

**Graphics Programming**

**Software Development for Computer Games**

**Bill Mitchell**

**S1621554**

*I confirm that the code contained in this file (other than that provided or authorised) is all my own work and has not been submitted elsewhere in fulfilment of this or any other award.*

*Bill Mitchell.*

**Contents**

[Gooch Shader 3](#_Toc7730897)

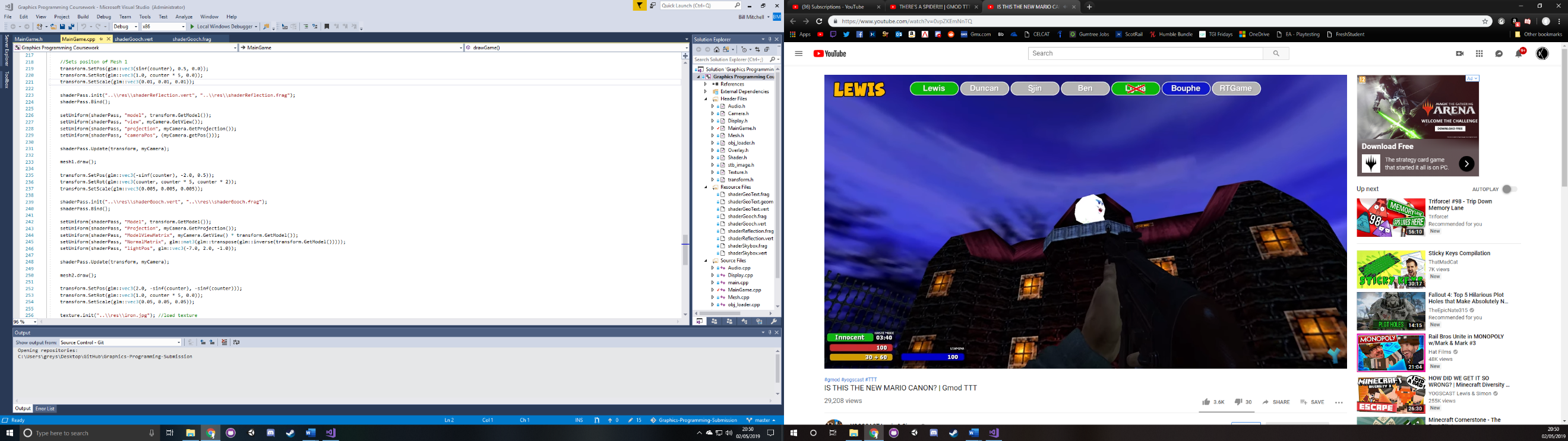
[Main Game 3](#_Toc7730898)

[shaderGooch.vert 3](#_Toc7730899)

[shaderGooch.frag 4](#_Toc7730901)

# Gooch Shader

## Main Game



The shader is initiated in the in the DrawGame() function of MainGame.cpp by calling the init() function of the shader script and passing in the location in the file of the shaderGooch fragment and vertex files. The shader is then bound, and the Uniforms are then passed in by making use of the SetUniform function within MainGame.cpp and the shader is passed in along with the name of the Uniform within the shader and the value of which that uniform should be set.

## shaderGooch.vert

The variables within shader.vert are; two Vector3’s, VertexPosition and VertexNormal, which are bound to a location when the shader is initialised and called from the location. Three Matrix4’s, Model (which holds the model matrix), Projection (which holds the projection matrix) and ModelViewMatrix, a Matrix3, NormalMatrix, and a Vector3 lightPos (which holds the position of the light) are passed into the shader as a uniform from MainGame as discussed earlier. There are also three variables set as out, which allow the variables to be passed to the fragment shader, or other files within the shader. These variables are NormDot, a float, ReflectVec and ViewVec, both Vector3’s.

## 

The main function within the vertex shader is used to set the values that are to be passed out to the fragment shader. This is done by first creating a local vector3 called Pos that is equal to a vector3 of the ModelViewProjectionMatrix multiplied by the VertexPosition which gets the position of the vertex.

Then a Vector3 called Normal is set to the NormalMatrix multiplied by the VertexNormal and normalized to give the normalised version of the Normal vector.

The final Vector3 to be created is light and it is the lightPos minus the Position of the vector and normalized to get whether the light is affecting the vector.

The View Vector is then set to the negative of the Pos vector and normalized. The Reflect Vector is set using the reflect function passing in the negative of the light vector and the Normal variables and then this is normalized. Finally, the NormDot variable is set which is the dot product of the Normal and light variables. These three variables are passed to the fragment shader to be used there.

gl\_Position is then set to the ModelViewProjection Matrix and multiplied by the VertexPosition which sets the position of the vertex as a Vector4 rather than as a Vector 3 set in the Pos variable.

## shaderGooch.frag