Tracking Feature Points: a New Algorithm

Dmitry Chetverikov and Judit Verestóy
Computer and Automation Research Institute
Image and Pattern Analysis Group
Budapest, Kende u.13-17, H-1111 Hungary
{mitya,judit}@leader.ipan.sztaki.hu

Abstract

A new algorithm is presented for feature point based motion tracking in long image sequences. Dynamic scenes with multiple, independently moving objects are considered in which feature points may temporarily disappear, enter and leave the view field. The existing approaches to feature point tracking [6, 3, 5, 4] have limited capabilities in handling incomplete trajectories, especially when the number of points and their speeds are large, and trajectory ambiguities are frequent. The proposed algorithm was designed to efficiently resolve these ambiguities.

1. Introduction

Feature point based motion tracking is a major domain of time-varying image analysis with numerous applications in object tracking, navigation, motion understanding and other areas. After extracting the feature points from the images, the feature points representing the same physical object point are identified in the whole image sequence by establishing correspondences between points of consecutive frames. A newly emerging application area of motion tracking is automatic, event-based video indexing for surveillance and scene monitoring. Typical events of interest in such tasks are appearance/disappearance, deposit/removal, entrance/exit, and motion/rest of objects.

In this paper, we use the following basic terms. Let F_k, $k = 1, 2, \ldots, M$, be the kth frame of the motion sequence, where M is the total number of the frames. When a point enters or leaves the view field in any frame $k \neq 1, M$, the trajectory is called *partial*. A trajectory is *broken* if the point temporarily disappears within the view field, and later reappears again. In this case, we speak of (temporary) *occlusion*, although feature points may, in general, become undetectable for other reasons as well. If a trajectory is broken, partial, or both, it is called *incomplete*. Entries, exits and temporal occlusions are referred to as *events*.

Algorithm	Self-init.	Occl.	Exit	Enter
Sethi & Jain [6]	$+$	$+$	$-$	$-$
Hwang [3]	$+$	$\pm$	$-$	$+$
Salari & Sethi [5]	$+$	$-$	$+$	$+$
Rang. & Shah [4]	$-$	$+$	$-$	$-$
Proposed	$+$	$\pm$	$+$	$+$

Table 1. Capabilities of tracking algorithms.

A survey of the existing feature point tracking approaches is given in [1]. Table 1 summarizes the capabilities of the algorithms to handle different events. The first column means operation without the prior knowledge of the initial correspondences, called *self-initialization*. The options ‘Occlusion’, ’Exit’ and ‘Enter’ refer to the corresponding events. ‘$+$’ is the presence, ‘$-$’ the absence of an option, while ‘$\pm$’ is a limited capability.

In this paper we propose a new tracking algorithm which performs well for dense sets of feature points in the presence of incomplete trajectories.

2. The new tracking algorithm

Consider a set of moving feature points. A point can only belong to a single trajectory. The points leave and enter the view field; temporary occlusions within the view field may also occur. Speeds can be significant: a point can cross the image within a sequence. A speed limit is known. The trajectories are smooth, that is, no drastic changes in velocity vectors are allowed. On the other hand, no consistent changes in velocities such as constant accelerations are assumed.

Let $P_{k,i}$ be a feature point in the kth frame. The matching algorithm has the following 3 steps: the initialization, the processing of the subsequent frames, and the post-processing. The initialization step operates on the first 3 frames and induces the tracking process. The points in F_2

are linked to the corresponding points in F_1 and F_3. Starting from F_3, a different matching procedure is applied. When all frames have been matched, a post-processing procedure is used to reconsider the points that temporarily disappeared and later re-appeared. This procedure attempts connecting the corresponding endpoints of the broken trajectories. These 3 steps are described below in more detail.

Initialization. The initializing step of the algorithm is illustrated in figure 1. In this figure, the filled circles are the moving points, the empty circles their locations projected to the neighboring frames. The solid lines show the competing trajectories. The dashed lines illustrate the formation of the search areas.

Consider an arbitrary point $P_{2,i}$ in the frame F_2 and try to find the corresponding points in F_1 and F_3. Using the maximum speed constraint, project the location of $P_{2,i}$ onto F_1 and F_3 and determine the *search areas* of $P_{2,i}$ in F_1 and F_3 as those regions that may contain the candidate matching points for $P_{2,i}$. Denote by $S^-_{k,i}$ and $S^+_{k,i}$ the search areas of $P_{k,i}$ in F_{k-1} and F_{k+1}, respectively.

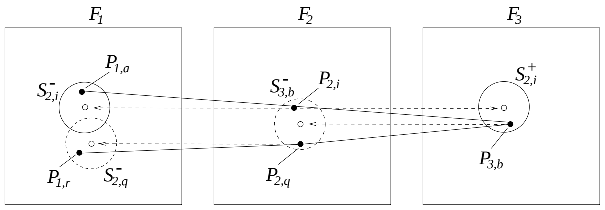

Figure 1. Initializing the tracking procedure.

Then consider all possible triplets of points $(P_{1,n}, P_{2,i}, P_{3,m})$, $P_{1,n} \in S^-_{2,i}$, $P_{3,m} \in S^+_{2,i}$, that contain the point $P_{2,i}$. Find the triplet $(P_{1,a}, P_{2,i}, P_{3,b})$ that minimizes for $k = 2$ the *cost function*

$$\begin{aligned} \delta(P_{k-1,n}, P_{k,i}, P_{k+1,m}) = \\ & w_1 \left(1 - \frac{\overline{P_{k-1,n}P_{k,i}} \cdot \overline{P_{k,i}P_{k+1,m}}}{\left\|\overline{P_{k-1,n}P_{k,i}}\right\| \cdot \left\|\overline{P_{k,i}P_{k+1,m}}\right\|} \right) \qquad (1) \\ + \quad & w_2 \left(1 - \frac{2\left[\left\|\overline{P_{k-1,n}P_{k,i}}\right\| \cdot \left\|\overline{P_{k,i}P_{k+1,m}}\right\|\right]^{\frac{1}{2}}}{\left\|\overline{P_{k-1,n}P_{k,i}}\right\| + \left\|\overline{P_{k,i}P_{k+1,m}}\right\|} \right) \end{aligned}$$

Here $\overline{P_{k-1,n}P_{k,i}}$ and $\overline{P_{k,i}P_{k+1,m}}$ are the vectors pointing from $P_{k-1,n}$ to $P_{k,i}$ and from $P_{k,i}$ to $P_{k+1,m}$, respectively. The first term in (1) penalizes changes in the direction, the second in the magnitude of the speed vector. The weights are $w_1 = 0.1$, $w_2 = 0.9$. This cost function is identical to the one used in [6].

Select the triplet $(P_{1,a}, P_{2,i}, P_{3,b})$ as the initial hypothesis and rank the remaining triplets according to their cost values. Test the initial hypothesis by scanning $S^-_{3,b}$, the search area of $P_{3,b}$ in F_2. Let $P_{2,q}$ be a point in $S^-_{3,b}$. (See figure 1.) If $S^-_{2,q}$ has a point $P_{1,r}$ such that $\delta(P_{1,r}, P_{2,q}, P_{3,b}) < \delta(P_{1,a}, P_{2,i}, P_{3,b})$, then the initial hypothesis is rejected and the second ranking hypothesis is considered and tested. Otherwise, the testing of the initial hypothesis proceeds with checking $P_{1,a}$ in the same way. If this check is also successful, $(P_{1,a}, P_{2,i}, P_{3,b})$ is output as the initial part of the trajectory of $P_{1,a}$. The correspondences established are not altered during further processing.

If all the hypotheses for $P_{2,i}$ have been rejected, this point is not linked to F_1 and F_3. It is still possible that $P_{2,i}$ will be linked to F_3 when the latter is processed. In such case $P_{2,i}$ *appears* and opens a trajectory.

When all points of F_2 have been processed, some points in the neighboring frames may remain unmatched. A point in F_1 that is not linked to any point in F_2 *disappears*, that is, either leaves the view field or temporarily disappears within the image, e.g., due to occlusion. An unmatched point in F_3 may later open a trajectory.

In the above description of the hypothesis testing procedure, the original hypothesis is rejected immediately when a 'stronger' triple $(P_{1,r}, P_{2,q}, P_{3,b})$ is found. However, $(P_{1,r}, P_{2,q}, P_{3,b})$ can itself be overruled by still another triple if the verification proceeds by testing $(P_{1,r}, P_{2,q}, P_{3,b})$. In this way, the original hypothesis can be restored. A deeper verification implies a longer sequence of simultaneous events whose probability decreases rapidly with the *verification depth* N_{ver}. The version described above is $N_{ver} = 1$. We use $N_{ver} = 2$ which is sufficient for point sets of reasonable density.

Processing the subsequent frames. In each frame, a point may have at most two links, a 'forward' one and a 'backward' one. A link indicates that the point is connected to a neighboring frame. Each link has a displacement vector assigned to it.

The matching procedure for $k > 2$ also operates with 3 consecutive frames. Consider the current frame $F_k, k > 2$. The previous frame F_{k-1} has just been processed. In F_{k-1}, the zero-link points (Z-points) are the appearing points whose potential correspondences to F_k are to be established. All the single-link points are connected to F_{k-2}, that is, they are backward-linked points abbreviated as B-points. These points have been marked as disappeared. They are only considered in the post-processing step. Most points usually have both links indicating continuous trajectories. In F_k, a point can have one backward link or no link at all. In F_{k+1}, all points are free. The procedure is illustrated in figure 2 where the lines show the symbolic backward and forward links. Only the dark points are considered in the hypothesis testing.

The feature points in F_k are processed similarly to the initializing step described above. The only difference is that the already established correspondences are used when available. They are not modified. Consequently, during the

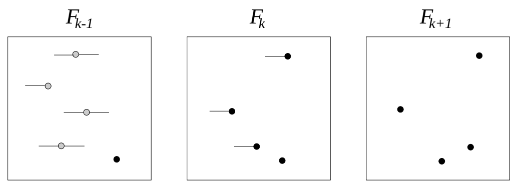

Figure 2. Processing frame $F_k, k > 2$.

hypothesis testing the B-points of F_k supply their previous displacements, while the Z-points are projected backwards onto F_{k-1} to find their candidate displacements. In F_{k-1}, the Z-points are only considered. These points may get linked to F_k and become forward-linked (F-) points.

This procedure provides a natural way to handle appearing and disappearing points, including the motion across the image border. The moving points establish their links in a competitive process that develops as the trajectories grow. When the final frame has been processed, the double-link points form the continuous trajectories. The B-points are the disappearing, the F-points the appearing points.

Post-processing of the broken trajectories. This procedure attempts connecting the broken trajectories. Currently, it is assumed that the maximum duration of an occlusion is 2 frames.

In figure 3, a broken trajectory is shown along with two alternative continuations and a separate, continuous trajectory. Consider a B-point P_e with the incoming velocity $\vec{v_e}$ and an F-point P_s with the outgoing velocity $\vec{v_s}$. A candidate occluded point is searched in the intersection of the two search areas S_e^+ and S_s^-. In this region, the point is found that minimizes the cumulative cost for the interpolated trajectory.

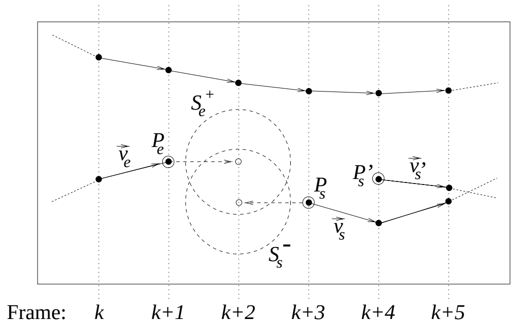

Figure 3. Post-processing.

A similar algorithm is then applied to P_e and P_s'. In this case, the search areas are extended as one steps forward, or backward, in time. Another candidate occluded path is obtained, this time with two interpolated points. Then the solution providing the least mean cost value is selected.

3. Tests and conclusions

A systematic study is in progress aimed at comparative performance evaluation of feature point tracking algorithms. Due to lack of space, this study is presented elsewhere [2]. A summary of the current test results is as follows.

Point density is a key parameter, with 40 trajectories per 200×200 size image being a critical value. For high densities exceeding this value, the performance of most algorithms starts to deteriorate faster.

The iterative algorithm [6] offers good trajectory tracking performance at very high computational cost. Completely wrong, 'oscillating' trajectories may occasionally appear.

The overall performance of [5] for low-to-medium densities is comparable to that of the proposed algorithm. The processing time of this iterative algorithm increases sharply with point density.

The algorithm [4] assumes that the initial correspondences are given. Even partial relaxation of this constraint may lead to unstable behavior, as a wrong initial link can propagate to distant trajectories. At low speeds, the procedure [4] can be applied to less smooth motion.

The proposed algorithm is best suitable for uncorrelated and incomplete trajectories, high speeds, and high point densities.

4. Acknowledgment

This work is supported by grants OTKA T026592 and INCO-COPERNICUS CT96-0742.

References

[1] D. Chetverikov and J. Verestóy. Motion tracking of dense feature point sets. In *Proc. 21st Workshop of the Austrian Pattern Recognition Group*, pages 233–242. Oldenbourg Verlag, 1997.

[2] D. Chetverikov and J. Verestóy. Selecting a feature point tracking algorithm for a computer vision application. In *Proc. 4th IEEE Workshop on Applications of Computer Vision*, 1998, submitted.

[3] V. Hwang. Tracking feature points in time-varying images using an opportunistic selection approach. *Pattern Recognition*, 22:247–256, 1989.

[4] K. Rangarajan and M. Shah. Establishing motion correspondence. *CVGIP: Image Understanding*, 54:56–73, 1991.

[5] V. Salari and I. K. Sethi. Feature point correspondence in the presence of occlusion. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 12:87–91, 1990.

[6] I. K. Sethi and R. Jain. Finding trajectories of feature points in a monocular image sequence. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 9:56–73, 1987.