Probabilistic Graphical Models and Their Applications

Bernt Schiele

Max Planck Institute for Informatics

slides adapted from Peter Gehler

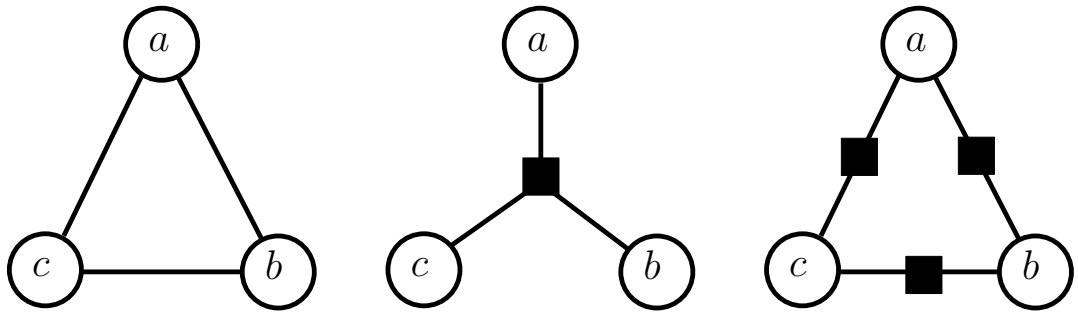
December 2, 2o2o

Today's topics

- Inference
  - Sum-Product (brief recap)
  - Max-Product
- Application
  - Human Pose Estimation

Relationship Potentials to Graphs

- Factor Graphs:



(a) (b) (c)

- Left: Markov Network
- Middle: Factor graph representation of $\phi(a, b, c)$
- Right: Factor graph representation of $\phi(a, b)\phi(b, c)\phi(c, a)$
- Different factor graphs can have the same Markov network (b,c)⇒(a)

Inference in Trees

Inference - what to infer?

- Given distribution

$$p(x) = p(x_1, \ldots, x_n) \tag{1}$$

- Inference: computing functions of the distribution, e.g.
  - mean
  - mode
  - marginal
  - conditionals

What to infer?

- Mean

$$\mathbb{E}_{p(x)}[x] = \sum_{x \in \mathcal{X}} x p(x)$$

- Mode (most likely state)

$$x^* = \operatorname*{argmax}_{x \in \mathcal{X}} p(x)$$

- Conditional Distributions

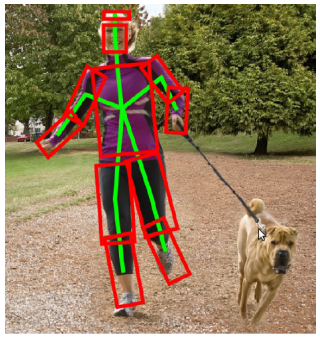
$$p(x_i, x_j \mid x_k, x_l) \quad \text{or} \quad p(x_i \mid x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$$

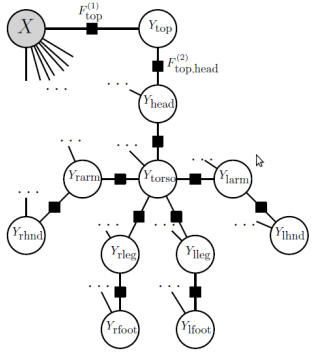
- Max-Marginals

$$x_i^* = \operatorname*{argmax}_{x_i \in \mathcal{X}_i} p(x_i) = \operatorname*{argmax}_{x_i \in \mathcal{X}_i} \sum_{(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)} p(x)$$

Example: Pictorial Structures for Human Pose Estimation

- Find body parts (i.e. find 2D locations and orientations of head, torso, lower/upper left/right arms/legs)





- inference for human body pose estimation:
  - calculating marginals (sum-product algorithm):

$$\operatorname*{argmax}_{x_i \in \mathcal{X}_i} p(x_i) = \operatorname*{argmax}_{x_i \in \mathcal{X}_i} \sum_{(x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_n)} p(x)$$

  - calculating mode (max-product algorithm):

$$x^* = \operatorname*{argmax}_{x \in \mathcal{X}} p(x)$$

Be careful: max marginals not the same as max mode

- The most likely state (max mode)

$$x^* = \underset{x_1,\ldots,x_n}{\operatorname{argmax}} p(x_1, \ldots, x_n) \tag{2}$$

  does not need to be the one for which the marginals are maximized:

- For all $i = 1, \ldots, n$

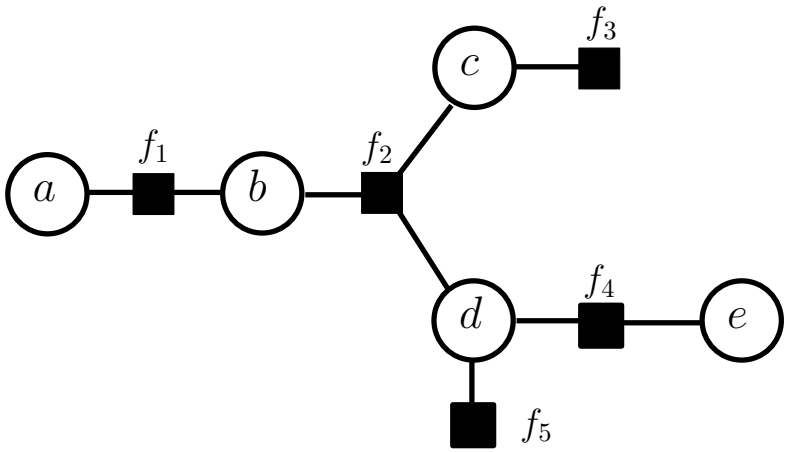
$$x_i^* = \underset{x_i}{\operatorname{argmax}} p(x_i) \tag{3}$$

- Example:

	$x_1 = 0$	$x_1 = 1$
$x_2 = 0$	0.3	0.4
$x_2 = 1$	0.3	0.0
marginal $p(x_1)$	0.6	0.4

General singly-connected factor graphs – 1

- Consider a branching graph:



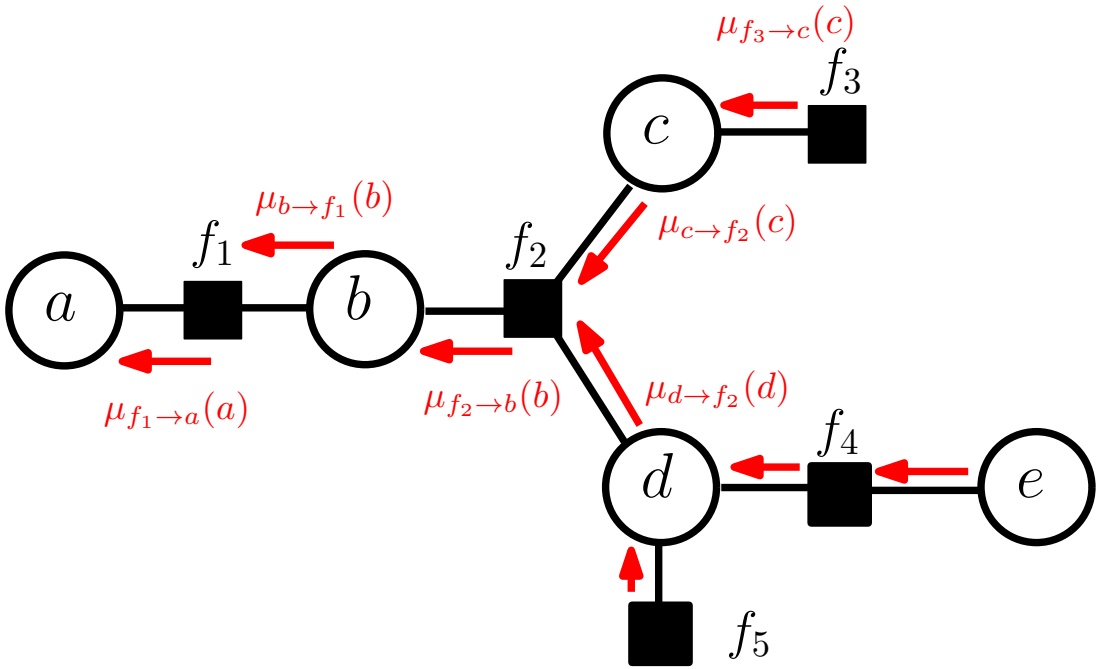
with factors

$$f_1(a,b)f_2(b,c,d)f_3(c)f_4(d,e)f_5(d) \tag{4}$$

- For example: find marginal $p(a)$

General singly-connected factor graphs – 2

- Idea: compute messages



Sum-Product Algorithm – Overview

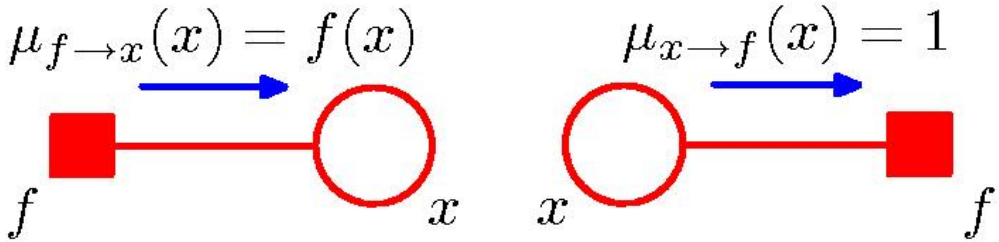
- Algorithm to compute all messages efficiently
- Assuming the graph is singly-connected

1. Initialization
2. Variable to Factor message
3. Factor to Variable message

- Then compute any desired marginals
- Also known as belief propagation

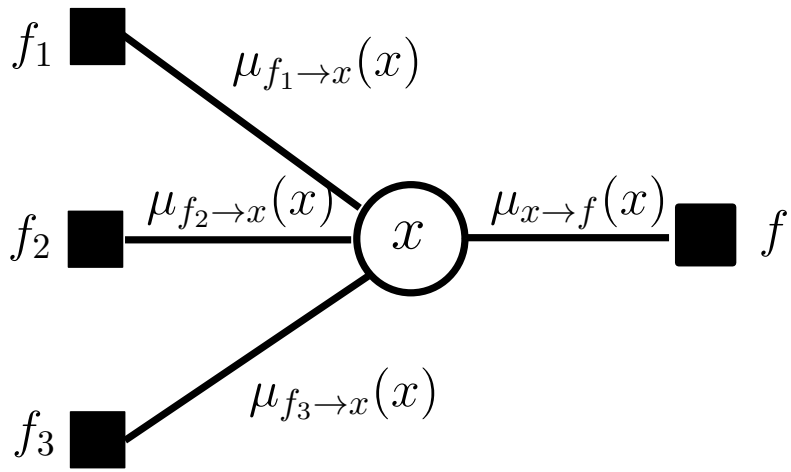
1. Initialization

- Messages from extremal (simplical) node factors are initialized to the factor (left)
- Messages from extremal (simplical) variable nodes are set to unity (right)



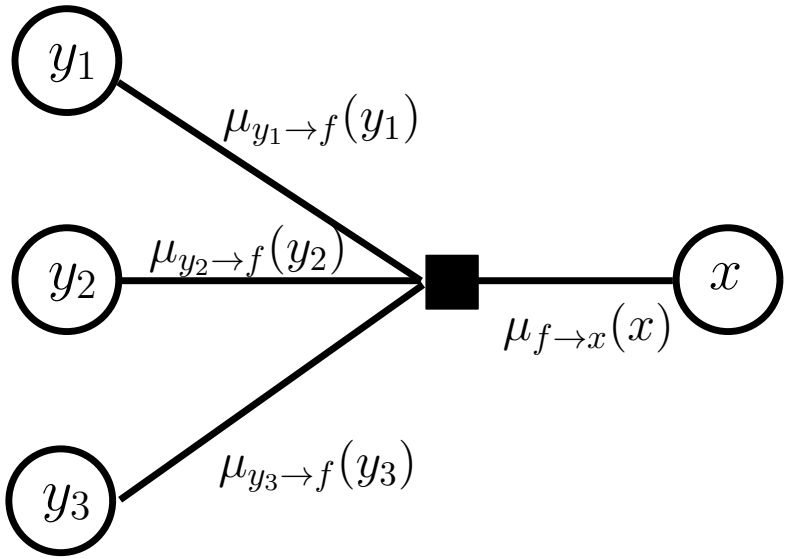
2. Variable to Factor message

$$\mu_{x \to f}(x) = \prod_{g \in \{\mathsf{ne}(x) \setminus f\}} \mu_{g \to x}(x) \tag{5}$$



3. Factor to Variable message

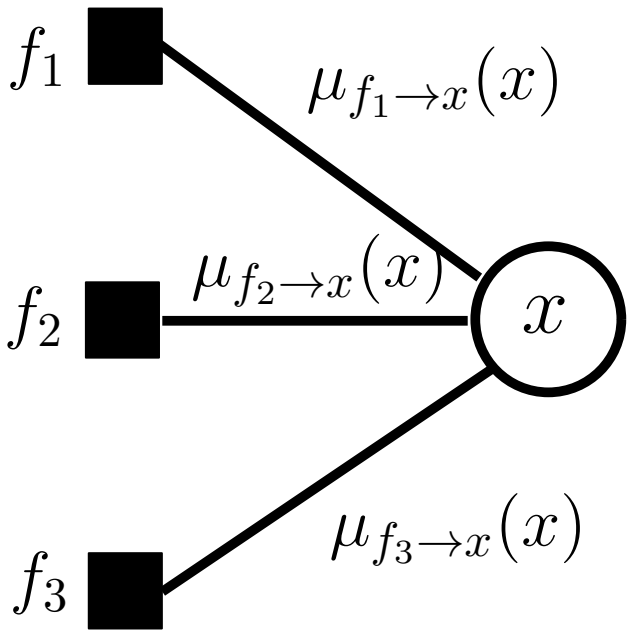
$$\mu_{f\to x}(x) = \sum_{y\in\mathcal{X}_f\setminus x} \phi_f(\mathcal{X}_f) \prod_{y\in\{\mathsf{ne}(f)\setminus x\}} \mu_{y\to f}(y) \tag{6}$$



- We sum over all states in the set of variables
- This explains the name for the algorithm (sum-product)

Marginal

$$p(x) \propto \prod_{f \in \mathsf{ne}(x)} \mu_{f \to x}(x) \tag{7}$$



So far..

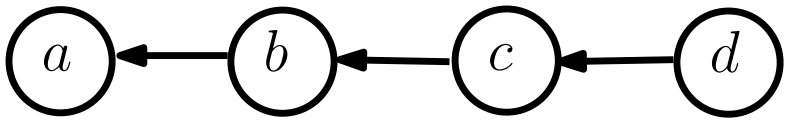
- So far marginals
- Now: finding the maximal state (mode)

Finding the maximal state: Max-Product

- For a given distribution $p(x)$ find the most likely state:

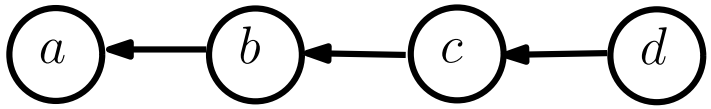
$$x^* = \operatorname*{argmax}_{x_1,\ldots,x_n} p(x_1, \ldots, x_n) \tag{8}$$

- Again use factorization structure to distribute the maximisation to local computations
- Example: chain



$$f(a, b, c, d) = \phi(a, b)\phi(b, c)\phi(c, d) \tag{9}$$

Example: Chain



$$
\begin{aligned}
\max_x f(x) &= \max_{x_1,x_2,x_3,x_4} \phi(x_1,x_2)\phi(x_2,x_3)\phi(x_3,x_4) \\
&= \max_{x_1,x_2,x_3} \phi(x_1,x_2)\phi(x_2,x_3) \underbrace{\max_{x_4} \phi(x_3,x_4)}_{\gamma(x_3)} \\
&= \max_{x_1,x_2} \phi(x_1,x_2) \underbrace{\max_{x_3} \phi(x_2,x_3)\gamma(x_3)}_{\gamma(x_2)} \\
&= \max_{x_1} \underbrace{\max_{x_2} \phi(x_1,x_2)\gamma(x_2)}_{\gamma(x_1)} \\
&= \max_{x_1} \gamma(x_1)
\end{aligned}
$$

Example: Chain

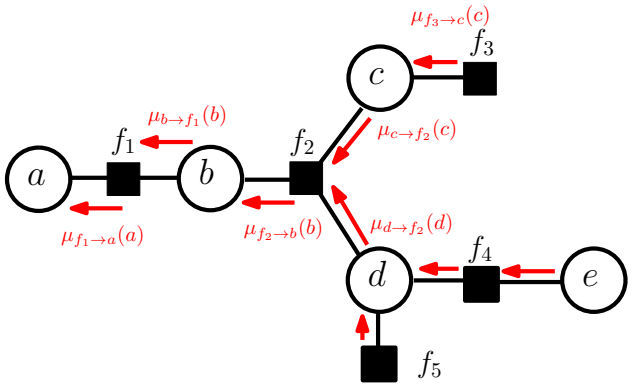
- Once computed the messages ($\gamma(\cdot)$) find the optimal values

$$
\begin{aligned}
x_1^* &= \operatorname*{argmax}_{x_1} \gamma(x_1) \\
x_2^* &= \operatorname*{argmax}_{x_2} \phi(x_1^*, x_2)\gamma(x_2) \\
x_3^* &= \operatorname*{argmax}_{x_3} \phi(x_2^*, x_3)\gamma(x_3) \\
x_4^* &= \operatorname*{argmax}_{x_4} \phi(x_3^*, x_4)\gamma(x_4)
\end{aligned}
$$

- this is called backtracking (an application of dynamic programming)

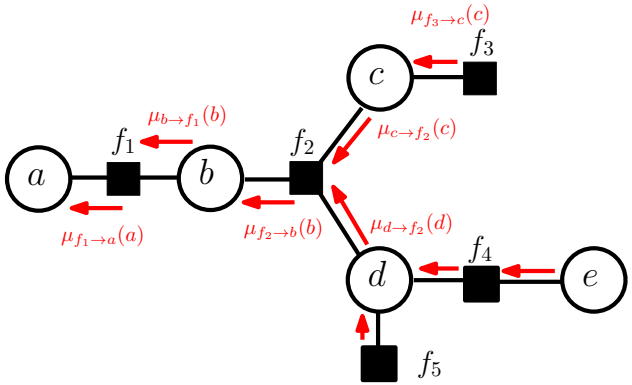
Trees

- Spot the messages:



$$\begin{aligned} \max_x f(x) &= \max_{a,b,c,d,e} f_1(a,b) f_2(b,c,d) f_3(c) f_4(d,e) f_5(d) \\ &= ? \end{aligned}$$

Trees



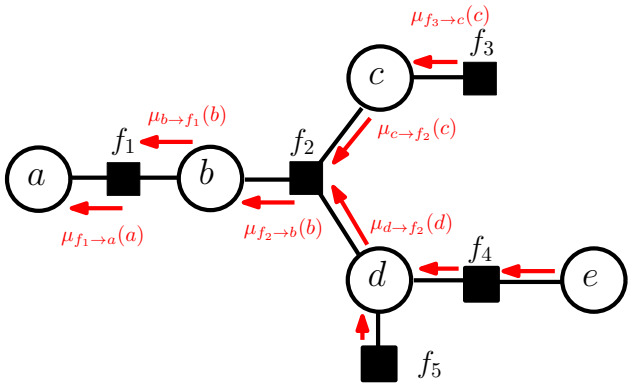
$$\max_x f(x) = \max_{a,b,c,d,e} f_1(a,b) f_2(b,c,d) f_3(c) f_4(d,e) f_5(d)$$

$$= \max_a \max_b f_1(a,b) \max_{c,d} f_2(b,c,d) f_3(c) \underbrace{f_5(d)}_{\mu_{f_5 \to d}(d)} \underbrace{\max_e f_4(d,e)}_{\mu_{f_4 \to d}(d)}$$

$$= \max_a \max_b f_1(a,b) \max_{c,d} f_2(b,c,d) \underbrace{f_3(c)}_{\mu_{c \to f_2}(c)} \underbrace{\mu_{f_5 \to d}(d) \mu_{f_4 \to d}(d)}_{\mu_{d \to f_2}(d)}$$

$$= \max_a \max_b f_1(a,b) \underbrace{\max_{c,d} f_2(b,c,d) \mu_{c \to f_2}(c) \mu_{d \to f_2}(d)}_{\mu_{f_2 \to b}(b)}$$

Trees



$$
\begin{aligned}
\max_x f(x) &= \max_{a,b,c,d,e} f_1(a,b) f_2(b,c,d) f_3(c) f_4(d,e) f_5(d) \\
&= \max_a \max_b f_1(a,b) \underbrace{\mu_{f_2\to b}(b)}_{\mu_{b\to f_1}(b)} \\
&= \max_a \underbrace{\max_b f_1(a,b)\mu_{b\to f_1}(b)}_{\mu_{f_1\to a}(a)} \\
&= \max_a \mu_{f_1\to a}(a)
\end{aligned}
$$

Max-Product Algorithm

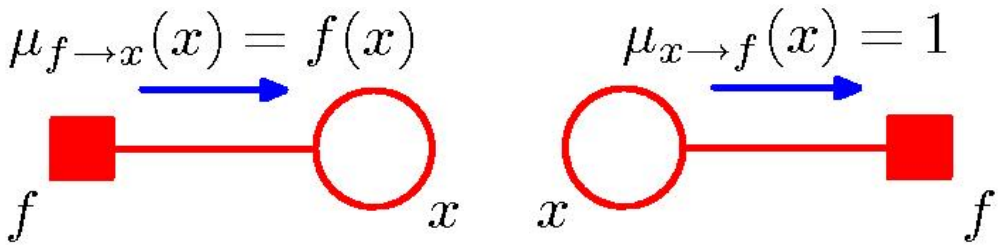
- So we need an algorithm to compute the messages
- Pick any variable as root

1. Initialisation (same as sum-product)
2. Variable to Factor message (same as sum-product)
3. Factor to Variable message

- Then compute the maximal state

1. Initialisation

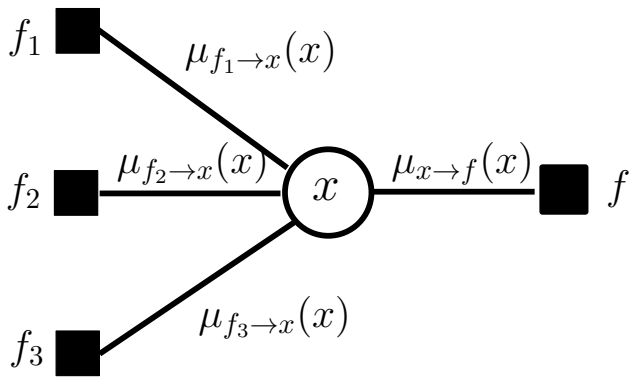
- Messages from extremal node factors are initialized to the factor
- Messages from extremal variable nodes are set to unity



- Same as for sum-product

2. Variable to Factor message

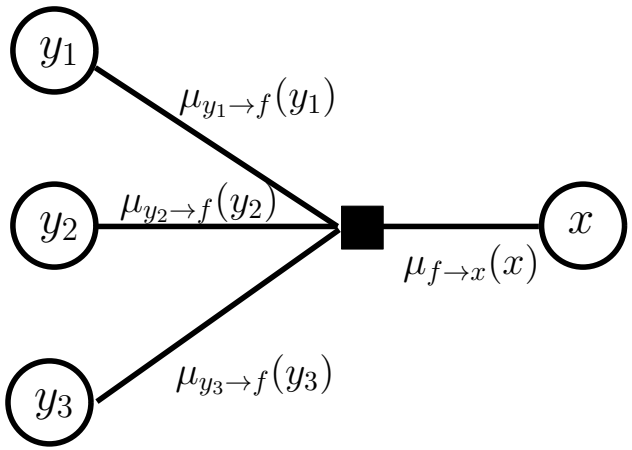
$$\mu_{x \to f}(x) = \prod_{g \in \{\mathsf{ne}(x) \setminus f\}} \mu_{g \to x}(x) \tag{10}$$



- Same as for sum-product

3. Factor to Variable message

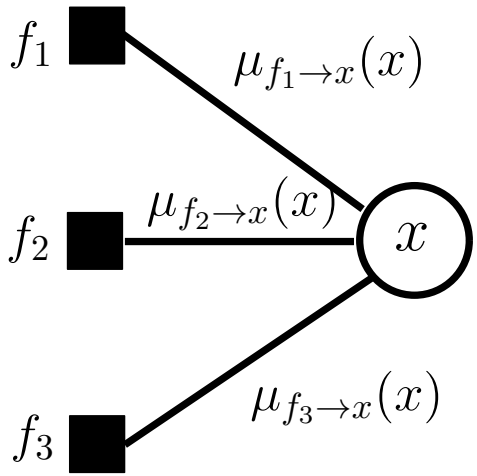
$$\mu_{f\to x}(x) = \max_{y\in\mathcal{X}_f\setminus x} \phi_f(\mathcal{X}_f) \prod_{y\in\{\mathsf{ne}(f)\setminus x\}} \mu_{y\to f}(y) \tag{11}$$



- Different message than in sum-product
- This is now a max-product

Maximal state of Variable

$$x^* = \operatorname*{argmax}_x \prod_{f \in \mathsf{ne}(x)} \mu_{f \to x}(x) \tag{12}$$



Comments

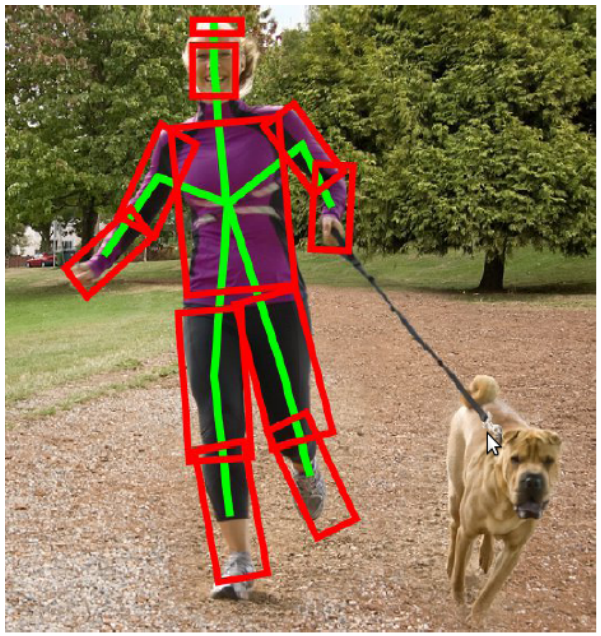
- Products of small probabilities may lead to numerical instabilities
- Take the logarithm

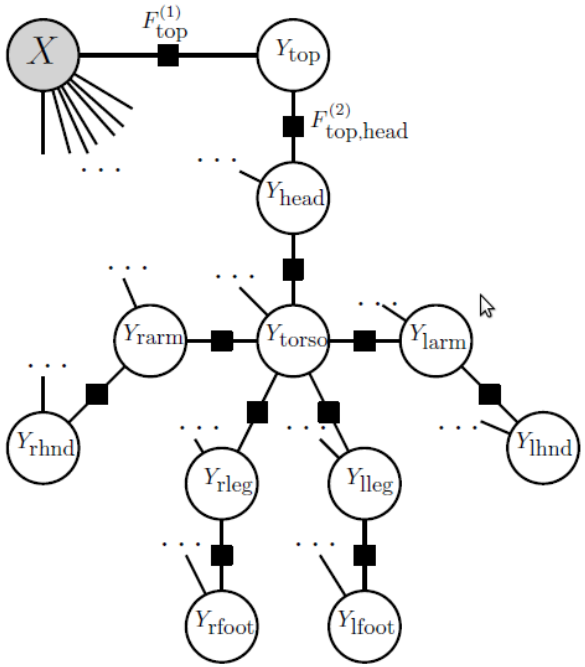
$$\ln\left(\max_x p(x)\right) = \max_x \ln p(x) \tag{13}$$

- Taking the logarithm replaces the products with sums (yields the max-sum algorithm)

Example: Pictorial Structures

- An Application: Human Body Pose Estimation
  - Find Locations of Body Parts

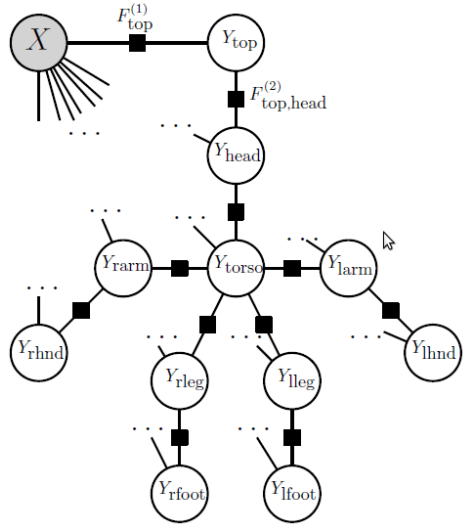




[Fischler& Elschlager, 1973],[Felsenzwalb& Huttenlocher, 2000]

Pictorial Structures

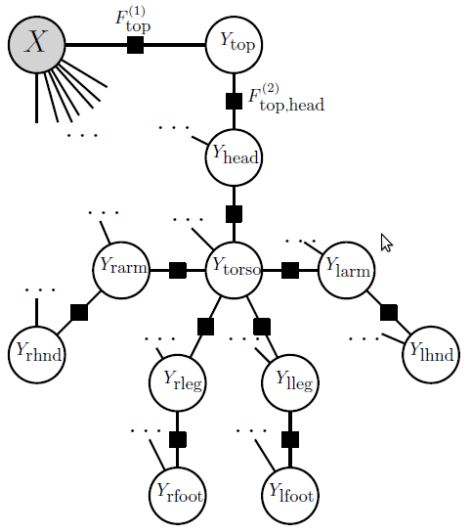
- Each body part one variable (torso,head,etc) (11 total)
- Each variable represented as tupel *e.g.* $y_{torso} = (x, y, s, \theta)$
  - (x, y) image coordinates
  - s scale
  - θ rotation of the part
- Discretize label space y (that is x, y, s, θ) in L states
  - size of L e.g. $L \approx 500,000 = 125 \times 125 \times 4 \times 8 =$ $xpos. \times ypos. \times scales \times orientations$



[Felsenzwalb& Huttenlocher, 2000]

Pictorial Structures

- Potentials:
  $E_{torso}(y_{torso}, X), E_{rarm}(y_{rarm}, X), \ldots,$
- Pairwise potentials:
  $E_{torso,rarm}(y_{torso}, y_{rarm}), \ldots,$
- k be the number of parts (11), L the size of the label space ($\approx 500,000$)
- Given new test image, max-product algorithm complexity is $\mathcal{O}(kL^2)$
- For specific potentials reduction to $\mathcal{O}(kL)$



[Felsenzwalb& Huttenlocher, 2000]

max planck institut
informatik

UNIVERSITÄT
DES
SAARLANDES

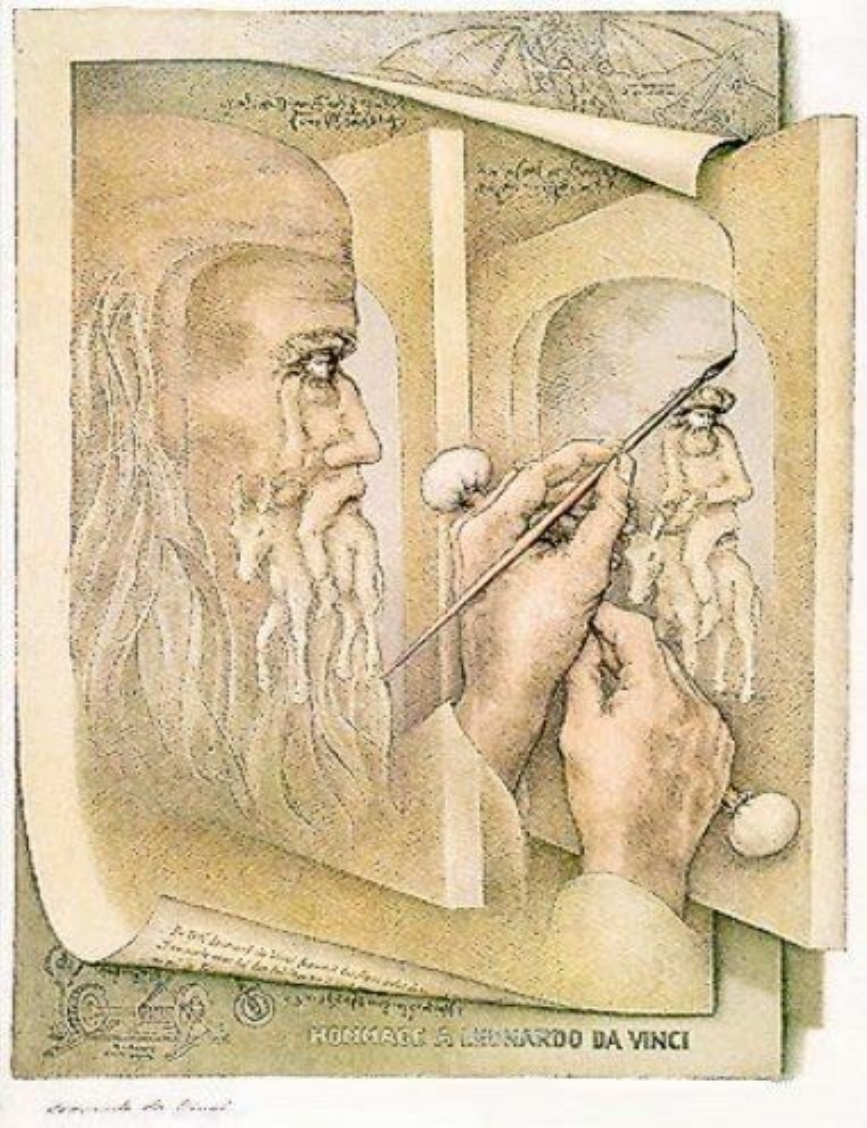
Graphical Models and Their Applications

Human Body Pose Estimation v1.0

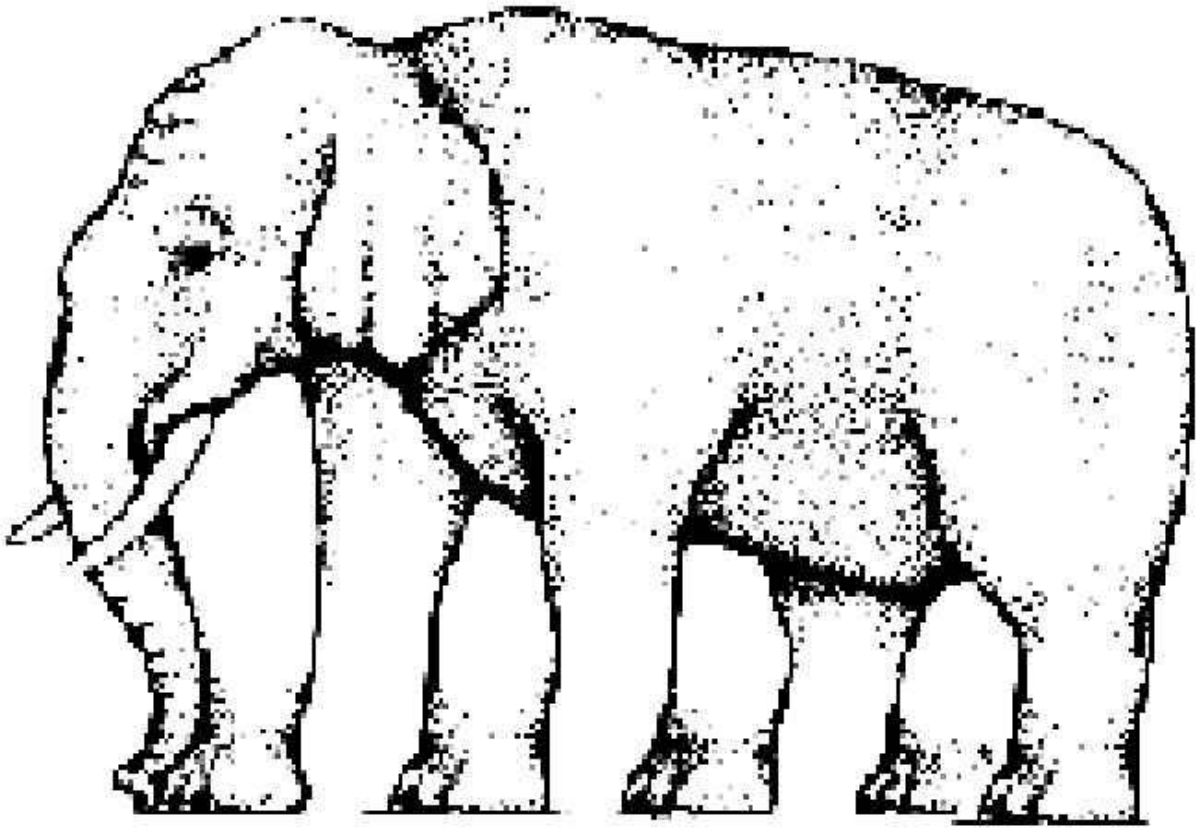
Bernt Schiele

http://www.mpi-inf.mpg.de/gm

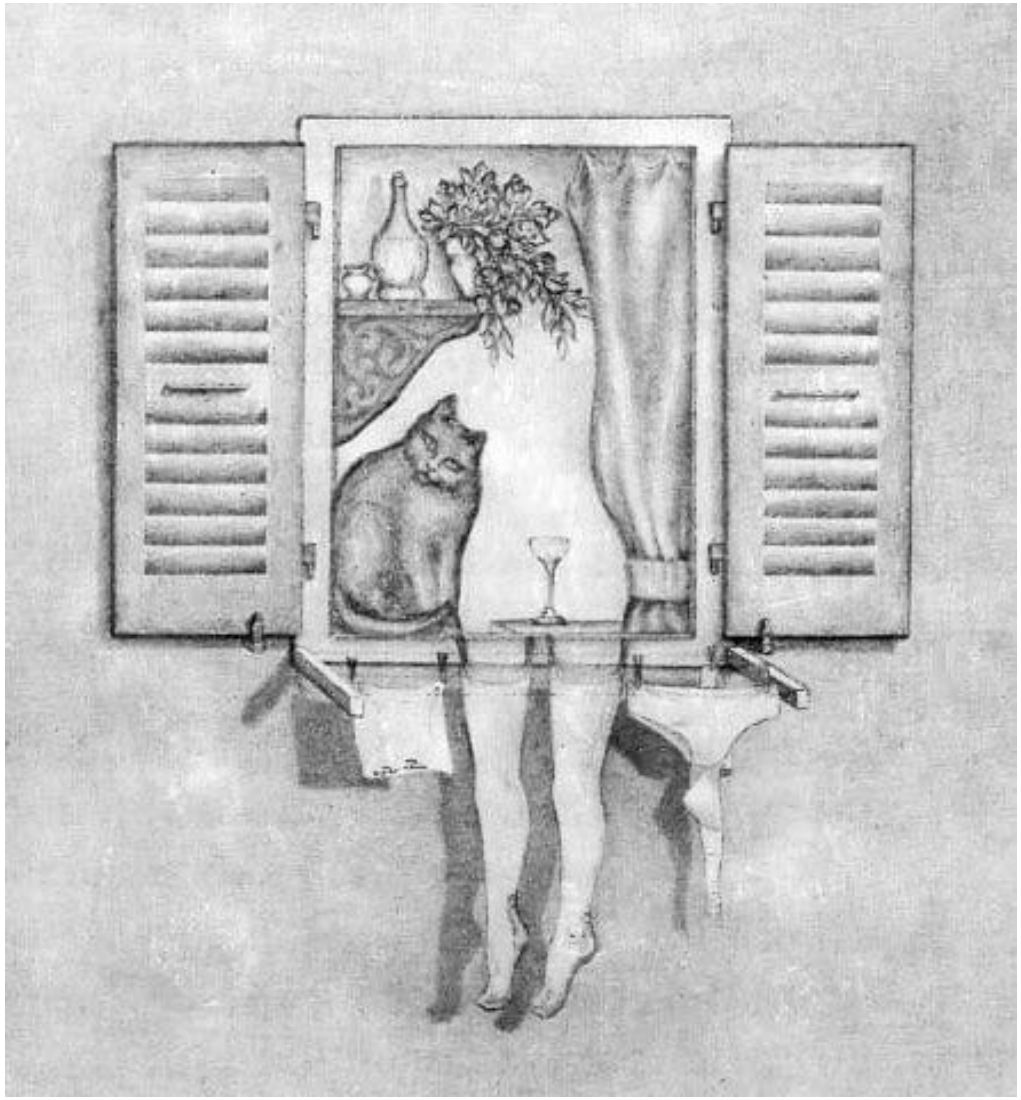
Motivation for Part-Based Models



Motivation for Part-Based Models



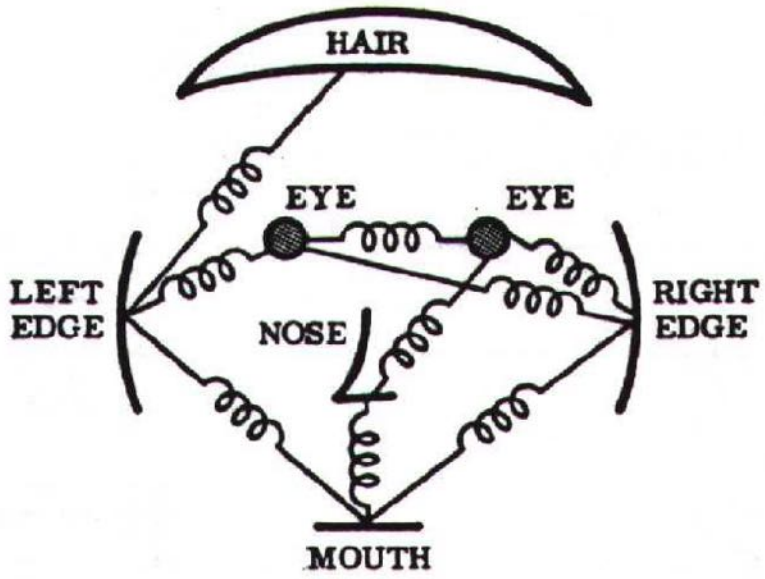
Motivation for Part-Based Models (also applicable to humans)



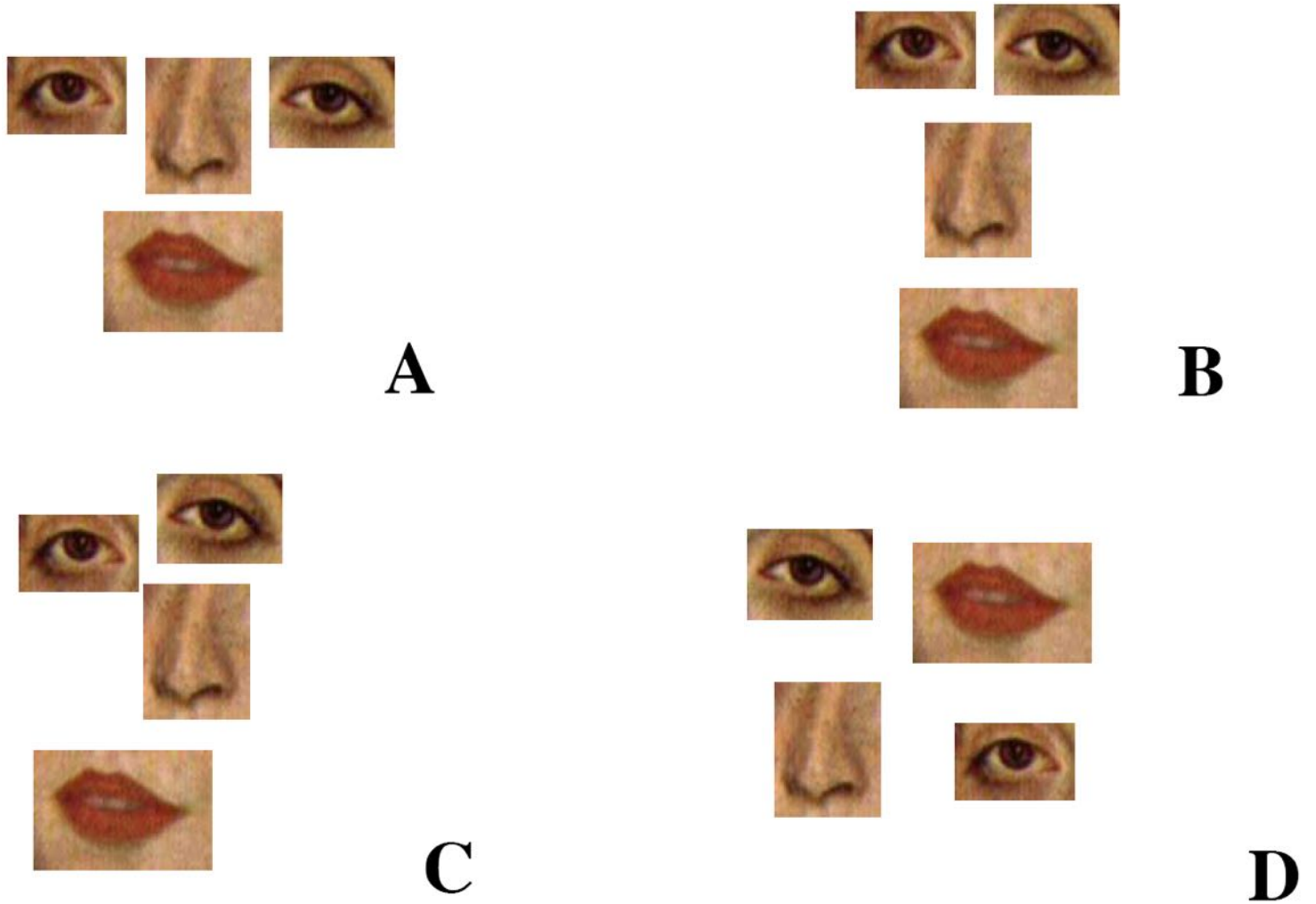
One or Two Faces ?

Part-Based Models

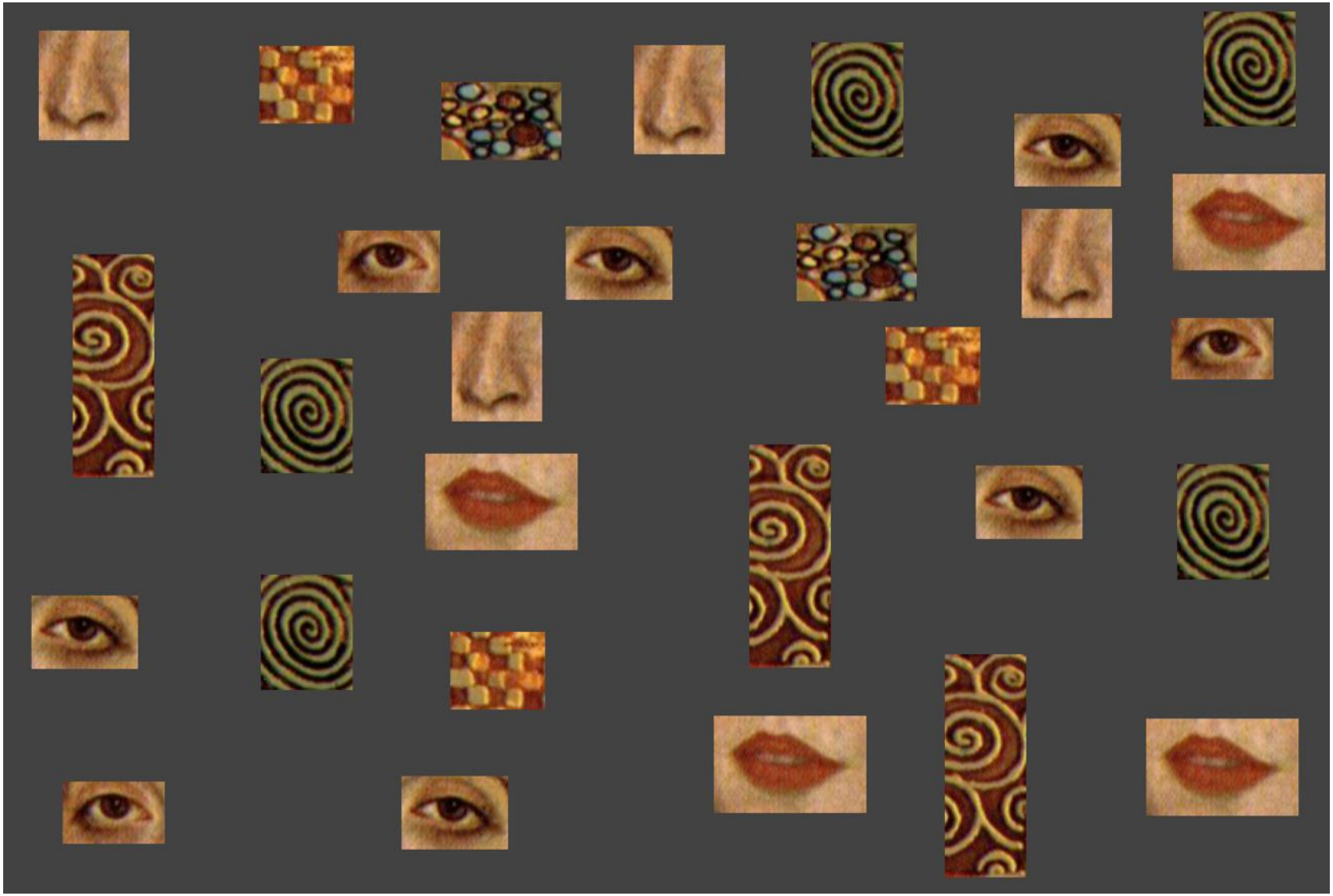
- Pictorial Structures [Fischler & Elschlager 1973]
  - Model has two components
    - parts (2D image fragments)
    - structure (configuration of parts)



The Role of Parts & Structure Deformations [Perona, Caltech]



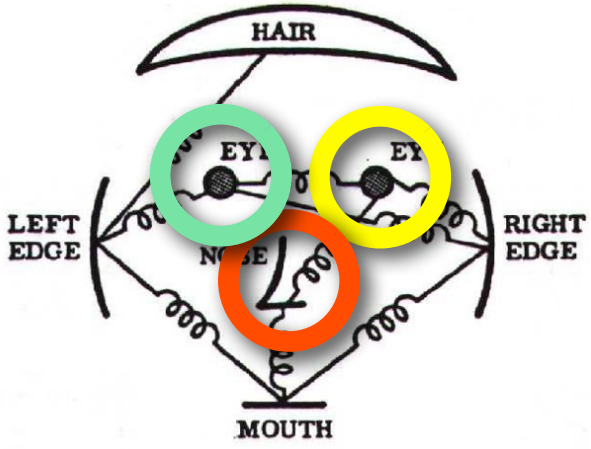
Clutter [Perona, Caltech]

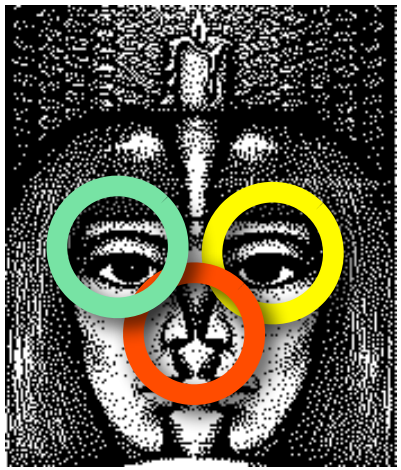


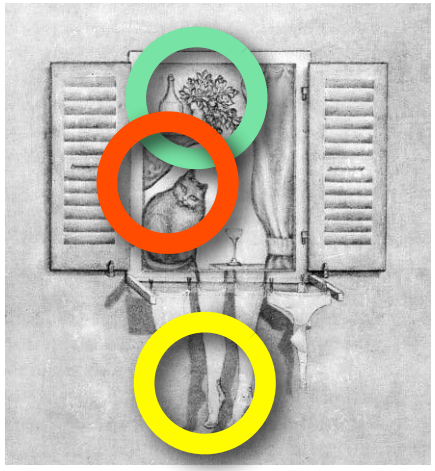
Example

Class of Object Models: Part-Based Models / Pictorial Structures

- Pictorial Structures [Fischler & Elschlager 1973]
  - Model has two components
    - parts (2D image fragments)
    - structure (configuration of parts)

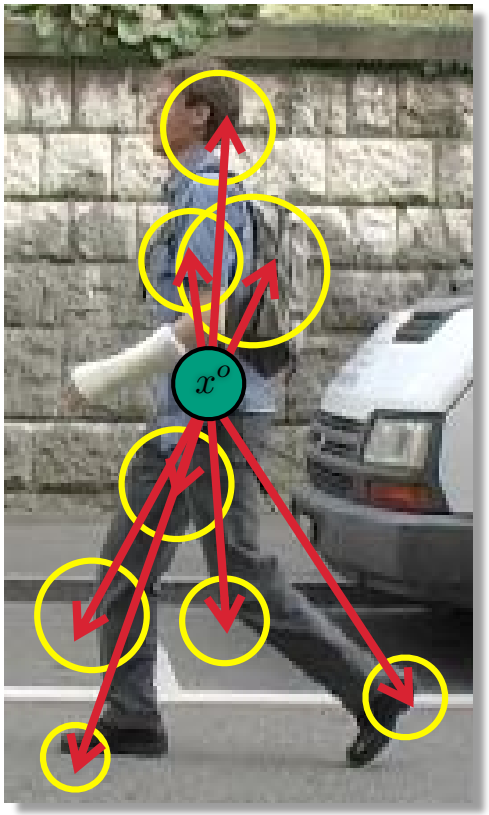






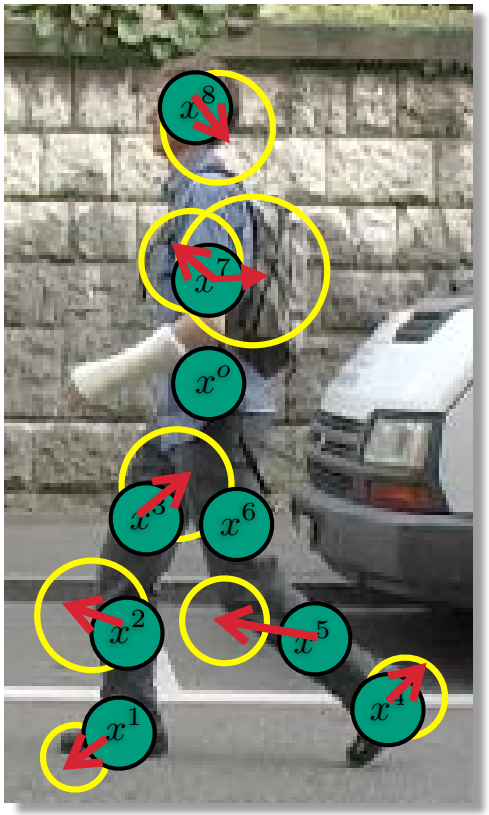
People Detection: partISM

- Appearance of parts:
  Implicit Shape Model (ISM)
  [Leibe, Seemann & Schiele, CVPR 2005]



People Detection: partISM

- Appearance of parts:
  Implicit Shape Model (ISM)
  [Leibe, Seemann & Schiele, CVPR 2005]
- Part decomposition and inference:
  Pictorial structures model
  [Felzenszwalb & Huttenlocher, IJCV 2005]



$$p(L|E) \propto p(E|L)p(L)$$

Body-part positions

Image evidence

Pictorial Structures Model

$$p(L|E) = \frac{p(E|L)p(L)}{p(E)} \propto p(E|L)p(L)$$

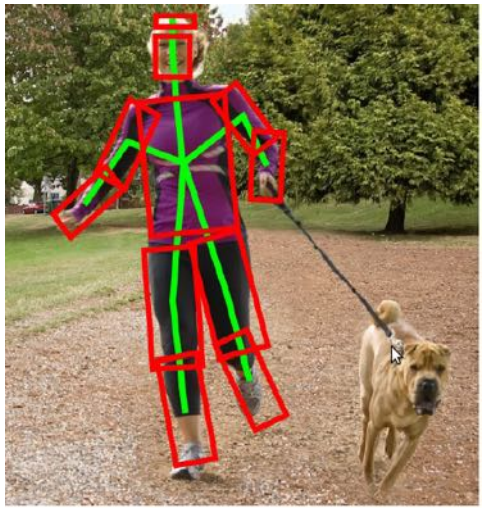
Body-part positions Image evidence

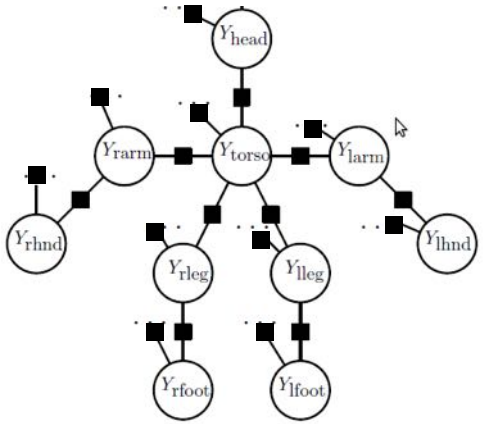
- Two Components
  - Prior (capturing possible part configurations): $p(L)$
  - Likelihood of Parts (capturing part appearance): $p(E|L)$

Human Body Pose Estimation

- well suited for graphical models:
  - prior p(L)
    - models kinematic dependencies between body parts
    - tree-structured prior (constraints between body parts) lead to efficient inference
    - generalized distance transform provide additional efficiency
  - likelihood of body parts p(E|L)
    - models possible appearances of body parts
    - substantial improvements in recent years in appearance modeling and detection

find body parts =
body pose estimation





Pictorial Structures: Model Components

[Andriluka,Roth,Schiele@cvpr09]

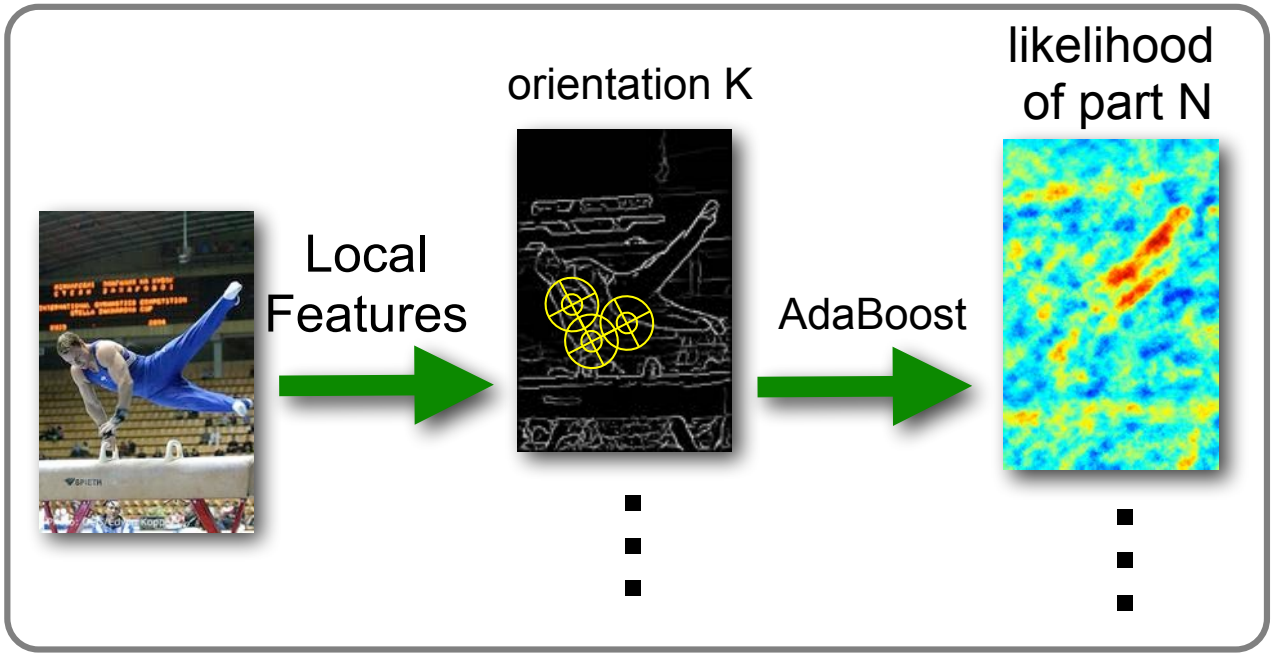
- Body is represented as flexible configuration of body parts

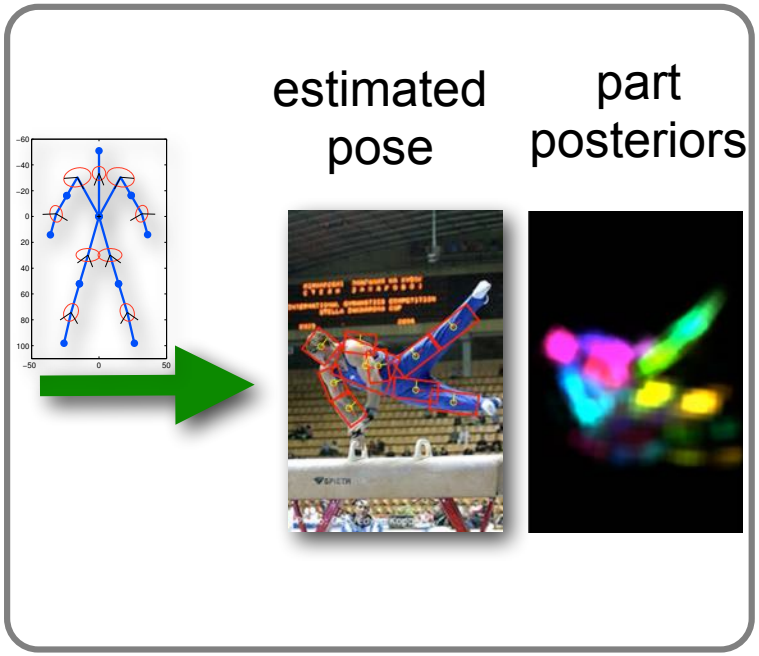
posterior over body poses

$$p(L|E) \propto p(E|L)p(L)$$

likelihood of observations

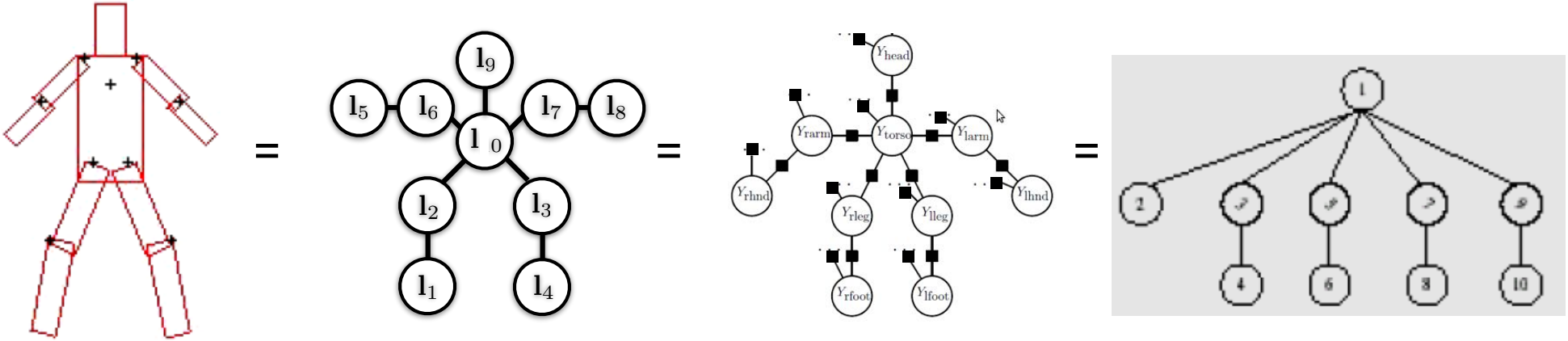
prior on body poses



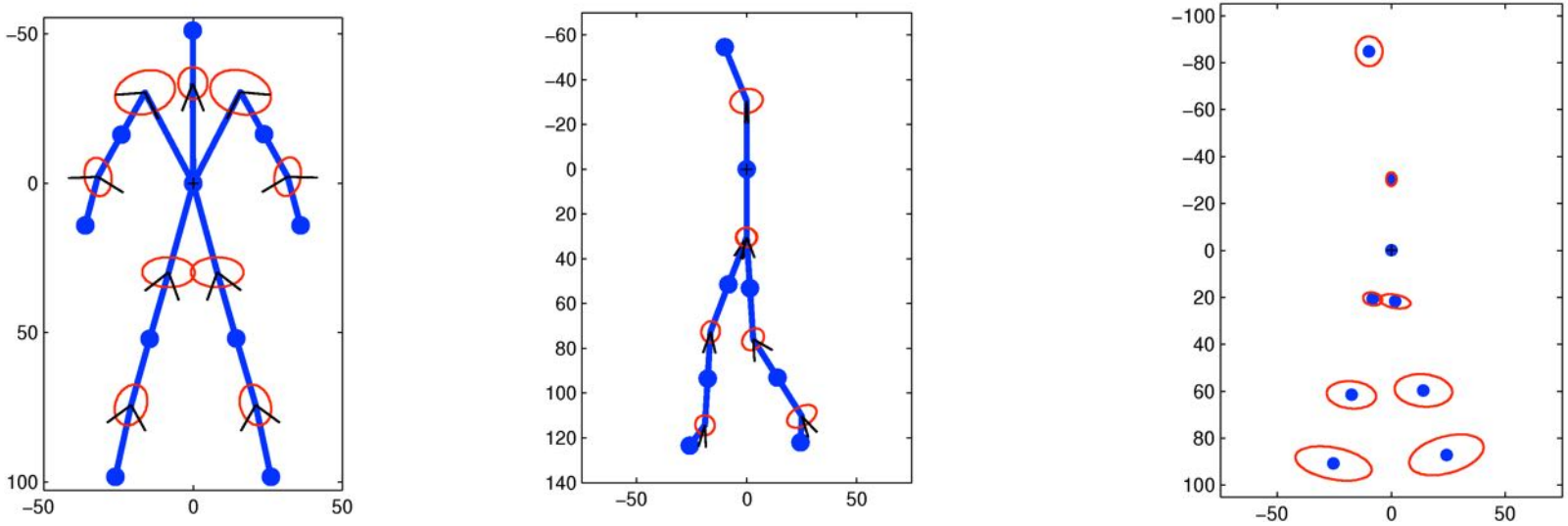


Human Body Pose Models - Prior p(L) (may be called “cardboard models”)

- e.g. Felzenswalb & Huttenlocher - ijcv’05

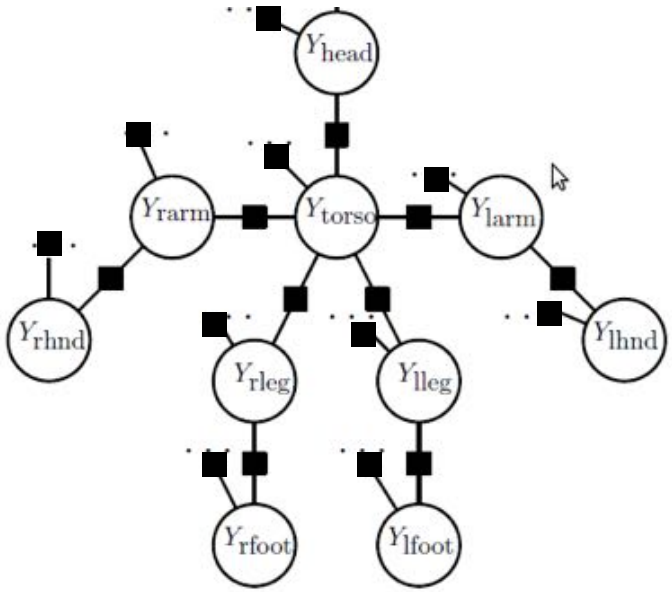


- e.g. Andriluka, Roth & Schiele - cvpr’09,ijcv’11



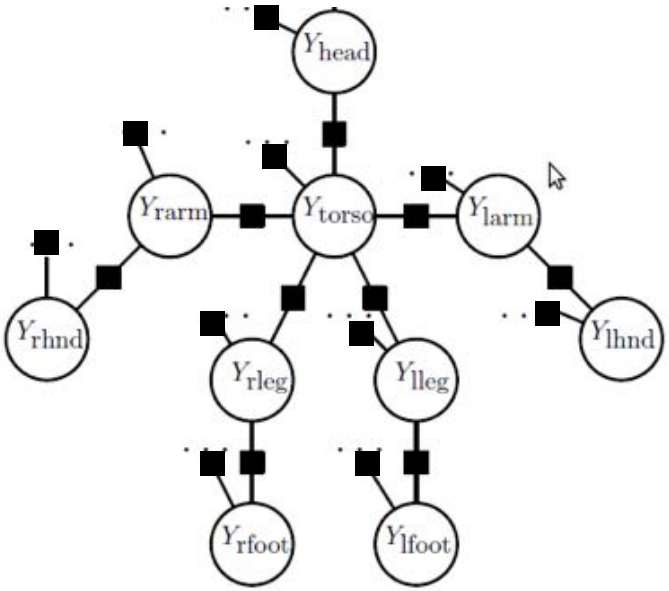
Pictorial Structures

- each body part one variable (torso, head, etc.)
  - before: 11 parts
  - in the following only 10 parts (head, torso, right/left upper/lower arm/leg)
- each variable represented as tuple
  - here: $y_i = l_i = (x_i, y_i, \theta_i, s_i)$
  - with (x,y) image coordinates, s scale, θ rotation of the part
- discretize label space (that is x, y, s, θ) in L states



Pictorial Structures

- potentials (= energies = factors)
  - unaries for each body part (torso, head, ...)
  - pairwise between connected body parts



- body pose estimation
  - max-product algorithm (MAP-estimate, best overall configuration)
  - sum-product algorithm (marginals of each part)

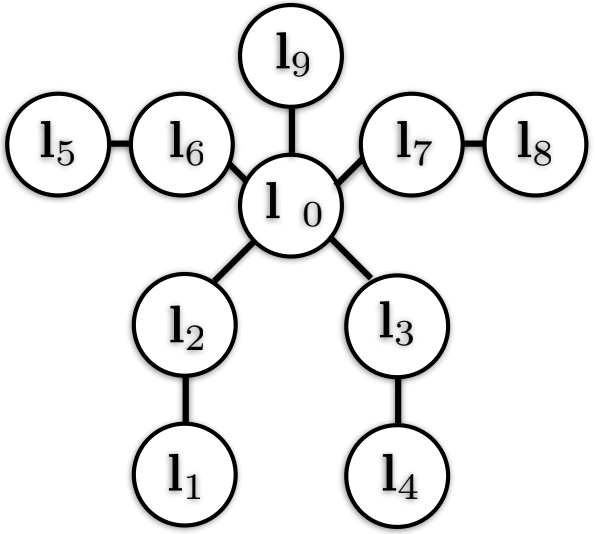
- complexity
  - k be the number of body parts (e.g. k = 10)
  - L the size of the label space (L e.g. several 100k)
  - max-product algorithm in general: $O(k L^2)$
  - for specific pairwise potentials: $O(k L)$

Kinematic Tree Prior

- Notation
  - from [Andriluka,Roth,Schiele@ijcv11]
  - body configuration:

$$L = \{l_0, l_1, \ldots, l_N\}$$

  - each body part: $l_i = (x_i, y_i, \theta_i, s_i)$

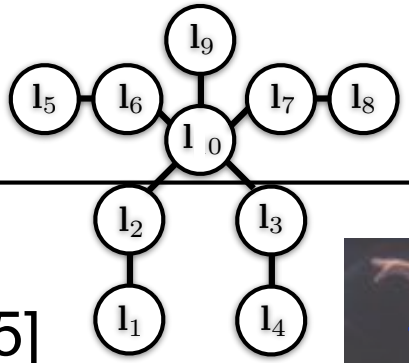


- Prior:

$$p(L) = p(l_0) \prod_{(i,j) \in G} p(l_i | l_j)$$

  - with p(l_0) assumed uniform
  - with p(l_i | l_j) modeled using a Gaussian

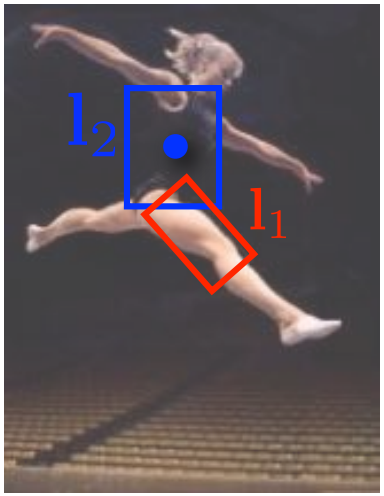
Kinematic Tree Prior

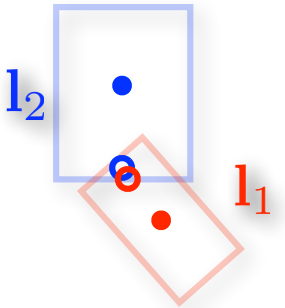


- Represent pairwise part relations [Felzenszwalb & Huttenlocher, IJCV'05]

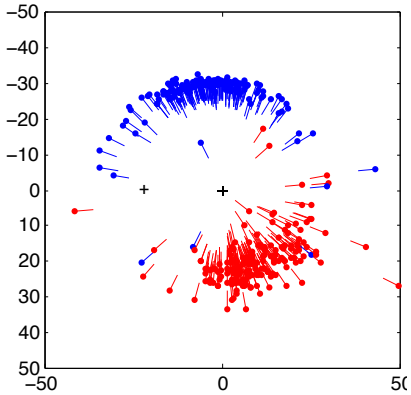
$$p(L) = p(l_0) \prod_{(i,j)\in G} p(l_i|l_j)$$

$$p(l_i|l_j) = \mathcal{N}(T_{ji}(l_i) - T_{ij}(l_j)|\mu^{ji}, \Sigma^{ji})$$

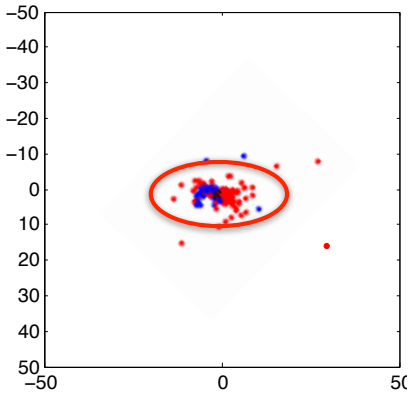




part locations relative to the joint



transformed part locations



Kinematic Tree Prior

- Transformation T

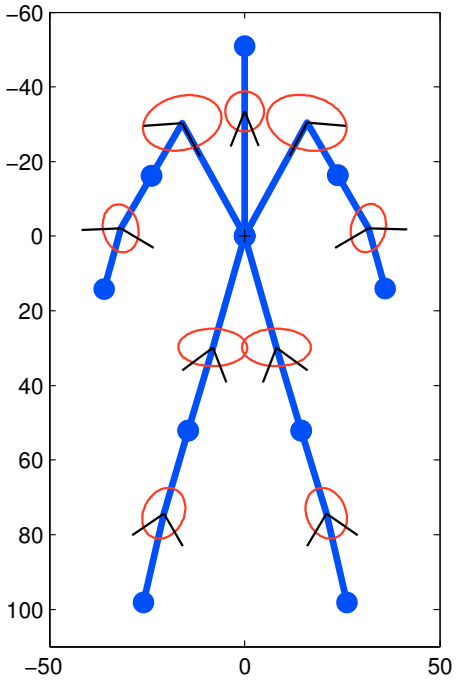
$$T_{ji}(l_i) = \begin{pmatrix} x_i + s_i d_x^{ji} \cos\theta_i - s_i d_y^{ji} \sin\theta_i \\ y_i + s_i d_x^{ji} \sin\theta_i + s_i d_y^{ji} \cos\theta_i \\ \theta_i \\ s_i \end{pmatrix}$$

  - with $l_i = (x_i, y_i, \theta_i, s_i)$
  - and $d^{ji} = \begin{pmatrix} d_x^{ji} \\ d_y^{ji} \end{pmatrix}$ position of the joint between parts i and j, represented in the coordinate system of part i

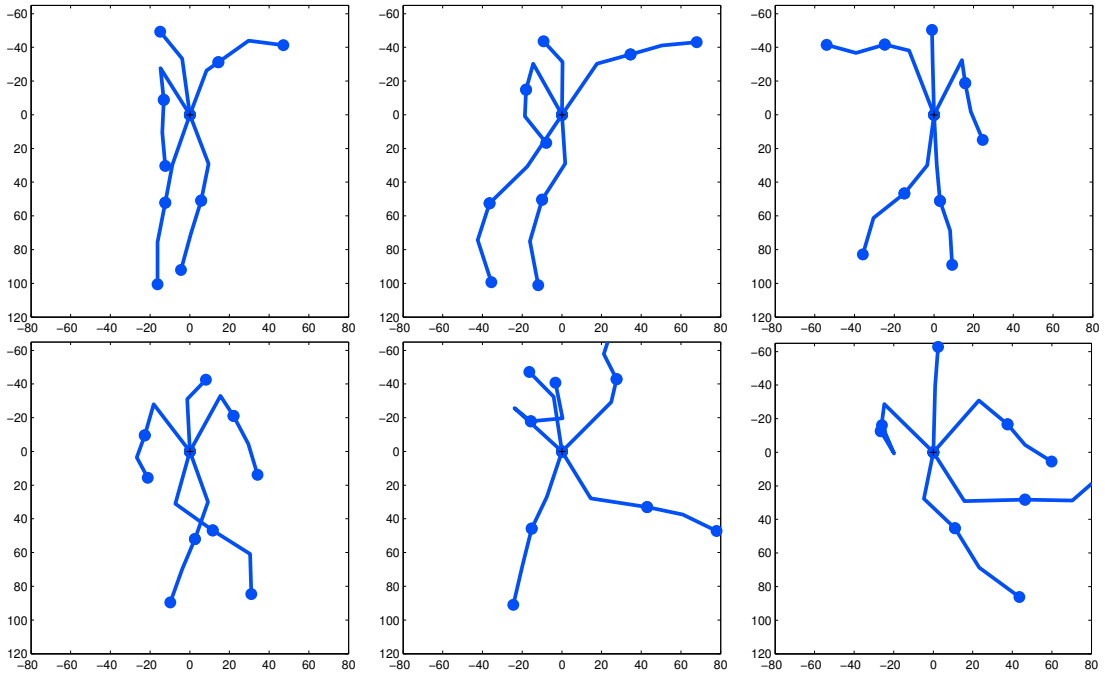
Kinematic Tree Prior

- Prior parameters: $\{T_{ji}, \Sigma^{ji}\}$
- Parameters of the prior are estimated with maximum likelihood

mean pose



several independent samples



Pictorial Structures: Model Components

[Andriluka,Roth,Schiele@cvpr09]

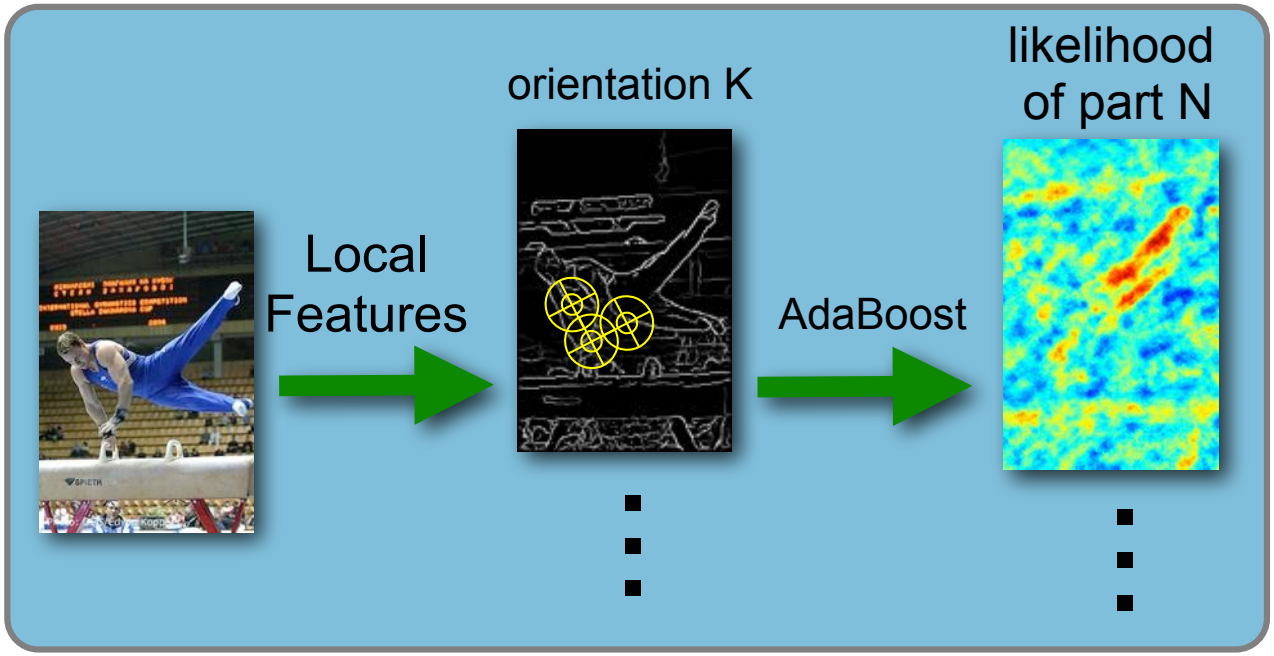
- Body is represented as flexible configuration of body parts

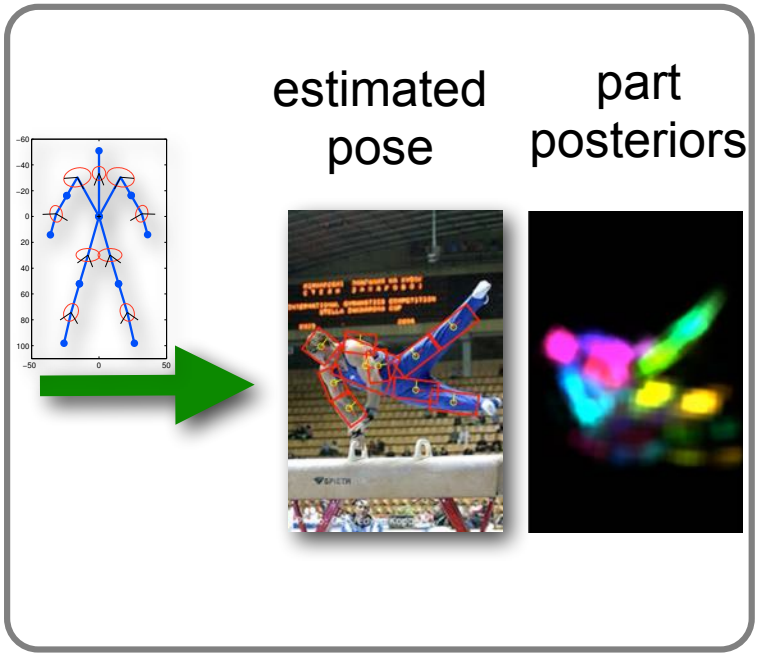
posterior over body poses

$$p(L|E) \propto p(E|L)p(L)$$

likelihood of observations

prior on body poses



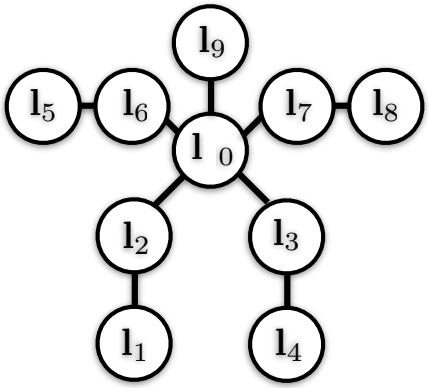


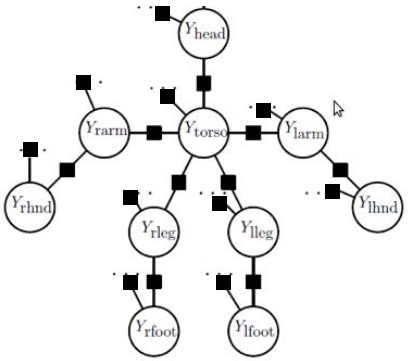
Likelihood Model

- Assumption:
  - evidence (image features) for each part independent of all other parts:

$$p(E|L) = \prod_{i=0}^{N} p(E|l_i)$$

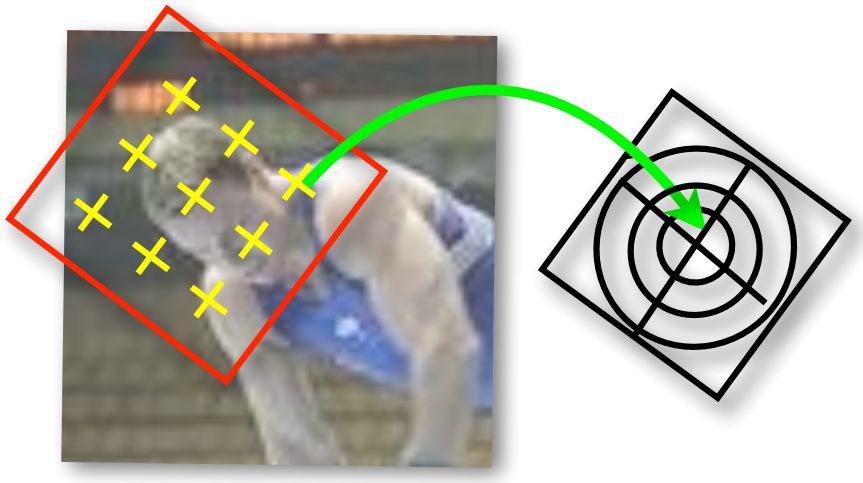
- assumption clearly not correct, but
  - allows efficient computation
  - works rather well in practice
  - training data for different body parts should cover “all” appearances





Likelihood Model

- Build on advances in object detection:
  - powerful image descriptor: Shape Context
    [Belongie et al., PAMI'02; Mikolajczyk&Schmid, PAMI'05]
  - robust feature vector, here histogram of gradients (first derivatives)
  - dense representation
  - discriminative model: AdaBoost classifier for each body part



- Shape Context: 96 dimensions (4 angular, 3 radial, 8 gradient orientations)

- Feature Vector: concatenate the descriptors inside part bounding box
- head: 4032 dimensions
- torso: 8448 dimensions

Likelihood Model

- Part 'likelihood' derived from boosting score:
  - $e_i(l_i)$ = feature of part i at 'location' l_i
  - weak classifiers (e.g. decision stump output): $h_t(e_i(l_i))$
  - strong AdaBoost classifier score: $\sum_t \alpha_{i,t} h_t(e_i(l_i)) \in [-1, +1]$
  - remember $l_i = (x_i, y_i, \theta_i, s_i)$

decision stump weight

decision stump output

$$\tilde{p}(E|l_i) = \max\left(\frac{\sum_t \alpha_{i,t} h_t(e_i(l_i))}{\sum_t \alpha_{i,t}}, \varepsilon_0\right)$$

part location

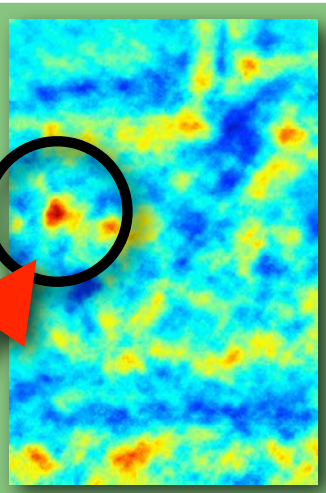
small constant to deal with part occlusions

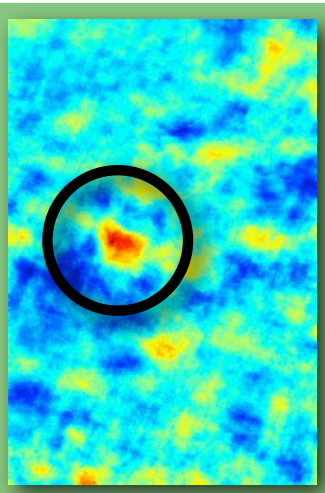
Likelihood Model

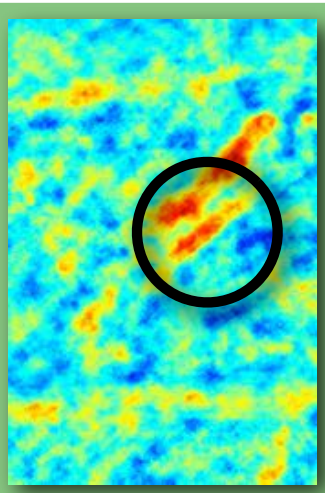
Input image

Head Torso Upper leg

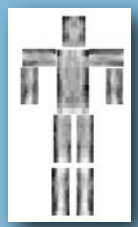
Our part likelihoods

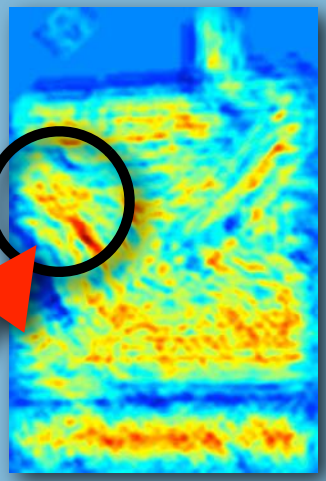


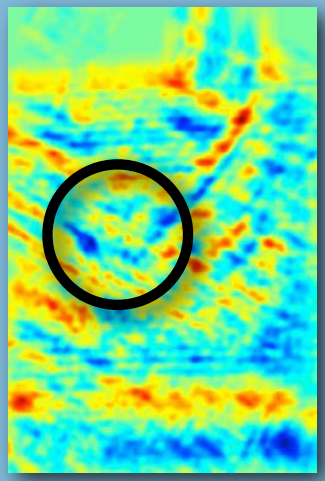


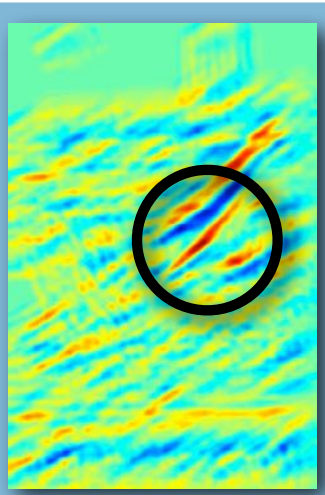


[Ramanan, NIPS'06]







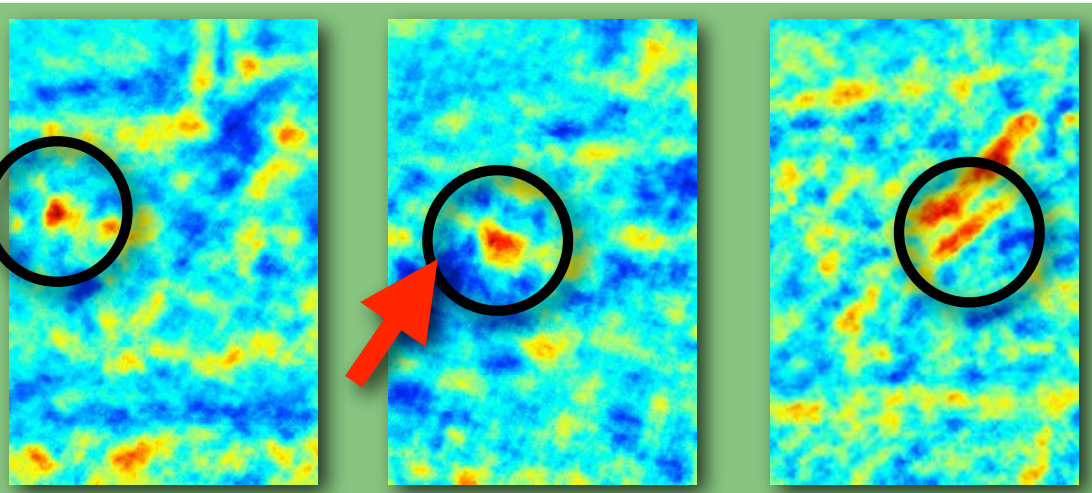


Likelihood Model

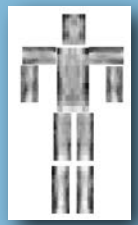
Input image

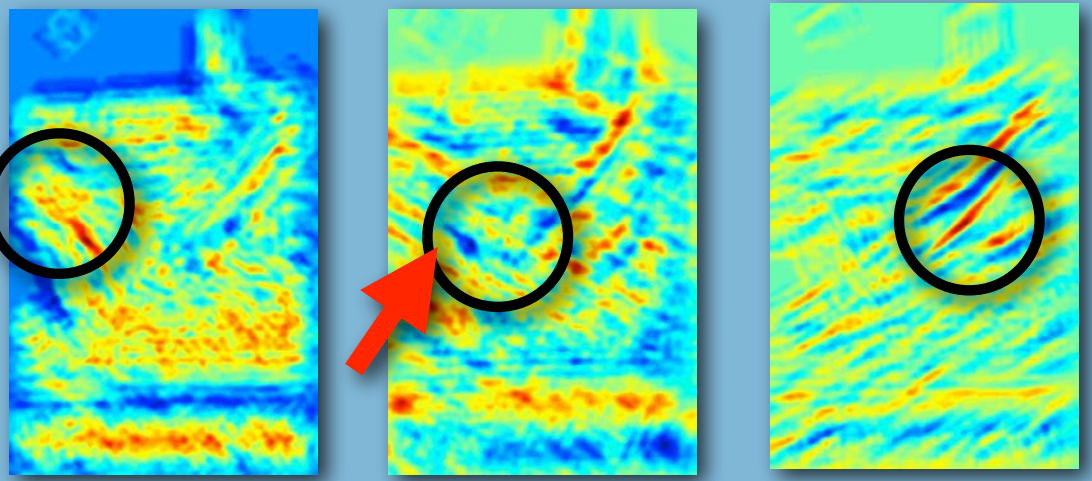
Head Torso Upper leg

Our part likelihoods



[Ramanan, NIPS'06]



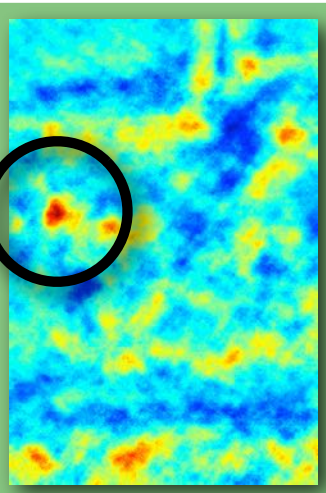


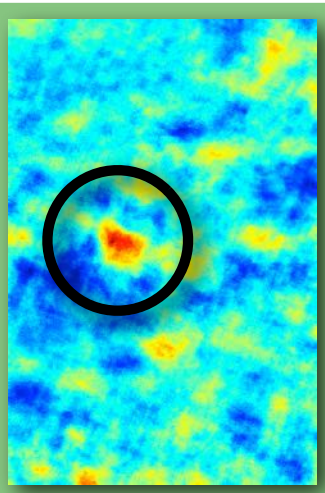
Likelihood Model

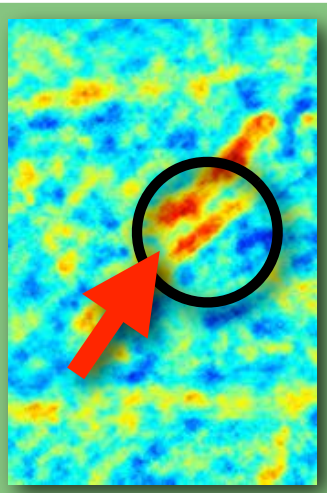
Input image

Head Torso Upper leg

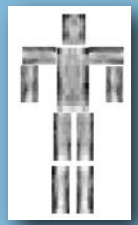
Our part likelihoods

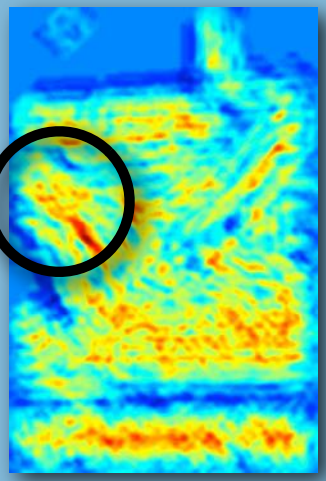


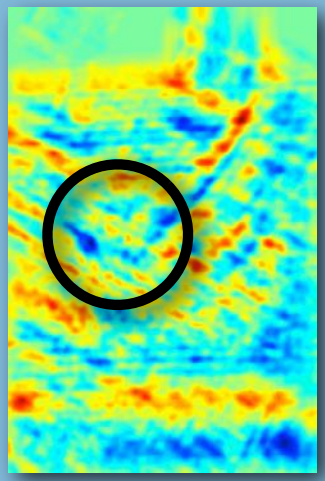


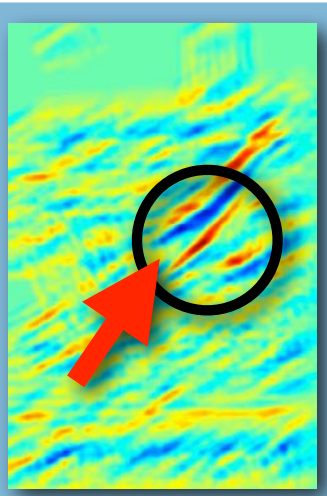


[Ramanan, NIPS'06]



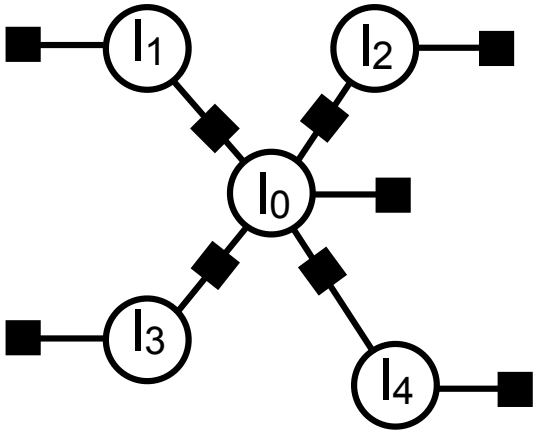






Efficient Inference - first for Star Model

- Location $L = \{l_0, l_1, \ldots, l_N\}$
  defines where each part is positioned in image



- best location given by

$$p(L|E) \propto p(L)p(E|L)$$

$$= p(l_0) \prod_{i=1}^{N} p(l_i|l_0) \prod_{i=0}^{N} p(e_i|l_i)$$

$$= \prod_{i=0}^{N} p(l_i|l_0) \prod_{i=0}^{N} p(e_i|l_i)$$

$$= \prod_{i=0}^{N} p(l_i|l_0)p(e_i|l_i)$$

Efficient Inference - first for Star Model

- best location given by MAP:

$$\max_L p(L|E) = \max_L \prod_{i=0}^{N} (p(l_i|l_0)p(e_i|l_i))$$

$$= \min_L \sum_{i=0}^{N} (-\ln p(l_i|l_0) - \ln p(e_i|l_i))$$

- consider case of 2 parts:

$$\min_{l_0,l_1} (-\ln p(e_0|l_0) - \ln p(e_1|l_1) - \ln p(l_1|l_0))$$

- rename things:

$$= \min_{l_0,l_1} (m_0(l_0) + m_1(l_1) + d(l_1, l_0))$$

Efficient Inference - first for Star Model

- assume d to have quadratic form, e.g.:

$$d(l_1, l_0) = ||l_1 - T_1(l_0)||^2$$

  - T_1 could be a simple offset of part 1 w.r.t. part 0
  - e.g. the case when $p(l_i \mid l_0)$ is Gaussian (see above)

- then

$$\min_{l_0, l_1} \left(m_0(l_0) + m_1(l_1) + d(l_1, l_0)\right)$$

$$= \min_{l_0} \left(m_0(l_0) + \min_{l_1} \left(m_1(l_1) + d(l_1, l_0)\right) \right)$$

  - with the second term a generalized distance transform (DT) algorithm exist to compute DT efficiently
  - thus:

$$= \min_{l_0} \left(m_0(l_0) + DT_{m_1}(T_1(l_0))\right)$$

- finding best part configuration can be done sequentially rather then simultaneously

Distance Transform

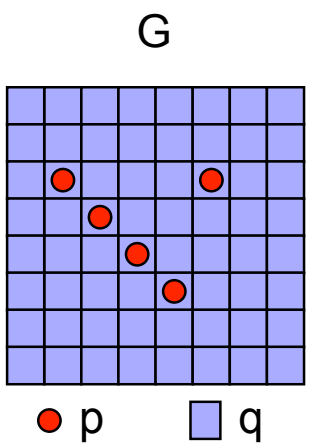
- given points $p \in P$ on a grid (e.g. image) G
- distance transform associates to each location $x \in G$ the distance to the nearest point $p \in P$:

$$DT_P(x) = \min_{p \in P} \{d(x, p)\}$$

  - or equivalent:

$$DT_P(x) = \min_{q \in G} \{d(x, q) + 1(q)\}$$

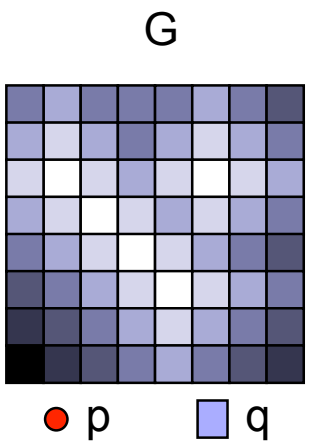
$$1(q) = \begin{cases} 0 & \text{if } q \in P \\ \infty & \text{otherwise} \end{cases}$$



example:

$$d(x, q) = |x - q|$$

$$DT_P(x) = \min_{q \in G} \{|x - q| + 1(q)\}$$



Generalized Distance Transform

- replace binary function 1(q) with function f(q):

$$DT_f(x) = \min_{q \in G} \{d(x,q) + f(q)\}$$

  - we can assign ‘soft’ membership of all grid elements to P
  - f(q) is sampled on the entire grid G

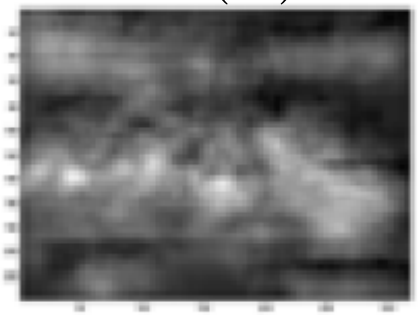
- in our case:
  - f corresponds to m_1
  - distance corresponds to $d(l_1, l_0) = ||l_1 - T_1(l_0)||^2$

$$DT_{m_1}(T_1(l_0)) = \min_{l_1} \{m_1(l_1) + d(l_1, l_0)\}$$

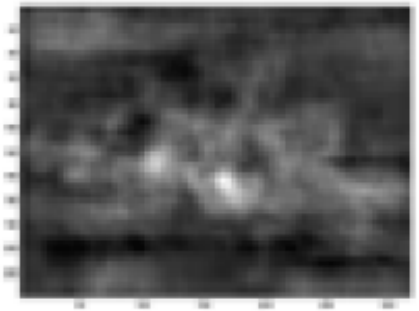
Example - 2 Part Model of Motorbikes

- Model
  - 2 parts (use both wheels), simple translation between them location given by x,y-position in image

$m_0(l_0)$



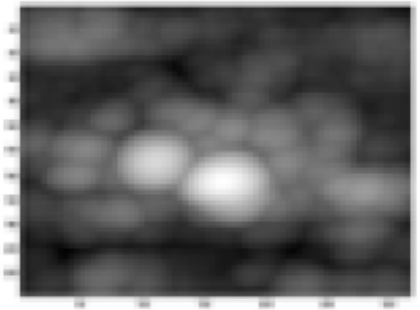
$m_1(l_1)$



- 1. part unaries (log probability) $m_0(l_0)$ and $m_1(l_1)$
- 2. distance transform of $m_1(l_1)$
- 3. simply find minimum of sum:

$$\min_{l_0} \left(m_0(l_0) + DT_{m_1}(T_1(l_0)) \right)$$

$DT_{m_1}(T_1(l_0))$

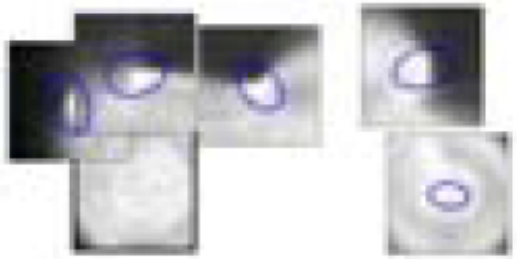


example from Dan Huttenlocher

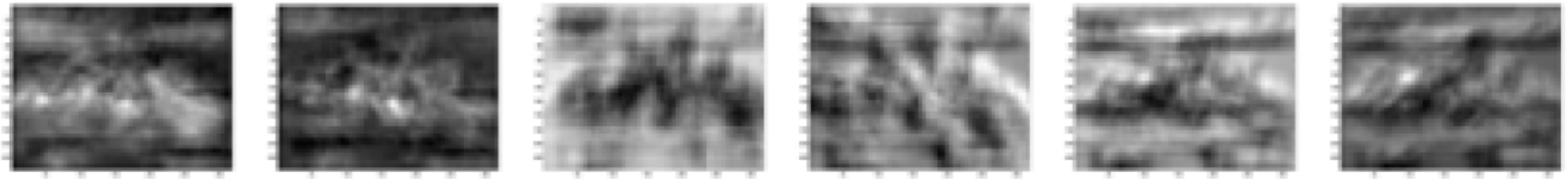
Example: Star Model of Motorbikes

- generalization of 2-part case:

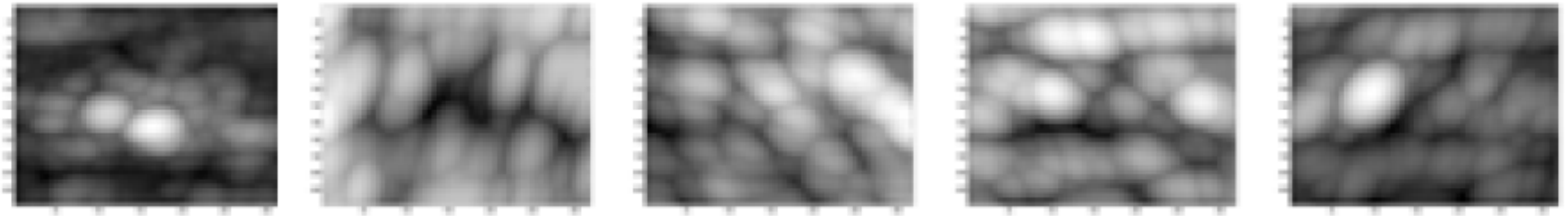
example from Dan Huttenlocher



- part likelihood maps (log likelihood) - cost O(LN)

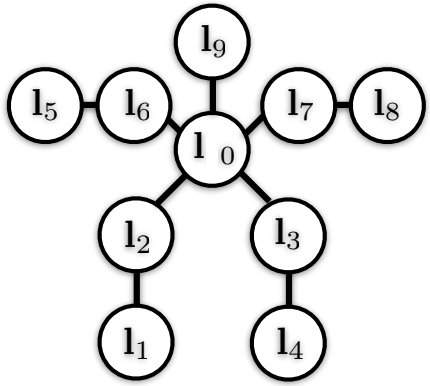


- and their distance transforms - cost O(LN)

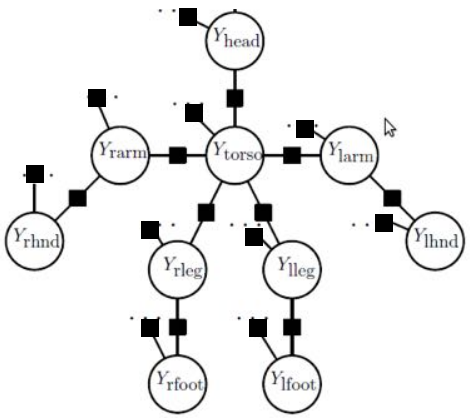


Tree-Structured Models

- distance transform can be used for any tree-structured model



- 2 differences w.r.t. star-model
  - relate position of parts to one another using tree-structured recursion
    - solve using Viterbi algorithm (for MAP)
    - solve using forward-backward algorithm for part marginals
  - parametrization of distance transform more involved
    - we have transformation T_{ij} for each connected pair of parts



Minimization over Tree Structures

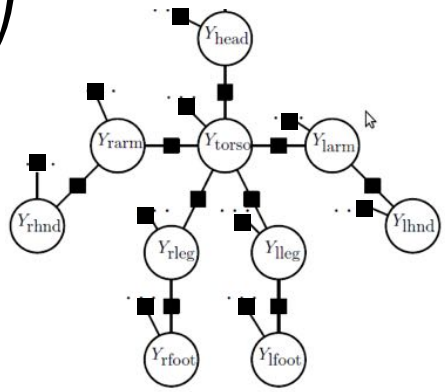
- use dynamic programming to minimize

$$\min_L \sum_{i=0}^{N} (-\ln p(e_i|l_i) - \ln p(l_i|l_j)) = \min_L \sum_{i=0}^{N} (m_i(l_i) + d_{ij}(l_i, l_j))$$

- can be expressed as a function for part pairs:
  - $B_c(l_j)$: cost of best location for part c given location l_j of part j
- recursive equation in terms of children C_j of part j passing recursive message to parent node l_i:

$$B_j(l_i) = \min_{l_j} \left(m_j(l_j) + d_{ij}(l_i, l_j) + \sum_{c \in C_j} B_c(l_j) \right)$$

  - for leaf node no children - so last term vanishes
  - for root note l_0 no parent - so second term vanishes



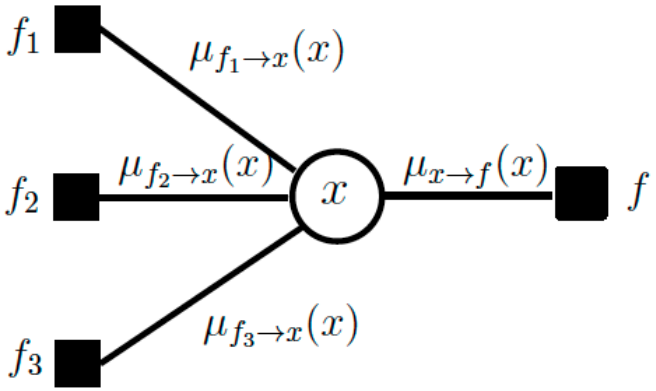
Efficient Algorithm for Trees

- MAP estimation algorithm
  - tree structure allows to use Viterbi style dynamic programming
    - for L locations and N parts: $O(N L^2)$ rather than $O(L^N)$
    - still typically slow as L is often in the order of 10^6 or 10^7
  - coupling with distance transform for finding best pair-wise locations can be done in linear time
    - resulting method then O(NL)

Relation to Max-Product (or Max-Sum) Rule

- Remember: variable to factor message:

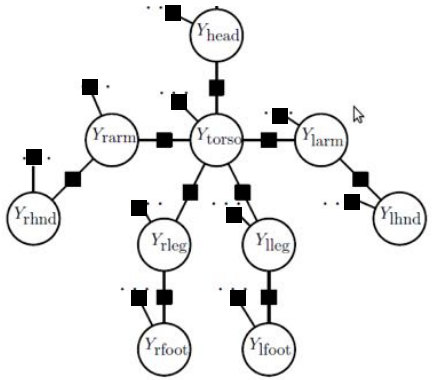
$$\mu_{x\to f}(x) = \prod_{g\in\{\mathrm{ne}(x)\setminus f\}} \mu_{g\to x}(x)$$



  - when using -ln(p(.)) rather than p(.) directly:

$$\mu_{x\to f}(x) = \sum_{g\in\{\mathrm{ne}(x)\setminus f\}} \mu_{g\to x}(x)$$

- in our case:
  - x corresponds to l_j,
  - f corresponds to d_{ij}

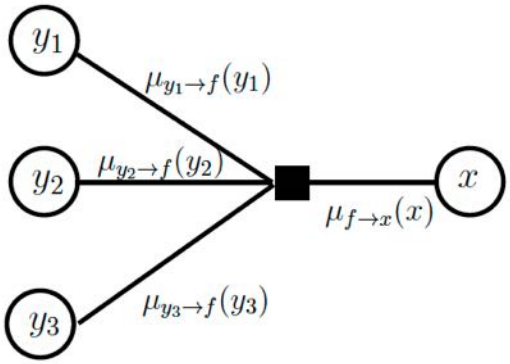


$$\mu_{l_j\to d_{ij}}(l_j) = \sum_{g\in\{\mathrm{ne}(l_j)\setminus d_{ij}\}} \mu_{g\to l_j}(l_j) = m_j(l_j) + \sum_{c\in C_j} B_c(l_j)$$

Relation to Max-Product (or Max-Sum) Rule

- Remember: factor to variable message:

$$\mu_{f\to x}(x) = \max_{y\in X_f\setminus x} \phi_f(X_f) \prod_{y\in\{\mathrm{ne}(f)\setminus x\}} \mu_{y\to f}(y)$$

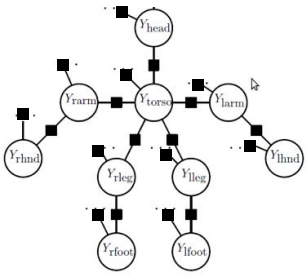


  - again when using -ln(p(.)) rather than p(.) directly:

$$\mu_{f\to x}(x) = \min_{y\in X_f\setminus x} -\ln(\phi_f(X_f)) + \sum_{y\in\{\mathrm{ne}(f)\setminus x\}} \mu_{y\to f}(y)$$

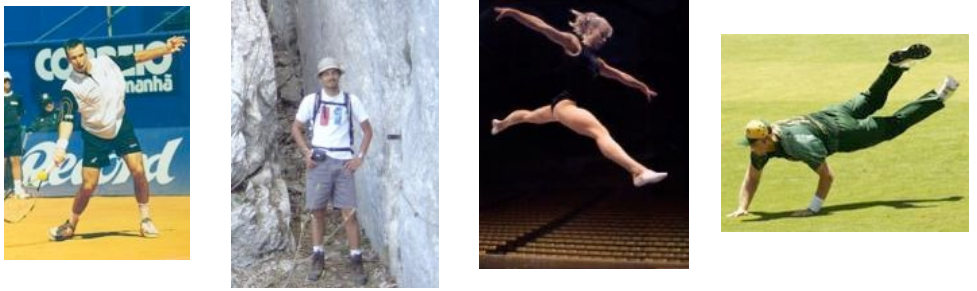
- in our case:
  - x corresponds to l_i, f corresponds to d_{ij}
  - for human kinematic tree prior: factor d_{ij} has only two neighbors: l_i and l_j

$$\begin{aligned}\mu_{d_{ij}\to l_i}(l_i) &= \min_{l_j}\left(-\ln(\phi_f(X_f)) + \mu_{l_j\to d_{ij}}(l_j)\right)\\ &= \min_{l_j}\left(d_{ij}(l_i,l_j) + \mu_{l_j\to d_{ij}}(l_j)\right)\end{aligned}$$



Evaluation Scenarios

1. Human Pose Estimation
   "People" dataset
   [Ramanan, NIPS'06]

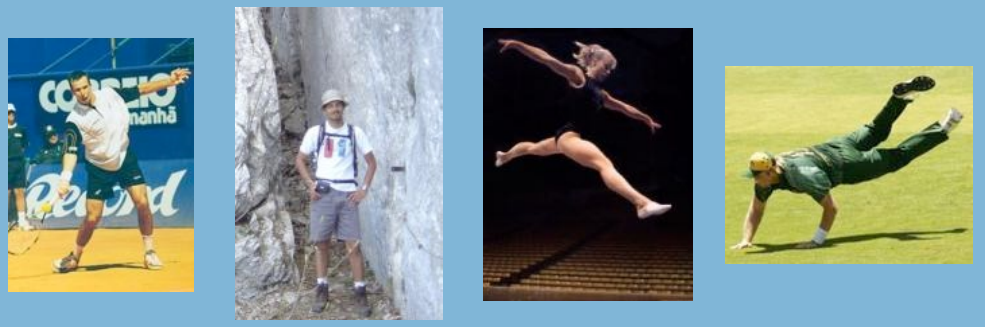


2. Upper-body Pose Estimation
   "Buffy" dataset
   [Ferrari et al., CVPR'08]

3. Pedestrian Detection
   "TUD Pedestrians" dataset
   [Andriluka et al., CVPR'08]

Evaluation Scenarios

1. Human Pose Estimation
   "People" dataset
   [Ramanan, NIPS'06]

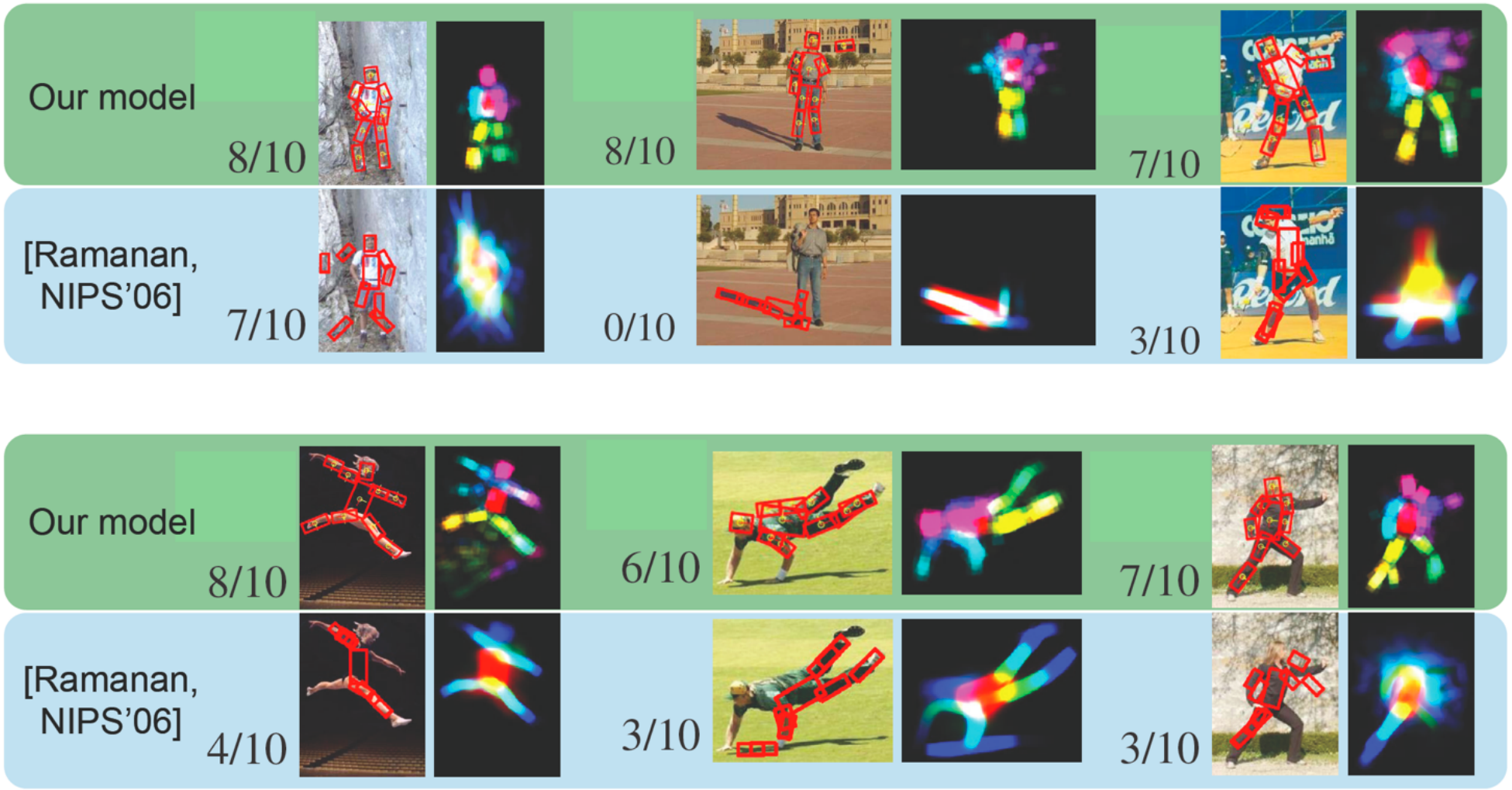


2. Upper-body Pose Estimation
   "Buffy" dataset
   [Ferrari et al., CVPR'08]

3. Pedestrian Detection
   "TUD Pedestrians" dataset
   [Andriluka et al., CVPR'08]

Part-Based Model: 2D Human Pose Estimation

[Andriluka,Roth,Schiele@cvpr09]



Scenario 1: Quantitative Results [cvpr'09]

Method	Torso	Upper legs	Lower legs	Upper arms	Forearms	Head	Total
[Ramanan, NIPS'06] 2nd parse	52	30	29	17	13	37	27
Our inference, edge features from [Ramanan, NIPS'06]	63	48	37	26	20	45	37
Our part detectors (SC)	29	12	18	3	4	40	14
Our prior, our part detectors (SC)	**81**	**63**	**55**	**47**	**31**	**75**	**55**
Our prior, our part detectors (SIFT)	78	58	54	44	31	66	52

Scenario 1: Model Parameters

- in particular - how big should L be?
  - remember - L depends on location (x,y), rotation and scale of parts
  - table 3 @ ijcv'11:

Discretization step	Torso	Upper leg		Lower leg		Upper arm		Forearm		Head	Total
8 px., 15°	**84.9**	67.8	60.5	63.4	45.4	48.3	48.3	32.2	29.8	75.6	55.6
4 px., 15°	83.9	67.8	61.46	65.9	46.8	**53.2**	46.3	32.7	32.7	76.1	56.7
2 px., 15°	83.9	**69.8**	61.5	66.3	46.8	50.7	49.8	33.7	33.7	76.1	57.2
2 px., 7.5°	**84.9**	68.3	62.0	66.3	**50.7**	**53.2**	**50.2**	**37.0**	31.2	78.5	58.2
1 px., 7.5°	**84.9**	69.3	**63.4**	**68.3**	48.8	51.7	**50.2**	**36.6**	**34.2**	**79.0**	**58.6**

  - unfortunately: the larger L - the better normally the performance
  - but: reasonable performance already with smallest L (8 px, 15°)

Scenario 1: Difference in Inference Method

- comparison of sum-product and max-product rule
  - table 6 @ ijcv'11

Method	Torso	Upper leg		Lower leg		Upper arm		Forearm		Head	Total
sum-product, 8 px., 15°	82.0	69.8	61.0	64.9	46.8	49.3	46.3	34.2	28.3	76.1	55.9
max-product, 8 px., 15°	79.0	64.4	57.0	59.0	41.5	42.4	41.5	32.2	30.7	69.8	51.8
sum-product, 1 px., 7.5°	**84.9**	69.3	63.4	**68.3**	48.8	51.7	**50.3**	**36.6**	**34.2**	**79.0**	58.6
max-product, 1 px., 7.5°	83.9	**70.2**	**65.4**	67.8	**50.7**	**53.7**	**50.2**	**36.6**	33.6	75.1	**58.7**

Evaluation Scenarios

1. Human Pose Estimation
   "People" dataset
   [Ramanan, NIPS'06]




2. Upper-body Pose Estimation
   "Buffy" dataset
   [Ferrari et al., CVPR'08]



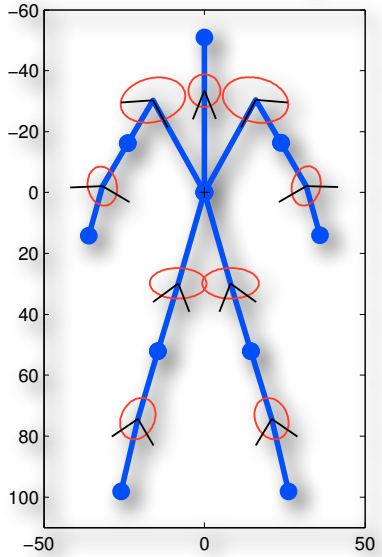
3. Pedestrian Detection
   "TUD Pedestrians" dataset
   [Andriluka et al., CVPR'08]



Estimated upper-body poses

Quantitative Results

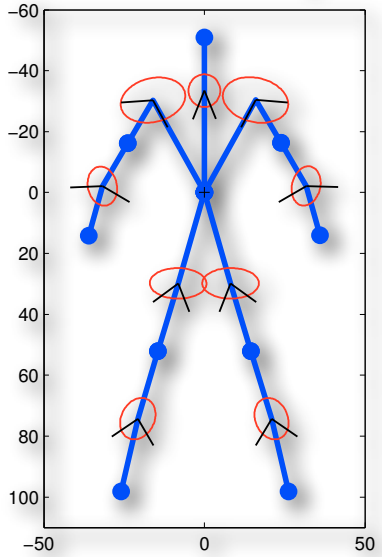
Method	Torso	Upper arm	Lower arm	Head	Total
[Ferrari et al. CVPR'08]	-	-	-	-	57.9
detectors only	18.9	6.8	3.1	47.2	14.3
full model	**90.7**	79.3	41.2	**95.9**	71.3



- generic model
- prior and appearance learned on the "People" dataset

Quantitative Results

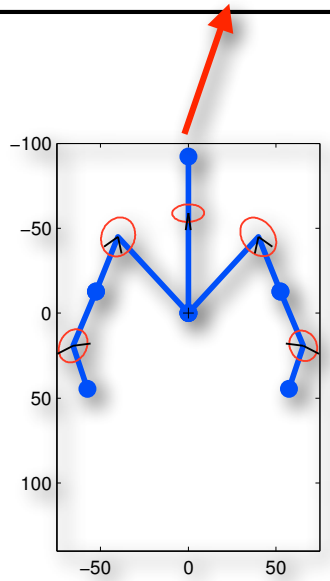
Method	Torso	Upper arm	Lower arm	Head	Total
[Ferrari et al. CVPR'08]	-	-	-	-	57.9
detectors only	18.9	6.8	3.1	47.2	14.3
full model	**90.7**	79.3	41.2	**95.9**	71.3
[Ferrari et al. CVPR'09]	-	-	-	-	**72.2**



- generic model
- prior and appearance learned on the "People" dataset

Quantitative Results

Method	Torso	Upper arm	Lower arm	Head	Total
[Ferrari et al. CVPR'08]	-	-	-	-	57.9
detectors only	18.9	6.8	3.1	47.2	14.3
full model	**90.7**	79.3	41.2	**95.9**	71.3
[Ferrari et al. CVPR'09]	-	-	-	-	72.2
full model, Buffy pose prior	**90.7**	**81.35**	**46.5**	95.5	**73.5**

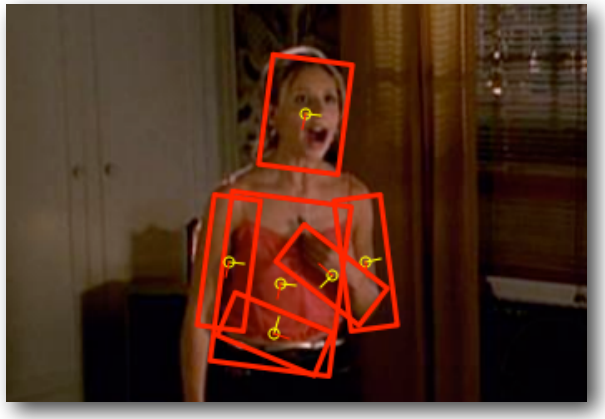


- specialized upper body prior
- appearance learned on the "People" dataset

Typical Failure Cases

Foreshortening

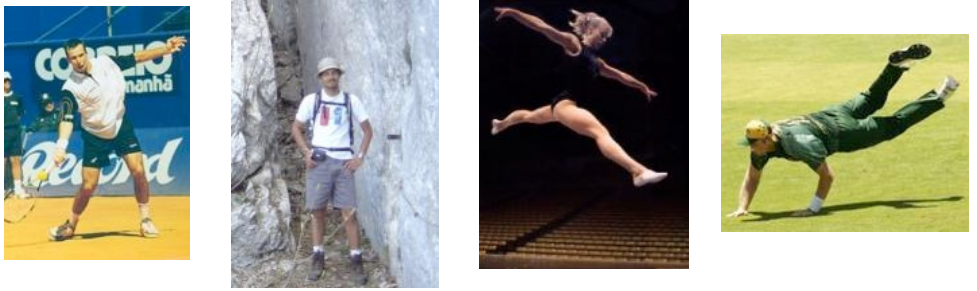
Part occlusion



Detections on other body parts

Evaluation Scenarios

1. Human Pose Estimation
   "People" dataset
   [Ramanan, NIPS'06]

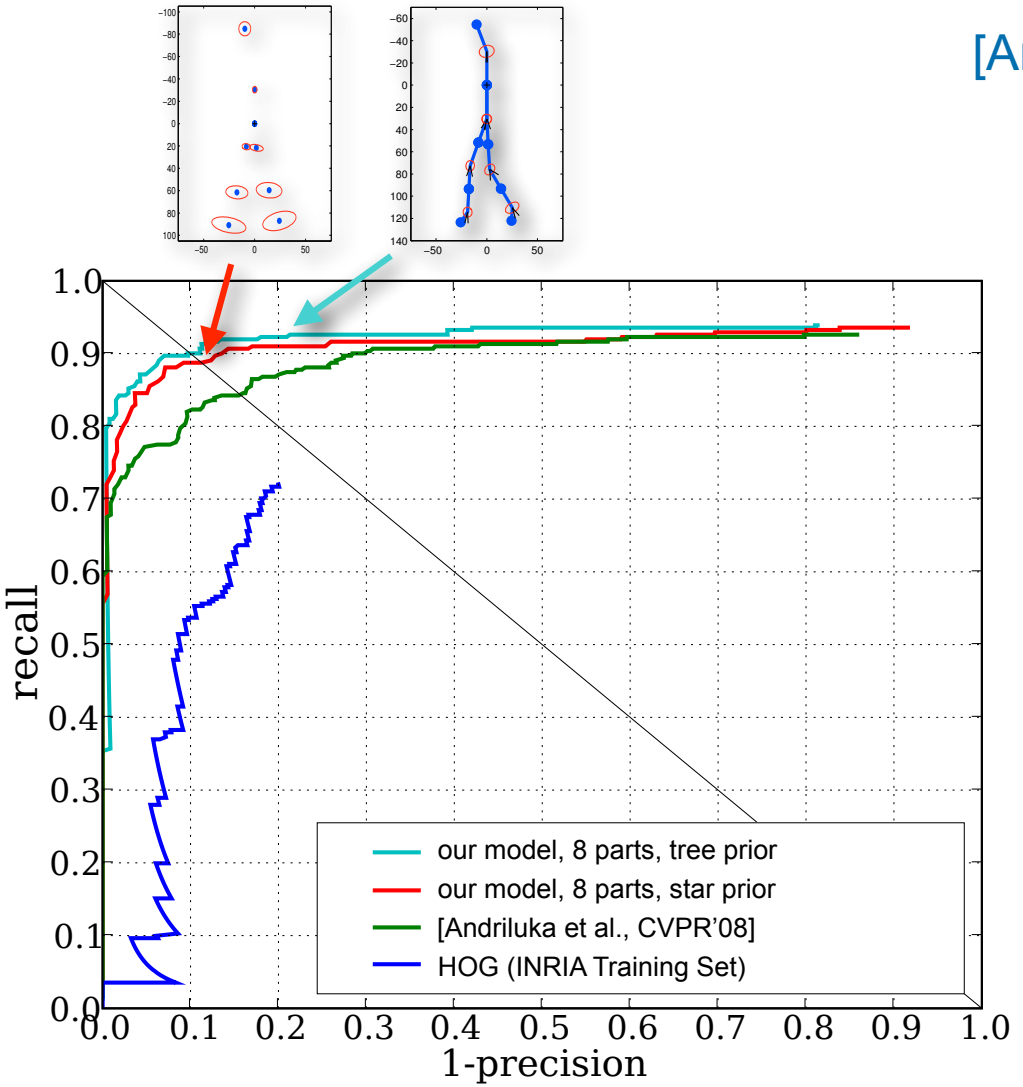


2. Upper-body Pose Estimation
   "Buffy" dataset
   [Ferrari et al., CVPR'08]

3. Pedestrian Detection
   "TUD Pedestrians" dataset
   [Andriluka et al., CVPR'08]

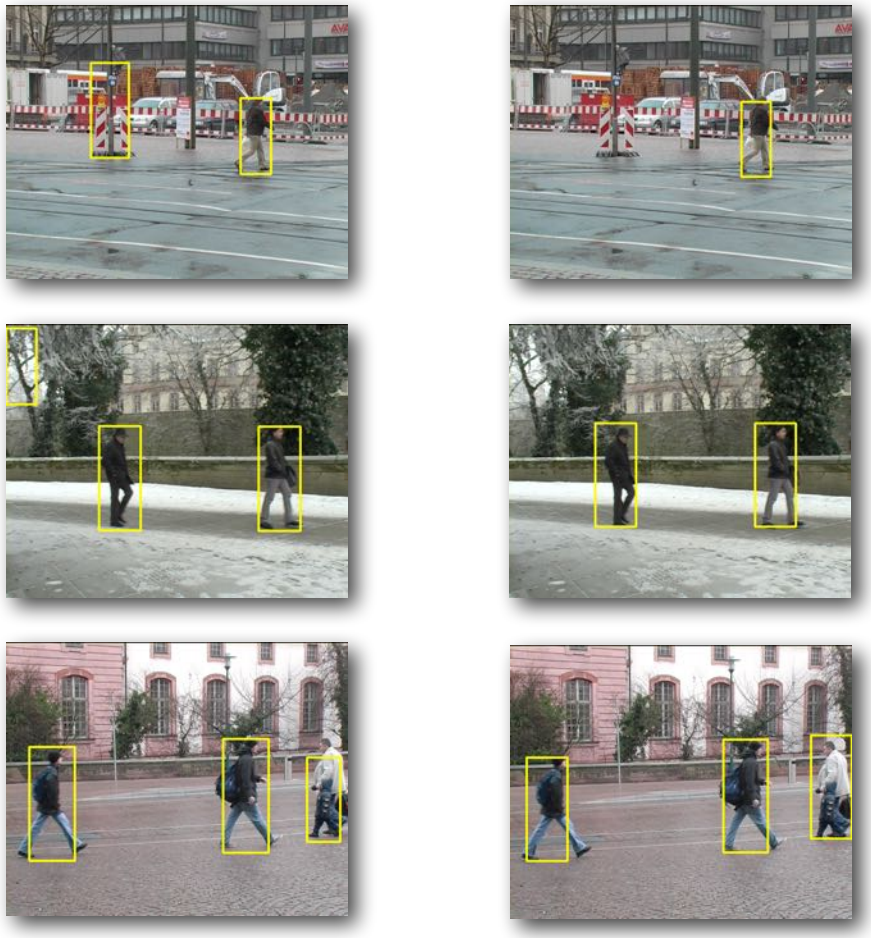
People Detection: Results

- Comparison with state-of-the art in people detection



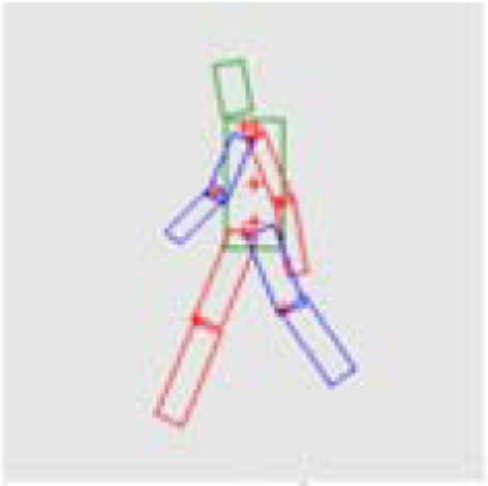
[Andriluka et al., CVPR'08]

This work



Limitations of Kinematic Trees

- represent only relationships between connected parts
- coordination (or relation) between limbs not encoded
  - critical e.g. for balance and many activities
  - example:
    these two configurations
    are equally probable
    under tree model:



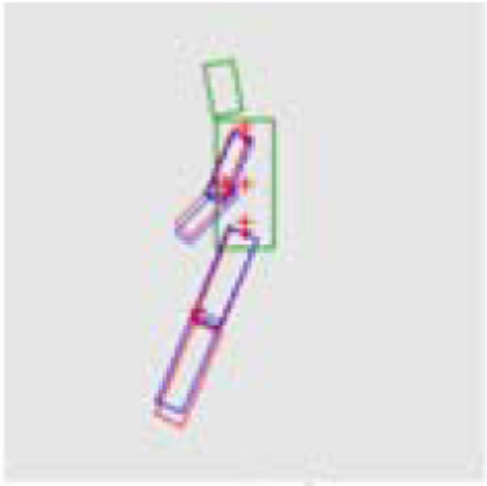
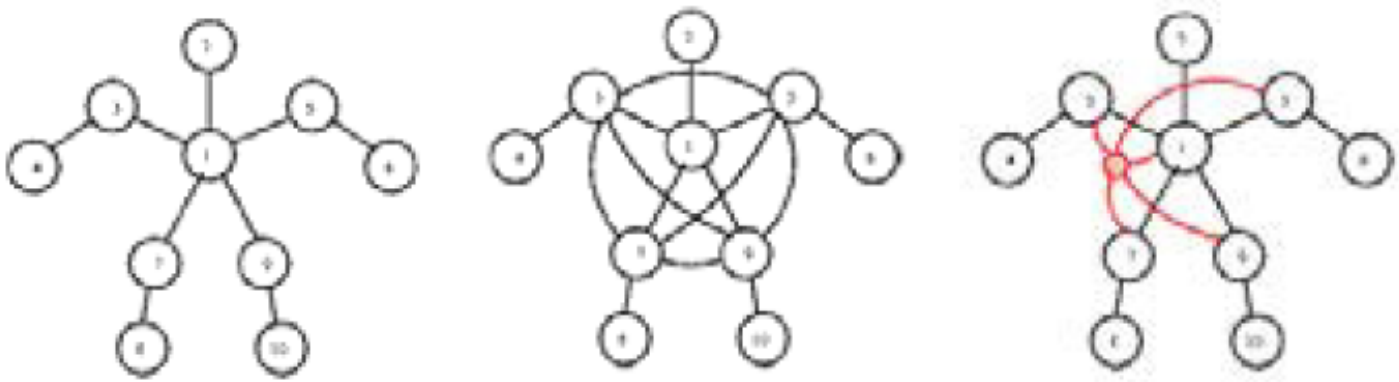


image from Dan Huttenlocher

Beyond Kinematic Tree Models

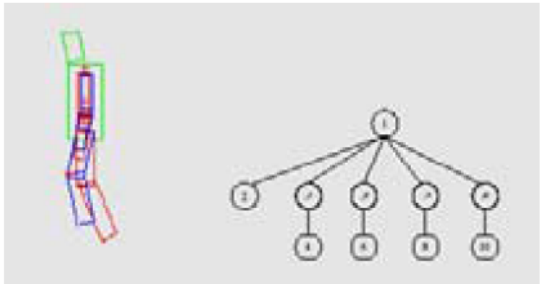
- non-tree models
  - larger cliques, latent variables
    - introduce additional variable corresponding to common factor of limb coordination
    - does not correspond to any part
    - dependencies e.g. among part orientations
  - still relatively efficient inference for small clique

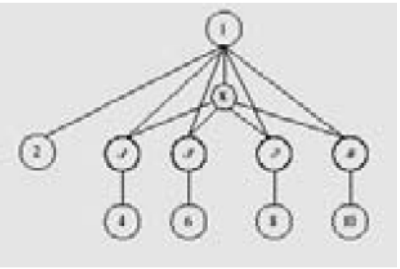


picture from Dan Huttenlocher

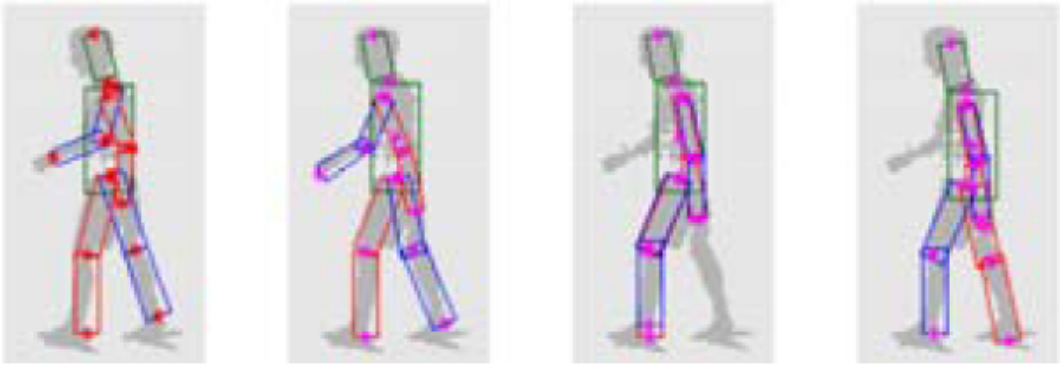
Beyond Kinematic Tree Models

- introduce additional variable corresponding to common factor (such as particular viewpoint or pose)
  - consistency between limb positions can be modeled, not captured by kinematic tree model
  - rather than directly connecting limbs which create large cliques



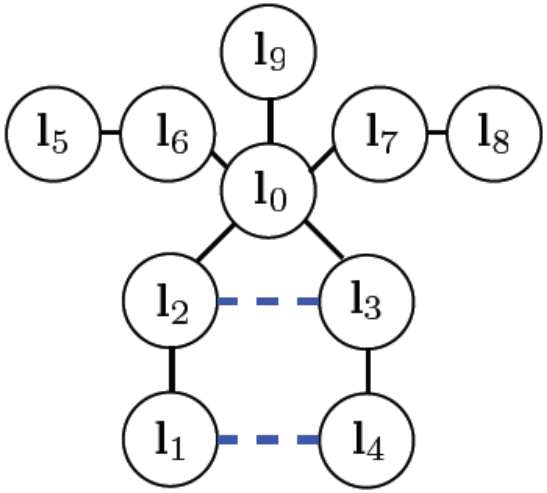


- sample result



example from Dan Huttenlocher

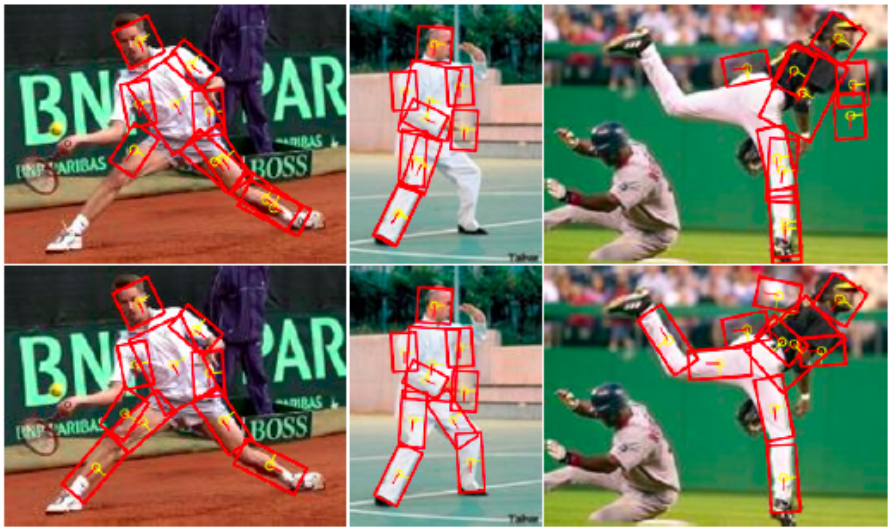
Beyond Kinematic Tree Models



- Repulsive Factor:
  - down-weight configurations that are overlapping:

$$f(l_i, l_j) = \begin{cases} \exp(-\beta) & : \quad IoU(l_i, l_j) > \delta \\ 1 & : \quad otherwise \end{cases}$$

  - IoU: Intersection over Union (measures overlap between parts)
  - here only between upper/lower legs
  - note: we use loopy belief propagation instead of exact inference



- Results: (table 5 @ ijcv11)

Method	Torso	Upper leg		Lower leg		Upper arm		Forearm		Head	Total
tree model	**84.9**	69.3	63.4	68.3	48.8	**51.7**	**50.2**	**36.6**	**34.2**	79.0	58.6
tree model + repulsive factor	**84.9**	**71.7**	**66.8**	**71.7**	**54.2**	51.2	**50.2**	36.1	**34.2**	**79.5**	**60.1**