Chapter 1.

Introduction to Computer Graphics

1.1 What is computer graphics?

People use the term "computer graphics" to mean different things in different contexts. Most simply, computer graphics are pictures that are generated by a computer. Everywhere you look today there are examples to be found, especially in magazines and on television. Books and magazines abound with pictures created on a computer. Some look so natural you can't distinguish them from photographs of a "real" scene. Others have an artificial or surreal feeling, intentionally fashioned to achieve some visual effect. And movies today often show scenes that never existed, but were carefully crafted by computer, mixing the real and the imagined.

"Computer graphics" also refers to the tools used to make such pictures. There are both hardware and software tools. Hardware tools include video monitors and printers that display graphics, as well as input devices like a mouse or trackball that let a user point to items and draw figures. The computer itself, of course, is a hardware tool, along with its special circuitry to facilitate graphical display or image capture.

As for software tools, you are already familiar with the usual ones: the operating system, editor, compiler, and debugger, which are found in any programming environment. For graphics there must also be a collection of "graphics routines" that produce the pictures themselves. For example, all graphics libraries have functions to draw a simple line or circle (or characters such as G). Some go well beyond this, containing functions to draw and manage windows with pull-down menus and dialog boxes, or to set up a "camera" in a three-dimensional coordinate system and to make "snapshots" of objects stored in some data base.

Finally, "computer graphics" often means the whole field of study that involves these tools and the pictures they produce. (So it's also used in the singular form: "computer graphics is...").

1.2 Where are Computer Graphics Used?

1.2.1 Art, Entertainment, and Publishing

- Computer graphics are widely used in the production of movies, television programs, books, games, and magazines.
- **Browsing on the World Wide Web:** the browser must rapidly interpret the data on a page and draw it on the screen as high quality text and graphics.
- **Slide, Book, and Magazine Design:** Computer graphics are used in **page layout** programs to design the final look of each page of a book or magazine. The user can interactively move text and graphics around to find the most pleasing arrangement.
- A **paint system** generates images. A common example of a paint system and photo manipulation system is *Adobe Photoshop*®.

1.2.2 Computer Graphics and Image Processing

- Computer graphics create pictures and images based on some description, or model, in a computer.
- Image processing improves or alters images that were created elsewhere.
    - Processing can remove noise from an image, enhance its contrast, sharpen its edges, and fix its colors.
    - Software routines can search for certain features in an image, and highlight them to make them more noticeable or understandable.

1.2.3 Process Monitoring

- Highly complex systems such as air traffic control systems must be monitored by a human to watch for impending trouble.
- An air traffic control system consists of monitors that display where nearby planes are situated.
    - The user sees a schematic representation for the process, giving the whole picture at a glance.
    - Various icons can flash or change color to alert the user to changes that need attention.

1.2.4 Displaying Simulations

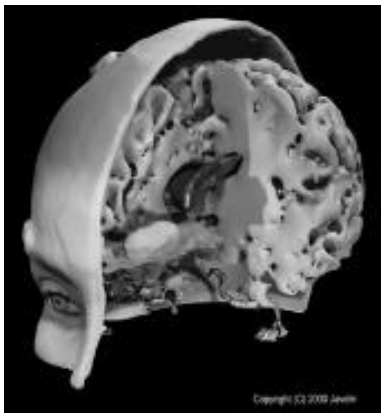
- **Flight simulator**: the system is a plane with a shape and flying characteristics, along with a world consisting of a landing field, mountains, other planes, and air, all modeled appropriately.

1.2.5 Computer-aided Design

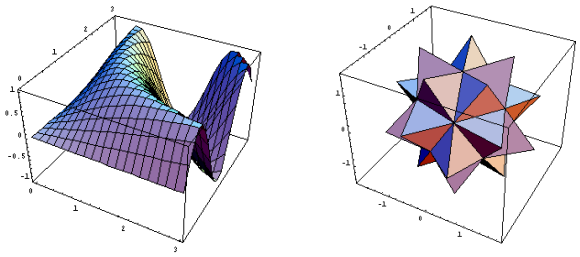
- E.g., drills, or houses. The computer version is easy to alter if necessary.
    - Analysis and simulation can be used also. The shape of the drill might look nice, but the casing might be too weak or too heavy, or might be uncomfortable to grip.
    - Algorithms can be applied to the model of the drill to analyze its weight and heft, and to test whether the inner workings of the drill will fit properly inside the casing.

1.2.6 Volume Visualization

- An image of a 3D object can be constructed by creating a number of slices. This approach has given rise to a special field of study within computer graphics called **volume visualization.**



1.2.7 Displaying Mathematical Functions



- E.g., Mathematica, MatLab

1.3 Elements of Pictures Created in Computer Graphics

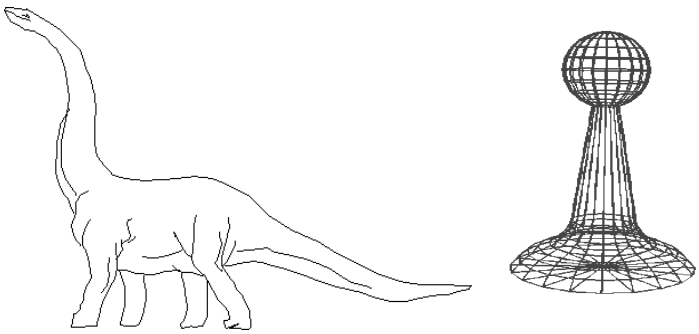
What makes up a computer drawn picture? The basic objects out of which such pictures are composed are called output primitives. One useful categorization of these is:

- polylines
- text
- filled regions
- raster images

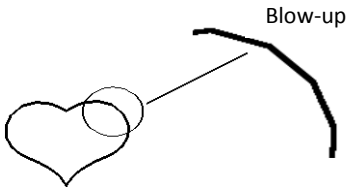
We will see that these types overlap somewhat, but this terminology provides a good starting point. The attributes of a graphic primitive are the characteristics that affect how it appears, such as color and thickness.

1.3.1 Polylines.

A polyline is a connected sequence of straight lines. Each of the examples in the figure below contains several polylines. In the first figure, one polyline extends from the nose of the dinosaur to its tail. And the “wireframe” picture of a chess pawn contains many polylines that outline its shape.



Note that a polyline can appear as a smooth curve. The figure below shows a blow-up of a curve revealing its underlying short line segments. The eye blends them into an apparently smooth curve



Pictures made up of polylines are sometimes called line drawings. Some devices, like a pen plotter, are specifically designed to produce line drawings. The simplest polyline is a single straight line segment. A line segment is specified by its two endpoints, say (x1, y1) and (x2, y2). A drawing routine for a line might look like

```
drawLine(x1, y1, x2, y2);
```

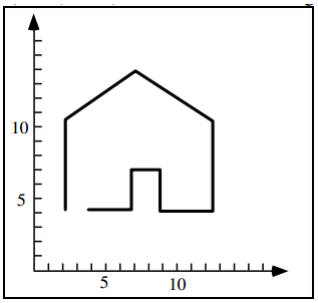
It draws a line between the two endpoints.

A special case arises when a line segment shrinks to a single point, and is drawn as a "dot". Even the lowly dot has important uses in computer graphics, as we see later. A dot might be programmed using the routine

```
drawDot(x1, y1);
```

When there are several lines in a polyline each one is called an edge, and two adjacent lines meet at a vertex. The edges of a polyline can cross one another, as seen in the figures. Polylines are specified as a list of vertices, each given by a coordinate pair: (x0, y0), (x1, y1), (x2, y2), ..., (xn, yn)

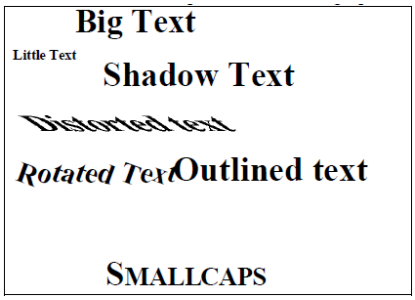
For instance, the polyline shown in the figure below is given by the sequence (2, 4), (2, 11), (6, 14), (12, 11), (12, 4), ... (what are the remaining vertices in this polyline?).



Important attributes of a polyline are the color and thickness of its edges, the manner in which the edges are dashed, and the manner in which thick edges blend together at their endpoints. Typically all of the edges of a polyline are given the same attributes.

1.3.2 Text

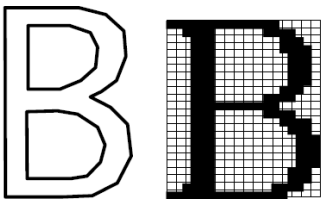
Some graphics devices have two distinct display modes, a **text mode** and a **graphics mode**. The text mode is used for simple input/output of characters to control the operating system or edit the code in a program. Text displayed in this



mode uses a built-in character generator. The character generator is capable of drawing alphabetic, numeric, and punctuation characters, and some selection of special symbols such as **©, ð,** and ⊕. Usually these characters can't be placed arbitrarily on the display but only in some row and column of a built-in grid.
A graphics mode offers a richer set of character shapes, and characters can be placed arbitrarily. Figure above shows some examples of text drawn graphically. A tool to draw a character string might look like:

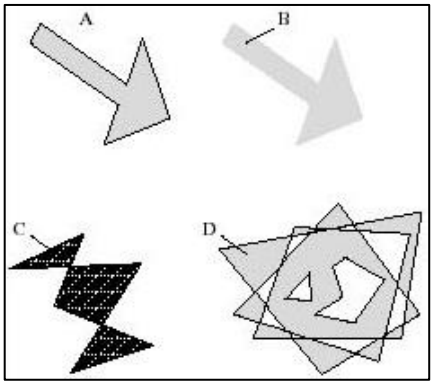
```
drawString(x, y, string);
```

It places the starting point of the string at position (*x*, *y*), and draws the sequence of characters stored in the variable string.



There are many text attributes, the most important of which are typeface, color, size, spacing, and orientation. The shape of each character can be defined by a polyline (or more complicated curves such as Bezier curves), or by an arrangement of dots (see the figure).

1.3.3 Filled Regions



The **filled region** (sometimes called "fill area") primitive is a shape filled with some color or pattern. The boundary of a filled region is often a polygon. The figure on the right shows several filled polygons. Polygon *A* is filled with its edges visible, whereas *B* is filled with its border left undrawn. Polygons *C* and *D* are non-simple. Polygon *D* even contains polygonal holes. Such shapes can still be filled, but one must specify exactly what is meant by a polygon's "interior", since filling algorithms differ depending on the definition.

To draw a filled polygon one would use a routine like:

```
fillPolygon(poly, pattern);
```

where the variable poly holds the data for the polygon - the same kind of list as for a polyline - and the variable pattern is some description of the pattern to be used for filling.

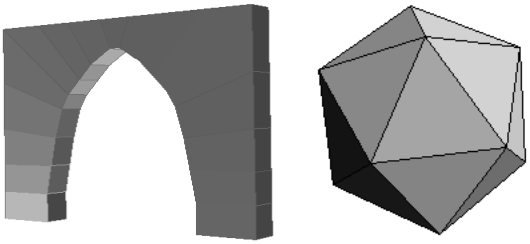
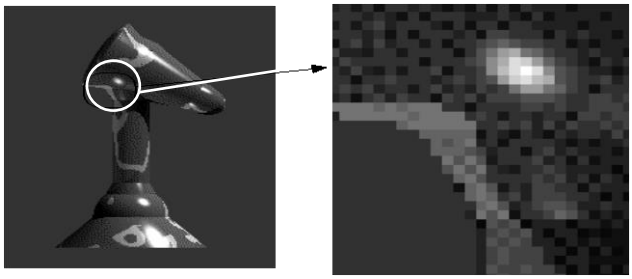


Figure above shows the use of filled regions to shade the different faces of a 3D object. The attributes of a filled region include the attributes of the enclosing border, as well as the pattern and color of the filling.

1.3.4 Raster Image

A **raster image** of a chess piece is shown below. It is made up of many small "cells", in different shades of gray, as revealed in the blow-up. The individual cells are often called "**pixels**" (short for "picture elements"). Normally your eye can't see the individual cells; it blends them together and synthesizes an overall picture.

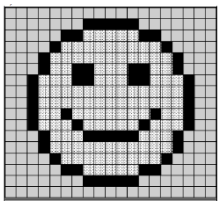


A raster image is stored in a computer as an array of numerical values. This array is thought of as being rectangular, with a certain number of rows and a certain number of columns. Each numerical value represents the value of the pixel stored there. The array as a whole is often called a "**pixel map"**.

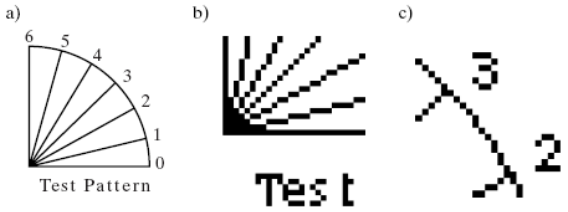
The term "**bitmap"** is also used, (although some people think this term should be reserved for pixel maps wherein each pixel is represented by a single bit, having the value 0 or 1.)

The figure below is represented by a 17 by 19 array (17 rows by 19 columns) of cells in three shades of gray. Suppose the three gray levels are encoded as the values 1, 2, and 7. The numbers show the values in the upper left 6 rows x 8 columns of the image.

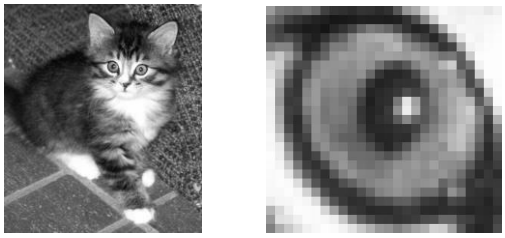
2	2	2	2	2	2	2	2
2	2	2	2	2	2	2	7
2	2	2	2	2	7	7	1
2	2	2	2	7	1	1	1
2	2	2	7	1	1	1	1
2	2	2	7	1	1	7	7



Raster images also frequently contain images of straight lines. A line is created in an image by setting the proper pixels to the line's color. But it can require quite a bit of computation to determine the sequence of pixels that "best fit" the ideal line between two given end points. Bresenham's algorithm (will be discussed later) provides a very efficient approach to determining these pixels. A raster image, shown below, features several straight lines, a circular arc, and some text characters.



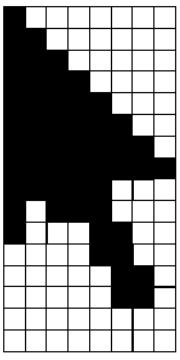
A photograph or television image can be digitized as described above. In effect a grid is placed over the original image, and at each grid point the digitizer reads into memory the "closest" color in its repertoire. The bitmap is then stored in a file for later use. The image of the kitten shown in the figure was formed this way.



Gray-scale Raster Images.

If there are only two pixel values in a raster image it is called *bilevel*. Figure shows a simple bilevel image, representing a familiar arrow-shaped cursor frequently seen on a computer screen. Its raster consists of 16 rows of 8 pixels each. The image shown at the left associates black with a 1 and white with a 0, but this association might just as easily be reversed (sometimes we use 1 for white and 0 for black). Since one bit of information is sufficient to distinguish two values, a bilevel image is often referred to as a "1 bit per pixel" image.

a).



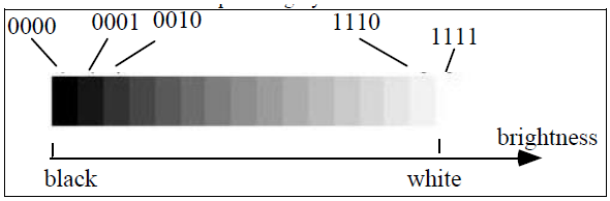
b).

1	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0
1	1	1	0	0	0	0	0
1	1	1	1	0	0	0	0
1	1	1	1	1	0	0	0
1	1	1	1	1	1	0	0
1	1	1	1	1	1	1	0
1	1	1	1	1	1	1	1
1	1	1	1	1	0	0	0
1	0	1	1	1	0	0	0
1	0	0	0	1	1	0	0
0	0	0	0	1	1	0	0
0	0	0	0	0	1	1	0
0	0	0	0	0	1	1	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

When the pixels in a gray-scale image take on more than two values, each pixel requires more than a single bit to represent it in memory. Gray-scale images are often classified in terms of their **pixel depth,** the number of bits needed to represent their gray levels. Since an *n*-bit quantity has 2^n possible values, there can be 2^n gray levels in an image with pixel depth *n*. The most common values are:

- 2 bits/pixel produce 4 gray levels
- 4 bits/pixel produce 16 gray levels
- 8 bits/pixel produce 256 gray levels

Figure below shows 16 gray levels ranging from black to white. Each of the sixteen possible pixel values is associated with a binary 4-tuple such as 0110 or 1110. Here 0000 represents black, 1111 denotes white, and the other 14 values represent gray levels in between.



Many gray scale images employ 256 gray levels, since this usually gives a scanned image acceptable quality. Each pixel is represented by some 8-bit values such as 01101110. The pixel value usually represents "brightness", where black is represented by 00000000, white by 11111111.

Color Raster Images.

Color images are desirable because they match our daily experience more closely than do gray-scale images. Color raster images have become more common in recent years as the cost of high quality color displays has come down. The cost of scanners that digitize color photos has also become reasonable.

Each pixel in a color image has a “color value”, a numerical value that somehow represents a color. There are a number of ways to associate numbers and colors, but one of the most common is to describe a color as a combination of amounts of red, green, and blue light. Each pixel value is a 3-tuple, such as (23, 14, 51), that prescribes the intensities of the red, green, and blue light components in that order.

The number of bits used to represent the color of each pixel is often called its **color depth.** Each value in the (red, green, blue) 3-tuple has a certain number of bits, and the color depth is the sum of these values. A color depth of 3 allows one bit for each component. For instance the pixel value (0, 1, 1) means that the red component is “off”, but both green and blue are “on”. In most displays the contributions from each component are added together, so (0,1,1) would represent the addition of green and blue light, which is perceived as cyan. Since each component can be on or off there are eight possible colors, as tabulated below. As expected, equal amounts of red, green, and blue, (1, 1, 1), produce white.

color value	displayed
0,0,0	black
0,0,1	blue
0,1,0	green
0,1,1	cyan
1,0,0	red
1,0,1	magenta
1,1,0	yellow
1,1,1	white

A color depth of 3 rarely offers enough precision for specifying the value of each component, so larger color depths are used. Because a byte is such a natural quantity to manipulate on a computer, many images have a color depth of eight. Each pixel then has one of 256 possible colors. The highest quality images, known as **true color** images, have a color depth of 24, and so use a byte for each component.