**Task 1**

In order to accomplish Task 1, the setTexture function in project2.js's MeshDrawer class has to be modified to support textures that are not powers of two dimensions. To support such textures, texture settings must be changed. A check to determine if image's width and height are powers of two is present in the current code. If not, we must adopt a new strategy:

Texture Wrapping: Assign gl.CLAMP\_TO\_EDGE to the texture wrapping parameters (gl.TEXTURE\_WRAP\_S and gl.TEXTURE\_WRAP\_T). With this modification, the texture may be accurately mapped even if its dimensions are not a power of two.

Texture Minification Filter: Set to gl.LINEAR in the texture filtering configuration (gl.TEXTURE\_MIN\_FILTER). Mipmaps, which are inappropriate for textures that are not powers of two, are not required when using this filtering option.

Texture Magnification Filter: Since it is often set to either gl.LINEAR or gl.NEAREST, the texture magnification filter does not need to be changed. So after this implementation result is below.

A close up of a ball

Description automatically generated

**Task 2**

The objective of Task 2 is to provide scene with basic lighting such as diffuse and ambient lighting. This calls for modifications to the fragment shader (meshFS) and the MeshDrawer class in a few places:

Constructor Modifications: Initialize extra uniform places for lighting related properties, including ambient light intensity and light position, in constructor.

Updating setMesh: The setMesh function handles normal vectors they are necessary for lighting computations, in addition to establishing vertex and texture coordinates.

Changing the draw method: Update draw method to provide shader access to computed light location and ambient light values. Make sure the shader receives normal vectors as well in order to calculate illumination.

Toggle lighting and adjust ambient light intensity by implementing the enableLighting and setAmbientLight methods they should change the corresponding uniform variables in the shader.

Fragment Shader (meshFS) Updating: To determine the final color depending on the texture color, diffuse light and ambient light, modify fragment shader. The computation of diffuse light is contingent upon direction of light and the surface normals.

A circular object with a black background

Description automatically generated

A close up of a football ball

Description automatically generated