Recognition III
EM, G/D, and SVMs

Linda Shapiro

CSE 455

What's Coming

- How does EM clustering work?
- How can it be used for generating features for image classification?
- A generative/discriminative system that uses both EM and neural nets
- How do support vector machines (SVMs) work?
- How are they used with the HOG feature for detection people (and other things)?
- What is the Deformable Parts Model (DPM)?

K-Means and EM Clustering

K-Means

- Clusters are represented by their means μ
- A vector is assigned to a single cluster whose mean it is closest to.

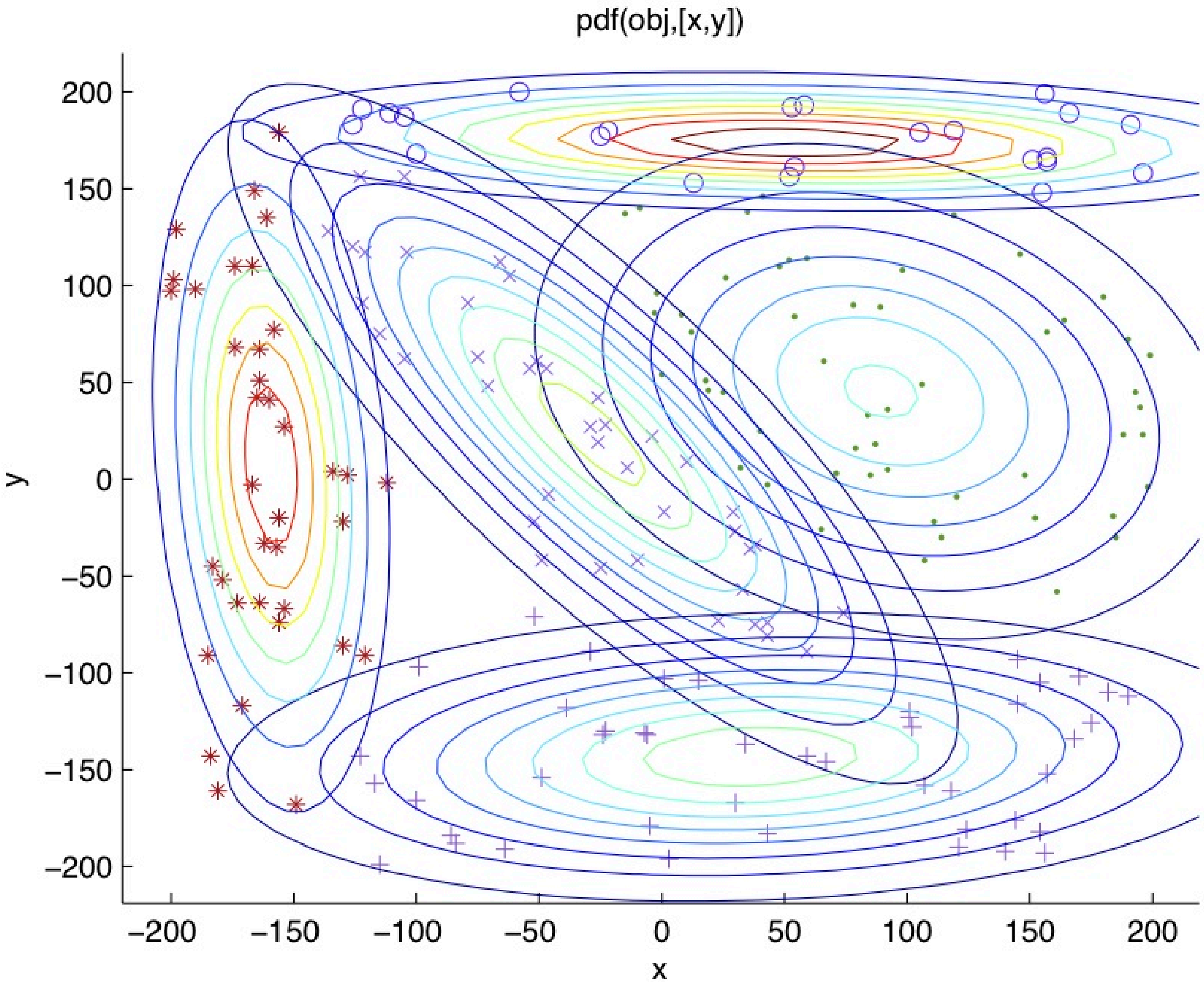
EM (Expectation-Maximization) Clustering

- Clusters are represented by their means, covariance matrices and weights μ, Σ, W
- A vector is soft assigned to each cluster (components) according to its probability of belonging to that cluster
- Usually we assume each cluster has a Gaussian distribution

K-Means and EM Clustering

- Both K-Means and EM start with a given K
- Both iterate between
  - assigning (or soft assigning) each vector to a (each) cluster
  - recalculating the parameters of each cluster
- The result of EM is called a mixture model and when using Gaussian components, it is a mixture of Gaussians

Mixture of Gaussians



Full EM Algorithm Multi-Dimensional

- **Boot Step:**
  - Initialize K clusters: $C_1, \ldots, C_K$

    (μ_j, Σ_j) and $P(C_j)$ for each cluster j.

- **Iteration Step:**
  - Expectation Step

$$p(C_j \mid x_i) = \frac{p(x_i \mid C_j) \cdot p(C_j)}{p(x_i)} = \frac{p(x_i \mid C_j) \cdot p(C_j)}{\sum_j p(x_i \mid C_j) \cdot p(C_j)}$$

  - Maximization Step

$$\mu_j = \frac{\sum_i p(C_j \mid x_i) \cdot x_i}{\sum_i p(C_j \mid x_i)} \qquad \Sigma_j = \frac{\sum_i p(C_j \mid x_i) \cdot (x_i - \mu_j) \cdot (x_i - \mu_j)^T}{\sum_i p(C_j \mid x_i)} \qquad p(C_j) = \frac{\sum_i p(C_j \mid x_i)}{N}$$

Yi Li's Generative Discriminative Classification: a BOW Approach

- Uses EM clustering with each component represented by a Gaussian distribution in feature space (color, texture, structure)
- Uses the K EM clusters to construct fixed length feature vectors representing positive and negative samples of the object class being learned.
- Feeds these vectors into a neural net to learn the class.

Problem Statement

Given: Some images and their corresponding descriptions




•••

{trees, grass, cherry trees} {cheetah, trunk} {mountains, sky} {beach, sky, trees, water}

To solve: What object classes are present in new images



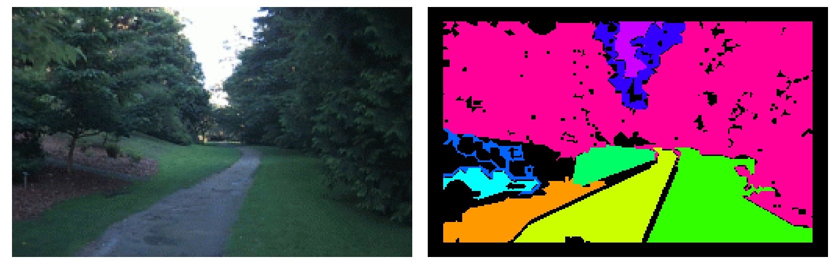

•••

? ? ? ?

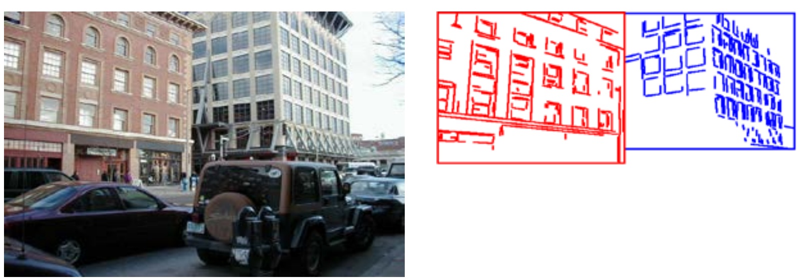
Image Features for Object Recognition

- Color

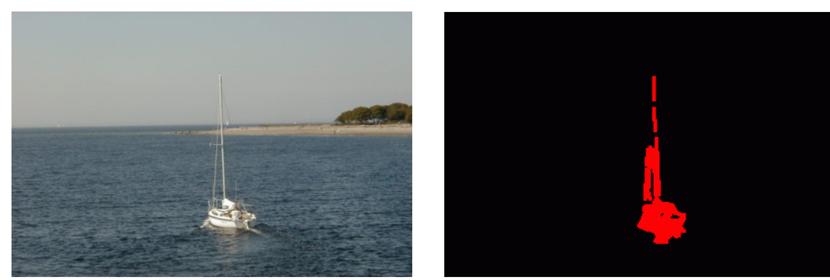
- Texture



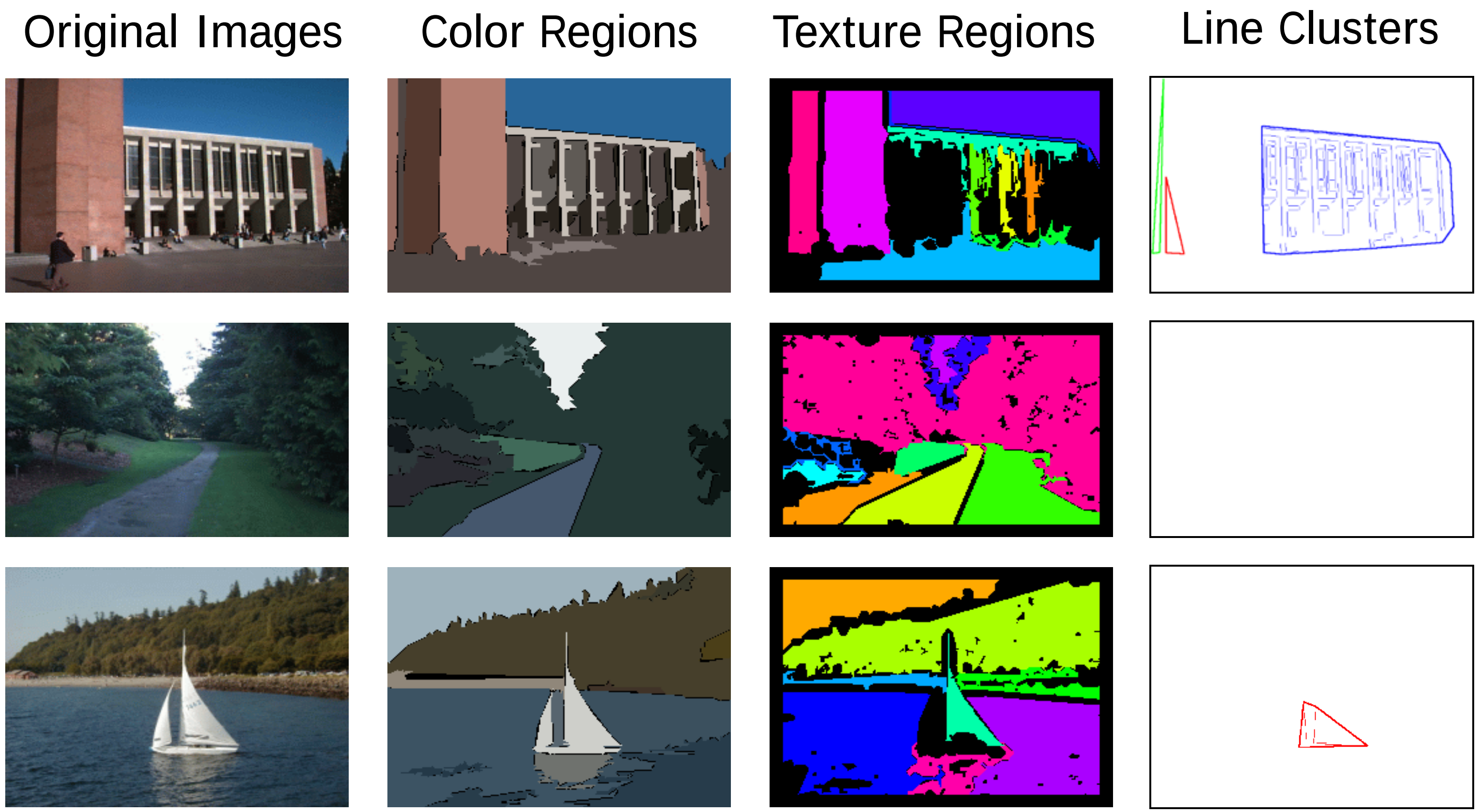
- Structure



- Context



Abstract Regions



Approach to Combining Different Feature Types

Phase 1:

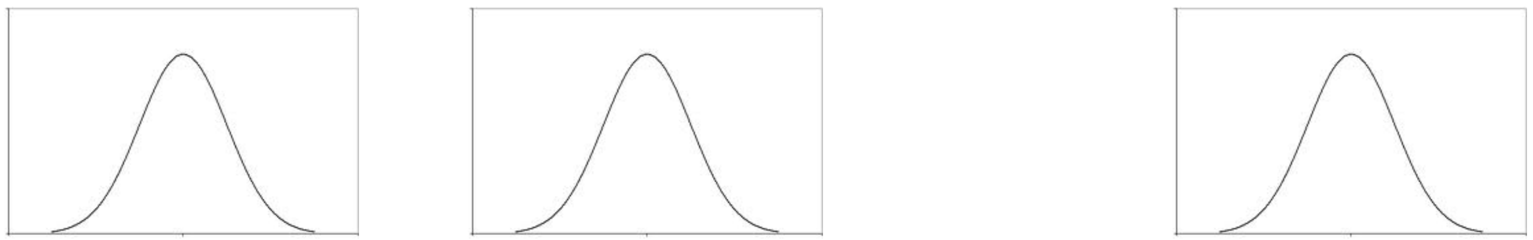
- Treat each type of abstract region separately
- For abstract region type a and for object class o, use the EM algorithm to construct a model that is a mixture of multivariate Gaussians over the features for type a regions.

Consider only abstract region type color (c) and object class object (o)

- At the end of Phase 1, we can compute the distribution of color feature vector X in an image containing object *o*.
- $$P(X \mid o) = \sum_{k=1 \text{ to } K} w_k \, N(X, \mu_k, \Sigma_k)$$
- K is the number of Gaussian components.
- The w's are the weights of the components.
- The μ's and Σ's are the parameters of the Gaussians
- P(X | o) is the probability of vector X given object o.

Color Clusters for Class o
We call them components.

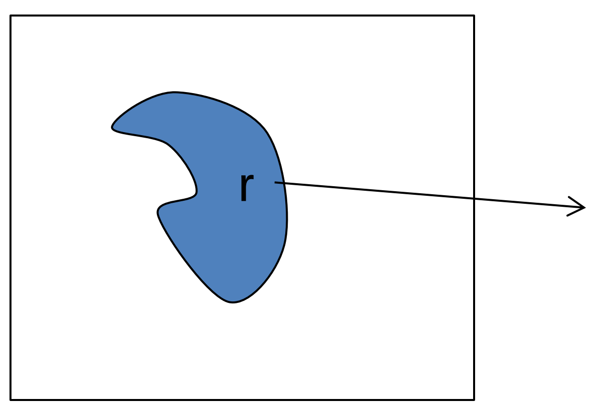
$$P(X \mid o) = \sum w_k \, N(X, \mu_k, \Sigma_k)$$



component 1 — $\mu_1 , \sum_1 , w_1$

component 2 — $\mu_2 , \sum_2 , w_2$

component K — $\mu_K , \sum_K , w_K$

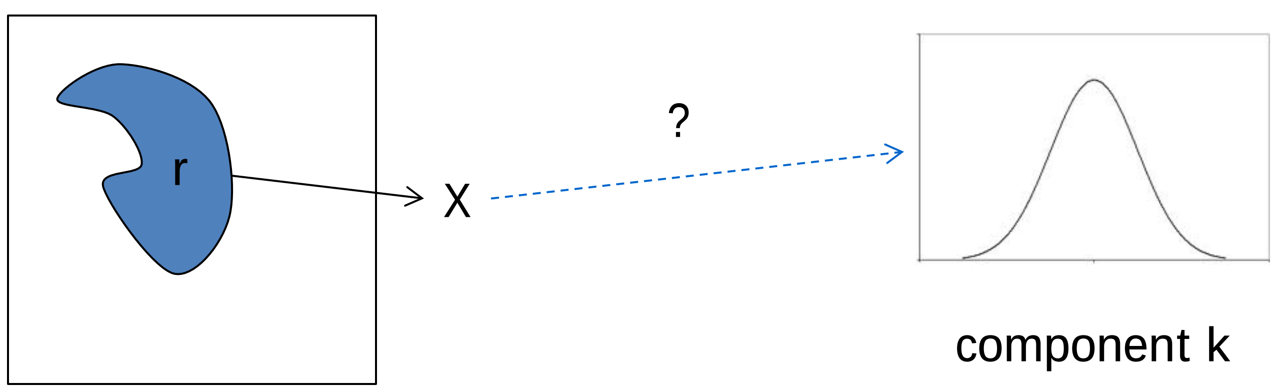


color feature vector
X for some region r

Now we can determine which components are likely to be present in an image.

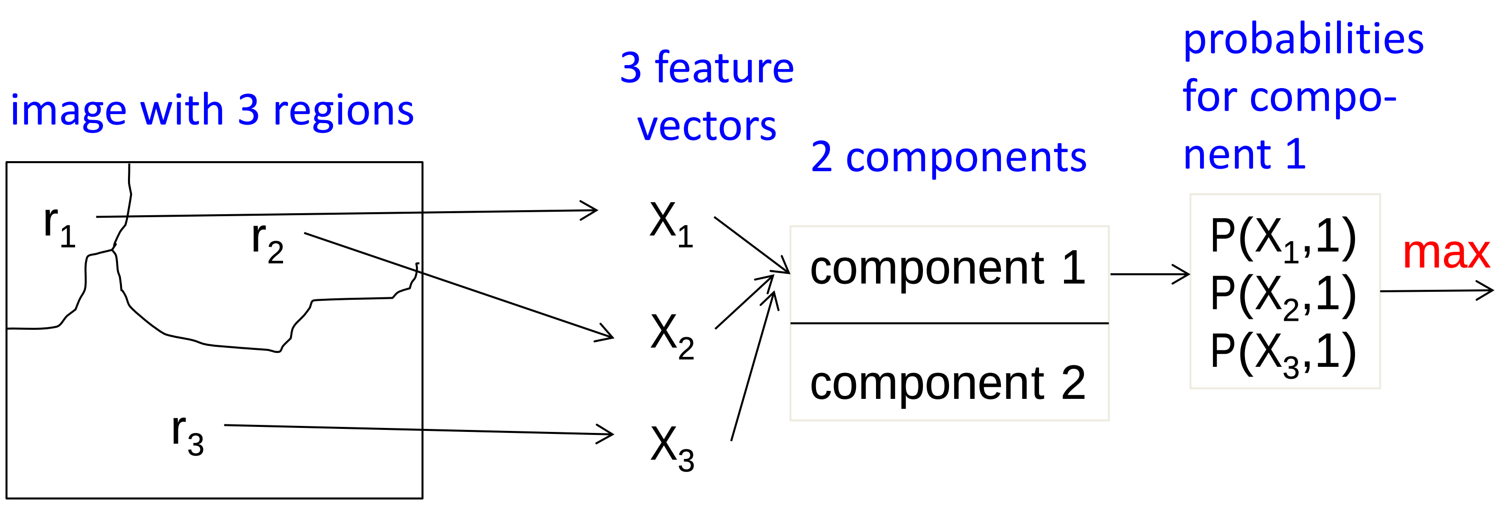
- The probability that the feature vector X from color region r of image I_i comes from component k is given by
- P(X,k) = w_k * N(X, μ_k, Σ_k)

$$f_{\mathbf{x}}(x_1,\ldots,x_k) = \frac{1}{(2\pi)^{k/2}|\boldsymbol{\Sigma}|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right)$$



And determine the probability that the whole image is related to component k as a function of the feature vectors of all its regions.

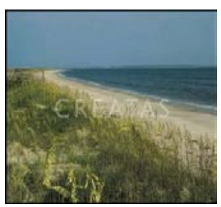
- Then the probability that image *I* has a region r that comes from component k is
- $P(I, k) = \max(P(X_r, k \mid r = 1,2,\ldots))$



Aggregate Scores for Color

maximum probability of all regions in first beach image belonging to component 1

Components

		1	2	3	4	5	6	7	8
beach		.93	.16	.94	.24	.10	.99	.32	.00
beach		.66	.80	.00	.72	.19	.01	.22	.02
not beach		.43	.03	.00	.00	.00	.00	.15	.00

beach training images

not beach training images

We now use positive and negative training images, calculate for each the probabilities of regions of each component, and form a training matrix.

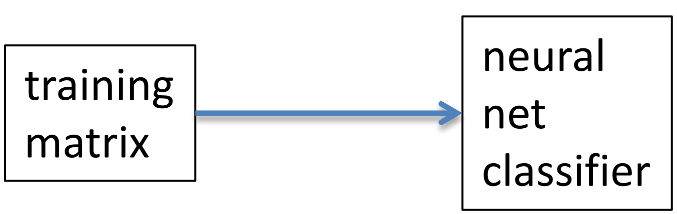
$$
\begin{bmatrix}
P(beach_1,1) & P(beach_1,2) & \ldots & P(beach_1,K) \\
P(beach_2,1) & P(beach_2,2) & \ldots & P(beach_2,K) \\
 & & & \\
P(beach_n,1) & P(beach_n,2) & \ldots & P(beach_n,K) \\
P(nbeach_1,1) & P(nbeach_1,2) & \ldots & P(nbeach_1,K) \\
P(nbeach_2,1) & P(nbeach_2,2) & \ldots & P(nbeach_2,K) \\
 & & & \\
P(nbeach_m,1) & P(nbeach_m,2) & \ldots & P(nbeach_m,K)
\end{bmatrix}
$$

Phase 2 Learning

- Let C_i be row i of the training matrix.

- Each such row is a feature vector for the color features of regions of image I_i that relates them to the Phase 1 components.

- Now we can use a second-stage classifier to learn $P(o|I_i)$ for each object class o and image I_i.



Multiple Feature Case

- We calculate separate Gaussian mixture models for each different features type:

- Color: C_i
- Texture: T_i
- Structure: S_i

- and any more features we have (motion).

Now we concatenate the matrix rows from the different region types to obtain a multi-feature-type training matrix.

$$\overset{\textit{color}}{\begin{pmatrix} C_1^+ \\ C_2^+ \\ . \\ . \\ C_1^- \\ C_2^- \\ . \\ . \end{pmatrix}} \quad \overset{\textit{texture}}{\begin{pmatrix} T_1^+ \\ T_2^+ \\ . \\ . \\ T_1^- \\ T_2^- \\ . \\ . \end{pmatrix}} \quad \overset{\textit{structure}}{\begin{pmatrix} S_1^+ \\ S_2^+ \\ . \\ . \\ S_1^- \\ S_2^- \\ . \\ . \end{pmatrix}} \longrightarrow \overset{\textit{everything}}{\begin{pmatrix} C_1^+ & T_1^+ & S_1^+ \\ C_2^+ & T_2^+ & S_2^+ \\ . & . & . \\ . & . & . \\ C_1^- & T_1^- & S_1^- \\ C_2^- & T_2^- & S_2^- \\ . & . & . \\ . & . & . \end{pmatrix}}$$

Final Neural Net Classifier

Training

$$\begin{pmatrix} C_1^+ & T_1^+ & S_1^+ \\ C_2^+ & T_2^+ & S_2^+ \\ \vdots & \vdots & \vdots \\ C_1^- & T_1^- & S_1^- \\ C_2^- & T_2^- & S_2^- \\ \vdots & \vdots & \vdots \end{pmatrix} \rightarrow \boxed{\text{Neural Net for beach}}$$

Testing

$\rightarrow \boxed{\text{Neural Net for beach}} \rightarrow$ NOT BEACH

ICPR04 Data Set with General Labels

	EM-variant with single Gaussian per object	EM-variant extension to mixture models	Gen/Dis with Classical EM clustering	Gen/Dis with EM-variant extension
African animal	71.8%	85.7%	89.2%	90.5%
arctic	80.0%	79.8%	90.0%	85.1%
beach	88.0%	90.8%	89.6%	91.1%
grass	76.9%	69.6%	75.4%	77.8%
mountain	94.0%	96.6%	97.5%	93.5%
primate	74.7%	86.9%	91.1%	90.9%
sky	91.9%	84.9%	93.0%	93.1%
stadium	95.2%	98.9%	99.9%	100.0%
tree	70.7%	79.0%	87.4%	88.2%
water	82.9%	82.3%	83.1%	82.4%
MEAN	**82.6%**	**85.4%**	**89.6%**	**89.3%**

Comparison to ALIP: the Benchmark Image Set

	ALIP	cs	ts	st	ts+st	cs+st	cs+ts	cs+ts+st
African	52	69	23	26	35	79	72	74
beach	32	44	38	39	51	48	59	64
buildings	64	43	40	41	67	70	70	78
buses	46	60	72	92	86	85	84	95
dinosaurs	100	88	70	37	86	89	94	93
elephants	40	53	8	27	38	64	64	69
flowers	90	85	52	33	78	87	86	91
food	68	63	49	41	66	77	84	85
horses	60	94	41	50	64	92	93	89
mountains	84	43	33	26	43	63	55	65
MEAN	**63.6**	**64.2**	**42.6**	**41.2**	**61.4**	**75.4**	**76.1**	**80.3**

Groundtruth Data Set

- UW Ground truth database (1224 images)
- 31 elementary object categories: *river* (30), *beach* (31), *bridge* (33), *track* (35), *pole* (38), *football field* (41), *frozen lake* (42), *lantern* (42), *husky stadium* (44), *hill* (49), *cherry tree* (54), *car* (60), *boat* (67), *stone* (70), *ground* (81), *flower* (85), *lake* (86), *sidewalk* (88), *street* (96), *snow* (98), *cloud* (119), *rock* (122), *house* (175), *bush* (178), *mountain* (231), *water* (290), *building* (316), *grass* (322), *people* (344), *tree* (589), *sky* (659)
- 20 high-level concepts: *Asian city* , *Australia*, *Barcelona*, *campus*, *Cannon Beach*, *Columbia Gorge*, *European city*, *Geneva*, *Green Lake*, *Greenland*, *Indonesia*, *indoor*, *Iran*, *Italy*, *Japan*, *park*, *San Juans*, *spring flowers*, *Swiss mountains*, *and Yellowstone*.

beach, sky, tree, water

people, street, tree

building, grass, people, sidewalk, sky, tree

building, bush, sky, tree, water

flower, house, people, pole, sidewalk, sky

flower, grass, house, pole, sky, street, tree

building, flower, sky, tree, water

boat, rock, sky, tree, water

building, car, people, tree

car, people, sky

boat, house, water

building

Groundtruth Data Set: ROC Scores

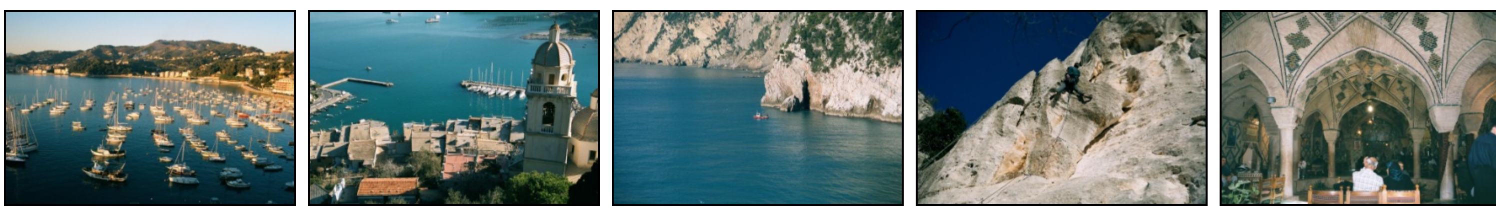
street	60.4	*tree*	80.8	*stone*	87.1	*columbia gorge*	94.5
people	68.0	*bush*	81.0	*hill*	87.4	*green lake*	94.9
rock	73.5	*flower*	81.1	*mountain*	88.3	*italy*	95.1
sky	74.1	*iran*	82.2	*beach*	89.0	*swiss moutains*	95.7
ground	74.3	*bridge*	82.7	*snow*	92.0	*sanjuans*	96.5
river	74.7	*car*	82.9	*lake*	92.8	*cherry tree*	96.9
grass	74.9	*pole*	83.3	*frozen lake*	92.8	*indoor*	97.0
building	75.4	*yellowstone*	83.7	*japan*	92.9	*greenland*	98.7
cloud	75.4	*water*	83.9	*campus*	92.9	*cannon beach*	99.2
boat	76.8	*indonesia*	84.3	*barcelona*	92.9	*track*	99.6
lantern	78.1	*sidewalk*	85.7	*geneva*	93.3	*football field*	99.8
australia	79.7	*asian city*	86.7	*park*	94.0	*husky stadium*	100.0
house	80.1	*european city*	87.0	*spring flowers*	94.4		

Groundtruth Data Set: Top Results

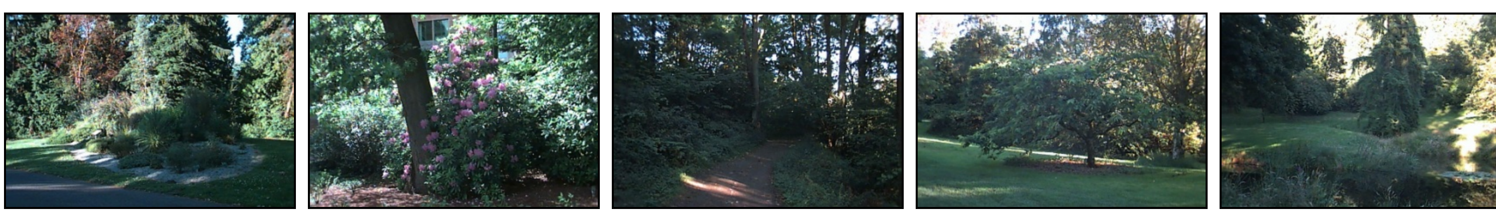
Asian city

Cannon beach

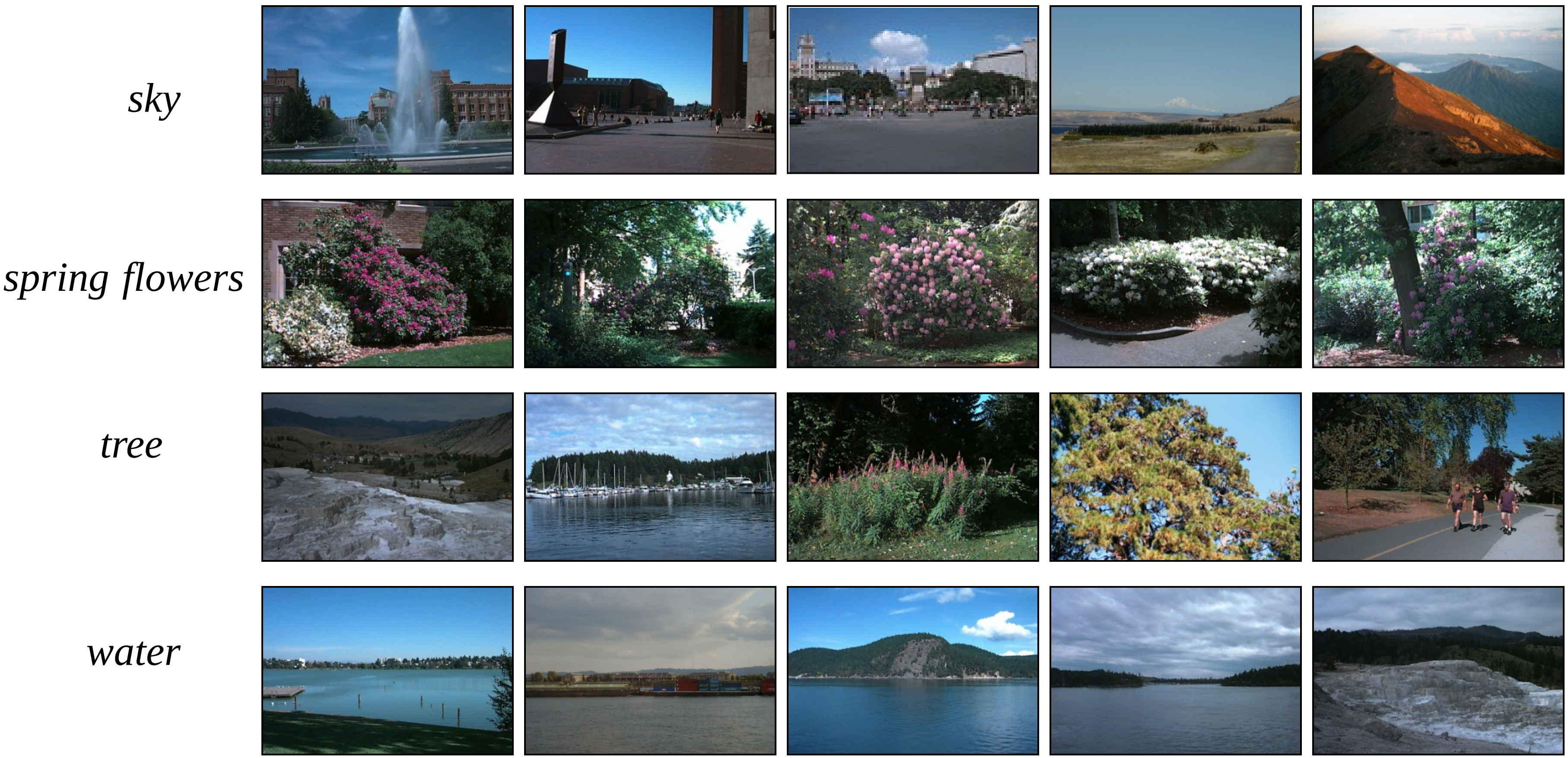
Italy



park



Groundtruth Data Set: Top Results



Groundtruth Data Set: Annotation Samples

tree(97.3), **bush**(91.6), **spring flowers**(90.3), **flower**(84.4), park(84.3), **sidewalk**(67.5), **grass**(52.5), **pole**(34.1)

sky(99.8), **Columbia gorge**(98.8), lantern(94.2), **street**(89.2), house(85.8), bridge(80.8), car(80.5), hill(78.3), boat(73.1), pole(72.3), **water**(64.3), mountain(63.8), **building**(9.5)

sky(95.1), **Iran**(89.3), house(88.6), **building**(80.1), boat(71.7), bridge(67.0), **water**(13.5), **tree**(7.7)

Italy(99.9), grass(98.5), **sky**(93.8), rock(88.8), **boat**(80.1), **water**(77.1), Iran(64.2), stone(63.9), bridge(59.6), **European**(56.3), sidewalk(51.1), **house**(5.3)

Comparison to Fergus and to Dorko/Schmid using their Features

Using their features and image sets, we compared our generative / discriminative approach to those of Fergus and Dorko/Schmid.

The image set contained 1074 airplane images, 826 motor bike images, 450 face images, and 900 background. Half were used to train and half to test. We added half the background images to the training set for our negative examples.

	Fergus	Dorko/Schmid	Ours
airplanes	90.2%	96.0%	96.6%
faces	96.4%	96.8%	96.5%
motorbikes	92.5%	98.0%	99.2%

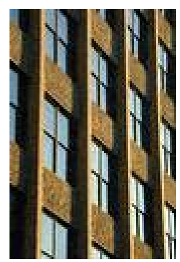
Structure Feature Experiments

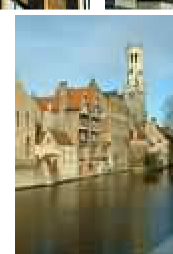
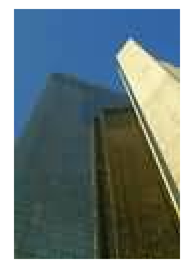
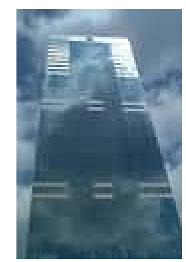
(from other data sets with more manmade structures)

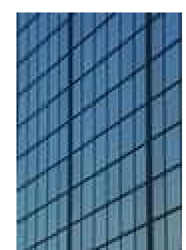
- 1,951 total from freefoto.com
- bus (1,013) house/building (609) skyscraper (329)









Structure Feature Experiments: Area Under the ROC Curves

1. Structure (with color pairs)
   - Attributes (10)
     - Color pair
     - Number of lines
     - Orientation of lines
     - Line overlap
     - Line intersection

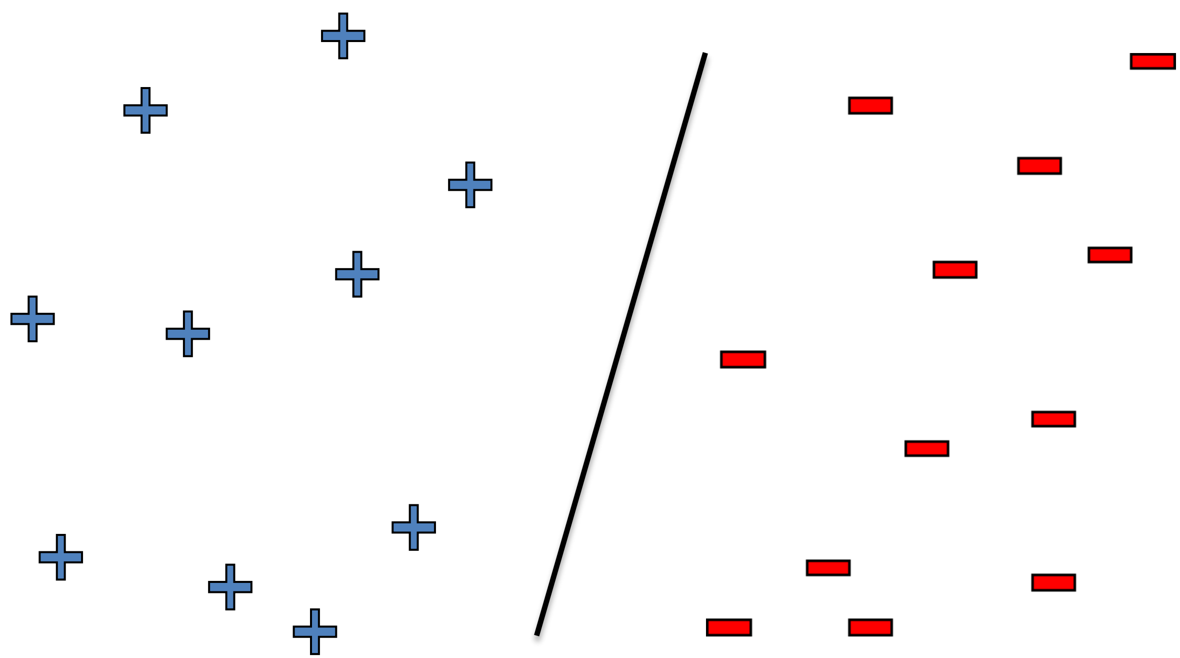
2. Structure (with color pairs) + Color Segmentation

3. Structure (without color pairs) + Color Segmentation

	bus	house/ building	skyscraper
Structure only	0.900	0.787	0.887
Structure + Color Seg	0.924	0.853	0.926
Structure^2 + Color Seg	0.940	0.860	0.919

Support Vector Machines

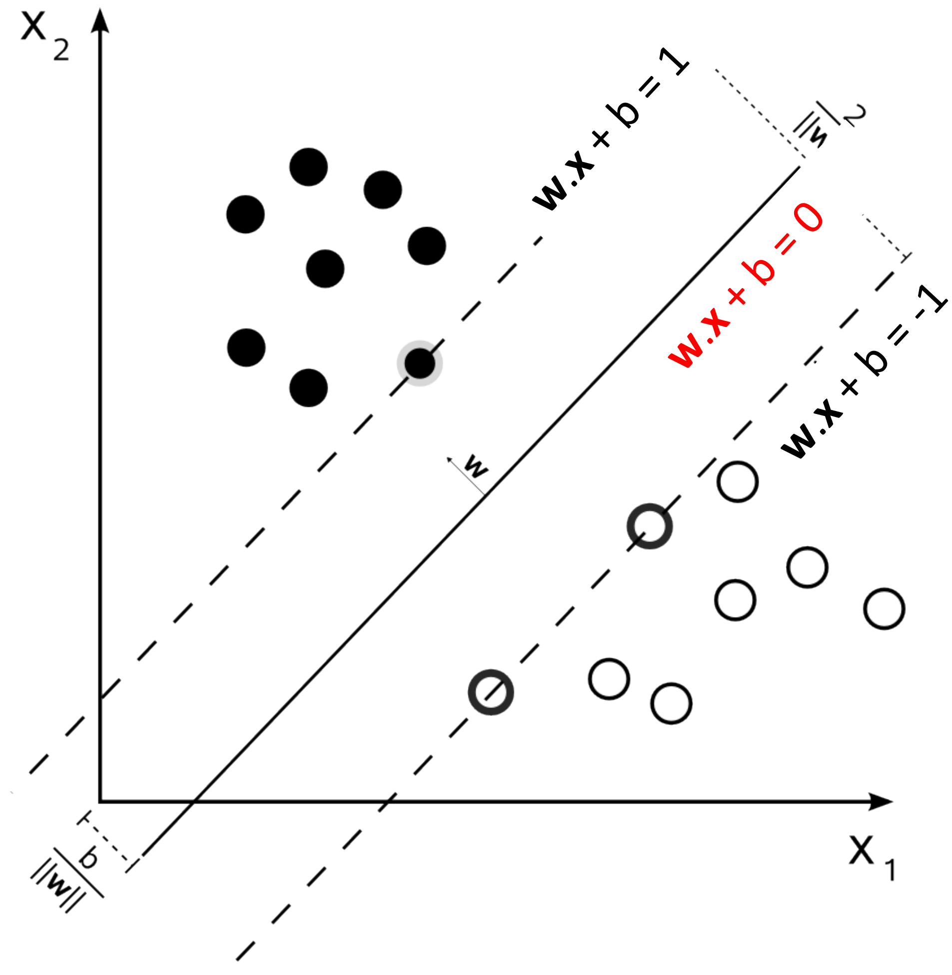
- 2 class problem
- n-dimensional feature vectors
- We want to separate the two classes with an (n-1)-dimensional hyperplane



Linear SVM

Given a training dataset
($\mathbf{X}_1$,y_1), ... , ($\mathbf{X}_n$,yn)
where y_i is 1 or -1, find the
maximum-margin hyperplane
that divides the y=1 group from
the y=-1 group.

The hyperplane will have an
equation of the form
w x + b = 0
where w is the normal vector
to the hyperplane.



modified from https://commons.wikimedia.org/w/index.php?curid=3566688

Hard Margin SVM

- If the training data are linearly separable, we select two parallel hyperplanes that separate the two classes and maximize the distance between them.
- The region between them is called the margin.
- The two hyperplanes are:
  - $\mathbf{w} \cdot \mathbf{x} + b = 1$
  - $\mathbf{w} \cdot \mathbf{x} + b = -1$
- The distance between them is $2/||\mathbf{w}||$
- To prevent data points falling into the margin, we add constraints:

$$\left.\begin{array}{l} \mathbf{w} \cdot \mathbf{x_i} + b \geq 1 \text{ if } y_i = 1 \\ \mathbf{w} \cdot \mathbf{x_i} + b \leq 1 \text{ if } y_i = -1 \end{array}\right\} \quad y_i(\mathbf{w} \cdot x_i + b) \geq 1, \quad 1 \leq i \leq n$$

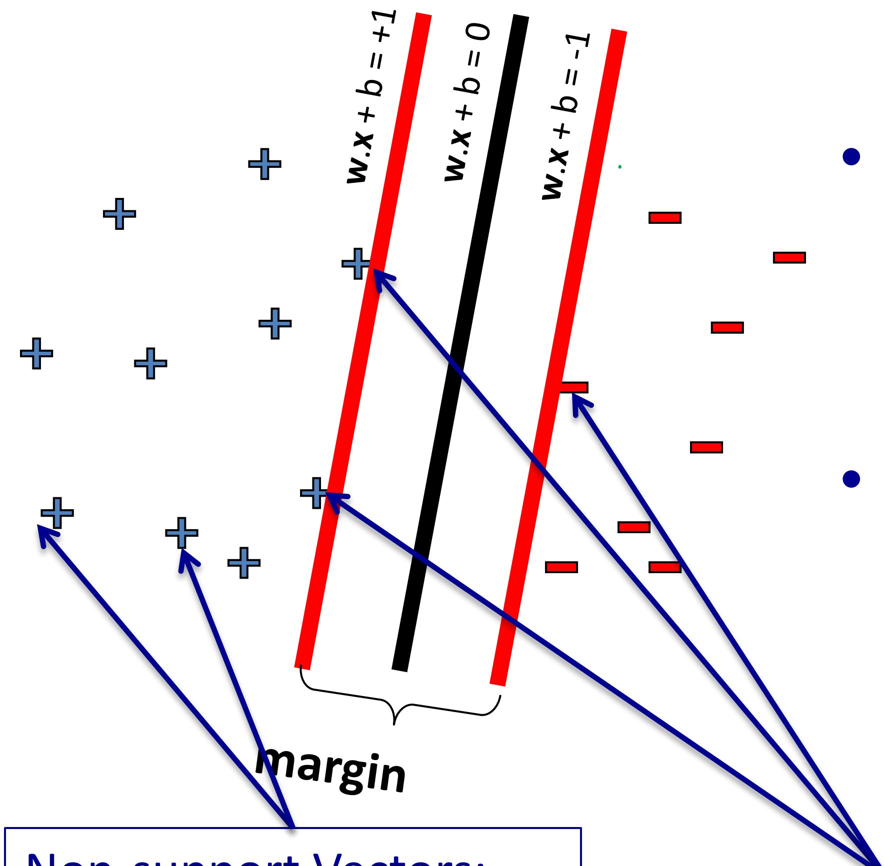
Solution: minimize w,b subject to $y_i (\mathbf{w} \cdot x_i + b) \geq 1$

Classifier: $\text{class}(\mathbf{x}) = \text{sgn}(\mathbf{w} \cdot \mathbf{x} + b)$

Support vector machines (SVMs)

$$\text{minimize}_{\mathbf{w},b} \quad \mathbf{w}.\mathbf{w}$$
$$\left(\mathbf{w}.\mathbf{x}_j + b\right) y_j \geq 1, \ \forall j$$

- Solve efficiently by quadratic programming (QP)
  - Well-studied solution algorithms
  - Not simple gradient ascent, but close
- Hyperplane defined by support vectors



Non-support Vectors:
- everything else
- moving them will not change w

Support Vectors:
- data points on the canonical lines

A bit of the theory

- We want to transform vectors x to another space $\varphi(x)$ where the classes can be separated.
- The kernel is related to this transform by
  $k(x_i, x_j) = \varphi(x_i) \cdot \varphi(x_j)$
- When performing classification that needs w, instead of $w \cdot \varphi(x)$ for some vector x, we use the kernel trick of replacing it with
  $\Sigma_i\, \alpha_i\, y_i\, k(x_i,x)$ where $w = \Sigma_i\, \alpha_i\, y_i\, \varphi(x_i)$

Classification

- The results for classification of a vector z becomes
- class = sgn(w ·φ(z)+b)
- $= \text{sgn}\left(\left[\sum_{i=1 \text{ to } n} c_i y_i k(x_i, z) \right] + b \right)$

Another Descriptor

Histograms of Oriented Gradients for Human Detection

Navneet Dalal and Bill Triggs

CVPR 2005

Overview

1. Compute gradients in the region to be described
2. Put them in bins according to orientation
3. Group the cells into large blocks
4. Normalize each block
5. Train classifiers to decide if these are parts of a human

Details

- Gradients

  $[-1\ 0\ 1]$ and $[-1\ 0\ 1]^T$ were good enough.

- Cell Histograms

  Each pixel within the cell casts a weighted vote for an orientation-based histogram channel based on the values found in the gradient computation. (9 channels worked)

- Blocks

  Group the cells together into larger blocks, either R-HOG blocks (rectangular) or C-HOG blocks (circular).

More Details

- Block Normalization

They tried 4 different kinds of normalization.
Let υ be the block to be normalized and e be a small constant.

L2-norm: $f = \frac{v}{\sqrt{\|v\|_2^2 + e^2}}$

L2-hys: L2-norm followed by clipping (limiting the maximum values of v to 0.2) and renormalizing,

L1-norm: $f = \frac{v}{(\|v\|_1 + e)}$

L1-sqrt: $f = \sqrt{\frac{v}{(\|v\|_1 + e)}}$

R-HOG compared to SIFT Descriptor

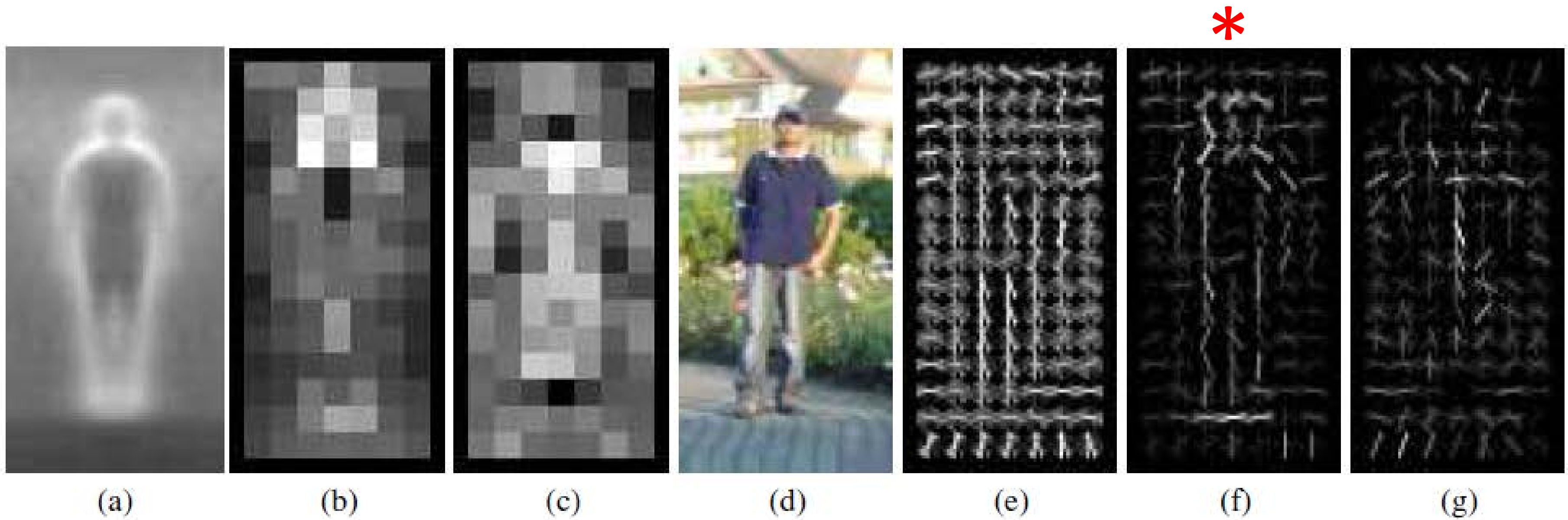
- R-HOG blocks appear quite similar to the SIFT descriptors.
- But, R-HOG blocks are computed in dense grids at some single scale without orientation alignment.
- SIFT descriptors are computed at sparse, scale-invariant key image points and are rotated to align orientation.

Example: Dalal-Triggs pedestrian

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

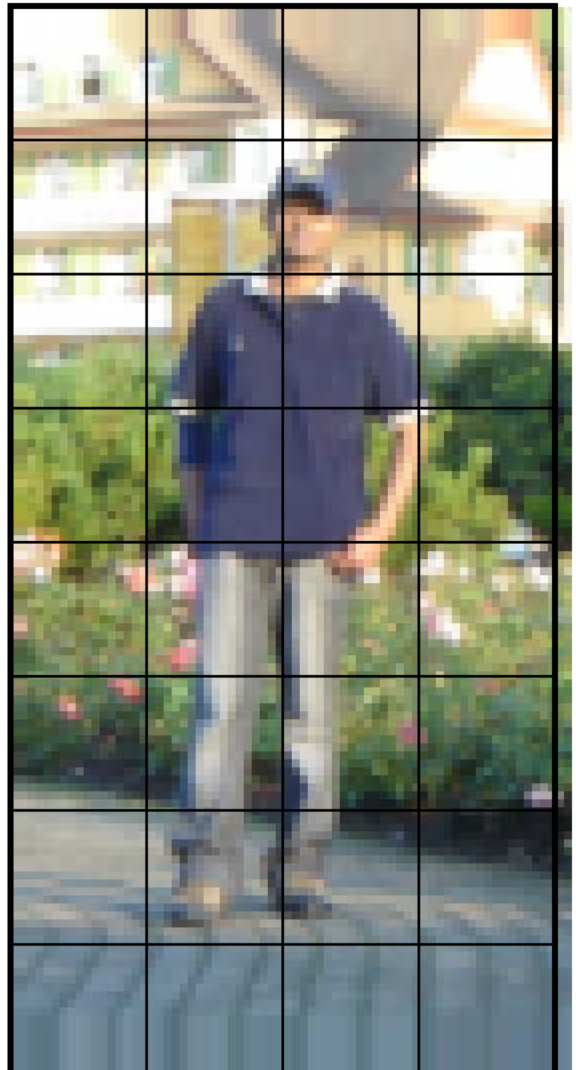
Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05

Pictorial Example



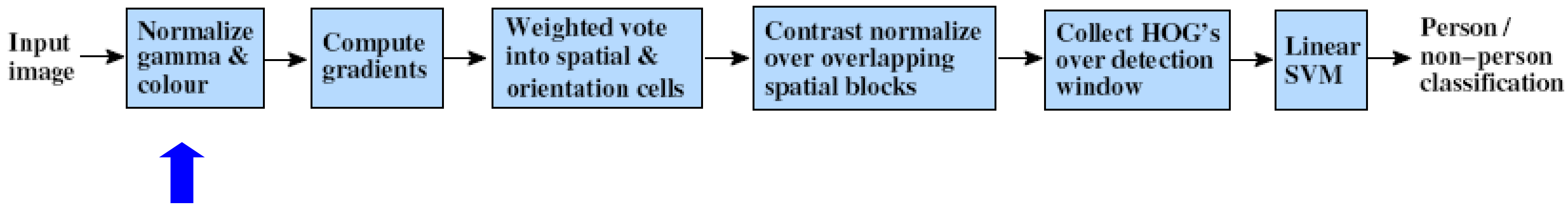
(a) average gradient image over training examples
(b) each “pixel” shows max positive SVM weight in the block centered on that pixel
(c) same as (b) for negative SVM weights
(d) test image
(e) its R-HOG descriptor
(f) R-HOG descriptor weighted by positive SVM weights
(g) R-HOG descriptor weighted by negative SVM weights

Input image → Normalize gamma & colour → Compute gradients → Weighted vote into spatial & orientation cells → Contrast normalize over overlapping spatial blocks → Collect HOG's over detection window → Linear SVM → Person / non–person classification



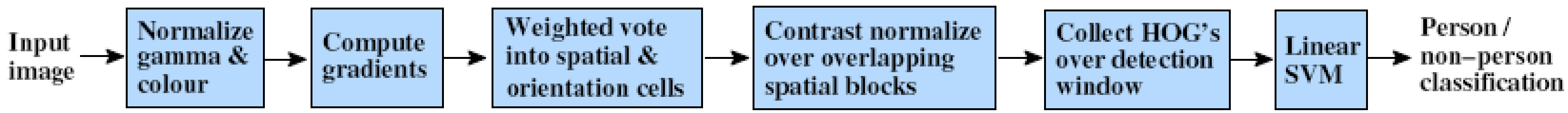
Slides by Pete Barnum

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



- Tested with
  - RGB
  - LAB
  - Grayscale

Slightly better performance vs. grayscale



Outperforms

-1	0	1

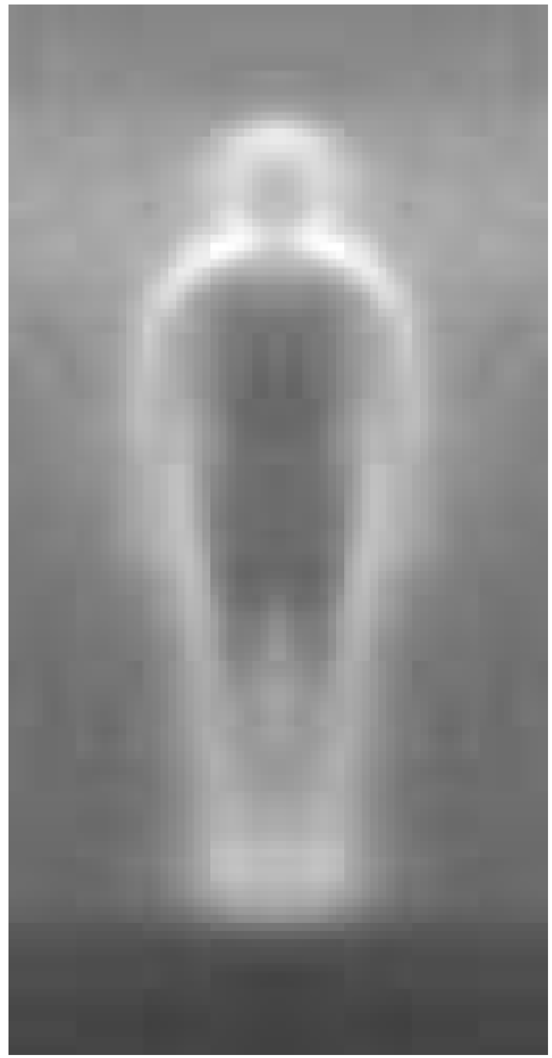
centered

-1	1

uncentered

1	-8	0	8	-1

cubic-corrected



0	1
-1	0

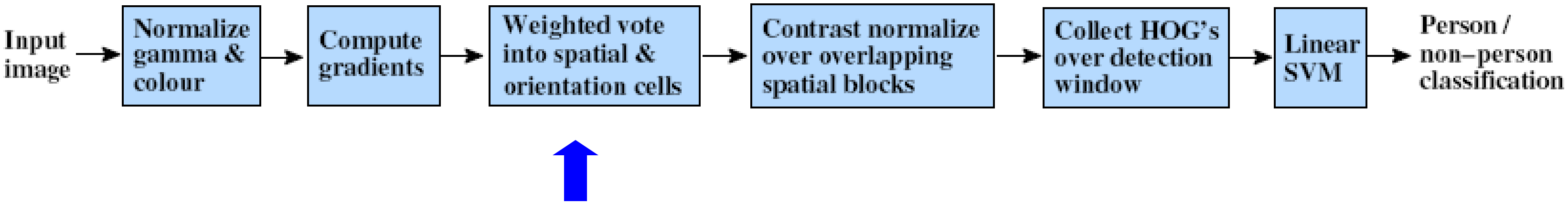
diagonal

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

Sobel

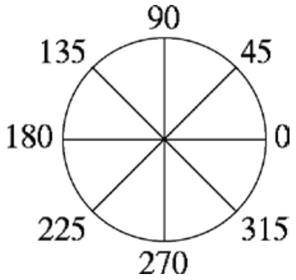
Slides by Pete Barnum

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

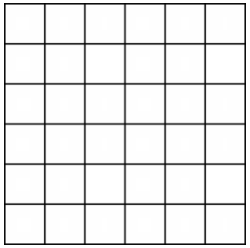


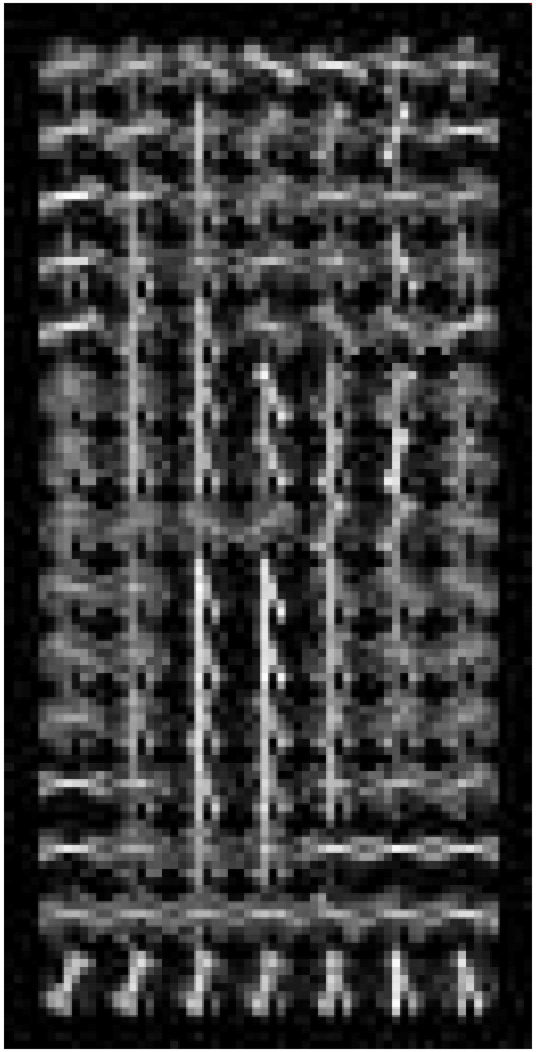
- Histogram of gradient orientations

Orientation: 9 bins (for unsigned angles)



Histograms in 8x8 pixel cells

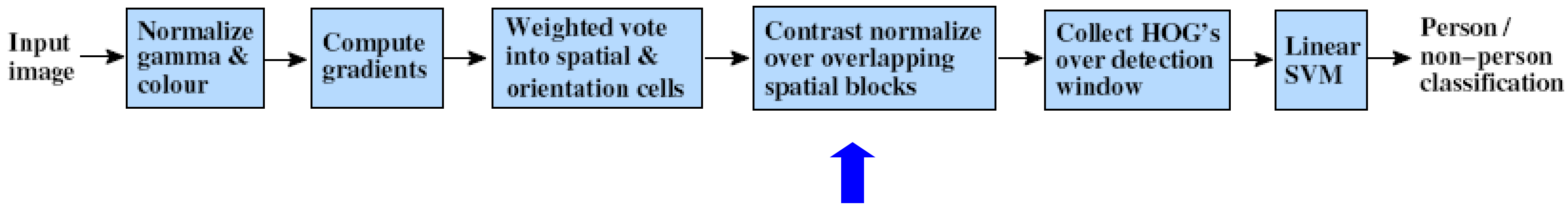




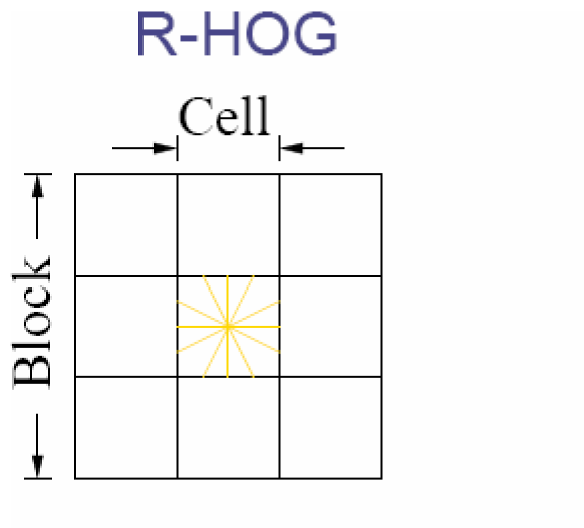
– Votes weighted by magnitude
– Bilinear interpolation between cells

Slides by Pete Barnum

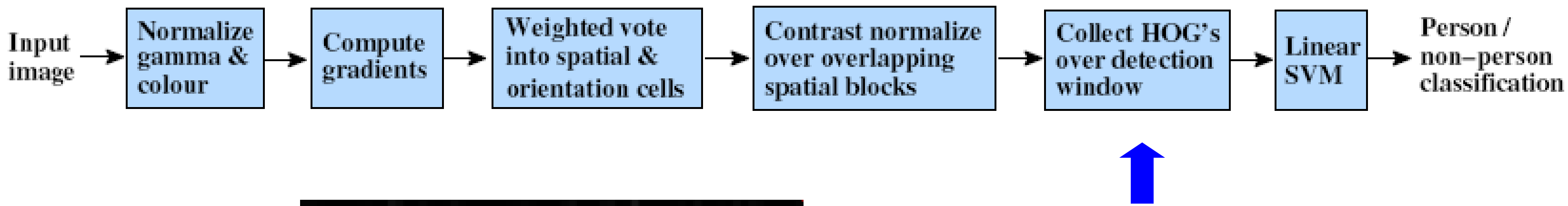
Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05



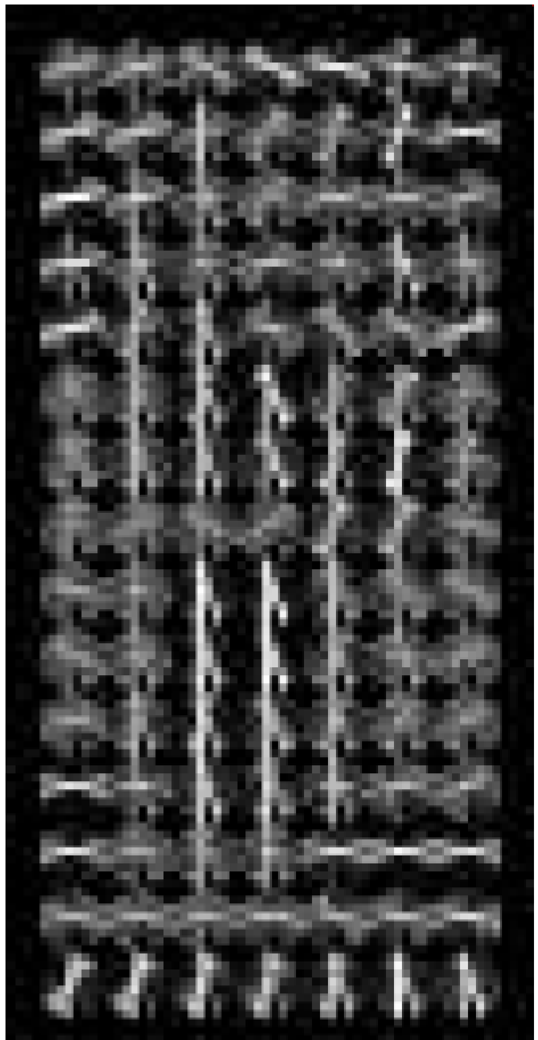
Normalize with respect to surrounding cells



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



X=



orientations

features = 15 x 7 x 9 x 4 = 3780

cells

normalizations by neighboring cells

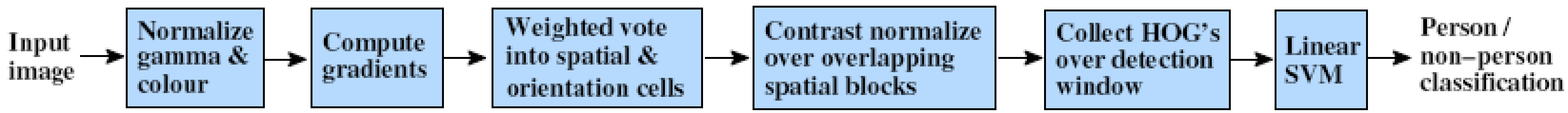
Slides by Pete Barnum

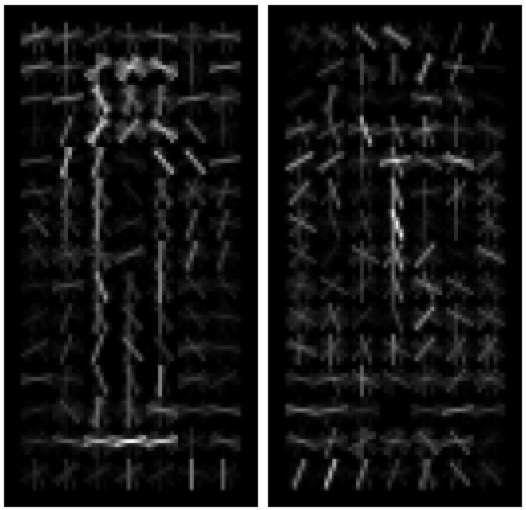
Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05

Training set

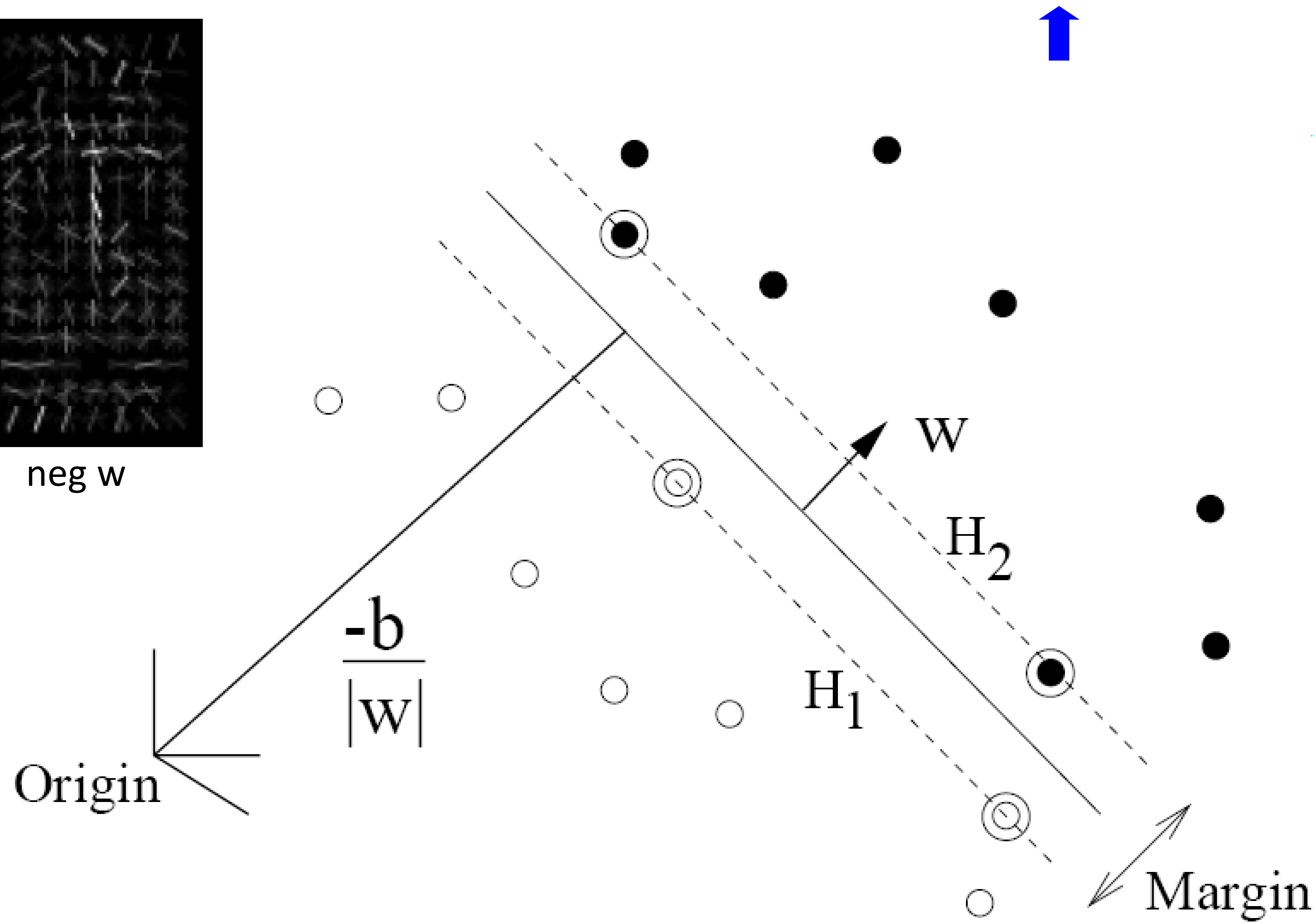
+

−



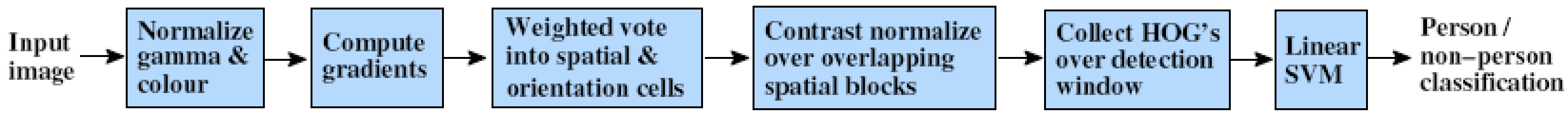


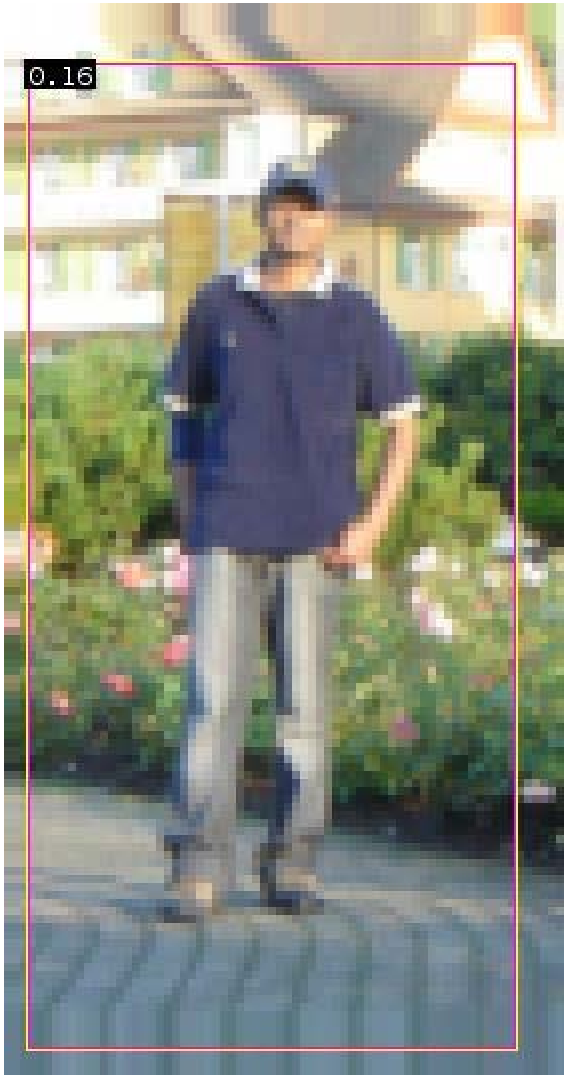
pos w neg w



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

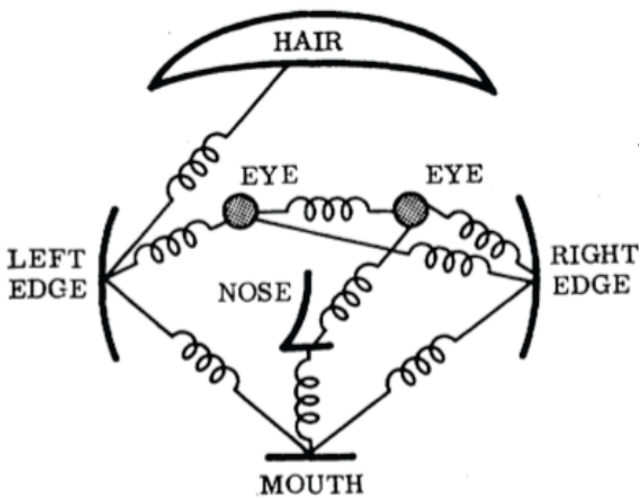
Deformable Parts Model

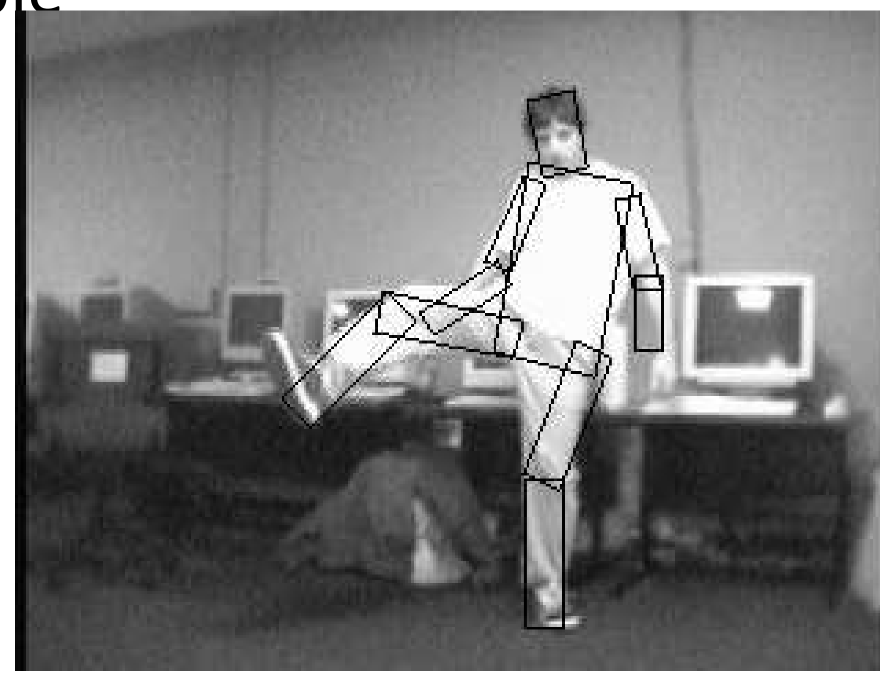
- Takes the idea a little further
- Instead of one rigid HOG model, we have multiple HOG models in a spatial arrangement
- One root part to find first and multiple other parts in a tree structure.

The Idea

Articulated parts model

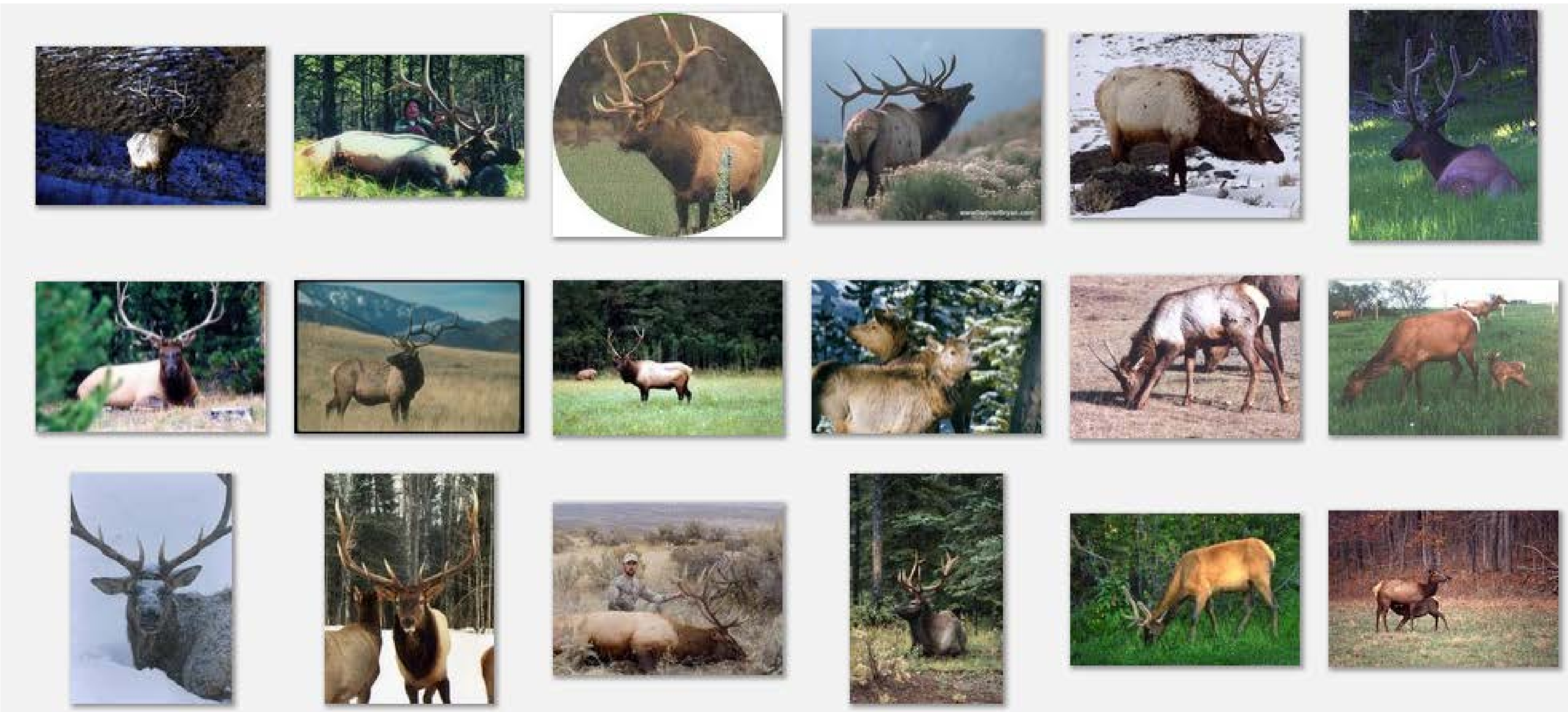
- Object is configuration of parts
- Each part is detectable





Images from Felzenszwalb

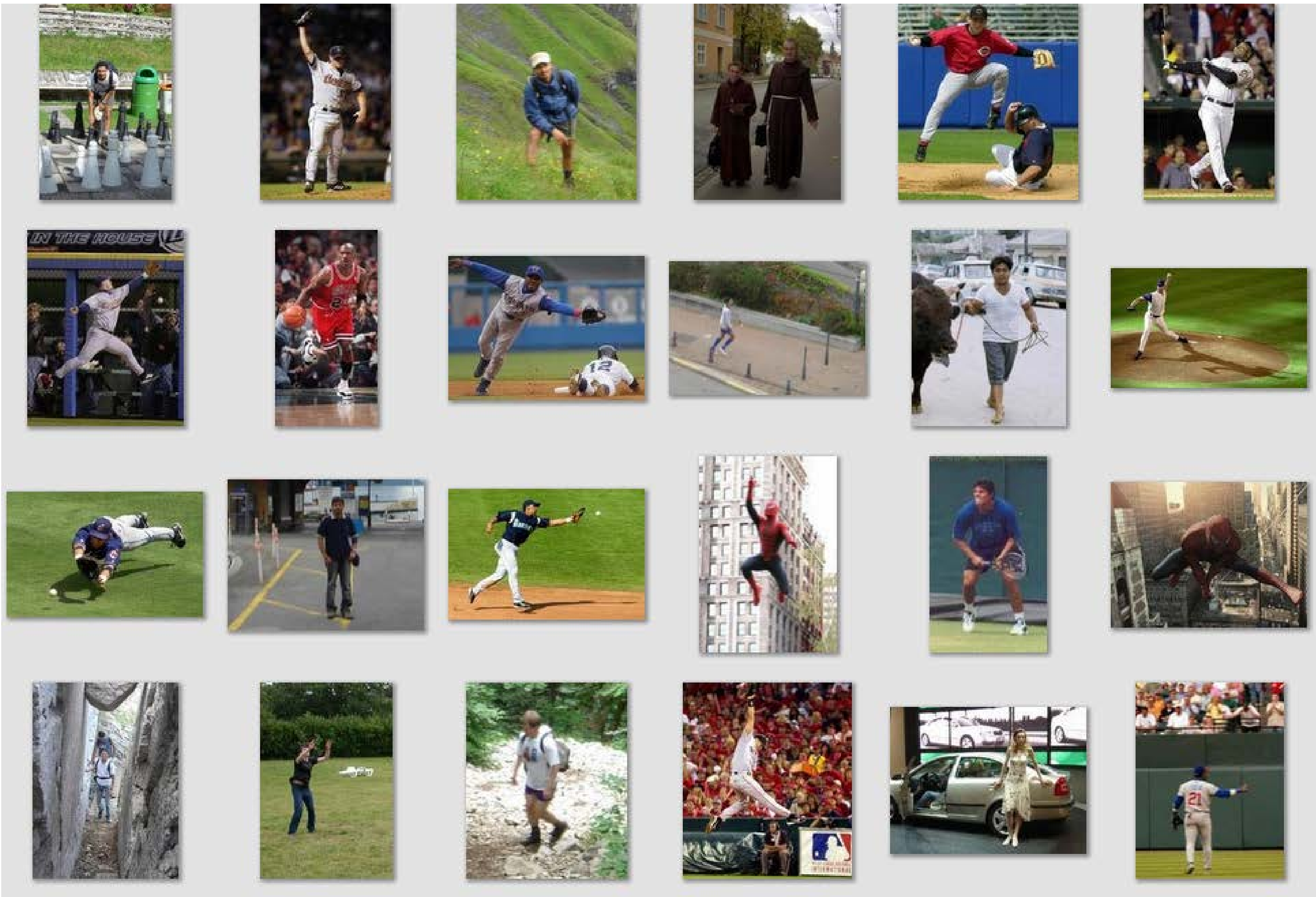
Deformable objects



Images from Caltech-256

Slide Credit: Duan Tran

Deformable objects

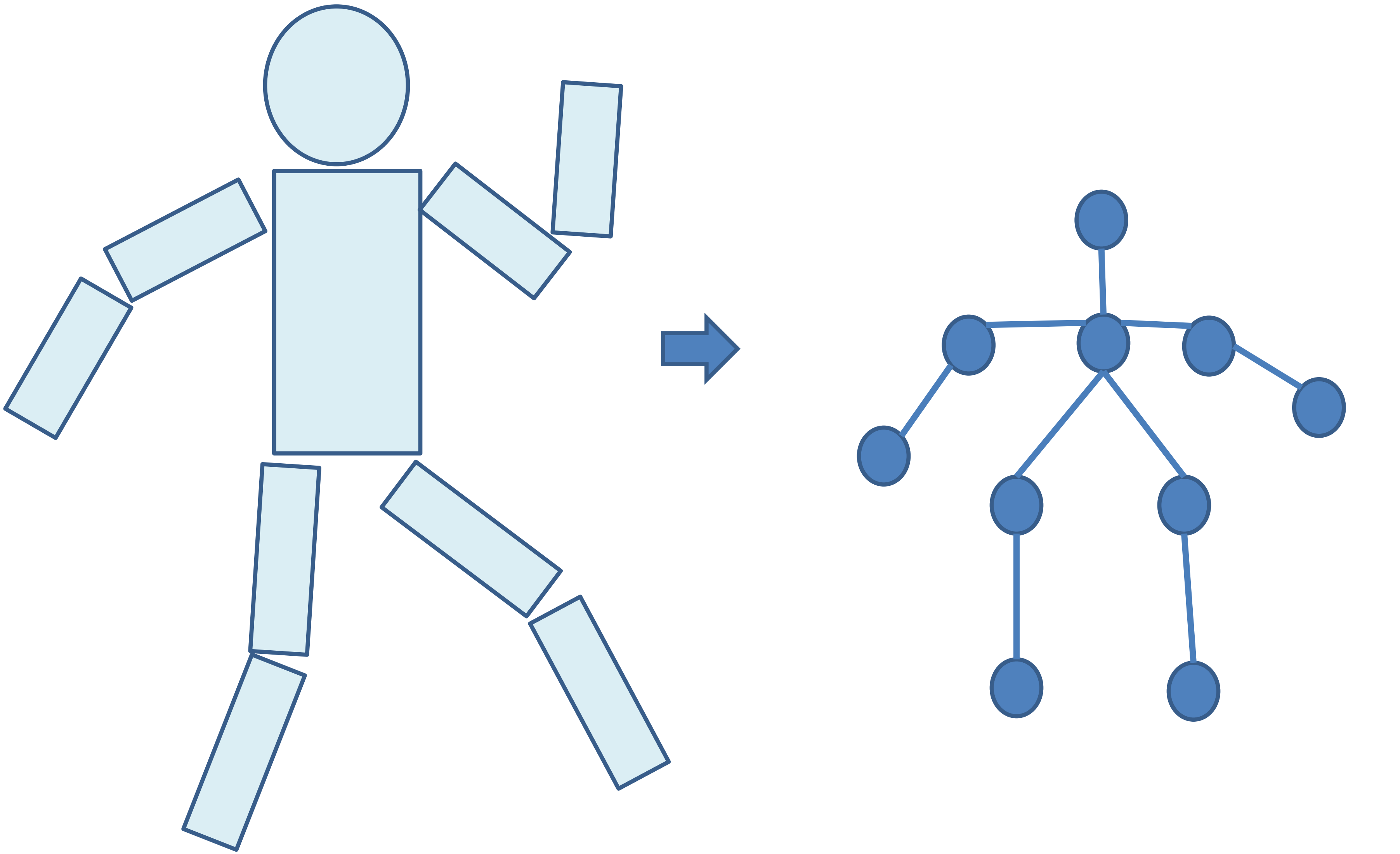


Images from D. Ramanan's dataset

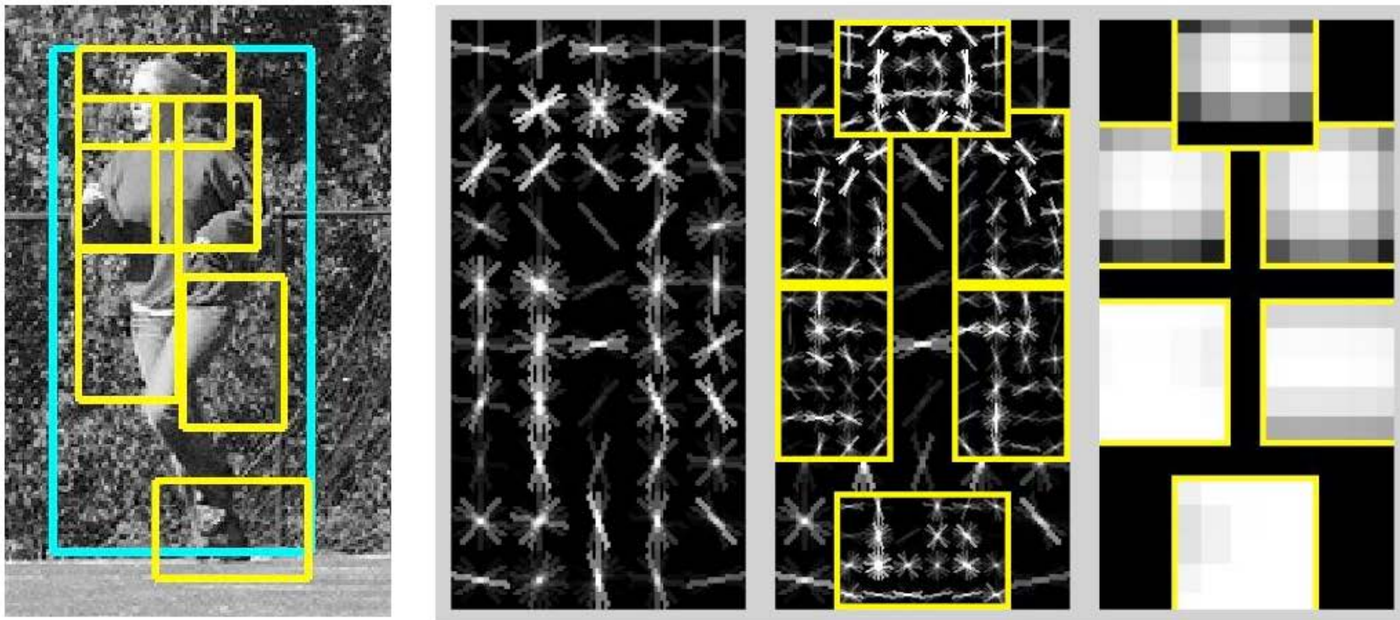
Slide Credit: Duan Tran

How to model spatial relations?

- Tree-shaped model



Model Overview

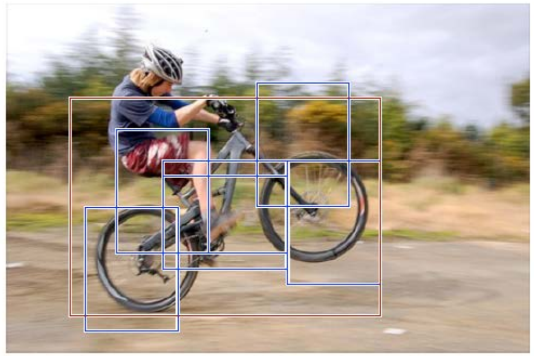


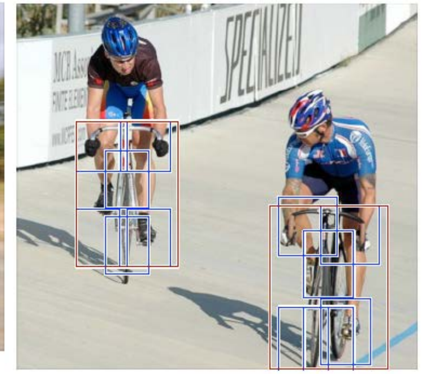
detection root filter part filters deformation models

Model has a root filter plus deformable parts

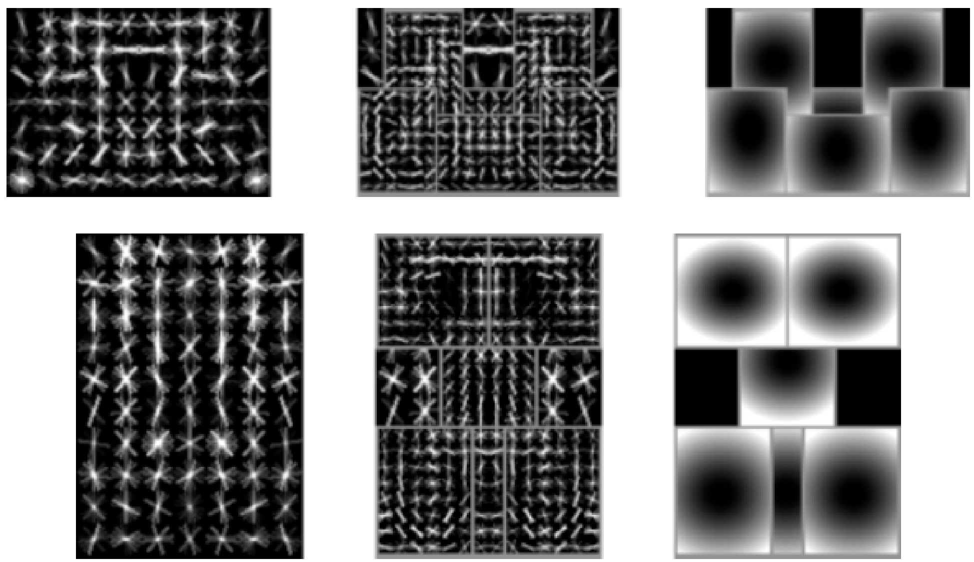
Hybrid template/parts model

Detections





Template Visualization



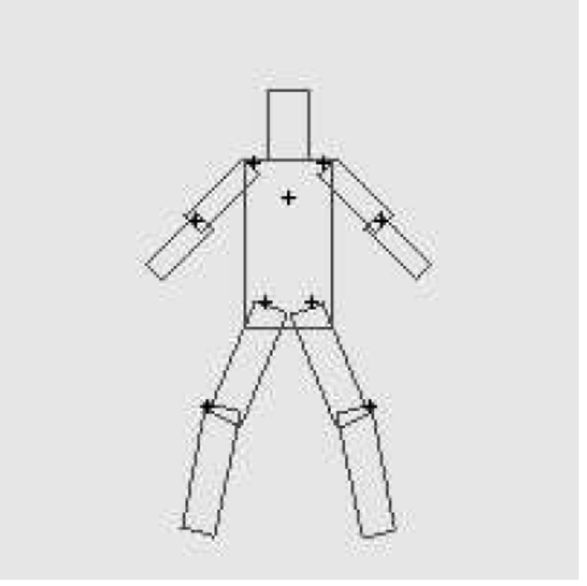
root filters
coarse resolution

part filters
finer resolution

deformation
models

Felzenszwalb et al. 2008

Pictorial Structures Model

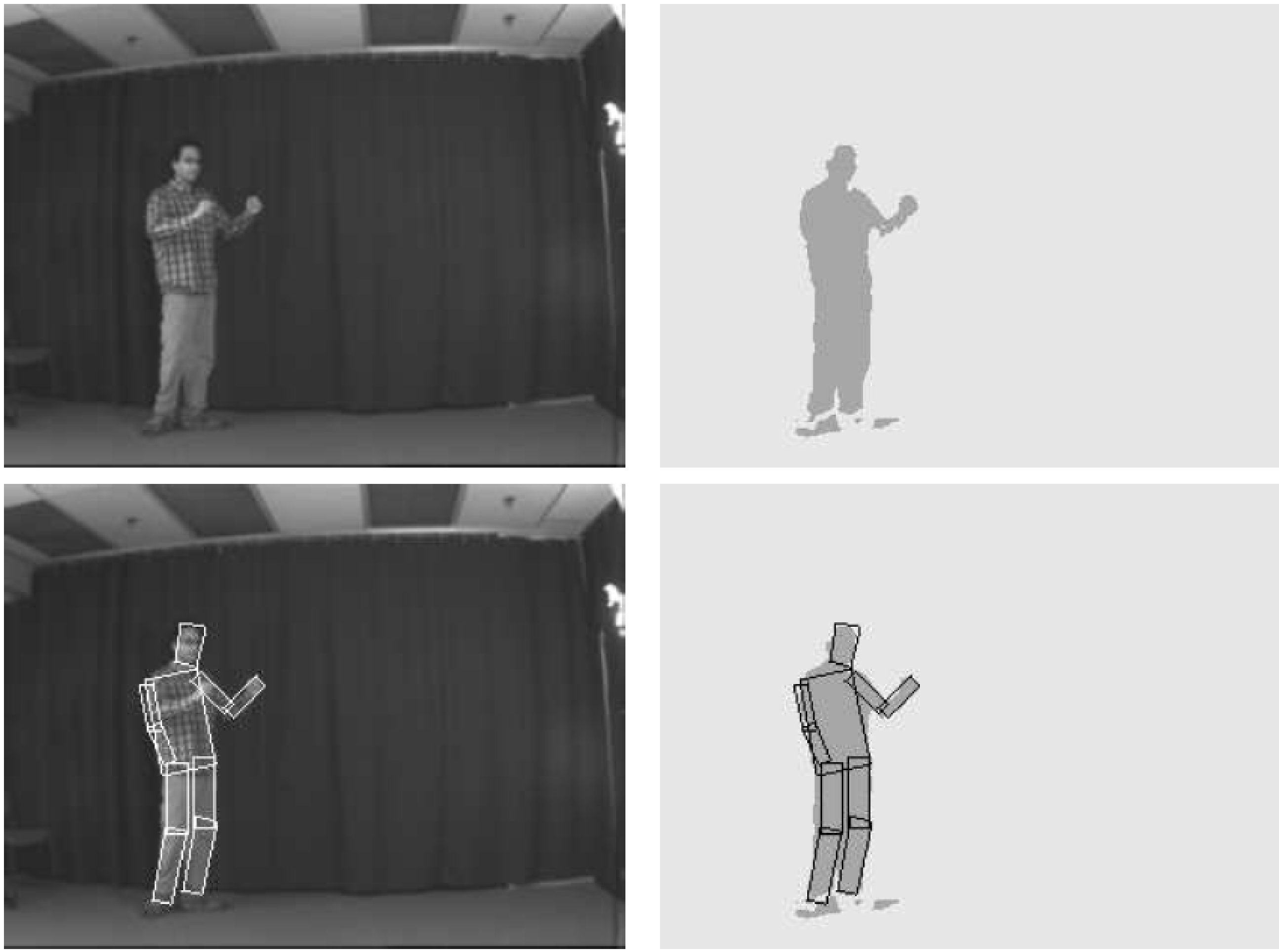


$$P(L|I,\theta) \propto \left(\prod_{i=1}^{n} p(I|l_i, u_i) \prod_{(v_i, v_j)\in E} p(l_i, l_j|c_{ij}) \right)$$

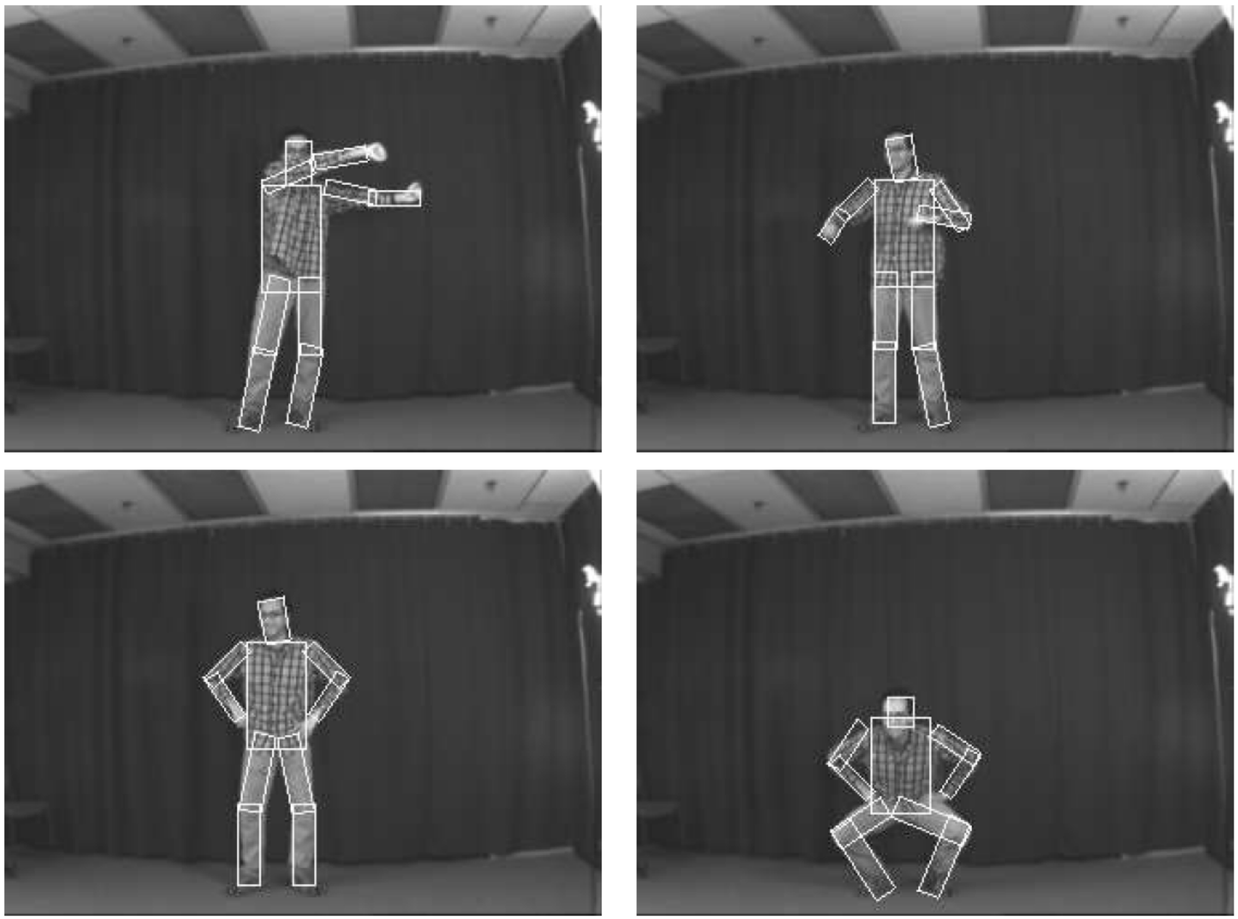
Appearance likelihood

Geometry likelihood

Results for person matching

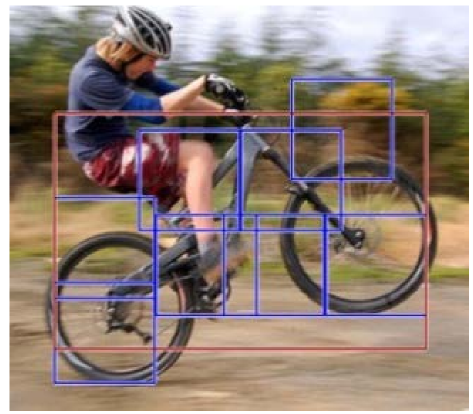


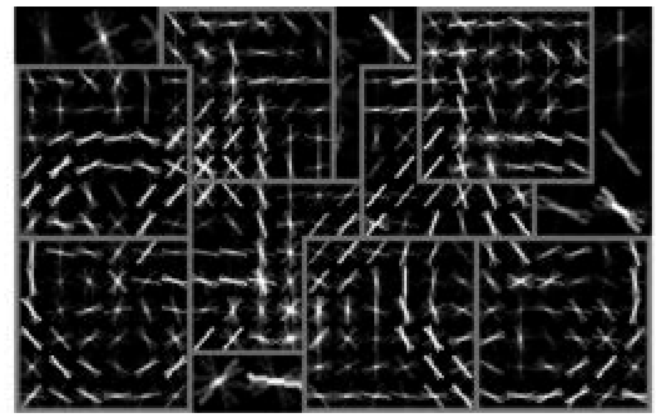
Results for person matching




1
2
3
4
5
6
a
b
c
d

2012 State-of-the-art Detector: Deformable Parts Model (DPM)





1. Strong low-level features based on HOG
2. Efficient matching algorithms for deformable part-based models (pictorial structures)
3. Discriminative learning with latent variables (latent SVM)

Felzenszwalb et al., 2008, 2010, 2011, 2012