What Is Image Processing?

The term *image processing* has become one of today's hottest key words in the applied computer sciences. What was once an expensive, time-consuming, and somewhat unpredictable endeavor has ripened into a mature discipline of its own. With the advent of inexpensive microprocessors, dense memory devices, and special purpose signal processing components, image processing has become a valuable tool in a variety of applications.

Image processing, in its general form, pertains to the alteration and analysis of pictorial information. We find instances of image processing occurring all the time in our daily lives. Perhaps the most common case is that of eyeglasses. Corrective eyeglasses serve to alter observed pictorial scenes in such a way that aberrations created by the eye are compensated for by correcting the image before its contact with the eye. Another common case of image processing is the adjustment of the brightness and contrast controls on a television set. By doing this, we enhance the image until its subjective appearance to us is most appealing. Even the water in a pond serves to alter the form of an image. The reflected image is not only reversed, but often exhibits distortion due to the water's motion. Probably the most powerful image processing system encountered in everyday life is the one comprised of the human eye and brain. This biological system receives, en-

hances, dissects, analyzes, and stores images at enormous rates of speed. Ironically, this system is taken more for granted than any other. All these are examples of image processing that are so commonly accepted that one rarely thinks of them as anything unique.

Methods of Image Processing

In the pursuit of image processing as a discipline, the objective is to visually enhance or statistically evaluate some aspect of an image not readily apparent in its original form. This objective is carried out through the development and implementation of the processing means necessary to operate upon images. Fundamentally, three techniques of implementing a process upon an image are available—one that is optical and two that are electronic, analog and digital. Although the analog and digital techniques are both electronic means, they differ considerably. Each of the three methods is found in routine use with the particular application defining the most practical approach to implementing the process in need.

Optical processing, as implied, uses an arrangement of optics to carry out a process. Eyeglasses are a form of optical image processing. An important form of optical processing is found in the photographic darkroom. For years, photographers have enhanced, manipulated, and abstracted images from one form to another, the object always being to produce a more favorable or appealing final print. This classical form of image processing has been refined through trial-and-error techniques, leaving today's photographer with a broad base of rules enabling quick and predictable results. The pioneers of the darkroom may probably be considered to be the first to use defined image processing techniques in their everyday work.

Analog processing of images refers to the alteration of images through electrical means. Of course, the image must be in an electrical form first. The most common example of this is the television image. The television signal is a voltage level that varies in amplitude to represent brightness throughout the image. By electrically altering this signal, we correspondingly alter the final displayed image appearance. The brightness and contrast controls on a television set serve to adjust the amplitude and reference of the video signal, resulting in the brightening, darkening, and alteration of the brightness range of the displayed image.

Digital image processing is a form of image processing brought on by the advent of the digital computer. Allowing the precise implementation of processes, this form provides the greatest flexibility and power for general image processing applications. Within the digital domain, an image is represented by discrete points of defined brightness. Each point has a numeric location within the image and a numeric brightness. By manipulating these

values of brightness within the image, the computer is capable of carrying out the most complex operations with relative ease. Furthermore, the flexibility in the programming of a computer allows operations to be modified quickly, a feature that optical and analog processing inherently do not support.

The recent availability of sophisticated semiconductor digital devices and compact powerful computers, coupled with advances in image processing algorithms, has brought digital image processing to the forefront. Because of this, a new industry has been born. The prime products of this industry are computer hardware, software, and special peripherals developed to support the needs of digital image handling and processing. This intense activity has led to the development of digital image processing systems suitable in price and capability to be used in low-end applications previously denied the opportunity.

Digital Image Processing: A Historical Evolution

The roots of digital image processing may be traced back to the early 1960s. It was at this time that NASA was energetically pursuing its lunar science program in an attempt to characterize the lunar surface in support of the future Apollo program. The Ranger program was established, in part, to image the lunar surface, relaying the pictures to Earthbound scientists for evaluation. After several previous Ranger missions during which the video equipment failed to function, Ranger 7 transmitted several thousand images back to Earth. These television images were taken from their original analog electronic form and converted to a digital form. Subsequent digital processing of these image data was then carried out to remove various camera geometric and response distortions. It was this processing of Ranger 7 imagery that ushered the digital computer into the world of image processing.

This initial work in digital image processing was done at NASA's Jet Propulsion Laboratory in Pasadena, California. NASA then continued this funding of research and development in support of its other space programs. Following the Ranger program was a series of planetary exploration probes, all supported by digital image processing. The Mariner project returned images from the planets Mars, Venus, and Mercury. Project Surveyor soft-landed cameras on the lunar surface. Pioneer 10 and 11 spacecraft sent fly-by images of Jupiter and Saturn. The Viking spacecraft, equipped with cameras, landed on the surface of Mars. More recently, two Voyager spacecraft encountered the planets Jupiter and Saturn, and returned a wide range of imagery aiding in the scientific studies of these planets.

In addition to NASA's planetary imagery, various other government agencies such as the United States Geological Survey support various image processing activities.

Earth-orbiting satellites such as LANDSAT, TIROS, NIMBUS, GOES, and a variety of military surveillance systems return electronic Earth surface imagery on a day-to-day basis. Furthermore, their data are routinely processed at ground receiving stations through various digital computer systems prior to their use.

Good background on the origin and evolution of digital image processing is found in Further Reading/References I-3 and II-1.

Although the space program provided the initial impetus and funding for the research and development of image processing, the applications are not restricted to space imagery. Today, image processing is found in medical, factory automation, and robotics control applications. The ever-declining price and increasing availability of digital systems for image acquisition and handling has brought high-power processing capabilities to the user who once only dreamed of such possibilities. The microcomputer revolution has allowed the consumer to pursue various low-level computing activities. One such boom in the industry is in computer graphics. Graphics may be thought of as the synthetic generation of pictorial imagery. Of course, the next logical step is to provide real-life imaging capabilities. The computer is truly gaining the ability to see, making vision and image generation the next man-machine interface.

One such application of a low-level imagery project is that of the amateur radio satellite, OSCAR 9. This spacecraft, built by the University of Surrey, England, is to support a variety of scientific studies including Earth imaging. Once the satellite becomes operational, images are to be transmitted on standard amateur radio bands, allowing virtually anyone to receive them. With low-level image handling capabilities, the ham will be able to receive and display these images in the comfort of his or her own home. By the same token, simple image handling capabilities also allow images to be transmitted from one individual to another over standard telephone lines or on cassette tapes. These endeavors, combined with industry support, bring digital image processing to the level where average individuals may pursue it.

In the following pages, the field of digital image data handling and processing is introduced in a manner comprehensible to the interested individual. Image processing is introduced, overviewing common techniques and implementations. Additionally, the basic electronic hardware is discussed, giving a block-level idea of the methods employed in handling and processing image data. The attempt made here is to supply a practical introduction and reference to the field of digital image processing.

Digital Image Processing: The Basics

In this chapter, we explore the fundamental elements of digital image processing. First, the field of image processing is discussed in an operational context, where the various reasons for using these techniques are laid out. Following this, the processes for carrying out these operations are defined. Finally, the hardware system for implementing the processes is overviewed. The purpose of this chapter is to briefly introduce these subjects and serve as a foundation for their use throughout this book.

Image processing is a field that encompasses a broad range of capabilities. Any action that operates upon or uses pictorial information falls within the discipline of image processing. Two terms that will help in the comprehension of the field must be defined. An *image operation* is any action upon an image that is defined from an application's standpoint. When we speak of a particular operation, we are explaining what the desired result is to be. An *image process*, on the other hand, defines how a given operation is to be implemented. A process is a means of carrying out an operation. These are two distinctly separate concepts, analogous in many ways to the differences between computer software and hardware. We may now discuss image processing from both the operational and processing viewpoint.

Operational Breakdown

Three classes of image operations may be used to suitably subdivide the field into manageable parts (see Figure 2-1). They are: (1) *image quality enhancement*—operations that subjectively or objectively modify the appearance, or qualities, of an image; (2) *image analysis*—operations that produce numeric information based on an image; and (3) *image coding*—operations that code an image into a new form. By breaking down operations into these categories, the study of image processing becomes more structured in an applications sense. By exploring these three operational areas of image processing, we set forth the groundwork for studying the application and implementation of digital image processing.

IMAGE QUALITY ENHANCEMENT

Image quality enhancement operations serve to enhance or in some way alter the qualities of an image. The desired result is an image of improved quality. The improvement in the quality of an image is often subjective and is related to the application as well as the judgment of the viewer. For instance, one application may require the sharpening of an image that appears blurred. Another application, however, may defocus an image so that sharp details are eliminated, making other features of the image more detectable. The applications are different, making each viewer's operation appear contrary to the other's interest. In short, one viewer's enhancement is another's degradation.

The product of a quality enhancement operation is an output image yielding a changed version of the original. These enhancements may be applied to images that are in some way degraded from what would be considered a "good" image. Alternatively, "good" images may be enhanced to produce output images that show certain features enhanced for easier viewing. In either case, the qualities of the original image are altered in some way, affecting the image's appearance to the viewer.

Image quality enhancements may be either subjective or objective. Subjective enhancements are used to make an image more visually appealing and may be applied until an image achieves this goal. Objective enhancement, however,

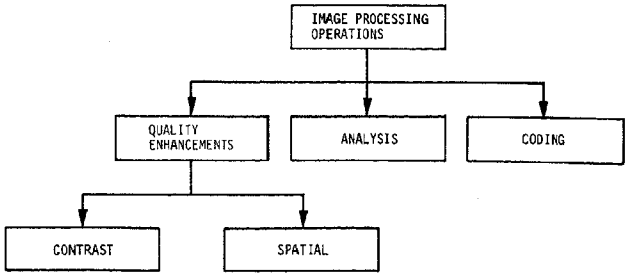


Figure 2-1 Operational breakdown of the image processing field.

corrects an image for known degradations and does not necessarily attempt to make the image more appealing. An example of objective quality enhancement is photometric correction, which will be discussed in Chapter 5. For the most part, we will discuss image quality enhancement in the subjective enhancement context.

Image quality enhancement operations may be conveniently broken into two subclasses, contrast and spatial enhancements. Contrast enhancements deal with the alteration of brightness within an image. Blacks, whites, and grays may be intensified or even suppressed, bringing out facets that were hard to see in the original. Spatial enhancements modify the content of detail within an image. Edges, for instance, may be accentuated, making viewing more appealing. A wide variety of contrast and spatial enhancements allow the viewer flexibility in the total modification or correction of an image.

The Human Visual System. When attempting the improvement of an image's quality, a knowledge of the characteristics of the human visual system is an important prerequisite. All images are ultimately processed by the visual system prior to the viewer's mental perception of them. An understanding of this fact not only helps us to define an image processing operation for maximum effectiveness, but also serves to indicate the limitations of the eye-brain system itself.

The eye is a complex unit that converts visual information into nerve impulses used by the brain to form a perceived image. The major functional components of the eye are illustrated in Figure 2-2. Light rays generated by a scene are collected by the *lens* and projected upon the surface of the *retina*. The *iris* serves to control the amount of light allowed to pass through the lens. The lens and iris are both physically protected by the *cornea*.

The retina is composed of light-sensitive elements known as *rods* and *cones*. On the order of 100 million of these *photoreceptors* serve to translate light intensity to nerve impulses. These impulses travel from the eye to the brain through nerve fibers within the *optic nerve*. The brain, in turn, deciphers the nerve impulse information to form what we perceive as an image.

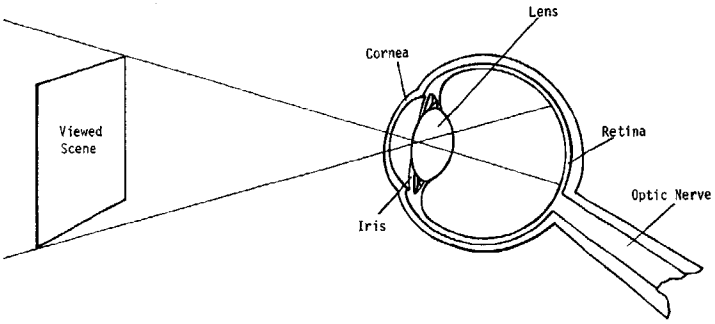


Figure 2-2 The human eye.

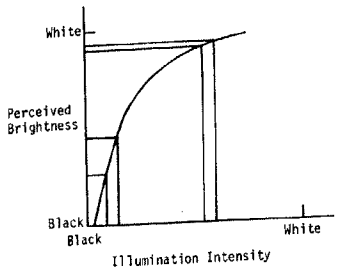


Figure 2-3 Typical logarithmic curve, similar to that of the eye's light intensity response.

What's interesting about the visual system is the way in which the photoreceptors respond to light intensity changes and interact with one another. With an understanding of these characteristics, we may better attack the problem of what constitutes image quality enhancement.

The impulses generated by the photoreceptors are then translated by the brain into perceived brightness. Research in this area of the visual system has shown that the relationship between impinging light upon a receptor and perceived brightness is not a linear function. This means that as the illumination intensity of a viewed object is changed, the viewer will not perceive an equal change in brightness. The actual response is logarithmic, appearing as a curve similar to that shown in Figure 2-3. In the dark regions, a slight illumination increase results in a large increase in perceived brightness. On the other hand, the same slight illumination increase in the bright regions yields a small increase in perceived brightness. The logarithmic response of the eye may be illustrated by Figures 2-4 and 2-5. In Figure 2-4, the brightness is incremented in equal illumination intensity steps from black to white. As we would expect from the graph in Figure 2-3, the dark regions are clustered at the left. The equal steps in the bright regions are virtually undetectable, making the entire right side appear white. Figure 2-5, however, illustrates intensities that are incremented in exponential steps, counteracting the eye's logarithmic response. The net result is a black-to-white transition that is perceived as happening in equal, or linear, steps.

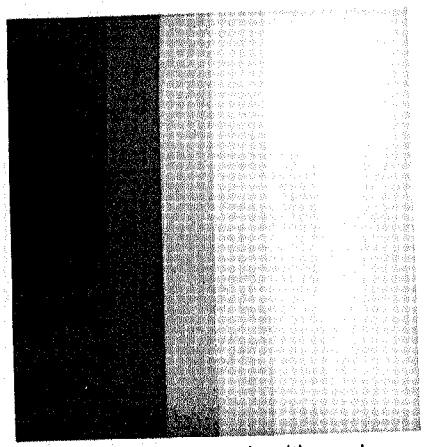


Figure 2-4 Step gray scale with equal intensity steps.

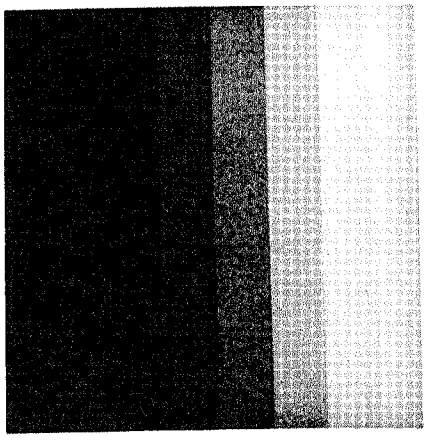


Figure 2-5
Step gray scale with exponential intensity steps (equal perceived brightness steps).

The bottom line of the eye's logarithmic response is that sensitivity in the dark regions of a viewed scene is much greater than that of bright regions. This is because a change of equal intensity is perceived as a greater change in a dark region than in a bright region. This is an important fact to remember, for in the processing of an image, simple darkening of bright regions can bring out previously undetectable detail.

In addition to the logarithmic response characteristic, interactions between photoreceptors cause important visual phenomena to occur. Two, in particular, illustrate the role of these interactions in the visual perception of brightness. One effect, referred to as *simultaneous contrast,* is an illusion where the perceived brightness of a region is

dependent on the intensity of the surrounding area. This effect is shown in Figure 2-6. The two squares are of the same intensity, but the one on the left appears brighter, due to its darker background. (Conversely, the square on the right appears darker because of its lighter background.) The visual system apparently adjusts its brightness response based on the average intensity of the viewed scene. Since the left side has an overall darker average intensity than the right side, the perceived brightness is increased. (Likewise, the perceived brightness of the right side is decreased.) Hence, the difference in the apparent brightnesses of the two squares becomes apparent.

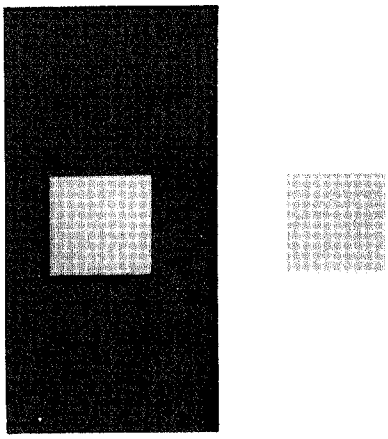
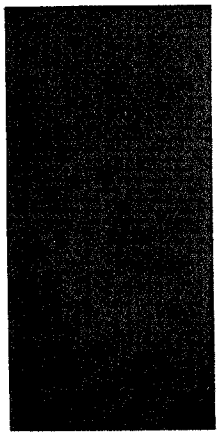


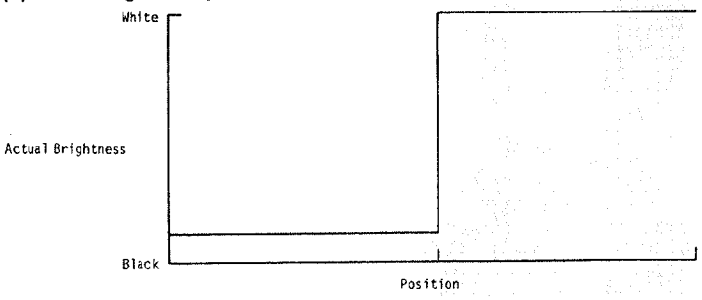
Figure 2-6 Simultaneous contrast —the two squares are the same intensity.

A second phenomenon, known as the *Mach band effect,* causes sharp intensity changes to be accentuated by the visual system. Figure 2-7 shows a sharp black-to-white transition along with the plots of actual brightness and perceived brightness change. The viewer sees a darker bar just to the left of the transition. Similarly, a lighter bar appears just to the right. These under- and overshoots are an artifact of the visual system. In fact, it turns out that without these additions, the transition does not appear nearly as sharp and crisp. The visual system actually adds these edge enhancements, sharpening everything we view. Figure

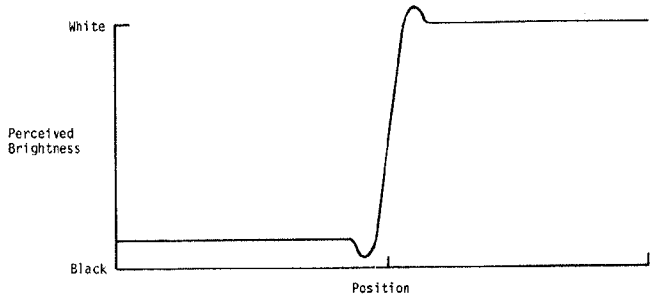
Figure 2-7
(a) Mach band effect —a dark band to the left and bright band to the right of the brightness transition may be seen.



(b) Actual brightness plot.



(c) Perceived brightness plot.



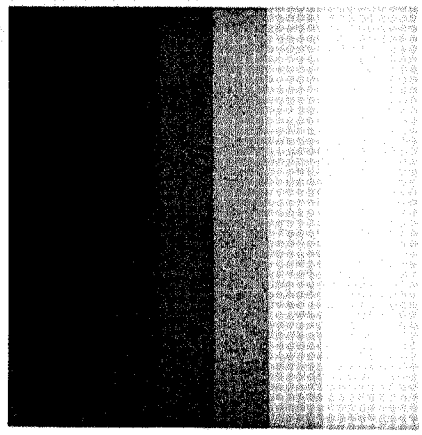
2-8 illustrates this effect upon a stair-step intensity increase.

Intensity response characteristics may be combined with photoreceptor interaction properties. The visual system responds to transitions within a scene depending on the amount of light intensity change present. Slowly varying transitions are detectable even when composed of only small-intensity changes, whereas very minute transitions must contain large-intensity changes before they are seen. This means that highly detailed regions of an image composed of subtle intensity changes may be rendered undetected. Increasing the intensity change, or contrast, can make these details visible. The visual system is most responsive to scene details of high contrast.

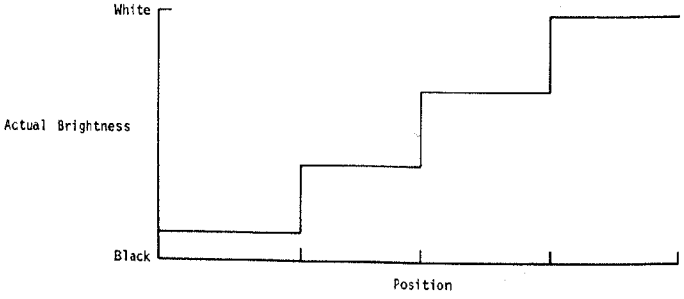
These visual phenomena indicate complex processes occurring within the human visual system. By using knowledge of the system's response and interactive characteristics, we may better apply image processing operations, yielding more natural contrast and spatial enhancements.

Contrast Degradations/Enhancements. Contrast degradations in an image are problems associated with poor brightness characteristics. The term *contrast* deals with the distribution of brightness within an image. An image may be said to exhibit poor contrast if either low- or high-contrast attributes are apparent, neither of which are generally considered visually appealing. When dealing with black-and-white images, high contrast is present when an image is composed primarily of dark black and bright white

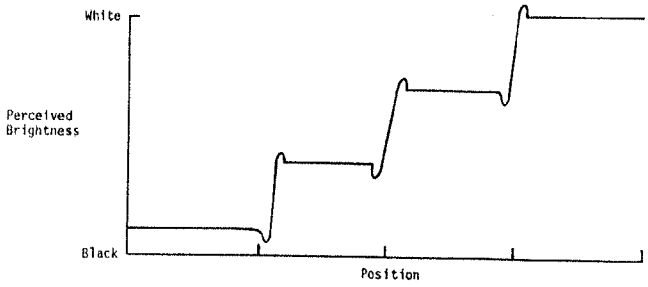
Figure 2-8
(a) Mach band effect seen in step gray scale.



(b) Actual brightness plot.



(c) Perceived brightness plot.



tones. Scene details are made up of harsh black-to-white transitions rather than the more natural, smooth gray tones. The appearance of a high-contrast image is that of intense boldness. Low-contrast images, on the other hand, are characterized by a washed-out look. Only middle gray tones exist, with dark black and bright white tones virtually nonexistent. Scene details appear subdued, making the viewing of such images difficult.

A well-balanced image of "good" contrast is composed of gray tones stretching from the dark blacks, through the grays, to the bright whites. Operations to correct an image of either low or high contrast are relatively simple. The results of these operations are corrected images where the overall gray tone balance is restored to a more natural distribution.

In contrast enhancement, simply achieving "good" image contrast is not always the ultimate goal. These techniques may further be used to make visible some aspect of the image that was previously hidden. In this type of enhancement, a resulting image of very high contrast, or other characteristic, may be desired. The results of these operations do not always yield aesthetically pleasing images but rather amplify some feature of interest.

Contrast enhancements are common image processing operations used to alter the overall brightness qualities of an image. They may be used to correct contrast deficiencies or to extract feature information not evident in the original.

Spatial Degradations/Enhancements. Spatial degradations in an image are problems associated with the presentation of image scene details. The term *spatial* deals with the two-dimensional nature of an image scene. An image may be said to exhibit poor spatial qualities if detailed areas are blurred or not well defined. Often edge details, such as black-to-white transitions, may be blurred, not exhibiting the sharp qualities generally associated with them. These types of spatial problems may be corrected or at least improved upon by relatively straightforward operations. The results are images in which spatial detail is restored to more accurately represent the detail of the original scene.

Additional spatial degradations include image noise, such as "snow" in a television image. Image scene geometric distortions also fall in this category.

As in contrast enhancement, correcting an image to a visually pleasing form is not always the pursued objective. Sometimes it is desired to enhance spatial details to an extreme, making object structure features more visible. Edge enhancement, where only object edge details are highlighted, is a common enhancement processing task.

Spatial enhancements are used to alter the spatial detail qualities of an image. They are often employed not only for image degradation correction but also in the extraction of object features not visible in the original.

IMAGE ANALYSIS

Image analysis operations produce nonpictorial results. Instead, the output is numeric or graphic information based on characteristics of the original image, with the objective of describing some aspect of the image and presenting the results to the viewer. Image analysis operations serve to describe image qualities, assisting in enhancement operations. Furthermore, descriptions of image scene features, automatic scene object measurements, and pattern recognition are all common analysis operations.

The most common analysis operation encountered in general image processing is that of the image histogram. The histogram relates, in bar graph form, the brightness distribution present in an image. Contrast information may be readily obtained from this graph, allowing the appropriate enhancement operation to be chosen. The brightness and contrast measurement given by the histogram is invaluable when attempting to correct an image for these degradations.

Aside from image quality measurements, analysis operations are most prevalent in automatic control applications. Such uses include the automatic dimensional measurement and classification of parts fabricated on an assembly line, automated security systems where certain known violations are watched for within a live video image, and remote sensing where aerial Earth images may be broken into various geological categories and tabulated for resource studies.

IMAGE CODING

The final class of image processing operations is that of image coding. These operations serve to reduce the amount of information necessary to describe an image. Two types of coding exist. The first codes an image in such a way that no information is lost. The reconstruction of the original image may be obtained uniquely from the coded version. The second codes an image into an abridged form. An example of abridged image coding would be to break the image into primitive subparts, or structures, coding only the location and orientation of each piece. While the second approach yields a considerably larger data reduction factor, the reconstructed image will often be far from an exact representation of the original. The choice of the coding scheme to be used is determined, as usual, by the application.

Reasons for using image coding lie particularly in the applications of image transmission and storage. Both make use of a limited medium and are therefore made more efficient through coding techniques. For instance, coding images into reduced forms allows either more image data to be transmitted in a given period of time or more to be stored in a given segment of a storage device. Image cod-

ing relates directly to conservation of equipment resources, particularly in bulk applications.

Processing Classification

The act of carrying out a computational operation is called a *process.* More specifically, the term *frame process* refers to an operation applied to an image frame, where *image frame* is simply a term used to denote an image in its entirety. All image processing operations may be said to be implemented through the use of a frame process. In the most general case, a frame process would be written in software and executed by a computer having access to the image data.

Previously, we broke the general field of image processing into three operational categories. The categories—quality enhancement, analysis, and coding—served to isolate these primary endeavors of research and application in such a way that each could be handled as a separate area of study. Although each plays a major role, we are primarily interested in quality enhancement. This is because most all images processed undergo some sort of quality enhancement either prior to or following other applied processes.

Because image quality enhancement operations are so popular, the most attention has been given to them. Image processing systems tend to support the enhancement operations to a higher level than the others. Also, since a fixed set of enhancements are often carried out on large sequences of related images, the speed with which the operation is carried out becomes of interest. For this reason, image processing systems often incorporate special-purpose hardware that allow the fast execution of certain families of enhancement operations.

In digital image processing, two processes are broken out of the general class of frame processing to be handled by high-speed hardware. These subsets are known as *point and group processes.* They serve to implement contrast and spatial enhancements, respectively. Furthermore, point processes are comprised of two parts—single image and dual image. The single image process allows standard contrast enhancements, while the dual image process adds the capability of combining multiple images. This overall process breakdown is illustrated in Figure 2-9.

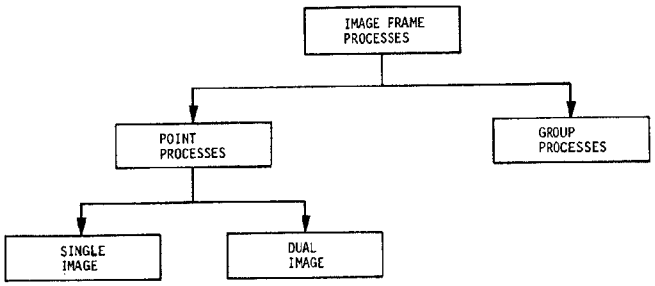


Figure 2-9 Process breakdown of image processing operations.

Implementation of point and group processes in special-purpose hardware allows the image quality enhancements to be handled expediently. Because of the predominant use of these operations, the overall image processing system throughout is increased as a result.

Generally, when speaking of a frame process, we are referring to a process other than the defined point or group process subsets. Using this premise, all image processing operations may be implemented through the use of one of four processes. The two point processes along with the group process provide for the bulk of image quality enhancements and are handled by high-speed hardware processors. The remaining operations involving other quality enhancements along with image analysis and coding are handled through the use of frame processes. Frame processes are generally executed through software programs by a host computer.

Part II of this book develops the above processing concepts. The reader will ultimately be armed with the knowledge to configure and execute image operations based on the requirements of a particular application.

The Processing System

A digital image processing system is a collection of hardware devices providing the digitization, storage, display, and processing of digital images. It is this system that yields the means of implementing a process upon a stored image. Figure 2-10 illustrates the common structure of a general purpose system.

The first order of business in any image processing system is to digitize an image. Most general-purpose systems will accept standard television video signals as inputs. The input device is normally a video camera imaging the scene of interest, whether live or a photographic print. The act of digitizing converts the analog electrical form of the video image into a digital form that may be stored in a digital memory device. Once the image exists in digital

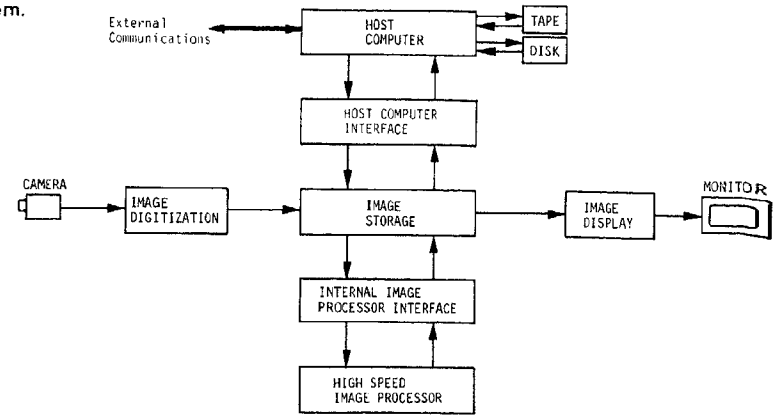


Figure 2-10 Basic image processing system.

memory, we are able to freeze it for subsequent display and processing.

The display of an image residing in memory is accomplished by repetitively reading the digital image data out to a display subsystem. Here, image data are reconverted to the standard television format and displayed on a television monitor. Once an image is brought into the image memory it is continuously displayed on the monitor.

The overseer of the image processing system is the host computer system. The host controls the system, deciding when to digitize, display, and process an image. The host also serves as the user's interface to the system. The host system, through the host computer interface, has direct access to the image memory such that it may carry out image processing tasks. Additionally, the host may transfer image data to and from long-term storage devices such as magnetic disk or tape. For small image processing systems, the host is often a microcomputer.

The final element of the digital image processing system is that of a high-speed image processor. Although the host computer has the ability to carry out any definable process upon a stored image, its speed of execution can be relatively slow. It becomes desirable to implement common image processes in hardware so that their execution time is reduced to a minimum. As mentioned earlier, the point and group processes are frequently handled in hardware. These processes allow the user the ability to modify image contrast and spatial attributes, as well as combine multiple images in an expedient manner. Since quality enhancement operations are most commonly used, added hardware can greatly augment the performance of the image processing system.

Part III of this book explores the basic configuration and design of the components of a general image processing system. Assuming reasonable skills in digital circuit design, the reader will gain the knowledge to configure and design a basic system.

The Digital Image

Digital image processing, by definition, operates upon pictorial information of a digital form. The conversion of every day images into this form is the most preliminary operation to occur prior to digital processing. Images of interest may be derived from a variety of sources: photographs, television, radar, scanning infrared detectors, acoustics, or X-rays—the list is virtually endless. No matter what the origin, however, the image must ultimately be placed into a format that the digital processor understands.

In general, digital image processing is carried out on standard television format images, because of the widespread standardization and acceptance throughout several related industries. In addition, almost any image may easily be converted to this format. From this point on we will assume all our processing to be carried out on standard black-and-white television images, with only brief departures to consider the differences in processing color images.

Let us now explore the terms, conventions, and format of the digital image.

Forming a Digital Image

A typical black-and-white photograph is composed of shades of gray spanning from black to white, and is known as a *continuous tone* image. This means that the various

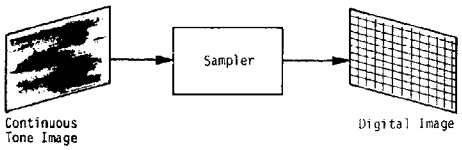


Figure 3-1 Conversion from continuous tone to digital image.

shades of gray blend together with no disruption to faithfully reproduce the elements of the original scene. The digital image processor, on the other hand, must work with discrete pieces of data on a one-by-one basis. To convert our continuous tone image to a digital image we must chop it into individual points of information. This "chopping" is referred to as *digitizing* or, more properly, *sampling,* because we are taking samples of the brightness of the photograph at specific locations within it. Each sample is given a numeric value based on its brightness, ranging from black through the grays to white. Additionally, each sample is assigned coordinates describing its location within the image. A sample is often referred to as a picture element, or *pixel,* because of its representation of a discrete element of the digital image.

An image is digitized into a square grid of pixels, each of which is labeled with a pair of coordinates—one defining the column that it is in and one the row. Column numbers range from 0, at the left-most side, to *n,* where *n* is the number of columns in the image. Likewise, rows are assigned numbers from 0, at the top-most side, to *m* where *m* is the number of rows in the image. As an example, the pixel with coordinates (200, 150) resides at the crossing point of column 200, row 150. To make better sense and (as we will see later) to conform more with television standards, the row coordinate is referred to as the *line* number and the column the *pixel* number in that line. This sampled image numbering convention is illustrated in Figure 3-2.

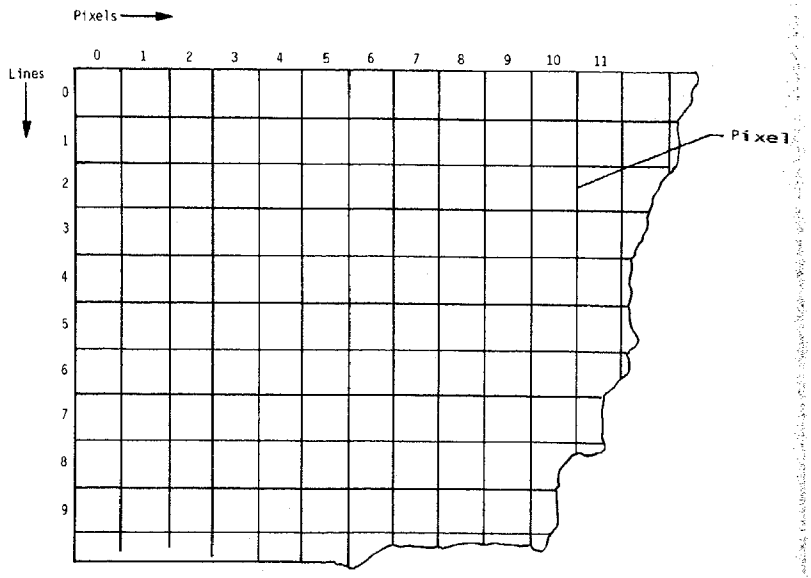


Figure 3-2 The discrete pixel numbering convention.

Our image has now been divided into individual, discrete points of information. Furthermore, each point has a defined location within the image, its coordinates, and a brightness value associated with it.

When dealing with the digitization of an image, there is always the question of how good the representation is when compared with the original. We define the limitations of the digitization process with the term *resolution.* Resolution is the separation of something into its basic subparts. In image processing, resolution may be broken into two definite types, *spatial* and *brightness,* with a third type, *frame rate,* playing a role not directly related to the visual appearance of an image.

Spatial Resolution

The term *spatial* refers to the concept of space, in our case a two-dimensional space. Our two-dimensional object is an image with a fixed height and width. When we speak of *spatial resolution,* we are describing how many pixels our digital image is divided into. Simply put, the finer this resolution is, the closer we approach the spatial appearance of the original image.

Optimally, we wish to digitize an image such that no information is lost in the translation from the original image to the digital image. This means that a properly displayed digital image will be identical to the original to an observer. In order to better understand the criteria for establishing the necessary number of samples required in a digital image, we must introduce the concept of *spatial frequency.* Any image contains scene detail in varying degrees. For example, details may range from the minutely detailed hairs on a person's head to the smoothly varying shades in the contours of the face. In quantifying this visual detail, we speak of spatial frequency or the rate at which brightness of an image changes from dark to light.

As an example of spatial frequency content in an image, let us examine the details present in Figure 3-3. If we look at a single line of the image as it crosses through varying details of the scene, the brightness of the scene encountered on the left half of the highlighted line is found to be erratic. These rapid changes in brightness are said to contain high spatial frequency. On the right side of the line, however, slowly varying shades of gray are observed. This portion is said to contain low spatial frequency. We must remember that when looking at a line of image we are actually only considering one-dimensional spatial frequency components. Spatial resolution takes into account the rate of change of brightness in an image going from left to right as well as top to bottom.

To make a decision about the necessary sampling rate needed to properly resolve an image, we use the classical *Nyquist Criterion,* also known as the Sampling Theo-

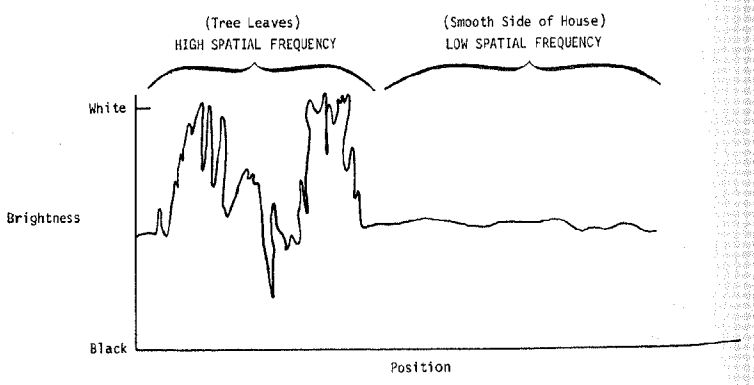


Figure 3-3
(a) A scene of varying spatial frequency detail.
(b) Brightness plot along highlighted line.

rem. This theory tells us in mathematical terms that to fully represent the rate of brightness change, or detail, in an original image we must sample it at a rate at least twice as high as the highest spatial frequency of the detail. In other words, if a particular detail in an original image varies from dark to light within a certain distance, our samples, or pixels, must be fine enough so that two of them fall upon the detail itself. It is also true that it becomes useless and wasteful to sample an image at a rate any faster than twice its maximum spatial frequency content. This may be further refined to say it is wasteful to sample at a rate faster than twice that of the finest detail *wished to be resolved in the digital image.* Some applications do not require that all details be present in the digitized image. Keep in mind, though, that once an image is digitized with a limit on the sampling frequency, the lost detail is gone forever.

At this point, we are ready to discuss spatial resolution of a digital image. The name of the game here is to take an image from a source, such as a photograph, and break it into enough discrete pixels so that the eye can detect no difference between the digitized image and the original. Figure 3-4 shows an image digitized to various spatial resolutions. The 32 line $\times$ 32 pixel image shows obvious coarseness where detail has been lost by the pixel "blocking" effect. As spatial resolution increases, the image looks more and more natural. A television, in fact, has 485 visible lines per image and the equivalent of roughly 380 pixels per line. Viewing the picture from a reasonable distance makes it difficult for the viewer to see any lines at all. However, up close, the lines are clearly visible. The chosen spatial resolution of a digital image must take into account three factors—the detail in the original to be seen in the digital image, the displayed size of the digital image, and the viewer's distance from it. As a rather gross exam-

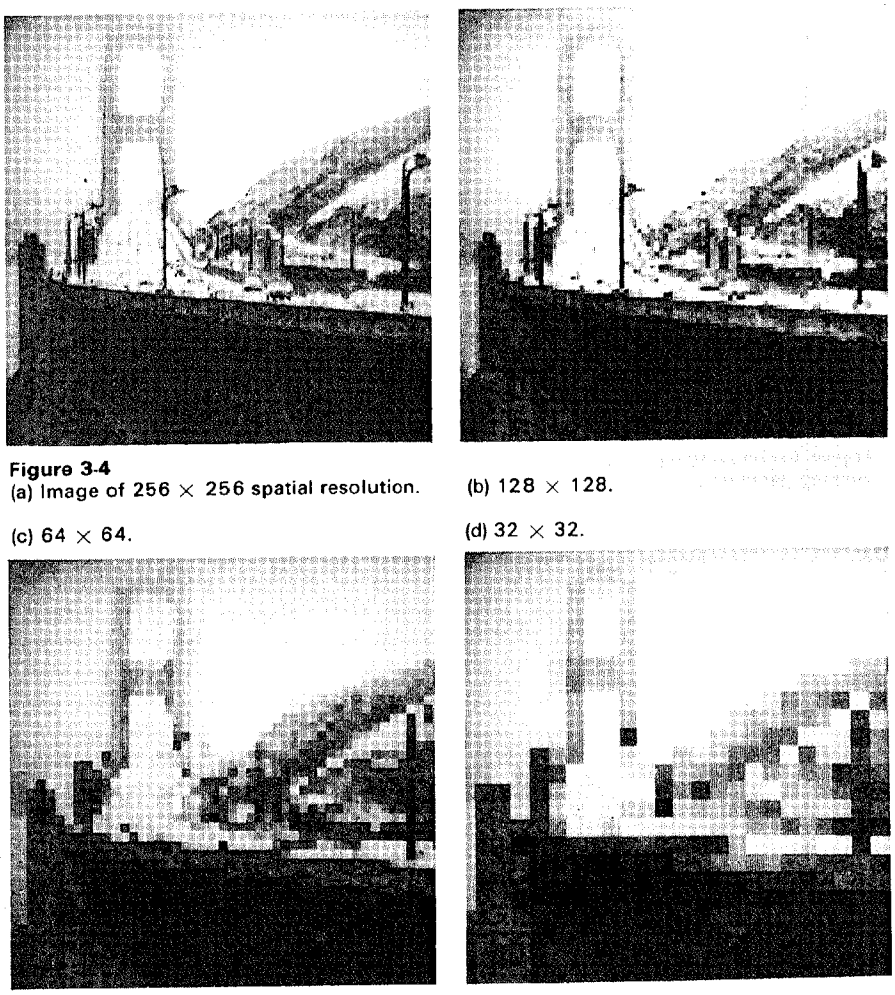


Figure 3-4
(a) Image of 256 × 256 spatial resolution. (b) 128 × 128. (c) 64 × 64. (d) 32 × 32.

ple, digitization of an image for use in a movie requires on the order of a 4,000 line × 7,500 pixel image so that when displayed on a standard movie screen, considerably enlarged from the original, the viewer does not see the individual pixel composition.

The distance that an image appears from its observer directly determines how much scene detail will be visible. For instance, a photograph held twelve inches from an observer will show considerably more detail than when held five feet away. Therefore, as the display-observer distance is increased, spatial resolution in the image may be decreased. Looking at the images in Figure 3-4, we see that the visible pixel "blocking" effect diminishes as the images are pulled farther and farther away from the eye. The display-observer viewing geometry is depicted in Figure 3-5. As the distance is doubled, so may the size of the observed display increase without any detectable spatial detail loss to the observer.

In practice, it is sometimes prohibitive to actually implement the spatial resolution required by the above con-

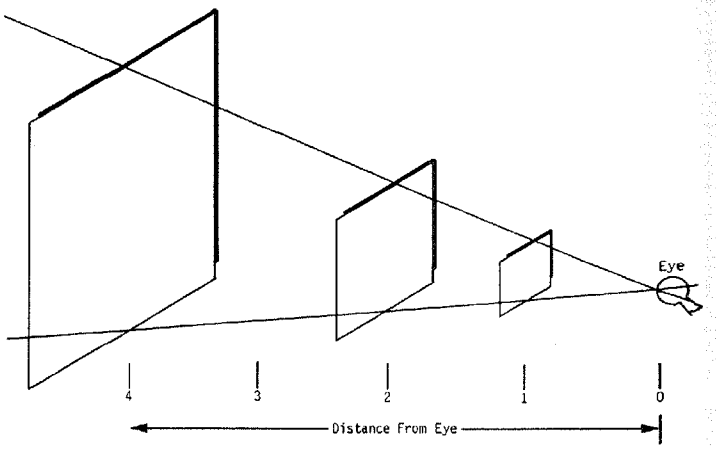


Figure 3-5 The display-observer viewing geometry.

siderations. This is because storage of the image in computer memory becomes unmanageable, while increased computational time required to process the image becomes undesirable. So, as in most endeavors, we are faced with a tradeoff, image resolution presented to the observer versus computer memory storage space and processing time.

It has become somewhat of a standard among image processor equipment manufacturers to chose a spatial resolution of 512 × 512 to be compatible with television formats. You will note that a resolution of 512 × 512 and not 485 × 380 is selected. This is because 512 is an even power of 2, a number convenient to the computer and digital processing hardware world. It should be noted that other resolutions are also common. 1024 × 1024 is often found in high-resolution applications where high detail must be resolved accurately within an image. On the other hand, 256 × 256 is quite acceptable for general use in education, machine process control, hobby, and many other applications. Once again, an image of any of these spatial resolutions is acceptable to the viewer when observed from an appropriate distance. All images appearing in this book are 256 × 256.

There is one other topic worth mentioning with regard to spatial resolution, the concept of *aliasing*. The phenomenon of aliasing has to do with the erroneous representation of original-image, high-frequency detail within the digitized image. Aliasing appears when the Nyquist Criterion is violated for a given spatial frequency present in the original image. This occurs when a detail within a scene has a spatial frequency greater than half the sampling frequency. In this case, the detail is said to be *undersampled*. The high-frequency detail ends up being translated to a lower frequency because some of the brightness transitions are missed in the sampling process. This is illustrated in Figure 3-6. When aliasing occurs, moiré patterns may appear in the digitized image. Unless an image of very repetitive high-frequency detail is undersampled, though, the effect of aliasing may generally be disregarded.

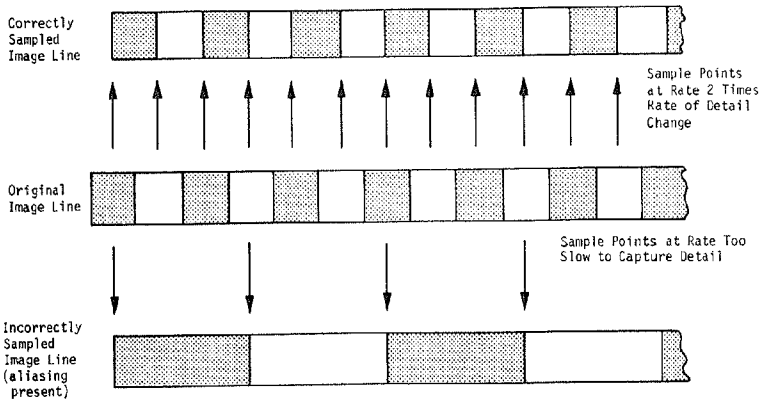


Figure 3-6 The aliasing effect when undersampling is present.

Brightness Resolution

The second resolution concerning digital images is that of *brightness.* As covered in the previous section, every pixel represents the brightness of the original image at the point of its sampling. The concept of *brightness resolution* is concerned with how accurately the digital pixel brightness compares to the original brightness at the same location in the image.

The digitizing process samples the original image at predetermined grid locations. Each sample brightness is then converted to an integer numeric value; this is known as *quantization.* The quantization operation converts an analog brightness level at a sample point to a numeric value within a certain accuracy tolerance, or brightness resolution. This process is carried out by an Analog-to-Digital, or A/D, converter and will be discussed in Chapter 6.

In quantizing the brightness of a pixel, we must first define to what accuracy the conversion will be made. For instance, conversion to a three-bit binary number allows each pixel to be represented by one of eight brightness levels. These levels are represented digitally by the binary numbers 000, 001, 010, 011, 100, 101, 110, and 111, spanning in brightness from black to white. The eight levels of brightness comprise what is called a *gray scale,* or in this case, the three-bit gray scale. Clearly evident in the three-bit gray scale, shown in Figure 3-7, are the eight distinct levels of brightness. Each brightness is easily discernable by the eye. Figure 3-8 shows an image quantized to brightness resolutions from one to six bits. Figure 3-9 breaks an image of six-bit brightness resolution into separate "bit-plane" images, illustrating the impact of each bit on the overall image.

We must pay attention to the number of levels of gray available to the brightness quantizer. As seen, the lower brightness resolution images allow the quantization to be visible to the eye. Increasing the number of bits representing brightness expands the gray scale so that it blends together more undetectably. For each added bit in

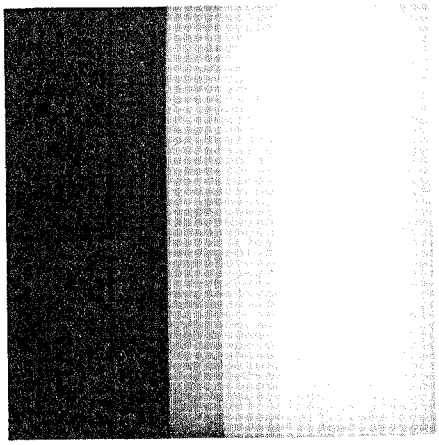
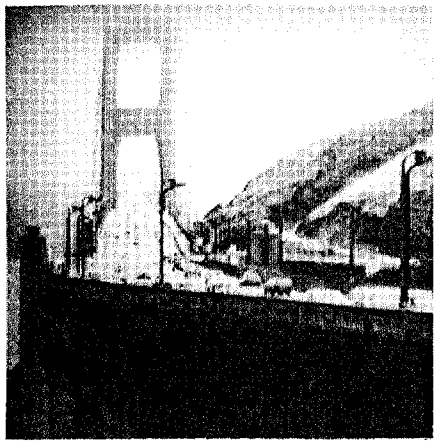
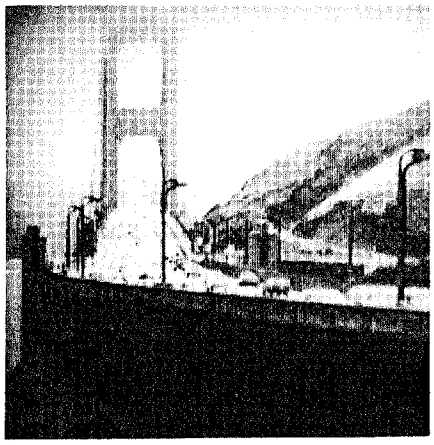


Figure 3-7 The 3-bit gray scale.

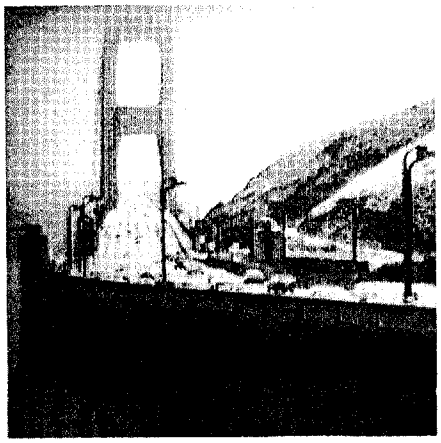
Figure 3-8
(a) Image of 6 bit = 64 gray levels resi



(b) 5 bit = 32 gray levels.



(c) 4 bit = 16 gray levels.

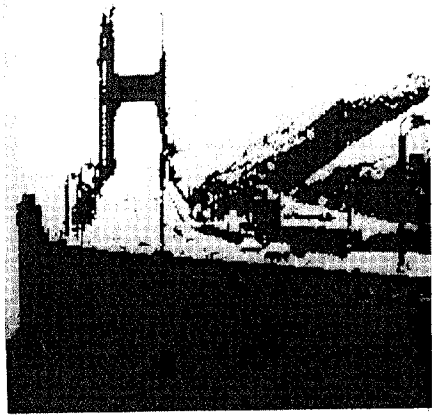


(d) 3 bit = 8 gray levels.

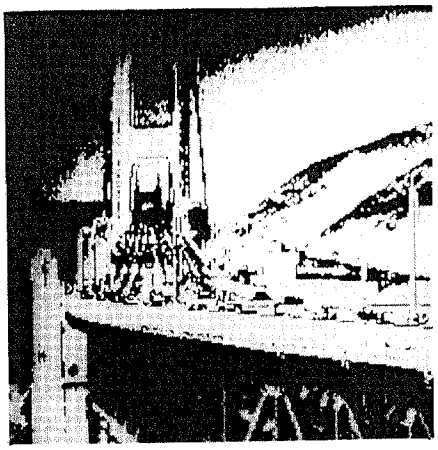
(e) 2 bit = 4 gray levels.

(f) 1 bit = 2 gray levels.

Figure 3-9
(a) Image bit planes, bit 6 (most significant bit).

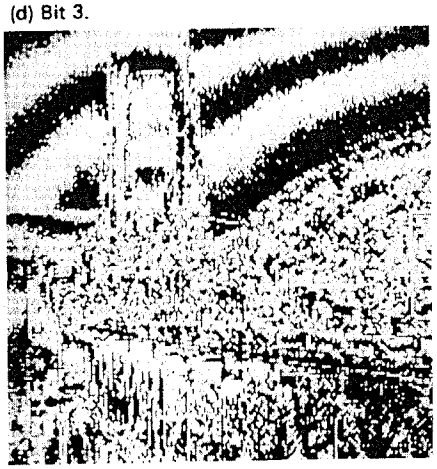


(b) Bit 5.

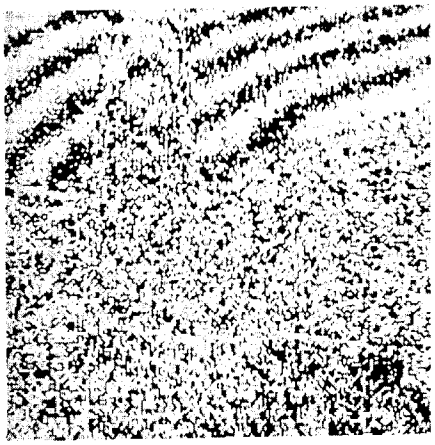


(c) Bit 4.

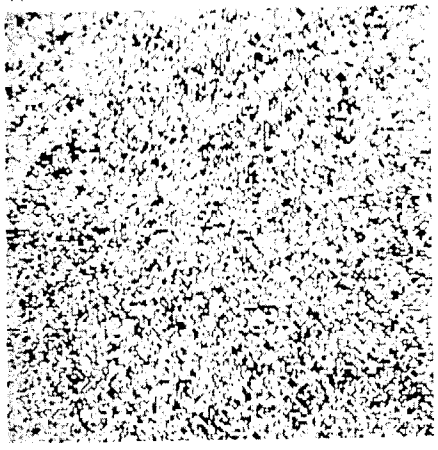
(d) Bit 3.



(e) Bit 2.



(f) Bit 1 (least significant bit).



the conversion, the brightness resolution of the gray scale is doubled.

Referring back to the two-bit image in Figure 3-8, we see the effect of the phenomenon known as *contouring*. Contouring occurs because of limited quantization—all pixels are allowed to fall within one of only four brightnesses. The continuous tone quality of the original is limited, making areas of the digital image abruptly change from one gray level to another, where the change was gradual in the original. Where resolution entered into detail representation in the spatial domain, it is contouring effects that are to be minimized by appropriate resolution in the brightness domain.

Before making the final decision of how many bits to assign to the gray scale, we must touch on the response characteristics of the eye. As we saw in Chapter 2, the eye is much more sensitive to intensity changes in dark regions of an image than in bright regions. The eye responds logarithmically to intensity change. This means that contouring may be visible in dark areas of a digitized image when not evident in lighter areas. For this reason, it is often beneficial to quantize brightness on a logarithmic scale rather than a linear scale. This means that the dark, or low-brightness areas, will be represented by more gray levels, leaving the high-brightness areas represented by fewer.

The logarithmic gray scale buys us the ability to place finer brightness resolution in the low end of the gray scale, where the eye is most sensitive to intensity change. In effect, we have linearized the response of the eye by making the gray scale logarithmic.

Image processor equipment manufacturers have generally adopted 8-bit logarithmic gray scales to represent digitized images. The 8-bit quantization assures no detectable change from one gray level to an adjacent one, where logarithmic quantization takes advantage of the response characteristics of the eye. For low-end user applications such as amateur radio, computer hobbyism, and some machine process control, 6-bit, 4-bit, and even lower gray scales are often acceptable. Additionally, linear quantization is often not a drawback for these uses. Because of the difficult hardware implications, we will limit our discussions in this book to an 8-bit linear gray scale.

Frame Rate

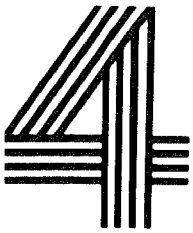
A more subtle form of resolution manifests itself in the image processor hardware display of digital images. With the normal display mode being that of a television-type device, we are interested in the rate at which the image is updated, known as *frame rate*. Commercial television updates its image entirely, using a technique called interlacing, every 1/30 of a second with little flicker ob-

servable to the viewer. Reducing flickering in a displayed image is important when viewer fatigue is of concern.

The use of standard television monitors is common in image processing, where the frame update rate of 30 times per second produces little distraction to the viewer. Because of this, most image processing equipment is designed around the commercial television standard. The standard calls for refreshing the displayed image in an *interlaced* format every 1/30 of a second. Interlacing gives the impression to the observer that a new frame is present every 1/60 of a second.

High-end image processors will often raise the frame rate to 1/100 of a second. The result is a rock-steady image display. This requires the use of higher-speed image memory and special-purpose display monitors. We will dwell on the standard 1/30 of a second interlace scheme. This approach is more conventional, especially when dealing with a standard television camera input, as we will.

Interlacing, frame rate, and spatial and brightness resolutions are discussed in Chapter 6, where the standard television video format is overviewed.



The Histogram

Image analysis operations deal with the generation of numerical descriptions of various image characteristics. An important class of these operations is used in the analysis of image degradations prior to enhancement. In particular, contrast attributes of an image are of major interest, giving a good overall quality assessment. A tool, known as the *image histogram,* gives us a concise, easy-to-read measure of this important parameter. In general terms, a histogram is defined as a frequency distribution graph of a set of numbers. Our special version is the gray level histogram, giving us a graphical representation of how many pixels within an image fall into the various gray level boundaries.

A histogram appears as a graph with "brightness" on the horizontal axis from 0 to 255 (for an 8-bit gray scale), and "number of pixels" on the vertical axis. To find the number of pixels having a particular brightness within an image, we simply look up the brightness on the horizontal axis, follow up the graph bar and read off the number of pixels on the vertical axis. Since all pixels must have some brightness defining them, adding the number of pixels in each brightness column will sum to the total number of pixels in the image.

The histogram gives us a convenient, easy-to-read representation of the concentration of pixels versus brightness in an image. Using this graph we are able to see im-

mediately whether the image is basically dark or light and high or low contrast. Furthermore, it gives us our first clues as to what contrast enhancements would be appropriately applied to make the image more subjectively pleasing to an observer.

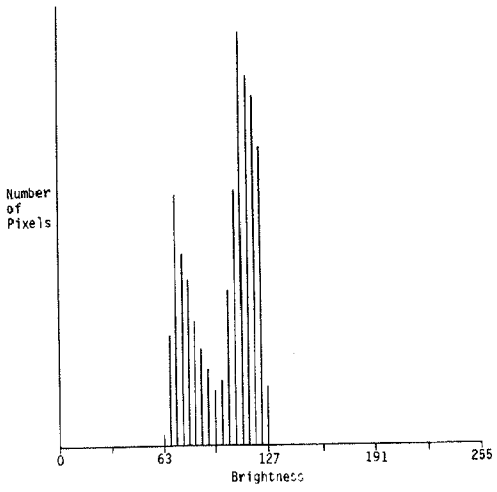
Contrast/Dynamic Range Indication

A term often used in describing an image of any sort is *contrast.* We intuitively understand contrast to mean how dull or sharp an image appears with respect to gray tones. Contrast in an image is clearly illustrated in the histogram. Low contrast appears as a mound of pixel brightnesses in the gray scale leaving other gray regions completely unoccupied. High contrast shows up as a bimodal histogram where two peaks exist at the outer brightness regions. We see that by "reading" the histogram, contrast parameters become evident, allowing us to further pursue the correct contrast enhancement approach. A well-balanced image is generally characterized by medium, or "good" contrast.

Dynamic range is a measure of how wide the occupied portion of the gray scale is. For instance, a mound of pixels falling between gray values 50 and 100 (within a range of 0 to 255), with none in the other regions, indicates a small dynamic range of brightness, whereas a wide gray scale distribution shows large dynamic range. An image with small dynamic range does not occupy all the available spread of gray values; the image has really only been quantized to a gray scale comprised of the occupied range. This indicates low brightness resolution along with low contrast. Large dynamic range generally implies a well-balanced image except if it is a bimodal distribution, in which case the image is high contrast. Three common histograms, along with their original images, are illustrated in Figures 4-1 through 4-3.

Figure 4-1
(a) Low contrast/low dynamic range image.

(b) The histogram.



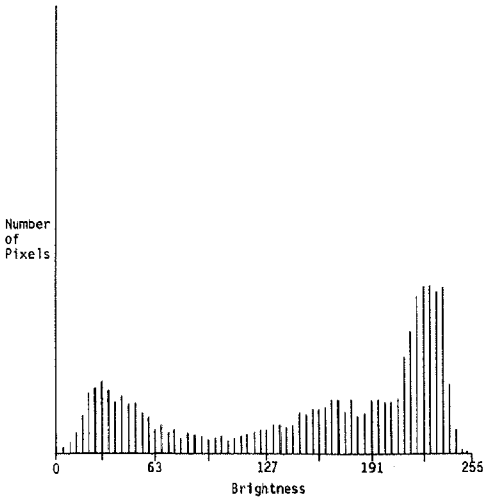


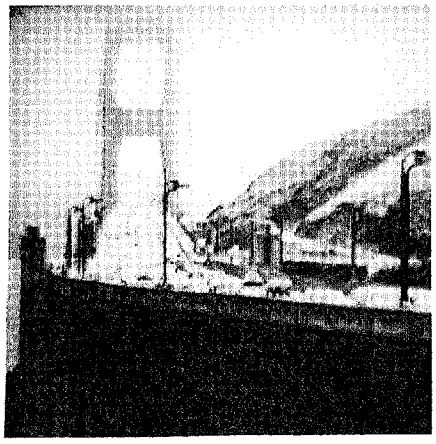
Figure 4-2
(a) High contrast/high dynamic range image. (b) The histogram.

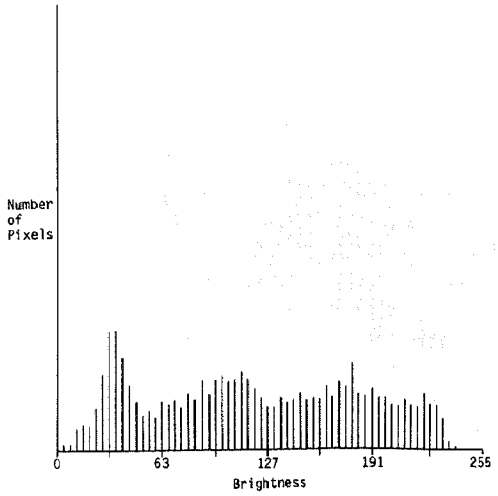
It should be noted that natural images generally are characterized by wide dynamic range and medium contrast. However, this is not always the case. It just may be the attributes of the original scene that dictate departures from a well-balanced image. Often, though, even if the original had low contrast or low dynamic range, a corrected version will be more appealing to the viewer.

Manipulation Effects

In dealing with the image degradations evidenced by the histogram, we will now touch on basic histogram manipulation, which will be discussed in more detail in Chapter 5. Two modifications are commonly implemented: *histogram sliding* and *histogram stretching*. These operations are meant to redistribute the histogram so that contrast and

Figure 4-3
(a) Good contrast/high dynamic range image. (b) The histogram.





dynamic range may be enhanced. Given the low contrast image and histogram of Figure 4-1 we see that the pixel gray scale distribution is clumped in one area of the graph. Not only is low contrast indicated here, but also low dynamic range. By sliding the "clump" to the left and then stretching it out to the right we effect a higher contrast and wider dynamic range. This makes the image better balanced and more natural as a result.

The sliding operation is simply the addition or subtraction of a constant brightness to all pixels in the image. A pixel of brightness 20 attains a brightness of 30 when 10 is added. Doing this to every pixel effectively slides the entire graph to the right by 10 gray levels. The basic effect of sliding is a lightening or darkening of the image.

Histogram stretching is the multiplication of all pixels in the image by a constant value. A histogram, with all pixels residing in the left half, will be spread out to occupy the entire gray scale range when multiplied by a constant of 2. This operation stretches the contrast and dynamic range of an image.

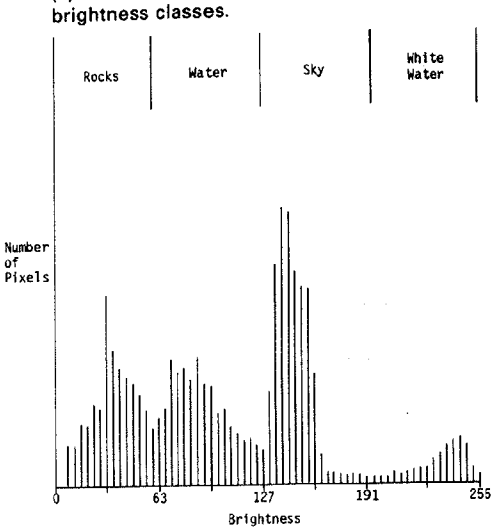
There exists an interesting point concerning the stretching of a low dynamic range image. In an image in which the histogram shows all pixels falling in the left half, only half of the gray levels are actually used. Although the image appears dark, we may stretch the histogram to make the distribution span the entire range. Dynamic range is increased—which is desired—but there are still only half the number of allowable gray levels occupied. (Remember, there were only half to begin with.) Where the original pixels fell, the original brightness resolution was maintained, but the image appeared dark. Now the stretching operation has stretched the original gray values to twice the range by skipping every other gray value in the case of multiplying by two. We have not lost brightness resolution, merely redistributed the original information.

Object Classification

Elementary classification of objects within an image scene is sometimes feasible through histogram analysis. The beach scene, illustrated in Figure 4-4, is primarily composed of four basic elements—white water, sky, calm water, and rocks. Each of the elements tend to be comprised of gray levels different from one another. What we are able to do is carry out a classification of scene objects based on their gray level compositions. Looking at the image histogram, the four regions of gray level concentration are seen as distinct peaks separated by valleys. For classification, the histogram may be broken into the four gray level regions, each labeled for the objects represented. Using a simple pixel point process (to be covered in Chapter 5), we may generate an image where only four gray levels appear, each representing one of the four classified ob-

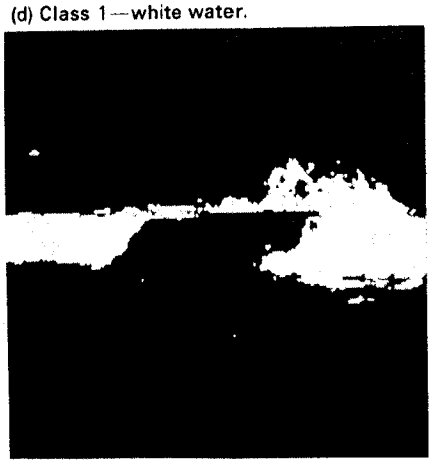
Figure 4-4
(a) Original beach scene.

(b) The histogram broken into four brightness classes.



(c) The original scene classified by use of four gray levels.

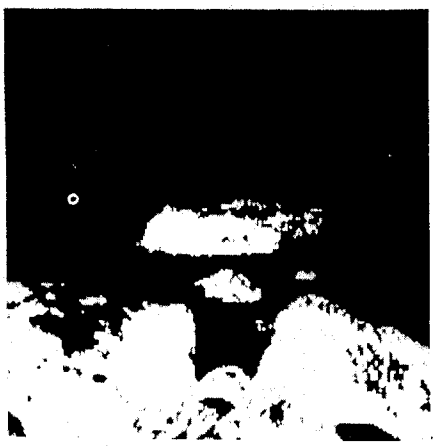
(d) Class 1—white water.



(e) Class 2—sky.

(f) Class 3—calm water.

(g) Class 4—rocks.



jects. Furthermore, four independent images may be generated, each highlighting the objects of one classification category.

Like all processing of data, there is a drawback, or undesirable artifact, to this technique. Objects in a scene may be composed of gray level regions that overlap, objects will then have portions that fall in another's classification. Additionally, unclassified areas may have pixels spanning the gray scale. Those pixels are then improperly classified as belonging to certain object groups.

The histogram object classification operation is basic. It works best on simple scenes with objects that have distinctly different gray scale occupancies. However, in certain applications, the simplicity of this technique may be exactly what is called for.

5 Picture Operations

We touched on various processing fundamentals in Chapter 2. Remember that all operations of interest to us fell within four process categories: point processing with one image, point processing with two images, group processing, and frame processing. These categories—though broad in an applications sense—each relate to a fundamental operation, with the exception of frame processing, which is meant to classify various functions not included in point or group processing.

Throughout this chapter, reference to Image Operation Studies will be made when appropriate. These studies are compiled in Part IV, "Processing in Action," and are provided to consolidate the image operations introduced in this chapter. It should be noted that the processing techniques presented here are not meant to represent an exhaustive study of image processing capabilities. Rather, they represent an overview of the most commonly used processes. The hope is that the reader will become sufficiently stimulated to gain any required depth in the field through additional studies. Good references to these topics include Further Reading/References I-1 through I-8.

In Chapter 3, a pixel within an image was spatially located by its line and pixel coordinates. Using the Cartesian coordinate system, the pixel coordinate is represented by x and the line coordinate by y. For instance, the pixel

located at the crossing point of line number 25 and pixel number 125 is denoted by the coordinates (125, 25). We take this convention a bit farther in this chapter, for we must now discriminate between input and output images.

An *input image* is defined as an image that is used as data to be processed. Any resulting image is referred to as an *output image*. So in calling out the coordinates of an image pixel, a prefix of either *I* or *O* is used to denote input or output image. Where multiple input images are used in an operation, a subscript may be appended to the *I* prefix.

The general case flow diagram of an image processing operation is depicted in Figure 5-1. Input images are denoted by $I_1(x,y)$ and $I_2(x,y)$. In an operation requiring a single input image, no subscript need be used. The output image, if present, is denoted by O(x,y). This basic diagram provides the fundamental representation of all image processing operations.

A typical image processing system possessing reasonable image quality representation may be based on an image resolution of 256 × 256, quantized to 8 bits. We will dwell on these resolution parameters throughout the proceeding development of topics.

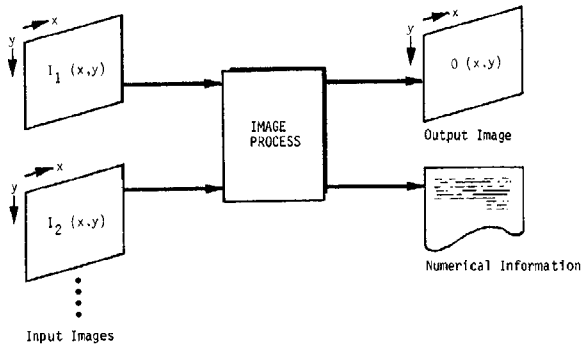


Figure 5-1 The image operation flow diagram.

Pixel Point Processing/Single Image

Pixel point processing is the most fundamental class of image processing operations. Used primarily in contrast enhancement operations, the pixel point process is a simple yet invaluable tool. Point processes allow the alteration of pixel gray scale occupancy. On a one-by-one basis, the gray level of each pixel in the input image is modified, often by a mathematical or logical relationship, to a new value and placed in the output image at the same spatial location. All pixels are handled individually. For instance, the pixel at coordinates I(x,y) in the input image is modified and returned to the output image at coordinate O(*x,y*). With this in mind, we note that point operations process pixel brightness attributes with no action on spatial attributes. Spatial processing, as we will see later, is handled by pixel group and frame processing.

The general equation for a point process is given by the equation

$$O(x,y) = M[I(x,y)]$$

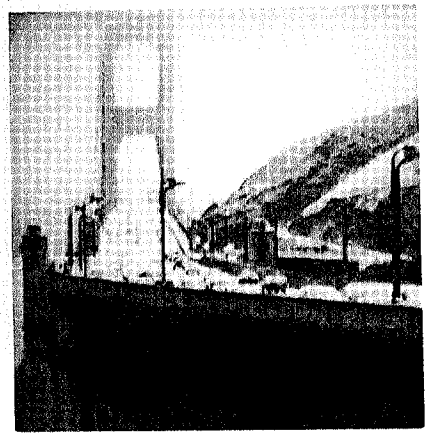
where M is the *mapping function.* It is implied that all pixels in the input image are operated upon. This means that the brightness of an output pixel residing at coordinates (x,y) is equal to the brightness of the input pixel at coordinates (x,y) after being modified by the function M. We refer to the function M as the mapping function because it maps input brightnesses to output brightnesses.

As an example, suppose that we wish to make a negative image from a positive one. Here, just like a photographic negative, the blacks in the input image will become white, the whites black, and the grays in between take on their respective negative qualities. This operation, often referred to as the *complement image operation,* does prove useful. As we learned, the eye responds better to slight changes in contrast in dark regions of an image than in light regions. With this process, slight contrast changes in the bright areas in the input image, normally undetectable, are transformed into the dark regions in the output image where they now become visible.

Figure 5-2 illustrates this operation along with the mapping function. By locating the input pixel gray level on the map's horizontal axis, moving up to the map point and across to the vertical axis, we acquire the respective output gray level. As expected, black (0) maps to white (255) and vice versa. All of the intermediate gray levels are correspondingly mapped, yielding the final complemented image. Image Operation Study #5 provides additional insight into this operation.

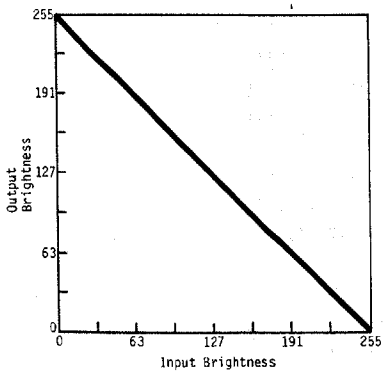
The uses for point processing are vast. As mentioned in the previous chapter, histogram manipulation is carried out by a point process. This class of point processes is re-

Figure 5-2
(a) Original image.



(b) Complemented image.

(c) The complement operation mapping function.



ferred to as *contrast enhancement.* Another operation known as *photometric correction,* deals with the correction of problems caused by photosensor incongruencies. One such use corrects spacecraft image sensor nonlinearities caused by size and weight constraints. The artist may make use of point processing when developing silkscreen overlays, graphics, and glass etching masks.

Point processing is a simple but truly fundamental element of digital image processing. No matter what type of operation is employed upon an image, some form of point processing is probably involved, even if it is used to simply clean up undesired artifacts left behind by another process.

CONTRAST ENHANCEMENT

We discussed histogram sliding and stretching in Chapter 2; now we will fully describe the uses and effects of these operations.

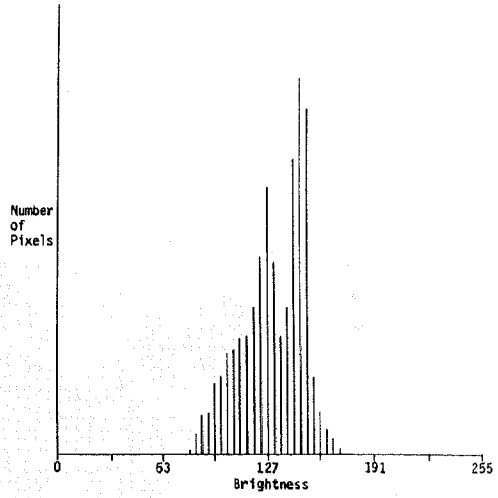
The first thing we do is generate a histogram of the image to be processed. This charted pixel gray level distribution of an image will often tell us immediately where the "sore" spots are. These troubled areas are expressed in terms of contrast and dynamic range of the distribution. Referring back to Figures 4-1 though 4-3, we see histograms of images with varying qualities—low contrast/low dynamic range, high contrast/high dynamic range, and good contrast/high dynamic range. These are the types of histograms most frequently encountered. By sliding and stretching the histograms, we may make an image of poor contrast quality quite respectable.

Contrast enhancement is a point process involving the addition, subtraction, multiplication, or division of a constant value to every pixel within the image. The histogram is useful in determining the operations to be employed and in measuring the accomplishments afterward.

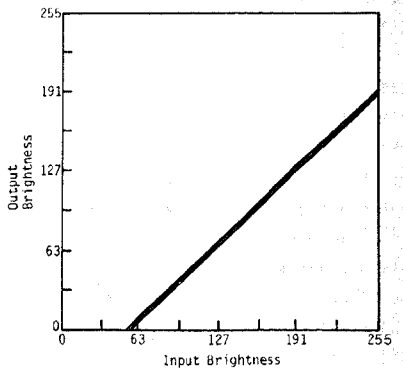
Figure 5-3 illustrates an image progressing through the contrast enhancement process. We see the original image histogram displaying a mound of gray levels occupied in the center. The image appears low in contrast. The enhancement of increased contrast may be accomplished by first sliding the mound of gray levels down to the dark area of the histogram. We do this by subtracting 60 from each pixel's brightness, using a point process with the appropriate map. The resulting image yields no more contrast; we have simply relocated the pixel brightness range so that the darkest pixels of the original are now actually full black. Now comes the stretching. To make the mound stretch the full range of grays we must multiply every brightness by 2. Black (0) remains 0, because $0 \times 2 = 0$. A pixel of brightness 10 becomes 20, and so forth, to the maximum-valued pixel in the input image of 120 increasing

Figure 5-3
(a) Original low-contrast image.

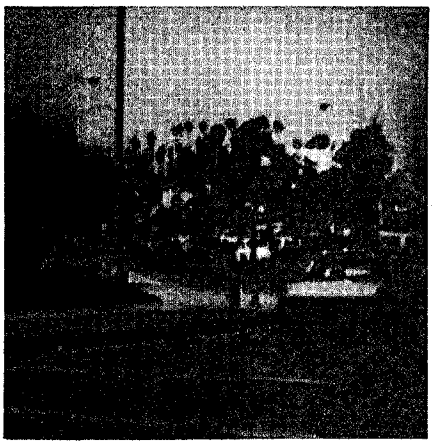
(b) The histogram.



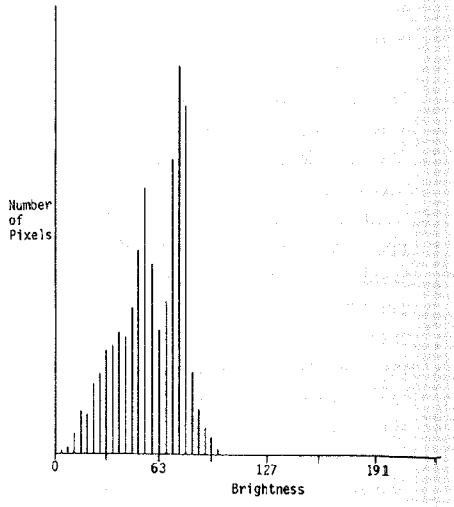
(c) Histogram slide map.

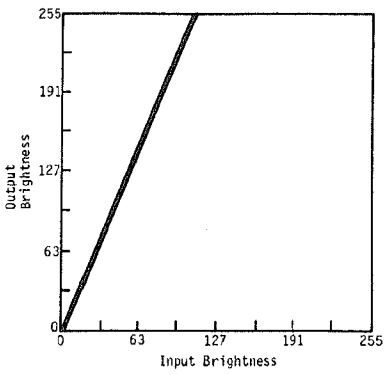


(d) Image after histogram slide.



(e) Histogram after slide.

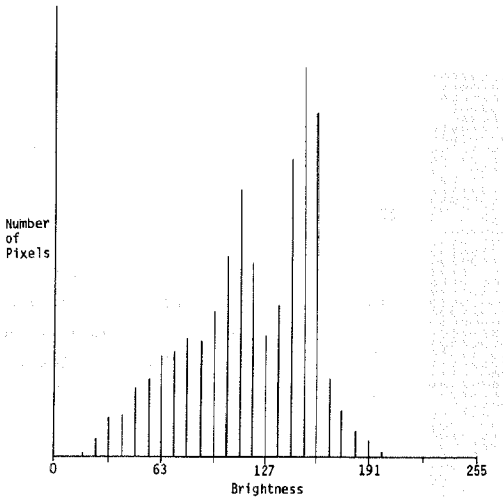




(f) Histogram stretch map.

(g) Image after histogram stretch.

(h) Histogram after stretch.



to 240. 240 is virtually the limit of the 256-level gray scale. Again, a pixel process is used to effect the stretch. Our image now appears well balanced with good contrast characteristics. The final histogram shows the same. These contrast enhancement operations are further discussed in Image Operation Studies #2, #3 and #4.

All images may be operated on in the above manner to acquire histograms that appear well balanced. But what if the desired result is that of high contrast? Many times production of an image of obtuse appearance is desired in order to make some attribute clearer. In short, the subjective criteria on which an image is judged good or bad are based on its intended application.

Some images are very low in contrast as a function of their origin, such as a scene with poor lighting. An example of this is a low-light-level original in which only a few gray values separate background from object. Figure 5-4 shows an image of a road sign taken under these undesirable lighting conditions. In order to read the lettering clear-

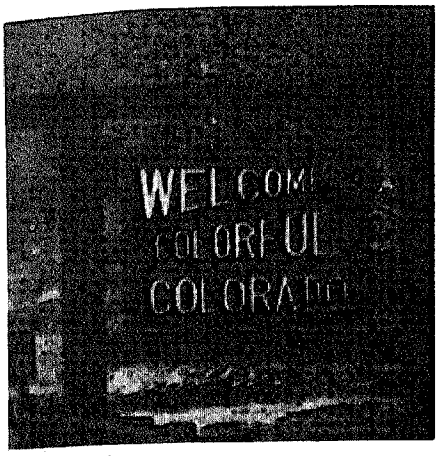
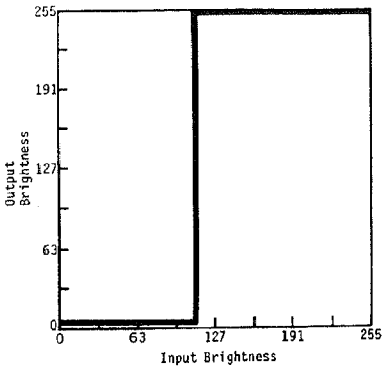


Figure 5-4
(a) Original low-contrast, dark image.

(c) Output image with sign lettering highlighted.



(b) Binary Contrast Enhancement map.

ly, we may implement what is known as a *binary contrast enhancement operation.* The mapping function illustrates that all pixels of brightness less than that of the chosen threshold brightness will be set to black (0), and those above will be set to white (255). By choosing the correct threshold value, we may force the lettering to go to white and the background to black. This is possible because in the original, the letters appear slightly brighter than the background. The processed image is characterized by sharply highlighted lettering appearing on a black background. Image Operation Study #1 discusses this operation in greater depth.

It is important to realize that contrast enhancement, like any image operation, does not have an absolute "goodness" quality for which we always aim. Different applications see contrast enhancement as meaning different things, depending on what the user wishes to ultimately see in the processed image.

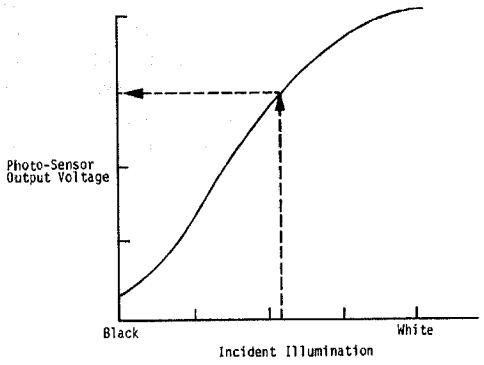
PHOTOMETRIC CORRECTION

In any system of image gathering and display, we encounter certain degradations which are based on the equipment used. These degradations come in the form of photometric and geometric distortions. *Photometric distortions* relate to brightness response incongruencies of a sensor device. *Geometric distortions* are spatially oriented. Ideally, the system should produce an image identical to the original scene with no degradations added. However, using available equipment, some distortions will occur, although they may be slight. Often space flight imaging equipment is purposely not optimized for low distortion due to size and weight considerations. In such a case, the idea is to characterize the degradations before flight and correct them as image data are received on the ground. Spacecraft movement may cause geometric smearing but if the distortion is well characterized, correction is often simple. Geometric corrections will be covered under frame processing.

The word *photometric* refers to the properties of light intensity response. In this case, we are concerned with the intensity response characteristics of a sensor device to illu-

mination. Photometric correction, therefore, deals with the correction of sensor illumination response deficiencies. An important word here is correction. This is not an enhancement as previously defined, but a correction based on factual knowledge of the distortion.

Though this process holds true for correcting degradations induced by any photosensor or display device, we will dwell on the spacecraft example. A spacecraft image sensor is generally comprised of a solid-state photosensor device, like a photo-diode, that converts light intensity to a voltage level. The voltage level, in turn, represents a sensed brightness gray level. The device is scanned horizontally, sweeping out lines of image data, while scanning vertically to sweep the entire frame. Of interest to us are the intensity response parameters of the photosensor device—if the light impinging the surface of the device doubles in brightness, does its output voltage level double? A typical photosensor device response may be similar to that seen in Figure 5-5. Using this curve we note nonlinearities where the response deviates from a straight line. To correct this effect, we use a mapping function that counteracts the bad effects by mapping the curved response back to a linear response.



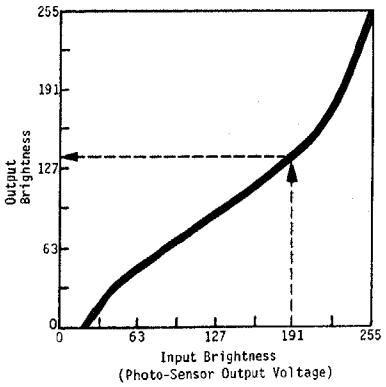


Figure 5-5
(a) Typical photosensor response curve.
(b) Map to correct the photosensor response nonlinearities.

We may further extend this operation to correct image data being sent to a display device in order to compensate for display distortions. For instance, a television monitor (or the like) may not produce a brightness on the screen, at any one point, linearly related to the driving voltage representing the point. These degradations, once characterized, may be corrected before sending the image to the monitor much in the same way as sensor correction was done.

ART APPLICATIONS

Today's graphic artist has at hand the power of image processing, with point operations playing a major role. These processes offer the ability to alter gray scale attributes.

Once photographic darkroom techniques, they may now be done quickly, repeatedly, and without chemicals, greatly increasing flexibility. Silkscreen mask generation is one such application.

Silkscreen prints are generally comprised of several colors—say four—all overlaid with one being the background color. To produce the overlay masks, the artist may strip all but the two high-order brightness bits from each pixel's gray level. This leaves a four-gray-level image. Each gray level represents what each overlay will look like. To generate an overlay, one gray level is selected and set to white while all other pixels assume black. By doing this four times, once for each gray level, the overlay masks are generated. Photographing the display monitor each time allows transposition to the silkscreen medium itself. Of course, with simple point processing, an endless variety of other interesting effects are at the hand of the inspired artist.

Pixel Point Processing/Dual Image

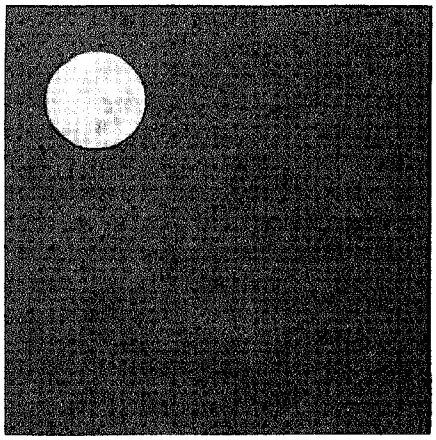
We have seen how point operations work on single images; let us now apply this technique to image pairs. Instead of mapping pixel brightness from one image to an output image, we now map two pixel brightnesses, one from each of two input images, into an output image. Again, we are talking with regard to point processes which implies each pixel is handled independently.

The mapping function for dual images becomes somewhat more involved than that of single images. With 8-bit input pixels, each may take on one of 256 different brightnesses. Since each pixel in an input pair is independent, we have a total of 256 × 256 different possible input combinations being mapped into 256 possible output gray levels. The map for this type of function must be displayed in three dimensions and, unlike the single-image point map, is not easily interpreted. For this reason, we generally work in terms of a *combination function.* This name is appropriate because it refers to the way in which the two input images are combined. Our dual image point process equation is of the form

$$O(x,y) = I_1(x,y) \;\&\; I_2(x,y)$$

where $I_1(x,y)$ and $I_2(x,y)$ represent the two input images. The symbol & is used to denote the combination function. As before, it is implied that all pixels in the images are operated upon.

Combination functions include mathematical and logical operators such as +, −, ×, /, AND, OR and EXclusive OR. We now have the ability to add, subtract, multiply, divide, AND, OR, and EXclusive OR image pairs, opening an additional world of point processing. For instance, dual-im-



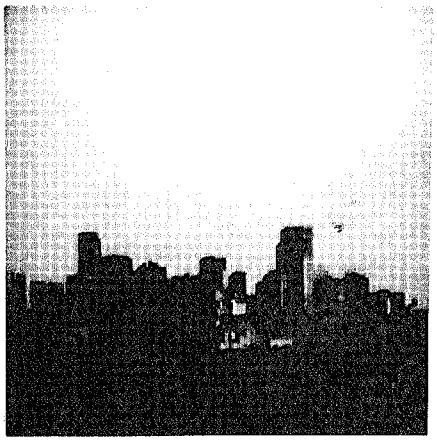
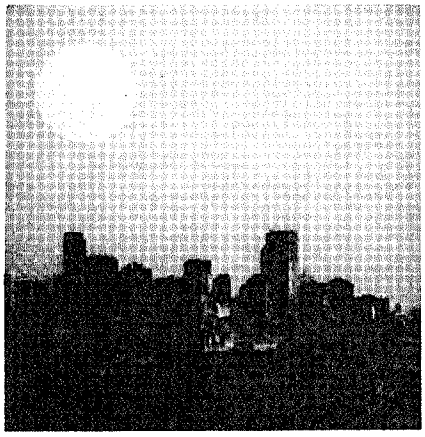


Figure 5-6
(a) Original image #1.

(b) Original image #2.

(c) Pixel-by-pixel addition of both images.



age addition operations are used for frame averaging in the reduction of random picture noise. Addition may also be used for the simple superimposing of two images; this is depicted in Figure 5-6. Subtraction techniques yield the ability to subtract out consistent background patterns and detect object motion from frame to frame. More on these operations may be found in Image Operation Studies #17, #18 and #19.

Pixel Group Processing

Pixel point processing allowed image gray-scale occupancy modification and image combination, which are both important image processing tools. What point operations did not allow was the spatial modification of scene detail within an image. Everything was handled pixel by pixel, with no interest in adjoining pixels. It turns out that when operating on any one particular pixel, adjoining pixels can give valuable information concerning brightness trends in the area being processed. These brightness trends open doors to the world of *spatial filtering.*

Earlier, the concept of spatial frequency was mentioned. Spatial frequency is the term used to define two-dimensional frequency. An image is said to be composed of many basic frequency subcomponents, ranging from high to low. Where rapid brightness transitions are prevalent, we have high spatial frequency. Slow transitions represent low frequency. Wherever a sharp edge is present—say, a transition from white to black within a one pixel distance—the highest frequencies in the image are found. Making use of this information, we may generate output images showing only the high-frequency or low-frequency components, a class of image processing known as spatial filtering. Additional spatial filtering operations make it possible to generate images that show only where individual sharp

transitions occur. These processes ultimately yield image edge detection and enhancement.

In dealing with spatial filtering, we talk about the *spatial convolution* operation. Convolution is a mathematical method used in signal analysis. Although the operation is mathematically complex, we may discuss it in an intuitive, pictorial manner. Spatial convolution is the method we use to calculate what is going on with the pixel brightnesses around the point of processing. As in point processing, we move across the image, pixel-by-pixel, placing a result at the same location in the output image as we are in the input image. It is the calculation that is different. The output pixel brightness becomes dependent on a group of pixels surrounding the pixel in which we are interested. By taking information about the center pixel's neighbors, we are able to calculate spatial frequency activity in the area and therefore are capable of making discretionary decisions regarding the area's spatial frequency content. Let us see how spatial convolution is carried out.

For every pixel in the input image, we calculate a value for the output image pixel by calculating a *weighted average* of it and its surrounding neighbors. The average is formed from a group of pixels, called a *kernel,* around, and including, the center pixel being processed. The dimensions are that of a square. The kernel may have the dimensions 1×1, which is the trivial case giving simply a point operation, 2×2, 3×3, and so on. The operation is said to increase in *degrees of freedom,* the larger the kernel size. This means that the flexibility of the spatial filter is increased by taking into account more neighboring pixels in the calculation. The accepted general-purpose kernel size is 3×3. This is because enough freedom is maintained and yet computation time is minimized.

The term *weighted average* is best described by first considering a conventional nonweighted average. As with any averaging of numbers, we add the numbers together and divide by the number of terms in the average. This gives us a single number based on the information present in all the numbers in the operation. For a 3×3 kernel, we would add the 9 pixel brightnesses together and divide the result by 9. A weighted average, on the other hand, is formed by attaching a multiplicative weighting factor to each term in the average. By altering these weighting factors, or *convolution coefficients,* certain pixels will have more or less influence on the overall average. In fact, correctly selecting the proper weighting coefficients allows us to carry out high and low pass image filters along with a variety of edge enhancement filters.

The mechanics of spatial convolution are fairly straightforward. In carrying out a 3×3 kernel convolution, nine weighting coefficients are defined and labeled *A* through *I.* This array of coefficients is called the *convolution mask.* Every pixel in the image is evaluated with its

eight neighbors, using this mask, to produce a resultant pixel value to be placed in the output image. A graphical representation of the operation is shown in Figure 5-7. The mask is placed over each input pixel. The pixel and its eight neighbors are multiplied by their respective weighting coefficients and summed. The result is placed in the output image at the same center pixel location. This operation ocurs for each pixel in the input image; in our case, 256 × 256 = 65,536 times. Each operation requires nine multiplications and nine additions. As we can see, a full image convolution requires on the order of a half-million multiplications and additions—not a quick process. Chapter 7 will show how special-purpose image processing hardware can drastically reduce this computational time.

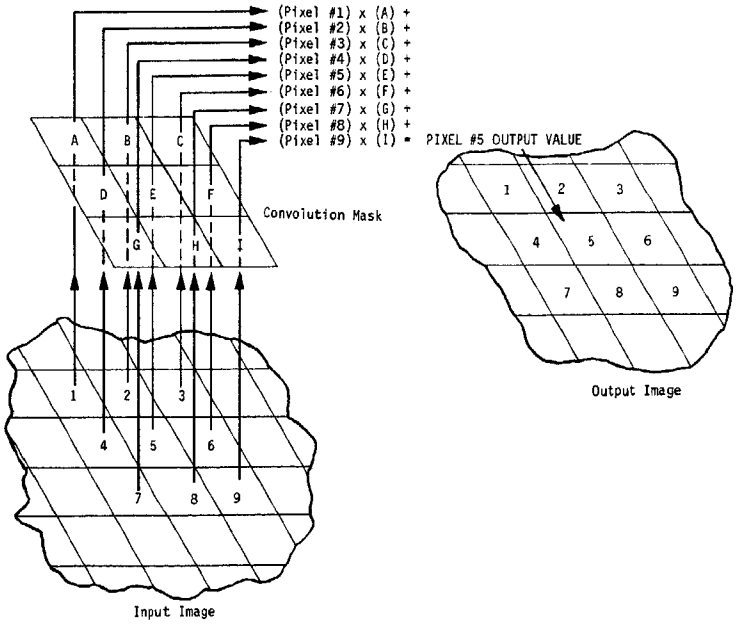


Figure 5-7 Spatial convolution calculation flow for pixel #5.

Research in image processing has yielded a variety of convolution masks for use as standard processing functions. Actually, the masks and the convolution process itself are no more than an application of standard mathematical functions in *linear system theory*. The mathematical foundations of linear system theory may be found in most texts dealing with electrical signal analysis. We will now review commonly used convolution masks and discuss how they work to effect our end goals in image processing.

SPATIAL FILTERING

The term *spatial filtering* implies the separation of frequency components within a two-dimensional base of data, or in our case, an image. The frequency components are spatial frequencies which relate to the rapidity of change in

gray levels over a certain spatial distance. Implementing a *high pass filter* will accentuate high-frequency details, leaving low-frequency details attenuated as a result. A *low pass filter* has the inverse effect. Edge enhancement operations are additional spatial filters with special edge detection properties. Since these filters play an important role in edge enhancement, they will be covered separately in the following section.

A spatial low pass filter has the effect of passing, or leaving untouched, low-spatial-frequency components of an image. High-frequency components are attenuated, leaving them virtually absent in the output image. A common low pass convolution mask is comprised of all nine coefficients having the value of 1/9:

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

Two aspects are immediately evident—the coefficients sum to 1 ($9 \times 1/9 = 1$) and they are all positive numbers. These two facts hold true for all low pass filter masks. Figure 5-8 illustrates a low passed image.

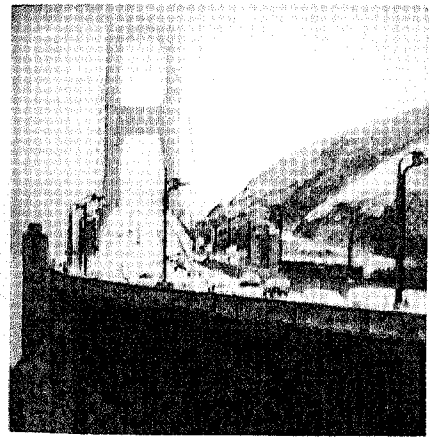
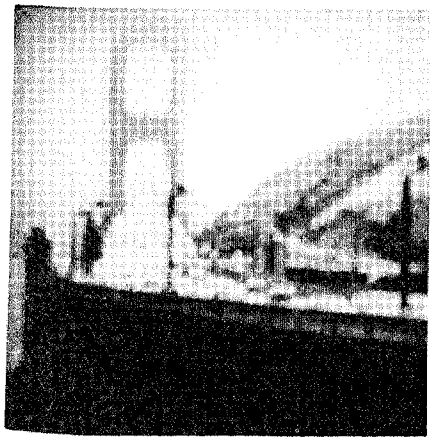


Figure 5-8
(a) Original image.

(b) Low pass filtered image.



To gain an intuitive feeling for how the low pass filter works, we may discuss the convolution output values as the mask is passed over regions of an image having different spatial frequency characteristics. If each pixel in a 3×3 group has the same brightness value, the result will be that of the constant pixel value. This correlates with the fact that there is a spatial frequency of 0 in the neighborhood—no gray level change at all. A frequency of 0 is the lowest possible and, of course, would be expected to be passed by the low pass filter unchanged. If the pixels in a neighborhood change rapidly from white to black every other location, the calculated output pixel value will be that of the average of all nine input values. As the mask moves over all pixels in a high-frequency area, they are replaced by their group averages, producing an output image that removes the high-frequency details. The visual effect is that of blurring. So we see that the output image is related to spatial frequency of the input image, slow-changing areas are left unchanged or changed slightly where fast-changing areas get averaged out to yield only the slow-changing aspects.

If we were to pass the low pass mask over an area with a single pixel width line having a constant background pixel brightness, we would expect the line to be blurred. This is because the line represents a high spatial frequency content. In fact, a sharply defined line actually is composed of frequency components spanning the spectrum of low to high frequencies. As the mask is moved over the line, pixels are replaced by the average of the bright line pixels and the constant background pixels. We

expect the resultant value to be somewhere between the two. The line becomes blurred into the background. Only the low-frequency components of the line remain at the end of the entire image convolution. The low pass filter is further discussed in Image Operation Study #6.

The low pass image filter becomes intuitively clear with a little studying. By running the mask over regions of varying details, we see the results corresponding with the passing of low-frequency details and the blocking of high-frequency details. All masks may be analyzed in this same intuitive manner.

Low pass filtering reduces high-frequency detail in an image. The effect is a mellowed, or slightly blurred, image. Low pass filtering allows analysis of low-frequency details of an image without the disruption of high-frequency details. It also plays an important role in the unsharp masking enhancement operation, discussed in Image Operation Study #9.

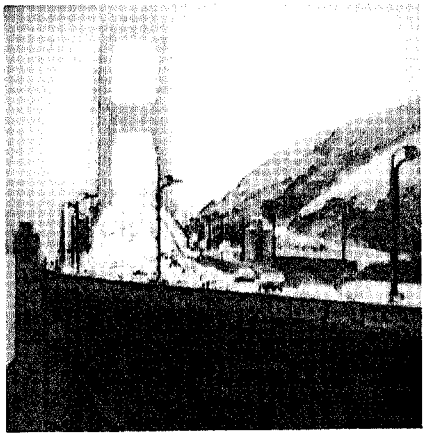
The high pass filter has basically the opposite effect of the low pass filter. It accentuates high-frequency spatial components while leaving low-frequency components untouched. A common high pass mask is comprised of a 9 in the center location with −1s in the surrounding locations:

$$\begin{matrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{matrix}$$

We note that the coefficients add to 1 and, furthermore, smaller coefficients surround the large positive center coefficient. A high passed image is illustrated in Figure 5-9.

The fact that the high pass mask contains a large positive coefficient in the center surrounded by smaller coefficients gives us a clue as to its operation. It tells us that the center pixel in the group of input pixels being processed carries a high weight whereas the surrounding ones act to oppose it. If the center pixel possesses a brightness vastly different from its immediate neighbors, the surrounding pixel effect becomes negligible and the output value becomes a brightened version of the original center pixel. The large difference indicates a sharp transition in gray level, and we would expect the high-frequency content transition to be accentuated in the output image. On the other hand, if the surrounding pixel brightnesses are large enough to counteract the center pixel's weight, the ultimate result is based more on an average of all pixels involved.

It may be interesting to note that if all pixels in a 3 × 3 group are equal, the result is simply the same value. This is equivalent to the low pass filter's response over constant regions. What this means is that this high pass filter does not attenuate low-frequency spatial components. Rather, it emphasizes high-frequency components while leaving low-frequency components untouched.



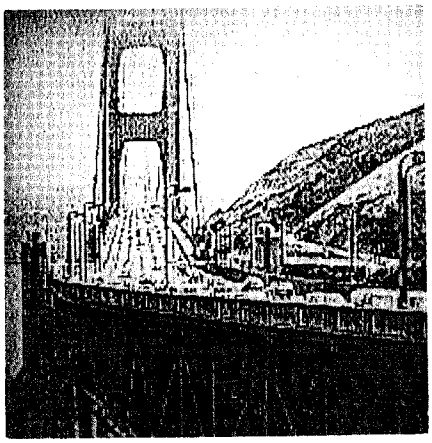


Figure 5-9
(a) Original image.

(b) High pass filtered image.

High pass filtering of an image adds accentuation to the edges, or transition areas, within it. This effect often gives the viewer a more pleasing image. Additionally, details muted by the background and low-frequency noise become apparent where they may have been barely visible in the original. Both high and low pass filter functions also fill important roles when used in conjunction with point processes. More on the high pass filter may be found in Image Operation Study #7.

EDGE DETECTION/ENHANCEMENT

Enhancement of edges in an image is an operation used in feature extraction, an important class of image processing. The operation basically reduces an image to display only its edge information. This information is then used for feature, or object, recognition by high-level algorithms. Additionally, a useful enhancement operation is carried out by adding the edge enhanced image back to the original using a dual image point process. The result is a crisper image displaying sharper edge detail.

Edge enhancements may be implemented through spatial filtering. Three particularly useful filters are found to be quite common in many image processing tasks. They are known as *shift and difference, gradient* and *Laplacian.* All three enhancements are based on the slope of pixel brightness occurring within a pixel group. To further define the term *slope* in an image context, think of the brightness of each pixel as being represented by a height coming out from the page toward the observer (see Figure 5-10). We see an image mound rather than the standard gray tone representation; the brighter the pixel, the higher the mound. By measuring the slope of the mound within any given pixel group, we have a value for how steep the incline is. A large value corresponds to a steep slope and means a large change in gray level. A small value indicates small slope which is equivalent to a small change in gray level. Since edges are, by definition, sharp brightness changes, large slopes indicate the presence of an edge.

The simplest edge enhancement operation is the shift and difference method. This procedure allows us to extract horizontal and vertical edge information. By shifting an image to the left by one pixel and then subtracting it from the original, vertical edges become apparent. On a pixel-by-pixel basis, we subtract the horizontal neighbor, giving a value of their brightness difference, or slope. Of course, a large difference is yielded by two adjacent pixels of greatly varying brightnesses. The result is an image appearing as an embossing (see Figure 5-11).

The analogous horizontal edge enhancement is implemented by shifting the image upward by one pixel and carrying out the subtraction. Since the success of all enhancements is evaluated subjectively, this edge enhance-

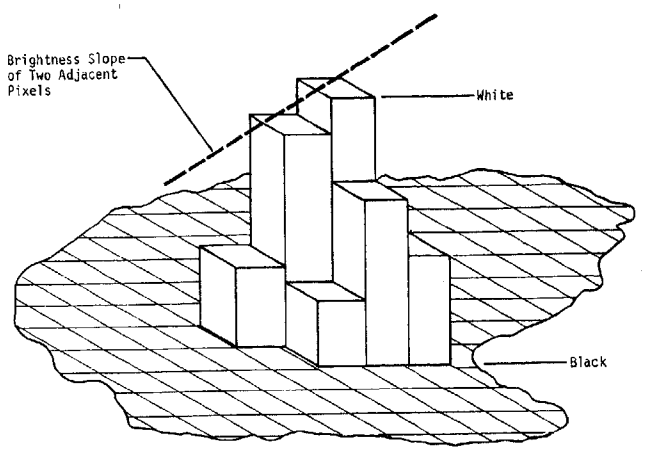


Figure 5-10 Pixel brightness represented by height, illustrating the concept of brightness slope.

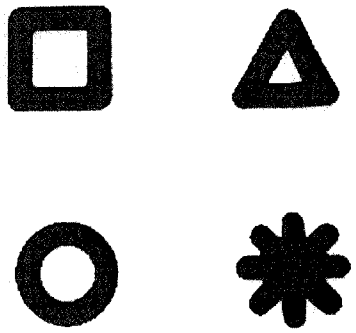
ment technique often proves valuable. The shift and difference operation may be carried out using a dual image subtraction process or a group process. Image Operation Study #10 discusses this operation in more depth.

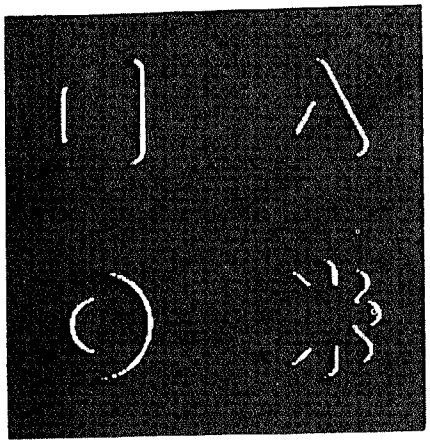
The gradient operation forms a directional edge enhancement. Using a 3 × 3 kernel, eight gradient images may be generated from an original. Each highlights edges oriented in one of the eight compass directions—N, NE, E, SE, S, SW, W, and NW. Figure 5-11 illustrates an East directional gradient. The mask oriented for the East direction is given.

$$\begin{matrix} -1 & 1 & 1 \\ -1 & -2 & 1 \\ -1 & 1 & 1 \end{matrix}$$

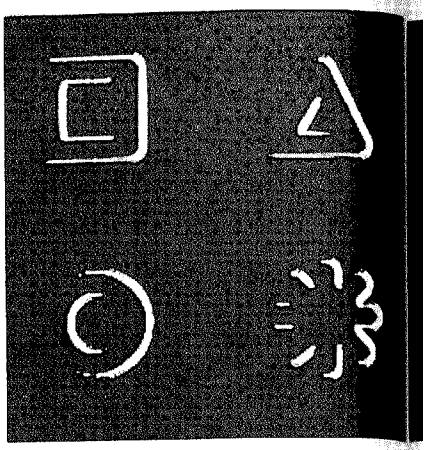
Note that the coefficients add to 0. This means that as the mask passes over a region of the image having a constant brightness, a result of 0 is produced. Of course, this represents a brightness slope of 0, which is exactly what a region of constant brightness has.

Using the East mask, a transition from dark to light, going left to right, will be accentuated. This is because a positive East brightness slope exists. Brightness slopes in other directions sum to a negative value, which is forced to 0, or black. The response of the gradient operation for a one-dimensional edge is seen in Figure 5-12. Where the gradient generates negative results, the output value is set to 0, since negative brightnesses are undefined. The gradient image appears black wherever the original image brightnesses are constant. Edges with the correct directional orientation in the original image are seen as white. Additional discussion of the gradient operation along with masks for all eight directional enhancements may be found in Image Operation Study #11.





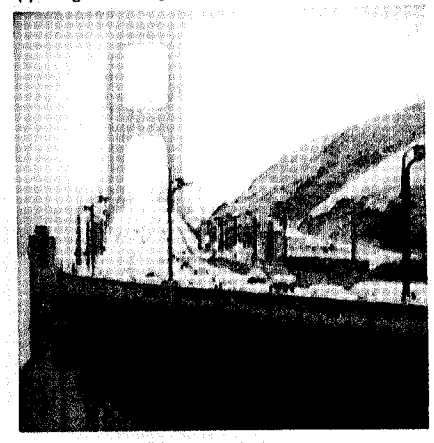
(b) Vertical shift and difference edge enhancement.



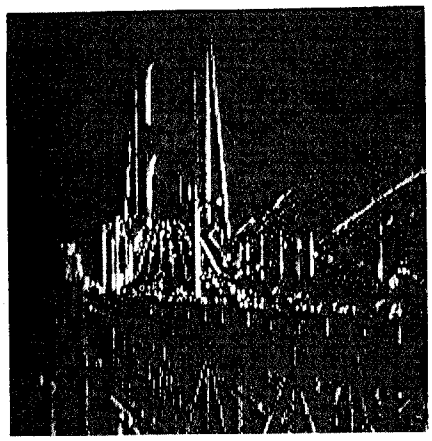
(c) East direction gradient edge enhancement.

Figure 5-11
(a) Original pattern image.

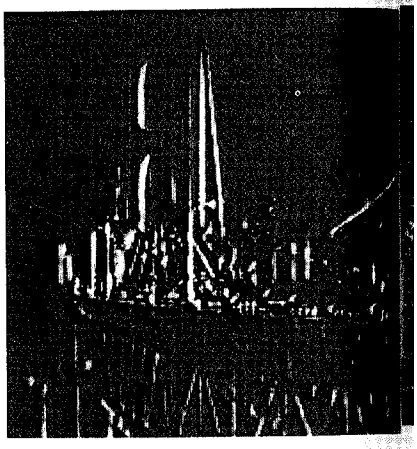
(f) Original image of bridge.



(g) Vertical shift and difference edge enhancement.



(h) East direction gradient edge enhancement.

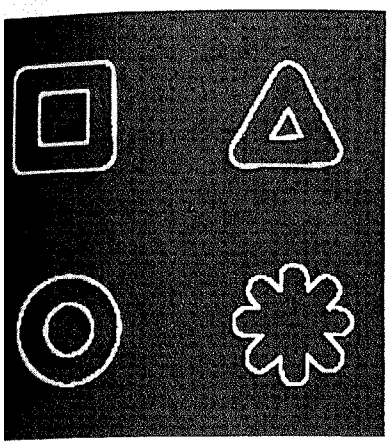


The Laplacian edge enhancement is an omnidirectional operation, highlighting all edges regardless of their orientation. This operation is based on the rate of change of the brightness slope within a 3 × 3 pixel group. The common Laplacian mask is comprised of an 8 in the center location with −1s in the surrounding locations.

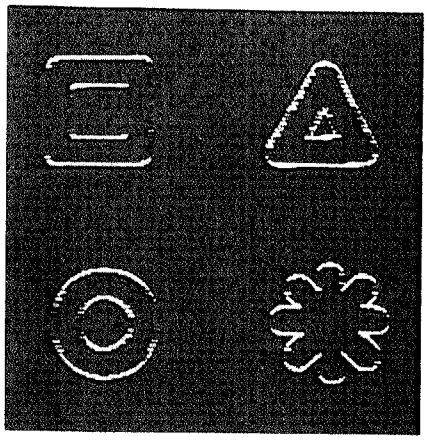
$$\begin{matrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{matrix}$$

The coefficients add to 0 and, as in the high pass filter mask, negative valued coefficients surround the large positive center coefficient. Figure 5-11 illustrates a Laplacian edge enhanced image.

The Laplacian enhancement generates sharper peaks at edges than does the gradient operation. Any brightness slope, whether positive or negative, is accentuated, giving

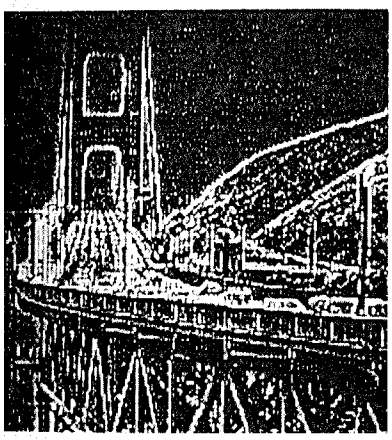


Laplacian edge enhancement.

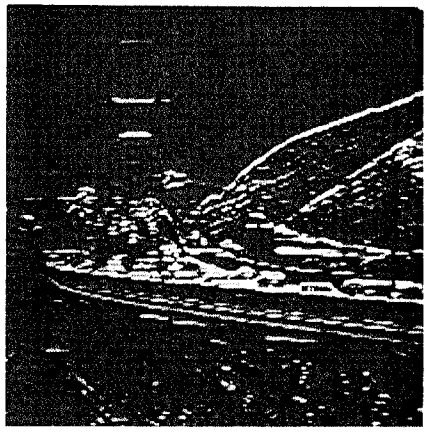


(e) Horizontal line detect.

aplacian edge enhancement.



(j) Horizontal line detect.



the Laplacian its omnidirectional quality. In the human visual system, the eye-brain system applies a Laplacian-like enhancement to everything we view. Because of this, a natural sharpening of images may be achieved by adding a brightness scaled Laplacian enhanced image to the original. The results of this procedure often produce a natural-looking sharpened image with subjectively pleasing qualities.

The Laplacian image appears black wherever the original brightnesses are constant or linearily changing. Edges made of nonlinear brightness transitions are highlighted as white. The Laplacian operation is discussed in additional depth in Image Operation Study #12.

The above methods for edge enhancements play a major role in machine vision. Whether the application is automated assembly-line material inspection or feature recognition, these processes are the first used to condition the raw images. Before a computer can actually attempt a

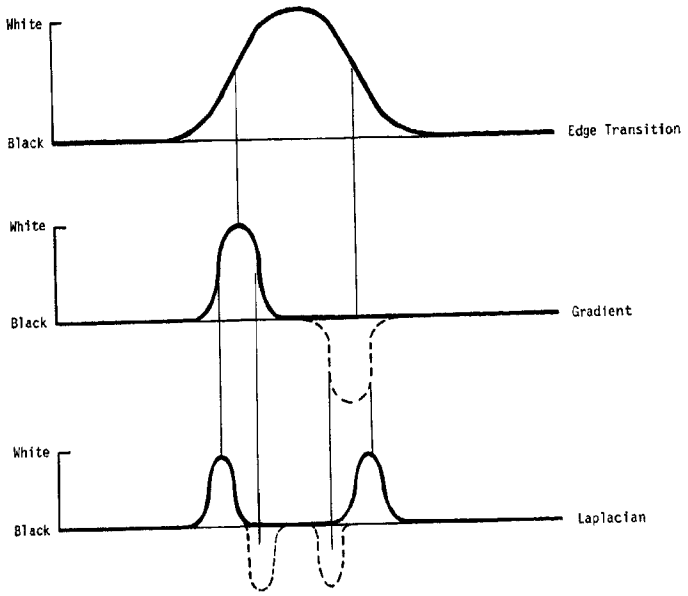


Figure 5-12 Response characteristics of the gradient and Laplacian edge enhancement operations.

recognition process, though, the image must generally be further processed. One such operation is the binary contrast enhancement, discussed earlier. Another operation often found useful is a *line segment enhancement.* A line enhancement operation emphasizes line segments within the image.

Figure 5-11 on pp. 54–55 illustrates an image processed by the horizontal line segment enhancement operation. The mask for this operation is given.

$$\begin{matrix} -1 & -1 & -1 \\ 2 & 2 & 2 \\ -1 & -1 & -1 \end{matrix}$$

Again, the coefficients add to 0. This tells us that constant brightness regions of the original image will become black when processed. Only line segments are left highlighted.

In factory process control applications, binary contrast enhancement is often evoked following the line segment enhancement. The ultimate result is an image more intelligible to computer object-recognition algorithms. More on line segment enhancement operations is found in Image Operation Study #13.

We have covered a section of image processing dealing with the enhancement of images based on brightness trend information within a 3×3 pixel group. Masks for these processes may be altered or entirely redefined to effect the user's end result. Especially powerful is the combination of point processing with group processing. The operations covered represent the most-used functions and serve as an introduction to the inner workings of group processing.

Frame Processing

Frame processing is the term given to the collection of image processing operations that do not fall within the confines of point or group processing. Frame processes encompass a wide variety of algorithms, each as different from another as point processes were from group processes. Because of this diversity, processing hardware within an image processing system rarely includes the capability of handling some, if any, frame processes. Instead the job is usually left to the host computer to be carried out in software. The actual processing implementation of these processes will be discussed in Chapter 7.

As would be expected, all image analysis and coding operations fall under the category of frame processing. Neither of these classes of image processing operations may be handled by the point or group processes. A common example of image analysis frame processing is that of the histogram, discussed in Chapter 4.

Frame processes are often time consuming, yet serve as useful and necessary functions for a variety of image processing applications. Three commonly used operations are overviewed here—geometric operations, image transforms, and data compression. Additionally, an operation known as the median filter is discussed in Image Operation Study #8.

GEOMETRIC OPERATIONS

Geometric operations, as applied to images, provide for the spatial reorientation of pixel data within an image scene. Pixel data from an input image may be transformed into new spatial locations, as defined by a geometric algorithm, producing a resulting image of altered characteristics. Geometric operations are often employed in image processing as a primary or ancillary function to processes from the point or group classes.

Three basic geometric processes allow for the sizing, orientation, and movement of images. They are image scaling, rotation, and translation. These operations permit the user to do simple pixel spatial transformation.

All geometric operations are performed by moving pixels from their original spatial coordinates in the input image to new coordinates in the output image. The general equation for these operations is

$$I[x,y] \rightarrow O[x',y']$$

where (x',y') are the transformed coordinates of the pixel brightness originally located at coordinates (x,y). Each geometric operation is, therefore, defined by a coordinate transform equation that defines the new x' and y' output coordinates for an input pixel at (x,y).

Image scaling deals with the enlarging and shrinking of an image or portion of an image. The general coordinate

transform equations for this function is given by the equations

$x' = Sx$ and
$y' = Sy$

where x and y are the coordinates of the input pixel being processed, x' and y' are the new output pixel coordinates and S is the scaling factor. S acts to modify the coordinates of the pixel brightness at (x,y) rather than the pixel brightness itself, as did the mapping function, M, in point processing. The scaling factor dictates the amount of magnification or demagnification to occur. For example, $S = 2$ represents a magnification of 2, where $S = 1/2$ would be a magnification of 1/2, or a shrinkage by a factor of 2. Figure 5-13 illustrates image magnification and shrinkage.

To see how the formula works, let us follow it through for a magnification of 2 applied to a 256 × 256 image. A pixel is retrieved from the input image at coordinates (x,y)—say location (67,67). The scaling factor acts to multiply both x and y by 2, yielding $(x'y')$ equal to (134,134). The original pixel brightness taken from location (67,67) in the input image is, therefore, placed at location (134,134) in the output image. Continuing the process across the input image line, we ultimately arrive at the pixel residing at (127,67). Applying the scaling factor, S, to the coordinates gives (254,134) as the new output coordinates. Incrementing to the next input pixel, (128,67), the scaling will yield output coordinates of (256,67)—out of range for a 256 × 256 output image. The processing of line 67 is complete. The pixel coordinates in line 67 have been expanded from between 0 through 127 to 0 through 254, a magnification of two. The analogous process happens on a line-by-line basis. In all, the pixel coordinates from pixel (0,0) to (127,127) are mapped to form a new output image filling the entire 256 × 256 image frame.

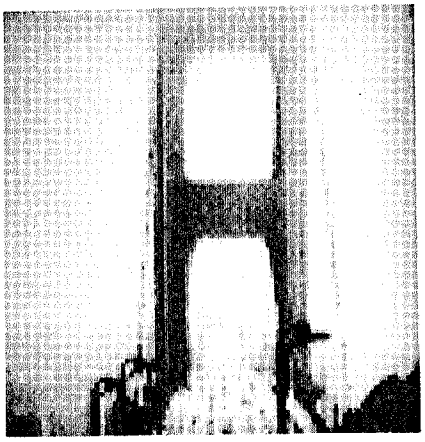
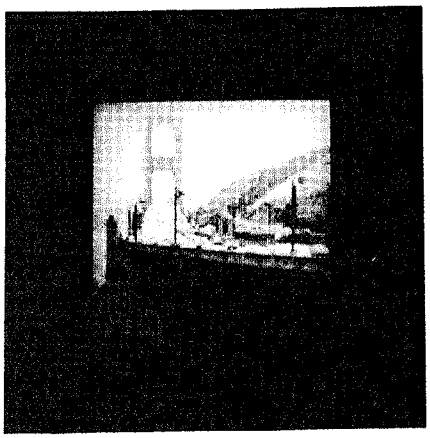


Figure 5-13
(a) Image magnification by factor of 2.

(b) Image shrinkage by factor of 2.



The obvious question is what happens to the odd lines and pixel locations in the output image, since nothing is directly mapped into them? Since we have effectively mapped a 128 × 128 image into a 256 × 256 image, there is no valid pixel information to be placed in the odd locations. To make the output image more appealing, however, the pixel to the immediate left is usually replicated into the odd pixel location. Likewise, the line of pixels above an odd line is replicated into the odd line below. The reduction in spatial resolution is the natural artifact of doing a magnification. In each axis, we have expanded half the pixel data into a full frame.

Image shrinking follows the same principles as magnification. In a shrinking by a factor of 2($S = 1/2$), the input image of size 256 × 256 will be mapped into an output image of size 128 × 128.

Scaling serves a variety of purposes in image processing. It may be used to simply crop a scene before further processing is evoked. For image composition, it allows several input images to be size adjusted before they are assembled into a final output image collage using a dual image point process image addition. Also, image registration between two input images in a dual point process may utilize the scaling operation. Image scaling is discussed further in Image Operation Study #14.

Image rotation provides the user with the ability to rotate images about a center point. The coordinate transform equations are

$$x' = x\cos\theta + y\sin\theta \text{ and}$$
$$y' = -x\sin\theta + y\cos\theta$$

where x and y are the input image pixel coordinates, x' and y' are the new output pixel location, (x',y'), and θ represents the angle of clockwise rotation of the image about the image center point. A θ of angle 0 to 360 degrees may be specified, allowing the rotation of an image through any required angle. Figure 5-14 illustrates an image rotated through an angle of 330°.

Figure 5-14 330° Image rotation.

The mathematics behind the rotation algorithm is an application of basic trigonometry. The derivation of the equations may be found in almost any text dealing with the subject. The center point of rotation is the center of the image and must therefore be defined as the pixel coordinate origin for this operation. Pixels then have the coordinates −127 to 128, going across the image from left to right. Likewise, lines from top to bottom have the coordinates −127 to 128. For the sake of following the process through, we will use the simple example of $\theta = 90°$. We calculate $\sin 90° = 1$ and $\cos 90° = 0$. The equations boil down to $x' = y$ and $y' = -x$. So plugging in the input pixel coordinates of (127,67), for example, yields the resulting output image coordinates of (67,−177). Applying the spatial transformation to all pixels within the input image will yield an output image rotated by 90°.

A concern in image rotation comes up when rotating an image by an angle that is not a multiple of 90°. As we saw in the example, the 90° rotation maps pixels one for one from the input to output image. This also holds true for 180° and 270° rotations for the $\sin\theta$ and $\cos\theta$ terms will be either 1 or 0. When rotating through an angle that is not a multiple of 90°, however, we have $\sin\theta$ and $\cos\theta$ terms that are fractional. As a result, the calculated output pixel coordinates rarely are integer numbers. So where does the pixel brightness go if (x',y') are not integer numbers? There are two ways to handle the problem. One is to place the pixel brightness at the nearest pixel location. This technique tends to turn straight lines into jagged lines when rotated, sometimes causing results disturbing to the viewer.

A second technique is called interpolation. A pixel falling between locations in the output image will always be somewhere in between four valid pixel locations. One form of interpolation divides the total pixel brightness into parts to be placed at the four valid pixel locations. The brightness division is determined by the distance that the transformed input pixel falls from each of the four output pixel locations. Since other pixels are also mapped to fractional pixel locations in the output image, there may be up to four input pixels contributing some brightness to any one output pixel location. Figure 5-15 shows this interpolation scheme.

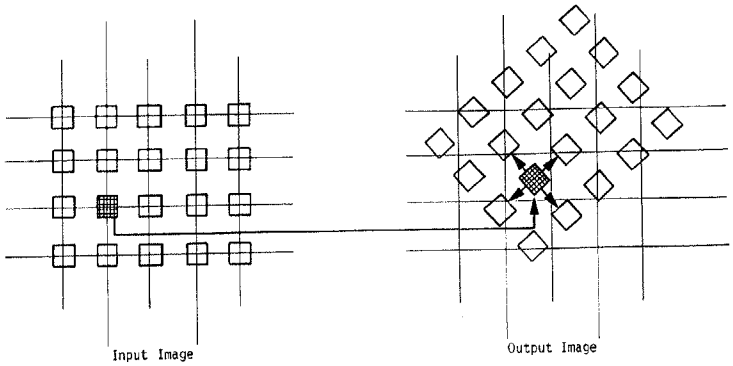


Figure 5-15 Pixel location interpolation.

Image rotation is used for many of the same reasons as scaling. The user is simply allowed an additional method of geometric manipulation for whatever reason may be appropriate. Image Operation Study #15 further discusses geometric rotation.

The last basic geometric operation is that of *image translation,* allowing the up-and-down, side-to-side movement of an image. The coordinate transform equations for image translation are given by the equations

$$x' = x + T_x \text{ and}$$
$$y' = y + T_y$$

where x and y comprise the input pixel coordinates, x' and y' form the output coordinates and T_x with T_y define the translation in the x and y directions. An input pixel is shifted side to side by the number of pixels indicated by T_x while the shift up and down is specified by T_y, thereby effecting the translation. All the input pixel coordinates are shifted by the amounts given by T_x and T_y.

For example, with $T_x = 10$ and $T_y = -20$, an input pixel at location (127,67) will be mapped to the location (127−20, 67+10), or (107,77), in the output image. When applied to all pixels in the input image, the net result will be an image moved to the right 10 pixels and up 20 pixels. Figure 5-16 shows a typical image translation.

Translation may be combined with scaling and rotation, netting the user the capability of total geometric image manipulation. This kind of image manipulation is useful in corrective geometric processing of many image scenes as a prelude to other operations. See Image Operation Study #16 for more on image translation.

A type of geometric transformation commonly used to correct spatially distorted images is known as *rubber sheet transformation.* This process may be thought of as working with an input image printed on a sheet of rubber. The rubber is then stretched and pinned down at selected points so that the original image is geometrically contorted to effect a desired end result. This type of geometric correction finds a variety of uses, for instance, spatial correction for an image sensor, an operation parallel to photometric correction.

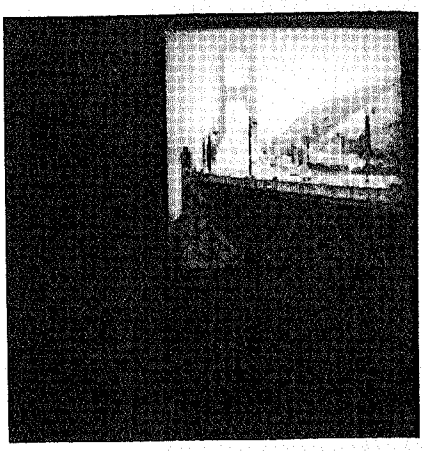


Figure 5-16 Image translation —64 pixels up, 50 pixels to right.

In an image sensor, there often exists some sort of spatial nonlinearity. This type of distortion is manifested as spatial bulges or contractions acting to distort the sensed image. These nonlinearities are usually slight and cause no problem to the user of the images. However, in certain applications—spaceborne imaging, for instance—weight and size constraints may force the design of an image sensor to be less than perfect. In these cases, the use of a rubber sheet transformation may be needed.

The basic approach to implementing a rubber sheet transformation is to define the mapping of input pixel locations to output pixel locations. This can be done with a massive spatial look-up table directly mapping input pixels into their new output locations. Implementing such a process can be very time consuming. Instead, we use a more general approach, still retaining a large degree of freedom in the definition of the transformation. Taking the rubber sheet image, we envision what areas of the image are to be stretched and to what degree. The stretching is then thought of as being carried out by pinning points of the sheet down. These points are called *control points.* By mathematically defining where the control points are located and what degree of stretching is to occur between them, we are able to calculate the input-to-output pixel transformation. Interpolation, as seen before, must also be used since the mapping will generally not be one to one.

The rubber sheet transformation idea may be extended to include the correction of viewing geometry problems and some sensor movement induced problems. Say, for example, an image is taken of the flat surface of an object at a perspective other than perpendicular to the surface plane. The resulting image is distorted in that square coordinate points on the surface appear quadrilateral in the image. By mapping the points of the imaged quadrilateral area into a square, we may recover the view of the area as it would have appeared in a perpendicular imaging ar-

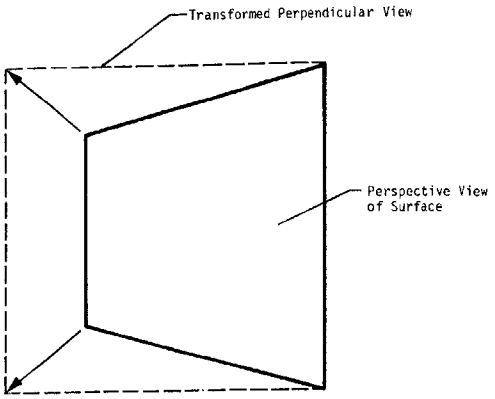


Figure 5-17 Rubber sheet transformation of quadrilateral area to square area.

rangement. Figure 5-17 illustrates this transformation. Likewise, mapping into another desired perspective angle is possible. Getting more involved is the perspective alteration necessary in correcting an image of a planetary object. In this case, the spherical curvature of the imaged surface must also be accounted for.

Some image sensor movement problems, as encountered in spacecraft, may also be handled by a rubber sheet transformation. In an imaging device that has a line of light sensors—common in Earth mapping satellites—we may experience sensor movement during the accumulation of enough lines to compose an entire image frame. This effect may, in some cases, yield an image of rectangular dimensions where the actual object area covered was square. We may use rubber sheet transformation to stretch out the shorter dimension of the rectangle and derive the original geometry of the object. Adding in movement that is not along the imaging axis adds to the complexity of this process.

All of the rubber sheet corrections reduce to straightforward mathematical models as long as the geometric distortion is precisely defined before processing commences. In the sensor spatial nonlinearity correction, the distortion may be characterized by imaging a square grid and deriving the proper equations to map whatever the sensor output image is into the original square grid pattern. From that point on, the geometric correction map will remain the same regardless of what is imaged. Likewise, as long as a spacecraft's movement is well defined and viewing perspective is known, the associated corrections pose no major difficulty.

We have discussed a variety of geometric corrections available to the user as needed. By picking and choosing the appropriate process or combination of processes, we have the flexibility to enhance geometrically poor images, making them more useful than they were in their original form.

TRANSFORMS

The discussion on spatial frequency content of an image allowed us to see that a scene is composed of varying spatial frequency components. Group processing operations then provided the ability to accentuate or attenuate certain frequency components of the image, depending on the coefficients used in the convolution mask—for instance, high and low pass filtering. Very powerful operations, known as *frequency transforms* can be used to give a pictorial view of the spatial frequency component breakdown of an image. Additionally, these transforms may aid in the specific filtering of undesired components.

An image scene is composed of two-dimensional spatial frequency components. These frequencies have varying orientation—horizontal, vertical, etc.—and are defined as having an amplitude and phase. It may be further stated that an image may be broken into these frequency components and then reconstructed from their subsequent summation.

A frequency transform gives us the ability to transform an image from the spatial domain to the spatial frequency domain and back again. Applying a frequency transform to an image yields a new image displaying the array of spatial frequencies, and their amplitudes and phases, present in the original image. The frequency image displays the presence of frequency components in an original image by the brightness of points at respective locations. Horizontal frequency is defined along the x-axis and vertical frequency along the y-axis. The brightness of a point in the image corresponds to the amplitude of the frequency component represented by the point's coordinates. Using this frequency domain image, we may easily analyze the spatial frequency content of an image.

Image filtering may be carried out in the frequency domain in a more intuitively straightforward manner than convolution in the spatial domain. To remove a particular frequency band from an image, we may simply set the corresponding area of that frequency image to zero, which removes those frequency components, and transform the frequency image back to the spatial domain. The drawback is in the two transforms that must be executed. Not only are these operations extremely computationally intensive in a time sense, but the cumulative mathematical errors can sometimes lead to problems.

The execution of a low pass filter on an image using the frequency transform method rather than spatial convolution proceeds as follows. A frequency transform is performed on the input image, yielding a spatial frequency image. The frequency image is then multiplied by a frequency mask image where the desired low-frequency components are equal to 1 and all others to 0. The resulting image is identical to the original frequency image in the

low-frequency region, but zero elsewhere; there are no longer any high-frequency components in the frequency image. This image is then inverse frequency transformed back to the spatial domain where we are left with a low passed version of the original image. High pass filtering is handled similarly by multiplying the frequency image by a mask image with the desired high-frequency regions equal to 1 and the low regions to 0. Of course, the frequency mask image may take on values other than just 0 and 1. By selecting appropriate numbers, the degree of frequency accentuation and attenuation may be controlled.

A great power of frequency transform filtering is the ability to do highly selective frequency filtering. For example, an image with a periodic noise, appearing as bands across the image, will have a frequency image with bright spots at the locations in the frequency domain where the noise frequency exists. By multiplying the frequency image by a mask that "zeros" the bright spot (thereby performing a band rejection filter at that frequency), the resulting spatial domain image will be devoid of the noise bands. An artifact of the filtration will be that any good image data comprised of the filtered spatial frequency will also be lost. However, the narrow frequency filtering allowed by the transform method will produce minimal disturbance to the rest of the image.

Frequency transforms come in a variety of forms. The distinguishing differences are the conventions used in which spatial frequencies are broken down into components. Although the exact transform type to be used is dependent on the application, the most common ones encountered are the Fourier, Hadamard, and Haar transforms. Derivations and applicability of these and other transforms may be found in texts dealing with signal processing and analysis.

Frequency transforms as applied to image processing can prove to be useful and sometimes invaluable tools. They are also time consuming, computationally intensive operations, generally leading to expensive implementation. Transforms are rarely supported on board image processing systems; they are most often handled by a host computer with fast numerical computation capabilities.

DATA COMPRESSION

Images of even moderate size are comprised of large amounts of data. Because of the ever-increasing desire to transmit and store images, it is common to code the data into a form less space consuming, thereby allowing speedier transmission or denser storage. There are numerous techniques currently employed to handle this type of coding, some more useful than others, depending on the application. Image data compression falls in the important class of image coding operations.

In an image of a natural scene, there will tend to be redundant information. When properly processed, this information may be reduced to a simpler form, producing an image requiring less data needed to describe it. To see this, we examine an image (such as Figure 3–3) from left to right along a single line. Gray levels go up and down. Except for areas composed of high spatial frequencies, these levels tend to change slowly or even remain constant over substantial lengths. Often, the majority of a line may be the same gray level. Using this knowledge, we may employ techniques of image coding that can be quite effective in the reduction of data necessary to reconstruct an original image. Two such methods in common use serve as good, easy-to-understand introductions to the area of image data compression. They are *run-length coding* and *differential pulse code modulation*, or DPCM.

Run-length coding may be implemented in a variety of ways; we will look at one method. On a line-by-line basis, we start at the beginning of a line. The brightness of the first pixel is noted. Traveling across the line, we count the number of subsequent pixels of the same brightness. When a different brightness is encountered, we place in the coded image file the constant brightness of the first group of pixels and how many there were. The process then repeats itself. For instance, if there were 53 pixels of the same brightness in the first group, they will be coded into two 8-bit binary numbers. The first number represents the brightness and the second represents the number of pixels in the length. The 53 8-bit pixels have been coded into two 8-bit numbers, a substantial data reduction.

Of course, if the brightness of pixels is changing every pixel location across the line, the run-length method will require twice the data storage of the original image itself. Remember, each time a brightness change is encountered, two 8-bit numbers are required to characterize the previous group of constant pixels, even if that group is only one pixel long. Modifications to the run-length algorithm may further enchance its characteristics, making it more efficient in high spatial frequency areas of an image.

A second coding method of interest is differential pulse code modulation, or DPCM. Instead of working with the principle that there will be long lengths of constant pixels, we assume that any adjacent pixel will tend to be near in brightness to its horizontal neighbor. As we travel across the line, the coded image file is loaded with the difference in brightness from one pixel to the next. Assuming that this change is never to be more than, say, eight gray levels, the coded change value need only be three bits long. Instead of storing the absolute brightness value of all pixels in an image, we simply store the change in brightness as we go from one pixel to the next. For 8-bit pixels; this technique represents an across the board data reduction of $(8 - 3)/8 = 63\%$.

An artifact of this coding technique shows up when the change in brightness from one pixel to the next is larger than eight gray levels. The DPCM method may require several pixel distances to settle in on the correct pixel brightness value after encountering a sharp change. The effect of this phenomenon manifests itself as an apparent horizontal low pass filtering of the image. By increasing the number of bits used to describe the pixel-to-pixel difference, this effect may be minimized. However, the improvement comes at the cost of data reduction efficiency.

Image coding techniques are far ranging. Often the algorithm to be used will be dependent on the type of images to be coded. For instance, both of the above methods work well except when dealing with images comprised largely of high-frequency components.

Implementation of image coding may often be handled in hardware as a final step in some sort of processing before the image is to be transmitted or stored. For general-purpose processors, however, the task is left to the host computer to be carried out in software.