Two Dimensional Viewing Functions and Clipping

CS116A

Chris Pollett

Nov. 10, 2004.

Outline

- OpenGL 2D Viewing
- Types of Clipping Algorithms
- 2D point clipping

OpenGL 2D Viewing

- OpenGL designed for 3D.
- Some 3D viewing routines can be adapted.
- Core library does support a viewport function
- GLU provides 2D clipping function
- GLUT allows one to manipulate display windows

OpenGL Projection Mode

- OpenGL does not support a separate viewing coordinate system
- Set clipping window as part of projection transformation.
- To specify projection transformation, first get into the correct matrix mode:

  glMatrixMode(GL_PROJECTION);

  For good measure can also set identity matrix:

  glLoadIdentity();

GLU Clipping Window

- To specify the clipping window do:
  gluOrtho2D(xwmin, xwmax, ywmin, ywmax);
- Coordinates are doubles.
- For 3D the effect of the above is to project out the z-axis.
- Has no effect on 2D scenes except to map to normalized coordinates.
- For OpenGL these have range between -1,1.

OpenGL Viewport Function

- To specify the viewport use:

  `glViewport(xvmin, yvmin, vpWidth, vpHeight);`
- xvmin, yvmin specify the position of the lower left corner of the viewport relative to the bottom of the display window.
- vpWidth, vpHeight give width and height.
- To get the info about the currently active viewport can use:

  `glGetIntegerv(GL_VIEWPORT, vpArray);`
- vpArray is a four element array.

GLUT Display Windows

- As we have already been using, GLUT is used to manipulate display windows for the programs we have been writing.
- To start up GLUT:

  ```
  glutInit(&argc, argv);
  ```
- Then to set up the display window:

  ```
  glutInitWindowPosition(xTopLeft, yTopLeft);
  glutInitWindowSize(dwWidth, dwHeight);
  glutCreateWindow("Title");
  ```
- The windowing system can choose to ignore info passed by GLUT
- At this point window is not yet displayed.

Display Mode and Color

- Next we set the display window parameters with:

  glutInitDisplayMode(mode);

  We'have already seen case where mode was
  GLUT_SINGLE | GLUT_RGB

- To set the background color:

  glClearColor(red, green, blue, alpha);

  or in index mode:

  glClearIndex(index);

Window IDs

- When create a window can obtain its ID:
  `windowID = glutCreateWindow("my window");`
- IDs start at 1.
- Can use ID to get rid of a display window:
  `glutDestroyWindow(windowID);`
- Can use ID to set active display
  `glutSetWindow(windowID);`
- To get active window:
  `currentWindowID = glutGetWindow();`

Repositioning and Resizing the Window

- To move the active window to a new position:
  ```
  glutPositionWindow(x, y);
  ```
- To change its size:
  ```
  glutReshapeWindow(width, height);
  ```
- To make the window fullscreen:
  ```
  glutFullScreen();
  ```
- To set a callback which will be called whenever the position or size of display is changed:
  ```
  glutReshapeFunc(myfunc);
  ```

Managing Multiple Displays

- Glut has many function for manipulating the window. For example:

```
glutIconifyWindow(); //shrink window to an icon
glutSetWindowTitle(“new title”);

// make window front window
glutSetWindow(windowID);
glutPopWindow();

//make window back window
glutSetWindow(windowID);
glutPushWindow();

glutHideWindow() //take window offscreen
glutShowWindow() // put onscreen / de-iconify
```

Subwindows

- One can create a subwindow of a display with:

  subwindowID = glutCreateSubWindow(windowID, xBottomLeft, yBottomLeft, width, height);
- Subwindow IDs can be used much like usual IDs but effect will be within the window the subwindow belongs to.
- One cannot iconify subwindows

Yet More Glut

- The shape of the cursor can be set with:

```
glutSetCursor(shape); /*  example shape's:
    GLUT_CURSOR_UP_DOWN,
    GLUT_CURSOR_CYCLE, GLUT_CURSOR_WAIT,
    GLUT_CURSOR_DESTROY */
```

- To set the display callback use:

```
glutDisplayFunc(myDisplayFunc);
```

- To force GLUT to call your function:

```
glutPostRedisplay();
```

- Finally, to get windows displayed and start the event loop call:

```
glutMainLoop();
```

GLUT…

- Sometimes it is useful to have a callback that is called even if there are no events to be processed.
- To set such a function in GLUT:

  glutIdleFunc(myIdleFunc);

- To query state parameters of GLUT:

  glutGet(param);

  Examples: GLUT_WINDOW_X, GLUT_WINDOW_WIDTH

Clipping Algorithms

- A **clipping algorithm** is a procedure for eliminating a portion of a picture outside of a specified region.
- Such an algorithm is most often used as final portion of viewing pipeline before image displayed to device
- There are many techniques for clipping depending on what kind of object is being clipped:
  - Point Clipping
  - Line Clipping
  - Fill-area Clipping
  - Curve Clipping
  - Text Clipping

2D Point Clipping Algorithm

- Suppose clipping region is given by xwmin, xwmax, ywmin, ywmax.
- Given a point (x,y) check if the inequalities:

  xwmin <= x <= xwmax

  and

  ywmin <= y <= ywmax hold.

  If not clip.

- This algorithm can be applied to scenes with particle systems like clouds lor smoke.