Display Lists

Mike Bailey
mjb@cs.oregonstate.edu

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

```c
void Sphere( float radius, int slices, int stacks )
{
    struct point top, bot;          // top, bottom points
    struct point p;
    NumLngs = slices;
    NumLats = stacks;
    Pts = new struct point[ NumLngs * NumLats ];
    for( int ilat = 0; ilat < NumLats; ilat++ )
    {
        float lat = -M_PI/2. + M_PI * (float)ilat / (float)(NumLats-1);
        float xz = cos( lat );
        float y   = sin( lat );
        for( int ilng = 0; ilng < NumLngs; ilng++ )
        {
            float lng = -M_PI + 2. * M_PI * (float)ilng / (float)(NumLngs-1);
            float x =  xz * cos( lng );
            float z = -xz * sin( lng );
            p = PtsPointer( ilat, ilng );
            p->x  = radius * x;
            p->y  = radius * y;
            p->z  = radius * z;
            p->nx = x;
            p->ny = y;
            p->nz = z;
            p->s = ( lng + M_PI    ) / ( 2.*M_PI );
            p->t = ( lat + M_PI/2. ) / M_PI;
        }
    }
}
```

**Drawing a Sphere**

Notice a lot of time-consuming Trig Function Calls!

```
GLfloat GL_QUADS();
for( int ilng = 0; ilng < NumLngs-1; ilng++ )
{
    p = PtsPointer( NumLats-1, ilng );
    DrawPoint( p );
    p = PtsPointer( NumLats-2, ilng );
    DrawPoint( p );
    p = PtsPointer( NumLats-2, ilng+1 );
    DrawPoint( p );
    p = PtsPointer( NumLats-1, ilng+1 );
    DrawPoint( p );
}
```

Even worse, the trig calls are inside single or nested for-loops!
You don’t want to execute all that code every time you want to redraw the scene, so draw it once, store the numbers in GPU memory, and call them back up later.

Without Display List:
The CPU re-computes and transmits the sphere coordinates every time it they need to be drawn.

With Display List:
The CPU computes and transmits the sphere coordinates once and then they are grabbed from GPU memory every time they need to be drawn.

A Common Display List Misconception

Let’s say that we are creating a rectangle in a Display List, like this:

```c
float L, W; // length and width global variables
int RectList; // rectangle display list
L = 10.; W = 5.;
gNewList( RectList, GL_COMPILE );
<< draw a rectangle using L and W >>
gEndList( );
```

Then, when we go to use the DL, we do this:

```c
L = 4.; W = 2.;
gCallList( RectList );
```

What size rectangle will it draw? 10x5? 4x2?

It will draw a 10x5 rectangle. Display Lists bake in the numbers. They retain no knowledge of what variables were used to create those numbers!