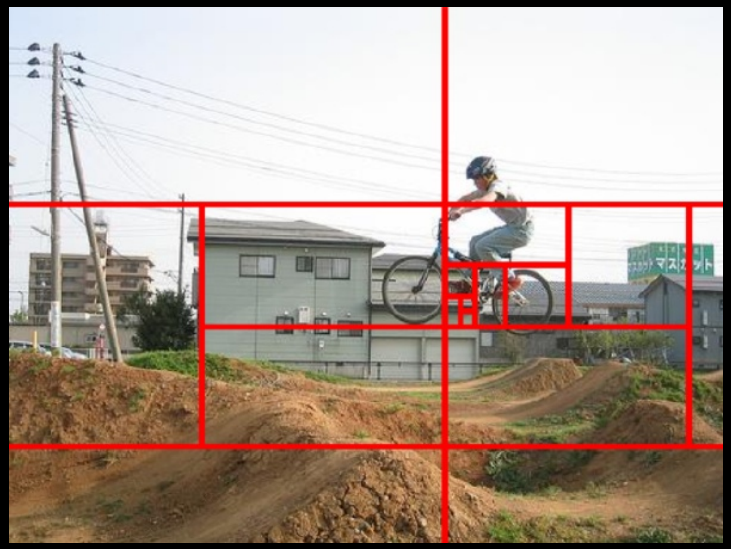
Machine Learning for Computer Vision

MVA – ENS Cachan



Rapid Object Detection with Deformable Part Models

Iasonas Kokkinos

Iasonas.kokkinos@ecp.fr

Center for Visual Computing
Ecole Centrale Paris

Galen Group
INRIA-Saclay

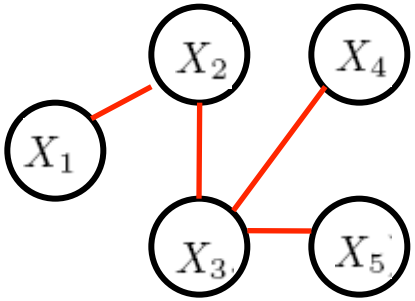
Recap from last lecture

Inference on tree-structured Graphs

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

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

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



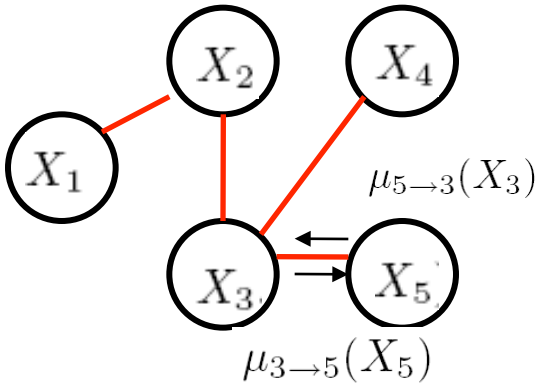
- Brute-force Max-Marginal computation:

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

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

- Exploit factorization

Max-Product algorithm



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

- Message from node i to node j

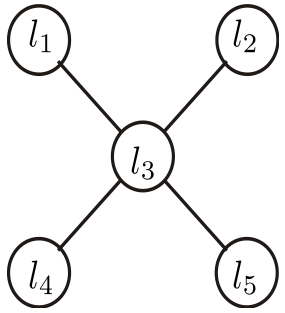
$$\mu_{i\to j}(X_j) \quad = \quad \max_{X_i} \Phi_i(X_i)\Psi_{i,j}(X_i,X_j) \prod_{k\in\mathcal{N}(i)\setminus j} \mu_{k\to j}(X_i)$$

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

Pictorial Structures



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

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

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

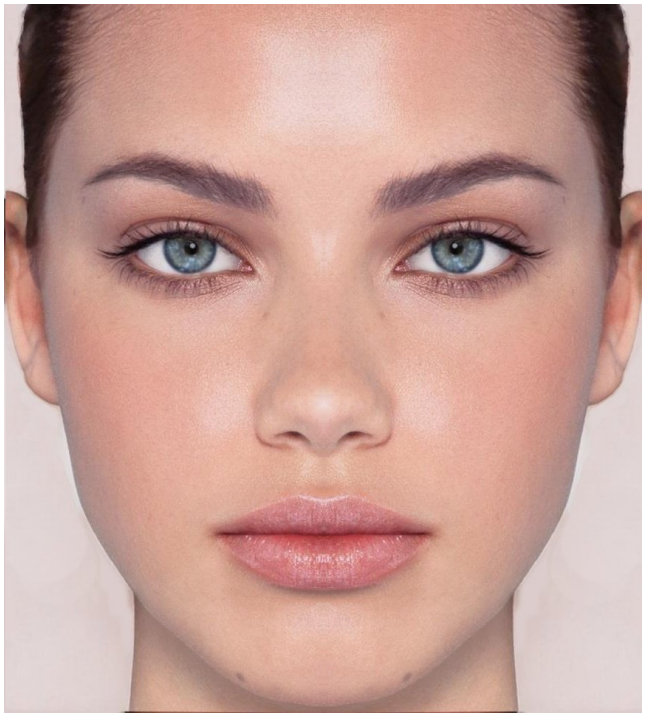
Unary terms **Pairwise terms**

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

- Efficient Implementation: Felzenszwalb & Huttenlocher, CVPR 2001/IJCV 05

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

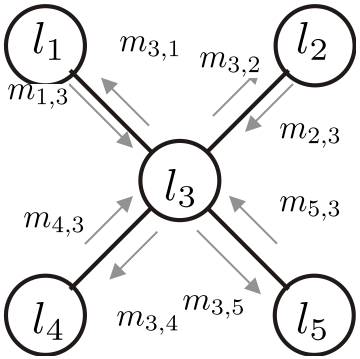
$$B_p(x, x_p) = -(h - h_p - \hat{h}_p)^2 \eta - (v - v_p - \hat{v}_p)^2 \nu$$

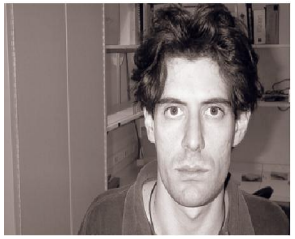
Object score

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

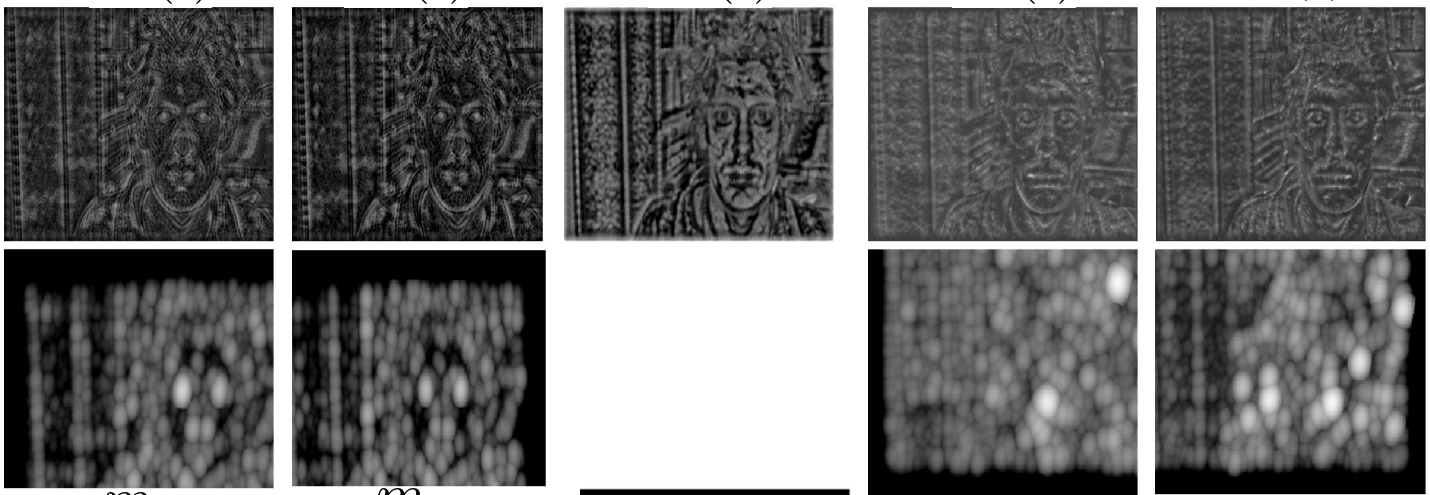
$$\mu_{i\to j}(X_j) = \max_{X_i} \Phi_i(X_i)\Psi_{i,j}(X_i, X_j) \prod_{k\in\mathcal{N}(i)\setminus j} \mu_{k\to j}(X_i)$$

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

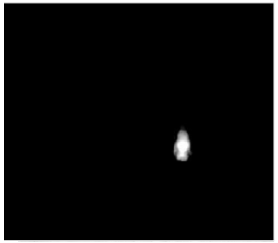




$\Phi_1(l)$ $\Phi_2(l)$ $\Phi_3(l)$ $\Phi_4(l)$ $\Phi_5(l)$

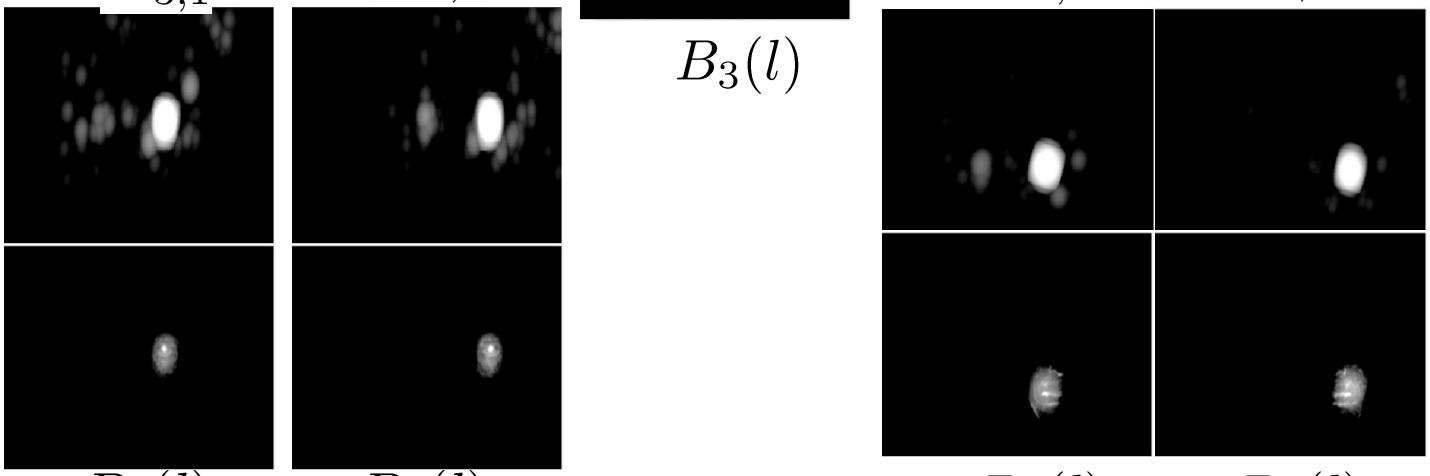


$m_{1,3}$ $m_{2,3}$ $m_{4,3}$ $m_{5,3}$



$B_3(l)$

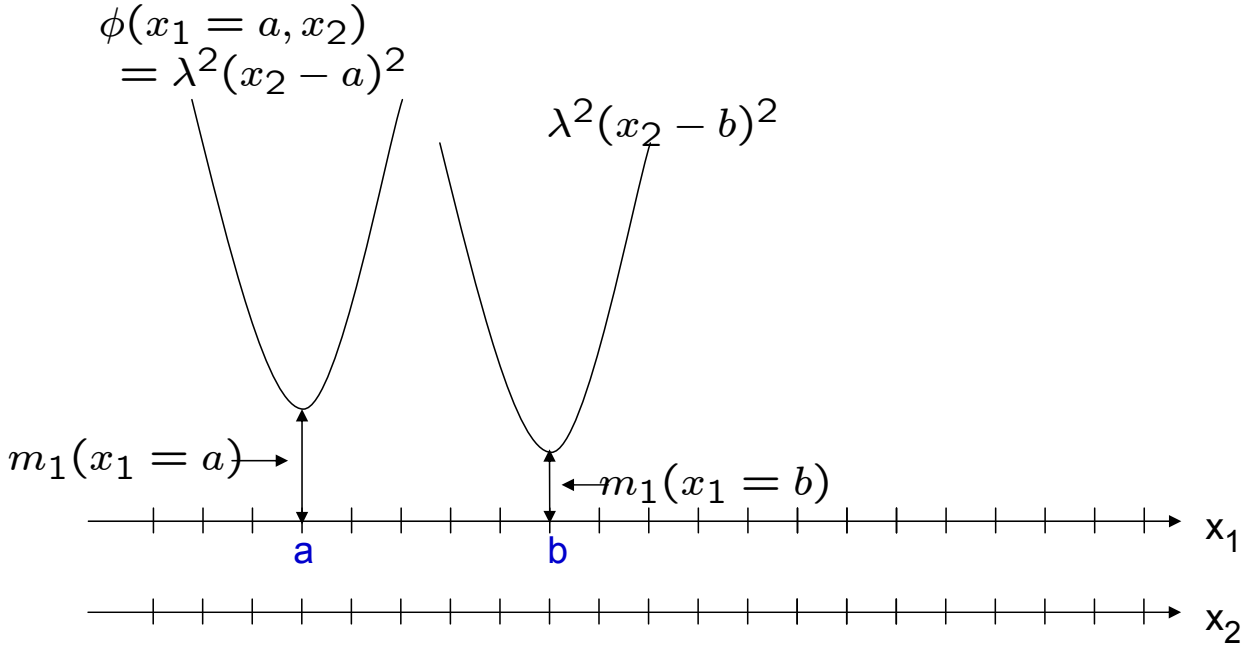
$m_{3,1}$ $m_{3,2}$ $m_{3,4}$ $m_{3,5}$



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

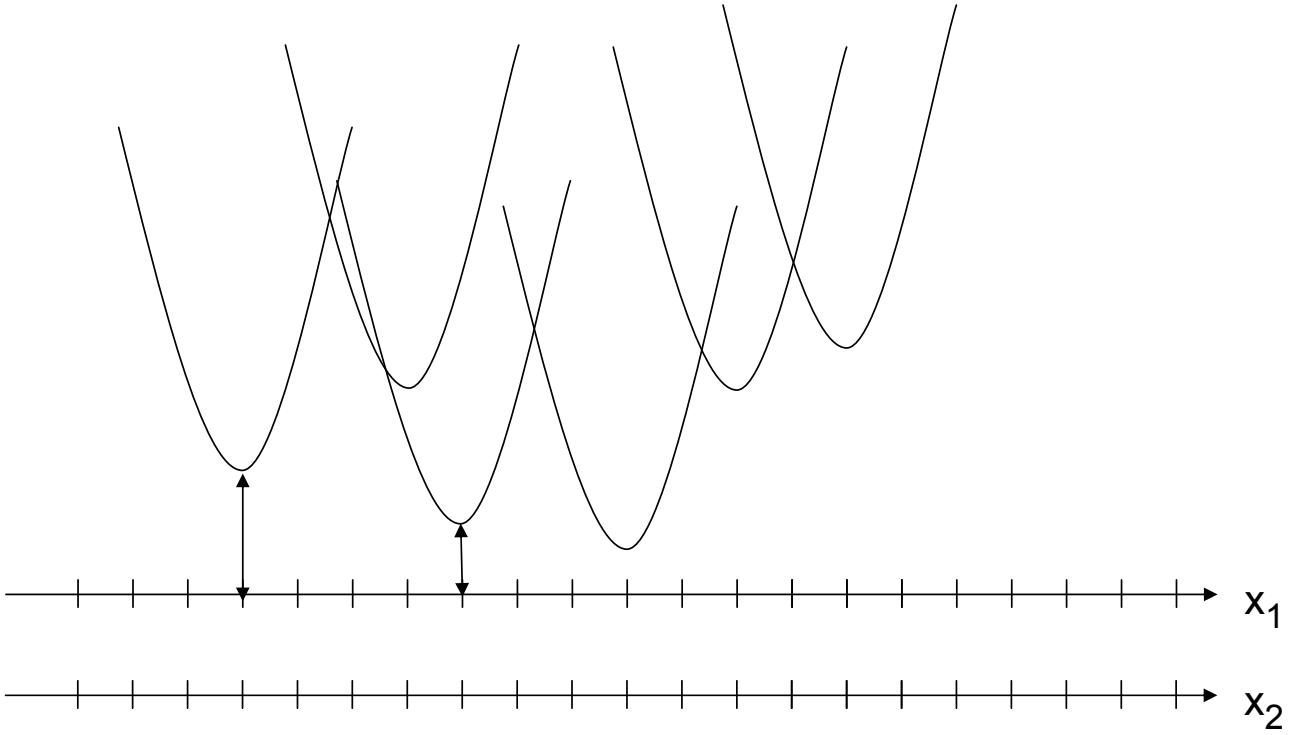
Generalized Distance Transforms- 1D

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



GDT- 1D

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

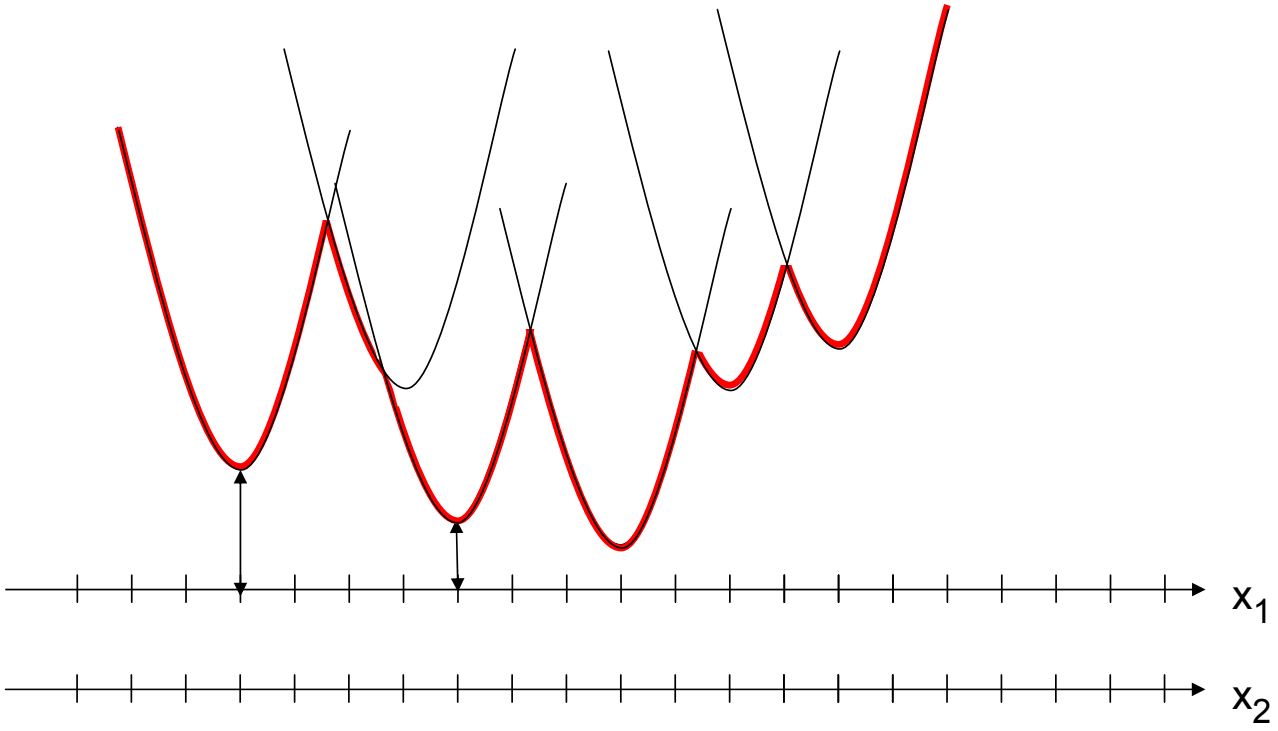


For each x_2

- 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 gen. distance transform

GDT- 1D

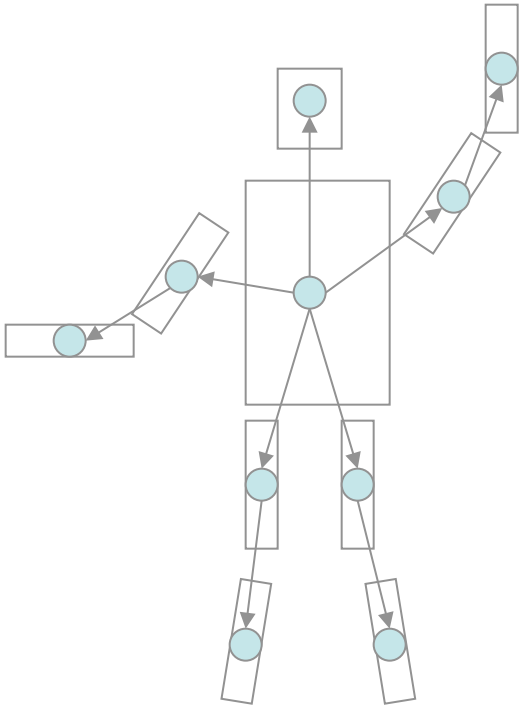
Plot $\min_{x_1}\{m_1(x_1) + \phi(x_1, x_2)\}$ as function of x_2

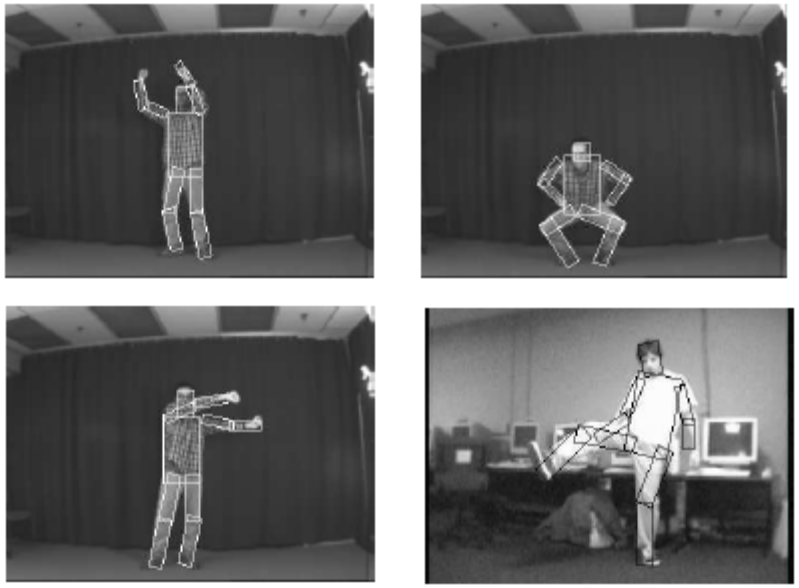


For each x_2

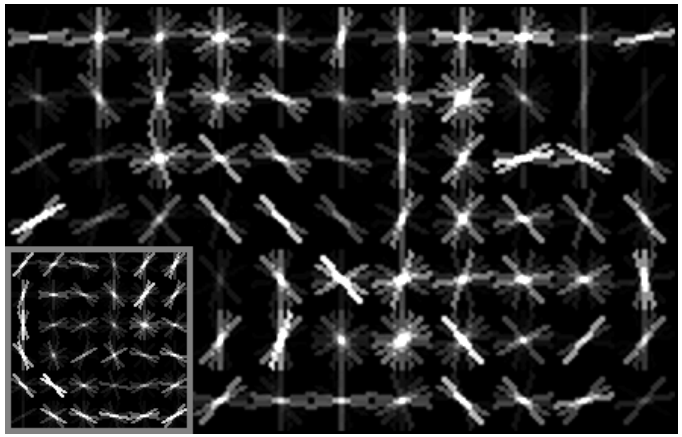
- Finding min over x_1 is equivalent finding minimum over set of offset parabolas
- Lower envelope computed in O(h) rather than $O(h^2)$ via gen. distance transform

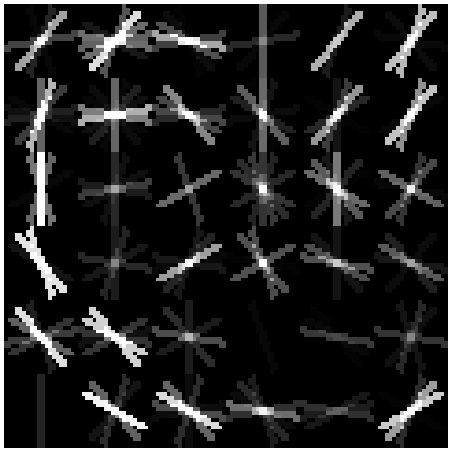
Detection Results



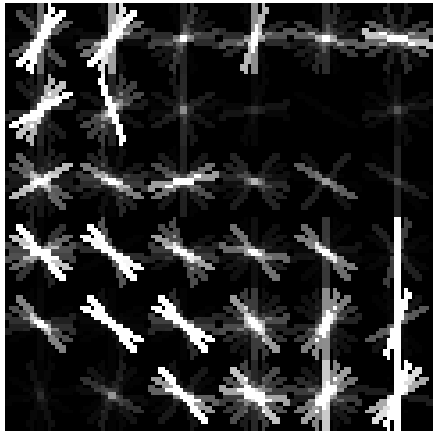


Part score computation





$$\mathbf{w}[y]$$

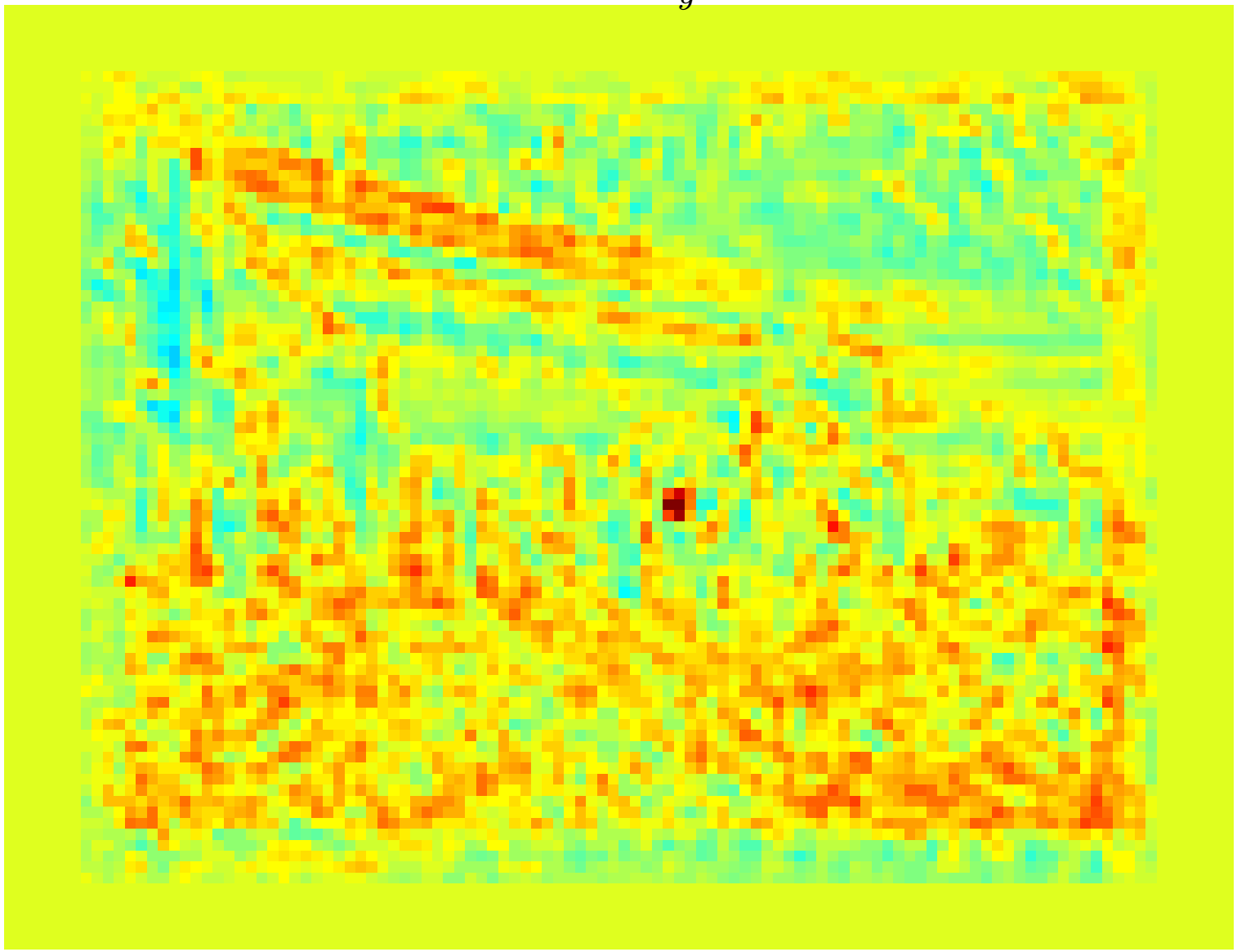


$$\mathbf{h}[x+y]$$

$$s[x] = \sum_{y} \langle \mathbf{h}[x+y], \mathbf{w}[y] \rangle$$

Part score

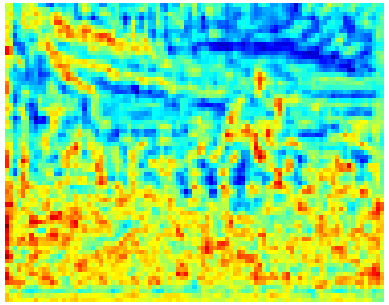
$\mathbf{h}[x]$ $s[x] = \sum_y \langle \mathbf{h}[x+y], \mathbf{w}[y] \rangle$



Object detection with Deformable Part Models (DPMs)

$$U_p(x') = \langle \mathbf{w}_p, \mathbf{H}(x') \rangle$$

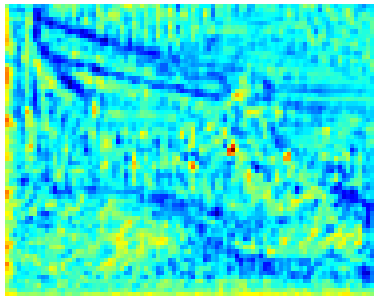
$$\max_{x'} [U_p(x') + B_p(x, x')]$$

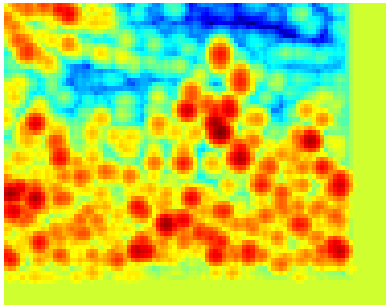


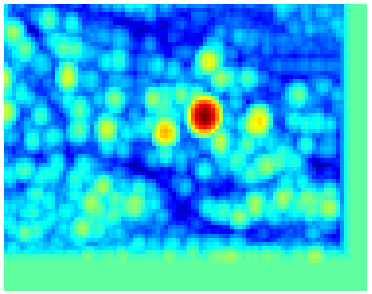
$p = 1$

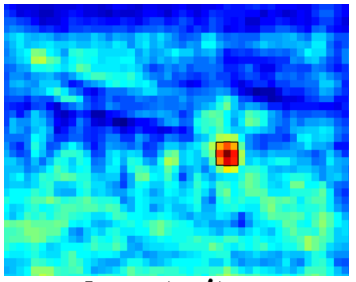
$\vdots$

$p = P$









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

Part-based models + discriminative framework

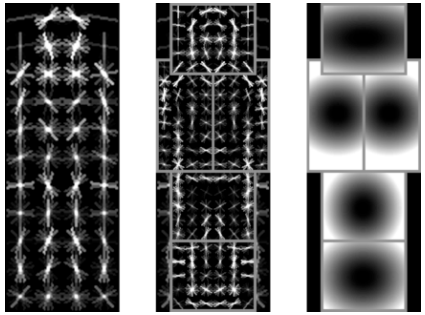
P. Felzenszwalb, D. McAllester, D. Ramanan`A Discriminatively Trained, Multiscale, Deformable Part Model' CVPR 2008

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

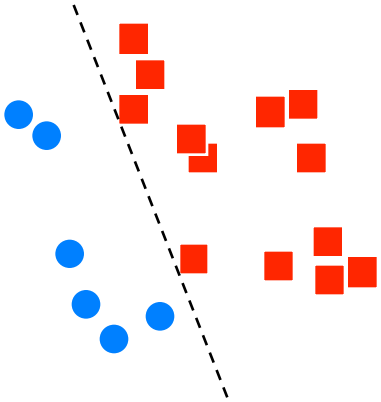
Discriminative Training of DPMs

- Training data consists of images with labeled bounding boxes.
- Need to learn the model structure, filters and deformation costs.

Training

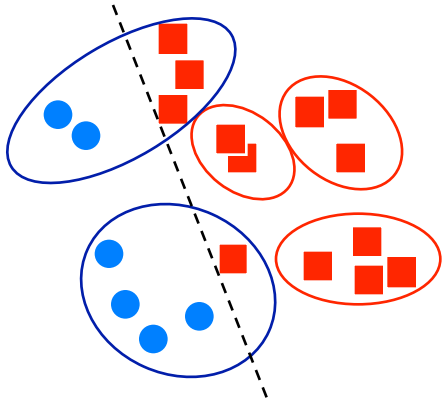


Multiple Instance Learning



Typical Learning

$$S = \{(x^i, y^i)\}$$



Multiple Instance Learning

$$S = \{(\underbrace{\{x^{i,1}, \ldots, x^{i,|B_i|}\}}_{B_i}, y^i)\}$$

$$y^i \in \{0, 1\} \; x^i \in \mathcal{X}$$

$$F : \mathcal{X} \rightarrow \{0, 1\}$$

Positive bag: at least one instance should be positive

Negative bag: no instance should be positive

$$\max_{x \in B_i} F(x) = y^i$$

Slide Credit: K. Grauman

Score function for DPM detection

$$\text{score}(p_0, \ldots, p_n) = \underbrace{\sum_{i=0}^{n} F_i \cdot \phi(H, p_i)}_{\text{“data term”}} - \underbrace{\sum_{i=1}^{n} d_i \cdot (dx_i^2, dy_i^2)}_{\text{“spatial prior”}}$$

filters

displacements

deformation parameters

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

concatenation filters and deformation parameters

concatenation of HOG features and part displacement features

SVM training for DPMs

Classifiers that score an example x using

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

β are model parameters
z are latent values

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

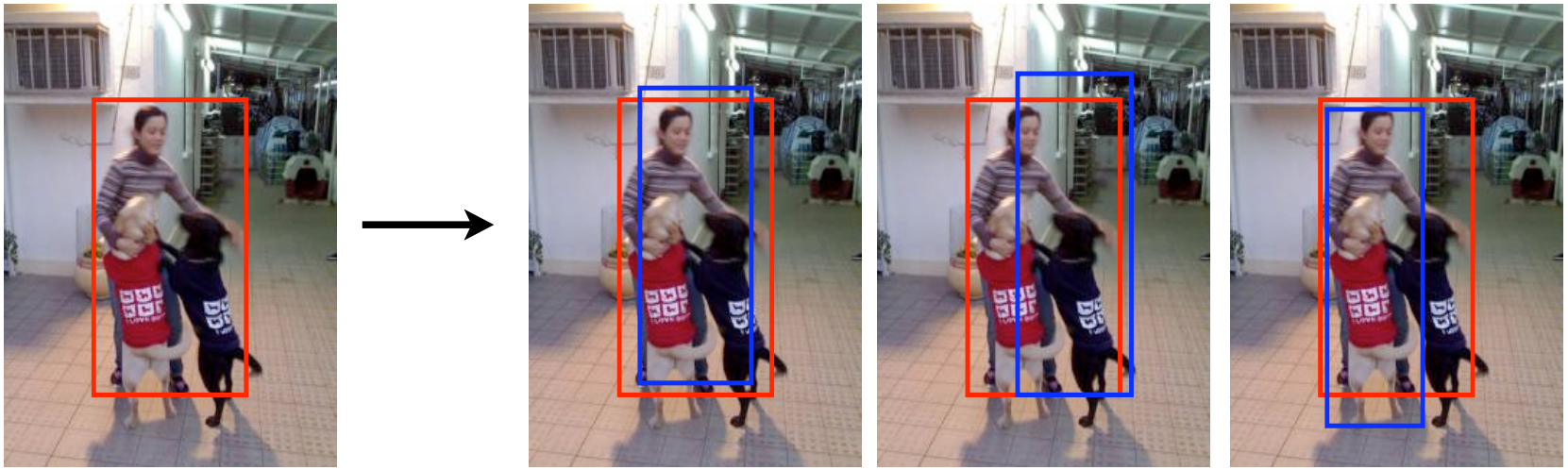
We would like to find β such that: $y_i f_\beta(x_i) > 0$

Minimize

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

Annotation & Latent Variables

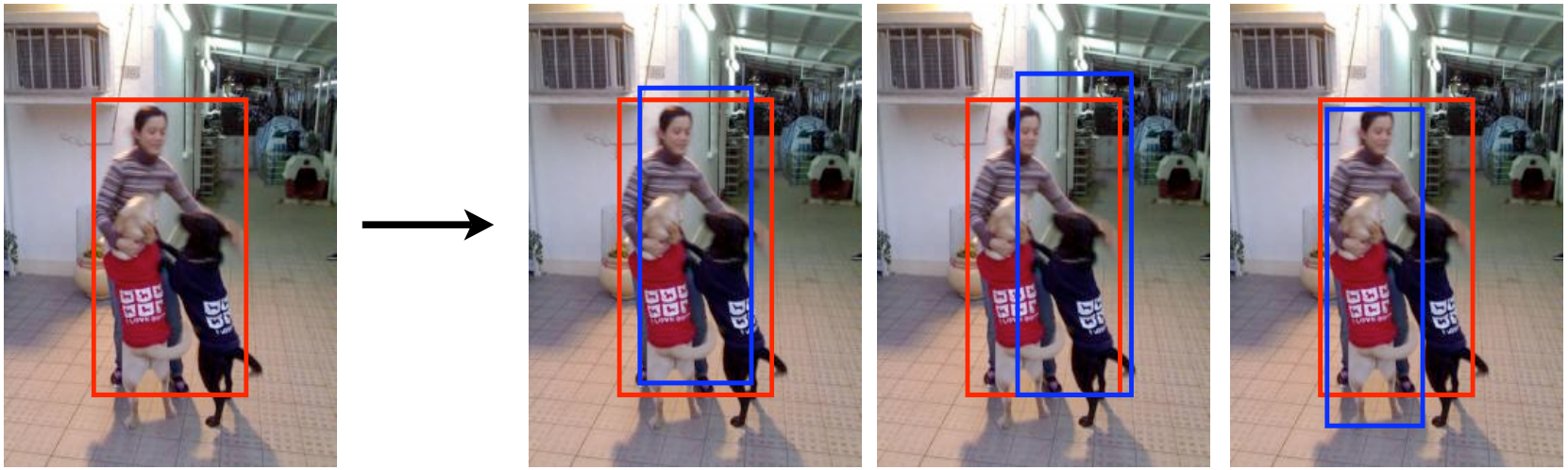
- Positive example specifies some z should have high score
- Bounding box defines range of root locations
  - Parts can be anywhere
  - This defines $Z(x)$



P.F. Felzenszwalb, R. B. Girshick, D. Ramanan and D. A. McAllester, a discriminatively trained multiscale deformable part model, IEEE PAMI, 2010

Annotation & Latent Variables

- Positive example specifies some z should have high score
- Bounding box defines range of root locations
  - Parts can be anywhere
  - This defines $Z(x)$



P.F. Felzenszwalb, R. B. Girshick, D. Ramanan and D. A. McAllester, a discriminatively trained multiscale deformable part model, IEEE PAMI, 2010

Latent SVM training

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

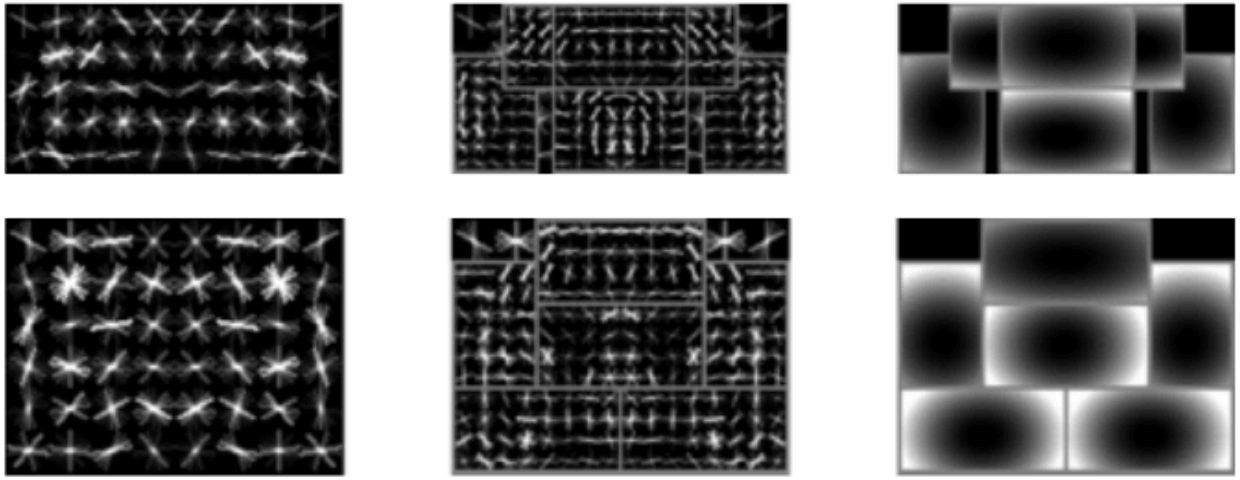
- Convex if we fix z for positive examples
- Optimization:
  - Initialize β and iterate:
    - Pick best z for each positive example
    - Optimize β via gradient descent with data-mining

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 β and iterate:
    - Pick best z for each positive example
    - Optimize β via gradient descent with data-mining

Car model



root filters
coarse resolution

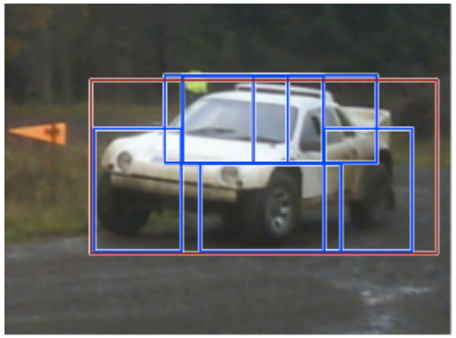
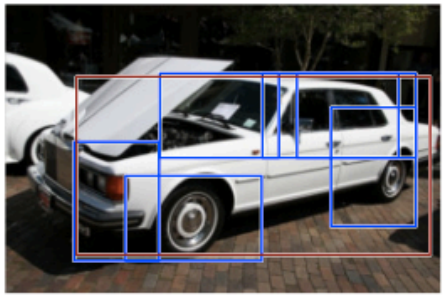
part filters
finer resolution

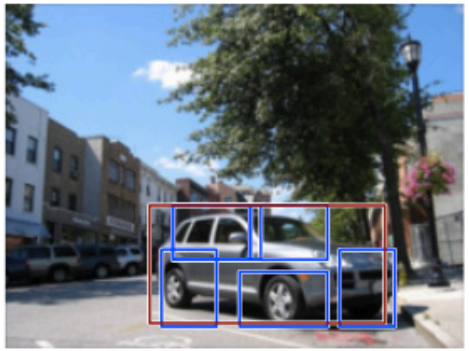
deformation
models

DPM results

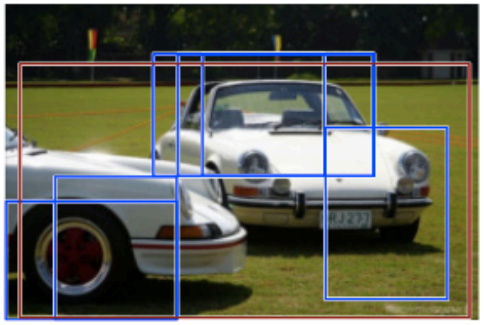
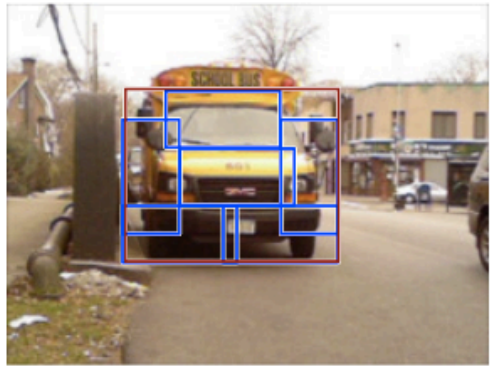
Car detections

high scoring true positives

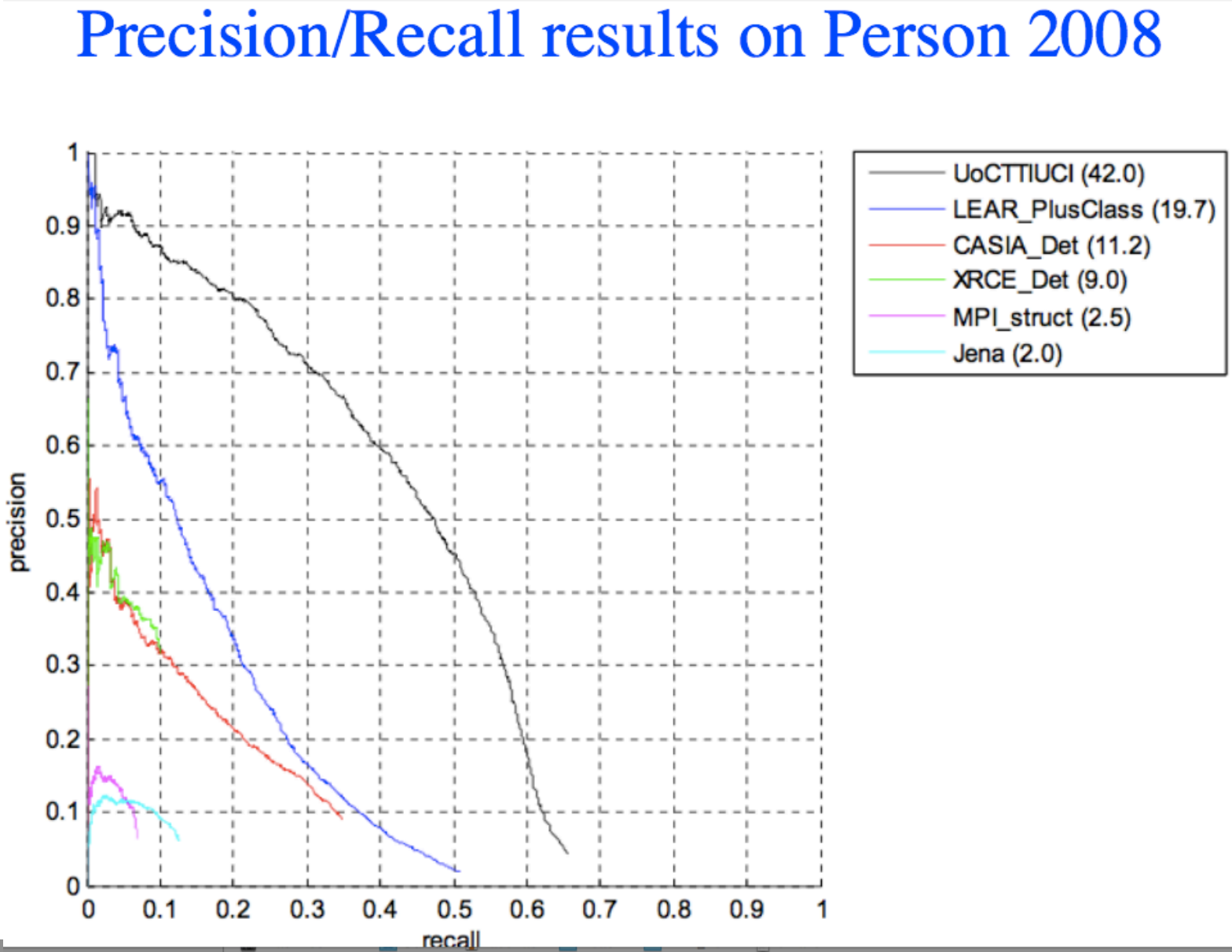





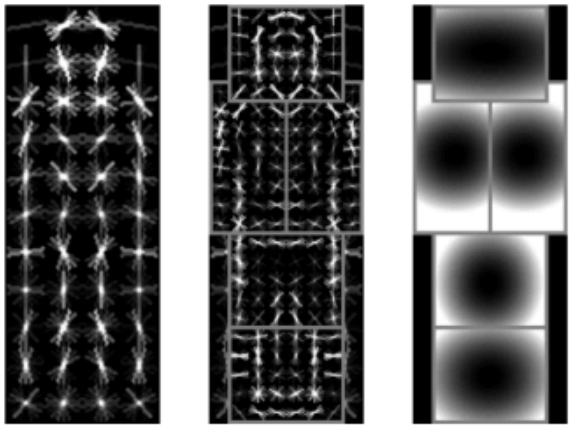
high scoring false positives

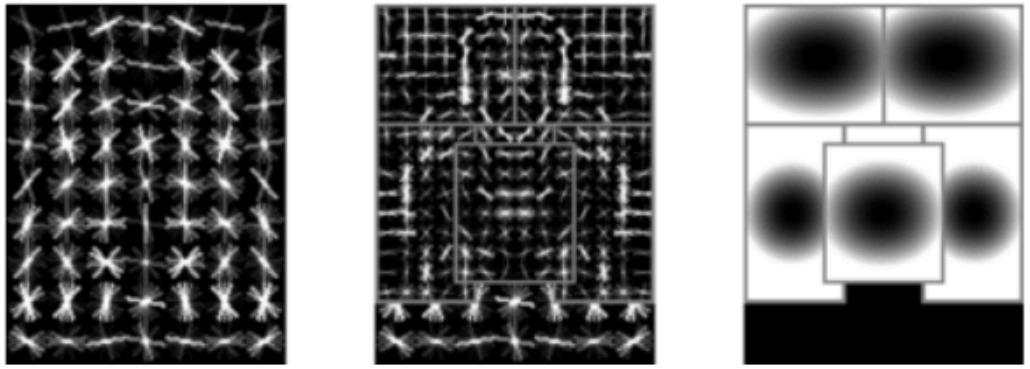



DPM results


Precision/Recall results on Person 2008
precision
recall
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
UoCTTIUCI (42.0)
LEAR_PlusClass (19.7)
CASIA_Det (11.2)
XRCE_Det (9.0)
MPI_struct (2.5)
Jena (2.0)

Person model





root filters
coarse resolution

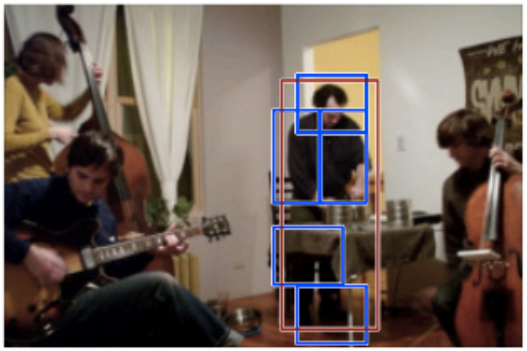
part filters
finer resolution

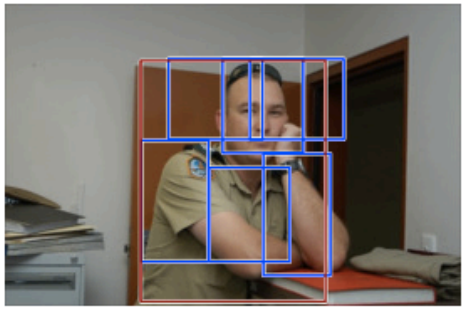
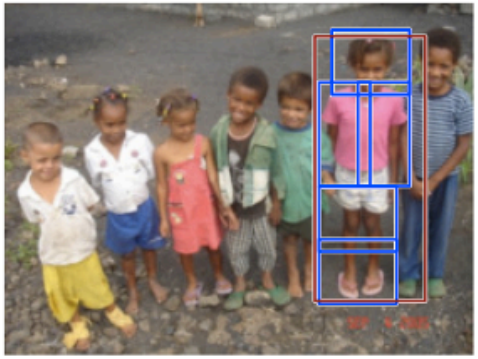
deformation
models

DPM results

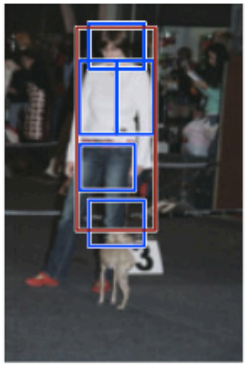
Person detections

high scoring true positives



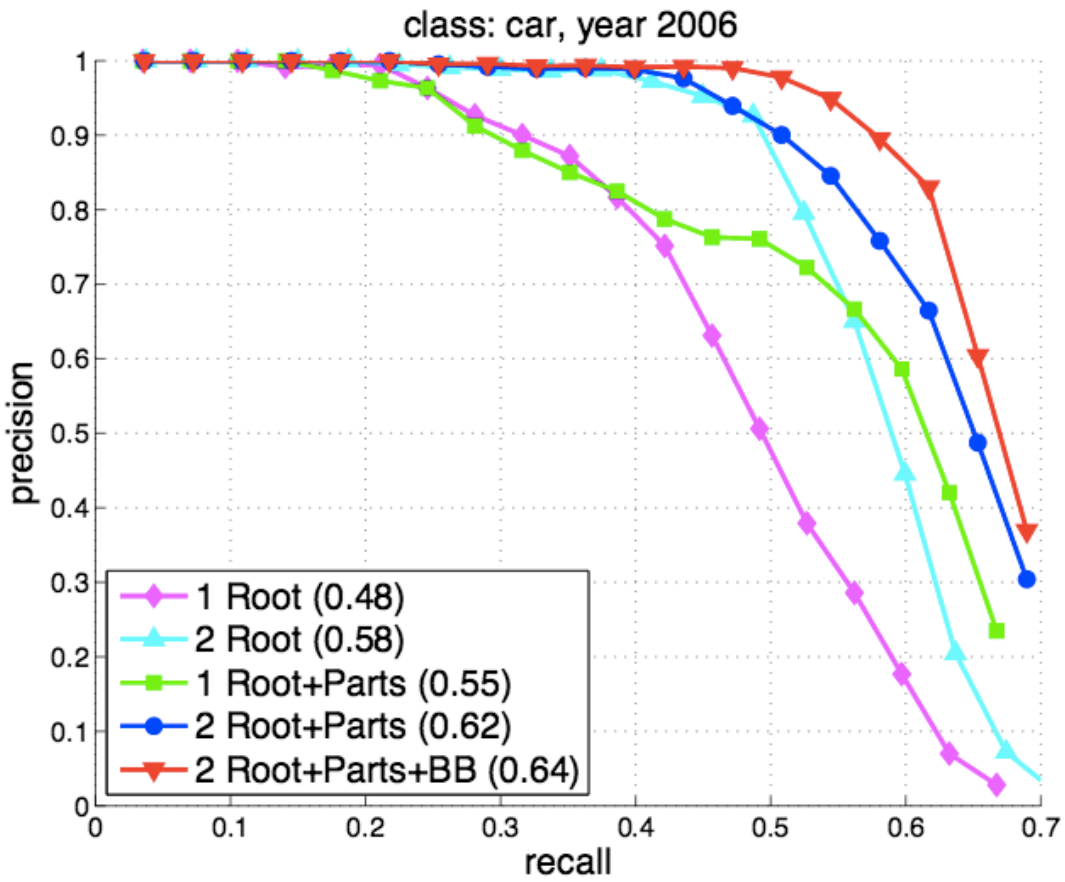



high scoring false positives
(not enough overlap)

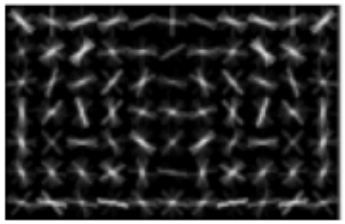
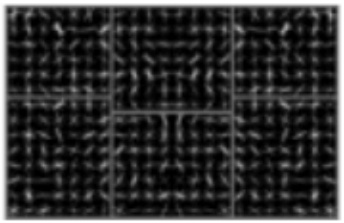
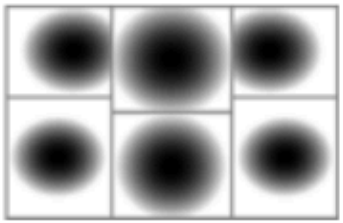
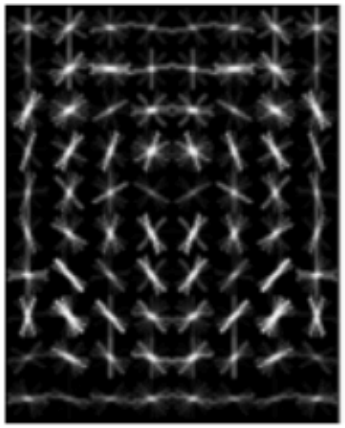
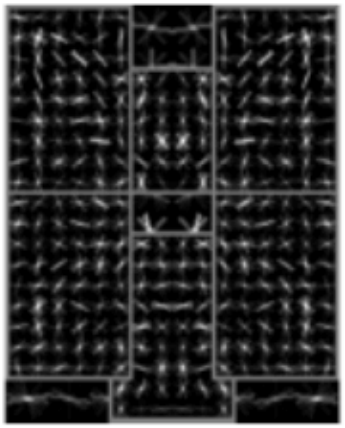
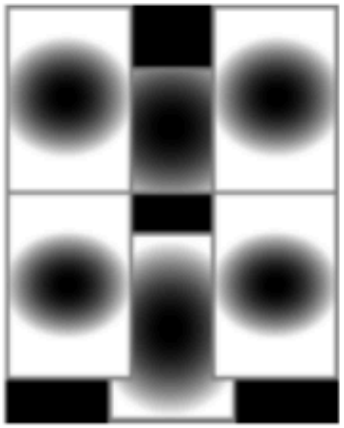



DPM results

Comparison of Car models on 2006 data



Cat model

root filters
coarse resolution

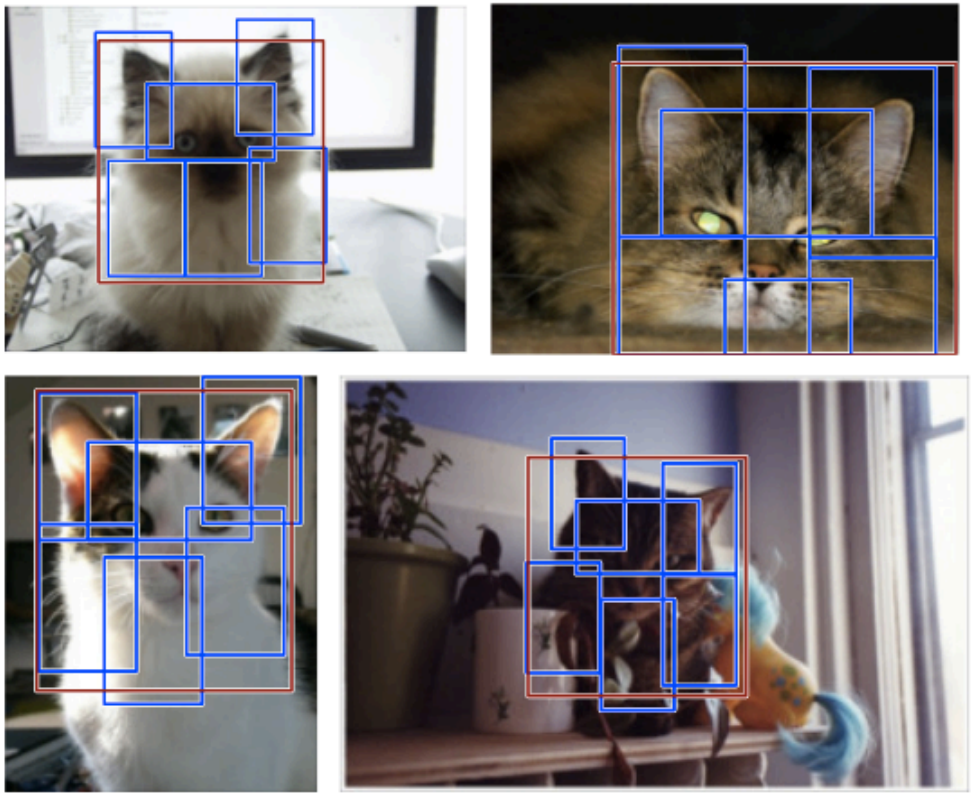
part filters
finer resolution

deformation
models

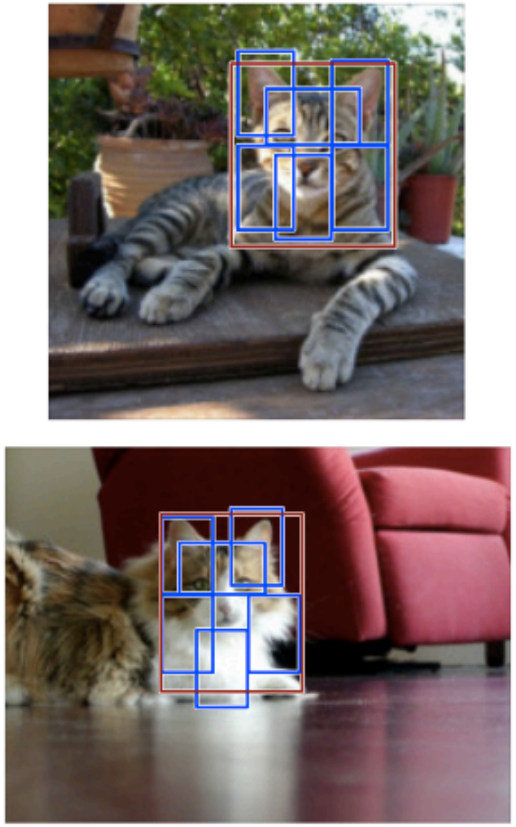
DPM results

Cat detections

high scoring true positives



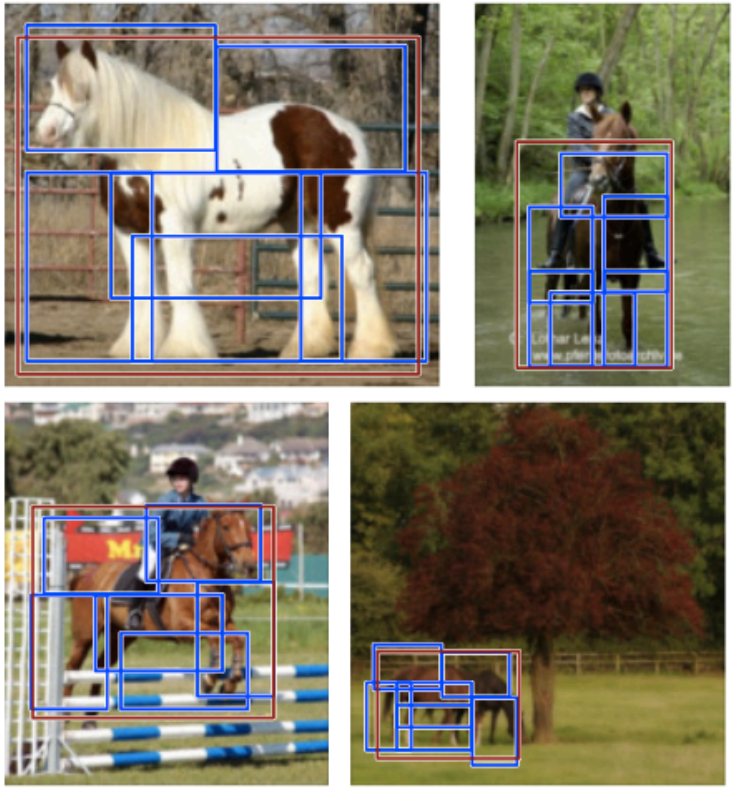
high scoring false positives
(not enough overlap)



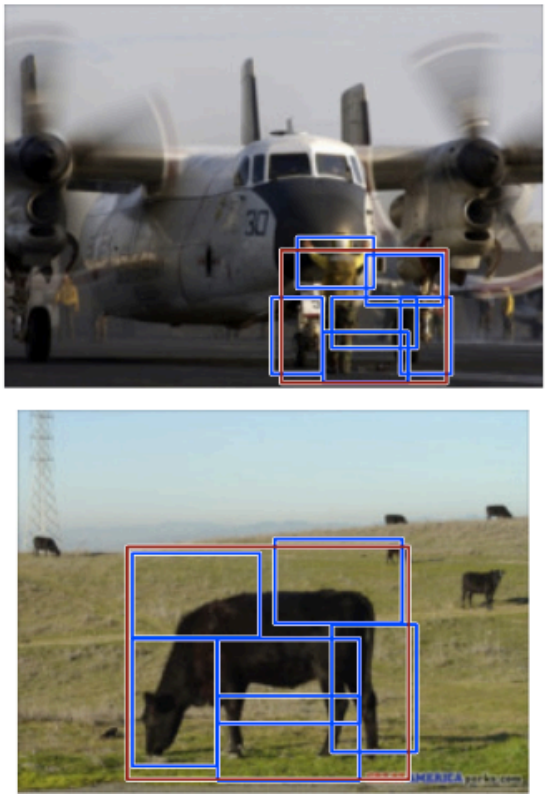
DPM results

Horse detections

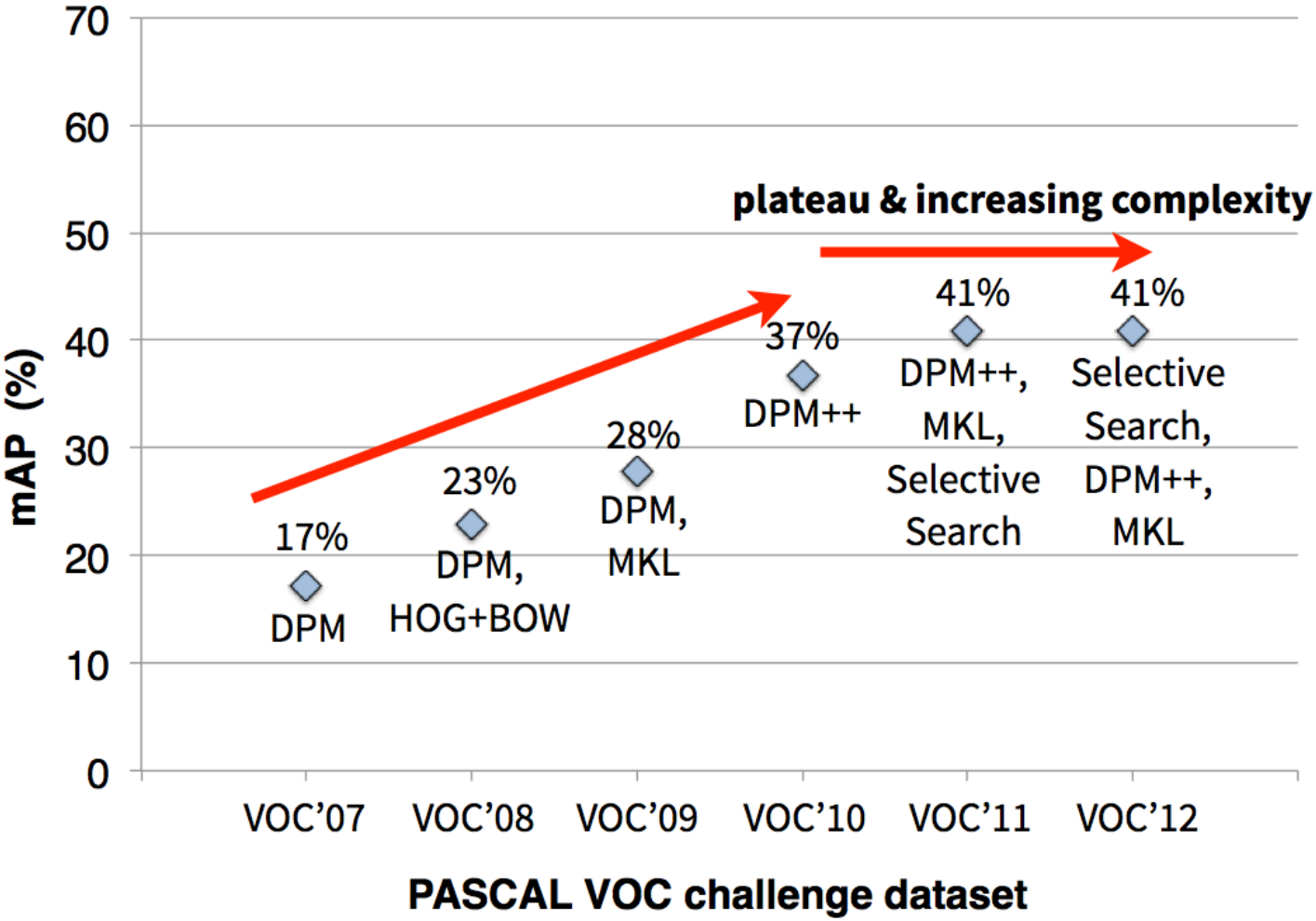
high scoring true positives



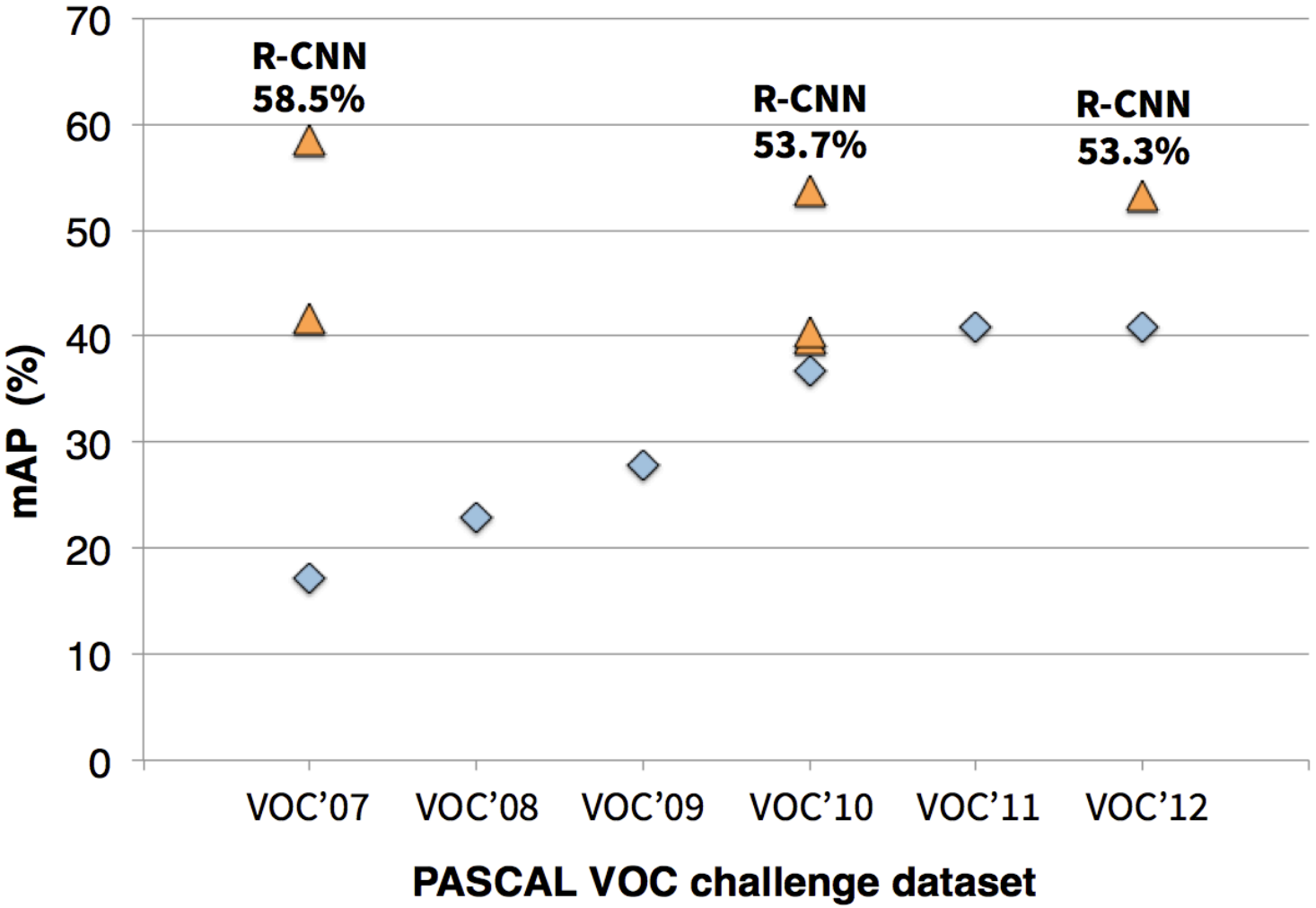
high scoring false positives



Object detection 2010-2012: 'plateau regime'



Object recognition, 2013



A. Krizhevsky, I. Sutskever, and G. Hinton. ImageNet classification with deep convolutional neural networks. *NIPS*13
P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. ICLR 14.
R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *CVPR* 14

DPMs & CNNs (2014)

Deformable Part Models are Convolutional Neural Networks

Ross Girshick[1] Forrest Iandola[2] Trevor Darrell[2] Jitendra Malik[2]
[1]Microsoft Research [2]UC Berkeley
rbg@microsoft.com {forresti,trevor,malik}@eecs.berkeley.edu

Deformable Part Models with CNN Features

Pierre-André Savalle[1], Stavros Tsogkas[1,2], George Papandreou[3], Iasonas Kokkinos[1,2]

[1] Ecole Centrale Paris,[2] INRIA, [3]TTI-Chicago *

End-to-End Integration of a Convolutional Network, Deformable Parts Model and Non-Maximum Suppression

Li Wan David Eigen Rob Fergus
Dept. of Computer Science, Courant Institute, New York University
wanli,deigen,fergus@cs.nyu.edu

Single viewpoint DPM=CNN

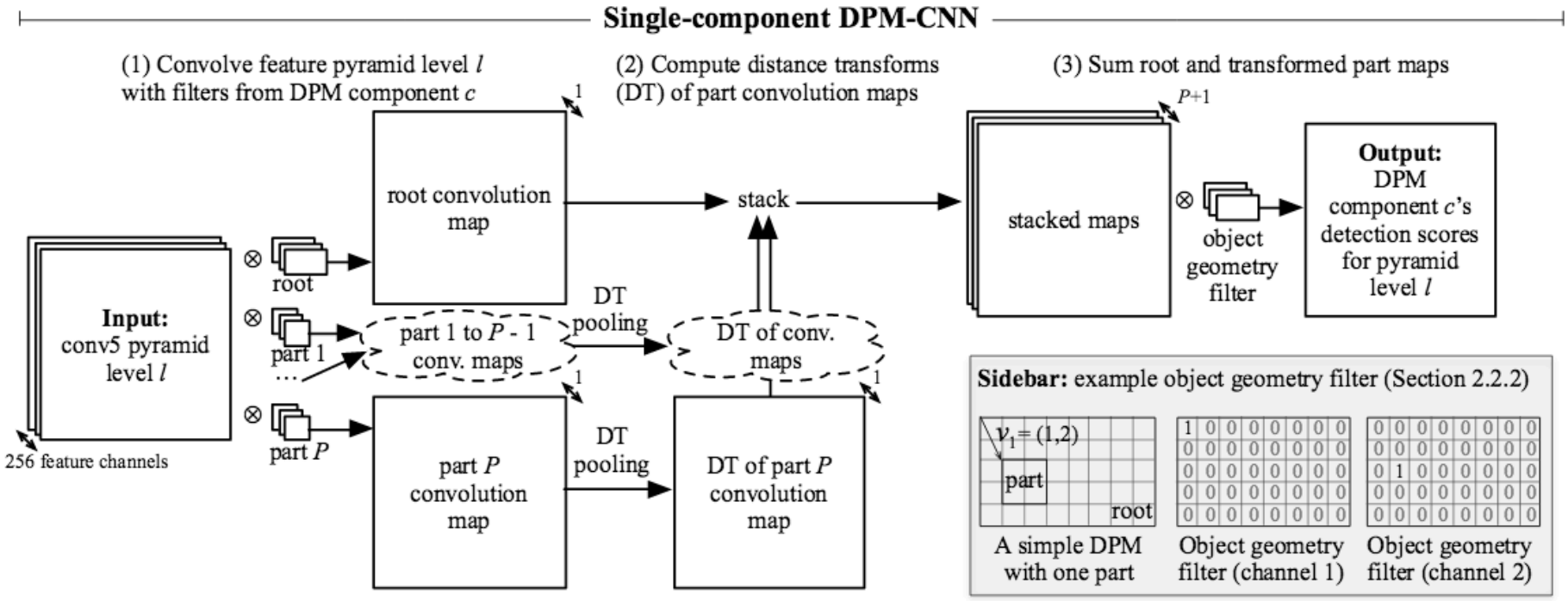


Figure 2. **CNN equivalent to a single-component DPM.** A DPM component can be written as an equivalent CNN by unrolling the DPM detection algorithm into a network. We present the construction for a single-component DPM-CNN here and then show how several of these CNNs can be composed into a multi-component DPM-CNN using a maxout layer (Figure 3). A single-component DPM-CNN operates on a feature pyramid level. (1) The pyramid level is convolved with the root filter and P part filters, yielding $P + 1$ convolution maps. (2) The part convolution maps are then processed with a distance transform pooling layer, which we show is a generalization of max pooling. (3) The root convolution map and the DT pooled part convolution maps are stacked into a single feature map with $P + 1$ channels and then convolved with a sparse object geometry filter (see sidebar diagram and Section 2.2.2). The output is a single-channel score map for the DPM component.

Viewpoint mixture DPM=CNN

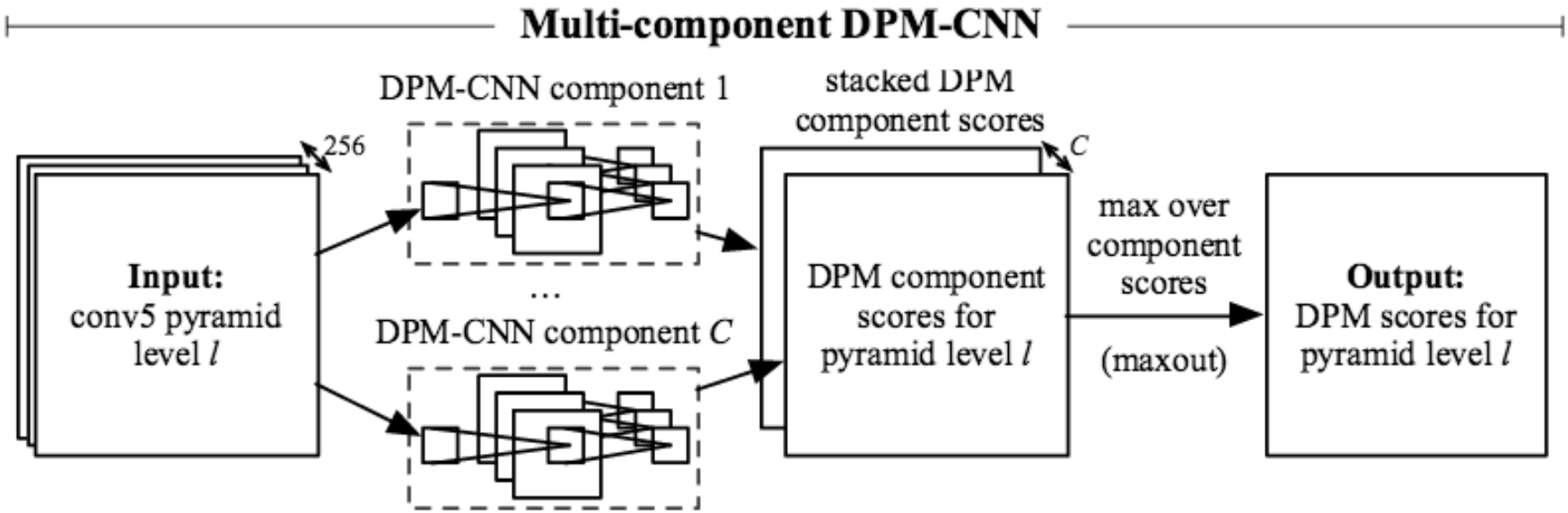


Figure 3. **CNN equivalent to a multi-component DPM.** A multi-component DPM-CNN is composed of one DPM-CNN per component (Figure 2) and a maxout [20] layer that takes a max over component DPM-CNN outputs at each location.

DPM=CNN

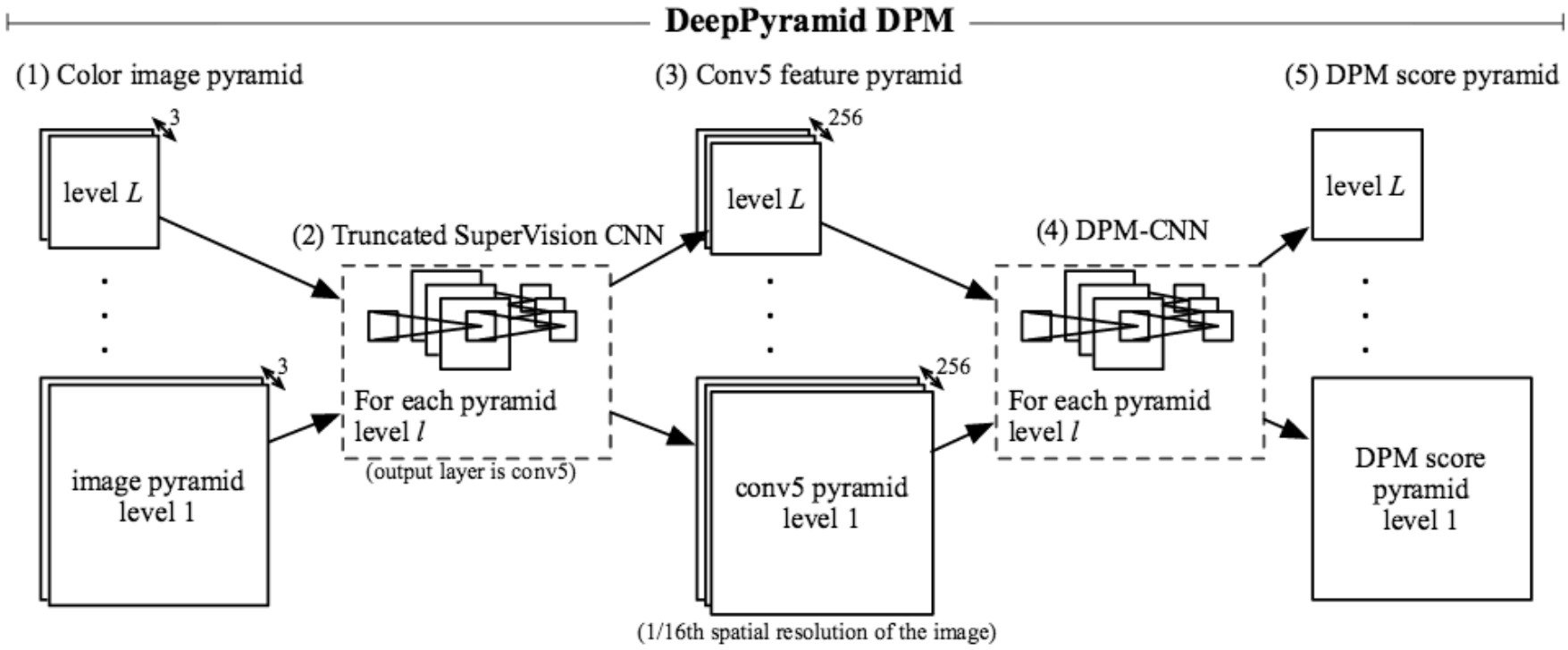


Figure 1. **Schematic model overview.** (1) An image pyramid is built from a color input image. (2) Each pyramid level is forward propagated through a fully-convolutional CNN (*e.g.*, a truncated SuperVision CNN [27] that ends at convolutional layer 5). (3) The result is a pyramid of $conv_5$ feature maps, each at 1/16th the spatial resolution of its corresponding image pyramid level. (4) Each $conv_5$ level is then input into a DPM-CNN, which (5) produces a pyramid of DPM detection scores. Since the whole system is the composition of two CNNs, it can be viewed as a single, unified CNN that takes a color image pyramid as input and outputs a DPM score pyramid.

DPM = CNN (Wan et al.)

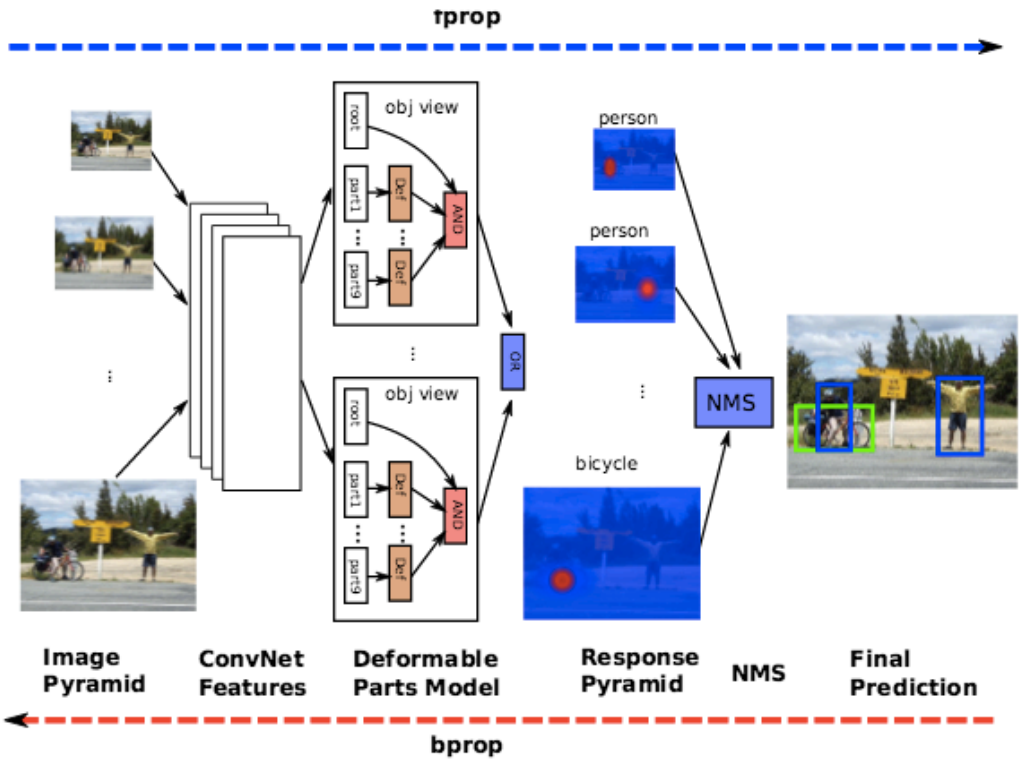


Figure 1. An overview of our system: (i) a convolutional network extracts features from an image pyramid; (ii) a set of deformable parts models (each capturing a different view) are applied to the convolutional feature maps; (iii) non-maximal suppression is applied to the resulting response maps, yielding bounding box predictions. Training is performed using a new loss function that enables back-propagation through all stages.

DPM = CNN (Girshick et al.)

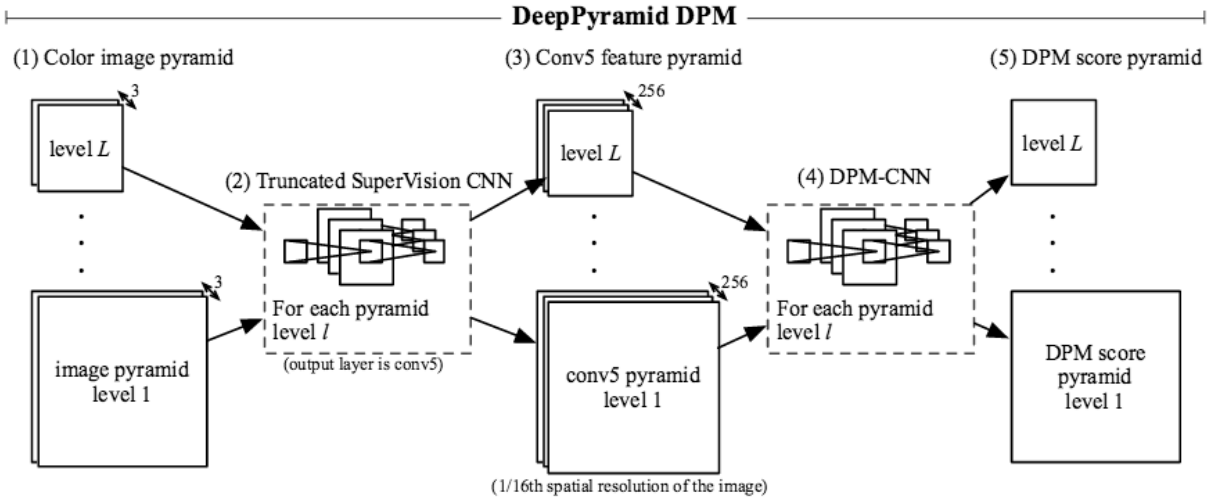


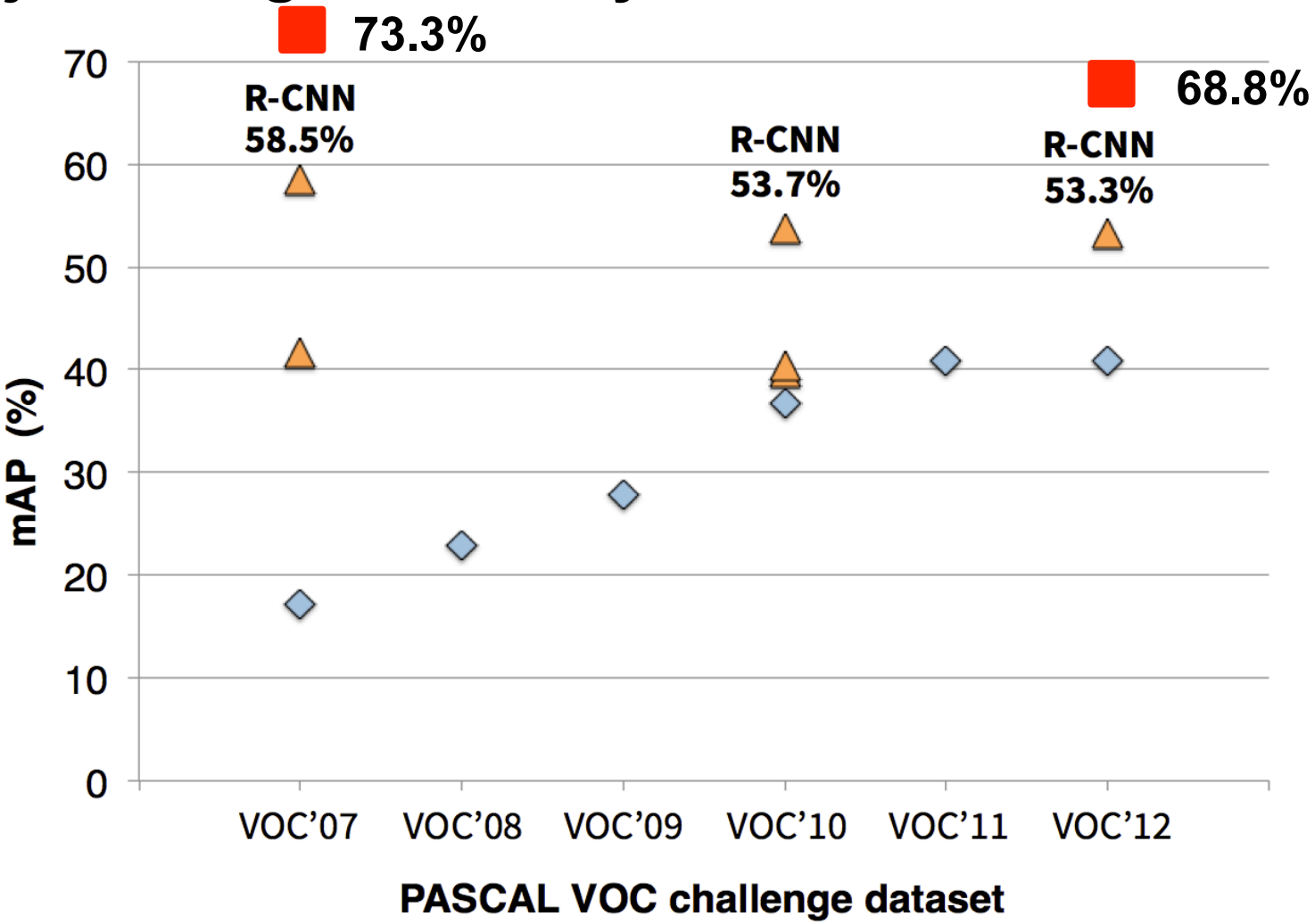
Figure 1. **Schematic model overview.** (1) An image pyramid is built from a color input image. (2) Each pyramid level is forward propagated through a fully-convolutional CNN (*e.g.*, a truncated SuperVision CNN [27] that ends at convolutional layer 5). (3) The result is a pyramid of $conv_5$ feature maps, each at 1/16th the spatial resolution of its corresponding image pyramid level. (4) Each $conv_5$ level is then input into a DPM-CNN, which (5) produces a pyramid of DPM detection scores. Since the whole system is the composition of two CNNs, it can be viewed as a single, unified CNN that takes a color image pyramid as input and outputs a DPM score pyramid.

DPMs as/vs CNNs

method	C	P	aero	bike	bird	boat	botl	bus	car	cat	chair	cow	table	dog	horse	mbike	pers	plant	sheep	sofa	train	tv	mAP
DP-DPM $conv_5$	3	0	41.2	64.1	30.5	23.9	35.6	51.8	54.5	37.2	25.8	46.1	38.8	39.1	58.5	54.8	51.6	25.8	**48.3**	33.1	49.1	56.1	43.3
DP-DPM $conv_5$	3	4	43.1	64.2	31.3	25.0	37.6	55.8	55.7	37.8	27.3	46.0	35.5	39.0	58.2	57.0	53.0	26.6	47.6	35.3	50.6	56.6	44.2
DP-DPM $conv_5$	3	8	42.3	65.1	32.2	24.4	36.7	56.8	55.7	38.0	**28.2**	47.3	37.1	39.2	61.0	56.4	52.2	26.6	47.0	35.0	51.2	56.1	44.4
DP-DPM max_5	3	0	44.6	65.3	32.7	24.7	35.1	54.3	56.5	40.4	26.3	**49.4**	43.2	**41.0**	61.0	55.7	**53.7**	25.5	47.0	39.8	47.9	**59.2**	45.2
C-DPM [34]	3	8	39.7	59.5	35.8	24.8	35.5	53.7	48.6	46.0	29.2	36.8	45.5	42.0	57.7	56.0	37.4	**30.1**	31.1	**50.4**	56.1	51.6	43.4
Conv-DPM [41]	3	9	48.9	67.3	25.3	25.1	35.7	58.3	60.1	35.3	22.7	36.4	37.1	26.9	64.9	62.0	47.0	24.1	37.5	40.2	54.1	57.0	43.3
Conv-DPM+FT [41]	3	9	50.9	**68.3**	31.9	**28.2**	**38.1**	**61.0**	**61.3**	39.8	25.4	46.5	**47.3**	29.6	**67.5**	**63.4**	46.1	25.2	39.1	45.4	**57.0**	57.9	**46.5**
HOG-DPM	6	0	23.8	51.3	5.1	11.5	19.2	41.3	46.3	8.5	15.8	20.8	8.6	10.4	43.9	37.6	31.9	11.9	18.1	25.7	36.5	35.4	25.2
HOG-DPM [18]	6	8	33.2	60.3	10.2	16.1	27.3	54.3	58.2	23.0	20.0	24.1	26.7	12.7	58.1	48.2	43.2	12.0	21.1	36.1	46.0	43.5	33.7
R-CNN [17] $pool_5$	n/a	n/a	**51.8**	60.2	**36.4**	27.8	23.2	52.8	60.6	**49.2**	18.3	47.8	44.3	40.8	56.6	58.7	42.4	23.4	46.1	36.7	51.3	55.7	44.2
fine-tuned variants of R-CNN																							
R-CNN FT $pool_5$	n/a	n/a	58.2	63.3	37.9	27.6	26.1	54.1	66.9	51.4	26.7	55.5	43.4	43.1	57.7	59.0	45.8	28.1	50.8	40.6	53.1	56.4	47.3
R-CNN FT fc_7	n/a	n/a	64.2	69.7	50.0	41.9	32.0	62.6	71.0	60.7	32.7	58.5	46.5	56.1	60.6	66.8	54.2	31.5	52.8	48.9	57.9	64.7	54.2
R-CNN FT fc_7 BB	n/a	n/a	**68.1**	**72.8**	**56.8**	**43.0**	**36.8**	**66.3**	**74.2**	**67.6**	**34.4**	**63.5**	**54.5**	**61.2**	**69.1**	**68.6**	**58.7**	**33.4**	**62.9**	**51.1**	**62.5**	**64.8**	**58.5**

Table 1. **Detection average precision (%) on VOC 2007 test.** Column C shows the number of components and column P shows the number of parts per component. Our method is DP-DPM (DeepPyramid DPM).

Object recognition, early 2015



A. Krizhevsky, I. Sutskever, and G. Hinton. ImageNet classification with deep convolutional neural networks. *NIPS*13
P. Sermanet, et. al. Overfeat: Integrated recognition, localization and detection using convolutional networks. ICLR 14.
R. Girshick, et. al. Rich feature hierarchies for accurate object detection and semantic segmentation. *CVPR* 14
S. Ren et. al., Object Detection Networks on Convolutional Feature Maps, arxiv, May 2015

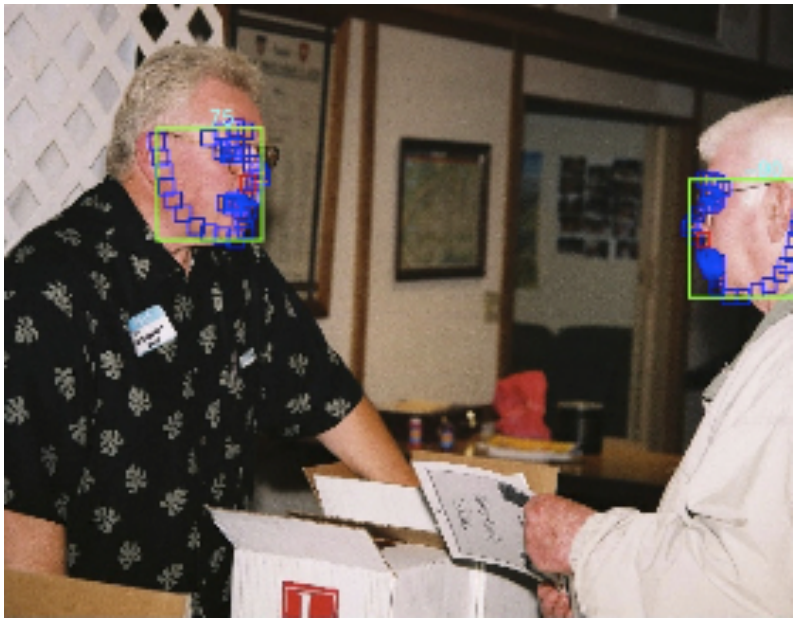
Object recognition, now

http://host.robots.ox.ac.uk:8080/leaderboard/displaylb.php?challengeid=11&compid=4

	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
Faster RCNN, ResNet (VOC+COCO) [?]	83.8	92.1	88.4	84.8	75.9	71.4	86.3	87.8	94.2	66.8	89.4	69.2	93.9	91.9	90.9	89.6	67.9	88.2	76.8	90.3	80.0	10-Dec-2015
ION [?]	76.4	87.5	84.7	76.8	63.8	58.3	82.6	79.0	90.9	57.8	82.0	64.7	88.9	86.5	84.7	82.3	51.4	78.2	69.2	85.2	73.5	23-Nov-2015
MNC baseline [?]	75.9	86.4	81.1	76.4	64.3	57.8	81.1	80.3	92.0	55.2	82.6	61.0	89.9	86.4	84.6	85.4	53.1	79.8	66.1	84.7	69.9	15-Dec-2015
Faster RCNN baseline (VOC+COCO) [?]	75.9	87.4	83.6	76.8	62.9	59.6	81.9	82.0	91.3	54.9	82.6	59.0	89.0	85.5	84.7	84.1	52.2	78.9	65.5	85.4	70.2	24-Nov-2015
LocNet [?]	74.8	86.3	83.0	76.1	60.8	54.6	79.9	79.0	90.6	54.3	81.6	62.0	89.0	85.7	85.5	82.8	49.7	76.6	67.5	83.2	67.4	06-Nov-2015
** HRCNN ** [?]	74.6	85.9	83.9	75.5	60.9	54.5	81.4	79.1	90.6	53.3	79.7	61.6	89.9	86.2	85.8	78.2	49.1	75.1	68.6	86.1	67.7	13-Nov-2015
MR_CNN_S_CNN_MORE_DATA [?]	73.9	85.5	82.9	76.6	57.8	62.7	79.4	77.2	86.6	55.0	79.1	62.2	87.0	83.4	84.7	78.9	45.3	73.4	65.8	80.3	74.0	06-Jun-2015
HyperNet_VGG [?]	71.4	84.2	78.5	73.6	55.6	53.7	78.7	79.8	87.7	49.6	74.9	52.1	86.0	81.7	83.3	81.8	48.6	73.5	59.4	79.9	65.7	12-Oct-2015
HyperNet_SP [?]	71.3	84.1	78.3	73.3	55.5	53.6	78.6	79.6	87.5	49.5	74.9	52.1	85.6	81.6	83.2	81.6	48.4	73.2	59.3	79.7	65.6	28-Oct-2015
Fast R-CNN + YOLO [?]	70.7	83.4	78.5	73.5	55.8	43.4	79.1	73.1	89.4	49.4	75.5	57.0	87.5	80.9	81.0	74.7	41.8	71.5	68.5	82.1	67.2	06-Nov-2015
MR_CNN_S_CNN [?]	70.7	85.0	79.6	71.5	55.3	57.7	76.0	73.9	84.6	50.5	74.3	61.7	85.5	79.9	81.7	76.4	41.0	69.0	61.2	77.7	72.1	09-May-2015
RPN [?]	70.4	84.9	79.8	74.3	53.9	49.8	77.5	75.9	88.5	45.6	77.1	55.3	86.9	81.7	80.9	79.6	40.1	72.6	60.9	81.2	61.5	01-Jun-2015
DEEP_ENSEMBLE_COCO [?]	70.1	84.0	79.4	71.6	51.9	51.1	74.1	72.1	88.6	48.3	73.4	57.8	86.1	80.0	80.7	70.4	46.6	69.6	68.8	75.9	71.4	03-May-2015
Networks on Convolutional Feature Maps [?]	68.8	82.8	79.0	71.6	52.3	53.7	74.1	69.0	84.9	46.9	74.3	53.1	85.0	81.3	79.5	72.2	38.9	72.4	59.5	76.7	68.1	17-Apr-2015
Fast R-CNN VGG16 extra data [?]	68.4	82.3	78.4	70.8	52.3	38.7	77.8	71.6	89.3	44.2	73.0	55.0	87.5	80.5	80.8	72.0	35.1	68.3	65.7	80.4	64.2	17-Apr-2015
UMICH_FGS_STRUCT [?]	66.4	82.9	76.1	64.1	44.6	49.4	70.3	71.2	84.6	42.7	68.6	55.8	82.7	77.1	79.9	68.7	41.4	69.0	60.0	72.0	66.2	20-Jun-2015
NUS_NIN_c2000 [?]	63.8	80.2	73.8	61.9	43.7	43.0	70.3	67.6	80.7	41.9	69.7	51.7	78.2	75.2	76.9	65.1	38.6	68.3	58.0	68.7	63.3	30-Oct-2014
BabyLearning [?]	63.2	78.0	74.2	61.3	45.7	42.7	68.2	66.8	80.2	40.6	70.0	49.8	79.0	74.5	77.9	64.0	35.3	67.9	55.7	68.7	62.6	12-Nov-2014
NUS_NIN [?]	62.4	77.9	73.1	62.6	39.5	43.3	69.1	66.4	78.9	39.1	68.1	50.0	77.2	71.3	76.1	64.7	38.4	66.9	56.2	66.9	62.7	30-Oct-2014
R-CNN (bbox reg) [?]	62.4	79.6	72.7	61.9	41.2	41.9	65.9	66.4	84.6	38.5	67.2	46.7	82.0	74.8	76.0	65.2	35.6	65.4	54.2	67.4	60.3	26-Oct-2014
R-CNN [?]	59.2	76.8	70.9	56.6	37.5	36.9	62.9	63.6	81.1	35.7	64.3	43.9	80.4	71.6	74.0	60.0	30.8	63.4	52.0	63.5	58.7	25-Oct-2014
YOLO [?]	57.9	77.0	67.2	57.7	38.3	22.7	68.3	55.9	81.4	36.2	60.8	48.5	77.2	72.3	71.3	63.5	28.9	52.2	54.8	73.9	50.8	06-Nov-2015
Feature Edit [?]	56.3	74.6	69.1	54.4	39.1	33.1	65.2	62.7	69.7	30.8	56.0	44.6	70.0	64.4	71.1	60.2	33.3	61.3	46.4	61.7	57.8	06-Sep-2014
R-CNN (bbox reg) [?]	53.3	71.8	65.8	52.0	34.1	32.6	59.6	60.0	69.8	27.6	52.0	41.7	69.6	61.3	68.3	57.8	29.6	57.8	40.9	59.3	54.1	13-Mar-2014
SDS [?]	50.7	69.7	58.4	48.5	28.3	28.8	61.3	57.5	70.8	24.1	50.7	35.9	64.9	59.1	65.8	57.1	26.0	58.8	38.6	58.9	50.7	21-Jul-2014
R-CNN [?]	49.6	68.1	63.8	46.1	29.4	27.9	56.6	57.0	65.9	26.5	48.7	39.5	66.2	57.3	65.4	53.2	26.2	54.5	38.1	50.6	51.6	30-Jan-2014

What is left for part-based models?

- All tasks where parts are desired
  - for fine-grained recognition, pose estimation, grasping, ….



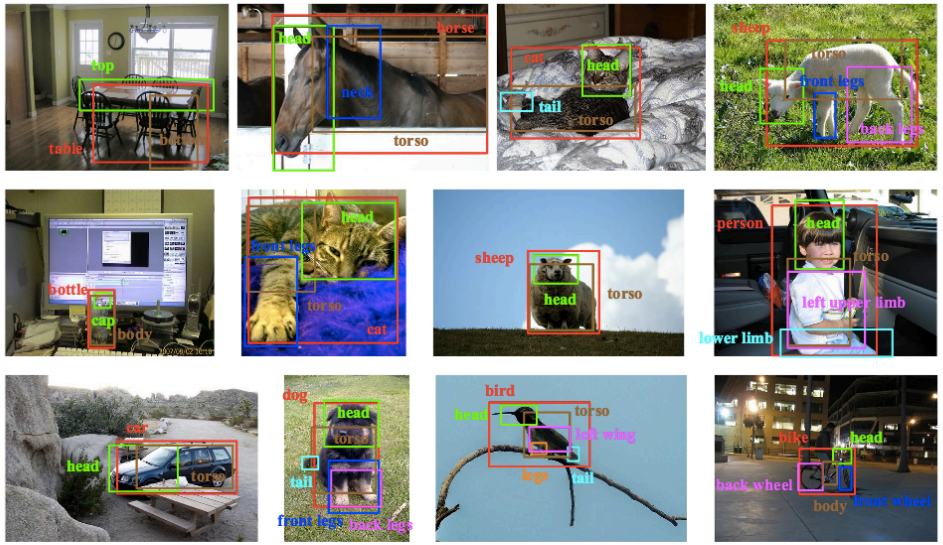
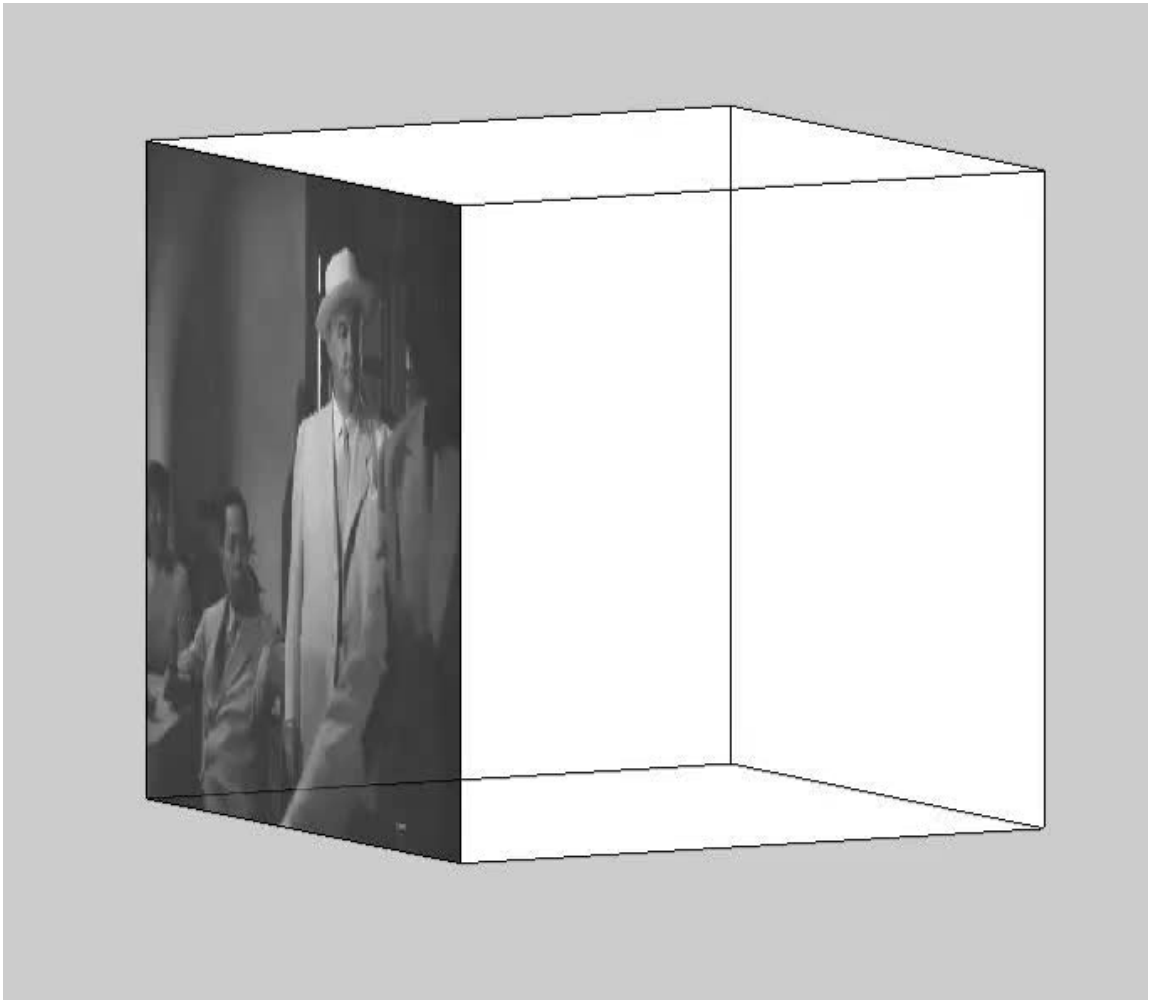


Figure 7: Visualization on some examples of the detected object-part configurations by DeePM (best viewed in color).

Face Detection, Pose Estimation, and Landmark Localization in the Wild, X. Zhu and D. Ramanan, CVPR 2012

DeePM: A Deep Part-Based Model for Object Detection and Semantic Part Localization, J. Zhu, et al. 2016 http://arxiv.org/abs/1511.07131

What is left for part-based models?

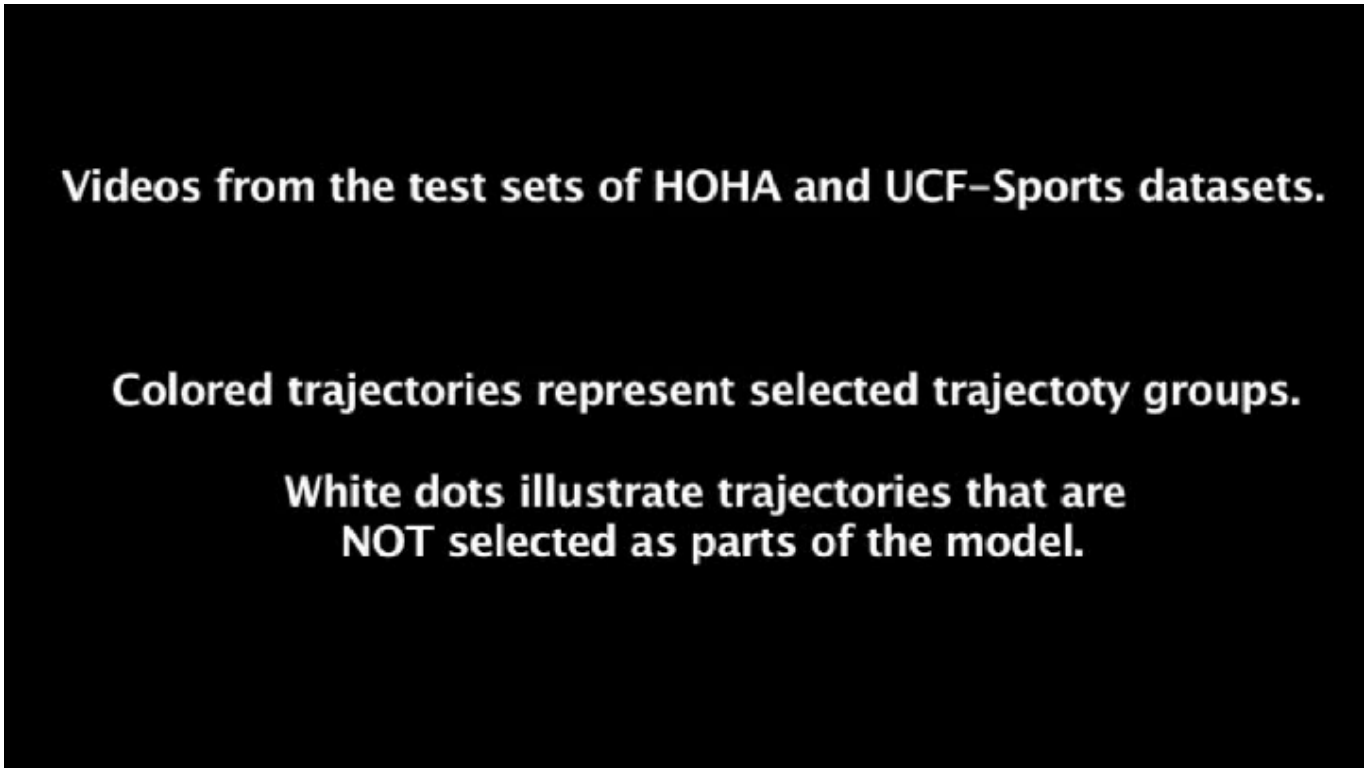


M. Raptis

S. Soatto

M. Raptis, **I.Kokkinos**, and S. Soatto, Discovering Discriminative Action Parts from Mid-Level Video Representations, CVPR, 2012

Space-time ‘contours’ for action recognition



M. Raptis

S. Soatto

M. Raptis, **I.Kokkinos**, and S. Soatto, Discovering Discriminative Action Parts from Mid-Level Video Representations, CVPR, 2012

What is left for part-based models?

Ill-posed tasks (e.g. 3D from 2D) where high-dimensional pose is needed

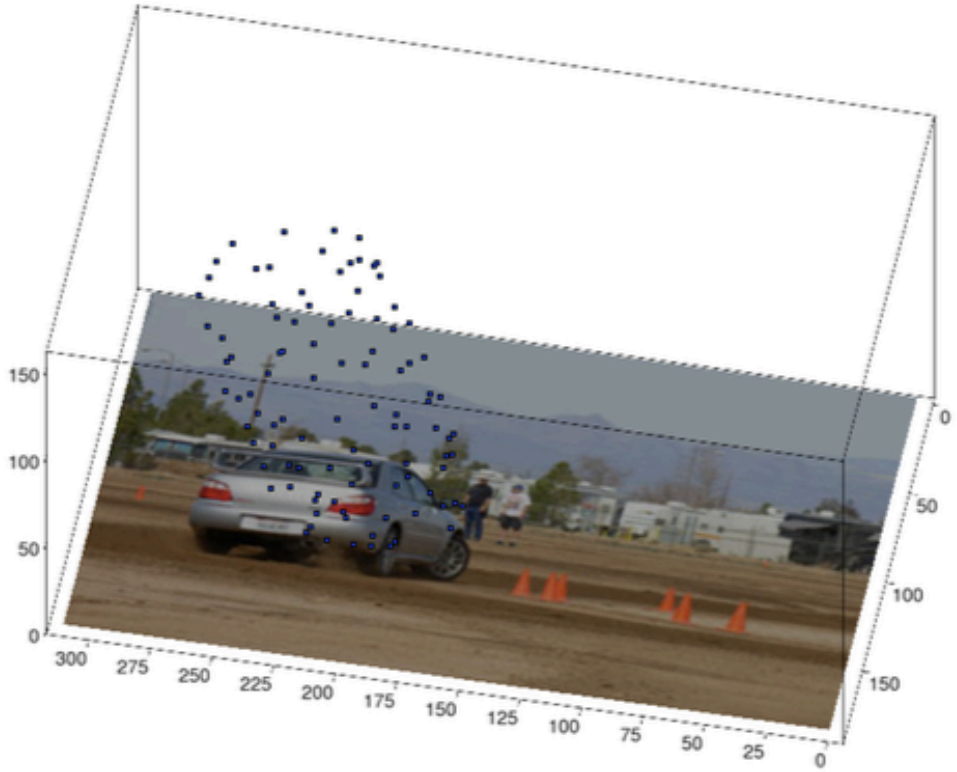
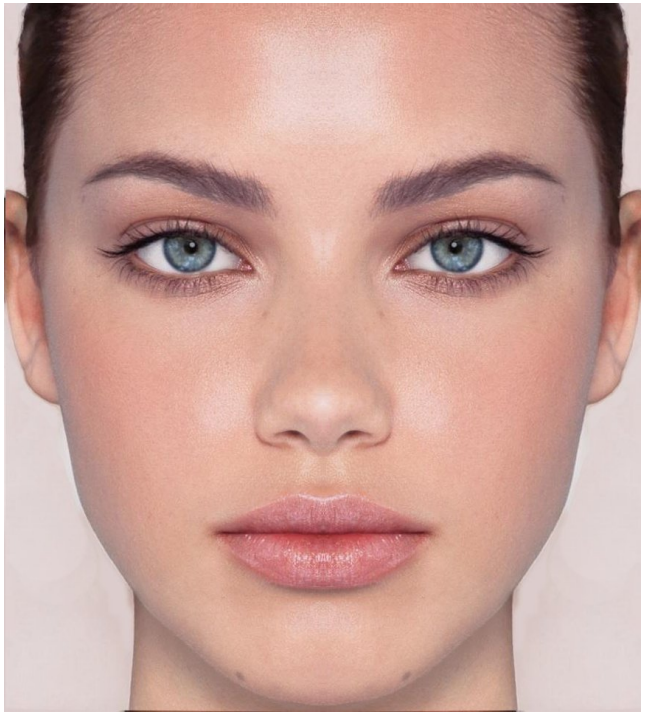


Figure 3.5: This figure shows how the algorithm reconstructs a car. The image plane containing the photo of the car is the input, the blue points represent the positions of the car model reconstructed in 3d.

Work in progress M. Berman, S. Kinauer, I. Kokkinos

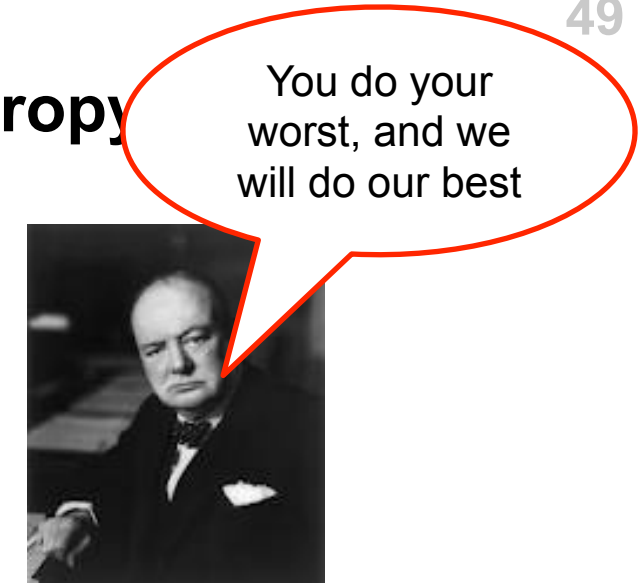
Fast Detection with Loopy Part Models



H. Boussaid

H. Boussaid and I. Kokkinos, Fast and Exact ADMM-based Discriminative Shape Segmentation using Loopy Part Models, CVPR 2014.

Lagrangian (e.g. SVMs, Maximum Entropy



- Constrained optimization problem:

$$\begin{aligned} \min_{w} \quad & f(w) \\ s.t. \quad & h_i(w) = 0, \quad i = 1 \dots l \\ & g_i(w) \leq 0, \quad i = 1 \dots m \end{aligned}$$

- Equivalent: $\min_{w} \quad f_{uc}(w) = f(w) + \sum_{i=1}^{l} I_0(h_i(w)) + \sum_{i=1}^{m} I_+(g_i(w))$

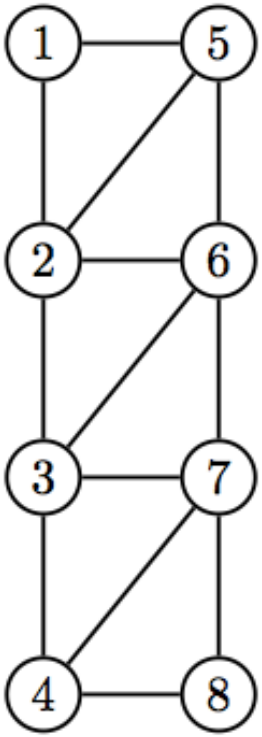
$$I_0(x) = \begin{cases} 0, & x = 0 \\ \infty, & x \neq 0 \end{cases}, \qquad I_+(x) = \begin{cases} 0, & x \leq 0 \\ \infty, & x > 0 \end{cases}$$

- `Soften' constraints: $L(w, \lambda, \mu) = f(w) + \sum_{i=1}^{l} \lambda_i h_i(w) + \sum_{i=1}^{m} \mu_i g_i(w), \quad \mu_i > 0 \forall i$

- Solve: $f(w^*) = \min_{w} \max_{\lambda, \mu : \mu_i > 0} L(w, \lambda, \mu)$

Problem Decomposition

$$\begin{aligned} \min_x \quad & \sum_i f^i(x) \\ \text{s.t.} \quad & x \in \mathcal{C} \end{aligned}$$



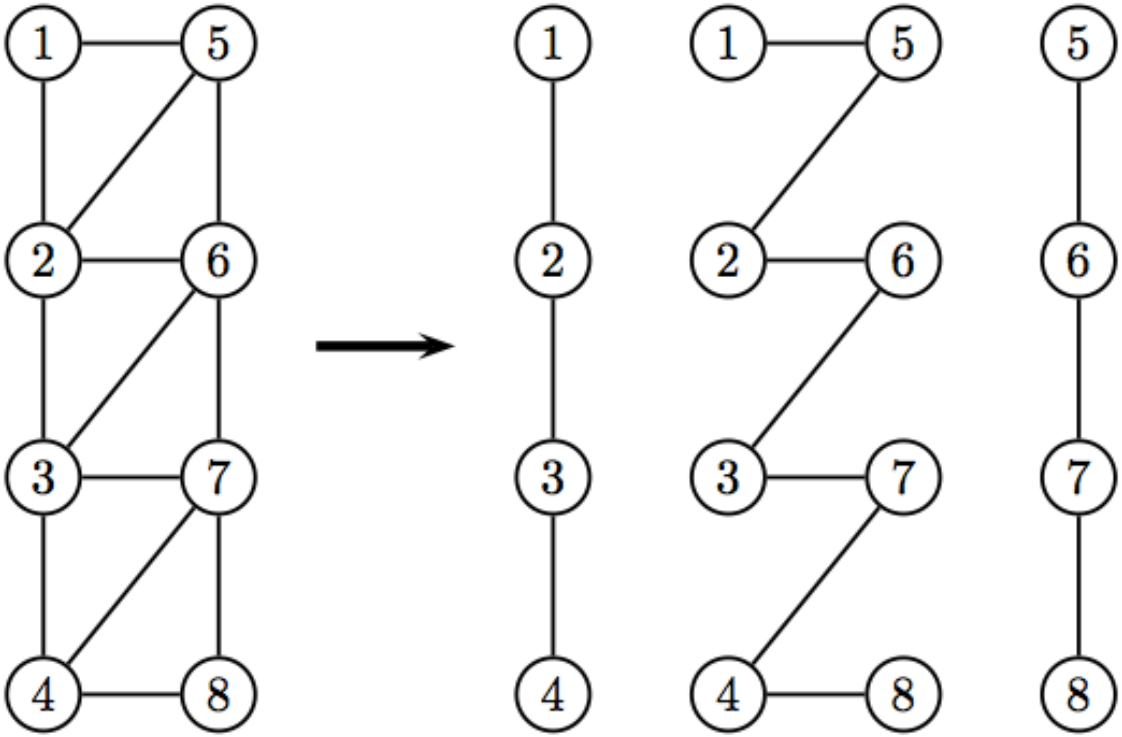
(a)

Loops in graph: Max Product no longer exact

3-clique graph: O(N^3) **Generalized Distance Transforms: O(N)**

Problem Decomposition

$$\begin{array}{ll} \min_x & \sum_i f^i(x) \\ \text{s.t.} & x \in \mathcal{C} \end{array} \qquad \begin{array}{ll} \min_{\{x^i\},x} & \sum_i f^i(x^i) \\ \text{s.t.} & x^i \in \mathcal{C},\ x^i = x \end{array}$$



Loops in graph: Max Product no longer exact

3-clique graph: O(N^3) **Generalized Distance Transforms: O(N)**

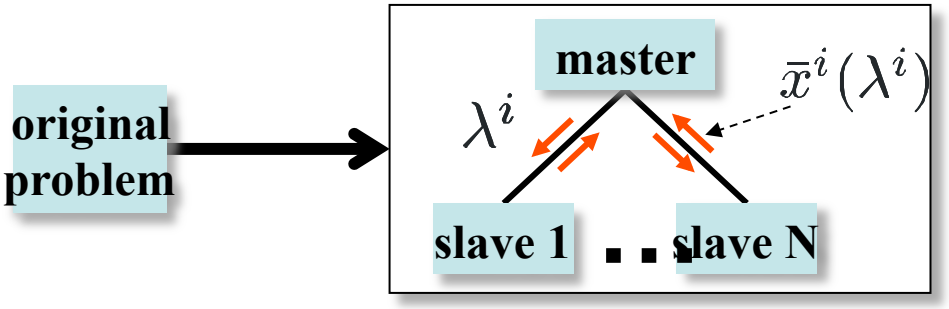
Dual Decomposition

$$g(\{\lambda^i\}) = \min_{\{x^i \in \mathcal{C}\}, x} \sum_i f^i(x^i) + \sum_i \lambda^i \cdot (x^i - x)$$

Dual **decomposes**: $g^i(\lambda^i) = \min_{x^i \in \mathcal{C}} f^i(x^i) + \lambda^i \cdot x^i$

Dual **ascent**: $\max_{\{\lambda^i\} \in \Lambda} g(\{\lambda^i\}) = \sum_i g^i(\lambda^i)$

$$\lambda^i \leftarrow \left[\lambda^i + \alpha_t \bar{x}^i(\lambda^i)\right]_\Lambda$$



(Dantzig, Wolfe -60, Bertsekas -80, Komodakis & Paragios-'08/'11)

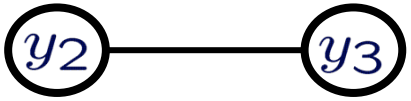
Toy example

$$\mathbb{1}(x) = \begin{cases} 0 & x = 0 \\ +\infty & x \neq 0 \end{cases}$$

$$E(y_1, y_2, y_3) = \mathbb{1}(y_1 + 1) + \left(y_2 - y_1 - \frac{1}{2}\right)^2 + y_2^2 + \left(y_3 - y_2 - \frac{1}{2}\right)^2 + \mathbb{1}(y_3 - 1)$$

$$E(y_1, y_2, y_3) = \left(y_2 + \frac{1}{2}\right)^2 + y_2^2 + \left(y_2 - \frac{1}{2}\right)^2$$

$$E_A(y_1, y_{2,A}) = \left(y_{2,A} + \frac{1}{2}\right)^2 + (y_{2,A})^2 + \lambda_A \left(y_{2,A} - y_{2,B}\right)$$



$$E_B(y_{2,B}, y_3) = \left(y_{2,B} - \frac{1}{2}\right)^2 + (y_{2,B})^2 + \lambda_B \left(y_{2,B} - y_{2,A}\right)$$

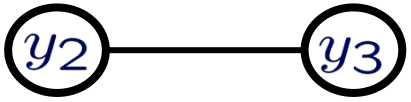
Toy example

$$\mathbb{1}(x) = \begin{cases} 0 & x = 0 \\ +\infty & x \neq 0 \end{cases}$$

$$E(y_1, y_2, y_3) = \mathbb{1}(y_1 + 1) + \left(y_2 - y_1 - \frac{1}{2}\right)^2 + y_2^2 + \left(y_3 - y_2 - \frac{1}{2}\right)^2 + \mathbb{1}(y_3 - 1)$$

$$E(y_1, y_2, y_3) = \left(y_2 + \frac{1}{2}\right)^2 + y_2^2 + \left(y_2 - \frac{1}{2}\right)^2$$

$$E_A(y_{2,A}) = 2\left(y_{2,A} + \frac{1 + \lambda_A}{4}\right)^2 + c_A$$

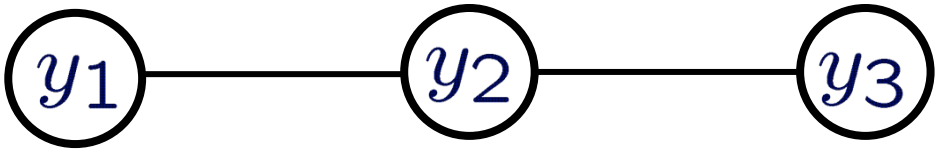


$$E_B(y_{2,B}) = 2\left(y_{2,B} + \frac{-1 + \lambda_B}{4}\right)^2 + c_B$$

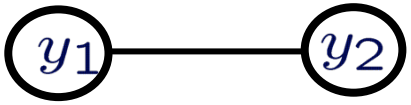
Dual Decomposition: Change λ's until the two parabolas meet

But: we (kind of) know where they should meet: half-way

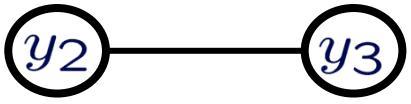
Toy example, continued



$$E(y_1, y_2, y_3) = \left(y_2 + \frac{1}{2}\right)^2 + y_2^2 + \left(y_2 - \frac{1}{2}\right)^2$$

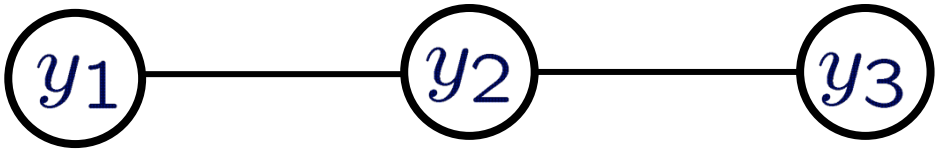


$$E_{\rho,A}(y_{2,A}) = \left(y_{2,A} + \frac{1+\lambda_A}{4}\right)^2 + \rho(y_{2,A} - v)^2$$

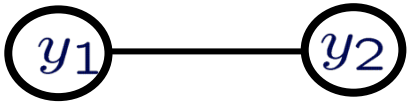


$$E_{\rho,B}(y_{2,B}) = \left(y_{2,B} + \frac{-1+\lambda_B}{4}\right)^2 + \rho(y_{2,B} - v)^2$$

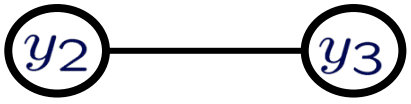
Toy example, continued



$$E(y_1, y_2, y_3) = \left(y_2 + \frac{1}{2}\right)^2 + y_2^2 + \left(y_2 - \frac{1}{2}\right)^2$$



$$E_{\rho,A}(y_{2,A}) = \left(y_{2,A} + \frac{1+\lambda_A}{4(1+\sqrt{\rho})} - \frac{\sqrt{\rho}}{1+\sqrt{\rho}}v\right)^2$$



$$E_{\rho,B}(y_{2,B}) = \left(y_{2,B} + \frac{-1+\lambda_B}{4(1+\sqrt{\rho})} - \frac{\sqrt{\rho}}{1+\sqrt{\rho}}v\right)^2$$

Alternating Direction Method of Multipliers (ADMM)

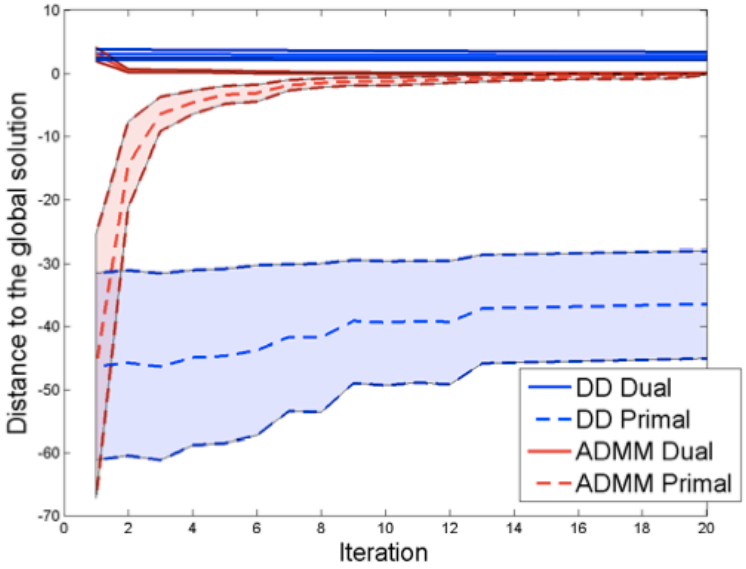
Augmented Lagrangian:

$$\mathcal{A}(\mathbf{X}, u, \lambda) = \sum_{i=1}^{N}(S_i(\mathbf{X}_i) + \sum_{r\in R}\lambda_i(r)\mathbf{X}_i(r) \; - \sum_{r\in R}(\sum_i \lambda_i(r))u(r) - \rho\sum_i\sum_r(\mathbf{X}_i(r) - u(r))^2$$

$$\mathbf{X}_i^{t+1} = \underset{\mathbf{X}_i}{\operatorname{argmax}}\, \mathcal{A}(\mathbf{X}_i, u^t, \lambda^t)$$

$$u^{t+1} = \underset{u}{\operatorname{argmax}}\, \mathcal{A}(\{\mathbf{X}_i^{t+1}\}, u, \lambda^t)$$

$$\lambda_i^{t+1}(r) = \lambda_i^t(r) - \alpha_t(\mathbf{X}_i^{t+1}(r) - u^{t+1}(r))$$



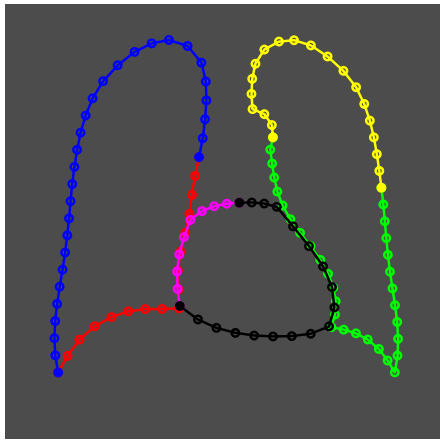
(Powell –'70, Gabay et al '76, Proximal splitting, Bregman iterations… Boy et al '11)

Application: Multi-Organ Segmentation & Localization

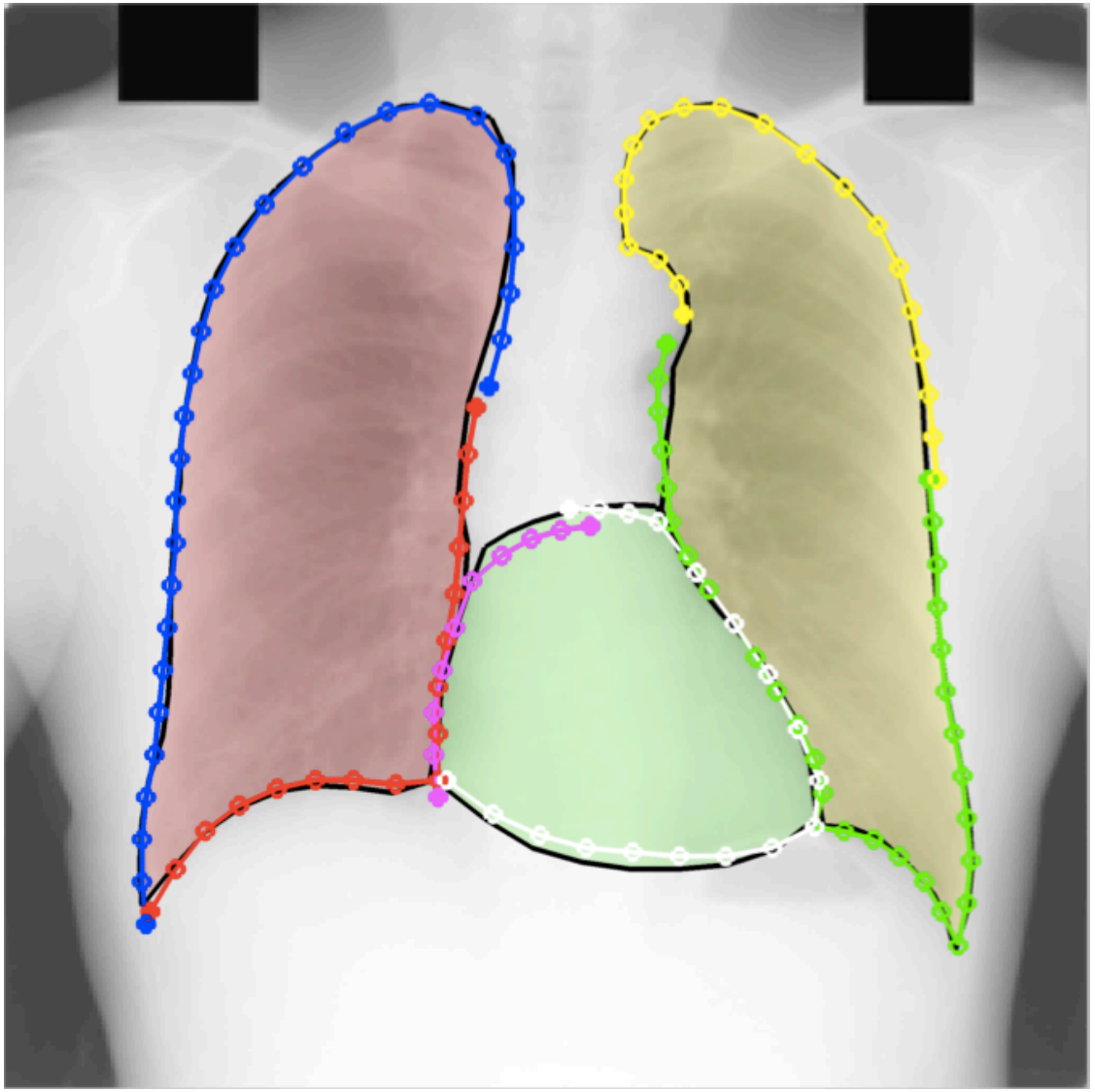
Input image

Ground truth

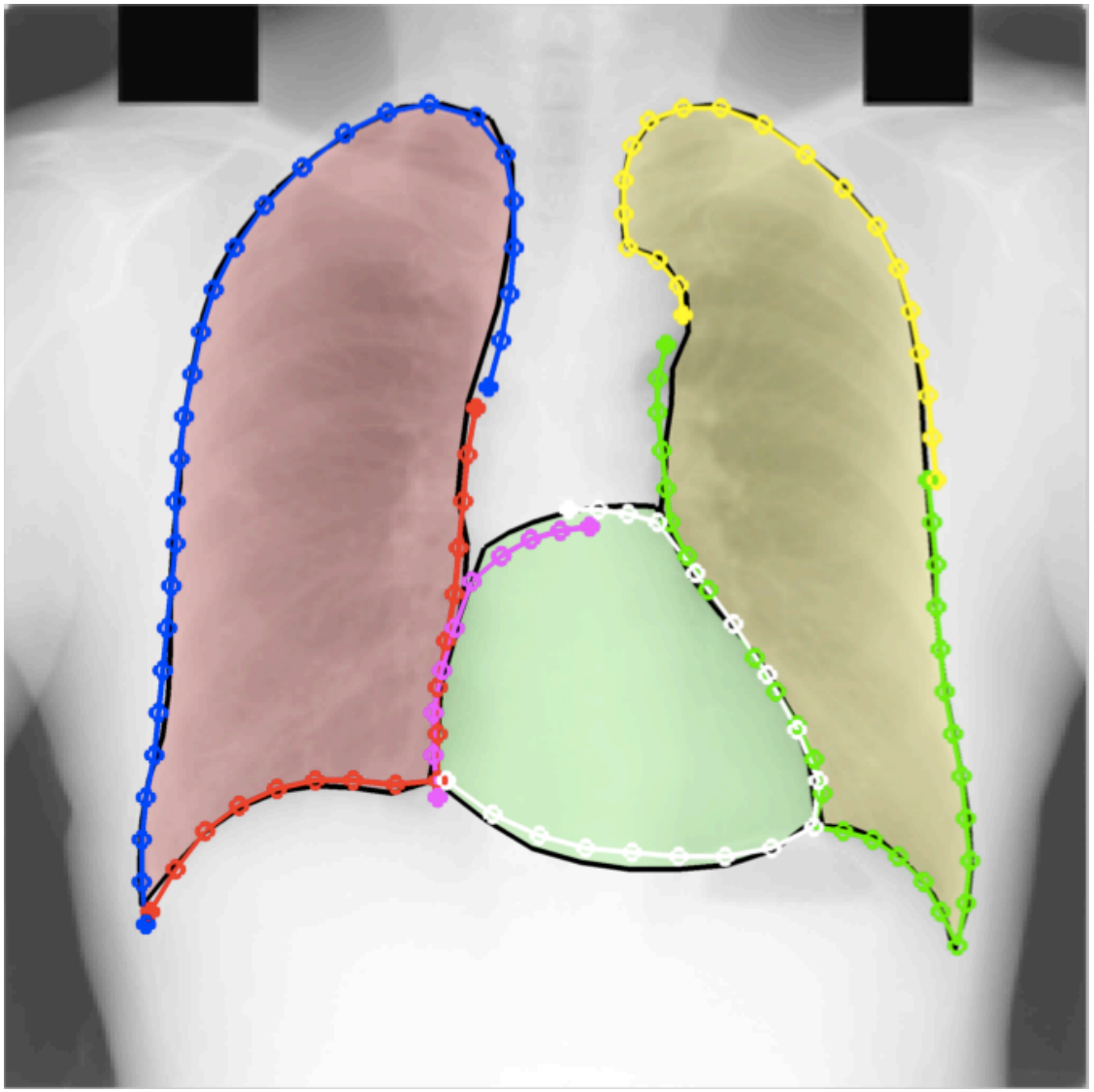
The model



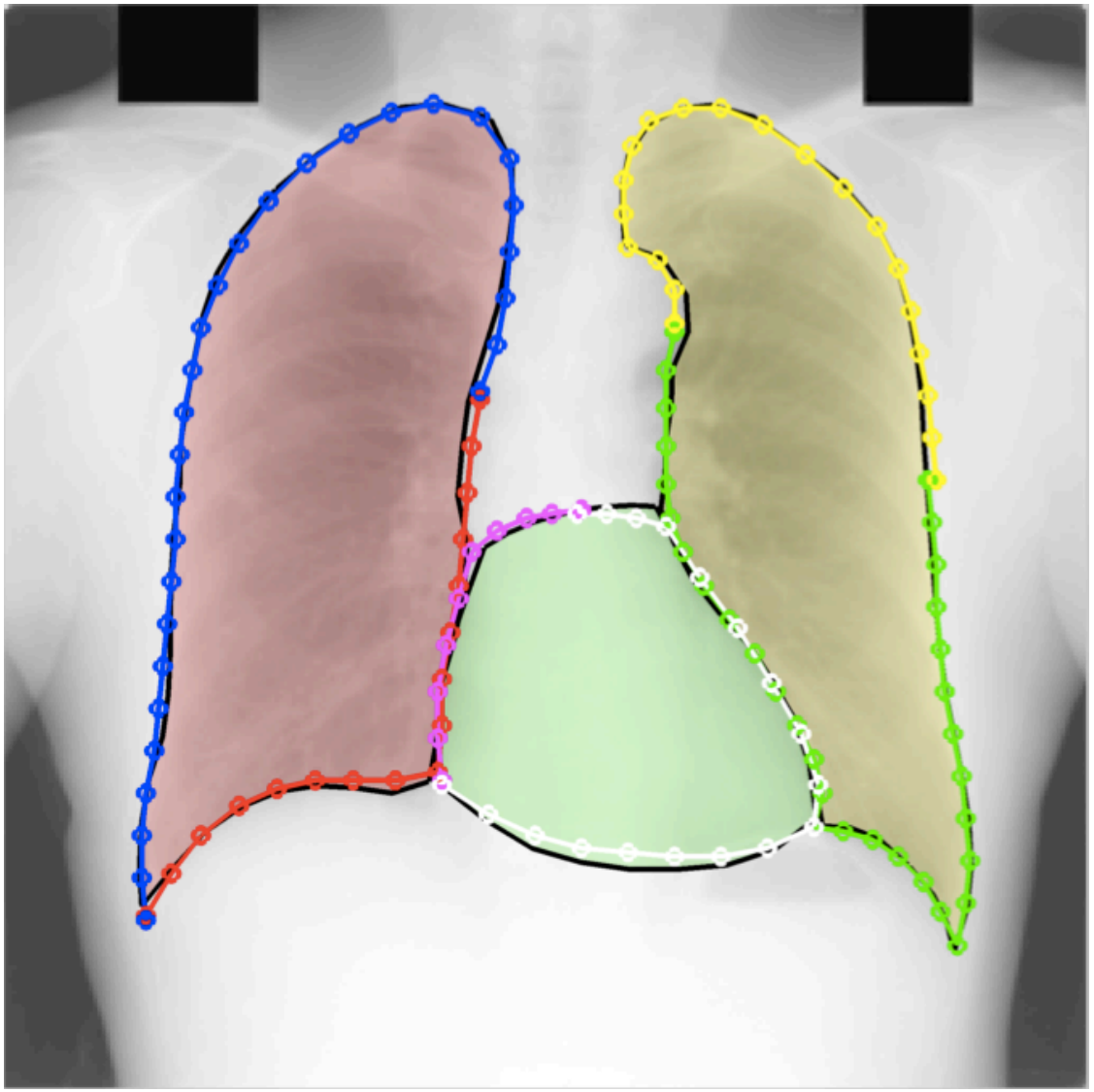
ADMM-1



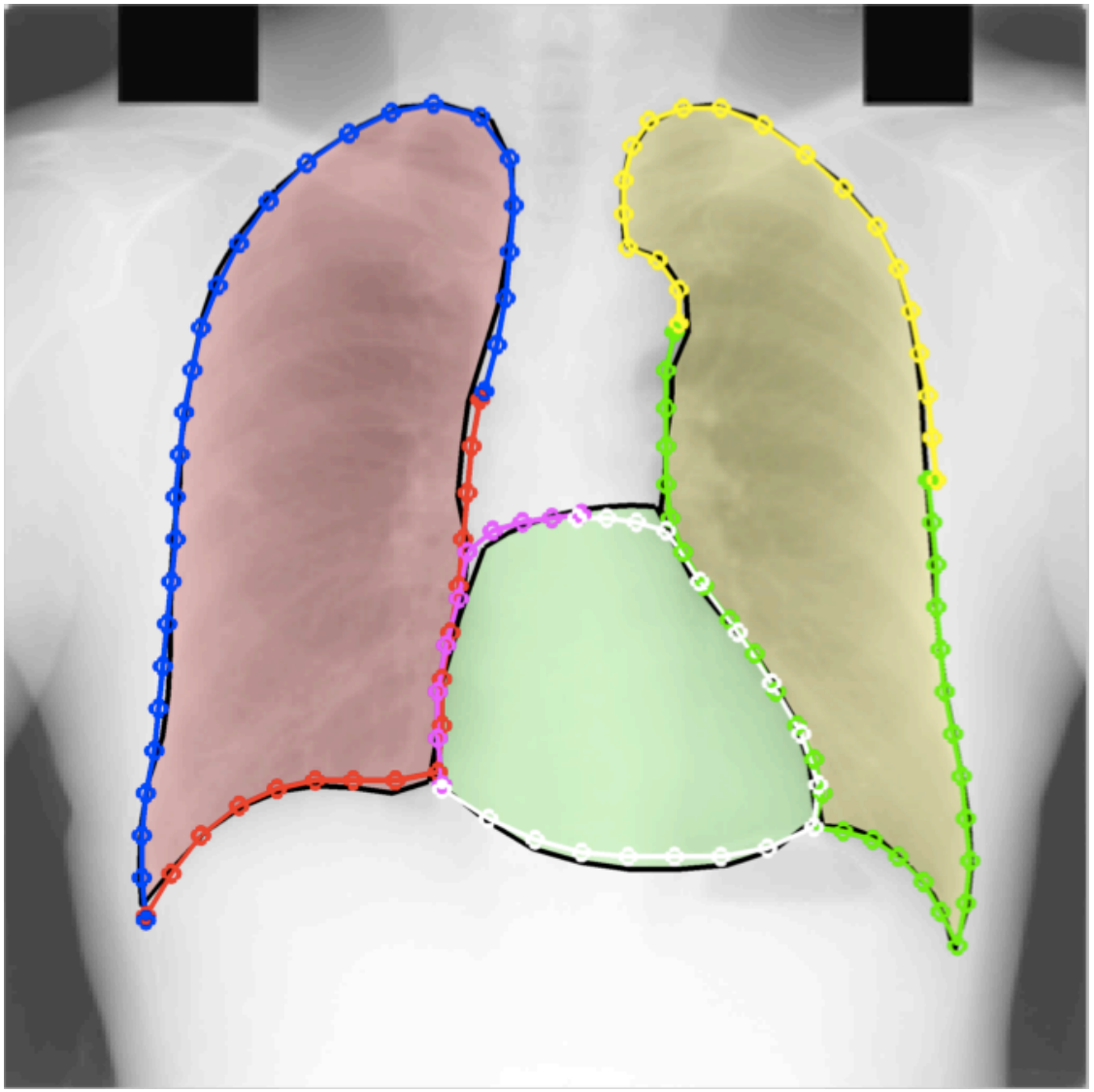
ADMM-1



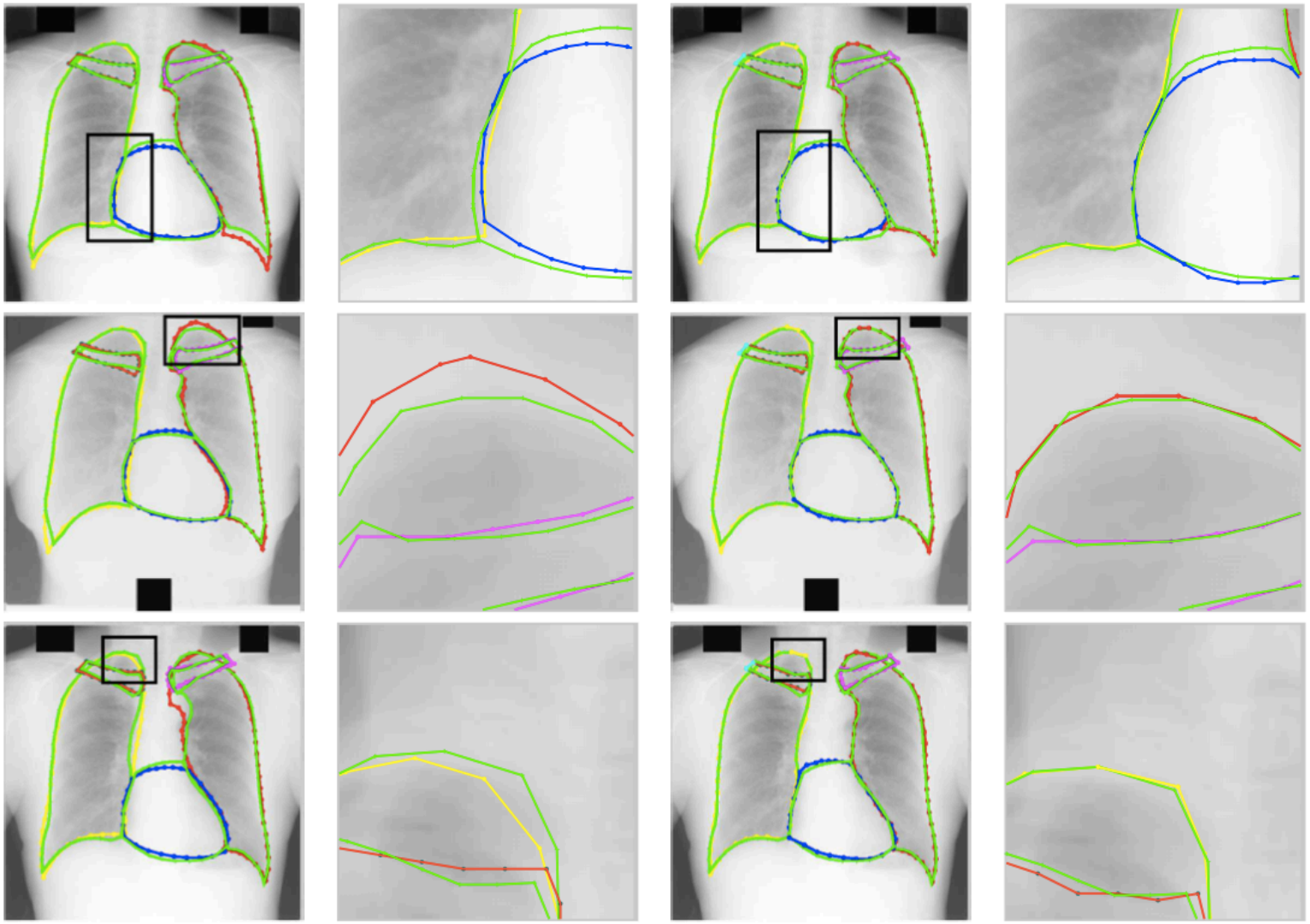
ADMM-2



ADMM-3



Comparison to loop-free baseline



a b c d

Loop-free (ISBI 14) **Loopy (CVPR 14)**

H. Boussaid, I.K., and N. Paragios, Discriminative Learning of Deformable Contour Models, ISBI, 2014.

H. Boussaid and **I. K**, Fast and Exact: Shape Segmentation using Structured Prediction and ADMM, CVPR 2014.

3D models – Explicit geometric modeling

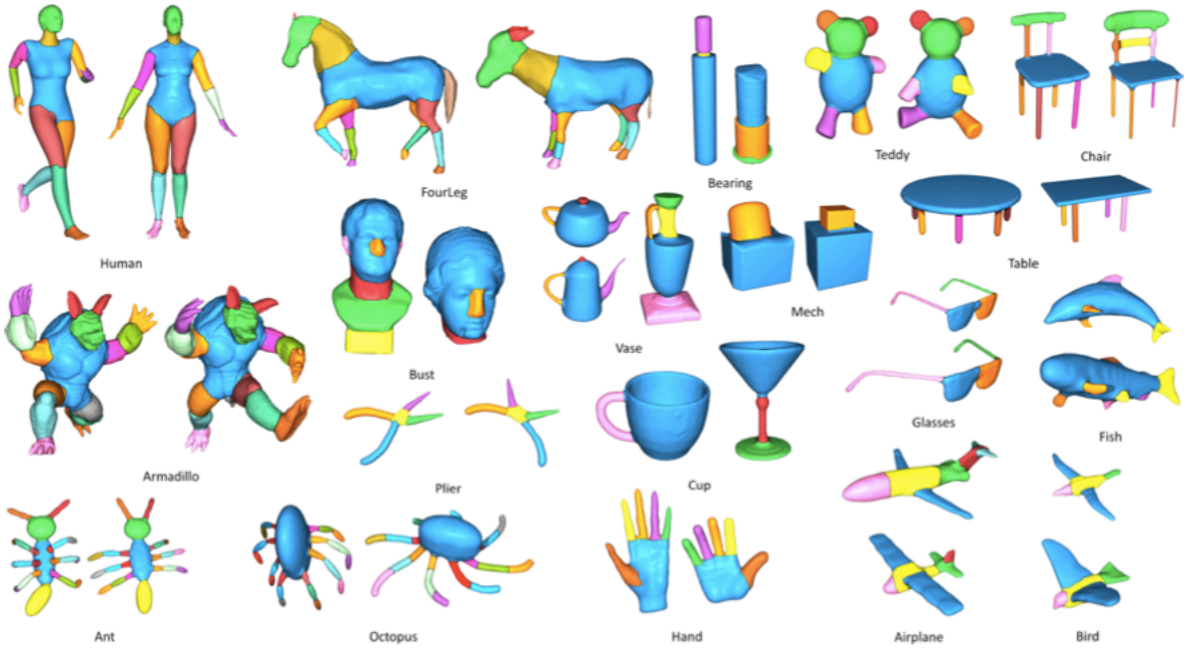
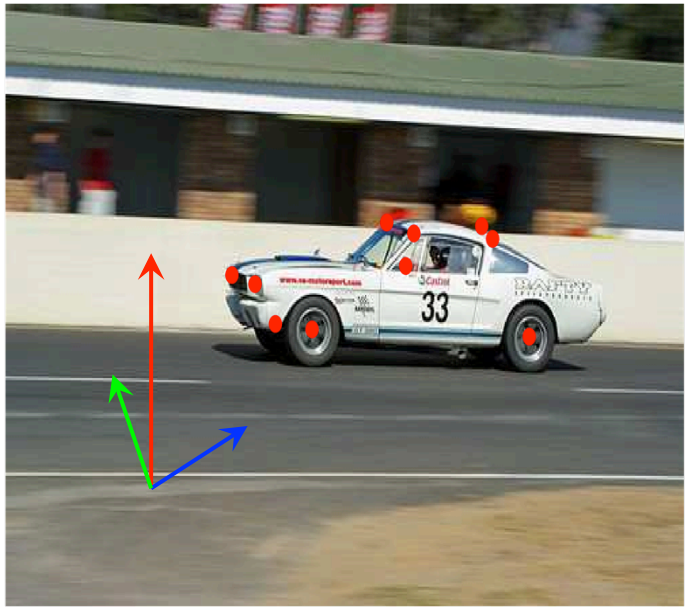


Figure 6: *Representative segmentations produced by our approach on the Princeton segmentation benchmark.*

3D object category detection: `holy grail' problem



Input data

- images
- few annotated keypoints
- *segmentation masks* → *can be provided by a CNN*

Evaluation

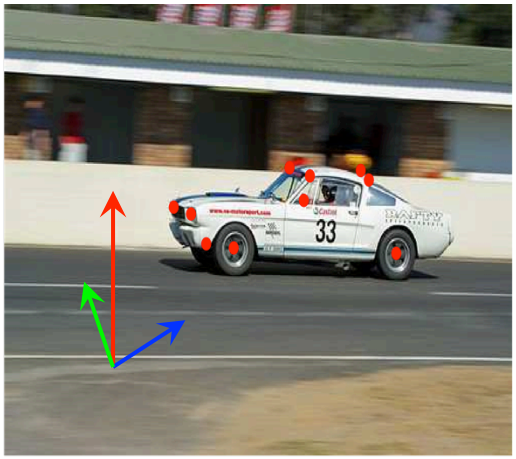
- compared to CAD models provided in Pascal VOC 3D+ [1]
- same setting as [2]

[1] Xiang, Y., Mottaghi, R., & Savarese, S. (2014, March). Beyond pascal: A benchmark for 3d object detection in the wild. In *Applications of Computer Vision (WACV), 2014 IEEE Winter Conference on* (pp. 75-82). IEEE.

[2] Kar, A., Tulsiani, S., Carreira, J., & Malik, J. (2015). Category-Specific Object Reconstruction From a Single Image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 1966-1974).

Pipeline

keypoints



viewpoint estimation
with rough 3d model

segmentation masks

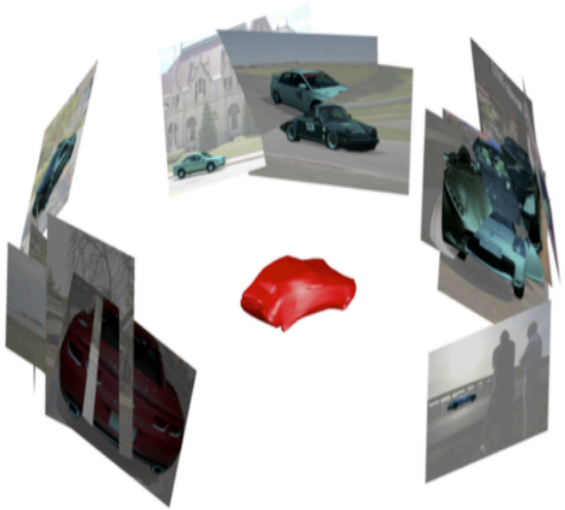
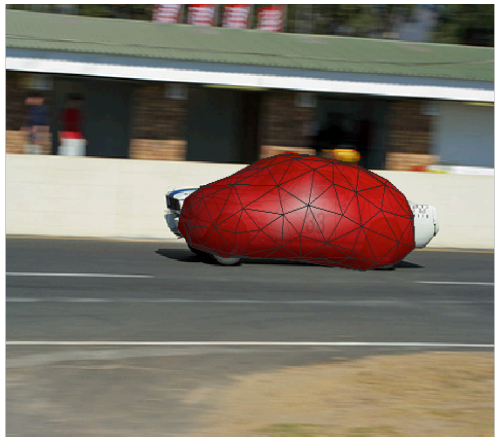


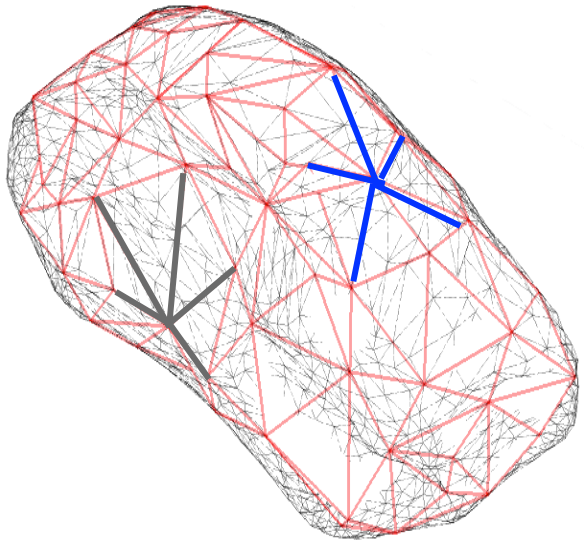
image from [2]



convex hull: 3d
shape initialization

Pipeline

Loopy DPM model



- decomposition of all edges of the 3d shape into star-shaped graphs
- **pairwise**: nominal displacements and variances for each edges estimated from training dataset
- **unaries**: affinity of each point of the mesh to the 2d image (invariant in Z)

Optimization:

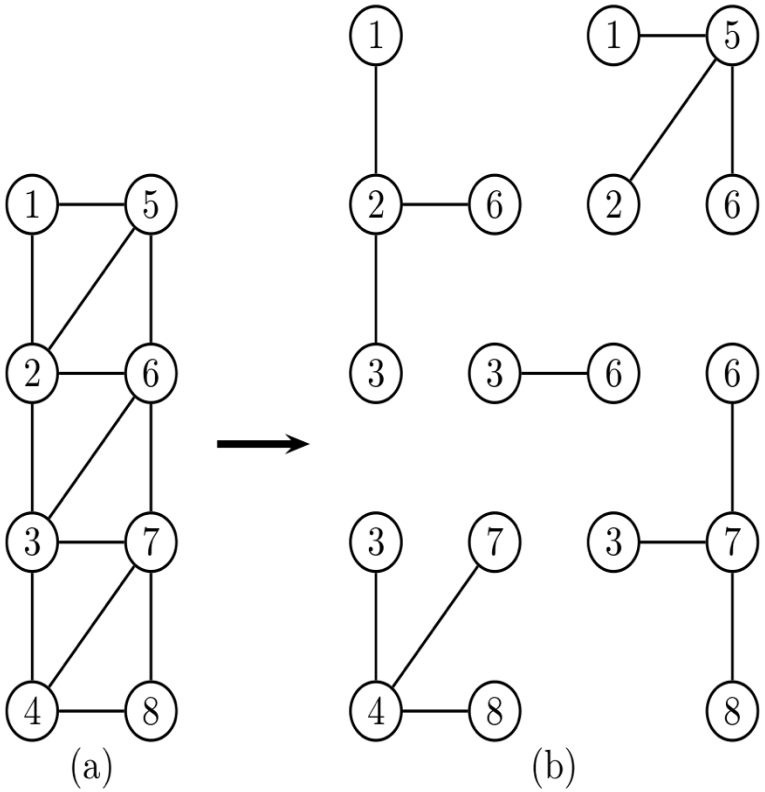
- Branch –and- bound for each slave + ADMM

Structured learning:

- first iteration: unaries only for annotated keypoints (provides ground truth)
- subsequent iterations: unaries and pairwise adjust to make the keypoints closer to the ground truth

ADMM - Branch&Bound

- In ADMM the complete graph is split into several star-shaped subgraphs



3D Branch & Bound

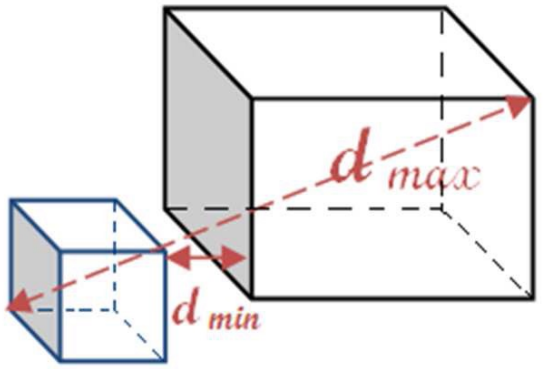
previously: every part gets a position (width*height ~ i.e. 640*480) in the 2d image

now: every part is assigned a 3d position (width*height*depth ~ 640*480*200)

Inference on a single slave with 5 parts needs less than 180ms in a 100*100*200 image space with Branch&Bound

With dynamic programming the inference takes about 1.5sec

$$\overline{\mu}_X^Y \doteq (\sum_{i \in Y} w_i) \max_{i \in Y} \max_{j \in X} K(x_i, x_j)$$



3D DPMs

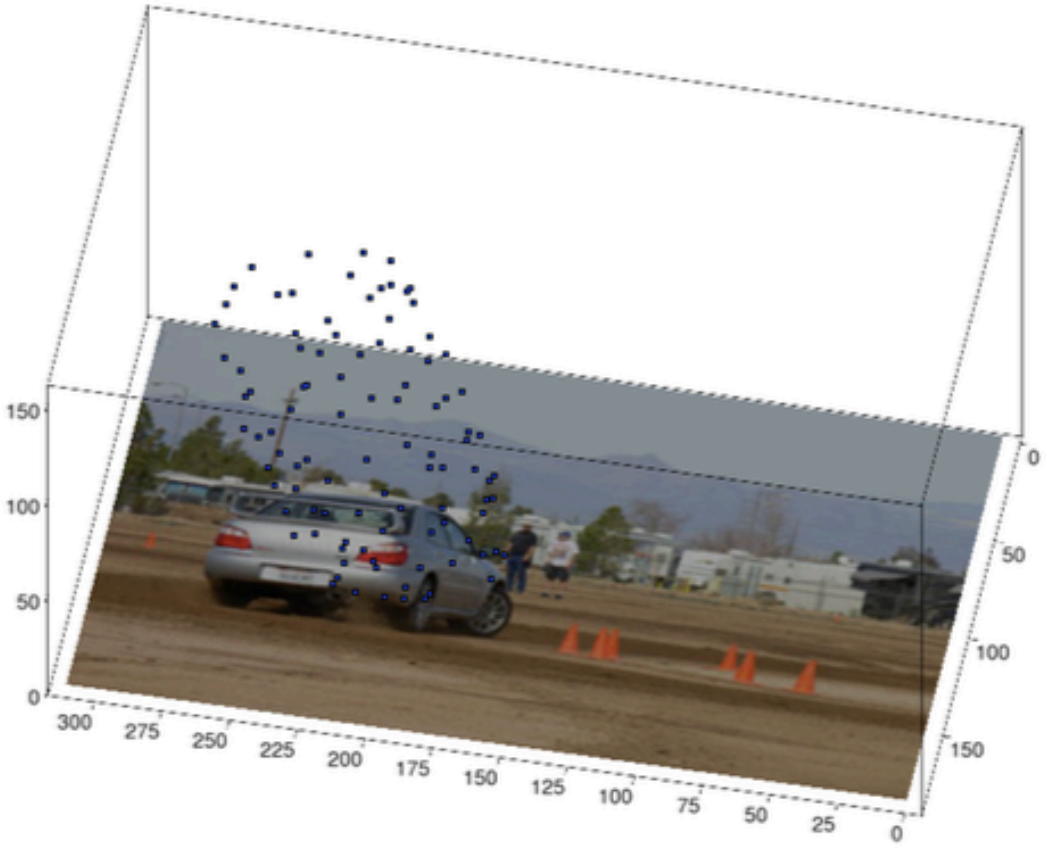


Figure 3.5: This figure shows how the algorithm reconstructs a car. The image plane containing the photo of the car is the input, the blue points represent the positions of the car model reconstructed in 3d.