Characters, Partitioning, Display Lists, and Window Reshaping

CS116A

Chris Pollett

Sep15, 2004.

Introduction

- Character Primitives
- OpenGL Character functions
- Picture Partitioning
- OpenGL Display Lists
- OpenGL Display-Window Reshape Function

Character Primitives

In a graphics system often still want to be able to display text: For example, you might want to label graphs, have signs on 3D buildings, display high score, etc.

The upshot is: Most graphics API's support some way to output text.

So one might ask what ways are there for storing outputting text in a graphical setting?

Characters in Graphics APIs

There are a number of parameters used to specify letters for rendering:

- typeface, which specifies the overall design of the letters. Ex: Courier, Helvetica, New York, etc.
- font: specifies particular form of character: 12 point Courier Italic. 1pt = 1/72 inches

Types of typefaces; Types of Fonts

- Two broad groups of typeface: *serif* and *sans-serif*. Ex: Palatino is a serif font; whereas, Gill sans Light is sans serif. Notice the latter does not have extra flourishes at the end of strokes.
- Former is easier to read lots of. The latter is easier to recognize single characters
- Fonts can also be classified according to whether they are proportional spaced (kerned letters) or monospaced.

Computer Representation

Two main types: bitmap (raster) font or outline (stroke or vector) font.

- As name implies bitmap fonts store a bitmap for each symbol. Need a new bitmap if change font-size or change to italics or bold. Everything needs to be multiples of pixel size.
- Outline fonts specify characters by giving values for various relative positions of points on the outline of a character.

OpenGL Character functions

- For bitmapped fonts can use:

```
glRasterPosition2i(x,y);
font = GLUT_BITMAP_9_BY_15;
character=‘b’;
glutBitmapCharacter(font, character);
glutBitmapCharacter(font, ‘o’);
```

- For Stroke Characters you can use:

```
font=GLUT_STROKE_MONO_ROMAN;
glutStrokeCharacter(font,’b’);
```

Picture Partitioning

- Some graphics libraries support the ability to create named sections of pictures and then allow one to move, remove, edit delete this named section.
- Example: could have a named section for coffee cups and could add many copies of such to world.
- Commonly used names for these partitions are: *structures, segment, or objects*.

OpenGL Display Lists

- OpenGL supports a way to store named sequences of instruction called a **display list**:

```
glNewList (listID, listMode);
…
glEndlist();
```

- listID is a positive integer ID for the list.
- listMode can be either GL_COMPILE or GL_COMPILE_AND_EXECUTE.
- To be sure to generate distinct IDs can use:

```
listID1= glGenLists(2);
listID2 = listID1+1;
```

More Display Lists

- To check if an ID is in use can use glIsList(listID);
- Here is a short example of creating and using a list:

```
GLuint myList;
myList = glGenLists(1);
glNewList(myList, GL_COMPILE);
glBegin(GL_TRIANGLES);
    glVertex2i(0,0);
    glVertex2i(10,100);
    glVertex2i(100, 10);
glEnd(); glEndList();
glCallList(myList); // draw it
```

Yet More Lists

Can draw multiple lists using: glListBase(offset) and glCallLists(nLists, arrayDataType, listIDArray)

Can delete lists with glDeleteLists(startID, nLists);

OpenGL Display-Window Reshape Function

- Window to our OpenGL applications often gets moved or resized messing up our drawings.
- To fix this can specify a reshape function using:

```
glutReshapeFunc(myReshape);
```

where myReshape has prototype:

```
void myReshape(int width, int height);
```