VGO

1. Vnímání barev

Tvorba barevného vjemu v lidském oku

- Lidské oko vnímá barvy od 380 do 720nm, světelný paprsek - světlo jdoucí z nějakého směru
- **monochromatické světlo** - má jen jednu vlnovou délku
- **světelné spektrum** - vlnové délky viditelného spektra
- **barva** - vnímání vlnových délek světelných paprsků
- **oko** - 2/3 vnitřního povrchu oka jsou fotoreceptory, rozlišení 1 megapixel, žlutá skvrna - největší koncentrace čípků, slepé místo - nejsou zde žádné fotoreceptory, mozek doplní, 2 různé buňky
  - Čípky - 8 milionů, 3 typy (RGB) jejich kombinace zajišťuje vnímání barev
  - Tyčinky - 10x citlivější než čípky, 120 milionů, nevnímají barvy jenom jas (night vision)
- Tvořeno 3 složkami R, G a B - trojdimenzionální prostor
- Nejcitlivější je oko na zelenou, pak červenou a nejméně na modrou
- **svítivost** - citlivost lidského oka na jas

Kolorimetrický experiment

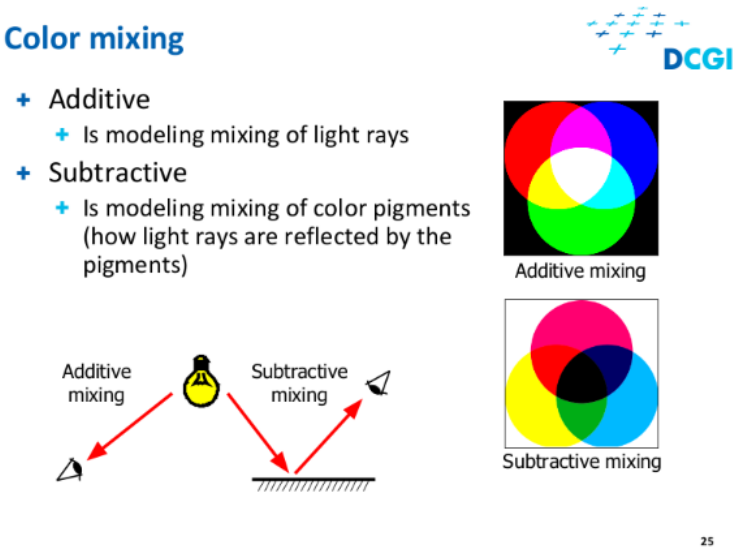
- 2 různá světla, jedno je monochromatické a druhé je složeno s RGB, účastník experimentu se snaží upravovat intensitu světel, aby se světla rovnala (ne vždy je to možné)

Srovnávací funkce (CIE RGB), barevné prostory CIE XYZ a CIE xyY, chromatický diagram.

- **CIE RGB** - definuje srovnávací funkce pro všechny 3 barvy RGB, tyto funkce neodpovídají přesným fyzikálním délkám, ale jsou to matematické transformace, tyto funkce udávají pro danou vlnovou délku λ jakou hodnotu mají mít jednotlivé RGB složky
- **CIE XYZ** - 2 hlavní cíle, srovnávací funkce jsou vždy kladné, jedna fce je rovna svítivosti, definuje i světla, která nelze fyzikálně realizovat
- **CIE xyY** - normalizovaný barevný prostor XYZ, pro daný jas Y můžeme zobrazit pomocí chromatického diagramu s osami xy
- **Chromatický diagram** - diagram znázorňují barevný prostor, kde platí že pokud máme světla reprezentované jako body například (A,B), tak lze různými intensitami, těchto světel získat jakoukoliv barvu na úsečce mezi nimi, to samé platí pro tři světla ABC

2. Barevné modely

Aditivní a subtraktivní skládání barev.



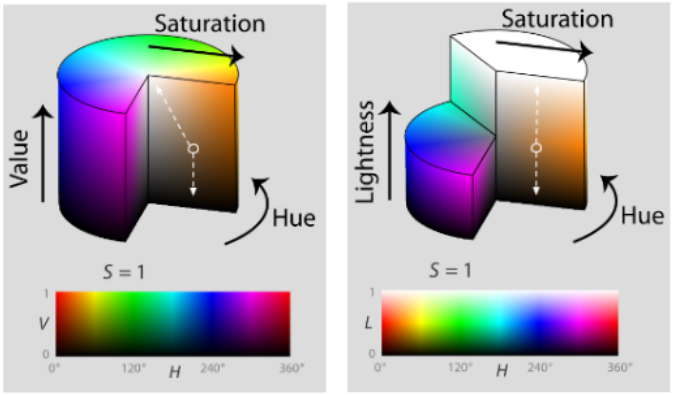
Barevné modely založené na primárních barvách RGB a CMY(K).

- reprezentují barvu
- je potřeba vzít v potaz:
  - Fyzikální realizovatelnost - například CIE XYZ popisuje barvy, které nejsou fyzikálně možné

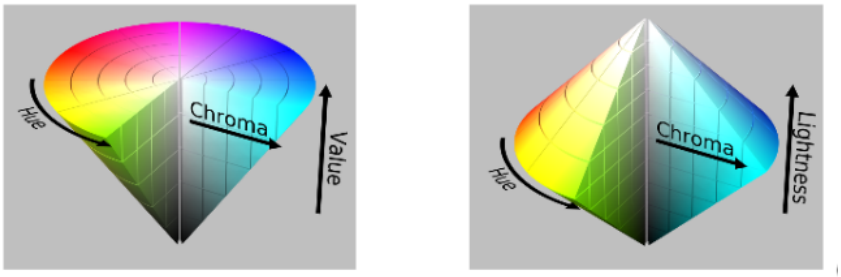
    - Efektivní kódování barev - některé modely berou v potaz limitaci oka
    - Vnímaná jednotvárnost - euklidovská vzdálenost mezi barvami v modelu odpovídá rozdílu mezi vnímanými barvami
    - Intuitivnost vytváření barev uživatelem v modelu
- **barevný rozsah (gamut)** - obsahuje všechny barvy, které zařízení může zobrazit, pro danou svítivost je gamut trojúhelník dán primárními barvami (viz chromatický diagram)
- **RGB** - aditivní skl8d8n9 RGB, displeje, projektory, barevný prostor je jednotková kostka, 0 až 1 nebo 0 až 255 (8 bitů kanál)
- **CMY(K)** - subtraktivní skládání, Cyan (azurová), Magenta (purpurová), Yellow, blacK, používáno v tisku, černá se používá protož spojení CMY vznikne jen šedá a použije se těchto barev hodně, barevný gamut menší než RGB a jelikož se pracuje s pigmenty tak nelze tisknout velmi jasné odstíny RGB
- Nevýhody modelů založených na primární barvách pro uživatele
    - Neprimární barvy se těžko skládají
    - Zmenšení sytosti barvy nebo odstínu se dělá těžko

Abstraktní barevné modely HSV a HSL.

- **HSV** - Hue (odstín), Saturation (Sytost), Value (Brightness, Jas), nekonzistentní - přidáváním černé se mění jen Value, ale přidáváním bíle se mění Value a Saturation (viz obrazek)
- **HSL** - Hue (odstín), Saturation (Sytost), Lightness (Jas), konzistentní - přidáváním černé a bílé se mění jenom Lightness



HSV HSL



- **Kvantizace barev** - barevné prostory se rozdělí na intervaly, může být uniformní nebo neuniformní, s N bity můžeme reprezentovat 2^n barev
    - 8 bit RGB - 256 barev
    - 16 bit RGB - 65536 barev
    - 24 bit RGB, 32 bit RGBA - 16.7 milionů barev

3. Rastrová grafika

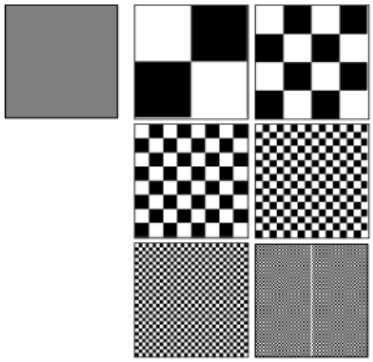
Obraz jako signál, vzorkování, alias a antialiasing.

- Matematická reprezentace 2D obrázku je funkce, která má spojitý definiční obor i obor hodnot
- Když chceme zobrazit obrázek musíme obory diskretizovat
- **Digitalizace** má 2 kroky - vzorkování a kvantizace barev
    - vzorkování - diskretizace obrázkové funkce, pro každý pixel našeho diskretizovaného obrázku vezmeme vzorek z původního, čím větší frekvence vzorkování tím větší je rozlišení
- Reprezentace pomocí 2D pole, každý prvek je pixel (picture element)
- **PPI** - počet pixelů na palec, zajišťuje stejnou velikost obrázku na různých zařízení

- **DPI** - počet bodů na palec, určuje kolik inkoustových teček bude na vytisklém obrázku na palec
- **Alias** - výsledek nedostatečné frekvence vzorkování funkce, je potřeba vzorkovat alespoň s frekvencí 2 * f, kde f je frekvence vzorkované funkce, například - vzorkování hodin (ručičkové na zdi), každých 10 minut funguje, ale pokud budeme vzorkovat každých 50 minut, bude to vypadat jako když jdou pozpátku
- **Anti-aliasing** - odstraňení efektu aliasingu
  - Supersampling - barva pixelu je průměr okolních barev, pro každý pixel se dělá několik vzorků, náhodná vs uniformní distribuce
  - Low pass filter - vysoké frekvence jsou odstraněny ze signálu před vzorkováním

Kvantizace barev, polotónování a dithering.

- **Kvantizace barev** - diskretizace rozmezí barev, rozdělíme původní rozmezí na intervaly a každému dáme jednu barvu, uniform nebo non-uniform
- **Polotónování** - každý pixel barvy kterou nelze vytvořit z barev na zařízení je nahrazen maticí (gridem) vytvořitelných barev, které z dálky vypadají

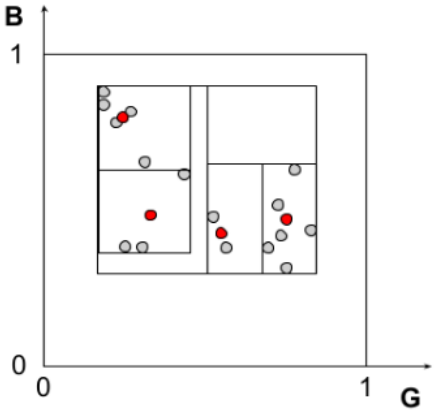


jako nevytvořitelná barvy, u barevným obrázků se použije pro každý barevný kanál zvlášť, tisk novin
- **Dithering** - snaží se docílit efektu polotónování bez zvýšení rozlišení, používáno buď ve spojení s polotónováním, určuje ho matice
  - Náhodný - větší než threshold 1 menší 0, nevypadá dobře
  - S maticí - předdefinovaná matice,
  - S polotónováním - udává vzor (pattern) polotónování na generaci různých odstínů barev
  - Bez - jelikož polotónování zvyšuje rozlišení, tak použití kvantizace barev a ditheringu zajistí podobný efekt, bez zvýšení rozlišení

Přímá reprezentace barev (direct color) a indexovaná reprezentace barev.

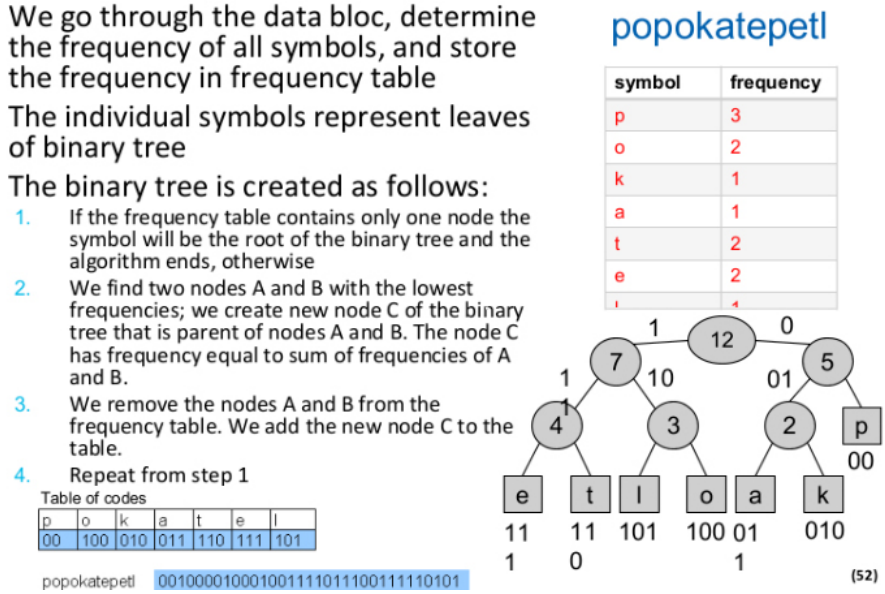
- **Přímá reprezentace** - v každém pixelu jsou jeho barvy, lze rozdělit obrázek na 3 grayscale obrázky, kde každý reprezentuje jednu barvu (+ 1 na alfu pokud je to RGBA)
  - 8 bit RGB - 256 barev
  - 16 bit RGB - 65536 barev
  - 24 bit RGB, 32 bit RGBA - 16.7 milionů barev
- **Indexované barvy** - obrázek obsahuje jen jeden kanál na indexy barev, unikátní barvy jsou uloženy v poli (paleta barev), průhlednost buď RGBA paleta nebo jedna barva s alfa 0
  - master color palette - 256 barev, bity 3R-3G-2B podle citlivosti oka, vzorek z jednotkové RGB kostky

- adaptive color palette - najdeme nejbližší barvy k originálu, balancujeme počet barev a error rate



Komprese rastrového obrazu. Základní formáty GIF, PNG a JPEG a jejich vlastnosti.

- metody jak zmenšit paměťovou náročnost obrázku, pro šetření místa, posílání přes síť
- **lossless** - žádné informace nejsou ztraceny, lze rekonstruovat původní obrázek
- **lossy** - některé informace jsou ztraceny, nelze rekonstruoval, používá znalost limitací lidského oka
- **RLE (run lenght encoding)** - lossless, wwwwwwbrrrrr -> 6w1b5r, vhodné pro indexované barvy, lze vyprodukovat negativní kompresi
- **Huffmanovo kódování** - lossless, vytvořeno pro posílání černobílých faxů, zjistí se frekvence všech symbolů a vytvoří se tabulka, z ní se vytvoří binární strom



- **LZW comprese** - lossless, v GIF, PNG, .zip, uloženo jako strom, Input ABRABABRA, během komprese se sestaví slovník všech použitých znaků a za sebou jdoucích znaků a pak se daný soubor uloží pomocí jejich indexů (1 znak = jedno číslo), tento postup sestavování se použije i při dekompresi, s tím, že musíme znát všechny použité znaky a jejich indexy (https://www.youtube.com/watch?v=1KzUiklae6k)
- **discrete cosine transformation** - předešlé metody nevhodné pro fotky, JPEG, lossy, volitelná kvalita komprese
  - transformace z RGB do YC_bC_r makrobloků (Y - jas, C_bC_r - red and blue differences)
  - discrete cosine transformation makrobloků
  - kvantizace barev pomocí matice
  - linearizace po diagonále
  - RLE encoding
  - Huffman encoding
- GIF - indexované barvy, lze alfa 0 nebo 1, LZW, podporuje jednoduché animace, vhodné pro limitované barevné palety
- PNG - podpora indexovaných barev, RGB i RGBA, lossless (příprava pak LZW), vhodné jak pro fotky (větší velikost než JPEG), tak pro loga a sketche
- JPEG - přímé barvy, žádná alfa, lossy DCT, lze zvolit kvalitu, vhodné pro fotky, nevhodné pro profesionální fotky (TIFF), nebo limitované barevné palety

Transformace rastrového obrazu.

- pomocí transformační matice, problém že po transformaci souřadnice pixelů nemusejí být integery

- dopředný mapování - pixel transformován maticí, problém je že několik pixelů může přepsat stejný finální pixel nebo vytvořit mezeru
- inverzní mapování - pro každý pixel výsledného obrázku je samplován původní obrázek, nejsou díry ani přepisy ale pořád to má problémy -> řešení rekonstrukce spojité image funkce a lineární nebo bilineární interpolace

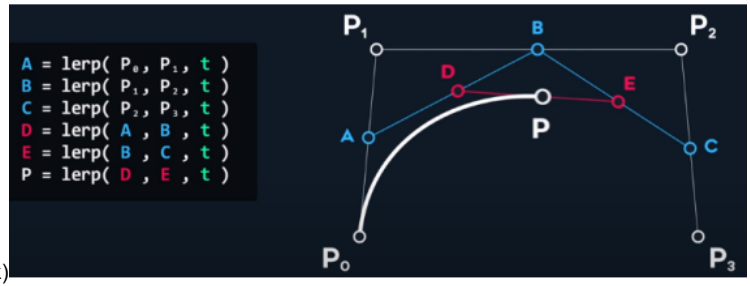
4. Vektorová grafika a reprezentace 2D objektů.

Reprezentace 2D objektů pomocí parametrických polynomiálních křivek

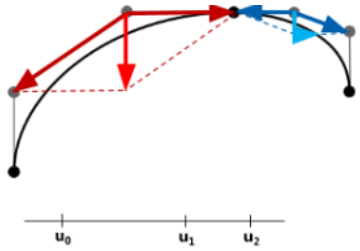
- Aproximační křivky - nemusí procházet vůbec kontrolními body, každý bod křivku "přitahuje" k sobě
- Interpolační křivky - prochází všemi kontrolními body, většinou má každý bod nějaká omezení jako jejich tečný vektor
- stupeň křivky = počet kontrolních bodů - 1
- řád křivky = počet kontrolních bodů
- změna kontrolních bodů má efekt na celý segment křivky, proto se rozdělují na jednotlivé segmenty
- uniform - čas strávený v každém segmentu je stejný, vzdálenosti v knot vectoru jsou stejné
- non-uniform - čas strávený v každém segmentu je různý, vzdálenosti v knot vectoru mohou být různé
- spojitosti segmentů
  - C0 - segment 2 začíná tam kde segment 1 končí, zachovává poziční spojitost
  - C1 - C0 + směr derivace (tečny) a jejich velikost v napojení u obou segmentů jsou stejné, zachovává rychlost
  - C2 - C1 + směr druhé derivace (tečny) a jejich velikost v napojení u obou segmentů jsou stejné, zachovává zryhlení
  - Geometrická - Cn => Gn, u geometrické se velikosti derivací nemusí rovnat
- racionální křivky - nelze použít křivky v perspektivní projekci, použijeme homogenní souřadnice [xw, yw, w], když je w=1, jsme v kartézské soustavě souřadnic, w reprezentuje váhu bodu, čím větší je, tím víc křivka následuje daný bod, nevhodné pro animace, spojitost se počítá v homogenních souřadnicích (narušují ji)

Bézierovy křivky

- aproximační křivka, první a poslední část je rovna prvnímu a poslednímu bodu, křivka je vždy uvnitř konvexního pláště bodů, nejpoužívanější 2 a 3 stupeň
- DeCasteljau - rekurzivní způsob jak najít bod na křivce, lze tímto rozdělit křivku na dvě (tímto způsobem se vykreslují), buď Bernsteinovy polynomy nebo lineární interpolací (obrázek)



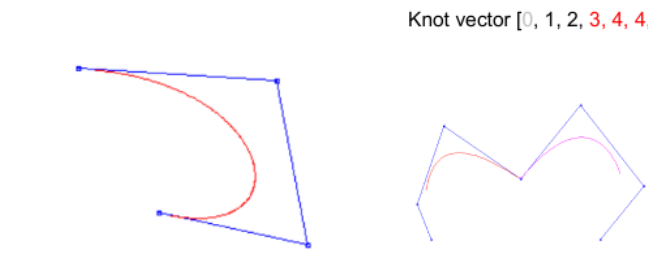
- C0 - konec jednoho segmentu v začátku druhého
- C1 - 3. a 4. bod prvního segmentu a 1. a 2 bod druhého segmentu leží na jedné přímce
- C2 - platí to co pro C1 + obrázek

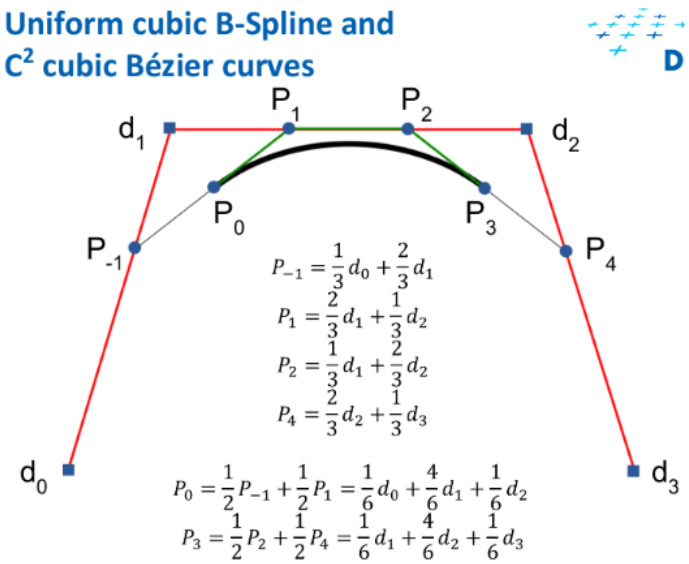


- Rendering - dělení na půl pomocí decasteljau v homogenním souřadnicovém systému, dokud nový segment nemá menší obsah než threshold

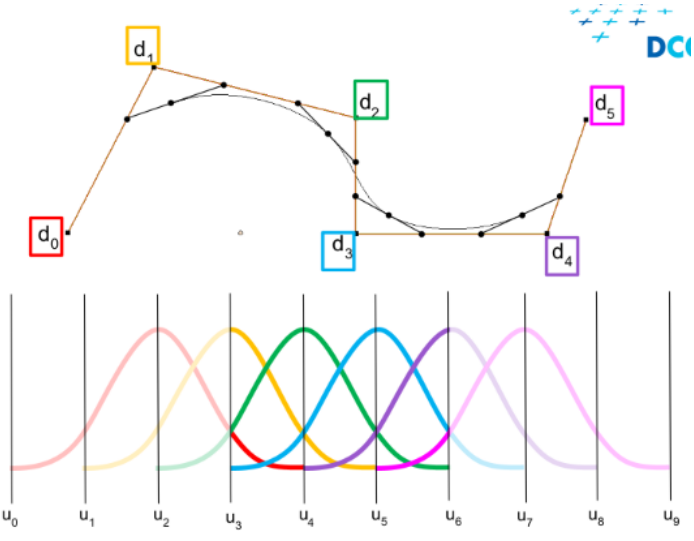
B-spline křivky

- aproximační křivka, segmenty se překrývají, na přidání segmentu stačí přidat jeden bod, jeden segment je coonsova křivka
- invariantní vůči afinním transformacím - pohybuje se pohybem kontrolních bodů
- racionální - invariantní vůči perspektivní projekci
- spojitost je rovno C^{n-r}, kde n je stupeň křivky a r násobnost knotu/bodu
- změna bodu ovlivní jen pár segmentů kolem
- knot vektor - posloupnost rostoucích čísel, udává váhu jednotlivých bodů, pro 4 body abcd je uniformní knot vector [0,1,2,3,4,5,6,7] - body + řád, úpravou lze křivku přiblížit nebo oddálit od bodu ale -> Non-Unifrorm





- B-spline to bezier
- NURBS (Non-Uniform Rational B-Spline) - nejvíc obecná křivka



- Bázové funkce - určují váhu jednotlivých bodů v rámci segmentů
- DeBoor - generalizace deCasteljau, lze použít na najití bodu na B-Spline
- rendering - reparametrizace křivky, zvětšit násobnost uzlů na n
- použití hlavně v modelování

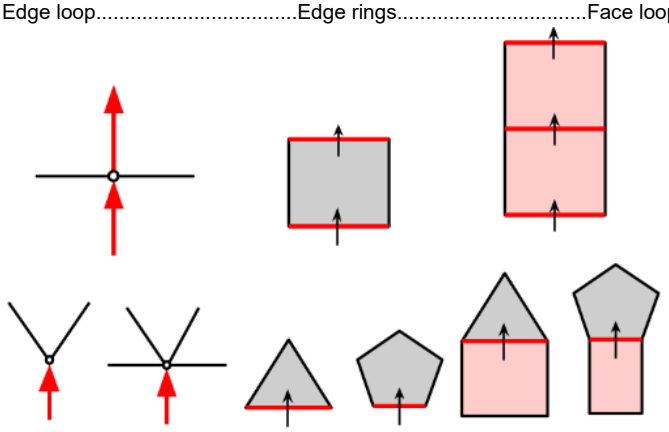
Formát SVG (+vector graphics)

- scalable vector graphics, formát tagů, reprezentováno instrukcemi jak vykreslit tvar, díky tomu lze zvětšit libovolně, aniž by byla horší kvalita
- nevýhoda jsou kvantizované barvy
- posloupnost kreslení - malířův algoritmus (renderování podle pořadí instrukcí), malířův problém (tři proužky se všechny navzájem překrývají, řešení je jeden rodělit na dva)

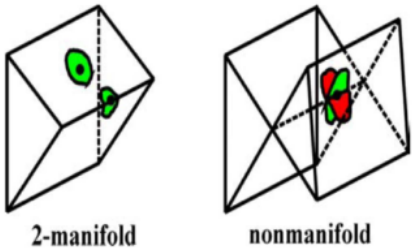
5. Reprezentace 3D objektů a principy 3D modelování

Polygonální reprezentace 3D objektů a datové struktury pro jejich reprezentaci, formát OBJ.

- Implicitní plochy - viz. kapitola níže
- Parametrické křivky a plochy - viz. kapitola níže
- Subdivision surfaces - viz. kapitola níže
- Volumetric representation - scany v nemocnicích, reprezentace mnoha 3D dat
- Procedurální - předpis jak vytvořit
- Boundary representation - reprezentace povrchu, složeno z polygonů
- Polygon - obsah roviny definovaný vrcholy, reprezentovaný seřazeným seznamem vrcholů, používají se trojúhelníky (vždy konvexní, vždy v rovině) na vykreslování a quady na modelování (dobré vlastnosti na operace viz. kapitola níže), CW nebo CWW (podle toho se dá určit front a back side), convex vs non-convex
- Polygonální polévka - nestrukturované, seznam nezávislých polygonů (stěn), vrcholy se opakují, nemáme informace o topologii, problémy s polygony se těžko opravují
- 3D mesh - strukturovaný, informace o topologii, vrcholy [x, y, z, 1]
  - stěny - tvořeny vrcholy, polygony, leží v jedné 3D rovině, aproximace zakřiveného povrchu)
  - hrany - strany polygonů, hodí se na topologické informace
- Topologie - sousednost {kombinace dvojic stěn, vrcholů, hran}
- Edge loop.....Edge rings.....Face loop



- normálové vektory - určují vektor, který svírá 90 stupňů s plochou polygonu, per face nebo per vertex
- Datové struktury pro renderování
  - Polygonální polévka - seznam nezávislých polygonů, každý face je reprezentován 3 vrcholy, zbytečný přebytek vrcholů
  - Seznam vrcholů a stěn - dva separátní listy, všechny vrcholy uloženy souřadnicemi, v listu se stěnami jsou 3 čísla určující indexy vrcholů, průměrně 2x méně paměťově náročné než soup, sousednost snadno zjistitelná, nemáme informace o hranách
- .obj - používá seznam vrcholů a stěn, uloženo pomocí znaků, "v" vertex, "vn" vertex normal, "vt" vertex texture, "f" face, definice materiálu
- Datové struktury pro modelování
  - Seznam sousednosti - máme list vrcholů, hran a stěn, a pro každý prvek těchto listů máme listy sousedních vrcholů, hran a stěn, velmi paměťově náročné, perfektní informace o topologii
  - Winged edge (okřídlená hrana) - pro každou hranu známe sousední vrcholy a stěny, pro každý vrchol a stěnu známe jednu hranu, paměťově méně náročné a operace stejně výpočetně náročné
  - Winged edge - half-edges - problém u normální winged edge je, že každá hrana je reprezentována jenom jednou a nemáme informace o CW a CWW stěn, half edges mají jeden sousední vrchol, jednu stěnu a následující, předešlou a sourozeneckou hranu
- Orientovatelný povrch - pokud každá hrana sousedí s 2 stěnami, které mají různou orientaci, mobius strip, klein bottle



- Manifolds - lokálně podobný povrch R^n, kruh (1-manifold), koule (2.manifold)
- Uzavřené a otevřené povrchy - uzavřený je vodotěsný, otevřený má v sobě díru

Bézierovy a B-Spline plochy.

- použity pro přesné popsání zakřivených ploch, lze transformovat pouze pomocí změny kontrolních bodů

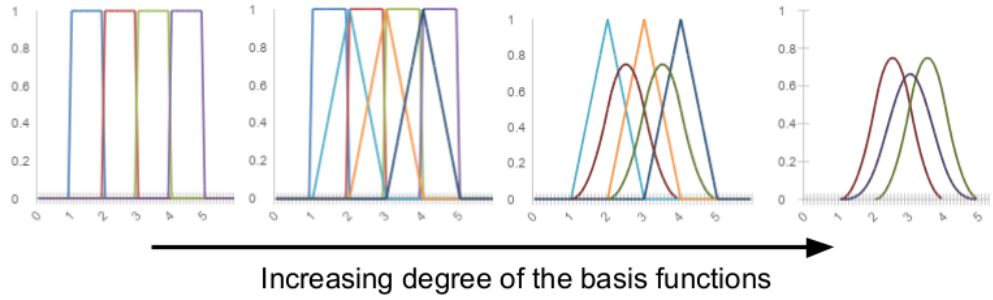
- požadavky
  - Přesnost - přesná reprezentace objektu
  - Kompaktnost - různorodé tvary jenom s pár kontrolními body, malé paměťové nároky, snadné editování
  - Invariantní vůči transformacím - afinní transformace a perspektivní projekce, racionální (homogenní souřadnice 4D, všechny algoritmy probíhají v hom. souř. a výsledek se promítne do 3D a W je použita jako váha bodu)
  - Lokalita změn - změny v kontrolních bodech mají efekt pouze na lokální okolí bodu, řešení --> piecewise
  - Spojitost - chceme spojí křivky, ideálně C2
  - Efektivní rendering - snadno a rychle lze vyrenderovat
- Z křivek na povrch - pokud máme několik křivek a aktuální "t" použijeme jako kontrolní body další křivky v parametrem třeba "u" a projedeme všechny varianty $u, t \in < 0, 1 >$, tak získáme křivkovou plochu

Bezier plochy

- aproximační polynomiální patch, obdélníkový, nejpoužívanější 2 nebo 3 stupeň, uvnitř konvexního trupu bodů, C2 lze převést na B-Spline a naopak
- je zde na vykreslování použit decasteljau akorát místo interpolace je použita bilineární interpolace (direct)
- Také existuje indirect decasteljau, který je složitější a výpočetně náročnější, ale poskytuje větší kontrolu na výslednou plochou. Místo rozdělování plochy vytvoří síť bodů iterativním upravováním hodnot parametrů.
- C0 (segmenty jsou napojeny), C1 (žádné ostré hrany, změny v derivaci/tečně jsou spojité), C2 (zajišťuje spojité zakřivení plochy)
- hodně kontrolních bodů, těžce upravovatelné

B-Spline plochy

- dokáže reprezentoval jakoukoliv bezier plochu, ale použije míň bodů, protože zajišťuje Cn spojitost, kde n je stupeň
- NURBS - nejvíce univerzální polynomiální plocha
- 2 knot vektory jeden pro každou parametrizaci (U a V)
- počet segmentů = počet bodů - 2, počet uzlů = počet bodů + 5

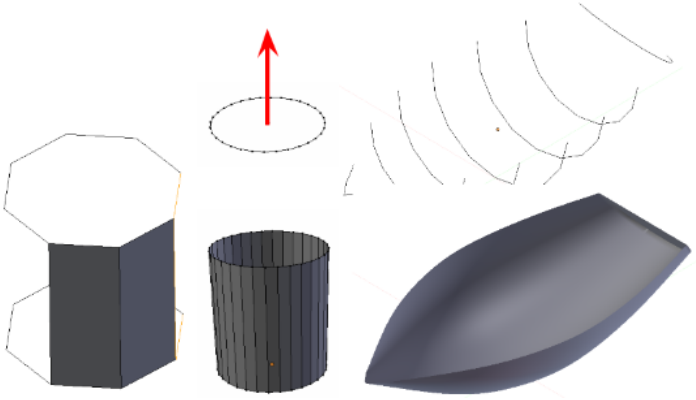


(Bázové funkce B-Spline a jejich odvození, platí i pro PGR a předešlé otázky)

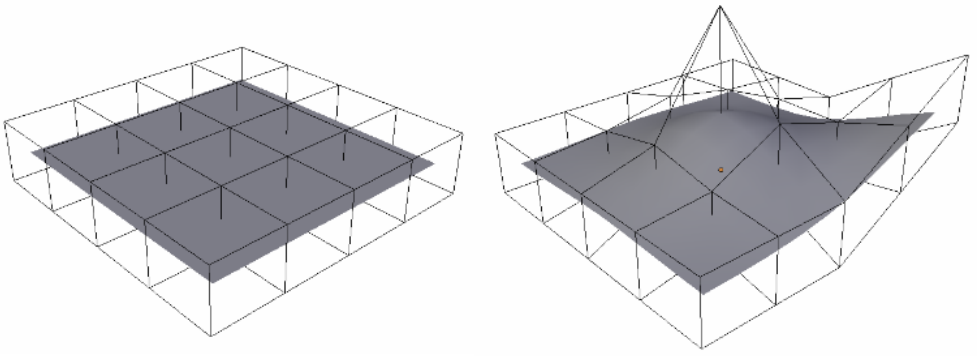
- pro vykreslení direct de boor (viz PGR nebo předešlá otázka), zase existuje i indirect varianta lišící se podobně jako de casteljau indirect

Základních 15 modelovacích operací s použitím polygonální a polynomiální reprezentace 3D objektů: blokování, bridge, extrude, loft, rotační plochy, volné modelování.

- Blocking - upravování vrcholů, hran a stěn primitiv, pro vytvoření komplexnějších objektů, používá se i průnik objektů
- Bridge - spojí 2 polygonové profily stěnami, stejný 3d objekt, najde vždy 2 optimální hrany a spojí je stěnou, 1. obrázek
- Extrude - vytvoří stěny taháním profilu podél přímky, profil může být vrchol, uzavřený/otevřený polygon (hrany) nebo stěny, duplikace profilu, posun duplikátu, bridge dvou profilů, 2. obrázek
- Loft - automatické bridgování několika profilů, 3. obrázek



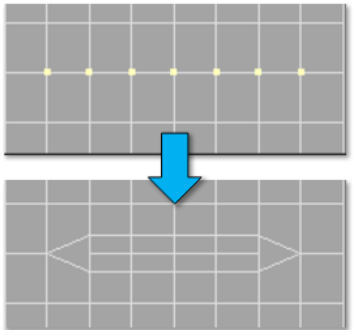
- Rotační plochy - rotace profilu v prostoru kolem osy, máme kontrolu nad úhlem rotace, počet kroků, osa, origin, duplikace profilu, pootočení o úhel kolem osy, bridge, repeat
- Free form modelling - přímé upravování vrcholů, hran, stěn, edge loops a face loops, pomocí translate, rotate, scale, lze upravovat jeden element nebo proporčně v okolí elementu
- Tvoření nové geometrie - nové vrcholy pouze z existujících (extrude), spojení vrcholů/hran pomocí hran/stěn
- Následují operace modifikují topologii meshe
- Knife - rozdělí hranu nebo stěnu na dvě
- Subdivide - rozdělí každý quad/triangle na 4
- Loop cut - rodzdělí každý quad podel face loop na 2 a více
- Snapping - při posunu vrcholu můžeme vrchol umístit přímo na místo jiného vrcholu
- Modifiers - nástroje upravující geometrii, můžeme je řetězit, dělí se na deformační (žádná nová geometrie)
  - Simple deform - twist, bend, stretch
  - Lattice (mřížka) - deformační mřížka, objekt by měl být celý uvnitř, měněním bodů mřížky se deformují body objektu (body mřížky "přitahují" body), existuje i deformační mesh (podobný mřížce, ale používá se pro komplexnější objekty)

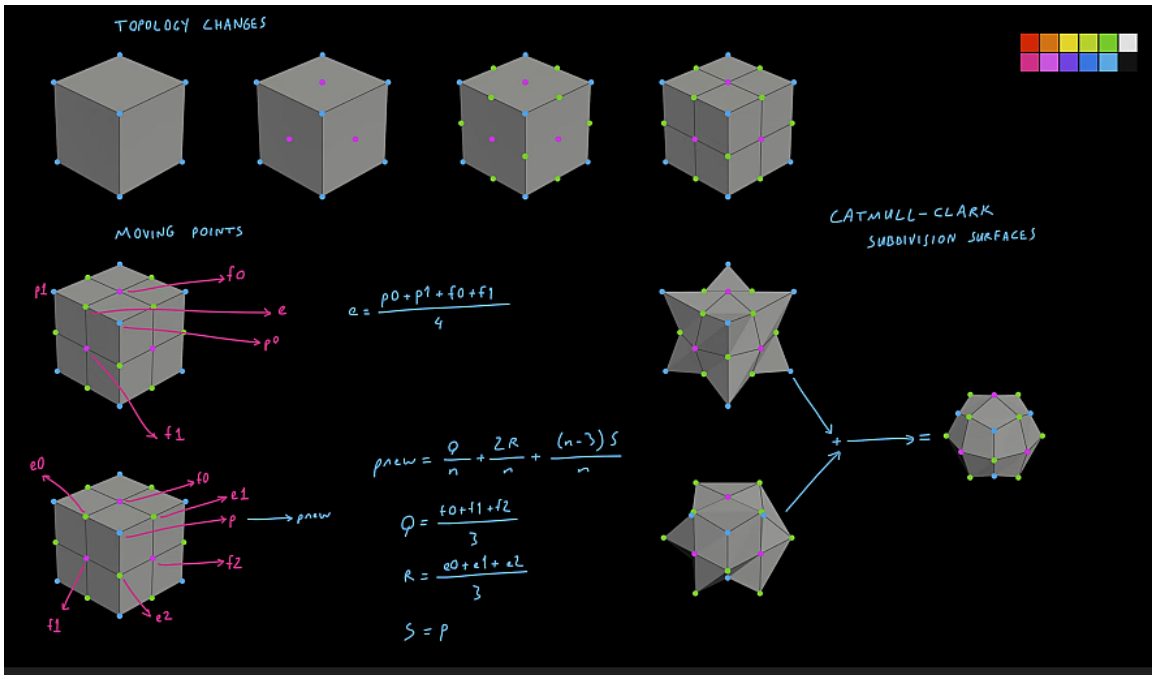


  - Curve - poloha objektu se nastaví na začátek křivky, pak je objekt roztažen podél dané křivky
- Modifiers - generující (vytvořena nová geometrie)
  - Mirror - podél osy se mesh zrcadlí, vhodné pro symetrické modelování (auto, obličej), modeluje jen půlku meshe
  - Array - několikrát duplikuje 3D objekt v nastavených intervalech a rotacích, lze následovat křivku, například řetěz
  - Bevel - zkosení ostrých hran objektu
  - Triangulate - převede všechny quady a n-gony na trojúhelníky
  - Decimate - zmenší počet polygonů objektu, používá edge collapse (progressive mesh) který z jedné edge udělá vrchol, vybraná například podle toho jak by se decimovaný mesh lišil od původního (global)/od předešlé decimace (local), LOD
  - Boolean - boolovské operace s 2 objekty, průnik, sjednocení, rozdíl, vhodné pro Constructive Solid Geometry (CSG)
  - Subdivision surfaces - každý quad na 4 quady, každý trojúhelník na 3 quady, simple - rozdělení bude placaté, catmull-clark - 3D objekt bude hladší s každou iterací, generalizace B-Spline surfaces

Dělené plochy (subdivision surfaces), jejich reprezentace a výhody při modelování oproti B-spline plochám.

- catmull clark, potřebuje quady, jinak vznikají artefakty, reprezentováno meshem a rodělujícími pravidli
- Nevýhody Bezier a B-Spline - nelze modelovat objekty s víc než 1 dírou, jsou limitovány obdélníkovou topologií, pro modelování komplexních objektů je potřeba ořezávat a napojovat B-Spline, nemůžeme ovládat pouze lokální level of detail
- Výhody catmull-clark - nemusíme spojovat několik b-spline, C2 by default (až na výjimky), lze ovládat lokální level of detail, lze definovat ostré hrany





Implicitní plochy a jejich reprezentace a využití při modelování.

- Modelování objektů, které se mohou spojit je těžké s polygonální reprezentací
- Reprezentováno implicitní rovnicí, pro každý bod (x, y, z) víme zda leží uvnitř, na, nebo vně tělesa, rovnice definuje distance field, v modelování se nazívají Blobs/Meta balls
- Pro každý meta ball známe jeho distance field (radius a falloff), tvar (primitivum podle rovnice), tam, kde je více meta balls tak se buď sjednocení nebo průnik
- Pro vykreslení povrchu se použijí Marching cubes nebo raycasting
- Není moc objektů, pro které by byl tento postup vhodný, maximálně vodní kapky, molekuly

Sochání (sculpting).

- vhodné pro objekty s velmi komplexním a přirozeným tvarem, možnost protnout sama sebe
- uživatel kreslí na 3D objekt barvu, která určuje jak se upravuje výšku vrcholů
- různé štětce na různé vzory
- využívá se multi-resolution 3D mesh

Další

- Cloth physics modelling
- Particle systems - vlasy
- Volumetrická data - kouř

6. Principy mapování textur

Zobrazení textury na 3D objektu s použitím UV mapování.

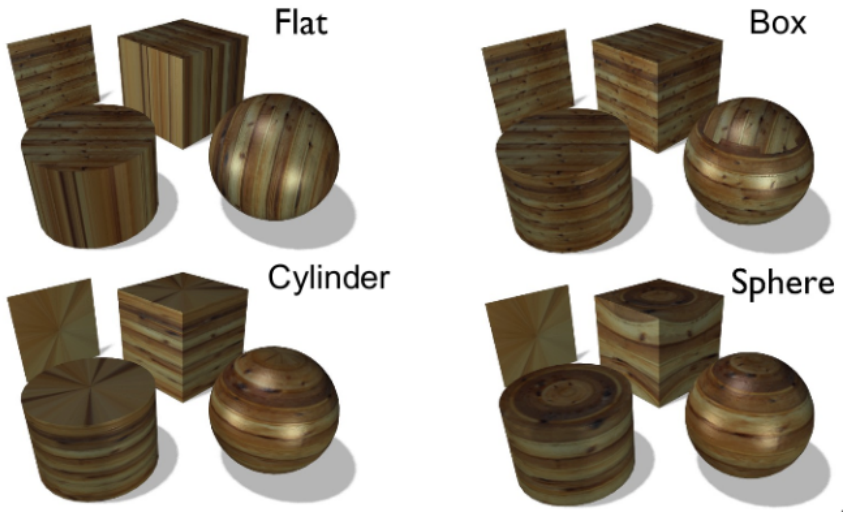
- textura - funkce reprezentující vlastnosti povrchu objektu, 1D, 2D, 3D
- mapování - definujeme pro každý vrchol, souřadný systém 2D textury je od 0 do 1 a definuje osu U a osu V, pro každá vrchol definujeme UV souřadnice, pro bod mezi vrcholy interpolujeme mezi nimi
- polygony mimo UV - repeat - souřadnice jsou modulovány, extend - souřadnice clamped na 0,1, Clip - defaultní barva se použije
- Mip-mapy - viz PGR
- Textury - barva, specular (definuje lesklost), roughness (definuje hrubost), normal (posun normál), displacement (posun vrcholů)

Různé způsoby vytvoření UV mapování a k čemu jsou vhodné.

- **UV unwrapping** - proces transformace 3D geometrie na 2D souřadný systém textury
- **Unwrapping pomocí švů** - definujeme podél jakých hran se model rozdělí na jednotlivé kusy v UV mapě, vhodné pro jednoduší modely
- **Unwrapping podle limitujícího úhlu** - automaticky, rozdělí se na kusy podle limitujícího úhlu mezi normálními vektory polygonů, s nižším limitem

se vytvoří víc separátních kusů v UV mapě

- **Lightmap packing** - vytvoří obdélníkový grid, tvar, sousednost polygonů a poměr stran není zachován, velmi efektivní využití prostoru, většinou více než jeden objekt, vhodné pro přepočítané osvětlení nebo ambient occlusion
- **Projekce** - projekce z různých pohledů, vhodné pro placaté objekty nebo objekty podobné projekčním primitvům
  - Box - 8 různých placatých projekcí, zvolí se pro polygon podle normály
  - Sphere - použije sférické souřadnice
  - Cylinder - podobné sférickým, používá z souřadnici



- Projekce z pohledu - použije aktuální pohled kameru na projekci, vhodné pro placaté povrchy