CS 554
Computer Vision

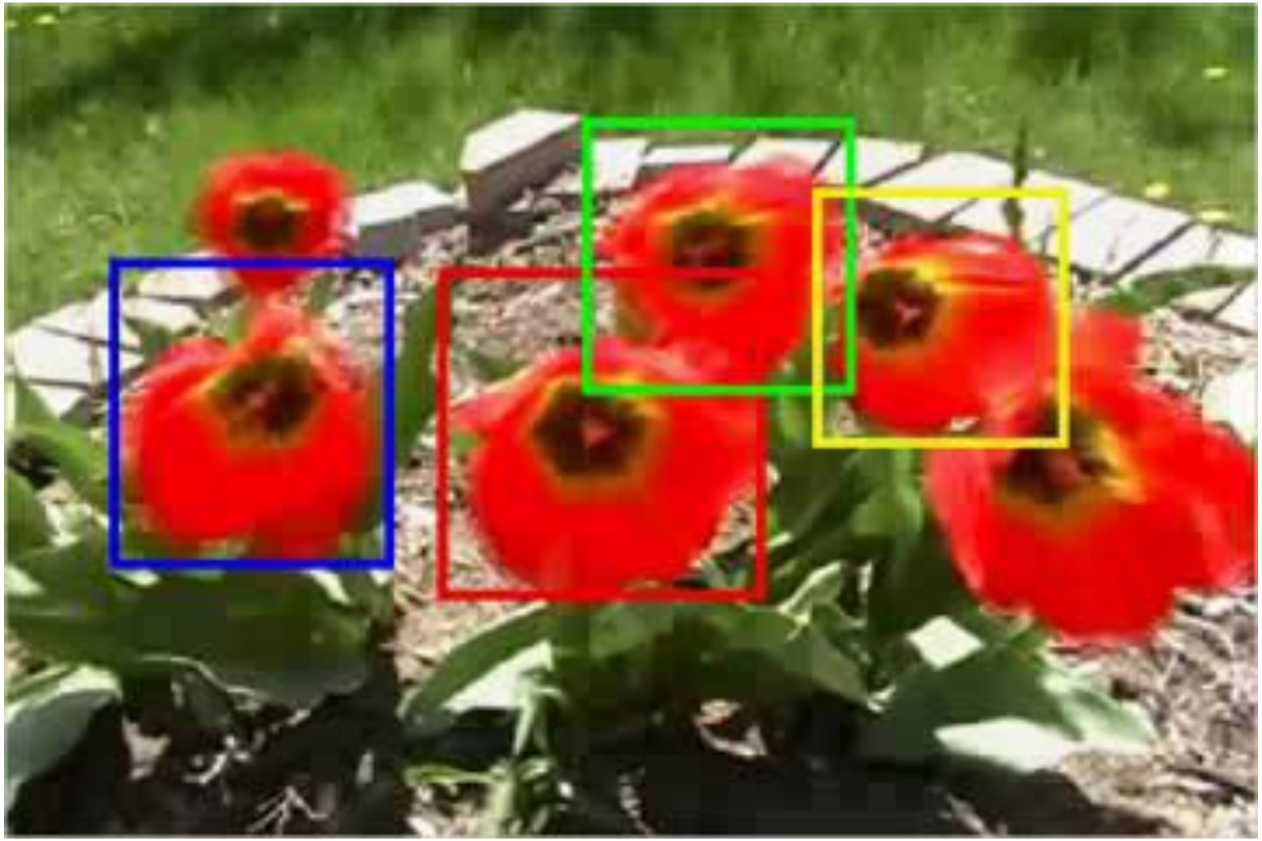
Tracking

Hamdi Dibeklioğlu

Slide Credits: L. van der Maaten

Tracking

Tracking aims to follow an object or object configuration in a video sequence:



Tracking

Every tracker consists of three main components:

- *Likelihood*: How well does a configuration explain the current observation?

Tracking

Every tracker consists of three main components:

- *Likelihood*: How well does a configuration explain the current observation?
- *Prior*: Given the previous object location, how likely is a configuration?

Tracking

Every tracker consists of three main components:

- *Likelihood*: How well does a configuration explain the current observation?
- *Prior*: Given the previous object location, how likely is a configuration?
- *Search strategy*: How do we maximize *posterior = likelihood x prior*?

Tracking

Every tracker consists of three main components:

- *Likelihood*: How well does a configuration explain the current observation?
- *Prior*: Given the previous object location, how likely is a configuration?
- *Search strategy*: How do we maximize *posterior = likelihood x prior*?

Recall *Bayes' rule:*

$$p(\mathbf{x}_t|\mathbf{x}_{t-1},\mathbf{I}_t) = \frac{p(\mathbf{I}_t|\mathbf{x}_t,\mathbf{x}_{t-1})p(\mathbf{x}_t|\mathbf{x}_{t-1})}{p(\mathbf{I}_t|\mathbf{x}_{t-1})}$$

Tracking

Every tracker consists of three main components:

- *Likelihood*: How well does a configuration explain the current observation?
- *Prior*: Given the previous object location, how likely is a configuration?
- *Search strategy*: How do we maximize *posterior = likelihood x prior*?

Recall *Bayes' rule:* $$p(\mathbf{x}_t|\mathbf{x}_{t-1},\mathbf{I}_t) = \frac{p(\mathbf{I}_t|\mathbf{x}_t,\mathbf{x}_{t-1})p(\mathbf{x}_t|\mathbf{x}_{t-1})}{p(\mathbf{I}_t|\mathbf{x}_{t-1})}$$

location and size
of object at time *t*

Tracking

Every tracker consists of three main components:

- *Likelihood*: How well does a configuration explain the current observation?
- *Prior*: Given the previous object location, how likely is a configuration?
- *Search strategy*: How do we maximize *posterior = likelihood x prior*?

Recall *Bayes' rule:* $$p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{I}_t) = \frac{p(\mathbf{I}_t|\mathbf{x}_t, \mathbf{x}_{t-1})p(\mathbf{x}_t|\mathbf{x}_{t-1})}{p(\mathbf{I}_t|\mathbf{x}_{t-1})}$$

location and size of object at time *t*

observed image at time *t*

Tracking

Every tracker consists of three main components:

- *Likelihood*: How well does a configuration explain the current observation?
- *Prior*: Given the previous object location, how likely is a configuration?
- *Search strategy*: How do we maximize *posterior = likelihood x prior*?

Recall *Bayes' rule:* $$p(\mathbf{x}_t|\mathbf{x}_{t-1},\mathbf{I}_t) = \frac{p(\mathbf{I}_t|\mathbf{x}_t,\mathbf{x}_{t-1})p(\mathbf{x}_t|\mathbf{x}_{t-1})}{\cancel{p(\mathbf{I}_t|\mathbf{x}_{t-1})}}$$

location and size of object at time *t*

observed image at time *t*

Tracking

Every tracker consists of three main components:

- *Likelihood*: How well does a configuration explain the current observation?
- *Prior*: Given the previous object location, how likely is a configuration?
- *Search strategy*: How do we maximize *posterior = likelihood x prior*?

Recall *Bayes' rule:*

$$p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{I}_t) = \frac{p(\mathbf{I}_t|\mathbf{x}_t, \cancel{\mathbf{x}_{t-1}})p(\mathbf{x}_t|\mathbf{x}_{t-1})}{\cancel{p(\mathbf{I}_t|\mathbf{x}_{t-1})}}$$

location and size of object at time *t*

observed image at time *t*

Tracking

Every tracker consists of three main components:

- *Likelihood*: How well does a configuration explain the current observation?
- *Prior*: Given the previous object location, how likely is a configuration?
- *Search strategy*: How do we maximize *posterior = likelihood x prior*?

Recall *Bayes' rule:* $p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{I}_t) \propto p(\mathbf{I}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{x}_{t-1})$

location and size of object at time *t*

observed image at time *t*

Tracking

Appearance likelihood can be any object-recognition model

Tracking

Appearance likelihood can be any object-recognition model

Motion prior can be a linear dynamical system

Tracking

Appearance likelihood can be any object-recognition model

Motion prior can be a linear dynamical system

Search strategy can be one of the following four approaches:

- Kalman (only for linear-Gaussian likelihood and prior) or particle filtering
- Lucas-Kanade algorithm (only for template matching using squared errors)
- Mean-shift tracking
- Sliding-window search (essentially brute-force search)

Kalman filter

Kalman filter

- We aim to infer the location of the object, $\mathbf{x}_t$, at time t

- A simple *motion prior* could be defined as:

$$p(\mathbf{x}_t|\mathbf{x}_{t-1}) \propto \exp\left(-\frac{1}{2\sigma^2}\|\mathbf{x}_t - \mathbf{A}\mathbf{x}_{t-1}\|^2\right)$$

- A simple *likelihood* model could be defined as:

$$p(\mathbf{I}_t|\mathbf{x}_t) \propto \exp\left(-\frac{1}{2\tau^2}\|\mathbf{B}\mathbf{x}_t - \phi(\mathbf{I}_t)\|^2\right)$$

- Combining the two via Bayes' rule: $p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{I}_t) \propto p(\mathbf{I}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{x}_{t-1})$

Kalman filter

- Combining the two via Bayes' rule: $p(\mathbf{x}_t|\mathbf{x}_{t-1},\mathbf{I}_t) \propto p(\mathbf{I}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{x}_{t-1})$

- Because both terms are Gaussian, the result is also Gaussian:

$$p(\mathbf{x}_t|\mathbf{x}_{t-1},\mathbf{I}_t) = \mathcal{N}(\mathbf{x}_t|\boldsymbol{\mu}_t,\boldsymbol{\Sigma}_t)$$

Kalman filter

- Combining the two via Bayes' rule: $p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{I}_t) \propto p(\mathbf{I}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{x}_{t-1})$

- Because both terms are Gaussian, the result is also Gaussian:

$$p(\mathbf{x}_t|\mathbf{x}_{t-1}) \propto \exp\left(-\frac{1}{2\sigma^2}\|\mathbf{x}_t - \mathbf{A}\mathbf{x}_{t-1}\|^2\right)$$

$$p(\mathbf{I}_t|\mathbf{x}_t) \propto \exp\left(-\frac{1}{2\tau^2}\|\mathbf{B}\mathbf{x}_t - \phi(\mathbf{I}_t)\|^2\right)$$

$$p(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t \mid \boldsymbol{A}\mathbf{x}_{t-1}, \sigma^2)$$

$$p(\mathbf{I}_t|\mathbf{x}_t) = \mathcal{N}(\phi(\mathbf{I}_t) \mid \boldsymbol{B}\mathbf{x}_t, \tau^2)$$

$$p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{I}_t) = \mathcal{N}(\mathbf{x}_t|\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$$

Kalman filter

- Combining the two via Bayes' rule: $p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{I}_t) \propto p(\mathbf{I}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{x}_{t-1})$

- Because both terms are Gaussian, the result is also Gaussian:

$$p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{I}_t) = \mathcal{N}(\mathbf{x}_t|\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$$

Kalman filter

- Combining the two via Bayes' rule: $p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{I}_t) \propto p(\mathbf{I}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{x}_{t-1})$

- Because both terms are Gaussian, the result is also Gaussian:

$$p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{I}_t) = \mathcal{N}(\mathbf{x}_t|\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$$

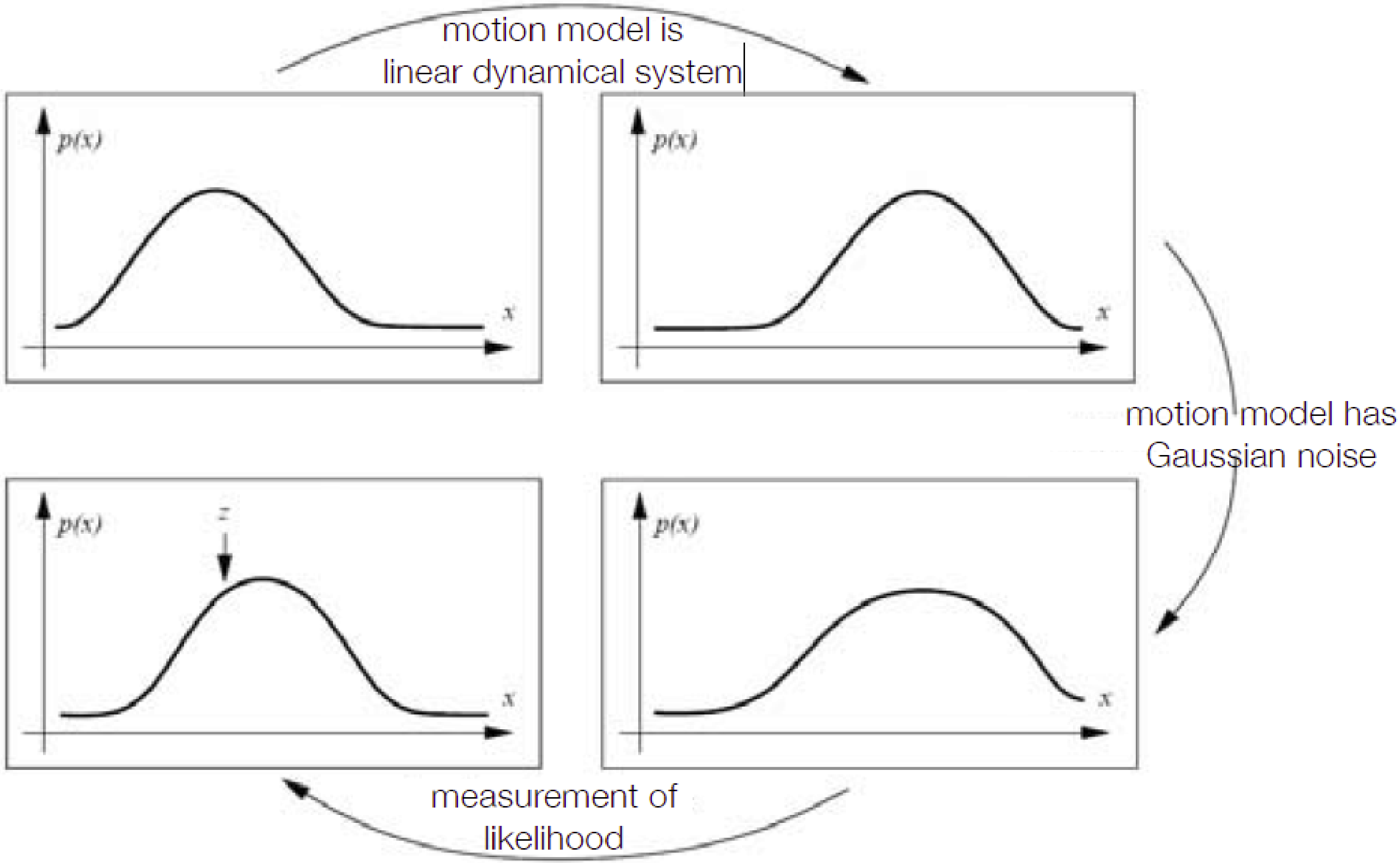
$$\boldsymbol{\mu}_t = \mathbf{A}\boldsymbol{\mu}_{t-1} + \mathbf{K}_t(\phi(\mathbf{I}_t) - \mathbf{B}\mathbf{A}\boldsymbol{\mu}_{t-1})$$

$$\boldsymbol{\Sigma}_t = (\mathbf{I} - \mathbf{K}_t\mathbf{B})\left(\mathbf{A}\boldsymbol{\Sigma}_{t-1}\mathbf{A}^{\mathrm{T}} + \sigma^2\mathbf{I}\right)$$

$$\mathbf{K}_t = \left(\mathbf{A}\boldsymbol{\Sigma}_{t-1}\mathbf{A}^{\mathrm{T}} + \sigma^2\mathbf{I}\right)\mathbf{B}^{\mathrm{T}}\left[\mathbf{B}\left(\mathbf{A}\boldsymbol{\Sigma}_{t-1}\mathbf{A}^{\mathrm{T}} + \sigma^2\mathbf{I}\right)\mathbf{B}^{\mathrm{T}} + \tau^2\mathbf{I}\right]^{-1}$$

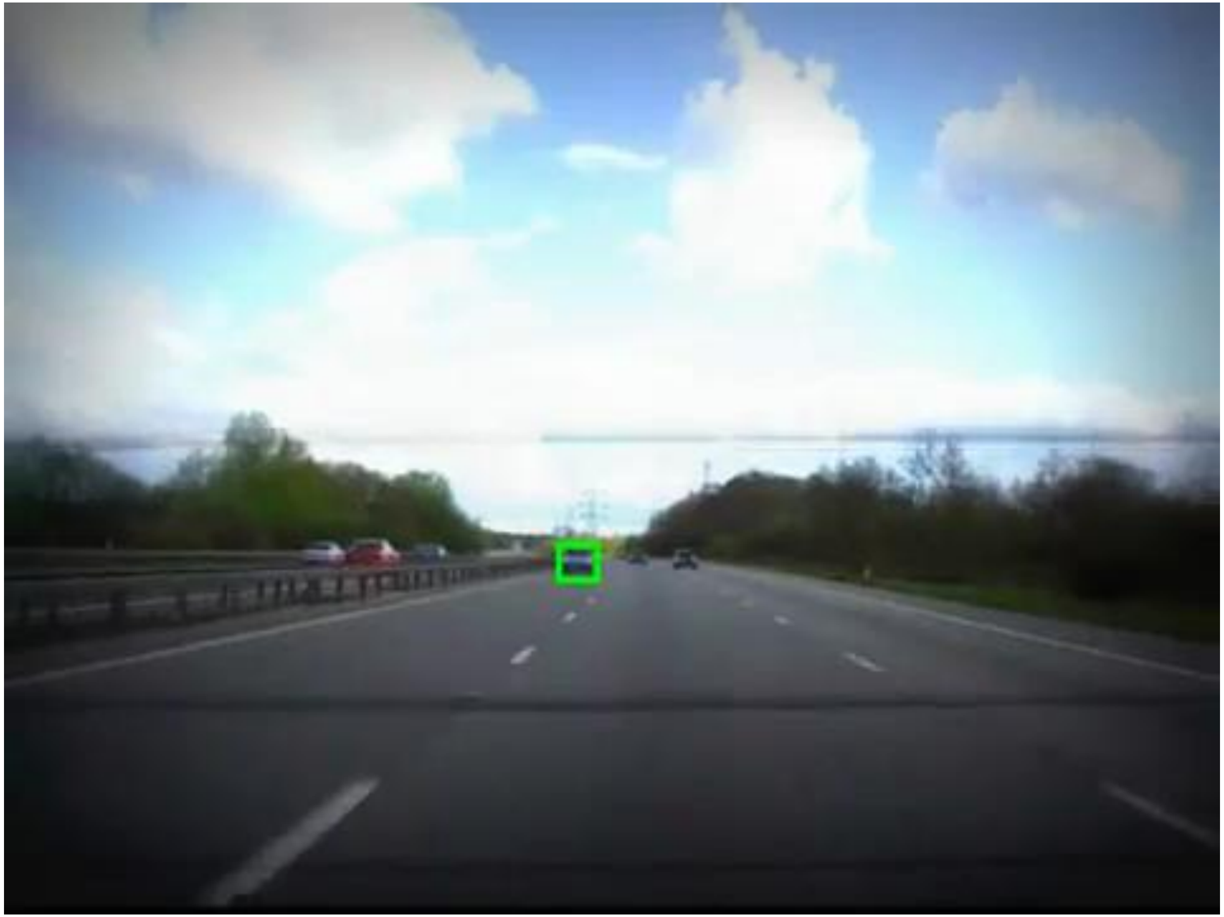
"Kalman gain matrix"

Kalman filter



Example: Kalman filter

Example: Kalman filter



Kalman filter

In practical settings, the Kalman filter does often not work very well:

- The Gaussian likelihood model is generally way too simple in practice: Gaussians are unimodal and can therefore only maintain a single hypothesis

Kalman filter

In practical settings, the Kalman filter does often not work very well:

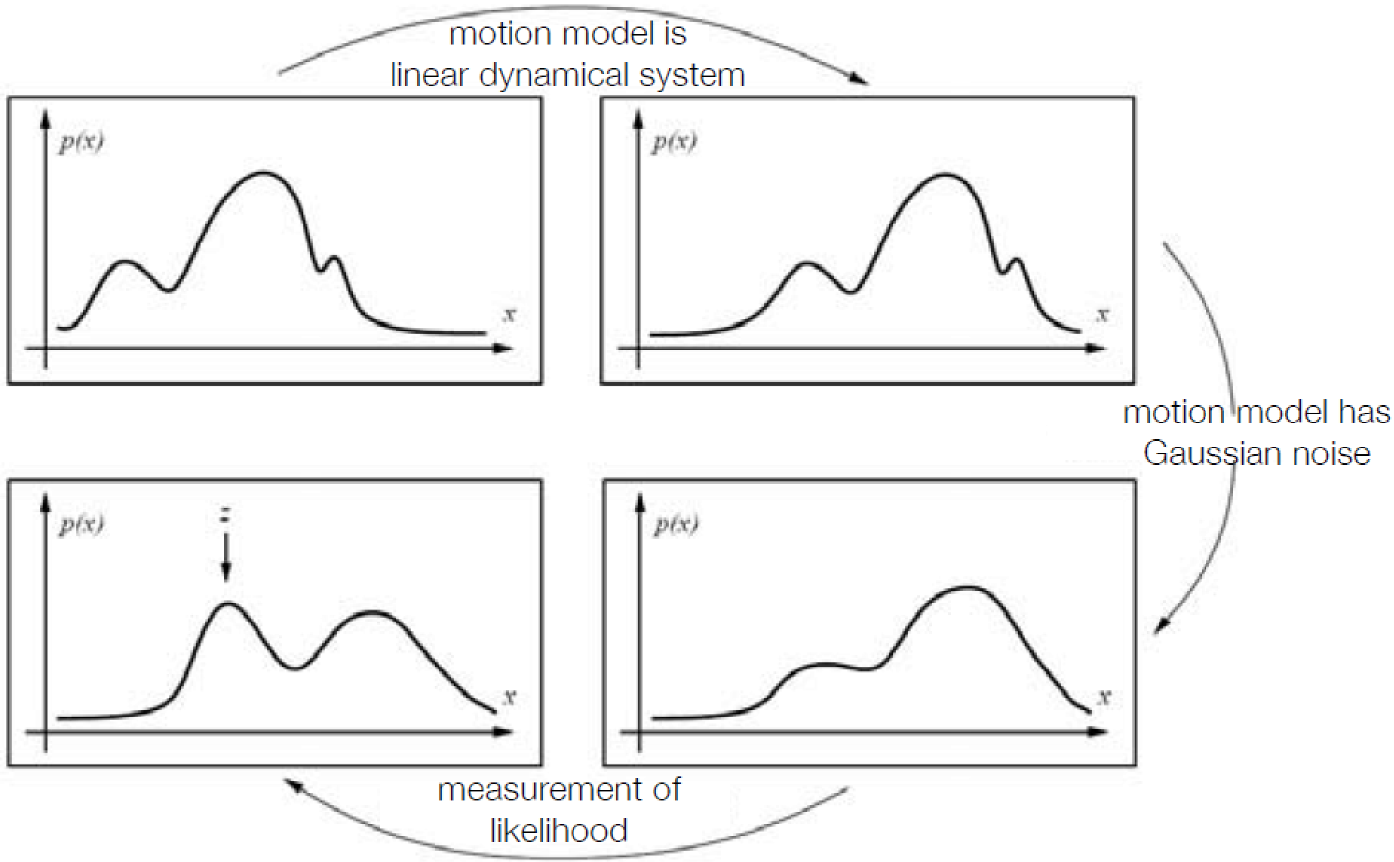
- The Gaussian likelihood model is generally way too simple in practice: Gaussians are unimodal and can therefore only maintain a single hypothesis

Changing the likelihood to something non-Gaussian complicates inference:

- Particle filters do this: solve inference problem by *importance sampling*

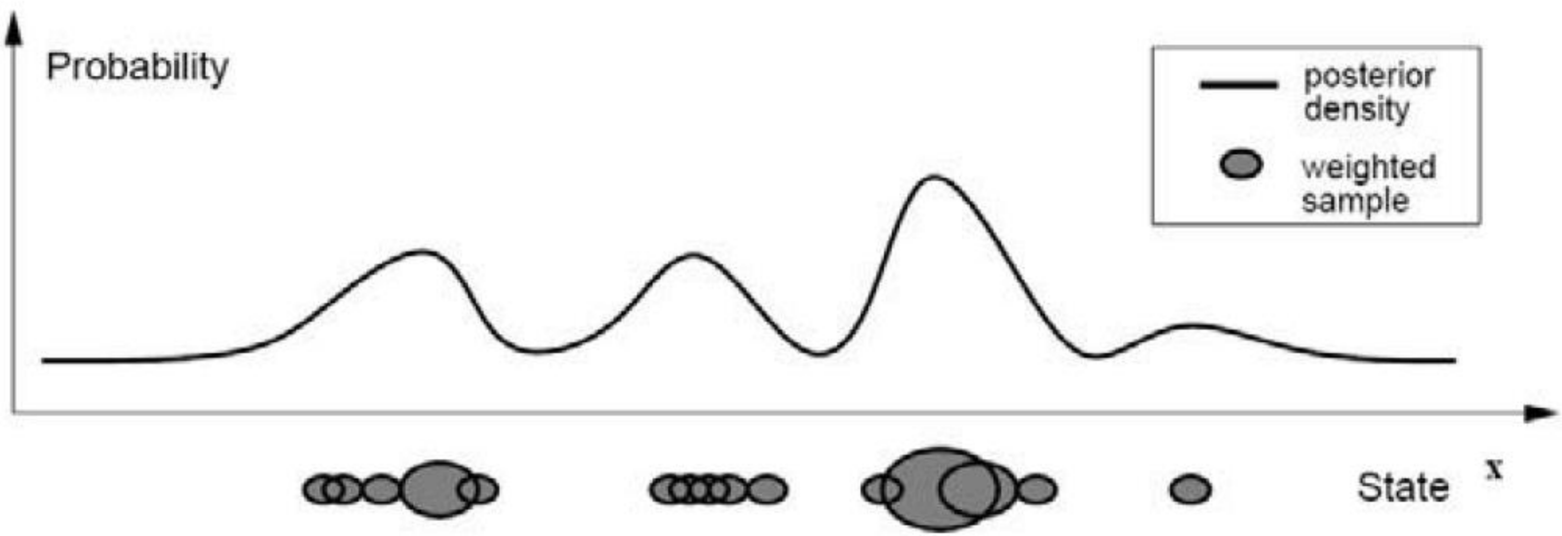
Particle filter

Particle filter



Particle filter (Condensation)

- For complex appearance likelihood functions, exact inferences are impossible
- Particle filters represent $p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{I}_t)$ by a set of weighted samples ("particles"):



Particle filter (Condensation)

- For complex appearance likelihood functions, exact inferences are impossible
- Particle filters sample from $p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{I}_t)$ via *sequential importance sampling*:

Particle filter (Condensation)

- For complex appearance likelihood functions, exact inferences are impossible

- Particle filters sample from $p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{I}_t)$ via *sequential importance sampling*:
  - Assume we have a set of N weighted samples $\{(\mathbf{s}_{t-1}^{(n)}, \pi_{t-1}^{(n)}), n = 1, \ldots, N\}$

Particle filter (Condensation)

- For complex appearance likelihood functions, exact inferences are impossible

- Particle filters sample from $p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{I}_t)$ via *sequential importance sampling*:
  - Assume we have a set of N weighted samples $\{(\mathbf{s}_{t-1}^{(n)}, \pi_{t-1}^{(n)}), n = 1, \ldots, N\}$
  - Pick sample $\mathbf{s}_{t-1}^{(n)}$ according to weights, and sample from $p(\mathbf{s}_t^{(n)}|\mathbf{s}_{t-1}^{(n)})$ (repeat N times)

Particle filter (Condensation)

- For complex appearance likelihood functions, exact inferences are impossible
- Particle filters sample from $p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{I}_t)$ via *sequential importance sampling*:
  - Assume we have a set of N weighted samples $\{(\mathbf{s}_{t-1}^{(n)}, \pi_{t-1}^{(n)}), n = 1, \ldots, N\}$
  - Pick sample $\mathbf{s}_{t-1}^{(n)}$ according to weights, and sample from $p(\mathbf{s}_t^{(n)}|\mathbf{s}_{t-1}^{(n)})$ (repeat N times)
  - Reweight samples according to $\pi_t^{(n)} \leftarrow \pi_{t-1}^{(n)} p(\mathbf{I}_t|\mathbf{s}_t^{(n)})$

Particle filter (Condensation)

- For complex appearance likelihood functions, exact inferences are impossible

- Particle filters sample from $p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{I}_t)$ via *sequential importance sampling*:
  - Assume we have a set of N weighted samples $\{(\mathbf{s}_{t-1}^{(n)}, \pi_{t-1}^{(n)}), n = 1, \ldots, N\}$
  - Pick sample $\mathbf{s}_{t-1}^{(n)}$ according to weights, and sample from $p(\mathbf{s}_t^{(n)}|\mathbf{s}_{t-1}^{(n)})$ (repeat N times)
  - Reweight samples according to $\pi_t^{(n)} \leftarrow \pi_{t-1}^{(n)} p(\mathbf{I}_t|\mathbf{s}_t^{(n)})$
  - Renormalize samples by $\pi_t^{(n)} \leftarrow \frac{\pi_t^{(n)}}{\sum_{n=1}^{N} \pi_t^{(n)}}$

Particle filter (Condensation)

- For complex appearance likelihood functions, exact inferences are impossible

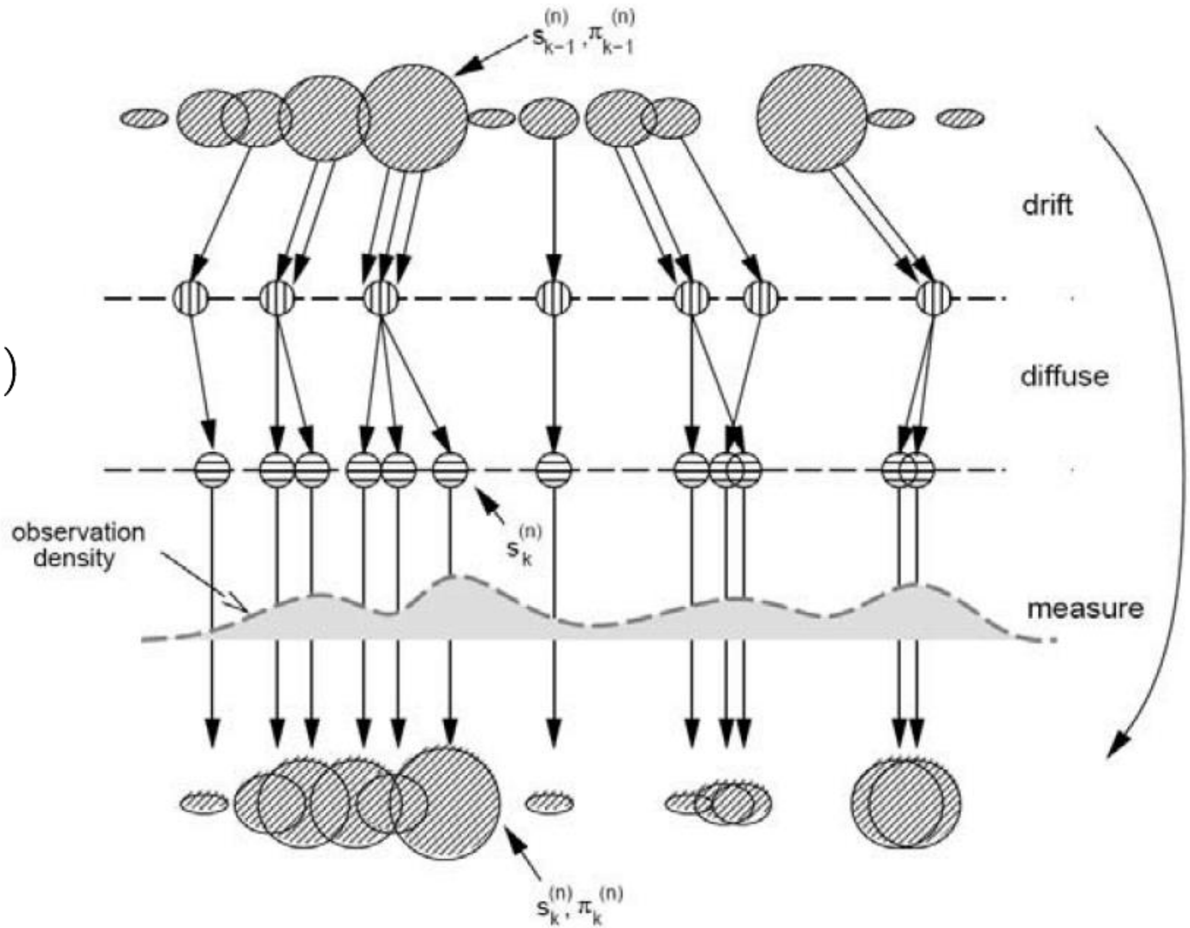
- Particle filters sample from $p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{I}_t)$ via *sequential importance sampling*:
  - Assume we have a set of N weighted samples $\{(\mathbf{s}_{t-1}^{(n)}, \pi_{t-1}^{(n)}), n = 1, \ldots, N\}$
  - Pick sample $\mathbf{s}_{t-1}^{(n)}$ according to weights, and sample from $p(\mathbf{s}_t^{(n)}|\mathbf{s}_{t-1}^{(n)})$ (repeat N times)
  - Reweight samples according to $\pi_t^{(n)} \leftarrow \pi_{t-1}^{(n)} p(\mathbf{I}_t|\mathbf{s}_t^{(n)})$
  - Renormalize samples by $\pi_t^{(n)} \leftarrow \frac{\pi_t^{(n)}}{\sum_{n=1}^{N} \pi_t^{(n)}}$
  - Prediction may be weighted average of particles, or most likely particle

Particle filter (Condensation)

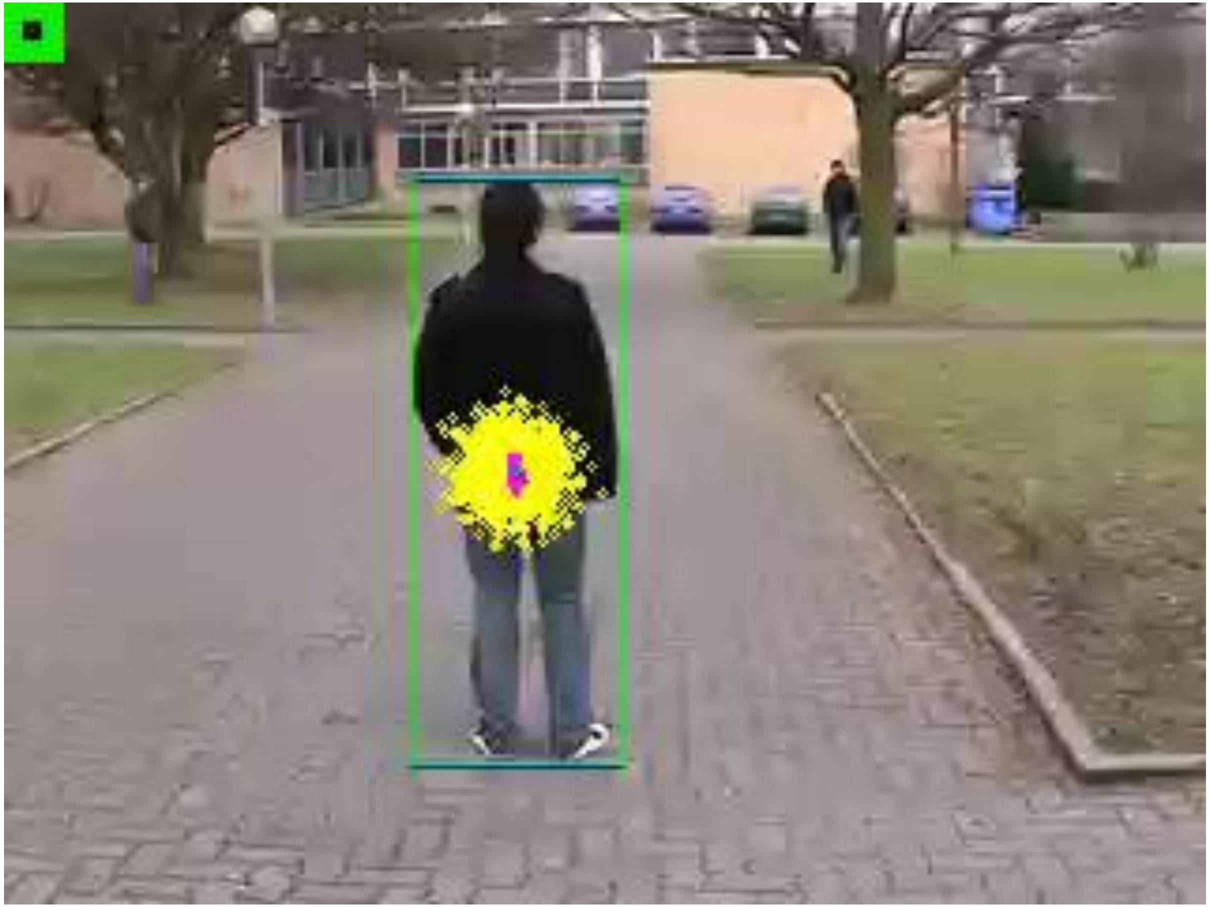
Pick sample $\mathbf{s}_{t-1}^{(n)}$ according to weights

Sample from $p(\mathbf{s}_t^{(n)}|\mathbf{s}_{t-1}^{(n)})$

Update weights $\pi_t^{(n)}$



Example: Particle filter



Tracking-by-detection

Tracking-by-detection

- Appearance likelihood: Your favorite object detector (V&J, D-T, *etc.*)
  - Now, we are using a *conditional likelihood*:

$$p(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{I}_t) \propto \exp\{s_L(\mathbf{x}_t, \mathbf{I}_t) + s_M(\mathbf{x}_t, \mathbf{x}_{t-1})\}$$

- Motion prior: May be virtually any motion model
- Search strategy: Sliding window detector at multiple scales (brute-force)

Tracking-by-detection

- Appearance likelihood: Your favorite object detector (V&J, D-T, *etc.*)
    - Now, we are using a *conditional likelihood*:

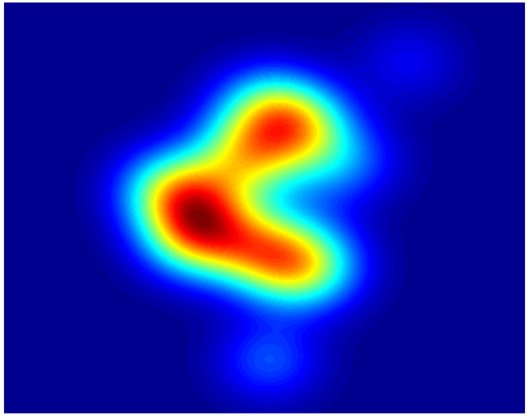
$$p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{I}_t) \propto \exp\left\{s_L(\mathbf{x}_t, \mathbf{I}_t) + s_M(\mathbf{x}_t, \mathbf{x}_{t-1})\right\}$$

- Motion prior: May be virtually any motion model
- Search strategy: Sliding window detector at multiple scales (brute-force)
- Potential problem: Object appearance may change over time

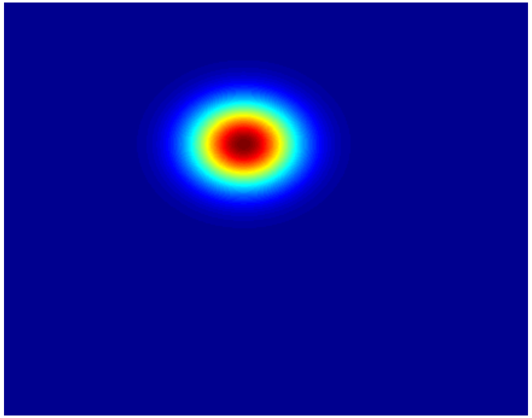
Track-learn-detect

Can we gather positive and negative examples to update appearance model?

- Positive example: Assume the track in the previous frame is correct
- Negative example: Example with high detection score but low motion prior



appearance likelihood

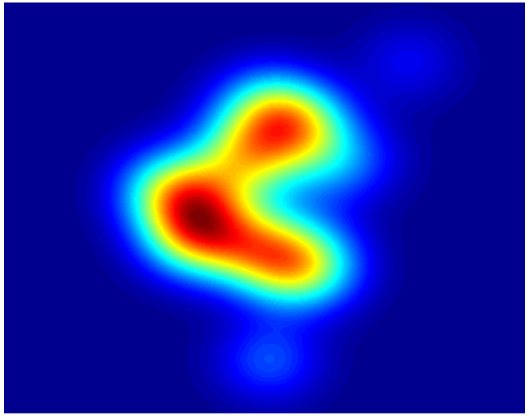


motion prior

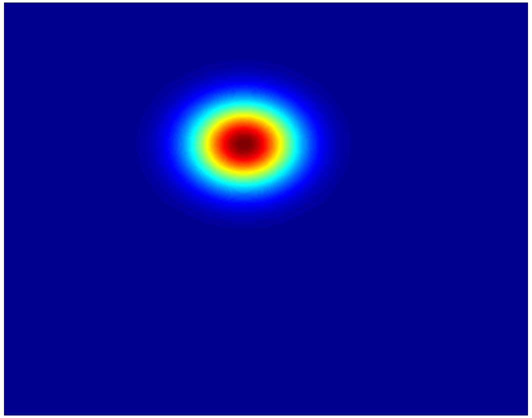
Track-learn-detect

Can we gather positive and negative examples to update appearance model?

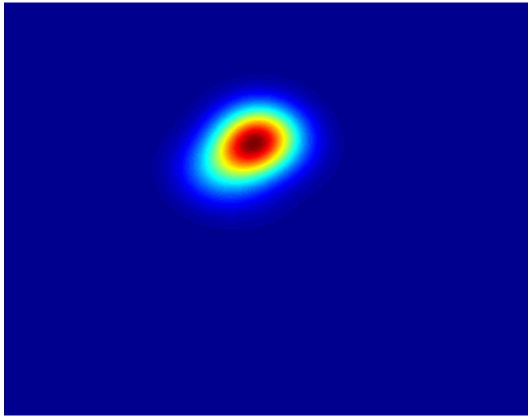
- Positive example: Assume the track in the previous frame is correct
- Negative example: Example with high detection score but low motion prior



appearance likelihood



motion prior

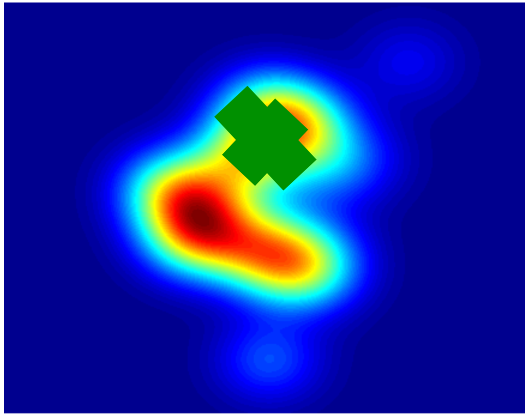


posterior over object location

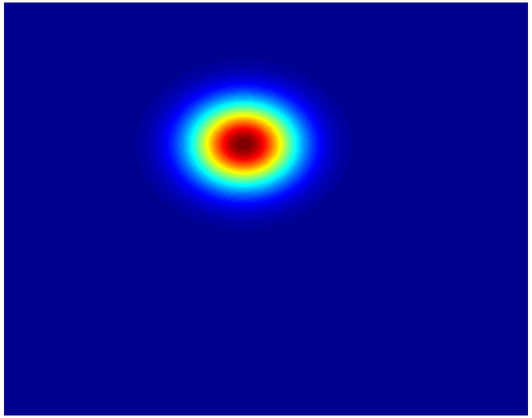
Track-learn-detect

Can we gather positive and negative examples to update appearance model?

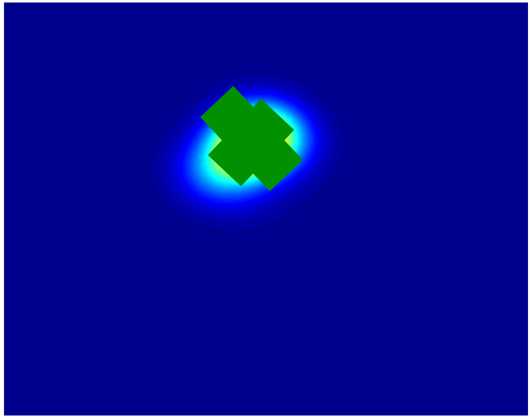
- Positive example: Assume the track in the previous frame is correct
- Negative example: Example with high detection score but low motion prior



appearance likelihood



motion prior

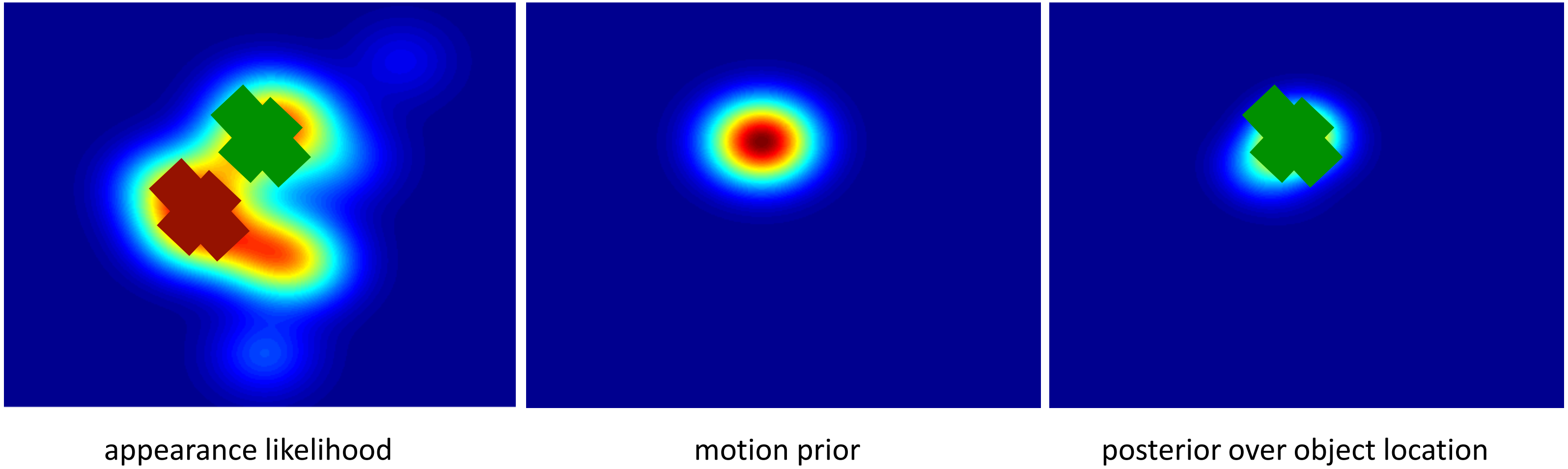


posterior over object location

Track-learn-detect

Can we gather positive and negative examples to update appearance model?

- Positive example: Assume the track in the previous frame is correct
- Negative example: Example with high detection score but low motion prior



Example: Track-learn-detect

Kanade-Lucas-Tomasi tracker

Kanade-Lucas-Tomasi Tracker

- Detect feature points in object using *Shi-Tomasi corner detector* (like Harris)

- Track feature points in the next frame by minimizing the squared error:

$$\sum_{n=1}^{N}\left[\mathbf{I}_{t-1}\left(\mathbf{x}_{t-1}^{(n)}\right)-\mathbf{I}_{t}\left(W\left(\mathbf{x}_{t-1}^{(n)};\mathbf{p}\right)\right)\right]^{2}$$

Kanade-Lucas-Tomasi Tracker

- Detect feature points in object using *Shi-Tomasi corner detector* (like Harris)

- Track feature points in the next frame by minimizing the squared error:

$$\sum_{n=1}^{N} \left[\mathbf{I}_{t-1}\left(\mathbf{x}_{t-1}^{(n)}\right) - \mathbf{I}_t\left(W\left(\mathbf{x}_{t-1}^{(n)};\mathbf{p}\right)\right)\right]^2$$

small image patch around
feature point at time *t-1*

Kanade-Lucas-Tomasi Tracker

- Detect feature points in object using *Shi-Tomasi corner detector* (like Harris)
- Track feature points in the next frame by minimizing the squared error:

$$\sum_{n=1}^{N} \left[\mathbf{I}_{t-1}\left(\mathbf{x}_{t-1}^{(n)}\right) - \mathbf{I}_t\left(W\left(\mathbf{x}_{t-1}^{(n)};\mathbf{p}\right)\right)\right]^2$$

small image patch around
feature point at time *t-1*

small image patch around
warped feature point at time *t*

Kanade-Lucas-Tomasi Tracker

- Detect feature points in object using *Shi-Tomasi corner detector* (like Harris)

- Track feature points in the next frame by minimizing the squared error:

$$\sum_{n=1}^{N}\left[\mathbf{I}_{t-1}\left(\mathbf{x}_{t-1}^{(n)}\right)-\mathbf{I}_t\left(W\left(\mathbf{x}_{t-1}^{(n)};\mathbf{p}\right)\right)\right]^2$$

small image patch around feature point at time *t-1*

small image patch around warped feature point at time *t*

- Note that this is yet another (non)linear least squares minimization problem!

Kanade-Lucas-Tomasi Tracker

- Detect feature points in object using *Shi-Tomasi corner detector* (like Harris)

- Track feature points in the next frame by minimizing the squared error:

$$\sum_{n=1}^{N} \left[\mathbf{I}_{t-1}\left(\mathbf{x}_{t-1}^{(n)}\right) - \mathbf{I}_t\left(W\left(\mathbf{x}_{t-1}^{(n)};\mathbf{p}\right)\right)\right]^2$$

small image patch around feature point at time *t-1*

small image patch around warped feature point at time *t*

- Note that this is yet another (non)linear least squares minimization problem!
    - The minimization can be performing using the Lucas-Kanade algorithm

Example: Kanade-Lucas-Tomasi Tracker

Reading material:

- M. Isard and A. Blake. "Condensation—Conditional Density Propagation for Visual Tracking." International Journal of Computer Vision 29(1), 5-28, 1998.
- Section 1 and 2 of “Lucas-Kanade 20 Years On: A Unifying Framework,” International Journal of Computer Vision 56(3), 221–255, 2004.