Chapter 1

Introduction

1.1 Machine Vision

The goal of a machine vision system is to create a model of the real world from images. *A machine vision system recovers useful information about a scene from its two-dimensional projections.* Since images are two-dimensional projections of the three-dimensional world, the information is not directly available and must be recovered. This recovery requires the inversion of a many-to-one mapping. To recover the information, knowledge about the objects in the scene and projection geometry is required.

To consider applications of a machine vision system and type of processing required, let us consider the three images shown in Figures 1.1 to 1.3. These figures show three different applications of a machine vision system. The information recovered by a vision system in the three cases is different. In the first figure, we are interested in diagnosis of a disease using computed tomography images. Machine vision systems help a physician to recover information by enhancing the images. Quantitative measurements on regions of interest can also be made easily available. Such systems are being developed for all imaging modes useful in different aspects of health care. Similar applications are being developed for inspection of industrial, agricultural, and other products. Machine vision systems have been used for quality control of products ranging from pizza to turbine blades, from submicron structures on wafers to auto-body panels, and from apples to oranges.

Figure 1.2 shows two pairs of images acquired by a mobile robot. Each pair represents a stereo pair at a particular time instant. These images are

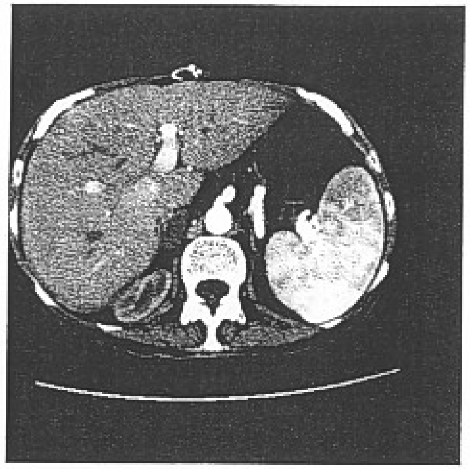


Figure 1.1: Medical images may be processed by a computer vision system to assist in diagnosis. This image is a contrast-enhanced CT (transverse X-ray computed tomography) image of a human liver displayed at the soft tissue window settings. (*Courtesy of Prof. Charles R. Meyer and Dr. Peyton Bland, Department of Radiology, University of Michigan, Ann Arbor.*)

used to recover three-dimensional structure of the environment by the robot for autonomously navigating in its environment. The information obtained from stereo and motion is combined to get a robust map of the environment at a resolution that is sufficient for the task. Such techniques are useful in autonomous navigation of automobiles, airplanes, tanks, and robots. We will see that there are many different methods to recover depth to points in images. Usually the results of several of these must be combined to recover reliable depth values.

Figure 1.3 shows an image of an arctic region taken from a satellite. Different regions in this image correspond to ice floes of different age. Such images are routinely taken by satellites. These images are extremely important in weather forecasting, global change analysis, agriculture and forestry, and other applications. Machine vision systems are playing an increasingly important role in analysis and information management of the exceedingly large volume of data collected by satellites.

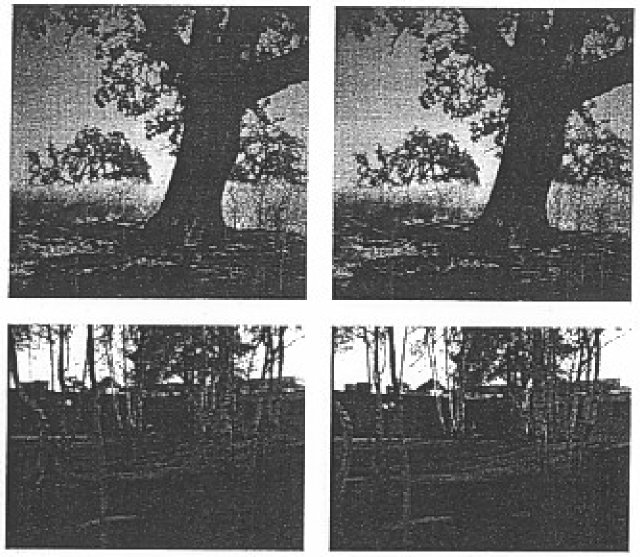


Figure 1.2: Two pairs of stereo images acquired by a mobile robot. These will be used in recovering the layout of the environment by the robot. (*Images provided by Robert Bolles at SRI.*)

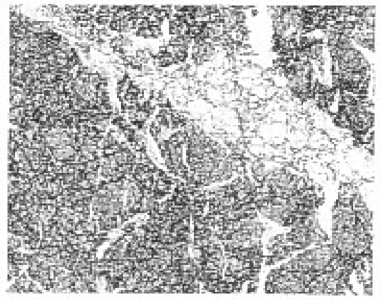


Figure 1.3: An image of the arctic region. This image should be analyzed to find the age and size of ice floes and other objects. (*Arctic image provided by Jason Daida at the University of Michigan.*)

1.2 Relationships to Other Fields

Many fields are related to machine vision. As we will see, techniques developed from many areas are used for recovering information from images. In this section, we briefly describe some very closely related fields. No effort is made to relate machine vision (also called computer vision) to other fields exhaustively.

Image processing is a well-developed field. Image processing techniques usually transform images into other images; the task of information recovery is left to a human user. This field includes topics such as image enhancement, image compression, and correcting blurred or out-of-focus images. On the other hand, machine vision algorithms take images as inputs but produce other types of outputs, such as representations for the object contours in an image. Thus, emphasis in machine vision is on recovering information automatically, with minimal interaction with a human. Image processing algorithms are useful in early stages of a machine vision system. They are usually used to enhance particular information and suppress noise.

Computer graphics generates images from geometric primitives such as lines, circles, and free-form surfaces. Computer graphics techniques play a significant role in visualization and virtual reality. Machine vision is the inverse problem: estimating the geometric primitives and other features from the image. Thus, computer graphics is the synthesis of images, and machine vision is the analysis of images. In the early days of these two fields, there was not much relationship between them, but in the last few years these two fields have been growing closer. Machine vision is using curve and surface representations and several other techniques from computer graphics, and computer graphics is using many techniques from machine vision to enter models into the computer for creating realistic images. Visualization and virtual reality are bringing these two fields closer.

Pattern recognition classifies numerical and symbolic data. Many statistical and syntactical techniques have been developed for classification of patterns. Techniques from pattern recognition play an important role in machine vision for recognizing objects. In fact, many industrial applications rely heavily on pattern recognition. Object recognition in machine vision usually requires many other techniques. We will discuss some aspects of statistical pattern recognition briefly in the discussion of object recognition.

Artificial intelligence is concerned with designing systems that are intelligent and with studying computational aspects of intelligence. Artificial intelligence is used to analyze scenes by computing a symbolic representation of the scene contents after the images have been processed to obtain features. Artificial intelligence may be viewed as having three stages: perception, cognition, and action. Perception translates signals from the world into symbols, cognition manipulates symbols, and action translates symbols into signals that effect changes in the world. Many techniques from artificial intelligence play important roles in all aspects of computer vision. In fact, computer vision is often considered a subfield of artificial intelligence.

Design and analysis of *neural networks* has become a very active field in the last decade [147, 148]. Neural networks are being increasingly applied to solve some machine vision problems. Since this field is in its infancy, there are no established techniques for machine vision yet. We will not discuss any neural network–based techniques in this book.

Psychophysics, along with cognitive science, has studied human vision for a long time. Many techniques in machine vision are related to what is known about human vision. Many researchers in computer vision are more interested in preparing computational models of human vision than in designing machine vision systems. Many of the techniques discussed in this book have a strong similarity to those in psychophysics. Our emphasis in this book will be on designing machine vision systems; we will not make any effort to relate the techniques discussed here to those in psychophysics.

Machine vision produces measurements or abstractions from geometrical properties. It may be useful to remember the equation

$$\text{Vision} = \text{Geometry} + \text{Measurement} + \text{Interpretation}. \tag{1.1}$$

As you will see in this book, machine vision comprises techniques for estimating features in images, relating feature measurements to the geometry of objects in space, and interpreting this geometric information.

1.3 Role of Knowledge

Decision making always requires knowledge of the application or goal. As we will see, at every stage in machine vision decisions must be made by the system. Emphasis in machine vision systems is [illegible] operation at each stage, and these systems should use knowledge to accomplish this. The knowledge used by the system includes models of features, image formation, models of objects, and relationships among objects. Without explicit use of knowledge, machine vision systems can be designed to work only in a very constrained environment for very limited applications. To provide more flexibility and robustness, knowledge is represented explicitly and used by the system. Our goal in this book is to point out the types of knowledge used by machine vision systems at different stages in order to make the reader aware of the issues that should be considered to make the system more adaptive and robust. We will see that knowledge is used by designers of systems in many implicit as well as explicit forms. The efficacy and efficiency of a system is usually governed by the quality of the knowledge used by the system. Difficult problems are often solvable only by identifying the proper source of knowledge and appropriate mechanisms to use it in the system.

1.4 Image Geometry

There are two parts to the image formation process:

1. The geometry of image formation, which determines where in the image plane the projection of a point in the scene will be located.
2. The physics of light, which determines the brightness of a point in the image plane as a function of scene illumination and surface properties.

This section introduces the first of these two parts, the geometry of image formation. Although an understanding of the physics of light is not necessary for understanding the fundamentals of most vision algorithms, such knowledge is often useful in building vision systems. For this reason, optics will be covered briefly in Chapter 8, and the physics of image formation, called radiometry, will be discussed in Chapter 9.

The basic model for the projection of points in the scene onto the image plane is diagrammed in Figure 1.4. In this model, the imaging system's center of projection coincides with the origin of the three-dimensional coordinate system. The coordinate system for points in the scene is the three-dimensional space spanned by the unit vectors **x**, **y**, and **z** that form the axes

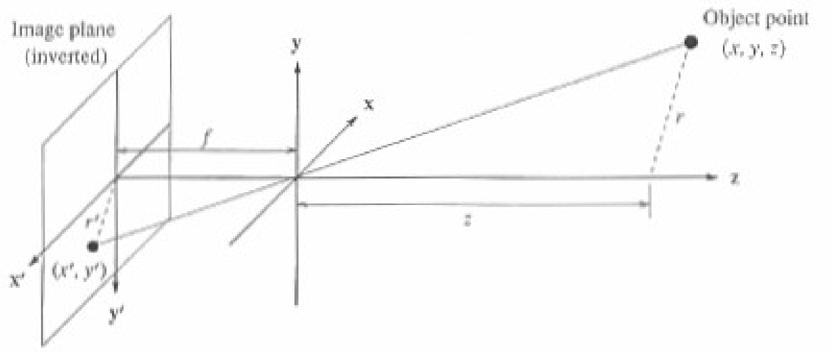


Figure 1.4: The point on the image plane that corresponds to a particular point in the scene is found by following the line that passes through the scene point and the center of projection.

of the coordinate system. A point in the scene has coordinates $(\mathbf{x}, \mathbf{y}, \mathbf{z})$. The x coordinate is the horizontal position of the point in space as seen from the camera, the y coordinate is the vertical position of the point in space as seen from the camera, and the z coordinate is the distance from the camera to the point in space along a line parallel to the **z** axis. The *line of sight* of a point in the scene is the line that passes through the point of interest and the center of projection. The line drawn in Figure 1.4 is a line of sight.

The image plane is parallel to the **x** and **y** axes of the coordinate system at a distance f from the center of projection, as shown in Figure 1.4. Note that the image plane in an actual camera is at a distance of f behind the center of projection (as shown in Figure 1.4) and the projected image is inverted. It is customary to avoid this inversion by assuming that the image plane is in front of the center of projection as shown in Figure 1.5. The image plane is spanned by the vectors $\mathbf{x}'$ and $\mathbf{y}'$ to form a two-dimensional coordinate system for specifying the position of points in the image plane. The position of a point in the image plane is specified by the two coordinates x' and y'. The point $(0, 0)$ in the image plane is the origin of the image plane. The position in the image plane of a point in the scene is found by intersecting the line of sight with the image plane according to the projection scheme as described in the following sections.

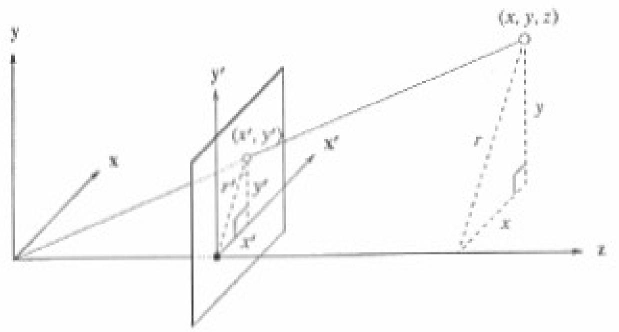


Figure 1.5: An illustration showing the line of sight that is used to calculate the projected point (x', y') from the object point (x, y, z).

1.4.1 Perspective Projection

The position (x', y') in the image plane of a point at position (x, y, z) in the scene is found by computing the coordinates (x', y') of the intersection of the line of sight passing through the scene point (x, y, z) with the image plane as shown in Figure 1.5.

The distance of the point (x, y, z) from the z axis is $r = \sqrt{x^2 + y^2}$, and the distance of the projected point (x', y') from the origin of the image plane is $r' = \sqrt{x'^2 + y'^2}$. The z axis, the line of sight to point (x, y, z), and the line segment of length r from point (x, y, z) to the z axis (perpendicular to the z axis) form a triangle. The z axis, the line of sight to point (x', y') in the image plane, and the line segment of length r' from point (x', y') to the z axis (perpendicular to the z axis) form another triangle. The two triangles are similar, so the ratios of the corresponding sides of the triangles must be the same:

$$\frac{f}{z} = \frac{r'}{r}. \tag{1.2}$$

The triangle formed from the x and y coordinates and the perpendicular distance r and the triangle formed from the image plane coordinates x', y' and the perpendicular distance r' are also similar triangles:

$$\frac{x'}{x} = \frac{y'}{y} = \frac{r'}{r}. \tag{1.3}$$

Combining Equations 1.2 and 1.3 yields the equations for perspective projection:

$$\frac{x'}{x} = \frac{f}{z} \quad \text{and} \quad \frac{y'}{y} = \frac{f}{z}. \tag{1.4}$$

The position of a point (x, y, z) in the image plane is given by the equations

$$x' = \frac{f}{z}x \tag{1.5}$$

$$y' = \frac{f}{z}y. \tag{1.6}$$

1.4.2 Coordinate Systems

In this presentation, we have assumed that the center of projection coincides with the origin of the three-dimensional space and that the camera axes are aligned with the coordinate system used to specify the location of a point in the scene. In general, the camera is displaced and rotated with respect to the three-dimensional coordinate system used for specifying the coordinates of points in the scene. The coordinates (x_a, y_a, z_a) in the absolute coordinate system must be transformed into the coordinates (x_c, y_c, z_c) of the point in the camera coordinate system before projecting the points onto the image plane. The general case is presented in Chapter 12. on calibration, which covers the mathematics of transforming coordinates between coordinate systems. Absolute coordinates are also called world coordinates.

Individual objects may have their own coordinate system, called model coordinates. The scene consists of object models that have been placed (rotated and translated) into the scene, yielding object coordinates in the coordinate system of the scene (absolute coordinates). The scene coordinates in the absolute coordinate system are transformed to camera coordinates before projection onto the image plane.

Throughout the book, subscripts are used to denote the coordinate system, and primes are used to denote the image plane coordinates of a point after projection onto the image plane. Subscripts are omitted when the coordinate system is clear, and the primes are omitted when it is clear that coordinates are in the image plane. These conventions allow us to be precise about the coordinate system when necessary and to distinguish between a point in camera coordinates and its projection onto the image plane, but they allow the notation to be simplified to the common usage for point coordinates in analytic geometry when we are presenting the equations for curves and surfaces.

In keeping with the convention, we will omit primes from image plane coordinates for the rest of this chapter and for Chapters 2 through 6, which cover machine vision algorithms that are restricted to the image plane.

1.5 Sampling and Quantization

A continuous function cannot be represented exactly in a digital computer. The interface between the optical system that projects a scene onto the image plane and the computer must sample the image at a finite number of points and represent each sample within the finite word size of the computer. This is sampling and quantization. Each image sample is called a pixel.

We will initially assume that images are sampled on a regular grid of squares so that the horizontal and vertical distances between pixels are the same throughout the image. We will generalize this assumption in Chapter 12. Each pixel is represented in the computer as a small integer. Frequently, the pixel is represented as an unsigned 8-bit integer in the range $[0, 255]$, with 0 corresponding to black, 255 corresponding to white, and shades of gray distributed over the middle values.

Many cameras acquire an analog image, which is then sampled and quantized to convert it to a digital image. The sampling rate determines how many pixels the digital image will have (the image resolution), and quantization determines how many intensity levels will be used to represent the intensity value at each sample point. As shown in Figures 1.6 and 1.7, an image looks very different at different sampling rates and quantization levels. In most machine vision applications, the sampling and quantizing rates are predetermined due to the limited choice of available cameras and image acquisition hardware; but in many applications it may be important to know the effects of sampling and quantizing. The image processing book [121] discusses the factors that should be considered in selecting appropriate sampling and quantization rates to retain the important information in images.

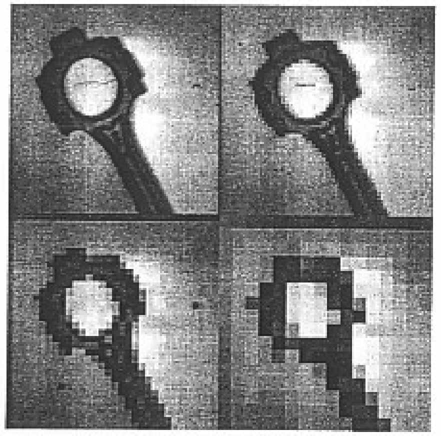


Figure 1.6: An image shown at many different spatial resolutions. *Top left:* Original image sampled at 256×256 and 128 gray levels. *Top right:* 64×64. *Bottom left:* 32×32. *Bottom right:* 16×16.

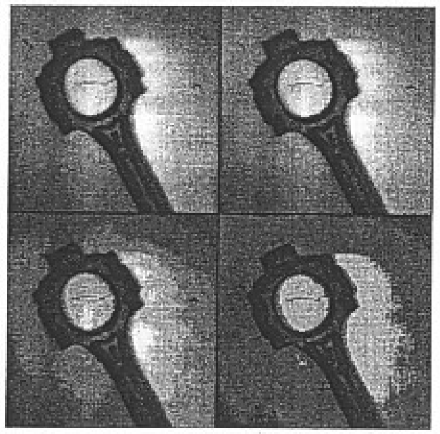


Figure 1.7: An image shown at many different gray level resolutions. *Top left:* Image sampled at 32 gray levels. *Top right:* 16 gray levels. *Bottom left:* 8 gray levels. *Bottom right:* 4 gray levels.

1.6 Image Definitions

It is important to understand the relationship between the geometry of image formation, described in previous sections, and the representation for images in the computer. There must be a bridge from the mathematical notation used to develop machine vision algorithms to the algorithmic notation used in programs.

A pixel is a sample of the image intensity quantized to an integer value. An image is a two-dimensional array of pixels. The row and column indices $[i,j]$ of a pixel are integer values that specify the row and column in the array of pixel values. Pixel $[0,0]$ is located at the top left corner of the image. The index i points down, and j points to the right. This index notation corresponds closely to the array syntax used in computer programs. The positions of points in the image plane have x and y coordinates. The y coordinate corresponds to the vertical direction, and the x coordinate corresponds to the horizontal direction. The y axis points up, and the x axis points to the right. Note that the directions corresponding to the two indices i and j in the pixel index $[i,j]$ are the reverse of the directions corresponding to the respective coordinates in the position (x,y).

The x and y coordinates are real numbers, stored as floating-point numbers in the computer. Image plane coordinates (x,y) can be computed from pixel coordinates $[i,j]$ of an $m \cdot n$ pixel array using the formulas

$$x = j - \frac{m-1}{2} \tag{1.7}$$

$$y = -\left(i - \frac{n-1}{2}\right), \tag{1.8}$$

which assume that the origin of the image plane coordinate system corresponds to the center of the image array.

In an imaging system, each pixel occupies some finite area on the image plane. Machine vision algorithms that depend on the exact shape of the pixel footprint will not be covered in this book, so we may assume for concreteness that the pixels partition the image plane into equal-sized squares. Positions in the image plane can be represented to fractions of a pixel. The coordinates (x_{ij}, y_{ij}) of the pixel with indices $[i,j]$ are the location of the center of the pixel in the coordinate system of the image plane. Since we are concerned only with the location of the center of the pixel in the image plane (the location at which the image sample was taken), the pixel may be further abstracted to a point in the image plane. The array of pixels in the computer program corresponds to the grid of image plane locations at which the samples were obtained, as illustrated in Figure 1.8.

In diagramming images, we may show the image as a square tessellation of a rectangular region, with each square shaded to indicate the image intensity of that pixel. This is purely a technique for visualization and does not imply that the shape of the pixel matters to the algorithm being discussed. For the purposes of nearly all of the algorithms discussed in this book, the image can be modeled as a square grid of samples of the image intensity, represented in the computer as an array of pixel values. The camera and digitizing electronics are designed to ensure that this assumption is satisfied. Some variations, such as different spacing between the rows and columns in the grid, distortions due to lens imperfections, and errors in the construction of the camera, can be removed through calibration without changing the algorithms that process the image (see Section 12.9).

To summarize, a pixel is both a gray value, which is a quantized sample of the continuous image intensity, and an image location, specified as the row and column indices in the image array. The image array is obtained by sampling the image intensity at points on a rectangular grid. Points in the image plane, specified with coordinates x and y, may lie between the grid locations at which pixels were sampled.

1.7 Levels of Computation

An image usually contains several objects. A vision application usually involves computing certain properties of an object, not the image as a whole. To compute properties of an object, individual objects must first be identified as separate objects; then object properties can be computed by applying calculations to the separate objects.

Definitions and algorithms for connectivity and segmentation that will allow different objects to be represented as distinct subimages will be presented in Chapters 2 and 3. For now, consider computer vision algorithms from the viewpoint of locality of computation. Consider each algorithm in terms of its input-output characteristics. Here our aim is to characterize operations so that we can discuss the nature of input and output and how best

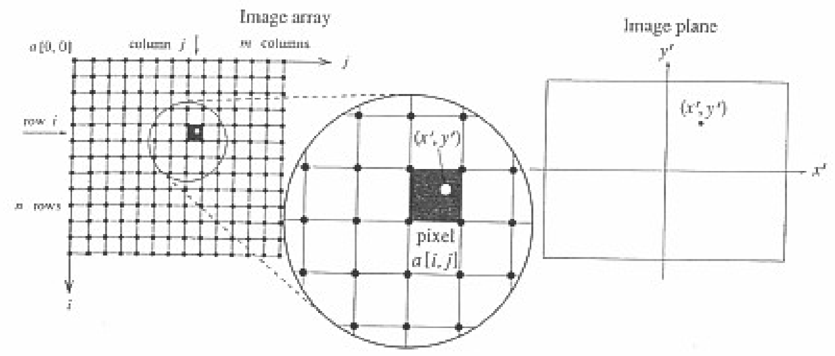


Figure 1.8: Relationship between image plane coordinates and image array indices. Note that the location of the origin of the x–y plane is arbitrary with respect to the image array.

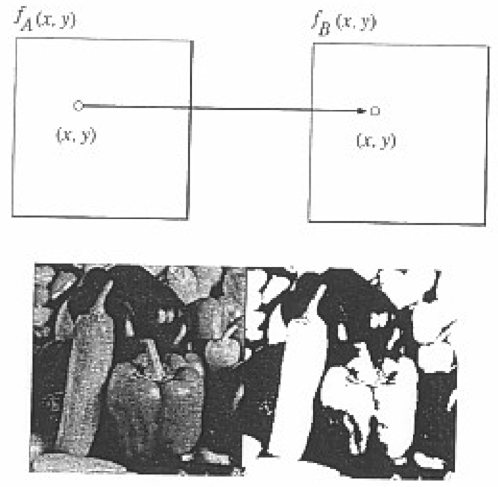
to implement these operations. Note that the input to a computer vision system is an image, and the output, unlike that of image processing systems, is some symbolic quantity denoting identity or location of an object, for instance. The amount of data processed by a vision system is very large, and that makes the computational requirements of a computer vision system very demanding. The last few years have witnessed many special architectures designed for computer vision. Since we want to discuss characteristics of operations to predict their computational requirements, we classify the levels of operations and study their general characteristics.

1.7.1 Point Level

Some operations produce an output based on only a point in an image. Thresholding is an example. A thresholding algorithm produces output values that depend only on the input value, for a preset threshold. Thus,

$$f_B[i,j] = O_{\text{point}}\{f_A[i,j]\}, \tag{1.9}$$

where f_A and f_B are the input and output images, respectively. This operation can be efficiently implemented using a lookup table (see Figure 1.9).



$$f_B[i,j] = O_{\text{point}}\{f_A[i,j]\}$$

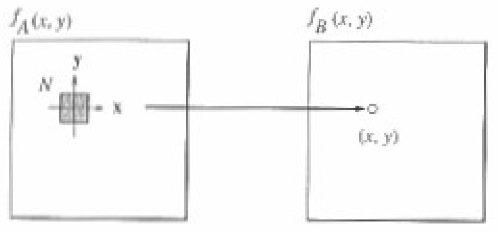
Figure 1.9: *Top:* Point operations are applied to individual image pixels and produce an output image as the result. *Bottom left:* Original image. *Bottom right:* Thresholded image where pixels from the original image with a gray level value greater than 128 are set to white, while the remaining pixels are set to black.

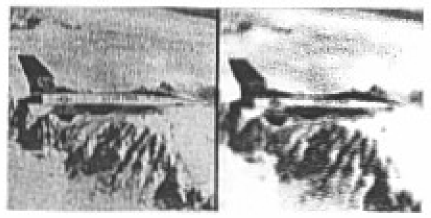
1.7.2 Local Level

A local operation produces an output image in which the intensity at a point depends on the neighborhood of the corresponding point in the input image. Thus,

$$f_B[i,j] = O_{\text{local}}\{f_A[i_k,j_l]; [i_k,j_l] \in N[i,j]\}. \tag{1.10}$$

An example of such an operation is shown in Figure 1.10. Smoothing and edge detection are local operations. Since these operations require values from a neighborhood in the input image, array processors or Single Instruction, Multiple Data (SIMD) machines may be suitable for implementing these operations. In general, these operations can be easily implemented on parallel machines and can often be performed in real time.





$f_D[i,j] = O_{\text{local}}\{f_A[i_k, j_l]; \quad [i_k, j_l] \in N[i,j]\}$

Figure 1.10: *Top:* Local operations are applied to pixel neighborhoods and produce an output image as the result. *Bottom left:* Original image. *Bottom right:* Smoothed image where each pixel value is the average gray value calculated from its 5×5 local neighborhood in the original image.

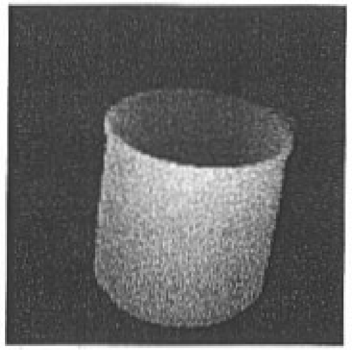
1.7.3 Global Level

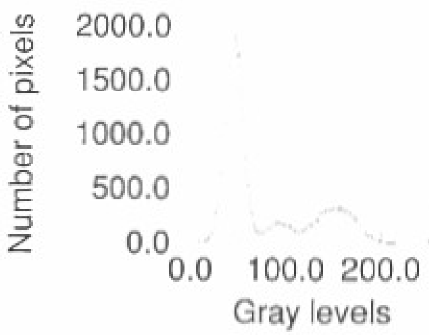
The output of certain operators depends on the whole picture. Such operations are called global operations:

$$P = O_{\text{global}}\{f[i,j]\}. \tag{1.11}$$

This operation is shown in Figure 1.11.

The output of these operators may be an image or it may be symbolic output. A histogram of intensity values and the Fourier transform are global operations. Global operations are responsible for the slowness of vision systems. We will see that most operations at higher levels are global in nature and pose the biggest challenge to designers of algorithms and architectures.





$P = O_{\text{global}}\{f[i,j]\}$

Figure 1.11: An example of a global operation: an image (left) and its histogram (right). A histogram is a plot of the number of pixels at each gray value contained in the image.

1.7.4 Object Level

Most applications of computer vision require properties to be computed at the object level. Size, average intensity, shape, and other characteristics of an object must be computed for the system to recognize it. Many other characteristics of an object must be determined for defect detection. Operators restricted to those pixels belonging to an object are occasionally applied to determine these properties. This leads to very difficult questions: What is an object? How do we find objects?

We will see that an object is defined in a particular context. In fact, many operations in machine vision are performed to find where a particular object is located in an image. Objects in images pose a *catch-22* situation. We must use all points that belong to an object to compute some of its characteristics, but we must use those characteristics to identify those points. We will see that significant efforts are spent to solve the *figure-ground* problem (separation of foreground pixels from background pixels) to group points into objects.

At this point, we should just remember that to understand the contents of an image, a machine vision system must perform several operations at the object level.

1.8 Road Map

This book provides a practical introduction to machine vision. The chapters are presented in a sequence that builds the application of machine vision from the simple binary world to the spatiotemporal world that we want our systems to understand. Roughly, the first half of the book works primarily in two dimensions, and the second half expands the coverage to 3-D. Chapter 2, on binary images, introduces basic terms and concepts used in machine vision. The techniques discussed here are used in all aspects of a vision system. We introduce the morphological approach here as well. Chapter 3 presents techniques to find regions in images. This chapter also covers techniques to represent regions. All vision systems have to use regions for their tasks, and hence the techniques covered in this chapter are essential in implementing systems. Chapter 4, on filtering, introduces techniques to enhance images and discusses fundamental aspects of filtering techniques. Edge detection techniques are very important and are an essential step in many machine vision systems. Chapter 5, on edge detection, introduces several edge detection techniques to show how such a simple operation involves complex processes. The next chapter presents techniques to represent contours. Edges are local; to use them one must group them into meaningful objects and represent them. Texture also plays an important role in many machine vision tasks such as surface inspection, scene classification, surface orientation, and shape determination. These topics are discussed in Chapter 7. The essentials of geometrical optics and radiometry are covered in Chapter 8 and 9 respectively. Color is an essential component of images in many applications *and is discussed in Chapter 10.*

Chapter 11 presents techniques to recover depth information from images using passive and active methods. The techniques discussed here take us to the three-dimensional from *two-dimensional* images. In the next chapter, we present techniques for camera calibration. To recover three-dimensional information from images, it is essential that we know the location and orientation of the camera and its parameters. Various techniques for camera calibration are discussed in this chapter. Representation and characteristics of curves and surfaces in space are discussed in Chapter 13. Techniques for interpolation and approximation are discussed in this chapter. Chapter 14 discusses dynamic vision. Techniques to detect changes in images, segment images based on motion characteristics, and track objects are presented. Structure from motion is briefly introduced here. Object recognition is one of the most common tasks that a vision system performs. We present fundamental aspects of object recognition in Chapter 15.

Most of the material included here can be covered in a semester. In a quarter course, some chapters may have to be omitted. In fact, the book may form a very good two-quarter sequence with inclusion of some advanced material. The sequence of chapters in a particular course depends on the application emphasis of the course. The first six chapters and Chapter 15, on object recognition, cover basic techniques, and we believe that they should be covered independent of the emphasis. In courses emphasizing image applications, Chapters 10 and 7 must also be covered. Courses emphasizing robotics and other three-dimensional information should cover depth, Chapter 11, and camera calibration, Chapter 12. People oriented toward mobile robotics and video will find Chapter 14 essential. Those interested in visualization, reverse engineering, and inspection will benefit from the material covered in Chapter 13, on curves and surfaces.

Our recommendation is to use the first six chapters in the sequence in which they are presented and then select later chapters based on your interest and emphasis. The chapters on camera calibration and curves and surfaces are more mathematical in nature, due to the nature of the problems addressed, than other chapters.

We consider the exercises a very important part of the course. We have presented conceptual problems, regular problems to test the understanding of techniques presented here, and programming exercises. We recommend that this course be made a laboratory-oriented course. Students should implement and experiment with many of the techniques that are presented here. Projects to implement simple vision systems are very important to a real understanding of machine vision techniques.

Further Reading

Machine vision is still a very active research area. Several conference proceedings (*Conf. on Computer Vision and Pattern Recognition, Int. Conf. on Computer Vision, Int. Conf. on Pattern Recognition, Int. Conf. on Robotics and Automation, Int. Conf. on Document Analysis and Recognition, Workshop on Computer Vision*, and numerous conferences of *SPIE*) describe the state of the art in this area. Several archival journals cover this research area (*IEEE Trans. Pattern Analysis and Machine Intelligence; Computer Vision, Graphics, and Image Processing; IEEE Trans. Image Processing; IEEE Trans. Systems, Man, and Cybernetics; Machine Vision and Applications; Int. J. Computer Vision; J. Opt. Soc. Am. A; Image and Vision Computing*; and *Pattern Recognition*) and are very good sources for studying the state of the art. Many special issues of journals, proceedings of conferences and workshops on special topics, and research books are also published every year, and these are usually good sources of information on particular aspects of machine vision. A list of such sources is provided in [140, pp. 702–706]. The volumes [140, 139] provide a collection of the best papers from the last decade.

Several introductory books have been written on this topic. Rosenfeld and Kak [206], Gonzalez and Woods [90], Pratt [196], Ballard and Brown [19], Horn [109], Jain [121], Levine [155], Wechsler [248], Nalwa [177], Haralick and Shapiro [103], and Schalkoff [211] are some of the books that have been used in introductory courses and by people interested in self-learning. The topics covered in this chapter are discussed in many books. Aspects of digitization are usually covered at length in books on image processing [121]. There are several excellent books on image processing [121, 196, 91] and computer graphics [81, 204] that provide information useful in machine vision. Readers interested in knowing more about pattern recognition should refer to [70]. There are also many good books on artificial intelligence [201, 251].

Readers interested in reading about human vision may find that some books in the above list provide information on this topic also. A few particularly good sources for human vision from a computational perspective are Marr [163], Mayhew and Frisby [166], Rock [203], and Gibson [86, 87]. Many books cover special application areas. A list of such books is provided in [140, 139]. Readers interested in computational vision with strong emphasis on human vision should see [155, 163].

Exercises

1.1 We interpret a variety of images effortlessly. In each of these cases, many types of knowledge are used. Consider the following applications and try to list all possible sources of knowledge that you use in

interpretation of these images.

a. Reading English text

b. Image of an indoor scene

c. Image of a road scene

d. Image of an airport

e. Remotely sensed image of an area

f. Microscopic image of an organism

g. X-ray image

h. Other

1.2 List the levels of computations on images. Give two examples of each level of computation.

1.3 Determination of the distance to different objects in a scene is an important operation in machine vision. Many techniques will be discussed in this book. Here let us consider the simple projection model. In stereo vision, two cameras are used to determine the distance to a point using triangulation. Assume that you have two cameras and they are placed such that their optical axes are parallel to each other. There is a prominent point P in the image such that its projections in two images are P_1 and P_2, respectively. Assuming that you know the location of the cameras and that the point appears in both images, how can you determine the distance to this point using perspective projections? Derive the relationship you can use to determine the distance. If you have freedom to select the location of the cameras, will you place them close to each other or far from each other? Justify your answer.

1.4 For a given picture, assume that you sample it using several sampling rates to represent it as an $N \times N$ array of pixels. At each pixel, you use the same number of bits to represent intensity values. Prepare a table showing the memory requirements for the image at different resolutions. Start at 16×16 and go to 4096×4096. Plot the memory requirements against N.

1.5 In the above exercise, now assume that pixels can be represented using different numbers of bits. One may use only 1 bit to represent a binary image and 24 bits to represent a color image. Though in most applications, single-channel intensity values are represented using only 8 bits, one may use 16 or more bits. Plot memory requirements for different numbers of bits; assume a fixed array size of 512×512 pixels.

1.6 Scan a picture using a tabletop scanner. You can select different sampling rates, expressed usually as dots per inch (dpi), and different intensity or color levels. Scan the image at many different resolutions and intensity ranges. See the memory requirements. Considering the memory requirements, at what resolution and number of intensity level will you save the image?

Now take at least two other images. These images should contain different amounts of detail. Repeat the above experiment. Would you store all these images using the same parameters or would you use different parameters? Why?

1.7 A surveillence camera is embedded in one of the walls of a room as shown in Figure 1.12. The optical axis of the camera is perpendicular to the wall, and the lens center is in the plane of the wall. The focal length of the lens is 0.05 meters. The x–z plane of the camera is parallel to the x–y plane of the world coordinate system. The image plane is behind the wall. Find the image plane coordinates of **(a)** the room corner A, and **(b)** the head of a person 2 meters tall standing at a distance of 3 m × 2 m from the corner as shown in Figure 1.12.

Computer Projects

1.1 Imaging geometry:

a. Consider a camera with the origin of its image plane located at (x_0, y_0, z_0) with reference to the world coordinates as shown in Figure 1.13. The camera axis passes through a point (x_c, y_c, z_c) in world coordinates. Assume that the x′ axis of the camera coordinate system is parallel to the x–y plane. The lens center is

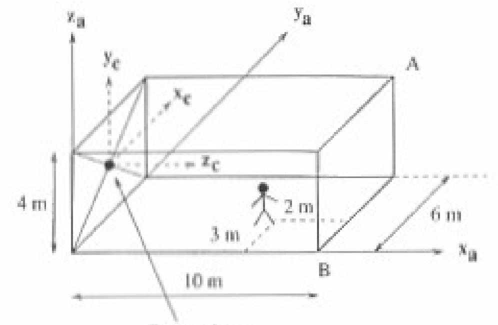


Figure 1.12: Diagram of surveillence camera and room setup for Exercise 1.7.

located at a distance of f from the origin of the image coordinates along the z′ direction.

b. A polyhedral object with vertices $P_1, P_2, \ldots$ is located in the world coordinate system at (x_1, y_1, z_1), $(x_2, y_2, z_2), \ldots$, respectively. The number of edges in the object is N. Assume that the pairs of vertices which are connected are known.

c. Write a program which will generate the image formed on the image plane. Assume that the object is a "stick figure" (i.e., you need not be concerned about hidden line removal). The image plane is limited in extent to $+1$ and -1 along both the x and y directions. Rescale this image to a 256×256 image.

d. In summary, your program should do the following:

- Ask for data file name.
- Read (x_0, y_0, z_0), (x_C, y_C, z_C), and f.
- Read the number of edges and vertices in the object.
- Read the coordinates of each vertex.
- Read the list of vertices which are connected to each other.

For example, consider a unit cube with one of its corners located at the origin and three of its edges along the positive x, y, z directions. Let the camera be located at (10, 10, 10) and pointing

In the above exercise, now assume that pixels can be represented using different numbers of bits. One may use only 1 bit to represent a binary image and 24 bits to represent a color image. Though in most applications, single-channel intensity values are represented using only 8 bits, one may use 16 or more bits. Plot memory requirements for different numbers of bits; assume a fixed array size of 512×512 pixels.

Scan a picture using a tabletop scanner. You can select different sampling rates, expressed usually as dots per inch (dpi), and different intensity or color levels. Scan the image at many different resolutions and intensity ranges. See the memory requirements. Considering the memory requirements, at what resolution and number of intensity level will you save the image?

Now take at least two other images. These images should contain different amounts of detail. Repeat the above experiment. Would you store all these images using the same parameters or would you use different parameters? Why?

A surveillence camera is embedded in one of the walls of a room as shown in Figure 1.12. The optical axis of the camera is perpendicular to the wall, and the lens center is in the plane of the wall. The focal length of the lens is 0.05 meters. The **x**–**z** plane of the camera is parallel to the **x**–**y** plane of the world coordinate system. The image plane is behind the wall. Find the image plane coordinates of (a) the room corner A, and (b) the head of a person 2 meters tall standing at a distance of 3 m × 2 m from the corner as shown in Figure 1.12.

mputer Projects

Imaging geometry:

a. Consider a camera with the origin of its image plane located at (x_0, y_0, z_0) with reference to the world coordinates as shown in Figure 1.13. The camera axis passes through a point (x_c, y_c, z_c) in world coordinates. Assume that the **x**′ axis of the camera coordinate system is parallel to the **x**–**y** plane. The lens center is

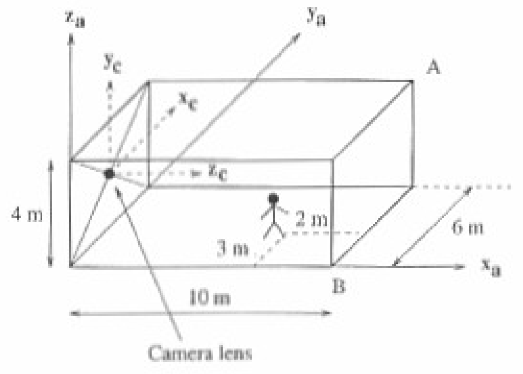


Figure 1.12: Diagram of surveillence camera and room setup for Exercise 1.7.

located at a distance of f from the origin of the image coordinates along the **z**′ direction.

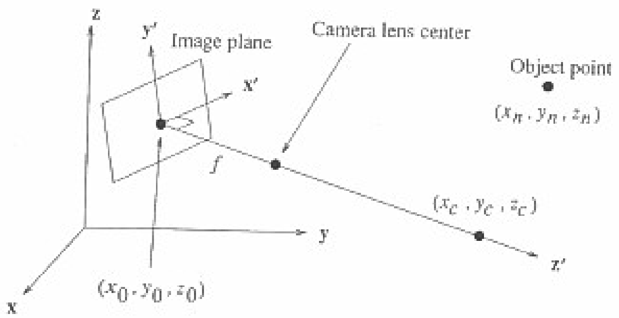
b. A polyhedral object with vertices $P_1, P_2, \ldots$ is located in the world coordinate system at $(x_1, y_1, z_1), (x_2, y_2, z_2), \ldots$, respectively. The number of edges in the object is N. Assume that the pairs of vertices which are connected are known.

c. Write a program which will generate the image formed on the image plane. Assume that the object is a "stick figure" (i.e., you need not be concerned about hidden line removal). The image plane is limited in extent to +1 and −1 along both the x and y directions. Rescale this image to a 256×256 image.

d. In summary, your program should do the following:

- Ask for data file name.
- Read (x_0, y_0, z_0), (x_C, y_C, z_C), and f.
- Read the number of edges and vertices in the object.
- Read the coordinates of each vertex.
- Read the list of vertices which are connected to each other.

For example, consider a unit cube with one of its corners located at the origin and three of its edges along the positive **x**, **y**, **z** directions. Let the camera be located at (10, 10, 10) and pointing



re 1.13: The camera coordinate system for Computer Project 1.1.

towards $(0, 0, 0)$. Let the camera's f be 3. The corresponding data file will be as follows:

(x_0, y_0, z_0), (x_C, y_C, z_C), and f:	10	10	10	0	0	0	3
Number of edges and vertices:	12	8					
Coordinates of each vertex:	0	0	0				
	0	0	1				
	0	1	0				
	1	0	0				
	1	1	1				
	1	1	0				
	1	0	1				
	0	1	1				
Connected vertices:	1	2					
	1	3					
	1	4					
	5	6					
	5	7					
	5	8					
	2	7					
	2	8					
	3	6					
	3	8					
	4	6					
	4	7					

Chapter 2

Binary Image Processing

An image contains a continuum of intensity values before it is quantized to obtain a digital image. The information in an image is in these gray values. To interpret an image, the variations in the intensity values must be analyzed. The most commonly used number of quantization levels for representing image intensities is 256 different gray levels. It is not uncommon, however, to see digital images quantized to 32, 64, 128, or 512 intensity levels for certain applications, and even up to 4096 (12 bits) are used in medicine. Clearly, more intensity levels allow better representation of the scene at the cost of more storage.

In the early days of machine vision, the memory and computing power available was very limited and expensive. These limitations encouraged designers of vision applications to focus their efforts on binary vision systems. A binary image contains only two gray levels. The difference this makes in the representation of a scene is shown in Figure 2.1.

In addition, designers noted that people have no difficulty in understanding line drawings, silhouettes, and other images formed using only two gray levels. Encouraged by this human capability, they used binary images in many applications.

Even though computers have become much more powerful, binary vision systems are still useful. First of all, the algorithms for computing properties of binary images are well understood. They also tend to be less expensive and faster than vision systems that operate on gray level or color images. This is due to the significantly smaller memory and processing requirements of binary vision. The memory requirements of a gray level system working