1. **Aufgabenblatt** zur Veranstaltung *Computergrafik und Visualisierung II* im Sommersemester 2020

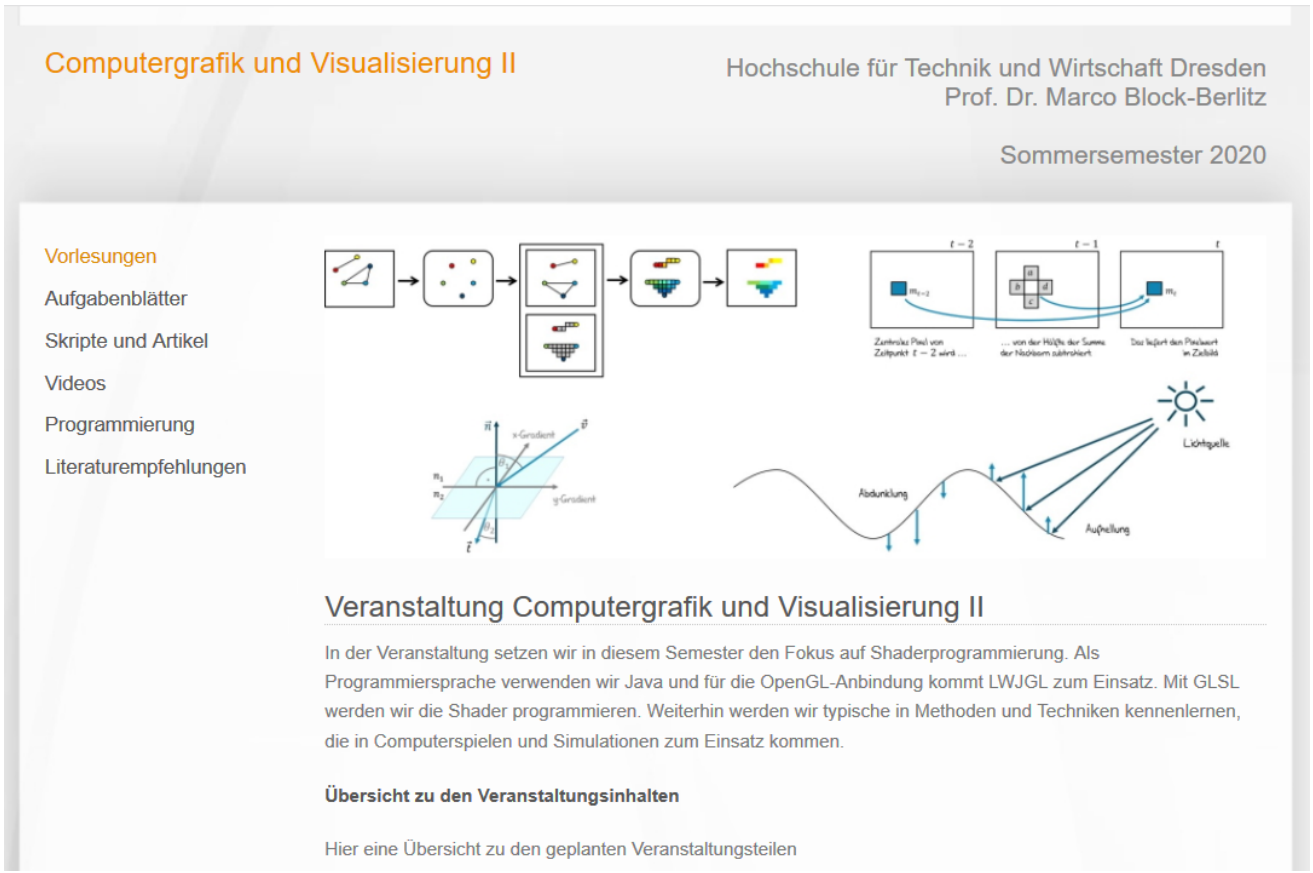
Bearbeitungszeitraum: 17.03.-30.03.2020

Hochschule für Technik und Wirtschaft Dresden, Friedrich-List-Platz 1, 01069 Dresden
Prof. Dr. Marco Block-Berlitz, Rainer Uhlemann

Im ersten Teil der Veranstaltung wollen wir uns noch einmal mit der **Linearen Algebra** auseinandersetzen und drei wichtige Klassen entwickeln, die wir bspw. für physikalische Simulationen in einem späteren Abschnitt benötigen werden. Daher ist es notwendig, diesen Abschnitt vollständig zu bearbeiten. Glücklicherweise können wir hier aus den Ergebnissen der Veranstaltung Programmierung III profitieren.

Das Aufgabenblatt ist in drei Teile gegliedert:

1. *Einarbeitung in die Grundlagen*: Im Selbststudium sollen hier die notwendigen theoretische Abschnitte bearbeitet werden. Dazu gibt es Literaturempfehlungen und Linksammlungen. Die Zusammenfassung der Materialien und weiterführenden Quellen werden neben Programmbeispielen auf der Veranstaltungswebseite zu finden sein. Diese finden Sie auf der Dozentenwebseite[1] in der Rubrik *Lehre*.



2. *Übungsaufgaben*: Die Übungsaufgaben haben den Zweck, das erworbene Wissen aus den Grundlagen zu prüfen und zu festigen. Weiterhin dienen Sie als Vorbereitung für die Klausur am Ende des Semesters. Die Lösungen zu den Übungsaufgaben sollen am Ende der Bearbeitungszeit per Email an beide Dozenten geschickt werden. Dazu werden wir Lyx oder Latex verwenden. Die Veranstaltung wurde bei Opal angelegt. Dort ist unter anderem ein Beispieldokument für die Abgabe in Lyx zu finden. Die Lösungen werden dann bewertet und es gibt eine individuelle Rückmeldung.

3. *Praktischer Teil*: Im praktischen Teil werden wir die theoretischen Grundlagen in Programmcode überführen und uns Schritt für Schritt in Richtung Shaderprogrammierung, Spieleentwicklung, Wegeplanung, Steuerungsverhalten und physikalischen Simulationen in der Computergrafik bewegen. Hier sind Sorgfalt und Übersicht bei der Programmentwicklung gefragt. Auch hier hilft uns das Wissen

[1] http://www.marco-block.de/

aus Programmierung III. Für den praktischen Teil gibt es Punkte. Die erreichten Punkte aller Aufgabenblätter ergeben am Ende die Belegnote. Die Dokumentation der Lösungen mit Erläuterungen zu den Codepassagen wird ebenfalls in Latex bzw. Lyx erwartet. Senden Sie die Lösungen ebenfalls an beide Dozenten per Email. Die Lösungen werden bewertet und es gibt eine individuelle Rückmeldung.
Hinweis: Die Lösungen der Übungsaufgaben und die Lösungen zum praktischen Teil sind spätestens am **30.03.2020** einzureichen.

1. Einarbeitung in die Grundlagen

Lesen und verstehen Sie die Inhalte zur Linearen Algebra von Kapitel 4 [1]. Hinweis: Falls Sie sich das Buch bisher noch nicht beschaffen konnten, finden Sie Kapitel 4 als Probekapitel auf der Buchwebseite[2]. Für die Bearbeitung der weiteren Kapitel in den folgenden Aufgabenblättern ist ein zeitnahe Beschaffung allerdings notwendig.

2. Übungsaufgaben

Hier orientieren wir uns ebenfalls in Kapitel 4 [1].

1. Warum notieren und behandeln wir Vektoren, die Farben repräsentieren, anders als Vektoren, die beispielsweise Richtungen oder Positionen repräsentieren?
2. Zeigen Sie anschaulich, dass $\vec{w} + \vec{v} = \vec{v} + \vec{w}$, also die Kommutativität für die Vektoraddition, gilt.
3. Berechnen Sie die Skalarprodukte $\vec{u} \bullet \vec{v}$, $\vec{u} \bullet \vec{w}$, $\vec{v} \bullet \vec{w}$ und $\vec{w} \bullet \vec{v}$ mit $\vec{u} = (-0.6, 0.8)$, $\vec{v} = (3, 4)$ und $\vec{w} = (4, 3)$.
4. Zeigen Sie, dass das Kreuzprodukt $\vec{v} \times \vec{w}$ zweier 3-dimensionaler Vektoren $\vec{v}$ und $\vec{w}$ einen Vektor liefert, der senkrecht zu $\vec{w}$ steht.
5. Beweisen Sie folgendes Theorem: Gegeben seien zwei 3-dimensionale Vektoren $\overrightarrow{v}$ und $\overrightarrow{w}$, dann erfüllt das Kreuzprodukt die folgende Gleichung:

   $$\|\vec{v} \times \vec{w}\| = \|\vec{v}\| \, \|\vec{w}\| \, sin\, \alpha$$

   Hinweis: Beginnen Sie zunächst damit, die linke Seite zu quadrieren und geeignet umzuformen. Es sollte folgender Term entstehen: $\|\vec{v}\|^2 \, \|\vec{w}\|^2 - (\vec{v}^T \vec{w})^2$. Die Auflösung des Kreuzprodukts zweier 3-dimensionaler Vektoren im rechtshändigen Koordinatensystem war ja gerade:

   $$\vec{a} \times \vec{b} = \begin{bmatrix} a_2 b_3 - a_3 b_2 \\ a_3 b_1 - a_1 b_3 \\ a_1 b_2 - a_2 b_1 \end{bmatrix}$$

   Anschließend können wir die Kosinusformel einsetzen und den cos-Term geschickt in sin überführen, da laut Pythagoras auch gilt $1 = cos^2\alpha + sin^2\alpha$.
6. Warum hätte die japanische Halb-Schwester des (auf Hawaii lebenden) trigonometrischen Gottes der Winkel „Sohcahtoa“, den folgenden Namen erhalten: „Cotaosechacosho“?

3. Praktischer Teil

1. **Vektor2D und Vektor3D** [10 Punkte]
   Erstellen Sie die zwei Klassen `Vektor2D` und `Vektor3D`, die entsprechend 2D- oder 3D-Vektoren repräsentieren können. Die Koordinaten werden jeweils durch `double` repräsentiert. Weiterhin sollen sinnvolle Hilfsmethoden angeboten werden. Es sind mindestens die folgenden Funktionen zu implementieren: `setPosition`, `isNullVector`, `add`, `sub`, `mult`, `div`, `isEqual`, `isNotEqual`, `length` und `normalize`. Dokumentieren Sie Ihre Entwicklung der Methoden `add` und `normalize` und mittels Test-Driven-Development in geeigneter Weise.
2. **Lineare Algebra** [20 Punkte]
   Schreiben Sie eine Klasse `LineareAlgebra`, die folgende statische Methoden (für 2D und 3D) zur Verfügung stellt: `add`, `sub`, `mult`, `div`, `isEqual`, `isNotEqual`, `length`, `normalize`, `euklDistance`, `manhattanDistance`, `crossProduct`, `dotProduct`, `cosEquation`, `sinEquation`, `angleRad`, `angleDegree`, `radToDegree`, `degreeToRad`, `determinante`, `abs` und `show`. Hinweis: Da die Klasse nur statische Methoden enthält, sollten sie die Einschränkung der Konstruktoren beachten.

Literatur

[1] Block-Berlitz, M.: *„Warum sich der Dino furchtbar erschreckte: Lehrbuch zu Beleuchtung und Rendering mit Java, LWJGL, OpenGL und GLSL“*, vividus Wissenschaftsverlag, 2019

[2] http://www.vividus-verlag.de/beleuchtung_und_rendering/index.html