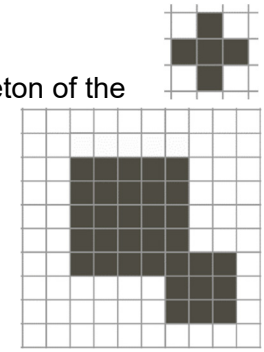
Video Signals

Date: 24 January 2022

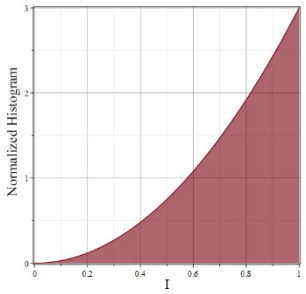
Ex.1.[9 pts]



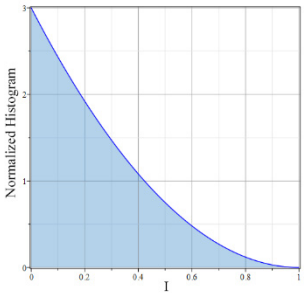
Given the structing element on the right, provide the morphological skeleton of the following black and white image (black=1 and white=0) indicating for each skeleton level the position of the seeds and the associated value.

Es.2. [11 pts]

In a grayscale continuous image (there is no discretization on the brightness levels) the grey values range from 0 to 1 with a histogram H proportional to the square of the intensity: $H(I) \propto I^2$ as depicted in the figure on the left.



We want to equalize the image: provide a transfer function $F(I)$ for the image brightness that allows to perform this task. [Hint: the histogram can be assumed as $H(I) = 3I^2$].



What further transfer function $F'(I)$ should be applied in order to convert the histogram into the blue one on the right? [Hint: the final histogram should be $H'(I) = 3(I-1)^2$].

Es.3. [11 pt to be done uploading the MATLAB code on Webeep]

You want to develop a method able to remove homogenues background from photos such as the one depicted below. In order to do so you choose to calculate the local entropy of the image in order to spot homogenues regions.

Write a MATLAB script able to perform the following steps:

Continues on the back

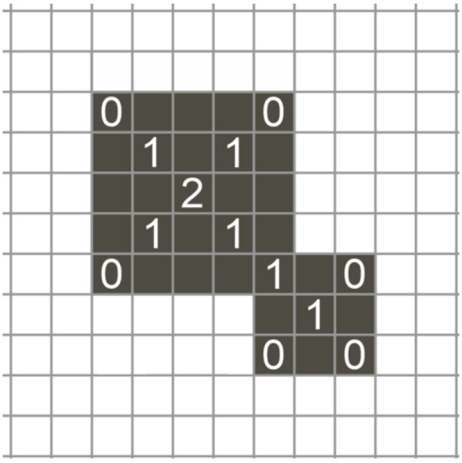
a) Read the 8-bit input color image (stored in the file *'test_img.png'*), convert it to a grayscale image and visualize it.
b) Initialize a matrix called *ent* which will contain the local entropy with the same size of the input image.
c) Consider every nonoverlapping 8x8 region inside of the image and for each one of them compute the following operations (*hint: you can use for i = start:step:end*):
    I. Calculate the 256 bins histogram of the 8x8 region and normalize it, in order to obtain *p* a vector with elements summing to 1.
    II. Calculate the region local entropy as $E = -\sum p \, log_2(p + 10^{-6})$
    III. Assign the obtained value *E* to the corresponding 8x8 region of *ent*
d) Using a linear transformation rescale the values stored in *ent* to [0,1] range and obtain *mask* binarizing *ent* with a fixed threshold of 0.5
e) Define a square structuring element with size equal to 20 and apply two morphological operations of your choice able to remove small positive regions and fill small black gaps.

Use *mask* in order to set to black the detected background of the color input image and visualize the result.

Solutions

Ex.1

The skeleton, in the figure below, is represented by the pixels with a number and, in particular, the value in these pixels represents the skeleton level.



Ex.2

According to the course material, the normalized histogram can be assimilated to a probability density function (pdf) and the transfer function for equalization will be the integral of the intensity (equivalent to the cumulative density function, cdf).

$$F(I) = \int_0^I h(\tau)\, d\tau = \int_0^I \frac{1}{3}\tau^2 d\tau = I^3$$

To map the equalized histogram into the translated blue parabola we have to apply the inverse procedure in order to map a uniform histogram into a the desired one. In this case the direct mapping from the parabolic into a uniform histogram would be:

$$F'(I) = \int_0^I h(\tau)\, d\tau = \int_0^I 3(\tau - 1)^2 \, d\tau \overset{\{\lambda = \tau - 1\}}{=} \int_{-1}^{I-1} 3\lambda^2 d\lambda = \lambda^3 \Big|_{-1}^{I-1} = (I-1)^3 + 1$$

So, the inverse mapping will be:

$$F'(I)^{-1} = \sqrt[3]{I-1}+1$$

If you want you can check the solution using matlab:

```
x=rand(1000000,1);
y=nthroot((x-1),3)+1;
histogram(y);
```

Ex.3 Matlab code

```
%a)
img = imread('test_img.png');
img_gray = rgb2gray(img);
figure
imshow(img_gray)

%b)
ent = zeros(size(img_gray));

%c)
grid_size = 8;
for i=1:grid_size:size(img_gray,1)
    for j=1:grid_size:size(img_gray,2)
        I = img_gray(i:(i+grid_size-1),j:(j+grid_size-1));
        %c1)
        p = imhist(I);
        p = p/numel(I);
        %c2)
        entropy = -sum(p.*log2(p+1e-6));
        %c3)
        ent(i:(i+grid_size-1),j:(j+grid_size-1)) = entropy;
    end
end
%d)
ent = (ent-min(ent(:)))/(max(ent(:))-min(ent(:)));
mask = im2bw(ent,0.5);
%d)
se = strel('disk', 20);
mask = imclose(mask, se);
mask = imopen(mask, se);
%f)
img = img .* uint8(mask);
figure
imshow(img)
```