CHAPTER 8

Geometric Operations

8.1 INTRODUCTION

Geometric operations change the spatial relationships among the objects in an image. Such operations may be thought of as moving things around within the image. The effect is the same as printing the image on a rubber sheet, stretching the rubber sheet, and tacking it down at various points. Actually, a geometric operation is much more general than that, since any point in the input image may move to any position in the output image. Such an unconstrained geometric operation would almost certainly scramble the image content, so geometric operations are generally constrained to preserve some semblance of order.

Two separate algorithms are required for a geometric operation. First, there must be an algorithm that defines the spatial transformation itself. This specifies the "motion" of each pixel as it "moves" from its initial to its final position in the image. Also required is an algorithm for gray-level interpolation. This is necessary because, in general, integer x, y positions in the input image map to fractional (noninteger) positions in the output image and conversely.

8.1.1 The Spatial Transformation

In most applications, it is desirable to preserve the continuity of curvilinear features and the connectivity of objects within the image. A less constrained spatial transformation algorithm would break up lines and objects and tend to "splatter" the contents of the image.

One could exhaustively specify the motion of each pixel in the image, but this would quickly become unwieldy, even for small images. It is more convenient to specify mathematically the spatial relationship between points in the input image and points in the output image. The general definition for a geometric operation is

$$g(x, y) = f(x', y') = f[a(x, y), b(x, y)] \tag{1}$$

where $f(x, y)$ is the input image and $g(x, y)$ is the output image. The functions $a(x, y)$ and $b(x, y)$ uniquely specify the spatial transformation. If they are continuous, connectivity will be preserved within the image.

8.1.2 Gray-Level Interpolation

The second requirement for a geometric operation is an algorithm for the interpolation of gray-level values. In the input image $f(x, y)$, the gray-level values are defined only at integral values of x and y. Eq. (1), however, will in general dictate that the gray-level value for $g(x, y)$ be taken from $f(x, y)$ at fractional (nonintegral) coordinate positions. If the geometric operation is considered a mapping from f to g, pixels in f can map to positions between pixels in g and vice versa. For the purposes of this discussion, we stipulate that pixels be located exactly at integral coordinates of the sampling grid.

Armed with a spatial transformation and an algorithm for gray-level interpolation, we are prepared to perform a geometric operation. Usually, the gray-level interpolation algorithm is permanently established in the computer program. The algorithm defining the spatial transformation, however, is specified uniquely for the task at hand. Since the gray-level interpolation algorithm is always the same, or one of several options, it is the spatial transformation that defines a particular geometric operation.

8.1.3 Implementation

One can adopt either of two approaches when implementing a geometric operation. One can think of the operation as transferring the gray levels from the input image to the output image, pixel by pixel. If an input pixel maps to a position between four output pixels, then its gray level is divided among the four output pixels according to the interpolation rule. We call this the *pixel carry-over* or *forward-mapping* approach. (See Figure 8–1.)

An alternative, and more effective, implementation is achieved by the *pixel-filling* or *backward-mapping* algorithm. In this case, the output pixels are mapped back into the input image, one at a time, to establish their gray levels. If an output pixel falls between four input pixels, its gray level is determined by gray-level interpolation (Figure 8–1). The backward spatial transformation is the inverse of the forward transformation.

The forward-mapping algorithm is somewhat wasteful, since many input pixels might map to positions outside the border of the output image. Furthermore, each output pixel might be addressed several times, with many input pixels contributing to its final gray-level value. If the spatial transformation involves demagnification, more than four input pixels would contribute. If magnification were involved, certain of the output pixels might be missed when no input pixels mapped to positions near their location.

The backward-mapping algorithm, however, generates the output image pixel by pixel, line by line. The gray level of each pixel is uniquely determined by one interpolation step between, at most, four input pixels. The input image, of course, must be accessed randomly in a manner defined by the spatial transformation, and this can be quite complex. Nevertheless, the pixel-filling approach is the more practical algorithm for general use.

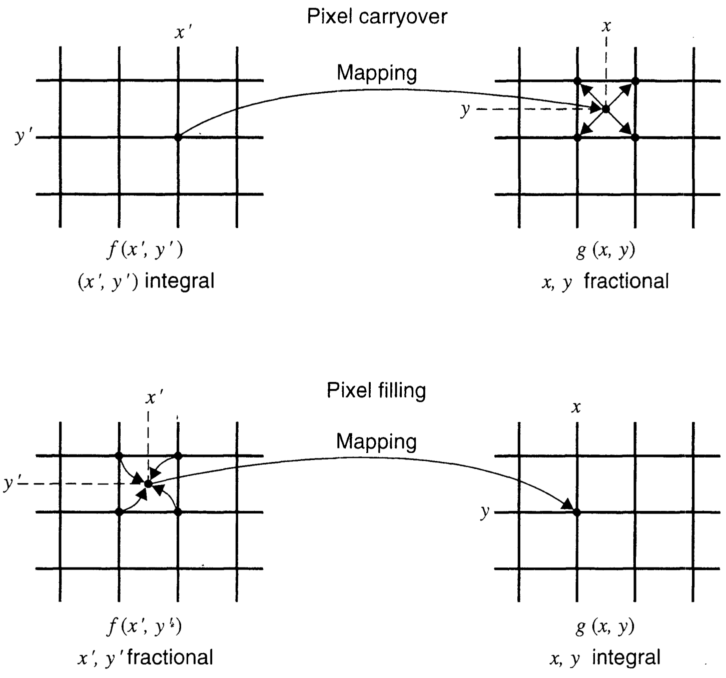


Figure 8–1 Pixel transfer

8.2 GRAY-LEVEL INTERPOLATION

Since output pixels map to fractional positions in the input image, they generally fall into the space between four input pixels. Interpolation is then necessary to determine what gray level corresponds to that position.

8.2.1 Nearest Neighbor Interpolation

The simplest interpolation scheme is the so-called zero-order, or *nearest neighbor,* interpolation. In this case, the gray level of the output pixel is taken to be that of the input pixel nearest the location to which the output pixel maps. This is computationally simple and produces acceptable results in many cases. However, nearest neighbor interpolation can introduce artifacts in images containing fine structure whose gray level changes significantly from one pixel to the next. Figure 8–2 shows an example of rotating images with nearest neighbor interpolation, with the resulting sawtooth effect at some of the edges.

8.2.2 Bilinear Interpolation

First-order, or *bilinear,* interpolation produces more desirable results than does zero-order interpolation, with only a slight increase in programming complexity and execution time. Since fitting a plane through four points is an overconstrained problem, first-order interpolation on a rectangular grid requires the bilinear function.

Let $f(x, y)$ be a function of two variables that is known at the vertices of the unit square. Suppose we desire to establish by interpolation the value of $f(x, y)$ at an arbitrary

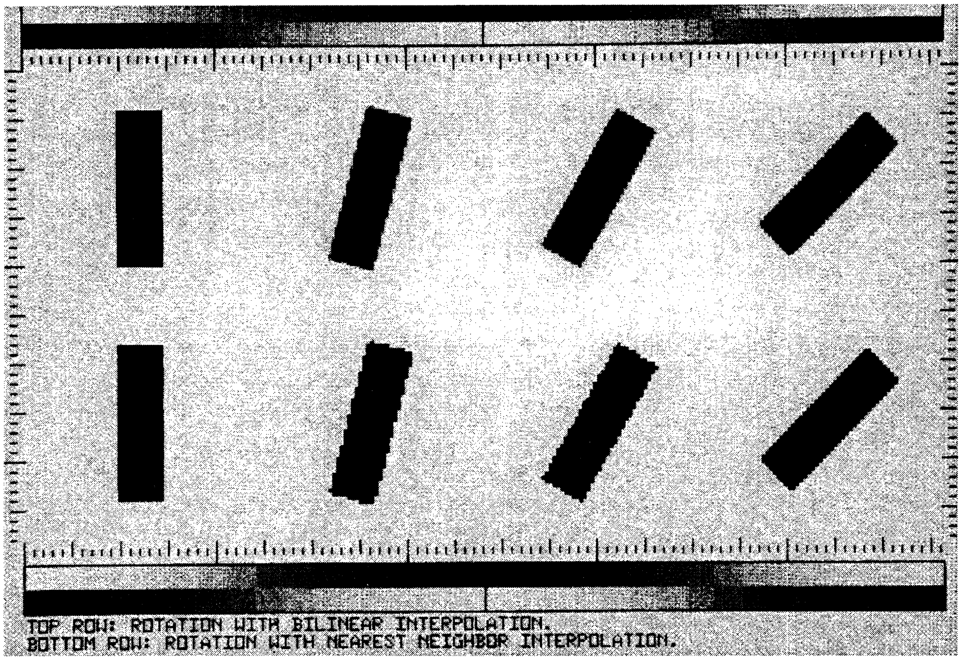

TOP ROW: ROTATION WITH BILINEAR INTERPOLATION.
BOTTOM ROW: ROTATION WITH NEAREST NEIGHBOR INTERPOLATION.

Figure 8–2 Comparison of zero-order and first-order gray level interpolation

point inside the square (Figure 8–3). We can do so by fitting a hyperbolic paraboloid, defined by the bilinear equation

$$f(x, y) = ax + by + cxy + d \tag{2}$$

through the four known values.

The four coefficients, a through d, are to be chosen so that $f(x, y)$ fits the known values at the four corners. There is a simple algorithm that produces a bilinear interpolation function

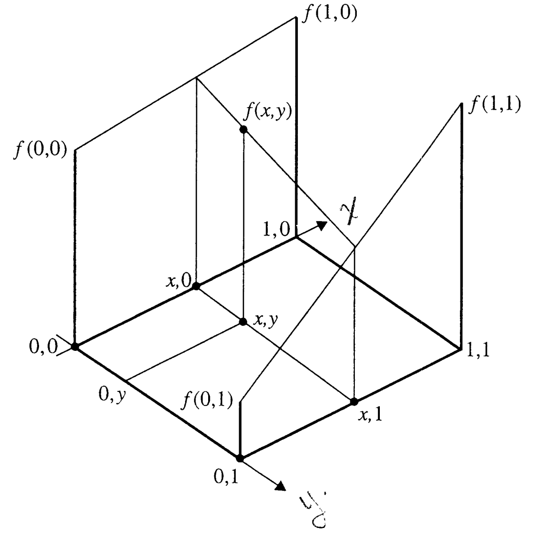


Figure 8–3 Bilinear interpolation

which fits $f(x, y)$ at the corners. First, we linearly interpolate between the upper two points to establish the value of

$$f(x, 0) = f(0, 0) + x\,[f(1, 0) - f(0, 0)] \tag{3}$$

Similarly, for the two lower points,

$$f(x, 1) = f(0, 1) + x\,[f(1, 1) - f(0, 1)] \tag{4}$$

Finally, we linearly interpolate vertically to determine the value of

$$f(x, y) = f(x, 0) + y\,[f(x, 1) - f(x, 0)] \tag{5}$$

Substituting Eqs. (3) and (4) into Eq. (5), expanding, and collecting terms produces

$$\begin{aligned} f(x, y) = {} & [f(1, 0) - f(0, 0)]\,x + [f(0, 1) - f(0, 0)]\,y \\ & + [f(1, 1) + f(0, 0) - f(0, 1) - f(1, 0)]\,xy + f(0, 0) \end{aligned} \tag{6}$$

which is in the form of Eq. (2) and is thus bilinear. Upon inspection, it is clear that Eq. (6) fits the four known values of $f(x, y)$ at the corners of the unit square.

Notice that if we hold either x or y constant, Eq. (2) becomes linear in the other variable. This illustrates that the hyperbolic paraboloid is a two-way ruled surface; that is, it intersects all planes parallel to the xz-plane and all planes parallel to the yz-plane in a straight line.

Bilinear interpolation can be implemented either directly, by Eq. (6), or by performing the triple linear interpolation given by Eqs. (3), (4), and (5). Since Eq. (6) involves four multiplications and eight additions or subtractions, geometric transformation programs typically do the latter, which requires only three multiplications and six additions or subtractions.

Although the foregoing development was performed on the unit square, it is easily generalized by an integer translation, after which x and y represent the fractional pixel position. Figure 8–2 compares bilinear with nearest neighbor interpolation.

When adjacent four-pixel neighborhoods are interpolated with the bilinear equation, the resulting surfaces match in amplitude at the neighborhood boundaries, but do not match in slope. Thus, a surface generated by piecewise bilinear interpolation is continuous, but its derivatives, in general, are discontinuous at the neighborhood boundaries.

8.2.3 Higher Order Interpolation

In geometric operations, the smoothing effect of bilinear gray level interpolation may degrade fine detail in the image, particularly if magnification is involved. In other applications, the slope discontinuities of bilinear interpolation may produce undesirable effects. In either of these cases, the extra computational efforts of higher order interpolation may be justified. A function similar to, but more complex than, Eq. (2) and having more than four coefficients is made to fit through a neighborhood of more than four points.

If the number of coefficients equals the number of points, the interpolating surface can be made to fit at every point. If the points outnumber the coefficients, a curve-fitting or error-minimizing procedure can be used. Examples of higher order interpolating functions are cubic splines, Legendre centered functions, and the function $\sin(\alpha x)/\alpha x$. The latter is discussed in later chapters. Higher order interpolation is usually implemented by convolution. A discussion of this is reserved for Part 2 of the text.

8.3 THE SPATIAL TRANSFORMATION

Eq. (1) gives the general expression for the spatial transformation. It is instructive to consider some less complex special cases before going on to general geometric operations.

8.3.1 Simple Transformations

If we let

$$a(x, y) = x \qquad b(x, y) = y \tag{7}$$

in Eq. (1), we have the identity operation, which merely copies f into g without modification.

If we let

$$a(x, y) = x + x_0 \qquad b(x, y) = y + y_0 \tag{8}$$

we have the translation operation, in which the point x_0,y_0 is translated to the origin, and features within the image are moved by an amount $\sqrt{x_0^2 + y_0^2}$. Using the formulation called *homogeneous coordinates* [1–9] we can consider the x–y plane to be the $z = 1$ plane of three-dimensional x, y, z space and write Eq. (8) compactly in matrix form as

$$\begin{bmatrix} a(x,y) \\ b(x,y) \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & x_0 \\ 0 & 1 & y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{9}$$

Letting

$$a(x, y) = x/c \qquad b(x, y) = y/d \tag{10}$$

will magnify the image by the factors c in the x-direction and d in the y-direction. The origin of the image (typically the upper left-hand corner) remains stationary as the image "expands." In homogeneous coordinates Eq. (10) is written as

$$\begin{bmatrix} a(x,y) \\ b(x,y) \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{c} & 0 & 0 \\ 0 & \frac{1}{d} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{11}$$

Letting $c = -1$ produces a reflection about the y-axis,

$$a(x, y) = -x \qquad b(x, y) = y \tag{12}$$

and similarly for d and the x-axis.

Finally, letting

$$a(x, y) = x\cos(\theta) - y\sin(\theta) \tag{13}$$

and

$$(x, y) = x\sin((\theta) + y\cos(\theta) \tag{14}$$

produces a rotation through an angle θ about the origin. This equation can be written in homogeneous coordinates as

$$\begin{bmatrix} a(x,y) \\ b(x,y) \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{15}$$

Clearly, we can combine translation with magnification to cause the image to "grow" about a point other than the origin. Likewise, we can combine translation with rotation to produce rotation about an arbitrary point.

Homogeneous coordinates provide a simple way to determine the formulas for compound transformations. For example, rotation about the point x_0,y_0 is accomplished by

$$\begin{bmatrix} a(x,y) \\ b(x,y) \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & x_0 \\ 0 & 1 & y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -x_0 \\ 0 & 1 & -y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{16}$$

The image is first translated so that the point x_0,y_0 is at the origin, then rotated through the angle θ, and then translated back to its origin. Multiplying out Eq. 16 yields the appropriate transformation equations. Other compound transformations can be constructed similarly. In the construction of the right-hand side of the equation, the sequence of operations is from left to right.

Separable Implementations. If an image is subjected to translation [Eq. (8)] or magnification [Eq. (11)], the output pixel addresses, $a(x, y)$ and $b(x, y)$, depend only on x and y, respectively. Thus, it is possible, and sometimes more efficient, to perform the operation in two steps. First it is done, for example, in the horizontal direction, producing an intermediate image. Then the vertical part of the operation proceeds, using the intermediate image as its input and producing the final result.

Catmull and Smith [10] have shown that it is possible to perform a rotation in the same type of two-step procedure. Solving for x in Eq. (13) yields

$$x = \frac{a(x,y) + y\sin(\theta)}{\cos(\theta)} \tag{17}$$

and substituting this into Eq. (14) leads to

$$b(x,y) = \frac{a(x,y)\sin(\theta) + y}{\cos(\theta)} \tag{18}$$

Thus, we can use Eq. (13), which is linear in x along any scan line, in combination with $b(x, y) = y$ in the first (horizontal-only) part of the operation. Then we can use Eq. (18), which is linear in y along any column, along with $a(x, y) = x$ in the second (vertical-only) part of the operation.

In this type of rotation, image features are "compressed" in the x-direction by the factor $\cos(\theta)$ in the first step, and then "expanded" in the y-direction in the second step. The technique fails at multiples of 90 degrees, where the cosine goes through zero, and inaccuracy restricts it to smaller angles.

For image registration applications, the required rotation angles are normally small. Even if this is not the case, rotation through multiples of 90 degrees can be done with simple row and column swapping. Thus, it is possible to rotate an image through any angle while keeping the actual rotation angle between plus and minus 45 degrees and the compression factor no less than 0.707. With this restriction, then, translation, magnification and rotation have one-dimensional implementations.

8.3.2 General Transformations

For relatively simple spatial transformations, it may be practical to use an analytic expression for Eq. (1). In many image-processing applications, however, the desired spatial transformation is relatively complex and not amenable to convenient mathematical expression. Furthermore, the desired pixel translations are frequently obtained from measurement of actual images, and it is desirable to specify the geometric transformation in these terms rather than in functional form.

An example of this is the geometric calibration of an image taken with a camera having geometric distortion. First, a rectangular grid target is digitized and displayed. Because of geometric distortion in the camera, the displayed grid pattern will not be exactly rectangular. (See Figure 8–4.) The desired spatial transformation is that which makes the grid pattern rectangular again, thereby correcting the distortion introduced by the camera. This same spatial transformation can then be used on subsequent images digitized by the same camera (assuming that the distortion is not scene dependent), thereby producing undistorted images.

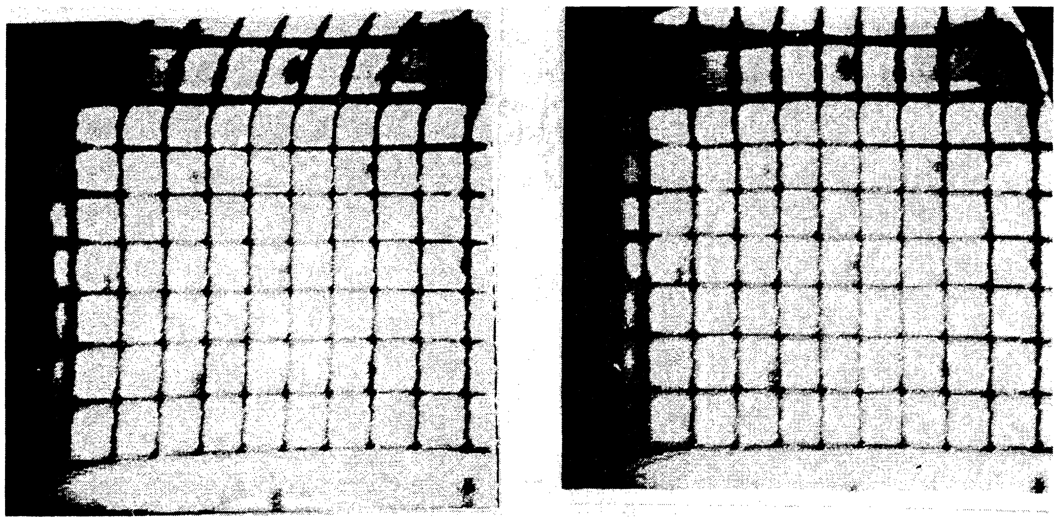


Figure 8–4 Geometric calibration of an early Ranger spacecraft camera: (a) before, (b) after (Courtesy NASA-JPL, from [19])

8.3.3 Specification by Control Points

It is convenient to specify the spatial transformation as a series of displacement values for selected *control points* in the image. Since only a small fraction of the pixels are actually specified, the displacements of noncontrol points must be determined by interpolation.

One way to do this is to develop functional expressions for $a(x, y)$ and $b(x, y)$ in Eq. (1). Commonly, a polynomial is used as the general form of the transformation expression. Its parameters are selected to make it fit the control points and their specified displacements. This is called *polynomial warping*. It is practical to use polynomials up to the fifth order for the transformation function [11].

In many cases, the limitations of polynomial warping will not accommodate the complex transformation required. Thus, some programs for geometric operations break the image up into polygonal regions and use piecewise bilinear mapping functions. The user specifies an input *control grid* made up of control points that form the vertices of contiguous quadrilaterals in the input image [11–16]. The input control grid maps to a grid of contiguous, horizontally oriented rectangles in the output image (Figure 8–5). The vertices (input control points) of the quadrilateral map directly to the corresponding vertices of the rectangle. Similarly, points inside an input quadrilateral map to points within the corresponding output rectangle.

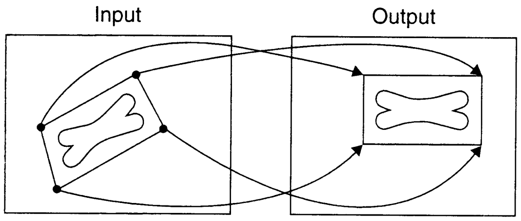


Figure 8–5 Spatial mapping of control points

8.3.4 Polynomial Warping

If the number of terms in the polynomial matches the number of control points, then the transformation can be designed to map the control points exactly as specified. Solving for the coefficients of the polynomial becomes an exercise in simultaneous linear equations, and a matrix inversion will normally produce the required result. (See Sec. 19.5.2.)

If there are more control points than terms in the polynomial, however, a fitting procedure must be used to determine the coefficients of the polynomial. In this case, the spatial transformation is a *best fit* to the control point specifications, and the mapping of individual control points does not occur exactly as specified.

Techniques for fitting one- and two-dimensional functions to a set of given data points are discussed in Section 19.5. The *pseudoinverse* technique (Sec. 19.5.2, Appendix 3) for determining the coefficients of the best fitting function is commonly used for polynomial warping. Other numerical methods, such as singular value decomposition ([17], Appendix 3) and orthonormal decomposition [11] may prove superior in practice.

Once the coefficients of the polynomial have been determined, the implementation is the same as before. There are numerical methods, however, such as Horner's nesting scheme [11,18], that can reduce the required number of computational steps. Even so, the task can be formidable when performing higher order warps on large images.

8.3.5 Control Grid Interpolation

If polynomial warping is impractical, the image must be warped in pieces. In the most common implementation, the input control points form a grid that maps to a grid of contiguous, horizontally oriented rectangles in the output image, as in Figure 8–5. The input control points map to the vertices of the corresponding rectangles, while points inside each input polygon map to points within the corresponding output rectangle.

Bilinear interpolation is a common choice for control grid interpolation, because it is computationally simple and produces a smooth mapping that preserves continuity and connectivity. The general expression for the bilinear spatial transformation is

$$G(x, y) = F(x', y') = F(ax + by + cxy + d, ex + fy + gxy + h) \tag{19}$$

The bilinear transformation is defined by the values of the eight coefficients a through h. By specifying that the four vertices of a quadrilateral map to the four vertices of the corresponding rectangle, we create two sets of four linear equations in four unknowns. The mapping from x' to x generates four equations in a, b, c, and d, and likewise for the mapping from y' to y and the coefficients e, f, g, and h. These sets of equations may be solved for a through h [recall Eq. (6)] to specify the bilinear spatial transformation algorithm that applies to all output points falling inside the rectangle.

While the spatial transformation algorithm could be implemented as Eq. (19), there is a more convenient and computationally efficient way of implementing it. By redefining the coefficients a and e, we can write Eq. (19) as

$$G(x, y) = F[x + dx(x, y), y + dy(x, y)] \tag{20}$$

where $dx(x, y)$ and $dy(x, y)$ are pixel displacements that are bilinear functions of x and y. Figure 8–6 shows these displacements with the input quadrilateral superimposed upon the output rectangle to which it maps. The problem now reduces to specifying dx and dy for all points inside the rectangle. Since $dx(x, y)$ and $dy(x, y)$ are bilinear in x and y, they become linear in x along each output line. Thus, for each line, we can define an increment, Δx, such that, assuming unit pixel spacing,

$$dx(x + 1, y) = dx(x, y) + \Delta x \tag{21}$$

and similarly for dy. The increment Δx changes from line to line, but is easily computed from the displacement values at the ends of the output rectangle. These can be interpolated between the given displacements at the vertices. Implementing Eq. (21) requires only two additions, one for dx and one for dy, at each output pixel to compute the coordinates of the corresponding input point.

The foregoing procedure specifies the spatial transformation for points falling inside the output rectangle. Frequently, a single quadrilateral-to-rectangle mapping is inadequate to specify the desired spatial transformations, and one can designate contiguous sets of quadrilaterals in the input image that map into contiguous sets of rectangles in the output image. It is not necessary, however, for the rectangles to cover the output image completely.

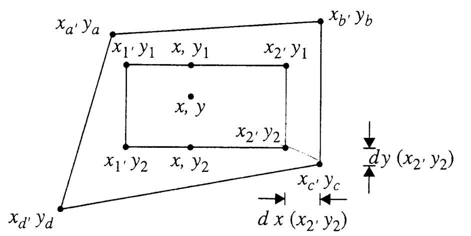


Figure 8–6 Control point displacements

Figure 8–7 shows an output image in which six contiguous rectangles are defined. Inside each of the rectangles, the spatial transformation is defined as described above. The figure also shows how the spatial transformation can be extrapolated outside the rectangles by which it is defined. The numbers inside the unspecified (dotted) rectangles indicate the control rectangles from which the bilinear coefficients are used [12]. For example, the spatial transformation used in the upper left-hand rectangle of the output image uses the bilinear coefficients for rectangle 1.

It is clear from the previous discussion that the bilinear transformation is continuous and unique at the vertices and boundaries of output rectangles. At each boundary, bilinear interpolation degenerates into linear interpolation between the two end points.

When specifying adjacent rectangles in the output image, one must make their vertices coincident. Similarly, adjacent quadrilaterals in the input image must have coincident vertices. Nonadjacent quadrilaterals, however, are not so constrained and may even overlap. Objects inside areas where input quadrilaterals overlap become duplicated in the output image.

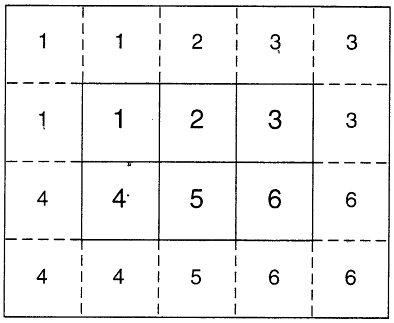


Figure 8–7 Control grid extrapolation

8.4 APPLICATIONS OF GEOMETRIC OPERATIONS

8.4.1 Geometric Calibration

An important application of geometric operations is the removal of camera-induced geometric distortion from digital images [13–16,19]. An example appears in Figure 8–4. Geometric calibration has proved important in extracting quantitative spatial measurements from a wide variety of digitized images. Certain images, such as those from satellites and airborne side-looking radar, are subject to rather severe geometric distortions. These images often require geometric correction prior to interpretation.

8.4.2 Image Rectification

Some imaging systems use non-rectangular pixel coordinates. Before images digitized with such a system can be viewed properly on ordinary display systems, they must be rectified, that is, transformed into rectangular pixel coordinates.

The Viking Lander spacecraft, for example, used an angle-scanning camera designed for digitizing Martian panoramas. It used a spherical coordinate system with scan lines spaced at equal angles of elevation, and its pixel spacing represented equal increments of azimuth angle. Figure 8–8(a) shows the distortion this design produced on rectangular displays, particularly for objects located near the camera.

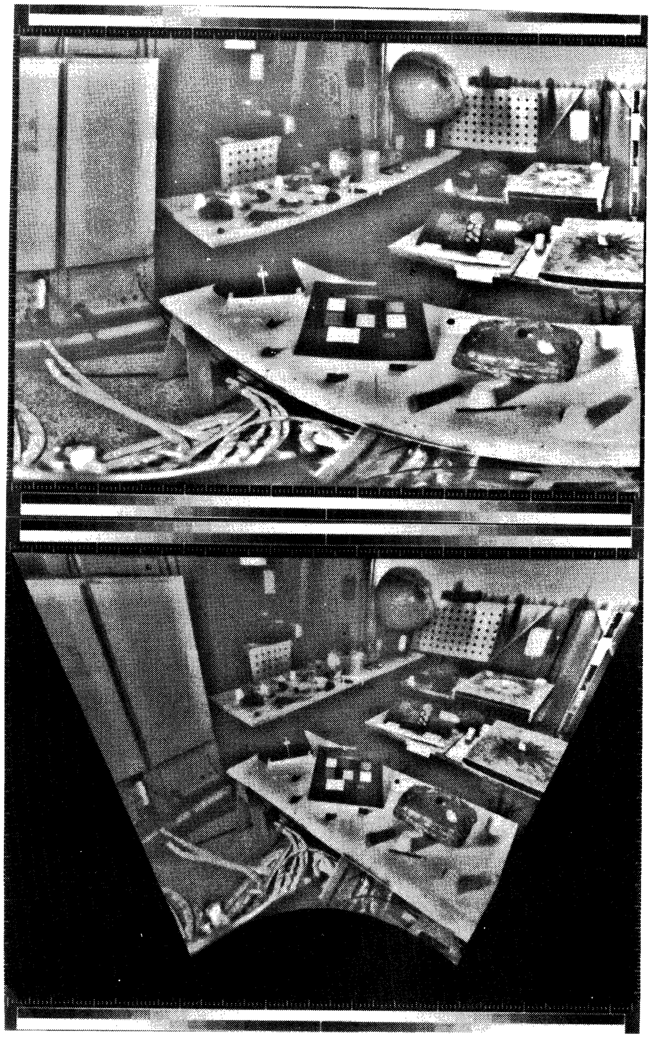


Figure 8–8 Viking Lander camera correction: (a) before, (b) after (Courtesy NASA-JPL)

Rectification of angle-scanned images for rectangular display involves the projection of a spherical surface onto a tangent plane. The projection lines emanate from the center of the sphere and carry points on its surface out to the plane. The relationship between input and output pixel location is derived in [14]. Figure 8–8(a) was rectified for rectangular display in Figure 8–8(b). Notice that the table edges appear straight, as they should, in the rectified image.

A free-roaming robot, like a human, requires wide-angle stereoscopic vision in order to navigate among obstacles, such as passing through doorways. A *fish-eye* lens can image a field of view approximately 180 degrees wide, but it does so with considerable distortion (Figure 8–9a,b). A properly designed geometric operation can rectify such an image into a rectangular coordinate system (Figure 8–9c,d) so that stereoscopic ranging techniques (discussed in Chapter 22) can locate the surrounding objects in three dimensions. In this example, a fifth-order polynomial warp, implemented in a polar coordinate system, rectified the images [20,21].

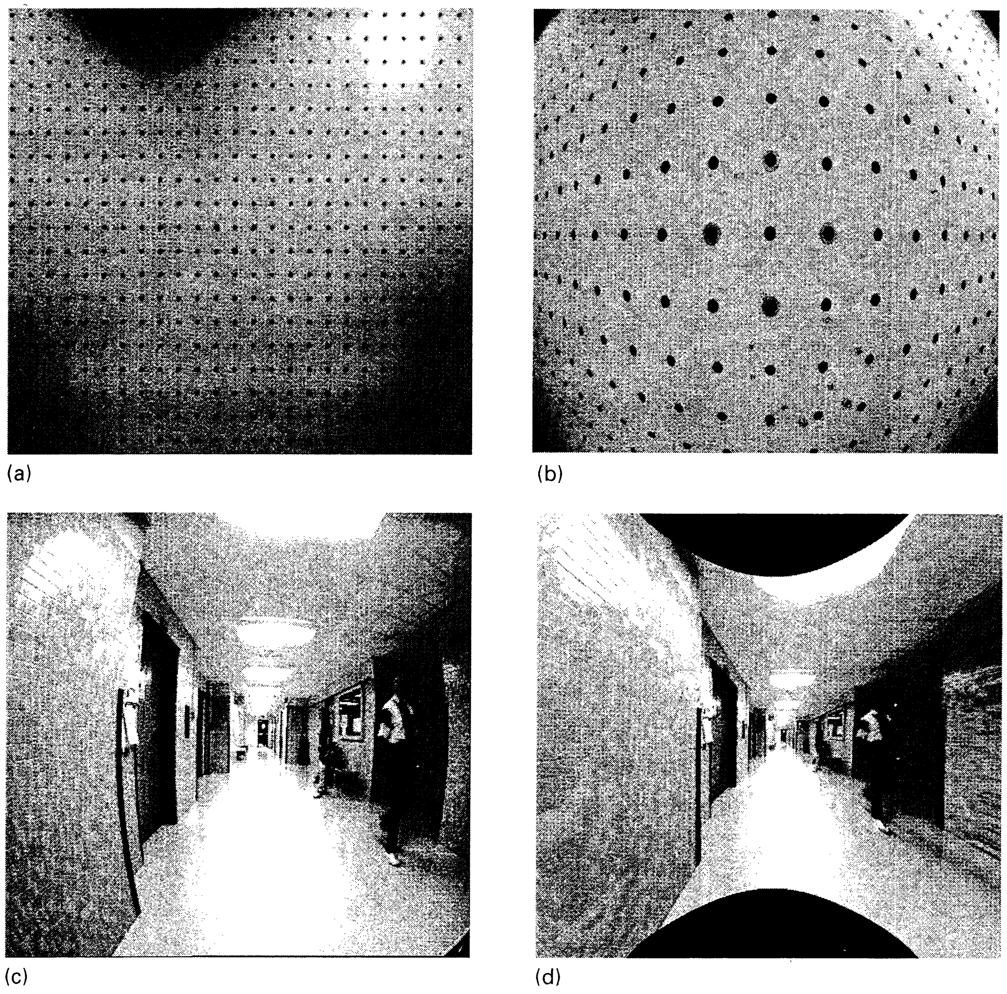


Figure 8–9 Geometric rectification of an image taken with a fish-eye lens: (a) test target, (b) fisheye image; (c) original, (d) rectified hallway image (Courtesy Shishir Shah, The University of Texas at Austin, from [20])

8.4.3 Image Registration

Another application of geometric operations is registering similar images for purposes of comparison. This is typified by image subtraction to detect motion or change. As pointed out in Chapter 7, if similar images are displaced slightly and subtracted, the difference image has a strong partial derivative component. This could easily mask the image differences of interest. If images of a stationary object can be digitized from a fixed camera position, they can be obtained in register. If this is not the case, however, it is likely that the images will have to be registered prior to subtraction.

While simple translation is easily accomplished, rotation or more complex distortion requires a geometric operation. Registration of film scan images is likely to involve translation and rotation. Serial sections of biological tissue, sliced on a microtome and photographed through a microscope, for example, are subject to rather severe geometric distortion. In such cases, simple translation and rotation are inadequate. Instead, one such image can be taken as a standard of reference and the others distorted to match it. Small features are located throughout the images and used to define control points. Chapter 7 shows examples of image subtraction in which careful registration is required.

8.4.4 Image Format Conversion

Geometric operations are sometimes useful simply for placing images into a format more convenient for interpretation. Figure 8–10(a) shows a photographic map of the chromo-

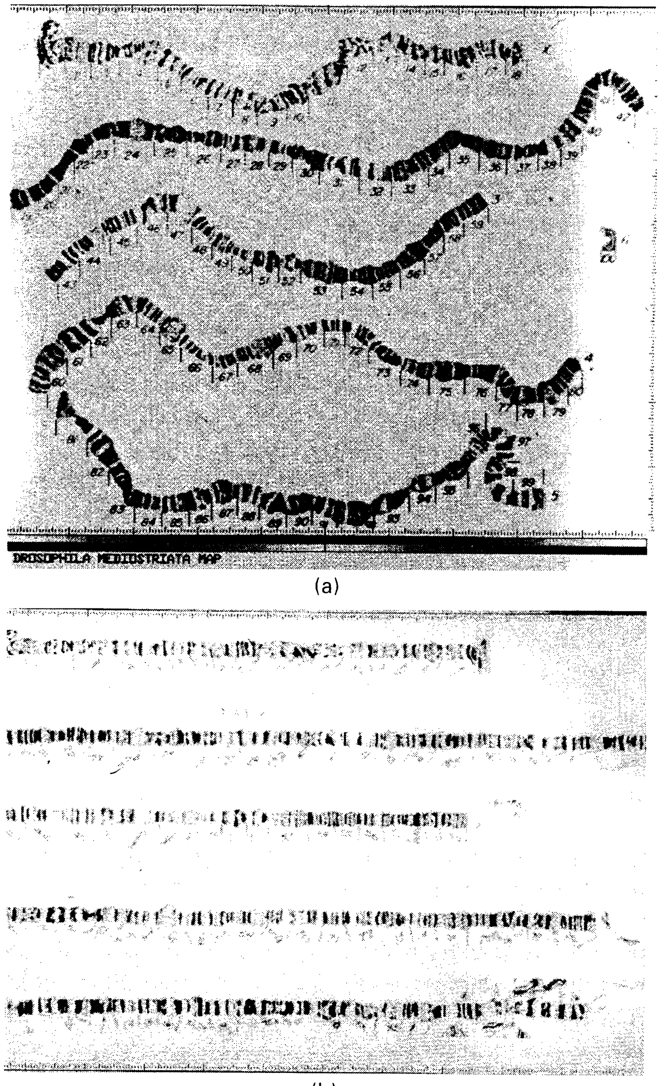


Figure 8–10 *Drosophila* chromosome map: (a) original, (b) straightened (Courtesy NASA-JPL)

somes of one species of the fruit fly *Drosophila*. The map is made by pasting up photographs of chromosomes taken through a microscope. Geneticists analyze the pattern of bands on the chromosomes to deduce patterns of evolution. The areas are numbered for reference.

Figure 8–10(b) shows the result of using a geometric operation to produce a map in which the chromosomes appear straight. In the input image, each chromosome was overlaid with a control grid of quadrilaterals, each with two sides parallel to the chromosome axis. These were mapped into horizontal strings of rectangles in the output image. In order to prevent axial distortion of the chromosome, the horizontal length of each rectangle was made equal to the mean of the two axial sides of the corresponding quadrilateral.

The numbers below chromosome 3 suffered less distortion than the others because a second row of quadrilaterals was defined below this chromosome. These were actually parallelograms with vertical ends and sides parallel to the chromosome axis. They mapped into a second row of rectangles falling beneath those that defined the straightened chromosome.

8.4.5 Map Projection

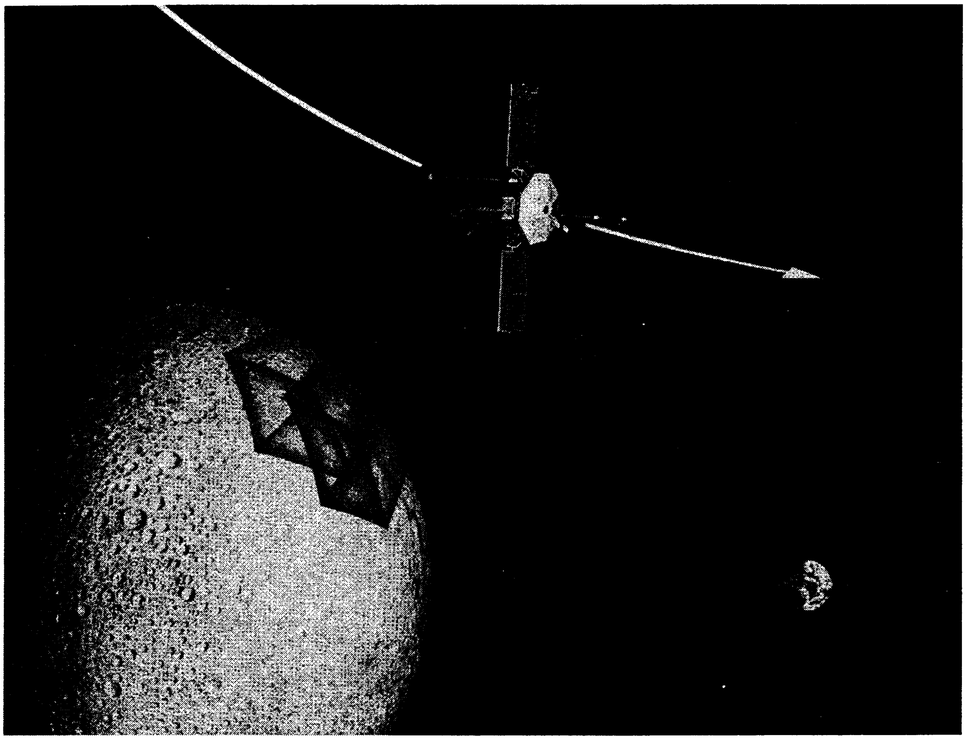
Another major application of geometric operations is projecting images for purposes of mapping. For example, it is necessary to produce photomosaic maps of the Earth, moon, and planets using images transmitted back from spacecraft. The borders of the spacecraft camera image project onto the planet's surface, forming a "footprint" with four curvilinear sides [Figure 8–11(a)]. The spherical surface of the planet is projected onto a flat surface to make a map [Figure 8–11(b)]. The "footprint" also projects onto the map, producing a further distorted four-sided figure.

A geometric operation can transform the spacecraft camera image into the form it should assume on the map. Multiple images processed in this way can be combined into a mosaic to form a photographic map of the planet. The task of determining the control points for projecting a given image is somewhat involved. The program must take the spacecraft viewing geometry and the desired cartographic projection parameters and generate input and output control grids.

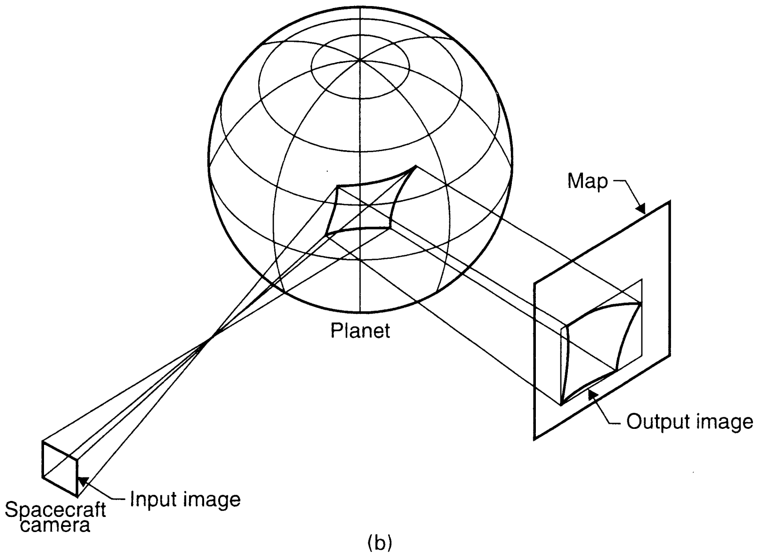
Determining the spatial transformation between the input and the projected image is a two-step process. Software used in the space program solves this problem by working backward from the output image to the input image. The specified cartographic projection technique defines the relationship between points in the output image and points on the planet's surface. The spacecraft viewing geometry determines the spatial relationship between points on the surface of the planet and pixel positions in the camera image. The program overlays a rectangular control grid on the output image and maps it back through the cartographic projection and the spacecraft viewing geometry to overlay it on the input image. The following section outlines this technique.

Cartography. The science of cartography is concerned with producing two-dimensional maps of spherical or ellipsoidal bodies. This is not a simple matter, because spherical surfaces cannot be flattened without distortion. Cartographers solve the problem by projecting the spherical surface onto a plane or onto a cylinder or cone that can be "unrolled" to form a flat surface [22,23].

Map Properties. There are three important properties that a particular map may or may not have, depending on its method of generation. A map is said to be *equidistant* if



(a)



(b)

Figure 8–11 Photographic mapping: (a) spacecraft camera "footprint," (b) map projection (Courtesy NASA-JPL)

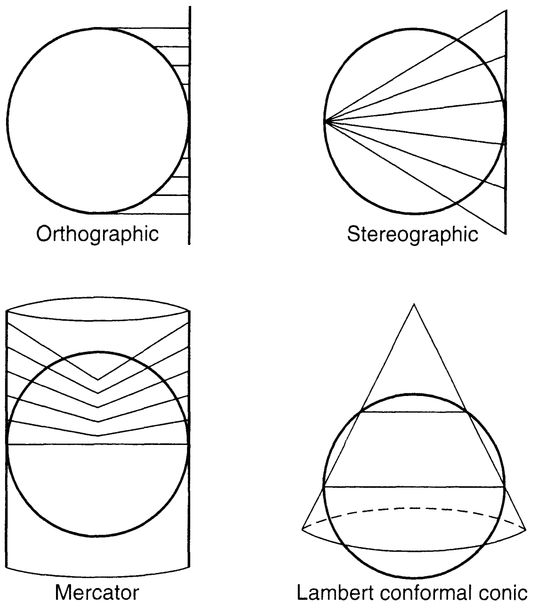


Figure 8–12 Cartographic projections

scale is preserved along certain lines. This means distances along those lines are proportional to the distance between corresponding points on the planet. A map has the property of *equivalence* if the area of a region is preserved in the projection. Such maps may be used for comparing the areas of different features. A map is *conformal* or *orthomorphic* if angles are preserved in the projection—that is, if lines on the surface intersect at the same angle as their projections on the map. A conformal map also preserves shape at a point. This means that the shape of small features is distorted only very slightly. The distortion of shapes becomes progressively more significant as the size of the features increases.

Cartographic Projections. There are three types of surfaces onto which surface features may be projected to form a two-dimensional map: the plane, the cylinder, and the cone. The last two must be cut along a line parallel to the axis and "unrolled" to form a flat map. The cone may be considered the general case, since the plane can be thought of as a cone with apex angle 180° and the cylinder a cone with apex angle 0°.

While many types of projections have been defined and used throughout cartographic history, four of the most important are the *orthographic*, the *stereographic*, the *Mercator*, and the *Lambert conformal conic* projections [24]. These projections differ in the techniques by which they are generated and in their properties. They are described next, with reference to Figure 8–12.

In the orthographic projection, surface features are projected onto a plane tangent to the sphere at a point called the *center of projection.* Features are projected along parallel lines normal to the plane. When the center of projection is a pole, the scale along parallels of latitude is constant. By contrast, the radial scale decreases away from the center of projection. There is little distortion of features near the center of projection. Parallels of latitude project as concentric circles centered on the pole, and meridians project as straight lines intersecting at the pole.

The orthographic projection is useful because it approximates viewing the planet from a large distance, and the eye is able to visualize the spherical shape of the planet. Because scale and shape are distorted, however, orthographic maps are of restricted quantitative use, except for small features near the center of projection.

The stereographic projection is similar to the orthographic projection, except that the projection rays emanate from a *perspective point* located directly opposite the center of projection. In the polar case, parallels of latitude project as concentric circles centered on the pole, and meridians project as radial lines intersecting at the pole. The scale along parallels and that along meridians increase away from the pole. They increase proportionately, however, so that at any point the longitude and latitude scales are the same. This makes the stereographic projection conformal, and shape is preserved locally. There is little distortion of features near the center of projection. Coupled with conformality, this property makes the stereographic projection quite useful.

The Mercator projection maps surface features onto a right circular cylinder that is tangent to the sphere at the equator. The cylinder axis is colinear with the polar axis of the sphere. Meridians map to equidistant vertical lines, and parallels map to circles on the cylinder, which open up to form horizontal lines on the map. Scale along latitude lines increases with distance from the equator. The projection is designed so that the perspectival point moves up the axis with increasing latitude, keeping the latitude and longitude scales equal and thus making the map conformal. Scale is exaggerated away from the equator, and features near the poles become quite large. The poles themselves cannot be mapped.

The vertical position of latitude lines is given by

$$y = R\ln\left[\tan\left(45 + \frac{\phi}{2}\right)\right] \tag{22}$$

where R is the planet's radius on the map and ϕ is latitude.

Historically, the Mercator projection has been used for navigation because a course of constant compass heading projects to a straight line on the map.

In the Lambert conformal conic projection, surface features are projected onto a cone having the same axis as the planet. The cone intersects the sphere at two parallels called the standard parallels. Meridians map to straight lines, and parallels map to circles inside the cone. When the cone is unrolled, the parallels become arcs and the meridians merge at the pole. The spacing of the parallels is adjusted to achieve conformality. The two standard parallels project at true scale: Scale decreases between them and is exaggerated outside of them.

8.4.5.1 Implementation

The steps necessary to project a spacecraft image for mapping purposes are the following:

1. Establish the spacecraft camera viewing geometry.
2. Determine an expression giving camera position as a function of the latitude and longitude of the corresponding point on the planet's surface.
3. Select the map projection parameters (type of projection, center of projection, etc.), and establish the borders of the output image on the map.
4. Determine an expression giving the latitude and longitude of a point on the planet's surface in terms of the pixel coordinates of the corresponding point on the map.

5. Combine the results of steps 2 and 4 to yield an expression giving camera pixel position as a function of position on the output map.
6. Overlay a rectilinear control grid on the output picture.
7. Use the expression of step 5 to map the output control points into the input image, thus establishing the input control grid.
8. Use the results of step 7 in a geometric operation to effect the projection.

The spacecraft viewing geometry may be established with reference to Figure 8–13. In this figure, the spacecraft is located at a distance R_s from the center of the planet, directly above the point at latitude ϕ_s and longitude λ_s. Point C is the perspective point that represents the nodal point (center) of the camera lens. Point p is in the camera image and corresponds to point p', which has longitude λ and latitude ϕ on the surface. The distance f represents the focal length of the lens and is exaggerated for clarity in the figure. The vector **Q** extends from C to p'. Notice that the vector

$$\mathbf{P} = \begin{bmatrix} x_p \\ y_p \\ f \end{bmatrix} \tag{23}$$

has components x_p and y_p, which are the camera pixel position coordinates. Since **P** and **Q** are colinear, they are related by a scale factor:

$$\mathbf{P} = \left(\frac{f}{Q_z}\right)\mathbf{Q} \tag{24}$$

From Figure 8–13, we see that

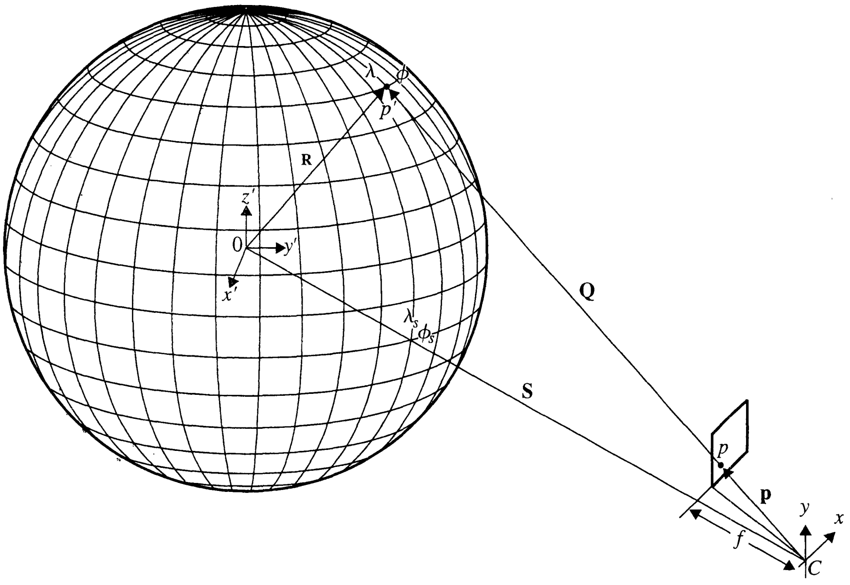


Figure 8–13 Spacecraft viewing geometry

$$\mathbf{Q} = \mathbf{R} - \mathbf{S} \tag{25}$$

which we can write in matrix notation as

$$\begin{bmatrix} Q_x \\ Q_y \\ Q_z \end{bmatrix} = [M] \begin{bmatrix} R\cos\phi\cos\lambda - R_s\cos\phi_s\cos\lambda_s \\ R\cos\phi\sin\lambda - R_s\cos\phi_s\sin\lambda_s \\ R\sin\phi - R_s\sin\phi_s \end{bmatrix} \tag{26}$$

where $[M]$ is the three-by-three matrix that transforms from planet-centered to spacecraft coordinates.

Finally, Eq. (24) implies that

$$x_p = \left(\frac{f}{Q_z}\right)Q_x \text{ and } y_p = \left(\frac{f}{Q_z}\right)Q_y \tag{27}$$

Several cartography texts develop equations that give map position in terms of latitude and longitude on the surface. Since we must work backward from map to planet, however, inverse forms of the equations are required. These are developed in [24] for the four projections mentioned above.

Spacecraft images often require both geometric correction and map projection, suggesting two sequential geometric operations. Pixel interpolation done twice, however, would reduce detail in the image, so the two geometric operations are usually combined into one execution that both corrects and projects the image.

8.4.5.2 Examples of Map Projection

Figure 8–14 illustrates the steps used in producing a photographic map. Part (a) is a Mariner 10 image of Mercury prior to correction for photometric and geometric distortion. In (b), the image has been subjected to a geometric operation to produce an orthographic projection. In (c), several neighboring orthographic projections have been combined to form a mosaic. Finally, in (d), a latitude and longitude grid has overlaid the orthographic mosaic.

Figure 8–15 shows a polar orthographic projection of images of Mars taken from Mariner 6 and Mariner 7 [25]. A mosaic of high-resolution, narrow-angle images has been inserted into a mosaic of wide-angle, low-resolution pictures of the entire polar area. Figure 8–16 shows a four-foot-diameter globe covered with 2,000 orthographic projections of Mariner 9 images of Mars [15].

8.4.6 Morphing

Several special effects that have become popular in the motion picture and television industries are based on geometric operations. *Morphing* is a technique that allows one object to transform gradually into another [26].

Suppose we have two images from which we wish to create a sequence of movie frames. That sequence is to depict the transformation of the object in the first scene into the object in the second scene. An example would be transforming the face of a cat into the face of a tiger. In a *dissolve,* the first image gradually fades out as the second fades in. This technique rarely produces a realistic looking transformation. With a morph, however, during a

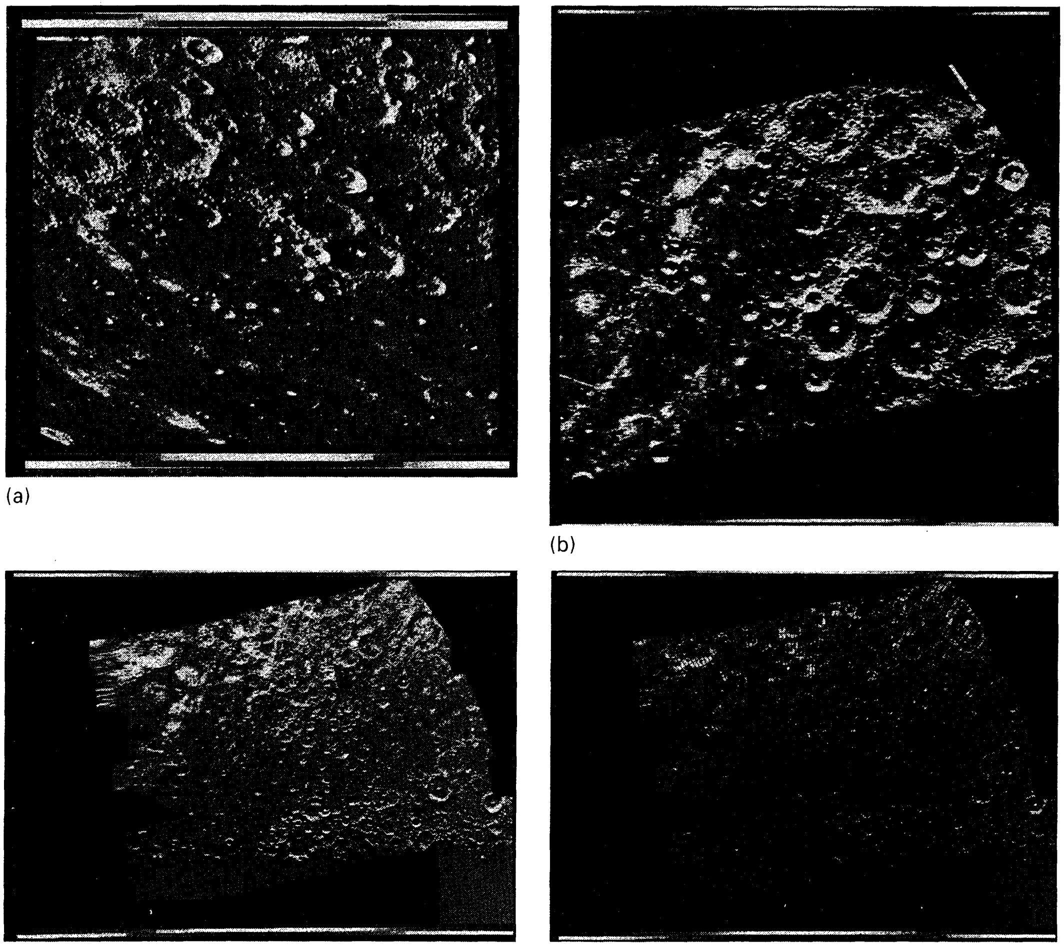

(a) (b) (c) (d)

Figure 8–14 Example of Mariner 10 map projection: (a) original image; (b) orthographic projection; (c) mosaic of several projections; (d) map grid overlay (Courtesy NASA-JPL)

dissolve points on the object are incrementally warped from their initial position to their final position, creating a more impressive result.

Figure 8–17 shows four frames from a morph sequence. Figures 8–17a and 8–17d are the initial and final images, respectively. Figure 8–17b and 8–17c represent the 40% and 70% points, respectively, in the sequence.

At each step in the sequence, both the initial and final images are warped so that their control points map to positions intermediate between their initial and final positions. This produces two sequences in which the marked features move gradually from their initial to their final positions. A dissolve between these two sequences completes the morph operation.

Morphing can also be done between two movie sequences. Here, since the objects are moving, the corresponding control points must be designated in each frame of each sequence. Most commonly, the control points are specified for only a few of the frames, and

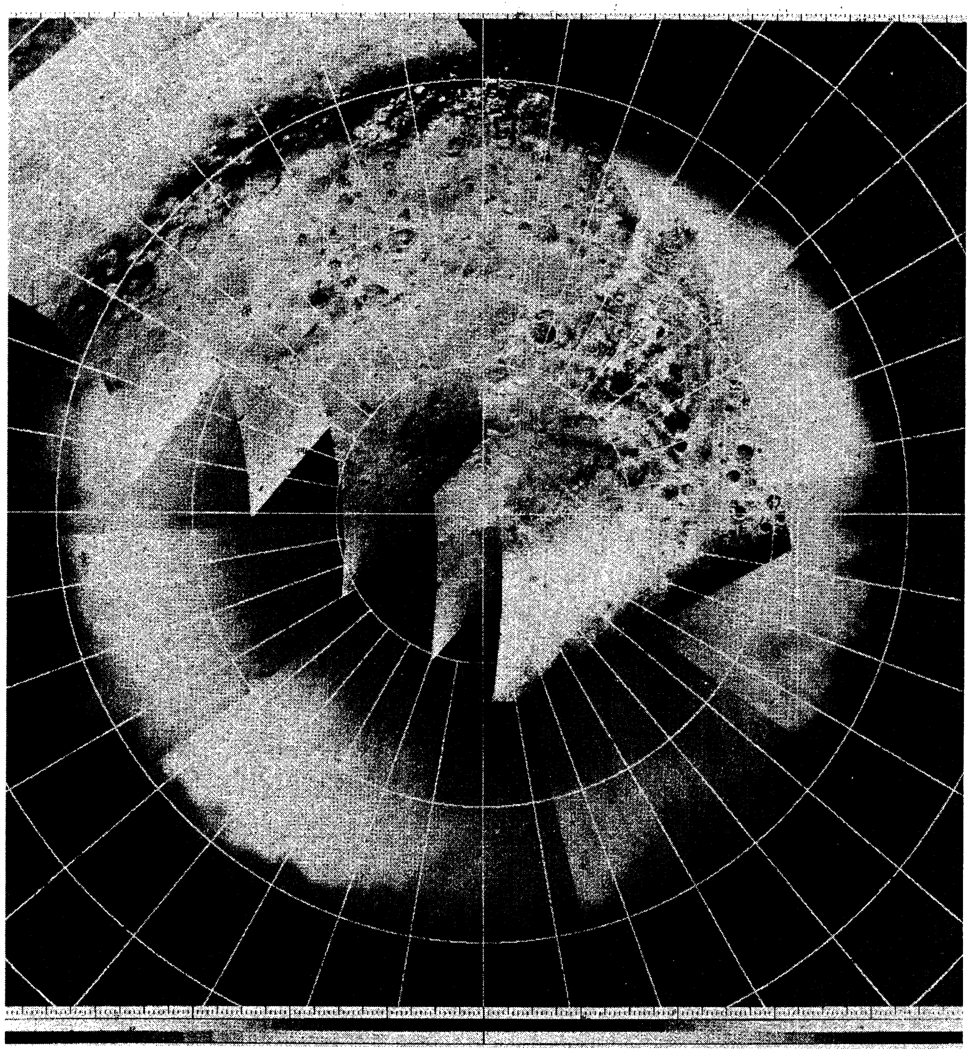


Figure 8–15 Polar orthographic map of Mars (Courtesy NASA-JPL, from [23])

spatial interpolation supplies the rest. At each frame in the sequence, the two images are warped so that their control points align. The position to which a pair of control points is mapped starts near the initial image position and gradually moves toward the final image position as the sequence progresses.

In practice, it is often only one object in the scene that is actually transformed, with the background remaining stationary. The object of interest is filmed against a black background. The finished morph sequence is then inserted into a scene containing the appropriate background.

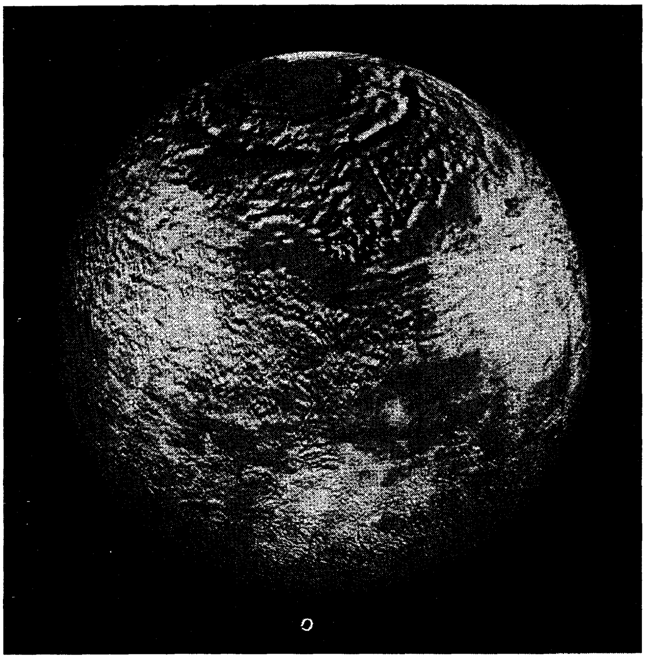


Figure 8–16 Mariner 9 photomosaic globe of Mars (Courtesy NASA-JPL)

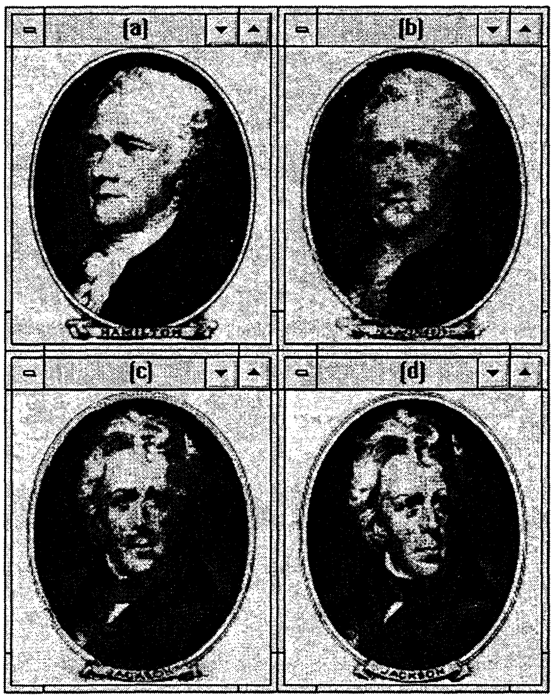


Figure 8–17 Image morphing sequence: (a) initial image, (b) 40% point, (c) 70% point, (d) final image.

8.5 SUMMARY OF IMPORTANT POINTS

1. A geometric operation requires a means for specifying its spatial transformation and an algorithm for gray-level interpolation.
2. A geometric operation can be thought of as mapping each output image pixel into the input image, where the ouput gray-level value is determined by interpolation.
3. Bilinear gray-level interpolation is generally superior to nearest neighbor interpolation, and it produces only a modest increase in program complexity and execution time.
4. A spatial transformation can be specified by a pair of control grids, one defined in the input image and one in the output image.
5. The input control points map to the corresponding output control points.
6. Between control points, a spatial transformation is obtained by interpolation.
7. Bilinear interpolation is useful for non-control-point interpolation.
8. Geometric operations are useful for digitizer calibration, display rectification, image registration, map projection, image reformatting for display, and visual special effects.

PROBLEMS

1. Let $F(221,396) = 18$, $F(221,397) = 45$, $F(222,396) = 52$, and $F(222,397) = 36$. What is $F(221.3,396.7)$, obtained by nearest neighbor interpolation? By bilinear interpolation? Write the bilinear equation (Eq. 2), showing the values of the coefficients. Draw a graph similar to Figure 8–3.
2. Let $F(109,775) = 113$, $F(109,776) = 109$, $F(110,775) = 105$, and $F(110,776) = 103$. What is $F(110.27,776.44)$, obtained by nearest neighbor interpolation? By bilinear interpolation? Write the bilinear equation (Eq. 2), showing the values of the coefficients. Draw a graph similar to Figure 8–3.
3. Write the geometric transformation required to rotate an image 33° counterclockwise about the point $x, y = 207,421$. Assume that 0,0 is at the upper left.
4. Suppose you have two digitized images of a canyon wall taken 100 years apart and you wish to detect changes due to erosion by image subtraction. You find a rock that is located at 303,467 in the first image and at 316,440 in the second image, and a stump that is located at 298,227 in the first image and at 311,200 in the second image. Has there been any (a) translation? (b) rotation? (c) change in scale? How much? Write the geometric transformation required to register the second image with the first prior to subtraction. Assume that there has been no geometric distortion beyond translation, rotation, and change in scale.
5. Suppose you have two digitized images of a section of a city taken from the top of a tall building 25 years apart and you wish to display changes by projecting an overlay of the two images. You find a corner of a building that is located at 103,84 in the first image and at 107,94 in the second image, and a window that is located at 433,504 in the first image and at 377,439 in the second image. Has there been any (a) translation? (b) rotation? (c) change in scale? How much? Write the geometric transformation required to register the second image with the first. Assume that there has been no geometric distortion beyond translation, rotation, and change in scale.
6. Suppose you have two digitized images of a movie star's face taken 30 years apart and you wish to include a fade between the two portraits in an upcoming documentary. You find that the cen-

ters of the film idol's pupils are located at 83,231 and 437,244 in the first image and at 64,281 and 479,370 in the second image. Has there been any (a) translation? (b) rotation? (c) change in scale? How much? Write the geometric transformation required to register the second image with the first. Assume that there has been no geometric distortion beyond translation, rotation, and change in scale.

7. Suppose you have a digitized photograph of the ground taken at an angle from behind the window of an airplane. You want to rectify the image so that it appears as if you are looking straight down. A square cotton field has corners at pixel coordinates (62, 85), (77, 128), (125, 134), and (140, 106). Derive the geometric transformation that will rectify the image. Plot the cotton field in the image before and after rectification.
8. Suppose you have film that was taken by a security camera during a daring daylight holdup of a bank. At one point in the series of images, one of the bandits ducks behind a counter and briefly removes his mask. Beside him is a chrome-plated vertical column 24 inches in diameter. The reflection of his face is visible in the shiny column, but is too distorted for identification. Derive the equation for a geometric transformation that will rectify the image of the bad guy. Assume that the column is parallel to the y-axis in the digitized image and the pixel spacing corresponds to 10 pixels per inch at the column. You may also assume that the radius of the column is negligible compared to its distance from the camera and from the villain.
9. The police have a photograph taken during the commission of a crime. Unfortunately, the robbery took place behind the tourist with the camera. You notice in the photo a large chrome-plated sphere that appears to be acting like a mirror. You use a film digitizer with 25-μ pixel spacing and find that the image of the ball measures 360 pixels in diameter. The actual ball is 3 feet in diameter, and it was 27 feet from the camera position. Develop an equation for the geometric transformation that will rectify the image of the robbers. You may assume that the radius of the ball is negligible compared to its distance from the camera and from the crime.
10. Suppose you have a photograph of a bite mark on the arm of a murder victim. The photo was taken at the autopsy of the victim. The body has since been cremated, but the district attorney needs a rectified picture of the bite mark to match against the bite of the suspect in order to get a conviction. Assume that the arm is a cylinder 80 mm in diameter lying parallel to the x-axis. Develop the equation for a geometric transformation that will "unroll" the bite mark for comparison with bites made on wax sheets by the suspect.
11. Develop the geometric transformation equations required to rotate an image 60 degrees counterclockwise about the point $x, y = 120,210$.
12. Develop the geometric transformation equations required to scale an image by 130 percent about the point $x, y = 64,120$ along a line 30 degrees counterclockwise from the x-axis and by 85 percent along a line 30 degrees counterclockwise from the y-axis.
13. Derive Eq. (18).

PROJECTS

These projects require access to an image-processing workstation with digitization and general geometric transformation capabilities.

1. Digitize an image of a friend taken with a wide-angle "fish-eye" lens. Develop equations that describe the distortion and use polynomial warping to correct it.
2. Digitize an image of a friend taken with a fish-eye lens. Use linear objects in the image as fiducial marks, and use a geometric transformation to correct the image.

3. Digitize an image of a friend taken with a fish-eye lens. Take a second image of a grid from the same camera position, and use a geometric transformation to correct the image
4. Digitize an image of a spherical hallway safety mirror, and use a geometric transformation to rectify the image. Use linear objects in the image as fiducial marks.
5. Digitize an image of a person reflected in a fun house mirror and then a second image containing a large grid. Use the same camera position both times. Use a geometric transformation to rectify the image of the person. Write a report, commenting on the accuracy of the results and any difficulties encountered.
6. Digitize an image of a large grid in a fun house mirror. Digitize an ordinary image of a friend of yours, and use a geometric operation to show what he or she would look like if seen in the mirror.
7. Digitize an image of a structure such as a house or building taken from an odd angle. Use a geometric operation to develop elevation (90 degree) views of the structure.
8. Develop an image-processing program that can be used to predict the effects of cosmetic and reconstructive surgery (a "nose job," chin augmentation, etc.). Use the program to determine what, if any, cosmetic surgery might improve your appearance or that of a friend or a celebrity.
9. Develop a geometric transformation program that will warp one facial photograph to match the features in another. Digitize a picture of a famous personality, digitize a picture of yourself in the same pose, and warp your picture to look like the other person.
10. Use a geometric transformation to "unroll" a picture of a poster wrapped around a pole.
11. Develop a geometric transformation program that will rotate, translate, and scale an image by specified amounts. Evaluate your implementation in terms of speed and accuracy.
12. Develop a geometric transformation program that will rotate, translate, and scale an image by specified amounts when the rotation angle is small (say, $\theta \leq 6°$). Use approximations to make execution of the program as fast as possible. Evaluate your implementation in terms of speed and accuracy.
13. Develop an image sequence that morphs an image of your face into that of a famous person.
14. View a movie containing morph operations (e.g., *Terminator 2*) on a video player with stop-motion capability. Examine the morph sequences in slow motion, and estimate how many control points were required and where they were located. Write a brief paper outlining your estimates, along with the digitizing, processing and display requirements for this project, as well as on the financial impact of these scenes on the producers.

REFERENCES

For additional reading, see Appendix 2.

1. L. G. Roberts, "Machine Perception of Three-Dimensional Solids," in J. D. Tippet et al, eds., *Optical and Electro-Optical Information Processing,* MIT Press, Cambridge, MA, 1965.
2. D. F. Rogers and J. A. Adams, *Mathematical Elements for Computer Graphics,* McGraw-Hill, New York, 1976.
3. J. D. Foley and A. van Dam, *Fundamentals of Interactive Computer Graphics,* Addison-Wesley, Reading, MA, 1982.
4. W. M. Newman and R. F. Sproul, *Principles of Interactive Computer Graphics* (2d ed.), McGraw-Hill, New York, 1979, Sec. 4.3.
5. D. H. Ballard and C. M. Brown, *Computer Vision,* Prentice-Hall, Englewood Cliffs, NJ, 1982, Sec. A1.7.

6. R. Nevatia, *Machine Perception,* Prentice-Hall, Englewood Cliffs, NJ, 1982, Sec. 3.2.
7. W. K. Pratt, *Digital Image Processing,* John Wiley and Sons, New York, 1991, Sec. 14.1.
8. R. C. Gonzales and R. C. Woods, *Digital Image Processing,* Addison-Wesley, Reading, MA, 1992, Sec. 2.5.
9. K. S. Fu, R. C. Gonzalez, and C. S. G. Lee, *Robotics: Control, Sensing, Vision, and Intelligence,* McGraw-Hill, New York, 1987.
10. E. Catmull and A. R. Smith, "3-D Transformation of Images in Scanline Order," *Computer Graphics,* **14**(3):279–285, 1980.
11. Z. Jericevic, D. M. Benson, J. Bryan, and L. C. Smith, "Geometric Correction of Digital Images Using Orthonormal Decomposition," *Journal of Microscopy,* **149**(3):233–245, 1987.
12. H. Frieden, "GEOM and LGEOM," in *Image Processing Laboratory Users Documentation of Applications Programs,* Jet Propulsion Laboratory Internal Document No. 900-670, Pasadena, CA, October 11, 1975.
13. J. E. Kreznar, *User and Programmer Guide to the MM'71 Geometric Calibration and Decalibration Programs,* Jet Propulsion Laboratory Internal Document No. 900-575, Pasadena, CA, March 1, 1973.
14. M. Girard, *Viking 75 Project Software Requirements Document for the GEOTRAN Program,* Jet Propulsion Laboratory Internal Document No. 620-52, Pasadena, CA, January 12, 1975.
15. W. B. Green, P. L. Jepsen, J. E. Kreznar, R. M. Ruiz, A. A. Schwartz, and J. B. Seidman, "Removal of Instrument Signature from Mariner 9 Television Images of Mars," *Applied Optics,* **14:**105–114, 1975.
16. D. A. O'Handley and W. B. Green, "Recent Developments in Digital Image Processing at the Image Processing Laboratory of the Jet Propulsion Laboratory," *Proc. IEEE,* **60**(7):821–828, July 1972.
17. C. L. Lawson and R. J. Hanson, *Solving Least Squares Problems,* Prentice-Hall, Englewood Cliffs, NJ, 1974.
18. W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C* (2d ed.), Cambridge University Press, Cambridge, U.K., 1992.
19. R. Nathan, *Digital Video Data Handling,* JPL Tech. Report 32-877, Pasadena, CA, January 5, 1966.
20. S. Shah and J. K. Aggarwal, "A Simple Calibration Procedure for Fish-Eye (High-Distortion) Lens Camera," *Proc. IEEE Int. Conf. on Robotics and Automation,* 3422–3427, 1994.
21. S. Shah and J. K. Aggarwal, "Depth Estimation Using Stereo Fish-Eye Lenses," *ICIP-94, Proc. IEEE Int. Conf. on Image Processing,* **2:** 740–744, 1994.
22. G. P. Keloway, *Map Projections,* Methuen and Co., London, 1946.
23. P. Richardis and R. K. Adler, *Map Projections,* American Elsevier, New York, 1972.
24. C. A. Rofer, A. A. Schwartz, J. B. Seidman, and W. B. Green, *Computer Cartographic Projections for Planetary Mosaics—Final Report* (three volumes), Jet Propulsion Laboratory Internal Document No. 900-636, Pasadena, CA, November 15, 1973.
25. A. R. Gillespie and J. M. Soha, "An Orthographic Photomap of the South Pole of Mars," *Icarus,* **16:**522, 1972.
26. G. Wolberg, *Digital Image Warping*, IEEE Computer Society Press, Los Alamitos, CA, 1990.