International Journal of Electrical, Electronic and Communication Sciences
ISSN: 2517-9438
Vol:4, No:2, 2010

Feature Point Reduction for Video Stabilization

Theerawat Songyot, Tham Manjing, Bunyarit Uyyanonvara, and Chanjira Sinthanayothin

***Abstract*—**Corner detection and optical flow are common techniques for feature-based video stabilization. However, these algorithms are computationally expensive and should be performed at a reasonable rate. This paper presents an algorithm for discarding irrelevant feature points and maintaining them for future use so as to improve the computational cost. The algorithm starts by initializing a maintained set. The feature points in the maintained set are examined against its accuracy for modeling. Corner detection is required only when the feature points are insufficiently accurate for future modeling. Then, optical flows are computed from the maintained feature points toward the consecutive frame. After that, a motion model is estimated based on the simplified affine motion model and least square method, with outliers belonging to moving objects presented. Studentized residuals are used to eliminate such outliers. The model estimation and elimination processes repeat until no more outliers are identified. Finally, the entire algorithm repeats along the video sequence with the points remaining from the previous iteration used as the maintained set. As a practical application, an efficient video stabilization can be achieved by exploiting the computed motion models. Our study shows that the number of times corner detection needs to perform is greatly reduced, thus significantly improving the computational cost. Moreover, optical flow vectors are computed for only the maintained feature points, not for outliers, thus also reducing the computational cost. In addition, the feature points after reduction can sufficiently be used for background objects tracking as demonstrated in the simple video stabilizer based on our proposed algorithm.

***Keywords*—** background object tracking, feature point reduction, low cost tracking, video stabilization.

I. Introduction

FEATURE tracking is a fundamental task in video stabilization, in which the feature points marked on the original frame are tracked across consecutive frames in order to construct motion models. Generally, feature point detection is required to perform on every single frame in order to track motions between two consecutive frames. The computational cost is very high because a large number of feature points are to be detected and tracked along the entire sequence of video frames. Efficient video stabilizers prefer low computational cost and reasonable quality. To alleviate the detection cost and optical flow computation cost, this can be achieved by cleaning the feature points and reusing them for certain frames along the video sequence.

T. Songyot, T. Manjing, B Uyyanonvara are with Sirindhorn International Institute of Technology, Thammasat University, P.O. Box 22, Pathum Thani 12120, Thailand (e-mail: tsongyot@gmail.com, tmanjing@gmail.com, bunyarit@siit.tu.ac.th)

C. Sinthanayothin is with National Electronics and Computer Technology Center, 112 Thailand Science Park, Klongluang, Phatum Thani 12120, Thailand (email: chanjira.Sinthanayothin@nectec.or.th)

Feature points can be selected by a corner detection algorithm, which detects corners and dots within a frame. There are many detection algorithms available. Kanade-Lucus-Tomasi (KLT) [1] algorithm is a detection algorithm based on Harris corner definition [2]. The KLT corner detection is used in our proposed algorithm because the algorithm is designed for detecting features that are good for tracking purpose [1]. Generally, an application is interested in a particular subset of feature points which share common characteristics because this subset is essentially sufficient to be used. For example, clustering matching technique is used to detect points that lie closely to one another without prior modeling [3]. In this paper, however, feature points that belong to background objects are our primary concern. It is assumed that most feature points belong to background scenes. H-C Chang, et.al. [4] proposed a motion modeling method based on iteratively trimmed least-square method. They used estimated standard deviation to identify the points with large errors as outliers, which are later discarded from the data set. In our algorithm, a similar approach is used but instead based on studentized residual because of its high sensitivity in detecting the outliers. In the optical flow computation, feature-based approaches [5] are more preferable than gradient-based approaches [2,6] because our proposed algorithm maintains the points which are directly applicable with feature-based approach. Lucas-Kanade technique [7] is of our choice in our algorithm. This paper also includes a simple video stabilizer based on the proposed algorithm to demonstrate its practical application.

II. Feature Points Reduction Algorithm

The reduction process is divided into 3 phases as shown in Fig 1. Initially, the algorithm starts by setting up a set T for maintaining feature points and a "template" which will be used for reference.

Phase I is called "feature point preparation". The feature points in tracking set T are checked against specific criteria if it is still qualified for tracking in the following steps or not. Whenever these points are not qualified, corner detection is to be performed on the frame queried previously. Since the set T is initially empty; therefore corner detection is required once at the beginning. Note that the frame used to reset the tracking set is the "template".

Phase II is called "optical flow computation". When feature points in the tracking set are ready, a new frame is queried. In this phase, only points in the tracking set are used to compute optical flows from the template to the new frame.

Phase III is called "feature point reduction". The feature

points are refined against the computed optical flows. The outlier elimination process removes, from the tracking set, points that belong to moving objects. This process also internally repeats for some iterations to meet termination criteria. The points remaining then belong to only steady objects or background scenes.

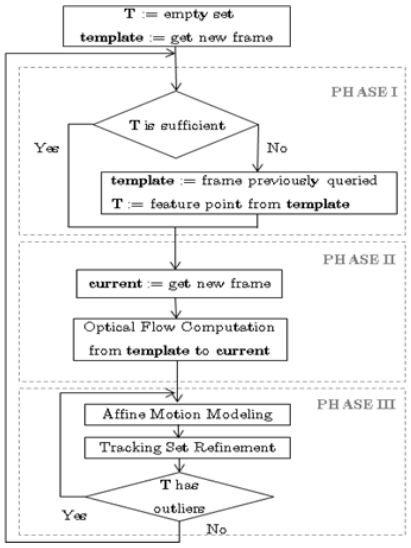


Fig. 1 Flow chart of the proposed algorithm

III. Feature Point Preparation

A. Feature Point Sufficiency for Motion Modeling

The set of feature points, as denoted by T, is said to be insufficient for future tracking if, when used for in the modeling process, points in the set cannot yield a reasonable accuracy. There are many measurements to determine the accuracy of modeling. To reduce the computational cost, mean square error (MSE) is used since it is already computed in Phase II and available throughout the algorithm. The motion model of which MSE is greater than an appropriate value, as shown in equation (1), will no longer yield a preferable accuracy. Note that the feature points are insufficient in the first loop because none of the points are presented.

$$MSE < \varepsilon \tag{1}$$

By observation during the experiment, the reduction process, which will be presented in phase III, can reduce the mean square error (MSE) of the model to roughly lower than 16 pixel^2. In our algorithm, the predefined ε is therefore chosen to be 16. If criterion in equation (1) fails (i.e. the all of the maintained feature points cannot estimate a model that meets a reasonable accuracy), those feature points are insufficient and should be revised in the next subsection.

B. Selection of Feature Points

Feature point (corner) detection is required to perform on the template frame to extract new points when the maintained points are no longer sufficient for tracking. The common definition of corners, which is defined by C. Harris and M. Stephens [2], are places where the matrix in equation (2) has two large eigenvalues λ_1, λ_2 over a local window of size K [8].

$$\begin{bmatrix} \sum_{-K\le i,j\le K} I_x^2(x+i,y+j) & \sum_{-K\le i,j\le K} I_x(x+i,y+j)I_y(x+i,y+j) \\ \sum_{-K\le i,j\le K} I_x(x+i,y+j)I_y(x+i,y+j) & \sum_{-K\le i,j\le K} I_y^2(x+i,y+j) \end{bmatrix} \tag{2}$$

In KLT corner detection, good corners are determined by thresholding the minimum value for eigenvalues λ_1, λ_2, as shown in equation (3).

$$\min(\lambda_1, \lambda_2) > \lambda_{\text{threshold}} \tag{3}$$

This thresholding criterion makes the KLT algorithm produce reliable results. A typical threshold value for $\lambda_{\text{threshold}}$ is between 0.1 and 0.01 [9]. In our experiment, the threshold $\lambda_{\text{threshold}}$ is chosen to be 0.1 so that the corners detected are very strong.

IV. Optical Flow Computation

Optical flow is a vector at a pixel indicating the motion change of that pixel toward another frame. There are many approaches to determine optical flows. Since our proposed algorithm already maintains the feature points, these points are directly applicable to the feature-based approach. In this paper, the most popular "Lucas-Kanade" optical flow technique is used [7]. Using the maintained feature points, the optical flows will present the motion of the feature points referencing from the template frame to the current frame.

V. Feature Point Reduction

By the assumption that the majority of feature points belongs to background scenes, points that move differently from the majority are considered outliers; they represent features that belong to moving objects. The reduction process is therefore to remove the points whose optical flow deviates from the majority as many as possible. This process repeats until all outliners are eliminated.

A. Affine Motion Estimation using Least Square Method

To recognize the motion of feature points, a motion model is to be constructed first. The motion model used in our proposed algorithm is the simplified affine motion model, presented in equation (4). This model was used in a video stabilization process, and yielded high performance with reasonable video quality [4]. Let (x,y) be any points in the template frame, and (x′,y′) be the corresponding points in the current frame. The motion from (x,y) to (x′,y′) is modeled by parameters a, b, c and d, as shown in equation (4).

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & -b \\ b & a \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} c \\ d \end{bmatrix} \tag{4}$$

Assuming *n* optical flow vectors are obtained from the computation in the previous phase, each vector is represented by the position (x_i,y_i) in the template frame, and the corresponding position (x_i',y_i') in the current frame; for $i = 1, 2, 3, \ldots n$. Using these vectors to estimate the motion model, multivariate linear regression method is a common approach. However, its computational cost is rather expensive. Our proposed algorithm adopts single-dependent-variable linear regression method by exploiting the fact that the covariance of x_i and y_i is zero. For estimating the parameters *a*, *b*, *c* and *d*, the least square solution to the linear system in equation (5) gives the same result as from the multivariate method with lower cost.

$$\begin{bmatrix} x_1 & -y_1 & 1 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ x_n & -y_n & 1 & 0 \\ y_1 & x_1 & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ y_n & x_n & 0 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} x_1' \\ \vdots \\ x_n' \\ y_1' \\ \vdots \\ y_n' \end{bmatrix} \tag{5}$$

Having all the points participate in motion modeling, the model obtained cannot truly represent the motion of the majority because of the presence of outliers. The result is a compromise between the majority and outliers. However, the number of feature points that belong to the majority is assumed to be much larger than the number of outliers. The estimated model is basically close the accurate model.

B. Outlier Elimination using Studentized Residuals

Based on the motion model computed, outliers are poorly fitted by the estimation. Because they are the minority, the estimated results for outliers are more deviated from actual position than for the other points. In outlier analysis, studentized residual [10] is a common approach for detecting outliers. Particularly, our algorithm employs externally studentized residuals because of its higher sensitivity to outliers. As shown in equation (6), a studentized residual is defined as the ratio of a residual (an error from the estimation) and its enhanced standard deviation.

$$t_i = \frac{e_i}{\sqrt{\sum_{j \neq i} e_i^2} \cdot \sqrt{1-h}} \tag{6}$$

For the sake of simplicity and efficiency, the approach to computing studentized residuals is slightly different since single-dependent-variable regression is used instead of multivariate regression; therefore, the calculation process is redefined and to be presented. The residual is the error distance between the estimated position and the actual position. The computation of the residual, denoted by e_i, is shown in equation (7).

$$e_i = \left\| \begin{bmatrix} x_i' \\ y_i' \end{bmatrix} - \left(\begin{bmatrix} a & -b \\ b & a \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \begin{bmatrix} c \\ d \end{bmatrix} \right) \right\| \tag{7}$$

The standard deviation for each point (x_i,y_i) is enhanced by an *h* value. Equation (8) defines the H_i matrix whose its two diagonal elements $H_i(1,1)$ and $H_i(2,2)$ can be used as the *h* value in standard deviation approximation. However, our single-dependent-variable approach gives 2 different *h* values separately for x_i and for y_i.

$$H_i = \phi_i (\Delta \Delta^t)^{-1} \phi_i^t \quad , where$$

$$\Delta = \begin{bmatrix} x_1 & -y_1 & 1 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ x_n & -y_n & 1 & 0 \\ y_1 & x_1 & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ y_n & x_n & 0 & 1 \end{bmatrix} \quad , and \quad \phi_i = \begin{bmatrix} x_i & -y_i & 1 & 0 \\ y_i & x_i & 0 & 1 \end{bmatrix} \tag{8}$$

In theory, the two *h* values can be statistically combined. However, the calculation would be unnecessarily complicated because the combined *h* value should is not significantly different from the two original *h* values. For simplicity and efficiency, both values of *h* are used to compute t_i separately, but only the larger t_i is selected as the final studentized residual.

Outliers are points where their studentized residuals are exceptionally large. In this paper, it is assumed that 95% of all possible feature points are the majority whereas the other 5% are considered outliers. According to the t-distribution, 95% of all possible feature points should not have the studentized residuals exceeding 2.132. The outliers (points of which t_i exceeds 2.132) are then eliminated from the maintained set.

C. Termination Criterion

Affine motion modeling and outlier elimination repeat for some iterations in order to build up a model that best estimates the motion of the majority, where outliers do not involve. The termination criterion determines when the iterative process should stop. In this paper, a very strong criterion is used: the reduction process terminates when no new outliers are identified. This criterion is equivalent to and can easily be inspected by equation (9). That is, the MSE remains unchanged after the elimination process.

$$MSE_{before} = MSE_{after} \tag{9}$$

Even though this criterion sounds too strong and may greatly reduce the size of the tracking set, statistical theory guarantees that studentized residuals are precise enough to eliminate only outliers.

VI. Application To Video Stabilization

A set of points that contain only features steady objects is a preferable characteristic for video stabilization. With the outliers presented there, the motion of camera can easily be misinterpreted. A simple video stabilizer is achieved by appending the stabilization phase to the algorithm. This phase begins with optical flow computation for the maintained feature points. Then, the reverse motion model for each frame is calculated by swapping (x_i,y_i) with (x_i',y_i'). The stabilized frames is produced by geometrically transforming each frame with respect to its calculated reverse motion model. Fig. 2 shows the additional flow chart for the simple video stabilizer. With this simple additional implementation, the output stabilized video is surprisingly of reasonable quality.

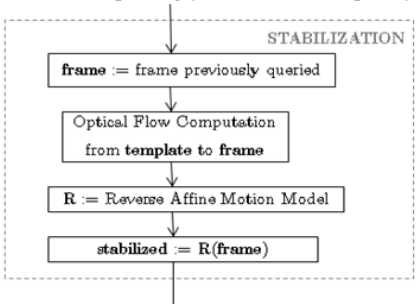


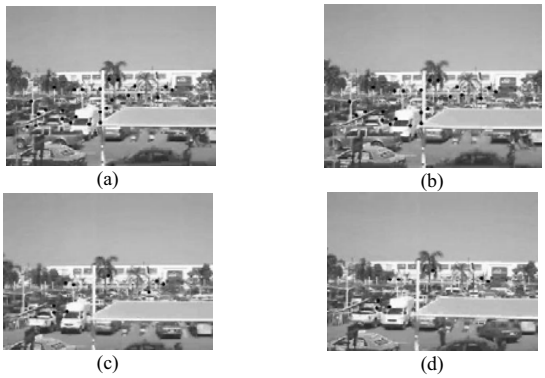
Fig. 2 Stabilization Phase

VII. Experimental Results

The following 3 experimental results were conducted on 40 sample videos to show 3 different viewpoints: effectiveness, efficiency and practical application. The samples videos were taken from both indoor and outdoor environment. All the samples are 240-frame in length with resolution of 320 × 240 pixel^2. To investigate failure, the stabilization phase is also appended.

A. Effectiveness of the Proposed Algorithm

After the experiment was conducted on the sample videos, 32 out of 40 samples were successfully stabilized whereas the other eight samples could not be completed. Two common characteristics that cause failure in samples are that the videos contain either repetitive/similar patterns or large sudden moving objects. Both cases can easily lead to optical flow errors, and lack of ability to maintain feature points.



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

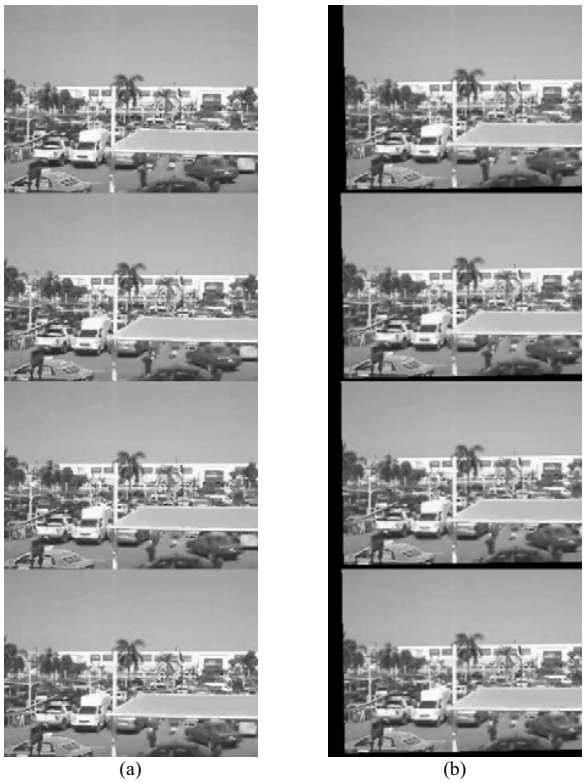
Fig. 3 Sample frames with tracked points

Fig. 3 shows 4 selected frames from a sample video with black dots indicating feature points of interest. Fig. 3 (a) and (b) depict frame 44th and 45th respectively. The feature points that belong to the running car on the bottom left of Fig. 3 (a) disappeared in Fig. 3 (b) because of the reduction process. Fig. 3 (c) and (d) depict frame 150th and 200th respectively. Over a period of 50 frames, the feature points on Fig. 3 (d) remain stable from Fig. 3 (c) since all of the outliers had been removed. This reflects that outliers on moving objects were gradually removed from frame to frame until the remaining points belong only to the background scene. These results imply that the reduction algorithm can effectively preserve feature points on background objects.

B. Efficiency of the Proposed Algorithm

Out of 32 video samples that worked properly, the proposed algorithm could exceptionally stabilize 25 samples without requiring the second round of feature point detection. Note that the detection on the first frame is mandatory. For the other seven samples, detection algorithm was performed ranging from 2 to 14 times. That is, for each video sample, less than 6% of the total frames required feature point detection. On average of 32 samples, the detection algorithm performed only twice for each video file, which roughly approximated 1% of the total frames. Compared to typical stabilizers which perform detection on every single frame, these results demonstrate significant improvement in computational cost.

C. Application to Video Stabilization



(a) (b)

Fig. 4 Frames from (a) the original video, (b) the stabilized video based on the proposed algorithm.

Fig. 4 (a) shows 4 successive frames from a sample video. Fig. 4 (b) shows the corresponding frames stabilized using optical flows that are based only on the maintained feature points. This sample video also contains several moving objects; for example, cars, people. The demonstration shows a very desirable result because the shakiness of the video is no longer noticeable by human eyes. Therefore, the feature points maintained by the reduction algorithm are sufficient for use in practical stabilization.

VIII. Conclusion

In this paper, an algorithm for feature point reduction is proposed. The number of feature points is reduced by eliminating outliers that belong to moving objects using statistical methods. The reduction algorithm can effectively remove the outlier while preserving points that belong to steady objects such as background scene. As a result, fewer of feature points involve in the optical flow computation, and the number of times the detection algorithm is required to perform is significantly reduced. The reduction algorithm therefore efficiently improves computational cost. Practical applications of the proposed algorithm are low-cost video stabilizations. Based on the proposed algorithm, the video stabilization can produce stabilized videos with reasonable quality.

Acknowledgment

This research is conducted as a research project at Sirindhorn International Institute of Technology (SIIT) and partly funded by Young Scientist and Technologists Programme (YSTP), operated under the supervision of National Science and Technology Development Agency (NSTDA), Patumtani, Thailand.

References

[1] J. Shi and C. Tomasi, "Good Features to Track", *IEEE Conference on Computer Vision and Pattern Recognition*, June 1993.
[2] C. Harris and M. Stephens, "A Combined Corner and Edge Detection", *Proceedings of the Fifth IEEE International Conference on Automatic Face and Gesture Recognition*, pp. 287-293, May 2002.
[3] D. P. Kottke and Y. Sun, "Motion Estimation Via Cluster Matching", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 11, pp. 1128-1132, November 1994.
[4] H-C. Changl, S-H. Lai, and K-R. Lu, "A Robust and Efficient Video Stabilization Algorithm", *IEEE International Conference on Multimedia and Expo (ICME)*, 2004.
[5] C. Morimoto and R. Chellappa, "Automatic digital image stabilization", *IEEE Intern. Conf. on Pattern Recognition*, pp. 660-665, 1997.
[6] T. Chen, "Video Stabilization Algorithm Using a Block-Based Parametric Motion Model", *EE392J Project Report*, winter 2000.
[7] B. Lucas and T. Kanade, "An Iterative Image Registration Technique with an Application to Stereo Vision", *Proceedings of Imaging Understanding Workshop*, pp. 121-130, 1981.
[8] K. G. Derpanis, "The Harris Corner Detector", October 2004.
[9] G. Bradski and A. Kaehler, *Learning OpenCV: Computer Vision with the OpenCV Library*, Sebastopol: O'Rielly Media, Inc.
[10] D. C. Montgomery, G. C. Runger and N. F. Hubele, *Engineering Statistics*, Hoboken: John Wiley & Sons, Inc.