Chapter 2

Overview of Graphics Systems

Exercise questions for this chapter investigate the characteristics and applications of computer-graphics hardware components, the general computer-graphics software concepts, and features of the OpenGL libraries.

Exercises

2-1. Operating characteristics for display technologies are discussed in Sections 2-1 through 2-3 and in the references cited at the end of the chapter.

2-2. Applications appropriate for the various display technologies are discussed in Sections 2-1 through 2-3 and in the listed references.

2-3. A system manual can be consulted to obtain screen dimensions and the number of pixel positions that can be displayed in the x and y directions. If x and y screen dimensions are not listed, they can be measured. The ratio of pixel positions to screen dimensions gives the resolutions for the horizontal and vertical directions. Aspect ratio is the ratio of the y and x resolutions.

Alternatively, resolution can be determined by displaying equal-pixel, straight-line segments in the x and y directions and measuring the length of each line.

For example, if 100-pixel lines in the x and y directions measure 4 cm and 5 cm, respectively, then

$$\text{x resolution} = \frac{100}{4} \text{ pixels per cm}$$

$$\text{y resolution} = \frac{100}{5} \text{ pixels per cm}$$

and

$$\text{aspect ratio} = \frac{\text{y resolution}}{\text{x resolution}} = \frac{4}{5}$$

To adjust dimensions of objects to maintain relative proportions, we can make the adjustments either to the vertical or to the horizontal dimensions. That is, vertical dimensions can be scaled by a factor of 4/5, or horizontal dimensions can be scaled by a factor of 5/4.

2-4. Frame-buffer size for each of the systems is

$$640 \times 480 \times 12 \text{ bits} \div 8 \text{ bits per byte} = 450 \text{ KB}$$

$$1280 \times 1024 \times 12 \text{ bits} \div 8 \text{ bits per byte} = 1920 \text{ KB}$$

$$2560 \times 2048 \times 12 \text{ bits} \div 8 \text{ bits per byte} = 7680 \text{ KB}$$

For 24 bits of storage per pixel, each of the above values is doubled.

2-5. Storage needed for the frame buffer is

$$800 \times 1000 \times 6 \text{ bits} \div 8 \text{ bits per byte} \approx 486 \text{ KB}$$

2-6. The times to load the two frame buffers are

$$640 \times 480 \times 12 \text{ bits} \div 10^5 \text{ bits per sec} \approx 36.9 \text{ sec}$$

and

$$1280 \times 1024 \times 24 \text{ bits} \div 10^5 \text{ bits per sec} \approx 314.6 \text{ sec} \approx 5.24 \text{ min}$$

2-7. Total bits in the printer frame buffer is

$$8.5 \times 11 \times 300^2 \approx 8.4 \times 10^6 \text{ bits}$$

Therefore, loading time is

$$\frac{8.4 \times 10^6 \text{ bits}}{32 \times 10^6 \text{ bps}} \approx 0.263 \text{ sec}$$

2-8. For the 640 by 480 system, the access rate is

$$640 \times 480 \times 60 = 1.8432 \times 10^7 \text{ pixels per sec}$$

Thus, access time is approximately 54 nanoseconds per pixel.

Similarly, for the 1280 by 1024 system, access rate is

$$1280 \times 1024 \times 60 = 7.86432 \times 10^7 \text{ pixels per sec}$$

Here, the access time is approximately 12.7 nanoseconds per pixel.

2-9. The diameter of screen pixels is

$$\frac{12}{1280} = \frac{9.6}{1024} = 9.375 \times 10^{-3} \text{ inches}$$

2-10. The scan rate for each pixel row is

$$60 \text{ frames/sec} \times 1024 \text{ lines/frame} = 61,440 \text{ lines/sec}$$

And the scan time is approximately 16.3 microseconds per scan line. (Scan time per frame is 1/60 sec, or approximately 16.7 milliseconds.)

2-11. Refresh time per frame is $1/r$ seconds, and the total retrace time during refresh of each frame is

$$t_{retrace} = t_{vert} + m \cdot t_{horiz}$$

The fraction of the time spent in retrace is $r \cdot t_{retrace}$.

2-12. The fraction of the total refresh time per frame spent in retrace is

$$60 \times (500 \times 10^{-6} + 1024 \times 5 \times 10^{-6}) \approx 0.337$$

2-13. Total number of available colors is $2^{24} = 16,777,216$. Using a different color for each screen pixel, we could display $512^2 = 262,144$ colors at any one time.

2-14. Three-dimensional monitors and stereographic systems are discussed in Sections 2-1 and in the references.

2-15. Input and output devices used with virtual-reality systems are discussed in Sections 2-4 and 2-5.

2-16. Design applications for VR include interactively constructing systems and interactively studying system operating characteristics. Some VR design applications are given in Sections 1-2, 1-3, 2-1, 2-4, and in the references at the end of Chapters 1 and 2.

2-17. Some applications for large-screen displays are presented in Section 2-3.

2-18. Graphics software packages are discussed in Section 2-8. Basically, packages for graphics programming contain functions for specifying individual geometric components (primitives, such as lines and circles), attributes of individual primitives, and parameters for various graphics operations such as viewing and geometric transformations. A package for a specific application, such as architectural design, allows a user to create scenes in terms of architectural features, such as doors, windows, hallways, stairs, and rooms. Then a building can be rotated or viewed from a given direction, such as from the front, from above, or from the left side.

2-19. The OpenGL core library contains hardware-independent functions, such as those for specifying primitives, attributes, geometric transformations, and three-dimensional viewing parameters. The GLU library contains functions for some other, more specialized operations, such as quadric-surface generation, B-spline surface generation, surface texture mapping, two-dimensional viewing, and some three-dimensional viewing operations. The GLUT library primarily provides hardware-dependent functions, such as those for display-window management and for interacting with input devices, but it also contains functions for generating various plane-surface, quadric-surface, and cubic-surface solids, such as a cube, sphere, cone, or teapot.

2-20. We set the color of a display window to light gray by selecting a value between 0.5 and 1.0 and assigning that value to all three RGB parameters in the `glClearColor` command. For example, we can set the display window to light gray with

```
glClearColor (0.9, 0.9, 0.9, 0.0);
```

Similarly, we obtain dark gray by selecting a value between 0.0 and 0.5 for the RGB color components.

2-21. The following statements set the lower-right corner of a 100 pixel by 75 pixel display window at screen coordinates (200, 200):

```
glutInitWindowPosition (100, 125);
glutInitWindowSize (100, 75);
```

2-22. A callback function is a procedure that is to be invoked whenever a particular action occurs. Such a procedure is registered by listing the procedure name in an appropriate OpenGL function.