Looking Glass Quilts for a Web Page - Embedded 3D Display, Using both OpenGL and Blender

Mike Bailey
mjb@cs.oregonstate.edu

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License

Here is How to Generate a Quilt from OpenGL

Here are some pre-defined constants you will need:

```c
const int QUILTWIDTH = 3360;
const int QUILTHEIGHT = 3360;
const int QUILTTOTALBLOCKS = QUILTNUMWIDTH * QUILTNUMHEIGHT;

const int BLOCKWIDTH = QUILTWIDTH / QUILTNUMWIDTH;
// 420

const int BLOCKHEIGHT = QUILTHEIGHT / QUILTNUMHEIGHT;
// 560

const float QUILTZ0P = 5.f;
const float QUILTDELTAEYE = 0.10f;

const float QUILT_ASPECT_Y_OVER_X = (float)BLOCKHEIGHT / (float)BLOCKWIDTH;
// 4:3 = 1.3333

const char * QUILTFILENAME = "Quilt_qs8x6a0.75.bmp";
```

```c
unsigned char QuiltArray[QUILTHEIGHT][QUILTWIDTH][3];
// holds the entire quilt

unsigned char BlockArray[BLOCKHEIGHT][BLOCKWIDTH][3];
// holds one block
```

Here are some pre-defined constants you will need:

```c
void Keyboard ( unsigned char c, int x, int y )
{
switch ( c )
{
    case 'w':
    case 'W':
        CreateAndWriteQuilt();
        break;
    . . .
    case 'q':
    case 'Q':
    case ESCAPE:
        DoMainMenu(QUIT);     // will not return here
        break;                          // happy compiler
    default:
        fprintf( stderr, "Don't know what to do with keyboard hit: '%c' (0x%0x)
        
```

```c
    SWITCH_TO_DISPLAY( )
    { gildrawwindow( MainWindow );
      ClearScreen( QUILTFILENAME );
      glutPBSave( QUILTFILENAME, QUILTTOTALBLOCKS/2, 0 );
      . . .
```

```c
    case ESCAPE:
    case QUIT:
        // will not return here
        break; // happy compiler
    default:
        fprintf( stderr, "Don't know what to do with keyboard hit: '%c' (0x%0x)
```
Generating the 48 Individual Images for the Quilt, II

// done rendering -- upload the pixels:
glReadBuffer(GL_FRONT);
glPixelStorei(GL_PACK_ALIGNMENT, 1);
for( int yb = 0; yb < BLOCKHEIGHT; yb++)
  for( int xb = 0; xb < BLOCKWIDTH; xb++)
    QuiltArray[y*BLOCKHEIGHT+yb][x*BLOCKWIDTH+xb] = BlockArray[yb][xb];
Sleep(200);
eyex += QUILTDELTAEYE;
WriteArray( QUILTFILENAME );

DisplayScene( ) – just do the scene drawing, I

DisplayScene() {
  // rotate the scene:
  glRotatef((GLfloat)Yrot, 0.f, 1.f, 0.f);
  glRotatef((GLfloat)Xrot, 1.f, 0.f, 0.f);
  // uniformly scale the scene:
  if (Scale < MINSCALE) Scale = MINSCALE;
  glScalef((GLfloat)Scale, (GLfloat)Scale, (GLfloat)Scale);
  // possibly draw the axes:
  if (AxesOn != 0)
    glColor3fv(&Colors[WhichColor][0]);
    glCallList(AxesList);
  // since we are using glScalef( ), be sure the normals get unitized:
  glEnable(GL_NORMALIZE);
  . . .
  if (SmoothOn) {
    glShadeModel(GL_SMOOTH);
  }
  if (LightingOn) {
    glEnable(GL_LIGHTING);
    glEnable(GL_LIGHT0);
    glTexEnvf(GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE, GL_MODULATE);
  }
  if (TextureOn) {
    glEnable(GL_TEXTURE_2D);
  }
  . . .
  DisplayScene( ) – just do the scene drawing, II

DisplayScene( ) – just do the scene drawing, II

DisplayScene( ) – just do the scene drawing, III

// draw the objects by calling up their display lists:
  glltranslate(0.f, -2.25f, 0.);
  glColor3f(0.8f, 0.2f, 0.2f);
  glmaterial(0.8f, 0.2f, 0.2f, 10.f);
  gllcalllist(SphereDL);
  gltranslate(0.f, 1.75f, 0.);
  glColor3f(0.8f, 0.8f, 0.2f);
  SetMaterial(0.8f, 0.8f, 0.2f, 8.f);
  gllcalllist(ConeDL);
  gltranslate(0.f, 2.5f, 0.);
  glColor3f(0.2f, 0.8f, 0.2f);
  SetMaterial(0.2f, 0.8f, 0.2f, 6.f);
  gllcalllist(TorusDL);
  glDisable(GL_LIGHTING);
  glDisable(GL_TEXTURE_2D);
  glShadeModel(GL_FLAT);
  . . .
void FrustumZ( float left, float right, float bottom, float top, float znear, float zfar, float zproj )
{
    if( zproj != 0.0 )
    {
        left   *= ( znear/zproj );
        right  *= ( znear/zproj );
        bottom *= ( znear/zproj );
        top    *= ( znear/zproj );
    }
    glFrustum( left, right, bottom, top, znear, zfar );
}

void OffAxisPersp( float fovxdeg, float aspect_y_over_x, float znear, float zfar, float z0p, float eyex )
{
    float tanfovx = tanf( (float)(M_PI/180.) * fovxdeg / 2.f );
    float right = z0p * tanfovx;
    float left  = -right;
    float bottom = aspect_y_over_x * left;
    float top    = aspect_y_over_x * right;
    left  -= eyex;
    right -= eyex;
    FrustumZ( left, right, bottom, top, znear, zfar, z0p );
    glTranslatef( -eyex, 0.0, 0.0 );
}

void Display( )
{
    glutSetWindow( MainWindow );
    glDrawBuffer( GL_BACK );
    glClear( GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT );
    glEnable( GL_DEPTH_TEST );
    glShadeModel( GL_FLAT );
    // set the viewport to a square centered in the window:
    GLsizei vx = glutGet( GLUT_WINDOW_WIDTH );
    GLsizei vy = glutGet( GLUT_WINDOW_HEIGHT );
    GLsizei v = vx < vy ? vx : vy;                  // minimum dimension
    GLint xl = ( vx - v ) / 2;
    GLint yb = ( vy - v ) / 2;
    glViewport( xl, yb,  v, v );
    // set the viewing volume:
    glMatrixMode( GL_PROJECTION );
    glLoadIdentity( );
    if( WhichProjection == ORTHO )
        glOrtho( -2.f, 2.f,     -2.f, 2.f,     0.1f, 1000.f );
    else
        gluPerspective( 70.f, 1.f,      0.1f, 1000.f );
    // place the objects into the scene:
    glMatrixMode( GL_MODELVIEW );
    glLoadIdentity();
    gluLookAt( 0.f, 0.f, 7.f,     0.f, 0.f, 0.f,     0.f, 1.f, 0.f );
    // draw just the scene:
    DisplayScene( );
    glutSwapBuffers( );
    glFlush( );
}

How the usual Display( ) function calls DisplayScene( )

How to Generate a Quilt from Blender

You can also generate these quilts automatically from Blender.

The first thing you need to do is retrieve a Blender Add-on.
The Add-on's name is **AliceLG.zip** and it can be found at [https://github.com/regcs/AliceLG/releases](https://github.com/regcs/AliceLG/releases)

Click here and save this file anywhere.
Do not un-zip it!
Run Blender.

Click on Edit→Preferences→Add-ons.

Click on Install. Then navigate to where you installed AliceLG.zip and select it. Don’t un-zip it!

Scroll down the list of Add-ons until you see View: Alice/LG

Click the checkbox to enable it. Then click the arrow and scroll down so you can see all the information.

Scroll down so you can see all the information.

Then click on Install Dependencies. This might take a while. Be patient.
When it’s done, you will see this:

Preferences:
All required Python modules were installed.
⚠️ Please restart Blender to activate the changes!

Now exit Blender and start it up again.

Getting the AliceLG.zip Blender Add-on, V

Finding the Looking Glass Tab in Blender

Hit the ‘n’ key and you will see the usual Transform panel, but something new has been added.

Click on the Looking Glass tab.

Using the Looking Glass Tab, I

Click here and select Camera
This will bring up a viewing volume, like this:

Using the Looking Glass Tab, II

Now adjust the Clip Start and Clip End to completely surround your scene.

Then adjust the Focal Plane to be roughly down the middle of the scene, like this:

Select Portrait, 48 Views
Go select the Output icon and navigate down to Output. Click here and navigate to some folder for Blender to render your quilt into. Click Accept and then give a filename. On my system, it ended up looking like this.

Going back to the Looking Glass tab, click on Render Quilt. This will trigger the rendering of 48 different scenes.

You can watch its progress at the bottom of the screen.

Navigate to https://blocks.glass

If you don’t have an account, sign up to get one. If you are under 18 years old, be sure to have your parents help you do this!

If you do have an account, Log in to it.
Click on the big Plus Sign

Drag-and-Drop your quilt file here, or click here and navigate

Click on Manage, then click on the name of the file you just uploaded

Wiggle the mouse left and right to see your hologram move in 3D

Click on Manage again then click here
That will bring up a bunch of things you can do with your new "hologram". Clicking here will copy the embedded HTML code, to get at this hologram from the web, to the clipboard. Paste that into one of your HTML web pages! Be sure this permission is set to Unlisted.

Clicking here will copy the embedded HTML code, to get at this hologram from the web, to the clipboard. Paste that into one of your HTML web pages! Be sure this permission is set to Unlisted.

https://lookingglassfactory.com/looking-glass-portrait