COMPUTERGRAFIK 1. BERICHT ZUR AUFGABE 2

ALEXANDER BUYANOV (806984) AND PHILLIP REDLICH (791806)

1. Bericht

Die erste Teilaufgabe befasste sich mit der Implementierung einer Klasse Ray, welche einen Strahl repräsentiert. Sie bildetdie Grundlage für das spätere Ray-Tracing. Zunächst haben wir uns kurz mit der Mathematik hinter dem Strahl beschäftigt. Die Implementierung selbst erwies sich als simpel, da es lediglich um einfache Rechung an einer Vektor-Geradengleichung handelte. Somit war auch der Zeitaufwand hier gering.

Der nächste Schritt in der Aufgabenstellung waren die Camera-Klassen. Auch hier begannen wir zuerst die Mathematik nachzuvollziehen um diese dann mit Verständnis anzuwenden. Bei der Implementierung haben wir zunächst die Musterklasse Camera geschrieben und den Konstruktor erstellt. Dabei haben wir die vergleichsweisen einfachen Berechnungen im Konstruktor durchgeführt. Die rayFor() Methode stelte das größte Problem dar. Aber durch klare Vorgaben der Formeln war dies letzendlich schnell gelöst.

Der letzte Teil der logischen Implementierung befasste sich mit der Geometrie. Hier haben wir einfache Formen im Raum durch die Klassen Sphere und Plane definiert. Wie zuvor war der Konstruktor kein Problem. Auch die Schnittberechnung in hit() an sich stellte kein Problem dar. Jedoch haben wir zunächst nur die Formeln übernommen und keine Überprüfungen durchgeführt. Das heißt wir hatten immer die Variable errechnet, jedoch das Schnittkriterium nicht durch einsetzen eben dieser überprüft. Erst nachdem wir die Klasse Hit definiert haben , wurde die Überprüfung ebenfalls implementiert. Im Endeffekt haben uns die hit() Methoden am meisten Zeit gekostet, da wir im späteren Verlauf noch Fehler verbessern mussten, welche bei der Visualisierung entstanden.

Im letzten Schritt der Bearbeitung sollten wir die Klassen RayTracer und World definieren, wobei World alle Objekte (im Moment Spheres und Planes) übergeben bekommt, welche sie in einer Methode hit() durchläuft und jeweils auf das Schnittkriterium prüft. RayTracer soll hierbei die Visualisierung übernehmen wobei für alle Pixel des zu erzeugenden Bildes ein Strahl erstellt wird, welche dann jeweils an die hit() Klasse von World übergeben wird. Hierbei haben wir zunächst die Implementierung in World zusammengefasst und vereinfacht. D.h. wir haben ein einfache Objekte direkt erstellt, getestet und zeichnen lassen, um die Logik vorerst zu testen. Durch das testen sind wir, wie zuvor erwähnt, auf einige Fehler gestoßen. Nachdem die Objekte erfolgreich erstellt wurden haben wir die Methoden auf die zwei vorgegebenen Klassen aufgeteilt. Auch hier gab es keine nennenswerten Probleme.

Zum Schluss möchten wir noch einmal zusammenfassen. Unsere Lösungsstrategie war relativ einfach und stark an der Aufgabenstellung orientiert. Wir haben uns zunächst mit dem Rechenzettel beschäftigt, wodurch wir eine grobeIdee für die Implementierung der einzelenen Methoden bekamen. Daraufhin sind wir die Klasse nach Aufgabenstellung durchgegangen und haben uns wenn nötig nochmal mit der Mathematik auseinandergesetzt. Probleme sind lediglich durch kleinere Fehler in den Rechnungen aufgettreten. Insgesamt war die Aufgabe angemessen in ihrer Schwierigkeit, wobei der Zeitaufwand grundsätzlich nicht sonderlich hoch war. Jedoch kam es durch aufkommende Fehler oft zu Verzögerungen.