CGI: BERICHT ZU AUFGABE 1, RAYTRACER

VON SEBASTIAN DASS, MAX NOVICHKOV, SIMON LISCHKA

1. Aufgabenstellung

1.1. **Implementierung einer Klasse zum Laden und Anzeigen eines Bildes, das mit einem File-Open Dialog geöffnet wird.**

1.2. **Pixelweise zeichnen einer Diagonale auf einem schwarzen Hintergrund, welches sich korrekt der Fenstergröße anpasst. Funktionalität zum Speichern des Bildes als PNG-Datei.**

1.3. **Generierung einer Matrizen- und Vektorenklasse, die spezifische Operationen zu 3er Matrizen, Vektoren, Punkten und Normalen anbietet.**

1.4. **Testen der Matrizen- und Vektorenklasse durch DurchfËhrung und Überprfung exemplarischer Rechnungen.**

2. Lösungsstrategien

2.1. **Die grundliegende Klassenstruktur wurde zunächst unter Zuhilfenahme der Java API-Doc geplant. Die Klasse wurden dann zunächst als leere Vorlagen generiert um nach dem Divide-and-Conquer Prinzip eine Stŕuckweise Erarbeitung des Programmes durch die Teammitglieder zu Ermöglichen.**

2.2. **Verschachtelte oder voneinander abhänginge Objekte wurden zunächst strukturell erarbeitet und getestet, damit anschließend die korrekte Funktionsweise im Gesamtkontext hergestellt werden konnte. Fehler in dem Ausgabebild o.ä. wurden zunächst in dieser abstrakteren Betrachtungsweise ignoriert.**

2.3. **Teaminterne Kommunikation und Informationsaustausch um Beiträge aller Mitglieder zu fördern wurde in regelmäßige Meetings vorgenommen.**

3. Implementierung

3.1. **Bei der Implementierung des ImageViewers ging es darum einen mglichst kurzen Lösungsweg zu finden. Aus diesem Grund griffen wir auf ein ImageIcon Objekt zu, welches für die korrekte Darstellung sorgt und die Größ ordnungsgemäß mit dem pack() Befehl anpasst.**

Date: 11.11.2013.

3.2. **Um ein Bild zu generieren und zu Speichern wurde in einer ImageCanvas Klasse ein BufferedImage Objekt als Instanzvariable festgelegt, welche nach extrahieren des Rasters zum Zeichnen in der paint() Methode des Canvas benutzt wird. Mit dieser Vorgehensweise ist es möglich, Klassenintern das BufferedImage Objekt auszutauschen, um die Größ des Bildes z.b. bei Veränderungen der Fenstergröße anzupassen.**

3.3. **Die Vektor- und Matritzenklassen wurden gemäß den Vorgaben implementiert und werfen IllegalArgumentExceptions, wenn die übergebenen Parameter falsch sind. Dies wird vor dem Durchführen der Kalkulation gepruft. Um ein zuverlässiges Testen der Funktionen zu Ermöglichen, wurde zudem ein getrenntes Testpackage angelegt, in dem die Tests mit JUnit durchgeführt werden. Diese Struktur ermoglicht eine problemlose Expandierung der Tests mit wachsendem Projektumfang.**

4. Besondere Probleme

4.1. **Ein Problem entstand bei der Implementierung der ImageViewer-Klasse, zunachst war es vorgesehen, das Bild auch auf ein Canvas zu zeichnen. Allerdings war es bis zum Abgabezeitpunkt nicht mglich, die Größen verschiedener Bilder bei jeder getesteten Datei korrekt auf das Canvas zu bertragen.**

5. Zeitbedarf

5.1. **Der Zeitbedarf wurde bei allen Teammitgliedern mit jeweils 10-15 Stunden angegeben. Dabei wurde viel Zeit damit verwendet, sich auf das GIT-System einzuarbeiten und die grundliegenden Komponenten der Bilddarstellung nachzuvollziehen.**