IEEE Conference on Computer
Vision and Pattern Recognition
(CVPR94) Seattle, June 1994

Good Features to Track

Jianbo Shi
Computer Science Department
Cornell University
Ithaca, NY 14853

Carlo Tomasi
Computer Science Department
Stanford University
Stanford, CA 94305

Abstract

No feature-based vision system can work unless good features can be identified and tracked from frame to frame. Although tracking itself is by and large a solved problem, selecting features that can be tracked well and correspond to physical points in the world is still hard. We propose a feature selection criterion that is optimal by construction because it is based on how the tracker works, and a feature monitoring method that can detect occlusions, disocclusions, and features that do not correspond to points in the world. These methods are based on a new tracking algorithm that extends previous Newton-Raphson style search methods to work under affine image transformations. We test performance with several simulations and experiments.

1 Introduction

Is feature tracking a solved problem? The extensive studies of image correlation [4], [3], [15], [18], [7], [17] and sum-of-squared-difference (SSD) methods [2], [1] show that all the basics are in place. With small inter-frame displacements, a window can be tracked by optimizing some matching criterion with respect to translation [10], [1] and linear image deformation [6], [8], [11], possibly with adaptive window size[14]. Feature windows can be *selected* based on some measure of texturedness or cornerness, such as a high standard deviation in the spatial intensity profile [13], the presence of zero crossings of the Laplacian of the image intensity [12], and corners [9], [5]. Yet, even a region rich in texture can be poor. For instance, it can straddle a depth discontinuity or the boundary of a reflection highlight on a glossy surface. In either case, the window is not attached to a fixed point in the world, making that feature useless or even harmful to most structure-from-motion algorithms. Furthermore, even good features can become occluded, and trackers often blissfully drift away from their original target when this occurs. No feature-based vision system can be claimed to really work until these issues have been settled.

In this paper we show how to monitor the quality of image features during tracking by using a measure of feature *dissimilarity* that quantifies the change of appearance of a feature between the first and the current frame. The idea is straightforward: dissimilarity is the feature's rms residue between the first and the current frame, and when dissimilarity grows too large the feature should be abandoned. However, in this paper we make two main contributions to this problem. First, we provide experimental evidence that pure translation is not an adequate model for image motion when measuring dissimilarity, but affine image changes, that is, linear warping and translation, are adequate. Second, we propose a numerically sound and efficient way of determining affine changes by a Newton-Raphson stile minimization procedure, in the style of what Lucas and Kanade [10] do for the pure translation model. In addition, we propose a more principled way to select features than the more traditional "interest" or "cornerness" measures. Specifically, we show that features with good texture properties can be defined by optimizing the tracker's accuracy. In other words, the right features are exactly those that make the tracker work best. Finally, we submit that using two models of image motion is better than using one. In fact, translation gives more reliable results than affine changes when the inter-frame camera translation is small, but affine changes are necessary to compare distant frames to determine dissimilarity. We define these two models in the next section.

2 Two Models of Image Motion

As the camera moves, the patterns of image intensities change in a complex way. However, away from

[0]This research was supported by the National Science Foundation under contract IRI-9201751

occluding boundaries and near surface markings, these changes can often be described as image *motion*:

$$I(x, y, t+\tau) = I(x-\xi(x, y, t, \tau), y-\eta(x, y, t, \tau)) \ . \quad (1)$$

Thus, a later image taken at time $t+\tau$ can be obtained by moving every point in the current image, taken at time t, by a suitable amount. The amount of motion $\delta = (\xi, \eta)$ is called the *displacement* of the point at $\mathbf{x} = (x, y)$.

The displacement vector δ is a function of the image position $\mathbf{x}$, and variations in δ are often noticeable even within the small windows used for tracking. It then makes little sense to speak of "the" displacement of a feature window, since there are different displacements within the same window. An *affine motion field* is a better representation:

$$\delta = D\mathbf{x} + \mathbf{d}$$

where

$$D = \begin{bmatrix} d_{xx} & d_{xy} \\ d_{yx} & d_{yy} \end{bmatrix}$$

is a deformation matrix, and $\mathbf{d}$ is the translation of the feature window's center. The image coordinates $\mathbf{x}$ are measured with respect to the window's center. Then, a point $\mathbf{x}$ in the first image I moves to point $A\mathbf{x} + \mathbf{d}$ in the second image J, where $A = \mathbf{1} + D$ and $\mathbf{1}$ is the 2×2 identity matrix:

$$J(A\mathbf{x} + \mathbf{d}) = I(\mathbf{x}) \ . \quad (2)$$

Given two images I and J and a window in image I, tracking means determining the six parameters that appear in the deformation matrix D and displacement vector $\mathbf{d}$. The quality of this estimate depends on the size of the feature window, the texturedness of the image within it, and the amount of camera motion between frames. When the window is small, the matrix D is harder to estimate, because the variations of motion within it are smaller and therefore less reliable. However, smaller windows are in general preferable for tracking because they are less likely to straddle a depth discontinuity. For this reason, a *pure translation* model is preferable during tracking, where the deformation matrix D is assumed to be zero:

$$\delta = \mathbf{d} \ .$$

The experiments in sections 6 and 7 show that the best combination of these two motion models is pure translation for tracking, because of its higher reliability and accuracy over the small inter-frame motion of the camera, and affine motion for comparing features between the first and the current frame in order to monitor their quality. In order to address these issues quantitatively, however, we first need to introduce our tracking method.

3 Computing Image Motion

Because of image noise and because the affine motion model is not perfect, equation (2) is in general not satisfied exactly. The problem of determining the motion parameters is then that of finding the A and $\mathbf{d}$ that minimize the *dissimilarity*

$$\epsilon = \int\int_W \left[J(A\mathbf{x} + \mathbf{d}) - I(\mathbf{x})\right]^2 w(\mathbf{x}) \, d\mathbf{x} \quad (3)$$

where W is the given feature window and $w(\mathbf{x})$ is a weighting function. In the simplest case, $w(\mathbf{x}) = 1$. Alternatively, w could be a Gaussian-like function to emphasize the central area of the window. Under pure translation, the matrix A is constrained to be equal to the identity matrix. To minimize the residual (3), we differentiate it with respect to the unknown entries of the deformation matrix D and the displacement vector $\mathbf{d}$ and set the result to zero. We then linearize the resulting system by the truncated Taylor expansion

$$J(A\mathbf{x} + \mathbf{d}) = J(\mathbf{x}) + \mathbf{g}^T(\mathbf{u}) \ . \quad (4)$$

This yields (see [16]) the following linear 6×6 system:

$$T\mathbf{z} = \mathbf{a} \quad (5)$$

where $\mathbf{z}^T = \begin{bmatrix} d_{xx} & d_{yx} & d_{xy} & d_{yy} & d_x & d_y \end{bmatrix}$ collects the entries of the deformation D and displacement $\mathbf{d}$, the error vector

$$\mathbf{a} = \int\int_W [I(\mathbf{x}) - J(\mathbf{x})] \begin{bmatrix} xg_x \\ xg_y \\ yg_x \\ yg_y \\ g_x \\ g_y \end{bmatrix} w \, d\mathbf{x}$$

depends on the difference between the two images, and the 6×6 matrix T, which can be computed from one image, can be written as

$$T = \int\int_W \begin{bmatrix} U & V \\ V^T & Z \end{bmatrix} w \, d\mathbf{x} \quad (6)$$

where

$$U = \begin{bmatrix} x^2 g_x^2 & x^2 g_x g_y & xy g_x^2 & xy g_x g_y \\ x^2 g_x g_y & x^2 g_y^2 & xy g_x g_y & xy g_y^2 \\ xy g_x^2 & xy g_x g_y & y^2 g_x^2 & y^2 g_x g_y \\ xy g_x g_y & xy g_y^2 & y^2 g_x g_y & y^2 g_y^2 \end{bmatrix}$$

$$V^T = \left[\begin{array}{cccc} xg_x^2 & xg_xg_y & yg_x^2 & yg_xg_y \\ xg_xg_y & xg_y^2 & yg_xg_y & yg_y^2 \end{array} \right]$$

$$Z = \left[\begin{array}{cc} g_x^2 & g_xg_y \\ g_xg_y & g_y^2 \end{array} \right] .$$

Even when affine motion is a good model, equation 5 is only approximately satisfied, because of the linearization of equation (4). However, the correct affine change can be found by using equation 5 iteratively in a Newton-Raphson style minimization [16].

During tracking, the affine deformation D of the feature window is likely to be small, since motion between adjacent frames must be small in the first place for tracking to work at all. It is then safer to set D to the zero matrix. In fact, attempting to determine deformation parameters in this situation is not only useless but can lead to poor displacement solutions: in fact, the deformation D and the displacement $\mathbf{d}$ interact through the 4×2 matrix V of equation (6), and any error in D would cause errors in $\mathbf{d}$. Consequently, when the goal is to determine $\mathbf{d}$, the smaller system

$$Z\mathbf{d} = \mathbf{e} \tag{7}$$

should be solved, where $\mathbf{e}$ collects the last two entries of the vector $\mathbf{a}$ of equation (5).

When monitoring features for dissimilarities in their appearance between the first and the current frame, on the other hand, the full affine motion system (5) should be solved. In fact, motion is now too large to be described well by the pure translation model. Furthermore, in determining dissimilarity, the whole transformation between the two windows is of interest, and a precise displacement is less critical, so it is acceptable for D and $\mathbf{d}$ to interact to some extent through the matrix V.

In the next two sections we discuss these issues in more detail: first we determine when system (7) yields a good displacement measurement (section 4) and then we see when equation (5) can be used reliably to monitor a feature's quality (section 5).

4 Texturedness

Regardless of the method used for tracking, not all parts of an image contain complete motion information (the *aperture problem*): for instance, only the vertical component of motion can be determined for a horizontal intensity edge. To overcome this difficulty, researchers have proposed to track corners, or windows with a high spatial frequency content, or regions where some mix of second-order derivatives is sufficiently high. However, there are two problems with these "interest operators". First, they are often based on a preconceived and arbitrary idea of what a good window looks like. The resulting features may be intuitive, but are not guaranteed to be the best for the tracking algorithm to produce good results. Second, "interest operators" have been usually defined for the pure translation model of section 2, and the underlying concept are hard to extend to affine motion.

In this paper, we propose a more principled definition of feature quality. With the proposed definition, a good feature is one that can be tracked well, so that the selection criterion is optimal by construction.

We can track a window from frame to frame if system 7 represents good measurements, and if it can be solved reliably. Consequently, the symmetric 2×2 matrix Z of the system must be both above the image noise level and well-conditioned. The noise requirement implies that both eigenvalues of Z must be large, while the conditioning requirement means that they cannot differ by several orders of magnitude. Two small eigenvalues mean a roughly constant intensity profile within a window. A large and a small eigenvalue correspond to a unidirectional texture pattern. Two large eigenvalues can represent corners, salt-and-pepper textures, or any other pattern that can be tracked reliably.

In practice, when the smaller eigenvalue is sufficiently large to meet the noise criterion, the matrix Z is usually also well conditioned. In fact, the intensity variations in a window are bounded by the maximum allowable pixel value, so that the greater eigenvalue cannot be arbitrarily large. In conclusion, if the two eigenvalues of Z are λ_1 and λ_2, we accept a window if

$$\min(\lambda_1, \lambda_2) > \lambda , \tag{8}$$

where λ is a predefined threshold.

Similar considerations hold also when solving the full affine motion system (5) for the deformation D and displacement $\mathbf{d}$. However, an essential difference must be pointed out: deformations are used to determine whether the window in the first frame matches that in the current frame well enough during feature monitoring. Thus, the goal is *not* to determine deformation *per se*. Consequently, it does not matter if one component of deformation cannot be determined reliably. In fact, this means that that component does not affect the window substantially, and any value along this component will do in the comparison. In practice, the system (5) can be solved by computing the pseudo-inverse of T. Then, whenever some component is undetermined, the minimum norm solution is computed, that is, the solution with a zero deformation along the undetermined component(s).

5 Dissimilarity

A feature with a high texture content, as defined in the previous section, can still be a bad feature to track. For instance, in an image of a tree, a horizontal twig in the foreground can intersect a vertical twig in the background. This intersection occurs only in the image, not in the world, since the two twigs are at different depths. Any selection criterion would pick the intersection as a good feature to track, and yet there is no real world feature there to speak of. The measure of *dissimilarity* defined in equation (3) can often indicate that something is going wrong. Because of the potentially large number of frames through which a given feature can be tracked, the dissimilarity measure would not work well with a pure translation model. To illustrate this, consider figure 1, which shows three out of 21 frame details from Woody Allen's movie, *Manhattan*. The top row of figure 2 shows the results of tracking the traffic sign in this sequence.

Figure 1: Three frame details from Woody Allen's *Manhattan*. The details are from the 1st, 11th, and 21st frames of a subsequence from the movie.

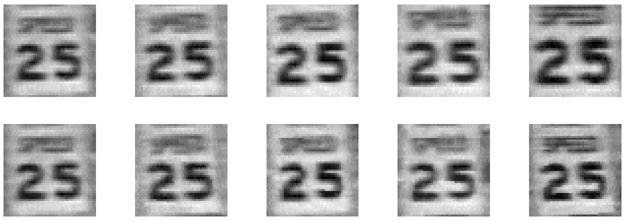

Figure 2: The traffic sign windows from frames 1,6,11,16,21 as tracked (top), and warped by the computed deformation matrices (bottom).

While the inter-frame changes are small enough for the pure translation tracker to work, the cumulative changes over 25 frames are rather large. In fact, the size of the sign increases by about 15 percent, and the dissimilarity measure (3) increases rather quickly with the frame number, as shown by the dashed and crossed line of figure 3. The solid and crossed line in the same figure shows the dissimilarity measure when also deformations are accounted for, that is, if the entire system (5) is solved for $\mathbf{z}$. This new measure of dissimilarity remains small and roughly constant. The bottom row of figure 2 shows the same windows as in the top row, but warped by the computed deformations. The deformations make the five windows virtually equal to each other.

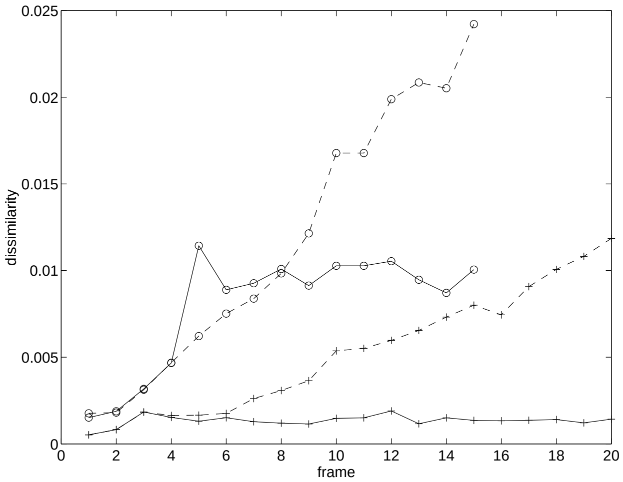

Figure 3: Pure translation (dashed) and affine motion (solid) dissimilarity measures for the window sequence of figure 1 (plusses) and 4 (circles).

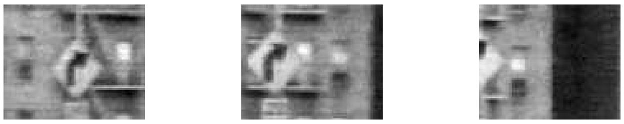

Figure 4: Three more frame details from *Manhattan*. The feature tracked is the bright window on the background, on the right of the traffic sign.

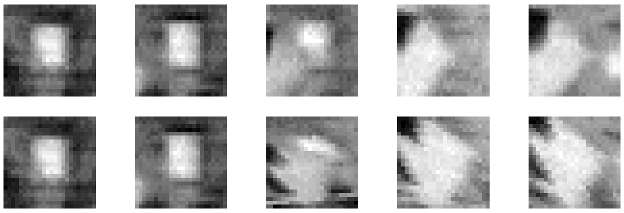

Figure 5: The bright window from figure 4 is occluded by the traffic sign in the middle frame (top). The bottom row shows the effects of warping by the computed deformation matrices.

The two circled curves in figure 3 refer to another feature from the same sequence, shown in figure 4. The top row of figure 5 shows the feature window through five frames. In the middle frame the traffic sign begins to occlude the original feature. The circled curves in figure 3 are the dissimilarity measures under affine motion (solid) and pure translation (dashed). The sharp jump in the affine motion curve around frame 4 indicates the occlusion. The bottom row of figure 5 shows that the deformation computation attempts to deform the traffic sign into a window.

6 Convergence

The simulations in this section show that when the affine motion model is correct our iterative tracking algorithm converges even when the starting point is far removed from the true solution. The first series of simulations are run on the four circular blobs shown in the leftmost column of figure 6. The three motions of table 1 are considered. To see their effects, compare the first and last column of figure 6. The images in the last column are the images warped, translated, and corrupted with random Gaussian noise with a standard deviation equal to 16 percent of the maximum image intensity. The images in the intermediate columns are the results of the deformations and translations to which the tracking algorithm subjects the images in the leftmost column after 4, 8, and 19 iterations, respectively. The algorithm works correctly, and makes the images in the fourth column of figure 6 as similar as possible to those in the fifth column.

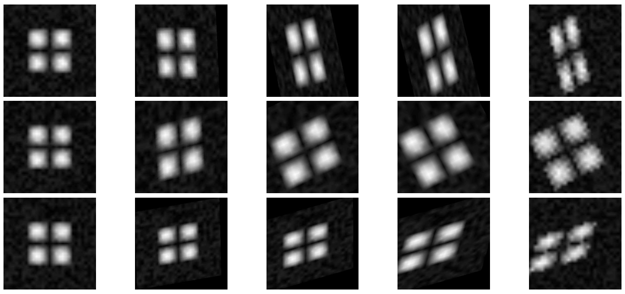

Figure 6: Original image (leftmost column) and warped, translated and noisy versions (rightmost column) for three different affine changes. The intermediate columns are the deformations computed by the tracker after 4,8,and 19 iterations.

Figure 7 plots the dissimilarity measure (as a fraction of the maximum image intensity), translation error (in pixels), and deformation error (Frobenius norm of the residual deformation matrix) as a function of the frame number (first three columns), as well as the intermediate displacements and deformations (last two columns). Deformations are represented in the fifth column of figure 7 by two vectors each, corresponding to the two columns of the transformation matrix $A = \mathbf{1} + D$. Table 1 shows the final numerical values.

Figure 8 shows a similar experiment with a more complex image (from MATLAB). Finally, figure 9 shows an attempt to match two completely different images: four blobs and a cross. The algorithm tries to do its best by aligning the blobs with the cross, but the dissimilarity (left plot at the bottom of figure 9) remains high throughout.

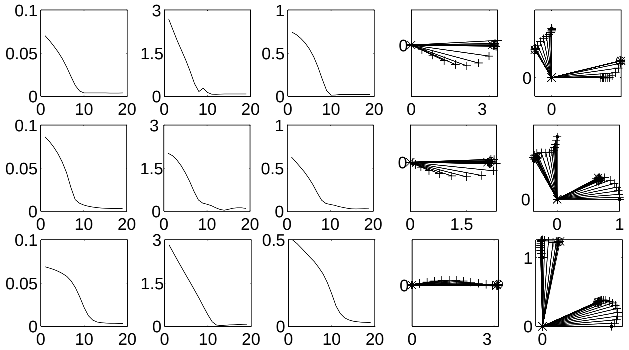

Figure 7: Dissimilarity (1^{st} column), displacement error (2^{nd}), and deformation error (3^{rd}) versus iteration number for figure 6. The last two columns are displacements and deformations computed during tracking, starting from zero. See text for units.

	True Deformation	Computed Deformation
1	$\begin{bmatrix} 1.409 & -0.342 \\ 0.342 & 0.563 \end{bmatrix}$	$\begin{bmatrix} 1.393 & -0.334 \\ 0.338 & 0.569 \end{bmatrix}$
2	$\begin{bmatrix} 0.658 & -0.342 \\ 0.342 & 0.658 \end{bmatrix}$	$\begin{bmatrix} 0.670 & -0.343 \\ 0.319 & 0.660 \end{bmatrix}$
3	$\begin{bmatrix} 0.809 & 0.253 \\ 0.342 & 1.232 \end{bmatrix}$	$\begin{bmatrix} 0.802 & 0.235 \\ 0.351 & 1.227 \end{bmatrix}$

	True Translation	Computed Translation
1	$\begin{bmatrix} 3 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 3.0785 \\ -0.0007 \end{bmatrix}$
2	$\begin{bmatrix} 2 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 2.0920 \\ 0.0155 \end{bmatrix}$
3	$\begin{bmatrix} 3 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 3.0591 \\ 0.0342 \end{bmatrix}$

Table 1: True and computed affine changes (in pixels) for the simulations of figure 6.

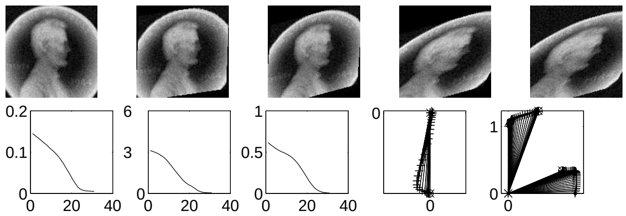

Figure 8: The penny at the top left is warped until it matches the transformed and noise-corrupted image at the top right. The bottom plots are as in figure 7.

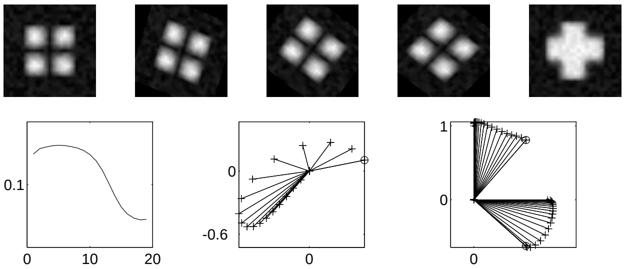

Figure 9: The blobs at the top left are warped as shown until they are as close as possible to the cross in the rightmost column. The bottom row shows dissimilarity, translation, and deformation versus iteration number.

7 Monitoring Features

This section presents some experiments with real images and shows how features can be monitored during tracking to detect potentially bad features. Figure 10 shows the first frame of a 26-frame sequence. A Pulnix camera equipped with a 16mm lens moves forward 2mm per frame. Because of the forward motion, features loom larger from frame to frame. The pure translation model is sufficient for inter-frame tracking but not to monitor features, as discussed below. Figure 11 displays the 102 features selected according to the criterion introduced in section 4. To limit the number of features and to use each portion of the image at most once, the constraint was imposed that no two feature windows can overlap in the first frame. Figure 12 shows the dissimilarity of each feature under the pure translation motion model, that is, with the deformation matrix D set to zero for all features. This dissimilarity is nearly useless for feature monitoring: except for features 58 and 89, all features have comparable dissimilarities, and no clean discrimination can be drawn between good and bad features.

From figure 13 we see that features 58 is at the boundary of the block with a letter **U** visible in the lower right-hand side of the figure. The feature window straddles the vertical dark edge of the block in the foreground as well as parts of the letters **Cra** in the word "Crayola" in the background. Six frames of this window are visible in the third row of figure 14. As the camera moves forward, the pure translation tracking stays on top of approximately the same part of the image. However, the gap between the vertical edge in the foreground and the letters in the background widens, and it becomes harder to warp the current window into the window in the first frame, thereby leading

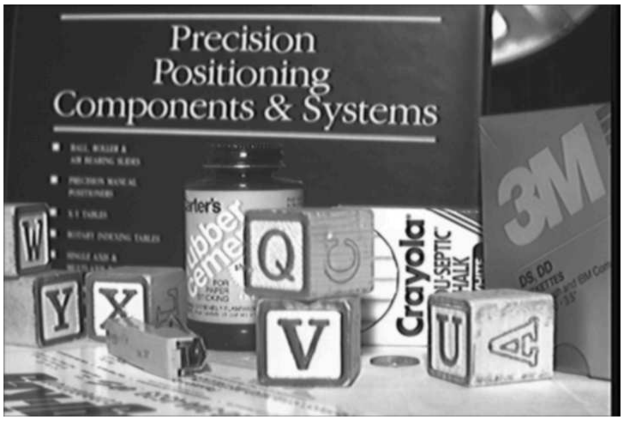

Figure 10: The first frame of a 26 frame sequence taken with a forward moving camera.

Figure 11: The features selected according to the texturedness criterion of section 4.

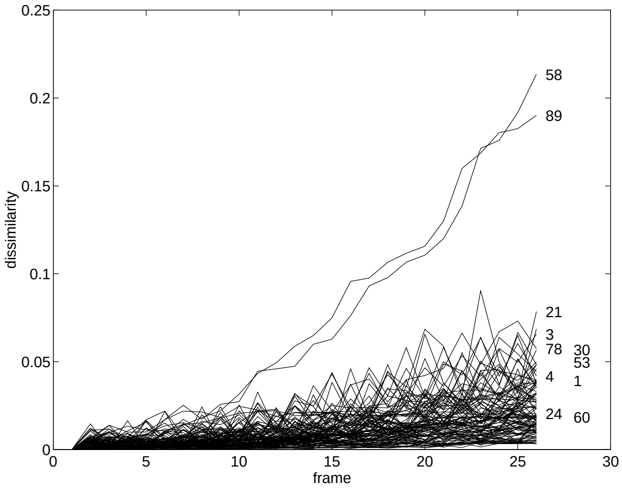

Figure 12: Pure translation dissimilarity for the features in figure 11. This dissimilarity is nearly useless for feature discrimination.

to the rising dissimilarity. The changes in feature 89 are seen even more easily. This feature is between the edge of the book in the background and a lamp partially visible behind it in the top right corner of figure 13. As the camera moves forward, the shape of the glossy reflection on the lamp shade changes as it becomes occluded (see the last row of figure 14).

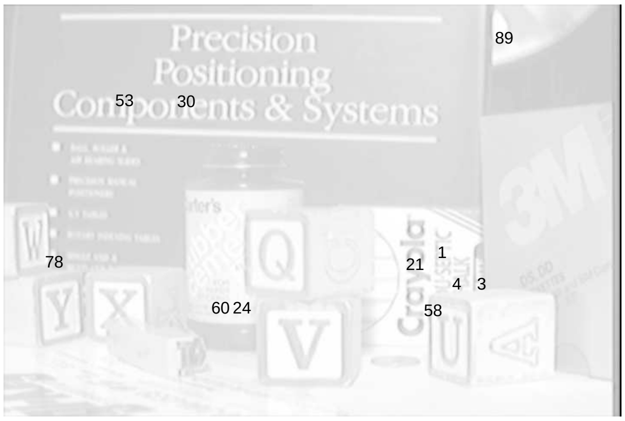

Figure 13: Labels of some of the features in figure 11.

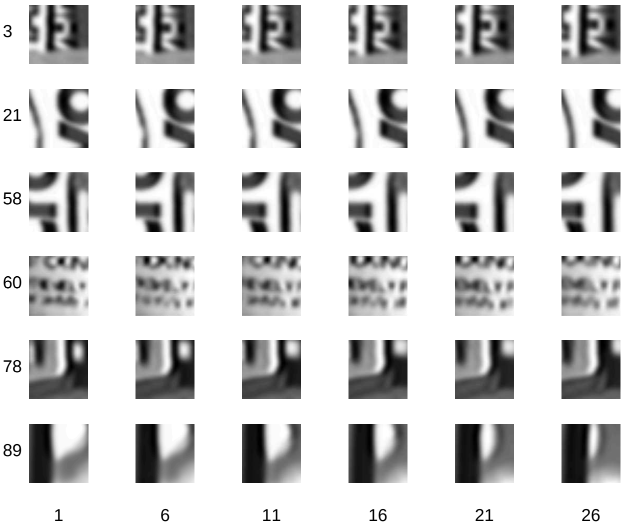

Figure 14: Six sample features through six sample frames.

Although these bad features would be detected because of their high dissimilarity, many other bad features would pass unnoticed. For instance, feature 3 in the lower right of figure 13 is affected by a substantial disocclusion of the lettering on the Crayola box by the **U** block as the camera moves forward, as well as a slight disocclusion by the "3M" box on the right (see the top row of figure 14). Yet with a pure translation model the dissimilarity of feature 3 is not substantially different from that of all the other features in figure 12. In fact, the looming caused by the camera's forward motion dominates, and reflects in the overall upward trend of the majority of curves in figure 12. Similar considerations hold, for instance, for features 78 (a disocclusion), 24 (an occlusion), and 4 (a disocclusion) labeled in figure 13.

Now compare the pure translation dissimilarity of figure 12 with the affine motion dissimilarity of figure 15. The thick stripe of curves at the bottom represents all good features, including features 1,21,30,53, labeled in figure 13. These four features are all good, being immune from occlusions or glossy reflections: 1 and 21 are lettering on the "Crayola" box (the second row of figure 14 shows feature 21 as an example), while features 30 and 53 are details of the large title on the book in the background (upper left in figure 13). The bad features 3,4,58,78,89, on the other hand, stand out very clearly in figure 15: discrimination is now possible.

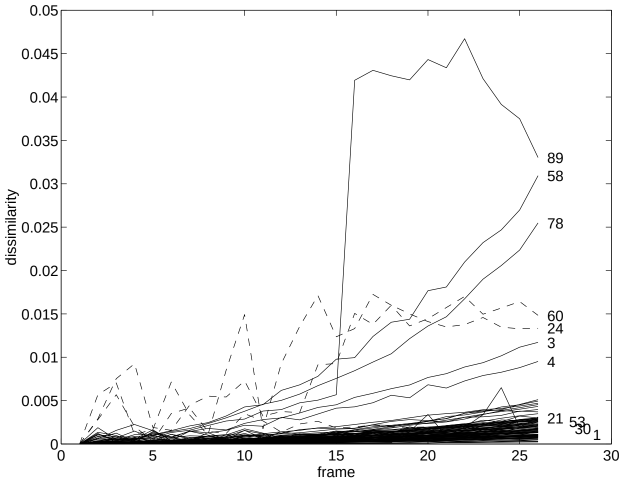

Figure 15: Affine motion dissimilarity for the features in figure 11. Notice the good discrimination between good and bad features. Dashed plots indicate aliasing (see text).

Features 24 and 60 deserve a special discussion, and are plotted with dashed lines in figure 15. These two features are lettering detail on the rubber cement bottle in the lower center of figure 13. The fourth row of figure 14 shows feature 60 as an example. Although feature 24 suffers an additional slight occlusion as the camera moves forward, these two features stand out from the very beginning, and their dissimilarity curves are very erratic throughout the sequence. This is because of aliasing: from the fourth row of figure 14, we see that feature 60 (and similarly feature 24) contains very small lettering, of size comparable to the image's pixel size (the feature window is 25×25 pixels). The matching between one frame and the next is haphazard, because the characters in the lettering are badly aliased. This behavior is not a problem: erratic dissimilarities indicate trouble, and the corresponding features ought to be abandoned.

8 Conclusion

In this paper, we have proposed a method for feature selection, a tracking algorithm based on a model of affine image changes, and a technique for monitoring features during tracking. Selection specifically maximizes the quality of tracking, and is therefore optimal by construction, as opposed to more *ad hoc* measures of texturedness. Monitoring is computationally inexpensive and sound, and helps discriminating between good and bad features based on a measure of dissimilarity that uses affine motion as the underlying image change model.

Of course, monitoring feature dissimilarity does not solve all the problems of tracking. In some situations, a bright spot on a glossy surface is a bad (that is, nonrigid) feature, but may change little over a long sequence: dissimilarity may not detect the problem. However, even in principle, not everything can be decided locally. Rigidity is not a local feature, so a local method cannot be expected to always detect its violation. On the other hand, many problems can indeed be discovered locally and these are the target of the investigation in this paper. Our experiments and simulations show that monitoring is indeed effective in realistic circumstances. A good discrimination at the beginning of the processing chain can reduce the remaining bad features to a few outliers, rather than leaving them an overwhelming majority. Outlier detection techniques at higher levels in the processing chain are then more likely to succeed.

References

[1] P. Anandan. A computational framework and an algorithm for the measurement of visual motion. *IJCV*, 2(3):283–310, 1989.

[2] P. J. Burt, C. Yen, and X. Xu. Local correlation measures for motion analysis: a comparative study. *IEEE CPRIP*, 269–274, 1982.

[3] C. Cafforio and F. Rocca. Methods for measuring small displacements in television images. *IEEE Trans.* IT-22:573–579, 1976.

[4] D. J. Connor and J. O. Limb. Properties of frame-difference signals generated by moving images. *IEEE Trans.* COM-22(10):1564–1575, 1974.

[5] L. Dreschler and H.-H. Nagel. Volumetric model and 3d trajectory of a moving car derived from monocular tv frame sequences of a street scene. *IJCAI*, 692–697, 1981.

[6] W. Förstner. Reliability analysis of parameter estimation in linear models with applications to mensuration problems in computer vision. *CVGIP*, 40:273–310, 1987.

[7] W. Förstner and A. Pertl. *Photogrammetric Standard Methods and Digital Image Matching Techniques for High Precision Surface Measurements.* Elsevier Science Pub., 1986.

[8] C. Fuh and P. Maragos. Motion displacement estimation using an affine model for matching. *Optical Engineering*, 30(7):881–887, 1991.

[9] L. Kitchen and A. Rosenfeld. Gray-level corner detection. TR, U. of Maryland, 1980.

[10] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. *IJCAI*, 1981.

[11] R. Manmatha and J. Oliensis. Extracting affine deformations from image patches - I: Finding scale and rotation. *CVPR*, 754–755, 1993.

[12] D. Marr, T. Poggio, and S. Ullman. Bandpass channels, zero-crossings, and early visual information processing. *JOSA*, 69:914–916, 1979.

[13] H. Moravec. *Obstacle avoidance and navigation in the real world by a seeing robot rover.* PhD, Stanford U., 1980.

[14] M. Okutomi and T. Kanade. A locally adaptive window for signal matching. *IJCV*, 7(2):143–162, 1992.

[15] T. W. Ryan, R. T. Gray, and B. R. Hunt. Prediction of correlation errors in stereo-pair images. *Optical Engineering*, 19(3):312–322, 1980.

[16] J. Shi and C. Tomasi. Good features to track. TR 93-1399, Cornell U., 1993.

[17] Qi Tian and Michael N. Huhns. Algorithms for subpixel registration. *CVGIP*, 35:220–233, 1986.

[18] G. A. Wood. Realities of automatic correlation problem. *Photogram. Eng. and Rem. Sens.*, 49:537–538, 1983.