CS 2770: Computer Vision

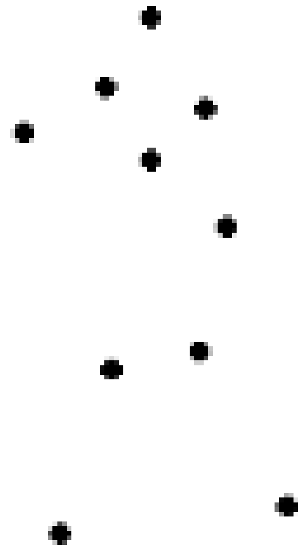
Motion, Tracking & Actions

Prof. Adriana Kovashka
University of Pittsburgh
March 19, 2019

Plan for this lecture

- Tracking how an object moves
- Modeling and replicating motion
- Recognizing human actions
- ConvNets for video

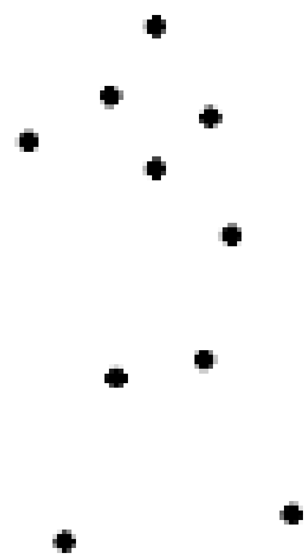
Motion: Why is it useful?



Derek Hoiem

Motion: Why is it useful?

- Even “impoverished” motion data can evoke a strong percept



G. Johansson, “Visual Perception of Biological Motion and a Model For Its Analysis", *Perception and Psychophysics 14, 201-211, 1973.*

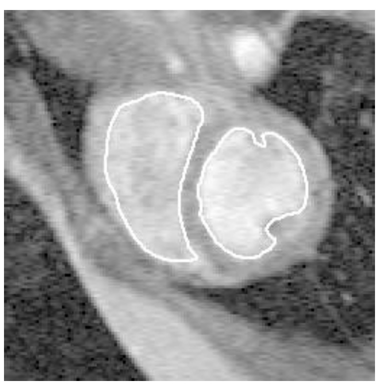
Derek Hoiem

Tracking: some applications

Body pose tracking, activity recognition

Censusing a bat population

Video-based interfaces



Medical apps

Surveillance

Tracking examples

Traffic: https://www.youtube.com/watch?v=DiZHQ4peqjg

Soccer: http://www.youtube.com/watch?v=ZqQIItFAnxg

Face: http://www.youtube.com/watch?v=i_bZNVmhJ2o

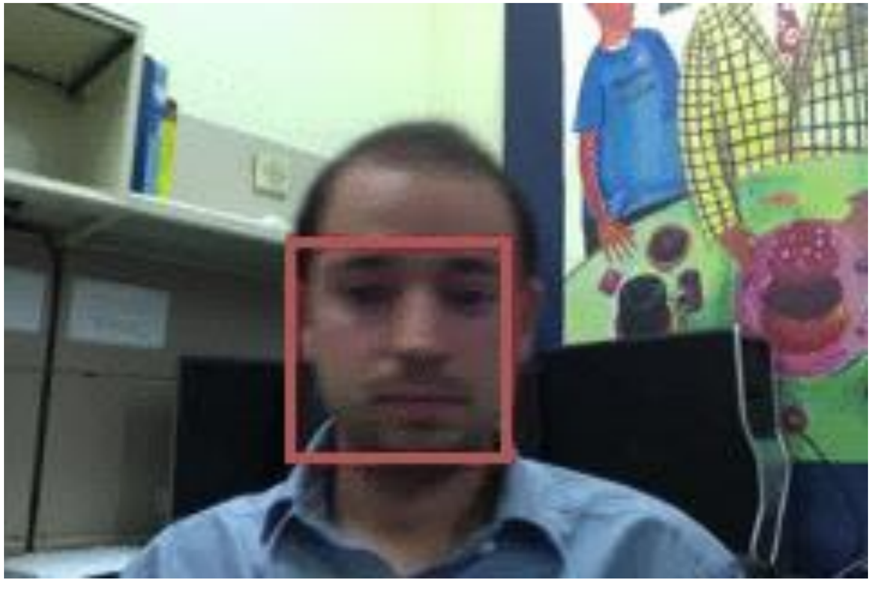
Body: https://www.youtube.com/watch?v=_Ahy0Gh69-M

Eye: http://www.youtube.com/watch?v=NCtYdUEMotg

Gaze: http://www.youtube.com/watch?v=-G6Rw5cU-1c

Things that make visual tracking difficult

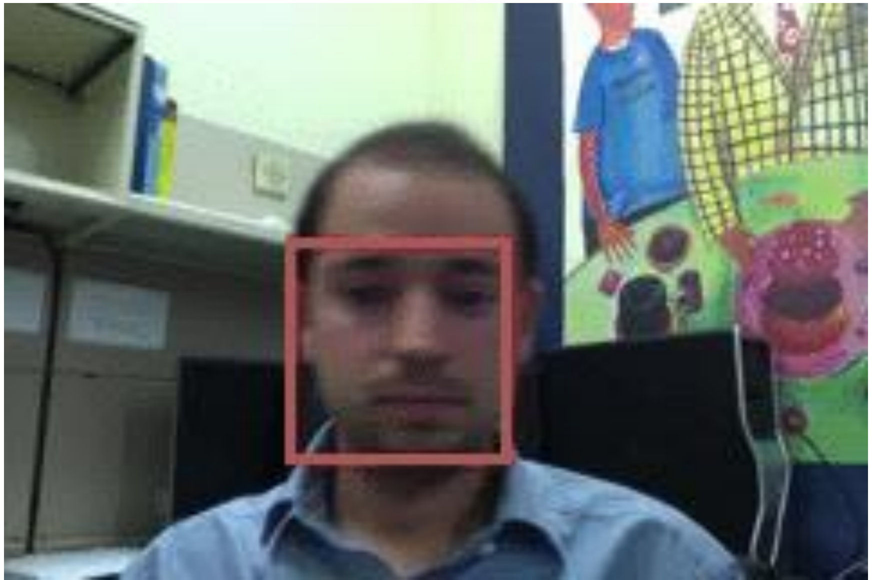
- Erratic movements, moving very quickly
- Occlusions, leaving and coming back
- Surrounding similar-looking objects



Adapted from Amin Sadeghi

Strategies for tracking

- Tracking by repeated detection
  - Works well if object is easily detectable (e.g., face or colored glove) and there is only one
  - Need some way to link up detections
  - Best you can do, if you can't predict motion



Amin Sadeghi

Strategies for tracking

- Tracking w/ dynamics: Using model of expected motion, *predict* object location in next frame
  - Restrict search for the object
  - Measurement noise is reduced by trajectory smoothness
  - Robustness to missing or weak observations
  - Assumptions: Camera is not moving instantly to new viewpoint, objects do not disappear/reappear in different places in the scene

Amin Sadeghi

Detection vs. tracking

t=1

t=2

...

t=20

t=21

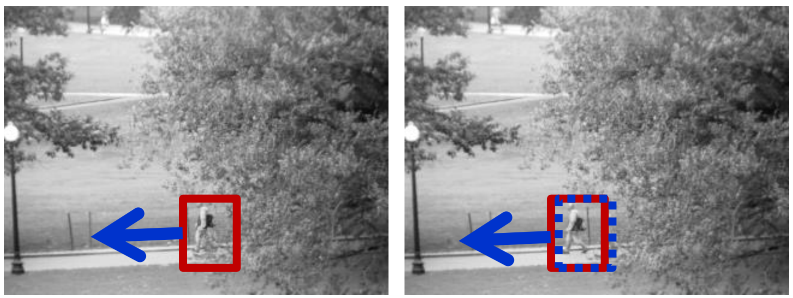
Kristen Grauman

Detection vs. tracking

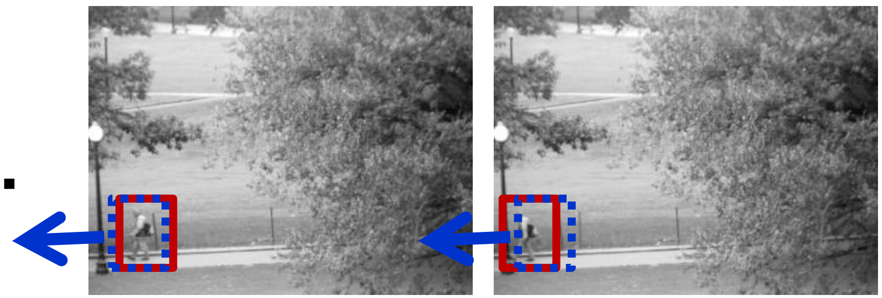
Detection: We detect the object independently in each frame and can record its position over time, e.g., based on detection window coordinates

Adapted from Kristen Grauman

Detection vs. tracking



…



Tracking with *dynamics*: We use image measurements to estimate position of object, but also incorporate position predicted by dynamics, i.e., our expectation of the object's motion pattern

Kristen Grauman

Tracking: prediction + correction

Time t Time t+1

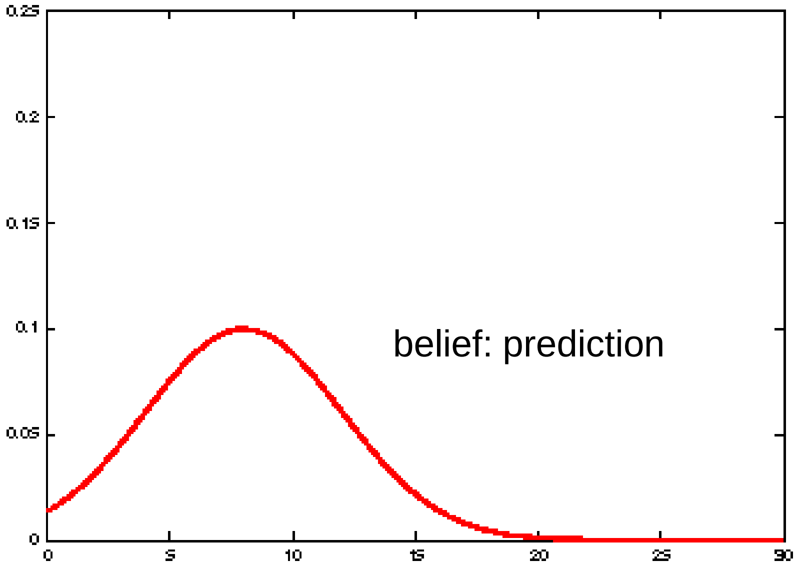
Belief

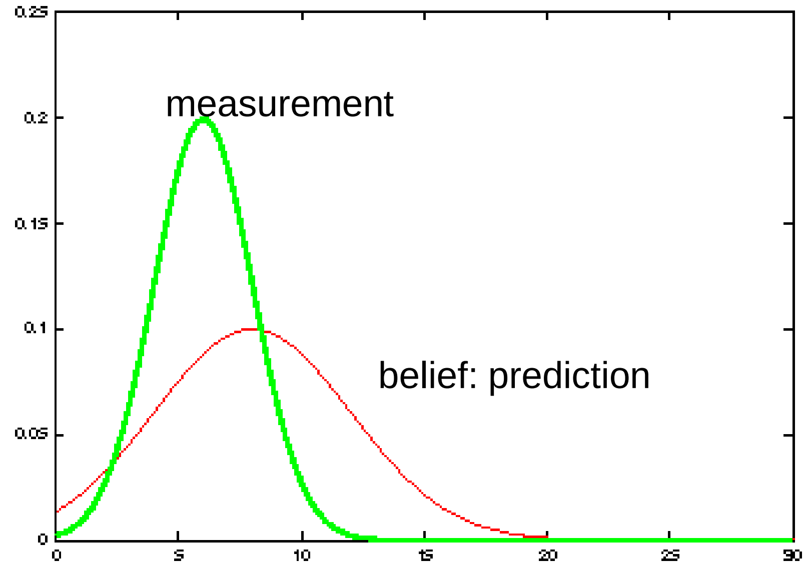
Measurement

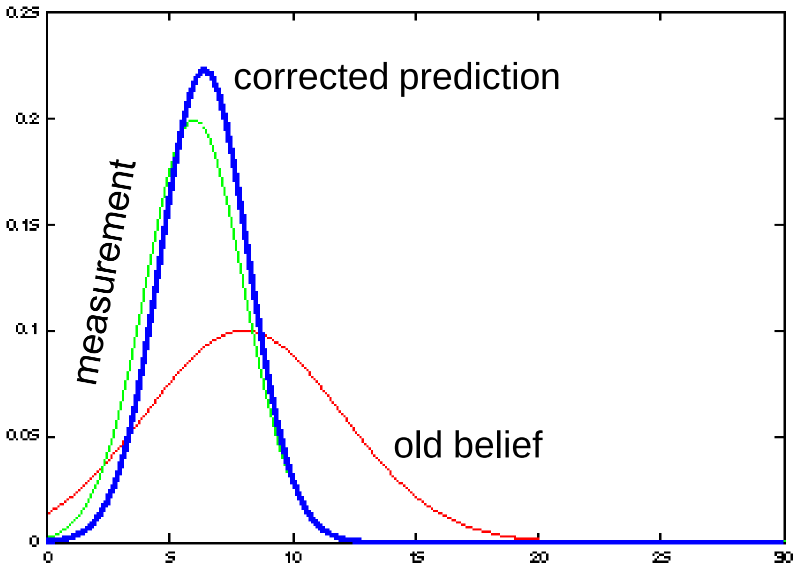
Corrected prediction

Kristen Grauman

Tracking: prediction + correction





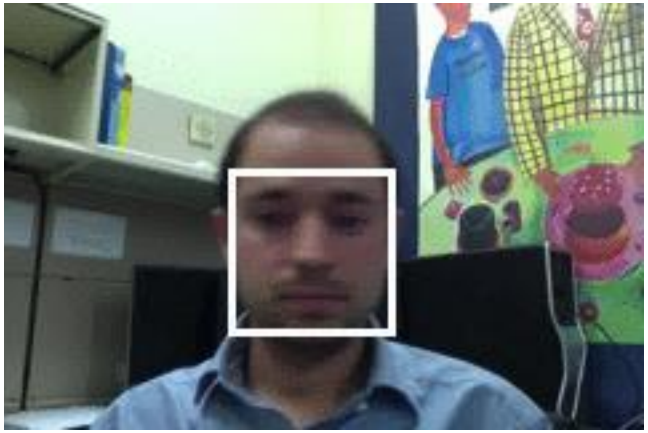


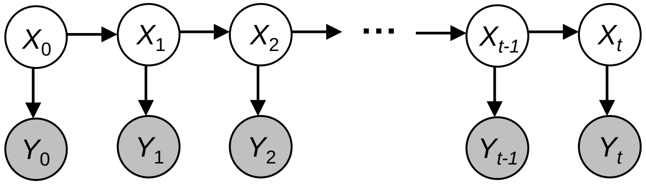
Time t Time t+1

Kristen Grauman

General model for tracking

- *State X*: The actual state of the moving object that we want to estimate but cannot observe
  - E.g. position, velocity
- *Observations Y*: Our actual measurement or observation of state X, which can be very noisy
- At each time t, the state changes to X_t and we get a new observation Y_t
- Our goal is to recover the most likely state X_t given:
  - All observations so far, i.e. $y_1, y_2, \ldots, y_t$
  - Knowledge about dynamics of state transitions





Adapted from Amin Sadeghi and Kristen Grauman

Steps of tracking

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

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

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

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

Kristen Grauman

Problem statement

- We have models for

  Likelihood of next state given current state (dynamics model):
  $$P\left(X_t \middle| X_{t-1}\right)$$

  Likelihood of observation given the state (observation or measurement model):
  $$P\left(Y_t \middle| X_t\right)$$

- We want to recover, for each t: $P\left(X_t \middle| y_0, \ldots, y_t\right)$

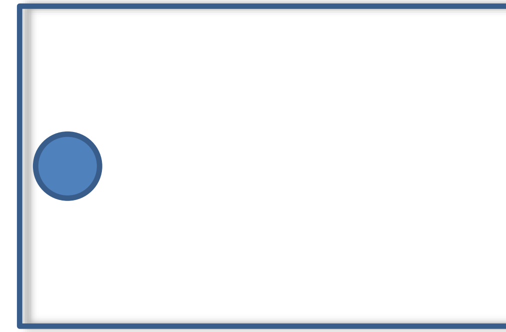
The Kalman filter

- Linear dynamics model: state undergoes linear transformation plus Gaussian noise

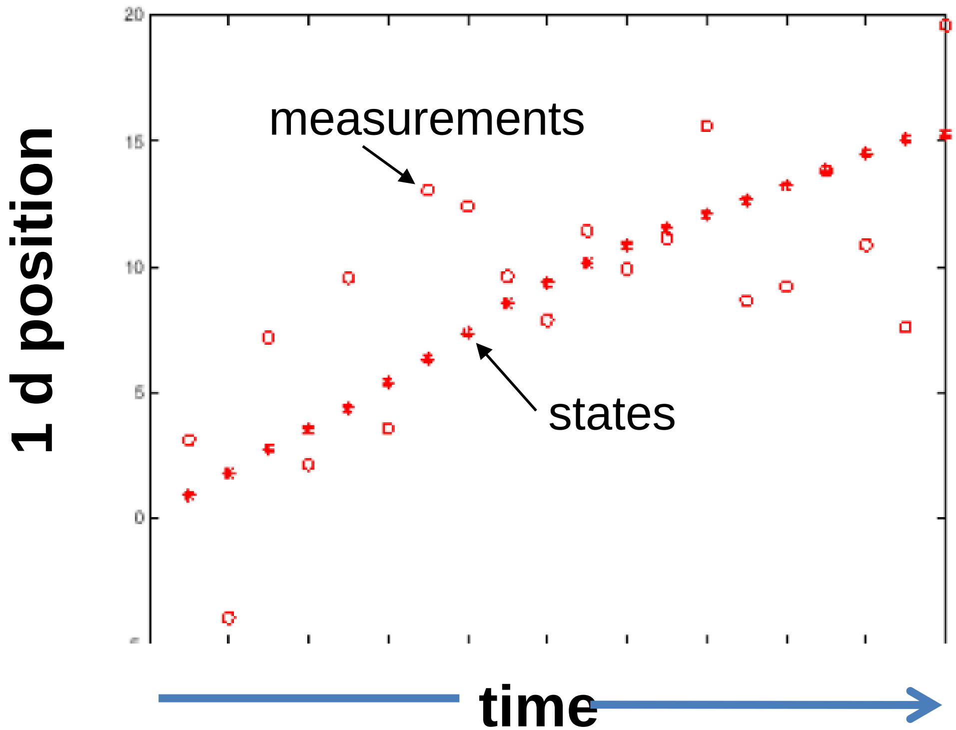
- Observation model: measurement is linearly transformed state plus Gaussian noise

- The predicted/corrected state distributions are Gaussian
  - You only need to maintain the mean and covariance
  - The calculations are easy

Example: Constant velocity (1D points)



1 d position



Kristen Grauman

Example: Constant velocity (1D points)

- State vector: position *p* and velocity *v*

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

$$x_t = \boxed{D_t x_{t-1} + noise} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} p_{t-1} \\ v_{t-1} \end{bmatrix} + noise$$

- Measurement is position only

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

Kristen Grauman

Prediction and correction

See hidden slides for derivation

Prediction:

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

t++

Correction:

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

Adapted from Amin Sadeghi

Prediction and correction

- Prediction:

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

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

Time advances: t++

Receive measurement

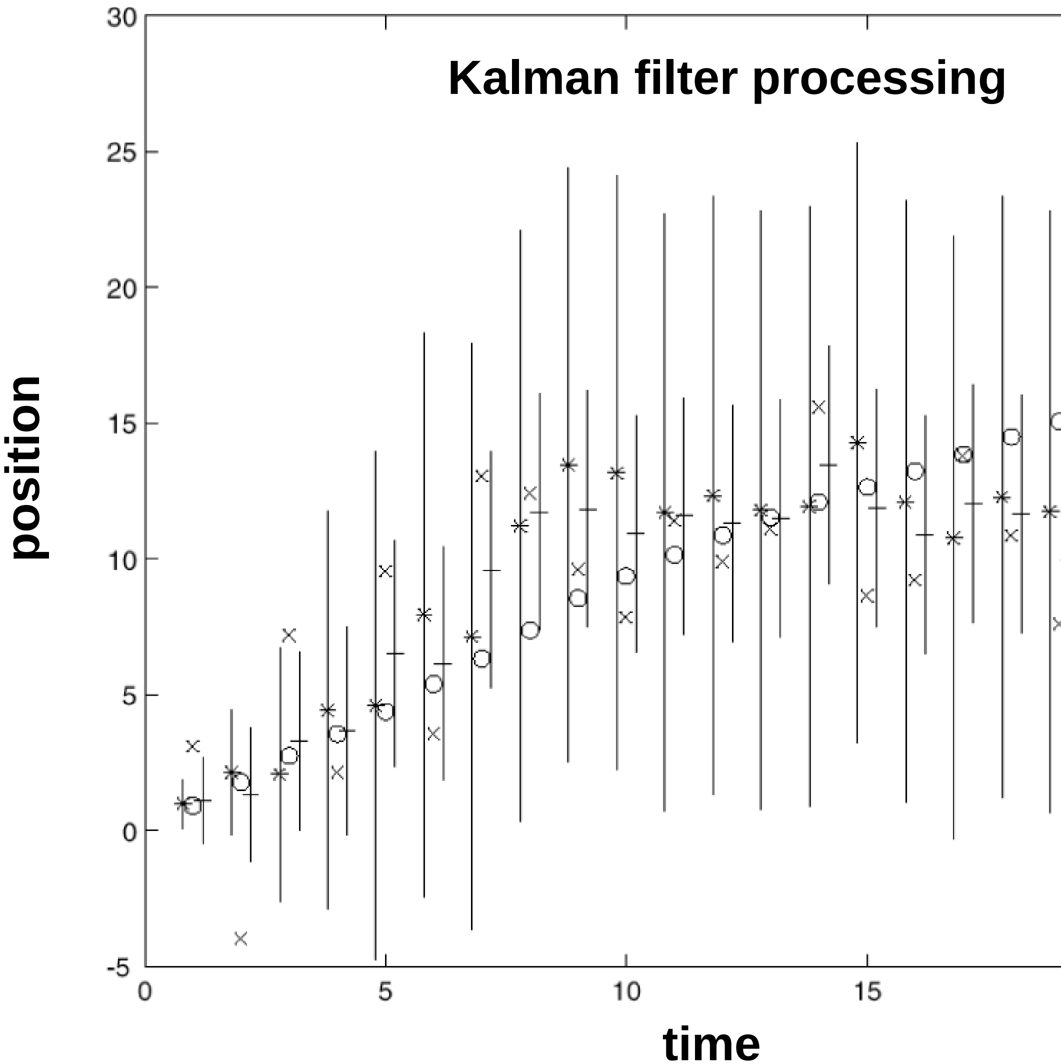
- Correction:

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

$$P\left(X_t \middle| y_0, \ldots, \boxed{y_t}\right)$$

Kristen Grauman

Example w/ constant velocity



o state

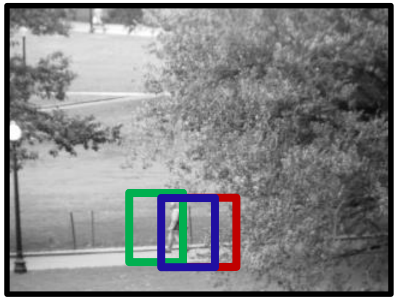
x measurement

* predicted mean estimate

+ corrected mean estimate

bars: variance estimates

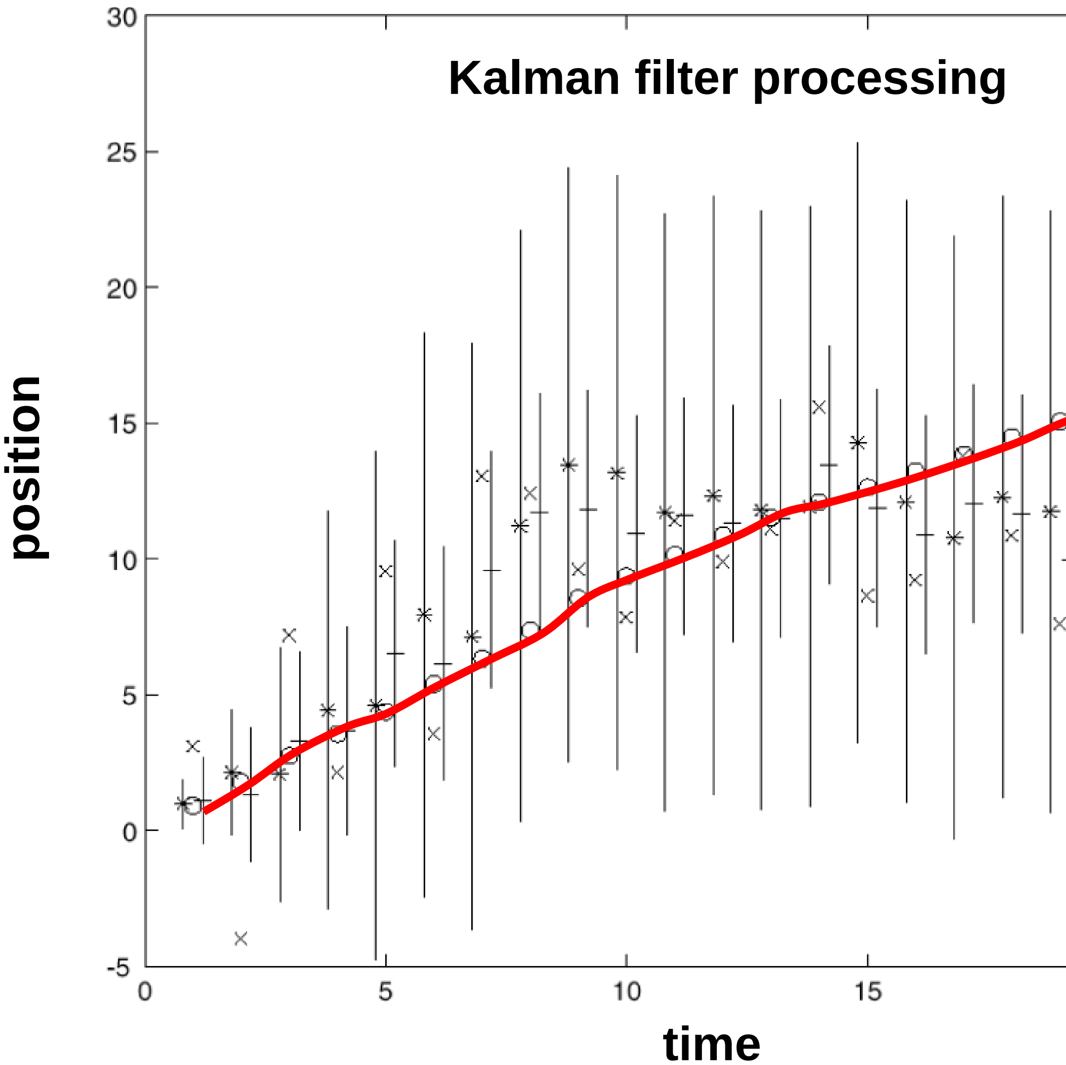
Time t



Time t+1

Adapted from Kristen Grauman

Example w/ constant velocity



o state

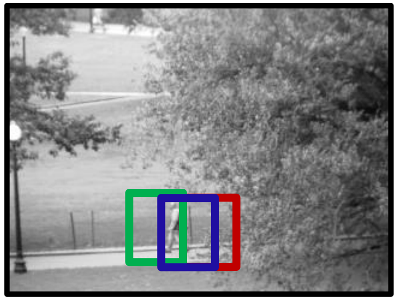
x measurement

* predicted mean estimate

+ corrected mean estimate

bars: variance estimates

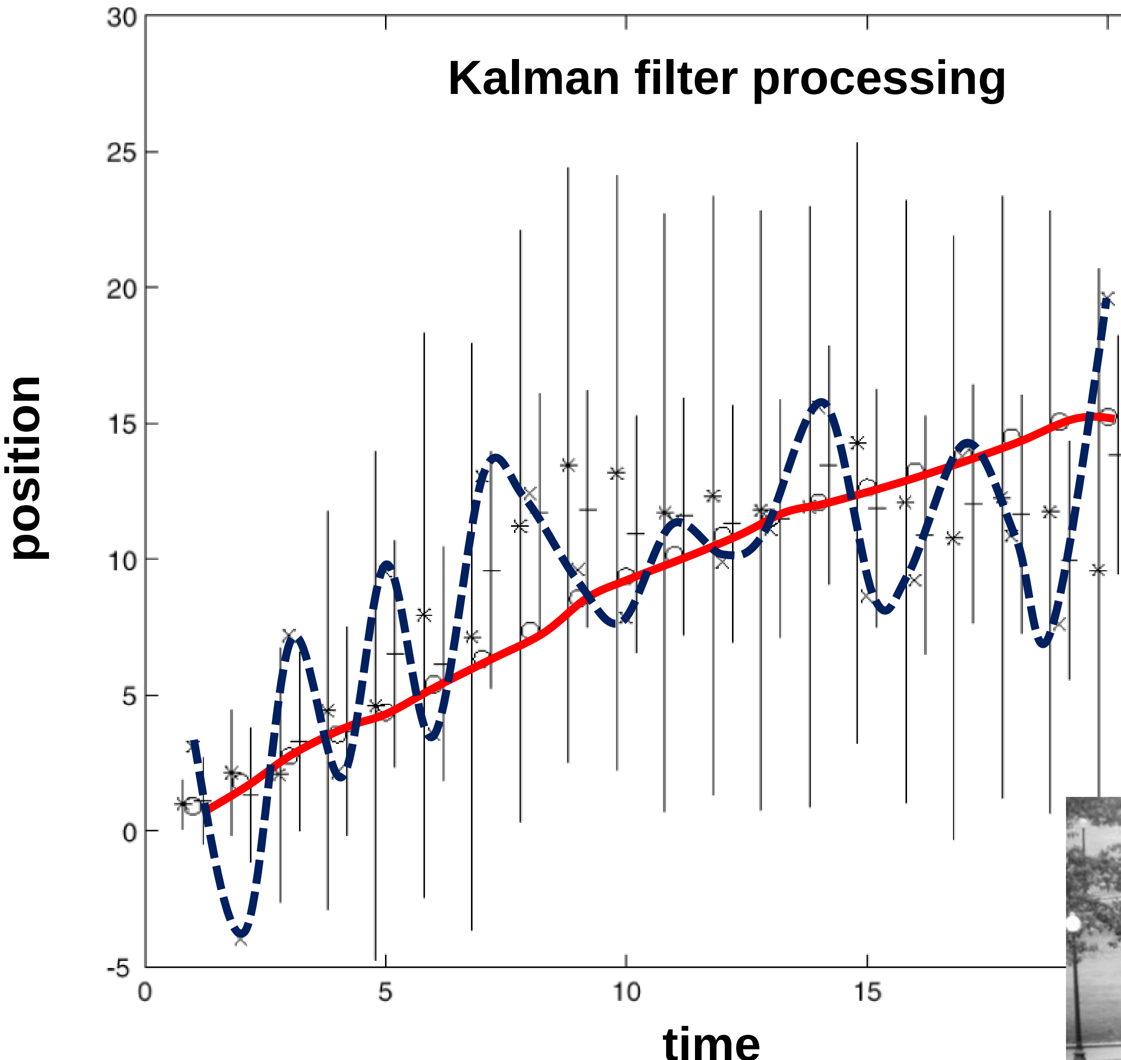
Time t



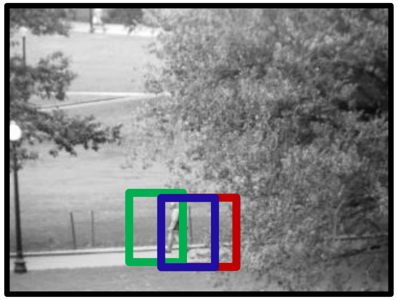
Time t+1

Adapted from Kristen Grauman

Example w/ constant velocity



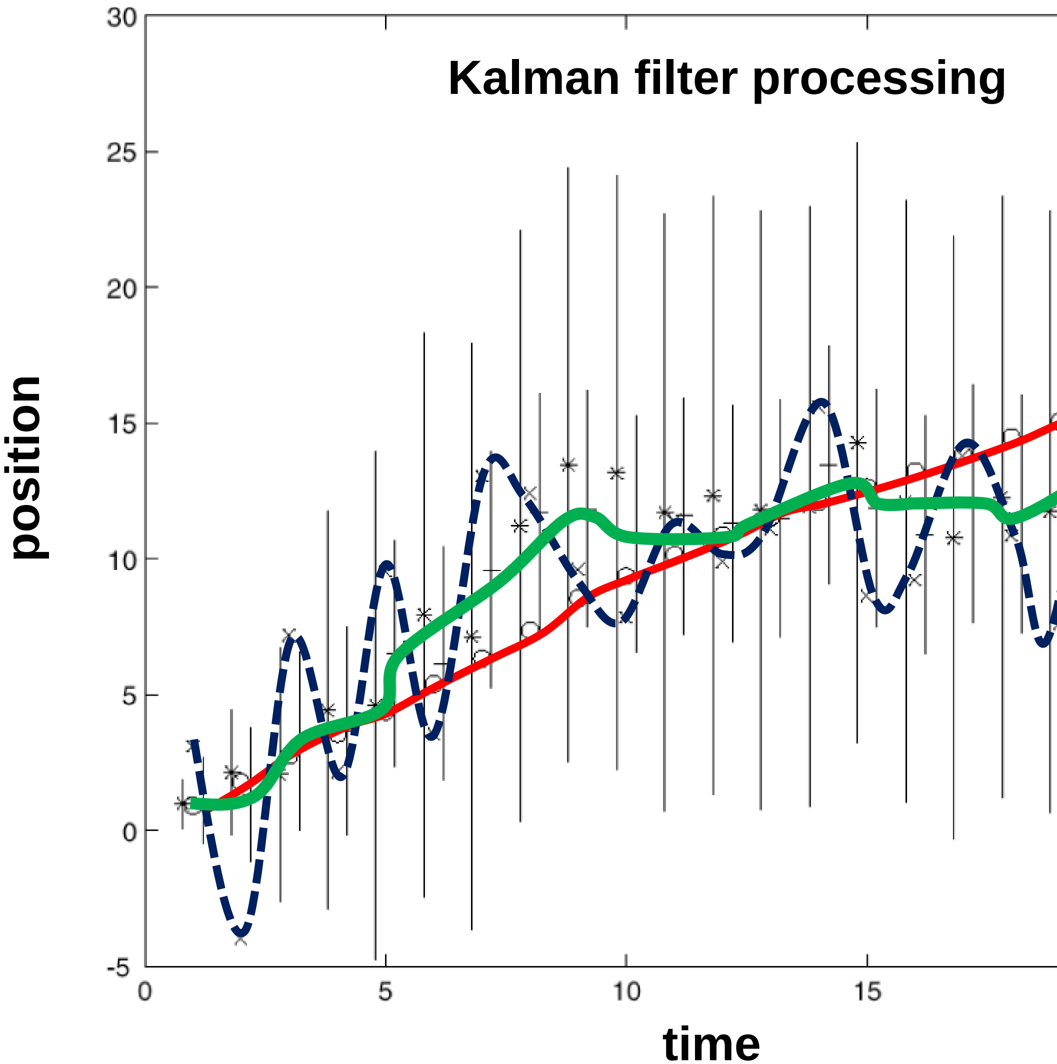
Time t



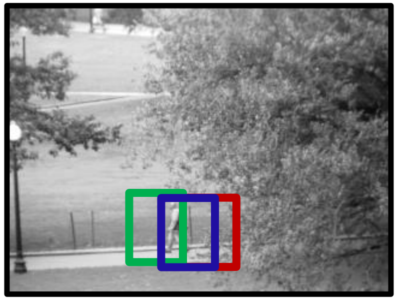
Time t+1

Adapted from Kristen Grauman

Example w/ constant velocity



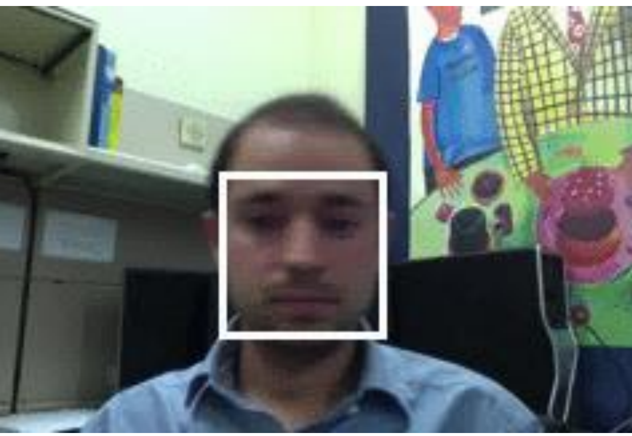
Time t



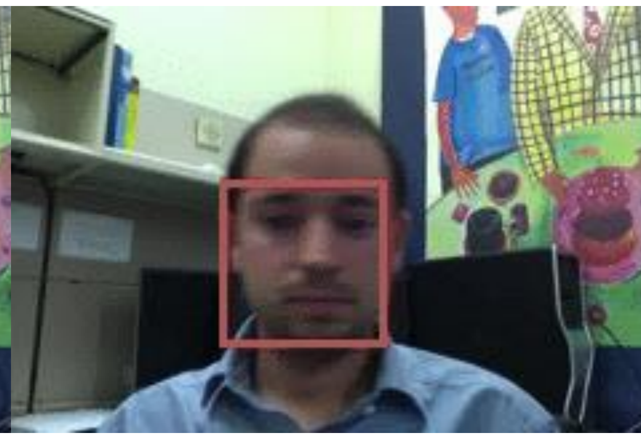
Time t+1

Adapted from Kristen Grauman

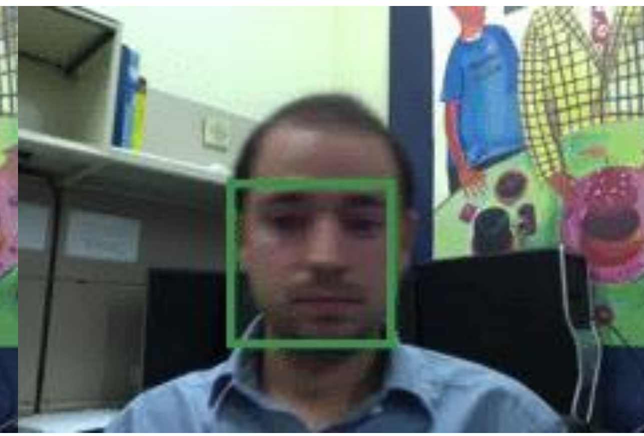
Example w/ constant velocity



Ground Truth



Observation

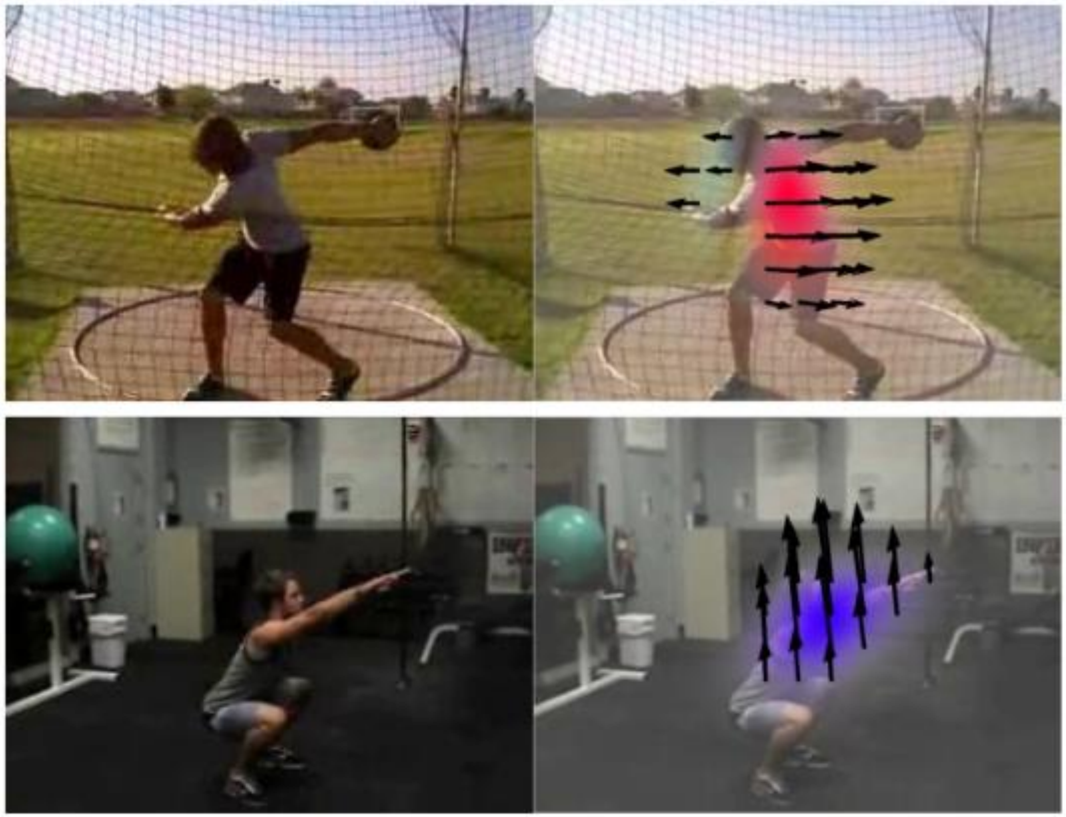


Correction

Plan for this lecture

- Tracking how an object moves
- Modeling and replicating motion
- Recognizing human actions
- ConvNets for video

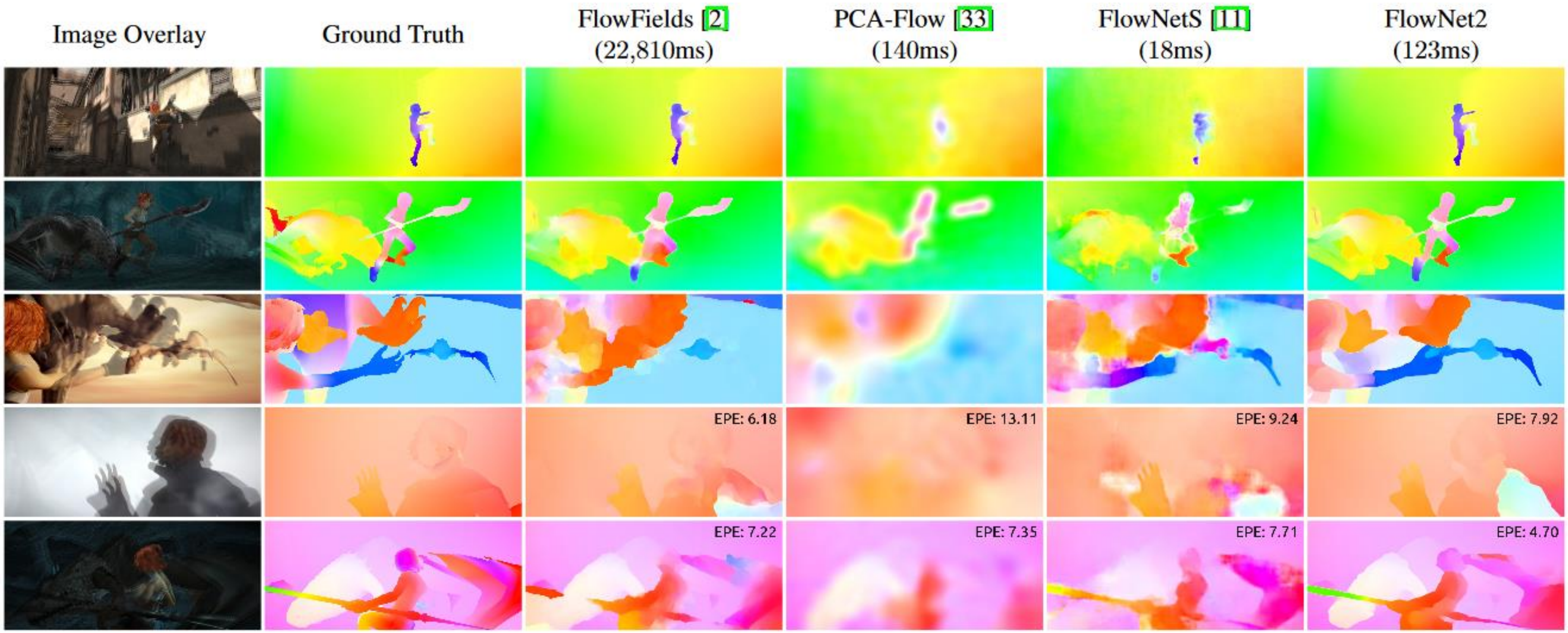
Modeling Motion: Optical Flow

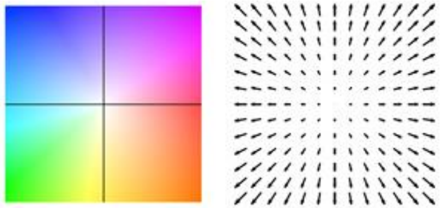


(a) Input Image (b) Prediction

Walker et al., "Dense Optical Flow Prediction from a Static Scene", ICCV 2015

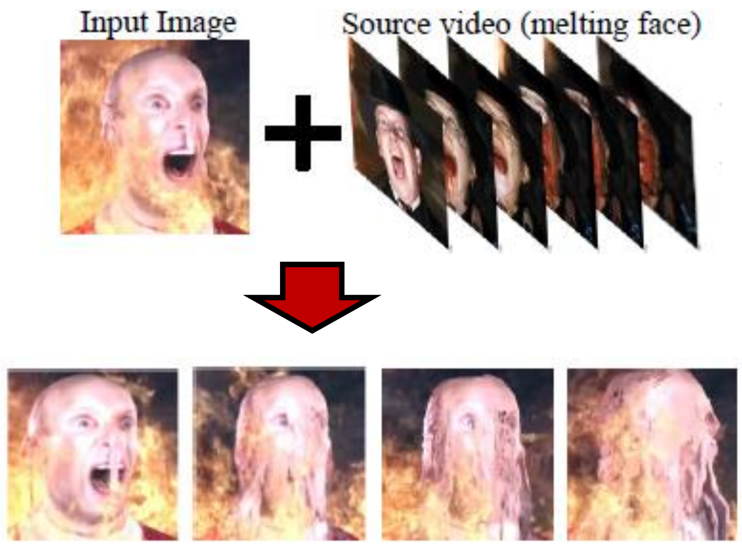
Modeling Motion: Optical Flow





Ilg et al., "FlowNet 2.0: Evolution of Optical Flow Estimation With Deep Networks", CVPR 2017

Transferring Motion

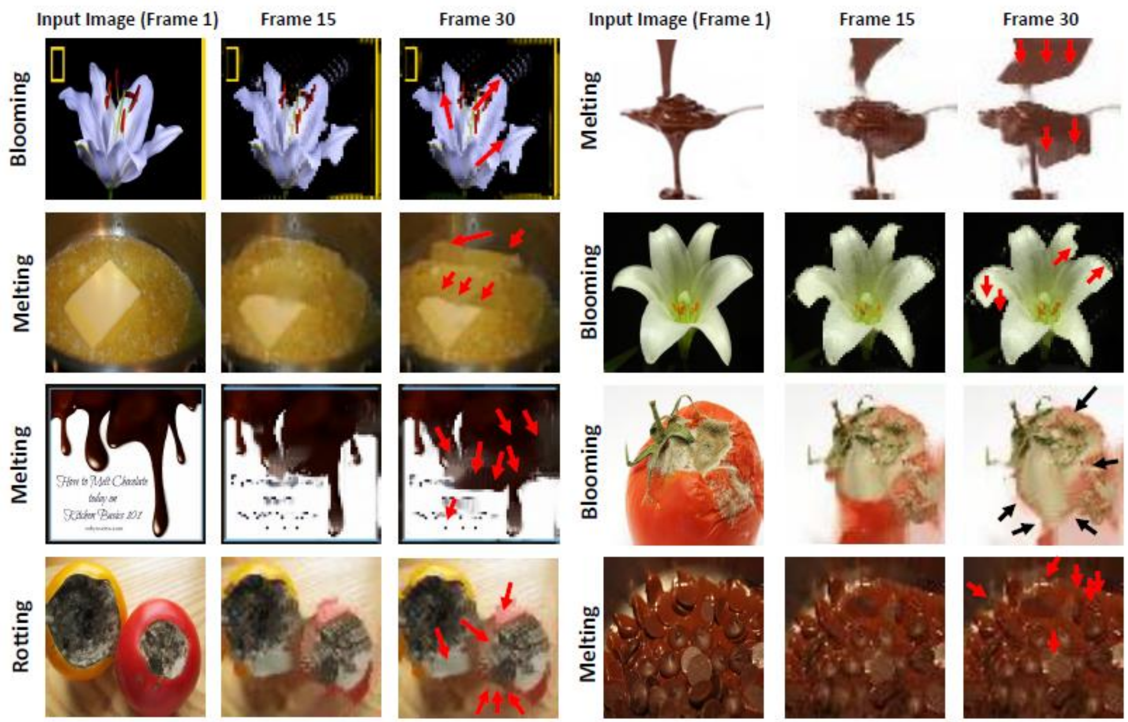


$$\mathcal{L}_{\text{flow}}(\mathbf{y}_{i-1}, \mathbf{y}_i; \mathbf{s}_{i-1}, \mathbf{s}_i) = \sum_l \frac{1}{C_l H_l W_l} \| \underbrace{\Xi(\mathbf{y}_{i-1}, \mathbf{y}_i)_l}_{\text{Optical flow in generated video}} - \underbrace{\Xi(\mathbf{s}_{i-1}, \mathbf{s}_i)_l}_{\text{Optical flow in source video}} \|_2^2$$

Key idea: Generate videos with similar flow patterns as source videos (+ many details).

Thomas, Song and Kovashka, in submission to ECCV 2018

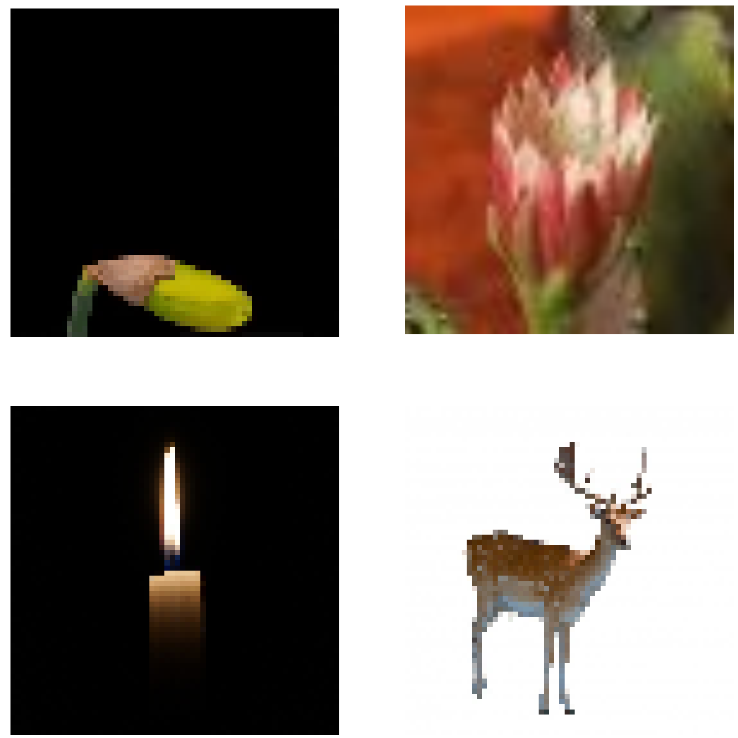
Transferring Motion



Thomas, Song and Kovashka, in submission to ECCV 2018

Transferring Motion

Baking



Blooming

Thomas, Song and Kovashka, in submission to ECCV 2018

Plan for this lecture

- Tracking how an object moves
- Modeling and replicating motion
- Recognizing human actions
- ConvNets for video

What is an action/activity?

Action: a transition from one state to another

- **What is the name of the action?**
- Who is the actor?
- How is the state of the actor changing?
- What (if anything) is being acted on?
- How is that thing changing?
- What is the purpose of the action (if any)?

Adapted from Derek Hoiem

How can we identify actions?

Motion

Pose

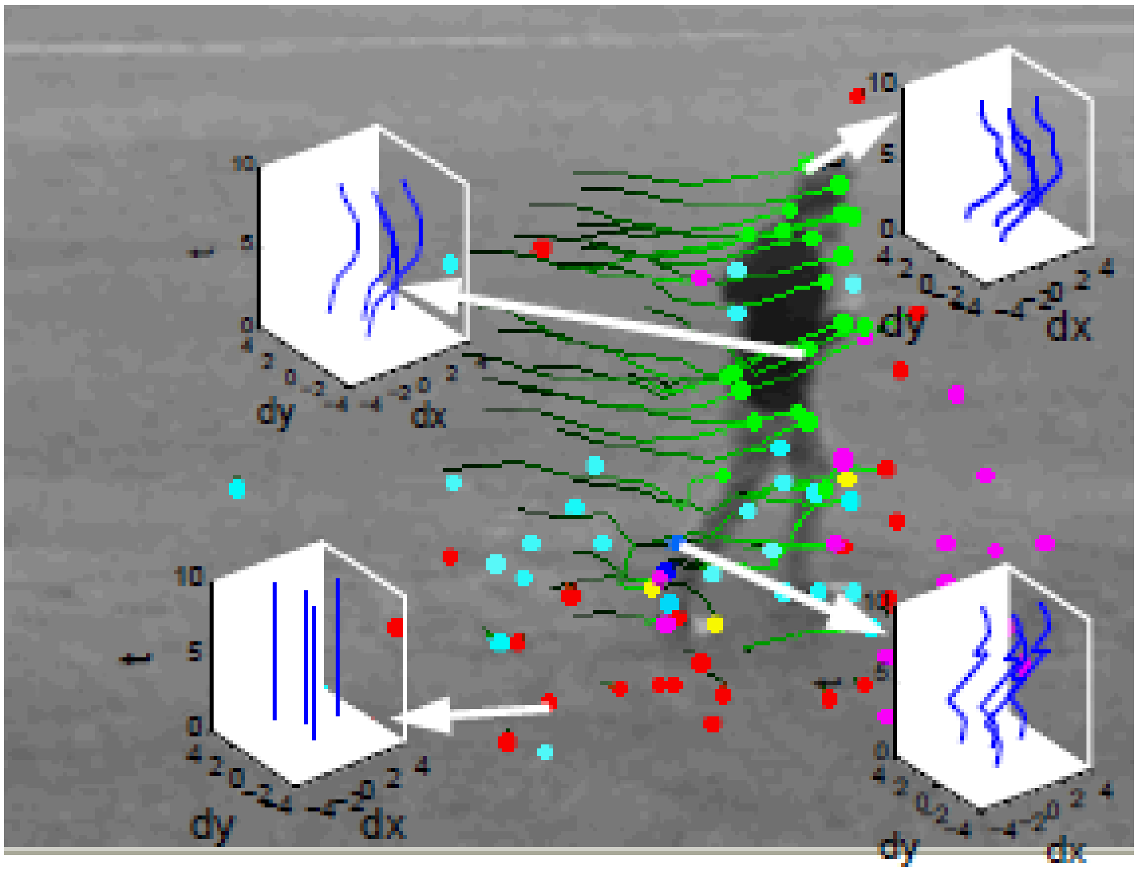
Held Objects

Nearby Objects

Derek Hoiem

Representing Actions

- Via tracked points

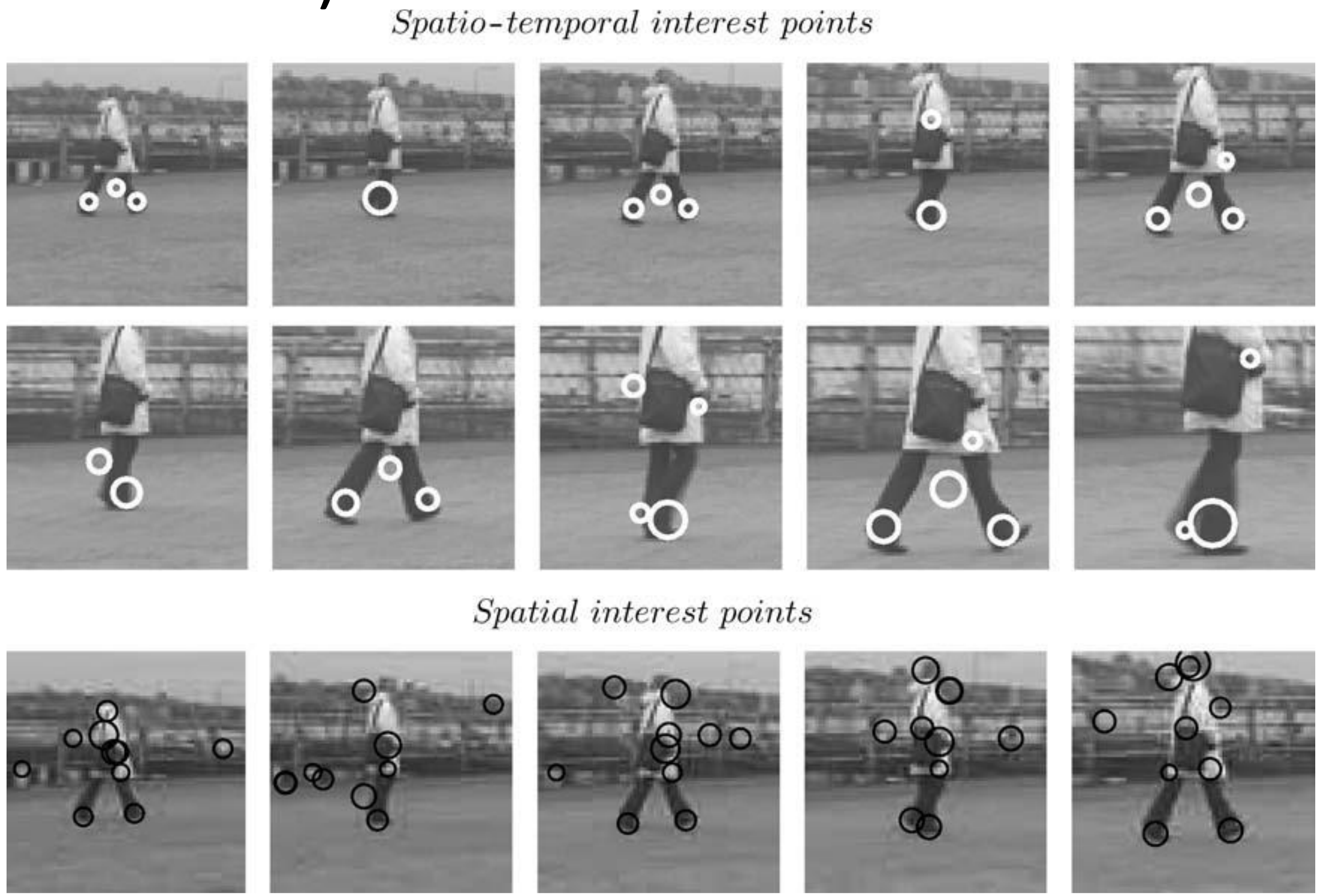


Adapted from Derek Hoiem

[Matikainen et al. 2009]

Representing Actions

- Via spatio-temporal interest points (corners in space+time)

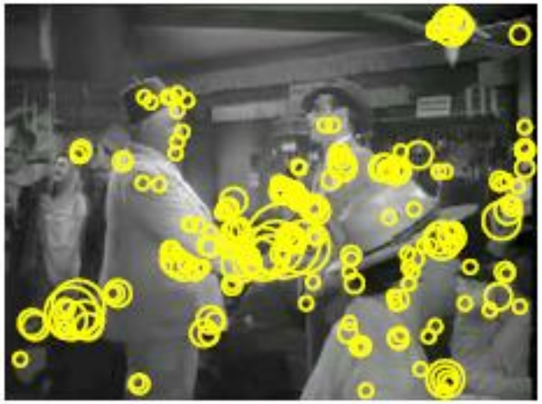


Adapted from Derek Hoiem

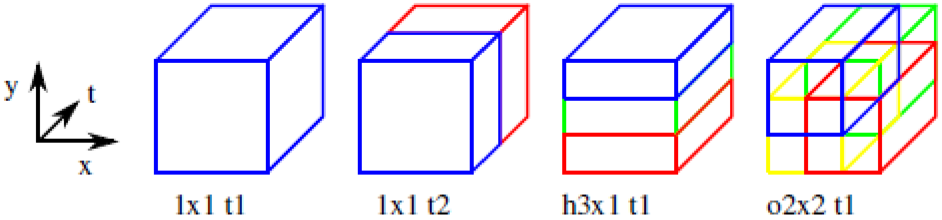
[Laptev 2005]

One Action Recognition Approach

- Space-time interest point detectors
- Descriptors
  - HOG, HOF
- Spatio-temporally-binned histograms
- SVMs with Chi-Squared Kernel



Interest Points



Spatio-Temporal Binning

Laptev et al., "Learning Realistic Human Actions from Movies", CVPR 2008

Adapted from Derek Hoiem

Results

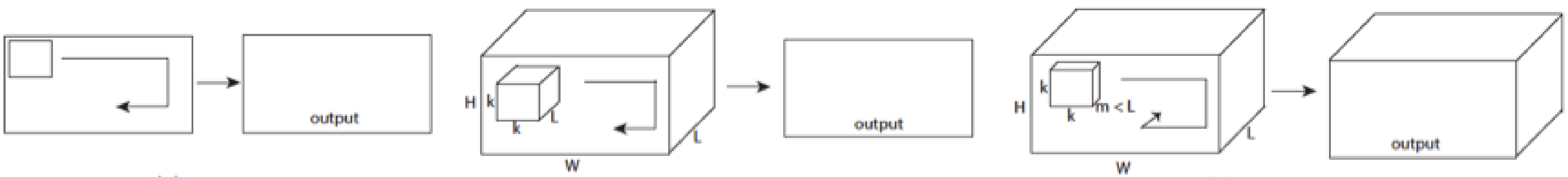
Task	HoG BoF	HoF BoF	Best channel	Best combination
KTH multi-class	81.6%	89.7%	91.1% (hof h3x1 t3)	91.8% (hof 1 t2, hog 1 t3)
Action AnswerPhone	13.4%	24.6%	26.7% (hof h3x1 t3)	32.1% (hof o2x2 t1, hof h3x1 t3)
Action GetOutCar	21.9%	14.9%	22.5% (hof o2x2 1)	41.5% (hof o2x2 t1, hog h3x1 t1)
Action HandShake	18.6%	12.1%	23.7% (hog h3x1 1)	32.3% (hog h3x1 t1, hog o2x2 t3)
Action HugPerson	29.1%	17.4%	34.9% (hog h3x1 t2)	40.6% (hog 1 t2, hog o2x2 t2, hog h3x1 t2)
Action Kiss	52.0%	36.5%	52.0% (hog 1 1)	53.3% (hog 1 t1, hof 1 t1, hof o2x2 t1)
Action SitDown	29.1%	20.7%	37.8% (hog 1 t2)	38.6% (hog 1 t2, hog 1 t3)
Action SitUp	6.5%	5.7%	15.2% (hog h3x1 t2)	18.2% (hog o2x2 t1, hog o2x2 t2, hog h3x1 t2)
Action StandUp	45.4%	40.0%	45.4% (hog 1 1)	50.5% (hog 1 t1, hof 1 t2)

Laptev et al., “Learning Realistic Human Actions from Movies”, CVPR 2008

Plan for this lecture

- Tracking how an object moves
- Modeling and replicating motion
- Recognizing human actions
- ConvNets for video

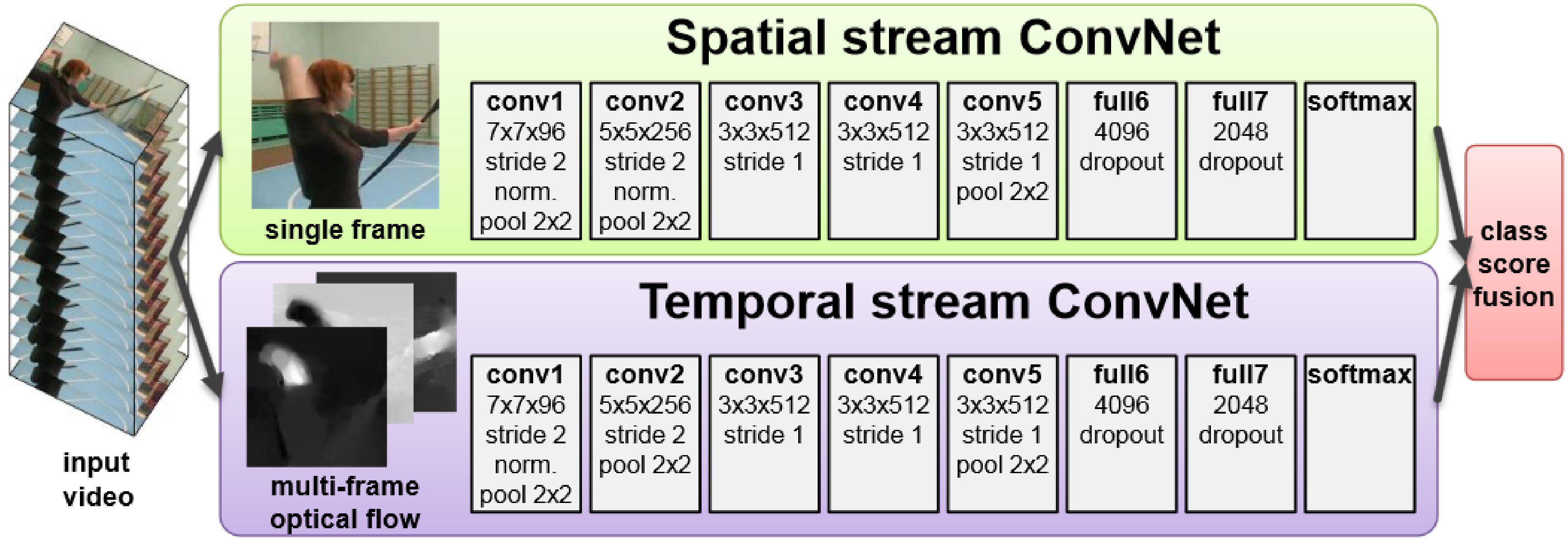
ConvNets for Video



(a) 2D convolution (b) 2D convolution on multiple frames (c) 3D convolution

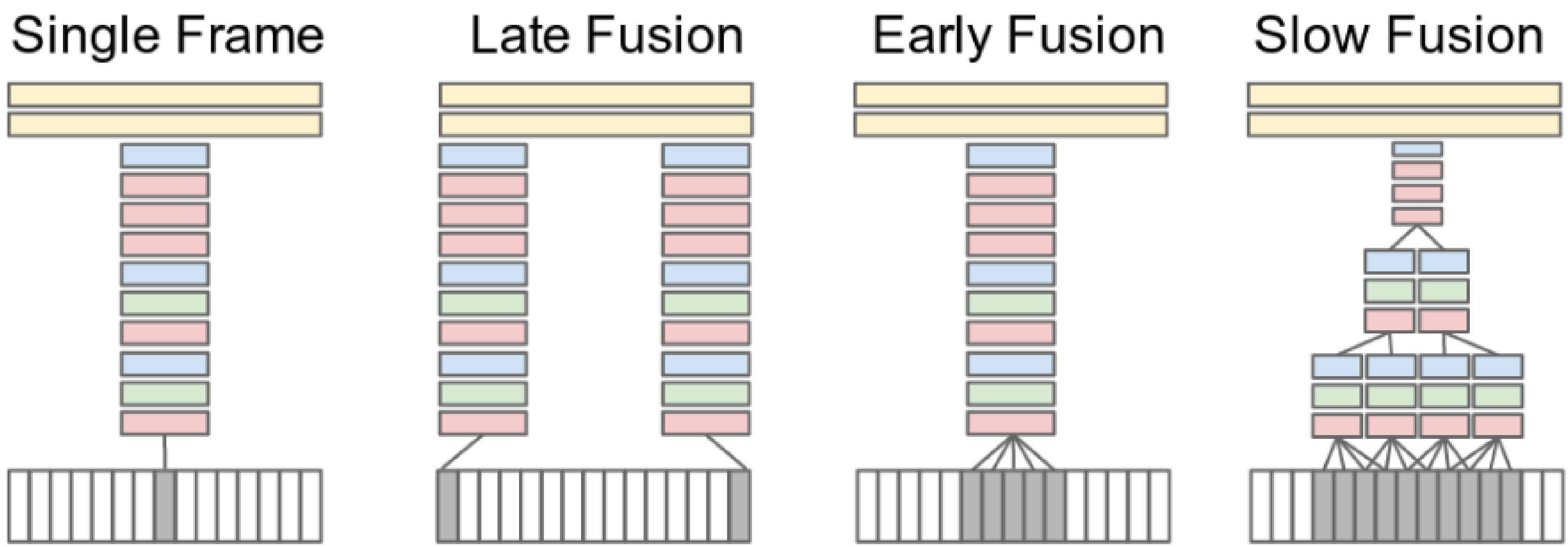
Tran et al., “Learning Spatiotemporal Features with 3D Convolutional Networks”, ICCV 2015

ConvNets for Video



Simonyan and Zisserman, “Two-Stream Convolutional Networks for Action Recognition in Videos”, NIPS 2014

ConvNets for Video



Karpathy et al., “Large-Scale Video Classification with Convolutional Neural Networks”, CVPR 2014

ConvNets for Video

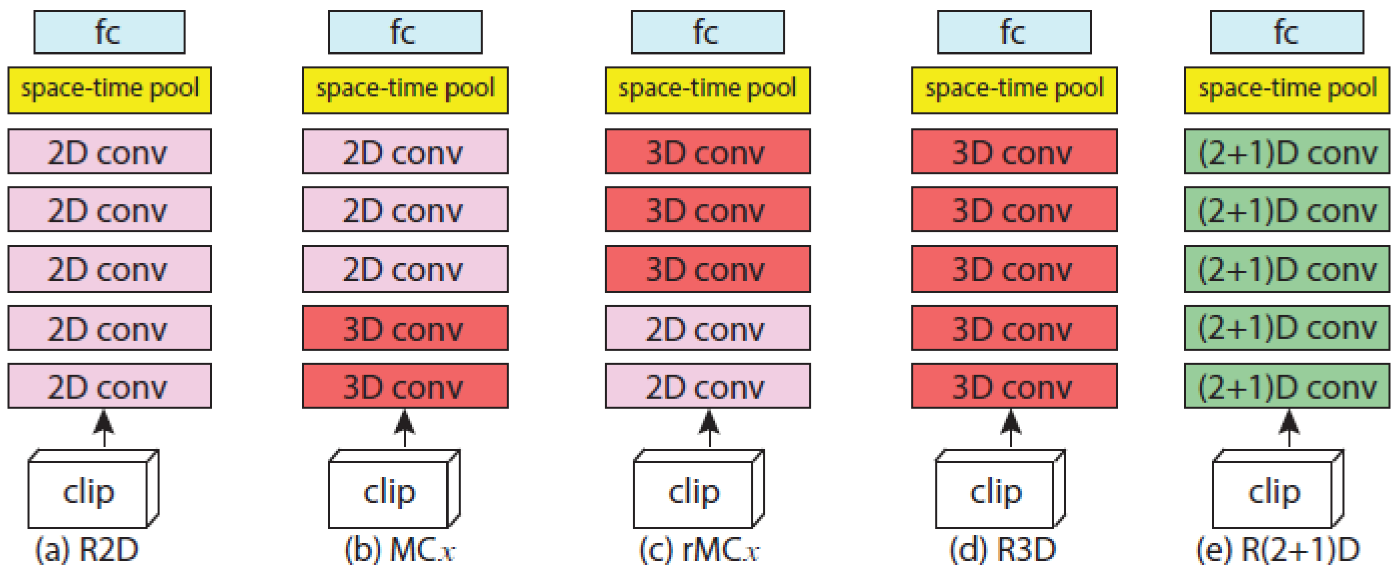
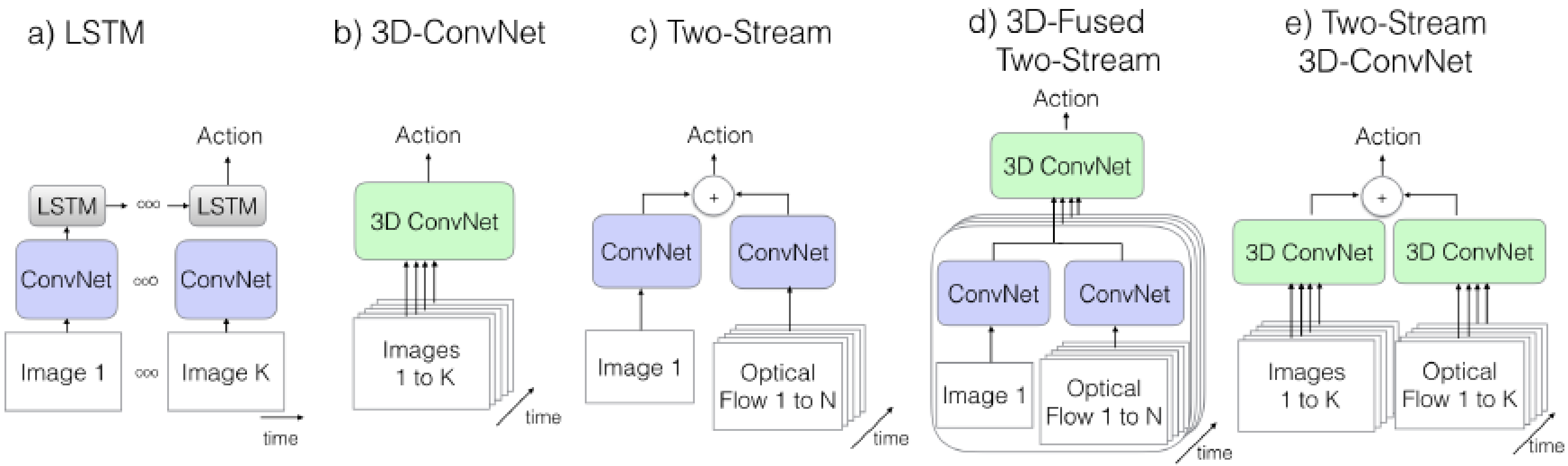


Figure 1. **Residual network architectures for video classification considered in this work.** (a) R2D are 2D ResNets; (b) MCx are ResNets with mixed convolutions (MC3 is presented in this figure); (c) rMCx use reversed mixed convolutions (rMC3 is shown here); (d) R3D are 3D ResNets; and (e) R(2+1)D are ResNets with (2+1)D convolutions. For interpretability, residual connections are omitted.

Tran et al., “A Closer Look at Spatiotemporal Convolutions for Action Recognition”, CVPR 2018

ConvNets for Video



Carreira and Zisserman, “Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset”, CVPR 2017