PG1 - okruhy na zkoušku v1.1

Jakub Konĕrza

March 10, 2025

Abstract
https://www.youtube.com/watch?v=rYmWbXZB564

1 Světlo

“ A řekl Bůh: Budiž světlo. A bylo světlo. “

1.1 Podstata světla

Světlo má duální charakter:

- Projevuje se jako elektromagnetické vlnění a proud částic (fotonů)
- Ve vakuu se šíří konstantní rychlostí 299 792 458 m/s
- Foton je charakterizován pouze svojí vlnovou délkou, která určuje jeho energii

1.2 Elektromagnetické spektrum

Elektromagnetické záření se dělí podle vlnové délky na:

- Rádiové vlny (delší jak 1m)
- Mikrovlny (1mm - 1m)
- Infračervené záření (760nm - 1mm)
- Viditelné světlo (390nm - 790nm)
- Ultrafialové záření (10nm - 400nm)
- Rentgenové záření (0.1nm - 10nm)
- Gamma záření (kratší jak 0.1nm)

1.3 Bílé světlo

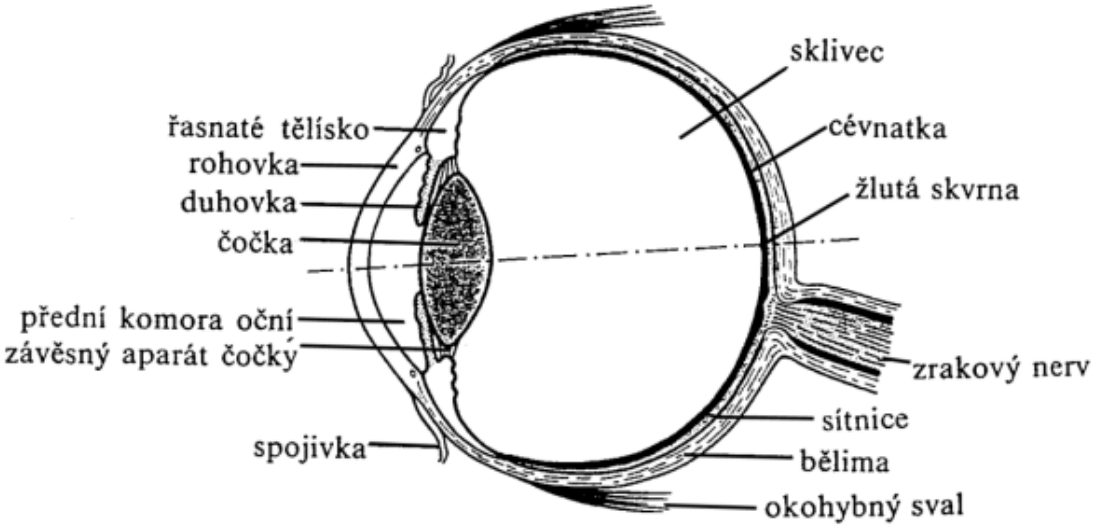
- Každý bílý světlo má rozdílný barevný spektrum. Je to jinak poskládaný.

1.4 Světelné efekty

- Odraz (změna směru světla)
- Lom (světlo prochází dvěma prostředími s různými indexy lomu)
- Disperze (rozklad světla na spektrální složky (různý barvy) při lomu, např. duha)
- Difrakce (např. ohyb na mřížce)
- Interference (více vln světla se kombinují a vytvářejí nový vlnový vzorec)
- Polarizace (polarizační filtry propouštějí jen světlo s určitou orientací, např. polarizační brýle)
- Luminiscence (vyzařování světla něčím, co nebylo zahřáto na vysokou teplotu, např. světlušky)

2 Lidské oko

2.1 Anatomie oka



2.2 Tyčinky a čípky

Tyčinky:

- Černobílé vidění
- Počet: +- 120 000 000
- Aktivní při nízké intenzitě světla
- Citlivé na pohyb (periferní vidění)

Čípky:

- Barevné vidění
- Počet: +- 8 000 000
- Aktivní při vyšší intenzitě světla
- 3 druhy podle fotopigmentů:
  - Modré (2%)
  - Červené (32%)
  - Zelené (64%)

3 RGB barevný model

3.1 Co to je?

RGB je "aditivní" barevný model, míchají se tam tři složky:

- Červená
- Zelená

- Modrá
- Nejčastěji se používají kódování:
  - 1 unsigned byte (8 bitů): 0 - 255
  - 1 unsigned short (16 bitů): 0 - 65535
  - 1 IEEE float (32 bitů): 0.0 - 1.0 (inf)
  - 1 half-float (16 bitů): 0.0 - 1.0 (inf)

4 CMYK barevný model

4.1 Charakteristika

CMYK používá čtyři základní barvy:

- Cyan (tyrkysová)
- Magenta (purpurová)
- Yellow (žlutá)
- Key/Black (černá)

4.2 Převody

Převod RGB na CMY (rozsah 0.0 - 1.0):

- $C = 1.0 - R$
- $M = 1.0 - G$
- $Y = 1.0 - B$

Převod CMY na CMYK:

- $K = MIN(C, M, Y)$
- $C' = C - K$
- $M' = M - K$
- $Y' = Y - K$

5 HSV a HSL barevné modely

5.1 Komponenty

HSV model obsahuje:

- Hue (barevný tón)
- Saturation (sytost)
- Value (jas)

HSL model obsahuje:

- Hue (barevný tón)
- Saturation (sytost)
- Lightness (světlost)

5.2 Využití

- Především pro interaktivní výběr barev
- Vizualizace skalárních hodnot pomocí barevné škály
- Převod mezi RGB a HSV/HSL vyžaduje komplexní algoritmus

6 Vzorkování a kvantizace

- Převod ze spojitého signálu na digitální (diskrétní)
- Příklad - digitální fotoaparát:
  - **Vzorkování**: převod spojitého signálu na diskrétní signál v pravidelných intervalech (třeba každou sekundu)
  - **Kvantizace**: převod spojitého signálu na konečný množství úrovní (např. úroveň float 5.2 uložíme jako integer 5 do rozsahu 0-255)

7 Pixel

Označení pro:

- Jeden bod rastrového obrázku
- Jeden svítící bod na monitoru

Poznámka: Pixel na obrazovce a v rastrovém obrazu nemusí být totožný - jeden pixel na monitoru může vzniknout z více/méně pixelů rastrového obrázku.

8 Rozlišení

- Tiskárny: DPI (dots per inch)
- Monitory: PPI (pixels per inch)

Poznámka: Informace o rozlišení nemusí být součástí souborů. Většinou to bývá v metadatech.

9 Dynamický rozsah

- Počet úrovní pro jeden barevný kanál určený počtem bitů. Například: 1bit (bitmapa, dvě barvy), 8 bitů (256 úrovní)
- SONY Alpha A9 III nahrává videa s barevnou hloubkou 10 bitů (1024 úrovní)

10 Barevné prostory

10.1 sRGB

- Nejčastěji používaný barevný prostor
- Převod z XYZ do RGB lineární pomocí transformační matice
- Gamma korekce = lidi vnímají změnu intenzity světla v tmavějších oblastech než ve světlých
- V sRBG se používá nelineární intenzita světla s exponentem přibližně 2.2

11 Komprese obrazové informace

11.1 Bezeztrátová komprese

- **RLE kódování** - nahrazuje sekvence opakujících se hodnot jedním symbolem (AAAAABBBB se nahradí 5A4B)
- **Huffmanovo kódování** - nahrazuje více používané symboly za symboly s kratší délkou (třeba když stokrát zmíníme pomeranč a jednou rybíz, tak pomeranč nahradíme "1" a rybíz "010". Samozřejmě počítáme s větším množstvím slov.
- **LZW kódování** - dynamický slovník během komprese (často opakované sekvence znaků nahrazujeme za nějaký kratší. Sekvence si ukládáme v slovníku)
- **DEFLATE** - kombinace LZ77 a Huffmanova kódování (LZ77 nahradí opakující se sekvence s odkazy na předchozí data, Huffmanovo kódování následně výsledek komprimuje znova

11.2 Ztrátová komprese

Člověk si nevšimne drobných nedokonalostí

- Lidské oko je citlivější na změnu jasu než na změnu barvy
- Lidské oko je méně citlivé na detaily/ostré hrany

12 JPEG komprese

12.1 Základní vlastnosti

- Ztrátový kompresní formát - byl přímo navržen, aby si uživatel nevšiml výrazného rozdílu a přesto se snížila velikost souboru
- Vytvořeno koncem 80. let Joint Picture Expert Group
- Uživatel si může nastavit úroveň komprese
- Režimy činnosti:
  - Sekvenční - komprimace po blocích (8x8 pixelů, převedení obrazu do frekvenční reprezentace pomocí diskrétní kosinové transformace DCT)
  - Progresivní - pro to nejrychlé zobrazení, viditelné na webu (obrázek se přenáší a dekóduje v několika iteracích, tudíž uživatel nejdřív vidí hrubý náhled a následně se obrázek zlepšuje)
  - Bezeztrátový - místo DCT se používá predikční metoda ([pixel vlevo + pixel nahoře] / 2 = predikce, tyto predikce se následně uloží a komprimují)
  - Hierarchický - používá se pro aplikace, kde je potřeba různé rozlišení obrazu (například přizpůsobení různým velikostem obrazovky)

12.2 Kroky komprese

- Převod do barevného prostoru YCbCr (Y=jas + dvě chrominantní složky)
- Převzorkování 4:4:4 (=všechny složky mají stejné rozlišení, 4:2:2 (= chrominantní složky jsou horizontálně vzorkovaný na poloviční rozlišení proti jasové složce)
- Rozdělení obrázků do makrobloků (rozdělení do 8x8 px bloků pro každou složku)
- Diskrétní kosinová transformace DCT (převod na frekvenční data)
- Kvantizace pomocí kvantizační matice (snižuje přesnost vysokofrekvenčních koeficientů)
- Linearizace Zig-Zag (přeskupuje 2D matici do 1D pole, což usnadňuje kódování a hlavně RLE
- RLE kódování
- Huffmanovo kódování

13 Rastrové formáty

13.1 TGA (Targa Truevision)

- Bitmapa (1bpp)
- Barevná paleta (8bpp) RGB
- 256 odstínů šedi (8bpp)
- RGB (24bpp), RGBA (32bpp)
- Volitelně RLE komprese

13.2 BMP (Bitmap Picture)

- **Vytvořen Microsoftem a IBM:** Formát BMP byl původně navržen pro ukládání grafiky v operačním systému Windows a poskytuje jednoduchý způsob ukládání rastrových obrázků.
- **Podporuje různé barevné hloubky:** BMP může uchovávat obrázky s různou úrovní detailu a počtem barev:
  - **1 bpp:** Černobílé obrázky.
  - **4 bpp:** Obrázky s maximálně 16 barvami.
  - **8 bpp:** Obrázky s maximálně 256 barvami.
  - **16 bpp:** Více barevných odstínů díky použití 65 536 barev.
  - **24 bpp:** Pravé barvy, kde každý pixel obsahuje 8 bitů pro červenou, zelenou a modrou složku (16,7 milionů barev).
- **Barevná paleta:** U obrázků s menší barevnou hloubkou (např. 4 nebo 8 bpp) BMP využívá barevnou paletu (Color Table), která definuje přesné RGB hodnoty pro každý index barev.
- **Volitelná RLE komprese:** BMP může používat jednoduchou kompresní metodu Run-Length Encoding (RLE), která je efektivní u obrázků s velkými plochami stejné barvy, ale u většiny použití zůstává nekomprimovaný, což zvyšuje velikost souboru.

13.3 GIF (Graphics Interchange Format)

- **256 barev:** Používá barevnou paletu s maximálně 256 barvami na snímek, což omezuje jeho použití u detailních obrázků.
- **Podpora animací:** Formát umožňuje ukládání vícesnímkových animací v jednom souboru.
- **Jednobitový alfa kanál:** Podporuje průhlednost, ale pouze jako zapnuto/vypnuto (plně průhledné nebo neprůhledné).
- **LZW komprese:** Bezeztrátová komprese, která redukuje velikost souboru bez ztráty kvality.

13.4 PNG (Portable Network Graphics)

- **Stupně šedi:** Podporuje obrázky pouze v odstínech šedi, vhodné pro jednoduchou grafiku.
- **Barevná paleta:** Podporuje paletové obrázky s až 24bitovou barevnou hloubkou (16,7 milionů barev).
- **RGBA (32bpp):** Umožňuje ukládání plnobarevných obrázků s průhledností, kde každý pixel obsahuje 8 bitů pro červenou, zelenou, modrou a alfa kanál.
- **Až 16 bitů na kanál:** Podporuje vysokou barevnou přesnost pro profesionální grafiku.
- **DEFLATE komprese:** Používá bezeztrátovou metodu komprese, která efektivně redukuje velikost souboru bez ztráty kvality.

13.5 JFIF (JPEG File Interchange Format)

- **RGB (24bpp):** Ukládá plnobarevné obrázky s 8 bity pro červenou, zelenou a modrou složku (16,7 milionů barev).
- **JPEG komprese:** Používá ztrátovou kompresi pro výrazné snížení velikosti souboru při zachování přijatelné kvality.
- **EXIF metadata:** Umožňuje ukládat dodatečné informace o obrázku, například parametry fotoaparátu a čas pořízení.

14 Vektorová grafika

- **Nezávislost na rozlišení:** Kvalita obrazu zůstává perfektní při libovolném zvětšení nebo zmenšení, protože data jsou definována matematickými rovnicemi, nikoli pixely.
- **Základní grafické prvky:**
  - **Úsečky, obdélníky, elipsy:** Jednoduché geometrické tvary.
  - **Bézierovy křivky:** Křivky, které umožňují vytvářet hladké zakřivené linie pomocí několika bodů. Tyto body určují tvar a směr křivky, což je užitečné například při kreslení písmen nebo složitějších tvarů.
  - **Text:** Podpora přesného vykreslování textu s možností úprav.
  - **Barevné přechody:** Plynulé přechody mezi dvěma nebo více barvami.
  - **Rastrové textury:** Možnost začlenění rastrových prvků pro složitější detaily.

14.1 SVG (Scalable Vector Graphics)

- **Specifikováno W3C**: SVG je standardizováno organizací World Wide Web Consortium (W3C), což zajišťuje interoperabilitu a širokou podporu mezi různými technologiemi a zařízeními.
- **XML formát**: SVG používá formát XML, který umožňuje popis vektorové grafiky ve formě textu. To umožňuje snadnou editaci, čitelnost pro člověka a možnost generování nebo manipulace pomocí programů.
- **Podpora filtrů a animací**: SVG umožňuje vytvářet pokročilé grafické efekty, jako jsou filtry (např. rozostření, stíny) a animace přímo integrované do souboru. Tyto animace mohou být definovány pomocí atributů SVG nebo prostřednictvím JavaScriptu.
- **CSS stylování**: Grafiku vytvořenou ve formátu SVG lze stylovat pomocí CSS, podobně jako HTML prvky. To poskytuje flexibilitu při změnách vzhledu bez úprav samotného SVG kódu.
- **Dobrá podpora v prohlížečích**: Většina moderních webových prohlížečů plně podporuje SVG, což zajišťuje spolehlivé zobrazení a kompatibilitu napříč platformami.

15 Video formáty

15.1 Multimediální kontejnery

- **Příklady formátů**: Mezi běžné multimediální kontejnery patří `AVI`, `MOV`, `ASF`, `Matroska (MKV)` a `RealMedia (RM)`. Každý z nich má specifické vlastnosti a oblasti využití, například `MKV` je známý svou flexibilitou a podporou více proudů a titulků.
- **Obsah různých druhů proudů**: Multimediální kontejnery mohou obsahovat různé typy datových proudů, například video (např. ve formátu H.264, VP9) a audio (např. AAC, MP3). Tyto proudy jsou v kontejneru organizovány a synchronizovány.
- **Dodatečné informace**: Kontejnery mohou kromě video a audio dat zahrnovat i další typy obsahu, jako jsou titulky (ve formátech SRT, ASS), informace o kapitolách (pro rychlou navigaci) a metadata (například informace o autorovi, názvu nebo roku vytvoření).

15.2 Komprese videa

- **Ztrátová a bezeztrátová komprese**:
  - *Ztrátová komprese*: Snižuje objem dat odstraněním méně viditelných nebo redundantních informací, což vede k menší velikosti souboru, ale za cenu drobné ztráty kvality (např. H.264, H.265).
  - *Bezeztrátová komprese*: Zachovává původní data v plné kvalitě, ale nabízí nižší úroveň redukce velikosti souboru (např. FFV1, Motion JPEG 2000).
- **DCT nebo JPEG komprese**: Discrete Cosine Transform (DCT) je základní metoda používaná v mnoha video kodecích, kde převádí obrazová data do frekvenční oblasti a redukuje informace méně důležité pro lidské vnímání. Tato technika je také klíčová pro JPEG kompresi obrázků.
- **Makrobloky**: Video je při kompresi rozděleno na malé části zvané makrobloky, obvykle o velikosti až 16×16 pixelů. Každý makroblok je zpracován samostatně, což zjednodušuje kompresi a zlepšuje efektivitu při práci s pohybem v obraze.
- **Časoprostorová korespondence**: Komprese videa využívá podobnosti mezi sousedními snímky (časová redundance) a uvnitř jednoho snímku (prostorová redundance) k redukci množství uložených dat.
- **IPB snímky**:
  - *I-snímky (Intra-coded Frames)*: Obsahují kompletní obrazová data a mohou být dekódovány samostatně.
  - *P-snímky (Predicted Frames)*: Ukládají rozdíly oproti předchozím snímkům, což šetří prostor.
  - *B-snímky (Bi-directionally Predicted Frames)*: Využívají informace jak z předchozích, tak z následujících snímků, což zajišťuje ještě vyšší kompresi.

15.3 WebM

- **Vytvořeno Googlem (2010)**: Formát WebM byl představen společností Google v roce 2010 jako otevřený, bezplatný standard pro efektivní streamování a přehrávání videí na webu.
- **Kontejner Matroska**: WebM používá jako základ kontejnerový formát Matroska, což umožňuje snadné zpracování různých typů multimediálního obsahu v jednom souboru.
- **Kodeky VP8, VP9**: Pro kompresi videa WebM využívá kodeky VP8 a VP9, které jsou optimalizované pro nízké datové toky a vysokou kvalitu obrazu. VP9 navíc poskytuje ještě lepší kompresní poměr než jeho předchůdce.
- **Zvuk: Vorbis**: Zvuková stopa ve WebM je komprimována pomocí kodeku Vorbis, který nabízí kvalitní zvuk při nízkém datovém toku. Novější verze mohou také podporovat kodek Opus.
- **Podpora v prohlížečích**: Formát WebM je nativně podporován v moderních webových prohlížečích, jako jsou Google Chrome a Mozilla Firefox, což z něj činí ideální volbu pro přehrávání videí přímo na webu bez nutnosti dodatečných pluginů.

16 Omezování barevného prostoru

Proces omezování barevného prostoru se zaměřuje na redukci počtu barev v obraze při zachování co nejpodobnějšího vizuálního dojmu. Tento postup se často využívá v grafice a tisku, kde je potřeba pracovat s omezenou barevnou škálou.

- **Použití barevné palety s omezenou velikostí**: Místo použití plného spektra barev (např. RGB s 16,7 miliony barev) se obraz převádí do omezené palety, která obsahuje pouze několik stovek nebo tisíců barev. Tato technika je využívána například ve formátech GIF nebo při optimalizaci obrázků pro web.

- **Filtry v grafických programech**: Grafické programy (např. Adobe Photoshop, GIMP) nabízejí filtry a nástroje pro snížení počtu barev v obraze. Tyto filtry umožňují nastavit úroveň barevné redukce a optimalizovat obraz pro konkrétní účel.
- **Barevný i černobílý tisk**: Omezení barevného prostoru je klíčové při přípravě obrazů pro tisk, kde se často využívají specifické palety (např. CMYK pro barevný tisk nebo stupně šedi pro černobílý tisk).

17 RGB na stupně šedi

Zjednodušení obrazových dat, kdy jsou zachovány pouze informace o jasu. Existují různé metody převodu RGB na stupně šedi, přičemž každá metoda má jiné zaměření na váhu jednotlivých složek.

17.1 RGB na stupně šedi (YUV)

Tato metoda odpovídá standardu BT.601, který je často používán v televizním vysílání. Klade důraz na vnímání jasu lidským okem, přičemž největší váha je přikládána zelené složce (G). Výpočet:

$$Y = 0.299R + 0.587G + 0.114B$$

17.2 RGB na stupně šedi (Y'IQ)

Tento přístup splňuje standard BT.709, který je využíván zejména v moderním digitálním videu, jako je HDTV. Vzorec:

$$Y = 0.2126R + 0.7152G + 0.0722B$$

17.3 RGB na stupně šedi (HSV)

Metoda HSV (Hue, Saturation, Value) definuje jas jednoduše jako maximální hodnotu ze tří složek (R, G, B):

$$V = \max(R, G, B)$$

Interpretace: Nejsvětlejší složka v daném bodě obrazu - rychlý výpočet, ale nezohledňuje lidské vnímání jasu. Je vhodná pro aplikace, kde není nutná přesná reprezentace světelnosti.

18 Omezení barevného prostoru černobílých obrázků

Techniky, které převádějí obrazy se stupni šedi na obrázky s omezeným počtem hodnot (např. jen černá a bílá).

18.1 Prahování

Prahování je základní a jednoduchá metoda, která převádí obraz na černobílý na základě porovnání s pevně stanovenou prahovou hodnotou (*threshold*).

```
for V_in in pixels.values():
    if V_in > threshold:
        V_out = max
    else:
        V_out = min
```

18.2 Náhodný rozptyl

Náhodný rozptyl (*random dithering*) přidává do obrazu náhodný šum, aby výsledný černobílý obraz více odpovídal původní předloze.

```
for V_in in pixels.values():
    if V_in > random(min, max):
        V_out = max
    else:
        V_out = min
```

Princip: Pixel je porovnáván s náhodnou hodnotou generovanou v rozsahu min a max možné hodnoty.

18.3 Maticový rozptyl

Maticový rozptyl (*ordered dithering*) nahrazuje hodnoty každého pixelu předem definovanou maticí.

- Metoda zvětšuje výslednou velikost obrazu podle velikosti použité matice.

Příklad matice vhodné pro displeje: Matice M_{2n} je definována rekurzivním vztahem:

$$M_{(2n)} = \begin{bmatrix} 4M_{(n)} & 4M_{(n)} + 3J_{(n)} \\ 4M_{(n)} + 2J_{(n)} & 4M_{(n)} + J_{(n)} \end{bmatrix}$$

Interpretace:

- $M_{(n)}$ je matice pro nižší rozlišení (např. $M_{(1)} = \begin{bmatrix} 0 \end{bmatrix}$).
- $J_{(n)}$ je matice obsahující jednotkové hodnoty o rozměru $n \times n$, která posouvá hodnoty pro rozlišení různých úrovní jasu.
- Rekurzivní vztah umožňuje generovat stále detailnější matice podle potřeby výsledné kvality a rozlišení.

19 Geometrické transformace

Pro 2D transformace používáme homogenní souřadnice s maticí:

$$\begin{bmatrix} X' \\ Y' \\ w' \end{bmatrix} = \begin{bmatrix} m_{1,1} & m_{1,2} & m_{1,3} \\ m_{2,1} & m_{2,2} & m_{2,3} \\ m_{3,1} & m_{3,2} & m_{3,3} \end{bmatrix} \begin{bmatrix} X \\ Y \\ w \end{bmatrix}$$

19.1 Posun

$$\begin{bmatrix} X' \\ Y' \\ w' \end{bmatrix} = \begin{bmatrix} 1 & 0 & dx \\ 0 & 1 & dy \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ w \end{bmatrix}$$

19.2 Zvětšení/zmenšení

$$\begin{bmatrix} X' \\ Y' \\ w' \end{bmatrix} = \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ w \end{bmatrix}$$

19.3 Otočení

$$\begin{bmatrix} X' \\ Y' \\ w' \end{bmatrix} = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ w \end{bmatrix}$$

20 Interpolace

20.1 Lineární interpolace

Lineární interpolace je metoda používaná pro odhadování cesty mezi dvěma známými body.

Příklad: Pokud znáte dvě hodnoty, například teplotu v 9:00 a 10:00, můžete odhadnout teplotu v 9:30. Předpokládá se, že změna teploty mezi těmito časy probíhá rovnoměrně.

20.2 Bilineární interpolace

Bilineární interpolace je dvourozměrné rozšíření lineární interpolace, používané například při zvětšování obrázků nebo práci s 2D datovými poli. Využívá hodnoty čtyř bodů, které tvoří rohy obdélníku kolem hledaného bodu.

Prakticky to znamená, že když zvětším obrázek tak se mi mezi pixely udělají "přechody". Nebo když grafik chce udělat přechod mezi barvama tak si tam navolí dvě barvy a ty mezi nimi se vytvoří samy.

Proces probíhá ve dvou krocích:

- Lineární interpolace mezi dvojicemi bodů podél jednoho směru (např. vodorovně).
- Interpolace podél druhého směru (např. svisle) s použitím výsledků z prvního kroku.

Příklad: Při zvětšování obrázku se bilineární interpolací určuje barva pixelu mezi čtyřmi sousedními pixely. Výsledná hodnota je vážený průměr těchto pixelů, což vytváří hladší přechody mezi barvami.

21 Alfa míchání

Alfa míchání je technika, která se používá pro kombinování dvou barev při zohlednění jejich průhlednosti (alfa kanálu). Alfa kanál má hodnoty v rozsahu $\langle 0, 1 \rangle$. Pomocí alfa míchání lze určovat, jak moc budou jednotlivé barvy ovlivněny při jejich kombinaci.

21.1 Alpha Over

Alpha Over je způsob míchání, při kterém se výsledná barva skládá z přední (foreground) a zadní (background) barvy. Pokud je alfa hodnota přední barvy 1, znamená to, že je plně neprůhledná a pozadí bude zakryté. Pokud je alfa hodnota menší než 1, pozadí bude stále viditelné.

21.2 Prolínačka

Prolínačka = plynulý přechod mezi dvěma barvami v čase. Hodnota alfa se postupně mění mezi počáteční a konečnou barvou, což umožňuje hladký přechod mezi těmito barvami.

21.3 Klíčování (Chroma Keying)

Odstranění určité barvy z obrazu, obvykle zelené barvy (často používané v televizní produkci) a její náhradě jiným pozadím. Pokud je hodnota zelené barvy příliš vysoká, alfa hodnota bude nastavena tak, aby část obrazu byla průhledná a nahrazena pozadím.

22 Histogram

Histogram je grafické znázornění rozdělení hodnot v obrázku, konkrétně pro jednotlivé barevné kanály.

22.1 Vytvoření histogramu pro jeden barevný kanál

Pro vytvoření histogramu pro jeden barevný kanál se postupuje takto:

- Nejprve se inicializuje "pole" = histogram, kde každé možné hodnotě (od 0 do N) přiřadíme počáteční hodnotu 0.

- Poté procházíme všechny hodnoty pixelů v obrázku a inkrementujeme příslušný index barvy v histogramu podle barvy pixelu.

Příklad kódu pro vytvoření histogramu pro jeden kanál:

```
for i in range(0, N):
    hist[i] = 0

for V_in in pixels.values():
    hist[V_in] += 1
```

23 Reprezentace 3D objektů

23.1 Základní dělení

Reprezentace 3D objektů se dělí na dvě hlavní kategorie:

- **Hraniční reprezentace** (zachycuje pouze povrch objektu):
  - *Plošková reprezentace*: Popisuje objekt pomocí vrcholů, hran a plošek.
  - *Bodová reprezentace*: Povrch tvořen jednotlivými body.
  - *Konstruktivní geometrie těles (CSG)*: Kombinace primitivních tvarů pomocí geometrických operací.
- **Objemová reprezentace** (zachycuje celý objem objektu, např. voxelové modely).

23.2 Plošková reprezentace

Nejrozšířenější způsob ukládání 3D objektů.

- **Vertexy**: 3D souřadnice vrcholů.
- **Hrany**: Spojení dvou vrcholů.
- **Plošky**: Spojení tří nebo více vrcholů.

23.3 Trojúhelník & N-úhelník

Plošky složené z libovolných n-úhelníků se převádějí na trojúhelníky díky jejich výhodám:

- Všechny body trojúhelníku leží v jedné rovině.
- Trojúhelník je vždy konvexní.
- Normálový vektor lze jednoznačně vypočítat z vrcholů.
- Snadná rasterizace (zobrazení na obrazovce).

23.4 Síť trojúhelníků

Síť trojúhelníků obsahuje:

- **Geometrickou strukturu**: Ukládá pouze souřadnice vrcholů.
- **Topologickou strukturu**: Ukládá informace o propojení vrcholů do hran a plošek.

23.5 Manifoldní a non-manifoldní objekty

Manifoldní objekty splňují pravidla, která zaručují, že jsou vyrobitelné a správně definované:

1. Každá hrana je sdílena přesně dvěma ploškami.
2. Žádná hrana neprotíná jinou plošku.
3. Žádný vrchol nespojuje dvě nesouvisející části objektu.

23.6 Datová struktura okřídlené hrany

Tato struktura je vhodná pro dynamické úpravy geometrie a topologie. Používá se pouze pro manifoldní objekty. Obsahuje:

- **Geometrii**: Informace o vrcholech.
- **Topologii**: Informace uložené u hran:
  - Odkazy na oba vrcholy hrany.
  - Odkazy na dvě sousedící plošky.
  - Odkazy na čtyři sousední hrany.
- Každá ploška obsahuje odkaz na jednu ze svých hran.

23.7 Vlastnosti vertexů, hran, plošek a rohů plošek

- **Vertexy**:
  - Váha (např. pro kinematiku).
  - Barva.
- **Hrany**:
  - Ostrost.
  - UV mapování.
- **Plošky**:
  - Typ stínování.
  - Textura a barva.
- **Rohy plošek**:
  - UV souřadnice.
  - Barva.

23.8 Transformace objektů

Pro transformace se používají homogenní souřadnice a transformační matice:

$$X' = \frac{X}{w}, Y' = \frac{Y}{w}, Z' = \frac{Z}{w}$$

23.8.1 Posun

Matice pro posun (dx, dy, dz):

$$\begin{bmatrix} 1 & 0 & 0 & dx \\ 0 & 1 & 0 & dy \\ 0 & 0 & 1 & dz \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

23.8.2 Zvětšení/zmenšení

Matice pro změnu měřítka:

$$\begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

24 Zobrazování prostorových dat

24.1 Zobrazení

- Zobrazení prostorových dat znamená převod 3D dat uložených v paměti počítače na 2D rastrový obrázek. Tento proces se nazývá **rendering**.
- Rendering může být:
  - **Real-time** – například hry
  - **Fotorealistický** – například Blender render

24.2 Zobrazovací řetězec

1. **První část:** Geometrická transformace
   - Orientace scény podle kamery.
   - Ořezání pohledovým objemem – odstranění částí mimo zorné pole.
   - Promítání – převod 3D souřadnic na 2D.
   - Transformace do souřadnic okna obrazovky.
2. **Druhá část:** Rendering
   - Aplikace lokálního osvětlení.
   - Rasterizace – převod vektorových dat na rastrový obraz.
   - Mapování textur – přidání detailů na povrchy.
   - Určení viditelnosti pixelů – zajištění správného překrývání objektů.

24.3 Orientace podle kamery

- Usnadňuje promítání transformací souřadnic tak, aby kamera byla v počátku soustavy souřadnic a směřovala ve směru osy $-Z$.
- Používá transformační matici T_{cam} k převodu souřadnic z *World Coordinate System* (WCS) do *View Coordinate System* (VCS).

24.3.1 Transformační matice

Transformační matice T_{cam} má následující podobu:

$$T_{cam} = \begin{bmatrix} p_x & p_y & p_z & 0 \\ h_x & h_y & h_z & 0 \\ -l_x & -l_y & -l_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

24.4 Další kroky renderingu

- **Osvětlovací model:** Vypočítává osvětlení povrchu pomocí zvoleného modelu (např. Phongův model).
- **Rasterizace:** Převádí vektorová data (body, čáry, polygony) na rastrový obrázek.
- **Mapování textur:** Používá UV souřadnice k přidání detailů na povrchy. Každému vrcholu polygony je přiřazena souřadnice v textuře.
- **Určení viditelnosti pixelů:**
  - Využívá *Z-buffer*, který uchovává vzdálenost fragmentů od kamery.
  - Fragment s nejmenší vzdáleností určuje výslednou barvu pixelu.

25 OpenGL

25.1 Historie

- OpenGL bylo vyvinuto firmou Silicon Graphics (SGI) v 90. letech jako API pro grafické akcelerátory.
- Konkuruje Direct3D od Microsoftu.
- Specifikaci spravuje Khronos Group; aktuální verzí je 4.6 z roku 2017.
- Varianty:
  - WebGL (pro webové prohlížeče)
  - OpenGL ES (pro mobilní zařízení)
- Vulkan je jeho modernější a výkonnější nástupce.

25.2 Základní vlastnosti

- Umožňuje akceleraci 2D a 3D grafiky.
- Je multiplatformní a nezávislé na operačním systému, grafickém akcelerátoru nebo správci oken.
- Neposkytuje přímou podporu pro práci s okny, uživatelským rozhraním, fonty ani událostmi – pro tyto funkce se využívají knihovny jako GLUT nebo GLFW.
- OpenGL je kompatibilní s mnoha programovacími jazyky a je součástí ovladačů grafické karty.
- Používá stavový automat a vlastní datové typy (např. `GLint`, `GLdouble`).
- Podporuje základní geometrická primitiva, jako body, úsečky, trojúhelníky nebo polygony.

25.3 Syntaxe funkcí

- Funkce mají prefix `gl`, např. `glVertex3f(1.0, 0.0, 1.0);`.

25.4 Základní primitiva

- Typy: `GL_POINTS`, `GL_LINES`, `GL_TRIANGLES`, `GL_POLYGON` atd.
- Vlastnosti primitiv lze ovlivnit funkcemi jako:
  - `glColor4f()` – barva
  - `glPointSize()` – velikost bodu
  - `glLineWidth()` – šířka čáry
  - `glPolygonMode()` – zobrazení ploch

25.5 Transformační matice

- Typy:
  - `GL_MODELVIEW` – transformace objektů a kamery
  - `GL_PROJECTION` – projekce na obrazovku
  - `GL_TEXTURE` – mapování textur
- Funkce: `glLoadIdentity()`, `glTranslate()`, `glScale()`, `glRotate()`.

25.6 Framebuffer

- Složky framebufferu:
  - Color buffer – barvy
  - Depth buffer – hloubka
  - Stencil buffer – maskování
  - Accumulation buffer – efekty jako motion blur

25.7 Kamera

- Typy projekce:
  - Ortogonální – rovinné promítání
  - Perspektivní – realistické zobrazení
- Pro práci s kamerou lze využít knihovnu GLU.

25.8 Vertex Arrays a VBO

- **Vertex Arrays**: efektivnější než tradiční display listy.
- **VBO (Vertex Buffer Objects)**:
  - Data jsou ukládána přímo do paměti GPU.
  - Vhodné pro statická data, protože snižují přenos dat mezi CPU a GPU.

25.9 Osvětlení

- OpenGL podporuje bodové světelné zdroje a reflektory.
- Používá Phongův osvětlovací model, který kombinuje ambientní, difuzní a spekulární složky.
- Materiály jsou definovány parametry:
  - `Ambient` – základní osvětlení
  - `Diffuse` – odraz od povrchu
  - `Specular` – lesk
  - `Shininess` – ostrost lesku

25.10 Texturování

- Textury lze nastavovat parametry jako:
  - `Wrap mode` – způsob opakování textury (`GL_REPEAT`).
  - `Filtry` – kvalita zobrazení (`GL_NEAREST`, `GL_LINEAR`).
  - UV souřadnice – určení polohy textury na objektu.

26 Historie: Web & 3D

26.1 Předchůdci WebGL

- **VRML (Virtual Reality Modeling Language)**:
  - Značkovací jazyk založený na XML pro 3D scény.
  - Vyžadoval instalaci pluginu, což komplikovalo jeho použití.
  - Umožňoval převážně statické scény bez interaktivních prvků.

- **Blender Player**:
  - Platforma pro interaktivní aplikace a hry.
  - Bylo nutné instalovat plugin pro použití.
  - Nabízel plnou interaktivitu a podporoval skriptování v Pythonu.
  - Byl omezený na použití s Blenderem.
- **Flash**:
  - Používal se pro interaktivní obsah, včetně 3D grafiky.
  - Vyžadoval instalaci Adobe Flash pluginu.

27 HTML5 & WebGL

- **WebGL**: Standard pro vykreslování 3D grafiky přímo v prohlížeči, založený na OpenGL ES 2.0.
- **Standardizace**: Vytvářena a udržována skupinou Khronos Group.
- **Podpora**: Funguje ve všech hlavních prohlížečích, roku 2016 měl WebGL podporu u 92 % uživatelů.

28 Zobrazovací řetězec WebGL

- Na rozdíl od OpenGL nepoužívá pevně daný zobrazovací řetězec.
- Vyžaduje použití **vertex shaderů** a **fragment shaderů**.

29 Vertex & Fragment Shader

29.1 Společné vlastnosti

- Programovací jazyk GLSL, podobný C.
- Kód je kompilován a vykonáván na grafické kartě.

29.2 Vertex Shader

- Transformuje jednotlivé vrcholy scény.
- Prováděné operace:
  - Geometrické transformace (rotace, zvětšení, posun).
  - Převod do souřadného systému kamery.
  - Projekce (perspektivní nebo rovinná).
- Výstupy: `gl_Position`, `gl_PointSize`.

29.3 Fragment Shader

- Aplikuje se na jednotlivé fragmenty (pixely) scény.
- Výsledek je ukládán do framebufferu.

30 Datové typy v shaderech

- Základní: `void`, `bool`, `int`, `float`.
- Vektory: `vec2`, `vec3`, `vec4` (pro `float`); `bvec2`, `bvec3`, `bvec4` (pro `bool`); `ivec2`, `ivec3`, `ivec4` (pro `int`).
- Matice: `mat2`, `mat3`, `mat4`.
- Textury: `sampler2D`, `samplerCube`.

31 Buffer management

31.1 Double-buffering

- Automaticky implementován ve WebGL.
- **Přední buffer**: Používá se pro zobrazování na obrazovce.
- **Zadní buffer**: Používá se pro vykreslování scény.
- Eliminuje problémy jako problikávání a vykreslovací artefakty.

31.2 Z-Buffer

- Řeší viditelnost objektů na základě jejich vzdálenosti od kamery.
- Uchovává hodnoty hloubky fragmentů.
- Aktivace pomocí `gl.enable(gl.DEPTH_TEST)`.

32 Vstupní data

32.1 ArrayBuffer & typová pole

- Používají se pro přenos dat mezi JavaScriptem a shadery.
- Typová pole z HTML5:
  - `Uint8Array`, `Uint16Array`, `Uint32Array`.
  - `Int8Array`, `Int16Array`, `Int32Array`.
  - `Float32Array`, `Float64Array`.

33 Proměnné v shaderech

33.1 Varying proměnné

- Používají se pro předávání hodnot mezi vertex a fragment shaderem.
- Hodnoty jsou lineárně interpolovány během rasterizace.

33.2 Uniform proměnné

- Konstantní během vykonávání shaderu.
- Často se používají pro předávání matic a dalších globálních dat.

34 Three.js

34.1 Základní informace

- Javascriptová knihovna pro implementaci 3D grafiky na webu
- Není nutná podpora WebGL (lze renderovat do canvasu pomocí SVG)
- Kód aplikace psán čistě v Javascriptu (bez nutnosti psát Vertex/Fragment shadery)

34.2 Důležité vlastnosti

- Hierarchická scéna
- Kamera s perspektivním i rovinným promítáním
- Světla bodová, směrová, kuželová a ambientní
- Objekty včetně základních primitiv
- Materiály pomocí předefinovaných i vlastních shaderů
- Textury

34.3 Renderer

- Detekce podpory WebGL pomocí Detector.js
- Možnosti rendereru:
  - WebGL renderer: `new THREE.WebGLRenderer()`
  - Softwarový renderer: `new THREE.CanvasRenderer()`
- Nastavení velikosti canvasu: `renderer.setSize(width, height)`

34.4 Kamera

34.4.1 Perspektivní kamera

- Parametry:
  - Úhel pohledu (ohnisková vzdálenost)
  - Poměr stran
  - Vzdálená a blízká ořezová plocha
- Vytvoření: `new THREE.PerspectiveCamera(view_angle, aspect, near, far)`

34.4.2 Ortogonální kamera

- Vytvoření: `new THREE.OrthographicCamera(left, right, top, bottom, near, far)`

34.5 Scéna

- Objekty organizovány do stromové struktury
- Vytvoření: `var scene = new THREE.Scene()`
- Přidání objektů: `scene.add(object)`

34.6 Objekty

34.6.1 Základní vytvoření

- Geometrie: `new THREE.BoxGeometry(1.0, 1.0, 1.0)`
- Materiál: `new THREE.MeshBasicMaterial({color: 0x00aa00})`
- Objekt: `new THREE.Mesh(geometry, material)`

34.6.2 Obecná geometrie

- Vytvoření: `new THREE.Geometry()`
- Přidání vertexů: `geometry.vertices.push(new THREE.Vector3(x, y, z))`
- Spojení do plošek: `geometry.faces.push(new THREE.Face3(0, 1, 2))`

34.7 Animace

```
var render = function () {
    requestAnimationFrame(render);
    // Změny objektů
    renderer.render(scene, camera);
};
render();
```

34.8 Osvětlení a materiály

- Barva pozadí: `renderer.setClearColorHex(0x333F47, 1)`
- Bodové světlo:

  ```
  var light = new THREE.PointLight(0xffffff);
  light.position.set(-100,200,100);
  scene.add(light);
  ```

- Materiály:
  - Phongův: `new THREE.MeshPhongMaterial({color: 0x00aa00})`
  - Lambertův: `new THREE.MeshLambertMaterial({color: 0x00aa00})`

34.9 Textury

- Loader textur: `new THREE.TextureLoader()`
- Načtení textury pomocí callback funkce
- Možnost sledování průběhu načítání

35 Parametrické křivky

35.1 Motivace

Parametrické křivky (=křivky reprezentované funkcí) se často používají v různých oblastech, například:

- Při tvorbě fontů (písma jsou často definována pomocí křivek).
- V animacích, kde křivky popisují pohyby (tzv. animační křivky).
- V grafickém designu, například pro návrh log, ikonek nebo ilustrací.

35.2 Způsoby vyjádření křivek

Existují tři hlavní způsoby, jak lze definovat křivky:

- **Explicitní vyjádření** (funkční závislost):

$$y = f(x)$$

- **Implicitní vyjádření**:

$$F(x, y) = 0$$

- **Parametrické vyjádření**:

$$\begin{aligned} x &= x(t), \\ y &= y(t), \quad 0 \leq t \leq 1, \\ z &= z(t) \end{aligned}$$

  Tento způsob je velmi flexibilní, protože umožňuje popsat složité tvary a pohyby.

35.3 Tečný vektor

- **Co to je:** Tečný vektor ukazuje směr křivky v konkrétním bodě.
- **Jak se určuje:** Tečný vektor je definován jako změna pozice křivky v daném bodě.
- **Jednoduché shrnutí:** Tečný vektor popisuje, jak křivka "směřuje" v bodě $Q(t_0)$. Je to vlastně směr, kterým křivka pokračuje v tomto bodě.

35.4 Spojitost

Spojitost křivek lze rozdělit do dvou typů:

- **Parametrická spojitost** C^n: Křivka má derivace až do řádu n v každém bodě.
- **Geometrická spojitost** G^n: Dvě křivky Q_1 a Q_2 jsou spojité, pokud platí:

$$q_1^{(n)} = kq_2^{(n)}, \quad k > 0$$

  kde k je kladná konstanta.

35.5 Polynomiální křivky

Polynomiální křivka je vyjádřena jako:

$$Q(t) = a_0 + a_1 t + a_2 t^2 + \cdots + a_n t^n.$$

36 Bézierovy křivky

- Křivka začíná v prvním a končí v posledním řídícím bodě.
- Leží uvnitř ohraničujícího polygonu určeného řídícími body.
- Změna polohy jednoho řídícího bodu ovlivní celou křivku.
- Nejčastěji se používají kubické Bézierovy křivky, které se snadno navazují.

36.1 Tečné vektory v koncových bodech

- **Co to je:** Tečný vektor ukazuje směr, kterým křivka pokračuje v krajním bodě.
- **Jak se určuje:**
  - Tečný vektor na začátku křivky závisí na prvních dvou řídících bodech.
  - Tečný vektor na konci křivky závisí na posledních dvou řídících bodech.
- **Jednoduché shrnutí:** Tečné vektory určují, jak křivka "startuje" a "končí". Jsou odvozeny z řídících bodů, které určují tvar křivky.

37 Coonsovy kubiky

- Coonsovy křivky jsou parametrické křivky používané v počítačové grafice a modelování.
- Slouží k vytvoření hladkých ploch definovaných pomocí dvou hraničních křivek a případně dalších řídících funkcí.
- Zajišťují, že výsledná plocha přesně prochází danými hraničními křivkami.
- Umožňují přesné modelování tvarů a přechodů mezi plochami.

38 NURBS

- **NURBS** (Non-Uniform Rational B-Splines) jsou matematický model používaný v počítačové grafice a CAD.
- Umožňují reprezentovat křivky a plochy pomocí parametrických rovnic.
- **Vlastnosti:**
  - Podporují přesnou reprezentaci jednoduchých tvarů (např. kružnic, elips) i složitých volnoplošných křivek.
  - Jsou definovány pomocí kontrolních bodů, vah a uzlového vektoru.
  - Nabízejí flexibilitu díky možnosti měnit hladkost a tvar bez zvýšení počtu bodů.
- Používají se ve 3D modelování, animacích, simulacích a průmyslovém designu.

39 Parametrické plochy

39.1 Motivace

Parametrické plochy jsou klíčovým konceptem v mnoha oblastech:

- **CAD/CAM aplikace:** Používají se k modelování povrchů pro výrobu a design.
- **Grafický design:** Pomáhají při tvorbě složitých tvarů a povrchů, jako jsou objekty ve 3D grafice.
- **Parametrické plochy:**
  - Plocha je definována pomocí parametrů u, v, které určují každý bod na ploše.
  - Bod na ploše je dán funkcí $\mathbf{r}(u, v) = [x(u, v), y(u, v), z(u, v)]$.
  - Příklady: koule, válec nebo složité 3D tvary.
- **Tečné vektory:**
  - Popisují směr plochy v bodě.

  - Jsou to derivace $\mathbf{r}_u = \frac{\partial \mathbf{r}}{\partial u}$ a $\mathbf{r}_v = \frac{\partial \mathbf{r}}{\partial v}$.
  - Určují tečnou rovinu k ploše v daném bodě.

- **Normálový vektor:**
  - Je kolmý k ploše v bodě.
  - Vypočítá se jako vektorový součin $\mathbf{n} = \mathbf{r}_u \times \mathbf{r}_v$.
  - Používá se např. pro výpočet osvětlení a stínů.

- **Hlavní křivky:**
  - Křivky, kde je plocha nejvíce nebo nejméně zakřivená.
  - Zakřivení v těchto směrech určuje geometrii plochy.
  - Důležité při analýze tvaru nebo konstrukci.

40 Bézierovy plochy

- **Co to je:** Bézierova plocha je hladká plocha definovaná řídícími body. Používá se v grafice a designu pro modelování povrchů.

- **Vlastnosti:**
  - Hrany plochy jsou Bézierovy křivky.
  - Plocha je uvnitř konvexní obálky řídících bodů.
  - Pohyb jednoho bodu mění celou plochu.
  - Prochází krajními řídícími body.

41 B-spline plochy

- **Co to je:** B-spline plocha je obecnější varianta Bézierovy plochy, která umožňuje větší kontrolu nad tvarem.

- **Vlastnosti:**
  - Hlavní křivky plochy jsou B-spline křivky.
  - Nabízí plynulejší přechody mezi plochami.
  - Změna bodu ovlivní jen blízkou oblast plochy.
  - Plocha zůstává uvnitř konvexní obálky řídících bodů.

42 NURBs plochy

- **Co to je:** NURBs (Non-Uniform Rational B-Splines) jsou zobecnění B-spline ploch, které zavádějí váhy pro větší flexibilitu.

- **Vlastnosti:**
  - Umožňují přesné modelování kruhů, elips a dalších složitých tvarů.
  - Váhy řídí, jak moc každý bod ovlivňuje plochu.
  - Jsou velmi přesné a univerzální pro design a animace.

43 Druhy animace

- Animace se dělí na interaktivní (např. simulátory, hry) a neinteraktivní (2D/3D video sekvence jako filmy nebo vizualizace).

- Lze ji rozlišit na nízkoúrovňovou (např. klíčování) a vysokoúrovňovou (např. simulace davů).

43.1 Story board

- Storyboard je vizuální forma scénáře, která usnadňuje představu o výsledném díle.
- Slouží jako nástroj pro komunikaci mezi umělci, tvůrci a programátory.

43.2 Stop motion

- Stop motion je technika animace, při které jsou loutky nebo objekty snímány po jednotlivých snímcích.
- Vyžaduje důkladnou přípravu a trpělivost, často bez přímého zapojení počítačové grafiky.

44 Obnovovací frekvence (Frame Rate)

- Obnovovací frekvence udává, kolikrát za sekundu je zobrazen snímek animace, a měří se v FPS (Frames Per Second).
- Typické hodnoty jsou 24, 25, 30, 50 nebo 60 FPS.
- Prokládaná video sekvence přenáší obraz s poloviční frekvencí: každý snímek obsahuje liché a sudé řádky odděleně. Tato technika byla využívána při televizním vysílání.

45 Keying alias Klíčování

- Klíčování vzniklo ve 20. letech 20. století ve studiu Walta Disneyho.
- Základ animace je tvořen klíčovými snímky, mezi nimiž byly dříve mezisnímky kresleny méně zkušenými animátory.
- V moderní animaci tyto mezisnímky vytváří automaticky počítač.

45.1 Animování rastrové grafiky

- Animace rastrové grafiky se podobá stop motion.
- Pokud software umožňuje převod ručně kreslené čáry na hladkou křivku, lze použít klíčování a interpolaci tvaru.

45.2 Interpolace parametrů

- **Co to je:** Interpolace určuje hodnoty mezi klíčovými snímky (např. mezi dvěma pozicemi objektu).
- **Typy interpolace:**
  - Konstantní – hodnota zůstává stejná jako v posledním klíčovém snímku.
  - Lineární – hodnoty se mění rovnoměrně mezi klíčovými snímky.
  - Polynomická – používá hladší přechody, často pomocí Bézierových křivek.

45.3 Extrapolace parametrů

- **Co to je:** Extrapolace určuje hodnoty mimo rozsah klíčových snímků (před prvním nebo za posledním).
- **Typy extrapolace:**
  - Konstantní – hodnota zůstává stejná jako v prvním nebo posledním klíčovém snímku.
  - Přímka – hodnota pokračuje v trendu podle směru tečného vektoru.
  - Cyklické opakování – hodnoty se pravidelně opakují podle klíčových stavů.

46 Animace postav

- Postava je obvykle reprezentována jako 3D objekt s ploškovou geometrií a vnitřní hierarchickou kostrou.
- Kostra obsahuje kosti, mezi kterými mohou být vztahy rodič-potomek. Konečná kost bez potomků se nazývá koncový efektor.

46.1 Dopředná kinematika

- Pohyb kostry je vytvářen postupným natáčením kostí.
- Pro interpolaci rotací se často používají kvaterniony.

46.2 Inverzní kinematika

- Určuje se poloha koncového efektoru, zatímco rotace ostatních kostí se dopočítávají.
- Existuje více možných řešení, omezení rotací a vazby však umožňují nalézt jednoznačné řešení.

47 Fyzikální simulace

- Fyzikální simulace zahrnují:
  - Kolize objektů,
  - Simulaci pružných a měkkých objektů,
  - Simulaci látek,
  - Částicové systémy,
  - Simulaci kapalin, ohně a kouře.