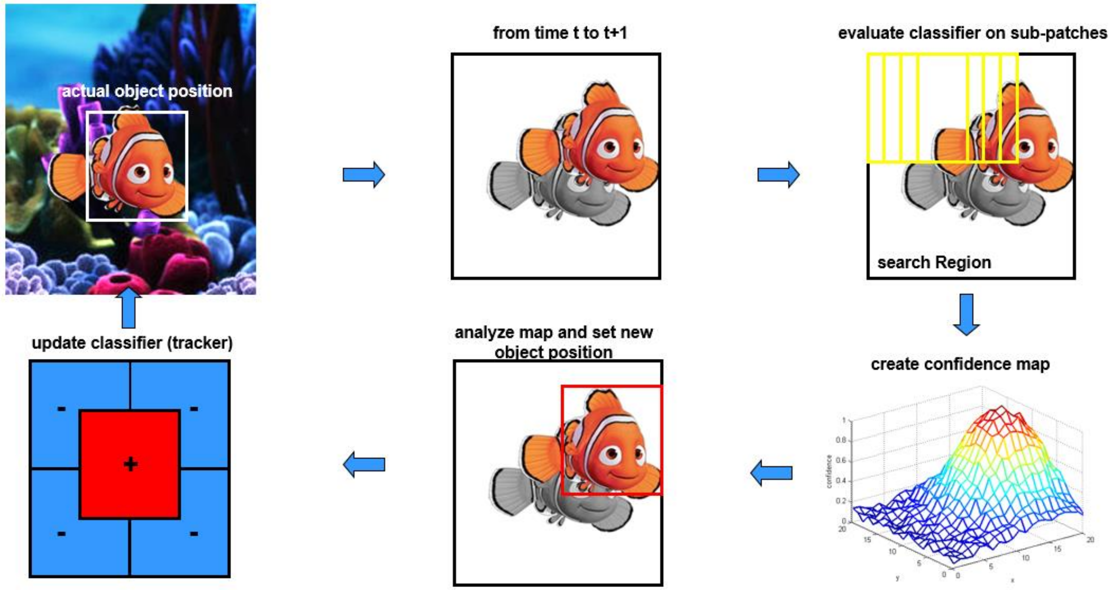
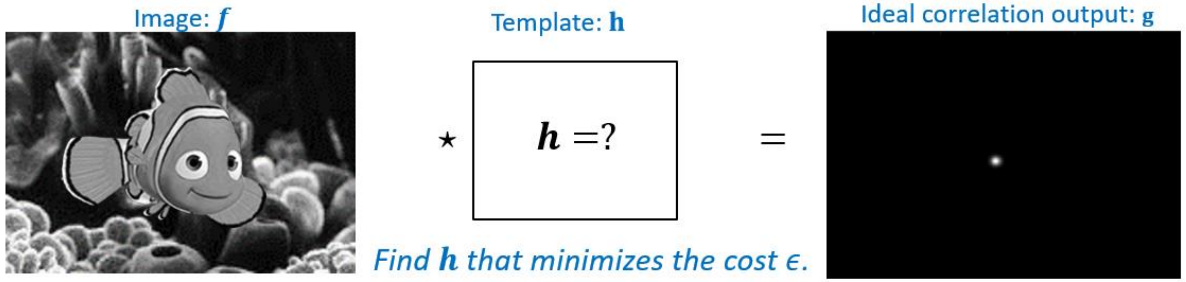
Previously at ACVM…

- Discriminative trackers
  - Adaboost
  - TMIL
  - Structured SVM



- Discriminative correlation filters
  - Linear classifiers (ridge regression learning)
  - Efficient computation via FFT



Univerza v Ljubljani

Advanced computer vision methods
Tracking by Recursive Bayes Filters
Part I: Introduction

Matej Kristan

Laboratorij za Umetne Vizualne Spoznavne Sisteme,
Fakulteta za računalništvo in informatiko,
Univerza v Ljubljani

Classes of trackers

- A tracker can be roughly classified by considering the following two properties:

Property 1: Batch tracking *vs.* Online tracking

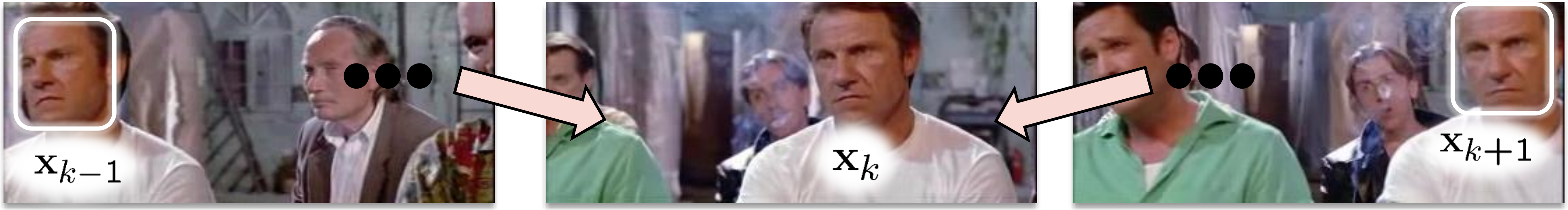
- How many images are considered to estimate the state at time-step t ?

Property 2: Non Bayesian *vs.* Bayesian tracking

- How is the notion of the target state encoded?

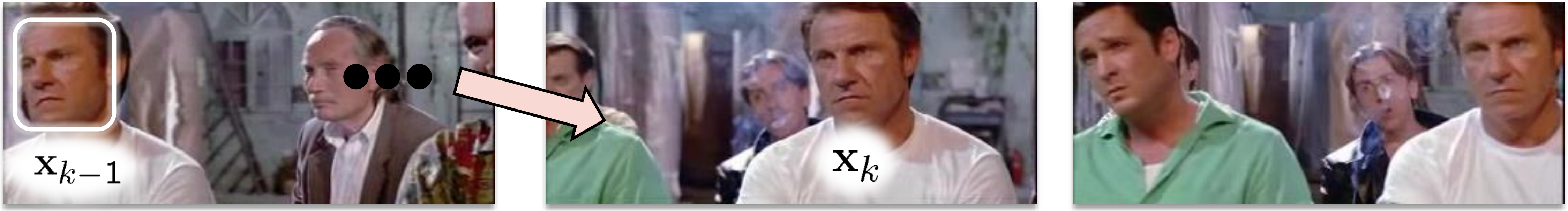
Online vs Batch tracking

- Batch tracking: Can consider all frames before t and after t to infer the target position at time-step t.



Potentially robust, appropriate for offline systems

- Online tracking: Can consider only frames before t to infer position at t.



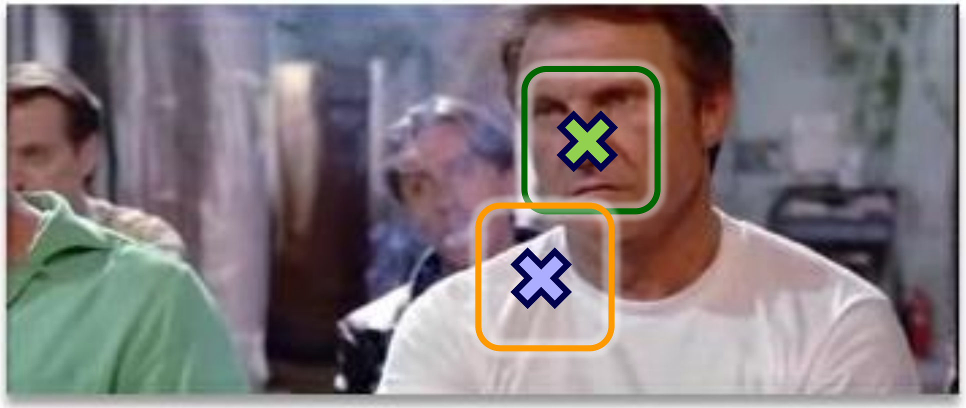
Potentially fast, appropriate for real-time systems

Non Bayesian vs Bayesian tracking

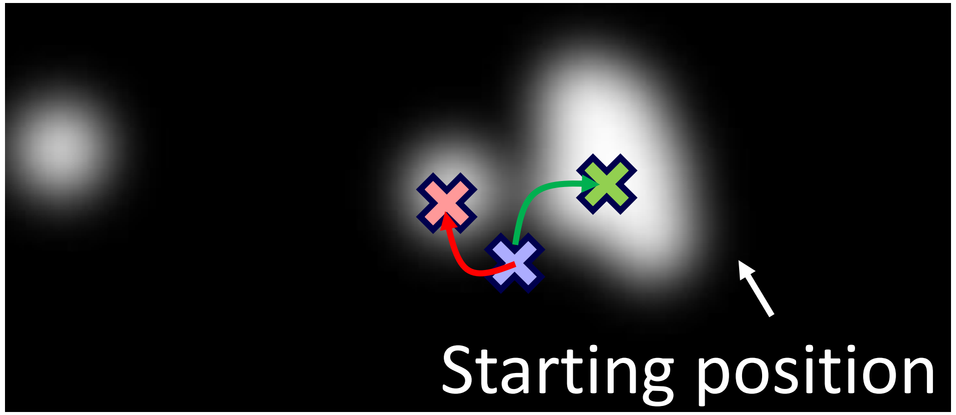
Question: "How is the information of the target state encoded?"

NON-BAYESIAN

- Local optimum
  - Gradient descent
  - Mean Shift
  - Greedy local search, etc.
- Fast convergence
- Single solution for the state value
  - But is it correct?



input image



similarity/probability

Non Bayesian vs Bayesian tracking

Question: "How is the notion of the target state encoded?"

BAYESIAN

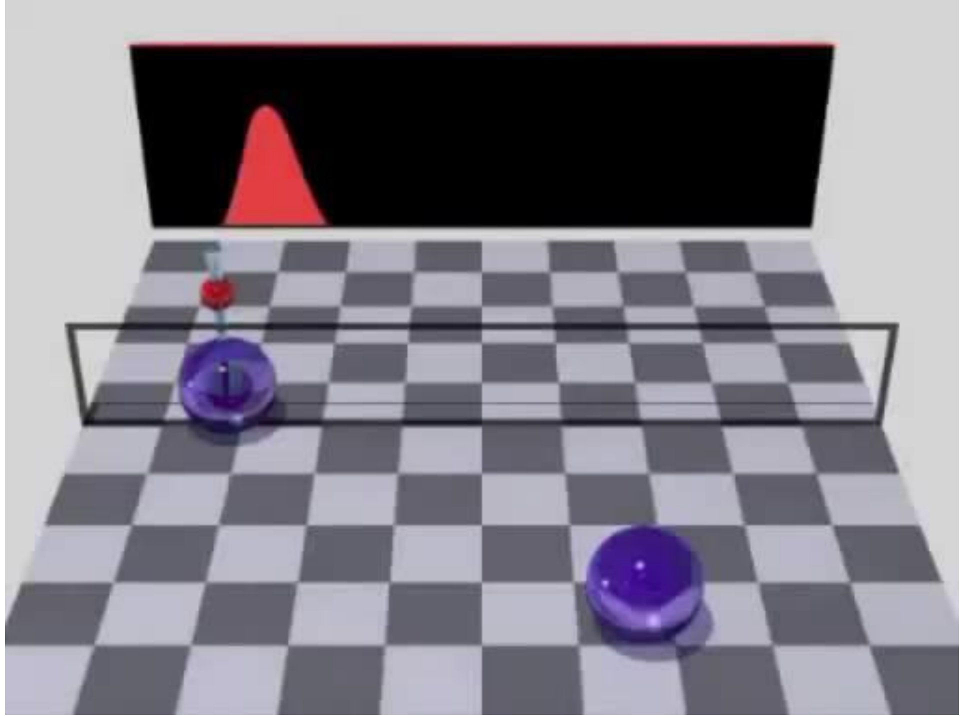
- Assign a probability to each position of the target
- Relevant info is encoded in the pdf over the target state!
- Implicitly remembers multiple hypotheses of "location".
- Interpret the pdf when required
- Typically slower

Non Bayesian vs Bayesian tracking

Question: "How is the notion of the target state encoded?"

BAYESIAN

- The pdf changes with time – an entire pdf is tracked
- Example of a pdf:
  - $p(Ball|x_k)$,
  - Expected value:
  - $\hat{x} = \langle x_k \rangle_{p(Ball|x_k)}$



Examples: Online tracking

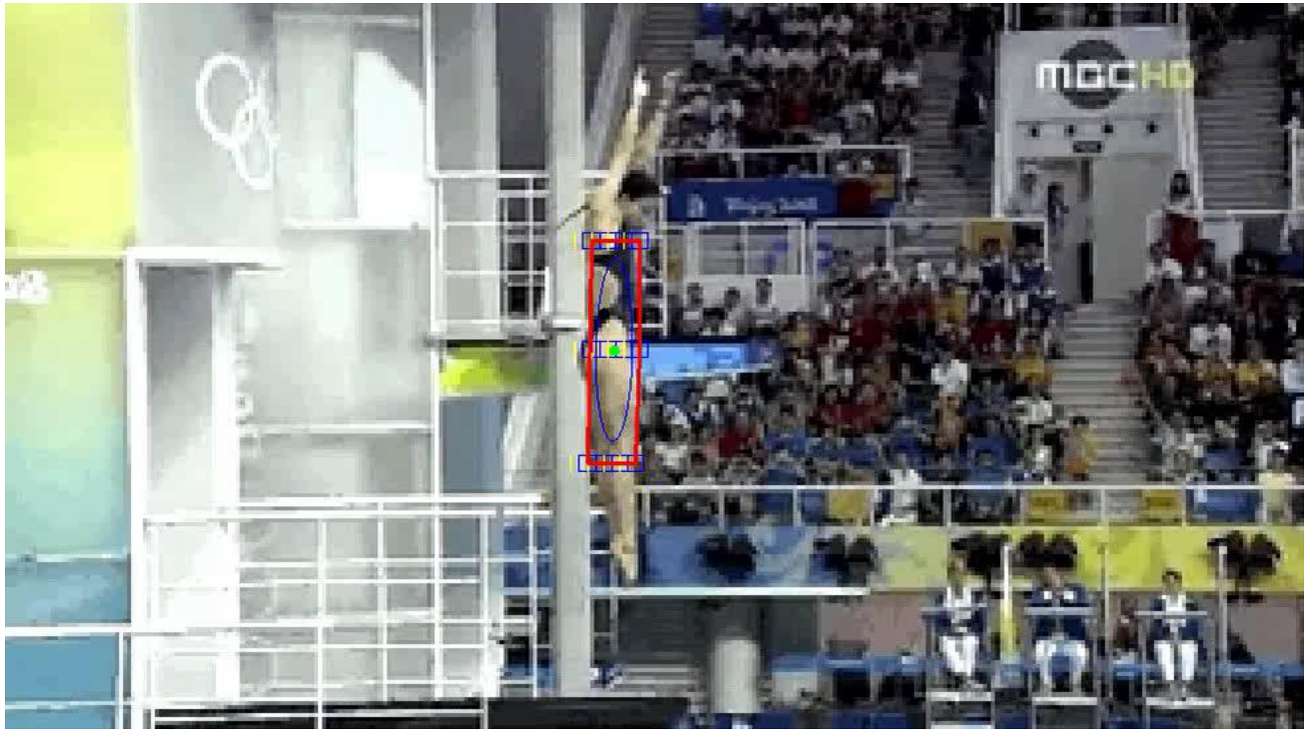
- Non Bayesian
  - Mean Shift

Comaniciu et al. “Kernel-Based Object Tracking”, IEEE TPAMI., 2003

- Fully Bayesian
  - Bayes recursive filters

Isard et al., “CONDENSATION -- conditional density propagation for visual tracking” IJCV, 1998

Practical challenges



- Change in appearance
- Level of detail
- Occlusion by visually similar objects
- Clutter
- Target motion
- Interacting targets

Čehovin, Kristan and Leonardis,"An adaptive coupled-layer visual model for robust visual tracking", ICCV2011

Practical challenges

- Change in appearance
- Level of detail
- Occlusion by visually similar objects
- Clutter
- Target motion
- Interacting targets

Kristan et. al,"Closed-world tracking of multiple interacting targets for indoor-sports applications“, CVIU2009

Practical challenges

- Change in appearance
- Level of detail
- Occlusion by visually similar objects
- Clutter
- Target motion
- Interacting targets

M. Kristan et al., "A Local-motion-based probabilistic model for visual tracking“, *Pattern Recognition*, 2009

Practical challenges

- Change in appearance
- Level of detail
- Occlusion by visually similar objects
- Clutter
- Target motion
- Interacting targets

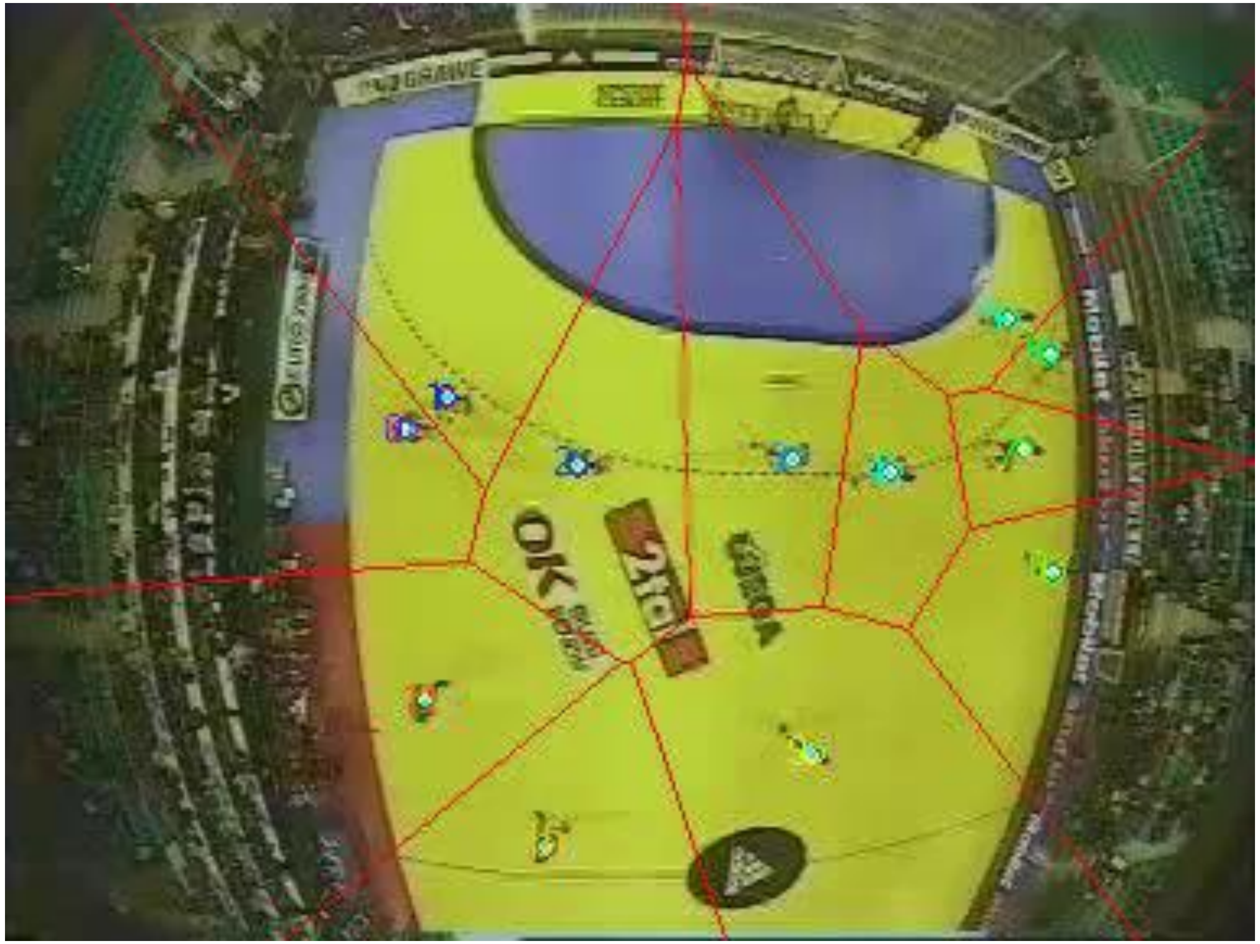
Z. Khan, T. Balch, and F. Dellaert, "MCMC-Based Particle Filtering for Tracking a Variable Number of Interacting Targets" IEEE TPAMI, 2005.

Practical challenges

- Change in appearance
- Level of detail
- Occlusion by visually similar objects
- Clutter
- Target motion
- Interacting targets

Isard, M. and Blake, A. "CONDENSATION --conditional density propagation for visual tracking", IJCV1998

Practical challenges

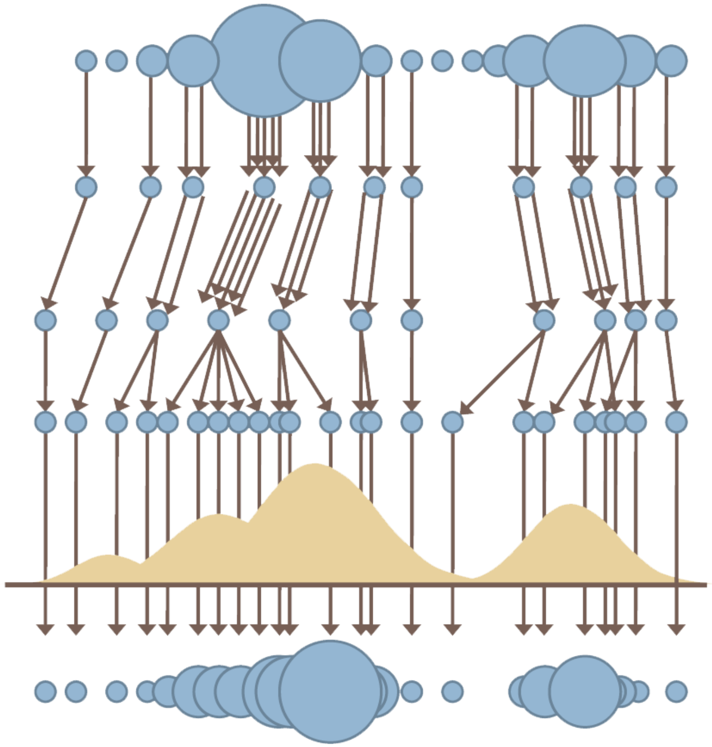


- Change in appearance
- Level of detail
- Occlusion by visually similar objects
- Clutter
- Target motion
- Interacting targets

Kristan et al.,"Closed-world tracking of multiple interacting targets for indoor-sports applications"*CVIU* 2009.

Practical challenges

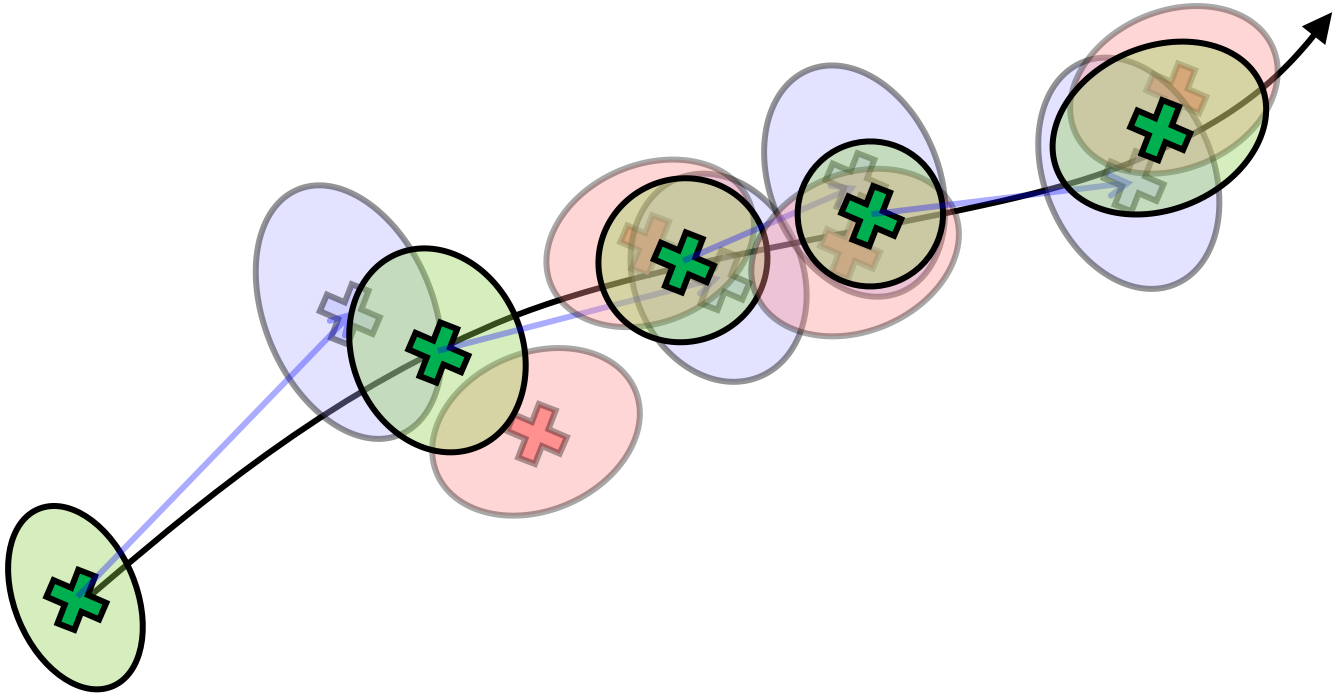
These issues can be addressed efficiently using probabilistic approaches!

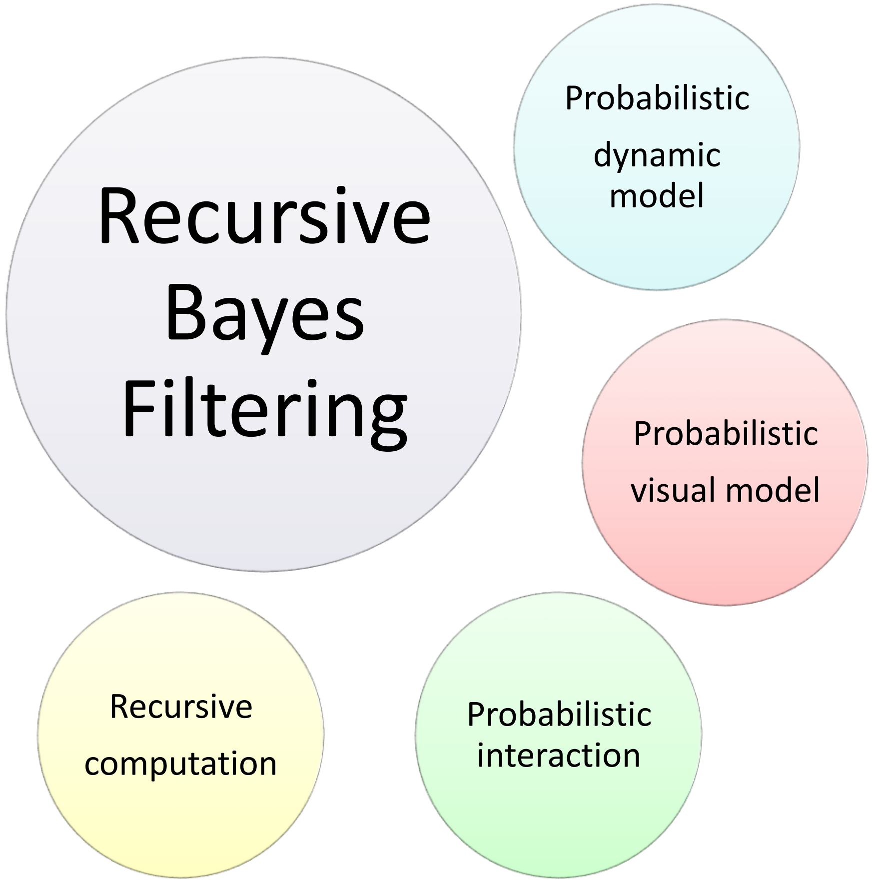


- Change in appearance
- Level of detail
- Occlusion by visually similar objects
- Clutter
- Target motion
- Interacting targets

Recursive Bayes Filters

- A principled way to address uncertainty in visual tracking





Consider tracking an airplane as an example

Observe a scene at *t-1*

Observe a scene at *t*

Observed scene at *t*

Bayesian tracking as a state estimation problem

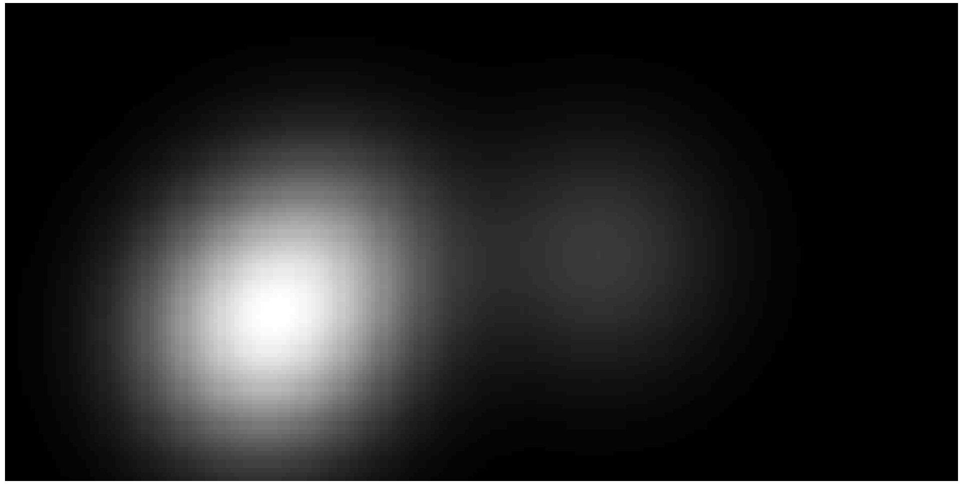
- State at time t: x_t (e.g., position)
- Measurement at t: y_t (e.g., location obtained by detector)

- Approach: Given all we know about the target and the measurements we take, what is the probability that a target is at state x_t?

From this:

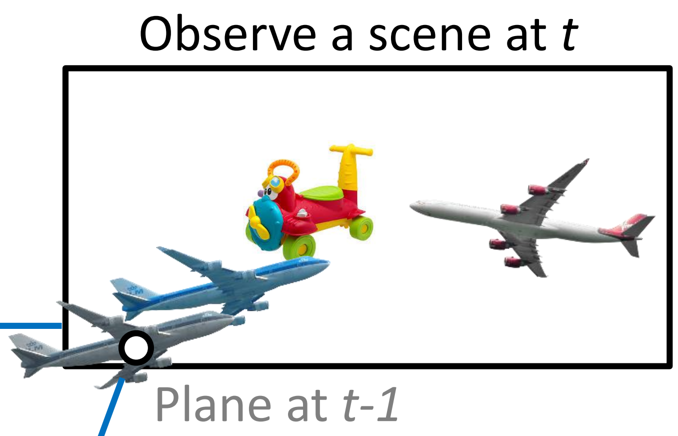
Observed scene at t

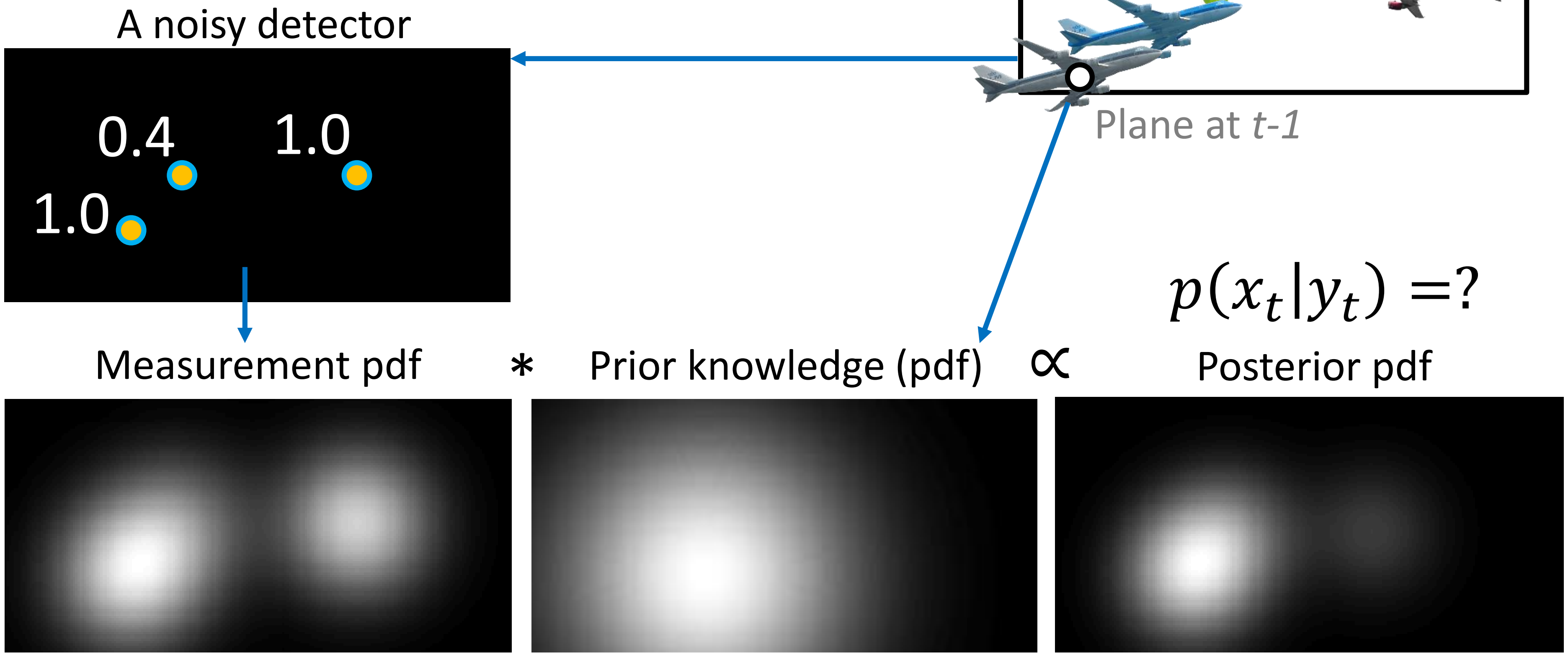
Infer this: $p(x_t | y_t) = ?$



What is a Bayes Filter?

- Key idea 1: Reason about the target states in terms of pdfs
  - Detector is uncertain
  - Can consider each “detection”,e.g., as a center of a Gaussian
  - Gives a measurement pdf

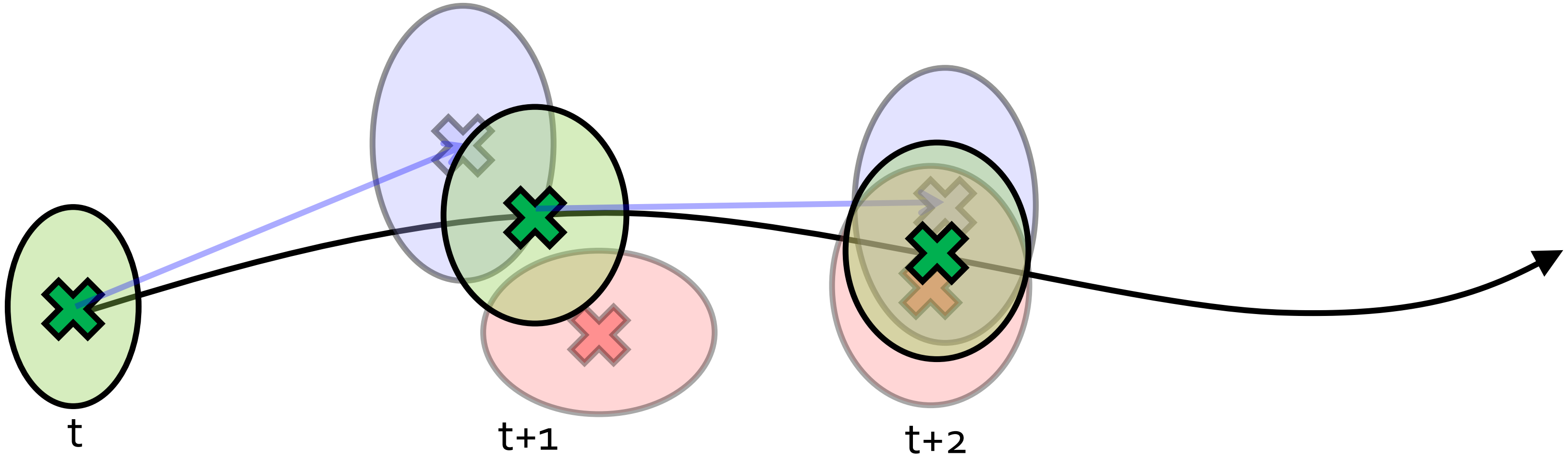




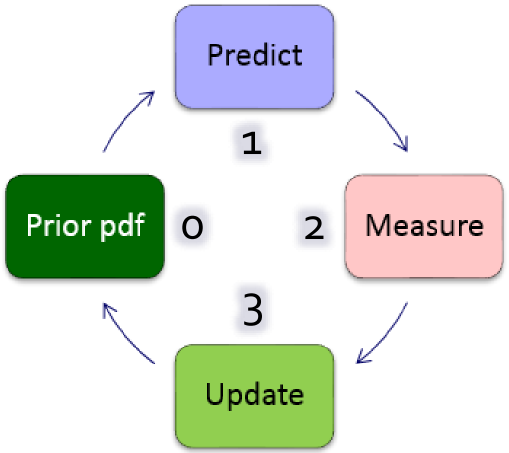
$$p(x_t|y_t) =?$$

What is a *Recursive* Bayes Filter?

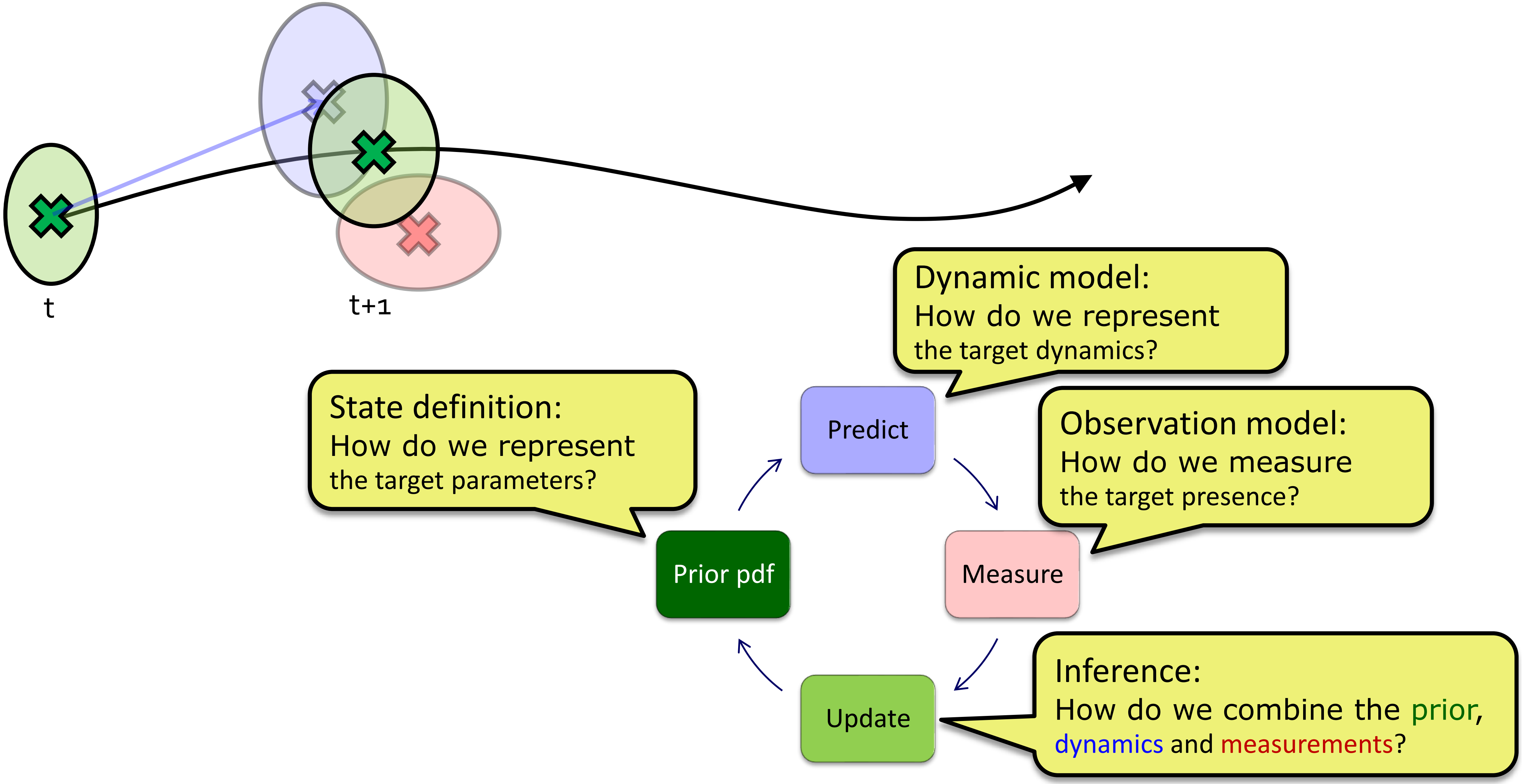
- Key idea 1: Encode beliefs about states in a pdf



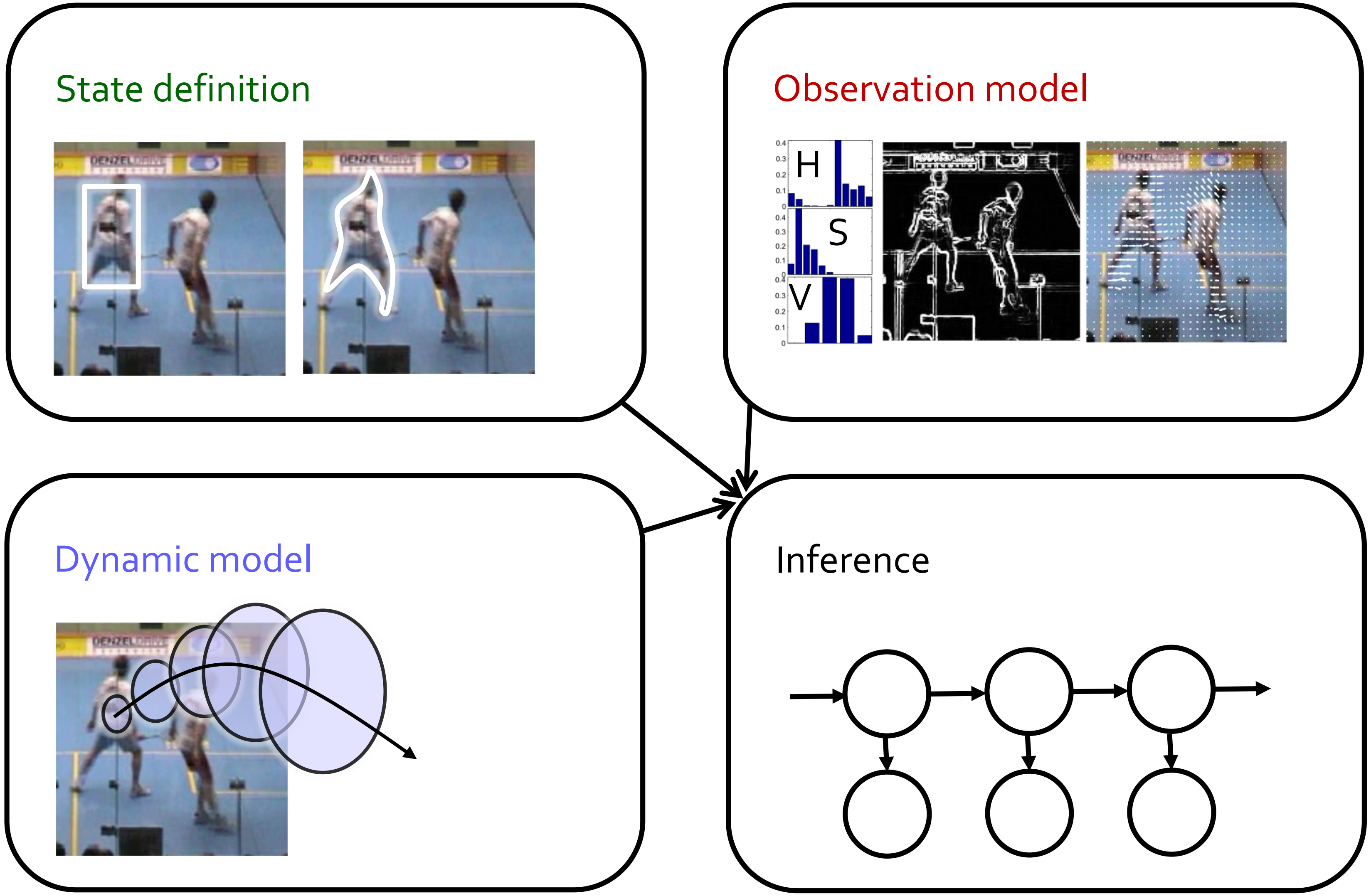
- Key idea 2: Recursively estimate the posterior
  - Predict from *uncertain* motion model
  - Measure from *uncertain* sensor
  - Update distribution



Recursive Bayes Filter: Key ingredients



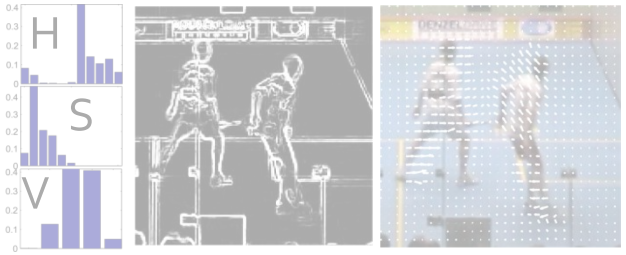
Key Ingredients



Key Ingredients

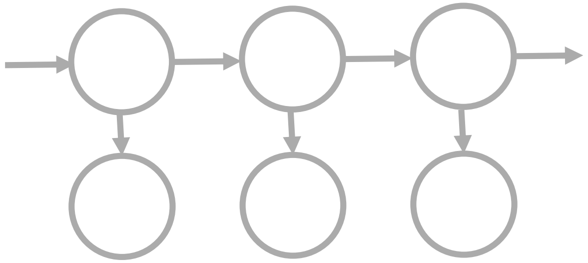
State definition

Observation model



Dynamic model

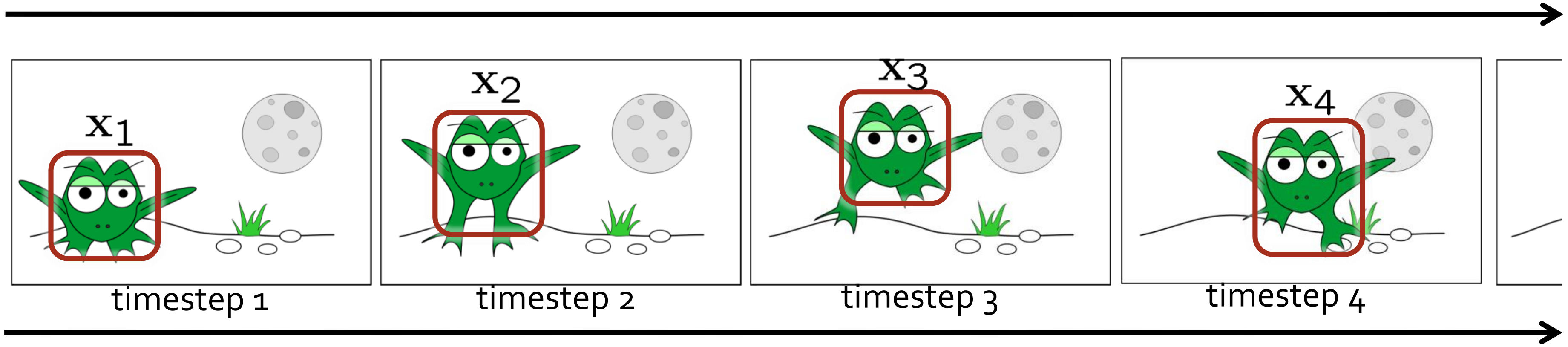
Inference



What is a state of the target?

- Target properties at a time-step
- Encodes parameters (which we want to estimate)

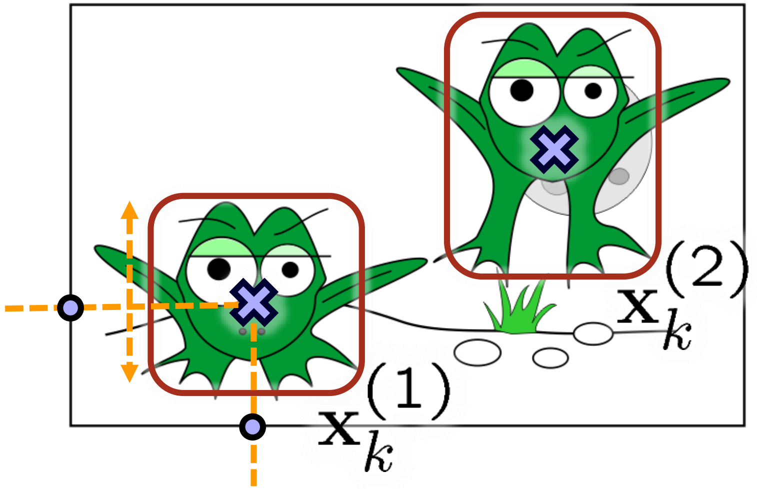
$$\mathbf{x}_{1:N} = \{\mathbf{x}_1, \ldots, \mathbf{x}_{k-1}, \mathbf{x}_k, \mathbf{x}_{k-1}, \ldots, \mathbf{x}_N\} = \{\mathbf{x}_k\}_{1:N}$$

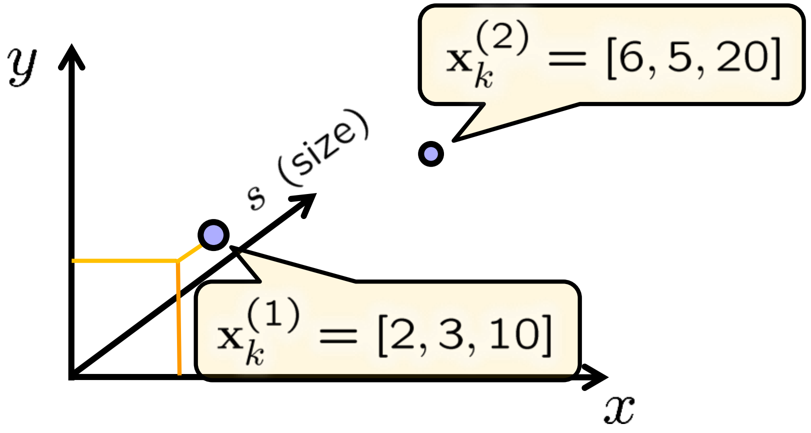


- Parametric form of the state $\boldsymbol{x}_k$ depends on the model by which we describe the target.

How do we define a state?

- Define the state by the target "free" parameters.
- Examples:
  - Location
    $$\mathbf{x}_k = [x, y]$$
  - Location + size
    $$\mathbf{x}_k = [x, y, s]$$
  - Location + velocity
    $$\mathbf{x}_k = [x, y, \dot{x}, \dot{y}]$$
  - Multiple objects (joint state)
    $$\mathbf{x}_k = \{\mathbf{x}_k^{(1)}, \mathbf{x}_k^{(2)}\}$$

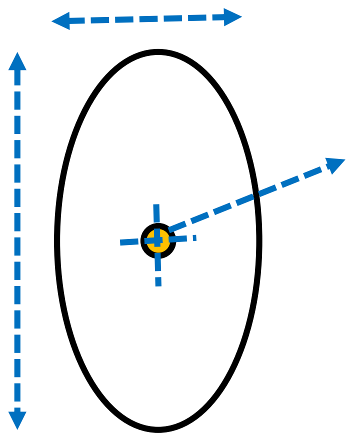




State definition: Example 1

- Axis-aligned blobs (bounding box, ellipse)
  - center
  - width + height
  - velocity

6D $\mathbf{x}_k = [x, y, \dot{x}, \dot{y}, H_x, H_y]$

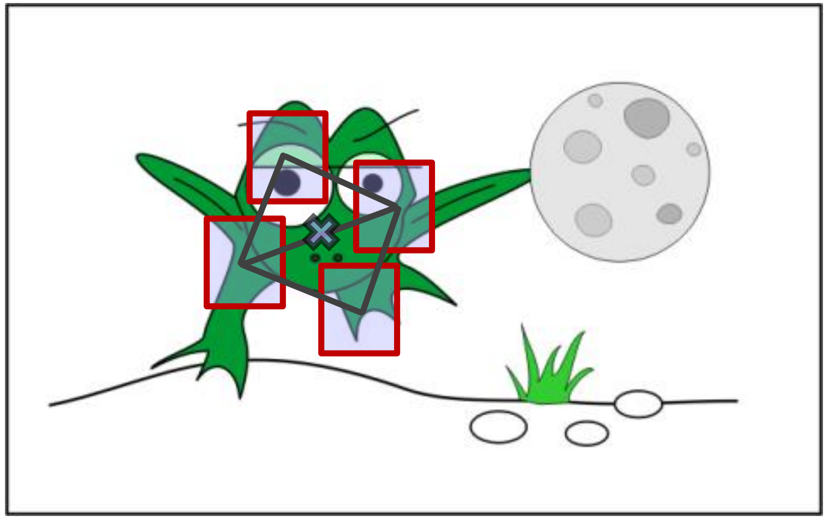


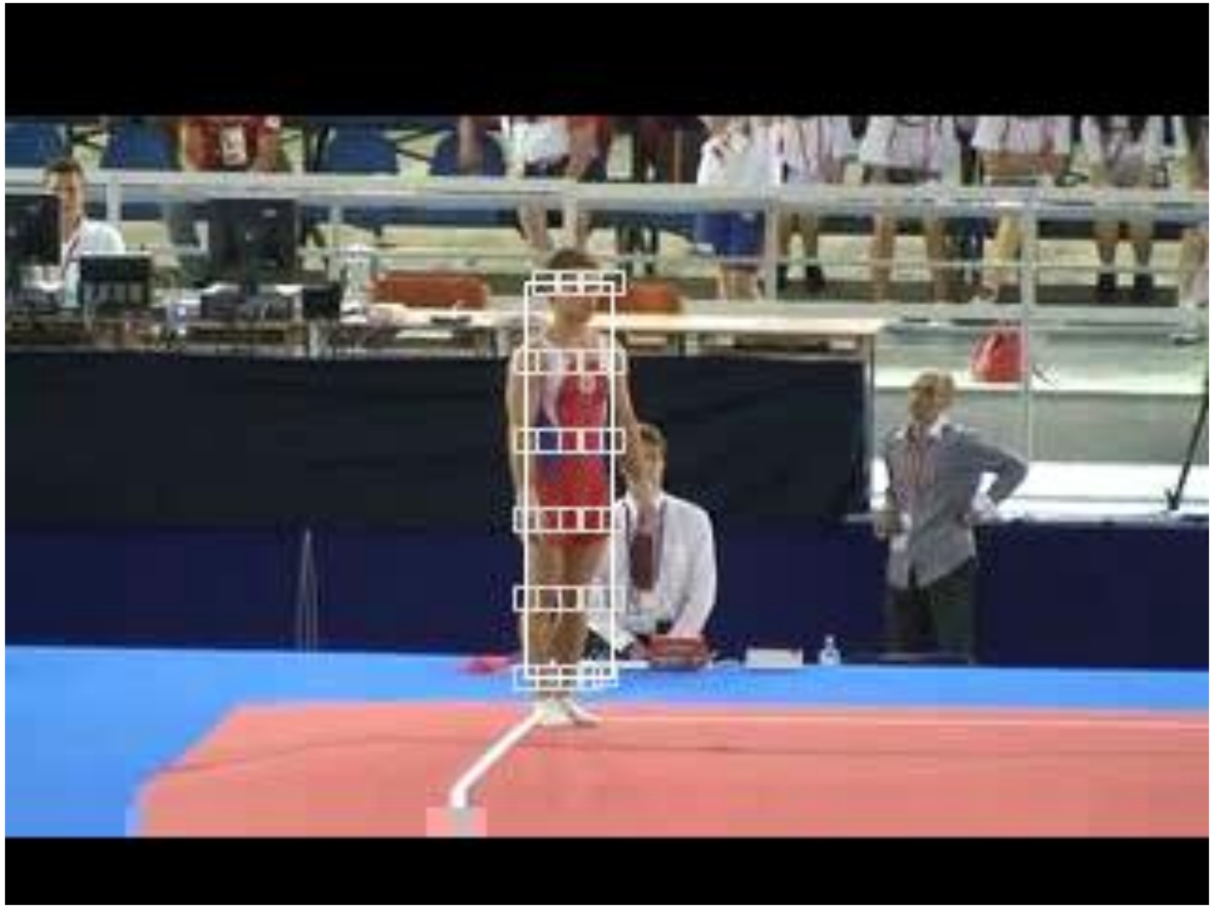
Kristan et al., "A Local-motion-based probabilistic model for visual tracking". *Pattern Recognition*, 2009.

State definition: Example 2

- Part-based models, Constellation models
  - Center, velocity
  - Relative part locations
  - Varying number of parts

$$\mathbf{x}_k = [x_C, y_C, \dot{x}_C, \dot{y}_C, \mathbf{x}_k^{(1)}, \mathbf{v}_k^{(1)}, \ldots, \mathbf{x}_k^{(N_k)}, \mathbf{v}_k^{(N_k)}]$$

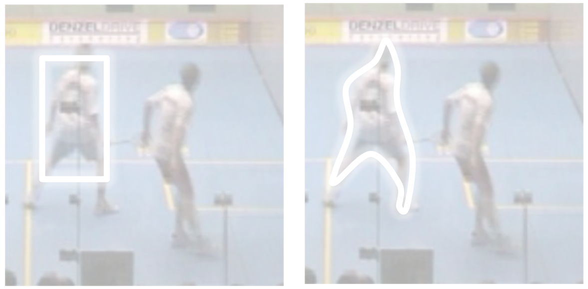




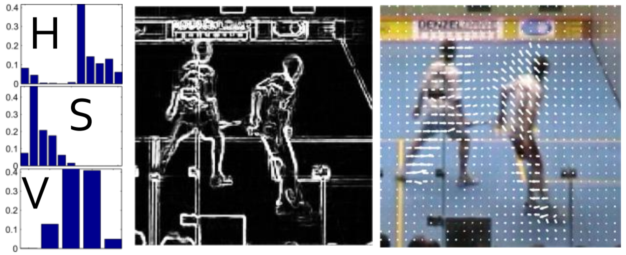
Čehovin, Kristan and Leonardis,"An adaptive coupled-layer visual model for robust visual tracking", TPAMI2013

Key Ingredients

State definition

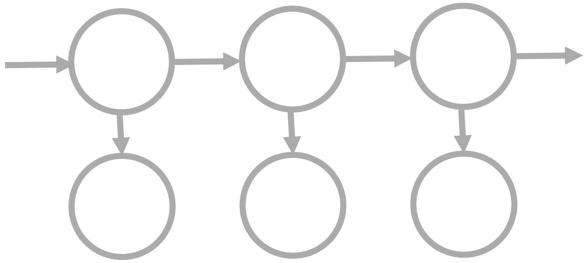


Observation model



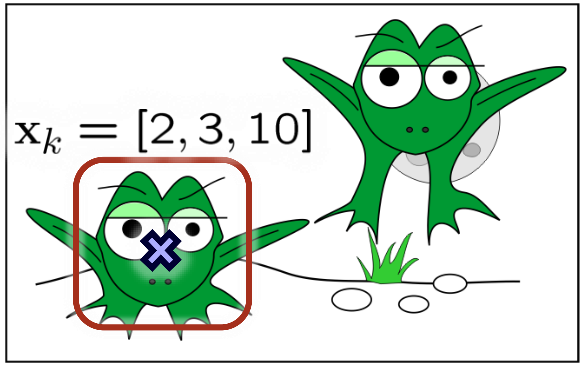
Dynamic model

Inference



What is the observation model?

- Transforms measurement into a probability
- The likelihood of observing y_k assuming the target is located at state x_k: $p(\mathbf{y}_k|\mathbf{x}_k)$

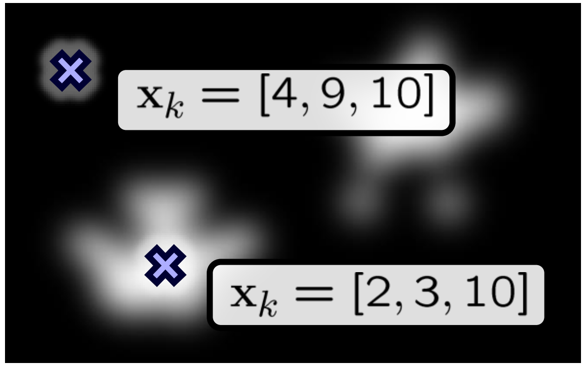


observed image

Low likelihood.

High likelihood.

$p(\mathbf{y}_k|\mathbf{x}_k)$



likelihood map

If the observation model was evaluated at each possible state x_k…

Observation model

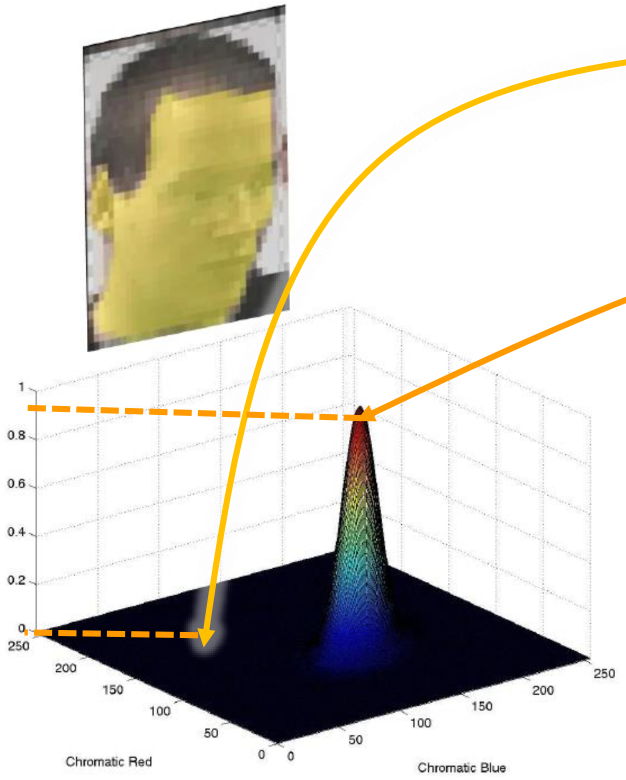
1. Choose a visual model
   (e.g., histograms, HOG, template, …)

2. Define similarity function with the visual model

3. Define a function that maps similarity to probability
   (i.e., zero similarity -> zero probability and vice versa)

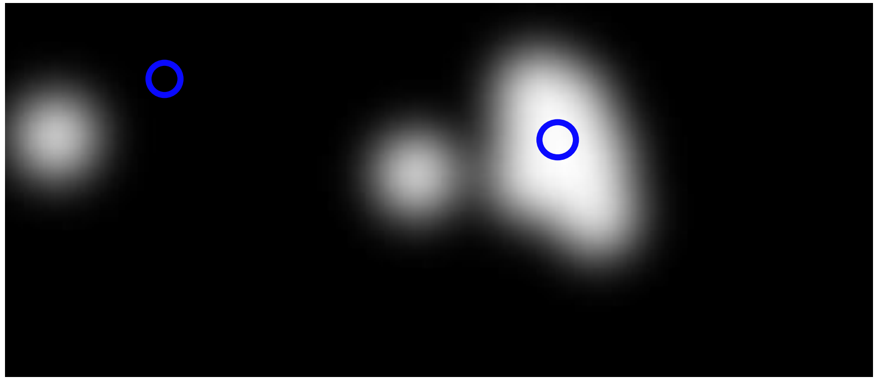
Observation model: Example 1

- Skin color sampled from a region
  - clusters in chromatic space – model by a Gaussian $p(y_k \mid x_k) \propto \exp(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu))$



input image

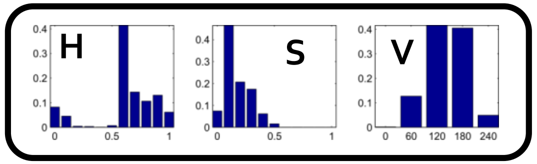
similarity/probability



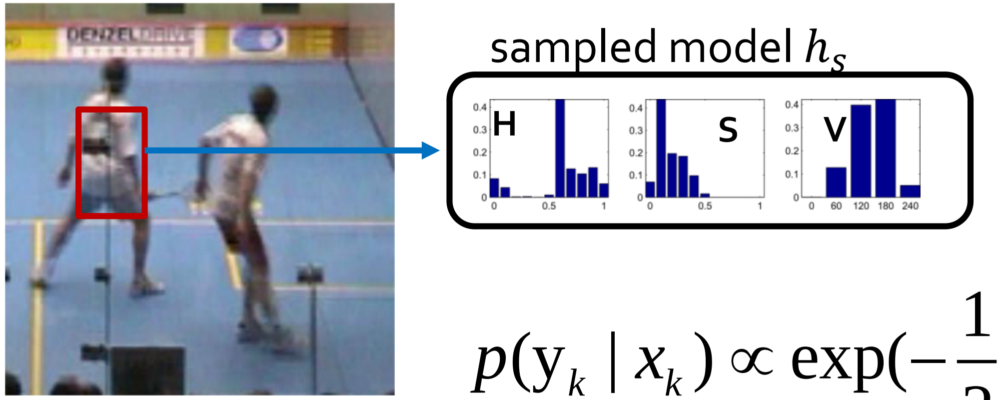
Observation model: Example 2

- Histograms
  - Color histograms
  - Hellinger distance between reference h_r and sampled histogram h_s: $d_{Hell}(h_r, h_s)$

reference model h_r



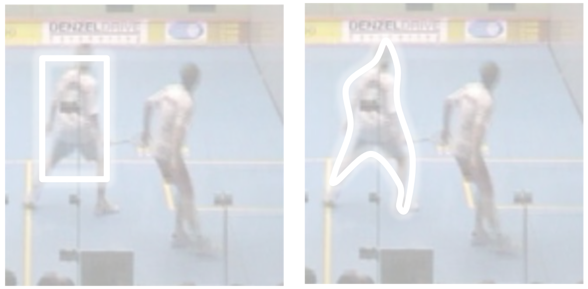
sampled model h_s



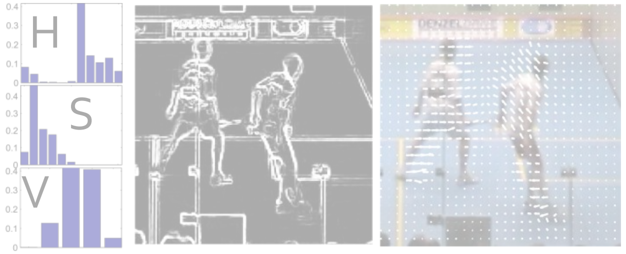
$$p(y_k \mid x_k) \propto \exp(-\frac{1}{2} d_{Hell}^2 / \sigma^2)$$

Key Ingredients

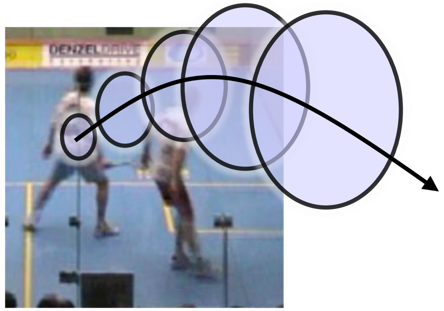
State definition



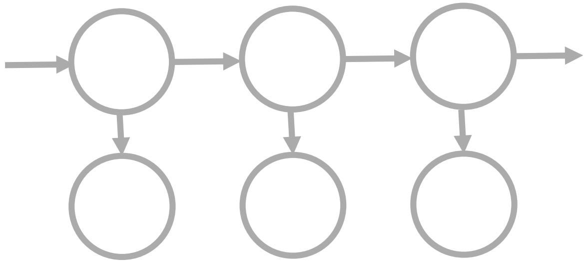
Observation model



Dynamic model

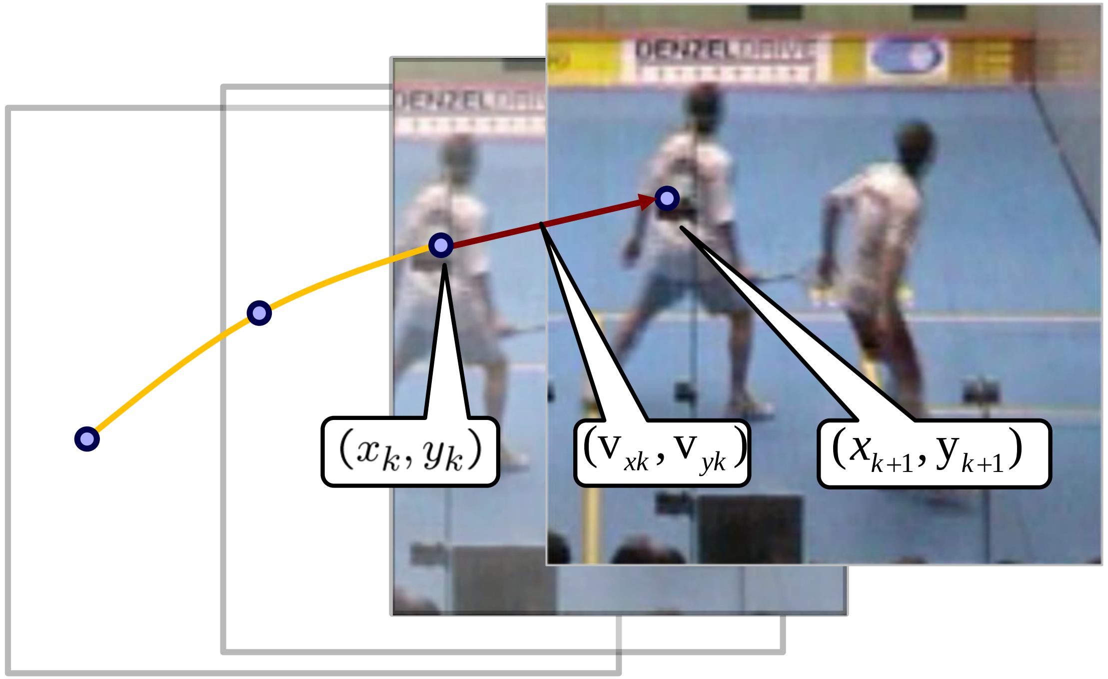


Inference



What is a dynamic model?

- Predicts the target state from its previous estimate.



$$x_{k+1} = x_k + v_{xk}\Delta t$$
$$y_{k+1} = y_k + v_{yk}\Delta t$$

- This is an example of a constant-velocity model
- Assumption: velocity at *k+1* is equal to velocity at *k*.

A constant velocity model

- 1D problem, but 2D state space with position and velocity

$$\mathbf{x} = \begin{bmatrix} x \\ \dot{x} \end{bmatrix} \longrightarrow \dot{\mathbf{x}} = F\mathbf{x} \qquad F = ?$$

- Velocity does not change:

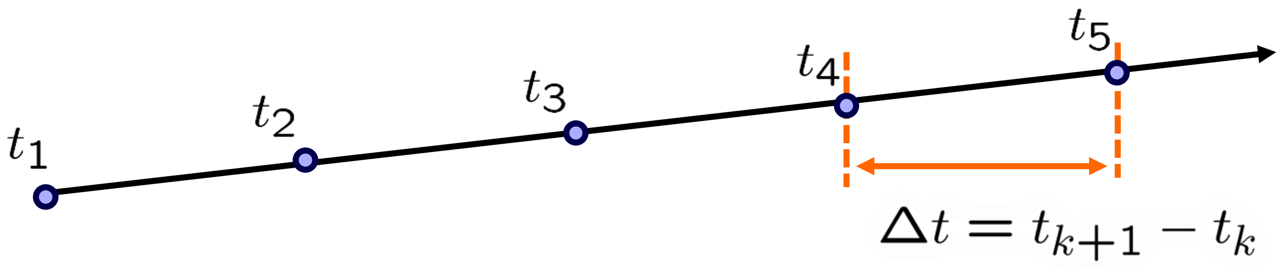
$$\dot{\mathbf{x}} = F\mathbf{x}$$

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x} \\ \ddot{x} \end{bmatrix} = \begin{bmatrix} & ? & \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \end{bmatrix}$$

From continuous to discrete

- A continuous motion is sampled at equally spaced time-steps (spacing Δt):

$$\dot{\mathbf{x}} = F\mathbf{x}$$



- Solution according to Stengel (p.84)

$$\mathbf{x}(t_k) = \mathbf{\Phi}(\Delta t)\mathbf{x}(\mathrm{t}_{k-1})$$

$$\mathbf{\Phi}(\Delta t) = \mathrm{e}^{F\Delta t} = I + F\Delta t + \tfrac{1}{2!}F^2\Delta t^2 + \tfrac{1}{3!}F^3\Delta t^3 + \cdots$$

Robert F. Stengel, *Optimal Control and Estimation*, Dover Books on Mathematics, 1994

From continuous to discrete

- For the constant-velocity model:

$$\dot{\mathbf{x}} = F\mathbf{x} \qquad F = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$$

Compute using your favorite symbolic toolbox:

$$\mathbf{\Phi}(\Delta t) = \mathrm{e}^{F\Delta t}$$

$$\mathbf{\Phi}(\Delta t) = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}$$

Matlab symbolic toolbox:

```
>> syms T
>> F = [0 1; 0 0]
>> Fi = expm(F*T)
```

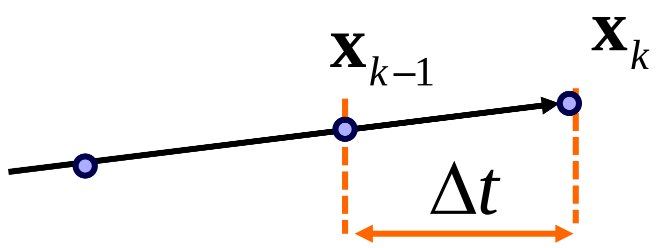
Python symbolic toolbox:

```
>> import sympy as sp
>> from sympy.interactive.printing import init_printing
>> init_printing(use_unicode=False, wrap_line=False)
T = sp.symbols('T')
>> F = sp.Matrix([[0, 1],[0, 0]])
>> Fi = sp.exp(F*T)
```

Discrete constant velocity model

- See if the derived CV model makes any sense:

$$\mathbf{x}_k = \mathbf{\Phi}(\Delta t)\mathbf{x}_{k-1} \quad , \quad \mathbf{\Phi}(\Delta t) = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \quad , \quad \mathbf{x}_k = \begin{bmatrix} x_k \\ \dot{x}_k \end{bmatrix}$$



$$\begin{bmatrix} x_k \\ \dot{x}_k \end{bmatrix} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_{k-1} \\ \dot{x}_{k-1} \end{bmatrix} \longrightarrow \begin{aligned} x_k &= x_{k-1} + \dot{x}_{k-1}\Delta t \\ \dot{x}_k &= \dot{x}_{k-1} \end{aligned}$$

But constant velocity is not a very realistic assumption…

A nearly-constant-velocity model

- Assume that acceleration is not zero, but is noisy:

$$\ddot{x} = w$$

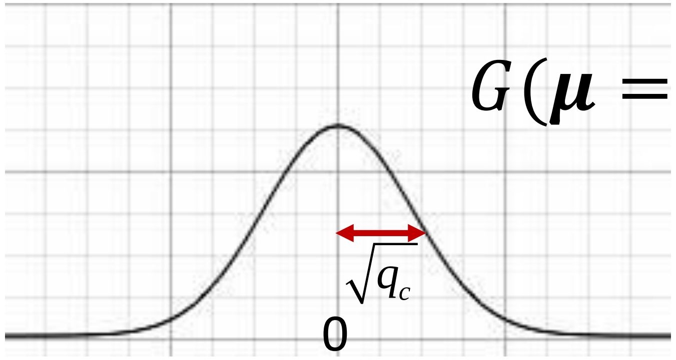
$$\mathbf{x} = \begin{bmatrix} x \\ \dot{x} \end{bmatrix} \qquad \dot{\mathbf{x}} = \begin{bmatrix} \dot{x} \\ \ddot{x} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \end{bmatrix} + \begin{bmatrix} ? \end{bmatrix} w$$

$$\dot{\mathbf{x}} = F\mathbf{x} + Lw$$

Stochastic– *a white noise sequence* specified by its covariance (spectral density) q_c!

Deterministic

$G(\boldsymbol{\mu} = 0, q_c)$



A discrete counterpart

- Solution of $\dot{\mathbf{x}} = F\mathbf{x} + Lw$ according to Stengel (p.84)

$$\mathbf{x}_k = \boldsymbol{\Phi}(\Delta t)\mathbf{x}_{k-1} + W_{k-1} \quad , \qquad \boldsymbol{\Phi}(\Delta t) = \mathrm{e}^{F\Delta t}$$ (deterministic)

W_{k-1} is a random variable: $W_{k-1} = \int_{t_{k-1}}^{t_k} \Phi(\tau)\,\mathrm{Lw}(\tau)\,\mathrm{d}\tau$

Governed by a pdf and specified by the covariance matrix:

$$Q_{k-1} = \int_0^{\Delta t} (\Phi(\xi)\,\mathrm{L})\,\mathrm{q}_c\,(\Phi(\xi)\,\mathrm{L})^T\,d\xi$$

Might want to apply Matlab/Python/Mathematica to solve for Q_{k-1}*…*

Robert F. Stengel, *Optimal Control and Estimation*, Dover Books on Mathematics, 1994

The covariance Q of a NCV

- Recall:

$$\mathbf{\Phi}(\Delta t) = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \qquad \dot{\mathbf{x}} = \begin{bmatrix} \dot{x} \\ \ddot{x} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} w$$

$$Q_{k-1} = \int_0^{\Delta t} (\Phi(\xi)\,\mathrm{L})\,\mathrm{q}_c\,(\Phi(\xi)\,\mathrm{L})^T\, d\xi \qquad Q = q \begin{bmatrix} \frac{1}{3}\Delta t^3 & \frac{1}{2}\Delta t^2 \\ \frac{1}{2}\Delta t^2 & \Delta t \end{bmatrix}$$

Matlab symbolic toolbox:

```
>>syms T q
>> Fi = [1 T;0 1]
>> L=[0 ;1 ]
>> Q=int((Fi*L)*q*(Fi*L)',T,0,T)
```

Python symbolic toolbox:

```
>> import sympy as sp
>> T, q = sp.symbols('T q')
>> Fi = sp.Matrix([[1, T],[0, 1]])
>> L = sp.Matrix([[0], [1]])
>>Q = sp.integrate( (Fi*L)*q*(Fi*L).T, (T, 0, T) )
```

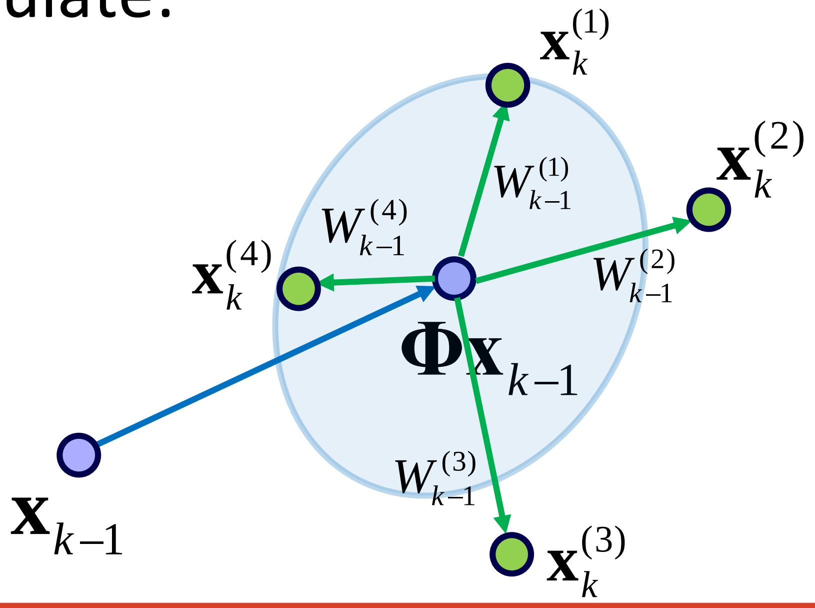
The nearly-constant-velocity model

- We are done: $\mathbf{x}_k = \mathbf{\Phi x}_{k-1} + W_{k-1}$

$$W_{k-1} \sim G(\mu = 0, Q)$$

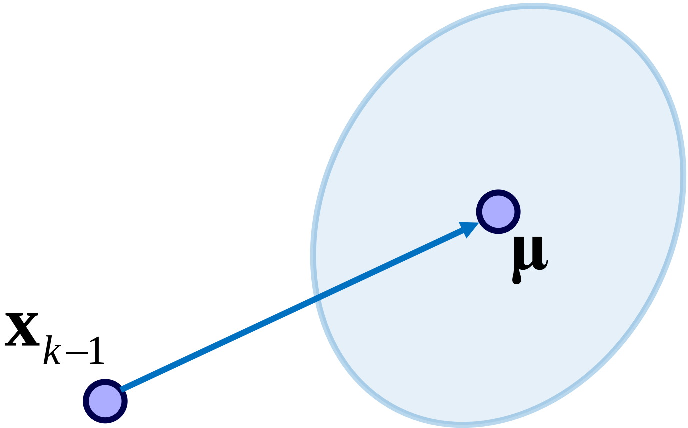
$$\mathbf{\Phi} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \qquad Q = q\begin{bmatrix} \frac{1}{3}\Delta t^3 & \frac{1}{2}\Delta t^2 \\ \frac{1}{2}\Delta t^2 & \Delta t \end{bmatrix}$$

Let's simulate:



Probabilistic model:

$$p(\mathbf{x}_k \mid \mathbf{x}_{k-1}) = \mathrm{G}(\boldsymbol{\mu} = \Phi \mathbf{x}_{k-1}, \mathrm{Q})$$



It is easy to extend to 2D or higher

- A 2D NCV example:
  - If you like compact derivation…

$$\mathbf{x} = \begin{bmatrix} x \\ \dot{x} \\ y \\ \dot{y} \end{bmatrix} \qquad \dot{\mathbf{x}} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ y \\ \dot{y} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$$

Will be part of the assignment…

- Or, more common and simpler (but equivalent):
  - Two separate instances of the NCV model that we derived in previous slides (one for x and one for y).

Random walk dynamic model

- Brownian motion – velocity is noise!

$$\mathbf{x} = \begin{bmatrix} x \\ \dot{x} \end{bmatrix} \qquad \dot{x} = w$$

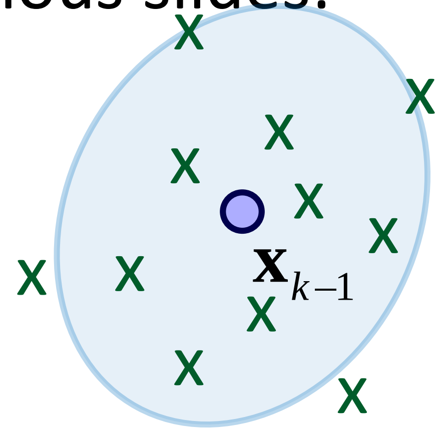
Step 1: Write the equation in a form of
$\dot{x} = Fx + Lw$

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x} \\ \ddot{x} \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} w$$

- Apply the same (Matlab/Python) derivation as in previous slides:

$$\mathbf{x}_k = \mathbf{x}_{k-1} + W_{k-1}$$

$$p(\mathbf{x}_k \mid \mathbf{x}_{k-1}) = \mathrm{G}(\boldsymbol{\mu} = \mathbf{x}_{k-1}, \mathrm{Q})$$



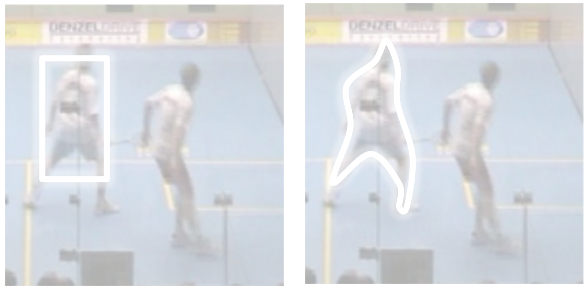
Widely used dynamic models

- Velocity is non-correlated:
  - Velocity modelled by a white noise sequence
  - Random Walk model (RW), Brownian motion
- Acceleration non-correlated:
  - Acceleration modelled by a white noise sequence
  - Nearly constant velocity (NCV)
- Derivative of acceleration (jerk) non-correlated:
  - Jerk modelled by a white noise sequence
  - Nearly constant acceleration (NCA)

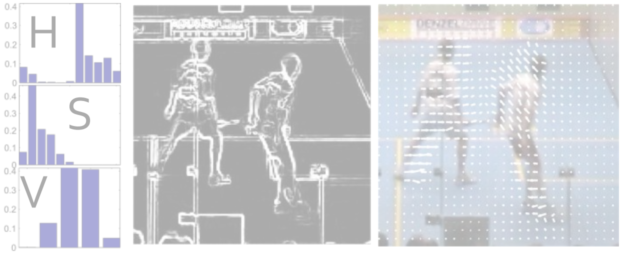
X. Rong Li, V. Jilkov P., Survey of maneuvering target tracking: Dynamic models, IEEE TAES2003

Key Ingredients

State definition

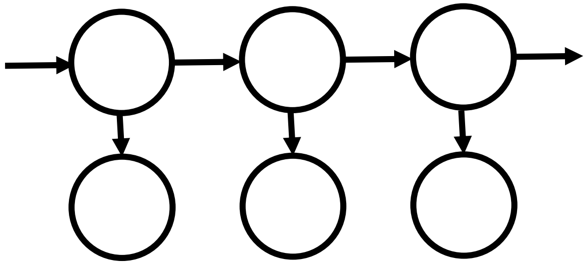


Observation model



Dynamic model

Inference



Probabilistic view

- Given a sequence of observations $\mathbf{y}_{1:k} = \{\mathbf{y}_i\}_{i=1:k}$
  (think about the observation in most abstract way – an image)

…

- …want to find the density over the current state $\mathbf{x}_k$

$$p(\mathbf{x}_k | \mathbf{y}_{1:k})$$

Using the Bayesian terminology:
The posterior over the $\boldsymbol{x}_k$

Towards Recursive Bayes Filter

- The goal is to rewrite the posterior in the current time-step *k* as a function of the posterior from the previous time-step *k-1*:

$$p(\mathrm{x}_k \mid y_{1:\mathrm{k}}) = f\,(p(\mathrm{x}_{k-1} \mid y_{1:\mathrm{k}-1}), \mathrm{y}_k)$$

Towards Recursive Bayes Filter

$$p(\mathrm{x}_k \mid y_{1:k}) = \frac{p(y_k \mid \mathrm{x}_k, y_{1:k-1})\,\mathrm{p}(\mathrm{x}_k \mid y_{1:k-1})}{p(y_k \mid y_{1:k-1})}$$

Assumption 1: *Current measurement is conditionally independent from all previous measurements given x_k.*

$$p(y_k \mid \mathrm{x}_k, y_{1:k-1}) \equiv p(y_k \mid \mathrm{x}_k)$$

$$p(\mathrm{x}_k \mid y_{1:k}) = \frac{p(y_k \mid \mathrm{x}_k)\,\mathrm{p}(\mathrm{x}_k \mid y_{1:k-1})}{p(y_k \mid y_{1:k-1})}$$

Towards Recursive Bayes Filter

- Expand the density $p(x_k \mid y_{1:k-1})$:

$$p(x_k \mid y_{1:k-1}) = \int p(x_k \mid x_{k-1}, y_{1:k-1})\, p(x_{k-1} \mid y_{1:k-1})\, dx_{k-1}$$

- Assumption 2: *Current state is conditionally independent from all previous measurements given x_{k-1}.*

$$p(x_k \mid x_{k-1}, y_{1:k-1}) \equiv p(x_k \mid x_{k-1})$$

$$p(x_k \mid y_{1:k-1}) = \int p(x_k \mid x_{k-1})\, p(x_{k-1} \mid y_{1:k-1})\, dx_{k-1}$$

The Bayes Recursive Filter

- Putting it all together:

$$p(\mathrm{x}_k \mid y_{1:\mathrm{k}}) = \frac{p(y_k \mid \mathrm{x}_k)\int \mathrm{p}(\mathrm{x}_k \mid \mathrm{x}_{k-1})\,\mathrm{p}(\mathrm{x}_{k-1} \mid y_{1:\mathrm{k}-1})\,\mathrm{dx}_{k-1}}{p(y_k \mid y_{1:\mathrm{k}-1})}$$

- The denominator does not depend on the state:

$$\underbrace{p(\mathbf{x}_k|\mathbf{y}_{1:k})}_{\substack{\text{posterior}\\ \text{at } k}} \propto \underbrace{p(\mathbf{y}_k|\mathbf{x}_k)}_{\substack{\text{likelihood}\\ \text{(observation model)}}} \int \underbrace{p(\mathbf{x}_k|\mathbf{x}_{k-1})}_{\substack{\text{motion model}\\ \text{(dynamic model)}}} \underbrace{p(\mathbf{x}_{k-1}|\mathbf{y}_{1:k-1})}_{\substack{\text{posterior}\\ \text{at } k\text{-}1}} d\mathbf{x}_{k-1}$$

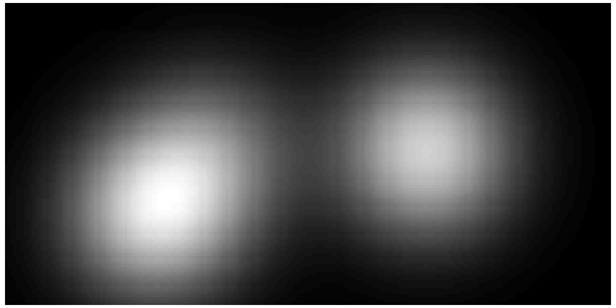
Recall the airplane tracking example

Observe a scene at *k-1*

Observe a scene at *k*

$$\int p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{x}_{k-1}|\mathbf{y}_{1:k-1})d\mathbf{x}_{k-1}$$

$p(\mathbf{y}_k|\mathbf{x}_k)$
likelihood

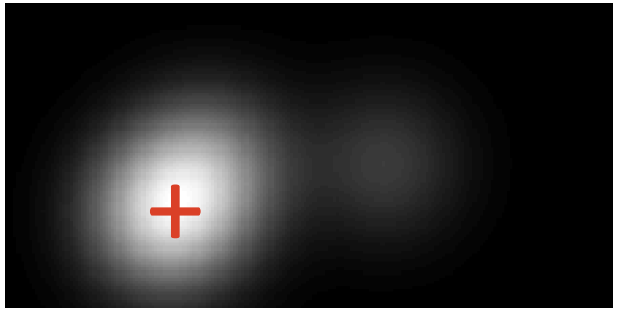


*

Predicted prior knowledge (pdf)

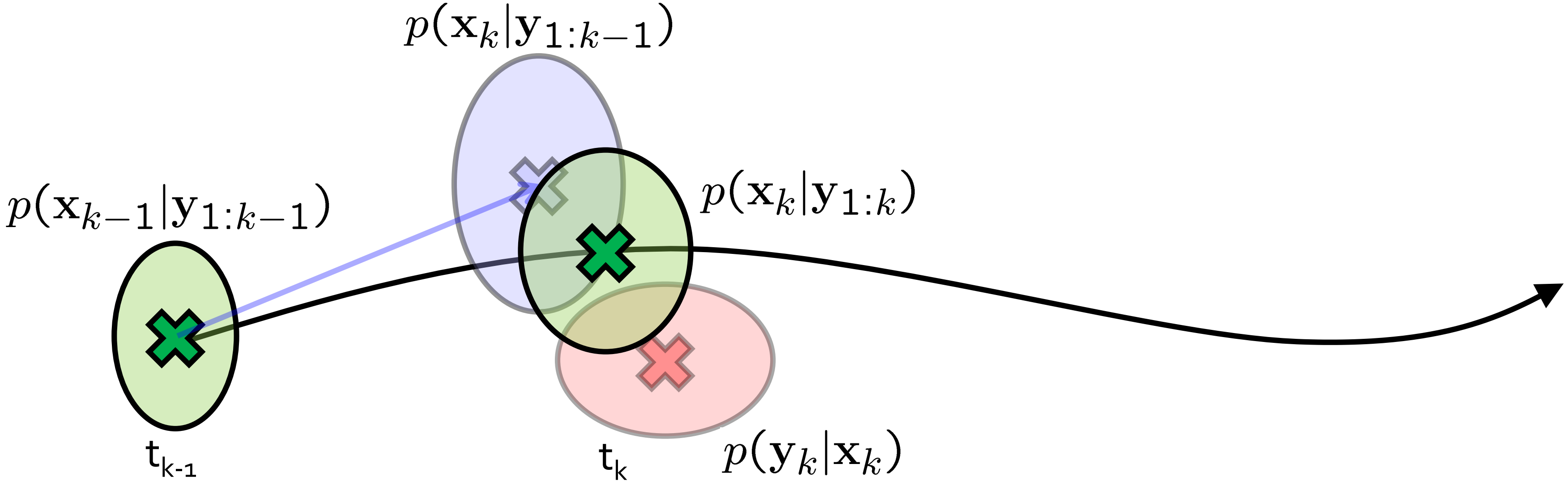
$\propto$

$p(\mathbf{x}_k|\mathbf{y}_{1:k})$
Posterior pdf



Recursive Bayesian Filter

- At each time-step estimate the posterior:



t_{k+1}

update measure predict

$$\underbrace{p(\mathbf{x}_k|\mathbf{y}_{1:k})}_{\text{posterior estimate}} \propto \underbrace{p(\mathbf{y}_k|\mathbf{x}_k)}_{\text{Observation model}} \int \underbrace{p(\mathbf{x}_k|\mathbf{x}_{k-1})}_{\text{motion model}} \underbrace{p(\mathbf{x}_{k-1}|\mathbf{y}_{1:k-1})}_{\text{posterior at k-1}} d\mathbf{x}_{k-1}$$

Acknowledgement

- Some images and parts of slides have been taken from the following talks:
  - Kevin Smith’s ”SELECTED TOPICS IN COMPUTER VISION – 2D tracking”