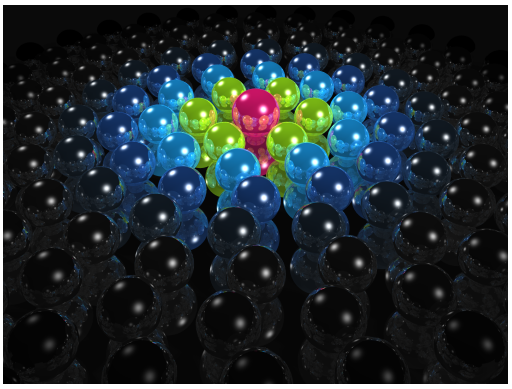


Computergrafik

Paul Sprotte Gabriel Dierks Josef Strunk

12. Mai 2014

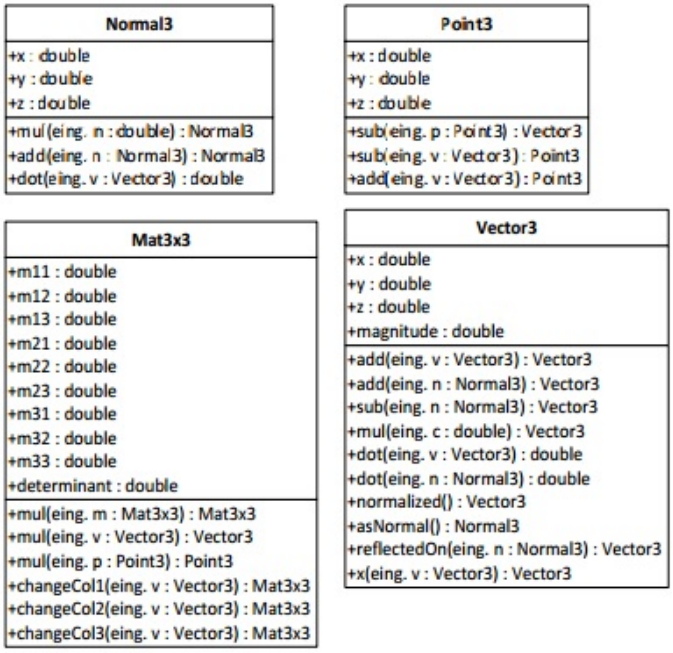
Inhaltsverzeichnis

Kapitel 1

Vorbereitung

1.1 Die Aufgabenstellung

In der Vorbereitungsaufgabe müssen zu erst einmal zwei kleine Programme geschrieben werden. Programm 1 soll ein Dialogfeld zum Öffnen einer Bilddatei starten und das gewählte Bild in einem passend großen Fenster darstellen. Mit dem Lösen der Aufgabe festigt der Programmierer noch einmal seine Kenntnisse mit dem Umgang vom Laden von Datei in Java. Programm 2 soll ein Fenster mit der Größe von 640x480 öffnen und in diesem ein komplett Code generiertes Bild anzeigen. Das Bild soll Pixel für Pixel „bemalt“ werden und bis auf eine rote Diagonale komplett schwarz sein. Hierbei lernt der Programmierer, wie er selbst ein Bild im Code erstellt und dieses Pixel für Pixel ansprechen und darstellen kann. Das ist vor allem wichtig in Hinblick auf den Raytracer wo wir das Bild ja auch darstellen in dem wir für jeden Pixel eine Ray „schießen“. Der zweite Teil der Aufgabe besteht darin eine eigene Matrizen- und Vektorenbibliothek zu schreiben hier für ist ein passendes Klassendiagramm gegeben.



Wichtig hier bei ist zu beachten, dass alle Klassen immutable sind, was bedeutet ihre Attribute sind final und werden nach ihrer Erzeugung nicht mehr geändert. So bedarf es auch keiner getter und setter mehr. Mit dieser Aufgabe bauen wir den Grundbaustein des Ray Traycers und implementieren alle mathematischen Voraussetzungen für die späteren Berechnungen. Für den kompletten Code sind alle Richtlinien der Code-Style-Guideline einzuhalten.

Code-Style-Guideline

- Document everything
- final is your friend
- Check parameters
- Check if a method can be called at the moment
- Keep objects consistent
- Throw RuntimeExceptions
- Take warnings serious and remove them
- Override hashCode() and equals()
- Override toString()
- Consider to make objects immutable

1.2 Bericht

In unserer Gruppe arbeiten wir mit einem Eclipse Plugin namens Saros. Dieses Plugin ermöglicht es zusammen in einem Eclipse Projekt zu arbeiten, man kann wie bei Google-Docs sehen was der andere macht mit ihnen zusammen in einer Datei zu schreiben und von allen bleibt das das Projekt aktuell. Und so haben wir uns an einem Freitag zusammengesetzt und per Skype die Aufgaben erarbeitet die Listen von oben nach unten. Nach dem wir sichergestellt hatten das wir alle Akzeptanzkritieren eingehalten haben, richtet wir uns das geforderte git ein und pushten unsere Quell-Datei in das Repositorium. Das erste Programm startet einen JFileChooser mit geeigneten Filtern, wenn ein Bild gewählt wurde wird es über ImageIO.read(chooser.getSelectedFile()) in eine BufferedImage gesteckt. Als nächstes wird eine JFrame erstellt und ihr mit frame.setSize(image.getWidth(), image.getHeight()) die passende Größe zugewiesen. Über den Umweg das BufferedImage in ein ImageIcon zu steckt um jenes wieder in ein JLabel zu stecken kann man am Ende das Bild der Frame zuweisen. Im letzten Schritt wird mit frame.setVisible(true) das Fenster sichtbar gemacht. Das zweite Programm öffnet eine JFrame in der geforderten Größe und baut die passende GUI. Das schwarze Bild mit der roten Diagonale wird in einer eigenen Canvas Klasse gemalt, der die erstellte Frame übergeben wird. In dieser wurde die repaint so überschrieben, das in ihr ein neues BufferedImage erstellt wird mit der Größe der übergebenen Frame. In dem man dem Image ein ColorModel verpasst und sich dann zugriff auf das Raster holt kann man jeden Pixel einzeln einfärben. In einer passenden Schleife wird jetzt die rote Diagonale gezeichnet. Die Canvas wird schlussendlich der erstellten Frame zu gewiesen und der Frame ein passender resize listener zugewiesen der im Falle die Canvas repaintet. Die Matrizen- und Vektorenbibliothek haben wir im Großen und Ganzen genau so implementiert, wie das Klassendiagramm es uns vorgegeben hat. Die Formel haben wir letztes Jahr in Mathe gelernt und waren schnell umgesetzt. Wir mussten nur darauf achten uns wirklich an alle Punkte der Code-Style-Guideline zu halten. Der Zeitbedarf kann auf gute 16 Stunden geschätzt werden. Einen kompletten Tag 10 Stunden am Stück, um grob alles fertig zu schreiben, einen zweiten Tag von 3 Stunden um zusammen alles zu kommentieren und die IllegalArgumentException wirklich überall einzubauen und einen dritten Tag um den Bericht zu schreiben und alles wie verlangt einzurichten.

Kapitel 2

Schnittberechnung

2.1 Die Aufgabenstellung

In der Schnittberechnung werden die grundlegenden Geometrischen Objekte implementiert dazu kommen die Camera Klassen und eine Raytracer Klasse, um die ersten Bilder anzeigen zu können.

2.2 Bericht

Der Schwierigkeitsgrad ist mit der 2ten Aufgabe schon gut angestiegen. Als erstes haben wir die Camera Klassen implementiert mit Hilfe der Folien war das kein Problem da nach sind wir gleich zu den Geometry Klassen übergegangen. Der Sphere und Plane waren kein Problem und wurden von uns wie auf den Folien umgesetzt. Erst als wir zu Axis Aligned Box gekommen sind gab es erste Probleme. Erst nach 100ten malen durch lesen der Aufgabenstellung haben wir verstanden was wir da wirklich machen müssen. Unsere Schnittberechnung sieht es so aus das wir erst mit allen normalen und den bekannten Ecken überprüfen welche Ebenen der Box wir sehen können und dann auf diese Ebenen die Ebenen Schnittberechnung anwenden. Danach müssen wir von allen Schnittpunkten nur noch heraus finden welcher der kleinste ist und dieser ist unser gesuchten Schnittpunkt. Ähnliche Probleme hatten wir mit dem Dreieck erst mit Hilfe von Ray Tracing from Ground Up haben wir es geschafft wirkliche zu verstehen wie wir diese Methode am besten implementieren und die Cramer Regeln richtig umzuschreiben. Die World und Raytracer Klassen waren dann Gott sei danke wieder etwas leichter und gingen gut von der Hand. Alles in allem hatten wir mit dieser Aufgabe anfangs große Probleme aber nach viel einlesen und arbeiten haben wir es geschafft alles selbst zu implementieren. Einziges Problem ist der Zeit noch das unsere Box noch viele Löcher durch Rundungsfehler anzeigt wir hoffen das mit ihnen zusammen noch verbessern zu können. Der Zeitbedarf kann auf gute 14 Stunden geschätzt werden. Einen Tag 6 Stunden, 3 Tage sacken lassen und nochmal einen kompletten Sonntag.