Dokumentation Computergrafik

Projektbeschreibung

Das Thema der Projektarbeit ist ein Rammbock der vor einem Burgtor steht und dieses dann mit dem Rammbock attackierd. Die Projektarbeit wurde für die Windows 32-Bit Plattform mit der Programmiersprache C++ entwickelt. Als IDE hierfür diente Visual Studio 2019 (VS16). Die für die Enwicklung genutzten API ist OpenGL32 zur Unterstützung wurden die Libraries GLEW, GLM für mathematische Funktionen, AssImp für das erstellen der Modelle innerhalb des Programms und SOIL2 um die zuvor in Blender erstellten und hinzugefügten Texturen auf die Modelle zu übertragen.

Zur Installation des Programmes genügt das Projekt in Visual Studio zu öffnen und einen 32-bit Release zu erstellen. In den dabei entstehenden Ordner Release müssen die Datein des Ordners "Externe" eingefügt werden. Danach Ordner Release in dem Ordner "RammbockComputergrafik" zu lokaliesieren diesen ggf. zu verschieben und die darin enthaltene .exe Datei auszuführen.

Quellen & Literatur

https://wiki.delphigl.com/index.php/Hauptseite

https://www.youtube.com/watch?v=yRYHly3bl2Q Teile der Tutorialreihe vom Channel Pilzschaf

http://www.opengl-tutorial.org/

http://ogldev.atspace.co.uk/index.html

https://learnopengl.com/

https://open.gl/

https://texture.ninja für Texturen

Datenfluss

Beim starten der Applikation wird von der WindowFabrik.cpp durch aufrufen der OpenWindow Methode das Fenster und der Kontext erzeugt welcher dem Nutzer erscheint. Das so erzeugte Objekt ruft dann die Show() Methode auf welche die geladenen Modelle anzeigt. Zunächst wird jedoch das Shader Objekt erzeugt. Beim erzeugen des Shader Objektes werden der Fragment- und Vertex-Shader mitgegeben. Diese werden ausgelesen und an das Shader-Programm gebunden. Der Shader wird innerhalb der Model.h Datei auf

die Modelle angewandt. Die Modelle werden durch AssImp ausgelesen und in die VBO, VAO und EBO's des Mesh.h gespeichert welche als Basis jedes Modells fungieren. Nachdem die Modelle als Mesh gespeichert werden wird die Textur des Modelles gespeichert. Beim wiederholten zeichnen der Modelle sorgen die Transformationsmatrix für die Bewegung des Modells und die Projekionsmatrix dafür das die Modelle in der richtigen Skalierung angezeigt werden.

Komponentenbeschreibung

main.cpp

Die main.cpp ist der Punkt des Systems die verschiedenen Funktionen des Programms zusammenführt. Die main.cpp erzeugt das Window Objekt das in dem die Modelle für den Nutzer sichtbar machen und führt die Show()-Methode aus die Modelle zeichnet.

Windowfabrik

enthält die Methode zum erzeugen des Windows

OpenWindow

Enthält alles was in dem geöffneten Window passiert von der Initialisierung von GLFW und GLEW beim erzeugen des Window. Bishin zum erzeugen des Shader-Objektes welches auf die zu zeichnenden Modelle angewandt wird und die Transformationsmatrix welche die Bewegung der Modelle bestimmt.

Shader

liest die core.vs und core.frag Dateien aus und bindet diese and das Shader Programm welches beim aufruf des Methode ShaderStarten() ausgeführt wird.

Window.h

enthält die Window-Klasse welche später durch die WindowFabrik aufgerufen wird.

MeinMesh

erzeugt die VBO,VAO und EBO's erzeugt in welchen die Vertices für die Position, Normals und Texturen gespeichert werden. Dies sind die Koordinaten für die einzelnen Punkte der Modelle

Model

liest die zuvor in Blender erstellten Datein der 3D-Modelle aus speichert die darin enthaltenen Daten in Meshes und wertet diese aus um später bei aufruf der Draw()-Methode diese innerhalb dies Window zu zeichnen. Hier werden auch die verschiedenen Lichtverhältnisse auf die Modelle angewendet. Zum auslesen der Datein wird AssImp

verwendet.

Code Konventionen

Eines der Leitbilder beim erstellen des Programmes war es dafür zu sorgen das eine Komponente des Programmes möglichst immer nur eine Aufgabe hat. Der Code innerhalb dieser Komponenten sollte möglichst lesbar sein auch für Dritte. Dazu trägt auch die Trennung der Verantwortung und die sprechende Bennenung von Variablen und Methoden.

Projektstruktur

Die Projektstruktur wurde mithilfe eines Diagrammes dargestellt welches UML CG benannt wurde.

Soll/Ist Vergleich

Ziel war es ein Modell eines Burgtores und eines Rammbockes zu erstellen, der Rammbock sollte die Ramme bewegen um so das Tor zu attackieren. Bedauerlicherweise gelang es mir trotz intensiver Fehlersuche nicht den Fehler zu behenben der dazu führte, das die von mir zum Laden der Modelle ausgewählte Bibliothek AssImp, die zuvor von mir erzeugten Modelle aus den erstellten Datei zu lesen. Da AssImp aus mir nicht ersichtlichen Gründen nicht in der Lage war diese zu öffnen, unabhängig von deren Datei Format. Die von mir erstelleten Modelle und verwendeten Texturen sind in der Projekt Mappe unter Modelle zu finden als auch gesondert beigefügt(Sowohl im .obj als auch im.blend Format).