1. **Abtract**

In this project we will build a C++ application to sumulating and rendering waves on water surfaces. It will also be able to…. We will use QT Creator for programming, QT for window management and OpenGL for 3D rendering.

1. **The function we will use**
2. **Realisation**

*2.1 Creating an interface with QT*

First oﬀ, we create a new QT widget application. This allows us to use QT Creators design feature to set up our application’s interface. A new OpenGLWidget is placed and will be used as a placeholder for a new custom class inheriting OpenGLWidgets functionality. This class, called OGLWidget, needs to implement the following methods: initializeGL (for setting up OpenGL), paintGL (for doing the actual rendering), resizeGL (for handling resizes of the display window). Additionally, the functions stepAnimation, SetMaterialColor and InitLightingAndProjection1 are used.

*2.2 Creating the data structure*

The data structure is separated in diﬀerent classes. The basic class "Waves" contains the information of the waves like sine waves, Height-Field, coordinate, direction vector, phase velocity, the frequency and the wave number.

The class "Wavesurface" contains the wavesfunction . Logic and data regarding the computation of quad meshes is stored in a seperate class, as are Bezier surfaces and rotational sweep surfaces. In order to allow for easier use of a two dimensional matrix of vertices, a wrapper class containing a two dimensional vector of vertices is introduces.

1. **Mesh**

After creating the required data structure, a method to make a mesh for the waves.

1. **Rendering as a wireframe**

Depending on the desired way of rendering the object, diﬀerent draw methods are implemented. These methods are then being called from the paintGL() function.

* 1. *Rendering as a solid*

After drawing the object as a wireframe we want to draw it as a solid cube with lighting. This is being achieved in the method drawQuads() which once again iterates over the list of quads. This time using GL\_Quads, the four vertices of a quad are connected and the area inbetween is ﬁlled. The normal vector for this is calculated using the cross product of the two diagonals vectors.

1. **At**
2. **Recalculate Mesh**
3. **Calculate the wave Height**
4. **Vertex Shader**
5. **References References**

[1] Prof. Dr. Martin Hering-Bertram, Lecture CG18\_1, HSB, 2018.

[2] Prof. Dr. Martin Hering-Bertram, Lecture CG18\_2, HSB, 2018.