CIS367 - GFX

Winter 2024

**HW 2.1 : Interaction and Animation**

**Due February 8th by 11:59pm**

This will get you thinking about interaction and animation. This is just a short little demo for how to animate/interact. Your formal HW is more detailed.

**Setup**

1. Make a copy of triangle.js/html in the demos folder and call them triangle-interact.js/html.
2. Resize the vertices so that all 1's are now 0.25's (don't remove the negatives)

**Interact**

1. Load in jQuery (in the <head> block) -- add this:

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.js"></script>

1. Add an event listener inside the window.onload function in triangle-interact.js:

window.addEventListener(

"keydown",

function (e) {

console.log("Key: " + e.key);

},

false

);

Check the log to make sure you're seeing things happen.

*Note – keyCode is deprecated. We’re now going to use the .key property and directly check the value.*

**Animate**

Let's move those vertices! Define a new set of floats in your vertex shader:

uniform float x;

uniform float y;

And have the gl\_Position now update (replace the gl\_Position = vPosition):

gl\_Position.x += x + vPosition.x;

gl\_Position.y += y + vPosition.y;

gl\_Position.z = 0.0;

gl\_Position.w = 1.0;

(Don't worry, you'll play with these in the formal lab)

**BACK TO JAVASCRIPT**

Create some globals at the top:

let x = 0.0;

let y = 0.0;

let xLoc, yLoc;

Link them to your shaders after the call to gl.useProgram:

xLoc = gl.getUniformLocation(program, "x");

yLoc = gl.getUniformLocation(program, "y");

And in render, update your x/y positions:

x += 0.1;

y += 0.1;

gl.uniform1f(xLoc, x);

gl.uniform1f(yLoc, y);

and at the bottom of the function, make it *animate*:

window.requestAnimationFrame(render);

Your triangle should fly off the screen. Neat, you'll do something similar in your HW. Let's use that jQuery to differentiate us now:

Add another global variable:

let dirs = [null, null]; // horizontal, vertical

And then in your call to addEventListener add this:

(Look in your web console for the corresponding key -- replace \_KEYCODE with the correct value)

if (e.key == LEFT\_KEYCODE) {

dirs[0] = false;

} else if (e.key == RIGHT\_KEYCODE) {

dirs[0] = true;

} else if (e.key == UP\_KEYCODE) {

dirs[1] = true;

} else if (e.key == DOWN\_KEYCODE) {

dirs[1] = false;

} else if (e.key == SPACE\_KEYCODE) {

dirs[0] = null;

dirs[1] = null;

}

And then at the bottom (in render), update your call to updating x and y:

if (dirs[0] === true) // move right

x += 0.01;

else if (dirs[0] === false) // move left

x -= 0.01;

if (dirs[1] === true) // move up

y += 0.01;

else if (dirs[1] === false) // move down

y -= 0.01;

**Upload this to your EOS or GitHub page, link to it in your main page, and copy/paste the link into Blackboard.**

Extra credit?  *Add other triangles to control or make it rotate on a new keypress. Let me know this feature is available when you submit.*