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 techniques are computationally expensive and should therefore 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 computational cost. The algorithm starts with examining the maintained feature points. Corner detection is required only when the feature points are insufficient. Then, optical flows are computed toward the new frames for only the maintained points. Further, the simplified affine motion model is developed by linear regression with some outliers present. Studentized residuals are used to eliminate outliers that belong to moving objects. The modeling and elimination processes repeat until no more outliers are identified. Finally, the entire algorithm repeats along the video sequence together with the points remaining from the previous iteration. As a practical application, an efficient video stabilization can be achieved by exploiting the computed motion models. Our study shows that feature points can be maintained for certain sequence of frames, thus having fewer optical flow computations. The number of times corner detection needs to be performed is also greatly reduced, thus significantly improving the computational cost. In addition, the feature points after reduction can be sufficiently used in background objects tracking as demonstrated in a simple video stabilizer based on the proposed algorithm.

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

I. INTRODUCTION

FEATURE tracking is one of the common processes in video stabilization. In the process, the feature points marked on the original frame are tracked across a number of successive frames in order to construct a motion model. Normally, a feature point detection algorithm is required to perform on every single frame in order to track motions between successive frames. This can result in higher computational cost because of a large number of feature points can be tracked along the sequence of video frames. However, most video stabilizers require low computational cost and acceptable level of quality. An algorithm for reducing the number of feature points and maintaining them so as to reduce the detection cost and optical flow computation cost is therefore preferable.

In this paper, feature points for tracking are prepared by a corner detection algorithm. Corner detection algorithms detect corner and dot within a frame, which will be used for tracking purpose. Kanade-Lucus-Tomasi (KLT) [1] corner detection is a detection algorithm based on Harris corner detection [2]. It is used in our proposed algorithm because the algorithm is designed for detecting features that are good for tracking purpose [3]. In some applications, a specific subset of feature points is generally good enough to be tracked. For example, clustering matching technique is applicable for points that lie closely to one another without prior modeling [4]. 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. [5] proposed a motion modeling 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, studentized residual is used instead because of its sensitivity in detecting the outliers. In the optical flow computation, feature-based approaches [6] are more preferable than gradient-based approaches [2,7] because our proposed algorithm maintains points which can directly be used with feature-based approach. This paper also present a simple video stabilizer based on the proposed algorithm to demonstrate its practical application.

II. FEATURE POINTS REDUCTION ALGORITHM

![Flow chart of the proposed algorithm](image)

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)
First, 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. Initially, feature points are empty therefore corner detection is required once at the beginning. In this paper, a frame used to reset the tracking set will be called a “template”.

Second, 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.

Last, 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 repeats for some iterations to meet termination criteria. The points remaining will belong to only steady objects or background scenes.

A. Feature Point Preparation

i. Sufficiency of Feature Points

The set of feature points, denoted by T, is said to be insufficient for future tracking if, when used for in 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 as it is 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.

$$\text{MSE}_{\text{model}} < \varepsilon$$

(1)

By observation during the experiment, the reduction process, which will be presented in section C, can reduce the mean square error (MSE) of the model to below 16 pixel\(^2\); at most 16 pixel\(^2\). In our algorithm, the predefined \(\varepsilon\) is therefore chosen to be 16. If all of the available feature points in the set cannot estimate a model that meets the criterion in equation (1), those feature points are said to be insufficient and should be revised in the next section.

ii. Feature Point Detection

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 \(\lambda_1, \lambda_2\) [8].

$$M = \begin{bmatrix} I_x^2 & I_x I_y \\ I_y I_x & I_y^2 \end{bmatrix}$$

(2)

In KLT corner detection, good corners are determined by thresholding the minimum value for eigenvalues \(\lambda_1, \lambda_2\), as shown in equation (3).

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

(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.

B. Optical Flow Computation

Optical flow is a vector at a pixel indicating the motion change of that pixel to a specific frame. There are many approaches to determine the optical flow. Since our proposed algorithm already maintains a set of feature points, these points can directly be used with the feature-based approach. In this paper, the most popular “Lucas-Kanade” optical flow technique is used [1]. By using the maintained feature points, optical flows will present the motion of the feature points from the template frame.

C. Feature Point Reduction

By the assumption that most feature points belong to background scenes, the points that move differently from the majority are considered outliers; they represent features points that belong to moving objects. The reduction process is therefore to remove the points that behave strangely as many as possible. This process repeats until all outliers are eliminated.

i. Affine Motion Modeling using Linear Regression

To recognize the motion of majority of feature points, a motion model is to be constructed first. In this paper, the motion model used in our proposed algorithm is a simplified affine motion model. This model was used in a video stabilization process, and yields high performance and reasonable video quality [5]. Let \((x, y)\) be a feature point in the template frame, and \((x', y')\) be the corresponding point in the current frame. The motion from \((x, y)\) to \((x', y')\) is modeled by 4 parameters, as shown in equation (4).

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

(4)

Assume \(n\) motion vectors obtained from the computation in the previous phase. The vectors are 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 \(i = 1, 2, 3, \ldots n\). To estimate the motion model, multivariate linear regression method is commonly used; however, it is computationally
expensive. By employing the fact that the covariance of \( x_i \) and \( y_i \) is zero, the least square solution to the linear system in equation (5) also gives the same result as from the multivariate method.

\[
\begin{bmatrix}
    x_i - y_i & 1 & 0 \\
    \vdots & \vdots & \vdots \\
    x_n - y_n & 1 & 0 \\
      y_i & 1 & 0 \\
      y_i & 0 & 1 \\
      y_n & 0 & 1 \\
\end{bmatrix} = \begin{bmatrix} \Delta \\ \phi \end{bmatrix}
\]

\[ \Delta = \begin{bmatrix} x_i - y_i & 1 & 0 \\
                    a & b & c \\
                    d & 1 & 0 \\
\end{bmatrix} \text{ and } \phi = \begin{bmatrix} x_i - y_i & 1 & 0 \\
                          y_i & 1 & 0 \\
\end{bmatrix} \]

(5)

Since all of the points participate in motion modeling, the model obtained does not truly represent the motion of the majority but the motion compromising 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. This estimated model is therefore mainly resembled to the model for only the majority.

**ii. Outlier Elimination Using Studentized Residuals**

From the computed motion model, the estimation of outliers will be much deviated from the actual position. In outlier analysis, studentized residuals are commonly used to detect outliers. Particularly, our algorithm employs externally studentized residuals because of its greater 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 standard deviation.

\[ t_i = \frac{e_i}{\sqrt{\sum_{j=1}^{n} e_j^2 \cdot (1 - h)}} \]  

(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 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).

\[ e_i = \begin{bmatrix} x_i' \\ y_i' \end{bmatrix} - \begin{bmatrix} a & -b \\ b & a \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} \]  

(7)

By using single dependent variable regression, it gives 2 different \( h \) values separately for \( x_i \) and for \( y_i \). Equation (8) defines the \( H_i \) matrix whose two diagonal elements \( H_i(1,1) \) and \( H_i(2,2) \) will be used as \( h \) in standard deviation approximation.

\[ H_i = \phi_i (\Delta \Delta')^{-1} \phi_i^T \]  

where

\[ \Delta = \begin{bmatrix} x_i - y_i & 1 & 0 \\
                    a & b & c \\
                    d & 1 & 0 \\
\end{bmatrix} \text{ and } \phi_i = \begin{bmatrix} x_i - y_i & 1 & 0 \\
                          y_i & 1 & 0 \\
\end{bmatrix} \]

Theoretically, the two \( h \) values can be statistically combined. However, the calculation would be complicated and the combined \( h \) value should not be significantly different from the original \( h \) values. For simplicity and efficiency, both values of \( h \) are therefore 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 very large. In this paper, it is assumed that 95% of all possible feature points are the majority; 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 therefore eliminated accordingly.

**iii. 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 simply be determined by equation (9). That is, the MSE remains unchanged after the elimination process.

\[ MSE_{before} = MSE_{after} \]  

(9)

Even though this criterion sounds too strong and may greatly reduce the size of the tracking set, the studentized residuals used are believed to be precise enough to eliminate only outliers.

**III. Application to Video Stabilization**

The proposed algorithm for feature points reduction can be very useful for video stabilization because only feature points that belong to background objects are involved in the computation processes: for example, optical flow computation. Based on this, optical flows for moving objects will not be computed, thus reducing the computation cost. Moreover, maintaining points allow reusing feature points from the frame nearby, therefore reducing the number of times corner detection needs to perform.
A simple video stabilizer can be achieved by appending the stabilization phase to the algorithm. In this phase, the process begins with optical flow computation for points in the tracking set. Then, the reverse motion model is calculated by swapping \((x_i, y_i)\) with \((x'_i, y'_i)\). By transforming the frame geometrically with respect to the calculated model, the stabilized frame is finally obtained. 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](image)

IV. EXPERIMENTAL RESULTS

The following 3 experimental results were conducted on 40 sample videos to show three different viewpoints: effectiveness, efficiency and practical application. These samples videos were taken from both indoor and outdoor environment using Kodak M1033 digital camera. All the samples are 240-frame in length with the solution of 320 × 240 pixel².

A. Effectiveness of the Proposed Algorithm

After the experiment was conducted with the sample videos, 32 out of 40 samples were successfully stabilized whereas the other eight samples could not be completed. From the experimental results, two common characteristics of the failed samples are that the videos contained either repetitive/similar patterns which can lead optical flow errors, or large moving objects which can greatly affect the motion modeling.

![Fig. 3 Sample frames with tracked points](image)

Fig. 3 shows 4 selected frames from the sample video with black dots showing feature points of consideration. 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 algorithm. This reflects that outliers on moving objects were gradually removed. 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. The result implies that the proposed algorithm can effectively preserve feature points on background objects.

B. Efficiency of the Proposed Tracking Algorithm

Out of all 32 video samples that work properly, the proposed algorithm could stabilize 25 samples without requiring a new round of corner detection. That is, some 78% of the 240-frame samples could be tested with only one tracking set. In the other seven samples, detecting algorithm was performed ranging from one to 13 times. The average number of times that the tracking set is reset is 0.72 times per one video sample or around 0.003 times per frame (calculated from all the samples which were successfully stabilized). These results demonstrate significant improvement in computational cost.

C. Application to Video Stabilization

![Fig. 4 Frames from (a) the original video, (b) the stabilized video based on the proposed algorithm.](image)
Fig. 4 (A) shows 4 successive frames from the sample video. Fig. 4 (B) shows corresponding frames stabilized using optical flows that are based only on the maintained feature points. This sample video also contains a number of moving objects; for example, cars, people. The result reflects that the maintained points are sufficient for optical flow computation in practical stabilization.

V. CONCLUSION

In this paper, an algorithm for feature point reduction is proposed. The number of feature points is reduced by eliminating feature points that belong to moving objects, which are known as outliers. The remaining points therefore belong to only steady objects such as background scene. Practical applications of the proposed algorithm are low-cost video stabilizations because fewer of feature points are involved in the optical flow computation, and the number of times the detection algorithm is required to perform is significantly reduced. 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