Computer Vision

18AI742

Dr. rer. nat. D. Antony Louis Piriyakumar,
Dean (R&D),
Cambridge Institute of Technology.

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

Tracking

Dr. D. Antony Louis Piriyakumar,
*Dean (*Research & Development)
Registered Indian patent agent (IN/PA 3041)

www.cambridge.edu.in

CAMBRIDGE INSTITUTE OF TECHNOLOGY

Tracking

Contents

1) Simple tracking strategies
2) Tracking using matching
3) Tracking linear dynamical models with Kalman filters
4) Conclusion
5) Q&A

$$\nabla \cdot \mathbf{E} = \rho/\varepsilon_0$$

$$\nabla \cdot \mathbf{B} = 0$$

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}$$

$$\nabla \times \mathbf{B} = \mu_0 \varepsilon_0 \frac{\partial \mathbf{E}}{\partial t} + \mu_0 \boldsymbol{j}_c$$

where

$$\nabla = \hat{\mathbf{i}} \frac{\partial}{\partial x} + \hat{\mathbf{j}} \frac{\partial}{\partial y} + \hat{\mathbf{k}} \frac{\partial}{\partial z}$$

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

Tracking

- Problem of generating an inference about motion of object in a sequence of images.
- Some measurements that appearat each tick of a (notional) clock.
- Encoding of the object's state and some model of how this state changes from tick to tick
- Infer the state of the world from the measurements and the model of dynamics.

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

Important applications

- Motion Capture: controlling a cartoon character, thousands of virtual extras in a crowd scene, or a virtual stunt avatar
- Recognition from Motion: able to determine the identity of the object from its motion
- Surveillance: monitor activities and give a warning
- Targeting:deciding what to shoot, and (b) hitting it

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

State of an object

- Encodes all the properties of the object needed to encode its motion
- position and velocity; position, velocity, and acceleration; position and appearance;
- new measurements that depend on the new state referred to as observations
- Model of how state of object changes with time - object's dynamics.
- Tracking involves exploiting both observations and dynamics to infer state.

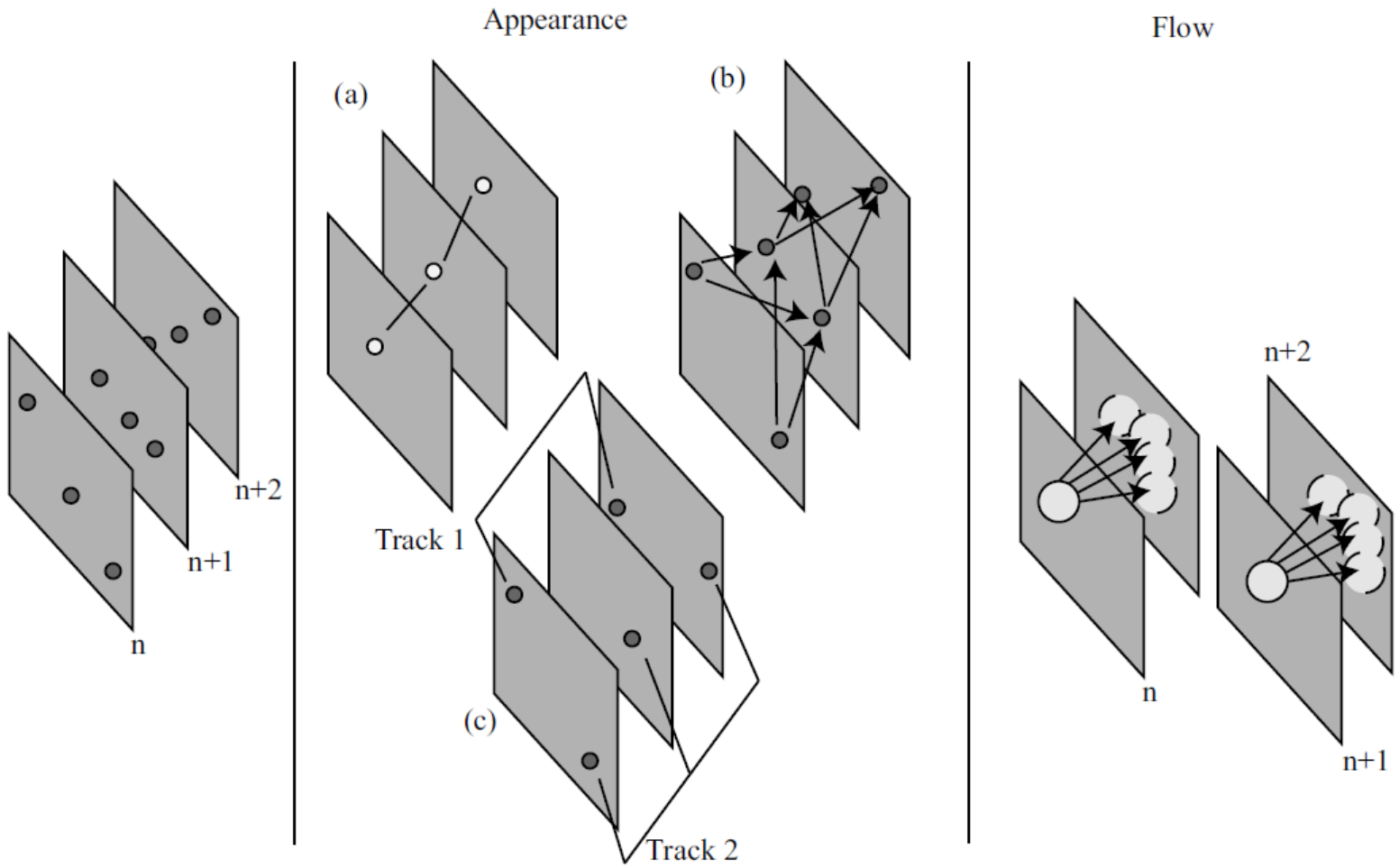
CAMBRIDGE
INSTITUTE OF TECHNOLOGY

Differentiations

$\mathbf{r}$		Position
$\frac{d\mathbf{r}}{dt}$	$= \dot{\mathbf{r}}$	Velocity (speed)
$\frac{d^2\mathbf{r}}{dt^2}$	$= \ddot{\mathbf{r}}$	Acceleration
$\frac{d^3\mathbf{r}}{dt^3}$	$= \dddot{\mathbf{r}}$	Jerk
$\frac{d^4\mathbf{r}}{dt^4}$	$= \ddddot{\mathbf{r}}$	Snap (jounce)
$\frac{d^5\mathbf{r}}{dt^5}$	$= \overset{\cdot\cdot\cdot\cdot\cdot}{\mathbf{r}}$	Crackle
$\frac{d^6\mathbf{r}}{dt^6}$	$= \overset{\cdot\cdot\cdot\cdot\cdot\cdot}{\mathbf{r}}$	Pop

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

In tracking problems, building space time paths followed by tokens



CAMBRIDGE
INSTITUTE OF TECHNOLOGY

Track

- Represents a timeline for a single object - Track
- copy the tracks from the previous frame to this frame
- then allocate object detector responses to tracks
- Each track will get at most one detector response
- Each detector response will get at most one track

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

Algorithm 11.1: Tracking by Detection

Notation:
Write $\mathbf{x}_k(i)$ for the k'th response of the detector in the ith frame
Write $t(k, i)$ for the k'th track in the ith frame
Write $*t(k, i)$ for the detector response attached to the k'th track in the ith frame
(Think C pointer notation)

Assumptions: We have a detector which is reasonably reliable.
We know some distance d such that $d(*t(k, i-1), *t(k, i))$ is always small.

First frame: Create a track for each detector response.

N'th frame:
Link tracks and detector responses by solving a bipartite matching problem.
Spawn a new track for each detector response not allocated to a track.
Reap any track that has not received a detector response for some number of frames.

Cleanup: We now have trajectories in space time. Link anywhere this is justified (perhaps by a more sophisticated dynamical or appearance model, derived from the candidates for linking).

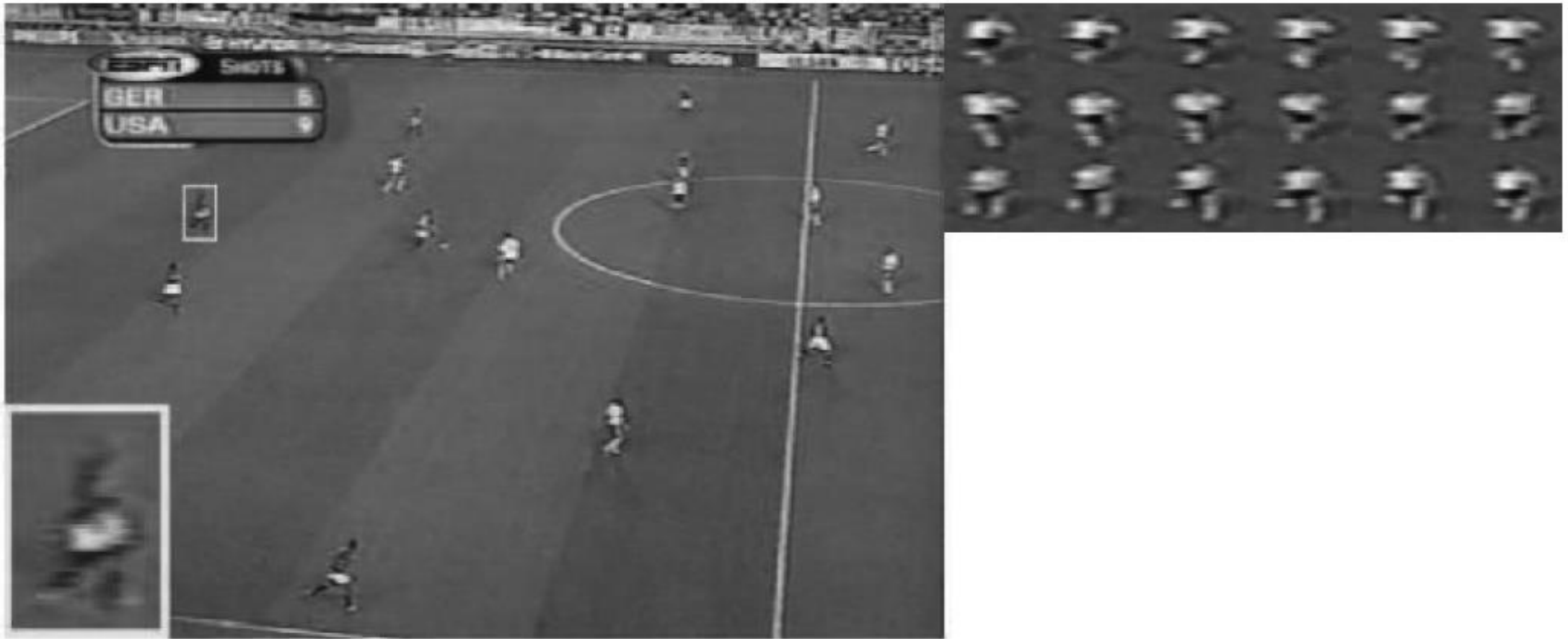
CAMBRIDGE
INSTITUTE OF TECHNOLOGY

Tracking Translations by Matching

- Model a player's motion with two components
- The first is the absolute motion of a box fixed around the player
- The second is the player's movement relative to that box
- stabilize a box around the object to contain all visual information
- Image stabilization

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

Stabilize an image box around a more interesting structure



Good features to track

$\mathcal{R}^{(n)}$ for the rectangle in the nth frame, $\mathcal{R}_{ij}^{(n)}$ for the i, jth pixel $\sum_{i,j}(\mathcal{R}_{ij}^{(n)} - \mathcal{R}_{ij}^{(n+1)})^2$

$$E(\boldsymbol{h}) = \sum_{\boldsymbol{u} \in \mathcal{P}_t} [I(\boldsymbol{u}, t) - I(\boldsymbol{u} + \boldsymbol{h}, t + 1)]^2$$

as a function of $\boldsymbol{h}$. The minimum of the error occurs when

$$\nabla_h E(\boldsymbol{h}) = 0.$$

Now if $\boldsymbol{h}$ is small, we can write $I(\boldsymbol{u} + \boldsymbol{h}, t + 1) \approx I(\boldsymbol{u}, t) + \boldsymbol{h}^T \nabla I$, where ∇I is the

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

Good features to track

$\mathcal{R}^{(n)}$ for the rectangle in the nth frame, $\mathcal{R}_{ij}^{(n)}$ for the i, jth pixel $\sum_{i,j}(\mathcal{R}_{ij}^{(n)} - \mathcal{R}_{ij}^{(n+1)})^2$

$$E(\boldsymbol{h}) = \sum_{\boldsymbol{u}\in\mathcal{P}_t} [I(\boldsymbol{u},t) - I(\boldsymbol{u}+\boldsymbol{h},t+1)]^2$$

as a function of $\boldsymbol{h}$. The minimum of the error occurs when

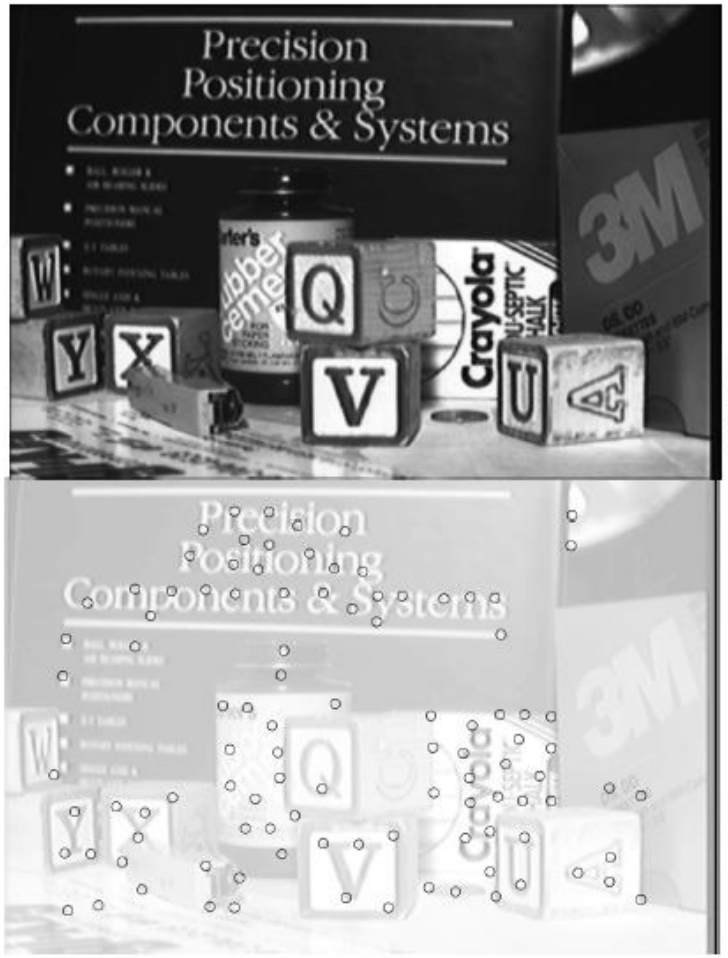
$$\nabla_h E(\boldsymbol{h}) = 0.$$

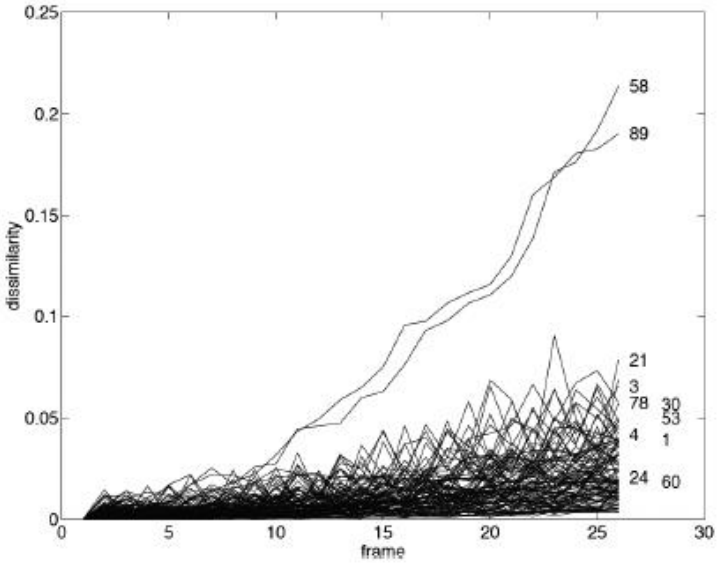
Now if $\boldsymbol{h}$ is small, we can write $I(\boldsymbol{u}+\boldsymbol{h},t+1) \approx I(\boldsymbol{u},t) + \boldsymbol{h}^T\nabla I$, where ∇I is the image gradient. Substituting, and rearranging, we get

$$\left[\sum_{\boldsymbol{u}\in\mathcal{P}_t} (\nabla I)(\nabla I)^T\right] \boldsymbol{h} = \sum_{\boldsymbol{u}\in\mathcal{P}_t} [I(\boldsymbol{u},t) - I(\boldsymbol{u},t+1)]\,\nabla I,$$

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

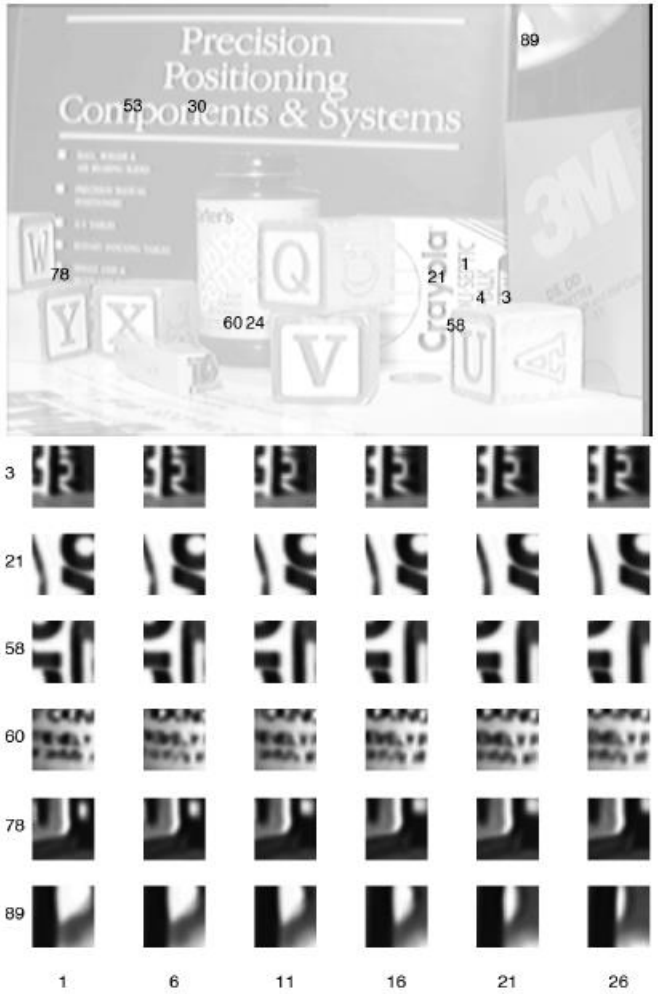
Better test to identify good tracks

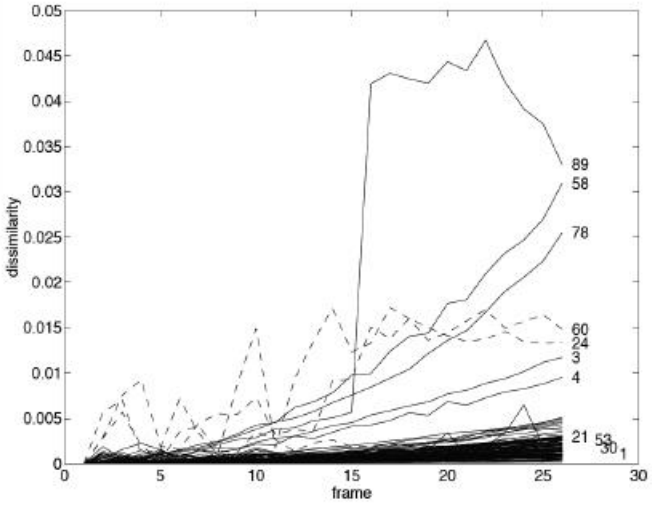




CAMBRIDGE
INSTITUTE OF TECHNOLOGY

Pattern in the neighborhood deforms





CAMBRIDGE
INSTITUTE OF TECHNOLOGY

Using affine transformations to confirm a match

- The patch is rotating in 3D, by checking the patch in frame n + 1 against frame 1
- Because the image patch is small, an affine model is appropriate.
- Affine model means that point x in frame 1 will become point Mx + c in frame t.
- Find location of it in next frame, and check whether patch matches original one

CAMBRIDGE INSTITUTE OF TECHNOLOGY

Tracking

Contents

1) Simple tracking strategies
2) Tracking using matching
3) Tracking linear dynamical models with Kalman filters
4) Conclusion
5) Q&A

$$\nabla \cdot \mathbf{E} = \rho/\varepsilon_0$$

$$\nabla \cdot \mathbf{B} = 0$$

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}$$

$$\nabla \times \mathbf{B} = \mu_0 \varepsilon_0 \frac{\partial \mathbf{E}}{\partial t} + \mu_0 \boldsymbol{j}_c$$

where

$$\nabla = \hat{\mathbf{i}} \frac{\partial}{\partial x} + \hat{\mathbf{j}} \frac{\partial}{\partial y} + \hat{\mathbf{k}} \frac{\partial}{\partial z}$$

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

Summary matching

- a domain of interest in the nth image, Dn, and we must search for a matching
- domain Dn+1 in the n + 1st image with complex motion model
- match summary representations of the whole domain
- represent a domain with a set of parameters
- summary of appearance within circle Dn and find the best-matching circle Dn+1

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

Flow-based matching

- Search for transformation of pixels in old domain that produces set of pixels that match well
- allows to exploit strong motion models
- To get a good new domain (matching one here)
- Matching is done despite the problems associated in the flow

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

Possible problems in flow with an example

- From frame to frame, the player's back is seen at different viewing angles.
- Individual pixels in one domain might have no corresponding pixels in the next.
- cloth may have folded slightly, motion blur in some frames,
- domain is largely white, with some yellow patches
- Summary representation of the domain might not change from frame to frame

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

Bhattacharyya coefficient

histogram can be represented as a vector of bin counts, and we write this vector as $\boldsymbol{p}(\boldsymbol{y})$; its uth component representing the count in the u'th bin is $p_u(\boldsymbol{y})$. We wish to find the $\boldsymbol{y}$ whose histogram is closest to that at $\boldsymbol{y}_n$. We are comparing two probability distributions, which we can do with the *Bhattacharyya coefficient*:

$$\rho(\boldsymbol{p}(\boldsymbol{y}), \boldsymbol{p}(\boldsymbol{y}_n)) = \sum_u \sqrt{p_u(\boldsymbol{y}) p_u(\boldsymbol{y}_n)}.$$

This will be one if the two distributions are the same and near zero if they are very different. To obtain a distance function, we can work with

$$d(\boldsymbol{p}(\boldsymbol{y}), \boldsymbol{p}(\boldsymbol{y}_n)) = \sqrt{1 - \rho(\boldsymbol{p}(\boldsymbol{y}), \boldsymbol{p}(\boldsymbol{y}_n))}.$$

CAMBRIDGE INSTITUTE OF TECHNOLOGY

Bhattacharyya coefficient

We will obtain $\boldsymbol{y}_{n+1}$ by minimizing this distance. We will start this search at $\boldsymbol{y}_{n+1}^{(0)}$. We assume that $\boldsymbol{y}_{n+1}$ is close to $\boldsymbol{y}_{n+1}^{(0)}$, and as a result, $\boldsymbol{p}(\boldsymbol{y}_{n+1})$ is similar to $\boldsymbol{p}(\boldsymbol{y}_{n+1}^{(0)})$. In this case, a Taylor expansion of $\rho(\boldsymbol{p}(\boldsymbol{y}), \boldsymbol{p}(\boldsymbol{y}_n))$ about $\boldsymbol{p}(\boldsymbol{y}_{n+1}^{(0)})$ gives

$$\begin{aligned}
\rho(\boldsymbol{p}(\boldsymbol{y}), \boldsymbol{p}(\boldsymbol{y}_n)) &\approx \sum_u \sqrt{p_u(\boldsymbol{y}_{n+1}^{(0)}) p_u(\boldsymbol{y}_n)} + \\
&\quad \sum_u (p_u(\boldsymbol{y}) - p_u(\boldsymbol{y}_{n+1}^{(0)})) \left(\frac{1}{2} \sqrt{\frac{p_u(\boldsymbol{y}_n)}{p_u(\boldsymbol{y}_{n+1}^{(0)})}} \right) \\
&= \frac{1}{2} \sum_u \sqrt{p_u(\boldsymbol{y}_{n+1}^{(0)}) p_u(\boldsymbol{y}_n)} + \frac{1}{2} \sum_u p_u(\boldsymbol{y}) \sqrt{\frac{p_u(\boldsymbol{y}_n)}{p_u(\boldsymbol{y}_{n+1}^{(0)})}}.
\end{aligned}$$

This means that, to minimize the distance, we must maximize

$$\frac{1}{2} \sum_u p_u(\boldsymbol{y}) \sqrt{\frac{p_u(\boldsymbol{y}_n)}{p_u(\boldsymbol{y}_{n+1}^{(0)})}}. \tag{11.1}$$

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

Kernel smoother

- Method to construct a histogram vector for the circle with center y.
- Tracking deforming object, pixels far away from center of two matching circles quite different
- Allow pixels far away from center to have much smaller effect on the histogram
- That is what is done by the kernel smoother

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

Kernel smoother

- feature vector (color) for the pixel at location xi in the circle as fi(n)
- the histogram bin corresponding to fi(n) as b(fi(n)).
- Each pixel votes into its bin in the histogram with a weight
- that decreases with ||xi – y || according to a kernel profile k

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

Kernel smoother

Section 9.3.4). Using this approach, the fraction of total votes in bin u produced by all features is

$$p_u(\boldsymbol{y}) = C_h \sum_{i \in \mathcal{D}_n} k(\| \frac{\boldsymbol{x}_i - \boldsymbol{y}}{h} \|^2) \delta \left[b(\boldsymbol{f}_i - u) \right]. \tag{11.2}$$

where h is a scale, chosen by experiment, and C_h is a normalizing constant to ensure that the sum of histogram components is one. Substituting Equation 11.2

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

Kernel smoother

into Equation 11.1, we must maximize

$$f(\boldsymbol{y}) = \frac{C_h}{2} \sum_i w_i k(\| \frac{\boldsymbol{x}_i - \boldsymbol{y}}{h} \|^2), \tag{11.3}$$

where

$$w_i = \sum_u \delta \left[b(\boldsymbol{f}_i - u) \right] \sqrt{\frac{p_u(\boldsymbol{y}_n)}{p_u(\boldsymbol{y}_{n+1}^{(0)})}}.$$

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

Kernel smoother (Mean shift)

$$y^{(j+1)} = \frac{\sum_i w_i x_i g\left(\left\| \frac{x_i - y^{(j)}}{h} \right\|^2\right)}{\sum_i w_i g\left(\left\| \frac{x_i - y^{(j)}}{h} \right\|^2\right)}.$$

CAMBRIDGE INSTITUTE OF TECHNOLOGY

Tracking with the Mean Shift Algorithm

Assume we have a sequence of N images; a domain $\mathcal{D}_1$,
in the first image represented by parameters
$\boldsymbol{y}_1$ (for a circular domain of fixed size, these would be the
location of the center; for a square, the center and edge length; and so on);
a kernel function k; a scale h; and a feature representation $\boldsymbol{f}$ of each pixel.

For $n \in [1, \ldots, N-1]$

Obtain an initial estimate $\boldsymbol{y}_{n+1}^{(0)}$ of the next domain
either from a Kalman filter, or using $\boldsymbol{y}_n$

Iterate until convergence

$$\boldsymbol{y}_{n+1}^{(j+1)} = \frac{\sum_i w_i \boldsymbol{x}_i g(\|\frac{\boldsymbol{x}_i - \boldsymbol{y}^{(j)}}{h}\|^2)}{\sum_i w_i g(\|\frac{\boldsymbol{x}_i - \boldsymbol{y}^{(j)}}{h}\|^2)}$$

where p_u, k, g are as given in the text

The track is the sequence of converged estimates $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_N$.

Tracking Using Flow

could have a family of flow models, as in Section 10.6.1, and find the best matching domain resulting from a flow model. We write the image as a function of space and time as $\mathcal{I}(x, y, t)$, and scale and translate time so that each frame appears at an integer value of t.

the best flow involves minimizing

$$\sum_{\boldsymbol{x} \in \mathcal{D}_n} w(\boldsymbol{x}) \rho(\mathcal{I}(\boldsymbol{x}, n), \mathcal{I}(\boldsymbol{x} + \boldsymbol{v}(\boldsymbol{x}; \theta), n + 1))$$

as a function of the flow parameters θ.

Tracking Using Flow

effect on the results. The usual solution is to adopt an M-estimator. A good choice of ρ is

$$\rho(u, v) = \frac{(u - v)^2}{(u - v)^2 + \sigma^2}$$

where σ is a parameter (there is greater detail on M-estimators in Section 10.4.1).

We now have the best value of θ, given by $\hat{\theta}$. The new domain is given by

$$\mathcal{D}_{n+1} = \left\{ \boldsymbol{u} \mid \boldsymbol{u} = \boldsymbol{x} + \boldsymbol{v}(\boldsymbol{x}; \hat{\theta}), \forall \boldsymbol{x} \in \mathcal{D}_n \right\}.$$

We can build domain models that simplify estimating $\mathcal{D}_{n+1}$; for example, if the

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

Important pragmatic difficulty with flow-based trackers

- detection-based tracker might not properly account for changes in illumination, aspect
- small errors in localization can accumulate in flow based tracker
- Loose clothing is a particularly important problem on the body configuration
- minor geometric phenomena can cause significant changes in image brightness

CAMBRIDGE INSTITUTE OF TECHNOLOGY

Tracking

Contents

1) Simple tracking strategies
2) Tracking using matching
3) Tracking linear dynamical models with Kalman filters
4) Conclusion
5) Q&A

$$\nabla \cdot \mathbf{E} = \rho/\varepsilon_0$$

$$\nabla \cdot \mathbf{B} = 0$$

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}$$

$$\nabla \times \mathbf{B} = \mu_0 \varepsilon_0 \frac{\partial \mathbf{E}}{\partial t} + \mu_0 \boldsymbol{j}_c$$

where

$$\nabla = \hat{\mathbf{i}} \frac{\partial}{\partial x} + \hat{\mathbf{j}} \frac{\partial}{\partial y} + \hat{\mathbf{k}} \frac{\partial}{\partial z}$$

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

Accurate representation of posterior on object state

- state of the object at the ith frame is typically written as Xi.
- measurements obtained in the ith frame are values of a random variable Yi
- In tracking, we wish to determine some representation of P(Xk|Y0, . . . , Yk).
- In smoothing, we wish to determine some representationof P(Xk|Y0, . . . , YN)

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

Xi form a Markov chain

- We assume measurements depend only on the hidden state, that is, that $P(Y_k|X_0,\ldots,X_N,Y_0,\ldots,Y_N) = P(Y_k|X_k)$.
- We assume that the probability density for a new state is a function only of the previous state; that is, $P(X_k|X_0,\ldots,X_{k-1}) = P(X_k|X_{k-1})$ or, equivalently, that X_i form a *Markov chain*.

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

Predictive density

predictive density

$$P(X_k|\boldsymbol{Y}_0 = \boldsymbol{y}_0, \ldots, \boldsymbol{Y}_{k-1} = \boldsymbol{y}_{k-1}))$$

is equal to

$$\int P(X_k|X_{k-1})P(X_{k-1}|Y_0, \ldots, Y_{k-1})dX_{k-1}$$

Data association: Some of the measurements obtained from the ith frame may tell us about the object's state. Typically, we use $P(\boldsymbol{X}_i|\boldsymbol{Y}_0 = \boldsymbol{y}_0, \ldots, \boldsymbol{Y}_{i-1} = \boldsymbol{y}_{i-1})$ to identify these measurements. For example, we might use this predictive

Posterior probability

manipulation of probability combined with the assumptions above yields that the *posterior*

$$P(X_k|\boldsymbol{Y}_0 = \boldsymbol{y}_0, \ldots, \boldsymbol{Y}_k = \boldsymbol{y}_k)$$

is given by

$$\frac{P(Y_k = \boldsymbol{y}_k|X_k)P(X_k|\boldsymbol{Y}_0 = \boldsymbol{y}_0, \ldots, \boldsymbol{Y}_k = \boldsymbol{y}_k)}{\int P(Y_k = \boldsymbol{y}_k|X_k)P(X_k|\boldsymbol{Y}_0 = \boldsymbol{y}_0, \ldots, \boldsymbol{Y}_k = \boldsymbol{y}_k)dX_k}.$$

Linear Measurements and Linear Dynamics

on the frame), and then adding a normal random variable of zero mean and known covariance (which again may depend on the frame). We use the notation

$$\boldsymbol{x} \sim N(\boldsymbol{\mu}, \Sigma)$$

to mean that $\boldsymbol{x}$ is the value of a random variable with a normal probability distribution with mean $\boldsymbol{\mu}$ and covariance Σ. We write $\boldsymbol{x}_k$ for the state at step k. Our model is that $P(Y_k|X_k = \boldsymbol{x}_k)$ is a Gaussian with mean $\mathcal{B}_k\boldsymbol{x}_k$ and covariance Σ. Using the notation above, we can write our model of measurements as

$$\boldsymbol{y}_k \sim N(\mathcal{B}_k\boldsymbol{x}_k, \Sigma_k).$$

This model may seem limited, but is very powerful (it is the cornerstone of a huge control industry). We do not need to observe the whole of the state vector at any given time to infer it. For example, if we have enough measurements of the

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

Simplest possible dynamical model

on the frame) and then adding a normal random variable of zero mean and known covariance. We can write our dynamical model as

$$\boldsymbol{x}_i \sim N(\mathcal{D}_i \boldsymbol{x}_{i-1}; \Sigma_{d_i});$$

$$\boldsymbol{y}_i \sim N(\mathcal{M}_i \boldsymbol{x}_i; \Sigma_{m_i}).$$

Notice that the covariances could be different from frame to frame, as could the matrices. Although this model appears limited, it is in fact extremely powerful; we

Constant Velocity model

Constant Velocity Assume that the vector $\boldsymbol{p}$ gives the position and $\boldsymbol{v}$ the velocity of a point moving with constant velocity. In this case, $\boldsymbol{p}_i = \boldsymbol{p}_{i-1} + (\Delta t)\boldsymbol{v}_{i-1}$ and $\boldsymbol{v}_i = \boldsymbol{v}_{i-1}$. This means that we can stack the position and velocity into a single state vector, and our model applies (Figure 11.7). In particular,

$$\boldsymbol{x} = \begin{Bmatrix} \boldsymbol{p} \\ \boldsymbol{v} \end{Bmatrix}$$

and

$$\mathcal{D}_i = \begin{Bmatrix} Id & (\Delta t)Id \\ 0 & Id \end{Bmatrix}.$$

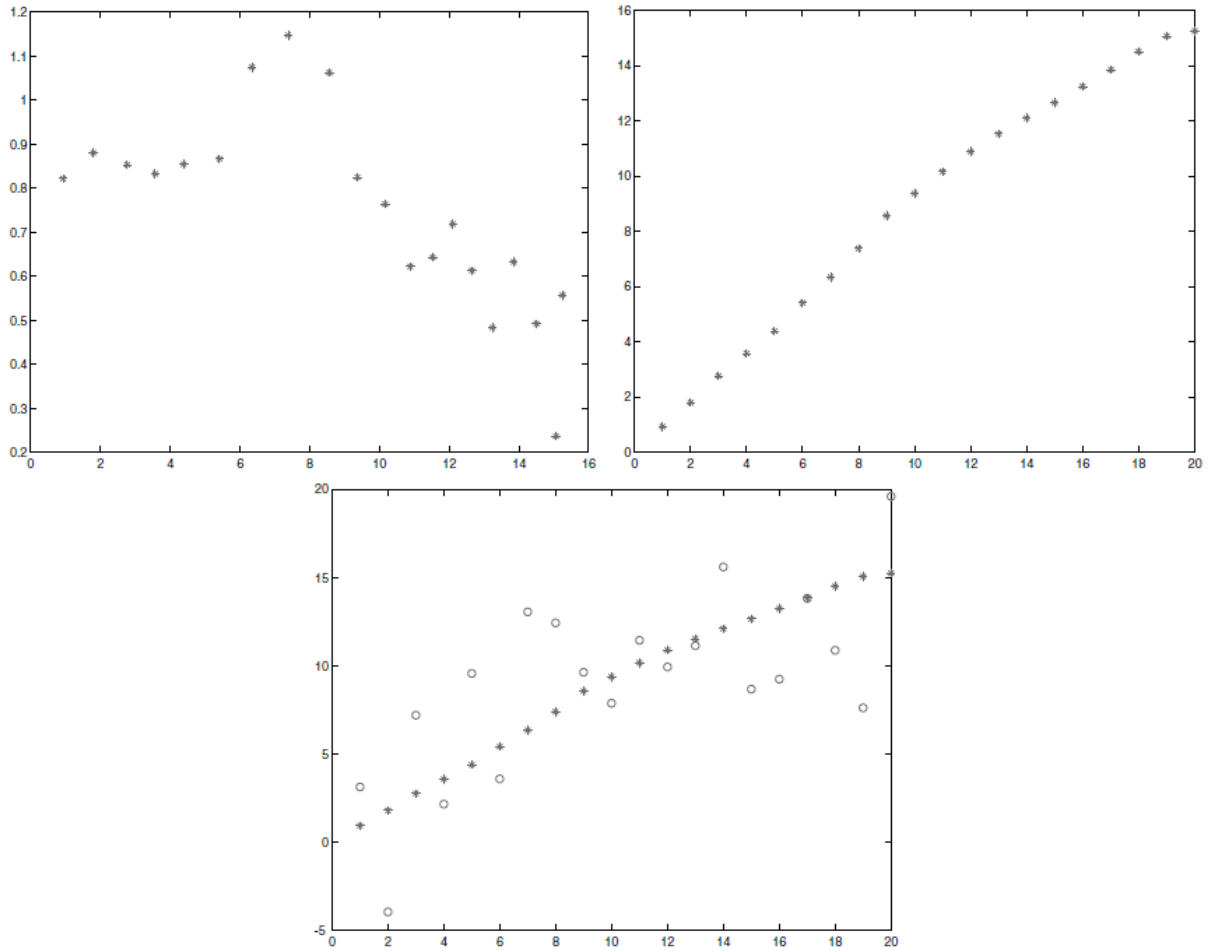
Notice that, again, we don't have to observe the whole state vector to make a useful measurement. For example, in many cases, we would expect that

$$\mathcal{M}_i = \begin{Bmatrix} Id & 0 \end{Bmatrix}$$

(i.e., that we see only the position of the point). Because we know that it's moving with constant velocity—that's the model—we expect that we could use these

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

A constant velocity dynamic model for a point on the line.



CAMBRIDGE INSTITUTE OF TECHNOLOGY

Constant Acceleration

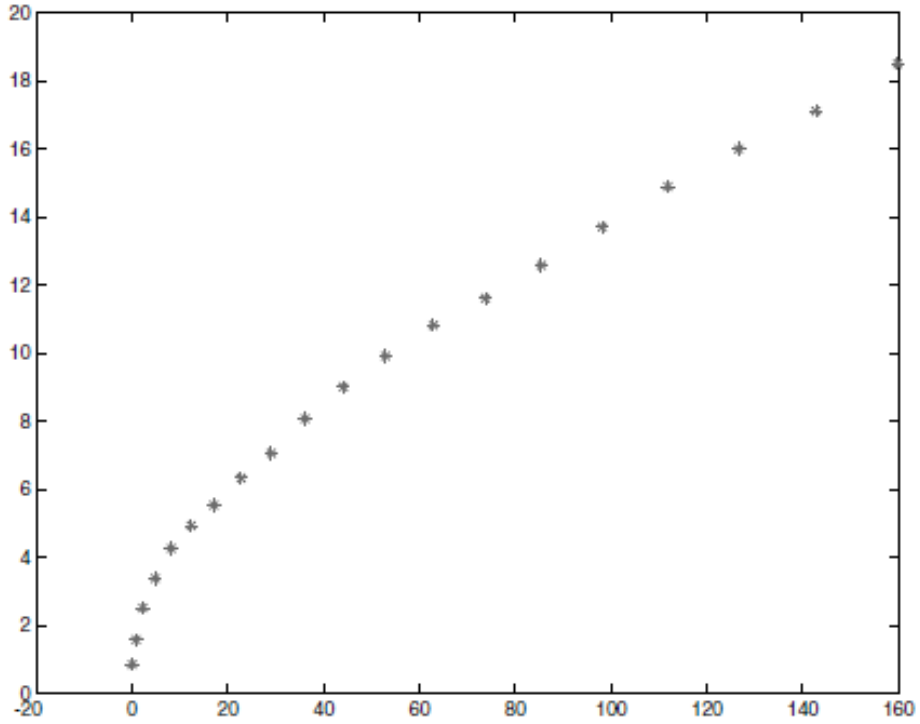
acceleration. In this case, $\boldsymbol{p}_i = \boldsymbol{p}_{i-1} + (\Delta t)\boldsymbol{v}_{i-1}$, $\boldsymbol{v}_i = \boldsymbol{v}_{i-1} + (\Delta t)\boldsymbol{a}_{i-1}$, and $\boldsymbol{a}_i = \boldsymbol{a}_{i-1}$. Again, we can stack the position, velocity, and acceleration into a single state vector, and our model applies (Figure 11.8). In particular,

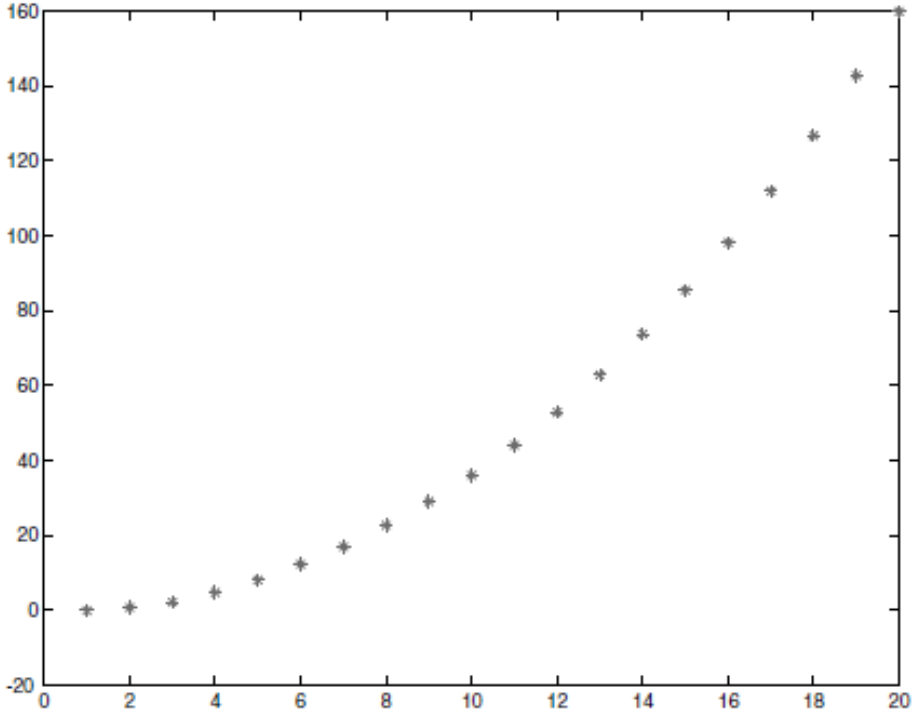
$$\boldsymbol{x} = \begin{Bmatrix} \boldsymbol{p} \\ \boldsymbol{v} \\ \boldsymbol{a} \end{Bmatrix}$$

$$\mathcal{D}_i = \begin{Bmatrix} Id & (\Delta t)Id & 0 \\ 0 & Id & (\Delta t)Id \\ 0 & 0 & Id \end{Bmatrix}. \qquad \mathcal{M}_i = \begin{Bmatrix} Id & 0 & 0 \end{Bmatrix}$$

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

a constant acceleration model for a point moving on the line





Periodic Motion

movement. Typically, its position p satisfies a differential equation such as

$$\frac{d^2p}{dt^2} = -p.$$

velocity as v and stacking position and velocity into a vector $\boldsymbol{u} = (p, v)$; we then have

$$\frac{d\boldsymbol{u}}{dt} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \boldsymbol{u} = \mathcal{S}\boldsymbol{u}.$$

$$\begin{aligned} \boldsymbol{u}_i &= \boldsymbol{u}_{i-1} + \Delta t \frac{d\boldsymbol{u}}{dt} \\ &= \boldsymbol{u}_{i-1} + \Delta t \mathcal{S}\boldsymbol{u}_{i-1} \\ &= \begin{pmatrix} 1 & \Delta t \\ -\Delta t & 1 \end{pmatrix} \boldsymbol{u}_{i-1}. \end{aligned}$$

Algorithm 11.3: The Kalman Filter.

Dynamic Model:

$$\begin{aligned} \boldsymbol{x}_i &\sim N(\mathcal{D}_i \boldsymbol{x}_{i-1}, \Sigma_{d_i}) \\ \boldsymbol{y}_i &\sim N(\mathcal{M}_i \boldsymbol{x}_i, \Sigma_{m_i}) \end{aligned}$$

Start Assumptions: $\overline{\boldsymbol{x}}_0^-$ and Σ_0^- are known
Update Equations: Prediction

$$\begin{aligned} \overline{\boldsymbol{x}}_i^- &= \mathcal{D}_i \overline{\boldsymbol{x}}_{i-1}^+ \\ \Sigma_i^- &= \Sigma_{d_i} + \mathcal{D}_i \sigma_{i-1}^+ \mathcal{D}_i \end{aligned}$$

Update Equations: Correction

$$\begin{aligned} \mathcal{K}_i &= \Sigma_i^- \mathcal{M}_i^T \left[\mathcal{M}_i \Sigma_i^- \mathcal{M}_i^T + \Sigma_{m_i}\right]^{-1} \\ \overline{\boldsymbol{x}}_i^+ &= \overline{\boldsymbol{x}}_i^- + \mathcal{K}_i \left[\boldsymbol{y}_i - \mathcal{M}_i \overline{\boldsymbol{x}}_i^-\right] \\ \Sigma_i^+ &= \left[Id - \mathcal{K}_i \mathcal{M}_i\right] \Sigma_i^- \end{aligned}$$

Algorithm 11.4: Forward-Backward Smoothing.

Forward filter: Obtain the mean and variance of $P(\boldsymbol{X}_i|\boldsymbol{y}_0,\ldots,\boldsymbol{y}_i)$ using the Kalman filter. These are $\overline{\boldsymbol{X}}_i^{f,+}$ and $\Sigma_i^{f,+}$.

Backward filter: Obtain the mean and variance of $P(\boldsymbol{X}_i|\boldsymbol{y}_{i+1},\ldots,\boldsymbol{y}_N)$ using the Kalman filter running backward in time. These are $\overline{\boldsymbol{X}}_i^{b,-}$ and $\Sigma_i^{b,-}$.

Combining forward and backward estimates: Regard the backward estimate as a new measurement for $\boldsymbol{X}_i$, and insert into the Kalman filter equations to obtain

$$\Sigma_i^* = \left[(\Sigma_i^{f,+})^{-1} + (\Sigma_i^{b,-})^{-1}\right]^{-1};$$

$$\overline{\boldsymbol{X}}_i^* = \Sigma_i^* \left[(\Sigma_i^{f,+})^{-1}\overline{\boldsymbol{X}}_i^{f,+} + (\Sigma_i^{b,-})^{-1}\overline{\boldsymbol{X}}_i^{b,-}\right].$$

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

The Kalman Filter.

Predict:

$$\hat{\mathbf{x}}_{t|t-1} = \mathbf{F}_t \hat{\mathbf{x}}_{t-1|t-1} + \mathbf{B}_t \mathbf{u}_t$$

$$\mathbf{P}_{t|t-1} = \mathbf{F}_t \mathbf{P}_{t-1|t-1} \mathbf{F}_t^T + \mathbf{Q}_t$$

Update:

$$\hat{\mathbf{x}}_{t|t} = \hat{\mathbf{x}}_{t|t-1} + \mathbf{K}_t \left(\mathbf{y}_t - \mathbf{H}_t \hat{\mathbf{x}}_{t|t-1} \right)$$

$$\mathbf{K}_t = \mathbf{P}_{t|t-1} \mathbf{H}_t^T \left(\mathbf{H}_t \mathbf{P}_{t|t-1} \mathbf{H}_t^T + \mathbf{R}_t \right)^{-1}$$

$$\mathbf{P}_{t|t} = (\mathbf{I} - \mathbf{K}_t \mathbf{H}_t) \mathbf{P}_{t|t-1}$$

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

The Kalman Filter.

where

$\hat{\mathbf{x}}$: Estimated state.

$\mathbf{F}$: State transition matrix (i.e., transition between states).

$\mathbf{u}$: Control variables.

$\mathbf{B}$: Control matrix (i.e., mapping control to state variables).

$\mathbf{P}$: State variance matrix (i.e., error of estimation).

$\mathbf{Q}$: Process variance matrix (i.e., error due to process).

$\mathbf{y}$: Measurement variables.

$\mathbf{H}$: Measurement matrix (i.e., mapping measurements onto state).

$\mathbf{K}$: Kalman gain.

$\mathbf{R}$: Measurement variance matrix (i.e., error from measurements).

Subscripts are as follows: $t|t$ current time period, $t-1|t-1$ previous time period, and $t|t-1$ are intermediate steps.

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

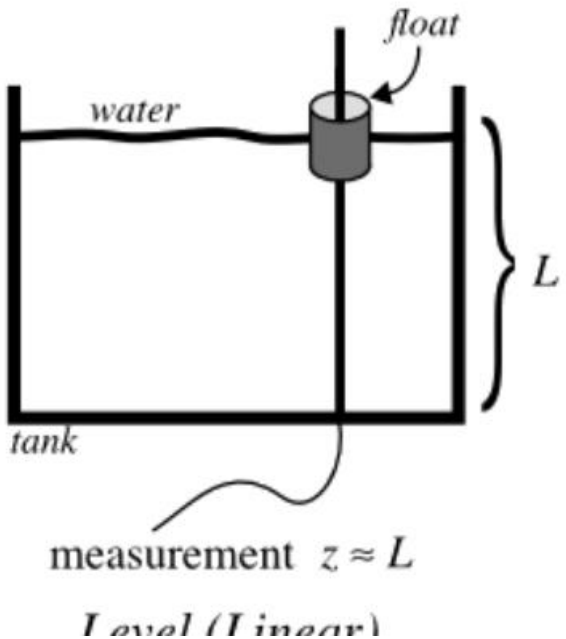
The Kalman Filter.

1. **Understand the situation:** Look at the problem. Break it down to the mathematical basics. If you don't do this, you may end up doing unneeded work.
2. **Model the state process:** Start with a basic model. It may not work effectively at first, but this can be refined later.
3. **Model the measurement process:** Analyze how you are going to measure the process. The measurement space may not be in the same space as the state (e.g., using an electrical diode to measure weight, an electrical reading does not easily translate to a weight).
4. **Model the noise:** This needs to be done for both the **state** and **measurement** process. The base Kalman filter assumes Gaussian (white) noise, so make the variance and covariance (error) meaningful (i.e., make sure that the error you model is suitable for the situation).
5. **Test the filter:** Often overlooked, use synthetic data if necessary (e.g., if the process is not safe to test on a live environment). See if the filter is behaving as it should.
6. **Refine filter:** Try to change the noise parameters (filter), as this is the easiest to change. If necessary go back further, you may need to rethink the situation.

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

The Kalman Filter - Understanding the situation

We consider a simple situation showing a way to measure the level of water in a tank. This is shown in the figure[a].



CAMBRIDGE
INSTITUTE OF TECHNOLOGY

The Kalman Filter - Model the state process

The first is the most basic model, the tank is level (i.e., the true level is constant $L = c$).

Using the equations from Page 2, the state variable can be reduced to a scalar (i.e., $\hat{\mathbf{x}} = x$ where x is the estimate of L).

We are assuming a constant model, therefore $x_{t+1} = x_t$, so $\mathbf{A} = 0$ and $\mathbf{F}_t = 1$, for any $t \geq 0$.

Control variables $\mathbf{B}$ and $\mathbf{u}$ are not used (i.e., both $= 0$).

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

The Kalman Filter -Model the measurement process

In our model, we have the level of the float. This is represented by $\mathbf{y} = y$.

The value we are measuring could be a scaled measurement (e.g., a 1 cm measurement on a mechanical dial could actually be about 10 cm in the "true" level of the tank). Think of your petrol gauge on your car, 1 cm can represent 10 L of petrol!

For simplicity, we will assume that the measurement is the exact same scale as our state estimate x (i.e., $\mathbf{H} = 1$).

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

The Kalman Filter - Model the noise

For this model, we are going to assume that there is noise from the measurement (i.e., $\mathbf{R} = r$).

The process is a scalar, therefore $\mathbf{P} = p$. And as the process is not well defined, we will adjust the noise (i.e., $\mathbf{Q} = q$).

We will now demonstrate the effects of changing these noise parameters.

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

The Kalman Filter - Test the filter

Predict:

$$x_{t|t-1} = x_{t-1|t-1}$$
$$p_{t|t-1} = p_{t-1|t-1} + q_t$$

Update:

$$x_{t|t} = x_{t|t-1} + K_t \left(y_t - x_{t|t-1}\right)$$
$$K_t = p_{t|t-1} \left(p_{t|t-1} + r\right)^{-1}$$
$$p_{t|t} = \left(1 - K_t\right) p_{t|t-1}$$

The Kalman Filter - Test the filter

The filter is now completely defined. Let's put some numbers into this model. For the first test, we assume the true level of the tank is $L = 1$.

We initialize the state with an arbitrary number, with an extremely high variance as it is completely unknown: $x_0 = 0$ and $p_0 = 1000$. If you initialize with a more meaningful variable, you will get faster convergence. The system noise we will choose will be $q = 0{\cdot}0001$, as we think we have an accurate model. Let's start this process.

Predict:

$$x_{1|0} = 0$$

$$p_{1|0} = 1000 + 0{\cdot}0001$$

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

The Kalman Filter - Test the filter

The hypothetical measurement we get is $y_1 = 0{\cdot}9$ (due to noise).
We assume a measurement noise of $r = 0{\cdot}1$.

Update:

$$K_1 = 1000{\cdot}0001(1000{\cdot}0001 + 0{\cdot}1)^{-1} = 0{\cdot}9999$$
$$x_{1|1} = 0 + 0{\cdot}9999\,(0{\cdot}9 - 0) = 0{\cdot}8999$$
$$p_{1|1} = (1 - 0{\cdot}9999)\,1000{\cdot}0001 = 0{\cdot}1000$$

So you can see that Step 1, the initialization of 0, has been brought close to the true value of the system. Also, the variance (error) has been brought down to a reasonable value.

The Kalman Filter - do one more step

Predict:

$$x_{t|t-1} = x_{t-1|t-1}$$
$$p_{t|t-1} = p_{t-1|t-1} + q_t$$

Update:

$$x_{t|t} = x_{t|t-1} + K_t \left(y_t - x_{t|t-1}\right)$$
$$K_t = p_{t|t-1} \left(p_{t|t-1} + r\right)^{-1}$$
$$p_{t|t} = \left(1 - K_t\right) p_{t|t-1}$$

Predict:

$$x_{2|1} = 0{\cdot}8999$$
$$p_{2|1} = 0{\cdot}1000 + 0{\cdot}0001 = 0{\cdot}1001$$

The hypothetical measurement we get this time is $y_2 = 0{\cdot}8$ (due to noise).

Update:

$$K_2 = 0{\cdot}1001 \left(0{\cdot}1001 + 0{\cdot}1\right)^{-1} = 0{\cdot}5002$$
$$x_{2|2} = 0{\cdot}8999 + 0{\cdot}5002 \left(0{\cdot}8 - 0{\cdot}8999\right) = 0{\cdot}8499$$
$$p_{2|2} = \left(1 - 0{\cdot}5002\right) 0{\cdot}1001 = 0{\cdot}0500$$

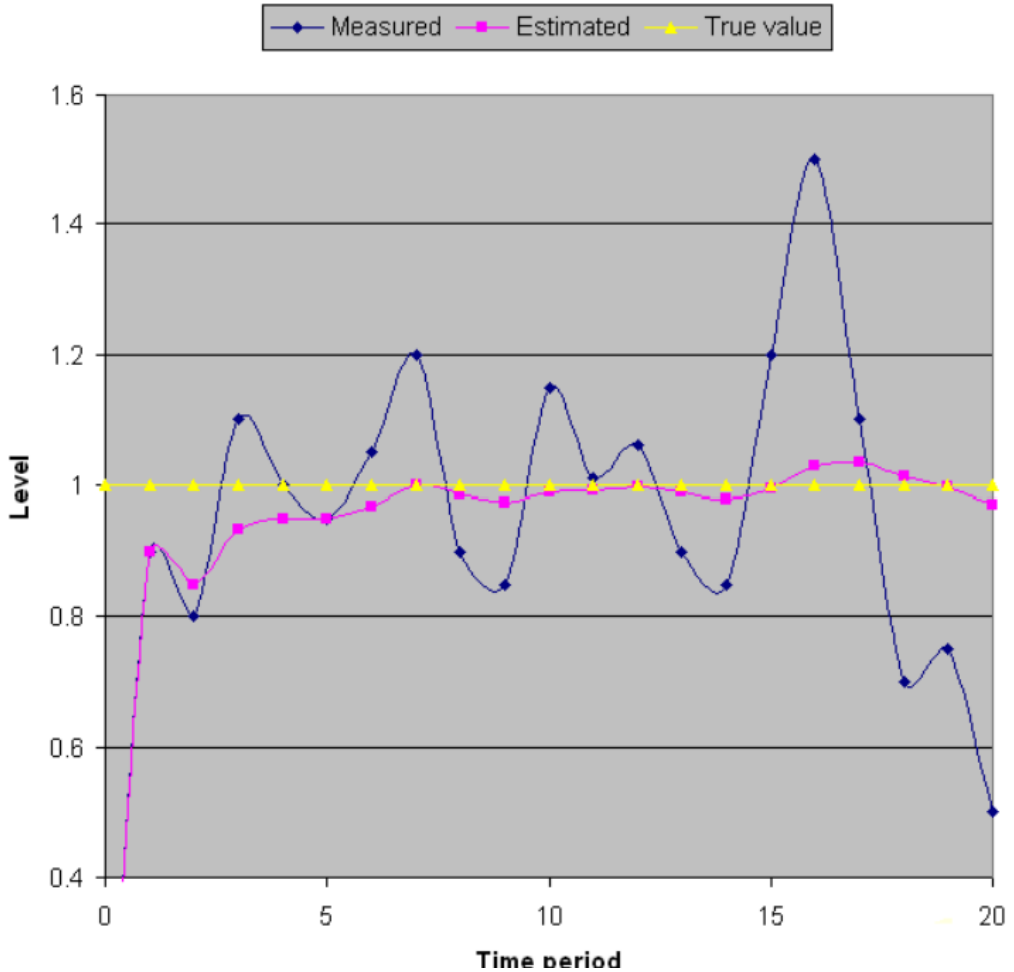
CAMBRIDGE
INSTITUTE OF TECHNOLOGY

The Kalman Filter - Test the filter

	Predict		Update			
t	$x_{t\|t-1}$	$p_{t\|t-1}$	y_t	K_t	$x_{t\|t}$	$p_{t\|t}$
3	0·8499	0·0501	1·1	0·3339	0·9334	0·0334
4	0·9334	0·0335	1	0·2509	0·9501	0·0251
5	0·9501	0·0252	0·95	0·2012	0·9501	0·0201
6	0·9501	0·0202	1·05	0·1682	0·9669	0·0168
7	0·9669	0·0169	1·2	0·1447	1·0006	0·0145
8	1·0006	0·0146	0·9	0·1272	0·9878	0·0127
9	0·9878	0·0128	0·85	0·1136	0·9722	0·0114
10	0·9722	0·0115	1·15	0·1028	0·9905	0·0103

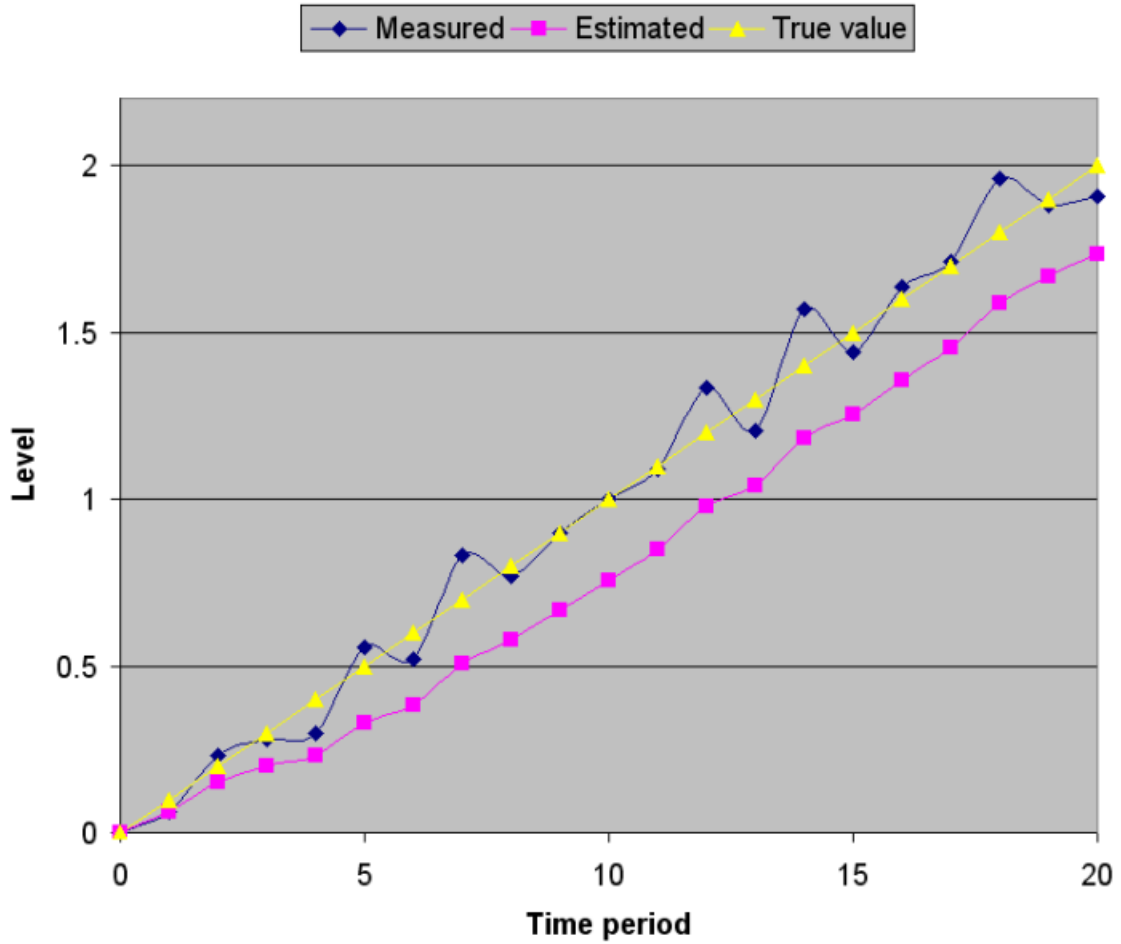
CAMBRIDGE
INSTITUTE OF TECHNOLOGY

The Kalman Filter - Test the filter with result as graph



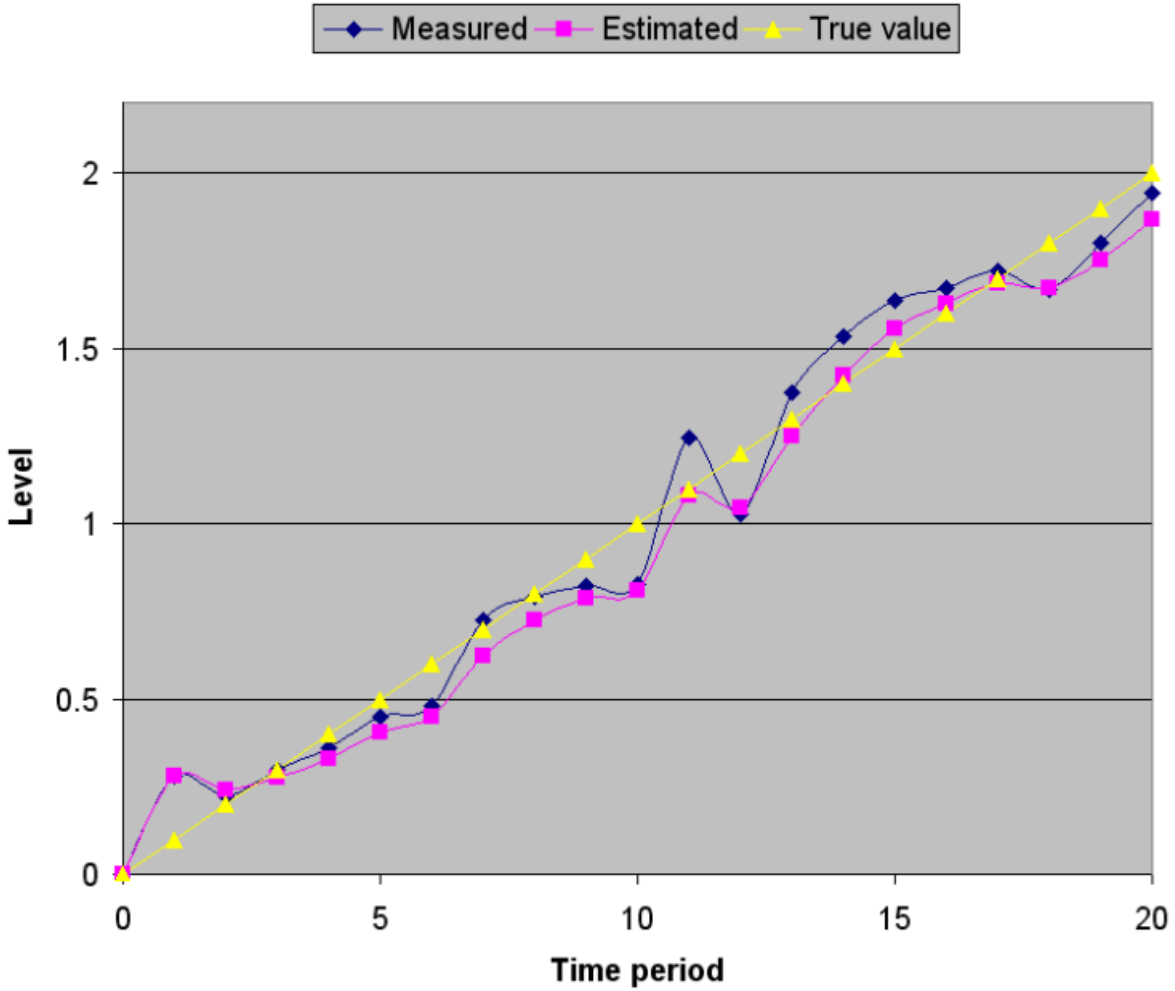
CAMBRIDGE
INSTITUTE OF TECHNOLOGY

The Kalman Filter - Test the filter with result as graph with q = 0.01



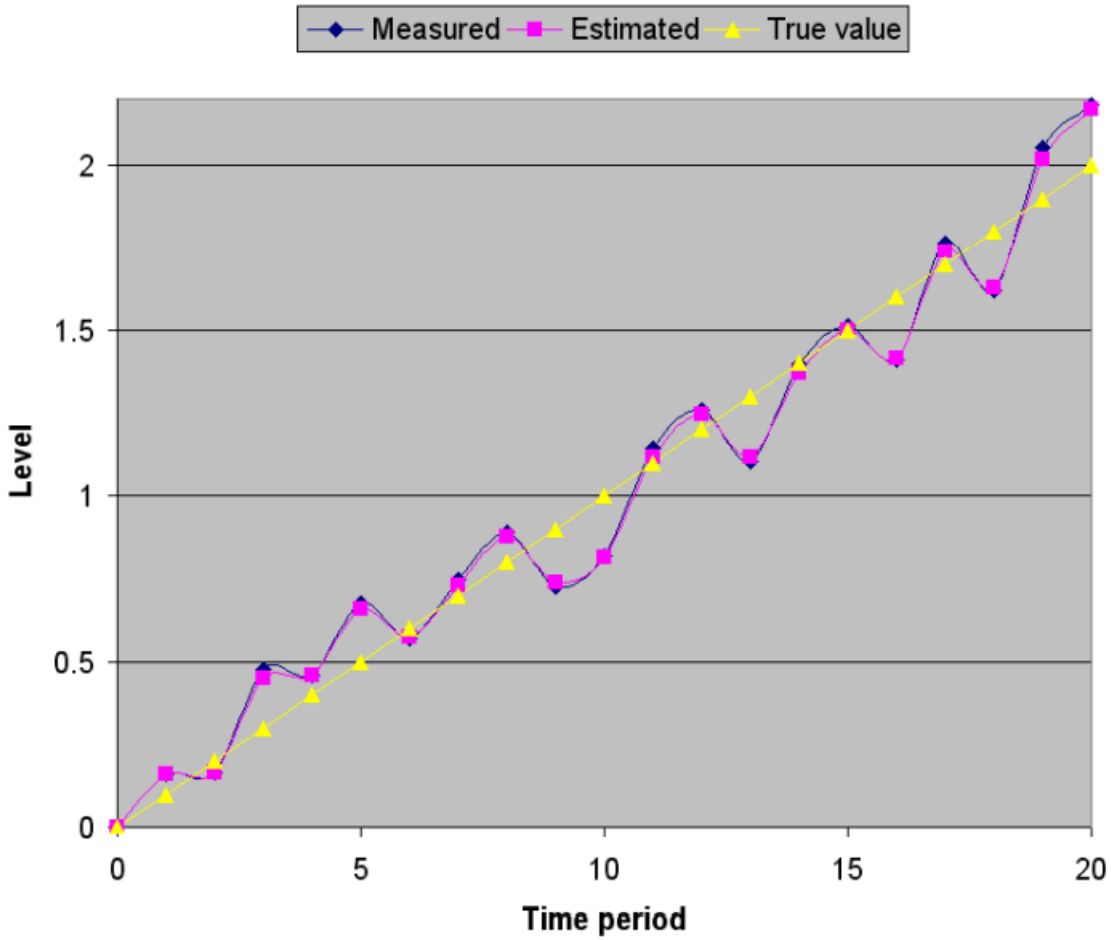
CAMBRIDGE
INSTITUTE OF TECHNOLOGY

The Kalman Filter - Test the filter with result as graph with q = 0.1



CAMBRIDGE
INSTITUTE OF TECHNOLOGY

The Kalman Filter - Test the filter with result as graph with q = 1



CAMBRIDGE INSTITUTE OF TECHNOLOGY

The Kalman Filter – Another example – the acceleration of the object will be equal to the acceleration due to gravity.

acceleration due to gravity. Defining the height of the object in meters, h, we have:

$$\ddot{h}(t) = -g \tag{18}$$

where g is the acceleration due to gravity ($g = 9.80665$ m/s^2). Integrating this relationship over a small time interval, Δt, gives

$$\ddot{h}(t) = \frac{\dot{h}(t) - \dot{h}(t - \Delta t)}{\Delta t} = -g \tag{19}$$

which is a backward difference equation, which is useful for Kalman filtering applications due to the recursive structure of the filter, i.e. each time step in the Kalman filter always references the previous time step. Simplifying this expression gives

$$\dot{h}(t) = \dot{h}(t - \Delta t) - g\Delta t \tag{20}$$

The Kalman Filter

Integrating again yields a commonly used kinematic equation relating successive positions of a particle with respect to time for constant acceleration

$$h(t) = h(t-\Delta t) + \dot{h}(t-\Delta t) - \frac{1}{2} g (\Delta t)^2 \tag{21}$$

Rather than consider these equations in terms of continuous time, t, it is beneficial to rewrite the equations in terms of a discrete time index, k, which is defined by $t = k\Delta t$. Additionally, discrete time values are traditionally written as subscripts rather than as a functional dependence, i.e.

$$\begin{aligned} h(t) &= h(k\Delta t) = h_k \\ h(t-\Delta t) &= h(k\Delta t - \Delta t) = h\left(\Delta t\left(k-1\right)\right) = h_{k-1} \end{aligned} \tag{22}$$

Rewriting the kinematic relationships for the example problem in terms of the discrete time variable, k, gives

$$\begin{aligned} \dot{h}_k &= \dot{h}_{k-1} - g\Delta t \\ h_k &= h_{k-1} + \dot{h}_{k-1}\Delta t - \frac{1}{2} g (\Delta t)^2 \end{aligned} \tag{23}$$

The Kalman Filter

velocity as a state. As a result, we now define the state vector for the Kalman filter as

$$\mathbf{x}_k = \begin{bmatrix} h_k \\ \dot{h}_k \end{bmatrix} \tag{24}$$

which results in the following expression

$$\mathbf{x}_k = \begin{bmatrix} h_{k-1} + \dot{h}_{k-1}\Delta t - \frac{1}{2} g (\Delta t)^2 \\ \dot{h}_{k-1} - g\Delta t \end{bmatrix} \tag{25}$$

With this definition, we can rewrite these equations in terms of the state vector, $\mathbf{x}$, as in

$$\mathbf{x}_k = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \mathbf{x}_{k-1} + \begin{bmatrix} -\frac{1}{2}(\Delta t)^2 \\ -\Delta t \end{bmatrix} g \tag{26}$$

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

The Kalman Filter

With this definition, we can rewrite these equations in terms of the state vector, $\mathbf{x}$, as in

$$\mathbf{x}_k = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \mathbf{x}_{k-1} + \begin{bmatrix} -\frac{1}{2}(\Delta t)^2 \\ -\Delta t \end{bmatrix} g \tag{26}$$

Now, we have the problem in necessary format for the Kalman filter

$$\mathbf{x}_k = \mathbf{F}_{k-1}\mathbf{x}_{k-1} + \mathbf{G}_{k-1}\mathbf{u}_{k-1} \tag{27}$$

where the system matrices $\mathbf{F}$ and $\mathbf{G}$ are given by

$$\mathbf{F}_{k-1} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}, \quad \mathbf{G}_{k-1} = \begin{bmatrix} -\frac{1}{2}(\Delta t)^2 \\ -\Delta t \end{bmatrix} \tag{28}$$

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

The Kalman Filter

and the input vector can be defined as

$$\mathbf{u}_{k-1} = g$$

the process noise covariance matrix, $\mathbf{Q}$, can be set to zero.

the equations at each time step. Note that for this particular problem the values of $\mathbf{F}$, $\mathbf{G}$, and $\mathbf{u}$ do not vary with respect to k. It is important to note in these equations that there is no process noise uncertainty term, $\mathbf{w}$. For this particular set of equations, we are assuming that there are no errors

The Kalman Filter

gives the position as a function of the states of the filter. There is some uncertainty in the measurement, which is noted in the equations by the measurement noise vector, **v**:

$$\mathbf{y}_k = h_k + \mathbf{v}_k \tag{30}$$

Since the position can be written in terms of the state vector, this can be rewritten as

$$\mathbf{y}_k = \begin{bmatrix} 1 & 0 \end{bmatrix} \mathbf{x}_k + \mathbf{v}_k \tag{31}$$

Now, we have the output equations of the system defined in the proper form as in (2), where the system matrix, **H**, is given by

$$\mathbf{H}_k = \begin{bmatrix} 1 & 0 \end{bmatrix} \tag{32}$$

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

The Kalman Filter

The considered measurement system has a standard deviation of error of 2 m, which is a variance of 4 m^2. Because of this, and the fact that there is only one term in the output vector, the resulting measurement noise covariance matrix reduces to a scalar value, $\mathbf{R} = 4\ m^2$.

In addition to the measurement noise, we also need to consider any uncertainty in the initial state assumption. The initial position is approximately known to be 105 m before the ball is dropped, while the actual initial position is 100 m. The initial guess was roughly determined, and should therefore have a relatively high corresponding component in the assumed initial covariance. For this example, we consider an error of 10 m^2 for the initial position. For the initial velocity, we assume that the object starts from rest. This assumption is fairly reasonable in this case, so a smaller uncertainty value of 0.01 m^2/s^2 is assumed. To aid the reader in the implementation of this example, the necessary parameters and definitions for this filtering application are summarized in Table 4.

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

The Kalman Filter

Term	Definition
State Vector	$\mathbf{x}_k = \begin{bmatrix} h_k \\ \dot{h}_k \end{bmatrix}$
Output Vector	$\mathbf{y}_k = h_k + \mathbf{v}_k$
Input Vector	$\mathbf{u}_{k-1} = g$
System State Matrix	$\mathbf{F}_{k-1} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}$
Input Matrix	$\mathbf{G}_{k-1} = \begin{bmatrix} -\frac{1}{2}(\Delta t)^2 \\ -\Delta t \end{bmatrix}$
Observation Matrix	$\mathbf{H}_k = \begin{bmatrix} 1 & 0 \end{bmatrix}$
Process Noise Covariance Matrix	$\mathbf{Q}_{k-1} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$
Measurement Noise Covariance Matrix	$\mathbf{R}_k = 4$
True Initial State Vector	$\mathbf{x}_0 = \begin{bmatrix} 100 \\ 0 \end{bmatrix}$
Assumed Initial State Vector	$\hat{\mathbf{x}}_0 = \begin{bmatrix} 105 \\ 0 \end{bmatrix}$
Assumed Initial State Error Covariance Matrix	$\mathbf{P}_0 = \begin{bmatrix} 10 & 0 \\ 0 & 0.01 \end{bmatrix}$
Time Increment	$\Delta t = 0.001$

The Kalman Filter

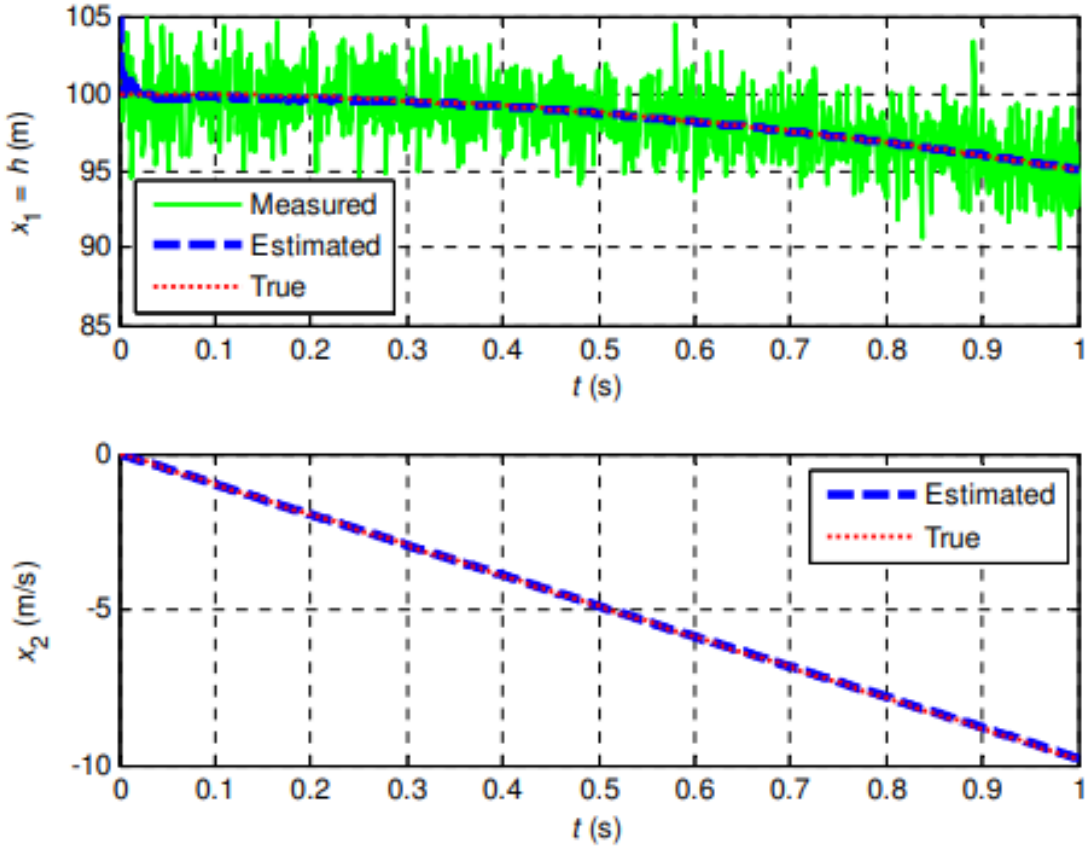


Figure 3. Kalman Filtering Example Estimated and True States

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

The Kalman Filter

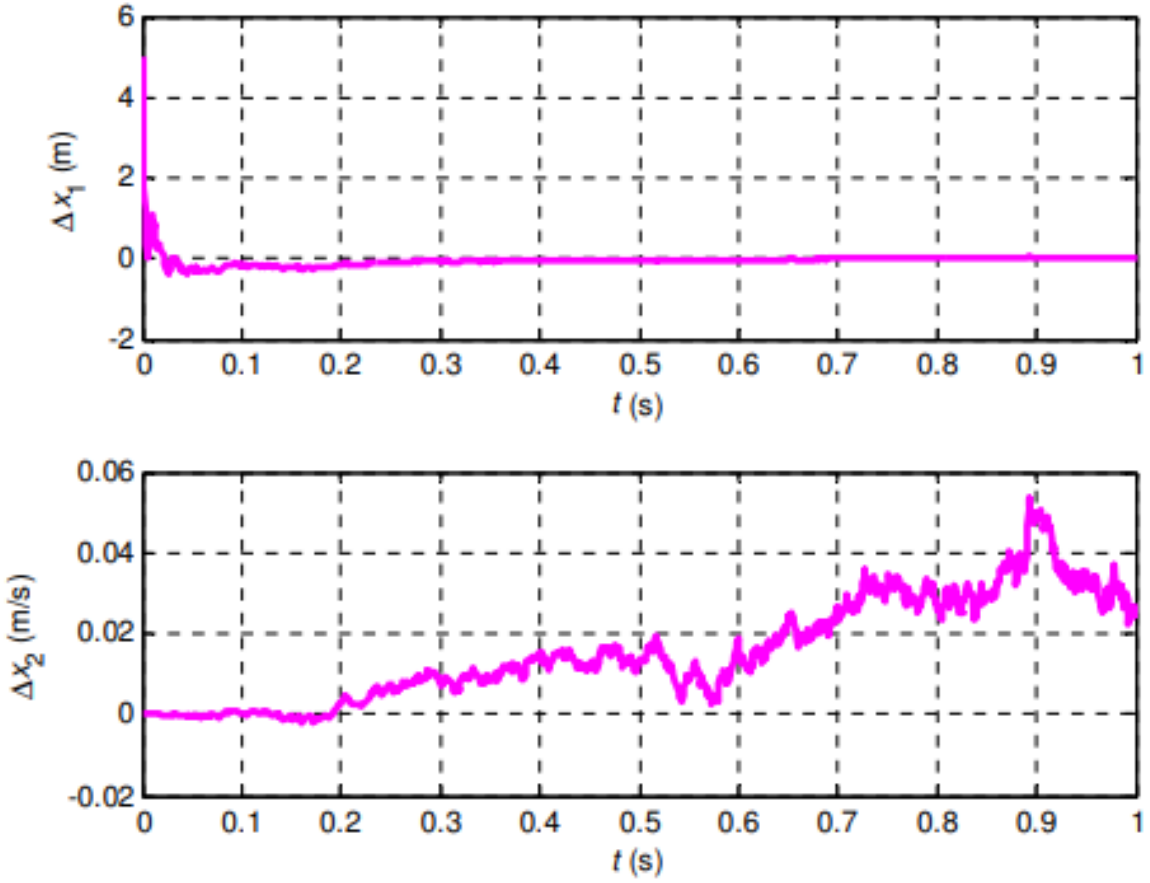


Figure 4. Kalman Filtering Example Estimation Errors

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

The Kalman Filter – key applications

The applications of a Kalman filter are numerous:

- Tracking objects (e.g., missiles, faces, heads, hands)
- Fitting Bezier patches to (noisy, moving, ...) point data
- Economics
- Navigation
- Many computer vision applications
  - Stabilizing depth measurements
  - Feature tracking
  - Cluster tracking
  - Fusing data from radar, laser scanner and stereo-cameras for depth and velocity measurements
  - Many more

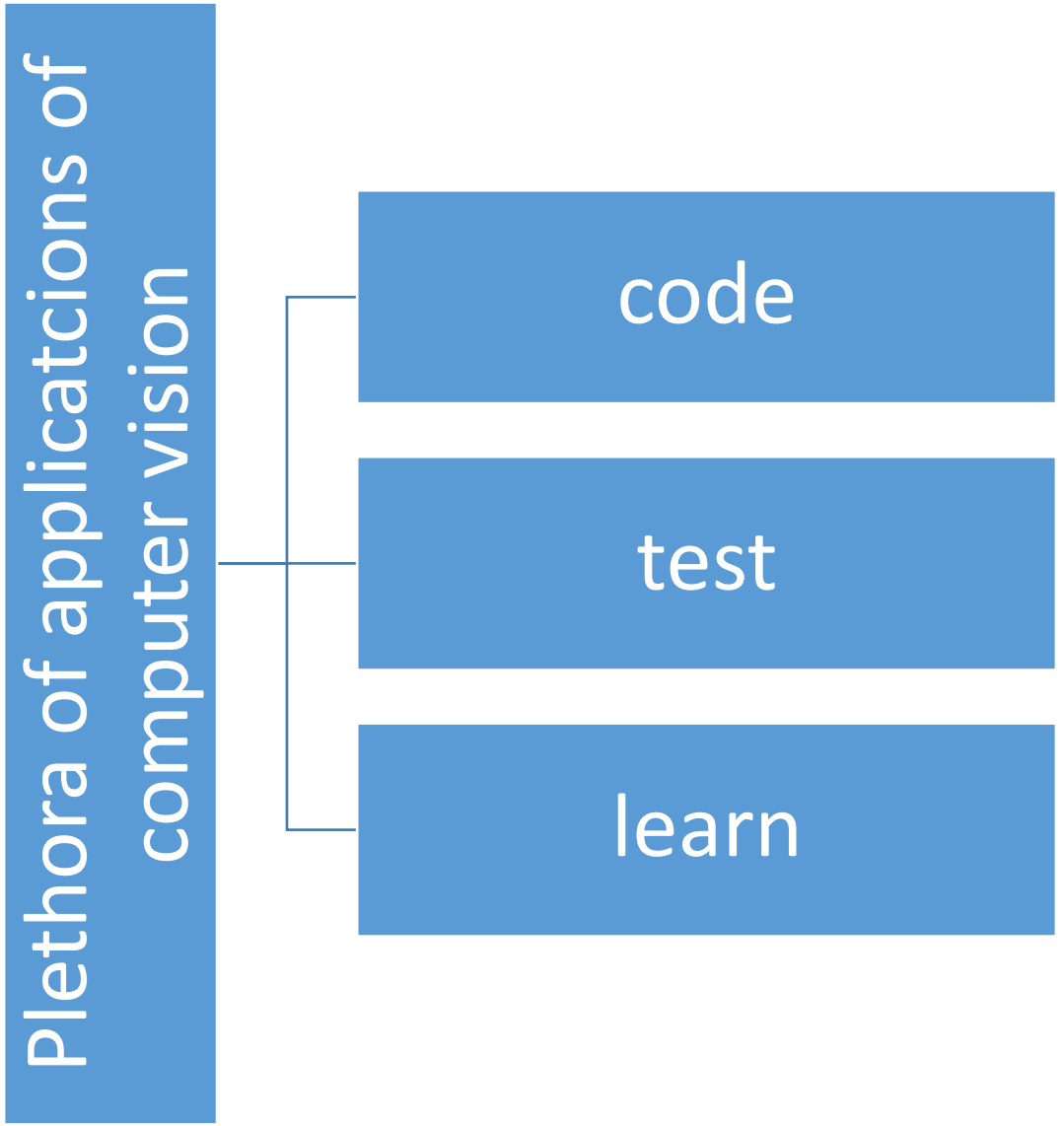
CAMBRIDGE
INSTITUTE OF TECHNOLOGY

The Kalman Filter - advantages

- Best with gaussian noise
- Simple computations less error margins
- Key is state vector representation

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

Conclusion



CAMBRIDGE
INSTITUTE OF TECHNOLOGY

Q&A

Contact

- **Prof. D. Antony Louis Piriyakumar**
  Dean (Research and development)
  Cambridge Institute of Technology
- K.R. Puram,
  560036 Bengaluru, India
- Mobile: +91 98459 25132
- E-mail:
  dean_rd@Cambridge.edu.in