LUCAS-KANADE WITHOUT ITERATIVE WARPING

Alex Rav-Acha *Shmuel Peleg*

School of Computer Science and Engineering
The Hebrew University of Jerusalem
91904 Jerusalem, Israel
E-Mail: {alexis,peleg}@cs.huji.ac.il

ABSTRACT

Many methods for motion computation and object tracking are based on the Lucas-Kanade (LK) framework [1]. We present a method which substantially speeds up the LK approach while preserving its accuracy. This acceleration is obtained by avoiding the iterative image warping, inherent to the LK framework. A three-fold speedup is observed on standard image alignment tasks. Our second contribution focuses on adopting a multi-frame approach in order to increase alignment accuracy and robustness. By utilizing the acceleration procedure, the complexity of this multi-frame alignment becomes comparable to that of the two-frame approach.

Index Terms— Motion analysis, video stabilization, image alignment, direct methods

1. INTRODUCTION

Many robust methods exist for motion computation. However, the iterative motion analysis proposed by Lucas & Kanade [1] still dominates the field of motion analysis. It is widely used for both parametric motion computations (For example [2, 3, 4]) and object tracking [5].

In this paper we describe how LK can be implemented with a computational cost of a single iteration without effecting the accuracy of the estimated motions. We also present a multi-frame alignment which uses all the pixels in several frames.

A similar problem was addressed in [6, 7] but have substantial computational cost. With our method, a multi-frame alignment is obtained with only slight additional computational cost compared to two-frames methods. As a result, it is more appropriate to online and real time applications.

We begin by a brief description of the LK method for computing motion between two frames. We show how it can be implemented without iterative warping. This idea can be used either as a stand-alone acceleration of traditional methods, or as a component in a multi-frame alignment introduced in Section 3. The proposed algorithms were extensively tested on image sequences, and a few results are presented in section 4.

2. LUCAS-KANADE WITHOUT ITERATIVE WARPING

We first describe LK for sub-pixel translation only, and in Section 2.3 we introduce the accelerated LK to larger and more general motions.

2.1. Brief Introduction to the LK method

Let I_1 and I_2 be a pair of images, and let the motion between the two images be a pure translation (u, v). Under the constant-brightness assumption [2], the translation (u, v) can be computed by minimizing the error:

$$Err(u,v) = \sum_{x,y \in R} (I_2(x+u, y+v) - I_1(x,y))^2. \quad (1)$$

The summation is over the region of analysis R. This region includes most of the image for many image alignment applications [2, 3], or only a window around a certain pixel for local motion computations [1, 5].

Let ∇I be the gradient of I_2, computed from the image intensities. The basic step in computing pure translation (u, v) is solving the set of equations $A\,[u,v]^T = b$, where the LK matrix A and the vector b are given by:

$$A = \sum_{x,y \in R} \nabla I(x,y) \nabla I(x,y)^T \quad (2)$$

$$b = - \sum_{x,y \in R} \nabla I(x,y)(I_2(x,y) - I_1(x,y))$$

In the iterative scheme, given an estimation of the image translation (u, v) from the current step, the image I_1 is warped towards the image I_2 (using back-warping) according to the current motion parameters, and the warped image is used to compute b in the next iteration, until convergence.

The LK matrix A does not change between the iterations, and is computed only once as described in detail by [4]. On the other hand, the term b (Eq. 2) varies in each iteration:

$$b^{(i+1)} = - \sum_{x,y} \nabla I(x,y) I_2(x,y) + \sum_{x,y} \nabla I(x,y) I_1^{(i)}(x,y) \quad (3)$$

where $I_1^{(i)}$ is obtained by warping I_1 towards I_2 according to the estimation of the motion between I_1 and I_2 after the i^{th} iteration.

2.2. Accelerating Sub-Pixel Translations

We propose to accelerate the computation of the term $b^{(i+1)}$ in Eq. 3 by avoiding the iterative image warping. When the relative translation between the frames is smaller than a pixel, image warping can be performed using a convolution: $I_1^{(i)} = I_1 * m^{(i)}$, where $m^{(i)}$ is a interpolation kernel whose size depends on the interpolation scheme. (Bilinear and bicubic interpolations require kernels of sizes 2×2 and 3×3 respectively).

Following this description we can examine the right term of Eq. 3, which is the only element that needs to be re-computed at every iteration, as I_1 is warped towards I_2.

$$\sum_{x,y} \nabla I(x,y) I_1^{(i)}(x,y) = \sum_{x,y} \nabla I(x,y)(I_1(x,y) * m^{(i)}) \quad (4)$$
$$= \sum_{k,l} m^{(i)}(k,l) \sum_{x,y} \nabla I(x,y) I_1(x-l, y-k)$$

Therefore, we can rewrite $b^{(i+1)}$ (from Eq. 3) as:

$$b^{(i+1)} = r + \sum_{k,l} m^{(i)}(k,l) s(k,l) \quad (5)$$

where the sum runs over all the locations (k,l) in the interpolation kernel, and

$$r = -\sum_{x,y} \nabla I(x,y) I_2(x,y), \quad (6)$$
$$s(k,l) = \sum_{x,y} \nabla I(x,y) I_1(x-k, y-l),$$

are all vectors of size 2×1 that remain constant throughout the iterations. Since only the values of $m^{(i)}$ change in each iteration, the rest of the terms can be computed only once. As a result, very few operations are needed per each iteration, independent of the size of the region of analysis.

To conclude, without effecting the accuracy of the LK, the number of operations is reduced to that of a single LK iteration plus a small number of operations that are done in each iteration. The additional operations consists mainly of solving the equations (Eqs. 2, 5). [1].

2.3. Large Translations & Parametric Motion

Translations that are larger than a pixel can be computed by expanding the table $s(k,l)$ (from Eq. 6). The additional number of entries in the table is determined by the number of possible (whole-pixel) translations. The table s is initially empty. When a value is needed from the table, the relevant term is computed only if it was not computed before.

In addition, a multi-resolution framework is used, where the motion parameters that were estimated in lower-resolution levels are used as initial estimations for finer levels [2]. In our experiments we found that under the multi-resolution scheme, the residual translation was almost always sub-pixel.

To generalize the LK acceleration to more general motions, the image can be divided into non-overlapping windows (usually 5×5 or 7×7), and assume a constant translation in each window. *Yet, we do not have to compute a translation for each window, but directly compute a single parametric motion for the entire region of analysis*. Using these small windows, we obtain almost identical acceleration to the translational motion model without noticeable loss in accuracy.

2.4. Runtime Evaluation

The speedup achieved with the proposed method is higher for those difficult scenes where the traditional LK converges slower than usual. When using a bilinear interpolation for the image warping, the speedup ratio in the total run time ranges from 2 to 4. (This speedup is a result of a reduction by a factor of 3-10 in the number of image warps). The total running time includes the computation of Gaussian pyramids and the computation of image derivatives. For many applications, these computations are done anyway, making the number of LK iterations be the main computational cost, and maximal speedup is possible. The speedup is also increased when a more accurate interpolation is being used for the image warping, such as a bicubic interpolation. Another case where the speedup is larger is when a regularization term is used which favors small motions. Such a regularization usually increases the number of iterations needed for convergence. Some typical numerical results are given in Table 1. [2]

3. ONLINE MULTI-FRAME ALIGNMENT

The robustness and stability of the LK method can be increased by masking outliers and by using multi-frame alignment in which each frame is aligned simultaneously to several preceding frames and not only to the previous one.

3.1. Masking of Outliers

When aligning a sequence of images, we can use the alignment of frame I_n to frame I_{n-1} to determine whether we should ignore some pixels in I_n before aligning I_{n+1}. Such pixels can include, for example, moving objects in the scene. A possible mask can be based on the intensity difference after alignment, divided by the local gradient:

[1] For some platforms, the postponement of the interpolation to a later stage also allows to perform the image warping with a better precision.

[2] The analysis was performed on a PC, where the memory is very fast. In other platforms (like DSP cards) the bottleneck of the computations is usually the number of passes over the image, which further increases the benefit of using our method.

Table 1. Performance (in frames per second) of the motion computations for some typical scenarios. (The evaluation was performed on a 2.4GHz Pentium4, with 512MB RAM). It can be seen that a significant speedup is achieved when avoiding the iterative image warping. The rows marked with (*) were stabilized by aligning all the frames to the first frame in the sequence. When stabilizing "Sequence2", only a few iterations were needed until convergence. This explains its better performance compared with the stabilization of "Sequence1"

Motion & Scene Properties	Original LK (fsp)	Fast LK (fsp)
Translation (320x240)	12.9	30.7
Rotation & Translation (320x240)	11.1	30.3
Translation (600x800)	1.7	5.0
Affine (320x240), Sequence1*	9.4	40.8
Affine (320x240), Sequence2*	17.7	43.4

$$M_n(x,y) = \begin{cases} 1 & if \ \frac{\sum_{W_{x,y}} (I_n - I_{n-1})^2}{\sum_{W_{x,y}} \|\nabla I_n\|^2} < r \\ 0 & otherwise, \end{cases} \quad (7)$$

where $W_{x,y}$ is a window around (x,y), and r is a threshold (We typically used $r = 1$). When aligning I_{n+1} to I_n, pixels in I_n with $M_n(x,y) = 0$ are ignored.

3.2. Simultaneously Aligning the Image to Several Preceding Frames

Assume that the images $I_0 \dots I_{n-1}$ have already been aligned and let I_n be the new frame to be aligned. To compute the motion parameters of I_n, we now minimize the residual error after aligning this frame to its preceding frames:

$$Err(u,v) = \quad (8)$$
$$\sum_{k<n} w_k^n \sum_{x,y} M_k(x,y)(\nabla I_k(x,y) \begin{bmatrix} u \\ v \end{bmatrix} + I_k(x,y) - I_n(x,y))^2.$$

This is a multi-frame version of LK, where we also added a validity mask $M_k(x,y)$, and pre-defined weights w_k^n which control the contribution of each frame to the registration process. We used exponentially decreasing weights $w_{k-1}^n = q \cdot w_k^n$. This weighting scheme allows a fast accumulative computation of the terms in the LK equations by multiplying the old terms by a scale factor.

Note also the use of the gradients ∇I_k which are estimated from the intensities of the preceding images $\{I_k\}$. This multi-frame alignment is a more accurate version of a simple temporal averaging suggested by [3].

4. MULTI-FRAME EXPERIMENTS

The proposed multiframe algorithm has been tested in various scenarios, including videos of dynamic scenes and videos in which the image motion does not fit the motion model. Concerning computational time, the performance of the multi-frame alignment was slightly slower than the traditional single-frame alignment. To show stabilization results in print, we have averaged the frames of the stabilized video. When the video is stabilized accurately, static regions appear sharp while dynamic objects are ghosted. When stabilization is erroneous, both static and dynamic regions are blurred.

Figure 2 compares sequence stabilization using multiframe alignment with original LK (with pyramids). A large portion of the scene consists of dynamic objects (the moving pedestrians). In spite of the dynamics, after multiframe alignment the sequences was correctly stabilized.

We also tested the algorithm on long sequences to evaluate the effectiveness of the multiframe alignment in reducing the drift of the motion computations. For very long sequences, it is crucial to reduce the drift without storing a huge amount of frames in the memory. An example is shown in Fig. 1. The road sequence (Fig. 1a) was stabilized using exponentially decreasing weights with different exponents. The drift-error was defined as the average error in the computed displacement between the first and last frames, for 4 selected points in the image. (The ground truth displacements were computed manually). The advantage of using a long history to reduce the motion drift is clear from the monotonically decreasing error in Fig. 1. For example, using an exponent of $q = 0.99$ gave an average error of about a pixel. (And the error continued to decrease with exponents closer to one.) The benefit of using a validity mask is also evident from the graph. The contribution of the validity mask is larger when using fewer frames for the alignment.

Finally, Figure 3 shows a couple of tracking results. The tracked objects were selected from the Edinburgh sequence (Fig. 2). The appearance of the tracked objects change during the tracking as they step towards or away from the camera. (We have not updated the scale of the objects during the tracking). To overcome the changes of appearance we used a relatively small exponent factor ($q = 0.5$). Nevertheless, the proposed method performed well even when the traditional LK failed. The difference in the performance of the two methods was most evident in the presence of a textured background.

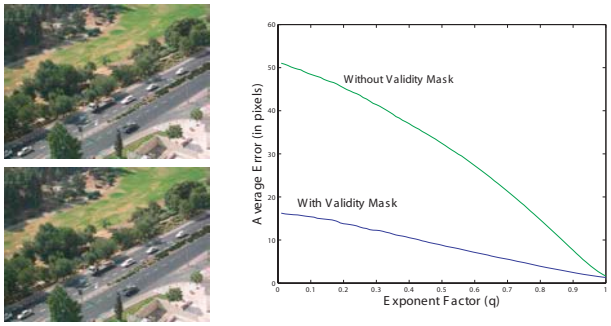

Fig. 1. The motion drift when stabilizing long sequences.
(left) Two frames out of a sequence of 1500 frames.
(right) The motion drift-error with and without a validity mask and as a function of the multiframe exponent. $q = 0$ stands for a traditional motion computation between pair of frames, and $q = 1$ is equivalent to giving equal weights to all previous frames. See text for further details.

Fig. 2. A sequence of 200 frames showing walking pedestrians at the Edinburgh festival. Some original frames are shown in top. The scene dynamics is visible by ghosting, but while using traditional LK (middle) the background is also blurry due to erroneous stabilization, the background appears much sharper using multi-frame alignment (bottom).

Using traditional LK, the tracker tended to lose the object and follow the background instead, while the multiframe tracker ignored the background and remained fixed on the object.

5. CONCLUDING REMARKS

This paper presented an algorithmic acceleration of the Lucas-Kanade method which avoids the iterative image warping used in the original method. This acceleration was also combined with a multiframe alignment to obtain a fast and robust alignment. Experimental results show improvement in both complexity and accuracy.

The multiframe alignment proposed in this paper overcomes the drawbacks of current multiframe alignment methods: high complexity or restrictive assumptions (such as small motion or large memory). We believe that by overcoming these drawbacks, the applicability of using multiframe alignment will be significantly increased.

Fig. 3. Two tracking results of pedestrians at the Edinburgh festival.
(left) The initial location, as drawn by the user.
(middle) The location of the tracker after $\sim$100 frames using traditional LK (In both cases the tracking failed).
(right) The location of the tracker after $\sim$100 frames after successfully tracking the object using multi-frame alignment.

The LK acceleration presented in this work can be used for other various applications, such as computing stereo or recovering the camera ego-motion. In all these applications, the LK method is widely used (in small windows), and therefore can be improved using the ideas presented in this paper.

6. REFERENCES

[1] B. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," *IJCAI*, pp. 674–679, 1981.

[2] J.R. Bergen, P. Anandan, K.J. Hanna, and R. Hingorani, "Hierarchical model-based motion estimation," in *ECCV*, Italy, May 1992, pp. 237–252.

[3] M. Irani, B. Rousso, and S. Peleg, "Computing occluding and transparent motions," *IJCV*, vol. 12, no. 1, pp. 5–16, January 1994.

[4] S. Baker and I. Matthews, "Lucas-kanade 20 years on: A unifying framework," *IJCV*, vol. 56, no. 3, pp. 221–255, 2004.

[5] J. Shi and C. Tomasi, "Good features to track," in *CVPR*, Seattle, June 1994.

[6] H.S. Sawhney and R. Kumar, "True multi-image alignment and its application to mosaicing and lens distortion correction," *PAMI*, vol. 21, no. 3, pp. 245–243, March 1999.

[7] M. Irani, "Multi-frame optical flow estimation using subspace constraints," in *ICCV*, Corfu, September 1999.