Multimedia Technology

Lecture 7: Image Features

Lecturer: *Dr*. Wan-Lei Zhao

Autumn Semester 2024

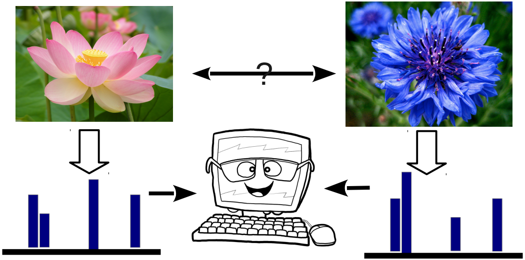
Email: wlzhao@xmu.edu.cn,

- We already know how the image is represented in computer
- We can process image as a matrix
- We can process image as a multi-variable function
- In this lecture, focus is turned on image features
- Why image feature?
- What are they
    - Image Local Features
    - Image Global Features
- How to extract them?

Outline

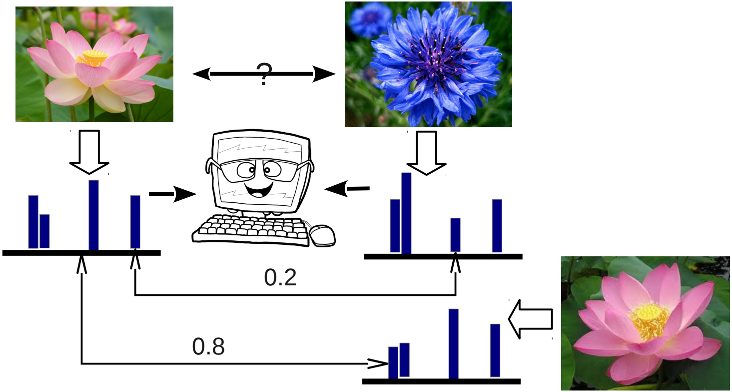
Why image features? (1)

- Images are not directly comparable by computer
- Features are the media to represent the images



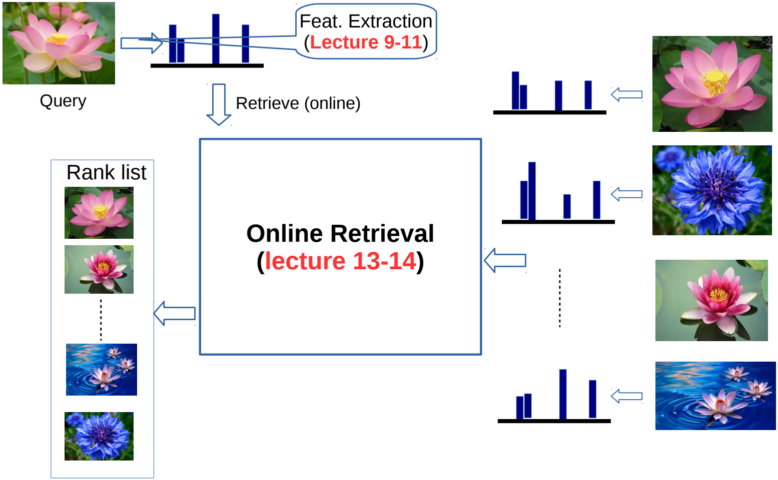
- Open issue: how the images are represented in our brain??

Why image features? (2)



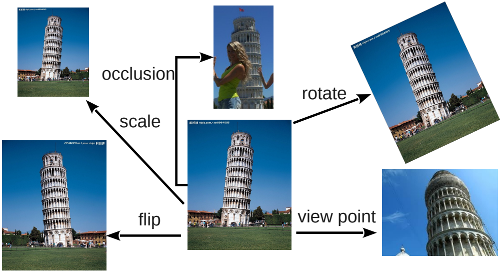
- Features should reflect the similarity as precise as possible

Framework of content based image retrieval (CBIR)



- CBIR = image features + retrieval/comparing method

Invariance properties of the features

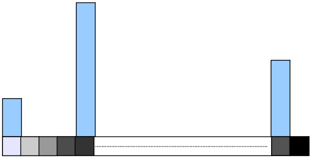


- Scale invariance is desired
- Rotation invariance is desired
- Flip invariance is desired
- Sometimes, color invariance is desired
- ...

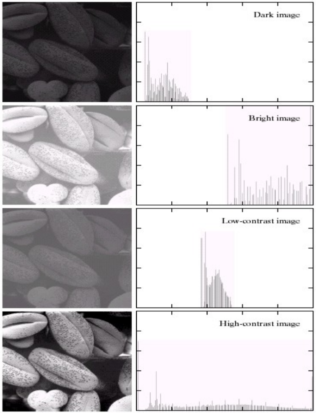
Image global features

- They are mostly statistical variables defined on image
- Easy to compute
- Popular global features:
  - Color Histogram
  - Color Moments
  - Histogram of Oriented Gradients
  - VLAD: globalized local feature (talk later)

Color Histogram (1)

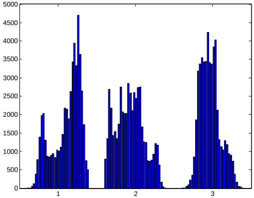


- Counting num. of pixels in each color range
- For color images:
  - Apply CH on each channel
  - Concatenate the vectors



Color Histogram (2)

(a) Original image



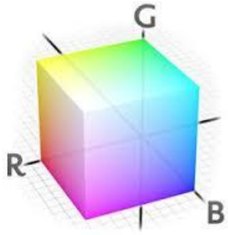
(b) Color histogram from RGB channels

Color Histogram (3)

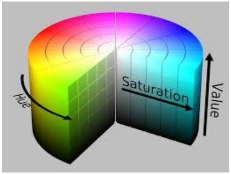
```
img1 = imread('lena.jpg');
r1   = im2double(img1(:,:,1))*256;
hr   = hist(reshape(r1,(225*225),1),32);
g1   = im2double(img1(:,:,2))*256;
hg   = hist(reshape(g1,(225*225),1),32);
b1   = im2double(img1(:,:,3))*256;
hb   = hist(reshape(b1,(225*225),1),32);
hist = [hr;hg;hb];
bar(hist);
```

Listing 1: Code to calculate histogram of 3 channels

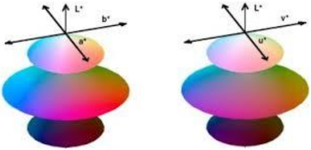
Color Spaces



(c) RGB color space



(d) HSV color space



(e) Lab color space

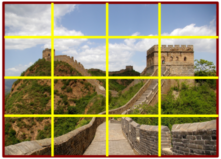
- Image can be represented with different color spaces
- There are formulars to do the transform in between
- Color histograms on HSV and Lab are more distinctive

Color Moment

- Intensity value distribution (on one channel) is viewed as a statistical variable
- We can calculate its moments (usually referred as central moment)
- Which is given below

$$\mu_n = E[X - E[x]]^n$$

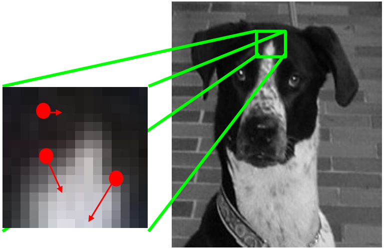
- E[x] is calculated by taking the average
- n is the order

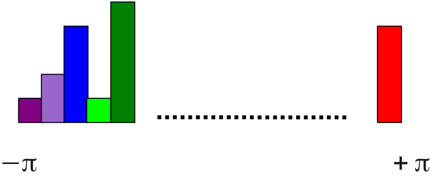


4x4 grid

- Moments are concatenated and moments from different blocks are concatenated
- It turns out to be simple but pretty successful

HOG: Histogram of Oriented Gradients





$$\bigtriangledown I(x,y) = \left(\frac{\partial I(x,y)}{\partial x}, \frac{\partial I(x,y)}{\partial y}\right) \quad (1)$$

- Orientation

$$\theta = tan^{-1}\frac{\frac{\partial I(x,y)}{\partial y}}{\frac{\partial I(x,y)}{\partial x}} \quad (3)$$

- Magnitude

$$m = \sqrt[2]{\frac{\partial I(x,y)}{\partial x}^2 + \frac{\partial I(x,y)^2}{\partial y}} \quad (2)$$

Partial Derivatives on Image

- Image is a multi-variable function f(x, y)
- When taking derivatives, dx=dy=1
- As a result

$$\frac{\partial I(x,y)}{\partial x} = I(x,y) - I(x-1,y)$$

- The second order derivative is given as

$$\begin{aligned}\frac{\partial^2 I(x,y)}{\partial x^2} &= [I(x+1,y) - I(x,y)] - [I(x,y) - I(x-1,y)] \\ &= [I(x+1,y) + I(x-1,y)] - 2 \cdot I(x,y)\end{aligned}$$

How well are the global features?

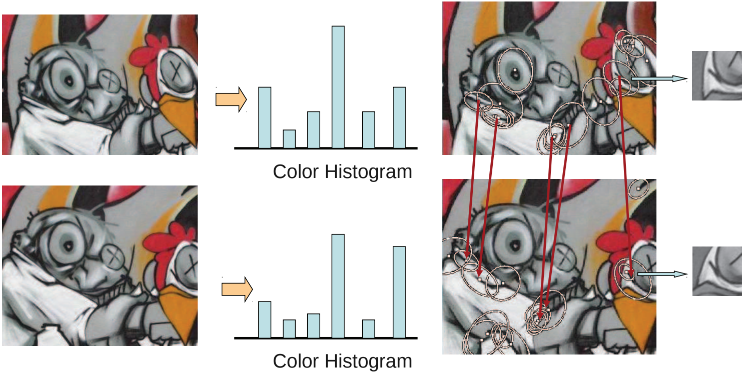
Scaling Rotation Affine Flip Occlusion

Transform	CH	HOG	Human vision
Scale	partially	partially	fully[1]
Rotation	fully	no	fully
Affine	partially	partially	fully
Flip	fully	no	fully
Occlusion	no	no	fully
Light/Color	vulnerable	vulnerable	fully
Blur	vulnerable	vulnerable	partially
View point	vulnerable	vulnerable	partially

- Homework:
  - Submit transformed images as query to Google image
  - See how well the system achieves invariances

[1] except for extreme cases

Global Feature *vs* Local Interest Point Feature



(a) Two paradigms for evaluating similarity between two images: global feature against local feature

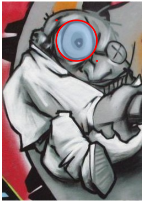
- Images are compared in a finer granularity

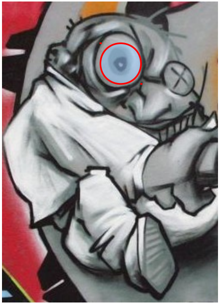
Scale Invariance: the concept (1)

- We already mentioned this concept before
- We study how we can achieve this for image local feature

- Local feature is detected, we cannot simply compare their pixel values
- They are simply not distinctive

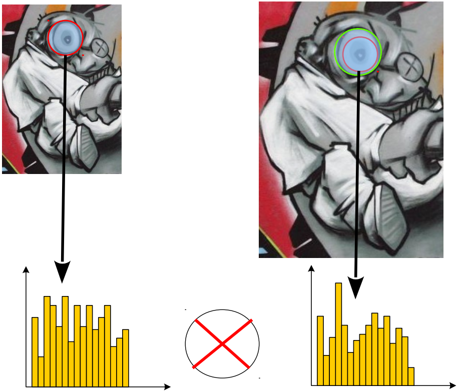
Scale Invariance: the concept (2)





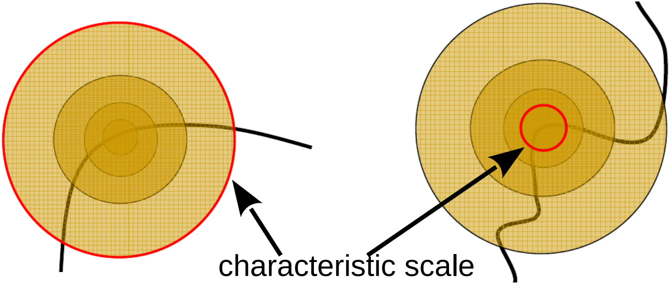
- We need to compare a local region
- But how to scope this local region?
- Fix the size for all points detected?

Scale Invariance: the concept (3)



- It is not working!!
- Fixed size does not cover the same region in two different images
- What we expect is the region scoped by green circle

Scale Invariance: the concept (4)

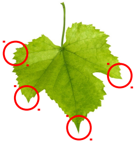


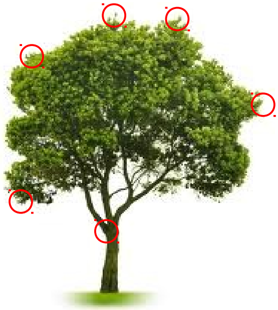
- We are looking for something like this
- The selected scale in two images cover the same local structure
- Notice that we detect corners from each image independently

Scale Invariance: scale space (1)

- In order to achieve scale invariance, let's look at scale space first
- What you can see you stand under tree?
- What you can see you stand 50 meters away from the tree?

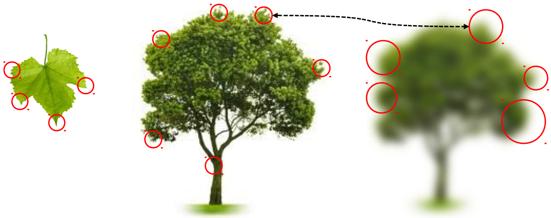
Scale Invariance: scale space (2)





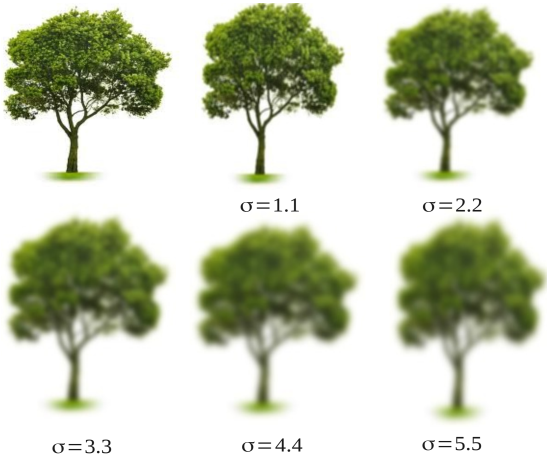
- Now you are asked to finger out corners from the leaf and from the tree
- Are you convinced?
- Can you still see the corners in one leaf when you stand 50 meters away?

Scale Invariance: scale space (3)



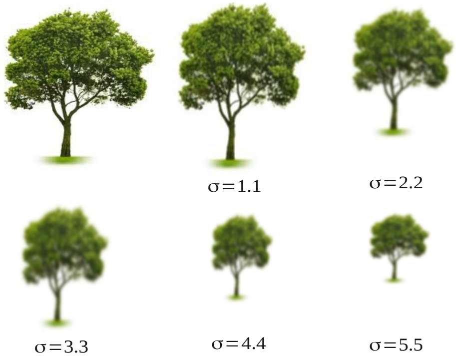
- Now let's move to 200 meters away from the tree
- What you can see??
- Conclusion: **certain corners only appear/survive in certain range of scales/watching distances**

Scale Invariance: scale space (4)



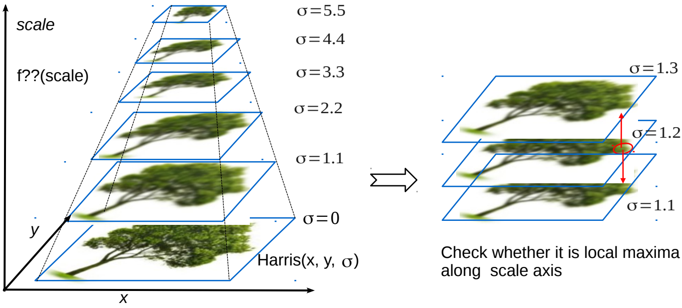
- Once the photo is taken, the distance between the tree and our watch position is fixed
- We simulate this by convolution with different σs

Scale Invariance: scale space (5)



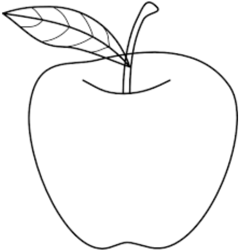
- When the image is blurred heavily, it is no need to keep its original size
- So finally we have above blurred image series
- As mentioned before, in different distances, you see different corners
- We want to find these stable points, which are visible within certain distance range

Scale Invariance: scale space (6)



- If we stack up this series of blurred images
- We have this pyramid
- Stable points are those which attain local maximum in scale space
- Local maximum means it is ‘**salient**’ within certain distance range

Why edges (2)

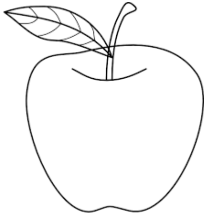


Why edges (3)

(a) (b) (c)

- Edges use less number of pixels than color blocks (Fig. (b)-(c)), however turn out to be more informative

Why edges (4)



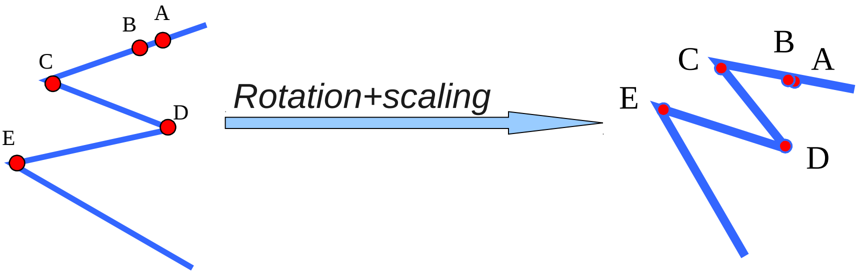
(a)

(b)

(c)

- Edges already carry most of the information to outline an object

Why corners instead of edges (1)

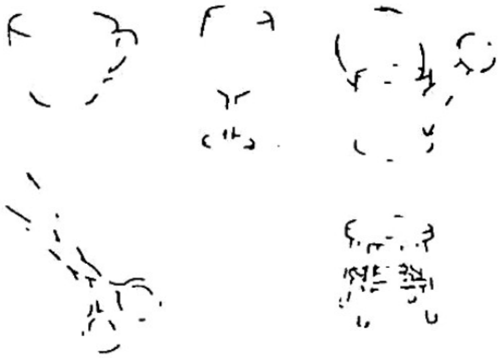


- Points (A, B) along the edges merged to each other
- The same for points in the flat region
- Corners (C, D, E) survive through scaling and rotation
- Different regions have different degree of robustness
- Corners are preferred

Why corners instead of edges (2)

- Guess what are the objects with edges only (corners are largely removed)

Why corners instead of edges (3)

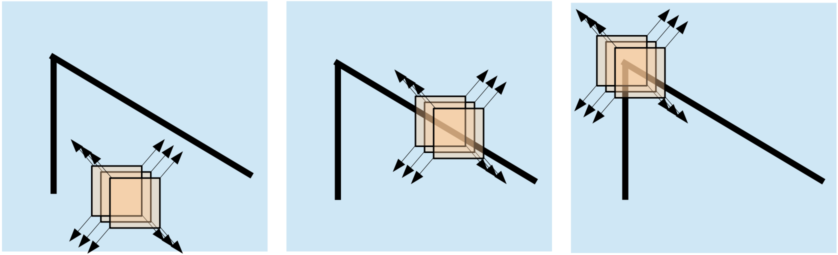


- Guess what are the objects with corners only

Why corners instead of edges (4)

- Check your answer

How to detect the corners



(a) (b) (c)

- Take an observing window, and move around
- No change in (a)
- Change happens along one direction in (b)
- Strong responses in two directions in (c)

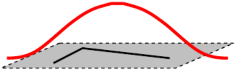
Formularize the probing procedure (1)

$$E(u,v)=\sum_{x,y}\underbrace{w(x,y)}_{\text{Window function}}\cdot[\underbrace{I(x+u,y+v)}_{\text{Shifted intensity}}-\underbrace{I(x,y)}_{\text{Intensity}}]^2$$

Window function $w(x,y)$ =

1 in window, 0 outside

or



Gaussian

- Measuring the energy changes
- w is the probing window
- u and v are the shifts in x and y directions

Formularize the probing procedure (2)

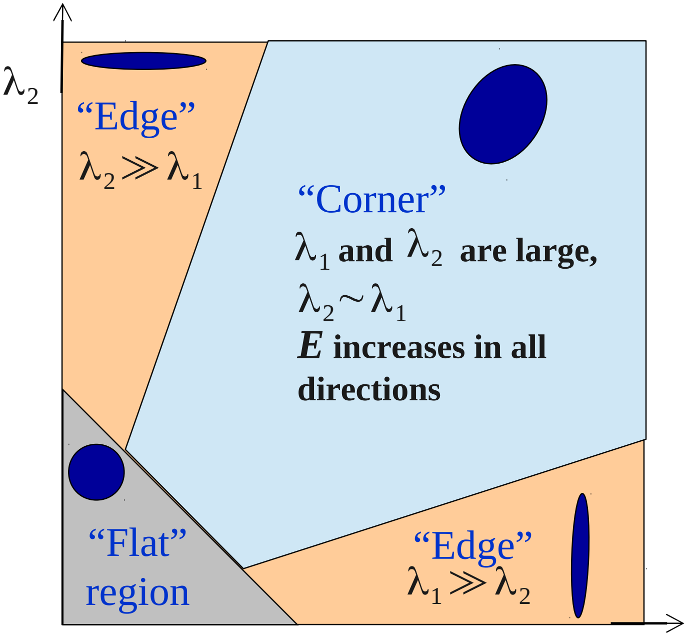
$$\boxed{I(x+u, y+v) \approx I(x,y) + u \cdot I_x(x,y) + v \cdot I_y(x,y)}$$

$$\begin{aligned}
E(u,v) &= \sum_{x,y} w(x,y)[I(x+u,y+v) - I(x,y)]^2 \\
&= \sum_{x,y} w(x,y)[I(x,y) + uI_x(x,y) + vI_y(x,y) - I(x,y)]^2 \\
&= \sum_{x,y} w(x,y)[uI_x(x,y) + vI_y(x,y)]^2 \\
&= [u, v] \left(\sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \right) \begin{bmatrix} u \\ v \end{bmatrix}
\end{aligned} \tag{4}$$

Formularize the probing procedure (3)

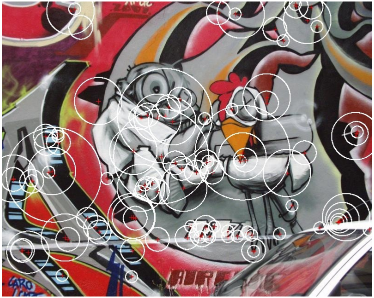
$$
\begin{aligned}
E(u,v) &= \sum_{x,y} w(x,y)[I(x+u,y+v) - I(x,y)]^2 \\
&= \sum_{x,y} w(x,y)[I(x,y) + uI_x(x,y) + vI_y(x,y) - I(x,y)]^2 \\
&= \sum_{x,y} w(x,y)[uI_x(x,y) + vI_y(x,y)]^2 \\
&= [u,v]\left(\sum_{x,y} w(x,y)\begin{bmatrix} I_x^2 & I_xI_y \\ I_xI_y & I_y^2 \end{bmatrix}\right)\begin{bmatrix} u \\ v \end{bmatrix}
\end{aligned}
\tag{5}
$$

$$
E(u,v) = [u,v] \cdot M \cdot \begin{bmatrix} u \\ v \end{bmatrix} \tag{6}
$$


λ_2
"Edge"
$\lambda_2 \gg \lambda_1$
"Corner"
λ_1 and λ_2 are large,
$\lambda_2 \sim \lambda_1$
E increases in all directions
"Flat" region
"Edge"
$\lambda_1 \gg \lambda_2$

Harris Detector: the results

- Steps:
  1. Compute Harris(x, y) for each pixel
  2. Select the points that attain local maximum (**Non-maximum suppression**)
  3. Consider points whose $Harris(x, y) > t_0$



- How to make use of these detected points?
- How the circle comes?

Hessian Detector

- In practice, Hessian function is also a good option

$$H(x,y,\sigma) = \begin{bmatrix} \frac{\partial^2 I(x,y,\sigma)}{\partial x^2} & \frac{\partial^2 I(x,y,\sigma)}{\partial x \partial y} \\ \frac{\partial^2 I(x,y,\sigma)}{\partial y \partial x} & \frac{\partial^2 I(x,y,\sigma)}{\partial y^2} \end{bmatrix}$$

$$Hessian(x,y,\sigma) = Det(H(x,y,\sigma)) * \sigma^2$$

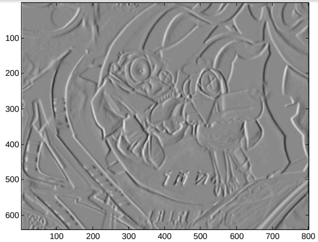
Hessian Detector: steps

1. Convert input color image into gray level image (optional)
2. Design Dxx, Dxy, Dxy Gaussian templates
3. Apply Dxx, Dxy, Dxy template on the gray level image
4. Calculate Hessian function based on following Equation

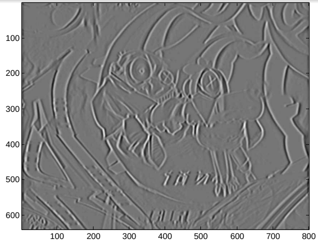
$$H(x, y, \sigma) = \sigma^4 * (Dxx * Dyy - Dxy^2)$$

5. Apply non-maximum suppression on Hessian function image
6. Display points higher than a threshold

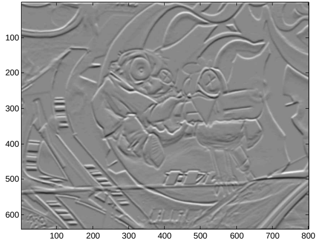
Hessian Detector: the derivatives



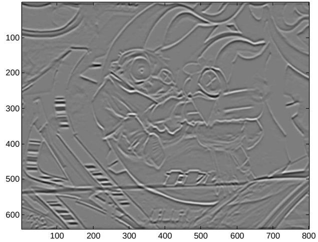
(d) imgDx



(e) imgDxx

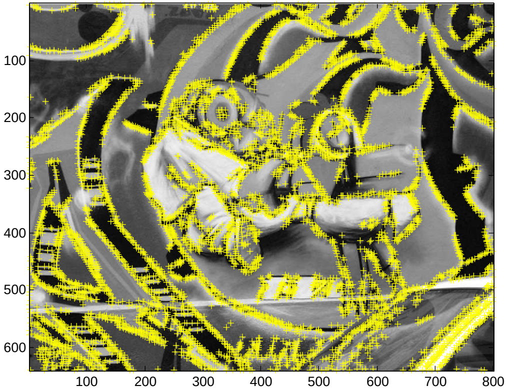


(f) imgDy



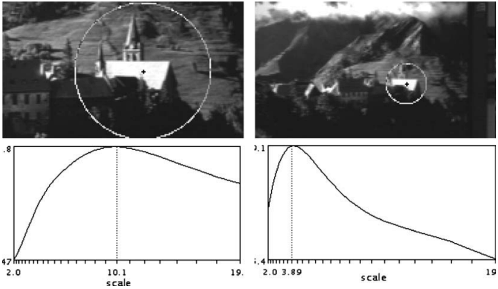
(g) imgDyy

Hessian Detector: the derivatives



Scale Invariance by Hessian Detector

- Calculate Hessian with a series of $\sigma_1, \sigma_2, \cdots$

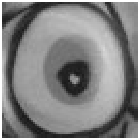


- Take the σ where $H(x, y, \sigma)$ achieves local maxima as the characteristic scale

$$H(x, y, \sigma) = \sigma^4 * (Dxx * Dyy - Dxy^2)$$

SIFT descriptor: the motivation

- Up to now, we are able to extract local point as a local patch
- To check whether two local patches are similar or not
- We need a feature represenation for this local patch
- There are dozens of options: CH, CM or Gabor (wavelet feature)
- SIFT turns out to be the most successful one

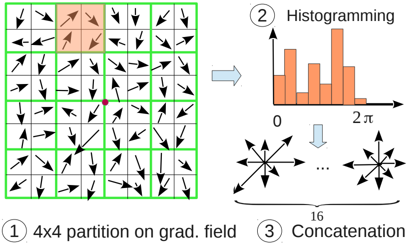
  

Overview about SIFT descriptor

- It is known the best local interest point (keypoint) descriptor
- Num. Of citations: 31,039 (up to 14th, Aug., 2015)
- It is distinctive, but still tolerant to small errors (displacement, lighting changes, deformation) in the local patch
- When people talk about image local features, people talk on the basis of SIFT

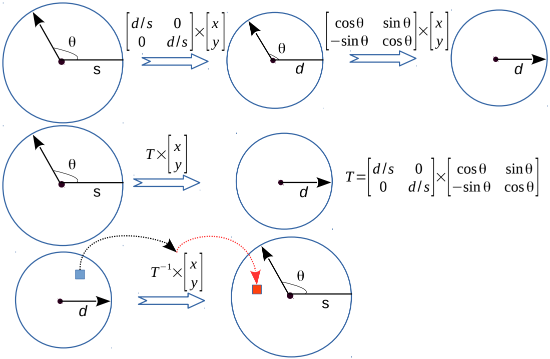
- Refer to: David G. Lowe, Distinctive Image Features from Scale-Invariant Keypoints, IJCV'04, pp. 91-110.

SIFT descriptor: the procedure



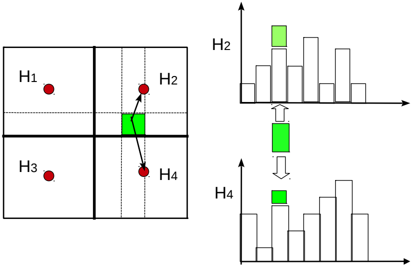
- Pre-processing steps:
- Normalize the patch to $2d \times 2d$
- Calculate gradient field
- SIFT generation:
  1. Partition gradient into 4×4 grid
  2. Compute gradient histogram on each block
  3. Concatenate 16 histograms as the final feature

Three tricks in SIFT Implementation: (1)



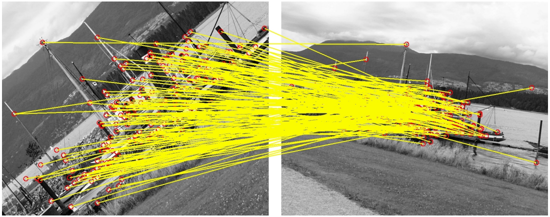
1. Extracting hole-less Normalized Patch
2. 2D interpolation
3. Removal of abnormal peaks

Three tricks in SIFT Implementation (2)

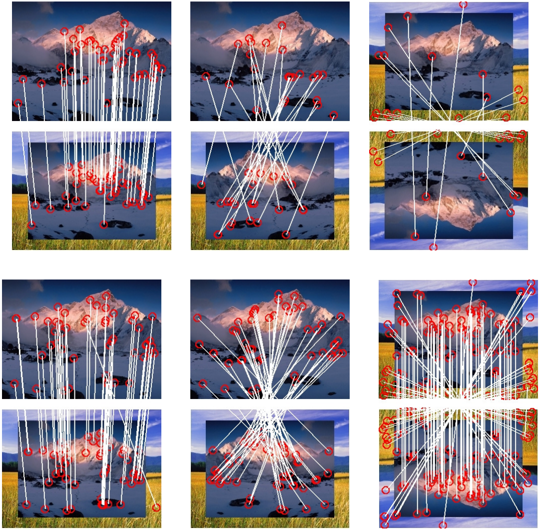


1. Extracting hole-less Normalized Patch
2. 2D interpolation
3. Removal of abnormal peaks



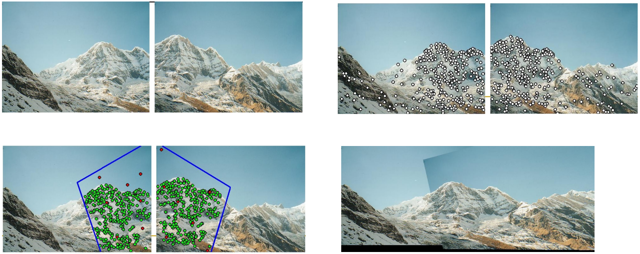



BASMATI
Product of Italy
Crisp Almond Cookies
NET WT 8.8 OZ
Peso Netto 250 g e



(a) Scale (b) Scale+flip (c) Rotate+flip

Image Stiching with SIFT

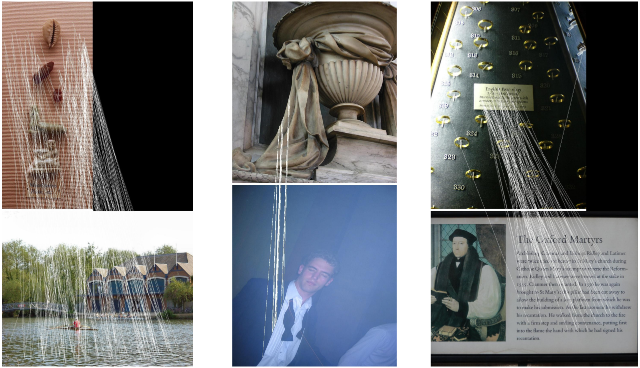


1. Matching image pairs first
2. Stich images with overlaps
3. Blending the stiched images

Summary over Geometric Invariances

- Based on different schemes, we can achieve
  1. **Scale invariance** by detecting points in scale-space
  2. **Rotation invariance** by estimating the dominant orientation
  3. **Affine invariance** by adapting to local structure
  4. **Flip invariance** by estimating the dominant angular moment

Dark side of Local feature (1)



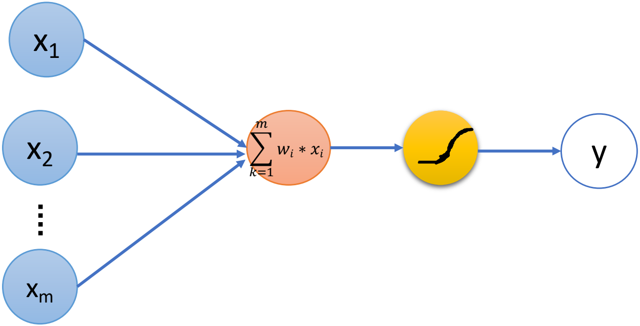
1. The computational cost is high
2. Vulnerable to big view-point changes
3. Vulnerable to deformation

Dark side of Local feature (2)

1. The computational cost is high
2. Vulnerable to big view-point changes
3. Vulnerable to deformation

Outline

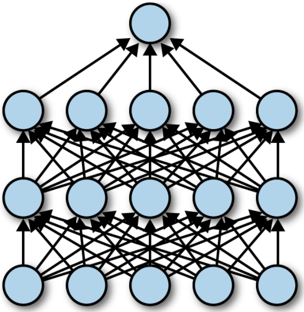
Perceptron: a Review



$$N(x) = \eta(\sum_{i=1}^{m} w_i \cdot x_i + b) \quad (7)$$

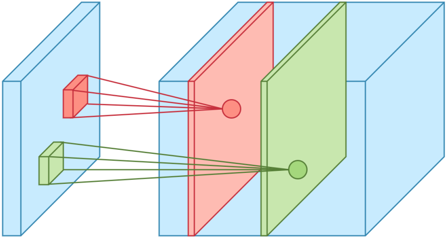
- The basic element for a neural Network is neuron
- It consists of a linear mapping with a non-linear activation

Dense Network: a Review



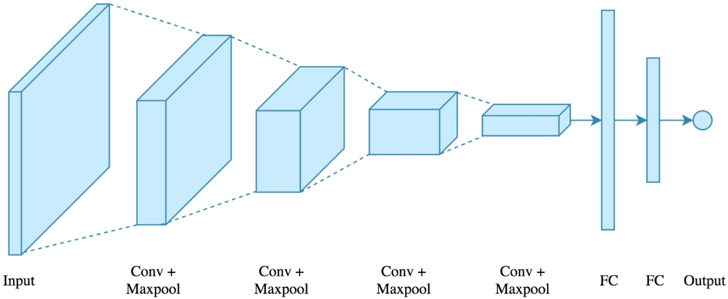
- Dense Network

Convolution Neural Network (1)



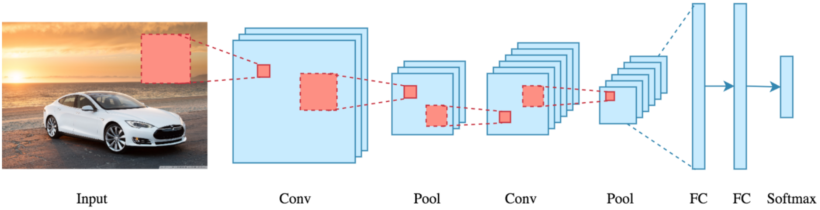
- One filter convolves over the whole image
- The resulting image is called a feature map
- One filter is trained to pick cerntain pattern from the image

Convolution Neural Network (2)



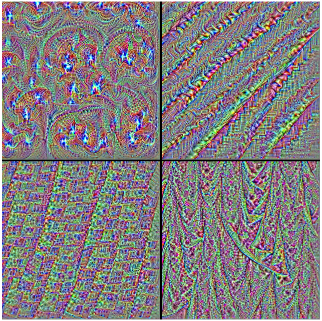
- A complete CNN is a stack-up of several CNN layers

Convolution Neural Network (3)



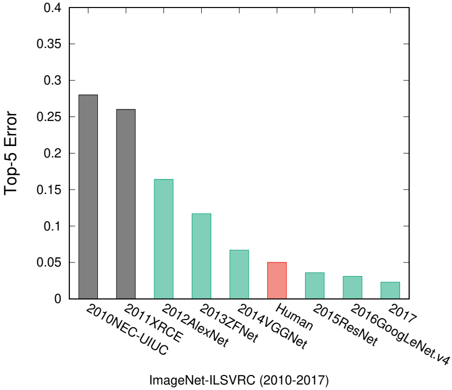
- CNN is first found to be powerful for image classification task
- The filter maintains the relative location pixels

Convolution Neural Network (4)



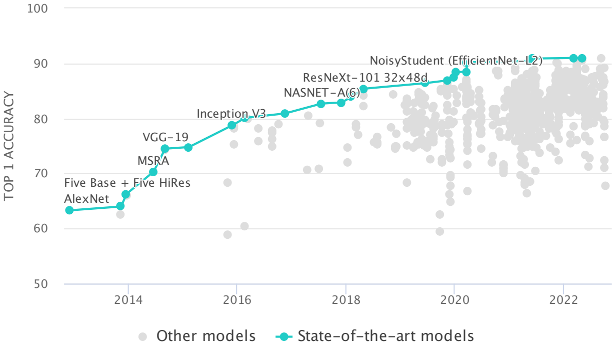
- A visualization of VGG-Block5 filters
- One filter picks a specific pattern from the feature maps of the previous layer

CNN for Classification (1)



- Year 2012 is the starting year of deep learning
- Supervised image classification is no longer a problem

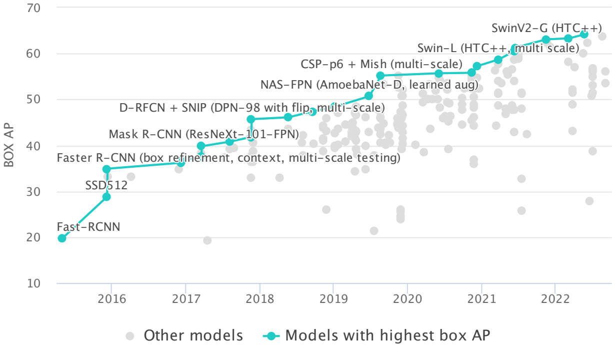
CNN for Classification (2)



- Supervised image classification is no longer a problem[2]
- It becomes a platform to verify the power of a deep model

[2] https://paperswithcode.com/sota/

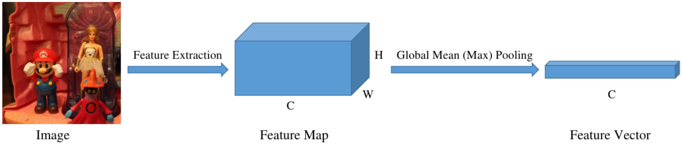
CNN for Object Detection



- The performance of Supervised Object Detection is also saturated[3]
- Online/semi-supervised/unsupervised Object detection are open issues

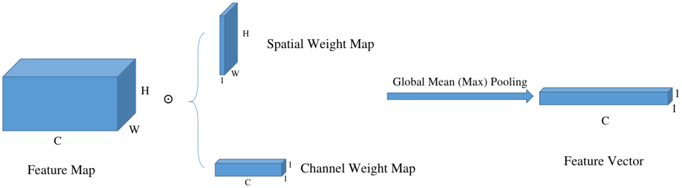
[3] https://paperswithcode.com/sota/

Global Max Pooling



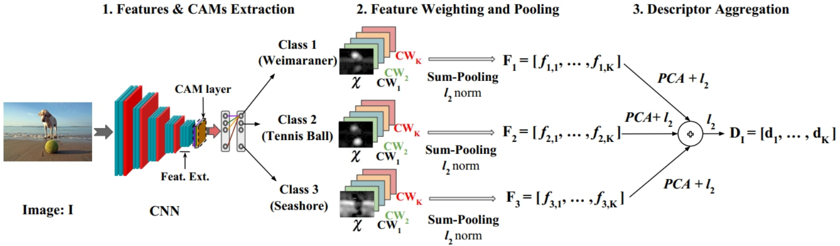
- One feature vector is produced for one image
- Max pooling is applied on one feature map
- Pooling results are concantenated into a fixed length vector

Global Weighted Max Pooling

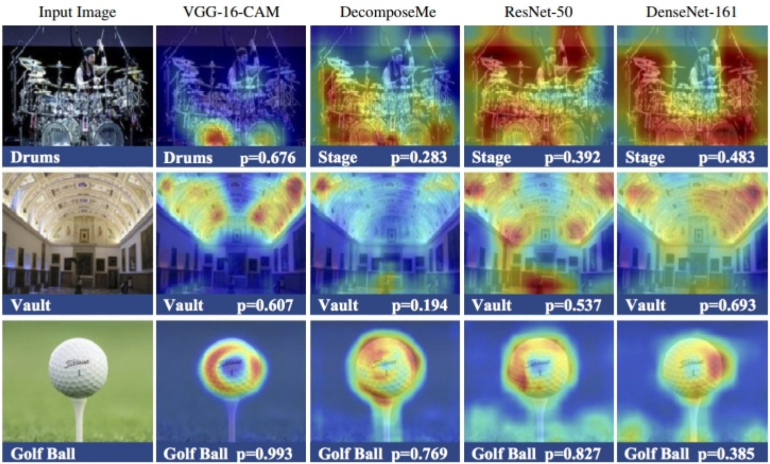


- As a variant, you can assign a weight according to the importance

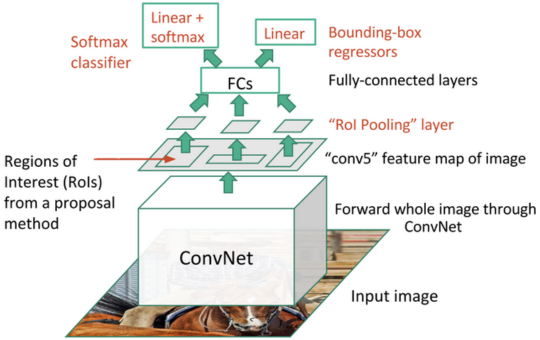
Global Max Pooling Example (1)



Global Max Pooling Example (2)



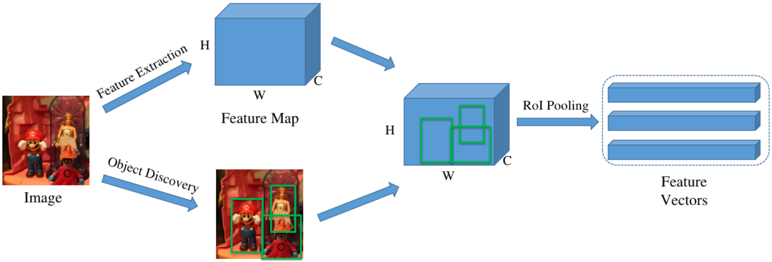
ROI Pooling (1)



- It was first proposed in Faster R-CNN paper[4]
- Different sizes of proposals are converted into the feature maps in the same size

[4] S. Ren, K. He, R. Girshick, J. Sun: Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks.

ROI Pooling (2)



- Given we want to build feature from a local region
- Maxpooling/average pooling is applied on a local

ROI pooling in animation

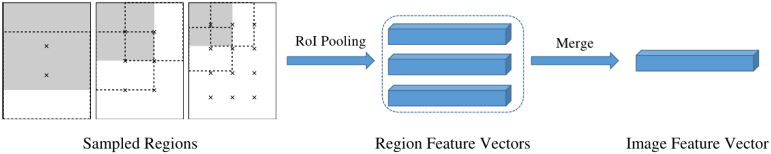
input

0.88	0.44	0.14	0.16	0.37	0.77	0.96	0.27
0.19	0.45	0.57	0.16	0.63	0.29	0.71	0.70
0.66	0.26	0.82	0.64	0.54	0.73	0.59	0.26
0.85	0.34	0.76	0.84	0.29	0.75	0.62	0.25
0.32	0.74	0.21	0.39	0.34	0.03	0.33	0.48
0.20	0.14	0.16	0.13	0.73	0.65	0.96	0.32
0.19	0.69	0.09	0.86	0.88	0.07	0.01	0.48
0.83	0.24	0.97	0.04	0.24	0.35	0.50	0.91

- Divide the proposals into fixed number of blocks[5]
- Take the maximum/average on each block

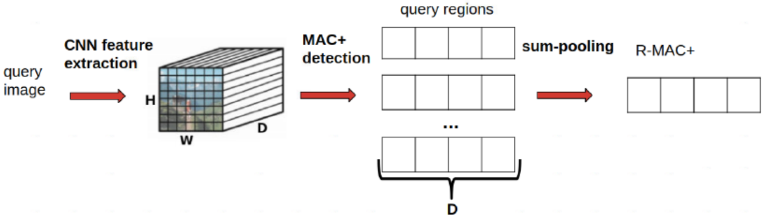
[5] View animation with Acrobat Reader

ROI Pooling Example: R-MAC (1)



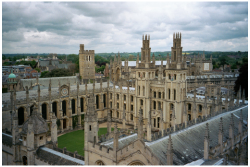
- R-MAC extracts local features from feature maps

ROI Pooling Example: R-MAC (2)



- Feature extraction for the query

Performane Comparison on Oxford5k: the dataset

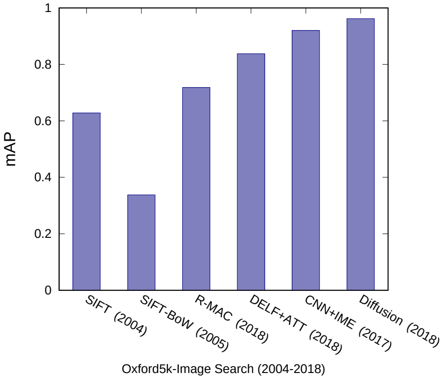





- There are *5063* images captured from Oxford University[6]
- *55* images are selected as the query

[6] https://www.robots.ox.ac.uk/ vgg/data/oxbuildings/

Performane Comparison on Oxford5k: the dataset



- The performance of SIFT (point-to-point matching) is already satisfactory, BUT slow
- Deep features are very successful for image search task

References

1. Distinctive Image Features from Scale-Invariant Keypoints, D. G. Lowe, *IJCV'10*
2. SURF: Speeded Up Robust Features, H. Bay and et al., *ECCV'06*
3. Flip invariant SIFT for Video Copy and Object Detetion, Wan-Lei Zhao and et al. *TIP'13*
4. Feature Detection with Automatic Scale Selection, Tony Lindeberg, *IJCV'98*, pp.79-116
5. Scale and affine invariant interest point detectors, Krystian Mikolajczyk and et al. *IJCV'02*, pp.63-86
6. Local Invariant Feature Detectors: A Survey, T. Tuytelaars and K. Mikolajczyk, NoW Publisher Inc. 2008
7. Class-weighted convolutional features for visual instance search, Jimenez A, Alvarez J M, Giro-i-Nieto X., 2017
8. Fast R-CNN, Girshick R., CVPR, 2015: 1440-1448
9. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, S. Ren, K. He, R. Girshick, J. Sun, ICCV, 2015
10. Particular object retrieval with integral max-pooling of CNN activations, Tolias G, Sicre R, Jégou H., 2015

Q & A

Thanks for your attention!