Title:

Robust Feature Tracking

Authors:

Ben Galvin, Brendan McCane, and Kevin Novins

Email:

mccane@atlas.otago.ac.nz

Address:

Computer Science Department,
Univeristy of Otago,
Dunedin,
New Zealand.

Abstract:

We describe a novel corner tracking system which substantially reduces algorithmic errors, while still maintaining a low dropout rate. This is done by combining the results of a conventional correspondence-based corner tracker with a novel relaxation-based tracker. Machine learning techniques are applied to further improve performance of the individual trackers. Preliminary results with synthetic data indicate the system has half the dropout rate of a correspondence based system.

Robust Feature Tracking

B. Galvin B. McCane K. Novins

Computer Science Department,
University of Otago,
Dunedin, New Zealand.

Abstract

We describe a novel corner tracking system which substantially reduces algorithmic errors, while still maintaining a low dropout rate. This is done by combining the results of a conventional correspondence-based corner tracker with a novel relaxation-based tracker. Machine learning techniques are applied to further improve performance of the individual trackers. Preliminary results with synthetic data indicate the system has half the dropout rate of a correspondence based system.

1 Introduction

In principle, image motion contains a lot of information that should allow us to extract 3D structure [2], segment the scene [6], find occlusion boundaries and even recognize objects [4]. Unfortunately this potential remains largely unrealised. This is at least partly due to the difficulties in computing reliable motion estimates.

The conventional approach to this problem requires solving the correspondence problem: given a set of features in two adjacent frames, find the features which correspond to each other. This is a non trivial problem. Users are usually left with the choice of only being able to track a small number of very obvious corners, or making a significant number of correspondence errors.

Tracking errors in any feature-based system can arise from two different sources: measurement error, and algorithmic failure. Measurement errors tend to be well behaved in the sense that they have zero mean and are normally distributed. These errors can be accounted for by employing filtering techniques and propagating confidence values throughout the system - a technique employed in this paper.

Algorithmic failures will often have much more serious consequences on any subsequent processing because they are usually persistent and are unable to be eliminated with filtering. Correspondence mistakes are a typical example of an algorithmic failure. Such errors are unbounded and may cause the corner trajectories and any subsequent processes to behave chaotically. The errors produced by algorithmic failures are a major obstacle to the design of robust vision systems.

In this paper we present a corner tracking technique that substantially reduces algorithmic errors while still maintaining a low dropout rate. This is done by merging the predictions of a novel relaxation-based corner tracker, with that of a conventional correspondence-based corner tracker. Because the two trackers have quite different properties there are few situations in which neither tracker makes a correct prediction. Before merging the predictions, a classification scheme is applied to the best prediction of each tracker. This classification scheme is designed to detect and eliminate predictions that are likely to produce errors. Preliminary results indicate that the system has less than half the dropout rate of a correspondence-based tracking system.

2 Corner Tracking

The corner tracking system is described in Figure 2. Both tracking units output multiple-predictions, each with a calculated confidence value. The best predictions from each tracker are then evaluated for correctness using a simple machine learning scheme. The correct predictions are then merged together to produce the final result.

2.1 Notation

We use the notation commonly used in Kalman filter theory to describe the current state of a corner. A corner which has been tracked for at least one frame is describe with the system state vector x:

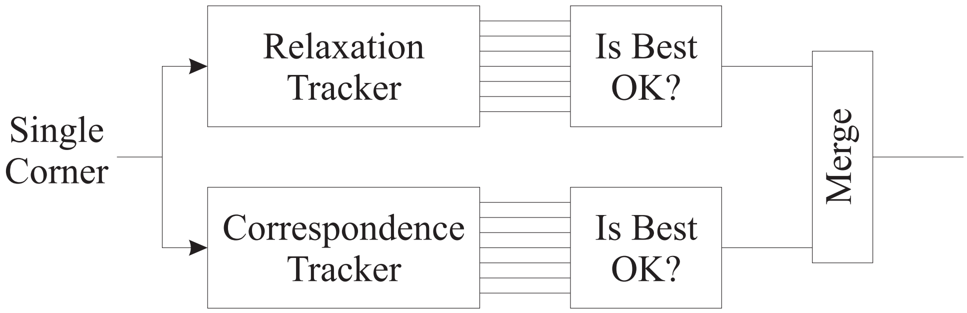


Figure 1: The corner tracking system. Note that this framework is easily extended to multiple tracker units.

$$x = \begin{bmatrix} p_x & p_y & v_x & v_y & g_x & g_y & c \end{bmatrix}^T \quad (1)$$

where p is the corner's position, v is the corner's velocity, g is the smoothed image gradient at p, c is the smoothed image color at p.

A *candidate* corner which hasn't been tracked yet is described with the vector z:

$$z = \begin{bmatrix} p_x & p_y & g_x & g_y & c \end{bmatrix}^T \quad (2)$$

2.2 Correspondence Tracker

Candidate corner positions in the current frame are found using the Harris Corner detector [1]. We define the 'difference' between a candidate corner in the current frame, z, and a corner tracked through previous frames, x, as

$$\begin{aligned} d(x,z) &= w_g \frac{|z_g - x_g|}{(|z_g| + |x_g| + 1)\sigma_g} + \\ & w_c \frac{|z_c - x_c|}{|x_g|\sigma_c} + \\ & w_p \frac{|z_p - x_p|}{\sigma_p} \end{aligned}$$

where σ denotes the estimated standard deviation, and w_g,w_c,w_p are constants. The first term in this equation represents the scaled difference between the gradient vectors at the original and candidate locations, scaled by the gradient's standard deviation. The standard deviation is used as a measure of reliability, and is extracted from the Kalman filter's covariance matrix. The second term represents the difference in image color between the original and candidate location, divided by the image gradient magnitude. Dividing by the image gradient normalises this terms response to varying corner strengths. Without this, strong corners may generate a very large response for very small positional errors. This term is also scaled by the standard deviation of the color component. The final term is the difference between the candidate corner location and predicted corner location (obtained from a linear motion model Kalman filter), scaled by the standard deviation in position.

The use of texture correlation was also considered however the extra computational burden was not worth the modest performance increase. As noted by Smith [5], there is little theoretical justification for the use of texture correlation since on average half of the texture surrounding a corner may belong to a background object .

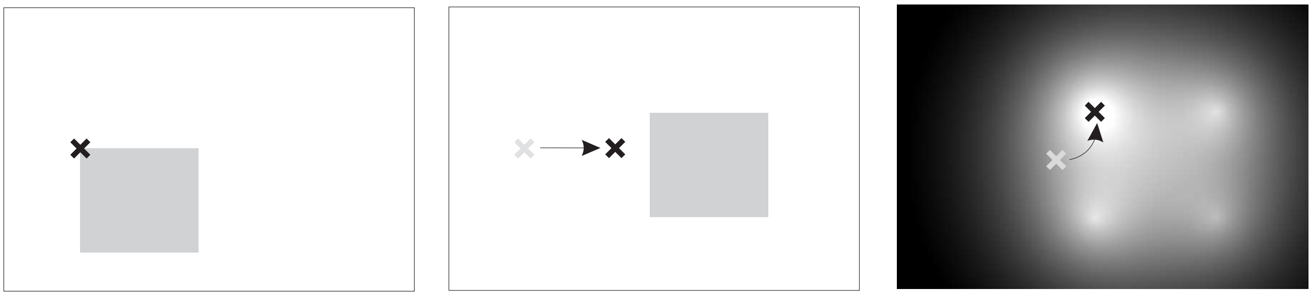
Our current implementation of the correspondence tracker is very simple. For each tracked corner, x, we add every candidate corner to the list of possible matches. For the correspondence tracker, we define the confidence that candidate corner z corresponds to the tracked corner x as

$$C(x,z) = \frac{1}{1 + d(x,z)}. \quad (3)$$

2.3 Relaxation Tracking

In this section we present a novel relaxation-based corner tracking technique. The process is analogous to the relaxation method used in the Active Contour framework of Kass et al.[3]. In fact the complete corner tracking system could be applied to contours, or any feature which can be tracked with both correspondence and relaxation techniques.

Corner relaxation is a very intuitive process: a corner is placed in its predicted position in the current frame, then repeatedly displaced in the direction which most rapidly decreases the error until a local



(a) A corner in the previous frame.

(b) The corner's position is updated using the current velocity estimate.

(c) The corner moves 'uphill' on its energy surface.

Figure 2: The corner relaxation algorithm.

minimum is reached. Positions are predicted using a Kalman filter and a linear motion model. This process is illustrated in Figure 2.3. We define the error caused by placing corner x at image position p as

$$e_x(p) = \frac{d(x, z(p))}{K(i)^\alpha} \quad (4)$$

where $z(p)$ is the corner observation vector generated at position p, $K(p)$ describes how 'corner-like' image location p is, and α is a small constant (3 in our examples).

The error has two components: the similarity between the original corner, x, and the new location, $z(p)$, and how 'corner-like' the new location is. Following Harris and Stephens [1], we define $K(x, y)$ as:

$$K(x, y) = det(A) - 0.04 * trace(A) \quad (5)$$

$$A(x, y) = \begin{bmatrix} \sum \frac{\partial I}{\partial x}\frac{\partial I}{\partial x} & \sum \frac{\partial I}{\partial x}\frac{\partial I}{\partial y} \\ \sum \frac{\partial I}{\partial y}\frac{\partial I}{\partial x} & \sum \frac{\partial I}{\partial y}\frac{\partial I}{\partial y} \end{bmatrix} \quad (6)$$

where $\frac{\partial I}{\partial x}$ and $\frac{\partial I}{\partial y}$ are the spatial image derivatives, and $\sum$ indicates the sum of values over a local region (a 5x5 region in our examples). An example energy field is shown in Figure 2(c).

The minimization process is very simple. We find the neighbor of p with the minimum $e_x(p)$ value, set p to this neighbor, and repeat this process until we reach a local minimum. If too many iterations occur the corner is not tracked, and a drop-out occurs.

To reduce errors further, we run this relaxation procedure from 4 locations in the neighborhood of the expected position of the corner (in our system these 'seed' points were 4 pixels from the expected corner position). If all of the relaxation procedures converge to the same position we can be very confident that we have the correct result. We quantify this by expressing the confidence that candidate corner z matches tracked corner x as

$$C(x, z) = \frac{1}{1 + d(x, z)}(n/N)^2 \quad (7)$$

where n is the number which converged to z, and N is the total number of relaxation procedures.

3 Classifying and Merging the Predictions

At this stage both tracking units have produced a list of candidate corners. In order to merge these predictions we require each tracker to evaluate its best candidate and make a binary decision about its correctness. If both the best relaxation candidate and the best correspondence candidate appear to be correct then we return the average of both. If on the other hand, only one of the candidates appears to be correct, we return that. If none appear to be correct, we have to accept a dropout.

The remainder of this section describes how the best prediction from each tracker is evaluated for correctness. We will denote the candidate corner with the highest confidence produced by the relaxation tracker as T^r_{best} and similarly, T^c_{best}, for the correspondence tracker.

First we will address the relaxation tracker's best prediction. We use three properties to determine if

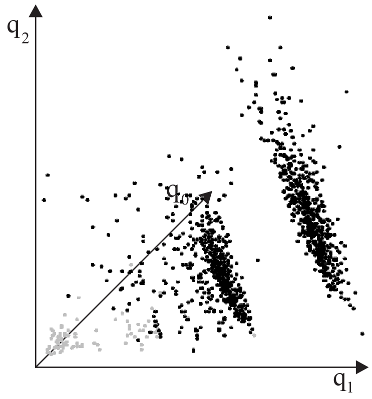


Figure 3: Scatter diagram of candidate corner attributes. Black indicates a correct candidate corner, gray indicates an incorrect candidate corner.

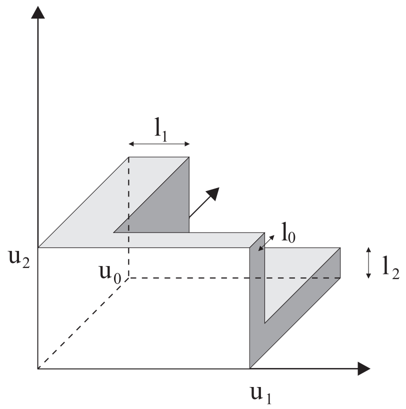


Figure 4: An example classification boundary. The parameters are found by training on ground-truth data.

T^r_{best} is likely to be correct. These properties are then passed to a simple classification scheme to determine the likelihood of an incorrect prediction.

The first property is that T^r_{best} should have a high confidence value:

$$q_0 = C(x, T^r_{best}) \tag{8}$$

The second property favors candidate's which are unique. We measure uniqueness by the ratio of our confidence in T^r_{best} to the sum of our confidence in all candidate corners:

$$q_1 = C(x, T^r_{best}) / \sum_i C(x, T^r_i) \tag{9}$$

where T^r_i is the ith candidate corner produced by the relaxation tracker. A q_1 of 1.0 indicates a unique solution, a q_1 close to 0 indicates that there were many candidate corners which were almost as good as T^r_{best}.

The final property favors predictions which are in similar locations to the best prediction of the correspondence tracker, T^c_{best}. We represent this as the distance between them, scaled between 0 and 1:

$$q_2 = 1/(|T^r_{best} - T^c_{best}| + 1) \tag{10}$$

A scatter-plot of these attributes generated from a synthetic sequence is shown in Figure 3. The incorrect predictions are clustered quite densely near the origin, and to a lesser extent along the axes.

Deciding if T^r_{best} is correct is now a matter of classifying the attribute vector q. We have experimented with a number of classification boundaries, however we have had the most success with this one: the candidate corner T^r_{best} is accepted if and only if

$$\begin{array}{l} (q_0 < u_0 \;\; AND \;\; q_1 < u_1 \;\; AND \;\; q_2 < u_2) \;\; AND \\ (q_0 < l_0 \quad OR \quad q_1 < l_1 \quad OR \quad q_2 < l_2) \end{array}$$

This corresponds to the classification boundary shown in Figure 4. The boundary parameters u and l are found by training the system on ground-truth data. This is done by minimizing a function of the number of dropouts and errors with respect to u and l. To avoid minimizing over 6 dimensions, we minimize with respect to u first, then fix u and minimize with respect to l.

To predict whether the best correspondence candidate corner, T^c_{best}, is correct, we use the properties described above, in conjunction with one more property: T^c_{best} should not violate the one-to-one correspondence constraint. We enforce this by penalizing the corner if any of the other tracked corners also match T^c_{best} with a high confidence value:

$$p_3 = C(x, T^c_{best}) / \sum_x C(x, T^c_{best}) \tag{11}$$

The resulting 4 dimensional vector can be classified by extending the boundary equation above to 4 dimensions.

The corners extracted by the Harris detector which were not used, are initialized as 'new corners' to be tracked through subsequent frames. This allows the system to continually generate and track new corners.

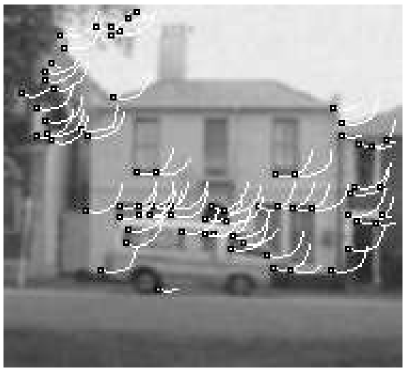


Figure 5: An example frame from the synthetic sequence. Corner trajectories are overlaid in white.

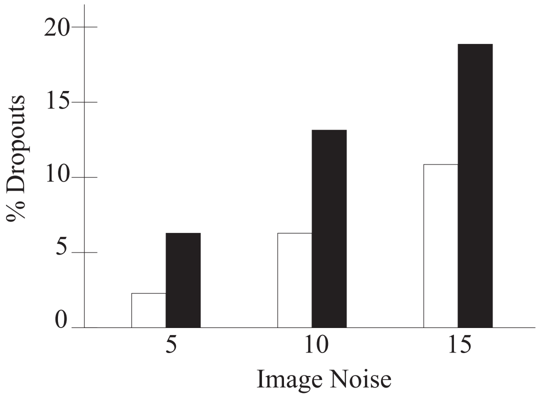


Figure 6: Dropout rates for the new corner tracking system (white) versus a conventional correspondence tracker (black). Algorithm parameters were selected to give an error rate for both algorithms of 0.5%. The sequence contained 20 frames, with an average of 60 tracked corners per frame.

4 Results

In order to quantify the performance of the new algorithm we have generated a simple synthetic image sequence. This was done by subjecting a real image to constant circular motion in the plane (so each corner follows a circular path of equal radius). Simulated Gaussian image noise was also added to each frame. A frame from the sequence with corner trajectories overlaid is displayed in Figure 5.

The dropout rates are shown in Figure 6. On average the new algorithm has at least half the dropout rate of a conventional correspondence tracker. Our implementation takes 5 seconds per frame on a Pentium II 400. Over half of this time is taken calculating the corner field and performing convolutions.

5 Conclusion

We have presented a novel corner tracking system which has half the dropout rate of a conventional correspondence-based corner tracker with the same error.

This performance was achieved by combining the results of a conventional correspondence-based corner tracker with the results of a novel relaxation-based tracker. A simple machine-learning technique was also applied to the best prediction from each tracker to remove incorrect predictions.

The efficient extraction of accurate low-level information remains a significant obstacle to the creation of stable and useful vision systems. Halving the dropout rate has two substantial benefits: i) we can track corners potentially for twice as many frames - substantially reducing error in motion prediction, and (ii) we can track more corners than before, giving a more complete description of the scene motion.

We are continuing work in this area. In particular we intend to perform more rigorous testing of the system, investigate the use of alternate attributes for the classification scheme, and using higher order features.

References

[1] C. Harris and M. Stephens. A combined corner and edge detector. In *Proceedings 4th Alvey Vision Conference (AVC88)*, pages 147–151, 1988.

[2] T.S. huang, J. Weng, and N. Ahuja. 3d motion from image sequences: Modeling, understanding, and prediction. In *Proceedings of the IEEE Workshop on Motion*, 1986.

[3] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, pages 321–331, 1988.

[4] M.K. Leung and T.S. Huang. An integrated approach to 3-d motion analysis and object recognition. *Proceedings of the IEEE on Pattern Matching and Machine Intelligence*, 13(10):1075–1097, October 1991.

[5] S.M. Smith and J.M. Brady. Asset-2: Real-time motion segmentation and shape tracking. *Proceedings of the IEEE on Pattern Matching and Machine Intelligence*, 17(8):814–820, 1995.

[6] J. Wang and E. Adelson. Representing moving images with layers. *IEEE Transaction on Image Processing*, 3(5):625–638, 1994.