HW1:Fundamentals

1、Exercises

1.1、If we consider an N-bit gray image as being composed of N 1-bit planes, with plane 1 containing the lowest-order bit of all pixels in the image and plane N all the highest-order bits, then given a 2048 2048, 256-level gray-scale image:

1. How many bit planes are there for this image?

2. Which panel is the most visually significant one?

3. How many bytes are required for storing this image? (Don't consider image headers and compression.)

Sol: 1、Since each plane store a bit for describing the gray level, there are totally 256 gray level, since 2^8 = 256, we need 8 bits to describe it, thus we need 8 bit planes for this image.

2、Since the plane N contains the highest-order bits for pixels in the image, a change of 1/0 can cause 128 gray levels' change in the pixel, thus plane 8 is the most visually significant one.

3、Since each plane contains one bit for each pixel in the image, the byte for a plane is 2048*2048*1 bits = 0.5*1024*1024 bits = 0.5*1024*1024 bytes = 0.5 MB, since there are 8 planes, the total size of this image is 8*0.5 = 4MB.

1.2、Figure 1 is a 5 5 image. Let V = {1; 2; 3} be the set of pixels used to define adjacency. Please report the lengths of the shortest 4-, 8-, and m-path between p and q. If a particular path does not exist, explain why.

$$\begin{array}{cccccc} & 3 & 4 & 1 & 2 & 0 \\ & 0 & 1 & 0 & 4 & 2(q) \\ & 2 & 2 & 3 & 1 & 4 \\ (p) & 2 & 0 & 4 & 2 & 1 \\ & 1 & 2 & 0 & 3 & 4 \end{array}$$

Figure 1: Adjacency.

Sol:

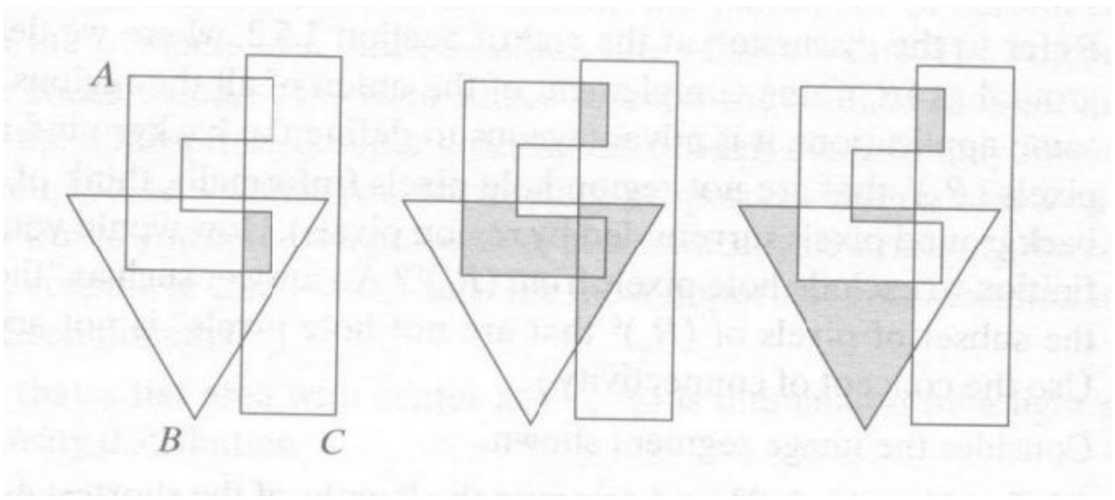
According to the definition, the 4-path or 8 path need the pixel in N4(p) or N8(p) and the gray-level in V, the m-path need either q in N4(p) or q in Nd(p) and the join of N4(p) and N4(q) is not in V.

First, let us consider the shortest 8-path, which is much easier, since the condition is quite loose, we can find many 8-paths, the length of shortest one is 4, the path is {p, 0, 3, 1, q}

Second, let us consider the shortest 4-path, there has no such path, because the 4-neighbors of q is 0, 4, 4, none of them meet the require of gray-level in V, thus we can not access q in 4-path.

Third, let us consider the shortest m-path, since the m-path is strict than 8-path, the length of it may be longer than 8-path, the length of shortest one is 5, one of the paths is {p, 0, 4, 2, 1, q}.

1.3、Figure 2 are three different results of applying logical operations on sets A;B and C. For each of the result, please write down one logical expression for generating the shaded area. That is, you need to write down three express- ions in total.



Sol:

For the first picture, obviously the shadow is total area shared by A、B and C. Thus it can be expressed as $A \cap B \cap C$.

For the second picture, the area is composed with three parts, each part is the join of either A、B, B、C or C、A. Thus it can be expressed as $(B \cap C) \cup (B \cap A) \cup (A \cap C)$.

For the third picture, the area is consist of two parts, the upper one can be expressed as $A \cap C - B$, the lower one can be expressed by $B - (A \cup C)$, the final expression is $((A \cap C) - B) \cup (B - (C \cup A))$.

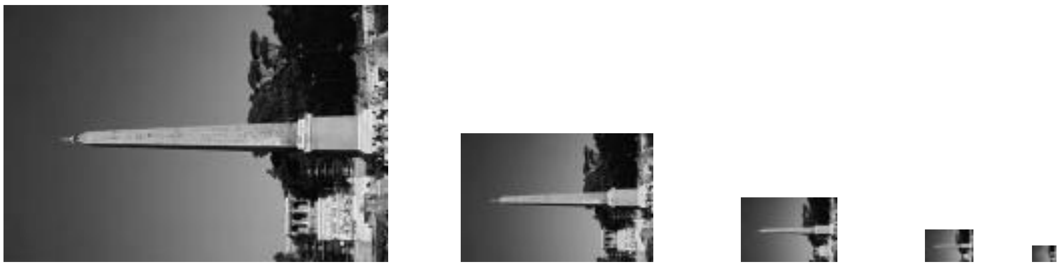
2、Programming Tasks

If you want to see the result produced by matlab, you can open the src folder, read the 'main.m' document, and type 'main' to call this function, then you will see the result.

2.2 Scaling

1、Use the function scale(input_img, size) to down-scale to 192*128, 96*64, 48*32, 24*16 and 12*8, the invoking of this function is scale(imread('24.png'), [192, 128]) , etc.

Here is the result:



192*128 96*64 48*32 24*16 12*8

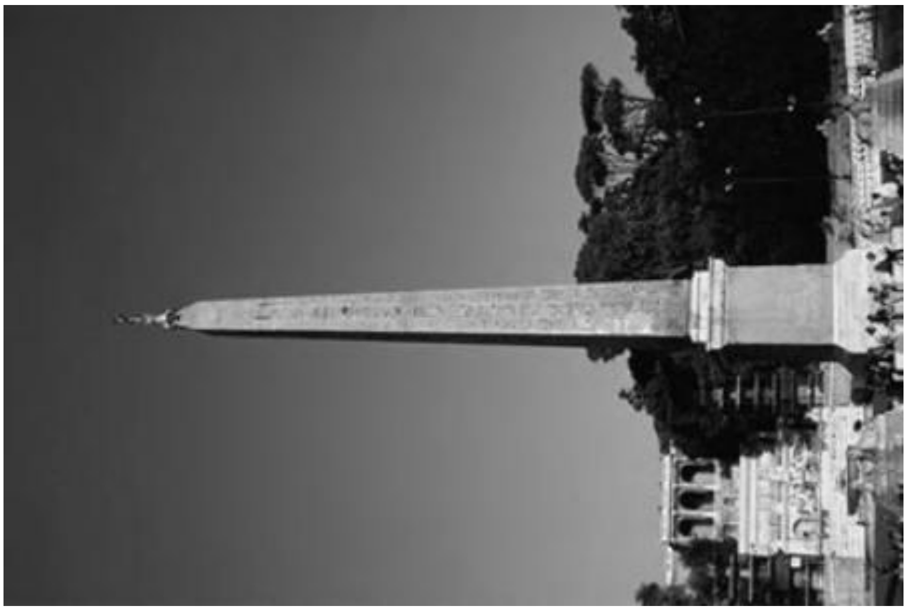
2、down-scale the image to 300*200, since the transformation is not integer times, we can only use bilinear interpolation algorithm to scale the image, the invoking of this function is scale3(imread('24.png'), [300, 200]).

Here is the result:

300*200

3、Up-scale the image to 450*300, like 2, we can only use bilinear interpolation algorithm to scale the image, the invoking of this function is scale3(imread('24.png'), [450, 300]).

Here is the result:



4、Scale the image to 450*300, like 2, we can only use bilinear interpolation algorithm to scale the image, the invoking of this function is scale3(imread('24.png'), [500 200]).

Here is the result:

5、When I first take this task, I remembered teacher has taught us to down-scale the image by multiplying with two matrix, if I have a origin matrix A of m*n, I want to change it to a matrix B of a*b, I need a matrix C of a*m, a matrix D of n*b, according to the linear algebra, C*A*D = B, Normally, left multiplying matrix is making combination of rows of A, the right multiplying matrix is making combination of columns of A, and the left multiplying matrix often has the following format:

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

The right multiplying matrix is often the transpose of it(with size n*b). The element in the matrix is each row(column) has n element with value 1/n, the remaining is 0, when multiplying, it takes the average of adjacent n elements and scales the rows(columns) to 1/n times.

But when scale factor is not integer, the method of multiplying matrix can't work, the bilinear interpolation algorithm is used most to solve this task.

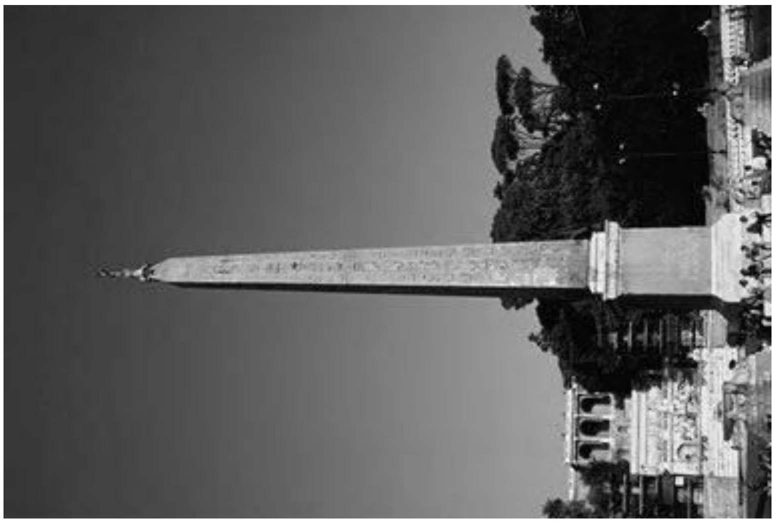
First, we want to reduce a m*n matrix A to a a*b matrix B, we calculate the factor w = m/a, h = n/b, when referring to a element in B, for example (i, j), the corresponding position in A is (i*w, j*h), this position may not be integer, thus we find the nearest four integer position in A, and taking the average of them as the result of B(i, j).

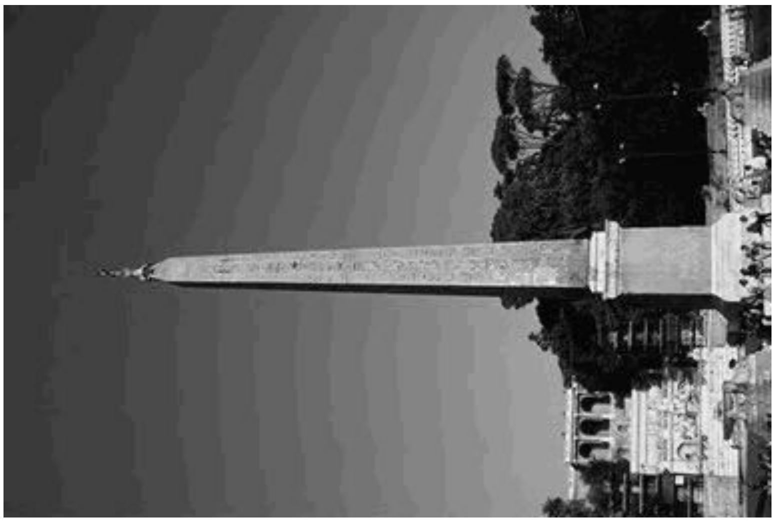
If we calculate all the elements in B, we have scaled a m*n matrix to a a*b matrix already.

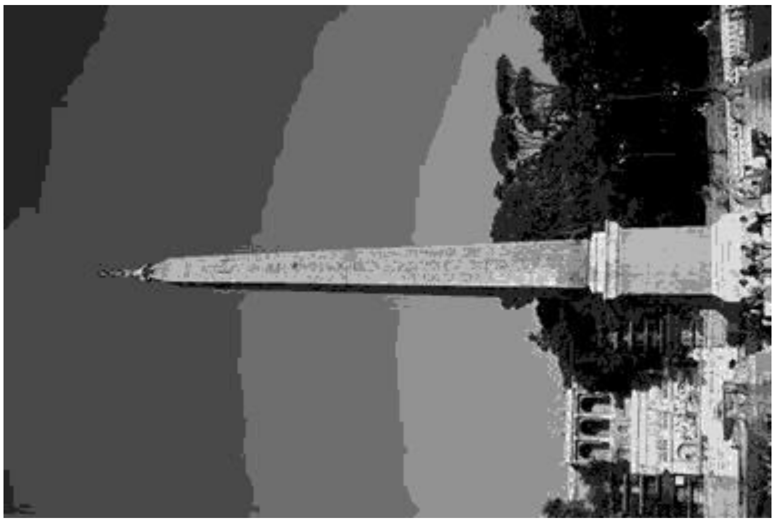
2.3 Quantization

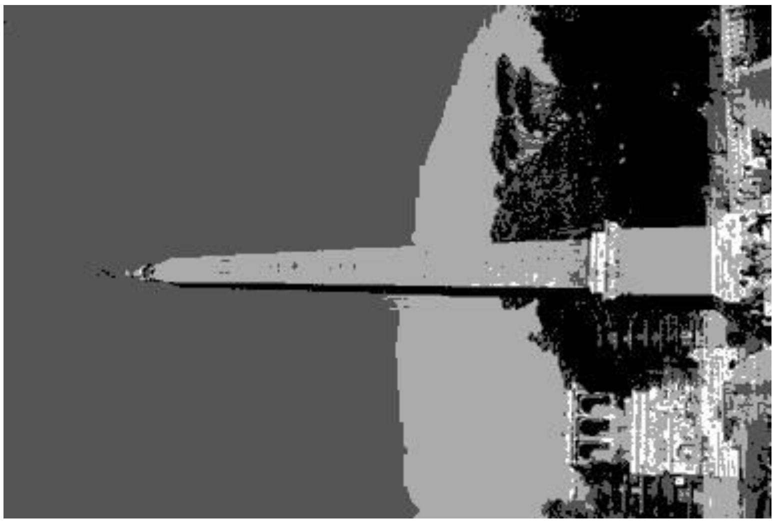
1、Reduce gray level resolution to 128; 32; 8; 4 and 2 levels, we use the interval method to divide the 256 into several intervals, and then map the origin value to corresponding value, the invoking of this method is quantize(input_img, level).

Here is the result:

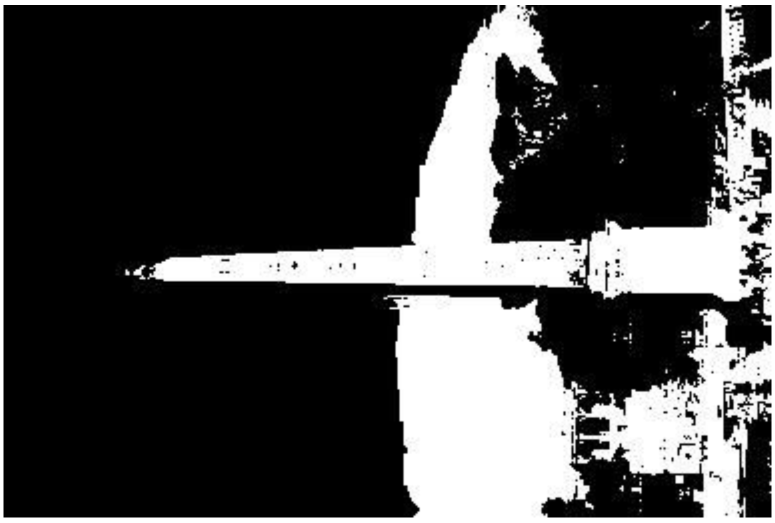

128 level


32 level


8 level



4 level



2 level

2、We can solve this task by considering the notion quantization, like transferring continuous variables into discrete variables, we need to divide the origin 256 levels into several interval, if the number of interval is n, which means that the number of endpoint is n+1, and we only need to consider this n+1 endpoint, and the gray level of this matrix is n+1, but how can we map the origin matrix into target matrix?

We visit each element in the matrix, and find which interval the element is in, find the nearest endpoint and take the endpoint's gray level as result. After visiting and modifying all elements in the matrix, the task has been done.