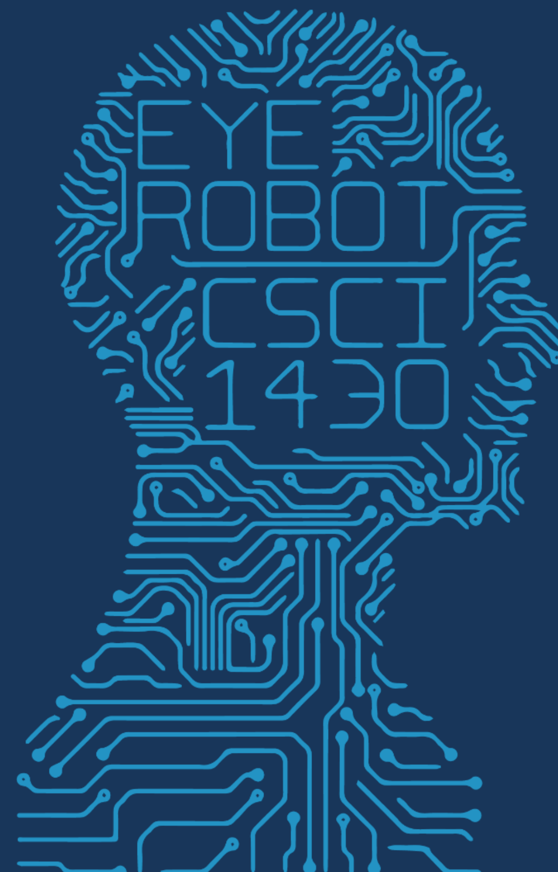
1950

Future Vision



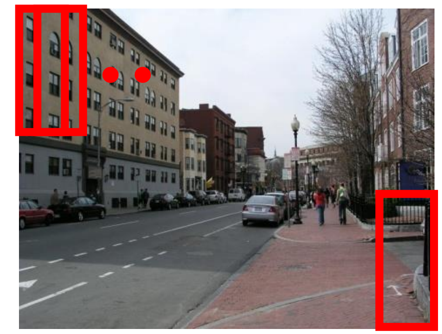
2017 MWF 1pm 368

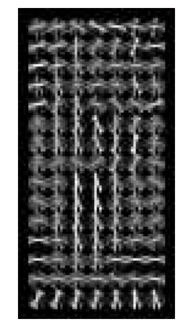
Computer Vision

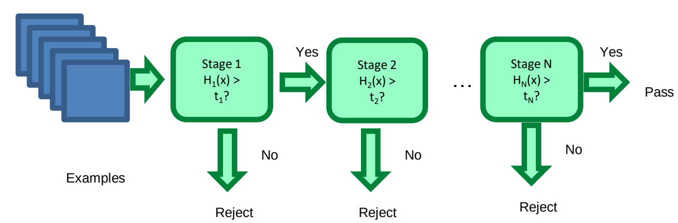

note:
black & white

Object detection

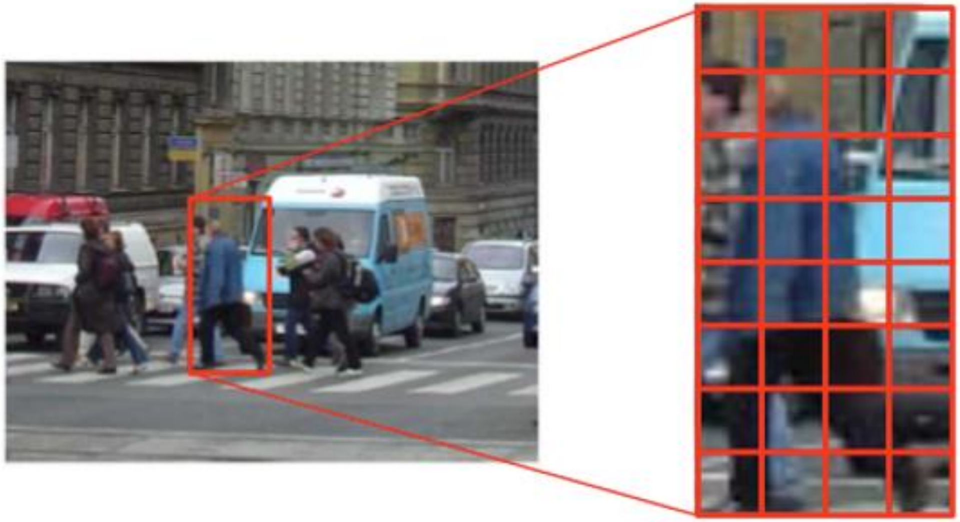
- Sliding window for search
- Features based on differences of intensity (gradient, wavelet, etc.)
- Boosting for feature selection
- Integral images, cascade for speed
- Bootstrapping to deal with many, many negative examples







Starting point: sliding window classifiers

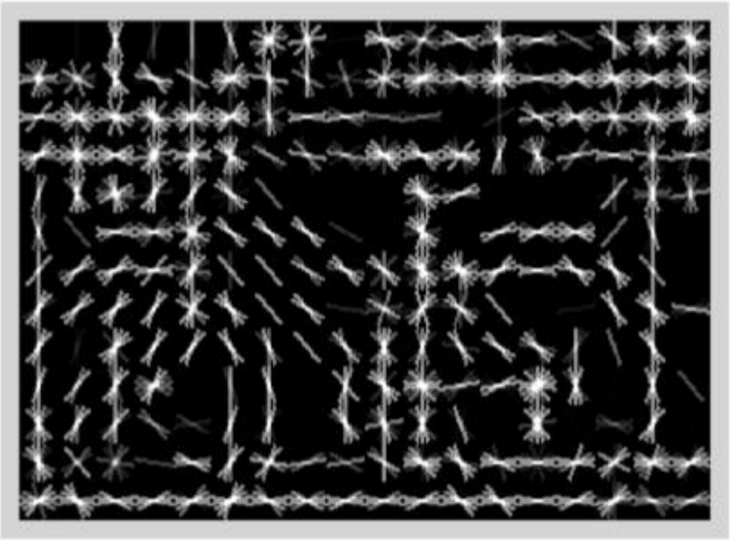


Feature vector
$x = [\ldots, \ldots, \ldots, \ldots]$

- Detect objects by testing each subwindow
  - Reduces object detection to binary classification
  - Dalal & Triggs: HOG features + linear SVM classifier
  - Previous state of the art for detecting people

Felzenszwalb

Histogram of Gradient (HOG) features

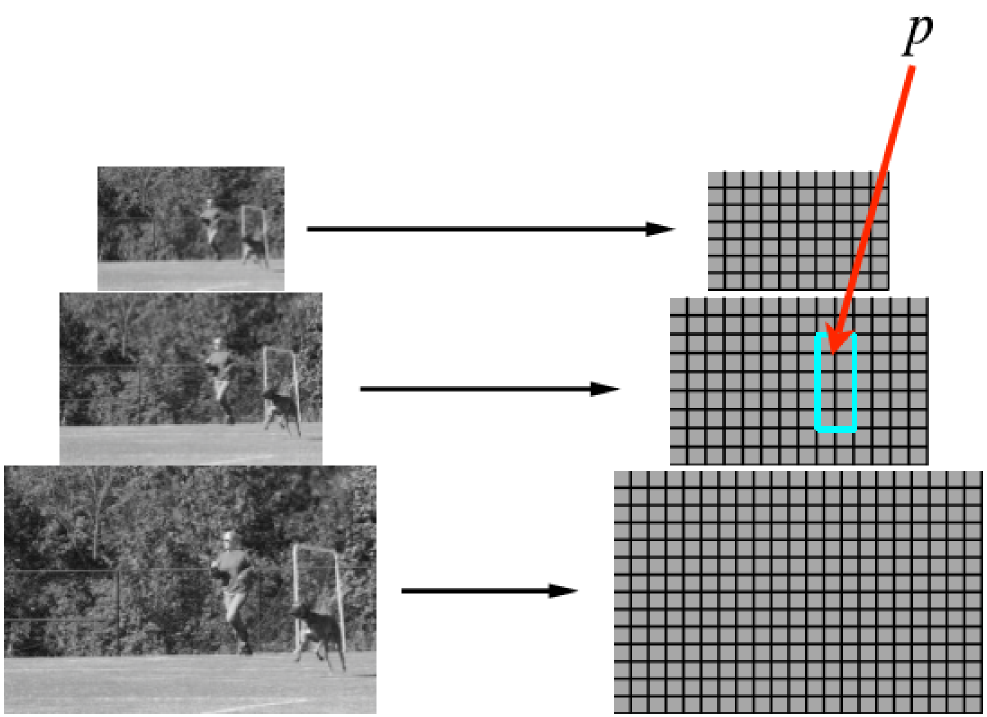


- Image is partitioned into 8x8 pixel blocks
- In each block we compute a histogram of gradient orientations
  - Invariant to changes in lighting, small deformations, etc.
- Compute features at different resolutions (pyramid)

Felzenszwalb

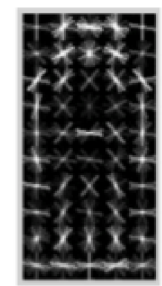
HOG Filters

- Array of weights for features in subwindow of HOG pyramid
- Score is dot product of filter and feature vector



HOG pyramid H

Filter F

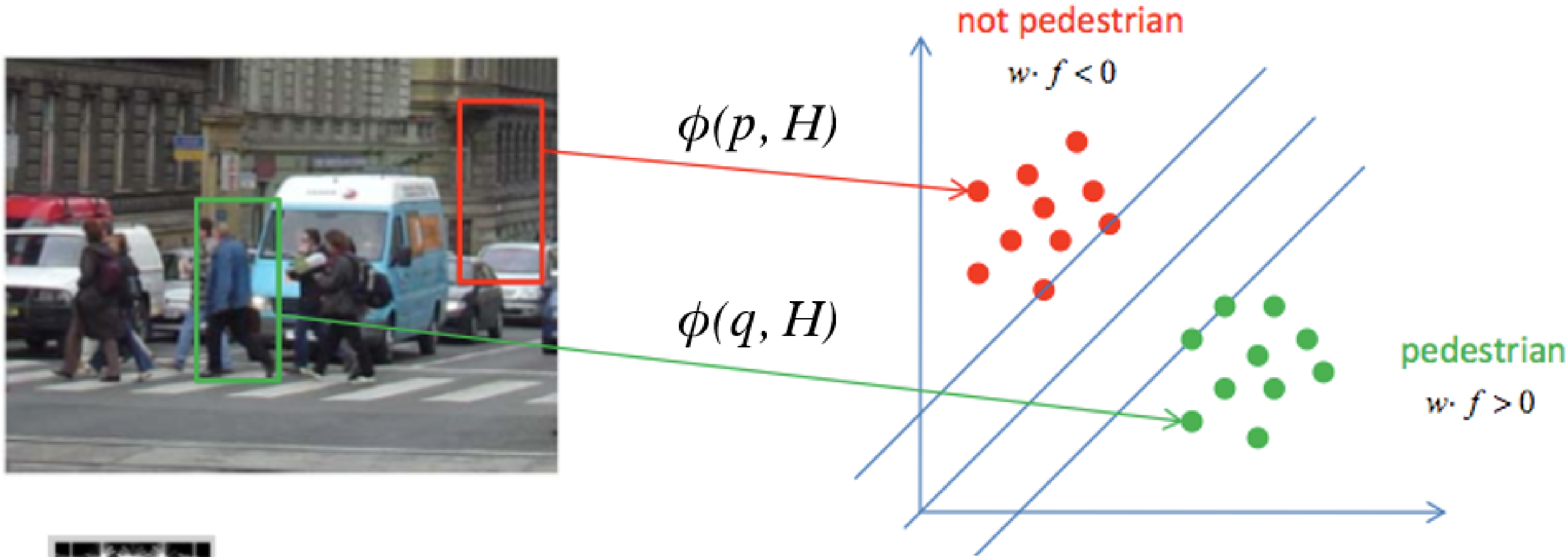


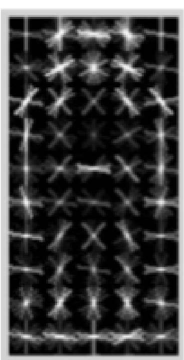
Score of F at position p is
$F \cdot \phi(p, H)$

$\phi(p, H)$ = concatenation of HOG features from subwindow specified by p

Felzenszwalb

Dalal & Triggs: HOG + linear SVMs





Typical form of
a model

There is much more background than objects

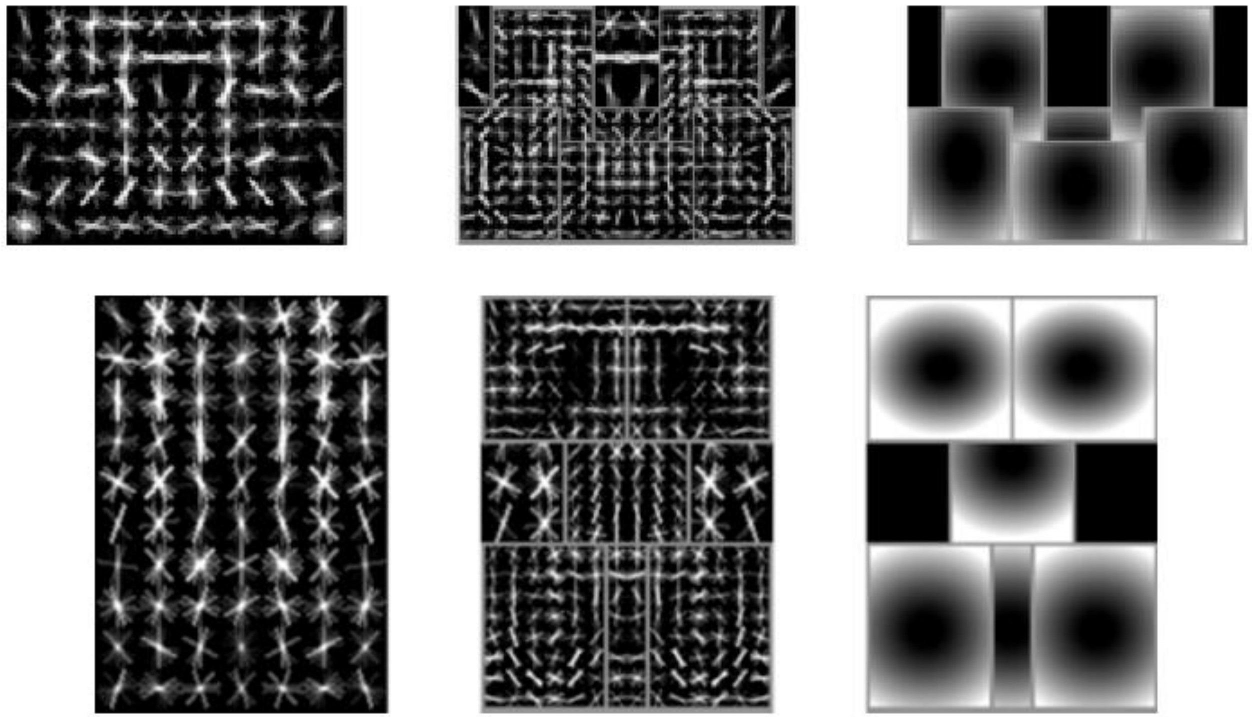
Start with random negatives and repeat:

1) Train a model

2) Harvest false positives to define "hard negatives"

Felzenszwalb

Discriminative part-based models



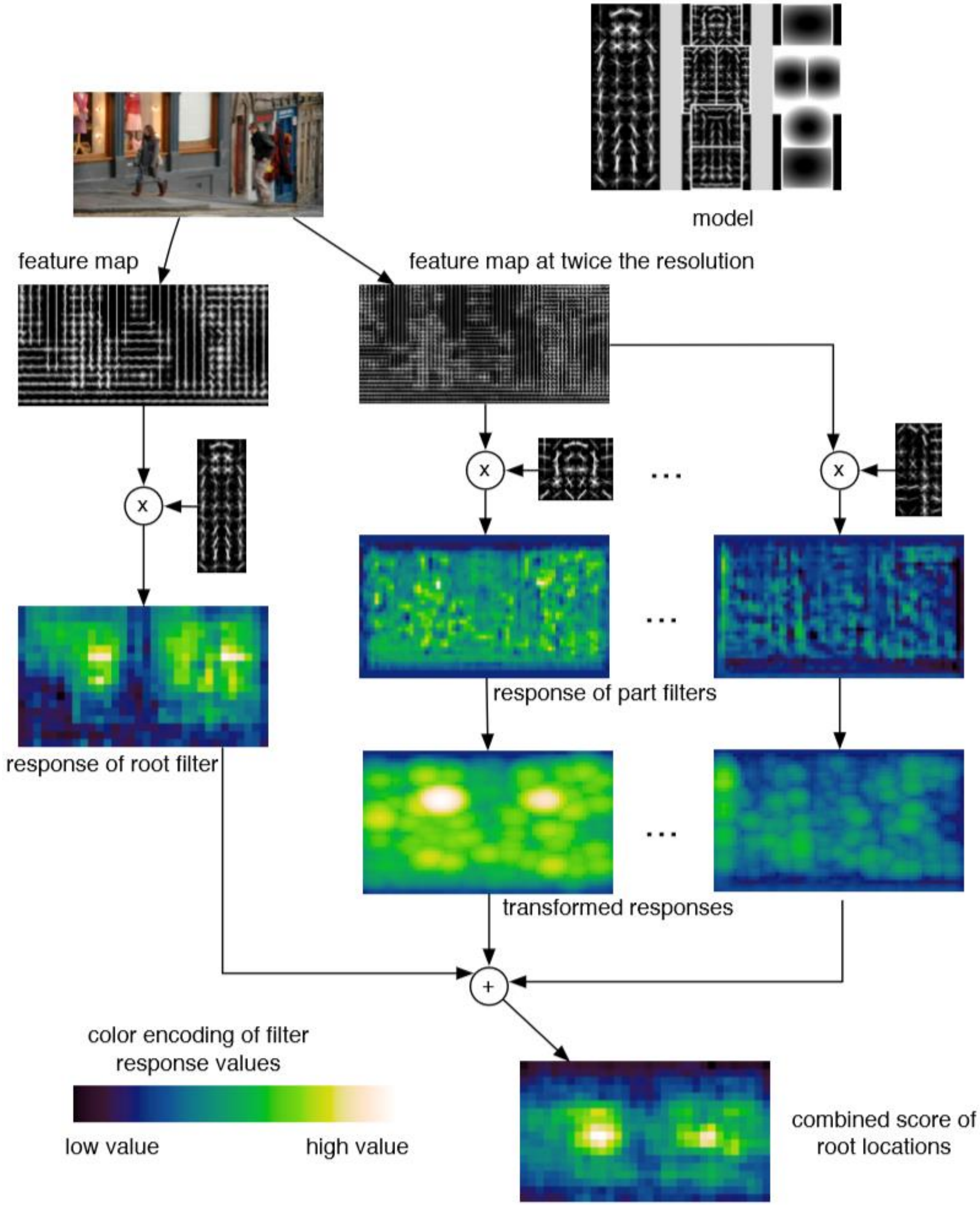
root filters
coarse resolution

part filters
finer resolution

deformation
models

Each component has a root filter F_0
and n part models (F_i, v_i, d_i)

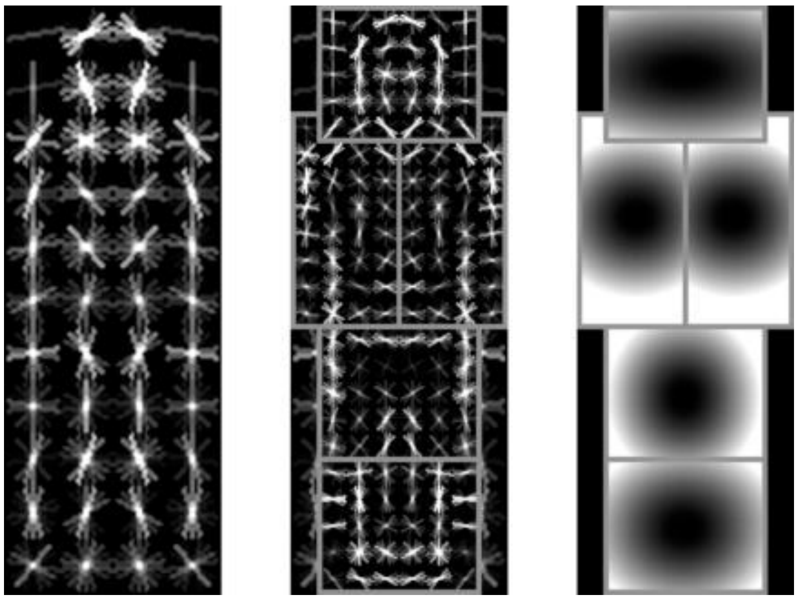
Felzenszwalb

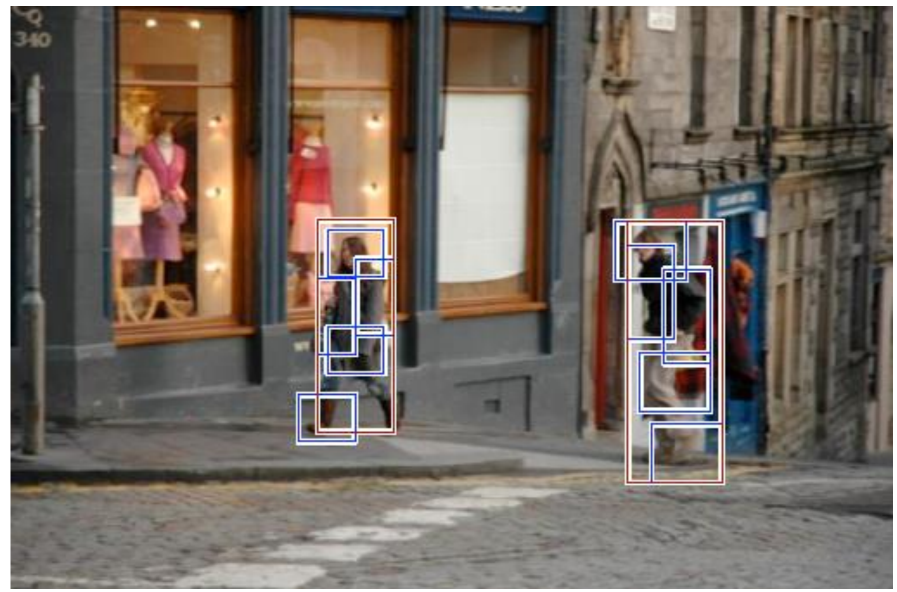


model
feature map
feature map at twice the resolution
x
x
x
...
...
...
response of part filters
response of root filter
transformed responses
+
color encoding of filter
response values
low value
high value
combined score of
root locations

Discriminative part-based models

Root filter | Part filters | Deformation weights



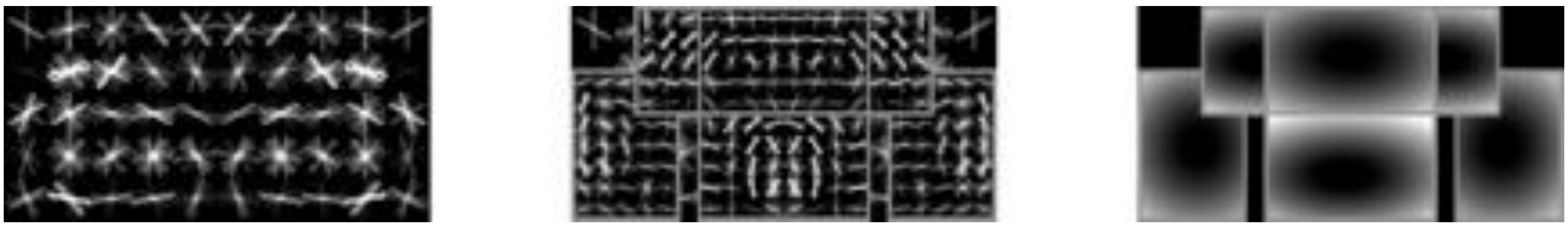


P. Felzenszwalb, R. Girshick, D. McAllester, D. Ramanan, [Object Detection with Discriminatively Trained Part Based Models](), PAMI 32(9), 2010

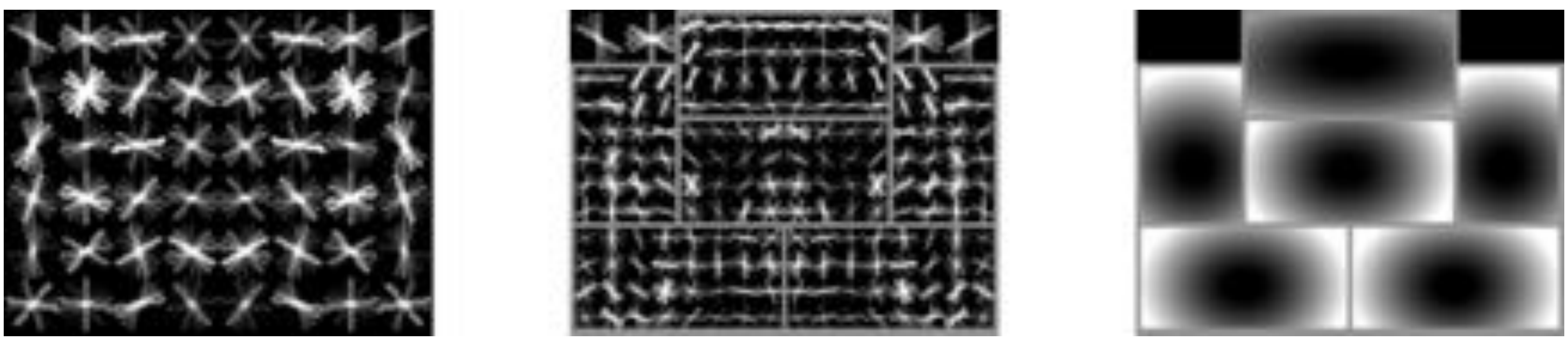
Felzenszwalb

Car model

Component 1

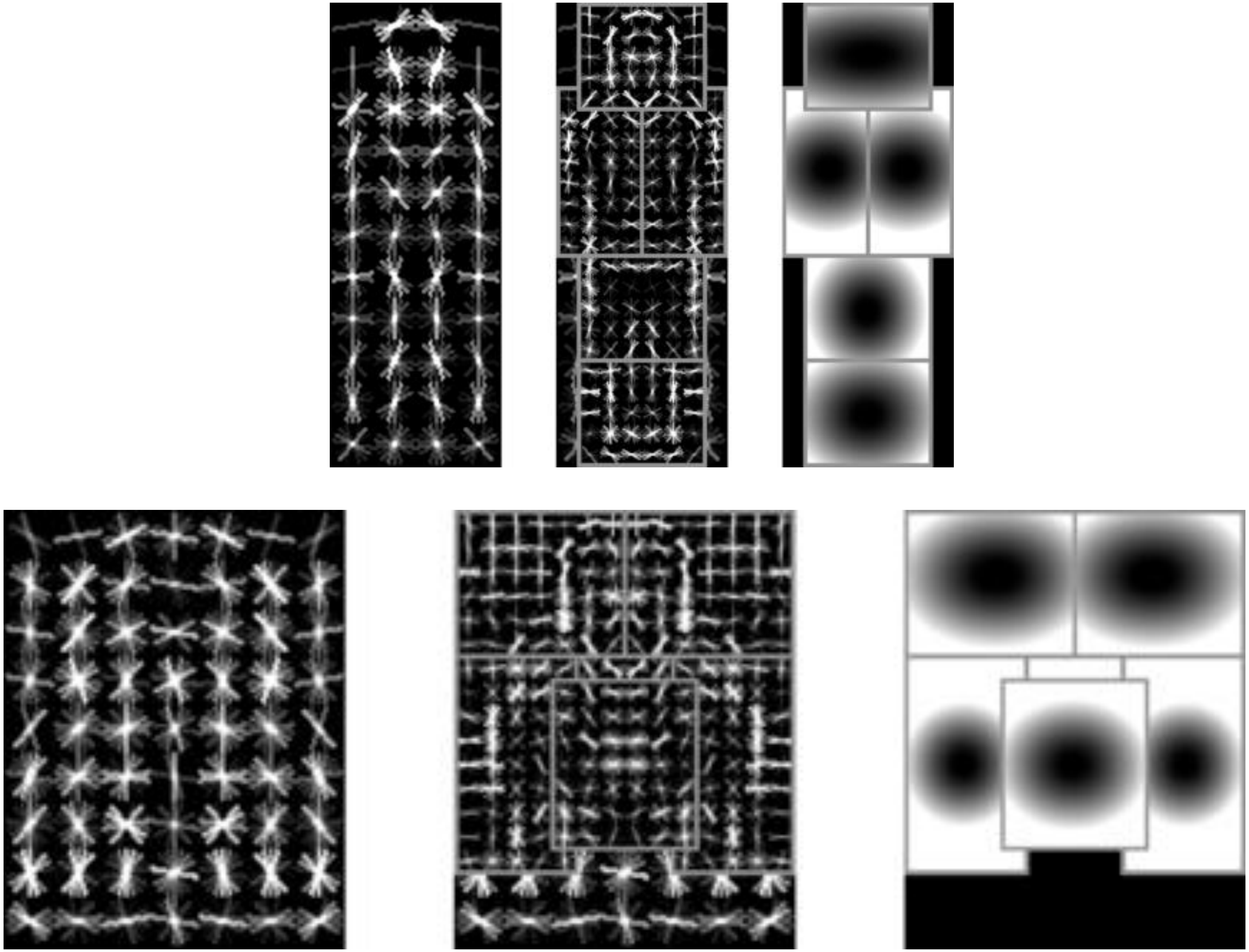


Component 2



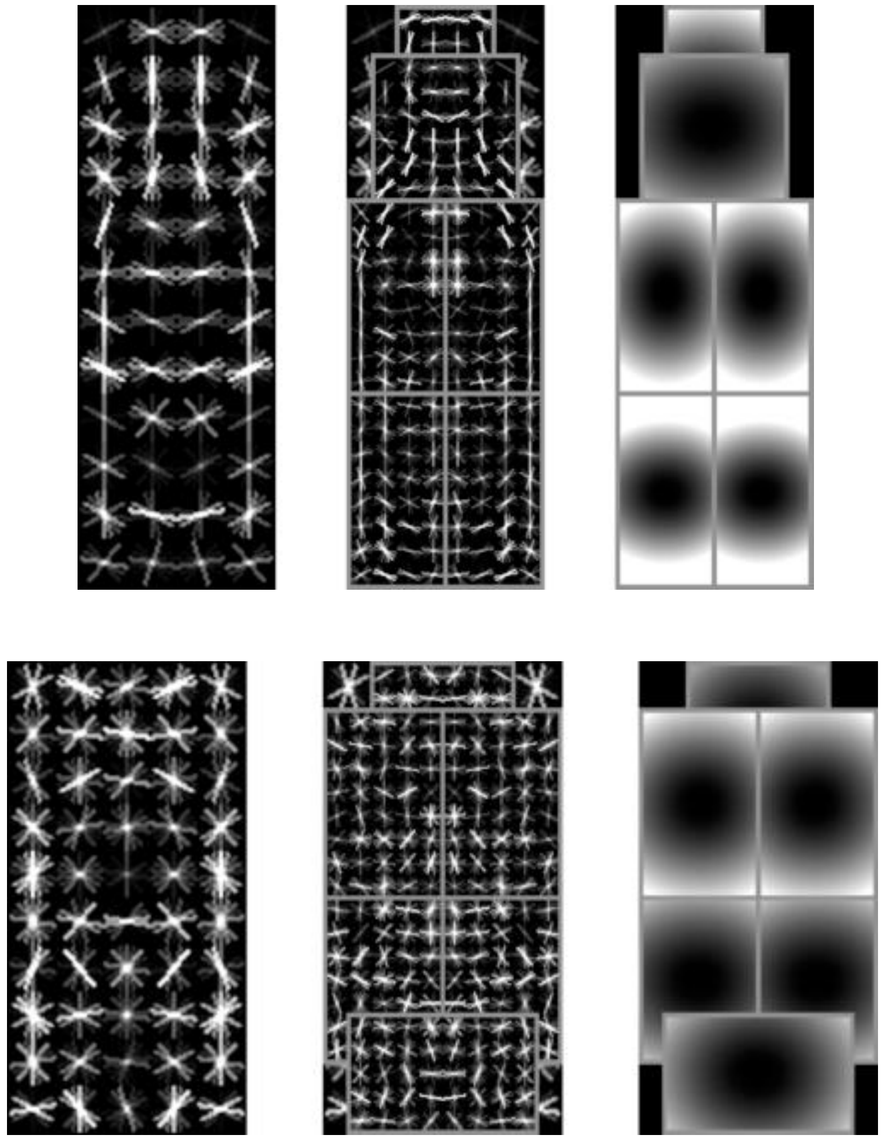
Felzenszwalb

Person model



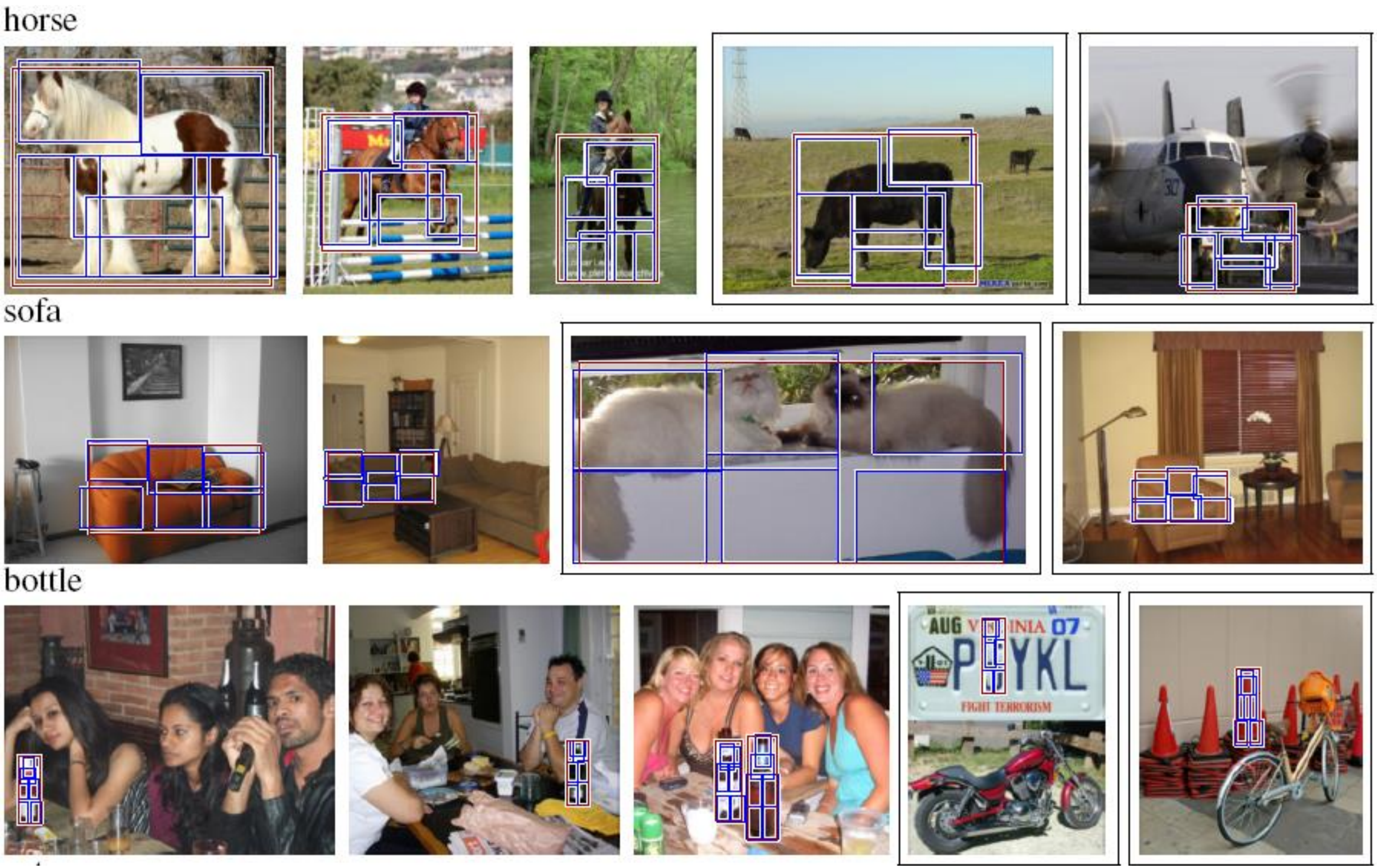
Felzenszwalb

Bottle model



Felzenszwalb

Good detections?



HOGgles (Vondrick et al. ICCV 2013)

Vondrick et al.

Vondrick et al.

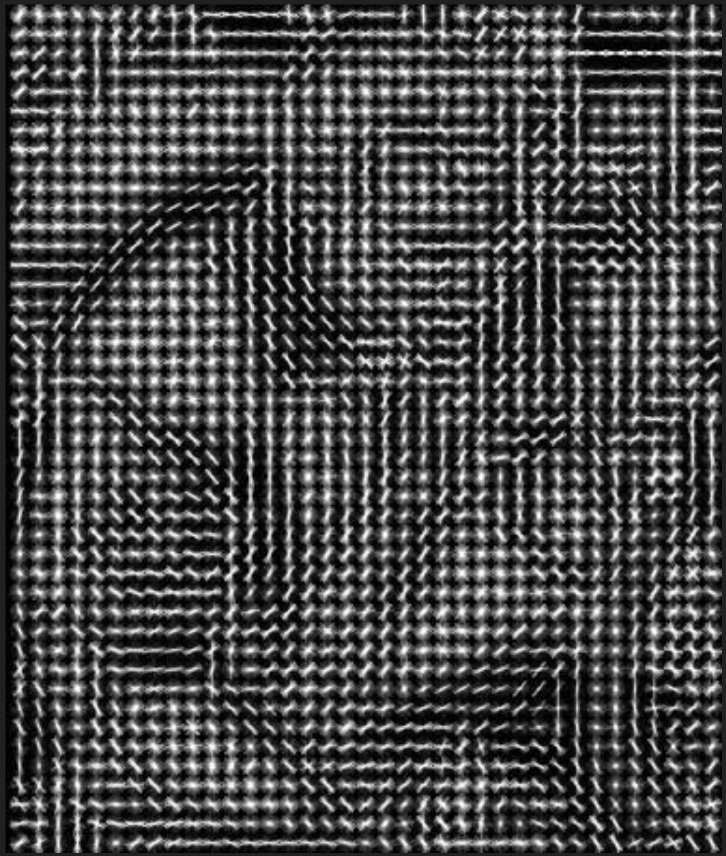
Vondrick et al.

Vondrick et al.

What information is lost?

Vondrick et al.

What information is lost?



Vondrick et al.

How can we ‘invert’ lossy HOG?

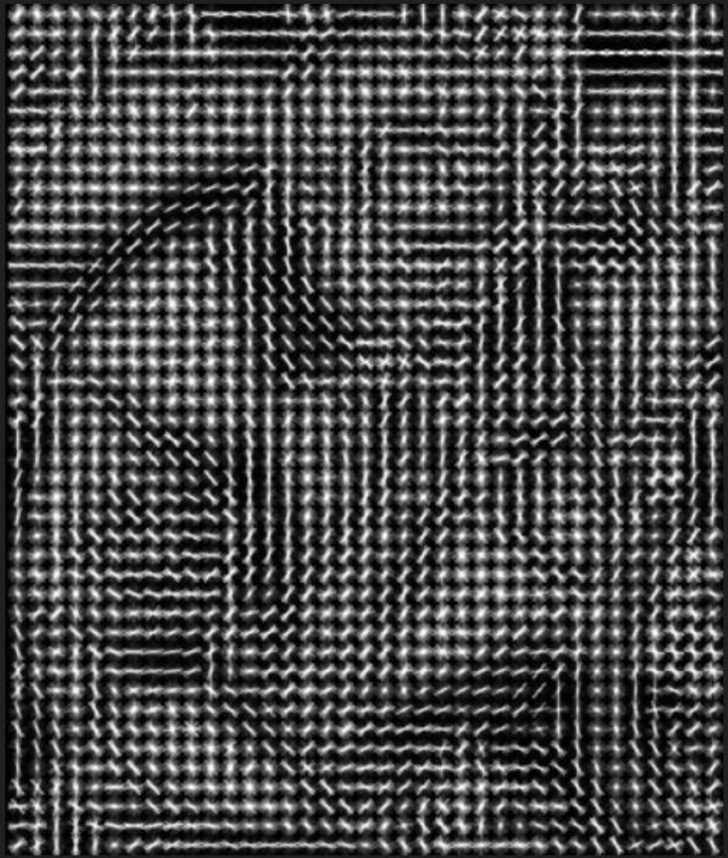
- Gradient computation
  - Without width or ‘edge blur’, i.e., not edges from Eldar 1999
- Oriented magnitude sum (via bins)
  - Loss of precision
  - Loss of specificity – any number of values can sum to the same total
- Normalization
  - No way to unnormalize without knowing normalization factors

Many different image patches translate to the same HOG feature : (

What information is lost?

$x = input\ patch$
$y = HOG\ descriptor$
$\phi(x) = HOG\ transform$

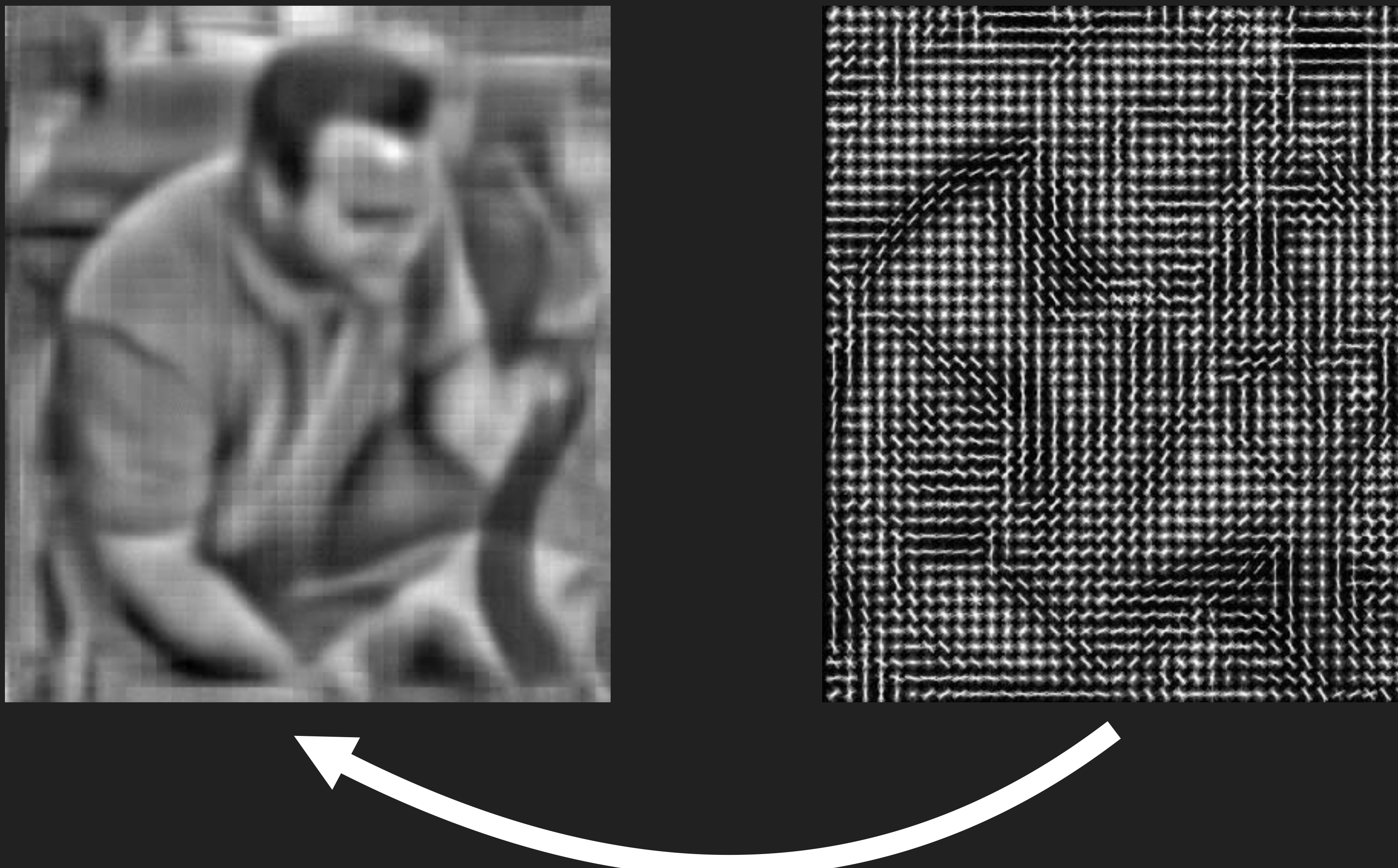
$$\min_{x \in \mathbb{R}^d} ||\phi(x) - y||_2^2$$



Hard to optimize!
Many-to-one = unconstrained!

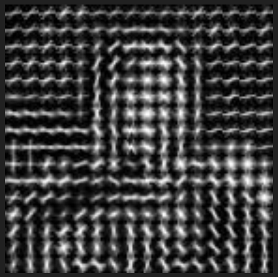
Vondrick et al.

What information is lost?



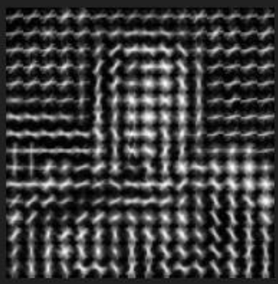
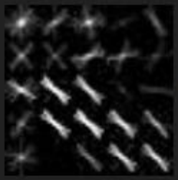
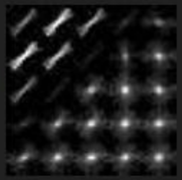
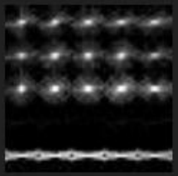
Vondrick et al.

Method: Paired Dictionary



Vondrick et al.

Method: Paired Dictionary

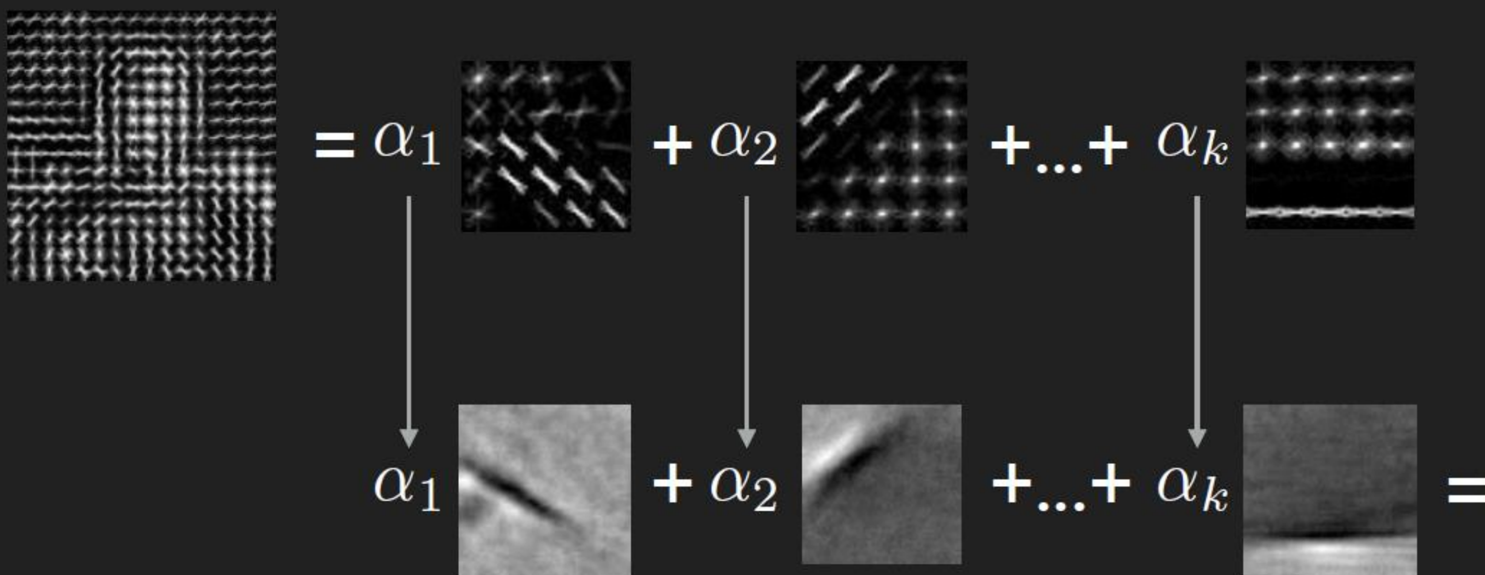
 $= \alpha_1$  $+ \alpha_2$  $+ \ldots + \alpha_k$ 

How to constrain (two parts):

1. Learn a basis over HOG windows

Vondrick et al.

Method: Paired Dictionary

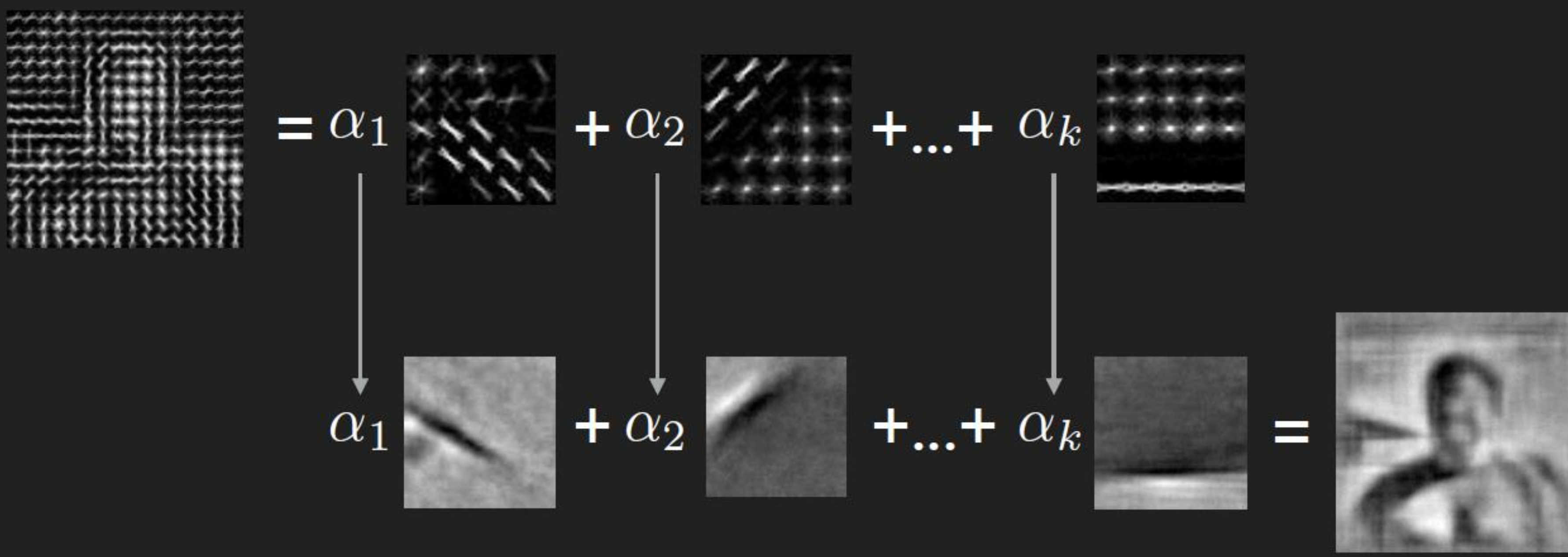


How to constrain (two parts):

1. Learn a basis over HOG windows
2. Simultaneously learn a basis over input windows, and *share the weights* $\alpha_1 \dots \alpha_k$ *over the training data*

Vondrick et al.

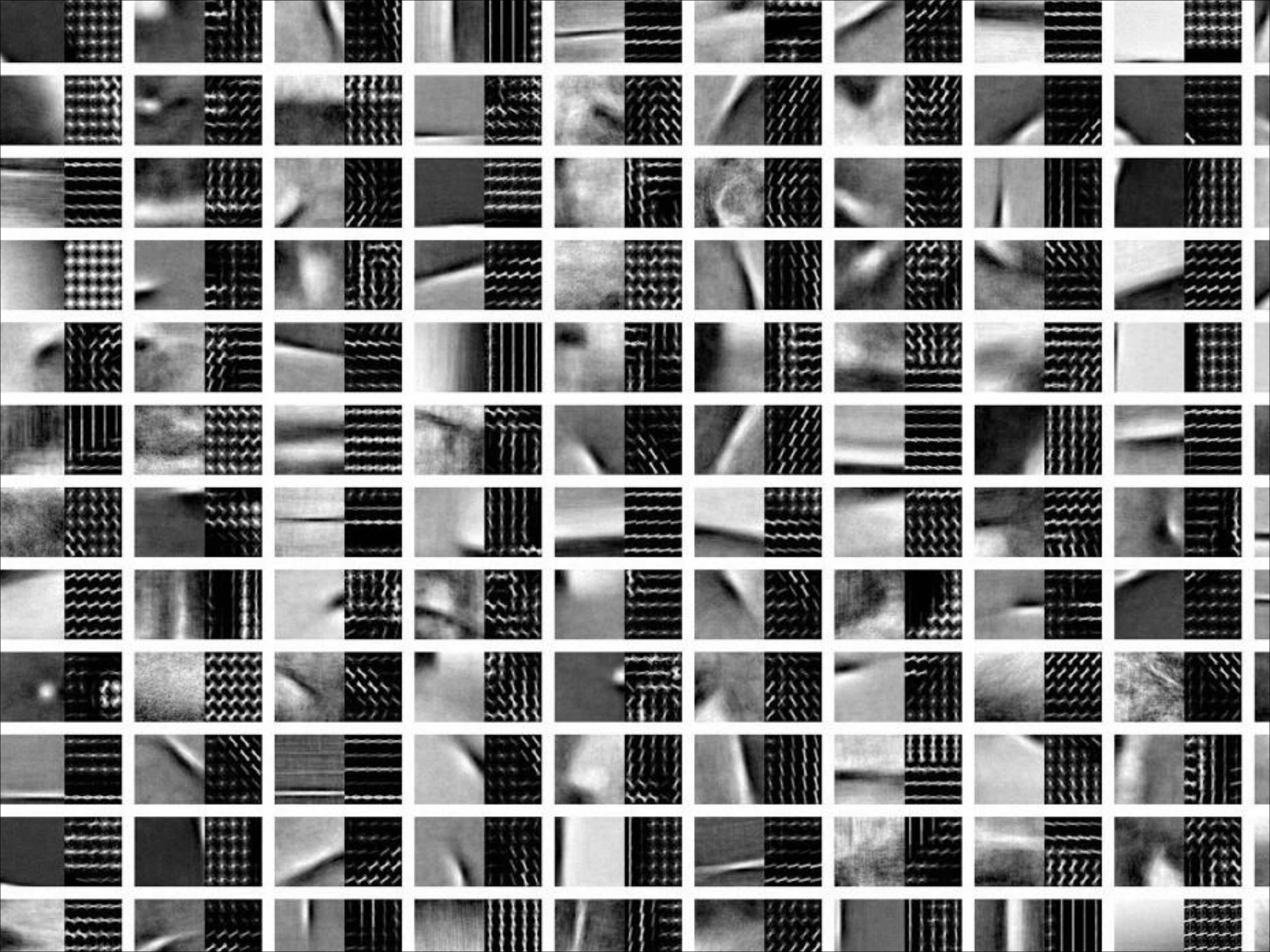
Method: Paired Dictionary

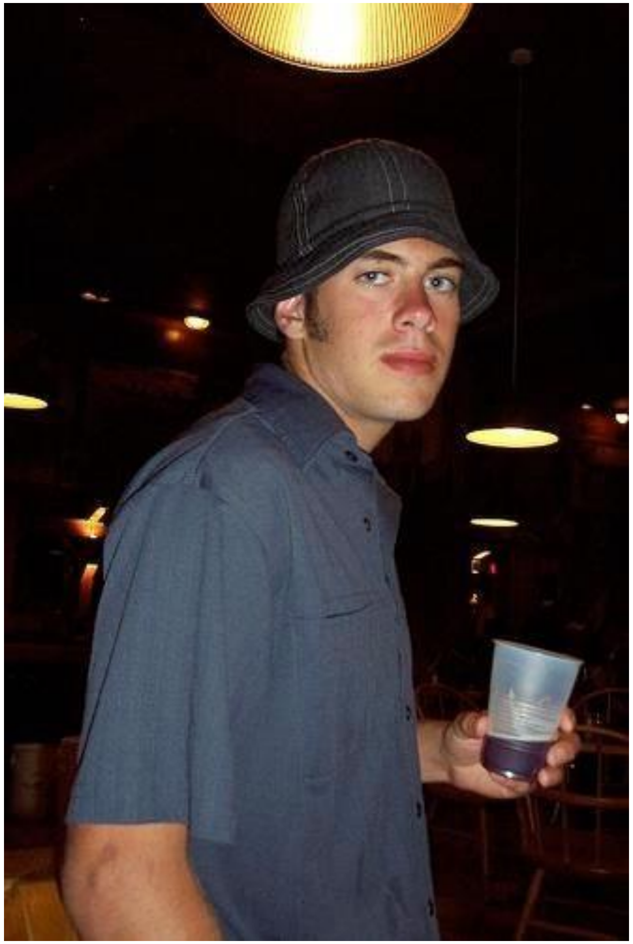


Inference to invert HOG:

1. Transform HOG patch into basis vectors
2. Take weights and apply to input basis

Vondrick et al.



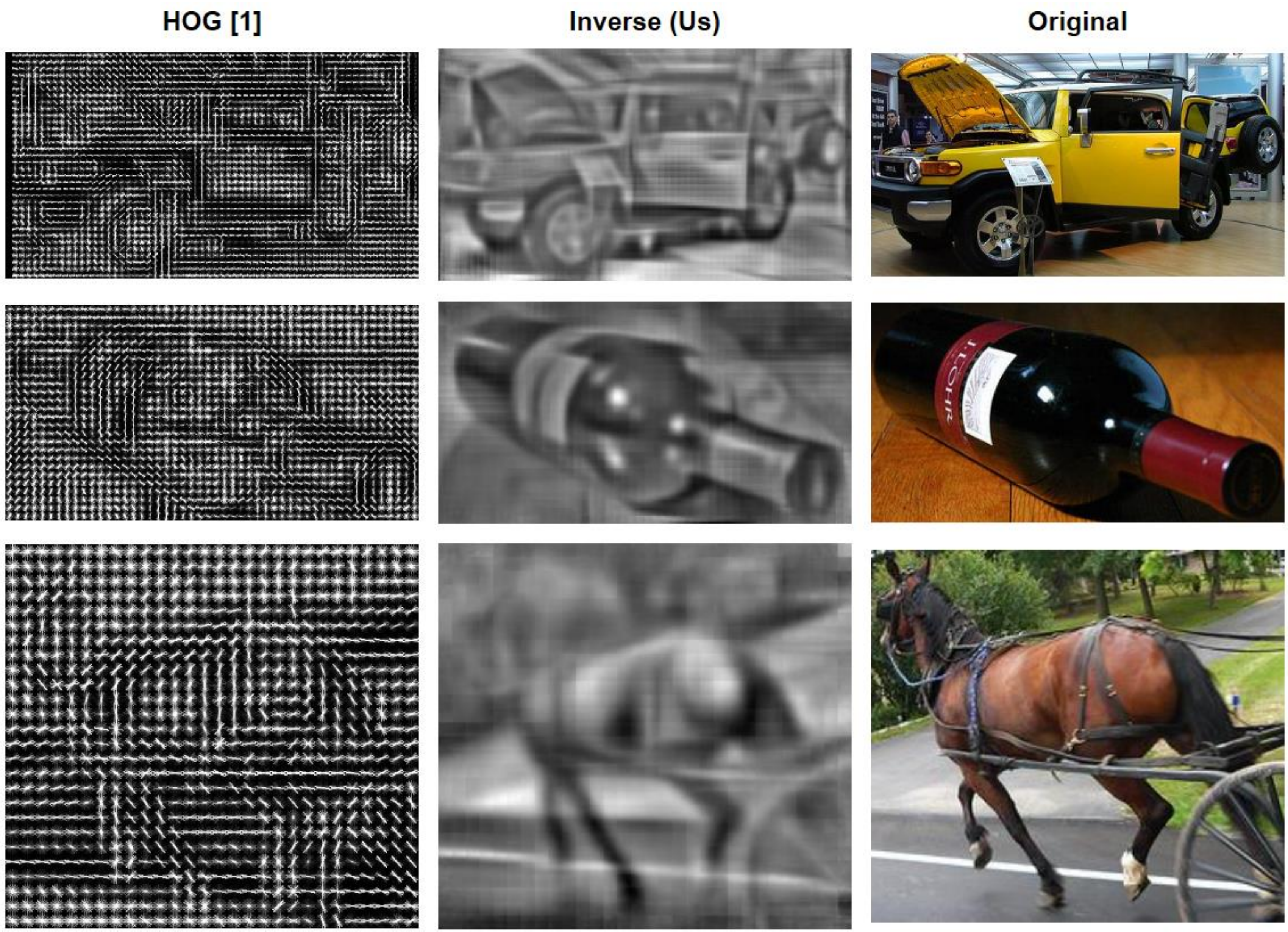


HumanVision

HOGVision

Vondrick et al.

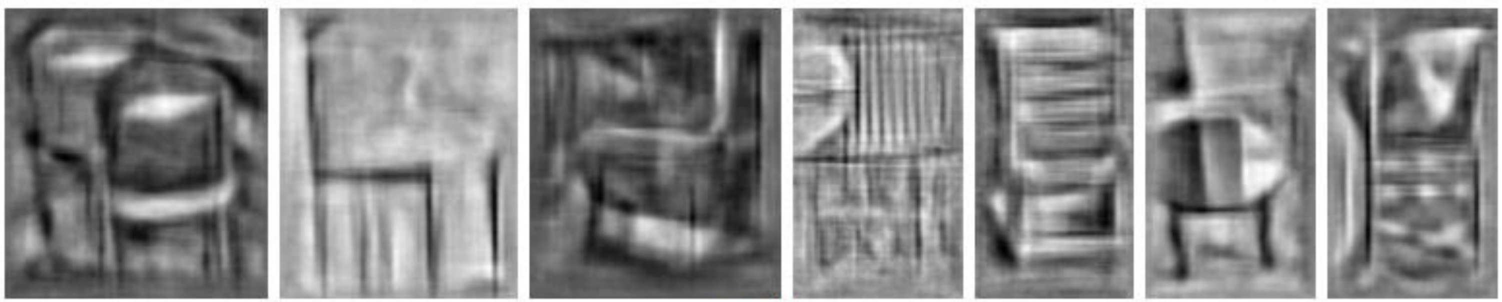
HOGgles (Vondrick et al. ICCV 2013)



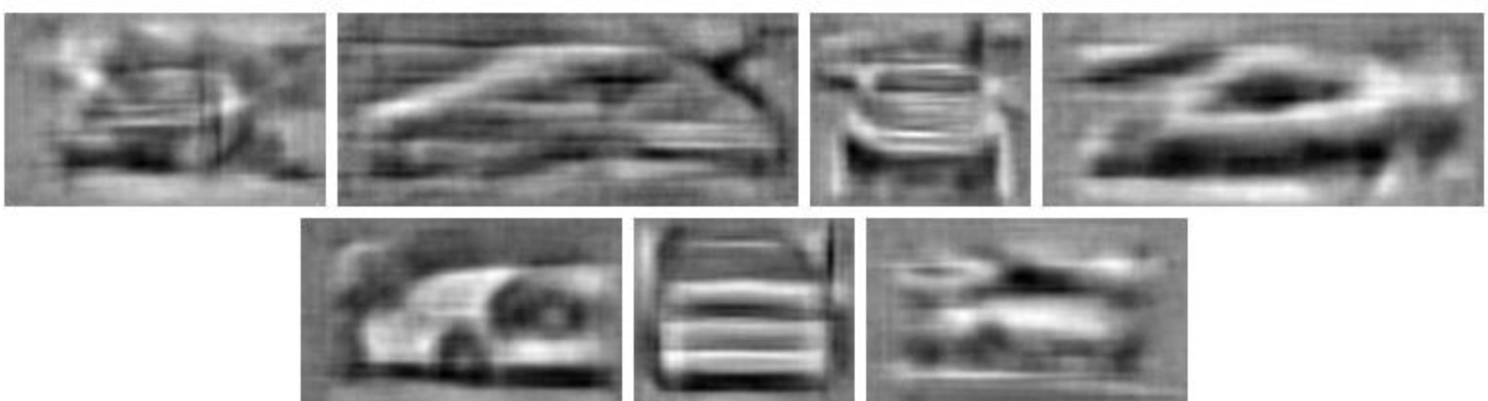
Visualizing Top Detections

We have visualized some high scoring detections from the deformable parts model. Can you guess which are false alarms? Click on the images below to reveal the corresponding RGB patch. You might be surprised!

Person



Chair



Car

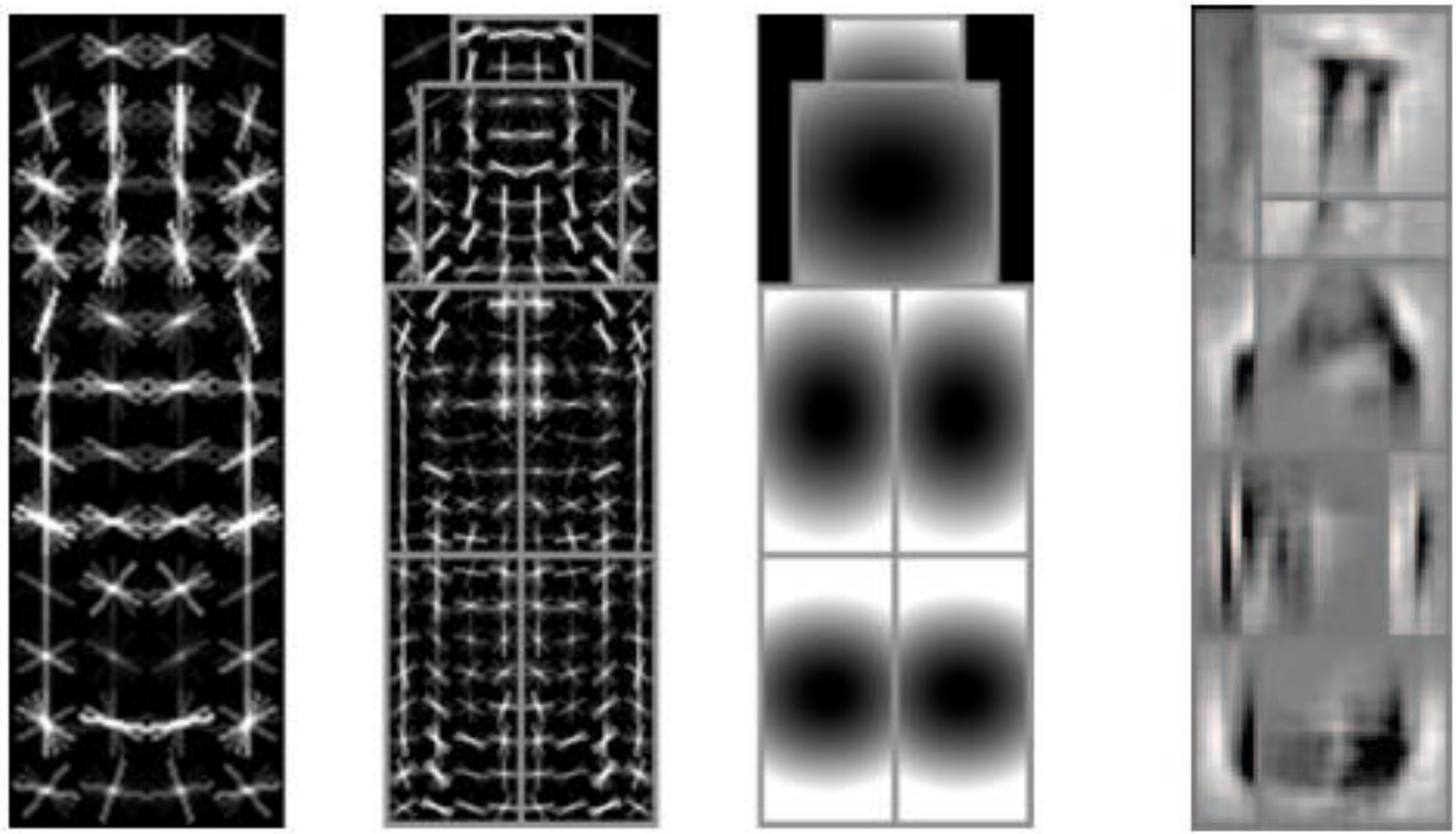
Vondrick et al.

Recursive HOG!

Original x $\qquad x' = \phi^{-1}(\phi(x)) \qquad x'' = \phi^{-1}(\phi(x'))$

Figure 11: We recursively compute HOG and invert it with a paired dictionary. While there is some information loss, our visualizations still do a good job at accurately representing HOG features. $\phi(\cdot)$ is HOG, and $\phi^{-1}(\cdot)$ is the inverse.

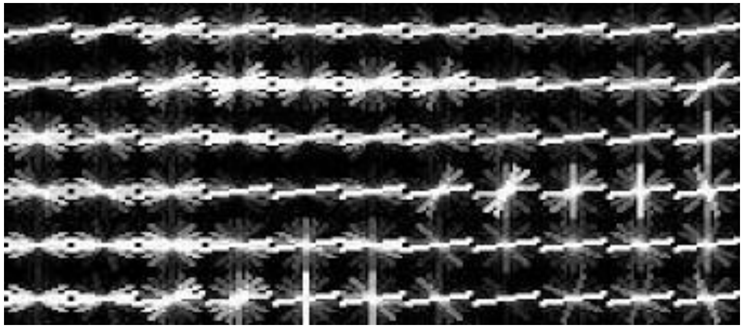
Bottle Deformable Parts Models + HOGgles



Why did the detector fail?

Vondrick et al.

Why did the detector fail?



Vondrick et al.

Why did the detector fail?

Vondrick et al.

Code Available

Try it on your projects!

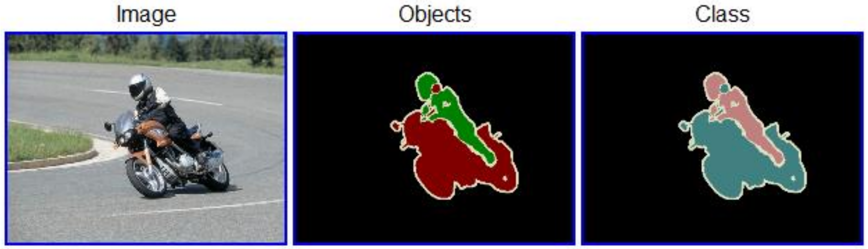
http://web.mit.edu/vondrick/ihog/

```
ihog = invertHOG(feat);
```

The PASCAL Visual Object Classes Challenge 2009 (VOC2009)

- Twenty object categories (aeroplane to TV/monitor)

- Three challenges:
  - Classification challenge (is there an X in this image?)
  - Detection challenge (draw a box around every X)
  - Segmentation challenge



Snavely

Dataset: Collection

- Images downloaded from **flickr**
  - 500,000 images downloaded and random subset selected for annotation

Dataset: Annotation

- "Complete" annotation of all objects
- Annotated over web with <u>written guidelines</u>
  - High quality (?)

Dataset: Annotation

- “Complete” annotation of all objects
- Annotated over web with <u>written guidelines</u>
  - High quality (?)

20 classes.
- Train / validation data has 11,530 images containing 27,450 ROI annotated objects and 6,929 segmentations.

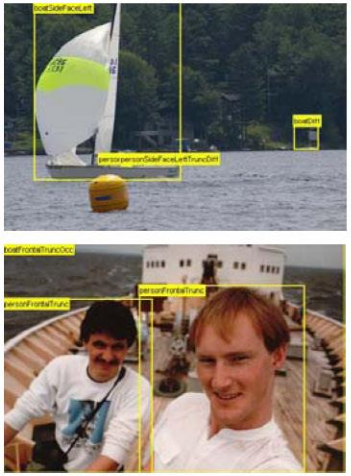
Examples

Aeroplane

Bicycle

Bird

Boat



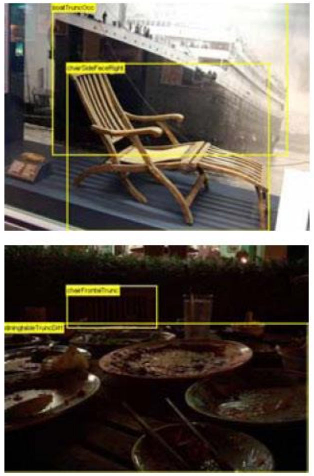
Bottle

Bus

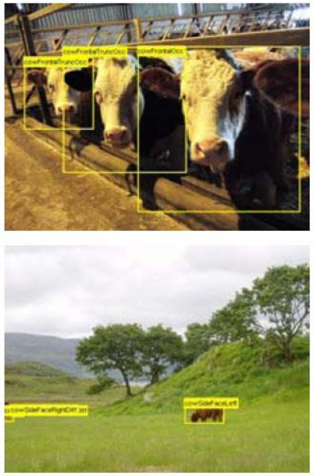
Car

Cat

Chair

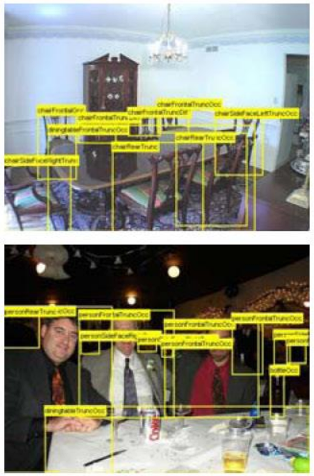


Cow



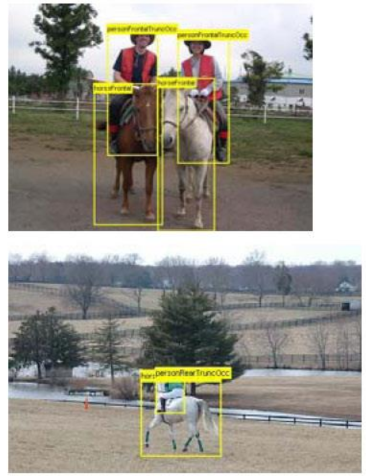
Examples

Dining Table



Dog

Horse

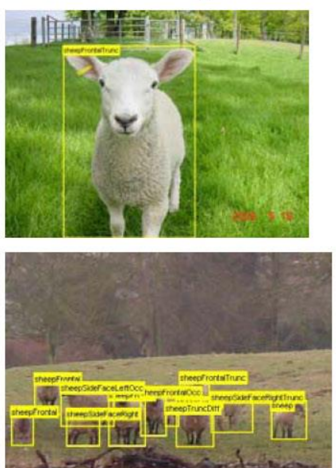


Motorbike

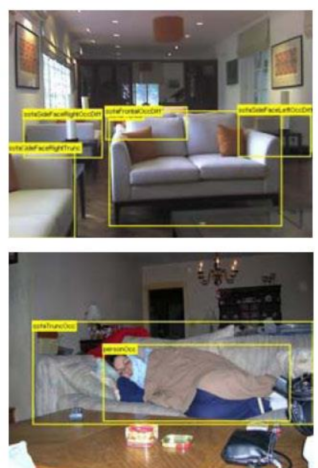
Person

Potted Plant

Sheep

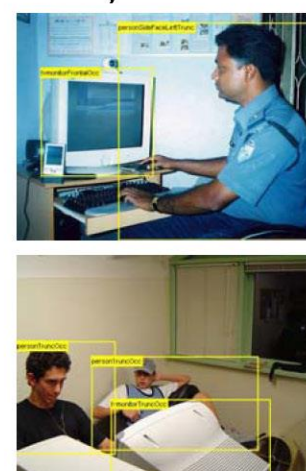


Sofa



Train

TV/Monitor



Classification Challenge

- Predict whether at least one object of a given class is present in an image

is there a cat?

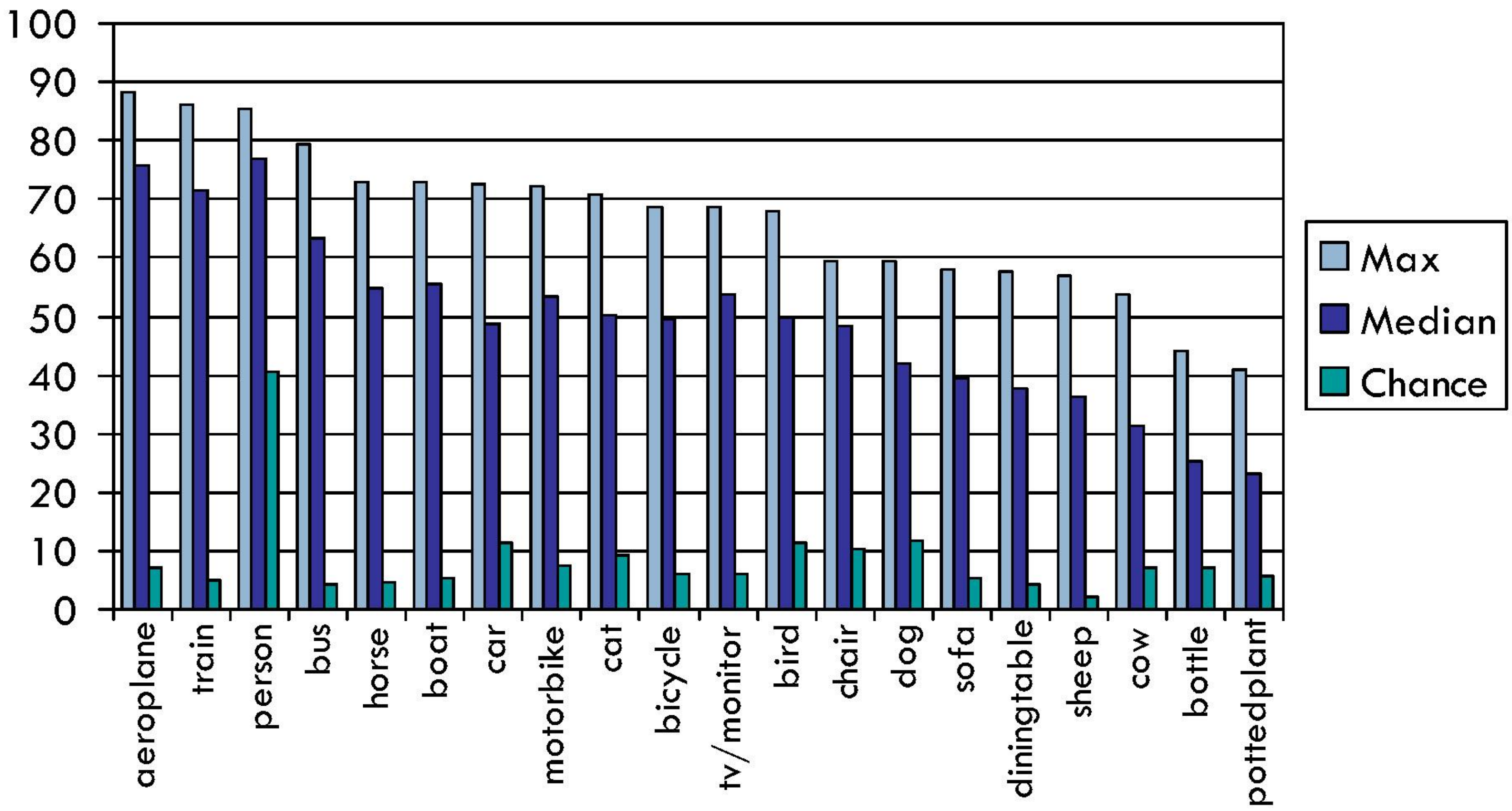
Results: AP by Method and Class

	aero plane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow	dining table	dog	horse	motor bike	person	potted plant	sheep	sofa	train	tv/ monitor
CVC_FLAT	85.3	57.8	66.0	66.1	36.2	70.6	60.6	63.5	55.1	44.6	53.4	49.1	64.4	66.8	84.8	37.4	44.1	47.9	81.9	67.5
CVC_FLAT-HOG-ESS	86.3	60.7	66.4	65.3	41.0	71.7	64.7	63.9	55.5	40.1	51.3	45.9	65.2	68.9	85.0	40.8	49.0	49.1	81.8	68.6
CVC_PLUS	86.6	58.4	66.7	67.3	34.8	70.4	60.0	64.2	52.5	43.0	50.8	46.5	64.1	66.8	84.4	37.5	45.1	45.4	82.1	67.0
FIRSTNIKON_AVGSRKDA	83.3	59.3	62.7	65.3	30.2	71.6	58.2	62.2	54.3	40.7	49.2	50.0	66.6	62.9	83.3	34.2	48.2	46.1	83.4	65.5
FIRSTNIKON_AVGSVM	83.8	58.2	62.6	65.2	32.0	69.8	57.7	61.1	54.5	44.0	50.3	49.6	64.6	61.7	83.2	33.4	46.5	48.0	81.6	65.3
FIRSTNIKON_BOOSTSRKDA	83.0	59.2	61.4	64.6	33.2	71.1	57.5	61.0	54.8	40.7	48.3	50.0	65.5	63.4	82.8	32.8	47.0	47.1	83.3	64.6
FIRSTNIKON_BOOSTSVMS	83.5	56.8	61.8	65.5	33.2	69.7	57.3	60.5	54.6	43.1	48.3	50.3	64.3	62.4	82.3	32.9	46.9	48.4	82.0	64.2
LEAR_CHI-SVM-MULT-LOC	79.5	55.5	54.5	63.9	43.7	70.3	66.4	56.5	54.4	38.8	44.1	46.2	58.5	64.2	82.2	39.1	41.3	39.8	73.6	66.2
NECUIUC_CDCV	88.1	68.0	68.0	72.5	41.0	78.9	70.4	70.4	58.1	53.4	55.7	59.3	73.1	71.3	84.5	32.3	53.3	56.7	86.0	66.8
NECUIUC_CLS-DTCT	88.0	68.6	67.9	72.9	44.2	79.5	72.5	70.8	59.5	53.6	57.5	59.0	72.6	72.3	85.3	36.6	56.9	57.9	85.9	68.0
NECUIUC_LL-CDCV	87.1	67.4	65.8	72.3	40.9	78.3	69.7	69.7	58.5	50.1	55.1	56.3	71.8	70.8	84.1	31.4	51.5	55.1	84.7	65.2
NECUIUC_LN-CDCV	87.7	67.8	68.1	71.1	39.1	78.5	70.6	70.7	57.4	51.7	53.3	59.2	71.6	70.6	84.0	30.9	51.7	55.9	85.9	66.7
UVASURREY_BASELINE	84.1	59.2	62.7	65.4	35.7	70.6	59.8	61.3	56.7	45.3	52.4	50.6	66.1	66.6	83.7	34.8	47.2	47.7	80.8	65.9
UVASURREY_MKFDA+BOW	84.7	63.9	66.1	67.3	37.9	74.1	63.2	64.0	57.1	46.2	54.7	53.5	68.1	70.6	85.2	38.5	47.2	49.3	83.2	68.1
UVASURREY_TUNECOLORKERNELSEL	85.0	62.8	65.1	66.5	37.6	73.5	62.1	62.0	57.4	45.1	54.5	52.5	67.7	69.8	84.8	39.1	46.8	49.9	82.9	68.1
UVASURREY_TUNECOLORSPECKDA	84.6	62.4	65.6	67.2	39.4	74.0	63.4	62.8	56.7	43.8	54.7	52.7	67.3	70.6	85.0	38.8	46.9	50.0	82.2	66.2

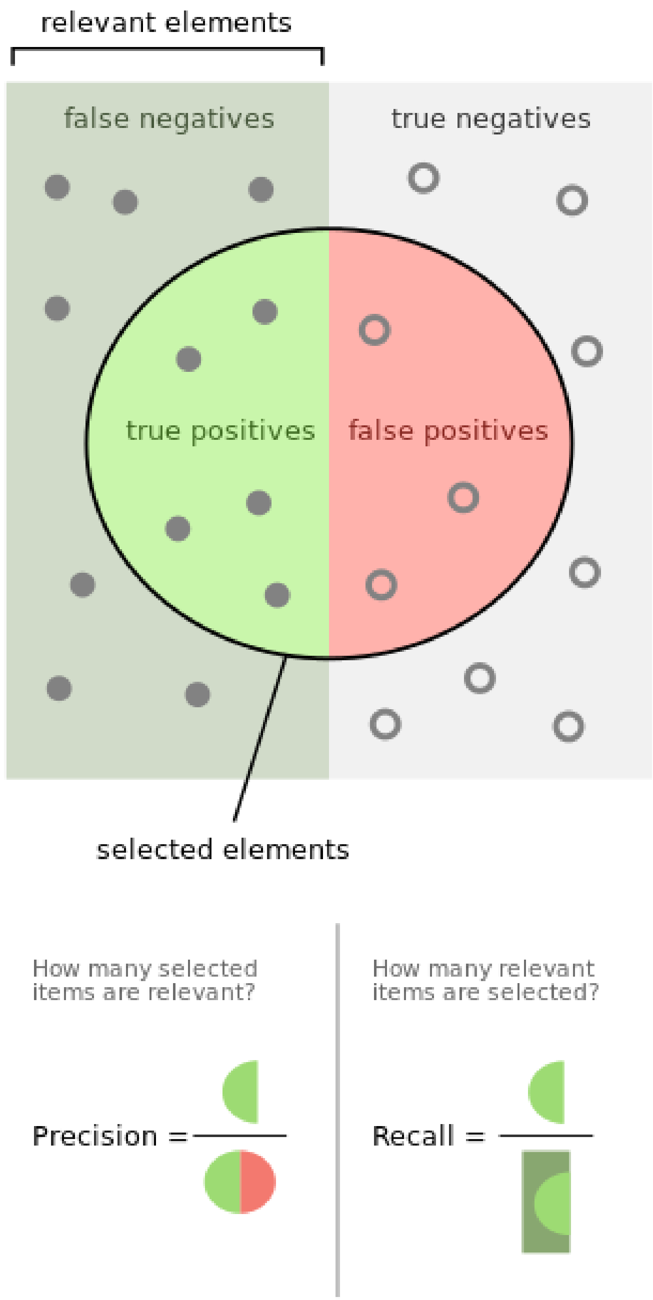
- Only methods in 1st, 2nd or 3rd place by group shown
- Groups: CVC, FIRST/Nikon, NEC/UIUC, UVA/Surrey

AP by Class

AP = average precision

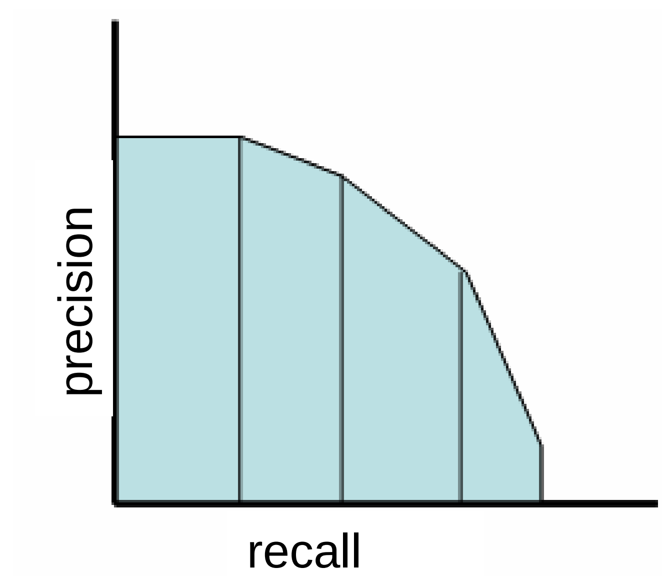


- Max AP: 88.1% (aeroplane) ... 40.8% (potted plant)



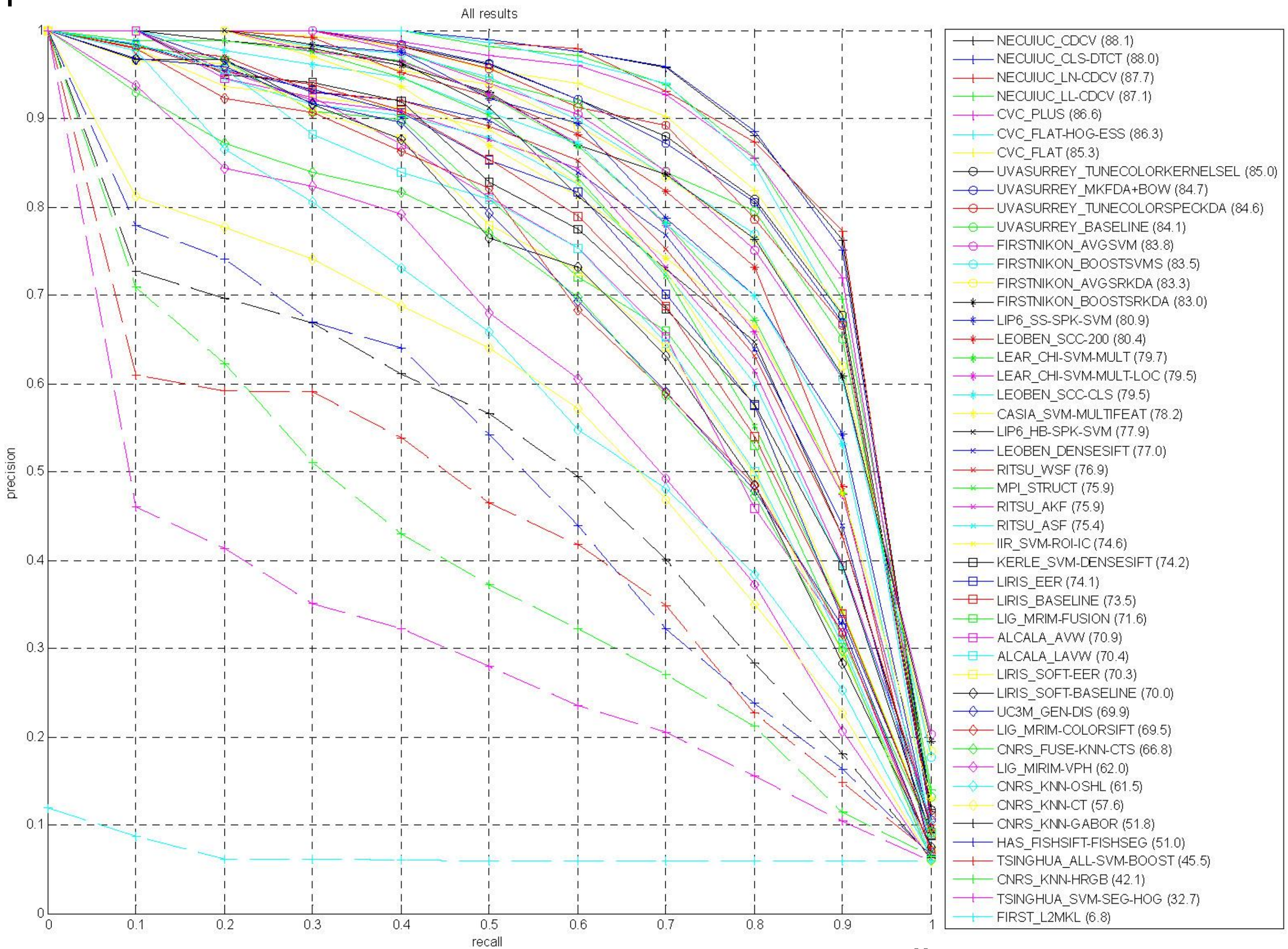
Set threshold on ‘detection’ to create one pair of precision / recall values.

Vary threshold across all values to generate precision / recall curves:



Precision/Recall: Aeroplane (All)

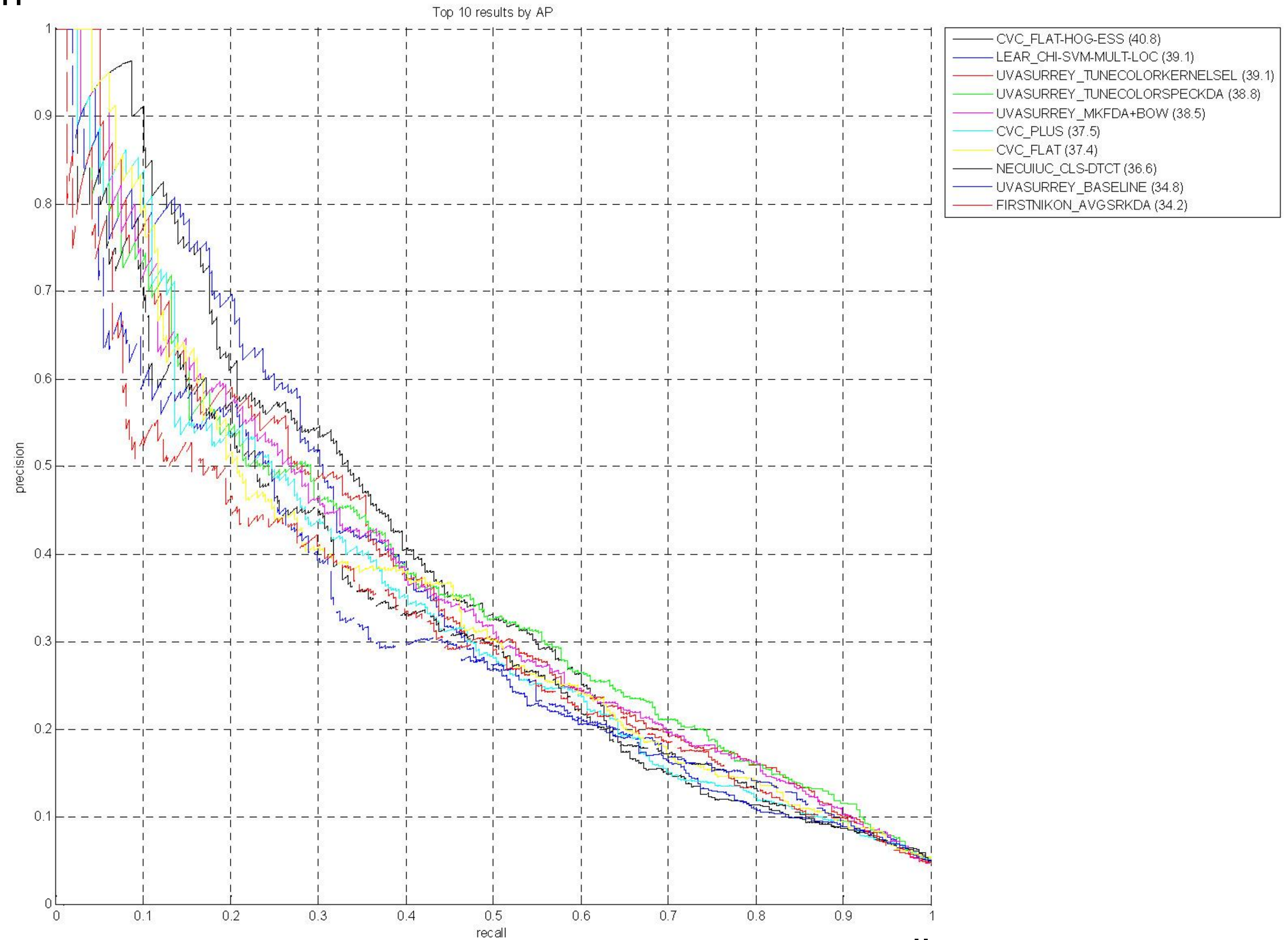
precision



recall

Precision/Recall: Potted plant (Top 10 by AP)

precision



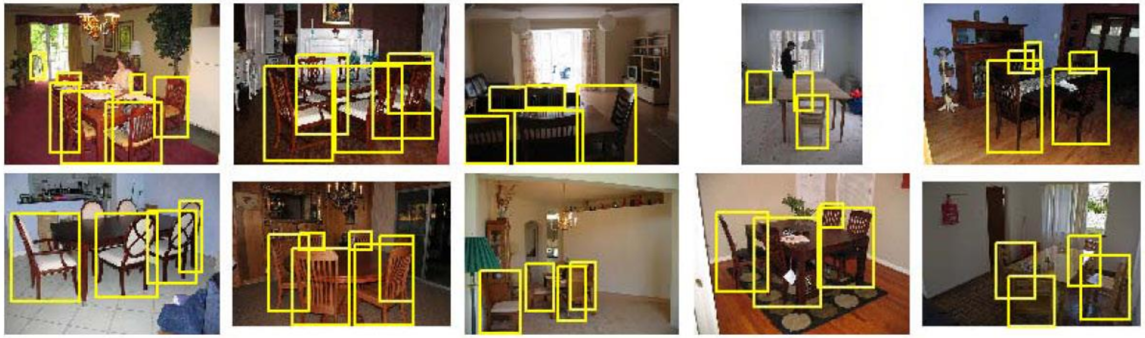
recall

Ranked Images: Aeroplane

- Class images: Highest ranked

Ranked Images: Chair

- Class images: Highest ranked

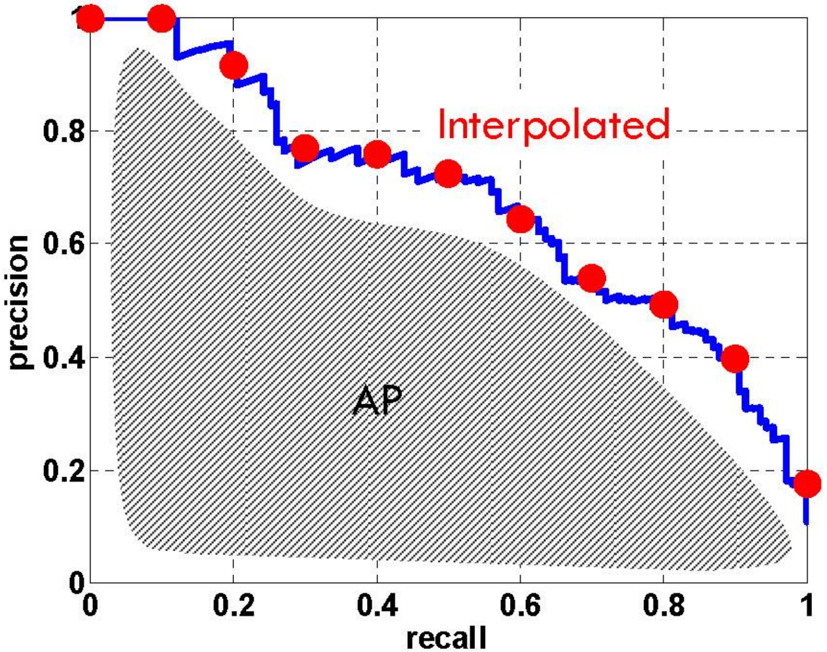


Detection Challenge

- Predict the bounding boxes of all objects of a given class in an image (if any)

Evaluation

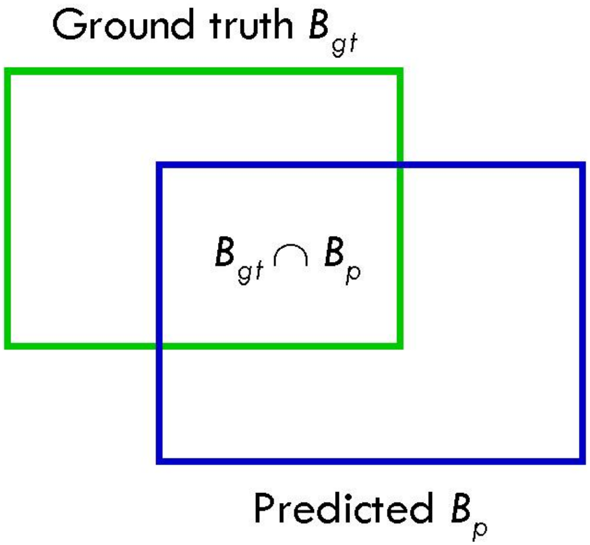
- Average Precision [TREC] averages precision over the entire range of recall
  - Curve interpolated to reduce influence of "outliers"



- A good score requires both high recall and high precision
- Application-independent
- Penalizes methods giving high precision but low recall

Evaluating Bounding Boxes

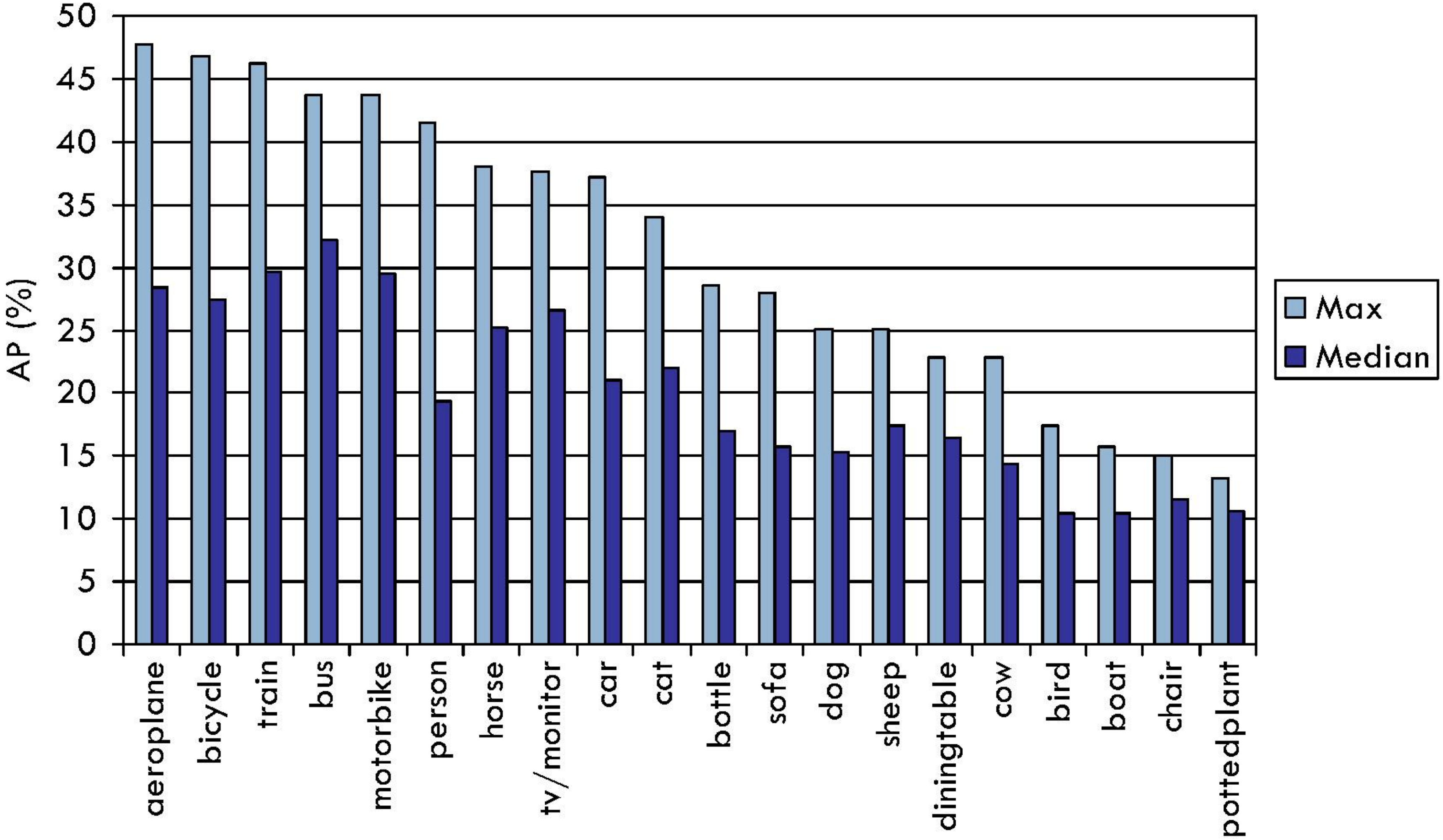
- Area of Overlap (AO) Measure



$$AO(B_{gt}, B_p) = \frac{|B_{gt} \bigcap B_p|}{|B_{gt} \bigcup B_p|}$$

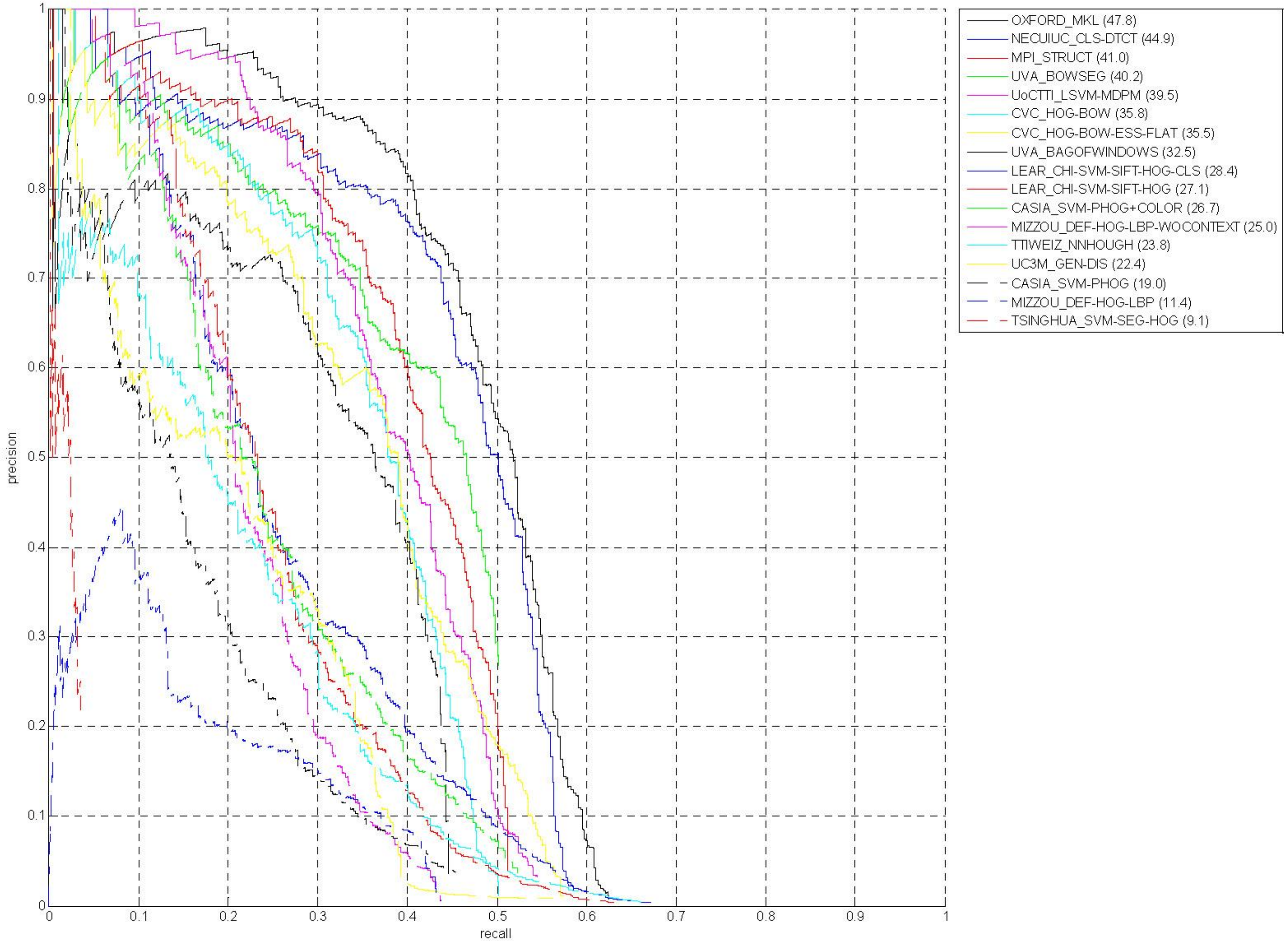
  - Need to define a threshold *t* such that $AO(B_{gt},B_p)$ implies a correct detection: 50%

AP by Class

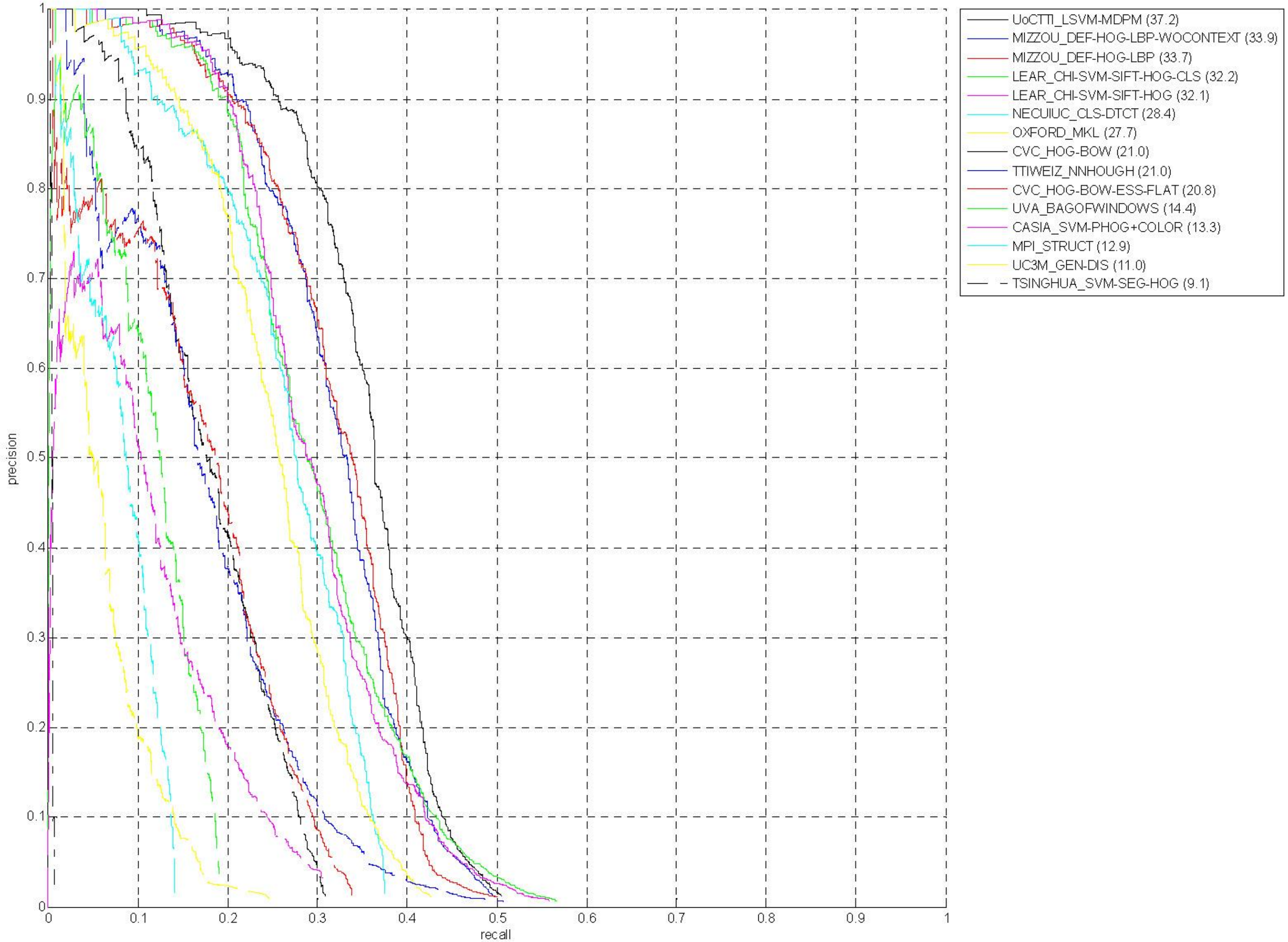


Chance essentially 0

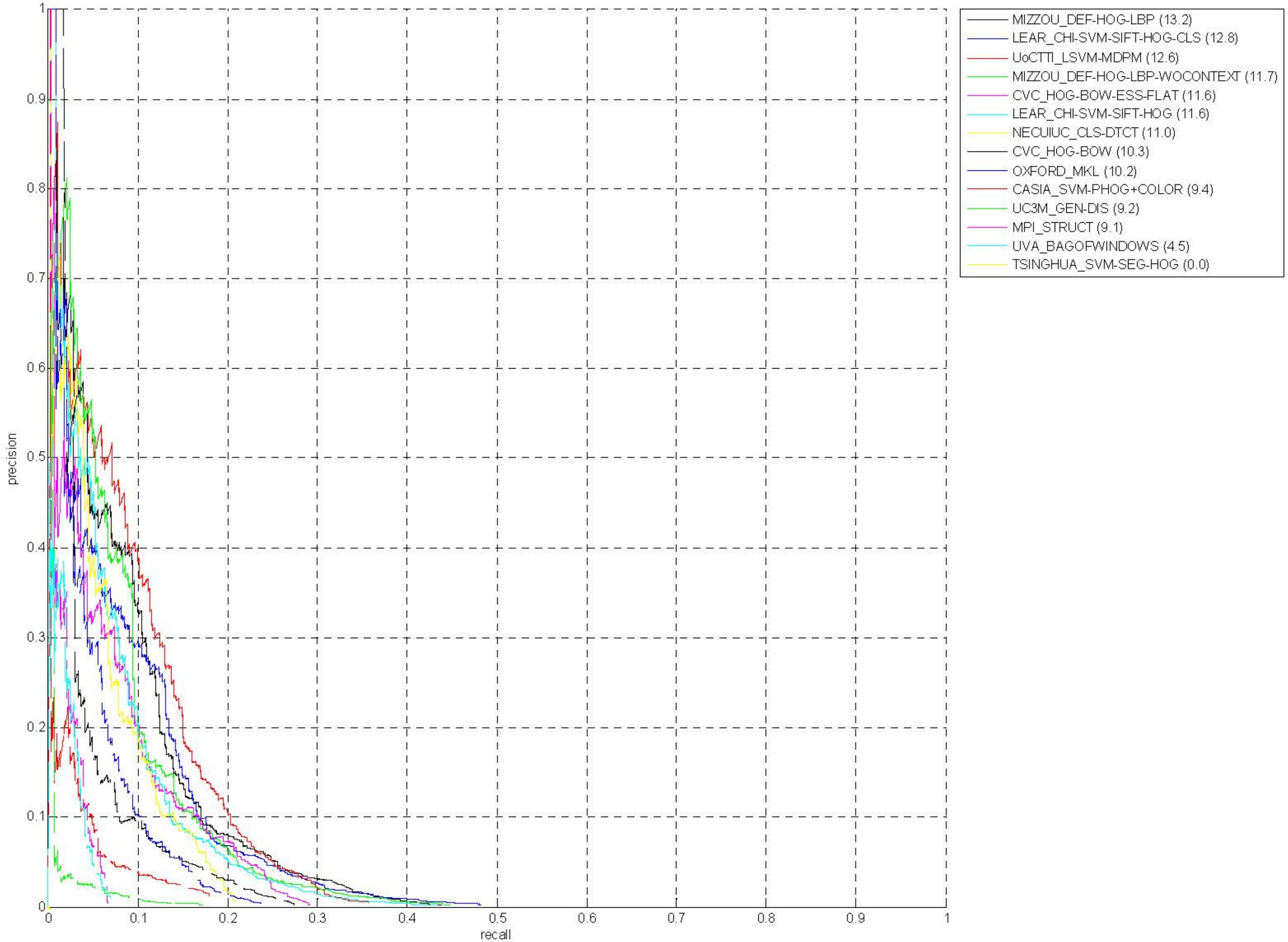
Precision/Recall - Aeroplane



Precision/Recall - Car



Precision/Recall – Potted plant



True Positives - Person

UoCTTI_LSVM-MDPM

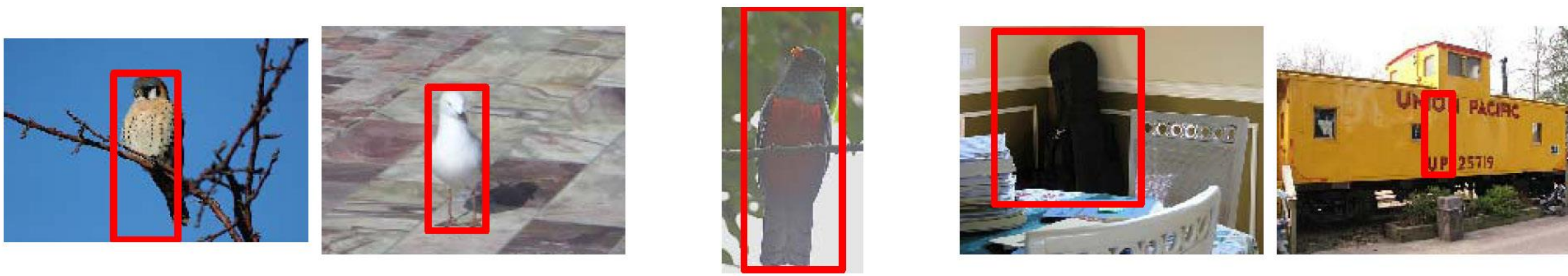
MIZZOU_DEF-HOG-LBP

NECUIUC_CLS-DTCT

False Positives - Person

UoCTTI_LSVM-MDPM

MIZZOU_DEF-HOG-LBP



NECUIUC_CLS-DTCT

“Near Misses” - Person

UoCTTI_LSVM-MDPM

MIZZOU_DEF-HOG-LBP

NECUIUC_CLS-DTCT

True Positives - Bicycle

UoCTTI_LSVM-MDPM

OXFORD_MKL

NECUIUC_CLS-DTCT

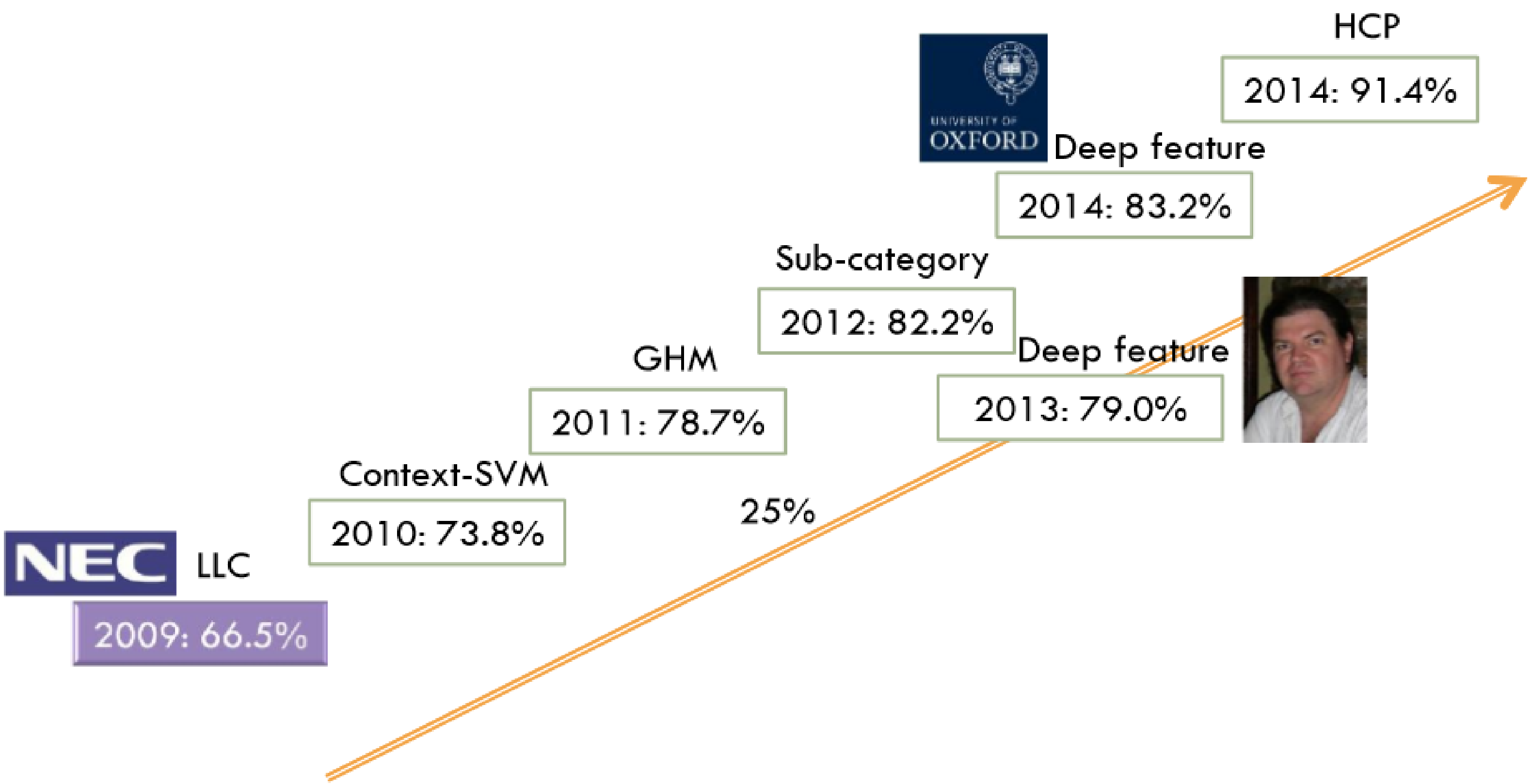
False Positives - Bicycle

UoCTTI_LSVM-MDPM

OXFORD_MKL

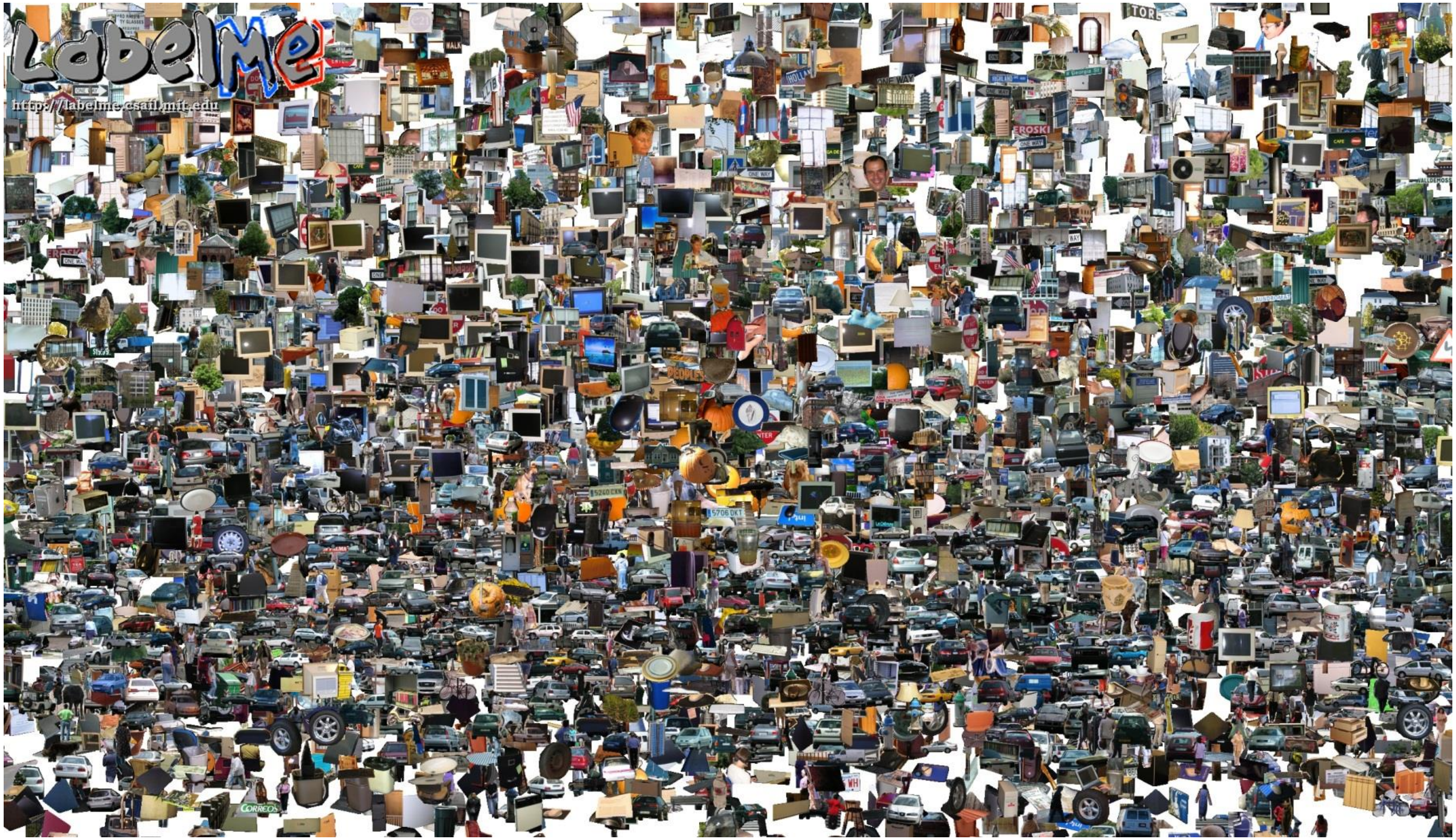
NECUIUC_CLS-DTCT

PASCAL VOC: 2010-2014



Shuicheng Yan

Opportunities of Scale



Many slides from James Hays, Alyosha Efros, and Derek Hoiem

Graphic from Antonio Torralba

Computer Vision so far

- The geometry of image formation
  - Ancient / Renaissance
- Signal processing / Convolution
  - 1800, but really the 50's and 60's
- Hand-designed Features for recognition, either instance-level or categorical
  - 1999 (SIFT), 2003 (Video Google), 2005 (Dalal-Triggs), 2006 (spatial pyramid)
- Learning from Data
  - 1991 (EigenFaces) but late 90's to now especially

What has changed in the last decade?

- The Internet
- Crowdsourcing
- Learning representations from the data these sources provide (deep learning)

Google and massive data-driven algorithms

A.I. for the postmodern world:

- all questions have already been answered…many times, in many ways
- Google is dumb, the “intelligence” is <u>in the data</u>

The Unreasonable Effectiveness of Data

Peter Norvig
Google

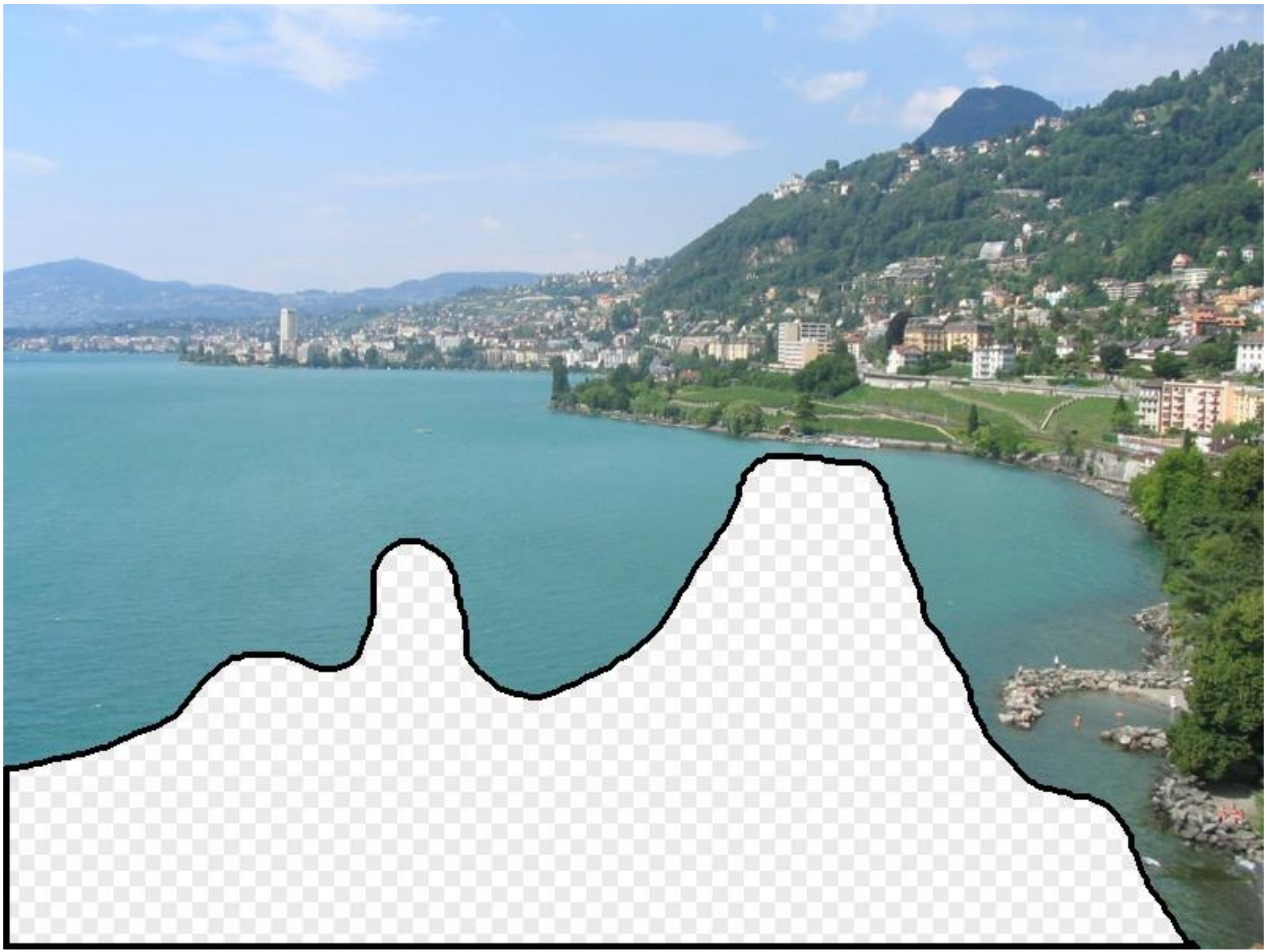
Peter Norvig

The Unreasonable
Effectiveness of Data

Big Idea

- Do we need computer vision systems to have strong AI-like reasoning about our world?
- What if invariance / generalization isn’t actually the core difficulty of computer vision?
- What if we can perform high level reasoning with brute-force, data-driven algorithms?

What should the missing region contain?



Which is the original?

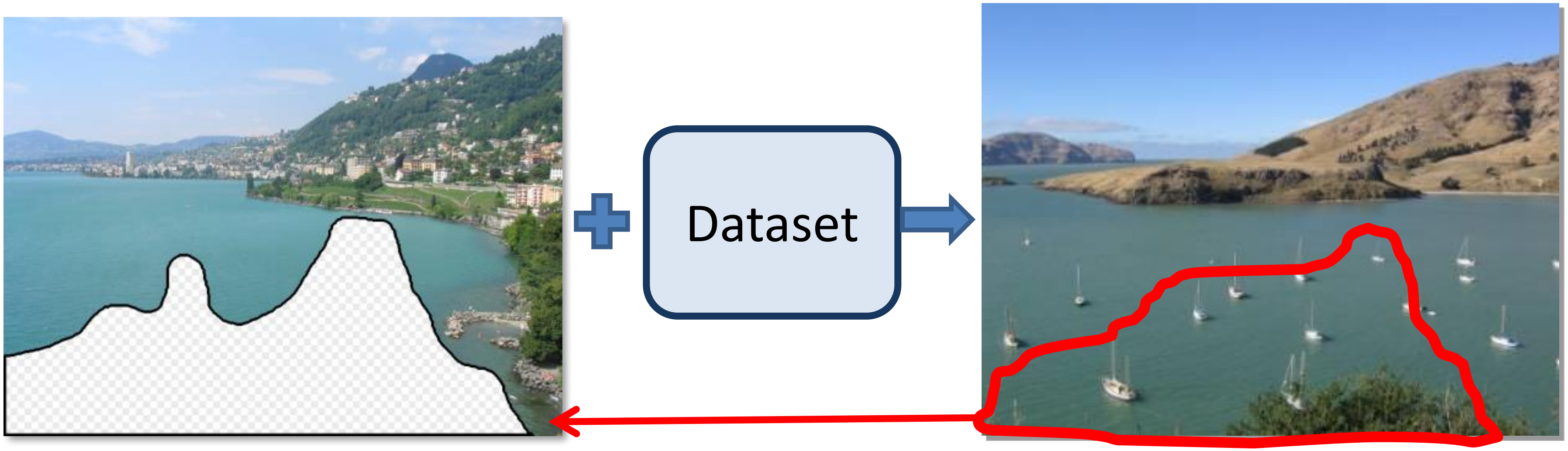
(a)

(b)

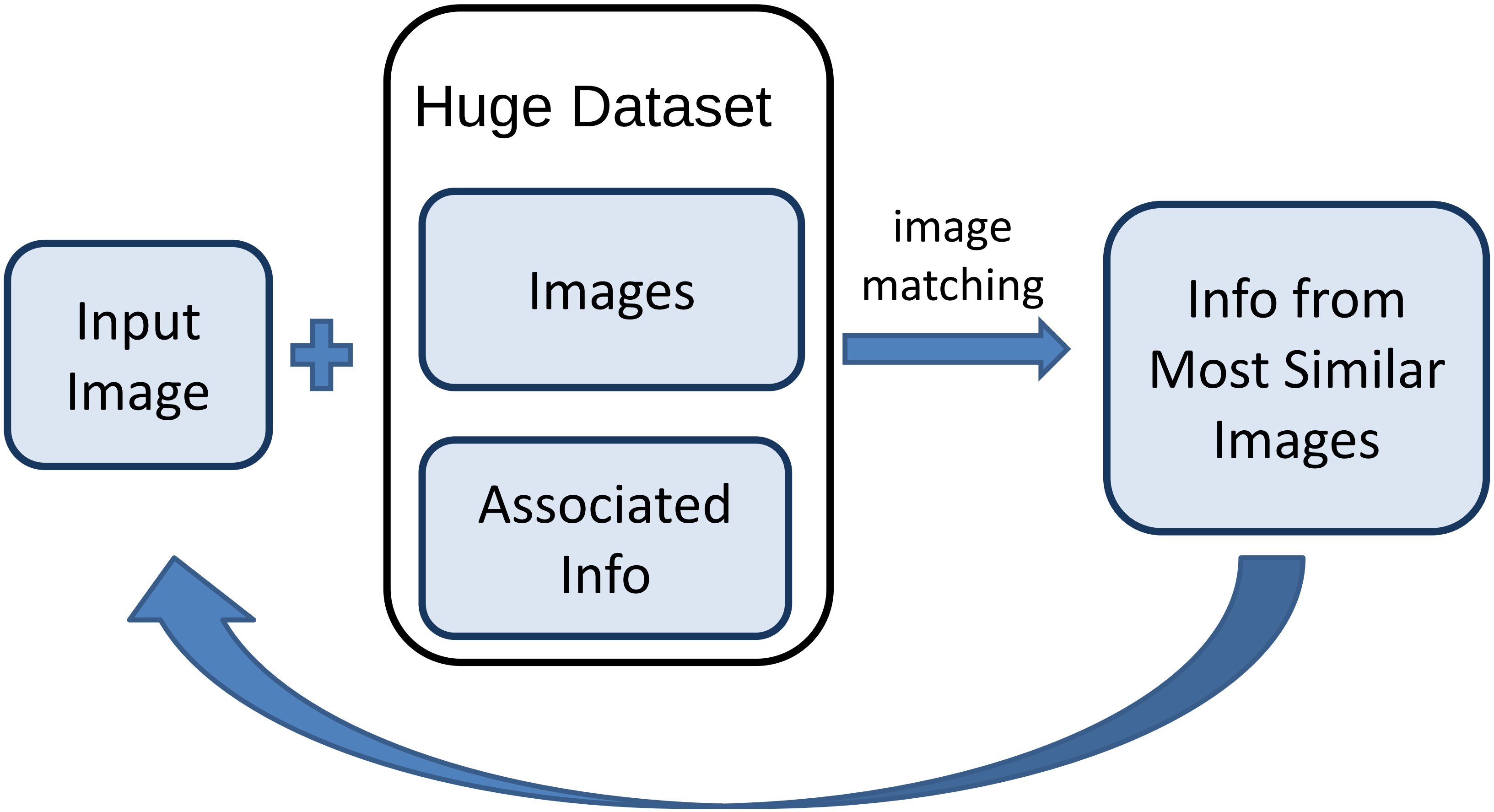
(c)

How it works

- Find a similar image from a large dataset
- Blend a region from that image into the hole

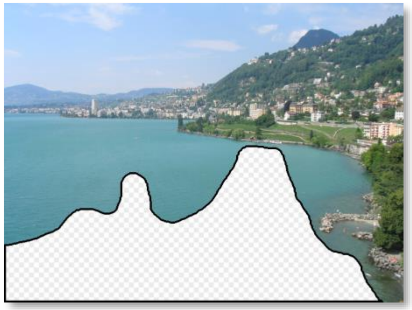


General Principal

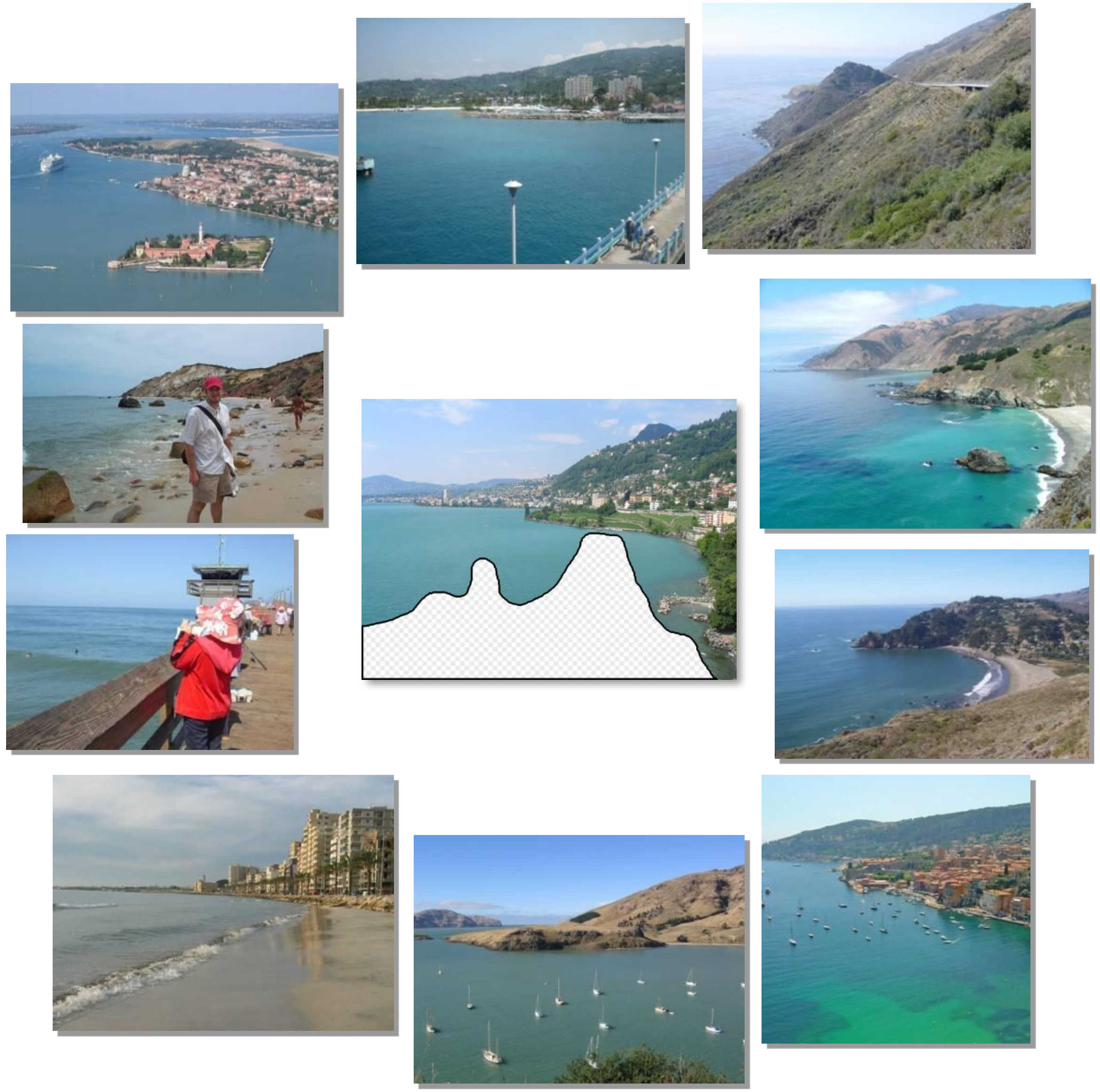


Hopefully, If you have enough images, the dataset will contain very similar images that you can find with simple matching methods.

How many images is enough?



Nearest neighbors from a
collection of 20 thousand images



Nearest neighbors from a collection of 2 million images

Image Data on the Internet

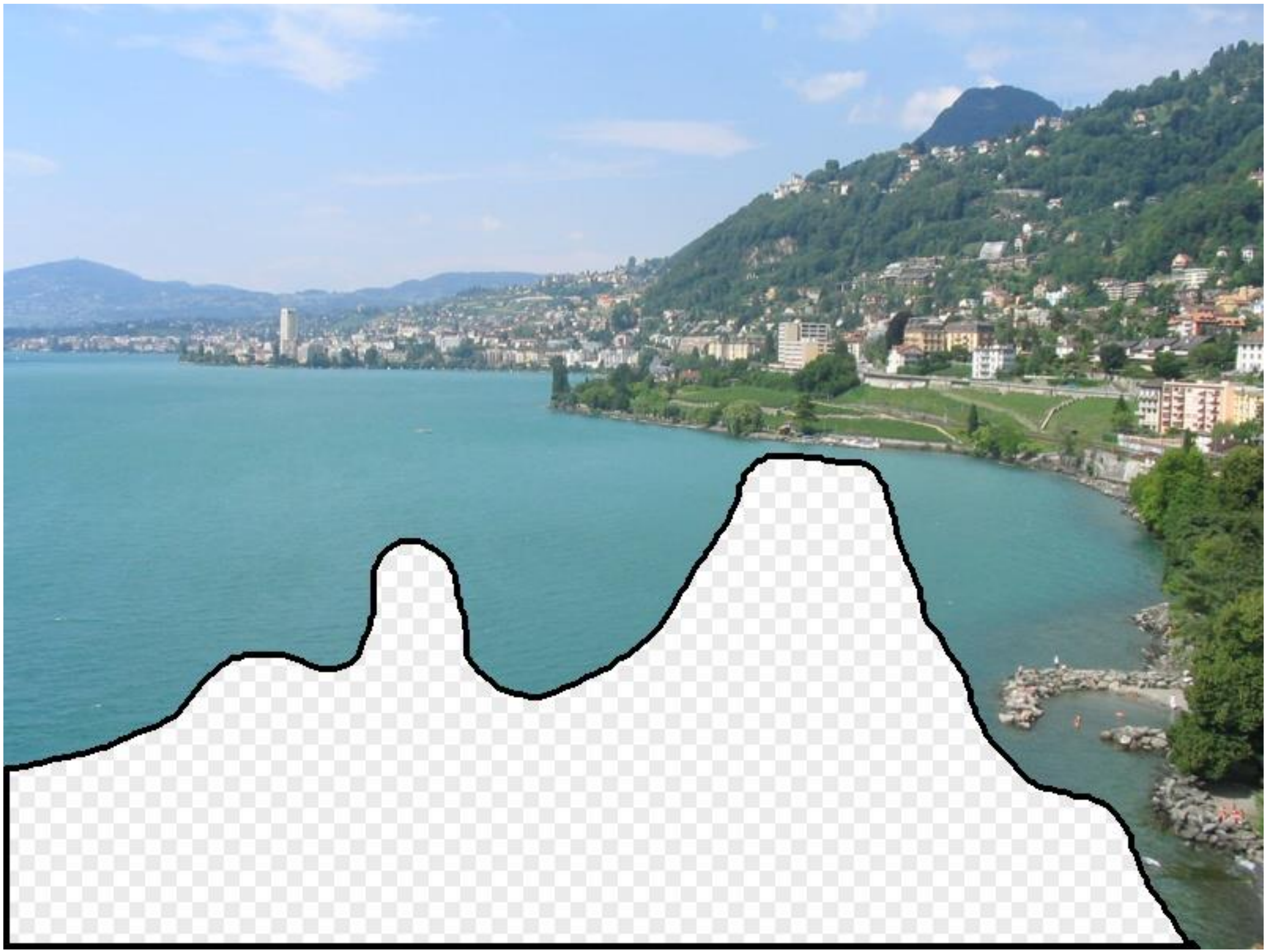
- Flickr (as of Nov 2013)
  - 10 billion photographs
  - 100+ million geotagged images
  - 3.5 million a day
- Facebook (as of Sept 2013)
  - 250 billion+
  - 300 million a day
- Instagram
  - 55 million a day

Image completion: how it works

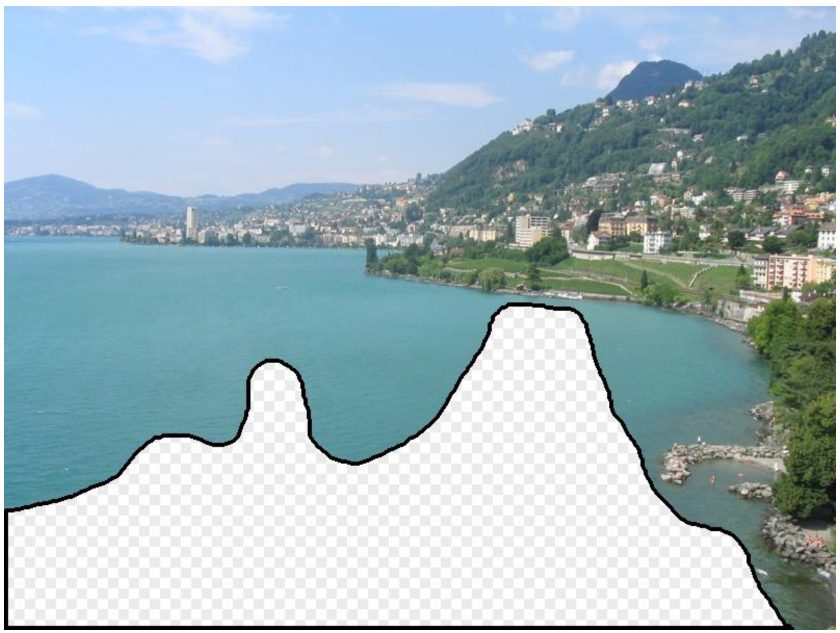
[Hays and Efros. Scene Completion Using Millions of Photographs. SIGGRAPH 2007 and CACM October 2008.]

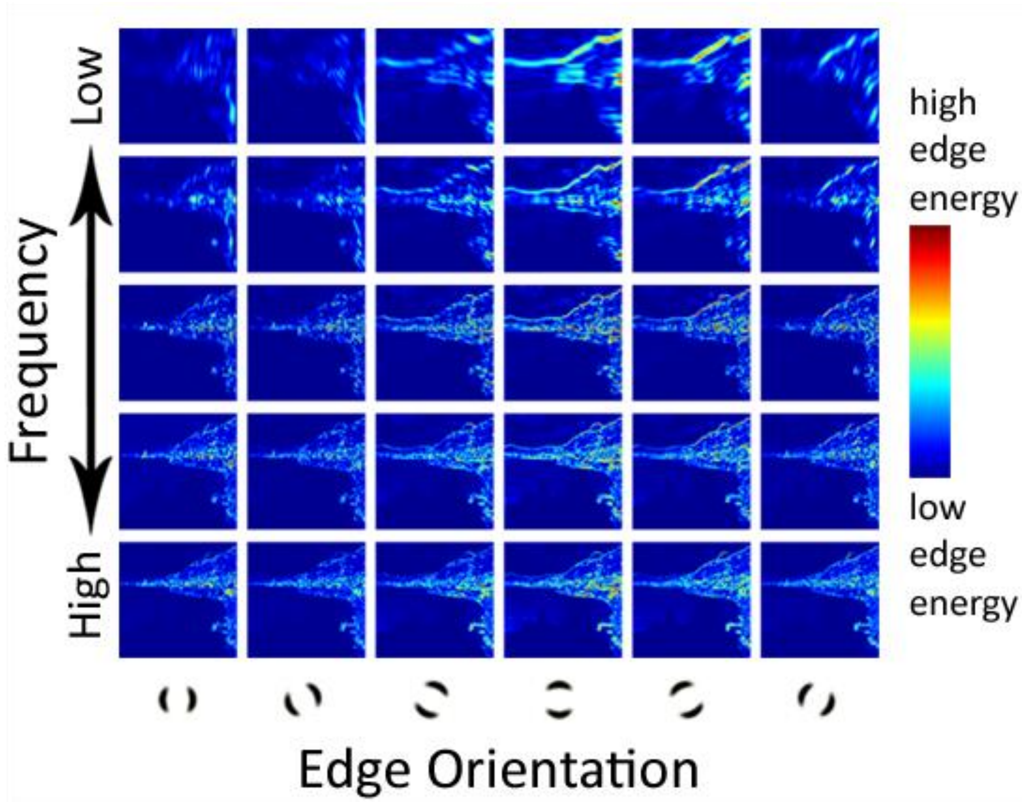
The Algorithm

Scene Matching

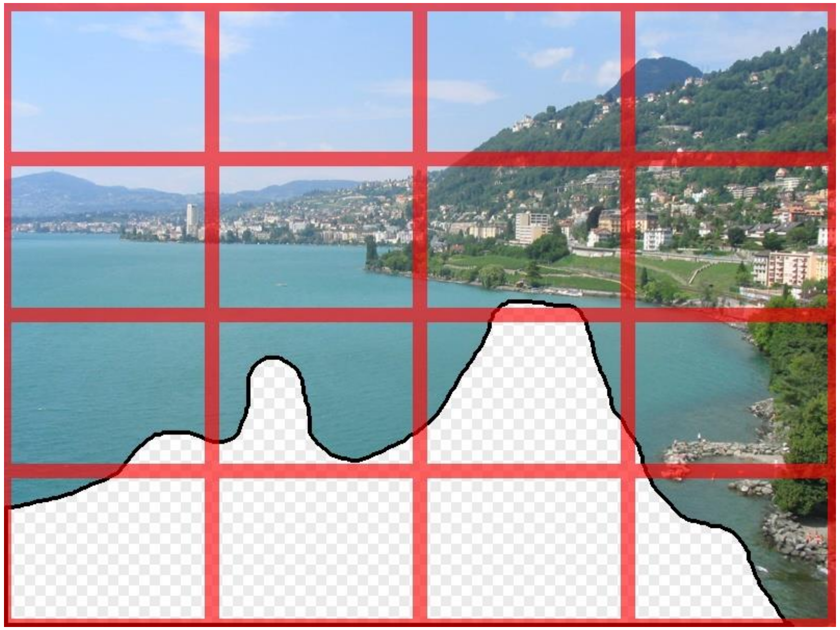


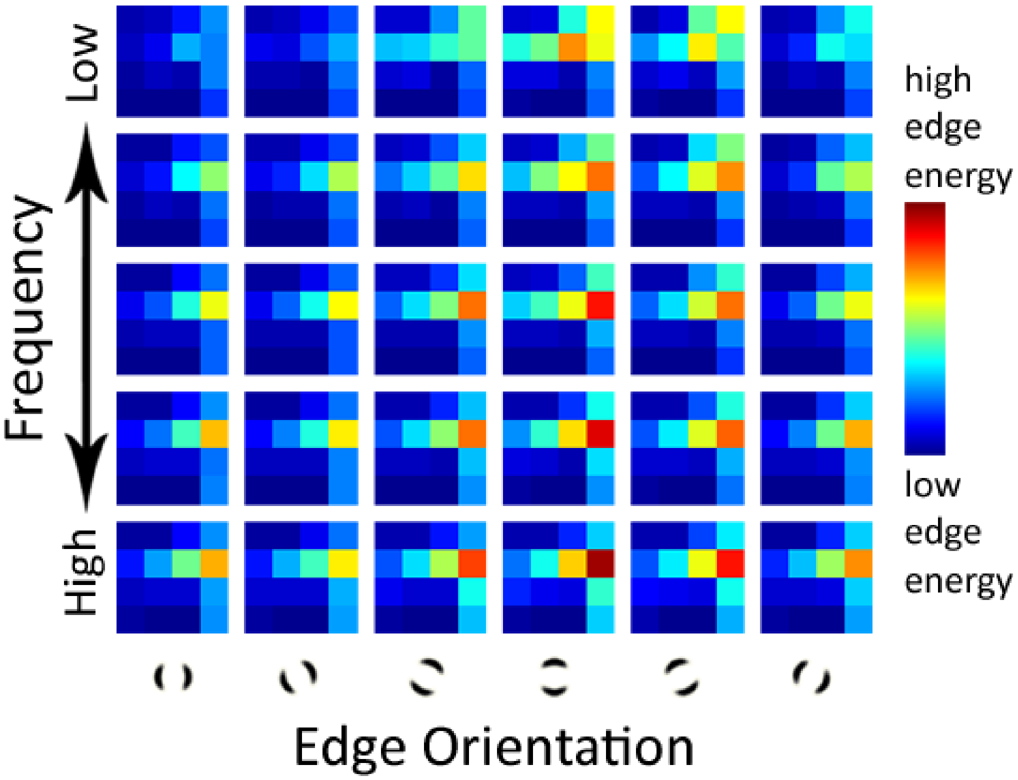
Scene Descriptor





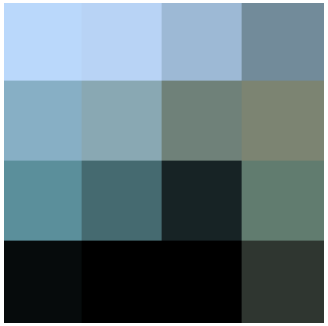
Scene Descriptor



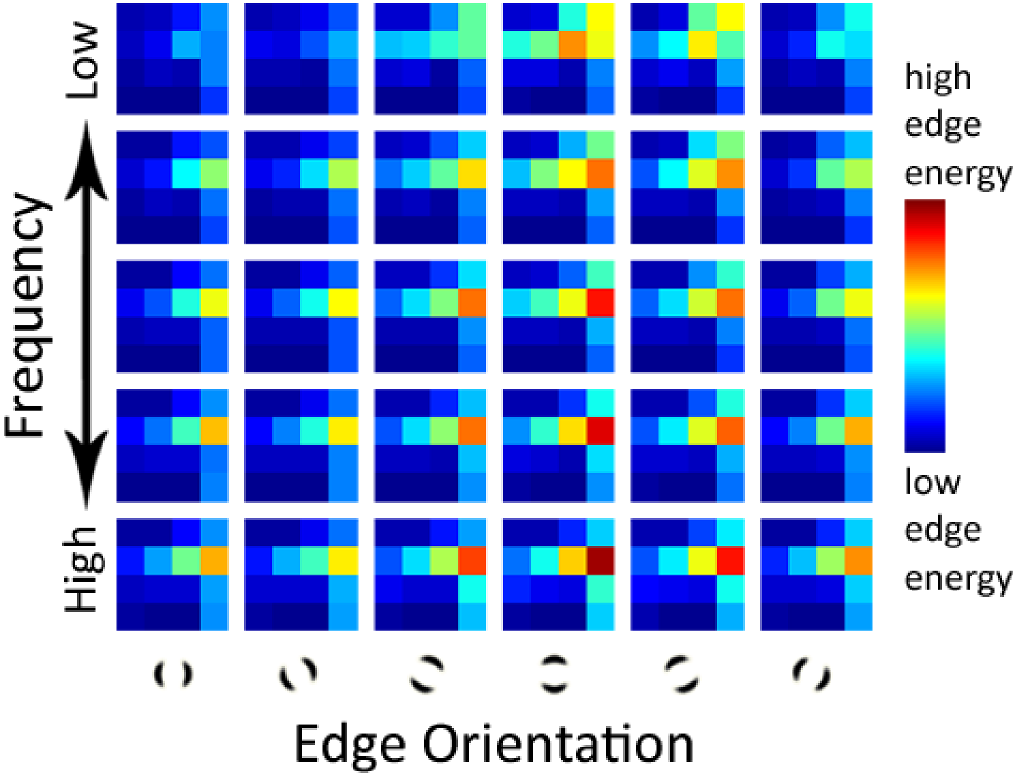


Scene Gist Descriptor
(Oliva and Torralba 2001)

Scene Descriptor

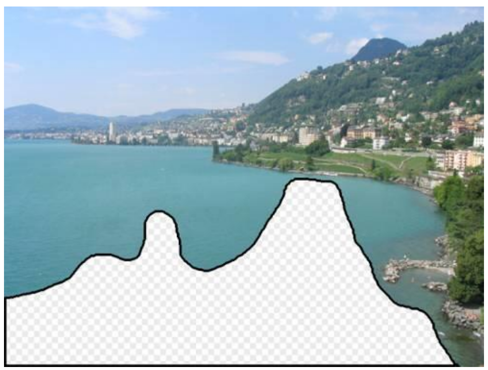


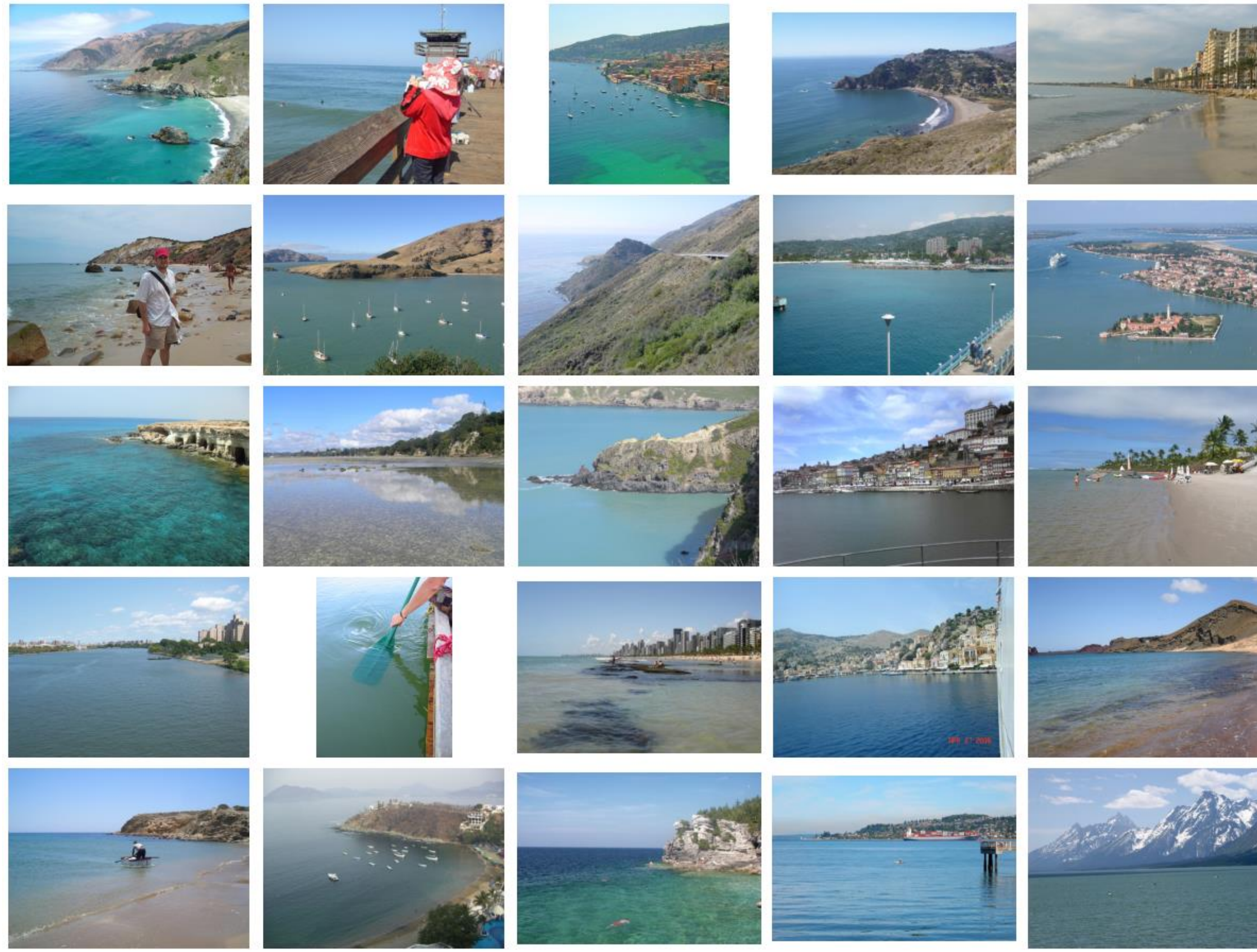
+



Scene Gist Descriptor
(Oliva and Torralba 2001)

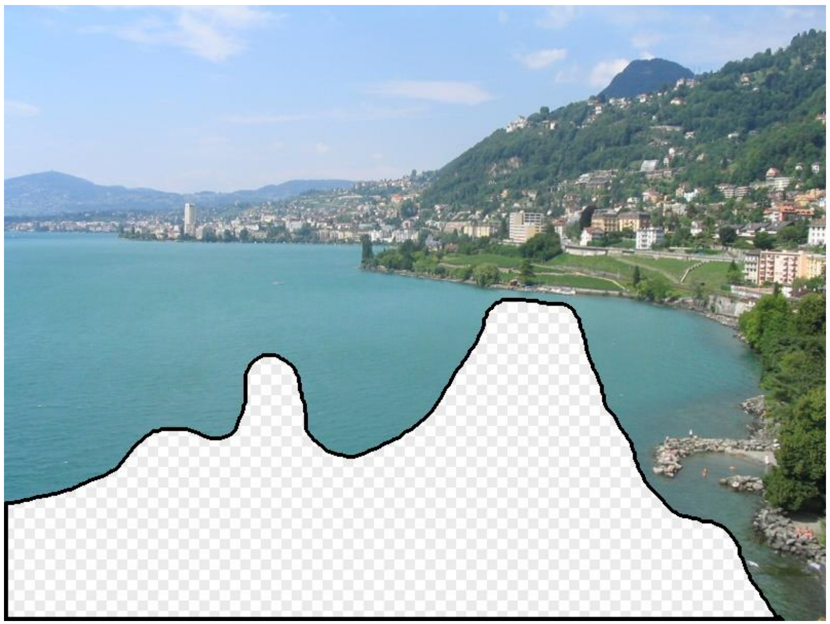
2 Million Flickr Images





... 200 total

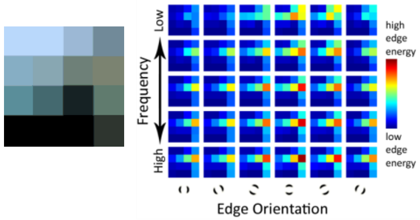
Context Matching



Graph cut + Poisson blending

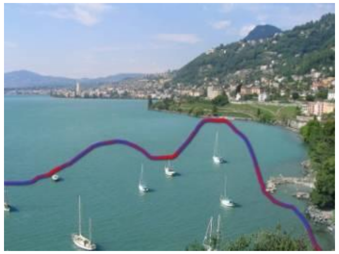
Result Ranking

We assign each of the 200 results a score which is the sum of:

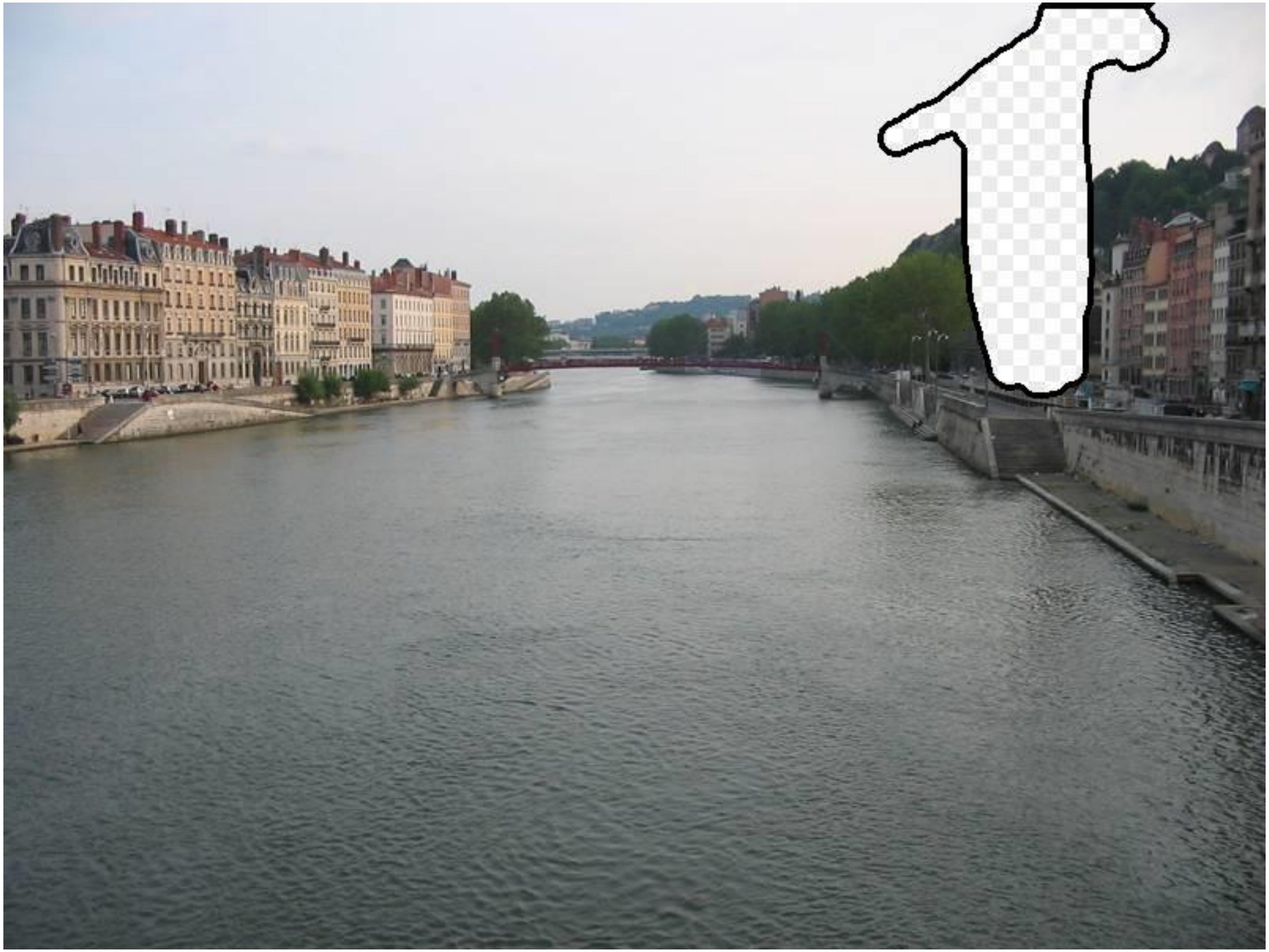


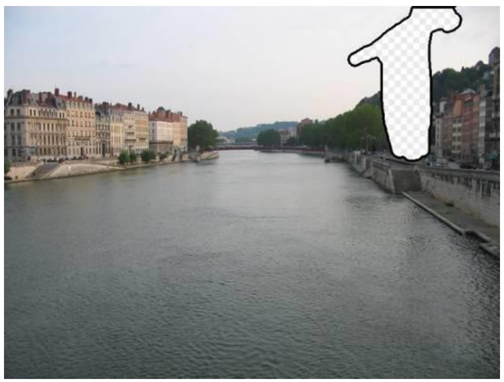
The scene matching distance

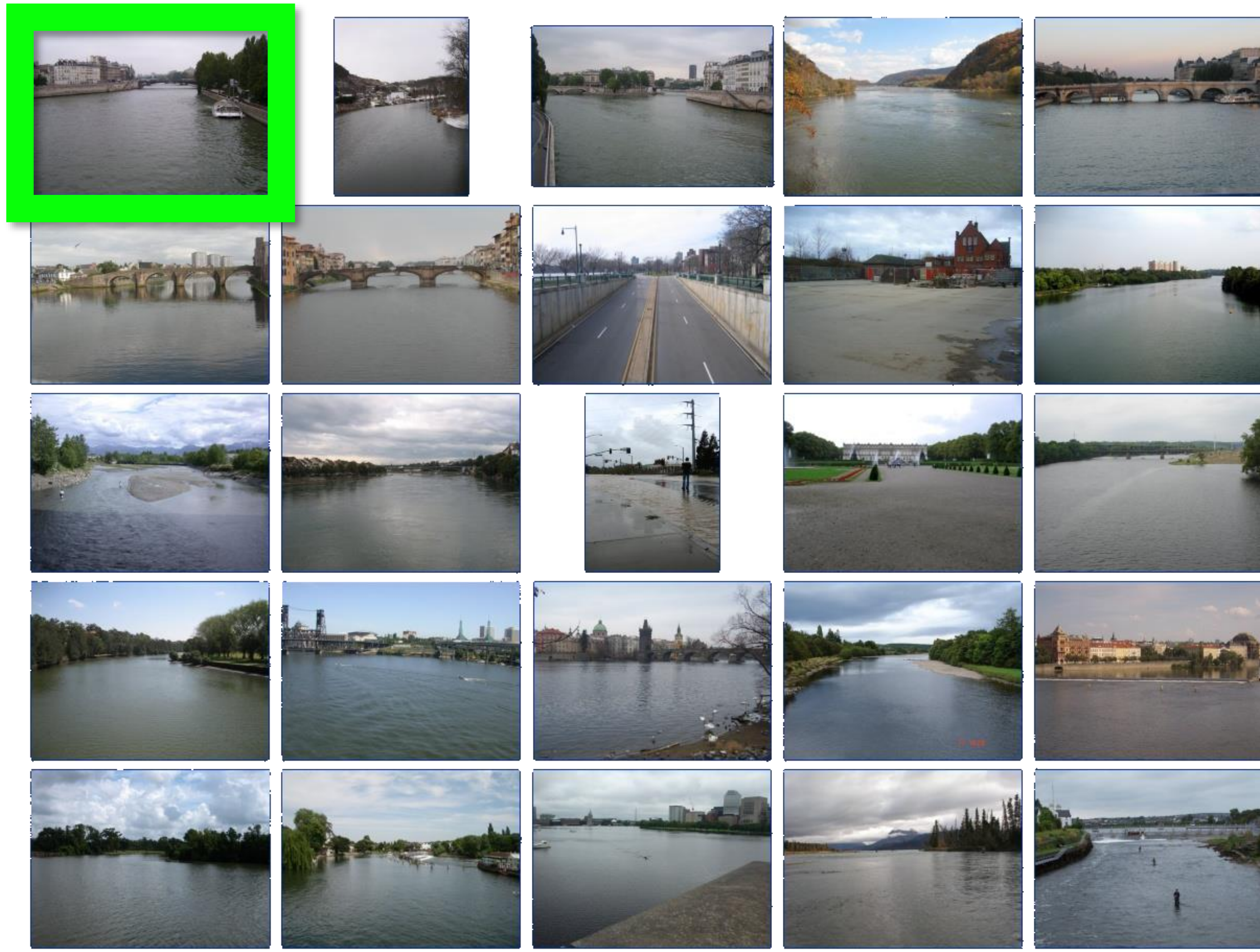
The context matching distance
(color + texture)



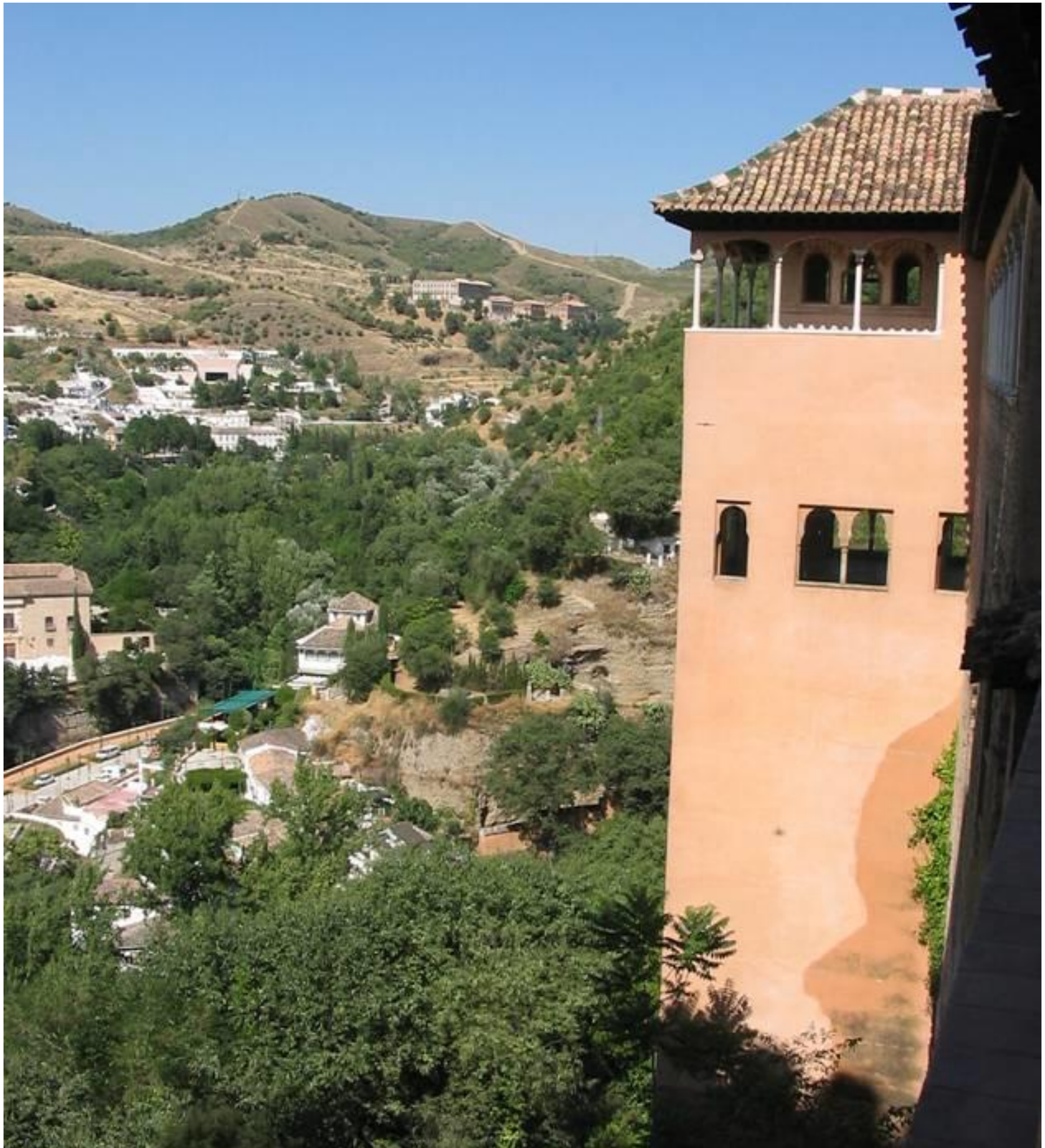
The graph cut cost

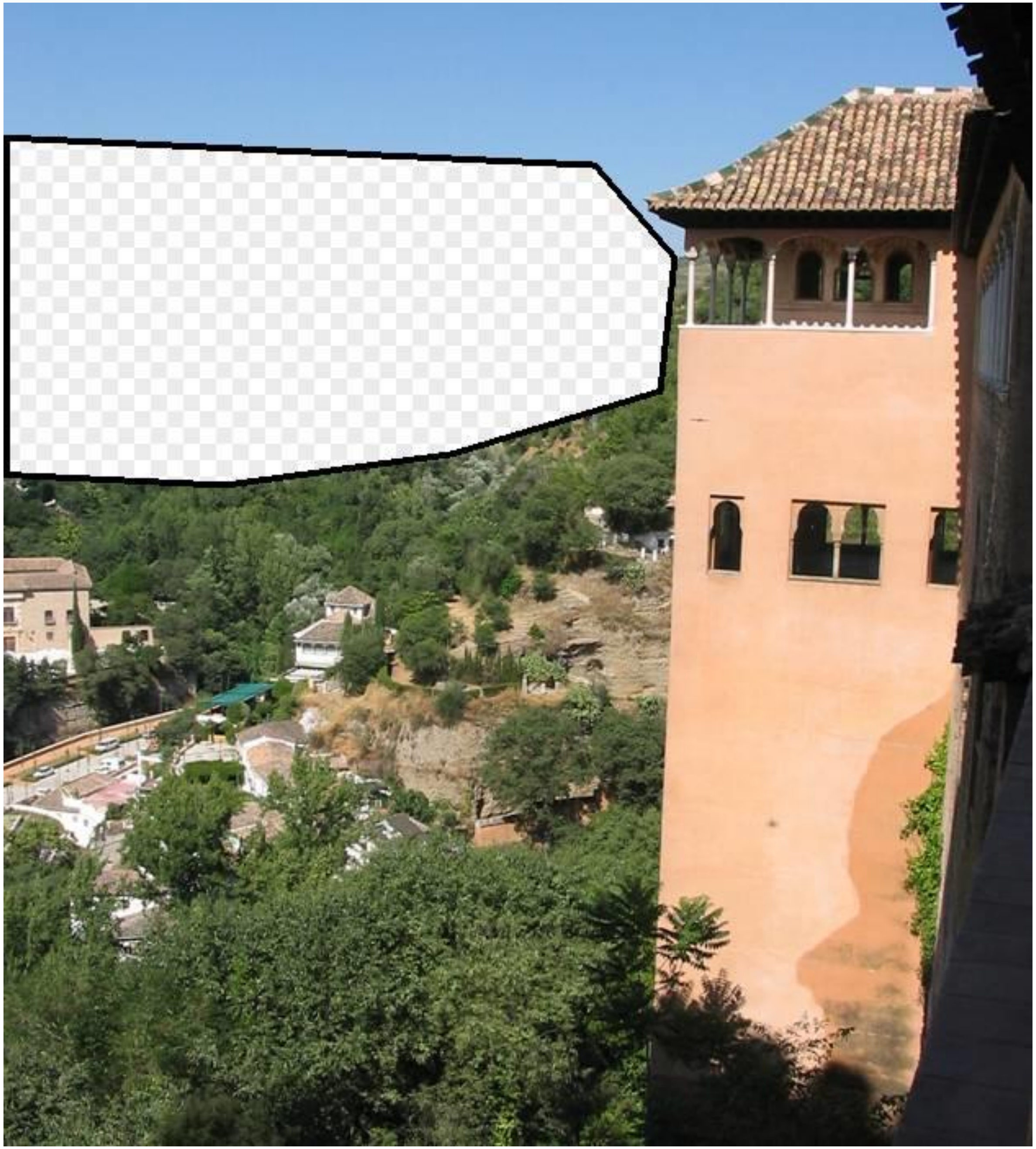






… 200 scene matches





Which is the original?