Ejercicio 5 – Proyecciones con WebGL

Este ejercicio tiene como objetivo implementar una aplicación WebGL poniendo en práctica todos los conceptos estudiados en el tema 5 de la asignatura “Proyecciones con WebGL”.

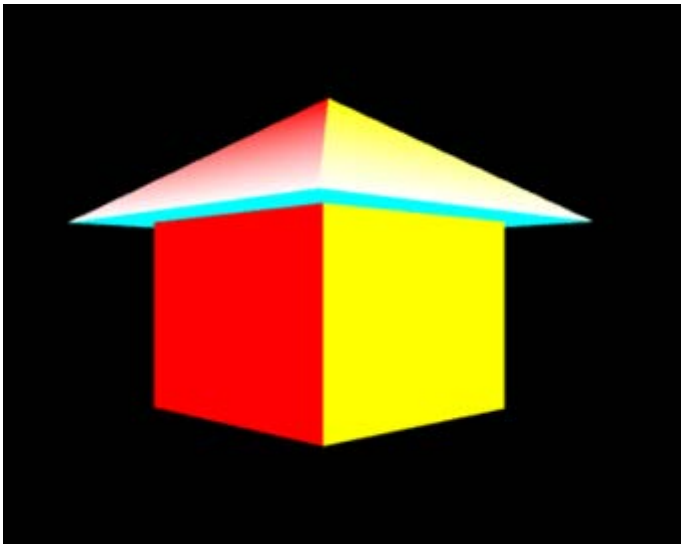
Como resultado de tu práctica deberás generar un **único fichero HTML** que deberás subir al Aula Virtual.

Puntos totales posibles del ejercicio: 10

Instrucciones

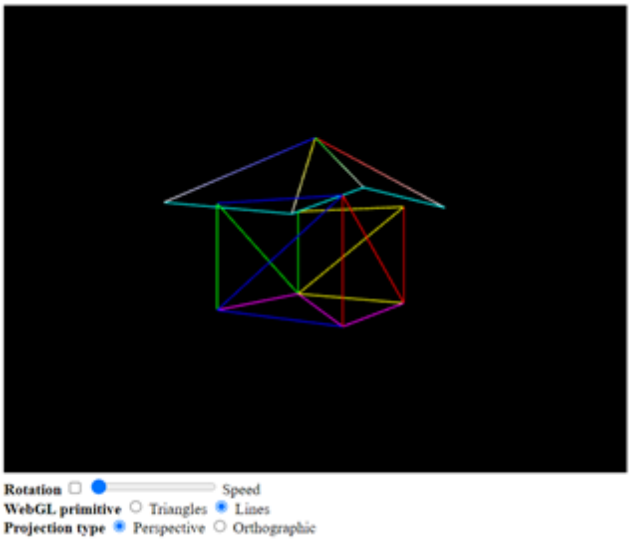
Partiendo del ejemplo visto en clase “proyección en perspectiva”, se pide hacer las siguientes modificaciones:

1. En lugar de un cubo, se dibujará la siguiente figura, la cual consta de una pirámide de 4 lados, y un cubo. Los colores pueden variar, pero hay que tener en cuenta que el degradado de la pirámide tiene que ser del color elegido para esa cara del cubo hacia el blanco. Además, esta figura estará rotando constantemente solo sobre su eje Y.



2. La rotación de la figura se podrá desactivar desde la interfaz de usuario mediante una casilla de verificación (*checkbox*).
3. La velocidad de rotación inicial (incremento de 0.01) se verá modificada de 1 a 3 a través de un botón de rango (*range*).
4. La primitiva WebGL usada para dibujar la escena se podrá elegir desde la interfaz de usuario mediante botones de opción (*radio buttons*). Las primitivas serán triángulos (`gl.TRIANGLES`, opción por defecto) y líneas (`gl.LINES`).
5. El tipo de proyección usado para dibujar la escena se podría elegir desde la interfaz de usuario mediante botones de opción (*radio buttons*). Las opciones son 2: proyección en perspectiva (opción por defecto) y ortogonal.

6. Se deberá incluir un manejador de evento que escuche la rueda del ratón (`wheel`) en toda la página web de modo que al girar la rueda hacia delante se decremente en una unidad la coordenada z de la posición inicial de la cámara (implementada con la función `mat4.lookAt()` del ejemplo original). Cuando la rueda gire en el sentido inverso, la coordenada z de la posición de la cámara se incrementará en 1 unidad.



Ayuda

Puedes incluir los controles necesarios en la interfaz de usuario como sigue:

```
<body onload="init()">
    <canvas id="myCanvas" width="640" height="480"></canvas><br>
    <b>Rotation</b>
    <input type="checkbox" name="rotation" checked>
    <input type="range" name="speed" min="1" max="3" value="1" step="1"> Speed<br>
    <b>WebGL primitive</b>
    <input type="radio" name="primitive" value="triangles" checked> Triangles
    <input type="radio" name="primitive" value="lines"> Lines<br>
    <b>Projection type</b>
    <input type="radio" name="projection" value="perspective" checked> Perspective
    <input type="radio" name="projection" value="orthographic"> Orthographic<br>
</body>
```

Para leer los valores de los diferentes campos (checkbox, radio, range) puedes usar las siguientes sentencias en JavaScript:

```
var rotationChecked = document.querySelector('input[name="rotation"]:checked');
var primitiveValue = document.querySelector('input[name="primitive"]:checked').value;
var projectionValue = document.querySelector('input[name="projection"]:checked').value;
var speed = document.getElementById("speed").value;
```

El manejador de eventos para controlar la rueda del ratón se puede implementar usando el siguiente fragmento de código. Ten en cuenta que la variable `z` se va a utilizar como coordenadas z de la posición de la cámara (`mat4.lookAt()`) únicamente cuando se utiliza la vista en perspectiva. El valor por defecto de esta coordenada en el ejemplo original es `3`:

```
// Event listener for mouse wheel
document.addEventListener('wheel', function (event) {
    z = event.wheelDelta > 0 ? z - 1 : z + 1;
});
```