OpenGL Examples

- From Nate Robins:

- Tutorials at:
GLUI

• GLUI is a GLUT-based C++ user interface library which provides controls such as
  – buttons,
  – checkboxes,
  – radio buttons,
  – spinners, etc.

• It is window-system independent, relying on GLUT to handle all system-dependent issues, such as window and mouse management.
GLUI Controls
Standalone GLUI windows
GLUI subwindows

Hello World!
GLUI - Simple Programming Interface

- GLUI provides default values for many parameters in the API and there are several ways to create a control:

```c
GLUI *glui;
...

glui->add_checkbox("Click me"); // Adds a simple checkbox with the name "Click me"

glui->add_checkbox("Click me", &state ); // The variable state will now be automatically update to reflect the state of the checkbox (live variable).

glui->add_checkbox( "Click me", &state, 17, callback_fn ); // Now we have a live variable, plus a callback function will be invoked (and passed the value '17') whenever the checkbox changes state.
```
Usage for standalone GLUI windows

Integrating GLUI with a new or existing GLUT application is very straightforward. The steps are:

1. Add the GLUI library to the link line (e.g., glui32.lib for Windows -lglui in Linux).
2. Include the file "glui.h" in all sources that will use the GLUI library.
3. Create your regular GLUT windows as usual. Make sure to store the window id of your main graphics window, so GLUI windows can later send it redisplay events:

   ```c
   int window_id = glutCreateWindow( "Main gfx window" );
   ```

4. Register your GLUT callbacks as usual (except the Idle callback, discussed below).
5. Register your GLUT idle callback (if any) with GLUI_Master (a global object which is already declared), to enable GLUI windows to take advantage of idle events without interfering with your application's idle events. If you do not have an idle callback, pass in NULL.

   GLUI_Master.set_glutIdleFunc( myGlutIdle );

   or

   GLUI_Master.set_glutIdleFunc( NULL );

6. In your idle callback, explicitly set the current GLUT window before rendering or posting a redisplay event. Otherwise the redisplay may accidently be sent to a GLUI window.

   void myGlutIdle( void )
   {
       glutSetWindow(main_window);
       glutPostRedisplay();
   }
Usage for standalone GLUI windows

7. Create a new GLUI window using

    GLUI *glui = GLUI_Master.create_glui( "name", flags, x, y );

    Note that flags, x, and y are optional arguments. If they are not specified, default values will be used. GLUI provides default values for arguments whenever possible.

8. Add controls to the GLUI window. For example, we can add a checkbox and a quit button with:

    glui->add_checkbox( "Lighting", &lighting );
    glui->add_button( "Quit", QUIT_ID, callback_func );
Usage for standalone GLUI windows

9. Let each GLUI window you've created know where its main graphics window is:
   \[ \text{glui->set_main_gfx_window( window_id );} \]

10. Invoke the standard GLUT main event loop, just as in any GLUT application:
    \[ \text{glutMainLoop();} \]

• Refer GLUI Manual on the usage for GLUI subwindows
GLUI API Example

4.2.5 Buttons

Buttons are used in conjunction with callbacks to trigger events within an application.

![Button](image)

**Button nested within panel**

**add_button, add_button_to_panel**

Adds a new button to a GLUI window, optionally nested within a panel (or rollout).

**Usage**

```c
GLUI_Button *GLUI: add_button( char *name, int id=-1,
                                 GLUI_Update_CB callback=NULL);
```

```c
GLUI_Button *GLUI: add_button_to_panel( GLUI_Panel *panel,
                                  char *name, int id=-1,
                                  GLUI_Update_CB callback=NULL);
```

- **name** - Name of button
- **id** - If callback is defined, it will be passed this integer value
- **callback** - Pointer to callback function (taking single int argument) to be called when the button is pressed
- **panel** - Existing panel (or rollout) to nest button in

**Returns:** Pointer to a new button control
Details about GLUT Functions

• In the next few slides we will concentrate on details of some GLUT functions.
• You can access more details from GLUT API.
GLUT Functions - 1

`glutInit(int argc, char *argv)`
initializes GLUT, and processes any command line arguments

`glutInitWindowPosition(int x, int y)`
specifies the location of the created window with respect to the screen

`glutInitWindowSize(int width, int height)`
specified the size of the created window in terms of pixels
glutInitDisplayMode(unsigned int mode)
sets the display mode, determined by bitwise OR’ing of
GLUT display mode bit masks:
− GLUT_RGBA: RGBA mode window. This is the default if
  neither GLUT_RGBA nor GLUT_INDEX are specified.
− GLUT_RGB: Same as GLUT_RGBA
− GLUT_INDEX: Color index mode window
− GLUT_SINGLE: Single buffered window
− GLUT_DOUBLE: Double-buffered window
− GLUT_ALPHA: Window width an alpha component to the
  color buffer(s).
− GLUT_DEPTH: Window with a depth buffer
GLUT Functions - 3

**glutCreateWindow(char *string)**

Creates a window with OpenGL context. It returns a unique identifier for the new window.

**glutMainLoop(void)**

Enters the GLUT event-Processing loop. This routine should be called at most once in a GLUT program. Once called, this routine will never return. It will call as necessary any callbacks that have been registered.
GLUT Functions - 4

**glutDisplayFunc(void (*func)(void))**
Whenever GLUT determines that the contents of the window should be redisplayed, the callback function specified is executed.

**glutPostRedisplay(void)**
 Raises an event of “the content of the window should be redisplayed”.

**glutSwapBuffers(void)**
 Performs a buffer swap on the layer in use for the current window.
GLUT Functions - 5

`glutReshapeFunc(void (*func)(int w, int h))`
Whenever GLUT determines that the size of the window is changed, the callback function specified is executed.

`glutKeyboardFunc(void (*func)(int key, int x, int y))`
Sets the keyboard callback for the current window. `glutGetModifiers` can be called to get ALT,CTRL, etc.

`glutSpecialFunc(void (*func)(int key, int x, int y))`
Sets the special keyboard callback for the current window. `glutGetModifiers` can be called to get ALT,CTRL, etc.
## Special Key Constants

- GLUT_KEY_F1
- GLUT_KEY_F2
- GLUT_KEY_F3
- GLUT_KEY_F4
- GLUT_KEY_F5
- GLUT_KEY_F6
- GLUT_KEY_F7
- GLUT_KEY_F8
- GLUT_KEY_F9
- GLUT_KEY_F10
- GLUT_KEY_F11
- GLUT_KEY_F12
- GLUT_KEY_LEFT
- GLUT_KEY_UP
- GLUT_KEY_RIGHT
- GLUT_KEY_DOWN
- GLUT_KEY_PAGE
- GLUT_KEY_PAGE
- GLUT_KEY_HOME
- GLUT_KEY_END
- GLUT_KEY_INSERT
GLUT Functions - 6

`glutMouseFunc(void (*func)(int button, int state, int x, int y))`
Whenever GLUT determines that a *mouse button is pressed or released*, the callback function specified is executed.
- button may be one of
  - GLUT_LEFT_BUTTON,
  - GLUT_MIDDLE_BUTTON,
  - GLUT_RIGHT_BUTTON
- State may be one of
  - GLUT_UP
  - GLUT_DOWN

`glutMotionFunc(void (*func)(int x, int y))`
Whenever GLUT determines that *mouse is moved with one or more buttons pressed*, the callback function specified is executed.

`glutPassiveMotionFunc(void (*func)(int x, int y))`
Whenever GLUT determines that *mouse is moved with no buttons pressed*, the callback function specified is executed.
GLUT Functions - 7

glutIdleFunc(void (*func)(void))
    Registers a function that’s to be executed if no other events are pending.

glutWireCube(Gldouble size)
    Draws a wireframe cube with the specified size

glutSolidSphere(Gldouble radius, Glint slices, Glint stacks)
    Draws a solid sphere with the specified parameters
GLUT Functions - 8

- Some other GLUT functions:
  
glutReshapeWindow, glutFullScreen, glutPopWindow, glutPushWindow, glutShowWindow, glutHideWindow, glutIconifyWindow, glutSetCursor, glutCreateMenu, glutSetMenu, glutGetMenu, glutDestroyMenu, glutAddMenuEntry, glutAddSubMenu, glutAttachMenu, glutDetachMenu, glutVisibilityFunc, glutEntryFunc
References

- Mainly from the resources of Ceng477 previous semesters, their references also valid for this document

- New references:
  - www.cse.unr.edu/~bebis/CS480/Notes/opengl_part1.ps.gz