Computergrafik
Prof. Dr. W. Kurth - Th. Mangoldt

VRML-Kurs

Teil 1

VRML:
Virtual Reality Markup Language
→ Virtual Reality Modelling Language

- Beschreibungssprache für VR-Szenen
- (kein Softwaresystem)
- Umsetzung durch *VRML-Browser*
- VRML-Szenen in ASCII-Dateien abgelegt
- Standardisierung: Dieselbe Szene durch verschiedene Browser darstellbar

VRML-Umfang:

3D-Grafik
+ Dynamik (Animation)
+ Interaktion
+ Klang
+ Internetfähigkeit (VRML-Browser als Plugin von Webbrowsern)

Versionen: VRML 1.0, 2.0, 97

Features:

Geometriebeschreibungen	Grundprimitive (Quader, Zylinder, Kegel, Kugel) polygonale Objekte elevation grids (Terrain-Gitter) extrudierte Objekte
Materialbeschreibungen	Farbe Schattierungen

Beleuchtungsbeschreibungen	
Texturbeschreibungen	
Textbeschreibungen	
Transformationen	Translationen Rotationen Skalierungen allgemeine Transformationen
Animationen	Zeit Interpolation
Hintergrund und Nebel	
Interaktionsbeschreibungen	
Multimediabeschreibungen	Video 3D-Klang
Strukturbeschreibungen	Gruppierung Prototypen
Detaillierungsbeschreibungen	LOD = level of detail
Kamerabeschreibungen	Viewpoints
Hypermediabeschreibungen	Anchors
Beschreibung von Programmierlogik	Java Javascript VRMLscript

- VRML kann mit immersiven Technologien umgesetzt werden
- Einbindung von Programmen beliebiger Komplexität
- Programme können in VRML-Szenen eingreifen (Anwendungen z.B.: Mehrbenutzersysteme, Datenbank-Anbindung, Simulationen)

Geschichte

- Erste Ideen zu virtuellen Welten im WWW von Mark Pesce und Tony Parisi, 1994
- im Rahmen der ersten internationalen Konferenz über das WWW (Mai 1994 bei CERN in Genf) wird eine Sitzung zu einer geplanten "Virtual Reality Markup Language" abgehalten (wichtige Rolle: Tim Berners-Lee, der Entwickler von HTML); das Akronym "VRML" wird geprägt

- Oktober 1994: VRML 1.0 wird von Tony Parisi und Gavin Bell präsentiert, basiert nach Entscheidung durch Internet-Abstimmung (Mailingliste) auf Open Inventor von SGI. Beginn der Unterstützung durch SGI, Netscape und Microsoft
- VRML Architecture Group (VAG; 8 technische Experten) wird im Anschluss an die SIGGRAPH'95 gegründet (August 1995)
- Anfang 1996 Aufruf zu Vorschlägen für VRML 2.0 durch die VAG
- August 1996: VRML 2.0-Spezifikation wird auf der SIGGRAPH'96 vorgestellt, nach offener Internet-Abstimmung auf Moving Worlds von SGI basierend; Gründung des VRML-Konsortiums (Vertreter von Firmen, Forschungseinrichtungen und Universitäten)
- Anfang 1997: Beginn der ISO-Standardisierung mit der Erarbeitung von VRML 97
- Ende 1997: VRML 97 wird standardisiert als ISO/IEC DIS 14772-1
- Weiterentwicklung: x3d

VRML-Browser

Cortona (Parallel Graphics), Download:
http://www.parallelgraphics.com/cortona/

BS Contact VRML (Bitmanagement Software)
http://www.bitmanagement.de

Cosmo Player (wird nicht mehr weitergepflegt)
World View
....
Seite mit Links zu verschiedenen Download-Sites:
http://www.web3d.org/vrml/vrml.htm

VRML-Browser als Plugin zu MS Internet Explorer bzw. zu Netscape.
Jeder Browser stellt Navigationshilfen für den 3D-Raum zur Verfügung
Anordnung und Bezeichnungsweisen browserspezifisch, aber ähnliche Grundfunktionen.

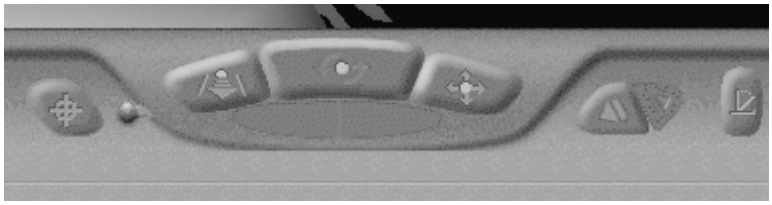
z.B. für Cosmo Player:

6 Haupt-Navigationsmodi

Gehen Neigen Gleiten	Go Tilt Slide	Bewegungen in der horizontalen Ebene Änderung des Blickwinkels Bewegungen in der vertikalen Ebene
Rotieren Schwenken Zoomen	Rotate Pan Zoom	lässt die Szene rotieren Eigenbewegung um eine Szene, die im Blickfeld bleibt Änderung des Abstands von der Szene

Die eigentliche Navigation erfolgt mit der Maus. Die Form des Mauscursors zeigt an, welcher Navigationsmodus aktiv ist. Zwischen der oberen und der unteren Dreiergruppe wird mit einem virtuellen Hebel umgeschaltet ("Steuerelemente ändern"; auch durch Hotkeys (Tastatur) möglich).

Schaltfläche mit Funktionen der ersten Gruppe

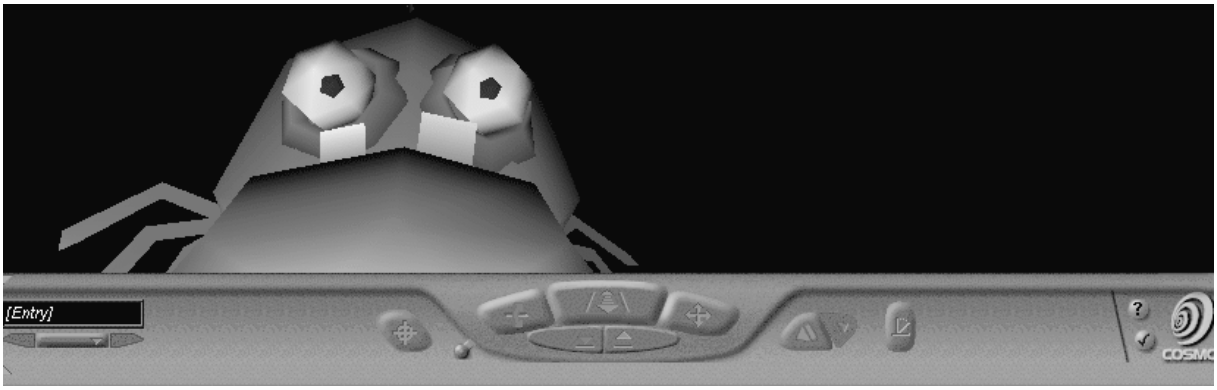


Schaltfläche mit Funktionen der zweiten Gruppe

Zusätzliche Sonderfunktionen:

Gravitation	Gravity	in hügeliger Landschaft am Boden bleiben (man folgt den Höhen und Tiefen der Oberfläche)
Treiben	Float	erlaubt, zu fliegen
Suchen	Seek	durch Anklicken eines Objekts bewegt man sich an dieses heran

Bewegung rückgängig machen	Undo move	
Bewegung wiederherstellen	Redo move	
Richten	Straighten	man wird in eine Orientierung senkrecht zur Horizontalebene gebracht
Viewpoint		man kann Blickpunkte speichern und mit Namen versehen
Preferences		Aktivierung des Voreinstellungs-Menüs
COSMO		Hyperlink zur Webseite von Cosmo
Help		Online-Hilfe (HTML)
Warnleuchten		werden aktiv bei Fehlern, *bei Anklicken erscheint die Fehlermeldung*



beim Cortona Viewer:
z.T. andere Namen für dieselben oder ähnliche Funktionen

walk, fly, study
plan, pan, turn, roll
goto, align, restore, fit

(interaktiv testen!)

Cortona Schaltleisten:

VRML-Dateien

Endung .wrl ("world")
ASCII-Datei (genauer: ab V. 2.0 UTF-8 Zeichensatz gem. ISO 10646-1:1993)

Sprachspezifikation:
http://www.vrml.org/technicalinfo/specifications/vrml97/index.htm

Aufbau einer VRML-Datei:

- Header (Version und Zeichensatzangabe, obligatorisch)
- Zeilenkommentare
- VRML-*Knoten*
- innerhalb der Knoten-Spezifikationen: *Felder* (= festgelegte Attributierungen von Knoten, denen Werte zugewiesen werden)
- PROTO-Statements
- ROUTE-Statements

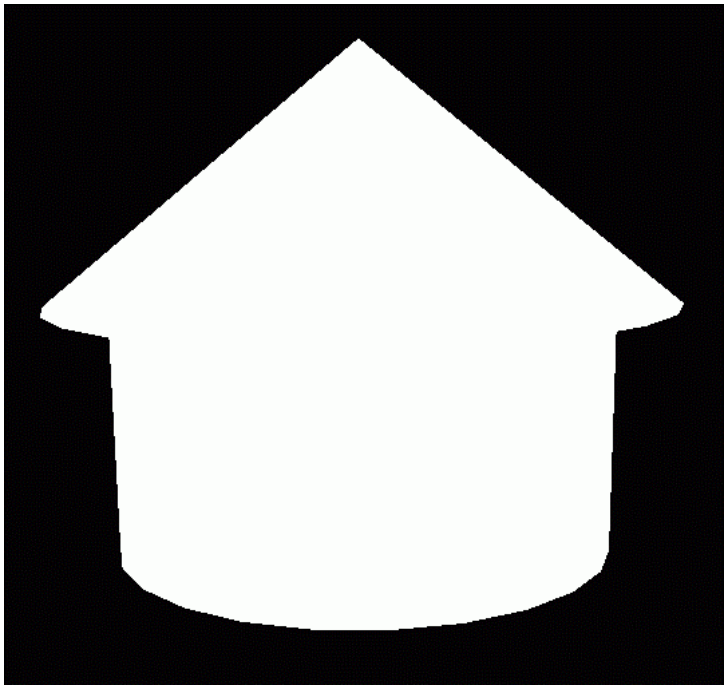
Zuordnung von Feldangaben zu Knoten mit geschweiften Klammern { ... }

Gruppierung von Knoten durch Gruppen-Knoten, "Kinder" in eckigen Klammern [...]

Beispiel einer VRML2.0 - Datei:

```
#VRML V2.0 utf8
Cylinder
    {
    height 2.0
    radius 2.0
    }
Transform
    {
    translation 0.0 2.0 0.0
    children
        [
        Cone
            {
            bottomRadius 2.5
            }
        ]
    }
```

Ergebnis:



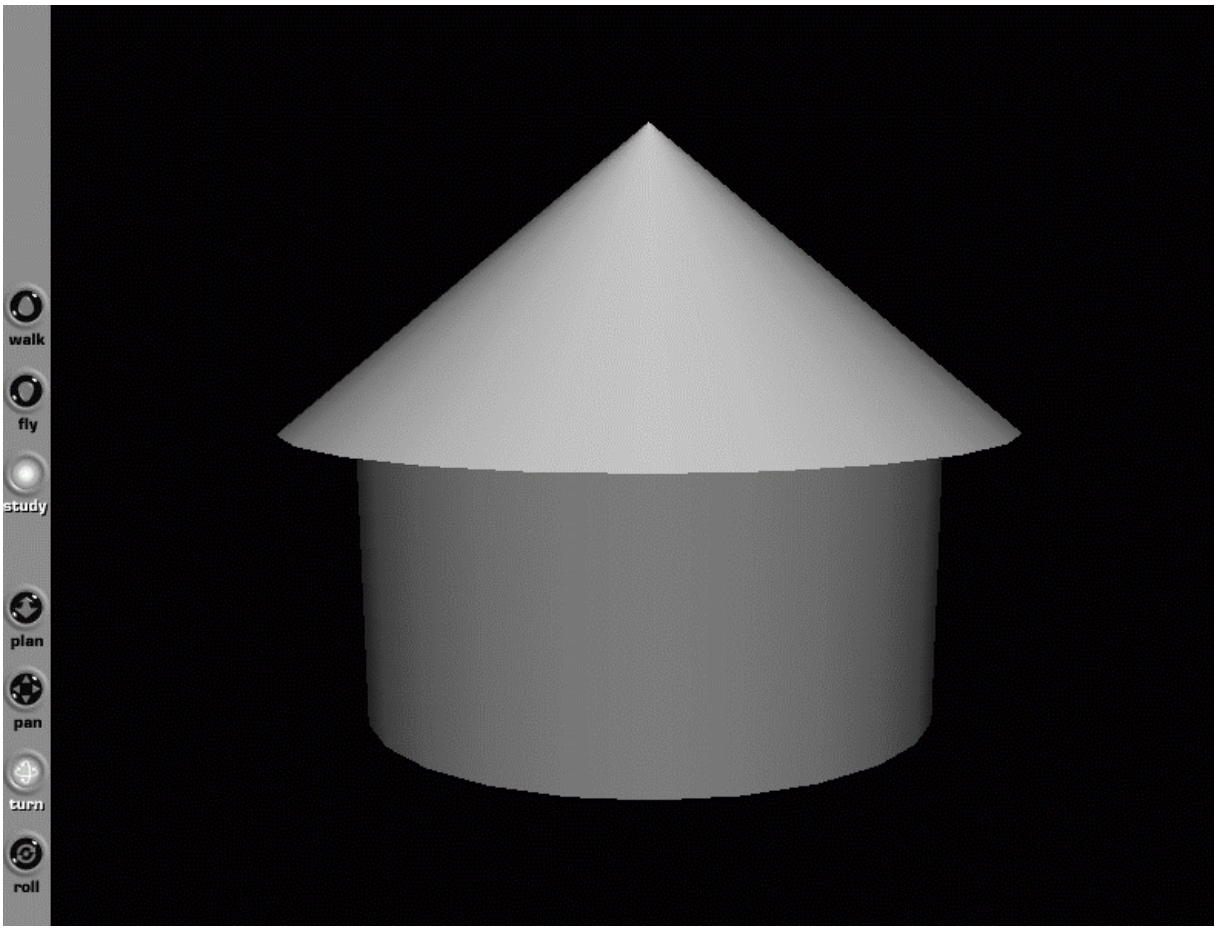
default für Material: weiß, "selbstleuchtend" (nicht reflektierend)

Mit selbst gesetzten Materialeigenschaften (hier: diffus reflektierende Oberfläche, grau bzw. weiß):

```
#VRML V2.0 utf8
Shape
   {
   geometry Cylinder
      {
      height 2.0
      radius 2.0
      }
   appearance Appearance
      {
      material Material
         { diffuseColor 0.5 0.5 0.5 }
      }
   }
Transform
   {
   translation 0.0 2.0 0.0
   children
      [
      Shape
         {
         geometry Cone
            {
            bottomRadius 2.5
            }
         appearance Appearance
             {
            material Material
                { diffuseColor 1.0 1.0 1.0 }
            }
         }
      ]
   }
```

Der Shape-Knoten gruppiert Geometrie und Materialeigenschaften!

Ergebnis:



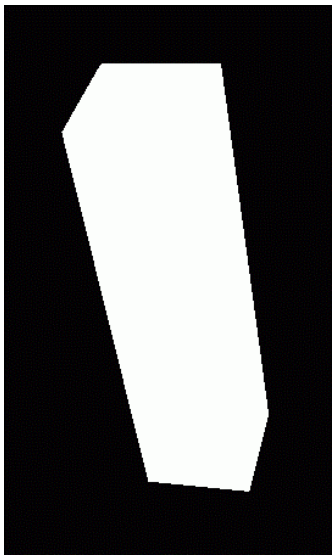
Einfachstes Objekt:

- Box (Quader)

nur die 3 Abmessungen (Länge, Höhe, Tiefe) werden hier angegeben:

```
#VRML V2.0 utf8
Box
    {
    size 1 4 1
    }
```

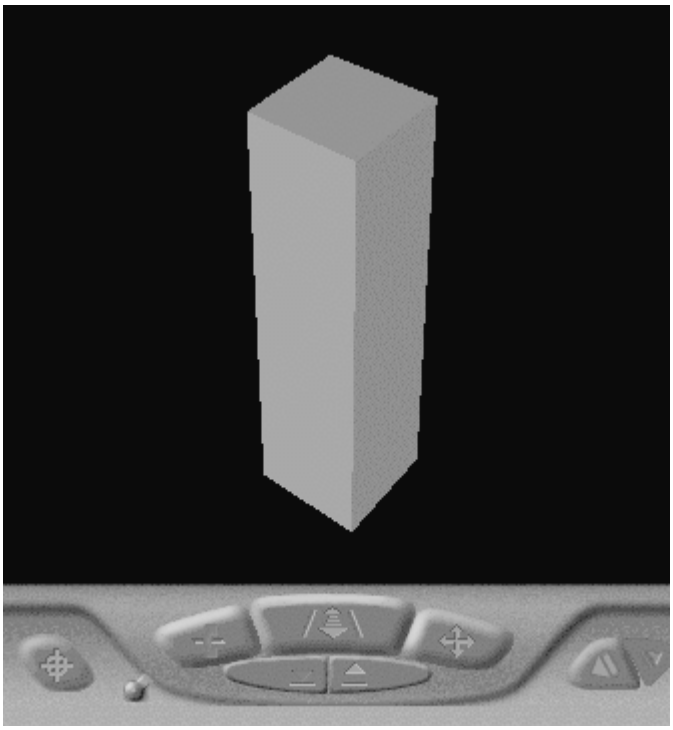
Ergebnis:



dasselbe mit Materialeigenschaften:

```
#VRML V2.0 utf8
Shape
   {
   geometry Box
      {
      size 1 4 1
      }
   appearance Appearance
      {
      material Material
         { diffuseColor 0.5 0.5 0.5 }
      }
   }
```

Ergebnis (hier im CosmoPlayer):



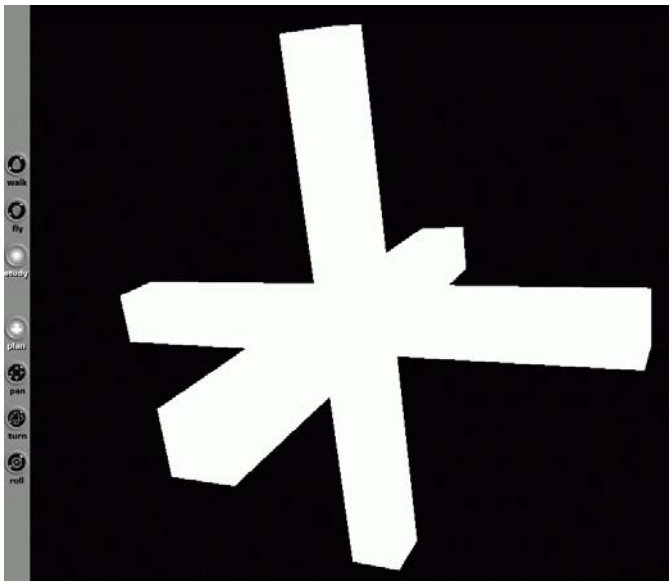
Felder, für die keine Spezifikation angegeben wird, werden automatisch mit Default-Werten besetzt.

Die Primitiv-Objekte (Box, Cone, Cylinder, Sphere) werden standardmäßig um den Nullpunkt zentriert.

Überlagerung von 3 verschieden dimensionierten Quadern zu einem "Koordinatenkreuz":

```
#VRML V2.0 utf8
Box
   { size 10 1 1 }
Box
   { size 1 10 1 }
Box
   { size 1 1 10 }
```

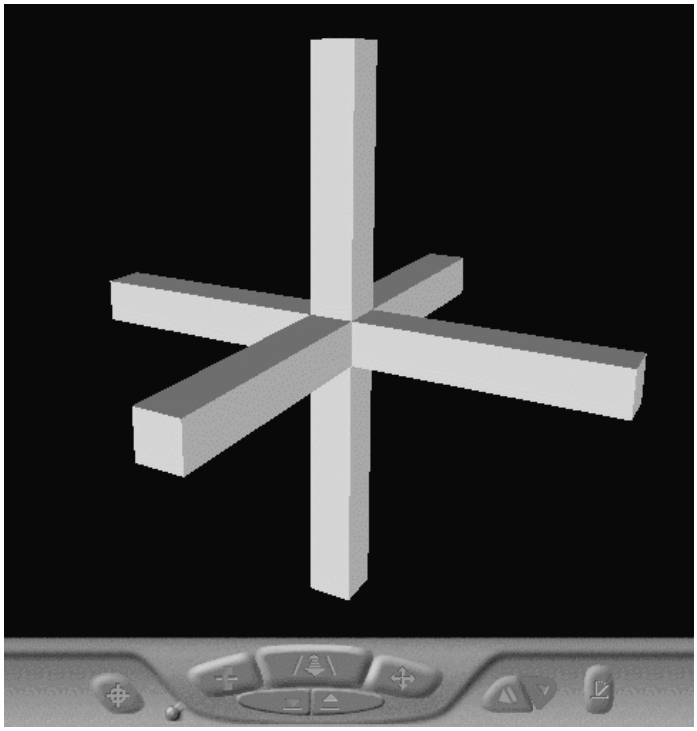
Ergebnis:



mit Materialfarbe:
es muss wieder mit **Shape**-Knoten gruppiert werden

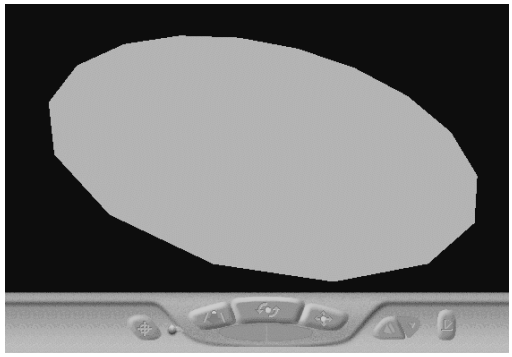
```
#VRML V2.0 utf8
Shape
   {
   geometry Box
      { size 12 1 1 }
   appearance Appearance
      {
      material Material { diffuseColor 1.0 1.0 1.0 }
      }
   }
Shape
   {
   geometry Box
      { size 1 12 1 }
   appearance Appearance
      {
      material Material { diffuseColor 1.0 1.0 1.0 }
      }
   }
Shape
   {
   geometry Box
      { size 1 1 12 }
   appearance Appearance
      {
      material Material { diffuseColor 1.0 1.0 1.0 }
      }
   }
```

Ergebnis:



Ansprechen von Teilen eines Primitivobjekts über spezielle Felder, z.B. beim Zylinder: **`side, top, bottom`**

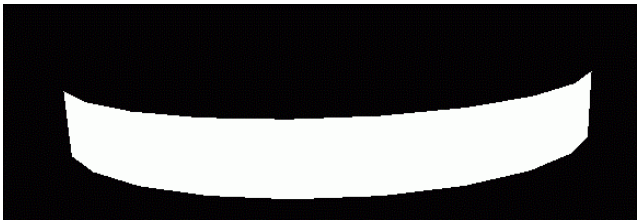
```
#VRML V2.0 utf8
Cylinder
    {
    bottomRadius 3
    height 0.7
    side FALSE
    }
```



Beachte: es ist jeweils nur die Außenseite der Polygone sichtbar! (Hier also je nach Blickwinkel "top" *oder* "bottom", obwohl beide noch "vorhanden" sind)

```
#VRML V2.0 utf8
Cylinder
   {
   radius 3
   height 0.7
   top FALSE
   bottom FALSE
   }
```

Ergebnis:

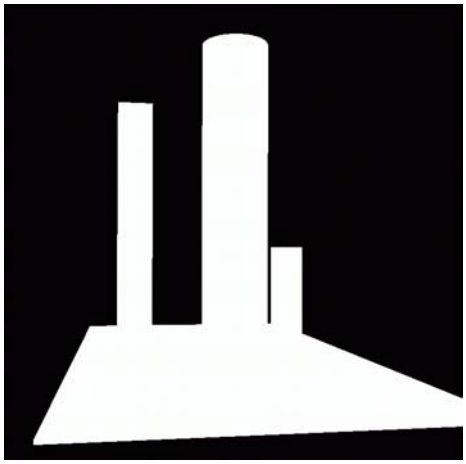


Verwendung von Transformationsknoten zur Positionierung von Objekten

Beispiel:

```
#VRML V2.0 utf8
Box { size 10 0.1 20 }    # Boden
Transform
   {
   translation -3 4 4
   children
      [
      Box { size 1 8 1 }
      ]
   }
Transform
   {
   translation 3 2 -1
   children
      [
      Box { size 1 4 1 }
      ]
   }
Transform
   {
   translation 0 5 4
   children
      [
      Cylinder { height 10 }
      ]
   }
```

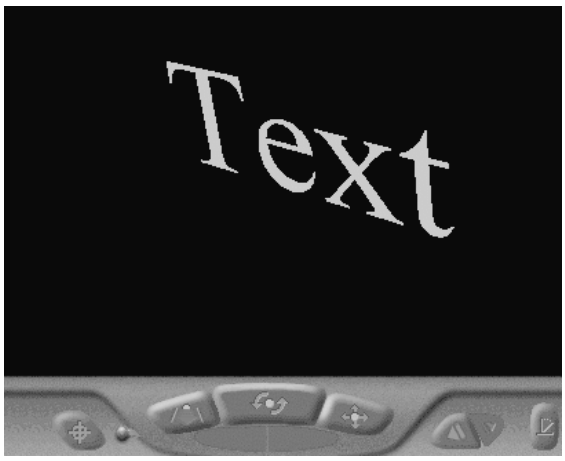
Ergebnis:



Textausgabe in VRML

```
#VRML V2.0 utf8
Text
    {
    string "Text"
    }
```

Ergebnis:



Präzisere Spezifikationen (Knotentypen, zulässige Felder für die Knoten...) im nächsten Teil!