ZÁKLADY 3D GRAFIKY

Scéna a její součásti

Modelovací nástroje

Běžné formáty pro popis scény

Zobrazení scény

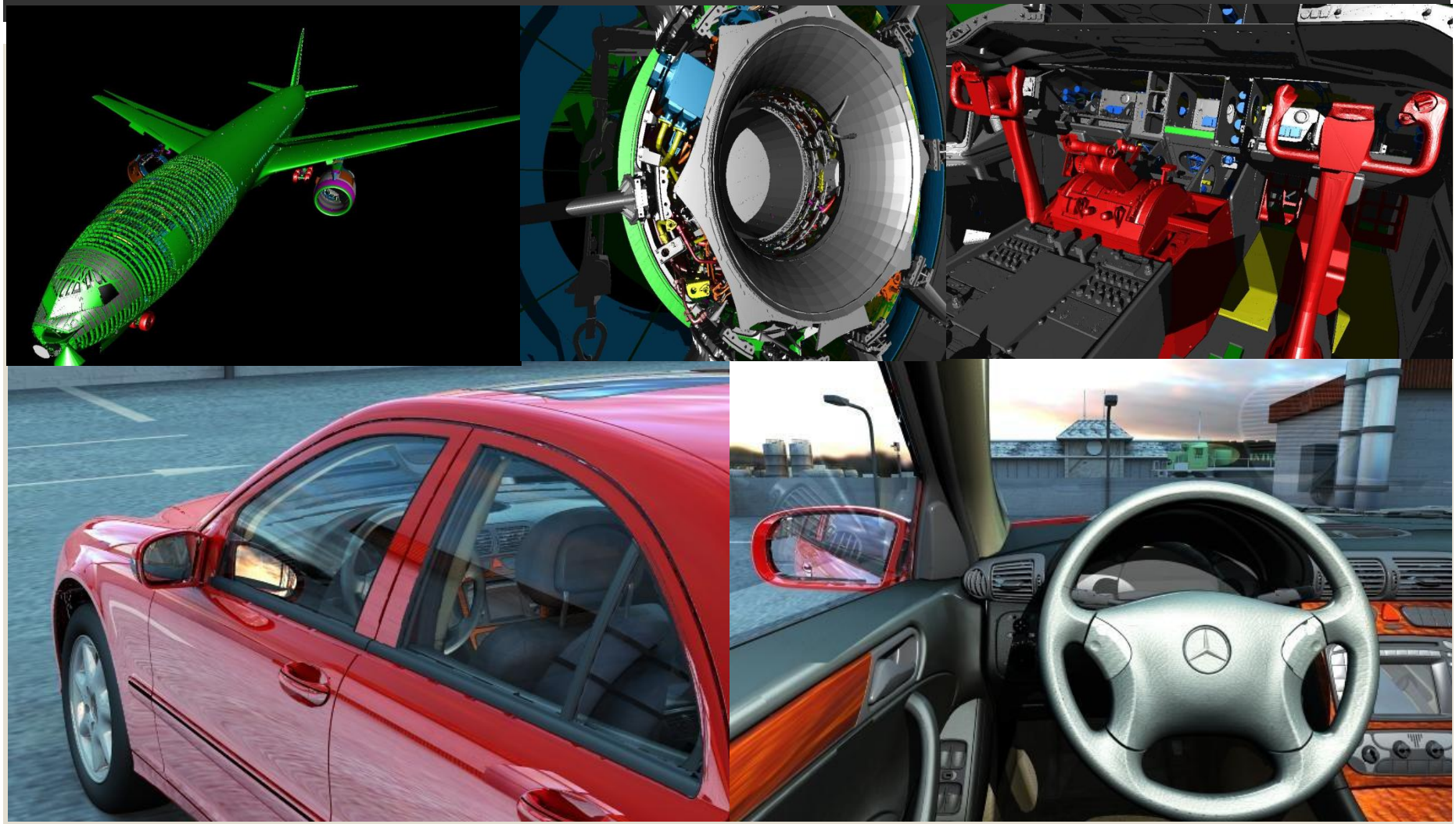
3D GRAFIKA

- Princip 3D grafiky:
  - Objekty modelovány vektorově ve 3D
  - Výsledek převedeme na 2D grafiku
    - Nejčastěji 2D bitmapová
- Oba procesy mohou být úzce provázány
  - Např. počítačové hry (XNA, DirectX, OpenGL)
- Oba procesy mohou být oddělené
  - Příprava 3D dat provedena v nástroji č. 1
  - Zobrazení (převod na 2D) proveden nástrojem č. 2
  - Např. filmový průmysl

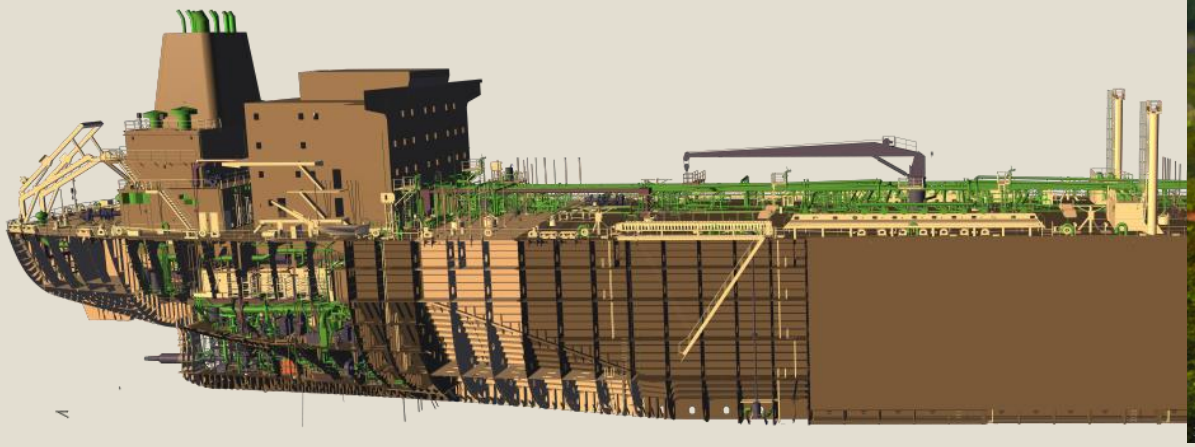
3D GRAFIKA

- Typické užití:
  - Počítačové hry a filmový průmysl
  - Augmented reality
  - Vizualizace terénu (např. Google Earth)
  - Vizualizace strojírenských součástek
  - Vizualizace architektonických budov
  - Vizualizace medicínských dat
  - …

3D GRAFIKA



3D GRAFIKA



3D GRAFIKA

PROGRAMOVÁNÍ 3D APLIKACÍ

- Různé přístupy dle výsledné aplikace:
  - Přístup č. 1 – programuji pouze rozhraní mezi modelovací a zobrazovací komponentou
  - Přístup č. 2 – provádím vytváření 3D objektů, zobrazováním se vůbec nezabývám
  - Přístup č. 3 – řeším zobrazování 3D objektů
- Přístup č. 1
  - Nejjednodušší: 3D data připraví někdo jiný, zobrazení provede „standardní" prohlížeč
  - Postačuje načíst data a převést do formátu akceptovatelného prohlížečem
  - Např. 3D galerie

PROGRAMOVÁNÍ 3D APLIKACÍ: PŘÍSTUP Č. 1

- Nejjednodušší: 3D data připraví někdo jiný, zobrazení provede „standardní“ prohlížeč
- Postačuje načíst data a převést do formátu akceptovatelného prohlížečem
- Např. 3D galerie

PROGRAMOVÁNÍ 3D APLIKACÍ: PŘÍSTUP Č. 2

- Připravím 3D objekty
  - Např. postava, automobil, strom, …
  - Např. šroub, matka, plech,
- Rozmístím je v prostoru
  - Typicky se využijí afinní transformace
    - Posun, rotace, změna měřítka
    - Významný rozdíl oproti 2D: při rotaci je třeba specifikovat osu, okolo které se bude otáčet
  - Transformace lze skládat
    - Např. střed kola automobilu se nachází v (1, 0.8, –0.5) vůči těžišti auta, auto umístěno v (10, 20, 0.8) vůči souřadnému systému světa (0, 0, 0)

PROGRAMOVÁNÍ 3D APLIKACÍ: PŘÍSTUP Č. 2

- 3D objekty v prostoru mohou být:
  - Nehybné (statické) – KIV/UPG
  - Interagovat mezi sebou – KIV/ZPG
- Výsledkem scéna
- Scéna zobrazena „standardním“ prohlížečem
- Např. 3D galerie, e-shop, realitky, kuchyňská studia …

SOUŘADNÝ SYSTÉM SCÉNY

- Kartézský souřadný systém
  - Natočení os bývá aplikačně závislé
  - Osa *z* někdy bývá vodorovná (optika), jinde vertikální, často rovněž ve směru od/do tradiční 2D roviny

Figure 1
2D Coordinate System

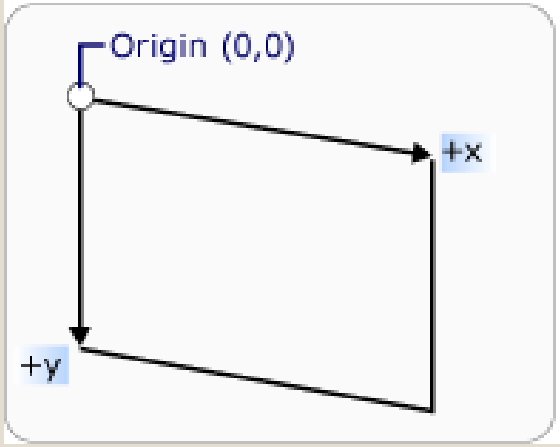
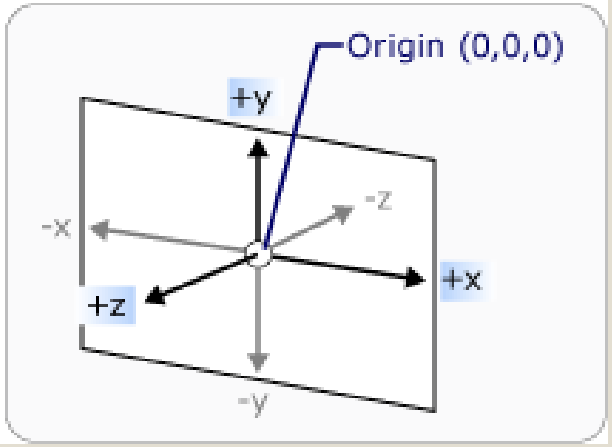
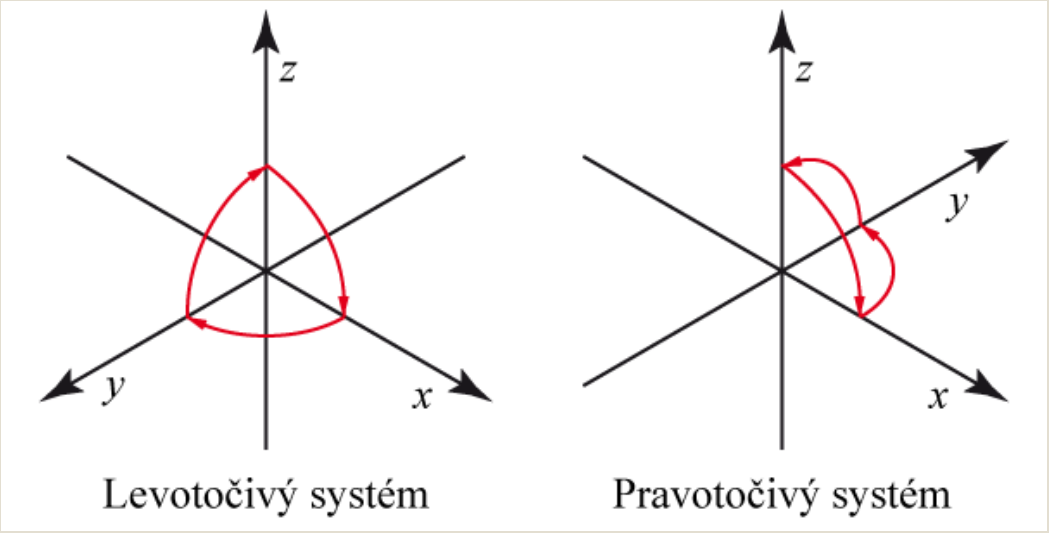


Figure 2
3D Coordinate System



SOUŘADNÝ SYSTÉM SCÉNY

- Točivost systému
  - Jakým směrem probíhá rotace o kladný úhel okolo os x, y, z
  - Levotočivý a pravotočivý systém
    - Nutno rozlišovat!



3D OBJEKTY

- Dva způsoby získání:
  - Databáze 3D objektů
  - Vlastní vytvoření 3D objektu
- Databáze 3D objektů
  - Existují jak volné tak komerční
    - Např. Ikea, Google Sketchup, …
  - Volné bývají hodně omezené
  - Komerční bývají často drahé (tisíce / objekt)
  - Nejjednodušší způsob
  - Často nejlevnější
    - A to i v případech, kdy objekt zakupujeme

3D OBJEKTY

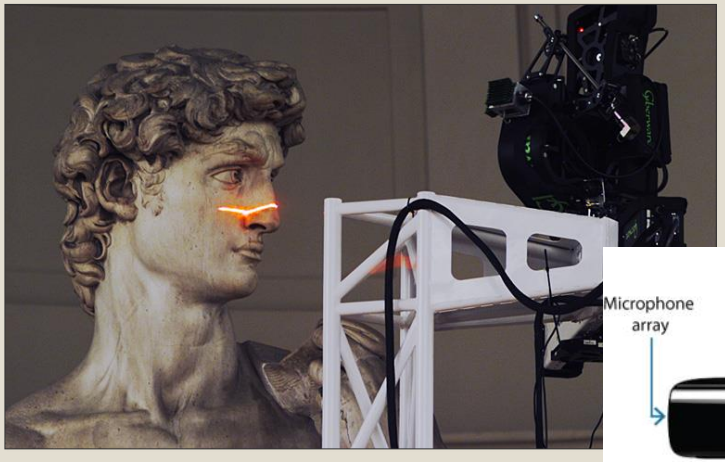
- Modely podléhají typicky licencím
- Nezřídka je ve 3D objektu skrytě watermark
- Vhodné např. v případech, kdy modelujeme:
  - Hrubou stavbu interiéru (např. bytové jádro) a pro dokreslení stavu chceme interiér osadit Ikea nábytkem
  - Návrh budovy a exteriér osadíme stromy, auty na parkovišti, postavami pro zvýšení realističnosti a zvýšení šance, že zaujmeme investora

3D OBJEKTY

- Vlastní vytvoření 3D modelu
  - Tvorba složitějších modelů časově náročná
    - Velice snadno více než 100 hodin práce
  - Základní principy:
    - 3D skenování
    - 3D modelování

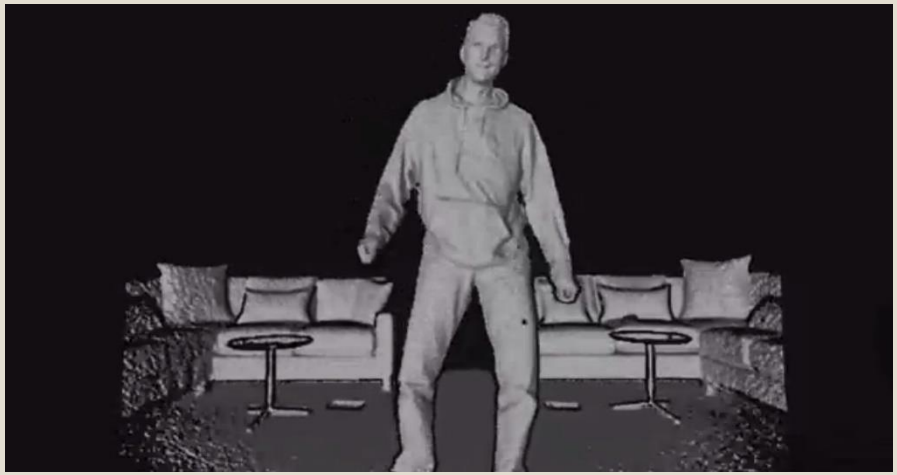
3D SKENOVÁNÍ

- Reálný objekt (resp. jeho fyzický model) oskenován 3D skenerem
  - Nejčastěji 3D laserový skener
- Alternativa: systém kamer resp. Kinnect



3D SKENOVÁNÍ

- Výsledkem zašuměná množina bodů ve 3D
- Data je nutné zpracovat:
  - Odšumění, odfiltrování outliers
  - Rekonstrukce povrchu → povrchová trojúhelníková síť
  - Čištění modelu
  - KIV/ZPOS, ZPG



3D MODELOVÁNÍ

- Lze použít dostupné modelovací nástroje
  - Nástroje technicky orientované (CAD)
    - Cílem přesná konstrukce v návaznosti na výrobu
  - Nástroje graficky orientované
    - Cílem snadné modelování zejména hladkých tvarů
- Objekty modelovány v těchto nástrojích:
  - Skládáním z jednodušších objektů
    - Analogie ke skládání regionů ve 2D
      - Booleovské operace: průnik, sjednocení, rozdíl, …
      - Základem kvádr, koule, kužel, válec, jehlan, torus, …

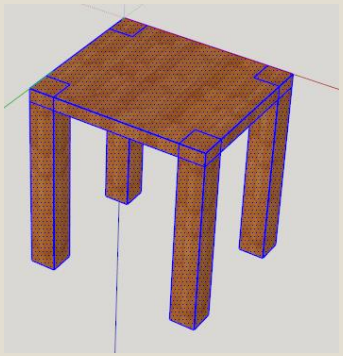
3D MODELOVÁNÍ

- Rotací 2D křivky okolo osy (tzv. resolve)
- Vytažením 2D křivky
  - Po úsečce (extrude) nebo obecné křivce (sweep)
- Manipulací s primitivy, ze kterých objekt vymodelován
  - Např. přemístění jednoho vrcholu nebo trojúhelníku

- Volně dostupné nástroje:
  - Google Sketchup, Blender, …
- Komerční:
  - Google Sketchup Pro, 3D Studio, AutoCAD, Maya, zBrush, ArchiCAD, …

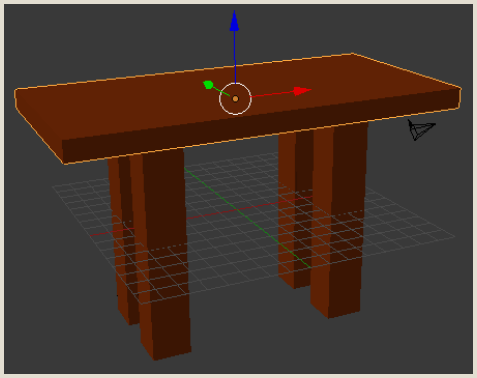
3D MODELOVÁNÍ

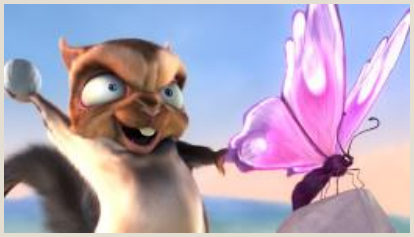
- Google Sketchup
  - http://www.sketchup.com
  - Intuitivní nástroj
  - Základem je 2D kreslení a manipulace s rovinnými plochami
  - Popis modelu uložen v textovém formátu
    - Úprava v jiné aplikaci



3D MODELOVÁNÍ

- Blender
  - http://www.blender.org/
  - Větší možnosti
    - Např. animace
  - Noční můra pro začátečníky
  - Modely objektů lze exportovat v různých formátech



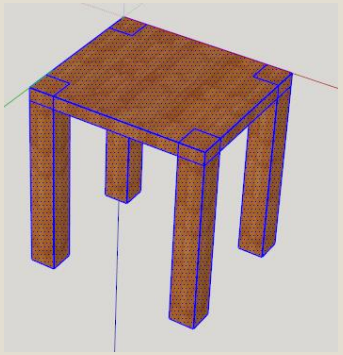


3D MODELOVÁNÍ

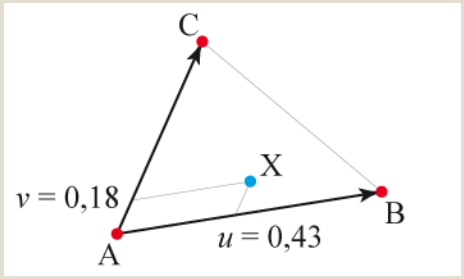
- Co když rozměry modelovaného objektu nejsou známé předem?
  - Parametry může určovat zákazník
    - Např. přeje si vyrobit sedačku širokou 4 m
  - Parametry může určovat dodavatel
    - Např. tloušťka desek, apod.
- Řešení: nutno použít vlastní aplikaci

3D MODELOVÁNÍ

- Jednoduché objekty:
  - Např. kvádr, koule, kužel, válec, jehlan, torus, …
  - Lze snadno vymodelovat
  - Lze skládat do o něco složitějších objektů
    - Např. židle = 4 kvádry (resp. válce) pro nohy + 1 kvádr pro sedátko + 1 kvádr pro opěrátko
  - Postačují pro většinu schematických vizualizací
    - Např. mobil, notebook, židle, stůl, stroje v továrně, …
  - Podpora prakticky ve všech nástrojích

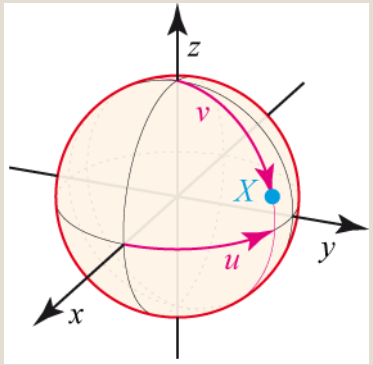


3D MODELOVÁNÍ



$$x = A_x + (B_x - A_x)u + (C_x - A_x)v$$
$$y = A_y + (B_y - A_y)u + (C_y - A_y)v$$
$$z = A_z + (B_z - A_z)u + (C_z - A_z)v$$

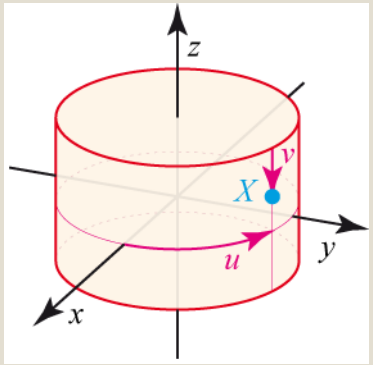
pro $0 \le u \le 1, 0 \le v \le 1, u + v \le 1$



$$x = r \cos u \sin v$$
$$y = r \sin u \sin v$$
$$z = r \cos v$$

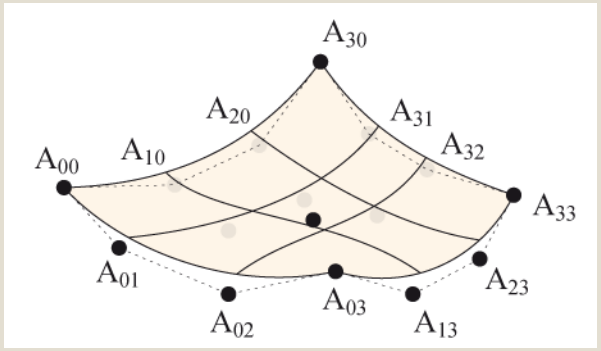
pro $0 \le u \le 2\pi, 0 \le v \le \pi$

3D MODELOVÁNÍ



$$x = r \cos u$$
$$y = r \sin u$$
$$z = v$$

pro $0 \le u \le 2\pi$, $0 \le v \le h$



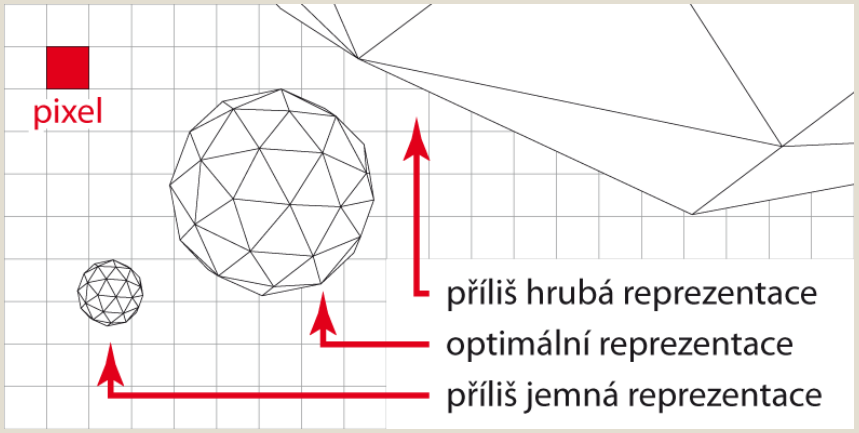
- Bézierův plát
  - Zobecnění Bézierovy křivky
  - Určen 4x4 vrcholy
  - Hladký povrch

3D MODELOVÁNÍ

- Složitější objekty:
  - Lze reprezentovat množinou primitiv (body, úsečky, trojúhelník, čtyřúhelník, …)
    - Nejčastěji používaný: trojúhelník
  - Mohou vzniknout převodem analyticky vyjádřených objektů (např. koule, válec, …)
  - Mohou vzniknout rotací 2D profilu kolem osy
    - Např. váza, láhev, hřebík, pohárek
    - Pro 2D profily typu lomená čára lze relativně snadno vymodelovat ve vlastní aplikaci
  - Modelování obecnějších modelů obtížné → KIV/APG, ZPOS, …

3D MODELOVÁNÍ

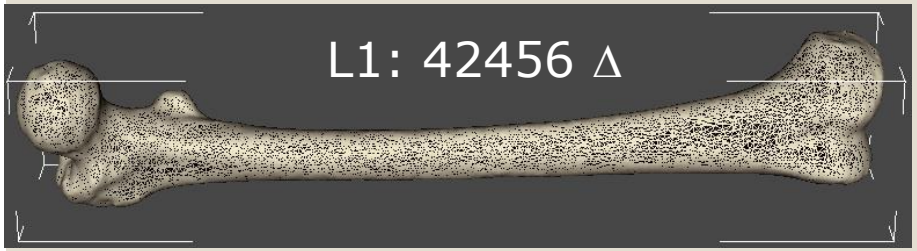
- Problém: kvalita na výstupu (obrazovce) je určena velikostí primitiv po jejich transformaci na 2D grafiku – tzv. promítání
  - Trojúhelník < 1 pixel => nelze rozpoznat rozdíl
  - Trojúhelník > mnoho pixelů => hrubý výsledek



3D MODELOVÁNÍ

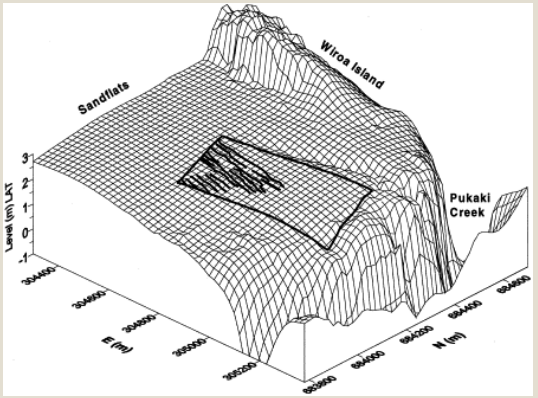
- Možné řešení: využívat analytického popisu objektů (koule, torus, válec, …)
  - Zobrazovací knihovna provede případný převod na množinu primitiv podle potřeby
  - Mnohé objekty nelze snadno vyjádřit analyticky
- Možné řešení: technika Level of Detail (LOD)
  - Více modelů pro tentýž objekt
  - Zobrazovací knihovna použije ten nejvhodnější
    - Typicky v závislosti na vzdálenosti od pozorovatele

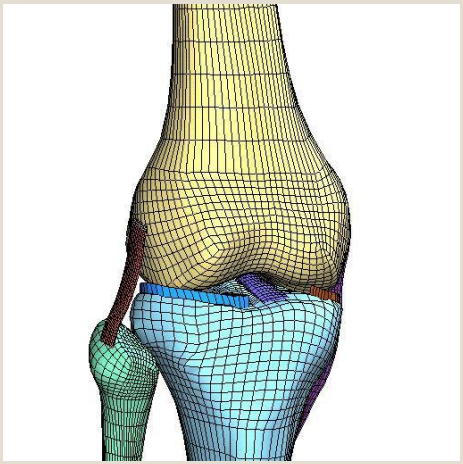
3D MODELOVÁNÍ



ČTYŘÚHELNÍKOVÉ SÍTĚ

- Přirozenější pro strojírenství, GIS, …
  - Body na pravidelné mřížce
- Vrcholy čtyřúhelníku nemusí ležet na rovině
- Pro zobrazení je čtyřúhelník automaticky rozdělen knihovnou na 2 trojúhelníky



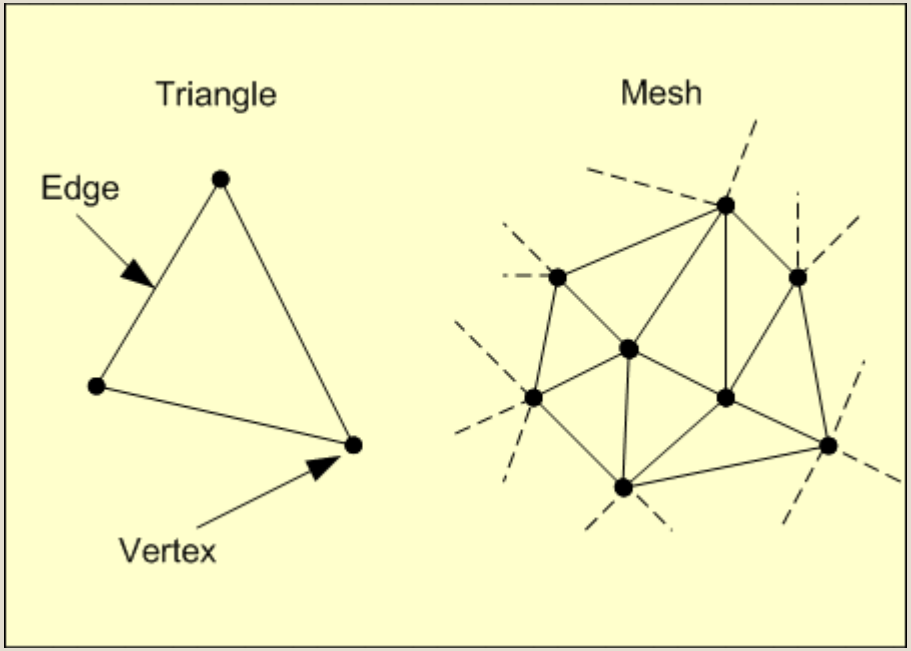


TROJÚHELNÍKOVÉ SÍTĚ

- Trojúhelník zadán třemi svými vrcholy
  - Může zdegenerovat na úsečku nebo bod
    - Nepřesnost reprezentace čísla v počítačích
    - Nepřesnost výpočtů
  - Nezdegenerovaný korektně reprezentuje rovinu
    - Na rozdíl od čtyřúhelníkových sítí
- Souřadnice udávány typicky ve floatech
  - Double pouze pro výpočty
  - Pomalejší a dvojnásobná velikost
  - GPU s double často neumí pracovat

TROJÚHELNÍKOVÉ SÍTĚ

- Trojúhelníková síť = množina trojúhelníků
- Nejčastěji specifikována dvojicí seznamů
  - Seznam vrcholů
  - Seznam trojúhelníků
- Seznam vrcholů
  - Souřadnice x, y, z
  - Bod sdílený více trojúhelníky uveden 1x
  - Reprezentováno obvykle jako jednorozměrné pole



TROJÚHELNÍKOVÉ SÍTĚ

- Seznam trojúhelníků
  - Trojice indexů do seznamu vrcholů
  - Reprezentováno nejčastěji jako jednorozměrné pole
  - Index 16-bitové nebo 32-bitové celé číslo
- Výhody reprezentace dvojicí seznamů:
  - Nemůže dojít z důvodu nepřesností k rozpadu sítě
  - Obvykle výrazná paměťová úspora
    - Euler-Poincaré charakteristika: V-E+F = konst
      - V = počet vrcholů, E = počet hran, F = počet stěn
    - Pro uzavřené (manifoldní) sítě bez děr: konst = 2
    - Důsledek pro trojúhelníkové sítě: poměr V:F:E = 1:2:3

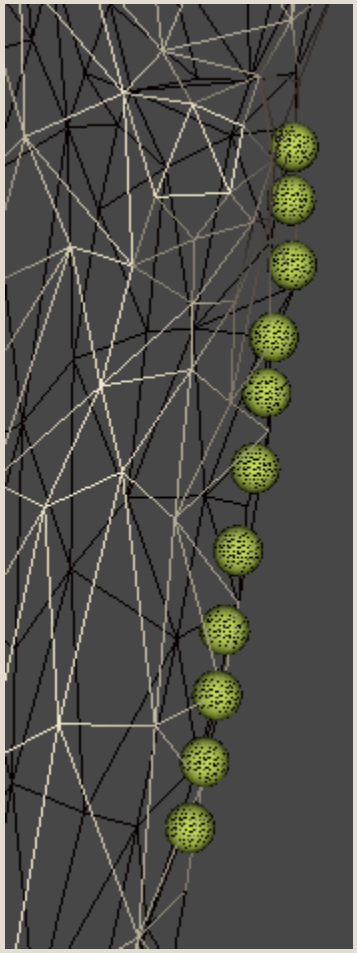
TROJÚHELNÍKOVÉ SÍTĚ

- Příklad:
  - 10 T jednoduchou reprezentací: 10*3*(3*4) = 360B
  - 10 T dvojicí seznamů: 10*3*2 + 5*(3*4) = 120B
    - Resp. 10*3*4 + 5*(3*4) = 180B
- V praxi mají trojúhelníkové sítě desítky až statisíce trojúhelníků

L0: 84912 Δ

IZOLOVANÉ BODY

- Např. pro označení významných míst na povrchu jiného objektu
- Lze modelovat buď jako
  - Trojúhelník či čtyřúhelník natáčející se k pozorovateli
    - Nemusí být vždy podporováno
  - 3D objekt (např. krychle, koule)
    - Krychle = 12 Δ!



IZOLOVANÉ HRANY

- Např. pro popisky umístěné v prostoru, znázornění souřadného systému
- Lze modelovat jako válec
  - Vzniká potenciální problém s 3D přístupem v důsledku příliš hubených Δ
  - Chybné vykreslení z důvodu numerických problémů

IZOLOVANÉ BODY A HRANY

- Výhoda 3D přístupu
  - Nemusím řešit, jak se bude scéna zobrazovat
  - Velikost bodu, resp. hrany proporcionální vůči vzdálenosti od pozorovatele
  - Body/hrany v zákrytu automaticky ignorovány
- Nevýhoda 3D přístupu
  - Body resp. hrany nemusí být vůbec vidět (< 1 px)
- Lze řešit jinak?

IZOLOVANÉ BODY A HRANY

- ANO: 2D přístup
  - 3D scéna zobrazena (bez izolovaných bodů a hran)
  - Přes výsledný obraz nakresleny na příslušných místech 2D objekty (např. čtvereček, kolečko, …)
- Výhoda 2D přístupu:
  - Všechny body a hrany jsou vidět
- Nevýhoda 2D přístupu:
  - NUTNO vědět něco o tom, jak se scéna zobrazuje
    - Jak se (x, y, z) transformuje na (x, y) obrazovky?
  - Všechny body a hrany jsou vidět
    - Jak zajistit, aby body v zákrytu nebyly vidět?
    - Jak odlišit vzdálenější body?

IZOLOVANÉ BODY A HRANY

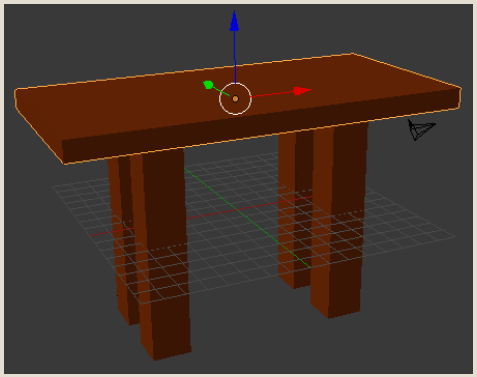
- 3D grafické knihovny obvykle podporují oba přístupy (pro body, hrany, text)
- 2D přístup nejčastěji používán pro legendu
  - Není umístěna v prostoru
  - Odpadá nutnost vědět, jak se 3D transformuje na 2D

PROGRAMOVÁNÍ 3D APLIKACÍ: PŘÍSTUP Č. 2

- Výslednou scénu uložím do některého z dostupných často používaných 3D formátů a nechám zobrazit „standardním“ prohlížečem

3D FORMÁTY

- Binární
  - Např. VTK, OBJ, X (DirecX), …
- Textové
  - Např. VTK, OBJ, STL, PLY, COLLADA (.dae), X, X3D, VRML, XAML, …



```
ply
format ascii 1.0
comment Created by Blender 2.71 (sub 0) - www.blender.org,
element vertex 24
property float x
property float y
property float z
property float nx
property float ny
property float nz
element face 6
property list uchar uint vertex_indices
end_header
10.000000 5.000000 5.500000 0.000000 0.000000 -1.000000
10.000000 -5.000000 5.500000 0.000000 0.000000 -1.000000
-10.000001 -4.999999 5.500000 0.000000 0.000000 -1.000000
-9.999996 5.000002 5.500000 0.000000 0.000000 -1.000000
10.000005 4.999997 6.500000 0.000000 -0.000000 1.000000
-9.999999 5.000000 6.500000 0.000000 -0.000000 1.000000
-10.000004 -4.999998 6.500000 0.000000 -0.000000 1.000000
9.999993 -5.000003 6.500000 0.000000 -0.000000 1.000000
10.000000 5.000000 5.500000 1.000000 -0.000001 0.000001
10.000005 4.999997 6.500000 1.000000 -0.000001 0.000001
```

```
solid Exported from
facet normal 0.0000
outer loop
vertex 10.000000 5.
vertex 10.000000 -5
vertex -10.000001 -
endloop
endfacet
facet normal -0.000000 0.000000 -1.000000
outer loop
vertex -10.000001 -4.999999 5.500000
vertex -9.999996 5.000002 5.500000
vertex 10.000000 5.000000 5.500000
endloop
```

3D FORMÁTY

- Různá složitost formátů
  - Některé pouze jeden objekt, jiné celé scény
  - Některé pouze statické scény, jiné dynamické
- Mnohé formáty požadují, aby součástí popisu scény byla informace, odkud a jak se na scénu pozorovatel (kamera) dívá

KAMERA

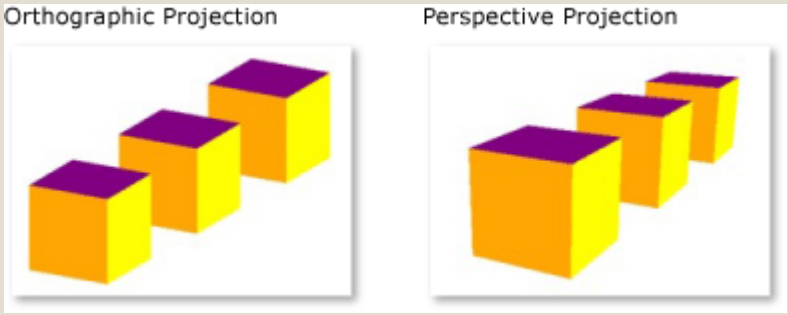
- Pozorovatel (kamera) umístěn v bodu E
- Pozorovatel se dívá do bodu C
  - C = střed výsledného obrázku
- Pro pozorovatele (kameru) nutno určit:
  - Vektor natočení vůči EC (Up)
    - Paralelní se svislou osou výsledného obrázku
    - Často bývá standardně totožné s osou y
      - Řešení: otočit celou scénou
  - Zorné pole (Field of View - FOVx a FOVy)

KAMERA

- Přední ořezová rovina (near clipping)
  - Bližší objekty jsou ignorovány
  - V mnoha 3D grafických systémech určuje bodu C
- Zadní ořezová rovina (far clipping)
  - Vzdálenější objekty jsou ignorovány
  - Vede k urychlení vykreslování složitých scén
  - Nemusí být nutné specifikovat vždy
- Typ kamery:
  - Ortogonální
  - Perspektivní

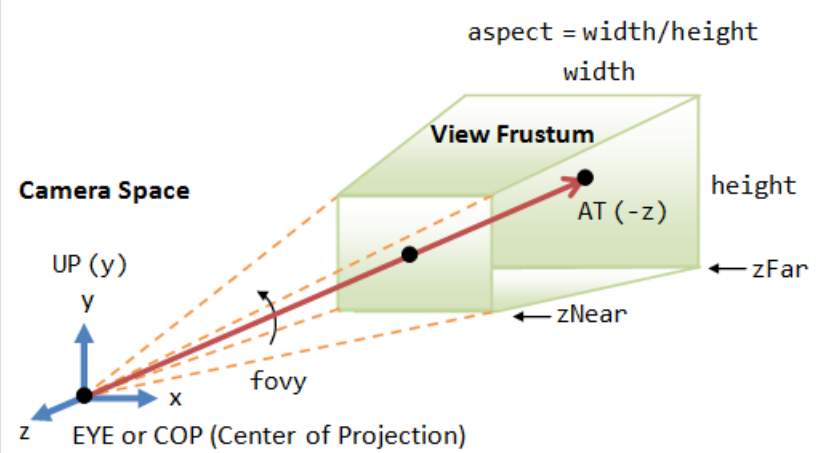
KAMERA

- Ortogonální kamera
  - Kolmé promítání (rovnoběžné paprsky)
  - Velikost zobrazení objektu nezávisí na jeho vzdálenosti od kamery
  - Vhodné pro technické zobrazování
- Perspektivní kamera
  - Paprsky se sbíhají
  - Velikost zobrazení objektu závisí na jeho vzdálenosti od kamery
  - Napodobuje lidské vnímání světa



KAMERA

- Zobrazovaný prostor (view frustum) je:
  - Kvádr pro ortogonální kameru
  - Komolý jehlan pro perspektivní kameru



Perspective Projection: The camera's view frustum is specified via 4 view parameters: fovy, aspect, zNear and zFar.

3D FORMÁTY

- VRML
  - Starší standardní formát
  - Prohlížení VRML souborů není podporováno nativně OS
  - V současnosti vytlačován X3D

```
#VRML V1.0 ascii
Separator {
    PerspectiveCamera { # nastavení pozorovatele (kamery)
        position        -8.6 2.1 5.6
        orientation     -0.1352 -0.9831 -0.1233  1.1417
        focalDistance   10.84
    }

…
```

3D FORMÁTY

- X3D
  - Založen na XML
  - Prohlížení VRML souborů není podporováno nativně OS

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE X3D SYSTEM "http://www.web3d.org/specifications/x3d-3.0.dtd" PUBLIC
"ISO//Web3D//DTD X3D 3.0//EN">

<X3D xsd:noNamespaceSchemaLocation="http://www.web3d.org/specifications/x3d-3.0.xsd"
xmlns:xsd="http://www.w3.org/2001/XMLSchema-instance" version="3.0" profile="Immersive">

<head>…</head>

<Scene>
  <Viewpoint position="0 -4 0" orientation="1 0 0 1.57"
        description="Extrusion Heart"/>
  <!-- popis scény -->
</Scene>
</X3D>
```

3D FORMÁTY

- XAML
  - Nativně lze prohlédnout na Windows Vista+
  - Na jiných platformách jsme v háji

```
<!-- Viewport3D is the rendering surface. -->
<Viewport3D Name="myViewport" >

  <!-- Add a camera. -->
  <Viewport3D.Camera>
    <PerspectiveCamera Position="0,0,-3" LookDirection="0,0,1"
                       UpDirection="0,1,0" FieldOfView="45"
                       FarPlaneDistance="20" NearPlaneDistance="1"/>
  </Viewport3D.Camera>

  <!-- Add models. -->
  <Viewport3D.Children>
        …
```

X3D OBJEKT

```
<Scene>
  <Shape>
    <IndexedFaceSet coordIndex="0 1 2">
      <Coordinate point="0 0 0 1 0 0 0.5 1 0"/>
    </IndexedFaceSet>
  </Shape>
</Scene>
```

XAML OBJEKT

```
<Viewport3D.Children>
 <ModelVisual3D>
  <ModelVisual3D.Content>
   <Model3DGroup>
     <GeometryModel3D>
       <GeometryModel3D.Geometry>
        <MeshGeometry3D
            Positions="0 0 0 1 0 0 0.5 1 0"
            TriangleIndices="0 2 1"/>
       </GeometryModel3D.Geometry>
      </GeometryModel3D>
   </Model3DGroup>
  </ModelVisual3D.Content>
 </ModelVisual3D>
</Viewport3D.Children>
```

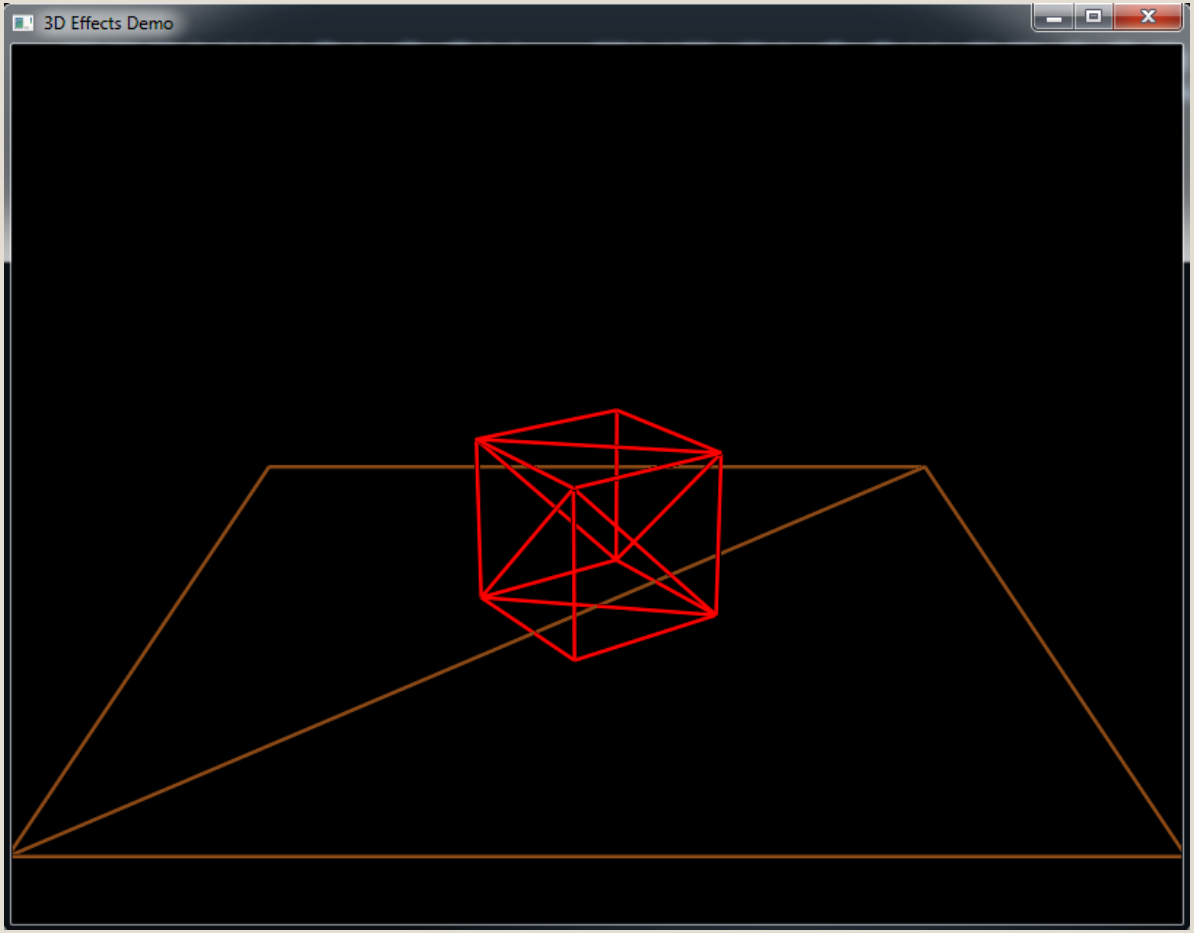
PROGRAMOVÁNÍ 3D APLIKACÍ: PŘÍSTUP Č. 3

- Řeším, jak bude scéna zobrazena
- Nejčastějšími důvody:
  - Zvýšení realističnosti výstupu na obrazovce
  - Zvýšení přehlednosti zobrazované scény
  - Zvýšení výkonu
  - Dynamická scéna
- Možná řešení:
  - Úprava nebo doplnění popisu scény
  - Vlastní zobrazení s použitím nějaké 3D knihovny (např. OpenGL, DirectX, XNA) ➔ KIV/ZPG, KIV/PH
  - Vlastní zobrazení scény ➔ KIV/ZPG, KIV/APG, KIV/GRG
- Např. ilustrace, montážní postupy, fotorealistické zobrazování

ZOBRAZENÍ SCÉNY

- Různé zobrazovací techniky
  - Různé výhody i nevýhody
- Drátěný model (wireframe)
  - Vypočte 2D pozice vrcholů
  - Spojí 2D vrcholy 2D úsečkami
  - Nejjednodušší (a nejrychlejší)
  - Dívám se skrz objekty
  - Výhoda:
    - Lze vidět (manipulovat) zakryté objekty
  - Nevýhoda:
    - Nepřehledné pro větší počet hran

ZOBRAZENÍ SCÉNY

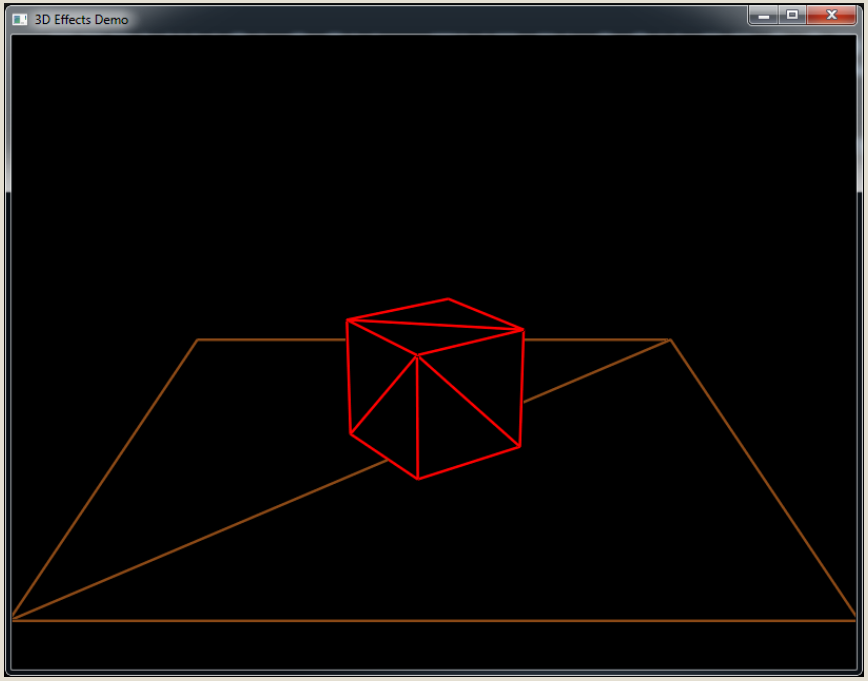


X3D A XAML

- Drátěný model není nativně v prohlížečích těchto formátů podporován
  - Chybí příslušný konfigurační element
- Řešení:
  - Objekt musí být vymodelován jako drátěný
  - Drát je válec o určitém průměru

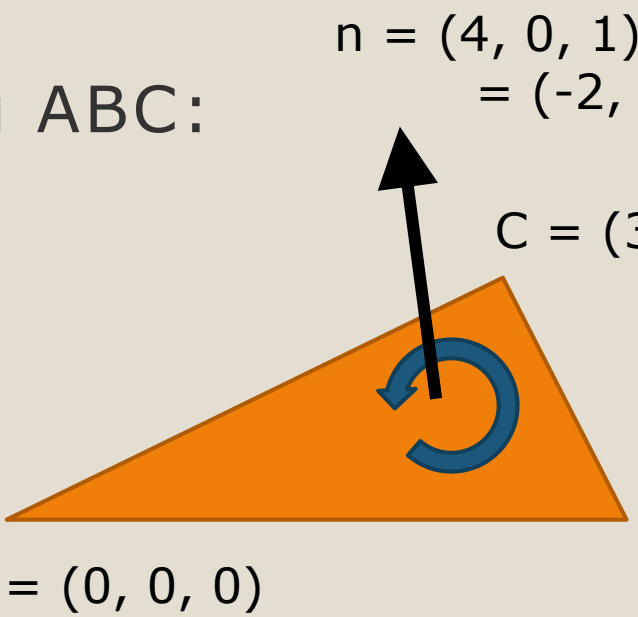
ZOBRAZENÍ SCÉNY

- Drátěný model s uvažováním viditelnosti
  - Hrany odvrácených trojúhelníků nejsou vykreslovány
  - Zakryté hrany nejsou vykreslovány
  - Algoritmy – viz ZPG
  - Výhody:
    - Dostatečně rychlé
    - Dává představu o tom, jak scéna vymodelována
  - „Nevýhoda“:
    - Vyžaduje znalost normály



NORMÁLA

- Normála (normálový vektor) = vektor kolmý v daném bodě k povrchu
  - Jedná se o gradient
- Normála trojúhelníku ABC:
  - n = (B – A) × (C – A)



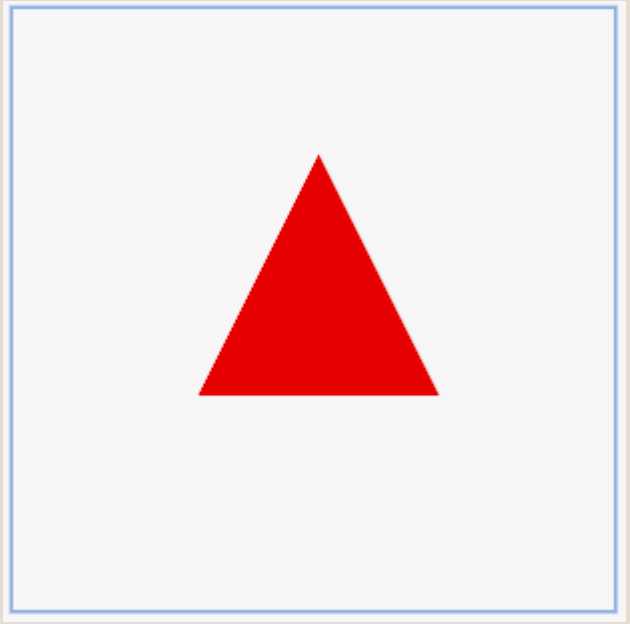
NORMÁLA

- Vrcholy trojúhelníků musí být zadávány v konzistentním pořadí
  - Po směru hodinových ručiček (CW)
  - Proti směru hodinových ručiček (CCW)
    - Normála směřuje ven z objektu
    - Častější, výhodnější
- Pro trojúhelníkové sítě si 3D grafické knihovny normálu vypočítají obvykle samy

NORMÁLA

- Příklad:

```
<GeometryModel3D.Geometry>
    <MeshGeometry3D Positions="0 0 0 1 0 0 0.5 1 0"
        TriangleIndices="0 1 2"
/>
</GeometryModel3D.Geometry>
```

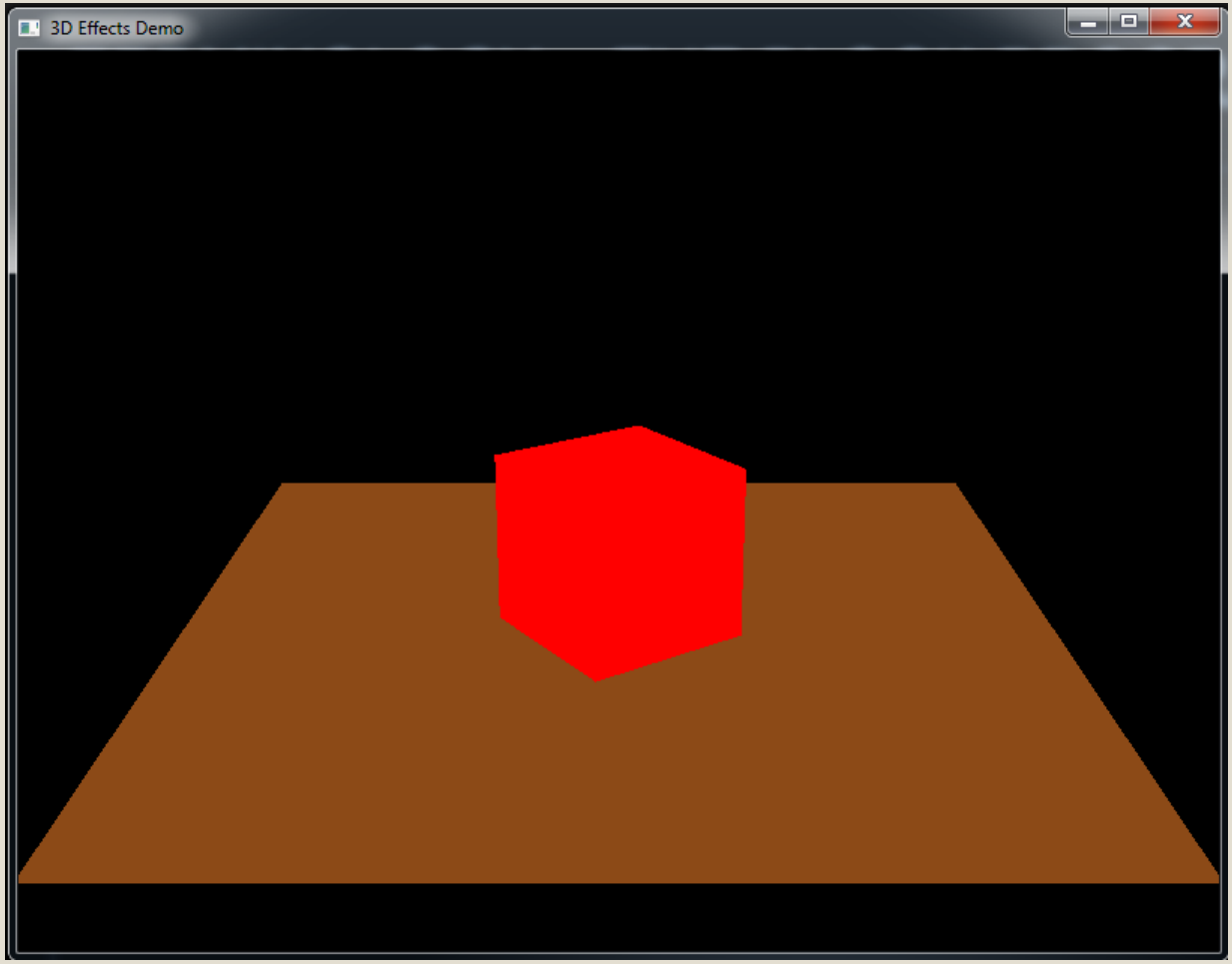


```
<GeometryModel3D.Geometry>
    <MeshGeometry3D Positions="0 0 0 1 0 0 0.5 1 0"
        TriangleIndices="0 2 1"
/>
</GeometryModel3D.Geometry>
```

ZOBRAZENÍ SCÉNY

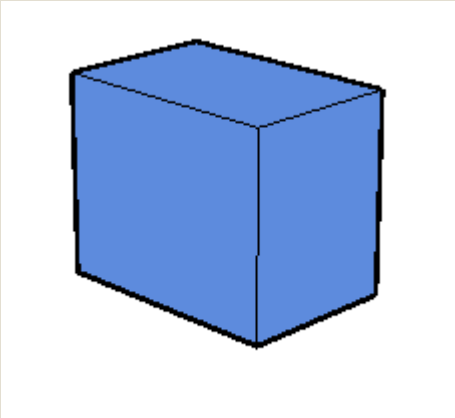
- Jednobarevné plošné zobrazení
  - Řeší rovněž problém viditelnosti
    - Vyžaduje znalost normál
  - 2D obrazce se vyplní jednou barvou
    - Barva stejná pro celý objekt
    - Vyžaduje specifikaci barvy objektu
  - Výhody:
    - Velmi rychlé (i pro velké sítě)
    - Vhodné např. pro schématická CAD zobrazení
  - Nevýhoda:
    - Realističnost nízká

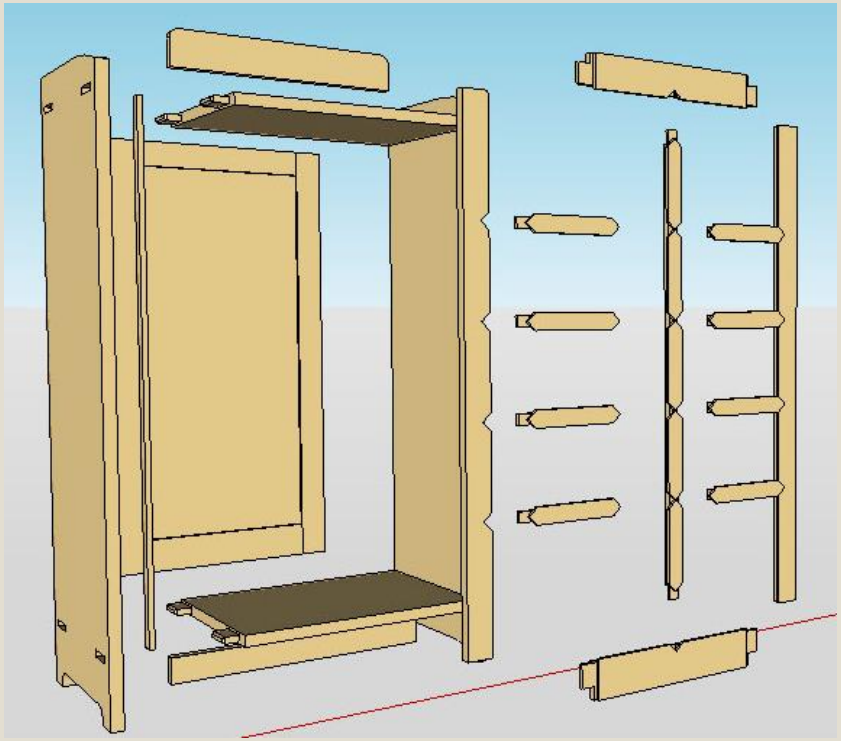
ZOBRAZENÍ SCÉNY



ZOBRAZENÍ SCÉNY

- Využívá se typicky v kombinaci se zobrazením obrysových hran (viz drátěný model)

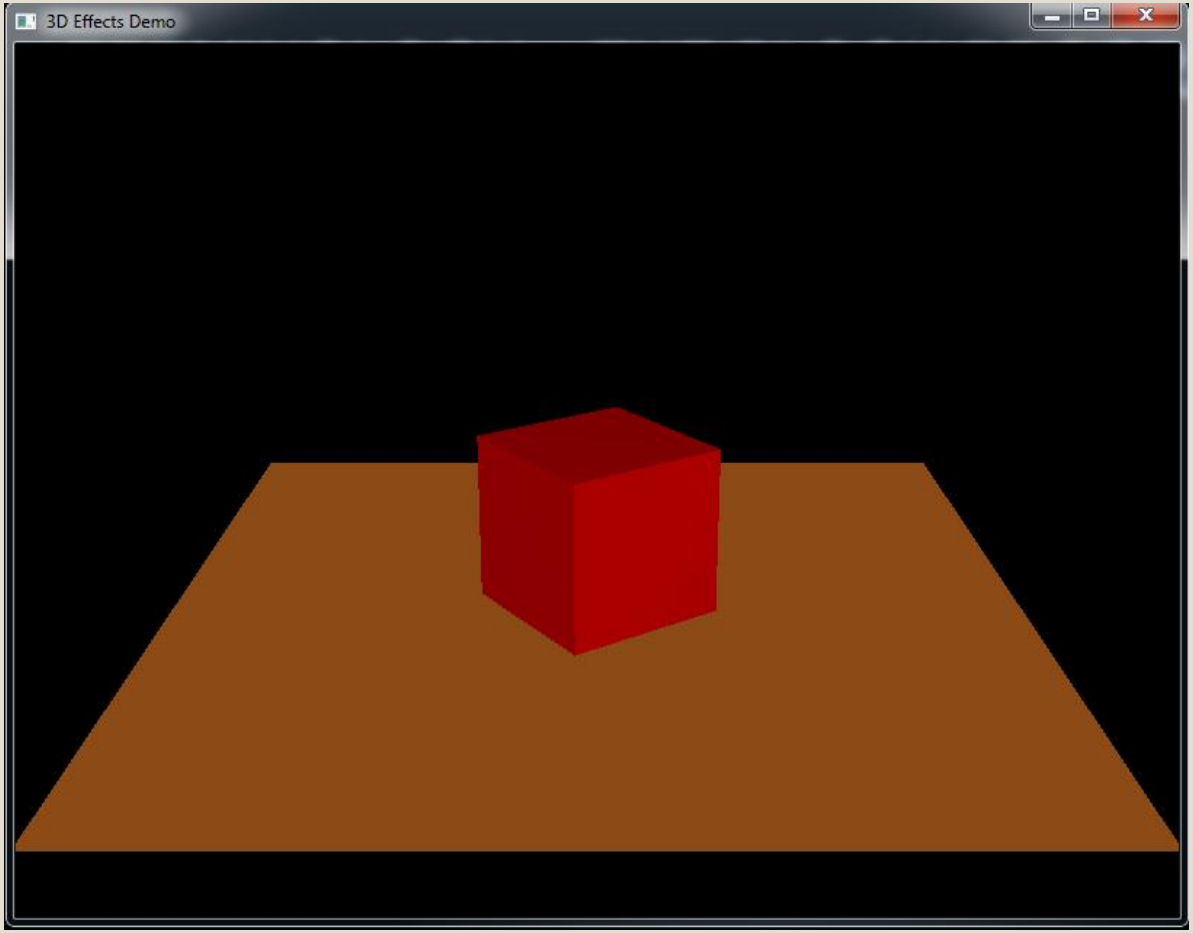




ZOBRAZENÍ SCÉNY

- Konstantní stínování
  - Obdoba jako jednobarevné plošné zobrazení
  - Jas trojúhelníků závislý na dopadajícím světle
    - Vyžaduje dodefinování světla
    - Vyžaduje dodefinování vlastností materiálů objektů
  - Výhody:
    - Velmi rychlé (i pro velké sítě)
    - Vhodné např. pro náhledy na tvary objektů
  - Nevýhoda:
    - Skoková změna jasu

ZOBRAZENÍ SCÉNY



SVĚTLA

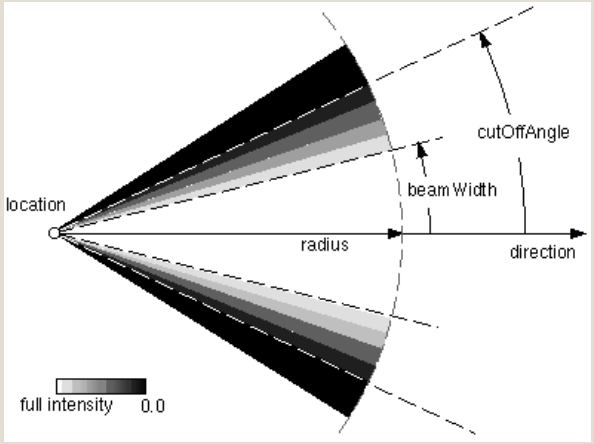
- Klíčové pro složitější zobrazovací přístupy
  - Přístupy označované jako stínování
- Pro scénu je nezbytné nadefinovat jeden nebo více zdrojů světla
  - Časová složitost vykreslování roste lineárně s počtem
  - Často počet podporovaných světel omezen
    - Nutno přepínat světla dle potřeby
- Pro každé světlo se specifikuje jeho barva
  - Barevnost a jas
  - Obvykle se specifikuje v RGB

SVĚTLA

- Modely zdrojů světla:
  - Rovnoběžné
  - Bodové
  - Kuželové
  - Ambientní (rozptýlené)
- Rovnoběžné světlo
  - Definováno směrem
  - Paprsky dopadají do scény rovnoběžně
  - Vhodné pro:
    - Exteriérové scény
    - Interiérové scény s velkou prosklenou stěnou

SVĚTLA

- Bodové světlo
  - Definováno svou pozicí ve scéně
  - Paprsky se šíří všemi směrem od pozice
  - Jas automaticky klesá se vzdáleností plochy od zdroje
  - Vhodné pro:
    - Lampy, lustry, … v interiérových scénách
- Kuželové světlo
  - Reflektor (Spotlight)
  - Paprsky se šíří ve směru kužele
  - Definováno pozicí, směrem šíření, úhlem šíření a velikostí útlumu



SVĚTLA

- Vhodné pro zvýraznění některých částí scény
  - Např. baterka
- Ambientní (rozptýlené)
  - Definováno pouze barvou
  - Přichází ze všech směrů
  - Bez rozptýleného světla jsou části scény, kam nedopadne světlo z ostatních světelných zdrojů, černé

MATERIÁLY OBJEKTŮ

- Určují jakým způsobem objekt světlo odráží
- Nejčastěji definováno trojicí parametrů pro:
  - Matný odraz
  - Lesklý odraz
  - Rozptýlený odraz
- Matný (difúzní) odraz
  - Část dopadajícího světla se pohltí
  - Část dopadajícího světla se odrazí ve všech směrech
    - Z místa se stává bodový zdroj světla
    - Určeno koeficientem z rozsahu 0-1

MATERIÁLY OBJEKTŮ

- Často definováno pro každou barevnou složku zvlášť
  - Např. (0.7, 0, 0.5) osvícené bílým světlem

(1, 1, 1)

odrazí „fialové“ světlo

(0.7, 0, 0.5)

MATERIÁLY OBJEKTŮ

- Lesklý (spekulární, zrcadlový) odraz
  - Část dopadajícího světla se pohltí
  - Část dopadajícího světla se odrazí dle zákona odrazu
    - Z místa se stává kuželový zdroj světla
    - Úhel kužele určen parametrem „ostrost odlesku“
      - Dokonalé zrcadlo – úhel = 0°
      - V praxi několik málo stupňů
    - Určeno koeficientem z rozsahu 0-1
    - Může být definováno pro každou barevnou složku zvlášť
      - Stejné složky = lesklé plasty
      - Stejné koeficienty jako pro difúzní odraz = kovy

MATERIÁLY OBJEKTŮ

- Rozptýlený odraz
  - Obdoba difúzního odrazu
  - Pracuje pouze s rozptýleným (ambientním) světlem

X3D

```
<SpotLight beamWidth='0.1745' color='0.8 0.8
0.2' cutOffAngle='0.7837' radius='10' global='true'/>
<Viewpoint … >
  <Shape>
    <Box/>
    <Appearance>
     <Material diffuseColor='0.8 0.4 0.2‘
       specularColor=' … '
       emissiveColor=' … '
      />
    </Appearance>
  </Shape>
```

XAML

```
<Model3DGroup>
 <DirectionalLight Color="White" Direction="1,1,3" />
 <GeometryModel3D>
  <GeometryModel3D.Geometry>
       …
  </GeometryModel3D.Geometry>
 <GeometryModel3D.Material>
  <MaterialGroup>
   <DiffuseMaterial Brush="Red"/>
   <SpecularMaterial Brush="White" SpecularPower="8"/>
   <EmissiveMaterial Brush="Blue"/>
  </MaterialGroup>
 </GeometryModel3D.Material>
 <GeometryModel3D.BackMaterial>
       …
</GeometryModel3D.BackMaterial>
```

ZOBRAZENÍ SCÉNY

- Konstantní stínování

ZOBRAZENÍ SCÉNY

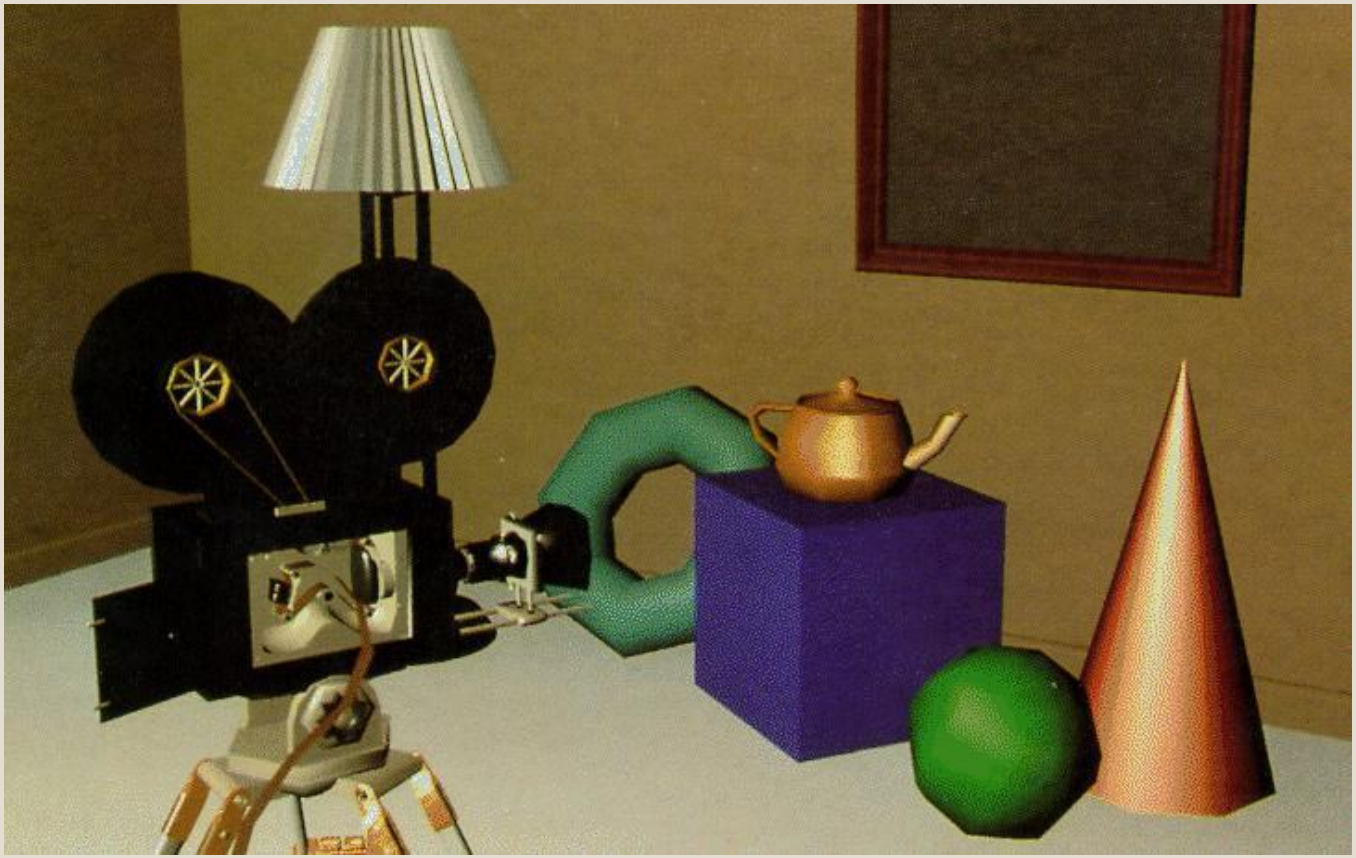
- Gouraud [Guró] stínování
  - Označováno též jako smooth shading
  - Jas barvy se na ploše 2D trojúhelníku mění v závislosti na dopadajícím světle
    - Vyžaduje specifikaci normál ve vrcholech sítě!
  - Výhody:
    - Vhodné pro „realistické“ vykreslení scény obsahující hladké povrchy (např. koule, kužel, …)
    - Implementováno nativně v GPU hardware
  - Nevýhoda:
    - Realističnost závisí na velikosti trojúhelníků

ZOBRAZENÍ SCÉNY

- Gouraud bez lesklých odrazů

ZOBRAZENÍ SCÉNY

- Gouraud s lesklými odrazy

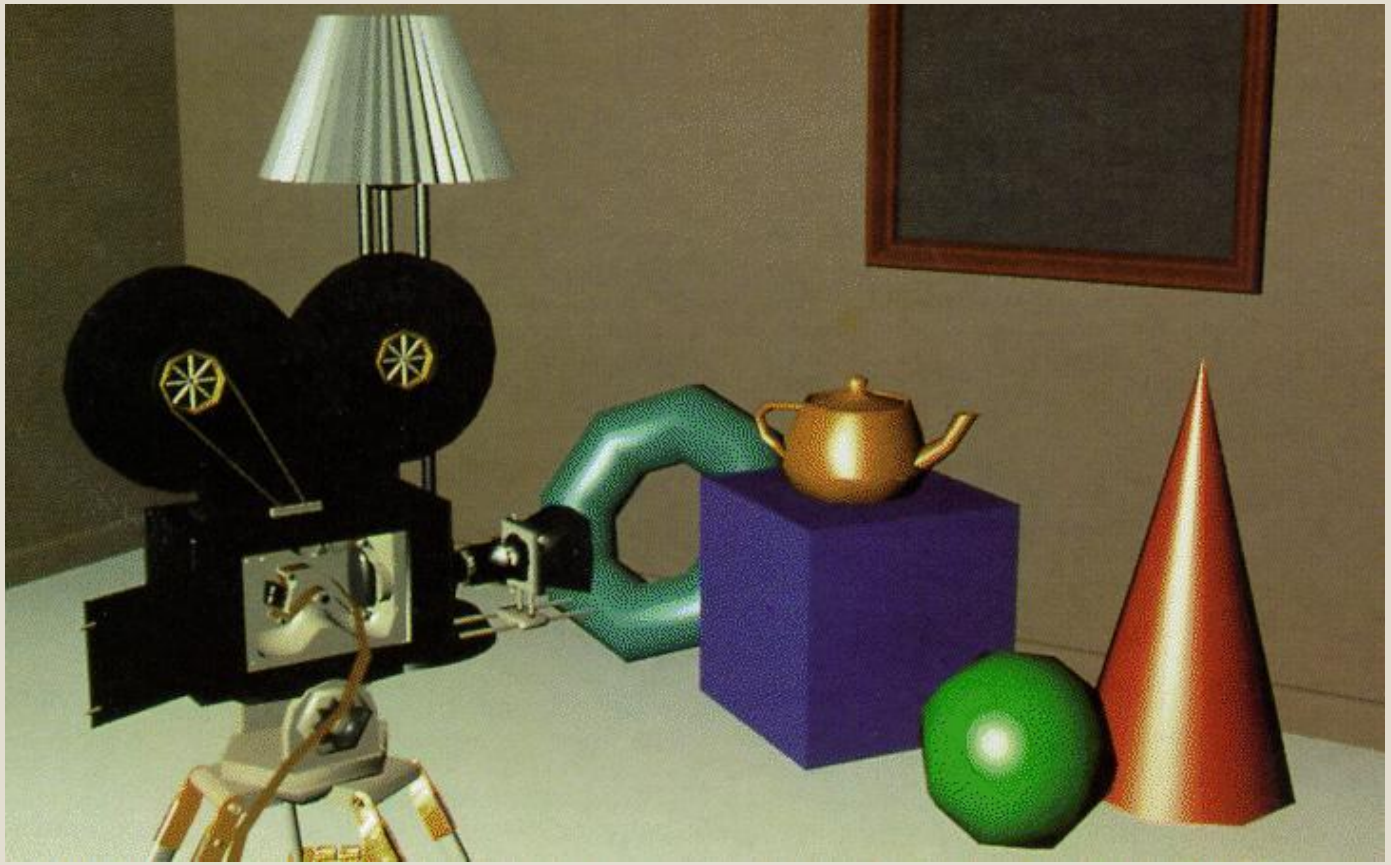


ZOBRAZENÍ SCÉNY

- Sofistikovanější stínování
  - Např. Phongovo stínování
  - Náročnost výpočtů na pozadí
  - Není standardně implementováno
  - Realističtější výsledky

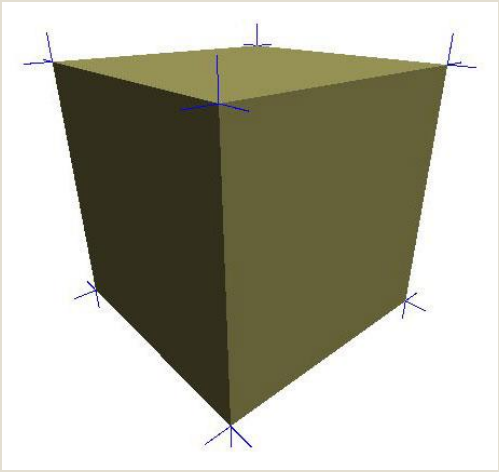
ZOBRAZENÍ SCÉNY

- Phongovo stínování



NORMÁLY VE VRCHOLECH Δ

- Pro hladké povrchy lze spočítat:
  - Analyticky (je-li povrch popsán analyticky)
  - Zprůměrováním normál okolních Δ
- Ve vrcholech ostrých (skutečných) hran objektu dvě nebo více normál ve směrech normál stykových ploch



X3D A XAML

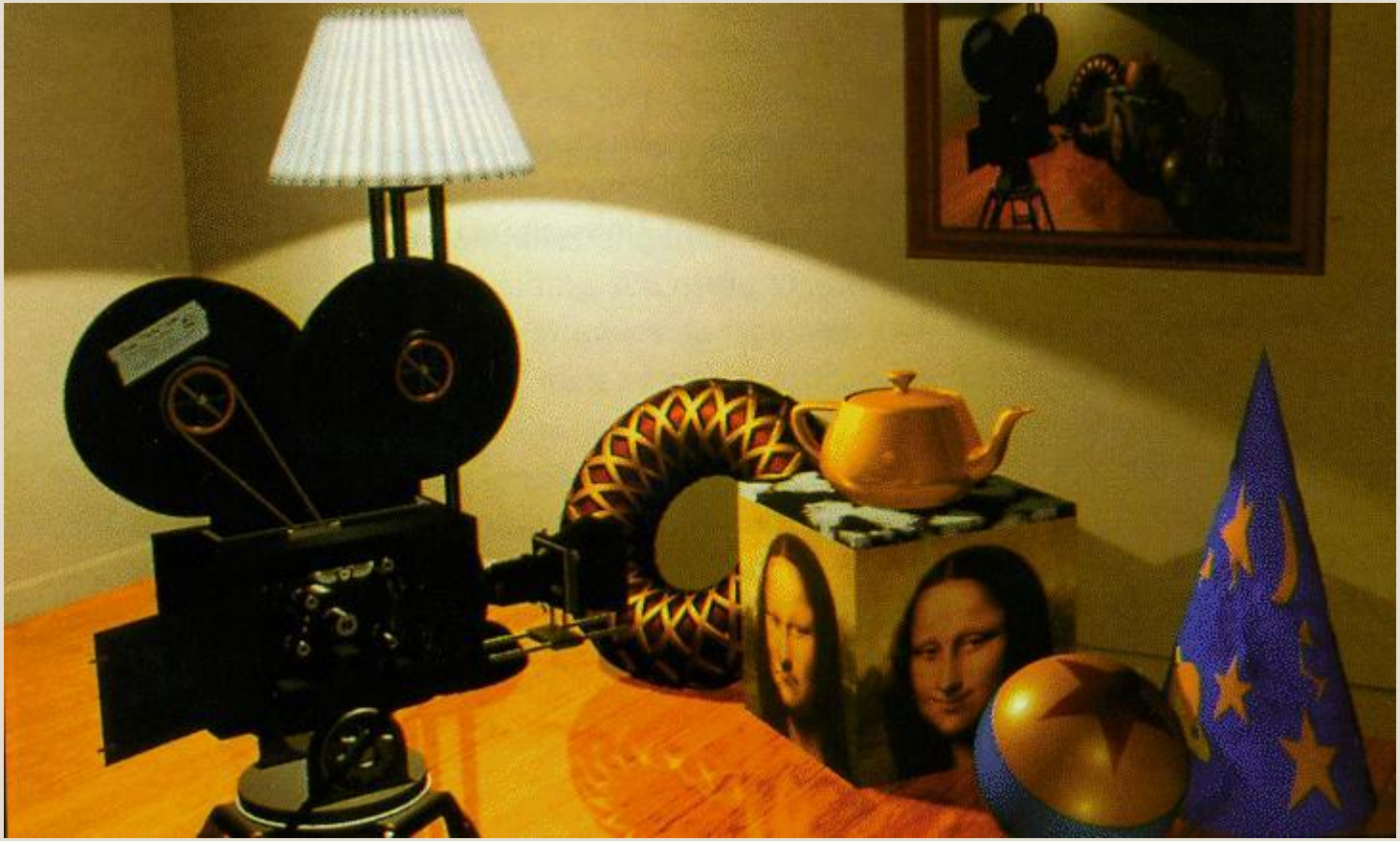
- <Shape>
  <IndexedFaceSet coordIndex=" … ">
  <Coordinate point="…"/>
  <Normal vector=" …"/>
  </IndexedFaceSet>

  - Pro předdefinované X3D objekty (box, sphere, …) je normála specifikována interně
- <MeshGeometry3D
  Positions=" … "
  Normals=" … "

„FOTO“ REALISTICKÉ MODELOVÁNÍ

- Možnost č. 1:
  - Reálné objekty modelovány velmi jemnou sítí
    - Promítnuté (transformované) trojúhelníky = cca 1 pixel
  - Každý trojúhelník má přiřazenu jednu barvu
    - Barva se modifikuje osvětlením
  - Vysoce neefektivní
- Možnost č. 2:
  - Parametry difúzního odrazu v libovolném bodě trojúhelníku uloženy ve 2D barevné textuře
  - Velmi rychlé a rozšířené

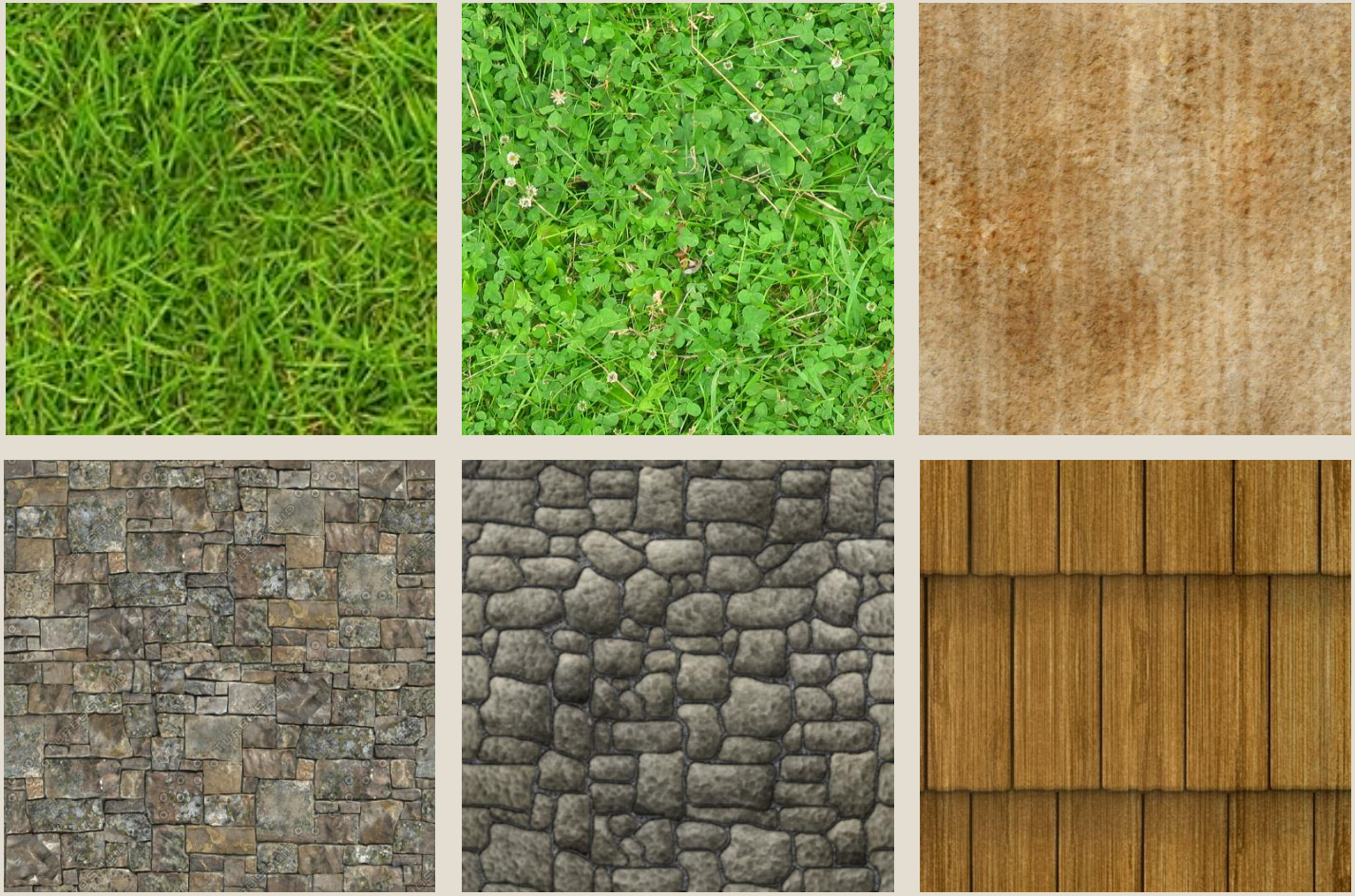
„FOTO“ REALISTICKÉ MODELOVÁNÍ



TEXTURA

- Mnoho významů a možností použití
  - Viz KIV/ZPG
- Pro naše účely: textura = bitmapový obrázek
- Typicky čtvercové o hraně velikosti mocniny 2
  - Např. 128x128, 256x256, 512x512
- Obsahem může být:
  - Uměle vygenerovaný obrázek
    - Např. mraky, mramor, odraz scény v zrcadle, …
  - Reálná fotografie

TEXTURA



TEXTURA

- Obrázek libovolné velikosti popsán dvojicí reálných souřadnic t, u (někdy též u,v; s, t)
  - Tzv. texturovací souřadnice
  - Interval 0 – 1
- Jeden objekt má typicky jen jednu texturu
- Textura může být sdílena více objekty
- Pro každý vrchol trojúhelníku specifikována bod textury (texturovacími souřadnicemi)

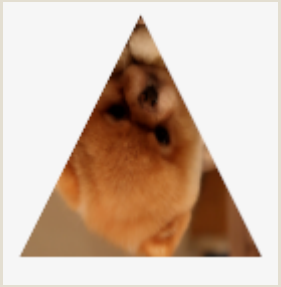
X3D

- ```
  <Shape>
    <IndexedFaceSet coordIndex=" … ">
      <TextureCoordinate point=" …"/>
            …
  </IndexedFaceSet>
  <Appearance>
    <ImageTexture repeatS='false'
      repeatT='false'url='"obrDet.png" …'
  ```
- Pro předdefinované X3D objekty (box, sphere, …) texturovací souřadnice specifikovány interně
- Textur může být vícero
  - Různá rozlišení – vybere se automaticky optimální
  ```

XAML

```
<GeometryModel3D.Geometry>
  <MeshGeometry3D Positions="…"
     TextureCoordinates="…" …>
</GeometryModel3D.Geometry>

<GeometryModel3D.Material>
 <DiffuseMaterial>
  <DiffuseMaterial.Brush>
    <ImageBrush ImageSource="texture.jpg"/>
</DiffuseMaterial.Brush>
 </DiffuseMaterial>
</GeometryModel3D.Material>
```



JEDNODUCHÉ ANIMACE

- Kamera rotuje okolo objektu
- V mnoha formátech podpora zcela chybí
  - Musí se řešit na úrovni vlastní zobrazovací aplikace
- X3D
  - Poměrně komplikovaný systém
- XAML
  - Totožné z principy animované 2D grafiky

KONEC

- Příště: dokumenty s grafickým obsahem