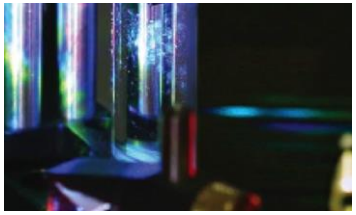
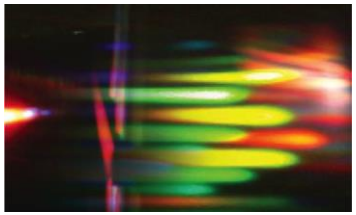
Imperial College London





COMP70058 Computer Vision

Lecture 11 – Feature Tracking

Stamatia (Matina) Giannarou, PhD
The Hamlyn Centre for Robotic Surgery

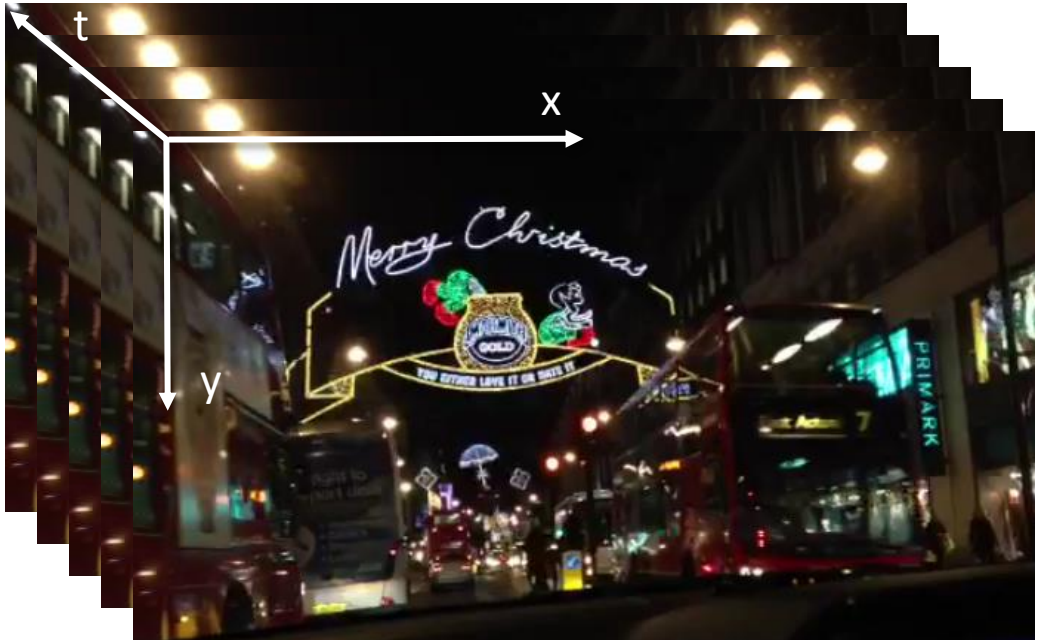
stamatia.giannarou@imperial.ac.uk

The Hamlyn Centre
for Robotic Surgery

Contents

- Tracking in Image Sequences
- Lucas-Kanade Tracker and Multiscale Implementation
- The need for incorporating temporal information for tracking
- Kalman filter
- Extended Kalman filter and other techniques
- Tracking, error matrices and error rates
- Receiver Operating Characteristic (ROC) analysis

Tracking in Image Sequences



- **Key assumptions**
  - the intensities of an object point remain the same across frames – brightness constancy
  - the motion of features between consecutive frames is small, and the camera position changes slowly – temporal persistence
  - neighbouring features belong to the same physical surface and have similar motion – spatial coherence

$$I(x + \Delta x, y + \Delta y, t + \Delta t) = I(x, y, t)$$

u v

for simplicity assume to be 1 (i.e., one time step)

Brightness Constancy Constraint

- Based on the brightness constancy, we can use Taylor series expansion to establish the relationship among intensity gradient along the *x, y, t* axes

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

$$\text{Because} \quad I(x+u, y+v, t+1) \approx I(x, y, t) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \frac{\partial I}{\partial t} + \ldots.$$

$$\text{Therefore} \quad I(x+u, y+v, t+1) - I(x, y, t) = \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \frac{\partial I}{\partial t}$$

$$\frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \frac{\partial I}{\partial t} = 0 \quad \text{or} \quad I_x u + I_y v + I_t = 0$$

- When represented in matrix form, we have

$$\begin{bmatrix} I_x & I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = -I_t$$

Gradient along x Gradient along y Gradient along t

Displacement to be recovered between adjacent image frames (t); one equation two unknowns, thus not possible. However, if we know there are a few points moving together, we can solve it by using the least-squares approach.

Brightness Constancy Constraint

- If we apply the spatial coherence constraint on a patch of size N × N we get N*N equations

$$\begin{bmatrix} I_x(p_1) & I_y(p_1) \\ I_x(p_2) & I_y(p_2) \\ \vdots & \vdots \\ I_x(p_N) & I_y(p_N) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(p_1) \\ I_t(p_2) \\ \vdots \\ I_t(p_N) \end{bmatrix}$$

- This system has more equations than unknowns and usually it is over-determined. The displacements can be estimated as:

$$\min \left\| \mathbf{Ad} - \mathbf{B} \right\|^2$$

Lucas-Kanade Algorithm

- The Lucas–Kanade method estimates an optimal solution for point displacements by solving the Least Squares problem:

$$(\mathbf{A}^T\mathbf{A})\mathbf{d} = \mathbf{A}^T\mathbf{B}$$

$$\begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = -\begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}$$

Have we seen this before?

- The summations are over all pixels in the NxN window where spatial coherence is assumed
- The Lucas–Kanade method estimates an optimal solution for the point displacements by solving the above least squares problem. Rather than using the Harris corner detector, it uses the Shi-Tomasi implementation, and therefore we call this Lucas-Kanade-Tomasi Tracker (LKT).

Lucas-Kanade Algorithm

When is the system solvable?

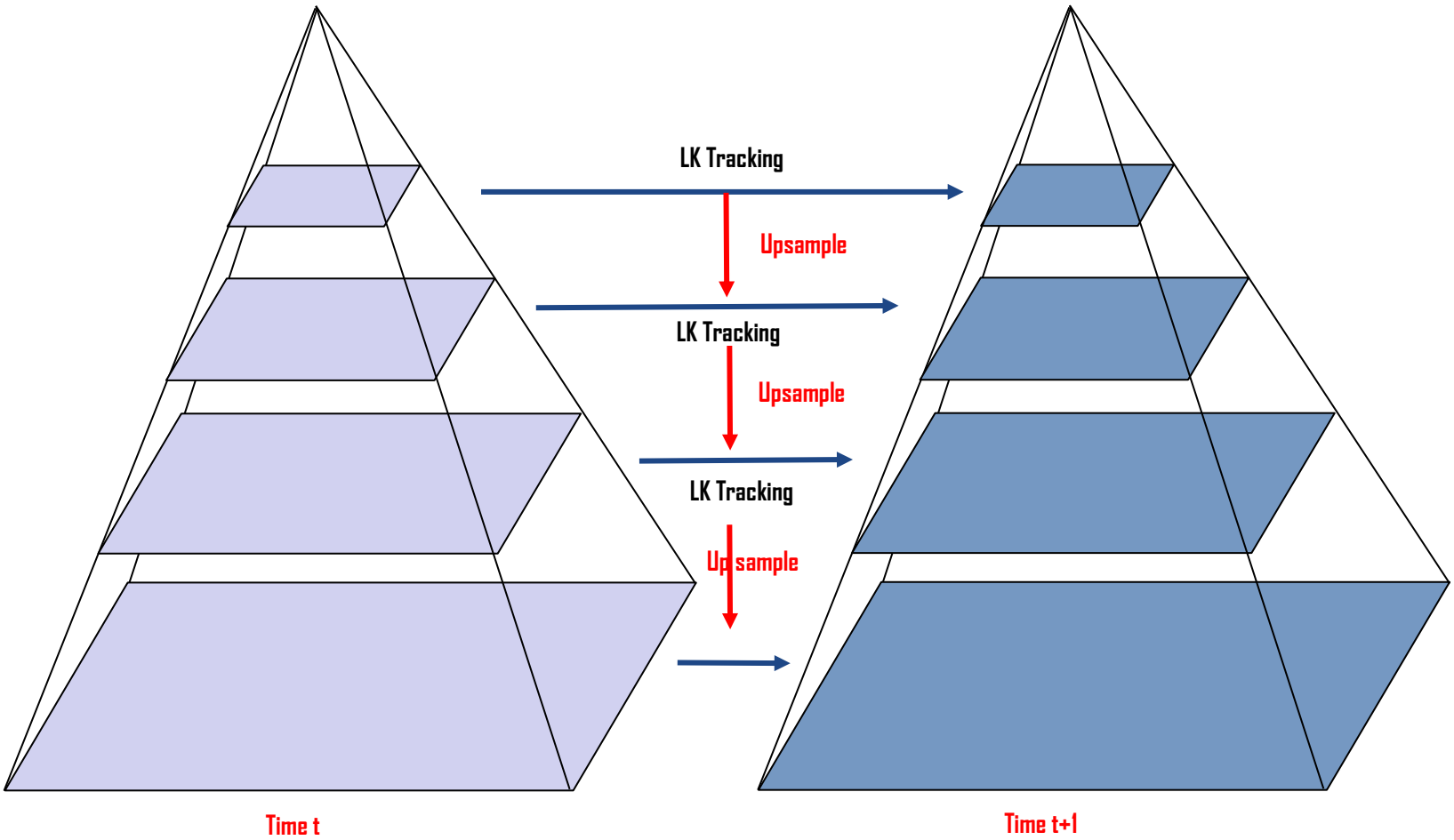
$$(\mathbf{A}^T\mathbf{A})\mathbf{d} = \mathbf{A}^T\mathbf{B}$$

- $\mathbf{A}^T\mathbf{A}$ should be invertible
- $\mathbf{A}^T\mathbf{A}$ should not be too small due to noise
- $\mathbf{A}^T\mathbf{A}$ should be well-conditioned
- The eigenvalues of $\mathbf{A}^T\mathbf{A}$ should satisfy $min(\lambda_1, \lambda_2) >$ a predefined threshold

How to Deal with Large Displacement

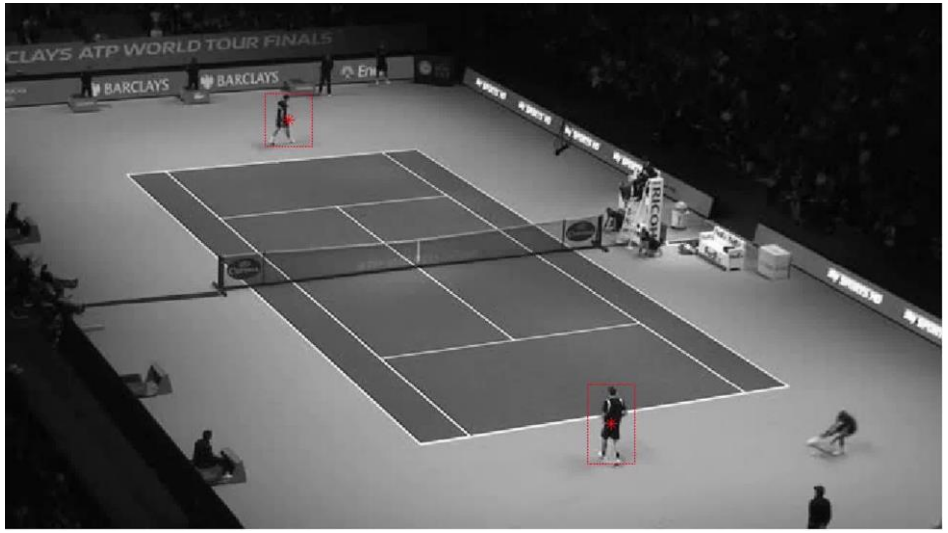
- The LK tracker assumes small displacement, thus only reliable for small motion
- For large motion, multi-resolution based on an image pyramid can be used

LK Tracking with Image Pyramid



Construct image pyramids, apply LK tracker to low-resolution images, propagate results to higher-resolution images, apply LK tracker …

LKT In Action



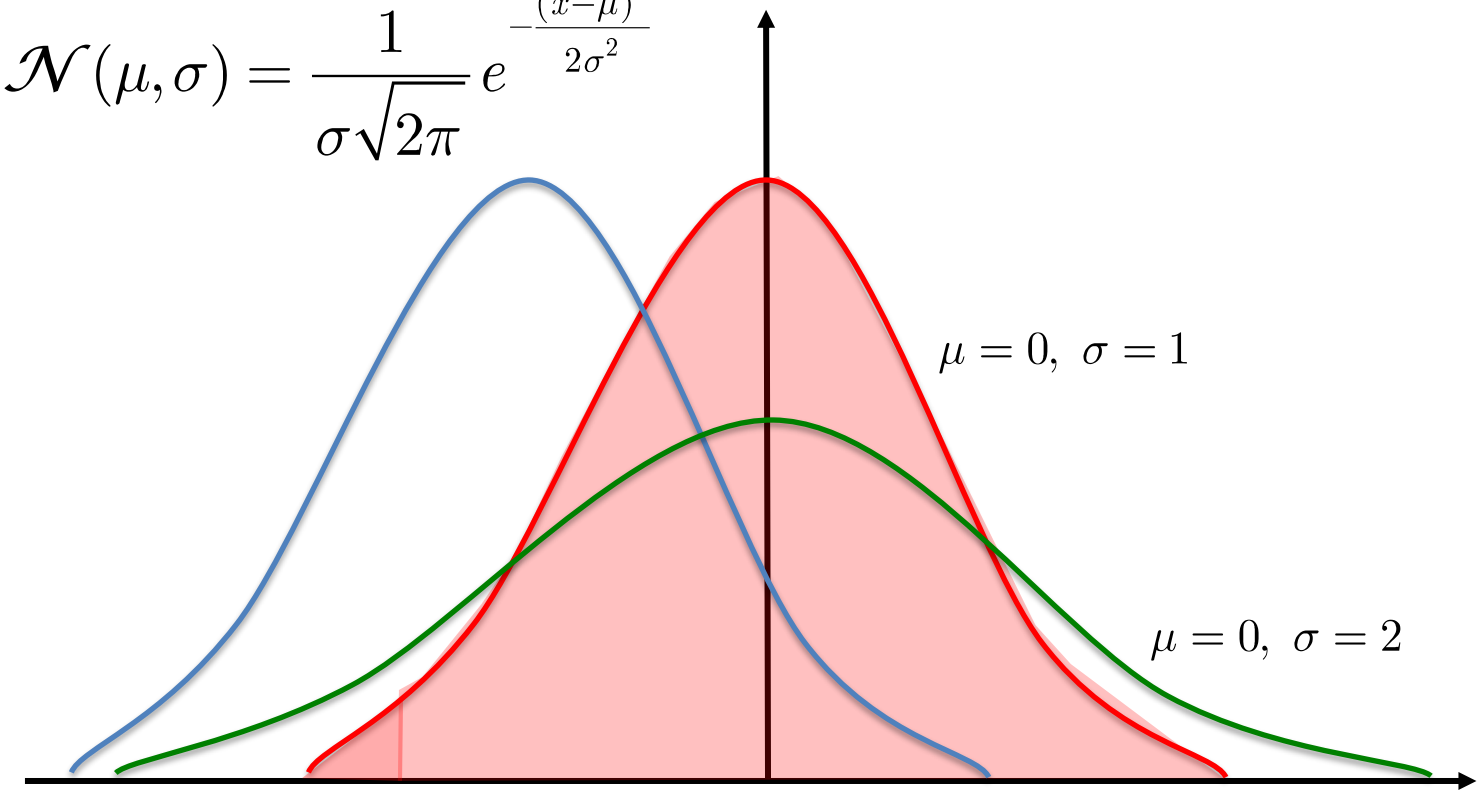
Why Incorporate Temporal Information

- Reliable tracking, particularly for long image sequences, is practically difficult
- Changes in appearance due to noise, illumination, deformation and pose variation can all contribute to tracking failure
- Occlusion, interaction among objects and cluttered background can also contribute tracking failure
- If we have some knowledge **(model)** about the dynamics of the moving objects, their positions at the next image frame can be predicted, which can be combined with image details **(observation)** to improve the robustness, as well as accuracy of the tracking results
- Kalman Filter, originally proposed by Rudolph E Kalman for recursive solution to discrete linear fitting, is particularly useful

Image from: National Academy of Engineering.
Kalman Filter Resources see: http://www.cs.unc.edu/~welch/kalman/

The Normal Distribution

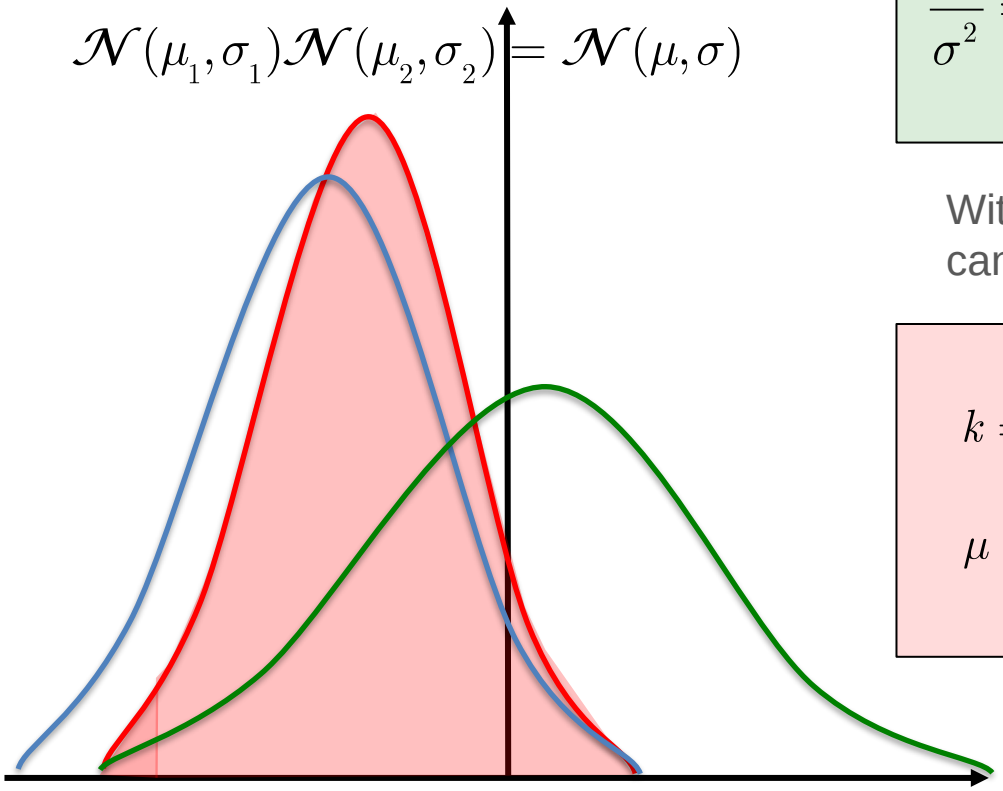
$$\mathcal{N}(\mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$



Multivariate Normal Distribution

- What happens if we combine two distributions together, would it still be a Gaussian? If so, what will be its parameters?

$$\mathcal{N}(\mu_1,\sigma_1)\mathcal{N}(\mu_2,\sigma_2) = \mathcal{N}(\mu,\sigma)$$



One can prove that (see http://www.lucamartino.altervista.org/2003-003.pdf for detailed derivation)

$$\frac{1}{\sigma^2} = \frac{1}{\sigma_1^2} + \frac{1}{\sigma_2^2} \quad \text{and} \quad \mu = \left(\frac{\mu_1}{\sigma_1^2} + \frac{\mu_2}{\sigma_2^2}\right)\sigma^2$$

With some simple re-arrangements, we can get:

$$k = \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2}$$

$$\mu = \mu_1 + k\left(\mu_2 - \mu_1\right) \quad \text{and} \quad \sigma^2 = \sigma_1^2 - k\sigma_1^2$$

Multivariate Normal Distribution

- What happens for higher dimensions? You will need to represent these in matrix form

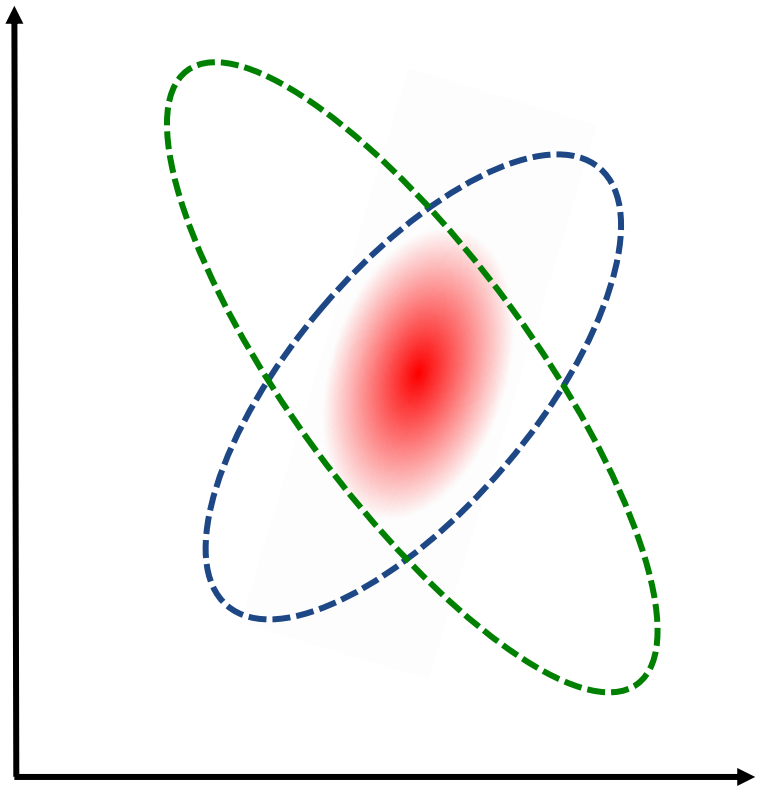
$$k = \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2}$$

$$\mu = \mu_1 + k\left(\mu_2 - \mu_1\right) \text{ and } \sigma^2 = \sigma_1^2 - k\sigma_1^2$$

$$\mathbf{K} = \Sigma_1\left(\Sigma_1 + \Sigma_2\right)^{-1}$$

$$\vec{\mu} = \vec{\mu}_1 + \mathbf{K}\left(\vec{\mu}_2 - \vec{\mu}_1\right)$$

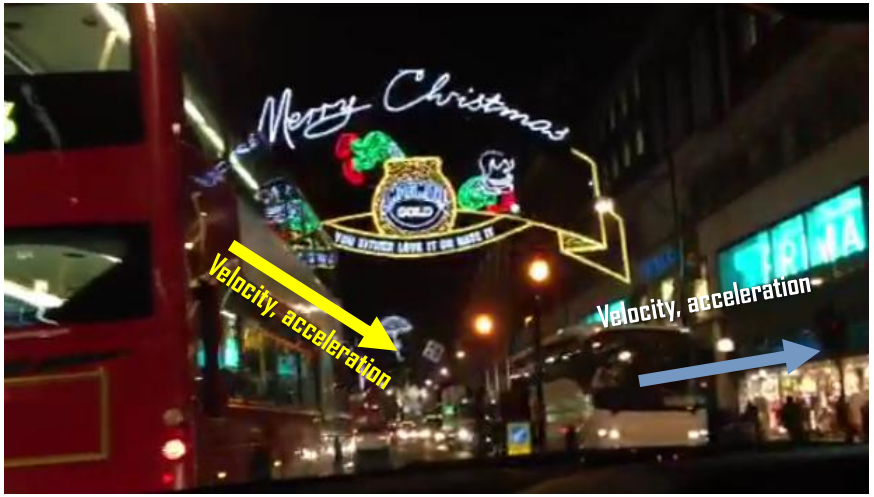
$$\Sigma = \Sigma_1 - \mathbf{K}\Sigma_1$$

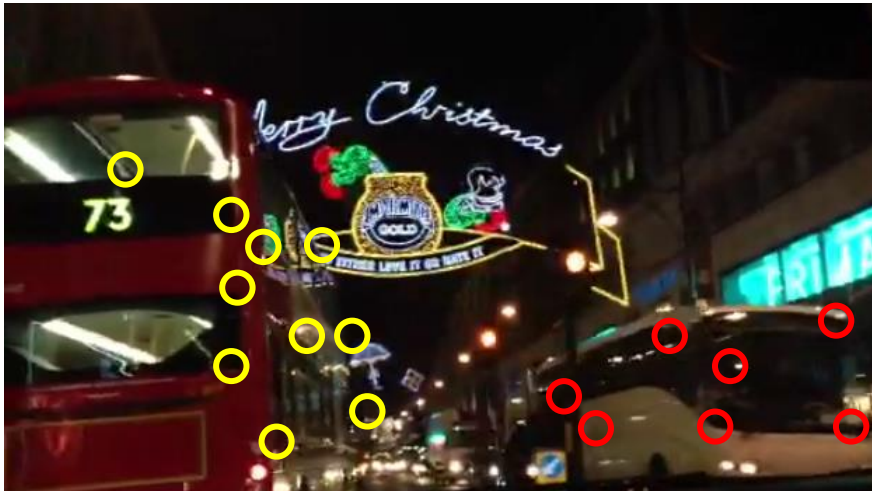


In practice, this means that **for the same variable**, if we can take two **different estimates**, each has its uncertainty, the combined result, although still uncertain, but will have improved accuracy

Kalman Filter

- It is a recursive algorithm that provides an optimal estimation method for linear dynamic systems with white Gaussian noise
- It describes the system with a **system dynamics (state) model** and a **measurement (observation)** model



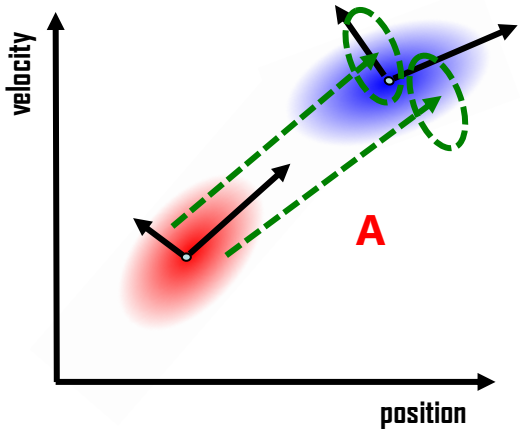


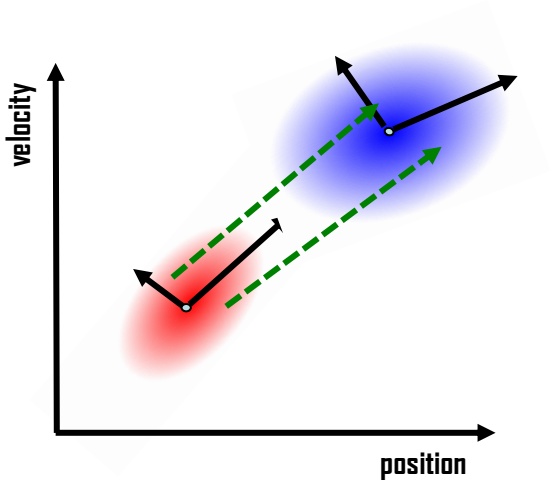
We know something about the dynamics of the objects

We also can measure something from the images (observations)

Kalman Filter

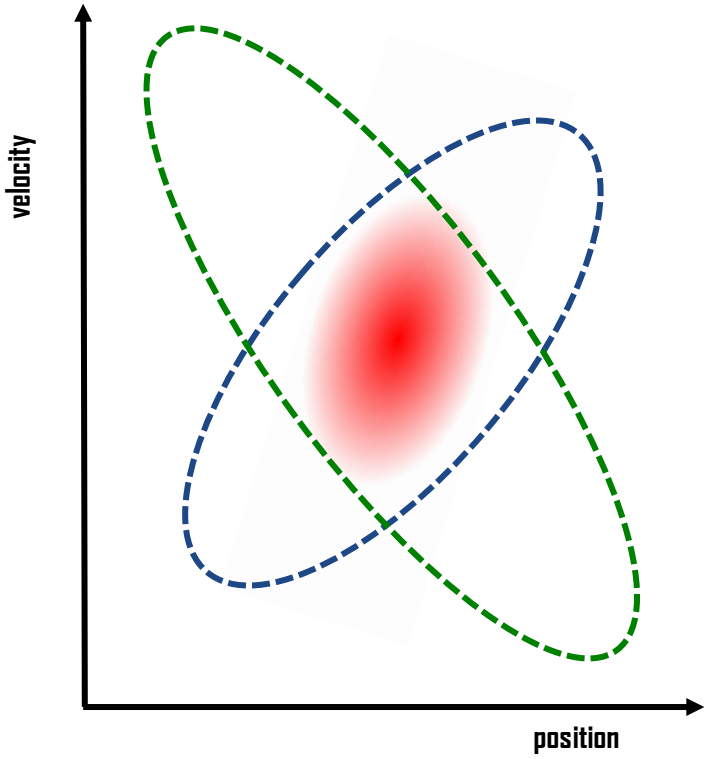
- A state vector X_{k-1} is composed of variables which are correlated with covariance matrix P_{k-1}
- Our aim is to use the **current state** and **predict the next state X_k**.
- A transition matrix **A** is introduced which takes every point in our original estimate and moves it to a new predicted location $X_k=A_kX_{k-1}$
- The covariance matrix is updated as $P_k=A_kP_{k-1}A'_k$
- We can model the uncertainty associated with external forces by adding one more term to the state $X_k=A_kX_{k-1}+B_kU_{k-1}$ and covariance matrix $P_k=A_kP_{k-1}A'_k+Q_k$



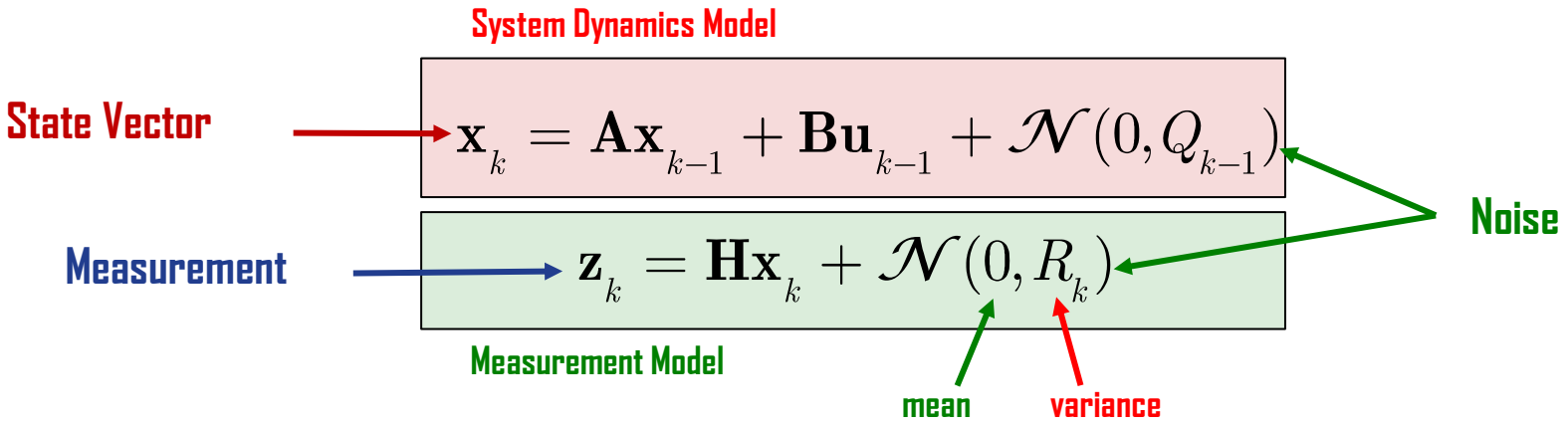


Kalman Filter

- We might have several sensors which give us information about the state of our system.
- The distribution of sensor readings we would expect to see is $\mathbf{H_k X_k}$ with covariance $\mathbf{H_k P_k H'_k}$
- The readings which we observe from the sensors $\mathbf{Z_k}$ will have covariance $\mathbf{R_k}$
- To combine these two estimates, we just need to multiply the distributions together.
- The mean of this distribution is the configuration for which **both estimates are most likely**, and is therefore the **best guess** of the true configuration given all the information we have.



Kalman Filter



$\mathbf{x}_i$ – the state vector containing the terms of interest for the objects (position, velocity, heading, acceleration etc);

$\mathbf{u}_i$ – the vector containing control inputs (steering, breaking etc);

A – called state transition matrix, determines the effect of each system state parameter at time t-1 on the system state at t (e.g. how velocity and acceleration will affect the position between image frames)

B – the control input matrix (e.g. how external force can affect the movement of the object)

$\mathbf{z}_i$ – the measurements we can make from the images

H – the transition matrix that maps the state vectors to the measurement space (e.g. projection of the object features onto the image space)

Kalman Filter

- The Kalman filter estimates the state vectors iteratively by following these two steps:

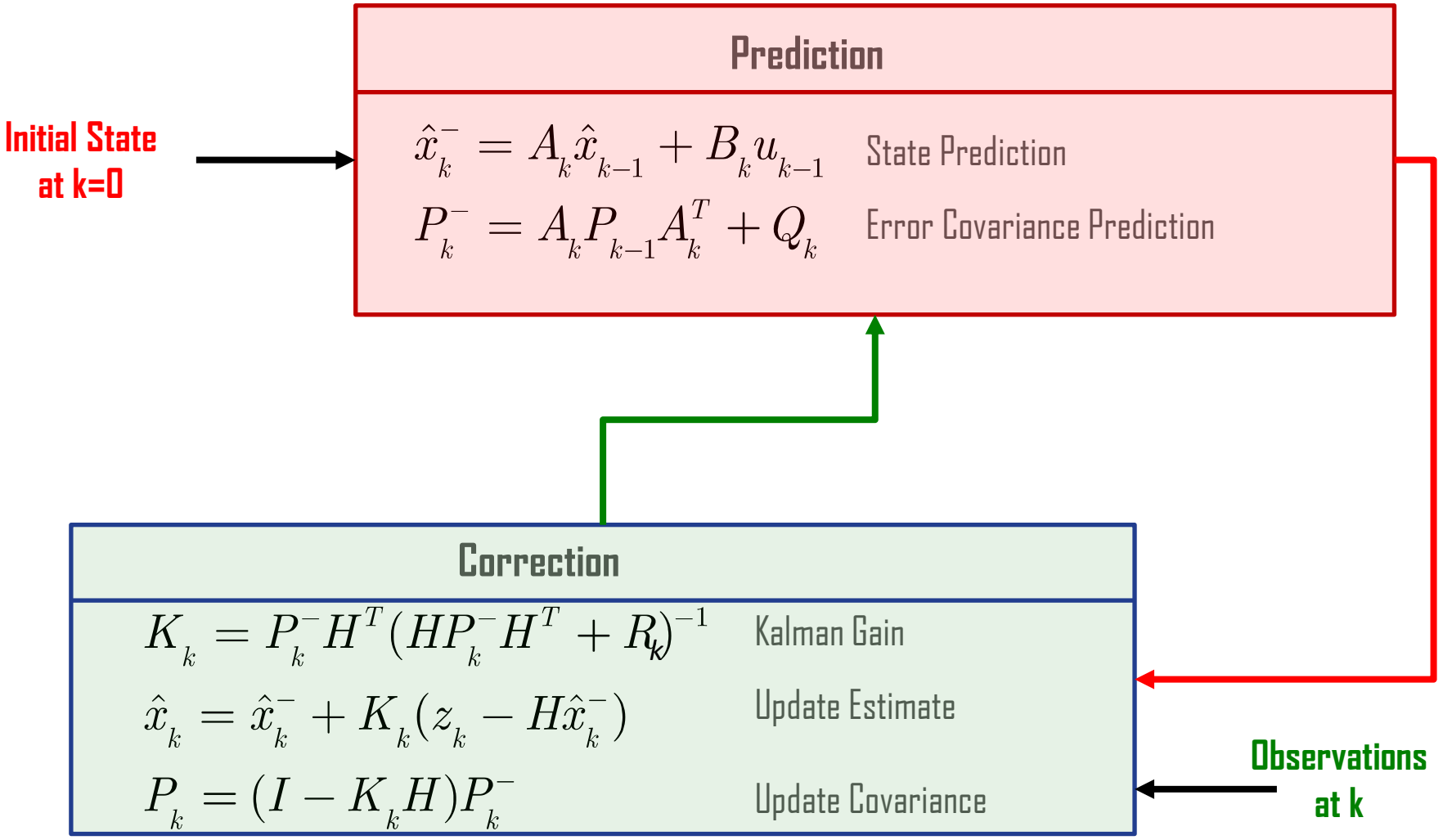
  - **Prediction**

    The state estimate and error covariance matrix at time t_k is predicted given the input measurements up to t_{k-1}

  - **Correction**

    Update the state estimate and error covariance matrices with the observation model

Kalman Filter



Kalman Filter Example

Suppose that we want to track an image feature detected at (x,y) in a video sequence. We assume that the feature is moving in the image with constant velocity.

System State Model can be written as

$$\mathbf{x}_k = \mathbf{A}\mathbf{x}_{k-1} + \mathcal{N}(0, Q_{k-1})$$

$$\begin{bmatrix} x_k \\ y_k \\ \dot{x}_k \\ \dot{y}_k \end{bmatrix} = \begin{bmatrix} 1 & 0 & \Delta_t & 0 \\ 0 & 1 & 0 & \Delta_t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ \dot{x}_{k-1} \\ \dot{y}_{k-1} \end{bmatrix} + \mathcal{N}(0, Q_{k-1})$$

Kalman Filter - Example

As what we can detect from the image is the location of the feature, the measurement model will be:

$$\mathbf{z}_k = \mathbf{H}\mathbf{x}_k + \mathcal{N}(0, R_k)$$

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

At each frame, the predicted and corrected feature positions are estimated. The estimated error covariance matrix gives us information about the accuracy of the state estimate (e.g. the uncertainty of the tracked position of the point in the image plane).

Practical Examples

Extended Kalman Filters (EKF)

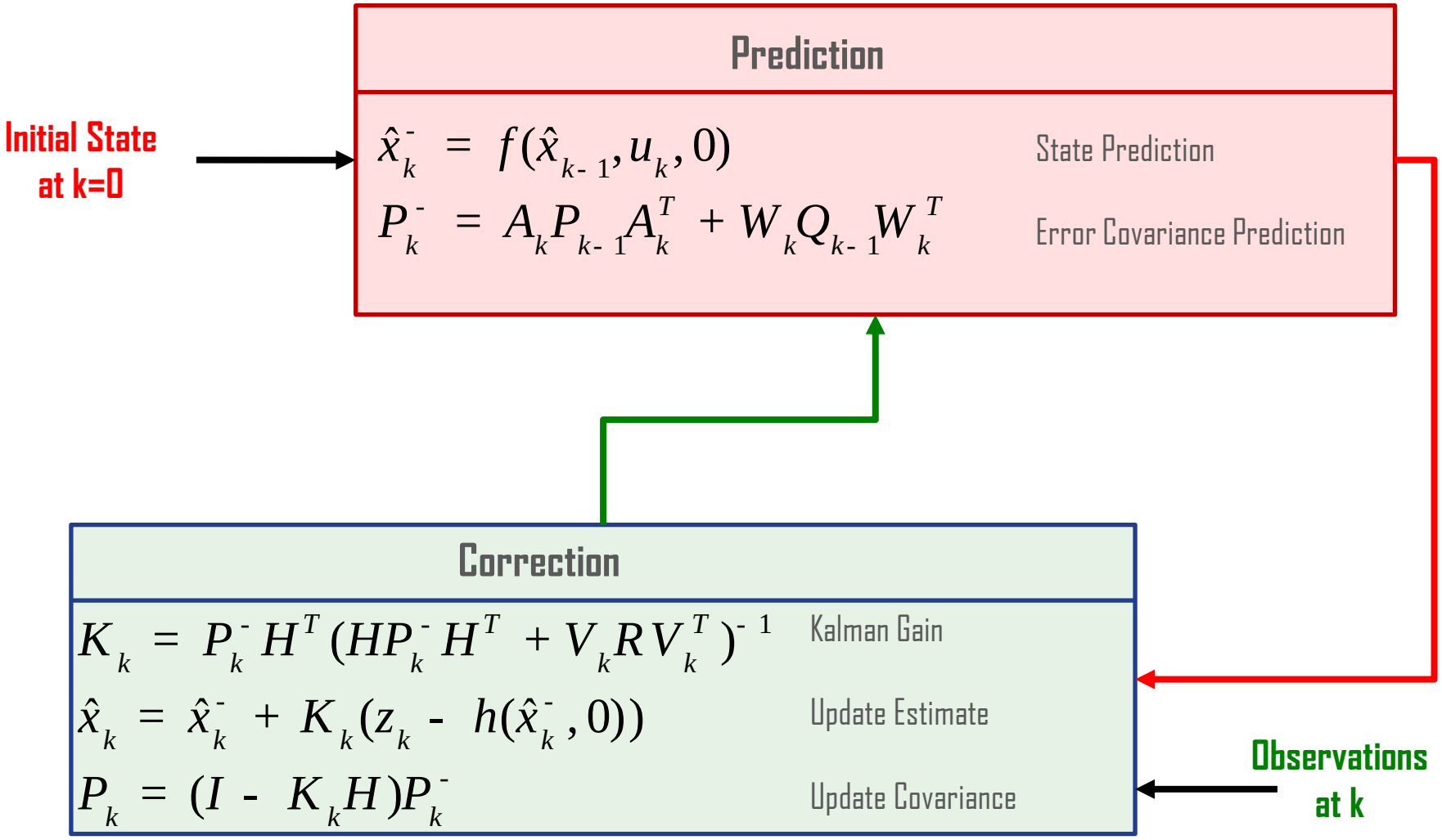
- Non-linear extension of the original Kalman filter
- The non-linear behaviours of the system's dynamics are approximated by local linearisation around the last state estimate
- Each iteration of the EKF consists of
  - The last filtered state estimate
  - Linearisation of the system dynamics around the last state estimate
  - Prediction of the Kalman filter to the linearised system dynamics just obtained
  - Linearise the observation model around the prediction
  - Apply the filtering or update part of the Kalman filter to the linearised observation model
- It is the EKF that we normally use for vision, particularly for complex scenes (e.g. robotic navigation in surgery)

$$x_k = f(x_{k-1}, u_{k-1}, w_{k-1})$$
$$z_k = h(x_k, v_k)$$

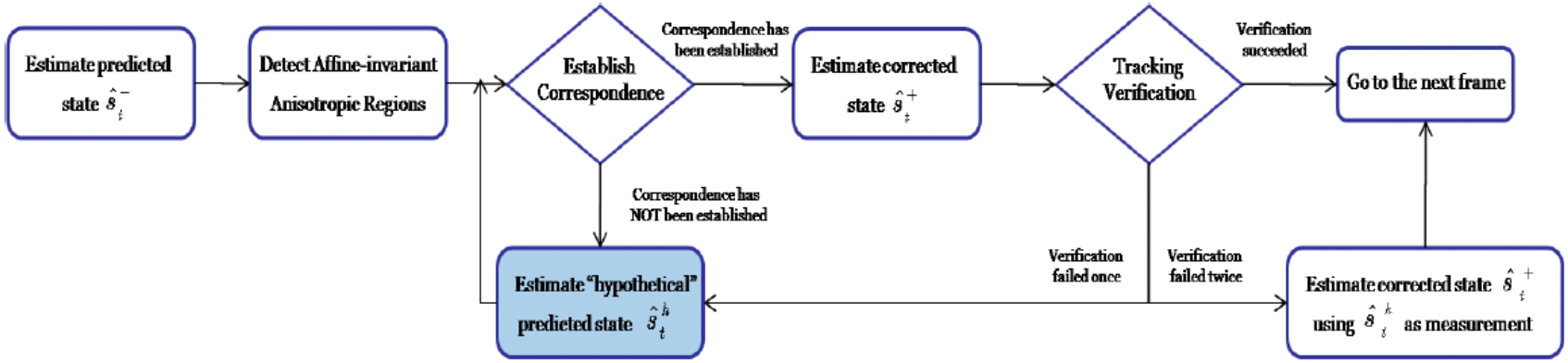
$$x_k \approx \widetilde{x_k} + A(x_{k-1} - \widehat{x_{k-1}}) + W w_{k-1}$$
$$z_k \approx \widetilde{z_k} + H(x_k - \widetilde{x_k}) + V v_k$$

- *A* is the Jacobian matrix of partial derivatives of f with respect to *x*
- *W* is the Jacobian matrix of partial derivatives of f with respect to *w*
- *H* is the Jacobian matrix of partial derivatives of h with respect to *x*
- *V* is the Jacobian matrix of partial derivatives of h with respect to *v*

Extended Kalman Filter

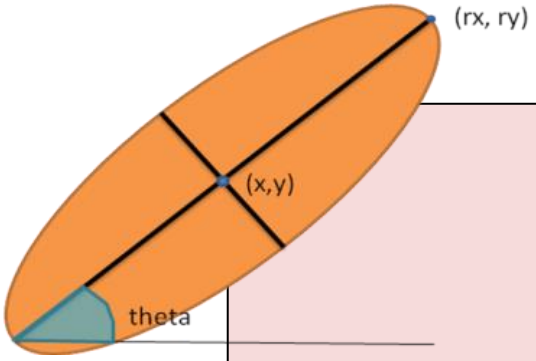


Probabilistic Tissue Tracking



S. Giannarou, M. Visentini-Scarzanella, G.Z Yang, "Probabilistic Tracking of Affine-Invariant Anisotropic Regions", IEEE Trans. Pattern Anal. Machine Intell., 35 (1), pp. 130-143, 2013

Probabilistic Tissue Tracking



System Dynamics Model

$$s_t = f(s_{t-1}, w_t) = \begin{bmatrix} x_t \\ y_t \\ u_t \\ v_t \\ \theta_t \\ \omega_t \\ r_t^x \\ r_t^y \\ k_t \end{bmatrix} = \begin{bmatrix} x_{t-1} + (u_{t-1} + w_{t-1}^u) \\ y_{t-1} + (v_{t-1} + w_{t-1}^v) \\ u_{t-1} + w_{t-1}^u \\ v_{t-1} + w_{t-1}^v \\ \theta_{t-1} + \omega_{t-1} + w_{t-1}^\omega \\ \omega_{t-1} + w_{t-1}^\omega \\ r_t^x \\ r_t^y \\ k_{t-1} \end{bmatrix}$$

$$\begin{bmatrix} r_t^x \\ r_t^y \end{bmatrix} = \begin{bmatrix} \cos(\omega_{t-1} + w_{t-1}^\omega) & -\sin(\omega_{t-1} + w_{t-1}^\omega) \\ \sin(\omega_{t-1} + w_{t-1}^\omega) & \cos(\omega_{t-1} + w_{t-1}^\omega) \end{bmatrix} \cdot \begin{bmatrix} r_{t-1}^x - x_{t-1} \\ r_{t-1}^y - y_{t-1} \end{bmatrix} + \begin{bmatrix} u_{t-1} + w_{t-1}^u + x_{t-1} \\ v_{t-1} + w_{t-1}^v + y_{t-1} \end{bmatrix}$$

S. Giannarou, M. Visentini-Scarzanella, G.Z Yang, "Probabilistic Tracking of Affine-Invariant Anisotropic Regions", IEEE Trans. Pattern Anal. Machine Intell., 35 (1), pp. 130-143, 2013

Probabilistic Tissue Tracking

Measurement Model

$$z_t = V s_t + \eta_t = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_t \\ y_t \\ u_t \\ v_t \\ \theta_t \\ \omega_t \\ r_t^x \\ r_t^y \\ k_t \end{bmatrix} + \eta_t$$

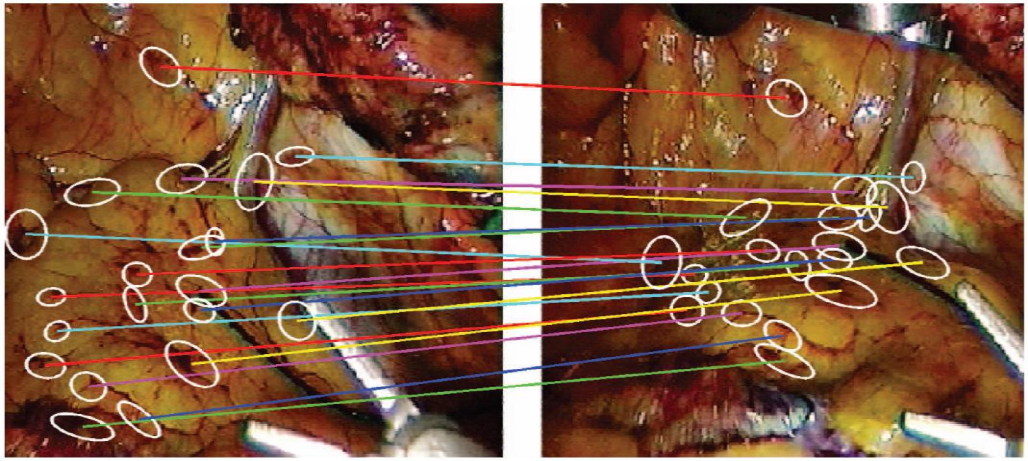
State Correction

$$\hat{s}_t^+ = \hat{s}_t^- + K_t(z_t - H\hat{s}_t^-)$$

S. Giannarou, M. Visentini-Scarzanella, G.Z Yang, "Probabilistic Tracking of Affine-Invariant Anisotropic Regions", IEEE Trans. Pattern Anal. Machine Intell., 35 (1), pp. 130-143, 2013

Probabilistic Tissue Tracking

To establish feature correspondences, the relative amount of overlap in the image area covered by the compared regions and the dissimilarity in c(x) of the compared features is used. Two regions correspond if the overlap error and dissimilarity in c(x) are sufficiently small.

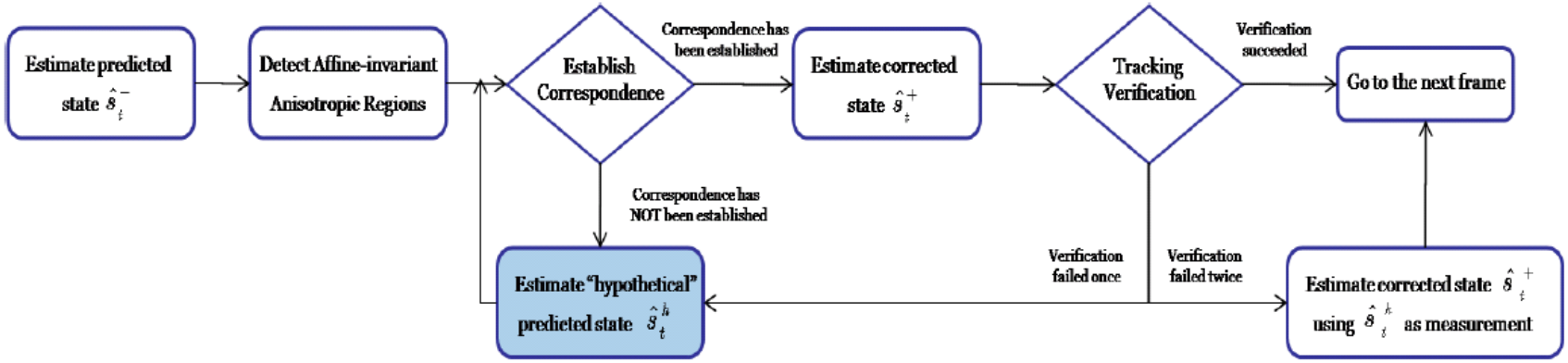


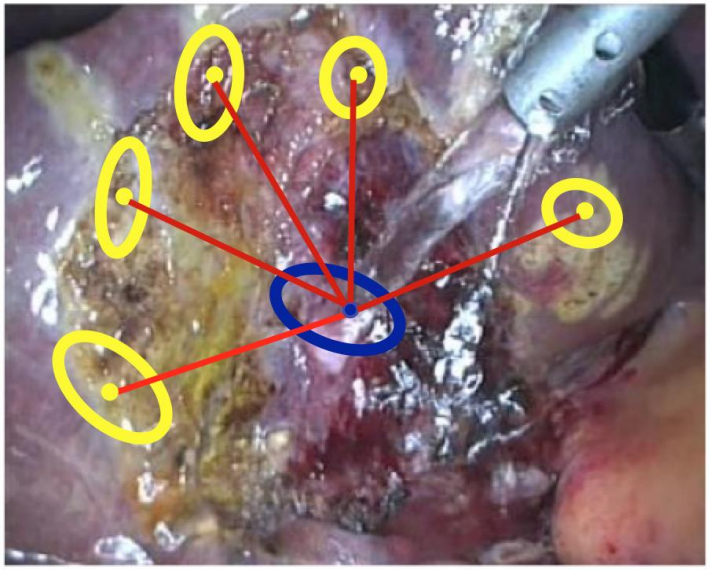
$$OE_{A,B} = 1 - \frac{A \cap B}{A \cup B}$$

$$CD_{A,B} = \frac{|c_A - c_B|}{\max_{n \in \Pi} |c_A - c_n|}$$

S. Giannarou, M. Visentini-Scarzanella, G.Z Yang, "Probabilistic Tracking of Affine-Invariant Anisotropic Regions", IEEE Trans. Pattern Anal. Machine Intell., 35 (1), pp. 130-143, 2013

Probabilistic Tissue Tracking





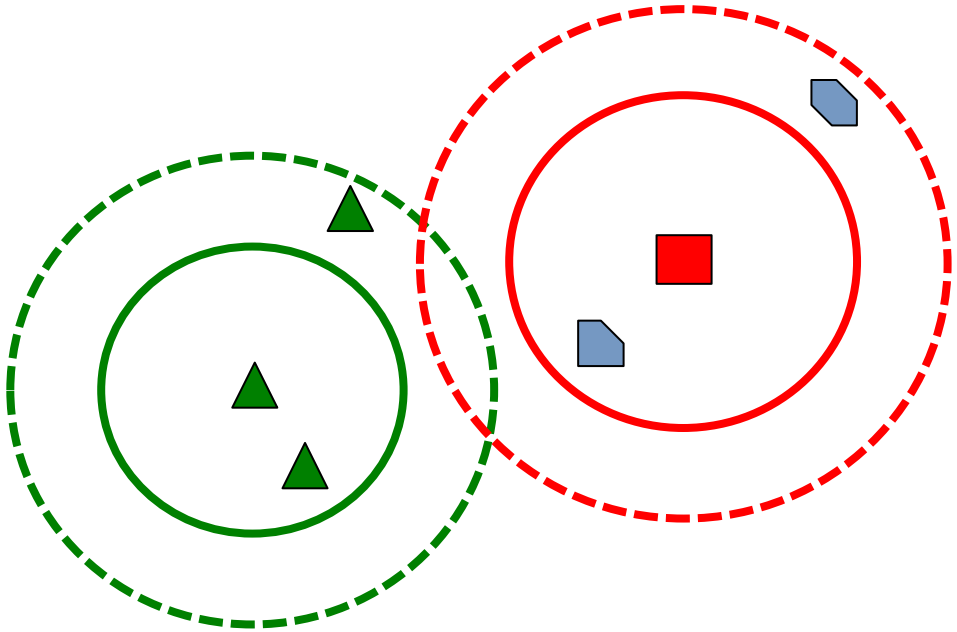
$$\hat{s}_t^h = \begin{bmatrix} \tilde{x}_t \\ \tilde{y}_t \\ \tilde{x}_t - \hat{x}_{t-1}^+ \\ \tilde{y}_t - \hat{y}_{t-1}^+ \\ \hat{\theta}_{t-1} \\ \hat{\omega}_{t-1} \\ \hat{r}_{t-1}^x + (\tilde{x}_t - \hat{x}_{t-1}^+) \\ \hat{r}_{t-1}^y + (\tilde{y}_t - \hat{y}_{t-1}^+) \\ \hat{k}_{t-1} \end{bmatrix}$$

S. Giannarou, M. Visentini-Scarzanella, G.Z Yang, "Probabilistic Tracking of Affine-Invariant Anisotropic Regions", IEEE Trans. Pattern Anal. Machine Intell., 35 (1), pp. 130-143, 2013

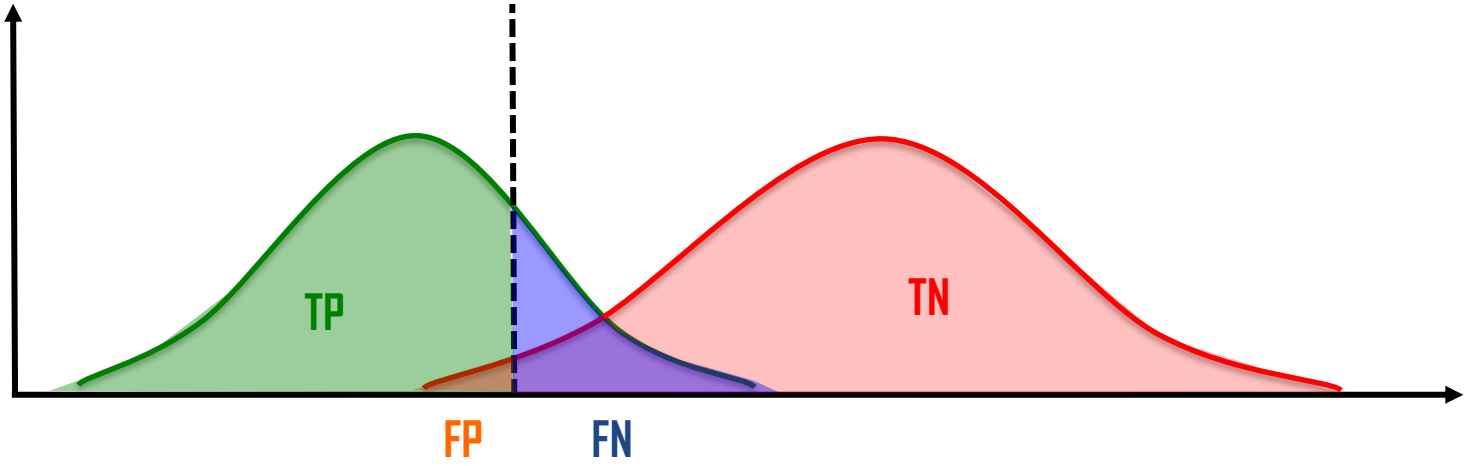
Tracking Errors

Matching and Errors

- **TP:** true positives (correct matches)
- **FN:** false negatives (valid matches are wrongly discarded)
- **FP:** false positives (wrong matches are incorrectly accepted)
- **TN:** true negatives (wrong matches are rejected correctly)



Calculation of Error Rates



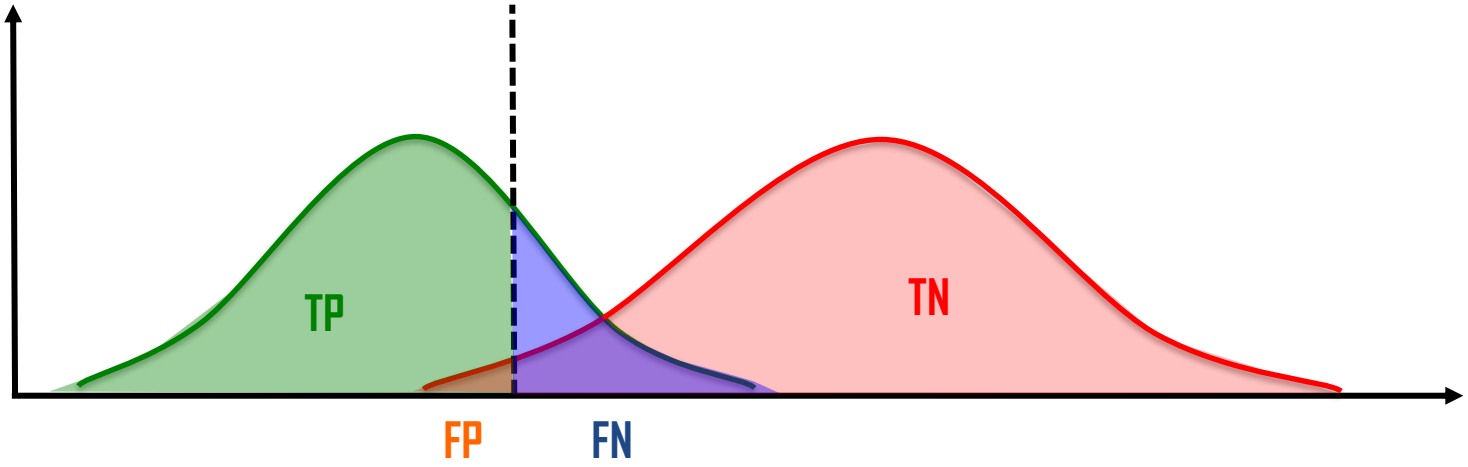
$$\text{True positive rate (TPR)} = \frac{TP}{TP + FN} = \frac{TP}{P}$$ Recall

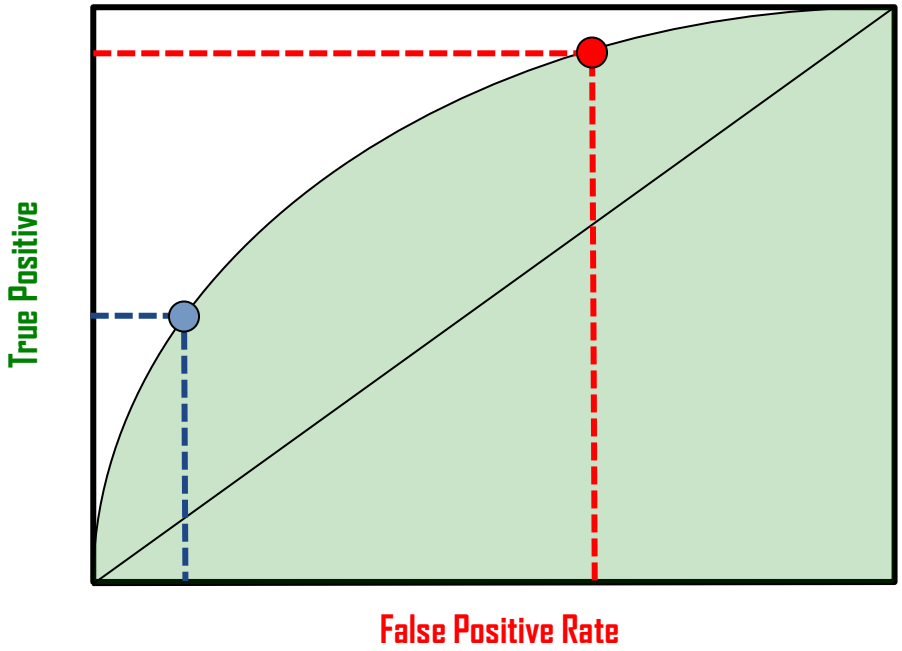
$$\text{False positive rate (FPR)} = \frac{FP}{FP + TN} = \frac{FP}{N}$$

$$\text{Positive predictive value (PPV)} = \frac{TP}{TP + FP}$$ Precision

$$\text{Accuracy (ACC)} = \frac{TP + TN}{P + N}$$

ROC – Receiver Operating Characteristic Analysis





Conclusions

- Tracking in Image Sequences
- Lucas-Kanade Tracker and Multiscale Implementation
- The need for incorporating temporal information for tracking
- Kalman filter
- Extended Kalman filter and other techniques
- Tracking, error matrices and error rates
- Receiver Operating Characteristic (ROC) analysis