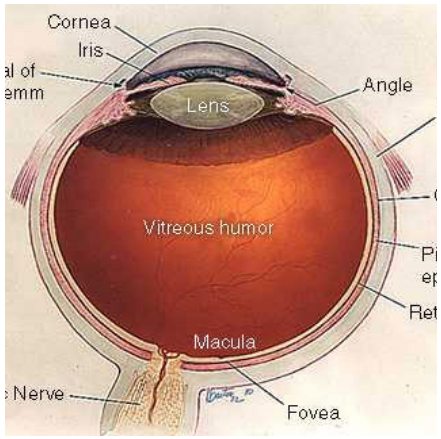
Features for Computer Vision

Alex Berg

Computer Science Department
Columbia University

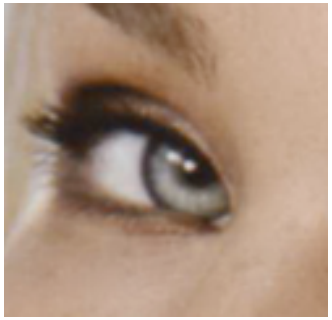
Why Vision? Light!

It is how we see other people, navigate our environment, communicate ideas, entertain, and **measure** the world around us.

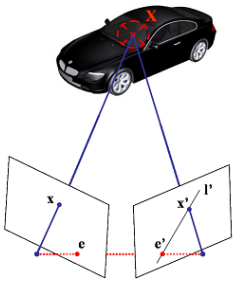


Why is light good for measurement?

Microscopy



Surveillance



3D Analysis / Navigation

Remote Sensing

- Plentiful, sometimes free
- Interacts with many things, but not too many
- Goes generally straight over distance
- Very small → high spatial resolution
- Easy to detect → cameras work, are cheap
- Fast, but not too fast → time of flight sensors
- Comes in flavors (wavelengths)
- We need to know **which bits** to measure!

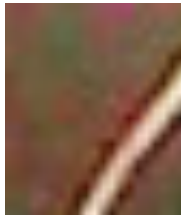
Recognition in Computer Vision

Deciding which bits to measure…

Range of Difficulty in Recognition

Easy

vs



or

Range of Difficulty in Recognition

Easy

vs

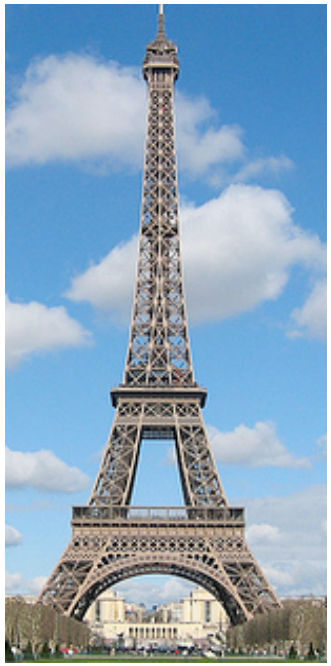
or

Recognizing the same image

Range of Difficulty in Recognition

Medium

Small change



vs

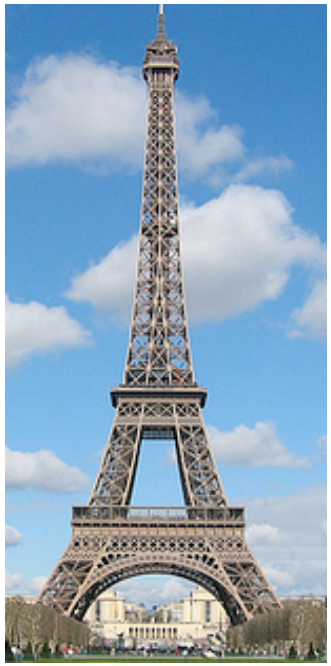
or

Recognizing the same object

Range of Difficulty in Recognition

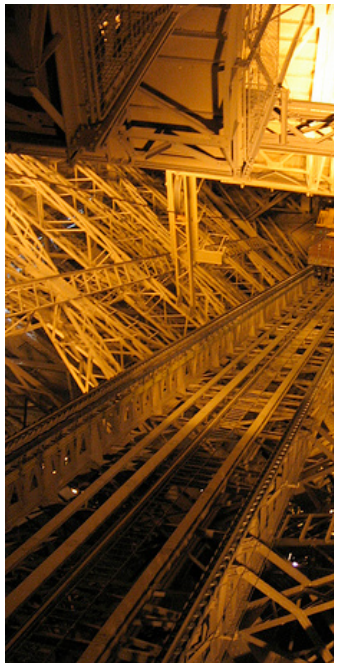
Medium

Large change



vs

or



Recognizing the same object

Range of Difficulty in Recognition

Medium to Difficult

vs

or

Recognizing the same object

Range of Difficulty in Recognition

Difficult

Chair vs

Recognizing the same object category

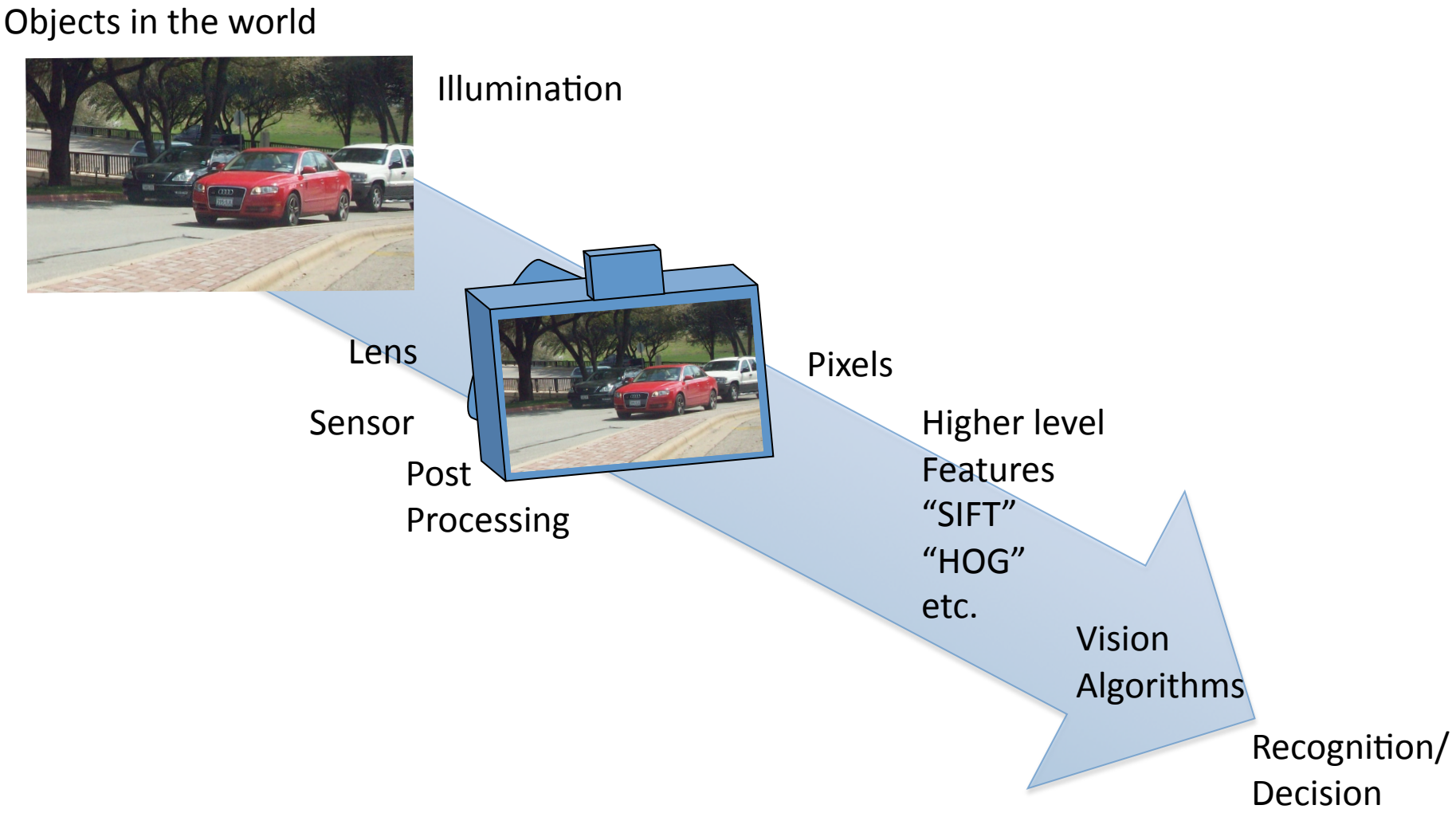
Range of Difficulty in Recognition

Difficult

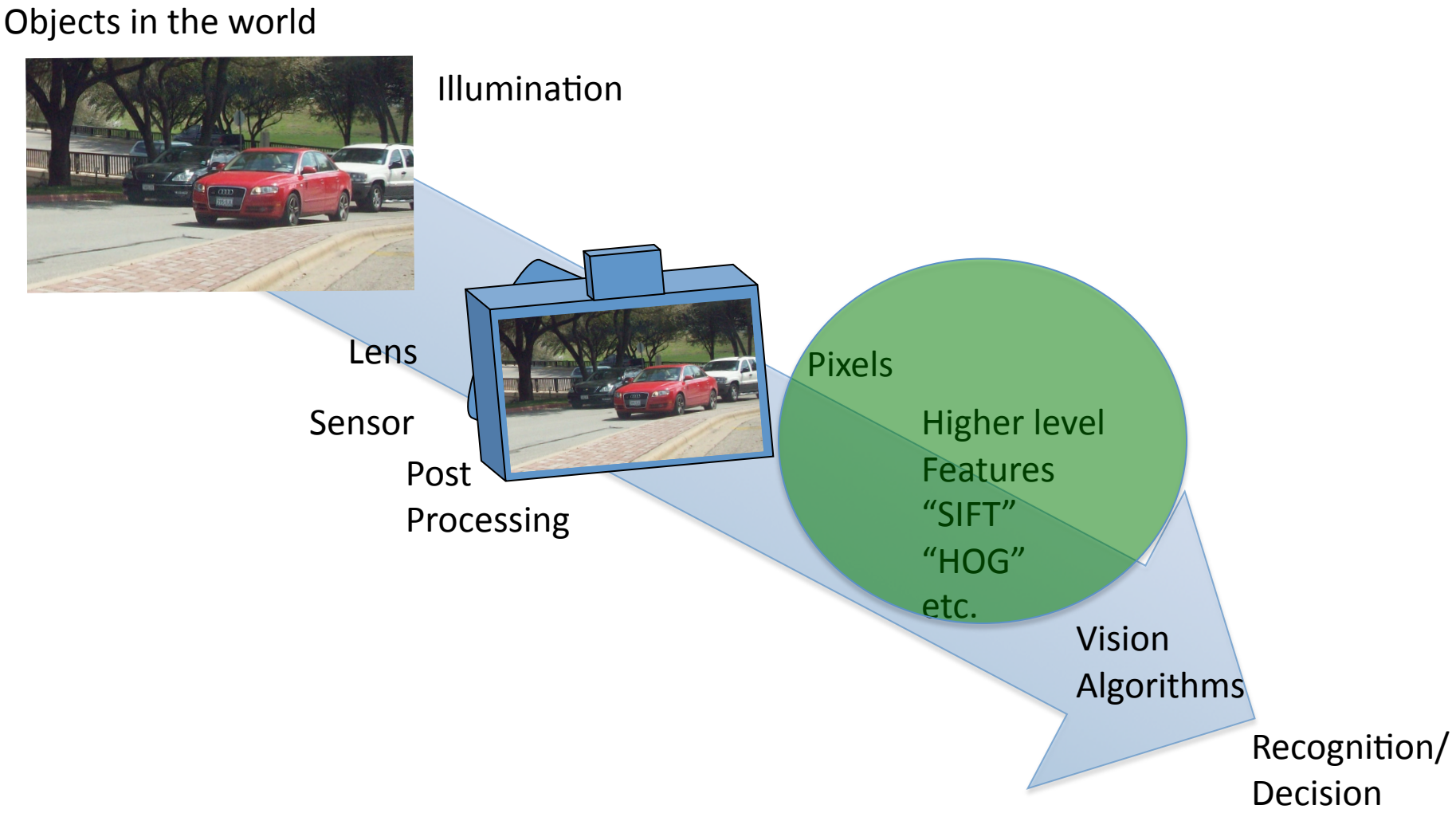
vs

Recognizing the same object category

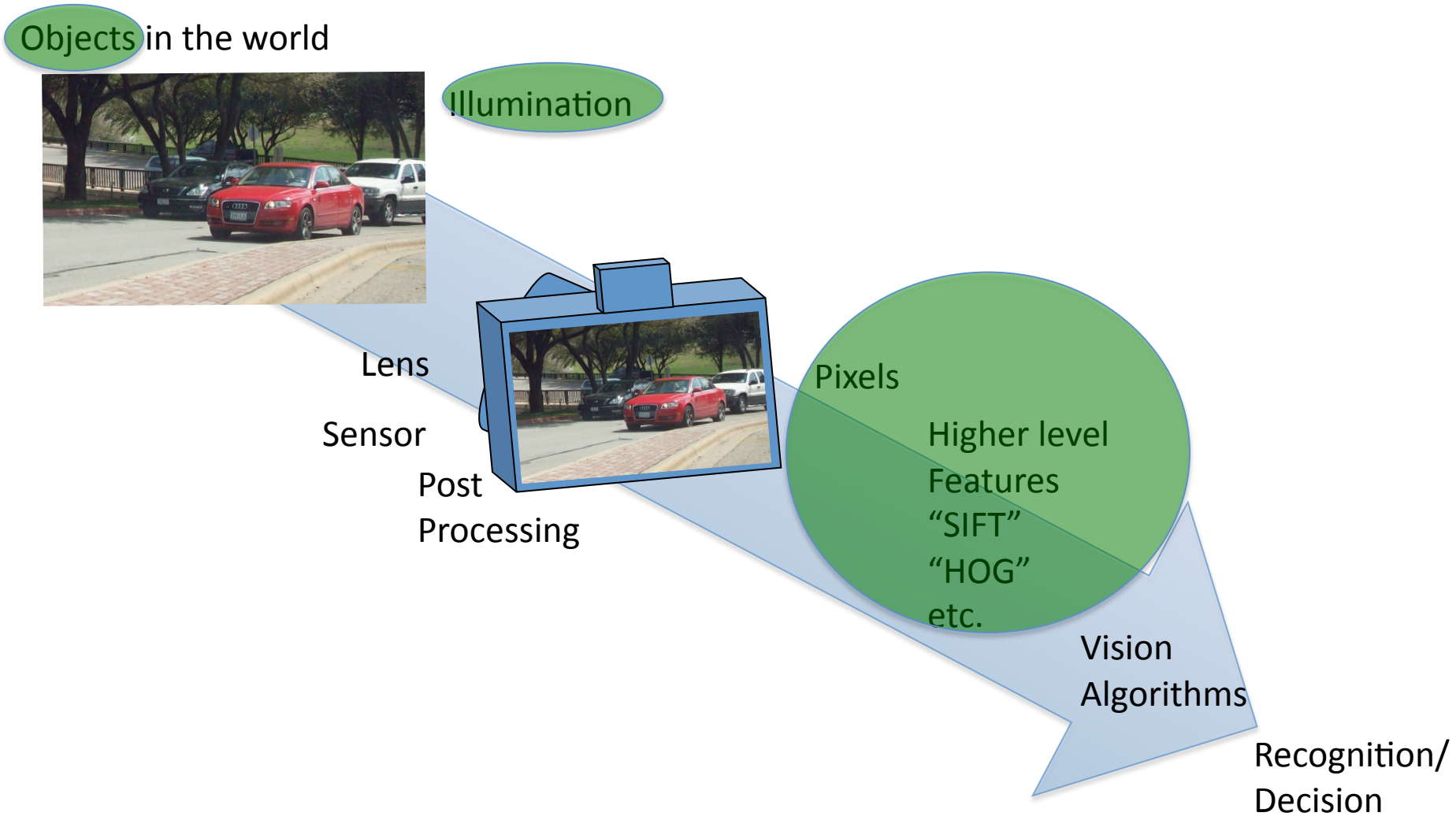
Where do Features Fit?



Where do Features Fit?



Where do Features Fit?



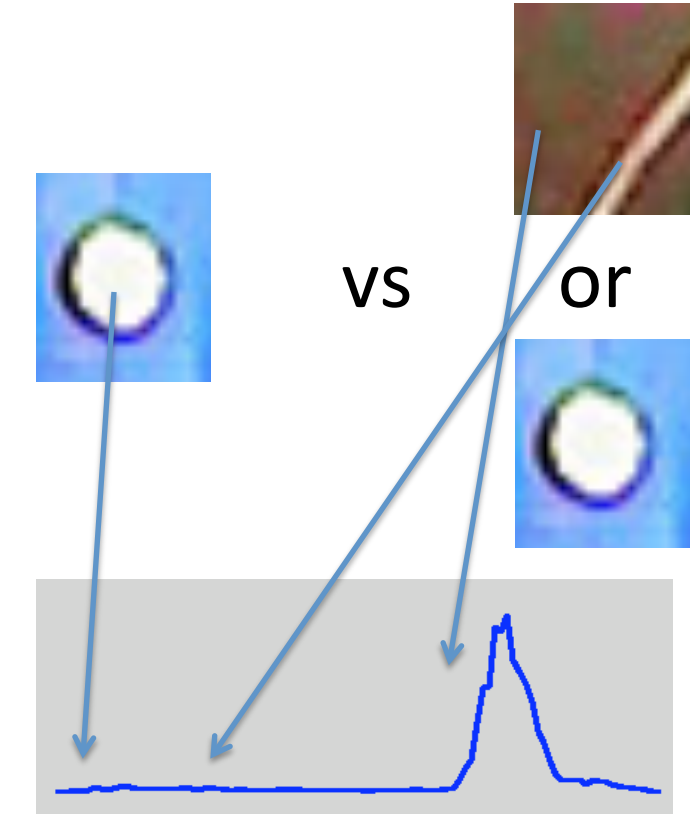
Control objects and lighting
(if possible)

Retroreflective Balls

Illumination from
near the cameras

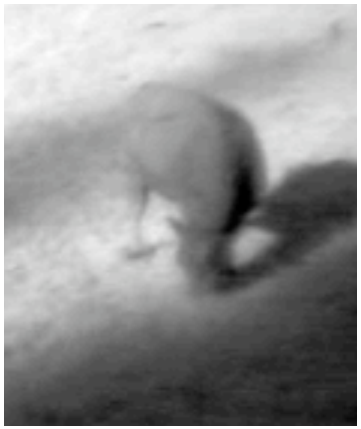
Here raw pixels are almost enough…

Easy



Multiple nearby pixels in a circle agreeing probably suffice.

Cows May Be Less Cooperative



Cows come in many brightnesses as does the background

Looking at shape

Similar shapes have very different pixel values

Looking at shape

Similar shapes have very different pixel values

Vision is difficult

Different images of the same thing often look different.

Sometimes images of different things look the same.

Despite all its useful qualities light only tells us about objects indirectly…

The usual suspects for problems are:

Pose
Illumination
Articulation
Intra-category variation

When are raw pixels enough?

Given *sufficient training data* and a *powerful classifier*
patches or windows of pixels would be enough – we wouldn't need any high level features.

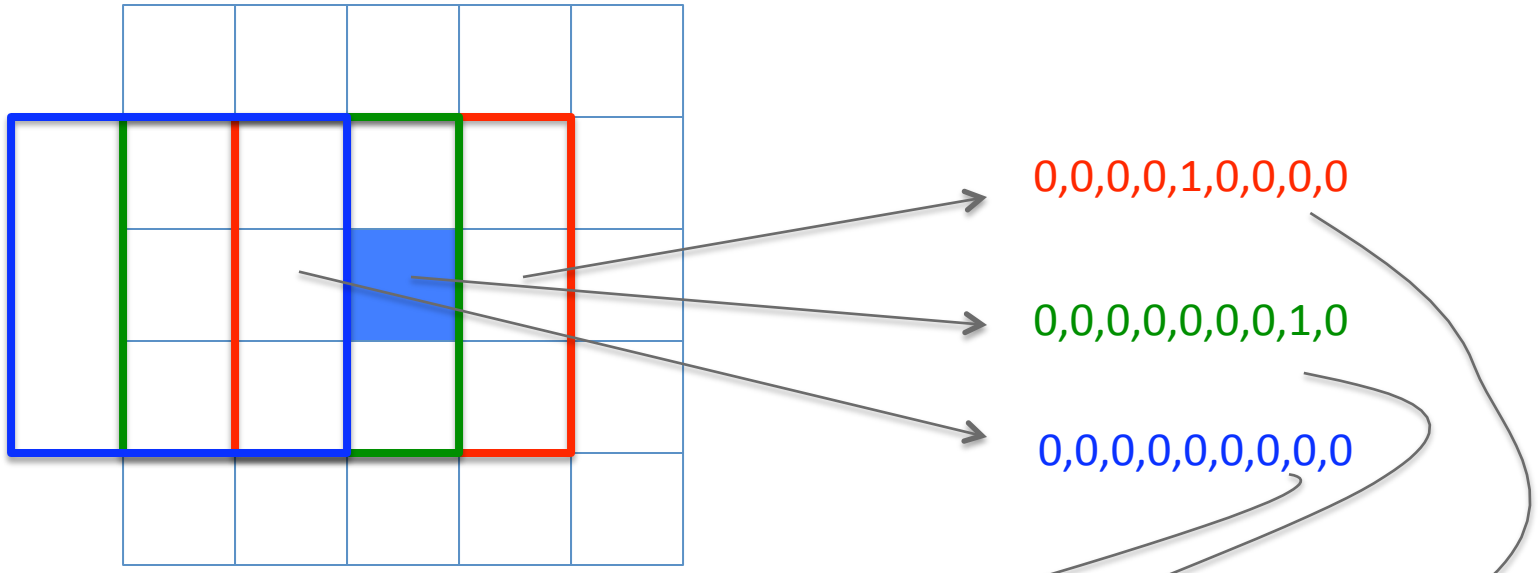
For a toy 10x10 pixel image with 10 brightness levels there are

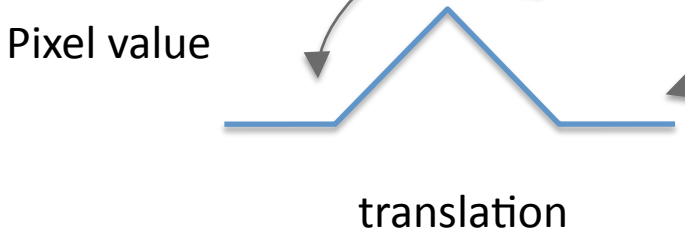
10^100 possibilities, you might imagine labeling all of them as face or not…

There is almost never enough training data for this approach,

Exceptions are when we can enumerate the positive examples.

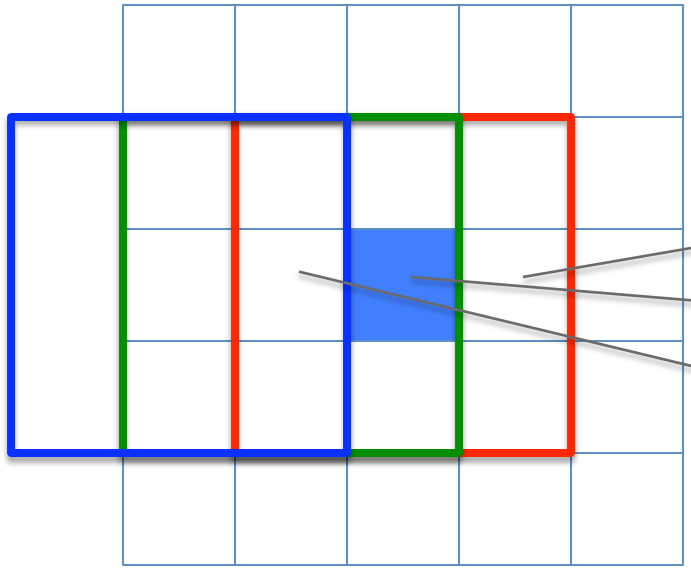
Simple example with translation





If not careful, this isn't even differentiable.
Need to be careful about smoothing and representation

Simple example with translation

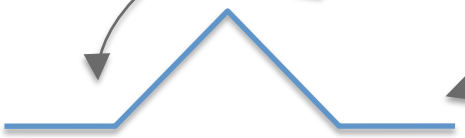


0,0,0,0,1,0,0,0,0

0,0,0,0,0,0,0,1,0

0,0,0,0,0,0,0,0,0

Of course we can get around simple translation by evaluating comparisons everywhere



Pixel value

translation

If not careful, this isn't even differentiable. Need to be careful about smoothing and representation

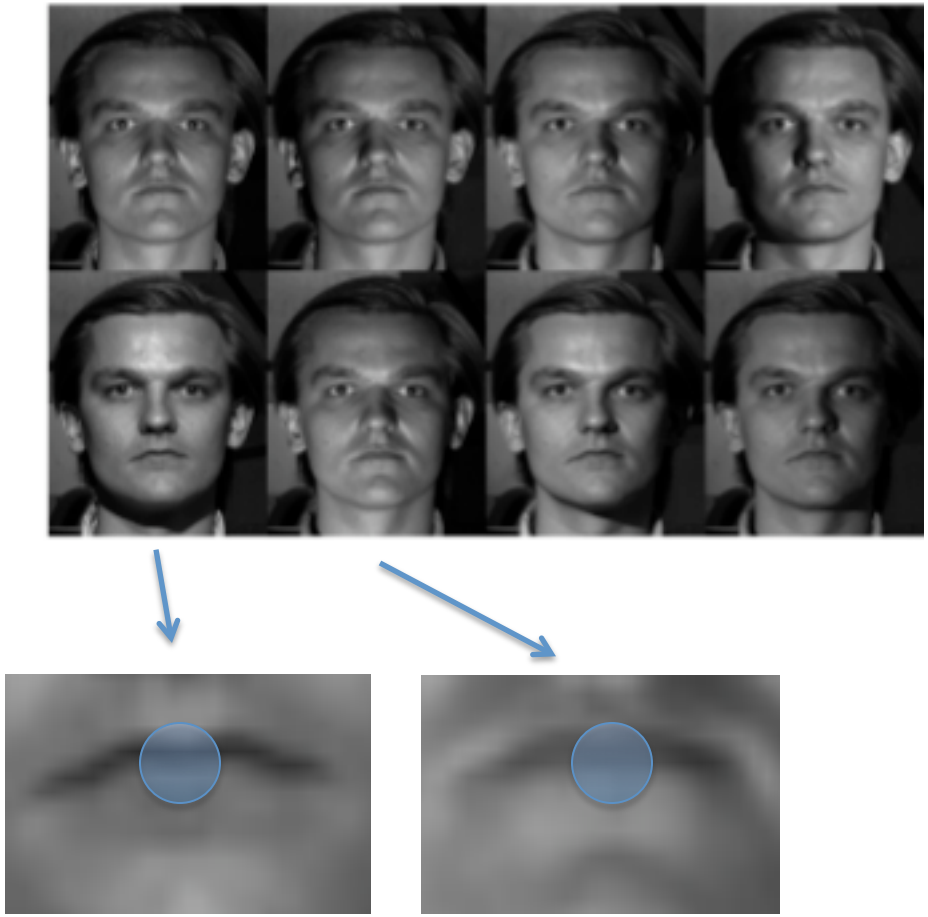
Can look at features as trying to straighten out appearance manifolds

Important to keep track of what you are throwing away

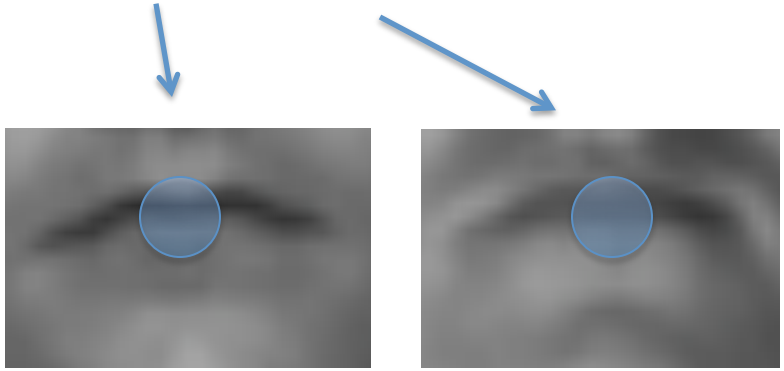
One attempt: edges

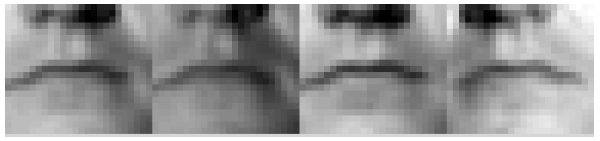
One attempt: edges

One attempt: edges



One attempt: edges



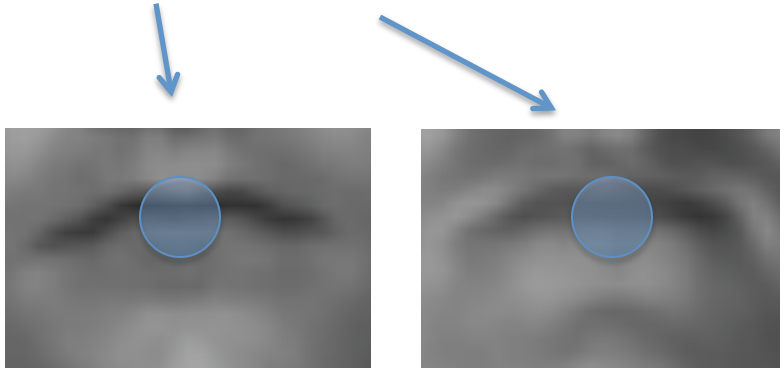


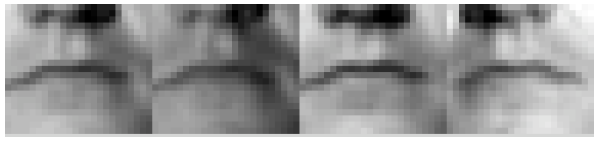
Image

Edge Energy

Orientation

One attempt: edges





Image

Edge Energy

Orientation

works because it indicates something physical about the object that is conserved across images

Edges help deal with variation in illumination

Illumination fields are often soft, so sharp changes may indicate something about the object not the illumination. Orientation and phase often preserved under lighting variations.

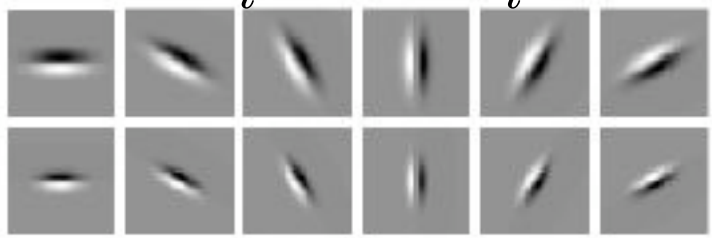
Possible sources of edges:
-Albedo variations on the object surface
-Surface structure on the object (changing surface normal, creases, holes)
-Boundaries of the object

Work from Simoncelli et al on the importance of orientation / phase for human perception.

Versions of edges

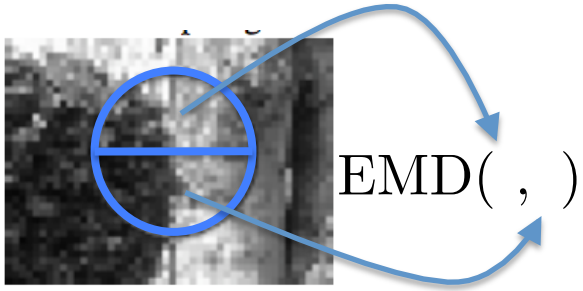
Brightness gradients, Haar wavelets

$$r_i = I * k_i$$

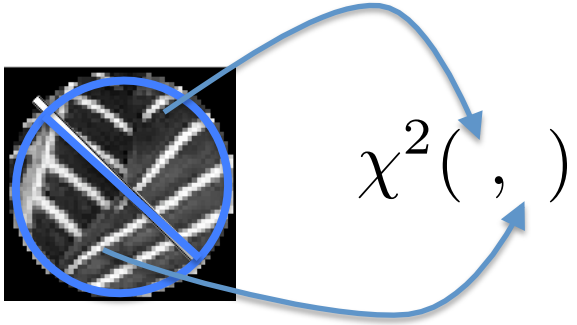


Multiple scales,
elongated filters

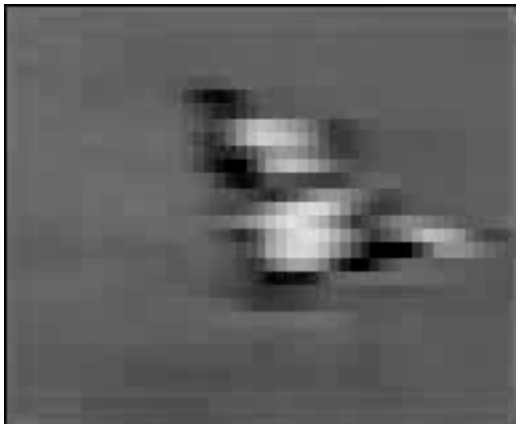
Color compass edges Ruzon & Tomasi



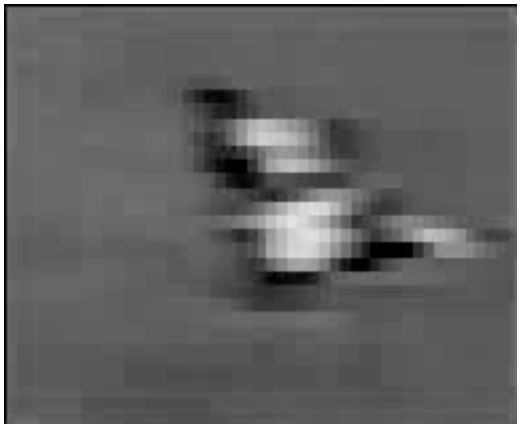
Texture (“compass”) Martin, Fowlkes, Malik

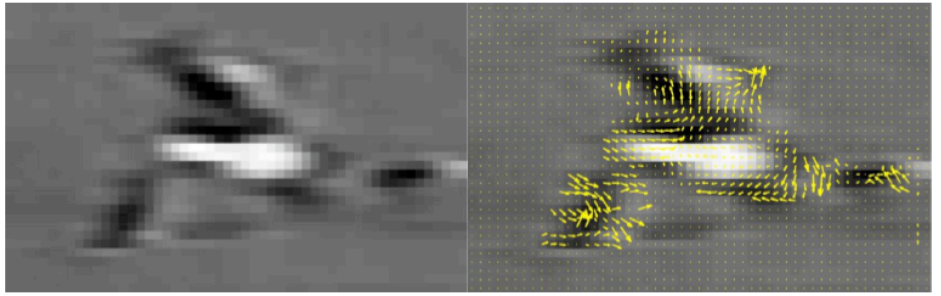


Is it the same? **What?**



Is it the same? **Optical Flow**





(a) original image (b) optical flow $F_{x,y}$

(c) F_x, F_y (d) $F_x^+, F_x^-, F_y^+, F_y^-$ (e) $Fb_x^+, Fb_x^-, Fb_y^+, Fb_y^-$

Color and texture do not match.
Too low resolution to extract edges.
Coarse optical flow (spatio-temporal gradient direction) features match.

Efros, Berg, Mori, Malik ICCV 2003

Review

Image -> feature $I(x,y) \rightarrow F(I(x,y))$

Oriented edge detection may be helpful

Transformed signals $I(T(x,y)) \neq I(x,y)$

Review

Image -> feature $I(x,y) \rightarrow F(I(x,y))$

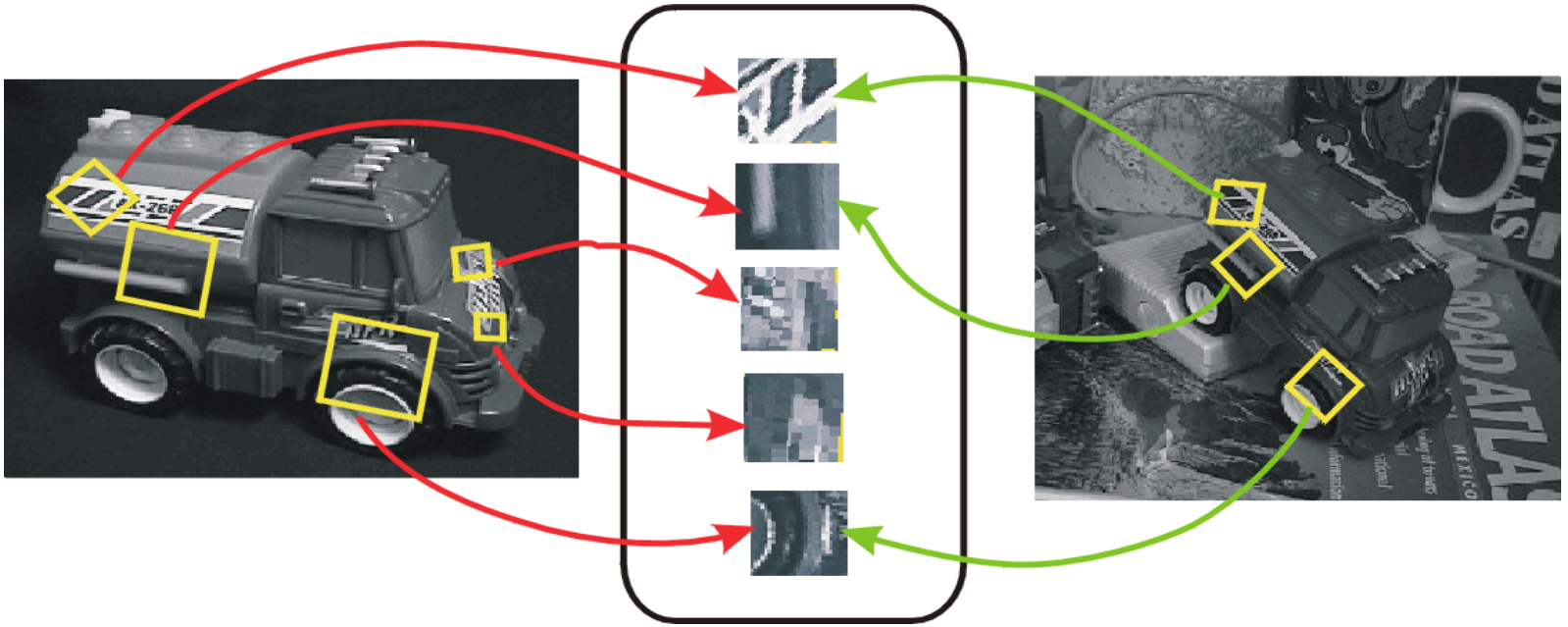
Oriented edge detection may be helpful

Transformed signals $I(T(x,y)) \neq I(x,y)$

Region of interest operators

“Histograms”

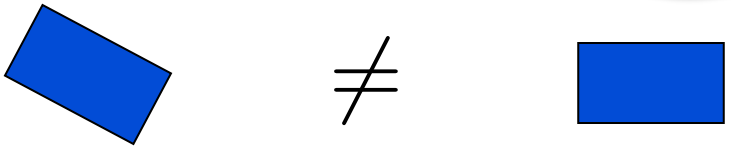
Region of interest operators



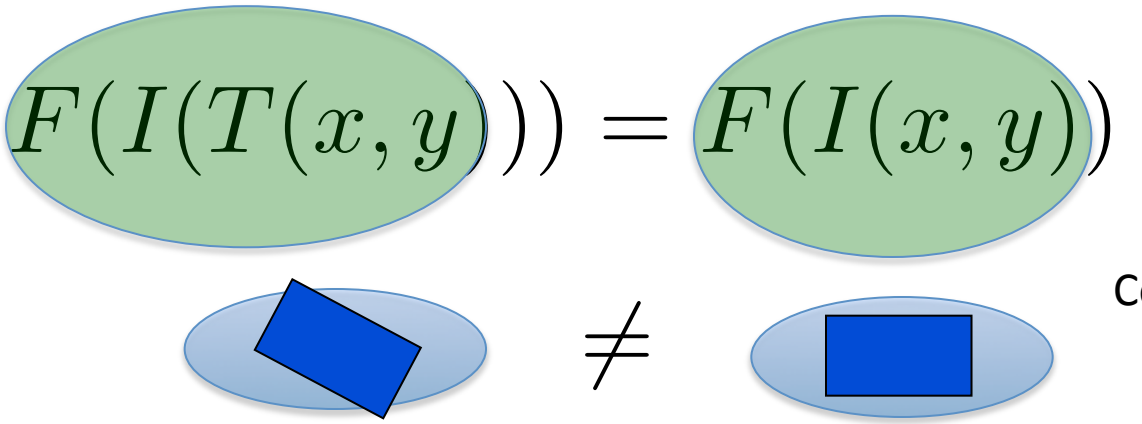
Slide from Lazebnik

Regions of Interest Invariance

$$F(I(T(x,y))) = F(I(x,y))$$



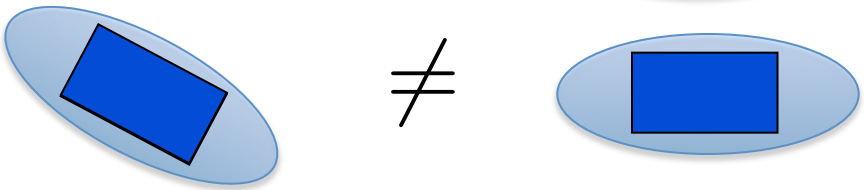
Regions of Interest
Invariance



Compute features in light blue region

Regions of Interest
Invariance

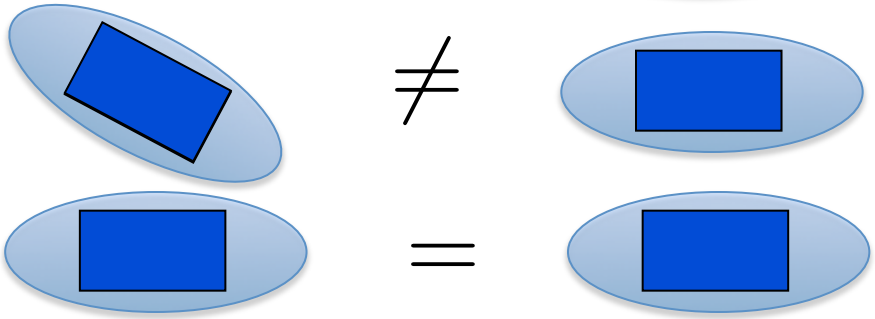
$$F(I(T(x,y))) = F(I(x,y))$$



Adapt region to image content (boxes)
Compute features

Regions of Interest
Invariance

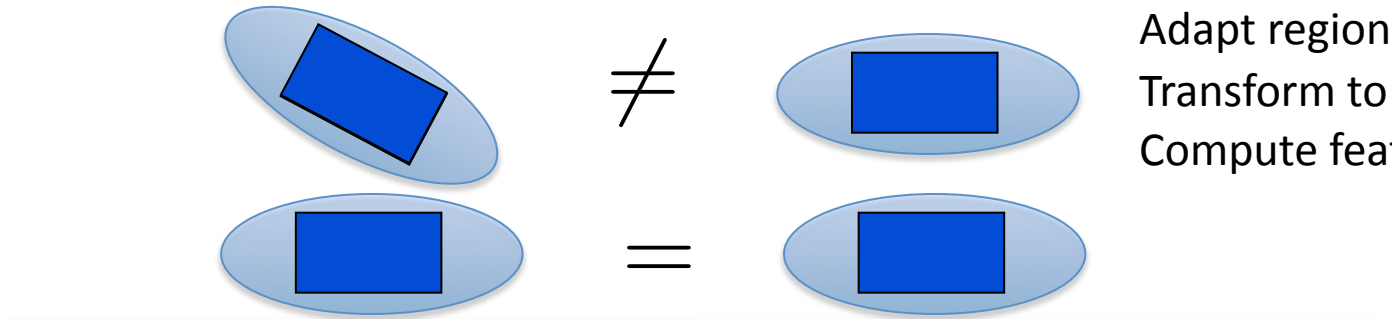
$$F(I(T(x,y))) = F(I(x,y))$$



Adapt region to image content (boxes)
Transform to canonical pose
Compute features

Regions of Interest
Invariance & Co-Variance

$$F(I(T(x,y))) = F(I(x,y))$$

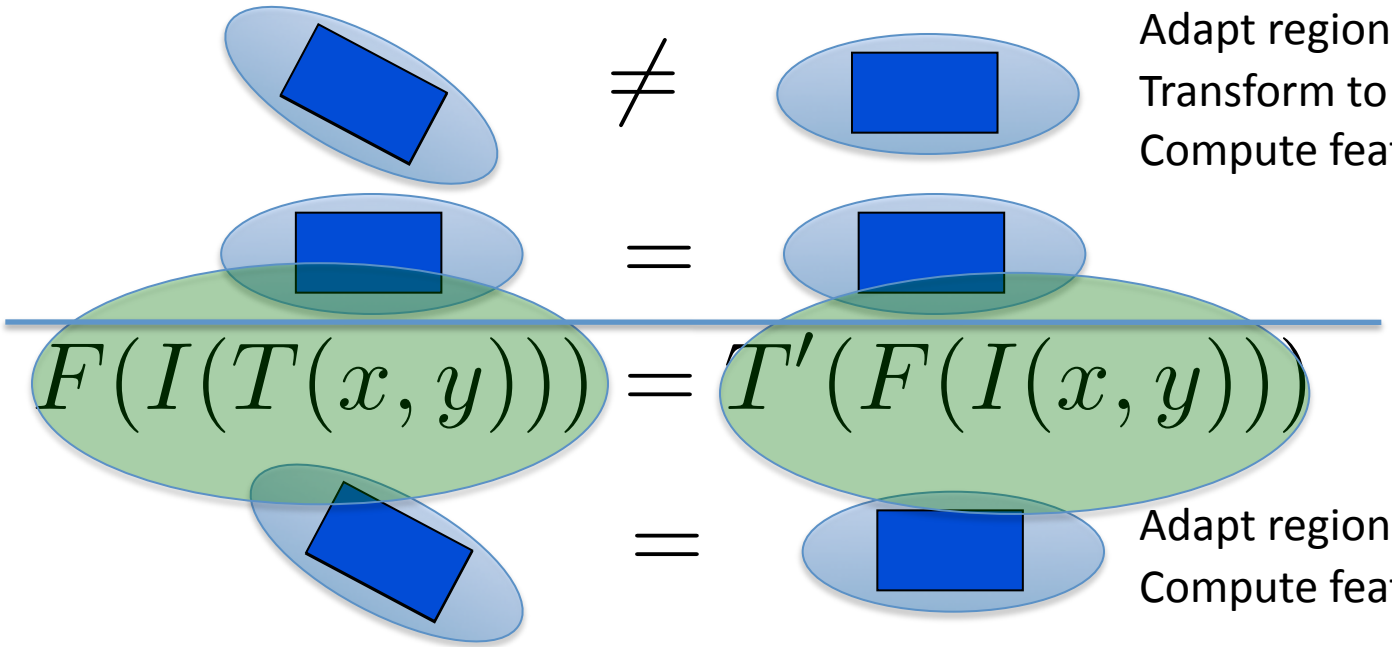


$$F(I(T(x,y))) = T'(F(I(x,y)))$$

Adapt region to image content (boxes)
Transform to canonical pose
Compute features

Regions of Interest
Invariance & Co-Variance

$$F(I(T(x,y))) = F(I(x,y))$$

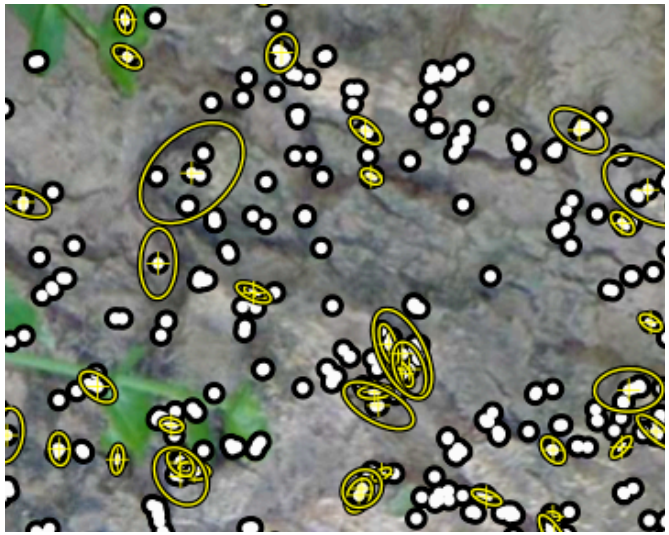
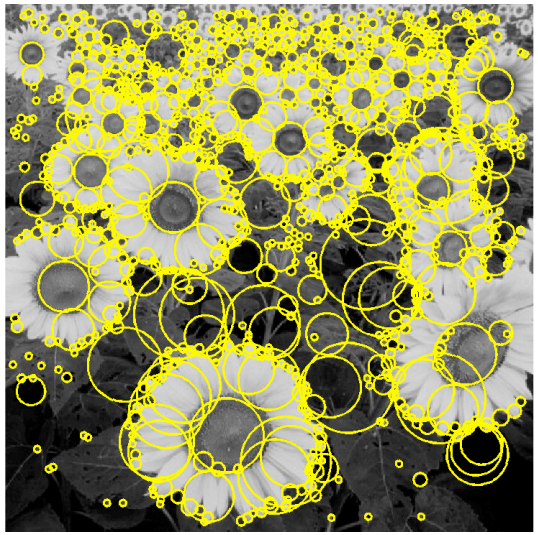
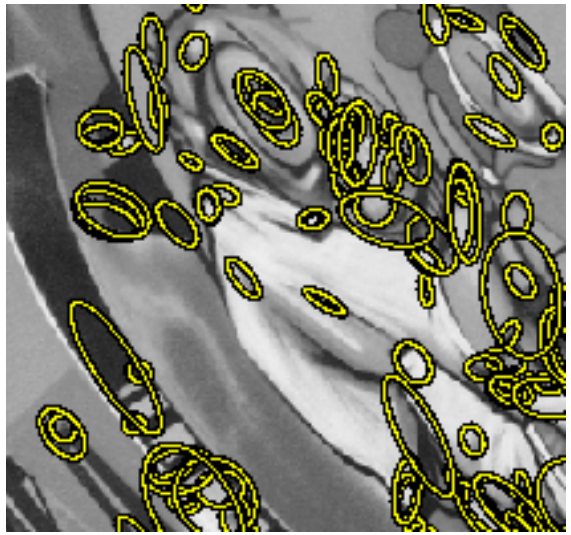


Adapt region to image content (boxes)
Transform to canonical pose
Compute features

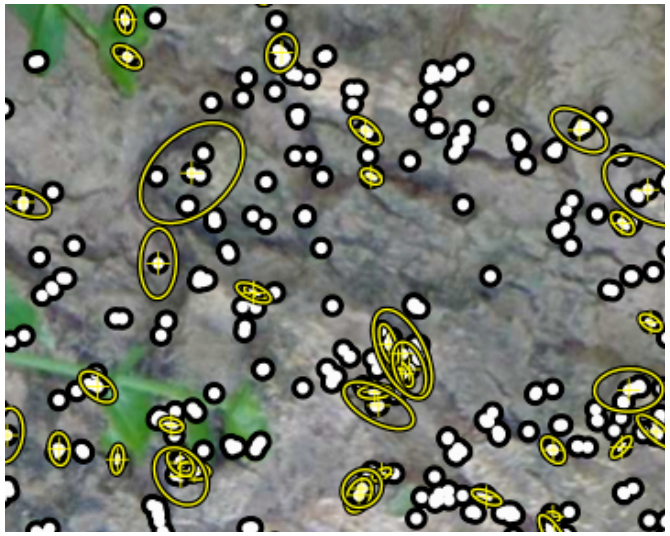
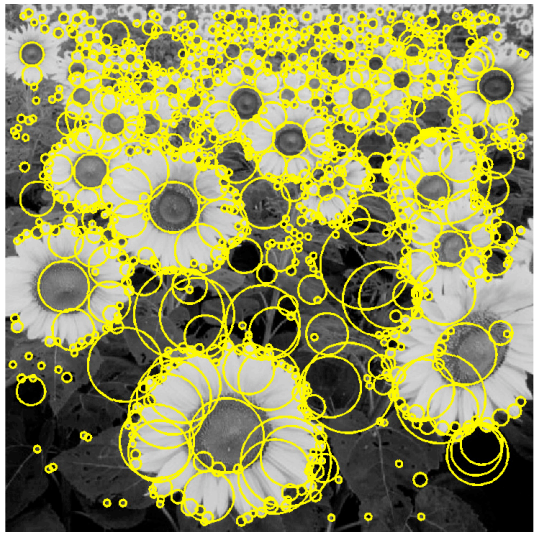
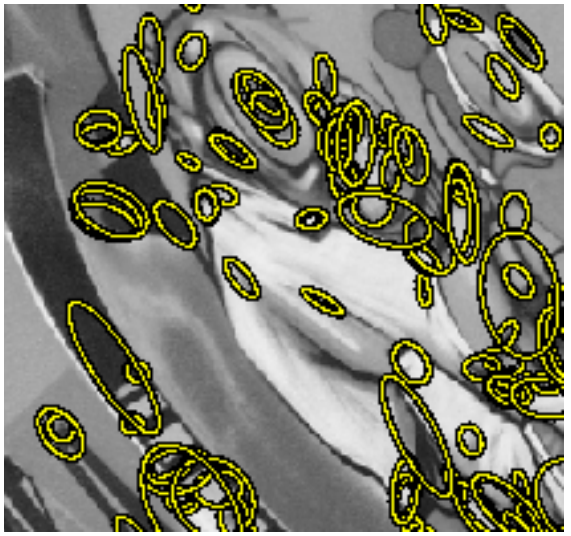
Adapt region to image content (boxes)
Compute features

Schmid & Mohr, Lowe

Region of Interest Operators

Region of Interest Operators

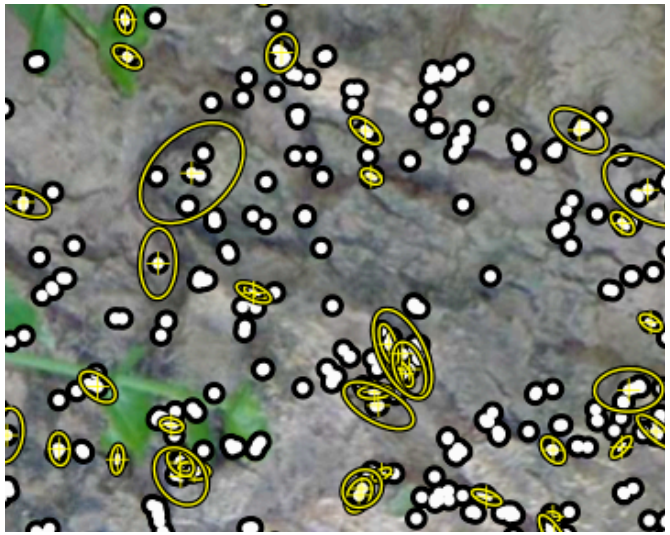
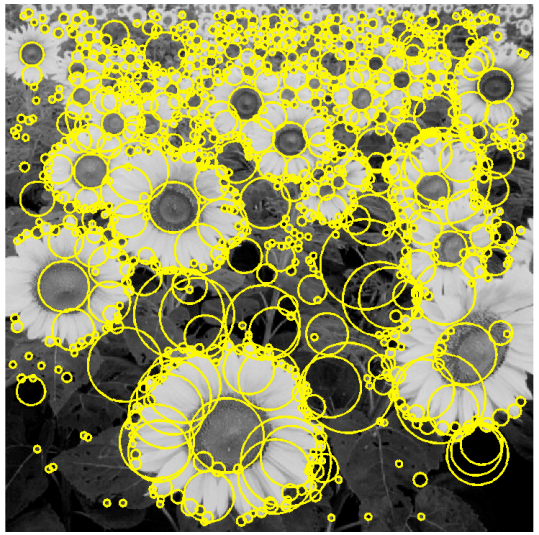
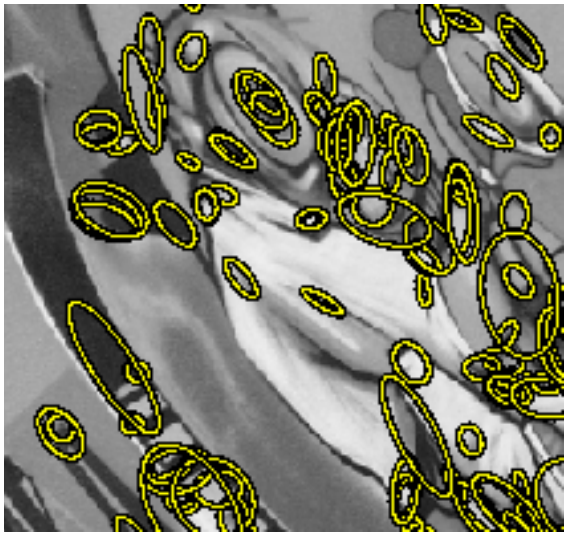




$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y)$$

$$\begin{aligned} D(x, y, \sigma) &= (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) \\ &= L(x, y, k\sigma) - L(x, y, \sigma). \end{aligned}$$

Look for local maxima

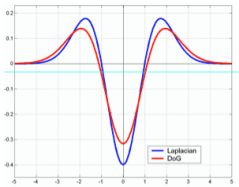
Region of Interest Operators

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y)$$

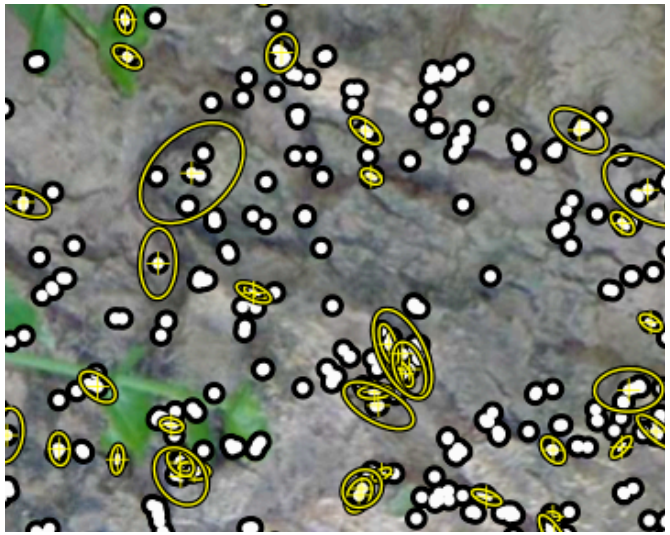
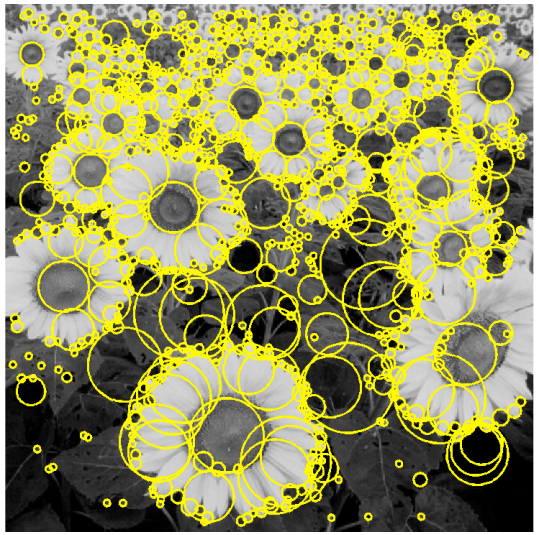
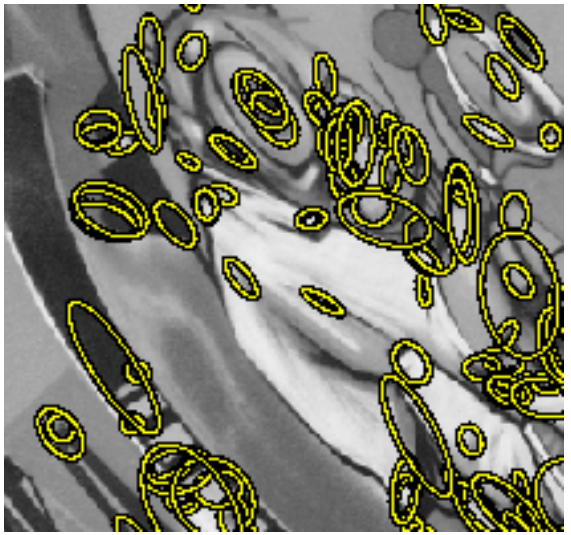
$$\begin{aligned} D(x, y, \sigma) &= (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) \\ &= L(x, y, k\sigma) - L(x, y, \sigma). \end{aligned}$$

Cross section looks like ->

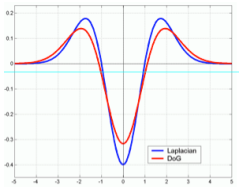


Look for local maxima, blobs

Region of Interest Operators

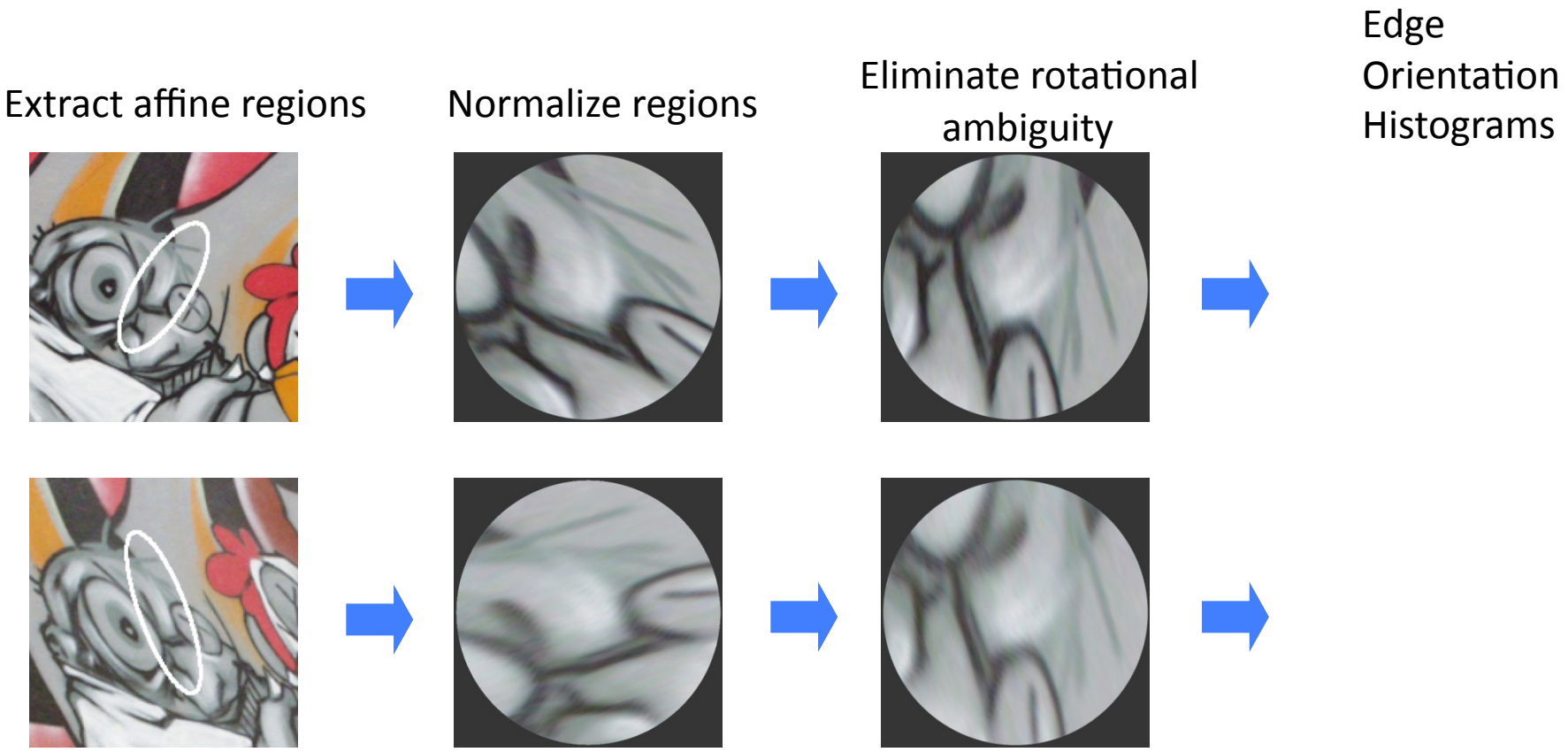
Cross section looks like ->



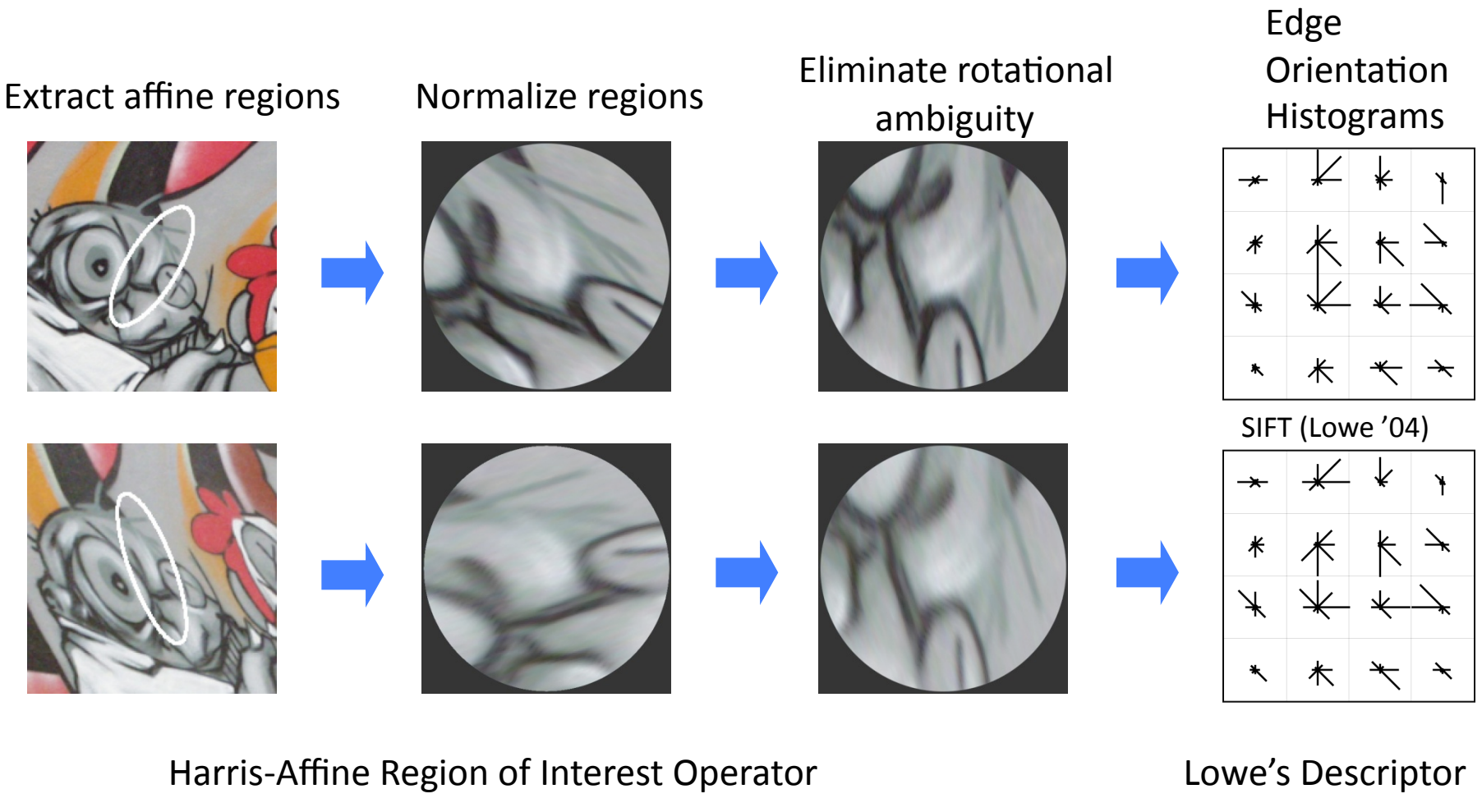
$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y)$$

$$\begin{aligned} D(x, y, \sigma) &= (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) \\ &= L(x, y, k\sigma) - L(x, y, \sigma). \end{aligned}$$

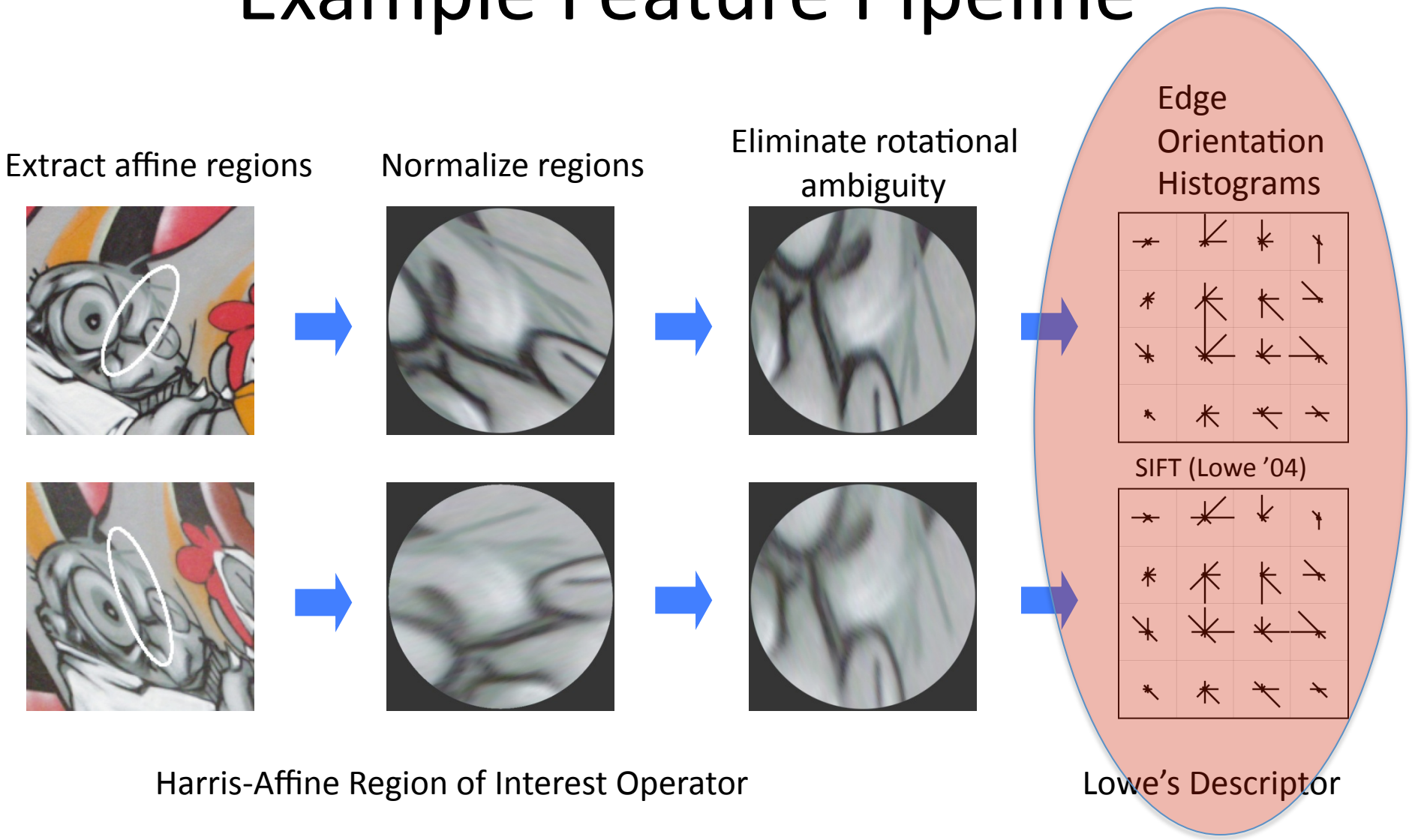
Example Feature Pipeline



Example Feature Pipeline

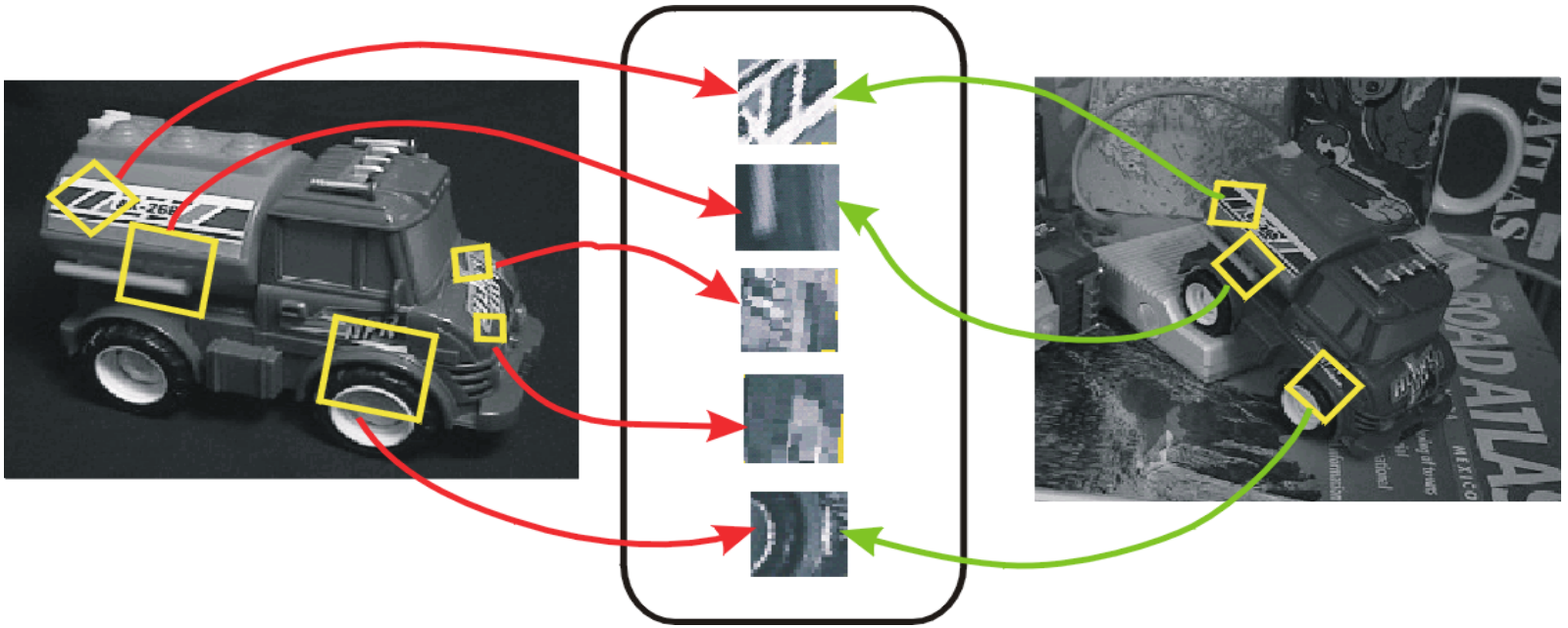


Example Feature Pipeline

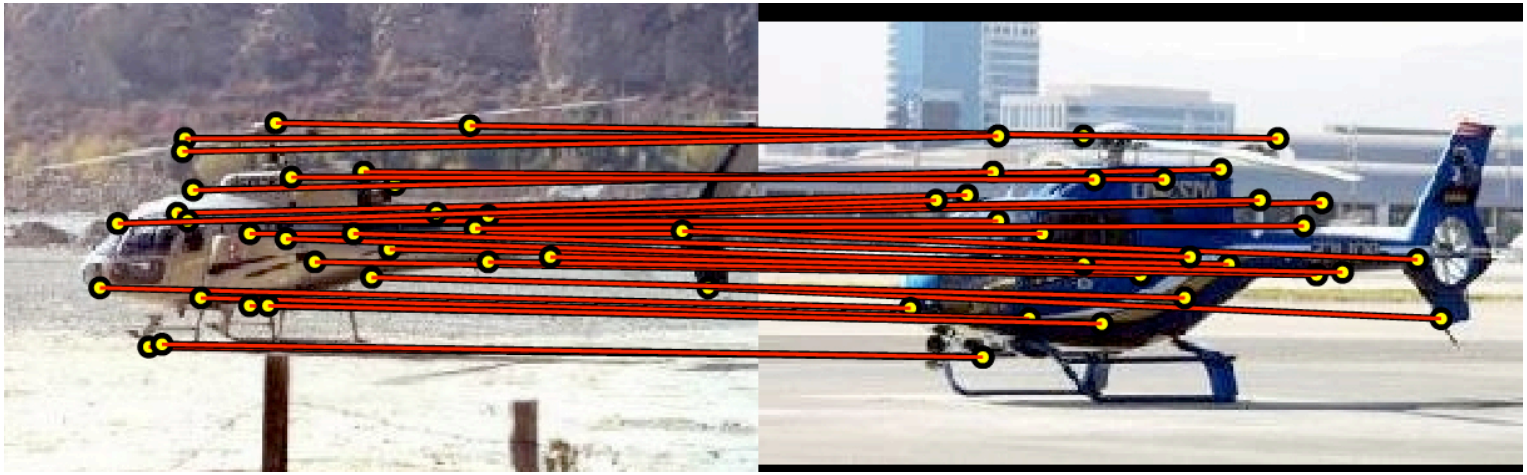


Matching for Alignment

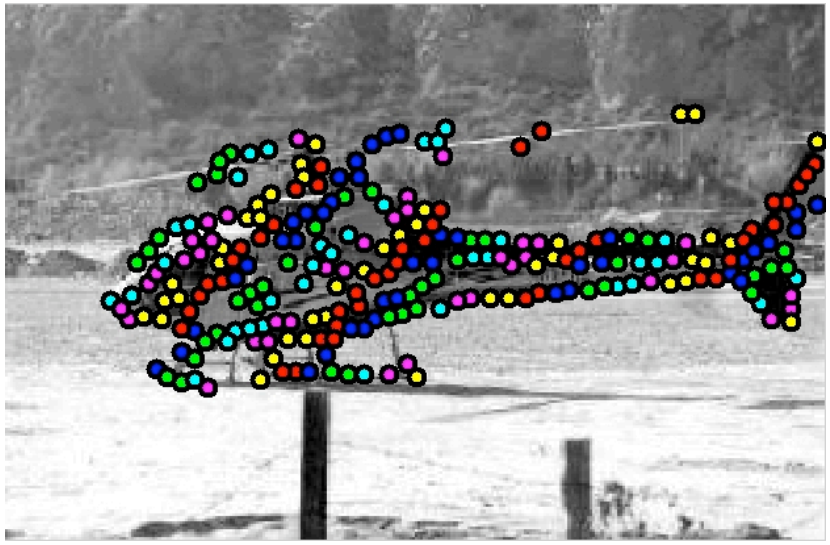
Use descriptors to compare features and enforce geometric constraints



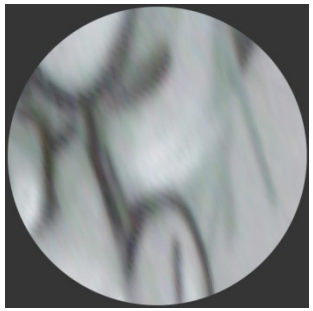
Match a few points

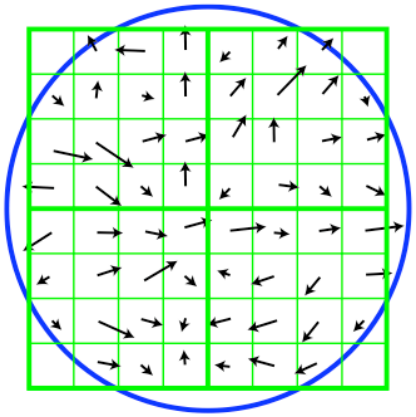


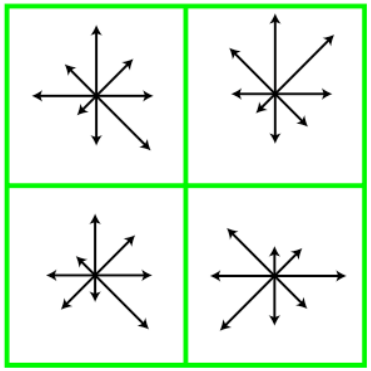
Dense Alignment



Sift 2



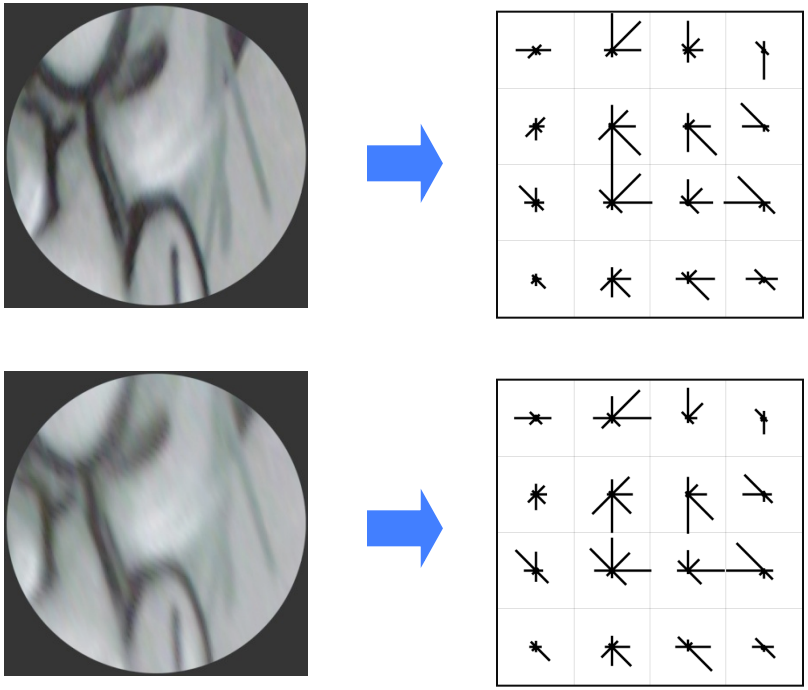




Example Feature Pipeline

Eliminate rotational ambiguity

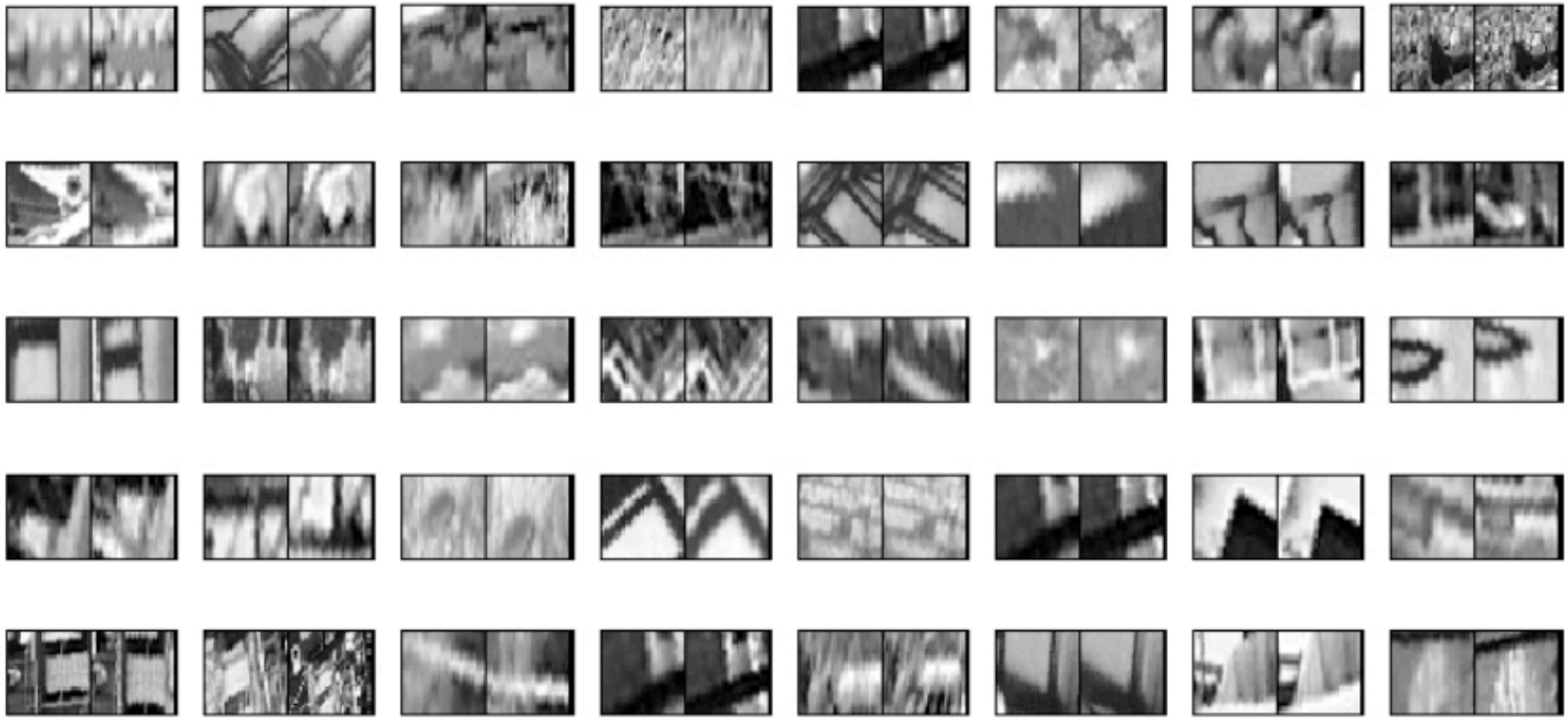
Edge Orientation Histograms



Remaining variation here

Needs to be handled here

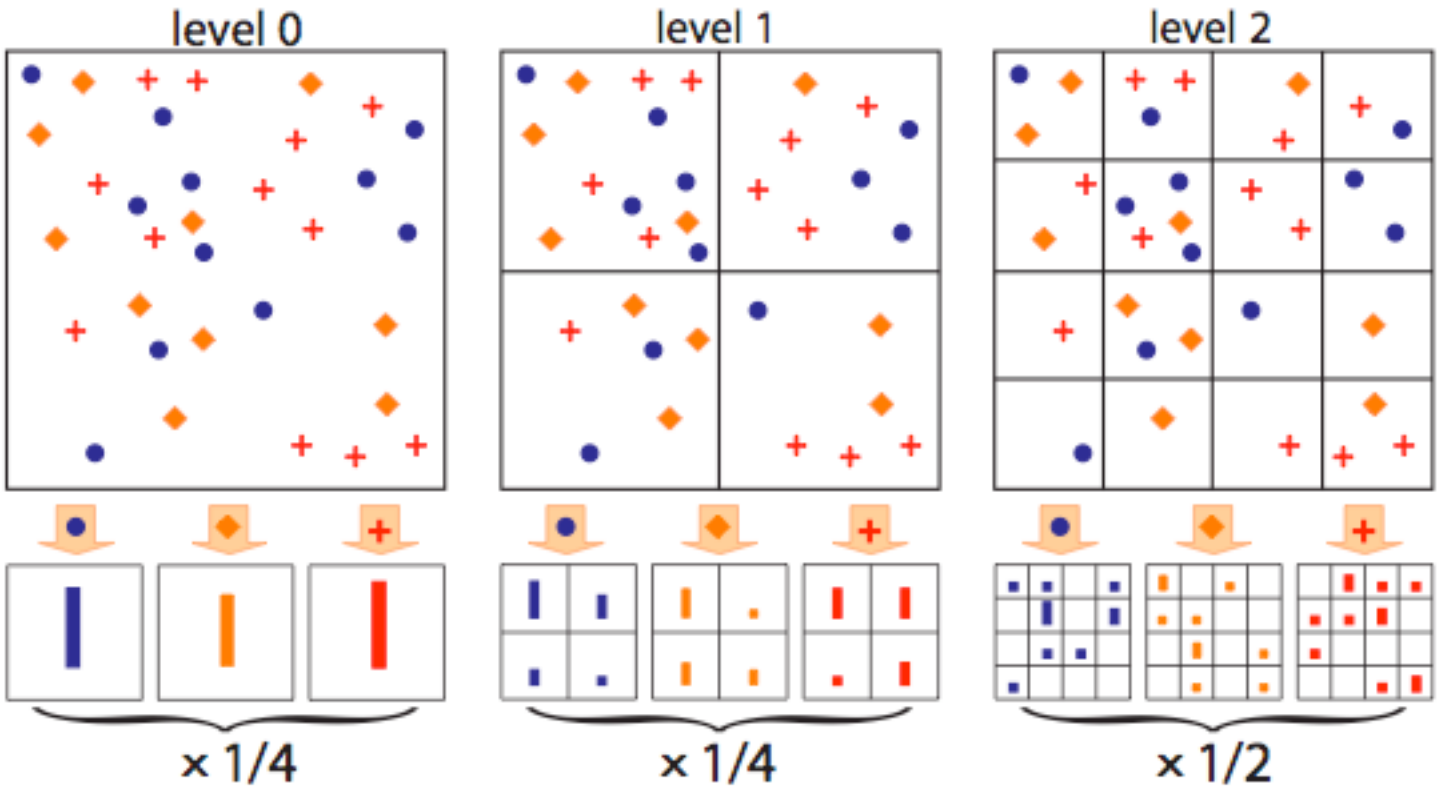
Matching affine covariant regions



Note that they still don't look exactly the same even on easy images!

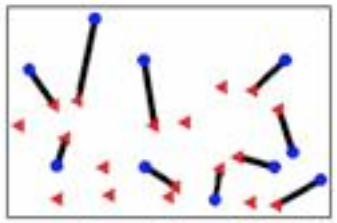
Lowe's orientation histogram helps, but Grauman & Darrell and Lazebnik et al have a neat alternative

Embedding



Grauman's Pyramid Match Kernel

"Match" score for sets X, Y,
of features:



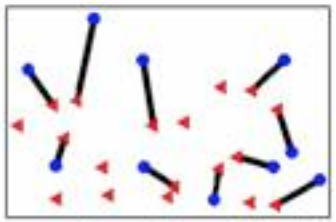
optimal partial
matching

$$\max_{\pi:\mathbf{X}\to\mathbf{Y}} \sum_{\mathbf{x}_i\in\mathbf{X}} \mathcal{S}(\mathbf{x}_i, \pi(\mathbf{x}_i))$$

Idea from Statistics: Mallow's 1972
Included the method of quantizing
feature space, which was rediscovered by
Rubner et al 1998 as the
Earth Mover's Distance.

Grauman's Pyramid Match Kernel

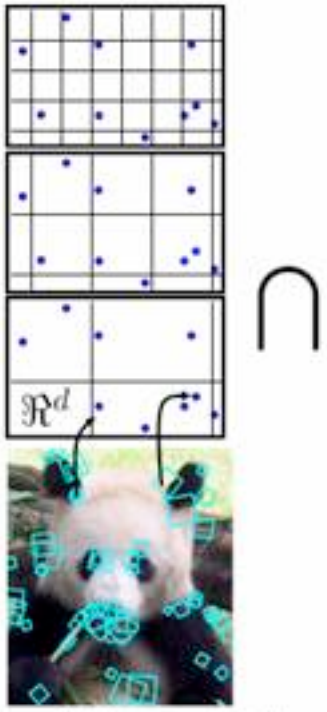
"Match" score for sets X, Y, of features:



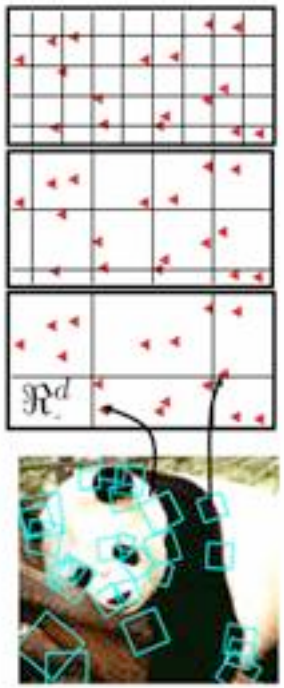
optimal partial matching

$$\max_{\pi:\mathbf{X}\to\mathbf{Y}} \sum_{\mathbf{x}_i\in\mathbf{X}} \mathcal{S}(\mathbf{x}_i, \pi(\mathbf{x}_i))$$

$\approx$



$\cap$



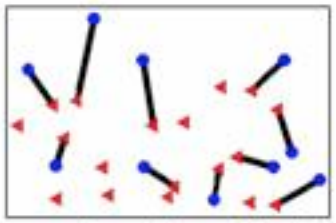
$\mathbf{X} = \{\vec{\mathbf{x}}_1, \ldots, \vec{\mathbf{x}}_m\}$ $\mathbf{Y} = \{\vec{\mathbf{y}}_1, \ldots, \vec{\mathbf{y}}_n\}$

Indyk and Thaper 2003
Showed how to embed points in a multiscale pyramid so that the l2 norm on the embedding approximated EMD

Idea from Statistics: Mallow's 1972
Included the method of quantizing feature space, which was rediscovered by Rubner et al 1998 as the Earth Mover's Distance (EMD)

Grauman's Pyramid Match Kernel

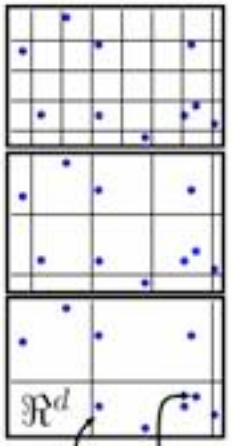
"Match" score for sets X, Y, of features:



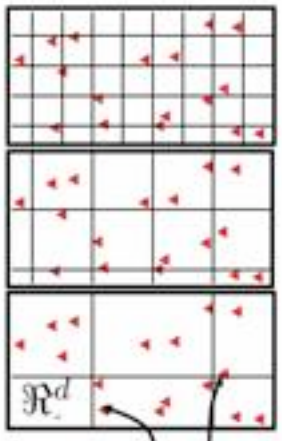
optimal partial matching

$$\max_{\pi:\mathbf{X}\to\mathbf{Y}} \sum_{\mathbf{x}_i\in\mathbf{X}} \mathcal{S}(\mathbf{x}_i, \pi(\mathbf{x}_i))$$

$\approx$



$\cap$



$\mathbf{X} = \{\vec{\mathbf{x}}_1, \ldots, \vec{\mathbf{x}}_m\}$ $\mathbf{Y} = \{\vec{\mathbf{y}}_1, \ldots, \vec{\mathbf{y}}_n\}$

Indyk and Thaper 2003
Showed how to embed points in a multiscale pyramid so that the l2 norm on the embedding approximated EMD

Grauman replaced l2 with histogram intersection.

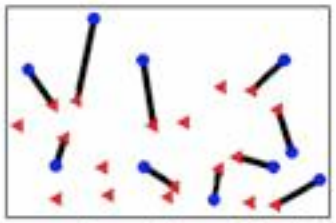
Histogram intersection

$$\mathcal{I}\left(H(\mathbf{X}), H(\mathbf{Y})\right) = \sum_{j=1}^{r} \min\left(H(\mathbf{X})_j, H(\mathbf{Y})_j\right)$$

Histogram Intersection / Min Kernel is positive definite, so we can use it for a Kernelized SVM

Grauman's Pyramid Match Kernel

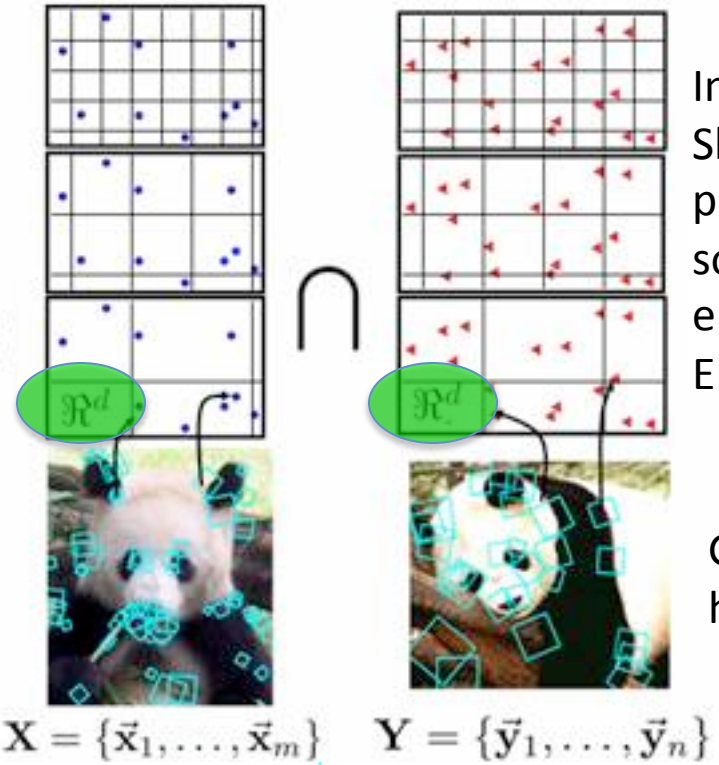
"Match" score for sets X, Y, of features:



optimal partial matching

$$\max_{\pi:\mathbf{X}\to\mathbf{Y}} \sum_{\mathbf{x}_i\in\mathbf{X}} \mathcal{S}(\mathbf{x}_i, \pi(\mathbf{x}_i))$$

$\approx$



Indyk and Thaper 2003
Showed how to embed points in a multiscale pyramid so that the l2 norm on the embedding approximated EMD

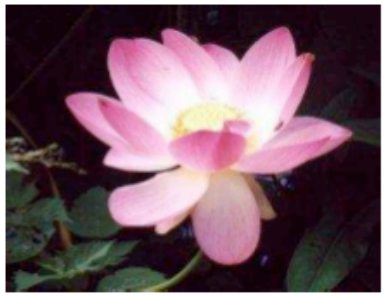
Grauman replaced l2 with histogram intersection.

Histogram intersection

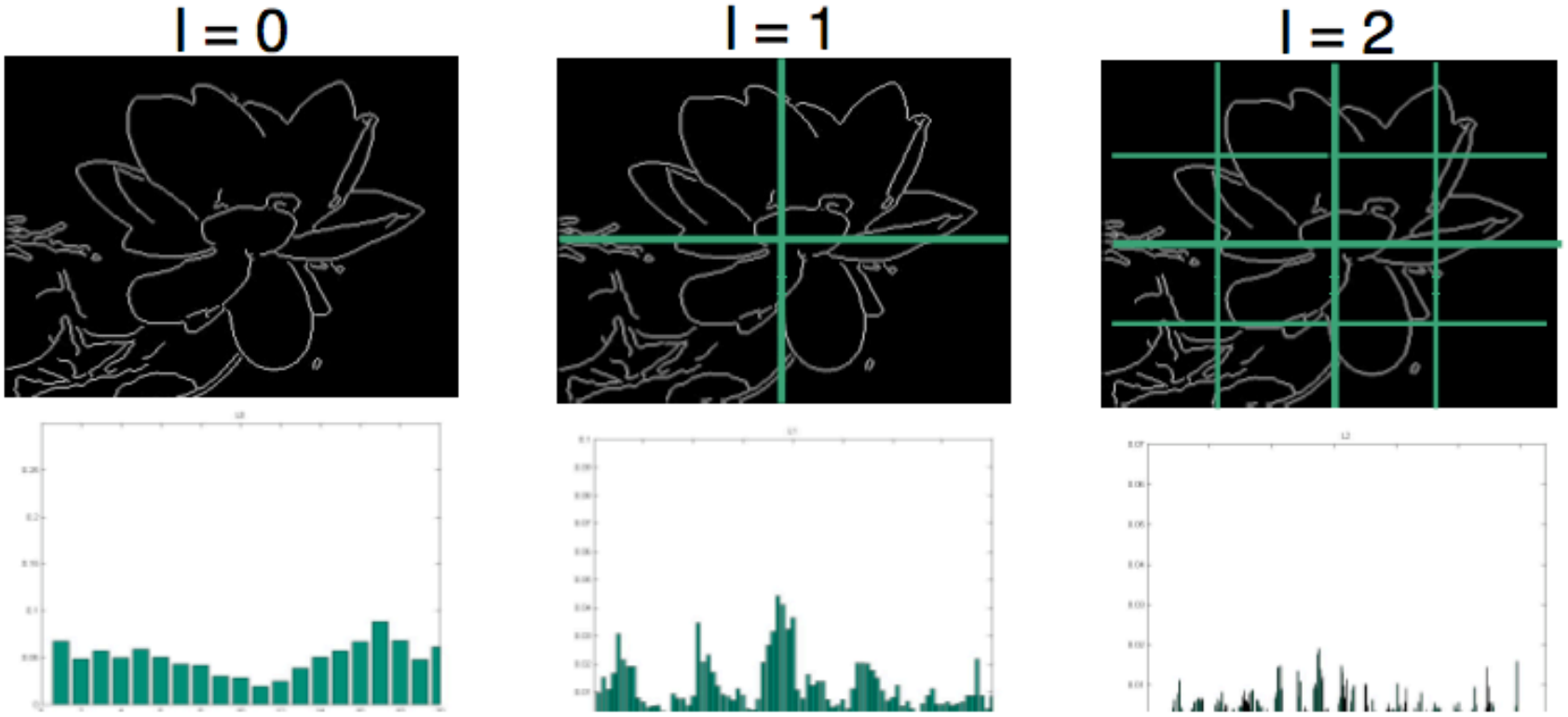
$$\mathcal{I}\left(H(\mathbf{X}), H(\mathbf{Y})\right) = \sum_{j=1}^{r} \min\left(H(\mathbf{X})_j, H(\mathbf{Y})_j\right)$$

Histogram Intersection / Min Kernel is positive definite, so we can use it for a Kernelized SVM

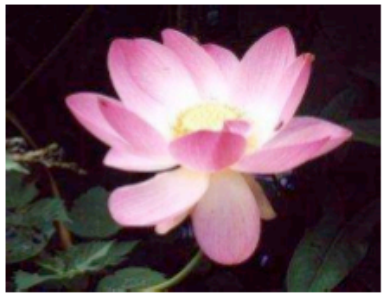
Spatial Pyramid Match (Lazebnik)



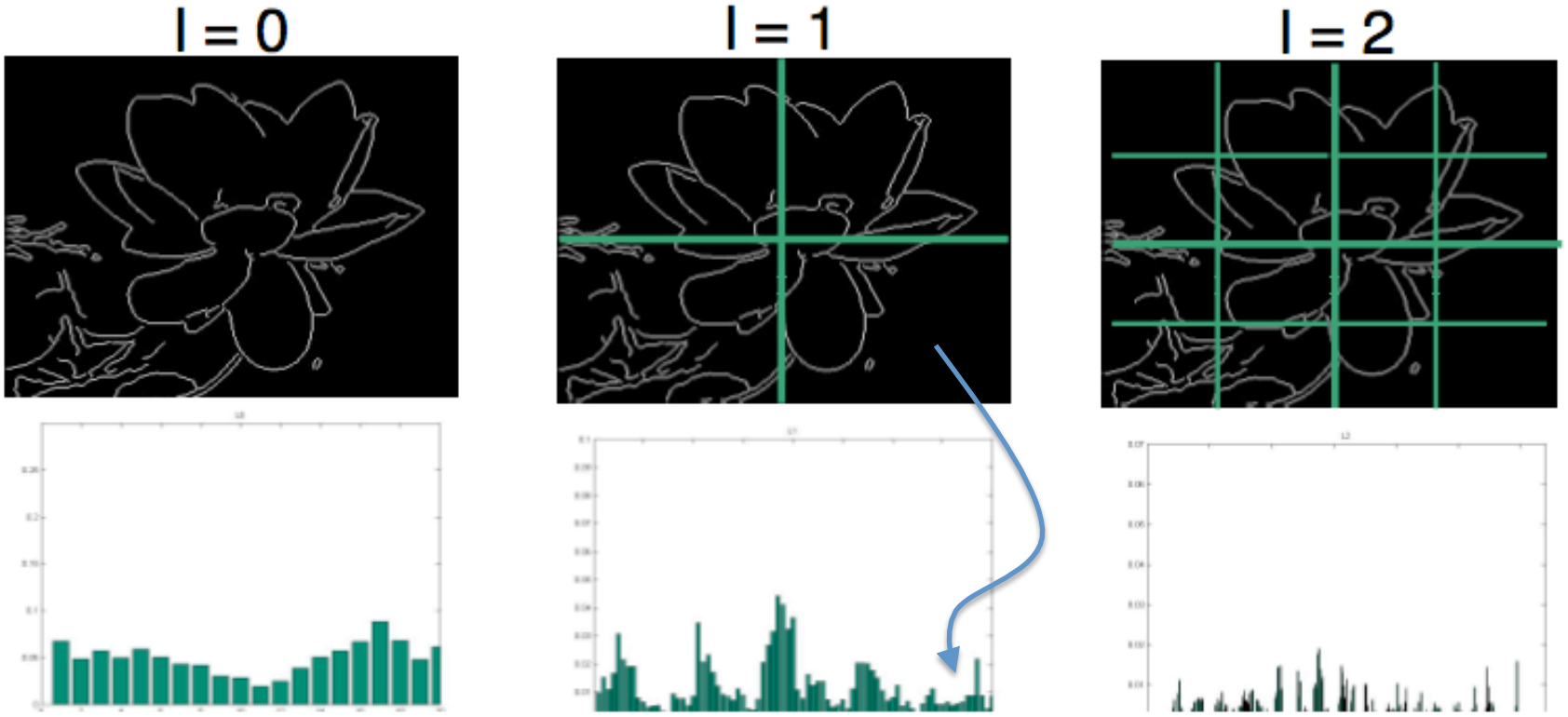
Only use pyramid for the spatial coordinates of features.



Spatial Pyramid Match (Lazebnik)

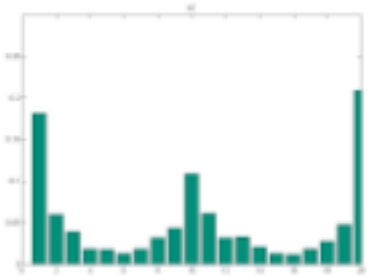


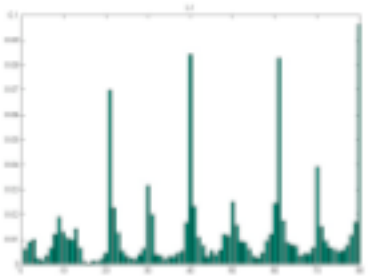
Applied to large region or whole image,
No interest point operator.

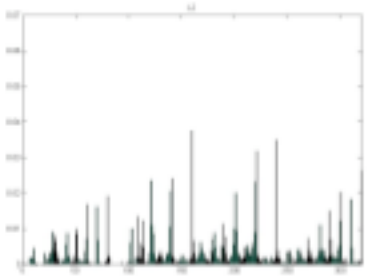


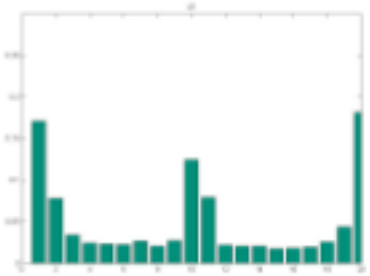
Rotation / scale invariance not always needed.

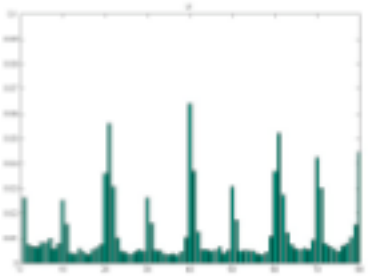
Airplanes on the runway are level.

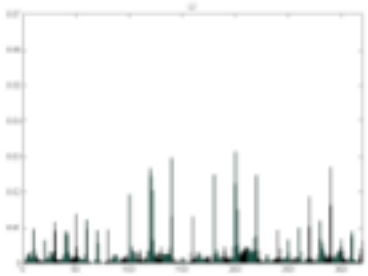




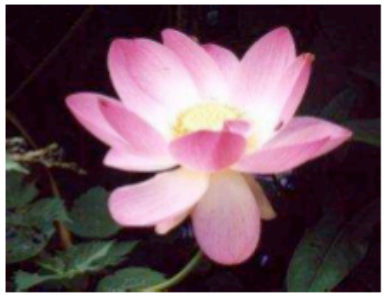








Spatial Pyramid Kernel (Lazebnik)

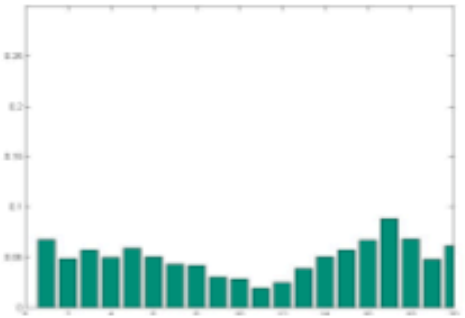


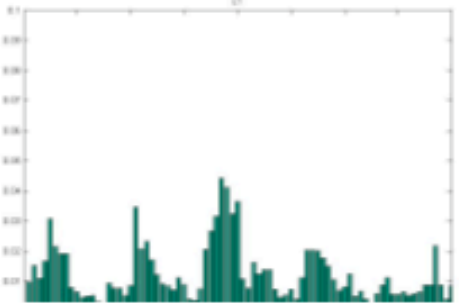
l = 0

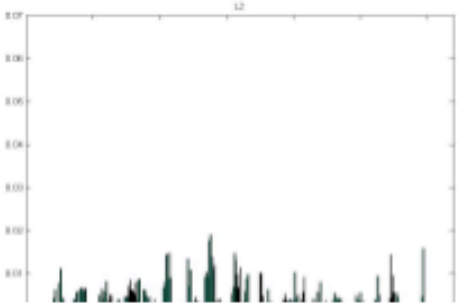
Distribution of edge features x, y, orientation, energy

$$E(x, y, o) =$$ Edge energy at x,y in orientation o

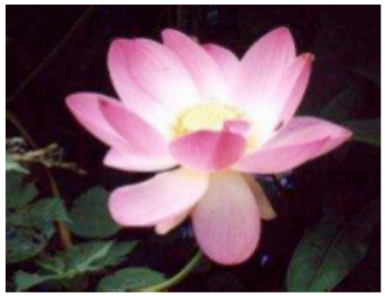
Histograms are just sums of different slices of E
(just a linear projection if E is represented discretely)







Spatial Pyramid Kernel (Lazebnik)



l = 0

Distribution of edge features x, y, orientation, energy

$E(x, y, o) =$ Edge energy at x,y in orientation o

Histograms are just sums of different slices of E
(just a linear projection if E is represented discretely)

Same for GIST, Shape Contexts, Geometric Blur, HOG etc.
The only impediment to an understanding of all of these features as simple projections of something like E() above is the min kernel…

Unified Feature Pipeline

Image

Edges/filter responses

Contrast Normalization

Projection

Comparison

L2
Inner product
Min Kernel

Max-Margin Additive Classifiers for Detection

Subhransu Maji (UC Berkeley)
Alex Berg (Columbia University)

Will be a talk at ICCV 2009 in Kyoto

Detection

Find pedestrians

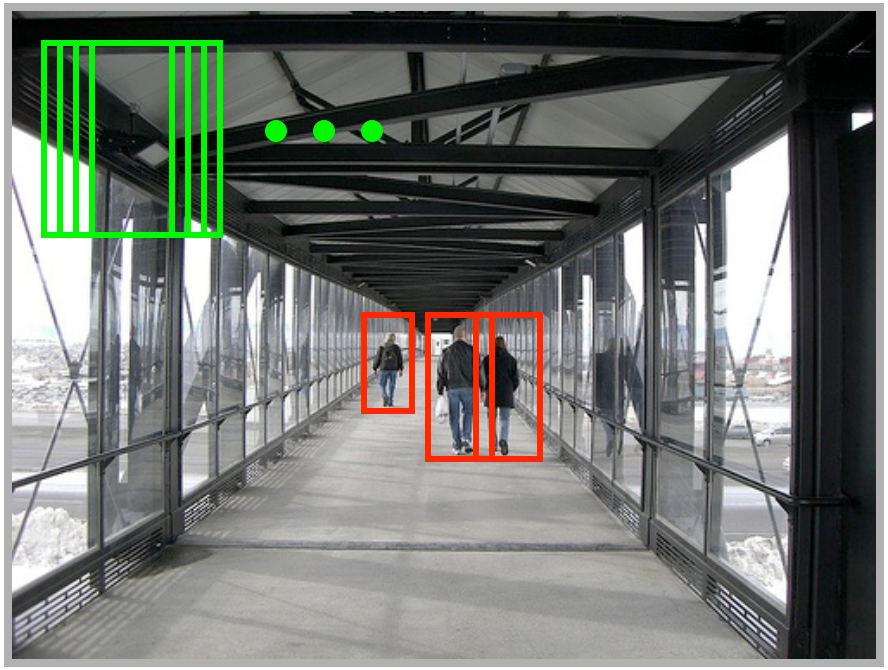
Detection

Find pedestrians

Detection

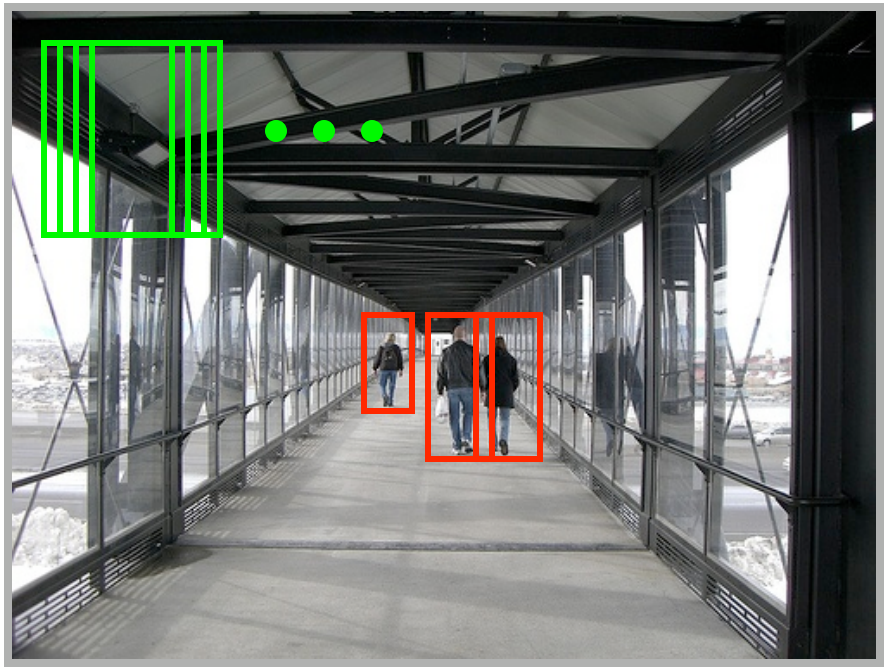
Find pedestrians

Detection



Find pedestrians

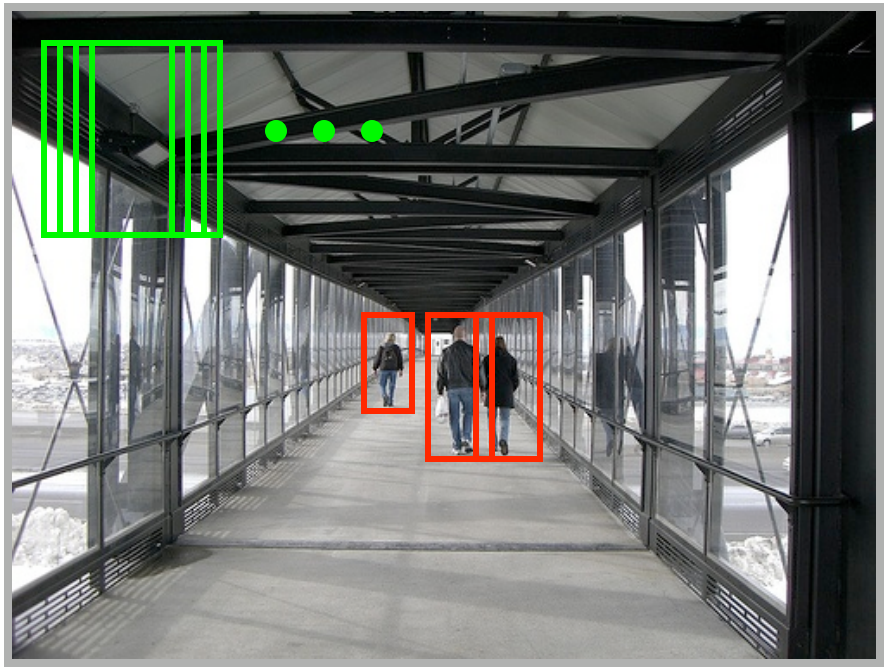
Detection



Find pedestrians

10^4 to 10^6 or more windows per image

Detection



Find pedestrians

10^4 to 10^6 or more windows per image

Boosting + Decision Trees
Viola & Jones (faces)

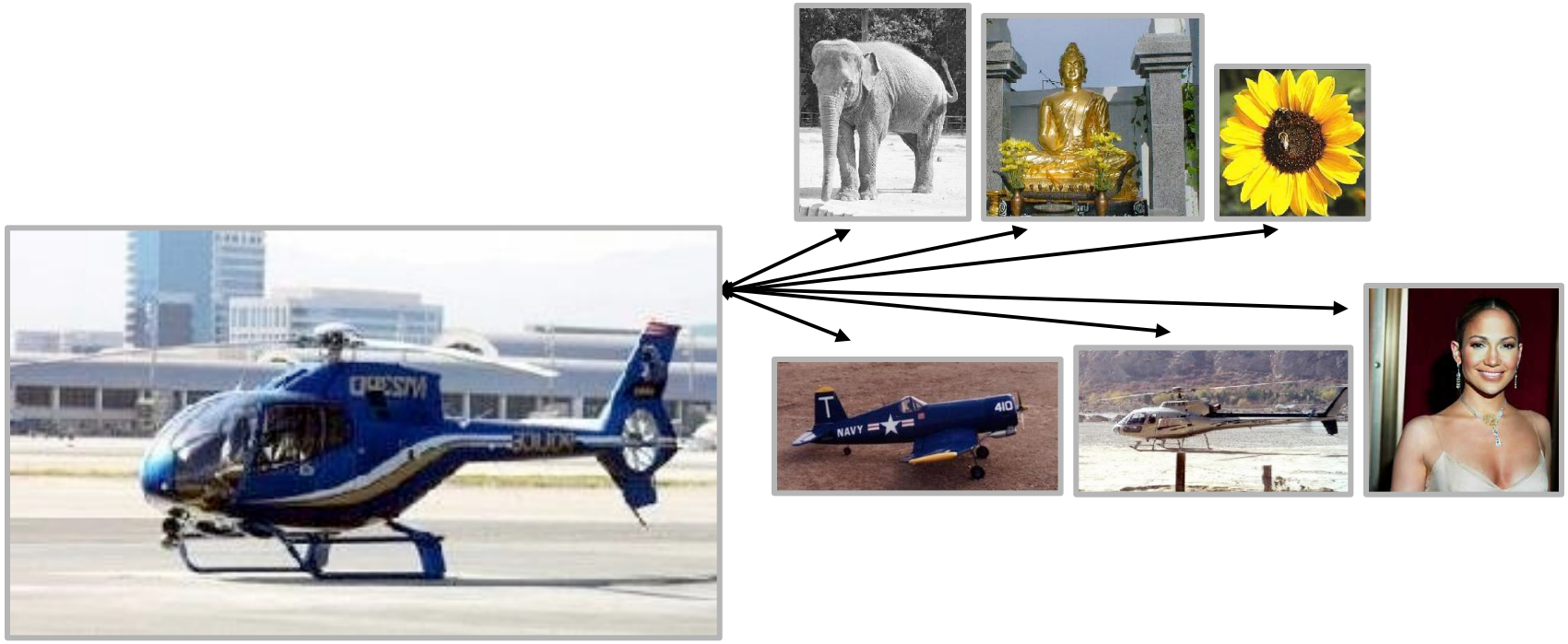
Linear Classifier
Dalal & Triggs (pedestrians)

Neural Networks
Rowley et al (faces)

Classification

What is this?

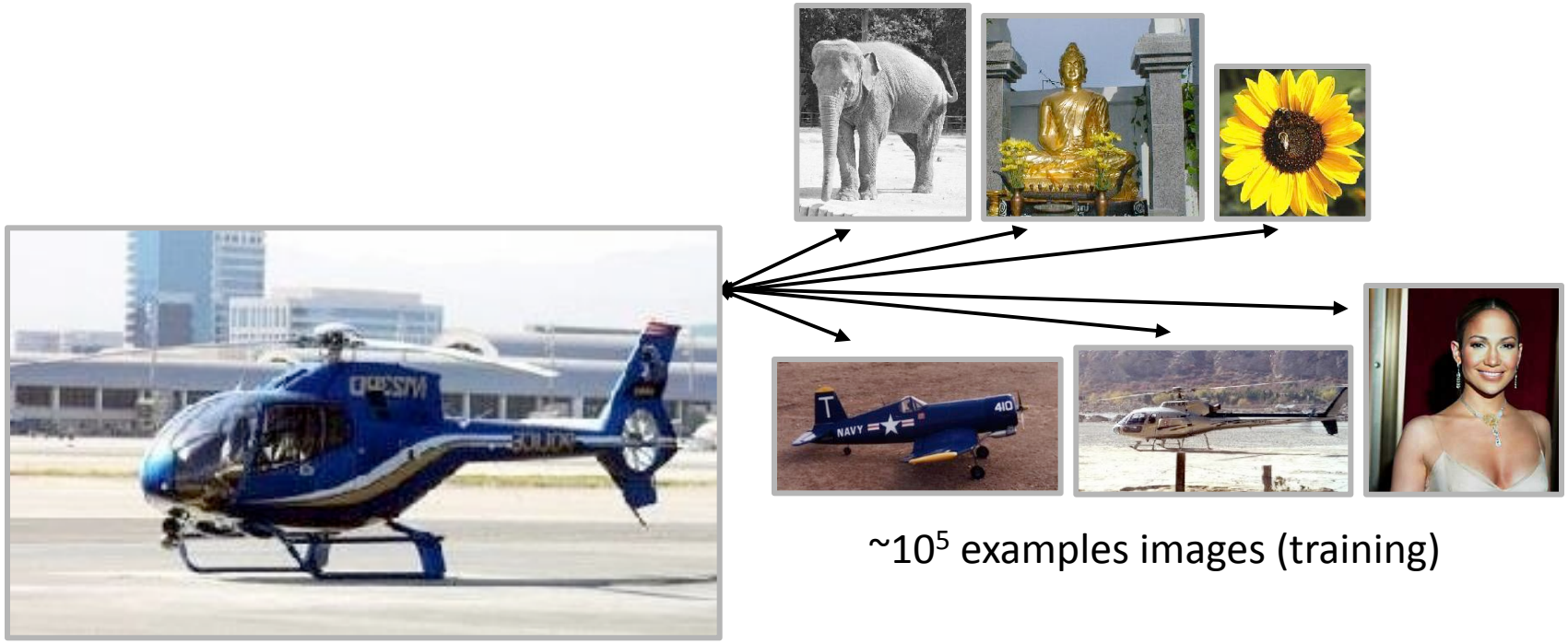
Classification



What is this?

Choose from many categories

Classification

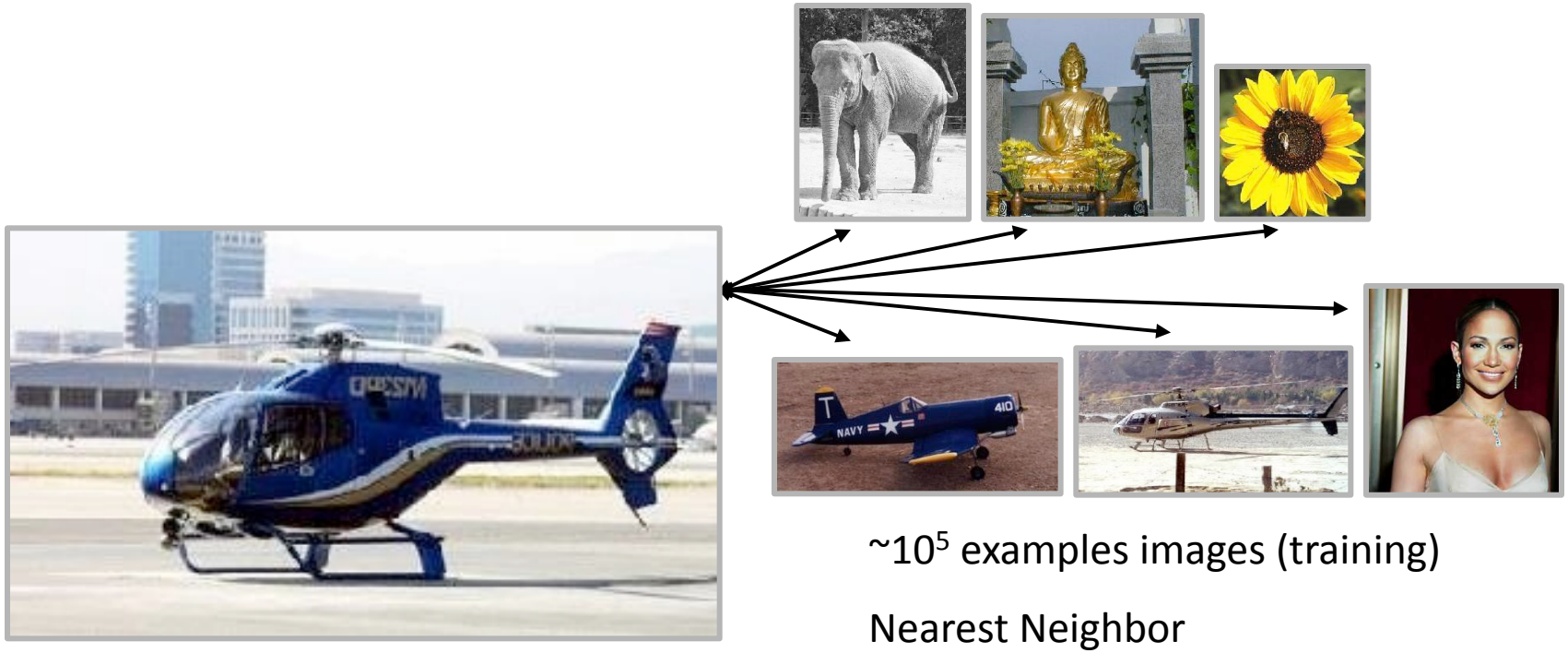


~10^5 examples images (training)

What is this?

Choose from many categories

Classification



What is this?

Choose from many categories

~10^5 examples images (training)

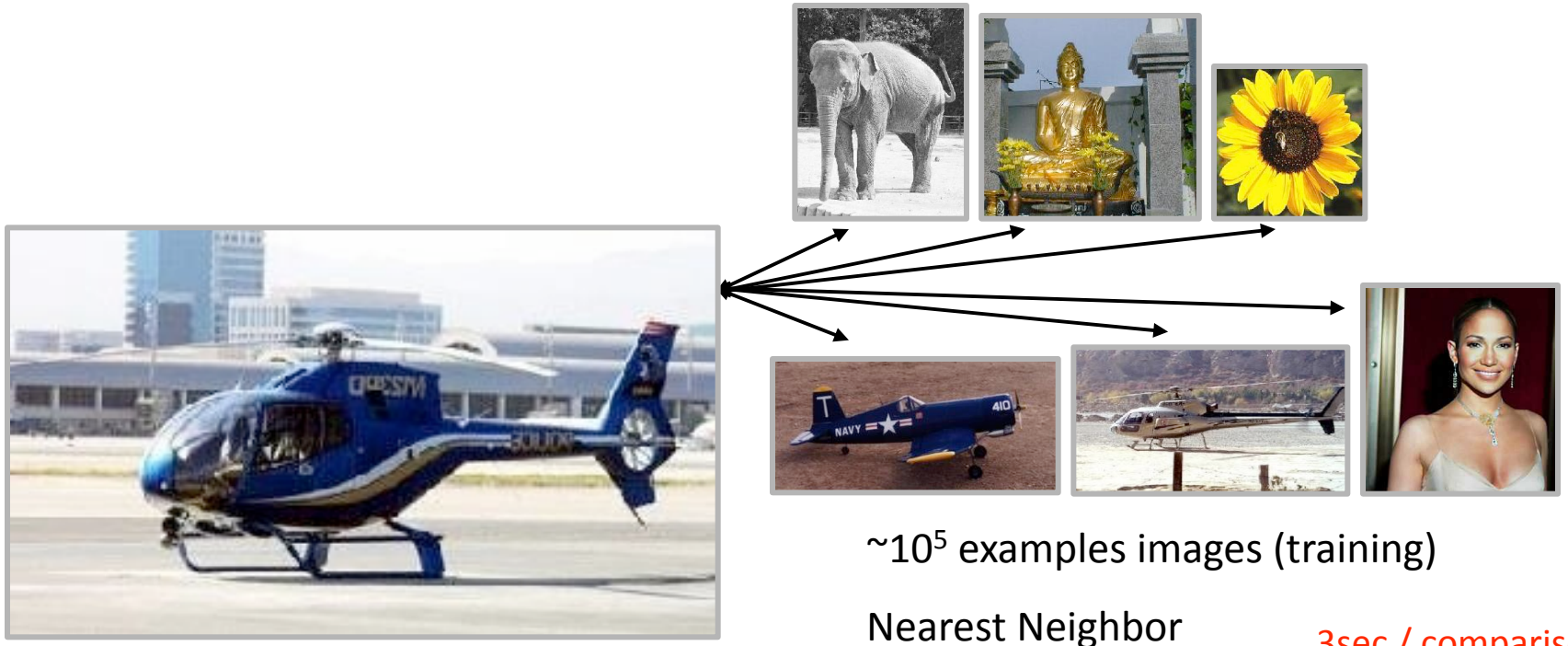
Nearest Neighbor
Berg (Caltech 101)

Kernelized SVM
Grauman et al (Caltech 101)

Combination of SVMs
Varma et al (Caltech 101)

(skipping model based methods)

Classification



What is this?

Choose from many categories

~10^5 examples images (training)

Nearest Neighbor
Berg (Caltech 101)

3sec / comparison

Kernelized SVM
Grauman et al (Caltech 101)

0.001 sec / comparison

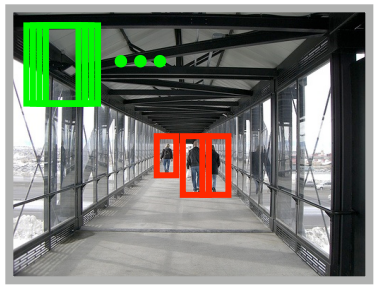
Combination of SVMs
Varma et al (Caltech 101)

Slow?

(skipping model based methods)

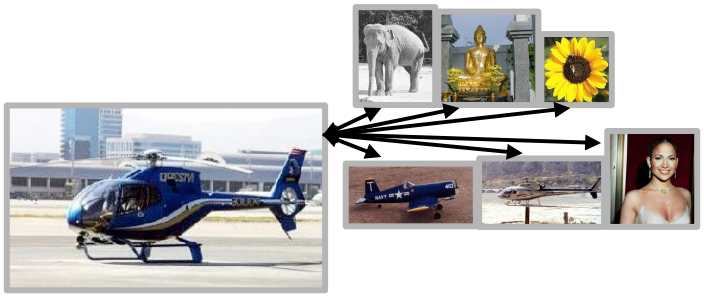
Caltech 101 – Fei-Fei Li, Pietro Perona 2004

Detection



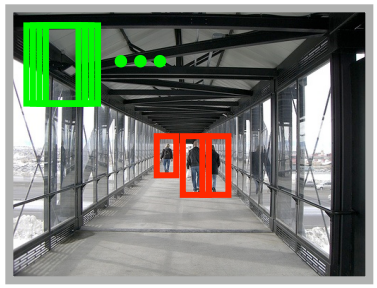
Linear Classifier

Classification



Kernelized SVM Classifier

Detection

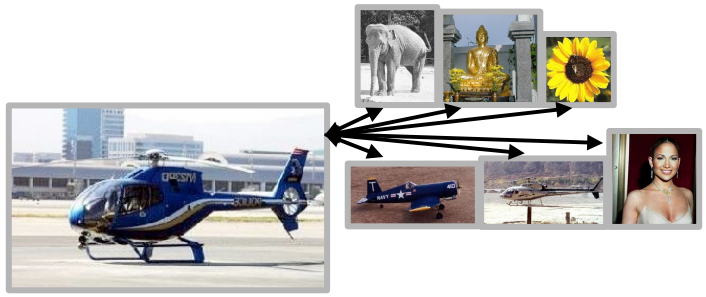


Linear Classifier

$$h(x) = \left(\sum_{i=1}^{\#\text{dimensions}} w_i x_i \right) + b$$

Decision function is *sign(h)*

Classification

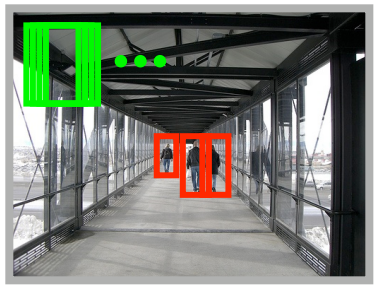


Kernelized SVM Classifier

$$h(x) = \sum_{j=1}^{\#\text{sv}} \alpha^j K(x, x^j) + b$$

Decision function is *sign(h)*

Detection



Linear Classifier

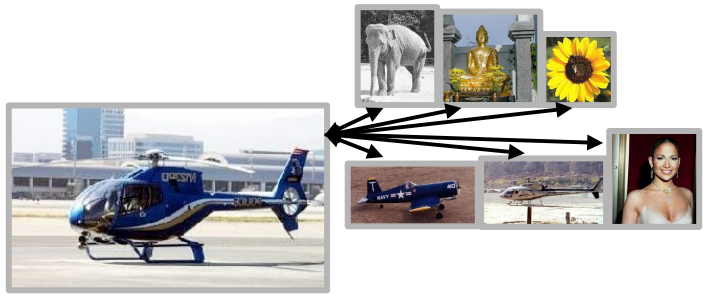
O(#dims)

$$h(x) = \left(\sum_{i=1}^{\#\text{dimensions}} w_i x_i \right) + b$$

Test feature vector

One coordinate of feature vector

Classification



Kernelized SVM Classifier

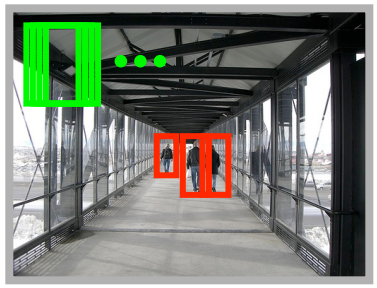
O(#dims x #sv)

$$h(x) = \sum_{j=1}^{\#\text{sv}} \alpha^j K(x, x^j) + b$$

Kernel Function (comparison)

Support Vector (training example)

Detection



Linear Classifier

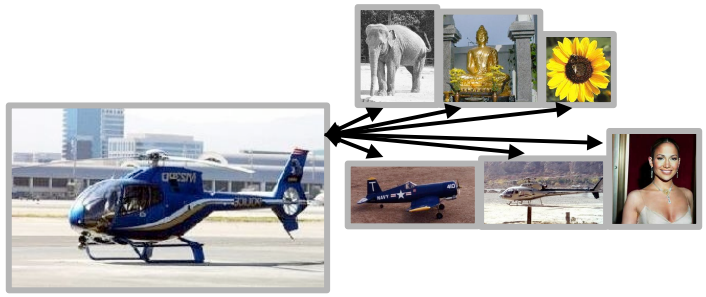
O(#dims)

$$h(x) = \left(\sum_{i=1}^{\#\text{dimensions}} w_i x_i \right) + b$$

Feature vector

One coordinate of feature vector

Classification



Kernelized SVM Classifier

O(#dims x #sv)

$$h(x) = \sum_{j=1}^{\#\text{sv}} \alpha^j K(x, x^j) + b$$

Kernel Function (comparison)

Support Vector (training example)

A SVM with *Additive* kernel can be evaluated efficiently

Maji, Berg, Malik CVPR 2008

If $$K(a,b) = \sum_{i=1}^{\#\text{dimensions}} K_i(a_i, b_i)$$

Then
$$h(x) = \sum_{j=1}^{\#\text{sv}} \alpha^j K(x, x^j) + b$$
$$= \sum_{j=1}^{\#\text{sv}} \alpha^j \left(\sum_{i=1}^{\#\text{dimensions}} K_i(x_i, x_i^j) \right) + b$$
$$= \sum_{i=1}^{\#\text{dimensions}} h_i(x_i)$$

A SVM with Additive Kernel can be Evaluated Efficiently

Maji, Berg, Malik CVPR 2008

If $$K(a,b) = \sum_{i=1}^{\#\text{dimensions}} K_i(a_i, b_i)$$

If you have an additive kernel…

Then
$$h(x) = \sum_{j=1}^{\#\text{sv}} \alpha^j K(x, x^j) + b$$
$$= \sum_{j=1}^{\#\text{sv}} \alpha^j \left(\sum_{i=1}^{\#\text{dimensions}} K_i(x_i, x_i^j) \right) + b$$
$$= \sum_{i=1}^{\#\text{dimensions}} h_i(x_i)$$

then the SVM decision function is additive.

A SVM with Additive Kernel can be Evaluated Efficiently

Maji, Berg, Malik CVPR 2008

If $$K(a,b) = \sum_{i=1}^{\#\text{dimensions}} K_i(a_i, b_i)$$

If you have an additive kernel…

Then $$h(x) = \sum_{j=1}^{\#\text{sv}} \alpha^j K(x, x^j) + b$$

$$= \sum_{j=1}^{\#\text{sv}} \alpha^j \left(\sum_{i=1}^{\#\text{dimensions}} K_i(x_i, x_i^j) \right) + b$$

$$= \sum_{i=1}^{\#\text{dimensions}} h_i(x_i)$$

then the SVM decision function is additive.

Evaluate these 1D functions efficiently using a look up table, spline (exact or approximate)

Intersection or Min Kernel

Maji, Berg, Malik CVPR 2008

$$K_{\min}(a,b) = \sum_{i=1}^{\#\text{dimensions}} min(a_i, b_i)$$

The Intersection or Min Kernel

Grauman et al use this on Multiscale Histograms to approximate the linear assignment problem (and do recognition)

Lazebnik et al refine this approach to only use multiple scales for position, and not for the features

Much follow on work

Intersection or Min Kernel

Maji, Berg, Malik CVPR 2008

$$K_{\min}(a,b) = \sum_{i=1}^{\#\text{dimensions}} min(a_i, b_i)$$

The Intersection or Min Kernel

$$\begin{aligned} h(x) &= \sum_{j=1}^{\#\text{sv}} \alpha^j K_{\min}(x, x^j) + b \\ &= \sum_{j=1}^{\#\text{sv}} \alpha^j \left(\sum_{i=1}^{\#\text{dimensions}} \min(x_i, x_i^j) \right) + b \\ &= \sum_{i=1}^{\#\text{dimensions}} h_i(x_i) + b \end{aligned}$$

Where $h_i(x_i) = \sum_{j=1}^{\#sv} \alpha^j \min(x_i, x_i^j)$

Intersection or Min Kernel

Maji, Berg, Malik CVPR 2008

$$K_{\min}(a,b) = \sum_{i=1}^{\#\text{dimensions}} min(a_i, b_i)$$

The Intersection or Min Kernel

$$\begin{aligned} h(x) &= \sum_{j=1}^{\#\text{sv}} \alpha^j K_{\min}(x, x^j) + b \\ &= \sum_{j=1}^{\#\text{sv}} \alpha^j \left(\sum_{i=1}^{\#\text{dimensions}} \min(x_i, x_i^j) \right) + b \\ &= \sum_{i=1}^{\#\text{dimensions}} h_i(x_i) + b \end{aligned}$$

The support vectors are constants, min(x_i , *constant*) is piecewise linear, so $h_i(x_i)$ is piecewise linear.

Where $h_i(x_i) = \sum_{j=1}^{\#sv} \alpha^j \min(x_i, x_i^j)$

Intersection or Min Kernel

Maji, Berg, Malik CVPR 2008

$$K_{\min}(a,b) = \sum_{i=1}^{\#\text{dimensions}} min(a_i, b_i)$$

The Intersection or Min Kernel

O(#dims x #sv) Becomes **O(#dims x log(#sv))** exact

or **O(#dims)** approx.

$$h(x) = \sum_{j=1}^{\#\text{sv}} \alpha^j K_{\min}(x, x^j) + b$$

$$= \sum_{j=1}^{\#\text{sv}} \alpha^j \left(\sum_{i=1}^{\#\text{dimensions}} \min(x_i, x_i^j) \right) + b$$

$$= \sum_{i=1}^{\#\text{dimensions}} h_i(x_i) + b$$

The support vectors are constants, min(x_i , *constant*) is piecewise linear, so $h_i(x_i)$ is piecewise linear.

Where $h_i(x_i) = \sum_{j=1}^{\#sv} \alpha^j \min(x_i, x_i^j)$

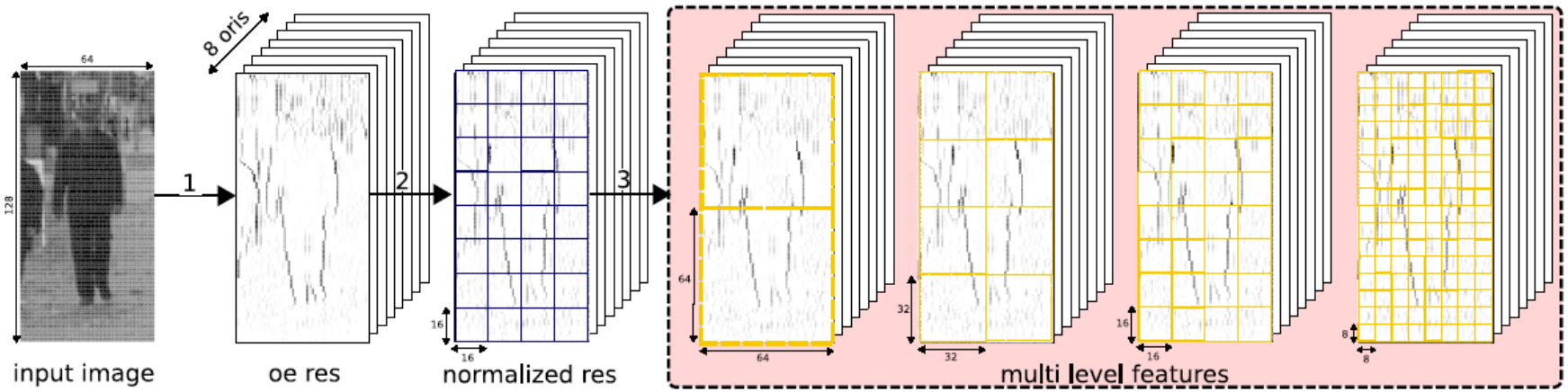
Time to Perform Classification

Maji, Berg, Malik CVPR 2008

Dataset	Model parameters		SVM kernel type		fast IKSVMs		
	#SVs	#features	linear	intersection	binary search	piecewise-const	piecewise-lin
INRIA Ped	3363	1360	0.07±0.00	659.1±1.92	2.57±0.03	0.34±0.01	0.43±0.01
DC Ped	5474±395	656	0.03±0.00	459.1±31.3	1.43±0.02	0.18±0.01	0.22±0.00
Caltech 101	175±46	1360	0.07±0.01	24.77±1.22	1.63±0.12	0.33±0.03	0.46±0.03

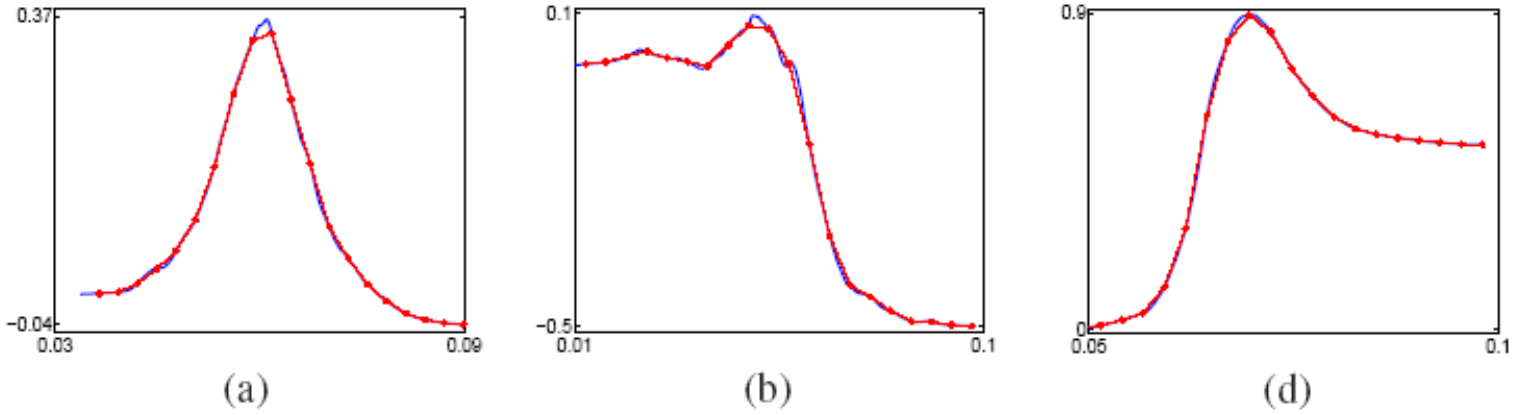
Times in seconds to classify 10,000 test vectors

Multiscale HOG features
(Very Similar to Spatial Pyramids)

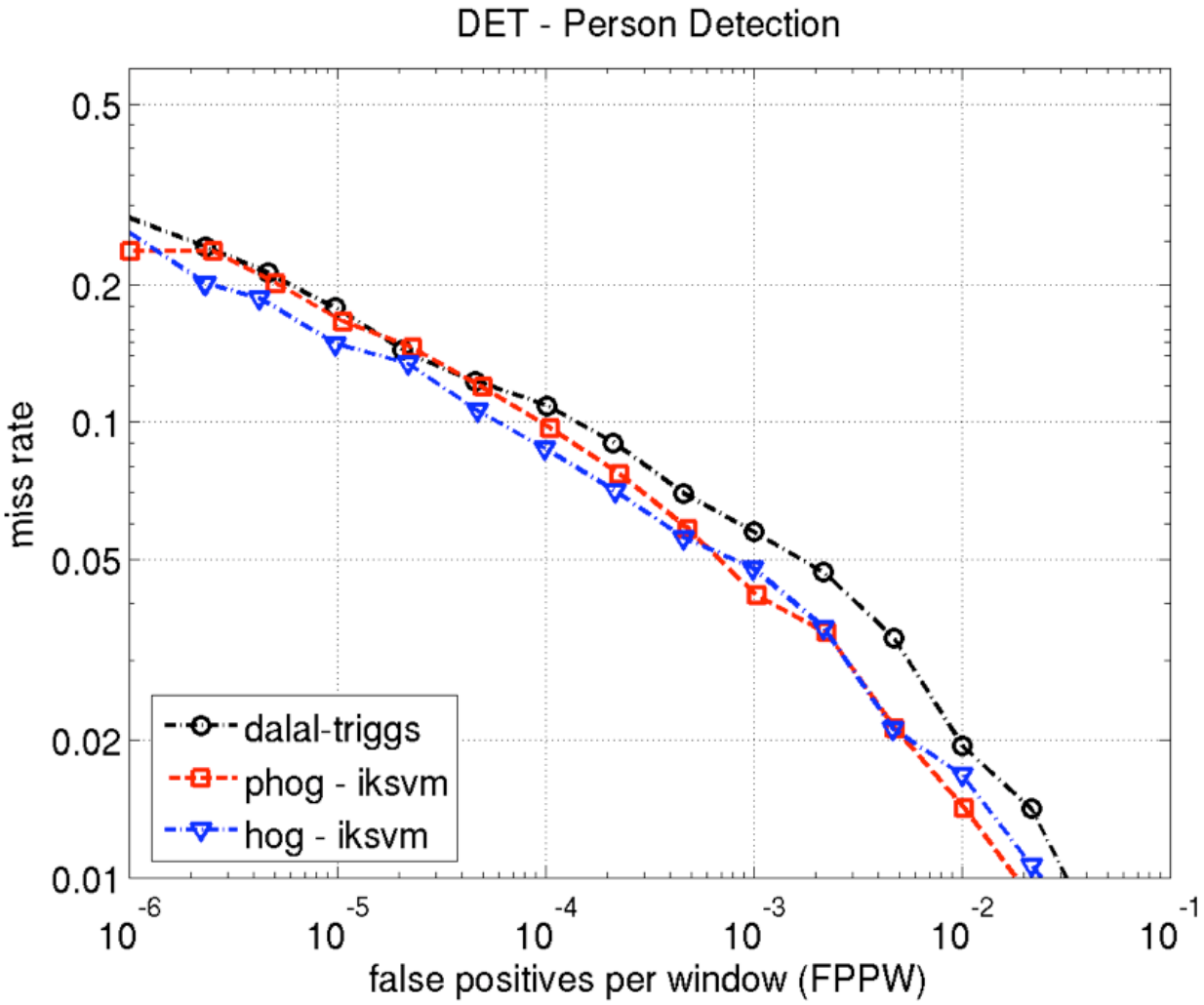


Based on histograms of response to eight orientated edge detections. Non-overlapping windows of integration and fixed size windows for contrast normalization allow efficient computation.

Example $h_i(x_i)$ and Approximations



Min Kernel “Better” than Linear



Min Kernel “Better” than Linear

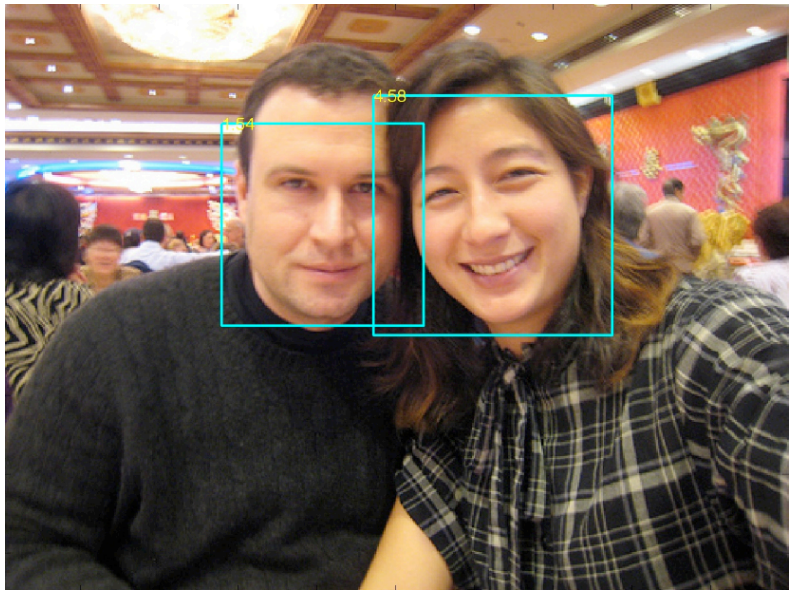
Caltech 101 with “simple features”
15 training examples per category

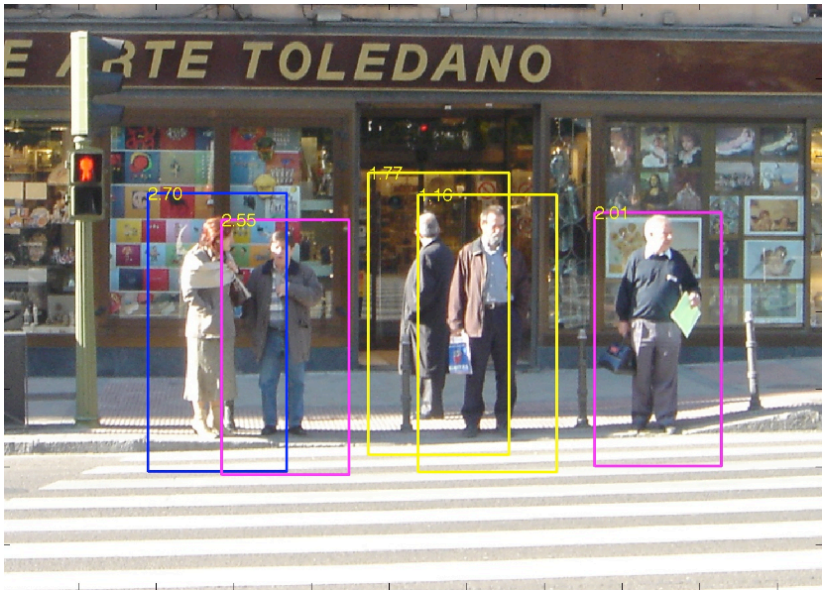
Linear SVM	40% correct
Min Kernel (IK) SVM	52% correct

Accuracy of Min Kernel vs Linear on Text classification

Accuracy Values					
Classification Method	**R8**	**R52**	**20Ng**	**Cade12**	**WebKb**
SVM (Linear Kernel)	0.9666(1)	0.9322(1)	0.8155(0.04)	0.5650(0.05)	0.8796(0.04)
SVM (Intersection Kernel)	0.9693(1)	0.9326(0.8)	0.8115(0.05)	0.5777(0.10)	0.9105(0.04)

Now we can use Min Kernel for Detection in Seconds Instead of Hours





Direct Training

It is possible to directly train classifiers with the same structure as the approximation without using support vectors at all. The formulation is very similar to a linear classifier, with different regularization. Can be trained efficiently using stochastic (sub)gradient descent.

Linear

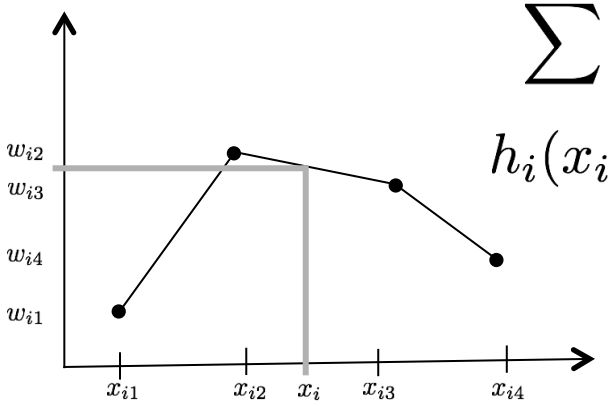
$$\begin{aligned} \text{minimize :} \quad & w'w + c\sum \xi^j & \\ \text{subject to :} \quad & y^i(w'x^j + b) & \geq 1 - \xi^j \\ & \xi^j & \geq 0 \end{aligned}$$

Piecewise Linear

$$\begin{aligned} \text{minimize :} \quad & \hat{w}'H\hat{w} + c\sum \xi^j & \\ \text{subject to :} \quad & \hat{y}^i(\hat{w}'\hat{x}^j + b) & \geq 1 - \xi^j \\ & \xi^j & \geq 0 \end{aligned}$$

$$\sum h_i(x_i) \qquad \sum \hat{h}_i(\hat{x}_i)$$

$$h_i(x_i) = w_i x_i \qquad \hat{h}_i(\hat{x}_i) = \hat{w}_i \hat{x}_i$$



$$x_i = \alpha x_{i2} + (1-\alpha)x_{i3}$$

$$\hat{x}_i = \begin{bmatrix} 0 & \alpha & 1-\alpha & 0 \end{bmatrix}$$

$$\hat{w}_i = \begin{bmatrix} w_{i1} & w_{i2} & w_{i3} & w_{i4} \end{bmatrix}$$

$$H = \begin{bmatrix} 1 & -1 & & & \\ -1 & 2 & -1 & & \\ & -1 & . & & \\ & & & 2 & -1 \\ & & & -1 & 1 \end{bmatrix}$$

Slightly different formulation

Linear

$$\min_w \frac{\lambda}{2} w'w + \frac{1}{m} \sum_i \ell\left(w; (x_i, y_i)\right)$$

Piecewise linear

$$\min_w \frac{\lambda}{2} \hat{w}' H \hat{w} + \frac{1}{m} \sum_i \ell\left(\hat{w}; (\hat{x}_i, y_i)\right)$$

Shalev-Schwartz, Singer, Srebro ICML 2007

$O\left(\frac{d}{\lambda\epsilon}\right)$ for ϵ accuracy

INPUT: S, λ, T, k

INITIALIZE: Choose $\mathbf{w}_1$ s.t. $\|\mathbf{w}_1\| \leq 1/\sqrt{\lambda}$

FOR $t = 1, 2, \ldots, T$

Choose $A_t \subseteq S$, where $|A_t| = k$

Set $A_t^+ = \{(\mathbf{x}, y) \in A_t : y \langle \mathbf{w}_t, \mathbf{x} \rangle < 1\}$

Set $\eta_t = \frac{1}{\lambda t}$

Set $\mathbf{w}_{t+\frac{1}{2}} = (1 - \eta_t \lambda)\mathbf{w}_t + \frac{\eta_t}{k} \sum_{(\mathbf{x},y) \in A_t^+} y\,\mathbf{x}$

Set $\mathbf{w}_{t+1} = \min\left\{1, \frac{1/\sqrt{\lambda}}{\|\mathbf{w}_{t+\frac{1}{2}}\|}\right\} \mathbf{w}_{t+\frac{1}{2}}$

OUTPUT: $\mathbf{w}_{T+1}$

Figure 1. The Pegasos Algorithm.

Shalev-Schwartz, Singer, Srebro ICML 2007

$O\left(\frac{d}{\lambda\epsilon}\right)$ for ϵ accuracy

$\|w\| \rightarrow \sqrt{w'Hw}$

INPUT: S, λ, T, k

INITIALIZE: Choose $\mathbf{w}_1$ s.t. $\sqrt{w_1' H w_1} \leq 1/\sqrt{\lambda}$

FOR $t = 1, 2, \ldots, T$

Choose $A_t \subseteq S$, where $|A_t| = k$

Set $A_t^+ = \{(\mathbf{x}, y) \in A_t : y \langle \mathbf{w}_t, \mathbf{x} \rangle < 1\}$

Set $\eta_t = \frac{1}{\lambda t}$

Set $\mathbf{w}_{t+\frac{1}{2}} = (1 - \eta_t \lambda H)\mathbf{w}_t + \frac{\eta_t}{k} \sum_{(\mathbf{x},y) \in A_t^+} y\,\mathbf{x}$

Set $\mathbf{w}_{t+1} = \min\left\{1, \frac{1/\sqrt{\lambda}}{\sqrt{w_{t+\frac{1}{2}}' H w_{t+\frac{1}{2}}}}\right\} \mathbf{w}_{t+\frac{1}{2}}$

OUTPUT: $\mathbf{w}_{T+1}$

Figure 1. The Pegasos Algorithm.

Maji, Berg, ICCV 2009

Shalev-Schwartz, Singer, Srebro ICML 2007

$O\left(\frac{d}{\lambda\epsilon}\right)$ for ϵ accuracy

$\|w\| \rightarrow \sqrt{w'Hw}$

INPUT: S, λ, T, k

INITIALIZE: Choose $\mathbf{w}_1$ s.t. $\sqrt{w_1' H w_1} \leq 1/\sqrt{\lambda}$

FOR $t = 1, 2, \ldots, T$

Choose $A_t \subseteq S$, where $|A_t| = k$

Set $A_t^+ = \{(\mathbf{x}, y) \in A_t : y \langle \mathbf{w}_t, \mathbf{x} \rangle < 1\}$

Set $\eta_t = \frac{1}{\lambda t}$

Set $\mathbf{w}_{t+\frac{1}{2}} = (1 - \eta_t \lambda H)\mathbf{w}_t + \frac{\eta_t}{k} \sum_{(\mathbf{x},y) \in A_t^+} y\,\mathbf{x}$

Set $\mathbf{w}_{t+1} = \min\left\{1, \frac{1/\sqrt{\lambda}}{\sqrt{w_{t+\frac{1}{2}}' H w_{t+\frac{1}{2}}}}\right\} \mathbf{w}_{t+\frac{1}{2}}$

OUTPUT: $\mathbf{w}_{T+1}$

Figure 1. The Pegasos Algorithm.

w and x are large but sparse, so we can get computation to scale with # non zeros.

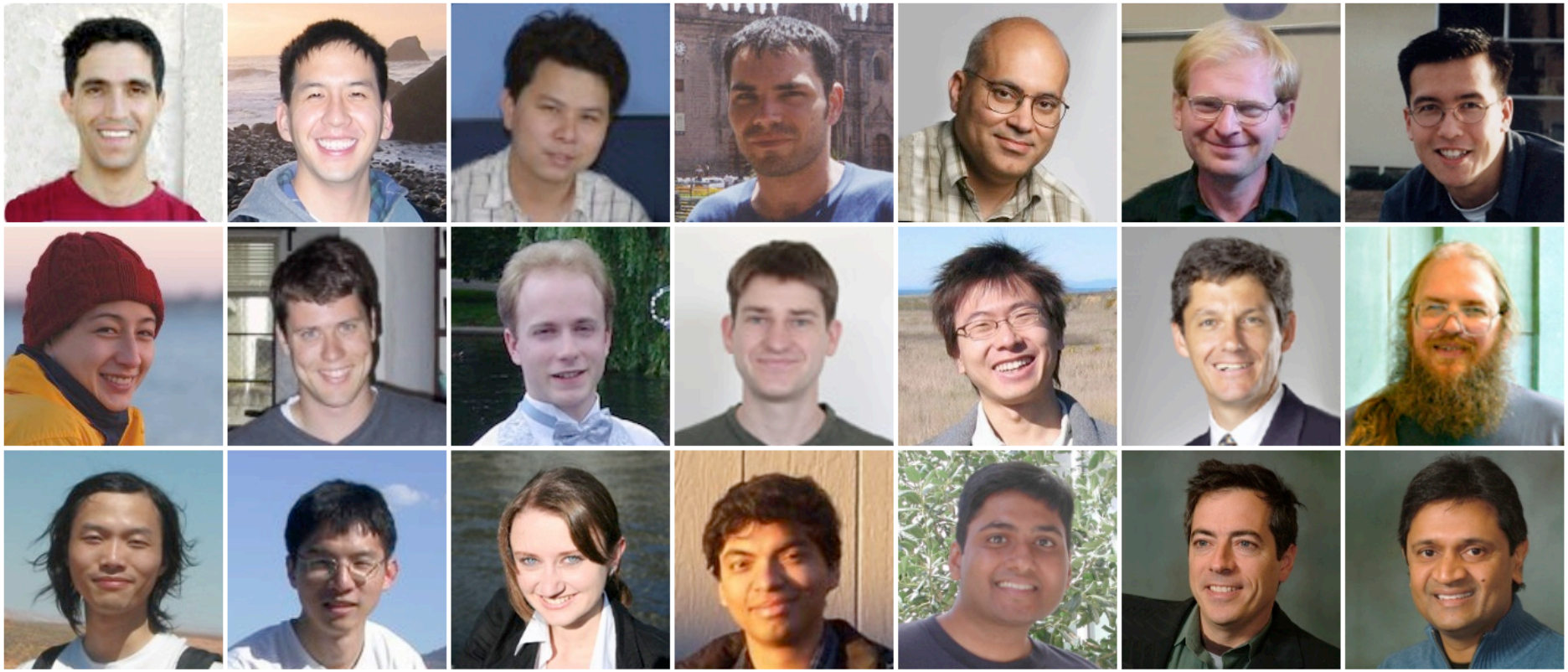
Conclusions

- SVM with an additive kernel is just sum of functions for each coordinate separately
- We can evaluate these classifiers fast enough to do sliding window detection
- Training time 100x faster than Viola Jones testing time ~10x slower than Violoa Jones, but may work for a broader range of categories
- Current work reducing training time to interactive
- May have applications beyond computer vision --
  the correct additional projections can approach arbitrary classifier.

Thank you

and my co-authors so far…

Alex Berg's coauthors (so far)



Attribute and Simile Classifiers for Face Verification

Neeraj Kumar, Alex Berg, Peter Belhumeur, Shree Nayar
Columbia University

Will be a talk at ICCV 2009 in Kyoto

Faces in the wild

as in *Names and Faces*
T. Berg et al
cvpr 2004

large variation in
pose illumination
expression lighting
etc.

Faces in the wild

as in *Names and Faces*
T. Berg et al
cvpr 2004

large variation in
pose illumination
expression lighting
etc.

Typical measures of face similarity (eg PCA+LDA) would say these faces are very different.

Faces in the wild

as in *Names and Faces*
T. Berg et al
cvpr 2004

large variation in
pose illumination
expression lighting
etc.

Typical measures of face similarity (eg PCA+LDA) would say these faces are very different.

Ferencz et al learned (discriminative) generalized linear models some other groups (caltech, UMass) have looked at similar approaches.

Faces in the wild

as in *Names and Faces*
T. Berg et al
cvpr 2004

large variation in
pose illumination
expression lighting
etc.

Erik Learned-Miller and the UMass group labeled a broader version of the names and faces data called: Labeled Faces in the Wild (LFW)

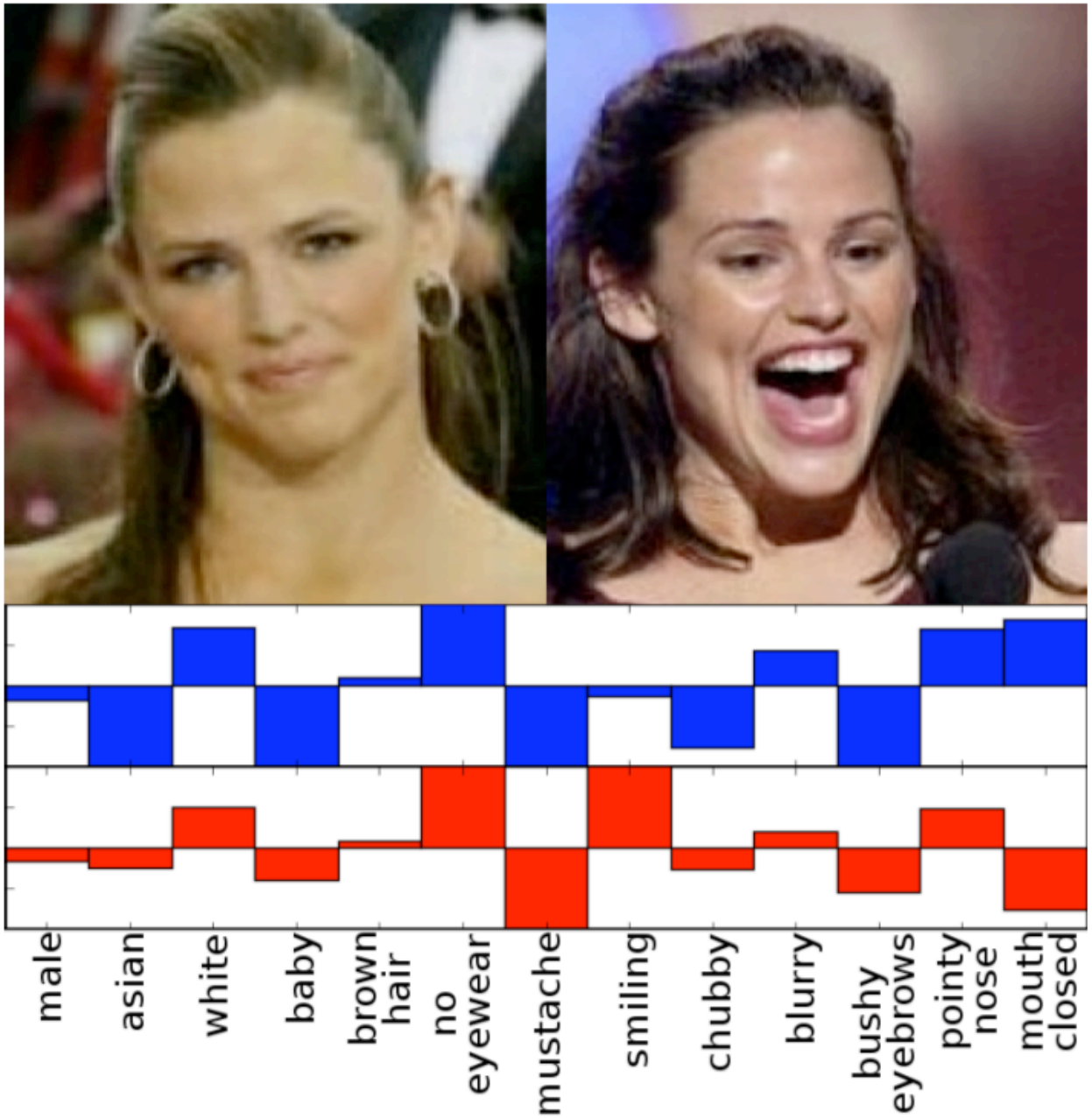
Results from many groups are available -- best are from Wolf et al and Nowak and Jurie.

Faces in the wild

as in *Names and Faces*
T. Berg et al
cvpr 2004

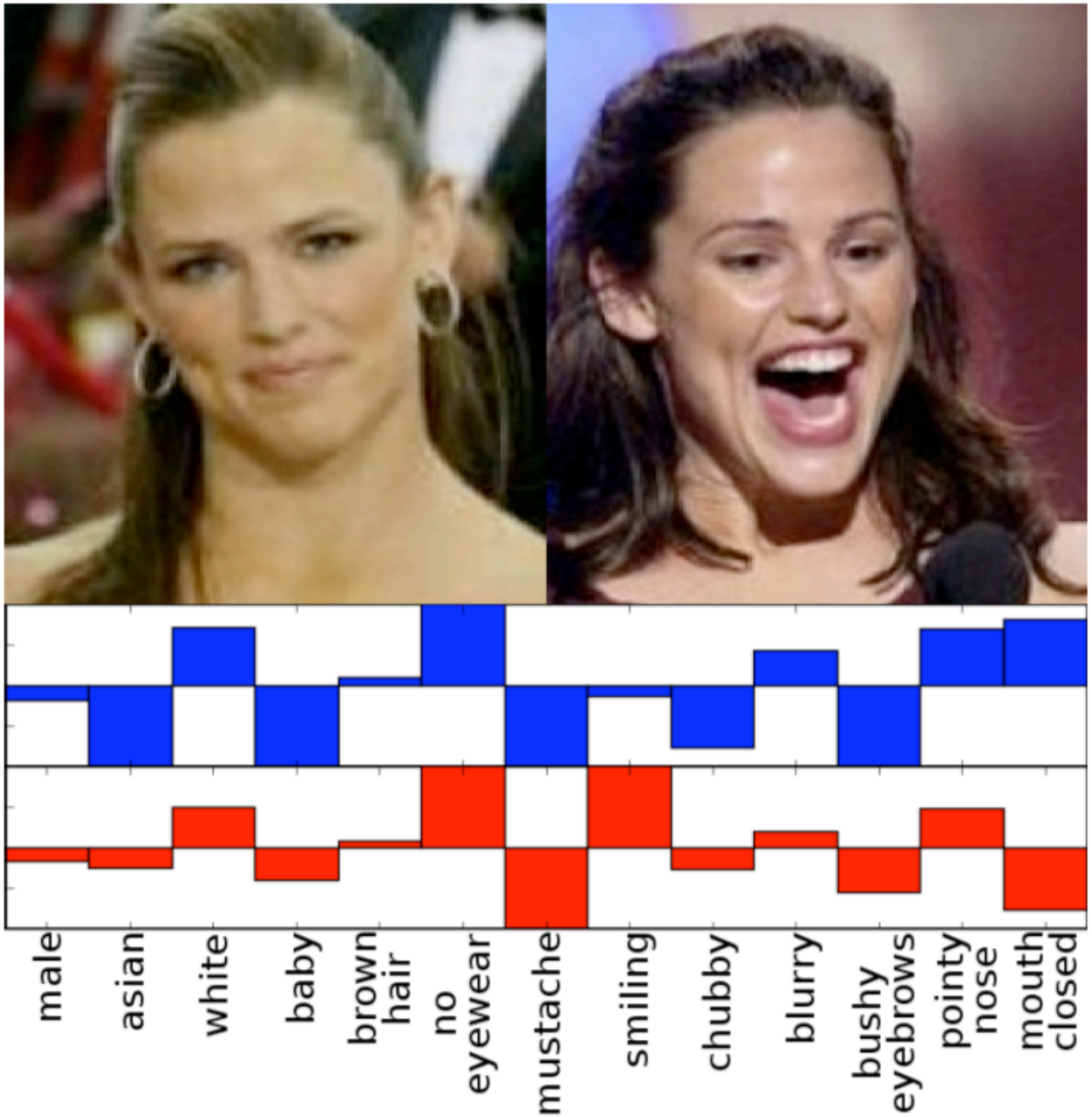
large variation in
pose illumination
expression lighting
etc.

We look at attributes of the faces that may be **robust** with respect to Identity, but are also **common** to many people
So that we can collect very large training sets for each attribute.

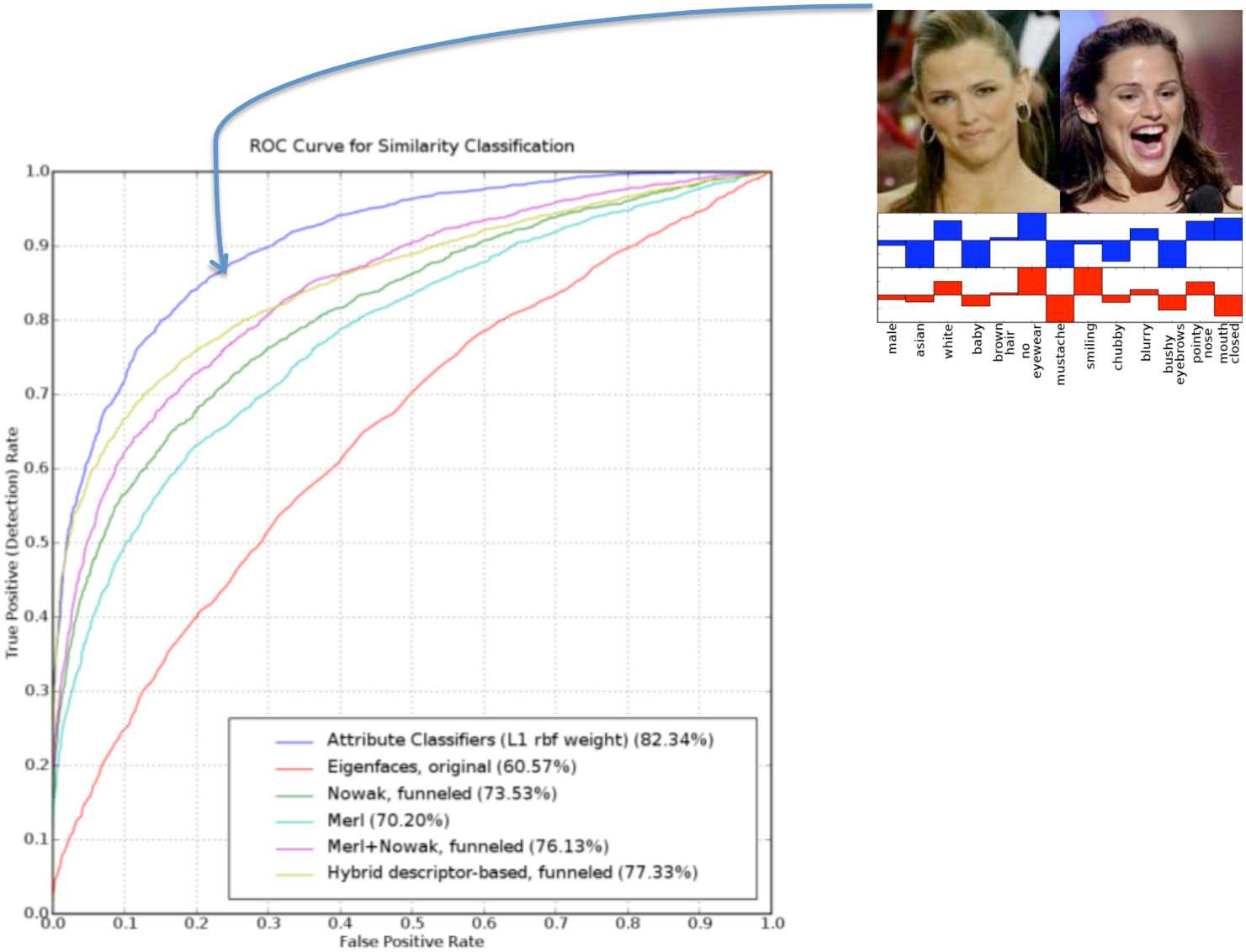


RGB, HSV,
Gradient,
Gradient Direction
Moments
Histograms
No normalization
Or l1 or “l2”
Various subparts of faces,
(requires alignment to a
single generic face)

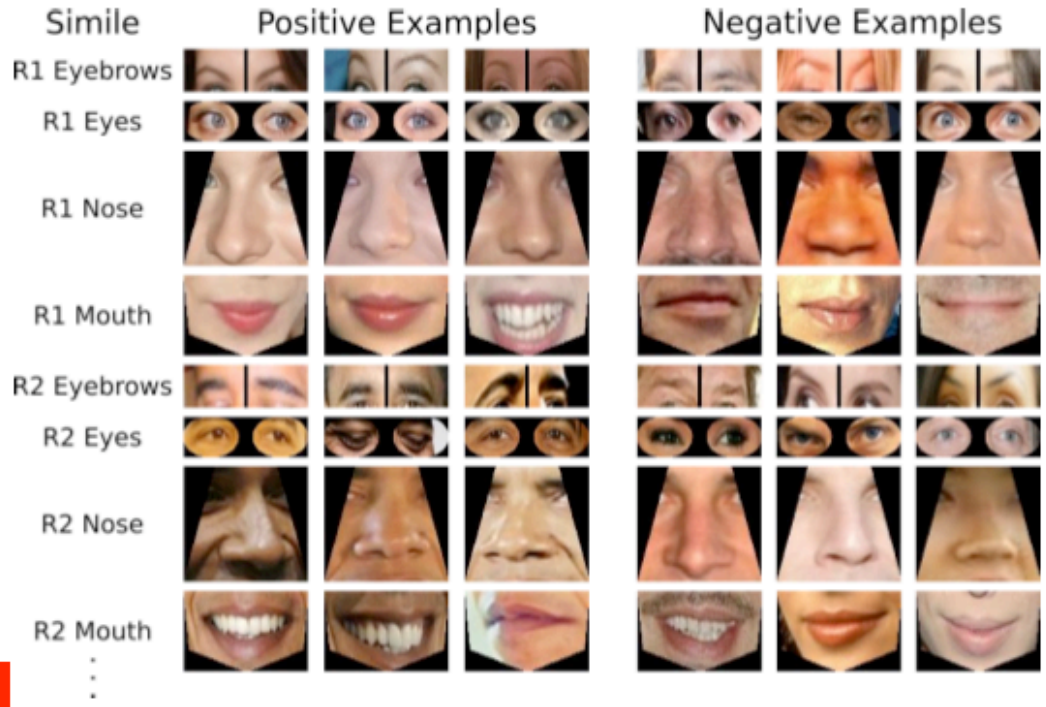
We look at attributes of the faces that may be **robust** with respect to Identity, but are also **common** to many people
So that we can collect very large training sets for each attribute.

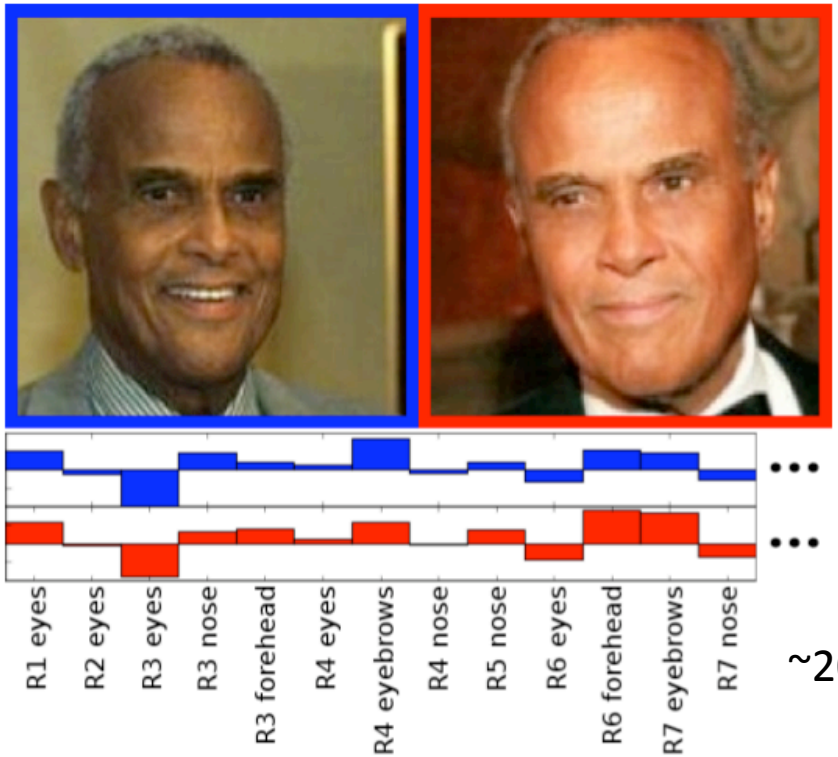


Some Training Data Collected Using Amazon's Mechanical Turk

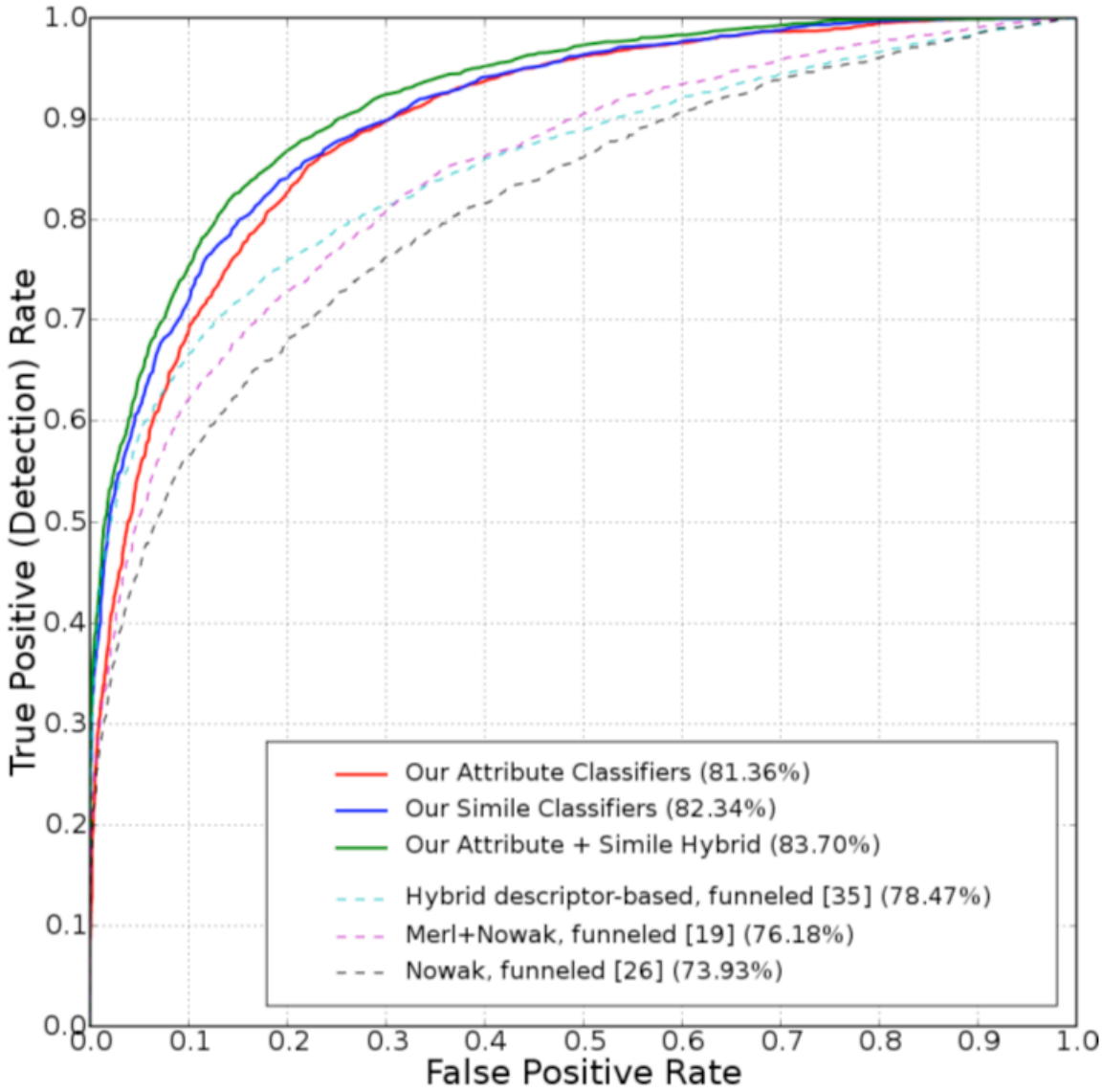

ROC Curve for Similarity Classification
True Positive (Detection) Rate
False Positive Rate
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
Attribute Classifiers (L1 rbf weight) (82.34%)
Eigenfaces, original (60.57%)
Nowak, funneled (73.53%)
Merl (70.20%)
Merl+Nowak, funneled (76.13%)
Hybrid descriptor-based, funneled (77.33%)
male
asian
white
baby
brown hair
no eyewear
mustache
smiling
chubby
blurry
bushy eyebrows
pointy nose
mouth closed

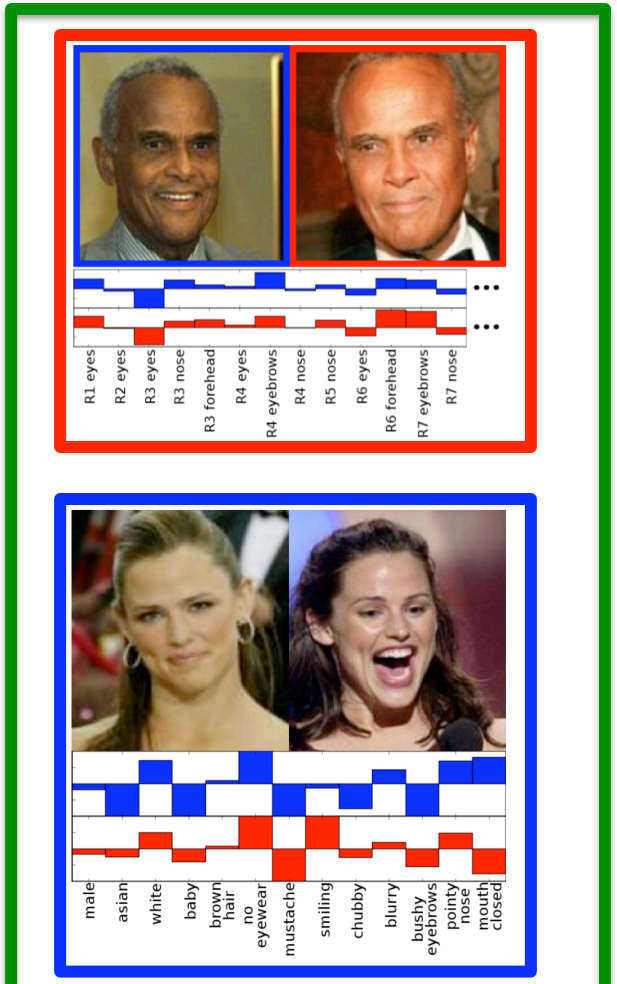
An individual person may have distinctive features (eg eyes)
So learn a classifier to recognize their eyes, (eg “she had Bette Davis Eyes”)



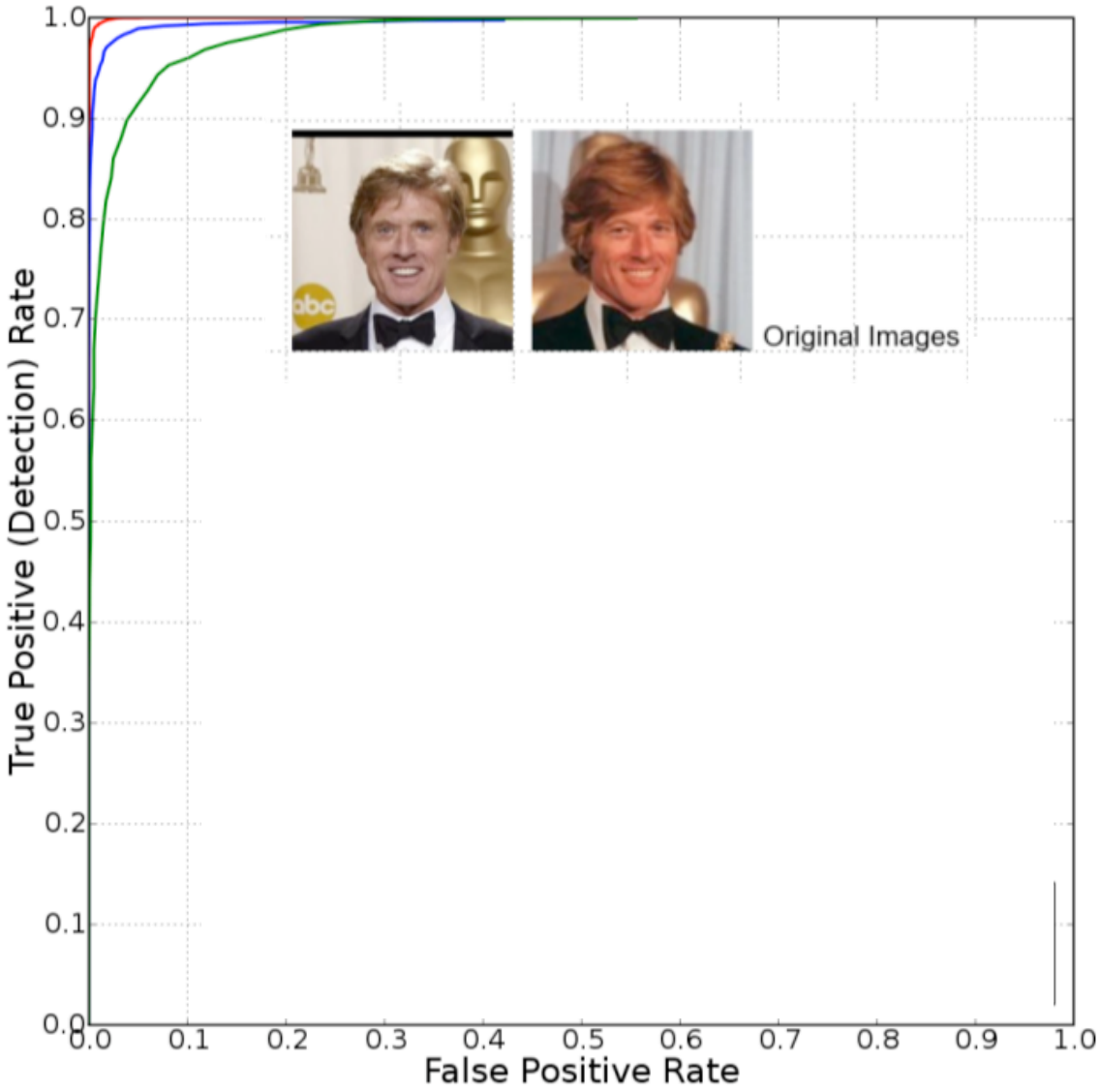


~2000…

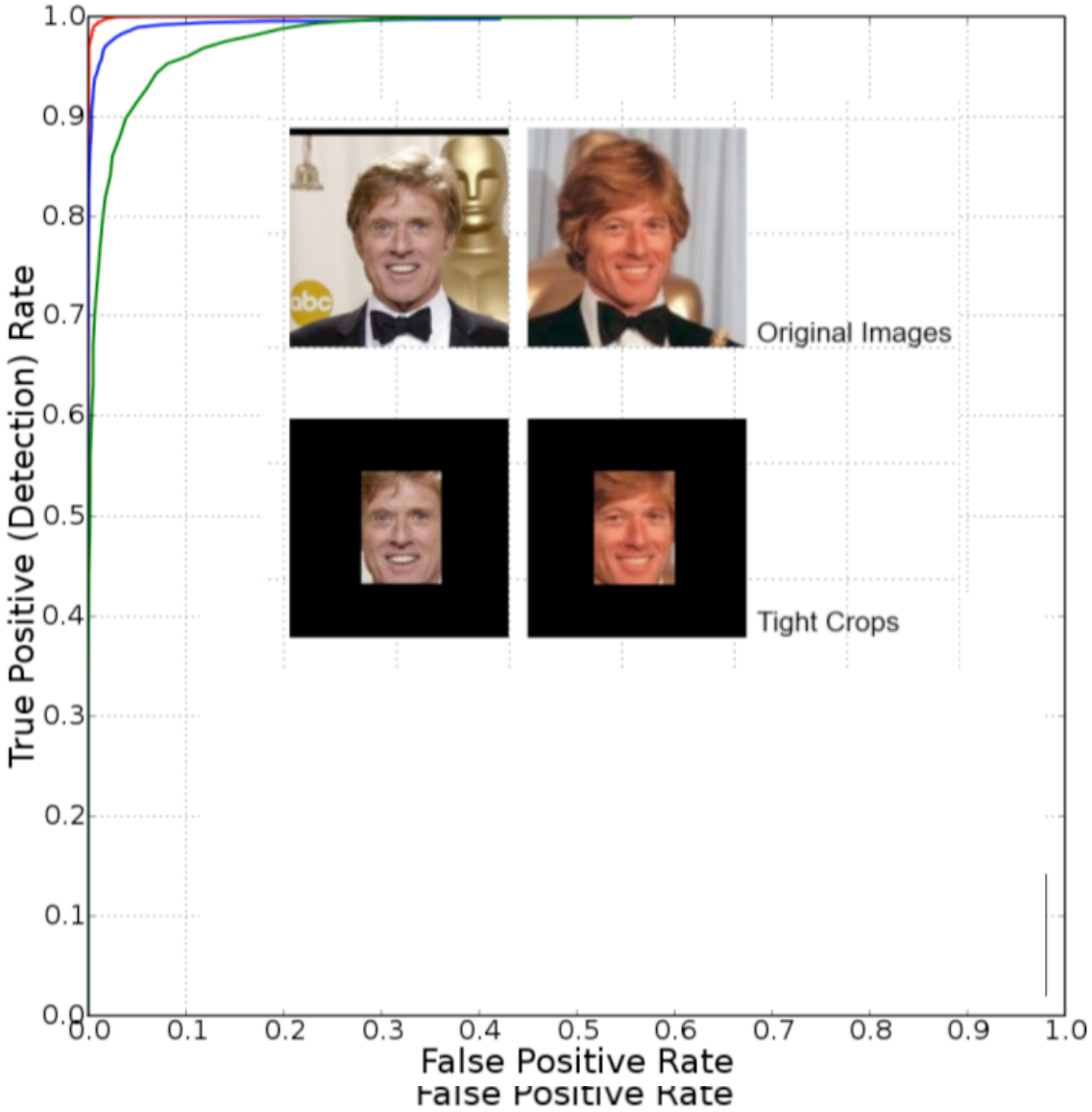

True Positive (Detection) Rate
False Positive Rate
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
Our Attribute Classifiers (81.36%)
Our Simile Classifiers (82.34%)
Our Attribute + Simile Hybrid (83.70%)
Hybrid descriptor-based, funneled [35] (78.47%)
Merl+Nowak, funneled [19] (76.18%)
Nowak, funneled [26] (73.93%)


R1 eyes
R2 eyes
R3 eyes
R3 nose
R3 forehead
R4 eyes
R4 eyebrows
R4 nose
R5 nose
R6 eyes
R6 forehead
R7 eyebrows
R7 nose
male
asian
white
baby
brown hair
no eyewear
mustache
smiling
chubby
blurry
bushy eyebrows
pointy nose
mouth closed

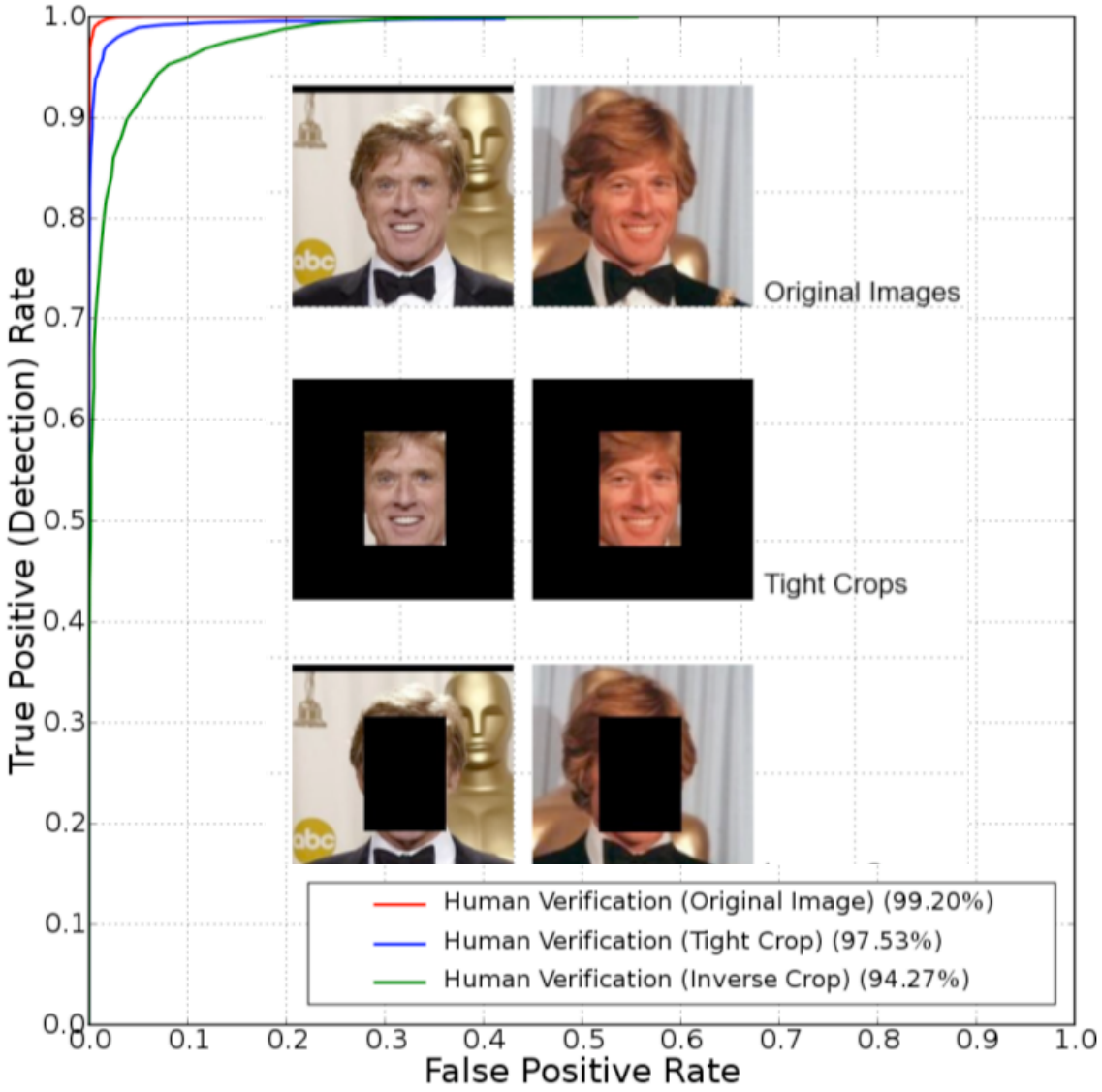
Humans are really good!



Humans are really good!



Humans are really good!
They don't even need to see the face!!!!



Other algorithms have access to all this background and we do substantially better looking only at the face…

But there is a long way to go to human performance even on a tight crop of the face….

New Large Dataset of Celebrities

This portion was used to train the similarity classifiers…

In total, over 200 people with 100+ images of each

Related To “Straightening Manifolds”

It is difficult to collect many example images of particular people,
but easy to collect millions of examples of Asians, or smiling people, etc
with huge variation including pose and settings.

The similarity of a face to any of these groups defines a simplified
coordinate for its position in the space of all faces.

Deformable Templates – Discrete Optimization

A.C. Berg, T.L. Berg, J. Malik
CVPR 05

Integer Quadratic Programming

$$\text{cost}(x) = \sum_{ij} c_{ij} x_{ij} + \sum_{ij,kl} H_{ij,kl} x_{ij} x_{kl}$$

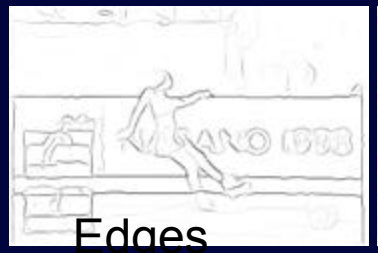
Binary indicator vector:
$X_{ij} = 1$ iff i matches to j

Appearance:
cost of matching
two local features
(geometric blur)

Geometry:
cost of matching
two pairs of features

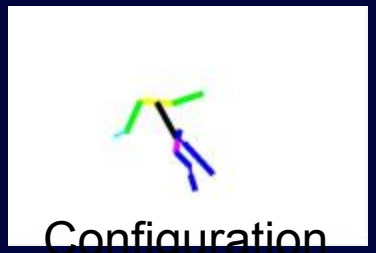
Keep quadratic framework, but use parallel edges as local features

X. Ren, A.C. Berg, J. Malik
ICCV 05

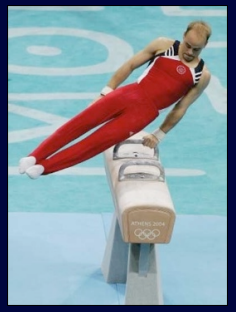


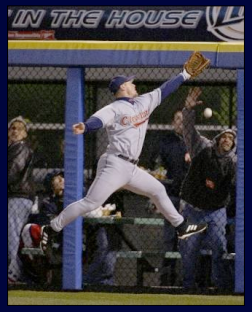
Edges

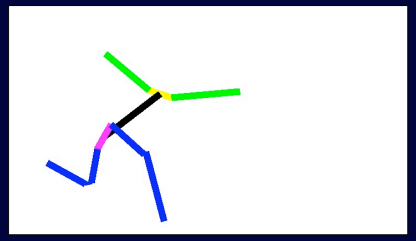
Parallel Edges

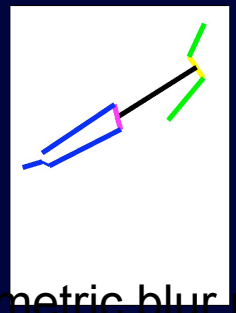


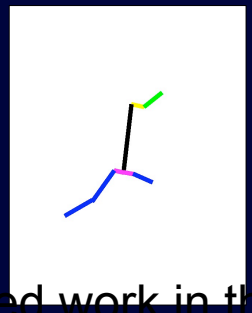
Configuration

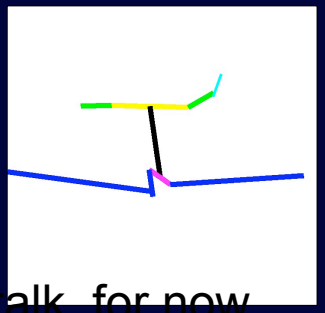










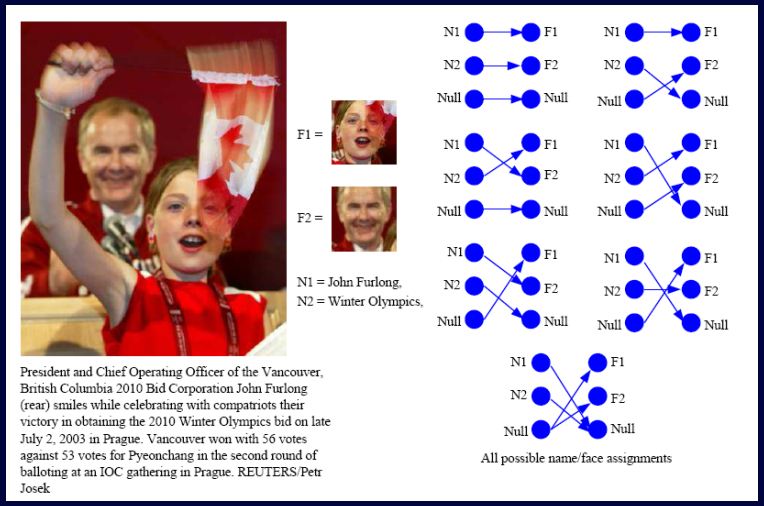


Only completely non-geometric blur related work in the talk, for now…

Keep patch features (geometric blur), but modify geometry model

T.L Berg, A.C. Berg, J. Edwards, D.A. Forsyth
NIPS 04

Align faces before representation



President and Chief Operating Officer of the Vancouver, British Columbia 2010 Bid Corporation John Furlong (rear) smiles while celebrating with compatriots their victory in obtaining the 2010 Winter Olympics bid on late July 2, 2003 in Prague. Vancouver won with 56 votes against 53 votes for Pyeonchang in the second round of balloting at an IOC gathering in Prague. REUTERS/Petr Josek

Cluster using face appearance and names from captions, get a very large data set of faces automatically

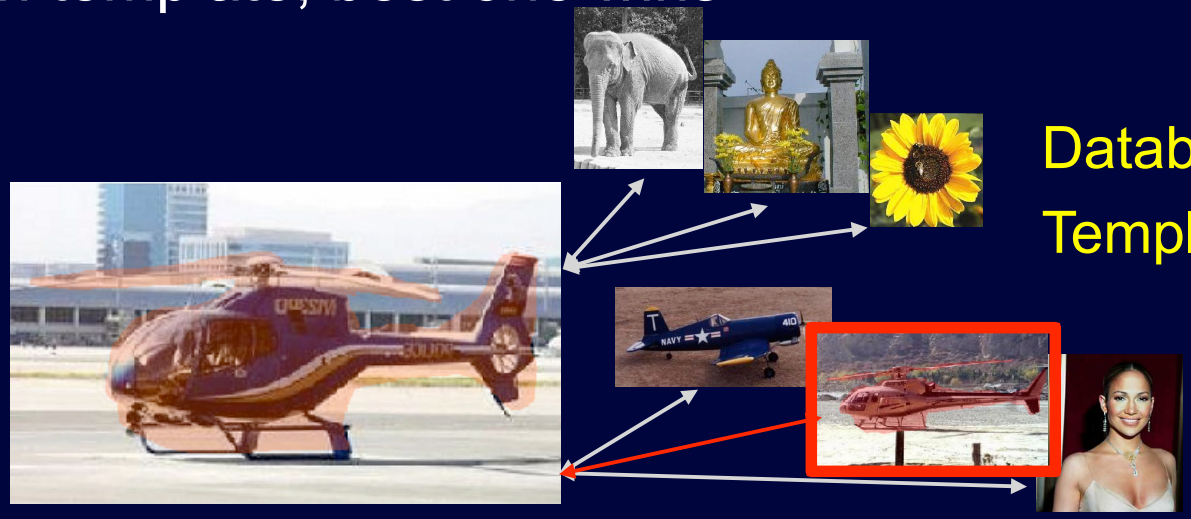
90% (or more) accurate

Deformable Template Matching with Exemplars for Recognition

- Use exemplars as deformable templates
- Find a correspondence between the query image and each template, best one wins

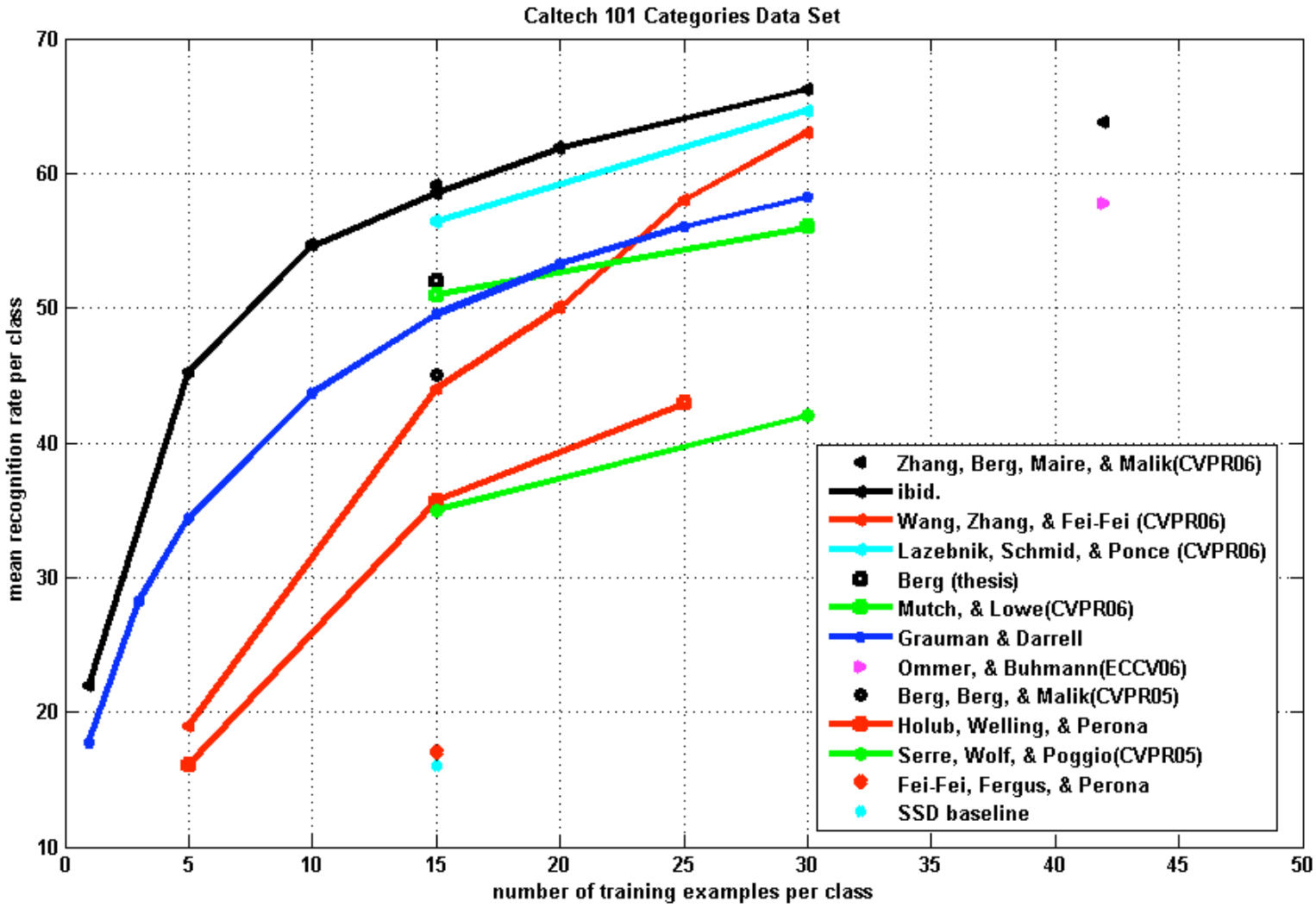
Deformable Template Matching with Exemplars for Recognition

- Use exemplars as deformable templates
- Find a correspondence between the query image and each template, best one wins

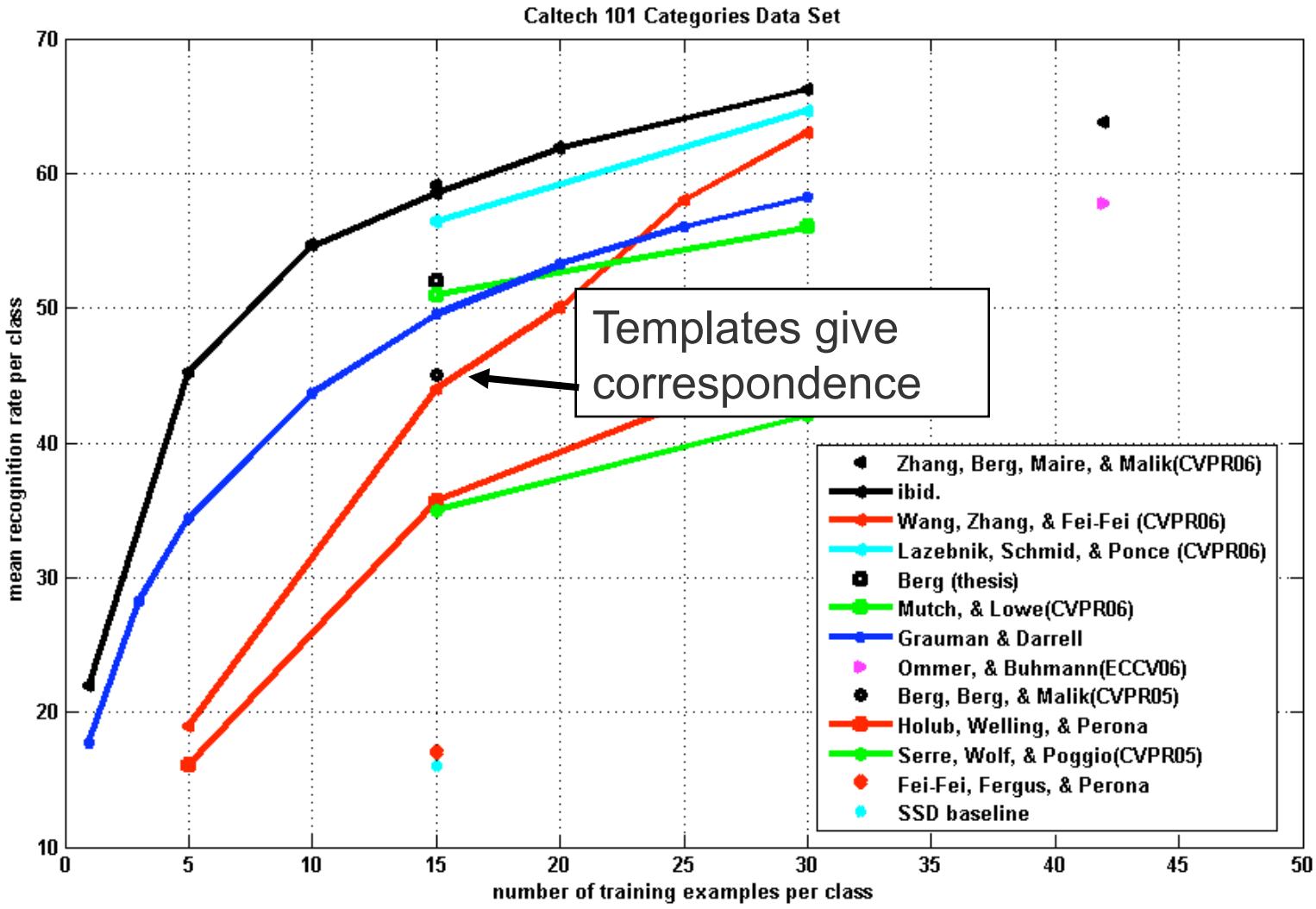


Best matching template is a helicopter

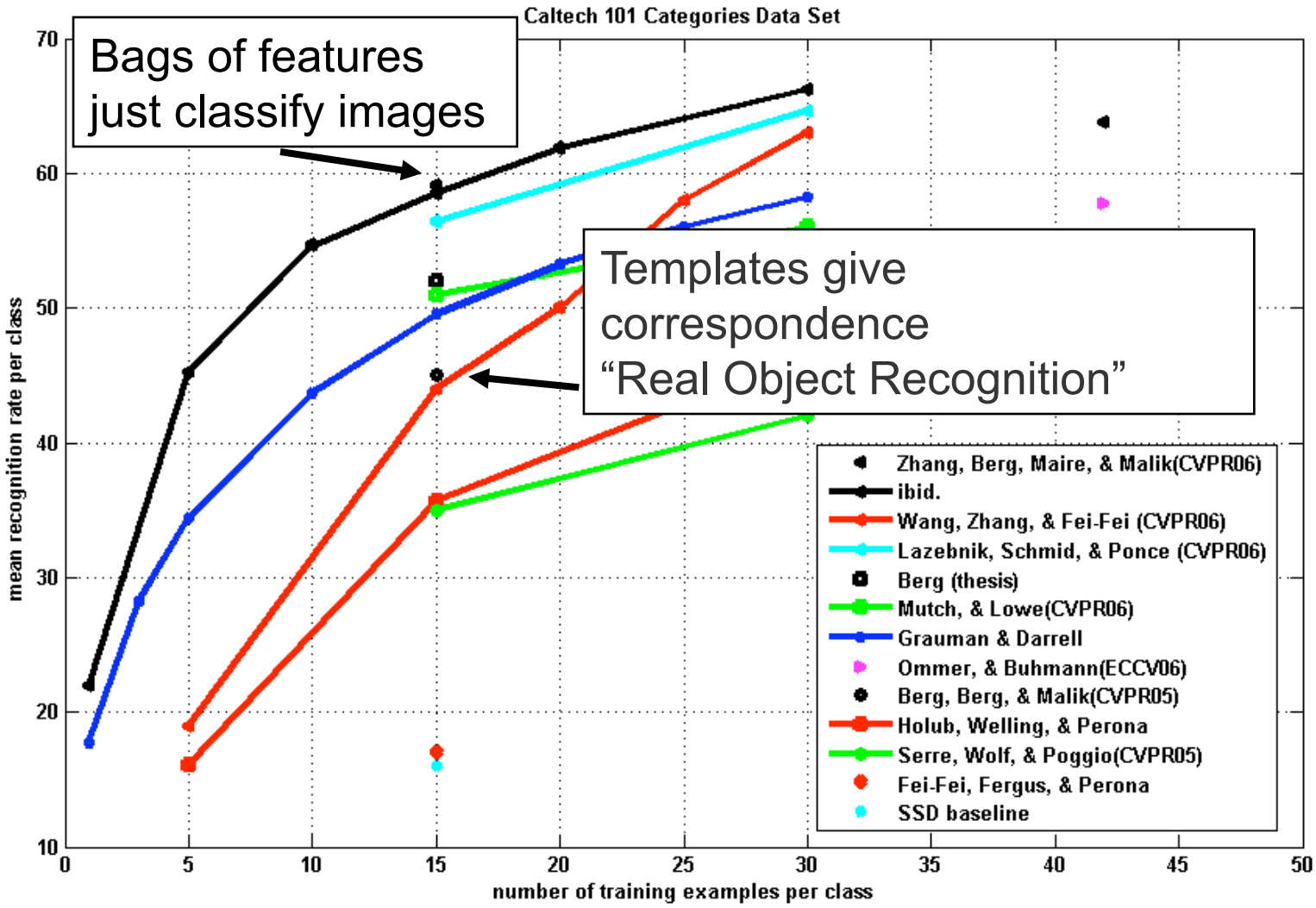
Many Results on Caltech-101



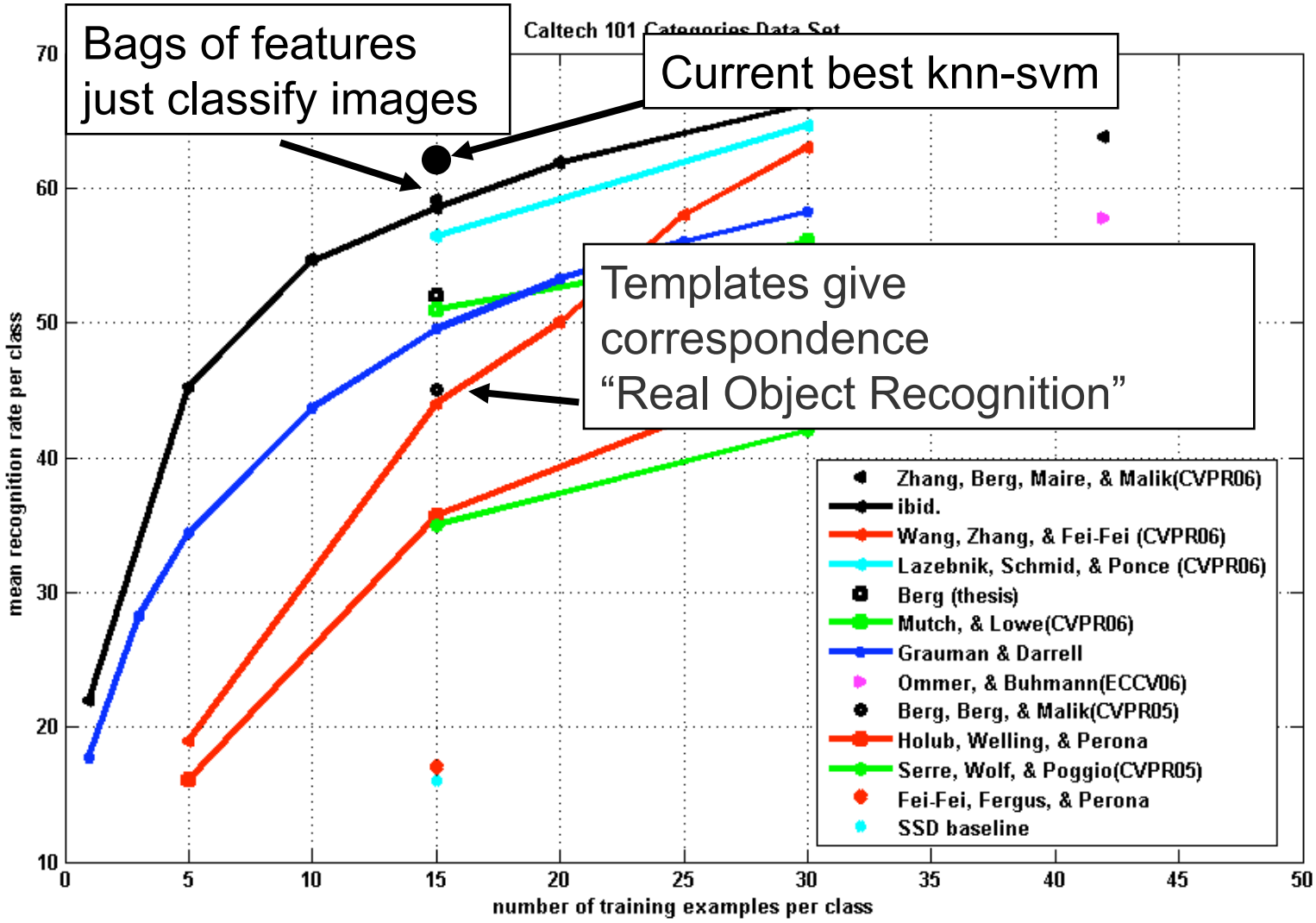
Many Results on Caltech-101



Many Results on Caltech-101



Many Results on Caltech-101



Deconstruction...

To classify images, forget about the objects,
just match features in the whole image.

geometric blur descriptors
A.C. Berg J. Malik CVPR 2001

- 1 NN classifier for each feature, most votes wins → 54%

  No Geometry Berg Thesis 2005

- Bag of features using max-margin learning for weights 60%

  No Geometry Frome Singer Malik NIPS 2006

- Bag of features using svm → 60%, **knn-svm → 62%**

  Rough position in image Zhang, Berg, Maire, Malik CVPR 2006

- Doesn't matter how you use geometry

  Spatial pyramid matching kernel Lazebnik et al → 59%

Deformable Templates – Discrete Optimization

A.C. Berg, T.L. Berg, J. Malik
CVPR 05

Integer Quadratic Programming

$$\text{cost}(x) = \sum_{ij} c_{ij} x_{ij} + \sum_{ij,kl} H_{ij,kl} x_{ij} x_{kl}$$

Binary indicator vector:
$X_{ij} = 1$ iff i matches to j

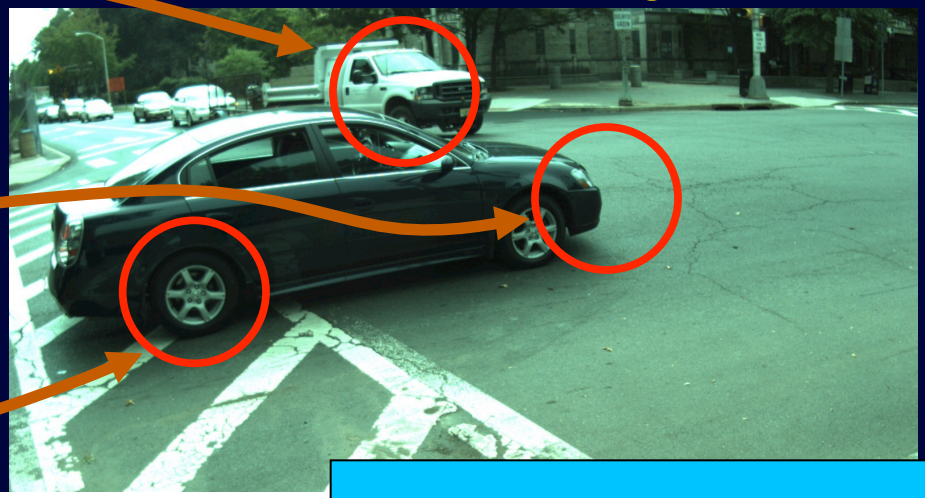
Appearance:
cost of matching
two local features
(geometric blur)

Geometry:
cost of matching
two pairs of features

Drop the Templates –> EASY Optimization

A.C. Berg, T.L. Berg, J. Malik
CVPR 05

Image



Model of Car

$$\text{cost}(x) = \sum_{ij} c_{ij} x_{ij} +$$

Easy Linear Programming

Binary indicator vector:
X_{ij} = 1 iff *i* matches to *j*

Appearance:
cost of matching two local features
(geometric blur)

No Geometry 54-60%

Add back in a little geoemtry & knn-svm

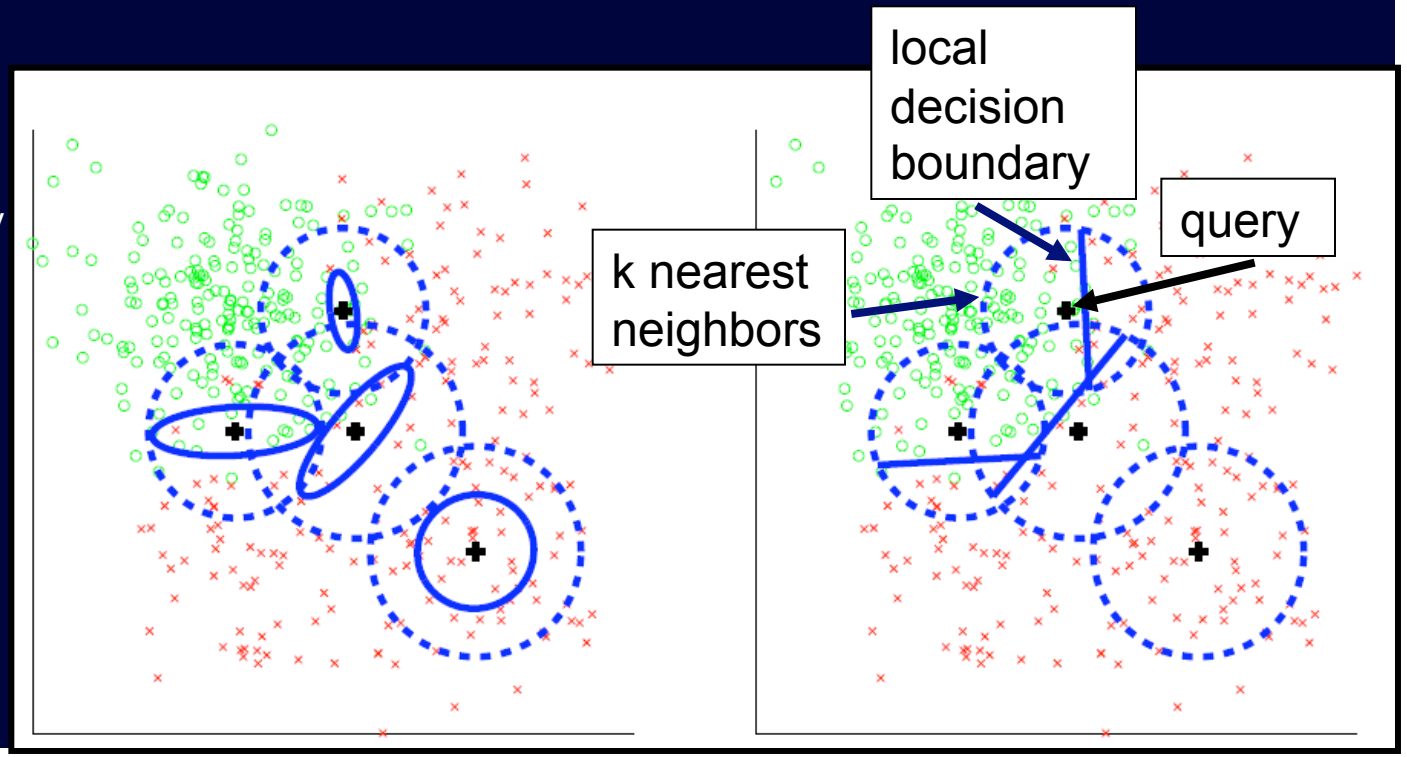
$$D^B(I_L \to I_R) = \frac{1}{m}\sum_{i=1}^{m} \min_{j=1..n}\left[\|F_i^L - F_j^R\|^2 + \frac{\lambda}{r_0}\|r_i^L - r_j^R\|\right]$$

$$D^B(I_L, I_R) = D^B(I_L \to I_R) + D^B(I_R \to I_L)$$

H. Zhang, A.C. Berg, M. Maire, J. Malik
CVPR 2006

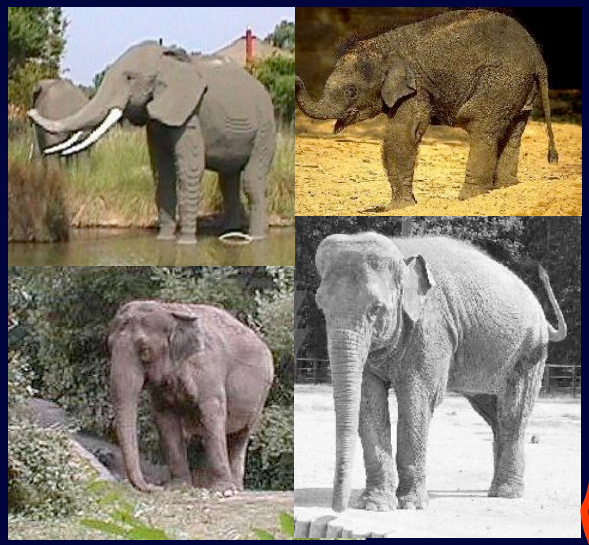
Basically append position to the feature vector + an svm gives 60%
+knn-svm gives another 2%

- Classify whole image
- Use geometric blur to model appearance similarity
- Match to approximately the right part of the image
- Use k-nearest neighbors to train a svm / query
- Get the current best results on *image classification*



Samples from Caltech-101
Split into Animals & Non Animals

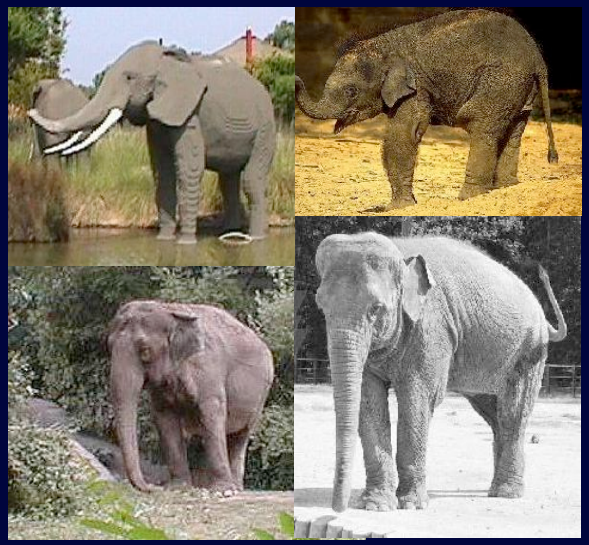
Since we are recognizing the whole image anyway



Samples from Caltech-101
Split into Animals & Non Animals

Classification rate
~90% correct

Same range as humans! (Thorpe et al but…



tired of looking at the same 9144 images?

Beyond Caltech 101 – TRECVID

-- Hundreds of thousands of key frames from shots in the TRECVID dataset

-- Broadcast video in English (US), Chinese, Arabic

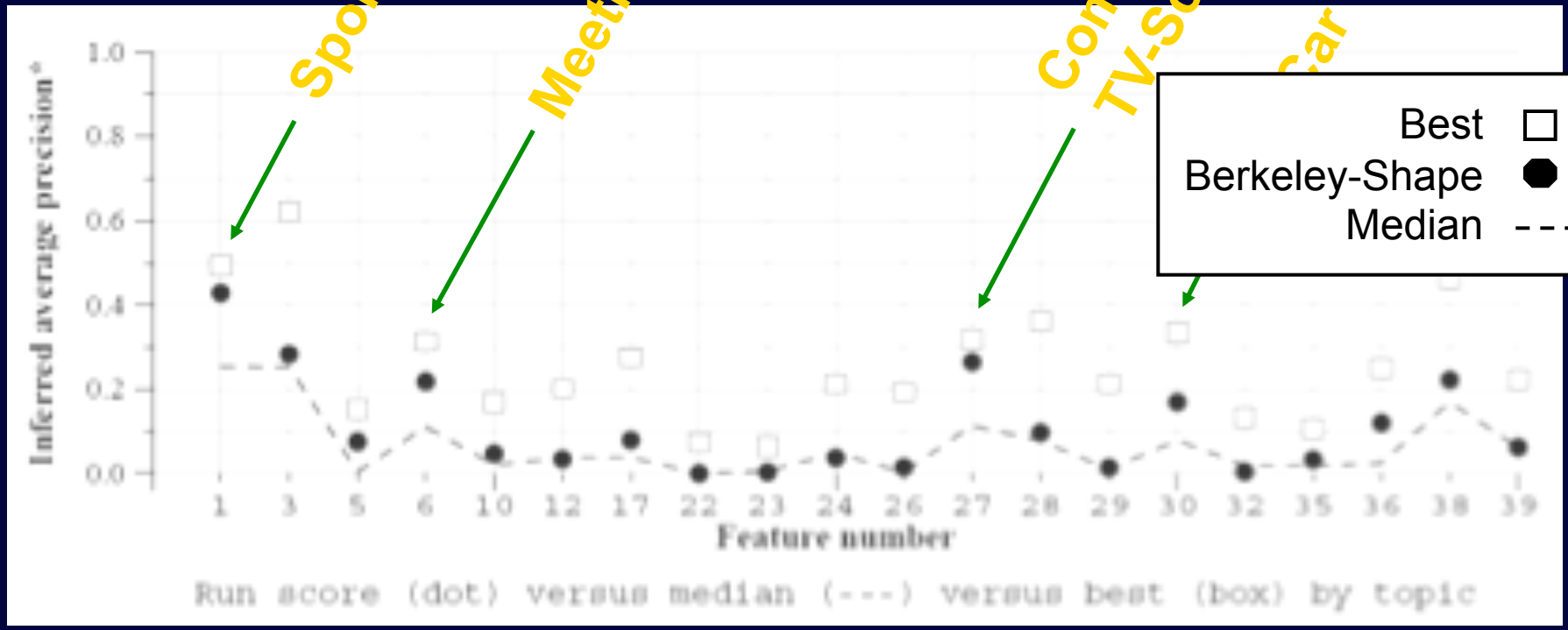
-- Training data labeled with >= 40 labels
weather, sports, person, car, business leader, etc.

-- Goal is to rank new shots by whether they contains these labels....

Slav Petrov, Arlo Faria, Pascal Michaillat, Alexander Berg,
Andreas Stolcke, Dan Klein, Jitendra Malik

TRECVID Results ’06

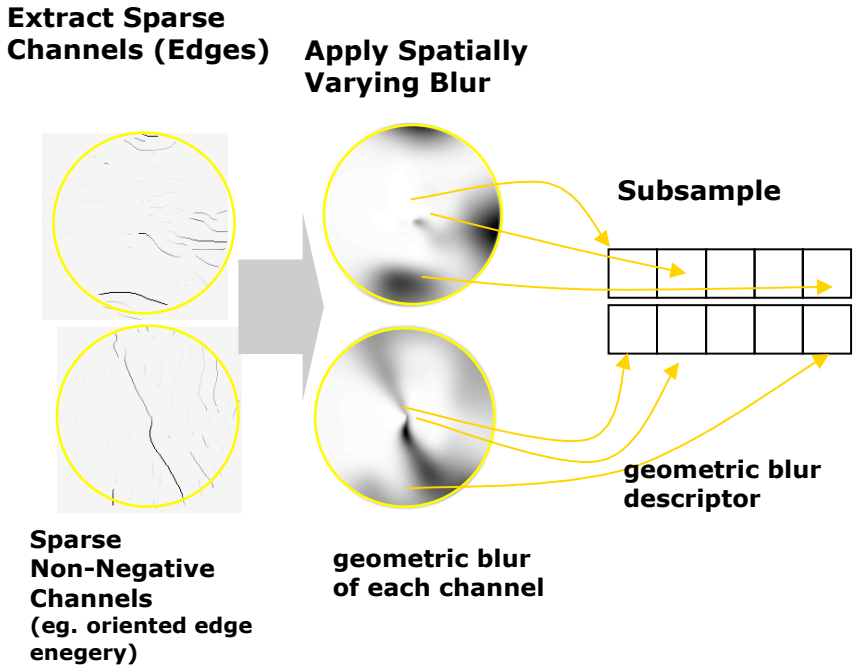
Results ’05
Berkeley-Shape mAP = 0.38
Best ’05 (IBM) mAP = 0.34



mAP = 0.11

Evaluating Similar Appearance

geometric blur based descriptor



Berg, Berg, Malik CVPR 2005

Berg, Malik CVPR 2001

$$G_I(x) = \int_{T \in \mathcal{T}} I(T(x))dT$$

-- Works well for recognition tasks

-- Berg Thesis 05 has theoretical and ecological validation

-- Can be comparable performance to SIFT on wide baseline matching

Scenes different than objects.

- Scenes are made up of stuff not things
- Often not distinguishable locally: sky vs wall vs street
- Template style geometric models may be too rigid for scenes
- Scene specific previous work
  - Contextual Guidance of Attention in Natural scenes: The role of Global features on object search

    Torralba, Oliva, Castelhano & Henderson Psychological Review 2006

    Object dependent saliency maps
  - Integrated Models for Scenes Objects and Parts

    Sudderth, Torralba, Freeman & Wilsky NIPS , ICCV 2005 and others

    Local Features, simple rigorous models for their arrangement
  - Geometric Context from a Single Image

    Hoiem, Efros, Hebert IJCV 2006, and others
- **So how bad is it anyway?**

Quantifying “Badness”

"When you can measure what you are speaking about and express it in numbers, you know something about it; but when you cannot measure it, when you cannot express it in numbers, your knowledge is of the meager and unsatisfactory kind."

--Lord Kelvin

1. Hand Label Sky, Foliage, Building, Street

2. Split into training and test
3. Compute features
4. Build classifiers / perform regression
5. Evaluate how much the features and classifiers tell about the labels

Relative Mutual Information

Mutual Information

$$R(X;Y) = \frac{I(X;Y)}{H(Y)}$$

$$= \frac{H(X) + H(Y) - H(X,Y)}{H(Y)}$$

Entropy

An old problem

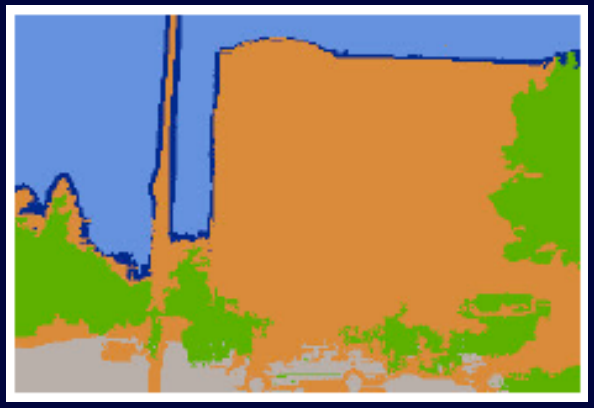
I stand at the window and see a house, trees, sky. Theoretically I might say there were 327 brightnesses and nuances of colour. Do I have “327”? No. I have sky, house, and trees.

– Max Wertheimer 1923

An old problem

I stand at the window and see a house, trees, sky. Theoretically I might say there were 327 brightnesses and nuances of colour. Do I have “327”? No. I have sky, house, and trees.

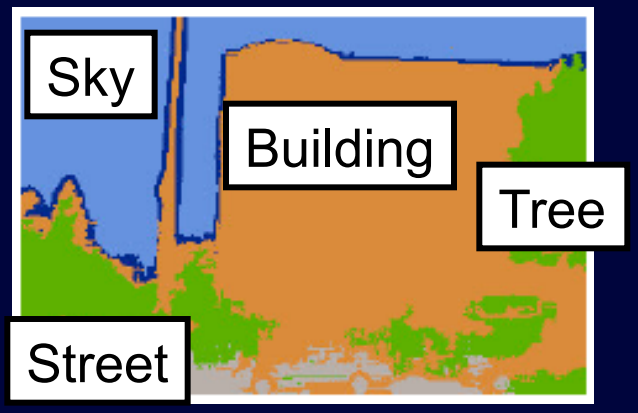
– Max Wertheimer 1923



An old problem

I stand at the window and see a house, trees, sky. Theoretically I might say there were 327 brightnesses and nuances of colour. Do I have “327”? No. I have sky, house, and trees.

– Max Wertheimer 1923

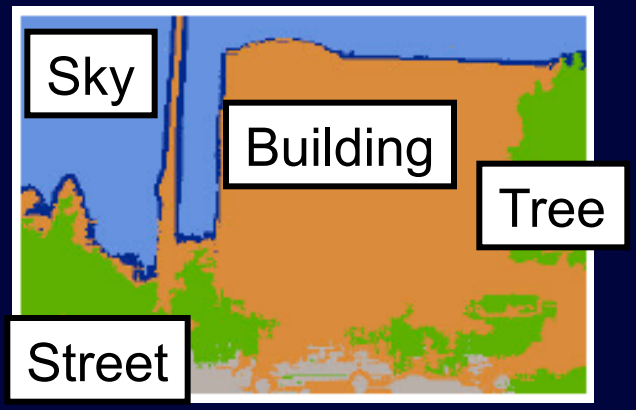


An old problem

I stand at the window and see a house, trees, sky. Theoretically I might say there were 327 brightnesses and nuances of colour. Do I have “327”? No. I have sky, house, and trees.

– Max Wertheimer 1923

Use this coarse parsing for
more detailed parsing of buildings

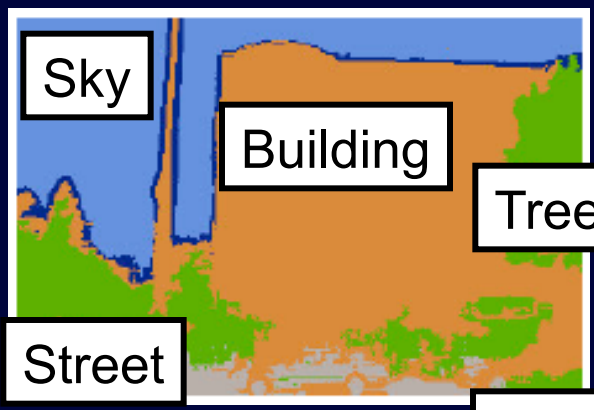


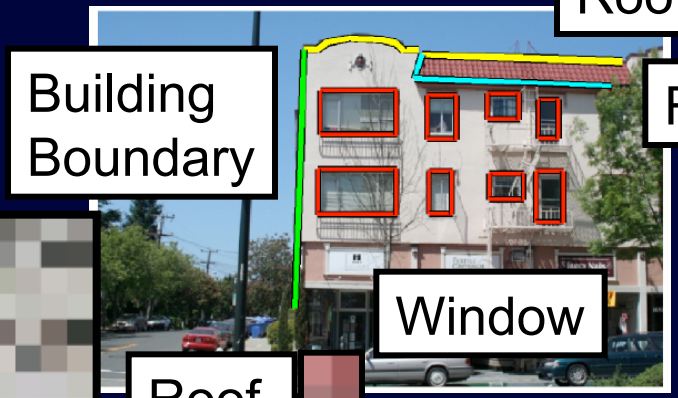
An old problem

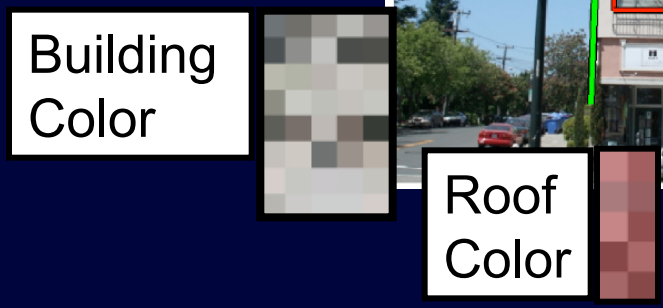
I stand at the window and see a house, trees, sky. Theoretically I might say there were 327 brightnesses and nuances of colour. Do I have “327”? No. I have sky, house, and trees.

– Max Wertheimer 1923

Use this coarse parsing for more detailed parsing of buildings





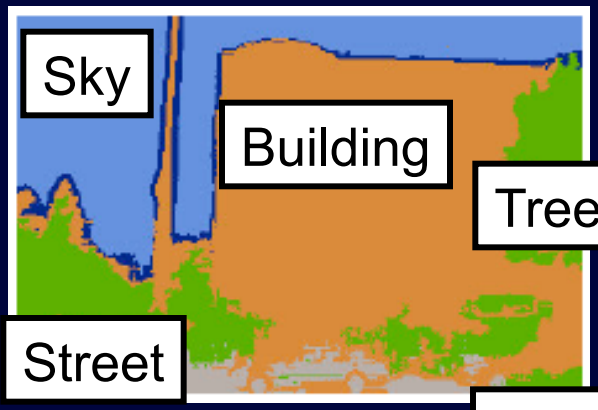


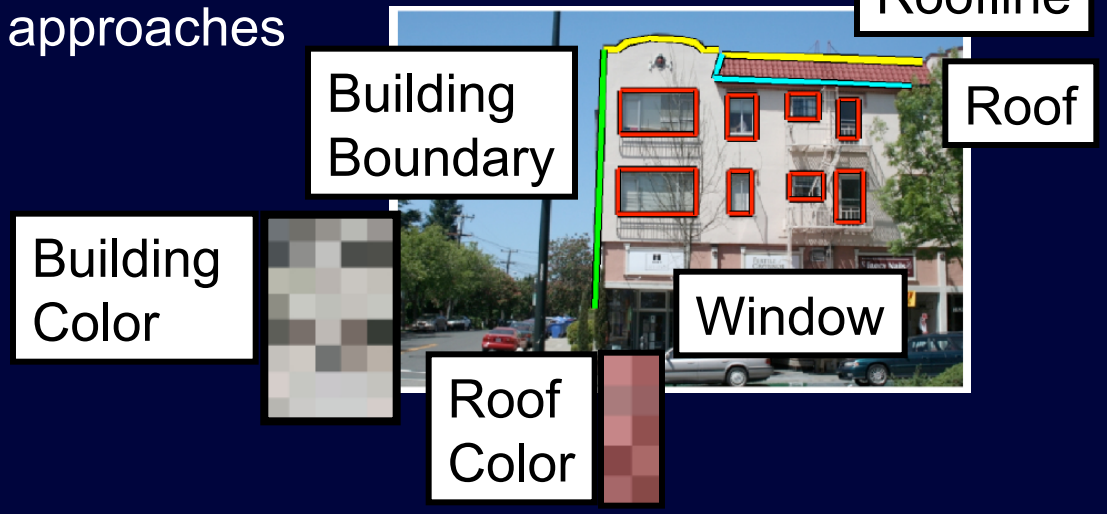
Why Architectural Scenes?

- Make up decent portion of our surroundings

- Microsoft, Amazon, etc. are collecting and trying to use a great deal of this type of data, anything automatic is helpful

- Stress current computational approaches to visual recognition...

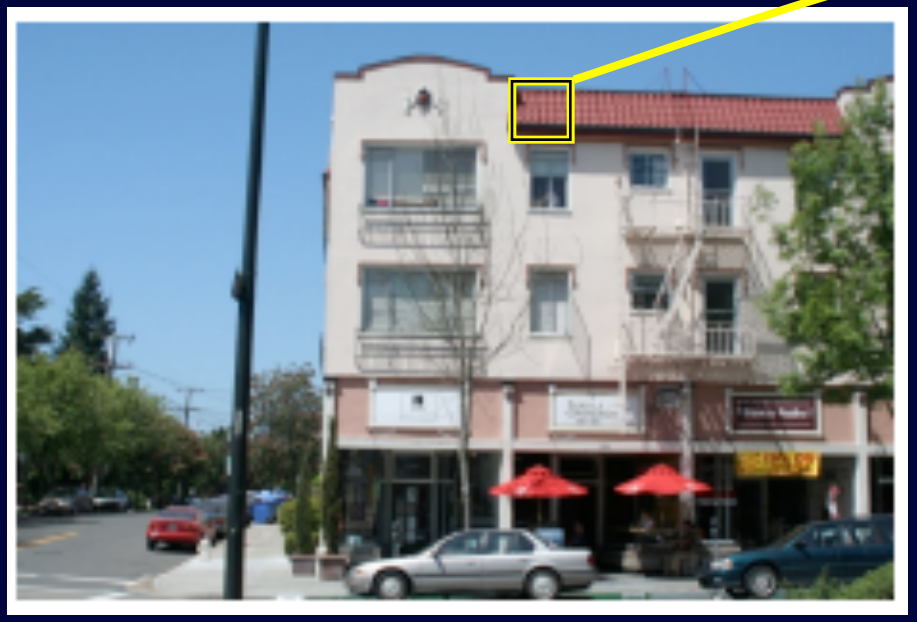




Our Method

Work with:
Floraine Grabler ETH Zurich,
Jitendra Malik U.C. Berkeley

Our Method : Features in a Patch



-Color of central pixel
-Color histogram
-Total edge energy
-Oriented edge energy
-Height in Image
-Lengths/Orientations of contours

KNN Density estimates → prob. of label (sky, tree, etc.) given feature(s)
except
2 SVMs for central pixel color and edge energy, one for sky one for trees

Our Method : Features in a Patch

Most likely category

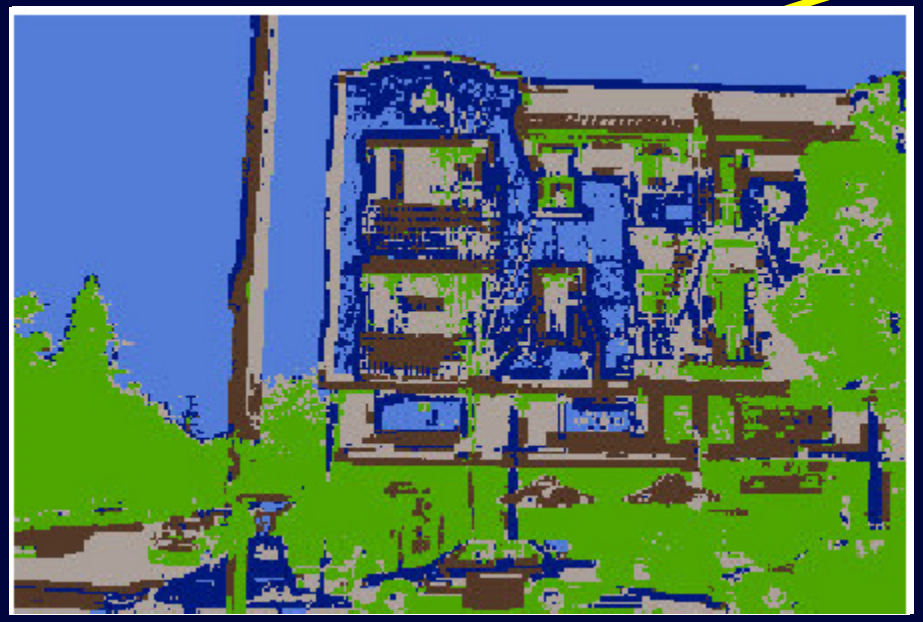
- **Building**
- **Tree/Folliage**
- **Street**
- **Sky**
- **Sky Mixed**

- **-Color of central pixel**
- -Color histogram
- -Total edge energy
- -Oriented edge energy
- -Height in Image
- -Lengths/Orientations of contours

KNN Density estimates → prob. of label (sky, tree, etc.) given feature

Our Method : Features in a Patch

Most likely category



-Color of central pixel
-Color histogram
-Total edge energy
-Oriented edge energy
-Height in Image
-Lengths/Orientations of contours

Building
Tree/Folliage
Street
Sky
Sky Mixed

KNN Density estimates → prob. of label (sky, tree, etc.) given feature

Our Method : Features in a Patch

Most likely category

- **Color of central pixel**
- Color histogram
- Total edge energy
- Oriented edge energy
- Height in Image
- Lengths/Orientations of contours

Building
Tree/Folliage
Street
Sky
Sky Mixed

KNN Density estimates → prob. of label (sky, tree, etc.) given feature

Our Method : Features in a Patch

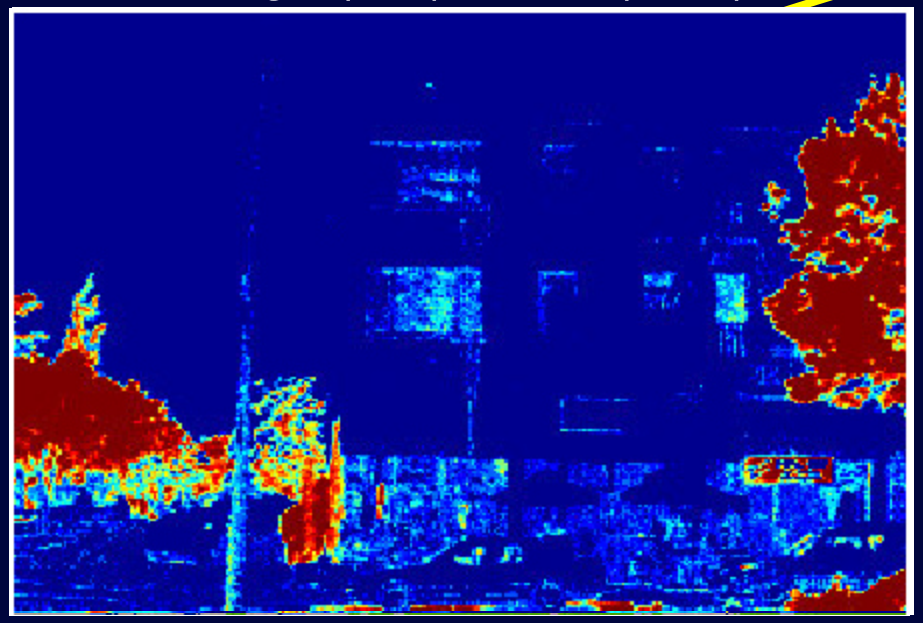
Sky (red) or not (blue)

-Color of central pixel
-Color histogram
-Total edge energy
-Oriented edge energy
-Height in Image
-Lengths/Orientations of contours

SVM for central pixel color and edge energy, to predict sky

Our Method : Features in a Patch

Foliage (red) or not (blue)



-Color of central pixel

-Color histogram

-Total edge energy

-Oriented edge energy

-Height in Image

-Lengths/Orientations of contours

SVM for central pixel color and edge energy, to predict foliage

Our Method : Features in a Patch

Most likely category

Building
Tree/Folliage
Street
Sky
Sky Mixed

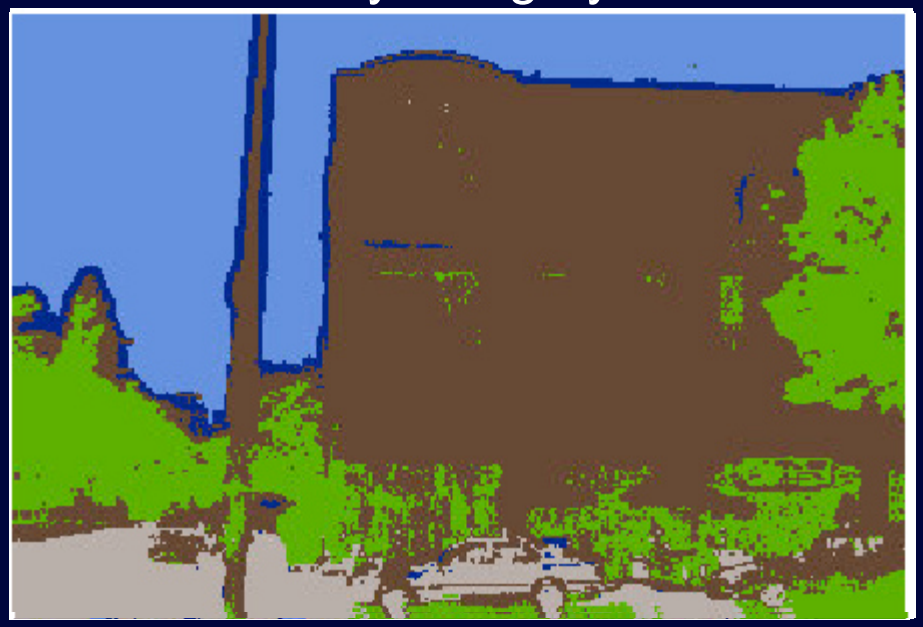
-Color of central pixel
-Color histogram
-Total edge energy
-Oriented edge energy
-Height in Image
-Lengths/Orientations of contours

Combination

Still some confusion, after all the building is street colored, So use a first pass of a detailed parse to make a building and sky model for this image…

Our Method : Features in a Patch++

Most likely category



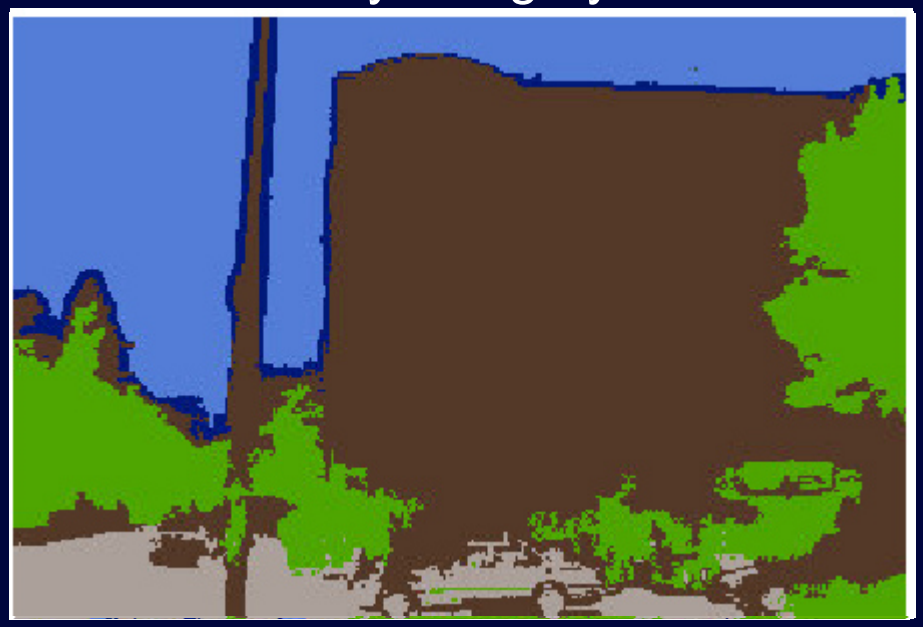
Building
Tree/Folliage
Street
Sky
Sky Mixed

-**Color of central pixel**
-Color histogram
-Total edge energy
-Oriented edge energy
-Height in Image
-Lengths/Orientations of contours

Image specific model

Our Method : Features in a Patch++

Most likely category



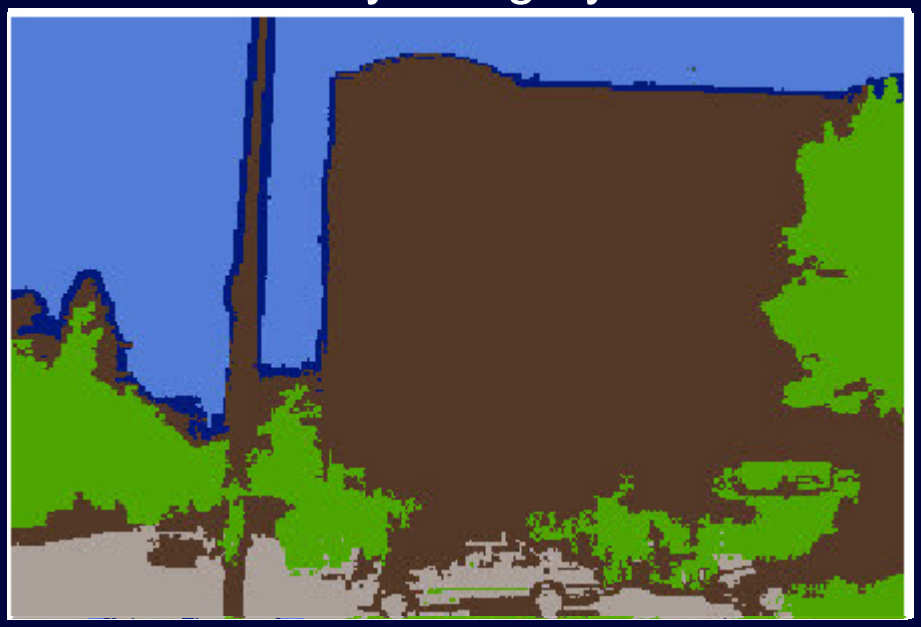
Building
Tree/Folliage
Street
Sky
Sky Mixed

-**Color of central pixel**
-Color histogram
-Total edge energy
-Oriented edge energy
-Height in Image
-Lengths/Orientations of contours

Image specific model
With some spatial smoothing
(driven by training data)

Our Method : Details

Most likely category



Building
Tree/Folliage
Street
Sky
Sky Mixed

This rough parsing helps find
-features defined by the parsing
-rooflines, sides of buildings
-power lines and dead trees
-windows and doors
-building color

Our Method : Detailed Parsing

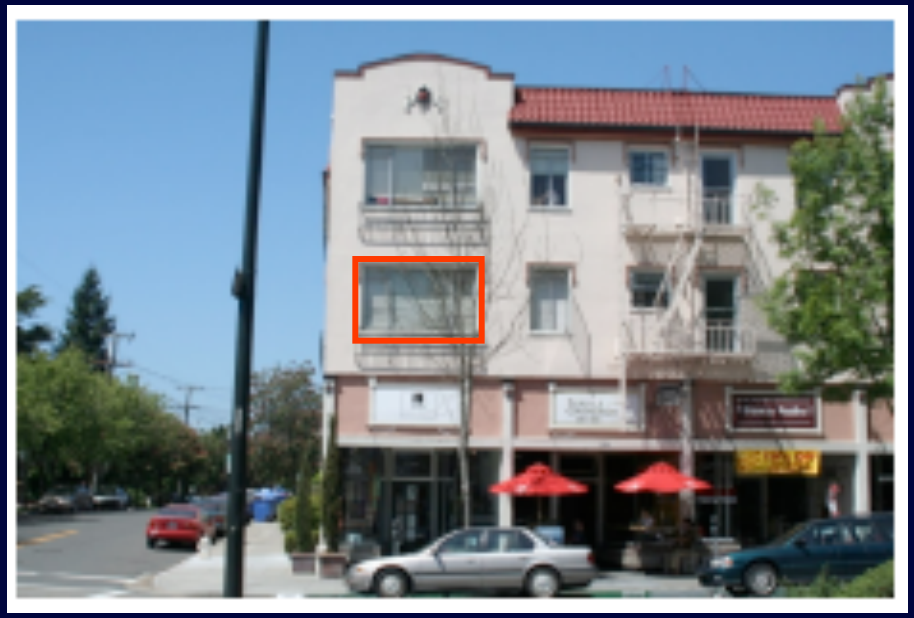
using windows as an example

Evaluate a hypotheses about window location and size by:

-The size and aspect ratio
-The surrounding rough labels
-The estimated building color

Our Method : Detailed Parsing

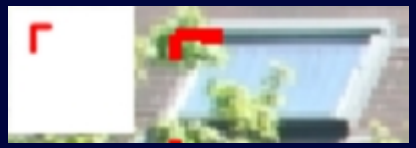
using windows as an example



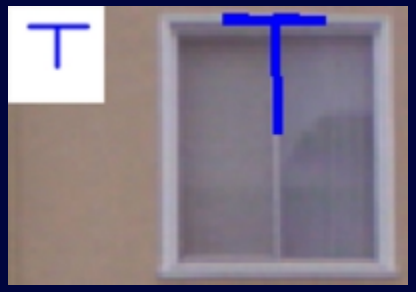
Evaluate a hypotheses about window location and size by:

-The size and aspect ratio
-The surrounding rough labels
-The estimated building color

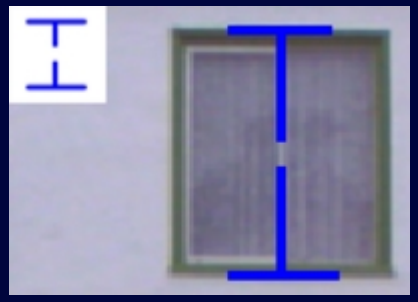
Finding Windows

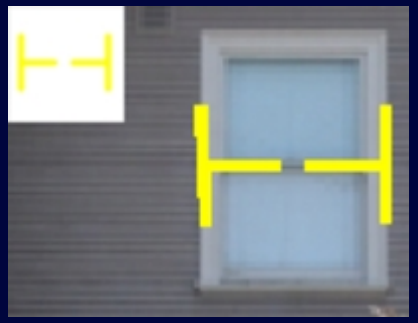
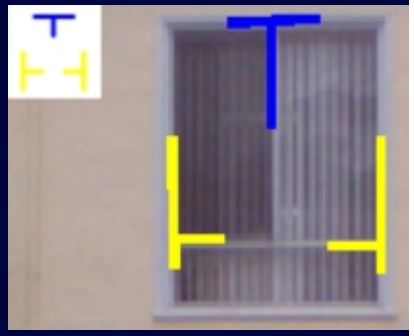



Various ways to form window hypotheses
Combine with model of building color and spatial context

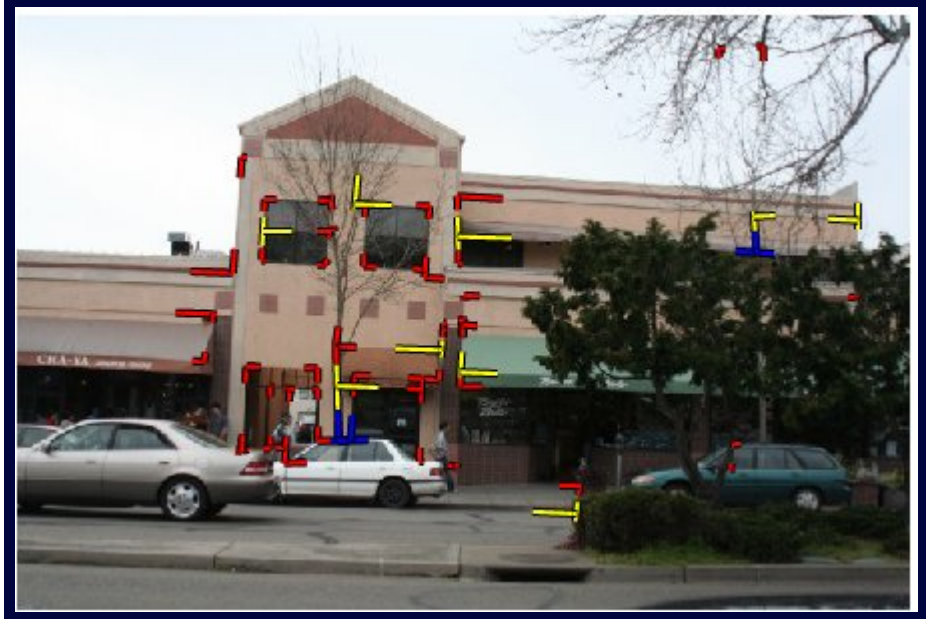




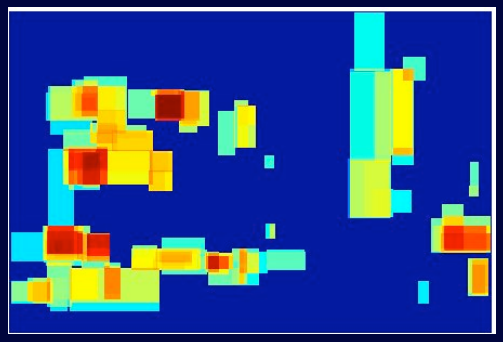


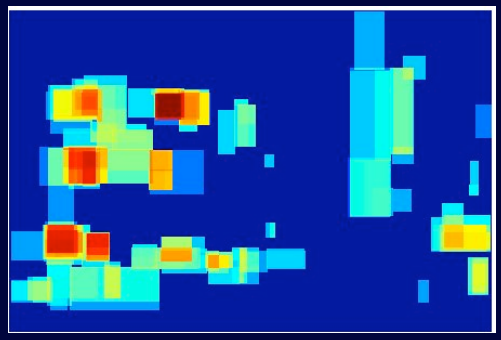
Finding Windows



Various ways to form window hypotheses
-- Hypotheses are reinforced by regularity
-- Windows should be under rooflines



Without configuration cue



With configuration cue

Look at many results…

How bad is it?

Simple patch features tell us about as much about the coarse scale parsing as the geometric context work.

Table 1. Relative Information Gain

	Sky	Foliage	Building	Street
Hoiem *et al.*				
por	0.07	0.12	0.03	0.06
sol	0.06	0.01	0.01	0.04
090	0.41	0.10	0.08	0.23
000	0.12	0.04	0.02	**0.38**
sky	0.73	0.08	0.08	0.09
This Work				
avg. texture	0.59	0.24	0.05	0.11
contours	0.56	0.23	0.04	0.07
color	0.55	0.22	0.04	0.03
tree svm	-	0.20	-	-
sky svm	0.63	-	-	-
generic	0.76	0.26	0.11	0.25
image specific	**0.77**	**0.29**	0.13	0.28
smoothing	0.75	0.29	**0.14**	0.27

Relative Mutual Information

Mutual Information

$$R(X;Y) = \frac{I(X;Y)}{H(Y)}$$

$$= \frac{H(X) + H(Y) - H(X,Y)}{H(Y)}$$

Entropy

How bad is it?

Simple patch features without segmentation tell us about as much about the coarse scale parsing as the output of the geometric context work, more for buildings and folliage.

Table 1. Relative Information Gain

	Sky	Foliage	Building	Street
Hoiem *et al.*				
por	0.07	0.12	0.03	0.06
sol	0.06	0.01	0.01	0.04
090	0.41	0.10	0.08	0.23
000	0.12	0.04	0.02	**0.38**
sky	0.73	0.08	0.08	0.09
This Work				
avg. texture	0.59	0.24	0.05	0.11
contours	0.56	0.23	0.04	0.07
color	0.55	0.22	0.04	0.03
tree svm	-	0.20	-	-
sky svm	0.63	-	-	-
generic	0.76	0.26	0.11	0.25
image specific	**0.77**	**0.29**	0.13	0.28
smoothing	0.75	0.29	**0.14**	0.27

Relative Mutual Information

Mutual Information

$$R(X;Y) = \frac{I(X;Y)}{H(Y)} = \frac{H(X) + H(Y) - H(X,Y)}{H(Y)}$$

Entropy

How bad is it?

Simple patch features (color histogram here) also give a fair amount of information about the geometric structure.

Simple patch features tell us about as much about the coarse scale parsing as the geometric context work.

Relative Mutual Information

Mutual Information

$$R(X;Y) = \frac{I(X;Y)}{H(Y)} = \frac{H(X)+H(Y)-H(X,Y)}{H(Y)}$$

Entropy

Table 2. Relative Information Gain on Hoeim et al Training/Testing

	sky	090	000
This Work			
color	0.62	0.26	0.19

Table 1. Relative Information Gain

	Sky	Foliage	Building	Street
Hoiem *et al.*				
por	0.07	0.12	0.03	0.06
sol	0.06	0.01	0.01	0.04
090	0.41	0.10	0.08	0.23
000	0.12	0.04	0.02	**0.38**
sky	0.73	0.08	0.08	0.09
This Work				
avg. texture	0.59	0.24	0.05	0.11
contours	0.56	0.23	0.04	0.07
color	0.55	0.22	0.04	0.03
tree svm	-	0.20	-	-
sky svm	0.63	-	-	-
generic	0.76	0.26	0.11	0.25
image specific	**0.77**	**0.29**	0.13	0.28
smoothing	0.75	0.29	**0.14**	0.27

How bad is it?

Simple patch features (color histogram here) also give a fair amount of information about the geometric structure.

Simple patch features tell us about as much about the coarse scale parsing as the geometric context work.

Table 2. Relative Information Gain on Hoeim et al Training/Testing

	sky	090	000
This Work			
color	0.62	0.26	0.19
geometric blur	0.28	0.26	0.36

Table 1. Relative Information Gain

	Sky	Foliage	Building	Street
Hoiem *et al.*				
por	0.07	0.12	0.03	0.06
sol	0.06	0.01	0.01	0.04
090	0.41	0.10	0.08	0.23
000	0.12	0.04	0.02	**0.38**
sky	0.73	0.08	0.08	0.09
This Work				
avg. texture	0.59	0.24	0.05	0.11
contours	0.56	0.23	0.04	0.07
color	0.55	0.22	0.04	0.03
tree svm	-	0.20	-	-
sky svm	0.63	-	-	-
generic	0.76	0.26	0.11	0.25
image specific	**0.77**	**0.29**	0.13	0.28
smoothing	0.75	0.29	**0.14**	0.27

Relative Mutual Information

Mutual Information

$$R(X;Y) = \frac{I(X;Y)}{H(Y)} = \frac{H(X)+H(Y)-H(X,Y)}{H(Y)}$$

Entropy

Where to Next?

- Forget being clever about local smoothing
  - Small number of global parameters

    eg. Sky color, building color, color constancy
- Soft Representation for spatial layout
- Geometry
  - Surfaces?
  - Occluding boundaries (rooflines, building sides) are "still" important we need to harness this
- Segmentation **should** help, need to find out how
- Thank you.

Thank you

and my co-authors so far…

Alex Berg's coauthors (so far)