Information Technology
Inside and Outside
- David Cyganski & John A. Orr

III. Graphics and Visual Information

6. Computer Graphics and Virtual Reality

Hoon-Jae Lee
hjlee@dongseo.ac.kr
http://cg.dongseo.ac.kr/~hjlee

6. Computer Graphics and Virtual Reality

- Objectives:
  - two fundamentally different ways to store image information (bit maps or scene descriptions);
  - the means of rendering stored image information into a visible image;
  - types of display devices, their features and limitations;
  - fundamentals of the Virtual Reality Modeling Language (VRML); and
  - basic means of making the artificial appear real (lighting, texture mapping, etc.).

6.1 Synthesizing Images

- In **synthesizing images**, it is only necessary to have the information describing aspects of the object, want to see in the image.
  - ***Vector graphics***, allows images such as mechanical drawings to be produced.
    ➔ This concept can be extended to describe many more features of the desired objects, such as surface color and texture, to permit so-called ***virtual reality*** images.
- The **Virtual Reality Modeling Language (VRML)** provides a set of capabilities to permit the construction, via a set of computer instructions, of a synthetic world through which the viewer may move.
- The **display device** on which the image will be produced or ***rendered***. This information includes such aspects as resolution in dots per inch, and color capabilities.

6.2 Two Ways to Store Images

- Two different representations of an image in computer:
  - **Vector graphics** ➔ By ***storing*** the light intensity (and possibly the color) of each point in the image
    - ✓ **Vector graphics** began with computerization of hand-drawn mechanical drawings, and has broadened in scope to become a central part of what is referred to as Virtual Reality.
  - **Bit-mapped graphics** ➔ By ***describing*** the image content.
    - ✓ ***vector graphics*** approach, and it is commonly used for Computer-Aided Design (**CAD**) software.
    - ✓ a **circle** might be represented by its **center**, its **radius**, its **color**, and the **width of the line** used to draw the circle.
- In the **digital photograph approach** (known as ***bit-mapped graphics***), the image is quantized spatially and with respect to the number of colors(or grey levels). The amount of storage required is determined by the number of pixels and the number of different colors (and/or light intensities).

6.2 Two Ways to Store Images(2)

- The circle: center at(0,0), radius of 20 mm, line width of 1 mm, and color black.(**Figure 6.1**)
- The intensity (black or white in this case) at each point in the image. (**Figure 6.2**)

Figure 6.1: Representation of a circle using vector graphics.

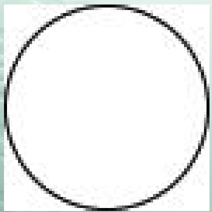
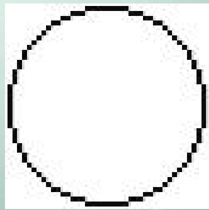


Figure 6.2: Representation of a circle using bit-mapped graphics.



6.3 Displaying the Bit-Mapped Image

- Any **digital image** contains **two fundamental aspects**:
  - (1) the digital representation of the **image** stored **in computer memory**; and
  - (2) the image that the user actually sees, which is referred to as the ***rendering***.
- **Basic characteristics for bit-mapped graphics:**
  - **Numbers of pixels** in the horizontal and vertical directions, **or** equivalently, **size and spatial resolution**;
  - **Presence or absence of color information**;
  - **Resolution in the intensity domain** (number of gray levels,number of bits); and
  - **Resolution in the color domain** (expressed in RGB, HLS, or other coordinates).

6.3 Displaying the Bit-Mapped Image (2)

- **Some additional parameters (in bit-mapped image):**
  - **Multiple views of the scene for stereo vision** or alternative viewpoints, or a complete 3D data map from which any view or set of views may be generated;
  - **Multiple-resolution information** so that image resolution may be matched to display resolution; and
  - **Additional information** pertaining to each pixel, such as type of entity represented (**object, foreground, background, etc**.).
- **Fundamental limitation in rendering the image** is imposed by the display device:
  - **a CRT display**,
  - **LCD display,**
  - **projection system,**
  - **laser printer,**
  - **and so forth.**

6.4 Display Device Formats

6.4.1 Visual Display Devices

- **VGA (Video Graphics Array) standard**
  - a resolution of **640 x 400pixels** (horizontal and vertical resolutions), with only **16 possible color** values for each pixel.
- **super VGA(SVGA)**
  - a resolution of **640 x 480** and **256 colors**
- **Computer displays** are of two fundamental types: CRTs and LCDs.
  - **CRT** stands for cathode ray tube → basis of **television** since its origin more than 50 years ago
  - **LCD** stands for liquid crystal display → controlling the pattern of transparent and opaque areas in the display
  - CRT displays must be continually ***refreshed***, or recreated
  - super VGA standard → all **307,200 pixels** must be refreshed
  - a refresh rate of **at least 50 times per second** to satisfy the human eye.→ 1 byte per pixel, or 15,360,000 bytes(256 colors)

6.4 Display Device Formats(2)

6.4.1 Visual Display Devices(2)

- Data storage, data retrieval, and monitor interface issues are usually performed by a dedicated piece of hardware called a ***video interface board*** **(or card).**
  - super VGA with a resolution of 640 x 480 and 256 colors → 307,200 bytes memory = 640 x 480 x 8-bits=640 x 480 Byte
  - Image board with a resolution of 1024 x 768 with 65,536 colors → 2 MB of memory (=1024 x 768 x 16-bits = 1024 x 768 x 2 B)

6.4 Display Device Formats(3)

6.4.2 Printers and Similar Output Devices

- The ***facsimile*** machine: **``Group 3'' standard**
  - minimizing telephone transmission time with the modems then available (2400 or 4800 bps)
  - The goal was **1 minute per page**(1 ppm)
  - **black and white only** (no gray scale), a vertical resolution of **3.85 lines per mm**, and a horizontal resolution of **1728 pixels per line**
  - digital encoding technique : reduce transmission time by **only transmitting data when black-white transitions occur**, and skipping over the large amount of white space on a typical page
- ***The dot matrix printer***.
  - tapping fine wires on a ribbon→ forming ink dots on the page
  - (a vertical array of seven) x (the height of the seven wires)
  - The resolution = a standard VGA display (without the color)

6.4 Display Device Formats(4)

6.4.2 Printers and Similar Output Devices(2)

- ***The laser printer.***
  - **300** dots per inch (dpi)
  - **600 dpi** in most current laser printers → This resolution is fine enough to make lines that appear nearly perfectly continuous to the eye
  - **a gray level** can be generated by varying the density of black dots from near zero (white) to all present (black).
- ***Color printers***.
  - three ribbon colors
  - three colors of ink at each location to form color images

6.5 From Numbers to Images

- **VGA standard formats** allow **256 different colors** (1 byte per pixel), the video display has three inputs to determine the color of each pixel (red, green, and blue intensities, each of which can have at least 256 levels).
- How should the 256 available codes be mapped into the 16,777,216 (=256 x 256 x 256) available display colors to provide a reasonable representation of the bit-mapped image?
  - two quantities are mapped together via **look-up table with 256 3-byte entries**
  - The look-up table is often called a ***palette*** (or a color map)
  - This overall system is referred to as ***indexed color*** or ***pseudo-color***
  - reduction of the ultimate set of over **16 million** video display colors into the **256**

6.5 From Numbers to Images(2)

- For example, 256 color VGA format
- The default color palette is a look-up table that maps the 256 VGA colors into one of a subset of the 16,777,216 video display colors
- The look-up table lists index values (1 through 256) in the left column. The next four columns list the corresponding red, green, and blue intensity values for the given index I, and for the three following indices (I + 1), (I + 2), and (I + 3).

DEFAULT VGA COLOR MAP

Index I	I + 0	I + 1	I + 2	I + 3
	R G B	R G B	R G B	R G B
0	0 0 0	0 0 42	0 42 0	0 42 42
4	42 0 0	42 0 42	42 21 0	42 42 42
8	21 21 21	21 21 63	21 63 21	21 63 63
12	63 21 21	63 21 63	63 63 21	63 63 63
16	0 0 0	5 5 5	8 8 8	11 11 11
20	14 14 14	17 17 17	20 20 20	24 24 24
24	28 28 28	32 32 32	36 36 36	40 40 40
28	45 45 45	50 50 50	56 56 56	63 63 63
32	0 0 63	16 0 63	31 0 63	47 0 63
36	63 0 63	63 0 47	63 0 31	63 0 16
40	63 0 0	63 16 0	63 31 0	63 47 0
44	63 63 0	47 63 0	31 63 0	16 63 0
...	...	...	...	...
252	0 0 0	0 0 0	0 0 0	0 0 0

6.5 From Numbers to Images(3)

- Clearly, 256 color values represent quite a limited set, compared to the millions of possible video display colors.
- This limitation is partially overcome by allowing the palette to be changed to whatever is most appropriate for displaying the image at hand.
- (1) even choosing the best 256 colors for a given image may not result in a high-quality rendering of the original image, and
- (2) the palette must be the same for the entire computer display (that is, different images indifferent windows *cannot* have different palettes).

6.6 Virtual Reality Modeling Language

- ***Virtual reality*** :
  The concept of displaying (rendering) images from their stored format is quite straightforward. This concept may be extended to the rendering of images not from stored image data but from a stored model of the world that will produce the image.
- In other words, we don't just take a picture of something in the world and then display it. We first synthetically create a virtual world, and then take virtual pictures of that!
- **The Virtual Reality Modeling Language** or **VRML** (pronounced vermeil by those in the business) is a scene description language for creating navigable virtual 3D ``worlds." This language was created specifically for use with the Internet, but has proven an excellent vehicle for all applications, including those that simply run on one computer and do not involve any Internet-based movement of the data.

6.6 Virtual Reality Modeling Language(2)

- The word *language*, appearing in the acronym VRML, may lead onto believe that it is a programming language such as C or C++; however, it is not.
- Unlike C and C++ programs, VRML descriptions are not turned into machine code for execution on a computer but are *interpreted* by a **VRML viewer** or **VRML-enabled Web browser**.

6.6.1 A Comparison of VRML and HTML

- In **HTML** the tag pairs <B></B> and <I></I> are used to specify the **appearance of text** in the page; in this case <B></B>designates text that should appear in bold type while <I></I>designates text that should appear in italic type.
- **VRML** provides **building blocks** for describing objects in virtual worlds. VRML's capabilities include the ability to specify object geometry and location,object surface material and/or color, and object behavior.

6.7 The Organization of a VRML Scene

- A collection of descriptions of objects that make up such a world is typically called ***VRML scene***, and the computer file into which it is placed will be called a ***source*** **file**, as this is the only source of info the computer will be using to form the renderings that we will view.

6.7.1 In VRML Everything Is a Node

- The fundamental unit of organization in VRML is called a **node**
- think of a node as an **empty box with a blank tag** on the outside
- To define a node that holds **the geometry of a cube**, **the width, length and height** would be put in box and the word ``**Cube**'' would be placed on the tag.
- The same method could also be used to define a node for **a cone, a sphere, or any other object** that might come to mind.

6.7 The Organization of a VRML Scene (2)

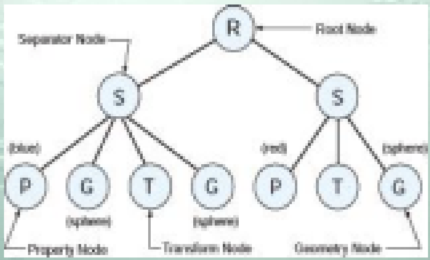
6.7.1 In VRML Everything Is a Node(2)

- The``Cube'' node as it would appear in a VRML source file.
  - ➔ Cube ***{ width 3 height 3 depth 3 }***
  - ➔ a cube that is three meters in length on all sides.
- In general, all **nodes** fall into one of **three broad categories**:
  - 1. ***Shape Nodes***--These nodes hold information about the geometry of an object. In VRML, geometries can be as simple as a box or as complicated as an automobile.
- 2. ***Property Nodes***--Property nodes hold information that can be used to modify geometry. The color of an object, the material it is made of, and the texture that should be applied to its surface are all examples of information that can be stored in a property node.
- 3. ***Group Nodes***--Group nodes allow many nodes to be grouped together and treated as one. For example, the body of a car is composed of a hood, doors, a windshield, and a trunk. In VRML, each of these pieces can be created as separate geometry nodes and assembled using a group node to form the body of the car.

6.7 The Organization of a VRML Scene (3)

6.7.2 The VRML Scene Graph

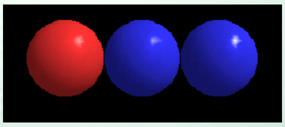
- **Nodes in scenes** are arranged in a hierarchical structure called a ***scene graph***
- **Figure 6.3:** The general structure of a VRML scene graph.



6.7 The Organization of a VRML Scene (4)

6.7.2 The VRML Scene Graph(2)

- **Figure 6.4:** An image rendered from the VRML file describing three balls.



- **#VRML V1.0 ascii**

```
Separator {
   Material { diffuse color 0 0 1 # Blue sphere }
   Sphere { radius 1 }
   Translation { translation 2 0 0 }
   Sphere { radius 1 } }
Separator {
   Material { diffuse color 1 0 0 # Red sphere }
   Translation { translation -2 0 0 }
   Sphere { radius 1 } }
```


6.8 Placing a Surface on a Virtual Object

- VRML supports ***texture mapping*** → can ***paste*** any picture onto the virtual surface of an object

- **#VRML V1.0 ascii**

```
separator {
    Texture 2
 { filename "my_image.jp
 g" }
    Sphere { radius 1 }}
```

- Figure 6.5: Four images of natural material are shown; from the upper left and clockwise: brick, a wire reinforced glass window material, bark mulch and ivy.

6.8 Placing a Surface on a Virtual Object(2)

Figure 6.6: Two balls onto which brick and glass images have been textured mapped.

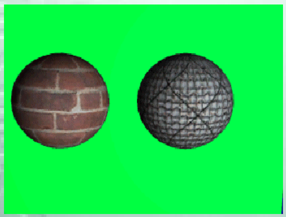
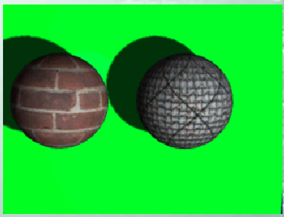


Figure 6.7: Shadowing has been added to the rendering of the two balls.



6.8 Placing a Surface on a Virtual Object(3)

- **Figure 6.8:**Several objects, onto each of which one of the previously shown textures has been mapped.