Die Lösungen für diese Übung sind abzugeben bis Sonntag den 12.05.2023 um 18:00h.

Raytracing

In dieser Übung werden wir Raymarching und Raytracing implementieren um sowohl implizite, als auch explizite Geometrie zu rendern.

Praktischer Aufgabenteil (7 Punkte)

Diese Woche bezieht sich die Lektion direkt auf das Ünbungsblatt. Lest diese Lektion bitte ausführlich durch und versucht den Code mit Hilfe der Beschreibung zu verstehen.

Es gibt 2 Teilaufgaben:

Raymarching (4 Punkte)

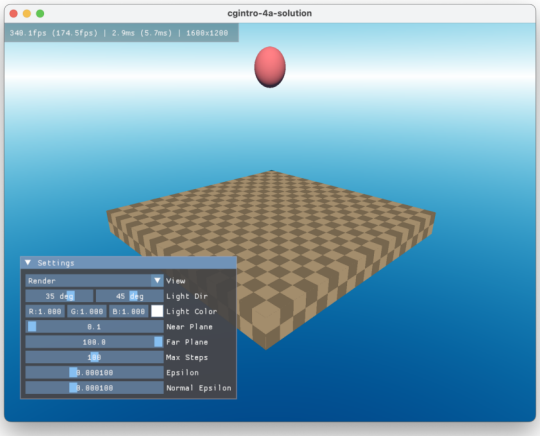

Abbildung 1: So sollte euer Ergebnis aussehen

Entpackt `cgintro-4a.zip` und implementiert folgende Funktionen in `raymarch.frag`:

a. Implementiert den Raymarching-Algorithmus, der in der Lektion beschrieben wird, in der Funktion `raymarchScene` *(2 Punkte)*.
b. Implementiert die Berechnung der Normalen in der Funktion `normalScene`. Die Normale ist der normalisiert Gradient des SDF, ein effizientes numerisches Verfahren zur Bestimmung des Gradienten findet Ihr in diesem Artikel von Inigo Quilez *(1 Punkt)*.

c. Nun solltet ihr eine Szene mit einem animierten Ball auf einer karierten Ebene sehen. Implementiert ein weiteres Objekt eurer Wahl, indem ihr `sdScene` und `colorScene` erweitert. SDFs dafür findet Ihr in diesem Artikel von Inigo Quilez *(2 Punkte)*.

Raytracing (2 Punkte)

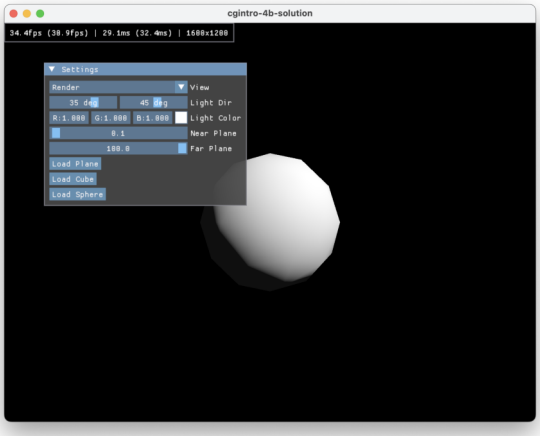

Abbildung 2: So sollte euer Ergebnis aussehen

Entpackt `cgintro-4b.zip` und implementiert den Strahl-Dreieck-Schnittpunkt-Test in `raytrace.frag`. Schaut euch dafür die Folie 18 aus Vorlesung `cgintro-04-Raytracing` an. Wenn kein Schnittpunkt vorhanden ist soll die Funktion `vec3(Inf)` zurückgeben, ist jedoch ein Schnittpunkt vorhanden, soll die Funktion `vec3(u, v, t)` zurückgeben, wobei t die Entfernung des Dreiecks auf dem Strahl ist und u, v und $w = 1 - u - v$ die baryzentrischen Koordinaten des Schnittpunktes sind.

Es sei $e_1 = v_1 - v_0, e_2 = v_1 - v_0, n = e_1 \times e_2$. Um u, v zu ermitteln können wir nun folgendes lineares Gleichungssystem lösen:

$$\left(\begin{array}{c|c|c} e_1 & e_2 & n \end{array} \right) \begin{pmatrix} u \\ v \\ x \end{pmatrix} = p = r(t) - v_0$$

Uns interessieren nur die Lösungen für u und v. Dafür können wir die Cramersche Regel anwenden und erhalten:

$$u = \frac{\det \left(\begin{array}{c|c|c} p & e_2 & n \end{array} \right)}{\det \left(\begin{array}{c|c|c} e_1 & e_2 & n \end{array} \right)}, \quad v = \frac{\det \left(\begin{array}{c|c|c} e_1 & p & n \end{array} \right)}{\det \left(\begin{array}{c|c|c} e_1 & e_2 & n \end{array} \right)}$$

Wir nutzen jetzt das sogenannte Spatprodukt, um die Determinate einer 3×3-Matrix zu bestimmen:

$$\det \left(\begin{array}{c|c|c} a & b & c \end{array} \right) = (a \times b) \cdot c$$

Damit ergeben sich nun folgende Formeln für unsere Determinaten:

$$\begin{aligned} \det \left(\begin{array}{c|c|c} e_1 & e_2 & n \end{array} \right) &= (e_1 \times e_2) \cdot n = n \cdot n, \\ \det \left(\begin{array}{c|c|c} p & e_2 & n \end{array} \right) &= (p \times e_2) \cdot n, \\ \det \left(\begin{array}{c|c|c} e_1 & p & n \end{array} \right) &= (e_1 \times p) \cdot n \end{aligned}$$

Theoretischer Aufgabenteil (3 Punkte)

Teilaufgabe 1

Gebet jeweils Quaternionen für folgende Rotationen an:

a. Eine Rotation um $80°$ um die z-Achse *(1 Punkt)*.
b. Eine Rotation um $120°$ um die Achse $\begin{pmatrix} 1 & 1 & 1 \end{pmatrix}^T$ *(1 Punkt)*.

Der Winkel der Rotation wird dabei als im mathematisch positiven Sinn angenommen.

Teilaufgabe 2

Wir haben in den Übungen bereits die Modell-, Kamera- und Projektionsmatrizen besprochen. Nach Anwendung dieser (und dem *Clipping*) liegen alle zu zeichnenden Vertices im Würfel $[-1, 1]^3 \subset \mathbb{R}^3$. Allerdings müssen die Positionen der Vertices noch auf *Screen-Koordinaten* abgebildet werden. Dieser Prozess wird als *Screen-Mapping* bezeichnet. Für diese Transformation muss neben der Größe des normierten Sicht-Volumens auch die Größe des Fensters auf dem Bildschirm bekannt sein. Mit diesen Informationen lässt sich eine Matrix ableiten, welche die Positionen in normierten Koordinaten auf Screen-Koordinaten transformiert.

a. Leiten Sie die Transformationsmatrix für den Screen-Mapping-Prozess unter Verwendung einer Viewport-Größe von 1024×768 her, bildet abei die z-Koordinate auf $[0, 1]$ ab *(1 Punkt)*.