Animation Effects using a Timer

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

Mike Bailey
mjb@cs.oregonstate.edu

Using Timers with Shaders

glman has a built-in Timer variable. You just need to declare it:

```
uniform float Timer;
```

Then, just use it in your code. It goes from 0. to 1. in 10 seconds, and then instantly back to 0.

Or, you can program a Timer yourself:

```c
float Timer;
const int MS_PER_CYCLE = 10*1000; // 10,000 ms = 10 seconds

void Animate() {
  int ms = glutGet(GLUT_ELAPSED_TIME);
  ms %= MS_PER_CYCLE;
  Timer = (float)ms / (float)MS_PER_CYCLE; // 0. to 1. in 10 seconds
  glutPostWindow(MainWindow);
}

void InitGraphics() {
  glutIdleFunc(Animate);
}
```

Fun With Zero-to-One:
There are many ways to map 0.→1. to a different function

<table>
<thead>
<tr>
<th>Function Type</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single ramp 0.→1.</td>
<td><code>float t = Timer;</code></td>
</tr>
<tr>
<td></td>
<td><code>float t = Timer*Timer;</code></td>
</tr>
<tr>
<td></td>
<td><code>float t = Timer*Timer*Timer;</code></td>
</tr>
<tr>
<td></td>
<td><code>float t = 10.*Timer^3 - 15.*Timer^4 + 6.*Timer^5</code></td>
</tr>
<tr>
<td>Double ramp 0.→1. →0.</td>
<td><code>float t;</code></td>
</tr>
<tr>
<td></td>
<td>if (Timer &lt;= .5 )</td>
</tr>
<tr>
<td></td>
<td><code>t = 2.*Timer;</code></td>
</tr>
<tr>
<td></td>
<td>else</td>
</tr>
<tr>
<td></td>
<td><code>t = 2. * ( 1. - Timer );</code></td>
</tr>
<tr>
<td>Smooth oscillation -1.→1.</td>
<td><code>float t = sin( 2.*π*Timer );</code></td>
</tr>
<tr>
<td>Smooth oscillation 0.→1.→-1.</td>
<td><code>float t = sin( 2.*π*Timer );</code></td>
</tr>
<tr>
<td>Faster oscillation</td>
<td><code>float t = sin( 2.*π*Timer );</code></td>
</tr>
<tr>
<td>Bigger oscillation</td>
<td><code>float t = Mag * sin( 2.*π*S*Timer );</code></td>
</tr>
</tbody>
</table>

```c
float t = sin( 2.*π*S*Timer );
```

```
void InitGraphics() {
  ..
  glutIdleFunc(Animate);
}
```
Sidebar: Why Do These Two Curves Match So Closely?

The Taylor Series expansion of \( y = \sin\left(\frac{\pi}{2}x\right) \) around \( x=0.5 \) is:

\[
y = \frac{1}{2} - \frac{x^2}{4} + \frac{x^4}{32} - \frac{x^6}{192} + \cdots
\]

which is somewhat close to: \( y = 3x^2 - 2x^4 \)