Virtuelle Realität (VR)
Grundlagen 3D
Grundlagen VRML

2. Virtuelle Realität

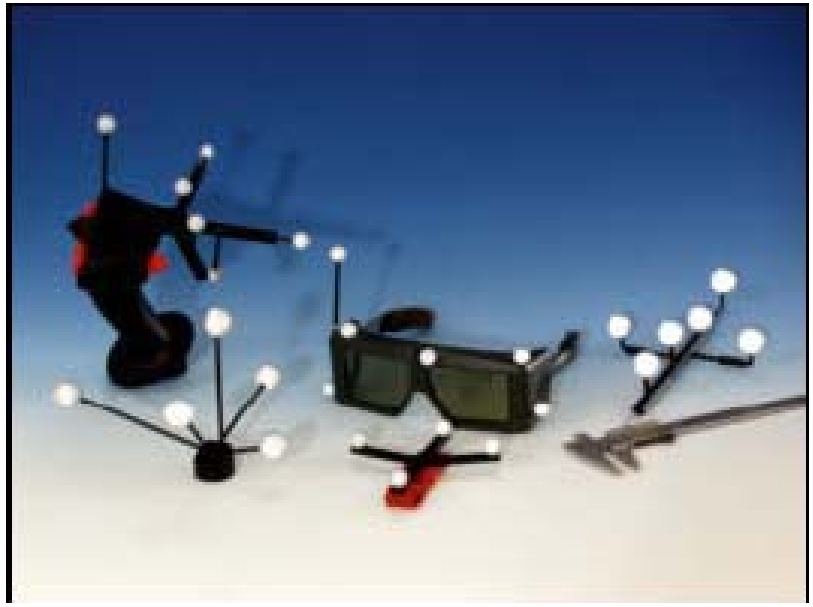
2.1 Was ist Virtual Reality?

- Unter Virtual Reality versteht man alle Konzepte, die unsere natürliche audiovisuelle und räumliche Wahrnehmung im Computer simulieren.

- VR umfasst Methoden zur räumlichen Darstellung von Objekten, für die räumliche Klangausbreitung und für eine Reihe von Interaktionsmöglichkeiten.

- Über ***Trackingsysteme*** können Bewegungen des Menschen in die synthetische Umgebung übertragen werden,

- über ***Sensoren*** kann man wie in der Realität Objekte der künstlichen Welt beeinflussen.




NVIDIA
GeForce
8800 GTX

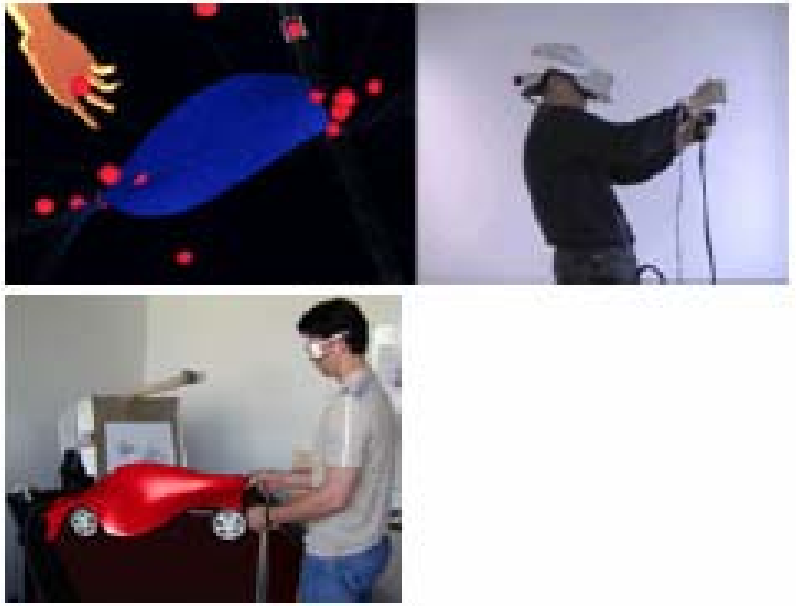
2.2 Anwendungsbereiche der VR

- Architektur: Walk through

- Städteplanung

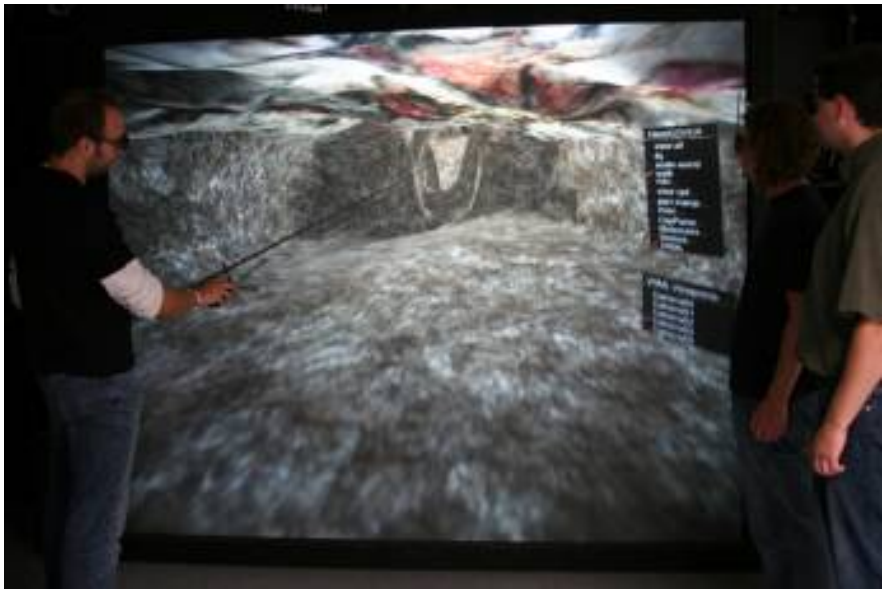
- Industrielle Fertigung: Montagesimulation, Entwurf von Fertigungsanlagen

- Design: Automobilindustrie: Prototypisierung



- Ausbildung und Training : Flugsimulation

- **Kulturelles Erbe: Simulationen in der Archäologie**
  (Ausgrabungssimulation)



- **Kulturelles Erbe: Simulationen in der Archäologie**
  (Ausgrabungssimulation)

Links: Die reale Höhle von Altamira, rechts: das 3D-Modell

2.3 Zentrale Aspekte der VR

2.3.1 VR-Systeme

- VR-Systeme bilden die Schnittstelle zwischen dem Benutzer und der Virtuellen Welt.
- Man unterscheidet:
  - *Desktop-VR:* Der Benutzer schaut von einem Standpunkt ausserhalb auf die zweidimensionale Darstellung der Virtuelle Welt.
  - *Stereo Vision*: Durch stereoskopische Sichtsysteme (z.B. Shutterbrille) entsteht trotz zweidimensionaler Darstellungsform ein realer dreidimensionaler Eindruck.
  - *Immersive Systeme*: Der persönliche Blickpunkt des Betrachters liegt vollständig in der Virtuellen Welt. Dies wird z.B. durch einen *HMD (Head Mounted Device)* erreicht.

2.3.2 Immersion

- Der Grad, inwieweit ein Benutzer die virtuelle Welt als eigene Realität wahrnimmt, wird als *Immersionsgrad (Immersion= „Eintauchen“)* eines VR-Systems bezeichnet.
- Werden alle Sinne konsistent angesprochen oder wird die reale Welt nicht mehr wahrgenommen, so liegt ein maximaler Immersionsgrad vor. Der Immersionsgrad kann also über die Abschirmung von der realen Welt oder die Anzahl der angesprochenen Sinne bestimmt werden.

2.3.3 Rendering

- Als *Rendern* bezeichnet man den Prozess, für eine Bewegung in einer Szene die gesamte Szene mit allen Objekten neu zu berechnen und darzustellen.
- Wie kann eine Szene optimal gerendert werden?

2.4 Räumliche Modellierung von Objekten

- Die Grundlage jeder räumlichen Modellierung ist ein *3D-Koordinatensystem*.
- Die Gestalt eines beliebigen Objekten wird durch Punkte im Koordinatensystem beschrieben.
- Verbindet man die Punkte miteinander, entsteht ein *Drahtmodell.*

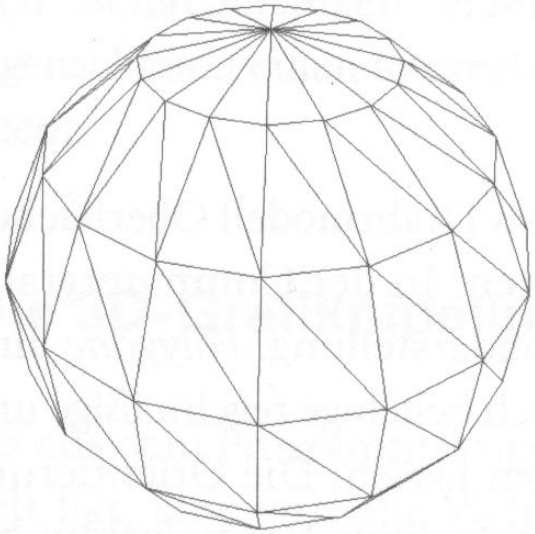


Abb. 1.3: Drahtmodell einer Kugeloberfläche

- Das Drahtmodell alleine erlaubt keine eindeutige Identifikation der Raumorientierung des Objekts.
- Die eindeutige Raumorientierung erreicht man mit *Perspektive* und *Verdeckung*.

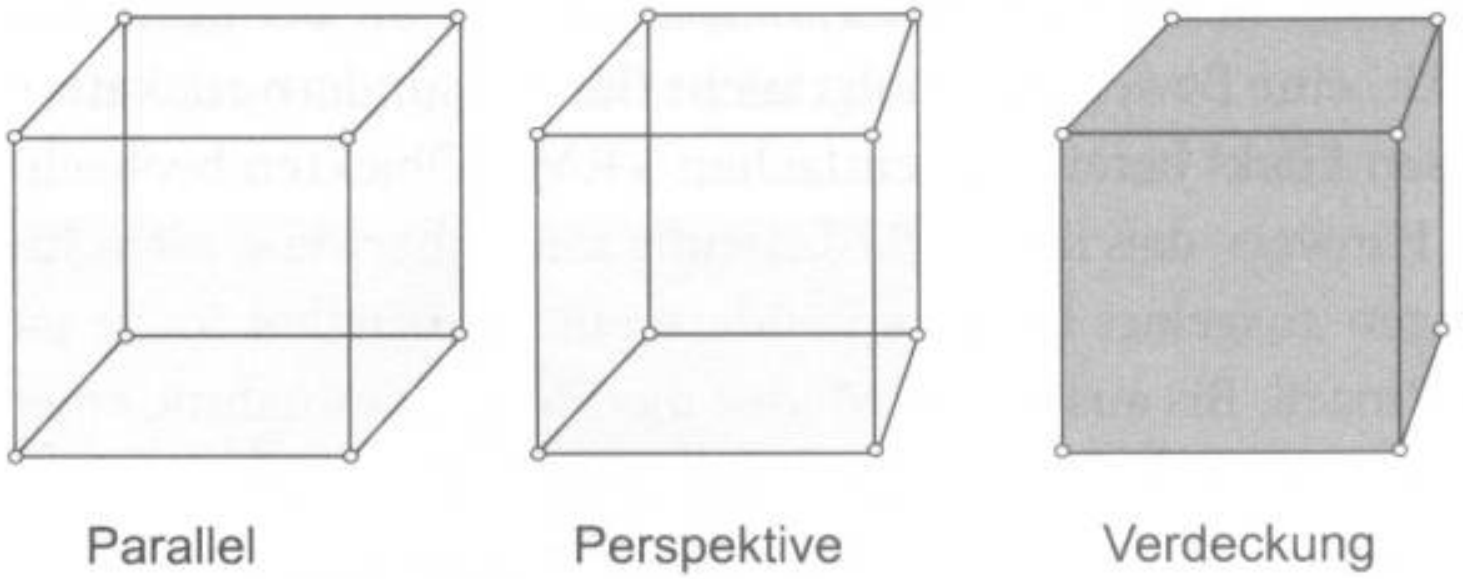


Abb. 1.4: Parallelprojektion, Perspektive und Verdeckung

Polygondarstellung

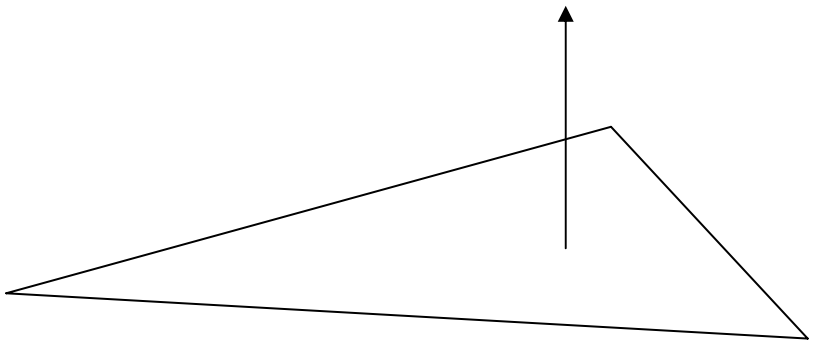
- Reale Objekte besitzen im Gegensatz zu einem Drahtmodell **Oberflächen mit physikalischen Eigenschaften**.

- In der Computergrafik verwendet man dazu die **Polygondarstellung**.

- ***Polygone*** sind kleine Flächeneinheiten, aus denen sich beliebige regelmäßige oder unregelmäßige Oberflächen zusammensetzen lassen.

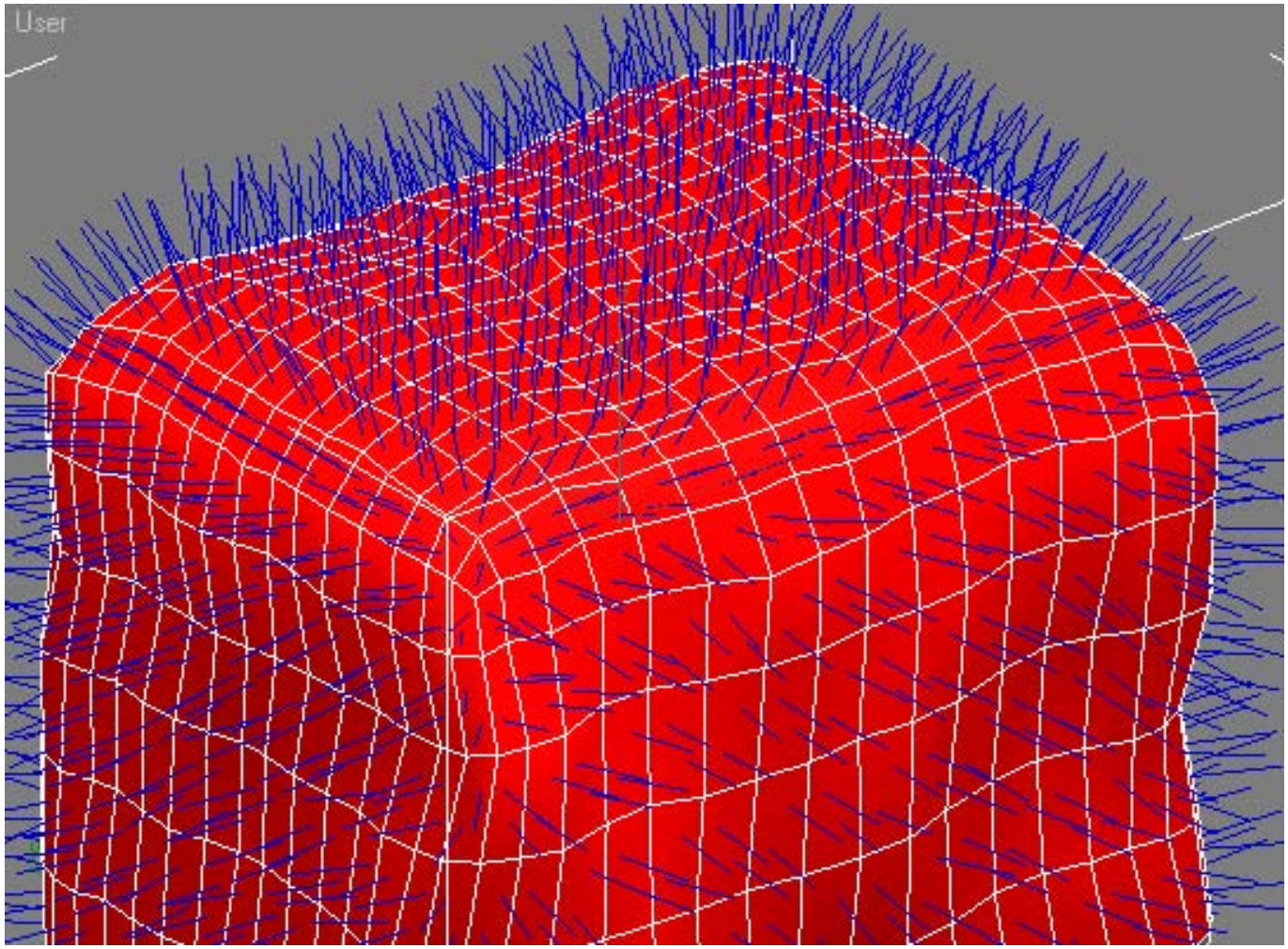
- Die Orientierung jedes Polygons im Raum wird durch einen gedachten ***Normalvektor*** beschrieben, der senkrecht auf dem Flächenstück steht. Die Normale zeigt also in die Richtung, von der aus das Polygon sichtbar ist. Betrachtet man das Polygon „von hinten“ ist es nicht sichtbar, da Polygone nur eine Vorderseite haben!

- Da in der Praxis nur eine **begrenzte Zahl an Polygonen** darstellbar ist, kann man sich realen Objekten mit der Polygondarstellung lediglich annähern.



Polygondarstellung

- Ein Beispiel aus der Praxis: Normalendarstellung mit 3ds Max:



Oberflächeneigenschaften

- Eine Szene, die nur in der Polygondarstellung gerendert wurde, erscheint trotz Perspektive und Verdeckung flach und unwirklich.
- Realistische Darstellung von künstlichen Objekten wird erst durch Oberflächeneffekte erzielt.
- Oberflächeneffekte kommen durch Beleuchtung einer Szene und die charakteristischen Schattenverläufe einer Oberfläche zustande.

2.5 Vom Programmskript zur virtuellen Welt: Die 3D-Grafikpipeline

- Der gesamte Weg den 3D-Daten hin zum 2D-Frame durchlaufen - vom Modell über mathematische Beschreibung einer Szene bis hin zur gerasterten Darstellung - wird als (3D-) Grafikpipeline (oder oft auch 3D-Rendering Pipeline) bezeichnet.

- Die Grafikpipeline besteht aus zwei Hauptteilen:
  1. Geometrie-Teil
  2. Render-Engine

1. Die Routinen im **Geometrieteil** ermitteln aus dem vorgegebenen Modell des Entwicklers eine dem jeweiligen Blickwinkel des Betrachters angepasste Szene.

2. Die **Render-Engine** berechnet daraus für jeden neuen Blickwinkel die entsprechenden Bilder, die dann auf dem dem Bildschirm sichtbar sind.

- Beispiele für Grafik-API: **DirectX**, **OpenGL**

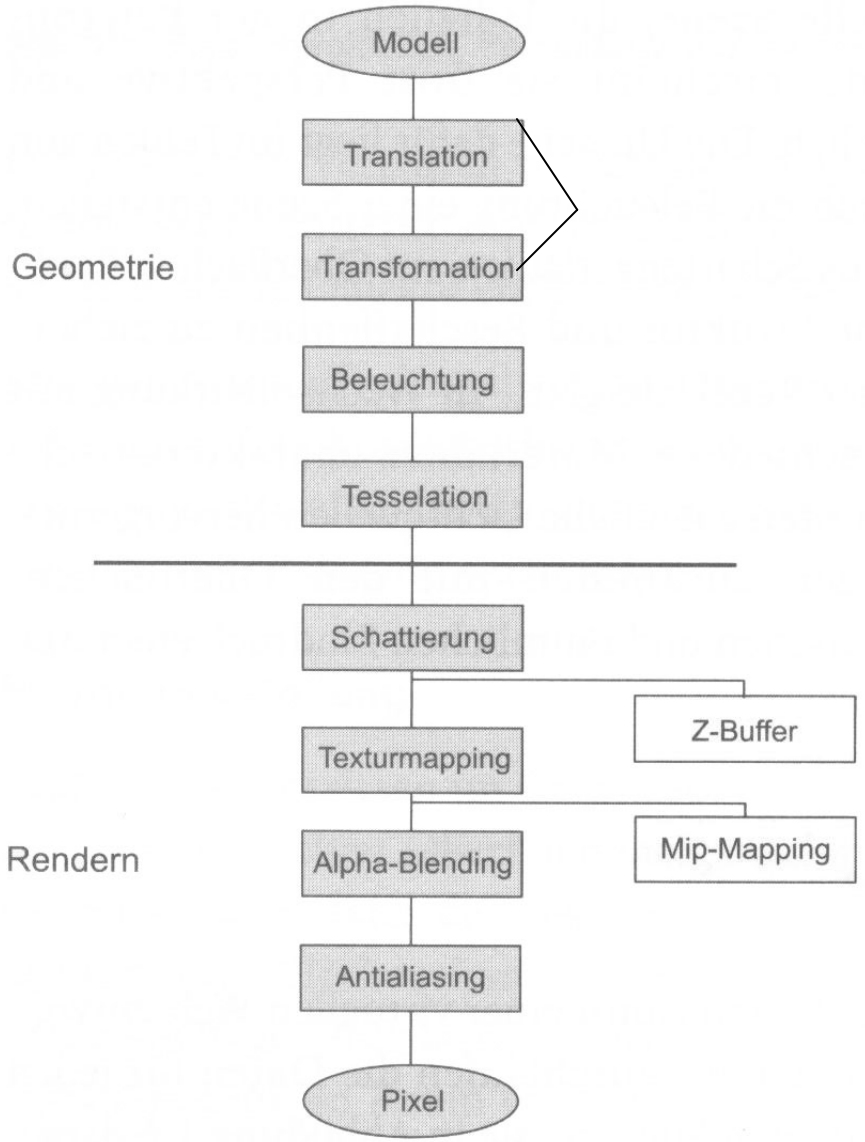


Abb. 1.6: Die 3D-Grafikpipeline

Transformationen

- Die Transformation enthält verschiedene Einzelschritte:
  - ***Translation***: Einfache Objekt- (Punkt-)Verschiebung im Raum
  - ***Skalierung***: Vergrößern/ Verkleinern von Objekten (Polygonen) um einen bestimmten Faktor.
  - ***Rotation***: Rotation eines Körpers um einen bestimmten Punkt.

Beleuchtung

- Berechnung des Einflusses verschiedener Lichtquellen auf die Objekte.
- Der Standpunkt des Beobachters ist wie auch in der realen Welt ausschlaggebend für die Szenenbeleuchtung.

Tesselation

- Zerlegung nichtregulärer Polygone in reguläre Formen, die besser von der Rendermaschine verarbeitet werden können (→ **Dreiecke**).

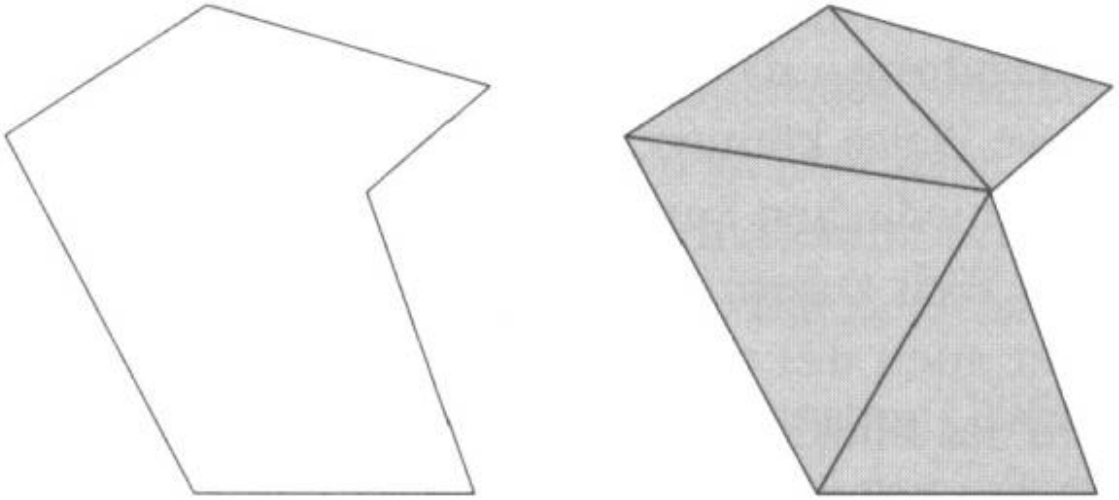


Abb. 1.7: Tesselation

Shading *(Schattierung)*

- Beim ***Flat-Shading***-Verfahren wird anhand eines Beleuchtungsmodells die Schattierung für jedes Polygon berechnet. Das Flat-Shading erfordert nur geringe Rechenleistung, liefert aber auch nur grobe Schattierungen.

- Beim **Gouraud-Shading werden,** im Gegensatz zum Flat-Shading, bei dem man einen einzigen Intensitätswert für das gesamte Polygon heranzieht, die Farbwerte an jedem Eckpunkt (Vertex) berechnet. Anschließend interpoliert das Verfahren linear diese Werte über die Kanten des Polygons und über jede Scanline, um so das Polygon zu schattieren.

- Beim ***Phong-Shading*** werden an den Eckpunkten (Vertices) eines Polygons die Normalen berechnet und dann zusätzlich die Normalen für jedes Pixel des Polygons interpoliert. Farbstärken berechnen sich aus den interpolierten Normalen. Die Ergebnisse des Phong Shadings sind qualitativ besser als die des Gouraud Shadings, allerdings sind die mathematischen Berechnungen aufwändiger, da sie für jedes Pixel durchgeführt werden.

Textur-Mapping

- Als ***Textur*** bezeichnet man Muster oder Bilder, die wie eine „Tapete“ auf die Oberfläche eines Drahtmodells aufgebracht werden.

- Mit **Texturen** können **Raumeffekte** vorgetäuscht werden, die sonst sehr aufwendig mit Polygonen modelliert werden müssten (→ Bump-Mapping, Normal-Mapping).

- Die **Zahl der Polygone** einer Szene kann so **reduziert** werden.

- **Textur-Mapping** ist eine wichtige Technik zur Erhöhung der realistischen Darstellung

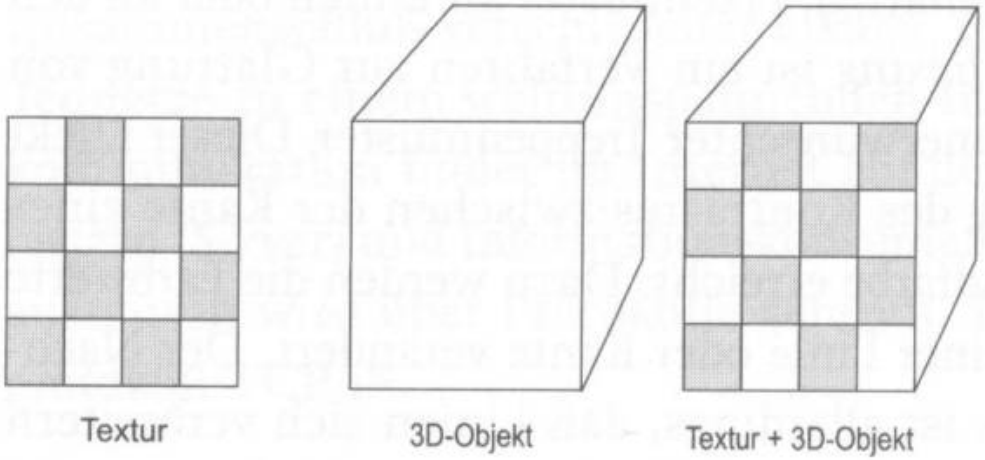
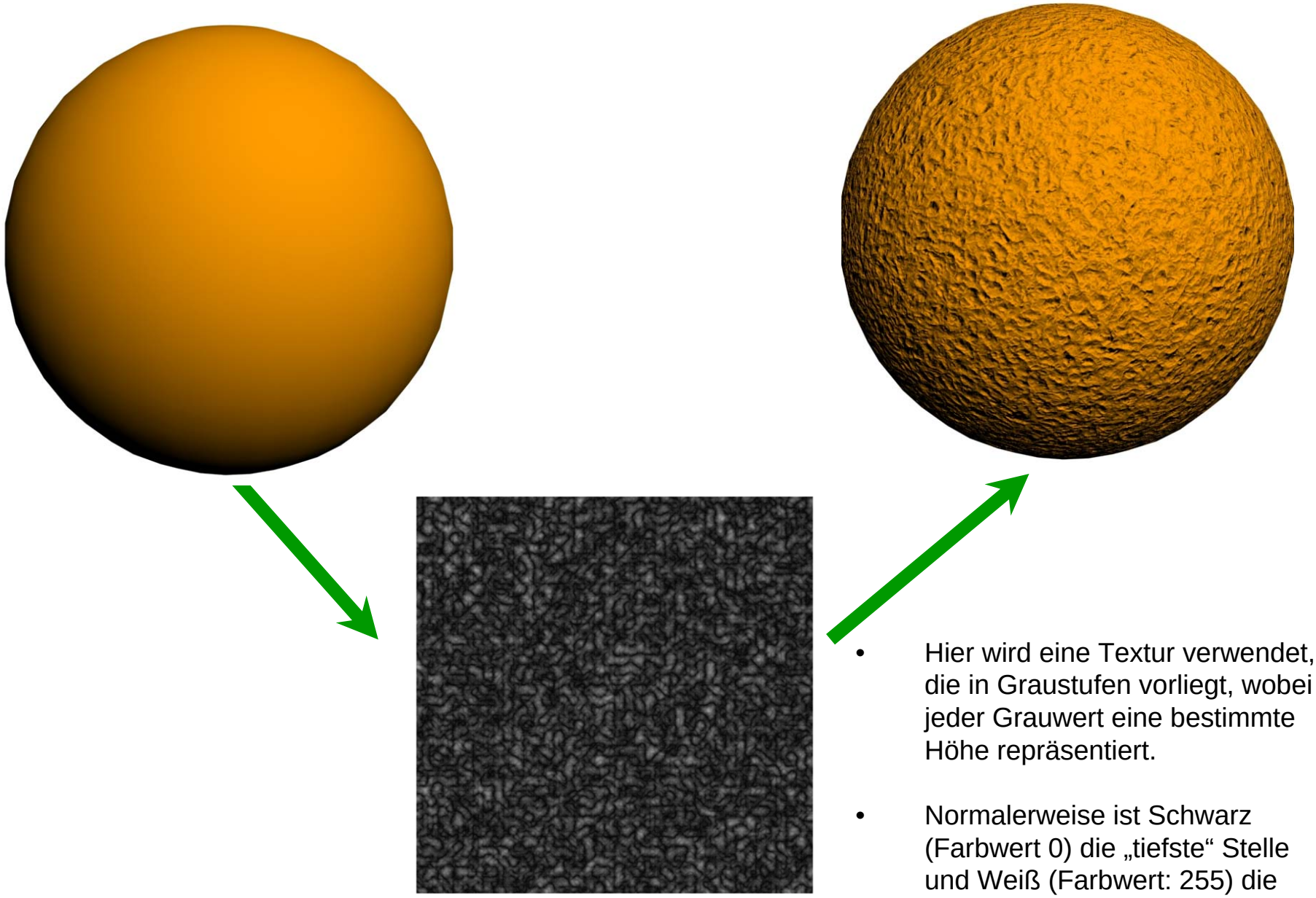


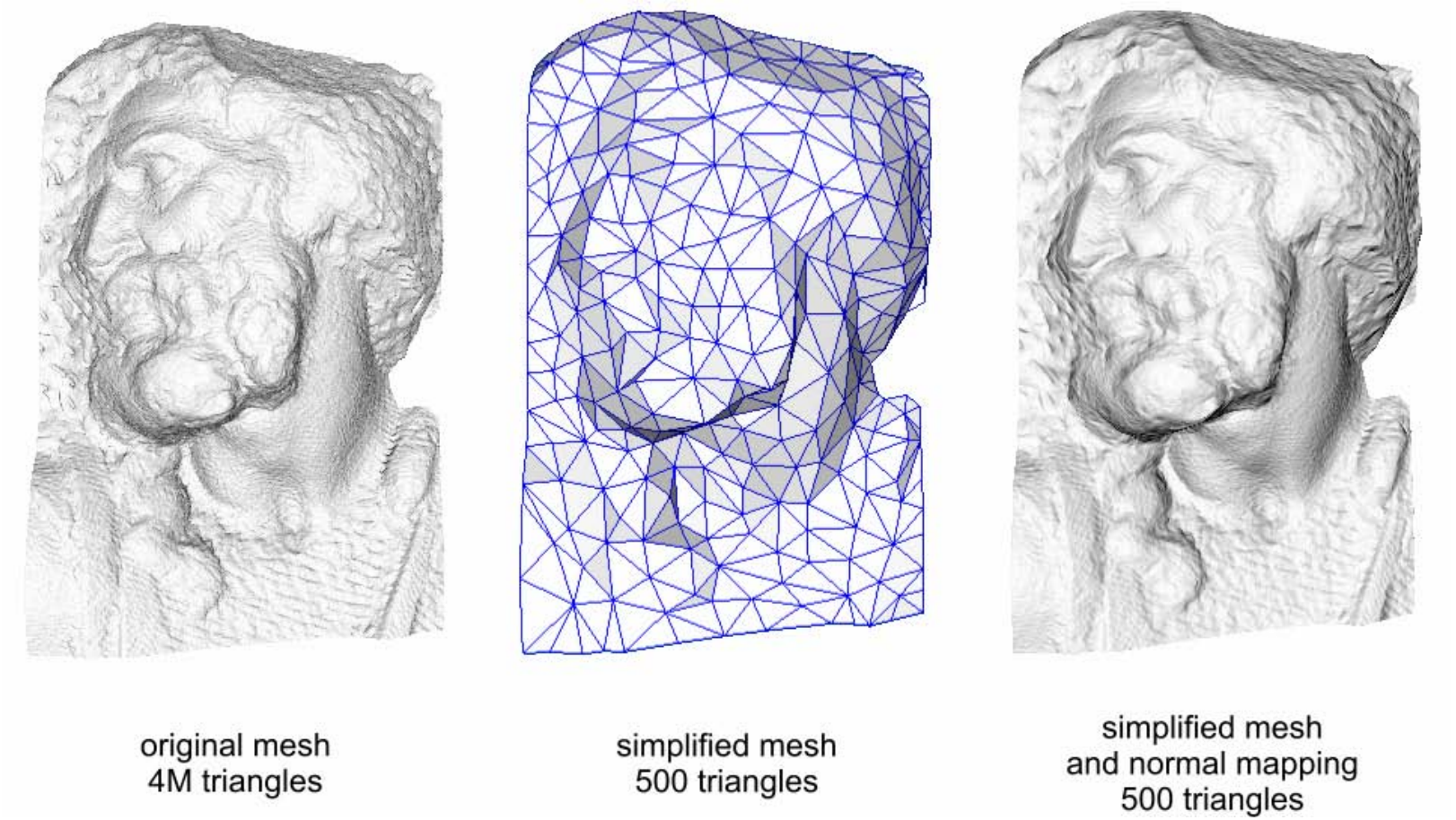
Abb. 1.11: Textur-Mapping

Bump-Mapping



- Hier wird eine Textur verwendet, die in Graustufen vorliegt, wobei jeder Grauwert eine bestimmte Höhe repräsentiert.
- Normalerweise ist Schwarz (Farbwert 0) die „tiefste“ Stelle und Weiß (Farbwert: 255) die „höchste“.

Normal-Mapping



- Eine **Normalmap** ist eine visuelle Repräsentation der **Flächennormalen**. Beim Normal-Mapping werden die für die **Beleuchtung relevanten Informationen** über die **Ausrichtung der Normalen** als Farbwerte (RGB = XYZ) in einer Bilddatei gespeichert.und so von einem hoch auf ein **niedrig aufgelöstes 3D-Modell** übertragen.

- Das Detail der Oberfläche bleibt somit optisch erhalten. Sichtbar ist das niedrige Detail lediglich an der Silhouette, die noch immer der des niedrig aufgelösten Modells entspricht. Häufig wird diese Technik im Bereich der Echtzeit-3D-Grafik (z.B. in Computerspielen) verwendet.

Normal-Mapping:

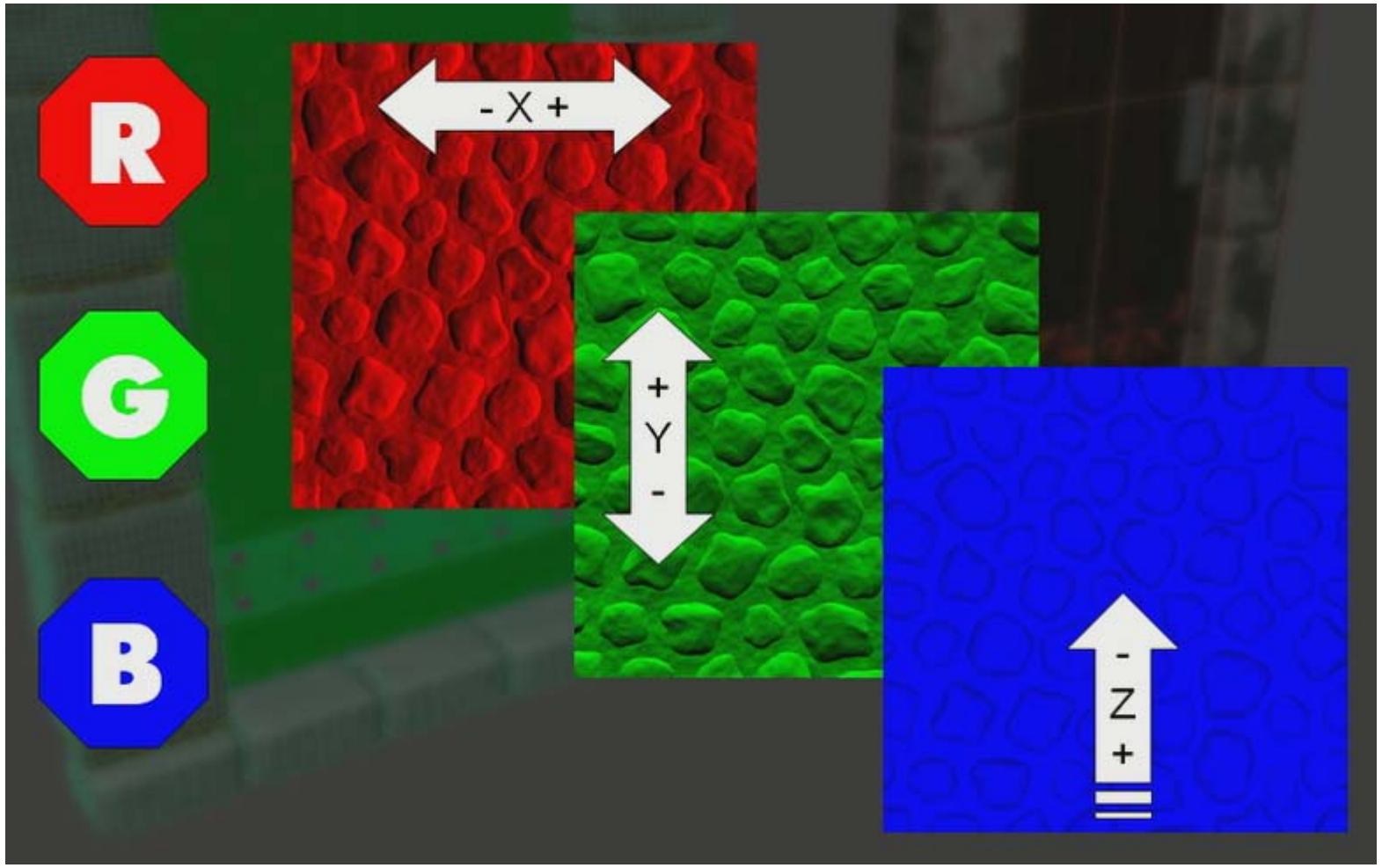
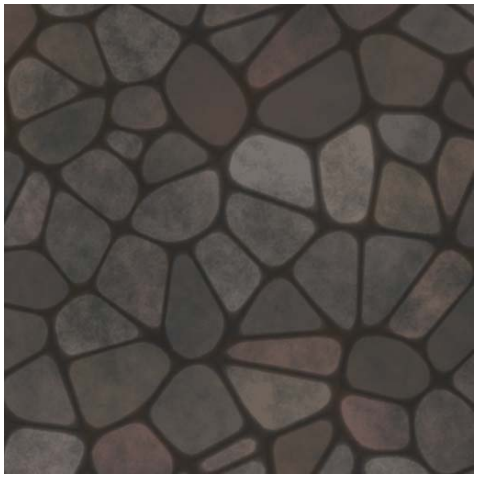


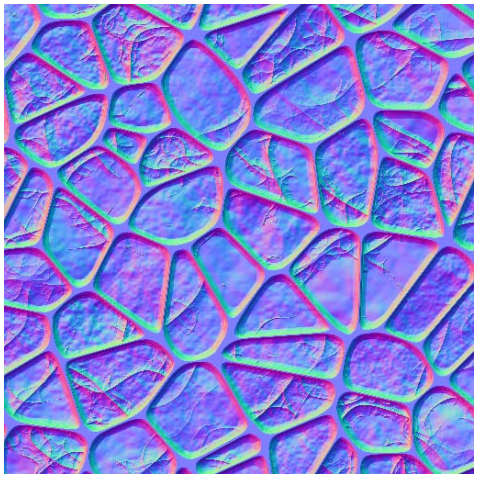
Abb.: Zuordnung der Raumkoordinaten eines Pixels zu den Farbwerten R, G, B.
(R = links/rechts, X ; G = hoch/runter, Y ; B = Tiefe im Raum, Z)

Normal-Mapping, Fortsetzung



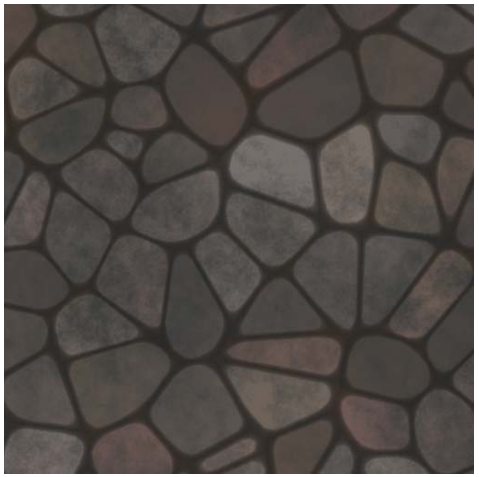
Diffusemap

+



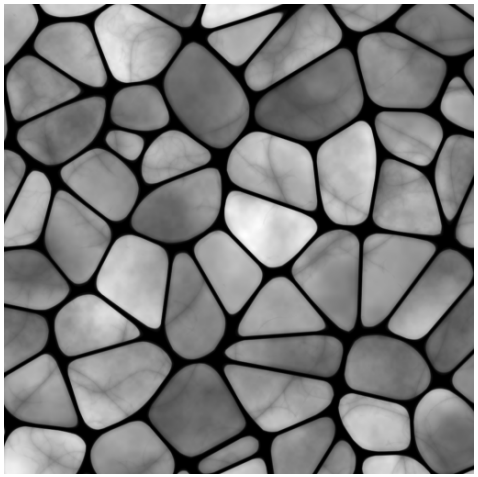
Normalmap

=



Diffusemap

+



Bumpmap

=

Normal-Mapping, Fortsetzung

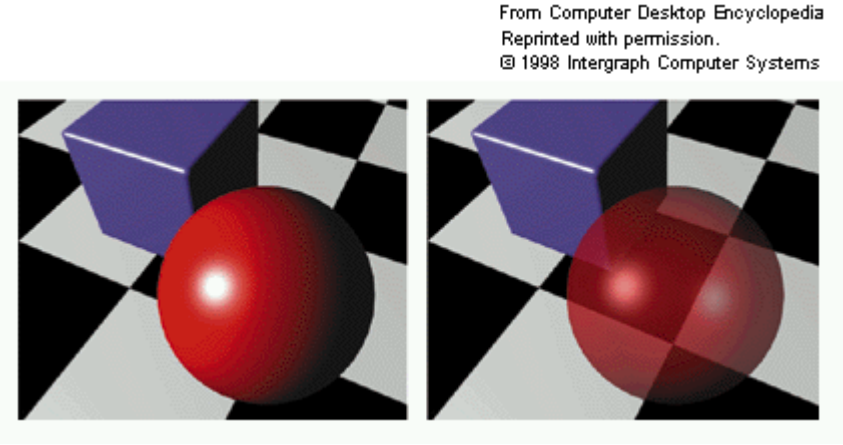
High-Polymodell
(73.728 Polygone)

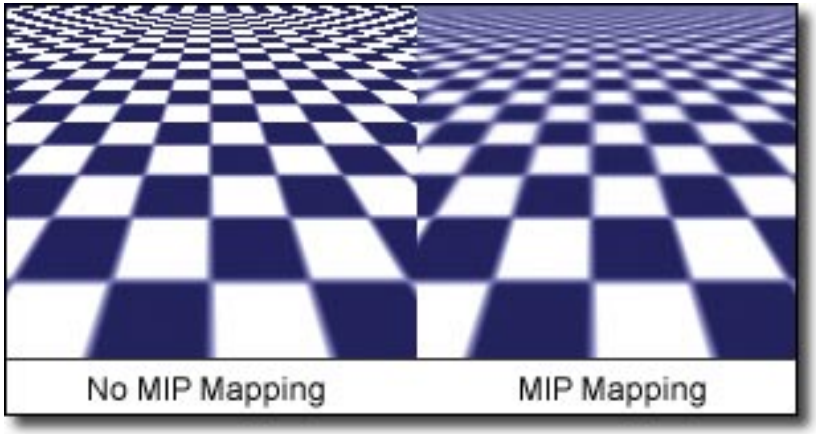
Low-Polymodell mit Normalmap
(612 Polygone)

Mip Mapping

- beim *Mip Mapping* werden gleichzeitig mehrere Versionen (Rasterbilder) einer Textur gespeichert (Mip Map), die sich in ihrem Detailliertheitsgrad, d.h. in der Auflösung, unterscheiden (Mip= Multum in parvo).

- Durch Mip Mapping kann die Qualität der Textur bei der Skalierung (also Vergrößerung oder Verkleinerung) eines Objektes mit einer Texturoberfläche beibehalten werden.





Alpha Blending

- ist ein Verfahren zur Kontrolle der Transparenz eines Objektes.

- Damit lassen sich lichtdurchlässige Oberflächen wie Glas oder Wasser simulieren.

- Auch atmosphärische Effekte wie Nebel oder *Depth Cueing*, bei denen ein Objekt mit zunehmender Distanz an Deutlichkeit verliert, können simuliert werden.

Antialiasing

- Antialiasing ist ein Verfahren zur Glättung von Kanten und zur Vermeidung von unerwünschten Treppenmustern.
- Der Effekt wird durch Reduzierung des Kontrastes zwischen der Kante des Objektes und dem Hintergrund erreicht.
- Nachteilig ist, dass Linien sich verbreitern und Kanten an Kontrast verlieren.

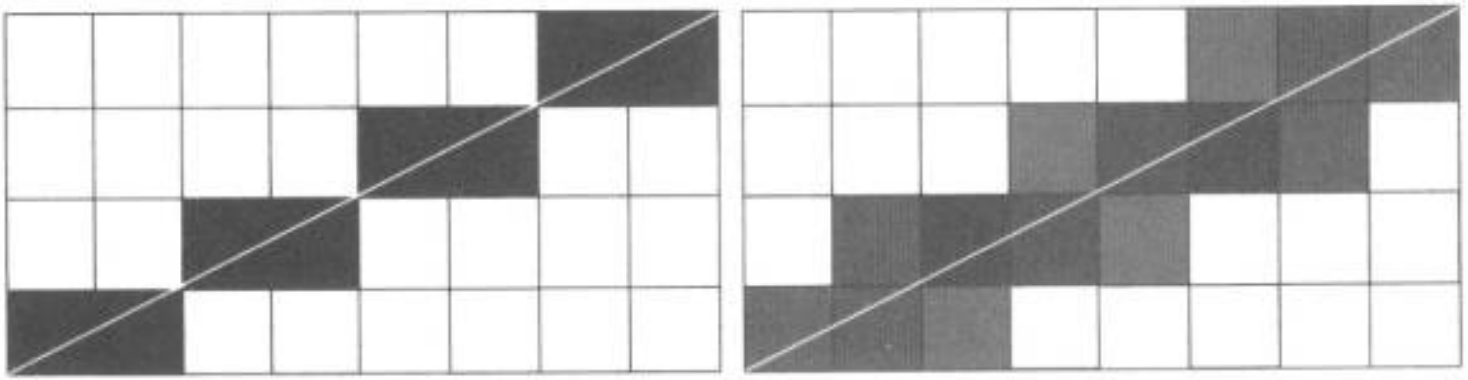


Abb. 1.12: Antialiasing

3. VRML

3.1 Was ist VRML?

- **VRML**(VirtualRealityModelingLanguage) ist eine Beschreibungssprache für 3D-Inhalte.

- VRML97 ist als ISO/IEC 14772 standardisiert.

- 1999 wurde dann VRML99 verabschiedet.

- Der VRML-Nachfolger **X3D**hat sich bis heute nicht etabliert.

3.2 Funktionalitäten von VRML

- VRML wurde im Wesentlichen für das **Internet** konzipiert.
- VRML erlaubt die Erschaffung von animierten und interaktiven 3D Inhalten.
- VRML unterstützt primitive 3D-Objekte (Würfel, Kubus, etc).
- VRML unterstützt 3D-Objekte mit komplexen Oberflächen (IndexedFaceSet).
- VRML hat die Fähigkeit, Objekte mit Materialien zu versehen (Texturen, allerdings **keine** Bumpmaps o.ä.).
- VRML unterstützt (statische) Licht-und Schatteneffekte.
- VRML unterstützt „**Kamerafahrten**“ bzw. vorberechnete **Animationen**, also die Möglichkeit, Objekte und Szenen aus verschiedenen Perspektiven darzustellen.
- VRML unterstützt Hyperlinks, die es erlauben, Objekte mit Webinhalten zu verbinden.
- Objekte können in VRML definiert und wiederverwendet werden.

3.3 VRML: Präsentation und Interaktion

- Die Interpretation, Ausführung, und Präsentation von VRML-Dateien wird typischerweise von einem Browser vorgenommen, welcher die im Szene-Graphen enthaltenen Objekte und Geräusche wiedergibt.

- Die Präsentation wird als Virtuelle Welt (*virtual world, virtual reality*) bezeichnet.

- Die Virtuelle Welt wird von einem bestimmten Standort aus gesehen dargestellt. Diese Position und Orientierung in der Welt wird als ***Viewpoint*** bezeichnet.

- Die visuelle Präsentation von Objekten in einer VRML-Welt folgt einem Modell zur Nachahmung der physikalischen Eigenschaften des Lichts.
  - Das VRML-Beleuchtungs-Modell beschreibt, wie Charakteristika der Erscheinung und das Licht in der virtuellen Welt kombiniert werden, um die angezeigten Farben zu erzeugen.

3.4 VRML: Browser

- Der VRML-Browser ist normalerweise ein Plug-in, welches von Internet-Browsern aufgerufen wird, um auf Webseiten eingebettete 3D-Inhalte anzuzeigen (Cosmo Player, Cortona VRML Client...).

- Der Browser kann bestimmte Navigations-Paradigmen (z.B. ***Walk*** oder ***Fly***) definieren, die es dem Benutzer ermöglichen, durch die virtuelle Welt zu navigieren.

- Der Browser kann dem Benutzer die Interaktion mit der Welt durch Sensor-Knoten ermöglichen, welche auf Interaktion mit geometrischen Objekten in der Welt, mit Bewegung durch die Welt oder mit Verstreichen von Zeit reagieren z.B.:
  - TimeSensor
  - TouchSensor
  - ProximitySensor

- Der VRML-Browser besteht aus drei Hauptbestandteilen:
  - Parser
  - Szene-Graph
  - Audio-visuelle Präsentation

3.4.1 VRML: Parser

- Die Parserkomponente liest eine VRML-Datei ein und kreiert einen Szene-Graphen.

3.4.2 VRML: Szene-Graph

- Der Szene-Graph besteht aus
  - Knoten, welche die Objekte und ihre Eigenschaften beschreiben. Jedes VRML-Objekt ist also ein Knoten in einer hierarchischen Baumstruktur. Verändert man die Position des Wurzelknotens, ändert sich die Position der darunterliegenden Knoten entsprechend.
    (Bsp.: Hat man z.B. ein Auto als Wurzelknoten und darunter die vier ausgerichteten Räder des Autos, ändern sich die Positionen der Räder entsprechend mit, wenn man das Auto bewegt.)

3.4.3 VRML: Audio-visuelle Präsentation

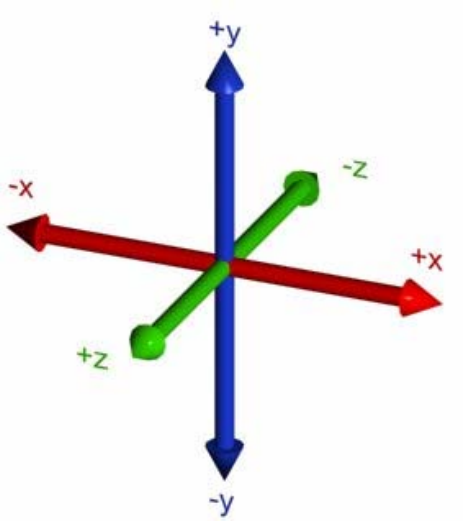
- Die **Browser-Komponente zur audio-visuellen Präsentation** übernimmt die Ausgabe des Szenegraphen.

3.5 Das Koordinatensystem

- Um ein Programm in VRML zu entwickeln, reicht es nicht aus, die einzelnen Sprachelemente zu kennen. Da VRML im Grunde eine Modellierungssprache für dreidimensionale Objekte ist, besteht die Aufgabe des Programmierers im Wesentlichen darin, räumliche Objekte zu definieren und sie in einem dreidimensionalen Raum anzuordnen. Dazu werden Koordinaten und ein Koordinatensystem benötigt

- VRML verwendet zur Definition und Darstellung von 3D-Objekten ein rechtsdrehendes, kartesisches Koordinatensystem. Rechsdrehend bedeutet, dass die Drehung entgegen des Uhrzeigersinns erfolgt.
  In einem kartesischen Koordinatensystem sind alle Raumachsen Geraden in konstantem Abstand zueinander.
  Das Koordinatensystem besitzt drei Raumachsen, x, y und z. Sämtliche Koordinatenangaben erfolgen in dieser Reihenfolge, wobei
  - **x die Breite**
  - **y die Höhe**
  - **z die Tiefe**

  des virtuellen Raums oder eines Objekts definieren.



3.6 VRML: Gestaltknoten

- Die Grundbausteine einer virtuellen Welt sind Objekte (oder Körper), die wie in der realen Welt eine Gestalt (oder Form) haben und aus einem bestimmten Material bestehen.

3.6.1 Formen

- Damit der Programmierer nicht jedes Objekt aus einzelnen Polygonen zusammenbauen muss, stellt VRML insgesamt vier einfache geometrische Grundformen zur Verfügung:
  - Quader
  - Zylinder
  - Kegel
  - Kugel
- Diese geometrischen Primitive sind Objekte (oder Körper), die wie in der realen Welt eine Gestalt (oder Form) haben und aus einem bestimmten Material bestehen.
- Vordefinierte Grundformen reichen in der Regel aus, um eine Vielzahl von Objekten in einer virtuellen Welt zu modellieren. Die geometrischen Primitive werden dabei wie virtuelle Bausteine aus einem Baukasten mit 4 Grundformen verwendet.

- Für andere einfache Geometrien wie ein Prisma, oder für komplexe Formen wie beispielsweise gewölbte Oberflächen reichen die vier Grundformen in der Regel nicht mehr aus.

- In VRML gibt es daher Knoten, mit denen im Raum Punkte, Linien und Flächen (Polygone) definiert werden können. Aus diesen Elementen können dann beliebige Strukturen zusammen gebaut werden (z.B. **IndexedFaceSet**).

- Prinzipiell kann in der Polygondarstellung jede beliebig geformte Oberfläche modelliert werden. Der Nachteil dieser Methode ist jedoch, dass mit zunehmender Detailliertheit einer Oberfläche der Modellierungsaufwand wächst und mit einer wachsenden Anzahl von Polygonen die Performance des VRML-Browsers deutlich abnimmt.

3.6.2 Materialeigenschaften

- Material- und Oberflächenbeschaffenheit sind weitere wichtige Eigenschaften eines Objekts. Die Materialeigenschaften spiegeln sich als Oberflächeneigenschaften wie Farbe und Textur, kurz in der Erscheinung wider.

- Farben werden in VRML in der RGB-Notation angegeben. Durch eine additive Mischung der drei Farbkomponenten und die Variation der Farbintensität kann jeder beliebige Farbton erzeugt werden.

- Texturen sind statische (GIF, JPEG, PNG) oder dynamische Bilder (MPEG), die auf die Oberfläche eines räumlichen Objekts aufgebracht werden, um strukturelle Details vorzutäuschen.

3.8 VRML: Sensoren

- Sensoren erlauben es VRML-Objekten und -Szenen, das Verstreichen von Zeit und Benutzer-Aktivitäten wahrzunehmen und zu reagieren.
  - **TimeSensor** kann dazu benutzt werden, Objekte nach einer bestimmten Zeitspanne zu aktivieren.
  - **TouchSensor** erlaubt die Reaktion auf Maus-Events, wie Klicken, Bewegung oder Ziehen.
  - **ProximitySensor** erkennen Objekte in einer bestimmten Entfernung und können benutzt werden, Entfernungssensitive Inhalte zu schaffen (z.B. könnte eine Animation starten, wenn sich der Betrachter einem Objekt der virtuellen Welt nähert).

3.9 VRML. Animationen

- Ein Film setzt sich aus einer Folge von statischen Bildern zusammen.

- Eine Animation entsteht dadurch, dass man die Einzelbilder schnell genug hintereinander abspielt. Bei ungefähr 30 Bildern pro Sekunde nimmt der Mensch keine Einzelbilder mehr wahr und es entsteht die Illusion eines flüssigen Bewegungsablaufes

- Um Objekte zu animieren, wird in VRML ein ähnliches Verfahren eingesetzt. Im Prinzip müssten Sie im Quelltext jede kleinste Änderung einer Position, einer Drehung oder eines Farbwertes definieren und anschließend den VRML-Browser anweisen, nacheinander jede einzelne Einstellung darzustellen.

- Stellt man sich eine Animation von 10 Sekunden Dauer mit 30 Einstellungen pro Sekunde vor, so sind das immerhin schon 300 Positionen, Winkel oder Farben, die man berechnen müsste. Glücklicherweise übernimmt einen Großteil dieser Arbeit der VRML-Browser mit Hilfe eines Verfahrens, das als **Keyframe-Animation** bezeichnet wird.

- **Keyframe-Animation** heißt, dass jede kontinuierliche Objektveränderung, sei es durch Änderung der Form, der Position oder durch eine Änderung der Materialeigenschaften, nur durch einige wenige Schlüsseleinstellungen (Keyframes) beschrieben wird. Alle anderen Parameter, die der VRML-Browser benötigt, um Veränderungen nicht abrupt, sondern langsam als Animation ablaufen zu lassen, werden automatisch durch den VRML-Browser berechnet.

- Das erspart nicht nur eine Menge Programmierarbeit, sondern erhöht auch deutlich die **Abarbeitungsgeschwindigkeit**. Die Abbildung auf der nächsten Seite zeigt schematisch am Beispiel der Bewegung eines Objektes in einer Ebene, wie man sich die Keyframe-Animation vorstellen kann.

- Wir stellen uns vor, dass ein Objekt auf verschiedenen Wegen von Punkt 1 nach Punkt 2 bewegt werden soll:
  - Zuerst betrachten wir den direkten Weg von Punkt 1 nach Punkt 2. Dafür brauchen wir nur den Anfangs- und Endpunkt anzugeben. Alle anderen Einstellungen werden unabhängig davon, wie weit Start- und Endpunkt voneinander entfernt sind, durch eine **lineare Interpolation** automatisch berechnet.
  - Anders verhält es sich, wenn wir das Objekt von Punkt 1 über Punkt 2 nach Punkt 3 bewegen wollen. Jetzt müssen wir eine zusätzliche Schlüsselposition in Punkt 2 angeben, da die Bewegung nicht mehr geradlinig verläuft, sondern das Objekt in Punkt 2 die Bewegungsrichtung ändert.
  - Alle anderen Positionen zwischen Punkt 1 und Punkt 2 sowie zwischen Punkt 2 und Punkt 3 kann der Browser wieder automatisch interpolieren, da auf diesen Strecken keine Abweichung von der Geraden auftritt.

- In VRML gibt es spezielle Knoten, die diese Berechnungen ausführen, die **Interpolatoren**.

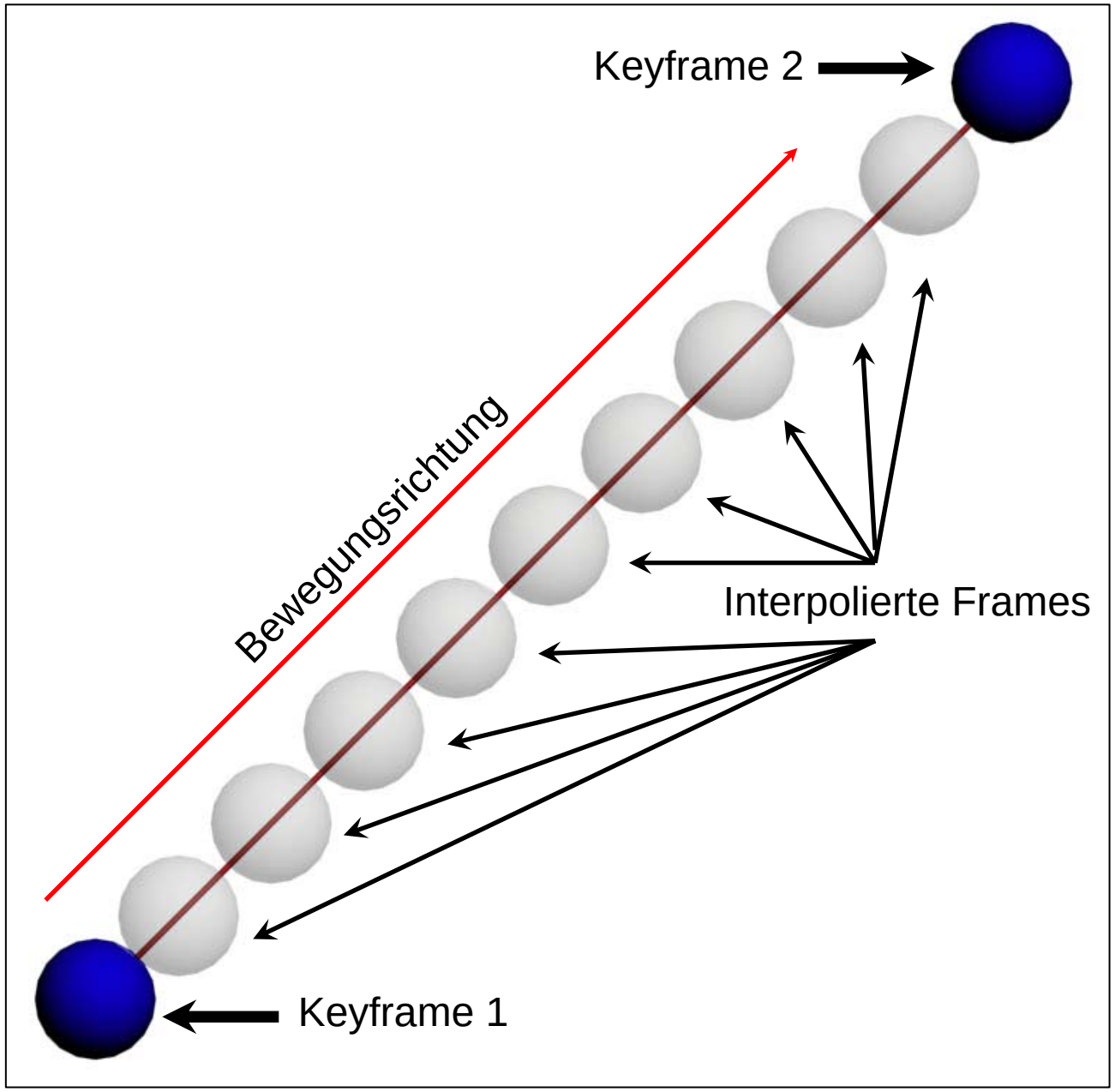
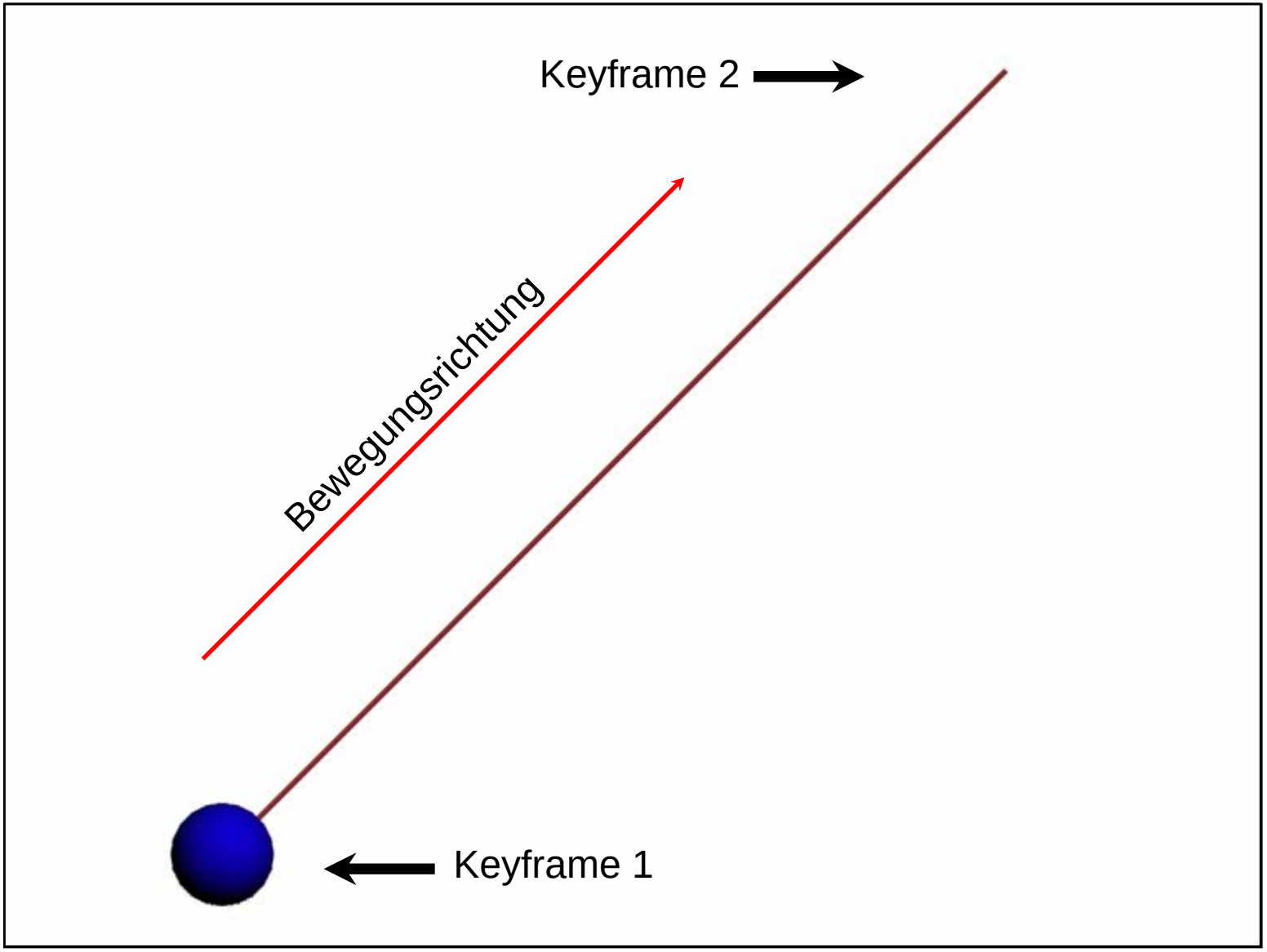
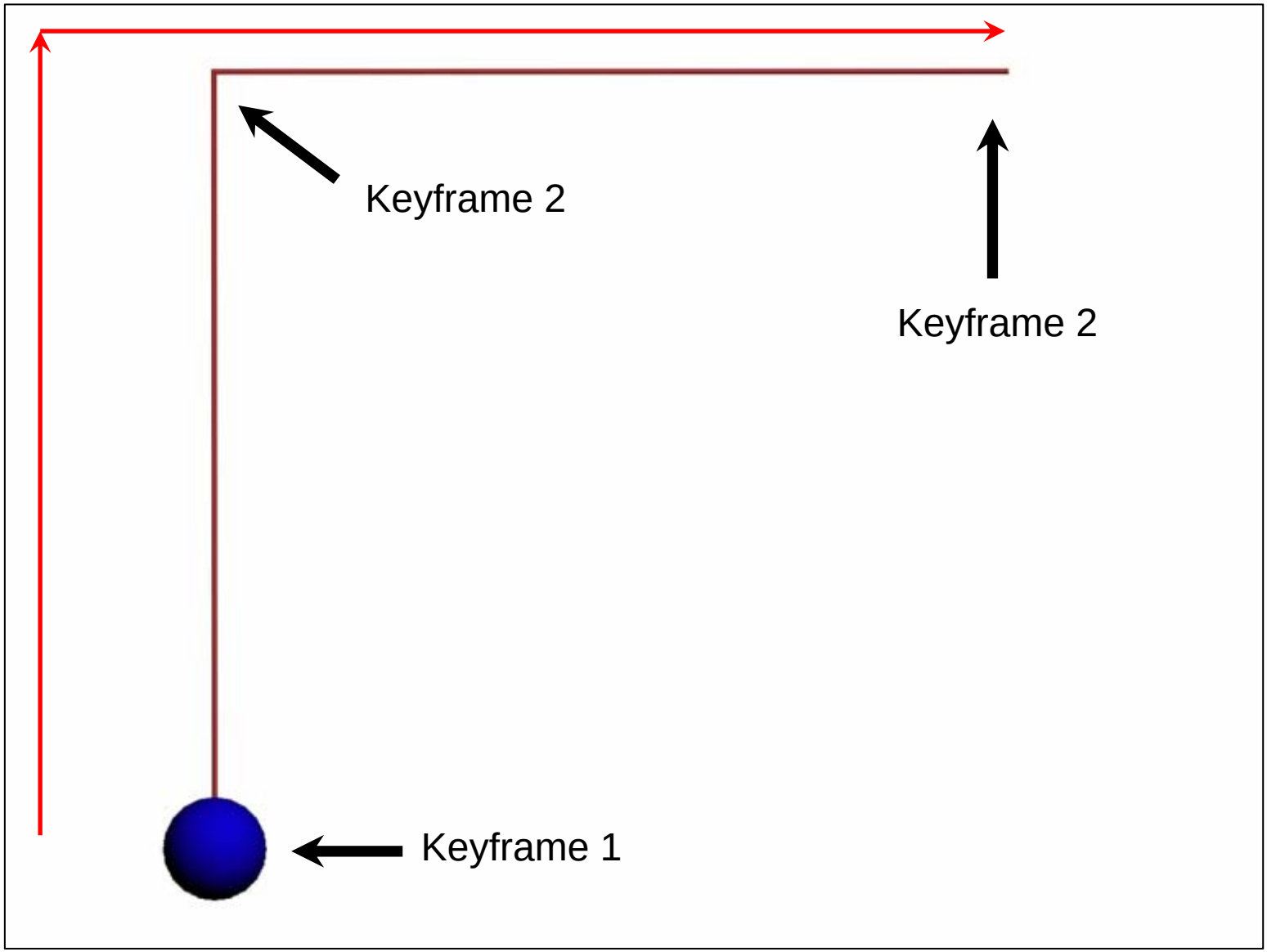


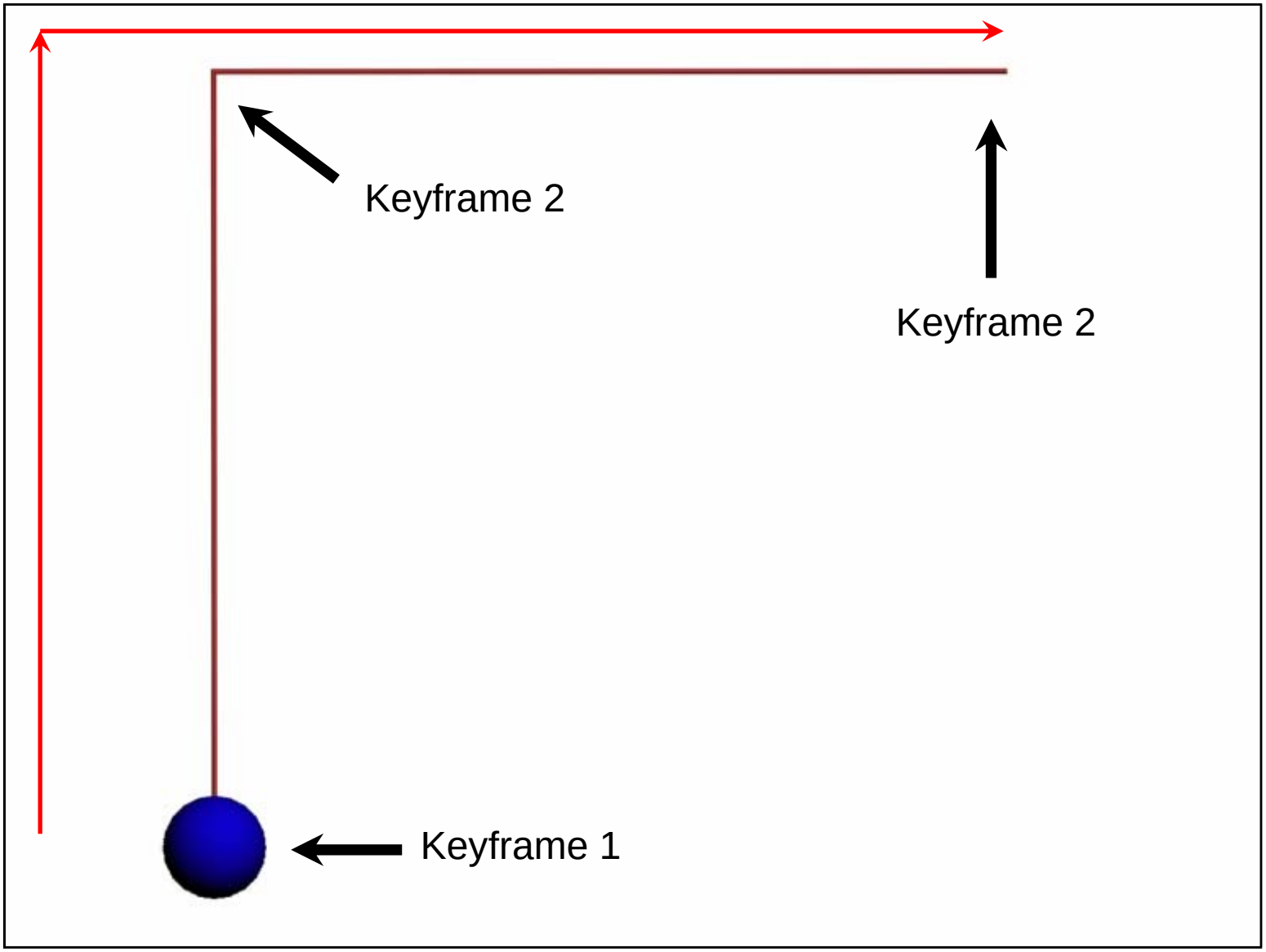
Abb.: Lineare Interpolation zwischen zwei Keyframes



Film: Lineare Interpolation zwischen zwei Keyframes



Film: Interpolation zwischen mehreren Keyframes



Film: Interpolation zwischen mehreren Keyframes

Aufgaben

1) **Wiederholen** Sie den Stoff dieser Sitzung **bis zur nächsten Sitzung** (siehe dazu den Link zur Sitzung auf der HKI-Homepage).
   Informieren Sie sich zusätzlich durch eigene Literaturrecherche!

2) Beantworten Sie die Fragen aus der Sammlung **„beispielhafte Klausurfragen“** zum Bereich **3D / VR** (soweit in dieser Sitzung behandelt).