

T15 - Tracking

Computer Vision, FCUP, 2013
Miguel Coimbra
Slides by Prof. Kristen Grauman

A new way to communicate

The mouse and keyboard were invented before the Internet even existed. Since then, countless technological advancements have allowed for much more efficient human computer interaction. Why then do we continue to use outdated technology? Introducing Gmail Motion -- now you can control Gmail with your body.



Try Gmail Motion

Don't have Gmail? [Create an account](#)

-  **Easy to learn**
Simple and intuitive gestures
-  **Improved productivity**
In and out of your email up to 12% faster
-  **Increased physical activity**
Get out of that chair and start moving today



Overview



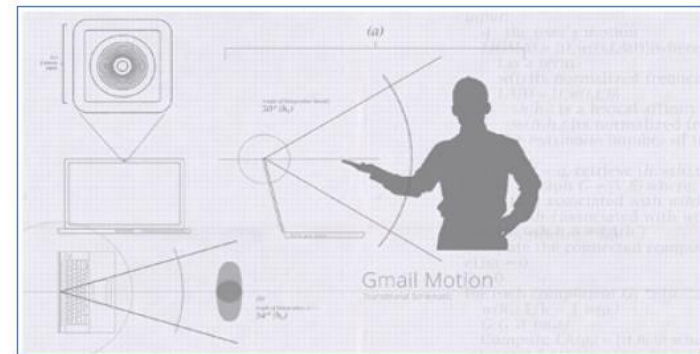
Paralangue expert



Movement specialist

How it works

Gmail Motion uses your computer's built-in webcam and Google's patented spatial tracking technology to detect your movements and translate them into meaningful characters and commands. Movements are designed to be simple and intuitive for people of all skill levels.



A new way to collaborate

Coming soon! Google Docs Motion ^{BETA} will introduce a new way to collaborate -- using your body:



Gesture recognition

Over 10,000 supported gestures



Emotion detection

Writing tone corresponds with emotions



Intuitive

Start being more productive in a few minutes



[Documents](#)



[Spreadsheets](#)



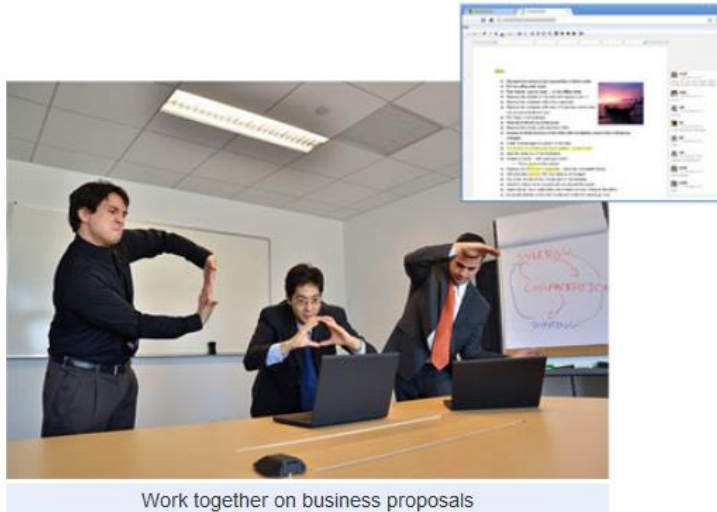
[Presentations](#)



[Drawings](#)



[Document list](#)



Work together on business proposals

Documents

Collaborate in real time using Google Docs Motion ^{BETA}. Building on the technology developed for [Gmail Motion](#) ^{BETA}, multiple computers can triangulate position to track each collaborator's inputs. Every feature will be supported, from brand new discussions to formatting, tables and translation.

Spreadsheets

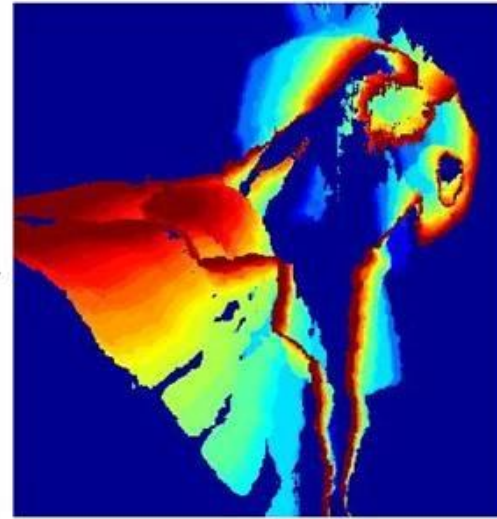
What do you get when you combine motion and Google spreadsheets? An incredible new way to create charts, add data, filter and work together. Docs Motion now makes it easy to adjust the look and feel of your charts. Pie charts are easy to make.



Pset 5



Depth map sequence



Motion History Image



kick!

Nearest neighbor action classification with
Motion History Images + Hu moments

Normalized Euclidean distance

$$D(h_1, h_2) = \sqrt{\sum_{i=1}^d \frac{(h_1(i) - h_2(i))^2}{\sigma_i^2}}$$

Normalize according to variance in each dimension

What does this do for our distance computation?

Leave-one-out cross validation

- Cycle through data points, treating each one as the “test” case in turn, and training with the remaining labeled examples.
- Report results over all such test cases

Outline

- Today: Tracking
 - Tracking as inference
 - Linear models of dynamics
 - Kalman filters
 - General challenges in tracking

Tracking: some applications



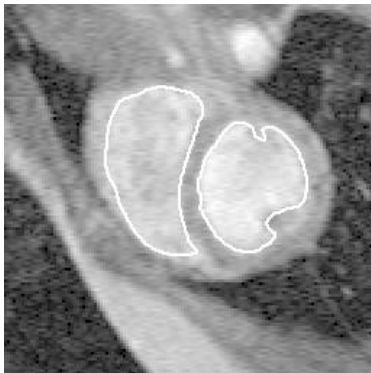
Body pose tracking,
activity recognition



Censusing a bat
population



Video-based
interfaces



Medical apps

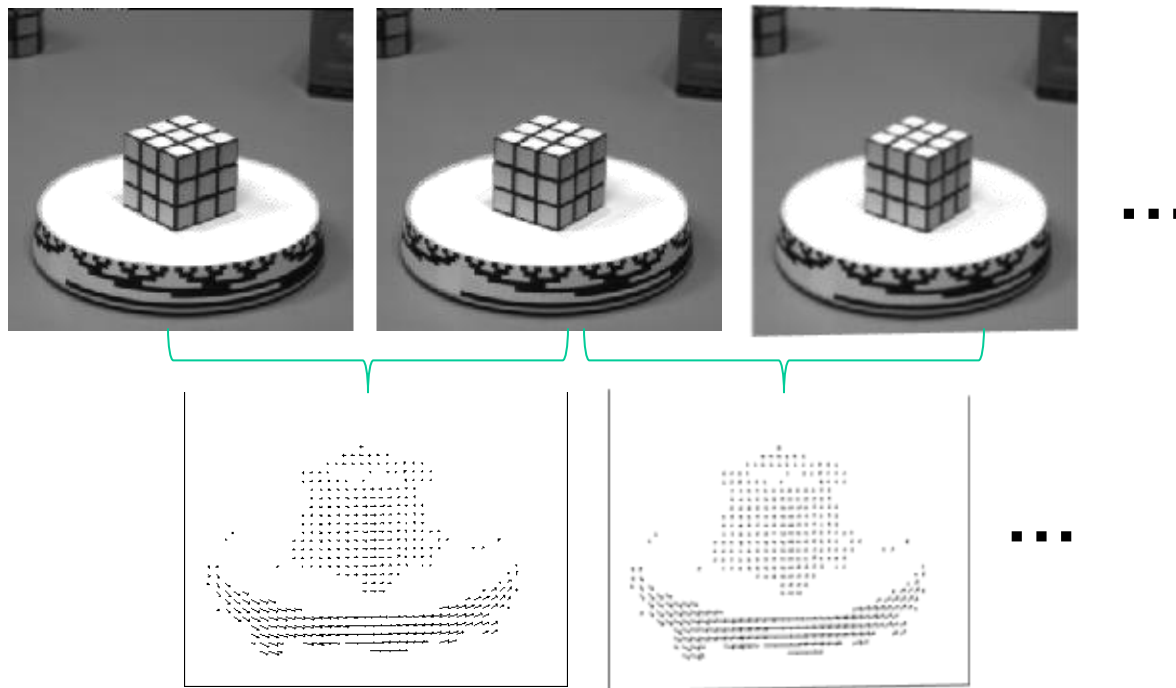


Surveillance

Why is tracking challenging?

Optical flow for tracking?

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

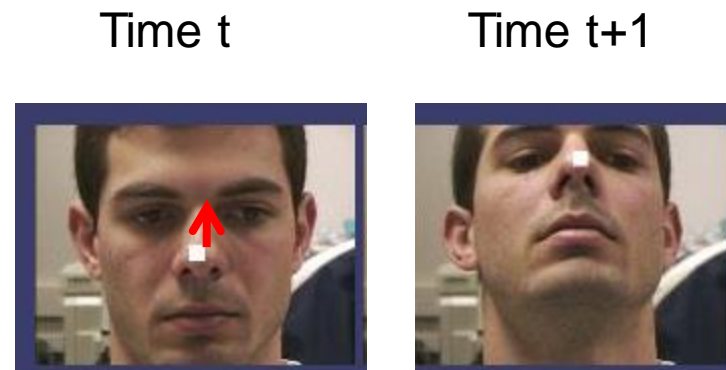
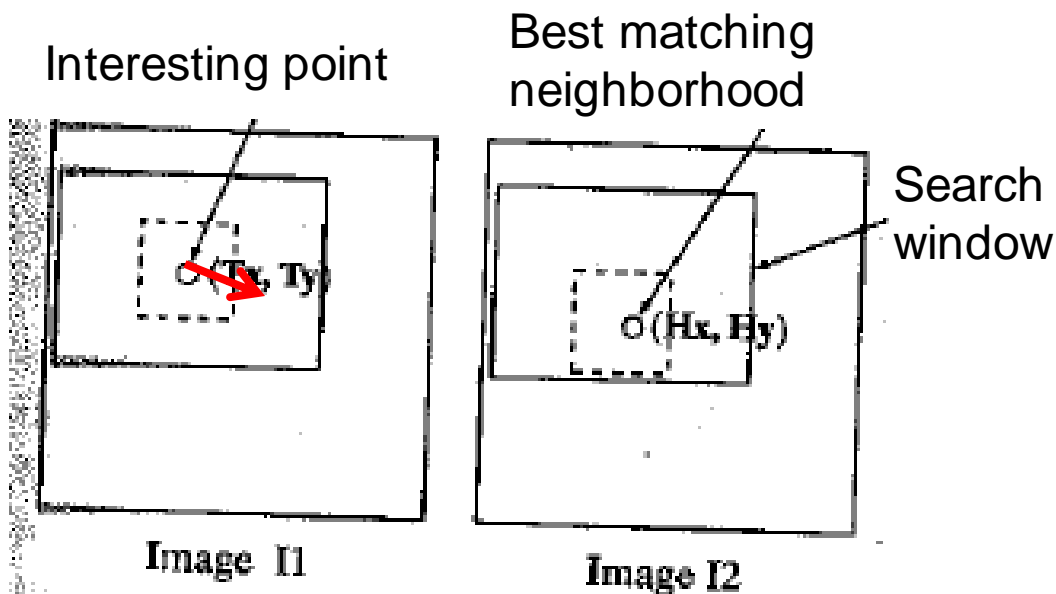


But flow only reliable for small motions, and we may have occlusions, textureless regions that yield bad estimates anyway...

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.

Example: A Camera Mouse

Video interface: use feature tracking as mouse replacement



- User clicks on the feature to be tracked
- Take the 15x15 pixel square of the feature
- In the next image do a search to find the 15x15 region with the highest correlation
- Move the mouse pointer accordingly
- Repeat in the background every 1/30th of a second

Example: A Camera Mouse

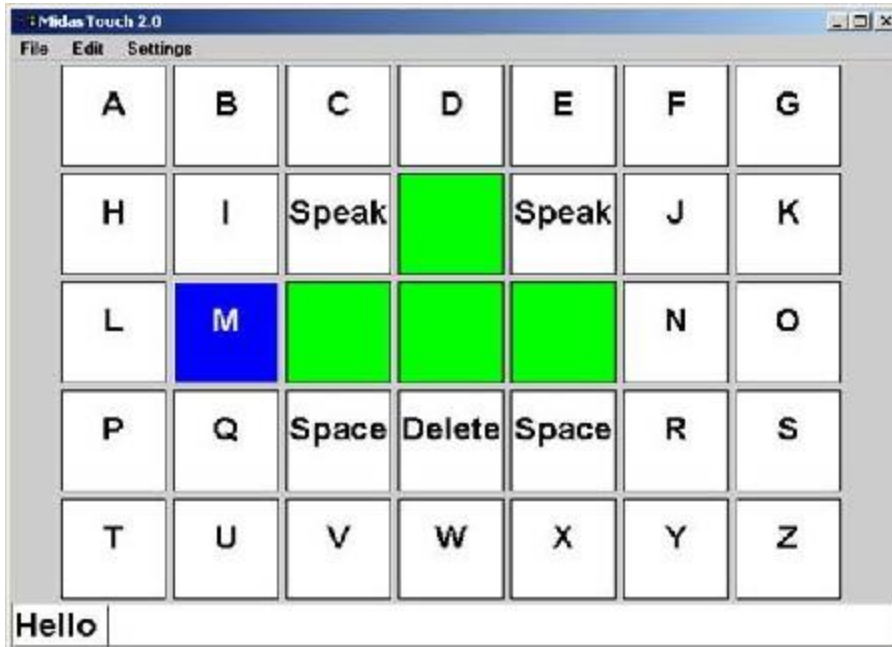
Specialized software for communication, games



James Gips and Margrit Betke
<http://www.bc.edu/schools/csom/eagleeyes/>

A Camera Mouse

Specialized software for communication, games



James Gips and Margrit Betke
<http://www.bc.edu/schools/csom/eagleeyes/>

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



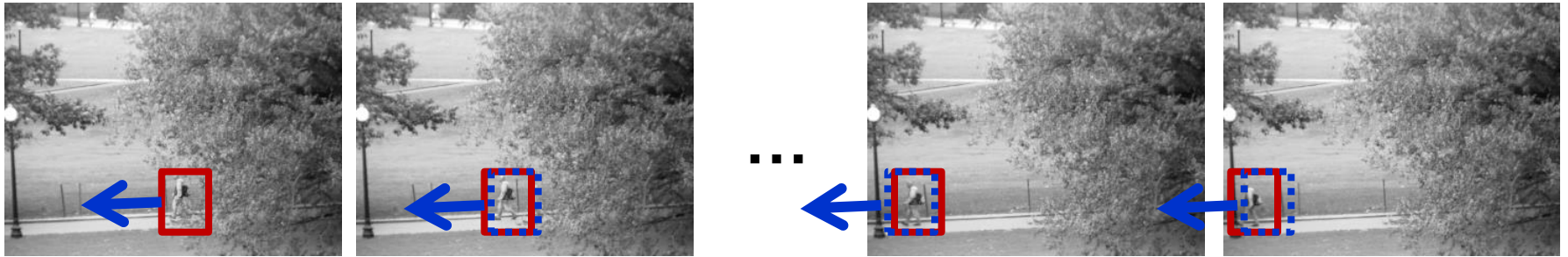
t=21

Detection vs. tracking



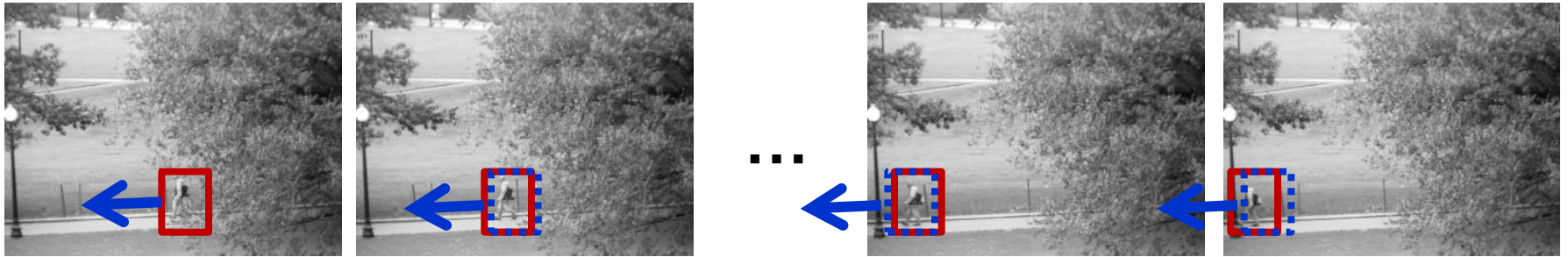
Detection: We detect the object independently in each frame and can record its position over time, e.g., based on blob's centroid or detection window coordinates

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.

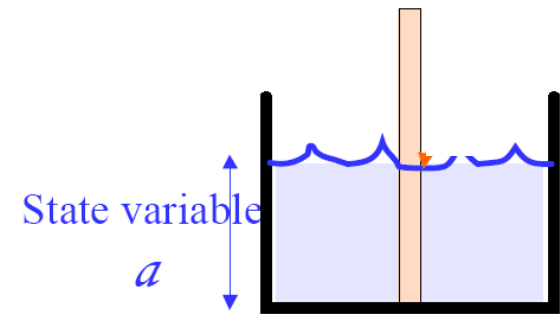
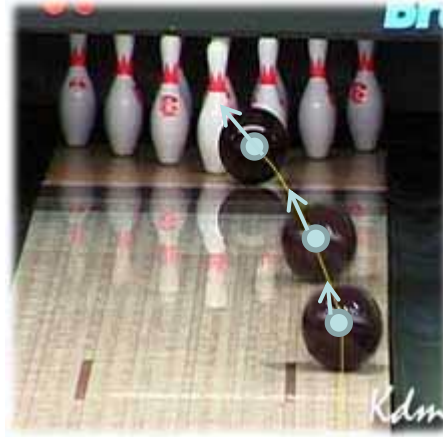
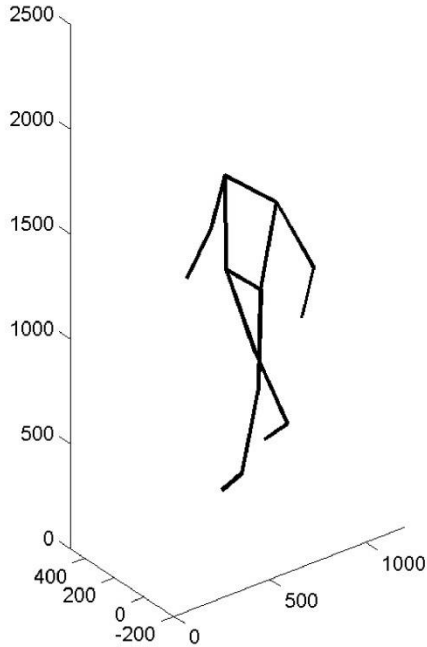
Tracking with dynamics

- 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 .
- At each time step, state changes (from X_{t-1} to X_t) and we get a new observation Y_t .

State vs. observation



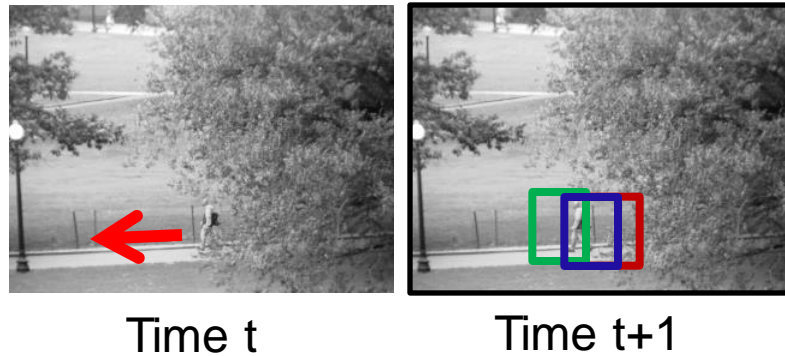
Hidden state : parameters of interest

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.

Tracking as inference: intuition

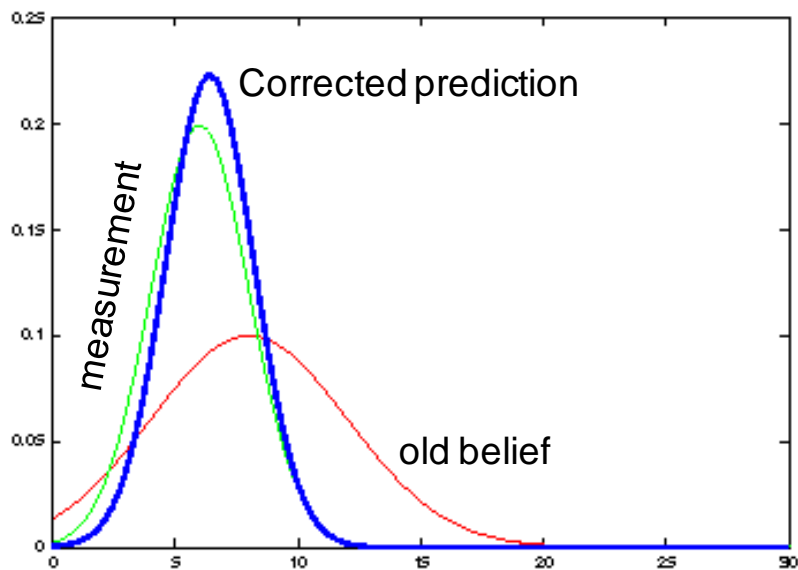
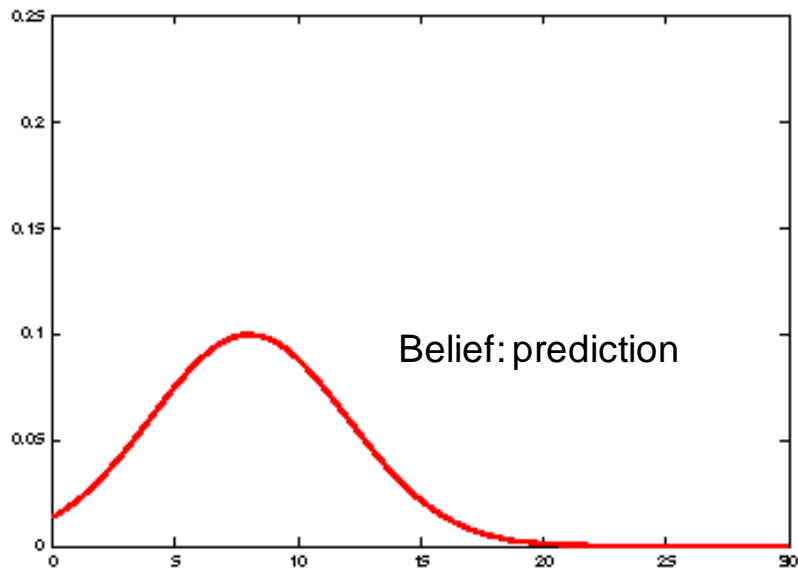


Belief

Measurement

Corrected prediction

Tracking as inference: intuition



Time t

Time t+1

Independence assumptions

- Only immediate past state influences current state

$$P(X_t | X_0, \dots, X_{t-1}) = P(X_t | X_{t-1})$$

dynamics model

- Measurement at time t depends on current state

$$P(Y_t | X_0, Y_0, \dots, X_{t-1}, Y_{t-1}, X_t) = P(Y_t | X_t)$$

observation model

Tracking as inference

- Prediction:

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

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

- Correction:

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

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

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?

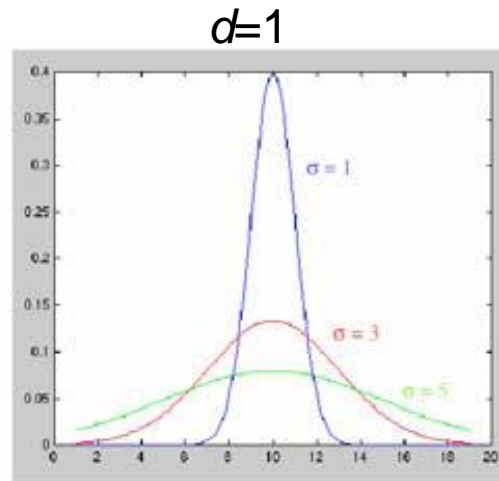
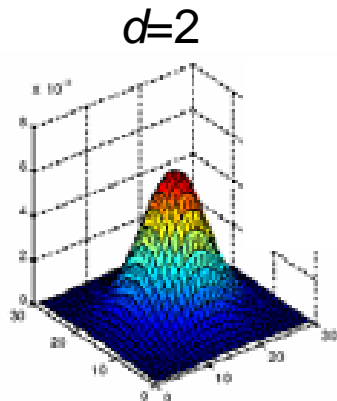
Representation: We'll consider the class of *linear* dynamic models, with associated Gaussian pdfs.

Updates: via the Kalman filter.

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 $d \times d$.



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

Linear dynamic model

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

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

$\begin{matrix} \uparrow & \nearrow & \nwarrow \\ n \times 1 & n \times n & n \times 1 \end{matrix}$

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

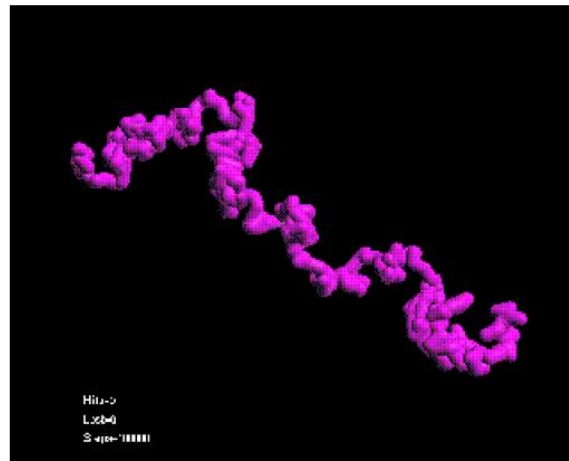
$$\mathbf{y}_t \sim N(\mathbf{M}\mathbf{x}_t; \Sigma_m)$$

$\begin{matrix} \uparrow & \nearrow & \nwarrow \\ m \times 1 & m \times n & n \times 1 \end{matrix}$

Example: randomly drifting points

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

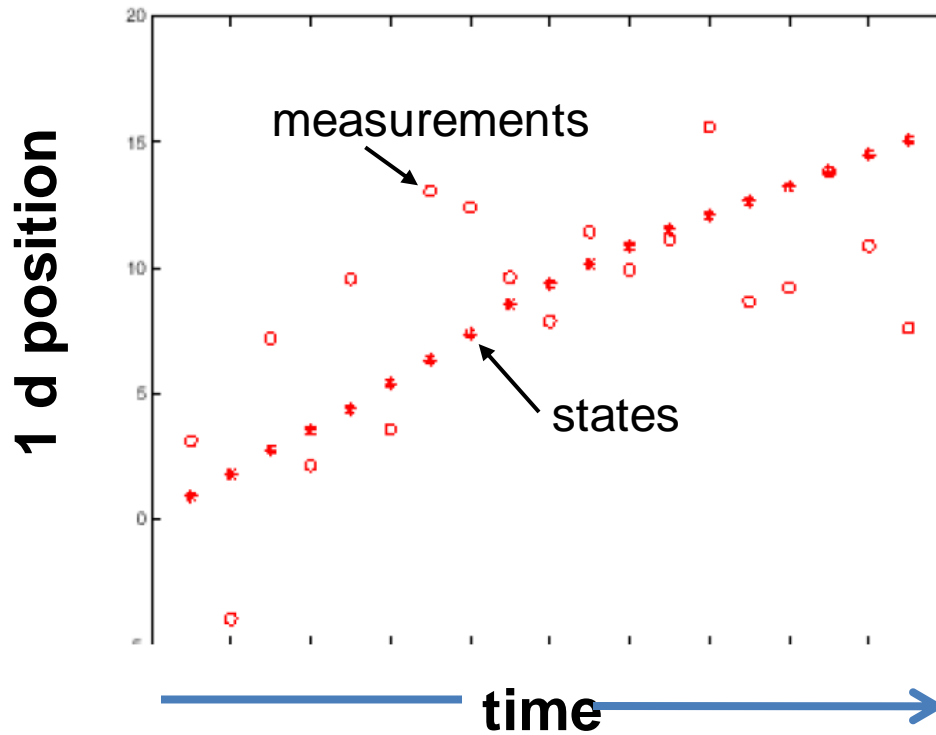
- Consider a stationary object, with state as position
- Position is constant, only motion due to random noise term.
- State evolution is described by identity matrix $\mathbf{D}=\mathbf{I}$



Example: Constant velocity (1D points)



1 d position



Example: Constant velocity (1D points)

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

$$\mathbf{y}_t \sim N(\mathbf{M}\mathbf{x}_t; \Sigma_m)$$

- State vector: position p and velocity v

$$\mathbf{x}_t = \begin{bmatrix} p_t \\ v_t \end{bmatrix} \quad p_t =$$

$$x_t = D_t x_{t-1} + noise =$$

- Measurement is position only

$$y_t = Mx_t + noise =$$

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

Kalman filter

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

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

Receive measurement

**Time update
("Predict")**

**Measurement update
("Correct")**

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

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

Mean and std. dev. of predicted state:

$$\mu_t^-, \sigma_t^-$$

Time advances: t++

Mean and std. dev. of corrected state:

$$\mu_t^+, \sigma_t^+$$

1D Kalman filter: Prediction

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

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

- Want to estimate predicted distribution for next state

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

- 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:

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

- Want to estimate corrected distribution given latest meas.: $P(X_t | y_0, \dots, y_t) = N(\mu_t^+, (\sigma_t^+)^2)$

- 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

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

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

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

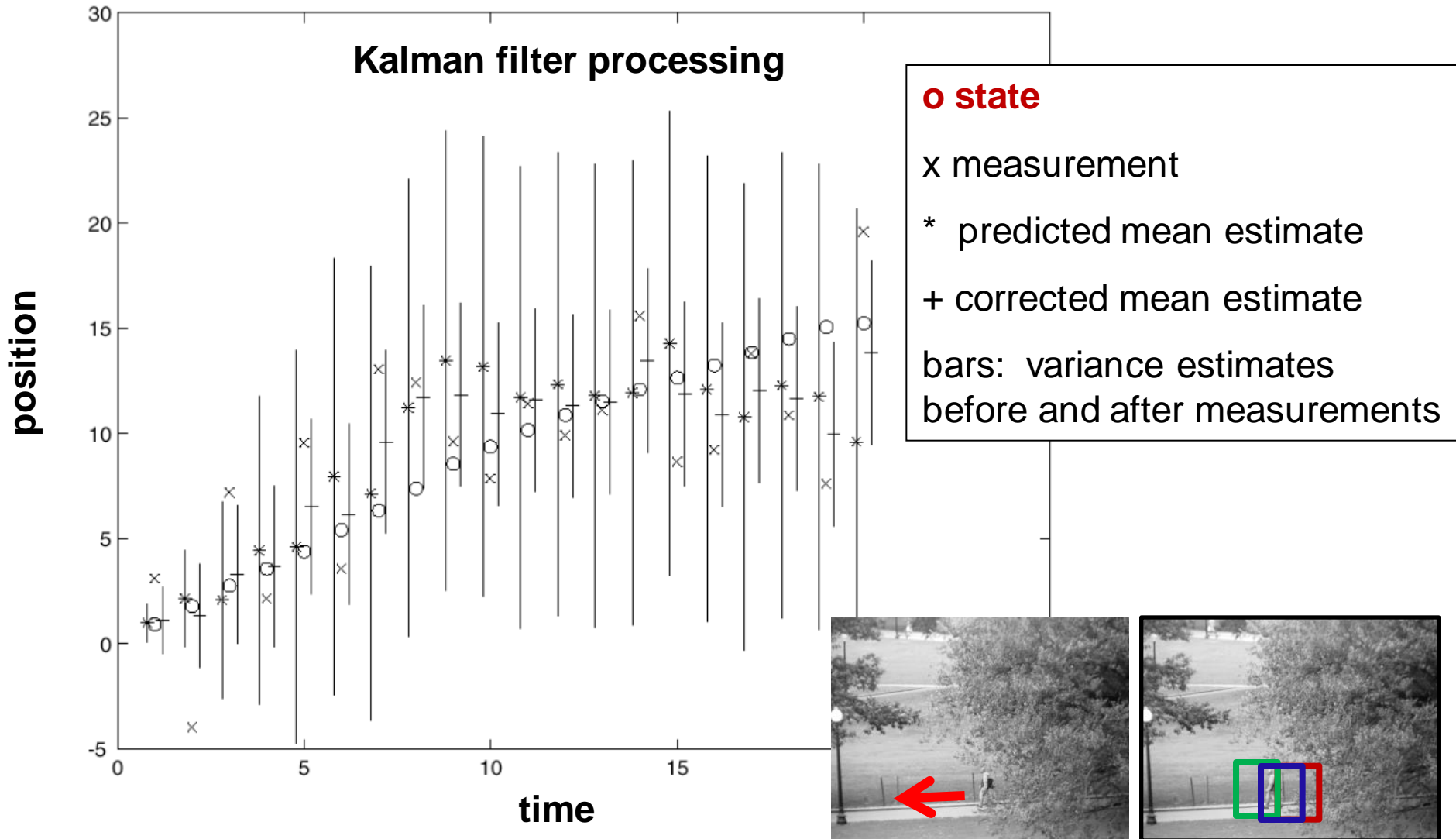
The measurement is ignored!

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

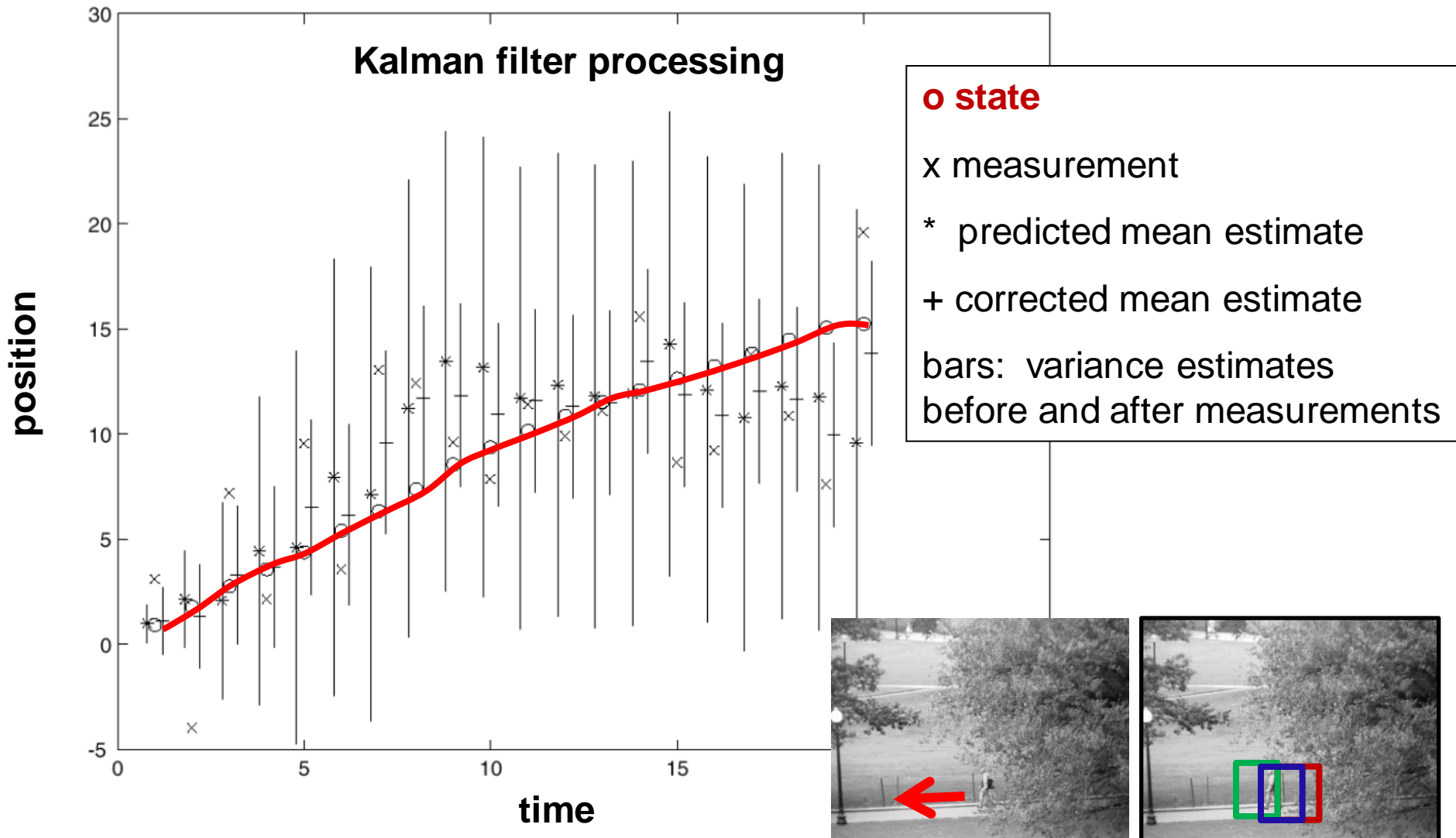
$$\mu_t^+ = \frac{y_t}{m} \quad (\sigma_t^+)^2 = 0$$

The prediction is ignored!

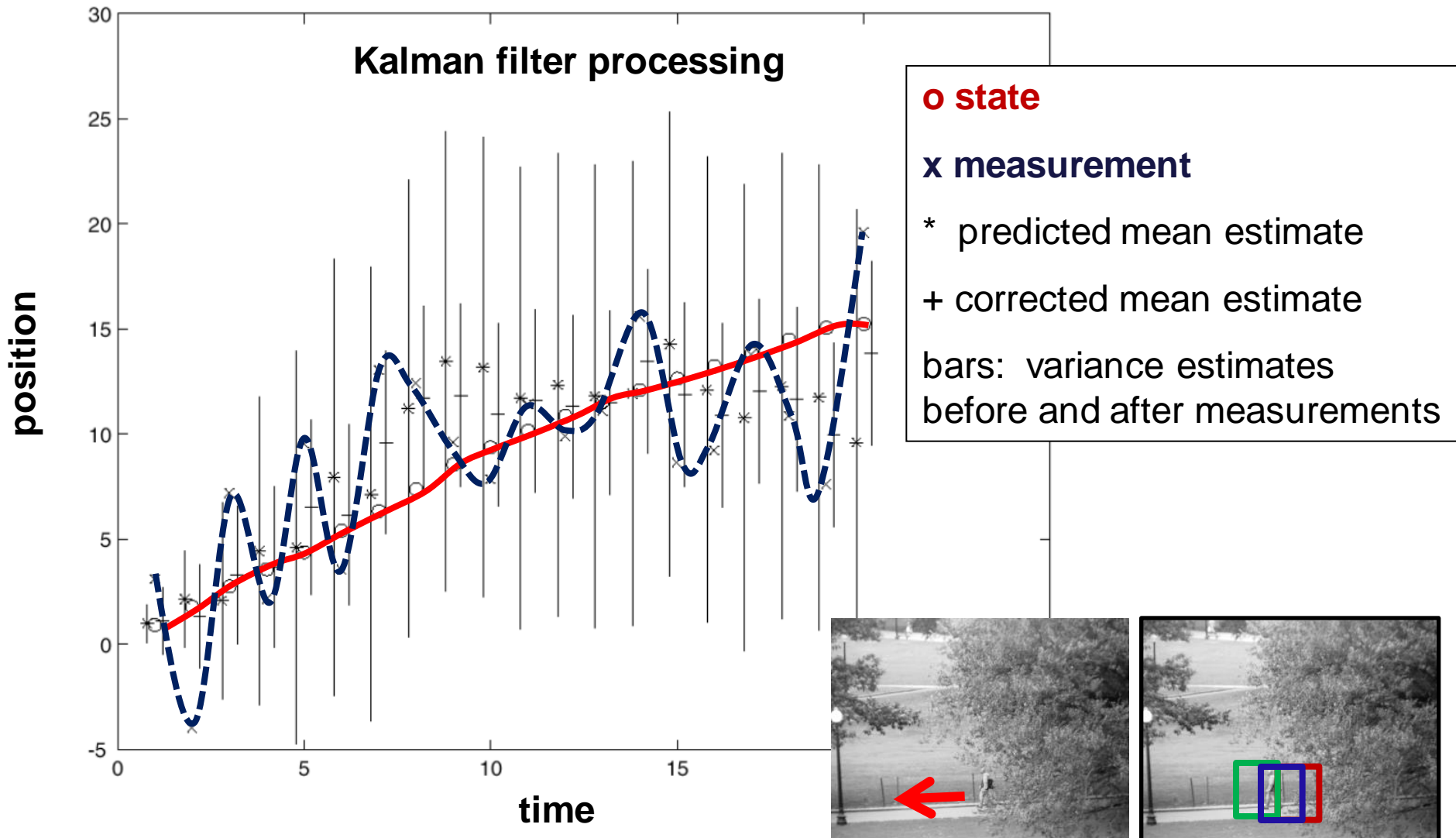
Constant velocity model



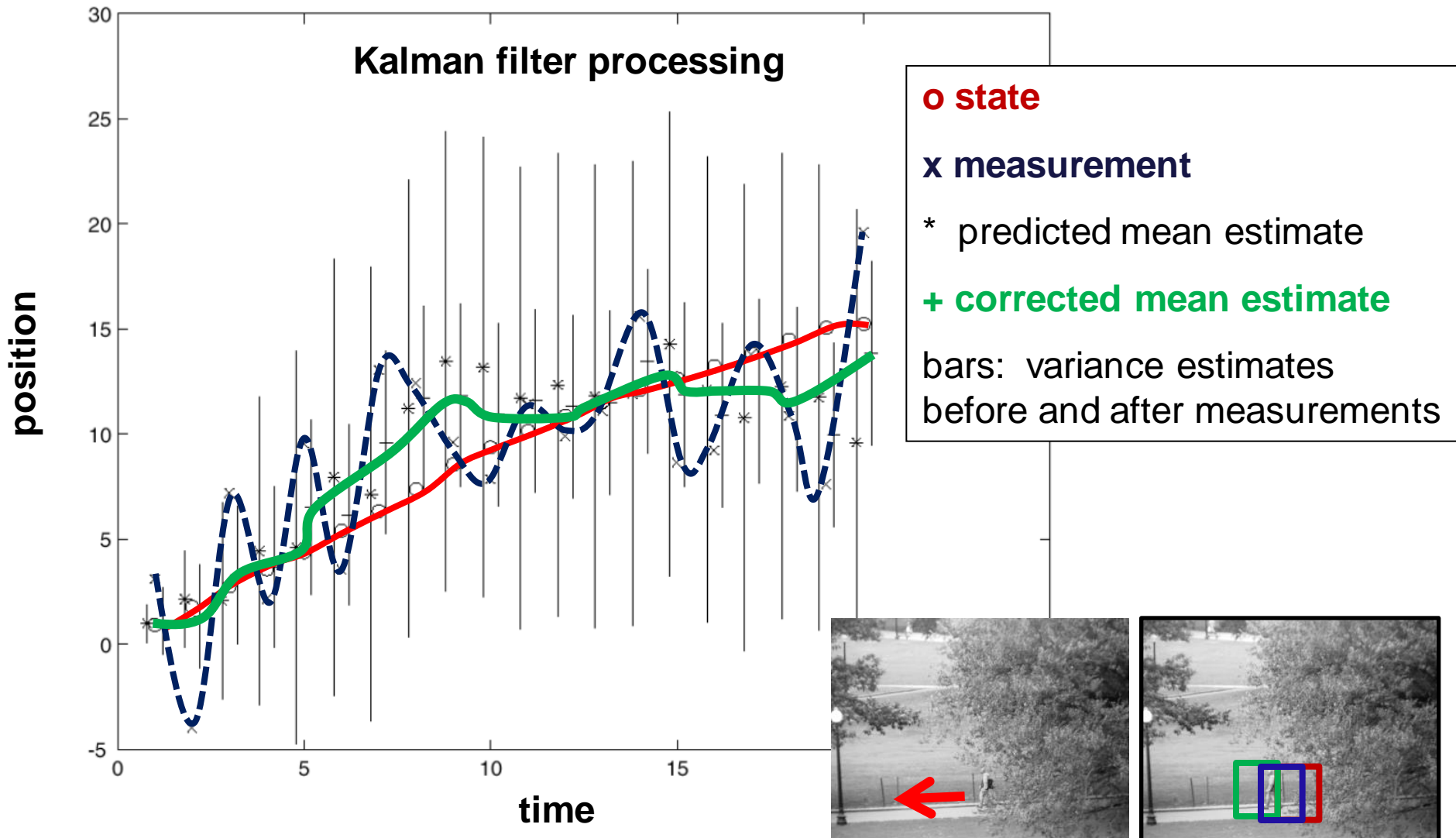
Constant velocity model



Constant velocity model



Constant velocity model



Detection, Tracking, and Censusing

Censusing natural populations of bats is important for understanding the ecological and economic impact of these animals on terrestrial ecosystems. Colonies of Brazilian free-tailed bats (*Tadarida brasiliensis*) are of particular interest because they represent some of the largest aggregations of mammals known to mankind. It is challenging to census these bats accurately, since they emerge in large numbers at night from their day-time roosting sites. We have used infrared thermal cameras to record Brazilian free-tailed bats in California, Massachusetts, New Mexico, and Texas. We have developed an automated image analysis system that detects, tracks, and counts the emerging bats.

Research Team

Faculty



[Margrit Betke](#)



[Cutler Cleveland](#)



[Thomas Kunz](#)



[Stan Sclaroff](#)

- Thomas G. Hallam, University of Tennessee
- [Nicholas C. Makris](#), Massachusetts Institute of Technology
- [Gary F. McCracken](#), University of Tennessee
- John K. Westbrook, US Department of Agriculture

Students and Postdocs

L. Allen, A. Bagchi, S. Crampton, D.E. Hirsh, J. Horn, N.I. Hristov, E. Immermann, E.Y. Lee, M. Procopio, J. Reichard, S. Tang

News

October 2007

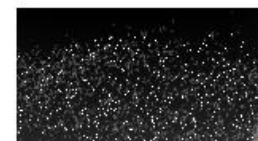
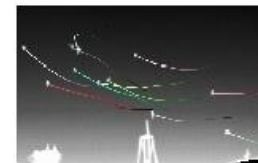
- EcoTracker 2.1 posted under Investigator Intranet

July 2007

- Redesigned Website posted
 - EcoTracker 2.0 posted under Investigator Intranet
 - [Video of EcoTracker in use](#)

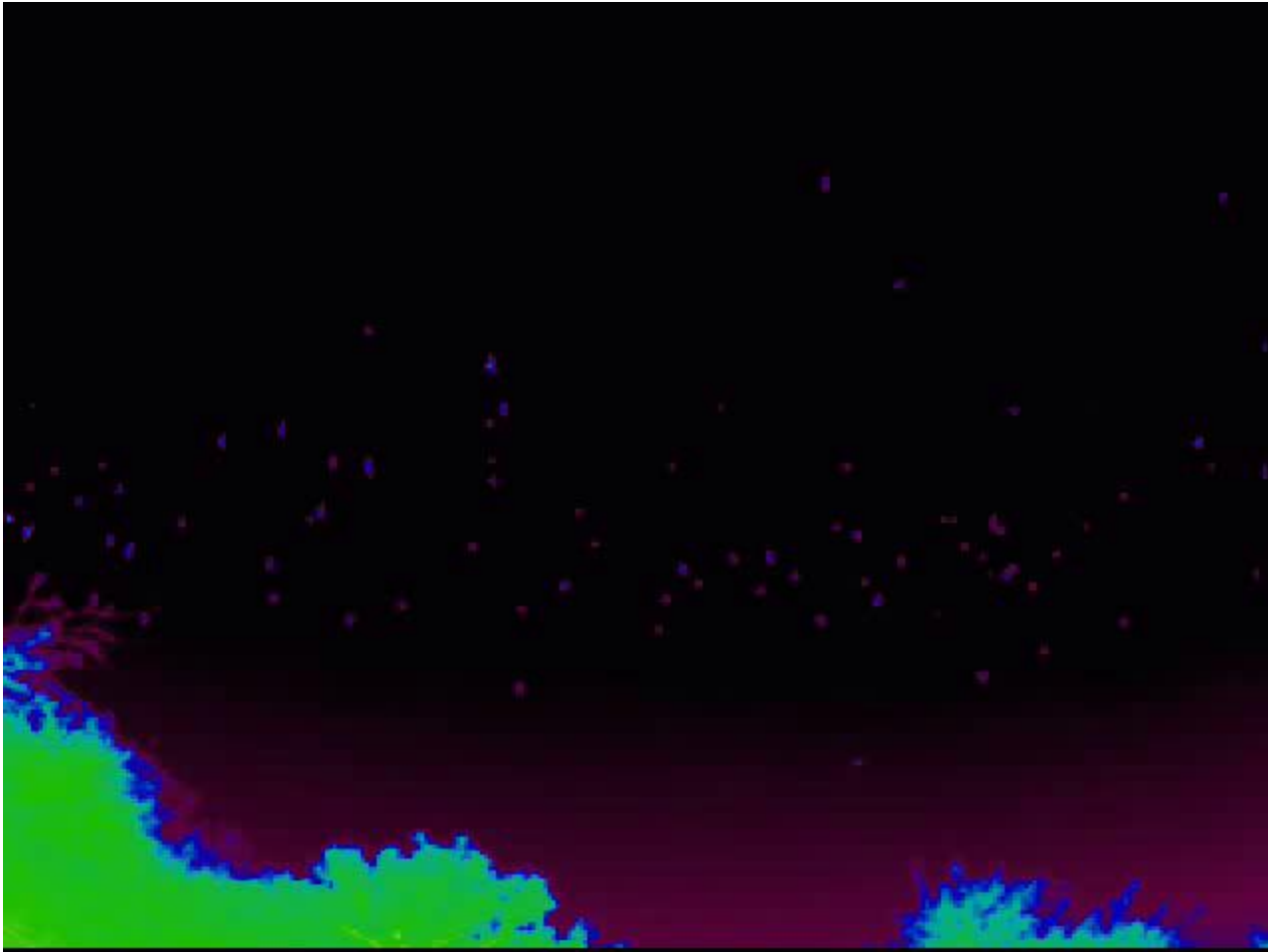
June 2007

[CVPR Paper](#)



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

A bat census



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

Video synopsis

- <http://www.vision.huji.ac.il/video-synopsis/>

Select synopsis below

CAMERA IN STUTTGART AIRPORT (SEE 24 HOURS IN 20 SECONDS!)

Typical Video Stream (24 Hours)
Stuttgart 2007-04-16 14:55:45

Synopsis with Less Collisions
Stuttgart 2007-04-05 12:31:41

Synopsis with More Collisions
Stuttgart 2007-04-05 00:44:18

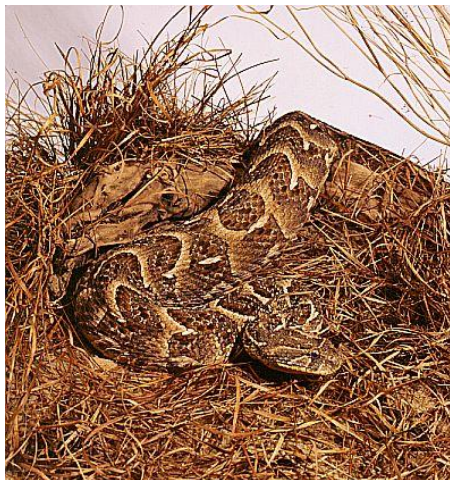


Tracking: issues

- **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?



Tracking: issues

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

Recall:

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: 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