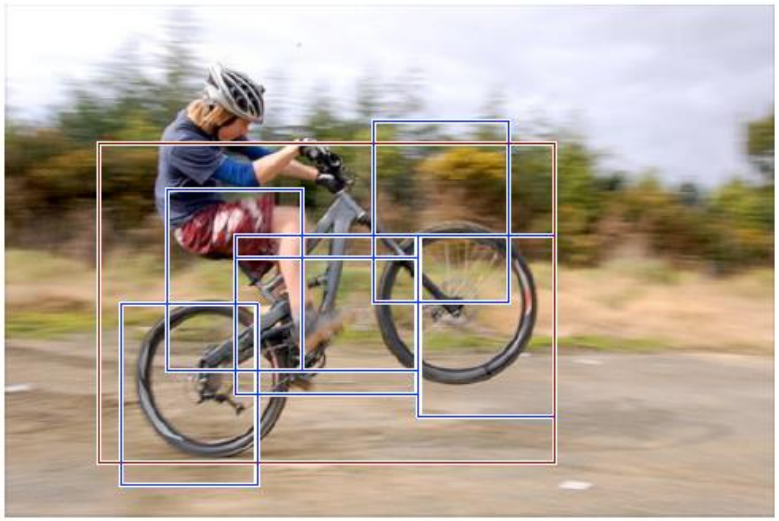
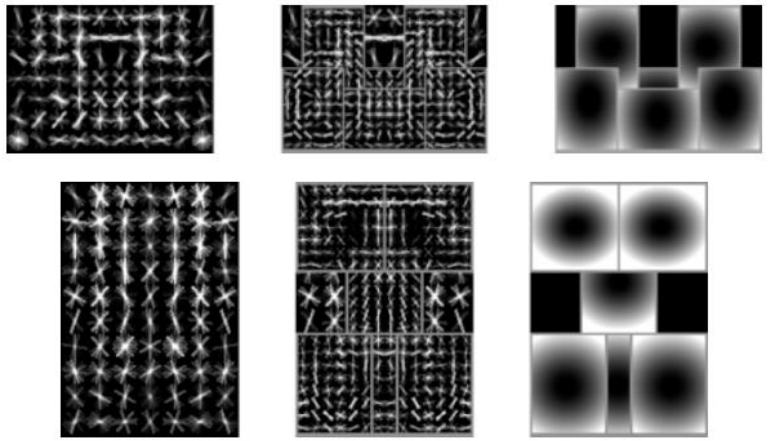
Object Detection with Part-based Models





Computer Vision

Jia-Bin Huang, Virginia Tech

Many slides from D. Hoiem, J. Hays

Administrative stuffs

- HW 5 (Scene categorization) is out
  - Due 11:59pm on Wed, November 16
- HW 3 graded
- HW 4 will be out this week
- Final project

Today's class

- **Statistical template matching**
  - Dalal-Triggs pedestrian detector (basic concept)
  - Viola-Jones detector (cascades, integral images)
  - R-CNN detector (object proposals/CNN)

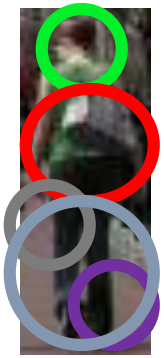
- **Deformable parts model**
  - Star-shaped model
    Example: Deformable Parts Model Felzenswalb et al. 2010
  - Tree-shaped model
    Example: Pictorial structures Felzenszwalb Huttenlocher 2005
  - Sequential prediction models

Review: Statistical template

- Object model = log linear model of parts at fixed positions

+3 +2 -2 -1 -2.5 = -0.5 ? > 7.5

Non-object



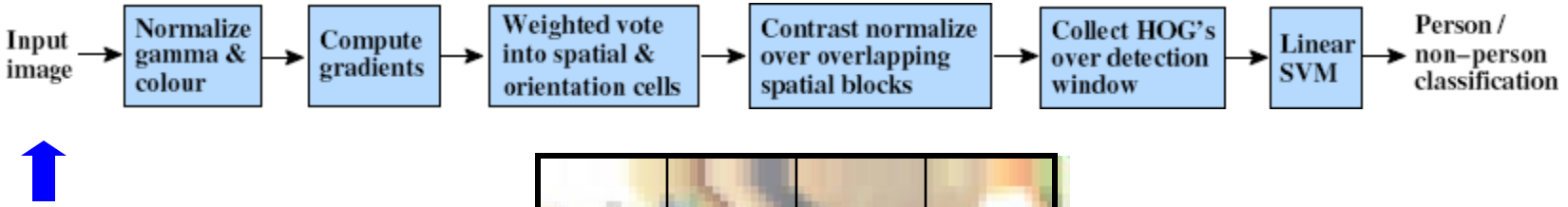
+4 +1 +0.5 +3 +0.5 = 10.5 ? > 7.5

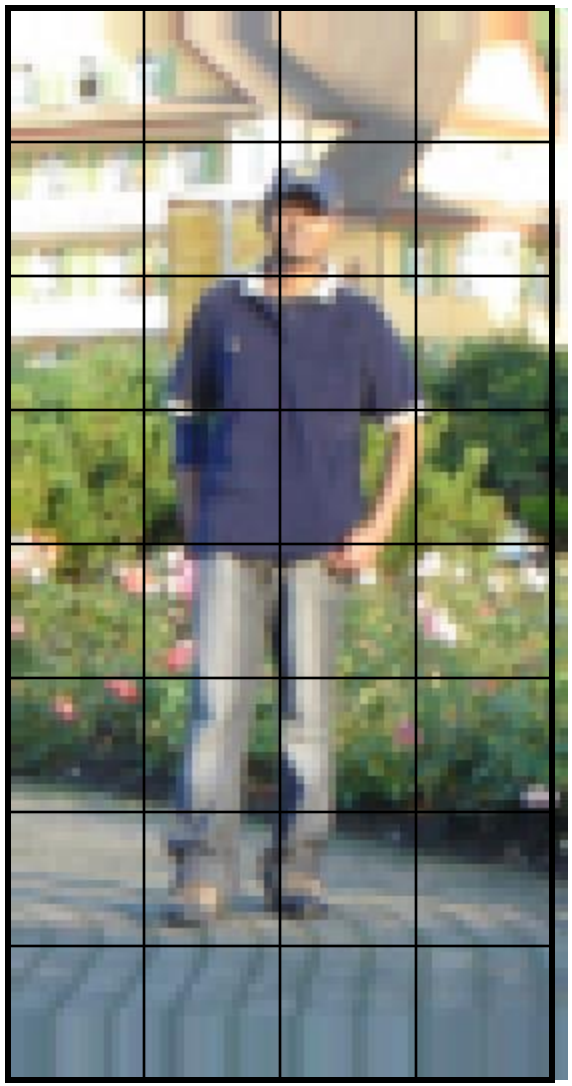
Object

Example: Dalal-Triggs pedestrian detector

1. Extract fixed-sized (64x128 pixel) window at each position and scale
2. Compute HOG (histogram of gradient) features within each window
3. Score the window with a linear SVM classifier
4. Perform non-maxima suppression to remove overlapping detections with lower scores

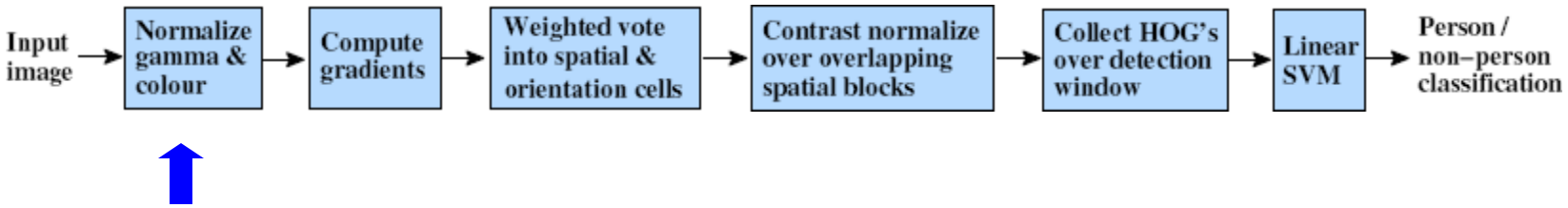
Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05





Slides by Pete Barnum

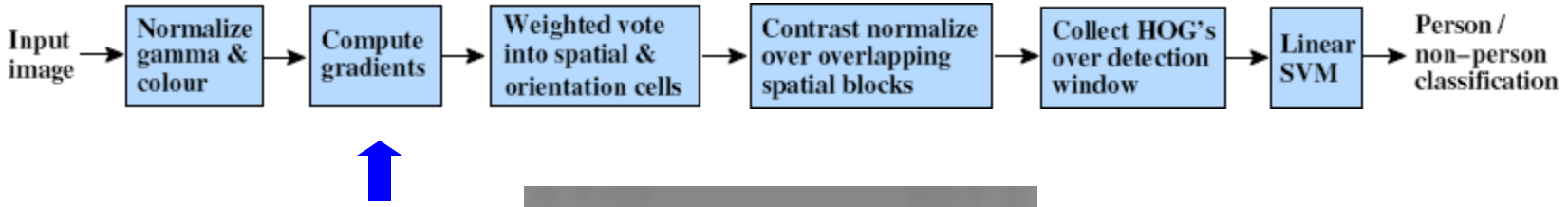
Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05



- Tested with
  - RGB
  - LAB
  - Grayscale

  } Slightly better performance vs. grayscale
- Gamma Normalization and Compression
  - Square root
  - Log

  } Very slightly better performance vs. no adjustment



Outperforms

-1	0	1

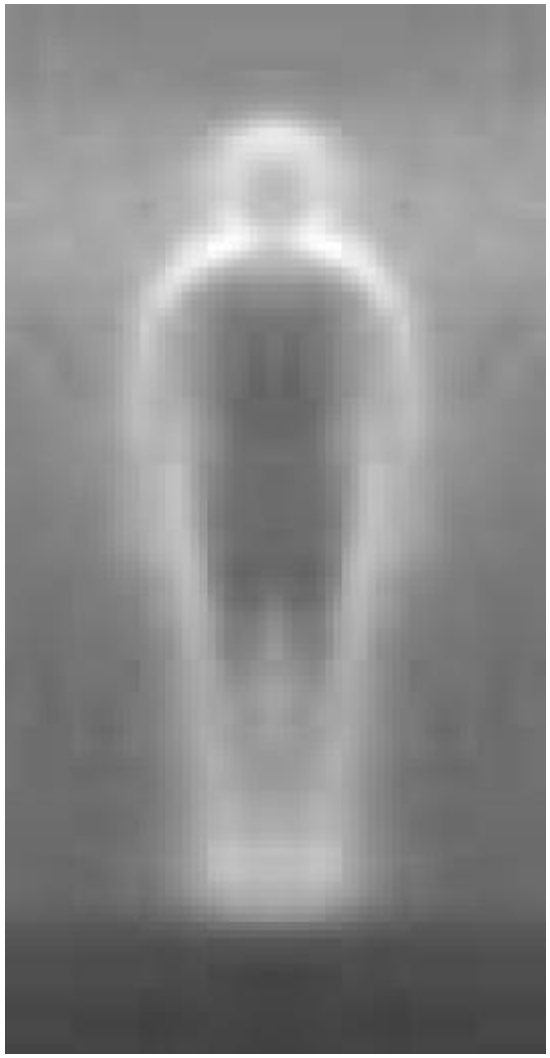
centered

-1	1

uncentered

1	-8	0	8	-1

cubic-corrected



0	1
-1	0

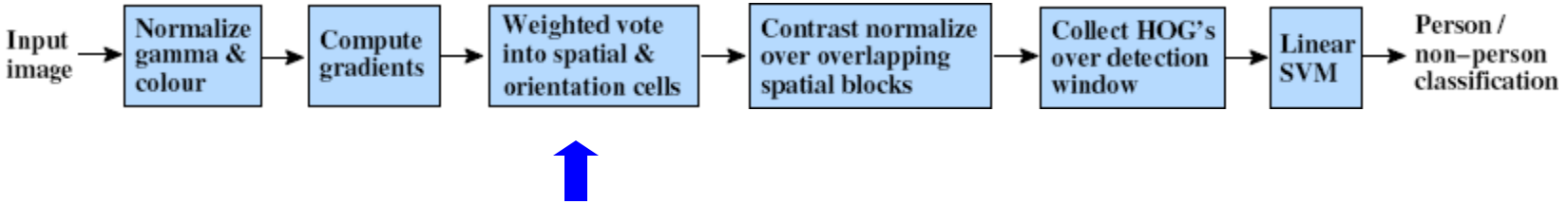
diagonal

-1	0	1
-2	0	2
-1	0	1

Sobel

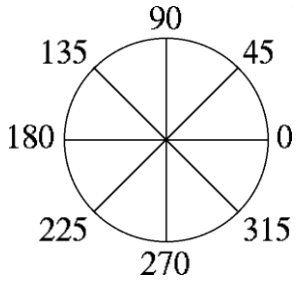
Slides by Pete Barnum

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05

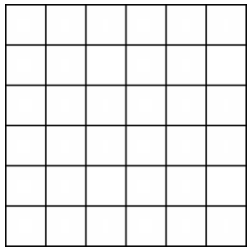


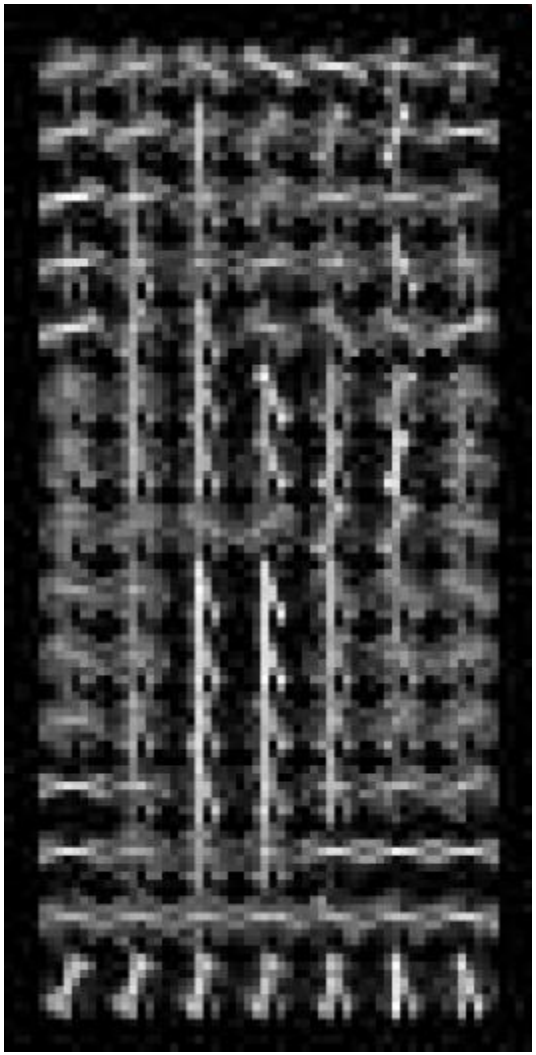
- Histogram of gradient orientations

Orientation: 9 bins (for unsigned angles)



Histograms in 8x8 pixel cells

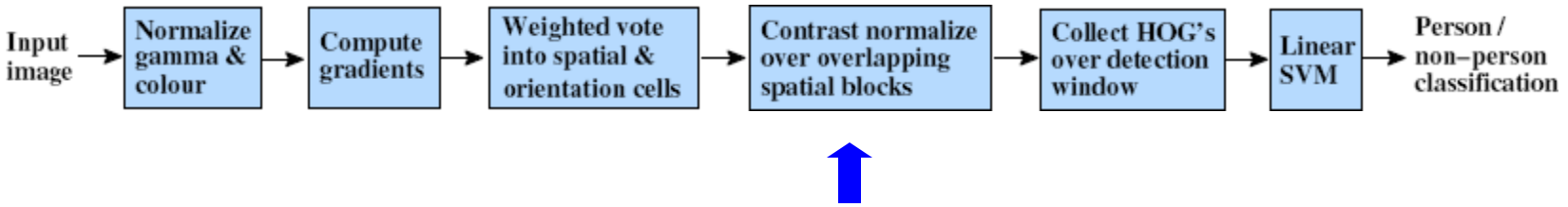




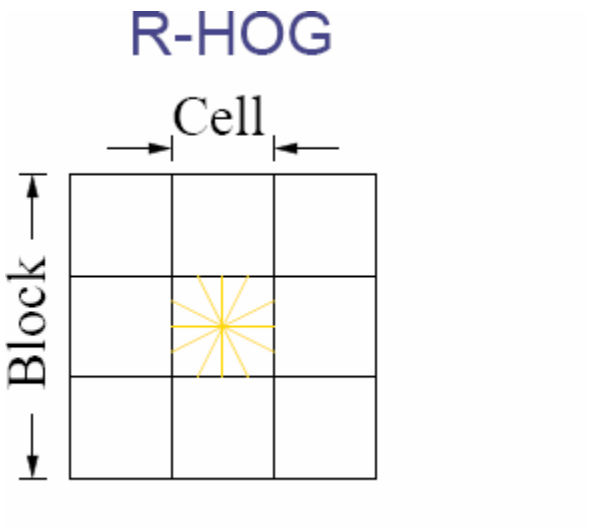
- Votes weighted by magnitude
- Bilinear interpolation between cells

Slides by Pete Barnum

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05



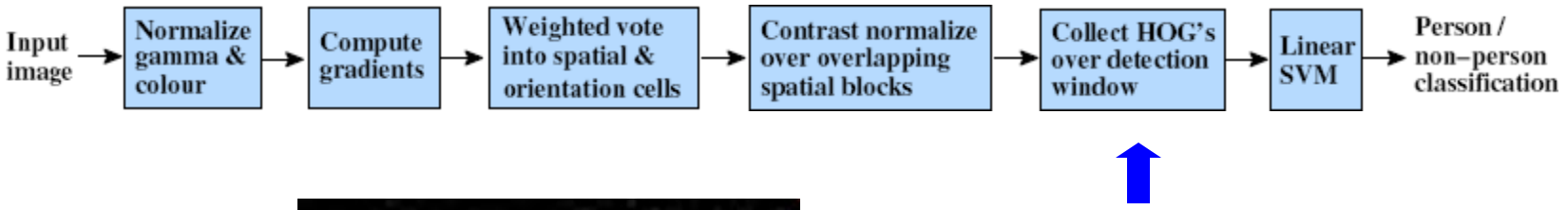
Normalize with respect to surrounding cells



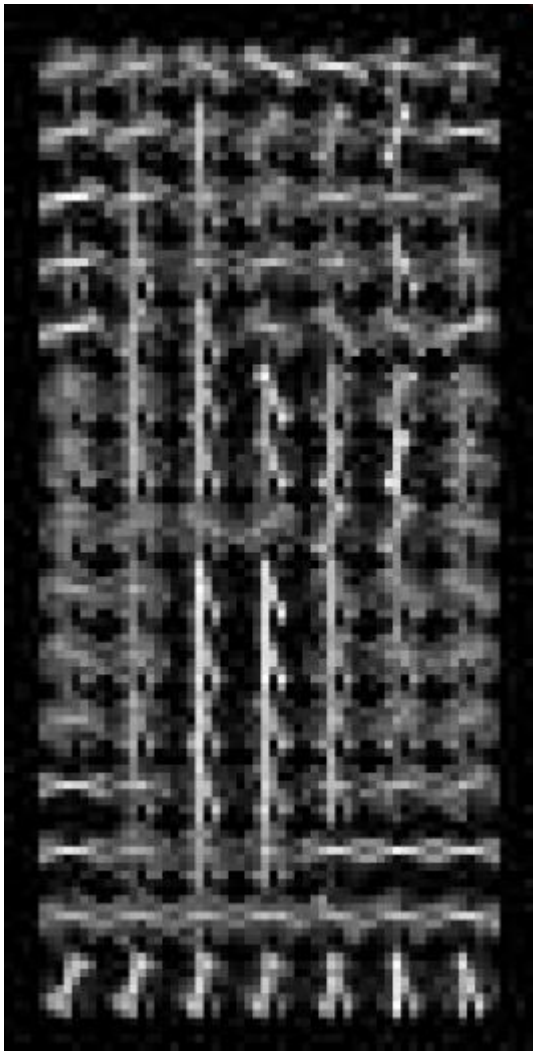
$$L2-norm: v \longrightarrow v/\sqrt{||v||_2^2+\epsilon^2}$$

Slides by Pete Barnum

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05



X=



orientations

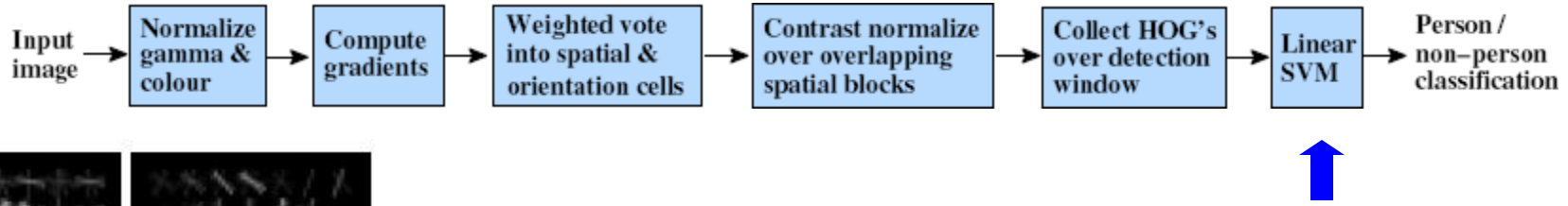
features = 15 x 7 x 9 x 4 = 3780

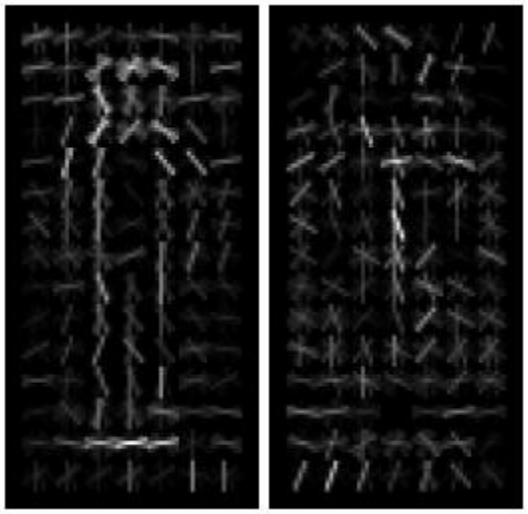
cells

normalizations by neighboring cells

Slides by Pete Barnum

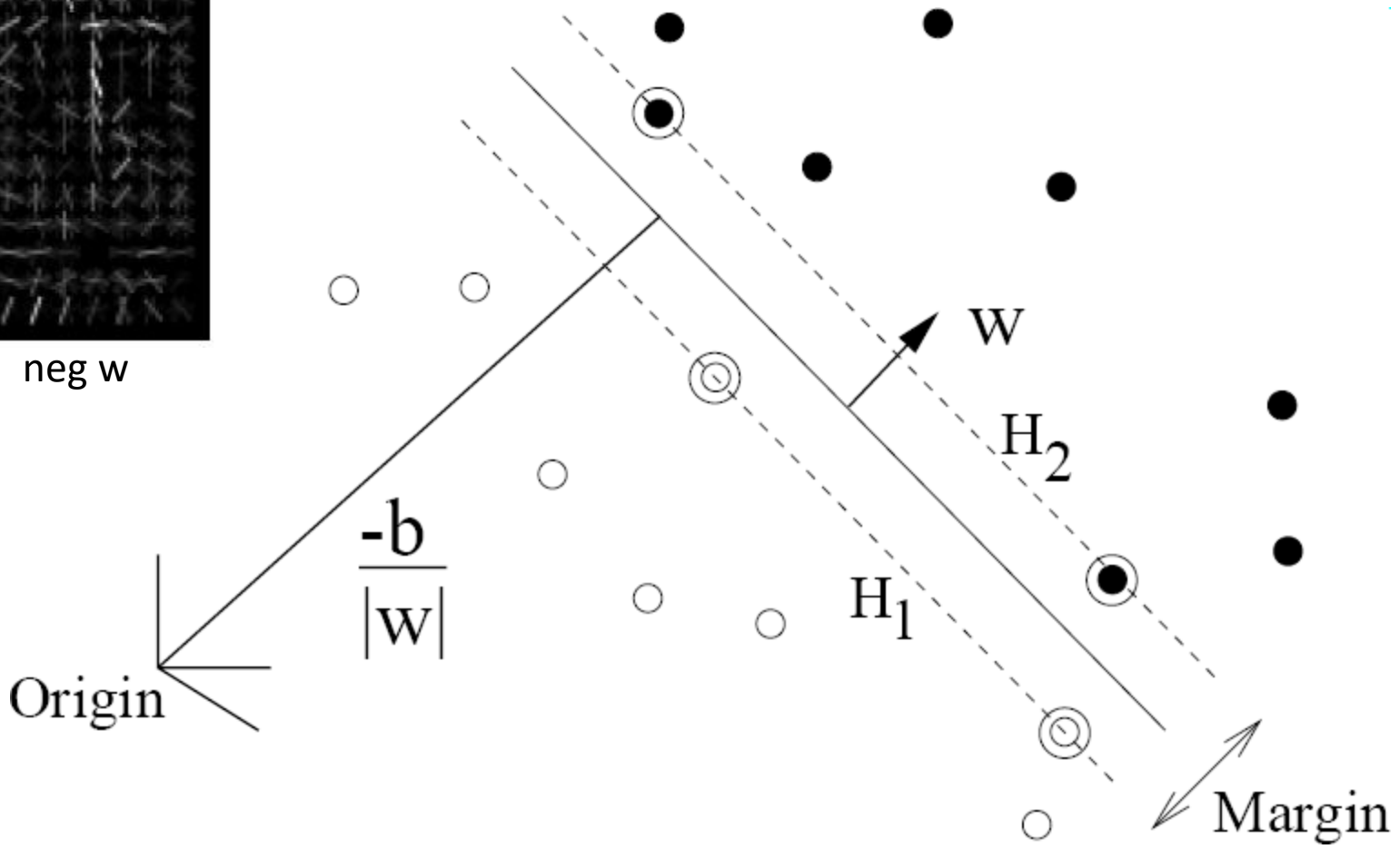
Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05





pos w

neg w



Slides by Pete Barnum

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05

$$0.16 = w^T x - b$$

$$sign(0.16) = 1$$

$$=>$$ pedestrian

Slides by Pete Barnum

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05

Detection examples

Something to think about…

- Sliding window detectors work
  - *very well* for faces
  - *fairly well* for cars and pedestrians
  - *badly* for cats and dogs

- Why are some classes easier than others?

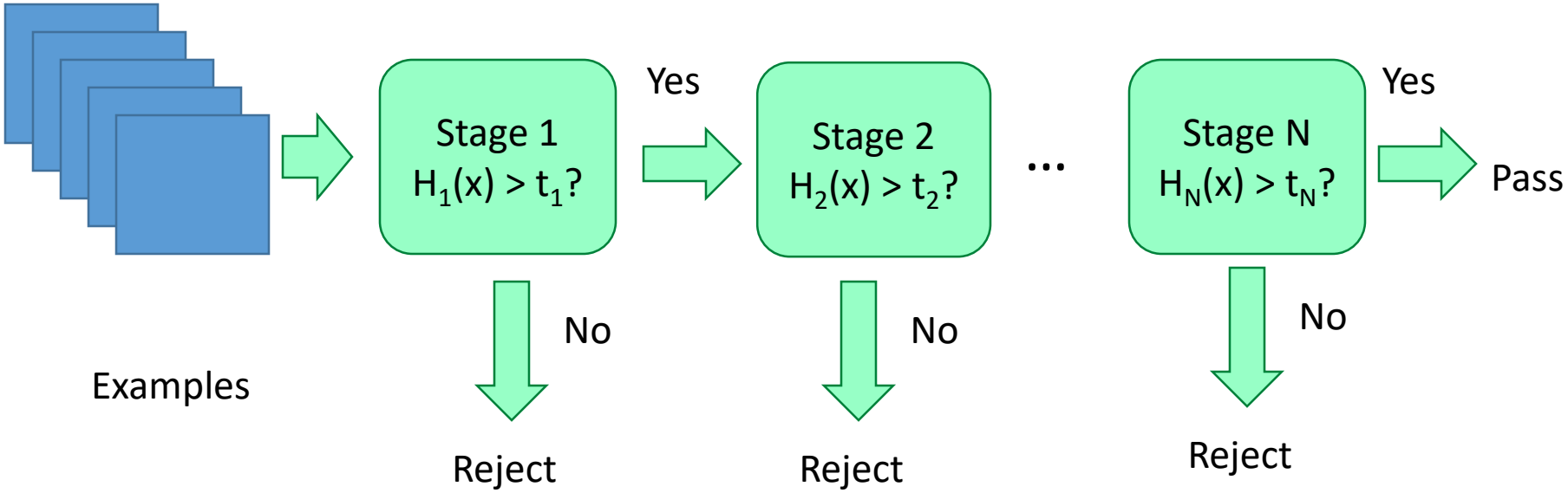
Viola-Jones sliding window detector

Fast detection through two mechanisms

- Quickly eliminate unlikely windows
- Use features that are fast to compute

Viola and Jones. Rapid Object Detection using a Boosted Cascade of Simple Features (2001).

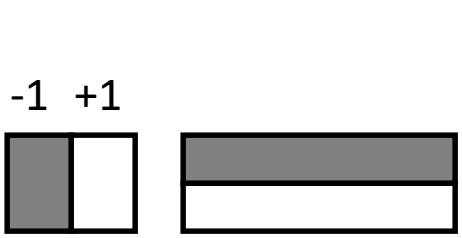
Cascade for Fast Detection



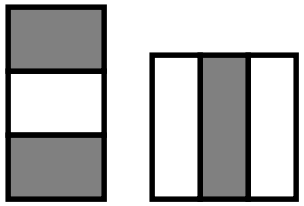
- Choose threshold for low false negative rate
- Fast classifiers early in cascade
- Slow classifiers later, but most examples don't get there

Features that are fast to compute

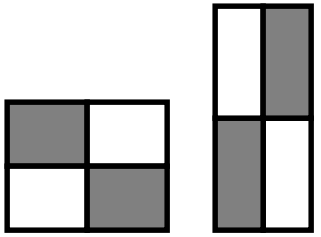
- "Haar-like features"
  - Differences of sums of intensity
  - Thousands, computed at various positions and scales within detection window



Two-rectangle features



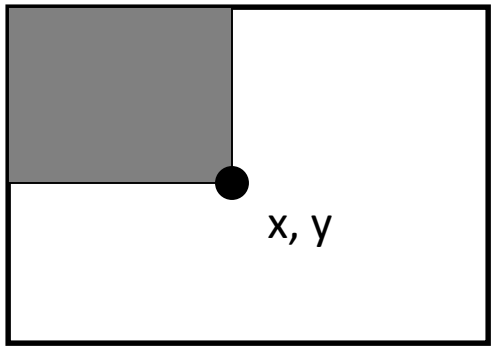
Three-rectangle features



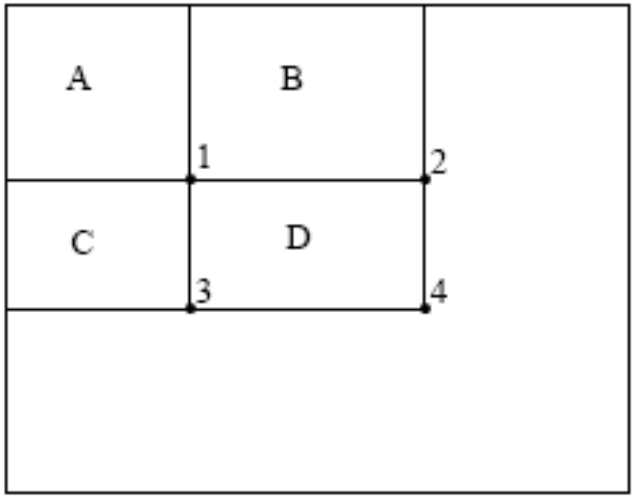
Etc.

Integral Images

- `ii = cumsum(cumsum(im, 1), 2)`



ii(x,y) = Sum of the values in the grey region



How to compute B-A?

How to compute A+D-B-C?

Feature selection with Adaboost

- Create a large pool of features (180K)
- Select features that are discriminative and work well together
  - "Weak learner" = feature + threshold + parity

$$h_j(x) = \begin{cases} 1 & \text{if } p_j f_j(x) < p_j \theta_j \\ 0 & \text{otherwise} \end{cases}$$

  - Choose weak learner that minimizes error on the weighted training set
  - Reweight

Adaboost

- Given example images $(x_1, y_1), \ldots, (x_n, y_n)$ where $y_i = 0, 1$ for negative and positive examples respectively.
- Initialize weights $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ for $y_i = 0, 1$ respectively, where m and l are the number of negatives and positives respectively.
- For $t = 1, \ldots, T$:

  1. Normalize the weights,

     $$w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}$$

     so that w_t is a probability distribution.

  2. For each feature, j, train a classifier h_j which is restricted to using a single feature. The error is evaluated with respect to w_t, $\epsilon_j = \sum_i w_i \left| h_j(x_i) - y_i \right|$.
  3. Choose the classifier, h_t, with the lowest error ϵ_t.
  4. Update the weights:

     $$w_{t+1,i} = w_{t,i} \beta_t^{1-e_i}$$

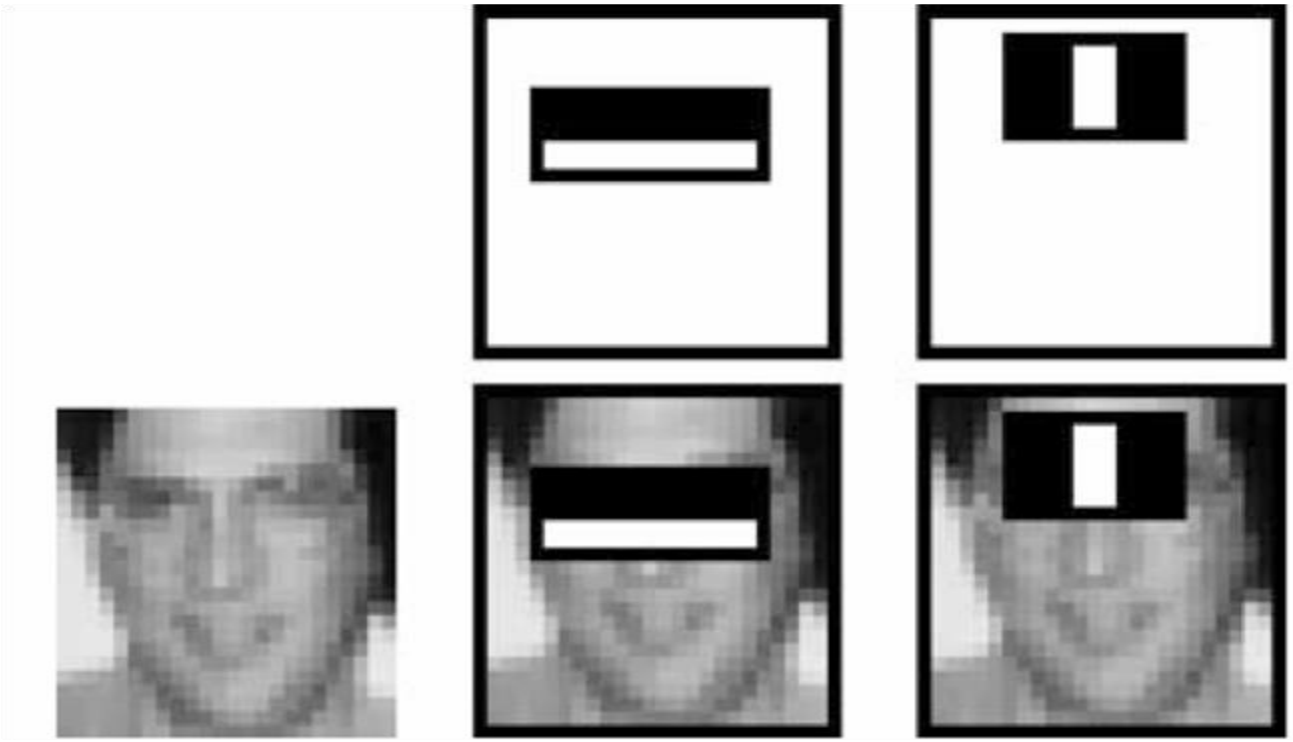
     where $e_i = 0$ if example x_i is classified correctly, $e_i = 1$ otherwise, and $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$.

- The final strong classifier is:

$$h(x) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & \text{otherwise} \end{cases}$$

where $\alpha_t = \log \frac{1}{\beta_t}$

Top 2 selected features

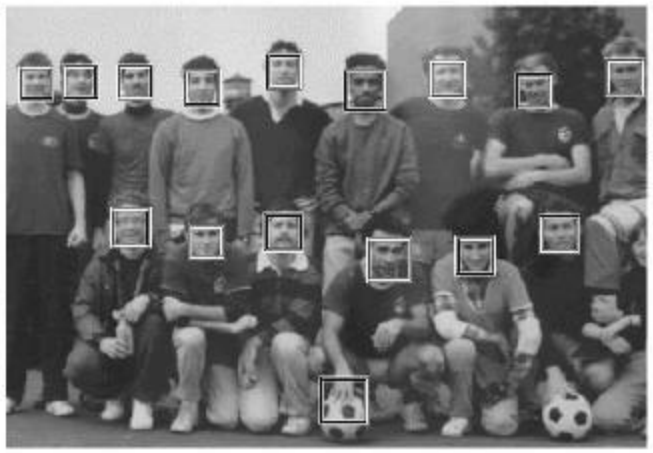


Viola-Jones details

- 38 stages with 1, 10, 25, 50 … features
  - 6061 total used out of 180K candidates
  - 10 features evaluated on average
- Training Examples
  - 4916 positive examples
  - 10000 negative examples collected after each stage
- Scanning
  - Scale detector rather than image
  - Scale steps = 1.25 (factor between two consecutive scales)
  - Translation 1*scale (# pixels between two consecutive windows)
- Non-max suppression: average coordinates of overlapping boxes
- Train 3 classifiers and take vote

Viola Jones Results

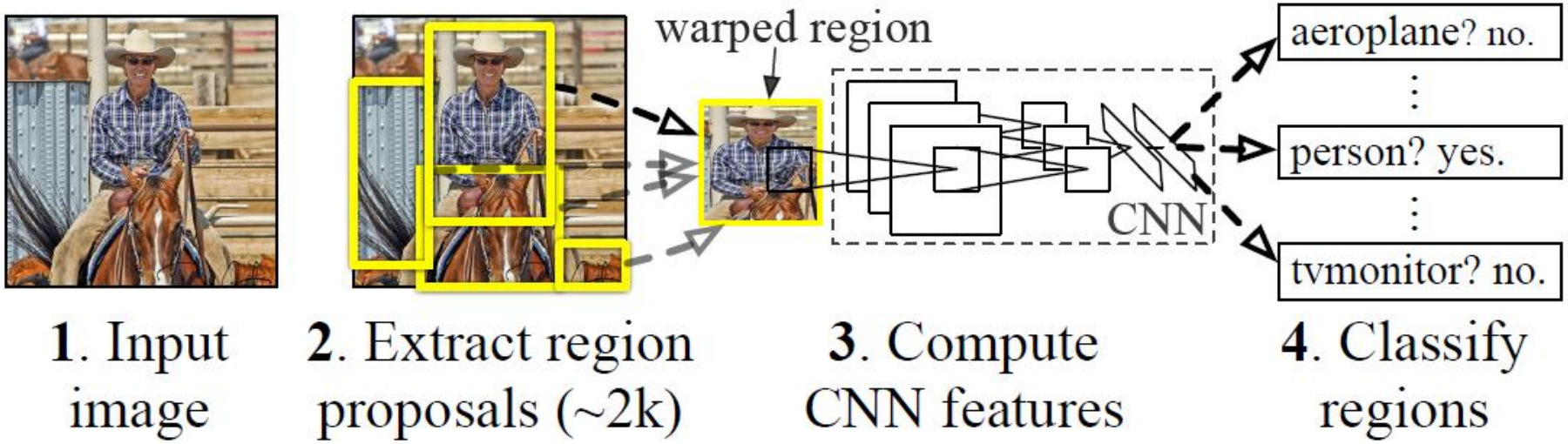
Speed = 15 FPS (in 2001)


Detector \ False detections	10	31	50	65	78	95	167
Viola-Jones	76.1%	88.4%	91.4%	92.0%	92.1%	92.9%	93.9%
Viola-Jones (voting)	81.1%	89.7%	92.1%	93.1%	93.1%	93.2 %	93.7%
Rowley-Baluja-Kanade	83.2%	86.0%	-	-	-	89.2%	90.1%
Schneiderman-Kanade	-	-	-	94.4%	-	-	-
Roth-Yang-Ahuja	-	-	-	-	(94.8%)	-	-

MIT + CMU face dataset

R-CNN (Girshick et al. CVPR 2014)



- Replace sliding windows with "selective search" region proposals (Uijilings et al. IJCV 2013)
- Extract rectangles around regions and resize to 227x227
- Extract features with fine-tuned CNN (that was initialized with network trained on ImageNet before training)
- Classify last layer of network features with SVM

https://people.eecs.berkeley.edu/~rbg/papers/r-cnn-cvpr.pdf

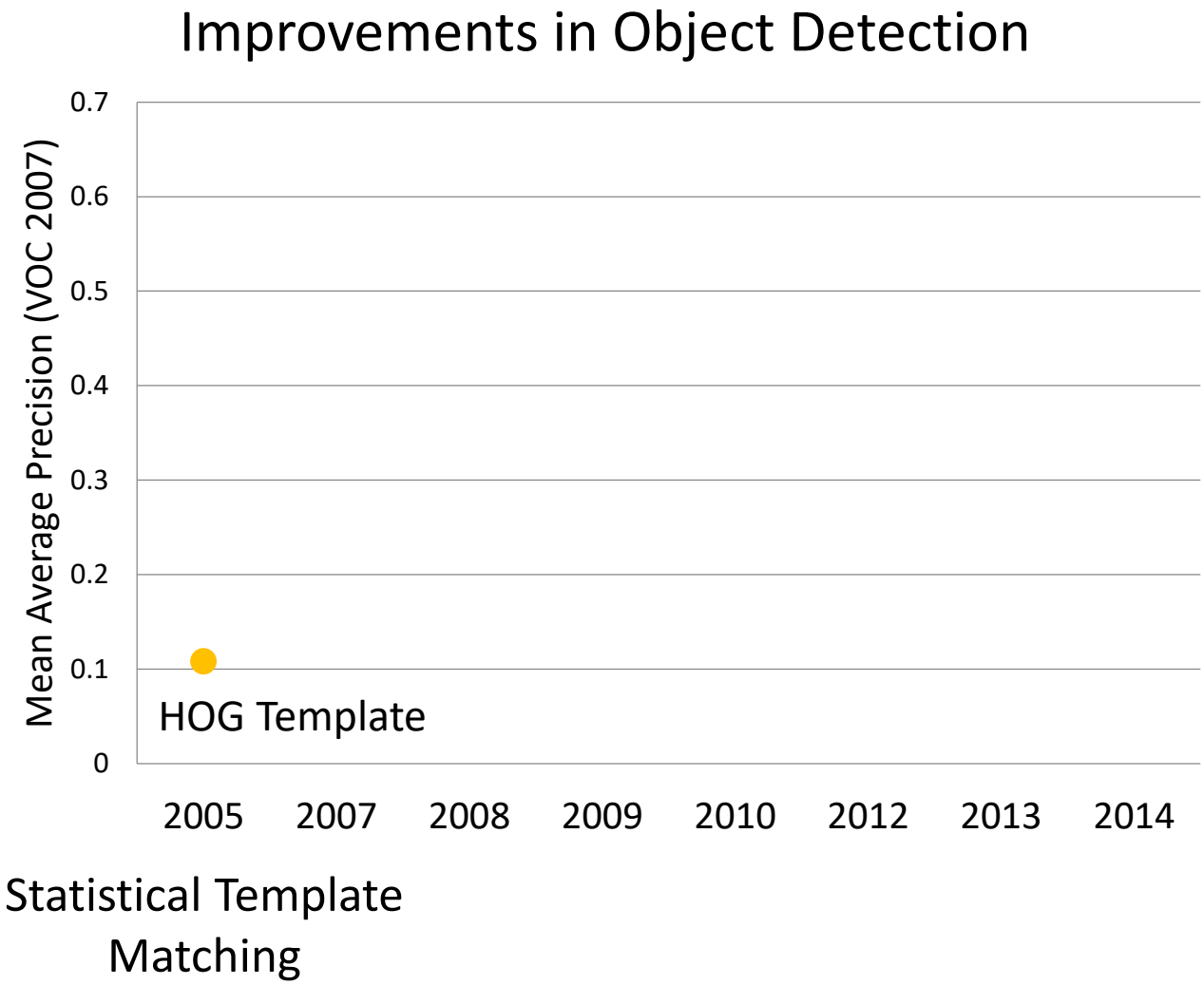
Sliding window vs. region proposals

Sliding window

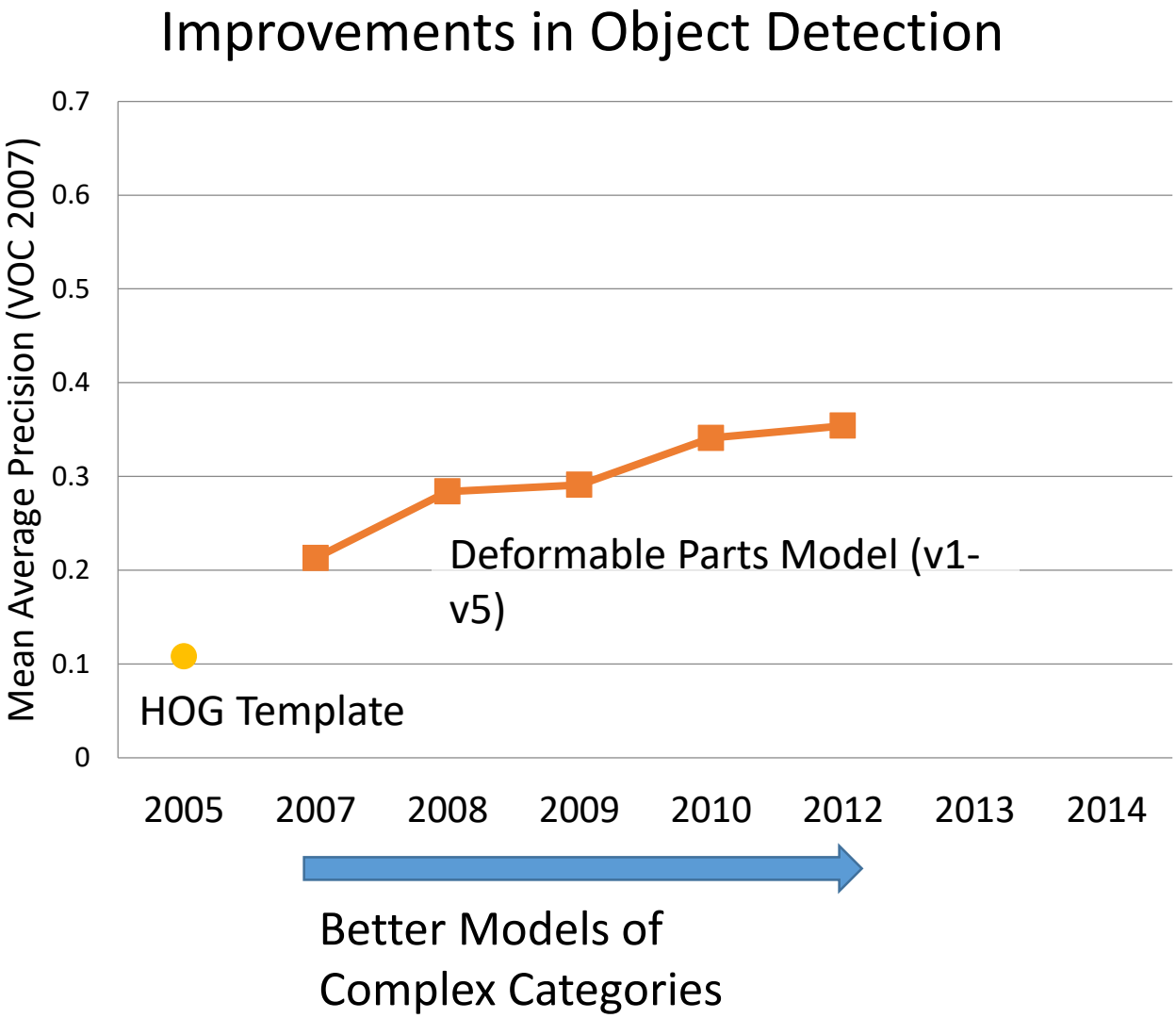
- Comprehensive search over position, scale (sometimes aspect, though expensive)
- Typically 100K candidates
- Simple
- Speed boost through convolution often possible
- Repeatable
- Even with many candidates, may not be a good fit to object

Region proposals

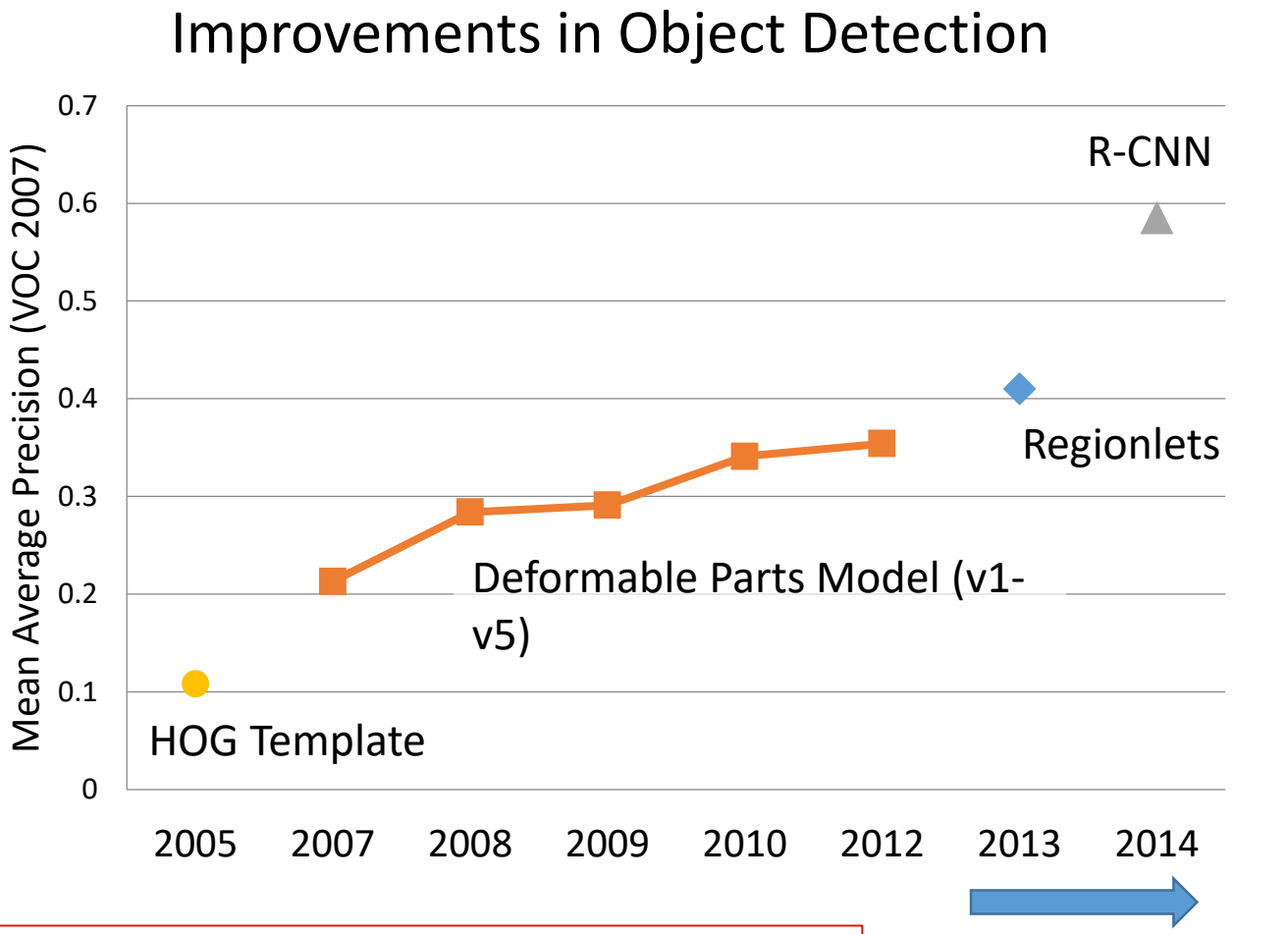
- Search over regions guided by image contours/patterns with varying aspect/size
- Typically 2-10K candidates
- Random (not repeatable)
- Requires a preprocess (currently 1-5s)
- Often requires resizing patch to fit fixed size
- More likely to provide candidates with very good object fit



HOG: Dalal-Triggs 2005



HOG: Dalal-Triggs 2005 DPM: Felzenszwalb et al. 2008-2012

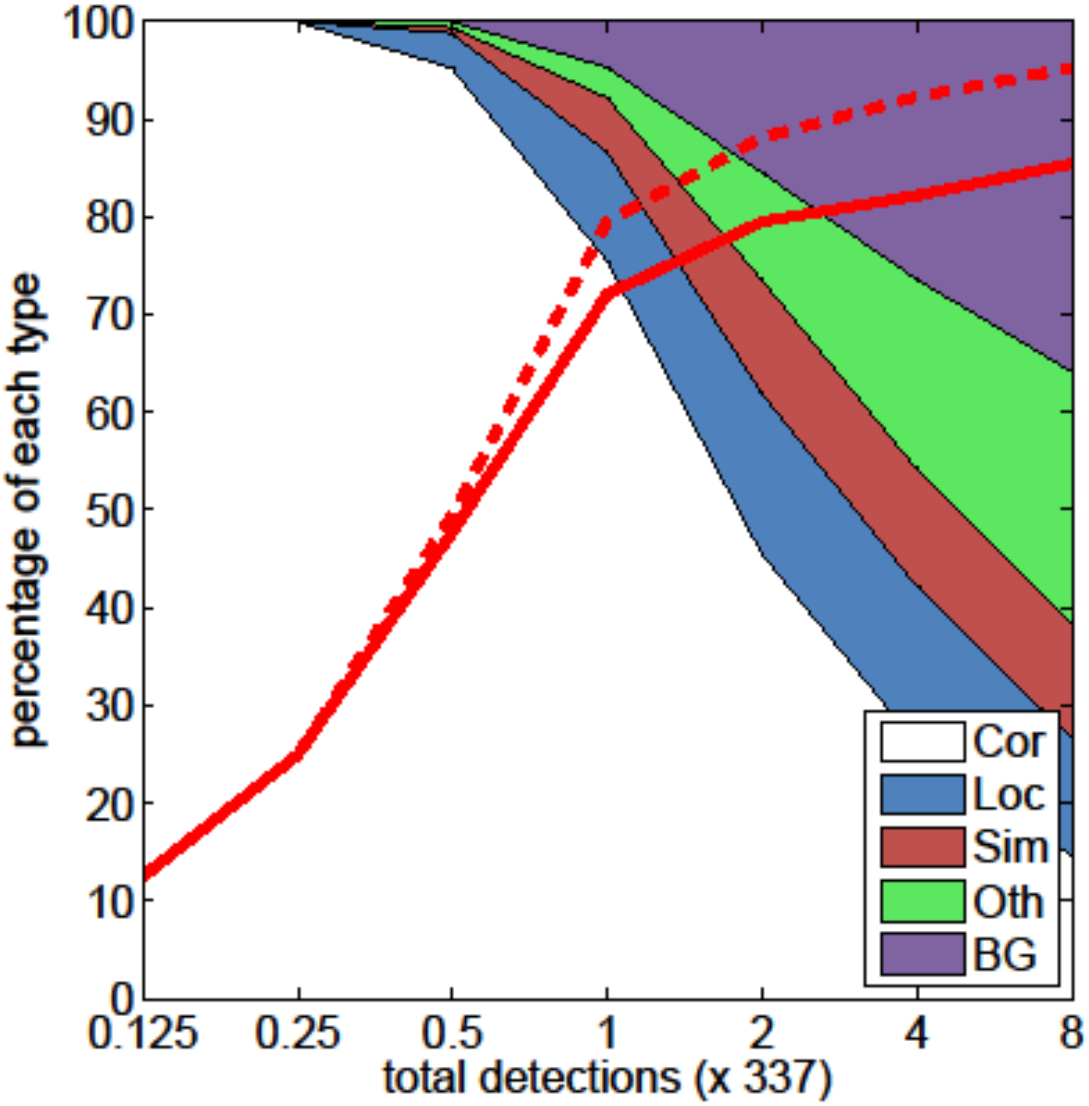


Key Advance: Learn effective features from massive amounts of labeled data *and* adapt to new tasks with less data

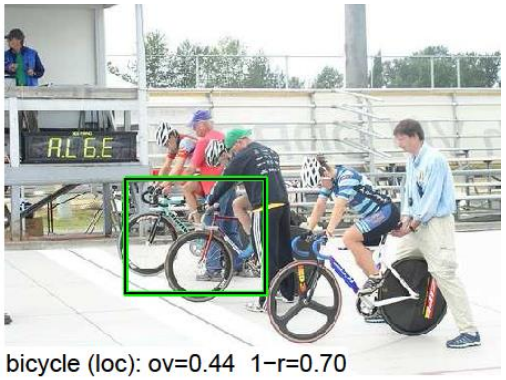
HOG: Dalal-Triggs 2005 DPM: Felzenszwalb et al. 2008-2012 Regionlets: Wang et al. 2013 R-CNN: Girshick et al. 2014

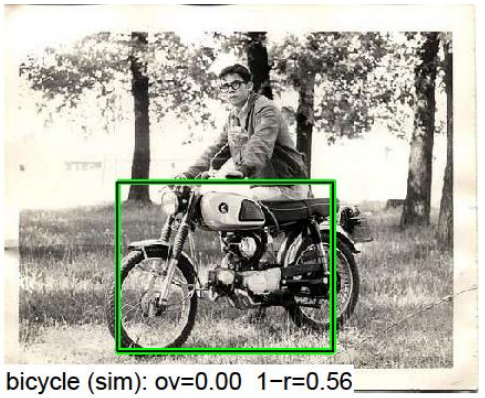
Mistakes are often reasonable

Bicycle: AP = 0.73



Confident Mistakes

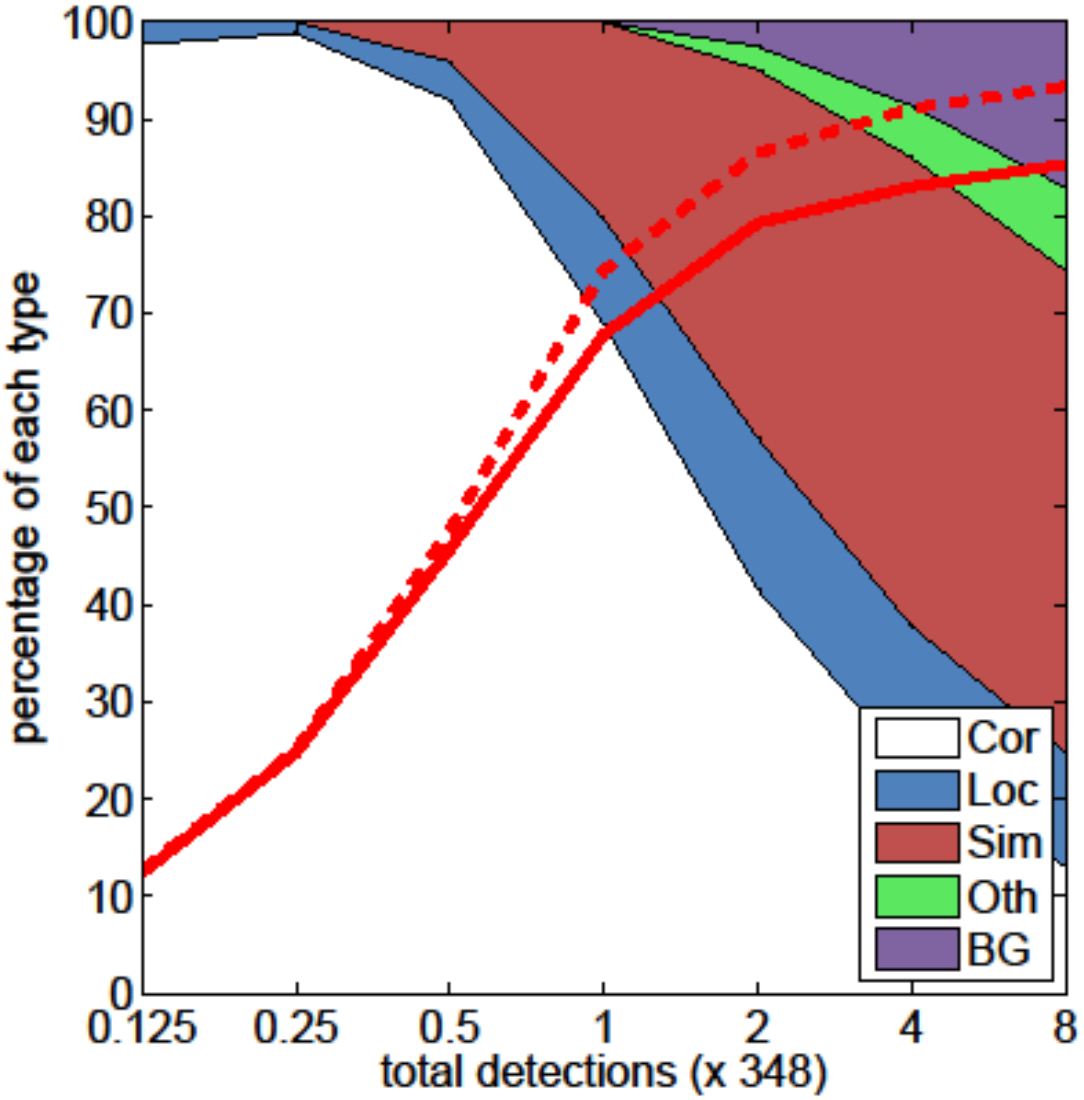




R-CNN results

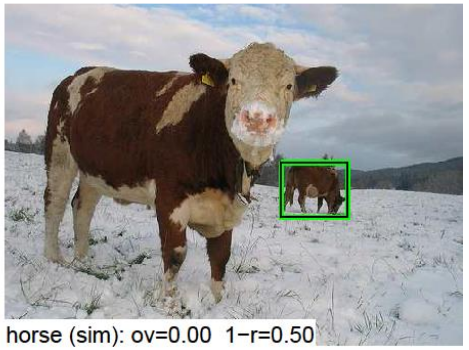
Mistakes are often reasonable

Horse: AP = 0.69

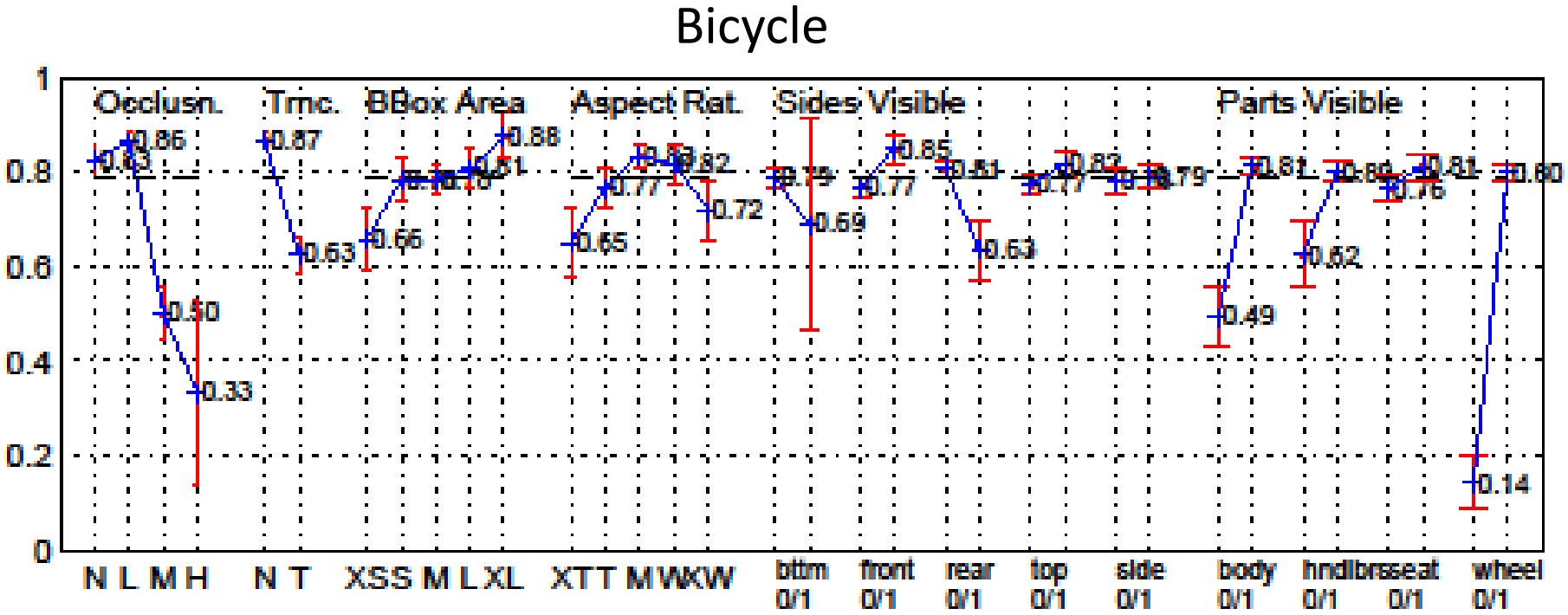


R-CNN results

Confident Mistakes



Misses are often predictable



Small objects, distinctive parts absent or occluded, unusual views

R-CNN results

Strengths and Weaknesses of Statistical Template Approach

Strengths

- Works very well for non-deformable objects: faces, cars, upright pedestrians
- Fast detection

Weaknesses

- Sliding window has difficulty with deformable objects (proposals works with flexible features works better)
- Not robust to occlusion
- Requires lots of training data

Tricks of the trade

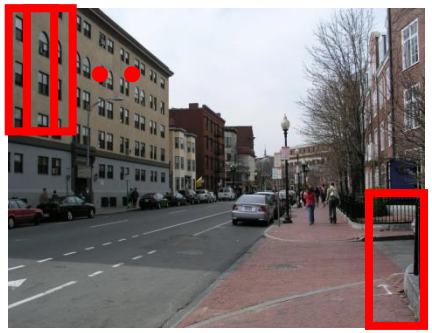
- Details in feature computation really matter
  - E.g., normalization in Dalal-Triggs improves detection rate by 27% at fixed false positive rate
- Template size
  - Typical choice for sliding window is size of smallest detectable object
  - For CNNs, typically based on what pretrained features are available
- "Jittering" to create synthetic positive examples
  - Create slightly rotated, translated, scaled, mirrored versions as extra positive examples
- Bootstrapping to get hard negative examples
  1. Randomly sample negative examples
  2. Train detector
  3. Sample negative examples that score > -1
  4. Repeat until all high-scoring negative examples fit in memory

Influential Works in Detection

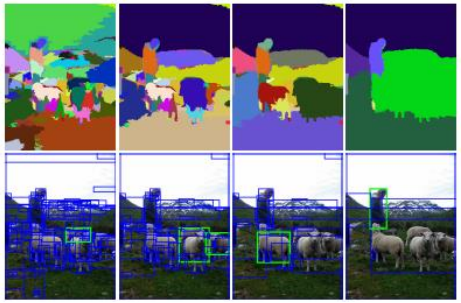
- Sung-Poggio (1994, 1998) : ~2100 citations
  - Basic idea of statistical template detection (I think), bootstrapping to get "face-like" negative examples, multiple whole-face prototypes (in 1994)
- Rowley-Baluja-Kanade (1996-1998) : ~4200
  - "Parts" at fixed position, non-maxima suppression, simple cascade, rotation, pretty good accuracy, fast
- Schneiderman-Kanade (1998-2000,2004) : ~2250
  - Careful feature/classifier engineering, excellent results, cascade
- Viola-Jones (2001, 2004) : ~20,000
  - Haar-like features, Adaboost as feature selection, hyper-cascade, very fast, easy to implement
- Dalal-Triggs (2005) : ~11000
  - Careful feature engineering, excellent results, HOG feature, online code
- Felzenszwalb-Huttenlocher (2000): ~1600
  - Efficient way to solve part-based detectors
- Felzenszwalb-McAllester-Ramanan (2008,2010)? ~4000
  - Excellent template/parts-based blend
- Girshick-Donahue-Darrell-Malik (2014) ~300
  - Region proposals + fine-tuned CNN features (marks significant advance in accuracy over hog-based methods)

Summary: statistical templates

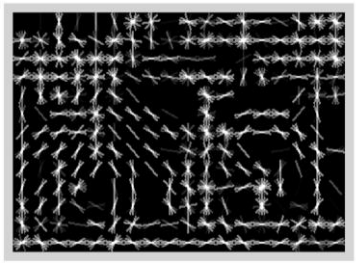
A. Propose Window → B. Extract Features → C. Classify → D. Post-process



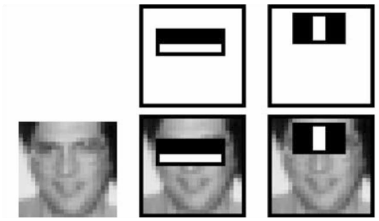
Sliding window: scan image pyramid



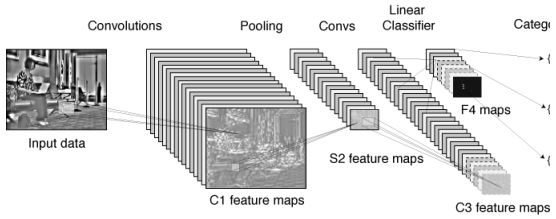
Region proposals: edge/region-based, resize to fixed window



HOG



Fast randomized features



CNN features

SVM

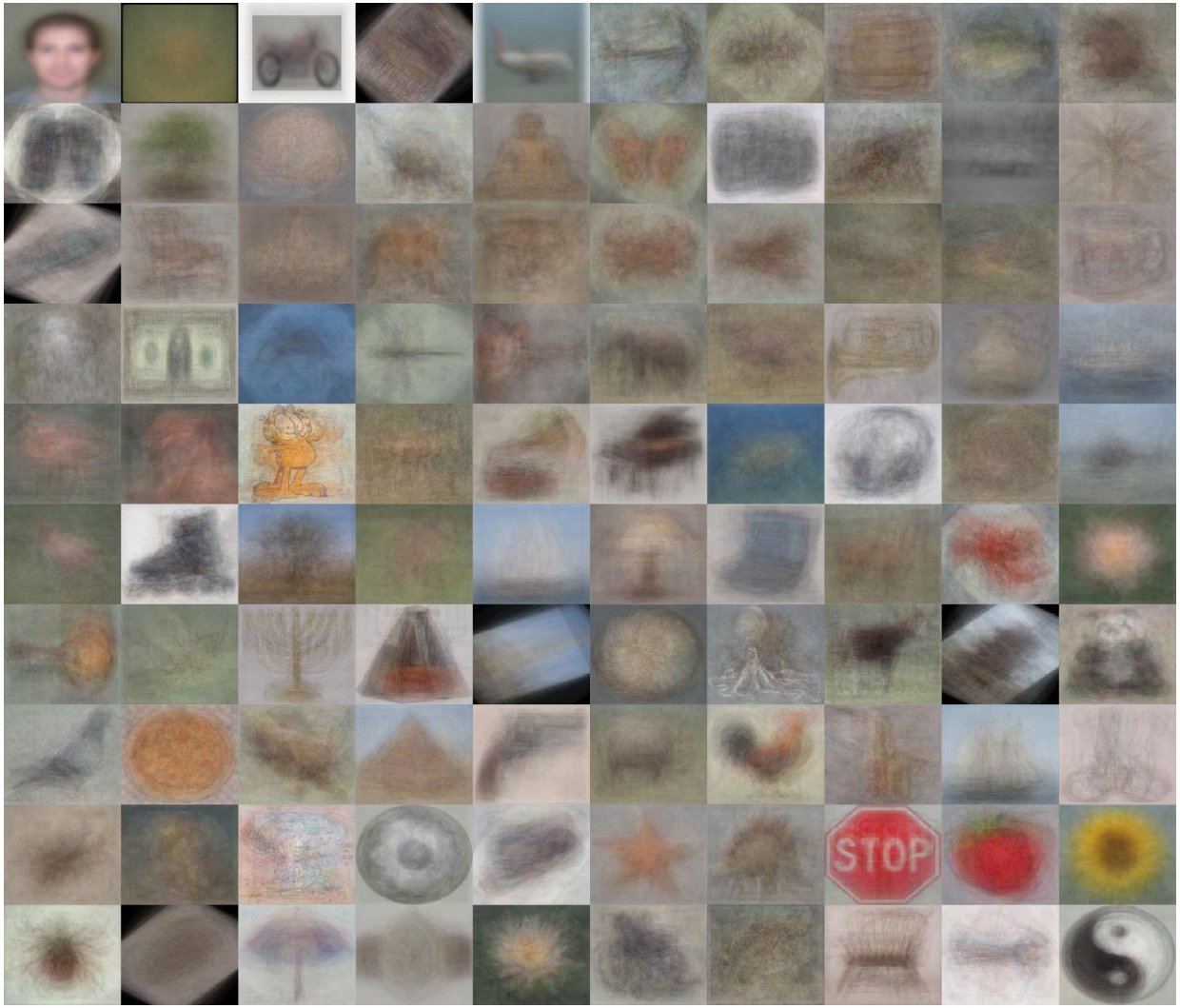
Boosted stubs

Neural network

Non-max suppression

Segment or refine localization

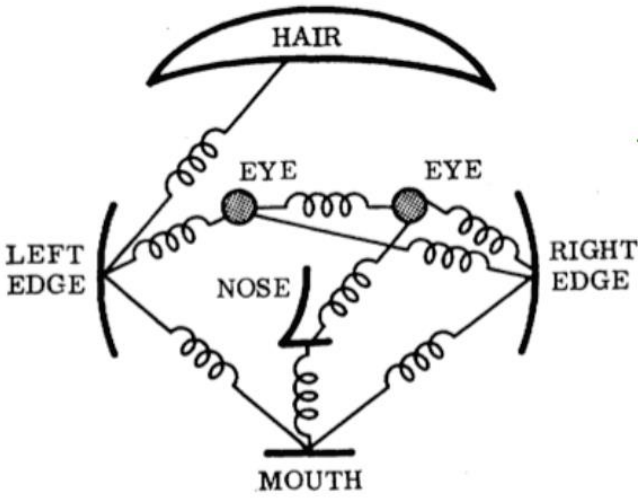
When do statistical templates make sense?

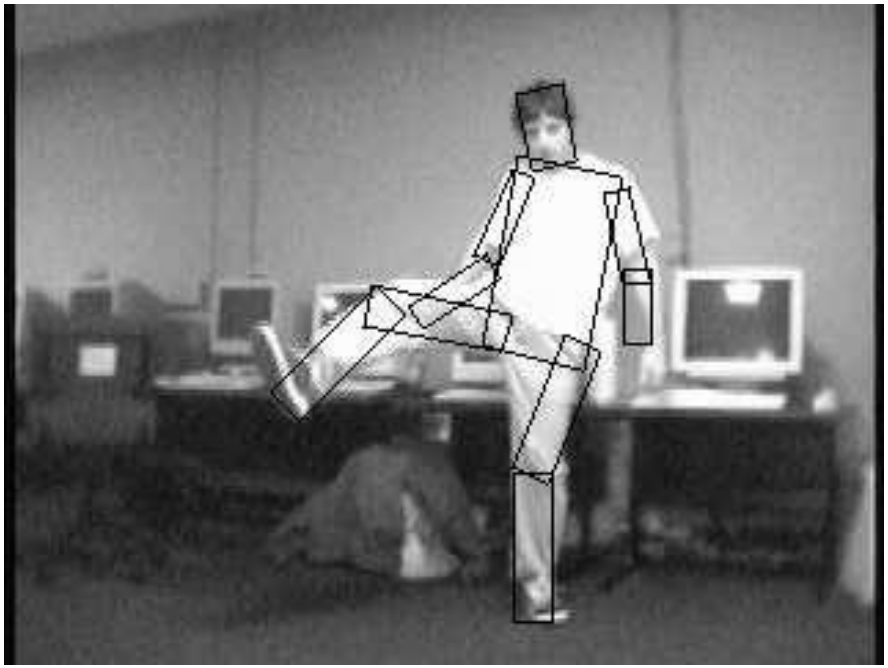


Caltech 101 Average Object Images

Object models: Articulated parts model

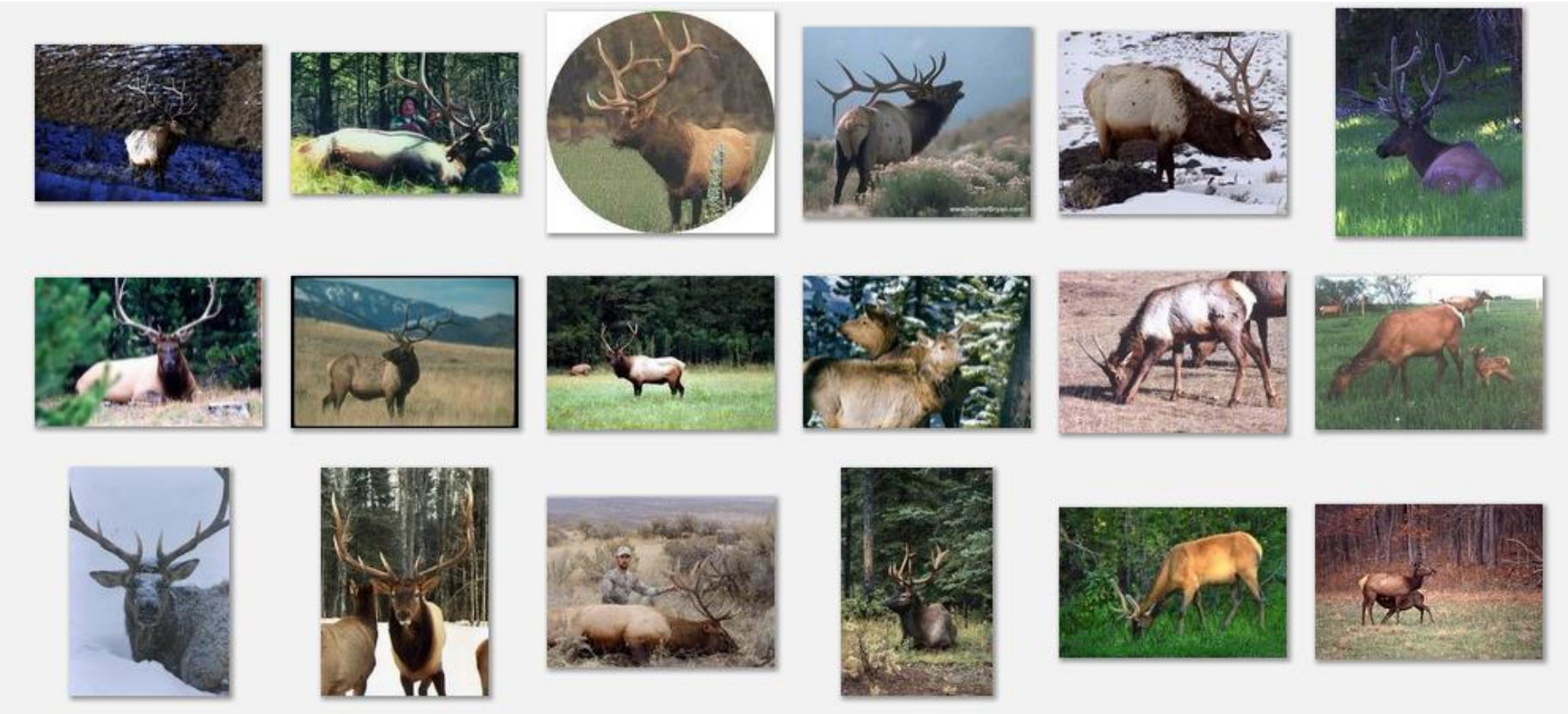
- Object is configuration of parts
- Each part is detectable





Images from Felzenszwalb

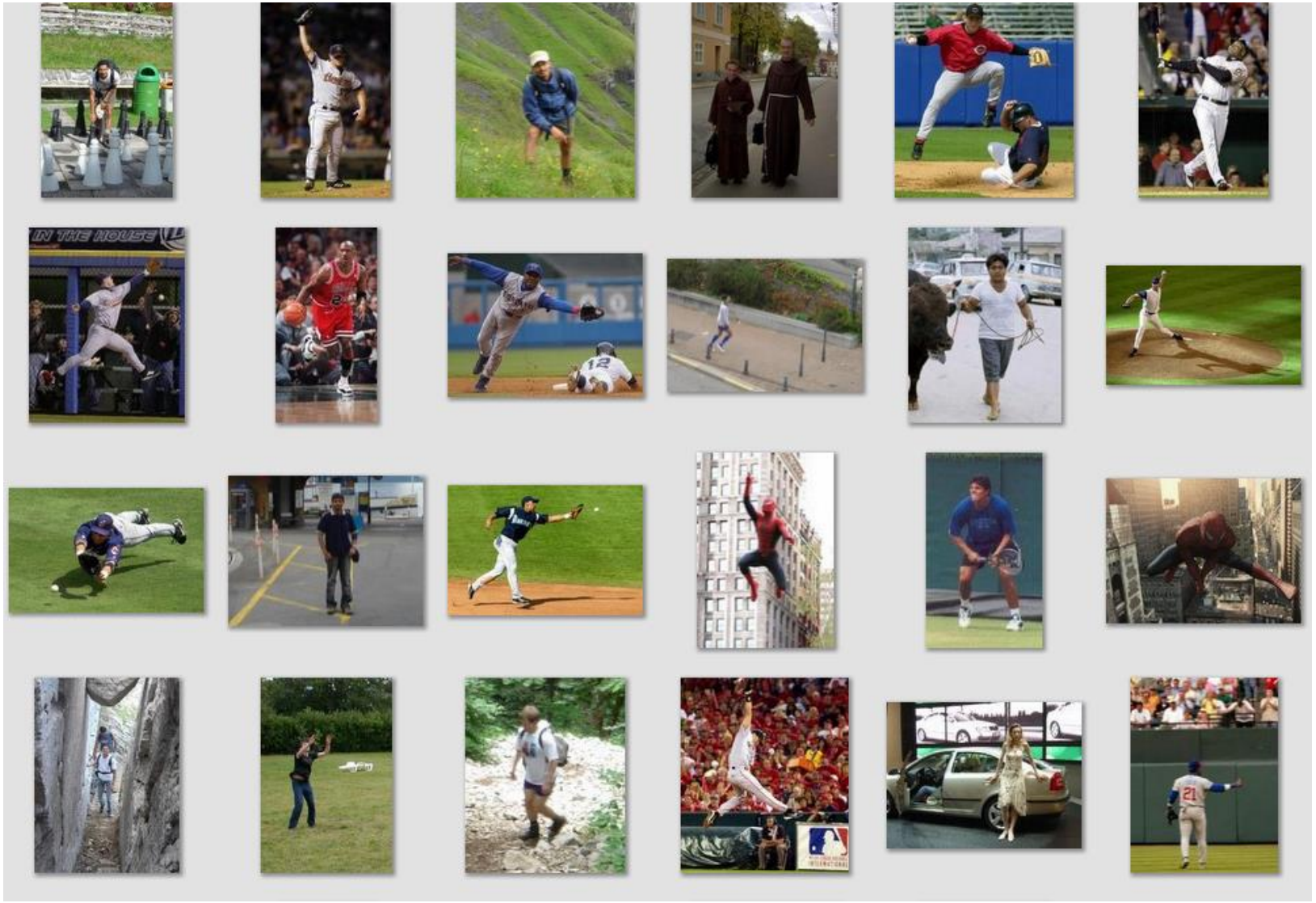
Deformable objects



Images from Caltech-256

Slide Credit: Duan Tran

Deformable objects



Images from D. Ramanan's dataset

Slide Credit: Duan Tran

Compositional objects

Parts-based Models

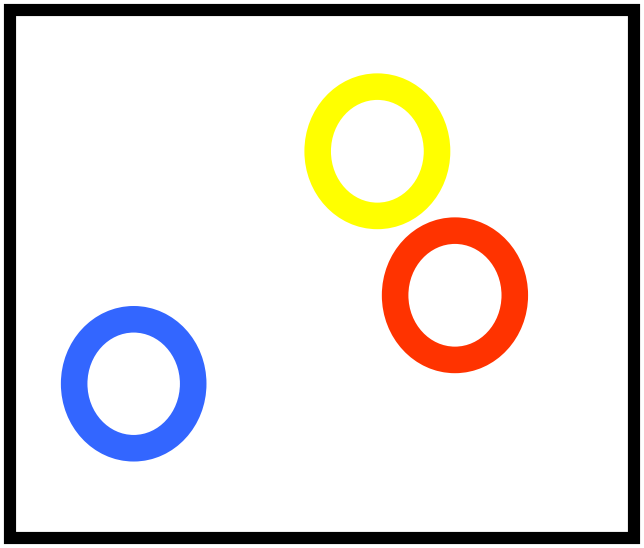
Define object by collection of parts modeled by

1. Appearance
2. Spatial configuration

Slide credit: Rob Fergus

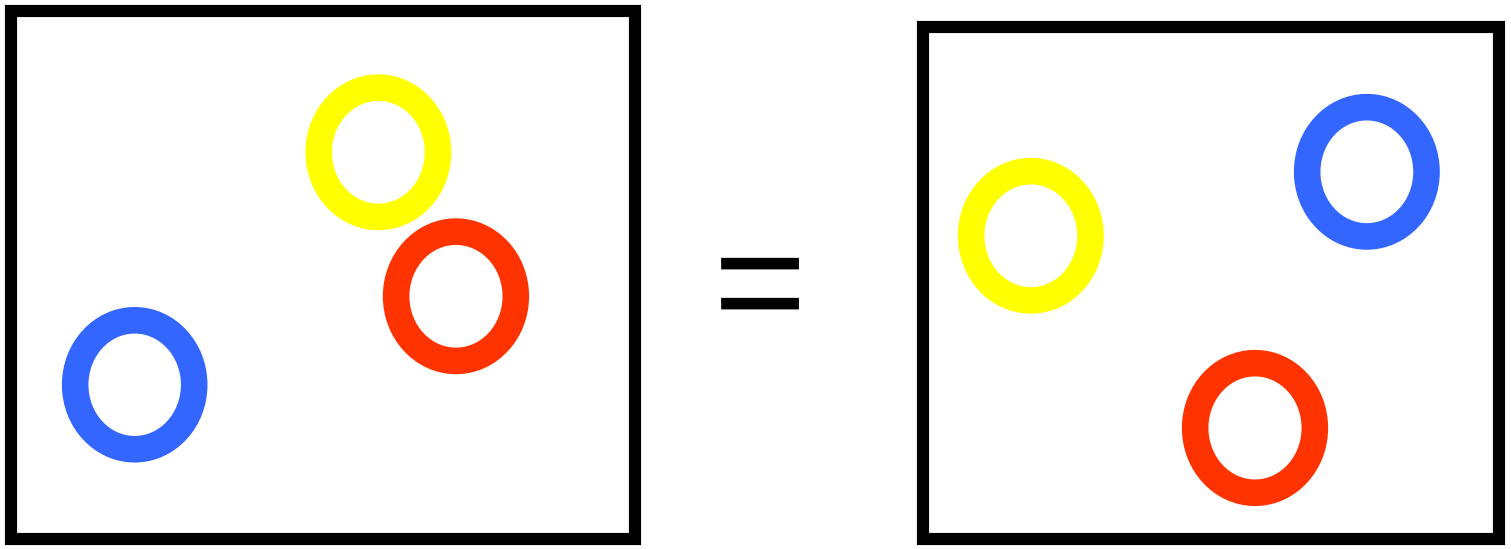
How to model spatial relations?

- One extreme: fixed template



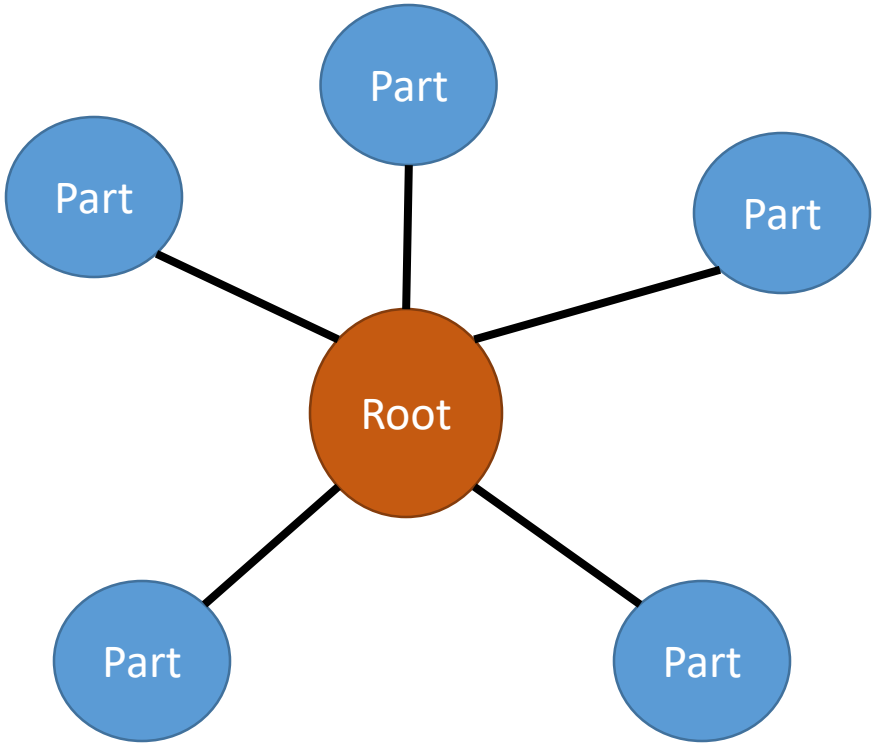
How to model spatial relations?

- Another extreme: bag of words



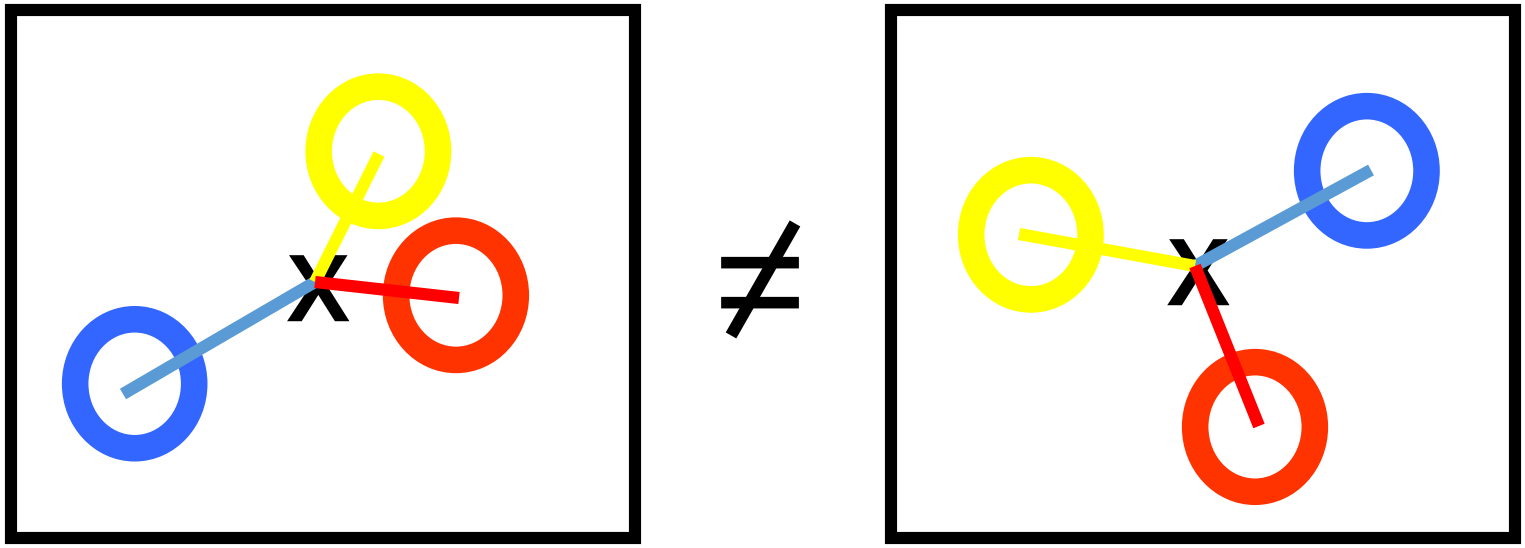
How to model spatial relations?

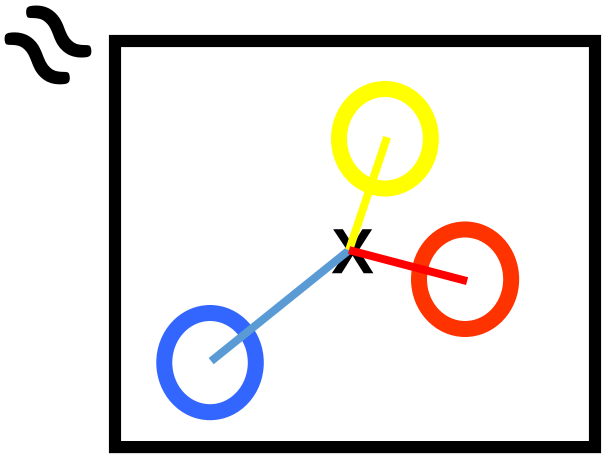
- Star-shaped model

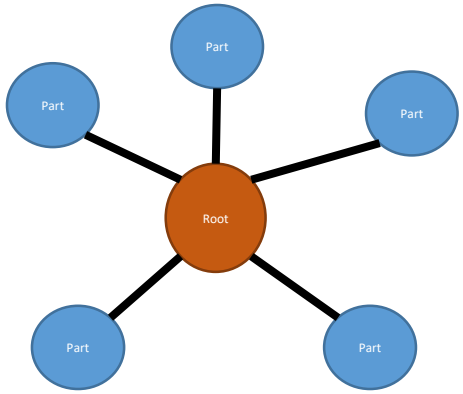


How to model spatial relations?

- Star-shaped model

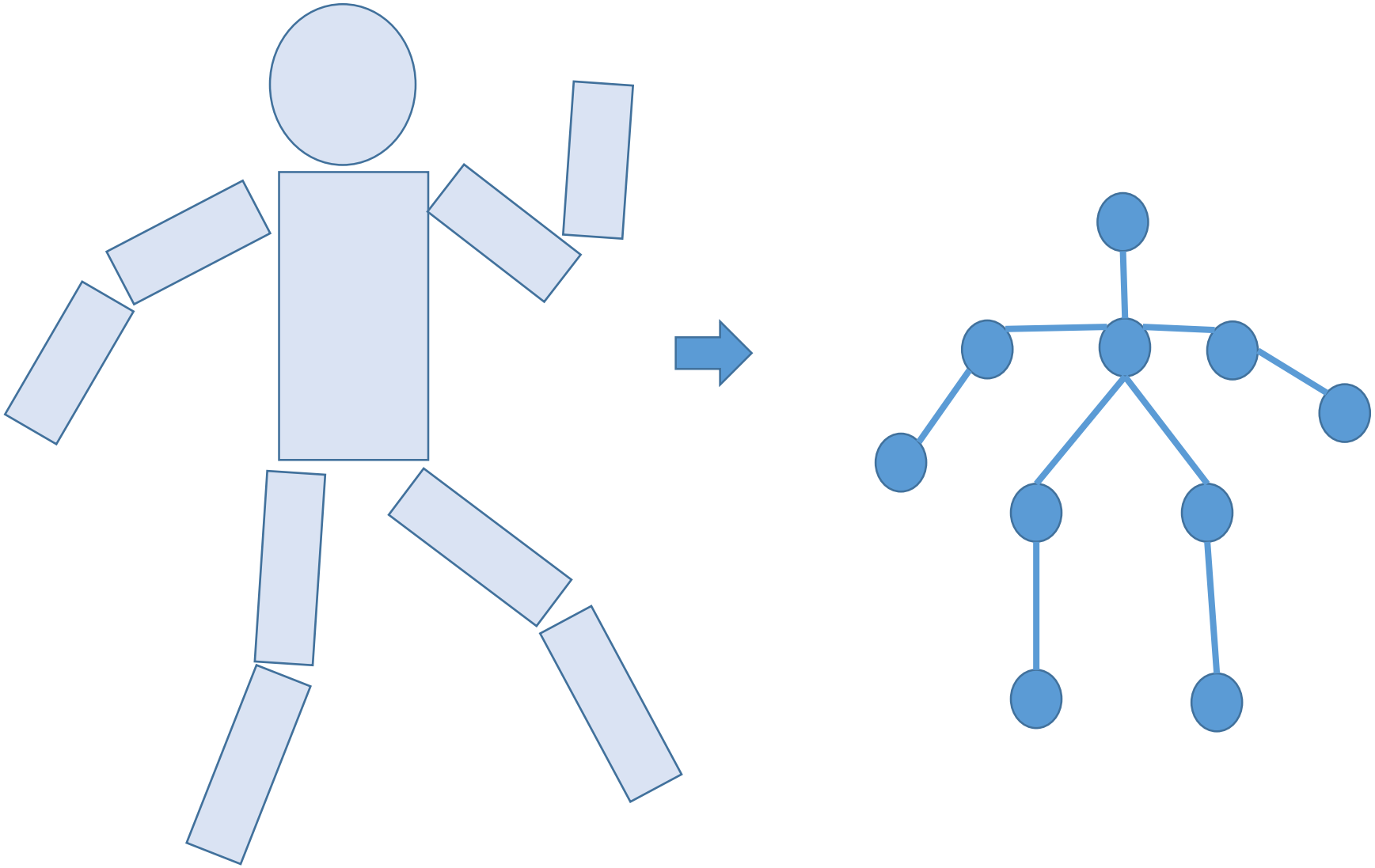






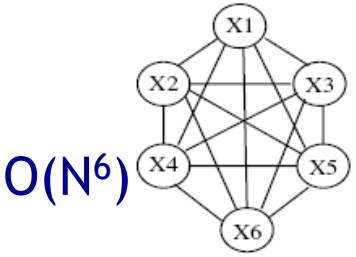
How to model spatial relations?

- Tree-shaped model



How to model spatial relations?

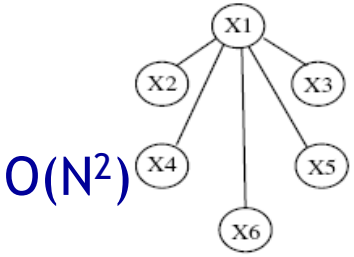
- Many others…



O(N⁶)

a) Constellation

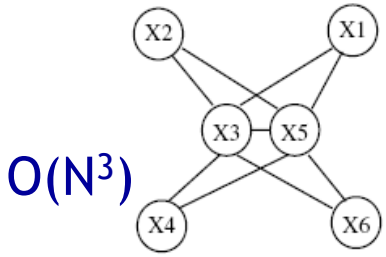
Fergus et al. ’03
Fei-Fei et al. ‘03



O(N²)

b) Star shape

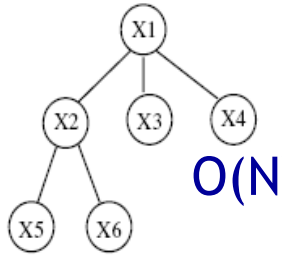
Leibe et al. ’04, ‘08
Crandall et al. ‘05
Fergus et al. ’05



O(N³)

c) k-fan ($k = 2$)

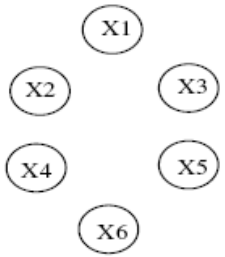
Crandall et al. ‘05



O(N²)

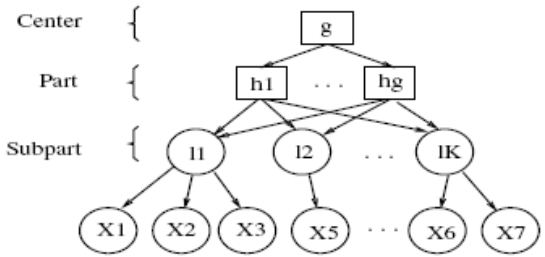
d) Tree

Felzenszwalb & Huttenlocher ‘05



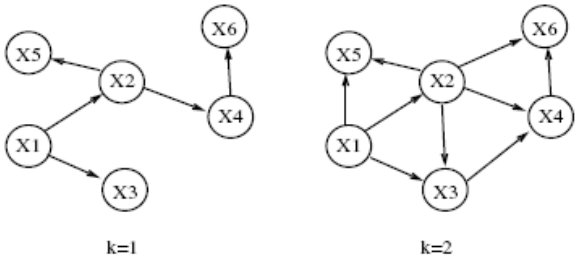
e) Bag of features

Csurka ’04
Vasconcelos ‘00



f) Hierarchy

Bouchard & Triggs ‘05



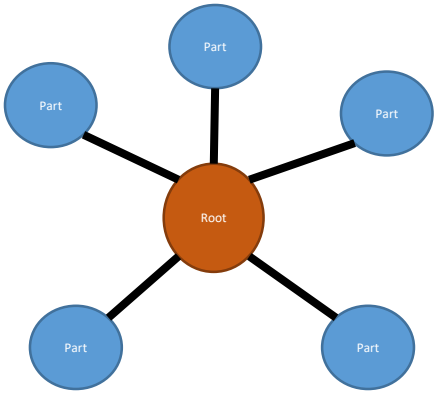
g) Sparse flexible model

Carneiro & Lowe ‘06

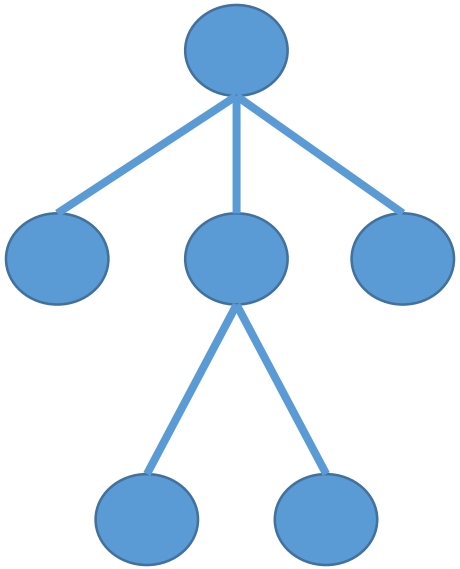
from [Carneiro & Lowe, ECCV’06]

Part-based Models

1. Star-shaped model
   - Example: Deformable Parts Model
     - Felzenswalb et al. 2010



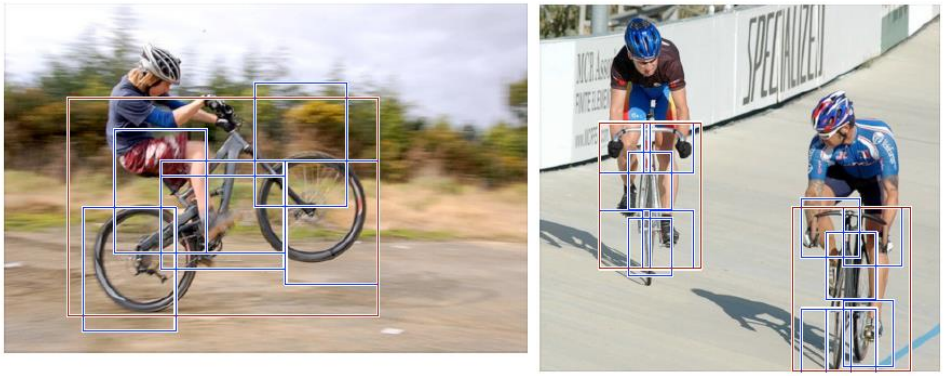
2. Tree-shaped model
   - Example: Pictorial structures
     - Felzenszwalb Huttenlocher 2005



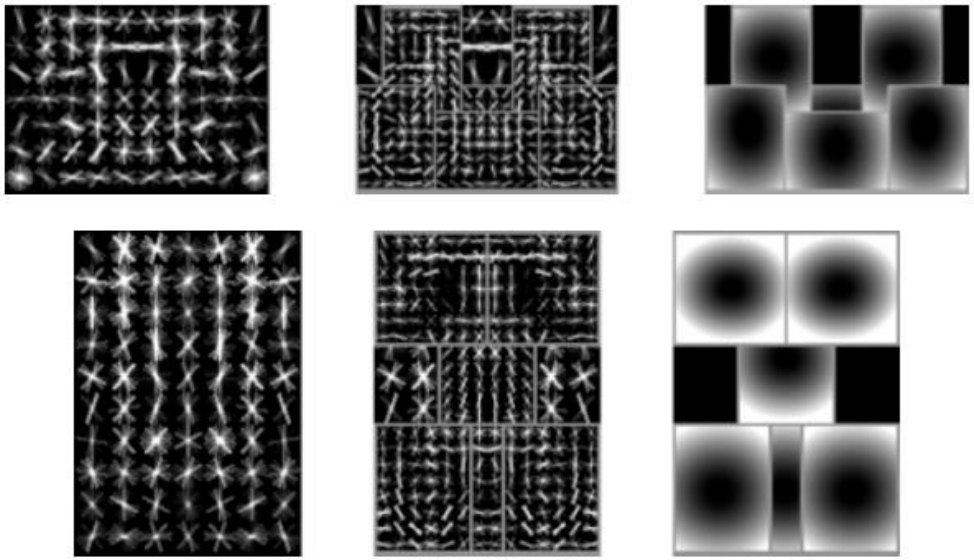
3. Sequential prediction models

Deformable Latent Parts Model (DPM)

Detections



Template Visualization



Felzenszwalb et al. 2008, 2010

Review: Dalal-Triggs detector

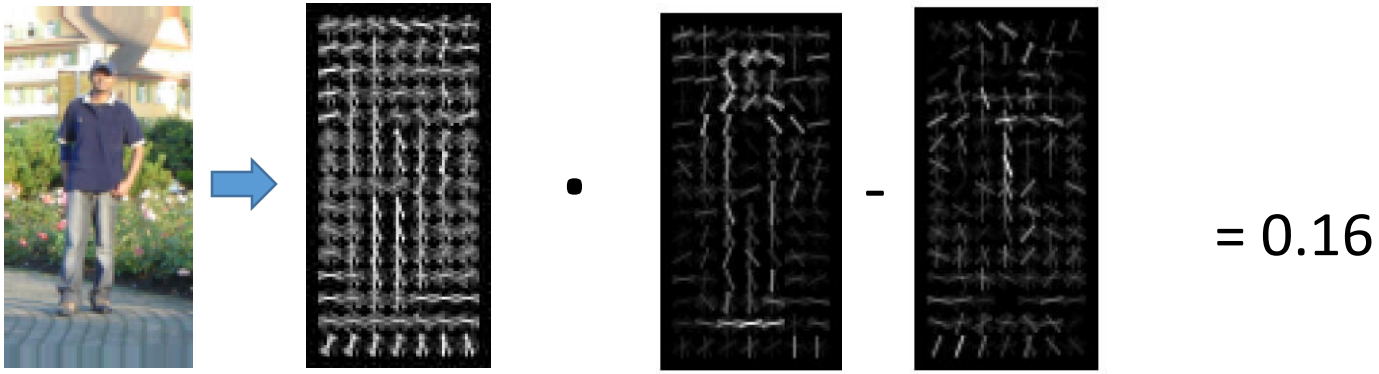
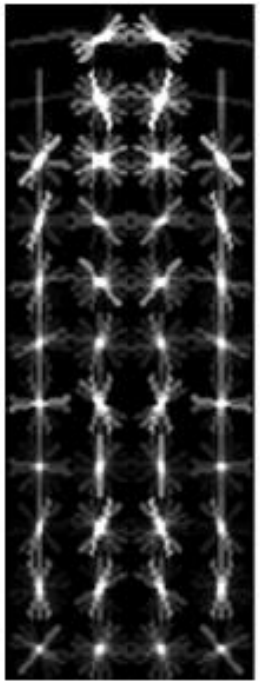


Image Window HOG SVM weights (pos/neg) score

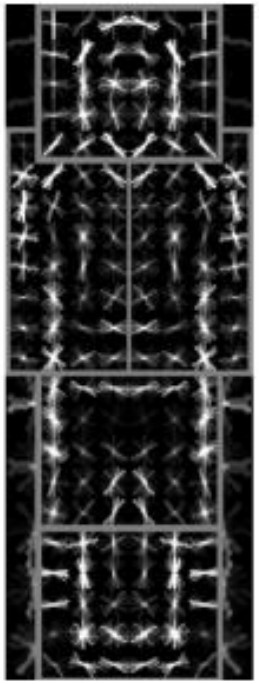
1. Extract fixed-sized (64x128 pixel) window at each position and scale
2. Compute HOG (histogram of gradient) features within each window
3. Score the window with a linear SVM classifier
4. Perform non-maxima suppression to remove overlapping detections with lower scores

Deformable parts model

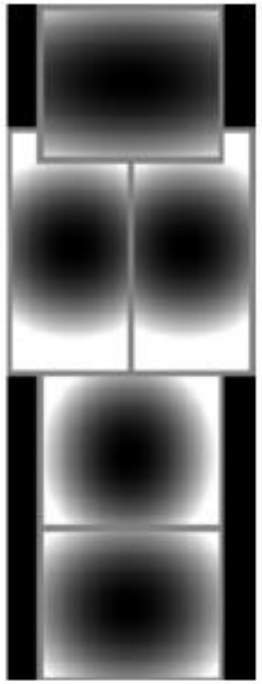
- Root filter models coarse whole-object appearance
- Part filters model finer-scale appearance of smaller patches
- For each root window, part positions that maximize appearance score minus spatial cost are found
- Total score is sum of scores of each filter and spatial costs



Root filter

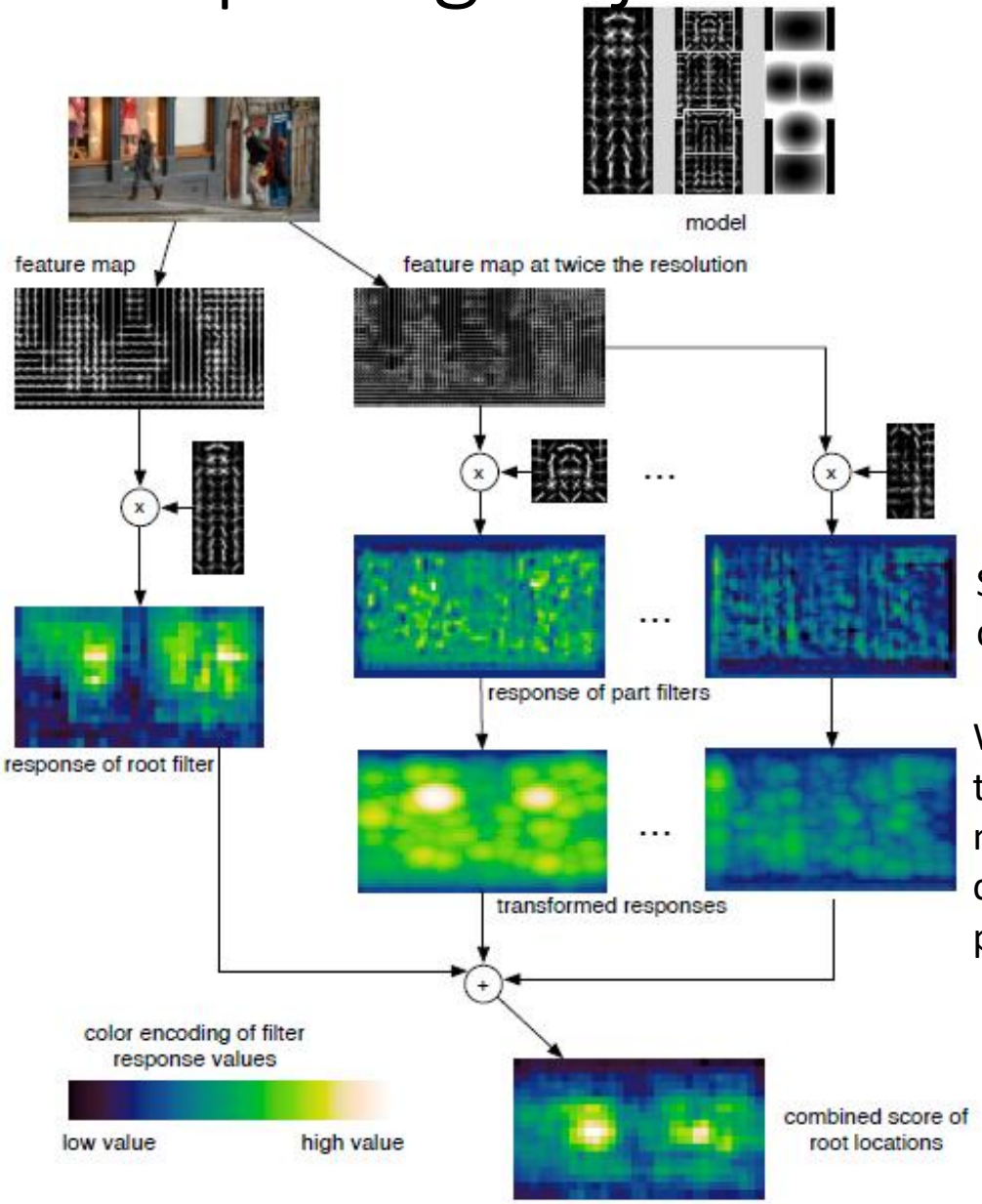


Part filters



Spatial costs

DPM: computing object score



Scores from root detector

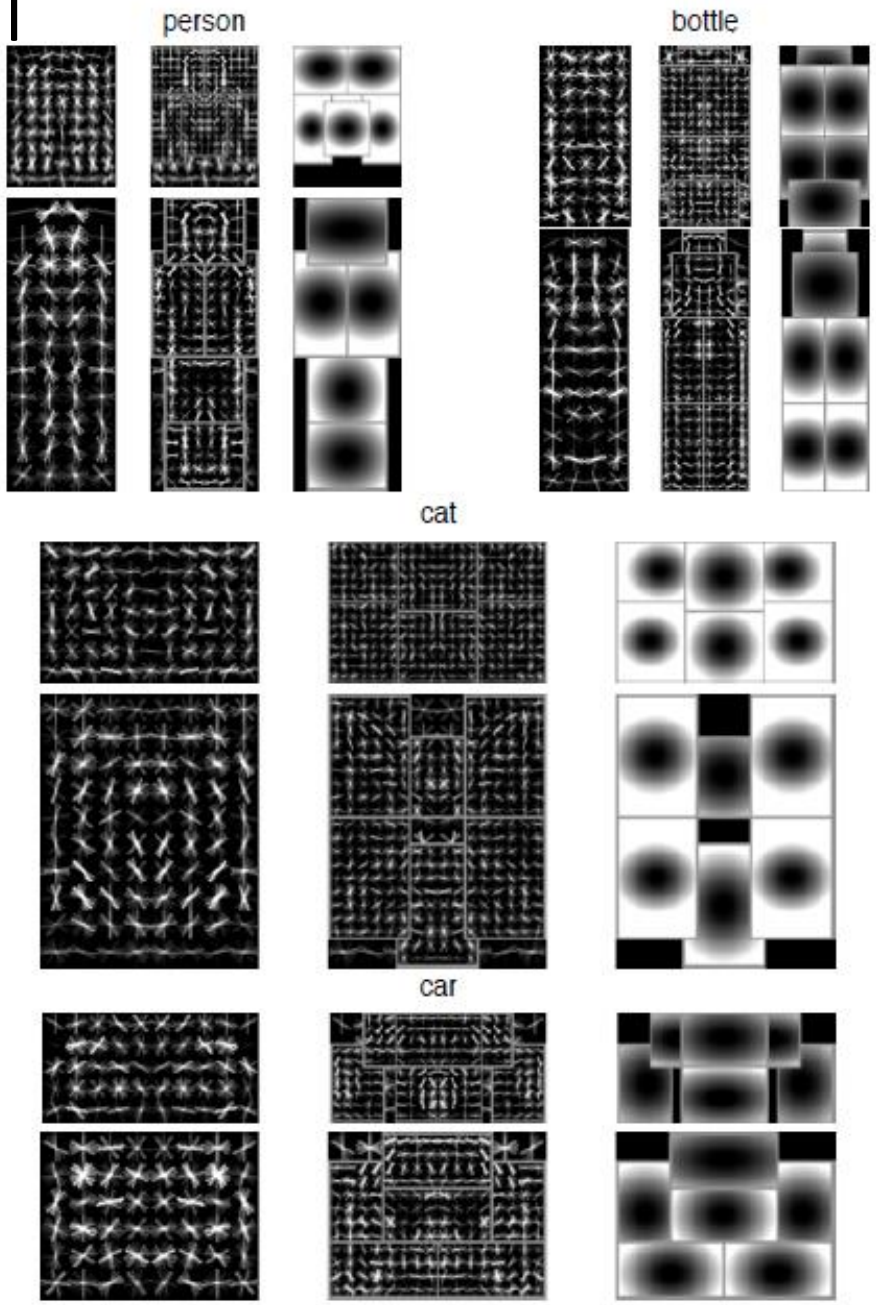
Scores from individual part detectors

With generalized distance transform, compute the maximum part score corresponding to each root position

DPM: mixture model

- Each positive example is modeled by one of M detectors

- In testing, all detectors are applied with non-max suppression



DPM: Training

```
1  F_n := ∅
2  for relabel := 1 to num-relabel do
3  |   F_p := ∅
4  |   for i := 1 to n do
5  |   |   Add detect-best(β, I_i, B_i) to F_p
6  |   end
7  |   for datamine := 1 to num-datamine do
8  |   |   for j := 1 to m do
9  |   |   |   if |F_n| ≥ memory-limit then break
10 |   |   |   Add detect-all(β, J_j, −(1 + δ)) to F_n
11 |   |   end
12 |   |   β := gradient-descent(F_p ∪ F_n)
13 |   |   Remove (i, v) with β · v < −(1 + δ) from F_n
14 |   end
15 end
```

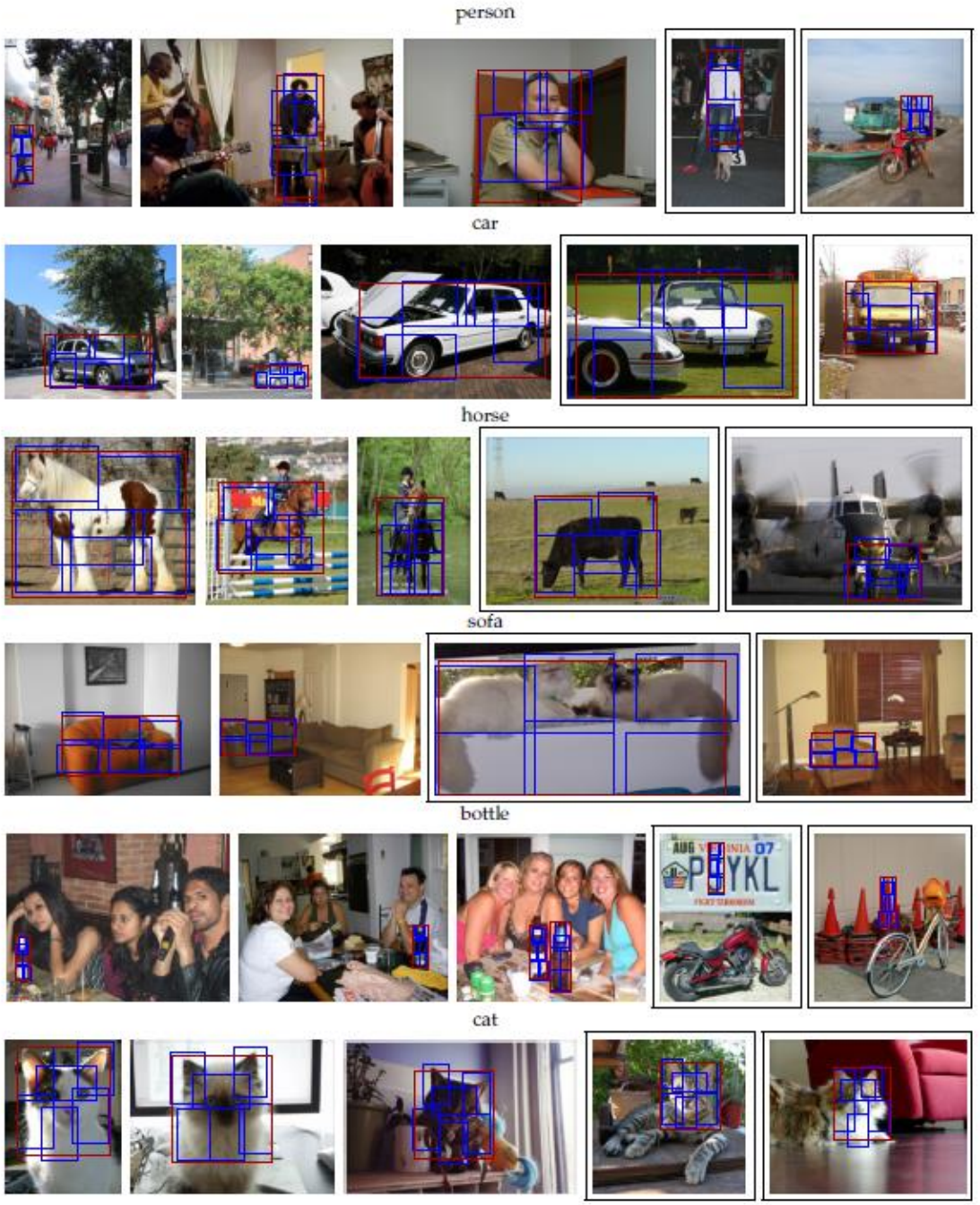
Procedure `Train`

Solve for latent parameters (root/part positions, mixture component) that maximize score and are consistent with ground truth bounding box

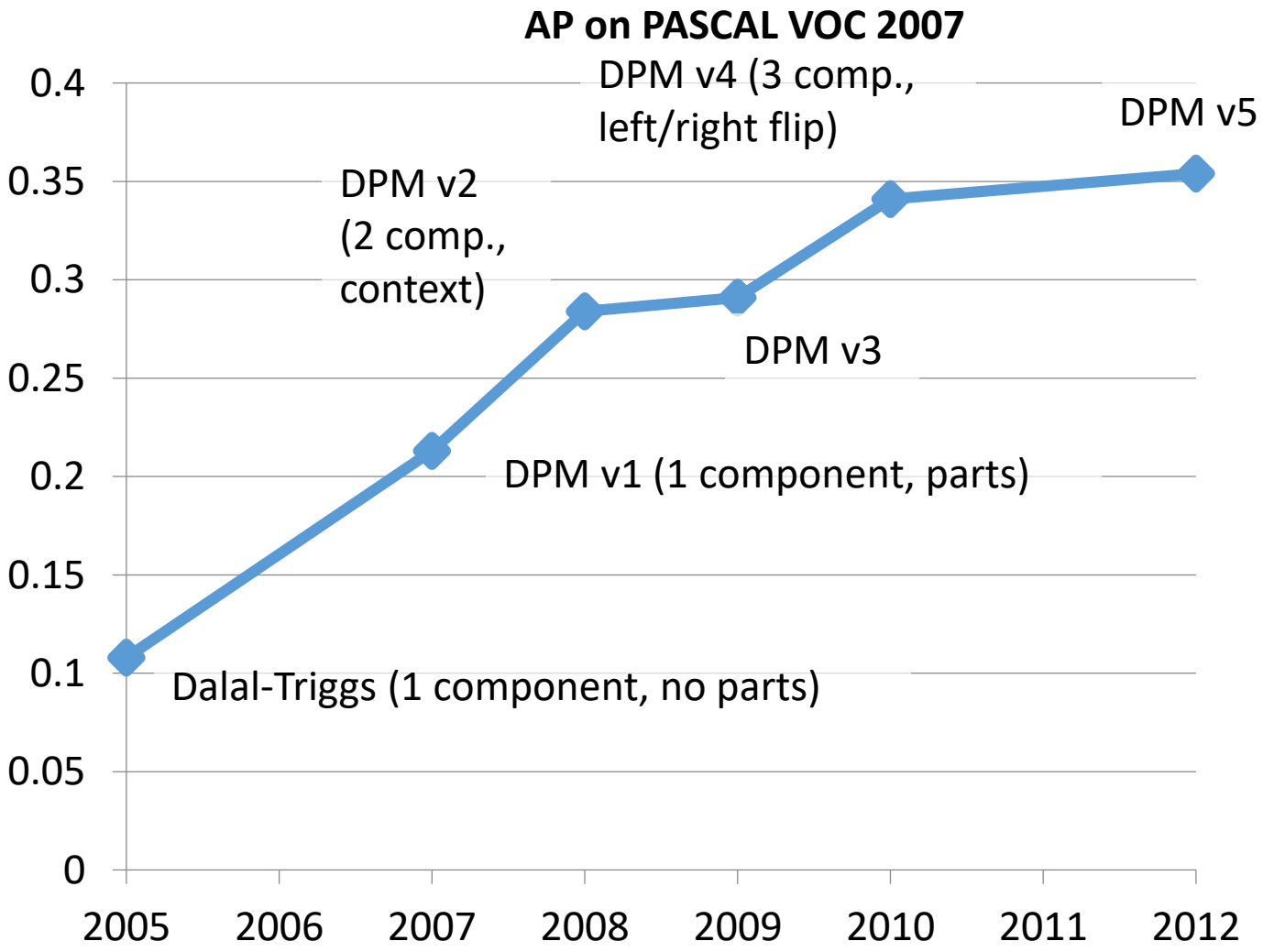
Add negative examples that achieve some minimum score (> 1 – delta)

Solve for SVM weights given current latent parameters and negative examples

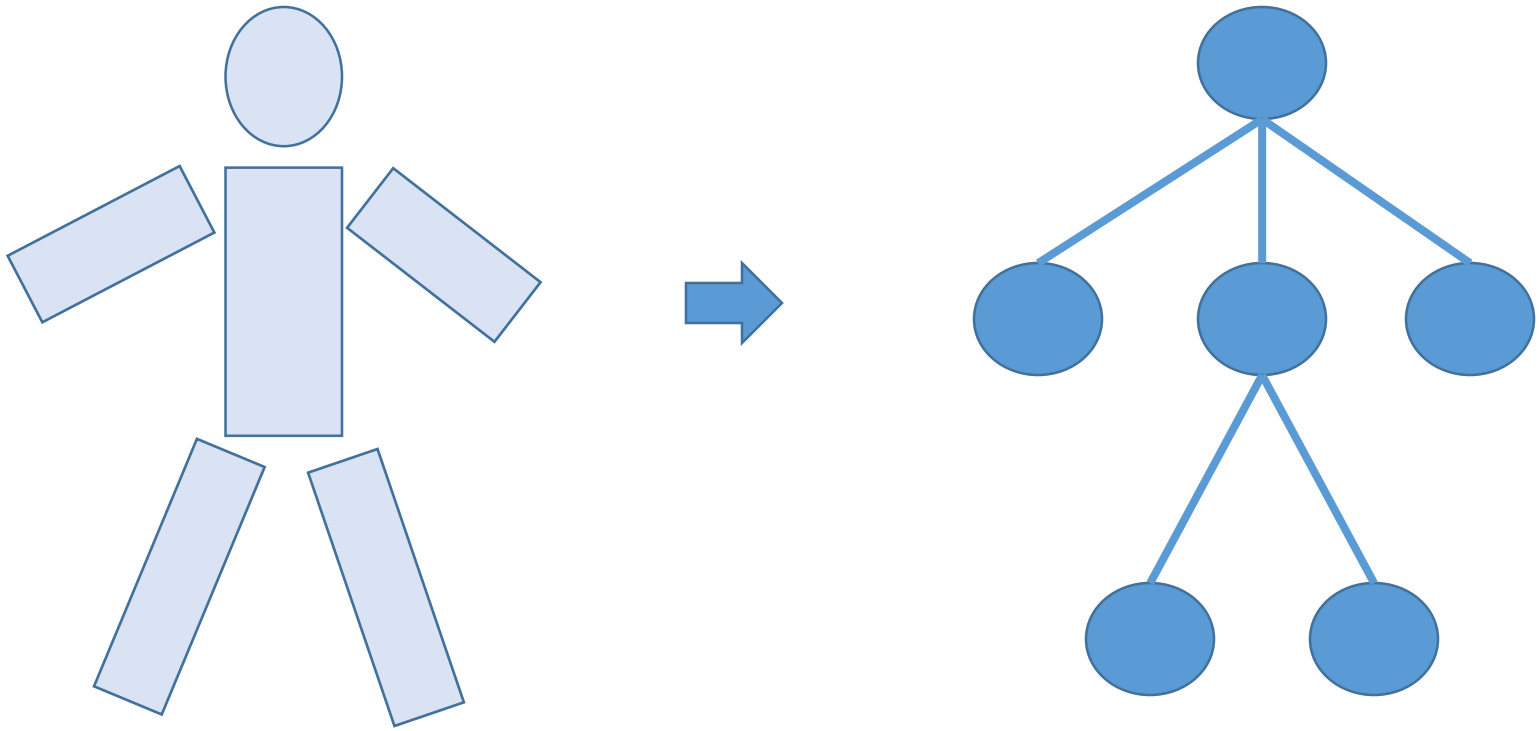
Results



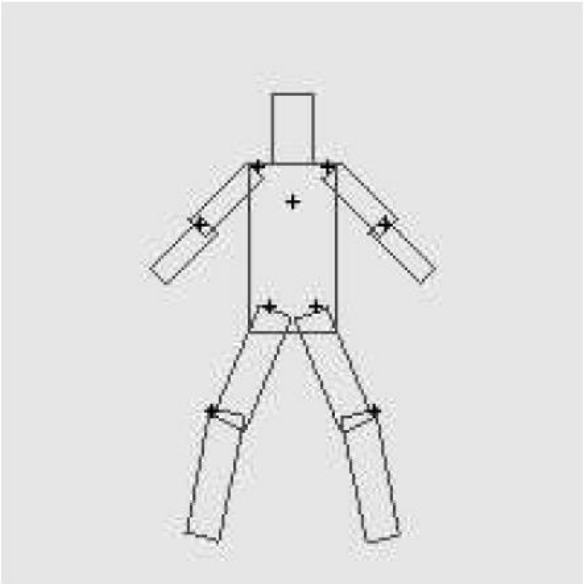
Improvement over time for HOG-based detectors

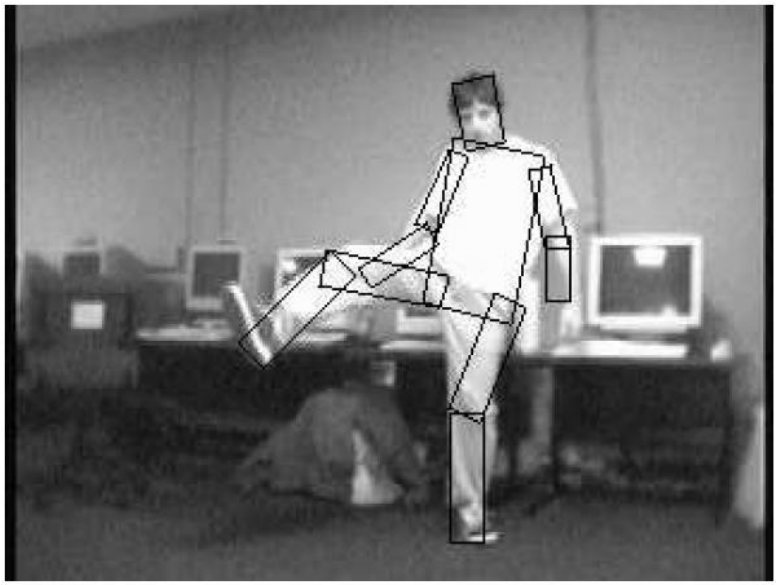


Tree-shaped model



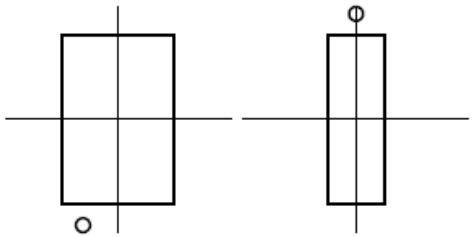
Pictorial Structures Model



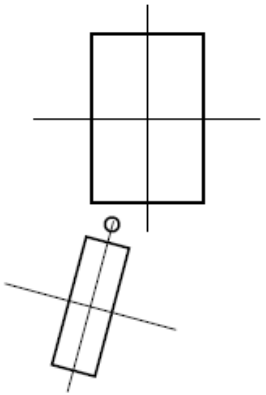


Part = oriented rectangle

Spatial model = relative size/orientation



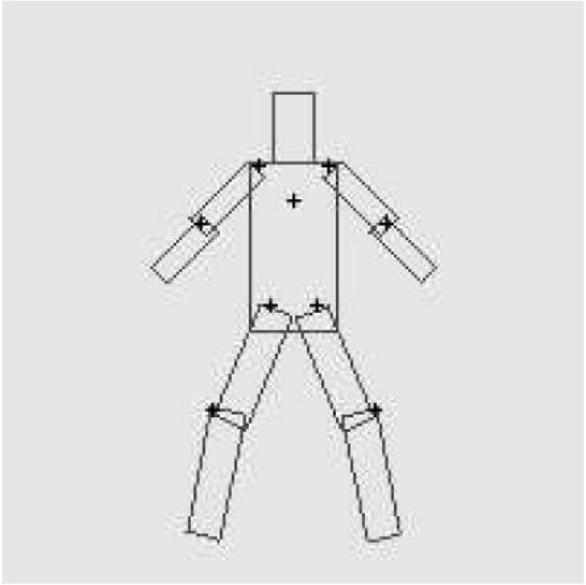
a



b

Felzenszwalb and Huttenlocher 2005

Pictorial Structures Model



$$P(L|I,\theta) \propto \left(\prod_{i=1}^{n} p(I|l_i, u_i) \prod_{(v_i,v_j)\in E} p(l_i, l_j|c_{ij}) \right)$$

Appearance likelihood

Geometry likelihood

Modeling the Appearance

- Any appearance model could be used
  - HOG Templates, etc.
  - Here: rectangles fit to background subtracted binary map
- Can train appearance models independently (easy, not as good) or jointly (more complicated but better)

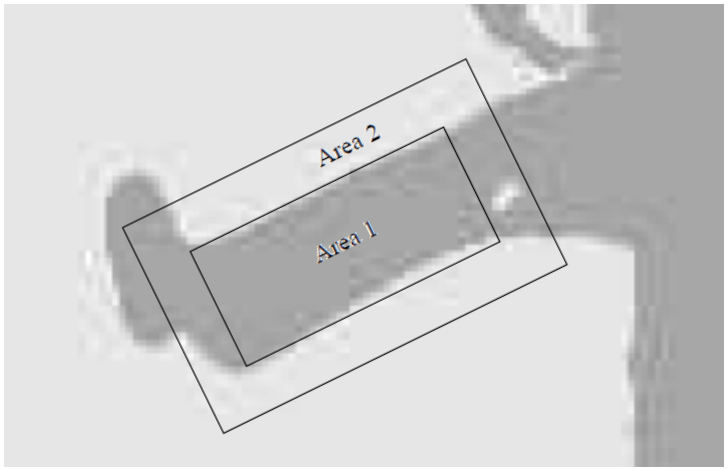
$$P(L|I,\theta) \propto \left(\prod_{i=1}^{n} p(I|l_i, u_i) \prod_{(v_i,v_j)\in E} p(l_i, l_j|c_{ij}) \right)$$

Appearance likelihood

Geometry likelihood

Part representation

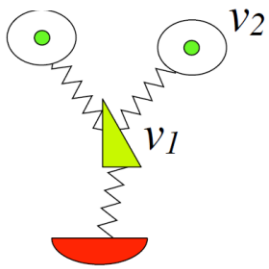
- Background subtraction



Pictorial structures model

Optimization is tricky but can be efficient

$$L^* = \arg\min_L \left(\sum_{i=1}^{n} m_i(l_i) + \sum_{(v_i,v_j)\in E} d_{ij}(l_i, l_j) \right)$$



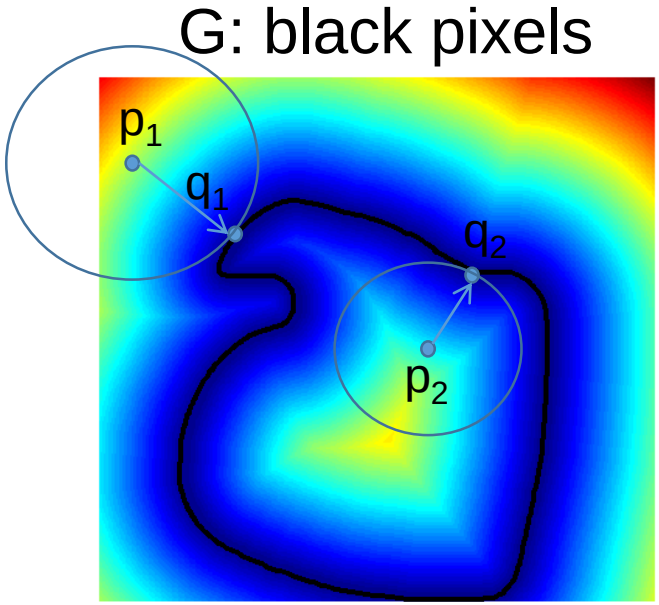
- For each l_1, find best l_2:

$$\text{Best}_2(l_1) = \min_{l_2} \left[m_2(l_2) + d_{12}(l_1, l_2) \right]$$

- Remove v_2, and repeat with smaller tree, until only a single part
- For k parts, n locations per part, this has complexity of $O(kn^2)$, but can be solved in ~$O(kn)$ using generalized distance transform

Distance Transform

- For each pixel p, how far away is the nearest pixel q of set G
  - $f(p) = \min_{q \in G} \; d(p, q)$
  - G is often the set of edge pixels



Distance Transform - Applications

- Set distances – e.g. Hausdorff Distance
- Image processing – e.g. Blurring
- Robotics – Motion Planning
- Alignment
  - Edge images
  - Motion tracks
  - Audio warping
- *Deformable Part Models*

Generalized Distance Transform

- Original form: $f(p) = \min_{q \in G} \; d(p, q)$

- General form: $f(p) = \min_{q \in [1,N]} \; m(q) + d(p, q)$

- For many deformation costs, $O(N^2) \rightarrow O(N)$

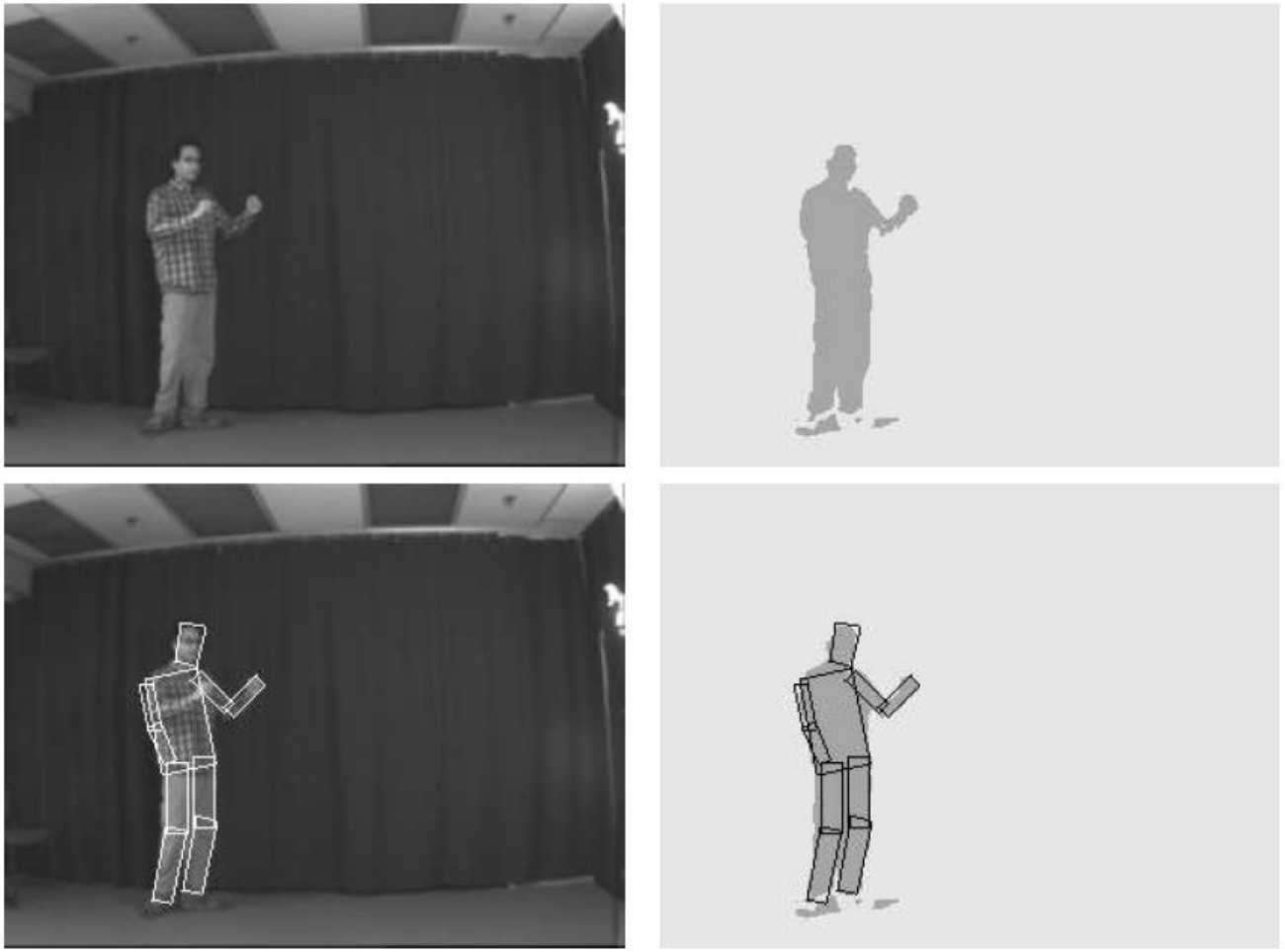
Quadratic $d(p, q) = \alpha(p - q)^2 + \beta(p - q)$

Abs Diff $d(p, q) = \alpha|p - q|$

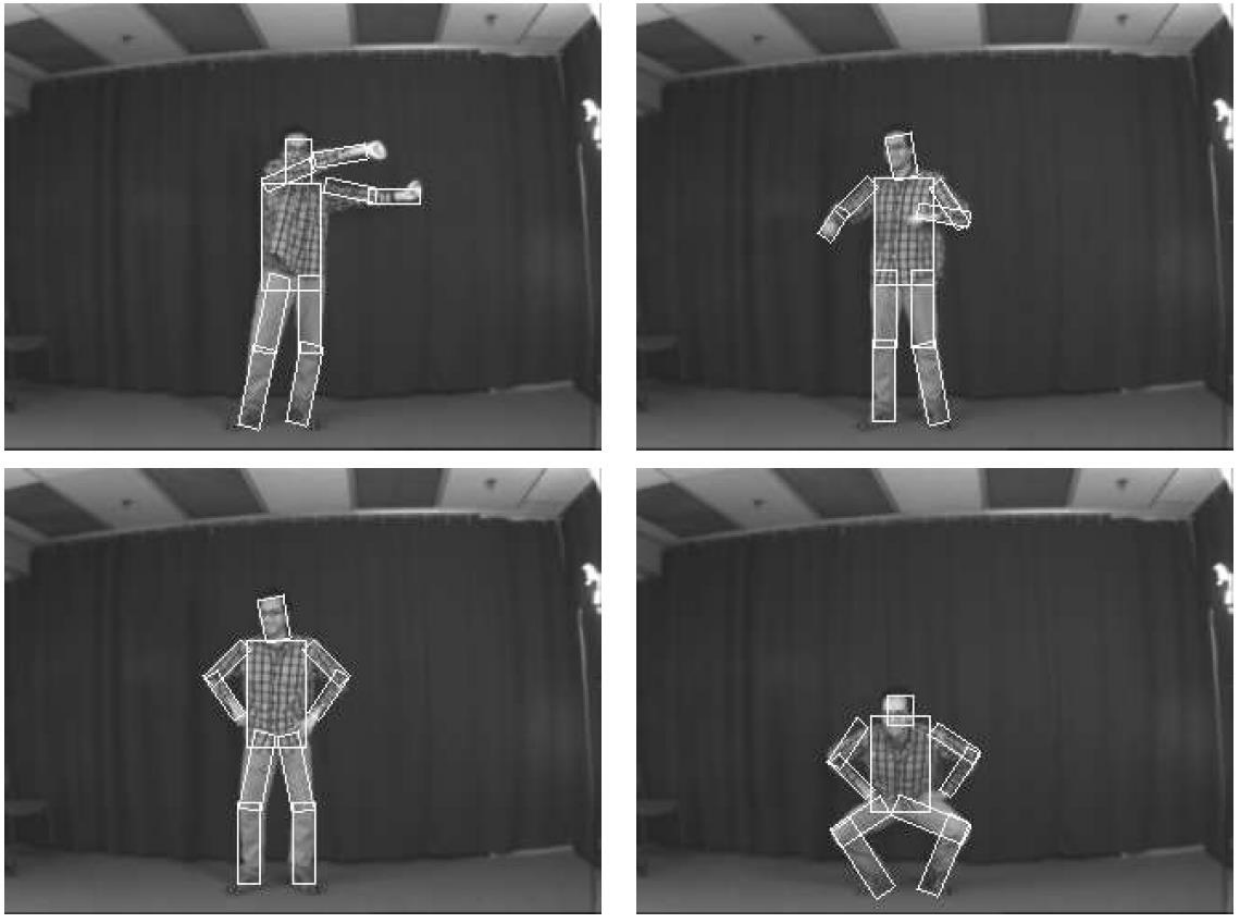
Min Composition $d(p, q) = \min(d_1(p, q), d_2(p, q))$

Bounded $d_\tau(p, q) = \begin{cases} d(p, q) & : |p - q| < \tau \\ \infty & : |p - q| \geq \tau \end{cases}$

Results for person matching



Results for person matching



Enhanced pictorial structures

- Learn spatial prior
- Color models from soft segmentation (initialized by location priors of each part)

BMVC 2009

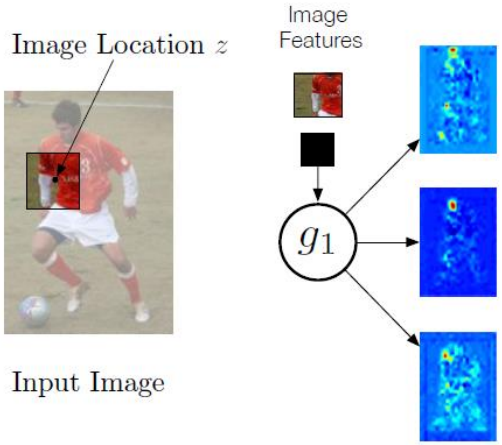
Which patch corresponds to a body part?

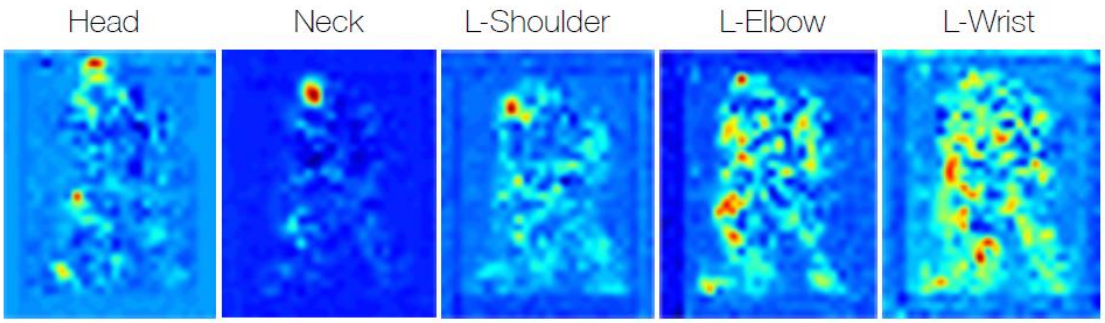
Example from Ramakrishna

Sequential structured prediction

- Can consider pose estimation as predicting a set of related variables (called structured prediction)
  - Some parts easy to find (head), some are hard (wrists)
- One solution: jointly solve for most likely variables (DPM, pictorial structures)
- Another solution: iteratively predict each variable based in part on previous predictions

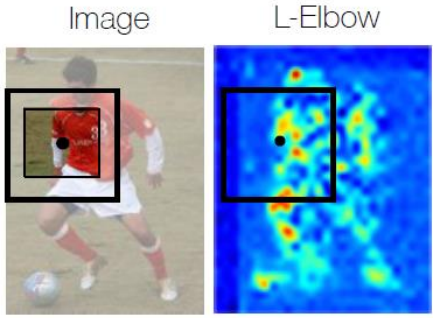
Pose machines

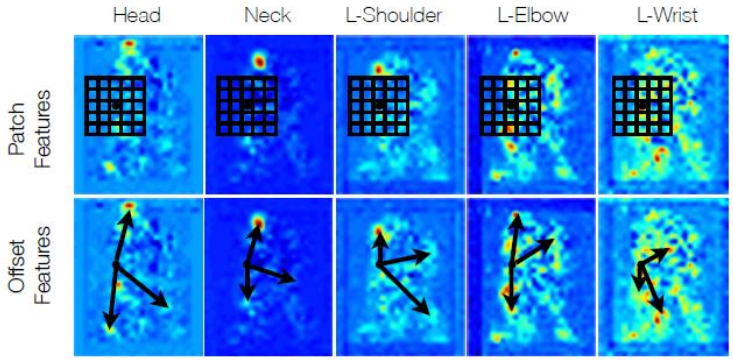




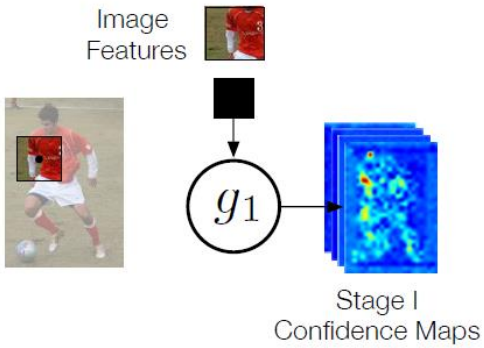
Local image evidence is weak

Certain parts are easier to detect than others

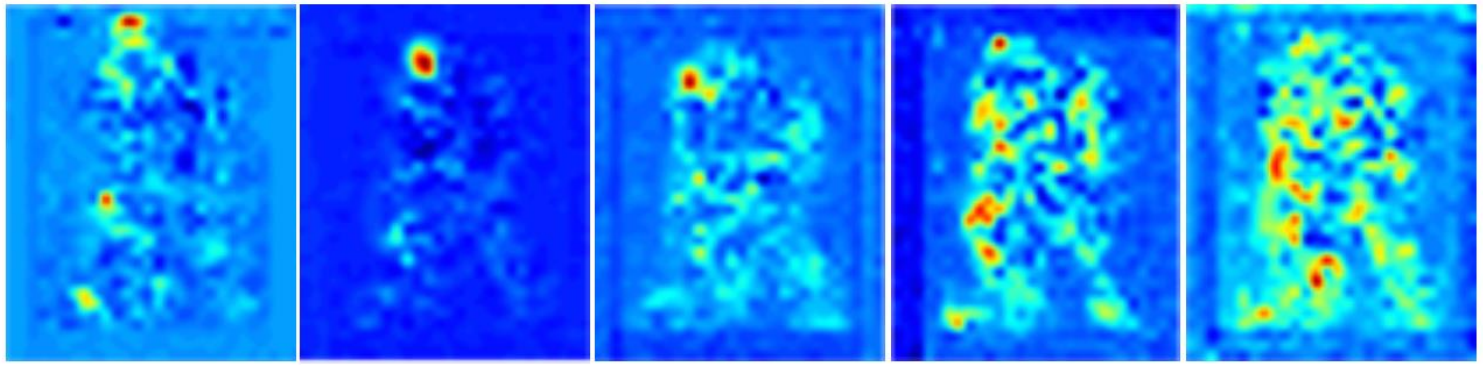




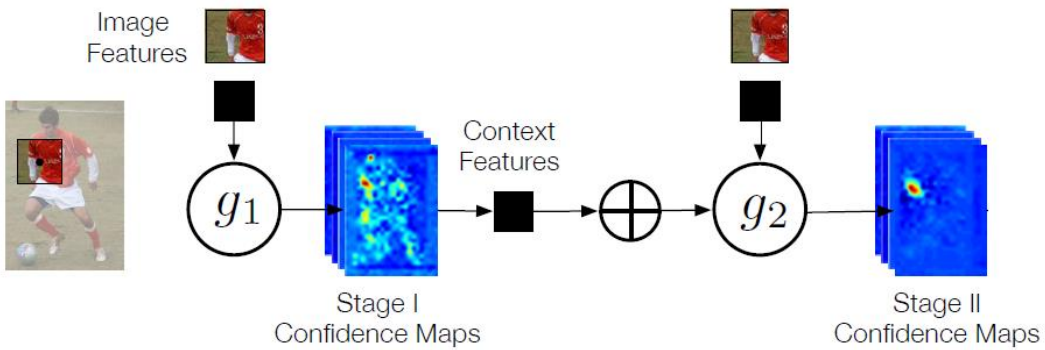
Ramakrishna et al. ECCV 2014



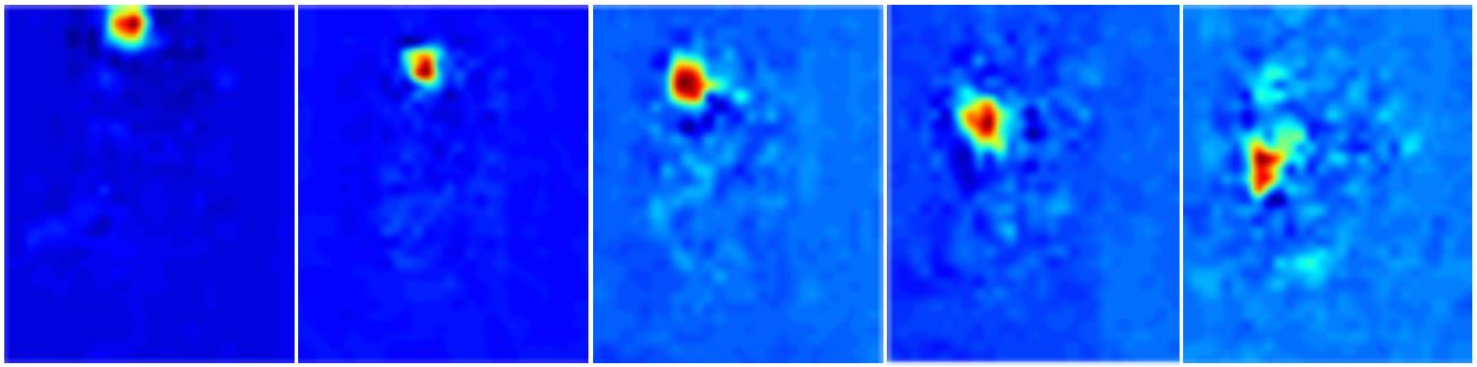
Stage I Confidence



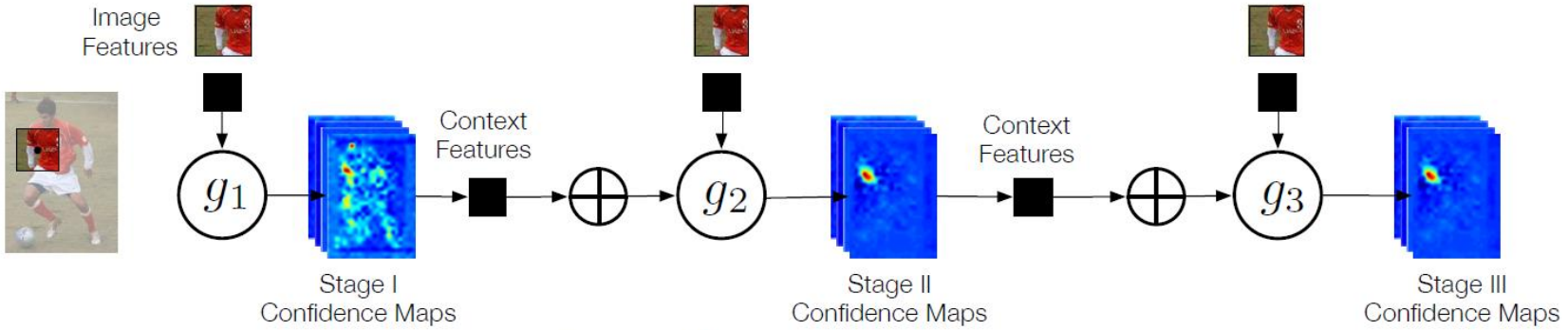
Head | Neck | L-Shoulder | L-Elbow | L-Wrist



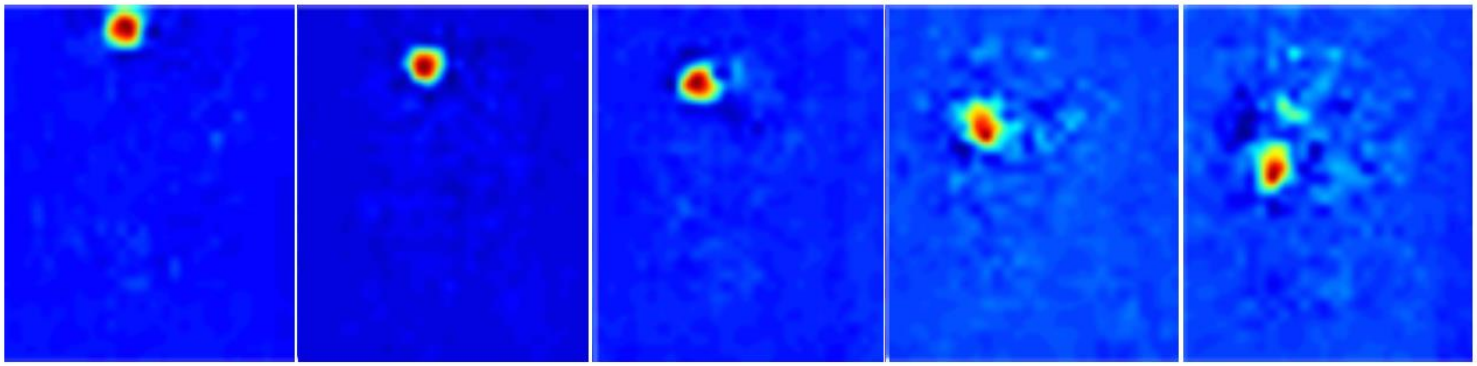
Stage II Confidence



Head Neck L-Shoulder L-Elbow L-Wrist



Stage III Confidence



Example results

General principle

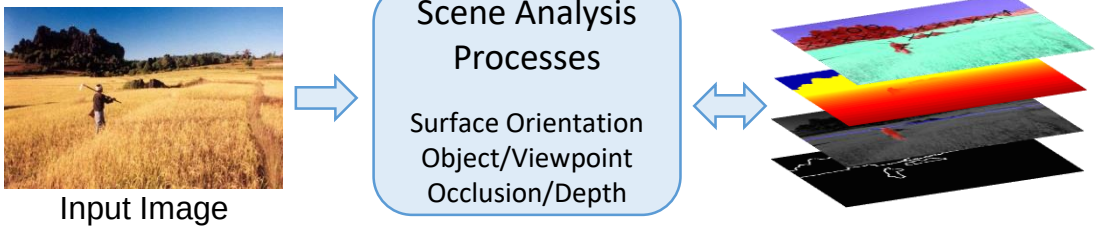
- "Auto-context" (Tu CVPR 2008): instead of fancy graphical models, create feature from past predictions and repredict

- Can view this as an "unrolled belief propagation" (Ross et al. 2011)

Tu Bai 2010: Auto-context
Ross Munoz Hebert Bagnell 2011: Message-Passing Inference Machines

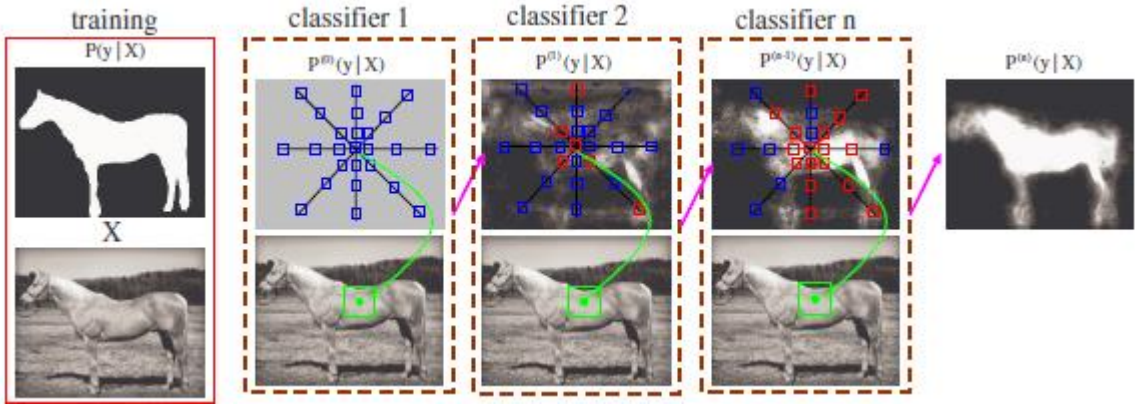
Many uses and variations on sequential structured prediction

Closing the Loop



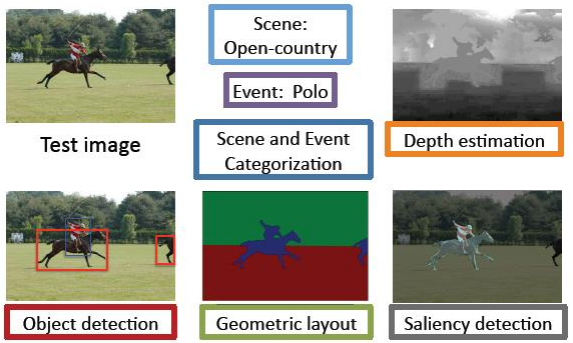
Hoiem Efros Hebert 2008

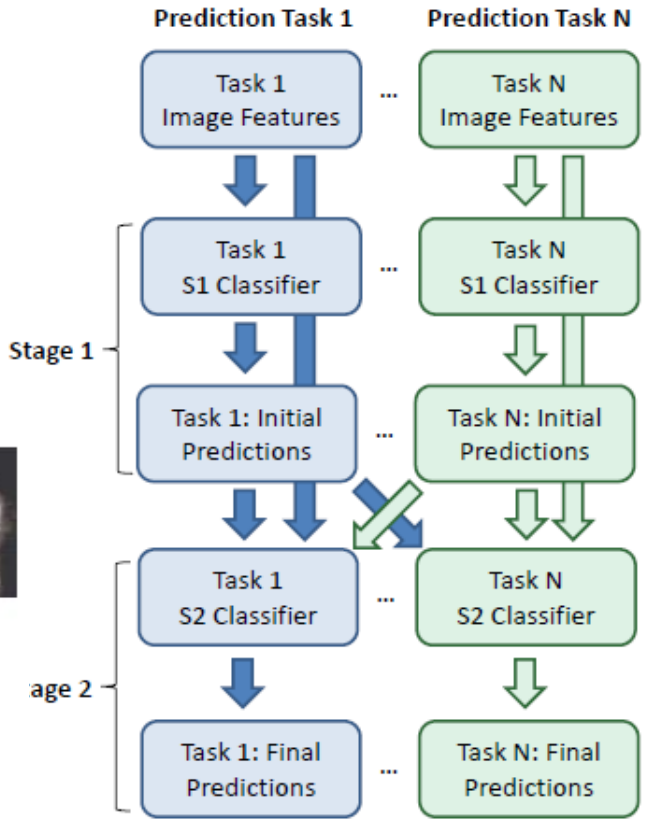
Autocontext



Tu 2008
Tu Bai 2010

Cascaded Classification Model

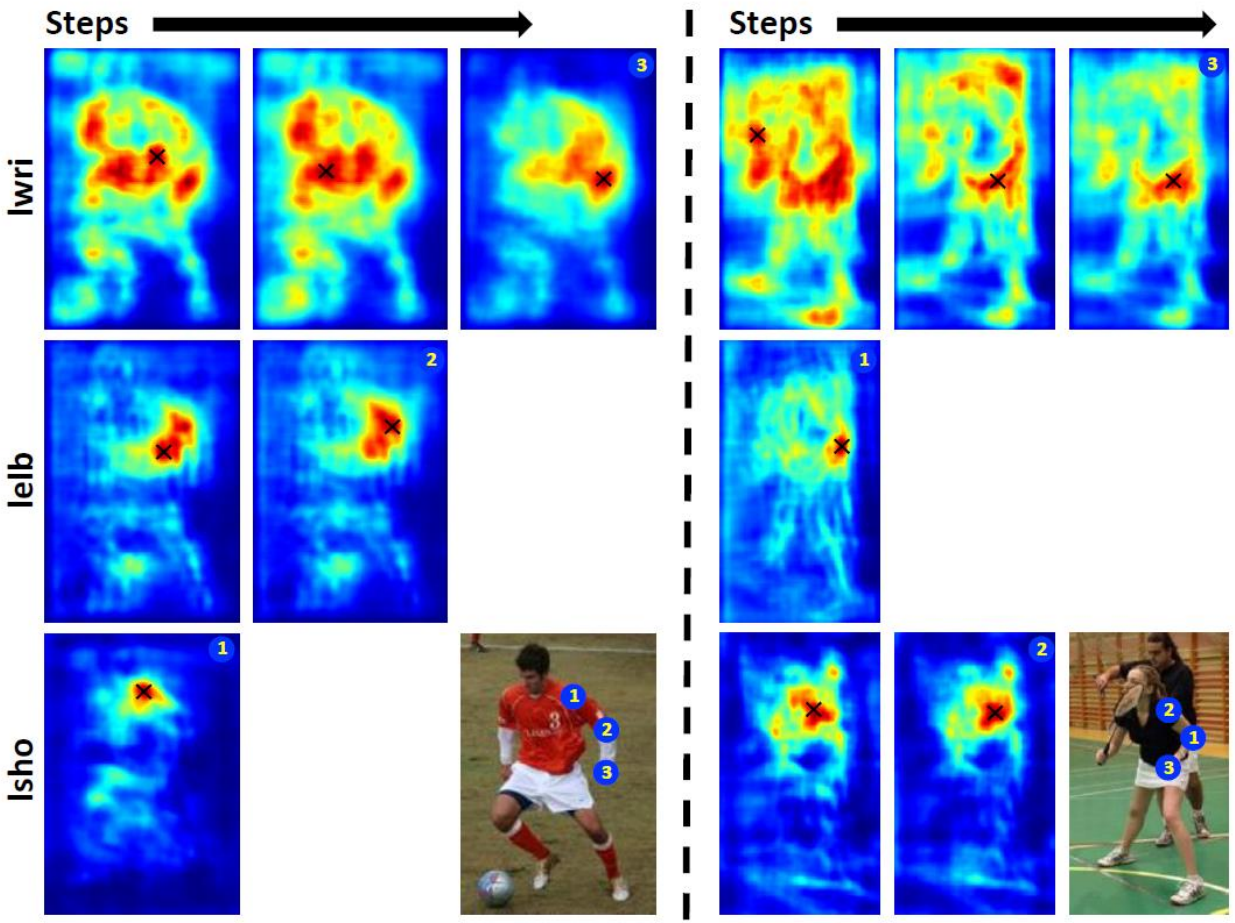




Heitz Gould Saxena Koller 2008
Li Kowdle Saxena Chen 2010

Learning to search for landmarks

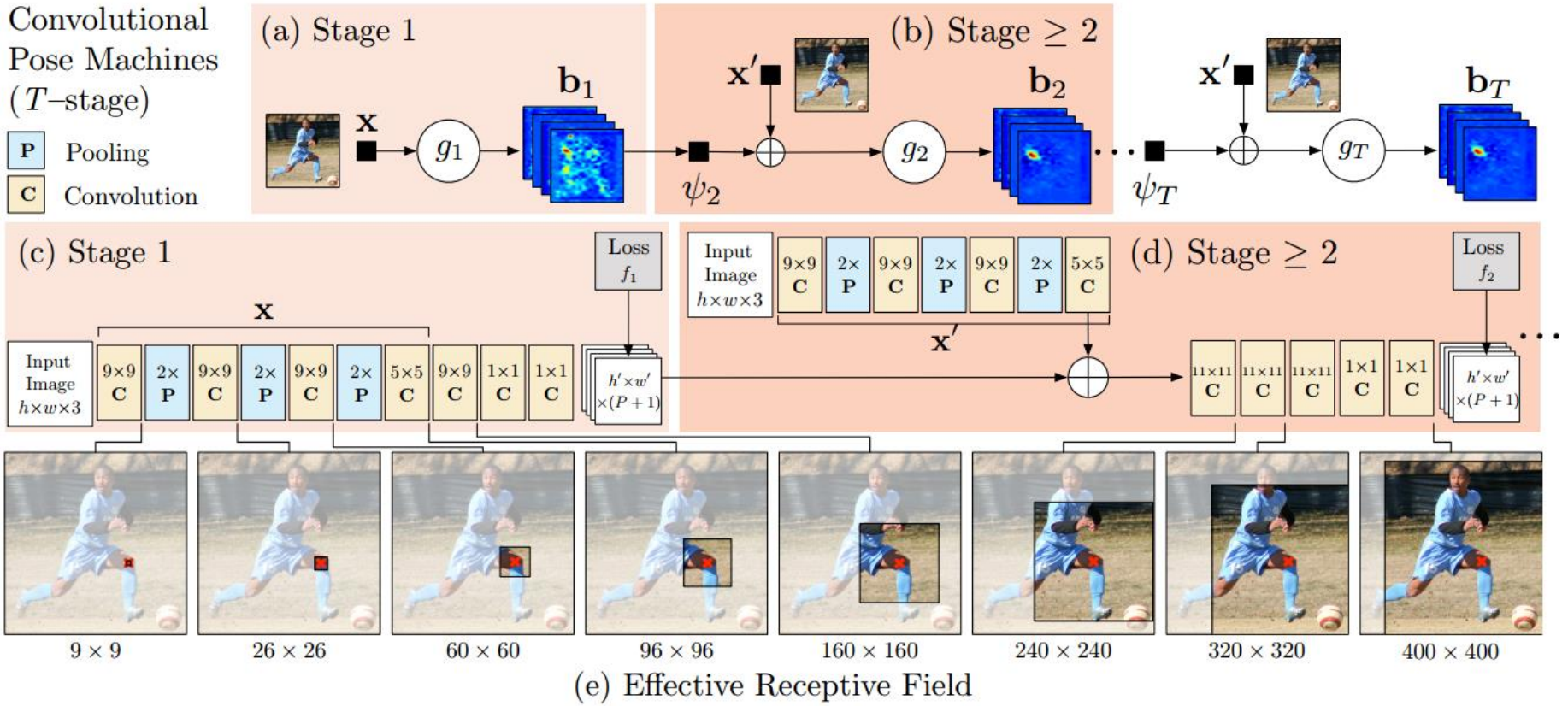
- Learn to find easy landmarks (body joints) first and s



Singh et al. CVPR 2015

Results: best (top) to worst (bottom)

Convolutional Pose Machine



CVPR 2016 https://arxiv.org/pdf/1602.00134.pdf

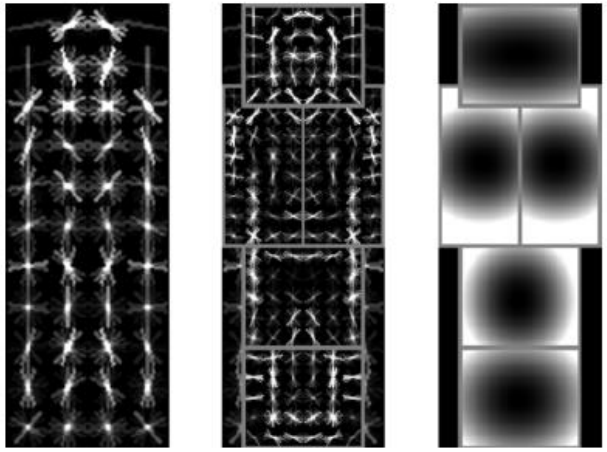
Graphical models vs. structured prediction

- Advantages of sequential prediction
  - Simple procedures for training and inference
  - Learns how much to rely on each prediction
  - Can model very complex relations

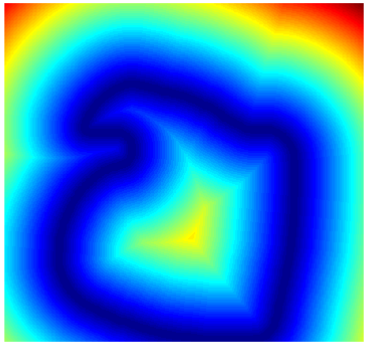
- Advantages of BP/graphcut/etc
  - Elegant
  - Relations are explicitly modeled
  - Exact inference in some cases

Things to remember

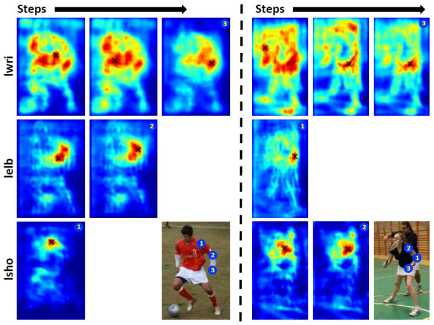
- Models can be broken down intc part appearance and spatial configuration
  - Wide variety of models



- Efficient optimization can be tricky but usually possible
  - Generalized distance transform is a useful trick



- Rather than explicitly modeling contextual relations, can encode through features/classifiers



Next class

- Visual tracking