CS 418: Interactive Computer Graphics

Basic Animation

Eric Shaffer

Simple Animation with WebGL

- Animation means we:
  - Draw some things
  - Move the geometry slightly
  - Draw the things again
  - Repeat….
- Things we will use:
  - glMatrix library for doing affine transformations of geometry http://glmatrix.net/
  - Code in a file called webgl-utils.js for some common tasks

Transforming Geometry in the Vertex Shader

```
<script id="shader-vs" type="x-shader/x-vertex">
  attribute vec3 aVertexPosition;
  attribute vec4 aVertexColor;
  uniform mat4 uMVMatrix;

  varying vec4 vColor;

  void main(void) {
      gl_Position =uMVMatrix*vec4(aVertexPosition, 1.0);
      vColor = aVertexColor;
  }
</script>
```

We can use the 3D transformations we've learned to alter geometry in the the vertex shader.

A Uniform variable is one that is the same for all the vertices being processed by the shader.

In the code at the left, we use Uniform 4x4 matrix to transform the coordinates of each vertex.

Getting the Transformation to the Shader

```
<script src="gl-matrix-min.js"></script>
```

```
var mvMatrix = mat4.create();

function setupShaders() {
…
shaderProgram.mvMatrixUniform =
                    gl.getUniformLocation(shaderProgram, "uMVMatrix");
}


function draw() {
…
  mat4.identity(mvMatrix);
  mat4.rotateX(mvMatrix, mvMatrix, degToRad(rotAngle));
  gl.uniformMatrix4fv(shaderProgram.mvMatrixUniform, false, mvMatrix);
  gl.drawArrays(gl.TRIANGLES, 0, vertexPositionBuffer.numberOfItems);
}
```

We will use the glMatrix library for vector and matrix math inside JavaScript code.

There are several steps to using the library and to create a matrix to be sent to the shader.

1. Include the library in the HTML file using <script> tags
2. We create a matrix
3. When we initialize the shader, we get a handle for the Uniform matrix in the shader
4. In the draw() function, we create a transformation and use the handle to send it to the shader

Animation

- So, we now need a way to
  - Draw multiple frames
  - Change the transformation each frame so the geometry appears to move

requestAnimationFrame

```
function startup() {
  canvas = document.getElementById("myGLCanvas");
  gl = createGLContext(canvas);
  setupShaders();
  setupBuffers();
  gl.clearColor(0.0, 0.0, 0.0, 1.0);
  gl.enable(gl.DEPTH_TEST);
  tick();
}

function tick() {
    requestAnimFrame(tick);
    draw();
    animate();
}
```

requestAnimFrame

tells the broswer you want to animate and gives it a function to call before the the next repaint.

The number of callbacks is usually 60 times per second, but will generally match the display refresh rate.

Animation

```
function animate() {
    var timeNow = new Date().getTime();
    if (lastTime != 0) {
        var elapsed = timeNow - lastTime;
        rotAngle= (rotAngle+1.0) % 360;
    }
    lastTime = timeNow;
}
```

With each frame, we update a global variable that is used to set the rotation matrix sent to the shader.

The time information isn't used here, but you may find it useful for your work.

Try it out

http://courses.engr.illinois.edu/cs418/index.html

Look for :
HelloAnimation.html
HelloAnimation.js
webgl-utils.js

and grab the latest verion of glMatrix:
http://glmatrix.net/