**Kapitel 9 Rendering**

**0901 Einleitung**

Rendern stammt vom englischen Wort „to render“ und heißt zu deutsch „etwas ausgeben“. Das Rendering bezeichnet den Vorgang, ein Bild zu generieren. Beim Rendern aus einer 3D-Szene werden insbesondere Meshes, Kameras und Lichtquellen berücksichtigt.

In der Rendering-Pipeline durchläuft ein Mesh mehrere Schritte, um am Ende als ein rasterisiertes Bild dargestellt werden zu können. Hierbei sind die wichtigsten Stationen die Umwandlung der Koordinaten des Meshes und die   
Rasterisierung.

Des Weiteren werden aber auch Sichtbarkeits- und Beleuchtungsberechnungen durchgeführt, Texturen gemappt und spezielle Effekte dargestellt.

**0902 Modell-Transformation**

Ein Objekt, dass sich in einem dreidimensionalen Raum aufhält, wird normalerweise durch sein lokales Koordinatensytem definiert. Ohne Transformationen würden alle Objekte im Weltursprung liegen.

Durch Translationen, Rotationen und Skalierungen wird ein Mesh an die gewünschte Stelle im Raum gebracht.

Die endgültige Position eines Objektes hängt von der Reihenfolge der Transformationen in der Szenenhirarchie ab.

Um diese Transformationsverkettung aufzulösen werden die Objekte in ein Weltkoordinatensystem übertragen.

Von nun an liegen die Koordinaten nicht mehr in lokalen Koordinaten, sondern in Weltkoordinaten vor.

**0903 Kamera-Transformation**

Bei der Kamera-Transformation, die auf Englisch viewing transformation genannt wird, werden zunächst die Betrachtungsparameter bestimmt.

Das heißt es wird die Betrachterposition, die Blickrichtung und das Sichtvolumen der Kamera bestimmt.

Beim Sichtvolumen oder auch View Frustrum handelt es sich um den Bereich, der von der Kamera eingefangen wird.

Nach dem Setzen der Kamera wird die Szene derartig transformiert, dass die Betrachterposition im Ursprung liegt. Dabei ist die Ausrichtung der Kamera entlang der z-Achse.

Nach diesem Schritt liegen die Weltkoordinaten des Meshes nun als Kamerakoordinaten vor.

**0904 Projektions-Transformation**

Nach der Modell- und der Kamera Transformation befinden sich alle Eckpunkte, welche auch Vertices genannt werden, an den gewünschten Positionen.

Bei der Projektions-Transformation wird das View Frustrum näher betrachtet. Das Frustrum hat die Form eines Pyramidenstumpfes und besteht aus einer Far Clipping Plane und einer Near Clipping Plane, die auch Projektionsebene genannt wird.

Die Meshes die dargestellt werden sollen liegen innerhalb des Frustrums.

Um eine orthografische Projektion zu ermöglichen, muss die Near Clipping Plane derartig transformiert werden, dass ein Einheitswürfel entsteht.

Daraufhin werden die Vertices orthogonal auf die Projektionsebene projiziert.

Nach diesem Schritt liegen die Koordinaten als Clip-Koordinaten vor.

**0905 Clipping**

Beim Clipping geht es darum Flächen, die vom sichtbaren Volumen nicht mehr eingefangen werden können

aus der Szene zu entfernen.

Nach der Projektionstransformation wird überprüft, welche Meshes vollständig im sichtbarem Bereich liegen.

Elemente die gänzlich außerhalb des Sichtfensters liegen werden komplett entfernt.

Für jede Kante des Sichtfensters wird geprüft, ob sich der Vertex eines Objekts inner- oder außerhalb der Kante befindet.

Punkte die innerhalb der Grenze liegen werden in ihrer Geometrie belassen, Punkte außerhalb entfernt.

An der Grenze des Sichtfensters werden neue Vertices kreiert. Dieses Verfahren wird auch Sutherland Hodgeman Clipping genannt.

**0906 Culling**

Beim Culling geht es darum Flächen, die vom Betrachter nicht mehr wahrgenommen werden können aus der Szene zu entfernen.

Durch das Backface-Culling werden die Polygone aus der Szene entfernt, die vom Betrachter abgewandt sind.

Ob eine Fläche sichtbar oder nicht sichtbar ist wird mit Hilfe des Normalenvektors entschieden.

Ein Normalenvektor ist ein Vektor einer Fläche, der zu diesem orthogonal ist.

Zeigt der Normalenvektor zum Beispiel in Richtung der Kamera, hat es zur Folge, dass der Betrachter die Vorderseite sieht. Ist der Normalenvektor n von der Kamera abgewandt heißt das, dass es sich bei der Fläche um eine Rückseite eines Meshes handelt.

Damit kann diese Fläche entfernt werden.

**0907 Rasterisierung**

Bis zu diesem Zeitpunkt liegt für jeden Vertex eines Polygons ein Farbwert vor. Um Das Bild jedoch später auf einem Monitor darstellen zu können, muss für jedes Pixel der Farbwert, durch Beleuchtungsberechnungen, berechnet werden.

Zur besseren Unterscheidbarkeit werden die Pixel in diesem Schritt Fragmente genannt.

Ein weiterer Grund für die Unterschiedliche Bezeichnung ist, dass ein Fragment mehrere Daten speichern kann.

Dazu gehört zum Beispiel der Alpha-Wert, der Transparenzen beschreibt, und der z-Wert, der für die Verdeckungsberechnung wichtig ist.

Bei diesem Schritt werden die Flächen in Fragmente aufgeteilt.

**0908 z-Buffer**

Bei einer Szene in der mehrere Objekte zu sehen sind, kann es dazu kommen, dass ein Objekt A vor einem Objekt B ist, oder dieses auch schneidet.

Um dies korrekt darzustellen, werden in der Computergrafik Verdeckungsberechnungen, wie z. B. der z-Buffer-Algorithmus angewandt.

Die Grundidee des z-Buffer-Algorithmuses ist es für jeden Pixel die Tiefeninformation bzw. den z-Wert zu speichern.

Es muss geprüft werden ob ein Pixel näher an der Kamera liegt als ein vorher berechneter. Dazu muss der z-Wert kleiner sein.

Falls ja, werden Farbwerte und z-Buffer für den Pixel überschrieben , andernfalls werden die alten Werte beibehalten.

**0909 Raytracing**

Beim realistischen Rendern liegt das Hauptaugenmerk auf der physikalischen Korrektheit der Darstellung des gerenderten Bildes.

Dafür sollte jedoch eine höhere Rechenzeiten in Kauf genommen werden.

Raytracing – zu Deutsch „Strahlen verfolgen“ – ist in erster Linie ein Algorithmus zu Verdeckungsberechnung.

Dies basiert auf dem Aussenden von Strahlen vom Betrachterblickpunkt aus. Abgesehen vom Betrachterblickpunkt ist eine Bildebene vorhanden, die in Pixel unterteilt ist und dem später gerenderten Rasterbild entspricht.

Für jeden Pixel wird ein Strahl ausgesandt. Diese werden Primärstrahl genannt.

Die Primärstrahlen ermitteln Schnittpunkte mit Primitiven.

Anschließend werden die gebrochenen bzw. die reflektierten Strahlen berechnet. Diese werden als Sekundärstrahlen bezeichnet.

Der Strahl endet, wenn er die maximale Anzahl von Schritten erreicht, auf kein weiteres Objekt oder auf eine Lichtquelle trifft.

Das Raytracing bringt den Vorteil, dass korrekte Objektspiegelungen und Schatten dargestellt werden können.

**0910 Raytracing – Interaktion**

Wähle mittels der Radio Button zwischen reinem Shading und Raytracing aus und betrachte die Änderungen.

**0911 Volumengrafik**

Volumengrafiken sind in der Lage transparente Objekte und Objekte ohne scharfe Abgrenzungen, wie z. B. Wolken, zu modellieren. Diese bestehen aus Voxeln. Voxel bezeichnet einen Gitterpunkt in einem dreidimensionalen Gitter. Dies entspricht einem Pixel in einem 2D-Bild, einer Rastergrafik.

Die Volumengrafik basiert auf dem Strahlentransport, derbeschreibt, wie sich Licht auf dem Weg durch ein Volumen verhält.

Beim Rendern einer Volumengrafik unterscheidet man vier Schritte:

1. der Klassifikation: Hier werden den Voxeln Materialeigenschaften gegeben.

Bei der Erzeugung des Voxels besitzt dieser zunäckhst nur eine Eigenschaft. Weitere müssen bei der Kalssifikation vom Benutzer vorgegeben werden.

Eine Eigenschaft könnte zum Beispiel sein, wie sehr das Voxel spiegeln soll.

2. der Interpolation: Da es sich bei Voxeln um Punkte handelt, ist es unwahrscheinlich, dass sie von einem Strahl getroffen werden. Deswegen werden die

Materialeigenschaften an Punkten zwischen den Voxeln aus benachbarten Voxeln angenähert.

3. dem Shading: Beim Shading wird bestimmt, wie viel Licht von einem Voxel aus in Richtung des Betrachters reflektiert wird und welche Farbe es hat.

4. der Composition: Beim durchqueren des Lichts durch Voxel ändert sich die Farbe und die Intensität. Bis der Lichtstrahl auf die Bildebene fällt, kann dieser mehrere Voxel durchqueren. Die letzten Eigenschaften des Strahles, färben den Pixel auf der Bildebene.