Mid-term Review

Key points

- Sampling (upsampling, interpolation, subsampling), quantization
- Spatial resolution vs. intensity resolution
- Thresholding
- Histogram, pdf, CDF
- Point operations: arithmetic operations, rounding and clipping, double/uint8 conversion, histogram stretching, histogram equalization
- Spatial filtering: border problem, linear spatial filter (average), median filter, max filter, min filter
- Gradient-based Edge detection: roberts, prewitt, sobel

Exercise 1-Scilab/Matlab

Write **ONE LINE** of MATLAB/SCILAB code for each of the following tasks.

1. Create a 5x5 average filter and store it in a variable *mask*.

2. Given a grayscale image *img*, find the *m x n* neighborhood whose centre is (x,y) and store it in a variable *Nb*;

3. store the 10th row of a 2D matrix *img* in a variable *row*

4. Given a grayscale image *I*, set all the pixels in *I* whose grey levels are bigger than 255 to be 255.

5. Replace the red channel of a colour image *img* with its blue channel.

6. store the value image of a colour image *img* in *V.*

Exercise 2-Point operations

For two colour images stored in variables *img1* and *img2* respectively, write a function called *img_add(img1,img2)* to generate an image *newimg* by adding *img1* and *img2* and display *newimg*.

Note: this function can only support colour images and the size of two colour images should be the same.

Exercise3-Spatial filtering

For a grayscale input image stored in a variable *img*, write a function *img_filter* to 1) blur the input image using a 5*5 average filter , 2)display the blurred image *newimg, and* 3)return the blurred image *newimg*.

Note: the boundary pixels **do not** need to be considered.

Exercise 1-Answer

Write **ONE LINE** of MATLAB/SCILAB code for each of the following tasks.

1. Create a 5x5 averaging filter and store it in a variable *mask*.

```
mask=ones(5,5)/25;
```

2. For 2D matrix *img*, extract out the *m x n* neighborhood whose centre is (x,y) and store it in a variable *Nb*.

```
Nb=img(x-(m-1)/2:x+(m-1)/2,y-(n-1)/2:y+(n-1)/2);
```

3. Store the 10th row of a 2D matrix *img* in a variable *row.*

```
row=img(10,:);
```

4. Given a grayscale image *I*, set all the pixels in *I* whose grey levels are bigger than 255 to be 255.

```
I(I>255)=255;
```

5. Replace the red channel of a color image *img* with its blue channel.

```
img(:,:,1)=img(:,:,3);
```

6. store the value image of a color image *img* in *V.*

```
V=uint8(sum(img,3)/3);
```

Exercise 2-Sample Answer

```
function newimg= imag_add(img1,img2)

if length(size(img1))~=3 || length(size(img2))~=3
return
end
[m n]=size(img1);
if size(img2,1)~=m||size(img2,2)~=n

  return
end

newimg=double(img1)+double(img2);
newimg(newimg>255)=255;
newimg=uint8(newimg);
imshow(newimg);
end
```

Exercise3-Sample answer

```
function newimg=img_filter(img)
 w=ones(5)/(5^2);
 [r, c]=size(img);
 a=2;
 b=2;
 img=double(img);
 newimg=zeros(r,c);

 for i=1+a:r-a
    for j=1+b:c-b
       neighbors=img(i-a:i+a,j-b:j+b);
       newimg(i,j)=sum(sum(w.*neighbors));
   end
 end

 newimg=newimg(1+a:r-a,1+b:c-b);
 newimg=round(newimg);
 newimg(newimg>255)=255;
 newimg(newimg<0)=0;
 newimg=uint8(newimg);
end
```