Image Processing Lecture-2

Image file types and basic operations

Hazırlayan: M. Kemal GÜLLÜ

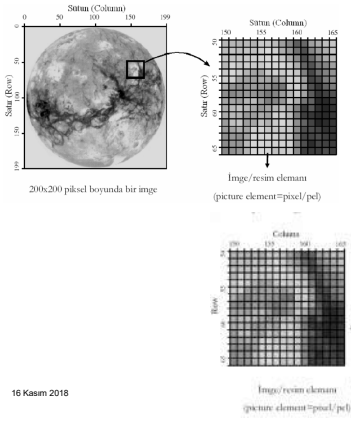


Image File Types

Images are stored in bmp, jpg, tiff, raw file formats.

What is the disk space required to store 8bit/pixel gray-scale image at 1000x1000 resolution?

The answer: 1000 x 1000 x 1 byte = 1.000.000 byte = 977 kbyte

When we consider color image (R,G,B color channels):

977 kbyte x 3 = 2931 kbyte= 2.86 Mbyte

Image File Types (raw)

- Only pixel intesity values are stored.
- No header to show image size.
- The pixel size must be know to read image properly.
- Typicall file operations in C and MATLAB are used to read this kind of files.

```
w=256;          % image width
h=256;          % image height
f=fopen('C:\Documents\lena.raw','r');
%f pointer

I=fread(f);     % The image is read from the disk
I=reshape(I,w,h);   % 2-d reshape operation
status=fclose(f);   % file closed

figure; imshow(uint8(I'));   % Image is shown on the screen
```

Image File Types (yuv)

- Color components are generally compressed via sub-sampling.
- No header file to store image size similar to raw format.
- Thus it is required to know file size before reading it.

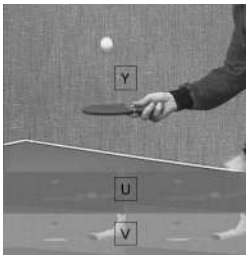


Image File Types (bmp)

- Commonly used format.
- Both compressed and uncompressed storage possible.
- In the case of uncompressed approach, a header contaning image size, bit-depth etc. are included to the original raw image data.
- Use the following MATLAB command to read it

```
I=imread('C:\Documents\lena.bmp');
```

- You have to use image processing libraries to read .bmp files such as C and Java.

Image File Types (Others)

- Tiff, jpeg, png, gif, pbm, pgm, hdf, pcx....
- Tiff is similar to bmp and has lossy compression mode.
- It supported 10 bit/pixsel, 16 bit/pixsel bit-depth storage.
- MATLAB's "imread" command is used to read it.
- JPEG is an lossy compression approach.
- It enables high compression ratio providing acceptable quality.
- MATLAB's "imread" command is used to read this file type as well.

Standard Image Sizes

- CIF: Common Intermediate Format
- VGA: Video Graphics Array
- SIF: Source Intermediate Format

SQCIF	128 × 96	VCD	352 × 240
QCIF	176 × 144	NTSC	720 × 480
CIF	352 × 288	PAL	720 × 576
4CIF	704 × 576	720p HD	1280 × 720
16CIF	1408 × 1152	1080p HD	1920 × 1080
VGA	640 × 480	4K UHDTV	3840 × 2160
QVGA	320 × 240	8K UHDTV	7680 × 4320
SCIF	352 × 240		

Transpose

$$B = A'$$

$$B(j,i) = A(i,j)$$

$$(i = 0,...,N-1,\ j = 0,...,M-1)$$

Rotation in horizontal axis

$$B(i, M-j-1) = A(i,j)$$

$$(i = 0,...,N-1,\ j = 0,...,M-1)$$

Rotation

- It is easy to rotate images at 90°, 180° and 270°.
- If it is required to rotate the image at arbitrary angle then some trigonometric operations need to be used.
- You may use "imrotate" available in MATLAB.

Ir=imrotate(I,angle,method);

angle: degree in counter-clock wise.

method: the interpolation approach will be used.
'nearest', 'bilinear', 'bicubic',

Ex:

Ir=imrotate(I,45, 'bilinear');

Image cropping

$$B(i,j) = A(n_1 + i, n_2 + j)$$

$$(i = 0,...,m_1 - 1,\ j = 0,...,m_2 - 1)$$

$$(n_1, n_2) \rightarrow \text{starting position}$$

$$(m_1, m_2) \rightarrow \text{ending position}$$

Image shifting

$$B(i,j) = A(i-n_1+1, j-n_2+1)$$
$$(i = n_1,\ldots,N, j = n_2,\ldots,M)$$
$$(n_1,n_2) \rightarrow \text{ starting point}$$

Image shifting

Example MATLAB for image shifting:

```
function [B]=my_shift(A,n1,n2)

[w,h]=size(A);

B=zeros(w,h);

for i=n1:w
   for j=n2:h

      B(i,j)=A(i-n1+1,j-n2+1);

   end
end
```

It is possible to eliminate for loops.

?

Image resize-zoom in

- Digital zoom-in means pixel size increase in images by making use of software approaches.

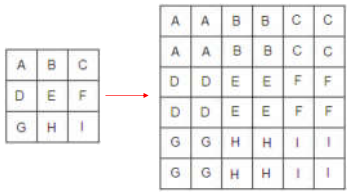
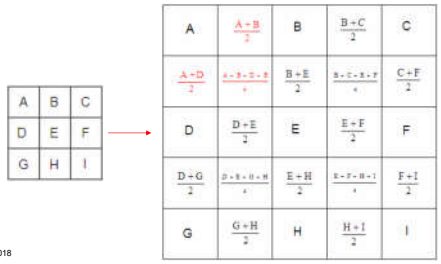


Image resize-zoom in

- A smoother approach.



boyut değiştirme-yakınlaştırma

- Which of the following is more clear?

Image resize-zoom out

- More than one pixel are processed to obtain a new pixel value in the smaller image.

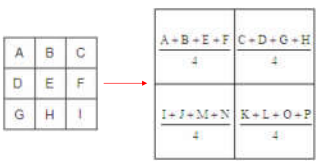


Image resize

- MATLAB's "imresize" function enables image resizing.

Is=imresize(I,ratio,method);

ration : resize ratio if ration>1 (bigger image), ratio<1 (smaller image).

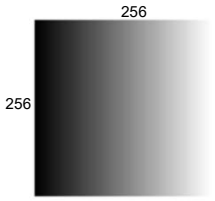
method : the interpolation approach.

Ex:

Is=imresize(I,0.97, 'bicubic');

How to create test images?

$$A = \begin{bmatrix} 0 & 1 & 2 & \dots & 255 \\ 0 & 1 & 2 & \dots & 255 \\ \vdots & & & & \\ 0 & 1 & 2 & \dots & 255 \end{bmatrix} \Big\} 256 \text{ rows}$$




```
for i = 1 : 256
  for j = 1 : 256
    A(i, j) = j − 1;
  end
end
```


Test images

- A circle located at (128,128) of radius 80 pixel

$$B(i,j) = \begin{cases} 255 & \text{if } \sqrt{(i-128)^2 + (j-128)^2} < 80 \\ 0 & \text{otherwise} \end{cases}$$

```
for i = 1 : 256
  for j = 1 : 256
    dist = ((i − 128)^2 + (j − 128)^2)^(.5);
    if (dist < 80)
      B(i, j) = 255;
    else
      B(i, j) = 0;
    end
  end
end
```


Test images

- ???


```
for i = 1 : 256
  for j = 1 : 256
    C(i, j) = A(i, j) ∗ B(i, j)/255;
  end
end
```


Image mean and variance

- Mean of an image

$$m_A = \frac{\sum_{i=0}^{N-1} \sum_{j=0}^{M-1} A(i,j)}{NM}$$

- Variance of an image

$$\sigma_A^2 = \frac{\sum_{i=0}^{N-1} \sum_{j=0}^{M-1} (A(i,j) - m_A)^2}{NM}$$

- Standard deviation of an image

$$\sigma_A = \sqrt{\sigma_A^2}$$