1. SPATIAL AND INTENSITY RESOLUTION

A widely used definition of image resolution is the largest number of discernible line pairs per unit distance (e.g., 100 line pairs per mm). Dots per unit distance is a measure of image resolution used in the printing and publishing industry. In the U.S., this measure usually is expressed as dots per inch (dpi). To give you an idea of quality, newspapers are printed with a resolution of 75 dpi, magazines at 133 dpi, glossy brochures at 175 dpi.

To be meaningful, measures of spatial resolution must be stated with respect to spatial units. Image size by itself does not tell the complete story. For example, to say that an image has a resolution of 1024×1024 pixels is not a meaningful statement without stating the spatial dimensions encompassed by the image. Size by itself is helpful only in making comparisons between imaging capabilities. For instance, a digital camera with a 20-megapixel CCD imaging chip can be expected to have a higher capability to resolve detail than an 8-megapixel camera, assuming that both cameras are equipped with comparable lenses and the comparison images are taken at the same distance.

Intensity resolution similarly refers to the smallest discernible change in intensity level. We have considerable discretion regarding the number of spatial samples (pixels) used to generate a digital image, but this is not true regarding the number of intensity levels. Based on hardware considerations, the number of intensity levels usually is an integer power of two, as we mentioned when discussing Eq. (2-11). The most common number is 8 bits, with 16 bits being used in some applications in which enhancement of specific intensity ranges is necessary. Intensity quantization using 32 bits is rare. Sometimes one finds systems that can digitize the intensity levels of an image using 10 or 12 bits, but these are not as common. Intensity resolution similarly refers to the smallest discernible change in intensity level. We have considerable discretion regarding the number of spatial samples (pixels) used to generate a digital image, but this is not true regarding the number of intensity levels. Based on hardware considerations, the number of intensity levels usually is an integer power of two. The most common number is 8 bits (number of intensity levels $= 2^8 = 256$), with 16 bits being used in some applications in which enhancement of specific intensity ranges is necessary. Intensity quantization using 32 bits is rare. Sometimes one finds systems that can digitize the intensity levels of an image using 10 or 12 bits, but these are not as common.

Unlike spatial resolution, which must be based on a per-unit-of-distance basis to be meaningful, it is common practice to refer to the number of bits used to quantize intensity as the “intensity resolution.” For example, it is common to say that an image whose intensity is quantized into 256 levels has 8 bits of intensity resolution. However, keep in mind that discernible changes in intensity are influenced also by

noise and saturation values, and by the capabilities of human perception to analyze and interpret details in the context of an entire scene. Figures 1 and 2 shows the effect of changing the spatial and the intensity resolutions.

Figure 1: Effects of reducing spatial resolution. The images shown are at: (a) 930 dpi, (b) 300 dpi, (c) 150 dpi, and (d) 72 dpi.

2. IMAGE INTERPOLATION

Interpolation is used in tasks such as zooming, shrinking, rotating, and geometrically correcting digital images. Our principal objective in this section is to introduce interpolation and apply it to image resizing (shrinking and zooming), which are basically image resampling methods.

Interpolation is the process of using known data to estimate values at unknown locations. We begin the discussion of this topic with a short example. Suppose that an image of size 500×500 pixels has to be enlarged 1.5 times to 750×750 pixels. A simple way to visualize zooming is to create an imaginary 750×750 grid with the same pixel spacing as the original image, then shrink it so that it exactly overlays the original image. Obviously, the pixel spacing in the shrunken 750×750 grid will be less than the pixel spacing in the original image. To assign an intensity value to any point in the overlay, we look for its closest

pixel in the underlying original image and assign the intensity of that pixel to the new pixel in the 750×750 grid. When intensities have been assigned to all the points in the overlay grid, we expand it back to the specified size to obtain the resized image.

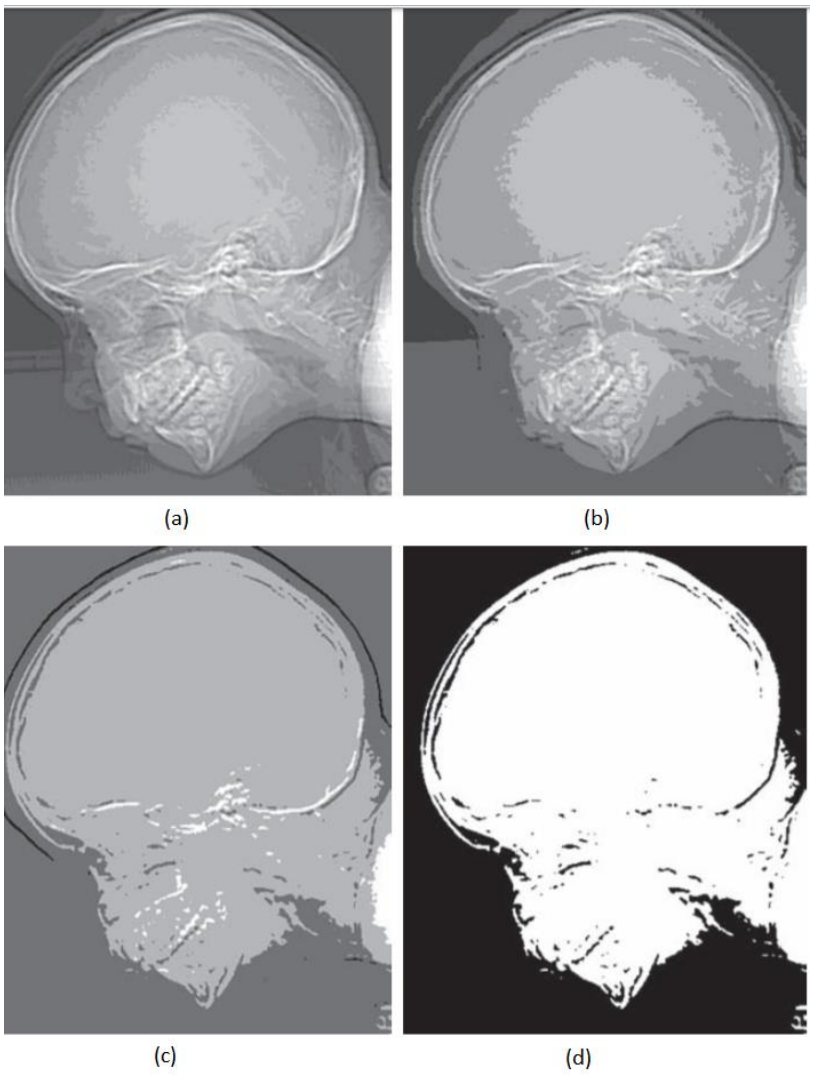

Figure 2: (a)-(d) Image displayed in 16, 8, 4, and 2 intensity levels.

The method just discussed is called *nearest neighbor interpolation* because it assigns to each new location the intensity of its nearest neighbor in the original image. This approach is simple but, it has the tendency to produce undesirable artifacts, such as severe distortion of straight edges. A more suitable approach is *bilinear interpolation*, in which we use the four nearest neighbors to estimate the intensity at a given location. Let (x, y) denote the coordinates of the location to which we want to assign an intensity value (think of it as a point of the grid described previously), and let $v(x, y)$ denote that intensity value. For bilinear interpolation, the assigned value is obtained using the equation

$$v(x, y) = ax + by + cxy + d$$

where the four coefficients are determined from the four equations in four unknowns that can be written using the *four* nearest neighbors of point (x, y). Bilinear interpolation gives much better results than nearest neighbor interpolation, with a modest increase in computational burden.

The next level of complexity is *bicubic interpolation*, which involves the sixteen nearest neighbors of a point. The intensity value assigned to point (x, y) is obtained using the equation

$$v(x, y) = \sum_{i=0}^{3} \sum_{j=0}^{3} a_{ij} x^i y^j$$

The sixteen coefficients are determined from the sixteen equations with sixteen unknowns that can be written using the sixteen nearest neighbors of point (x, y). Generally, bicubic interpolation does a better job of preserving fine detail than its bilinear counterpart. Bicubic interpolation is the standard used in commercial image editing applications, such as Adobe Photoshop and Corel Photopaint.

Although images are displayed with integer coordinates, it is possible during processing to work with subpixel accuracy by increasing the size of the image using interpolation to "fill the gaps" between pixels in the original image. Figure 3 shows the results of applying the three algorithms on the same image.

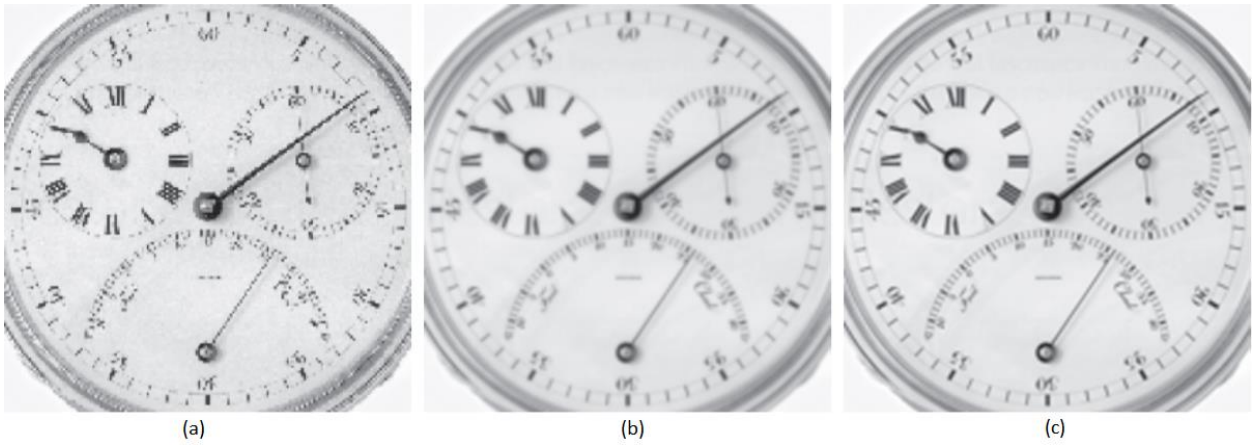

Figure 3: (a) Image reduced to 72 dpi and zoomed back to its original 930 dpi using nearest neighbor interpolation. (b) Image reduced to 72 dpi and zoomed using bilinear interpolation. (c) Same as (b) but using bicubic interpolation.

3. SOME BASIC RELATIONSHIPS BETWEEN PIXELS

In this section, we discuss several important relationships between pixels in a digital image. When referring in the following discussion to particular pixels, we use lowercase letters, such as p and q.

NEIGHBORS OF A PIXEL

A pixel p at coordinates (x, y) has two horizontal and two vertical neighbors with coordinates

$$(x+1, y), (x-1, y), (x, y+1), (x, y-1)$$

This set of pixels, called the 4-*neighbors* of p, is denoted $N_4(p)$.

The four *diagonal* neighbors of p have coordinates

$$(x+1, y+1), (x+1, y-1), (x-1, y+1), (x-1, y-1)$$

and are denoted $N_D(p)$. These neighbors, together with the 4-neighbors, are called the 8-*neighbors* of p, denoted by $N_8(p)$. The set of image locations of the neighbors of a point p is called the *neighborhood* of p. The neighborhood is said to be *closed* if it contains p. Otherwise, the neighborhood is said to be *open*.

ADJACENCY, CONNECTIVITY, REGIONS, AND BOUNDARIES

Let V be the set of intensity values used to define adjacency. In a binary image, $V = \{1\}$ if we are referring to adjacency of pixels with value 1. In a grayscale image, the idea is the same, but set V typically contains more elements. For example, if we are dealing with the adjacency of pixels whose values are in the range 0 to 255, set V could be any subset of these 256 values. We consider three types of adjacency:

1) 4-adjacency. Two pixels p and q with values from V are 4-adjacent if q is in the set $N_4(p)$.

2) 8-adjacency. Two pixels p and q with values from V are 8-adjacent if q is in the set $N_8(p)$.

3) m-adjacency (also called mixed adjacency). Two pixels p and q with values from V are m-adjacent if

a) q is in $N_4(p)$, or

b) q is in $N_D(p)$ *and* the set $N_4(p) \cap N_4(q)$ has no pixels whose values are from V.

Mixed adjacency is a modification of 8-adjacency, and is introduced to eliminate the ambiguities that may result from using 8-adjacency. For example, consider the pixel arrangement in Fig. 4(a) and let $V = \{1\}$. The three pixels at the top of Fig. 4(b) show multiple (ambiguous) 8-adjacency, as indicated by the dashed lines. This ambiguity is removed by using m-adjacency, as in Fig. 4(c). In other words, the center and upper-right diagonal pixels are not m-adjacent because they do not satisfy condition (b).

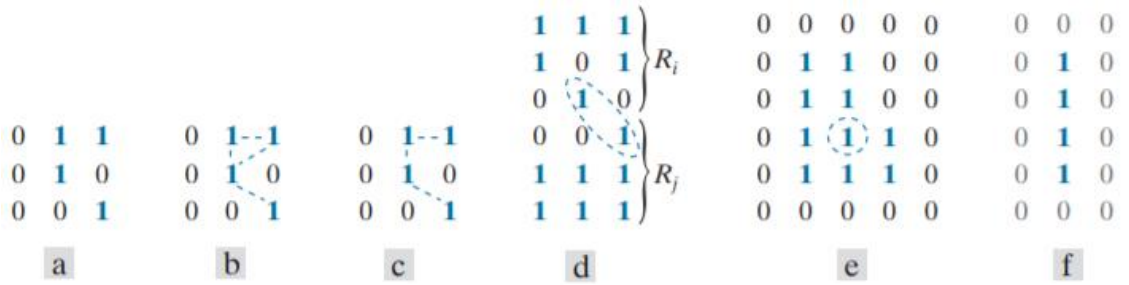

Figure 4: (a) An arrangement of pixels. (b) Pixels that are 8-adjacent (adjacency is shown by dashed lines). (c) m-adjacency. (d) Two regions (of 1's) that are 8-adjacent. (e) The circled point is on the boundary of the 1-valued pixels only if 8-adjacency between the region and background is used. (f) The inner boundary of the 1-valued region does not form a closed path, but its outer boundary does.

A *digital path* (or *curve*) from pixel p with coordinates (x_0, y_0) to pixel q with coordinates (x_n, y_n) is a sequence of distinct pixels with coordinates

$$(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)$$

where points (x_i, y_i) and (x_{i-1}, y_{i-1}) are adjacent for $1 \leq i \leq n$. In this case, n is the length of the path. If $(x_0, y_0) = (x_n, y_n)$ the path is a closed path. We can define 4-, 8-, or m-paths, depending on the type of adjacency specified. For example, the paths in Fig. 4(b) between the top right and bottom right points are 8-paths, and the path in Fig. 4(c) is an m-path.

Let S represent a subset of pixels in an image. Two pixels p and q are said to be connected in S if there exists a path between them consisting entirely of pixels in S. For any pixel p in S, the set of pixels that are connected to it in S is called a connected component of S. If it only has one component, and that component is connected, then S is called a connected set.

Let R represent a subset of pixels in an image. We call R a region of the image if R is a connected set. Two regions, R_i and R_j are said to be adjacent if their union forms a connected set. Regions that are not adjacent are said to be disjoint. We consider 4- and 8-adjacency when referring to regions. For our definition to make sense, the type of adjacency used must be specified. For example, the two regions of 1's in Fig. 4(d) are adjacent only if 8-adjacency is used.

Suppose an image contains K disjoint regions, $R_k, k = 1, \dots, K$ none of which touches the image border[1]. Let R_{u} denote the union of all the K regions, and let $(R_{\mathrm{u}})^c$ denote its complement (recall that the

[1] We make this assumption to avoid having to deal with special cases. This can be done without loss of generality because if one or more regions touch the border of an image, we can simply pad the image with a 1-pixel-wide border of background values.

complement of a set A is the set of points that are not in A). We call all the points in R_u the foreground, and all the points in $(R_u)^c$ the background of the image.

The *boundary* (also called the *border* or *contour*) of a region R is the set of pixels in R that are adjacent to pixels in the complement of R. Stated another way, the border of a region is the set of pixels in the region that have at least one background neighbor. Here again, we must specify the connectivity being used to define adjacency. For example, the point circled in Fig. 4(e) is not a member of the border of the 1-valued region if 4-connectivity is used between the region and its background, because the only possible connection between that point and the background is diagonal. As a rule, adjacency between points in a region and its background is defined using 8-connectivity to handle situations such as this.

The preceding definition sometimes is referred to as the *inner border* of the region to distinguish it from its *outer border*, which is the corresponding border in the background. This distinction is important in the development of border-following algorithms. Such algorithms usually are formulated to follow the outer boundary in order to guarantee that the result will form a closed path. For instance, the inner border of the 1-valued region in Fig. 4(f) is the region itself. This border does not satisfy the definition of a closed path. On the other hand, the outer border of the region does form a closed path around the region.

If R happens to be an entire image, then its *boundary* (or *border*) is defined as the set of pixels in the first and last rows and columns of the image. This extra definition is required because an image has no neighbors beyond its border. Normally, when we refer to a region, we are referring to a subset of an image, and any pixels in the boundary of the region that happen to coincide with the border of the image are included implicitly as part of the region boundary.

The concept of an edge is found frequently in discussions dealing with regions and boundaries. However, there is a key difference between these two concepts. The boundary of a finite region forms a closed path and is thus a "global" concept. Edges are formed from pixels with derivative values that exceed a preset threshold. Thus, an edge is a "local" concept that is based on a measure of intensity-level discontinuity at a point. It is possible to link edge points into edge segments, and sometimes these segments are linked in such a way that they correspond to boundaries, but this is not always the case. The one exception in which edges and boundaries correspond is in binary images. Depending on the type of connectivity and edge operators used, the edge extracted from a binary region will be the same as the region boundary. This is intuitive. Conceptually, until we arrive at Chapter 10, it is helpful to think of edges as intensity discontinuities, and of boundaries as closed paths.

4. GRAPHICS FILE FORMAT

A graphics file format is the format in which graphics data--data describing a graphics image--is stored in a file. Graphics file formats have come about from the need to store, organize, and retrieve graphics data in an efficient and logical way.

Traditionally, graphics refers to the production of a visual representation of a real or imaginary object created by methods known to graphic artists, such as writing, painting, imprinting, and etching. The final result of the traditional graphics production process eventually appears on a 2D surface, such as paper or canvas. Computer graphics, however, has expanded the meaning of graphics to include any data intended for display on an output device, such as a screen, printer, plotter, film recorder, or videotape.

There are a number of different types of graphics file formats. Each type stores graphics data in a different way. Bitmap, vector, and metafile formats are by far the most commonly used formats, and we focus on these.

BITMAP FORMATS

Bitmap formats are used to store bitmap data. Files of this type are particularly well-suited for the storage of real-world images such as photographs and video images. Bitmap files, sometimes called raster files, essentially contain an exact pixel-by-pixel map of an image. A rendering application can subsequently reconstruct this image on the display surface of an output device. Microsoft BMP, PCX, TIFF, and TGA are examples of commonly used bitmap formats.

VECTOR FORMATS

Vector format files are particularly useful for storing line-based elements, such as lines and polygons, or those that can be decomposed into simple geometric objects, such as text. Vector files contain mathematical descriptions of image elements, rather than pixel values. A rendering application uses these mathematical descriptions of graphical shapes (e.g., lines, curves, and splines) to construct a final image. AutoCAD DXF and Microsoft SYLK are examples of commonly used vector formats.

METAFILE FORMATS

Metafiles can contain both bitmap and vector data in a single file. The simplest metafiles resemble vector format files; they provide a language or grammar that may be used to define vector data elements, but they may also store a bitmap representation of an image. Metafiles are frequently used to transport

bitmap or vector data between hardware platforms, or to move image data between software platforms. WPG, Macintosh PICT, and CGM are examples of commonly used metafile formats.

COMPRESSING DATA

Compression is the process used to reduce the physical size of a block of information. By compressing graphics data, we're able to fit more information in a physical storage space. Because graphics images usually require a very large amount of storage space, compression is an important consideration for graphics file formats. Almost every graphics file format uses some compression method.

BASIC CONCEPTS

1. Pixel Depth

Pixel depth is the number of bits used to encode the information of each pixel. Every image is stored in a file and kept in the memory of a computer as group of bytes. Bytes are group of 8 bits and represent the smallest quantity that can be stored in the memory of a computer. This means that if a 256 by256 pixels image has a pixel depth of 12 or 16 bits, the computer will always store two bytes per pixel and then the pixel data will require 256×256×2=131,072 bytes of memory in both cases.

2. Photometric Interpretation

The photometric interpretation specifies how the pixel data should be interpreted for the correct image display as a monochrome or color image. To specify if color information is or is not stored in the image pixel values, we introduce the concept of samples per pixel (also known as number of channels). Monochrome images have one sample per pixel and no color information stored in the image. A scale of shades of gray from black to white is used to display the images. The number of shades of gray depends clearly from the number of bits used to store the sample that, in this case, coincide with the pixel depth.

3. Metadata

Meta data are information that describe the image. It can seem strange, but in any file format, there is always information associated with the image beyond the pixel data. This information called metadata is typically stored at the beginning of the file as a header and contains at least the image matrix dimensions, the spatial resolution, the pixel depth, and the photometric interpretation. Thanks to metadata, a software application is able to recognize and correctly open an image in a supported file format simply by a double-click or dragging the image icon onto the icon of the application.

4. Pixel Data

This is the section where the numerical values of the pixels are stored. According to the data type, pixel data are stored as integers or floating-point numbers using the minimum number of bytes required to represent the values

References and further reading:

Digital Image Processing, 4th edition, Gonzalez, Rafael and Woods, Richard, 2018

The Encyclopedia of Graphics File Formats. http://www.fileformat.info/mirror/egff/ (accessed in November 2019)

Larobina, M., & Murino, L. (2014). Medical image file formats. Journal of digital imaging, 27(2), 200-206.