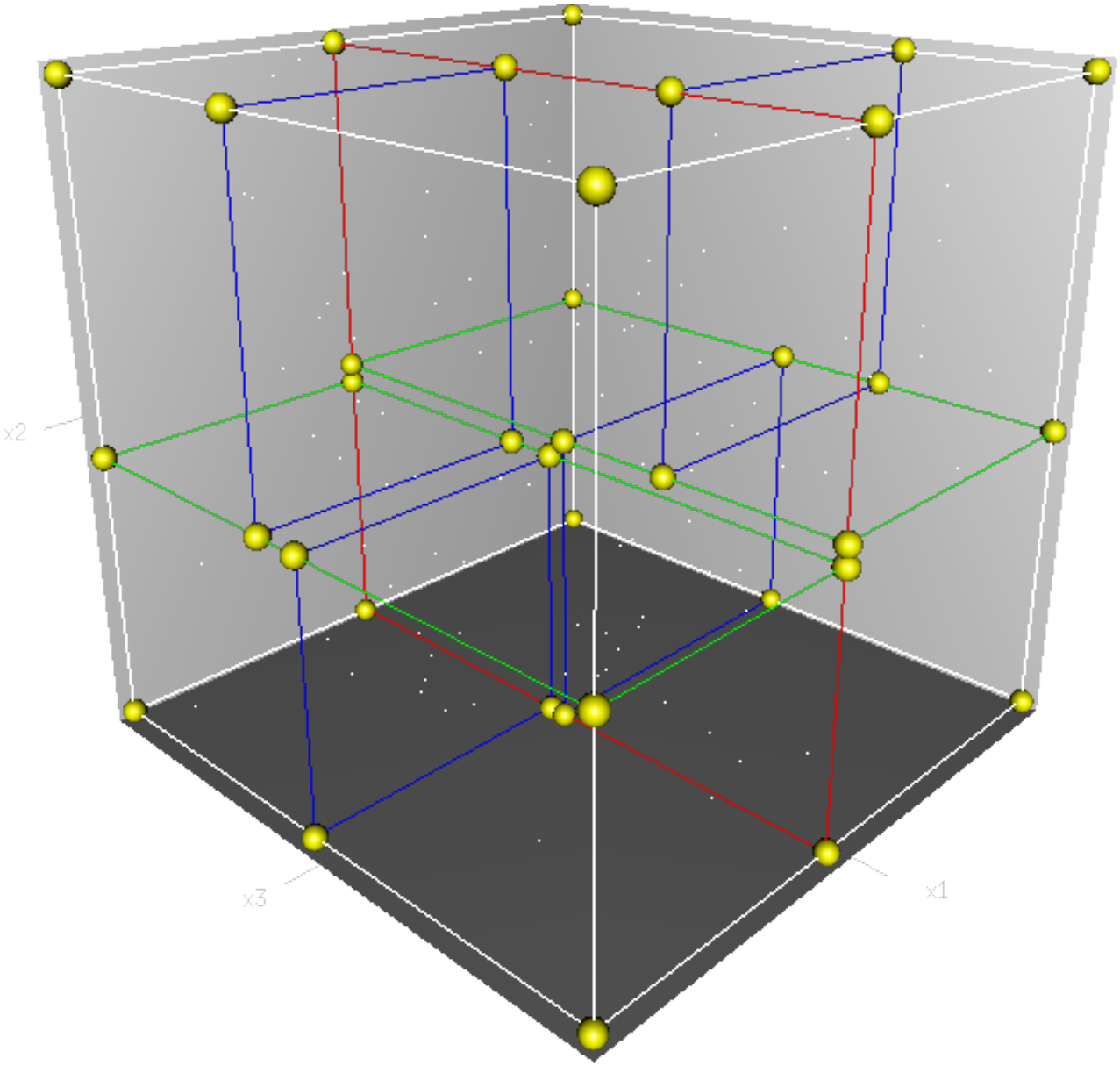

x2
x3
x1

Z-Buffer

Der Z-Buffer enthält für jedes Pixel (u, v) nach der Rasterung einen Wert zwischen -1 und 1. Dieser Wert ist der Abstand zum nächsten Punkt eines Polygons in Clipping-Koordinaten von diesem Pixel auf der Projektionsebene. Wir haben somit eine Funktion

$$Z - Buffer : \mathbb{N} \times \mathbb{N} \to [-1, 1] \, .$$

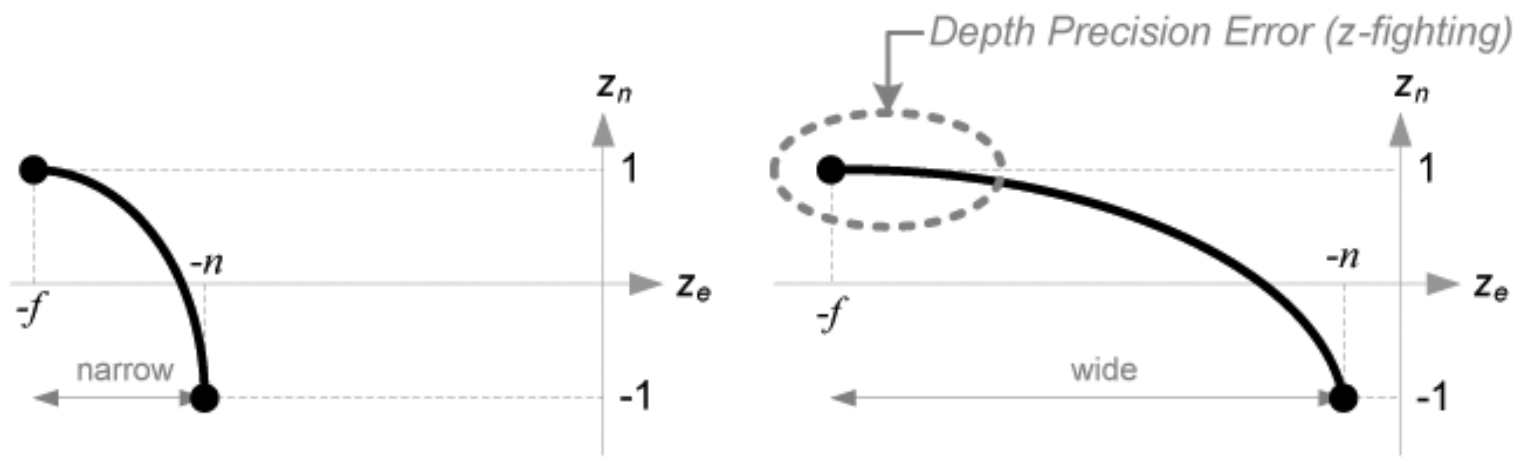
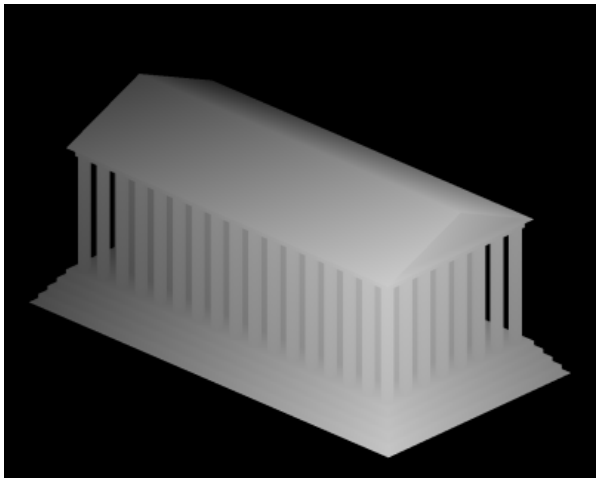


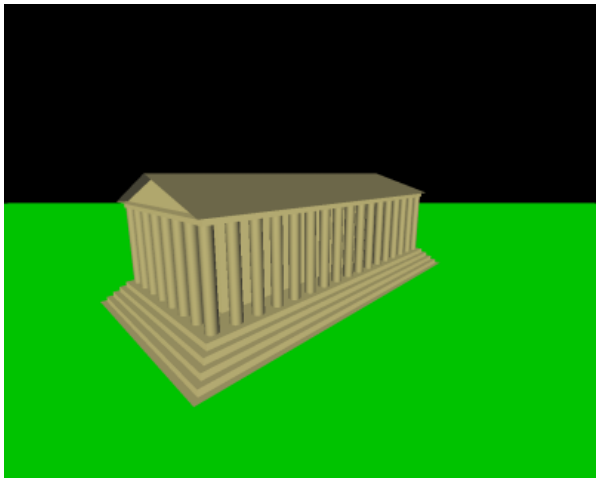
Figure: Z-Buffer-War

Shadowmap

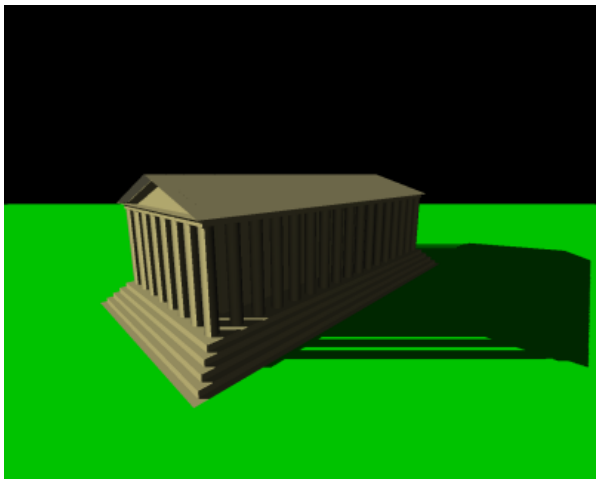
Das Rendern einer Szene mit Schatten unter Zuhilfenahme von einer Shadow Map geschieht im Wesentlichen in zwei Schritten. Als erstes wird die Szene aus der Sicht des Lichts gerendert und die Tiefeninformation (z-Buffer) in eine Textur gerendert. Anschließend wird die Szene normal gerendert, wobei für jedes Pixel der Abstand des zugehörigen Vertex zur Lichtquelle (in Clipping Coordinaten) mit dem Wert der Shadow Map verglichen wird. Ist der Wert in der Shadow Map kleiner, wird das Pixel schattiert gerendert.



(a) Tiefeninformation aus Sicht des Lichtes gerendert



(b) Rendering ohne Shadow mapt



(c) Rendering mit Shadow map

Texturemapping

Ein Texturemapping ist eine Abbildung $T : \mathbb{R}^2 \rightarrow \mathbb{R}^3$, die Bildpunkte den Vertices des Meshes zuordnet.

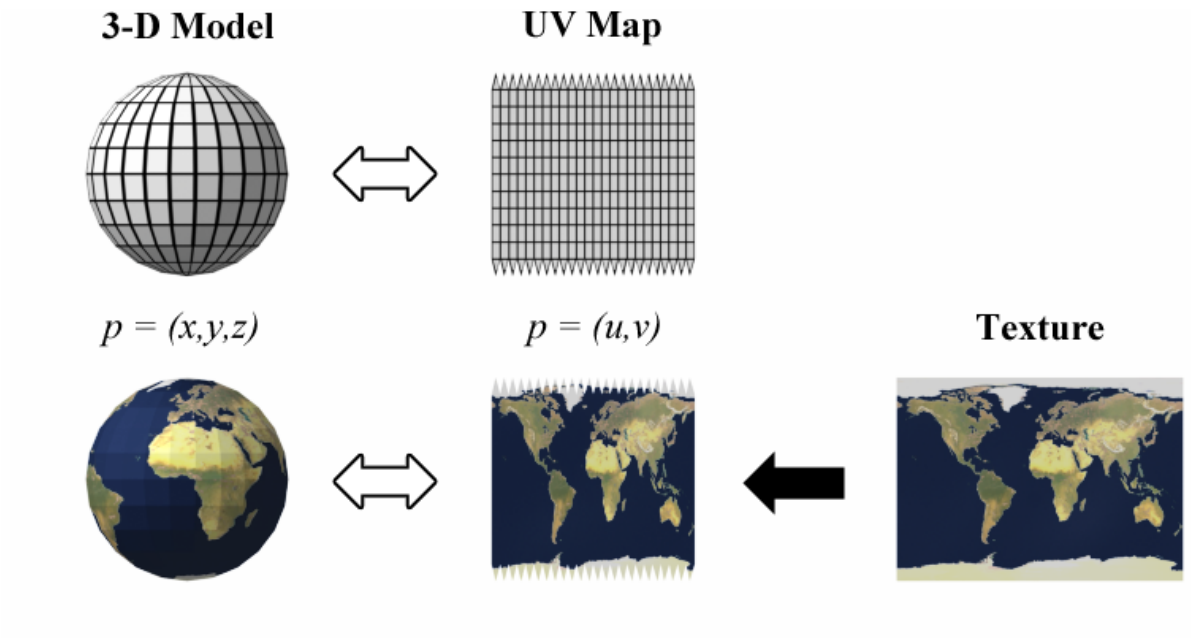


Figure: uv mapping

Figure: uv mapping

Bumpmapping

Die relativen Höheninformationen liegen in einer Textur (Hightmap) in Form von Graustufen vor – der sogenannten Heightmap. Jeder Grauwert steht für eine bestimmte Höhe. Für die Lichtberechnung werden Normalen berechnet, die einem Netz entprechen, welches entsteht, wenn man die Vertices entlang der ursprünglichen Normale um den Betrag in der Bumpmap verschiebt.

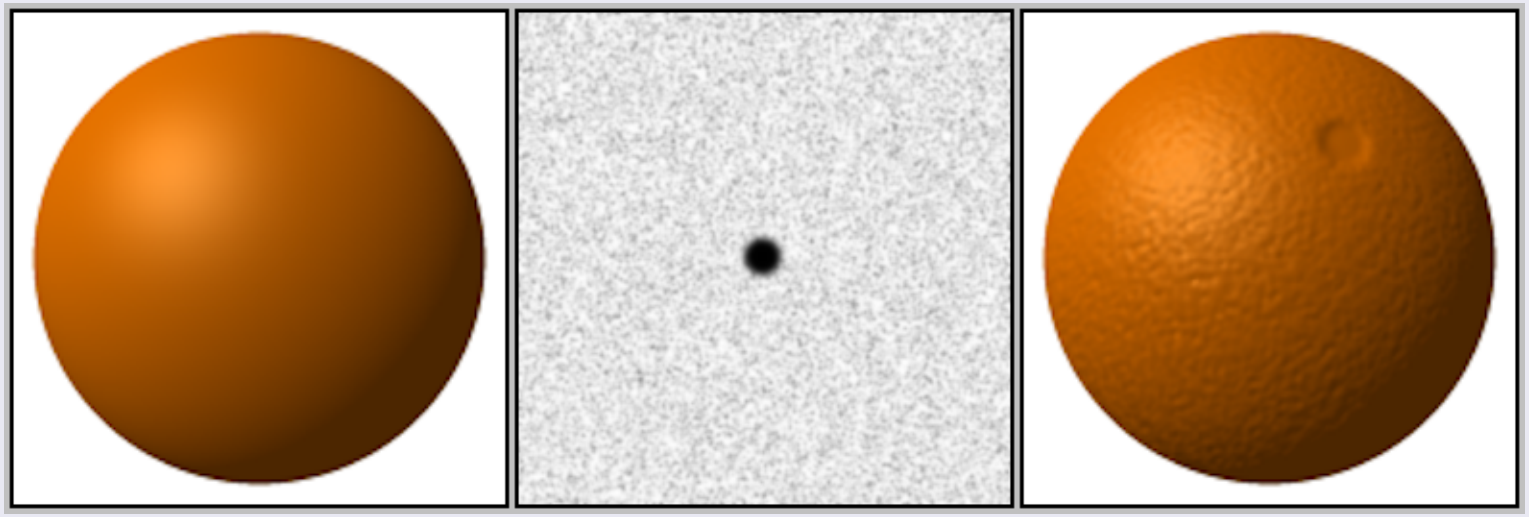


Figure: Bumpmapping

Bumpmapping

Wie beim Bumpmapping wird eine Heightmap als Textur mitgegeben. Im Gegensatz dazu werden die Punkte des Gitternetzes (Vertices) entsprechend diesen Texturinformationen entlang ihrer Normalen, das heißt senkrecht zur Oberfläche, tatsächlich verschoben.

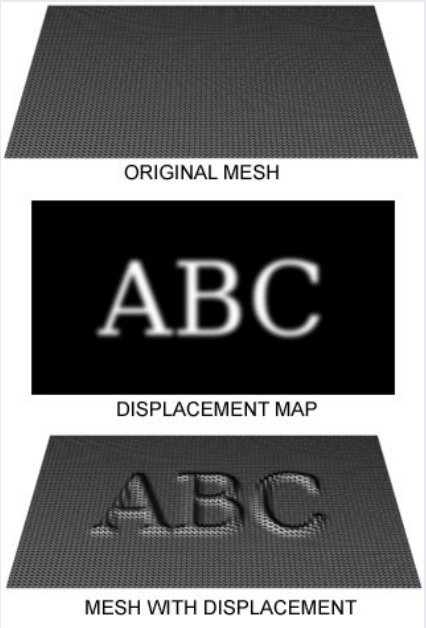


Figure: Displacementmapping

Perlin Noise

Perlin Noise ist ein Algorithmus zur Erzeugung zufälliger, organischer Strukturen. Die Entwicklung geht auf Ken Perlin zurück, der ihn 1982 für den Film Tron entwickelte, wofür er 1997 einen Oscar erhielt

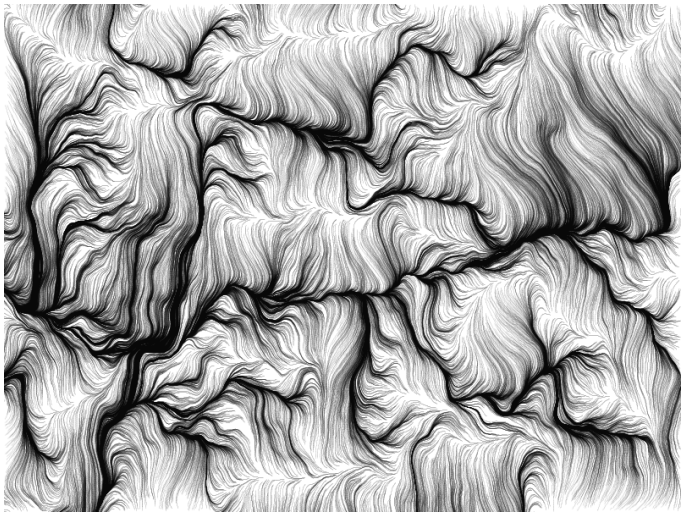


Figure: Anwenung Perlin Noise

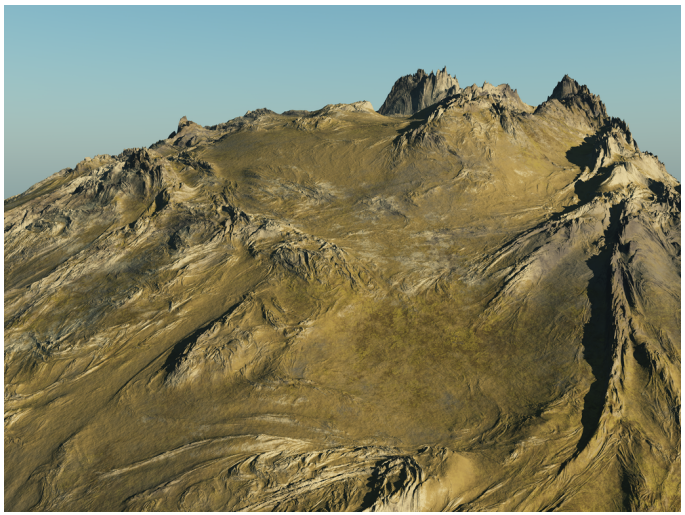


Figure: Anwendung Perlin Noise

Perlin Noise

Generiere Zufallsvektoren an Gitterpunkten.

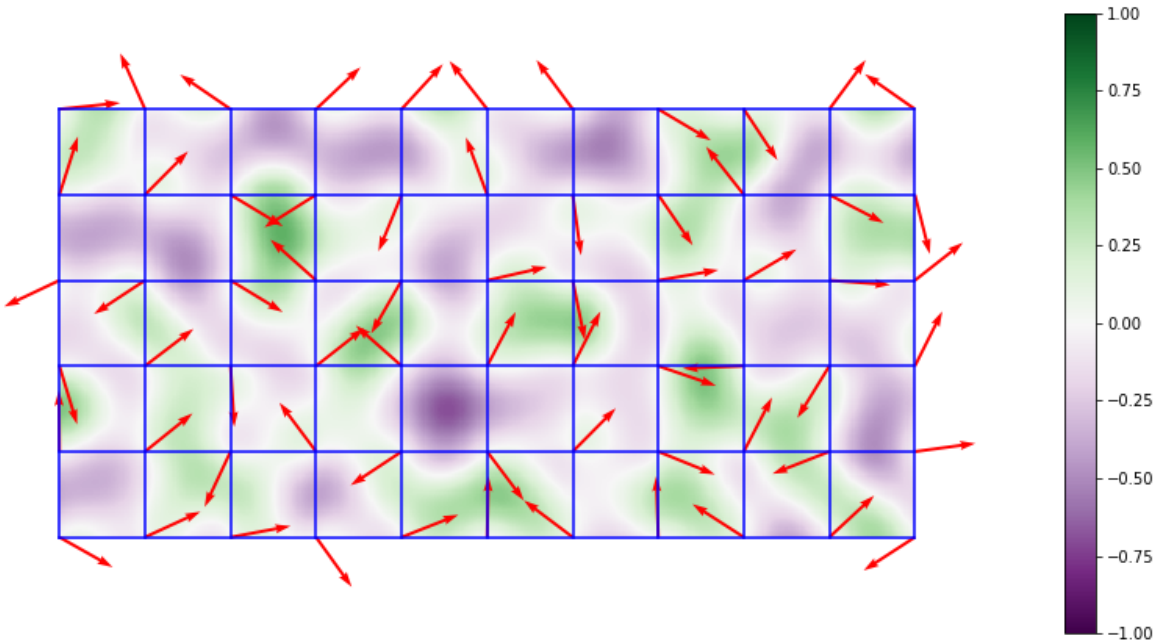


Figure: Zufallsvektoren

Perlin Noise

Berechne Skalarprodukte von Punkt mit Zufallsvektoren in zugehörigem Gittereckpunkten. Interpoliere zwischen diesen Werten.

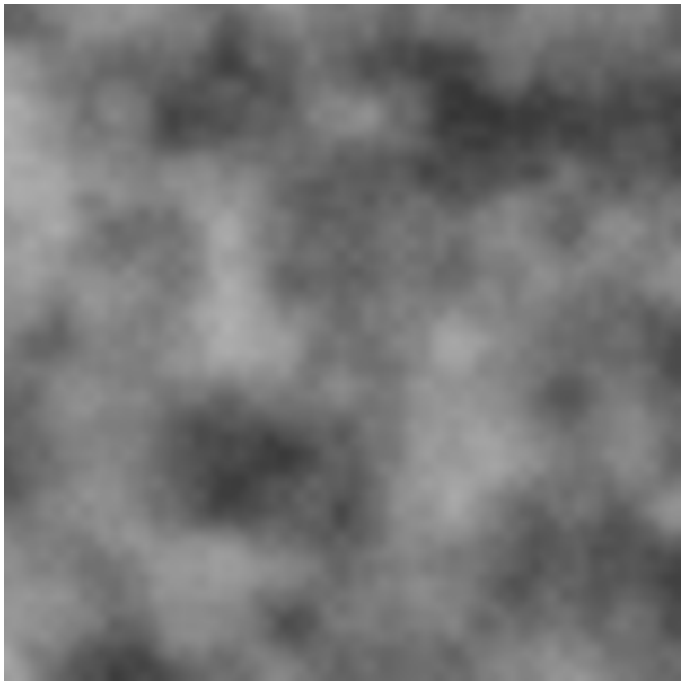


Figure: Perlin Noise Textur

Kd-Trees

Kd-Trees sind balancierte Binärbäume, mit deren Hilfe Bereichsabfragen effizient implementiert werden können.