Lab 2
Introduction to Image Processing

Acknowledge: the slides are modified from those of previous years.

Introduction

- Basic concepts: read/display, color image/gray scale image
- Edge detection
- Flip an image
- Scale an image: reduce or enlarge

Background

- Digital images consist of pixels (picture elements).
- The brightness and color information for each pixel is represented by a number in a two dimensional array.
- The pixel values in an 8-bit gray scale image can take any value from 0 to 255.
- Black color is encoded by a value of 0 and white color by a value of 255.
- A color image is stored in a three-dimensional array (R, G, B).

Task 1

- Use `imread( )` to open a color image (DailyShow.jpg)
- Use `rgb2gray( )` to convert the color image to a gray level image
- Use `imshow( )` to display the image
- Use `size( )` to check the image size (NxM)
- In your report
  - Include the (1)Original and (2)8-bit gray scale image and specify its dimensions.

Edge detection

- Changes or discontinuities in an image amplitude attribute such as luminance
- Sobel row gradient operator (h1): detect vertical edges

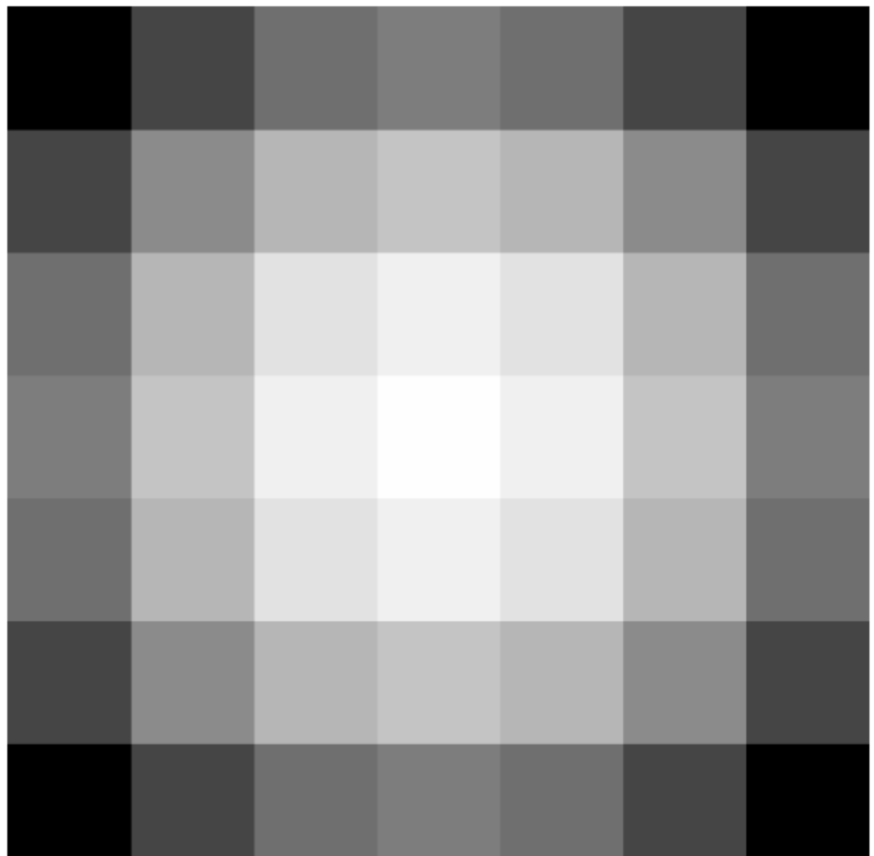
$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

- Sobel column gradient operator (h2): detect horizontal edges

$$\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

Edge detection

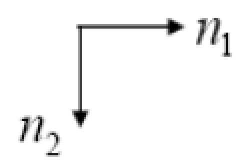
- Edge detection aims to identify the border with strongest gray scale change.



Edge detection

- Differentiation can be expected to sharpen the image
- In discrete domain, two-dimentional differentiation $\frac{\partial f(n_1, n_2)}{\partial n_1}$

  $\frac{\partial f(n_1, n_2)}{\partial n_2}$



- Emphasize the center row more

$$\frac{\partial f(n_1, n_2)}{\partial n_1} \approx [f(n_1+1, n_2-1) - f(n_1-1, n_2-1)]$$
$$+2[f(n_1+1, n_2) - f(n_1-1, n_2)]$$
$$+[f(n_1+1, n_2+1) - f(n_1-1, n_2+1)]$$

Edge detection

- Vertical Sobel Operator (h1) $\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$
- Emphasize the center column more

$$\frac{\partial f(n_1, n_2)}{\partial n_2} \approx [f(n_1-1, n_2+1) - f(n_1-1, n_2-1)]$$
$$+ 2[f(n_1, n_2+1) - f(n_1, n_2-1)]$$
$$+ [f(n_1+1, n_2+1) - f(n_1+1, n_2-1)]$$

- Horizontal Sobel Operator (h2) $\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$

Task 1

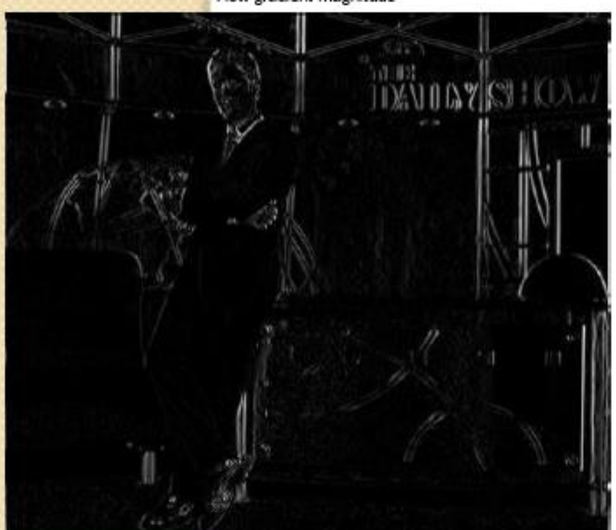
- Use `conv2()` to convolve your gray scale DailyShow.jpg with the h1 and h2, respectively.
- M1 is the row gradient (vertical edge) image (by convolving the gray scale DailyShow.jpg with h1
- M2 is the column gradient (horizontal edge) image (by convolving the gray scale DailyShow.jpg with h2)
- In your report display
  - |M1| is the magnitude of row gradient.
  - |M2| is the magnitude of column gradient.
  - $((M1^2 + M2^2)^{0.5})$ is the magnitude of the overall gradient.

Task 1

Original image

Gray scale image

Row gradient magnitude



Column gradient magnitude

Overall gradient magnitude

Task 1- even more

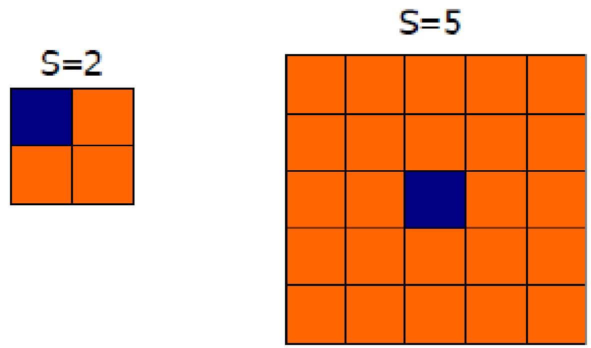
- In some versions of Matlab, when we use an image as input to the `conv2` function, its pixels should be floating-point values
  - `% X_vedge = conv2(double(X_gray), double(h1));`
  - `X_vedge = conv2(X_gray, h1);`
- `Imshow ()` should map the smallest value in the image to black, and the highest value in the image to white
  - `imshow(abs(X_vedge),[])`

Task 2

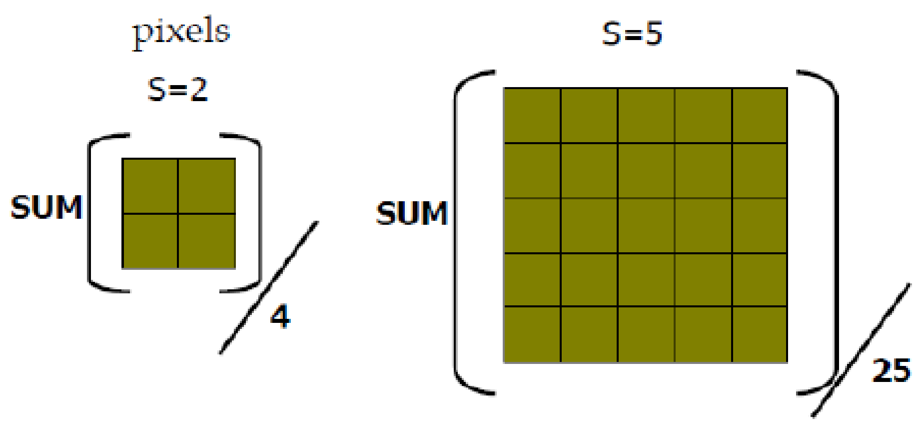
- Use your own image and repeat edge detection part of Task 1.

Task 3 Scaling

- Picking one out of S^2 pixels
  - Keep the center pixel when S is odd
  - Keep one of the 4 center pixels when S is even

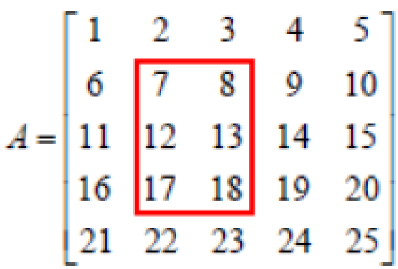


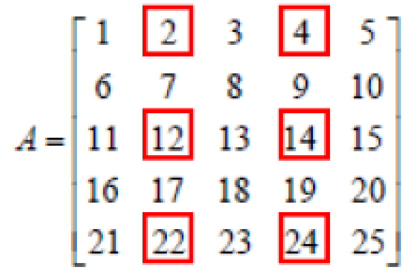
- Picking the average of each block with S^2 pixels



Task 3 Scaling

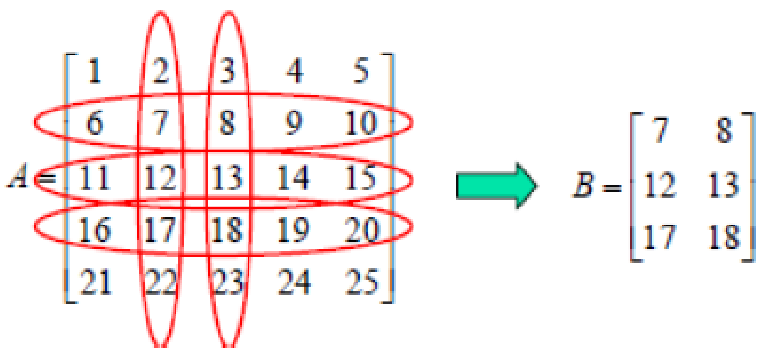
- Extract a sub-array

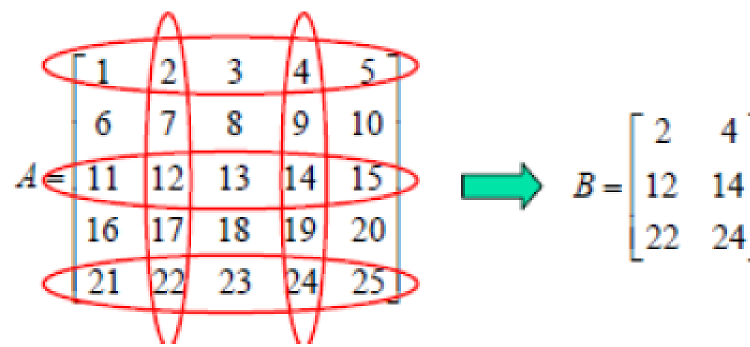




B = A(2:4,2:3);

B = A(1:2:end,2:2:end);





- Sum or average a two-dimensional array
  - `sum(sum(A));`
  - `mean(mean(A));`

Task 3 Scaling

Simple scaling, S = 2

Simple scaling, S = 5

Average scaling, S = 2

Average scaling, S = 5

Task 4 Flipping

- How the following images look like when compared to the original image $x[n,m]$
  - $x[N-n+1, m]$
  - $x[n, M-m+1]$
  - $x[N-n+1, M-m+1]$
- Use `fliplr()` and `flipud()` to very the answers

Task 4 Flipping

Gray scale image

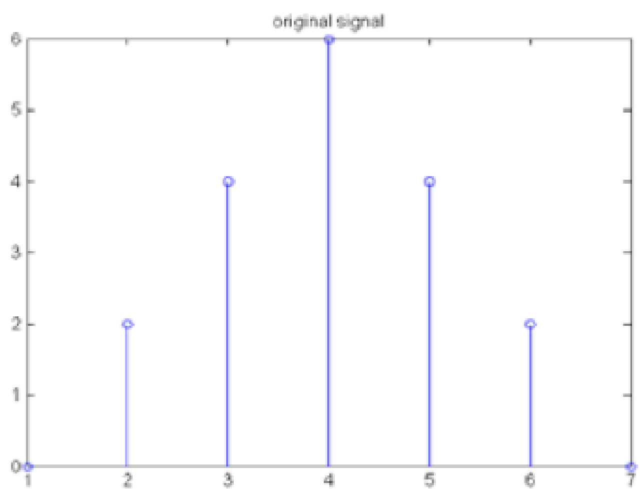
Horizontally flipped image

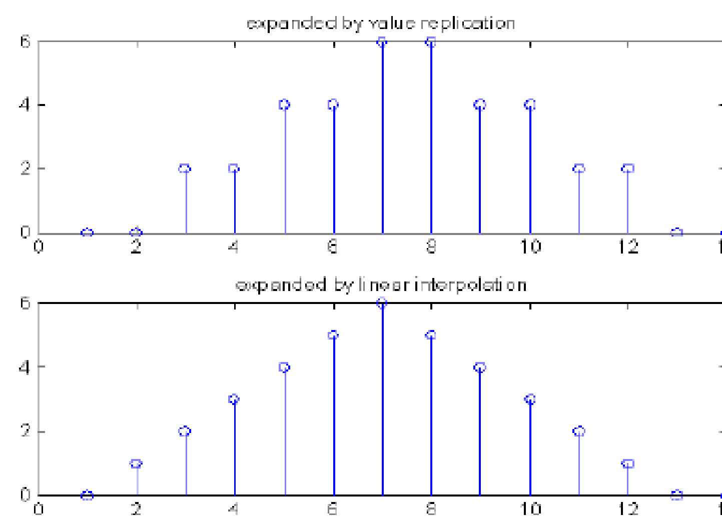
Vertically flipped image

Horizontally and vertically flipped image

Task 5: Expanding

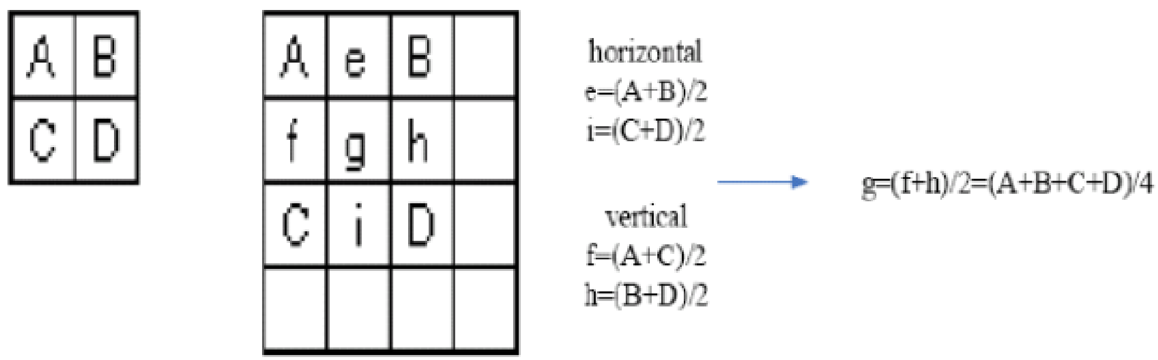
- Expand a 1-D signal [0,2,4,6,4,2,0] by a factor of 2
  - double each sample : value replication;
  - make the new samples half way between the previous samples :2 taps linear interpolation.





Task 5: Expanding

- Use "bilinear interpolation" to deal with a 2-D signal



- Use `interp2()` to expand the input image (DailyShow.jpg) with dimension NxM to a 2Nx2M image
  - `X_2large = interp2(double(X_gray));`

Task 5: Expanding

Gray scale image

Expanded the image by 2