Block Processing Video Stabilization

J.Narendra Babu, M.E(PhD), M.Nageswariah, S.Shajahan, A.Maheswari

myece88@gmail.com
Electronics & Communications Engg. D ept., SIETK, PUTTUR,Andhra Pradesh, I ndia

Abstract- The removal of unwanted vibrations in a video sequence induced by camera motion is an essential part of video acquisition in industrial, military and consumer applications. In this paper, we present a new image processing method to remove such vibrations and reconstruct a video sequence void of sudden camera movements. This approach to separating unwanted vibrations from intentional camera motion is based on a block matching motion estimation framework. The estimated parameters of interframe camera motion are the noisy observations of the intentional camera motion parameters. Video stabilization algorithm consists of a motion estimation (ME) block and a motion correction (MC) block. ME estimates the motion between frames and can be divided as a local motion estimator and a global motion decision unit. Basically the local motion estimator will return the estimated dense optical flow information between successive frames using typical block-based methods. The global motion decision unit will then determine an appropriate global transformation that best characterizes the motion described by the given optical flow information. Finally MC warps the current frame using the filtered global transformation information and generates the stabilized video sequence.

Index Terms- Video stabilization, motion estimation, motion compensation, motion vector.

I. Introduction

Video footage from hand-held camcorders is typically jerky due to small, unwanted camera movements. Removal of those undesired movements requires video stabilization techniques. As its name suggests, video stabilization is the process of generating a compensated video sequence where image motion by the camera's undesirable shake or jiggle is removed. Digital image processing techniques are often used to perform such a task and are favorable over mechanical or optical video stabilization approaches since modern VLSI techniques will allow a more compact camera design.

The implementation of the algorithm has to be cheap in the context of used memory and CPU power. The image stabilization method, Digital Image Stabilization (DIS) meets these demands, as it uses the image stream for stabilization and therefore does not need any additional equipment. This report describes one method of DIS, called block matching motion estimation, which uses matching the blocks with in the frames to estimate the motion

II. Video Stabilization Algorithm Structure

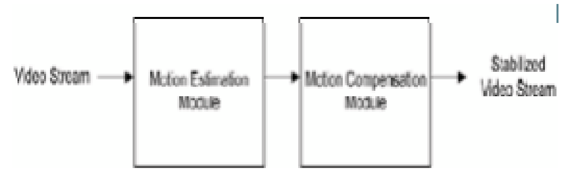

Figure. 1 :

As in any video stabilization algorithm, the crucial component of this algorithm lies on motion estimation. Since in real world, camera motion often involves some type of global transformation, it is particularly important to get an accurate estimate of the global motion when performing motion estimation. To account for various types of motions such as rotation, zoom and so on, applying parametric motion models is highly preferred over pure block-based motion methods. On the other hand, to better illustrate the algorithm, it is, however, always advantageous to start with the simplest scenario, which in our case is the translational camera motion which we shall describe in next section.

III. Motion Estimation Module

The block matching motion estimation algorithm is based on the following assumptions that: each frame in the given image sequence is distinct, and the image instability is the result of translation, rotation, skewing and scaling between frames. Through analyzing image frames, the motion vectors (including amounts of translation, rotation and scaling), which are the basis of compensation processing, can be calculated. Motion estimation between Frames is usually based on a rigid motion model as follows

$$\begin{bmatrix} x_{new} \\ y_{new} \end{bmatrix} = \begin{bmatrix} \sigma_x & 0 \\ 0 & \sigma_y \end{bmatrix} \begin{bmatrix} \cos\alpha & \sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x_{old} \\ y_{old} \end{bmatrix} + \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}$$

The above given model is explained in the following text. In the formula, xnew, xold are horizontal coordinates of corresponding pixels in input frame and reference frame; ynew, yold are vertical coordinates of corresponding pixels in input frame and reference frame; Δx, Δy are translation amounts between two frames; θ and α are the rotation and deformation angles between two frames respectively. The two factors σx, σy are the scaling factors.

Above equation can be rewritten as follows:

$$\begin{bmatrix} x_{new} \\ y_{new} \end{bmatrix} = A \begin{bmatrix} x_{old} \\ y_{old} \end{bmatrix} + \begin{bmatrix} \Delta y \\ \Delta y \end{bmatrix}$$

where, A is a sequence of rotation, scaling and angular deformation. And it can be decomposed in the form:

$$A = A_S A_D A_R$$
$$= \begin{bmatrix} \sigma_x & 0 \\ 0 & \sigma_y \end{bmatrix} \begin{bmatrix} \cos\alpha & \sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

Matrix A is a 4x4 matrix. So, it is in the form:

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$$

Hence,

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} = \begin{bmatrix} \sigma_x & 0 \\ 0 & \sigma_y \end{bmatrix} \begin{bmatrix} \cos\alpha & \sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

$$= \begin{bmatrix} \sigma_x & 0 \\ 0 & \sigma_y \end{bmatrix} \begin{bmatrix} \cos(\alpha-\theta) & \sin(\alpha-\theta) \\ \sin(\alpha+\theta) & \cos(\alpha+\theta) \end{bmatrix}$$

$$= \begin{bmatrix} \sigma_x \cos(\alpha-\theta) & \sigma_x \sin(\alpha-\theta) \\ \sigma_y \sin(\alpha+\theta) & \sigma_y \cos(\alpha+\theta) \end{bmatrix}$$

By solving above Equation

$$\sigma_x = \sqrt{a^2{}_{11} + a^2{}_{12}}$$

$$\sigma_y = \sqrt{a^2{}_{21} + a^2{}_{22}}$$

$$(\alpha - \theta) = \operatorname{atan2}(a_{12}, a_{11})$$

$$(\alpha + \theta) = \operatorname{atan2}(a_{21}, a_{22})$$

To find the values of α and θ, we need to solve above equation simultaneously,

Now, we have six variables to estimate, these values are show in Table 1. Table 1 :

Motion Vector	Description
x y σ	The Scaling Factor in x axis
σ	The Scaling Factor in y axis
θ	Rotation angle
α	Deformation angle
Δx	Translation in x axis
Δy	Translation in y axis

IV. Motion Compensation Module

The result of the motion estimation process described in the last section is capable of computing the motion vectors between two frames. The objective of motion compensation is to keep some kind of history of the motion estimates in order to create a stabilized sequence. We have seen that the DIS proposed is based on a hypothesis that the image instability in image sequence is the result of translation, rotation, skewing and scaling between frames. So, by knowing these motion vectors which are estimated in the last section, an image can be constructed.

An image can be constructed using the hypothesis in Equation which was given bellow:

$$\underline{x}'_n = A\underline{x}_n + C$$

Assume $\underline{x}_n$ is a feature point in an image at time=t where n is the image number. And assume $\underline{x}'_n$ is the same feature point in the same image at time=t+1 where n is the image number.

$$\underline{x}'_n = \begin{bmatrix} x'_n \\ y'_n \end{bmatrix}, \quad \underline{x}_n = \begin{bmatrix} x_n \\ y_n \end{bmatrix} \quad \text{and} \quad C = \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}$$

In order to estimate the value of A and C, we need 6 images; that is three images at time=t and the same 3 images but at time=t+1. This can be written mathematically as follows

$$\underline{x}'_1 = A\underline{x}_1 + C$$
$$\underline{x}'_2 = A\underline{x}_2 + C$$
$$\underline{x}'_3 = A\underline{x}_3 + C$$

These equations can also be expanded to the following equations:

$$x'_1 = a_{11}x_1 + a_{12}y_1 + \Delta x$$
$$y'_1 = a_{21}x_1 + a_{22}y_1 + \Delta y$$

$$x'_2 = a_{11}x_2 + a_{12}y_2 + \Delta x$$
$$y'_2 = a_{21}x_2 + a_{22}y_2 + \Delta y$$

$$x'_3 = a_{11}x_3 + a_{12}y_3 + \Delta x$$
$$y'_3 = a_{21}x_3 + a_{22}y_3 + \Delta y$$

These equations can be solved simultaneously to find the values of a11, a12, a21, a22, Δx and Δy. The above computation can be expressed in the form of matrix algebra as follows:

$$\begin{bmatrix} x_1' \\ x_2' \\ x_3' \\ y_1' \\ y_2' \\ y_3' \end{bmatrix} = \begin{bmatrix} x_1 & y_1 & 1 & & & \\ x_2 & y_2 & 1 & & & \\ x_3 & y_3 & 1 & & & \\ & & & x_1 & y_1 & 1 \\ & & & x_2 & y_2 & 1 \\ & & & x_3 & y_3 & 1 \end{bmatrix} \begin{bmatrix} a_{11} \\ a_{12} \\ \triangle x \\ a_{21} \\ a_{22} \\ \triangle y \end{bmatrix}$$

Which is of the form:

$$\underline{p} = P\underline{a}$$

standard pseudo inverse computation computes an optimal estimate of a such that:

$$\underline{a} = (P^t P)^{-1} P^t p$$

As we know already, pixels of an image occupy integer coordinates. We can note from above Equation that the destination pixels may lie between the integer coordinates. So, in order to create an image from these pixels, destination pixels are interpolated at the integer coordinates.

V. Results

To test the efficiency of the video stabilization algorithm, simulated video sequences were generated as follows: take one real life video sequence which was intentionally unstable. This video sequence (eg. Vasu.avi) was simulated using MATLAB 7.6 version. The images shown in fig 3 are unstable frames and the images in fig 4 are stabilized frames. Input and out put videos are accommodated in CD-ROM. The following steps are implemented to achieve the stabilized video.

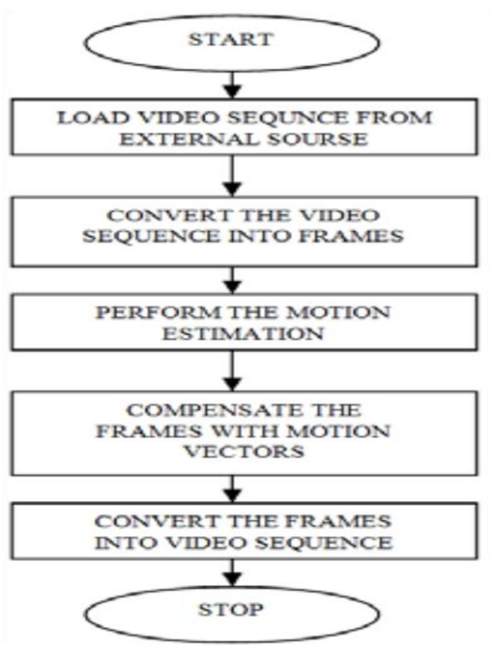

Figure 2 : Steps Followed for video stabilization

After completion of all the steps we will get stabilized video

Figure 3 : Few Input (unstable) video frames

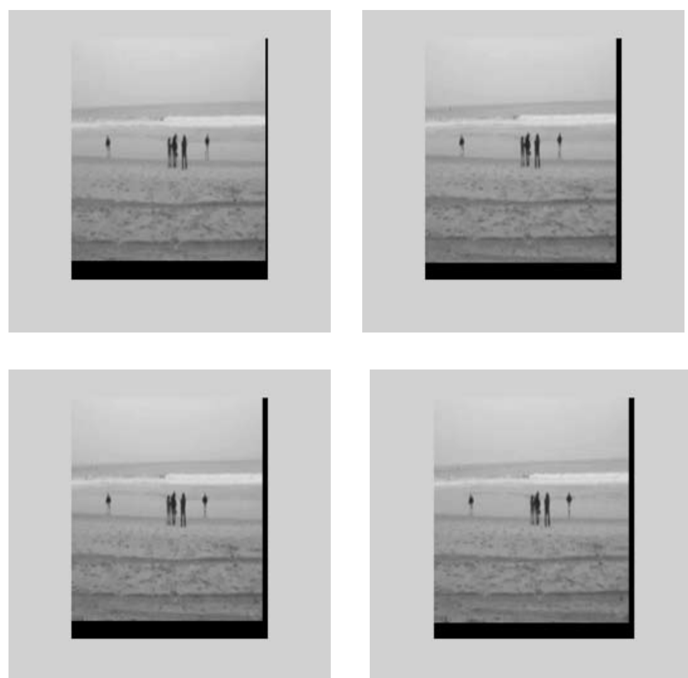

Figure 4 : Few output (stable) video frames

I. Conclusion

This report describes the video stabilization algorithm using a block-based motion model. In particularly shows how to apply this algorithm to translational and rotational camera motions. Experimental results have indicated good performance from this algorithm

References

[1] A. Litvin, J. Konrad, and W.C. Karl, .Probabilistic video Stabilization using Kalman Filtering and mosaicking,. in Proc. Of IS&T/SPIE Symposium on Electronic Imaging, Image and Video Communications., 2003, pp. 663.674.

[2] Ting Chen 'Video Stabilization Algorithm Using a Block- Based Motion Model' EE392J Project Report, Winter 2000 Information Systems Laboratory Department of Electrical Engineering, Stanford University, CA 94305, USA

[3] Alan C. Brooks. 'Real-Time Digital Image Stabilization'. EE 420 Image Processing Computer Project Final Paper, MARCH 2003

[4] Hany Farid, Jeffrey B. Woodward. 'Video Stabilization and Enhancement' Department of Computer Science, Dartmouth College.

[5] 'Video Stabilization through Dynamic Analysis of Frames Signatures' 1STMicroelectronics, AST, Imaging Group, Catania Lab, Italy – 2University of Catania – Italy.

[6] 'DIGITAL IMAGE PROCESSING' second edition, by rafeel c.Gonzalez and Richard E.woods, Pearson Prentice Hall publications.

[7] 'Fast Video Stabilization Algorithms', Mohammed A. Alharbi, Captain, RSAF AFIT/GCS/ENG/06-02.

[8] Full-Frame Video Stabilization with Motion Inpainting, IEEE Transactions on pattern analysis and machine intelligence,vol28,no.7,july 2006

[9] "Video stabilization based on a 3D perspective camera model", Guofeng Zhang ,Wei Hua , Xueying Qin , Yuanlong Shao , Hujun Bao. Springer-Verlag 2009.