ACCV2002: The 5th Asian Conference on Computer Vision, 23--25 January 2002, Melbourne, Australia.

Image Stabilization using Practical Information from Aerial View

Toshio Moriya*,**　Haruo Takeda*,**
*Systems Development Laboratory, Hitachi, Ltd.
1099 Ohzenji, Asao, Kawasaki, 215 Japan,
**Nara Research Center, Telecommunications Advancement Organization of Japan
moriya@sdl.hitachi.co.jp　takeda@sdl.hitachi.co.jp

Abstract

In this paper, we describe a practical method for stabilizing an image sequence captured by a wide-angle camera mounted on a helicopter. For our application, an adjustment of 3-D rotation of the image is effective, and an appropriate camera pose estimation method is required. We demonstrate that the method using vanishing point is most suitable to our purpose. Since no information about it can be found in the image, we adopt other features that have similar properties as the vanishing point, and we prove that they are adequate for the stabilization even though the estimation is not strictly correct. We also show an actual algorithm that can avoid a problem that the de-rotated image may include non-picturized area.

1. Introduction

Several kinds of immersive display environments have been developed in the field of virtual reality. We have developed a multi-screen type system. A viewer surrounded by large flat screens can experience a highly immersive feeling in wide-angle images. This is similar to previous systems [3] but adopts a motion-based ride [12] (figure 1). Using this environment, we have developed an application in which the viewer can experience a sensation of flying around the world while riding a virtual flying machine.

Since the images are displayed not only on a front screen but also on right, left, and bottom screens, the original image needs to be provided in a special form. Images that have such an extremely wide view angle are known as omni-directional images.

In our application, the image used is an aerial view taken from a high place in the sky. We shot such an image with a camera mounted on a helicopter. Since the camera system needs to be compact, we have chosen it with a fish-eye lens. To capture a high-resolution image, we used a 35 mm movie film camera instead of a video camera. By taking a digital scanning from the film, we could obtain a more than $4{,}000 \times 3{,}000$ pixels resolution image.

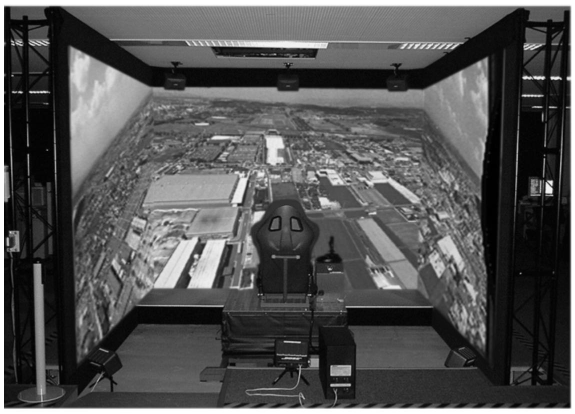


Figure 1. System environment.

This paper discusses a method to stabilize an image captured by this shooting method. Since the helicopter is jolting while it flies, it is unavoidable that some unexpected vibrations are mixed into the image. Therefore, we considered a method to adjust them by image processing.

Several studies about image stabilization have been already done. In Ref. [6], a 2-D affine estimation and image registration algorithm was implemented, and the stabilization in real-time could be performed based on the 2-D affine model. In Ref. [11], a parallel pipeline image processing hardware was developed. It tracks a small set of features and the images are warped by using a 2-D feature-based multi-resolution motion estimation technique. Ref. [9] computed ego-motion from 2-D stabilization by estimating the 2-D affine motion of a single image region, which is used to cancel the 3-D rotation of the camera motion, then the 3-D camera translation is computed by finding the FOE in the translation-only sequence. Ref [16] assumed that a camera was mounted on a vehicle, and it moved on the planar surface. Using these constraints, the depth of the scene was estimated and the image was converted.

We can roughly divide such previous works into two types. One is a type that aims a real-time performance by using some approximation such as the affine model. Another is a type that uses particularities of the image such as the scene captured from a moving vehicle, and estimate the 3-D structure of the scene. Since our application uses a passive movie but has a singularity that the projection screen is very large and wide, high accuracy is more important than real-time computation. Therefore we make the latter type of approach.

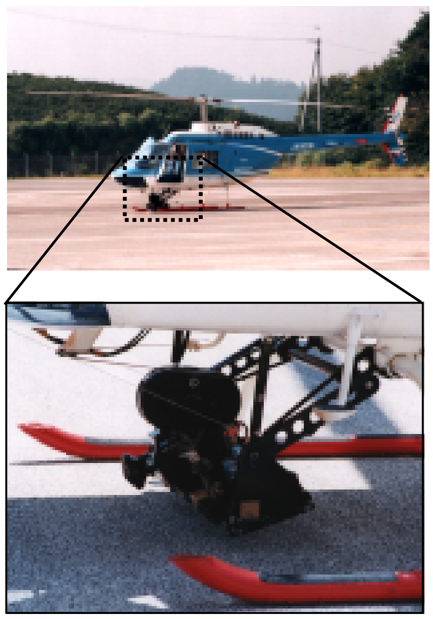
A basic idea for the stabilization in this paper is a modification by making the 3-D rotation. This idea has already been mentioned in the previous works. But in this paper, we demonstrate that 3-D rotation is more effective for our application (aerial view), and propose a practical method to compute the rotation by using the information about specific objects in the scene.

The conversion from fish-eye format into the perspective format is accurately performed by using the intrinsic parameters of the fish-eye lens [15]. Therefore we will deal the image in the perspective format rather than the fish-eye format, in the following discussion.

2. Camera Setup

As figure 2 shows, we set up a 35 mm movie film camera on the lower forward side of a helicopter. The camera was fixed to the foot frames under the cabin and oriented 45 degrees down toward the full-frontal direction.

Under such a condition, the problem is that the captured image invariably contains unexpected vibrations from the jolts of the helicopter due to aeronautic reasons such as air instability or mechanical reasons such as engine vibrations. One method to solve this problem is to use some physical attachment device that absorbs the transmission of the helicopter's jolts to the camera. These are known as a spring-based stabilizer, a gyroscopic stabilizer, or steady-cam systems. Because we used a 35 mm film camera, which is larger and heavier than a conventional video camera, the attachment devices designed for the video cameras were not suitable. Furthermore, the helicopter we used was not specialized for aerial photography and was not equipped with a fixing bracket for a camera device. These restrictions should not reduce the flexibility of the shooting plan, and should not increase costs. For these reasons, we think it meaningful to establish a method to capture the stabilized images by image processing in the post process instead of using physical devices in the actual process.



3. Analysis of Image Vibration in our Application

There are two factors that give an image vibration: translational movement of the camera and rotational movement of the camera. We now consider the influence of these factors by using an actual value. We assume that the camera shoots the ground from the helicopter at an altitude of 150.0 m. When the helicopter moves 0.1 m translationally due to jolts in consecutive frames (1/30 sec), it generates an azimuth angle $\Delta\theta \approx$ 0.1/150.0 radian. If we represent this angle on 640×480 pixel perspective image whose horizontal view angle is 90 degrees, it can be estimated as $\Delta x = \frac{640}{2}\frac{\tan\Delta\theta}{\tan(90°/2)}$ and its actual value is calculated as only Δx = 0.21 pixels. On the other hand, we also consider the case where the helicopter's forefront point moves 0.1 m, but its center of the gravity located 2.0 m from the forefront point does not move (figure 3). This movement generates a rotation of $\Delta\psi \approx$ 0.1/2.0 radian, and this azimuth makes deviation Δx = 16.0 pixels on the same perspective image. As these estimations shows, the image vibration, i.e., the deviation of some image point caused by the movement of the camera, is mostly generated by the rotational factor rather than the translational factor. This aspect is particular to our shooting condition in which all of the shot objects are located far from the camera (more than 150.0 m). If there is an object near the camera, the translational movement of the camera also gives a perceptible deviation in the image.

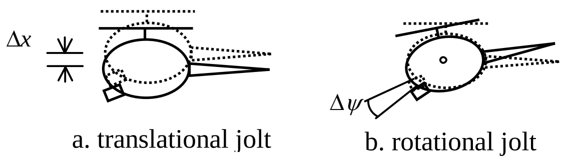


Figure 3. Movement of the camera.

4. Principle of Image Stabilization

One of the most primitive stabilizing methods is to translate and rotate the images in 2-D so that some feature points become stable in the sequential images. Some commercialized systems have been developed based on this principle, and it is practical enough if we assume a use with normal images. However, if we consider wide-angle images, this method becomes inadequate. As we mentioned above, the deviation (vibration) is mostly caused by 3-D rotation of the camera. If the view angle is narrow, the 3-D rotation of the camera can be approximated by 2-D translation and rotation of the image. But if the view angle is wide as in our application, a 2-D technique is not suitable, because the same rotation $\Delta\phi$ makes quite a different deviation in accordance with the areas in the image plane. Therefore we make the stabilization by using 3-D rotation of the image.

It is well known that an arbitrarily directional image can be generated from any image [15]. By using this method, the stabilization can be accomplished so that all of the frame's images are modified to be oriented in the same direction. Such a 3-D rotation is calculated as

$$\boldsymbol{m}_i' = \boldsymbol{R}\boldsymbol{m}_i \qquad (1)$$

where $\boldsymbol{m}_i$ denotes the 2-D position (x_i, y_i) on the original image represented by the spherical mapping expression $\frac{1}{\sqrt{x_i^2+y_i^2+f^2}}\begin{pmatrix}x_i & y_i & f\end{pmatrix}^t$, and $\boldsymbol{m}_i'$ denotes the 2-D position (x_i', y_i') on the rotated image represented by spherical mapping expression, and f is focal length.

5. Camera Pose Estimation

To make a stabilization, namely to make a 3-D rotation of the image, it is necessary to know the camera orientation of each frame, therefore, a 3-D camera pose estimation method from the captured images is required.

There are some different kinds of estimation method [4][5][7][8][14]. It is a matter of choice to consider the most effective one by taking into account of the actual application. Since our application is specifically defined and the properties of the images we deal with are clearly limited, these considerations are practical and significant.

Generally, the purpose of the camera pose estimation is to calculate an accurate value of the camera parameters in order to estimate 3-D position of the shooting object. However, our purpose is different. It is not required to estimate a correct value as long as the stabilization can be performed effectively.

In the pose estimation methods that use some correspondence of the image of the general points, the camera translation and rotation are calculated concurrently. Specifically, the rotation is estimated to satisfy the epipole constraint, and at the same time, the camera translation is calculated by using the position of the epipole.

However if we select a vanishing point as a feature of the images, the process can be apparently separated [2]. This property is convenient for our purpose in which only the camera rotation is needed.

When several vanishing points $\boldsymbol{p}_{vi}^{(j)}$ (i=1,2...N) in the camera-j image are observed in the same position of $\boldsymbol{p}_{vi}'^{(k)}$ (i=1,2...N) in the camera-k image after rotating $\boldsymbol{R}^{(kj)}$, the orientations of both the cameras are the same direction. This relationship is denoted as

$$\boldsymbol{p}_{vi}^{(j)} = \boldsymbol{R}^{(kj)}\boldsymbol{p}_{vi}^{(k)} \quad (i = 1,2,...,N), \quad (2)$$

where $\boldsymbol{p}_{vi}'^{(k)} = \boldsymbol{R}^{(kj)}\boldsymbol{p}_{vi}^{(k)}$, and $\boldsymbol{R}^{(kj)}$ is a relative rotation between camera-j and camera-k [2].

If we compare with the estimation method that uses general point, the method using the vanishing point has advantage as follows:

(1) Robust to the measurement error.
(2) The number of the points can be few (minimum two).
(3) Calculation is very simple.

6. Practical Method

6.1 Selection of the Features

The vanishing point is very effective as a feature of the image to estimate the camera pose, especially the rotation. However, in an actual image that we used, the vanishing point did not appear directly, and no supplementary information such as pairs of parallel lines could be found to determine it . Therefore, we try to find other features that are not genuine vanishing points but have similar properties.

The images we use for stabilization were taken from an aerial view and taken by a fish-eye lens. The image was captured from about a 150 m height. We selected a day with ideal weather conditions to shoot a fine picture. Gazing at the images captured under such conditions, we identified two important characteristics: (1) Both the ground surface and the sky were throughout all of the frames. There were some mountains on the farther side of the ground and some clouds in the sky. (2) Because the shooting was done in the daytime, the shadow of the helicopter needed to appear in the scenes. We take advantage of these characteristic features of the images for the 3-D pose estimation. Specifically we use: (1) images of the mountains, (2) images of the clouds, (3) image of the helicopter's shadow.

Because the mountains and clouds are normally located in the distance, we can easily assume that they may substitute for the vanishing point. In addition, we can also substitute the shadow of the helicopter.

6.2 Properties of the Helicopter's Shadow

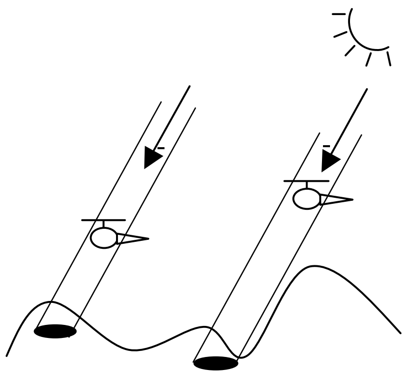


Figure 4. Helicopter's shadow.

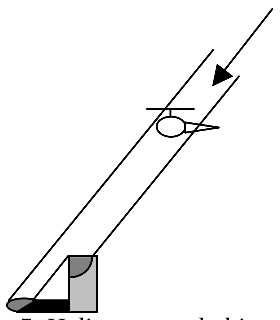


Figure 5. Helicopter and object shadows.

As figure 4 shows, the shadow of the helicopter occurs on the opposite side of the sun. Because the sun is at an extreme distance, it can be treated as a point of infinite distance, and thus the direction towards it from the camera position is constant, even if the camera moves. Hence the direction from the camera to the shadow is also constant. Therefore, the image of the helicopter's shadow has the same property as a vanishing point. As figure 4 indicates, it is significant that this property is valid regardless of the shape of the ground on which the shadow is projected. Also, as figure 5 indicates, the helicopter's shadow can appear anywhere even when there is a shadow of another object.

6.3 Incompleteness of the Features as a Point at Infinity

We selected three features in the scene to estimate the 3-D pose of the camera. However, if we compare them with an actual vanishing point (an image of a point at infinity), they have the following aspects of incompleteness.

(1) Because the positions of the mountain and clouds are not true infinite distances, their images move a little when the camera is translated, even if the orientation of the camera is completely constant.
(2) The shapes of the clouds are not consistent and the position changes with time.
(3) Because the sun moves in relation to the earth, the direction toward the helicopter's shadow varies.

However, these inadequacies have little effect on the stabilization. We will explain these points by using an actual value.

6.3.1 Distance to the mountain

Geographic conditions show that the mountain we selected was located more than 10 km from the shooting area. The helicopter flew at about 1.85 m per frame or 200 km/h. This translational movement of the helicopter results in azimuth angles of up to $\Delta\theta \approx$ 1.85/10,000 radians, and these angles have 0.052 pixel deviations using 640×480-pixel perspective images when the horizontal view angle is 90 degrees. This value seems to be negligible.

6.3.2 Movement of the clouds

To estimate the movement of the clouds, we assumed that they were 3,000 m high and that the wind speed at that altitude was 100 m/minute. If the clouds were projected onto an image near the ground-line (less than 30 pixels above the ground-line in the perspective image described previously), the distance from the camera to the clouds could be determined, and it was calculated to be 32.6 km. We considered the azimuth angle generated by the movement of the camera (200 km/h) and the movement of the object (100 m/minute) located at a far distance (32.6 km), and found that the deviation brought by this azimuth in the 640×480-pixel perspective image was only 0.022 pixels; it was almost negligible.

6.3.3 Movement of the sun

Because the earth makes a full rotation in 24 hours, the sun moves 0.00013 degrees per frame. This value is so small and was practically negligible.

As we mentioned above, the incompleteness of the features as a point at infinity is insignificant, as long as we consider the relationship of the camera movement between the consecutive two frames.

However, this incompleteness is significant if we consider the relationship between discontiguous frames such as the first and *3,000-th* frames, where the parallax is not negligible because translation of the helicopter is long, the clouds shape changes, and the sun moves. Therefore, we propose the following algorithm to stabilize the images by taking into account these problems.

7. Procedures and Algorithm

7.1 Outline of the Procedure

An outline of the stabilizing procedure is as follows:

Step 1. Select the features in the scene.
Step 2. Track the motion of the features.
Step 3. Estimate the rotation between two successive frames.
Step 4. Calculate the absolute rotation of each frame.
Step 5 Estimate the global rotation.
Step 6. Calculate the adjustment rotation.
Step 7. Make 3-D rotation of the image in each frame.

7.2 Rotation Estimation Method

The rotation estimation in Step 3 is calculated as follows: $\boldsymbol{m}_i^{(k)}$ denotes the image position of feature-*i* in the *k-th* frame. $\boldsymbol{m}_i^{(k+1)}$ denotes the image position of the same feature in the (*k*+1)-*th* frame. When $\boldsymbol{m}_i^{(k)}$ and $\boldsymbol{m}_i^{(k+1)}$ (*i = 1, 2, ..., N*) are given, the relative camera rotation $\boldsymbol{R}$ between the *k-th* image and (*k*+1)-*th* image can be calculated by using equation (2) as

$$\sum_{i=1}^{N}\left(\boldsymbol{m}_i^{(k+1)} - \boldsymbol{R}^{(k)}\boldsymbol{m}_i^{(k)}\right)^2 \leftarrow \min . \qquad (3)$$

The solution to equation (3) is obtained [1][10] as $\boldsymbol{R}^{(k)} = \boldsymbol{V}\boldsymbol{U}^t$, where $\boldsymbol{V}$ and $\boldsymbol{U}$ are orthogonal matrices obtained by singular value decomposition on matrix $\boldsymbol{M}^{(k)}$ as $\boldsymbol{M}^{(k)} = \boldsymbol{V}\Lambda\boldsymbol{U}^t$, where $\boldsymbol{M}^{(k)}$ is computed as $\boldsymbol{M}^{(k)} = \sum_{i=1}^{N}\boldsymbol{m}_i^{(k+1)}\{\boldsymbol{m}_i^{(k)}\}^t$.

7.3 Algorithm to Stabilize

In this subsection, we detail Step 3 to 6 in the procedures. In our explanation, we denote the combination of rotations $\boldsymbol{R}_1$ and $\boldsymbol{R}_2$ as $R_{12} = R_1 + R_2$, rather than as $\boldsymbol{R}_{12} = \boldsymbol{R}_2\boldsymbol{R}_1$, because it is simpler to explain. Of course, in the actual computation, we calculated a proper rotation for this notation.

7.3.1 Estimate the rotation

Figure 6 illustrates the rotational movement of the helicopter and camera; where t denotes the frame number. The heavy-dotted line $R^{(base)}(t)$ denotes the rotational factor of the global movement of the helicopter, assuming that $R^{(base)}(0)$ is the base. The helicopter was navigated according to this line. If the helicopter and the mounted camera can move without any jolting, the images of each frame can also be captured according to this line.

In practice, however, much jolting occurred, and we illustrated these jolting in each frame with heavy-arrowhead $E(t)$. According to the jolting, the original images captured by the ideal camera were distorted, as the white dot $R^{(actual)}(t)$ shows.

In this step, relative rotation $\boldsymbol{R}^{(k,k+1)}$ between the *k-th* and (*k*+1)-*th* frames is estimated. We illustrated it as a fine-arrowhead $\Delta R(k+1)$.

7.3.2 Calculate the absolute rotation

Estimation error is unavoidable using the process described earlier. The main reasons are: (1) the features used in calculating are incomplete as a point at infinity, and they may be movable, and (2) during feature tracking, there was the some measurement error. Therefore, we define the relationship between the true value of $\Delta R(k)$ and its calculated value $\Delta R'(k)$, as

$$\Delta R'(k) = \Delta R(k) + \delta_k + \varepsilon_k , \qquad (4)$$

where δ_k is error caused by the incompleteness of the features, and ε_k is error caused by other problems, such as measurement error. However, we can assume that kinds of errors are very small compared with $\Delta R(k)$.

We can calculate the absolute direction of each frame image, namely, the relative rotation between the *0-th* and *t-th* frames, by using $R^{(actual)}(t) = \sum_{k=0}^{t}\Delta R(k)$. However, because the value we use is $\Delta R'(k)$ rather than $\Delta R(k)$, it has to be calculated by $R^{(calc)}(t) = \sum_{k=1}^{t}\Delta R'(k)$. Using equation (4), we can rewrite this equation as

$$R^{(calc)}(1) = R^{(actual)}(1) + \delta_{(1)} + \varepsilon_{(1)},$$
$$R^{(calc)}(2) = R^{(actual)}(2) + \delta_{(1)} + \delta_{(2)} + \varepsilon_{(1)} + \varepsilon_{(2)}$$
$$\vdots$$
$$R^{(calc)}(t) = R^{(actual)}(t) + \sum_{k=1}^{t}\delta_{(k)} + \sum_{k=1}^{t}\varepsilon_{(k)} . \quad (5)$$

Figure 7 shows the relationship between the true rotation $R^{(actual)}(t)$, and the calculated rotation $R^{(calc)}(t)$. As the figure and equation (5) show, the deviation is cumulated according to the proceeding frame, even though each δ_k and ε_k are small.

7.3.3. Estimate of the global movement

The obtained value thus far is only $R^{(calc)}(t)$. If we rotate all the images so that the *k-th* frame image rotates $\{R^{(calc)}(k)\}^{-t}$, all of them orient in the same direction as the *0-th* frame, and thus the stabilization is established.

However, as previously stated, we have to accept that $R^{(calc)}(t)$ includes a high degree of error. This error does not cause the stabilization problems, because relative movement among neighboring frames is only important for smoothness. Application of rotation $\{R^{(calc)}(k)\}^{-1}$ to all frames is equivalent to application of

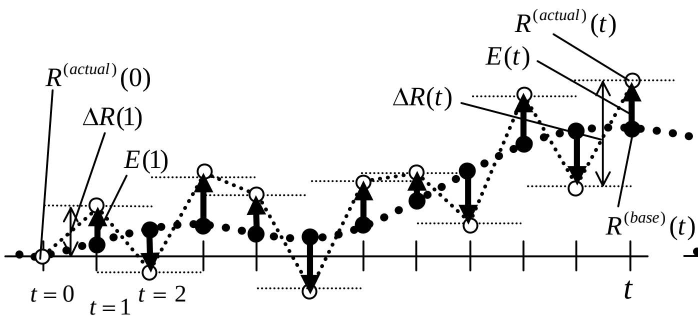


Figure 6. Rotational movement of the mounted camera. The black dots represent each frame's direction when no jolting occurred. The white dots represent each frame's direction under actual conditions (with jolting).

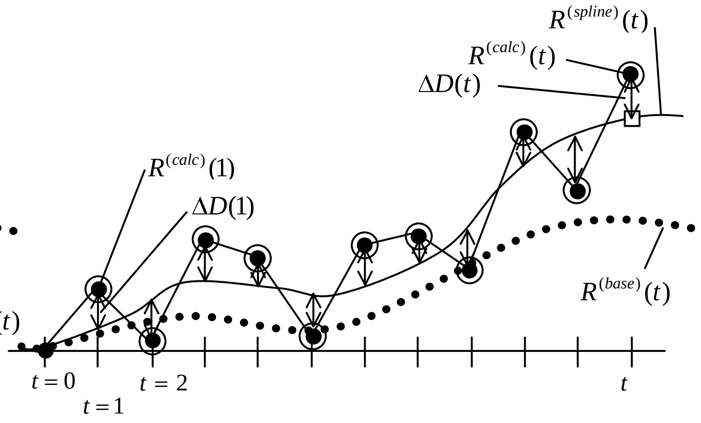


Figure 8. Estimation of the global movement (rotational factor) of the helicopter. The heavy-dotted line reflects actual movement. The fine line indicates estimated movement. The deviation $\Delta D(t)$ between the estimated movement (fine line) and calculated absolute rotation (black dot) is computed.

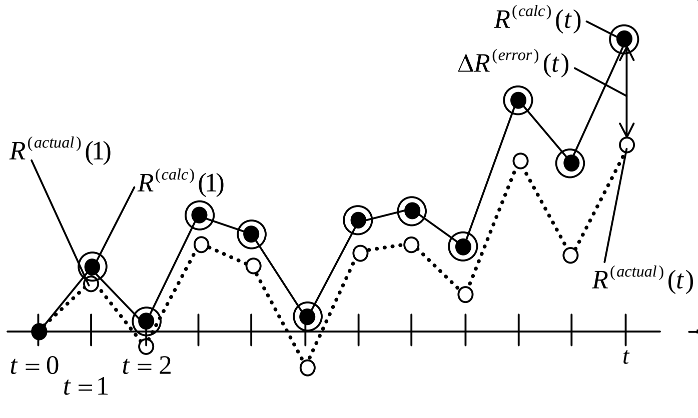


Figure 7. Relationship between true rotation (white-dot) and calculated rotation (black-dot). The deviation increases cumulatively.

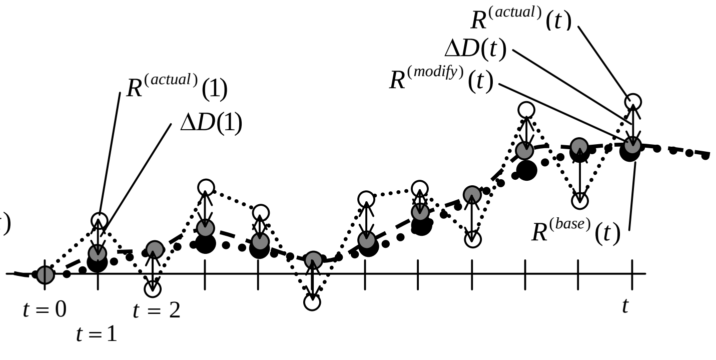


Figure 9. Relationship between the direction of the actual image (white dot) and the direction of the modified image (gray dot). We cannot know the value $R^{(modify)}(t)$, nor $R^{(base)}(t)$, but the modified image can be obtained so that the two values are approximately the same.

relative rotation $\{\Delta R'(j)\}^{-1}$ between the successive frames cumulatively (i.e., the *j-th* and (*j+1*)-*th* frames; *j* = 0,1,2, ..., *k*). In this process, the value $\{\Delta R'(j)\}^{-1}$ has a small amount of error, as shown in equation (4); therefore the stabilization is well established.

But, there is another problem. If we rotate $\{R^{(calc)}(k)\}^{-1}$, each frame image orients in markedly different directions as they were actually shot. This generates not only a problem that the display image does not orient the direction expected by the photographer, but also cases in which a non-picturized area has to be displayed. To avoid these problems, the global movement of the camera has to be estimated, but this estimation is actually impossible. So we will explain how to modify this problem without directly estimating it.

First, we smooth the discrete value $R^{(calc)}(t)$ (t = 0,1,2, ...) by using spline functions, so that the sum of squares of deviation between $R^{(calc)}(t)$ and the spline curve becomes minimum. In our application, we used a cubic spline, and determined the number of knots based on the number of frames. Figure 8 illustrates this function as $R^{(spline)}(t)$. Note that the function is far from the actual value $R^{(base)}(t)$, because the data used to make it includes a large amount of error.

Next, we calculate a difference $\Delta D(t)$ between the discrete value $R^{(calc)}(t)$ and its smoothed function $R^{(spline)}(t)$, and this value $\Delta D(t)$ is used in the next procedure.

7.3.4 Image rotation

Finally, we rotate $\{\Delta D(k)\}^{-1}$ each frame image (*k* = 1,2,3, ...). Figure 9 illustrates the results. The white dot $R^{(actual)}(t)$ denotes the direction of the given image. The gray dot $R^{(modify)}(t)$ denotes the direction of the modified image rotated by $\{\Delta D(k)\}^{-1}$.

Even though the value of $R^{(actual)}(t)$ (white dot) and the value of $R^{(modify)}(t)$ (gray dot) are neither known nor estimated in the above process, the line determined by the gray dots (fine-dotted line) is similar to the line (heavy-dotted line) that represents the actual global movement of the helicopter. Thus, we can improve the problems mentioned in 7.3.3.

7.4 Implementation and Consideration

According to the principle described above, we stabilized an actual image. We implemented the stabilizing function as a plug-in software on a non-linear digital film making system. In this section, we refer only to key points.

(1) Number of feature points: We selected two mountains, two clouds, and the shadow of the helicopter as features (Figure 10). However, in the frame where the shadow of helicopter could not be identified clearly, we avoided using this information.

(2) Motion tracking: Motion tracking was executed by template matching. We used the *k-th* frame's image as a template when we searched for the (*k+1*)-*th* frame's image, and it was always updated.

(3) Treatment of rotation matrix: In making the splines, we used the rotation matrix as a parametric expression. For detail, we expressed the rotation matrix as three parameters --row, yaw, and pitch-- and made three splines as a function of t.

(4) Image development: After adjusting the rotation so as to stabilize the camera, the fish-eye image was converted into a multi-screen image. Because our multi-screen environment has a 270° perspective ($90° \times 3$; left, front, and the right), but the fish-eye had only $180°$, so we developed the fish-eye image by stretching it to $300°$ view-angle non-linearly, so that the front image had less distortion than the images on both sides, because the viewer usually pays attention mostly to the front screen. Since there was an additional margin of $300° - 270° = 30°$ ($\pm 15°$), we could generate the entire image without any black area (non picturized area) if the rotational adjustment was less than $15°$. Figure 11 shows an example of 3-D rotation using a fish-eye format. The borders of each screen before and after the rotation are shown.

8. Conclusion

We described a practical method for stabilizing an image sequence captured by a camera mounted on a helicopter.

We demonstrated that 3-D rotation is necessary and performs adequately if we consider the properties of our shooting method, shooting object, and display environment. We also demonstrated that a camera-pose estimation uses vanishing points was more effective and practical for our purpose than using general points. However, no information about the vanishing point could be found, so we selected other features that have similar properties as a point of infinity. What we selected was not complete, but it was adequate for our purpose. During the actual implementation, a problem occurred with the orientation of the display image; it was sometimes far from the intended orientation. We proposed preventing this problem by using a smoothing function. The function were not used to improve stability, but were used solely to adjust the global movement of the camera. By using this method, we obtained the desired image quality, even though we could not estimate correct values of the rotation.

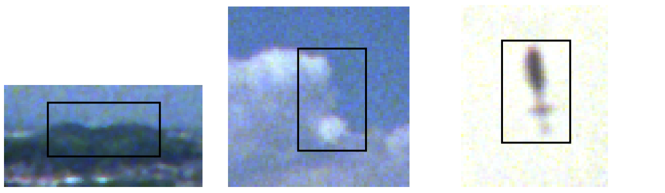


Figure 10. Templates of the features used to track the motion.

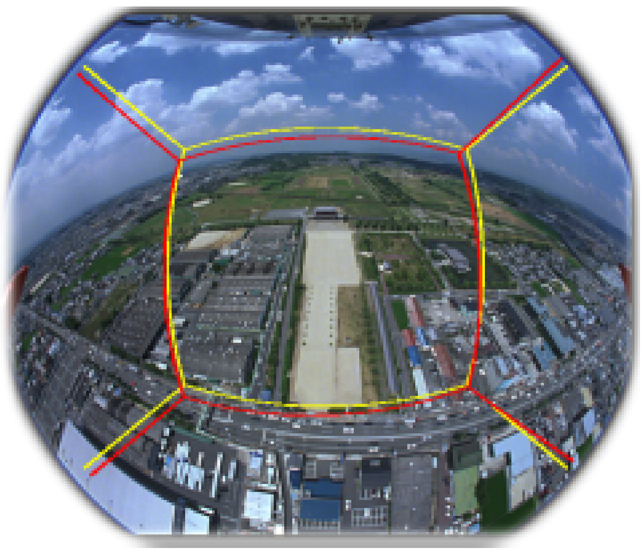


Figure 11. Example of 3-D rotation of fish-eye image. Red lines indicate borders of multi-screen in the previous frame's image. Yellow lines indicate these borders after appropriate rotation for stabilization.

References

[1] K. S. Arun, T. S. Huang and S. D. Blostein, "Least-squares fitting of two 3-D point sets," IEEE Trans. PAMI, Vol.9, 1987.

[2] B.Caprile and V.Torre, "Using Vanishing Points for Camera Calibration," International Journal of Computer Vision, No. 4, 1990.

[3] C. Cruz-Neira, D. J. Sandn, T. A. Defanti, "Surround-screen projection-based virtual reality: the design and implementation of the CAVE," ACM Proc. SIGGRAPH, 1993.

[4] O. D. Faugeras, Three-dimensional computer vision, MIT press, 1993.

[5] O. D. Faugeras and S. J. Maybank, "Motion from point matches: Multiplicity of Solutions," International Journal of Computer Vision, Vol. 4, 1988.

[6] M. Hansen, P. Andan, K. Dana, G. Wal and P. Burt, "Real-time scene stabilization and mosaic construction," IEEE Proc. CVPR, 1994.

[7] R. I. Hartley, "In defence of the 8-point algorithm," IEEE Proc. ICCV, 1995.

[8] R. Hartley, "A linear method for reconstruction from lines and points," IEEE Proc. ICCV '95, 1995.

[9] M. Irani, B. Rousso, and S. Peleg, "Recovery of ego-motion using image stabilization," IEEE Proc. CVPR, 1994.

[10] K. Kanatani, Geometric Computation for Machine Vision, Oxford Science Publications, 1993.

[11] C. Morimoto and R. Chellappa, "Fast electronic digital image stabilization for off-road navigation," Real Time Image, No. 5, Oct., 1996.

[12] T. Moriya and H. Takeda, "Image generation for immersive multi-screen environment with a motion ride," IEEE Proc. VR2001, 2001.

[13] R. Szeliski and H. Y. Shum, "Creating full view panoramic image mosaics and environment maps," ACM Proc. SIGGRAPH, 1997.

[14] B. Triggs, "Factorization methods for projective structure and motion," IEEE Proc. CVPR '96, 1996.

[15] Y. Xiong and K. Turkowski, "Creating image-based VR using a self-calibrating fisheye lens," IEEE Proc. CVPR '97, 1997.

[16] Z.Zhu, G.Xu and X.Lin, "Constructing 3D Natural Scene from Video Sequences with Vibrated Motions," IEEE Proc. VRAIS '98, 1998.