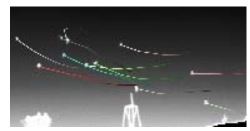
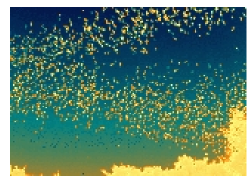
# Tracking

Tuesday, Nov 24



Kristen Grauman
UT-Austin



## Announcements

- Pset 5 out tonight, due 12/4
  – Shorter assignment
  – Auto extension til 12/8

- I will not hold office hours tomorrow 5-6 pm due to Thanksgiving

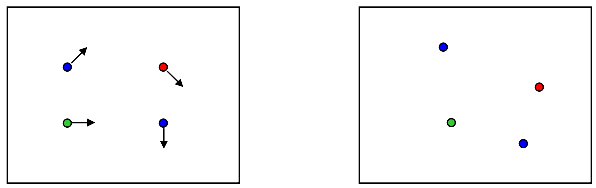
# Outline

- Last time: Motion
  - – Motion field and parallax
  - – Optical flow, brightness constancy
  - – Aperture problem
- Today: Tracking
  - – Tracking as inference
  - – Linear models of dynamics
  - – Kalman filters
  - – General challenges in tracking

## Motion estimation techniques

- Direct methods
  - Directly recover image motion at each pixel from spatio-temporal image brightness variations
  - Dense motion fields, but sensitive to appearance variations
  - Suitable for video and when image motion is small

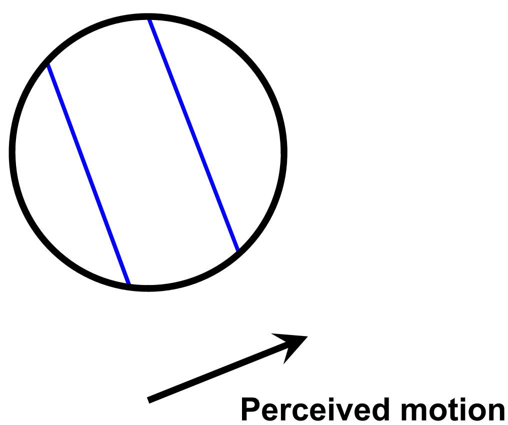
## Direct methods: Estimating optical flow



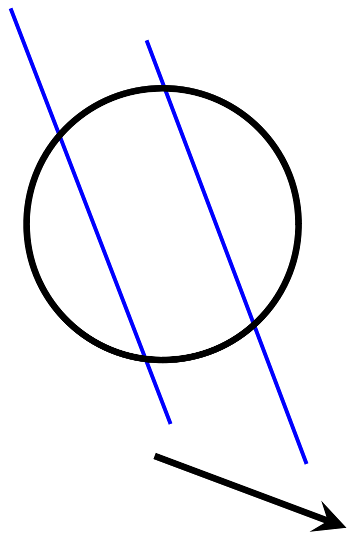
$I(x,y,t-1)$    $I(x,y,t)$

- Given two subsequent frames, estimate the apparent motion field between them.
- Key assumptions
  - **Brightness constancy:** projection of the same point looks the same in every frame
  - **Small motion:** points do not move very far
  - **Spatial coherence:** points move like their neighbors

## The aperture problem



## The aperture problem




## Solving the aperture problem (grayscale image)

- How to get more equations for a pixel?
- **Spatial coherence constraint:** pretend the pixel's neighbors have the same (u,v)
  - If we use a 5x5 window, that gives us 25 equations per pixel

$$0 = I_t(\mathbf{p_i}) + \nabla I(\mathbf{p_i}) \cdot [u \;\; v]$$

$$\begin{bmatrix} I_x(\mathbf{p_1}) & I_y(\mathbf{p_1}) \\ I_x(\mathbf{p_2}) & I_y(\mathbf{p_2}) \\ \vdots & \vdots \\ I_x(\mathbf{p_{25}}) & I_y(\mathbf{p_{25}}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(\mathbf{p_1}) \\ I_t(\mathbf{p_2}) \\ \vdots \\ I_t(\mathbf{p_{25}}) \end{bmatrix}$$

$$\underset{25\text{x}2}{A} \;\; \underset{2\text{x}1}{d} = \underset{25\text{x}1}{b}$$

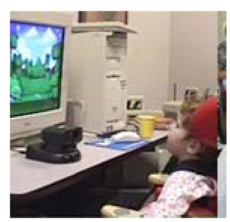
# Outline

- Last time: Motion
  - Motion field and parallax
  - Optical flow, brightness constancy
  - Aperture problem
- Today: Tracking
  - Tracking as inference
  - Linear models of dynamics
  - Kalman filters
  - General challenges in tracking

# Tracking: some applications

Body pose tracking, activity recognition

Censusing a bat population



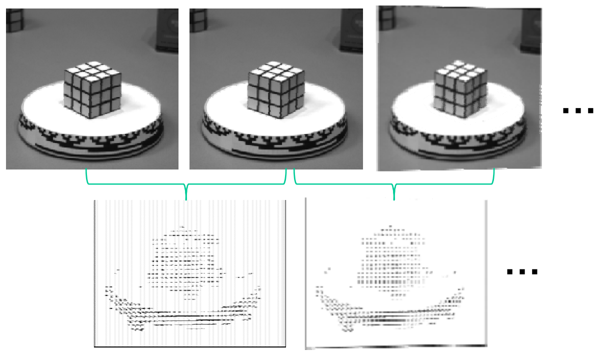
Video-based interfaces

Medical apps


Surveillance

## Optical flow for tracking?

If we have more than just a pair of frames, we could compute flow from one to the next:

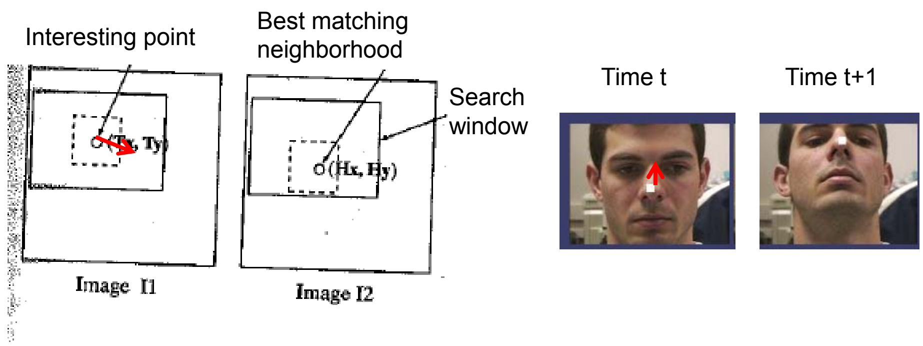



But flow only reliable for small motions, and we may have occlusions, textureless regions that yield bad estimates anyway…

## Motion estimation techniques

- Direct methods
  - Directly recover image motion at each pixel from spatio-temporal image brightness variations
  - Dense motion fields, but sensitive to appearance variations
  - Suitable for video and when image motion is small
- **Feature-based methods**
  - Extract visual features (corners, textured areas) and track them over multiple frames
  - Sparse motion fields, but more robust tracking
  - Suitable when image motion is large (10s of pixels)

# Feature-based matching for motion

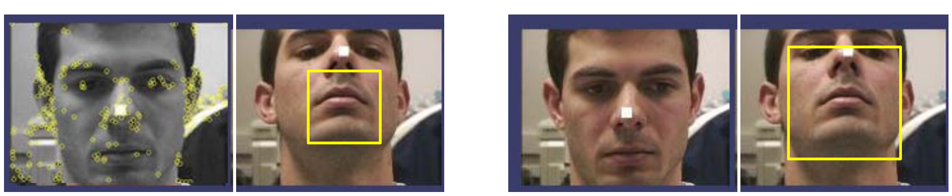



Search window is centered at the point where we last saw the feature, in image I1.

Best match = position where we have the highest normalized cross-correlation value.

# Feature-based matching for motion

- For a discrete matching search, what are the tradeoffs of the chosen **search window** size?



- Which patches to track?
  - Select interest points – e.g. corners
- Where should the search window be placed?
  - Near match at previous frame
  - **More generally, taking into account the expected *dynamics* of the object**

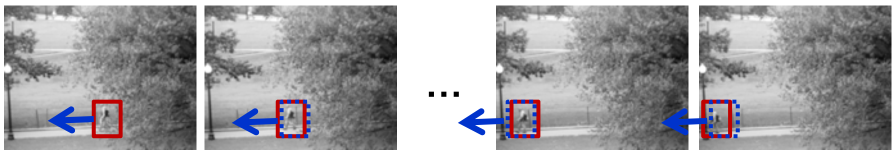
# Detection vs. tracking

t=1 t=2 t=20 t=21

# Detection vs. tracking

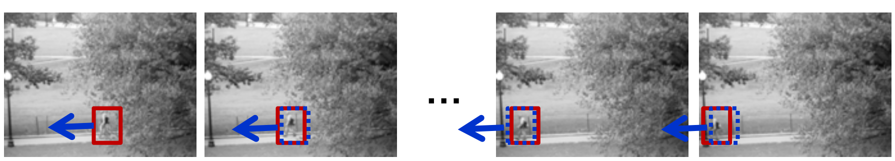
Detection: We detect the object independently in each frame and can record its position over time, e.g., based on blob's centroid or detection window coordinates

# Detection vs. tracking



Tracking with *dynamics*: We use image measurements to estimate position of object, but also incorporate position predicted by dynamics, i.e., our expectation of object’s motion pattern.

Detection vs. tracking



Tracking with *dynamics*: We use image measurements to estimate position of object, but also incorporate position predicted by dynamics, i.e., our expectation of object’s motion pattern.

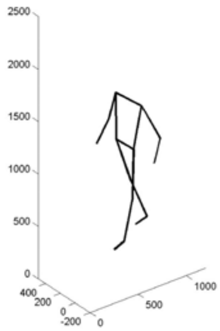
## Tracking with dynamics

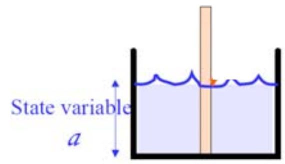
- Use model of expected motion to *predict* where objects will occur in next frame, even before seeing the image.
- **Intent**:
  - Do less work looking for the object, restrict the search.
  - Get improved estimates since measurement noise is tempered by smoothness, dynamics priors.
- **Assumption**: continuous motion patterns:
  - Camera is not moving instantly to new viewpoint
  - Objects do not disappear and reappear in different places in the scene
  - Gradual change in pose between camera and scene

## Tracking as inference

- The *hidden state* consists of the true parameters we care about, denoted *X*.
- The *measurement* is our noisy observation that results from the underlying state, denoted *Y*.

# State vs. observation




Hidden state : parameters of interest

Measurement : what we get to directly observe

# Tracking as inference

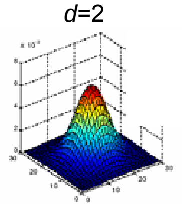
- The *hidden state* consists of the true parameters we care about, denoted *X*.
- The *measurement* is our noisy observation that results from the underlying state, denoted *Y*.
- At each time step, state changes (from $X_{t-1}$ to $X_t$ ) and we get a new observation $Y_t$.
- Our goal: recover most likely state $X_t$ given
  - All observations seen so far.
  - Knowledge about dynamics of state transitions.

# Notation reminder

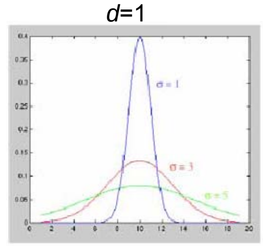
$$\mathbf{x} \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$$

- Random variable with Gaussian probability distribution that has the mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$.
- $\mathbf{x}$ and $\boldsymbol{\mu}$ are $d$-dimensional, $\boldsymbol{\Sigma}$ is $d$ x $d$.

*d*=2

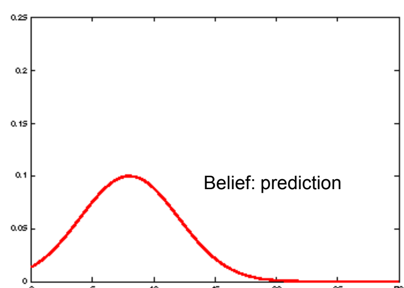


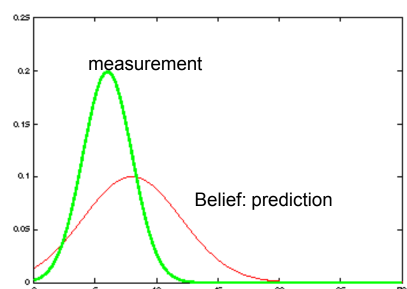
*d*=1

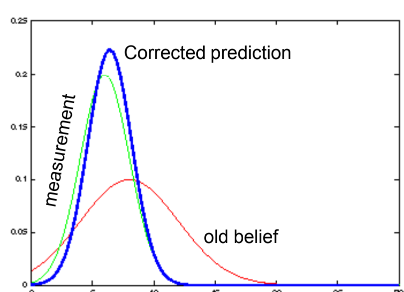


If x is 1-d, we just have one $\boldsymbol{\Sigma}$ parameter - → the variance: $\sigma^2$

# Tracking as inference: intuition




Time t Time t+1

## Standard independence assumptions

- Only immediate past state influences current state

$$P(\boldsymbol{X}_i|\boldsymbol{X}_1,\ldots,\boldsymbol{X}_{i-1}) = P(\boldsymbol{X}_i|\boldsymbol{X}_{i-1})$$

- Measurements at time i only depend on the current state

$$P(\boldsymbol{Y}_i,\boldsymbol{Y}_j,\ldots\boldsymbol{Y}_k|\boldsymbol{X}_i) = P(\boldsymbol{Y}_i|\boldsymbol{X}_i)P(\boldsymbol{Y}_j,\ldots,\boldsymbol{Y}_k|\boldsymbol{X}_i)$$

## Tracking as inference

- Prediction:
  - Given the measurements we have seen **up to** this point, what state should we predict?

$$P\left(X_t \middle| y_0,\ldots,\boxed{y_{t-1}}\right)$$

- Correction:
  - Now given the **current** measurement, what state should we predict?

$$P\left(X_t \middle| y_0,\ldots,\boxed{y_t}\right)$$

# Questions

- How to represent the known dynamics that govern the changes in the states?
- How to represent relationship between state and measurements, plus our uncertainty in the measurements?
- How to compute each cycle of updates?

**Representation**: We'll consider the class of *linear* dynamic models, with associated Gaussian pdfs.

**Updates**: via the Kalman filter.

# Linear dynamic model

- Describe the *a priori* knowledge about
  - System dynamics model: represents evolution of state over time, with noise.

$$\underbrace{\mathbf{x}_t}_{n \times 1} \sim N(\underbrace{\mathbf{D}}_{n \times n}\underbrace{\mathbf{x}_{t-1}}_{n \times 1};\mathbf{\Sigma}_d)$$

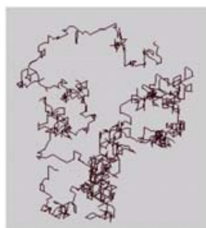
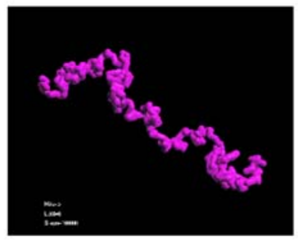
  - Measurement model: at every time step we get a noisy measurement of the state.

$$\underbrace{\mathbf{y}_t}_{m \times 1} \sim N(\underbrace{\mathbf{M}}_{m \times n}\underbrace{\mathbf{x}_t}_{n \times 1};\mathbf{\Sigma}_m)$$

## Example: randomly drifting points

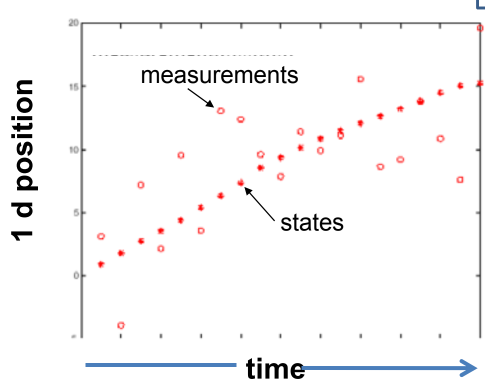
$$\mathbf{x}_t \sim N(\mathbf{D}\mathbf{x}_{t-1}; \mathbf{\Sigma}_d)$$

- Consider a stationary object, with state as position
- Position is constant, only motion due to random noise term.
- State evolution is described by identity matrix **D=I**

cic.nist.gov/lipman/sciviz/images/random3.gif
http://www.grunch.net/synergetics/images/ranc
3.jpg

## Example: Constant velocity (1D points)

## Example: Constant velocity (1D points)

$$\mathbf{x}_t \sim N(\mathbf{D}\mathbf{x}_{t-1}; \mathbf{\Sigma}_d)$$
$$\mathbf{y}_t \sim N(\mathbf{M}\mathbf{x}_t; \mathbf{\Sigma}_m)$$

- State vector: position *p* and velocity *v*

$$x_t = \begin{bmatrix} p_t \\ v_t \end{bmatrix} \qquad \begin{aligned} p_t &= p_{t-1} + (\Delta t) v_{t-1} + \varepsilon \\ v_t &= v_{t-1} + \xi \end{aligned}$$

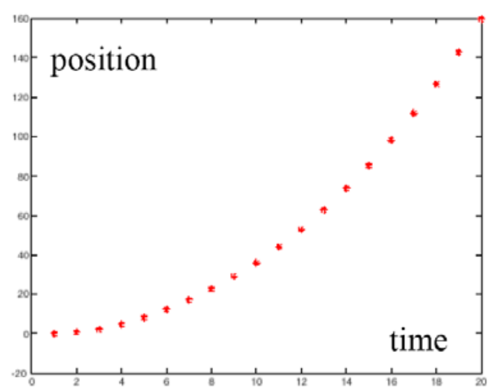
(greek letters denote noise terms)

$$x_t = D_t x_{t-1} + noise = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} p_{t-1} \\ v_{t-1} \end{bmatrix} + noise$$

- Measurement is position only

$$y_t = M x_t + noise = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} p_t \\ v_t \end{bmatrix} + noise$$

## Example: Constant acceleration (1D points)



## Example: Constant acceleration (1D points)

$$\mathbf{x}_t \sim N(\mathbf{D}\mathbf{x}_{t-1};\mathbf{\Sigma}_d)$$
$$\mathbf{y}_t \sim N(\mathbf{M}\mathbf{x}_t;\mathbf{\Sigma}_m)$$

- State vector: position *p*, velocity *v*, and acceleration *a.*

$$x_t = \begin{bmatrix} p_t \\ v_t \\ a_t \end{bmatrix} \qquad \begin{aligned} p_t &= p_{t-1} + (\Delta t)v_{t-1} + \varepsilon \\ v_t &= v_{t-1} + (\Delta t)a_{t-1} + \xi \\ a_t &= a_{t-1} + \zeta \end{aligned}$$

(greek letters denote noise terms)

$$x_t = D_t x_{t-1} + noise = \begin{bmatrix} 1 & \Delta t & 0 \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_{t-1} \\ v_{t-1} \\ a_{t-1} \end{bmatrix} + noise$$

- Measurement is position only

$$y_t = Mx_t + noise = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} p_t \\ v_t \\ a_t \end{bmatrix} + noise$$

# Questions

- How to represent the known dynamics that govern the changes in the states?
- How to represent relationship between state and measurements, plus our uncertainty in the measurements?
- How to compute each cycle of updates?

**Representation**: We'll consider the class of *linear* dynamic models, with associated Gaussian pdfs.

**Updates**: via the Kalman filter.

# The Kalman filter

- Method for tracking linear dynamical models in Gaussian noise
- The predicted/corrected state distributions are Gaussian
  - Only need to maintain the mean and covariance
  - The calculations are easy (all the integrals can be done in closed form)

# Kalman filter

Know corrected state from previous time step, and all measurements up to the current one → Predict distribution over next state.

*Receive measurement*

Know prediction of state, and next measurement → Update distribution over current state.

**Time update ("Predict")**

**Measurement update ("Correct")**

$P(X_t | y_0, \ldots, y_{t-1})$

*Time advances: t++*

$P(X_t | y_0, \ldots, y_t)$

Mean and std. dev. of predicted state:

$\mu_t^-, \sigma_t^-$

Mean and std. dev. of corrected state:

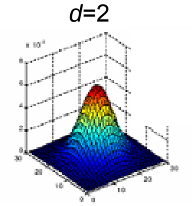
$\mu_t^+, \sigma_t^+$

# Notation reminder

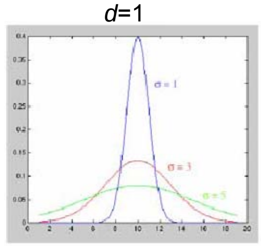
$$\mathbf{x} \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$$

- Random variable with Gaussian probability distribution that has the mean vector **μ** and covariance matrix **Σ**.
- **x** and **μ** are *d*-dimensional, **Σ** is *d* x *d*.

*d*=2



*d*=1



If x is 1-d, we just have one **Σ** parameter -→ the variance: $\sigma^2$

## 1D Kalman filter: **Prediction**

- Have linear dynamic model defining predicted state evolution, with noise

$$X_t \sim N\left(dx_{t-1}, \sigma_d^2\right)$$

- Want to estimate predicted distribution for next state

$$P\left(X_t \middle| y_0, \ldots, y_{t-1}\right) = N\left(\mu_t^-, (\sigma_t^-)^2\right)$$

- Update the mean:

$$\mu_t^- = d\mu_{t-1}^+$$

- Update the variance:

$$(\sigma_t^-)^2 = \sigma_d^2 + (d\sigma_{t-1}^+)^2$$

## 1D Kalman filter: **Correction**

- Have linear model defining the mapping of state to measurements:

$$Y_t \sim N\left(mx_t, \sigma_m^2\right)$$

- Want to estimate corrected distribution given latest meas.: $P\left(X_t \middle| y_0, \ldots, y_t\right) = N\left(\mu_t^+, (\sigma_t^+)^2\right)$
- Update the mean:

$$\mu_t^+ = \frac{\mu_t^- \sigma_m^2 + m y_t (\sigma_t^-)^2}{\sigma_m^2 + m^2 (\sigma_t^-)^2}$$

- Update the variance:

$$(\sigma_t^+)^2 = \frac{\sigma_m^2 (\sigma_t^-)^2}{\sigma_m^2 + m^2 (\sigma_t^-)^2}$$

## Prediction vs. correction

$$\mu_t^+ = \frac{\mu_t^- \sigma_m^2 + m y_t (\sigma_t^-)^2}{\sigma_m^2 + m^2 (\sigma_t^-)^2} \qquad (\sigma_t^+)^2 = \frac{\sigma_m^2 (\sigma_t^-)^2}{\sigma_m^2 + m^2 (\sigma_t^-)^2}$$

- What if there is no prediction uncertainty $(\sigma_t^- = 0)$?

$$\mu_t^+ = \mu_t^- \qquad (\sigma_t^+)^2 = 0$$

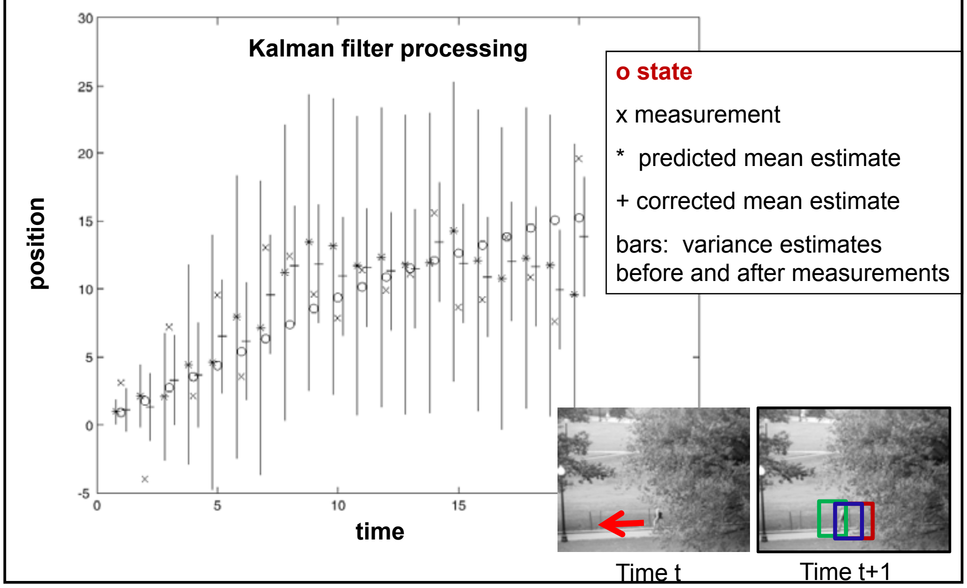
The measurement is ignored!

- What if there is no measurement uncertainty $(\sigma_m = 0)$?

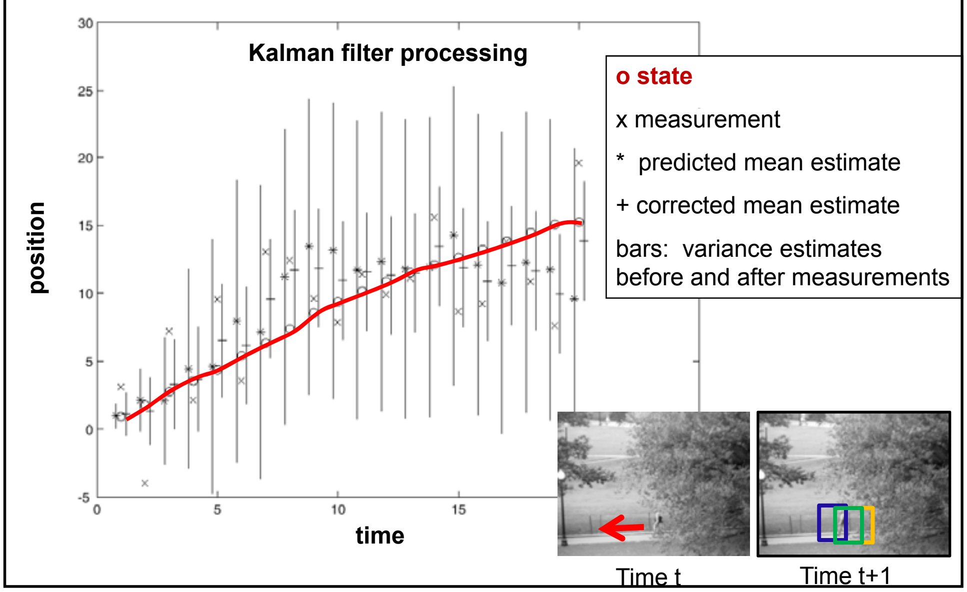
$$\mu_t^+ = \frac{y_t}{m} \qquad (\sigma_t^+)^2 = 0$$

The prediction is ignored!

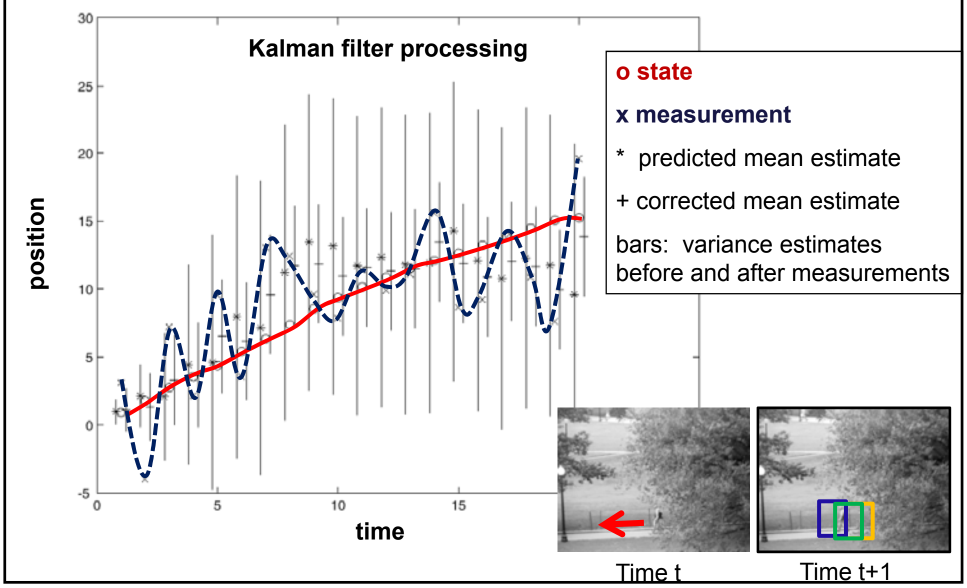
# Constant velocity model



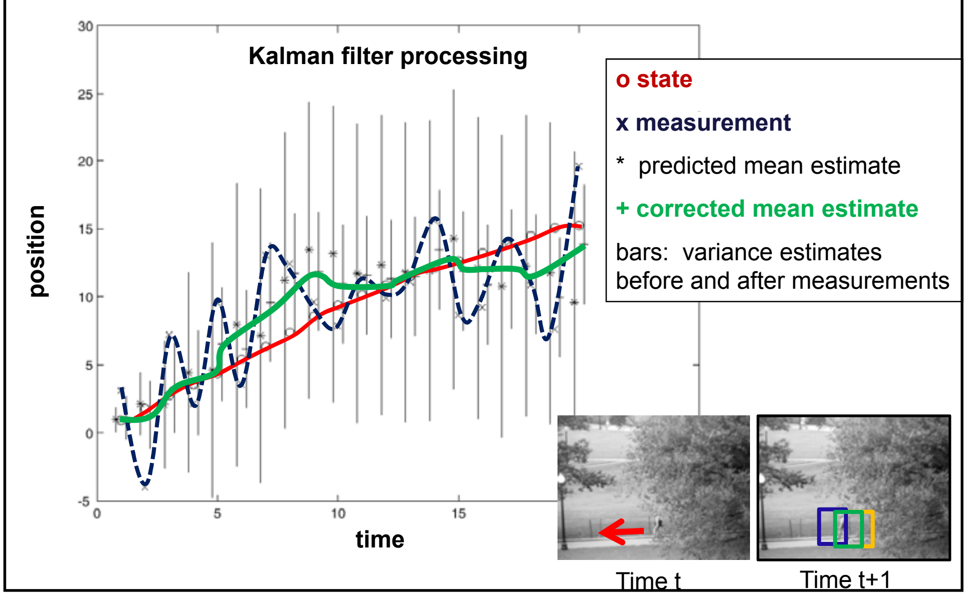

# Constant velocity model



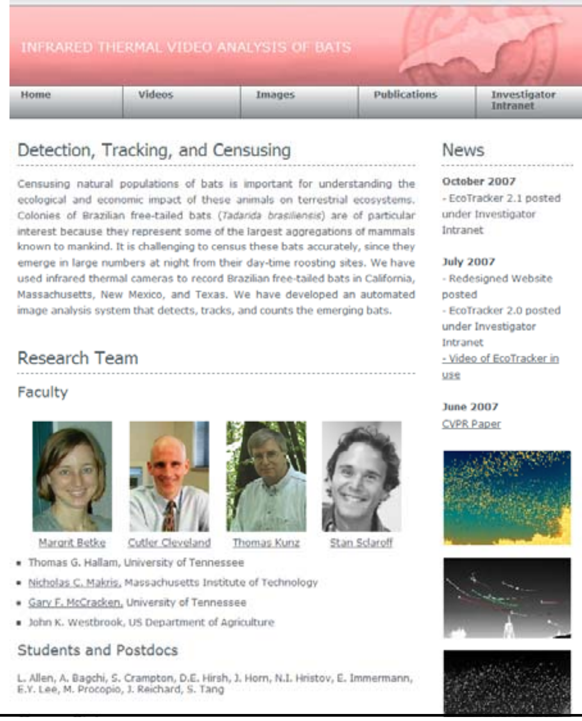
# Constant velocity model

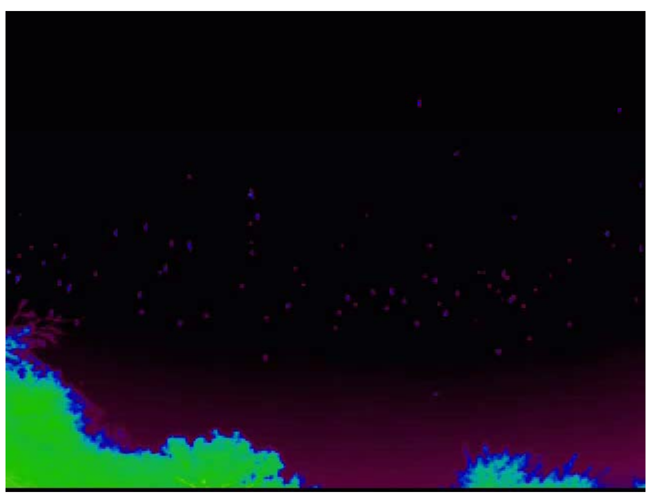



# Constant velocity model



- http://www.cs.bu.edu/~betke/research/bats/



# Outline

- Last time: Motion
  - Motion field and parallax
  - Optical flow, brightness constancy
  - Aperture problem
- Today: Tracking
  - Tracking as inference
  - Linear models of dynamics
  - Kalman filters
  - General challenges in tracking

# Tracking: issues

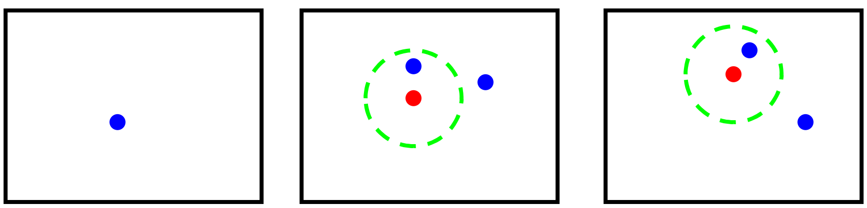
- **Initialization**
  - Often done manually
  - Background subtraction, detection can also be used
- **Data association**, multiple tracked objects
  - Occlusions, clutter

# Tracking: issues

- **Initialization**
  - Often done manually
  - Background subtraction, detection can also be used
- **Data association**, multiple tracked objects
  - Occlusions, clutter
  - Which measurements go with which tracks?

## Data association

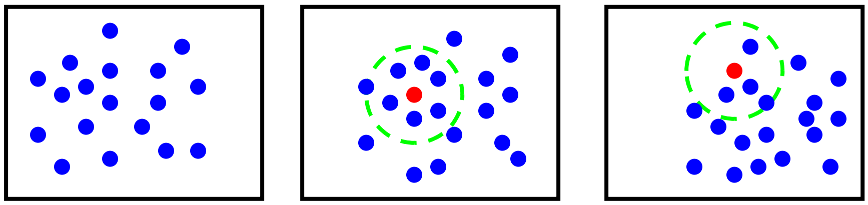
- Simple strategy: only pay attention to the measurement that is "closest" to the prediction



Slide credit: Lana Lazebnik

## Data association

- Simple strategy: only pay attention to the measurement that is “closest” to the prediction

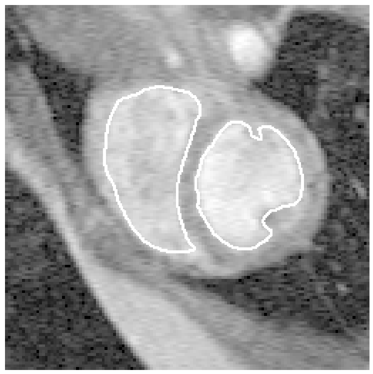


# Tracking: issues

- **Initialization**
  - Often done manually
  - Background subtraction, detection can also be used
- **Data association**, multiple tracked objects
  - Occlusions, clutter
- **Deformable** and articulated objects

# Tracking via deformable contours

1. Use final contour/model extracted at frame *t* as an initial solution for frame *t+1*
2. Evolve initial contour to fit exact object boundary at frame *t+1*
3. Repeat, initializing with most recent frame.



Tracking Heart Ventricles
(multiple frames)

# Tracking via deformable contours

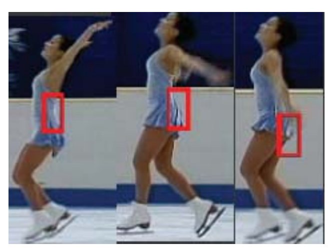
Visual Dynamics Group, Dept. Engineering Science, University of Oxford.

Applications: Traffic monitoring
Human-computer interaction
Animation
Surveillance
Computer assisted diagnosis in medical imaging

# Tracking: issues

- **Initialization**
  - Often done manually
  - Background subtraction, detection can also be used
- **Data association**, multiple tracked objects
  - Occlusions, clutter
- **Deformable** and articulated objects
- **Constructing accurate models** of dynamics
  - E.g., Fitting parameters for a linear dynamics model
- **Drift**
  - Accumulation of errors over time

## Drift



D. Ramanan, D. Forsyth, and A. Zisserman. Tracking People by Learning their Appearance. PAMI 2007.

# Summary

- Tracking as inference
  - Goal: estimate posterior of object position given measurement
- Linear models of dynamics
  - Represent state evolution and measurement models
- Kalman filters
  - Recursive prediction/correction updates to refine measurement
- General tracking challenges