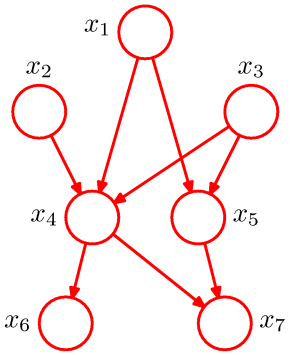
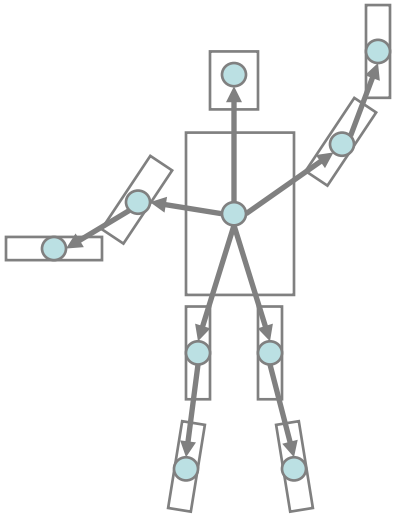
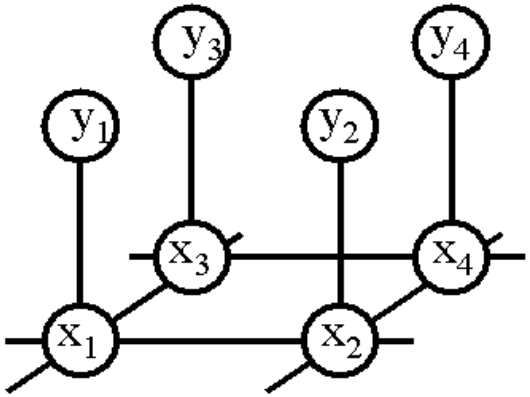
Introduction to Supervised Learning







Lecture 10: Introduction to Structured Prediction

Iasonas Kokkinos

i.kokkinos@cs.ucl.ac.uk

University College London

Tentative Course Schedule: generative & discriminative

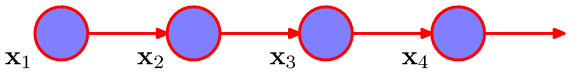
- 1st week: Introduction, fundamental concepts
- 2nd week: Linear Regression
- 3rd week: Logistic Regression
- 4th week: Support Vector Machines
- 5th week: Ensemble Models (Adaboost, Random Forests)
- 6th week: Unsupervised learning (K-means, PCA, Sparse Coding)
- 7th week: Deep Learning (neural networks, backpropagation, SGD)
- 8th week: Probabilistic modelling (hidden variable models, EM)
- 9th week: Intro to Structured Prediction (Random Fields, Graphical Models)
- 10th week: review and applications

Lecture outline

- Introduction
  - Graphs and computation
  - Graphical models
- Chain-structured graphical models
- Tree-structured graphical models
- Loopy graphical models

Problem 1: measuring the length of a queue

while only complaining to your closest neighbors

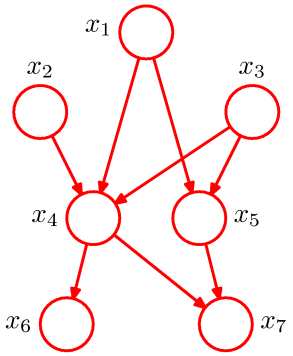


Problem 2: two queues

Problem N: N queues

Lecture outline

- Introduction
  - Computation on graphs
  - Graphical models
- Chain-structured graphical models
- Tree-structured graphical models
- Loopy graphical models



Graphical models

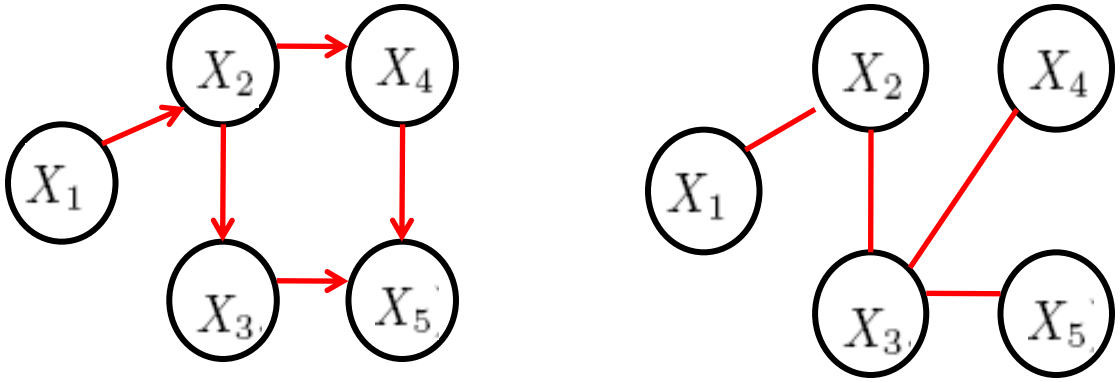
Blend of graph theory and probability

Graph nodes: random variables

Graph edges: relationships

Directed graphs: Bayesian Networks

Undirected graphs: Markov Random Fields



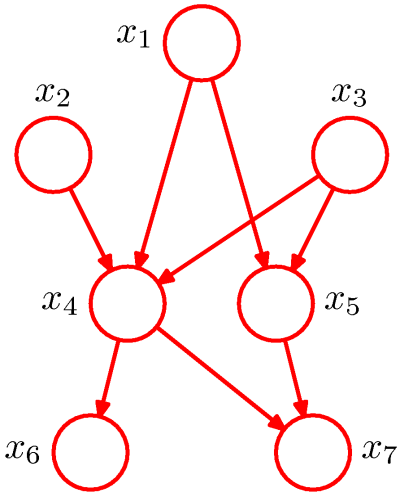
Bayesian Networks

Directed Acyclic Graph $G(V, E)$

Nodes - V : random variables

Edges - E : dependencies

Leave from `parents', arrive at children



Distribution of each child conditioned on parent: $P(X_v|X_{\pi_v})$

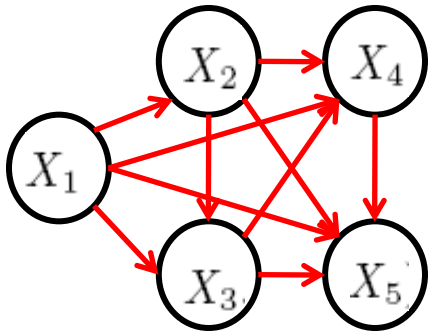
$$P(X_v|\emptyset) = P(X_v)$$

Joint probability distribution: $P(X) = \prod_v P(X_v|X_{\pi_v})$

Conditional independencies

We can trivially obtain a bayesian network for a probability distribution

$$P(X_1, X_2, X_3, X_4, X_5) = P(X_1)P(X_2|X_1)P(X_3|X_2, X_1)P(X_4|X_3, X_2, X_1)P(X_5|X_4, X_3, X_2, X_1)$$

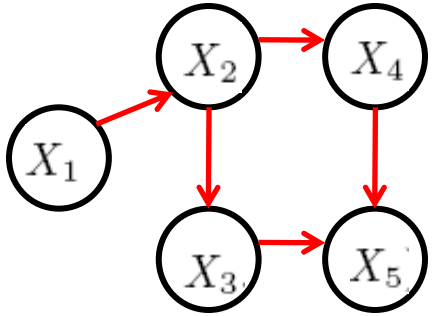


Fully connected graph!

Bayesian net

Non-trivial cases: some conditional independence

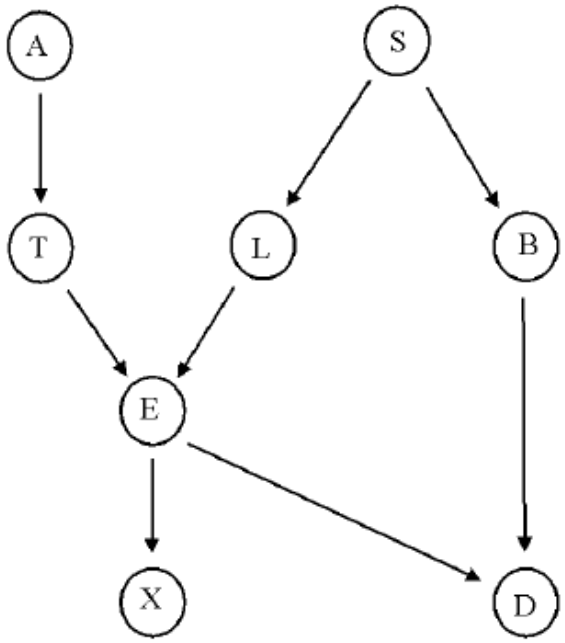
$$\begin{aligned} P(X_3|X_2, X_1) &= P(X_3|X_2) \\ P(X_4|X_3, X_2, X_1) &= P(X_4|X_2) \\ P(X_5|X_1, X_2, X_3, X_4, X_5) &= P(X_5|X_3, X_4) \end{aligned}$$



Main interest: exploit independencies for training & inference

A Bayesian Network-based Expert System

- `Asia Network'
  - A: trip to Asia
  - T: turbeculosis
  - S: smoking
  - L: Lung Cancer
  - B: Bronchitis
  - E: Turbeculosis/Lung Cancer
  - X-ray results
  - D: Dyspnea



- Given: X-rays, Dyspnea, patient went to Asia, patient smokes
- Wanted: posterior probability of Bronchitis

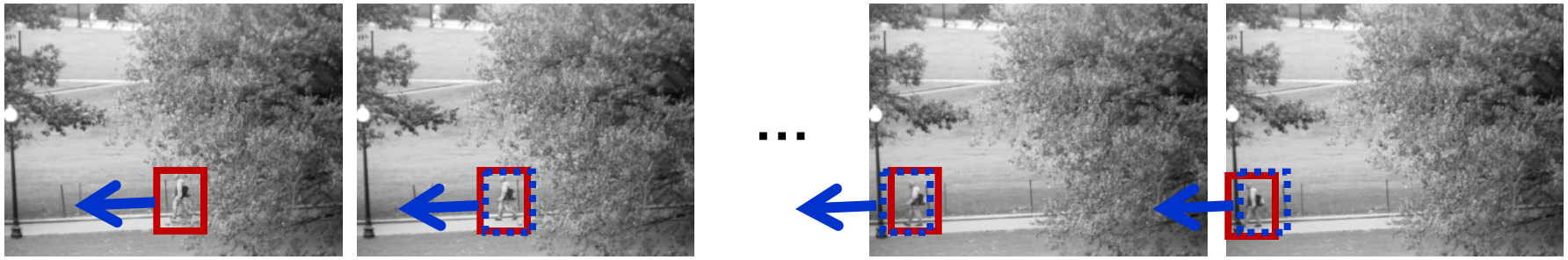
Detection vs. Tracking

t=1 t=2 … t=20 t=21

Detection vs. Tracking

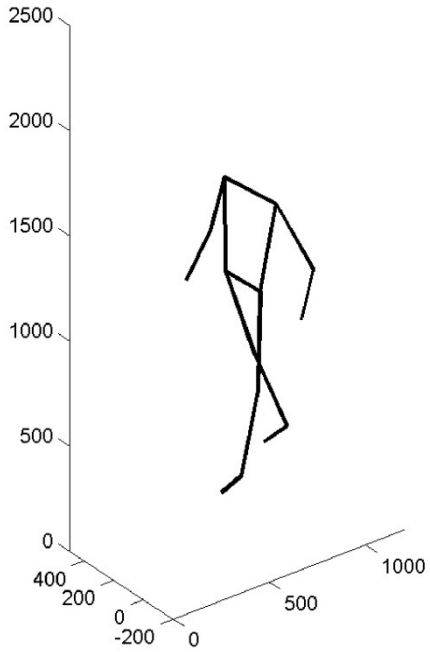
- Detection: each frame independently

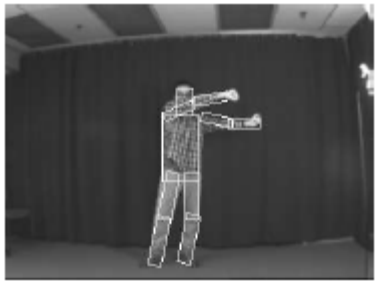
Detection vs. Tracking



- Tracking with *dynamics*: combine image measurements with our expectation of the object's motion.
  - Restrict search for the object
  - Measurement noise is reduced by trajectory smoothness.

`State space’

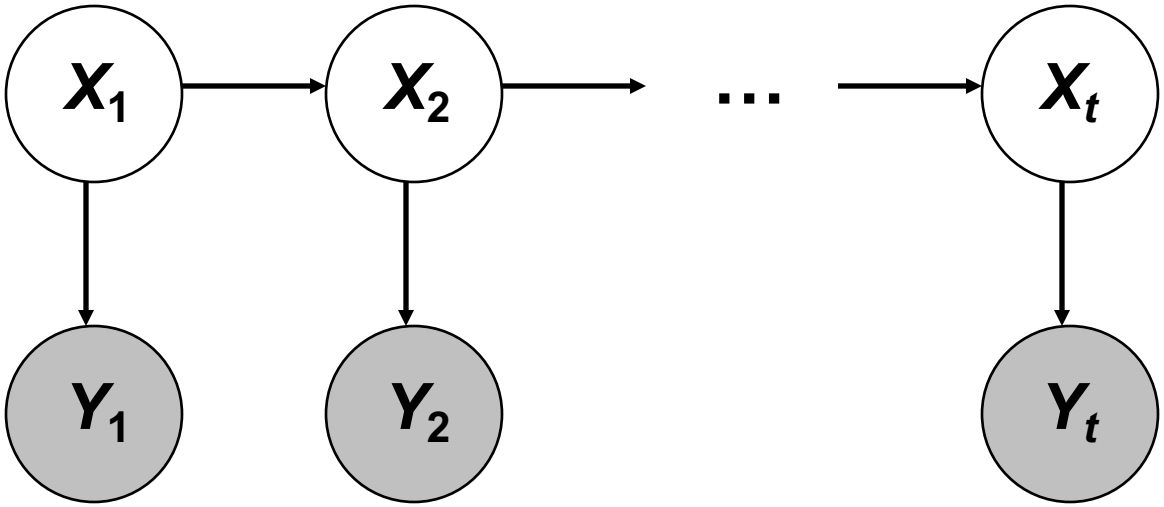




- X: System state (parameters of interest)
- Y: Measurement (what we observe)

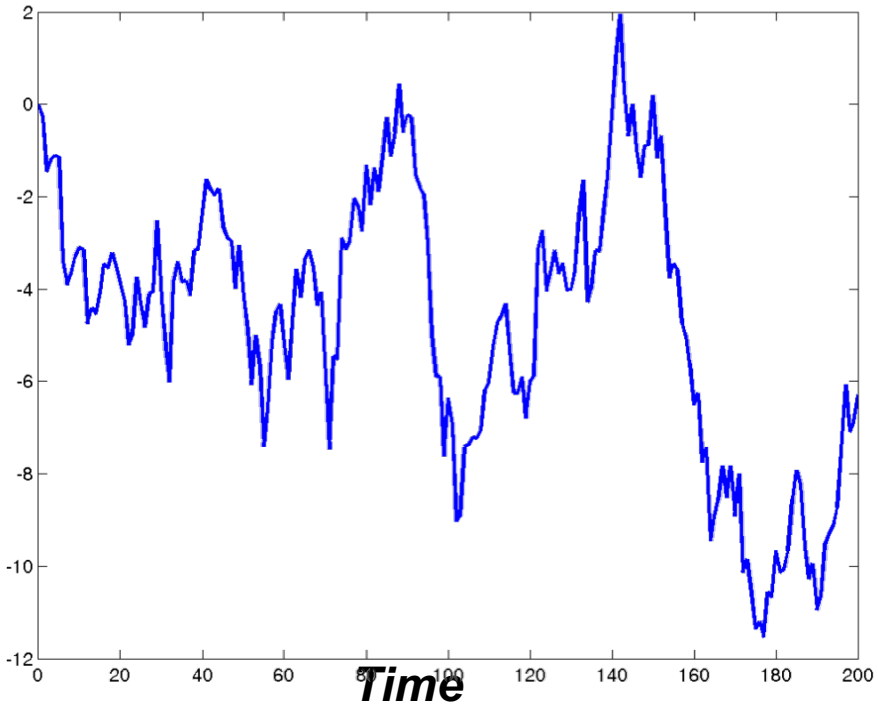
General Model for Tracking

- The moving object of interest is characterized by an underlying *state* X
- State X gives rise to *measurements* or *observations* Y
- At each time t, the state changes to X_t and we get a new observation Y_t



Simple Linear Dynamics

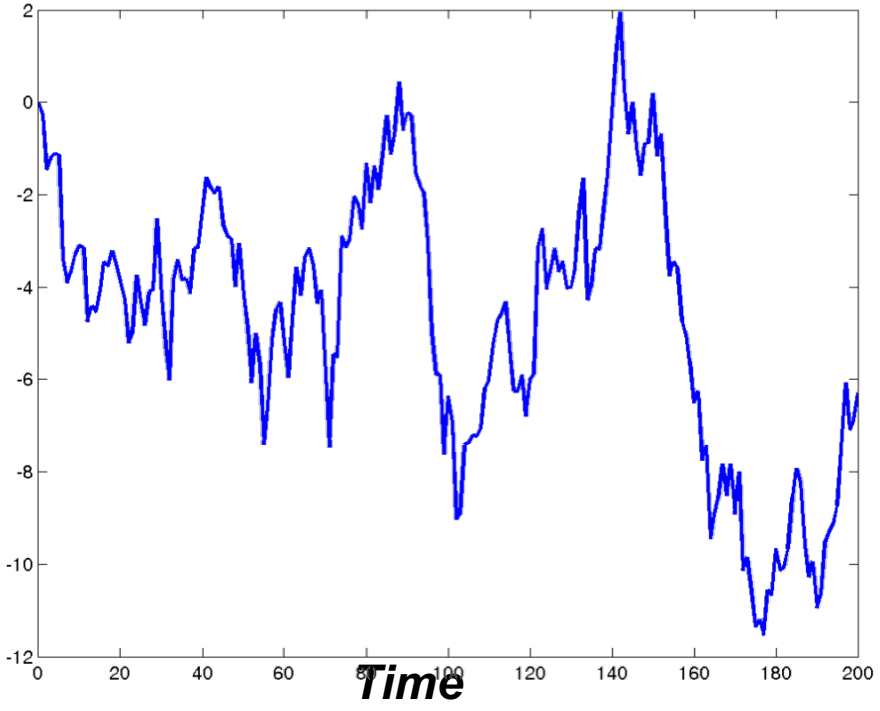
Brownian Motion



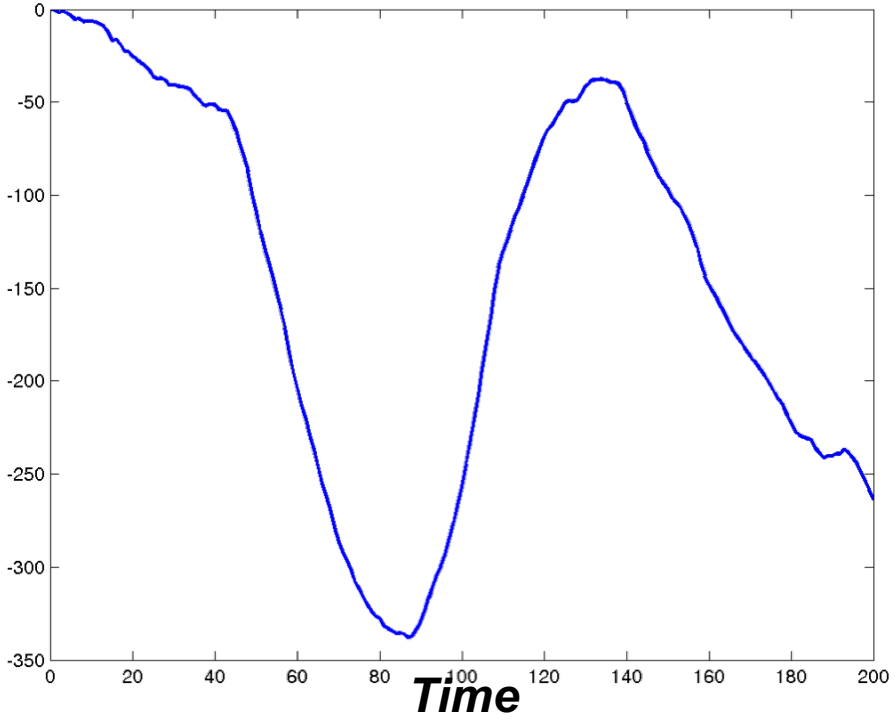
$$x_{t+1} = x_t + w_t$$

Simple Linear Dynamics

Brownian Motion



Constant Velocity



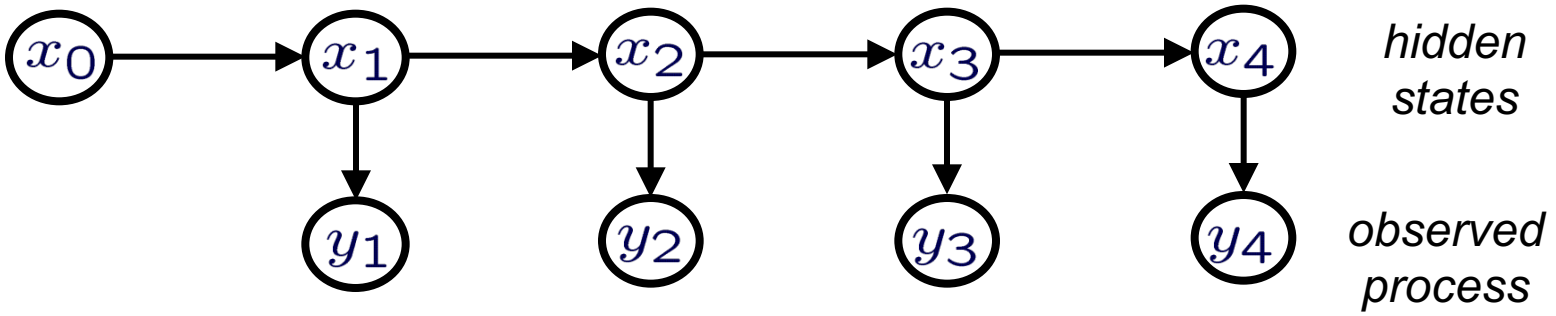
$$x_{t+1} = x_t + w_t$$

$$\begin{bmatrix} x_{t+1} \\ \delta_{t+1} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_t \\ \delta_t \end{bmatrix} + w_t$$

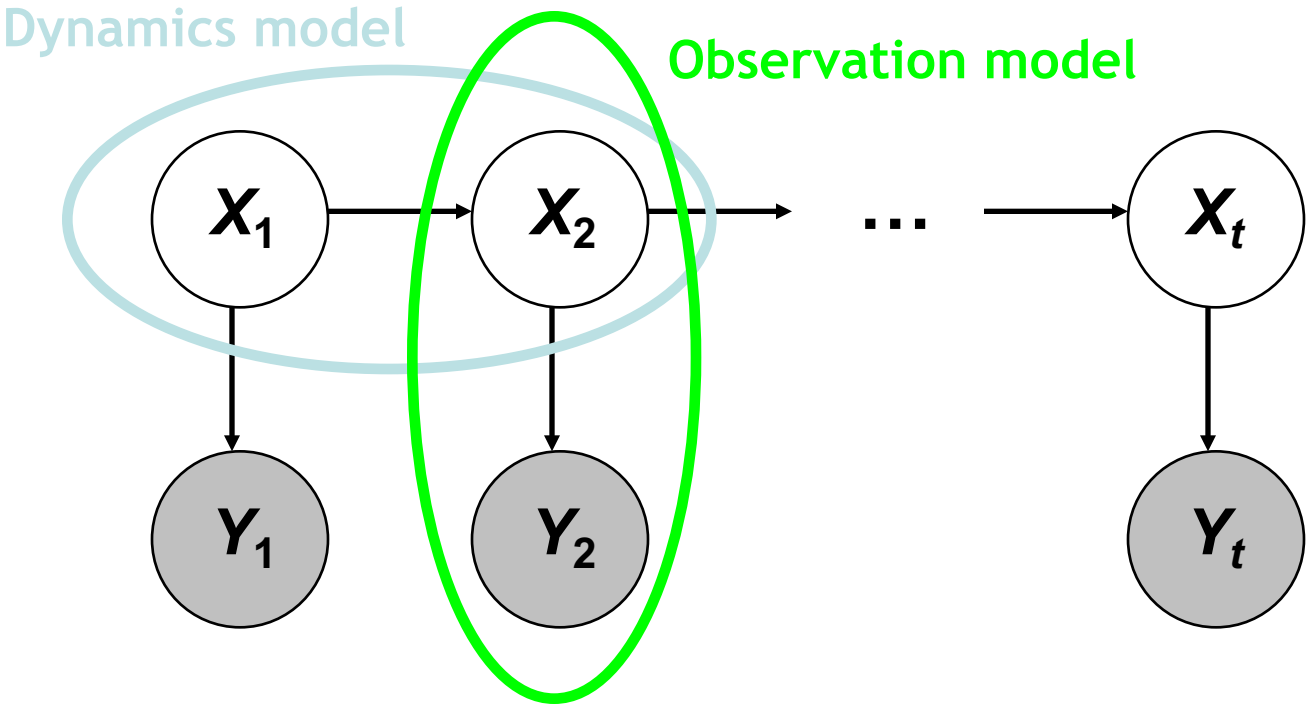
Linear Dynamical Systems

- Noise in State, noise in Observations

$$
\begin{aligned}
x_t &= Ax_{t-1} + v_t, \quad v_t \propto N(0, \Sigma) \\
y_t &= Cx_t + w_t, \quad w_t \propto N(0, Q) \\
x_1 &= \mu_0 + u, \quad u \propto N(0, V)
\end{aligned}
$$



Markov Assumptions



$$P(X_t|X_{t-1}, X_{t-2}, X_1) = P(X_t|X_{t-1})$$

$$P(Y_t|X_1, \ldots X_T, Y_1, \ldots Y_{t-1}, Y_{t+1}, \ldots Y_T) = P(Y_t|X_t)$$

Tracking as Induction

- Base case:
  - Prior prediction of state (in absence of any evidence): $P(X_0)$
  - At the first frame, *correct* this given the value of $Y_0=y_0$

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

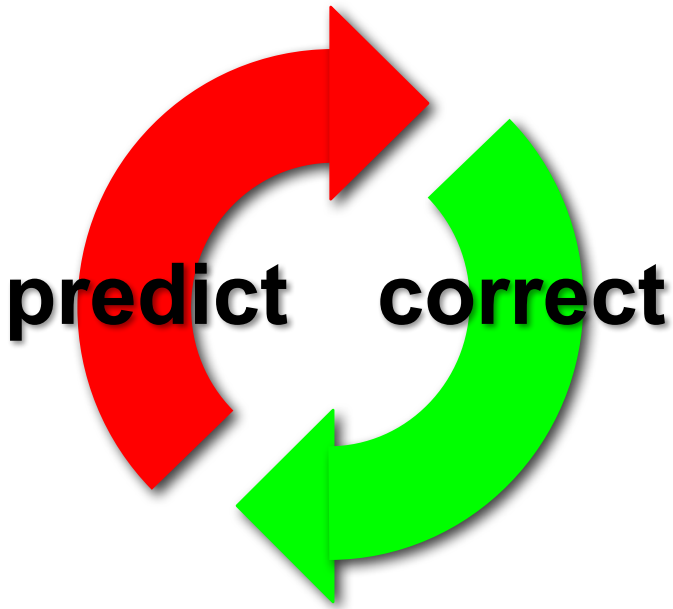
Posterior prob. of state given measurement

Likelihood of measurement

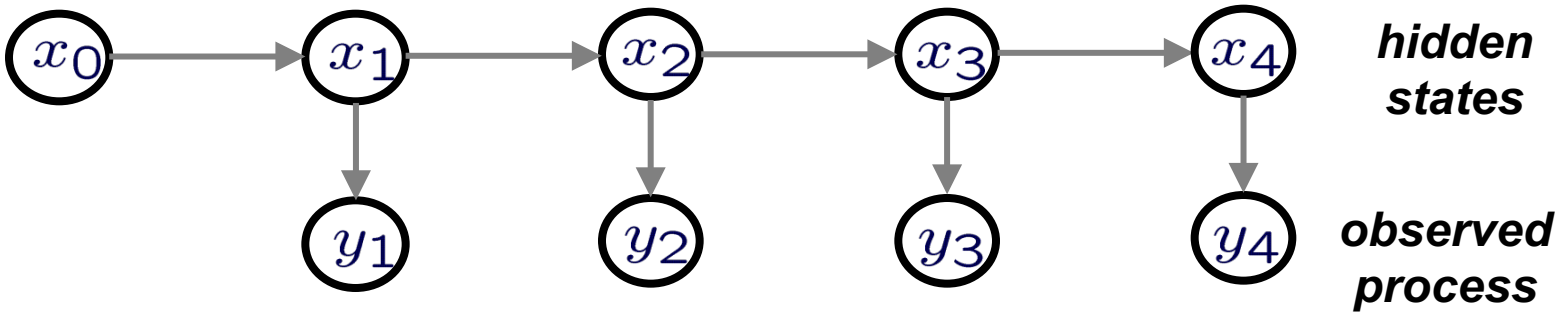
Prior of the state

Tracking as Induction

- Base case:
  - Assume we have initial prior that predicts state in absence of any evidence: $P(X_0)$
  - At the first frame, *correct* this given the value of $Y_0=y_0$
- Given corrected estimate for frame t:
  - Predict for frame t+1
  - Correct for frame t+1



Hidden Markov Models



- Dependent sequence of observations
- Conditioned on state sequence, observations become independent

Parameters

Observation probabilities $P(y_t|x_t)$

$\pi(y_t = k|x_t = m) = n_{k,m}$ discrete

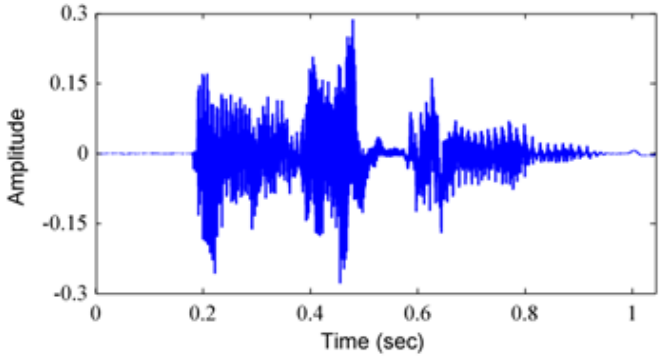
$P(y_t|x_t = k) = N(y_t; \mu_k, \Sigma_k)$ continuous

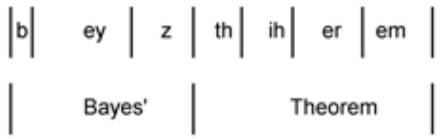
Transition probabilities $\pi(x_{t+1} = l|x_t = m) = \alpha_{l,m}$

Starting probabilities $\pi(x_0 = m) = \beta_m$

HMMs & Speech Recognition

- Input: Air pressure signal

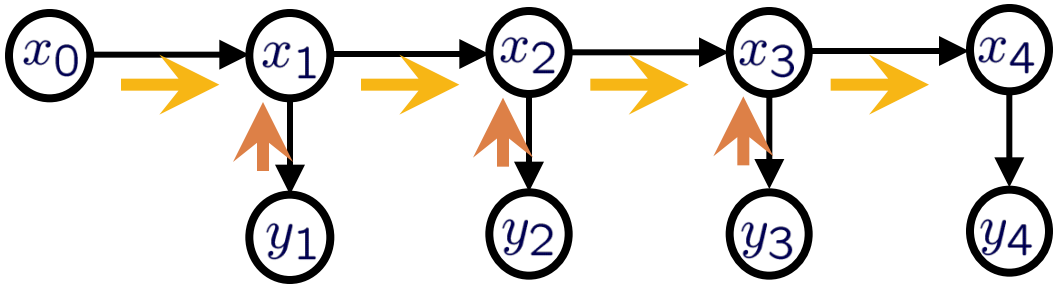




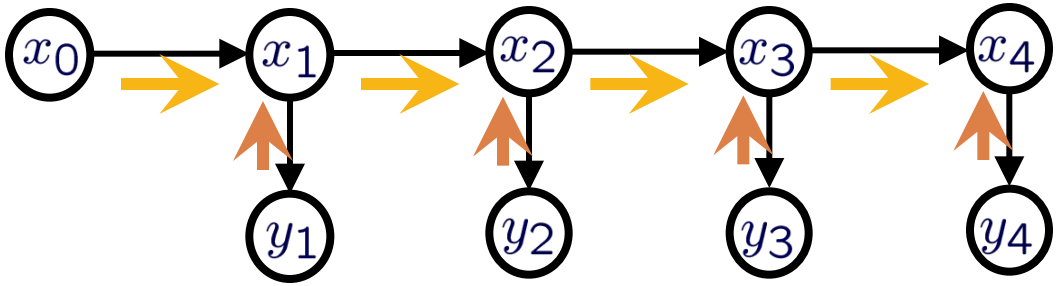
- Output: words/phrases
- Probabilistic model: signal given word
- Hidden Variables:
  phonemes (/ae/,/ey/,/z/…)
  phone subunits
- Hidden variables: not independent
  - Show,shoe,she,sell,sea,
  - Sv?
    (Sven,svelte,..)
  - Sb?

Inference Tasks

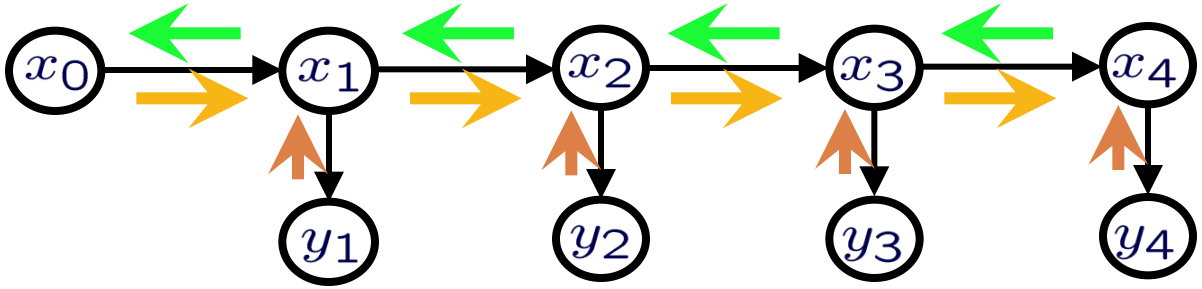
- Prediction



- Filtering
  (Kalman Filter)

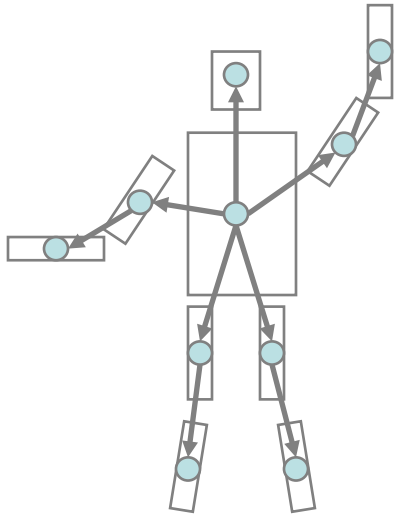


- Smoothing



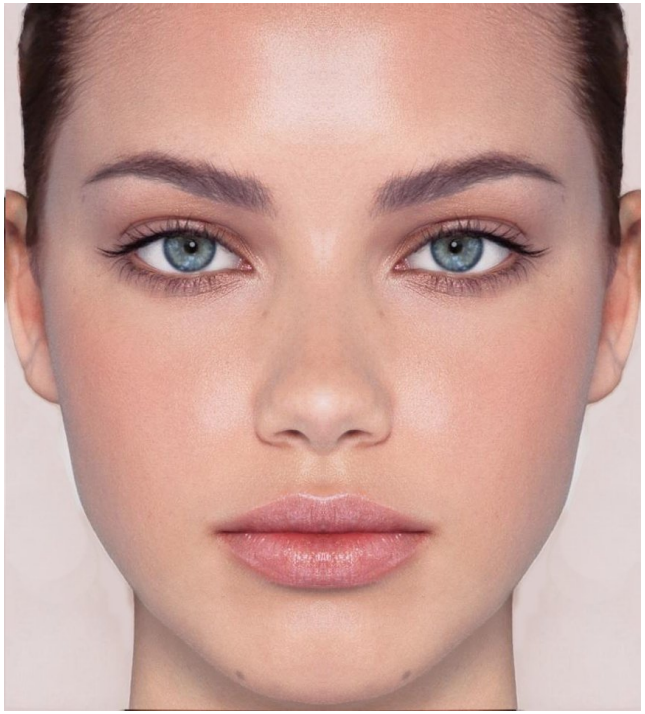
Lecture outline

- Introduction
- Chain-structured graphical models
- Tree-structured graphical models
  - Pictorial Structures
  - Max/Sum-Product algorithm
- Loopy Graphical Models



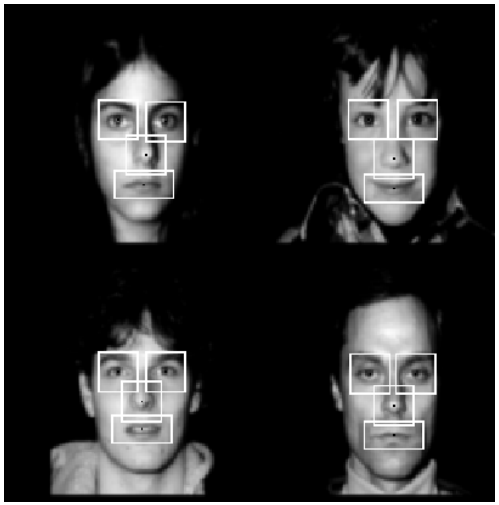
Problem N: N queues

Part-based Models

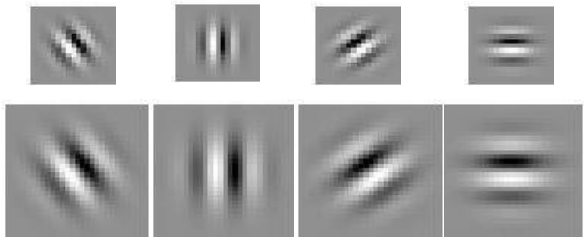


Unary terms (appearance model)

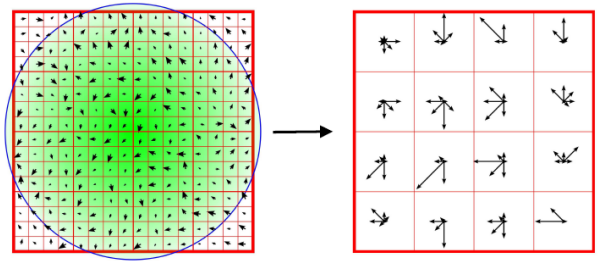
- Consider a patch of the image around the location of the part
- Construct statistical model for appearance at part location



Filterbank responses



SIFT features

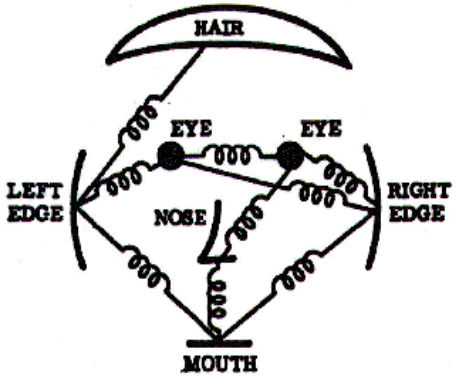


$\Phi_1(l)$ $\Phi_2(l)$ $\Phi_3(l)$ $\Phi_4(l)$ $\Phi_5(l)$

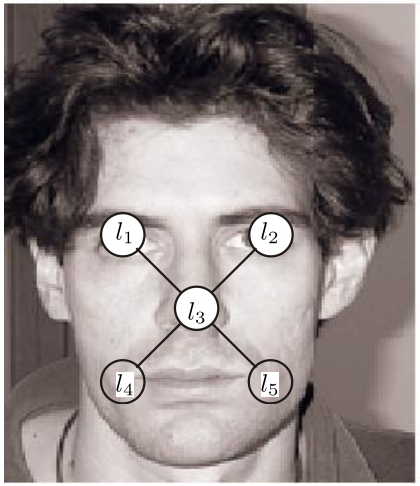
Left eye **Right eye** **Nose** **Left mouth** **Right mouth**

Pairwise terms (deformation model)

- Deformable objects as physical systems
  - Masses: driven towards good image locations
  - Springs: enforce relations among parts



- Star models (tree-structured models)
  - Pick single part as root (e.g. the nose)
  - Model other parts (eyes, mouth corners) w.r.t. root



- Probability of configuration $L = (l_1, l_2, l_3, l_4, l_5)$

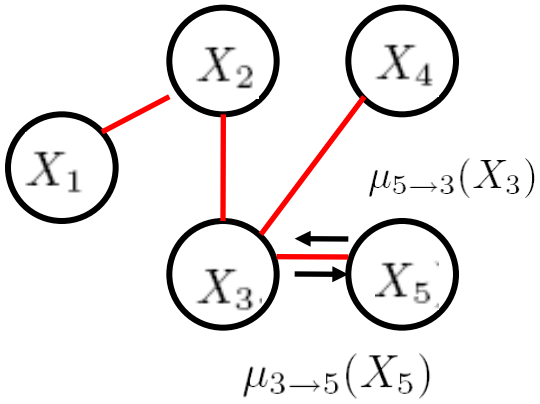
$$P(L|I) \propto P(I|L)P(L) = \prod_i \Phi_i(I_i|l_i) \prod_{(i,j)\in\mathcal{C}} \Psi_{i,j}(l_i, l_j)$$

Unary Terms Pairwise Terms

- How can we maximize over L?

Lecture outline

- Introduction
- Chain-structured graphical models
- Tree-structured graphical models
  - Pictorial Structures
  - Max/Sum-Product algorithm
- Loopy Graphical Models

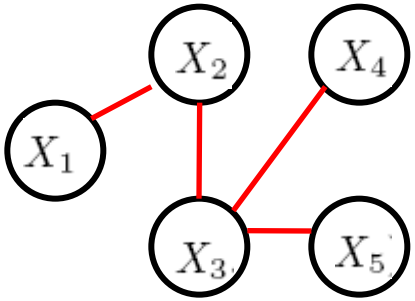


Computation on tree-structured Graphs

- Assume we want to find the most likely value of X_1

$$P(X) = \frac{1}{Z}\prod_{i=1}^{5}\Phi_i(X_i)\prod_{(i,j)\in\mathcal{C}}\Psi_{i,j}(X_i,X_j)$$

$$\mathcal{C} = \{(1,2),(2,3),(3,4),(3,5)\}$$



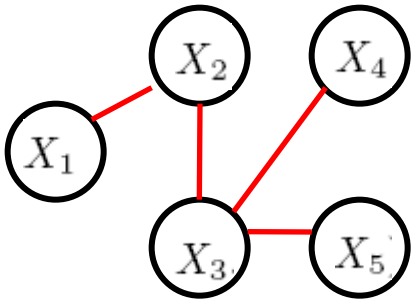
- Brute-force maximization:

$$\max P(X_1) = \max_{X_2,X_3,X_4,X_5} P(X_1,X_2,X_3,X_4,X_5)$$

$$|X_1| = K \rightarrow K^4$$

- Exploit factorization

Exploiting the factorization

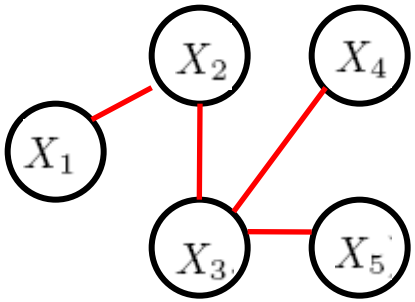


$$P(X) = \frac{1}{Z}\prod_{i=1}^{5}\Phi_i(X_i)\prod_{(i,j)\in\mathcal{C}}\Psi_{i,j}(X_i,X_j)$$

$$\mathcal{C} = \{(1,2),(2,3),(3,4),(3,5)\}$$

$\max P(X_1)$

Exploiting the factorization

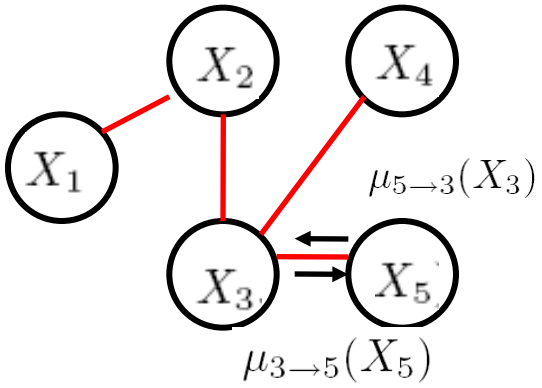


$$P(X) = \frac{1}{Z}\prod_{i=1}^{5}\Phi_i(X_i)\prod_{(i,j)\in\mathcal{C}}\Psi_{i,j}(X_i,X_j)$$

$$\mathcal{C} = \{(1,2),(2,3),(3,4),(3,5)\}$$

$$\begin{aligned}
& \max P(X_1) \\
= \; & \max_{X_2,X_3,X_4,X_5}\Phi(X_1)\Phi(X_2)\Phi(X_3)\Phi(X_4)\Phi(X_5)\Psi(X_1,X_2)\Psi(X_2,X_3)\Psi(X_3,X_4)\Psi(X_3,X_5) \\
= \; & \Phi(X_1)\max_{X_2}\Phi(X_2)\Psi(X_1,X_2)\max_{X_3,X_4}\Phi(X_3)\Psi(X_2,X_3)\Phi(X_4)\Psi(X_3,X_4)\underbrace{\max_{X_5}\Phi(X_5)\Psi(X_3,X_5)}_{\mu_5(X_3)} \\
= \; & \Phi(X_1)\max_{X_2}\Phi(X_2)\Psi(X_1,X_2)\max_{X_3}\Phi(X_3)\Psi(X_2,X_3)\mu_5(X_3)\underbrace{\max_{X_4}\Phi(X_4)\Psi(X_3,X_4)}_{\mu_4(X_3)} \\
= \; & \Phi(X_1)\max_{X_2}\Phi(X_2)\Psi(X_1,X_2)\underbrace{\max_{X_3}\Phi(X_3)\Psi(X_2,X_3)\mu_5(X_3)\mu_4(X_3)}_{\mu_3(X_2)} \\
= \; & \Phi(X_1)\underbrace{\max_{X_2}\Phi(X_2)\Psi(X_1,X_2)\mu_3(X_2)}_{\mu_2(X_1)}
\end{aligned}$$

Max-Product algorithm

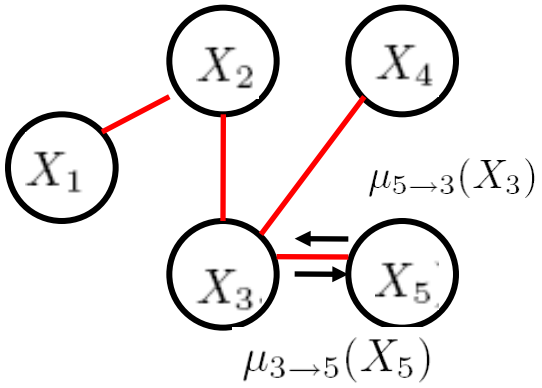


- Distributed computation on a graph
  - `message-passing'
- Message from node i to node j

$$\mu_{i\to j}(X_j) \quad = \quad \max_{X_i} \Phi_i(X_i)\Psi_{i,j}(X_i,X_j) \prod_{k\in\mathcal{N}(i)\setminus j} \mu_{k\to j}(X_i)$$

  - `Given all I know from the rest, this is where you should be'
- Beliefs: $$B_j(X_j) \quad = \quad \Phi_j(X_j) \prod_{i\in\mathcal{N}(j)} \mu_{i\to j}(X_j)$$
- At convergence $$B_j(X_j) = \max P(X_j)$$

Sum-Product algorithm



- Distributed computation on a graph
  - `message-passing'

- Message from node i to node j

$$\mu_{i\to j}(X_j) = \sum_{X_i} \psi_i(X_i)\psi_{i,j}(X_i, X_j) \prod_{k\in\{\mathcal{N}(i)\setminus j\}} \mu_{k\to i}(X_i)$$

  - `Given all I know from the rest, this is where you should be'

- Beliefs: $$B_j(X_j) = \Phi_j(X_j) \prod_{i\in\mathcal{N}(j)} \mu_{i\to j}(X_j)$$

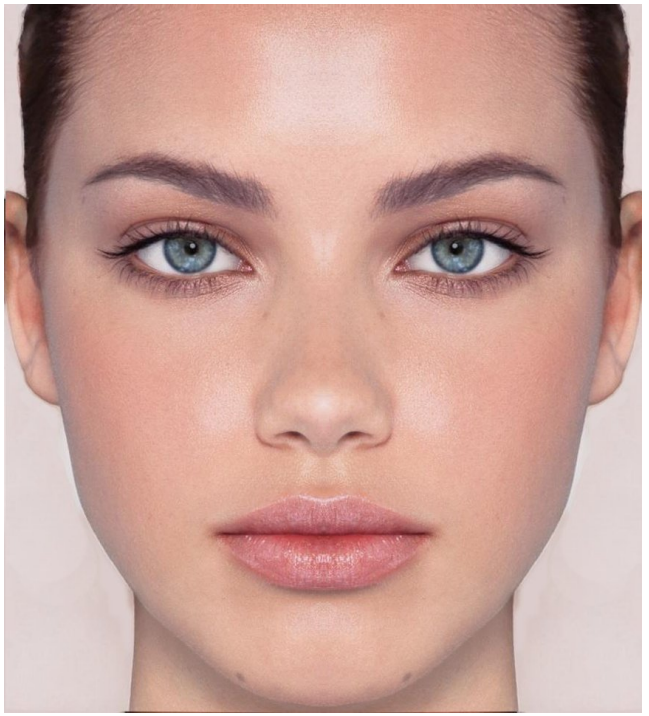
- At convergence $$P(X_i) = B(X_i)$$

Max/Sum-Product algorithms

- Asychronous inference
- Local computations
- Guaranteed optimality for tree-structured graphs
- Equivalent algorithms:
  - Tracking: Kalman Filtering (Sum-Product)
  - HMMs: Alpha-Beta (Sum-Product)
  - Coding/HMMs: Viterbi Algorithm (Max-Product)
  - Bayesian Nets: Belief Propagation (Sum-Product)
- Max Product: Dynamic Programming

Deformable Part Models (DPMs)

$$\mathbf{S}(\mathbf{x}) = \sum_{p=1}^{P} U_p(x_p) + B_p(x, x_p)$$



Local appearance

$$U_p(x_p) = \langle w_p, H(x_p) \rangle$$

Pairwise compatibility

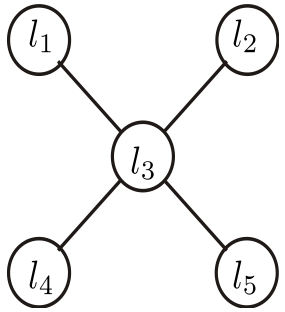
$$B_p(x, x_p) = -(h - h_p - \hat{h}_p)^2 \eta - (v - v_p - \hat{v}_p)^2 \nu$$

Object score

$$S(x) = \sum_{p=1}^{P} \max_{x'} \left[U_p(x') + B_p(x, x') \right]$$

Pictorial Structures

- Graphical model
  - Nodes: part locations
  - Edges: dependencies - relative locations



- Object localization: maximization with respect to $L = (l_1, l_2, l_3, l_4, l_5)$

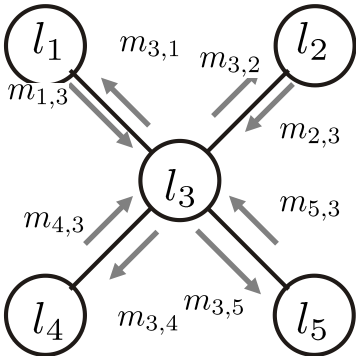
$$P(L|I) \propto P(I|L)P(L) = \prod_i \Phi_i(I_i|l_i) \prod_{(i,j)\in\mathcal{C}} \Psi_{i,j}(l_i, l_j)$$

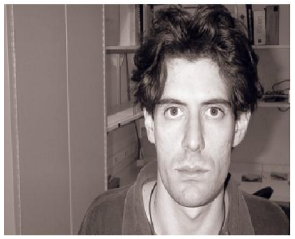
Unary terms **Pairwise terms**

- Message Passing: $\mu_{i\to j}(X_j) = \max_{X_i} \Phi_i(X_i)\Psi_{i,j}(X_i, X_j) \prod_{k\in\mathcal{N}(i)\setminus j} \mu_{k\to j}(X_i)$

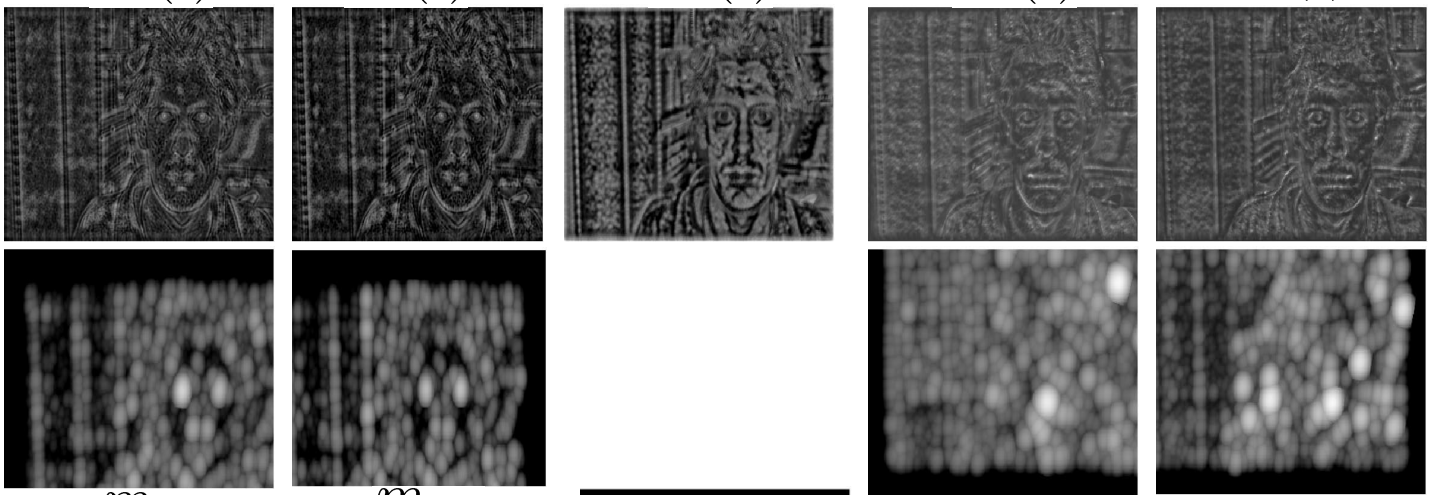
$$\mu_{i\to j}(X_j) = \max_{X_i} \Phi_i(X_i)\Psi_{i,j}(X_i,X_j) \prod_{k\in\mathcal{N}(i)\setminus j} \mu_{k\to j}(X_i)$$

$$B_j(X_j) = \max P(X_j)$$

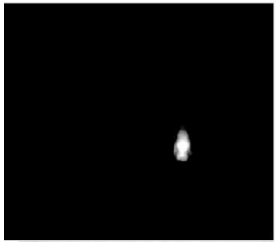




$\Phi_1(l)$ $\Phi_2(l)$ $\Phi_3(l)$ $\Phi_4(l)$ $\Phi_5(l)$

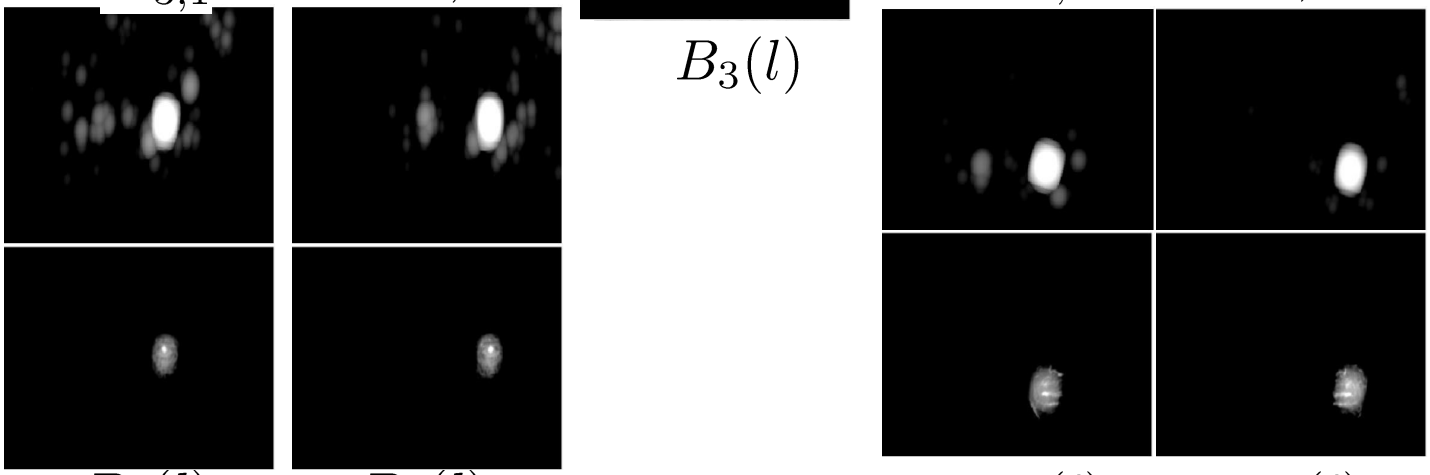


$m_{1,3}$ $m_{2,3}$ $m_{4,3}$ $m_{5,3}$



$B_3(l)$

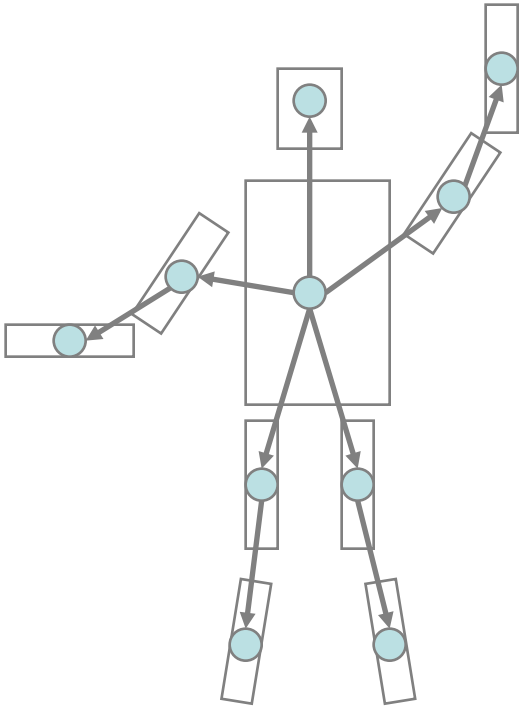
$m_{3,1}$ $m_{3,2}$ $m_{3,4}$ $m_{3,5}$



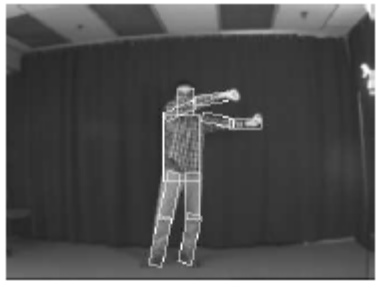
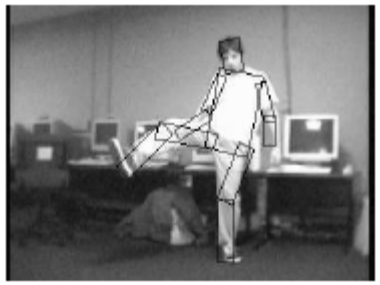
$B_1(l)$ $B_2(l)$ $B_4(l)$ $B_5(l)$

Detection Results



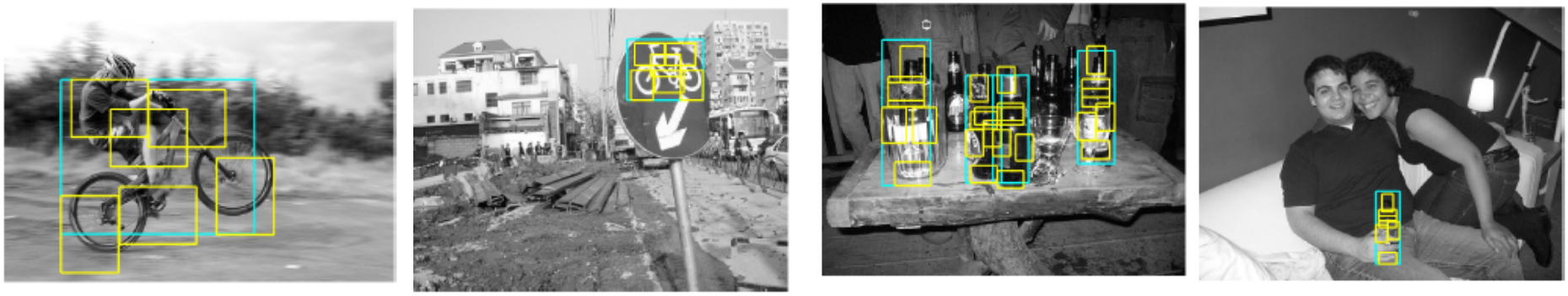


Part-based models + discriminative framework

P. Felzenszwalb, D. McAllester, D. Ramanan`A Discriminatively Trained, Multiscale, Deformable Part Model' CVPR 2008

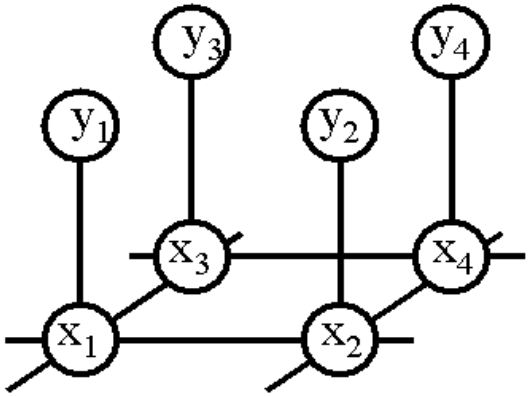
- Linear classifier gives individual part cost
- Gaussian-like cost for spatial offsets
- Learn cost for parts and deformations
  - Discriminative framework



- State-of-the-art, 2008

Lecture outline

- Introduction
- Chain-structured graphical models
- Tree-structured graphical models
  - Pictorial Structures
  - Max/Sum-Product algorithm
- Loopy Graphical Models
  - Markov Random Fields
  - Graph Cuts
  - Gaussian Random Fields



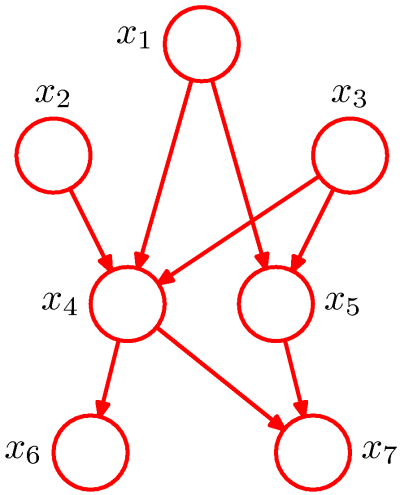
Bayesian Networks

Directed Acyclic Graph $G(V, E)$

Nodes - V : random variables

Edges - E : dependencies

Leave from `parents', arrive at children

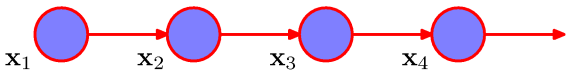


Distribution of each child conditioned on parent: $P(X_v|X_{\pi_v})$

$$P(X_v|\emptyset) = P(X_v)$$

Joint probability distribution: $P(X) = \prod_v P(X_v|X_{\pi_v})$

Bayesian network for Brownian motion:

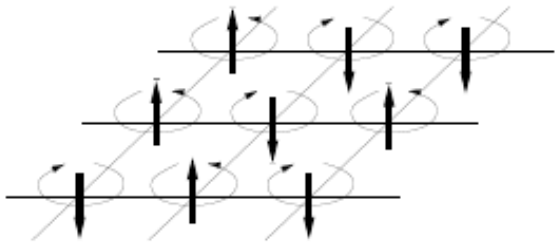


Connection with statistical physics

Idea: image pixels are elements of a large physical system

Ising model: iron atoms on a lattice

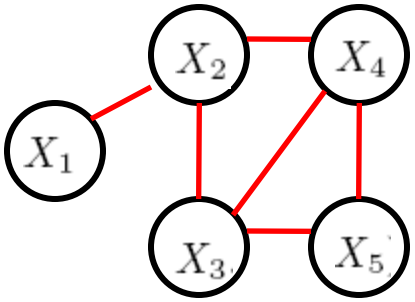
- **Model spin orientations**



- **Interactions an ents**
  - **Nearby spins tend to be similar**
- **Interactions among pixel and environment**
  - **External magnetization**

Markov Random Fields

- Undirected Graph $G(V, E)$



- Maximal graph cliques (fully-connected subsets)

$$\mathcal{C} = \{(X_1, X_2), (X_2, X_3), (X_3, X_4, X_5)\}$$

- Clique potentials

$$\psi_c(x_c) > 0, \forall x_c$$

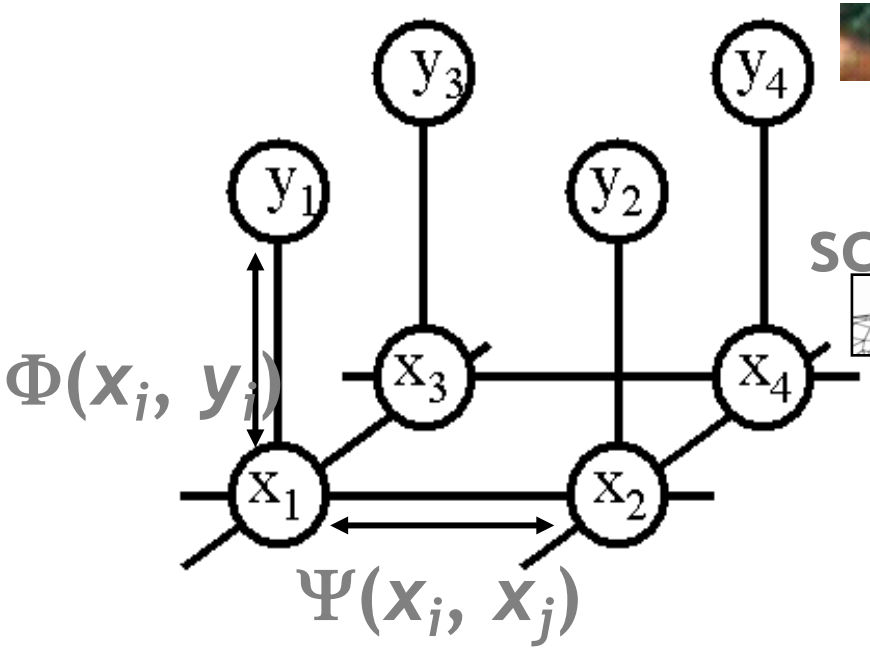
- Joint PDF

$$P(X) = \frac{1}{Z} \prod_{c \in \mathcal{C}} \psi_c(X_c)$$

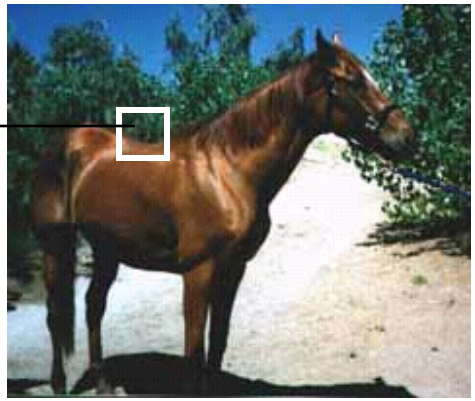
$$Z = \sum_X \prod_{c \in \mathcal{C}} \psi_c(X_c)$$

MRFs for Vision

$$P(X,Y) = \frac{1}{Z}\prod_i \Phi(Y_i, X_i) \prod_{i,j\in\mathcal{E}} \Psi(X_i, X_j)$$

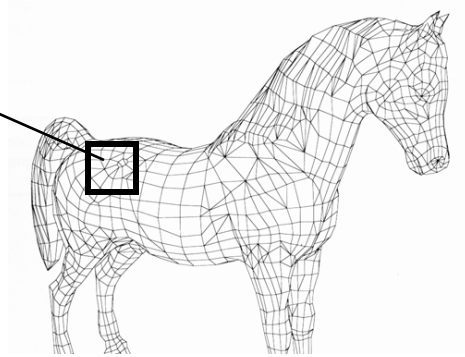


image



image

scene



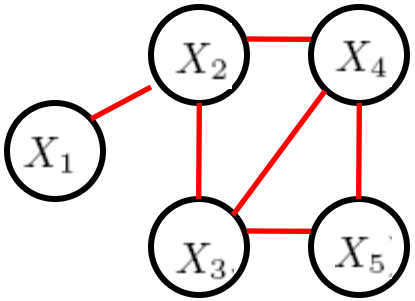
scene

Network Joint Probability

$$P(X,Y) = \frac{1}{Z} \prod_{i} \Phi(Y_i, X_i) \prod_{i,j \in \mathcal{E}} \Psi(X_i, X_j)$$

Markov Random Fields

- Undirected Graph $G(V, E)$



- Maximal graph cliques (fully-connected subsets)

$$\mathcal{C} = \{(X_1, X_2), (X_2, X_3), (X_3, X_4, X_5)\}$$

- Clique potentials $\psi_c(x_c) > 0, \forall x_c$

$$p(x_1, \ldots, x_N) = \frac{1}{Z} \prod_{c \in \mathcal{C}} f_c\Big(\{x_j | j \in c\}\Big) = \frac{1}{Z} \prod_{c \in C} f_c(\mathbf{x}_c)$$

normalization term (partition function)

cliques of the graph

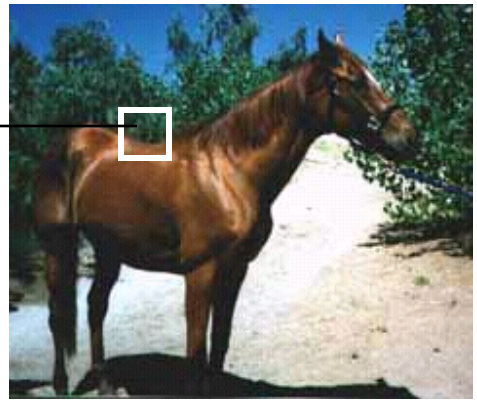
factors or potentials (non-negative, but unnormalized)

variables in clique c

MRFs for Vision

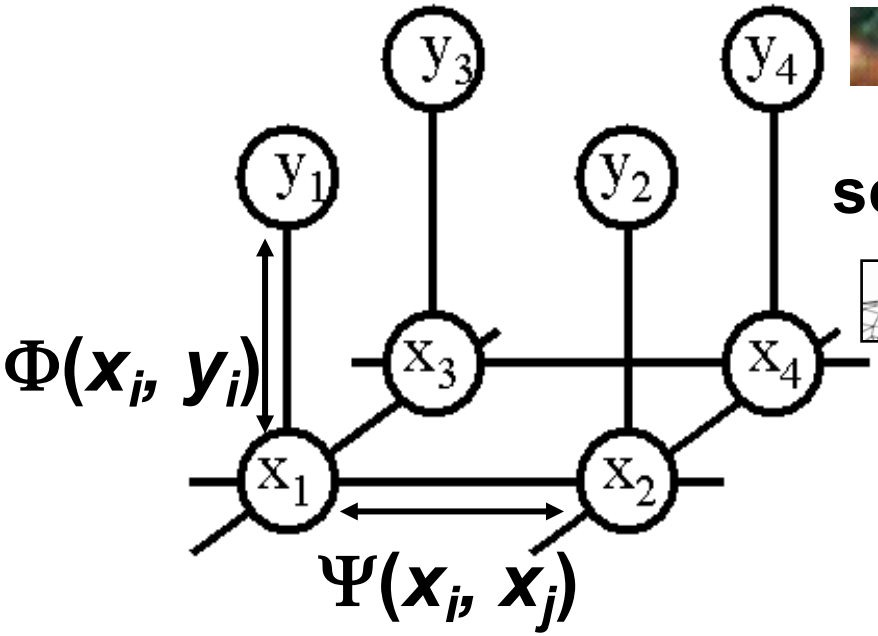
$$P(X,Y) = \frac{1}{Z}\prod_i \Phi(Y_i, X_i) \prod_{(i,j)\in\mathcal{C}} \Psi(X_i, X_j)$$

image patches



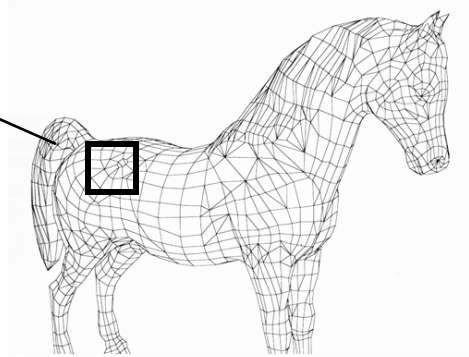
image

scene patches



$\Phi(x_i, y_i)$

$\Psi(x_i, x_j)$



scene

$$P(X|Y) = ?$$

Network Joint Probability

$$P(X,Y) = \frac{1}{Z} \prod_{i} \Phi(Y_i, X_i) \prod_{i,j \in \mathcal{E}} \Psi(X_i, X_j)$$

Scene

Image

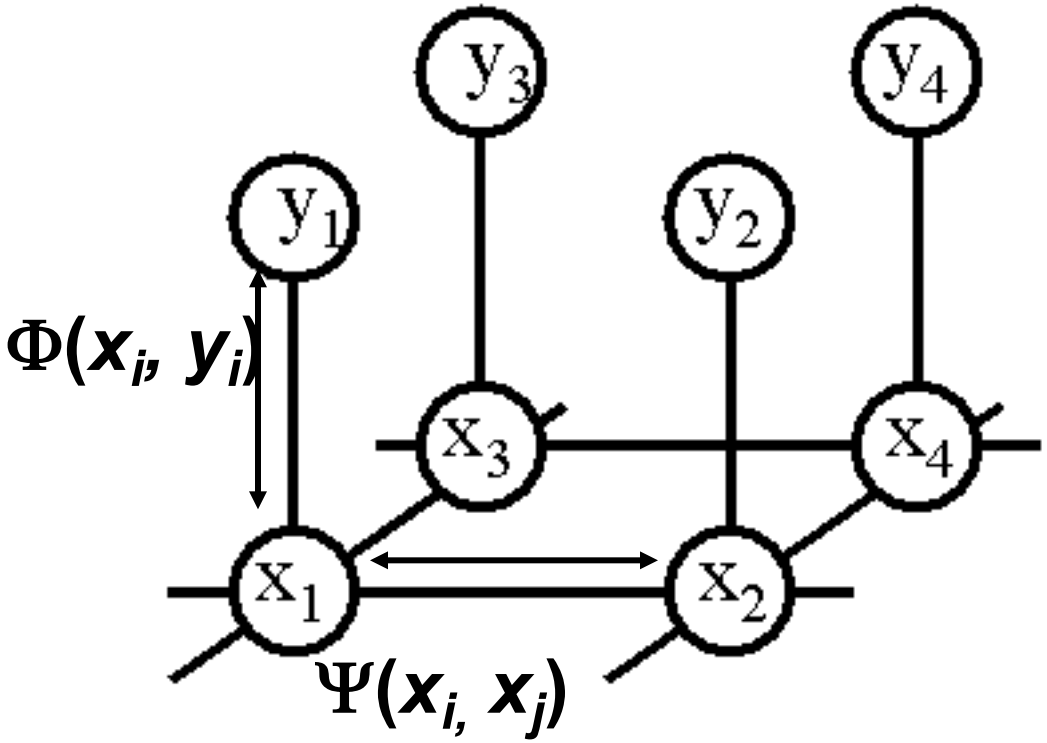
Image-scene compatibility function

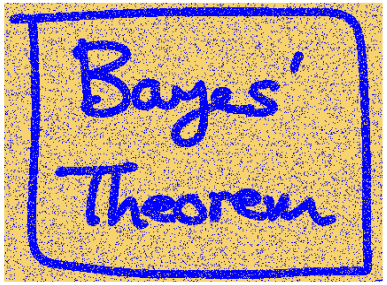
Local observations

Scene-scene compatibility function

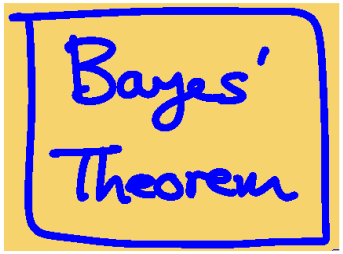
Neighboring scene nodes

MRFs for Denoising



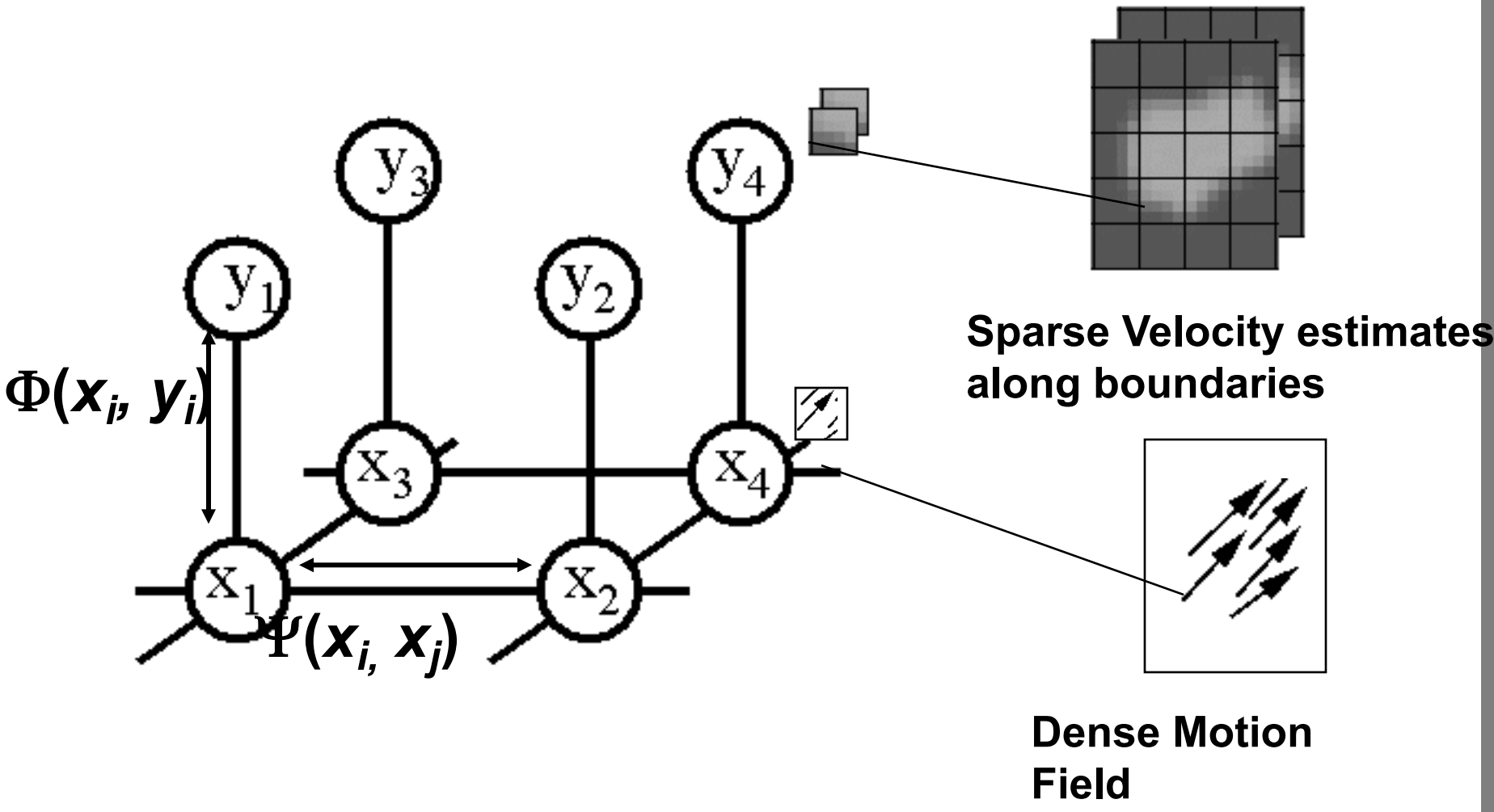


Noisy Pixel Intensities



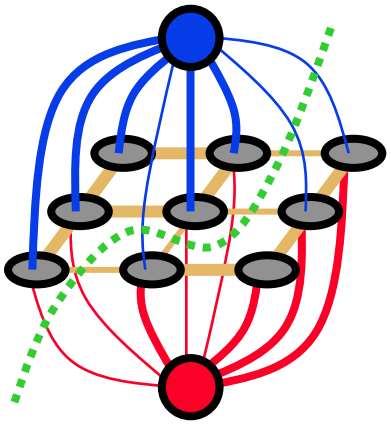
Clean Image

MRFs for Motion

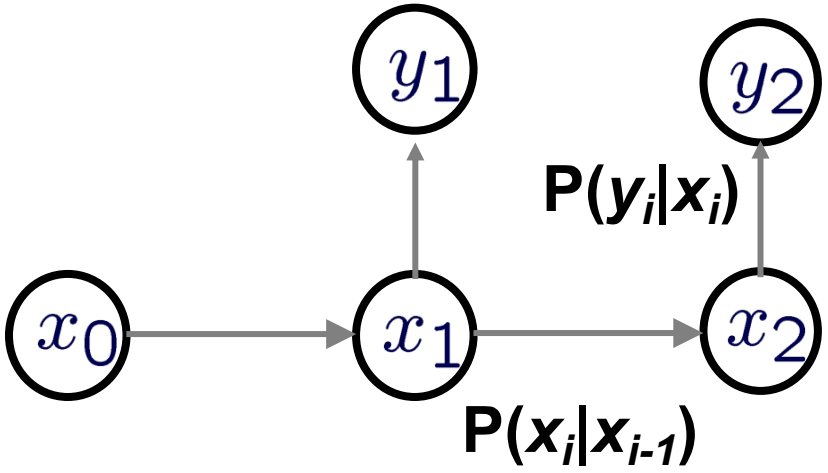


Lecture outline

- Introduction
- Chain-structured graphical models
- Tree-structured graphical models
  - Pictorial Structures
  - Max/Sum-Product algorithm
- Loopy Graphical Models
  - Markov Random Fields
  - Graph Cuts
  - Gaussian Random Fields



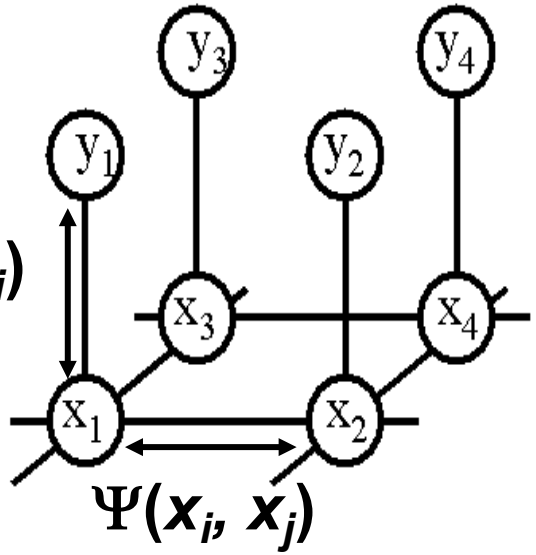
Loops: Exact Computation becomes Hard!



observed process

$\Phi(x_i, y_i)$

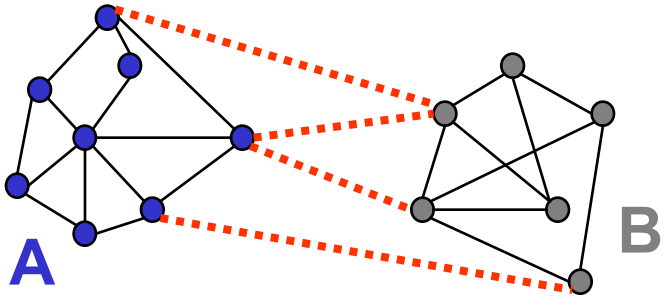
hidden states



Inference: Max/Sum Product (Polynomial)

NP-hard

Graph Cut

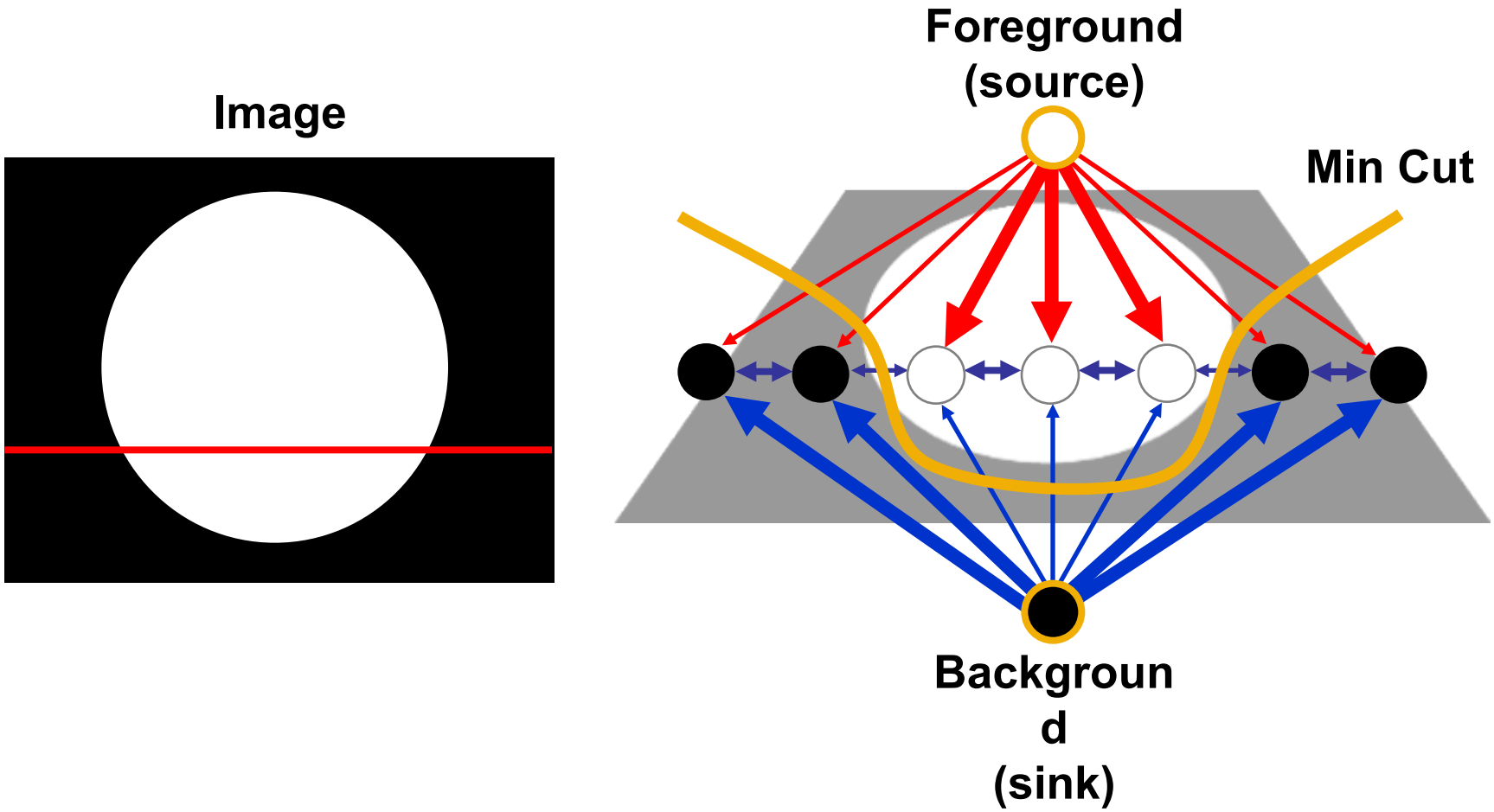


- Set of edges whose removal makes a graph disconnected
- Cost of a cut
  - Sum of weights of cut edges:

$$cut(A,B) = \sum_{p \in A, q \in B} w_{p,q}$$

- A graph cut gives us a segmentation
  - What is a “good” graph cut and how do we find one?

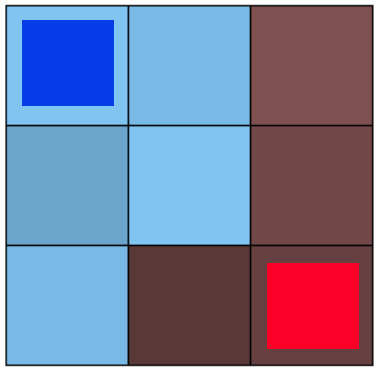
Graph Cuts for Image Segmentation

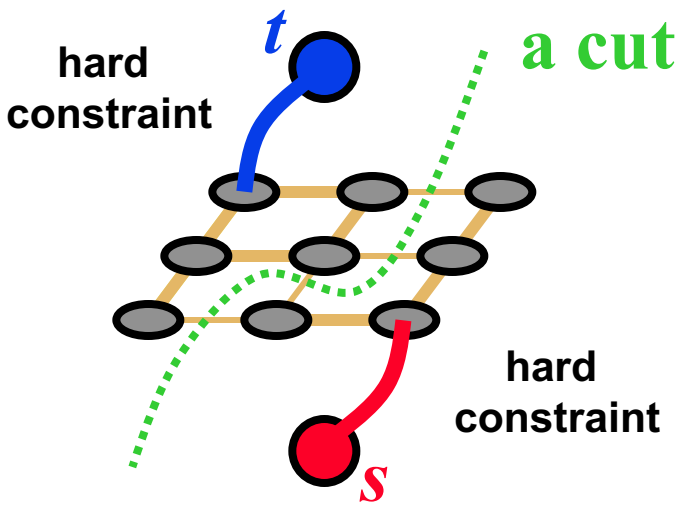


- Segmentation by s-t mincut
  - *Cut:* separating source and sink
  - *Min Cut:* Global minimal energy in polynomial time (1MPixel/sec)

Graph Cuts for Optimal Boundary Detection

- Idea: convert MRF into source-sink graph





Minimum cost cut can be computed in polynomial time

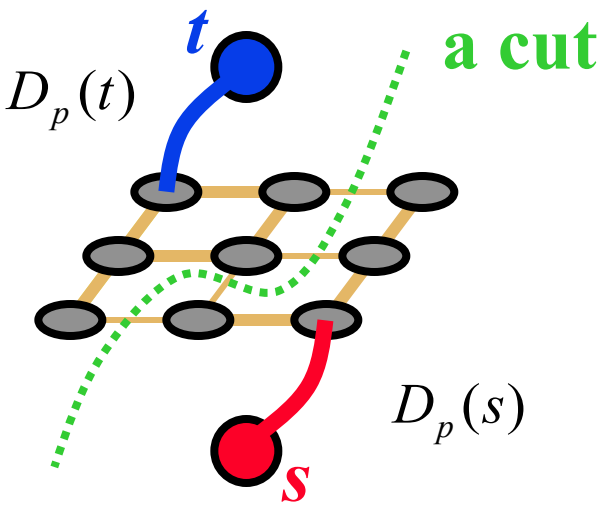
(max-flow/min-cut algorithms)

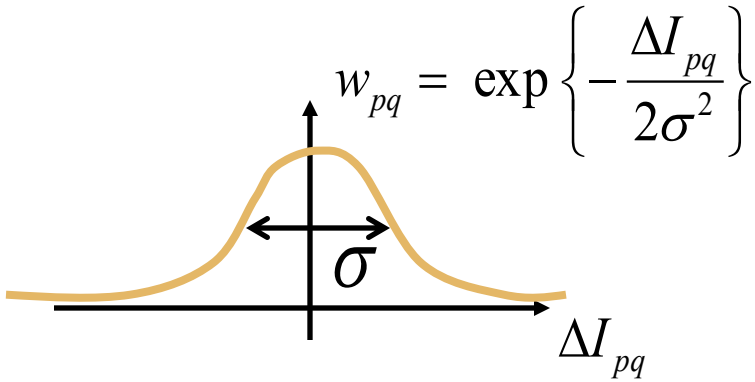
Simple Example of Energy

Regional term **Boundary term**

$$E(L) = \sum_{p} D_p(L_p) + \sum_{pq \in N} w_{pq} \cdot \delta(L_p \neq L_q)$$

t-links **n-links**

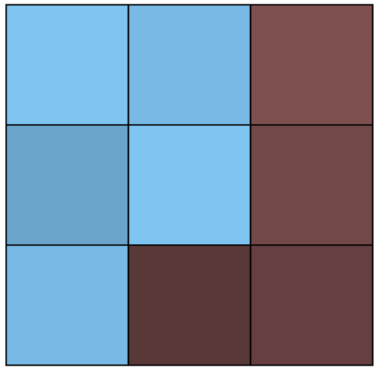


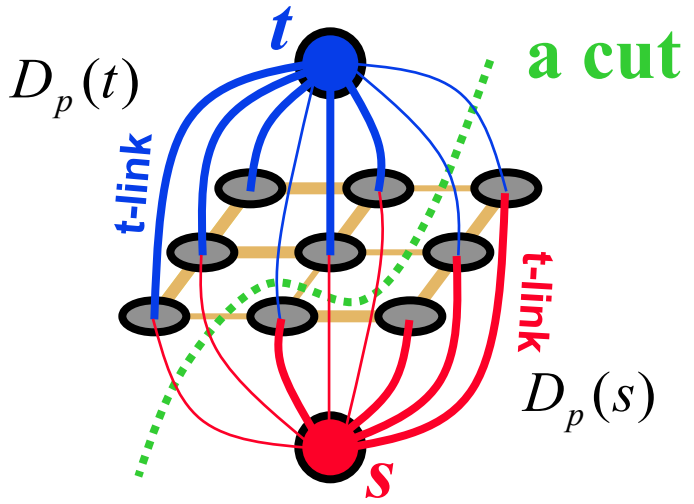


$$L_p \in \{s, t\}$$

(binary object segmentation)

Adding Regional Properties





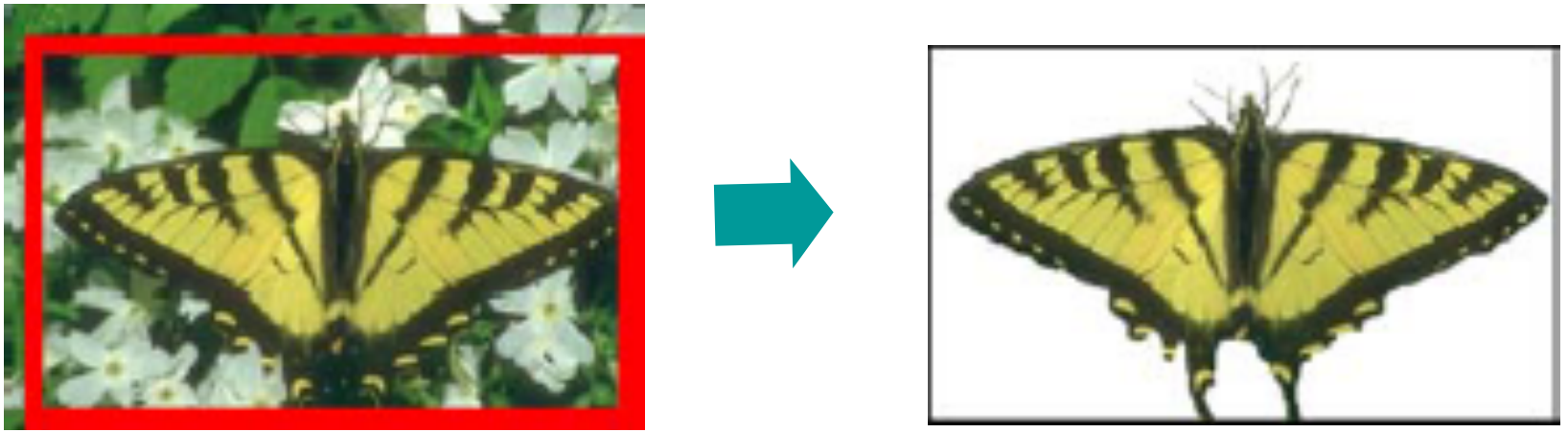
"expected" intensities of
object and background
I^s and I^t
can be re-estimated

$$D_p(s) \propto \exp\left(-\| I_p - I^s \|^2 / 2\sigma^2\right)$$
$$D_p(t) \propto \exp\left(-\| I_p - I^t \|^2 / 2\sigma^2\right)$$

EM-style optimization

Iterative learning of regional color-models

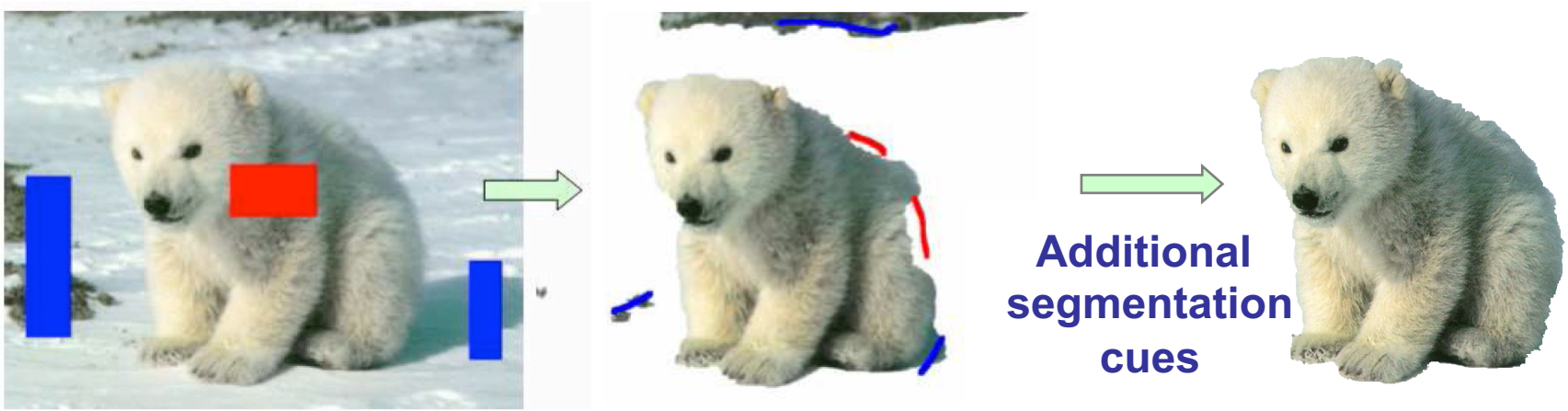
- GMMRF cuts (Blake et al., ECCV04)
  Region competition with Gaussian mixture distributions
- Grab-cut (Rother et al., SIGGRAPH 04)



parametric regional model – Gaussian Mixture (GM)
designed to guarantee convergence

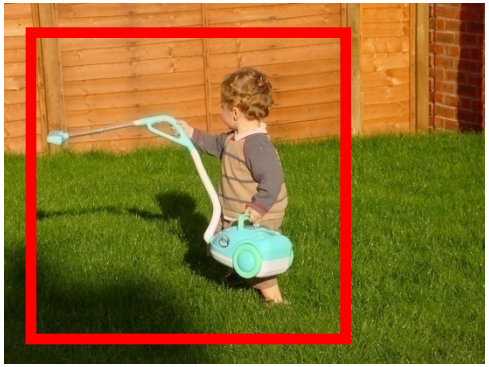
GraphCut Applications: “GrabCut”

- Procedure
  - User marks foreground and background regions with a brush.
  - This is used to create an initial segmentation which can then be corrected by additional brush strokes.



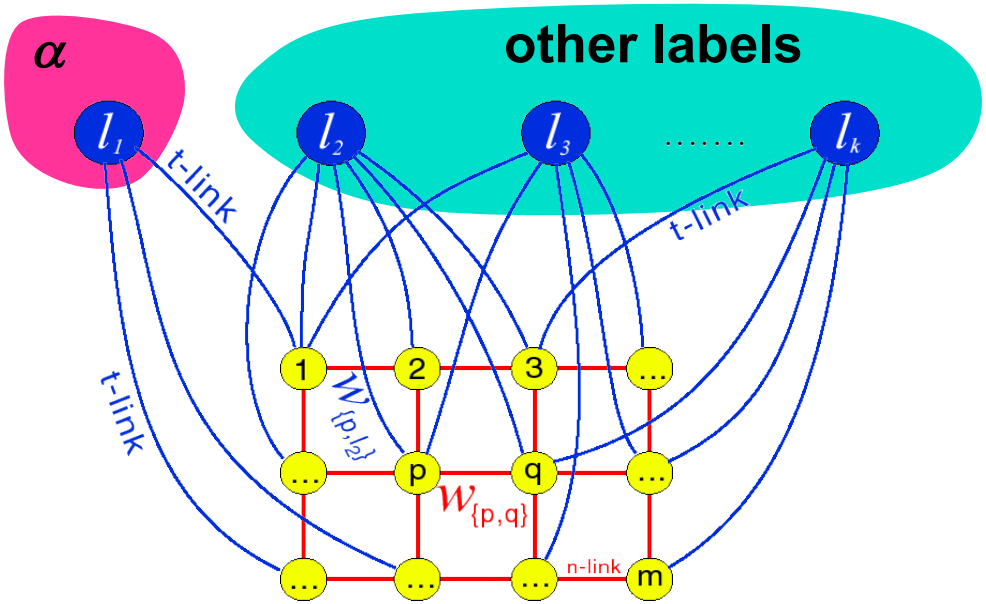
User segmentation cues

GrabCut: Example Results



More labels: α-Expansion

- Basic idea: k vs rest
  - Break multi-way cut computation into a sequence of binary s-t cuts.



Lecture outline

- Introduction
- Chain-structured graphical models
- Tree-structured graphical models
  - Pictorial Structures
  - Max/Sum-Product algorithm
- Loopy Graphical Models
  - Markov Random Fields
  - Graph Cuts
  - Gaussian Random Fields

Markov Random Fields in Vision

$$P(X,Y) = \frac{1}{Z}\prod_i \Phi(Y_i, X_i) \prod_{(i,j)\in\mathcal{C}} \Psi(X_i, X_j)$$

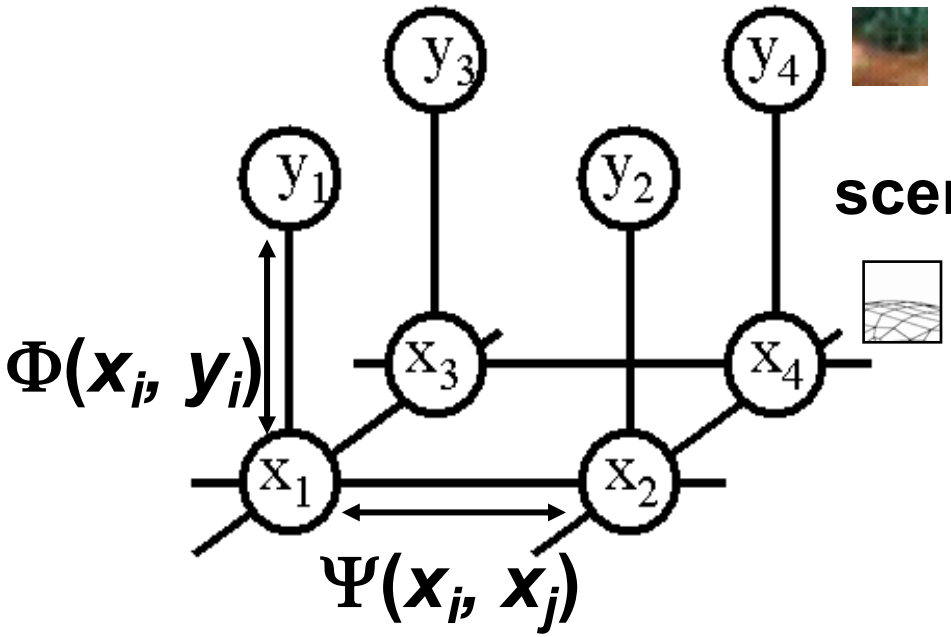
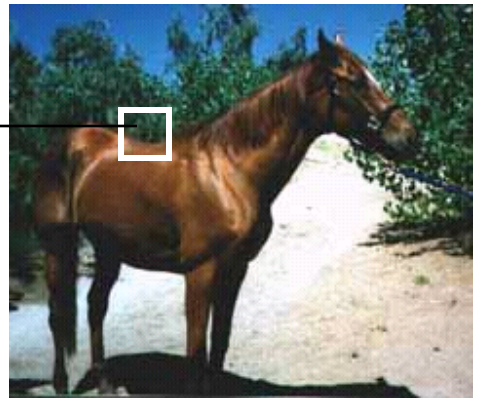
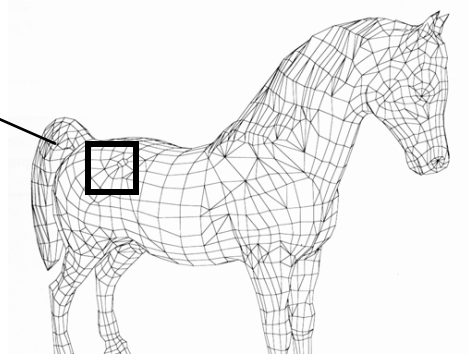


image patches

scene patches



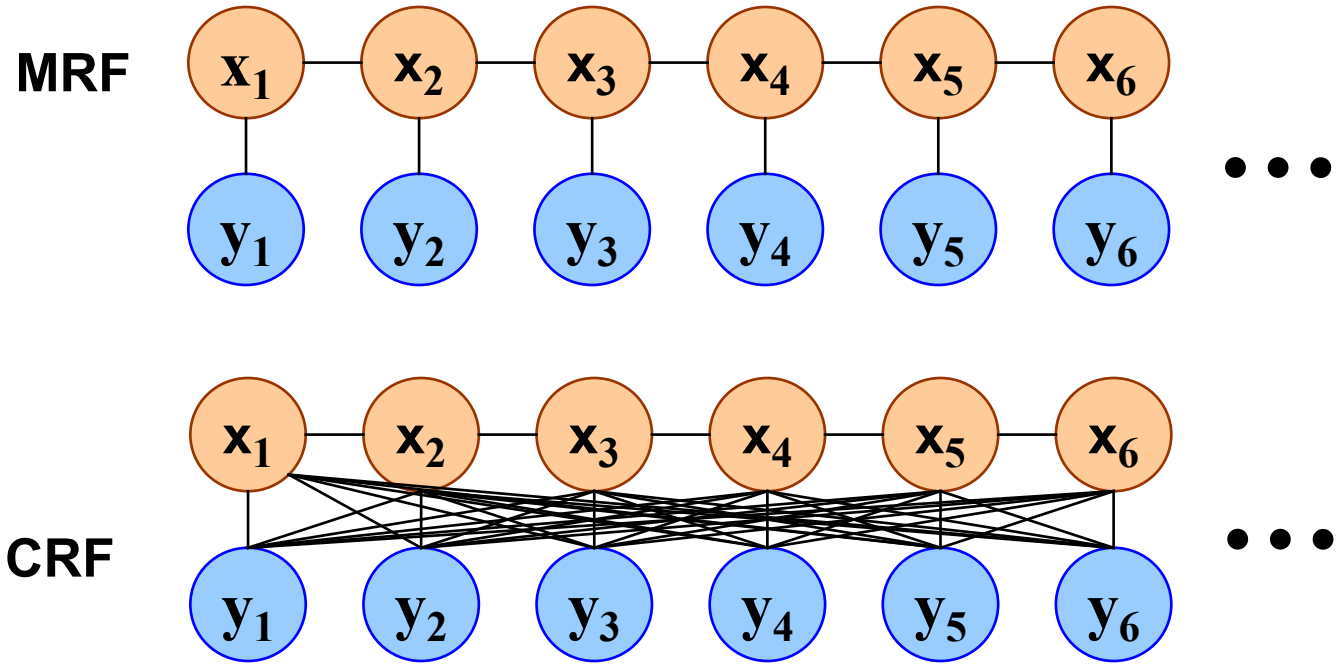
image



scene

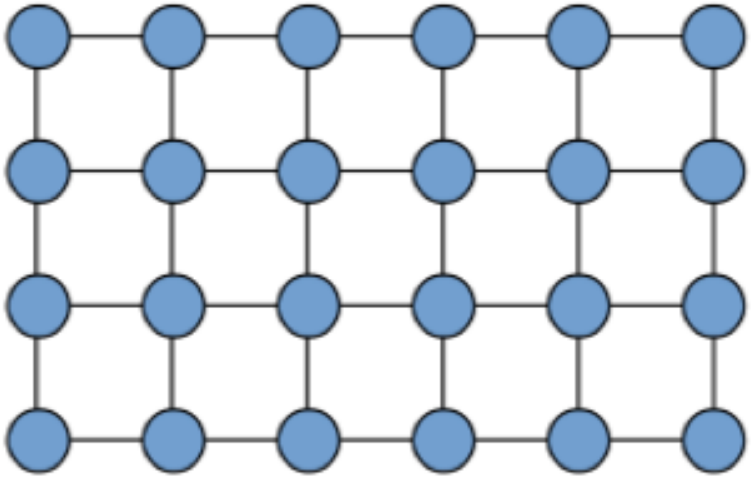
$$P(X|Y) = ?$$

Conditional Random Fields



CRFs: keep MRF tools, drop Bayesian aspect

Conditional Random Field

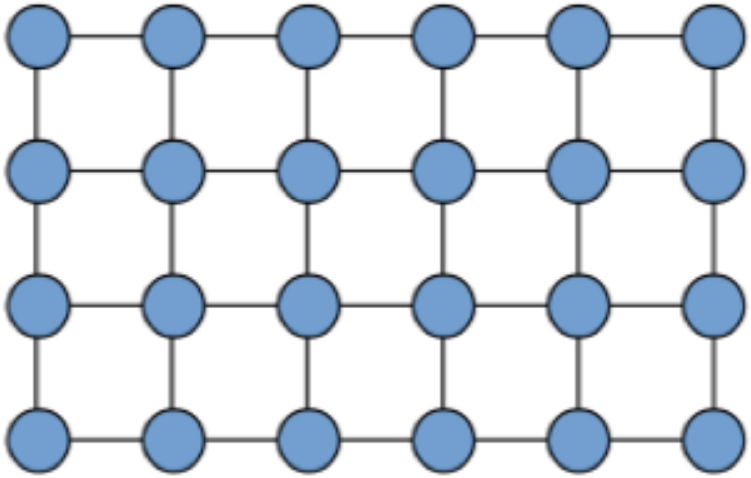


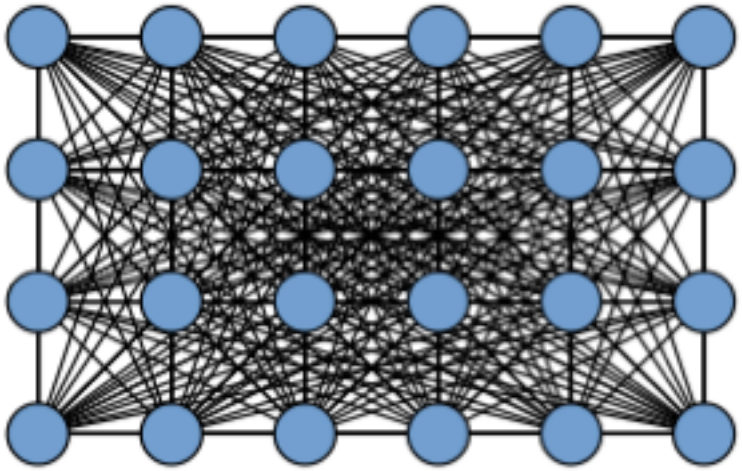
$$P(\mathbf{x}|\mathbf{y}) = \frac{1}{Z_{\mathbf{y}}} \exp\left(-E_{\mathbf{y}}(\mathbf{x})\right)$$

$$E_{\mathbf{y}}(\mathbf{x}) = \sum_{i} U_i(\mathbf{x}_i, \mathbf{y}) + \sum_{i,j \in \mathcal{E}} B_{i,j}(\mathbf{x}_i, \mathbf{x}_j, \mathbf{y})$$

pre-softmax FCNN **Pairwise terms**

Dense Conditional Random Field





$$P(\mathbf{x}|\mathbf{y}) = \frac{1}{Z_{\mathbf{y}}} \exp\left(-E_{\mathbf{y}}(\mathbf{x})\right)$$

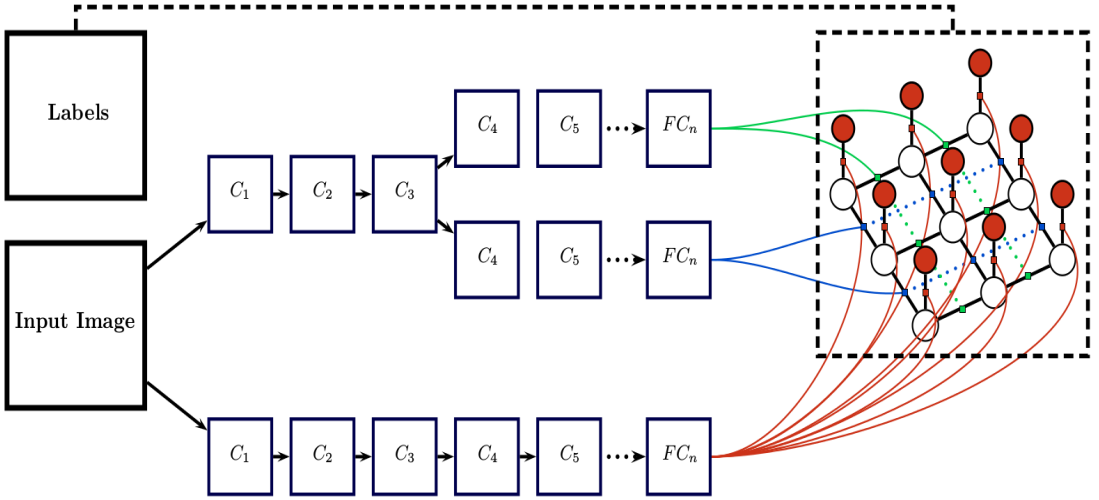
$$E_{\mathbf{y}}(\mathbf{x}) = \sum_i U_i(\mathbf{x}_i, \mathbf{y}) + \sum_{i,j \in \mathcal{E}} B_{i,j}(\mathbf{x}_i, \mathbf{x}_j, \mathbf{y})$$

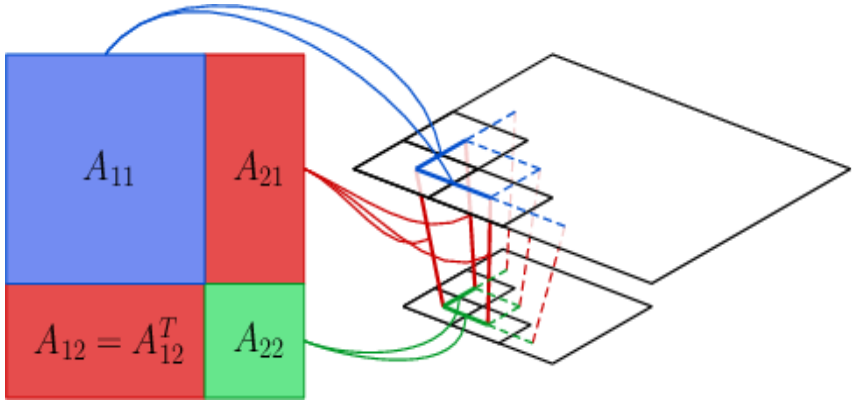
pre-softmax FCNN **Pairwise terms**

Philipp Krähenbühl and Vladlen Koltun, Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials, NIPS 2011

Fast, Exact, and Multi-Scale Inference for FCNN-CRF

S. Chandra

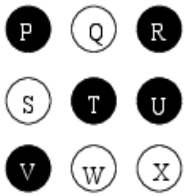




S. Chandra, I. Kokkinos, Fast, Exact and Multi-Scale Inference for Semantic Image Segmentation with Deep Gaussian CRFs, arXiv:1603.08358

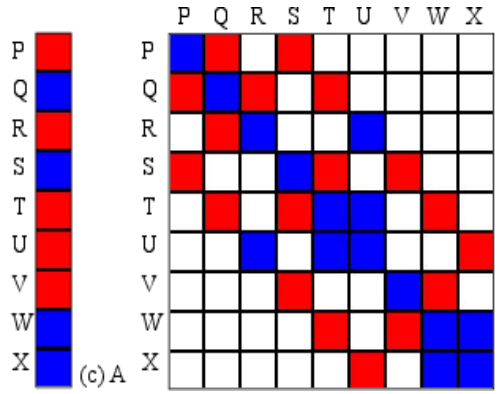
Gaussian Random Fields: Random Fields for dummies

$$\pi(\mathbf{x}) = \frac{1}{Z}\exp\left(-\frac{1}{2}\mathbf{x}^T\Theta\mathbf{x} + \theta^T\mathbf{x}\right)$$



$$\Theta\mathbf{x}^* = \theta$$

$\mathbf{x}^*$



θ Θ

Maximum-A-Posteriori inference =
Minimum Mean-Squared Error inference =
solution of linear system

Gaussian MRF: blurry samples (hard to have outliers)

Gaussian CRF: image-based pairwise terms (e.g. discontinuity -preserving)

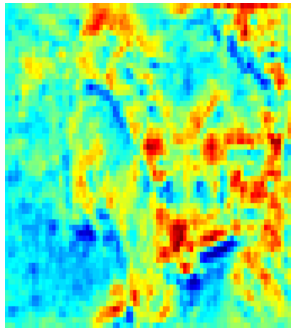
Jancsary, Nowozin, Sharp & Rother, Regression Tree Fields, CVPR12

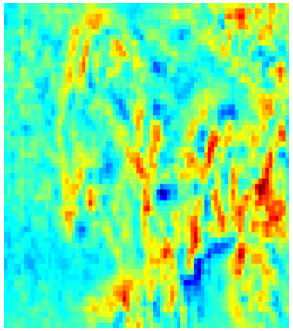
Tappen, Liu, Adelson & Freeman, Learning Gaussian CRFs for low-level vision, CVPR07

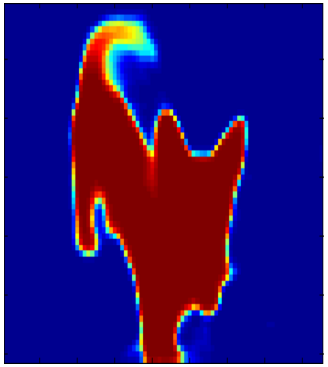
Deep Gaussian Conditional Random Field

Pairwise terms

Posterior



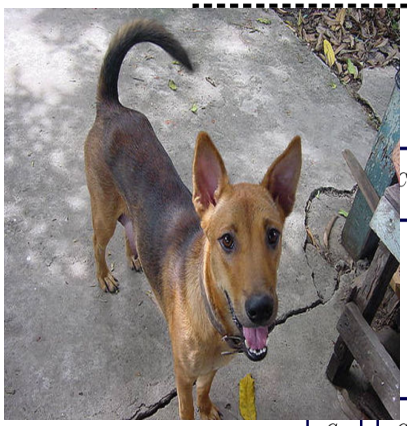


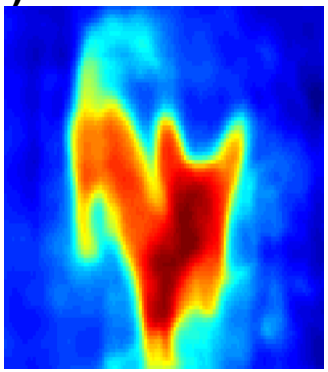


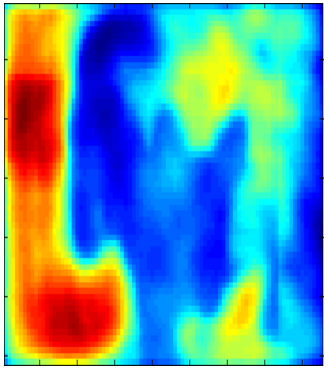
Dog-Background (Vertical)

Dog-Background (Horizontal)

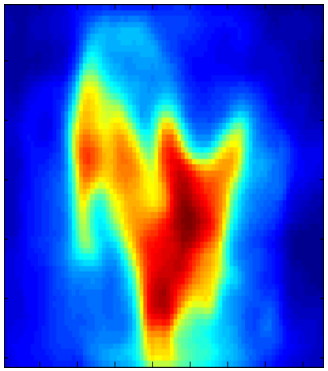
Outputs

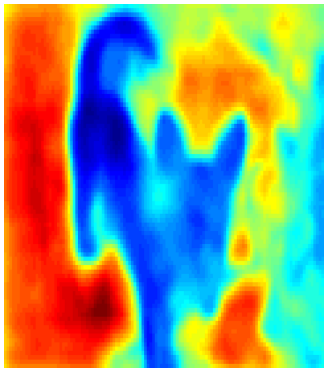




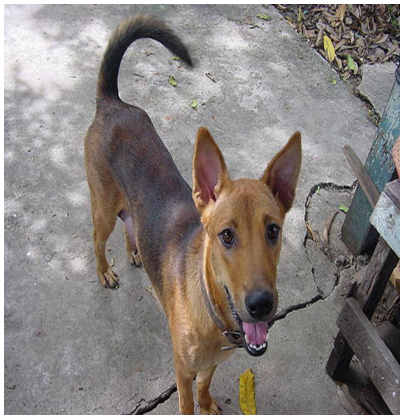


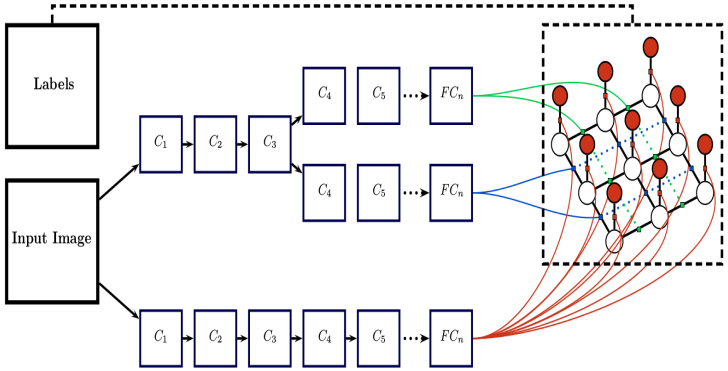
Unary terms

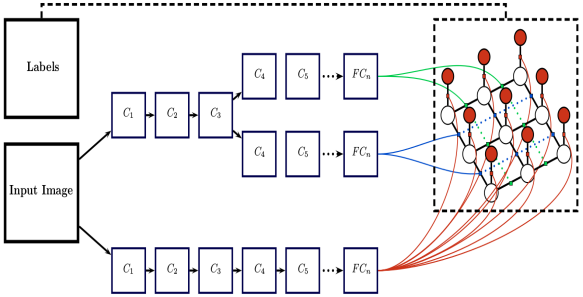


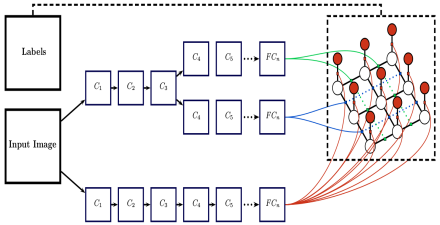


Naïve Multi-Resolution Semantic Segmentation







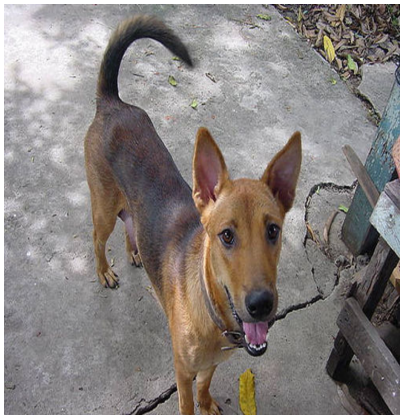


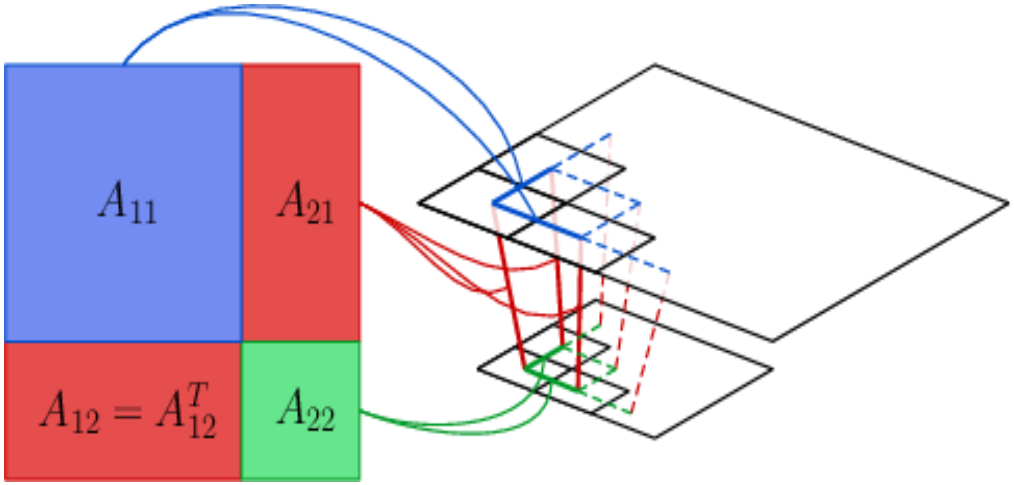
Fuse

L.-C. Chen, Y. Yang, J. Wang, W. Xu and A. Yuille, ‘Attention to Scale: Scale-aware Semantic Image Segmentation, CVPR 2016

I. Kokkinos, Pushing the Boundaries of Boundary Detection using Deep Learning, ICLR 2016

Linear systems & Multi-resolution CRFs

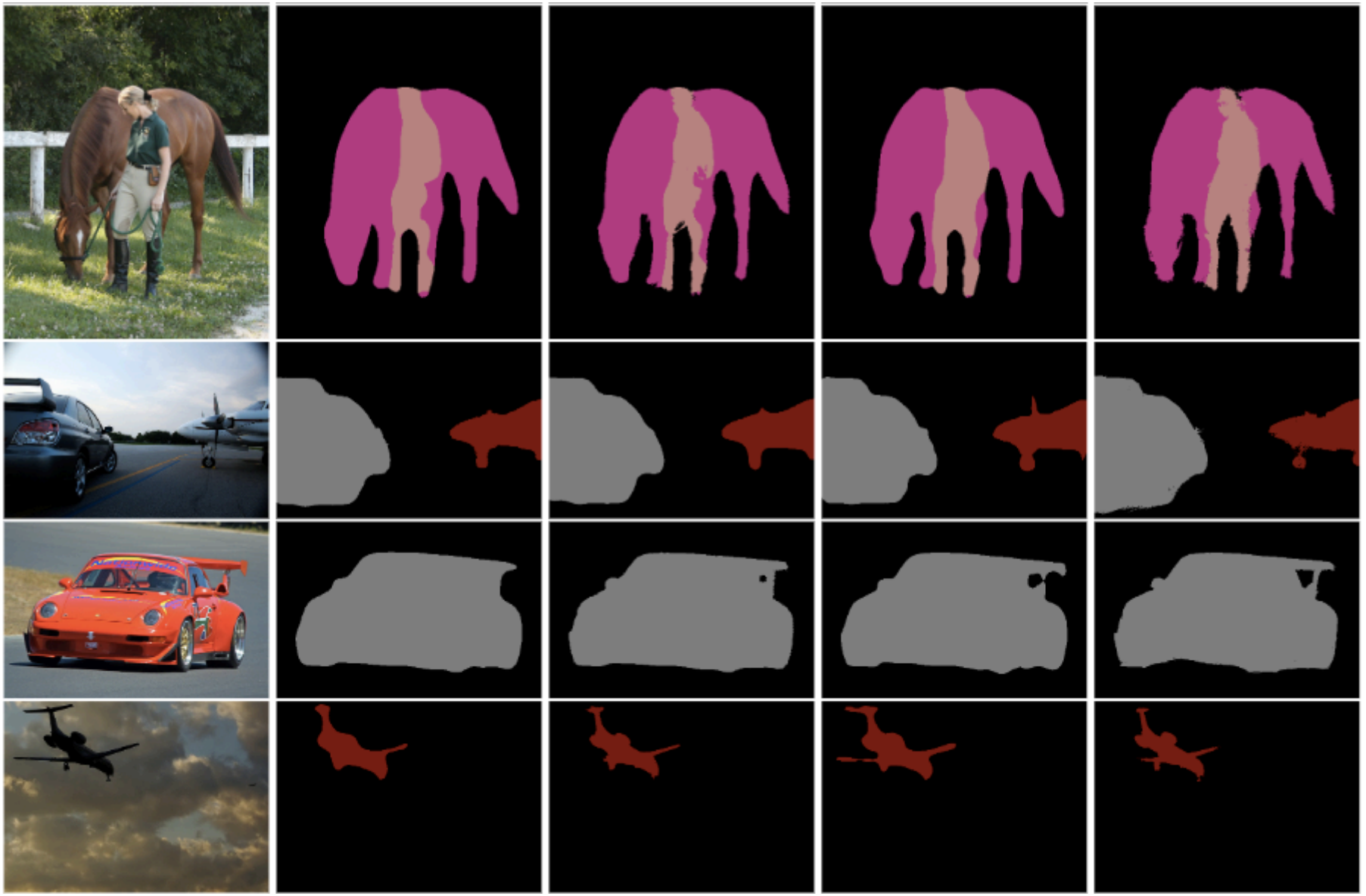




Learn to enforce coupling of different results

Consistently better results than decoupled learning!

Improvements/Complementarity with DenseCRF



FCNN +DCRF +GCRF +DCRF +GCRF

Quantitative Results

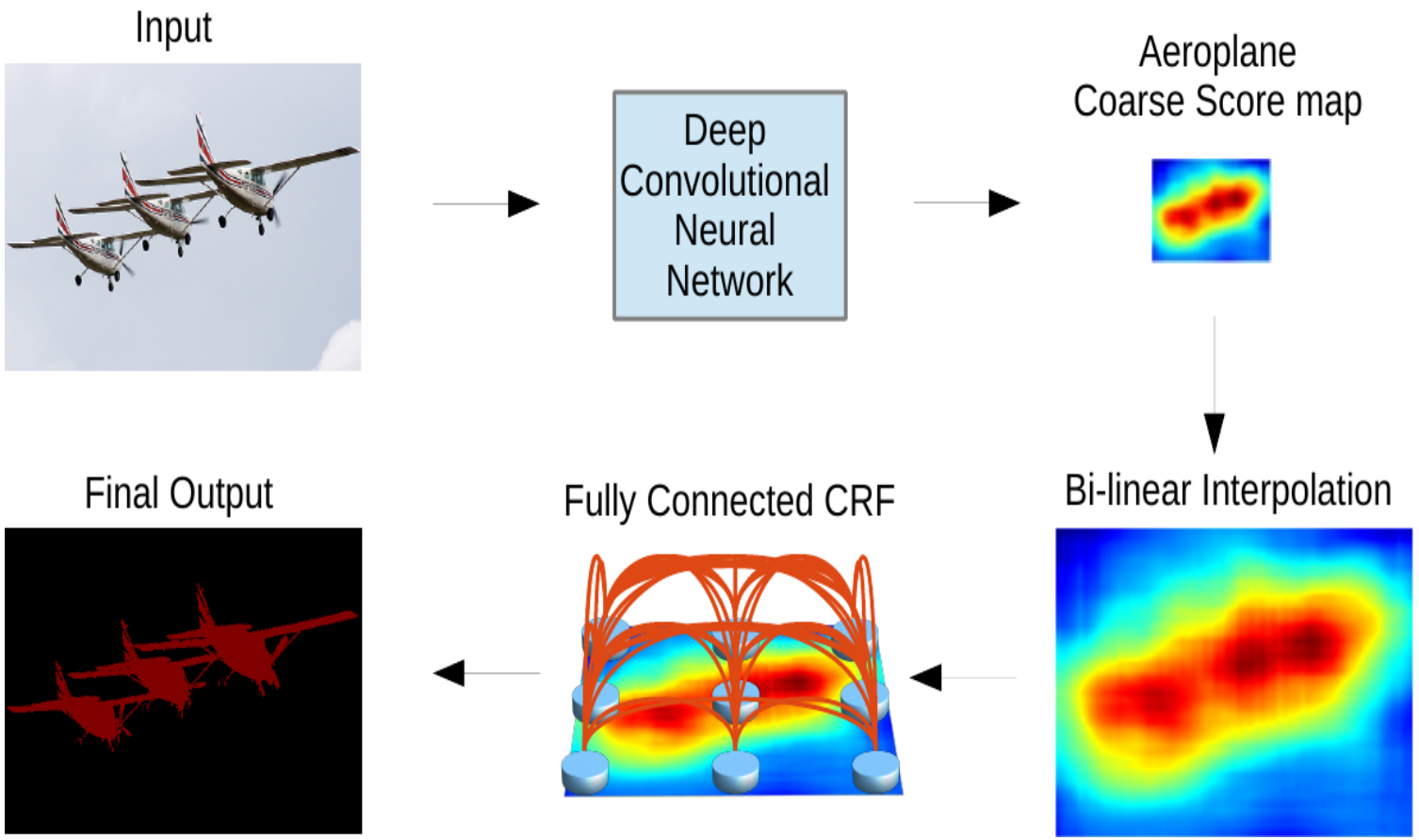
Method	IoU	IoU after *dense CRF*
Basenet	72.72	73.78
QO_4	73.41	75.13
QO_4^{mres}	73.86	75.46

Pascal Leaderboard:

http://host.robots.ox.ac.uk:8080/leaderboard/displaylb.php?challengeid=11&compid=6

	mean	aero plane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow	dining table	dog	horse	motor bike	person	potted plant	sheep	sofa	train	tv/ monitor	submission date
RRR-ResNet152-COCO-MultiScale [?]	81.9	94.4	67.0	92.8	72.4	78.3	93.9	89.6	92.3	40.2	88.1	76.1	86.2	91.6	87.7	87.3	69.3	88.7	66.1	87.1	75.7	13-Sep-2016
CentraleSupelec Deep G-CRF [?]	80.2	92.9	61.2	91.0	66.3	77.7	95.3	88.9	92.4	33.8	88.4	69.1	89.8	92.9	87.7	87.5	62.6	89.9	59.2	87.1	74.2	12-Aug-2016
DeepLabv2-CRF [?]	79.7	92.6	60.4	91.6	63.4	76.3	95.0	88.4	92.6	32.7	88.5	67.6	89.6	92.1	87.0	87.4	63.3	88.3	60.0	86.8	74.5	06-Jun-2016
LRR_4x_ResNet_COCO [?]	79.3	92.4	45.1	94.6	65.2	75.8	95.1	89.1	92.3	39.0	85.7	70.4	88.6	89.4	88.6	86.6	65.8	86.2	57.4	85.7	77.3	18-Jul-2016
Adelaide_VeryDeep_FCN_VOC [?]	79.1	91.9	48.1	93.4	69.3	75.5	94.2	87.5	92.8	36.7	86.9	65.2	89.1	90.2	86.5	87.2	64.6	90.1	59.7	85.5	72.7	13-May-2016
LRR_4x_COCO [?]	78.7	93.2	44.2	89.4	65.4	74.9	93.9	87.0	92.0	42.9	83.7	68.9	86.5	88.0	89.0	87.2	67.3	85.6	64.0	84.1	71.5	16-Jun-2016

FCNN-DenseCRF: Accurate & Sharp



P. Krähenbühl and V. Koltun, Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials, NIPS 2011

L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy and A. Yuille, Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs, ICLR 2015

Dense CRF: smart choice of pairwise term

$$\psi_{i,j}(l, l') = \mu(l, l') \sum_{m=1}^{M} w_m k_m(\mathbf{f}_i, \mathbf{f}_j)$$

$$= [l \neq l'] \left[w_1 \exp\left(-\frac{\|p_i - p_j\|^2}{2\sigma_a^2} - \frac{\|I_i - I_j\|^2}{2\sigma_b^2} \right) + w_2 \exp\left(-\frac{\|p_i - p_j\|^2}{2\sigma_\gamma^2} \right) \right]$$

Potts model

'Bilateral kernel'

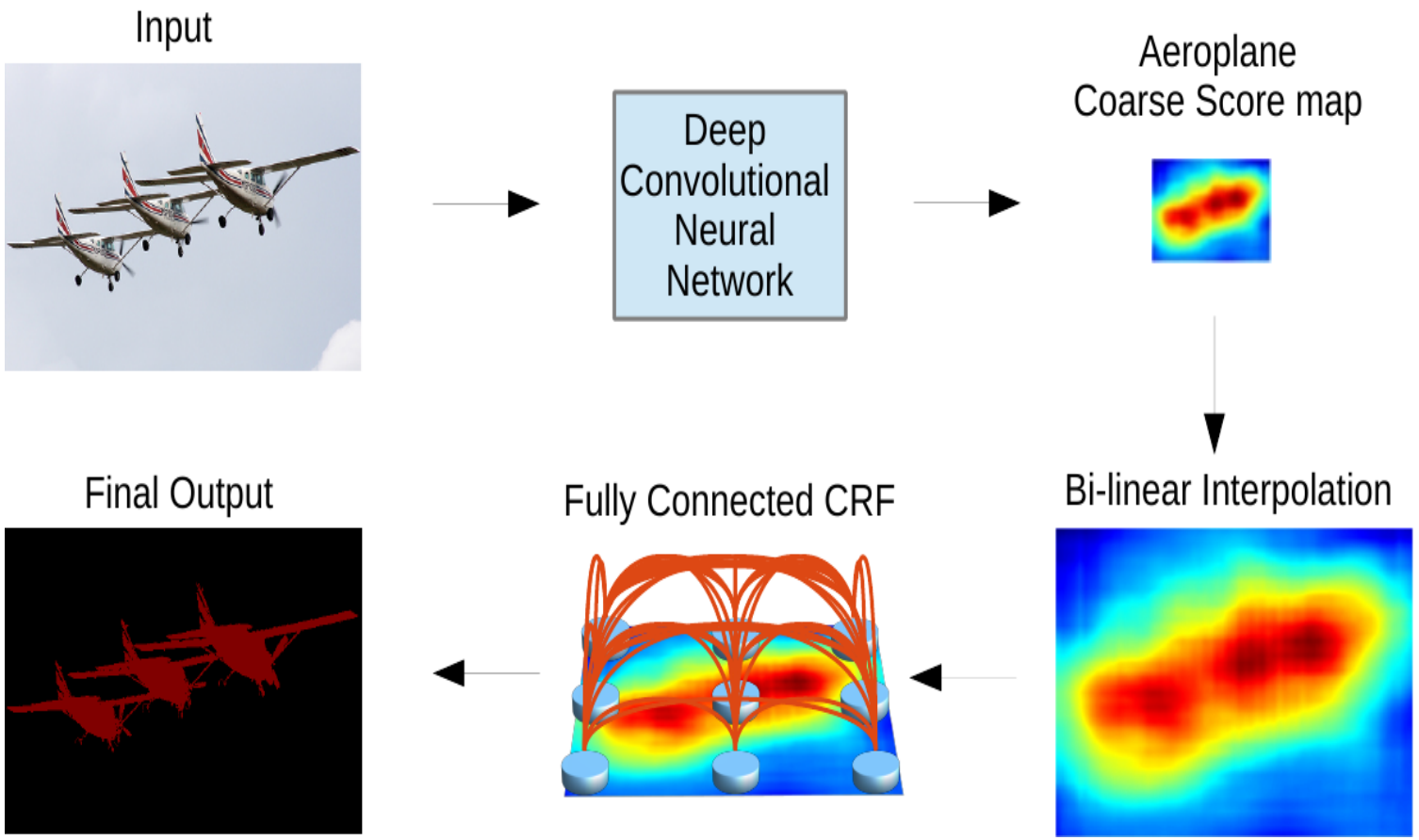
Spatial proximity

Mean Field Updates:

$$Q_i(l) = \frac{1}{Z_i} \exp\left\{ -\psi_i(l) - \sum_{l'} \mu(l, l') \sum_{m=1}^{M} w_m \sum_{j \in \mathcal{N}_i} k_m(\mathbf{f}_i, \mathbf{f}_j) Q_j(l') \right\}$$

Efficient high-dimensional convolutions using the Permutohedral Lattice

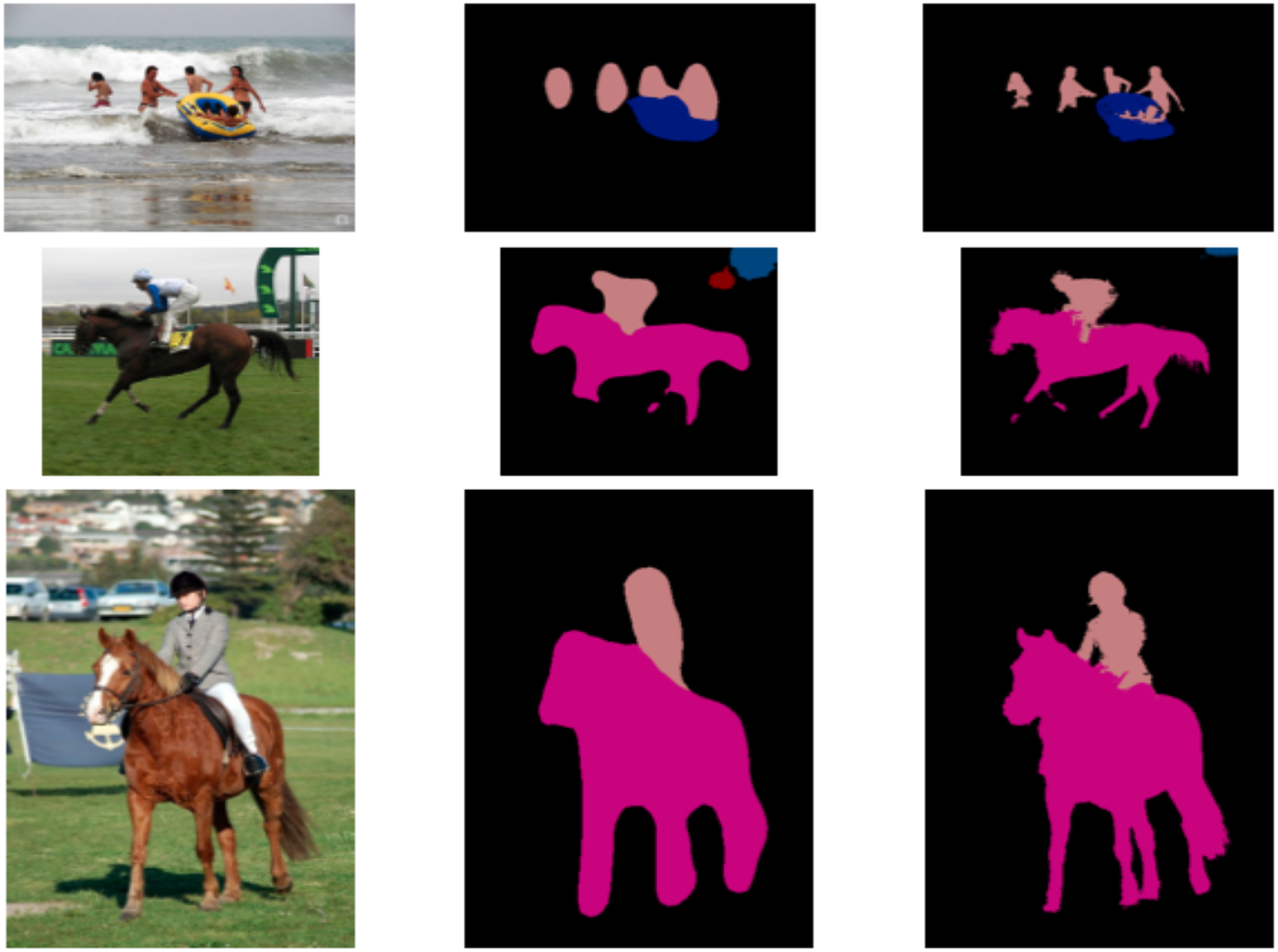
FCNN-DenseCRF: Accurate & Sharp



P. Krähenbühl and V. Koltun, Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials, NIPS 2011

L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy and A. Yuille, Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs, ICLR 2015

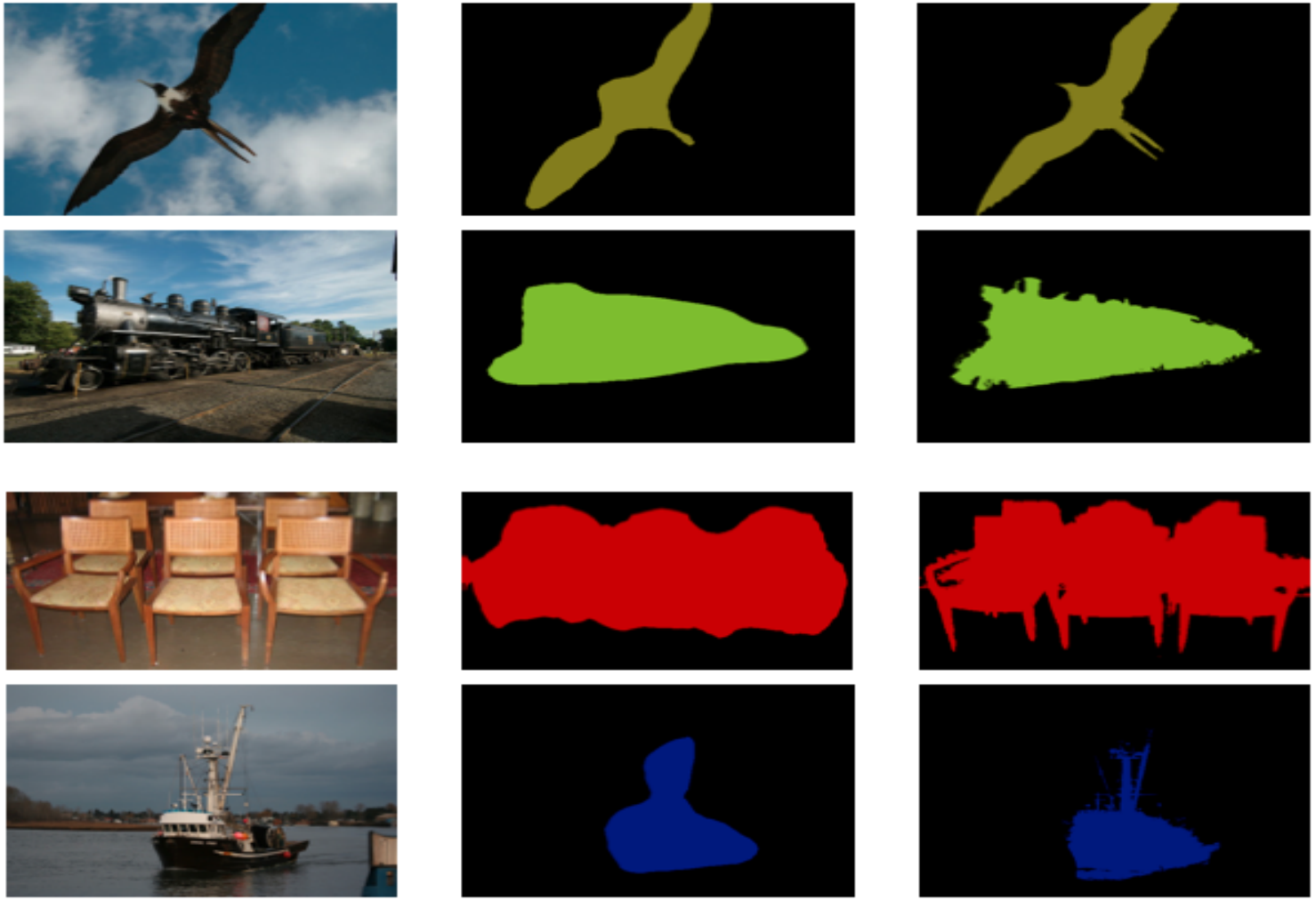
Qualitative Results



FCNN

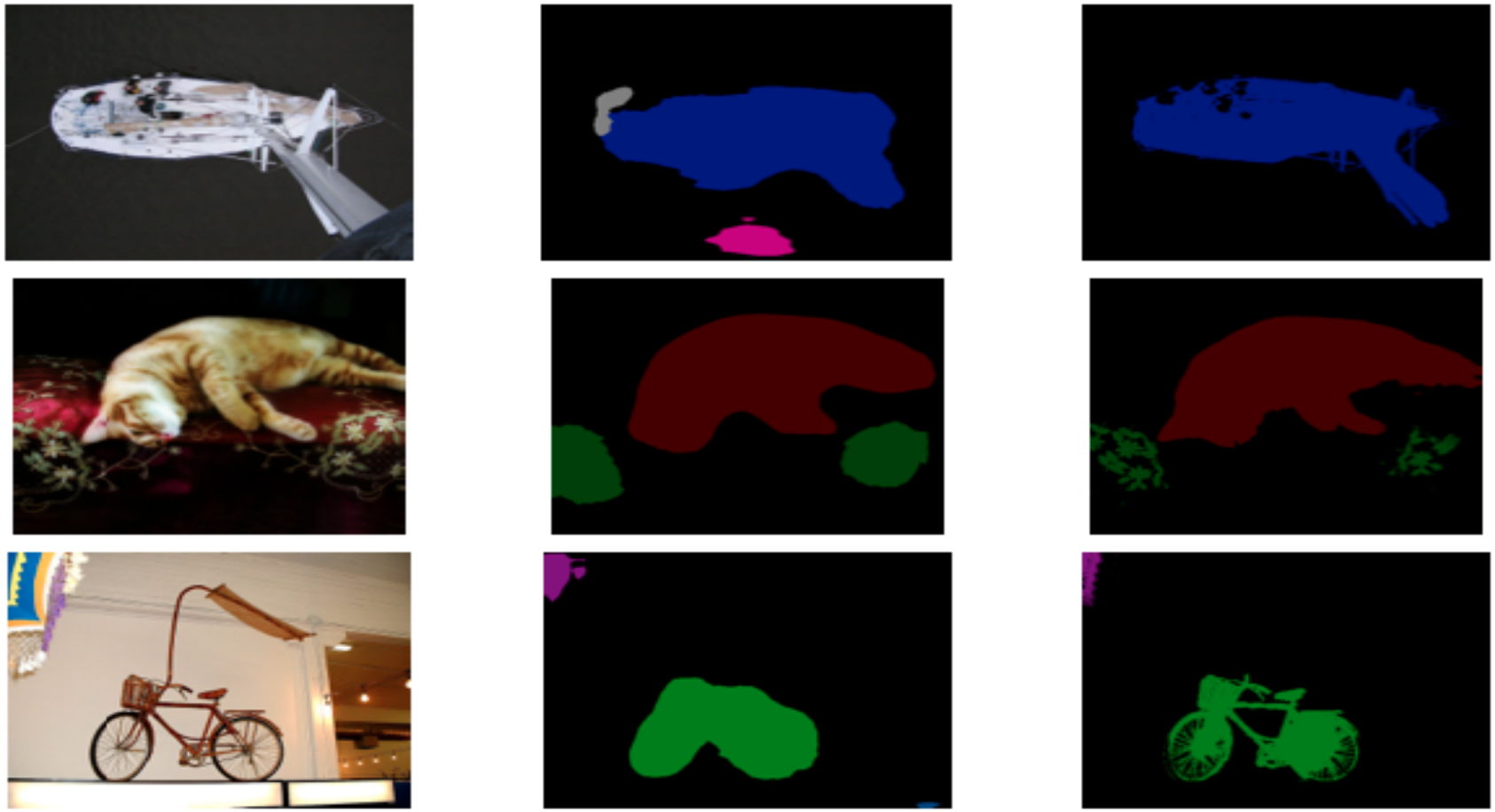
FCNN-DCRF

Qualtiative Results



FCNN FCNN-DCRF

Qualitative Results



FCNN

FCNN-DCRF

Indicative Results

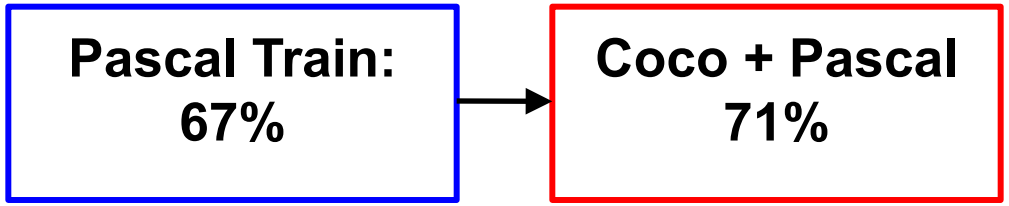
FCNN

FCNN-DCRF

Comparison to state-of-the-art, 2014

Method	mean IOU (%)
MSRA-CFM	61.8
FCN-8s	62.2
TTI-Zoomout-16	64.4
DeepLab-CRF (our)	66.4
DeepLab-MSc-CRF (our)	67.1

G. Papandreou, et al, Weakly- and Semi-Supervised Learning of a DCNN for Semantic Image Segmentation, arxiv 2015



74.7 end-to-end

S. Zheng, et al. CRFs as recurrent neural networks. In ICCV, 2015.

Updated results (Deeplab v2)

Method	mIOU
DeepLab-CRF-LargeFOV-COCO [58]	72.7
MERL_DEEP_GCRF [88]	73.2
CRF-RNN [59]	74.7
POSTECH_DeconvNet_CRF_VOC [61]	74.8
BoxSup [60]	75.2
Context + CRF-RNN [74]	75.3
QO_4^{mres} [66]	75.5
DeepLab-CRF-Attention [17]	75.7
CentraleSuperBoundaries++ [18]	76.0
DeepLab-CRF-Attention-DT [63]	76.3
H-ReNet + DenseCRF [89]	76.8
LRR_4x_COCO [90]	76.8
DPN [62]	77.5
Adelaide_Context [40]	77.8
Oxford_TVG_HO_CRF [87]	77.9
Context CRF + Guidance CRF [91]	78.1
Adelaide_VeryDeep_FCN_VOC [92]	79.1
DeepLab-CRF (ResNet-101)	79.3
Ensemble DeepLab-CRF (ResNet-101)	79.6

TABLE 5: Performance on PASCAL VOC 2012 *test* set. We have added some results from recent arXiv papers on top of the official leadearboard results.

Updated results (Deeplab v2)

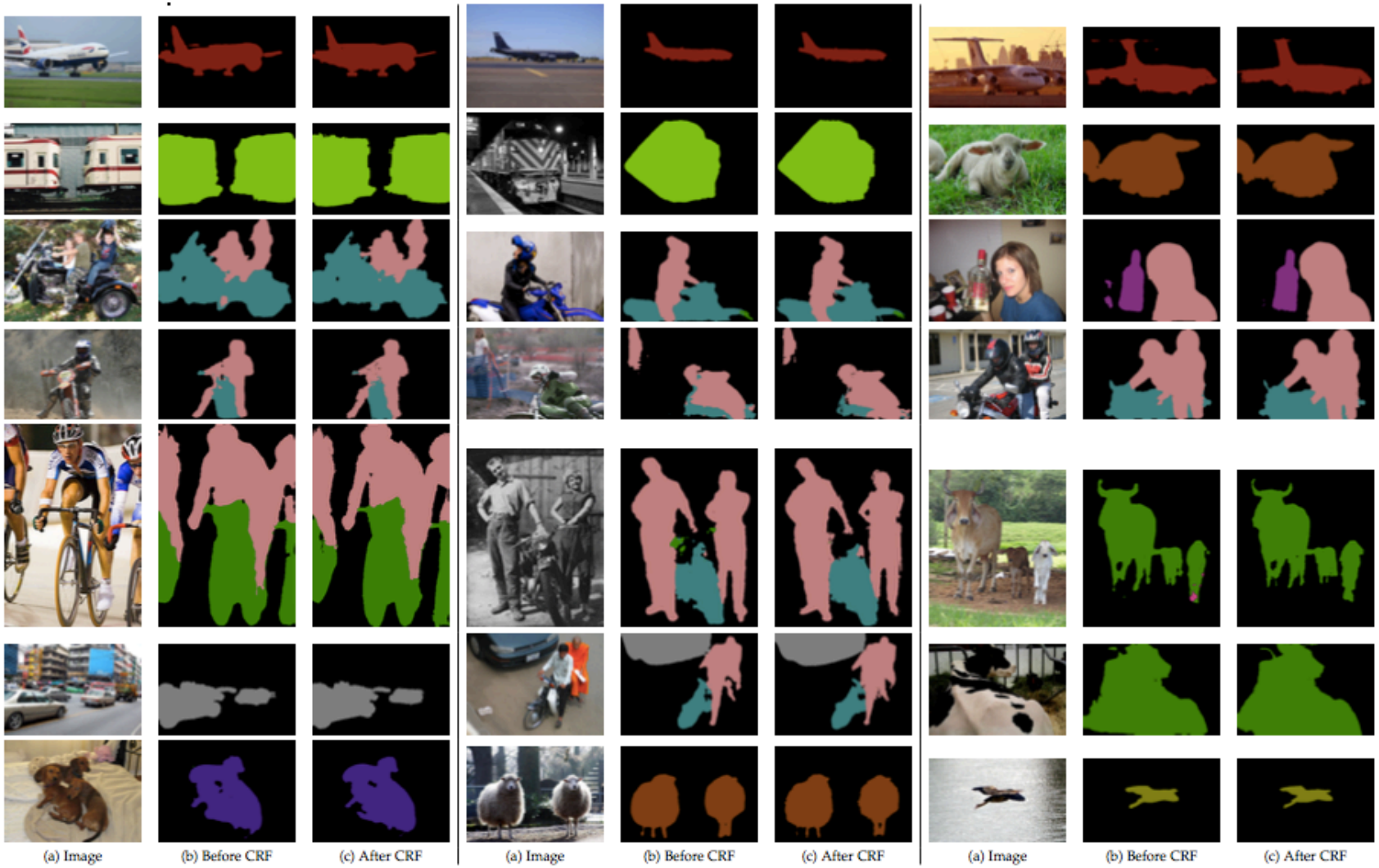
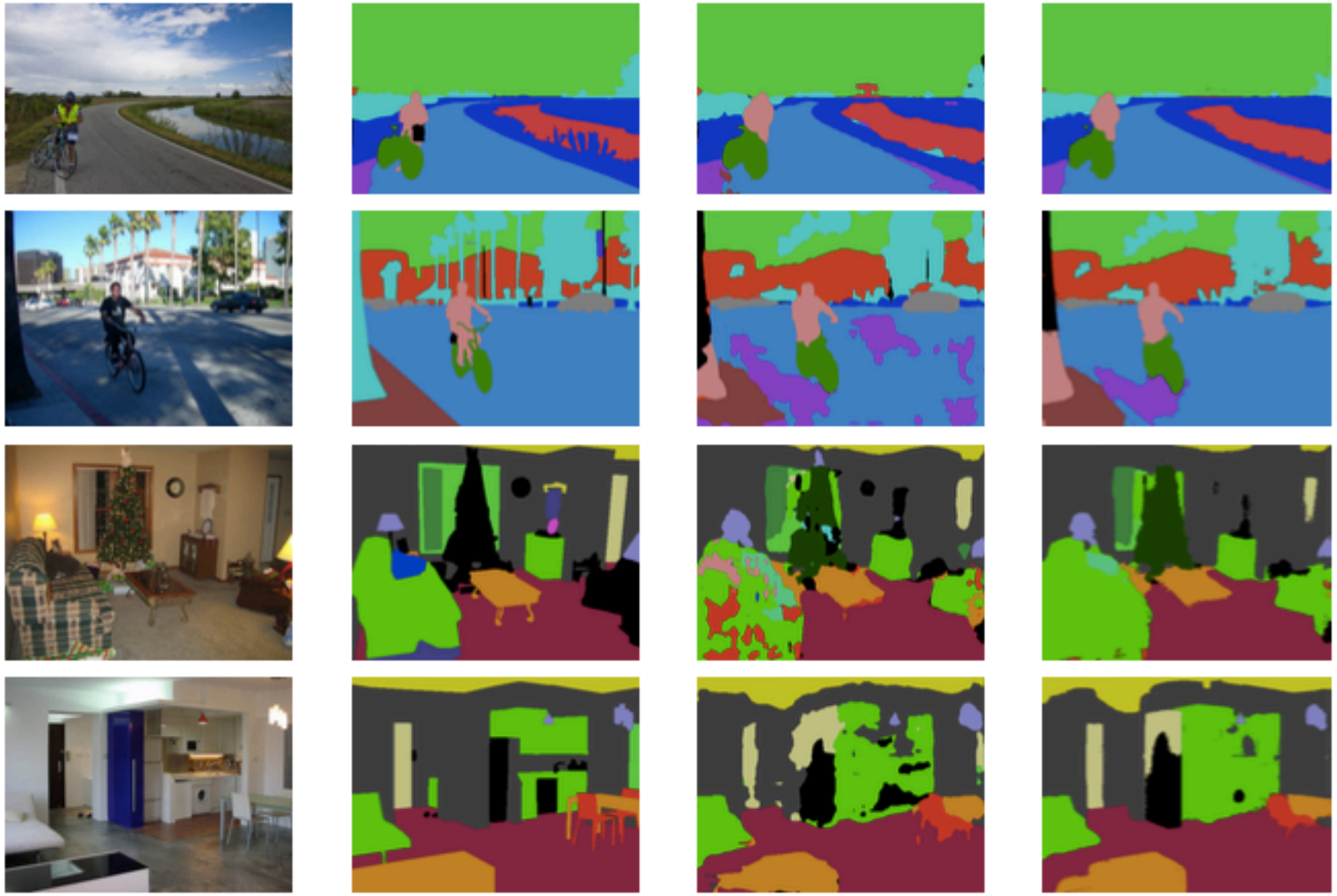


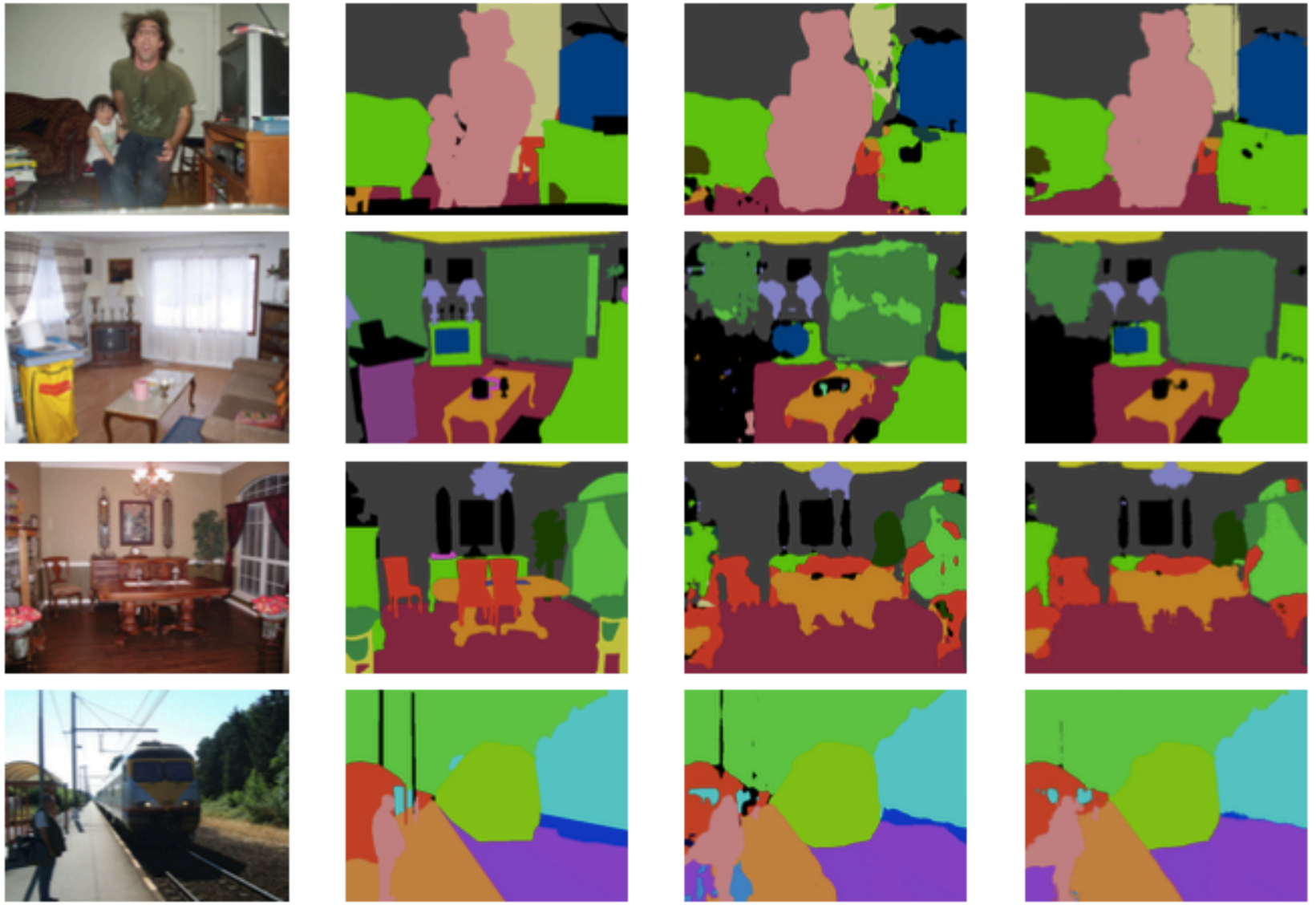
Fig. 7: Visualization of some VOC 2012 *val* results. For each row, we show the input image, the segmentation result before CRF, and the refined segmentation result after Fully Connected CRF (DeepLab-CRF).

Updated results (Deeplab v2)



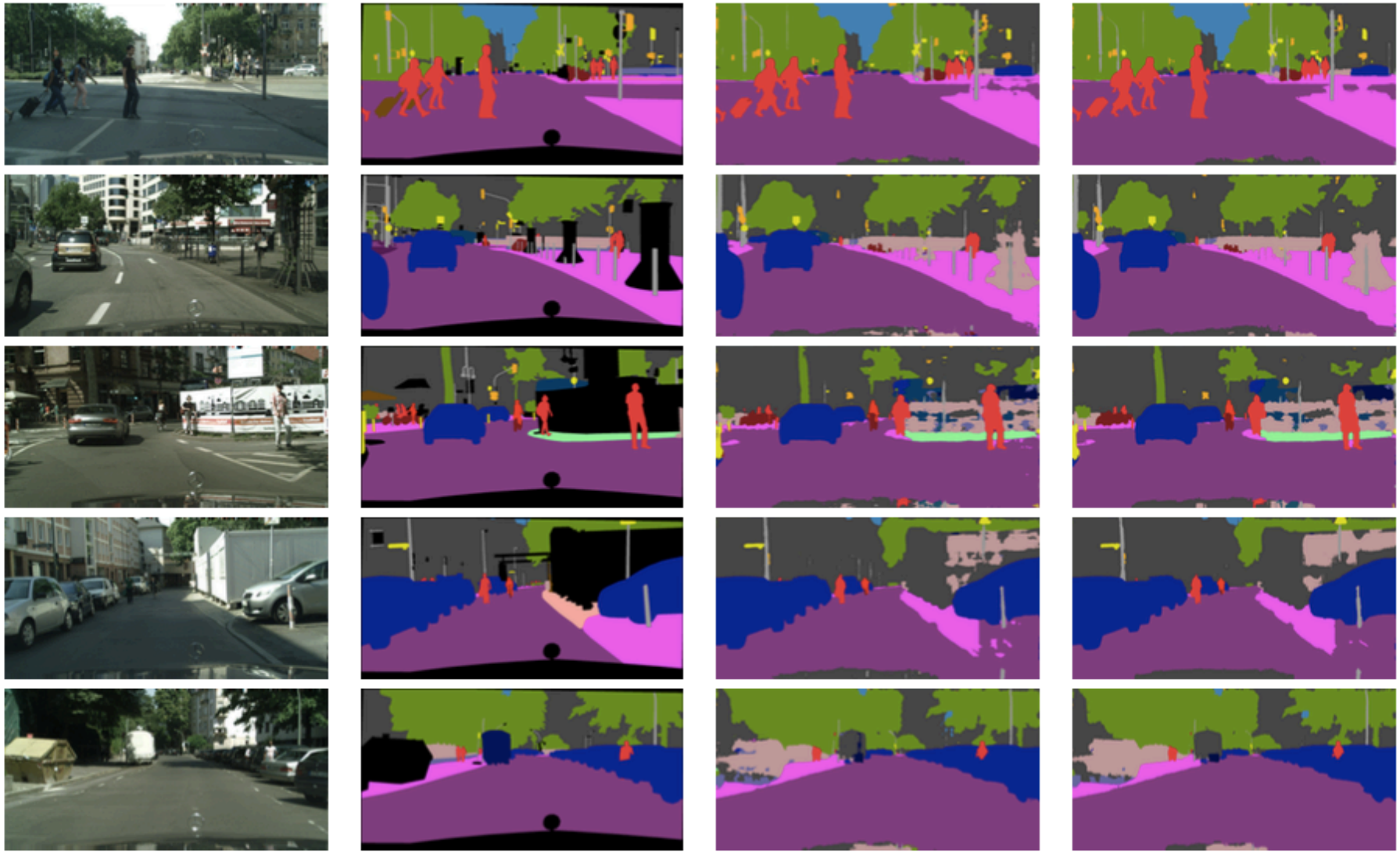
Ground truth FCNN FCNN-DCRF

Updated results (Deeplab v2)



Ground truth FCNN FCNN-DCRF

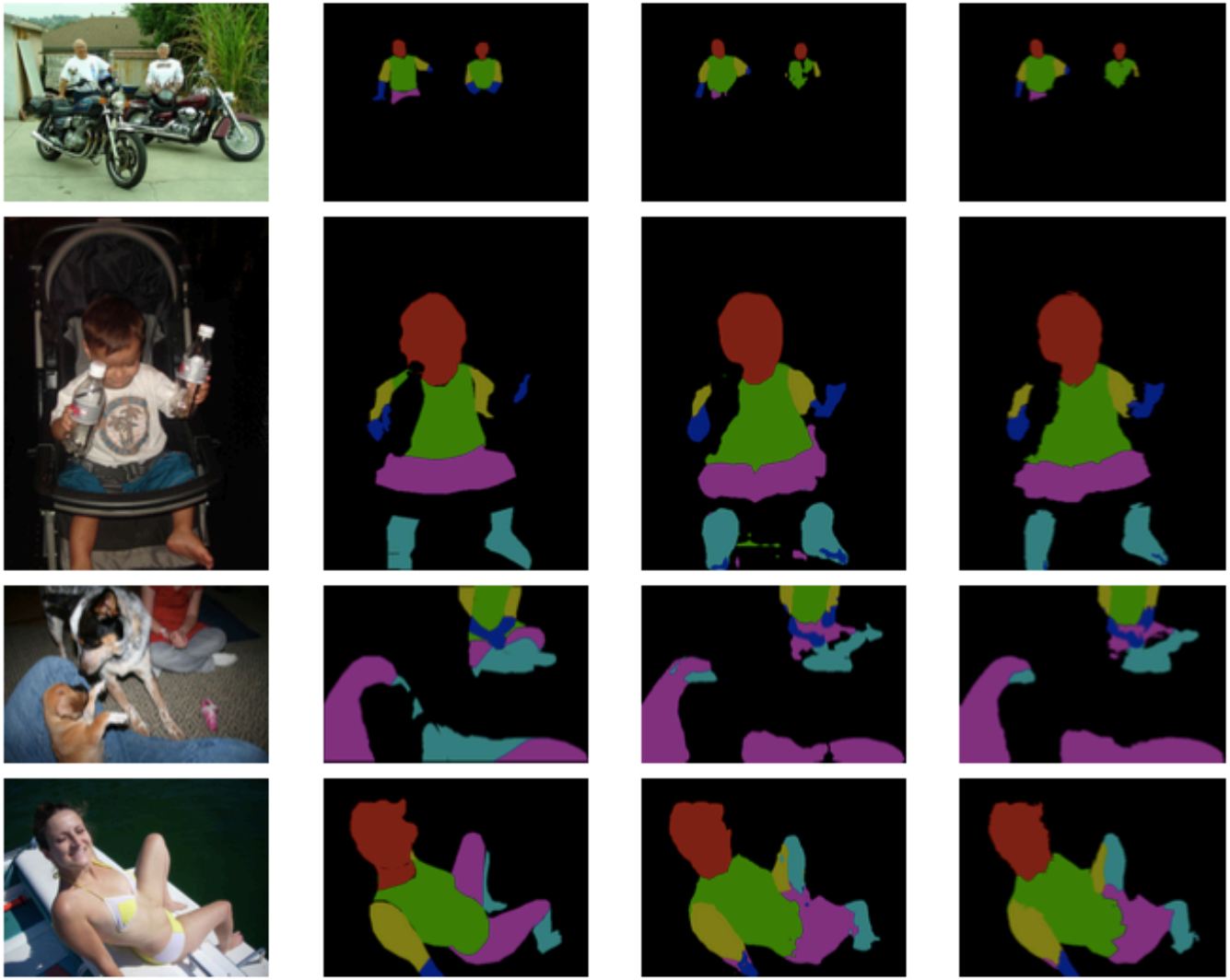
Updated results (Deeplab v2)



(a) Image (b) G.T. (c) Before CRF (d) After CRF

Fig. 11: Visualization results on Cityscapes. For each row, we show the input image, the ground-truth, and our DeepLab results before and after CRF.

Updated results (Deeplab v2)



See also: S. Tsogkas, G. Papandreou, I. Kokkinos, and A. Vedaldi, Semantic Part Segmentation using high-level guidance, Arxiv, 2015

Steps of Tracking

: predict next state of the object given past measurements

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

: combine prediction and currrent measurement to yield new distribution.

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

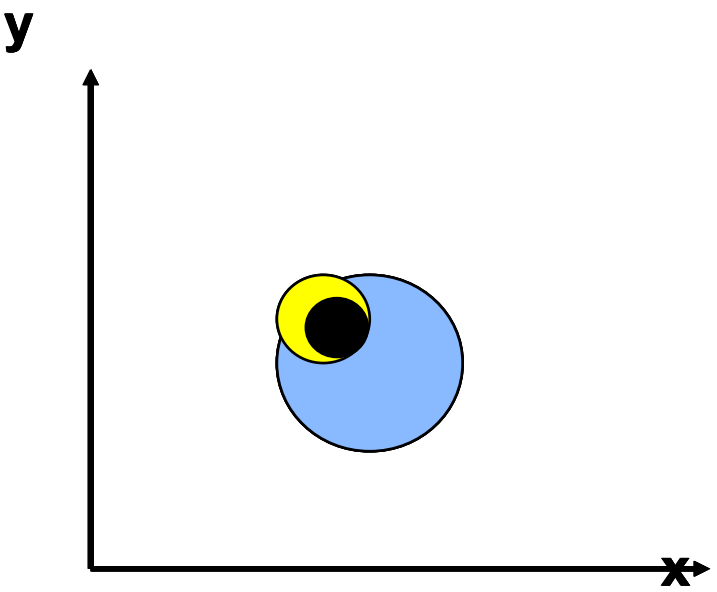
Tracking: from t to t+1

Initial position

Prediction

Measurement

Update



Induction Step: Prediction

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

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

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

Law of total probability

$$P(A) = \int P(A, B) dB$$

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}

$$P(A, B) = P(A \mid B) P(B)$$

Induction Step: Prediction

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

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

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

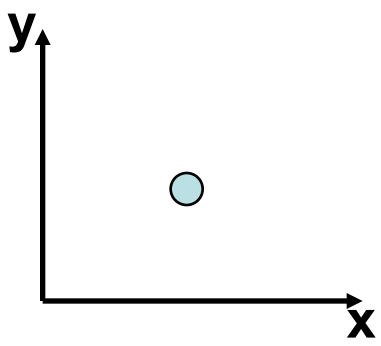
$$= \int P(X_t \mid X_{t-1}, y_0, \ldots, y_{t-1}) P(X_{t-1} \mid y_0, \ldots, y_{t-1}) dX_{t-1}$$

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

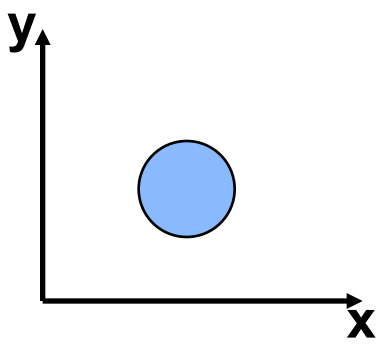
Independence assumption

What we just described:

Initial position



Prediction



Induction Step: Correction

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

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

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

Bayes rule

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

Induction Step: Correction

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

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

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

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

Independence assumption
(observation y_t depends only on state X_t)

Induction Step: Correction

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

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

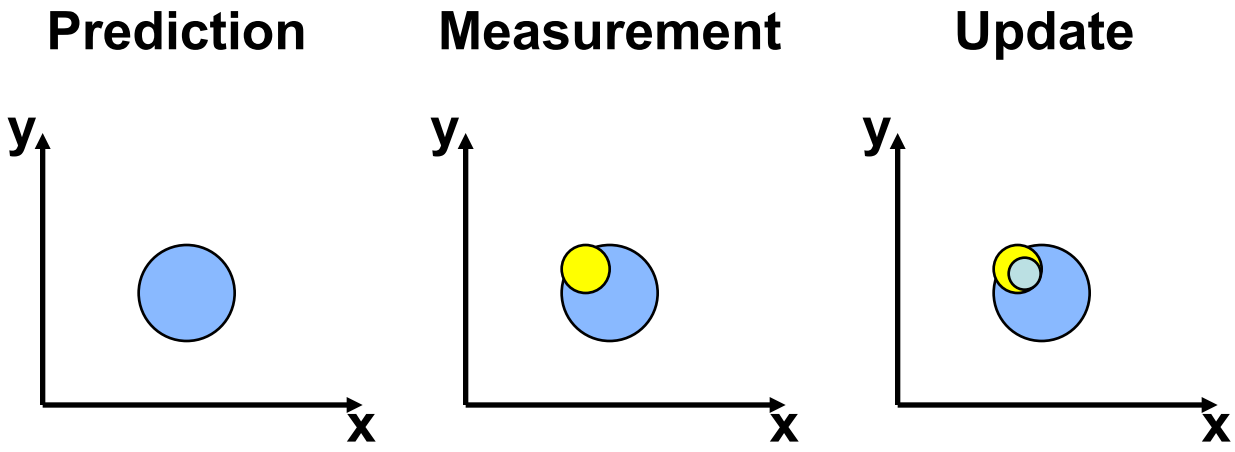
$$= \frac{P(y_t | X_t, y_0, \ldots, y_{t-1}) P(X_t | y_0, \ldots, y_{t-1})}{P(y_t | y_0, \ldots, y_{t-1})}$$

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

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

Conditioning on X_t

What we just described



Summary: Prediction and Correction

- Prediction:

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

Summary: Prediction and Correction

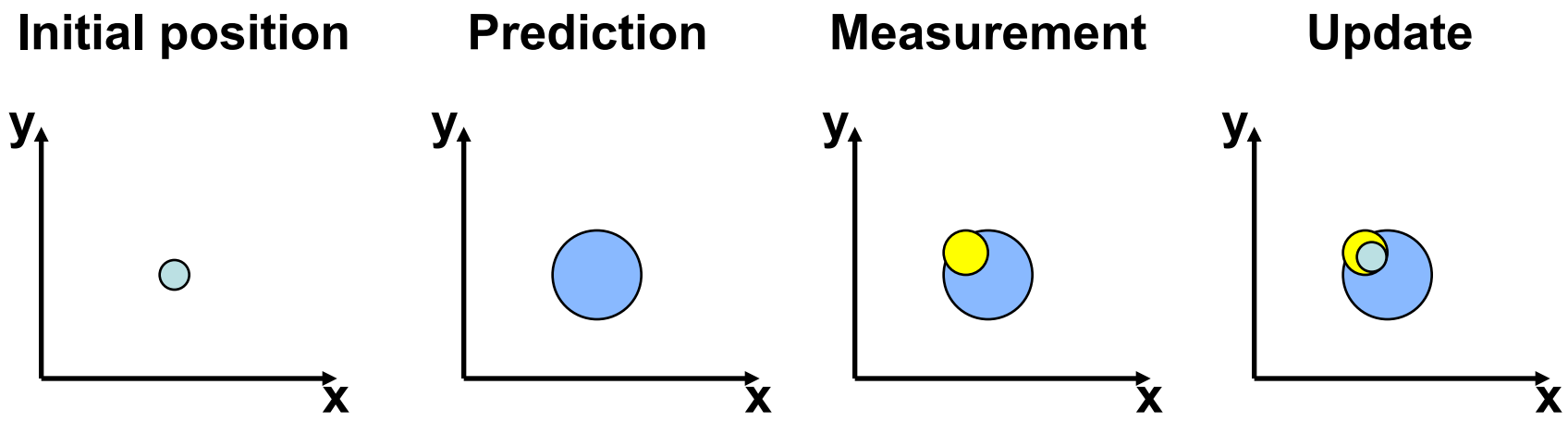
- Prediction:

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

- Correction:

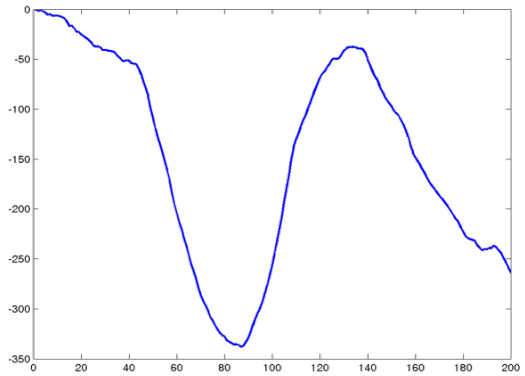
$$P(X_t \mid y_0, \ldots, y_t) = \frac{\overbrace{P(y_t \mid X_t)}^{\text{Observation model}} \overbrace{P(X_t \mid y_0, \ldots, y_{t-1})}^{\text{Predicted estimate}}}{\int P(y_t \mid X_t) P(X_t \mid y_0, \ldots, y_{t-1}) dX_t}$$

Predict-measure-update



This lecture

- Tracking
  - ➢ Linear Dynamical Models



$$\begin{bmatrix} x_{t+1} \\ \delta_{t+1} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_t \\ \delta_t \end{bmatrix} + w_t$$

Linear Dynamical Models

- Linear Dynamics Assumption

$$x_t = \underbrace{D_t}_{N \times N} x_{t-1} + e_d, \quad e_d \sim N(0, \Sigma_d)$$

- Linear Appearance Assumption

$$y_t = \underbrace{M_t}_{M \times N} x_t + e_m, \quad e_m \sim N(0, \Sigma_m)$$

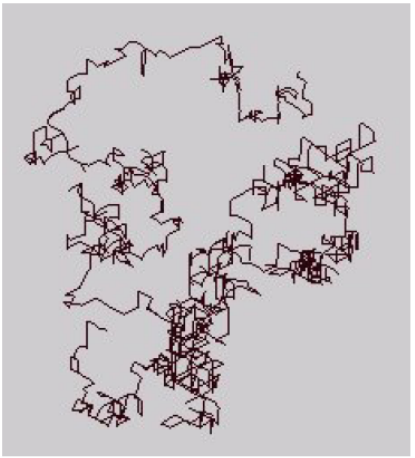
Example: Randomly Drifting Points

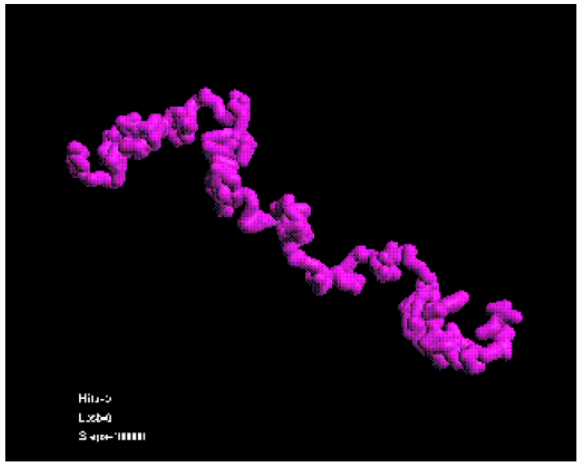
- State: position of object
  - Only motion: random noise term.

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

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

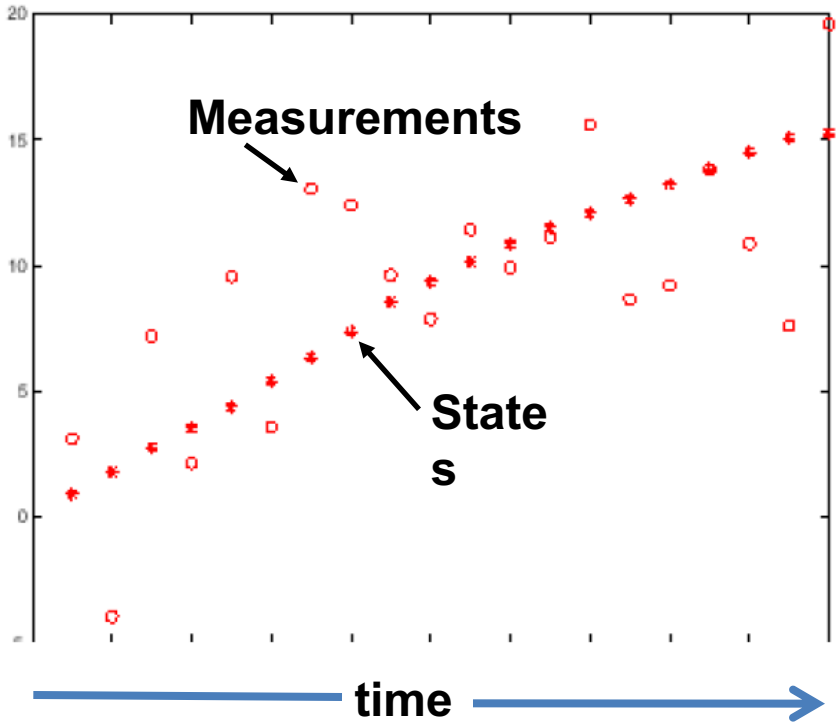
$$x_t = D_t x_{t-1} + noise = I p_{t-1} + noise$$





cic.nist.gov/lipman/sciviz/images/random3.gif
http://www.grunch.net/synergetics/images/rand
3.jpg

Example: Constant Velocity (1D Points)



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

$$y_t = M x_t + noise =$$

Example: Constant Velocity (1D Points)

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

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

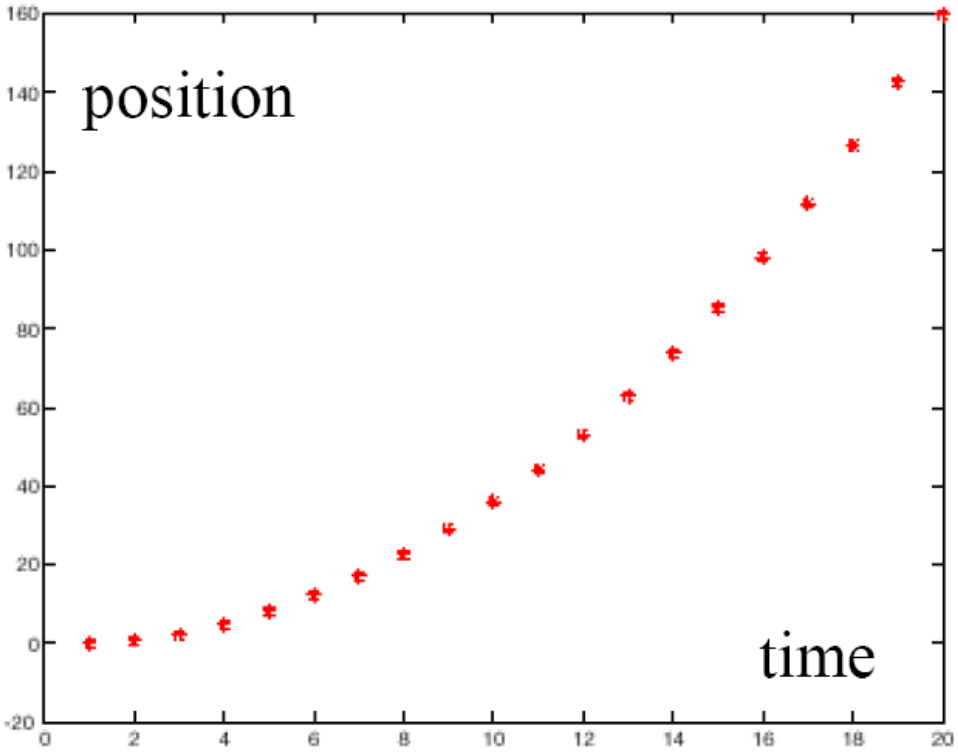
(greek letters denote noise terms)

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

- Measurement is position only

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

Example: Constant Acceleration (1D Points)



Example: Constant Acceleration (1D Points)

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

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

$$p_t = p_{t-1} + (\Delta t) v_{t-1} + \varepsilon$$

$$v_t =$$

$$a_t =$$

(greek letters denote noise terms)

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

- Measurement is position

$$y_t = M x_t + noise =$$

Example: Constant Acceleration (1D Points)

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

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

(greek letters denote noise terms)

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

- Measurement is position only

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

General Motion Models

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

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

- Substitute variables to transform this into linear system

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

`With a large enough state-space, the universe is Markov' U. Grenander

The Kalman Filter

- Linear dynamical models + Gaussian noise
  - Linear transformation of Gaussian r.v.: Gaussian r.v.

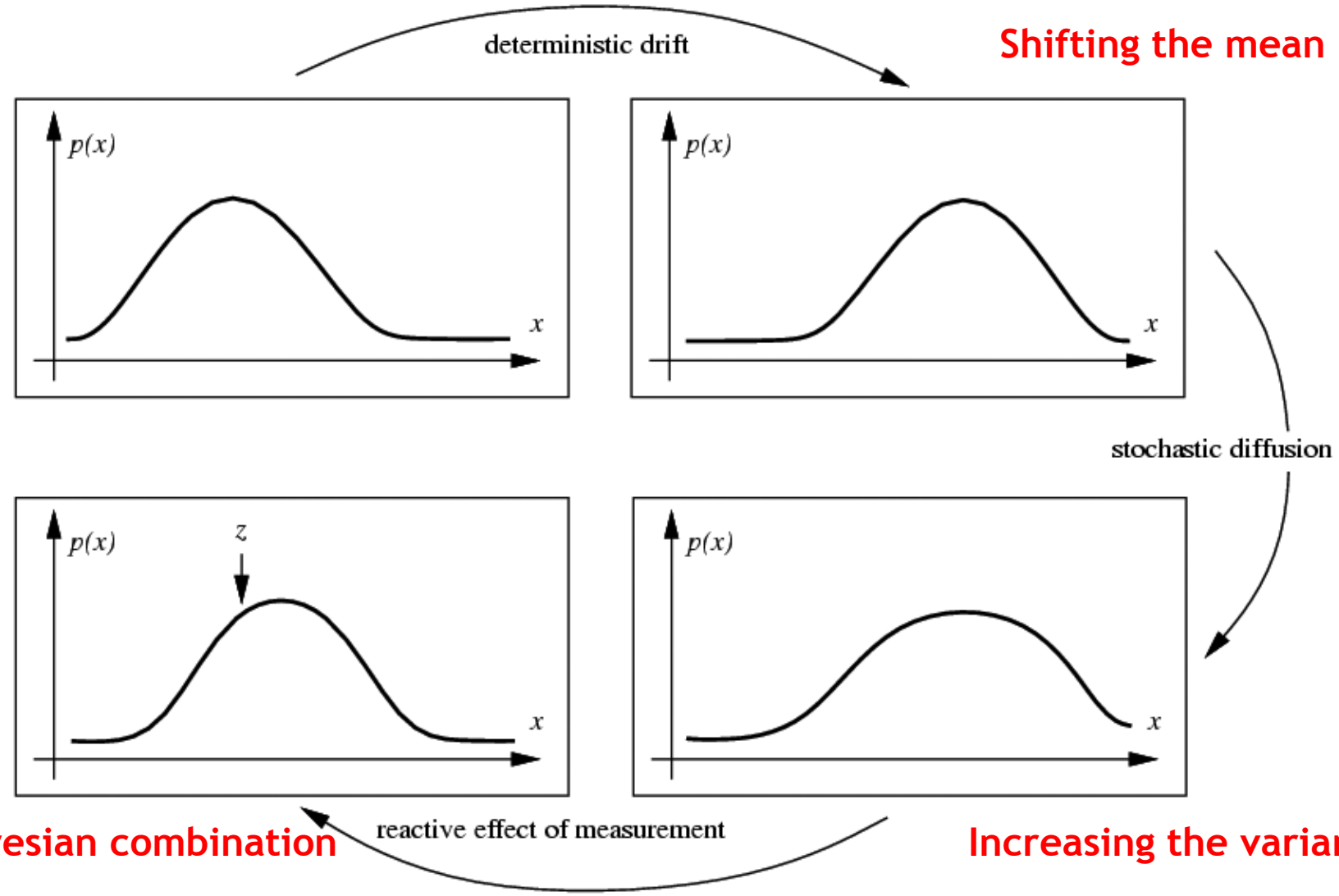
$$x_t \sim N(D_t x_{t-1}, \Sigma_d)$$

$$y_t \sim N(M_t x_t, \Sigma_m)$$

- The predicted/corrected state distributions are Gaussian
  - Analytical expressions for Gaussian parameters

- Applications: finance, aerospace, robotics …. vision
  - `Top 10 algorithms of the century'

Propagation of Gaussian densities



Kalman Filter for 1D State

Want to represent and update

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

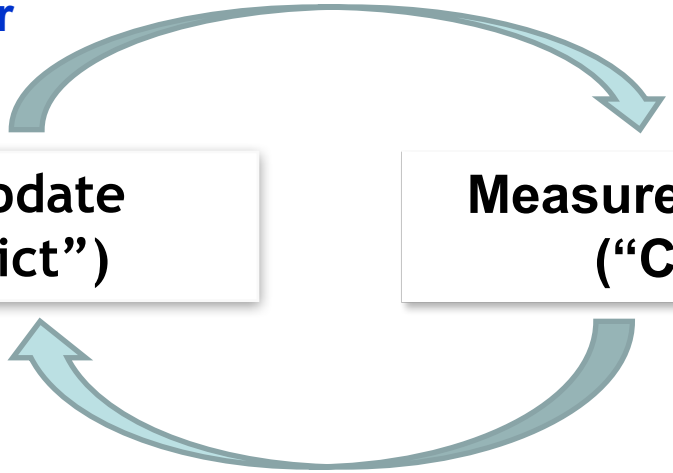
$$P(x_t | y_0, \ldots, y_t) = N(\mu_t^+, (\sigma_t^+)^2)$$

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")

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

Mean and std. dev. of predicted state:

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

Time advances

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

Mean and std. dev. of corrected state:

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

1D Kalman Filter: Prediction

- Given:
  - Linear dynamic model

$$x_t \sim N(dx_{t-1}, \sigma_d)$$

  - Estimated distribution for previous state

$$P(x_{t-1}|y_1, \ldots y_{t-1}) = N(\mu_{t-1}^+, \sigma_{t-1}^+)$$

- Wanted: predicted distribution for next state

$$P(x_t|y_1, \ldots y_{t-1}) = N(\mu_t^-, \sigma_t^-)$$

- New parameters

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

1D Kalman Filter: Correction

- Given
  - Prediction

$$P(x_t|y_1,\ldots y_{t-1}) \quad = \quad N(\mu_t^-,\sigma_t^-)$$

  - Measurement model

$$y_t \quad \sim \quad N(mx_t,\sigma_m)$$

- Wanted

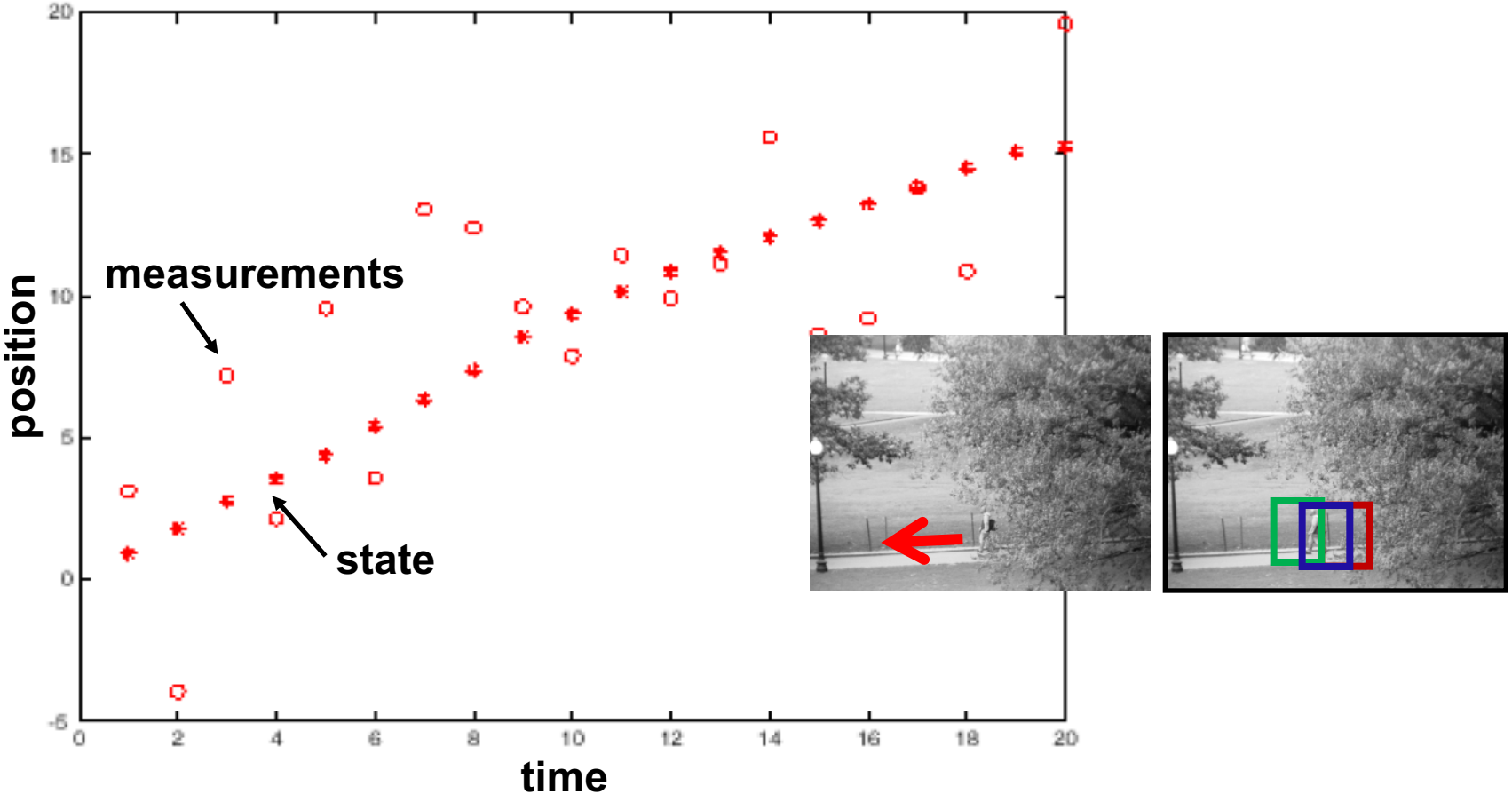
$$P(x_t|y_1,\ldots y_t) \quad \propto \quad P(y_t|x_t)P(x_t|y_1,\ldots,y_{t-1})$$

- Gaussian

$$N(\mu_t^+,\sigma_t^+)$$

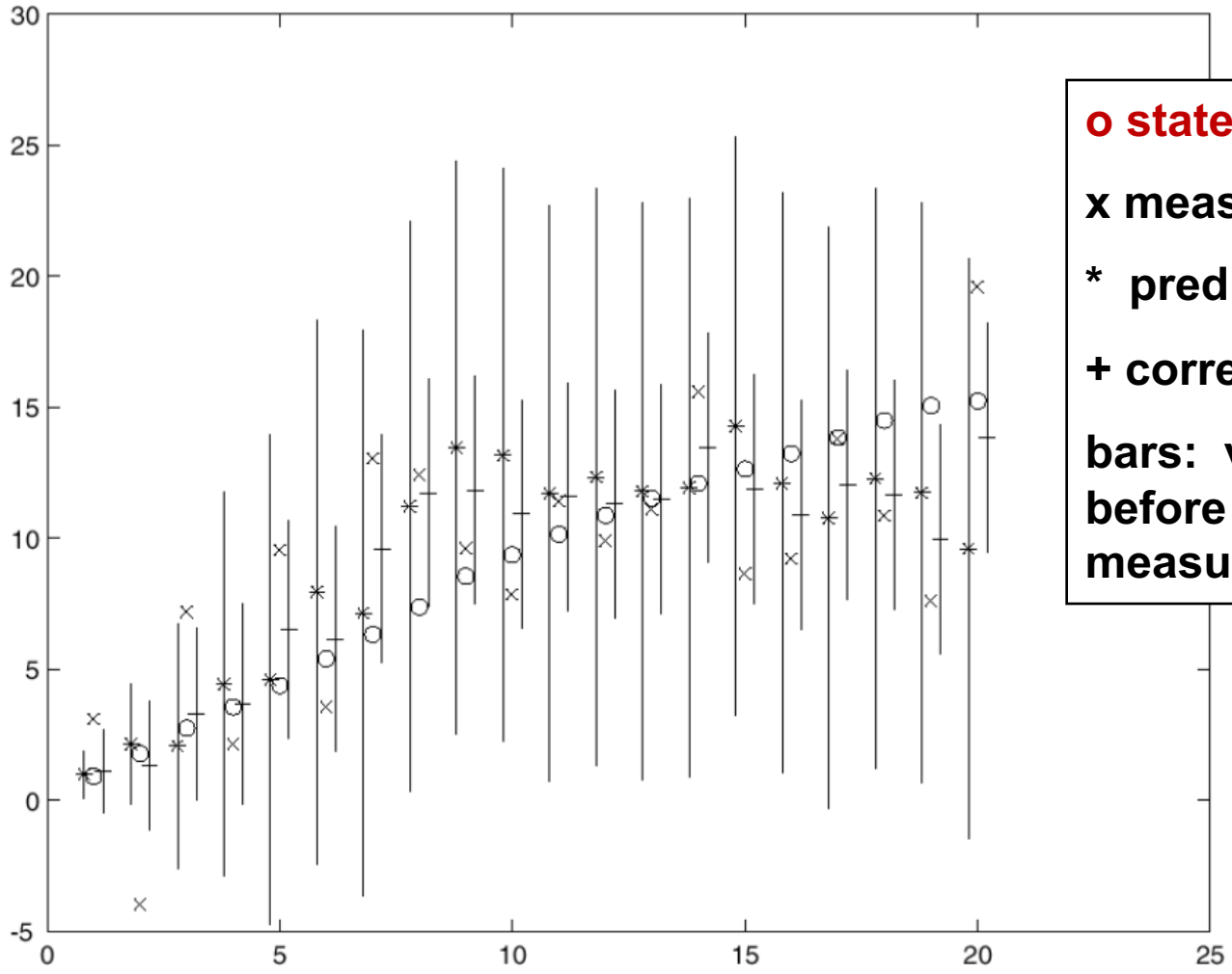
$$\mu_t^+ \quad = \quad \frac{\mu_t^-\sigma_m^2 + m y_t(\sigma_t^-)^2}{\sigma_m^2 + \left(m\sigma_t^-\right)^2} \qquad \frac{1}{\sigma_t^+} = \frac{1}{\left(\sigma_t^-\right)^2} + \frac{1}{\sigma_m^2}$$

Recall: Constant Velocity Example



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

Constant Velocity Model



o state

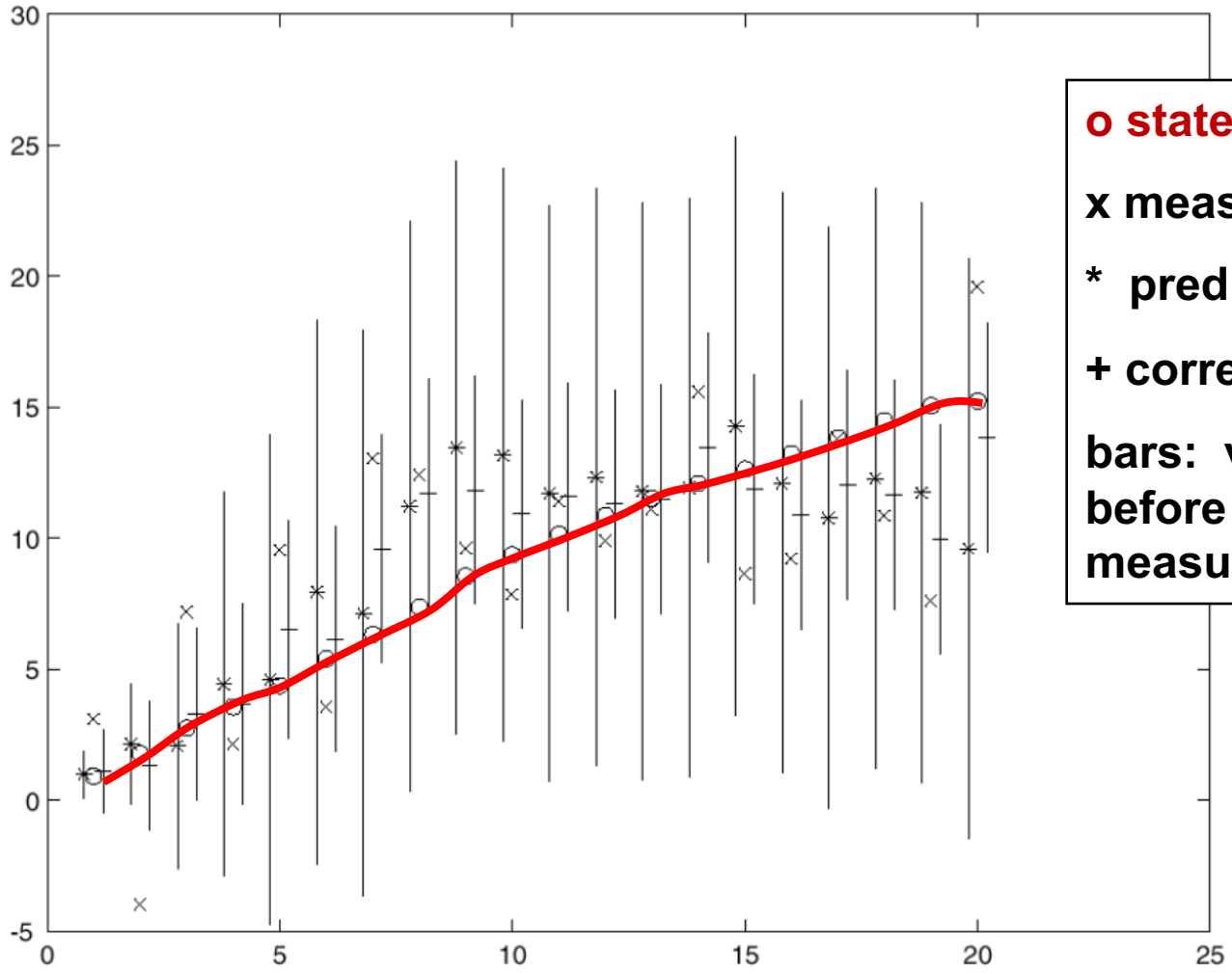
x measurement

*** predicted mean estimate**

+ corrected mean estimate

bars: variance estimates before and after measurements

Constant Velocity Model



o state

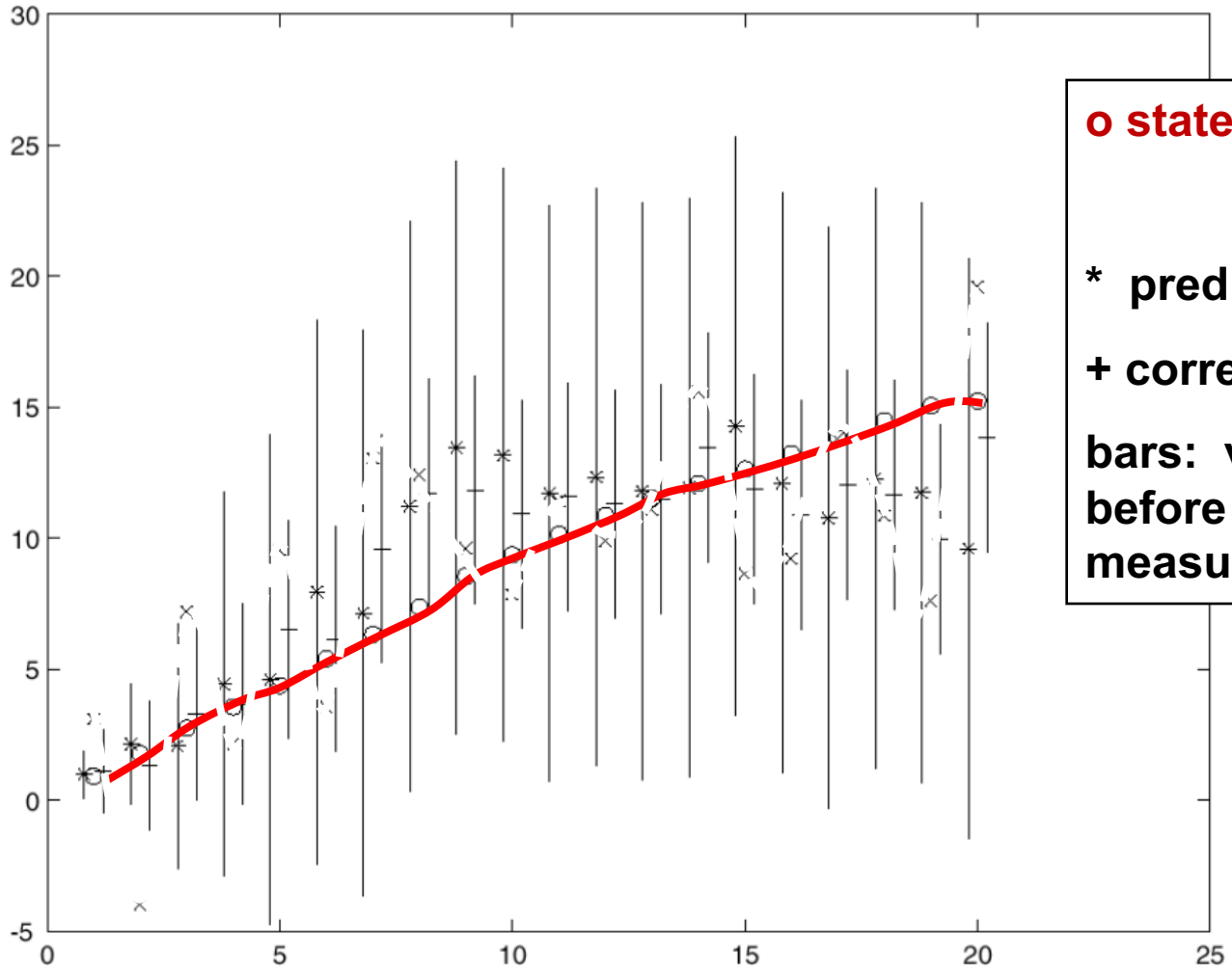
x measurement

*** predicted mean estimate**

+ corrected mean estimate

bars: variance estimates before and after measurements

Constant Velocity Model



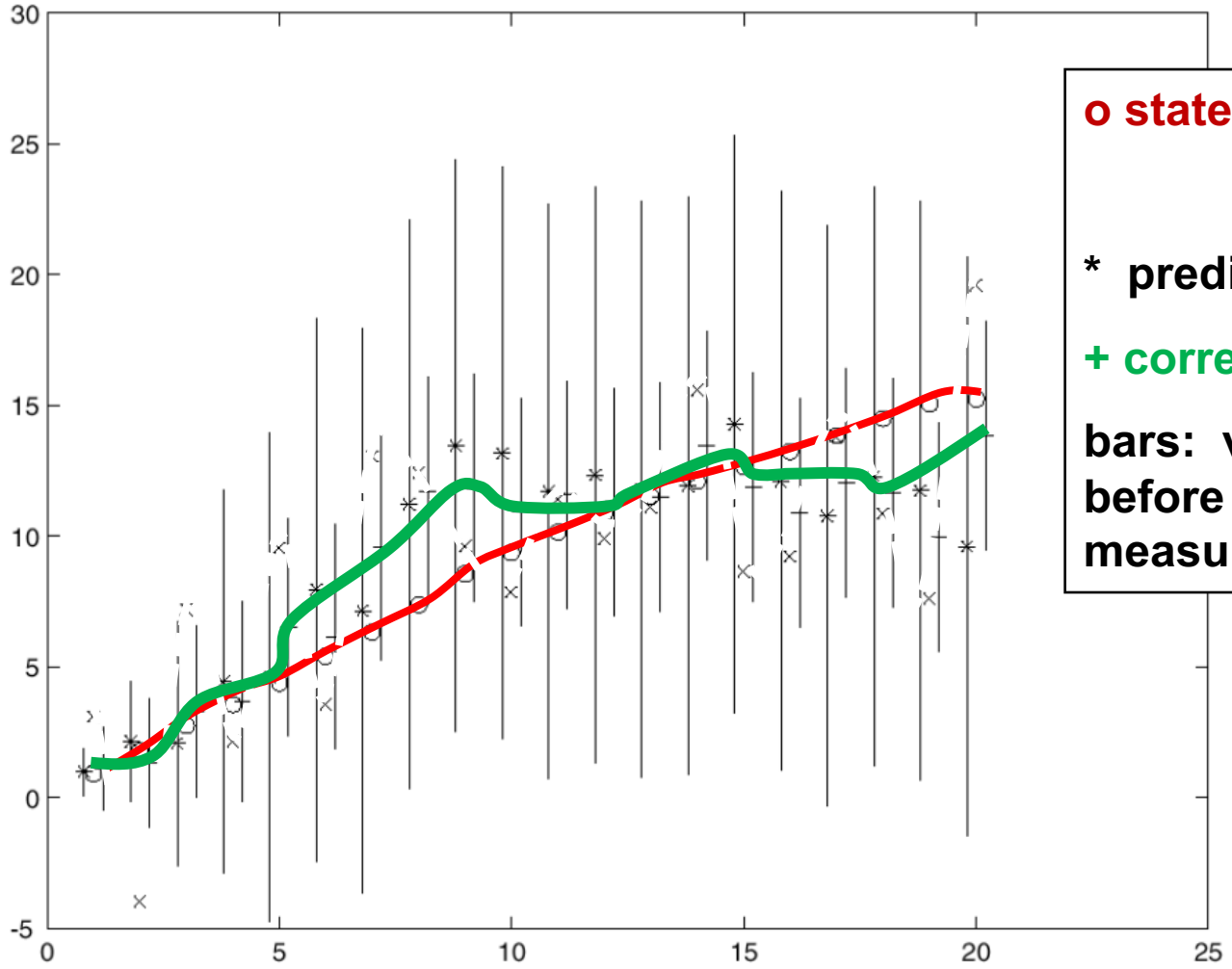
o state

*** predicted mean estimate**

+ corrected mean estimate

bars: variance estimates before and after measurements

Constant Velocity Model



o state

*** predicted mean estimate**

+ corrected mean estimate

bars: variance estimates before and after measurements

Kalman Filter: General Case (>1dim)

- What if state vectors have more than one dimension?

PREDICT

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

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

CORRECT

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

$$x_t^+ = x_t^- + K_t \left(y_t - M_t x_t^- \right)$$

"residual"

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

More weight on residual when measurement error covariance approaches 0.

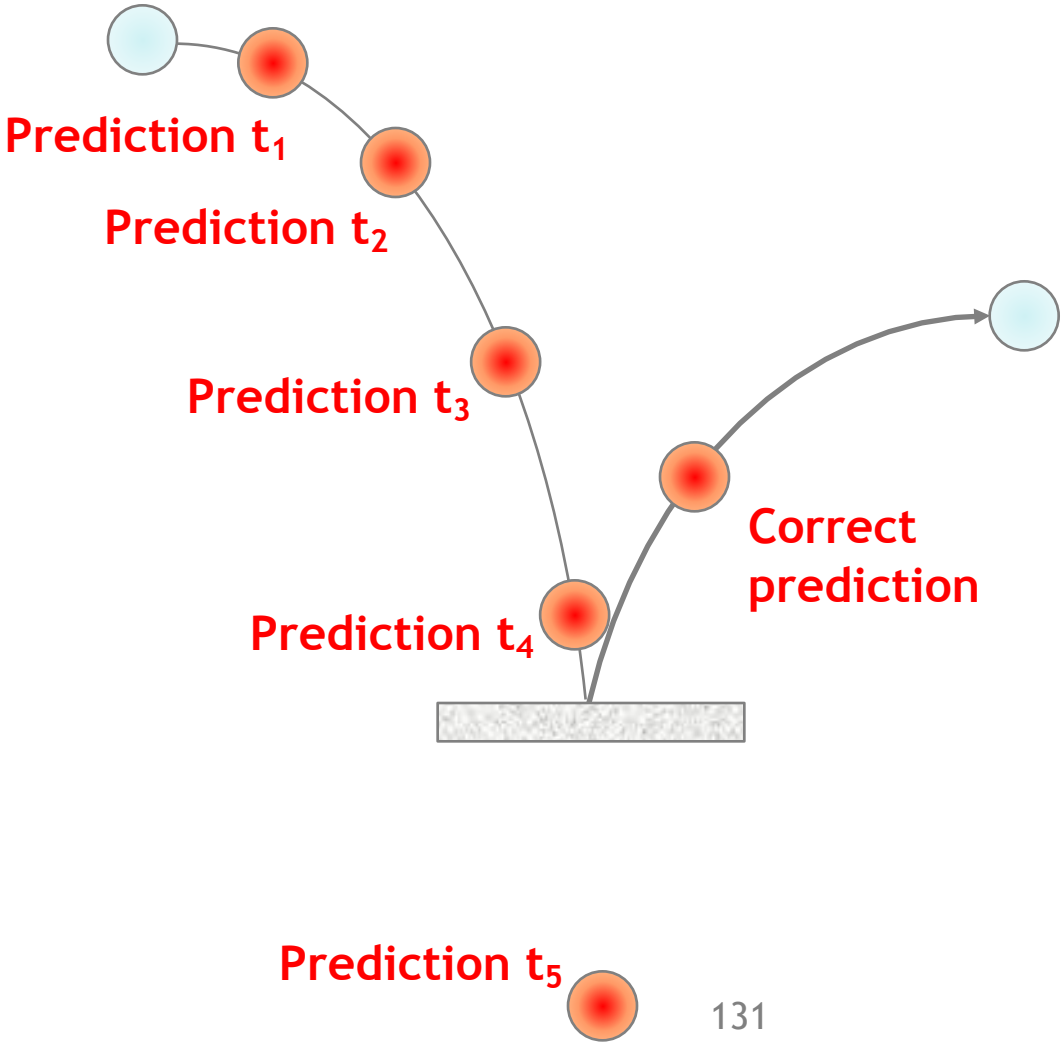
Less weight on residual as a priori estimate error covariance approaches 0.

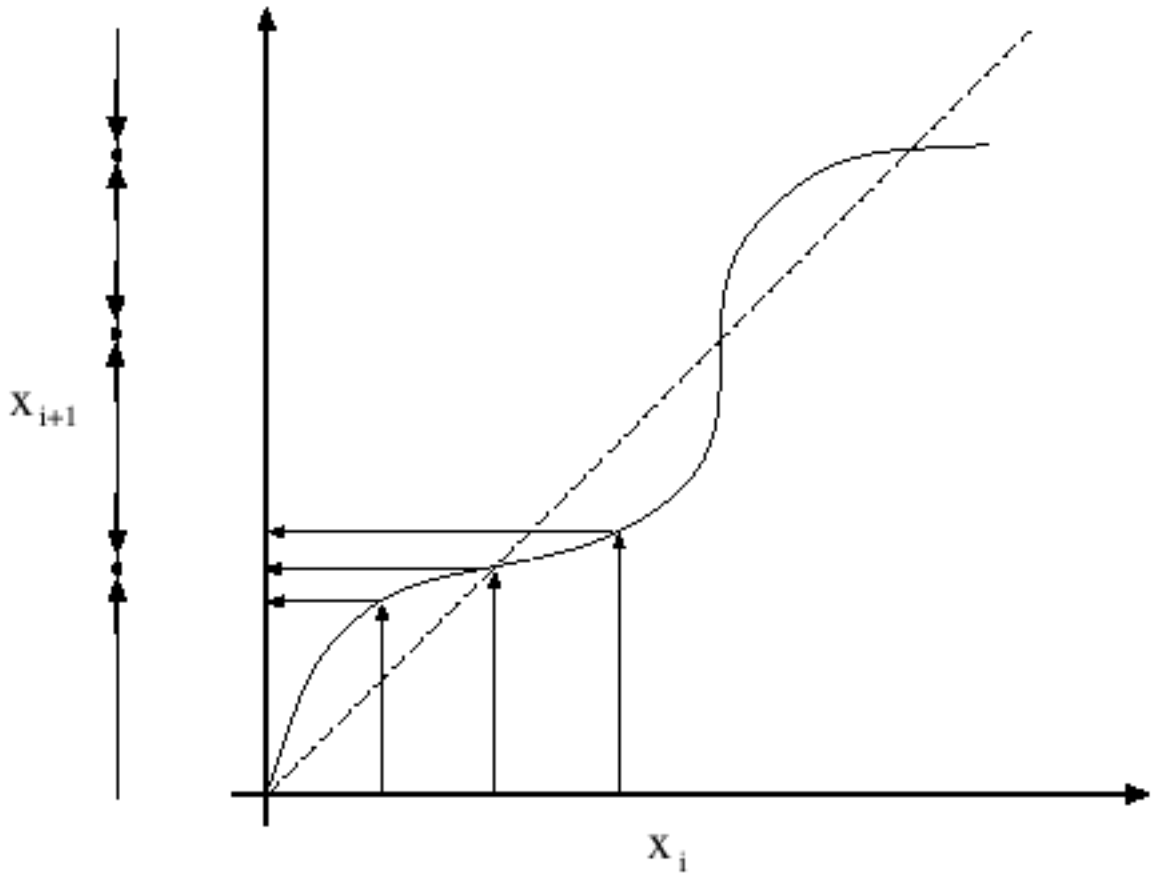
This lecture

- Tracking
  - Nonlinear models

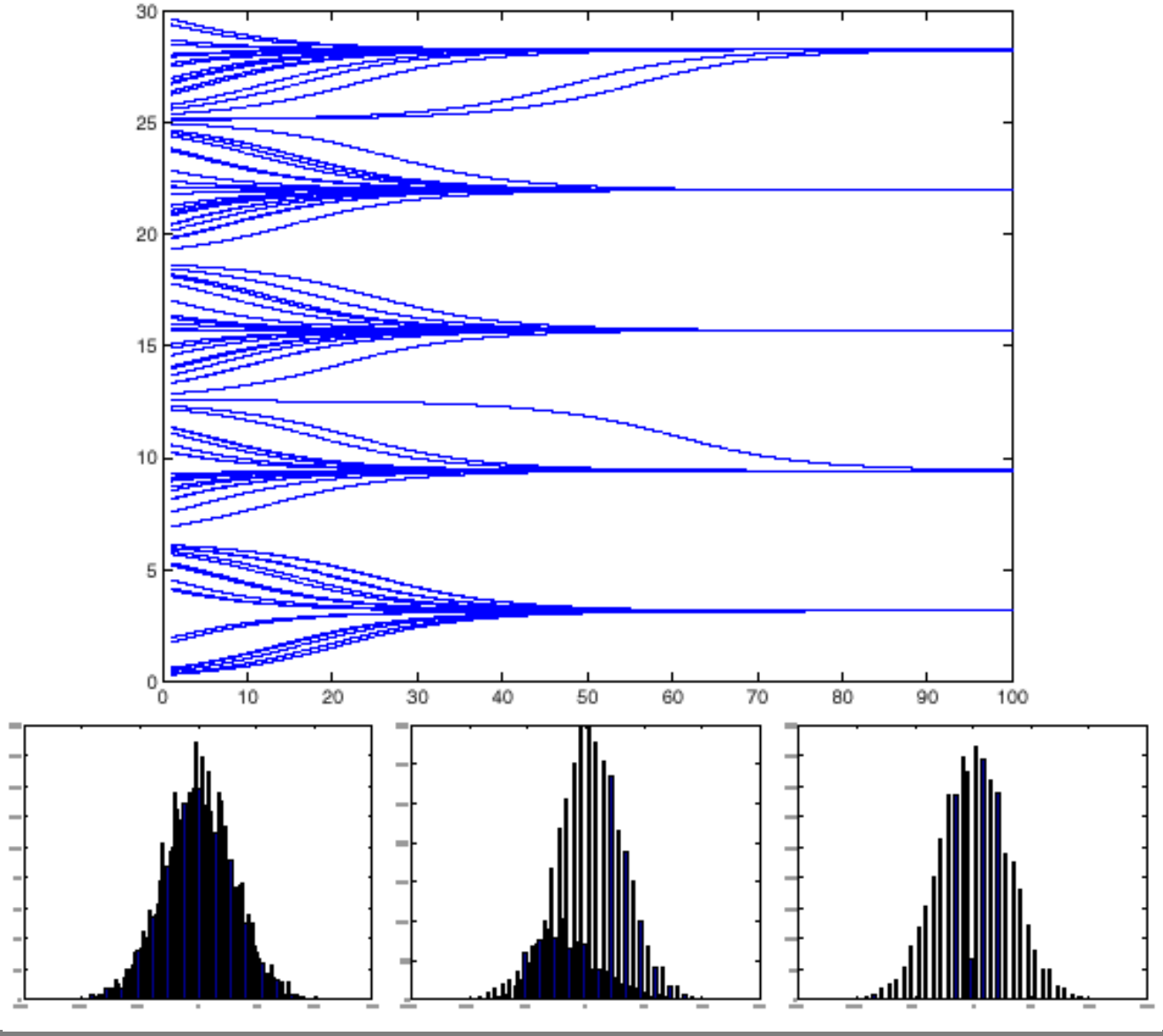
Ball Example: What Goes Wrong Here?

- Assuming constant acceleration model
- Prediction is too far from true position to compensate…

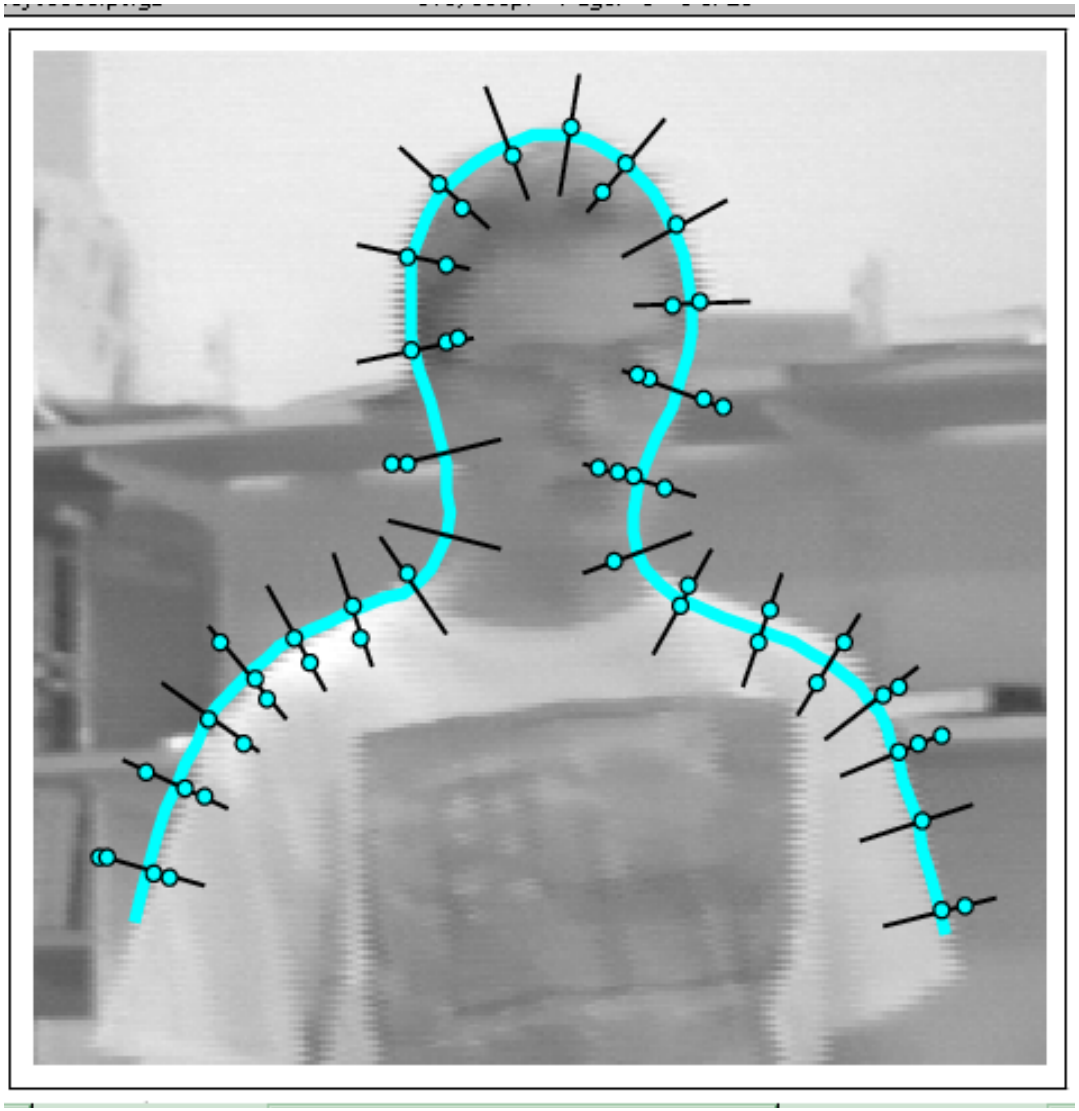




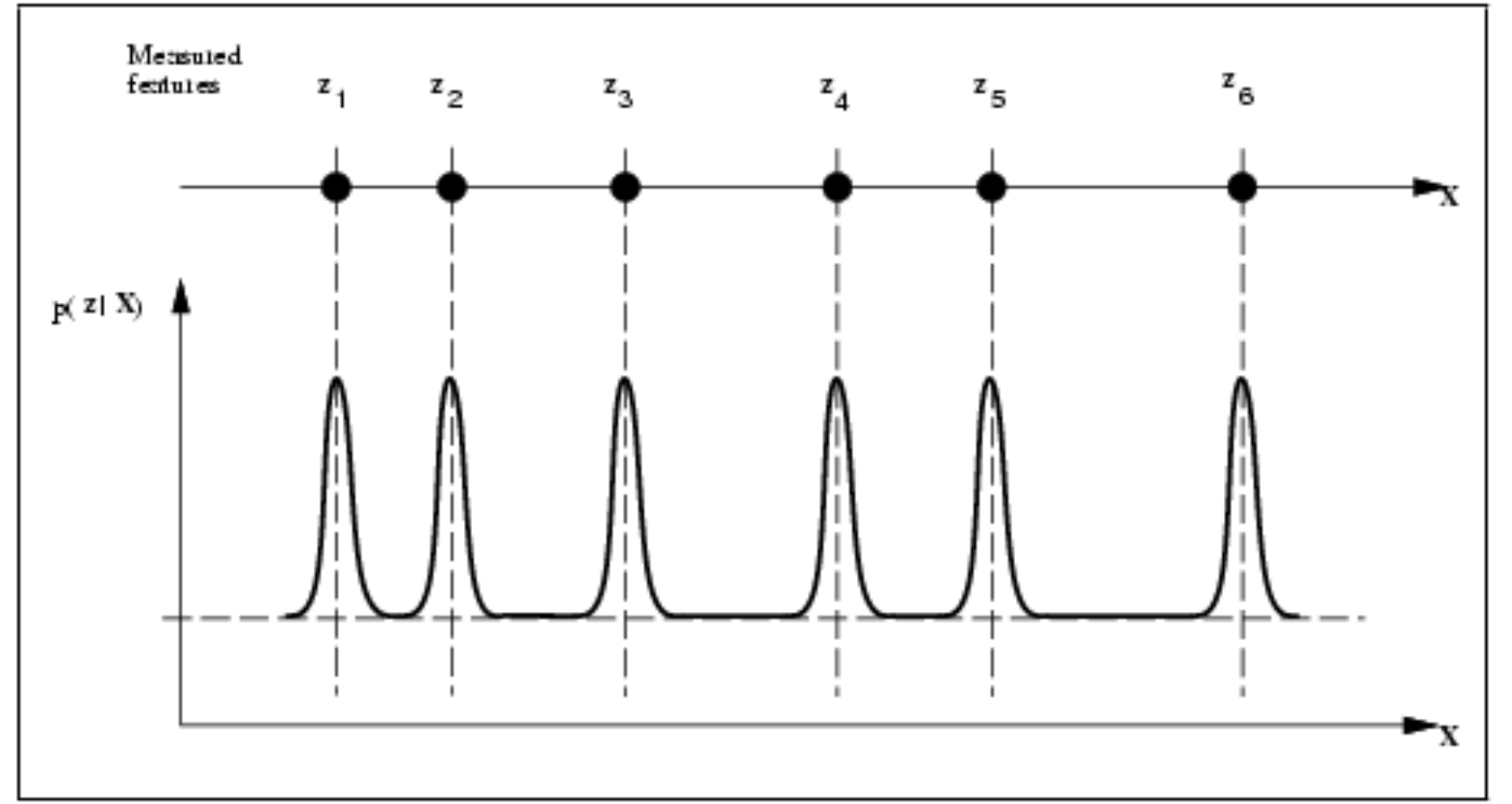
$$x_{i+1} = x_i + \epsilon \sin(x_i)$$


30
25
20
15
10
5
0
0
10
20
30
40
50
60
70
80
90
100

Nonlinear Observation Likelihoods

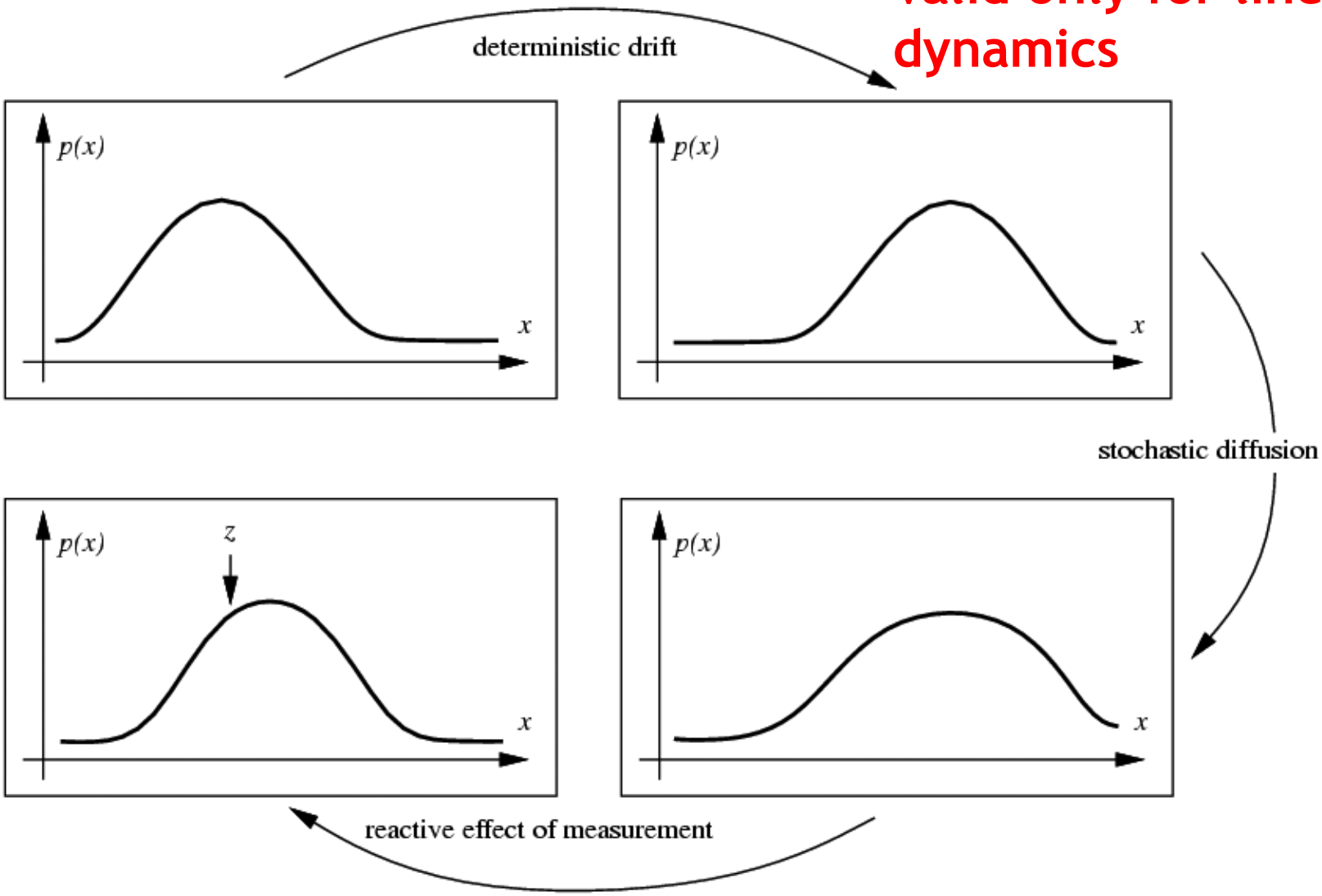


Visual clutter → observational nonlinearity

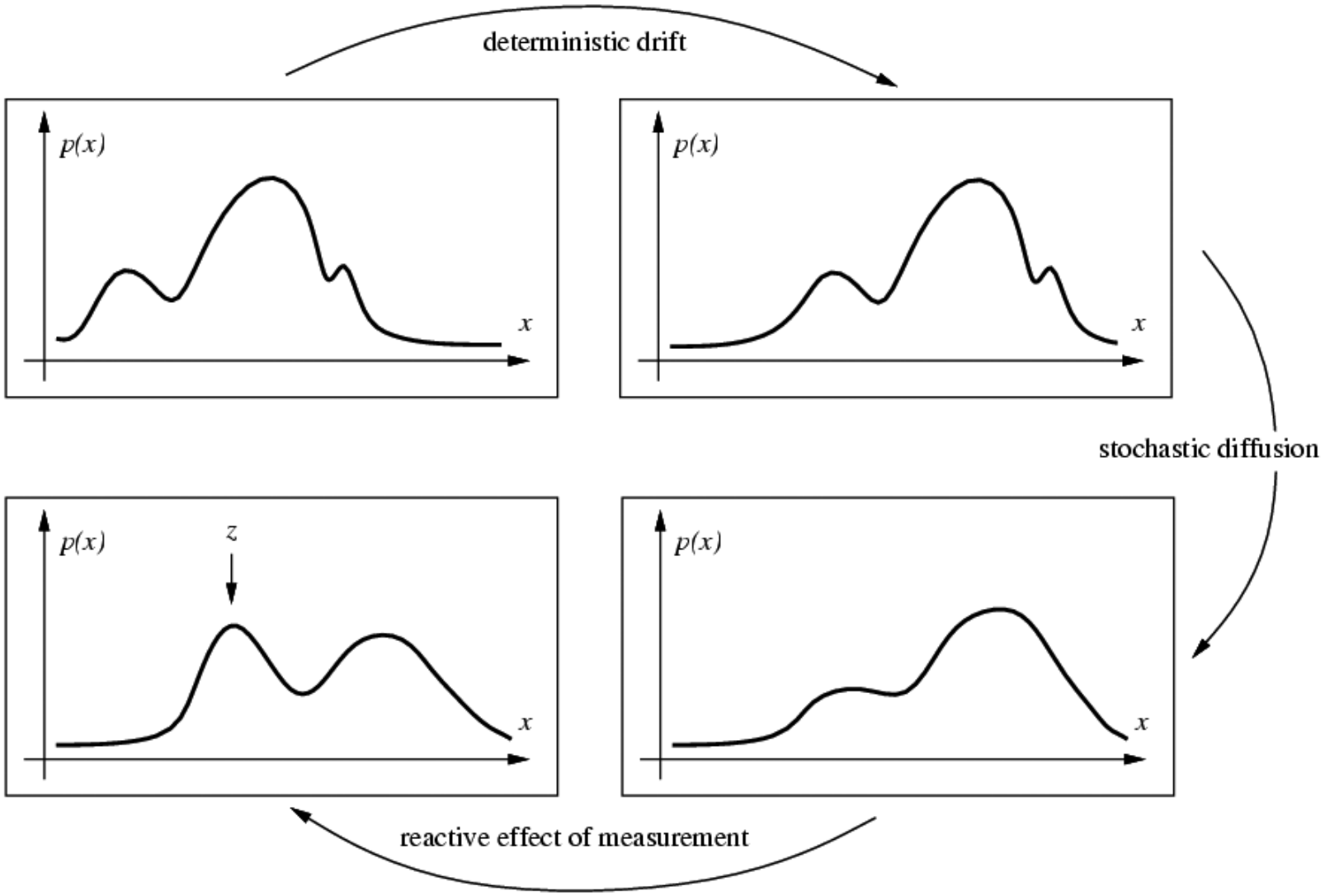


Propagation of Gaussian densities

Valid only for linear dynamics



Propagation of General Densities



Dealing with multi-modal distributions

- Mixture modelling?
  - How many mixtures?
  - Iterative fitting (EM)

- Nonparametric density estimate

$$\hat{f}(x) = \frac{1}{N} \sum_{i=1}^{N} K(|x - x^i|)$$

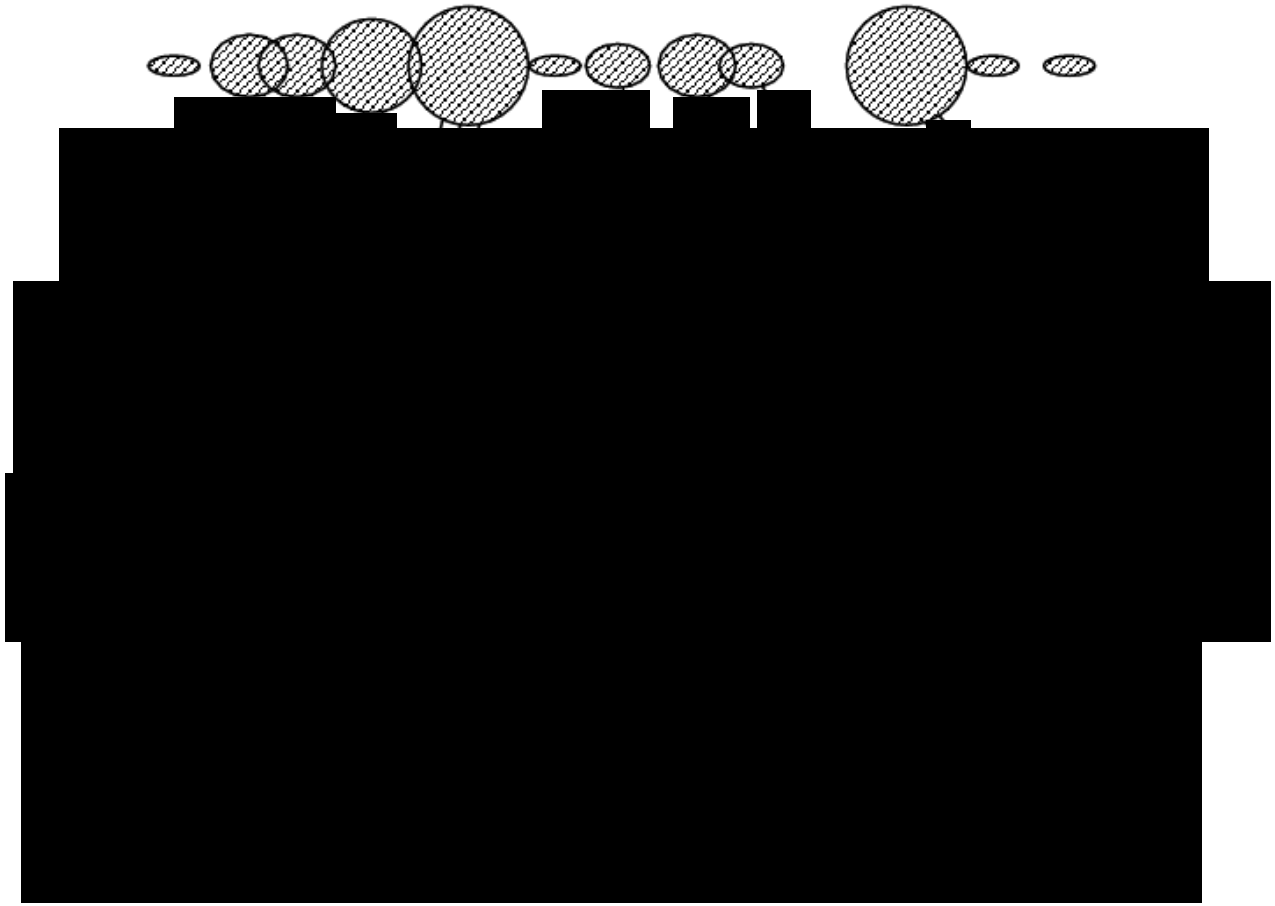
e. g. $$K(|x - x^i|) = \frac{1}{(\sqrt{2\pi}\sigma)^D} \exp(-\frac{|x - x^i|^2}{2\sigma^2})$$

Particle Filtering

- Use sampling to propagate densities over time (i.e., across frames in a video sequence).
- At each time step, represent posterior $P(X_t|Y_t)$ with weighted sample set.
- Previous time step's sample set $P(X_{t-1}|Y_{t-1})$ is passed to next time step as the effective prior.

M. Isard and A. Blake, CONDENSATION -- conditional density propagation for visual tracking, IJCV 29(1):5-28, 1998

Particle Filtering



Start with weighted samples from previous time step

Sample and shift according to dynamics model

Spread due to randomness; this is predicted density $P(X_t|Y_{t-1})$

Weight the samples according to observation density

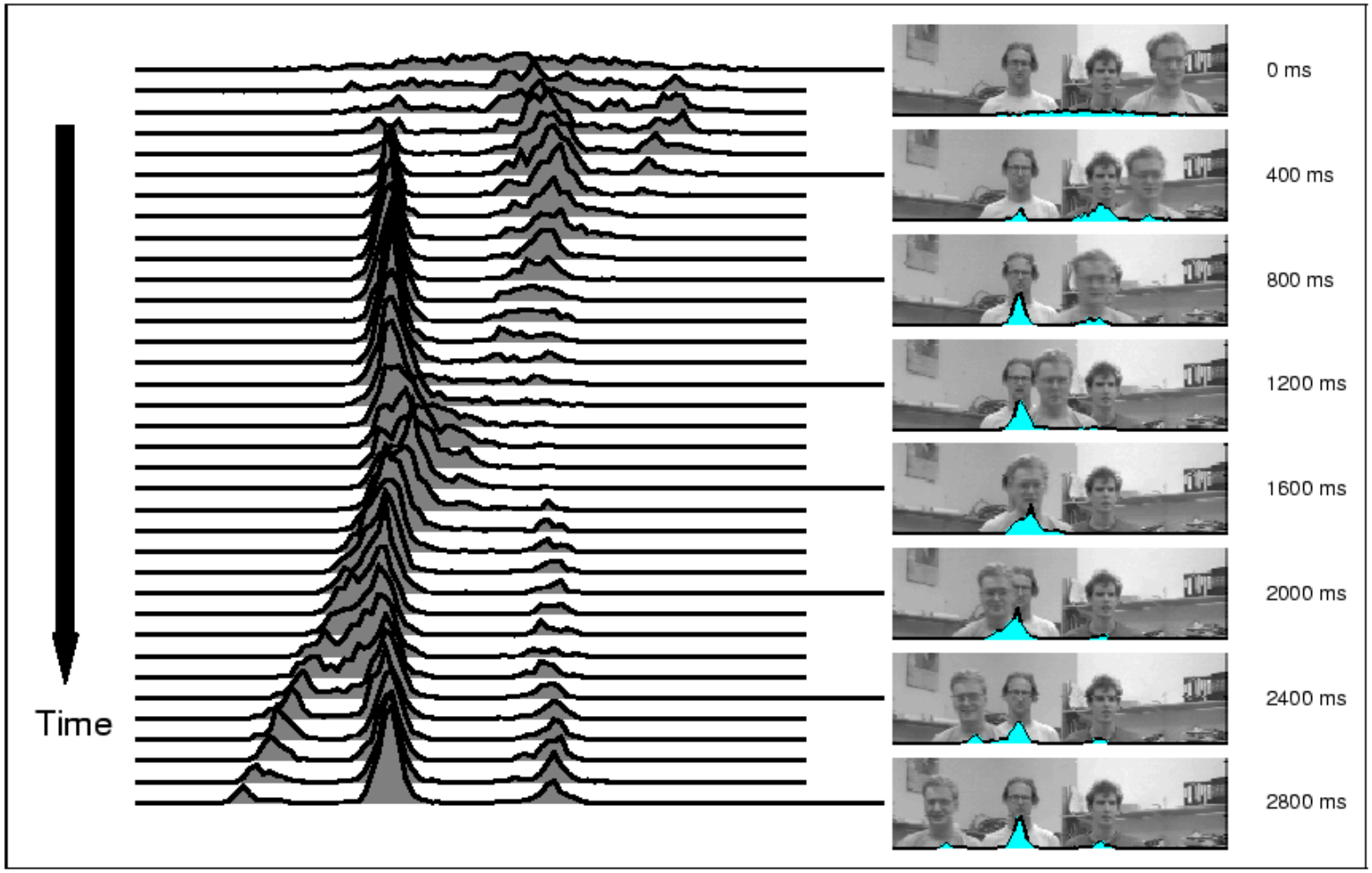
Arrive at corrected density estimate $P(X_t|Y_t)$

M. Isard and A. Blake, CONDENSATION -- conditional density propagation for visual tracking, IJCV 29(1):5-28, 1998

Particle Filtering – Visualization

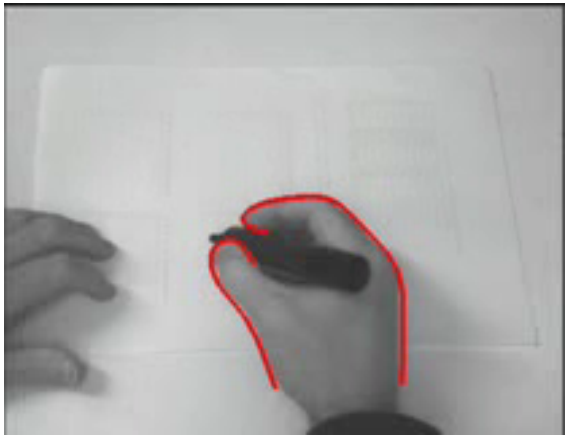
Code and video available from
http://www.robots.ox.ac.uk/~misard/condensation.html

Particle Filtering Results



http://www.robots.ox.ac.uk/~misard/condensation.html

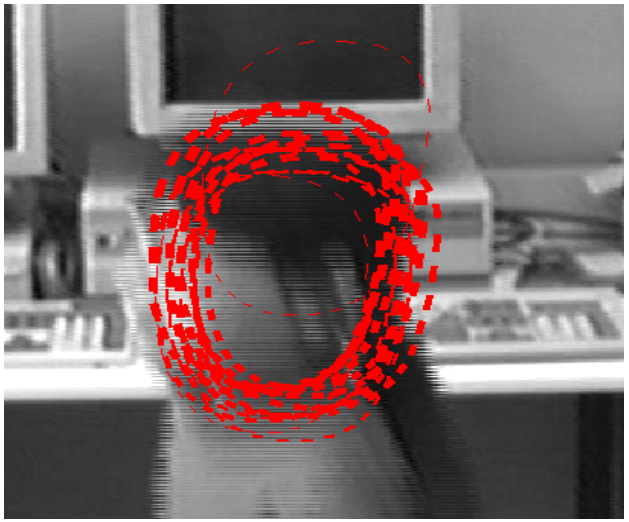
Particle Filtering Results



http://www.robots.ox.ac.uk/~misard/condensation.html

Obtaining a State Estimate

- Note that there's no explicit state estimate maintained—just a "cloud" of particles
- Can obtain an estimate at a particular time by querying the current particle set, e.g. by computing the average



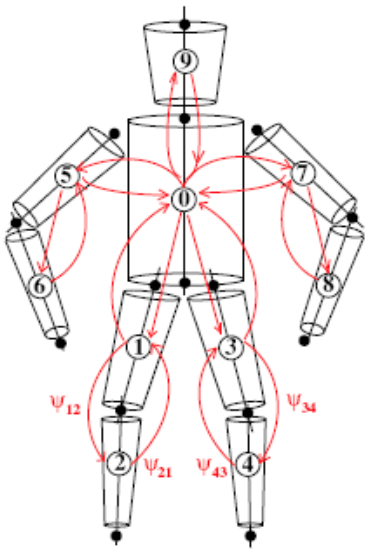
State samples (thickness proportional to weight)

From *Isard & Blake, 1998*

Mean of weighted state samples

Extensions

- Inference on Markov Random Fields
  - Main problem: high-dimensional state space



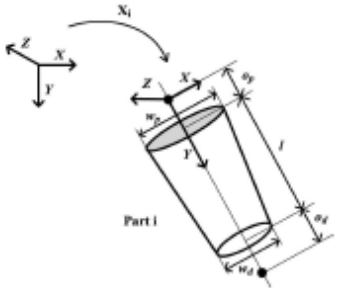


Figure 2: *Parameterization of part i.*

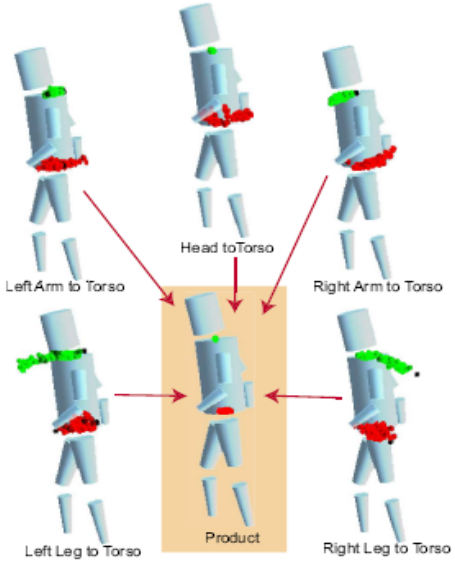
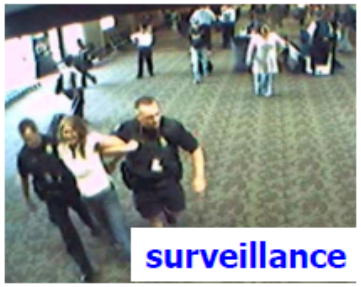


Figure 4: *Message Product.* The head, upper arms, and upper legs send messages to the torso. Samples from these messages are illustrated by showing the predicted torso location with green balls. The distribution over the orientation of the torso is illustrated by showing a red ball at the distal end of the torso for each sample. While any single message respresents uncertain information about the torso pose, the product of these messages tightly constrains the torso position and orientation.

$$m_{i,j}(c_j) = \sum_{c_i} \Phi_i(c_i)\Psi_{i,j}(c_i, c_j) \prod_{k \in \{\mathcal{N}(i) \setminus j\}} m_{k,i}(c_i)$$

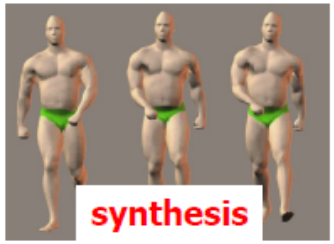
Applications

- Human Motion Analysis


surveillance

search


sport analysis


synthesis

Human Motion Analysis


biomechanics

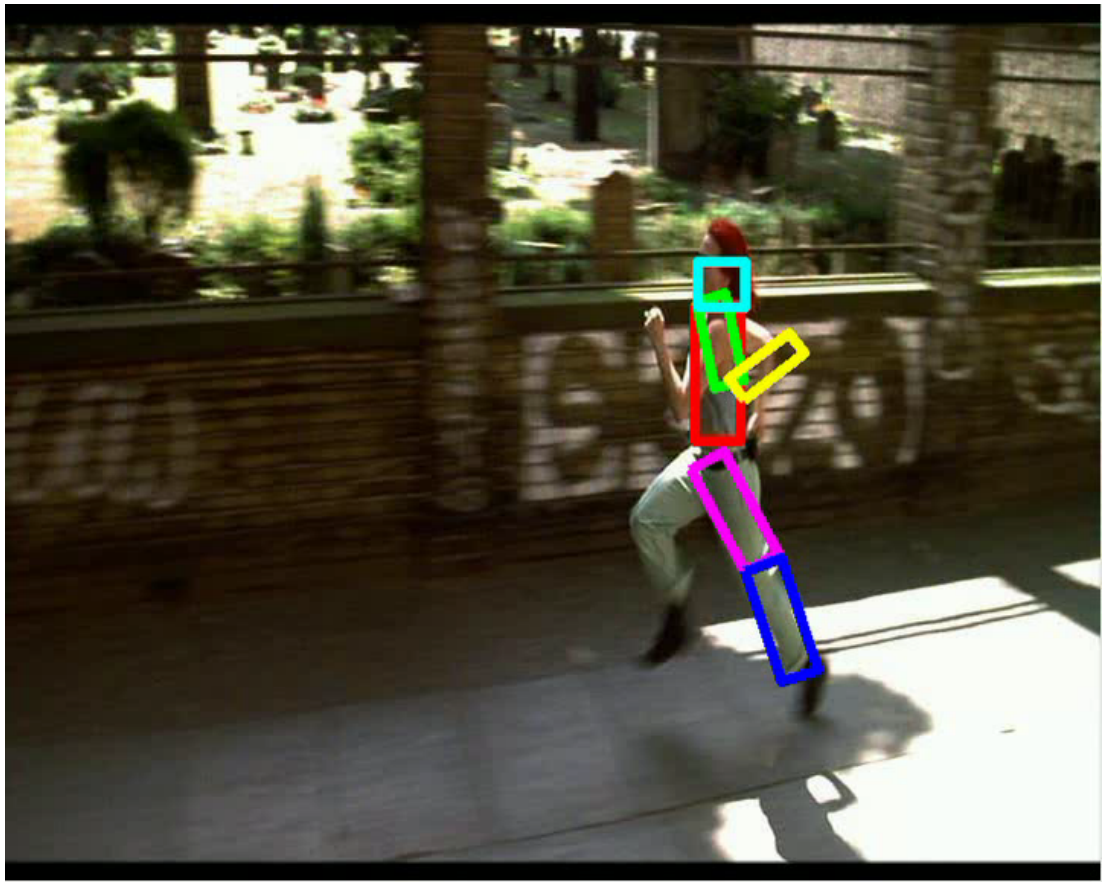

motion capture


game interfaces


sign language

Current state-of-the-art

- `Tracking People by Learning their Appearance' Ramanan, Forsyth & Zisserman, PAMI 2007
  - Tracking-by-detection

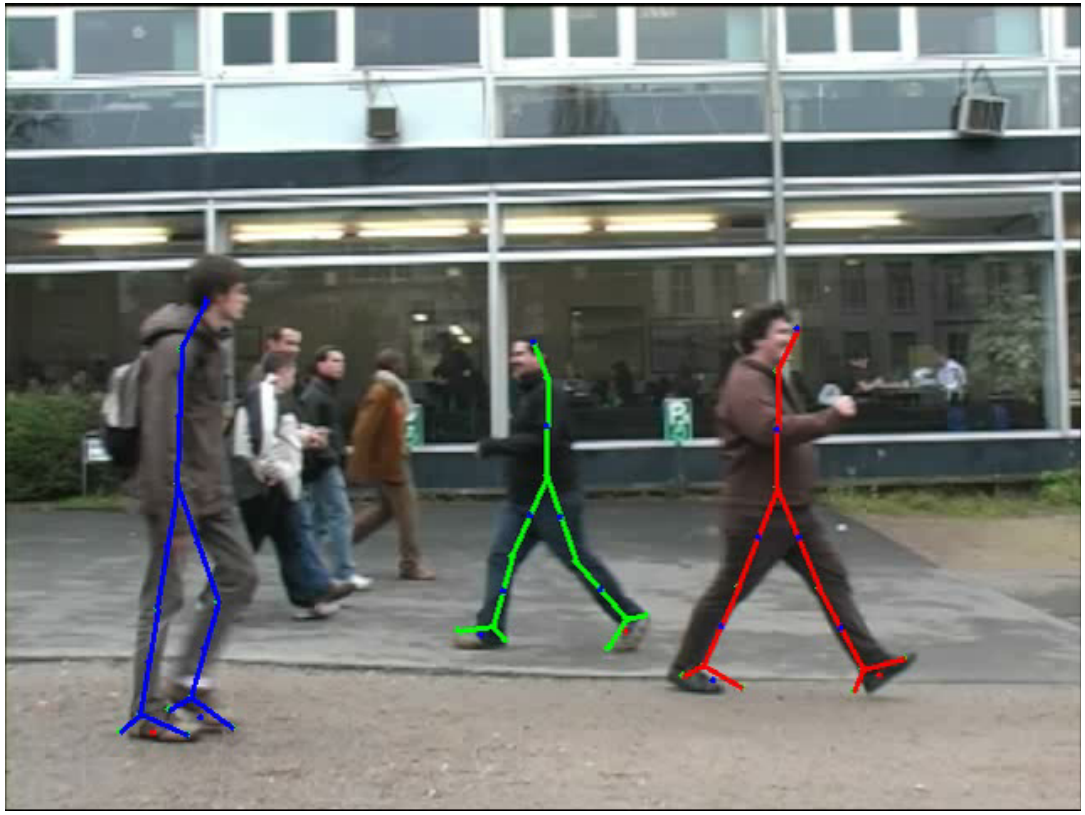


Current state-of-the-art

- `Hierarchical Models for 3D Visual Inference', CVPR 2007, Atul Kanaujia, Cristian Sminchisescu and Dimitris Metaxas
  - 3D pose estimation

Current state-of-the-art

- Multiple people tracking & pose estimation
  - `People tracking by detection and people detection by tracking', CVPR 2008, M. Adriluka, S. Roth and B. Schiele



Current state-of-the-art

- Multiple people tracking & pose estimation