Camera Stabilization Based on 2.5D Motion Estimation and Inertial Motion Filtering

Zhigang Zhu[1*], Guangyou Xu[1], Yudong Yang[1], Jesse S. Jin[2]

[1]Department of Computer Science and Technology, Tsinghua University
Beijing, 100084, China. Email: {zhuzhg, xgy-dcs}@mail.tsinghua.edu.cn
[2]School of Computer Science and Engineering, The University of New South Wales
Sydney 2052, Australia. Email: jesse@cse.unsw.edu.au

Abstract -This paper presents a novel approach to stabilize video sequences digitally. A 2.5D inter-frame motion model is proposed so that the stabilization system can work in situations where significant depth changes are present and the camera has both rotational and translational movements. An inertial model for motion filtering is proposed in order to eliminate the vibration of the video sequences and to achieve good perceptual properties. The implementation of this new approach integrates four modules: pyramid-based motion detection, motion identification and 2.5D motion parameter estimation, inertial motion filtering, and affine-based motion compensation. The stabilization system can smooth unwanted vibrations of video sequences in real-time. We test the system on IBM PC compatible machines and the experimental results show that our algorithm outperforms many algorithms that require parallel pipeline image processing machines.

Keywords: tele-operation, video stabilization, motion estimation, motion filtering, inertial model.

I. INTRODUCTION

The goal of camera stabilization is to remove unwanted motions from a dynamic video sequence. It plays an important role in many vision tasks such as tele-operation, robot navigation, ego-motion recovery, scene modeling, video compression and detection of independent moving objects [1-6,9-10,13]. There are two critical issues that decide the performance of a digital video stabilization system. One is inter-frame motion detection. The other is motion filtering.

The first issue depends on what kind of motion model is selected and how image motion is detected. It has been pointed out [1,5] that human observers are more sensitive to rotational vibrations, and only camera rotation can be compensated without the necessity of recovering depth information. Hansen *et al*.[2] used an affine model to estimate the motion of an image sequence. The model results in large errors for rotational estimations if there are large depth variations within images and the translation of the camera is not very small. Duric *et al*.[5] and Yao *et al*.[6] dealt with this problem by detecting faraway-horizon lines in images since the motion of a horizon line is not affected by the small translation of the video camera. They assumed that long straight horizon lines would exist in common out-door images and it will have large gradients in gray scale images. This assumption works only if there is a horizontal line and the points around that line are far away. It will fail in many cases where the horizon line is not very clear or just cannot be seen, or the points along the horizon line are not so far away.

The second issue is critical for eliminating unwanted vibrations while preserving the smooth motion. The algorithm in Hansen *et al*.[2] chose a reference frame at first, and then aligned the successive frames to that frame by warping them according to the estimated inter-frame motion parameters. The reconstructed mosaic image was displayed as the result of stabilization. This approach works fine if there are no significant depth changes in the field of view and/or the moving distance of the camera is not long. However, it will fail in cases involving panning or tracking motion (e.g. Fig. 2), where accumulated errors will override the real inter-frame motion parameters. Duric *et al*.[5] assumed that camera carriers would move in a steady way between deliberate turns. Thus they tried to stabilize the video sequence by fitting the curves of motion parameters with linear segments. The resulting

[*] The author is currently on a leave in the Computer Science Department, University of Massachusetts at Amherst, MA 01003. Email: zhu@cs.umass.edu, or zhuzhg@mail.tsinghua.edu.cn. This work was supported by China Advanced Research Project during 1990-1996 and by China National Science Foundation during 6/1995-9/1997.

sequences would be steady during each line segment but the moving speed would change suddenly between two segments. Another problem in these approaches is that image sequence is delayed for several frames in order to get a better line fitting. Yao *et al.*[6] proposed a kinetic vehicle model for separating residual oscillation and smooth motion of the vehicle (and the camera). This model works well when kinetic parameters of the camera's carrier, such as vehicle's mass, the characteristics of the suspension system, the distances between front and rear ends, and the center of gravity, are known and the camera is bound to the vehicle firmly. However, in most cases, either we do not know these parameters, or the camera carrier is not a grounded vehicle and the camera is not firmly bound to the carrier.

Real-time performance is also a basic requirement for video stabilizers. There are many researches on video stabilization for military and tele-operation utilities as well as applications in commonly known commercial camcorders. Hansen *et al.*[2] implemented an image stabilization system on a parallel pyramidal hardware (VFE-100), which is capable of estimating motion flow in a coarse-to-fine multi-resolution way. The motion parameters were computed by fitting an affine motion model to the motion flow. The stabilized output was given in the form of a registration-based mosaic of input images. Their system stabilized 128×120 images at 10 frames per second with inter-frame displacement up to ±32 pixels. Morimoto *et al.* [3] gave an implementation of a similar method on Datacube Max Video 200, a parallel pipeline image processing machine. Their prototype system achieves 10.5 frames per second with image size 128×120 and maximum displacement up to ±35 pixels. Zhu *et al.*[1] implemented a fast stabilization algorithm on a parallel pipeline image processing machine called PIPE. The whole image was divided into four regions and several feature points were selected within these regions. Each selected point was searched in the successive frame in order to find a best match using correlation method. The detected motion vectors were combined in each region to obtain a corresponding translation vector. A median filter was used to fuse four motion vectors into a plausible motion vector for the entire image. A low-pass filter was then used to remove the high frequency vibrations of the sequence of motion parameters. The system runs at 30 frames per second with image size 256×256 pixels and can handle maximum displacement from -8 to 7 pixels.

In this paper we proposed a new digital video stabilization approach based on a 2.5D motion model and an inertial filtering model. Our system achieves 26.3 frames per second (fps) on an IBM PC compatible using an Intel Pentium 133MHz microprocessor without any image processing accelerating hardware. The image size is 128×120 and maximum displacement goes up to ±32 pixels. The frame rate reaches 33 fps by using Intel PII/233MHz with MMX when image size is 360×270 with maximum inter-frame displacement up to ±64 pixels.

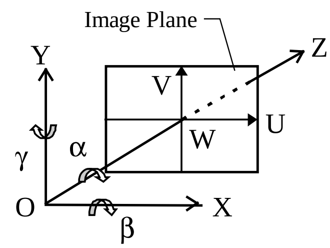


Fig. 1. Coordinate systems of the camera and the image

II. INTER-FRAME MOTION MODEL

In the rest of our discussion, we assume the scene is static and all motions in the image are caused by the movement of the camera. A reference coordinate system *O-XYZ* is defined for the moving camera, and the optical center of the camera is *O* (see Fig. 1). *W-UV* is the image coordinate system, which is the projection of *O-XYZ* onto the image plane. The camera motion has 6 degrees of freedom: three translation components and three rotation components. In other words, it can be interpreted as that the scene being viewed has 6 motion parameters since we use the camera as the reference. Considering only an inter-frame case, we represent three rotational angles (roll, pitch and yaw) by (α, β, γ) and three translation components as (T_x, T_y, T_z). A space point (x, y, z) with image coordinate (u, v) will move to (x', y', z') in the next time with the image point moving to (u', v'). Suppose the camera focal length f is f' after the motion. Under a pin hole camera model, the relation between these coordinates are[1]

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} a & b & c \\ i & j & k \\ l & m & n \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} - \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix}$$

and

$$\begin{cases} u' = f' \dfrac{au + bv + cf - fT_x / z}{lu + mv + nf - fT_z / z} \\ v' = f' \dfrac{iu + jv + kf - fT_y / z}{lu + mv + nf - fT_z / z} \end{cases} \tag{1}$$

If the rotation angle is small, e.g., less than 5 degrees, Eq. (1) can be approximated as

$$\begin{cases} u' \approx f' \dfrac{u + \alpha v - \gamma f - fT_x / z}{\gamma u - \beta v + f - fT_z / z} \\ v' \approx f' \dfrac{-\alpha u + v + \beta f - fT_y / z}{\gamma u - \beta v + f - fT_z / z} \end{cases} \tag{2}$$

Let

$$s = (\gamma u - \beta v + f - fT_z / z) / f' \tag{3a}$$

we will have

$$\begin{cases} s \cdot u' = u + \alpha v - \gamma f - fT_x / z \\ s \cdot v' = -\alpha u + v + \beta f - fT_y / z \end{cases} \quad (3)$$

A. Affine motion model

From Eqs. (2) and (3) we have the following observations:

<1> *Pure rotation*. If the translations of the camera are zero, the images will only be affected by rotation and the effect is independent of the depths of scene points. When the rotation angle is very small (e.g., less than 5 degrees), camera rolling is the main factor of image rotation, and the effect of the pitch and yaw can be approximated by 2D translations of the image.

<2> *Small Translation and constrained scene*. If image points are of the same depth, or the depth differences among image points are much less than the average depth, small camera translation will mainly cause homogeneous scaling and translation of 2D images.

<3> *Zooming* . Changing camera focal length will only affect the scale of images.

<4> *General motion*. If depths of image points vary significantly, even small translations of the camera will make different image points move in quite different ways.

For cases <1>,<2> and <3>, a 2D affine (rigid) inter-frame motion model can be used.

$$\begin{cases} s \cdot u' = u + \alpha v - T_u \\ s \cdot v' = v - \alpha u - T_v \end{cases} \quad (4)$$

where s is a scale factor, (T_u, T_v) is the translation vector, and α is the rotation angle. Given more than 3 pairs of corresponding points between two frames, we can obtain the least square solution of motion parameters in Eq. (4). Since this model is simple and works in many situations, it has been widely adapted [1-6].

B. The 2.5D motion model

Although the affine motion model in the previous section is simple in calculation, it is not appropriate for the general case (<4>), where depths of scene points change significantly and the camera movement is not very small. It is impossible to estimate motion without the depth information. However, recovering surface depth in a 3D model is too complicated and depth perception itself is a difficult research topic. We aim at a compromise solution involving depth information at estimation points only.

Careful analysis of Eqs. (1), (2) and (3) reveals the following results.

(1) *Dolly movement*. If the camera moves mainly in Z direction, the scale parameters of different image points depends on their depths (Eq.(3a)).

(2) *Tracking movement*. If the camera moves mainly in X direction, different image points will have different horizontal translation parameters related to their depths. Similarly, if the camera moves mainly in Y direction, different image points will have different vertical translation parameters corresponding to their depths (Eq.(3)).

(3) *Rotational/zooming movement*. Panning, tilting, rolling, zooming and their combinations generate homogeneous image motion (Eq.(3)).

(4) *Translational movement*. If camera does not rotate, we have a direct solution of relative camera translation parameters and scene depths (Eq.(1) or (2)).

(5) *General movement*. If camera moves in arbitrary directions, there is no direct linear solution of these motion parameters. This is the classic "structure from motion (SfM)" problem [7,8,9] and will not be addressed here.

Cases (1) (2) and (3) cover the most common camera motions: dolly (facing front), tracking (facing side), panning/tilting/rolling (rotation), and zooming movement. We proposed a 2.5D inter-frame motion model by introducing a depth related parameter for each point. It is an alternative between 2D affine motion and the real 3D model.

The model consists of three sub-models: dolly, H-tracking and V-tracking models. For dolly movement (involving rotation and zooming, i.e. case (1) with case (3)), we have $T_x \approx 0$, $T_y \approx 0$, hence the 2.5D dolly model is

$$\begin{cases} s_i \cdot u_i' = u_i + \alpha v_i - T_u \\ s_i \cdot v_i' = v_i - \alpha u_i - T_v \end{cases} \quad (i=1..N) \quad (5)$$

For horizontal tracking movement (involving panning, rolling and zooming), we have $T_z \approx 0$, $T_y \approx 0$. The 2.5D H-tracking model is

$$\begin{cases} s \cdot u_i' = u_i + \alpha v_i - T_{ui} \\ s \cdot v_i' = v_i + \alpha u_i - T_v \end{cases} \quad (i=1..N) \quad (6a)$$

Similarly, for vertical tracking movement (involving tilting, rolling and zooming), we have $T_z \approx 0$, $T_x \approx 0$. The 2.5D V-tracking model is

$$\begin{cases} s \cdot u_i' = u_i + \alpha v_i - T_u \\ s \cdot v_i' = v_i + \alpha u_i - T_{vi} \end{cases} \quad (i=1..N) \quad (6b)$$

In most situations, the dominant motion of the camera satisfies one of the 2.5D motion models. It should be noticed that in cases (5), (6a) and (6b), there are N+3 variables in 2N equations. Different N variables contribute to the solution in each case, i.e. s_i for depth-related scale,

T_{ui} for depth-related horizontal translation and T_{vi} for depth-related vertical translation, respectively.

III. Motion Detection and Classification

A. A numerical method for parameter estimations

If there are more than three ($N \geq 3$) correspondences between two frames, the least square solution can be obtained for any of three cases mentioned above. Since the forms of equations are quite similar in three cases, we take the 2.5D dolly motion model as an example. By dedicatedly arranging the order of the $2N$ equations, we have the following matrix from Eq. (5):

$$\mathbf{Ax} = \mathbf{b} \tag{7}$$

where

$$\mathbf{A} = \left[\begin{array}{cccc|ccc} u_1 & 0 & \dots & 0 & -v_1 & 1 & 0 \\ 0 & u_2 & \dots & 0 & -v_2 & 1 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & u_N & -v_N & 1 & 0 \\ v_1 & 0 & \dots & 0 & u_1 & 0 & 1 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & v_N & u_N & 0 & 1 \end{array}\right] \quad \mathbf{x} = \begin{bmatrix} s_1 \\ s_2 \\ \dots \\ s_N \\ \alpha \\ T_u \\ T_v \end{bmatrix} \quad \mathbf{b} = \begin{bmatrix} u_1 \\ u_2 \\ \dots \\ u_N \\ v_1 \\ \dots \\ v_N \end{bmatrix}$$

We can solve Eq.(7) by a QR decomposition using the Householder transformation. Because of the special form of Matrix $\boldsymbol{A}$, the Householder transformation can be realized efficiently: the first N columns only require $(N+1)\times 8$ multiplies and N square root operations; then the Householder transformation is only performed on an $N\times 3$ dense submatrix. $\boldsymbol{A}$ can be stored in a $2N\times 4$ array. In fact this approach requires less computation than that of 2D affine model with a $2N\times 4$ dense matrix. We have also found that this method is numerically stable, since in most cases Eq. (7) has a condition number less than 500. Notice that the condition number of the equation for 2D affine model is between 150 and 300. A detailed discussion and numerical testing result of this model can be found in [12].

B. Motion detection and classification

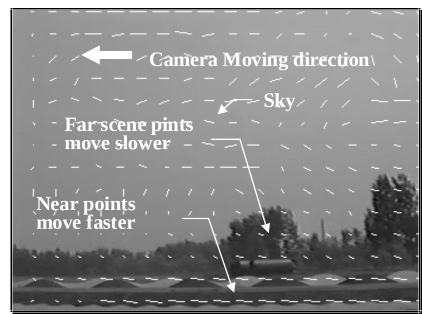
The detection of the image velocities is the first step in motion estimation. Instead of building a registration map between two frames for distinguished features, like in [6], the image velocities of the representative points are estimated by a pyramid-based matching algorithm applied to the original gray-scale images. The algorithm consists of the following steps:

(1) building the pyramids for two consecutive images;

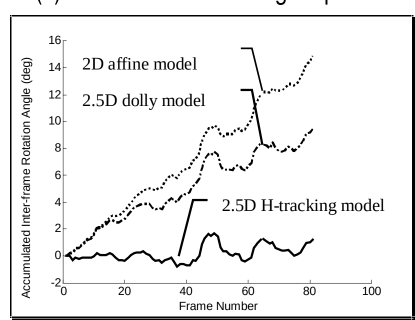
(2) finding the image velocity for each small block (e.g. 8×8, 16×16) in the first frame by using a coarse-to-fine correlation matching strategy; and

(3) calculating the belief value of each match by combing the texture measurement and the correlation measurements of the block. This step is important because the value is used as a weight in the motion parameter estimation.

Fig. 2 shows the result of rotation angle estimation for one of the testing sequences, where *2D affine model*, *2.5D dolly model* and *2.5D H-tracking model* of our approach are used for comparisons. The original sequence is taken by a camera mounted on a moving-forward vehicle and looked at the side of the vehicle. The vibration is mainly camera rolling. The accumulated rolling angle (global rotation) is approximately zero. The motion satisfies the horizontal tracking model. Motion vectors in Fig. 2(a) shows that image velocities are quite different due to the different depths of the scenes. The curves in Fig. 2(b) are the accumulated inter-frame rotation angles estimated by three models respectively. False global rotation appears in the affine model.



(a) One frame of a testing sequence



(b) Accumulated inter-frame rotation angle

Fig. 2. Estimation errors using different motion models

An important issue in our 2.5D motion model is the correct selection of the motion model (dolly, H-tracking or V-tracking). False global rotation is also derived if an improper motion model is applied, as shown in Fig 2(b) where the dolly model is a wrong representation for the type of the movement. The classification of the image motion is vital. In the simple way, the motion model can be selected by a human operator since the qualitative model class can be easily identified by a human observer, and the type of the motion within a camera snap usually lasts for certain time period. The selection of motion models can also be made automatically by the computer.

Careful investigations of the different cases of camera movements indicate that the motion can be classified by the patterns of image flows[14], or the accumulated flows. Dolly (and/or zooming), H-tracking (and/or panning), V-tracking (and/or tilting) and rolling movements result in four distinctive flow patterns. It is possible to classify them automatically although the discussion will go beyond the scope of this paper.

IV. MOTION FILTERING

The second vital stage of video image stabilization is the elimination of unwanted high frequency vibrations from the detected motion parameters. The differences between smoothed motion parameters and original estimated motion parameters can be used to compensate the images in order to obtain the stabilized sequence. Removing vibrations (high frequency) is virtually a temporal signal filtering (motion filtering) process with the following special requirements.

- The smoothed parameters should not be biased from the real gentle changes. Bias like phase shift, amplification attenuation may cause undesired results.
- The smoothed parameters should comply with the physical regularities. Otherwise, the result will not satisfy human observers.

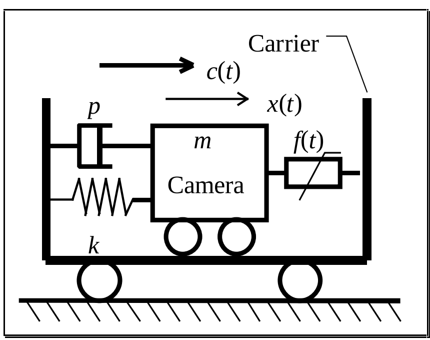


Fig. 3. Inertial motion filtering

Based on the above considerations, we proposed a generic inertial motion model as the motion filtering model. This filter fits physical regularities well and has little phase shift in a large frequency range. The only requirement is to set some assumptions of sequence's dynamics. In the context of image stabilization, we should make it clear that which kind smooth motion needs to be preserved, and what vibration (fluctuation) needs to be eliminated. There are some differences for three kinds of motions in our model.

For the dolly movement, the smooth motion is mainly translation along the optical direction, possibly with deliberate zooming and rotation of 1 to 3 degrees of freedom. The fluctuations are mainly included in three motion parameters: α, T_u and T_v (Eq.(5)), which may be caused by bumping and rolling of the carrier (vehicle). The depth-related scale parameter s_i for each image block is related to the zooming and dolly movement and the depths of the scene. In order to separate the depth information from the motion parameter, we calculate the fourth motion parameter by

$$s = \sum_{i=1}^{N} s_i \tag{8}$$

The fluctuation caused by dolly and zooming will be extracted from this parameter.

For the H-tracking movement, the smooth translational component is included in parameter T_{ui}, which is depth related. Apart from three affine parameters s, α and T_v, we separate the depth information by averaging the horizontal components of all image blocks and obtain the fourth motion parameters by

$$T_u = \sum_{i=1}^{N} T_{u_i} \tag{9}$$

A more detailed discussion can be found in [13]. Similarly four motion parameters s, α, T_u and T_v can be obtained in the case of V-tacking movement.

Since four motion parameters are independent of each other, we can treat them separately. For example, Fig. 3 is the schematic diagram of an inertial filter where $c(t)$ is carrier's displacement, $x(t)$ is camera's displacement, m is the mass of the camera, k is elasticity of a spring, p is the damping parameter, and $f(t)$ is a controllable feedback compensator. Assuming $f(t) = 0$, the kinematic equation can be written as

$$m\frac{d^2x}{dt^2} + p\left(\frac{dx}{dt} - \frac{dc}{dt}\right) + k(x-c) = 0 \tag{10}$$

In fact, it can be considered as a forced oscillation device. The intrinsic oscillation frequency is $f_n = \frac{1}{2\pi}\sqrt{\frac{k}{m}}$, the damping ratio is $\xi = \frac{p}{2\sqrt{mk}}$, the relative frequency is $\lambda = \frac{f}{f_n}$. The frequency response can be expressed as

$$H(\lambda) = \sqrt{\frac{1+4\xi^2\lambda^2}{(1-\lambda^2)^2 + 4\xi^2\lambda^2}}\,. \tag{11}$$

When sampling rate (i.e. frame rate) is much higher than f_n, we can directly write the discrete form of Eq. (10) as a differential equation

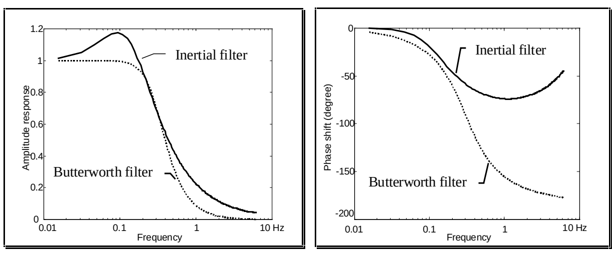
$$x(n) = a_1 x(n-1) + a_2 x(n-2) + b_1 c(n) + b_2 c(n-1) + f(n) \tag{12}$$

with

$$a_1 = \frac{2m + pT}{m + pT + kT^2} \qquad a_2 = \frac{-m}{m + pT + kT^2}$$

$$b_1 = \frac{pT + kT^2}{m + pT + kT^2} \quad b_2 = \frac{-pT}{m + pT + kT^2}.$$

and $f(n)$ is the feedback. This is a 2nd order digital filter. By choosing suitable values for these parameters, high frequency vibrations can be eliminated. For example, if frame frequency is 25Hz, and high frequency vibrations have frequency f_v larger than 0.5Hz, we can choose the termination frequency $f_T = 0.3$Hz. Assuming that $k = 1$, $\xi = 0.9$, we have $f_n = 0.1288$, $m = 1.5267$, and $p = 2.2241$. Compared to the 2nd order Butterworth low-pass filter (LPF) that has $f_T = 0.3$Hz, the inertial filter has a similar terminating attribute but with much less phase delay. Fig. 4 shows the comparison results.



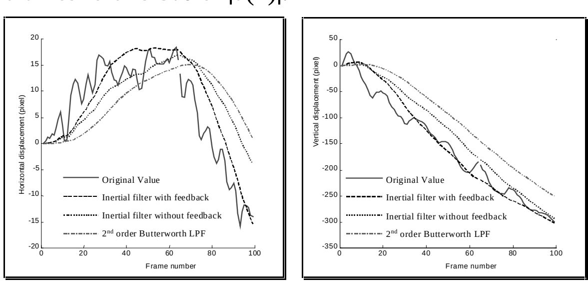
(a) Amplitude response (b) Phase response

Fig. 4. Frequency response of inertial filter and Butterworth LPF

For a better tuning in filter response and reduction of phase shift, proper feedback compensations are added. Negative feedback can reduce phase shift but will increase the bandwidth of the filter. An effective way is to use threshold-controlled feedback: if $|x(n)-c(n)|$ is greater than a predefined value, the feedback is turned on till the error value is less than a threshold. Fig. 5 is the experimental results on real video sequence images. The feedback function is

$$f(n) = -0.005\ |x(n)-c(n)|$$

and threshold is 8% of $|c(n)|$.



(a) Horizontal displacement (b) Vertical displacement

Fig. 5. Comparison of different filtering methods

V. EXPERIMENTAL RESULTS

We have implemented a stabilization system based on the 2.5D motion model and inertial motion filtering. Our system consists of four modules as shown in Fig. 6. Feature matching is performed by multi-resolution matching of small image blocks. Reliability of each matched block is evaluated by accumulating gray scale gradient values and the sharpness of a correlation peak because areas without strong texture (such as sky, wall and road surface) may have large matching errors. The evaluation value of each image block is use as a weight of a corresponding equation pair (e.g. Eq.(5)) when they are used in solving inter-frame motion parameters. Calculated inter-frame motion parameters are then accumulated as global motion parameters related to the reference frame and they are passed through the inertial filter. The differences of filtered motion parameters and original ones are used to warp current input frame to obtain the stabilized output.

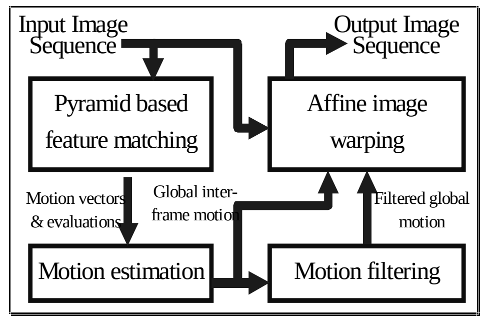


Fig. 6. Diagram of the stabilization system

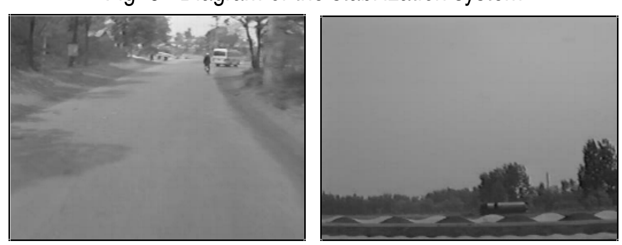


Fig. 7. Key frames of testing sequences (a) Dolly movement with severe vibration; (b) Tracking movement with rolling vibration

Table 1 Execution time [12] of each stage

CPU	image (pixels)	block (pxls)	match (ms)	motion (ms)	warp (ms)	rate (f/s)
486/66	128×120	16×16	95	4	54	6.54
486/66	360×270	20×20	468	17	333	1.22
P5/133	128×120	16×16	25	1	12	26.32
P5/133	360×270	20×20	122	3	77	4.95
PII/233	384×288	16×16	27	2	25	18.52

Programs are written in C and have been executed on IBM PC compatibles with an Intel 80486DX2/66MHz, an Intel Pentium/133MHz, and PII/233MHz (with MMX) CPUs respectively. The execution time for each module is listed in Table 1. Notice that MMX instructions are only used in the matching module of the last implementation. Further potentials of optimization are to be further investigated, e.g. in the warping module. Fig. 7 shows images from several testing sequences. Some stabilization results are available on the Internet

http://vis-www.cs.umass.edu/~zhuzhg
http://vision.cs.tsinghua.edu.cn/~zzg/stabi.html
http://www.cse.unsw.edu.au/ ~vip/vs.html

VI. CONCLUSION

We present a new video image stabilization scheme that is featured by a 2.5D inter-frame motion model and an inertial model based motion filtering method. Our empirical system on an Intel Pentium 133MHz PC performs better than other existing systems that require special image processing hardware. Implementation in Intel PII/233MHz PC with MMX technology achieves near real-time performance for 384x288 image sequences. Besides the most obvious applications such as video enhancement and tele-operation, the method has also been applied to several other vision tasks, such as 3D natural scene modeling and surveillance [13].

Automatic selection of optimal inter-frame motion models and inertial filter parameters based on input images need further research.

VII. REFERENCES

[1] Z. G. Zhu, X. Y. Lin, Q. Zhang, G. Y. Xu, D. J. Shi, "Digital image stabilization for tele-operation of a mobile robot," *High Technology Letters*, **6**(10):13-17, 1996 (in Chinese).

[2] M. Hansen, P. Anadan, K. Dana, G. van de Wal, P. Burt, "Real-time scene stabilization and Mosaic Construction", *Proc of IEEE CVPR*, 1994, 54-62.

[3] C. Morimoto, R. Chellappa, "Fast electronic digital image stabilization", *Technical Report of CVL*, 1995, University of Maryland.

[4] Q. Zheng, R. Chellappa, "A computational vision approach to image registration", *IEEE Transactions on Image Processing*, 1993, **2**(3):311-326.

[5] Z. Duric, A. Rosenfeld, "Stabilization of image sequences", *Technical Report CAR-TR-778*, July, 1995, University of Maryland.

[6] Y. S. Yao, R. Chellappa, "Electronic stabilization and feature tracking in long image sequences", *Technical Report CAR-TR-790*, Sept. 1995, University of Maryland.

[7] A. Azarbayejani, A. P. Pentland, "Recursive estimation of motion, structure, and focal length", *IEEE Trans Pattern Recognition and Machine Intelligence*, 1995, **17**(6):562-575.

[8] A. Shashua, "Projective structure from uncalibrated images: structure from motion and recognition", *IEEE Trans Pattern Recognition and Machine Intelligence*, 1994, **16**(8):778-790.

[9] J. Oliensis, "A linear solution for multiframe structure from motion", *Technical report of University of Massachusetts*, 1994, University of Massachusetts.

[10] M. Irani, B. Rousso, S. Peleg, "Recovery of ego-motion using image stabilization", *Proc of IEEE CVPR*, June, 1994: .454-460.

[11] C. Morimoto, P. Burlina, R. Chellappa, and Y.S. Yao, "Video Coding by model-based stabilization", *Technical Report of CVL*, 1996, University of Maryland.

[12] Z. G. Zhu, G. Y. Xu, Jesse S. Jin, Y. Yang, "A computational model for motion estimation", *Symposium on Image, Speech, Signal Processing, and Robotics*, Hong Kong, Sep. 3-4, 1998.

[13] Z. G. Zhu, G. Y. Xu, X. Y. Lin, "Constructing 3D natural scene from video sequences with vibrated motions", *IEEE VRAIS'98* , Atlanta, GA, March 14-18,1998:105–112

[14] C. Fermuller, "Passive navigation as a pattern recognition problem", *Int. J. Computer Vision*, 1995, **14**:147-158.