



# **VEKTORIA-MANUAL TEXTURIERUNG**



# Inhalt



/// **UEBERSICHT**

/// **OBERFLAECHEN**

/// **BILDER**

/// **TEXTUREN**

/// **MATERIALIEN**



# Knotenobjekte der Szeneraphen



## Virtuelle Szene

### Gruppenknoten

- Gruppenbehälter
- Transformationen
- Switchknoten (Level-of-Detail, Animation etc.)
- ...

### Blattknoten

- Lichter
- Geometrien
- Kameras
- Materialien, Texturen, Images
- Farben
- ...

## Reale Szene

### Gruppenknoten

- Gruppenbehälter
- Computer
- Kanäle
- Switchknoten (an, aus)
- ...

### Blattknoten

- Eingabegeräte
- Sichtsysteme
- Sichtfenster
- Soundgeräte
- Basis-Render-API
- ...



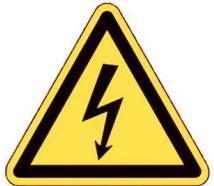
## Kapitel 2

# Kapitel 2



/// **OVERFLAECHEN**





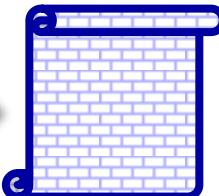
Überblick über Knotenobjekte für Oberflächen

# Unterscheiden Sie bei Oberflächen!

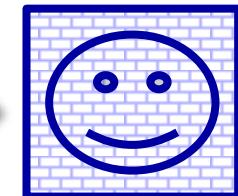
## Material



## Textur



## Bild



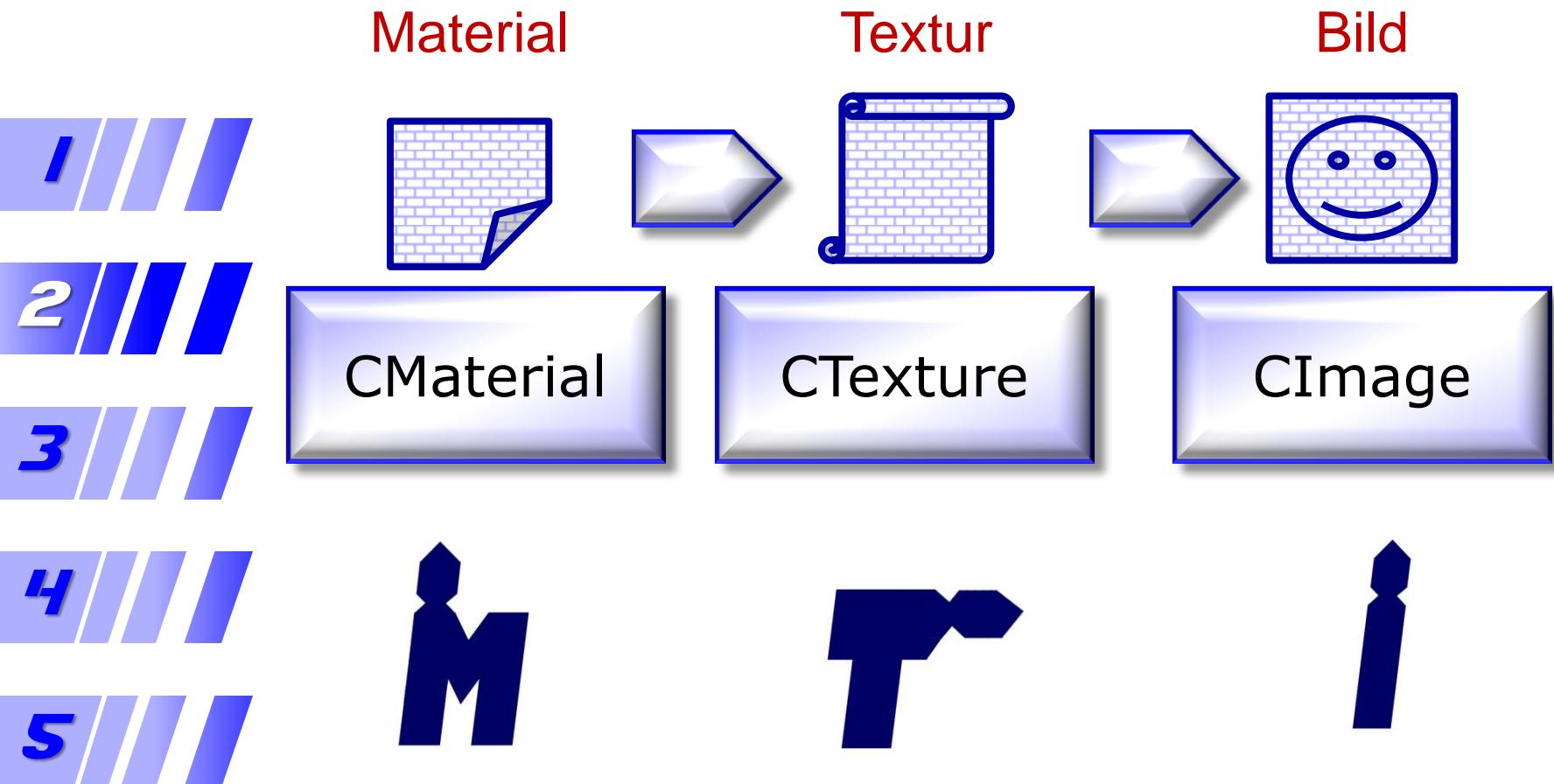
1 // / / / Ein Material wird auf eine Geometrie aufgebracht (ge- mappt). Es kann mehrere Texturen enthalten. Zusätzlich hat es eventuell physikalische Parameter (Rollwiderstand, etc.)

2 // / / Eine Textur hat optische Parameter und eventuell einen Verweis auf ein Bild. Es gibt verschiedene Arten von Texturen (Image, Glow, Bump, etc. )

3 // / / Ein Bild enthält die eigentlichen 2D-Pixeldaten. Es hat einen Verweis auf eine JPG-, BMP-... oder TGA- Datei.



# Oberflächenklassen in Vektoria



## Kapitel 3

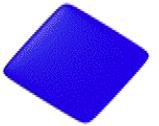
# Kapitel 3



1 // / / /

2 // / / /

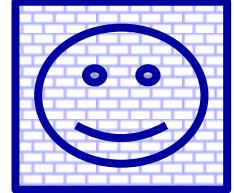
/// **BILDER**



4 // / / /

5 // / / /





Pixelbilder

# Images

Ein Bild enthält 2D-Pixeldaten.

Es hat einen Verweis auf eine Datei.

Es gibt verschiedene Bilddateiformate.



Je nach GameEngine/Szenegraf sind  
andere Formate ladbar, oft möglich sind  
z.B.:

- JPG
- BMP
- TGA
- PNG
- GIF

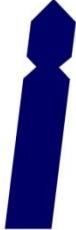
Auch die Bittiefe (8 bit, 24 bit, 32 bit) und  
die Kanalanzahl müssen beachtet werden!





Pixelbilder

# Climage



## Initialisierungs- und Finalisierungsmethode:

void Init(char \* acPath);

acPath gibt den kompletten Pfad der Textur an (inklusive Dateisuffix).

Es sind absolute und relative Pfade möglich.

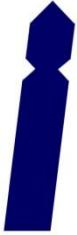
void Fini();





Pixelbilder

# Climage



Vektoria kann zurzeit folgende Dateiformate lesen:

- JPG, JPEG
- JPE,
- BMP & DIB
- TGA
- TIF, TIFF
- PNG
- GIF

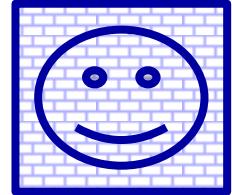
Jeweils monochromatisch, trichromatisch (RGB) und  
trirochromatisch mit Alpha (RGBA)  
Auch verschiedene Bitraten (8 bit, 24 bit, 32 bit) sind  
möglich





Pixelbilder

# Images



Falls Sie nicht nur mit Vektoria arbeiten, sondern auch mit einem anderen Szenegrafen bzw. einer anderen Game Engine, müssen Sie bei der Auswahl der Bilddateien aufpassen:

- Viele Game Engines können keine komprimierten Bilddateiformate wie JPG, JPEG, PNG oder GIF verarbeiten.
- Die meisten tun sich bei der Verarbeitung von BMP schwer.
- Auch bei der Annahme der Bittiefe (2, 8, 16, 24, 32 bit) und der Kanalanzahl gibt es beträchtliche Unterschiede.
- Dateien mit Alphakanälen werden nicht von allen Game Engines/Szenegrafen verarbeitet.
- Einige akzeptieren nur Bilddateien, deren Pixelkantenanzahl einer Zweierpotenz entspricht (z.B. 64\*64 Pixel)



## Kapitel 4

# Kapitel 4

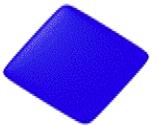


1 // / / /

2 // / / /

3 // / / /

/// TEXTUREN

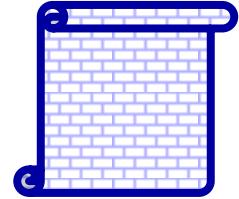


5 // / / /



# Knotenobjekte der Szeneraphen

## Texturen allgemein

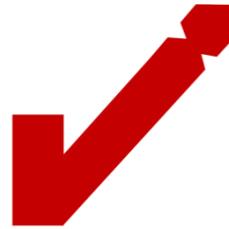


Synonyme in anderen Szeneraphen: Skins, Texsurfaces

Arten:

- Zielort: Surface, Background ...
- Herkunft: Prozedurale vs. Image Texturen
- Anzahl: Single-, Dual-, Multitextures
- Interpretation: Farbgebung (i.d.R.), Dot3-Bumpmap, Lightmap, Shadowmap, Vertex Displacement Map ...
- Animation: Videotexturen, Gif-Texturen, statische Texturen
- Transparenz: Alpha-Texturen, Chromakeying-Texturen, opake Texturen
- Granularitätsstufen: Mipmapped (mit und ohne Blending)
- Aufbringungsweise: clamped, tiled, scaled, ...
- Dimension: 2D, 3D





# CMaterial: Die Materialklasse in Vektoria

Folgende Texturarten sind in Vektoria möglich:



- Diffuse

Farbe und Helligkeit auf der beleuchteten Seite

- Glow

Farbe und Helligkeit auf der unbeleuchteten Seite

- Specular

Stärke des Glanzlichtes (Highlight)

- Bump

Normalenwerte für das DOT3-Bumpmapping

- Height

Höhentextur für Parallax Occlusion-Mapping

- Environment

Reflexions-Umgebungstextur

- Thickness

Dickentextur

- Sky

Spezielle Textur für Skydomes u.ä.

- Sprite

Spezielle Textur für 2D-Sprites (Overlays, Writings  
Wribels, Billboards und Backgrounds)





# CMaterial: Die Materialklasse in Vektoria

## Die Textursetzmethoden der Klasse CMaterial:

1 // / / /  
void SetTextureDiffuse(CTexture \*ptexture);  
void SetTextureGlow(CTexture \*ptexture);  
void SetTextureSpecular(CTexture \*ptexture);  
void SetTextureBump(CTexture \*ptexture);  
void SetTextureHeight(CTexture \*ptexture);  
void SetTextureEnvironment(CTexture \*ptexture);  
void SetTextureThickness(CTexture \*ptexture);  
void SetTextureSky(CTexture \*ptexture);  
void SetTextureSprite(CTexture \*ptexture);

2 // / / /  
3 // / / /  
4 // / / /  
5 // / / /

 **ptexture** ist ein Zeiger  
auf ein Texturobjekt



# CMaterial: Schnelltexturierungsmethoden

## Die Schnelltexturierungsmethoden:

```
CImage * MakeTextureDiffuse (char * acPath);  
CImage * MakeTextureGlow(char * acPath);  
CImage * MakeTextureSpecular(char * acPath);  
CImage * MakeTextureBump(char * acPath);  
CImage * MakeTextureHeight(char * acPath);  
CImage * MakeTextureEnvironment(char * acPath);  
CImage * MakeTextureThickness(char * acPath);  
CImage * MakeTextureSky(char * acPath);  
CImage * MakeTextureSprite(char * acPath);
```

5 Es wird ein Zeiger auf das erzeugte Image-Objekt zurückgegeben.

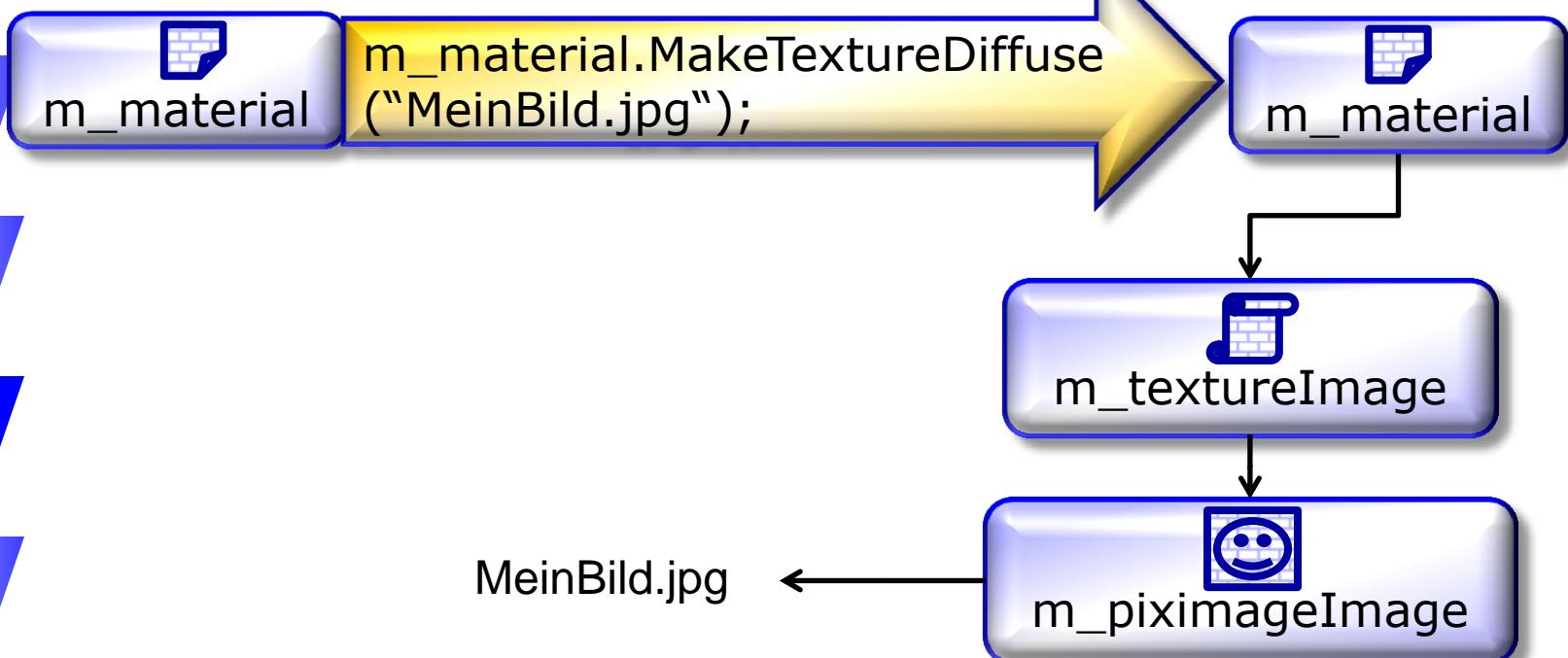
acPath ist ein Pfad auf die Bilddatei. Es sind absolute und relative Pfade möglich. Achtung! Das Dateisuffix muss mit angegeben werden!



# CMaterial: Die Materialklasse in Vektoria



Wie funktionieren die Schnelltexturierungsmethoden? –  
Beispiel:





Materialien

# Übung zu Materialien



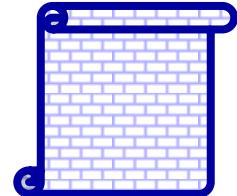
Erzeugen Sie ein Material mit einer beliebigen Diffuse-Textur! Mappen Sie das Material auf eine Kugel!

Freiwillige Zusatzaufgabe für die schnellen Nerds:

- Erzeugen Sie zusätzlich noch eine Glow-Map!



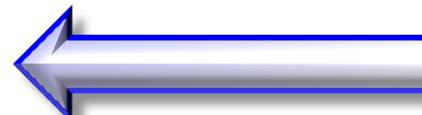
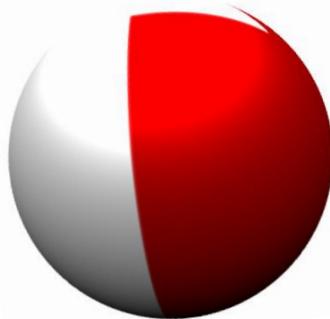
# Texturmappingmethoden



Texturabschneidung  
(texture clamping)



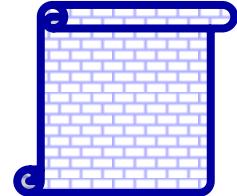
Texturkachelung  
(texture tiling)



Texturskalierung  
(texture scaling)



# Bumpmapping



Dot3-Bumpmapping erzeugt eine Oberflächenreliefanmutung durch Modifikation der Normalenvektoren, eine Szene wirkt dadurch sofort komplexer und realistischer, aber es bietet...



...keine räumliche Verschiebung bei Kameraanimationen.



...keine Selbstokklusion (Verdeckung hinterer Strukturen).



...keine Silhouettenbildung.



...keine Vesikelbildung (biologische Abspaltung).



Abhilfe würde das Quaoaring-Bumpmapping liefern, das 2001 an der Universität Frankfurt entwickelt wurde,



es ist bisher allerdings in keiner einzigen Engine implementiert!



Zumindest die ersten drei Mankos kann Displacement Mapping vermeiden.



Vektoria bietet jedoch kein Displacement Mapping an,

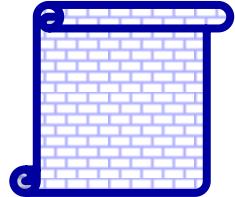


aber bald!



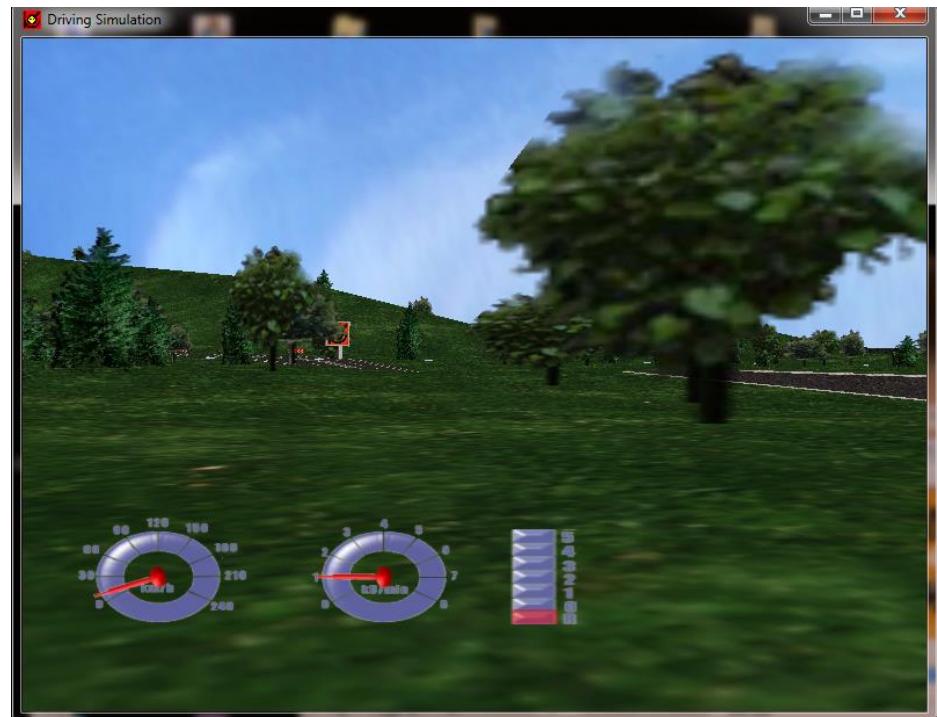
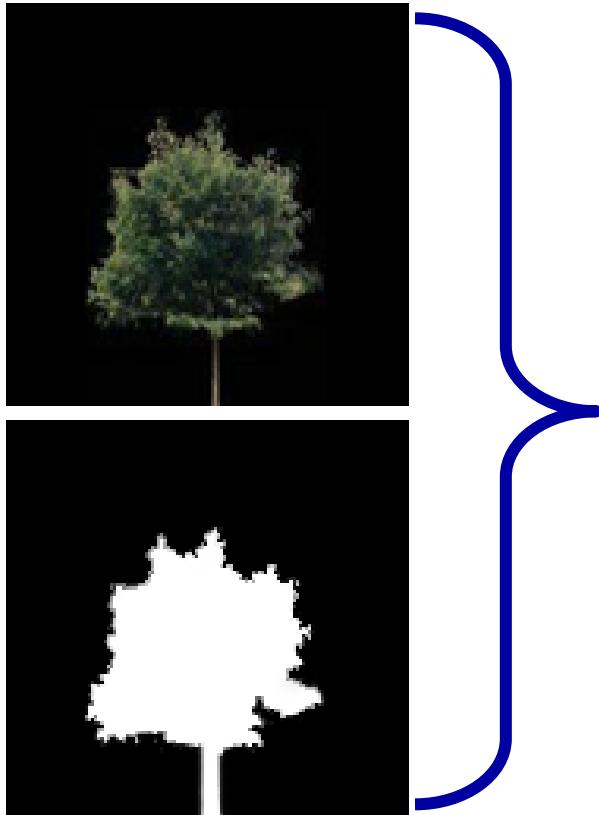
## Texturen

# Texturen mit Alphakanal



■ Zusätzlicher Alphakanal (zu Rot-, Grün- und Blaukanal)  
gibt Transparenz an

- 1 // / / /
- 2 // / / /
- 3 // / / /
- 4 // / / /
- 5 // / / /





Materialien und Texturen

## Material mit Alphakanaltextur



Um eine Alpha-Imagetextur anzuzeigen, reicht es nicht, dem Material mit Hilfe der Methode **MakeTextureDiffuse** den Pfad einer Textur mit Alphakanal zu übergeben!



Es muss zusätzlich der Alphakanal mit der Methode **SetTransparencyOn** eingeschaltet werden!





# Material mit Alphakanaltextur

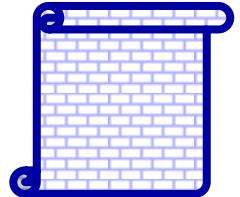
Soll ein Material mit einer gleichmäßigen Transparenz angezeigt werden, wird keine Aplhatextur benötigt,

Es kann der Alphawert direkt mit

`SetTransparency(float frTransparency)` angegeben werden. Der fraktionale Parameter sollte zwischen 0.0 und 1.0 liegen. Dabei gilt z.B.:

```
SetTransparency(0.0f); // opak (total undurchsichtig)  
SetTransparency(0.1f); // wenig transparent (10%)  
SetTransparency(0.3f); // wenig transparent (30%)  
SetTransparency(0.5f); // halbtransparent (50%)  
SetTransparency(0.8f); // sehr transparent (80%)  
SetTransparency(1.0f); // total unsichtbar (sinnlos ;-))
```





# Texturen mit Farbschlüssel

Chromatischer Schlüssel (hier: Magenta) kann ebenfalls Transparenz anzeigen

- Vorteil: Kein zusätzlicher Alphakanal notwendig
- Nachteil: Keine Semitransparenz möglich





Materialien und Texturen

## Material mit Farbschlüssel



Um ein Material mit einer Farbschlüsseltextratur zu erzeugen, muss einfach nur einmal beim Material die Methode

`SetChromaKeyingOn()`  
aufgerufen werden

Das linke obere Pixel wird dann automatisch als Farbschlüssel (engl. chroma key) genommen.

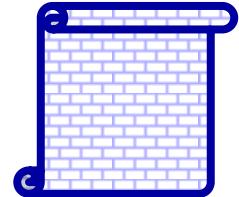


Achtung! Nur Pixel, deren RGB-Werte exakt dem Farbschlüssel entsprechen, werden transparent angezeigt.



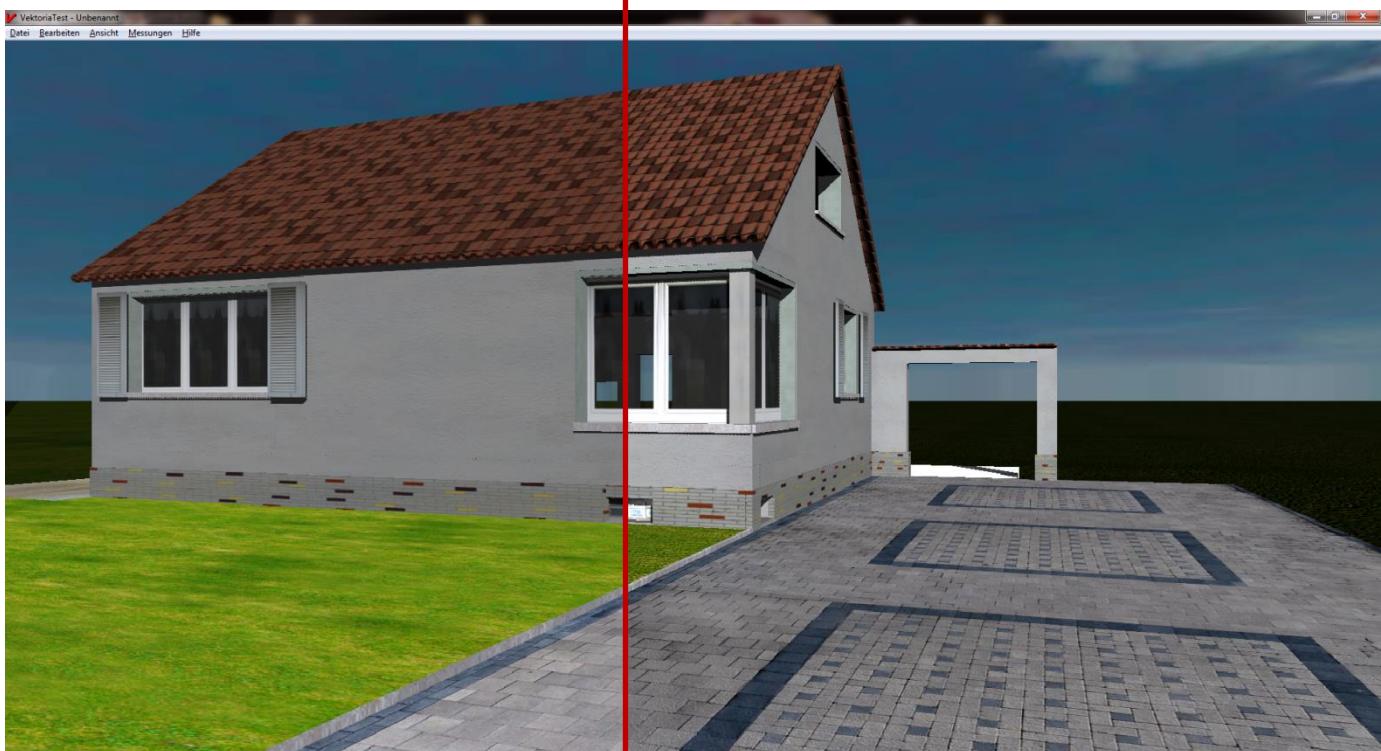
Texturen

# Dot3-Bumpmapping



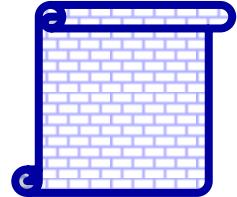
ohne Bumpmapping

mit Bumpmapping



# Texturen

# Bumpmapping



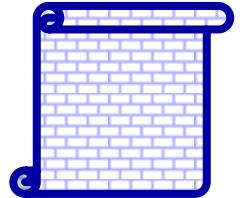
Beim DOT3-Bumpmapping werden die Normalenvektoren mit Hilfe des Vektorproduktes (engl. dot 3 product) „verbogen“.

Dazu werde die RGB-Werte der Bumpmapping-Textur folgendermaßen uminterpretiert:

- Rot: Skalar in Richtung der Oberflächentangente
  - Grün: Skalar in Richtung der Oberflächenbitangente
  - Blau: Skalar in Richtung der Oberflächennormale

Da meist in Richtung der Oberflächennormalen verschoben wird, sehen DOT3-Bumpmaptexturen bläulich aus.

Texturen



# Dot3-Bumpmapping

1 // / / /



2 // / / /



3 // / / /



4 // / / /

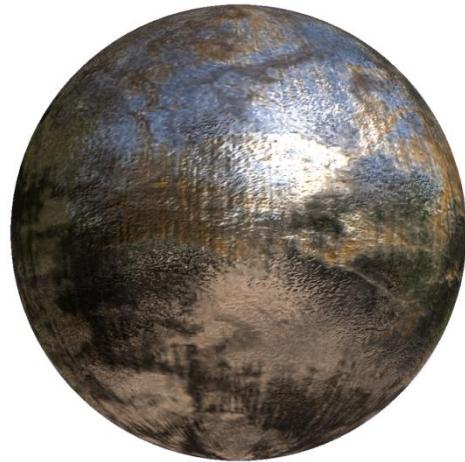
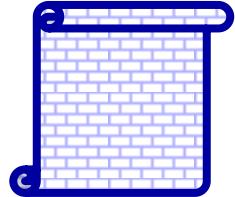


5 // / / /



✓ Texturen

# Environmental & Bump



3 Rostige  
Christbaumkugel



Goldplatten-  
Kugel



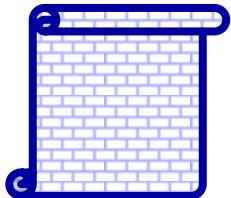
Gluckernde  
Wasserkugel

Specular-  
Environment-  
Bump- und  
Diffuse-Mapping

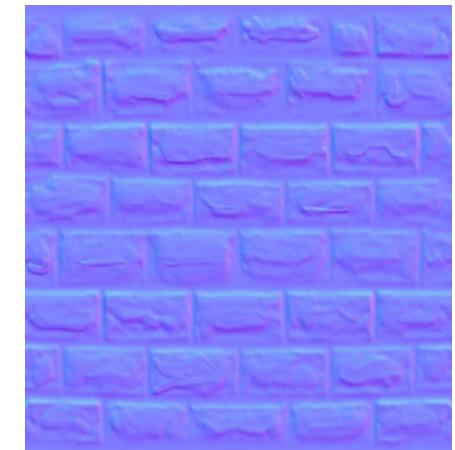
Specular-  
Environment-  
Refraction- und  
Bump-Mapping  
sowie SetAniOn



Texturen



# Erzeugung Bumpmap-Textur



1 // / / /  
2 // / / /  
3 // / / /  
4 // / / /  
5 // / / /

Image-  
Textur

Highmap-  
Textur

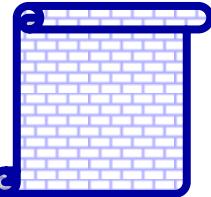
Bumpmap-  
Textur

Pixelbearbeitungs-  
programm

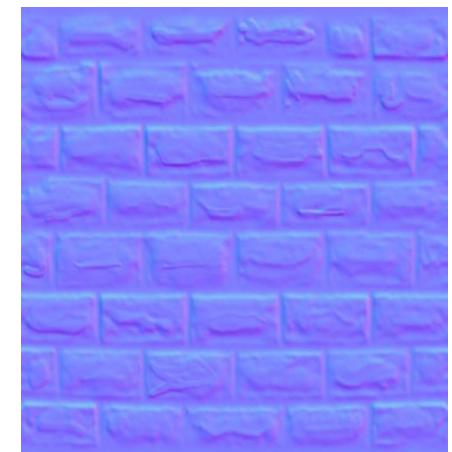
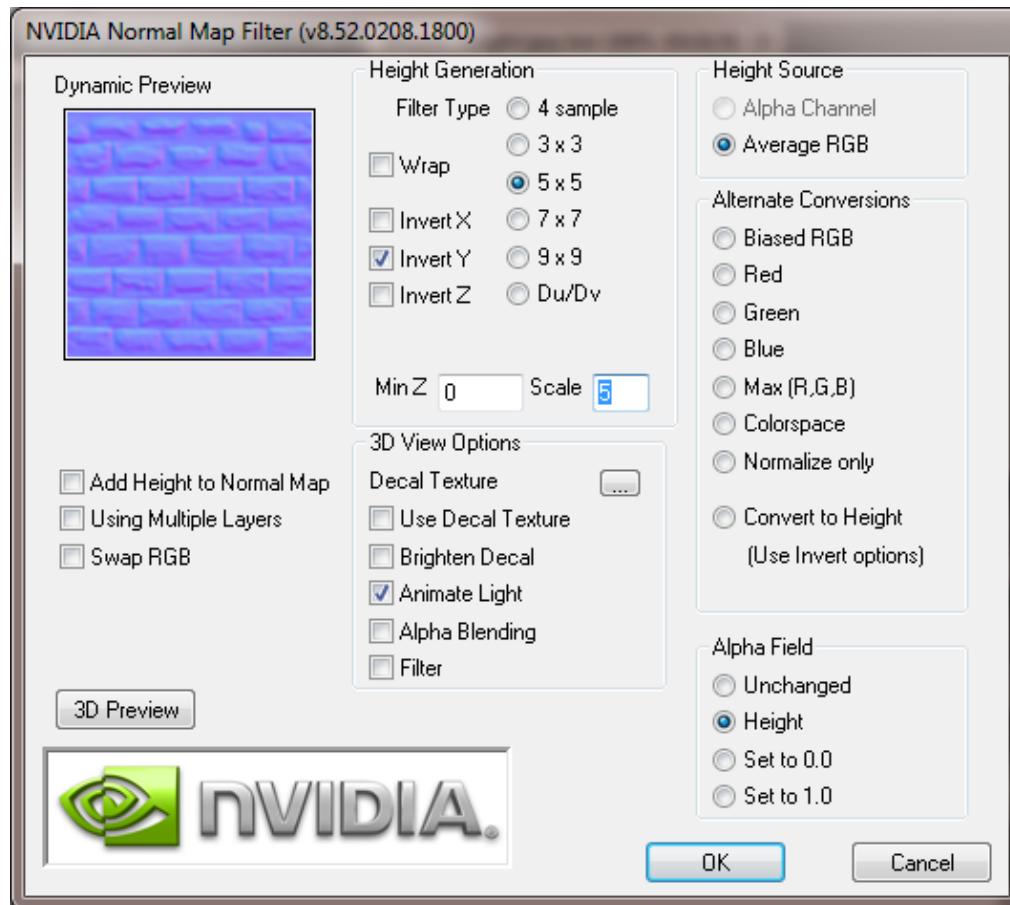
Normalmap-  
Filter



Texturen



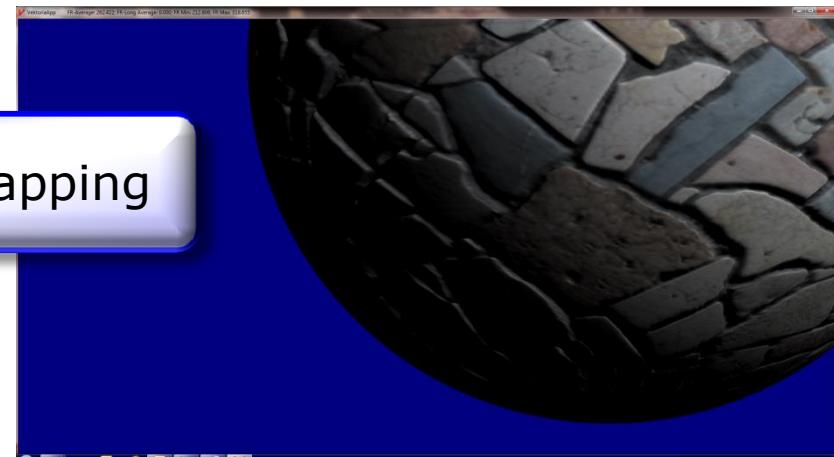
# Einstellungen Normal Map Filter



# Parallax Occlusion Mapping (POM)

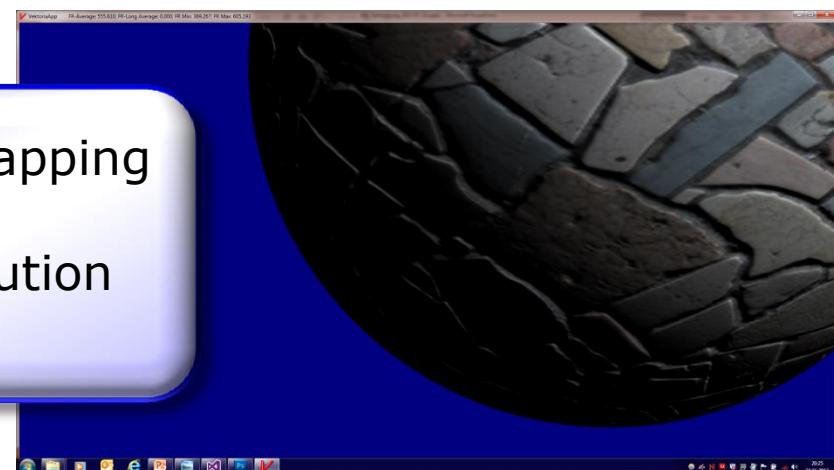
1 // / / /

DOT3-Bumpmapping



2 // / / /

Parallax Occlusion Mapping  
kann die Realistik von  
Bumpmapping noch  
erhöhen.



3 // / / /

DOT3-Bumpmapping  
&  
Parallax Occlusion  
Mapping

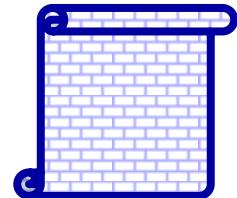
4 // / / /

5 // / / /



Texturen

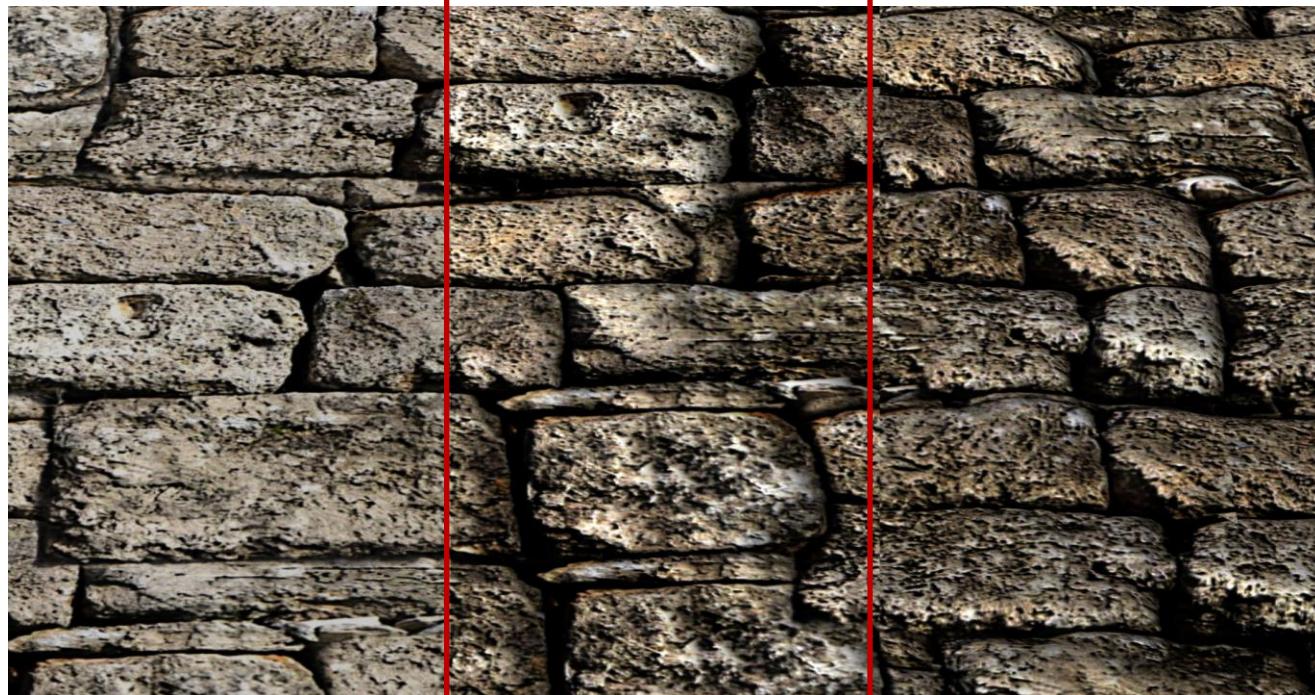
# Parallax Occlusion Mapping

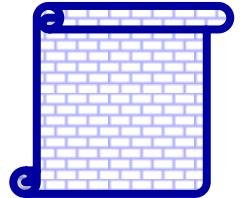


1 // / / /  
ohne  
Bumpmapping

2 // / / /  
mit  
Bumpmapping

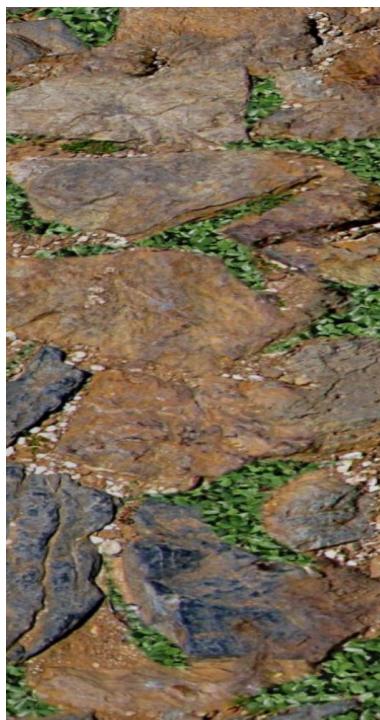
3 // / / /  
mit Bumpmapping  
& POM





# Parallax Occlusion Mapping

1 // / / /  
ohne  
Bumpmapping



2 // / / /  
mit  
Bumpmapping



3 // / / /  
mit Bumpmapping  
& POM

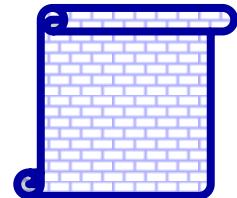


- 1 // / / /
- 2 // / / /
- 3 // / / /
- 4 // / / /
- 5 // / / /

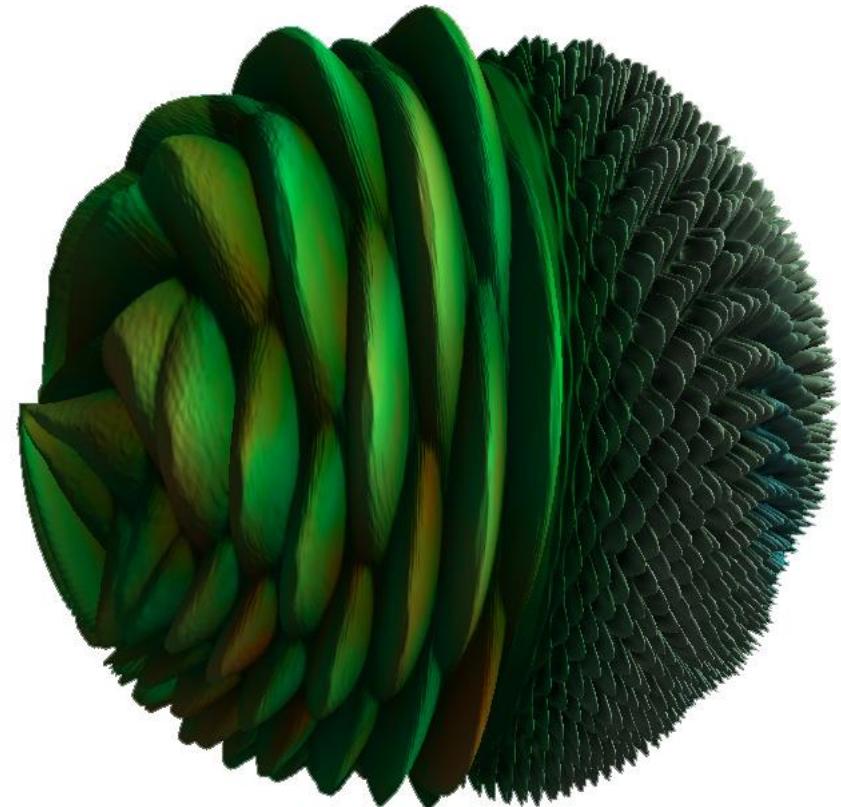


Texturen

# Quaoaring-Bumpmapping

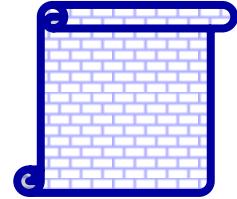


- 1 // / / /
- 2 // / / /
- 3 // / / /
- 4 // / / /
- 5 // / / /



# Knotenobjekte der Szeneraphen

## Animation von Texturen



- 1 // / / / • Videotexturen, „Animated GIF“-Texturen
- 2 // / / / • Veränderung der Ausgangsparameter bei Prozeduralen Texturen
- 3 // / / / • Texture Warping (Verändern der Texturkoordinaten)
- 4 // / / / • Texture Weighting (Verändern der Gewichte bei Multitextures über die Zeit)
- 5 // / / / • Generelle Parametermodifikation (z.B. Änderung des Höhenfaktors bei Bumpmaps oder des Transparenzwertes einer Textur über die Zeit)
- Ersetzen der Textur (bzw. Texturlink verändern)



# CTexture: Die Texturklasse in Vektoria

## Die Grundmethoden:

1 // / / /  
void Init(  
CImage \* pimage,  
int eKind = 0);

2 // / / /  
3 // / / /  
4 // / / /  
5 // / / /  
void Fini();

Initialisiert den Computer. Der Parameter eKind kann zurzeit folgende Werte annehmen:

S\_TEXTURE\_DIFFUSE  
S\_TEXTURE\_GLOW  
S\_TEXTURE\_SPECULAR  
S\_TEXTURE\_BUMP  
S\_TEXTURE\_HEIGHT  
S\_TEXTURE\_ENVIRONMENTAL

Finalisiert die Texturklasse.

## Kapitel 5

# Kapitel 5



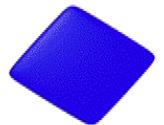
1 // / / /

2 // / / /

3 // / / /

4 // / / /

/// MATERIALIEN



PROF. DR. TOBIAS BREINER  
HS KEMPTEN

38 VON 56  
TEXTURIERUNG



# CMaterial: Die Materialklasse in Vektoria

## Die Grundmethoden:

1 // / / /  
void Init();

Initialisiert ein Material  
(optional, Konstruktor  
reicht aus)

2 // / / /  
3 // / / /  
4 // / / /  
5 // / / /  
void Fini();

Finalisiert ein Material.  
(ebenfalls optional)



Materialien, Texturen und Bilder

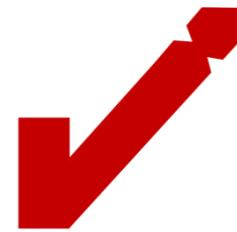
# Übung „Goldkugel“



Erzeugen Sie mittels der Schnelltexturierungs-methoden eine möglichst realistische Goldkugel aus dem Märchen „Hans im Glück“!



Freiwillige Übung für  
die schnellen Nerds:  
Lassen Sie die Gold-  
kugel spontan aufblitzen!



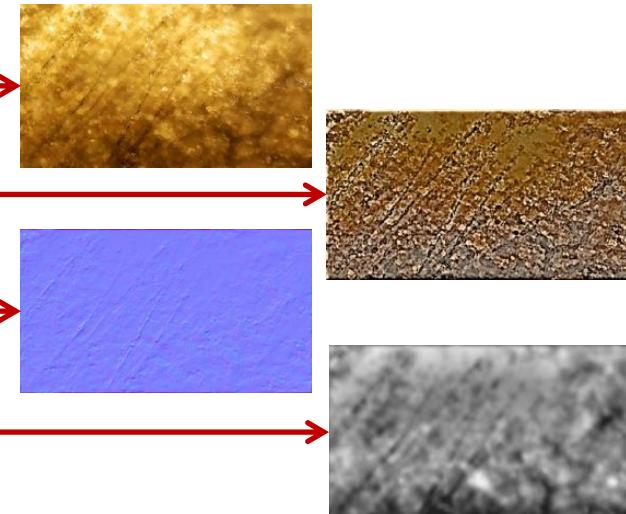
# Lösung „Goldkugel“ (1/2)



Nehmen Sie die „Hallo Kugel!“-Lösung aus dem vorherigen Foliensatz und erweitern sie diese um folgendes Material!

...

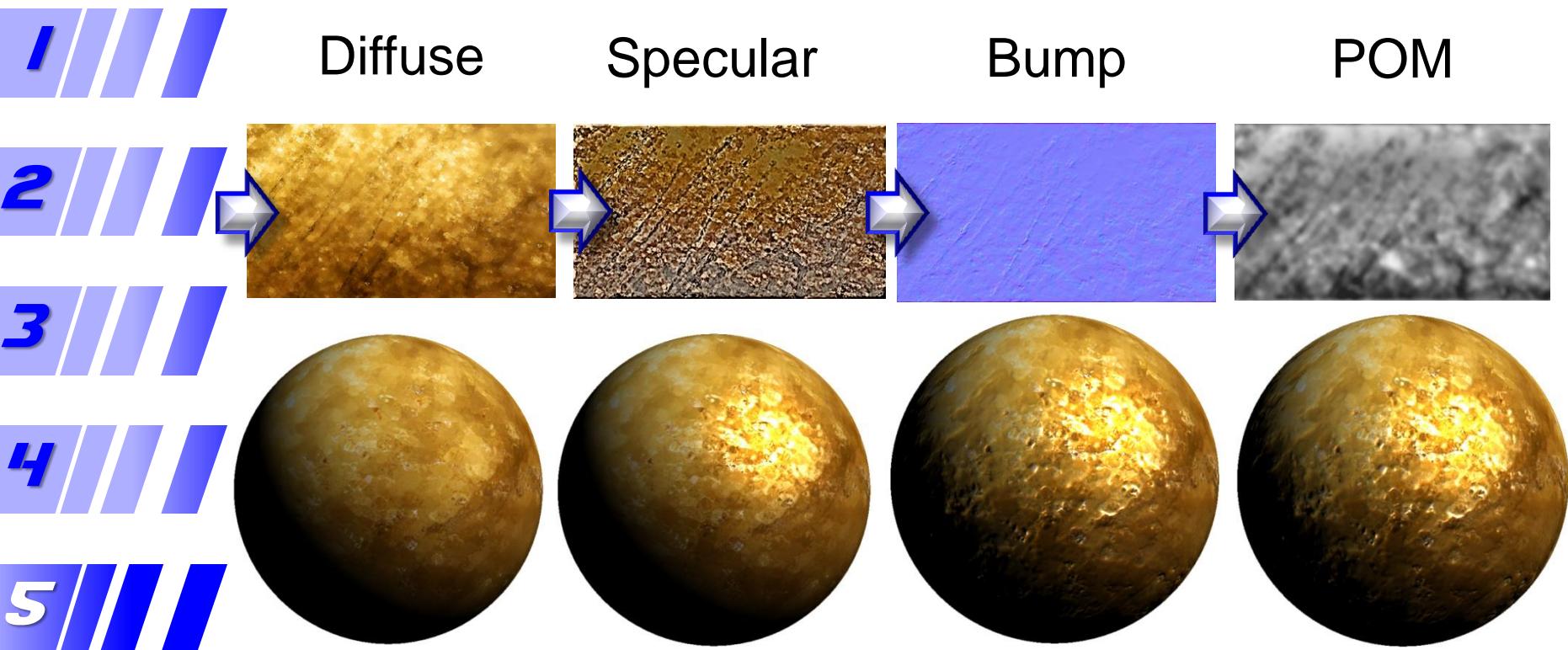
```
m_zm.MakeTextureDiffuse  
    ("gold_diffuse.jpg"); →  
m_zm.MakeTextureSpecular  
    ("gold_specular.jpg"); →  
m_zm.MakeTextureBump  
    ("gold_bump.jpg"); →  
m_zm.MakeTexturePOM  
    ("gold_pom.jpg"); →  
...
```



1 // / / /  
2 // / / /  
3 // / / /  
4 // / / /  
5 // / / /



# Lösung „Goldkugel“(2/2)





Materialien

# Übung zu Materialien



- 1 // / / /
- 2 // / / /
- 3 // / / /
- 4 // / / /
- 5 // / / /
- Erzeugen Sie mit Hilfe der Schnelltexturierungsmethoden eine Erdkugel mit
    - Diffuse Map
    - Glow Map (Städte leuchten auf Nachtseite der Erde)
    - Specular Map (Ozeane glänzen mehr als Festland)
    - Bump Map (Oberflächenrelief durch Berge und Täler)

- Freiwillige Zusatzaufgabe für die schnellen Nerds:
- Erzeugen Sie die Bump Map selbst!



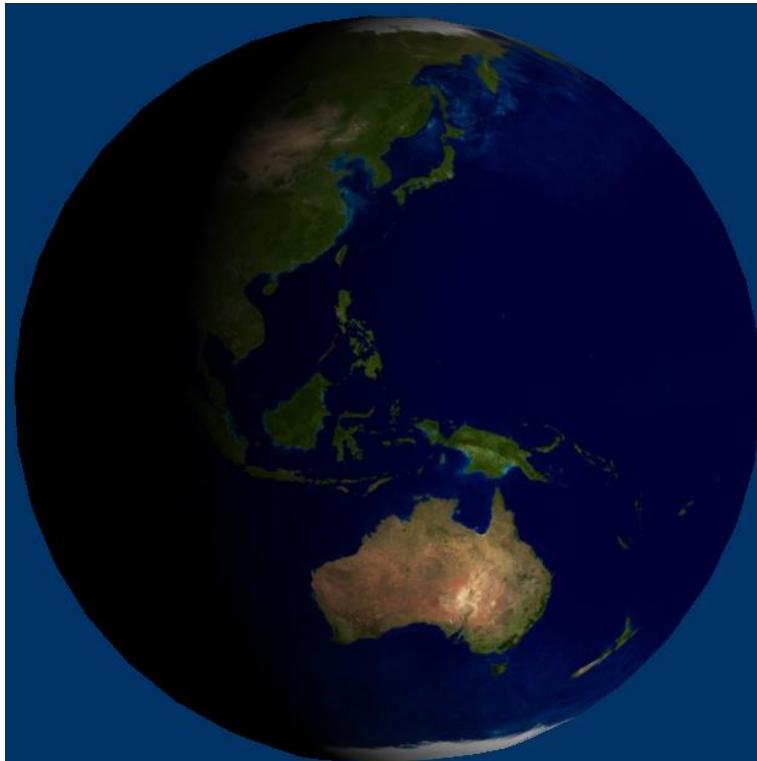


Materialien

# Lösungsausgabe



- 1 // / / /
- 2 // / / /
- 3 // / / /
- 4 // / / /
- 5 // / / /



nur Diffuse



Diffuse & Glow





Materialien

# Lösungsausgabe



- 1 // / / /
- 2 // / / /
- 3 // / / /
- 4 // / / /
- 5 // / / /



Diffuse, Glow & Specular



Diffuse, Glow, Specular & Bump





Materialien

# Lösungsausgabe



1 // / / /

2 // / / /

3 // / / /

4 // / / /

5 // / / /



Nahaufnahme des Übergangs zwischen Glow und Image



# CMaterial: Die Materialklasse in Vektoria

## Die Schaltmethoden der Klasse CMaterial:

1 // / / /  
void SetShadingOn();  
void SetShadingOff();

2 // / / /

3 // / / /

4 // / / /  
5 // / / /

Kann das gesamte  
Shading für  
Testzwecke an- und  
ausschalten.

# CMaterial: Die Materialklasse in Vektoria

Die Glow-Modifikationsmethoden der Klasse  
CMaterial:

 1 // / / / void SetTextureGlowWhite(); // leuchtet im Schatten

 2 // / / / void SetTextureGlowBlack(); // dunkel im Schatten  
// (Default)

 3 // / / / void SetTextureGlowAsDiffuse(); // selbes Bild wie bei Diffuse

 4 // / / / void SetTextureGlowAsAmbient(); // leuchtet in der  
// Ambientfarbe

 5 // / / / Verändert das Aussehen der  
lichtabgewandten Seite.



# CMaterial: Die Materialklasse in Vektoria

Die Specular-Modifikationsmethoden der Klasse CMaterial:

1 // / / / void SetTextureSpecularWhite(); // Helles Glanzlicht (Default)

2 // / / / void SetTextureSpecularBlack(); // Kein Glanzlicht

3 // / / / void SetTextureSpecularAsDiffuse(); // Glanzlicht verwendet  
// dasselbe Bild  
// wie diffuse Textur

4 // / / / Verändern das Aussehen des  
5 // / / / Glanzlichtes (Highlight)



# CMaterial: Die Materialklasse in Vektoria

Die Transparenzmethoden der Klasse CMaterial:

`void SetTransparency(float frTransparency);`

Setzt die Durchsichtigkeit, z.B. wenn  
`frTransparency = 0.0 => Opak,`  
`frTransparency = 0.5 => Halbdurchsichtig`  
`frTransparency = 1.0 => voll durchsichtig`

`void SetTransparencyOn();`

`void SetTransparencyOff();`

Schaltet Transparenz dediziert an  
und aus (wichtig für Sortierung)



Materialien

# Übung zu Materialien



Erzeugen Sie drei Wolkenschichten mit Hilfe von  
Alpha-Texturen!



Freiwillige Zusatzaufgabe für die schnellen Nerds:

- Lassen Sie die Wolkenschichten gegeneinander verschieben!



Materialien

# Ausgabe Alphatexturierung



1 // / / /

2 // / / /

3 // / / /

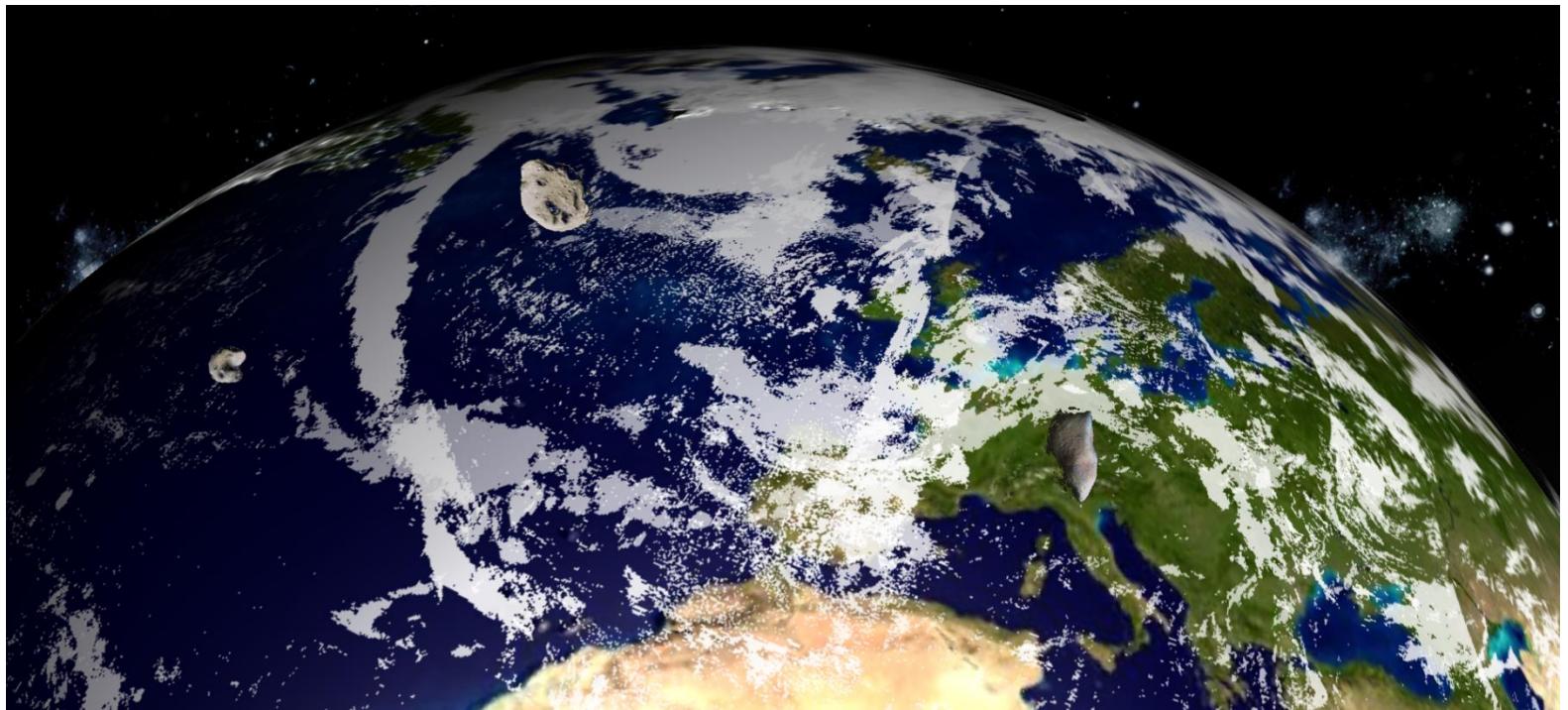
4 // / / /

5 // / / /



Knotenobjekte von Vektoria

# Ausgabe Alphatexturierung



1 // / / /

2 // / / /

3 // / / /

4 // / / /

5 // / / /



# CMaterial - Modifikationsmethoden



Weitere Modifikationsmethoden der Klasse CMaterial:

`void SetColorAmbient(CColor color);`

`void SetBumpStrength(float fBumpStrength);`

`void SetSpecularSharpness(float  
fSpecularSharpness);`

`void SetDiffuseSharpness(float fDiffuseSharpness);`



Verändern das Aussehen des  
Glanzlichtes (Highlight)



# CMaterial - Modifikationsmethoden

Weitere Modifikationsmethoden der Klasse CMaterial:

void SetColorAmbient(CColor color);



Setzt die ambiente Farbe (virtuelle Farbe des Hintergrunds)

void SetBumpStrength(float fBumpStrength);



Setzt die Stärke der Bump Map-Textur (Default = 1.0). Auch negative Werte sind möglich, Berge werden dann zu Tälern und umgekehrt.



# CMaterial - Modifikationsmethoden



Weitere Modifikationsmethoden der Klasse CMaterial:

1 // / / /  
void SetSpecularSharpness(float  
fSpecularSharpness);

2 // / / /  
void SetDiffuseSharpness(float fDiffuseSharpness);

3 // / / /  
4 // / / /  
5 // / / /  
  
Setzt die Schärfe (Fokussierung und „Kleinheit“)  
des Glanzlichtes und der diffusen Schattierung.





# CMaterial – Animierte Materialien

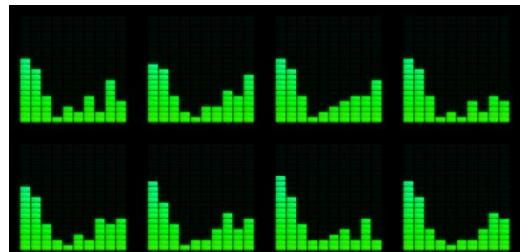
`SetAni(ixPics, iyPics, fFps);`

Erzeugt animierte Materialien,  
dabei bedeuten:

- `ixPics`: Anzahl der Unterbilderspalten
- `iyPics`: Anzahl der Unterbilderzeilen
- `fFps`: Bildwiederholrate

`ixPics = 4`

`iyPics = 2`



1 // / /

2 // / /

3 // / /

4 // / /

5 // / /





# CMaterial – Bot-Materialien

SetBot(ixPics, iyPics);  
SetPic(ixPic, iyPic);

1 // / / / **2** // / / / **3** // / / / **4** // / / / **5** // / / /

**SetBot** erzeugt ein steuerbares Material (= Bot-Material), dabei bedeuten:

- ixPics: Anzahl der Unterbilderspalten
- iyPics: Anzahl der Unterbilderzeilen

Mit **SetPic** wird dann das Unterbild eines Bot-Materials ausgewählt werden, dabei bedeuten:

- ixPic: Spaltennummer
- iyPic: Zeilennummer

Achtung, die Spalten- und Zeilennummern beginnen mit Null!



# CMaterial – Bot-Materialien (Beispiel)

- 1 // / / /
- 2 // / / /
- 3 // / / /
- 4 // / / /
- 5 // / / /



```
m_zm.MakeTextureDiffuse  
("textures\\FrequencyMeter.jpg");  
...  
m_zm.SetBot(4, 2);  
m_zm.SetPic(1, 1);
```

ixPics = 4

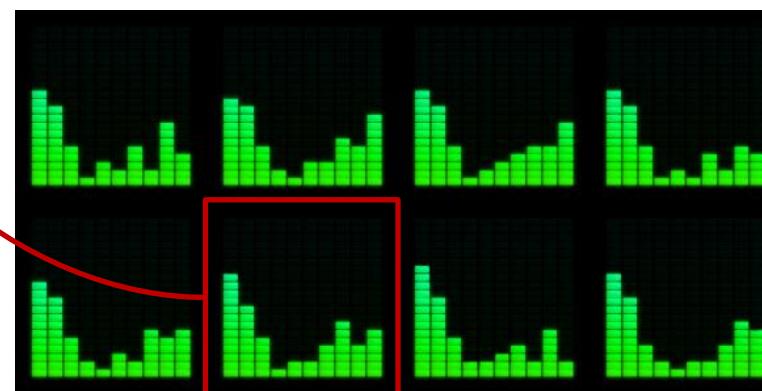
0

1

2

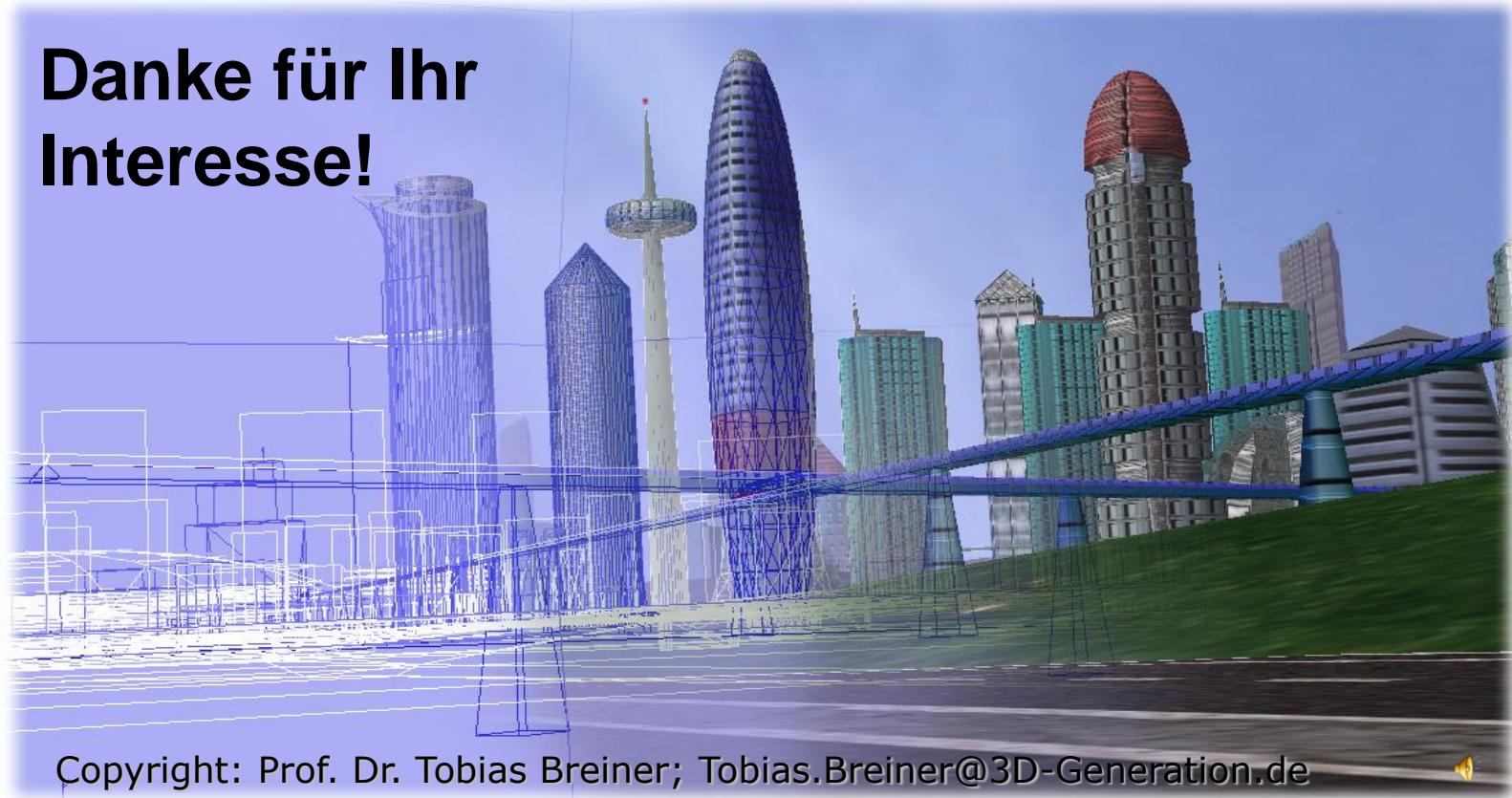
3

ixPic



# |||||GAME ||||OVER

Danke für Ihr  
Interesse!



Copyright: Prof. Dr. Tobias Breiner; Tobias.Breiner@3D-Generation.de

