7-9 Monday – 309-GD2

Xử lý ảnh
INT3404 1

Giảng viên: TS. Nguyễn Thị Ngọc Diệp

Email: ngocdiep@vnu.edu.vn

Slide & code: https://github.com/chupibk/INT3404_1

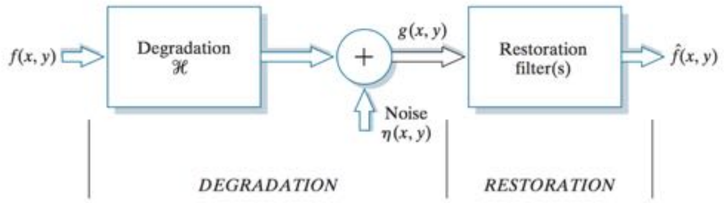
Final project - Registration form

- https://forms.gle/Qpp1hbX9QPG875gT8
- Speadsheet to edit:
  - https://docs.google.com/spreadsheets/d/1c13rgmfNdIpJEPD544qDY2sHxH-ZOxzobeW8fI1CI_w/edit#gid=82493842
  - → Edit with care!
- Final project schedule: week 13, 14, 15

Final project schedule

Group name	Problem	Schedule
HCA_image	trích xuất data trong bảng từ ảnh scan (excel) để trả về file excel	Week 14
Naive solver	Sudoku solver, mục tiêu: đọc được số từ ảnh chụp điện thoại	Week 14
gnissecorpegami	Đề tài chọn: Sudoku solver Mục tiêu: Viết một chương trình nhận ảnh chụp một câu đố Sudoku 9x9 và trả lại bức ảnh sau khi đã điền hết các ô trống còn lại	Week 14
SODOKU	Giải trò chơi Sodoku bằng phương pháp xử lý ảnh và học máy. Đầu vào sẽ chứ những hình ảnh chứa những câu đố Sodoku, đầu ra là ảnh Sodoku đã được giải.	Week 15
Resnet	Sudoku Solver - Đây là bài toán giải trò chơi Sudoku, với input là 1 image câu đố sudoku, sau khi cho qua mô hình của chúng em, output đầu ra sẽ là 1 image câu đố đã được giải. ^^	Week 15
Tứ kỵ sĩ	Sudoku solver	Week 13
UET_IP	Xây dựng 1 project phân đoạn đường cao tốc, nhận dạng làn đường, nhận dạng biển báo. Đầu vào là 1 video đường cao tốc, đầu ra là video đã được highlight đường, làn đường, biển báo.	Week 13
TicTacToe	Tic-tac-toe, mục tiêu: có thể đưa ra ảnh output xác định trạng thái trò chơi hiện tại (đang chơi, X thắng, O thắng)	Week 14
Gonzalez's Descendants	Mô hình cho phép nhận diện các khuôn mặt có trong bức ảnh cho trước	Week 13
allenGroup	recover 3D scene from single image!	Week 15
Naive Solver	Giải sudoku từ ảnh chụp được	Week 14
N-A-M-E-K	Đầu vào là ảnh game X-O, đầu ra là gợi ý cho nước đi tiếp theo.	Week 15
Team TTTN	Selected problem: Chess Board Reader. Goal: Correctly detect and identify a chessboard and the configuration of its pieces.	Week 14, Week 15
Untitled	Giải rubic 3x3x3	Week 13
15gg	sudoku solver, giải bài toán sudoku qua hình ảnh	Week 15
Image Killer	Đề tài: Tự động chấm điểm bài thi trắc nghiệm. Mục tiêu: dùng các kỹ thuật xử lý ảnh đã học để có thể tìm được số câu trả lời đúng trong mẫu phiếu trả lời trắc nghiệm được dùng trong kỳ thi THPTQG.	Week 13
17020646	lottery ticket recognition	Week 15

Week 10 recall: image restoration



Spatial domain

$$g(x,y) = (h \star f)(x,y) + \eta(x,y)$$

Frequency domain

$$G(u,v) = H(u,v)F(u,v) + N(u,v)$$

h(x, y) : degradation function
n(x, y) : additive noise term

Restoration seeks to find filters that apply the process in reverse (deconvolution filters)

Final exam exemption – special opportunity

- Write a report that clarifies three following questions:

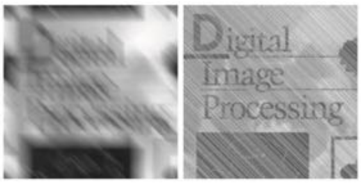
1. Explain the intuition and the derivation of the degradation function modeling

$$H(u,v) = e^{-k(u^2 + v^2)^{5/6}}$$

2. Explain the derivation of the Wiener filter and why it is equivalent to minimize mean square error

$$\hat{F}(u,v) = \left[\frac{H^*(u,v)}{|H(u,v)|^2 + S_\eta(u,v)/S_f(u,v)} \right] G(u,v)$$

3. Explain why the inverse filter result (right image) caused the black regions in the original image (left) less black.



Exemption rules

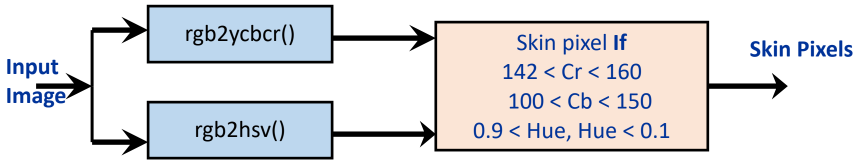
- Answer all three questions correctly
- Only one student who made the best report
    - Fastest and most correct
- Submit via email:
    - Mail to: ngocdiep@vnu.edu.vn
    - Title: [final exam exemption] Student ID
    - Deadline: Nov 10, 2019 23:59 (Hanoi time)

Homework 3: Face segmentation

Submitted:
https://drive.google.com/drive/folders/0B8pLkD4XxnvgfnlLU3pBc3R1bUxyenYzbG04eFhDU21wZzZabDBnNG54MUExZUdtM3llbnM

- Skin segmentation using (Cr, Cb, Hue) space

RGB to YCbCr
RGB to HSV
Threshold to determine skin regions



Input Image

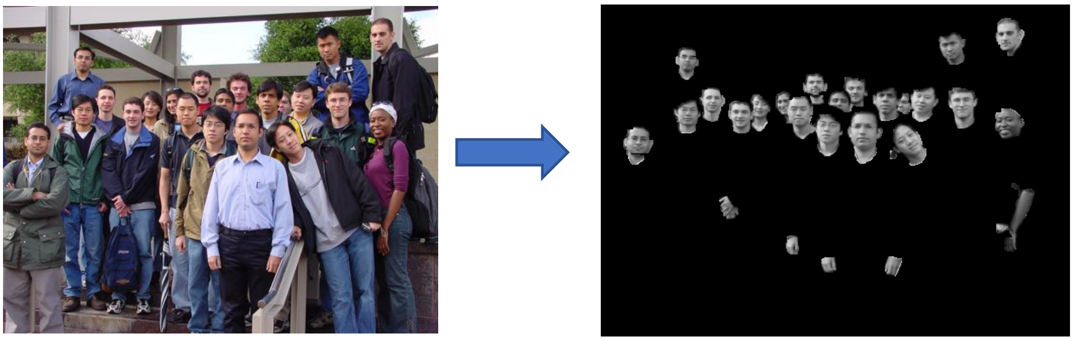
rgb2ycbcr()

rgb2hsv()

Skin pixel **If**
142 < Cr < 160
100 < Cb < 150
0.9 < Hue, Hue < 0.1

Skin Pixels

- Cleanup using morphological operators



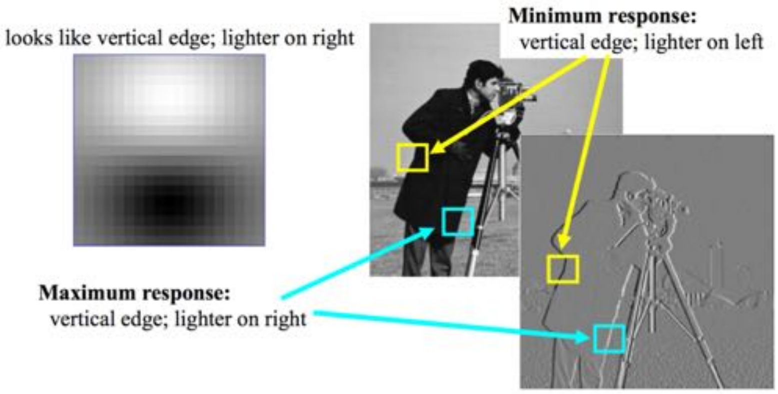
Source: EE 368, Deepesh Jain et. al.

Lịch trình

Tuần	Nội dung	Yêu cầu đối với sinh viên
1	Giới thiệu môn học Làm quen với OpenCV + Python	Cài đặt môi trường: Python 3, OpenCV 3, Numpy, Jupyter Notebook
2	Phép toán điểm (Point operations) – Điều chỉnh độ tương phản – Ghép ảnh	Làm bài tập 1: điều chỉnh gamma tìm contrast hợp lý
3	Histogram - Histogram equalization - Phân loại ảnh dùng so sánh histogram	Thực hành ở nhà
4	Phép lọc trong không gian điểm ảnh (linear processing filtering) - làm mịn, làm sắc ảnh	Thực hành ở nhà Tìm hiểu thêm các phép lọc
5	Tìm cạnh (edge detection)	Thực hành ở nhà
6	Các phép toán hình thái (Erosion, Dilation, Opening, Closing) - tìm biển số	Làm bài tập 2: tìm barcode
7	Chuyển đổi không gian - miền tần số (Fourier) - Hough transform	Thực hành ở nhà
8	Phân vùng (segmentation) - depth estimation - threshold-based - watershed/grabcut	Đăng ký thực hiện bài tập lớn
9	Mô hình màu Chuyển đổi giữa các mô hình màu	Làm bài tập 3: Chuyển đổi mô hình màu và thực hiện phân vùng
10	Mô hình nhiễu -Giảm nhiễu -Khôi phục ảnh -Giảm nhiễu chu kỳ - Ước lượng hàm Degration -Hàm lọc ngược, hàm lọc Wiener	Thực hành ở nhà
11	Template matching – Image Matching	Làm bài tập 4: puzzle
12	Nén ảnh	Thực hành ở nhà
13	Hướng dẫn thực hiện đồ án môn học	Trình bày đồ án môn học
14	Hướng dẫn thực hiện đồ án môn học	Trình bày đồ án môn học
15	Tổng kết cuối kỳ	Ôn tập

Cross correlation

- Cross correlation with a filter can be viewed as comparing a little "picture" of what you want to find against all local regions in the image

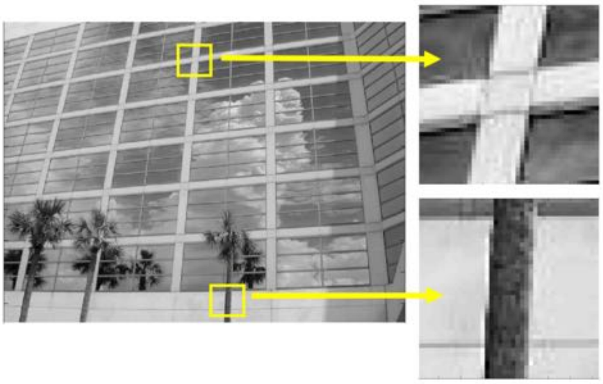

looks like vertical edge; lighter on right
Minimum response:
vertical edge; lighter on left
Maximum response:
vertical edge; lighter on right

Cross correlation

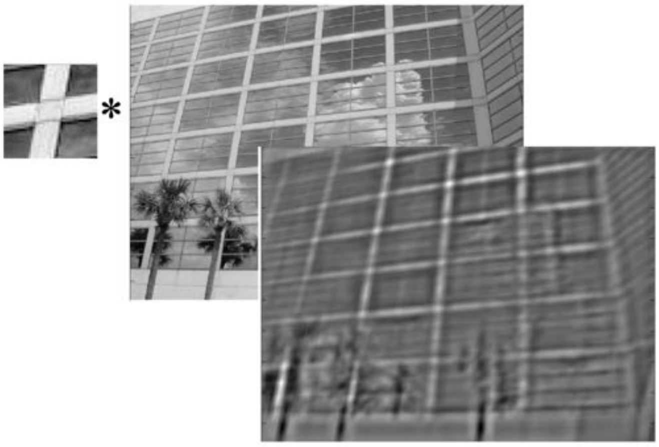
- Cross correlation with a filter can be viewed as comparing a little “picture” of what you want to find against all local regions in the image
- For this reason, it is sometimes called “matched filtering”

Template matching

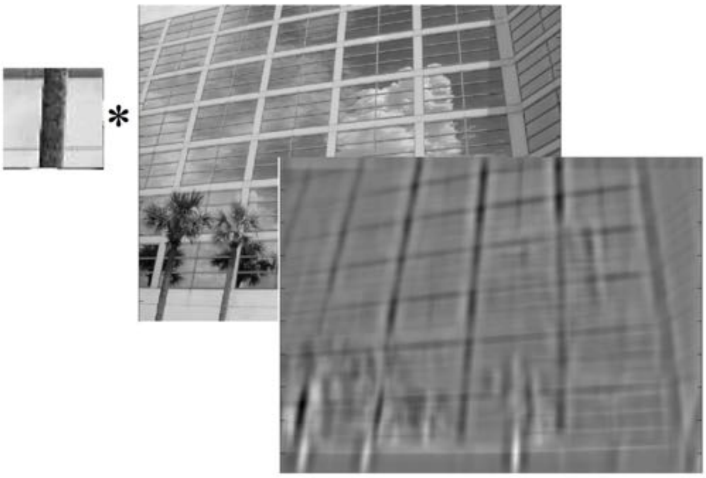
- What if we cut little pictures out from an image, then tried to convolve them with the same or other images?



Template matching



Template matching



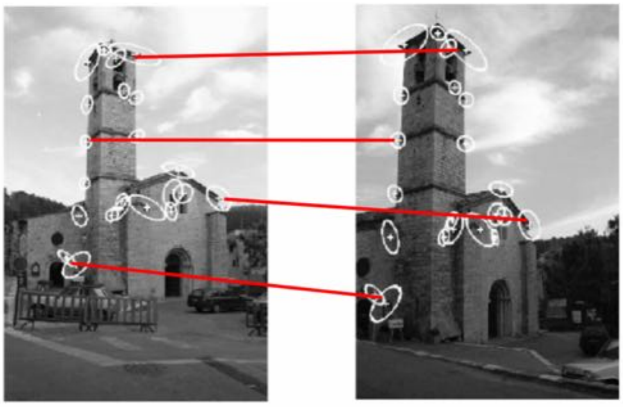
Actually…

I subtracted the mean gray value from both the image and the template before doing cross correlation.

Why?

Correspondence problem

- Vision tasks such as stereo and motion estimation require finding corresponding feature across two or more views



Correspondence problem

- Basic assumptions:
  - Most scene points are visible in both images
  - Corresponding image regions are similar

Correspondence problem

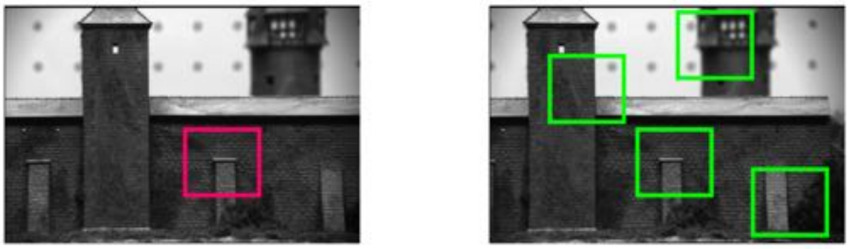
- Is a ”search” problem
  - Given an element in the left image, search for the corresponding element in the right image
  - We will typically need geometric constraints to reduce the size of the search space
- We must choose:
  - Elements to match
  - A similarity measure to compare elements

Correspondence problem

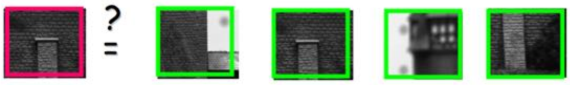
- Two classes of algorithms:
  - Correlation-based algorithms
    - Produce a DENSE set of correspondences
  - Feature-based algorithms
    - Produce a SPARSE set of correspondences

Correlation-based algorithms

Elements to be matched are image patches of fixed size

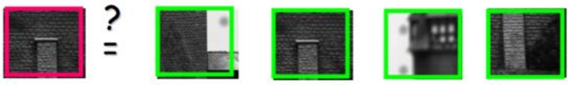


Task: what is the corresponding patch in the second image?



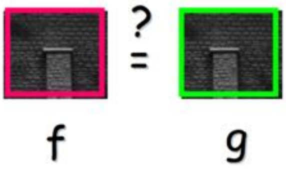
Correlation-based algorithms

Task: what is the corresponding patch in the second image?



1 – Need an appearance similarity function
2 – Need a search strategy to find location with highest similarity. Simplest (but least efficient) approach is exhaustive search

Comparing windows



Some possible measures:

$$\max_{[i,j]\in R} |f(i,j) - g(i,j)|$$

$$\sum_{[i,j]\in R} |f(i,j) - g(i,j)|$$

$$\left.\begin{array}{l} SSD = \displaystyle\sum_{[i,j]\in R} (f(i,j) - g(i,j))^2 \\ C_{fg} = \displaystyle\sum_{[i,j]\in R} f(i,j)g(i,j) \end{array}\right\} \text{Most popular}$$

Correlation $\mathbf{C_{fg}}$

$$C_{fg} = \sum_{[i,j]\in R} f(i,j)g(i,j)$$

If we are doing exhaustive search over all image patches in the second image, this becomes cross-correlation of a template with an image!

Example

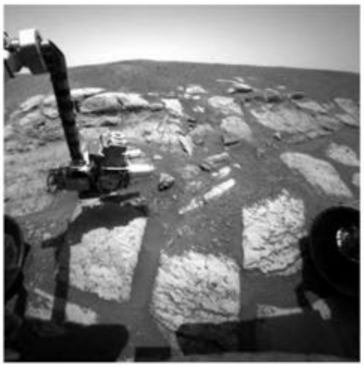


Image 1

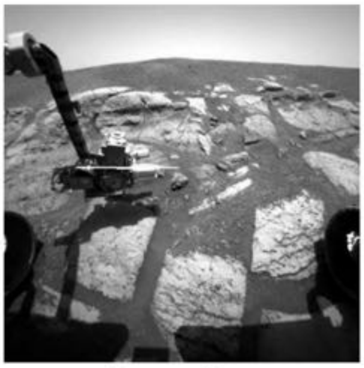
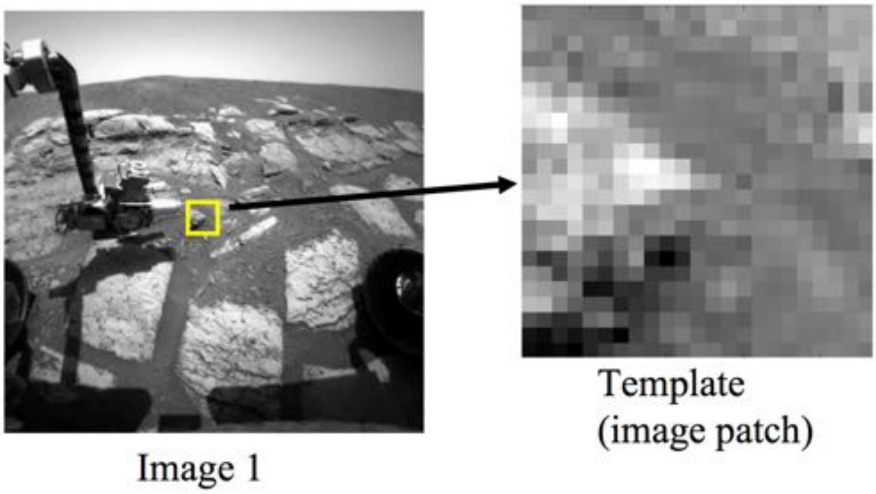


Image 2

Note: this is a stereo pair from the NASA mars rover.
The rover is exploring the “El Capitan” formation

Example



Example: Raw cross-correlation

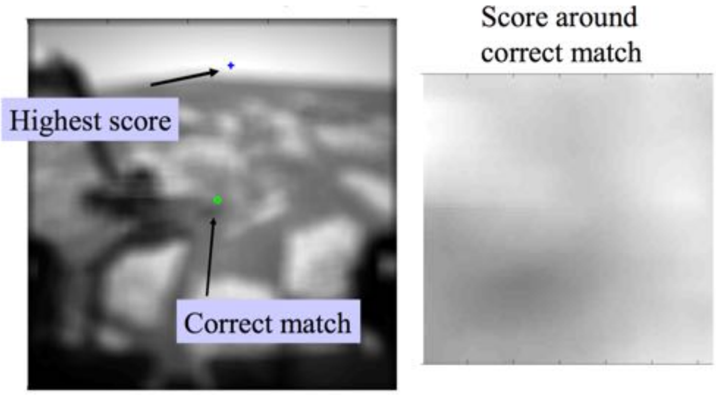


Image 2

Example: Cross-correlation

Note that score image looks a lot like a blurry version of image 2.

This clues us in to the problem with straight correlation with an image template.

Problem with correlation of raw image template

Consider correlation of template with an image of constant grey value:

$$\begin{array}{|c|c|c|}\hline a & b & c \\ \hline d & e & f \\ \hline g & h & i \\ \hline \end{array} \otimes \begin{array}{|c|c|c|}\hline v & v & v \\ \hline v & v & v \\ \hline v & v & v \\ \hline \end{array}$$

Result: v*(a+b+c+d+e+f+g+h+i)

Problem with correlation of raw image template

Now consider correlation with a constant image that is twice as bright

$$\begin{array}{|c|c|c|}\hline a & b & c \\ \hline d & e & f \\ \hline g & h & i \\ \hline \end{array} \otimes \begin{array}{|c|c|c|}\hline 2v & 2v & 2v \\ \hline 2v & 2v & 2v \\ \hline 2v & 2v & 2v \\ \hline \end{array}$$

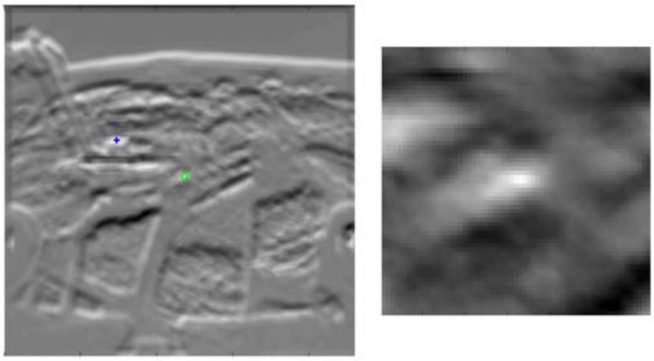
$$\begin{aligned}\text{Result: } & 2{*}v{*}(a+b+c+d+e+f+g+h+i) \\ & > v{*}(a+b+c+d+e+f+g+h+i)\end{aligned}$$

Larger score, regardless of what the template is!

Solution

- Subtract off the mean value of the template
- In this way, the correlation score is higher only when darker parts of the template overlap darker parts of the image, and the brighter parts of the template overlap brighter parts of the image

Correlation, zero-mean template



Better! But highest score is still not the correct match.
Note: highest score IS best within local neighborhood of correct match.

"SSD" or "Block matching"
(Sum of squared differences)

$$\sum_{[i,j]\in R} (f(i,j) - g(i,j))^2$$

1 – The most popular matching score
2 – T&V claim it works better than cross-correlation

Relation between SSD and Correlation

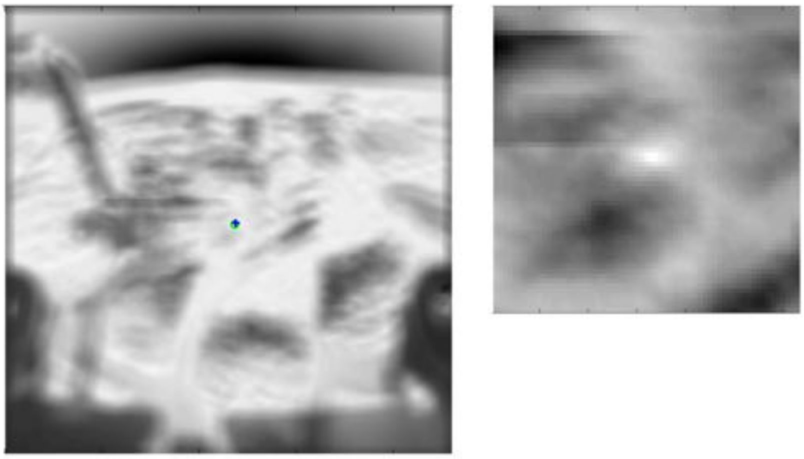
$$SSD = \sum_{[i,j]\in R} (f-g)^2$$

$$= \sum_{[i,j]\in R} f^2 + \sum_{[i,j]\in R} g^2 - 2\sum_{[i,j]\in R} fg$$

$$C_{fg} = \sum_{[i,j]\in R} f(i,j)g(i,j)$$

Correlation!

SSD



Best match (highest score) in image coincides with correct match in this case!

Handling intensity changes

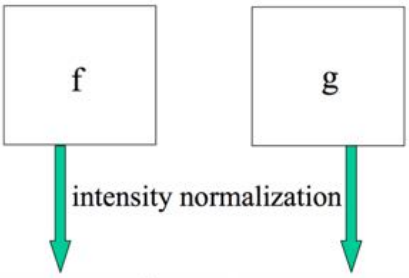
- the camera taking the second image might have different intensity response characteristics than the camera taking the first image
- Illumination in the scene could change
- The camera might have auto-gain control set, so that it's response changes as it moves through the scene.

Intensity normalization

- When a scene is imaged by different sensors, or under different illumination intensities, both the SSD and the C_{fg} can be large for windows representing the same area in the scene!
- A solution is to NORMALIZE the pixels in the windows before comparing them by subtracting the mean of the patch intensities and dividing by the std.dev.

$$\hat{f} = \frac{f - \bar{f}}{\sqrt{\sum(f - \bar{f})^2}} \qquad \hat{g} = \frac{g - \bar{g}}{\sqrt{\sum(g - \bar{g})^2}}$$

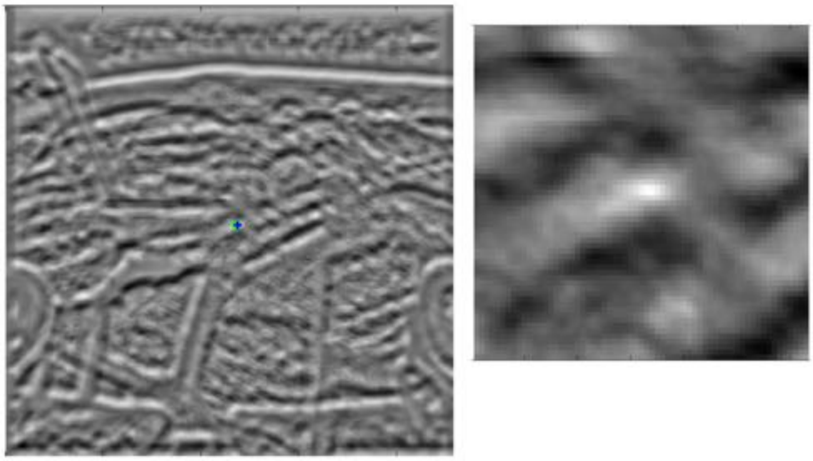
Normalized cross correlation



$$\hat{f} = \frac{f - \bar{f}}{\sqrt{\sum(f - \bar{f})^2}} \qquad \hat{g} = \frac{g - \bar{g}}{\sqrt{\sum(g - \bar{g})^2}}$$

$$\boxed{\text{NCC(f,g)} = C_{fg}(\hat{f}, \hat{g}) = \sum_{[i,j]\in R} \hat{f}(i,j)\hat{g}(i,j)}$$

Normalized cross correlation



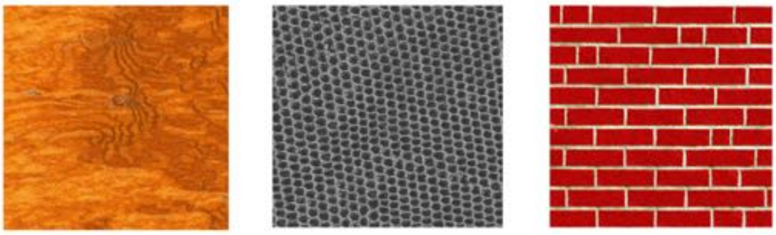
Highest score also coincides with correct match.
Also, looks like less chances of getting a wrong match.

Normalized cross correlation

- Important point about NCC:
  - Score values range from 1 (perfect match) to -1 (completely anti-correlated)
- Intuition: treating the normalized patches as vectors, we see they are unit vectors. Therefore, correlation becomes dot product of unit vectors, and thus must range between -1 and 1

Texture analysis

What is texture?



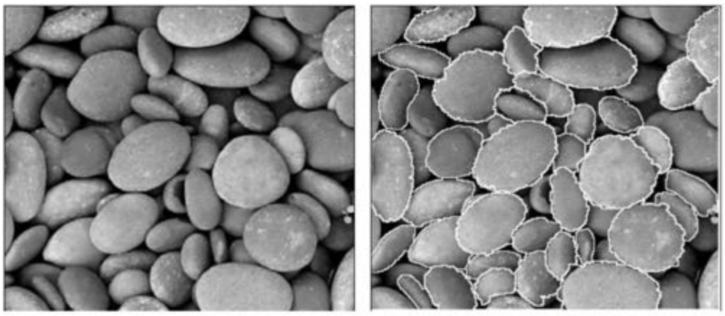
- An image obeying some statistical properties
- Similar structures repeated over and over again
- Often has some degree of randomness

Texture analysis

- Structural approach
- Statistical approach
- Fourier approach

Structural approach to describing texture

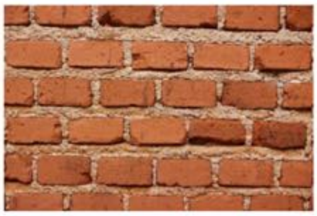
- A texture is a set of texture elements or **texels** occurring in some regular or repeated pattern



http://vision.ai.uiuc.edu/~sintod/

Aspects of texture

- Size/granularity (sand versus pebbles versus boulders)
- Directionality/Orientation
- Random or regular (stucco versus bricks)



Problem with structural approach

- What/where are the texels?
- Extracting texels in real images may be difficult or impossible

Statistical approach to texture

- Characterize texture using statistical measures computed from grayscale intensities (or colors) alone
- Less intuitive, but applicable to all images and computationally efficient
- Can be used for both classification of a given input texture and segmentation of an image into different regions

Some (simple) statistical texture measures

- Edge density and direction

- Use an edge detector as the first step in texture analysis
- The number of edge pixels in a fixed-size region tells us how busy that region is
- The directions of the edges also help characterize the texture

Two edge-based texture measures

1. edgeness per unit area

$F_{edgeness} = |\{ p \mid \text{gradient\_magnitude}(p) \geq \text{threshold}\}| / N$

where N is the size of the unit area

2. edge magnitude and direction histograms

$F_{magdir} = (H_{magnitude}, H_{direction})$

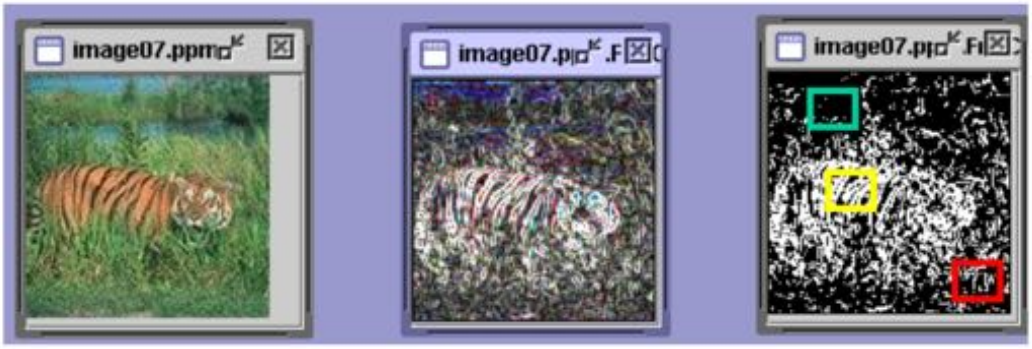
where these are the normalized histograms of gradient magnitudes and gradient directions, respectively.

Example

Original Image

Frei-Chen Edge Image

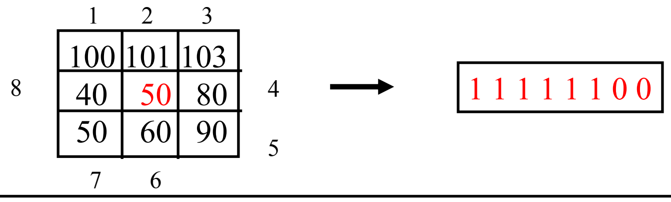
Thresholded Edge Image



Different F_{edgeness} for different regions

Local binary pattern measure

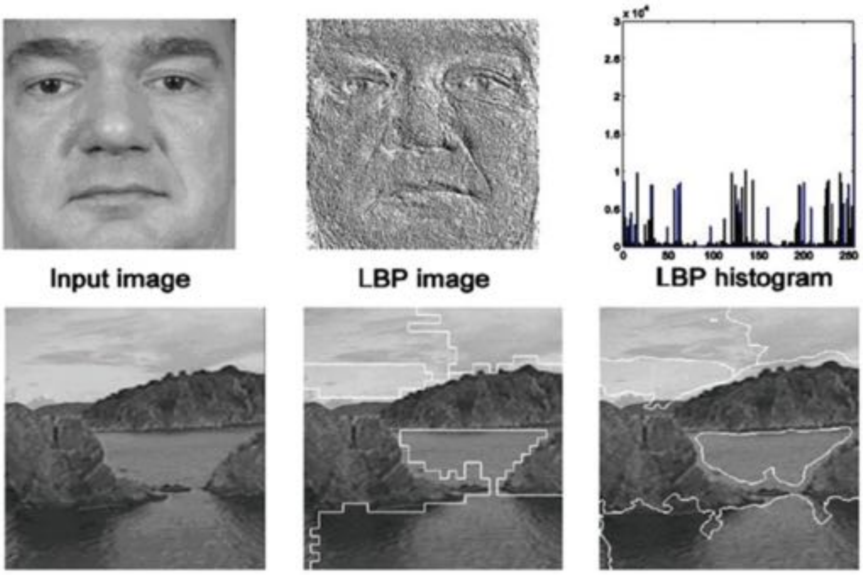
- For each pixel p, create an 8-bit number $b_1\ b_2\ b_3\ b_4\ b_5\ b_6\ b_7\ b_8$, where $b_i = 0$ if neighbor i has value less than or equal to p's value and 1 otherwise.
- Represent the texture in the image (or a region) by the histogram of these numbers.



100	101	103
40	50	80
50	60	90

1 1 1 1 1 1 0 0

LBP example



Texture energy features

- Use texture filters applied to the image to create filtered images from which texture features are computed
- Laws’ Technique (Laws, 1980)
  - Filter the input image using texture filters
  - Compute texture energy by summing the absolute value of filtering results in local neighborhoods around each pixel
  - Combine features to achieve rotational invariance

Law’s texture masks

- L5 (Level) = [1 4 6 4 1]
  - Gaussian: gives a center-weighted local average
- E5 (Edge) = [-1 -2 0 2 1]
  - Gradient: responds to row or col step edges
- S5 (Spot) = [-1 0 2 0 -1]
  - LOG: detects spots
- R5 (Ripple) = [1 -4 6 -4 1]
  - Gabor: detect ripples

Law’s texture masks (2D)

- Creation of 2D masks
- 1D masks are “multiplied” to construct 2D masks:
  - Mask E5L5 is the “product” of E5 and L5

$$\underset{\text{E5}}{\begin{bmatrix} -1 \\ -2 \\ 0 \\ 2 \\ 1 \end{bmatrix}} \times \underset{\text{L5}}{\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \end{bmatrix}} = \underset{\text{E5L5}}{\begin{bmatrix} -1 & -4 & -6 & -4 & -1 \\ -2 & -8 & -12 & -8 & -1 \\ 0 & 0 & 0 & 0 & 0 \\ 2 & 8 & 12 & 8 & 2 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}}$$

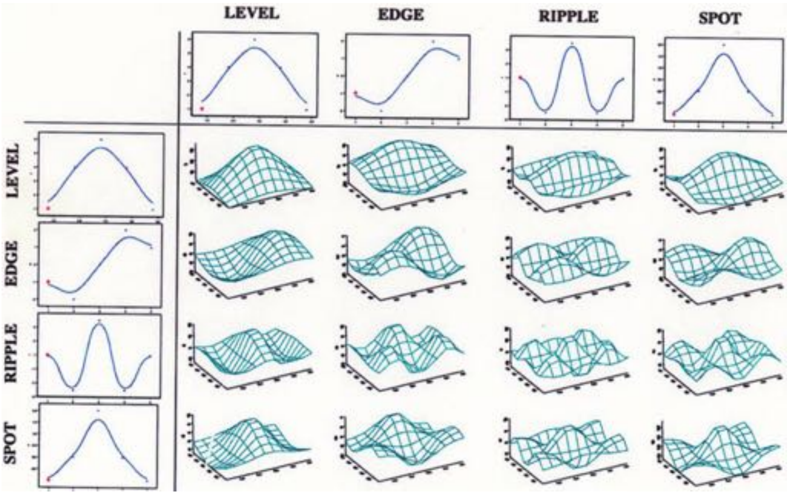
9D feature vector for each pixel

- Subtract mean neighborhood intensity from pixel (to reduce illumination effects)
- Filter the neighborhood with 16 5x5 masks
- Compute energy at each pixel by summing absolute value of filter output across neighborhood around pixel
- Define 9 features as follows (replace each pair with average)

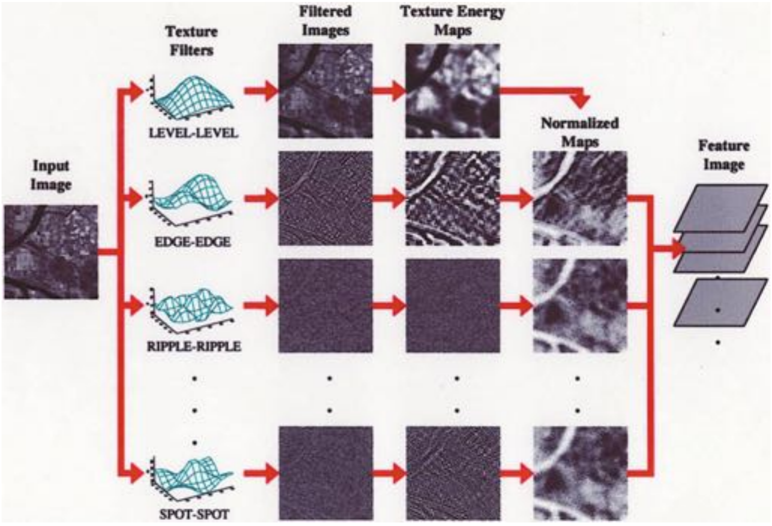
L5E5/E5L5
L5R5/R5L5
E5S5/S5E5
S5S5
R5R5

L5S5/S5L5
E5E5
E5R5/R5E5
S5R5/R5S5

Law’s filters



Law’s process



Texture energy features from sample images

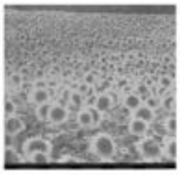
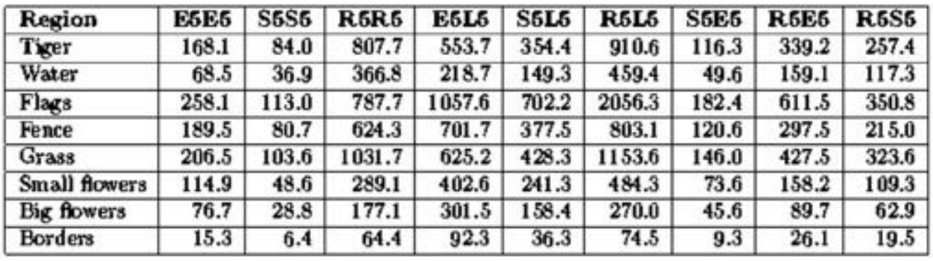
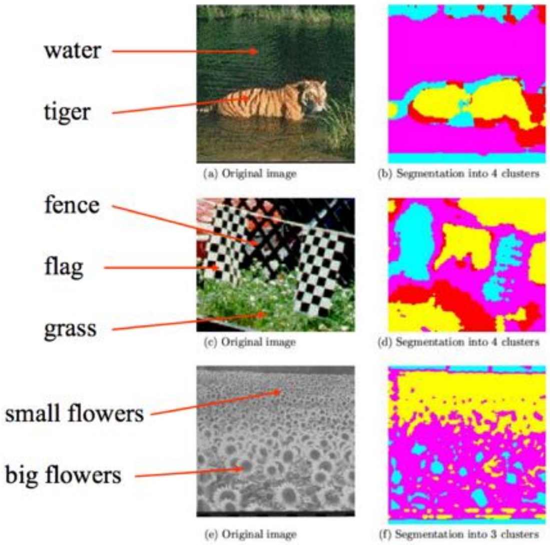


Table 7.2: Laws texture energy measures for major regions of the images of Figure 7.8.



Region	E5E5	S5S5	R5R5	E5L5	S5L5	R5L5	S5E5	R5E5	R5S5
Tiger	168.1	84.0	807.7	553.7	354.4	910.6	116.3	339.2	257.4
Water	68.5	36.9	366.8	218.7	149.3	459.4	49.6	159.1	117.3
Flags	258.1	113.0	787.7	1057.6	702.2	2056.3	182.4	611.5	350.8
Fence	189.5	80.7	624.3	701.7	377.5	803.1	120.6	297.5	215.0
Grass	206.5	103.6	1031.7	625.2	428.3	1153.6	146.0	427.5	323.6
Small flowers	114.9	48.6	289.1	402.6	241.3	484.3	73.6	158.2	109.3
Big flowers	76.7	28.8	177.1	301.5	158.4	270.0	45.6	89.7	62.9
Borders	15.3	6.4	64.4	92.3	36.3	74.5	9.3	26.1	19.5

Using texture energy for segmentation



(a) Original image (b) Segmentation into 4 clusters

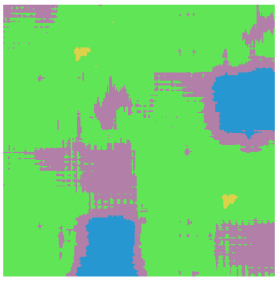
(c) Original image (d) Segmentation into 4 clusters

(e) Original image (f) Segmentation into 3 clusters

Statistical features - Entropy

$$Entropy = -\sum_i p_i \cdot \ln(p_i) \qquad p_i = \frac{\text{number of pixels with intensity value i}}{\text{total number of pixels}}$$

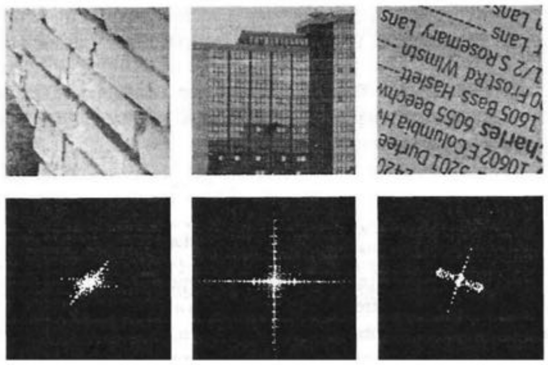
Original image



Entropy

Fourier approach: power spectrum and textures

Concentrated power → regularity
High frequency power → fine texture
Directionality → directional texture



Homework 4: Making & Solving puzzle

1. Find an image
2. Crop it into at least 6x6 pieces
3. Try to put the pieces to the whole image again

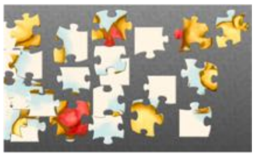
Note: It’s okay to use the original image as a guide

Homework 4: Challenge mode

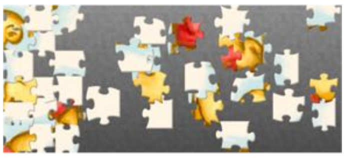
- For those who wants some extra points on final exam
- Note: all code in Python
- Maximum extra points: +5
- Implement the following (don't forget to comment the code):
  1. Create puzzle cut path (+1)
     Pieces must include "holes" (i.e., interlocking)
     Reference: https://blog.wolfram.com/2012/06/28/designing-jigsaw-puzzles-with-mathematica/
  2. Do cutting and display (+1)
  3. Solve the puzzle
     1. With original image guide (+1)
     2. Without original image (+2)
     3. With piece rotated (+1)

HW4 – challenge mode example

1. Cut path



2. Cut & display



3.2. Rotated pieces

Submission link

- https://forms.gle/s6xzWtYPoPWXD9698
- Deadline: 2 weeks
  - Nov 24, 2019 23:59