Computer Vision 2
WS 2018/19

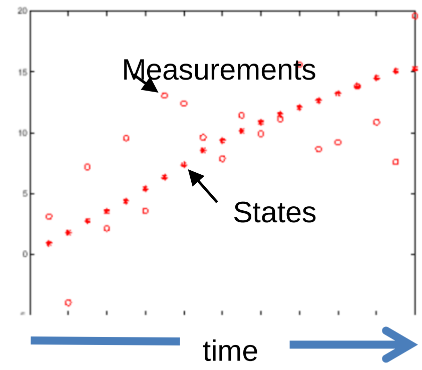
Part 7 – Tracking with Linear Dynamic Models
07.11.2018

Prof. Dr. Bastian Leibe

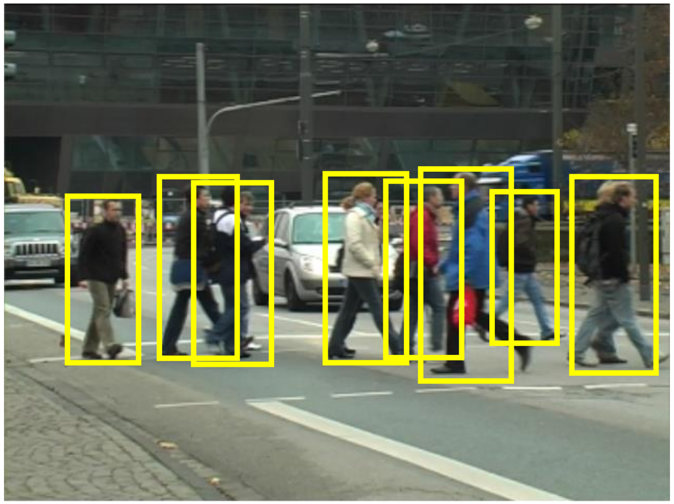
RWTH Aachen University, Computer Vision Group
http://www.vision.rwth-aachen.de

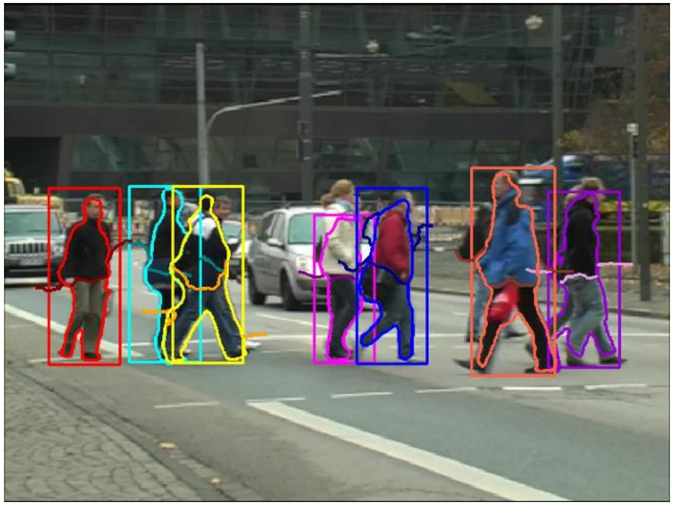
Course Outline

- Single-Object Tracking
- Bayesian Filtering
  - Kalman Filters, EKF
  - Particle Filters
- Multi-Object Tracking
- Visual Odometry
- Visual SLAM & 3D Reconstruction
- Deep Learning for Video Analysis



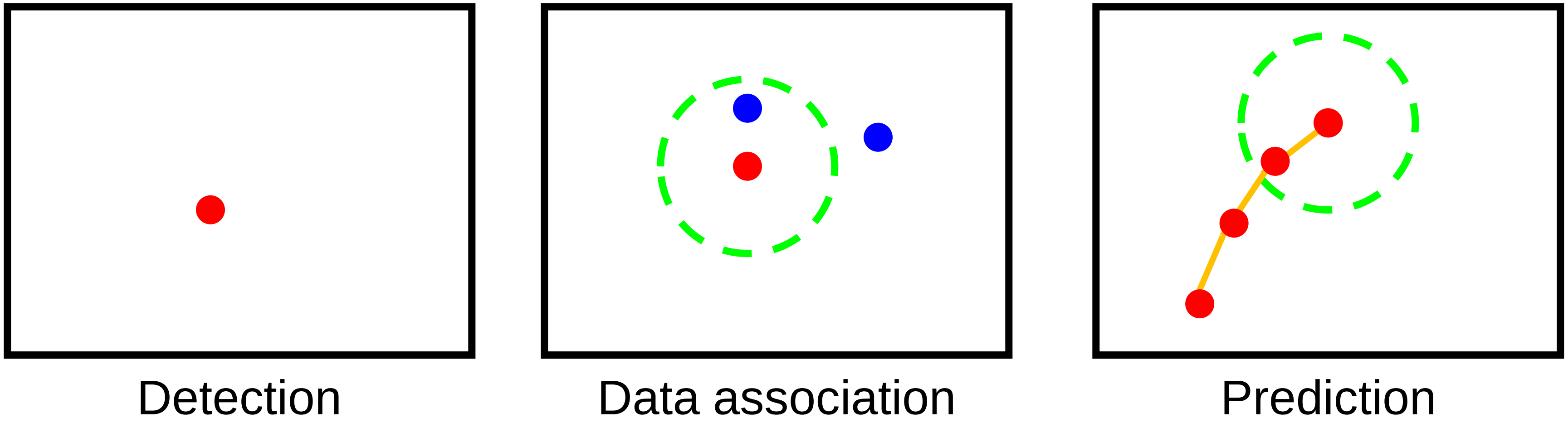
Recap: Tracking-by-Detection





- Main ideas
  - Apply a generic object detector to find objects of a certain class
  - Based on the detections, extract object appearance models
  - Link detections into trajectories

Recap: Elements of Tracking



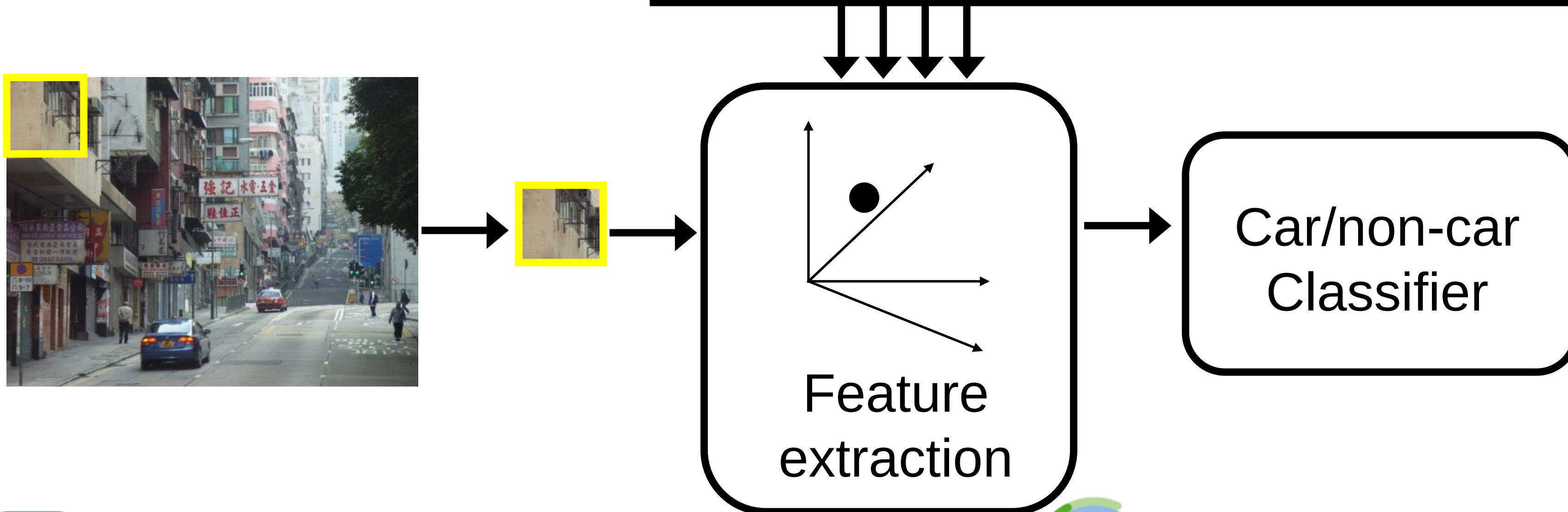
- Detection
  - *Where are candidate objects?*

Last lecture

- Data association
  - *Which detection corresponds to which object?*
- Prediction
  - *Where will the tracked object be in the next time step?*

Recap: Sliding-Window Object Detection

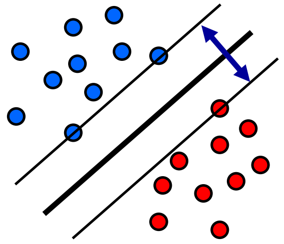
- For sliding-window object detection, we need to:
  1. Obtain training data
  2. Define features
  3. Define a classifier



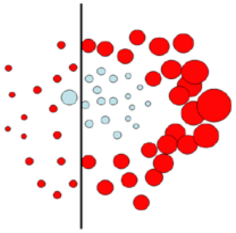
Slide credit: Kristen Grauman

Recap: Object Detector Design

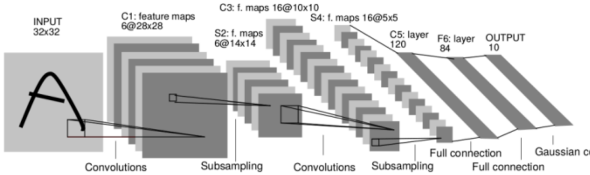
- In practice, the classifier often determines the design.
  - Types of features
  - Speedup strategies
- We looked at 3 state-of-the-art detector designs
  - Based on SVMs



  - Based on Boosting

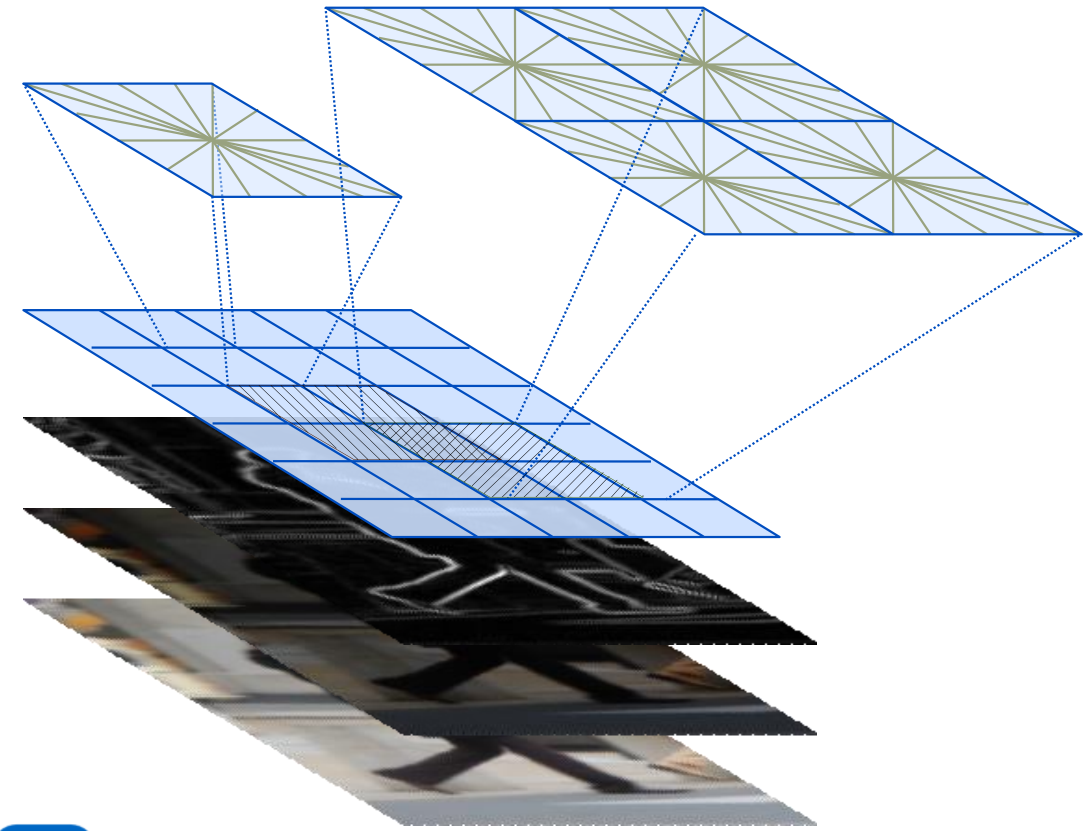


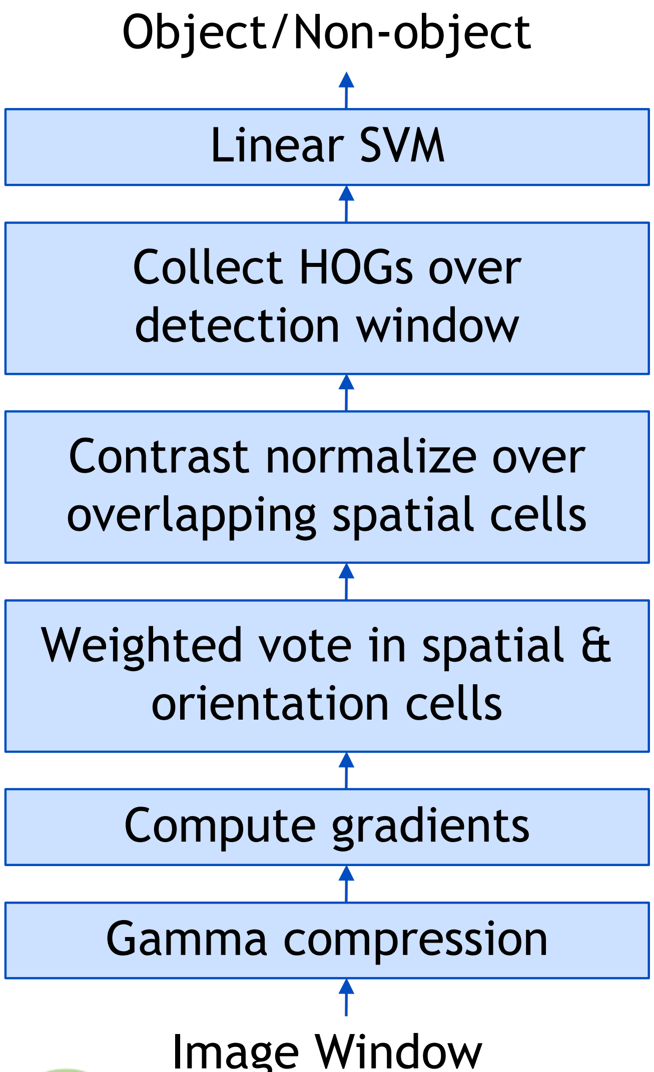
  - Based on CNNs



Recap: Histograms of Oriented Gradients (HOG)

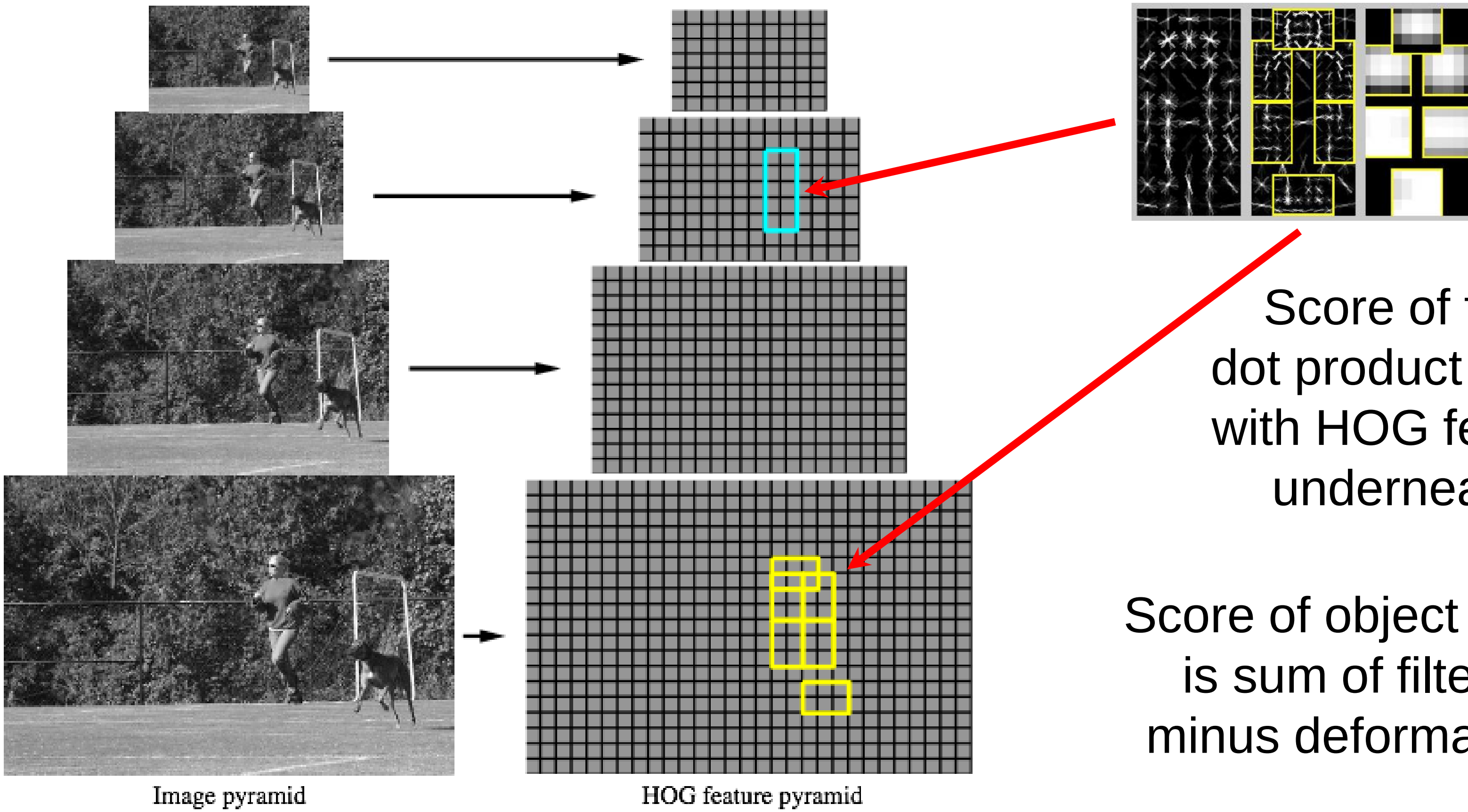
- Holistic object representation
  - Localized gradient orientations





Slide adapted from Navneet Dalal

Recap: Deformable Part-based Model (DPM)



Score of filter:
dot product of filter
with HOG features
underneath it

Score of object hypothesis
is sum of filter scores
minus deformation costs

- Multiscale model captures features at two resolutions

[Felzenszwalb, McAllister, Ramanan, CVPR'08]

Slide credit: Pedro Felzenszwalb

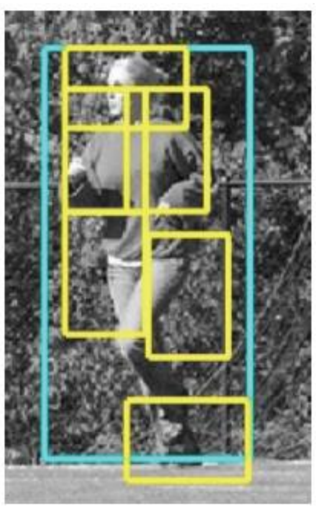
Recap: DPM Hypothesis Score

$$\text{score}(p_0, \ldots, p_n) = \underbrace{\sum_{i=0}^{n} F_i \cdot \phi(H, p_i)}_{\text{“data term”}} - \underbrace{\sum_{i=1}^{n} d_i \cdot (dx_i^2, dy_i^2)}_{\text{“spatial prior”}}$$

filters

displacements

deformation parameters



$$\text{score}(z) = \beta \cdot \Psi(H, z)$$

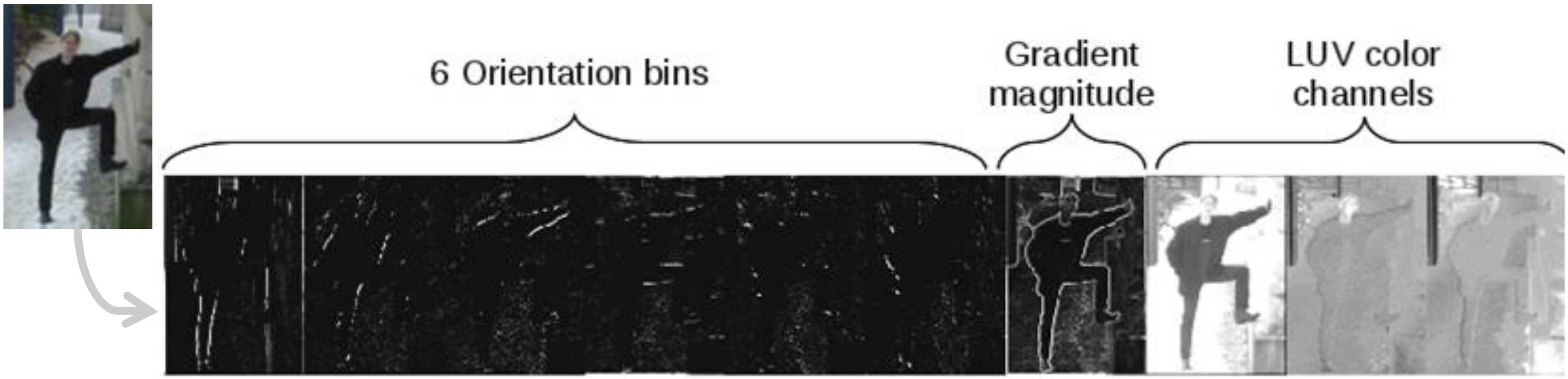
concatenation filters and deformation parameters

concatenation of HOG features and part displacement features

[Felzenszwalb, McAllister, Ramanan, CVPR’08]

Slide credit: Pedro Felzenszwalb

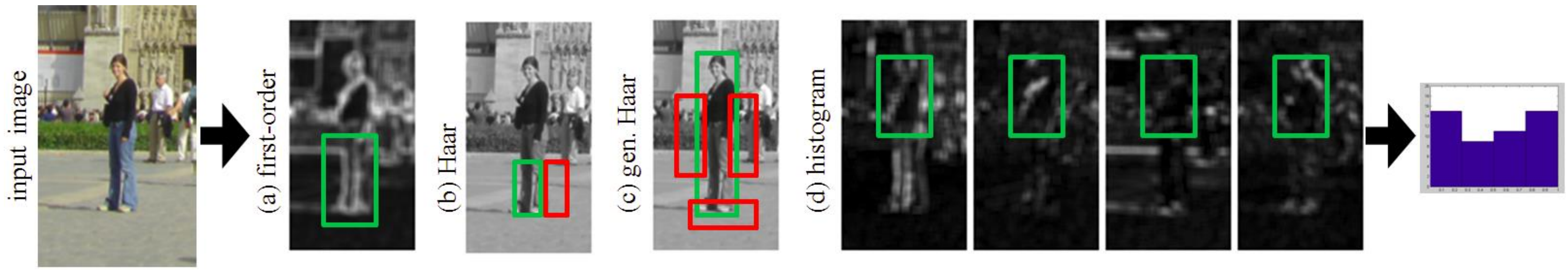
Recap: Integral Channel Features



- Generalization of Haar Wavelet idea from Viola-Jones
  - Instead of only considering intensities, also take into account other feature channels (gradient orientations, color, texture).
  - Still efficiently represented as integral images.

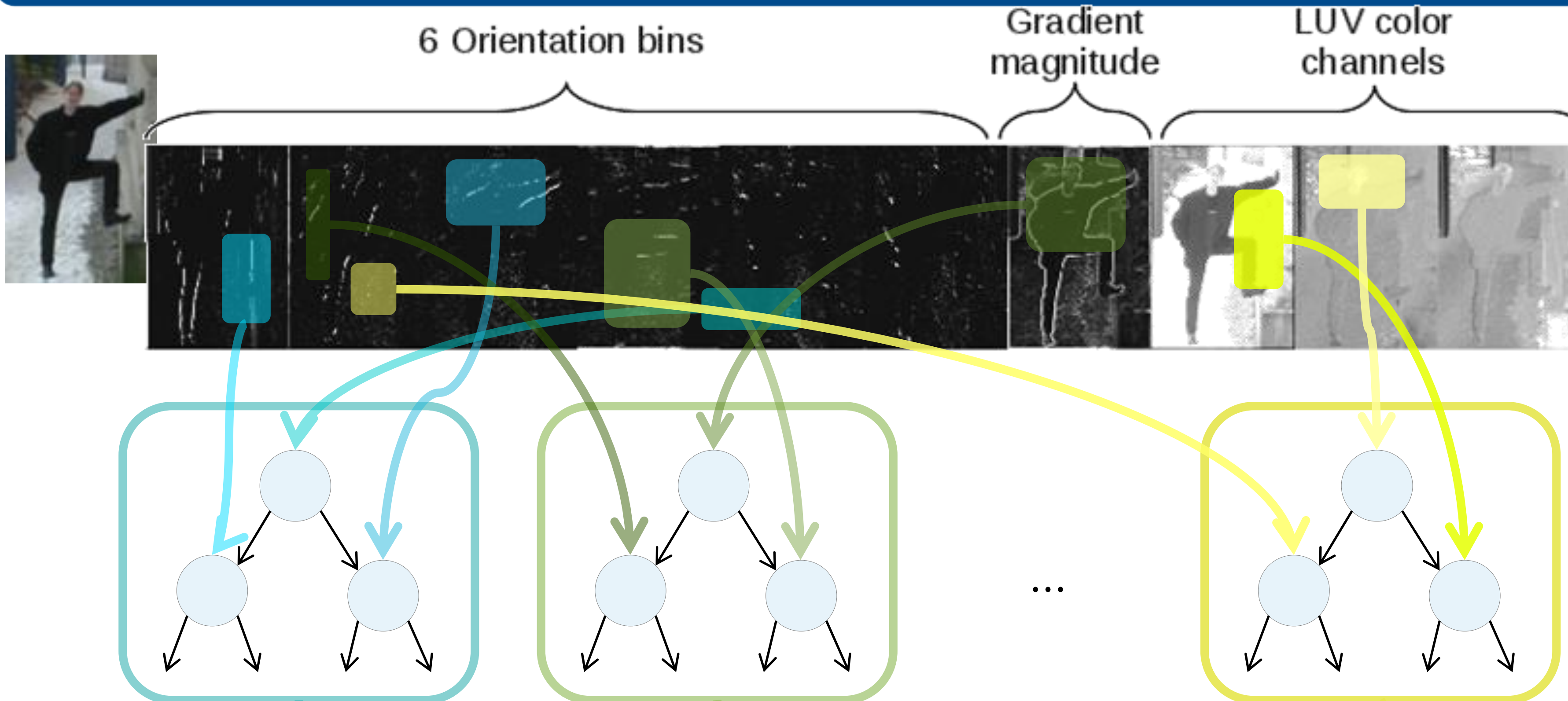
P. Dollar, Z. Tu, P. Perona, S. Belongie. Integral Channel Features, BMVC'09.

Recap: Integral Channel Features



- Generalize also block computation
  - 1st order features:
    - Sum of pixels in rectangular region.
  - 2nd-order features:
    - Haar-like difference of sum-over-blocks
  - Generalized Haar:
    - More complex combinations of weighted rectangles
  - Histograms
    - Computed by evaluating local sums on quantized images.

Recap: VeryFast Classifier Construction

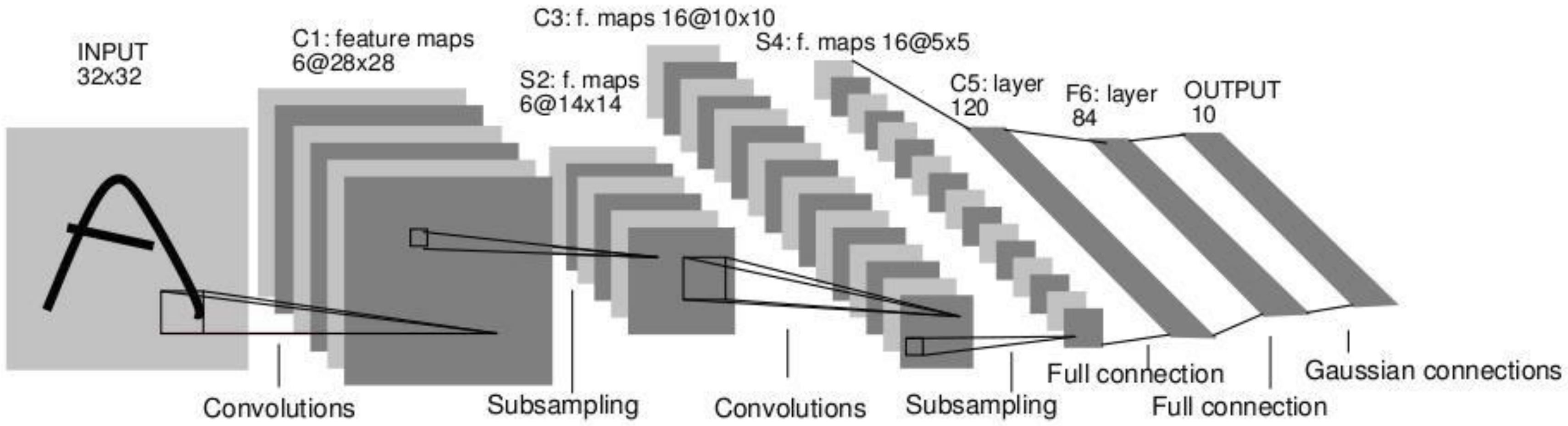


$$score = w_1 \cdot \boxed{h_1} + w_2 \cdot \boxed{h_2} + \dots + w_N \cdot \boxed{h_N}$$

- Ensemble of short trees, learned by AdaBoost

Slide credit: Rodrigo Benenson

Recap: Convolutional Neural Networks

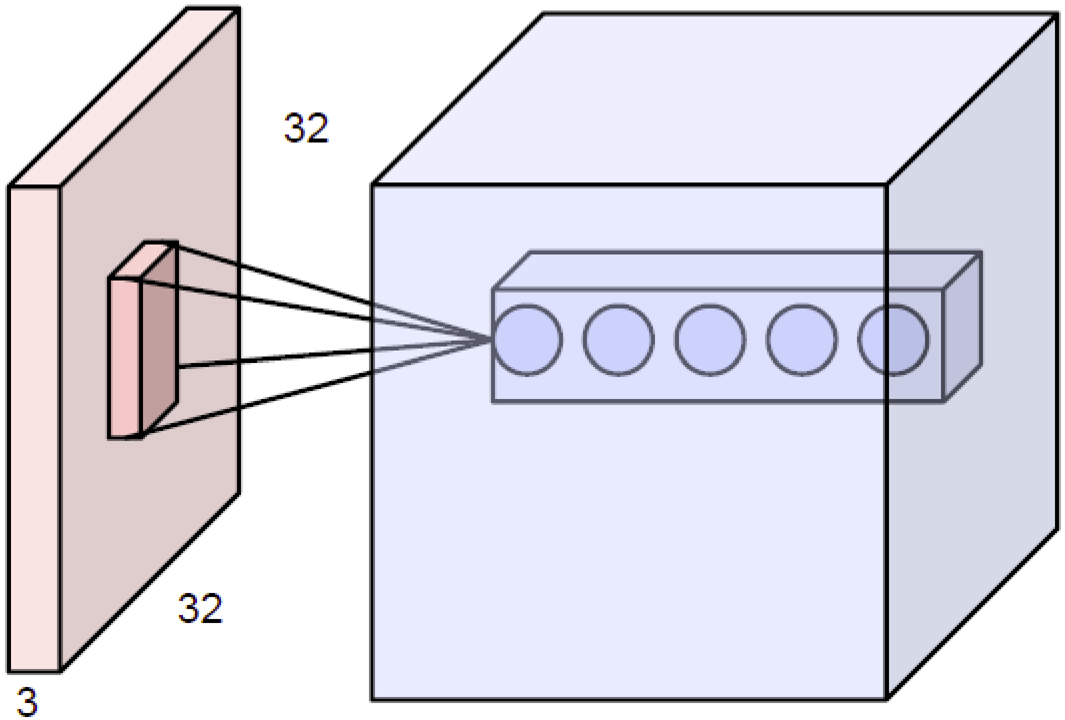


- Neural network with specialized connectivity structure
  - Stack multiple stages of feature extractors
  - Higher stages compute more global, more invariant features
  - Classification layer at the end

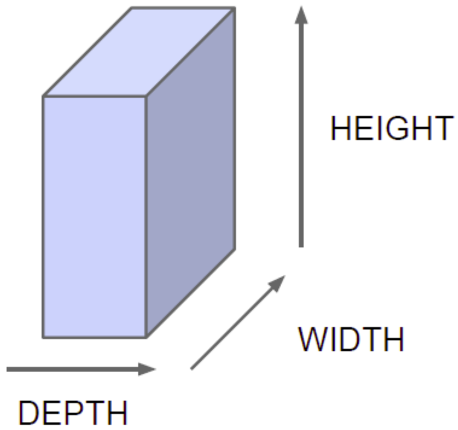
Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE 86(11): 2278–2324, 1998.

Slide credit: Svetlana Lazebnik

Recap: Convolution Layers



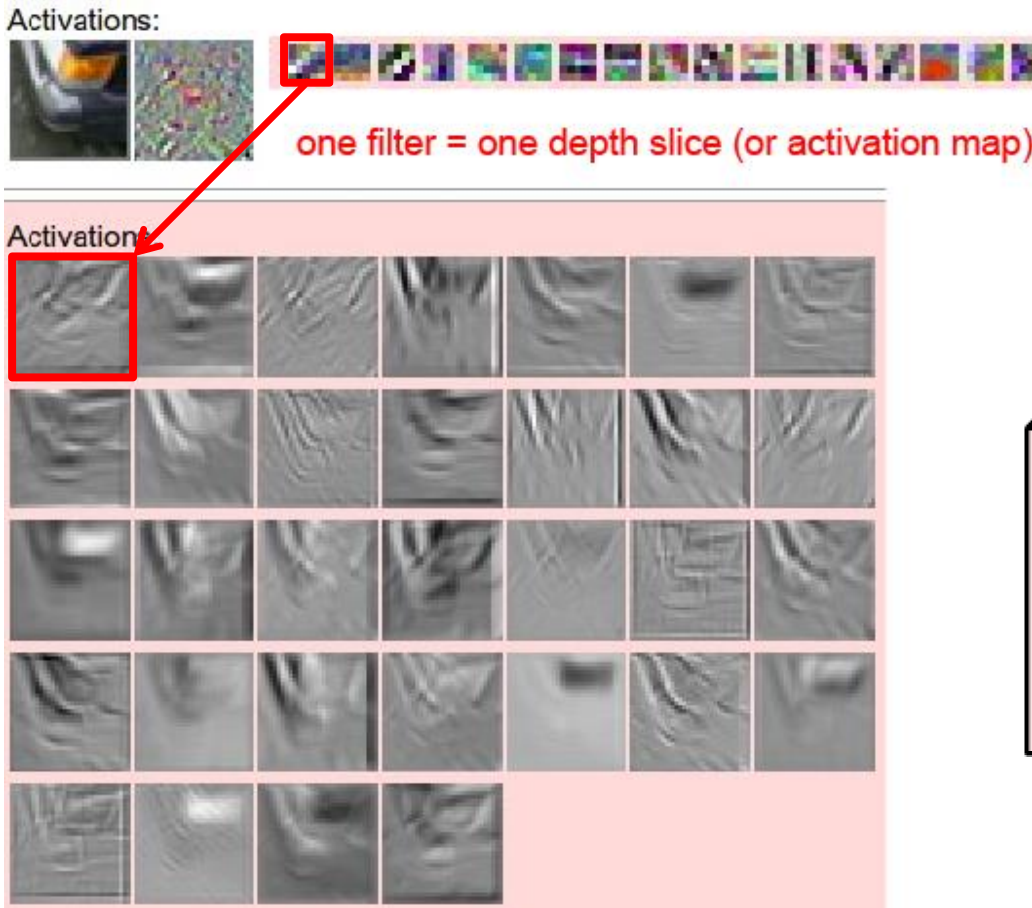
Naming convention:



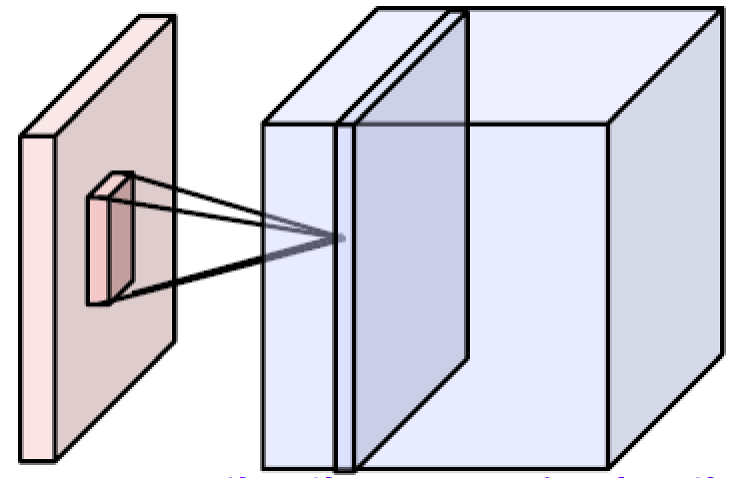
- All Neural Net activations arranged in 3 dimensions
  - Multiple neurons all looking at the same input region, stacked in depth
  - Form a single [$1 \times 1 \times$depth] depth column in output volume.

Slide credit: FeiFei Li, Andrej Karpathy

Recap: Activation Maps



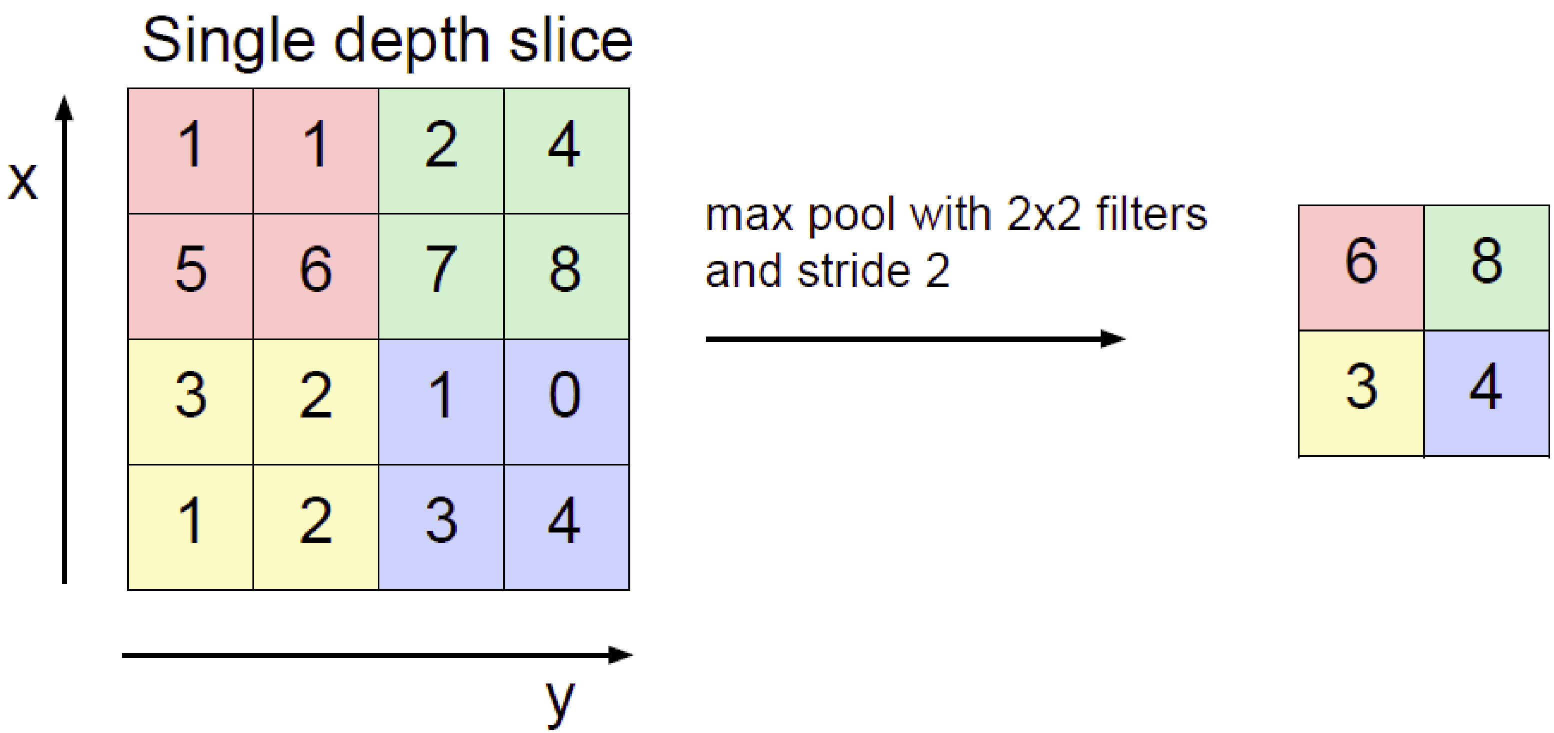
5×5 filters



Activation maps

Slide adapted from FeiFei Li, Andrej Karpathy

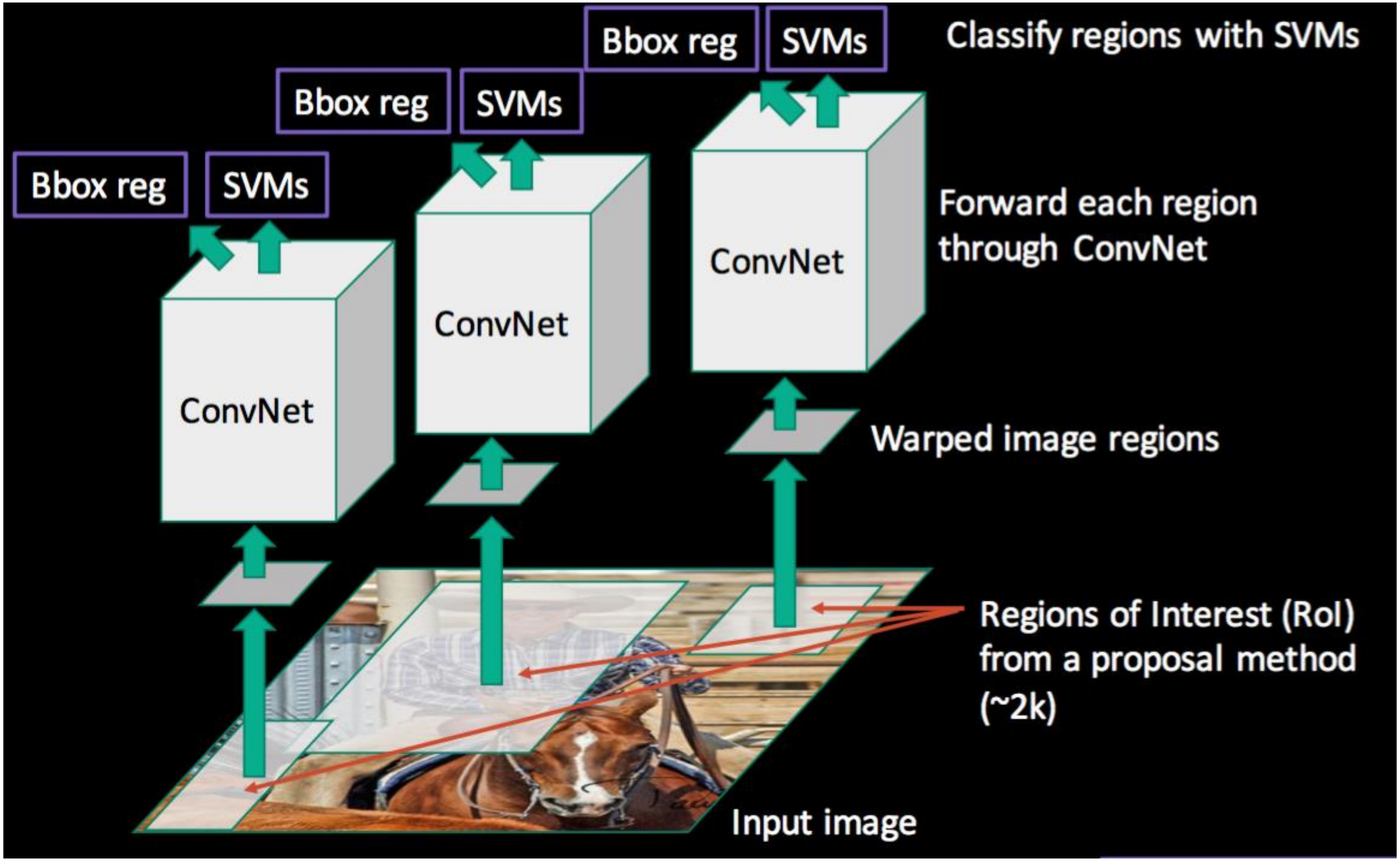
Recap: Pooling Layers



- Effect:
  - Make the representation smaller without losing too much information
  - Achieve robustness to translations

Slide adapted from FeiFei Li, Andrej Karpathy

Recap: R-CNN for Object Detection

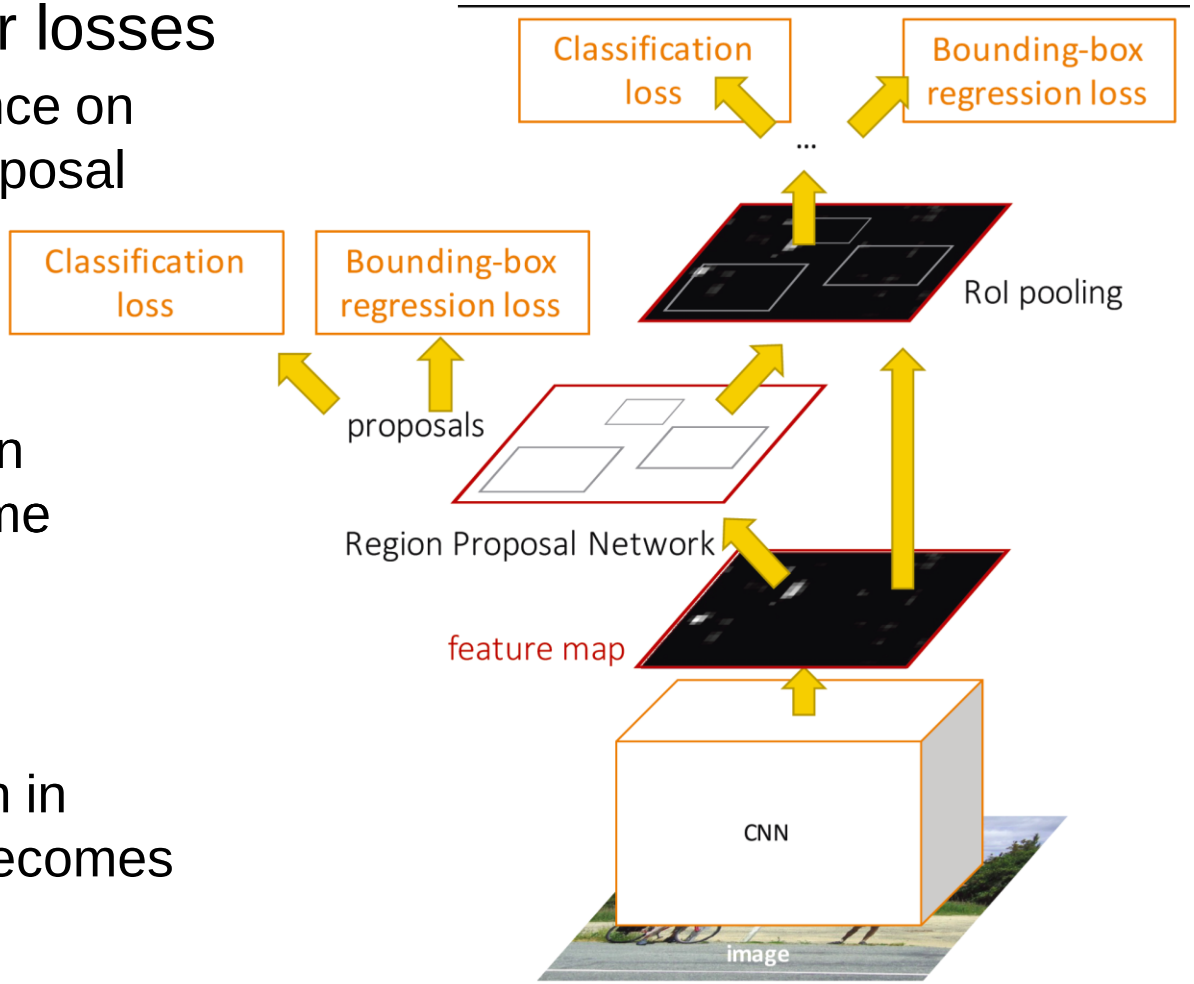


Slide credit: Ross Girshick

Recap: Faster R-CNN

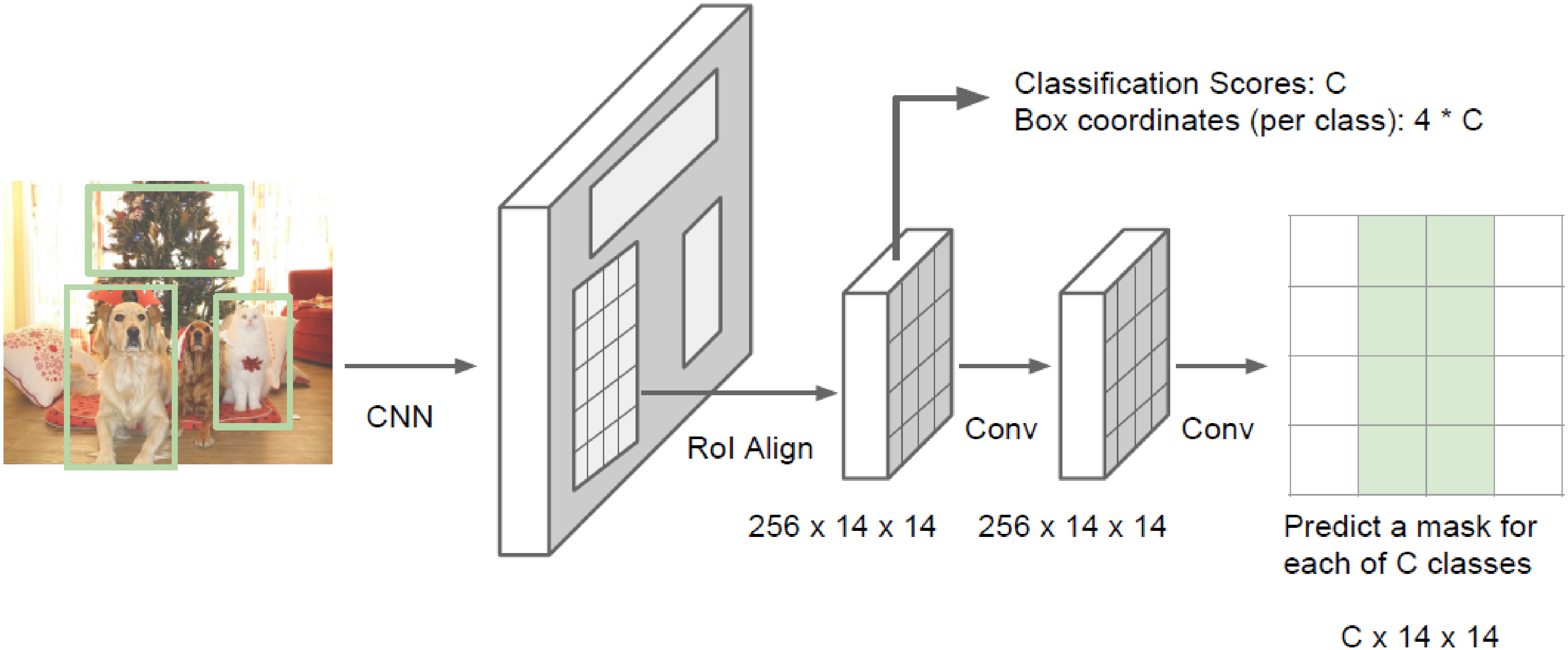
- One network, four losses
  - Remove dependence on external region proposal algorithm.
  - Instead, infer region proposals from same CNN.
  - Feature sharing
  - Joint training

    ⇒ Object detection in a single pass becomes possible.



Slide credit: Ross Girshick

Most Recent Version: Mask R-CNN

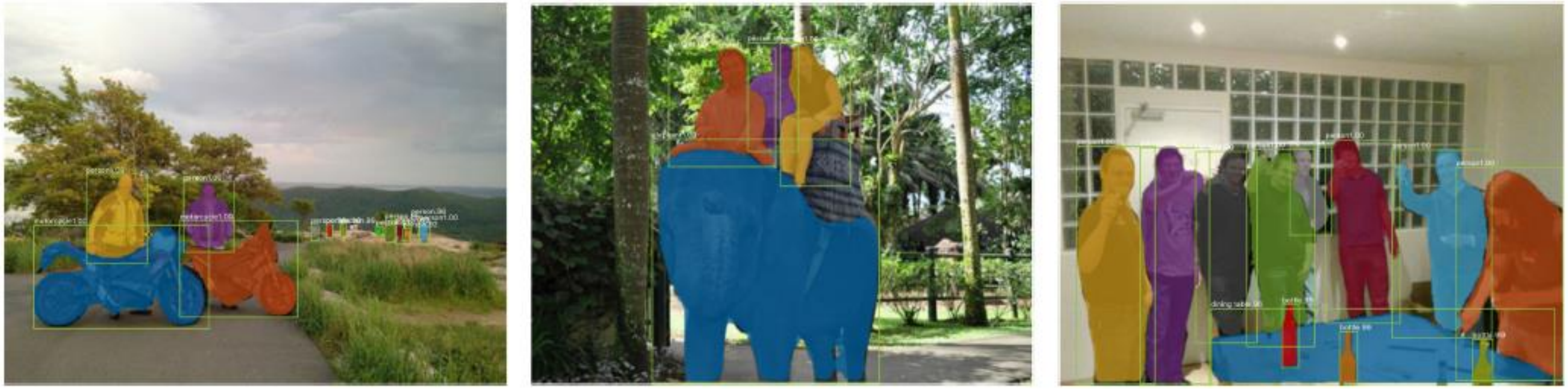


K. He, G. Gkioxari, P. Dollar, R. Girshick, [Mask R-CNN](), arXiv 1703.06870.

Slide credit: FeiFei Li

Mask R-CNN Results

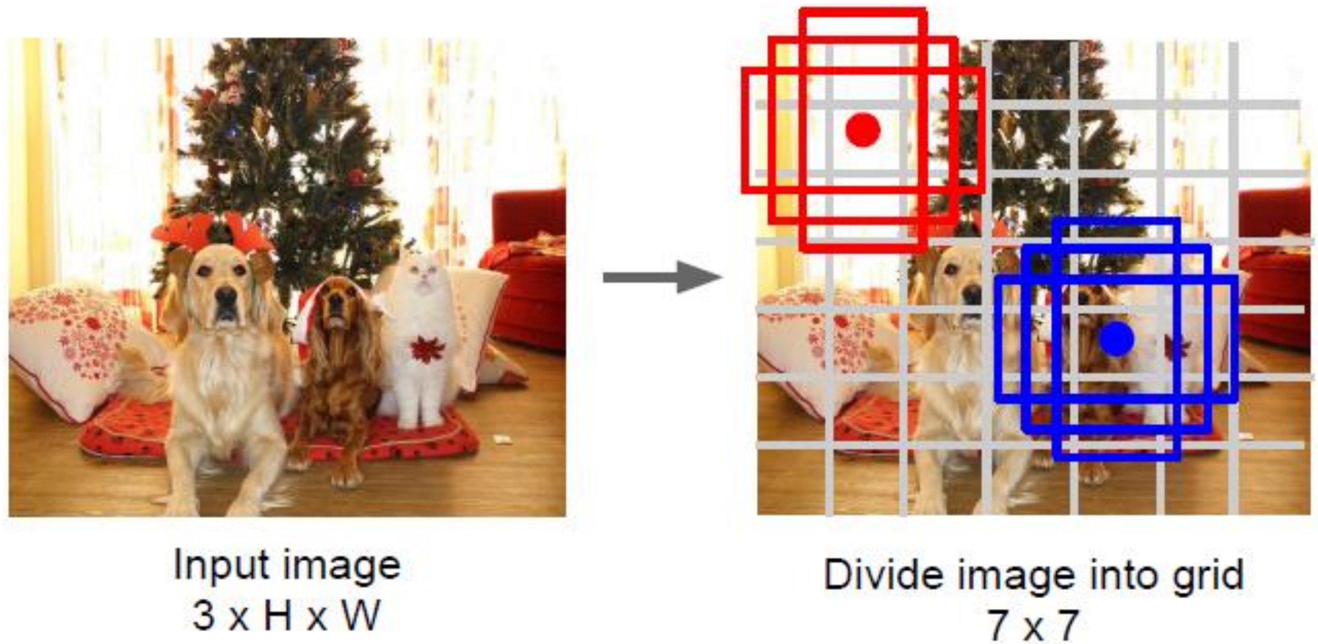
- Detection + Instance segmentation



- Detection + Pose estimation

Figure credit: K. He, G. Gkioxari, P. Dollar, R. Girshick

YOLO / SSD



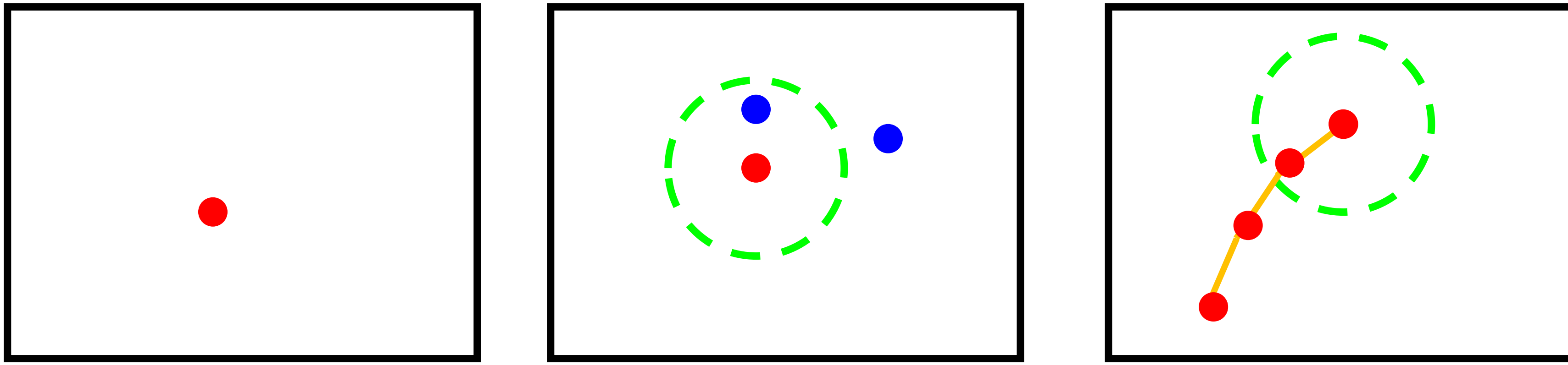
- Idea: Directly go from image to detection scores
- Within each grid cell
  - Start from a set of anchor boxes
  - Regress from each of the B anchor boxes to a final box
  - Predict scores for each of C classes (including background)

Slide credit: FeiFei Li

You Can Try All of This At Home…

- Detector code is publicly available
  - HOG:
    - Dalal's original implementation: http://www.navneetdalal.com/software/
    - Our CUDA-optimized *groundHOG* code (>80 fps on GTX 580) http://www.vision.rwth-aachen.de/software/groundhog
  - DPM:
    - Felzenswalb's original implementation: http://www.cs.uchicago.edu/~pff/latent
  - VeryFast
    - Benenson's original implementation: https://bitbucket.org/rodrigob/doppia/
  - YOLO
    - Joe Redmon's original implementation (YOLO v3): https://pjreddie.com/darknet/yolo/

Recap: Elements of Tracking



Detection Data association Prediction

- Detection
  - *Where are candidate objects?*

Last lecture

- Data association
  - *Which detection corresponds to which object?*
- Prediction
  - *Where will the tracked object be in the next time step?*

Today's topic

Today: Tracking with Linear Dynamic Models

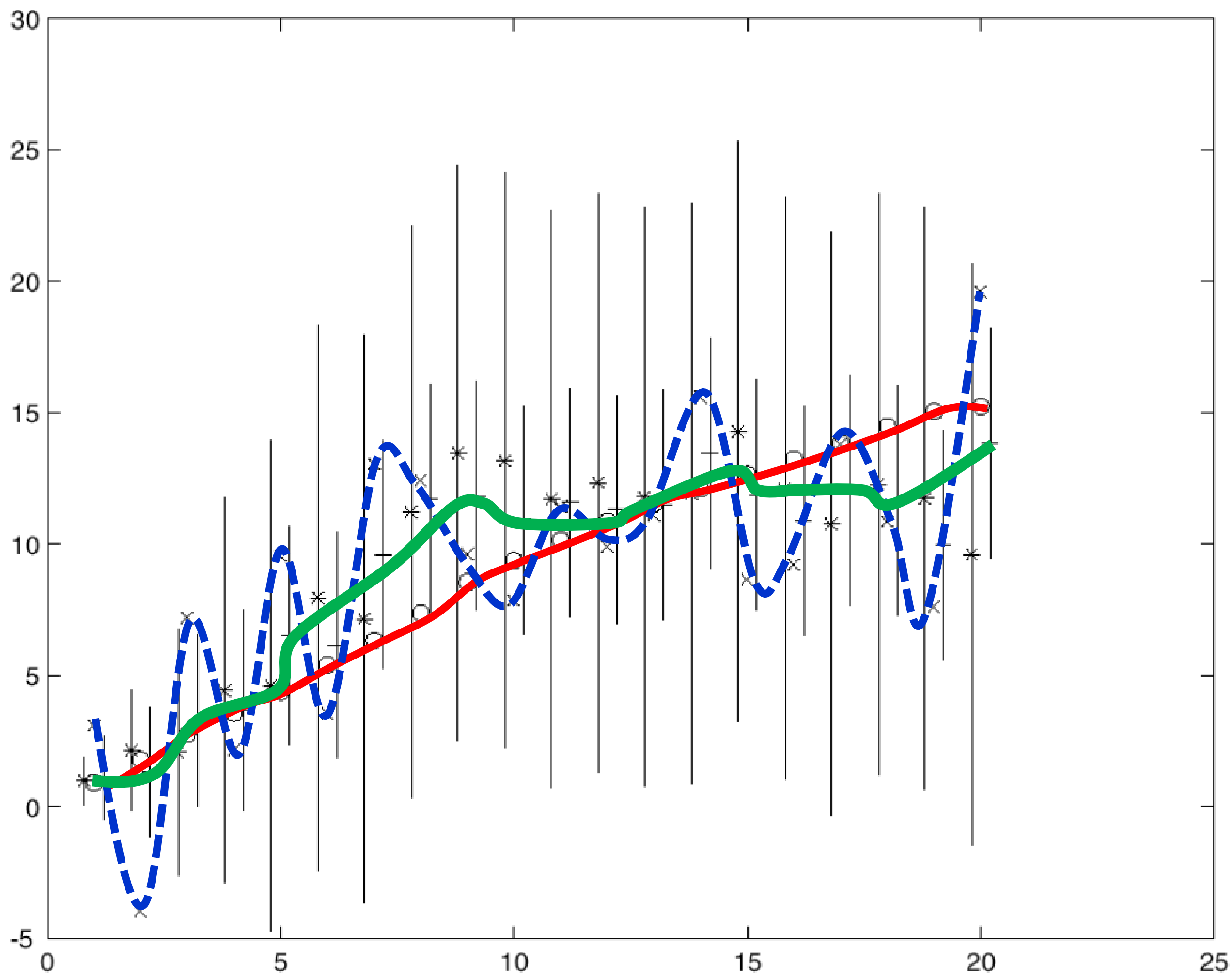


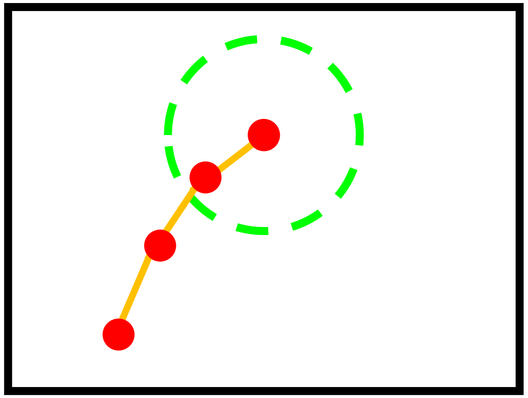
Figure from Forsyth & Ponce

Topics of This Lecture

- Tracking with Dynamics
  - Detection vs. Tracking
  - Tracking as probabilistic inference
  - Prediction and Correction
- Linear Dynamic Models
  - Zero velocity model
  - Constant velocity model
  - Constant acceleration model
- The Kalman Filter
  - Kalman filter for 1D state
  - General Kalman filter
  - Limitations

Tracking with Dynamics

- Key idea
  – Given a model of expected motion, predict where objects will occur in next frame, even before seeing the image.

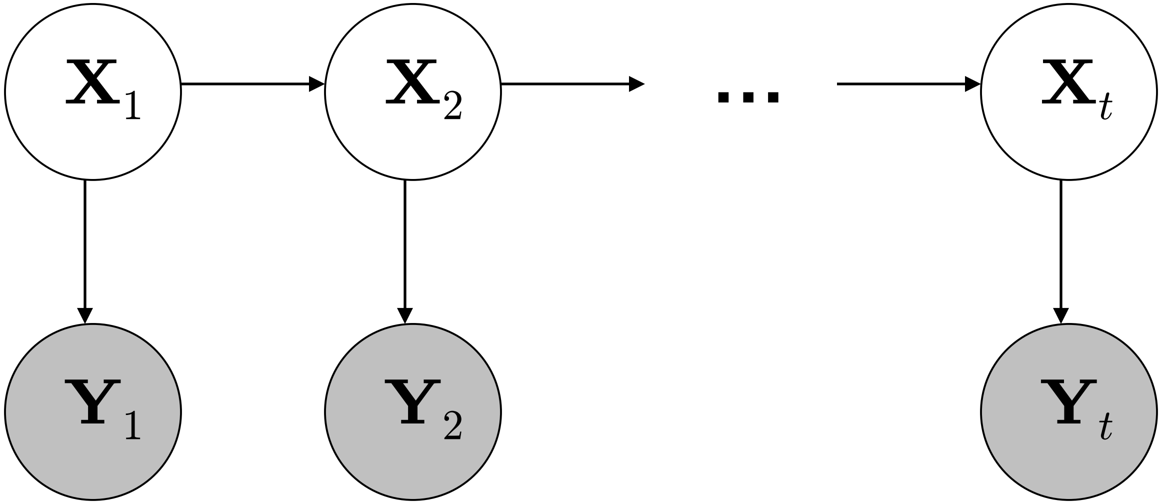


- Goals
  – Restrict search for the object
  – Improved estimates since measurement noise is reduced by trajectory smoothness.
- Assumption: continuous motion patterns
  – Camera is not moving instantly to new viewpoint.
  – Objects do not disappear and reappear in different places.
  – Gradual change in pose between camera and scene.

Slide adapted from S. Lazebnik, K. Grauman

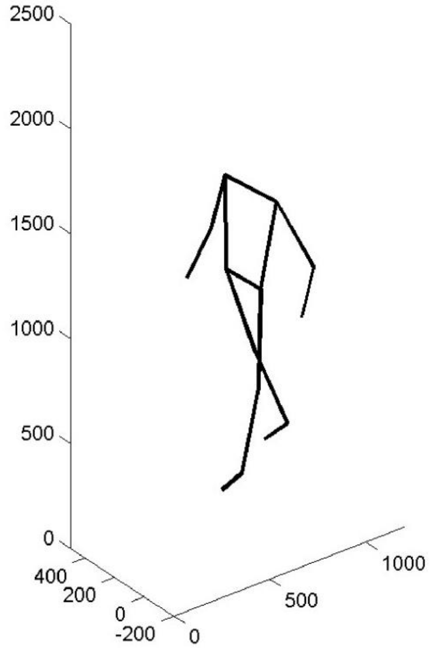
General Model for Tracking

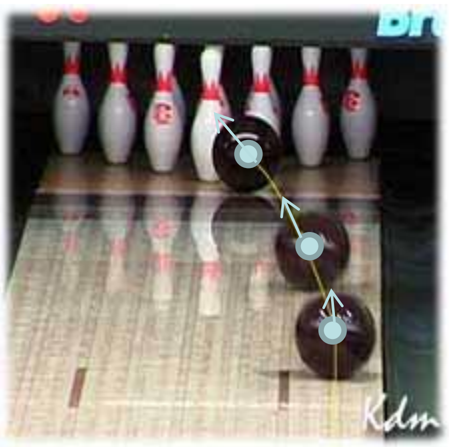
- Representation
  - The moving object of interest is characterized by an underlying *state* $\mathbf{X}$.
  - State $\mathbf{X}$ gives rise to *measurements* or *observations* $\mathbf{Y}$.
  - At each time t, the state changes to $\mathbf{X}_t$ and we get a new observation $\mathbf{Y}_t$.

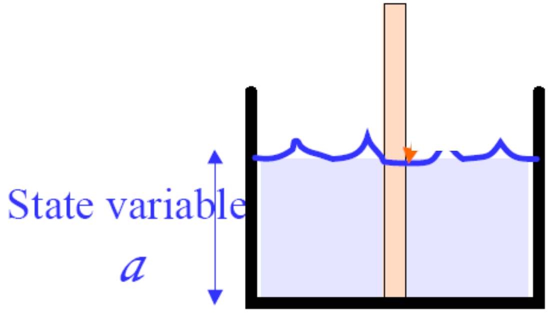


Slide credit: Svetlana Lazebnik

State vs. Observation







- Hidden state : parameters of interest
- Measurement: what we get to directly observe

Slide credit: Kristen Grauman

Tracking as Inference

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

Slide credit: Kristen Grauman

Steps of Tracking

- Prediction:
  - What is the next state of the object given past measurements?

$$P\left(X_t \middle| Y_0 = y_0, \ldots, Y_{t-1} = y_{t-1}\right)$$

- Correction:
  - Compute an updated estimate of the state from prediction and measurements.

$$P\left(X_t \middle| Y_0 = y_0, \ldots, Y_{t-1} = y_{t-1}, Y_t = y_t\right)$$

- Tracking
  - Can be seen as the process of propagating the posterior distribution of state given measurements across time.

Simplifying Assumptions

- Only the immediate past matters

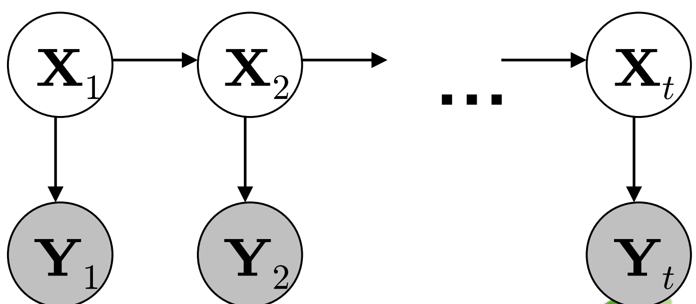
$$P(X_t \mid X_0, \ldots, X_{t-1}) = \boxed{P(X_t \mid X_{t-1})}$$

Dynamics model

- Measurements depend only on the current state

$$P(Y_t \mid X_0, Y_0 \ldots, X_{t-1}, Y_{t-1}, X_t) = \boxed{P(Y_t \mid X_t)}$$

Observation model



Slide credit: Svetlana Lazebnik

Tracking as Induction

- Base case:
  - Assume we have initial prior that *predicts* state in absence of any evidence: $P(\mathbf{X}_0)$
  - At the first frame, *correct* this given the value of $\mathbf{Y}_0 = \mathbf{y}_0$

$$P(X_0 \mid Y_0 = y_0) = \frac{P(y_0 \mid X_0) P(X_0)}{P(y_0)} \propto P(y_0 \mid X_0) P(X_0)$$

Posterior prob. of state given measurement

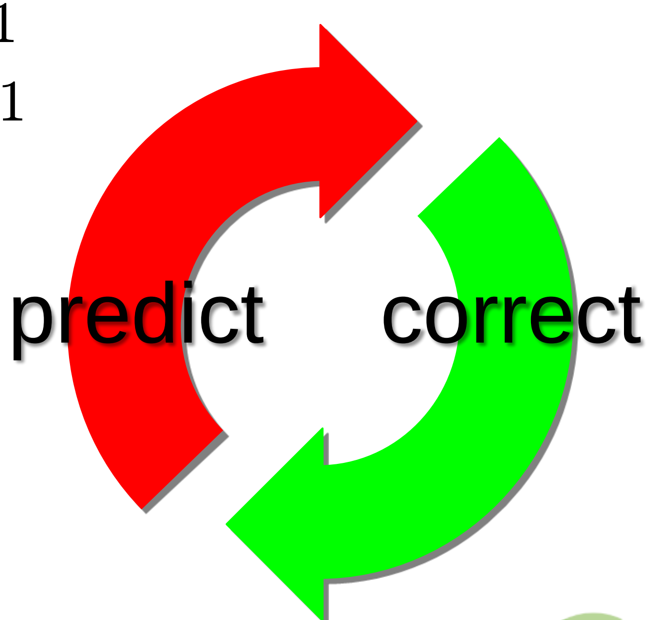
Likelihood of measurement

Prior of the state

Slide credit: Svetlana Lazebnik

Tracking as Induction

- Base case:
  - Assume we have initial prior that *predicts* state in absence of any evidence: $P(\mathbf{X}_0)$
  - At the first frame, *correct* this given the value of $\mathbf{Y}_0 = \mathbf{y}_0$
- Given corrected estimate for frame t:
  - Predict for frame $t+1$
  - Correct for frame $t+1$



Slide credit: Svetlana Lazebnik

Induction Step: Prediction

- Prediction involves representing $P\left(X_t \middle| y_0, \ldots, y_{t-1}\right)$ given $P\left(X_{t-1} \middle| y_0, \ldots, y_{t-1}\right)$

$$P\left(X_t \middle| y_0, \ldots, y_{t-1}\right)$$
$$= \int P\left(X_t, X_{t-1} \middle| y_0, \ldots, y_{t-1}\right) dX_{t-1}$$

Law of total probability

$$\boxed{P(A) = \int P(A, B)\, dB}$$

Slide credit: Svetlana Lazebnik

Induction Step: Prediction

- Prediction involves representing $P(X_t | y_0, \ldots, y_{t-1})$ given $P(X_{t-1} | y_0, \ldots, y_{t-1})$

$$P(X_t | y_0, \ldots, y_{t-1})$$
$$= \int P(X_t, X_{t-1} | y_0, \ldots, y_{t-1}) dX_{t-1}$$
$$= \int P(X_t \mid X_{t-1}, y_0, \ldots, y_{t-1}) P(X_{t-1} \mid y_0, \ldots, y_{t-1}) dX_{t-1}$$

Conditioning on X_{t-1}

$$\boxed{P(A, B) = P(A | B) P(B)}$$

Slide credit: Svetlana Lazebnik

Induction Step: Prediction

- Prediction involves representing $P(X_t | y_0, \ldots, y_{t-1})$ given $P(X_{t-1} | y_0, \ldots, y_{t-1})$

$$
\begin{aligned}
&P(X_t | y_0, \ldots, y_{t-1}) \\
&= \int P(X_t, X_{t-1} | y_0, \ldots, y_{t-1}) dX_{t-1} \\
&= \int P(X_t \mid X_{t-1}, \cancel{y_0, \ldots, y_{t-1}}) P(X_{t-1} \mid y_0, \ldots, y_{t-1}) dX_{t-1} \\
&= \int P(X_t \mid X_{t-1}) P(X_{t-1} \mid y_0, \ldots, y_{t-1}) dX_{t-1}
\end{aligned}
$$

Independence assumption

Slide credit: Svetlana Lazebnik

Induction Step: Correction

- Correction involves computing $P(X_t | y_0, \ldots, y_t)$ given predicted value $P(X_t | y_0, \ldots, y_{t-1})$

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

$$= \frac{P(y_t \mid X_t, y_0, \ldots, y_{t-1}) P(X_t \mid y_0, \ldots, y_{t-1})}{P(y_t \mid y_0, \ldots, y_{t-1})}$$

Bayes' rule

$$\boxed{P(A|B) = \frac{P(B|A)P(A)}{P(B)}}$$

Slide credit: Svetlana Lazebnik

Induction Step: Correction

- Correction involves computing $P(X_t | y_0, \ldots, y_t)$ given predicted value $P(X_t | y_0, \ldots, y_{t-1})$

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

$$= \frac{P(y_t \mid X_t, \cancel{y_0, \ldots, y_{t-1}})P(X_t \mid y_0, \ldots, y_{t-1})}{P(y_t \mid y_0, \ldots, y_{t-1})}$$

$$= \frac{P(y_t \mid X_t)P(X_t \mid y_0, \ldots, y_{t-1})}{P(y_t \mid y_0, \ldots, y_{t-1})}$$

Independence assumption
(observation y_t depends only on state X_t)

Slide credit: Svetlana Lazebnik

Induction Step: Correction

- Correction involves computing $P(X_t | y_0, \ldots, y_t)$ given predicted value $P(X_t | y_0, \ldots, y_{t-1})$

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

$$= \frac{P(y_t \mid X_t, y_0, \ldots, y_{t-1}) P(X_t \mid y_0, \ldots, y_{t-1})}{P(y_t \mid y_0, \ldots, y_{t-1})}$$

$$= \frac{P(y_t \mid X_t) P(X_t \mid y_0, \ldots, y_{t-1})}{P(y_t \mid y_0, \ldots, y_{t-1})}$$

$$= \frac{P(y_t \mid X_t) P(X_t \mid y_0, \ldots, y_{t-1})}{\int P(y_t \mid X_t) P(X_t \mid y_0, \ldots, y_{t-1}) dX_t}$$

Conditioning on X_t

Slide credit: Svetlana Lazebnik

Summary: Prediction and Correction

- Prediction:

$$P(X_t \mid y_0,\ldots,y_{t-1}) = \int \underbrace{P(X_t \mid X_{t-1})}_{\text{Dynamics model}} \underbrace{P(X_{t-1} \mid y_0,\ldots,y_{t-1})}_{\text{Corrected estimate from previous step}} dX_{t-1}$$

Slide credit: Svetlana Lazebnik

Summary: Prediction and Correction

- Prediction:

$$P(X_t \mid y_0,\ldots,y_{t-1}) = \int \underbrace{P(X_t \mid X_{t-1})}_{\text{Dynamics model}} \underbrace{P(X_{t-1} \mid y_0,\ldots,y_{t-1})}_{\text{Corrected estimate from previous step}} dX_{t-1}$$

- Correction:

$$P(X_t \mid y_0,\ldots,y_t) = \frac{\overbrace{P(y_t \mid X_t)}^{\text{Observation model}} \overbrace{P(X_t \mid y_0,\ldots,y_{t-1})}^{\text{Predicted estimate}}}{\int P(y_t \mid X_t) P(X_t \mid y_0,\ldots,y_{t-1}) dX_t}$$

Slide credit: Svetlana Lazebnik

Topics of This Lecture

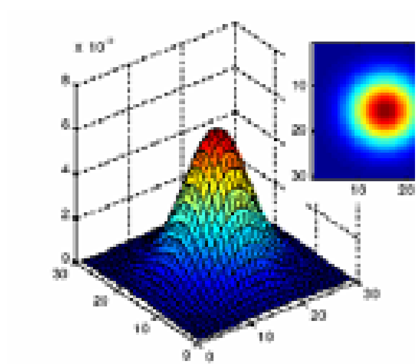
- Tracking with Dynamics
  - Detection vs. Tracking
  - Tracking as probabilistic inference
  - Prediction and Correction
- Linear Dynamic Models
  - Zero velocity model
  - Constant velocity model
  - Constant acceleration model
- The Kalman Filter
  - Kalman filter for 1D state
  - General Kalman filter
  - Limitations

Notation Reminder

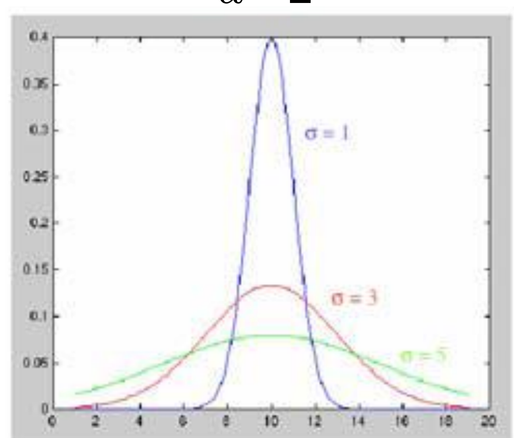
$$\mathbf{x} \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$$

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

d =2



d =1



If $\mathbf{x}$ is 1D, we just have one Σ parameter: the variance σ^2

Slide credit: Kristen Grauman

Linear Dynamic Models

- Dynamics model
  - State undergoes linear tranformation D_t plus Gaussian noise

$$\underset{n\times 1}{\boldsymbol{x}_t} \sim N\left(\underset{n\times n}{\boldsymbol{D}_t}\,\underset{n\times 1}{\boldsymbol{x}_{t-1}}, \Sigma_{d_t}\right)$$

- Observation model
  - Measurement is linearly transformed state plus Gaussian noise

$$\underset{m\times 1}{\boldsymbol{y}_t} \sim N\left(\underset{m\times n}{\boldsymbol{M}_t}\,\underset{n\times 1}{\boldsymbol{x}_t}, \Sigma_{m_t}\right)$$

Slide credit: Svetlana Lazebnik, Kristen Grauman

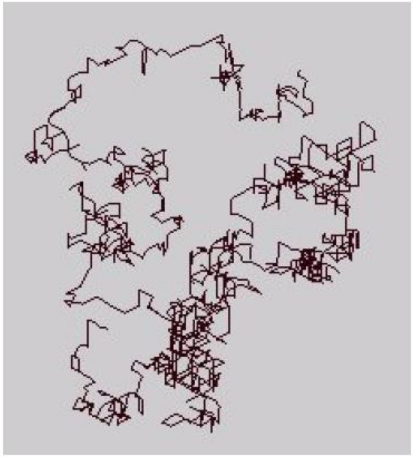
Example: Randomly Drifting Points

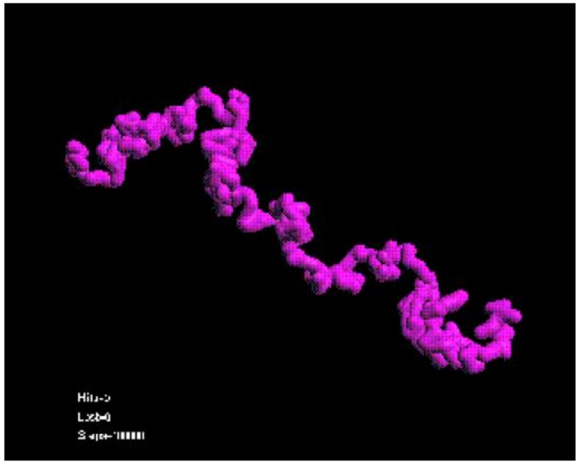
- Consider a stationary object, with state as position.
  - Position is constant, only motion due to random noise term.

$$x_t = p_t \qquad p_t = p_{t-1} + \varepsilon$$

$\Rightarrow$ State evolution is described by identity matrix $D = I$

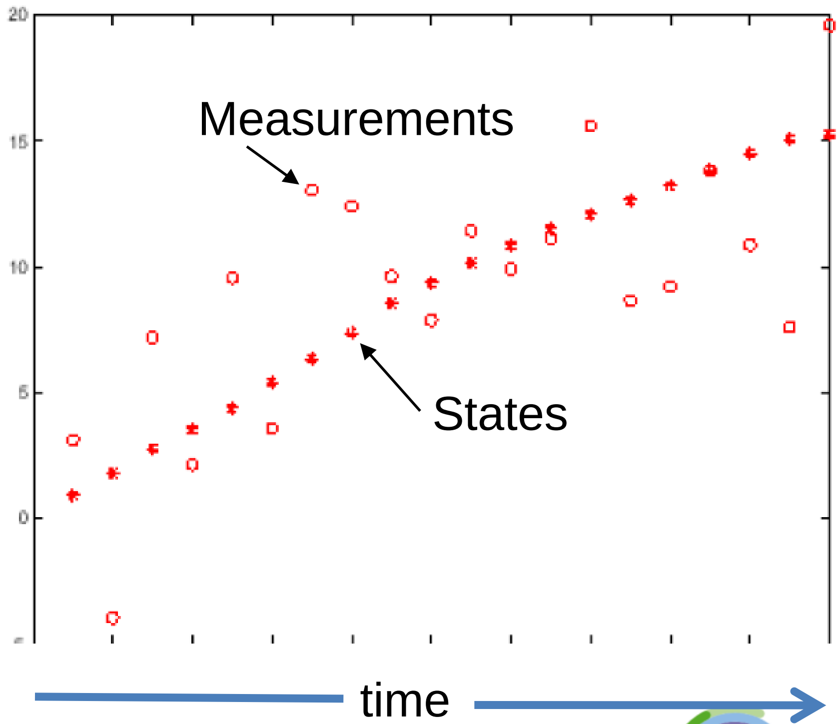
$$x_t = D_t x_{t-1} + noise = I p_{t-1} + noise$$





Slide credit: Kristen Grauman

Example: Constant Velocity (1D Points)



Slide credit: Kristen Grauman

Figure from Forsyth & Ponce

Example: Constant Velocity (1D Points)

- State vector: position p and velocity v

$$x_t = \begin{bmatrix} p_t \\ v_t \end{bmatrix} \qquad p_t =$$

(greek letters denote noise terms)

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

- Measurement is position only

$$y_t = M x_t + noise =$$

Slide credit: Svetlana Lazebnik, Kristen Grauman

Example: Constant Velocity (1D Points)

- State vector: position p and velocity v

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

(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

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

Slide credit: Svetlana Lazebnik, Kristen Grauman

Example: Constant Acceleration (1D Points)

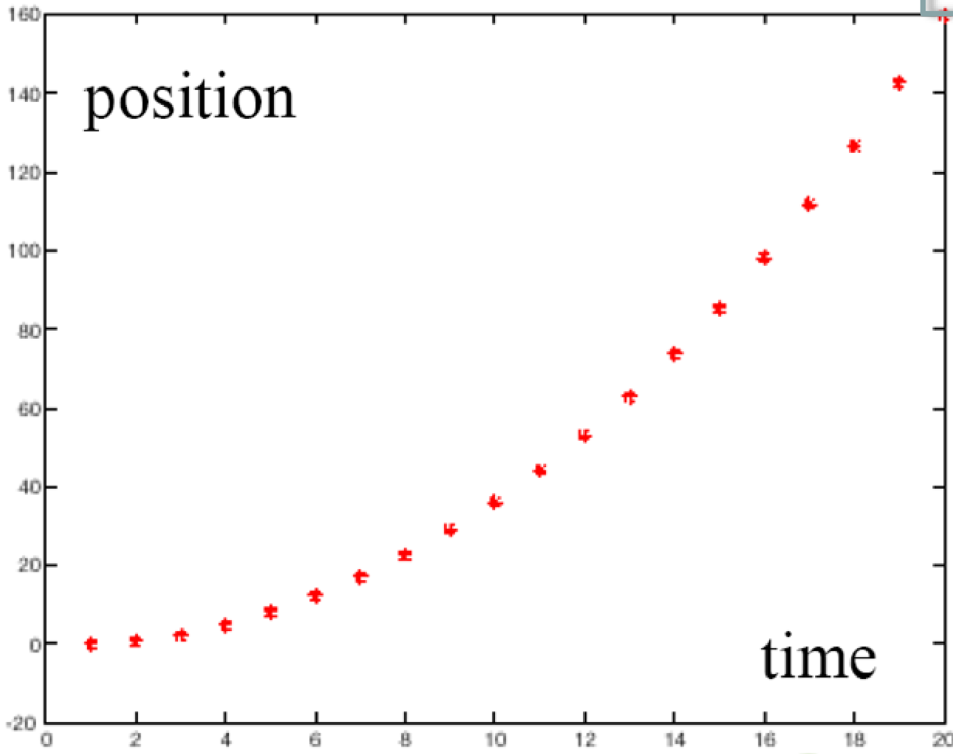


Figure from Forsyth & Ponce

Slide credit: Kristen Grauman

Example: Constant Acceleration (1D Points)

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

$$x_t = \begin{bmatrix} p_t \\ v_t \\ a_t \end{bmatrix} \qquad \begin{aligned} p_t &= p_{t-1} + (\Delta t) v_{t-1} + \tfrac{1}{2}(\Delta t)^2 a_{t-1} + \varepsilon \\ v_t &= \\ a_t &= \end{aligned}$$

(greek letters denote noise terms)

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

- Measurement is position only

$$y_t = Mx_t + noise =$$

Slide credit: Svetlana Lazebnik, Kristen Grauman

Example: Constant Acceleration (1D Points)

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

$$x_t = \begin{bmatrix} p_t \\ v_t \\ a_t \end{bmatrix} \qquad \begin{aligned} p_t &= p_{t-1} + (\Delta t) v_{t-1} + \tfrac{1}{2} (\Delta t)^2 a_{t-1} + \varepsilon \\ v_t &= v_{t-1} + (\Delta t) a_{t-1} + \xi \\ a_t &= a_{t-1} + \zeta \end{aligned}$$

(greek letters denote noise terms)

$$x_t = D_t x_{t-1} + noise = \begin{bmatrix} 1 & \Delta t & \frac{1}{2}(\Delta t)^2 \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_{t-1} \\ v_{t-1} \\ a_{t-1} \end{bmatrix} + noise$$

- Measurement is position only

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

Slide credit: Svetlana Lazebnik, Kristen Grauman

Recap: General Motion Models

- Assuming we have differential equations for the motion
  - E.g. for (undampened) periodic motion of a linear spring

$$\frac{d^2 p}{dt^2} = -p$$

- Substitute variables to transform this into linear system

$$p_1 = p \qquad p_2 = \frac{dp}{dt} \qquad p_3 = \frac{d^2 p}{dt^2}$$

- Then we have

$$x_t = \begin{bmatrix} p_{1,t} \\ p_{2,t} \\ p_{3,t} \end{bmatrix} \qquad \begin{aligned} p_{1,t} &= p_{1,t-1} + (\Delta t) p_{2,t-1} + \tfrac{1}{2}(\Delta t)^2 p_{3,t-1} + \varepsilon \\ p_{2,t} &= p_{2,t-1} + (\Delta t) p_{3,t-1} + \xi \\ p_{3,t} &= -p_{1,t-1} + \zeta \end{aligned} \qquad D_t = \begin{bmatrix} 1 & \Delta t & \tfrac{1}{2}(\Delta t)^2 \\ 0 & 1 & \Delta t \\ -1 & 0 & 0 \end{bmatrix}$$

Topics of This Lecture

- Tracking with Dynamics
  - Detection vs. Tracking
  - Tracking as probabilistic inference
  - Prediction and Correction
- Linear Dynamic Models
  - Zero velocity model
  - Constant velocity model
  - Constant acceleration model
- The Kalman Filter
  - Kalman filter for 1D state
  - General Kalman filter
  - Limitations

The Kalman Filter

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

Slide credit: Svetlana Lazebnik

The 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")

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

$$P\left(X_t \mid y_0, \ldots, y_t\right)$$

Mean and std. dev. of predicted state:

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

Time advances: t++

Mean and std. dev. of corrected state:

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

Slide credit: Kristen Grauman

Kalman Filter for 1D State

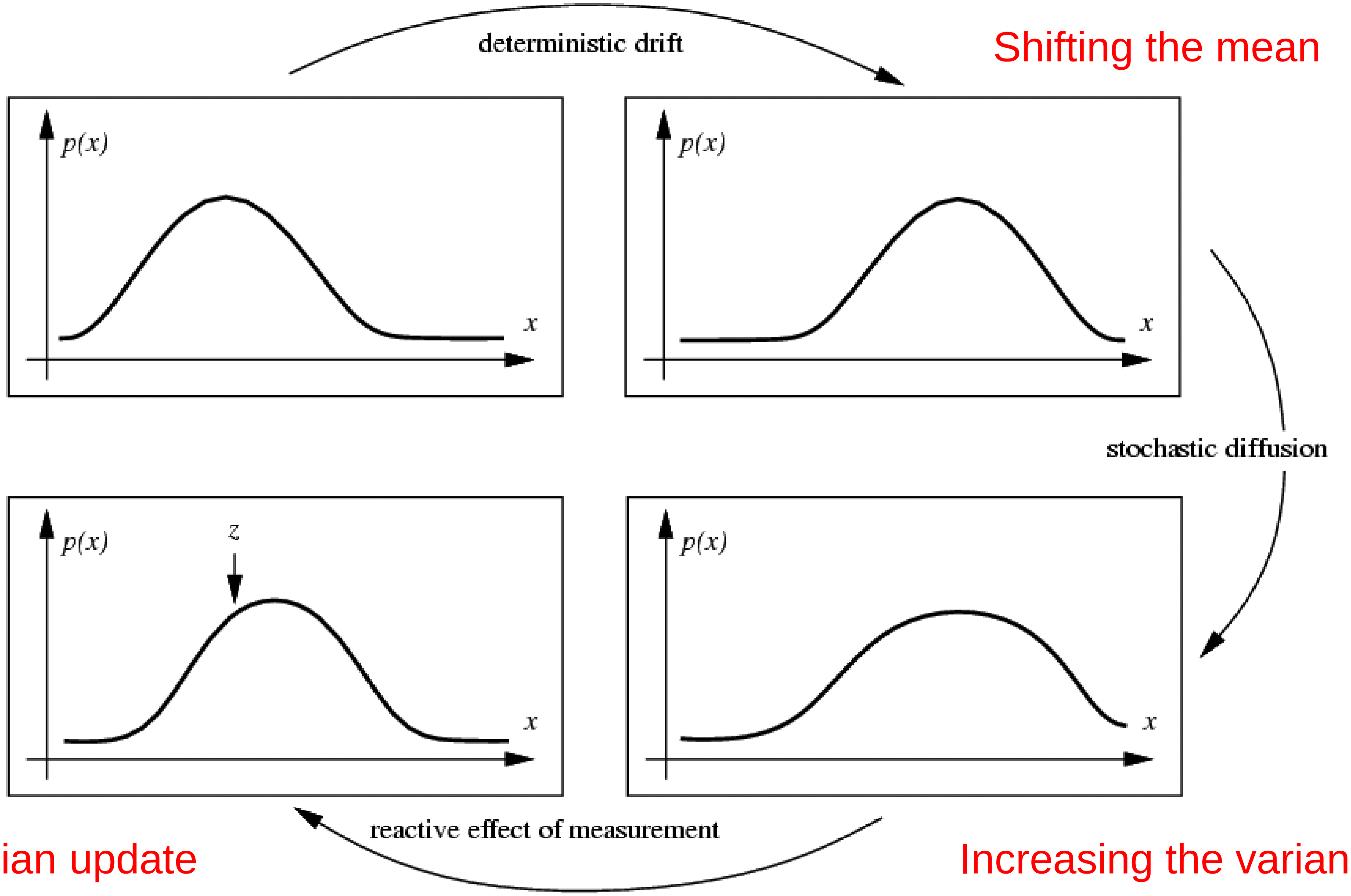
- Want to represent and update

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

$$P\left(x_t \middle| y_0, \ldots, y_t\right) = N\left(\mu_t^+, (\sigma_t^+)^2\right)$$

Propagation of Gaussian densities

Shifting the mean



Bayesian update

Increasing the variance

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:

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

for derivations,
see F&P Chapter 17.3

- Update the variance:

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

Slide credit: Kristen Grauman

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 measurement: $$P\left(X_t \middle| y_0, \ldots, y_t\right) = N\left(\mu_t^+, (\sigma_t^+)^2\right)$$
- 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:
$$(\sigma_t^+)^2 = \frac{\sigma_m^2 (\sigma_t^-)^2}{\sigma_m^2 + m^2 (\sigma_t^-)^2}$$

Slide credit: Kristen Grauman

Derivations: F&P Chapter 17.3

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!

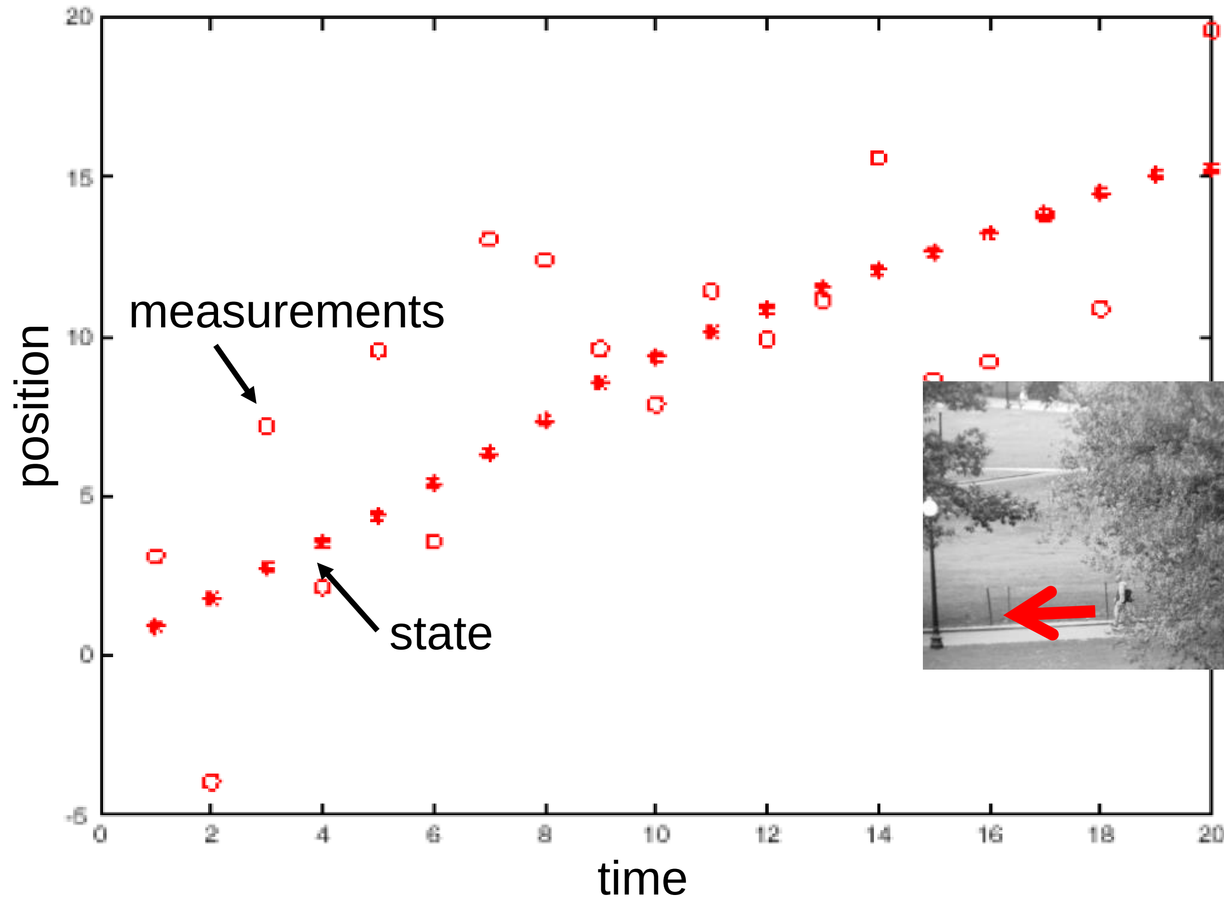
- What if there is no measurement uncertainty $(\sigma_m = 0)$?

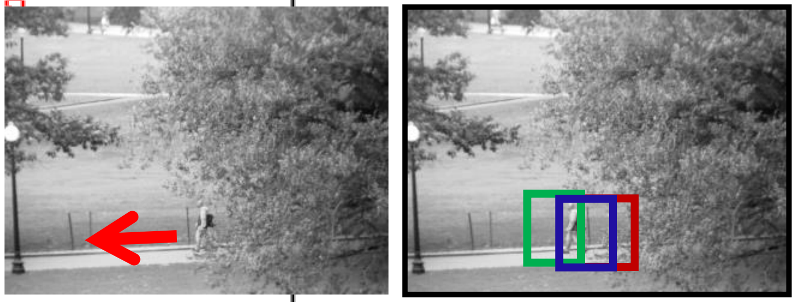
$$\mu_t^+ = \frac{y_t}{m} \qquad (\sigma_t^+)^2 = 0$$

The prediction is ignored!

Slide credit: Kristen Grauman

Recall: Constant Velocity Example



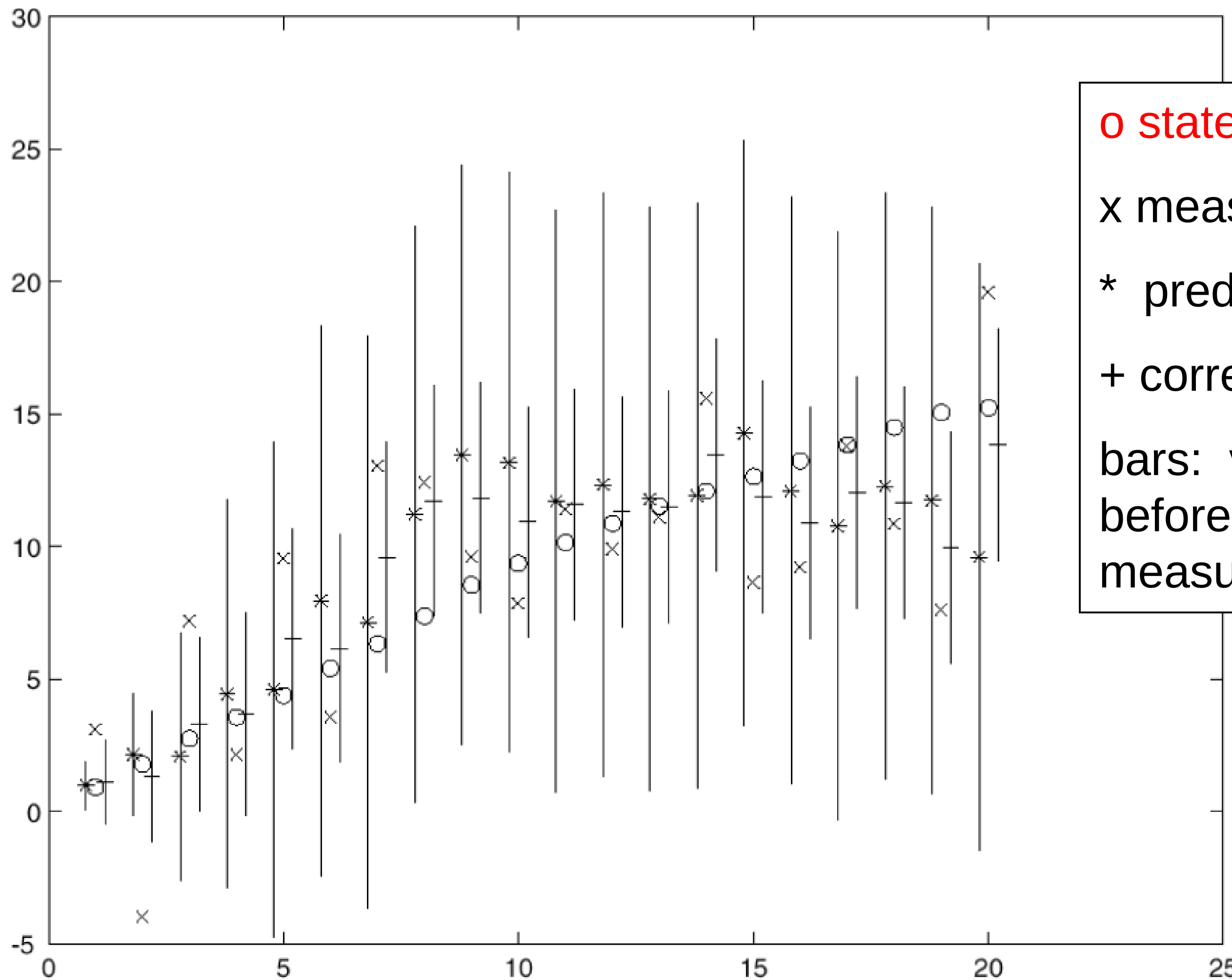


State is 2D: position + velocity
Measurement is 1D: position

Slide credit: Kristen Grauman

Figure from Forsyth & Ponce

Constant Velocity Model



o state

x measurement

* predicted mean estimate

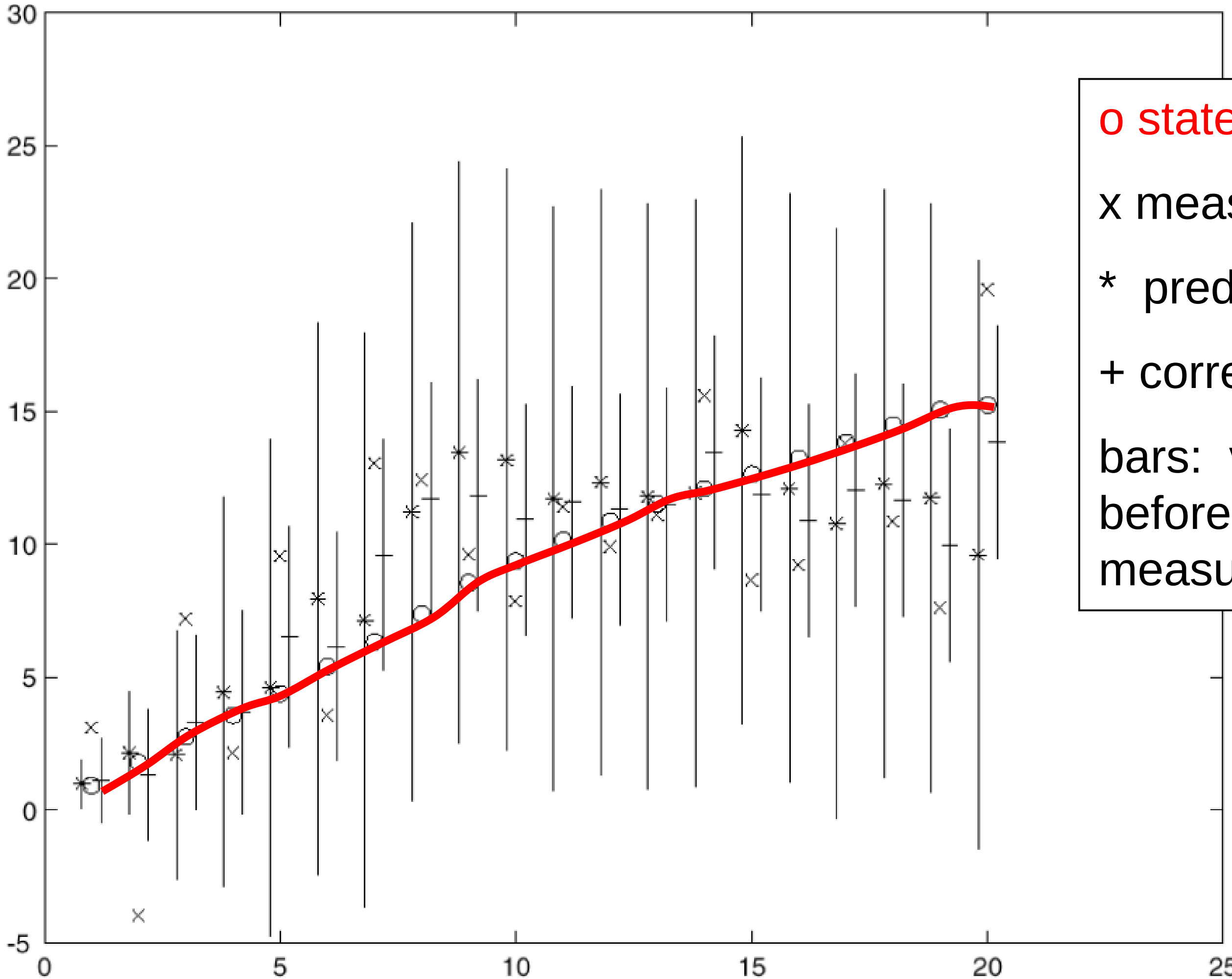
+ corrected mean estimate

bars: variance estimates before and after measurements

Slide credit: Kristen Grauman

Figure from Forsyth & Ponce

Constant Velocity Model



o state

x measurement

* predicted mean estimate

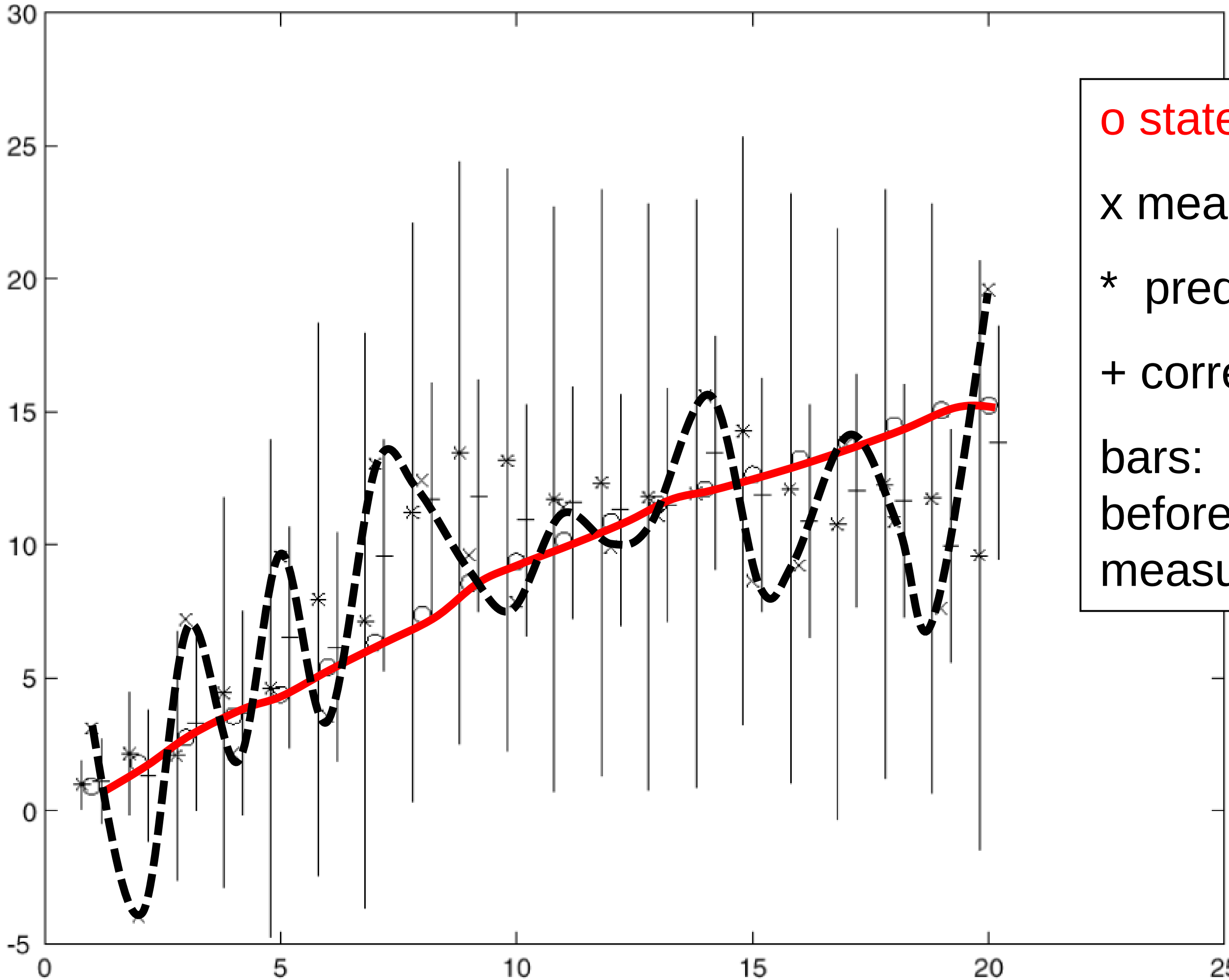
+ corrected mean estimate

bars: variance estimates before and after measurements

Slide credit: Kristen Grauman

Figure from Forsyth & Ponce

Constant Velocity Model



o state

x measurement

* predicted mean estimate

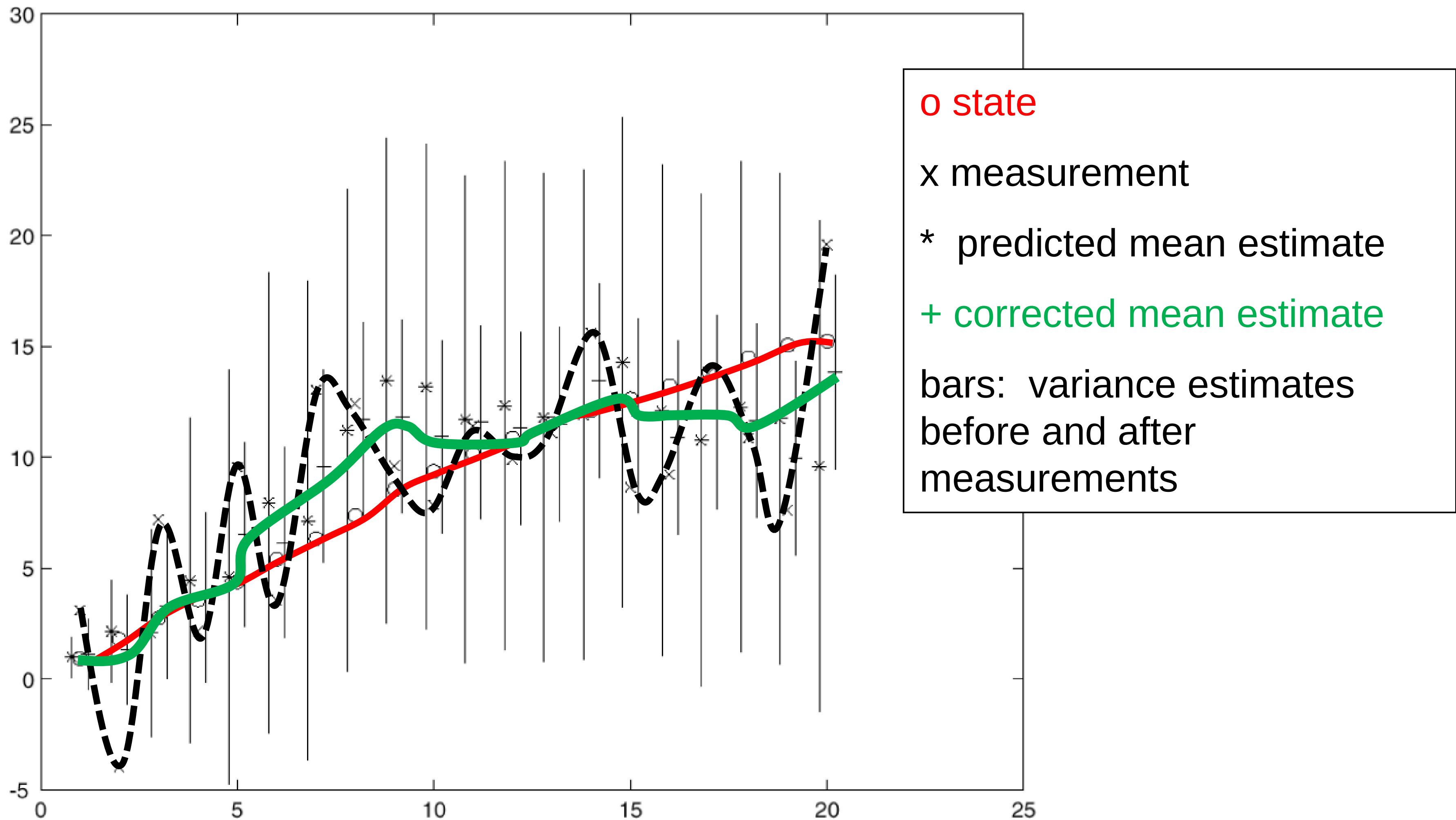
+ corrected mean estimate

bars: variance estimates before and after measurements

Slide credit: Kristen Grauman

Figure from Forsyth & Ponce

Constant Velocity Model



Slide credit: Kristen Grauman

Figure from Forsyth & Ponce

Kalman Filter: General Case (>1dim)

PREDICT

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

$$\Sigma_t^- = D_t \Sigma_{t-1}^+ D_t^T + \Sigma_{d_t}$$

CORRECT

$$K_t = \Sigma_t^- M_t^T \left(M_t \Sigma_t^- M_t^T + \Sigma_{m_t} \right)^{-1}$$

$$x_t^+ = x_t^- + \underbrace{K_t}_{\text{"Kalman gain"}} \underbrace{\left(y_t - M_t x_t^- \right)}_{\text{"residual"}}$$

$$\Sigma_t^+ = \left(I - K_t M_t \right) \Sigma_t^-$$

"residual"

"Kalman gain"

More weight on residual when measurement error covariance approaches 0.

Less weight on residual as a priori estimate error covariance approaches 0.

for derivations, see F&P Chapter 17.3

Slide credit: Kristen Grauman

Summary: Kalman Filter

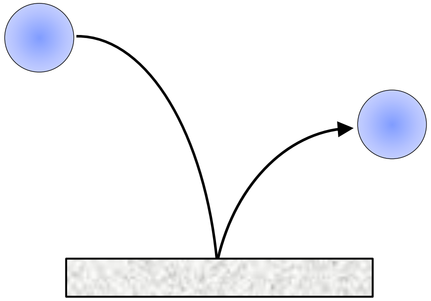
- Pros:
  - Gaussian densities everywhere
  - Simple updates, compact and efficient
  - Very established method, very well understood
- Cons:
  - Unimodal distribution, only single hypothesis
  - Restricted class of motions defined by linear model

Slide adapted from Svetlana Lazebnik

Why Is This A Restriction?

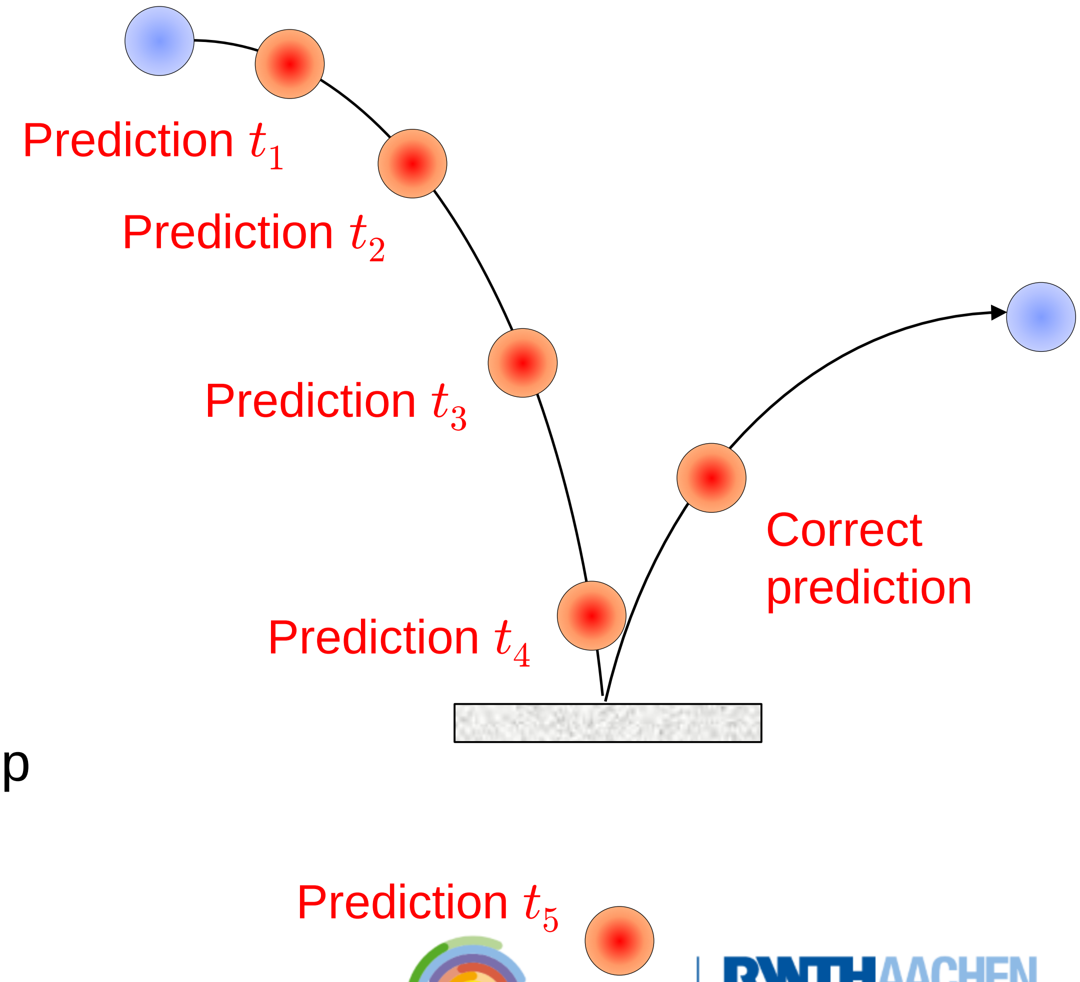
- Many interesting cases don't have linear dynamics
  - E.g. pedestrians walking

  - E.g. a ball bouncing



Ball Example: What Goes Wrong Here?

- Assuming constant acceleration model
- Prediction is too far from true position to compensate…
- Possible solution:
  - Keep multiple different motion models in parallel
  - I.e. would check for bouncing at each time step



References and Further Reading

- A very good introduction to tracking with linear dynamic models and Kalman filters can be found in Chapter 17 of
  - D. Forsyth, J. Ponce, *Computer Vision – A Modern Approach*. Prentice Hall, 2003