λ Lounge

OpenGL in Common Lisp

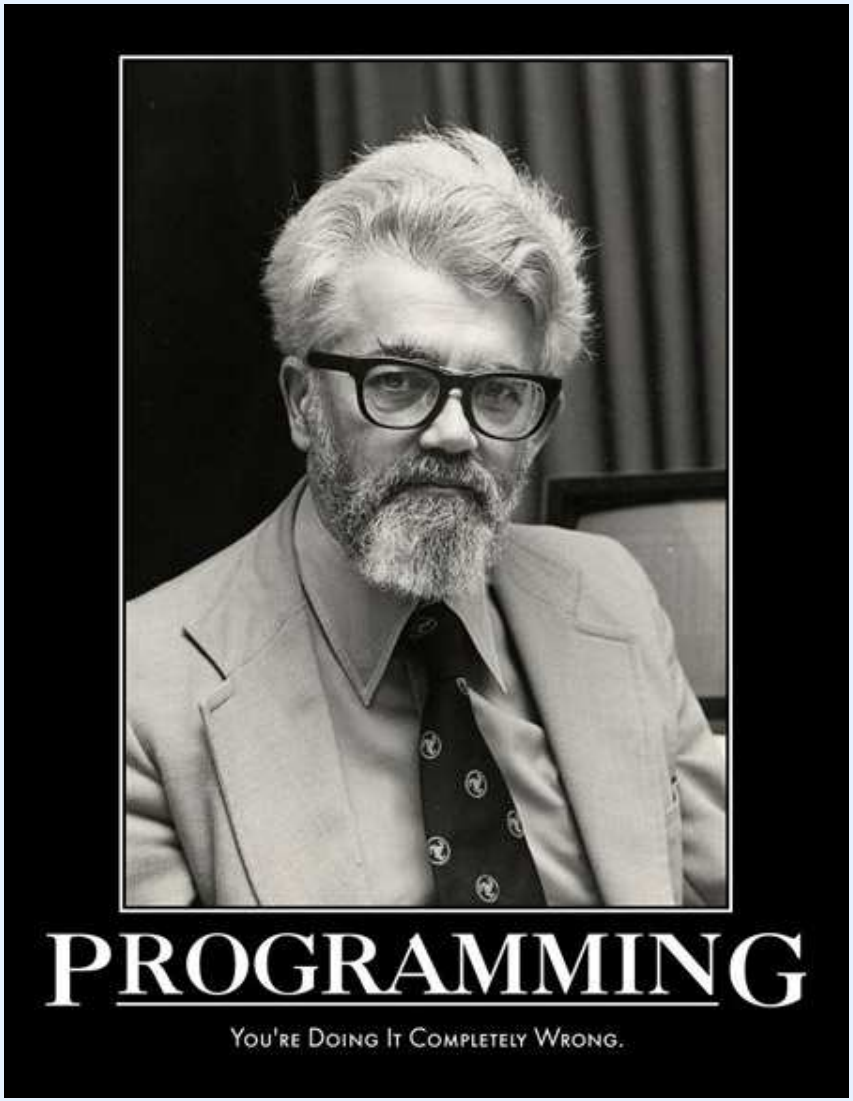
Christopher Mark Gore

http://www.cgore.com

cgore@cgore.com

Thursday, November 7, AD 2013

Lisp is Cool!



3D is Cool!

Except for 3D Pitfall, which looks really lame.

Getting Started

1. Install Linux.

   ```
   http://aptosid.com
   ```

2. Install SBCL and some libraries.

   ```
   apt-get install sbcl{,-doc,-source} \
   cl-{asdf,cffi}
   ```

3. Install Emacs and SLIME *(Not strictly required.)*

   ```
   apt-get install emacs{,-goodies-el} cl-swank \
   cl-swank slime common-lisp-controller
   ```

4. Install OpenGL.

   ```
   apt-get install libgl1-mesa-dev \
   libglu1-mesa{,-dev} libglut3{,-dev} ...
   ```

Extra Libraries

- Σ, my library of random useful things in Common Lisp.
  `https://github.com/cgore/sigma`
  (Almost completely undocumented.)
- `cl-opengl`, or else we need to do a lot more work.
  `https://github.com/3b/cl-opengl`
  This library provides `cl-glu` and `cl-glut`.

Getting Libraries via Quicklisp

Quicklisp is the less irritating way to get Common Lisp libraries. It is available at `http://www.quicklisp.org`.

```
curl -O http://beta.quicklisp.org/quicklisp.lisp
sbcl --load quicklisp.lisp
(quicklisp-quickstart:install)
(ql:quickload "cl-opengl")
```

Now we should have a working OpenGL in Common Lisp.

Hello Cube

The simplest thing to do in 3D is a plain cube. This is a good test to see if the libraries and dependencies are all okay. Cf. `hello-cube.lisp`, **run** `(hello-cube)`.

Handling Keypresses

It would be nice if we could quit the program just by pressing **Esc**. Cf. `quit-button.lisp`, run `(quit-button)`.

```
(defmethod glut:keyboard
           ((w quit-button-window) key x y)
  (declare (ignore x y))
  (when (eql key #\Esc)
    (glut:destroy-current-window)))
```

Changing Colors

We would like to be able to change the colors of the cube. Cf. `colors.lisp`, run `(colors)`.

We need new accessors on the window class:

```
((red :accessor red :initform 1)
 (green :accessor green :initform 1)
 (blue :accessor blue :initform 1))
```

We change the color definition:

```
(gl:color (red w) (green w) (blue w))
```

We call an update function:

```
(glut:post-redisplay)
```

Moving the Camera

We would like to be able to move around the camera within the scene. Cf. `movement.lisp`, run `(movement)`.

We make class attributes and keyboard code like with the colors, and update the camera like this:

```
(glu:look-at (eye-x w) (eye-y w) (eye-z w)
             0 0 0  ; look pos
             0 1 0) ; up vector
```

Smooth Lighting

Wireframes are kind of ugly, and you don't really need OpenGL to make them. So, we would like to use smooth lighting model on solid objects.

We have to almost completely gut the implementation of the `glut:display-window` and the `glut:display` methods. We also change out the cube out for a more complex scene.

Cf. `lighting.lisp`, run `(lighting)`.

Projection Matrices

We use `gl:with-pushed-matrix` to work on a local projection matrix.

This is so that we can apply local translations, which we do via `gl:translate`, *"pushing out"* the local matrix in some direction. Thus, the cubes are not all at the center of the scene.

Rotations can also be applied via projection matrices, and probably other fun stuff too.

Rotation

Rotation is achieved via the `gl:rotate` function, which takes:

- θ, the angle of rotation.
- x, for the x axis.
- y, for the y axis.
- z, for the z axis.

Cf. `rotate.lisp`, run `(rotate)`.

The GLUT Clock

We typically want stuff to happen in the background. Cf. `glut-clock.lisp`, **run** `(glut-clock)`.

That Was Ugly! We Need Double Buffering!

This is pretty easy to do. Cf. `double-buffered.lisp`, run `(double-buffered)`.

The important bits are changing the window mode to:
`:mode '(:double :rgba)`
And replacing `(gl:flush)` for `(glut:swap-buffers)`.

Other Cool Demos

- `rainbow-cubes.lisp` – lots of colorful cubes in a cube.
- `blue-cubes.lisp` – the same, just all blue.

What I Still Need to Learn

- Applying textures seems complicated. Cf. `cl-opengl/examples/misc/render-to-texture.lisp` for an example.
- Loading 3D models from files. I have no idea how to do this.
- Lots of other stuff . . .

Questions?