Jihočeská univerzita v Českých Budějovicích
Pedagogická fakulta
Katedra informatiky

Jazyk VRML 2.0

Bakalářská práce

VYPRACOVALA : Marie Serbusová
VEDOUCÍ PRÁCE: PaedDr. Petr PEXA
OBOR : Výpočetní technika a informatika

Anotace:

Cílem diplomové práce je zpracovat uživatelskou příručku jazyka VRML 2.0 jako prostředku pro tvorbu virtuálního modelování v prostředí WWW a provést porovnání se starší verzí VRML 1.0. Součástí diplomové práce bude také konkrétní prezentace virtuální reality, vytvořená v jazyce VRML 2.0. Práce by měla být v ČR unikátní publikací zabývající se touto problematikou.

Prohlašuji, že jsem bakalářskou práci na téma Jazyk VRML 2.0 vypracovala samostatně a použila jsem jen pramenů, které cituji a uvádím v seznamu použité literatury.

V Českých Budějovicích dne 9. března 2005

……..

Marie Serbusová

Je mou milou povinností poděkovat na tomto místě PaedDr. Petru Pexovi za spolupráci, cenné rady, podnětné připomínky, které mi pomohli při vypracování této práce. Dále bych chtěl poděkovat své rodině a přátelům, za odborné rady a trpělivost při psaní této práce.

Obsah

1 Úvod

VRML je zkratka pro Virtual Reality Modeling Language. Dle názvu se jedná o jazyk pro popis světů virtuální reality. A to je přesně to, čím VRML chce být, k čemu směřuje. VRML je od svých počátků spojeno s Internetem.

K prohlížení VRML dokumentů potřebujete speciální prohlížeč (browser), i když novější verze programu jako je Netscape Navigator mají již tuto schopnost integrovanou.

2 Historie VRML

Počátky jazyka VRML můžeme najít v místech, kde se rodily systémy moderní prostorové grafiky, totiž ve firmě Silicon Graphics, Inc. Její programátoři navrhli koncem 80. let knihovnu pro práci s prostorovými objekty, nazvanou *Inventor*. Byla vybudována jako nadstavba nad známou a úspěšnou základní grafickou knihovnou GL. začátkem devadesátých let došlo k inovaci – vznikla nová základní grafická knihovna s názvem *OpenGL* a k ní nová aplikační knihovna *OpenInventor*.

Na vytvoření VRML se dohodla skupinka zájemců na první výroční konferenci WWW v Ženevě na jaře 1994. Během podzimu téhož roku byla představena první pracovní verze. Konečně 26. května 1995 byla dokončena oficiální specifikace VRML 1.0. Současně se vznikem VRML 1.0 je založena nezávislá skupina programátorů a návrhářů nazvaná VAG (Vrml Architecture Group). Tato skupina oslovuje tvůrce systémů pro virtuální realitu, internetovou veřejnost a odborníky na grafiku s výzvou k vytvoření specifikace budoucího jazyka VRML 2.0.

V dubnu roku 1996 je z celkem osmi návrhů vybrán společný návrh firem Sony a Silicon Graphics s pracovním názvem Moving Worlds, který se stává základem pro vznik VRML 2.0. Je podroben více než ročnímu doplňování, úpravám a upřesňování ve veřejné elektronické diskusi. Z neformální skupiny VAG se stává VRML Consorcium, Inc., které zahájí okamžitě spolupráci s mezinárodní standardizační organizací ISO na vznik VRML ve formě normy. Toto úsilí je korunováno úspěchem a to v roce 1997, kdy norma dostává název VRML 97 (č. n. ISO/EC 14772-1:1997). V roce 1999 se VRML Consorcium přejmenovalo na Web3D Consorcium.

Pro úplnost je třeba dodat že označení VRML 2.0 je pouze programátorské označení jazyka VMRL 97.

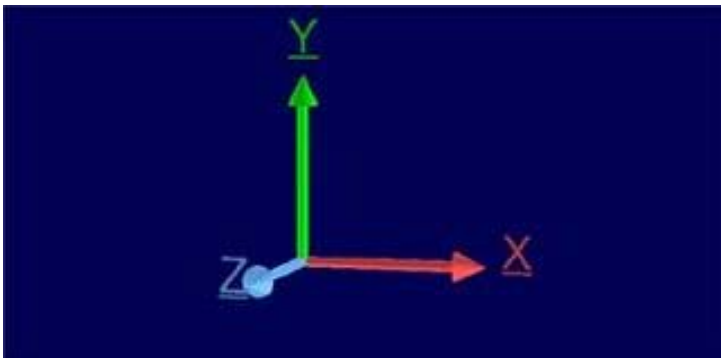
2.1 Standarty se vztahem k VRML 2.0

Aplikace VRML 2.0 by si měla rozumět s řadou standartů:

- JPEG – ISO/IEC IS 10918-1
- MIDI – specifikaci udržuje International MIDI Association
- MPEG – ISO/IEC IS 11172-1
- PNG – specifikace viz W3C
- RURL – IETF RFC 1808
- URL – IETF RFC 1738
- UTF8 – ISO/IEC 10646-1
- WAV – specifikaci vydal IBM a Microsoft

3 Souřadnicový systém

Pro prostorovou orientaci je bezpochyby důležitá orientace systému ve VRML. Osy jsou orientovány jak je vidět na obrázku (Obrázek 1) tak, že kladný směr osy Y směřuje vzhůru, kladný směr osy X jde směrem doprava a kladný směr osy Z směrem k uživateli.



Obrázek 1: Souřadnicový model

4 Struktura souboru VRML

Soubor s popisem v jazyce VRML 2.0 se skládá z uzlů a tvoří hierarchickou (stromovou) strukturu. V roli parametrů uzlů se mohou objevovat jiné uzly. Každý z těchto uzlů má následující vlastnosti:

- Může být pojmenován.
- Obsahuje parametry; parametrem může být i uzel, potom vzniká tzv. vztah rodič-potomek, který je základem pro tvorbu virtuální scény.
- Do parametru lze vložit odkaz na již existující uzel.
- Uzly si mohou předávat data.
- Všechny identifikátory se rozlišují podle malých a velkých písmen.

Soubor s popisem virtuálního světa můžeme rozdělit do několika základních částí. `Na` začátku je vždy hlavička. Její tvar je neměnný a informuje prohlížeč, o jaký typ souboru se jedná. Tento řádek je povinný. Za hlavičkou většinou následují úvodní informace o virtuálním světě. Jsou to informace o souboru a jeho tvůrci (`WorldInfo`), seznam zajímavých míst uvnitř virtuálního světa (`Viewpoing`) a způsob procházení světem (`NavigationInfo`). Třetí, nejrozsáhlejší část tvoří vlastní popis virtuálního světa. Pořadí těchto částí není závazné, ale je dobrým zvykem ho kvůli čitelnosti dodržovat. Doporučená konstrukce je ukázána v následujíc tabulce.

#VRML V2.0 utf8	Hlavička souboru VRML
WorldInfo {…} Viewpoint {…} NavigationInfo {…}	Úvodní všeobecné informace o virtuálním světě
Transform {…} Group PositionInterpolator {…}	Popis těles, jejich vlastností, definice prvků potřebných pro animace a interakci s uživatelem.
ROUTE … TO …	Propojení dynamických a statických prvků z předchozí části.

Tabulka 1: Struktura VRML souboru

Soubory, které popisují VRML-světy, mají příponu wrl a lze je pro úsporu místa zkomprimovat programem gzip, přičemž si zachovají příponu wrl. Prohlížeč sám rozpozná takový soubor a automaticky ho dekomprimuje.

Důležité je si uvědomit že ve VRML se zadávají vzdálenosti v metrech, čas v sekundách a úhly v radiánech. Barvy se popisují výhradně pomocí RGB v rozsahu 0 - 1.

4.1 WorldInfo (Globální informace)

`WorldInfo` je nepříliš Významný informační uzel. Umisťuje se do souboru vždy na začátek a slouží pro dokumentační účely. Dále bych chtěla podotknout, že všechny parametry, které jsou uvedeny v následujících tabulkách, jsou inicializační hodnoty, tzn. pokud nenastavíme jinak, jsou nastaveny tyto parametry.

Parametr	**Iniciální hodnota**	**Význam**
title	„“	Jméno virtuálního světa
info	[]	Posloupnost řetězců s libovolným významem

Tabulka 2: Parametry uzlu WorldInfo

```
WorldInfo{
     title „Virtuální svět“
     info [ “Autor: Marie Serbusová”,
            “Datum: 26. května 2004”
]
}
```

4.2 Viewpoint (Stanoviště)

K určování toho, co uživatel uvidí a pod jakým úhlem, slouží uzel `Viewpoint`. Pohledů lze definovat dle potřeby libovolné množství s tím, že první je považován za inicializační a je použit vždy při načtení scény. Mezi jednotlivými pohledy se přechází většinou plynule letem s postupným natočením do nového směru pohledu. Parametry zapsané kurzívou jsou parametry takové, které slouží k dynamičnosti VRML světa a budu se jimi zabývat ke konci publikace. Zde jsou uvedeny jenom pro kompletnost.

Parametr	**Iniciální hodnota**	**Význam**
position	0 0 10	Poloha avatara
orientation	0 0 1 0	Natočení avatara podle osy a úhlu; osa se zadává vektorem s hodnotami v rozsahu [-1 1], úhel je libovolný
fieldOfView	0.785398	Zorný úhel v rozsahu (0, π)
jump	TRUE	Povolení plynulého přechodu na stanoviště
description	„“	Jméno stanoviště uváděné prohlížečem

Parametr	Iniciální hodnota	Význam
set_bind		*Aktivování stanoviště, tj. umístění avatara do dané polohy a orientace*
bindTime		*Čas aktivace stanoviště*
isBound		*Stanoviště se stalo aktivním nebo naopak bylo nahrazeno jiným stanovištěm*

Tabulka 3: Parametry uzlu ViewPoint

```
Viewpoint {
  position              0 0 10
  orientation    0 0 1 0
  description    „Zepredu“
  fieldOfView    0.785398
}
```

5 Avatar (Návštěvník)

Je to pojmenování virtuálního dvojníka, který prochází virtuálním světem. To co avatar vidí, je zobrazeno v okně prohlížeče. Avatar má své rozměry, které mu například brání procházet malými průchody.

Velitelské stanoviště není nic jiného, než systém pro ovládání avatara. V prohlížeči je tento systém reprezentován prvky na ovládacím panelu aplikace.

5.1 NavigationInfo (Ovládání avatara)

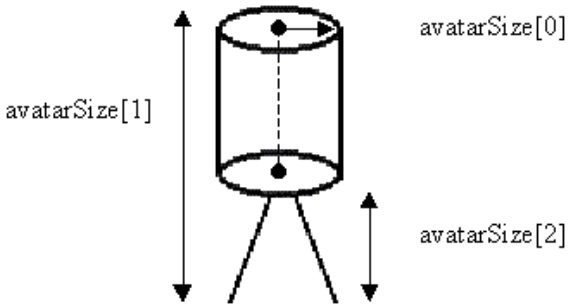
Vlastnosti avatara a některé prvky velitelského stanoviště lze zadat pomocí uzlu nazývaného `NavigationInfo`. Předdefinované hodnoty jeho parametrů definují avatara, jehož vlastnosti jsou podobné člověku (jak je vidět v Tabulce 4). Proto bývá přímé použití tohoto uzlu s jinými parametry vzácné.

Je-li v souboru více navigačních uzlů, použije se právě první z nich. Pouze jeden takový uzle může být aktivní.

Parametr	Iniciální hodnota	Význam
avatarSize	[0.25 1.6 0.75]	Rozměrové charakteristiky avatara viz Obrázek 2
headlight	TRUE	Zapnuté čelní světlo (baterka)
visibilityLimit	0.0	Dohled v metrech, nulová hodnota značí nekonečno

Parametr	Iniciální hodnota	Význam
speed	1.0	Rychlost pohybu v m/s
type	[„WALK“, „ANY“]	Velitelské stanoviště
set_bind		*Aktivování vlastností avatara a způsobu jeho ovládání*
isBound		*Uzel se stal akvitním nebo naopak byl nahrazen jiným*

Tabulka 4: Parametry uzlu NavigationInfo



Obrázek 2: Geometrický Význam parametru avatarSize

5.1.1 AvatarSize (Rozměry avatara)

Trojice rozměrů `avatarSize` určují postupně poloměr válce, druhý výškové umístění jeho horní podstavy, v jejímž středu jsou uloženy oči a třetí rozměr udává výškové umístění dolní podstavy válce, pod níž se nacházejí avatarovy nohy.

5.1.2 Headlight (Svítilna)

Druhý parametr `headlight` je důležitý pro zkoumání neosvětlených virtuálních světů. Kde není světlo, avatar nic nevidí. Proto je každý avatar vybaven čelním světlem podobně, jako když mají horníci na helmách připevněny malé reflektory. Čelní baterka sleduje směr pohledu avatara. Vypnutí baterky nastavením na hodnotu FALSE se používá tam, kde je umělý svět záměrně osvětlen dalšími zdroji světla.

5.1.3 VisibilityLimit (Zrakové schopnosti)

Zrakové schopnosti avatara (parametr `visibilityLimit`) jsou nastaveny do nekonečna (hodnota 0). Z praktického hlediska je však vhodné nastavit parametr na

hodnotu několik desítek metrů, protože prohlížeč pak nemusí vzdálenější objekty vykreslovat.

5.1.4 Speed (Rychlost)

Poslední charakteristikou avatara je jeho rychlost, která se nastavuje v parametru `speed`. Hodnota parametru se udává v metrech za sekundu. Je-li nastavena rychlost na nulu, avatar se nepohne z místa, může se však otáčet. Toho lze využít ve speciálních případech, kdy avatara postupně navádíme na zajímavá, avšak pouze stacionární stanoviště.

5.1.5 Type (Řízení avatara)

Poslední parametr navigačního uzlu je seznam povolených metod řízení avatara z velitelského stanoviště – parametr `type`. Jsou tři základní způsoby pohybu avatara doplněné dvěma dalšími možnostmi:

„WALK“ Běžná chůze při které se avatar pohybuje po zemi či podložce. Je-li povrch pod ním zvlněný, avatar ho kopíruje a navozuje pocit klesání a stoupání. Je zapnuta detekce kolizí s objekty tak, aby se při pokusu o průchod objektem avatar zastavil.

„FLY“ Stejné chování jako při chůzi s tím, že je vyřazeno působení přitažlivosti.

„EXAMINE“ Způsob, při kterém je avatar nejméně omezován běžnými fyzikálními zákony. Je určen především ke zkoumání objektů. Proto je vypnuta přitažlivost a avatar může kolem objektu kouřit ze všech stran. Je vypnuto testování kolizí, takže objekty jsou průchozí skrz naskrz.

„ANY“ Nejedná se o způsob ovládání, ale o doporučení prohlížeči, že může uživateli povolit přepínání mezi předchozími základními způsoby. Není-li hodnota ANY zadána, lze volit jen mezi těmi způsobe, které jsou v parametru `type` vyjmenovány.

„NONE“ Hodnota NONE se v parametru type objevuje samostatně a určuje, že prohlížeč má skrýt veškeré ovládací prvky. Současně přestane převádět pohyb myši na pohyb avatara. Tento způsobe je vhodný pouze tehdy, pokud jsou ve virtuálním prostředí aktivní tělesa, schopná po doteku teleportovat avatara na další stanoviště.

```
NavigatorInfo{
avatarSize              [0.25 1.6 0.75]
     headlight          TRUE
     visibilityLimit    0.0
     speed              1.0
     type               [„WALK“, „ANY“]
}
```

6 Základní tělesa a transformace

6.1 Základní tělesa

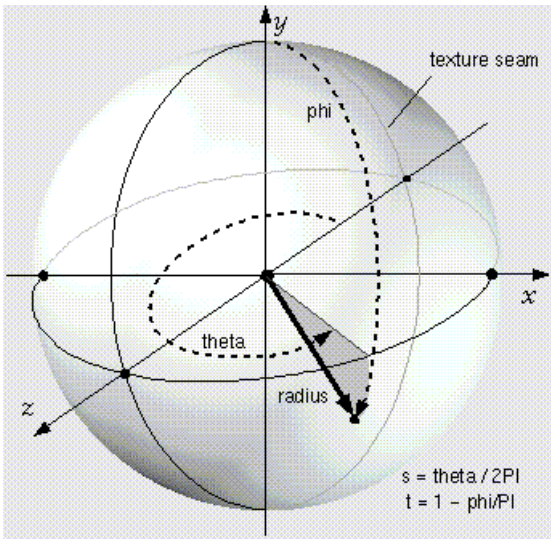
Pro stavitele jsou připravena pouze čtyři základní geometrická tělesa – koule, kvádr, kužel a válec. Všechna jsou iniciálně umístěna tak, aby jejich těžiště bylo v počátku souřadné soustavy. Výjimkou je kužel, který je situován tak, že v počátku souřadnic leží střed jeho osy.

6.1.1 Koule (Sphere)

Nejjednodušším tělesem používaným ve VRML je koule (`Sphere`). Její konstrukce je velice jednoduchá, protože obsahuje pouze jediný parametr a to poloměr (`radius`).

Parametr	**Iniciální hodnota**	**Význam**
radius	1	Nezáporný poloměr koule

Tabulka 5: Parametr uzlu Sphere



Obrázek 3: Umístění a rozměry uzlu Sphere

6.1.2 Kvádr (Box)

Dalším jednoduchým tělesem, které můžeme při sestavování virtuálního světa použít je kvádr (`Box`). Je sestaven ze šesti jednostranných ploch. Pokud není stanoveno jinak, je směr stran shodný se směrem os souřadnicového systému a těžiště je umístěno do počátku souřadnicové soustavy.

Parametr	**Iniciální hodnota**	**Význam**
size	2 2 2	Nezáporné rozměry kvádru

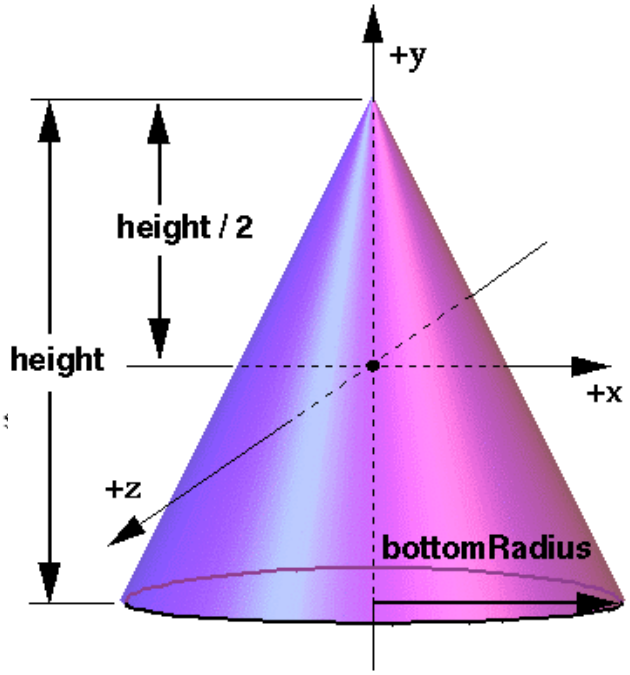
Tabulka 6: Parametr uzlu Box

6.1.3 Kužel (Cone)

Předposledním těleso je kužel (`Cone`). Je tvořeno sítí jednostranných ploch pokrývajících povrch kužele (Obrázek 5). Povrch je tvořen rotačním pláštěm s kruhovou podstavou. Je-li tento druh těles postaven na nějaké podložce, je výhodné pomocí parametrů uzlu vypustit podstavu, která není vidět a tím zrychlit zobrazování modelu.

Parametr	**Iniciální hodnota**	**Význam**
botttomRadius	1	Nezáporný poloměr podstavy kuželu
height	2	Nezáporná výška
side	TRUE	Povelení vykreslování pláště
bottom	TRUE	Povolení vykreslování podstavy.

Tabulka 7: Parametry uzlu Cone



Obrázek 4: Umístění a rozměry uzlu Box

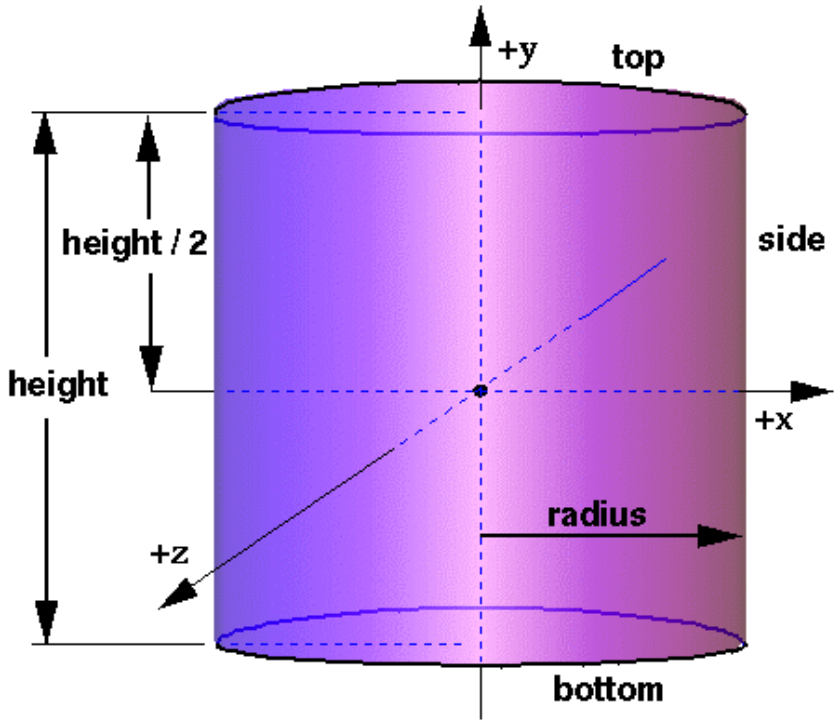
Obrázek 5: Umístění a rozměry uzlu Cone

6.1.4 Válec (Cylinder)

Poslední základní těleso je válec (`Cylinder`). Jak je definován dostatečně názorně ukazuje Obrázek 6. Jeho vlastnosti jsou velmi podobné vlastnostem kužele.

Parametr	Iniciální hodnota	Význam
height	2	Nezáporná výška válce
radius	1	Nezáporný poloměr
side	TRUE	Povolení vykreslování pláště
bottom	TRUE	Povolení vykreslování dolní podstavy
top	TRUE	Povolení vykreslování horní podstavy

Tabulka 8: Parametry uzlu Cylinder



Obrázek 6: Umístění a rozměry uzlu Cylinder

6.2 Transformace

Abychom mohli do virtuálního světa vložit některé z výše uvedených těles, musíme se naučit pracovat se dvěma uzly. Jejich úkolem je vytvářet stromové struktury a shrnovat jednoduší objekty a jejich vlastnosti do jednoho celku.

6.2.1 Transform (Umístění skupiny)

Tento uzel je tzv. skupinový, který definuje souřadnicový systém pro všechny své potomky (zapsané do parametru `children`). Určuje jim měřítko, osu a úhel natočení, posunutí a další.

Parametr	Iniciální hodnota	Význam
scale	1 1 1	Měřítko
scaleOrientation	0 0 0 1	Osa a úhel natočení provedeného před změnou měřítka
rotation	0 0 1 0	Osa a úhel natočení
center	0 0 0	Vztažný bod, vůči kterému se provádí otočení
translation	0 0 0	Posunutí
children	prázdný uzel	Seznam potomků
bboxSize	-1 –1 –1	Kladné délky stran pomocné obálky ve tvaru kvádru; výjimka [-1 –1 –1] indikuje dosud nespecifikovanou velikost kvádru
bboxCenter	0 0 0	Souřadnice středu pomocné obálky ve tvaru kvádru
addChildren		Seznam připojovaných uzlů
removeChildren		Seznam odstraňovaných uzlů

Tabulka 9: Seznam parametrů uzlu Transform

6.2.2 Shape (Zobrazitelný objekt)

Uzel `Shape` provazuje definice geometrických tvarů a vlastností povrchu jednoho objektu

Parametr	Iniciální hodnota	Význam
appearence	NULL	Vzhled povrchu objektu
geometry	NULL	Geometrie objektu

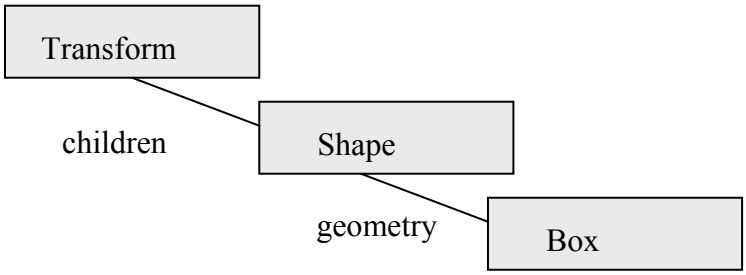
Tabulka 10: Seznam parametrů uzlu Shape

Následující příklad je typickou ukázkou základního tvaru stromu, definujícího polohu tělesa. rodičovský uzel `Transform` má jediného potomka – uzel `Shape`. Ten má opět jednoho potomka, který určuje geometrii tělesa. Parametr `apperance` je zatím nevyužit, a proto je vzhled kvádru nepříliš zajímavý – těleso je matné a bílé, což odpovídá iniciálnímu nastavení barev objektů. Rodičovský uzel má nastaven

parametr měřítka. rodičovský uzel má nastaven parametr měřítka. Výsledný kvádr proto bude mít délku stran 6, 2 a 4 metry, neboť v iniciálním tvaru je reprezentována kostkou o délce hrany 2 metry.

```
#VRML V2.0 utf8
Transform{
     scale 3 1 2
     children Shape {geometry Box{} }
}
```

K takto jednoduchému umístění tělesa odpovídá níže zobrazený strom.



Obrázek 7: Stromová struktura základního tělesa

Pro zapsání téhož samého příkladu je vhodnější následujíc zápis, poněvadž umožňuje pozdější dynamické změny měřítka a polohy pří zachování neměnného poměru stran kvádru.

```
#VRML V2.0 utf8
Transform {
     children Shape { geometry Box {6 2 4} }
}
```

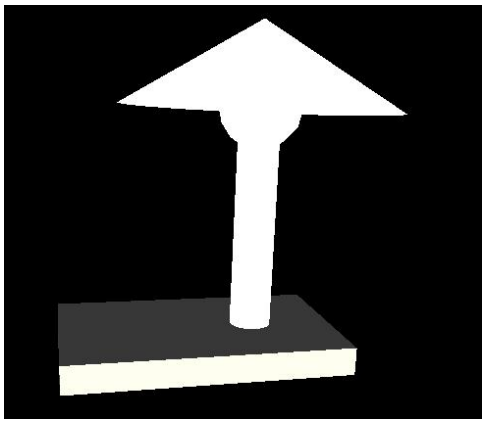
6.2.3 Příkaz DEF a USE

Mnohonásobné opakování popisu takto podobných objektů je z lidského hlediska pracné, z počítačového dokonce zbytečné. Existuje možnost, jak jednou zapsanou informaci opakovaně využít. Jakmile jednou pojmenujeme nějaký uzel (příkazem `DEF`), můžeme v další části souboru vytvořit jeho kopii příkazem `USE`. Vždy musíme uvést jméno, které jsme danému uzlu dříve přiřadili.

Příklad Lampicka.wrl

```
#VRML V2.0 utf8
Viewpoint {
  position -0.006 0.09 0.5
  orientation 0 0 1 0
  fieldOfView 0.785398
  description "Zepredu"
}
DEF Lampa Transform {
  children [
    DEF Stojanek Transform {
      children Shape {
       geometry Box {
         size 0.12 0.02 0.2}
      }
      translation 0 0.01 0.04
    }
    DEF Nozicka Transform {
      children   Shape {
       geometry  Cylinder {
         radius  0.015
         height  0.18
         top     FALSE
         bottom  FALSE}
      }
      translation      0 0.09 0
    }
    DEF Zarovka Transform {
      children   Shape {
       geometry  Sphere {
         radius  0.03}
      }
      translation      0 0.18 0
    }
    DEF Kloboucek Transform {
      children   Shape {
         geometry Cone {
           bottomRadius 0.1
           height 0.06}
      }
      translation      0 0.21 0
    }
  ]
}
```

Výše popsaný příklad lampičky vypadá ve virtuálním světě asi takto. Pro lepší viditelnost jsem obrázek natočila, takže neodpovídají parametry v uzlu `Viewpoint`.



Obrázek 8: Lampička

7 Povrchy těles

Uzel `Shape` má kromě parametru `geometry` ještě druhý parametr – `appearance`. Tento uzel jsme doposud nevyužili, ačkoliv právě on má zásadní vliv na vzhled tělesa, jak ostatně sám jeho název napovídá.

7.1 Appearance (Vzhled povrchu)

Do parametru `appearance` se umísťuje pouze jediný možný uzel, který se nazývá shodně s parametrem – `Appearance`. Dovoluje definovat dva možné vhledy povrchu těles:

- barvu (parametr `material`)
- texturu (parametry `texture` a `textureTransform`)

Parametr	**Iniciální hodnota**	**Význam**
material	NULL	Barevné vlastnosti povrchu
texture	NULL	Textura na povrchu
textureTransform	NULL	Umístění a natočení textury

Tabulka 11: Seznam parametrů uzlu Appearance

Do žádného z parametrů uzlu `Appearance` se nepřiřazují číselné hodnoty, ale opět uzly. Jejich jména jsou odvozena ze jmen parametrů:

- do parametru `material` lze přiřadit pouze uzel `Material`
- do parametru texture lze přiřadit jeden z uzlů ImageTexture, PixelTexture a MovieTexture,
- do parametru textureTransform lze přiřadit pouze uzel `TextureTransform`.

7.1.1 Material (Barevné vlastnosti)

Uzel `material` je určen k zadávání barvy, která bude stejná na celým povrchu. Jméno uzlu je trochu zavádějící. Normálně bychom si pod ním mohli představit takové pojmy, jako jsou například železo, dřevo, papír apod. Ve skutečnosti tento uzel označuje pouze barevné charakteristiky povrchu.

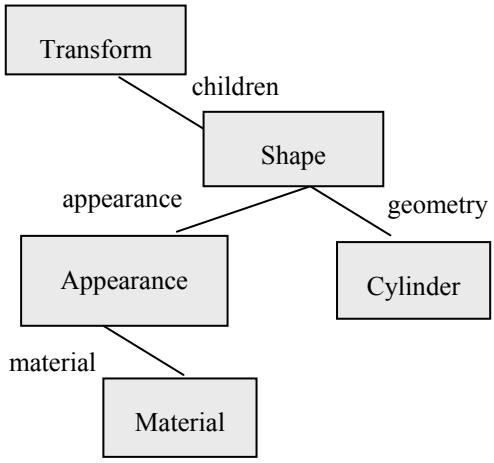
Parametr	Iniciální hodnota	Význam
ambientIntensity	0.2	Světlost povrchu získaná odrazem od okolního světla, tedy bez ohledu na úhel dopadu světla na povrch
diffuseColor	0.8 0.8 0.8	Základní složení barvy povrchu ve složkách RGB
specularColor	0 0 0	Barva světla odraženého od přímých zdrojů světla
shininess	0.2	Rozmazané (0.0) nebo ostré (1.0) odlesky od přímých zdrojů světla
emissiveColor	0 0 0	Vyzařovaná barva, luminiscence; není ovlivňována žádnými zdroji světla
transparency	0	Průhlednost; je-li rovna jedné, těleso zmizí

Tabulka 12: Seznam parametrů uzlu Material

Pro většinu jednoduchým těles vystačíme s nastavením jediného parametru – `diffuseColor`. Ten definuje matný vzhled. Odstín barvy se mění podle úhlu dopadajícího světla. Přímo proti světlu je barva nejjasnější.

Typický zápis takto barevného tělesa může vypadat takto:

```
#VRML V2.0 utf8
Transform{
  children Shape{
    appearance Appearance {
      material Material { diffuseColor 0.5 0.5 1}
    }
   geometry Cylinder{}
  }
}
```



Obrázek 9: Stromová struktura matného modrého válce

7.1.2 Texture

I když použití barvy přináší výrazné zlepšení vzhledu těles, nevěrnějšího vzhledu povrchu dosáhneme teprve využitím textur. Textura je obrazový vzorek, který je nanášen na povrch tělesa. Pro výběr barevného vzoru lze použít několika možností, každá z nich je svázána se samostatným uzlem:

- Obrázek uložený v samostatným souboru (uzel `ImageTexture`)
- Opakující se kombinace barev zapsaná v uzlu (`PixelTexture`)
- Přehrávaná video-sekvence uložená v souboru (`MovieTexture`)

7.1.2.1 ImageTexture (Textura určená obrázkem)

Určení obrázku a jeho případného opakovaného nanášení jako textury na povrchu.

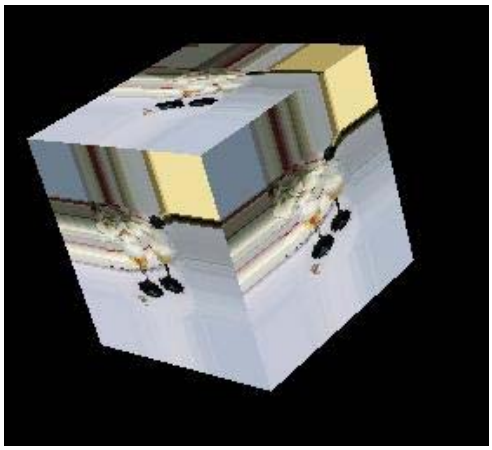
Parametr	**Iniciální hodnota**	**Význam**
url	[]	Seznam adres s umístěním obrázku
repeatS	TRUE	Povolení opakování textury ve vodorovném směru
repeatT	TRUE	Povolení opakování textury ve svislém směru

Tabulka 13: Seznam parametrů uzlu ImageTexture

Povolené formáty obrazových souborů jsou JPEG a PNG, podporován je také GIF a v některých prohlížečích vektorový formát CGM.

Příklad Obrazek.wrl

```
#VRML V2.0 utf8
Transform{
  children
    Shape{
      geometry Box {}
      appearance Appearance {
        texture ImageTexture{
       url "Marriage.jpg"
       repeatS FALSE
       repeatT FALSE
      }
      textureTransform
        TextureTransform{
          scale 2 2
          translation -0.25 -0.25
        }
       }
    }
}
```



Obrázek 10: Použití transformace textury bez opakování (viz barevná příloha)

7.1.2.2 PixelTexture (Textura určená vzorkem)

Popis obdélníkového vzorku barevných bodů a jejich případného opakovaného nanášení jako textury na povrch.

Parametr	Iniciální hodnota	Význam
image	0 0 0	Typ, šířka, výška a definice vzorku barevných bodů
repeatS	TRUE	Povolení opakování textury ve vodorovném směru
repeatT	TRUE	Povelní opakování textury ve svislém směru

Tabulka 14: Seznam parametrů uzlu PixelTexture

7.1.2.2.1 Image (Obrázek)

Obsah parametru image má ze všech parametrů VRML nejsložitější vnitřní strukturu. První dvě čísla znamenají rozměry, tj. počet dále definovaných barev pixelů v osách x a y. Třetí číslo určuje způsob zápisu jednotlivých barev. Může nabývat následujících hodnot:

0 – žádná barva (tato hodnota je rezervována jen pro iniciální nastavení)

1 – odstíny šedi v rozsahu 0 (černá) až 255 (bílá)

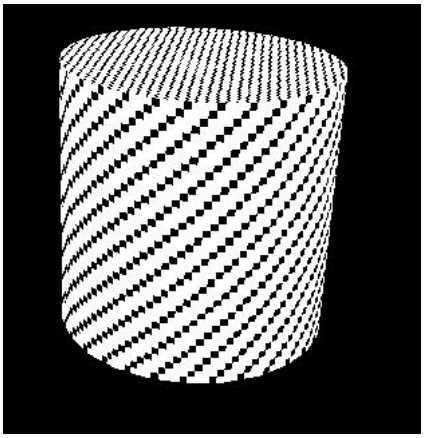
2 – odstíny šedi + informací o průhlednosti (oboje v rozsahu 0 – 255)

3 – barva složená ze složek RGB, každá v rozsahu 0 – 255

4 – barva tvořená složkami RGB a + informací o průhlednosti

Příklad textura.wrl

```
#VRML V2.0 utf8
Transform{
  children Shape{
    geometry Cylinder{}
    appearance Appearance{
      texture PixelTexture{
        image 4 1 1
                0x00 0xFF 0xFF 0xFF
                0xFF 0x00 0xFF 0xFF
                0xFF 0xFF 0x00 0xFF
                0xFF 0xFF 0xFF 0x00}
      textureTransform TextureTransform {scale 31 10}
     }
    }}
```



Obrázek 11: Ukázka šrafovaného válce

7.1.2.3 MovieTexture (Pohyblivá textura)

Určení souboru s animací nebo videosekvencí a jeho případného opakovaného nanášení jak textury na povrchu. Druhou možností je využití samotné zvukové stopy videozáznamu jako zdroje zvuku.

Parametr	**Iniciální hodnota**	**Význam**
url	[]	Seznam adres s umístěním videosouboru
startTime	0	Čas zahájení přehrávání
stopTime	0	Čas ukončení přehrávání
speed	1.0	Rychlost přehrávání; záporná hodnota znamená přehrávání pozpátku
loop	FALSE	Povolení přehrávání ve smyčce
repeatS	TRUE	Povolení opakování textury ve vodorovném směru
repeatT	TRUE	Povolení opakování textury ve svislém směru
duration_changed		Původní délka sekvence v sekundách; je vyslána po načtení souboru do paměti
isActive		Bylo zahájeno nebo ukončeno přehrávání sekvence

Tabulka 15: Seznam parametrů uzlu MovieTexture

Povolený formát videosouboru je MPEG-1, některé prohlížeče podporují také animovaný GIF.

7.1.3 TextureTransform (Transformace textury)

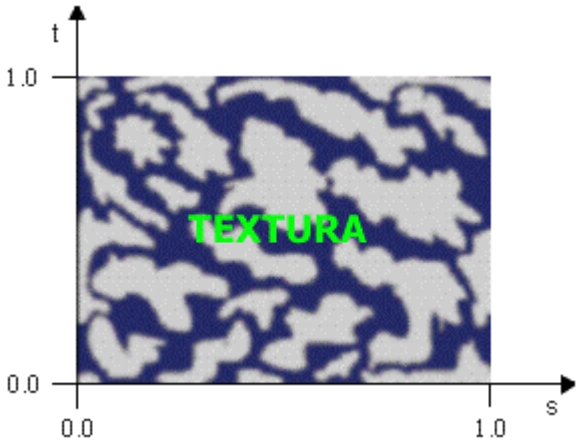
Výše uvedené tři druhy uzlů popisující zdroj dat, ze kterého budou získávána obrazová data použitá jako textura. Další uzel je pak určen k tomu, aby definoval způsob mapování textury na povrch.Tímto uzlem je `TextureTransform`, který popisuje posunutí, natočení a změna měřítka textury.

Parametr	**Iniciální hodnota**	**Význam**
translation	0 0	Posunutí
roration	0	Úhel otočení vzhledem ke vztažnému bodu
scale	1 1	Změna měřítka vhledem ke vztažnému bodu
center	0 0	Poloha vztažného bodu

Tabulka 16: Seznam parametrů uzlu TextureTransform

Transformace jsou prováděny v pevném pořadí. Nejprve posunutí (translation), poté otočení (rotation) a na závěr změna měřítka (scale). Otočení a změna měřítka jsou prováděny vzhledem ke společnému vztažnému bodu (center).

Transformace se nevztahuje přímo k textuře, nýbrž k jejímu definičnímu oboru mapovanému do povrchové oblasti o rozměru 1x1.



Obrázek 12: Souřadnicový systém textury

7.2 Použití textury na základní tělesa

7.2.1 Koule

Při namapování textury na kouli je pokryt celý povrch koule. Textura je nabalena na kouli odzadu (úhel `phi` – Obrázek 3) ve směru proti pohybu hodinových ručiček (úhel `theta` na Obrázku 3). Spojení textury (šev) je na zadní straně koule na rovině *yz*.

7.2.2 Kvádr

Textura je mapována jednotlivě na každou stranu krychle (na každou stranu krychle je mapována celá textura). Na přední, zadní, pravé a levé straně krychle je textura mapována tak, že levý dolní roh každé strany odpovídá počátku mapovacích souřadnic *s* a *t*. Na horní straně krychle, pokud směřuje k pozorovateli (rotace o 90° okolo osy x) také odpovídá levý dolní roh strany počátku texturovacích souřadnic *s* a *t* (viz Obrázek 12). Stejně je tomu i se spodní stranou krychle, směřuje-li k pozorovateli (rotace o 90°okolo osy x opačným směrem).

7.2.3 Kužel

Na stěnu kužele je textura nabalena odzadu proti směru hodinových ručiček (při pohledu shora). Svislé spojení textury (šev) je na zadní straně v rovině *yz*. Na podstavu je z textury o jednotkové velikosti vykrojeno kolečko. Toto kolečko je na mapováno na podstavu tak, že pokud podstava směřuje k pozorovateli (rotace o 90° okolo osy x) odpovídá texturovací souřadnice *s* vodorovném směru a souřadnice *t* svislému směru (viz Obrázek 12).

7.2.4 Válec

Textura namapovaná na válec je jinak mapována na stěnu válce, jinak na obě podstavy. Na stěnu je textura nabalena odzadu proti směru hodinových ručiček (při pohledu zhora). Svislé spojení textury (šev) je na zadní straně v rovině *yz*. Na obě podstavy je textura aplikovaná jinak, z prostředku textury o jendotkové velikosti je vykrojeno kolečko a to je namapováno na podstavy. Na horní podstavě válce, pokud směřuje k pozorovateli (rotace o 90° okolo osy x) odpovídá texturovací souřadnice s vodorovnému směru a souřadnice t svislému směru (viz Obrázek 12). Stejně je tomu i se spodní podstavou, pokud směřuje k pozorovateli (rotace o 90° okolo osy x v opačném směru). Na texturu je možné aplikovat transformace pomocí uzlu `TextureTransform`.

8 Obecná tělesa

Je samozřejmé, že nám na vytvoření virtuálního světa nestačí pouze čtyři základní tělesa. Potřebujeme další obecnější uzly a máme jich rovnou šest. Jsou to následující:

1. Množina ploch (`IndexedFaceSet`)
2. Opláštění (`Extrusion`)
3. Výšková mapa (`ElevationGrid`)
4. Množina čar (`IndexedLineSet`)
5. Množina bodů (`PointSet`)
6. Nápis (`Text`)

Na rozdíl od jednoduchých těles mají tyto uzly poměrně složitou strukturu. V mnoha případech jsou jejich parametry tvořeny dalšími, pomocnými uzly. Touto složitostí platíme za možnost téměř libovolného tvarování objektů.Ve virtuálních světech jsou téměř všechny objekty modelovány pomocí rovinných ploch.

Plocha může být až překvapivě složitý útvar, vyznačující se řadou vlastností. Většina z nich má přitom přímý dopad na efektivitu zobrazení:

- **Rovinná plocha** – má všechny své vrcholy položeny do téže roviny.
- **Konvexní plocha** – tato plocha jde lépe rozdělit na trojúhelníky a dále zpracovat jednoduchými metodami.
- **Jednostranná plocha** – je zobrazována pouze z jedné strany; touto plochou je tvořen povrch jednoduchých těles.
- **Oboustranná plocha** – zvyšují výpočetní nároky na zobrazení; zbytečně zvyšují výpočetní nároky

8.1 IndexedFaceSet (Množina ploch)

Tento uzel patří mezi ty nečastěji používané pro popis plochy. Dovoluje definovat zcela obecné těleso nebo libovolně zakřivenou prostorovou plochu tím, že popíše malé plošky, které daný objekt pokrývají. Každá plocha z množiny se definuje zapsáním indexů vrcholů, nikoliv konkrétním uvedením jejich souřadnic. Pokud je některý vrchol sdílen několika plochami, stačí zapsat jeho souřadnice pouze jednou a zapamatovat si jeho pořadí v seznamu vrcholů. U všech ploch, které tento vrchol používají je pak uvedeno jediné číslo, a to jeho pořadové číslo v seznamu vrcholů.

Parametr	Iniciální hodnota	Význam
coord	NULL	Seznam vrcholů v uzlu `Coordinate`
normal	NULL	Seznam normál v uzlu `Normal`
color	NULL	Seznam barev v uzlu `Color`
texCoord	NULL	Seznam souřadnic textury v uzlu `Texture-Coordinate`
coordIndex	[]	Posloupnost indexů vrcholů jednotlivých ploch (zakončené číslem –1)
normalIndex	[]	Posloupnosti indexů normál pro jednotlivé vrcholy či jednotlivé plochy
texCoordIndex	[]	Posloupnosti indexů souřadnic textury pro jednotlivé vrcholy
colorPerVertex	TRUE	Barvy v parametru `colorIndex` se vztahují na vrcholy, jinak na plochy
normalPerVertex	TRUE	Normály v parametru `normalIndex` se vztahují na vrcholy, jinak na plochy
convex	TRUE	Všechny plochy jsou konvexní
ccw	TRUE	Přivrácené strany všech ploch jsou zadávány proti směru hodinových ručiček
solid	TRUE	Všechny plochy jsou jednostranné
creaseAngle	0	Nezáporný úhel, až do kterého jsou dvě sousední plochy považovány za oblé
set_coordIndex		Změna parametru `coordIndex`
set_normalIndex		Změna parametru `normalIndex`
set_colorIndex		Změna parametru `colorIndex`
set_texCoordIndex		Změna parametru `texCoordIndex`

Tabulka 17: Seznam parametrů uzlu IndexedFaceSet

Uzel `IndexedFaceSet` se podobně jako základní tělesa zapisuje do parametru geometry rodičovského uzlu `Shape`. Stejně je tomu i u dalších pěti uzlů níže uvedených.

8.1.1 Coordinate (Souřadnice prostorových bodů)

Definuje souřadnice bodů v prostoru, používaných jako vrcholy v obecných geometrických objektech.

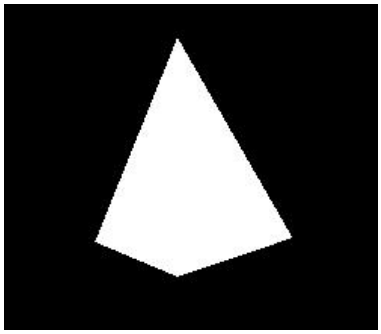
Parametr	Iniciální hodnota	Význam
point	[]	Seznam bodů

Tabulka 18: Parametr uzlu Coordinate

Ukázka pravidelného čtyřbokého jehlanu – jak se pracuje s vrcholy a plochami.

Příklad IndexFaceSet.wrl

```
#VRML V2.0 utf8
Transform{
  children Shape{
    geometry IndexedFaceSet{
      coord Coordinate { point [
          1 0 0, 0 0 -1,          # body podstavy
        -1 0 0, 0 0 1,
          0 2 0]}                 #vrchol jehlanu
      coordIndex [ 3 2 1 0 -1,    #podstava
           0 1 4 -1, 1 2 4 -1,    #stena
           2 3 4 -1, 3 0 4 -1]
    }
  }
}
```



Obrázek 13: Ukázka jehlanu

8.1.2 Normal (Normálové vektory)

Definice normál v trojrozměrném prostoru, doplňujících informace v ploškových geometrických objektech.

Parametr	Iniciální hodnota	Význam
vector	[]	Seznam normálových vektorů

Tabulka 19: Parametr uzlu Normal

Všechny vektory by měly mít jednotkovou délku. Mnoho prohlížečů však pro jistotu veškeré normálové vektory při načítání normalizuje, tj. upravuje jejich délku na jedna.

8.1.3 Color (Barvy)

Definice difúzních barev používaných k obarvení prvků v obecných geometrických objektech.

Parametr	Iniciální hodnota	Význam
color	[]	Seznam barev v modelu RGB

Tabulka 20: Parametr uzlu Color

8.1.4 TextureCoordinate (Souřadnice textury)

Definice bodů v rovině, které slouží jako vztažné body pro nanesení textury v ploškových geometrických uzlech.

Parametr	Iniciální hodnota	Význam
point	[]	Seznam bodů v rovině

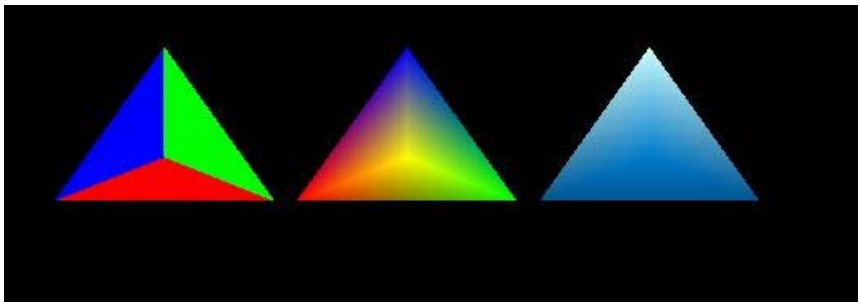
Tabulka 21: Parametr uzlu TextureCoordinate

Následující příklad ukazuje různé způsoby nastavení parametrů `IndexedFaceSet` pro útvar, tvořený třemi trojúhelníky se společným vrcholem.Všechny trojúhelníky leží v jedné rovině, přesto lze pomocí barev vytvořit iluzi, že jejich společný vrchol poněkud vystupuje.

Příklad Trojuhelniky.wrl

```
#VRML V2.0 utf8
Transform{
  children [
    Transform{
      translation -2.2 0 0
      children Shape {
     geometry IndexedFaceSet {
       coord DEF VRCHOLY Coordinate {
           point [-1 0 0, 1 0 0,
                  0 1.4 0, 0 0.4 0]
```

```
        }
        color DEF BARVY Color {
          color [1 0 0, 0 1 0, 0 0 1, 1 1 0]
        }
        coordIndex[0 1 3 -1, 1 2 3 -1, 2 0 3 -1]
        colorPerVertex FALSE
         }
       }
      }
      Transform {
        children Shape{
       geometry IndexedFaceSet{
        coord USE VRCHOLY
        color USE BARVY
        coordIndex [0 1 3 -1, 1 2 3 -1, 2 0 3 -1]
       }
        }
      }
      Transform{
        translation 2.2 0 0
        children Shape{
       appearance Appearance{
        material Material{
          diffuseColor 0 0.6 1
          specularColor 1 1 1
           }
       }
       geometry IndexedFaceSet{
        coord USE VRCHOLY
        normal Normal {vector [-.58 -.58 .58, .58 -.58 .58,
                          0 0 1, 0 -0.6 0.8]}
        coordIndex [0 1 3 -1, 1 2 3 -1, 2 0 3 -1]
       }
        }
      }
    ]
}
```



Obrázek 14: Rovinné trojúhelníky se stejnou geometrií, ale s různými parametry.
Zleva postupně plochy s jedinou barvou, plochy s barvami ve vrcholech, jednobarevné plochy s normálami ve vrcholech. (viz barevná příloha)

8.2 Extrusion (Opláštění)

Tento uzel je určen pro popis takových objektů, jejichž povrch vznikne postupným přesouváním dvourozměrného profilu po prostorové trajektorii. Profil je tvořen jedinou lomenou čárou, která může být při přesunu transformována. Počet jejích bodů se však nemění. Také trajektorie je definována jedinou lomenou čárou.

Odborně se takové objekty nazývají translační nebo rotační tělesa. Vždy je lze charakterizovat rovinným profilem, který je dále posouván a otáčen v prostoru.

Počet parametrů uzlu `Extrusion` je oproti množině ploch nižší, některé z nich mají dokonce stejná jména i význam jako parametry množiny ploch:

Parametr	**Iniciální hodnota**	**Význam**
crossSection	[1 1, 1 -1, -1 -1, -1 1 1 1]	Posloupnost bodů v rovině určující profil, tj. obrysovou křivku
spine	[0 0 0, 0 1 0]	Posloupnost bodů v prostoru určující trajektorii, po které je tažen obrys
scale	1 1	Seznam kladných měřítek (v rovině xz) pro každou polohu obrysu
orientation	0 0 1 0	Seznam otočení pro každou polohu obrysu
beginCap	TRUE	Povolení vykreslování dolní podstavy
endCap	TRUE	Povolení vykreslování horní podstavy
convex	TRUE	Obrys je konvexní
ccw	TRUE	Obrys je zadán proti směru hodinových ručiček
solid	TRUE	Všechny plochy jsou jednostranné
creaseAngle	0	Nezáporný úhel, až do kterého jsou dvě sousední plochy považovány za oblé
set_crossSection		Změna parametru `crossSection`
set_spine		Změna parametru `spine`
set_scale		Změna parametru `scale`
set_orientation		Změna parametru `orientation`

Tabulka 22: Seznam parametrů uzlu Extrusion

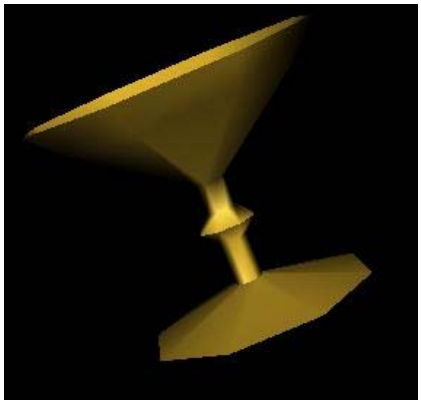
Jak je vidět v předchozí tabulce, iniciální hodnoty definují jednoduché těleso se čtvercovým uzavřeným profilem (`crossSection`). Profil je opakovaně umísťován do prostoru podle poloh, které jsou uvedeny v parametru `spine`. Standardně je tedy čtverec umístěn nejprve do počátku souřadnic a poté o 1 metr výše ve směru osy y. Opláštěním těchto dvou ploch profilu vzniknou čtyři boční plochy. Každá plocha (kromě první) může být z následujícího profilu zmenšena či zvětšena (`scale`) a navíc otočen (`orientation`).

Standardně je dále vytvořena dolní podstava určená první polohou profilu (`beginCap`) a horní podstava určená poslední polohou profilu (`endCap`). Není-li stanoveno jinak, všechny vzniklé plochy jsou jednostranné (`solid`).

Následující příklad ukazuje vytvoření jednoduchého poháru, který je definován uzavřeným profilem z osmi vrcholů a trajektorií o osmi bodech. Povrch je z materiálu připomínající zlato.

Příklad Pohar.wrl

```
#VRML V2.0 utf8
Transform{
  children Shape{
    appearance Appearance {
      material Material {
      diffuseColor 1 0.8 0.2
      specularColor 1 1 1}
    }
    geometry Extrusion{
      crossSection [
1 0, 0.7 0.7,
            0 1, -0.7 0.7,
            -1 0, -0.7 -0.7,
            0 -1, 0.7 -0.7
            1 0]
      spine [   0 0   0,
                0 0.2 0,
                0 0.5 0,
                0 0.6 0,
                0 0.7 0,
                0 0.9 0,
                0 1.4 0,
                0 1.6 0]
        scale [ 0.8 0.8,
                  0.1 0.1,
                  0.1 0.1,
                  0.2 0.2,
                  0.1 0.1,
                  0.1 0.1,
                  0.8 0.8,
                  1   1]
        solid FALSE
        endCap FALSE
        creaseAngle 1
      }
    }
  }
```



Obrázek 15: Zlatavý pohár definovaný uzavřeným profilem z osmi vrcholů a trajektorií o osmi bodech. (viz barevný příloha)

8.3 ElevationGrid (Výšková mapa)

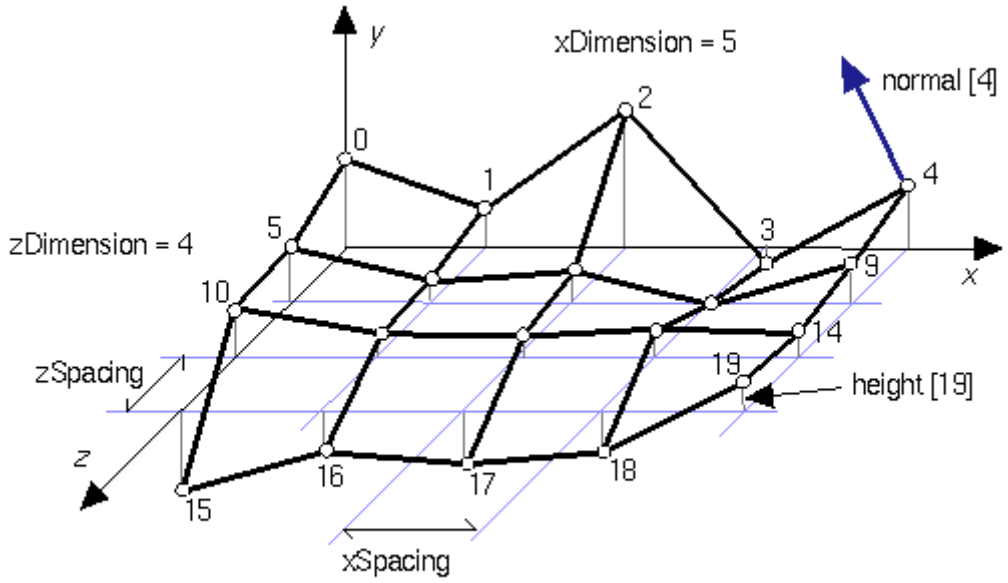
Poslední možností jak se ve VRML zhostit popisu plochy je výšková mapa (ElevationGrid). Slouží především k vytvoření základního tvaru krajiny, definované pomocí pravoúhlé sítě vrcholů s příslušnými výškami.

K vytvoření základního tvaru krajiny využívá pravoúhlé sítě vrcholů s příslušnými výškami jak je to možné vidět na Obrázku 16. Implicitně je mapa umístěna do roviny XZ, což je rovina kde se předpokládá podložka. Počty vrcholů ve směru jednotlivých os se mohou lišit, rovněž i vzdálenosti mezi jednotlivými řadami lze pomocí parametrů snadno měnit.

Přehled základních parametrů tohoto uzlu je uveden v následující tabulce.

Parametr	**Iniciální hodnota**	**Význam**
color	NULL	Seznam barev v uzlu `Color`
normal	NULL	Seznam normál v uzlu `Normal`
texCoord	NULL	Seznam souřadnic textury v uzlu `Texture-Coordinate`
xDimension	0	Počet vzorků (vrcholů sítě) v ose x
zDimension	0	Počet vzorků v ose z
xSpacing	1.0	Kladná vzdálenost mezi vzorky v ose x
zSpacing	1.0	Kladná vzdálenost mezi vzorky v ose z
height	[]	Pole výšek všech vrcholů sítě
colorPerVertex	TRUE	Barvy v parametru `color` se vztahují na vrcholy, jinak na celé plochy sítě
normalPerVertex	TRUE	Normály v parametru `normal` se vztahují na vrcholy, jinak na plochy
ccw	TRUE	Přivrácená strana mapy je vidět při pohledu shora (proti ose y)
solid	TRUE	Mapa je jednostranná
creaseAngle	0	Mezní úhel pro určení hladkého napojení sousedních ploch
set_height		Změna parametru `height`

Tabulka 23: Seznam parametrů uzlu ElevationGrid

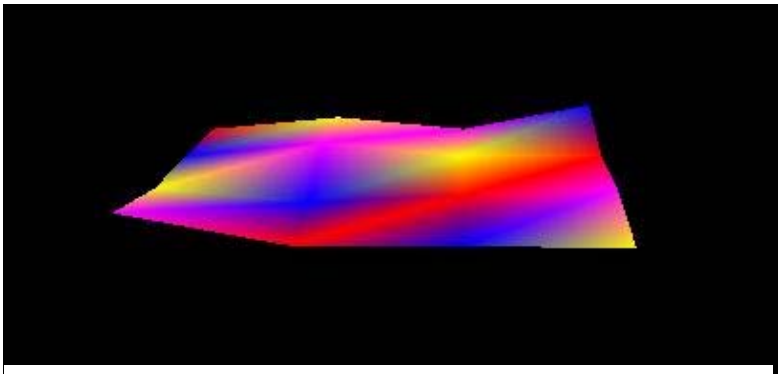


Obrázek 16: Parametry výškové mapy

Výšková mapa se nejčastěji kombinuje s obrazovou texturou. Geometrie mapy může být doplněna údaji o barvách, normálách a souřadnicích textury pro jednotlivé vrcholy.

Příklad ElevationGrid.wrl

```
#VRML V2.0 utf8
WorldInfo{
 info["ElevationGrid"]
}
NavigationInfo{
type "EXAMINE"}
Viewpoint{
  position 20 30 100
  orientation 1 0 0 -.3}
Shape{
 geometry ElevationGrid{
  xDimension 4
  zDimension 4
  xSpacing 10
  zSpacing 10
  height[
0, 1, 0, 2,
1, 2, 1, 1,
2, 1, 1.5, 2,
4, 2, 2, 1.8,
  ]
  solid FALSE
  creaseAngle 2
  color Color{
   color [
1 0 0, 1 1 0, 1 0 1, 0 0 1,
0 0 1, 1 0 1, 1 1 0, 1 0 0,
1 1 0, 0 0 1, 1 0 0, 1 0 1,
1 0 1, 1 0 0, 0 0 1, 1 1 0
   ]
  }
 }
}
```



Obrázek 17: Barevná výšková mapa (viz barevná příloha)

8.4 IndexedLineSet (Množina čar)

Uzel `IndexedLineSet` popisuje množinu lomených čar v prostoru. Vrcholy jsou definovány v poli `coord` a jednotlivé segmenty lomených čar jsou určeny seznamem indexů do pole vrcholů, tento seznam je uložen v poli `coordIndex`. Oddělovačem v tomto poli je –1, její výskyt znamená, že skončila definice jednoho segmentu a začíná definice dalšího.

Parametr	**Iniciální hodnota**	**Význam**
coord	NULL	Seznam vrcholů v uzlu Coordinate
color	NULL	Seznam barev v uzlu Color
coordIndex	[]	Seznam indexů do pole vrcholů, definovaných v poli coord. Je-li nejvyšší index v poli N, pole coord by mělo obsahovat N+1 souřadnic.
colorIndex	[]	Seznam indexů do pole barev, definovaných v poli color.
colorPerVertex	TRUE	Určuje, zda je v uzlu definována barva pro každý vrchol (TRUE) nebo pro každou lomenou čáru (FALSE).
set_coordIndex		Nastavuje hodnoty v poli coordIndex
set_colorIndex		Nastavuje hodnoty v poli colorIndex

Tabulka 24: Seznam parametrů uzlu IndexedLineSet

Velikost bodů a tloušťku čar nelze stanovit. Tyto hodnoty bývají shodné s velikostí jednoho bodu na obrazovce. Při změně vzdálenosti u nich proto dochází k relativnímu zmenšení či zvětšení vůči okolním objektům.

Čáry a body je třeba umisťovat těsně nad povrch jiných objektů (ploch), nikoliv přesně na ně. Vlivem zaokrouhlovacích chyb by při zobrazování mohlo dojít k jejich zakrytí.

8.5 PointSet (Množina bodů)

Uzel `PointSet` definuje soubor bodů v prostoru. Tento uzel se může objevit pouze v poli geometry uzlu `Shape`. Obsahuje pouze dva parametry, do nichž se zapisují uzly se souřadnicemi bodů a jejich barvami. Počet barev by měl být shodný s počtem bodů.

Parametr	Iniciální hodnota	Význam
coord	NULL	Obsahuje uzel `Coordinate` se seznamem souřadnic bodů
color	NULL	Obsahuje uzel `Color` se seznamem barev, jedna pro každý bod

Tabulka 25: Seznam parametrů uzlu PointSet

Nechceme-li definovat barvy pro každý bod zvlášť, ponecháme parametr `color` prázdný a zadáme jedinou barvu materiálu do sourozeneckého uzlu `Appearance`.

Množiny čar a bodů mohou dobře zastoupit textury. Nejsou pochopitelně vkládány do uzlu `Appearance`, nýbrž je s nimi zacházeno jako s geometrickými objekty. Následující příklad ukazuje, jak za pomocí tří čar a šestnácti bodů vyrobit dopisní obálku.

Body, které symbolizují místo pro zámku, nemají v uzlu `PointSet` přiřazeny individuální barvy. Namísto toho je použita společná „luminiscenční“ červená barva v sourozeneckém uzlu `Appearance` u společného rodiče `Shape`. Zelené čáry určující linky pro adresu jsou naopak obarveny barvou uvnitř uzlu `IndexedLineSet`.

Příklad Obalka.wrl

```
#VRML V2.0 utf8
Transform{
  children [
    Shape {geometry Box { size 12 8 0.1} }
    DEF RAMECEK Transform {
      translation 3.5 1.5 0.06
      children Shape {geometry
     PointSet {coord Coordinate {
       point [0 0 0, 0.3 0 0, 0.7 0 0, 1 0 0,
            1 0.3 0, 1 0.6 0, 1 0.9 0, 1 1.2 0,
            1 1.5 0, 0.3 1.5 0, 0.7 1.5 0, 0 1.5 0,
            0 1.2 0, 0 0.9 0, 0 0.6 0, 0 0.3 0]
       }
```

```
        }
        appearance Appearance {material Material
          {emissiveColor 1 0 0}}
         }
       }
       Transform {
         translation 0 -2 0.06
         children Shape { geometry
        IndexedLineSet{ color Color {color 0 1 0.3}
          coord Coordinate {point [0 0 0, 5 0 0, 0 1 0, 5 1 0,
                          0 2 0, 5 2 0]}
          coordIndex[0 1 -1, 2 3 -1, 4 5 -1]
          colorPerVertex FALSE
          colorIndex [0, 0, 0]
        }
         }
       }
    ]
}
```

Obrázek 18: Ukázka použití čar a bodů

8.6 Text (Nápis)

Ve virtuálním světě bychom očekávali od nápisů 3D provedení, ale budeme zklamáni. Virtuální světy se snaží co nejvíce přiblížit světům skutečným, ve kterých jsou nápisy plošné, namalované většinou na rovinném podkladu.

Uzel `Text` vykreslí zadané řetězce, které jsou vidět z obou stran a je umístěn do roviny z = 0. Atributy textu definuje uzel `FontStyle` v poli fontStyle. Pokud je zadáno více řetězců, každý je vykreslen na nový řádek, mezera mezi řádky je definována v uzlu `FontStyle`.

Parametr	Iniciální hodnota	Význam
fontStyle	NULL	Styl písma v uzlu FontStyle.
string	[]	Řetězec nebo řetězce, které mají být zobrazeny, v kódování UTF-8.
maxExtent	0.0	Nezáporná hodnota omezující maximální rozměr nápisu ve směru daném stylem písma; hodnota 0 ruší jakékoliv omezení.
length	[]	Seznam nezáporných délek pro každý z řetězců; hodnota 0 znamená libovolnou délkou.

Tabulka 26: Seznam parametrů uzlu Text

Texty, které mají být zobrazovány, se zapisují do parametru `string`. V něm může být několik textových řetězců, každý z nich je vykreslen na novém řádku. Dále lze určit šířku, kterou nesmí text přesáhnout. Pokud jediný z textových řetězců tento rozměr přesáhne, jsou rovnoměrně stlačeny všechny nápisy v uzlu `Text`.

8.6.1 FontStyle (Styl písma)

Komplexní popis písma podle zadaného fontu, jeho velikosti, umístění a směru psaní.

Parametr	Iniciální hodnota	Význam
language	„“	Dvojznaková zkratka použité abecedy
family	„SERIF“	Seznam rodin písma
style	„PLAIN“	Styl pro danou rodinu písma
size	1.0	Kladná výška písma
spacing	1.0	Nezáporná hodnota, definující vzdálenost mezi řádky jako spacing x size
horizontal	TRUE	Psaní ve vodorovném směru
leftToRight	TRUE	Psaní na řádku zleva doprava
topToBottom	TRUE	Psaní řádků shora dolů
justify	„BEGIN“	Hlavní a vedlejší způsob umístnění textu

Tabulka 27: Seznam parametrů uzlu FontStyle

8.6.1.1 Language

Definice použitého jazyka, ve tvaru dvouznakového kódu jazyka (např.: Angličtina = „en“), následovaného dvouznakovým kódem oblasti (např.: Taiwan = „TW“), kde se jazyk v něčem liší podle národnosti. Prázdný řetězec znamená, že se použije systémové nastavení.

8.6.1.2 Family

Definuje rodinu fontu. Lze vybírat mezi fonty patkovými – *SERIF* (např. Times Roman), bezpatkovými – *SANS* (např. Arial, Helvetica) nebo s konstantní šířkou znaku – *TYPEWRITER* (např. Courier). Jaký font přesně prohlížeč použije už definováno není.

Hondnota „“ znamená jendoduché písmo.

8.6.1.3 Style

Definuje styl písma. Lze vybírat mezi jednoduchým písmem – *PLAIN*, tučným písmem – *BOLD*, kurzívou – *ITALIC* nebo tučnou kurzívou – *BOLD ITALIC*.

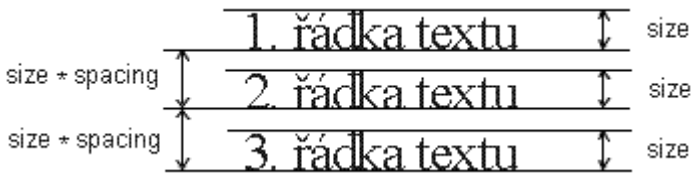
Hodnota „“ znamená jednoduché písmo.

8.6.1.4 Size

Výška každého řádku textu psané horizontálně nebo šířka každé řádky textu psané vertikálně.

8.6.1.5 Spacing

Definuje vzdálenost mezi řádky (viz Obrázek 19), jestli půjde o mezeru mezi sloupci nebo mezi řádkami záleží na nastavení hodnoty horizontal. Např. hodnota 1 znamená text s jednoduchými mezerami, hodnota 2 znamená dvojité řádkování



Obrázek 19: Parametry uzlu FontSize

8.6.1.6 Horizontal

Definuje hlavní směr psaní textu, hodnota *TRUE* znamená vodorovně, hodnota *FALSE* znamená svisle

8.6.1.7 LeftToRight

Pro text psaný ve vodorovném směru (`horizontal` = *TRUE*) definuje, zda je text psán ve směru zleva doprava – *TRUE*, nebo zprava doleva – *FALSE*. Je-li hlavní směr psaní textu svisle (`horizontal` = *FALSE*), znamená toto pole, kde bude následovat další sloupec textu, zda vpravo (`leftToRight` = *TRUE*) či vlevo (`leftToRight` = *FALSE*) od předešlého sloupce.

8.6.1.8 TopToBottom

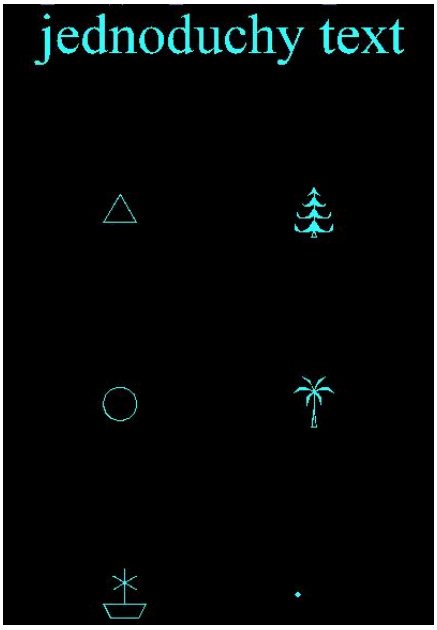
Pro text psaný ve svislém směru (`horizontal` = *FALSE*) definuje, zda je text spán ve směru shora dolů – *TRUE*, nebo zdola nahoru – *FALSE*. Je-li hlavní směr psaní textu vodorovně (`horizontal` = *TRUE*), znamená toto pole, kde bude následovat další řádka textu, zda nad (`topToBottom` = *FALSE*) či pod (`topToBottom` = *TRUE*) předešlou řádkou.

8.6.1.9 Justify

První hodnota definuje, zda jsou řádky nebo sloupce zarovnány na začátky, na středy nebo na konce. Např. pro vodorovný text psaný odleva doprava hodnot *BEGIN* znamená doleva zarovnaný text, hodnota *MIDDLE* znamená text zarovnaný na střed a hodnota *END* určuje text zarovnaný na pravý okraj. Druhá hodnota v poli indikuje zarovnání celého bloku textu ve vedlejším směru. Není-li zadaná žádná hodnota, implicitní zarovnání ve vedlejším směru je *FIRST*.

Příklad Text.wrl

```
#VRML V2.0 utf8
Transform {children [
  Transform {
    translation -1 1 0
    children Shape {
      appearance DEF MODRA Ap-
pearance {
     material Material {
       diffuseColor 0.2 1 1}
        }
      geometry Text{
     string "jednoduchy text"}
    }
  }
  Transform {
      children Shape {
        geometry Text {
       string ["CAU","baf"]
       fontStyle FontStyle {
         family "Symap"
 style "BOLD"
horizontal FALSE spacing 1
         size 3
       }
        }
        appearance USE MODRA
      }
    }
]}
```



Obrázek 20: Ukázka textu a druhu písma

Používejte uzel `Text` co nejméně. V prohlížeči je každé písmenko převáděno na mnoho trojúhelníků, což výrazně zpomaluje zobrazování. Často je lepší místo uzlu `Text` použít obrazovou texturu, byť její vzhled není tak pěkný.

9 Iluze prostoru

Trojrozměrný prostor vytvořený v paměti počítače by byl jen směsicí nevýrazných objektů, kdybychom do něj nevložili světlo a neumožnili povrchu objektů reagovat s ním. V reálném světě existence světla znamená základní podmínku našeho vidění. Světelné paprsky umocňují dojem prostoru, který je nám prezentován na ploché obrazovce.

K zvýšení dojmu reality přispívá značnou měrou i zvuk. Jednoduché zvuky ujišťují návštěvníka virtuálního světa o tom, že objekty se chovají stejně jako ve skutečnosti. Dalším Významem zvuků spočívá v tom, že usnadňují orientaci v trojrozměrném prostoru.

V neposlední řadě zde máme také možnost ovlivnit virtuální svět pomocí pozadí, bilbordů a také mlhou, ve které se objekty postupně ztrácejí.

9.1 Zdroje světla

Ve virtuálním prostoru existují čtyři druhy světelných zdrojů, tedy objektů, ze kterých se šíří světelné paprsky.

1. čelní svítilna avatara
2. zdroj rovnoběžných paprsků (`DirectionalLight`)
3. bodový zdroj (`PointLight`)
4. reflektor, směrový zdroj (`SpotLight`)

O avatarově čelní svítilně jsme mluvili v souvislosti s uzlem `NavigationInfo` v kapitole NavigationInfo (Ovládání avatara). Protože se v prostoru pohybuje právě jeden avatar, existuje právě jedna čelní svítilna. Zbylé tři zdroje světla se mohou vyskytovat v mnoha exemplářích.

Po umístění a nastavení zdrojů světla je vhodné vypnout avatarovu čelní svítilnu. Její směrový svit potlačuje optické jevy vzniklé použitím individuálních světelných zdrojů.

9.1.1 DirectionalLight (Směrový zdroj světla)

Uzel `DirectionalLight` definuje zdroj směrového světla, jehož pomyslné paprsky směrují podél zadaného vektoru v prostoru. Světlo definované tímto uzlem osvětluje všechny své „sourozence“, tzn. všechny potomky skupinového uzlu, jehož potomkem je i uzel `DirectionalLight` a všechny další potomky těchto uzlů. Intenzita paprsků je konstantní, není ovlivňována žádnou vzdáleností.

Parametr	**Iniciální hodnota**	**Význam**
direction	0 0 –1	Směr světla určený vektorem z počátku do definovaného bodu
color	1 1 1	Barva paprsků
intensity	1	Intenzita světla v rozmezí 0 až 1
ambientIntensity	0	Hodnota pole `intensity` vynásobená tímto polem definuje příspěvek tohoto světla k celkové intenzitě světla ve scéně
on	TRUE	Zapnutí či vypnutí světelného zdroje

Tabulka 28: Seznam parametrů uzlu DirectionalLight

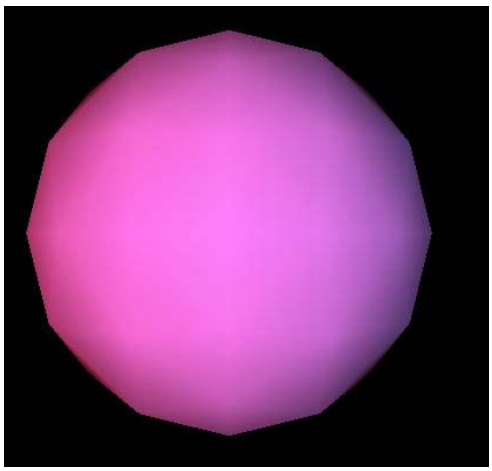
Příklad Directional.wrl

```
#VRML V2.0 utf8

DirectionalLight {
  on                TRUE
  intensity         1
  ambientIntensity  1
  color             1  0  0
  direction         0.5 0 -1
}

Viewpoint {
  position 0 0 8
  description "Entry"
}

Shape {
  appearance Appearance {
    material Material {
     diffuseColor 0.50 0.50 1.0
    }
  }
  geometry Sphere {
    radius 2
  }
}
```



Obrázek 21: Ukázka osvětlení pomocí uzlu DirectionalLight (viz barevná příloha)

Další obrazové ukázky nebudu uvádět, poněvadž z obrázků není nic vidět. Můžete si prohlédnout rozdíly mezi osvětlením na přiloženém CD v souborech nazvaných stejně, jako popisované uzly.

9.1.2 PointLight (Bodový zdroj světla)

Uzel `PointLight` popisuje bodový zdroj světla v prostoru. Světlo svítí rovnoměrně všemi směry a je ovlivňováno aktuálními transformacemi. Tento světelný zdroj osvětluje celý virtuální svět, tedy nejen sourozenecké stromy.

Parametry `location` a `radius` jsou ovlivňovány případnými transformacemi rodičovského uzlu `Transform`.

Parametr	Iniciální hodnota	Význam
location	0 0 0	Pozice světla v prostoru, v lokálních souřadnicích
radius	100	Definuje do jaké vzdálenosti od světla, jsou ještě osvětleny
attenuation	1 0 0	Trojice nezáporných koeficientů pro výpočet útlumu světla
color	1 1 1	Barva světla
intensity	1	Intenzita světla
ambientIntensity	0	Příspěvek zdroje k nepřímému osvětlení virtuálního světa v rozsahu [0,1]
on	TRUE	Zapnutí či vypnutí světelného zdroje

Tabulka 29: Seznam parametrů uzlu PointLight

Označíme-li trojici koeficientů útlumu (`attenuation`) jako $[a_0,a_1,a_2]$, pak je celkový útlum světelného paprsku ve vzdálenosti *d* od zdroje světla vypočítán podle vzorce $(1/a_0+a_1d+a_2d^2)$. Je-li výsledek větší než jedna, celkový útlum je nastaven na jedna. Aby nedocházelo k dělení nulou, je trojice koeficientů [0 0 0] vždy chápána jako [1 0 0].

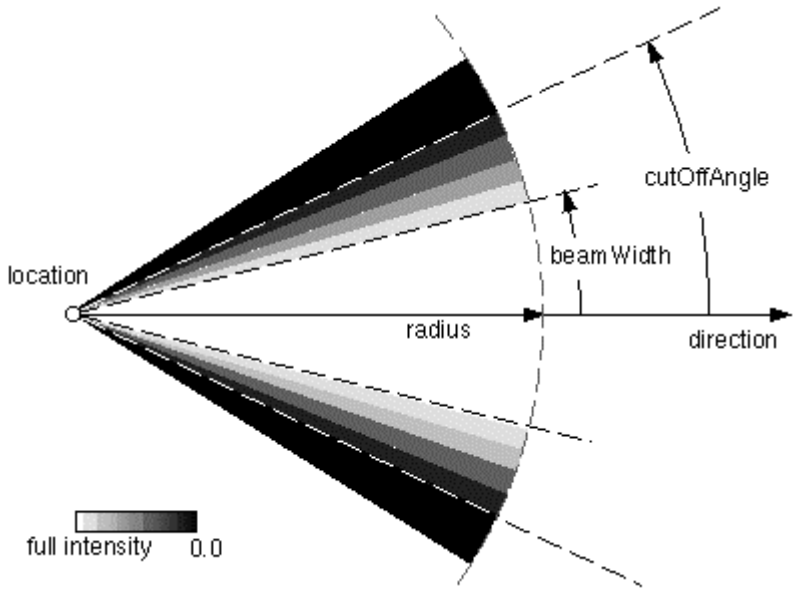
9.1.3 SpotLight (Reflektor)

Uzel `SpotLight` popisuje zdroj směrového světla, které má tvar kuželu (reflektor). V popisu jsou definovány dva souosé kužely. Jeden, jehož vrcholový úhel je roven hodnotě pole `beamWidth` (v radiánech), uvnitř tohoto kuželu svítí světlo plnou intenzitou. Druhý kužel je definován vrcholovým úhlem `cutOffAngle`, je-li tento úhel větší než `beamWidth`, klesá intenzita v prostoru mezi oběma kuželi až na 0. Je-li hodnota `beamWidth` větší než `cutOffAngle`, jsou obě hodnoty nastaveny podle `cutOffAngle` a světlo svítí uvnitř kuželu plnou intenzitou.

Parametr	Iniciální hodnota	Význam
location	0 0 0	Pozice světla v prostoru, v lokálních souřadnicích
direction	0 0 –1	Směr osy světelného kužele
beamWidth	1.570796	Vrcholový úhel kužele, definujícího prosto s plnou intenzitou světla

Parametr	Iniciální hodnota	Význam
cutOffAngle	0.785398	Vrcholový úhel kuželu, definujícího prostor, ve kterém postupně ubývá intenzita světla až na 0
radius	100	Definuje do jaké vzdálenosti od světla, jsou objekty ještě osvětleny
attenuation	1 0 0	Trojice nezáporných koeficientů pro výpočet útlumu světla
color	1 1 1	Barva paprsků
intensity	1	Intenzita světla
ambientIntensity	0	Příspěvek zdroje k nepřímému osvětlení virtuálního svta v rozsahu [0,1]
on	TRUE	Zapnutí či vypnutí světelného zdroje

Tabulka 30: Seznam parametrů uzlu SpotLight



Obrázek 22: Parametry uzlu SpotLight

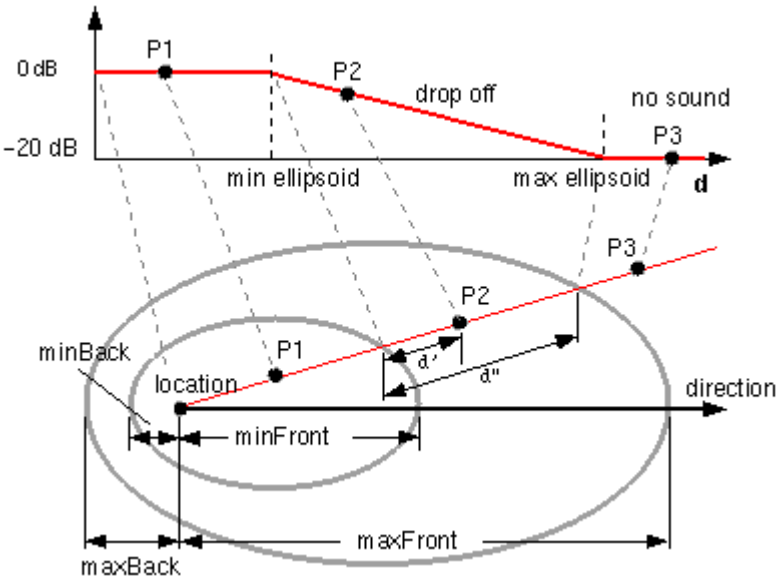
9.2 Zvuk

Do virtuálního prostředí můžeme umístit několik zdrojů zvuku. Umístění, směr a dosah šíření a další charakteristiky jednoho zdroje zvuku se definují v uzlu `Sound`. Ten má jediného potomka – uzel, který určuje soubor se zaznamenaným zvukem a časový průběh jeho přehrávání. Zvlášť jsou tedy popsány geometrické vlastnosti zvukového zdroje a zvlášť vlastní prezentace zvuku.

K popisu dosahu a útlumu zvuku slouží dvojice prostorových elips (elipsoidů), se shodnou hlavní osou a jedním společným ohniskem. V prostoru uvnitř elipsy se zvuk šíří bez jakéhokoliv útlumu. Má stále stejnou hlasitost. Teprve za hranicí vnitřní elipsy začíná slábnout, a to s lineární odstupňovanou intenzitou (v dB). za hranicí vnější elipsy není slyšet vůbec.

Parametr	**Iniciální hodnota**	**Význam**
source	NULL	Uzly `AudioClip` nebo `MovieTexture`, které obsahují URL zvukového souboru určeného pro přehrávání.
location	0 0 0	Umístění zdroje zvuku
direction	0 0 1	Určuje hlavní směr šíření zvuku a směr hlavní osy elipsoidů definujících maximální dosah zvuků
minBack	1	Nezáporná vzdálenost zadního vrcholu elipsy plné slyšitelnosti
minFront	1	Nezáporná vzdálenost předního vrcholu elipsy plné slyšitelnosti
maxBack	10	Nezáporná vzdálenost zadního vrcholu elipsy plného útlumu
maxFront	10	Nezáporná vzdálenost předního vrcholu elipsy plného útlumu
intensity	1	Intenzita zvuku
priority	0	Definice důležitosti zvuku ve scéně; zvuky s prioritou rovnou 1 budou určitě přehrány, naopak priorita rovná 0 je např. pro zvukové pozadí které nemusí být přehráváno
spatialize	TRUE	Povolení generování prostorového zvuku podle orientace avatara

Tabulka 31: Seznam parametrů uzlu Sound



Obrázek 23: Parametry uzlu Sound

Ve skutečném světě většinou slyšíme v každém okamžiku celou řadu zvuků. Některé splývají, jiné snadno rozlišujeme. Uzlů typu `Sound` proto můžeme definovat větší počet. Jestliže se elipsy jejich dosahu překrývají, zní v daném místě několik zvuků naráz.

Parametr source obsahuje uzel, podle jehož parametrů lze vyhledat zvukový soubor. Existují právě dva takové uzly – `MovieTexture` a `AudioClip`. S prvním z nich jsme se již setkali v kapitole MovieTexture (Pohyblivá textura).

9.2.1 AudioClip (Zvukový soubor)

Uzel `AudioClip` obsahuje informace o umístění souboru obsahujícího nahraný zvuk a jak tento zvuk přehrát. Zvuk může být teoreticky v jakémkoliv formátu, ale pro zachování kompatibility, by měl být alespoň jeden *url* se souborem WAVE.

Parametr	**Iniciální hodnota**	**Význam**
url	[]	Seznam adres s umístěním zvukového souboru
description	„“	Informační text
startTime	0	Čas zahájení přehrávání
stopTime	0	Čas ukončení přehrávání; hodnota je ignorována, pokud je menší nebo rovna hodnotě `startTime`

Parametr	Iniciální hodnota	Význam
pitch	1.0	Kladná rychlost přehrávání (tempo)
loop	FALSE	Povolení přehrávání ve smyčce
duration_changed		Původní délka zvuku v sekundách; je vyslána po načtení souboru do paměti
isActive		Bylo zahájeno nebo ukončeno přehrávání

Tabulka 32: Seznam parametrů uzlu AudioClip

9.2.1.1 Duration_changed

Doba po kterou trvá zvuk s hodnotou `pitch` nastavenou na 1.0. Typicky je vyslána pokud se změní doba trvání zvukového klipu, to se může stát pokud je nahrán nový soubor nebo je změněn právě přehrávaný soubor. Změna hodnoty `pitch` nevyvolá tuto událost. Je-li –1 znamená to, že soubor ještě není nahrán.

9.3 Mlha

Tvůrci jazyka VRML zavedli také uzel, který je zaměřený na popis mlhy, kouře, zkrátka takového okolního prostředí, které opticky ovlivňuje celkový obraz pozorovaného virtuálního světa. Jmenuje se `Fog` a má pouhé tři parametry.

9.3.1 Fog (Mlha)

Uzel `Fog` definuje oblast se snižující se viditelností. Prohlížeč míchá barvu mlhy a barvu vykreslovaných objektů, se zvětšující se vzdáleností roste i intenzita mlhy. Objekty, které jsou dále než hodnota pole `visibilityRange` jsou úplně zakryty mlhou.

Parametr	Iniciální hodnota	Význam
color	1 1 1	Barva mlhy ve složkách RGB
fogType	„LINEAR“	Způsob houstnutí mlhy („LINEAR“ nebo „EXPONENTIAL“)
visibilityRange	0	Maximální vzdálenost, ve které pozorovatel uvidí jakýkoliv objekt skrz mlhu. Je-li nastaveno na 0, nebo méně, mlha ve scéně nebude

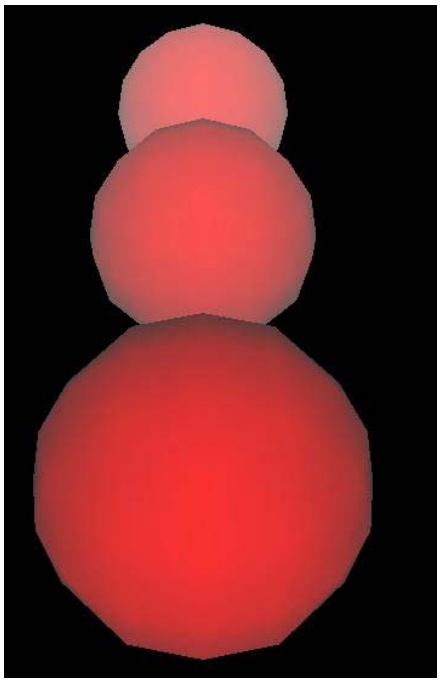
Parametr	Iniciální hodnota	Význam
set_bind		Aktivování mlhy
isBound		Mlha se stala aktivní nebo naopak byla nahrazena jinou mlhou

Tabulka 33: Seznam parametrů uzlu Fog

Uzel `Fog` patří mezi uzly, které jsou do scény připojovány pomocí událostí. Prohlížeč si udržuje zásobník všech `Fog` uzlů, a na vrcholu zásobníku je právě aktuální (připojený k prohlížeči; právě viditelný, mající účinek na scénu). Chceme-li uložit nový uzel `Fog` na vrchol zásobníku a učinit jej aktuálním, je třeba mu zaslat událost `set_bind` s hodnotou *TRUE*. Dosud aktuální uzel je posunut v zásobníku níže.

Příklad Fog.wrl

```
#VRML V2.0 utf8
Viewpoint {
  position 0 3.5 6.5
  orientation 1 0 0 -0.62
}
DEF Koule Shape {
  appearance Appearance {
    material Material {
      diffuseColor 1 0 0}
  }
  geometry Sphere {
    radius 1
  }
}
Transform {
  translation 0 0 3
  children [ USE Koule ]
}
Transform {
  translation 0 0 -3
  children [ USE Koule ]
}
Fog {
   fogType           "LINEAR"
   visibilityRange   20
   color             1 1 1
}
```



Obrázek 24: Ukázka použití mlhy (viz barevná příloha)

9.4 Pozadí

Zatímco mlha svým způsobem kazí vhled virtuálního světa, další uzel se kterým se seznámíme, dokáže obraz světa výrazným způsobem zlepšit a dodat mu skutečnou prostorovou hloubku. Takovým „zázračným“ uzlem je `Background`. Lze jej používat několika způsoby – od jednoduché výplně pozadí jedinou barvou, přes škálu barev měnících se průběžně s výškou nad horizontem až po panoramatické obrázky obklopující ze všech stran virtuální svět.

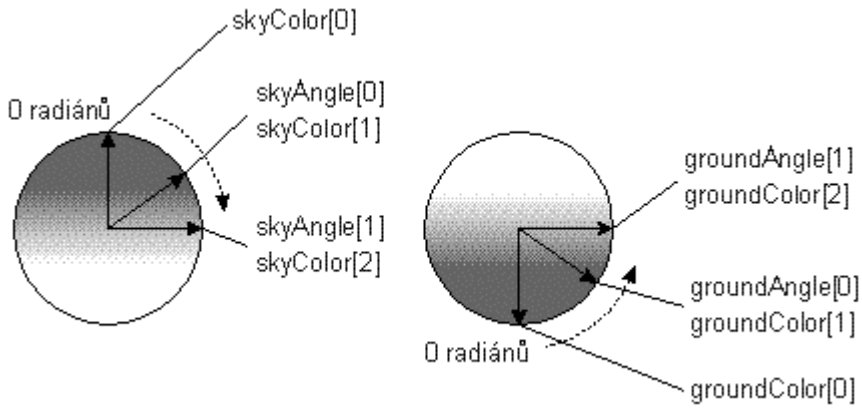
9.4.1 Background (Pozadí)

Uzel `Background` se používá k simulování pozadí scény, včetně barvy oblohy a země. Uzel Background není ovlivňován transformacemi posunu a změny měřítka v hierarchii scény. Rotační transformace otáčejí s tímto uzlem jako s jakýmkoliv jiným geometrickým objektem.

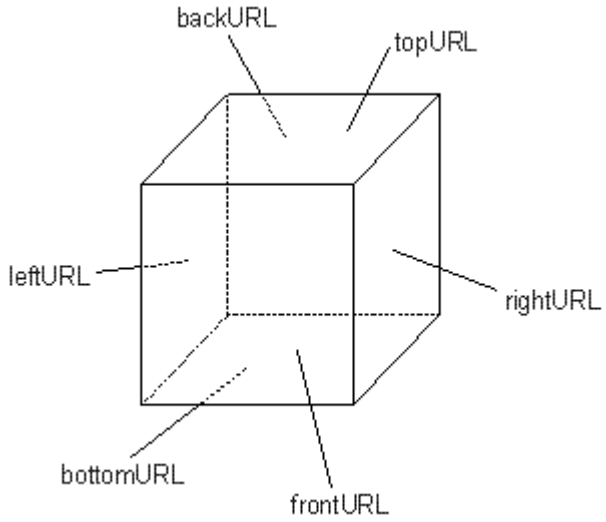
Pozadí může mít dva tvary. Buď je to krychle (viz Obrázek 26) nebo to může být koule s barevnými přechody (viz Obrázek 25)

Parametr	**Iniciální hodnota**	**Význam**
backUrl, bottomUrl, frontUrl, leftUrl, rightUrl, topUrl	[]	Šest obrázků, které jsou mapovány na vnitřní strany panoramické krychle.
skyColor	0 0 0	Barvy, které jsou použity pro „oblohu“
skyAngle	[]	Hraniční úhly (v radiánech), pod kterými jsou postupně vidět barvy na obloze definované v poli `skyColor`
groundColor	[]	Barvy, které jsou použity pro „zem“
groundAngle	[]	Hraniční úhly (v radiánech), pod kterými jsou postupně vidět barvy na zemi definované v poli `groundColor`
set_bind		Aktivování pozadí
isBound		Pozadí se stalo aktivním nebo naopak bylo nahrazeno jiným

Tabulka 34: Seznam parametrů uzlu Background



Obrázek 25: Popis parametrů kulového pozadí



Obrázek 26: Popis krychlového pozadí

Příklad Background.wrl další ukázka Background2.wrl

```
#VRML V2.0 utf8
Background{
  skyColor[0 0 1,
        0.6 0.8 1,
        1 1 1]
  skyAngle[1.5, 1.57]
  groundColor[ 1 1 0.3,
                0.7 0.8 0
           0.3 0.5 0]
  groundAngle[1.4, 1.57]
}
```

9.4.2 Billboard

Uzel `Billboard` je skupinový uzel, který otáčí svou lokální soustavou souřadnic okolo osy rotace (`axisOfRotation`) tak, aby osa z mířila směrem k pozorovateli. Tímto způsobem lze jednoduše vytvořit „prostorovou“ iluzi např. stromu ve scéně, protože půjde o otáčející se 2D obrázek.

Pokud je nastavena osa rotace na (0 0 0) potomek uzlu rotuje okolo počátku soustavy souřadnic a neustále míří směrem k pozorovateli.

Parametr	**Iniciální hodnota**	**Význam**
children	[]	Seznam potomků
axisOfRotation	0 1 0	Osa otáčení
bboxSize	-1 –1 –1	Velikosti hraničního boxu (x y z), který obklopuje potomky uzlu Billboard; (-1 –1 –1) indikuje dosud nespecifi-kovanou velikost kvádru
bboxCenter	0 0 0	Střed hraničního boxu, který obklopuje potomky uzlu `Billboard`
addCildren		Seznam připojovaných uzlů – potomků
removeChildren		Seznam odstraňovaných uzlů – potomků

Tabulka 35: Seznam parametrů uzlu Billboard

Příklad Billboard.wrl

```
#VRML V2.0 utf8
Transform{
  children DEF STROM Billboard{
    children Shape{
      geometry IndexedFaceSet{
     coord Coordinate {point[-1 0 0, 1 0 0, 1 3 0, -1 3 0]}
     coordIndex[0 1 2 3 -1]
     texCoord TextureCoordinate{ point[0 0, 1 0, 1 1,0 1]}
     texCoordIndex[0 1 2 3 -1]
      }
      appearance Appearance {
     texture ImageTexture {url "tree1.png"}
      }
    }
  }
}
Transform { translation 3 0 0.5 children USE STROM}
Transform { translation -3 0 5 children USE STROM}
```

10 Speciální uzly

V předchozích kapitolách jsme se seznámili se všemi základními uzly VRML, které se používají ke stavbě virtuálních světů a tvoří tedy jakési základní kameny. Mezi uzly byly definovány vztahy rodič-potomek, které vytvářely v mnoha případech i několikastupňové stromy. V kořenu těchto stromů stál většinou jediný uzel – `Transform`. Jeho úkolem bylo seskupit potomky a dodat jim potřebné transformace.

10.1 Group (Skupina)

Zcela nejjednodušším skupinovým uzlem je uzel `Group`. Má stejné parametry jako `Transform` kromě těch, které jsou zaměřeny na transformace. Lze říci, že tento uzel nemá jiný úkol než být jakýmsi kontejnerem na uzly. S jeho pomocí uspořádáme uzly v souboru do přehlednějších skupin, každá z nich může být samozřejmě pojmenována konstrukcí DEF

Parametr	**Iniciální hodnota**	**Význam**
chilren	[]	Seznam potomků
bboxSize	-1 –1 –1	Velikost hraničního boxu, který obklopuje potomky uzlu `Group`
bboxCenter	0 0 0	Střed hraničního boxu, který obklopuje potomky uzlu `Group`
addChildren		Seznam připojovaných uzlů
removeChildren		Seznam odstraňovaných uzlů

Tabulka 36: Seznam parametrů uzlu Group

10.1.1 Ohraničující box

Pole `bboxCenter` a `bboxSize` jsou nepovinná a slouží k popisu maximálního ohraničujícího boxu pro potomky v tomto skupinovém uzlu. Prohlížeč používá tento box k optimálnímu rozhodnutí, zda má být kreslena tato skupina, či nikoliv.

Ohraničující box musí být dost rozsáhlý, aby obsahoval všechny potomky uzlu včetně případného dosahu světel a zvuků. Má-li být scéna animována a je šance, že se bude velikost skupiny měnit, ohraničující box musí být dostatečně velký, aby pokryl celou případnou animaci skupiny.

10.2 Anchor (Teleportace, odkaz)

Dalším skupinovým a současně i prvním z interaktivních uzlů je `Anchor`. V prostředí virtuální reality bychom mu spíše mohli říkat teleportace nebo odkaz. Všichni jeho potomci jsou citliví na aktivitu kurzoru. Reakce na aktivaci objektu může být dokonce trojího typu:

1. přechod na nové stanoviště v právě prohlíženém světě
2. nahrazení aktuálního světa jiným s případným přechodem na určité stanoviště v novém světě
3. aktivace hypertextového odkazu (http), nejčastěji s WWW-stránkou.

Parametr	**Iniciální hodnota**	**Význam**
url	[]	URL souboru, který bude nahrán
Parameter	[]	Dodatečná informace pro VRML nebo HTML prohlížeč.
description	“”	Textový popis uzlu Anchor, který je zobrazen v místě url v některých prohlížečích
children	[]	Seznam potomků
bboxSize	-1 –1 –1	Kladné délky stran pomocné obálky ve tvaru kvádru; výjimka [-1 –1 -1] indikuje dosud nespecifikovanou velikost kvádru
bboxCenter	0 0 0	Souřadnice středu pomocné obálky ve tvaru kvádru
addChildren		Přidá daný uzel do seznamu potomků, jestliže se uzel již v seznamu nachází je událost ignorována
removeChildren		Odebere daný uzel ze seznamu potomků

Tabulka 37: Seznam parametrů uzlu Anchor

Jméno cílového stanoviště (uzlu `Viewpoint`) se v parametru `url` zapisuje za adresu oddělenou znakem ‘#’. V cílovém souboru musí být jméno stanoviště zapsáno pomocí konstrukce DEF. V následujících kapitolách budu zobrazovat minimální počet obrázků programů, protože následující příklady jsou nejlépe vidět na příkladech světech.

Příklad Anchor.wrl

```
#VRML V2.0 utf8
Anchor {
  children
    Shape {
      appearance Appearance {
        material Material {
          diffuseColor 1.0 0.5 1.0
        }
      }
      geometry Sphere {
        radius 1.5
      }
    }
  description   "Zobrazí se všechna tělesa."
  parameter      [ ]
  url           "základní telesa.wrl"
}
```

Příklad Anchor2.wrl

```
#VRML V2.0 utf8
DEF JSEM-VLEVO Viewpoint{
  description "Predni"
  position -3 0 5
  orientation 0 1 0 -0.7}
DEF JSEM-VPRAVO Viewpoint{
  description "Bocni"
  position 6 0 5
  orientation 0 1 0 0.7}

Anchor{
  url "#JSEM-VPRAVO"
  children Transform {
    rotation 0 0 1 -1.57
    children DEF KUZEL Shape{
      geometry Cone{}
        appearance Appearance {
        material Material{
          diffuseColor 1 0.75 0.4
        }
        }
      }
    }
  }
Anchor{
  url "#JSEM-VLEVO"
  children Transform{
    translation 3 0 0
    rotation 0 0 1 1.57
    children USE KUZEL
  }
}
```

Příklad Anchor3.wrl

```
#VRML V2.0 utf8
Anchor {
  children
    Shape {
     appearance Appearance {
      material Material {
       diffuseColor 1.0 0.5 1.0
        }
      }
      geometry Sphere {
          radius 1.5
        }
      }
  description   "Presun na
WWW.centrum.cz"
  parameter     "Target= CENT-
RUM - VRML"
  url
"http://www.centrum.cz"
}
```

10.3 LOD (Stupeň detailu)

Uzel `LOD` umožňuje prohlížeči zobrazit různé reprezentace objektů v závislosti na vzdálenosti od nich. Vzdálenost se počítá od aktuálního stanoviště transformovaného do lokálních souřadnic uzlu `LOD` k bodu definovanému v poli `center`. Jestliže je vzdálenost menší než první hodnota v poli `range`, je zobrazena první úroveň definovaná v poli `level`. Je-li vzdálenost mezi první a druhou hodnotou v poli `range` je zobrazena druhá úroveň atd.

V poli `level` by mělo být o jeden uzel více než je hodnota v poli `range`. Jestliže je uzlů méně, poslední z nich je pro nejnižší úrovně detailů opakován. Je-li uzlů více, přebývající jsou ignorovány.

Vzdálenosti a úrovně detailů by měly být voleny tak, aby změna úrovně detailu nebyla postřehnutelná. Prohlížeč sám může zvolit zobrazovanou úroveň detailu aby uspokojil požadavky na určitý počet snímků za sekundu, nebo může zobrazit nižší úroveň detailu, zatímco vyšší je teprve nahrávána.

Parametr	Iniciální hodnota	Význam
level	[]	Uzly, které reprezentují jeden nebo více objektů v různých úrovních detailu, od nejvyšší k nejnižší
center	0 0 0	Bod, vůči kterému se měří vzdálenost objektu od avatara
range	[]	Mezní vzdálenosti pro zobrazování jednotlivých úrovní detailu

Tabulka 38: Seznam parametrů uzlu LOD

Zkušený pozorovatel si všimne, že jednotlivé reprezentace použité v uzlu `LOD` se mění skokově. Výrazné je to u „nejhorší“ reprezentace, která se najednou objeví nebo najednou zmizí. Optické plynulosti docílíme kombinací s globálním uzlem `Fog` – z mlhy se hladce vynoří nejvzdálenější reprezentace.

Příklad LOD.wrl

```
#VRML V2.0 utf8
LOD{
  range [15, 30, 40]
  level[
    Transform {        #model 0 - kuzel
      translation 0 1.5 0
      children Shape {
     appearance DEF MODRA Appearance{
       material Material {diffuseColor 0.2 0.3 1} }
     geometry Cone {
```

```
        bottomRadius 1
        height 3}
       }
      }
      Shape {            #model 1 - ctyrboky jehlan
        appearance USE MODRA
        geometry IndexedFaceSet{
       coord Coordinate {point [-1 0 1, 1 0 1, 1 0 -1,
                          -1 0 -1, 0 3 0]}
       coordIndex[0 1 4 -1, 1 2 4 -1, 2 3 4 -1,
                3 0 4 -1, 0 3 2 1 -1 ]
        }
      }
      Billboard {        #model 2 - trojuhelnik na billboardu
        children Shape{
       geometry IndexedFaceSet{
        coord Coordinate { point[1 0 0, 0 3 0, -1 0 0]}
        coordIndex[0 1 2]
        colorPerVertex FALSE
        color Color {color 0.2 0.3 1}
         }
       }
      }
      Group {}           #model 3 - nic
    ]
}
```

10.4 Inline (Vložení)

K pospojování více virtuálních objektů a světů do většího celku slouží uzle `Inline`. Do jeho parametru s již známým jménem `url` zapíšeme seznam adres, na kterých prohlížeč hledá soubory VRML. Po úspěšném nalezení prvního souboru je jeho obsahu vložen do toho místa ve stromové struktuře aktuálního světa, ve kterém je uzel `Inline` uveden.

Vložený soubor je proto ovlivňován vlastnostmi případných rodičovských uzlů. Je-li některý z jeho rodičů uzlem `Transform`, bude vložený objekt příslušným způsobem transformován, je-li jeho rodičem Billboard, bude dynamicky natáčen podle polohy avatara.

Parametr	Iniciální hodnota	Význam
url	[]	Seznam adres virtuálního světa či objektu
bboxSize	-1 –1 –1	Velikosti hraničního boxu, který obklopuje potomky uzlu `Inline`
bboxCenter	0 0 0	Střed hraničního boxu, který obklopuje potomky uzlu `Inline`

Tabulka 39: Seznam parametrů uzlu Inline

Příklad Inline.wrl

```
#VRML V2.0 utf8
LOD { range [10]
  level [
    LOD { level[
      Inline { url "Pohar.wrl"}
      Inline { url "Fog.wrl"}
    ]
  }
  LOD {level [
    Inline { url "Trojuhelniky.wrl"}
    Group{}
    ]
  }
  ]
}
```

10.5 Switch (Přepínač, volba)

Prvním uzlem, který se stará o několik potomků, aniž by dovolil jejich současné zobrazení je `Switch`. Ve svém parametru `whichChoice` obsahuje celočíselnou hodnotu potomka, který má být zobrazen. Číslo minus jedna znamená, že všichni potomci jsou „skryti“, nezáporná hodnota vybírá potomka v pořadí v jakém je zapsán do souboru. Potomci se zapisují do parametru `choice`, první z potomků má pořadí nula.

Existence více potomků v uzlu `Switch` by byla nesmyslná, kdybychom neměli možnost měnit hodnotu parametru `whichChoice`.

Vzhledem k tomu, že uzel `Switch` je skupinový, musí obsahovat takové potomky, kteří by mohli stát samostatně v souboru VRML, tj. nejčastěji `Group` nebo `Transform`. Nelze proto do uzlu `Switch` zařadit přímo textury, materiály, normály apod.

Parametr	Iniciální hodnota	Význam
choice	[]	Seznam potomků – možností k výběru
whichChoice	-1	Číslo zobrazovaného potomka; číslo –1 znamená žádný potomek

Tabulka 40: Seznam parametrů uzlu Switch

11 Definice vlastních uzlů

Vývojáři jazyka VRML 2.0 mysleli také na vytvoření vlastních uzlů a popisu jejich vlastností. Zavedli zde konstrukci `PROTO` – prototyp. Pomocí této konstrukce zkrátka dokážeme popsat to, co nám v jazyce VRML chybí. Prototyp představuje vzor uzlu, který můžeme vkládat do stromové struktury VRML a modifikovat nastavením jeho parametrů.

Příklad DatTyp.wrl

```
#VRML V2.0 utf8
EXTERNPROTO Stul [
  field SFColor       barva
  field SFVec3f       posunuti
  field SFRotation    natoceni
  field SFVec3f       meritko]
  "Stolecek.wrl#MujStolek"
Group {
  children [
    Stul {}
    Stul { posunuti -2 0 0}
    Stul { barva 0 1 0.2
           posunuti 2 0 0}
    Stul { barva 1 0.3 0
           meritko 0.5 0.5 0.5}
    Stul { posunuti -2.5 0 2.5
           meritko 1.8 0.6 1
           barva 1 1 0}
    Stul { posunuti 3 0 2
           natoceni 0 1 0 0.6
           meritko 0.8 0.8 0.8
           barva 0 0 1}
  ]
}
```

Program je tvořen dvěma částmi. V první jsou popsány parametry stolku, ve druhé je tento stolek opakovaně vkládán do virtuálního světa jako běžný uzel se jménem `Stul`.

Dříve, než uvedeme obsah souboru Stolecek.wrl, všimneme si první části ukázky. Hned za hlavičkou souboru jsou zapsány údaje, které blíže specifikují parametry stolku Nejprve je uvedeno klíčové slovo EXTERNPROTO, které oznamuje, že následující prototyp bude definován v samostatném (externím) souboru a že v aktuálním souboru nenalezneme nic víc, než parametry tohoto prototypu.

V hranatých závorkách je pak uveden seznam parametrů. V ukázce je každý parametr charakterizován třemi slovy:

```
field       <typ dat>          <jméno parametru>
```

Slovo `field` patří přímo do jazyka VRML a slouží k označení takového parametru, který má předdefinovanou hodnotu.

11.1 Datové typy

Každý parametr má jednoznačně určeno, jaké hodnoty do něj lze ukládat. Na výběr je několik datových typů, přičemž většina z nich se může vyskytovat ve dvou variantách. Buď v podobě jediné samotné hodnoty (předpona **SF** – Single Field), nebo jako seznam hodnot (předpona **MF** – Multiple Field), jehož délka obecně není omezena. Pod pojmem jediná hodnota rozumíme současně i hodnotu jednoho strukturovaného parametru, například barvy, která je ve skutečnosti dána třemi složkami R, G a B.

Datový typ		**Význam**	**Defaultní hodnota**
SFBoll	---	logická hodnota	FALSE
SFColor	MFColor	barva ve složkách RGB každá v intervalu <0,1>	0 0 0; [ ]
SFFLoat	MFFloat	číslo s desetinnou tečkou	0.0; [ ]
SFimage	---	strukturovaný vzor pixelů	(0 0 0)
SFInt32	MFInt32	celé číslo v rozsahu 32 bitů	0; []
SFNode	MFNode	uzel VRML	NULL; []
SFRotation	MFRotation	osa (vektor v prostoru) a úhel rotace (v radiánech)	0 0 1 0; []

Datový typ		Význam	Defaultní hodnota
SFString	MFString	textový řetězec	„“
SFTime	MFTime	čas v sekundách	[]
SFVec2f	MFVec2f	vektor v rovině	(0 0); []
SFVec3f	MFVec3f	vektor v prostoru	(0 0 0); []

Tabulka 41: Typy dat

Nyní se již můžeme vrátit zpět k programu a pokračovat v rozboru jeho první části. Všechny čtyři parametry prototypu Stul jsou datovými typy obsahujícími jednu hodnotu. Na konci seznamu parametrů je uvedena adresa souboru, ve které se nachází definice prototypu. V našem případě je to soubor Stolecek.wrl.

Příklad Stolecek.wrl

```
#VRML V2.0 utf8
PROTO MujStolek [
  field SFColor barva .6 .5 .1
  field SFVec3f posunuti 0 0 0
  field SFRotation natoceni 0 1 0 0
  field SFVec3f meritko 1 1 1 ]
{
  Transform {
    translation IS posunuti
    rotation IS natoceni
    scale IS meritko
    children [
      Transform {
            translation 0 1.1 0
       children Shape {
              appearance DEF BARVA Appearance {
                material Material {diffuseColor IS barva}
              }
              geometry Box {size 1.2 0.2 1.2}
       }
  }
    Transform {
     translation -.5 0.5 -.5
     children DEF NOHA Shape {
       appearance USE BARVA
       geometry Cylinder { height 1 radius .1 top FALSE}
     }
    }
      Transform {
       translation .5 .5 -.5
```

```
            children USE NOHA}
           Transform {
            translation -.5 .5 .5
            children USE NOHA}
           Transform {
            translation 0.5 0.5 0.5
            children USE NOHA}
         ]
        }
      }
```

Prototyp stolku je uvozen slovem `PROTO`, za nímž následuje jméno prototypu. Všimněme si, že právě toto jméno je vyhledáváno v adresách zapsaných v příkazu `EXTERNPROTO`. Za jménem následuje seznam parametrů nově definovaného uzlu, tentokrát včetně iniciálních hodnot. Nakonec je ve složených závorkách vytvořen prototyp z uzlů VRML, ať již standardních nebo nově zavedených v nějaké dřívější konstrukci `PROTO`. Tělesa mají přiřazenu slečnou barvu a všechna jsou potomky hlavního rodičovského uzlu `Transform`.

Jako poslední musíme zajistit předání parametrů nově deklarovaného uzlu do parametrů již existujících uzlů, jinými slovy je třeba zavést přiřazovací příkaz. Ten je realizován slovem IS. Tímto způsobem jsou nové transformační parametry `posunuti`, `natoceni` a `meritko` zapsány do skutečných parametrů `translation`, `rotation` a `scale`. Podobně je parametr barva předán parametru `diffuseColor` uzlu `Material`.

12 Dynamika VRML

Virtuální světy, které jsme se naučili navrhovat v předchozích kapitolách, mají statický charakter. Jedinou dosud uvedenou interakcí je aktivace potomků uzlu `Anchor`, po níž následuje přesun na nové stanoviště nebo teleportace do dalšího světa. Abychom dokázali změnit statické světy na dynamické, musíme obohatit škálu našich prostředků o dva další prvky:

1. dynamické uzly, které reagují na chování uživatele, resp. avatara, a vysílají informace v podobě v tzv. událostí (anglicky *event*),
2. mechanismus předávání událostí mezi uzly – konstrukce `ROUTE`.

12.1 Událost

Událost je základním prostředkem, který umožňuje „rozhýbání“ statických světů. Můžeme si ji představit jako datový záznam, který je předáván mezi uzly v okamžiku, kdy z nějakého důvodu dojde ke změně hodnoty parametru uzlu. O uzlu, v němž událost na základě takového podnětu vznikne, říkáme, že *vyslal událost*.

Podobně uzel, k němuž byla data o události dopravena k dalšímu zpracování, událost *přijal* a přijatá data uložil do svého parametru.

12.2 ROUTE … TO …

Tato jazyková konstrukce vytváří spojení mezi uzlem, který generuje událost a uzlem, který ji přijímá. Uzly, na které se odkazuje, by měli být již nadefinovány. Typy vstupní a výstupní události si musí odpovídat. Například je nepřípustné nadefinovat spojení z SFFloat do SFInt32 nebo z SFFloat do MFFloat.

Syntaxe má následující podobu:

```
ROUTE <NázevUzlu>.<NázevPole> TO <NázevUzlu>.<NázevPole>
```

Místo Názvu pole může být také Název události, které se to týká.

```
Group {
  children [
    ……# popis mistnosti
    DEF SVETLO SPOTLIGHT {on FALSE} # Zpočátku světlo nesvítí
    Sound {source DEF HUDBA AudioClip {…}
    }
    DEF AKCE ProximitySensor{…}
  ]
}
ROUTE AKCE.isActive    TO SVETLO.on
ROUTE AKCE.enterTime TO HUDBA.startTime
ROUTE AKCE.exitTime    TO HUDBA.stopTime
```

Uzel `ProximitySensor` (Detektor přítomnosti viz ProximitySensor (Detektor přítomnosti)), nazvaný `AKCE`, vysílá v příkladu dvě události při vstupu avatara do určité oblasti a dvě při avatarově odchodu. Z příkladu je vidět, že události obsahují data různých typů. Pro zapnutí a vypnutí světla je třeba přenášet údaje `TRUE` a `FALSE`, tedy typu `SFBool`. K přehrávání zvuku je nutno dodávat časové údaje, tj. data typu `SFTime`. Uzel, který přijímá událost, proto smí zapisovat dodávané údaje pouze do těch parametrů, jejichž datový typ je shodný s datovým typem konkrétní události.

12.3 Vztah parametrů k událostem

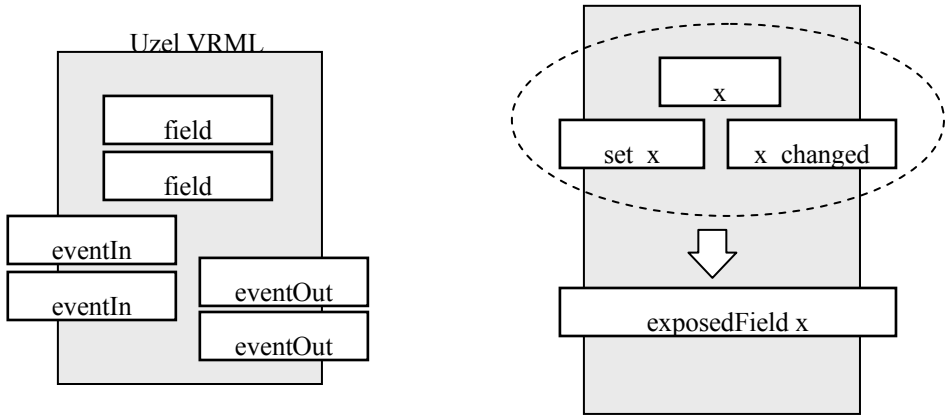
Až doposud jsme chápali parametry uzlu jako datový typ s iniciální hodnotou. Alespoň pro práci se statickými světy nám tyto údaje postačovaly. V tomto okamžiku, kdy se zabýváme i dynamikou, nám ale nestačí pouze toto určení, ale potřebujeme u parametrů znám jejich vztah k událostem. Existují tři základní třídy parametrů:

`field` Statická veličina s iniciální hodnotou. Lze ji změnit pouze zapsáním nové hodnoty v souboru VRML, nikoliv však dynamicky

`eventIn` Parametr schopný přijmout událost daného datového typu. Hodnota parametru je změněna přijetím události.

`eventOut` Parametr schopný vyslat událost v okamžiku, kdy dojde ke změně jeho hodnoty. Tato změna je nejčasněji vyvolána chováním avatara nebo je důsledkem zpracování jiných událostí.

Zobrazíme-li symbolicky jeden uzle VRML jako obdélník, statické parametry třídy `field` jsou umístěny zcela uvnitř, nedostupné pro dynamické akce. Parametry `eventIn` pro příjem událostí jsou posunuty doleva, parametry `eventOut` pro vyslání události doprava. U mnoha uzlů je žádoucí, aby nějaký parametr měl nejen svoji iniciální hodnotu, ale aby současně dokázal přijmout událost, která jeho hodnotu změní a aby tuto změnu vyslal v podobě události dalším uzlům. Tato třída parametrů se nazývá `exposedField`. Každý parametr této třídy v sobě zahrnuje schopnosti všech tří základních tříd.



Obrázek 27: Třídy parametrů uzlů VRML

`exposedField` Parametr s iniciální hodnotou, který je schopný přijmout události měnící jeho hodnotu a také po změně své hodnoty událost vysílat.

Zajímavé je, jakým způsobem se zachází se jmény parametrů. Pokud se parametr třídy `exposedField` jmenuje například `poloha`, pak ve skutečnosti představuje tři parametry základních tříd, z nichž každý má svoje pojmenování a lze s ním i samostatně pracovat:

1. `field` `poloha` <iniciální hodnota>
2. `eventIn` `set_poloha`
3. `eventOut` `poloha_changed`

Předpona „`set_`“, resp. přípona „`_changed`“, je často používána i ve jménech parametrů, které události pouze přijímají, resp. vysílají. Používání této dohody usnadní návrháři práci, protože podle jména lze snadno odhadnout charakter parametru.

V předchozích kapitolách jsme parametry uzlů nezatřiďovali do tříd. Ve většině případů šlo o parametry třídy `field` a `exposedField`. Detailní popis uzlů naleznete v Příloha – Detailní popis VRML uzlů. Parametry třídy `eventIn` a `eventOut` jsou vlastní především dynamickým uzlům, a proto se s nimi v dosud uvedených uzlech setkáme jen výjimečně. Následující přehled uvádí ty statické uzly, který zpracovávají události speciálními parametry určenými pouze k vysílání či pouze k přijímání událostí:

Parametr třídy `eventInt`	**Činnost**	**Výskyt v uzlech**
`SFBool set_bind`	Aktivování uzlů, z nichž pouze jeden z každého druhu může být aktivní	`Background, Fog, NavigationInfo, Viewpoint`
`MFNode addChildren, MFNode removeChildren`	Přidání a odebrání seznamu potomků skupinovým uzlům	`Anchor, Billboard, Group, Transform`

Parametr třídy `eventOut`		
`SFBool isBound`	Informace o tom, že daný byl aktivován nebo deaktivován	`Background, Fog, NavigationInfo, Viewpoint`
`SFBool isActive`	Informace o tom, že nahrávka je přehrávána nebo ukončena	`AudioClip, MovieTexture`
`SFTime duration_changed`	Informace o délce nahrávky po jejím načtení ze souboru	

Tabulka 42: Přehled parametrů statických uzlů, používaných výhradně pro vysílání a příjem událostí

Příkaz `ROUTE` zasílá události mezi pojmenovanými uzly. Velmi důležitá je skutečnost, že pokud přiřadíme příkazem `DEF` jméno nějakému uzlu a následujícími příkazy `USE` vytvoříme nové uzly, jediný příkaz `ROUTE` zajistí zaslání události všem těmto stejně pojmenovaným uzlům.

12.4 Schéma dynamické akce

Události jsou tím, co rozhýbe statický svět, dodá mu dynamiku, animace, schopnost reakce na avatarovo jednání. Pro lepší orientaci s zacházením s událostmi, je zde následující obrázek. Na něm je schematicky naznačen průběh libovolné dynamické akce. To však neznamená, že každá událost musí obsahovat všechny uvedené prvky nebo přesně dodržovat jejich pořadí. mnoho dynamických akcí je tvořeno jen třemi či čtyřmi prvky.

Na obrázku je pět uzlů, které si postupně předávají události.

Obrázek 28: Logické schéma obecné dynamické akce

12.4.1 Čidlo

Na počátku každé aktivity musí stát prvek, který detekuje příčinu dynamické akce, ať je to vstup avatara do místnosti, aktivace kurzoru nad obrazem virtuálního objektu nebo jiná událost. Takovému prvku můžeme říkat Čidlo. Ve VRML máme k dispozici celou řadu rozličných čidel.

12.4.2 Logika

Za čidlem často následuje prvek, který rozhoduje o tom, zda byly splněny veškeré podmínky pro skutečnou zahájení dynamické akce. Příkladem je virtuální videopřehrávač, který nezahájí činnost dříve, než jej avatar zapne do elektrické sítě. Jiným příkladem je otevření dveří trezoru až po nastavení správné číselné kombinace. Prvek, zvaný Logika, obsahuje většinou lokální paměť realizovanou uzlem třídy `field`. Ke složitějšímu vyhodnocování podmínek slouží uzel `Script`, který bude důkladně popsán v kapitole 13 Script.

12.4.3 Časovač

Časovač je zodpovědný za časový průběh akce. Nemusí se starat pouze o správný as zahájení a ukončení, ale může vhodným způsobem měnit dynamiku děje. Otevírané dveře e například mohou nejprve prudce „rozletět“ a poté krátce dobrzdit v mstě úplného otevření. Špatně namířená kulička na kulečníku naopak bude svoji iniciální vysokou rychlost snižovat postupně až do úplného zastavení.

12.4.4 Pohon

Viditelná dynamická akce je vždy realizována postupnou změnou hodnot některého parametru, ať již jde o polohu, rozměr či natočení objektu, změnu jeho barvy, rozsvícení světla apod. Prvek, nazvaný Pohon, dokáže na základě předem definovaných počátečních a koncových hodnot průběžně vypočítat nové hodnoty, a to v souladu s dynamikou dodávanou časovačem.

12.4.5 Cíl

Na konci řetězce událostí je Cíl, na němž je dynamická akce viditelná. Bývá to běžný statický uzel, do jehož parametrů (nejčastěji třídy `exposedField`) se zasílají hodnoty způsobující změnu jeho vzhledu. Jednoduše lze říci, že všechny čtyři předchozí prvky sloužily k tomu, aby se s cílovým prvkem „něco stalo“, a to v souladu s představami tvůrce dynamického virtuálního světa.

12.5 Časová souvislost

Na závěr se musíme zmínit o časové souvislosti. Uvedli jsme, že každá událost s sebou nese údaj o čase, ve kterém vznikla. Tomuto údaji se říká *časové razítko*. Čas je v něm zapsán absolutně, a to počtem sekund, který uplynuly od půlnoci 1. ledna 1970. Čas přitom může obsahovat i necelá čísla a popisovat tak děje v rozsahu milisekund apod. Přestože je čas udáván absolutně, ve rituálním světech se pracuje s časem relativním, tedy vztaženým k nějaké události, typicky k aktivitě avatara.

Časová razítka mají velký Význam při větvení událostí, tj. v takových situacích, kdy jedna událost vyvolá celý řetěz následných událostí. Aby byl co nejvíce potlačen vliv různých časových prodlev způsobených zatížením počítače, časové razítko původní události je beze změny okopírováno do všech bezprostředně následujících událostí.

U parametrů třídy `exposedField` není třeba v příkazu ROUTE uvádět jména s předponou `set_` a příponou `_changed`, protože tyto přípony jsou k nim doplněny automaticky podle toho, zda parametr událost vysílá či přijímá. Jejich plné uvedení však zvyšuje čitelnost souboru.

12.6 Manipulátory

Manipulátory umožňují návštěvníkovi virtuálního světa změnit polohu vybraného objektu nebo skupiny objektů. Lze říci, že avatar s jejich pomocí dokáže svoji virtuální rukou uchopit objekt a změnit jeho vlastnosti popsatelné transformacemi v prostoru.

V okamžiku, kdy je na obrazovce přesunut kurzor nad objekt, který je pod vlivem manipulátoru, lze aktivováním kurzoru (stisknutím tlačítka myši) s objektem manipulovat – měnit jeho polohu, orientaci či měřítko.

Existují celkem tři manipulátory:

1. `CylinderSensor`
2. `PlaneSensor`
3. `SphereSensor`

Manipulátory se navzájem liší nejen ve způsobu převodu pohybu aktivovaného kurzoru na výpočet nových polohových vlastností objektů, ale i tím, jaká výsledná data poskytují.

Všechny manipulátory mají dvě stejné výstupní události, `trackPoint_changed` a `<value>_changed`. Tyto události jsou generovány v každém okamžiku po dobu, kdy je myš aktivována stisknutím tlačítka na odpovídajícím geometrickém objektu a stále se stisknutým tlačítkem libovolně posouvána. Odpovídajícím geometrickým objektem se rozumí takový, který je potomkem rodičovského uzlu příslušného manipulátoru.

Mechanismus vybrání geometrického objektu pomocí myši je následující: Myš slouží k řízení ukazatele ve virtuálním prostoru. Tento ukazatel je mapován na rovinu s konstantní vzdáleností od avatara a kolmou na linii pohledu. Pozice ukazatele definuje vztahový vektor, který se používá k určení vybraného geometrického objektu. Vztahový vektor je definován spojnicí mezi pozicemi avatara a ukazatele. Geometrický objekt je vybrán pokud jím prochází přímka určená vztahovým vektorem. Po-

kud tato přímka protíná více geometrických objektů, které jsou potomky rodičovského uzlu manipulátoru, tak je vybrán objekt, jenž je nejbližší k virtuálnímu ukazateli.

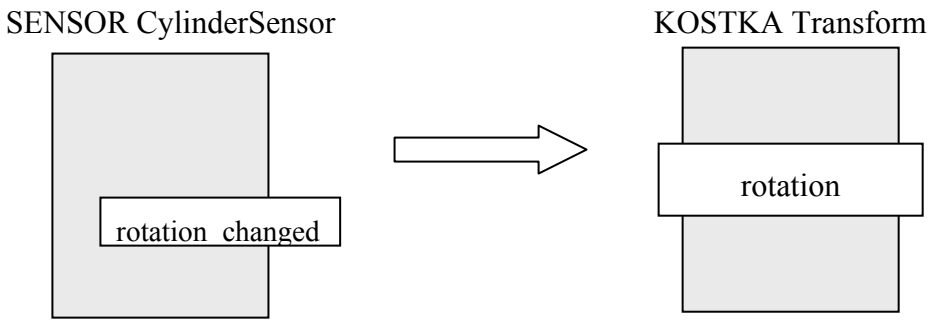
12.6.1 CylinderSensor (Válcový manipulátor)

`CylinderSensor` převádí pohyb myši na rotaci okolo neviditelného válce, který je umístěn v Y-ové ose souřadného systému uzlu. Válcový manipulátor je nejsložitějším manipulátorem, protože musí zajišťovat pohyb avatarovy ruky jak po pomyslném rotačním plášti válce, tak po jeho podstavách.

Typ parametru	**Parametr**	**Iniciální hodnota**	**Význam**
exposedField	enabled	TRUE	Povolení práce manipulátoru
	offset	0	Obsahuje úhel, který se přičítá k velikosti rotace neviditelného válce (disku).
	autoOffset	TRUE	Povolení automatické aktualizace `offset` po skončení činnosti manipulátoru
	minAngle	0	Dolní mez vyhodnoceného úhlu otočení; je z intervalu [-2π, 2π]
	maxAngle	-1	Horní mez vyhodnoceného úhlu otočení; je z intervalu [-2π, 2π]
	diskAngle	0,262	Kladný úhel menší než π/2, rozlišující, zda z pomyslného válce se použije podstava nebo plášť.
eventOut	isActive		Zahájení a ukončení činnosti manipulátoru.
	trackPoint_changed		Měnící se poloha bodu na pomyslném válci, na který ukazuje kurzor při práci s manipulátorem.
	rotation_changed		Měnící se hodnota úhlu odvozená z pohybu vstupního zařízení kolem osy válce.

Tabulka 43: Seznam parametrů uzlu CylinderSensoru

Je-li nastaven parametr `autoOffset` na `TRUE`, je po skončení činnosti manipulátoru aktualizována hodnota parametru `offset` a vyslána událost `offset_changed`.



Obrázek 29: Vyslání události manipulátorem

Parametr `diskAngle` určuje, zda pohyb kurzoru bude manipulátorem chápán jako pohyb po plášti pomyslného válce či jako pohyb po kruhové podstavě pomyslného válce. Vektor určený aktuální polohou avatara a bodem na povrchu pomyslného válce svírá s osou pomyslného válce úhel, jehož hodnota je porovnávána s velikostí parametru `diskAngle`. Je-li svíraný úhel menší, pracuje se s podstavou válce a ze strany uživatele je očekáván kruhový pohyb kurzoru. V opačném případě se pracuje s pláštěm a uživatel může generovat otáčení pouze translačním pohybem kurzoru.

Příklad CylinderSensor.wrl

```
#VRML V2.0 utf8
DEF POHYB Transform {
  children [
    DEF KOSTKA Transform {
      children [
        Shape {
          appearance Appearance {
            material Material { diffuseColor 1 0 0 }
          }
          geometry Box {}
        }
      ]
    }
    DEF SENZOR CylinderSensor {}
  ]
}
ROUTE SENZOR.rotation_changed TO KOSTKA.rotation
```

12.6.2 PlaneSensor (Rovinný manipulátor)

`PlaneSensor` převádí pohyb myši na posun v rovině XY svého lokálního souřadného systému. Tento manipulátor má navíc další dva parametry, které omezují pohyb na jinak nekonečně velké pomyslné manipulační desce umístěné v rovině xy (z = 0). Vzhledem k tomu, že jde o pohyb v rovině, parametry obsahují souřadnice pouze ve dvourozměrném prostoru.

Typ parametru	Parametr	Iniciální hodnota	Význam
exposedField	enabled	TRUE	Povolení práce manipulátoru
	offset	0 0 0	Základní posunutí, který je vždy přičítáno k nově vyhodnocenému posunutí
	autoOffset	TRUE	Povolení automatické aktualizace `offset` po skončení činnosti manipulátoru
	minPosition	0 0	Dolní mez vyhodnoceného posunutí
	maxPosition	-1 –1	Horní mez vyhodnoceného posunutí
eventOut	isActive		Zahájení a ukončení činnosti manipulátoru
	trackPoint_changed		Měnící se poloha bodu na pomyslné desce, na který ukazuje kurzor při práce s manipulátorem
	translation_changed		Měnící se hodnota posunutí odvozená z pohybu vstupního zařízení

Tabulka 44: Seznam parametrů uzlu PlaneSensor

Manipulátor sám o sobě nemění transformace žádného uzlu, pouze pro ně vypočítává vhodné parametry.

Je-li nastaven parametr `autoOffset` na `TRUE`, je po skončení činnosti manipulátoru aktualizována hodnota parametru offset a vyslána událost offset_changed.

Je-li libovolná ze souřadnic `maxPosition` menší než odpovídající souřadnice `minPosition`, vypočítávaná posunutí nejsou omezena. V opačném případě je pohyb kurzoru, který by generoval posunutí mimo meze, v odpovídajícím směru ignorován. V případě rovnosti jedné ze souřadnic `maxPosition` a `minPosition` převádí manipulátor pohyb kurzoru na pohyb po úsečce.

Příklad PlaneSensor.wrl

```
#VRML V2.0 utf8
DEF RODIC Transform {
  rotation 1 0 0 1.62
  translation 0 1.1 0
  children [
    DEF SENZOR  PlaneSensor {
      maxPosition  3  3
      minPosition -3 -3 }
    DEF OBJEKT Transform {
      rotation 1 0 0 -1.62
      children [ Shape {
```

```
          appearance Appearance {
            material Material { diffuseColor 1 0 0 }
        }
         geometry Box {}
         }]
       }
    ]
  }
  Shape {
    appearance Appearance {
      material Material { diffuseColor 0 1 0 }
    }
    geometry Box { size 8 0.2 8}
  }
  ROUTE SENZOR.translation_changed TO OBJEKT.translation
```

12.6.3 SphereSensor (Kulový manipulátor)

`SphereSensor` převádí pohyb myší na rotaci okolo středu svého souřadného systému. Velikost pomyslné koule, po které se pohybuje avatarova ruka, je automaticky nastavena tak, aby koule obklopila geometrii sourozeneckých uzlů, tedy objektů citlivých na avatarovu ruku.

Typ parametru	**Parametr**	**Iniciální hodnota**	**Význam**
exposedField	enabled	TRUE	Povolení práce manipulátoru
	offset	0 1 0 0	Základní osa a úhel otočení, se kterými jsou vždy složeny nově vyhodnocené hodnoty
	autoOffset	TRUE	Povolení automatické aktualizace `offset` po skončení činnosti manipulátoru
eventOut	isActive		Zahájení a ukončení činnosti manipulátoru
	trackPoint_changed		Mnící se poloha bodu na pomyslné kouli, na který ukazuje kurzor při práci s manipulátorem
	rotation_changed		Měnící se osa s otočení odvozené z pohybu vstupního zařízení

Tabulka 45: Seznam parametrů uzlu SphereSensor

Je-li nastaven parametr `autoOffset` na `TRUE`, je po skončení činnosti manipulátoru aktualizována hodnota parametru `offset` a vyslána událost `offset_changed`.

Příklad SphereSensor.wrl

```
#VRML V2.0 utf8
DEF RODIC Transform {
  children[
    DEF SENZOR SphereSensor { }
    DEF OBJEKT Transform {
      children [ Shape {
     appearance Appearance {
          material Material { diffuseColor 1 0 0 }
     }
        geometry Box {}
      }]
    }
  ]
}
ROUTE SENZOR.rotation_changed TO OBJEKT.rotation
```

12.7 Senzory

Mezi senzory, kromě manipulátorů, patří tyto typy uzlů:

1. `Collision`
2. `ProximitySensor`
3. `TimeSensor`
4. `TouchSensor`
5. `VisibilitySensor`

12.7.1 Collision (Detekce nárazu)

Standardně může u všech objektů ve scéně dojít ke kolizi s jiným objektem scény. Prohlížeč by měl detekovat kolize mezi pozorovatelem a geometrickými objekty scény a zabraňovat pozorovateli „vstoupit“ do nich. Uzel `Collision` je skupinový uzel, který umožňuje vypnout detekci kolizí pro své potomky, určit alternativní objekty, u kterých se bude detekovat kolize, a posílat události signalizující, že došlo ke kolizi mezi pozorovatelem a některým z geometrických objektů, jež je potomkem tohoto uzlu. Pokud není ve scéně použit žádný uzel `Collision`, prohlížeč by měl detekovat kolize s každým objektem scény.

Typ parametru	Parametr	Iniciální hodnota	Význam
exposedField	children	[]	Seznam potomků
	collide	TRUE	Povolení detekování kolizí s avatarem
field	bboxSize	-1 –1 –1	Rozměry ohraničujícího boxu. Hodnoty –1 –1 –1 indikuje dosud nespecifikovanou velikost kvádru
	bboxCenter	0 0 0	Střed ohraničujícího boxu
	proxy	NULL	Uzel nebo strom s náhradní geometrií, která se použije pro detekce kolize
eventIn	addChildren		Seznam připojovaných uzlů
	removeChildren		Seznam odstraňovaných uzlů
eventOut	collideTime		Čas nárazu avatara

Tabulka 46: Seznam parametrů uzlu Collision

Je-li definován náhradní objekt pro výpočet kolizí (proxy), žádný z potomků (children) není na kolizi otestován. Náhradní objekt není pochopitelně nikdy zobrazován. Obálka ve tvaru kvádru (bbox) neslouží jako náhradní reprezentace, pouze jako doplňující informace.

Příklad Collision.wrl

```
#VRML V2.0 utf8
DEF COLIZE Collision {
  children [
    Shape {
      appearance Appearance {
        material DEF MUJ Material { diffuseColor 0 1 0 }
      }
      geometry Box {}
    }
  ]
}
```

12.7.2 ProximitySensor (Detektor přítomnosti)

`ProximitySensor` generuje události v okamžiku, když avatar vstoupí, opustí nebo se pohybuje v prostoru, který je definován imaginárním kvádrem.

Typ parametru	Parametr	Iniciální hodnota	Význam
exposedField	center	0 0 0	Těžiště kvádru vymezujícího sledovanou oblast
	size	0 0 0	Nezáporné rozměry kvádru
	enabled	TRUE	Povolení práce detektoru
eventOut	isActive		Vstup a výstup avatara do, resp. ze sledované oblasti
	enterTime		Čas vstupu avatara do sledované oblasti
	exitTime		Čas opuštění sledované oblasti avatarem
	position_changed		Poloha avatara uvnitř oblasti se změnila
	orientation_changed		Orientace avatara uvnitř oblasti se změnila.

Tabulka 47: Seznam parametrů uzlu ProximitySensor

Rozměry sledované oblasti jsou transformovány podle parametrů případného rodičovského uzlu `Transform`.

`ProximitySensor`, jehož pole `size` obsahuje hodnotu (0 0 0), nemůže generovat žádné události – tento stav je ekvivalentní s nastavením pole `enabled` na `FALSE`.

Existuje-li několik různých uzlů `ProximitySensor`, jejichž sledované oblasti se překrývají, mohou být přítomnost a pohyb avatara detekovány současně několika uzly.

Je-li jeden uzle `ProximitySensor` násobně vložen na různá místa pomocí konstrukcí `DEF` a `USE`, vznikne násobná oblast jako sjednocení jednotlivých kvádrů. Kvádry se však v tomto případě nesmějí překrývat.

12.7.3 TimeSensor (Časovač)

Významnou roli v každé interaktivní akci hraje čas. Časování akcí zajišťuje uzle `TimeSensor`, který hraje jednoznačnou roli časovače.

Na časovač můžeme pohlížet jako na zařízení, které je schopno průběžně vysílat události – časové impulsy. Frekvence vysílání těchto impulsů nemůžeme ovlivnit, protože závisí na výkonu počítače.

Typ parametru	**Parametr**	**Iniciální hodnota**	**Význam**
exposedField	enabled	TRUE	Povolení práce časovače
	startTime	0	Čas zahájení činnosti časovače
	stopTime	0	Čas ukončení činnosti časovače
	cycleInterval	1	Nezáporná délka časové smyčky
	loop	FALSE	Povolení generování časových událostí v nekonečné smyčce
eventOut	isActive		Zahájení a ukončení práce časovače
	time		Plynule generovaný absolutní čas
	cycleTime		Čas zahájení další časové smyčky
	fraction_changed		Plynule generovaná poměrná hodnota uplynulého času v rámci jedné smyčky

Tabulka 48: Seznam parametrů uzlu TimeSensor

Je-li povolena časová smyčka, nabývá parametr `fraction_changed` nulové hodnoty pouze jedinkrát, na úplném počátku. Veškeré další průchody začátkem nových časových smyček jsou ohodnoceny číslem 1, tj. jako ukončení průchodu předchozí časovou smyčkou.

Pokud je hodnota `stopTime` menší než hodnota `startTime`, časovač může pracovat neustále v časové smyčce. V opačném případě ukončí svoji práci po dosažení času `stopTime`, když předtím proběhlo libovolné množství časových smyček.

Příklad TimeSensor.wrl

```
#VRML V2.0 utf8
DEF RODIC Transform {
  children [
    Shape {
      appearance Appearance {
        material Material { diffuseColor 1 0 0 }
      }
```

```
      geometry Box {}
    }
    DEF DOTEK TouchSensor {}
  ]
}
DEF CASOVAC TimeSensor {}
DEF ANIMACE OrientationInterpolator {
  key      [       0,          .5,         1.0 ]
  keyValue [ 0 1 0 0, 0 1 0 3.14, 0 1 0 6.28 ]
}
ROUTE DOTEK.touchTime TO CASOVAC.startTime
ROUTE CASOVAC.fraction_changed TO ANIMACE.set_fraction
ROUTE ANIMACE.value_changed TO RODIC.rotation
```

12.7.4 TouchSensor (Detektor dotyku)

Dynamický uzel `TouchSensor` je citlivý na aktivitu avatarovy ruky, avšak negeneruje žádné transformace. Je schopen pracovat jen jako detektor dotyku zjišťovat, zda avatarova ruka nad nějakým objektem pouze „přejela“ (přesunutí kurzoru), či zda se avatar pokusil objekt „uchopit“ (aktivace kurzoru).

Typ parametru	**Parametr**	**Iniciální hodnota**	**Význam**
exposedField	enabled	TRUE	Povolení práce časovače
eventOut	isOver		Kurzor se dostal nad objekt nebo jej opustil.
	isActive		Stav tlačítka pro aktivaci kurzoru se změnil
	touchTime		Čas uvolnění tlačítka pro aktivaci kurzoru
	hitPoint_changed		Bod na povrchu objektu, na který ukazuje kurzor, se změnil
	hitNormal_changed		Normála v bodě, který odpovídá parametru `hit_Point_changed`
	hitTexCoord_changed		Souřadnice textury v bodě, který odpovídá parametru `hit_Point_changed`

Tabulka 49: Seznam parametrů uzlu TouchSensor

Příklad TouchSensor.wrl

```
#VRML V2.0 utf8
DEF RODIC Transform {
  children [
    Shape {
      appearance Appearance {
        material DEF MOJE Material { diffuseColor 0 1 0 }
      }
      geometry Box {}
    }
    DEF DOTYK TouchSensor {}
  ]
}
DEF CASOVAC TimeSensor { cycleInterval 2.0 }
DEF ANIMACE ColorInterpolator {
  key      [ 0,        1.0 ]
  keyValue [ 1 0 0 , 0 1 0 ]
}
ROUTE DOTYK.touchTime TO CASOVAC.startTime
ROUTE CASOVAC.fraction_changed TO ANIMACE.set_fraction
ROUTE ANIMACE.value_changed TO MOJE.diffuseColor
```

V příkladu je uvedena animace, která bude vysvětlena níže v kapitole Interpolace.

12.7.5 VisibilitySensor (Detektor viditelnosti)

`VisibilitySensor` detekuje změnu viditelnosti prostoru vymezeného imaginárním kvádrem během avatarova procházení virtuálním světem. Tento senzor se typicky používá pro detekci toho, zda uživatel může v daném okamžiku vidět určitý objekt nebo oblast scény, a k následnému spuštění nebo zastavení nějaké akce.

Typ parametru	**Parametr**	**Iniciální hodnota**	**Význam**
exposedField	center	0 0 0	Těžiště kvádru, jehož viditelnost je sledována
	size	0 0 0	Nezáporné rozměry kvádru
	enabled	TRUE	Povolení práce detektoru

Typ parametru	Parametr	Iniciální hodnota	Význam
eventOout	isActive		Část kvádru se objevila na obrazovce.
	enterTime		Čas, ve kterém nabude parametr `isActive` hodnoty `TRUE`
	exitTime		Čas, ve kterém nabude parametr `isActive` hodnoty `FALSE`

Tabulka 50: Seznam parametrů uzlu VisibilitySensor

Rozměry sledované oblasti jsou transformovány podle parametrů případného rodičovského uzlu `Transform`.

Existuje-li několik různých uzlů `VisibilitySensor`, jejichž sledované oblasti se překrývají, může být jejich viditelnost detekována současně několika uzly.

12.8 Interpolátory

Interpolátor slouží ve virtuálním světě jako pohon událostí. Dříve, než popíšeme, zastavíme se nad pojmem *interpolace*. Tímto slovem se označuje proces vyhledávání mezilehlých hodnot na základě předem daných konstant, nazývaných klíčové hodnoty. Každá klíčová hodnota se vztahuje k nějakému času. Také nově vypočítávaná hodnota je určena časem (T_i) a lze ji proto snadno nalézt mezi dvojicí sousedních klíčových hodnot. Vypočítávaná veličina leží v grafu na spojnici klíčových hodnot, tedy na přímce. Proto se tomuto typu interpolace říká lineární.

Lineární interpolace umožňuje na základě malého množství klíčových hodnot generovat velké množství nových hodnot. To je příhodné pro animace. Chceme-li například pohybovat nějakým tělesem po prostorové dráze, stačí zvolit interpolátor pro výpočty prostorových souřadnic a zadat mu ty polohy tělesa, v nichž se dráha výrazněji zakřivuje. Interpolátor pak dopočítá souřadnice všech ostatních poloh.

Existuje celkem šest různých uzlů, které provádějí interpolaci. Každý z nich je zaměřen na jiný typ vypočítávaných hodnot V_i. Všechny mají společné to, že první z klíčových hodnot se vztahuje k počátečnímu času označenému jako nula a poslední ke koncovému času označenému jako jedna.

12.8.1 ColorInterpolator (Interpolace barvy)

Tento interpolátor vysílá událost obsahující právě jednu barvu, tedy údaj typu `SFColor`. Tím je rozsah možných aplikací omezen na interpolaci barev světelných zdrojů a jednotlivých parametrů uzlu `Material`. Poslední parametrem, který lze takto interpolovat, je barva mlhy.

Typ paramet-ru	Parametr	Iniciální hodnota	Význam
exposedField	key	[]	Neklesající posloupnost řídících klíčů
	keyValue	[]	Seznam barev RGB – hodnot pro odpovídající řídící klíče
eventIn	set_fraction		Aktuální řídící hodnota
eventOout	value_changed		Interpolovaná barva

Tabulka 51: Seznam parametrů uzlu ColorInterpolator

Příklad ColorInterpolator.wrl

```
#VRML V2.0 utf8
DEF MOJEBARVA ColorInterpolator {
  key [0.0, 0.5, 1.0]
  keyValue [1 0 0, 0 0 1, 1 0 0]
}
DEF MOJEHODINY TimeSensor {
  cycleInterval 10.0      # 10 sekund animace
  loop TRUE              # povolení opakování
}
Shape {
  appearance Appearance {
    material DEF MUJMATERIAL Material { }
  }
  geometry Sphere { }
}
ROUTE MOJEHODINY.fraction_changed TO MOJEBARVA.set_fraction
ROUTE MOJEBARVA.value_changed TO MUJMATERIAL.set_diffuseColor
```

12.8.2 CoordinateInterpolator (Interpolace souřadnic)

`CoordinateInterpolator` provádí lineární interpolaci mezi množiny `MF-Vec3f` hodnot. Používá se k interpolaci mezi skupinami bodů. Počet vektorů v poli `keyValue` musí být celistvým násobkem počtu hodnot v poli `key`. Tento násobek pak určuje počet vektorů, který bude obsahovat výstupní událost `value_changed`.

Typ paramet-	Parametr	Iniciální	Význam

ru		hodnota	
exposedField	key	[]	Neklesající posloupnost řídících klíčů
	keyValue	[]	Pole seznamů souřadnic – hodnot pro odpovídající řídící klíče
eventIn	set_fraction		Aktuální řídící hodnota
eventOut	value_changed		Seznam interpolovaných souřadnic

Tabulka 52: Seznam parametrů uzlu CoordinateInterpolator

Protože uzel generuje najednou celý seznam souřadnic, počet jednotlivých souřadnic v seznamu `keyValue` by měl být N x M, kde M je požadovaná délka výstupního seznamu souřadnic a N počet hodnot v parametru `key`.

```
#VRML V2.0 utf8
#ukázka zápisu CoordinateInterpolátoru
CoordinateInterpolator{
  key[0,1]
  keyValue[0 1 0, 0 -1 0]
}
```

12.8.3 NormalInterpolator (Interpolace normál)

Tento interpolátor provádí lineární interpolaci mezi množinou `MFVec3f` hodnot. Uzel je vhodný pro transformaci normálových vektorů. Počet normál v poli `keyValue` musí být celistvým násobkem počtu klíčových hodnot v poli `key`. Tento násobek pak určuje počet normál, který bude obsahovat výstupní událost `value_changed`.

Interpolace probíhá na povrchu jednotkové koule. Výstupní hodnoty při interpolaci mezi dvěma body P a Q, nacházejícími se na povrchu jednotkové koule, budou odpovídat bodům ležícím podél nejkratšího oblouku, které na povrchu této koule spojuje oba body. Všechny výstupní vektory jsou normalizovány.

Typ parametru	Parametr	Iniciální hodnota	Význam
exposedField	key	[]	Neklesající posloupnost řídících klíčů
	keyValue	[]	Pole seznamů normál – hodnot pro odpovídající řídící klíče
eventIn	set_fraction		Aktuální řídicí hodnota
eventOut	value_changed		Seznam interpolovaných normál

Tabulka 53: Seznam parametrů uzlu NormalInterpolator

Pokud body, mezi jejichž souřadnicemi se interpoluje, leží na povrchu jednotkové koule přímo proti sobě (přímka, která by se spojovala, by procházela středem koule), potom je lze spojit nekonečným množstvím stejně dlouhých oblouků a pro interpolaci se použije libovolný z nich.

```
        #VRML V2.0 utf8
        #ukázka zápisu NormalInterpolátoru
        NormalInterpolator{
          key[0,1]
          keyValue[0 1 0, 0 -1 0]
}
```

12.8.4 OrientationInterpolator (Interpolace orientace)

Tento uzle provádí lineární interpolaci mezi množinou hodnot typu SFRotation. Rotace jsou chápány jako absolutní v rámci lokálního souřadného sytému objektu. Pole keyValue musí obsahovat přesně tolik hondot typu SFRotation, kolik prvků obsahuje pole key, jinak se generuje chyb a výsledek interpolace je nedefinovaný.

Orientace reprezentuje konečnou pozici objektu po provedení rotace. OrientationInterpolator provádí interpolaci mezi dvěma orientacemi výpočetem nejkratší cesty mezi nimi na jednotkové kouli. interpolace probíhá lineárně po oblouku podél této cesty.

Typ parametru	**Parametr**	**Iniciální hodnota**	**Význam**
exposedField	key	[]	Neklesající posloupnost řídících klíčů
	keyValue	[]	Seznam orientací – hodnot pro odpovídající řídicí klíče
eventIn	set_fraction		Aktuální řídicí hodnota
eventOut	value_changed		Interpolovaná orientace

Tabulka 54: Seznam parametrů uzlu OrientationInterpolator

Příklad OrientationInterpolator.wrl

```
        #VRML V2.0 utf8
        DEF MojeRotace OrientationInterpolator {
          key [0.0, 0.5, 1.0]
          keyValue [0 1 0 0, 0 1 0 3.14, 0 1 0 6.28]
        }
        DEF CAS TimeSensor {
          cycleInterval 5.0
          loop  TRUE
        }
        DEF OBJEKT Transform {
```

```
  children [
    Shape {
      appearance Appearance {
        material Material { diffuseColor 1 0 0 }
      }
      geometry Box { }
    }
  ]
}
ROUTE CAS.fraction_changed TO MojeRotace.set_fraction
ROUTE MojeRotace.value_changed TO OBJEKT.rotation
```

12.8.5 PositionInterpolator (Interpolace polohy)

PositionInterpolator provádí lineární interpolaci mezi množinou SFVec3f hodnot. Vektory jsou chápány jako absolutní určení pozice. Pole keyValue musí obsahovat stejný počet prvků jako pole key.

Typ paramet-ru	Parametr	Iniciální hodnota	Význam
exposedField	key	[]	Neklesající posloupnost řídících klíčů
	keyValue	[]	Seznam prostorových souřadnic – hodnot pro odpovídající řídicí klíče
eventIn	set_fraction		Aktuální řídicí hodnota
eventOut	value_changed		Interpolované souřadnice bodu

Tabulka 55: Seznam parametrů uzlu PositionInterpolator

Příklad PositionInterpolator.wrl

```
#VRML V2.0 utf8
DEF OBJEKT Transform {
  children [
    Shape {
      appearance Appearance {
        material Material { diffuseColor 1 0 0 }
      }
      geometry Box {}
    }
  ]
}
DEF CASOVAC TimeSensor{
  loop TRUE
  cycleInterval 4.0
}
DEF ANIMACE PositionInterpolator {
```

```
  key[0, 0.5, 1.0]
  keyValue [-1 0 0, 1 0 0, -1 0 0]
}
ROUTE CASOVAC.fraction_changed TO ANIMACE.set_fraction
ROUTE ANIMACE.value_changed TO OBJEKT.translation
```

12.8.6 ScalarInterpolator (Interpolace čísla)

Tento uzel provádí lineární interpolaci mezi množinou `SFFloat` hodnot. Tento interpolátor je vhodný pro parametry typu `SFFloat`, jako jsou `width`, `radius`, `intensity` atd. Pole `keyValue` musí obsahovat stejný počet prvků jako pole `key`.

Typ parametru	Parametr	Iniciální hodnota	Význam
exposedField	key	[]	Neklesající posloupnost řídících klíčů
	keyValue	[]	Seznam čísel – hodnot pro odpovídající řídící klíče
eventIn	set_fraction		Aktuální řídicí hodnota
eventOut	value_changed		Interpolované číslo

Tabulka 56: Seznam parametrů uzlu ScalarInterpolator

```
#VRML V2.0 utf8
#ukázka zápisu ScalarInterpolátoru
ScalarInterpolator{
  key[0, .25, .5, .75, 1]
  keyValue[0, 1, 2, 1, 0]
}
```

13 Script

Uzel `Script` se používá k programování chování objektů ve scéně. Můžeme do něj vložit program, s jehož pomocí rozšíříme chybějící vlastnosti běžných uzlů a který zajistí mnohem bohatší škálu interaktivních možností.

13.1 Script

Do uzlu `Script` pochopitelně nelze vkládat programy v libovolném programovacím jazyku. V současné době jsou povoleny pouze dva jazyky. Prvním z nich je **Java** – moderní, rozsáhlý, objektově orientovaný jazyk, který je svými schopnostmi srovnatelný s klasickými programovacími jazyky, jakým je například C++. Druhým jazykem, jímž se definují funkce v uzlu `Script`, je jazyk **ECMAScript**. Toto jmé-

no je oficiálním názvem jazyka Netscape JavaScript v podobě mezinárodní normy ISO. Jde o jednoduchý, přímo interpretovaný jazyk, který se v některých rysech podobá jazyku Java, avšak dovoluje snadnou a intuitivní práci, byť s omezenou škálou prostředků.

Typ parametru	**Parametr**	**Iniciální hodnota**	**Význam**
exposedField	url	[]	Adresy souboru s obslužnými funkcemi či jejich přímý zápis v uzlu
field	directOutput	FALSE	Funkce mohou přímo zasílat události jiným uzlům
	mustEvaluate	FALSE	Okamžité vyvolání funkcí po přijetí události uzlem
+ libovolný počet parametrů typu:			
Typ parametru	**Typ dat**	**Název**	**Inicializační hodnota**
field	typ dat	jméno	Inicializační hodnota
eventInt	typ dat	jméno	
eventOut	typ dat	jméno	

Tabulka 57: Seznam parametrů uzlu Script

13.1.1 Url

Nejdůležitějším parametrem je `url`, který obsahuje adresu souboru, v němž jsou zapsány potřebné funkce uzlu. Síla skriptu spočívá v možnosti zavedení libovolného počtu dalších parametrů. V uzlu Script jsou pak parametry zpracovávány vnitřními funkcemi. Platí tato pravidla:

1. Parametr třídy `field` musí mít definovánu iniciální hodnotu. Funkce skriptu mohou tuto hodnotu měnit, parametr tedy slouží jako tzv. lokální proměnná.
2. Ke každému parametru třídy `eventIn` musí existovat stejnojmenná funkce. Tato funkce se provede pokaždé, když parametr získá novou hodnotu, tj. když přijme událost. Každá taková funkce má proto dva vstupní parametry – přijímanou hodnotu a čas vzniku události.
3. Z parametru třídy `eventOut` je vyslána událost pokaždé, když je přiřazena nová hodnota do stejnojmenné proměnné uvnitř libovolné z funkcí. Je-li uvnitř funkce přiřazena hodnota do téže proměnné několikrát, vyšle se z odpovídajícího parametru jen jediná událost s poslední hodnotou.

4. Skript nesmí obsahovat parametr třídy `exposedField`. Toto omezení lze překonat zavedením trojice parametrů `field`, `eventIn` a `eventOout` s odpovídajícími názvy `x, set_x` a `x_changed`.

Prví praktický příklad skriptu má za úkol převést vstupní hodnoty typu SFBool (TRUE a FALSE) na čísla 1 či 0. Zavedeme v něm tedy parametr třídy eventIn a pojmenujeme jej set_bool. Jméno parametru může být sice libovolné, ale je vhodné dodržovat zavedené konvence VRML a výběrem jména naznačit Význam parametru. Parametr třídy eventOout proto podle této zvyklosti pojmenujeme value_changed.

```
Script{
  eventIn SFBool set_bool
  eventOut SFInt32 value_changed
  url "javascript:
    function set_bool (hodnota, cas)
    {if (hodnota) value_changed = 1;
    else value_changed = 0;}"
}
```

Podle výše uvedených pravidel je potřeba ke každému vstupnímu parametru napsat stejnojmennou funkci. V našem případě potřebujeme jen jedinou funkci, s názvem `set_bool`. Její definici umístíme přímo do skriptu, a proto musí parametr `url` obsahovat text začínající klíčovým slovem `javascript`.

13.1.2 MustEvaluate

Tento parametr je nastaven na hodnotu `FALSE`, prohlížeč nemusí reagovat na každou vstupní událost vyhodnocováním funkcí uvnitř skriptu. Může provést jen ty funkce, které zapisují hodnoty do výstupních parametrů propojených konstrukcí `ROUTE` s jinými uzly. Hodnota `TRUE` parametru `mustEvaluate` zajistí vždy korektní chování skriptu, ovšem za cenu možného zpomalení výpočtů.

13.1.3 DirectOutput

Druhým parametrem ovlivňujícím efektivitu je `directOutput`. Používá se tehdy, pokud skript zasílá události dalším uzlům. Existují dva základní způsoby předávání dat. První způsob je realizován již známou konstrukcí `ROUTE`, která propojuje parametr skriptu s parametrem jiného uzlu. V tomto případě nemá skript možnost ovlivnit jiné parametry uzlu nežli ty, se kterými je výslovně definováno propojení. Při druhém způsobu se uzel uvede mezi parametry skriptu, tedy jako parametr typu `SFNode` nebo `MFNode`. Skript pak může přistupovat k jeho parametrům nejen pomocí konstrukce `ROUTE`, ale i přímým zápisem v programovacím jazyku. Parametr `directOutput` určuje, zda je tento přímý zápis povolen nebo ne. Hodnota `TRUE` umožňuje přímou manipulaci se všemi parametry daného uzlu, ale od `prohlížeče` vyžaduje provádění časově náročnějších kontrol.

13.2 Speciální funkce ECMAScriptu

Funkce uvnitř skriptů obsluhují buď jednotlivé vstupní události, nebo slouží jako pomocné funkce pro dílčí výpočty. Navíc však existují tři funkce s předem definovanými jmény a vlastnostmi, které mohou být zahrnuty do libovolného skriptu.

13.2.1 Initialize ()

Funkce se provede právě jednou, těsně před tím, než je svět prezentován uživateli. Jejím úkolem je nastavit iniciální hodnoty, provést případnou iniciální posloupnost příkazů.

13.2.2 Shutdown ()

Funkce se provede právě jednou, po skončení prohlížení světa, tj. před přechodem do dalšího světa či WWW-stránky. Existence funkce umožňuje, aby například byla zaznamenána poslední poloha avatara ve virtuálním světě a obnovena při jeho další návštěvě.

13.2.3 EventsProcessed ()

Funkce se provádí pokaždé, když je skriptu zaslána nějaká událost. Slouží ke snížení počtu dalších vysílaných událostí, protože může „akumulovat“ dílčí vstupní události různých typů a vyslat jednu výstupní událost až při dosažení určitého stavu, splnění podmínek. Kupříkladu trezor se třemi zámky nemusí vysílat informaci o svém stavu po každém zasunutí jednotlivého klíče, ale až po úplném odemčení.

Prázdné závorky za jmény funkcí říkají, že funkce nemají žádné parametry. Práce s těmito funkcemi je charakteristická pro složitější světy vytvářené zkušenějšími programátory.

14 Software Cosmo Worlds 2.0

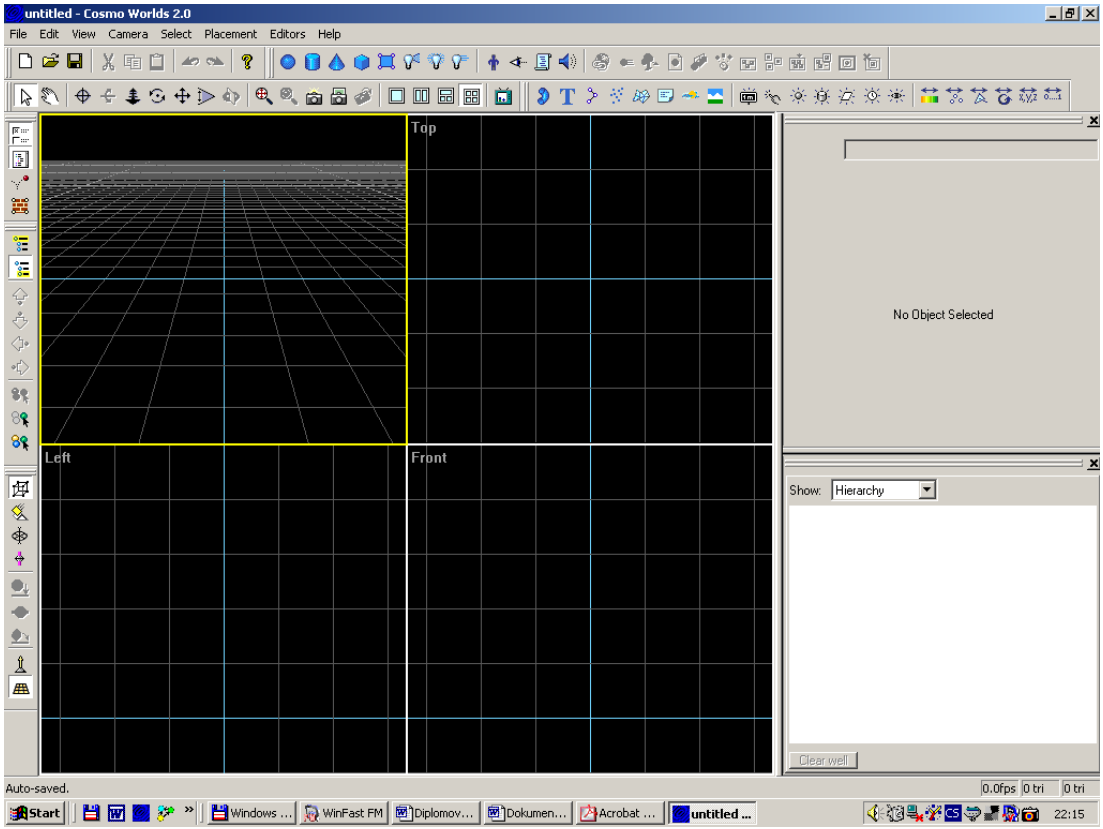
Cosmo Worlds firmy Silicon Graphics (SGI) je špičkový modelovací program plně podporující VRML 2.0. S jeho použitím se dají jednoduše vytvářet VRML světy, které mohou tvořit součást WWW stránek. Tento program není zaměřen na vytváření pouze jednoduchých scén a těles, ale dovoluje modelovat poměrně složité objekty, které mohou být následně rozpohybovány v plynulých animacích.

14.1 Požadavky na systém

Cosmo Worlds můžete nainstalovat na počítač s operačním systémem Windows 95, 98 nebo Windows NT 4.0. Aby byla zajištěna alespoň minimální pracovní rychlost, doporučují autoři programu procesor Pentium 90 Mhz a 32 MB (raději 64 MB) paměti RAM.Průměrná instalace ukrojí z vašeho pevného disku 27 MB. Pokud si budete přát vytvářet programy v Javě, budete nuceni nainstalovat JDK 1.1.4 (Java Development Kit). Pro výsledné zobrazení budete potřebovat Internet Explorer 3.0.2 nebo vyšší společně s některým z prohlížečů VRML (Cosmo Player 2.0). Pro práci s rozsáhlejšími scénami se doporučuje grafická karta alespoň s rozlišením 800 x 600 pixelů a s barevnou hloubkou aspoň 8 bitů (256 barev).

14.2 Uživatelské prostředí

Uživatelské prostředí tohoto programu je mimořádně propracované a efektivním způsobem pomáhá uživateli při vytváření objektů. Skládá se z mnoha panelů nástrojů a pomocných oken,která zobrazují aktuální stav animací, barev, poloh a tvarů. Každý z těchto panelů může být uživatelem umístěn na libovolné místo pracovní plochy nebo připojen k ostatním panelům.



Obrázek 30: Program Cosmo Worlds 2.0

14.2.1 Panely pro vytváření objektů

14.2.1.1 Panel Create

Obrázek 31: Panel nástrojů Create

Prvním z celé plejády panelů nástrojů je panel pro vytváření jednoduchých objektů (`Sphere`, `Cylinder`, `Cone`, `Box` a `IndexFaceSet`) a světel (`kuželové`, `bodové` a `přímé`). Dále pak umožňuje přidat do scény uzly `NavigationInfo`, `Viewpoint`, `Script`, `Sound`. Další v řadě jsou `Anchor`, `Collision`, `Billboard`, `LOD` a `Switch`.

Pro práci se skupinami dodali autoři modeláře k tomuto panelu několik tlačítek. S jejich pomocí lze vytvořit sjednocení několika uzlů, přidat uzel k již existujícímu sjednocení, odebrat uzel ze sjednocení a zrušit celé sjednocení.

14.2.1.2 Panel Create Extras

Obrázek 32: Panel nástrojů Create Extras

Druhým panelem pro tvorbu objektů je panel Create Extras, který slouží k vytváření složitějších objektů (`Extrusion`, `Text`, `IndexedLineSet`, `PointSet`) a k simulaci přírodních scén (`ElevationGrid`, `WorldInfo`, `Fog` a `Background`).

S pomocí tohoto panelu lze dále vytvářet senzorové uzly (`ProximitySensor`, `TouchSensor`, `SphereSensor`, `CylinderSensor`, `PlaneSensor`, `TimeSensor` a `VisibilitySensor`) a interpolátory (`ColorInterpolator`, `CoordinateInterpolator`, `NormalInterpolator`, `OrientationInterpolator`, `PositionInterpolator` a `ScalarInterpolator`).

14.2.1.3 Panel Standard

Obrázek 33: Panel Standard

Panel Standard slouží k ovládání programu. První ikona slouží k otevření nového souboru. Další pak k otevření již stávajícího souboru, uložení, vyjmutí, kopírování, vložení objektů. Krok zpět a dopředu a samozřejmě nesmí chybět nápověda.

14.2.1.4 Panel Camera

Obrázek 34: Panel Camera

Panel Camera slouží pro možnosti nastavení pracovní plochy a pohledy na ní. První ikona slouží k vybírání objektů, ikona „ručičky“ zase slouží pro otáčení scény a prohlížení ze všech úhlů. Další ikona slouží k přiblížení vybraného objektu, rotace tělesa, přiblížení a oddálení scény, automatické otáčení scény, posunutí scény na objekty, které potřebujeme vidět. Dále je zde také zoom tělesa a pohled kamery. Další řada tlačítek slouží pro zobrazení scény (celá scéna, jenom obsazená scéna a pak pohledy). Posledních pět ikon slouží k nastavení pracovní plochy. Podle počtu okének v ikoně, se na ploše nastaví odpovídající počet pohledů na scénu. Poslední tlačítko slouží k prohlídnutí scénu (pokud máte nainstalovaný Cosmo Player 2.0).

14.2.1.5 Panely Editors a Placement

Obrázek 35: Panely Editors a Placement

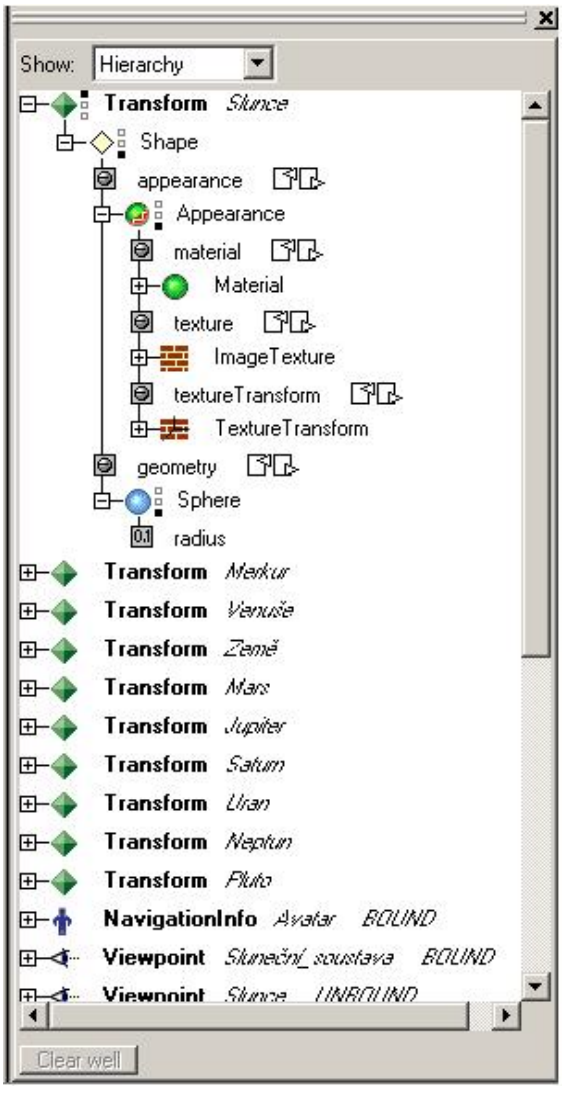
Panel Editors slouží k zapínaní a vypíná parametrů, editace, animace a textury. Panel Placement slouží k zobrazování pracovní plochy. Další plochy mají již jenom detailní charakter. Zobrazení všech parametrů nebo jenom těch, které potřebujeme atd.

14.2.2 Struktura a scény

V okně Struktura scény je obsažena struktura celé scény. Jednotlivé uzly jsou řazeny hierarchicky do několika stromů podle toho, který uzel je jejich rodičem. Nemá-li uzel žádného rodiče, potom se stává kořenem nového stromu. Takové struktuře se potom říká les.

Protože autoři programu předpokládali, že některé stromy (podstromy) mohou být rozsáhlé, připravili pro uživatele dva prvky, s jejichž pomocí lze redukovat velikost zobrazovacích stromů:

- Při zobrazování stromů vám pomohou symboly [-] a [+]. Symbol [+] znamená,že strom (podstrom) v sobě skrývá další podstromy, ale příslušný podstrom není rozbalen. Rozbalíte ho kliknutím na tento symbol. Symbol [-] znamená,že strom (podstrom) je rozbalen. Kliknutím na tento symbol ho zabalíte.
- Při zobrazování atributů u jednotlivých uzlů vám pomůže přepínač, který se používá při zobrazování atributů uvnitř uzlů. Pokud je přepínač v horní poloze, nebudou se atributy zobrazovat vůbec. Při zapnuté prostřední poloze



Obrázek 36: Stromová struktura

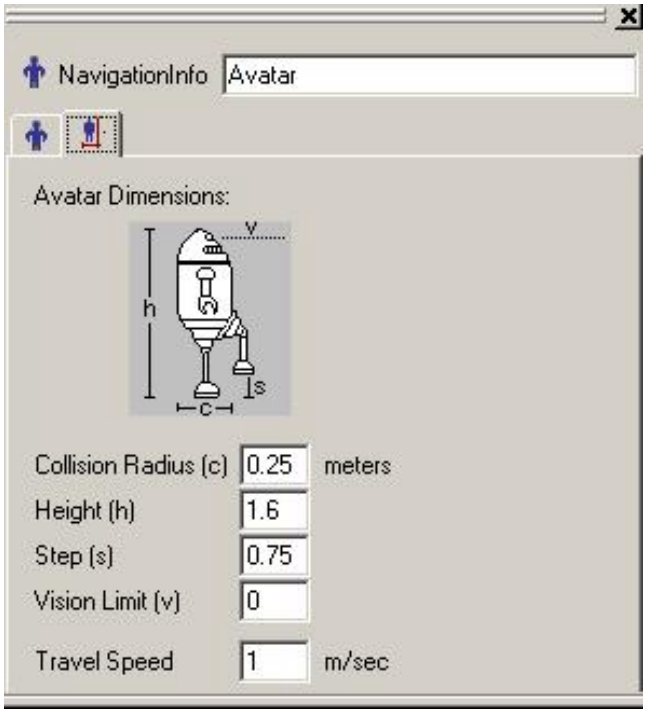
jsou zobrazeny pouze "nejlepší" atributy, tzn. atributy, které mají nějaký zásadní Význam pro daný uzel. Poslední poloha přepínače znamená, že budou zobrazeny všechny atributy, které uzel osahuje.

Dialogové okno Struktura scény neslouží pouze k zobrazování. S jeho pomocí lze zadávat i konkrétní hodnoty do atributů v zobrazených uzlech, a to pouhým kliknutím na atribut a zadáním hodnoty přímo z klávesnice.

Posledním úkolem tohoto okna je umožnit uživateli propojovat jednotlivé uzly prostřednictvím událostí. Atributy sloužící k zasílání zpráv se dělí do třech skupin. První skupinu tvoří atributy, které mohou pouze vysílat svou hodnotu (`eventOut`). Druhá skupina ji může pouze přijímat (`eventIn`) a poslední skupinu tvoří atributy, které jsou schopné hodnoty přijímat, tak i vysílat (`exposedField`). Cosmo Worlds je odlišuje pomocí dvou ikon. První ikona v podobě listu papíru obsahuje šipku, která směřuje dovnitř, což představuje příchozí událost. Druhá ikona pak obsahuje šipku vedoucí ven neboli výstupní událost. Pokud může být hodnota atributu jak přijímána, tak i vysílána, budou za atributem uvedeny obě ikony.

14.2.3 Interaktivní editace vlastností

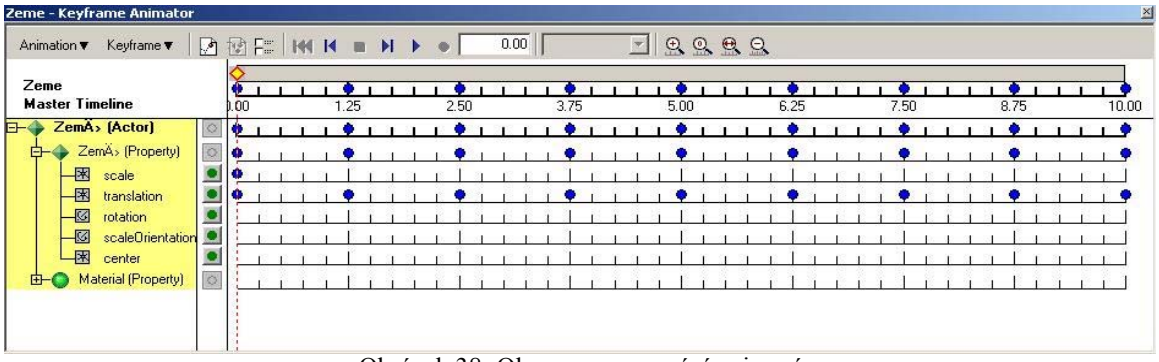
Některé vlastnosti uzlů se dají nastavit přímo v okně se strukturou scény, další možnost se pak naskýtá v okně Property Inspector, které bývá většinou součást pravého panelu nástrojů a nelze ho přehlédnout. Jeho obsah se mění podle kontextu uživatelovy práce, respektive podle toho,který uzel si uživatel právě vybral.



Obrázek 37: Okno Property Inspector

14.2.4 Nástroje pro animaci

Pro vytváření animací připravili autoři nástroj Keyframe Animátor neboli editor klíčových snímků, s jehož pomocí lze jednoduchým způsobem rozpohybovat jedno i několik těles.



Obrázek 38: Okno na zpracování animací

14.2.5 Script Editor

Posledním významným prvkem je na modeláři jeho textový editor, který dovoluje jednak psát programy v jazyce JavaScriptu. Další příjemnou vlastností editoru je jeho schopnost přidávat (ubírat) k již existujícím událostem události další s tím, že si může uživatel vybrat,zda se jedná o událost typu `eventIn` nebo `eventOut`, nebo zda se jedná o atribut bez možnosti s ním pracovat jako s událostí (`field`). Dále pak umožňuje určit typ dané události, a to opět výběrem ze všech možností.

15 Závěrečné shrnutí

Ten, kdo chce plně využít všech možností jazyka VRML, se však neobejde bez kompletní specifikace všech parametrů. Abecedně řazený přehled všech uzlů VRML, jejich parametrů včetně iniciálních hodnot naleznete v Příloze.

Nejprve znovu shrňme, do jakých logických skupin můžeme uzly rozdělit:

- Informační - NavigationInfo, Viewpoint, WorldInfo
- Geometrie (tvar)
  - Tělesa - Box, Cone, Cylinder, Sphere
  - Plochy - ElevationGrid, Extrusion, IndexedFaceSet
  - Ostatní - IndexedLineSet, PointSet, Text
  - Pomocné - Shape, FontStyle
  - Datové - Coordinate, Normal, TextureCoordinate
- Vzhled povrchu - Appearance, Color, ImageTexture, Material, MovieTexture, pixelTexture, TextureTransform
- Prostředí - AudioClip, Background, DirectionalLight, Fog, PointLight, Sound, SpotLight
- Skupinové - Anchor, Billboard, Group, inline, LOD, Switch, Transform
- Speciální - Script, TimeSensor
- Manipulátory - CylinderSensor, PlaneSensor, SphereSensor
- Interpolátory - ColorInterpolator, CoordinateInterpolator, NormalInterpolator, OrientationInterpolator, PositionInterpolator, ScalarInterpolator
- Detektory - Collision, ProximitySensor, TouchSensor, VisibilitySensor

Uzly v rámečku patří mezi vázané, tedy ty, z nichž v každém okamžiku je platný (aktivní) maximálně jeden toho jména.

Další přílohou je CD, na kterém je uveden soubor s diplomovou prací, dále pak zdrojové kódy k programům a také obrázky.

Na závěr bych chtěla uvést, že většina příkladů má své praktické ukázky. Tyto příklady jsou uloženy v adresáři Teoretická část\Ukázky VRML světů a mají shodné názvy jako jsou probírané uzly.

Seznam použité literatury

Knihy:

[1] J. Zrzavý: VRML tvorba dokonalých www stránek, Grada 1999

[2] J. Žára: VRML97 laskavý průvodce virtuálními světy, Computer press 1999

Elektronická publikace:

[3] http://sgi.felk.cvut.cz/LaskavyPruvodce/index.html

[4] http://www.vrml.org/Specifications/VRML97

[5] http://www.cosmosoftware.com/

[6] http://www.cgg.cvut.cz/vyuka/VRML/tutorial/pasmo/vrml97.html

[7] http://home.pf.jcu.cz/~pepe/Diplomky/Sik/vrml.htm

[8] http://www.cpress.cz/knihy/vrml/

[9] http://www.ocnus.com/models/models.html

[10] http://www.geocities.com/MotorCity/4630/criar/cosmoworlds.html

[11] http://www.technion.ac.il/guides/Cosmo/worlds/

[12] http://home.pf.jcu.cz/~pepe/Diplomky/haman.doc

[13] http://home.pf.jcu.cz/~pepe/Diplomky/zelezny.pdf

[14] http://www.cgg.cvut.cz/publications/diplom/AdamecJaroslav/download_folder/JaroslavAdamec.pdf

[15] http://sgi.felk.cvut.cz/~apg/apg-tutorials01/ch03.html#web-2001s05

Příloha – Detailní popis VRML uzlů

Anchor (Teleportace, odkaz)

Typ parametru	**Název**	**Iniciální hodnota**	**Význam**
ExposedField	url	[ ]	Seznam adres jiných světů, jmen stanovišť a WWW-stránek
	parameter	[ ]	Seznam doplňujících parametrů předávaných prohlížeči po aktivaci
	description	„“	Informační text
	children	[ ]	Seznam potomků
Field	bboxSize	-1 –1 –1	Kladné délky stran pomocné obálky ve tvaru kvádru; výjimka [-1 –1 –1] indikuje dosud nespecifikovanou velikost kvádru
	bboxCenter	0 0 0	Souřadnice středu pomocné obálky ve tvaru kvádru
EventIn	addChildren		Seznam připojovaných uzlů – potomků
	removeChildren		Seznam odstraňovaných uzlů - potomků

AudioClip (Zvukový soubor)

Typ parametru	**Název**	**Iniciální hodnota**	**Význam**
ExposedField	url	[ ]	Seznam adres s umístěním zvukového souboru
	description	„“	Informační text
	startTime	0	Čas zahájení přehrávání
	stopTime	0	čas ukončení přehrávání
	pitch	1.0	kradná rychlost přehrávání (tempo)
	loop	FALSE	povolení přehrávání ve smyčce
eventOut	duration_changed		původní délka zvuku v sekundách; je vyslána po načtení souboru do paměti
	isActive		bylo zahájeno nebo ukončeno přehrávání

Background (Pozadí, panorama)

Typ parametru	Název	Iniciální hodnota	Význam
exposedField	backUrl	[ ]	seznam adres s umístěním obrázku pro zadní stěnu obklopující krychle
	bottomUrl	[ ]	
	frontUrl	[ ]	
	leftUrl	[ ]	
	RightUrl	[ ]	
	topUrl	[ ]	
	skyColor	0 0 0	seznam barev pro postupné přechody na sférické obloze
	skyAngle	[ ]	rostoucí posloupnost nezáporných úhlů, pro které jsou dány barvy oblohy
	groundColor	[ ]	seznam barev pro postupné přechody na sférické zemi (podlaze)
	groundAngle	[ ]	rostoucí posloupnost nezáporných úhlů, po které jsou dány barvy země
eventInt	set_bind		aktivování pozadí
eventOut	isBound		pozadí se stalo aktivním nebo naopak bylo nahrazeno jiným

Billboard

Typ parametru	Název	Iniciální hodnota	Význam
exposedField	children	[ ]	seznam potomků
	axisOf	0 1 0	osa otáčení
field	bboxSize	-1 –1 –1	kladné délky stran pomocné obálky ve tvaru kvádru; výjimka (-1 –1 –1) indikuje dosud nespecifikovanou velikost kvádru
	bboxCenter	0 0 0	souřadnice středu pomocné obálky ve tvaru kvádru
eventIn	addChildren		seznam připojovaných uzlů – potomků
	removeChildren		seznam odstraňovaných uzlů – potomků

Collision (Detekce nárazu)

Typ parametru	Název	Iniciální hodnota	Význam
exposedField	children	[ ]	seznam potomků
	collide	TRUE	povolení detekování kolizí s avatarem
field	bboxSize	-1 –1 –1	kladné délky stran pomocné obálky ve tvaru kvádru; výjimka (-1 –1 –1) indikuje dosud nespecifikovanou velikost kvádru
	bboxCenter	0 0 0	souřadnice středu pomocné obálky ve tvaru kvádru
	proxy	NULL	uzel nebo strom s náhradní geometrií, která se použije pro detekce kolize
eventIn	addChildren		seznam připojovaných uzlů – potomků
	removeChildren		seznam odstraňovaných uzlů – potomků
eventOut	collideTime		čas nárazu avatara

Color (Barvy)

Typ parametru	Název	Iniciální hodnota	Význam
exposedField	color	[ ]	seznam barev v modelu RGB

ColorInterpolator (Interpolace barvy)

Typ parametru	Název	Iniciální hodnota	Význam
exposedField	key	[ ]	neklesající posloupnost řídících klíčů
	keyValue	[ ]	seznam barev RGB – hodnot pro odpovídající řídící klíče
eventIn	set_fraction		aktuální řídící hodnota
eventOut	value_changed		interpolovaná barva

CoodinateInterpolator (Interpolace souřadnic)

Typ parametru	Název	Iniciální hodnota	Význam
exposedField	key	[ ]	neklesající posloupnost řídících klíčů
	keyValue	[ ]	pole seznamů souřadnic – hodnot pro odpovídající řídící klíče
eventIn	set_fraction		aktuální řídící hodnota
eventOut	value_changed		seznam interpolovaných souřadnic

CylinderSensor (Válcový manipulátor)

Typ parametru	Název	Iniciální hodnota	Význam
exposedField	enabled	TRUE	povolení práce manipulátoru
	offset	0	základní úhel, který je vždy přičítán k nově vyhodnocenému úhlu
	autoOffset	TRUE	povolení automatické aktualizace offset po skončeni činnosti manipulátoru
	minAngle	0	dolní mez vyhodnoceného úhlu otočení; je v intervalu (-2pí, 2pí)
	maxAngle	-1	hodní mez vyhodnoceného úhlu otočení; je v intervalu (-2pí, 2pí)
	diskAngle	0.262	kladný úhel menší než Pí/2, rozlišující, zda z pomyslného válce se použije podstava nebo plášť
eventOut	isActive		zahájení a ukončení činnosti manipulátoru
	trackPoint_changed		měnící se poloha bodu na pomyslném válci, na který ukazuje kurzor při práci s manipulátorem
	rotation_changed		měnící se hodnota úhlu odvozená z pohybu vstupního zařízení kolem osy válce

DirectionalLight (Směrový zdroj světla)

Typ parametru	Název	Iniciální hodnota	Význam
exposedField	direction	0 0 –1	směr světelných paprsků
	color	1 1 1	barva paprsků
	intensity	1	iniciální intenzita paprsků v rozsahu (0,1)
	ambientIntensity	0	příspěvek zdroje k nepřímému osvětlení virtuálního světa v rozsahu (0,1)
	on	TRUE	zapnutí či vypnutí světelného zdroje

ElevationGrid (Výšková mapa)

Typ parametru	Název	Iniciální hodnota	Význam
exposedField	color	NULL	seznam barev v uzlu Color
	normal	NULL	seznam normál v uzlu Normal
	texCoord	NULL	seznam souřadnic textury v uzlu TextureCoordinate
field	xDimension	0	počet vzorků (vrcholů sítě v ose x)
	Zdimension	0	počet vzorků (vrcholů sítě v ose z)
	xSpacing	1.0	kladná vzdálenost mezi vzorky v ose x
	zSpacing	1.0	kladná vzdálenost mezi vzorky v ose z
	height	[ ]	pole výšek všech vrcholů sítě
	colorPerVertex	TRUE	barvy v parametru color se vztahují na vrcholy, jinak na celé plochy sítě
	normalPerVertex	TRUE	normály v parametru normal se vztahují na vrcholy, jinak na plochy
	ccw	TRUE	přivrácená strana mapy je vidět při pohledu shora (proti ose y)
	solid	TRUE	maje je jednostranná
	creaseAngle	0	nezáporný úhel, až do kterého jsu dvě sousední plochy považovány za oblé
eventIn	set_height		změna parametru height

Extrusion (Opláštění)

Typ parametru	Název	Iniciální hodnota	Význam
field	crossSection	[1 1, 1 –1, -1 –1, -1 1, 1 1]	posloupnost bodů v rovině určující obrysovou křivku (obecně neuzavřenou)
	spine	[0 0 0, 0 1 0]	posloupnost bodů v prostoru určující trajektorii, po které je tažen obrys
	scale	1 1	seznam kladných měřítek (v rovině xz) pro každou polohu obrysu
	orientation	0 0 1 0	seznam otočení pro každou polohu obrysu
	beginCap	TRUE	povolení vykreslování dolní podstavy
	endCap	TRUE	povolení vykreslování horní podstavy
	convex	TRUE	obrys je konvexní
	ccw	TRUE	obrys je zadán proti směru hodinových ručiček
	solid	TRUE	všechny plochy jsou jednostrané
	creaseAngle	0	nezáporný úhel, až do kterého jsou dvě sousední plochy povazovány za oblé
eventIn	set_crossSection		změna parametru crossSection
	set_spine		změna parametru spine
	set_scale		změna parametru scale
	set_orentation		změna parametru orientation

Fog (Mlha)

Typ parametru	Název	Iniciální hodnota	Význam
exposedField	color	1 1 1	barva mlhy ve složkách RGB
	fogType	„LINEAR“	způsob houstnutí mlhy („Linear“ nebo „Exponential“)
	visibilityRange	0	nezáporná vzdálenost maximálního „barevného“ dohledu; hodnota 0 zcela odstraňuje mlhu
eventIn	set_bind		aktivování mlhy
eventOut	isBound		mlha se stala aktivní nebo naopak byla nahrazena jinou mlhou

FontStyle (Styl písma)

Typ parametru	Název	Iniciální hodnota	Význam
field	language	„“	dvojznaková zkratka použité abecedy
	family	„SERIF“	seznam rodin písma
	style	„PLAIN“	styl pro danou rodinu písma
	size	1.0	kladná výška písma
	spacing	1.0	nezáporná hodnota, definující vzdálenost mezi řádky jako spacing x size
	horizontal	TRUE	psaní ve vodorovném směru
	leftToRight	TRUE	psaní na řádku zleva doprava
	topToBottom	TRUE	psaní řádků shora dolů
	justify	„BEGIN“	hlavní a vedlejší způsob umístění textu

Group (Skupina)

Typ parametru	Název	Iniciální hodnota	Význam
exposedField	children	[ ]	seznam potomků
field	bboxSize	-1 –1 –1	kladné délky stran pomocné obálky ve tvaru kvádru, výjimka (-1 –1 –1) indikuje dosud nespecifikovanou velikost kvádru
	bboxCenter	0 0 0	souřadnice středu pomocné obálky ve tvaru kvádru
eventIn	addChildren		seznam připojovaných uzlů – potomků
	removeChildren		seznam odstraňovaných uzlů – potomků

IndexedLineSet (Množina čar)

Typ parametru	Název	Iniciální hodnota	Význam
exposedField	coord	NULL	seznam vrcholů v uzlu Coordinate
	color	NULL	seznam barev v uzlu Color
field	coordIndex	[ ]	psti indexů vrcholů jednotlivých čar (za-končené číslem –1)
	colorIndex		psti indexů barev pro jednotlivé vrcholy či celé čáry
	colorPevVertex	TRUE	barvy v parametru colorIndex se vztahují na vrcholy, jinak na celé čáry
eventIn	set_coordIndex		změna parametru coordIndex
	set_colorIndex		změna parametru colorIndex

Inline (Vložení)

Typ parametru	Název	Iniciální hodnota	Význam
exposedField	url	[ ]	seznam adres virtuálního světa či objektu
field	bboxSize	-1 –1 –1	kladné délky stran pomocné obálky ve tvaru kvádru, výjimka (-1 –1 –1) indikuje dosud nespecifikovanou velikost kvádru
	bboxCenter	0 0 0	souřadnice středu pomocné obálky ve tvaru kvádru

LOD (Stupeň detailu)

Typ parametru	**Název**	**Iniciální hodnota**	**Význam**
exposedField	level	[ ]	seznam reprezentací objektu s postupně klesající přesností (množstvím detailů)
field	center	0 0 0	bod, vůči kterému se měří vzdálenost objektu od avatara
	range	[ ]	rostoucí posloupnost kladných vzdáleností indikujících přepnutí represezntace

NormalInterpolator (Interpolace normál)

Typ parametru	**Název**	**Iniciální hodnota**	**Význam**
exposedField	key	[ ]	neklesající pst řídících klíčů
	keyValue	[ ]	pole seznamů normál – hodnot pro odpovídající řídící klíče
eventIn	set_fraction		aktuální řídící hodnota
eventOut	value_changed		seznam interpolovaných normál

OrientationInterpolator (Interpolace orientace, tj. natočení)

Typ parametru	**Název**	**Iniciální hodnota**	**Význam**
exposedField	key	[ ]	neklesající posloupnost řídících klíčů
	keyValue	[ ]	seznam orientací – hodnot pro odpovídající řídící klíče
eventIn	set_fraction		aktuální řídící hodnota
eventOut	value_changed		interpolovaná orientace

PlaneSensor (Rovinný manipulátor)

Typ parametru	**Název**	**Iniciální hodnota**	**Význam**
exposedField	enagled	TRUE	povolení práce manipulátoru
	offset	0 0 0	základní posunutí, který je vždy přičítáno k nově vyhodnocenému posunutí
	autoOffset	TRUE	povolení automatické aktualizace offset po skončení činnosti manipulátoru
	minPosition	0 0	dolní mez vyhodnoceného posunutí
	maxPosition	-1 –1	horní mezi vyhodnoceného posunutí
eventout	isActive		zahájení a ukončení činnosti manipulátoru
	trackPoint_changed		měnící se poloha bodu na pomyslné desce, na který ukazuje kurzor při práci s manipulátorem
	translation_changed		měnící se hodnot posunutí odvozená z pohybu vstupní zařízení

PointLight (Bodový zdroj světla)

Typ parametru	**Název**	**Iniciální hodnota**	**Význam**
exposedField	location	0 0 0	umístění bodového zdroje světla
	radius	100	nezáporný dosah osvětlení
	attenuation	1 0 0	trojice nezáporných koeficientů pro výpočet útlumu světla
	color	1 1 1	barva paprsků
	intensity	1	iniciální intenzita paprsků v rozsahu (0,1)
	ambientIntensity	0	příspěvek zdroje k nepřímému osvětlení virtuálního světa v rozsahu (0,1)
	on	TRUE	zapnutí či vypnutí světelného zdroje

PointSet (Množina bodů)

Typ parametru	Název	Iniciální hodnota	Význam
exposedField	coord	NULL	seznam bodů v uzlu Coordinate
	color	NULL	seznam barev bodů v uzlu Color

PositionInterpolator (Interpolace polohy)

Typ parametru	Název	Iniciální hodnota	Význam
exposedField	key	[ ]	neklesající pst řídících klíčů
	keyValue	[ ]	seznam prostorových souřadnic – hodnot pro odpovídající řídící klíče
eventIn	set_fraction		aktuální řídící hodnota
	value_changed		interpolované souřadnice bodu

ProximitySensor (Detektor přítomnosti)

Typ parametru	Název	Iniciální hodnota	Význam
exposedField	center	0 0 0	těžiště kvádru vymezujícího sledovanou oblast
	size	0 0 0	nezáporné rozměry kvádru
	enabled	TRUE	povolení práce detektoru
eventOut	isActive		vstup a výstup avatara do , resp. ze sledované oblasti
	enterTime		čas vstupu avatara do sledované oblasti
	exitTime		čs opuštění sledované oblasti avatarem
	position_changed		poloha avatara uvnitř oblasti se změnila
	orientation_changed		orientace avatara uvnitř oblasti se změnila

ScalarInterpolator (Interpolace čísla)

Typ parametru	Název	Iniciální hodnota	Význam
exposedField	key		neklesající pst řídících klíšů
	keyValue		seznam čísel – hodnot pro odpovídající řídící klíče
eventIn	set_fraction		aktuální řídící hodnota
eventOut	value_changed		interpolované číslo

Script (Skript)

Typ parametru	Název	Iniciální hodnota	Význam
exposedField	urt	[ ]	adresy souboru s obslužnými funkcemi či jejich přímý zápis v uzlu
field	directOutput	FALSE	funkce mohou přímo zasílat události jiným uzlům
	mustEvaluate	FALSE	okamžité vyvolání funkcí po přijetí události uzlem
+libovolný počet parametrů typu:			
field	jméno	inic. hodnota	
eventIn	jméno		
eventOut	jméno		

Sound (Zdroj zvuku)

Typ parametru	Název	Iniciální hodnota	Význam
exposedField	source	NULL	uzel se zvukovým záznamem
	location	0 0 0	umístnění zdroje zvuku
	direction	0 0 1	směr vysílaného zvuku (hlavní osa obou charakteristických eliptických oblastí)
	minBack	1	nezáporná vzdálenost zadního vrcholu elipsy plné slyšitelnosti
	minFront	1	nezáporná vzdálenost předního vrcholu elipsy plné slyšitelnosti

Typ parametru	Název	Iniciální hodnota	Význam
	maxBack	10	nezáporná vzdálenost zadního vrcholu elipsy plného útlumu
	maxFront	10	nezáporná vzdálenost předního vrcholu elipsy plného útlumu
	intensity	1	intenzita zvuku v rozsahu (0, 1)
	priority	0	priorita mezi více zdroji zvku v rozsahu (0, 1)
field	spatialize	TRUE	povolení generování prostorového zvuku podle orientace avatara

SphereSensor (Kulový manipulátor)

Typ parametru	Název	Iniciální hodnota	Význam
exposedField	enabled	TRUE	povolení práce manipulátoru
	offset	0 1 0 0	základní osa a úhel otočení, se kterými jsou vždy složeny nově vyhodnocené hodnoty
	autoOffset	TRUE	povolení automatické aktualizace offset po skončení činnosti manipulátoru
eventOut	isActive		zahájení a ukončení činnosti manipulátoru
	trackPoint_changed		měnící se poloha bodu na pomyslné kouli, na který ukazuje kurzor při práce s manipulátorem
	rotation_changed		měnící se osa a otočení odvozené z pohybu vstupního zařízení

SpotLight (Reflektor)

Typ parametru	Název	Iniciální hodnota	Význam
exposedField	location	0 0 0	umístnění vrcholu světelného kužele
	direction	0 0 –1	směr osy světelného kužele

Typ parametru	Název	Iniciální hodnota	Význam
	beamWidth	1,570796	kladný úhel (v rozsahu do Pi/2) mezi osou a površkou vnitřního kužele, v němž je plný světelný tok
	cutOffAngle	0,785398	kladný úhel (v rozsahu do Pi/2) mezi osou a površkou vnějšího kužele, v němž klesá světelný tok
	radius	100	nezáporný dosah osvětlení
	attenuation	1 0 0	trojice nezáporných koeficientů pro výpočet útlumu světla
	color	1 1 1	barva poprsků
	intensity	1	iniciální intenzita paprsků v rozsahu (0, 1)
	ambientIntensity	0	příspěvek zdroje k nepřímému osvětlení virtuálního světa v rozsahu (0, 1)
	on	TRUE	zapnutí či vypnutí světelného zdroje

Switch (Přepínač, volba)

Typ parametru	Název	Iniciální hodnota	Význam
exposedField	choice	[ ]	seznam potomků – možností k výběru
	whichchoice	-1	číslo zobrazovaného potomka; hodnota –1 znamená žádný výběr

Text (Nápis)

Typ parametru	**Název**	**Iniciální hodnota**	**Význam**
exposedField	fontStyle	NULL	styl písma v uzlu FontStyle
	string	[ ]	posloupnost textových řetezců
	maxExtent	0.0	nezáporná hodnota omezující maximální rozměr nápisu ve směru daném stylem písma: hodnota 0 ruší jakékoliv omezení
	length	[ ]	seznam nezáporných délek pro každý z řetězců; hodnota 0 znamená libovolnou délku

TextureCoordinate (Souřadnice textury)

Typ parametru	Název	Iniciální hodnota	Význam
exposedField	point	[ ]	seznam bodů v rovině

TimeSensor (Časovač)

Typ parametru	**Název**	**Iniciální hodnota**	**Význam**
exposedField	startTime	0	absolutní čas, po jehož dosažení zahájí časovač práci
	stopTime	0	absolutní čas, po jehož dosažení ukončí časovač práci
	cycleInterval	1	nezáporná délka časové smyčky; v sekundách
	loop	FALSE	povolení generování časových událostí v nekonečné smyčce
	enabled	TRUE	povolení práce časovače

Typ parametru	Název	Iniciální hodnota	Význam
eventOut	isActive		bylo zahájeno či ukončeno generování časových událostí časovačem
	time		plynule generovaný absolutní čas
	cycleTime		absolutní čas zahájení další časové smyčky
	fraction_changed		plynule generovaná poměrná hodnota uplynulého času v rámci jedné smyčky, tj. v rozsahu (0,1)

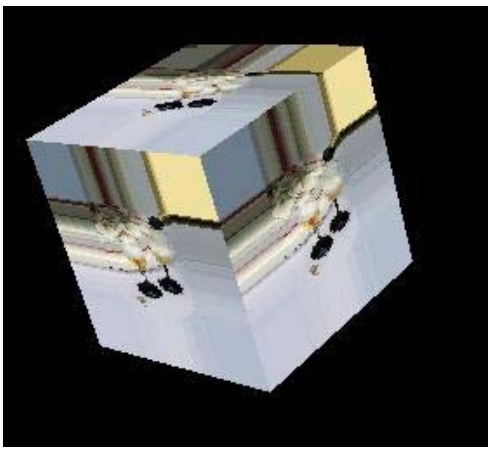
TouchSensor (Detektor dotyku)

Typ parametru	Název	Iniciální hodnota	Význam
exposedField	enabled	TRUE	povolení práce detektoru
eventOut	isOver		kurzor se dostal nad objekt nebo jej opustil
	isActive		stav tlačítka pro aktiaci kurzoru se změnil
	touchTime		čas uvolnění tlačítka pro aktivaci kurzoru
	hitPoint_changed		bod na povrchu objektu, na který ukazuje kurzor, se změnil
	hitNormal_changed		normála v bodě, který odpovídá parametru hitPoint_changed
	hitTexCoord_changed		souřadnice textury v bodě, který odpovídá parametru hit-Point_changed

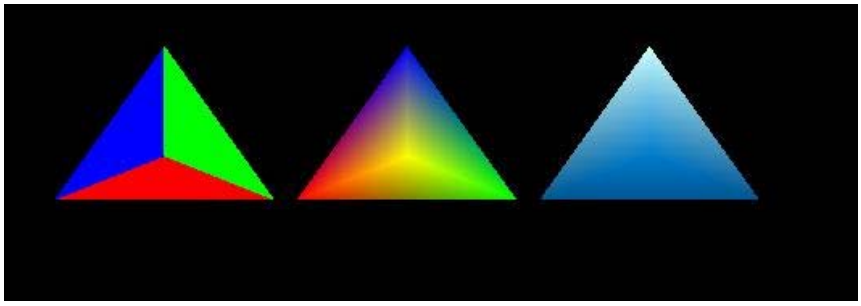
VisibilitySensor (Detektor viditelnosti)

Typ parametru	**Název**	**Iniciální hodnota**	**Význam**
exposedField	center	0 0 0	těžiště kvádru, jehož viditelnost je sledována
	size	0 0 0	nezáporné rozměry kvádru
	enabled	TRUE	povolení práce detektoru
eventOut	isActive		část kvádru se objevila na obrazovce, resp. kvádr přestal být viděn (zobrazován)
	enterTime		čas, ve kterém nabude parametr isActive hodnotu TRUE
	exitTime		čas, ve kterém nabude parametr isActive hodnotu FALSE

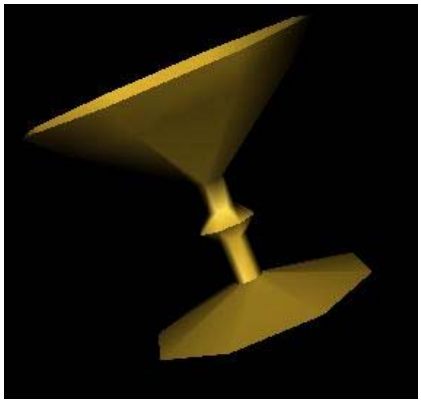
Barevná příloha



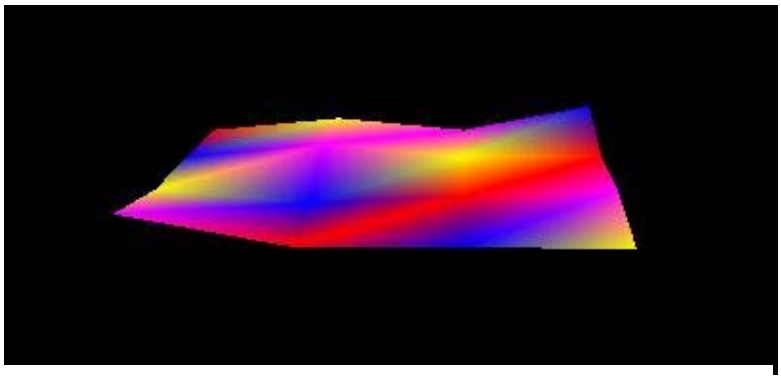
Obrázek 10: Použití transformace textury bez opakování



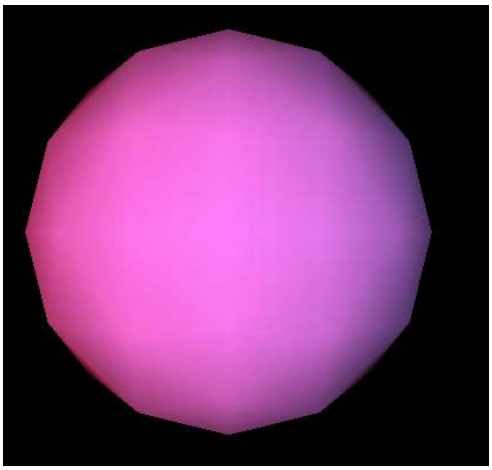
Obrázek 14: Rovinné trojúhelníky se stejnou geometrií, ale s různými parametry.
Zleva postupně plochy s jedinou barvou, plochy s barvami ve vrcholech, jednobarevné plochy s normálami ve vrcholech.



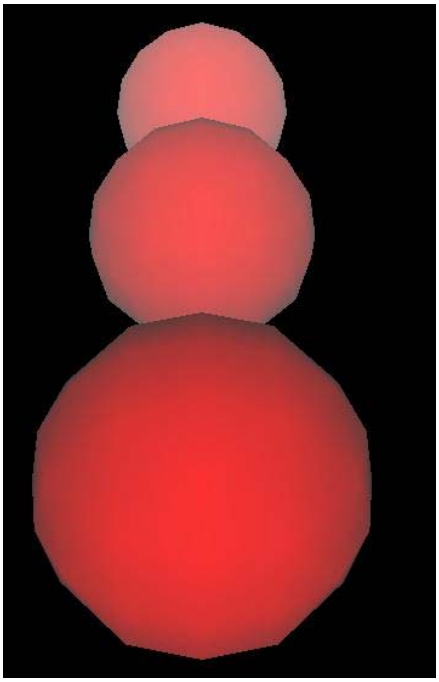
Obrázek 15: Zlatavý pohár definovaný uzavřeným profilem z osmi vrcholů a trajektorií o osmi bodech.



Obrázek 17: Barevná výšková mapa



Obrázek 21: Ukázka osvětlení pomocí uzlu DirectionalLight



Obrázek 24: Ukázka použití mlhy