Shader (HLSL)

Universität Freiburg -
Softwarepraktikum SS08

Struktur und Inhalt

- **Einleitung**
- Shader
- Beispiele
- Shader in XNA
- NVIDIA FX Composer

Einleitung - Rendering in XNA

- Shader basiertes Rendering
  - vertex daten → pixel output
  - Grafikverarbeitung auf der Grafikkarte
  - → hohe Performance
- XNA „BasicEffect“ oder eigene Shader

Einleitung - Was ist ein Shader?

- beschreibt Darstellung von Vertices und Pixeln
- Möglichkeit Vertex-/Pixeldaten zu manipulieren (Farbe, Position, Textur,…)

➔ Lighting, Shading, Transparenz, Multitexturing, Postprocessing (z.B. Motion Blur), …

- für fortgeschrittene Grafikelemente ist der „BasicEffect“ von XNA nicht ausreichend

Einleitung - FFP

- vor 2001: Kommunikation mit Grafikhardware über Fixed Function Pipeline (FFP)
- zunehmende Komplexität von Hardware, DirectX API, FFP

➔ neuer Ansatz für individuellere Grafikgestaltung: direkte Ausführung von Assembler Code

➔ „Shader“

Einleitung - HLSL

- mit Assembler direktes Arbeiten auf der Hardware, aber komplex und aufwendig
- ➔ Entwicklung von Hochsprachen für die Shader Programmierung
- Microsoft: High Level Shader Language (HLSL)
- HLSL .fx Dateien direkt von XNA unterstützt
- HLSL ähnelt stark C#-Syntax (z.B. Kontrollfluss for-/while-/do-Schleifen, if-else etc.)

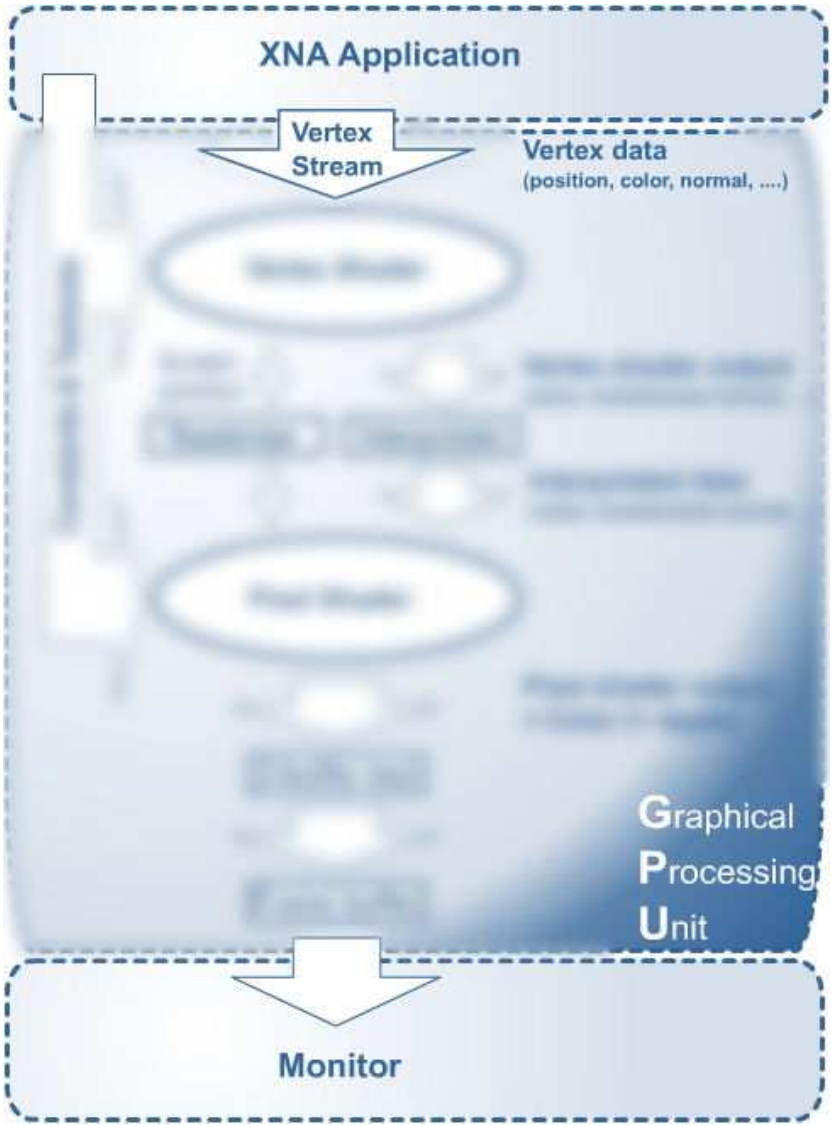
Struktur und Inhalt

- Einleitung
- **Shader**
- Beispiele
- Shader in XNA
- NVIDIA FX Composer

Shader

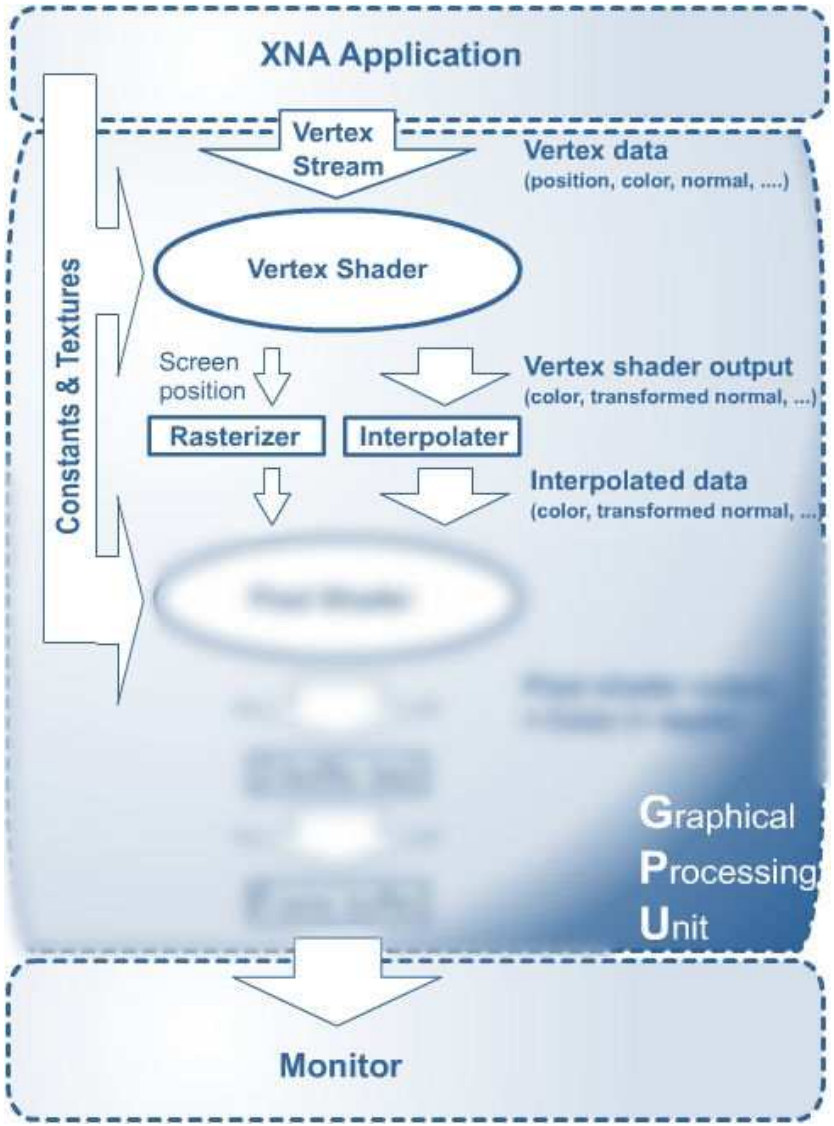
Ziel: Anzeige verschieden-farbiger Pixel die eine Spielszene formen

- Bereitstellen von Vertex Daten (Manuell / Models / Sprites)
- Übergabe der Vertex Daten an die GPU



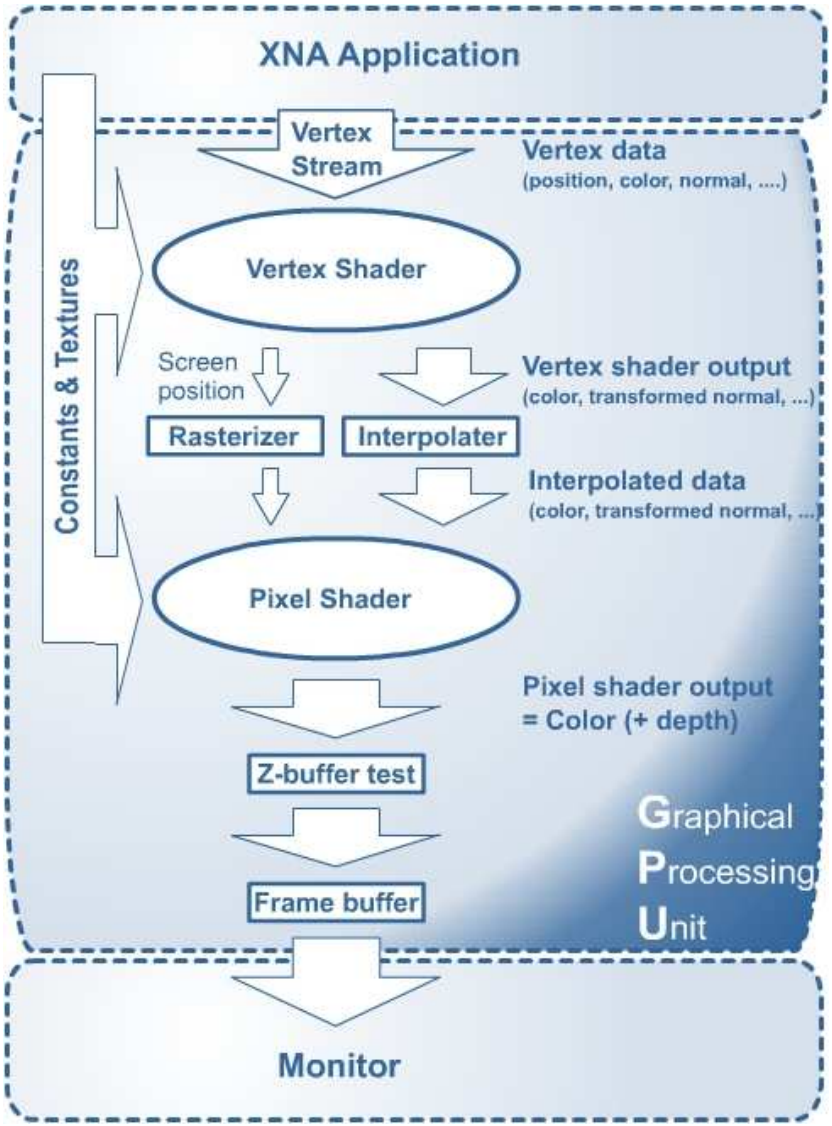
Shader

- Übergabe von Parametern aus XNA
- Vertex Processing:
  - Transformation des Vertex Input
- Rasterization & Interpolation



Shader

- Pixel Processing:
  - Verarbeitung jedes interpolierten Pixels
- Z-Buffer:
  - Auflösen von Verdeckungen
- Frame Buffer:
  - Finales Bild bestehend aus Pixeln



Vertex Shader (VS)

- Berechnungen für jeden Vertex der Szene
- Manipulation von Vertex Daten
- Berechnung von Daten für Pixel Shader
- z.B. per-vertex Lighting, Positionierung, Texturkoordinaten etc.
- In/Out: Position, Farbe/Texturkoordinaten, Normalen…

Pixel Shader (PS)

- Berechnungen für jeden Pixel der Szene
- z.B. per-pixel lighting, Färbung, Texture Sampling etc.
- teurer als per-vertex Berechnungen
- In: interpolierter Output vom VS
- Out: finale Farbe (des Pixels)

Techniques / Passes

- Pass: definiert ein Paar von Vertex- und Pixel Shader das im entsprechenden Durchgang genutzt wird
- Technique: beschreibt eine „Maltechnik“ mit mindestens einem Pass

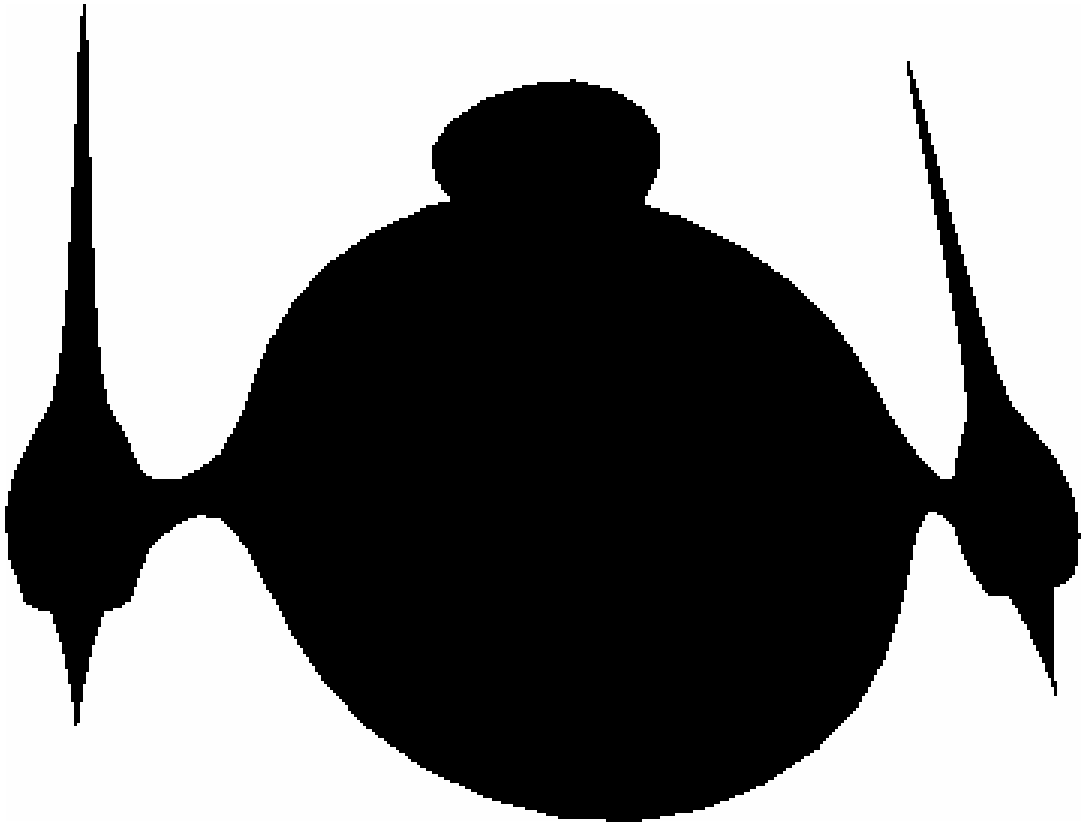
- HLSL Syntax, Typen, intrinsic Functions etc. in den Beispielen

Struktur und Inhalt

- Einleitung
- Shader
- **Beispiele**
- Shader in XNA
- NVIDIA FX Composer

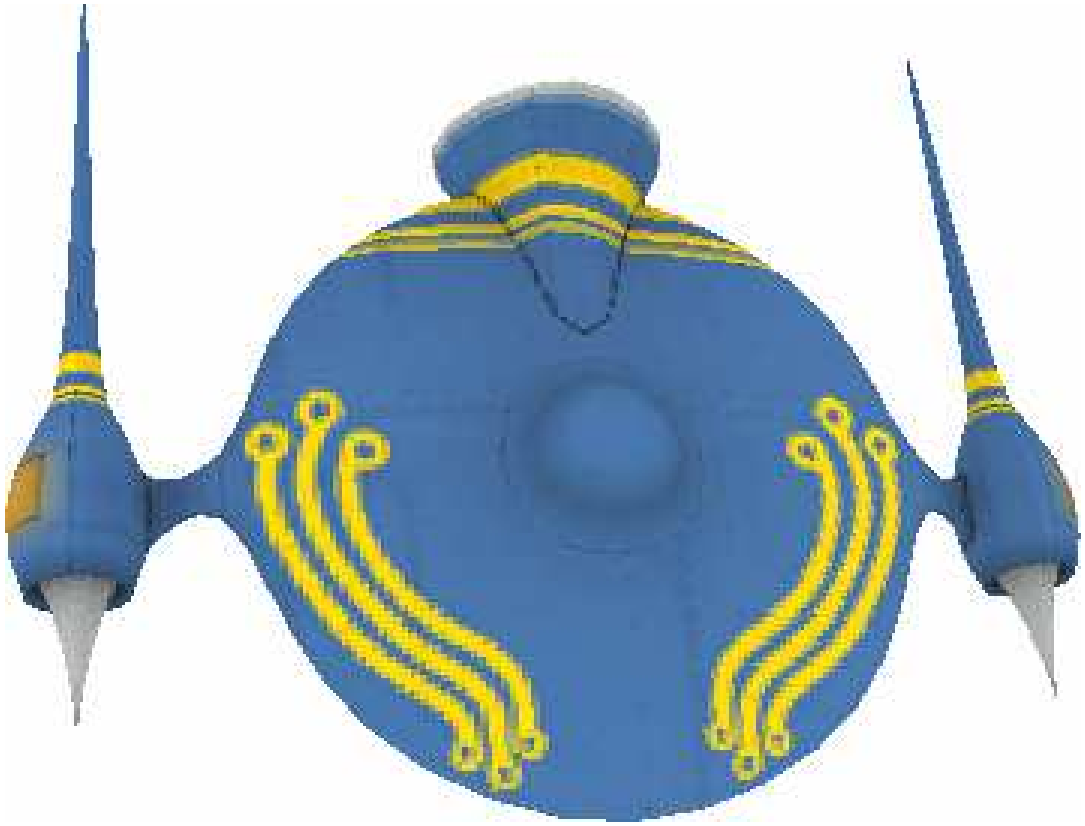
1. Shader - Färben

- Ergebnis:



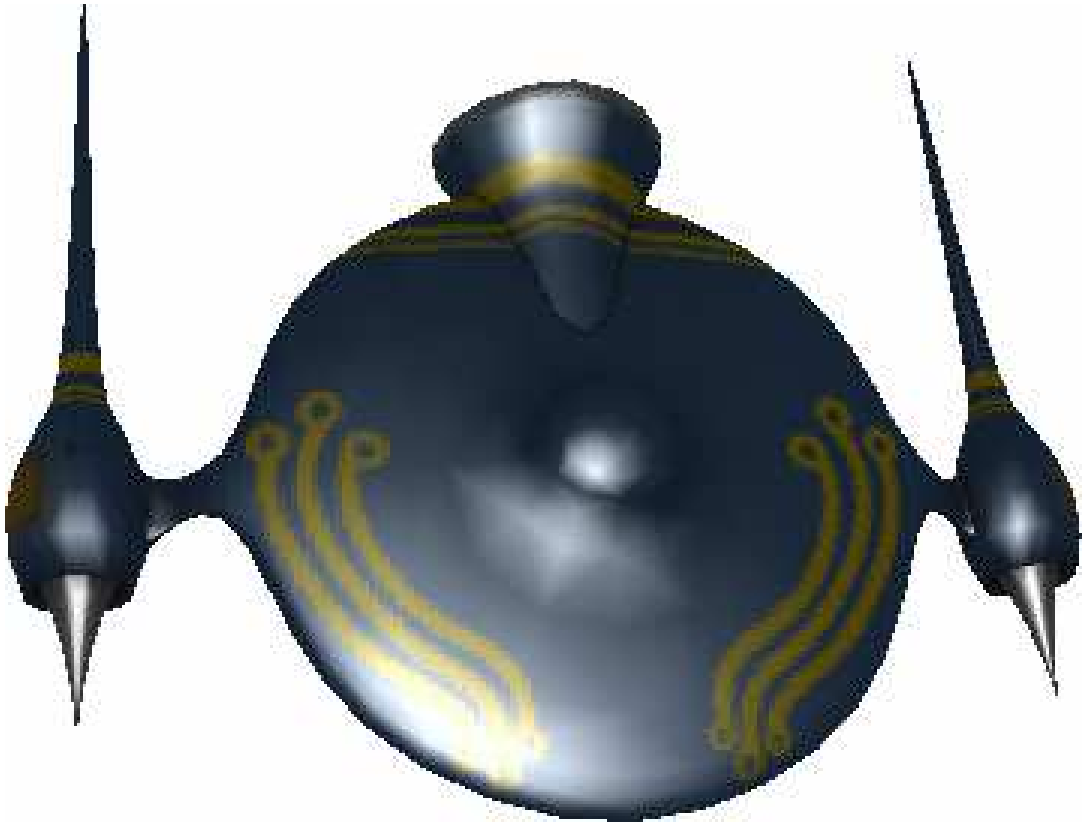
2. Shader - Textur

- Ergebnis:



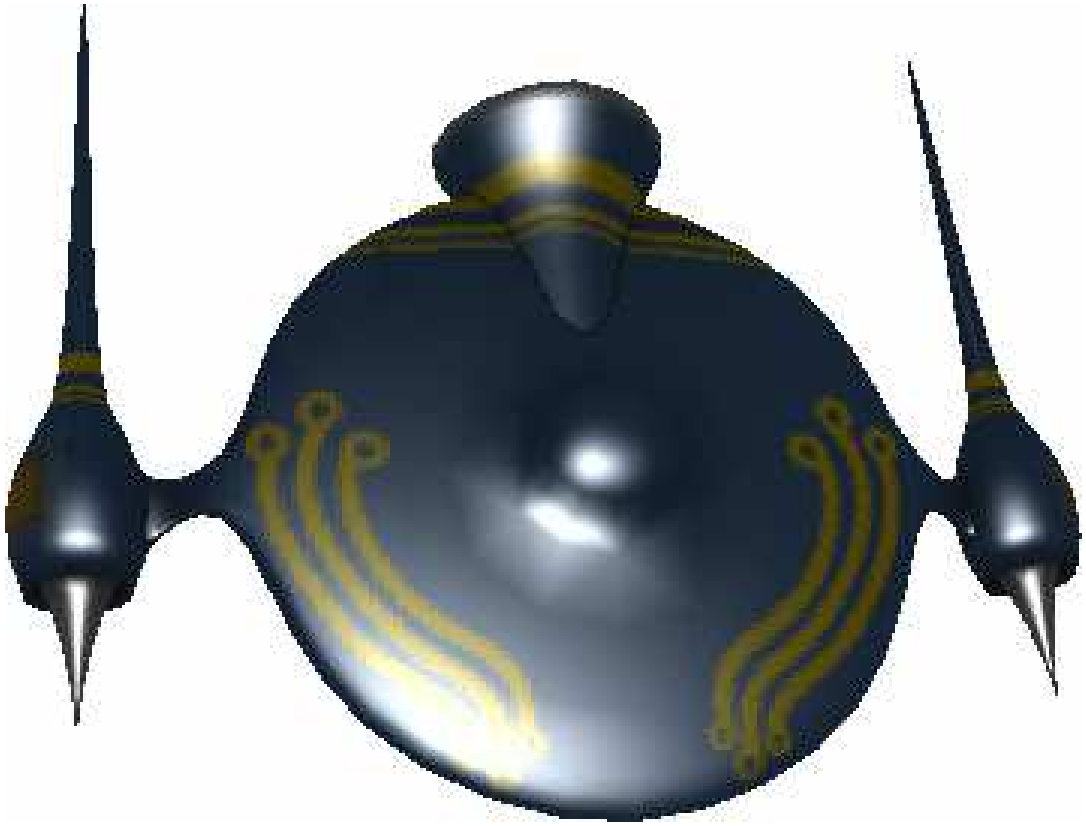
3. Shader - vertex-based Lighting

- Ergebnis:

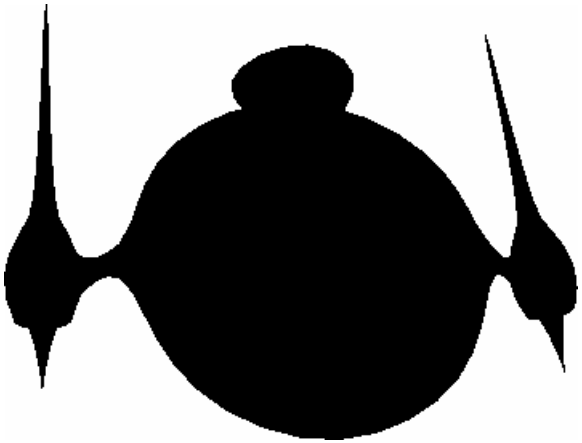


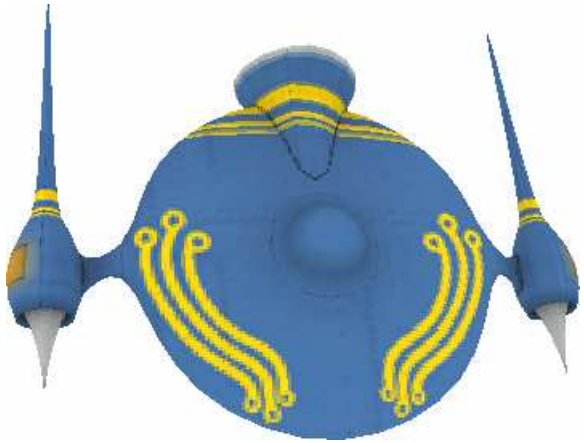
4. Shader - pixel-based Lighting

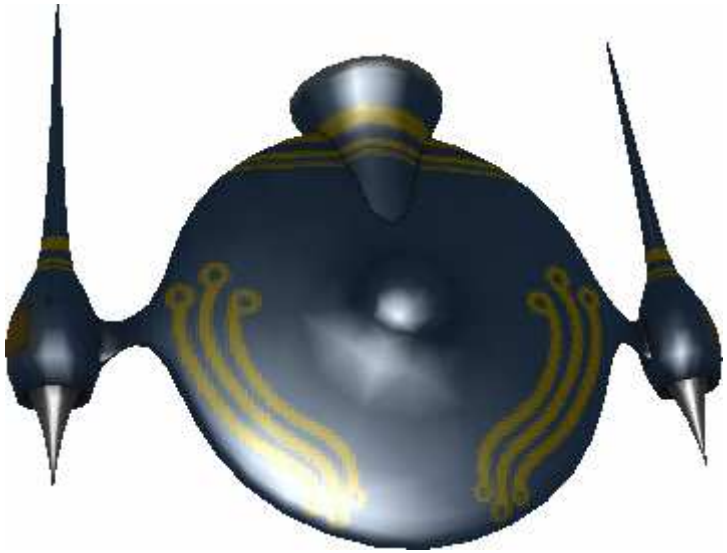
- Ergebnis:

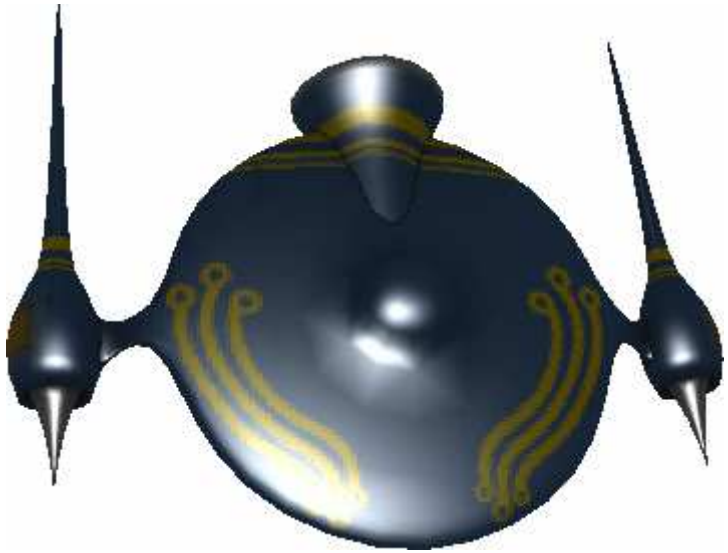


Shader Beispiele









Struktur und Inhalt

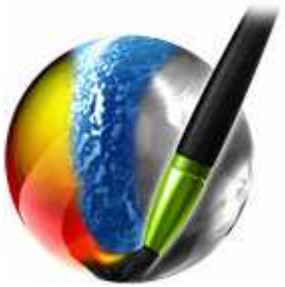
- Einleitung
- Shader
- Beispiele
- **Shader in XNA**
- NVIDIA FX Composer

Struktur und Inhalt

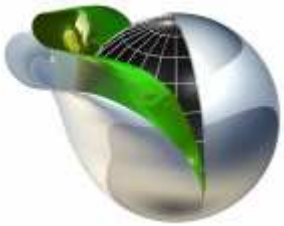
- Einleitung
- Shader
- Beispiele
- Shader in XNA
- **NVIDIA FX Composer**

NVIDIA FX Composer

- NVIDIA FX Composer 2.0 / 2.5 Beta 2
  - HLSL Syntax Highlighting
  - Shader Compiler
  - Shader Datenbank
  - uvm.



- NVIDIA Shader Debugger Beta 2



- http://developer.nvidia.com/

Quellen

- http://www.riemers.net/
- http://creators.xna.com/

- Microsoft - XNA Game Studio Creator‘s Guide - An Introduction to XNA Game Programming - Stephen Cawood, Pat McGee
- Microsoft - XNA Unleashed - Graphics and Game Programming for Xbox 360 and Windows - Chad Carter