Introduction to Visual Computing

Prof. Feng Liu

Winter 2020

http://www.cs.pdx.edu/~fliu/courses/cs410/

03/03/2020

Last Time

- Machine learning approach to object recognition
  - Classifiers
  - Bag-of-features models

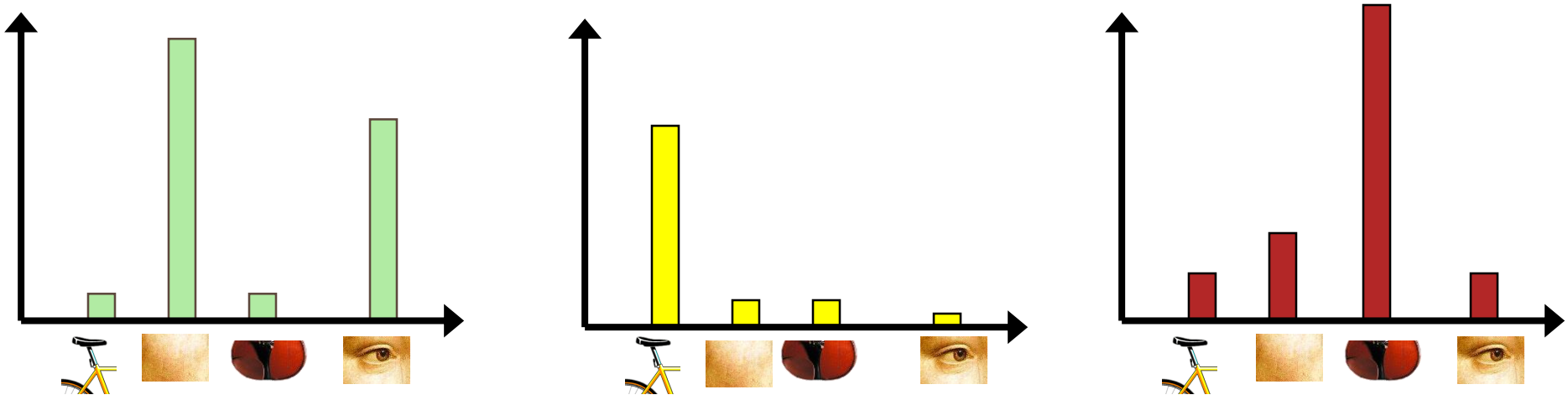
Today

- Image classification
- Visual Saliency
- In-class project presentation on March 10 and 12
  - 6 minutes, including 1-minute Q&A session
    - https://docs.google.com/spreadsheets/d/1kboeAkmbJovuyOIVRK4ku2Z-drmKzdaMKn06tXij6sc/edit#gid=0
  - If you cannot attend, make a video record of your presentation and submit it before your scheduled presentation time

The slides for the topic of image classification are mostly used from Prof. S. Lazebnik.

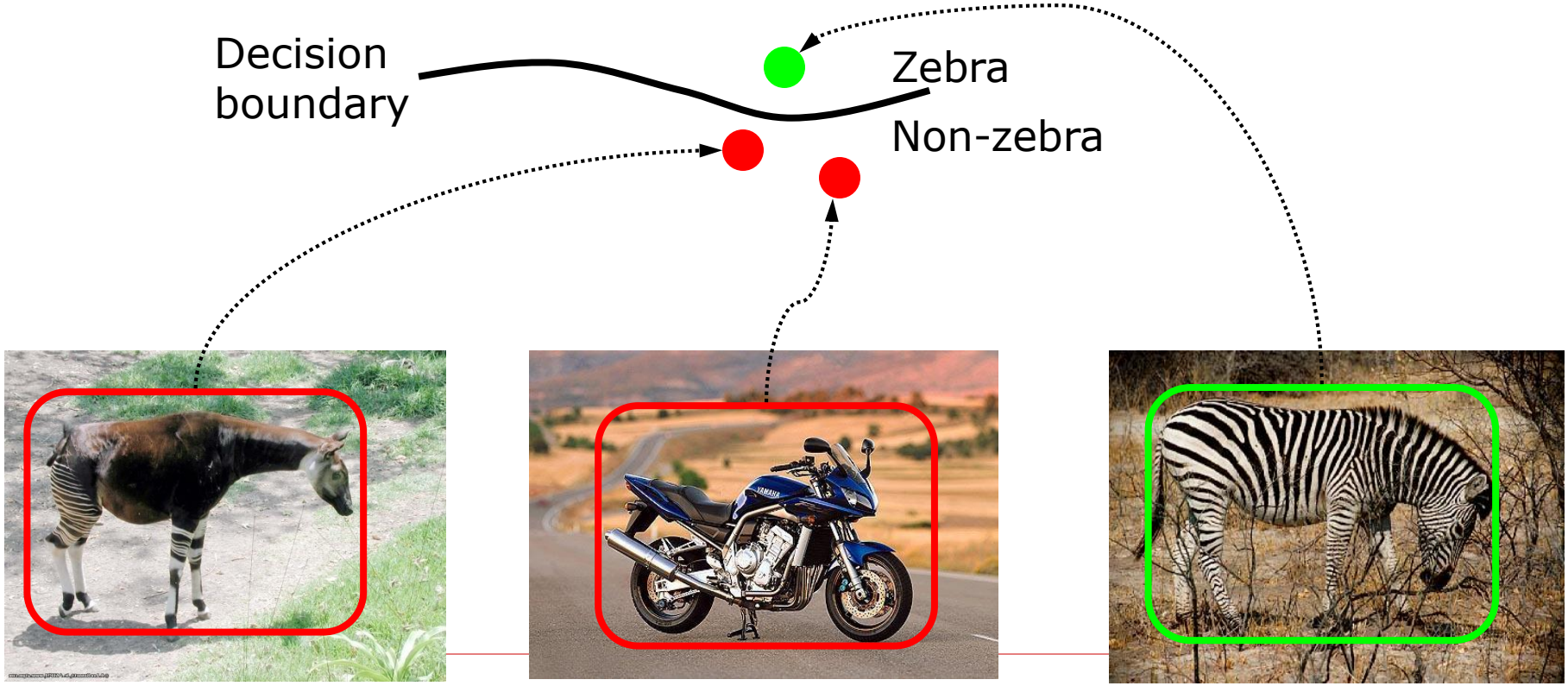
Image classification

- Given the bag-of-features representations of images from different classes, how do we learn a model for distinguishing them?



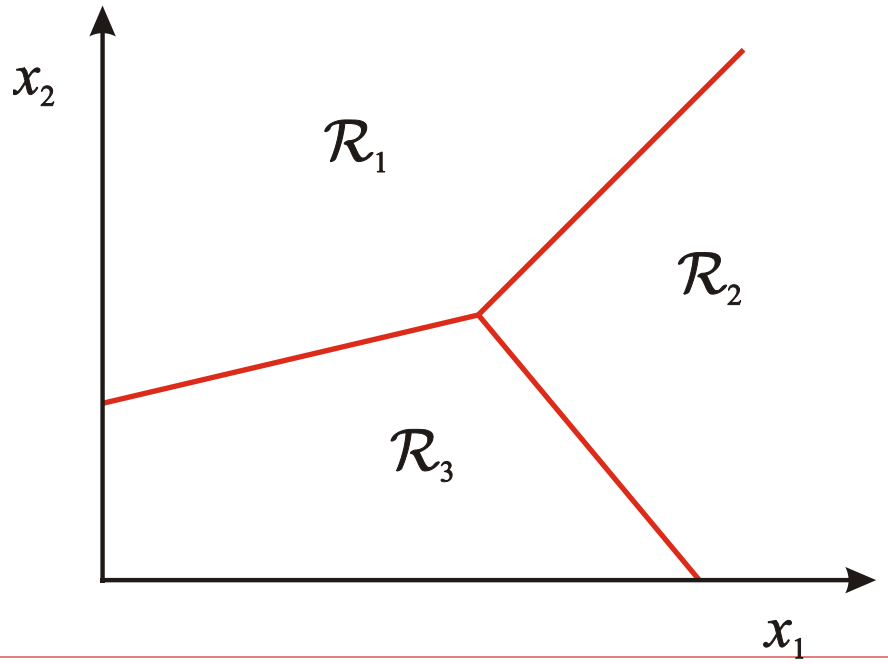
Classifiers

- Learn a decision rule assigning bag-of-features representations of images to different classes



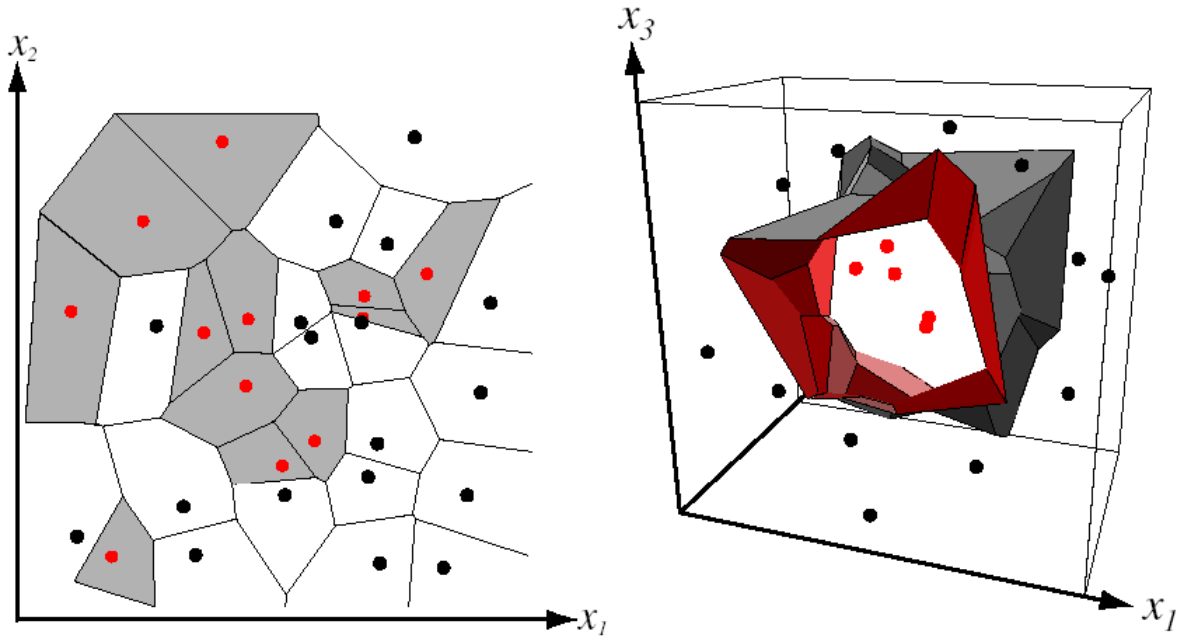
Classification

- Assign input vector to one of two or more classes
- Any decision rule divides input space into *decision regions* separated by *decision boundaries*



Nearest Neighbor Classifier

- Assign label of nearest training data point to each test data point



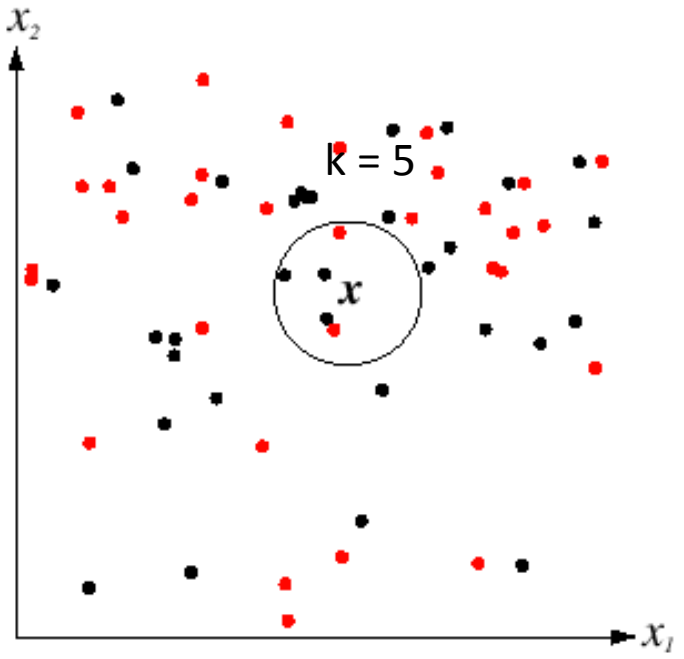
from Duda *et al.*

Voronoi partitioning of feature space
for two-category 2D and 3D data

Source: D. Lowe

K-Nearest Neighbors

- For a new point, find the k closest points from training data
- Labels of the k points "vote" to classify
- Works well provided there is lots of data and the distance function is good



Source: D. Lowe

Functions for comparing histograms

- L1 distance: $$D(h_1, h_2) = \sum_{i=1}^{N} | h_1(i) - h_2(i) |$$
- χ^2 distance: $$D(h_1, h_2) = \sum_{i=1}^{N} \frac{(h_1(i) - h_2(i))^2}{h_1(i) + h_2(i)}$$
- Quadratic distance (*cross-bin distance*):

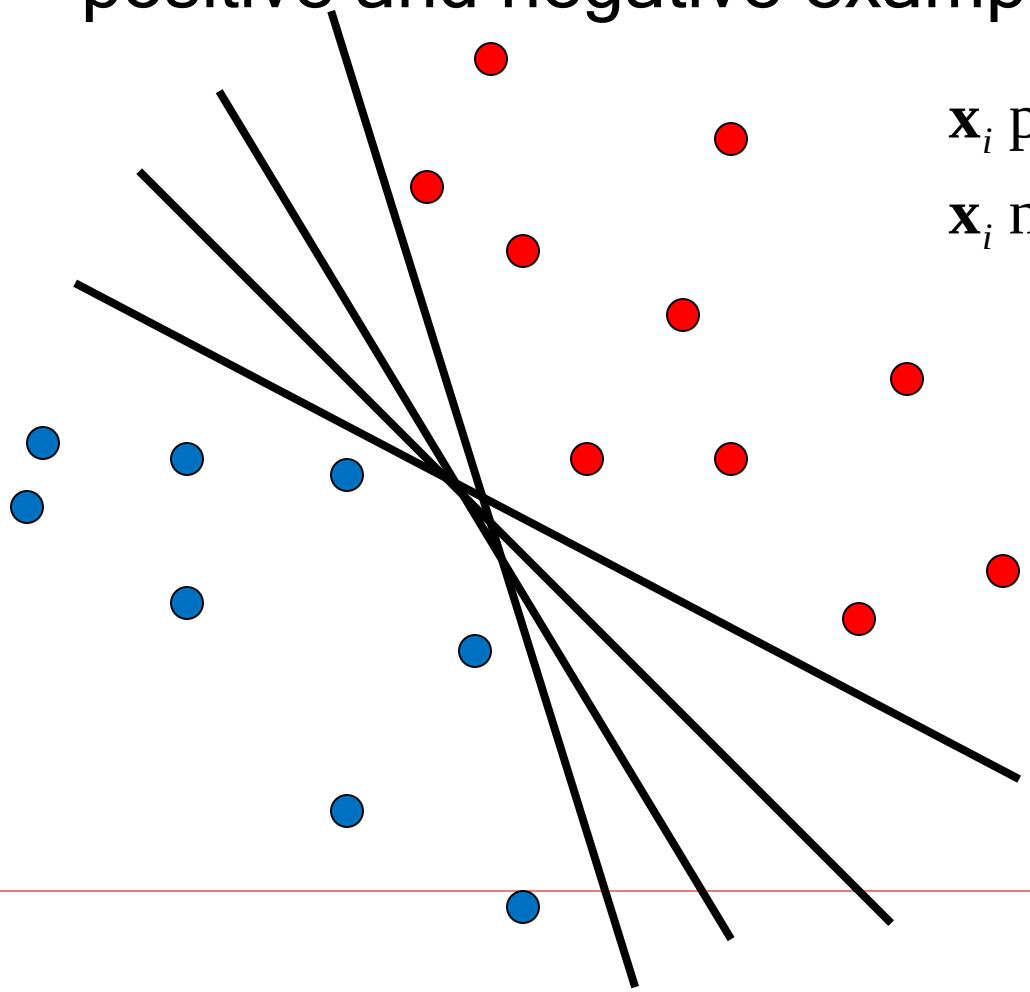
$$D(h_1, h_2) = \sum_{i,j} A_{ij} (h_1(i) - h_2(j))^2$$

- Histogram intersection (similarity function):

$$I(h_1, h_2) = \sum_{i=1}^{N} \min(h_1(i), h_2(i))$$

Linear classifiers

- Find linear function (*hyperplane*) to separate positive and negative examples



$\mathbf{x}_i$ positive: $\mathbf{x}_i \cdot \mathbf{w} + b \geq 0$

$\mathbf{x}_i$ negative : $\mathbf{x}_i \cdot \mathbf{w} + b < 0$

Which hyperplane is best?

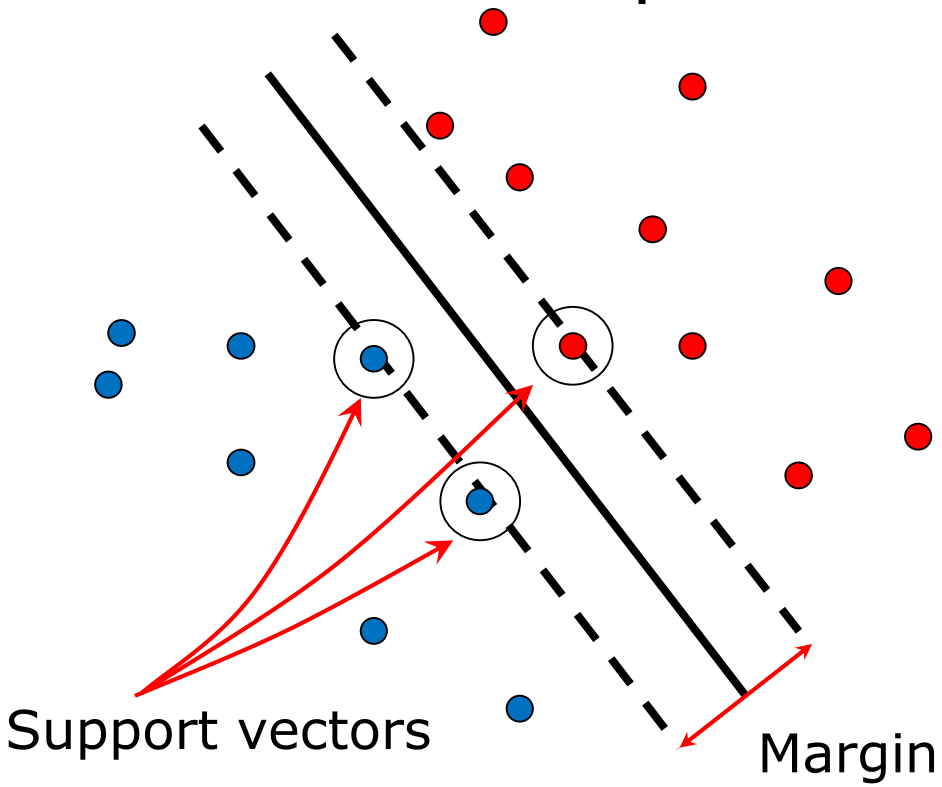
Support vector machines

- Find hyperplane that maximizes the *margin* between the positive and negative examples

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Support vector machines

- Find hyperplane that maximizes the *margin* between the positive and negative examples



$\mathbf{x}_i$ positive $(y_i = 1)$: $\mathbf{x}_i \cdot \mathbf{w} + b \geq 1$

$\mathbf{x}_i$ negative $(y_i = -1)$: $\mathbf{x}_i \cdot \mathbf{w} + b \leq -1$

For support vectors, $\mathbf{x}_i \cdot \mathbf{w} + b = \pm 1$

Distance between point and hyperplane: $\dfrac{|\mathbf{x}_i \cdot \mathbf{w} + b|}{\|\mathbf{w}\|}$

Therefore, the margin is $2 / \|\mathbf{w}\|$

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Finding the maximum margin hyperplane

1. Maximize margin $2/\|\mathbf{w}\|$
2. Correctly classify all training data:

$$\mathbf{x}_i \text{ positive} (y_i = 1): \quad \mathbf{x}_i \cdot \mathbf{w} + b \geq 1$$

$$\mathbf{x}_i \text{ negative} (y_i = -1): \quad \mathbf{x}_i \cdot \mathbf{w} + b \leq -1$$

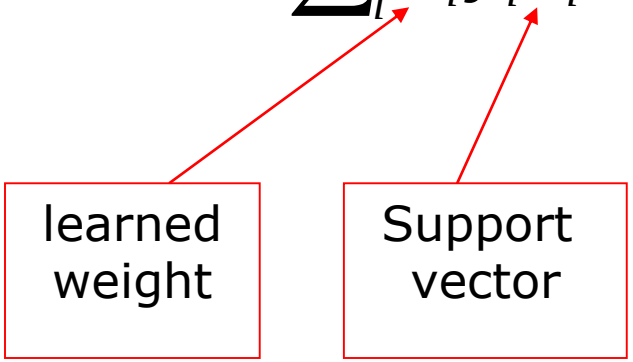
- *Quadratic optimization problem*:
- Minimize $\frac{1}{2} \mathbf{w}^T \mathbf{w}$

  Subject to $y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1$

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Finding the maximum margin hyperplane

- Solution: $\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$



C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Finding the maximum margin hyperplane

- Solution: $\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$

  $b = y_i - \mathbf{w} \cdot \mathbf{x}_i$ for any support vector

- Classification function (decision boundary):

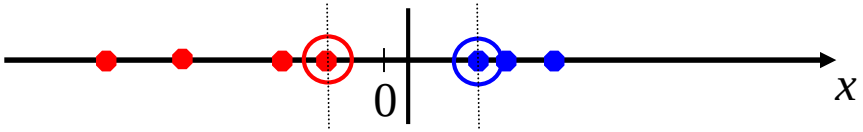
$$\mathbf{w} \cdot \mathbf{x} + b = \sum_i \alpha_i y_i \mathbf{x}_i \cdot \mathbf{x} + b$$

- Notice that it relies on an *inner product* between the test point $\boldsymbol{x}$ and the support vectors $\boldsymbol{x}_i$
- Solving the optimization problem also involves computing the inner products $\boldsymbol{x}_i \cdot \boldsymbol{x}_j$ between all pairs of training points

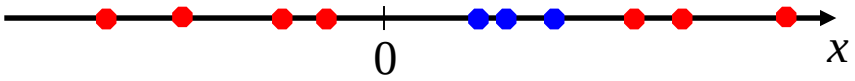
C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Nonlinear SVMs

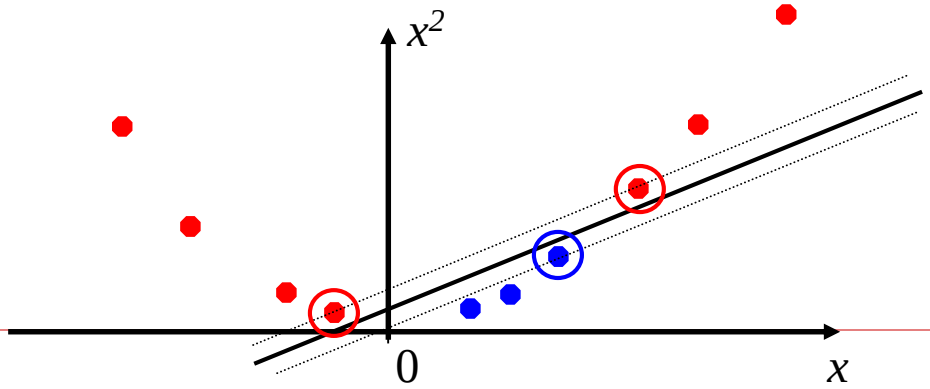
- Datasets that are linearly separable work out great:



- But what if the dataset is just too hard?



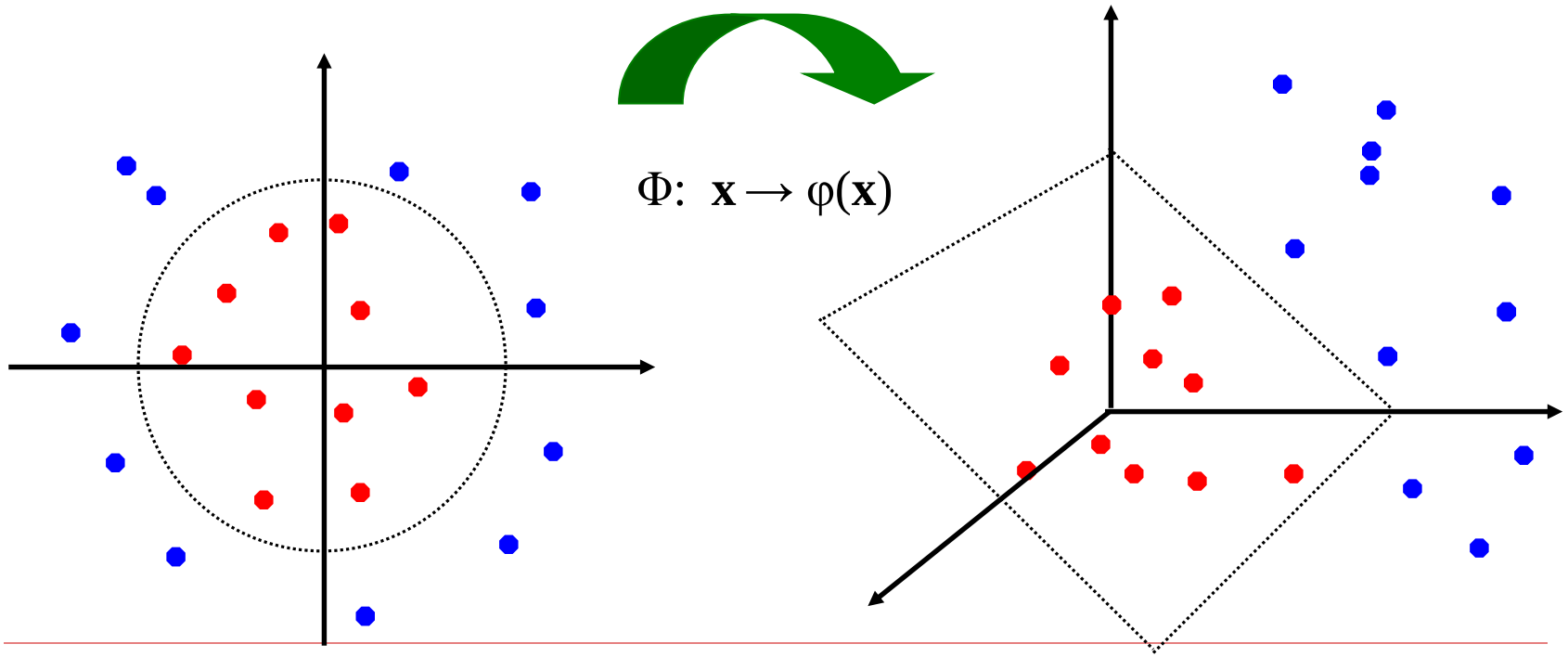
- We can map it to a higher-dimensional space:



Slide credit: Andrew Moore

Nonlinear SVMs

- General idea: the original input space can always be mapped to some higher-dimensional feature space where the training set is separable:



Slide credit: Andrew Moore

Nonlinear SVMs

- *The kernel trick*: instead of explicitly computing the lifting transformation $\varphi(\mathbf{x})$, define a kernel function K such that

$$K(\mathbf{x}_i, \mathbf{x}_j) = \varphi(\mathbf{x}_i) \cdot \varphi(\mathbf{x}_j)$$

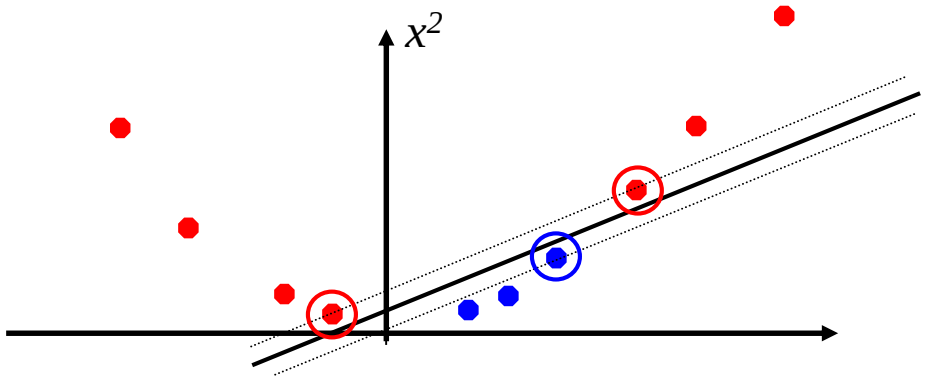
- (to be valid, the kernel function must satisfy *Mercer's condition*)
- This gives a nonlinear decision boundary in the original feature space:

$$\sum_i \alpha_i y_i \varphi(\mathbf{x}_i) \cdot \varphi(\mathbf{x}) + b = \sum_i \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b$$

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Nonlinear kernel: Example

- Consider the mapping $\varphi(x) = (x, x^2)$



$$\varphi(x) \cdot \varphi(y) = (x, x^2) \cdot (y, y^2) = xy + x^2 y^2$$

$$K(x, y) = xy + x^2 y^2$$

Summary: SVMs for image classification

1. Pick an image representation (in our case, bag of features)
2. Pick a kernel function for that representation
3. Compute the matrix of kernel values between every pair of training examples
4. Feed the kernel matrix into your favorite SVM solver to obtain support vectors and weights
5. At test time: compute kernel values for your test example and each support vector, and combine them with the learned weights to get the value of the decision function

SVMs: Pros and cons

- Pros
  - Many publicly available SVM packages: http://www.kernel-machines.org/software
  - Kernel-based framework is very powerful, flexible
  - SVMs work very well in practice, even with very small training sample sizes
- Cons
  - No “direct” multi-class SVM, must combine two-class SVMs
  - Computation, memory
    - During training time, must compute matrix of kernel values for every pair of examples
    - Learning can take a very long time for large-scale problems

Summary: Classifiers

- Nearest-neighbor and k-nearest-neighbor classifiers
  - L1 distance, χ^2 distance, quadratic distance, histogram intersection
- Support vector machines
  - Linear classifiers
  - Margin maximization
  - The kernel trick
  - Kernel functions: histogram intersection, generalized Gaussian, pyramid match
- Of course, there are many other classifiers out there
  - Neural networks, boosting, decision trees, …

Today

- Image classification
- Visual Saliency

Importance analysis

- Required in computer graphics and vision problems
  - Machine vision is still incapable of fully semantic visual understanding
  - Saliency as an alternative to semantic visual understanding, an indication of importance

Visual saliency

"Visual saliency is the distinct subjective perceptual quality which makes some items in the world stand out from their neighbors and immediately grab our attention."

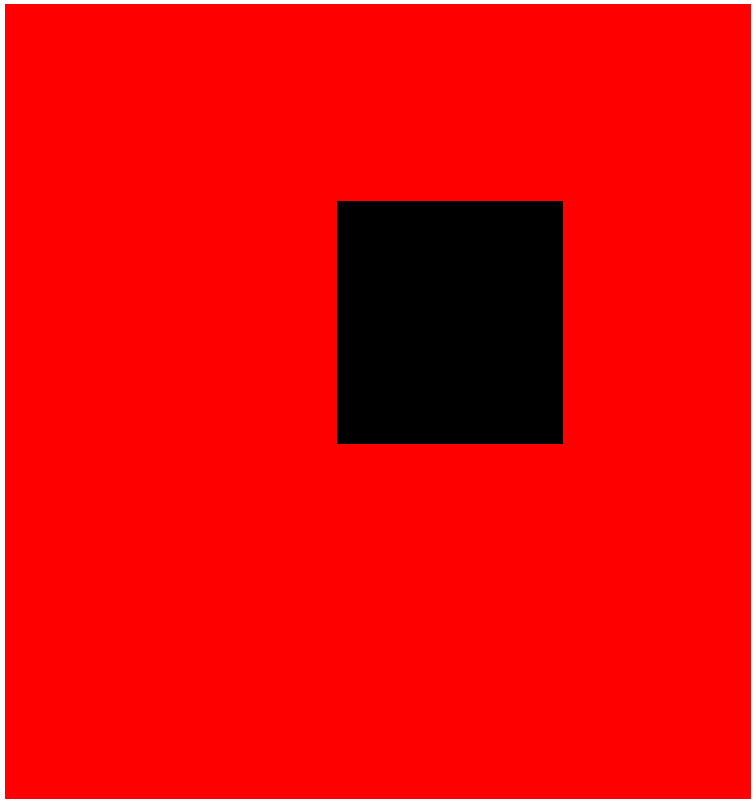
[Itti 2007]

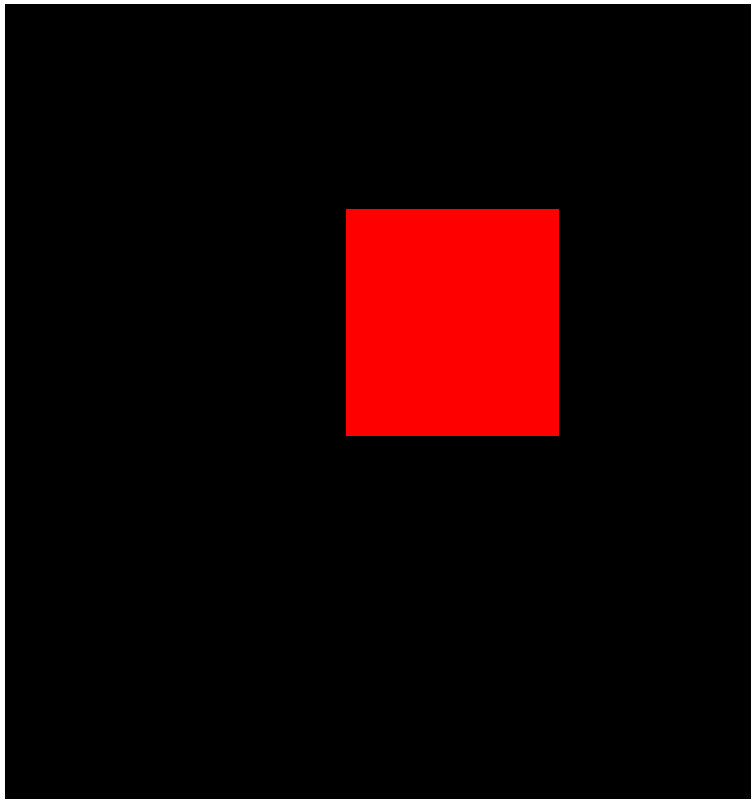
Visual saliency != Importance

Saliency measurement

- Saliency measures low-level stimuli to human visual system
- Saliency is closely related to low-level feature contrast [Lamming D. 1991, Nothdurft 2000]
  - Spatial and range features
    - Color, orientation, shape ……
  - Temporal features
    - Velocity

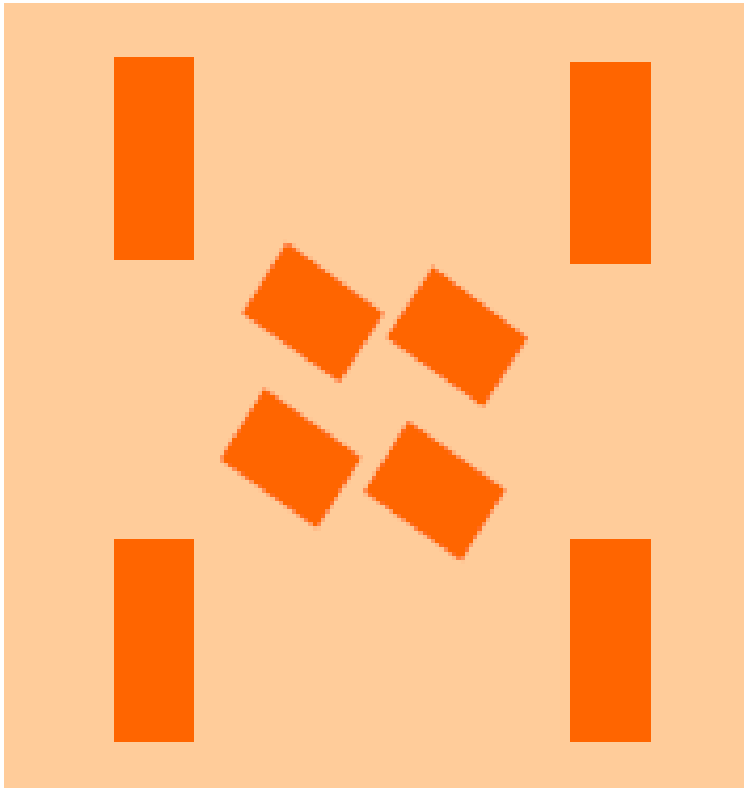
Saliency from color contrast

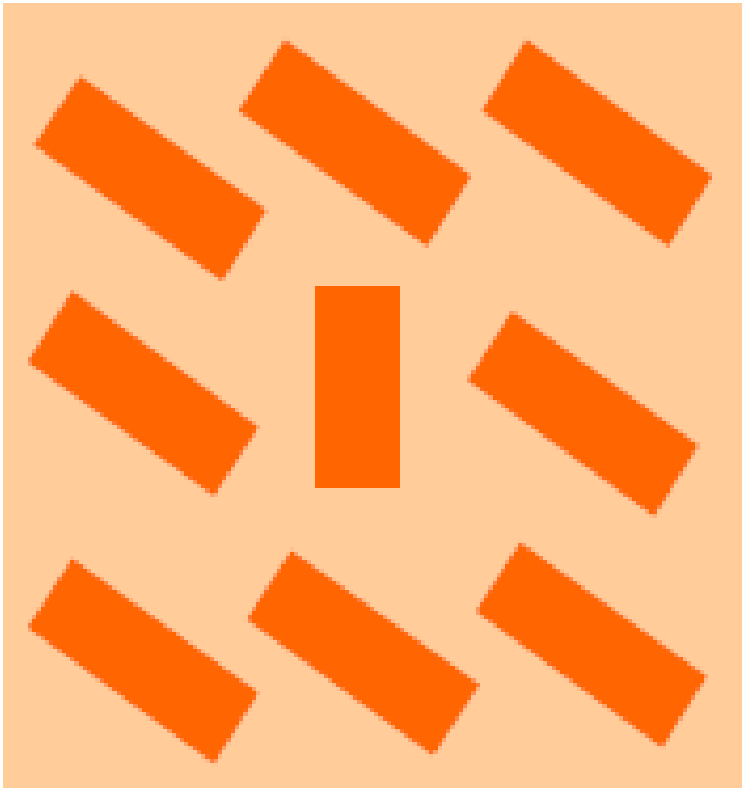




Reprint from [Ma et al. 2003]

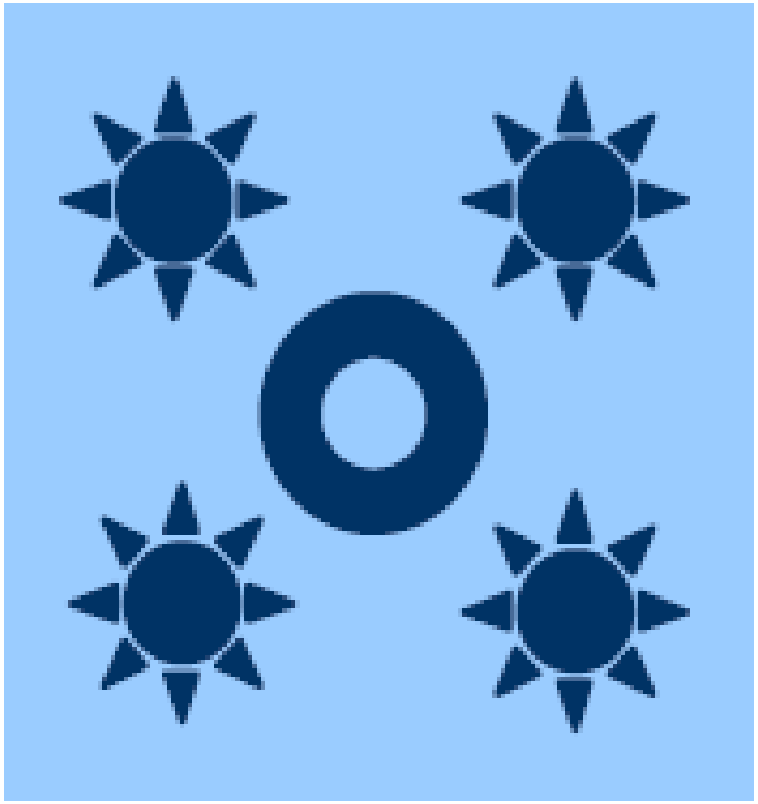
Saliency from orientation contrast

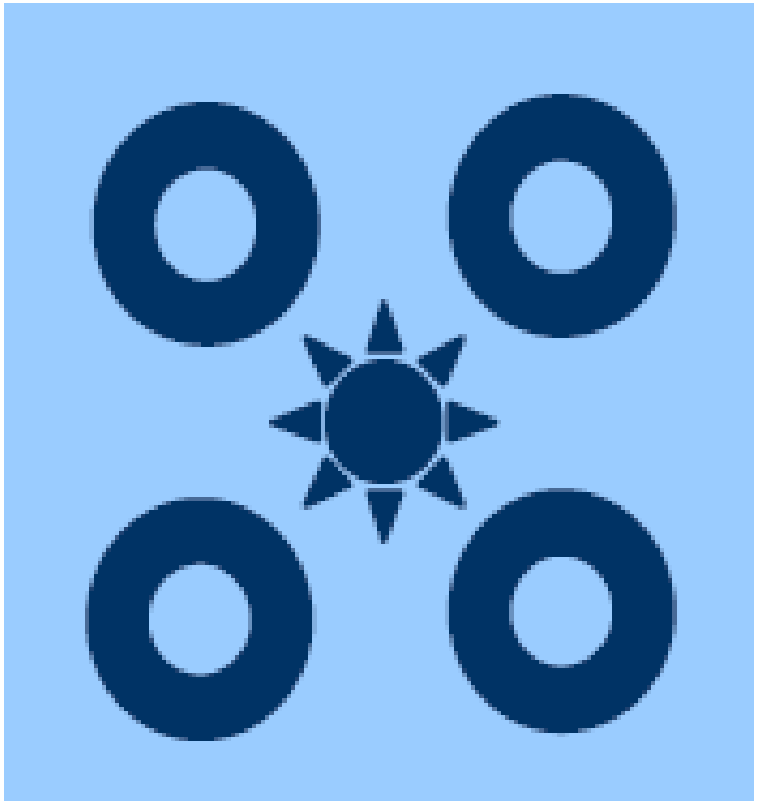




Reprint from [Ma et al. 2003]

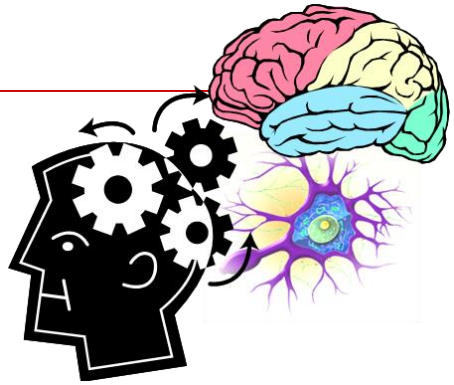
Saliency from shape contrast





Reprint from [Ma et al. 2003]

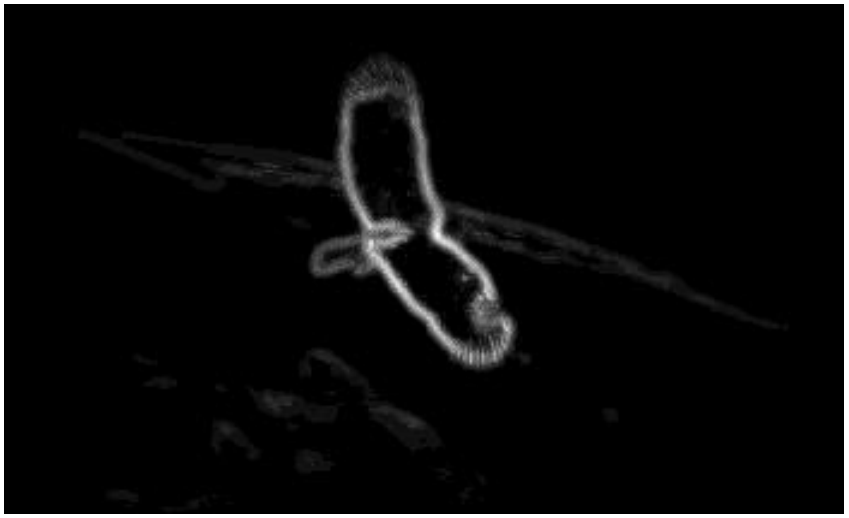
Computational visual saliency



- Biological model-based methods
  - Calculate saliency based on simplified biological model
  - Basically, calculate the feature contrast as saliency
- Data-driven methods
  - Learn a model to predict saliency from given labeled data

Pixel/patch-based methods

Original image

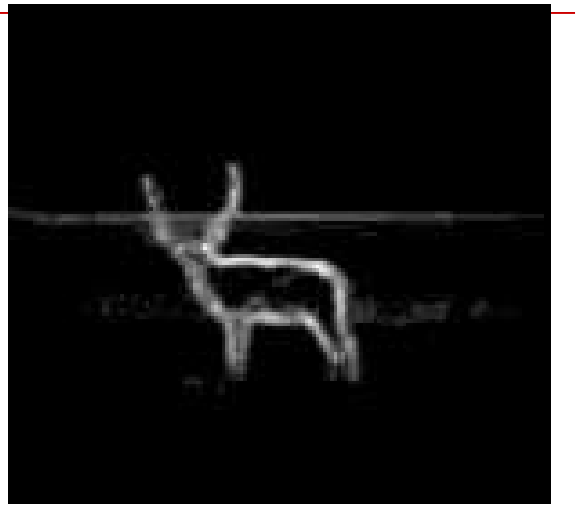


Saliency map

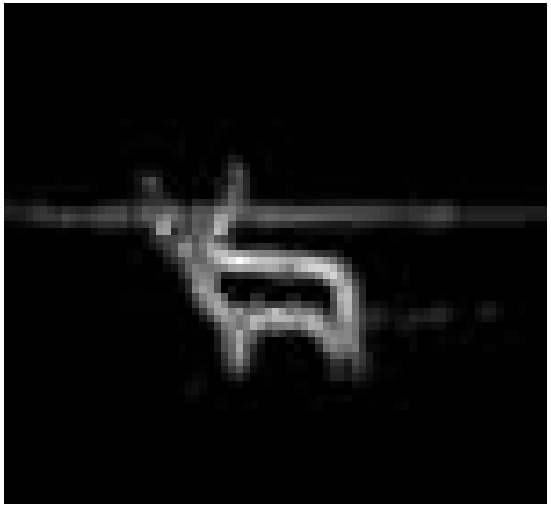
[Itti et al. 99,01,04], [Ma et al. 03]

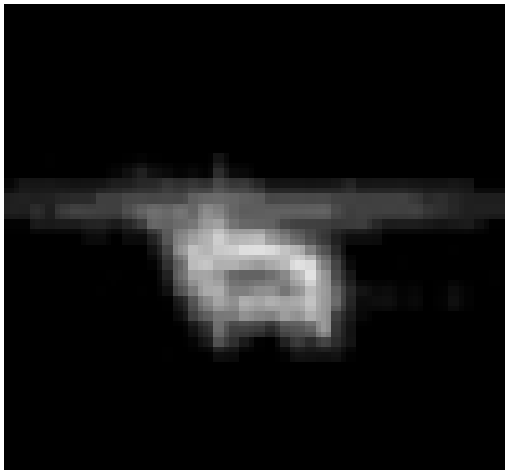
Scale-dependent

Original image



Saliency map at level 0 and level 1

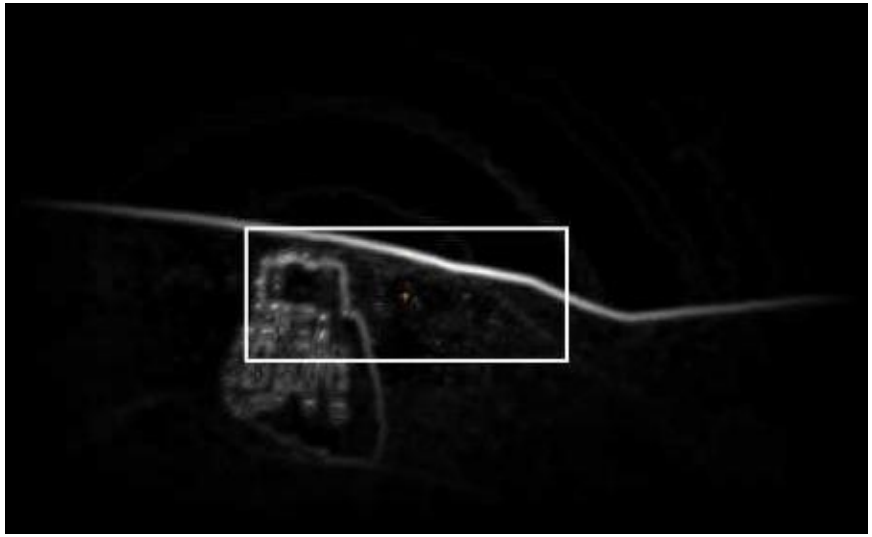




Saliency map at level 2 and level 3

Poor region/object localization

Original image



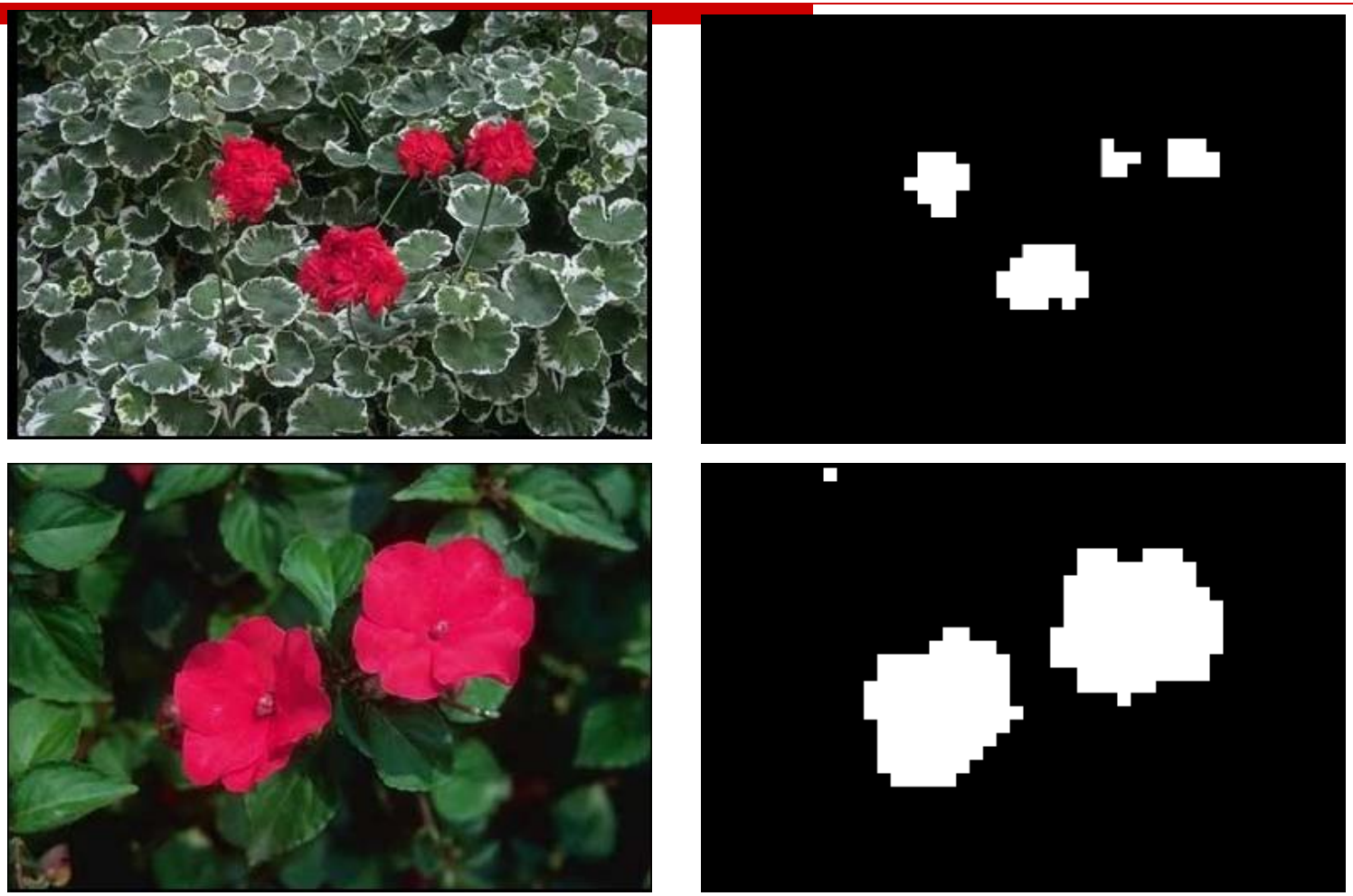
Saliency map

Region-based methods

[Hu et al. 2005]

- Segment image into regions
- Calculate the region feature contrast as saliency

Examples



Original images Saliency maps

Reprint from [Hu et al. 2005]

Limitation

Original image

Coarse image segmentation

Fine image segmentation

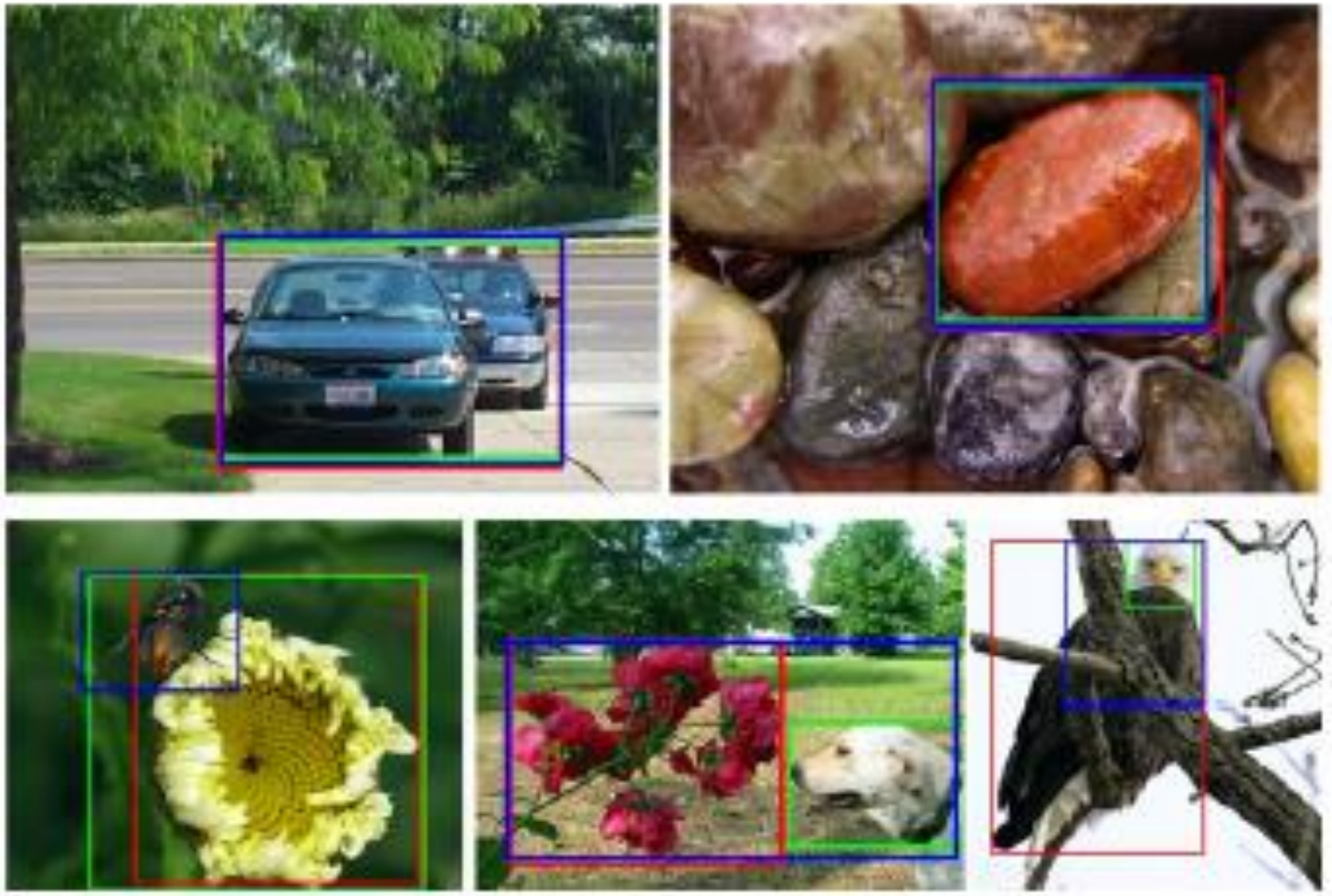
Saliency map

Learning to Detect Salient Object

- Extract image features
- Use Conditional Random Field to model
  - Relationship between features and saliency
  - Interaction between saliency of pixels
- Train a CRF model using labeled data
- Predict saliency using CRF model

Liu, Tie, Jian Sun, Nan-Ning Zheng, Xiaoou Tang, and Heung-Yeung Shum. "Learning to Detect A Salient Object." In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1-8. IEEE, 2007.

Training data



Reprint from [Liu et al. 07]

Examples

Input images

Itti's results

Predicated results

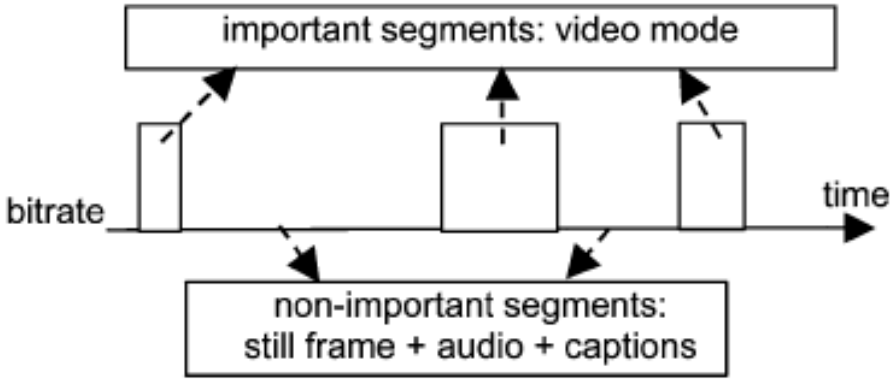
Reprint from [Liu et al. 07]

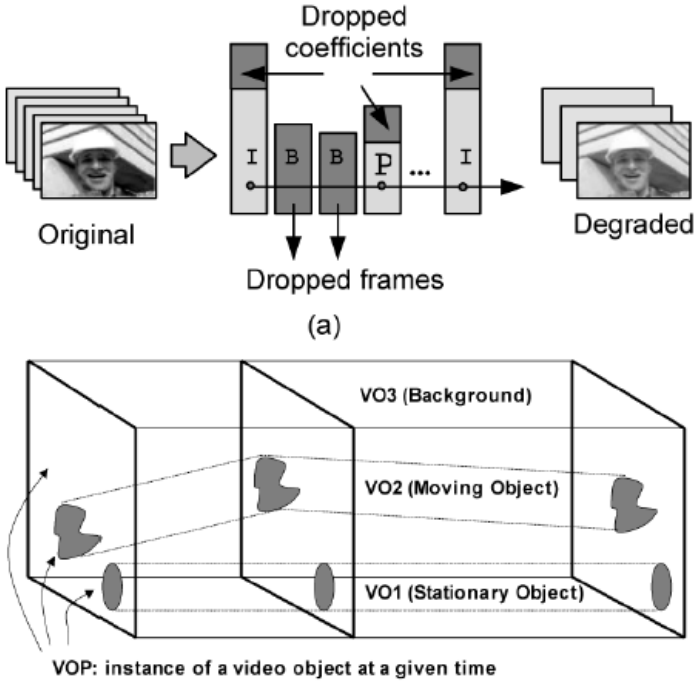
Conclusion

- Visual saliency measures low-level stimuli to human visual system
- Visual saliency can be computed either based on biological models or by data-driven methods
- Visual saliency serves as an approximation of importance, and can be helpful in a lot of applications

Applications

- Image and video editing
- Robot navigation
- Multimedia transmission
- Multimedia compression





[Chang et al, 05]

Next Time

- Face detection