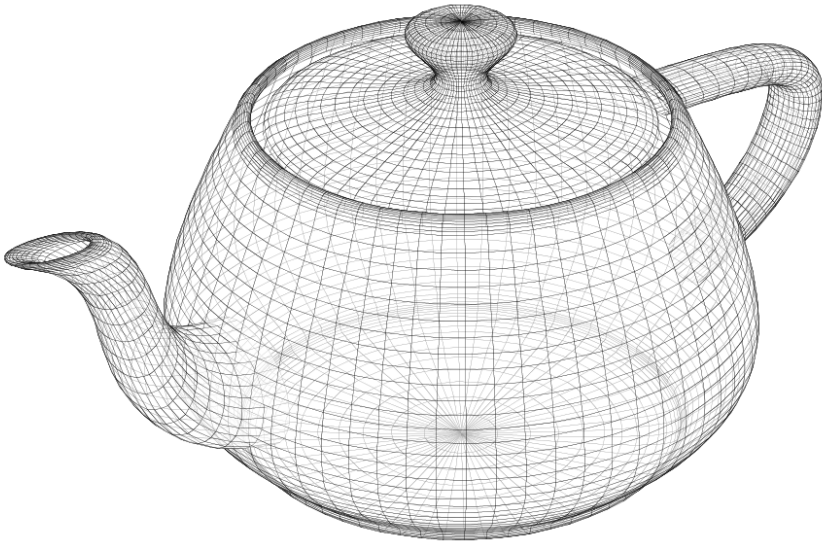


Distance Fog

...and quaternions for navigation



Interactive Computer Graphics
Professor Eric Shaffer

The background image is a foggy scene. In the foreground, there is a large, dark, leafless tree with many branches. To the left, there is a large, yellowish-brown building. In the background, there is a building with a dome and a clock tower. The sky is blue with some clouds. The word "Fog" is written in a white box in the top right corner.

Fog

Fog is an example of participating media

These are typically some sort of particulate

scatters and otherwise changes light

There are a number of sophisticated ways to render such phenomena

we'll use a simple one

For something complex, check out

<http://www.gdcvault.com/play/1023519/Fast-Flexible-Physically-Based-Volumetric>

Distance Fog

Just mix a fragment color with a fog color

The farther the fragment from the viewer, more fog color

...and less original color



Fog in WebGL and GLSL

- We implement fog in the fragment shader
- Compute the distance from fragment to camera

`gl_FragCoord.xyz` is the window-space position.
`gl_FragCoord.w` contains the inverse of the clip-space w

```
float fogCoord = (gl_FragCoord.z/gl_FragCoord.w);
```

- And define a fog color
 - White is a good choice...especially with a white background

```
vec4 fogColor = vec4(1.0, 1.0, 1.0, 1.0);
```

Mixing in Fog

Use linear interpolation to mix fog with the fragment color

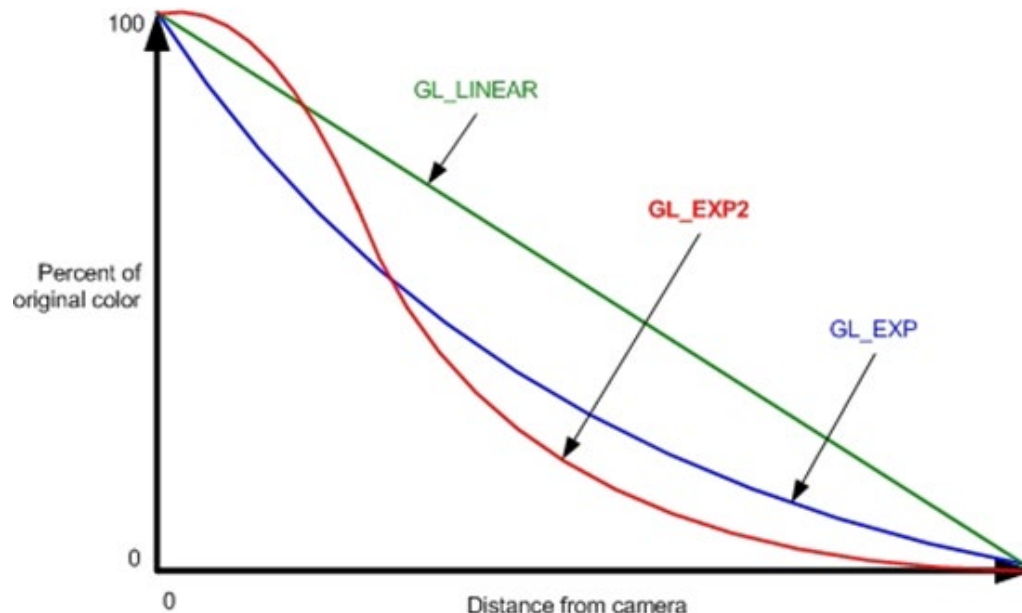
```
gl_FragColor = mix(fogColor, fragColor, fogFactor );
```

- `fragColor` is the color computed by your shading equation
 - Probably Blinn-Phong
- The `mix` function is built-in GLSL function that performs lerp
- But what is `fogFactor`?

Fog Factor

```
const float LOG2 = 1.442695;  
float fogDensity = 0.0005  
float fogFactor = exp2( -fogDensity * fogDensity * fogCoord * fogCoord * LOG2 );  
fogFactor = clamp(fogFactor, 0.0, 1.0);
```

- Fog factor determines how much of the color is fog
 - A value of 1 means all fog in this case...a value of 0 means no fog
- You can see the curve below...we're computing GL_EXP2



On the next foggy day, you can try to verify by eyesight if fog works this way...

Fog Factor Debugging

- In case you have trouble, try just implementing the linear curve for the fogFactor...see if that works.
- Can also render your fogCoord values
 - Set `gl_FragColor` to `(fogCoord, fogCoord , fogCoord , 1.0)`
 - See if the image makes sense...darker means fragment is closer

Flight: Orientation

- Lots of ways to implement flight...
- MP requires the use of quaternions
- One option to change orientation:
 - Set up an initial view using `mat4.lookat`
 - Keep a quaternion that records current orientation
 - Each frame:
 - Capture key presses as Euler angles
 - Construct a temporary quaternion based on the Euler angles.
 - **`quat.fromEuler` in `glmMatrix` library...get current release!**
 - Update the orientation quaternion using the temporary quaternion
 - How?
 - Update the view matrix using the orientation quaternion
 - How?

Flight: Moving Forward

- Keep a user set speed factor
 - User can adjust
- Move in the direction of the current orientation
 - By $\text{speedFactor} * \text{directionVector}$
 - You'll need to experiment to find appropriate speedFactor
- What's the current directionVector?
 - Think about how you could compute it....