# DIP Assignment 1: Fundamentals

16327109 谢昆成

## 1. Exercise

### 1.1

(1) Since the image is 64-level gray-scale and $2^6 = 64$, 6 bit planes are there for this image.

(2) The most significant one is the sixth plane, for it is the highest-order bit in the image and has the most weight in gray.

(3) $1024 * 2048 * 6bits = 12582912bits = 1572864bytes$

So 1572864 bytes are required for storing the image.

### 1.2

4-path: not exist, for there is no a path from p to q where all nodes are 4-adjacency.

8-path: 4

m-path: 5

## 2. Programming Tasks

### 3.1

(1) Down-scale to 192 x 128 (width: 192, height: 128), 96 x 64, 48 x 32, 24 x 16 and 12 x 8, then manually paste your results on the report. (10 Points)

192*128

96*64

48*32

24*16



12*8



(2) Down-scale to 300 x 200, then paste your result. (5 Points)

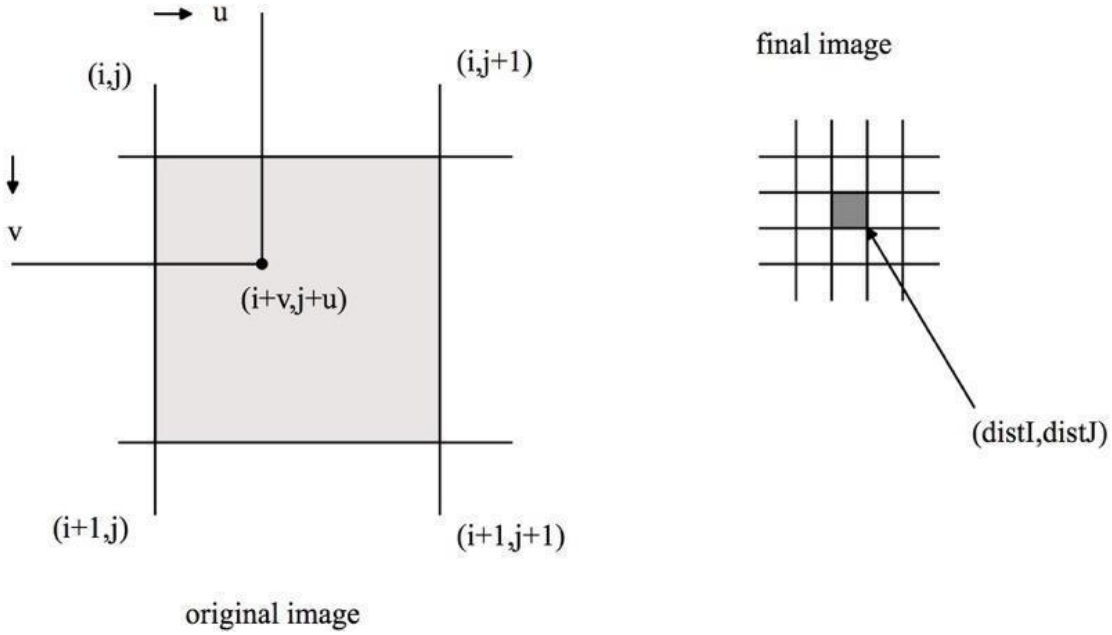
(3) Up-scale to 450 x 300, then paste your result. (5 Points)

(4) Scale to 500 x 200, then paste your result. (5 Points)

(5) Please discuss how you implement the scaling operation in details, i.e., the “scale" function, in less than 2 pages. Please focus on the algorithm part. If you have found interesting phenomenons in your scaling results, analyses and discussions on them are strongly welcomed and may bring you bonuses. But please don’t widely copy/paste your codes in the report, since your codes are also submitted. (20 Points)

Let (distI,distJ) denote the coordinates of the location in the final image to which we want to assign an intensity value, and let $f'(distI, distJ)$ denotes the intensity value in final image and $f(i,j)$denotes the value in original image.

Then (distI,distJ) correspond to the coordinates (i+v,j+u) in the source image. In bilinear interpolation, we first use linear interpolation in x direction for (i,j+u) and (i+1,j+u) with the 4 points near to (i+v,j+u) and then use linear interpolation in y direction for (i+v,j+u) with (i,j+u) and (i+1,j+u).




So, we can get the formulations:

$$f'(distI, distJ) = f(distI * \frac{original\ width}{final\ width}, distJ * \frac{orginal\ height}{final\ height}) = f(i+v, j+u)$$

$$f(i, j+u) = u * f(i,j) + (1-u) * f(i, j+1)$$

$f(i+1,j+u) = u * f(i+1,j) + (1-u) * f(i+1,j+1)$

$f(i+v,j+u) = v * f(i,j+u) + (1-v) * f(i+1,j)$

These equations may be written in the simplified form as follows.

$f'(distI, distJ) = f(i+v,j+u) = v*u*f(i,j) + v*(1-u)*f(i,j+1) + (1-v)*u*f(i+1,j) + (1-v)*(1-u)*f(i+1$

Now we can write the algorithm.

First we need to get i, j, u and v

$i = floor(distI * \frac{original\ width}{final\ width})$

$j = floor(distJ * \frac{orginal\ height}{final\ height})$

$v = distI * \frac{original\ width}{final\ width} - i$

$u = distJ * \frac{orginal\ height}{final\ height} - j$

We then substitute them in $f(i+v,j+u)$ to get the answer.

I find that the scaled images with bilinear interpolation have been introduced some noise in vertical and horizontal direction especially when it is up-scaled. In my view, it is that we first use linear interpolation in x direction and then use linear interpolation in y direction, making it unsmooth. I think it will be better to use bicubic interpolation which involves 16 nearest neighbors of a point.

## 3.2

(1) Reduce gray level resolution to 128, 64, 32, 8 and 2 levels, then paste your results respectively. Note that, in practice computers always represent “white" with the pixel value of 255, so you should also follow this rule. For example, when the gray level resolution is reduced to 4 levels, the resulting image should contain pixels of 0, 85, 170, 255, instead of 0, 1, 2, 3. (10 Points)

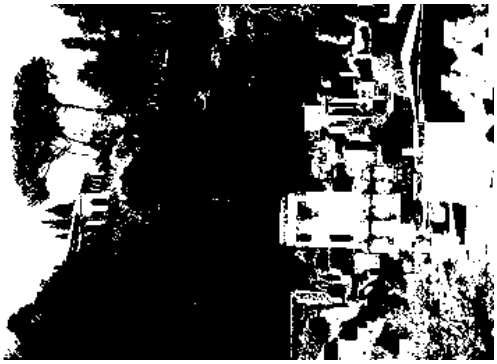
128

64

32

8

2



(2) Please discuss how you implement the quantization operation in details, i.e., the “quantize" function, in less than 2 pages. Again, please focus on the algorithm part. Analyzing and discussing interesting experiment results are also welcomed, but please do not widely copy/paste your codes in the report (15 Points).

To quantize a gray-level image, we first divide 256 gray levels into given gray levels, like [0, 85, 170, 255] if the given gray level is 4.

Let $f(x, y)$ be the value of the original point$(x, y)$ and the $f'(x, y)$ denotes the intensity value in final image. $m$ be the given gray level.

The gap between two adjacent level is $floor(256/(m-1))$

And the level $f'(x, y)$ belong to is $floor(f(x, y) * m/256)$

So, we can use the following formulation to calculate $f'(x, y)$ .

$f'(x, y) = floor(f(x, y) * m/256) * floor(256/(m-1))$

We find that the image is still acceptable in 32 levels. It means that we are unsensitive to detailed change and can't distinguish many gray levels in an image. Resort to quantization we can compress an image effectively.