 images	1.2°

Table 2 summarises the results of Figure 15. From Table 2, our assumption on 'simple colour image performs better than complex colour image' is confirmed since the average error is 1.2° compared 7.5°. In some cases, the rotation angle cannot be estimated (the NULL cases), it may due to the following reasons:

There are fewer than four feature points in every cluster after mean-shift clustering execution. It takes at least four matching pairs to determine the homography matrix.

When calculating the optimal rotation matrix from equations (20) and (21), the values r_{11}, r_{13}, r_{31}, r_{33} are very different. Since r_{11} and r_{33} are both for estimation of $\cos\theta$ and r13 and $-r_{31}$ are both for estimation of $\sin\theta$, very inconsistent values in r_{11}, r_{13}, r_{31}, r_{33} lead to no conclusion in estimating the rotation angle θ.

Experiment III The impact of the shape of objects.

Assumption III Smooth shaped object image performs better than irregular shaped object image.

In characteristic of homography matrix, we know the relationship between feature points and parallax. If feature points are in the same plane, there is no parallax. Therefore, if the object surface is flat, the feature points will be in the same plane. Otherwise, feature points will not be in the same plane. Based on this, we make Assumption III. To verify the assumption, a 'piggy bank toy' is used as the irregular shaped object. Their images are shown in Figure 17. Again, images in Figure 13 are used for the smooth shaped object images. These two set of images are taken under the same conditions and the same background.

Figure 17 Different angles of object in image of experiment in Experiment III (see online version for colours)

Figure 17 shows the result of experiment and Table 3 summarises the results of Figure 18. As in Table 3, the CD box images have a much better performance comparing to the 'piggy bank toy' images. The experiment result implies that feature point lying on the same plane is very important. When feature points are not in same plane, as in 'piggy bank toy' images, the rotation matrix is prone to error. Therefore, Assumption III is correct.

Figure 18 Result of experiment and comparison in Experiment III (see online version for colours)

Experiment III: Shape of Objects		
Angle	**Experiment (Irregular)**	**Comparison (Square)**
L0°↔L10°	9°	9°
L0°↔L20°	42°	19°
L0°↔L30°	7°	30°
L0°↔R10°	12°	9°
L0°↔R20°	18°	21°
L0°↔R30°	11°	29°
L10°↔L20°	84°	10°
L10°↔L30°	9°	21°
L10°↔R10°	46°	19°
L10°↔R20°	NULL	30°
L20°↔L30°	6°	10°
L20°↔R10°	31°	24°
R10°↔R20°	9°	9°
R10°↔R30°	57°	16°
R20°↔R30°	52°	10°

Table 3 Result of analysis in experiment III (angle of rotation is 0°~30° in image)

	Sum number	*True number*	*Error*	*Average error*
Experiment (irregular)	30 images	12 images	18 images	18.9°
Comparison (square)	30 images	30 images	0 images	1.2°

Figure 19 shows three results by our method. Figures 19(a) and 19(b) are video series in the parking lot with total length of nine seconds. Figures 19(c) and 19(d) are longer videos of 30 seconds. Each scene has a different number of films owing to different frame rates and cut into 640 * 480 frame size. The frame may be from the same or different videos. The programme will select the most appropriate results and resume playing video.

4.3 Advantages and disadvantages of the proposed method

In Hofsetz et al.'s (2004) method, authors use IBR and geometry reconstruction technique to reconstruct the 3D environment through depth images. In depth images, the authors detect features by edge detection, and search the epipolar lines in all other images to find the best match. Calculate the sum of squared differences (SSD) for each image to compare the areas of interest along the epipolar lines in two or more images, and find the depth images by DeLaunay interpolation and smoothing. In our proposed method, we find feature by SIFT algorithm, and compute the optimal rotation matrix of frames in each video and compute the optimal angle between frames. We did not consider geometry reconstruction. We only use photographic methods when we obtain videos. Therefore, we ignored the geometry and focused on the corresponding relationship of angle and direction in each videos.

In Ohta et al.'s (2007) method, authors use IBR in 3D video that are constructed from multi-views camera. In Ohta et al.'s (2007) method, objects must be established by a 3D model that is a single plane in 3D space. Therefore, objects (soccer players) must be extracted first. In object extracting, authors use background subtraction and binarisation to recognise objects and background. Second, the algorithm tags the foreground by image labelling. Third, the algorithm segments foreground regions depend on region sizes (i.e., finding large area). Therefore, objects can be determined by foreground regions and by constructing a simple 3D model of foreground regions. Ohta et al.'s (2007) method must use real-time hardware for camera, which can be expensive. In our proposed method, the positions of objects can be ignored. This is due to the fact that our source are continues video frames when user moves camera angles.

We summarise Christian Hofsetz's method, Yuichi Ohta's method, and our method, from the perspectives of relevant theory, input, presentation environment, distortion and reality, cost, and computation time. Relevant theory is the related theories/algorithms used in each method. Input is the input format of each method. Presentation environment is the display environment in each method. Distortion and reality are to evaluate the distortion and reality of results in each method (we consider foreground and background, respectively). Cost and computing time are considering the cost in equipment and computing time in processing. We present the analysis result of the above items in Table 4.

Figure 19 Three calculating results by our method (see online version for colours)

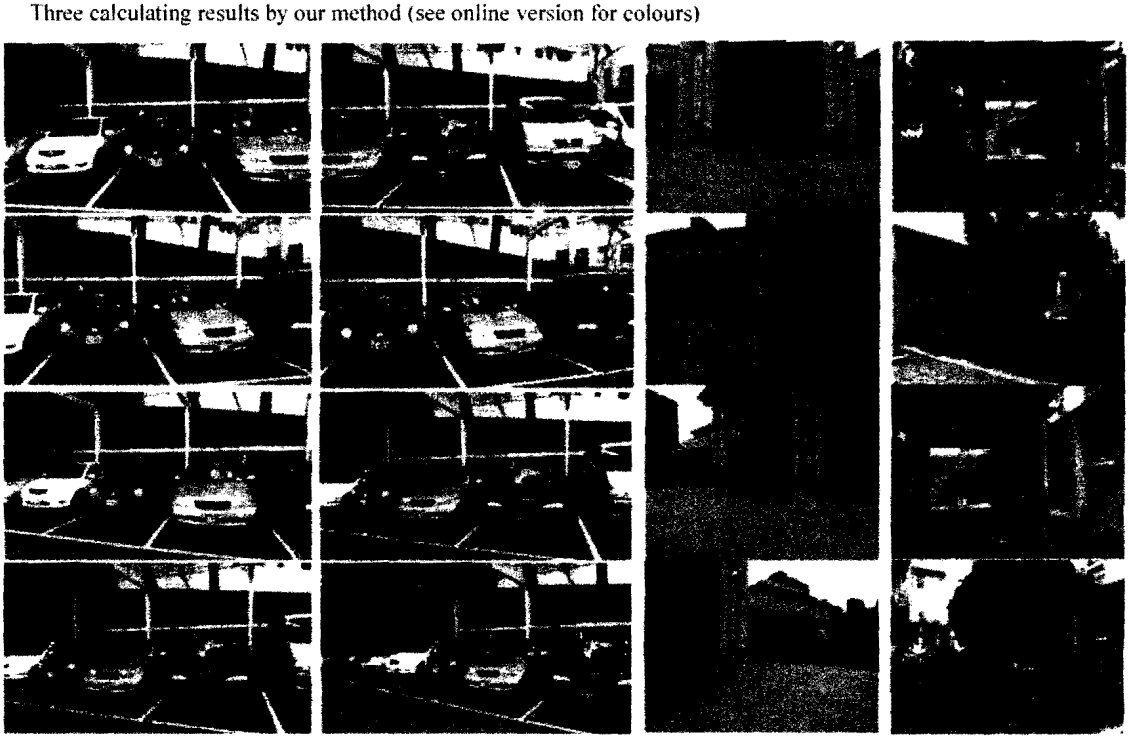


Table 4 Analysis and comparison

	Christian Hofsetz's method (Hofsetz et al., 2004)	*Yuichi Ohta's method (Ohta et al., 2007)*	*Proposed method*
Relevant theory	Image-based rendering	Image-based rendering	Image-based rendering
	Depth extraction	3D technology	SIFT algorithm
	Epipolar geometry	(modelling, texture, 3D CG space)	Mean-shift algorithm
	Edge detected	Image labelling	Homography matrix
	Perspective projection approximation	Stereo matching technique	
Input	Images	Video	Video
Presentation environment	3D space	3D CG space	Video
Distortion	Foreground: middle	Foreground: high	Foreground: low
Reality	Background: high (no information)	Foreground: high	Foreground: low
Cost	High	Low	High
Compute time	Low	Expensive	Low

5 Conclusions

This paper proposes a method for calculating the angle of rotation due to camera motion. The application of this technology is different from a traditional interactive video, which allows basic video interactions such as zoom-in/zoom-out and changing region of focus, with the camera fixed. We use the SIFT algorithm to find feature points between two frames from difference angles and record the coordinate in video to ensure the continuity when the user moves the viewpoint. In our method, we compute an optimal rotation matrix and define the position of camera with precise view angles in video. A series of experiments were tested to justify our achievement and show our limitations. Although the system can be implemented by the users in real-time, the length of video could lead to a performance problem during the pre-processing stage. Another future work is on how to guide users to make a better source video in terms of selecting camera positions and the speed to move the camera. Better selection of camera motion results in a more precise rotation matrix. Thus, the arbitrary viewing angle can be constructed in a precise manner. The mechanism we have proposed is an initial stage for constructing real 3D videos, in which objects are presented in 3D relative coordinates. If so, the changing of viewing angles will also include changing of distance between the camera and the object.

References

Adelson, E.H. and Bergen, J.R. (1991) 'The plenoptic function and the elements of early vision', in M. Landy and J.A. Movshon (Eds.): *Computation Models of Visual Processing*, pp.3–20, MIT Press, Cambridge.

Ballan, L., Brostow, G.J., Puwein, J. and Pollefeys, M. (2010) 'Unstructured video-based rendering: interactive exploration of casually captured videos', in *ACM Trans. on Graphics*, Vol. 29, No. 4, pp.87:1–87:11.

Bay, H., Ess, A., Tuytelaars, T. and Van Gool, L. (2008) 'SURF: speeded up robust features', *Computer Vision and Image Understanding (CVIU)*, Vol. 110, No. 3, pp.346–359.

Blinn, J.F. and Newell, M.E. (1976) 'Texture and reflection in computer generated images', in *CACM*, pp.542–547.

Buehler, C., Bosse, M., McMillan, L., Gortler, S.J. and Cohen, M.F. (2001) 'Unstructured lumigraph rendering', in *SIGGRAPH 2001 Conference Proceedings, ACM SIGGRAPH Annual Conference Series*, pp.425–432.

Chen, S.E. and Williams, L. (1993) 'View interpolation for image synthesis', in *SIGGRAPH '93*, Anaheim, California, August 1–6, in *Computer Graphics Proceedings, Annual Conference Series*, pp.279–288.

Cheng, Y. (1995) 'Mean shift, mode seeking, and clustering', in *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. 17, No. 8, pp.790–799.

Debevec, P., Taylor, C. and Malik, J. (1996) 'Modeling and rendering architecture from photographs: a hybrid geometry- and image-based approach', in *SIGGRAPH '96*, pp.11–20.

Debevec, P.E., Yu, Y. and Borshukov, G.D. (1998) 'Efficient view-dependent image-based rendering with projective texture mapping', in *Eurographics Rendering Workshop*, pp.105–116.

Fitzgibbon, A., Wexler, Y. and Zisserman, A. (2003) 'Image-based rendering using image-based priors', in *International Conference on Computer Vision (ICCV)*, pp.1176–1183.

Gortler, S.J., Grzeszczuk, R., Szeliski, R. and Cohen, M.F. (1996) 'The lumigraph', in *SIGGRAPH '96*, pp.43–54.

Heigl, B. (1999) 'Plenoptic modeling and rendering from image sequences taken by hand-held camera', in *Mustererkennung, DAGM-Symposium*, Vol. 21, pp.94–101.

Hofsetz, C., Ng, K., Chen, G. and McGuinness, P., Max, N. and Liu, Y. (2004) 'Image-based rendering of range data with estimated depth uncertainty', in *IEEE Computer Graphics and Applications*, Vol. 24, No. 4, pp.34–41.

Levoy, M. and Hanrahan, P. (1996) 'Light field rendering', in *SIGGRAPH '96*, pp.31–42.

Lowe, D.G. (2004) 'Distinctive image features from scale-invariant keypoints', *International Journal of Computer Vision*, Vol. 60, No. 2, pp.91–110.

McMillan, L. and Bishop, G. (1995) 'Plenoptic modeling: an image-based rendering system', in *SIGGRAPH '95, in Computer Graphics Proceedings, Annual Conference Series*, SIGGRAPH, pp.39–46.

Narayanan, P.J. (1995) 'Virtualized reality: concepts and early results', in *IEEE Workshop on the Representation of Visual Scenes*, IEEE, pp.69–76.

Ohta, Y., Kitahara, I., Kameda, Y., Ishikawa, H. and Koyama, T. (2007) 'Live 3D video in soccer stadium', in *International Journal of Computer Vision*, Vol. 75, No. 1, pp.173–187.

Seitz, S. and Dyer, C. (1996) 'View morphing', in *SIGGRAPH '96. Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, pp.21–30.

Zitnick, C.L., Kang, S.B., Uyttendaele, M., Winder, S. and Szeliski, R. (2004) 'High-quality video view interpolation using a layered representation', in *ACM Transactions on Graphics (TOG) – Proceedings of ACM SIGGRAPH*, TOG Homepage.Vol. 23, No. 3, pp.600–608.