

# **12. TEXTUREN | DREHBUCH**

## **COMPUTERGRAFIK.ONLINE**

Hochschule Furtwangen University | Fakultät Digitale Medien

Betreuer: Prof. Jirka Dell'Oro-Friedl | Projektstudium SoSe 18

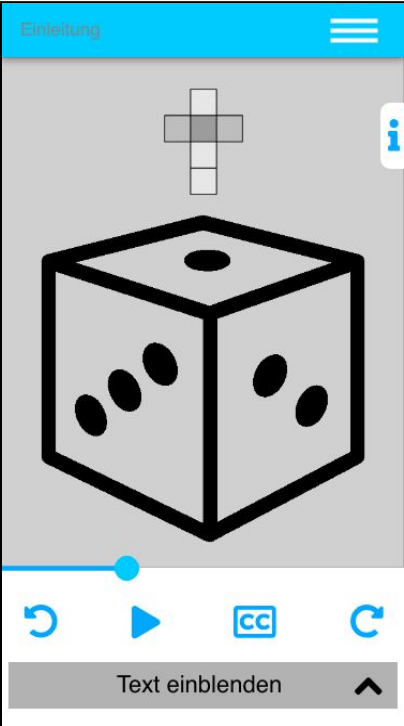
Version: 1.4 | Letzte Änderung: 06.12.2018

Autor: Lisa Würstle MKB 5

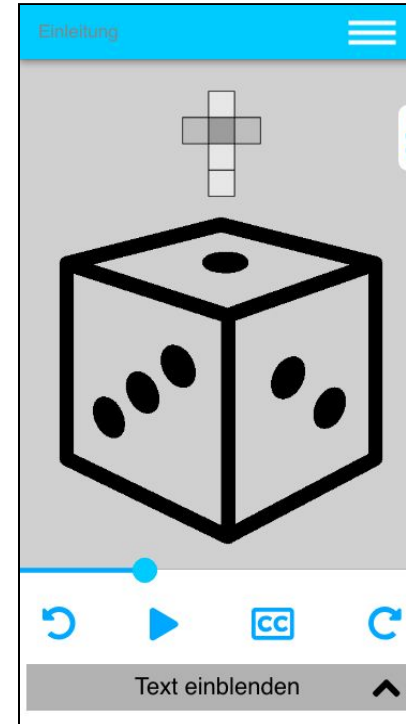
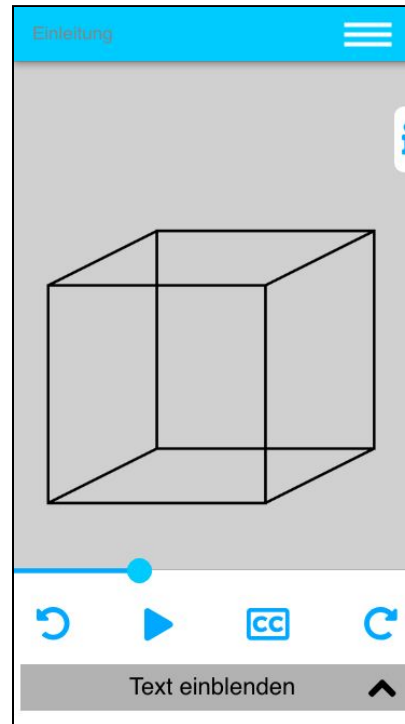
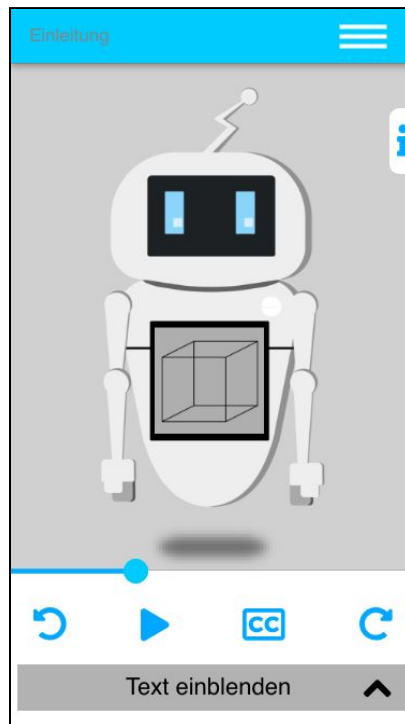
## Inhaltsverzeichnis

12.1	(A) Einleitung	2, 3
12.2	(I) Einleitung	4, 5
12.3	(A) Texturkoordinaten	6, 7
12.4	(A) UV-Mapping	8, 9
12.5	(I) UV-Mapping	10
12.6	(A) Mip-Mapping	11, 12
12.7	(A) Bump-Mapping	13
12.8	(A) Normal-Mapping	14
12.9	(A) Displacement-Mapping	15
12.10	(I) Vergleich	16, 17
12.11	(A) Environment-Mapping	18
12.12	(A) Kubisches Environment-Mapping	19, 20

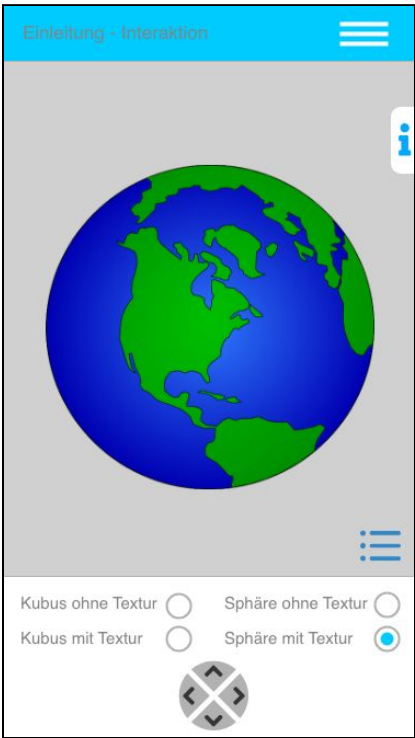
## 12.1 (A) Einleitung

Finaler Screen	Sprechertext	Screentext (i)	Regieanweisung
	<p>#120101 Eine Textur dient im Bereich der Computergrafik dazu Oberflächen von Objekten realistischer und detailreicher darzustellen.</p> <p>#120102 Die Textur besteht meistens aus einem zweidimensionalen Bild, welches dem 3D-Objekt eine Struktur verleiht. Wie die Textur auf das jeweilige 3D-Objekt projiziert wird, kann auf verschiedene Arten und Weisen geschehen. Diese Verfahren nennt man Mapping-Verfahren.</p> <p>#120103 Der Begriff Mapping bedeutet Zuordnung oder Abbildung.</p>	<ul style="list-style-type: none"> <li>- Ziel: Oberfläche eines Objekts realistischer wirken zu lassen</li> <li>- Lösung: 2D-Bild = 2D-Textur</li> <li>- verschiedene Mapping-Verfahren</li> <li>- Mapping = Zuordnung / Abbildung</li> </ul>	<p>#120101 Einblenden des Roboters Monitor: Einblenden eines Kubus ohne Texturierung (Fade in Monitor) Der Kubus fängt an sich um die eigene Achse zu drehen</p> <p>#120102 Eine Spielwürfeltextur erscheint auf dem Kubus (darüber erscheint die aufgeklappte Textur)</p>

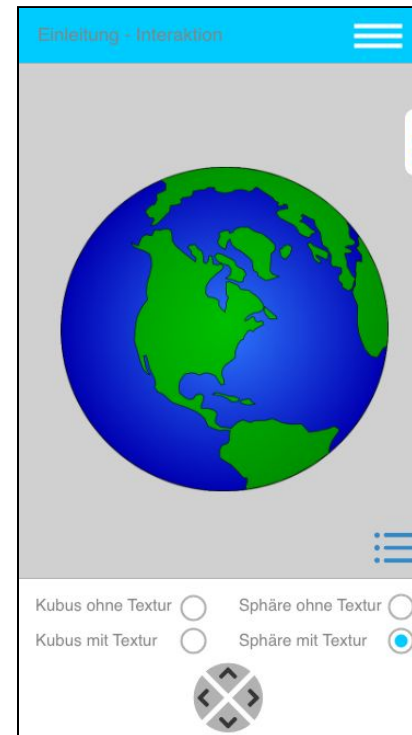
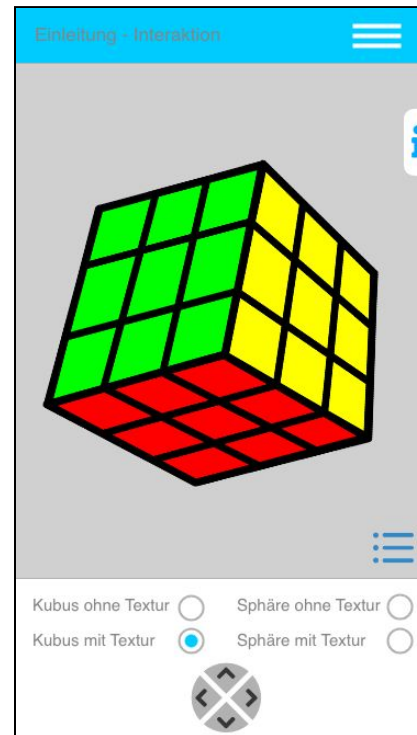
Grafikablauf:



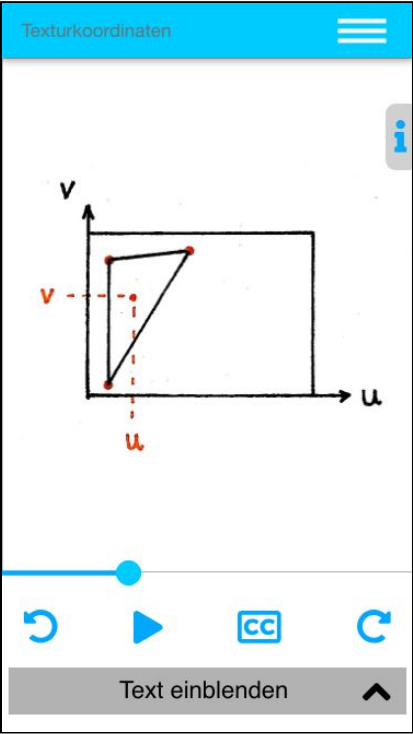
## 12.2 (I) Einleitung

Finaler Screen	Sprechertext	Screentext (i)	Regieanweisung
	<p>#120201 Wähle den Kubus oder die Sphäre aus. Suche dir anschließend eine Textur aus der vordefinierten Liste aus und lasse dir den Vergleich von dem 3D-Objekt ohne und mit Textur anzeigen. Betrachte das 3D-Objekt dabei von allen Seiten.</p>	<p>Wähle den Kubus oder die Sphäre aus. Suche dir anschließend eine Textur aus der vordefinierten Liste aus und lasse dir den Vergleich von dem 3D-Objekt ohne und mit Textur anzeigen. Betrachte das 3D-Objekt dabei von allen Seiten.</p>	<p>#120201 Aufgabe wird gesprochen (über das i kann sich der Nutzer die Aufgabe anzeigen lassen)</p> <p>Klickt der Nutzer auf das Listen-Symbol öffnet sich eine vordefinierte Liste in dem er sich eine Textur aussuchen kann</p> <p>Der Nutzer kann zwischen den Auswahlmöglichkeiten "Kubus ohne Textur", "Kubus mit Textur", "Sphäre ohne Textur" und "Sphäre mit Textur" auswählen, wobei die Punkte "mit Textur" nur ausgewählt werden können, wenn der Nutzer eine Textur ausgewählt hat</p> <p>3D-Objekt kann vom Nutzer beliebig über das Steuerkreuz gedreht und von allen Seiten betrachtet werden</p>

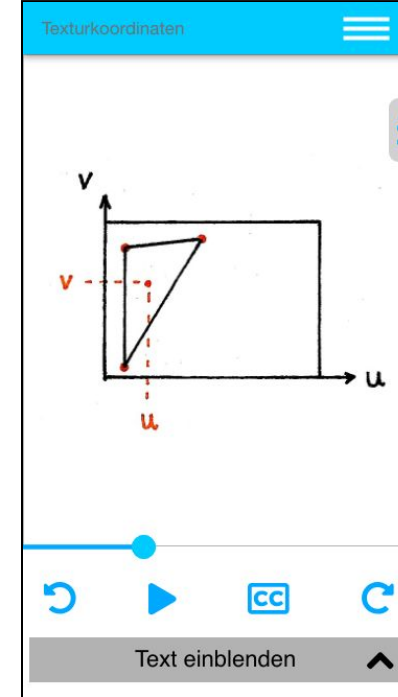
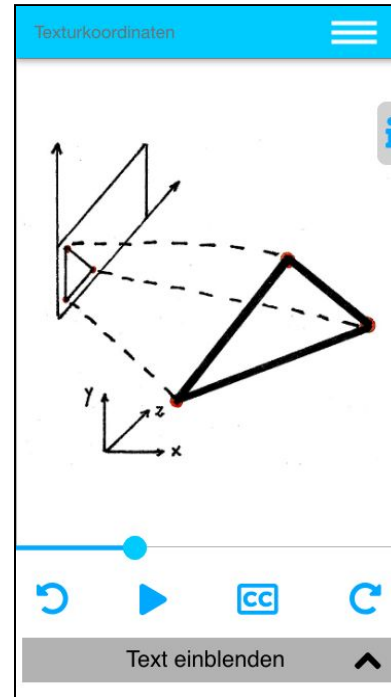
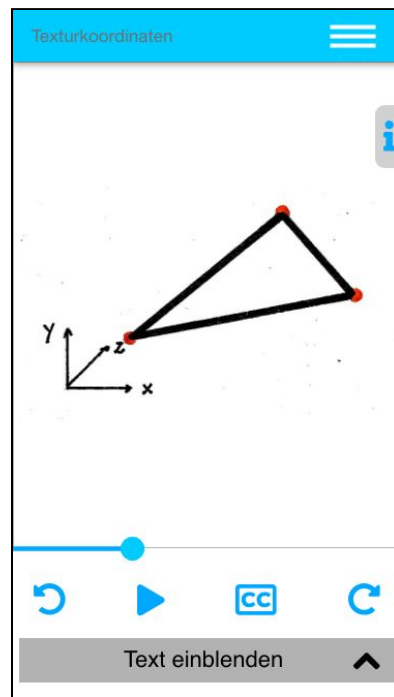
Grafikablauf:



### 12.3 (A) Texturkoordinaten

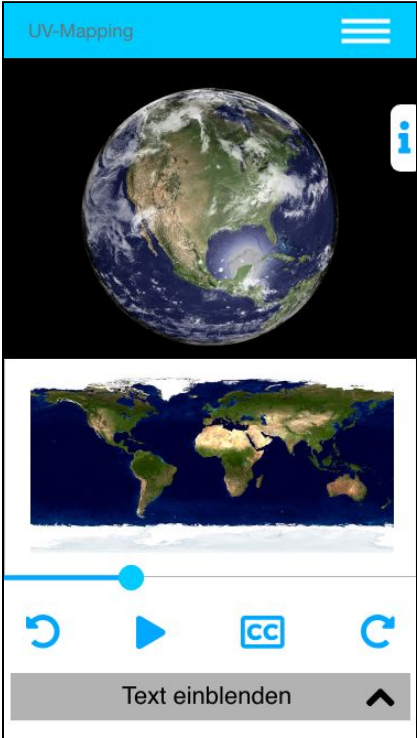
Finaler Screen	Sprechertext	Screentext (i)	Regieanweisung
	<p>#120301 Nun stellt sich die Frage, wie die Textur auf das Mesh projiziert wird. Zuerst wird die zu texturierende Fläche im 3D-Raum definiert.</p> <p>#120302 Anschließend ordnet der Benutzer die definierten 3D-Koordinaten mit Hilfe geeigneter Werkzeuge den Eckpunkten des Polygons zu. Die Textur besteht aus Pixeln, welche auch Texel genannt werden.</p> <p>#120303 Die Texturkoordinaten werden in einem kartesischen Koordinatensystem dargestellt. Die Achsen werden in der Regel mit u und v beschriftet um sie von x und y unterscheiden zu können. Der Wertebereich ist dabei jeweils von 0 bis 1.</p>	<ul style="list-style-type: none"> <li>- Umwandlung von Objektraum-Koordinaten (x, y, z) in Texturkoordinaten (u, v)</li> <li>- Zuordnung der Koordinaten mit Hilfe geeigneter Werkzeuge</li> <li>- Pixel auf der Textur = Texel</li> </ul>	<p>#120301 Einblenden eines Dreiecks im 3D-Raum</p> <p>Eckpunkte werden markiert</p> <p>#120302 Einblenden eines kartesischen Koordinatensystems in dem eine 2D-Textur liegt Dreieck wird in 2D-Raum projiziert (Zoom zum Koordinatensystem) und jeder Eckpunkt erhält eine genaue Position auf der Textur</p> <p>#10030103 Die Achsen werden mit u und v beschriftet Ein sichtbarer Pixel erhält eine genaue Texturposition</p>

Grafikablauf:

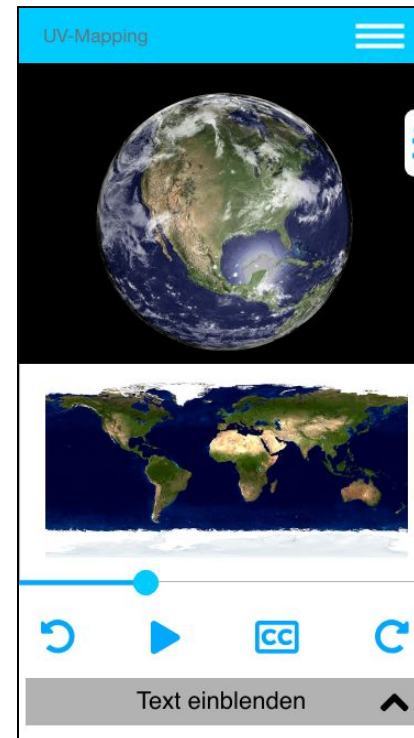
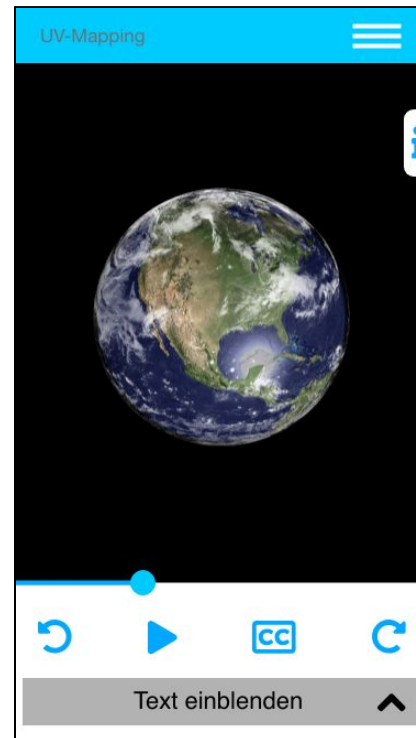




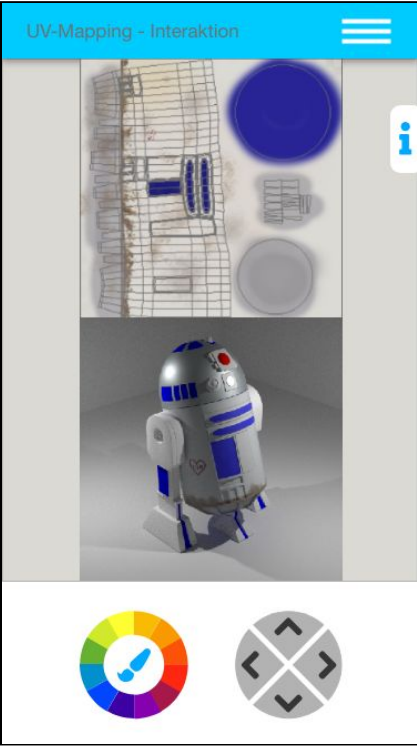
## 12.4 (A) UV-Mapping

Finaler Screen	Sprechertext	Screentext (i)	Regieanweisung
	<p>#120401 Beim UV-Mapping bildet man die Textur mit einer einfachen Abwicklung in der geeigneten Software von Hand. Der Prozess wird auch als Unwrapping bezeichnet.</p> <p>#120402 Bildlich kann man sich die Abwicklung so vorstellen, dass das 3D-Objekt an bestimmten Stellen "aufgeschnitten" und anschließend "abgewickelt" wird, wie in diesem Fall die Weltkugel.</p> <p>#120403 Versuche den Prozess der Abwicklung nachzuvollziehen.</p>	<ul style="list-style-type: none"> <li>- Texture entsteht durch einfache Abwicklung von Hand</li> <li>- Abwicklung = Unwrapping</li> <li>- Objekt wird "aufgeschnitten" und anschließend "abgewickelt"</li> </ul>	<p>#120401 Einblenden der Weltkugel</p> <p>#120402 Weltkugel wird aufgeschnitten und anschließend abgewickelt</p> <p>Texture Map (siehe Screen) entsteht</p>

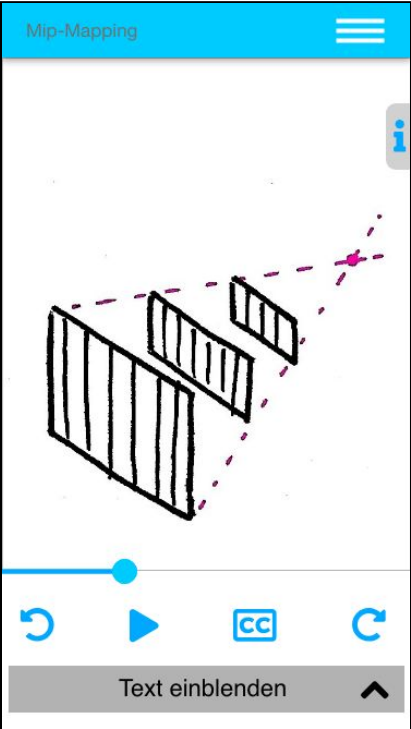
Grafikablauf:



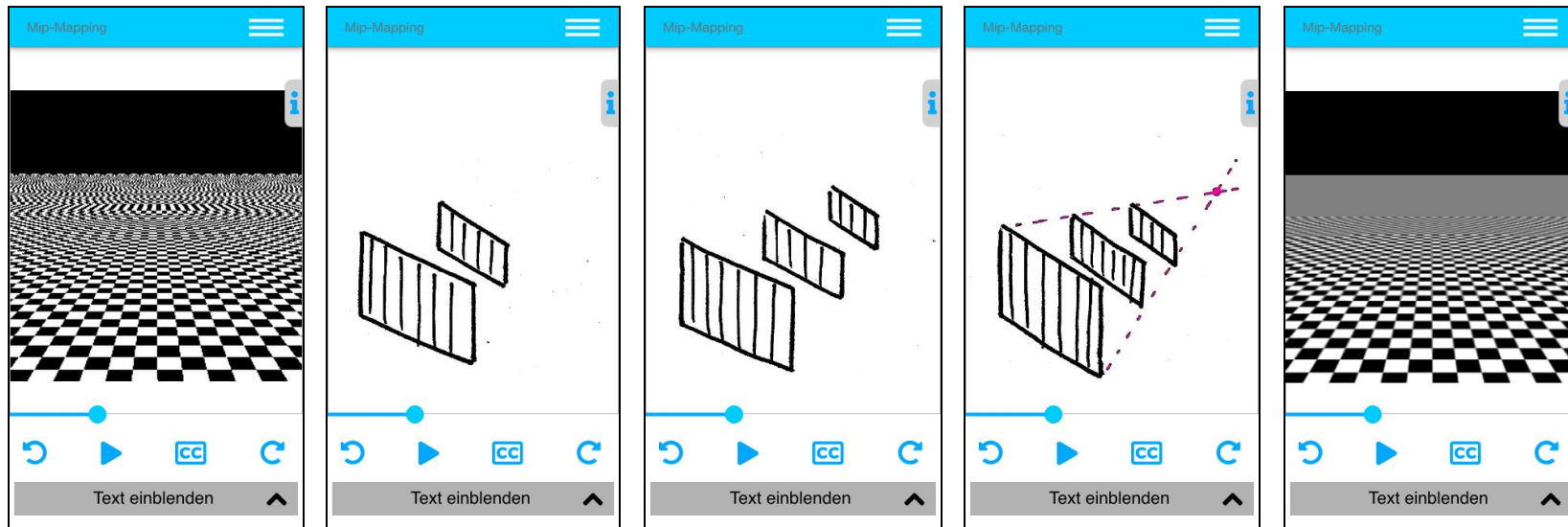
## 12.5 (I) UV-Mapping

Finaler Screen	Sprechertext	Screentext (i)	Regieanweisung
	<p>#120501 Wähle den Pinsel und suche dir eine beliebige Farbe aus. Nun kannst du die von Hand abgewickelte Textur des Roboters oder den Roboter direkt bemalen.</p>	<p>Wähle den Pinsel und suche dir eine beliebige Farbe aus. Nun kannst du die von Hand abgewickelte Textur des Roboters oder den Roboter direkt bemalen.</p>	<p>#120501 Aufgabe wird gesprochen (über das i kann sich der Nutzer die Aufgabe anzeigen lassen)</p> <p>Klickt der Nutzer auf die Textur, vergrößert sich diese</p> <p>Klickt der Nutzer auf den Roboter, vergrößert sich dieser</p> <p>Roboter kann vom Nutzer beliebig über das Steuerkreuz gedreht und von allen Seiten betrachtet werden</p>

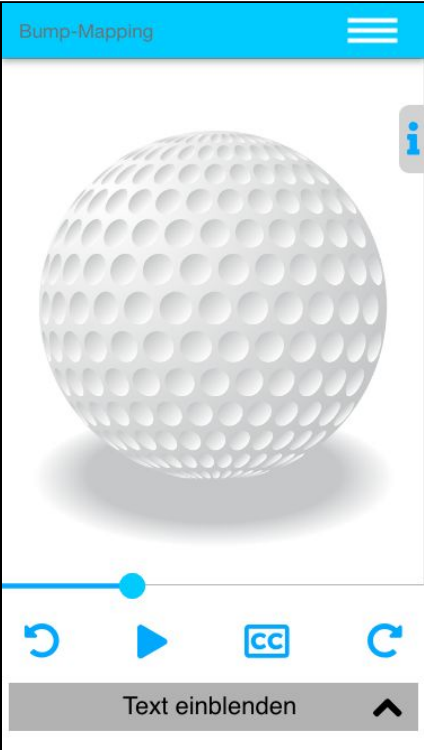
## 12.6 (A) Mip-Mapping

Finaler Screen	Sprechertext	Screentext (i)	Regieanweisung
	<p>#120601 Das Mip-Mapping ist eine Anti-Aliasing-Methode. In der Computergrafik tritt der Alias-Effekt beim Betrachten von einer Textur mit hoher Auflösung aus weiter Entfernung auf, wodurch Muster und Verzerrungen entstehen, die im Originalbild nicht enthalten sind.</p> <p>#120602 Beim Mip-Mapping werden von einer Textur mehrere vorberechnete skalierte Texturen mit sinkender Auflösung berechnet. Dabei wird in jedem Schritt die Kantenlänge des Originals halbiert.</p> <p>#120603 Ist das texturierte Polygon nahe beim Betrachter oder groß skaliert, kommt eine große Textur mit hoher Auflösung zum Einsatz. Ist es jedoch weiter entfernt oder klein skaliert, wird ein kleineres Mip-Map-Level verwendet.</p> <p>#120604 Der große Vorteil des Mip-Mappings besteht darin, dass die verschiedenen Maps zum Zeitpunkt des Renderns bereits vorberechnet sind.</p>	<ul style="list-style-type: none"> <li>- Anti-Aliasing-Methode</li> <li>- Entstehung von Verzerrungen und ungewollten Mustern</li> <li>- Kantenlänge wird bei jedem Schritt halbiert</li> <li>- Objekt nahe beim Betrachter: große Textur mit hoher Auflösung</li> <li>- Objekt weiter entfernt vom Betrachter: kleine Textur mit niedriger Auflösung</li> </ul>	<p>#120601 Einblenden eines Gitters mit Alias Effekt</p> <p>#120602 Einblenden einer großen Map mit hoher Auflösung, nahe beim Betrachter Einblenden einer zweiten Map mit halbierten Kantenlängen</p> <p>#120603 Einblenden einer dritten Map mit nochmals halbierten Kantenlängen  Einblenden des Fluchtpunktes als gestrichelte Linien</p> <p>#120604 Einblenden des Gitters mit Mip-Mapping und ohne Alias-Effekt</p>

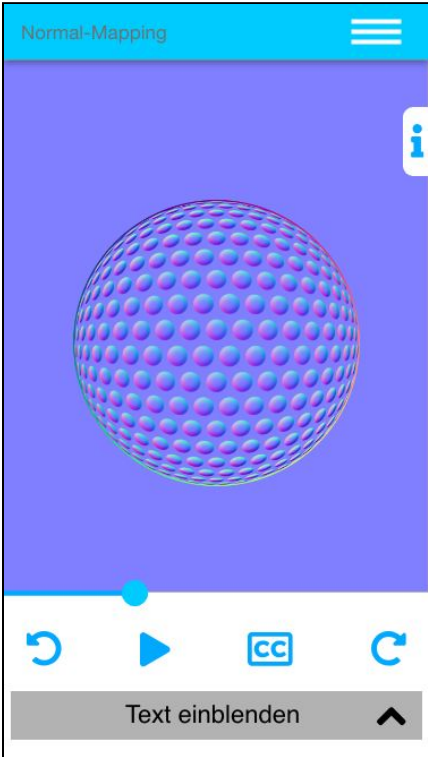
Grafikablauf:




## 12.7 (A) Bump-Mapping

Finaler Screen	Sprechertext	Screen text (i)	Regieanweisung
	<p>#120701 Eine normale 2D-Textur verleiht einem Objekt keine Oberflächeneigenschaften wie Höhe und Tiefe. Das Bump-Mapping dient dazu, einem 3D-Objekt diese Eigenschaften zu verleihen, ohne dabei die Geometrie des 3D-Objektes zu beeinflussen.</p> <p>#120702 Beim Bump-Mapping wird auf Pixelebene lediglich die Normale der Oberfläche verändert. Es wird mit Hilfe von Schattierung und Reflektion eine Illusion von Tiefe auf dem 3D-Objekt erzeugt. Das Bump-Mapping verwendet in der Regel verschiedene Graustufen und beschreibt damit Höhenunterschiede auf dem Objekt. Somit ist das Bump-Mapping ein reiner Beleuchtungseffekt.</p> <p>#120703 Der Vorteil des Bump-Mapping besteht darin, dass bei diesem Mapping-Verfahren der Speicherplatz und die Rendering-Zeit ziemlich gering bleiben.</p>	<ul style="list-style-type: none"> <li>- Erzeugung von Tiefe ohne Beeinflussung der Geometrie</li> <li>- Veränderung der Normalen der Oberfläche</li> <li>- Illusion durch Graustufen</li> <li>- Beleuchtungseffekt</li> <li>- Vorteil: geringe Erhöhung des Speicherplatzes und der Rendering-Zeit</li> </ul>	<p>#120701 Einblenden einer Sphäre mit Bump Mapping</p> <p>Sphäre dreht sich um sich selbst</p>

## 12.8 (A) Normal-Mapping


Finaler Screen	Sprechertext	Screentext (i)	Regieanweisung
	<p>#120801 Das Normal-Mapping ist eine Abwandlung des Bump-Mapping-Verfahrens. Es zielt ebenfalls darauf ab, einen größeren Detailreichtum eines Objektes in Form von Höhe und Tiefe zu erzeugen.</p> <p>#120802 Beim Normal-Mapping wird die Anzahl der Polygone nicht erhöht und die Oberfläche nur optisch verformt. Alle Informationen über die Ausrichtung der Normalen, die für die Beleuchtung wichtig sind, werden in Form von RGB-Werten von einem hoch aufgelösten auf ein niedrig aufgelöstes Objekt übertragen. Die optischen Details gehen dabei nicht verloren.</p> <p>#120803 Der Nachteil dieses Mapping-Verfahrens ist, dass es stark vom Betrachtungswinkel abhängig ist.</p>	<ul style="list-style-type: none"> <li>- Abwandlung des Bump-Mapping</li> <li>- Ziel: größeren Detailreichtum</li> <li>- Illusion durch Übertragung von RGB-Werten</li> <li>- optische Details bleiben erhalten</li> <li>- Nachteil: winkelabhängig</li> </ul>	<p>#120801 Einblenden einer Sphäre mit Normal-Mapping</p> <p>Sphäre dreht sich um sich selbst</p>

## 12.9 (A) Displacement-Mapping

Finaler Screen	Sprechertext	Screen text (i)	Regieanweisung
	<p>#120901 Das Displacement-Mapping löst das selbe Problem wie das Bump- und das Normal-Mapping, es verleiht dem 3D-Objekt ebenfalls Vertiefungen und Erhöhungen.</p> <p>#120902 Der Unterschied zwischen den Mapping-Verfahren ist, dass beim Displacement Mapping die Geometrie des 3D-Objektes verändert wird. Das Displacement-Mapping findet also auf polygonaler Ebene statt.</p> <p>#120903 Durch "Verschieben" beziehungsweise "Verdrängen" des Materials werden dem 3D-Objekt die gewünschten Oberflächeneigenschaften verliehen.</p>	<ul style="list-style-type: none"> <li>- Vertiefungen und Erhöhungen der Oberfläche des Objektes</li> <li>- Veränderung der Form durch Veränderung der Geometrie des 3D-Objektes</li> <li>- displacement = "verschieben" bzw. "verdrängen" des Materials</li> </ul>	<p>#120901 Einblenden einer Sphäre mit Displacement-Mapping</p> <p>Sphäre dreht sich um sich selbst</p>



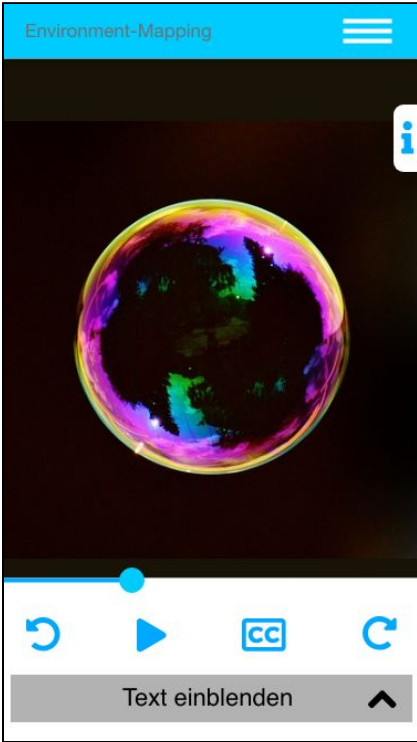
## 12.10 (I) Vergleich

Finaler Screen	Sprechertext	Screentext (i)	Regieanweisung
	<p>#121001 Lasse dir nun den Unterschied zwischen dem Normal-Mapping und dem Displacement-Mapping anzeigen. Betrachte die texturierten 3D-Objekte dabei von allen Seiten.</p>	<p>Lasse dir nun den Unterschied zwischen dem Normal-Mapping und dem Displacement-Mapping anzeigen. Betrachte die texturierten 3D-Objekte dabei von allen Seiten.</p>	<p>#121001 Aufgabe wird gesprochen (über das i kann sich der Nutzer die Aufgabe anzeigen lassen)</p> <p>Auswahlmöglichkeiten "Normal-Mapping" "Displacement-Mapping"</p> <p>Der Nutzer kann sich das Objekt über das Steuerkreuz von allen Seiten ansehen</p> <p>(Eventuell Intensitätsregler, über den der Nutzer die Intensität des jeweiligen Mapping-Verfahrens beliebig anpassen kann)</p>

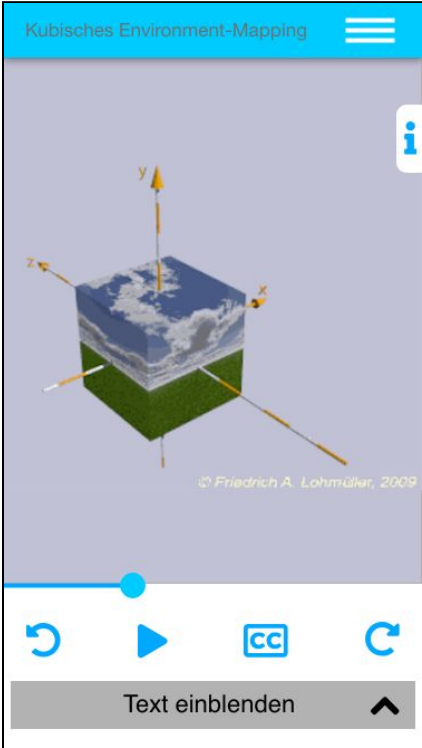
Grafikablauf:



## 12.11 (A) Environment-Mapping

Finaler Screen	Sprechertext	Screentext (i)	Regieanweisung
	<p>#121101 Mit dem Environment-Mapping werden in der Computergrafik Spiegelungen, durchlässige Objekte und Beleuchtungs-Simulationen dargestellt.</p> <p>#121102 Dabei wird die Umgebung, bestehend aus Objekten und Lichtquellen, in der sich das Objekt befinden soll, als eine Umgebungs-Textur gespeichert und auf das 3D-Objekt projiziert. Wie dieser Vorgang funktioniert erfährst du im nächsten Kapitel. Das 3D-Objekt muss im Verhältnis zu seiner Umgebung eher klein sein, damit die Spiegelung realistisch wirkt.</p> <p>#121103 Beim Environment-Mapping wird in zwei Arten unterschieden: das sphärische und das kubische Environment-Mapping. In Folge wird nur Letzteres beschrieben.</p>	<ul style="list-style-type: none"> <li>- Simulation von spiegelnden Objekten</li> <li>- Speichern einer Umgebungstextur (umliegende Objekte und Lichtquellen)</li> <li>- Sphärisches Environment-Mapping</li> <li>- Kubisches Environment-Mapping</li> </ul>	<p>#121101 Einblenden eines spiegelnden 3D-Objektes (hier: Seifenblase, später: spiegelnder Roboter)</p>

## 12.12 (A) Kubisches Environment-Mapping

Finaler Screen	Sprechertext	Screentext (i)	Regieanweisung
	<p>#121201 Bei dem kubischen Environment-Mapping, wird die Umgebung auf einem Kubus abgebildet.</p> <p>#121202 Es werden sechs 2D-Umgebungs-Texturen verwendet, die zusammen die Fläche eines Kubus bilden.</p> <p>#121203 Die sechs einzelnen Umgebungs-Texturen entstehen durch das Fotografieren oder Rendern aus Position des Objektmittelpunkts in einem Winkel von 90 Grad. Dabei müssen die sechs Würfelflächen randlos abgedeckt werden.</p>	<ul style="list-style-type: none"> <li>- Kubisches Environment-Mapping: Abbildung der Umgebung auf einen Kubus</li> <li>- Vorteil: keine Verzerrung</li> <li>- Verwendung von sechs 2D-Umgebungs-Texturen</li> </ul>	<p>#121201 Einblenden von sechs 2D-Umgebungs-Texturen Die sechs 2D-Umgebungs-Texturen legen sich um den Kubus und decken alle Flächen randlos ab</p> <p>2D-Textur klappt wieder auf Vorgang wiederholt sich, solange der Sprecher spricht</p> <p>Quelle und Code-Beispiel: Friedrich A. Lohmüller, <a href="http://www.f-lohmueller.de/pov_tut/backupgrnd/p_sky9d.htm">http://www.f-lohmueller.de/pov_tut/backupgrnd/p_sky9d.htm</a></p>

Grafikablauf:

