Category-level localization and human pose estimation

Josef Sivic

Slides from Andrew Zisserman and Deva Ramanan

Also includes slides from: Ondra Chum, Alyosha Efros, Mark Everingham, Pedro Felzenszwalb, Rob Fergus, Kristen Grauman, Bastian Leibe, Fei-Fei Li, Marcin Marszalek, Pietro Perona, Bernt Schiele, Jamie Shotton, Andrea Vedaldi
Announcements

• Assignment 2 was due today. **Have you sent it?**

• Assignment 3 is out. 

• Topic ideas for the final projects (any questions?):
Send us your **project proposal** by this **Friday (Nov 9)**.
What we would like to be able to do...

- Visual scene understanding
- **What** is in the image and **where**

- Object categories, identities, properties, activities, relations, …
Recognition Tasks

- **Image Classification**
  - Does the image contain an aeroplane?

- **Object Class Detection/Localization**
  - Where are the aeroplanes (if any)?

- **Object Class Segmentation**
  - Which pixels are part of an aeroplane (if any)?
Feature: Histogram of Oriented Gradients (HOG)

- tile 64 x 128 pixel window into 8 x 8 pixel cells
- each cell represented by histogram over 8 orientation bins (i.e. angles in range 0-180 degrees)
Window (Image) Classification

- HOG Features
- Linear SVM classifier

Feature Extraction

Training Data

Classifier

$F(x)$

$x$

pedestrian/Non-pedestrian

$P(c|x) \propto F(x)$
Learned model

\[ f(x) = w^T x + b \]
What do negative weights mean?

\[ wx > 0 \]
\[ (w^+ - w^-)x > 0 \]
\[ w^+ > w^-x \]

Complete system should compete pedestrian/pillar/doorway models

Discriminative models come equipped with own bg
(avoid firing on doorways by penalizing vertical edges)
Object Detection with Discriminatively Trained Part Based Models

Pedro F. Felzenszwalb, David Mcallester, Deva Ramanan, Ross Girshick
PAMI 2010

Matlab code available online:
http://www.cs.brown.edu/~pff/latent/
Approach

- Mixture of deformable part-based models
  - One component per “aspect” e.g. front/side view
- Each component has global template + deformable parts
- Discriminative training from bounding boxes alone
Example Model

- One component of person model

root filters
coarse resolution

part filters
finer resolution

deformation models
Starting Point: HOG Filter

- Search: sliding window over position and scale
- Feature extraction: HOG Descriptor
- Classifier: Linear SVM

Score of $F$ at position $p$ is $F \cdot \varphi(p, H)$

$\varphi(p, H) = \text{concatenation of HOG features from subwindow specified by } p$

Dalal & Triggs [2005]
Object Hypothesis

- Position of root + each part
- Each part: HOG filter (at higher resolution)

\[ z = (p_0, ..., p_n) \]
- \( p_0 \): location of root
- \( p_1, ..., p_n \): location of parts

Score is sum of filter scores minus deformation costs
Score of a Hypothesis

\[ \text{score}(p_0, \ldots, p_n) = \sum_{i=0}^{n} F_i \cdot \phi(H, p_i) - \sum_{i=1}^{n} d_i \cdot (dx_i^2, dy_i^2) \]

- Appearance term
- Spatial prior

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

- Concatenation of filters and deformation parameters
- Concatenation of HOG features and part displacement features

- Linear classifier applied to feature subset defined by hypothesis
Training

• Training data = images + bounding boxes
• Need to learn: model structure, filters, deformation costs
Latent SVM (MI-SVM)

Classifiers that score an example \( x \) using

\[ f_\beta(x) = \max_{z \in Z(x)} \beta \cdot \Phi(x, z) \]

\( \beta \) are model parameters
\( z \) are latent values

Training data \( D = (\langle x_1, y_1 \rangle, \ldots, \langle x_n, y_n \rangle) \) \( y_i \in \{-1, 1\} \)

We would like to find \( \beta \) such that: \( y_i f_\beta(x_i) > 0 \)

Minimize

\[ L_D(\beta) = \frac{1}{2} ||\beta||^2 + C \sum_{i=1}^{n} \max(0, 1 - y_i f_\beta(x_i)) \]

Regularizer

“Hinge loss” on one training example

SVM objective
Latent SVM Training

\[
L_D(\beta) = \frac{1}{2}\|\beta\|^2 + C \sum_{i=1}^{n} \max(0, 1 - y_i f_\beta(x_i))
\]

- Convex if we fix \( z \) for positive examples

- Optimization:
  - Initialize \( \beta \) and iterate:
    - Pick best \( z \) for each positive example
    - Optimize \( \beta \) with \( z \) fixed

- Local minimum: needs good initialization
  - Parts initialized heuristically from root

Alternation strategy
Person Model

root filters
coarse resolution

part filters
finer resolution

deformation models

Handles partial occlusion/truncation
Car Model

- root filters
  coarse resolution
- part filters
  finer resolution
- deformation models
Car Detections

high scoring true positives  high scoring false positives
Person Detections

- high scoring true positives
- high scoring false positives (not enough overlap)
Precision/Recall: VOC2008 Person
Precision/Recall: VOC2008 Bicycle

![Graph showing precision-recall curves for various datasets with different metrics.]
Comparison of Models
Summary

• **Multiple features** and multiple **kernels** boost performance

• Discriminative learning of model with latent variables for **single feature** (HOG):
  – Latent variables can learn best alignment in the ROI training annotation
  – Parts can be thought of as local SIFT vectors
  – Some similarities to Implicit Shape Model/Constellation models but with discriminative/careful training throughout

NB: Code available for latent model!
Current Research Challenges

• Context (See class on scenes and objects on Dec 3).
  – from scene properties: GIST, BoW, stuff
  – from other objects
  – from geometry of scene, e.g. Hoiem et al CVPR 06

• Occlusion/truncation
  – Winn & Shotton, Layout Consistent Random Field, CVPR 06
  – Vedaldi & Zisserman, NIPS 09
  – Yang et al, Layered Object Detection, CVPR 10

• 3D
  – Zhu&Ramanan, CVPR’12 (view-based representation of faces)

• Scaling up – thousands of classes
  – Torralba et al, feature sharing
  – ImageNet

• Weak and noisy supervision
Pictorial structure model re-visited: efficient fitting

Let’s have a closer look at the LSVM deformable part-based model…
Object Hypothesis

- Position of root + each part
- Each part: HOG filter (at higher resolution)

\[
z = (p_0, \ldots, p_n)
\]

- \(p_0\) : location of root
- \(p_1, \ldots, p_n\) : location of parts

Score is sum of filter scores minus deformation costs
What is the cost of fitting the PS model?

- For fixed (learned) $F_i$ and $d_i$
- For simplicity, consider only single scale of the pyramid
- Parts can appear anywhere in the image (h=number of pixels)

\[
\text{score}(p_0, \ldots, p_n) = \sum_{i=0}^{n} F_i \cdot \phi(H, p_i) - \sum_{i=1}^{n} d_i \cdot (dx_i^2, dy_i^2)
\]

- $p_0$: location of root
- $p_1, \ldots, p_n$: location of parts

Fitting cost: Naïve search is $O(nh^2)$
What is the cost of fitting the PS model?

- For fixed (learned) $F_i$ and $d_i$
- For simplicity, consider only single scale of the pyramid
- Parts can appear anywhere in the image ($h$=number of pixels)

\[
\text{score}(p_0, \ldots, p_n) = \sum_{i=0}^{n} F_i \cdot \phi(H, p_i) - \sum_{i=1}^{n} d_i \cdot (dx^2_i, dy^2_i)
\]

Fitting cost: Naïve search is $O(nh^2)$

Need to evaluate the deformation cost of each part with respect to the root.
Maximization of the PS score can be re-written as a **minimization** of the following cost function on a “star” graph:

$$f(x) = \sum_{v_i \in V} m_i(v_i) + \sum_{e_{ij} \in E} \phi(v_i, v_j)$$

- **Graph** \((V, E)\)
- **Vertices** \(v_i\) for \(i = 1, \ldots, n\)
- **Edges** \(e_{ij}\) connect \(v_i\) to other vertices \(v_j\)
Dynamic programming on graphs

- **Graph** \((V, E)\)
- **Vertices** \(v_i\) for \(i = 1, \ldots, n\)
- **Edges** \(e_{ij}\) connect \(v_i\) to other vertices \(v_j\)

\[
f(x) = \sum_{v_i \in V} m_i(v_i) + \sum_{e_{ij} \in E} \phi(v_i, v_j)
\]
Dynamic programming - review

- Discrete optimization
- Each variable $x$ has a finite number of possible states
- Applies to problems that can be decomposed into a sequence of stages
- Each stage expressed in terms of results of fixed number of previous stages
- The cost function need not be convex
- The name “dynamic” is historical
- Also called the “Viterbi” algorithm
- Let’s first consider a chain:
Consider a cost function $f(x): \mathbb{R}^n \to \mathbb{R}$ of the form

$$f(x) = \sum_{i=1}^{n} m_i(x_i) + \sum_{i=2}^{n} \phi_i(x_{i-1}, x_i)$$

where $x_i$ can take one of $h$ values

e.g. $h=5$, $n=6$

$$f(x) = \begin{cases} m_1(x_1) + m_2(x_2) + m_3(x_3) + m_4(x_4) + m_5(x_5) + m_6(x_6) \\ \phi(x_1, x_2) + \phi(x_2, x_3) + \phi(x_3, x_4) + \phi(x_4, x_5) + \phi(x_5, x_6) \end{cases}$$

Complexity of minimization:

- exhaustive search $O(h^n)$
- dynamic programming $O(nh^2)$
Key idea: the optimization can be broken down into $n$ sub-optimizations

**Step 1:** For each value of $x_2$ determine the best value of $x_1$

- Compute
  
  $$S_2(x_2) = \min_{x_1} \{ m_2(x_2) + m_1(x_1) + \phi(x_1, x_2) \}$$
  $$= m_2(x_2) + \min_{x_1} \{ m_1(x_1) + \phi(x_1, x_2) \}$$

- Record the value of $x_1$ for which $S_2(x_2)$ is a minimum

To compute this minimum for all $x_2$ involves $O(h^2)$ operations
Step 2: For each value of $x_3$ determine the best value of $x_2$ and $x_1$

- Compute

$$S_3(x_3) = m_3(x_3) + \min_{x_2} \{ S_2(x_2) + \phi(x_2, x_3) \}$$

- Record the value of $x_2$ for which $S_3(x_3)$ is a minimum

Again, to compute this minimum for all $x_3$ involves $O(h^2)$ operations

Note $S_k(x_k)$ encodes the lowest cost partial sum for all nodes up to $k$ which have the value $x_k$ at node $k$, i.e.

$$S_k(x_k) = \min_{x_1, x_2, \ldots, x_k} \sum_{i=1}^{k} m_i(x_i) + \sum_{i=2}^{k} \phi(x_{i-1}, x_i)$$
Viterbi Algorithm

- Initialize $S_1(x_1) = m_1(x_1)$
- For $k = 2 : n$
  
  $S_k(x_k) = m_k(x_k) + \min_{x_{k-1}} \{S_{k-1}(x_{k-1}) + \phi(x_{k-1}, x_k)\}$
  
  $b_k(x_k) = \arg \min_{x_{k-1}} \{S_{k-1}(x_{k-1}) + \phi(x_{k-1}, x_k)\}$

- Terminate
  
  $x^*_n = \arg \min_{x_n} S_n(x_n)$

- Backtrack
  
  $x_{i-1} = b_i(x_i)$

Complexity $O(nh^2)$
Dynamic programming on graphs

- Graph $(V, E)$
- Vertices $v_i$ for $i = 1, \ldots, n$
- Edges $e_{ij}$ connect $v_i$ to other vertices $v_j$

$$f(x) = \sum_{v_i \in V} m_i(v_i) + \sum_{e_{ij} \in E} \phi(v_i, v_j)$$

So far have considered chains

1 2 3 4 5 6
Different graph structures

- Fully connected: $O(h^n)$
- Tree structure: $O(nh^2)$
- Star structure: $O(nh^2)$

Can use dynamic programming

- n parts
- h positions (e.g. every pixel for translation)
Distance transforms for DP
Special case of DP cost function

- Distance transforms
  - $O(nh^2) \rightarrow O(nh)$ for DP cost functions
  - Assume model is quadratic, i.e.
    \[
    \phi(x_{k-1}, x_k) = \lambda^2(x_{k-1} - x_k)^2
    \]

Recall that we need to compute

\[
\min_{x_{k-1}} \{ S_{k-1}(x_{k-1}) + \phi(x_{k-1}, x_k) \}
\]

e.g. for $k = 2$, compute for each value of $x_2$

\[
\min_{x_1} \{ m_1(x_1) + \phi(x_1, x_2) \}
\]

Plot $\min_{x_1} \{ m_1(x_1) + \phi(x_1, x_2) \}$ as function of $x_2$
Plot \( \{m_1(x_1) + \phi(x_1, x_2)\} \) as function of \( x_2 \)

\[
\phi(x_1 = a, x_2) = \lambda^2 (x_2 - a)^2
\]

\[
l^2 (x_2 - b)^2
\]
Plot \( \{m_1(x_1) + \phi(x_1, x_2)\} \) as function of \( x_2 \) for each \( x_1 \)

For each \( x_2 \)
- Finding min over \( x_1 \) is equivalent finding minimum over set of offset parabolas
- Lower envelope computed in \( O(h) \) rather than \( O(h^2) \) via distance transform

Felzenszwalb and Huttenlocher '05
For each $x_2$
- Finding min over $x_1$ is equivalent to finding minimum over set of offset parabolas
- Lower envelope computed in $O(h)$ rather than $O(h^2)$ via distance transform
1D Examples

\[ D_f(x_2) = \min_{x_1} \{ m_1(x_1) + \phi(x_1, x_2) \} \]
1D Examples

\[ f(p) \]

\[ D_f(q) \]
1D Examples

\[ f(p) \]

\[ D_f(q) \]
**Generalized distance transform**

Given a function $f : \mathcal{G} \to \mathbb{R}$,

$$D_f(q) = \min_{p \in \mathcal{G}} \left( \|q - p\|^2 + f(p) \right)$$

- for each location $q$, find nearby location $p$ with $f(p)$ small.
- equals DT of points $P$ if $f$ is an indicator function.

$$f(p) = \begin{cases} 
0 & \text{if } p \in P \\
\infty & \text{otherwise}
\end{cases}.$$
There is a simple geometric algorithm that computes $D_f(p)$ in $O(h)$ time for the 1D case.

- similar to Graham’s scan convex hull algorithm.
- about 20 lines of C code.

The 2D case is “separable”, it can be solved by sequential 1D transformations along rows and columns of the grid.

See *Distance Transforms of Sampled Functions*, Felzenszwalb and Huttenlocher.
“Lower Envelope” Algorithm

Add first

Add second

Try adding third
Remove second
Try again and add

...
Algorithm for Lower Envelope

• Quadratics ordered left to right
• At step j consider adding j-th quadratic to LE of first j-1 quadratics

  – Maintain two ordered lists
    • Quadratics currently visible on LE
    • Intersections currently visible on LE

  – Compute intersection of j-th quadratic and rightmost quadratic visible on LE
    • If to right of rightmost visible intersection, add quadratic and intersection to lists
    • If not, this quadratic hides at least rightmost quadratic, remove it and try again

Code available online: http://people.cs.uchicago.edu/~pff/dt/
Running Time of LE Algorithm

• Considers adding each of h quadratics just once
  – Intersection and comparison constant time
  – Adding to lists constant time
  – Removing from lists constant time
    • But then need to try again

• Simple amortized analysis
  – Total number of removals O(h)
    • Each quadratic once removed never considered for removal again

• Thus overall running time O(h)
Coming back to fitting pictorial structures

Maximization of the PS score can be re-written as a minimization of the following cost function on a “star” graph:

\[ f(x) = \sum_{v_i \in V} m_i(v_i) + \sum_{e_{ij} \in E} \phi(v_i, v_j) \]

As the spatial prior is a quadratic function of part positions, \((x_i, y_i)\), finding the optimal configuration of parts can be done in \(O(nh)\) time, instead of naïve \(O(nh^2)\).
Part Detection

Response of filter in l-th pyramid level

\[ R_l(x, y) = F \cdot \phi(H, (x, y, l)) \]

cross-correlation

Transformed response

\[ D_l(x, y) = \max_{dx, dy} (R_l(x + dx, y + dy) - d_i \cdot (dx^2, dy^2)) \]

Distance transform computed in linear time (spreading, local max, etc)
System

- Feature map
- Feature map at twice the resolution
- Response of part filters
- Transformed responses
- Combined score of root locations

Color encoding of filter response values
Other applications of PS models: facial feature detection in images

The goal: Localize facial features in faces output by face detector

- Parts $V = \{v_1, \ldots, v_n\}$
- Connected by springs in a star configuration to nose (can be a tree)
- Quadratic cost for springs

high spring cost
Example part localizations in video
Example of a model with 9 parts

Support parts-based face descriptors
Provide initialization for global face descriptors

Code available online: http://www.robots.ox.ac.uk/~vgg/research/nface/index.html
Summary

- Pictorial structure models with tree configuration of parts can be fitted in $O(nh^2)$. \{n=number of parts, h=number of pixels\}

- For quadratic pair-wise terms this can be reduced to $O(nh)$.

- This can lead to significant speed-ups if $h$ is large (e.g. number of pixels).

Other applications:
- Facial feature finding
- Fitting articulated models
Human Pose Estimation
Objective and motivation

Determine human body pose (layout)

Why? To recognize poses, gestures, actions
Activities characterized by a pose
Activities characterized by a pose
Activities characterized by a pose
Challenges: articulations and deformations
Challenges: of (almost) unconstrained images

- varying illumination and low contrast
- moving camera and background
- multiple people
- scale changes
- extensive clutter
- any clothing
Pictorial Structures

- Intuitive model of an object
- Model has two components
  1. parts (2D image fragments)
  2. structure (configuration of parts)
- Dates back to Fischler & Elschlager 1973
From earlier: objects

Mixture of deformable part-based models
  • One component per “aspect” e.g. front/side view
Each component has global template + deformable parts
Discriminative training from bounding boxes alone
Localize multi-part objects at arbitrary locations in an image

- Generic object models such as person or car
- Allow for articulated objects
- Simultaneous use of appearance and spatial information
- Provide efficient and practical algorithms

To fit model to image: minimize an energy (or cost) function that reflects both

- **Appearance**: how well each part matches at given location
- **Configuration**: degree to which parts match 2D spatial layout
Long tradition of using pictorial structures for humans

Finding People by Sampling
Ioffe & Forsyth, ICCV 1999

Pictorial Structure Models for Object Recognition
Felzenszwalb & Huttenlocher, 2000

Learning to Parse Pictures of People
Ronfard, Schmid & Triggs, ECCV 2002
Felzenszwalb & Huttenlocher

NB: requires background subtraction
Variety of Poses
Variety of Poses
**Objective:** detect human and determine upper body pose (layout)

Model as a graph labelling problem

- Vertices $\mathcal{V}$ are parts, $a_i, i = 1, \ldots, n$
- Edges $\mathcal{E}$ are pairwise linkages between parts
- For each part there are $h$ possible poses $p_j = (x_j, y_j, \phi_j, s_j)$
- Label each part by its pose: $f : \mathcal{V} \rightarrow \{1, \ldots, h\}$, i.e. part $a$ takes pose $p_{f(a)}$. 
Pictorial structure model – CRF

- Each labelling has an energy (cost):

  \[ E(f) = \sum_{a \in \mathcal{V}} \theta_a; f(a) + \sum_{(a,b) \in \mathcal{E}} \theta_{ab}; f(a)f(b) \]

  - unary terms (appearance)
  - pairwise terms (configuration)

- Fit model (inference) as labelling with lowest energy

Features for unary:
- colour
- HOG
  for limbs/torso
Unary term: appearance feature I - colour

colour posteriors
Unary term: appearance feature II - HOG

Dalal & Triggs, CVPR 2005

Histogram of oriented gradients (HOG)

HOG of image

HOG of lower arm template (learned)

L2 Distance
Pairwise terms: kinematic layout

\[ \theta_{ab;ij} = w_{ab}d(|i-j|) \]
Pictorial structure model – CRF

Each labelling has an energy (cost):

\[ E(f) = \sum_{a \in V} \theta_a; f(a) + \sum_{(a,b) \in E} \theta_{ab}; f(a)f(b) \]

- Unary terms (appearance)
- Pairwise terms (configuration)

Fit model (inference) as labelling with lowest energy

Features for unary:
- colour
- HOG
  for limbs/torso
- $n$ parts
- For each part there are $h$ possible poses $p_j = (x_j, y_j, \phi_j, s_j)$
- There are $h^n$ possible labellings

**Problem:** any reasonable discretization (e.g. 12 scales and 36 angles for upper and lower arm, etc) gives a number of configurations $10^{12} - 10^{14}$

$\Rightarrow$ Brute force search not feasible
Are trees the answer?

• With $n$ parts and $h$ possible discrete locations per part, $O(h^n)$

• For a tree, using dynamic programming this reduces to $O(nh^2)$

• If model is a tree and has certain edge costs, then complexity reduces to $O(nh)$ using a distance transform [Felzenszwalb & Huttenlocher, 2000, 2005]
Kinematic structure vs graphical (independence) structure

Graph $G = (V,E)$

Requires more connections than a tree
Articulated Pose Estimation with Flexible Mixtures of Parts

Yi Yang & Deva Ramanan
Articulated pose estimation (by Wikipedia) recovers the pose of an articulated object which consists of joints and rigid parts.
Applications

<table>
<thead>
<tr>
<th>Action</th>
<th>HCI</th>
<th>Gaming</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Image" /></td>
<td><img src="image2.png" alt="Image" /></td>
<td><img src="image3.png" alt="Image" /></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Segmentation</th>
<th>Object Detection</th>
<th>......</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image4.png" alt="Image" /></td>
<td><img src="image5.png" alt="Image" /></td>
<td></td>
</tr>
</tbody>
</table>
Unconstrained Images
Classic Approach

Part Representation
- Head, Torso, Arm, Leg
- Location, Rotation, Scale

Pictorial Structure
- Unary Templates
- Pairwise Springs

Marr & Nishihara 1978
Fischler & Elschlager 1973
Felzenszwalb & Huttenlocher 2005

Lan & Huttenlocher 2005
Sigal & Black 2006
Ramanan 2007
Epshteian & Ullman 2007
Wang & Mori 2008
Ferrari etc. 2008

Andriluka etc. 2009
Eichner etc. 2009
Singh etc. 2010
Johnson & Everingham 2010
Sapp etc. 2010
Tran & Forsyth 2010
Problem

How to capture **affine** deformations of limbs?

<table>
<thead>
<tr>
<th>In plane rotation</th>
<th>Out plane rotation</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Images" /></td>
<td><img src="image2" alt="Images" /></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Foreshortening</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image3" alt="Images" /></td>
<td><img src="image4" alt="Images" /></td>
</tr>
</tbody>
</table>

Naïve brute-force evaluation is expensive
Our Approach – “Mini” Parts

Capture affine deformations with “mini” part model
Example: Arm Approximation
Example: Torso Approximation
Our Approach

• Extension of Pictorial Structure Model

• Why?

  Flexibility: General affine warps (orientation, foreshortening, ...)

  Speed: Mixtures of local templates + dynamic programming
Linear-Parameterized Pictorial Structure Model

\[ S(I, L) \]

- Image;
- Number of parts
- Locations of parts
Linear-Parameterized Pictorial Structure Model

- $V$: Vertices
- $a_i$: Unary template for part $i$
- $\phi(I, l_i)$: Local image features at location $l_i$

$$S(I, L) = \sum_{i \in V} \alpha_i \cdot \phi(I, l_i)$$

- $V$: Vertices
- $a_i$: Unary template for part
- $\phi$: Local image features at location $l_i$
Linear-Parameterized Pictorial Structure Model

\[ S(I, L) = \sum_{i \in V} \alpha_i \cdot \phi(I, l_i) + \sum_{i,j \in E} \beta_{ij} \cdot \psi(l_i, l_j) \]

- E: Edges
- Psi: Spatial features between i and j
- Beta: Pairwise springs between part and part
Our Flexible Mixture Model

\[ S(I, L, M) = \sum_{i \in V} \alpha_i^{m_i} \cdot \phi(I, l_i) + \sum_{i,j \in E} \beta_{i,j}^{m_i m_j} \cdot \psi(l_i, l_j) \]

- **M**: Mixtures of parts
- **alpha**: Unary template for part with mixture
- **Beta**: Pairwise springs between part I with mixture \( m_i \) and part j with mixture \( m_j \)
Our Flexible Mixture Model

\[ S(I, L, M) = \sum_{i \in V} \alpha_i^{m_i} \cdot \phi(I, l_i) + \sum_{i,j \in E} \beta_{i,j}^{m_i m_j} \cdot \psi(l_i, l_j) + S(M) \]

- **M**: Mixtures of parts
- **\( \alpha \)**: Unary template for part with mixture
- **\( \beta \)**: Pairwise springs between part with mixture and part with mixture
Co-occurrence “Prior”

$S(M) = \sum_{i,j \in E} b_{ij}^{m_i m_j}$

- Pairwise co-occurrence prior between part with mixture and part with mixture
Inference & Learning

\[
\text{Inference: } \max_{L,M} S(I,L,M)
\]

For a tree graph \((V,E)\): dynamic programming
### Inference & Learning

<table>
<thead>
<tr>
<th>Inference</th>
<th>max_{L,M} S(I,L,M)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>For a tree graph (V,E): dynamic programming</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Learning</th>
<th>min_{w} \frac{1}{2} |w|</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>s.t. \forall n \in \text{pos} \ w \cdot \phi(I\downarrow n, z\downarrow n) \geq 1</td>
</tr>
</tbody>
</table>

Given labeled positive \{I\downarrow n, L\downarrow n, M\downarrow n\} and negative \{I\downarrow n\},

write \( z\downarrow n = (L\downarrow n, M\downarrow n) \), and \( S(I, z) = w \cdot \phi(I, z) \)

\( \forall n \in \text{pos} \quad w \cdot \phi(I\downarrow n, z\downarrow n) \geq 1 \)
## Benchmark Datasets

<table>
<thead>
<tr>
<th>PARSE Full-body</th>
<th>BUFFY Upper-body</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="http://www.ics.uci.edu/~dramanan/papers/parse/index.html" alt="PARSE Full-body" /></td>
<td><img src="http://www.robots.ox.ac.uk/~vgg/data/stickmen/index.html" alt="BUFFY Upper-body" /></td>
</tr>
</tbody>
</table>

**PARSE Full-body**

[Website Link](http://www.ics.uci.edu/~dramanan/papers/parse/index.html)

**BUFFY Upper-body**

[Website Link](http://www.robots.ox.ac.uk/~vgg/data/stickmen/index.html)
How to Get Part Mixtures?

Solution:
Cluster relative locations of joints w.r.t. parents
Articulation

$K$ parts, $M$ mixtures $\Rightarrow K^\uparrow M$ unique pictorial structures

Not all are equally likely --- “prior” given by $S(M)$
Qualitative Results
Diagnosis

Performance vs number of types per part

- 14 parts (joints) vs 27 parts (joints + midpoints)
- More parts and types/mixtures help
Quantitative Results

% of correctly localized limbs

<table>
<thead>
<tr>
<th>Method</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ramanan 2007</td>
<td>27.2</td>
</tr>
<tr>
<td>Andrikluka 2009</td>
<td>55.2</td>
</tr>
<tr>
<td>Johnson 2010a</td>
<td>56.4</td>
</tr>
<tr>
<td>Singh 2010</td>
<td>60.9</td>
</tr>
<tr>
<td>Johnson 2010b</td>
<td>66.2</td>
</tr>
<tr>
<td>Our Model</td>
<td>74.9</td>
</tr>
</tbody>
</table>

All previous work use explicitly articulated models
### Quantitative Results

% of correctly localized limbs

<table>
<thead>
<tr>
<th>Method</th>
<th>Head</th>
<th>Torso</th>
<th>U. Legs</th>
<th>L. Legs</th>
<th>U. Arms</th>
<th>L. Arms</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ramanan 2007</td>
<td>52.1</td>
<td>37.5</td>
<td>31.0</td>
<td>29.0</td>
<td>17.5</td>
<td>13.6</td>
<td>27.2</td>
</tr>
<tr>
<td>Andrikluka 2009</td>
<td>81.4</td>
<td>75.6</td>
<td>63.2</td>
<td>55.1</td>
<td>47.6</td>
<td>31.7</td>
<td>55.2</td>
</tr>
<tr>
<td>Johnson 2010a</td>
<td>77.6</td>
<td>68.8</td>
<td>61.5</td>
<td>54.9</td>
<td>53.2</td>
<td>39.3</td>
<td>56.4</td>
</tr>
<tr>
<td>Singh 2010</td>
<td>91.2</td>
<td>76.6</td>
<td>71.5</td>
<td>64.9</td>
<td>50.0</td>
<td>34.2</td>
<td>60.9</td>
</tr>
<tr>
<td>Johnson 2010b</td>
<td>85.4</td>
<td>76.1</td>
<td>73.4</td>
<td>65.4</td>
<td>64.7</td>
<td>46.9</td>
<td>66.2</td>
</tr>
<tr>
<td>Our Model</td>
<td><strong>97.6</strong></td>
<td><strong>93.2</strong></td>
<td><strong>83.9</strong></td>
<td><strong>75.1</strong></td>
<td><strong>72.0</strong></td>
<td><strong>48.3</strong></td>
<td><strong>74.9</strong></td>
</tr>
</tbody>
</table>

1 second per image
### Quantitative Results

% of correctly localized limbs

#### Subset of Buffy Testset

<table>
<thead>
<tr>
<th>Method</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tran 2010</td>
<td>62.3</td>
</tr>
<tr>
<td>Andrikluka 2009</td>
<td>73.5</td>
</tr>
<tr>
<td>Eichner 2009</td>
<td>80.1</td>
</tr>
<tr>
<td>Sapp 2010a</td>
<td>85.9</td>
</tr>
<tr>
<td>Sapp 2010b</td>
<td>85.5</td>
</tr>
<tr>
<td>Our Model</td>
<td>89.1</td>
</tr>
</tbody>
</table>

All previous work use explicitly articulated models
## Quantitative Results

% of correctly localized limbs

<table>
<thead>
<tr>
<th>Method</th>
<th>Head</th>
<th>Torso</th>
<th>U. Arms</th>
<th>L. Arms</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tran 2010</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>62.3</td>
</tr>
<tr>
<td>Andrikluka 2009</td>
<td>90.7</td>
<td>95.5</td>
<td>79.3</td>
<td>41.2</td>
<td>73.5</td>
</tr>
<tr>
<td>Eichner 2009</td>
<td>98.7</td>
<td>97.9</td>
<td>82.8</td>
<td>59.8</td>
<td>80.1</td>
</tr>
<tr>
<td>Sapp 2010a</td>
<td>100</td>
<td>100</td>
<td>91.1</td>
<td>65.7</td>
<td>85.9</td>
</tr>
<tr>
<td>Sapp 2010b</td>
<td>100</td>
<td>96.2</td>
<td>95.3</td>
<td>63.0</td>
<td>85.5</td>
</tr>
<tr>
<td>Our Model</td>
<td>100</td>
<td>99.6</td>
<td>96.6</td>
<td>70.9</td>
<td>89.1</td>
</tr>
</tbody>
</table>

Ours | 5 seconds VS 5 minutes | next best
Human Detection
Conclusion

- Model affine warps with a part-based model
Conclusion

• Model affine warps with a part-based model
• Exponential set of pictorial structures
Conclusion

• Model affine warps with a part-based model
• Exponential set of pictorial structures
• Rigid vs flexible relations
Conclusion

• Model affine warps with a part-based model
• Exponential set of pictorial structures
• Rigid vs flexible relations
• Supervision helps
Further ideas:

Human Pose Estimation Using Consistent Max-Covering, Hao Jiang, ICCV 09

Max-margin hidden conditional random fields for human action recognition, Yang Wang and Greg Mori, CVPR 09

Adaptive pose priors for pictorial structures, B. Sapp, C. Jordan, and B. Taskar, CVPR 10