Deepspace 16er Blech
Second Submission

Update 1:
Die Steuerung wurde umgestellt, so dass nun der Bewegungsradius nicht mehr durch die Geschwindigkeit eingeschränkt wird. Ausgelegt ist die Steuerung auf eine Tastatur und Maus Steuerung. Die Maus gibt die Blick- und Fahrtrichtung vor, mit W/Left Shift/Up Arrow lässt sich beschleunigen, mit S/Space/Down Arrow lässt sich abbremsen. Zusätzlich besteht die Möglichkeit mir A und D nach links oder rechts zu strafen.

Update 2 : 2ter Effektpunkt
Ein simpler Partikeleffekt der auf der Graphikkarte berechnet und erneuert wird wurde eingebaut. So wird nun, immer wenn eine Bierkiste zerstört wird, ein Haufen Partikel erzeugt, um die Tragik des Verlustes herauszustreichen (es war davor zu unauffällig wie die Bierkisten einfach verschwanden). Vorlagen waren die bereitgestellten Folien und das entsprechende Video, sowie http://www.opengl-tutorial.org/intermediate-tutorials/billboards-particles/particles-instancing/ (auch aus den Folien), und http://ogldev.atspace.co.uk/www/tutorial28/tutorial28.html.

Das Raumschiff lässt sich mit LEFT SHIFT beschleunigen und mit SPACE abbremsen, es fährt dabei immer in die Blickrichtung der Kamera. Zusätzlich gibt es die Möglichkeit mit WASD in Richtung Bildschirmrand (z.B: W führt zum oberen Bildschirm Rand) zu steuern. Mit zunehmender Geschwindigkeit, nimmt die Möglichkeit zu lenken ab, dies soll nicht nur realistisch sein, sondern auch das Spiel erschweren.

Gameplay Ablauf:
Nach dem laden (leider keine loadscreen) kann man das raumschiff nicht bewegen, um die anfangsposition nicht zu gefährden, da die Maus die Blickrichtung gleich am anfang ändern kann. Deswegen wird mit Enter bestätigt dass das Spiel gestartet wird und die Controls aktiviert werden. Danach sollte man versuchen den vorhandenen Hindernissen auszuweichen und die ISS zu erreichen. Bei Erreichen der ISS wird eine Nachricht ausgegeben, ob das Spiel gewonnen oder verloren wurde.

Der Spieler startet mit 3 Bierkisten, bei jeder Kollision verliert er eine davon. Wird die ISS ohne Bier erreicht, gilt das Spiel als verloren (und eine entsprechende Nachricht wird ausgegeben).

Implementierung – Kurzbeschreibung
Modelle und Texturen werden aus obj. und image dateien gelesen und als VAO verpackt. Die vorhandenen Modelle sind als Dateien vorhanden, die in-game szene ist hardgecoded. Die Kometen werden in Position, Bewegungsrichtung und Größe jedoch zufällig verteilt.

Bewegungen werden in die Modelmatrix einberechnet, den rest übernimmt der vertexshader.
Die Kamera befindet sich quasi auf einer Kugel, die um das Schiff aufgespannt wird. Mit lookAt() wird das Schiff fixiert. Bei Bewegung des Schiffs, bewegt sich die Kamera mit. Die Bewegungsrichtung des Schiffs wird relativ zu Kameraposition und Schiffposition berechnet
(vektor von Schiff zu Kamera errechnen, per Kreuzprodukt werden Bewegungsvektoren errechnet)

Manche Modelle werden mit einem Normalshader gezeichnet, Farben entsprechen dabei den Normalvektoren. Die Kollisionserkennung ist selbst implementiert, und basiert auf Sphären (da die meisten unserer Objekte sehr kugelähnlich sind). Ein animiertes Objekt, ein Satellit, dessen „Arme“ sich drehen, ist von der Startposition aus gesehen, links unten zu finden. Komplexe Objekte sind zum Beispiel die ISS, die auch unser Ziel darstellt.

Die Modelle wurden in Maya und Blender erstellt, die Grundlage für die ISS kommt von der NASA.

Effekte:
Momentan ist nur ein Effekt implementiert, nämlich ein Static Level of Detail. Dieser ist bei den Modellen der ISS, sowie bei den Modellen der Hindernisse zu sehen (also bei dem animierten Sateliten, und dem

Hindernis, dass direkt zwischen ISS und der Startposition des Spielers steht) Die Modelle wurden in Maya erstellt, bzw. im Falle der ISS vereinfacht. Es gibt von allen LOD Modellen drei Versionen, einmal low, medium und highresolution. LOD wurde ohne wirkliche Vorlage umgesetzt.

Externe Bibliotheken:

ASSIMP (http://assimp.sourceforge.net/)
FreeImage (http://freeimage.sourceforge.net)
GLEW (http://glew.sourceforge.net/)
GLFW (http://www.glfw.org/)
GLM (http://glm.g-truc.net/)
ogg_to_pcm und wwwviaudio (werden nicht verwendet, sind aber im repository)
Die Methode zum laden der Direct Draw Surface und zum Berechnen der UV und Vertex Values der Schriftart wurde von http://www.opengl-tutorial.org/ übernommen

Features:

Die Modelle werden mit einem Modelloader und Assimip geladen
Texturenimages werden geladen und verwendet
Eine einfache Collissiondetection mit Win-Condition
Steuerung, mit beschleunigen und bremsen.
Bierkisten die automatisch mit dem Raumschiff mitfliegen
Rotierende Satelliten
Static LOD
Randomisierte Kometen
Realistische Beschränkung der Navigationsfähigkeit bei zunehmender Geschwindigkeit