

Computer Vision CITS4240

School of Computer Science & Software Engineering
The University of Western Australia

Corner Detection and Tracking

Introduction

In an earlier lecture, we looked at motion analysis, in which we learned that 3D structure can be recovered from motion using the feature-based methods or the optical flow methods. In this lecture, we will study the feature-based methods in more detail. In particular, we will study how corners can be detected and tracked in video sequences. As a comparison, we will also study the SIFT descriptor. After that, we will study the classical tracking technique known as the Kalman Filter.

Corner features

Corner features, sometimes referred to as *interest points*, are image features characterized by their high intensity changes in the horizontal and vertical directions. For instance, if a square object is present in the image then its four corners are very good interest points.

Corners are used in shape analysis and motion analysis. As we have already seen in an early lecture, motion is ambiguous at an edge. Corners, being 2D image features, do not cause such ambiguity in motion analysis. Corners can be detected either directly from the greyscale images or as the intersection of digital contours (usually lines) if these contours have been found. We will focus on the first approach below.

The two most commonly used corner detectors to-date are the Harris corner detector [1] (widely used in Europe) and the Kanade-Lucas-Tomasi (KLT) corner detector [4, 6] (in the US). These two corner detectors share a common property in that they both operate on the *local structure matrix*, C , which has the form

$$C = \begin{bmatrix} \left(\frac{\partial I}{\partial x}\right)^2 & \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} \\ \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} & \left(\frac{\partial I}{\partial y}\right)^2 \end{bmatrix} = \begin{bmatrix} \frac{\partial I}{\partial x} \\ \frac{\partial I}{\partial y} \end{bmatrix} \begin{bmatrix} \frac{\partial I}{\partial x} & \frac{\partial I}{\partial y} \end{bmatrix}. \quad (1)$$

In practice, the partial derivatives $\partial I/\partial x$ and $\partial I/\partial y$ are often replaced, respectively, by the convolutions with a 1D horizontal and vertical Prewitt masks.

Let $\lambda_1 \geq \lambda_2$ be the two eigenvalues of the matrix C . As C is symmetric and positive semi-definite, both λ_1 and λ_2 are non-negative. Some useful geometric interpretation can be drawn from these eigenvalues:

- In a uniform and homogeneous region, $\lambda_1 = \lambda_2 = 0$.
- At the location of a step edge, $\lambda_1 > \lambda_2 = 0$. The corresponding eigenvector for λ_1 is associated with the direction that is orthogonal to the edge.
- At the location of a corner, $\lambda_1 \geq \lambda_2 > 0$. The larger are the values of λ_1 and λ_2 , the higher are the contrasts of the edges orthogonal to the directions of the corresponding eigenvectors.

Thus, the eigenvectors encode the edge directions and the eigenvalues encode the edge magnitudes. This implies that a corner should be marked at a location where the smaller eigenvalue, λ_2 , is large enough.

The Harris corner detector

For the Harris corner detector, the local structure matrix is smoothed by a Gaussian filter. That is,

$$C_{Harris} = w_G(\sigma) * \begin{bmatrix} \left(\frac{\partial I}{\partial x}\right)^2 & \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} \\ \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} & \left(\frac{\partial I}{\partial y}\right)^2 \end{bmatrix}, \quad (2)$$

where $w_G(\sigma)$ is an isotropic Gaussian filter with standard deviation σ and the operation $*$ denotes convolution. A measure of the *corner response* at each pixel coordinates (x, y) is then defined by

$$r(x, y) = \det(C_{Harris}(x, y)) - \kappa (\text{trace}(C_{Harris}(x, y)))^2, \quad (3)$$

where κ is an adjustable constant and $C_{Harris}(x, y)$ is the 2×2 local structure matrix at coordinates (x, y) .

To prevent too many corner features lumping together closely, a non-maximal suppression process on the corner response image is usually carried out to suppress weak corners around the stronger ones. This is then followed by a thresholding process. Altogether, the Harris corner detector requires three additional parameters to be specified: the constant κ , the radius, d , of the neighbourhood region for suppressing weak corners, and the threshold value t .

In terms of λ_1 and λ_2 , $\det(C_{Harris}) = \lambda_1 \lambda_2$ and $\text{trace}(C_{Harris}) = \lambda_1 + \lambda_2$. Let $\lambda_1 = \lambda$ and $\lambda_2 = \alpha \lambda$, where $0 \leq \alpha \leq 1$. The corner response r can be written as

$$\begin{aligned} r &= \lambda^2 (\alpha - \kappa (1 + \alpha)^2) \geq 0 \\ \implies \kappa &\leq \frac{\alpha}{(1 + \alpha)^2}. \end{aligned} \quad (4)$$

If two equally strong edges are present then $\alpha = 1$. This constrains that $\kappa \leq 1/4$. If one of the edges is very weak, i.e., α is small, then

$$\begin{aligned} r &\approx \lambda^2 (\alpha - \kappa) \geq 0 \\ \implies \kappa &\lesssim \alpha \ll 1. \end{aligned} \tag{5}$$

Thus, κ serves roughly as a lower bound for the ratio between the magnitude of the weaker edge and that of the stronger edge. Larger value of κ corresponds to a less sensitive detector and yields less corners; smaller value of κ corresponds to a more sensitive detector and yields more corners.

Figure 1 shows two sample outputs of the Harris corner detector on an indoor scene. In Figure 1(a), κ was set to 0.1, yielding more corners than Figure 1(b), which has κ set to 0.15.

The KLT corner detector

The Kanade-Lucas-Tomasi (KLT) corner detector [4, 6] was proposed a few years after the Harris corner detector. It has two parameters:

- a threshold value, λ_{\min} , on the second eigenvalue λ_2 , and
- a neighbourhood window radius of \tilde{d} .

The algorithm can be described as follows:

1. For each image point (x, y) :
 - (a) construct the local structure matrix over a $(2\tilde{d} + 1) \times (2\tilde{d} + 1)$ neighbourhood \mathcal{R} around (x, y) as follows:

$$C_{KLT}(x, y) = \begin{bmatrix} \sum \sum_{\mathcal{R}} \left(\frac{\partial I}{\partial x}\right)^2 & \sum \sum_{\mathcal{R}} \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} \\ \sum \sum_{\mathcal{R}} \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} & \sum \sum_{\mathcal{R}} \left(\frac{\partial I}{\partial y}\right)^2 \end{bmatrix}. \tag{6}$$

- (b) compute the smallest eigenvalue, λ_2 , of the matrix $C_{KLT}(x, y)$;
 - (c) if $\lambda_2 > \lambda_{\min}$, save (x, y) into a potential corner list, L .
2. Sort L in decreasing order of λ_2
 3. Scan the sorted list from top to bottom and select points in the list in sequence. Points that fall inside the neighbourhood \mathcal{R} of any selected points are removed.

The output produced by the KLT corner detector is a list of corner points that have $\lambda_2 > \lambda_{\min}$ and the neighbourhood \mathcal{R} of these points do not overlap. There are no simple criteria for determining the size of \tilde{d} . Common values used for \tilde{d} are within the range 2 to 10.

While a Gaussian smoothing operation is not involved in the construction of the C_{KLT} matrix in (6), the summation operation over the neighbourhood \mathcal{R} provides a uniform



(a) $\kappa = 0.1$



(b) $\kappa = 0.15$

Figure 1: Examples showing the corners detected by the Harris corner detector. In both examples, $\sigma = 1$, $t = 1500$ (threshold), and $d = 5$ (neighbourhood radius for non-maximal suppression).

smoothing for the image gradients around each image point (x, y) . The parameters λ_{\min} and \tilde{d} for the KLT corner detector are, respectively, similar to the parameter κ and the parameter d used in the non-maximal suppression of the Harris corner detector.

Figure 2 shows the corners detected by the KLT corner detector. There is a minor discrepancy between the implementation and the description given above for this detector: Rather than letting the user define the value of \tilde{d} , the user should specify the number of corners to be detected. In Figure 2, $\lambda_{\min} = 1$ and 200 corners were specified.



Figure 2: An example showing the corners detected by the KLT detector. In this example, $\lambda_{\min} = 1$ and 200 corners were detected.

The SIFT keypoint detector

The SIFT (Scale Invariant Feature Transform) keypoint detector was proposed by Lowe in 1999 [2, 3]. The SIFT features are feature vectors that represent local image measurements, which have been reported to be relatively invariant to image translation, scaling and rotation and partially invariant to changes in illumination and local image deformations.

The main steps involved in the SIFT detector in locating keypoints are (see Figure 3:

- The input image, $I(x, y)$, is convolved with a number of Gaussian filters whose standard deviations $\{\sigma_1, \sigma_2, \dots\}$ differ by a fixed scale factor. That is, $\sigma_{j+1} = k\sigma_j$ where k is a constant scalar that should be set to $\sqrt{2}$. The convolutions yield a small number of smoothed images, denoted by $\{G(x, y, \sigma_1), G(x, y, \sigma_2), \dots\}$.
- The adjacent smoothed images are then subtracted to yield a small number (3 or 4) of DoG (Difference-of-Gaussian) images. That is,

$$D(x, y, \sigma_j) = G(x, y, \sigma_{j+1}) - G(x, y, \sigma_j).$$

- The smoothed images from Step 1 are subsampled and the procedure in Step 2 is repeated on the subsampled images, yielding a number of DoG images over the scale space.
- Each point in these DoG images is then examined. A keypoint is marked at a location where the point is a local minimum or maximum of its 8 neighbours on the same scale and of its 9 neighbours on the scales above and below (see Figure 4).

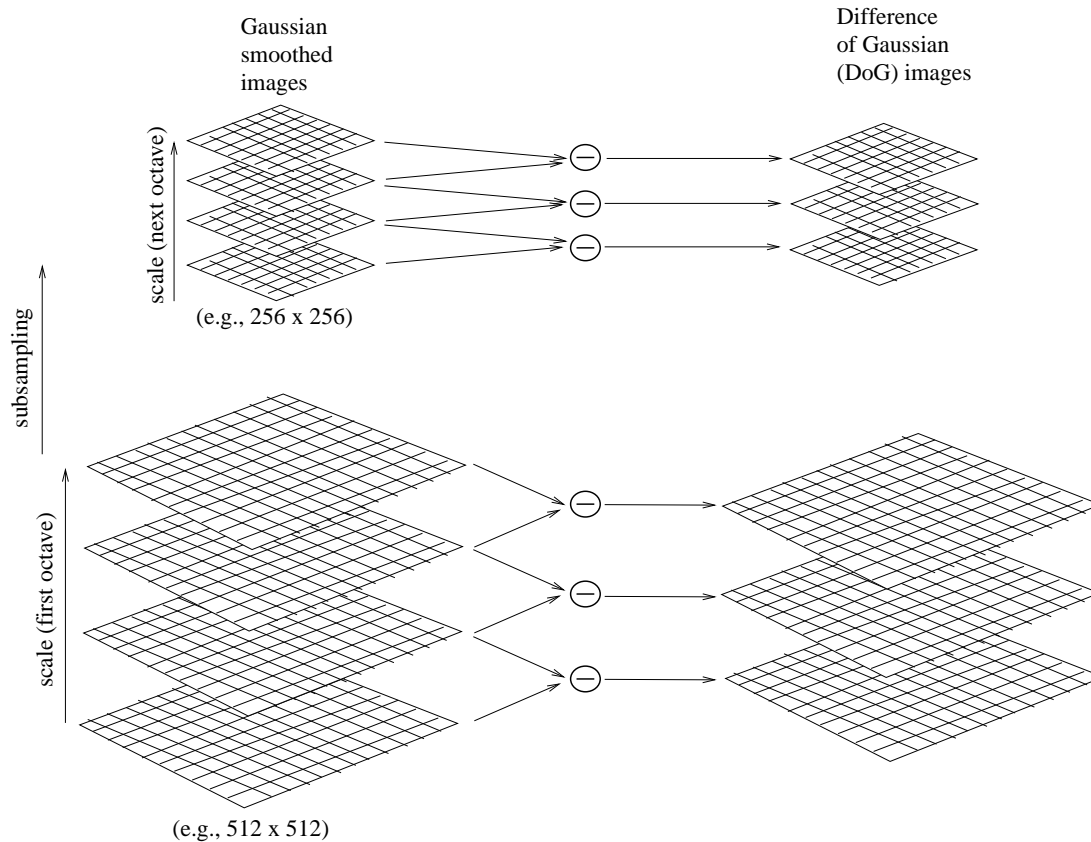


Figure 3: The DoG image pyramid used by the SIFT detector to locate keypoints.

The keypoints identified from the above steps are then examined for possible elimination if the two local principal curvatures of the intensity profile around the keypoint exceeds a specified threshold value. This elimination step involves estimating the ratio between the two eigenvalues of the 2×2 Hessian matrix (i.e., the second partial derivatives) of the local image intensity around each keypoint.

Having found the keypoint locations, the surrounding intensity and gradient information around each keypoint are then examined for the coding of the SIFT descriptor at that keypoint. The image gradients are computed and classified into 8 orientations for a 4×4 histogram array (see Figure 5), giving a SIFT descriptor of 128 elements long for each keypoint.

Unlike the Harris and KLT corner detectors, the SIFT keypoints do not often coincide at corner features. However, the SIFT descriptor of each keypoint does provide useful information for wide baseline stereo matching and for object recognition. It has been applied in these types applications with very promising results.

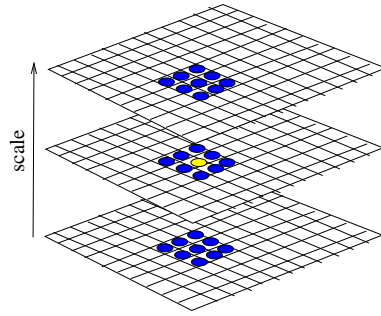


Figure 4: A keypoint must be a local minimum or maximum of its 26 neighbours.

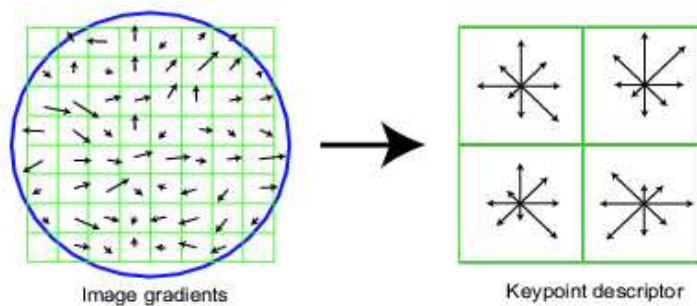


Figure 5: Extracted from [3]: This figure shows a simplified example of a 2×2 descriptor array computed from an 8×8 window of image gradient vectors. The SIFT detector reported in [3] works on 16×16 windows of image gradient vectors, giving descriptors of 128 elements in length.

Figure 6 shows the outputs of the SIFT keypoint detector on two images of a model house taken from two different viewpoints. Altogether, 1225 and 1360 keypoints were detected by SIFT for the two images shown in Figure 6(a) and (b). To show that the keypoints are relatively stable under the change of the camera's viewpoint, the pairwise distances of the keypoint descriptors (128-vectors) from the two set of keypoints were computed. Two keypoints were considered to be a match if their distance in the 128-dimensional space is less than a given threshold value. The 291 matching keypoints found are shown in Figure 6(c) and (d). Note that, due to the similarities in appearances of portions of the scene (e.g., the two identical-looking chimneys), some incorrect matches are evident. Otherwise, the majorities of the matches are correct, revealing the rotation and translation of the camera between the two views.

The maximally stable extremal regions for wide baseline stereo matching were recently proposed by Matas et al [5]. Interested students are referred to [5] for details.

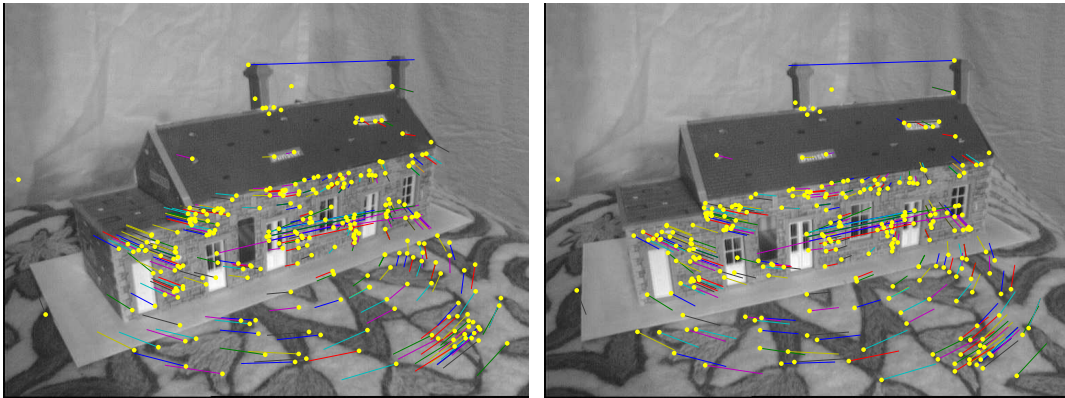
Corner feature trackers

In motion analysis of video images, the displacement of corner features is very small from one video frame to the next as the camera motion is small between video frames. This allows the corner features to be tracked.



(a)

(b)



(c)

(d)

Figure 6: (a) and (b): The input greyscale images for testing the SIFT operator. (c) and (d): The 291 matching keypoints are shown as yellow dots. The colour vectors indicate the displacement of the matching keypoints to the other image. The vectors attached to each pair of matching keypoints have the same colour, which was chosen randomly from a small set of available colours.

The approach adopted by the KLT tracker [4, 6] can be described as follows. Let $I(\cdot, \cdot, t)$ and $I(\cdot, \cdot, t + \Delta_t)$ be the two consecutive image frames in a video sequence being considered. Assuming that the imaged scene is static and that the only motion in the image is induced by camera motion, then

$$I(x, y, t + \Delta_t) = I(x - \Delta_x, y - \Delta_y, t),$$

where the amount of motion (Δ_x, Δ_y) is called the *displacement* of the point at (x, y) between the time instant t and $t + \Delta_t$. In general, (Δ_x, Δ_y) is a function of x, y, t , and Δ_t .

Let $\mathbf{x} = (x, y)^\top$ and $\mathbf{d} = (\Delta_x, \Delta_y)^\top$. Define $J(\mathbf{x}) = I(\mathbf{x}, t + \Delta_t)$ and let us drop the time variable for brevity. Then

$$J(\mathbf{x}) = I(\mathbf{x} - \mathbf{d}) + \epsilon.$$

where ϵ is a noise or error term. The displacement vector \mathbf{d} should therefore be chosen so

as to minimize the error, ϵ , over a given tracking window \mathcal{D} :

$$\epsilon = \int_{\mathcal{D}} w [I(\mathbf{x} - \mathbf{d}) - J(\mathbf{x})]^2 d\mathbf{x} \quad (7)$$

where w is a weighting function, commonly set to 1 (uniform function) or replaced by a Gaussian function.

Assuming that the displacement vector \mathbf{d} is small, $I(\mathbf{x} - \mathbf{d})$ can be approximated by its Taylor expansion to the linear term:

$$I(\mathbf{x} - \mathbf{d}) = I(\mathbf{x}) - \mathbf{g} \cdot \mathbf{d},$$

where $\mathbf{g} = (\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y})^\top$ is the image gradient vector at \mathbf{x} . and (7) becomes

$$\epsilon = \int_{\mathcal{D}} w [I(\mathbf{x}) - \mathbf{g} \cdot \mathbf{d} - J(\mathbf{x})]^2 d\mathbf{x}. \quad (8)$$

It can be verified that \mathbf{d} can be estimated linearly via

$$G\mathbf{d} = \int_{\mathcal{D}} w(I(\mathbf{x}) - J(\mathbf{x}))\mathbf{g} d\mathbf{x}, \quad (9)$$

where $G = \int_{\mathcal{D}} w\mathbf{g}\mathbf{g}^\top d\mathbf{x}$ is a symmetric 2×2 coefficient matrix.

The size of the tracking window \mathcal{D} is often difficult to determine as the image motions of the corner features depend on two unknown entities: the 3D motion of the camera and the distance of the object from the camera. One often needs to set this window size based on some prior knowledge about the video sequence.

When the baseline between the camera's optical centres at the first and the current video frames is widened, the KLT tracker could start to lose tracks of the corner features or produce outlying tracks, as the tracker is not affine invariant. For motion analysis, an outlier detection scheme must be employed to remove these outlying tracks.

The Kalman Filter

This section is a brief account of a classical tool of optimal estimation theory, the *linear Kalman Filter*.

Imagine that we have a physical system modelled by a time-dependent *state vector*, $\mathbf{x}(t)$, and a set of equations, called the *system model*. The system model is a vector equation describing the evolution of the state in time. We denote the discrete equally spaced time instants by $t_k = t_0 + k\Delta_t$, with $k = 0, 1, \dots$ and Δ_t is the sampling interval. Let \mathbf{x}_k be the state $\mathbf{x}(t_k)$. The linear system model can be written in matrix-vector form as

$$\mathbf{x}_k = \Phi_{k-1}\mathbf{x}_{k-1} + \mathbf{w}_{k-1},$$

where Φ_{k-1} denotes the *state transition matrix* at time t_{k-1} and \mathbf{w}_k is a random vector modelling the additive system noise, which is often assumed to be $\mathcal{N}(\mathbf{0}, Q_k)$. In general, Φ is a function of time and so it can account for complicated dynamics.

We also assume that, at any time instant t_k , a noisy measurement of the state vector (or at least some components of it) is taken, and that the following relationship between the measurements and the true system state holds:

$$\mathbf{z}_k = H_k \mathbf{x}_k + \mathbf{v}_k.$$

In this equation, \mathbf{z}_k is the vector of measurements at time t_k , H_k is known as the *measurement matrix*, and \mathbf{v}_k is a random vector modelling the measurement noise. Again, \mathbf{v}_k is often assumed to follow $\mathcal{N}(\mathbf{0}, R_k)$.

The state vector to be estimated at each time instant is represented by $\hat{\mathbf{x}}_{k|k}$, where the subscript $j|k$ means “the entity at time t_j , given the input measurement observed up to time t_k ”. Also of interest is the error covariance matrix $P_{k|k}$, which is a measure of the estimated accuracy of the state estimate $\hat{\mathbf{x}}_{k|k}$.

The Kalman Filter has two distinct phases: the *predict* and the *update* phases. Assume that we have obtained an initial estimate of the state vector $\hat{\mathbf{x}}_{0|0}$ and its associated error covariance matrix $P_{0|0}$.

- In the predict phase, predict the state estimate and the error covariance matrix at time t_k given the input measurement up to time t_{k-1} :

$$\hat{\mathbf{x}}_{k|k-1} = \Phi_k \hat{\mathbf{x}}_{k-1|k-1} \tag{10}$$

$$P_{k|k-1} = \Phi_k P_{k-1|k-1} \Phi_k^\top + Q_k \tag{11}$$

- In the update phase, take into account the latest input measurement (i.e., at time t_k) and update the state estimate and the error covariance matrix. The following steps are involved:

- Compute the innovation or measurement residual:

$$\tilde{\mathbf{y}}_k = \mathbf{z}_k - H_k \hat{\mathbf{x}}_{k|k-1}. \tag{12}$$

- Compute the innovation (or residual) covariance matrix:

$$S_k = H_k P_{k|k-1} H_k^\top + R_k. \tag{13}$$

- Compute the Kalman gain:

$$K_k = P_{k|k-1} H_k^\top S_k^{-1}. \tag{14}$$

- Finally, compute the updated state estimate:

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + K_k \tilde{\mathbf{y}}_k, \tag{15}$$

and the covariance matrix of $\hat{\mathbf{x}}_{k|k}$:

$$P_{k|k} = (I - K_k H_k) P_{k|k-1}. \tag{16}$$

Example. Suppose that we want to track the image motion of a point (e.g., a corner), given that its initial position has been detected. Suppose also that we have some prior knowledge that the velocity of the point is constant in both the x - and y -directions in the image. We can represent our state vector as $\mathbf{x} = (x, y, \dot{x}, \dot{y})^\top$, where $(\dot{x}, \dot{y})^\top$ is the velocity vector. We incorporate the \dot{x} and \dot{y} into our state vector to make it more convenient for predicting the position of the point in the next time instant.

Our linear system model can be written as

$$\begin{bmatrix} x_k \\ y_k \\ \dot{x}_k \\ \dot{y}_k \end{bmatrix} = \begin{bmatrix} 1 & 0 & \Delta_t & 0 \\ 0 & 1 & 0 & \Delta_t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ \dot{x}_{k-1} \\ \dot{y}_{k-1} \end{bmatrix} + \mathbf{w}_{k-1},$$

or $\mathbf{x}_k = \Phi \mathbf{x}_{k-1} + \mathbf{w}_{k-1}$, (17)

where Δ_t is the sampling interval. In this example, we have Φ unchanged over time. So $\Phi_k = \Phi$, for all k . Depending on how much knowledge we have about the noise term, \mathbf{w}_{k-1} may also be modelled as *isotropic* as well as invariant over time. So, $\forall k$, $\mathbf{w}_k = \mathbf{w} \sim \mathcal{N}(\mathbf{0}, \sigma^2 I)$, for some known standard deviation σ .

As what we can detect from the image is the location of the point, we have the following relationship between the input measurement, $(\tilde{x}_k, \tilde{y}_k)^\top$, and the true state vector of the system:

$$\begin{bmatrix} \tilde{x}_k \\ \tilde{y}_k \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_k \\ y_k \\ \dot{x}_k \\ \dot{y}_k \end{bmatrix} + \mathbf{v}_k$$

or $\mathbf{z}_k = H \mathbf{x}_k + \mathbf{v}_k$. (18)

Again, we can model H to be invariant over time for this example. Thus, $H_k = H$, for all k . We may also simplify the noise term \mathbf{v}_k in a similar way and let $\mathbf{v}_k = \mathbf{v}$, for all k , with $\mathbf{v} \sim \mathcal{N}(\mathbf{0}, s^2 I)$, for some known standard deviation s .

The tracking process then proceeds as described by the procedure (Equations (11) and (16)) above. At each time instant, we can assess the accuracy of the state estimate by studying the estimated error covariance matrix. In the example above, $P_{k|k}$ is a 4×4 covariance matrix. The 2×2 submatrix formed by the first two rows and columns of $P_{k|k}$ tells us the uncertainty of the tracked position of the point in the image plane.

References

- [1] Chris Harris and Mike Stephens. “A Combined Corner and Edge Detector”. *Proc. of The Fourth Alvey Vision Conference*, Manchester, pp. 147-151. 1988.
- [2] D. G. Lowe. “Object recognition from local scale-invariant features”, *International Conference on Computer Vision*, Corfu, Greece, pp. 1150-1157, Sep 1999.
- [3] D. G. Lowe. “Distinctive image features from scale-invariant keypoints”, *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91-110, 2004.

- [4] B. D. Lucas and T. Kanade. “An Iterative Image Registration Technique with an Application to Stereo Vision”, *International Joint Conference on Artificial Intelligence*, pp. 674-679, 1981.
- [5] J. Matas, O. Chum, M. Urban, and T. Pajdla. “Robust Wide Baseline Stereo from Maximally Stable Extremal Regions”, *British Machine Vision Conference*, 2002.
- [6] C. Tomasi and T. Kanade. “Detection and Tracking of Point Features”, Carnegie Mellon University, Technical Report CMU-CS-91-132, Apr 1991, Pittsburgh, PA, USA,