

Tracking Feature Points

CS 763

References:

- ❑ Shi and Tomasi, “Good Features to Track”, Cornell University Technical Report, 1993.
- ❑ Tomasi and Kanade, “Detection and Tracking of Point Features”, Carnegie Mellon University Technical Report, 1991.

Problem statement

- Input: video sequence
- Output: A list of “good” feature points marked out in frame 1, and their positions in all subsequent video frames.

Problem statement: demo

<http://www.youtube.com/watch?v=pmKtNQphq1E>

<http://www.youtube.com/watch?v=Jw1CCR0tAcY>

Problems in feature tracking

- What is meant by “good” feature points? Why isn't every point a good feature point?
- Can the tracker make mistakes somewhere? If so, can we identify the mistake?
- What do you do when a feature point gets occluded? Or disappears?
- What do you do when a new object appears in the video?

Motion estimation: Model 1

- Consider $I(x,y,t+1) = I(x+u,y+v,t)$ where (u,v) = displacement at pixel (x,y) at time t .
- We want to estimate (u,v) .
- But estimating (u,v) at individual pixels is unreliable due to image noise and the aperture problem.
- So consider a small patch **W** of pixels around (x,y) . Estimate (and *assume*) a common (u,v) for the entire window.



Motion Estimation: Model 1

$$I(x, y, t + 1) = I(x + u, y + v, t) + n(x, y) = I(x, y, t) + uI_x + vI_y + n(x, y)$$

$$E(u, v) = \sum_{(x, y) \in W} (I(x, y, t + 1) - I(x, y, t) - uI_x - vI_y)^2$$

$$\frac{\partial E}{\partial u} = \sum_{(x, y) \in W} -2I_x(I_t - uI_x - vI_y) = 0$$

$$\frac{\partial E}{\partial v} = \sum_{(x, y) \in W} -2I_y(I_t - uI_x - vI_y) = 0$$

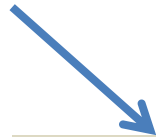
Taylor series about x and y assuming (u, v) to be small.

This is very similar to the equations for the Lucas-Kanade method for optical flow

$$\begin{pmatrix} \sum_{(x, y) \in W} I_x^2(x, y) & \sum_{(x, y) \in W} I_x(x, y)I_y(x, y) \\ \sum_{(x, y) \in W} I_x(x, y)I_y(x, y) & \sum_{(x, y) \in W} I_y^2(x, y) \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} \sum_{(x, y) \in W} I_x(x, y)I_t(x, y) \\ \sum_{(x, y) \in W} I_y(x, y)I_t(x, y) \end{pmatrix}$$

Motion Estimation : Model 1

$$\begin{pmatrix} \sum_{(x,y) \in W} I_x^2(x,y) & \sum_{(x,y) \in W} I_x(x,y)I_y(x,y) \\ \sum_{(x,y) \in W} I_x(x,y)I_y(x,y) & \sum_{(x,y) \in W} I_y^2(x,y) \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} \sum_{(x,y) \in W} I_x(x,y)I_t(x,y) \\ \sum_{(x,y) \in W} I_y(x,y)I_t(x,y) \end{pmatrix}$$



Local structure tensor:
denoted as **G**

$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} \sum_{(x,y) \in W} I_x^2(x,y) & \sum_{(x,y) \in W} I_x(x,y)I_y(x,y) \\ \sum_{(x,y) \in W} I_x(x,y)I_y(x,y) & \sum_{(x,y) \in W} I_y^2(x,y) \end{pmatrix}^{-1} \begin{pmatrix} \sum_{(x,y) \in W} I_x(x,y)I_t(x,y) \\ \sum_{(x,y) \in W} I_y(x,y)I_t(x,y) \end{pmatrix}$$

Motion estimation : Model 2

- The assumption of a single displacement/translation (u,v) for the whole window **W** can be unreasonable.
- Solution: Use an **affine** transformation model
 - each pixel in the patch now has its own displacement.

Motion Estimation : Model 2

$$\begin{pmatrix} x \\ y \end{pmatrix} \rightarrow \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} u \\ v \end{pmatrix} = \mathbf{A} \begin{pmatrix} x \\ y \end{pmatrix} + \mathbf{d}$$

$$\begin{aligned} I(x, y, t+1) &= I(A_{11}x + A_{12}y + u, A_{21}x + A_{22}y + v, t) = I(x + A_{11}x + A_{12}y - x + u, y + A_{21}x + A_{22}y - y + v, t) \\ &= I(x, y, t) + (D_{11}x + D_{12}y + u)I_x(x, y, t) + (D_{21}x + D_{22}y + v)I_y(x, y, t) + n(x, y) \end{aligned}$$

$$\mathbf{A} - \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \mathbf{D}$$

$$E(D, u, v) = \sum_{(x,y) \in W} (I_t(x, y, t) - (D_{11}x + D_{12}y + u)I_x - (D_{21}x + D_{22}y + v)I_y)^2$$

Taking Taylor series about (x, y, t)

$$\frac{\partial E}{\partial \mathbf{D}} = \mathbf{0}_{2 \times 2}, \quad \frac{\partial E}{\partial \mathbf{d}} = \mathbf{0}_{2 \times 1}$$

We drop the (x, y, t) on I_x , I_y and I_t henceforth only for the sake of brevity

Motion Estimation

$$\begin{aligned} I(x, y, t+1) &= I(A_{11}x + A_{12}y + u, A_{21}x + A_{22}y + v, t) \\ &= I(x, y, t) + (D_{11}x + D_{12}y + u)I_x(x, y, t) + (D_{21}x + D_{22}y + v)I_y(x, y, t) + n(x, y) \end{aligned}$$

$$\mathbf{A} - \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \mathbf{D}$$

$$E(\mathbf{D}, u, v) = \sum_{(x,y) \in W} (I(x, y, t+1) - I(x, y, t) - (D_{11}x + D_{12}y + u)I_x - (D_{21}x + D_{22}y + v)I_y)^2$$

$$\frac{\partial E}{\partial \mathbf{D}} = \mathbf{0}_{2 \times 2}, \quad \frac{\partial E}{\partial \mathbf{d}} = \mathbf{0}_{2 \times 1}$$

$$\sum_{(x,y) \in W} (I(x, y, t+1) - I(x, y, t) - (D_{11}x + D_{12}y + u)I_x - (D_{21}x + D_{22}y + v)I_y) \begin{pmatrix} I_x \\ I_y \end{pmatrix} (x \quad y) = \mathbf{0}_{2 \times 2}$$

$$\sum_{(x,y) \in W} (I(x, y, t+1) - I(x, y, t) - (D_{11}x + D_{12}y + u)I_x - (D_{21}x + D_{22}y + v)I_y) \begin{pmatrix} I_x \\ I_y \end{pmatrix} = \mathbf{0}_{2 \times 1}$$

Note:

$$\frac{\partial E}{\partial \mathbf{D}} = \begin{pmatrix} \frac{\partial E}{\partial D_{11}} & \frac{\partial E}{\partial D_{12}} \\ \frac{\partial E}{\partial D_{21}} & \frac{\partial E}{\partial D_{22}} \end{pmatrix}$$

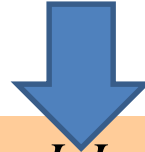
This is by the definition of derivative w.r.t. matrix \mathbf{D} or vector $\mathbf{d} = (u,v)$. The same definition is applicable to matrices or vectors of larger size.

$$\frac{\partial E}{\partial \mathbf{d}} = \begin{pmatrix} \frac{\partial E}{\partial u} \\ \frac{\partial E}{\partial v} \end{pmatrix}$$

Motion Estimation : Model 2

$$\sum_{(x,y) \in W} (I(x, y, t+1) - I(x, y, t) - (D_{11}x + D_{12}y + u)I_x - (D_{21}x + D_{22}y + v)I_y) \begin{pmatrix} I_x \\ I_y \end{pmatrix} \begin{pmatrix} x & y \end{pmatrix} = \mathbf{0}_{2 \times 2}$$

$$\sum_{(x,y) \in W} (I(x, y, t+1) - I(x, y, t) - (D_{11}x + D_{12}y + u)I_x - (D_{21}x + D_{22}y + v)I_y) \begin{pmatrix} I_x \\ I_y \end{pmatrix} = \mathbf{0}_{2 \times 1}$$



$$\sum_{(x,y) \in W} \begin{pmatrix} x^2 I_x^2 & x^2 I_x I_y & xy I_x^2 & xy I_x I_y & x I_x^2 & x I_x I_y \\ x^2 I_x I_y & x^2 I_y^2 & xy I_x I_y & xy I_y^2 & x I_x I_y & x I_y^2 \\ xy I_x^2 & xy I_x I_y & y^2 I_x^2 & y^2 I_x I_y & y I_x^2 & y I_x I_y \\ xy I_x I_y & xy I_y^2 & y^2 I_x I_y & y^2 I_y^2 & y I_x I_y & y I_y^2 \\ x I_x^2 & x I_x I_y & y^2 I_x^2 & y I_x I_y & I_x^2 & I_x I_y \\ x I_x I_y & x I_y^2 & y I_x I_y & y I_y^2 & I_x I_y & I_y^2 \end{pmatrix} \begin{pmatrix} D_{11} \\ D_{21} \\ D_{12} \\ D_{22} \\ u \\ v \end{pmatrix} = \sum_{(x,y) \in W} \begin{pmatrix} x I_x I_t \\ x I_y I_t \\ y I_x I_t \\ y I_y I_t \\ I_x I_t \\ I_y I_t \end{pmatrix}$$

This matrix is obtained by rearranging the equations in the blue box in the form $\mathbf{Cz} = \mathbf{b}$ where \mathbf{C} and \mathbf{b} are known vectors and \mathbf{z} is a vector of unknown values (in this case the elements of \mathbf{D} and \mathbf{d})



You can estimate \mathbf{D} and (\mathbf{u}, \mathbf{v}) by least squares/inverse

Motion Estimation: Model 2

- Model 2 has more degrees of freedom than Model 1 (6 versus 2).
- Model 2 is more general, but less robust to noise.
- Model 2 will need larger patches to work well. This increases the fraction of patches that straddle boundaries of different objects.
- Motion between consecutive frames must be small (necessary for tracking to work) – hence most of the entries of \mathbf{D} will be small.
- Hence model 1 is preferred for motion estimation between consecutive frames.
- Model 2 has another application in tracking!

Motion estimation: model 2 (iterative computation)

- The key equations rest upon the assumption of small motion.
- In some cases (example, if you are comparing patches from image frames at vastly different time instants), this assumption is invalid.
- In such cases, one has to determine the motion parameters (\mathbf{D} and \mathbf{d}) iteratively.
- You loop over the following steps until convergence: (a) estimate \mathbf{D} and \mathbf{d} to align window \mathbf{W}_1 with window \mathbf{W}_2 , (b) warp \mathbf{W}_1 as per \mathbf{D} and \mathbf{d} and go back to earlier step.

What's a good feature point?

- One that can be tracked well.
- A point whose surrounding patch has nice properties!

$$\begin{pmatrix} \sum_{(x,y) \in W} I_x^2(x,y) & \sum_{(x,y) \in W} I_x(x,y)I_y(x,y) \\ \sum_{(x,y) \in W} I_x(x,y)I_y(x,y) & \sum_{(x,y) \in W} I_y^2(x,y) \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} \sum_{(x,y) \in W} I_x(x,y)I_t(x,y) \\ \sum_{(x,y) \in W} I_y(x,y)I_t(x,y) \end{pmatrix}$$

- Structure tensor \mathbf{G} should be invertible.

What's a good feature point?

- Let the two eigenvalues of \mathbf{G} be λ_1, λ_2 ($\lambda_1 \geq \lambda_2$).
- If the patch \mathbf{W} has constant intensity, $\lambda_1 = \lambda_2 = 0$. \mathbf{G} is a null matrix.
- If the patch \mathbf{W} is noisy but otherwise has constant intensity, $\lambda_1 \approx 0, \lambda_2 \approx 0$. \mathbf{G} is ill-conditioned.
- If the patch has constant but large gradient, then $\lambda_1 \gg 0, \lambda_2 = 0$. \mathbf{G} has rank 1.
- If the patch has constant but large gradient and some noise, then $\lambda_1 \gg 0, \lambda_2 \approx 0$. \mathbf{G} has rank 2 but is ill-conditioned.

Code at:

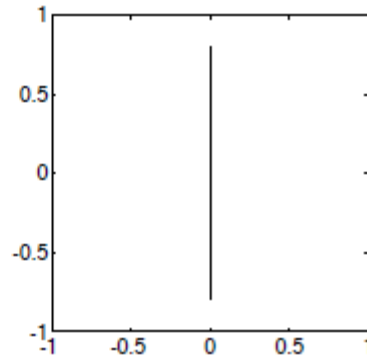
http://www.cse.iitb.ac.in/~ajitvr/CS763_Spring2015/eigvals_structure_tensor.m

What's a good feature point?

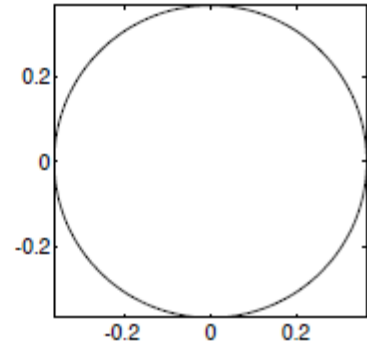
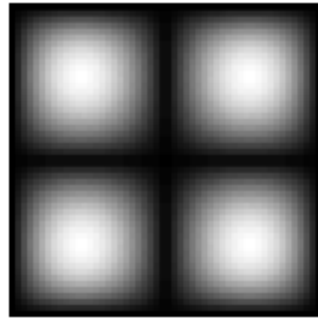
- If the patch contains a strong edge against a constant intensity background, then $\lambda_1 \gg 0$, $\lambda_2 \approx 0$. **G** has rank 2 but is ill-conditioned.
- If the patch is rich in texture with significant intensity, then $\lambda_1 \gg 0$, $\lambda_2 \gg 0$. This also occurs at corner points. **G has rank 2 and is well-conditioned. It allows for reliable motion estimate.**
- The minor eigen-value gives a good idea of the “goodness” of the feature point for tracking. Larger is better.

Code at:

http://www.cse.iitb.ac.in/~ajitvr/CS763_Spring2015/eigvals_structure_tensor.m



Ellipse with axes of width given by the eigenvalues λ_1, λ_2



Source of images (and in fact, most of this presentation):

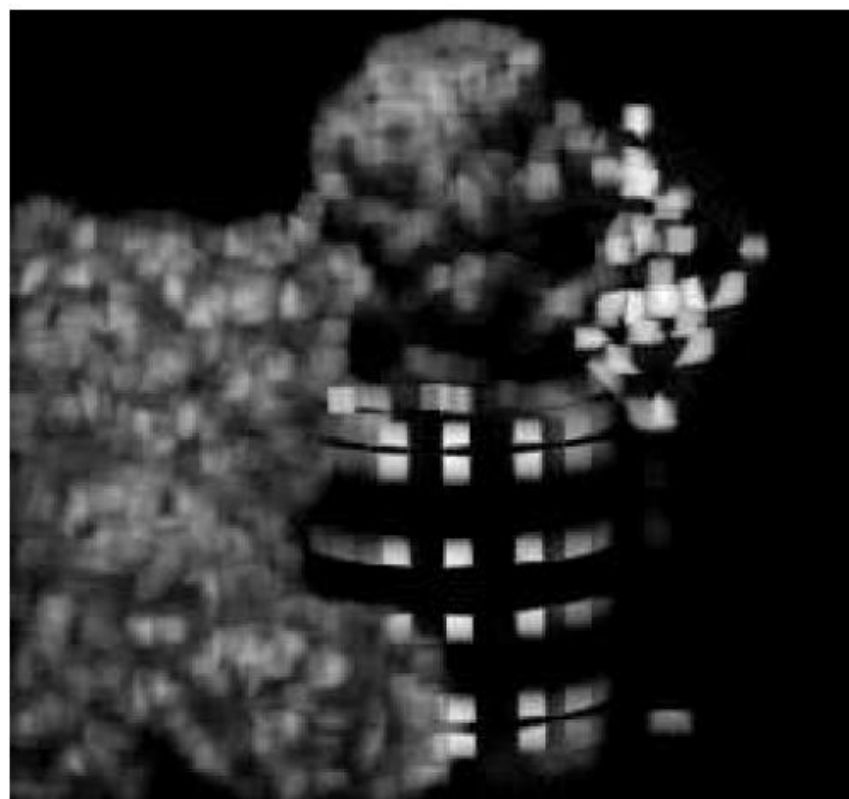
Shi and Tomasi, "Good features to track", Cornell University Technical Report, 1993

https://www.cs.duke.edu/~tomasi/papers/shi/TR_93-1399_Cornell.pdf

What's a good feature point?

- Pick feature points whose minor eigen-value exceeds some threshold τ .
- How to pick τ ?
- Take a training set consisting of uniform intensity patches and another of textured patches.
- The minor eigen-values will have a huge difference. That gives you a good estimate of τ .

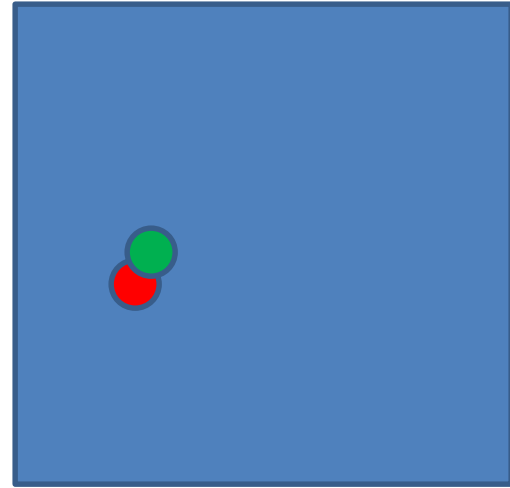
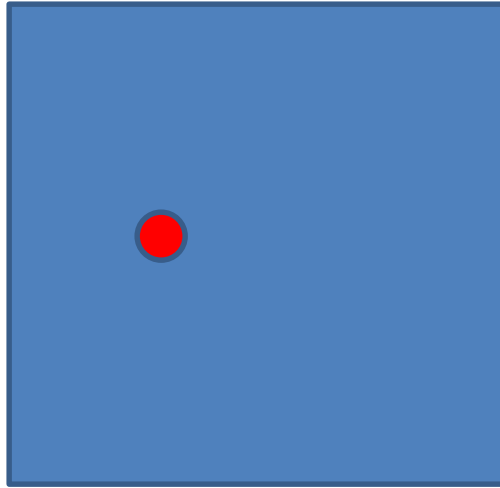
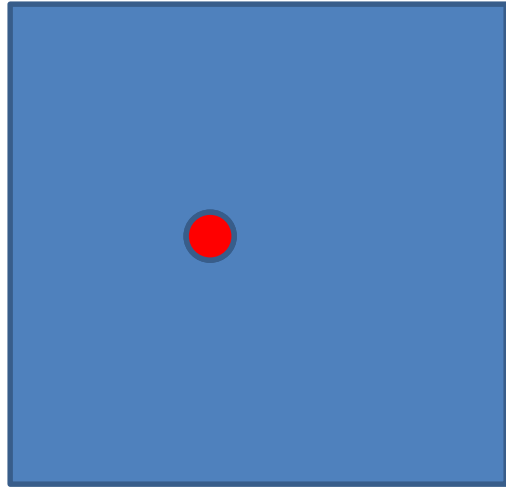






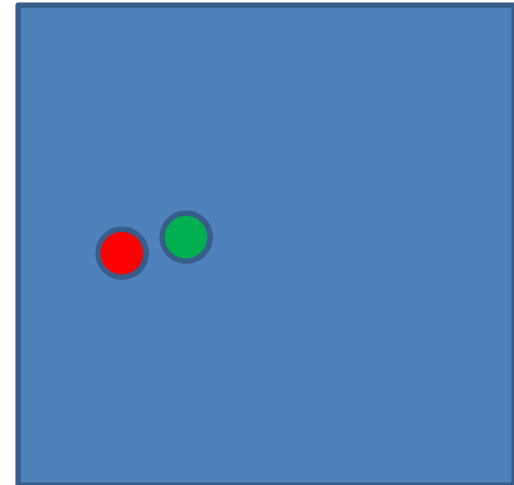
What's a good feature point? Criterion 2

- Feature points satisfying the minor eigen-value criterion may still be bad.
- Example: the corner point created by intersection of a moving object with a static one is a bad feature point! It cannot be tracked as it disappears in subsequent frames!
- How do you know the point was tracked correctly?

Problems in feature tracking



-  Ground truth position
-  Predicted position



What's a good feature point? Criterion 2

- Compute a dissimilarity measure between a patch containing the (tracked) feature point in the current frame (let's call it \mathbf{W}_c) and the original patch (let's call it \mathbf{W}_f) in the first frame.
- If the measure is small, then we have tracked the point well.
- If the measure is large, the tracking has failed, and this feature point should be abandoned.

What's a good feature point? Criterion 2

- Dissimilarity measure 1:
 - Find the best translation between \mathbf{W}_c and \mathbf{W}_f .
 - Find squared difference between \mathbf{W}_c and *translated* \mathbf{W}_f .
- Dissimilarity measure 2:
 - Find the best affine transformation between \mathbf{W}_c and \mathbf{W}_f .
 - Find squared difference between \mathbf{W}_c and *affine transformed* \mathbf{W}_f .

Synthetic Results: Motion estimation

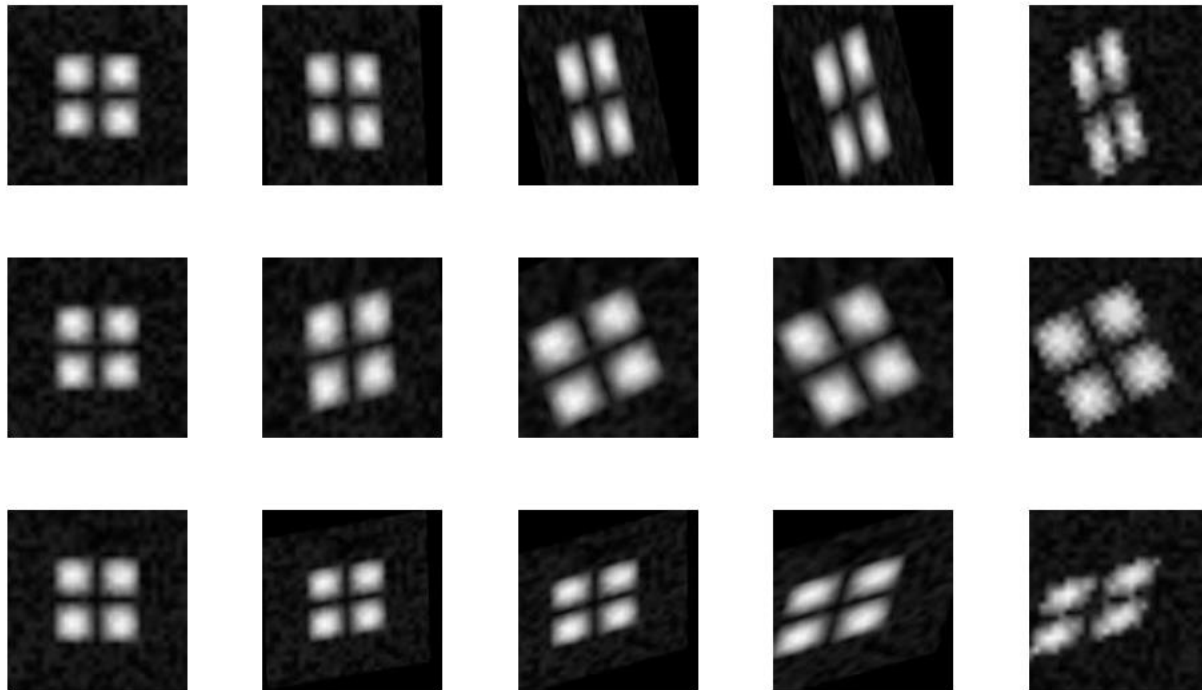


Figure 6.1: Original image (leftmost column) and image warped, translated and corrupted by noise (rightmost column) for three different motions. The intermediate columns are the images in the leftmost column deformed and translated by 4,8,and 19 iterations of the tracking algorithm.

Synthetic Results: Motion estimation

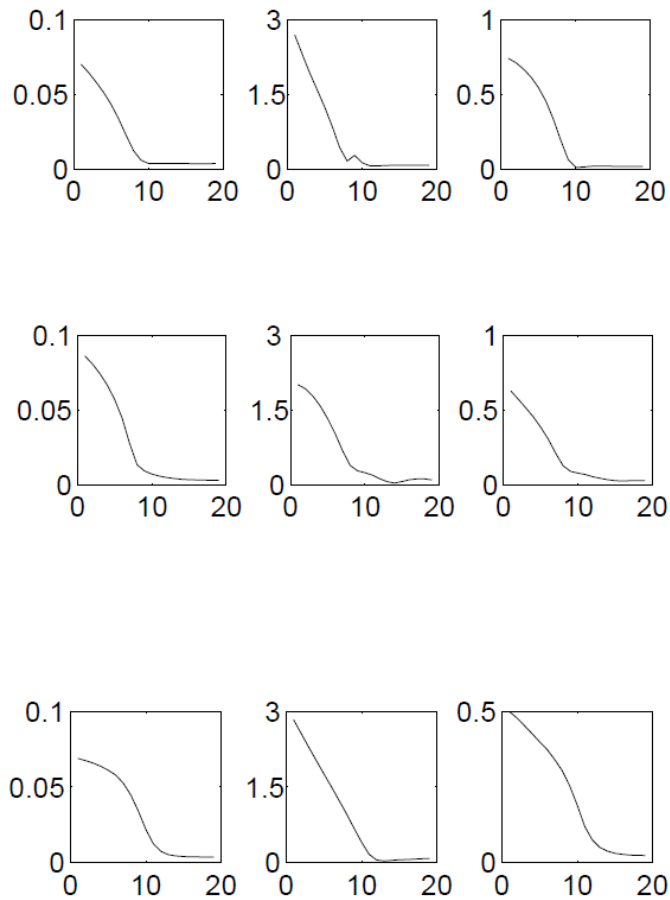


Figure 6.2: The first three columns show the dissimilarity, displacement error, and deformation error as a function of the tracking algorithm's iteration number. The last two columns are displacements and deformations computed during tracking, starting from zero displacement and deformation.

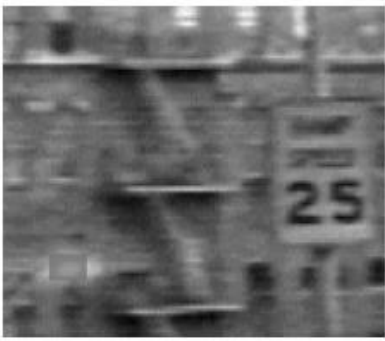


Figure 5.1: Three frame details from Woody Allen's *Manhattan*. The details are from the 1st, 11th, and 21st frames of a subsequence from the movie.

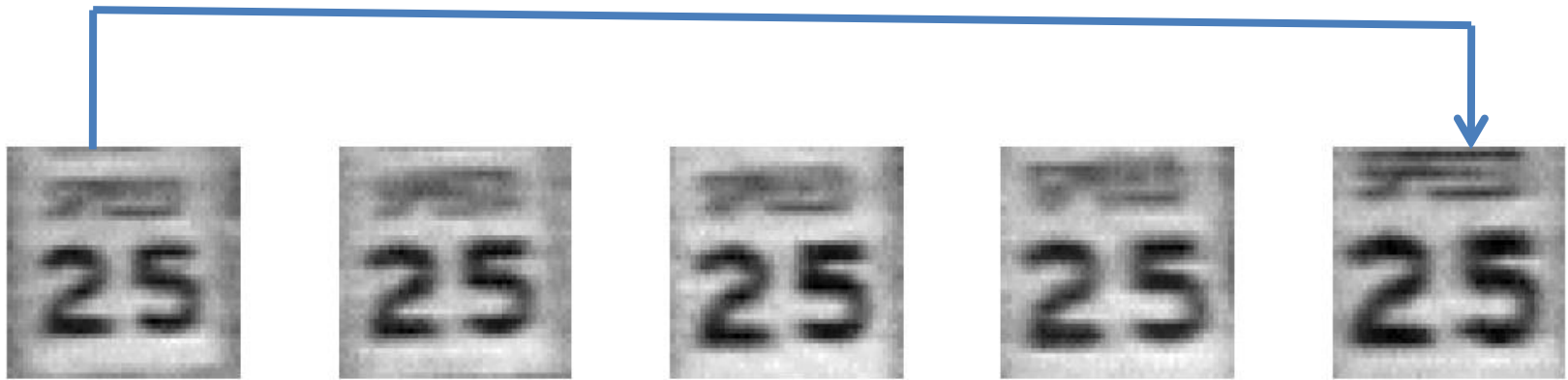


Figure 5.2: A window that tracks the traffic sign visible in the sequence of figure 5.1. Frames 1,6,11,16,21 are shown here.

Simple translation model will fail here!

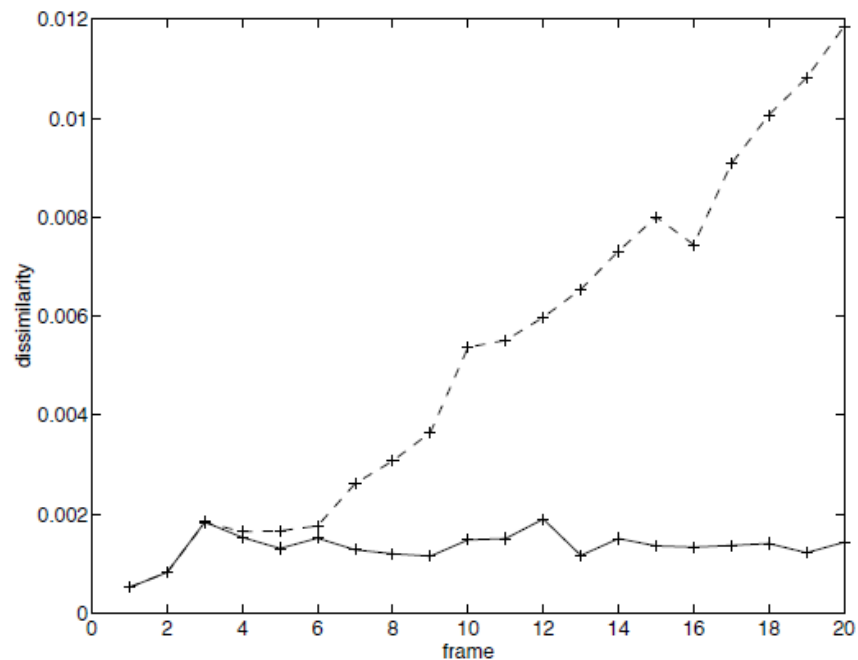


Figure 5.3: Pure translation (dashed) and affine motion (solid) dissimilarity measures for the window sequence of figure 5.2.



Figure 5.4: The same windows as in figure 5.2, warped by the computed deformation matrices.

Effect of occlusion



Figure 5.5: Three frame details from Woody Allen's *Manhattan*. The feature tracked is the bright window on the background, just behind the fire escape on the right of the traffic sign.



Figure 5.6: The bright window from figure 5.5, visible as the bright rectangular spot in the first frame (a), is occluded by the traffic sign (c).

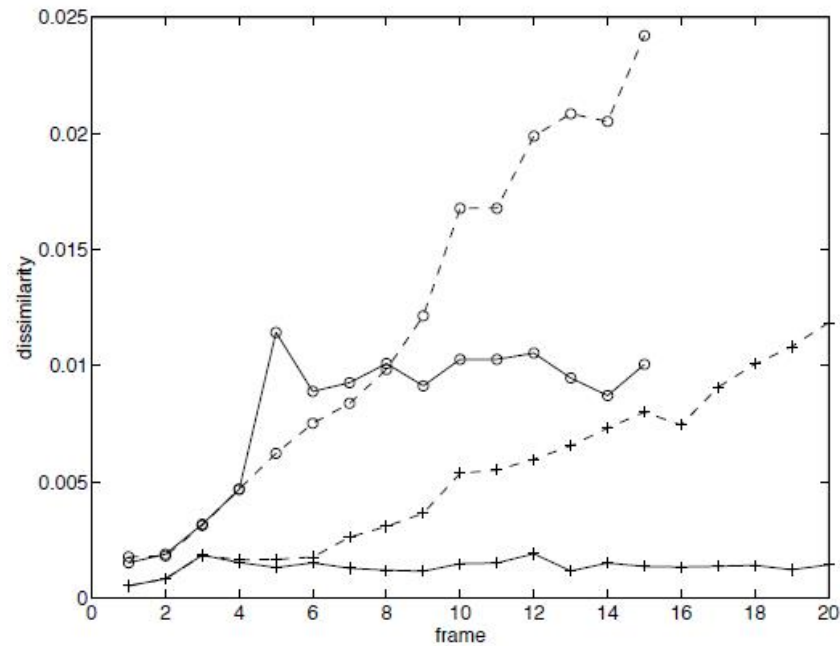


Figure 5.7: Pure translation (dashed) and affine motion (solid) dissimilarity measures for the window sequence of figure 5.1 (plusses) and 5.5 (circles).

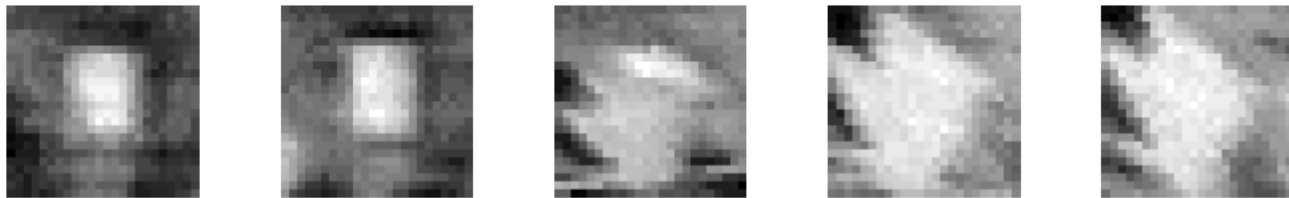


Figure 5.8: The same windows as in figure 5.6, warped by the computed deformation matrices.

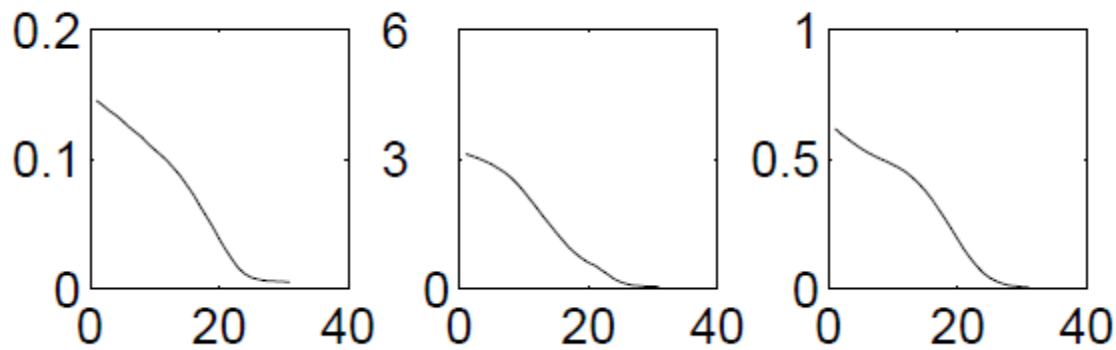
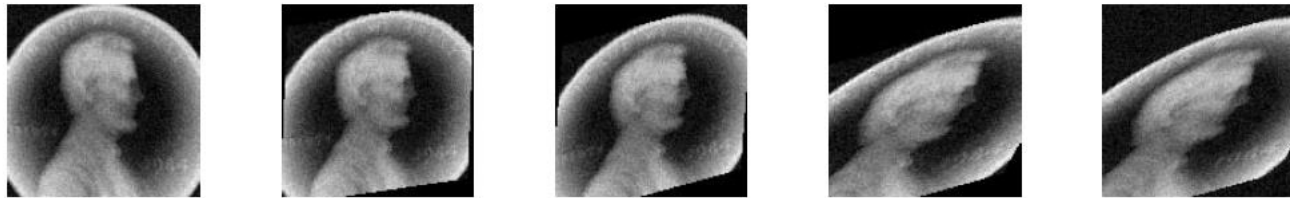


Figure 6.3: The penny in the leftmost column at the top is warped through the intermediate stages shown until it matches the transformed and noise-corrupted image in the rightmost column. The bottom row shows plots analogous to those of figure 6.2.

Experiments with a real sequence

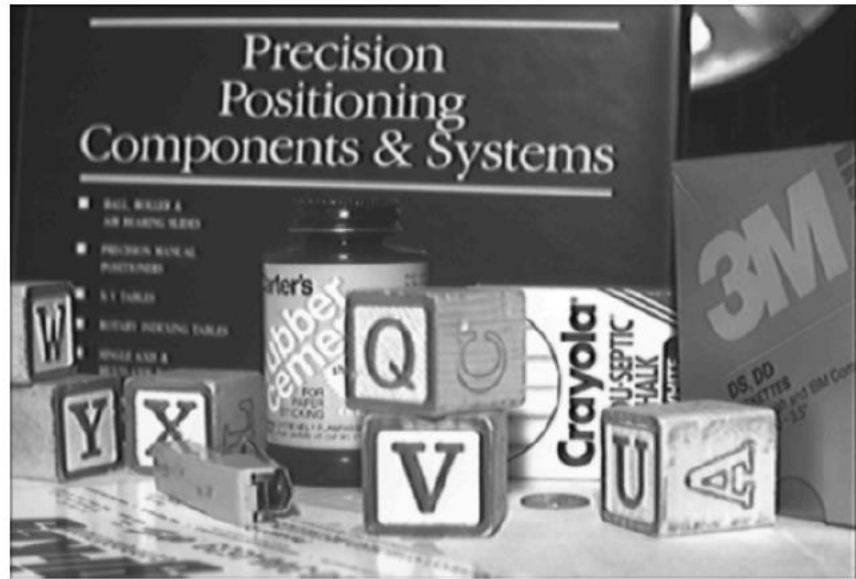


Figure 7.1: The first frame of a 26 frame sequence taken with a forward moving camera.

Camera moving forward at the rate of 2 mm per frame. Focal length of lens is 16 mm.



Figure 7.2: The features selected according to the texturedness criterion of chapter 4.

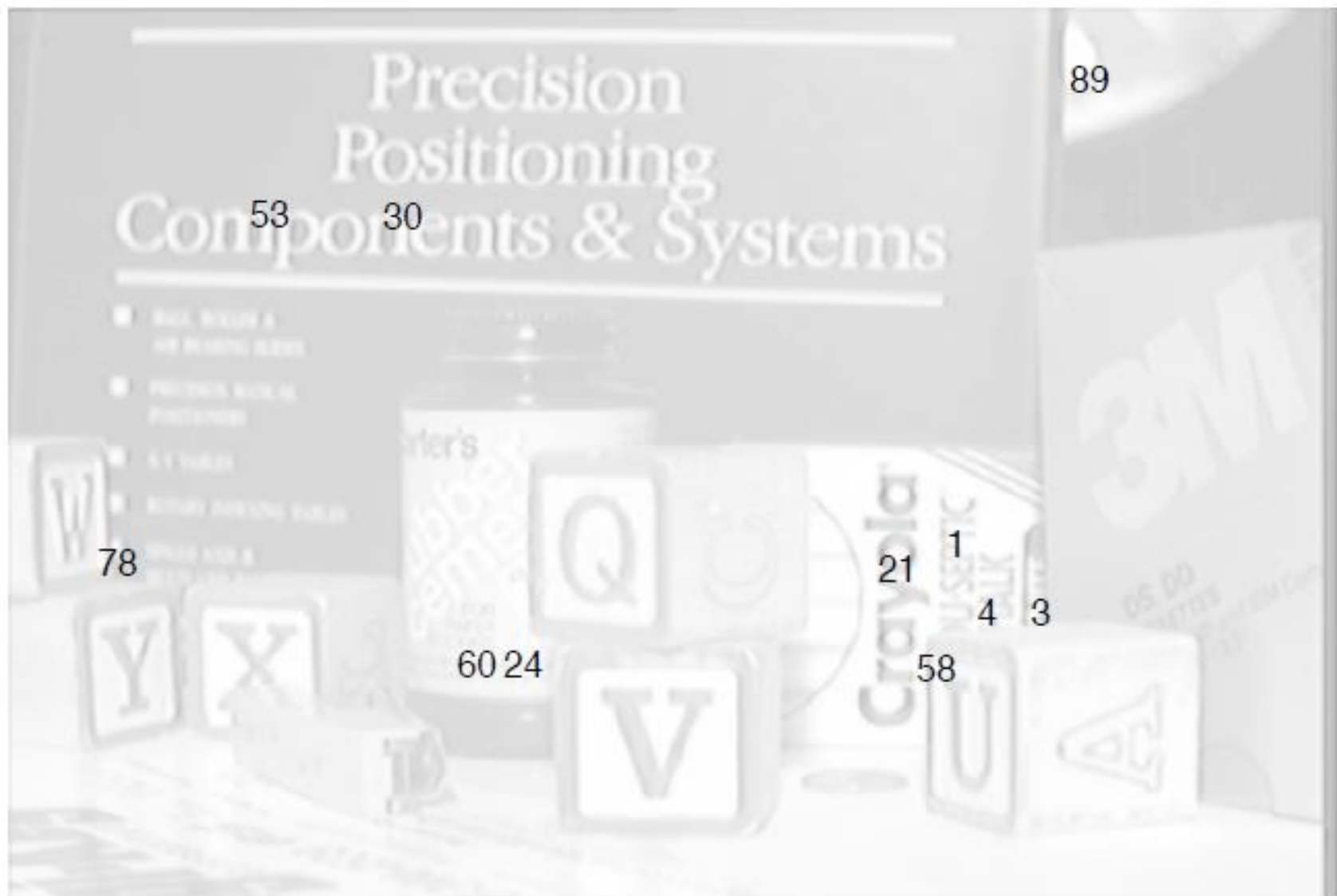


Figure 7.4: Labels of some of the features in figure 7.2.

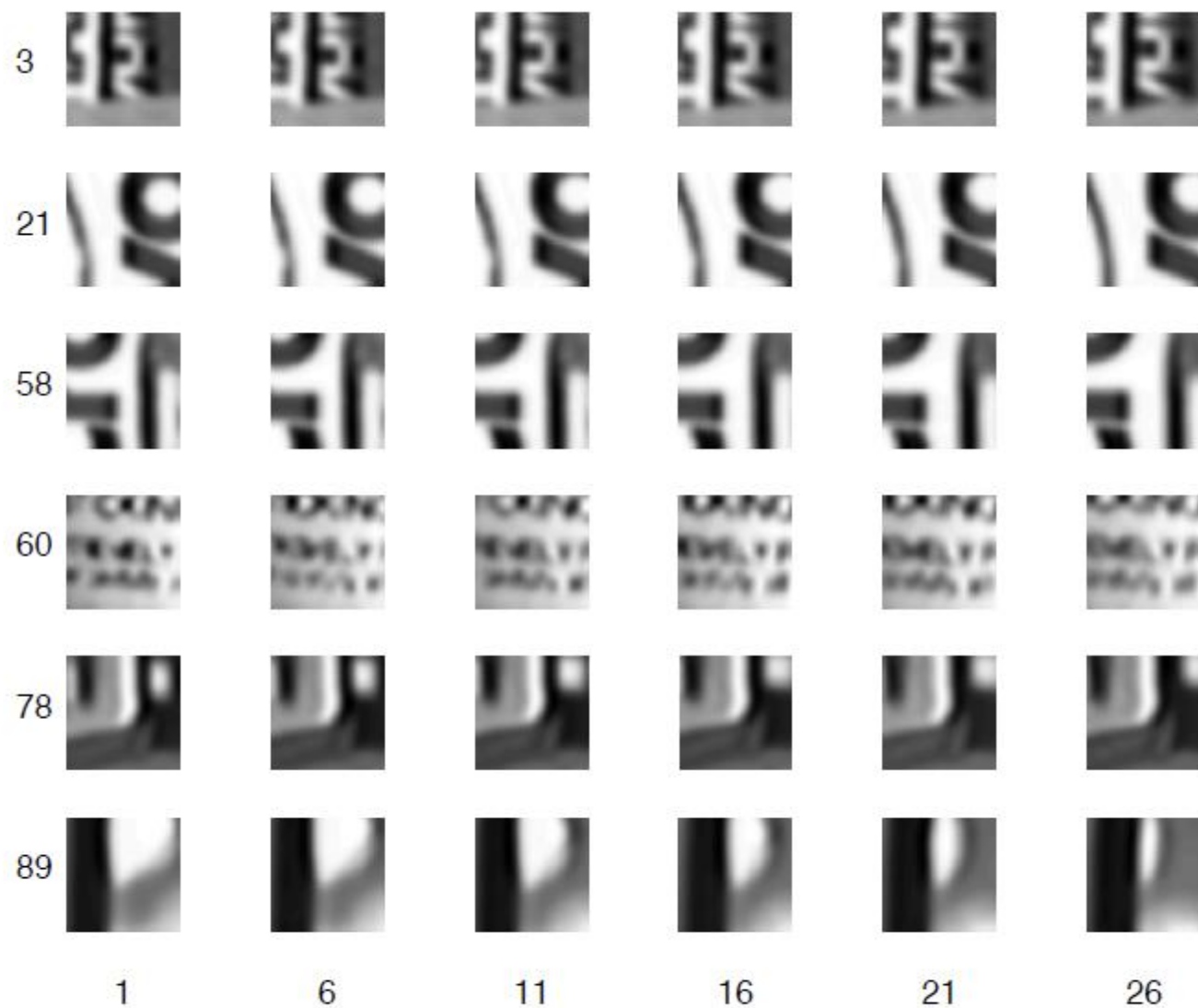


Figure 7.5: Six sample features through six sample frames.

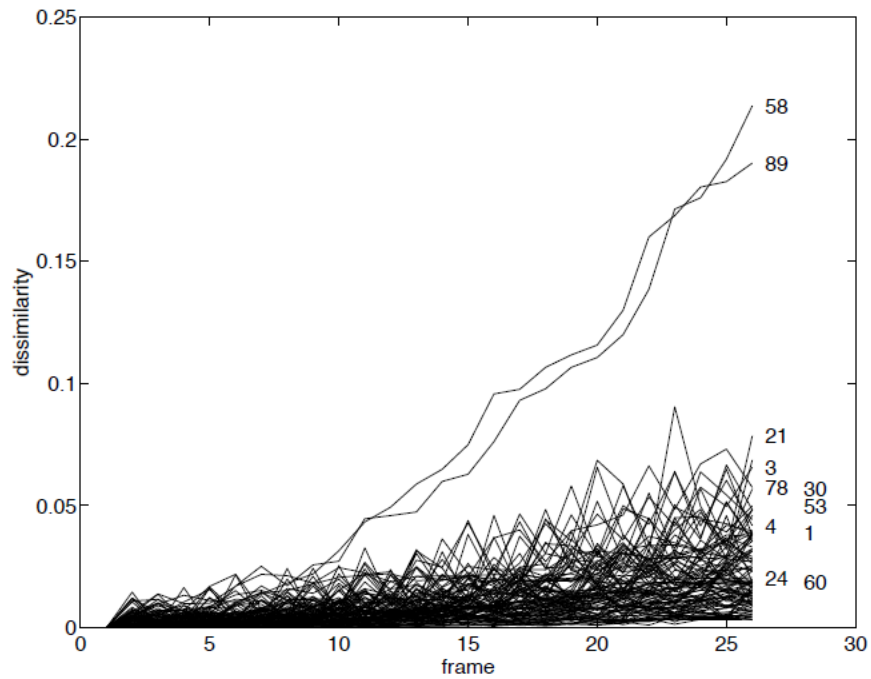


Figure 7.3: Pure translation dissimilarity for the features in figure 7.2. This dissimilarity is nearly useless for feature discrimination.

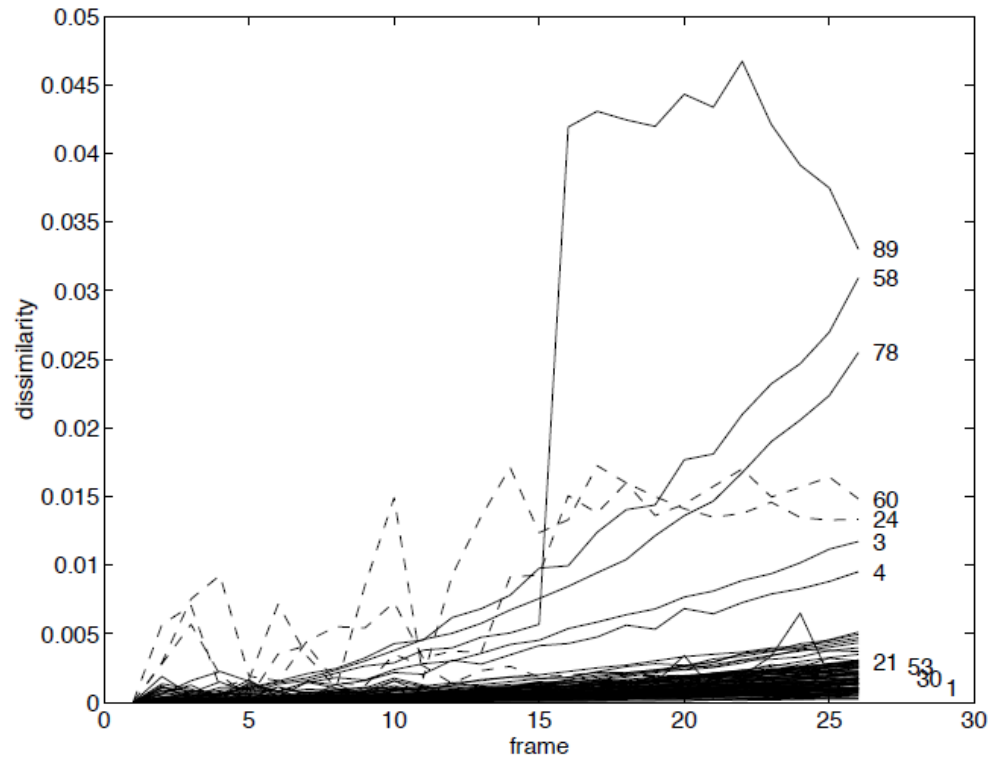


Figure 7.6: Affine motion dissimilarity for the features in figure 7.2. Notice the good discrimination between good and bad features. Dashed plots indicate aliasing (see text).

Algorithm summary

- Use the minor eigen-value method to detect salient feature points in frame 1.
- Determine motion between patches around feature points in frame t and patches at corresponding locations in frame $t+1$. Use the translation-only model (i.e. Model 1).
- Check for validity of tracking using the affine model (Model 2) based dissimilarity measure. If the measure falls above a certain threshold, discard the feature point, i.e. don't track it any further.
- Repeat for $t=1$ to T .

Appearance of new objects

- When a new object appears in the scene, it yields new feature points which will be ignored by the present version of the algorithm.
- Can be alleviated by running a feature detector (our minor eigen-value method) on every k -th frame of the video and initializing new feature points (take care to ensure they aren't too close to any existing tracked points).

Applications – panorama generation from videos

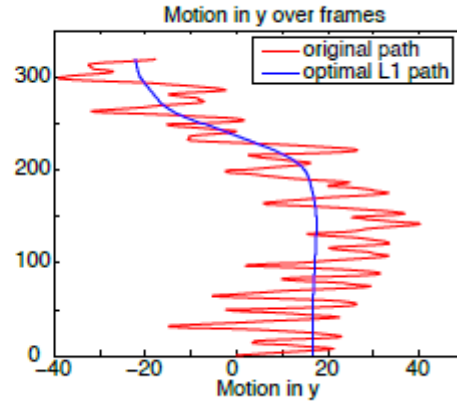
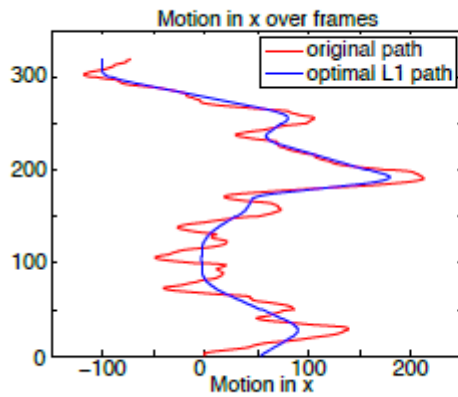
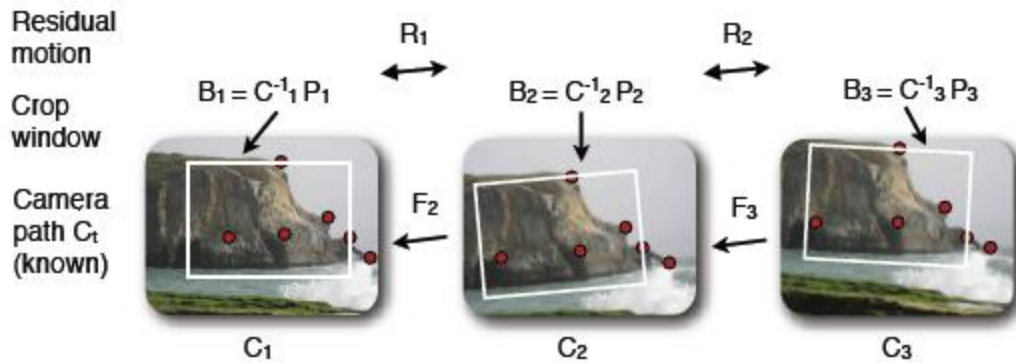


Applications – panoramas from videos

- Input: a sequence of ordered images of a large scene from consecutive viewpoints
- Output: a large image mosaic
- Method: obtain salient points in first image and track them in subsequent images
- This will give us pairs of corresponding points in consecutive images. Use their coordinates to determine the homography transformation in between the images.

Applications – Video stabilization

- Obtain salient points in first frame. Track them in the next k frames.
- Obtain affine/homography transformations between consecutive frames of the video.
- This will produce a sequence of transformation coefficients.
- Smooth this sequence.
- Warp the images as per new (smoothed) motion estimates – giving non-shaky video.



<http://www.cc.gatech.edu/cpl/projects/videstabilization/>

Grundmann, Kwatra, Essa, “Auto-directed video stabilization using robust L1 optimal paths”, ICCV 2011.

Applications – Structure from Motion

- Input: video sequence (T frames) of an object undergoing rotational motion (assume orthographic projection)
- Aim: Given a sequence of P corresponding feature points (i.e. 2D coordinates) in all images, determine (1) the 3D coordinates of all the points, and (2) the rotational motion at each time instant.