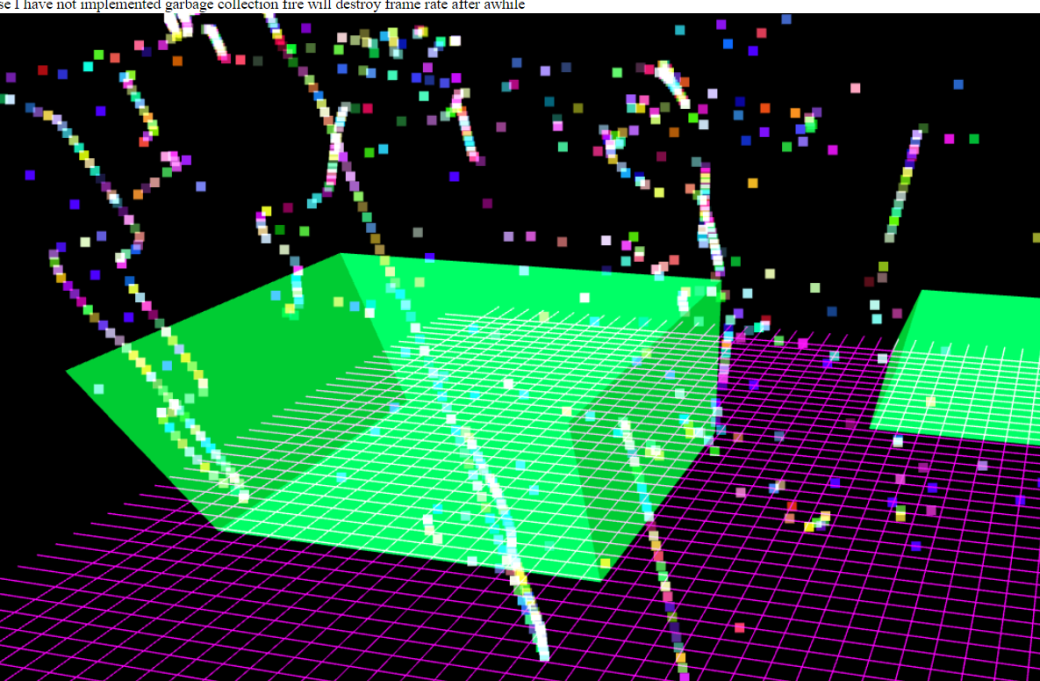


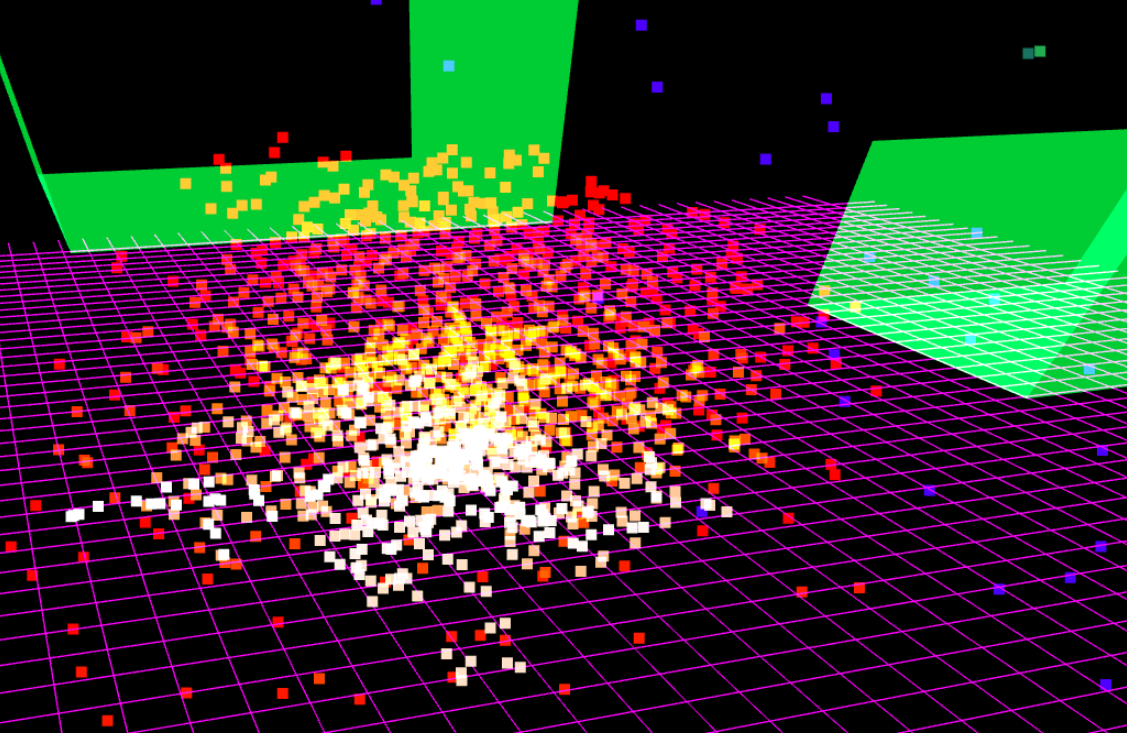
* This is the initial screen. The camera is initially in motion to help you get a sense of 3d

**CAMERA CONTROL:**

* The camera keys a,z,s,x,d,c, move the position of the camer
* the arrow keys and f and v keys can help you rotate the eye lookat point on the camera (a little funky because I have it set to move the lookat point around the origin). Having the camera circle also messes with the parameters so the easiest way to get the camera to a specif point is to restart it and immediate turn off the 'camera circling'

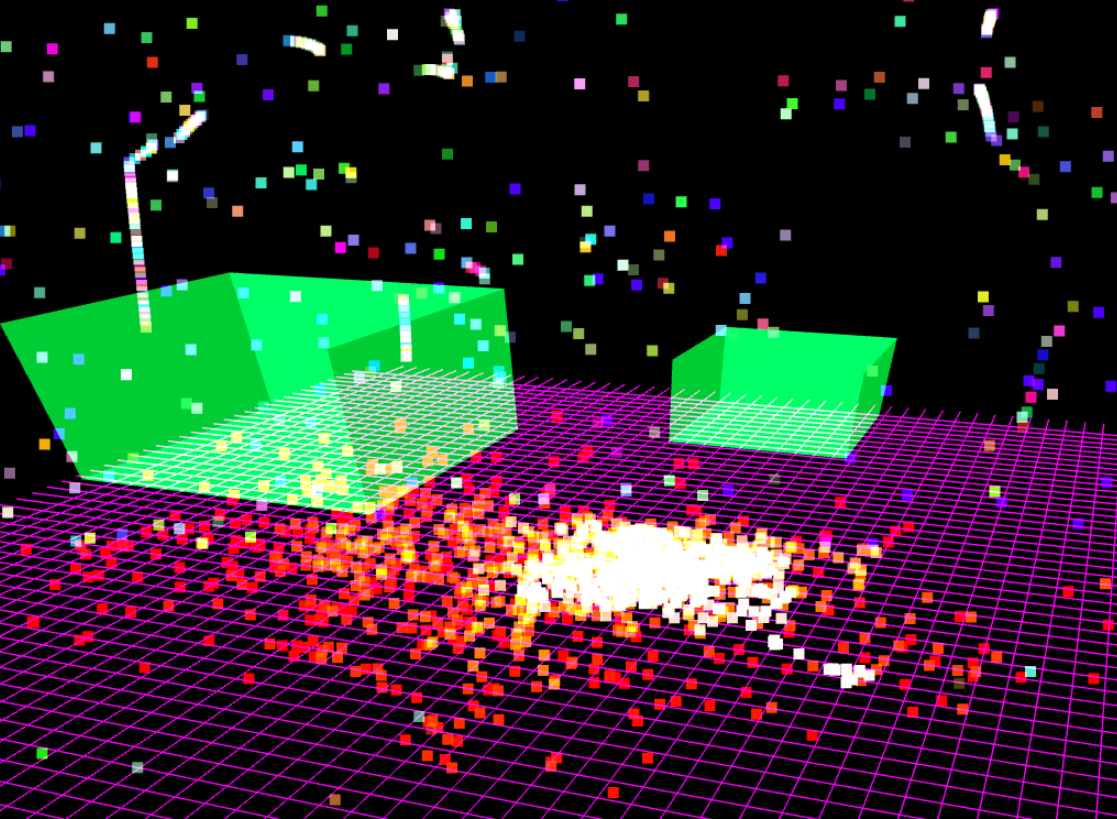
**SHOOTING PARTICLES**

**You can shoot particles with the cursor. Just point and click. It's pretty fun =)**

**In order to do this I had to you the inverse of the projection and world matrix to convert the click point to a 3d point. The particles you shoot will eventually disappear  
**

**FIRE:**

* **The fire shoots particles mostly up with a little randomness side to side.**
* **There is also a "wind" force that pushes the particles up**
* **as the particles age, they become lighter and the wind pushes them harder**
* **the color changes from white to orange to red with age**

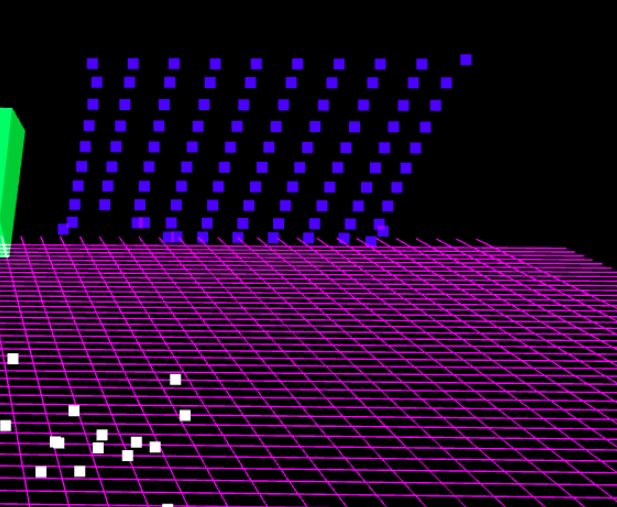
****

**ELECTROSTATICS:**

* **Pushing the add charge button adds an electrostatic in the middle of the page. I gave all particles charge to make it's effects apparent**

**SPHERE AND BOX:**

* **There is a sphere and box in the scene that exclude particles**
* **The smaller box is actually a sphere in terms of exclusion - I just did not make is render correctly**

****

**CLOTH:**

**-These blue balls are a clothe of springs linked together**

**-I am using explicit integration so it explodes from energy**

**CODE GUIDE:**

**I used an object oriented approach**

**1. render class**

**-handles setting up webgl, making the scene, and camera control**

**2. box class**

**-draws a box (it has its own buffer object)**

**3. particle system**

**-handles all particle movement and particle drawing**

**4. drawables**

**-the box class and system have their own draw functions**

**5.ground plane**

**-also is a drawable. Has it's own buffer object**

**6. SPECIFICS OF PARTICLE SYSTEM**

**-the specifics of the particle system are the same as described in class. there are a few state vectors that update according to the methods described in class and the slides**