Digital Image Processing

Lecture 22nd May, 2017

Image Scaling

- Zooming (up scaling, resizing upward) requires two steps
  - The Creation of new pixel locations
  - Assignment of gray levels to new pixel locations

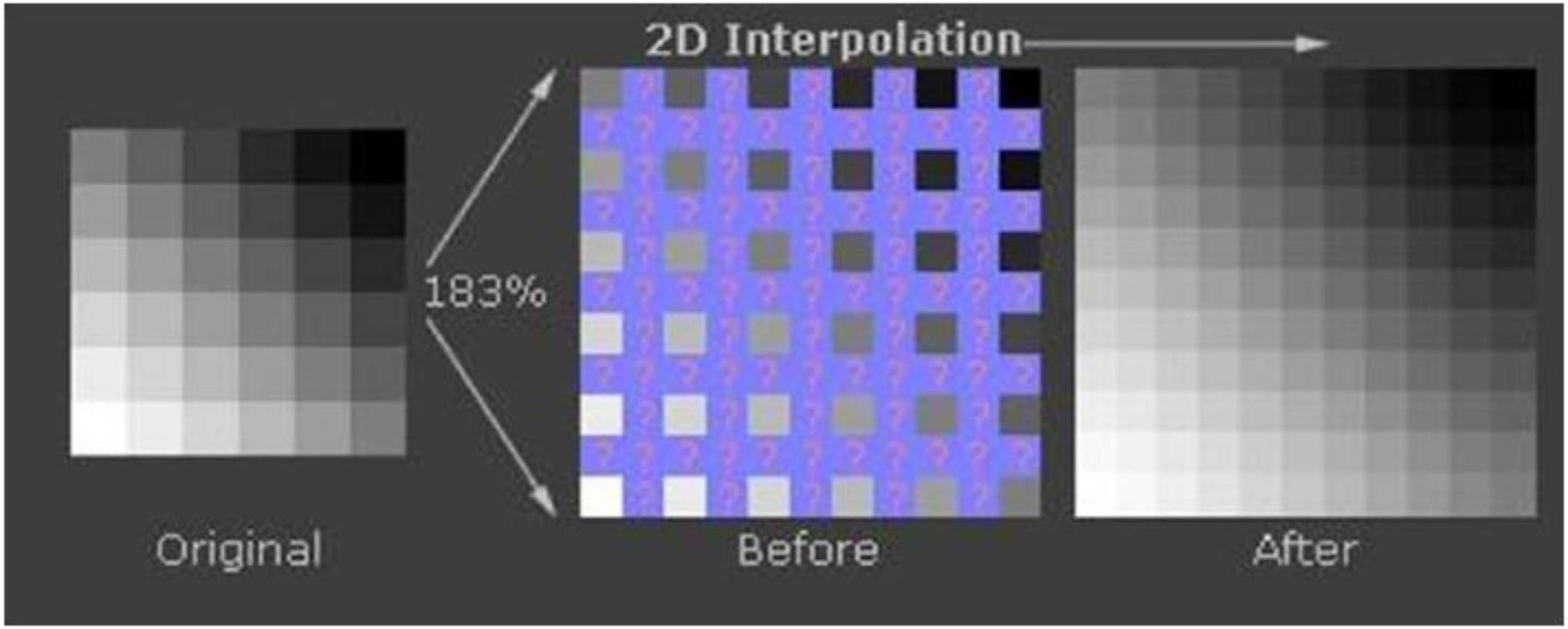


Image Scaling

- Zooming (up scaling, resizing upward) can be achieved by the following techniques::

  - Nearest neighbor Interpolation
  - Pixel Replication
  - Bilinear Interpolation
  - Bicubic Interpolation

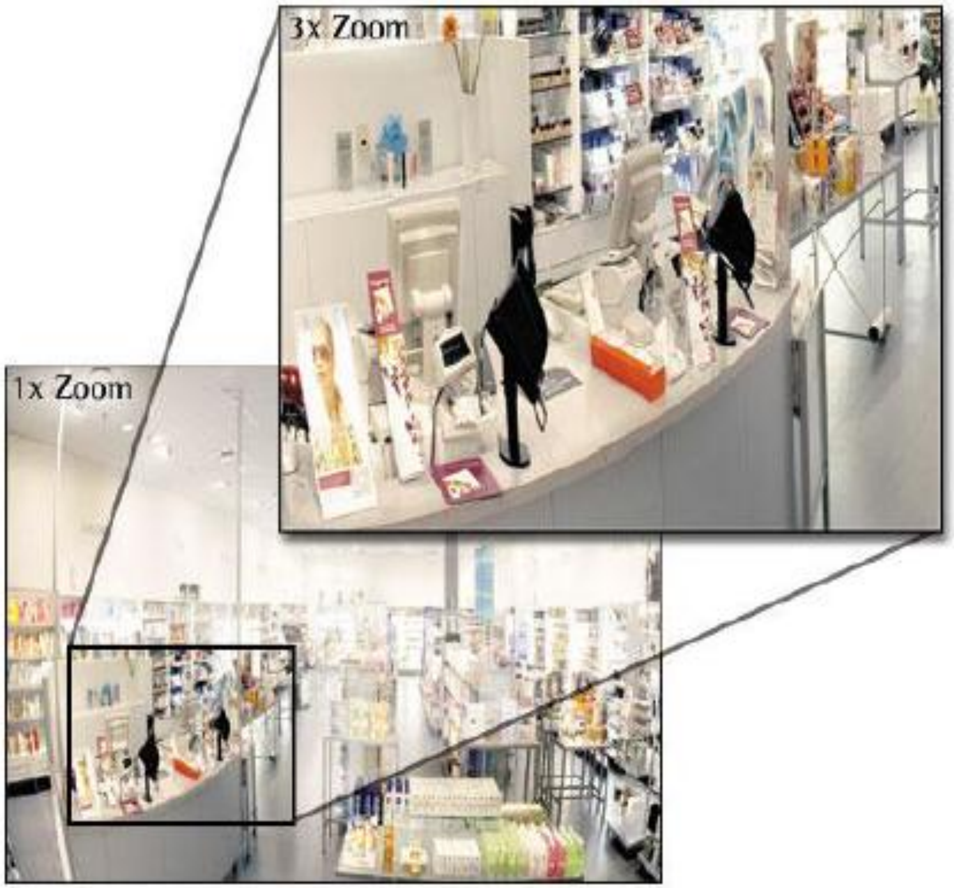


Image Scaling

- Nearest neighbor Interpolation

  - Suppose that we have an image of size 500 x 500 and we want to enlarge it to 1.5 times 750 x 750 pixels.

  - For any zooming approach we have to create an imaginary grid of the size which is required over the original image. In that case we will have an imaginary grid of 750 x 750 over an original image.

  - Obviously the spacing in the grid would be less than one pixel because we fitting it over a smaller image. In order to perform gray level assignment for any point in the overlay, we look for the closest pixel in the original image and assign its gray level to new pixel in the grid.

  - When finished with all points in the grid, we can simply expand it to the originally specified size to obtain the zoomed image. This method of gray level assignment is called nearest neighbor interpolation

Image Scaling

- **Pixel Replication**

  - Pixel replication is applicable when we want to increase the size of an image an integer number of times.

  - For example to double the size of an image we can duplicate each column, this doubles the size of image in horizontal direction. Then we duplicate each row of the enlarged image to double the size in the vertical direction

  - The same procedure can be applied to enlarge the image by any integer number of times (triple, quadruple and so on)

  - The gray level assignment of each pixel is predetermined by the fact that new locations are exact duplicate of old locations

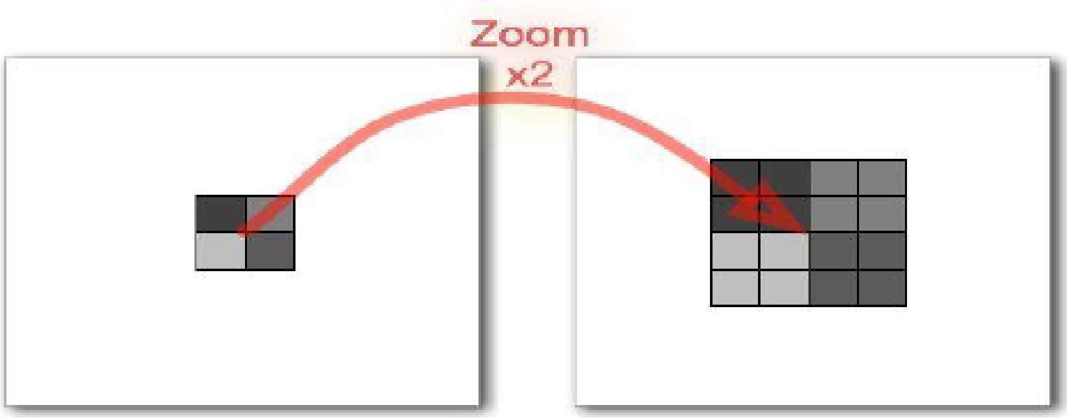


Image Scaling

- **Bilinear Interpolation**
  - Bilinear interpolation considers the closest 2x2 neighborhood of known pixel values surrounding the unknown pixel.
  - It then takes a weighted average of these 4 pixels to arrive at its final interpolated value. This results in much smoother looking images than nearest neighbor.
  - The diagram below is for a case when all known pixel distances are equal, so the interpolated value is simply their sum divided by four.
  - In case the distance varies then The closer pixels are given more weightage in the calculation

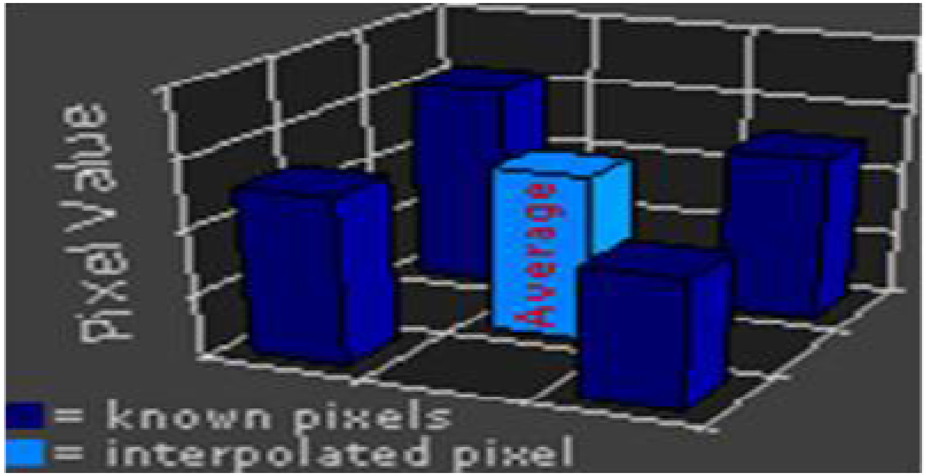


Image Scaling

- The key idea is to perform linear interpolation first in one direction, and then again in the other direction.

- Suppose that we want to find the value of the unknown function f at the point $P = (x, y)$.

- It is assumed that we know the value of f at the four points $Q11 = (x1, y1)$, $Q12 = (x1, y2)$, $Q21 = (x2, y1)$, and $Q22 = (x2, y2)$.

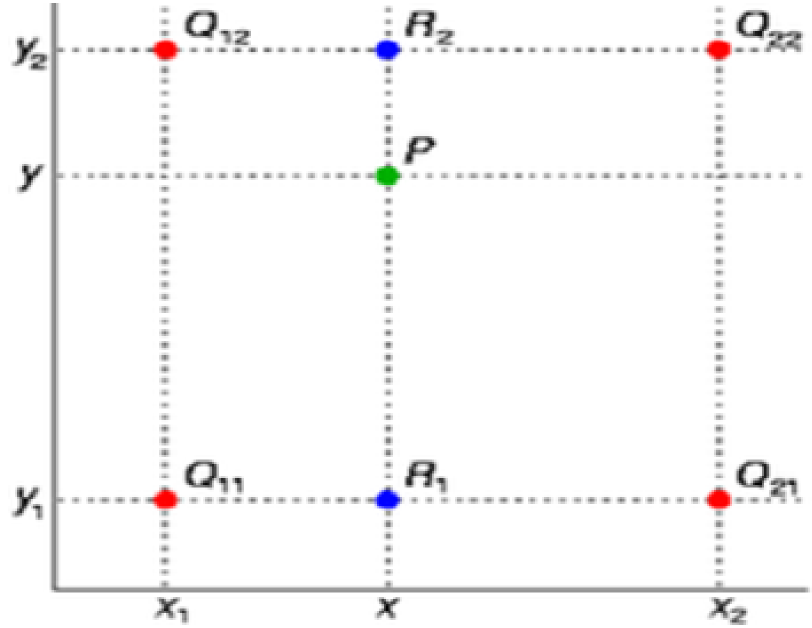


Image Scaling

We first do linear interpolation in the x-direction. This yields

$$f(R_1) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{11}) + \frac{x - x_1}{x_2 - x_1} f(Q_{21})$$

where $R1 = (x,y1)$,

$$f(R_2) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{12}) + \frac{x - x_1}{x_2 - x_1} f(Q_{22})$$

where $R2 = (x,y2)$.

$$f(P) \approx \frac{y_2 - y}{y_2 - y_1} f(R_1) + \frac{y - y_1}{y_2 - y_1} f(R_2).$$

We proceed by interpolating in the y-direction.

Image Scaling

This gives us the desired estimate of $f(x, y)$.

$$
\begin{aligned}
f(x,y) \approx\ & \frac{f(Q_{11})}{(x_2-x_1)(y_2-y_1)}(x_2-x)(y_2-y) \\
&+ \frac{f(Q_{21})}{(x_2-x_1)(y_2-y_1)}(x-x_1)(y_2-y) \\
&+ \frac{f(Q_{12})}{(x_2-x_1)(y_2-y_1)}(x_2-x)(y-y_1) \\
&+ \frac{f(Q_{22})}{(x_2-x_1)(y_2-y_1)}(x-x_1)(y-y_1).
\end{aligned}
$$

Image Scaling

- **Bicubic Interpolation**

  - Bicubic goes one step beyond bilinear by considering the closest 4x4 neighborhood of known pixels-- for a total of 16 pixels.
  - Since these are at various distances from the unknown pixel, closer pixels are given a higher weighting in the calculation.
  - Bicubic produces noticeably sharper images than the previous two methods, and is perhaps the ideal combination of processing time and output quality
  - For this reason it is a standard in many image editing programs (including Adobe Photoshop), printer drivers and in-camera interpolation

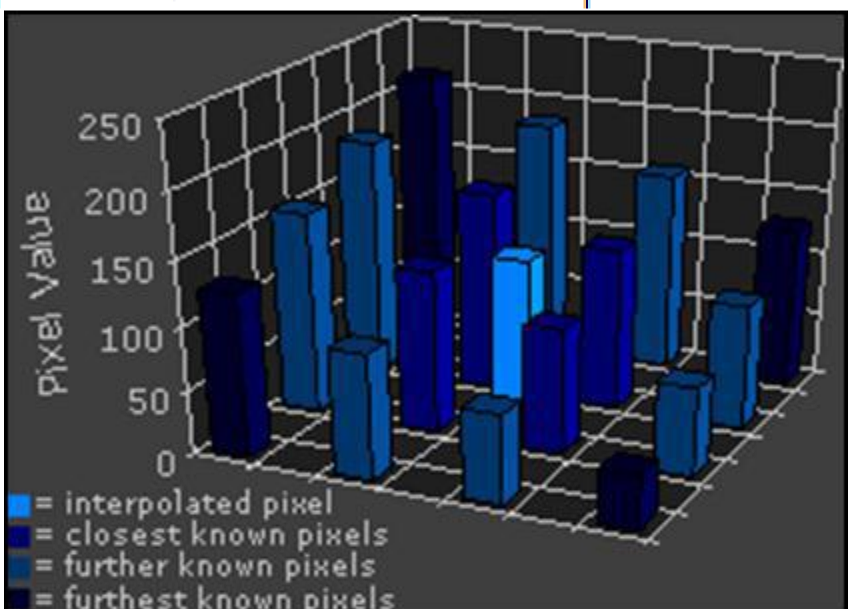


Image Scaling

- Shrinking (Down scaling, resizing downward)
  - Image shrinking is done in the similar manner as zooming with one difference as now the process of pixel replication is row column deletion. Now we can delete every second column and row for shrinking

Distance Measures

- For pixels p, q and **z** with coordinates (x, y), (s, t), and (v, w), respectively, D is a distance function or metric if

  - ✓ D (p, q) >= 0 (D (p, q) = 0 if p=q)
  - ✓ D (p, q) = D (q, p)
  - ✓ D (p, **z**) <= (D (p, q) + D (q, **z**)

- Three different ways to calculate distance depending upon the traversing criteria of pixels are::

  - ✓ Euclidean distance
  - ✓ City-block distance or D4 distance.
  - ✓ D8 distance or chessboard distance.

Distance Measures

- Euclidean distance

  The Euclidean distance between p and q is defined as

$$D_e(p, q) = [(x-s)^2 + (y-t)^2]^{1/2}$$

- City-block distance or D4 distance or Manhaten distance

  The City-block distance between p and q is defined as

$$D4(p, q) = | x - s | + | y - t |$$

- The pixels having a distance D4 from (x, y) less than or equal to some value r from a diamond centered at (x, y). E.g the pixels with D4 distance <=2 from (x, y) (the center point) form the following contours of constant distance

		2		
	2	1	2	
2	1	0	1	2
	2	1	2	
		2		

Distance Measures

- D8 distance or chessboard distance.

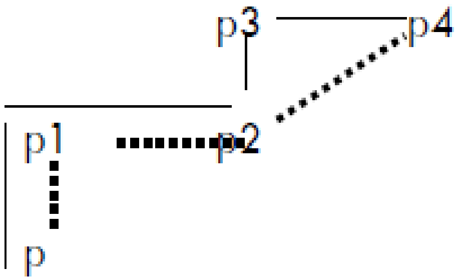
$$D8(p, q) = \max(|x - s|, |y - t|)$$

- ✓ The pixels having a D8 distance from (x, y) less than or equal to some value r from a squared centered at (x, y). E.g the pixels with D8 distance <=2 from (x, y) (the center point) form the following contours of constant distance
- ✓ The pixels with D8 = 1 are the 8-neighbors of (x, y)

2	2	2	2	2
2	1	1	1	2
2	1	0	1	2
2	1	1	1	2
2	2	2	2	2

Distance Measures

- Consider the following arrangement of pixels and assume that p,p2 and p4 have value 1 and p1 and p3 have value 0 or 1:



- Suppose we have V={1} [Adjacency criteria]

- If p1 and p3 are zero, the length of the shortest m-path (Dm distance) between p and p4 is 2

- If p1=1 then p2 and p will no longer be m-adjacent and the length of the shortest m-path becomes 3 [Path will be p p1 p2 p4]

- If p3=1 and p1=0 then the length of the shortest m-path will also be 3

- Finally if both p1 and p3 are 1 then the length of the shortest m-path between p and p4 is 4 [Path will be p p1 p2 p3 p4]