Übungsblatt 3

Willkommen zur dritten Übung der Veranstaltung *Generative Computergrafik* im Sommersemester 2023. Ziel dieses Übungsblatts ist, dass Sie sich intensiver mit *Raytracings* vertraut machen.

Dieses Blatt enthält die erste von drei verpflichtenden Abgaben. Es ist bis Montag, den 15. Mai 2023 um 23:55 Uhr über `https://read.mi.hs-rm.de` abzugeben. Spätere Abgaben oder Abgaben auf anderem Wege, wie z.B. per Email können leider nicht berücksichtigt werden und müssen mit 0 Punkten bewertet werden. Sie sollten daher **frühzeitig mit der Arbeit an diesem Blatt beginnen** und evtl. auch schon früh erste lauffähige Versionen ihrer Lösung hochladen.

> Um das Praktikum zu bestehen müssen Sie *insgesamt (über alle drei Abgaben) mindestens 50% aller erreichbaren Punkte* erreichen, wobei *bei jeder der drei Abgaben mindestens 25% der erreichbaren Punkte* erreicht werden müssen.

Aufgabe 1. Implementieren Sie einen *interaktiven Raytracer in Python*. Verwenden Sie als *Startpunkt* die Implementierung, die hier `https://www.excamera.com/sphinx/article-ray.html` beschrieben ist. Den vollständigen Quellcode finden Sie unter `https://github.com/jamesbowman/raytrace/blob/master/rt3.py`.

Ihr Programm soll dabei Bilder, wie in Abbildung 1 dargestellt, erzeugen können. Die Bilder sollen dabei nicht (nur), wie in Übungsblatt 1 in eine Datei gespeichert werden, sondern in einer GUI angezeigt werden. Erweitern Sie hierzu das Beispielprogramm `raytracerTemplate.py` geeignet, welches Sie zusammen mit der Hilfsdatei `rendering.py` auf der Webseite zur Veranstaltung *Generative Computergrafik* finden.

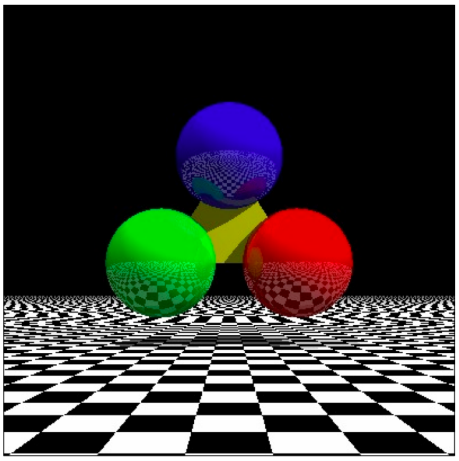

Abbildung 1: Raytracing einer Szene, die, wie die Szene von Übungsblatt 1, aus einer texturierten Ebene, drei Kugeln mit unterschiedlichen Materialeigenschaften, einem Dreieck und einer texturierter Grundebene besteht. Der Raytracer schießt dabei nicht nur Primärstrahlen, sondern rekursiv auch Sekundärstrahlen und kann dadurch Schatten und Spiegelungen darstellen. Die Szene wird von einer Lichtquelle beleuchtet. Als Beleuchtungsmodell wurde das Modell von Phong verwendet.

Konkret soll der von Ihnen realisierte Raytracer die folgende Funktionaliät besitzen:

1. (5 Punkte) Darstellung einer Szene, die aus drei Kugeln mit unterschiedlichen Materialeigenenschaften und einer mit einem Schachbrettmuster versehenen Ebene besteht in der GUI, wie sie in `raytracerTemplate.py` vorgegeben ist. Bei der dargestellten Szene soll es sich dabei nicht um die Szene aus `https://www.excamera.com/sphinx/article-ray.html` handeln. Vielmehr soll die dargestellte Szene so aussehen, wie die Szene in Abbildung 1 nur ohne das Dreieck zwischen den drei Kugeln.

   Hinweis: Im Wesentlichen ist hier der Code aus dem Git-Repository `https://github.com/jamesbowman/raytrace/blob/master/rt3.py` so anzupassen, dass keine Bild-Datei geschrieben wird, sondern das Ergebnisbild in der GUI angezeigt wird. Der Raytracer soll dabei auch geeignet auf entsprechende Änderungen der Fenstergröße reagieren.

2. (5 Punkte) Erweiterung der Anwendung, so dass die darstellte Szene interaktive um die Rotations-Achse, die durch den Szenenschwerpunkt und den up-Vektor der Kamera definiert ist, gedreht werden kann.

   Hinweis: Sie können es sich hier gerne auch einfach machen und als up-Vektor den Vektor $(0, 0, 1)^\top$ (die z-Achse) festlegen. Gedreht werden soll dabei bei Druck auf den Button `Rotate -` (Keyboard: '`n`') in mathematisch negativem Sinn und bei Druck auf den Button `Rotate +` (Keyboard: '`p`') in mathematisch positivem Sinn. Drehen Sie dabei jeweils um einen fest vorgegeben Winkel α (z.B. $\alpha = \pi/10$)

3. (10 Punkte) Erweiterung der Anwendung, so dass auch Dreiecke dargestellt werden können. Ihr Raytracer sollte nun also in der Lage sein, die in Abbildung 1 zu sehende Szene vollständig darzustellen. Weiterhin muss es auch hier wieder möglich sein, diese Szene durch Druck auf die Tasten '`n`' und '`p`' interaktiv zu drehen.