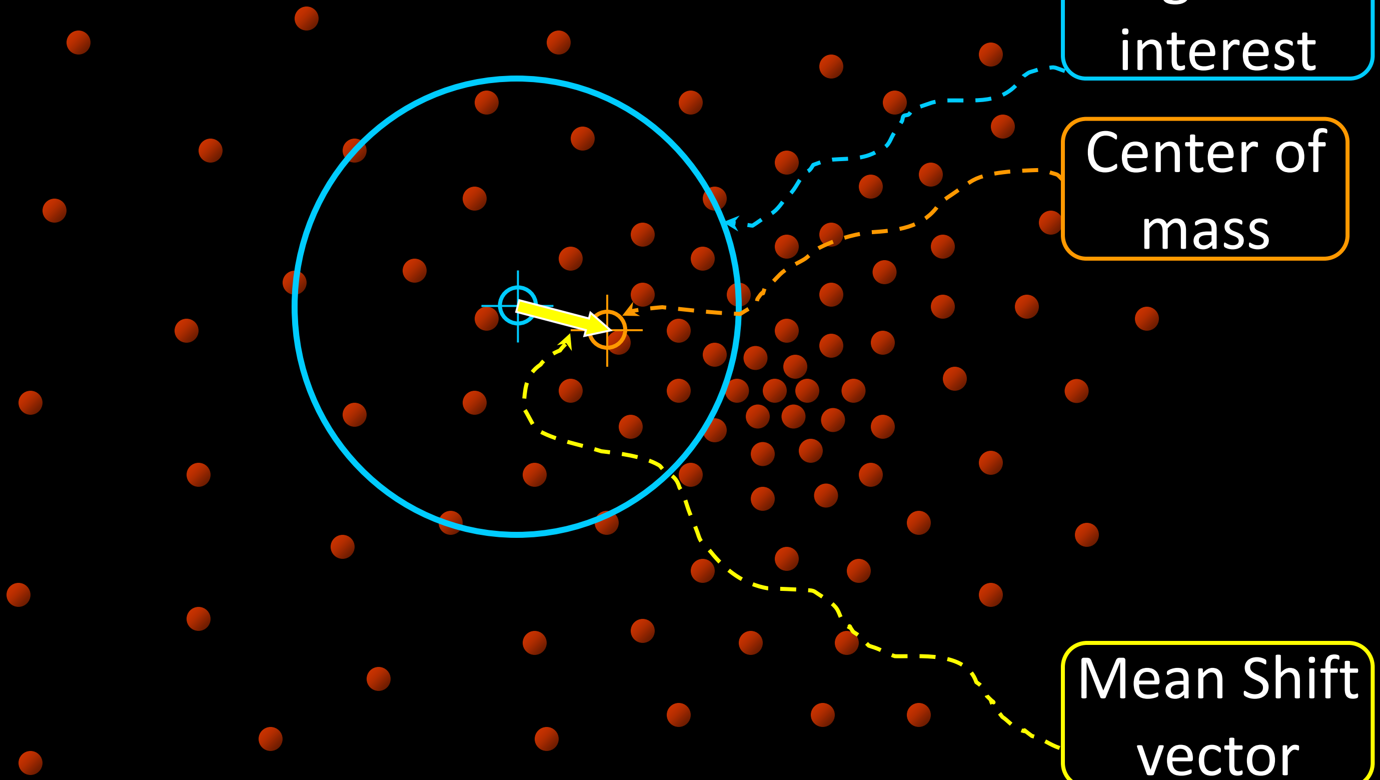
CS4495/6495
Introduction to Computer Vision

7D-L1 *Tracking considerations*

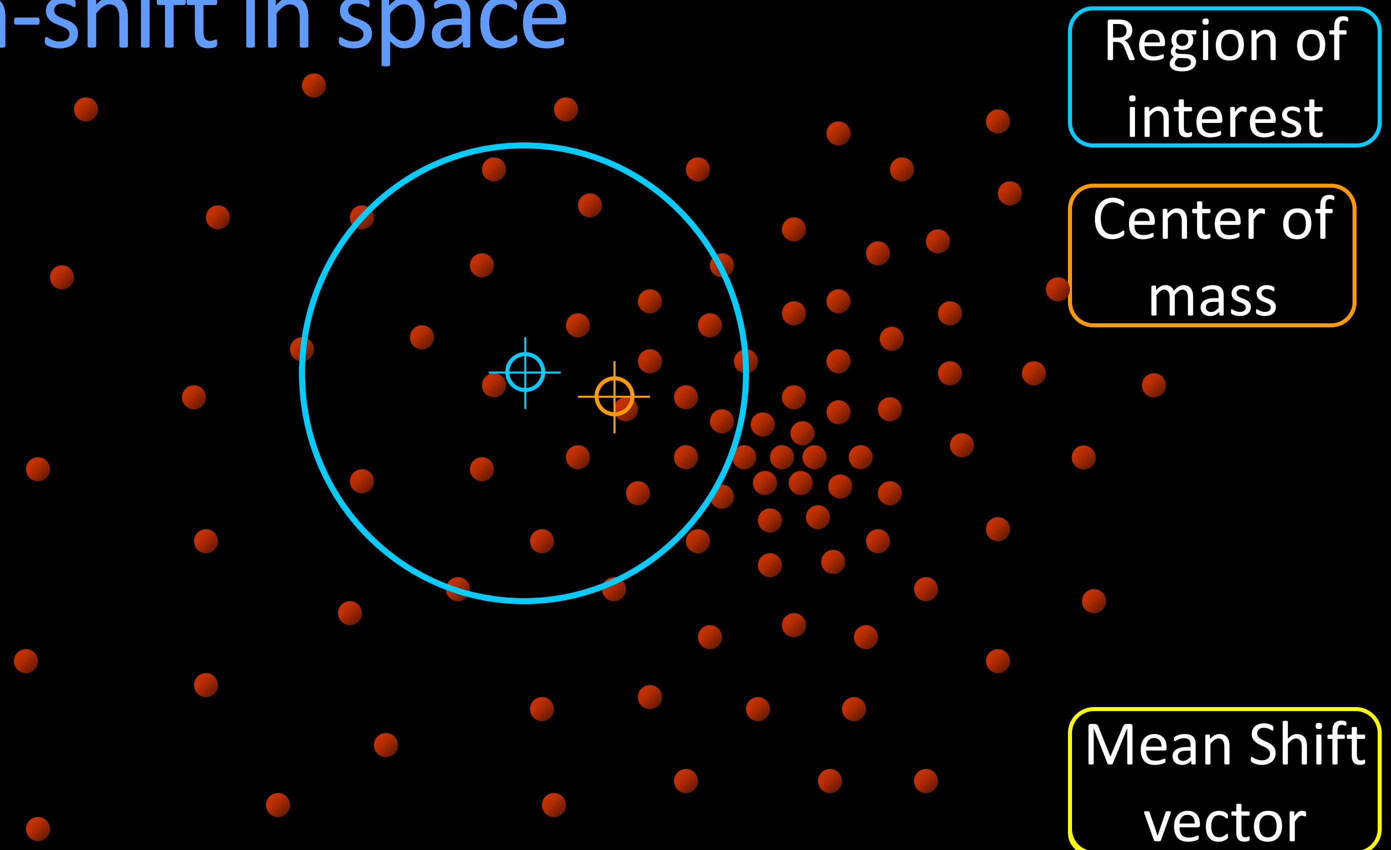
Remember Mean-shift (or preview?)

- Mean-shift – easiest to introduce when doing segmentation.
- The idea is to find the *modes* of a distribution, or a probability density.
- The assumption is you have a set of instances drawn from a PDF and you want to find the mode.

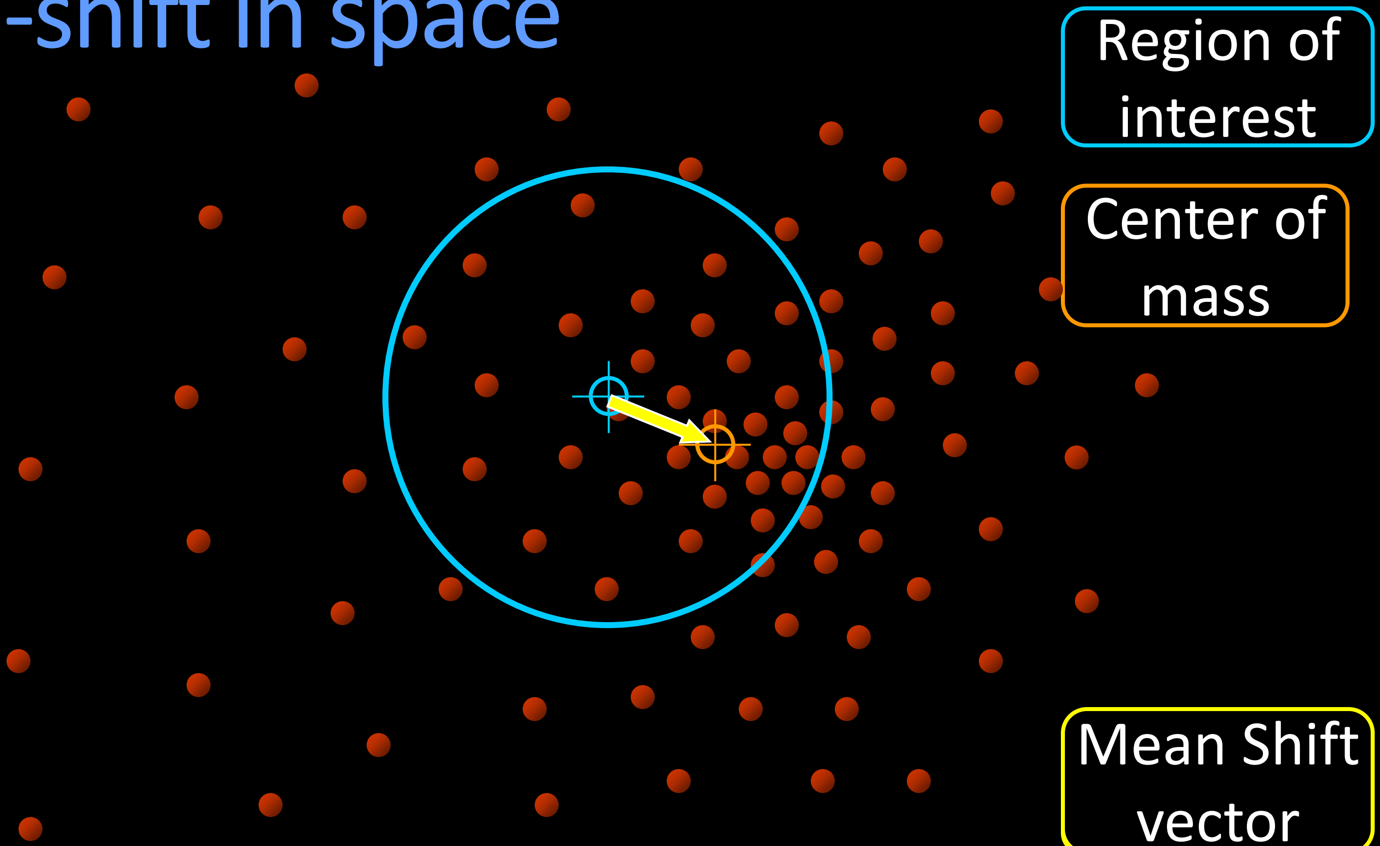
Mean-shift in space



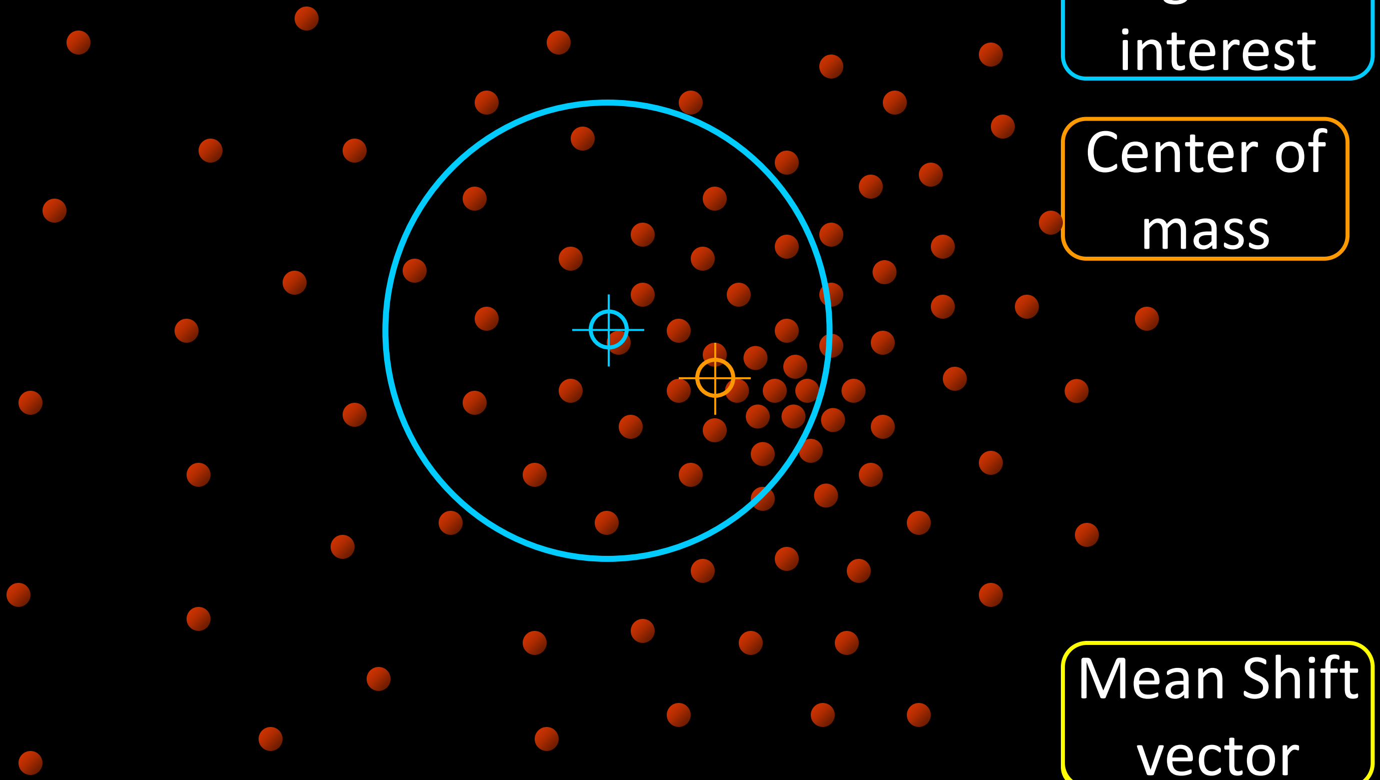
Mean-shift in space



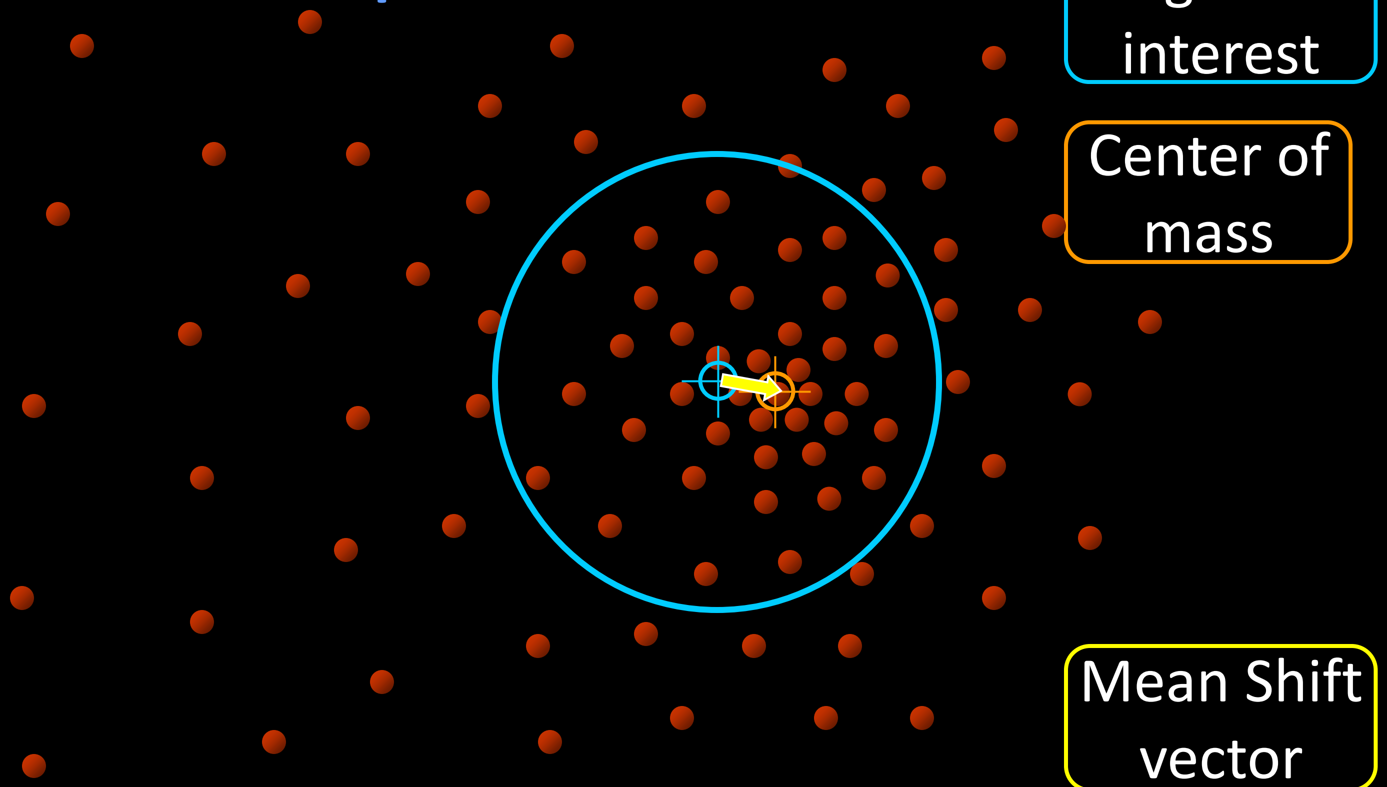
Mean-shift in space



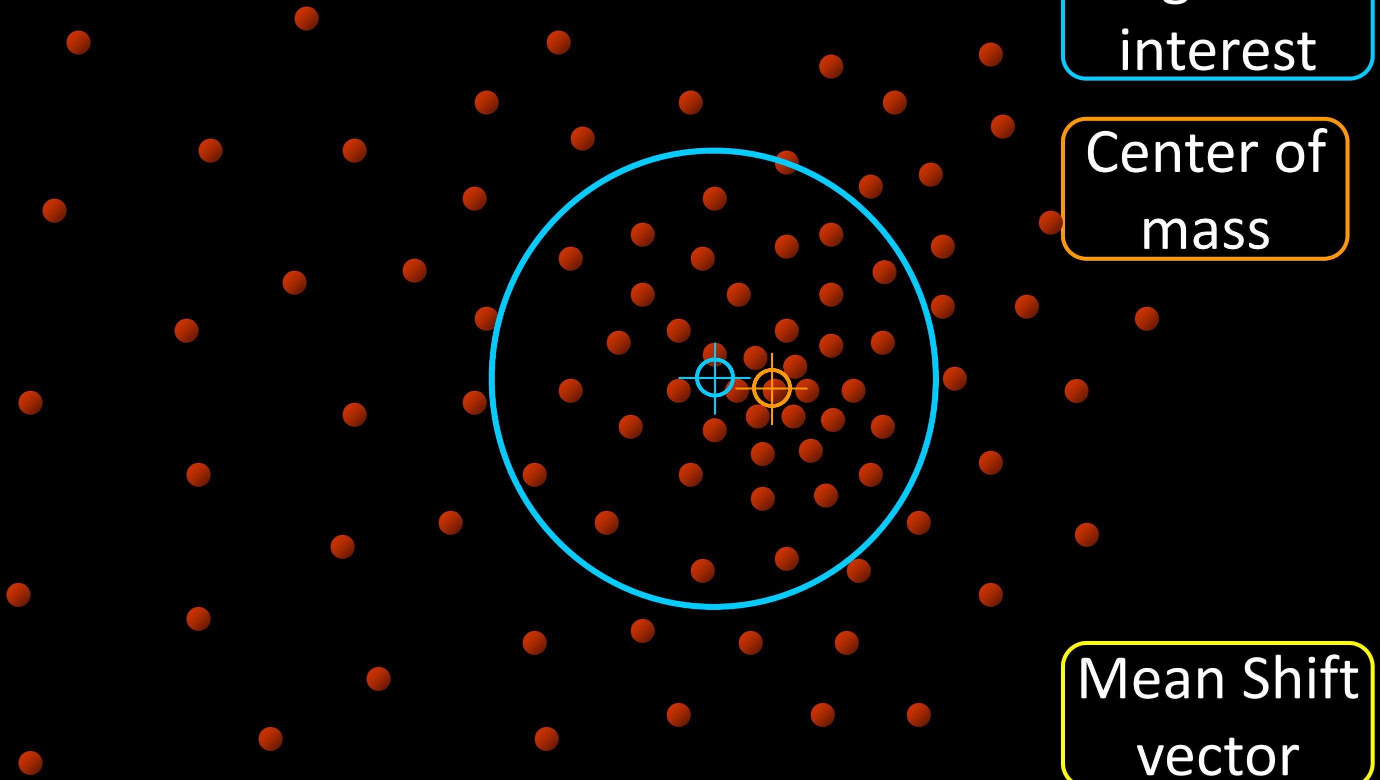
Mean-shift in space



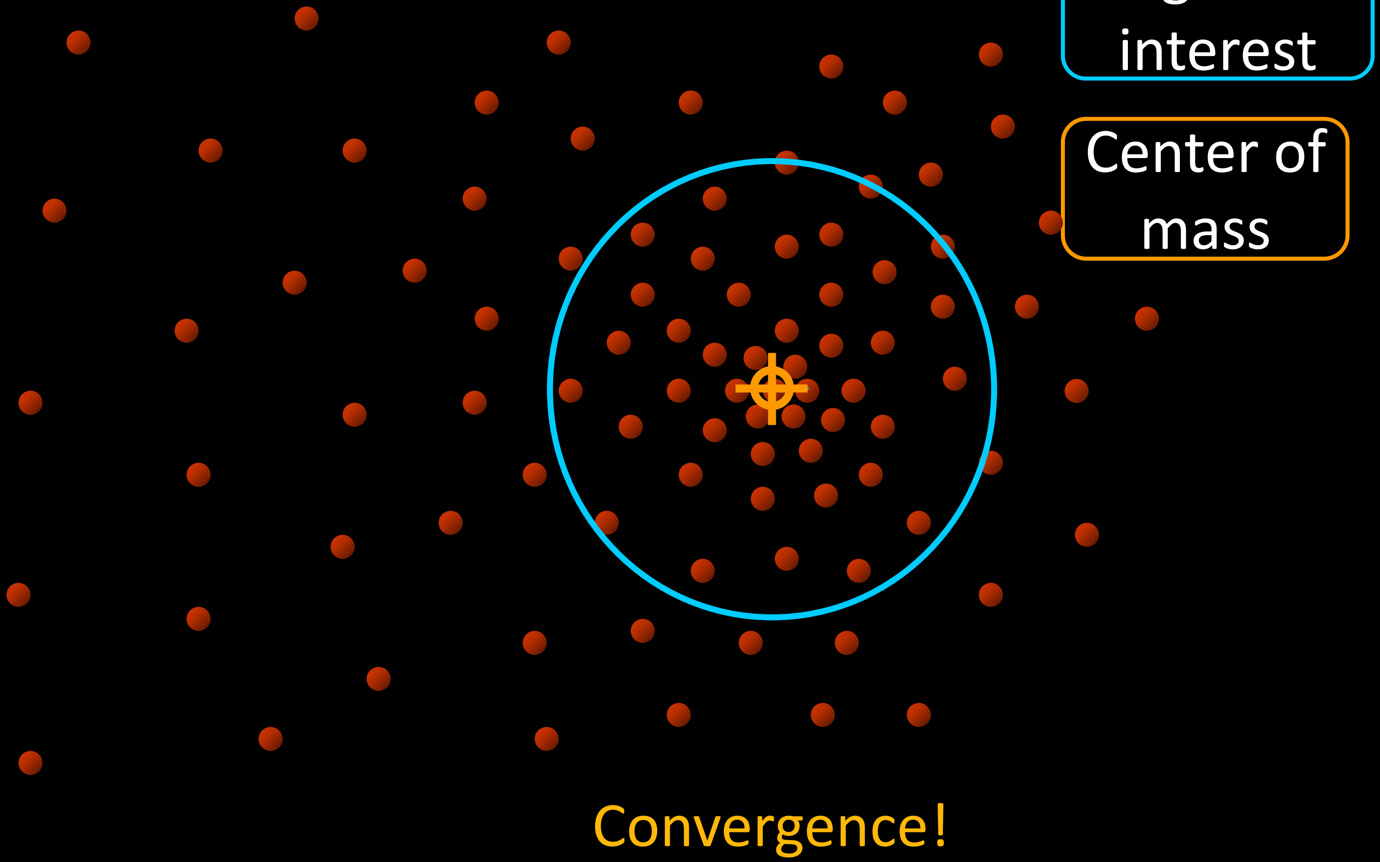
Mean-shift in space



Mean-shift in space

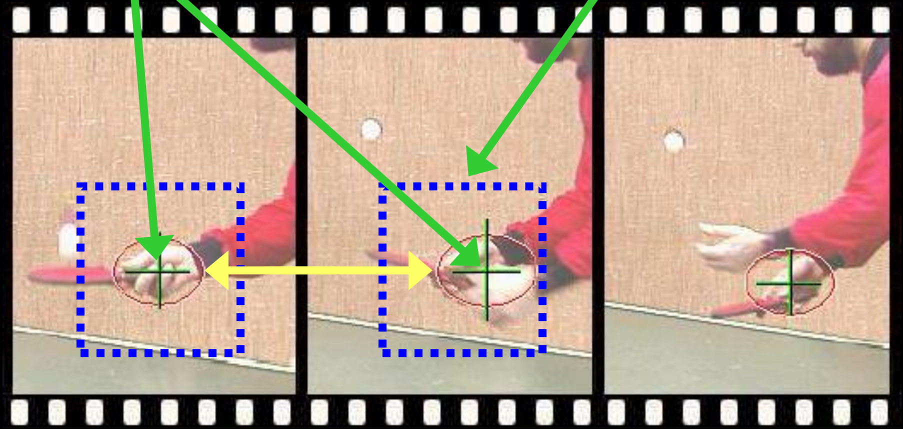


Mean-shift in space



Mean-shift Object Tracking

Start from the position of the model in the current frame → Search neighborhood in next frame → Find best by maximizing a similarity func.



Model

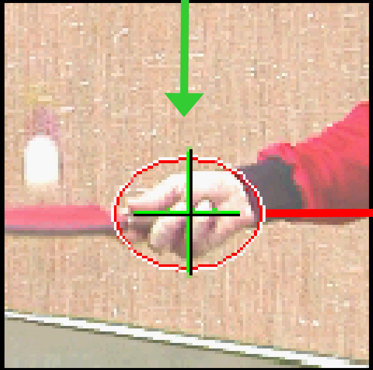
Candidate

▪ ▪ ▪ Current frame → ▪ ▪ ▪

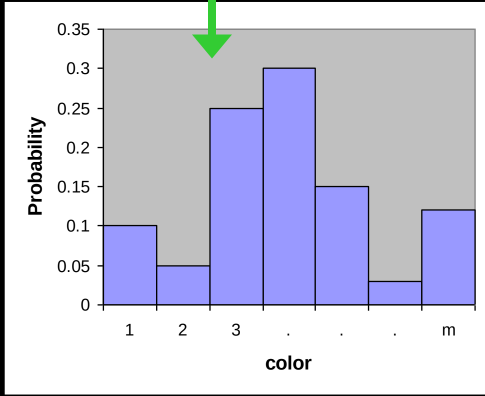
Repeat the same process in the next pair of frames

Mean-shift Object Tracking: Representation

Choose a reference target model → Choose a feature space → Represent the model by its PDF in the feature space



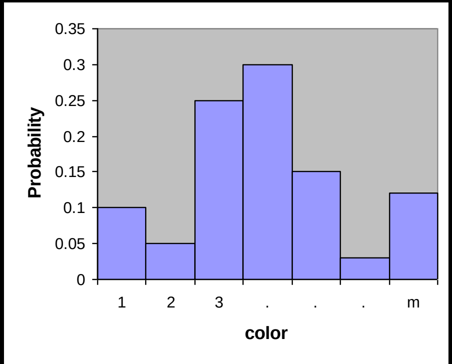
Quantized Color Space

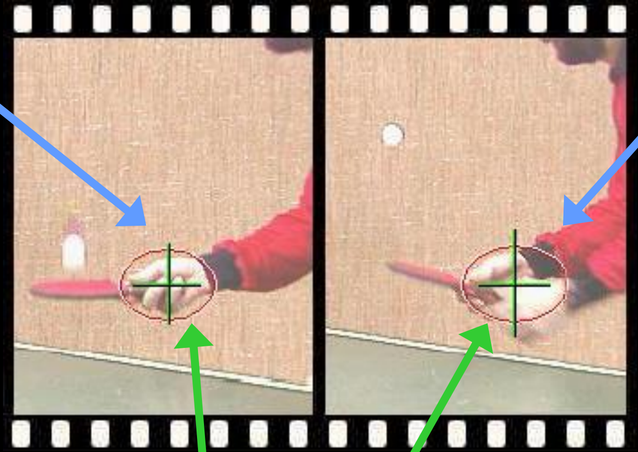


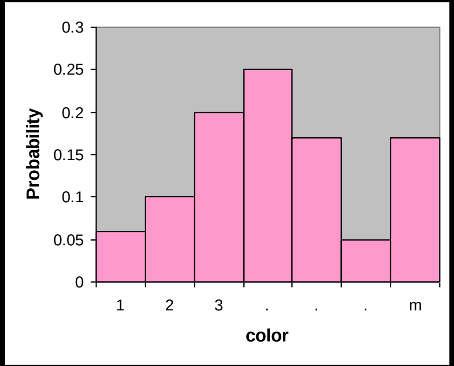
Kernel Based Object Tracking, by Comaniniu, Ramesh, Meer

Target Model
(centered at 0)

Target Candidate
(centered at y)







$$\vec{q} = \{q_u\}_{u=1..m} \quad \sum_{u=1}^{m} q_u = 1$$

$$\vec{p}(y) = \{p_u(y)\}_{u=1..m} \quad \sum_{u=1}^{m} p_u = 1$$

Similarity Function: $f(y) = f\left[\vec{q}, \vec{p}(y)\right]$

Mean-shift Object Tracking: Similarity Function

Target model: $\vec{q} = \left(q_1, \ldots, q_m\right)$

Target candidate: $\vec{p}\left(y\right) = \left(p_1\left(y\right), \ldots, p_m\left(y\right)\right)$

Similarity function: $f\left(y\right) = f\left[\vec{p}\left(y\right), \vec{q}\right] = ?$

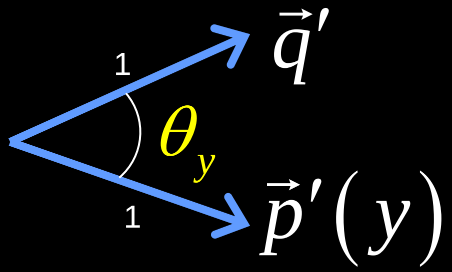
Mean-shift Object Tracking: Similarity Function

The Bhattacharyya Coefficient

$$\vec{q}' = \left(\sqrt{q_1}, \ldots, \sqrt{q_m}\right)$$

$$\vec{p}'(y) = \left(\sqrt{p_1(y)}, \ldots, \sqrt{p_m(y)}\right)$$

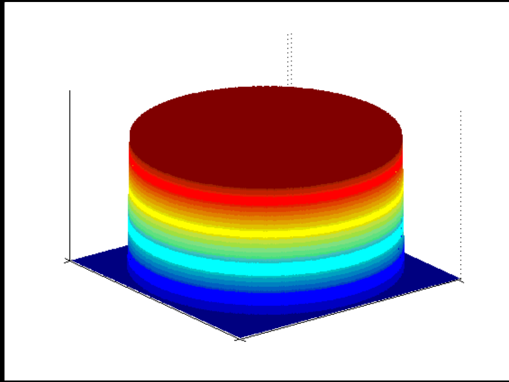
$$f(y) = \boxed{\sum_{u=1}^{m} \sqrt{p_u(y)\, q_u}} = \frac{p'(y)^T q'}{\|p'(y)\| \cdot \|q'\|} = \cos\theta_y$$



Mean-shift Object Tracking: *Gradient*

- In the examples before, we computed the mean or density over a fixed region.
- That's actually a *uniform kernel:*

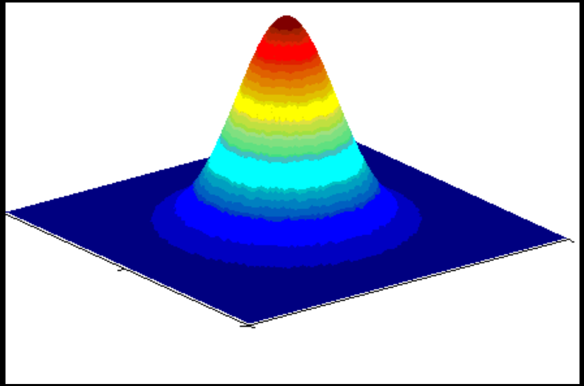
$$K_U(\mathbf{x}) = \begin{cases} c & \|\mathbf{x}\| \le 1 \\ 0 & \text{otherwise} \end{cases}$$



Mean-shift Object Tracking: *Gradient*

- Could instead use a differentiable, isotropic, monotonically decreasing kernel
- For example: *normal (Gaussian)*

$$K_N(\mathbf{x}) = c \cdot \exp\left(-\frac{1}{2}\|\mathbf{x}\|^2\right)$$

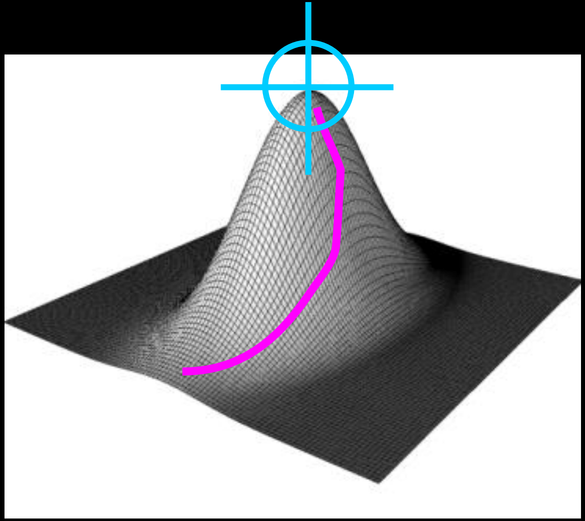


- Can also have a scale factor
- *Differentiable…*

Mean-shift Object Tracking: *Gradient*

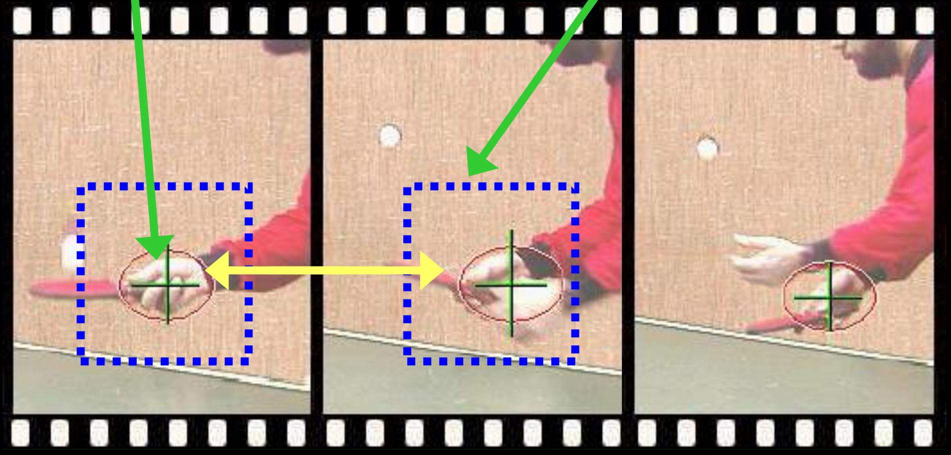
Why a gradient?

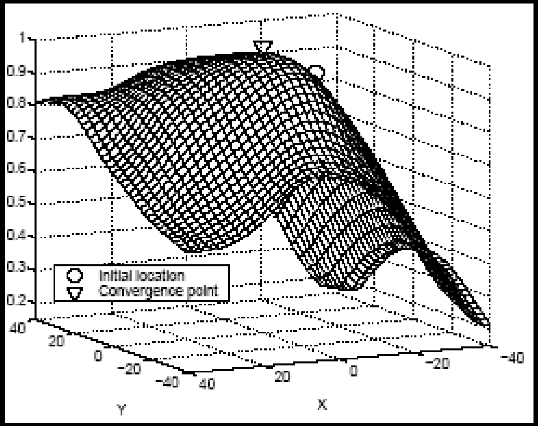
- You can move to the mode without blind search:



Mean-shift Object Tracking

Start from the position of the model in the current frame → Search neighborhood in next frame → Find best by maximizing a similarity func.





Mean-shift Tracking Results

Feature space: $16 \times 16 \times 16$ quantized RGB

Target: manually selected on 1st frame

Average mean-shift iterations: 4

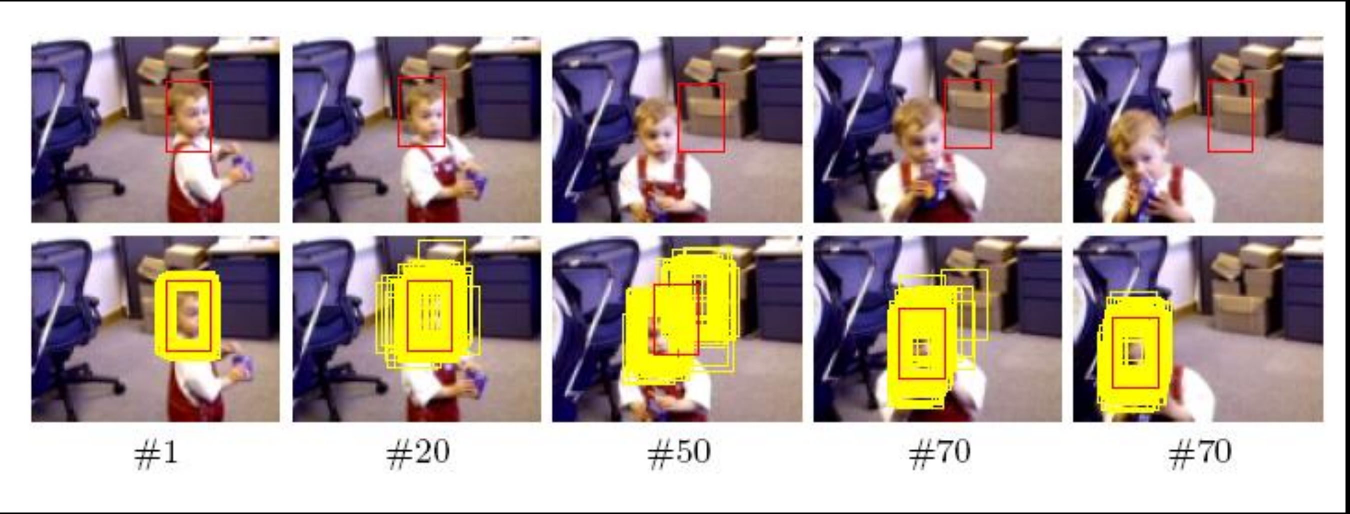
Or just another sensor model…

- The notion of “best” is back to our “single” hypothesis – like Kalman.
- Could just use the similarity function as a sensor model for particle filtering…

An unfair comparison…

Mean-shift

Probabilistic particle filters



Tracking people by learning their appearance

Person model =

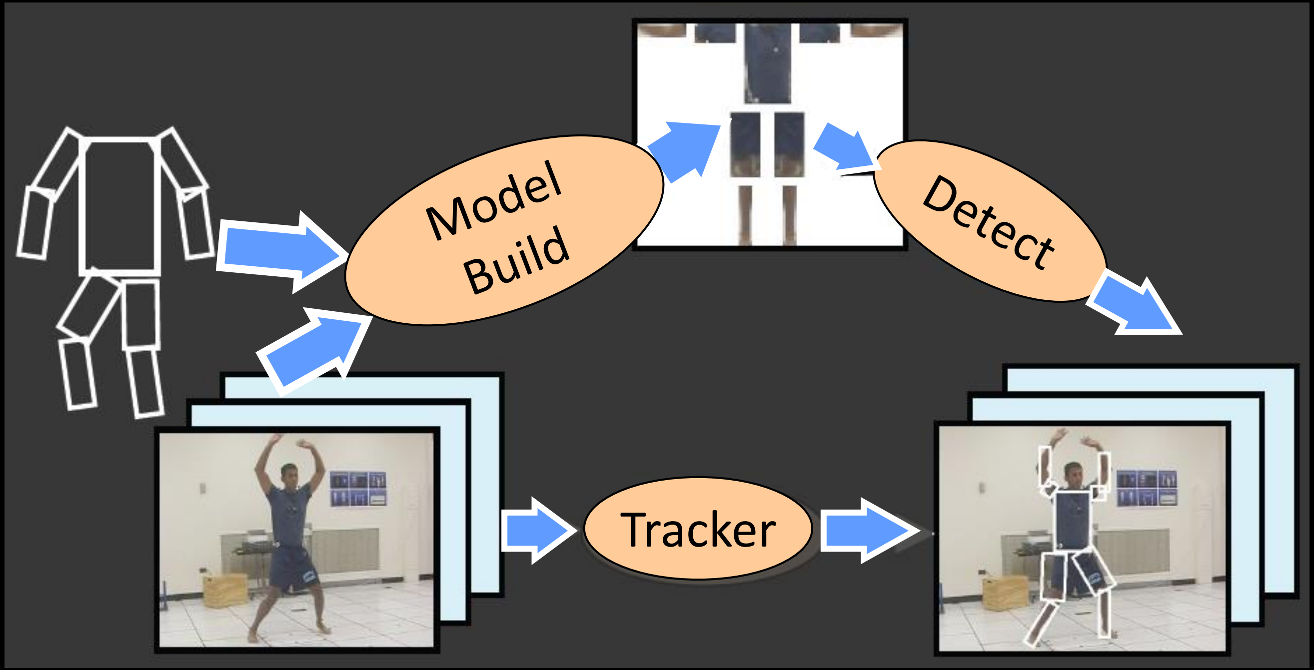
appearance

\+ structure

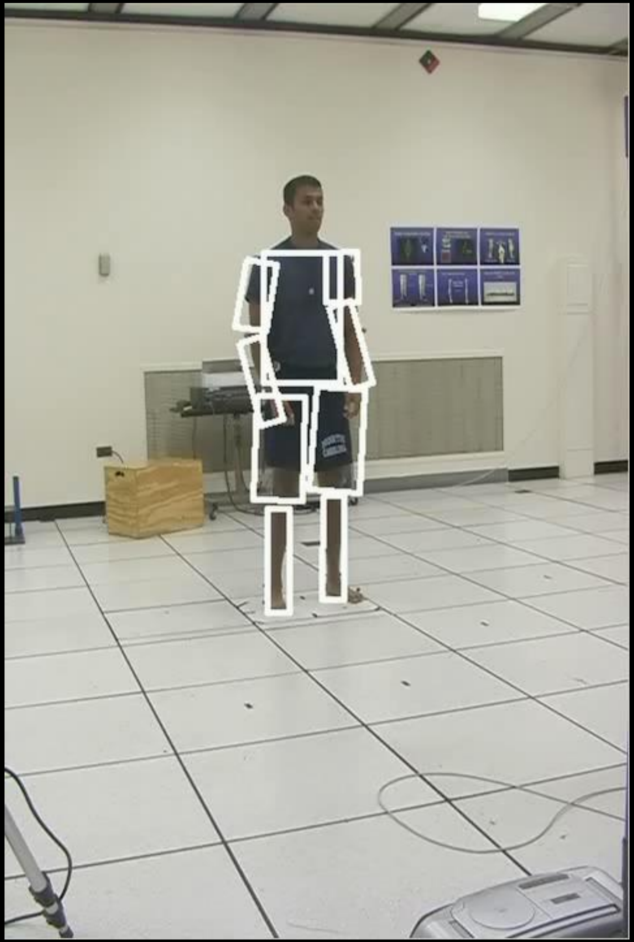
\+ dynamics

Structure and dynamics are generic, but appearance is person-specific

Tracking people by learning their appearance



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



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

Tracking issues

- Initialization
  - Manual
  - Background subtraction
  - Detection

Tracking issues

- Initialization
- Obtaining observation and dynamics model
  - Dynamics model: learn from real data (pretty difficult), learn from “clean data” (easier), or specify using domain knowledge (aka you are the smart one).
  - Generative observation model – some form of ground truth required.

Tracking issues

- Initialization
- Obtaining observation and dynamics model
- Prediction vs. correction
  - If the dynamics model is too strong, will end up ignoring the data
  - If the observation model is too strong, tracking is reduced to repeated detection

Tracking issues

- Initialization
- Obtaining observation and dynamics model
- Prediction vs. correction
- Data association
  - What if we don’t know which measurements to associate with which tracks?

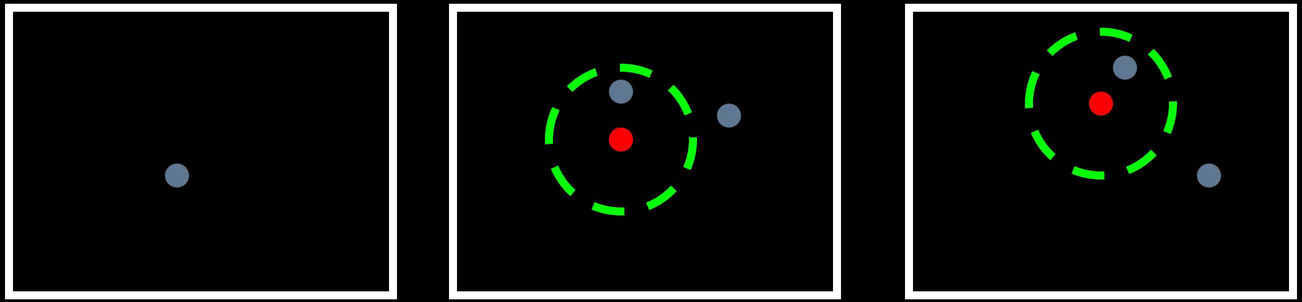
Data association

- So far, we've assumed the entire measurement to be relevant to determining the state
- In reality, multiple objects or *clutter* (uninformative measurements)

Data association: Determining which measurements go with which tracks

Data association

Simple strategy: Only pay attention to the measurement that is *closest* to the prediction



Source: Lana Lazebnik

Data association

More sophisticated strategy: Keep track of multiple state/observation hypotheses

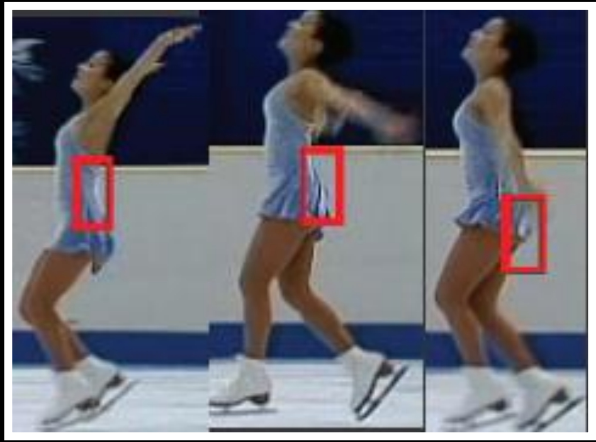
- Can be done with a set of particles (how?)

Each particle is a hypothesis about current state

Tracking issues

- Initialization
- Obtaining observation and dynamics model
- Prediction vs. correction
- Data association
- Drift
  - Errors caused by dynamical model, observation model, and data association tend to accumulate over time

Drift



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

Tracking: Summary

- Cool part of computer vision!
- Key elements: Probabilistic state (prediction), measurements, & combining them (correction)
- CV’s contribution to tracking: Maintaining a consistent interpretation over time