Deformable Parts Model

Carlo Tomasi

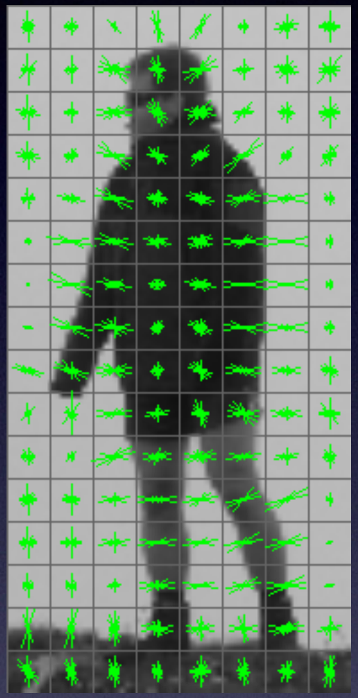
Models for Person Detection

picture by Li, Fergus, Torralba

bag of features:
no shape model
Sivic *et al.* 2003
Csurka *et al.* 2004
…

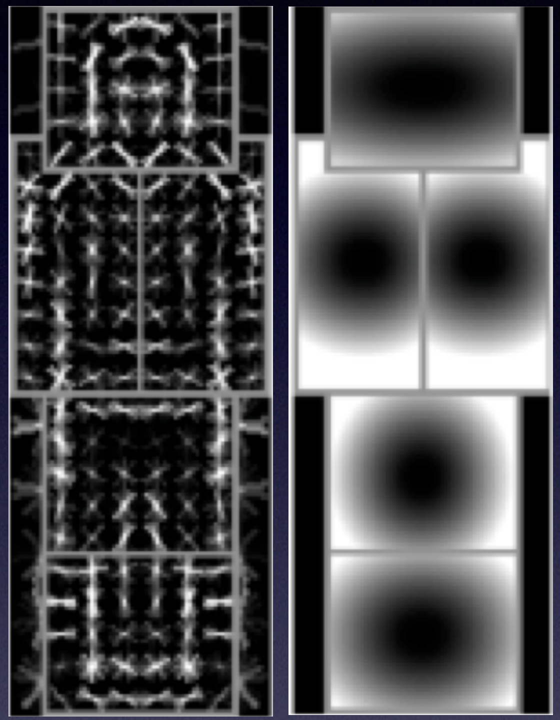
picture by Mekonnen, Lerasle, Herbulot

sparse features:
fixed constellation
of Haar features
Viola & Jones 2001
…



picture by Jürgen Brauer

grid of features:
histograms of
gradients
Dalal & Triggs 2005
…



picture by Felzenszwalb, Girshick, McAllester, Ramanan

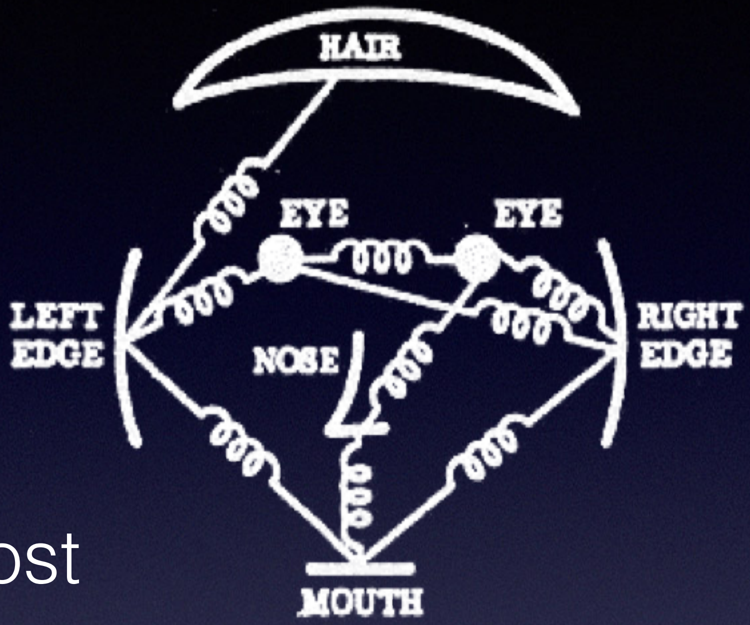
deformable parts:
flexible constellation
of HOG features
Felzenswalb *et al* 2008
…

Trade-Offs

- All are sliding-window methods: expensive but embarrassingly parallel
- Bags of features: general, simple, but no shape
- Sparse features and feature grids: simple, but “people as popsicles”
- Deformable parts: accounts for body articulation, but more expensive to train and run

Deformable Parts Model

- An old idea: Fischler and Elschlager, *Pictorial Structures*, 1973

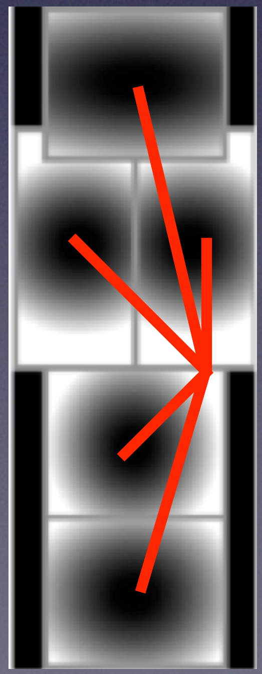


$$\{\hat{\mathbf{x}}_p\} = \arg\max_{\{\mathbf{x}_p\}} \sum_p f(\mathbf{x}_p) - \sum_{p,q} d(\mathbf{x}_p, \mathbf{x}_q)$$

data fit — any pair — deformation cost

- Key difficulty is combinatorially explosive matching complexity during detection

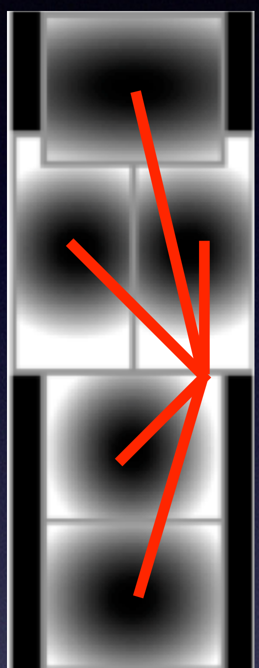
- Solution [Felzenszwalb & Huttenlocher 2000; Felzenszwalb, Girshick, McAllester, Ramanan, 2010]: any pair ⟶ star graph



- Use dynamic programming

Deformable Parts Model

root: the whole window; parts: subwindows



object instance hypothesis

root position hypothesis

part position hypothesis

anchor position of part relative to the root

$$\mathrm{score}(\mathbf{x}_0) = f_0(\mathbf{x}_0) + \sum_{p=1}^{n} \max_{\mathbf{x}_p} [f_p(\mathbf{x}_p) - d_p(\mathbf{x}_p - \mathbf{a}_p(\mathbf{x}_0))]$$

root-specific measure of fit to data

part-specific measure of fit to data

part-specific measure of root-part deformation

orange: model (to be learned)

- object detections are strong local maxima of score($\mathbf{x}_0$)
- corresponding $\mathbf{x}_p$ yield model-to-instance correspondence
- mixtures of DPMs handle large intra-class variations

Fit and Deformation Measures

image features

$$f_p(\mathbf{x}) = \boldsymbol{\beta}_p^T \boldsymbol{\varphi}(\mathbf{x})$$

linear function
of features defined
on an image pyramid

$$\boldsymbol{\eta} = [x - a_p\,,\;\; y - b_p\,,\;\; (x - a_p)^2,\;\; (y - b_p)^2]^T$$

$$d_p(\mathbf{x} - \mathbf{a}_p(\mathbf{x}_0)) = \boldsymbol{\delta}_p^T \boldsymbol{\eta}$$

quadratic function
of part-anchor
displacement

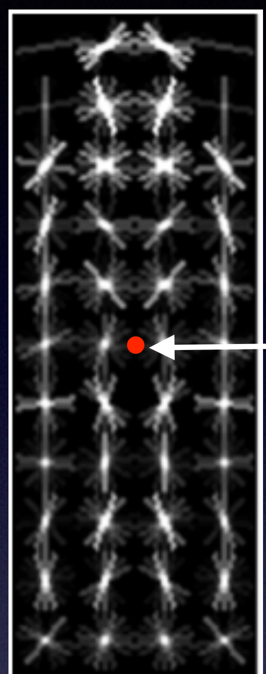
parabola!

Training determines model parameters

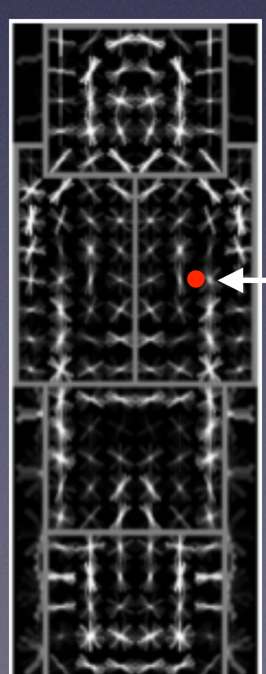
$$\mathbf{w}^T = (\boldsymbol{\beta}_0; \boldsymbol{\beta}_1, \boldsymbol{\delta}_1, \mathbf{a}_1, \ldots, \boldsymbol{\beta}_n, \boldsymbol{\delta}_n, \mathbf{a}_n)$$

$\mathbf{x}_0 = (x_0, y_0, \ell_0)$ and $\mathbf{x}_p = (x_p, y_p, \ell_0 - \lambda)$ for $p \geq 1$ are instance parameters

Features



$\mathbf{x}_0 = (x_0, y_0, \ell_0)$



twice the resolution

$\mathbf{x}_p = (x_p, y_p, \ell_0 - \lambda)$ for $p \geq 1$

- HOG features for both root and parts
  [Dalal & Triggs 2006]
- Part HOGs are one octave finer than root HOG
- Some dimensionality reduction through PCA saves both training and detection complexity
- All features computed on a fine image pyramid for scale sensitivity

Training

part location hypotheses

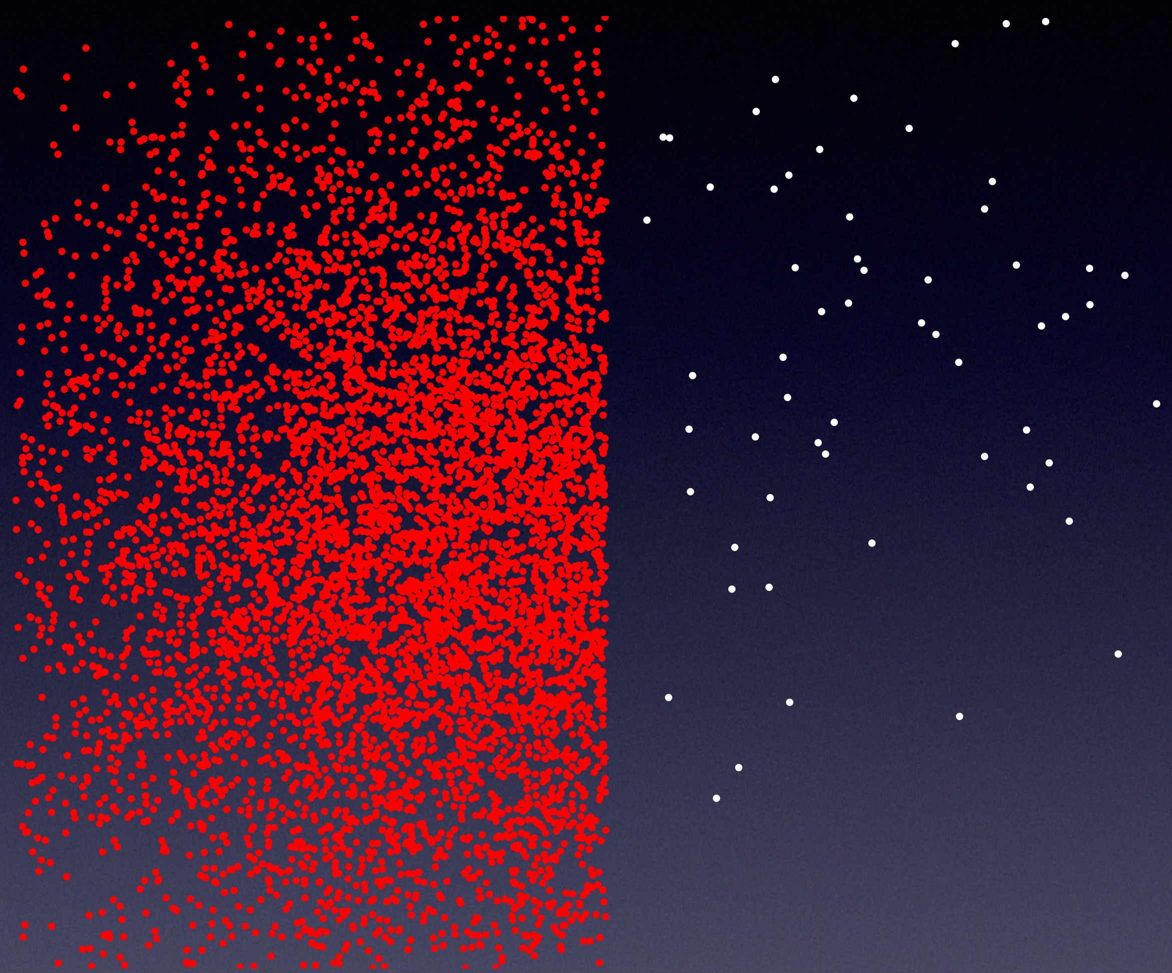
- Standard SVM classifier: $y = \text{sign}(\mathbf{w}^T \mathbf{f} - b)$
- Latent SVM classifier: $y = \text{sign} \max_{\mathbf{x}} [\mathbf{w}^T \mathbf{f}(\mathbf{x}) - b]$
- This new problem leads to nearly the same optimization problem as the standard soft-margin SVM:

$$\arg\min_{\mathbf{w}} \left[\frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^{N} \max\{0, 1 - y_i \max_{\mathbf{x}} [\mathbf{w}^T \mathbf{f}(\mathbf{x}) - b]\} \right]$$

- However, the optimization target is no longer convex
- Use stochastic gradient descent to optimize, but lose global convergence guarantees
- Requires careful initialization

Choosing Negative Examples

- Many more negative than positive examples ("not a person")
  - Using all examples would lead to slow training and many support vectors (slow detection)
  - Random subset would lead to poor representation along boundary
  - Use data mining techniques to choose "hard" negative examples



- start with random subset
- learn classifier
- collect misclassified negatives
- repeat

Detection

$$\text{score}(\mathbf{x}_0) = f_0(\mathbf{x}_0) + \sum_{p=1}^{n} \max_{\mathbf{x}_p}[f_p(\mathbf{x}_p) - d_p(\mathbf{x}_p - \mathbf{a}_p(\mathbf{x}_0))]$$

$\forall \mathbf{x}$ in the pyramid, compute $\boldsymbol{\varphi}(\mathbf{x})$

$\forall p \in \{0, \ldots, n\}$, $\forall \mathbf{x}$ in the pyramid, compute $f_p(\mathbf{x}) = \boldsymbol{\beta}_p^T \boldsymbol{\varphi}(\mathbf{x})$

$\forall p \in \{1, \ldots, n\}$, $\forall \mathbf{a}$ in the pyramid, compute $D_p(\mathbf{a}) = \max_{\mathbf{x}}[f_p(\mathbf{x}) - d_p(\mathbf{x} - \mathbf{a})]$

$\forall \mathbf{x}_0$ in the pyramid, compute $\text{score}(\mathbf{x}_0) = f_0(\mathbf{x}_0) + \sum_{p=1}^{n} D_p(\mathbf{a}_p(\mathbf{x}_0))$

select high-scoring root positions by non-maximum suppression

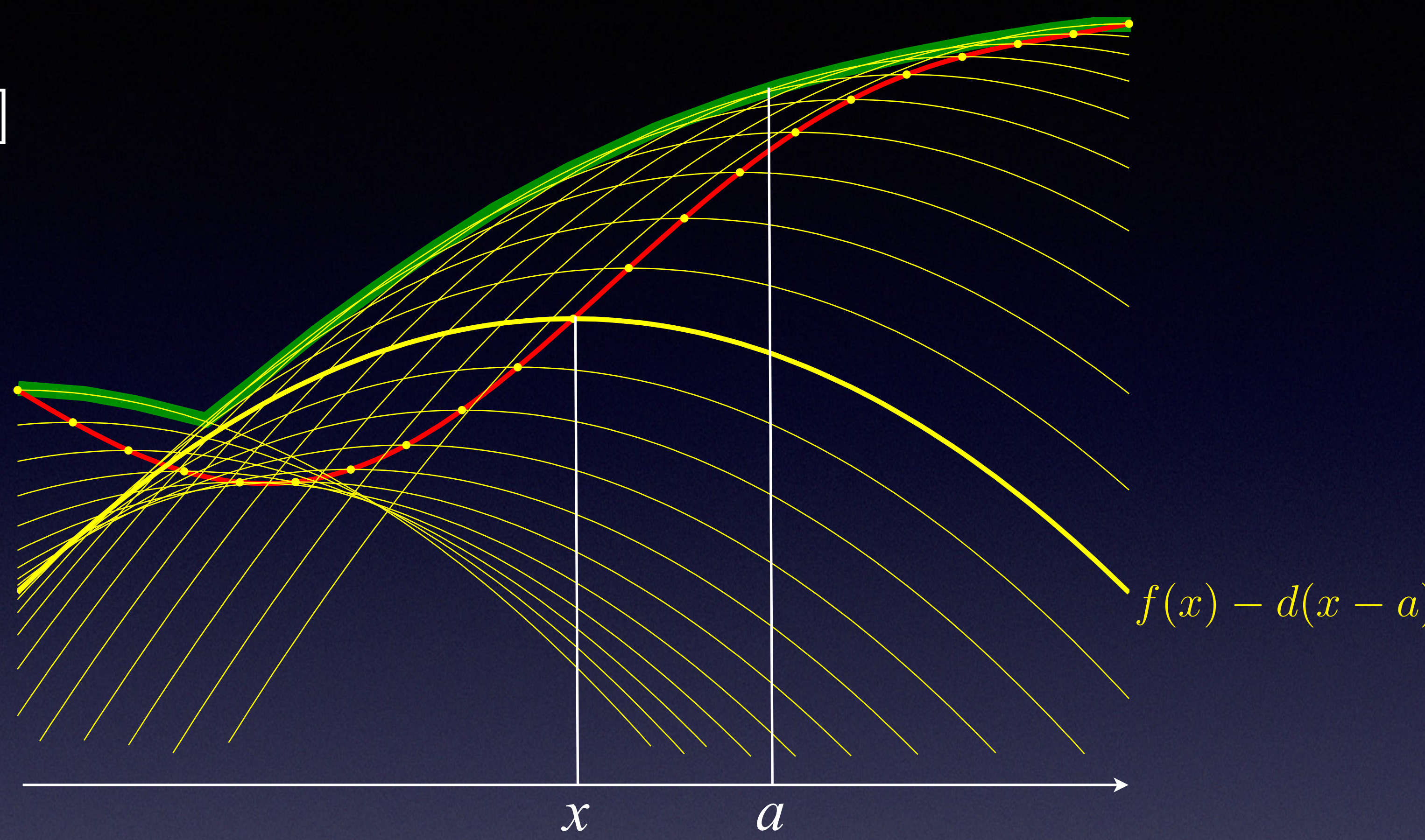
Generalized Distance Transform

[Rosenfeld and Pfaltz 1966,
Felzenszwalb and Huttenlocher 2004]

One dimension:

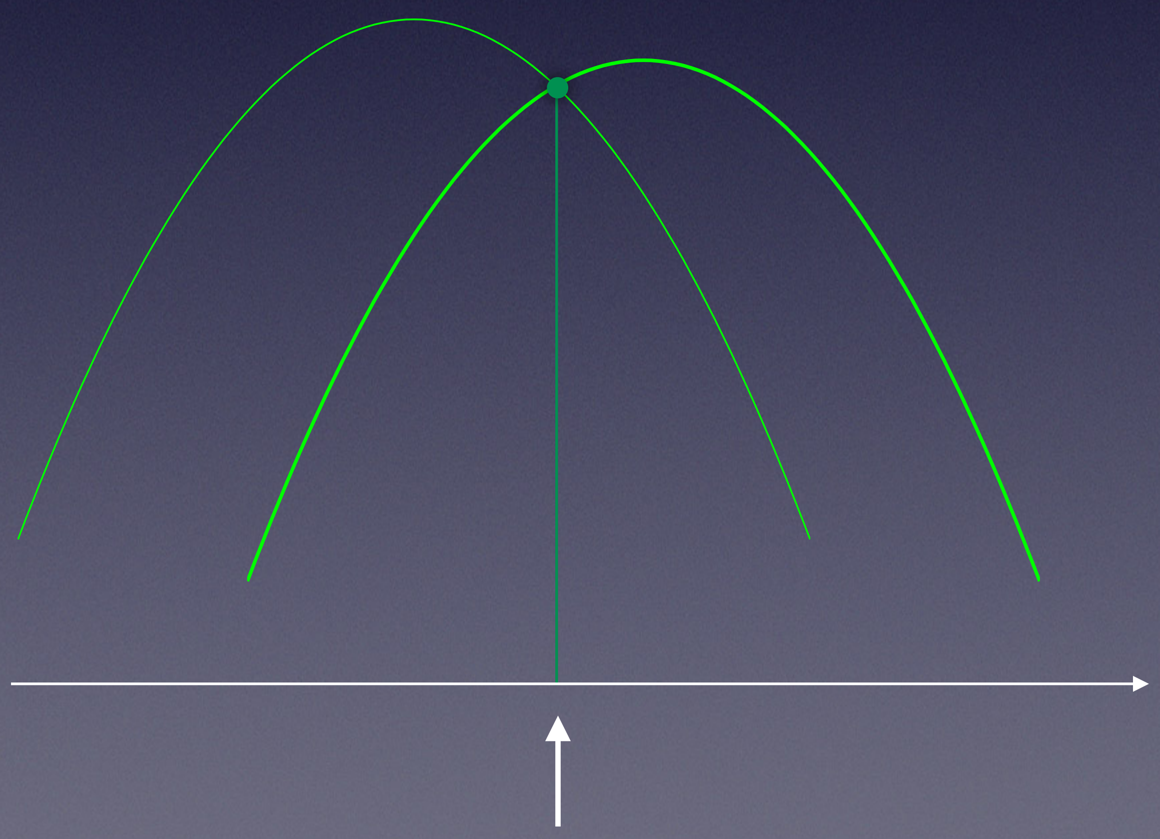
$$D(a) = \max_x [f(x) - d(x-a)]$$

A single left-to-right sweep does the job:



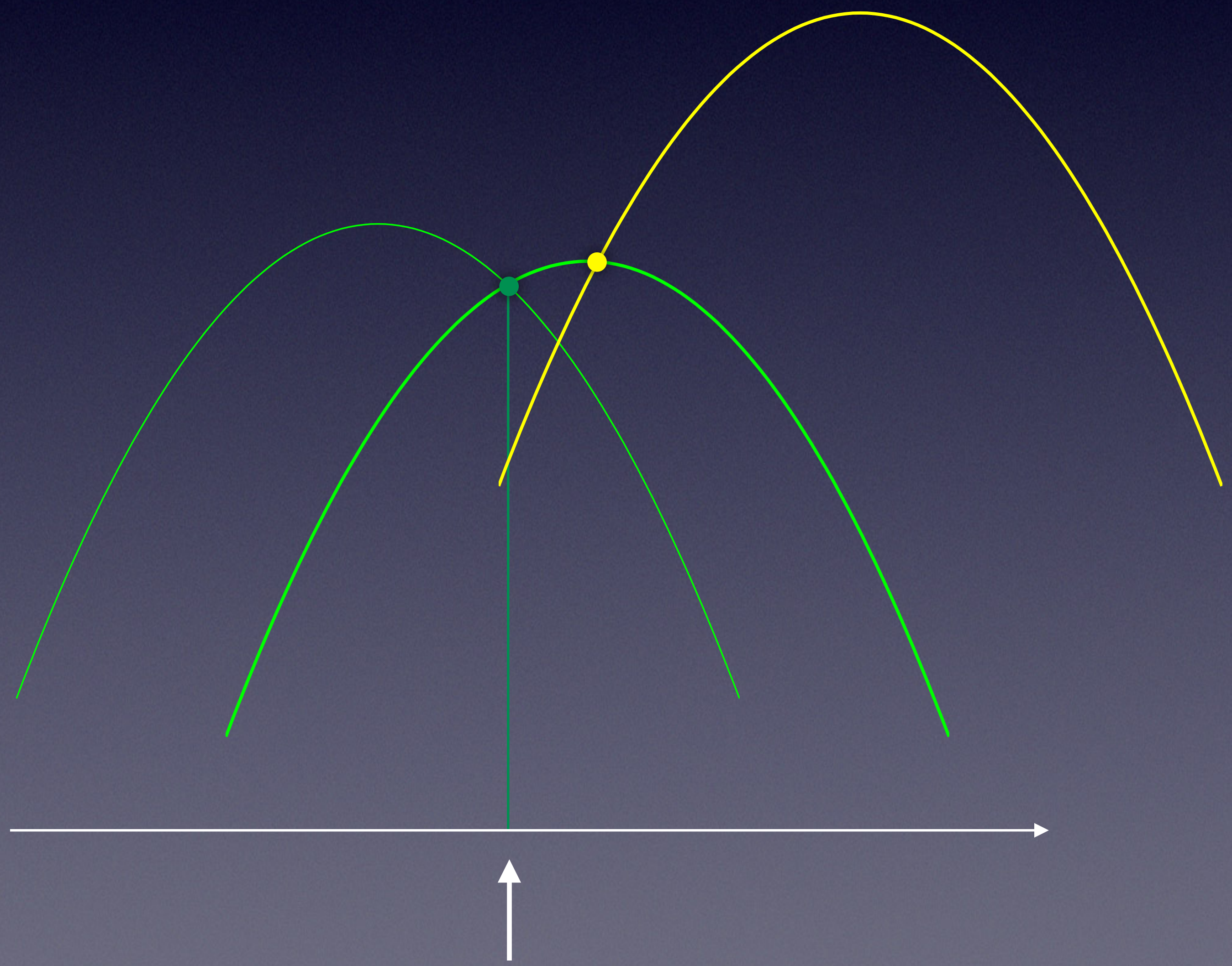
- Not all parabolas touch the upper envelope
- Examine parabolas one at a time, left to right
- Keep track of whether and where the new parabola intersects the last parabola in the envelope (intersections are solutions of second-degree equations)
- Connect relevant parabola pieces to form upper envelope

Two Cases



last parabola intersection in the current envelope

Two Cases



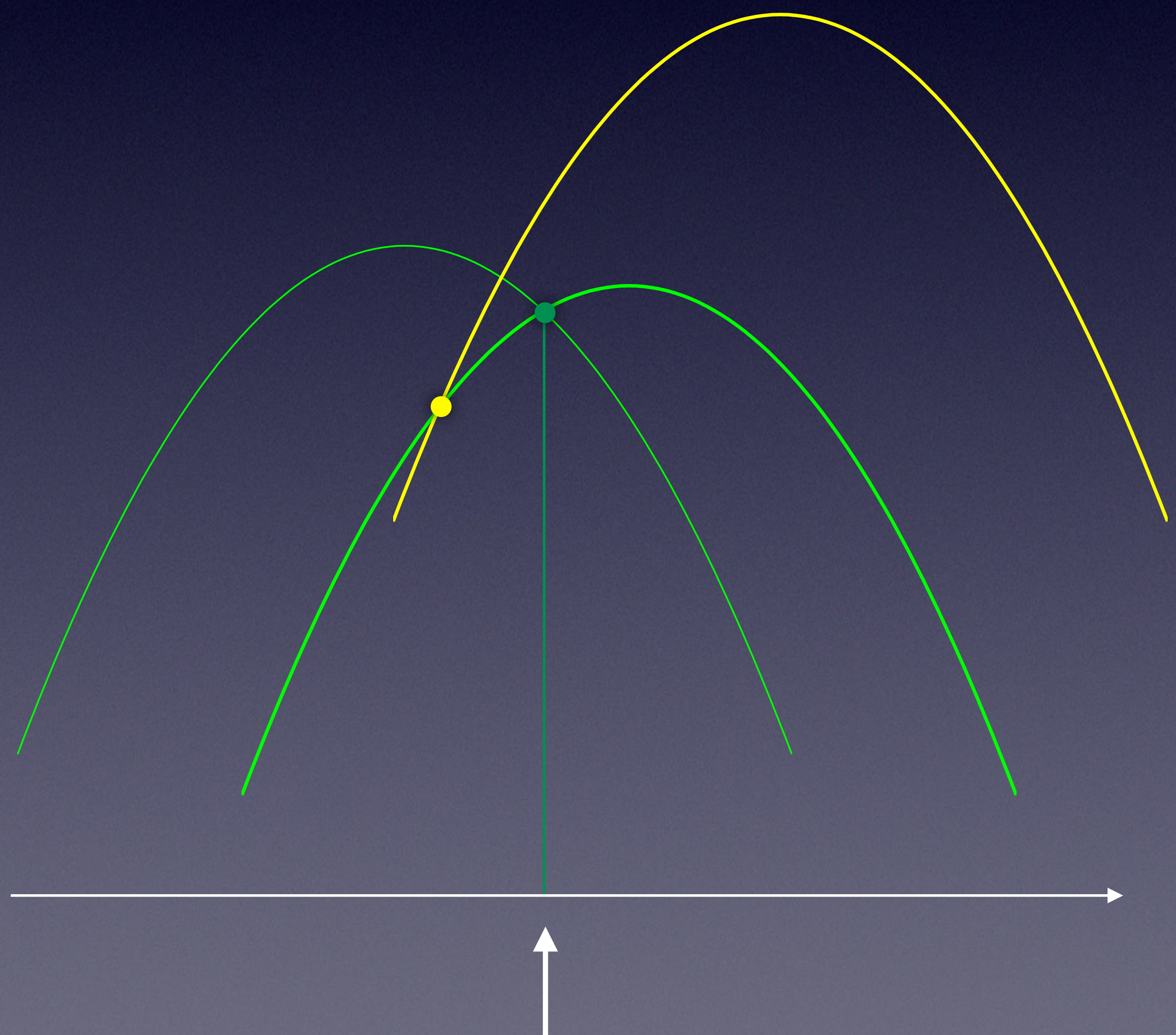
Intersection of new parabola
and last envelope parabola
is to the right of the
last envelope intersection

⬇

add the new parabola
to the envelope

last parabola intersection in the current envelope

Two Cases



Intersection of new parabola and last envelope parabola is to the left of the last envelope intersection

⬇

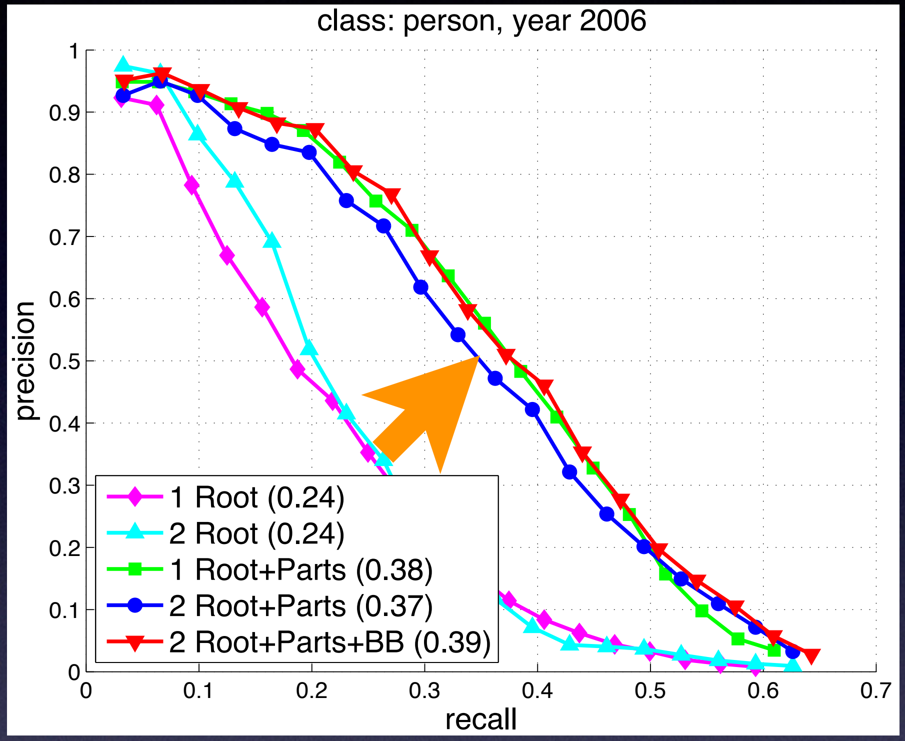
remove the last parabola from the envelope

last parabola intersection in the current envelope

Distance Transform in 2D

$$
\begin{aligned}
& \max_{\mathbf{x}}[f(\mathbf{x}) - d_2(\mathbf{x} - \mathbf{a})] \\
= \; & \max_{(x,y)}[f(x,y) - d(x-a) - d(y-b)] \\
= \; & \max_{x}\{ \overbrace{\max_{y}[f(x,y) - d(y-b)]}^{\text{1D case for each } x} - d(x-a)\} \\
= \; & \underbrace{\max_{x}[D(x,b) - d(x-a)]}_{\text{1D case for each } b}
\end{aligned}
$$

Performance of DPM



Effect of including parts

Average Precision(%) in PASCAL VOC ▶
[DPM implementation, data, and participants change every year]

Year	DPM	Best
2005		12
2006		16
2007		22
2008	27	42
2009	36	43
2010	45	47
2011	46	**52**
2012	46	46

RUNNING TIMES

4 hours to train on typical PASCAL VOC database.
2 seconds per image on standard laptop

◀ Zhang *et* al., Inst. of Automation, Chinese Acad. Science.
Based on DPM, richer part location model, some context information