EVROPSKÁ UNIE
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY
OP Vzdělávání pro konkurenceschopnost
INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ

Vysoká škola báňská – Technická univerzita Ostrava

POČÍTAČOVÁ GEOMETRIE A GRAFIKA

Teoretické základy

Ivo Špička, Robert Frischer

Ostrava 2012

Recenze: Ing. Michal Červinka, Ph.D.
Mgr. Tomáš Fismol

Název: Počítačová geometrie a grafika
Autor: Ivo Špička – Robert Frischer
Vydání: první, 2010
Počet stran: 114
Náklad: 20

Studijní materiály pro studijní obor Automatizace a počítačová technika v průmyslových technologiích fakulty FMMI
Jazyková korektura: nebyla provedena.

Určeno pro projekt:

Operační program Vzděláváním pro konkurenceschopnost
Název: Personalizace výuky prostřednictvím e-learningu
Číslo: CZ.1.07/2.2.00/07.0339
Realizace: VŠB – Technická univerzita Ostrava
Projekt je spolufinancován z prostředků ESF a státního rozpočtu ČR

ISBN 978-80-248-2590-8

POKYNY KE STUDIU

Počítačová geometrie a grafika

Pro celofakultní obory FMMI jste obdrželi studijní balík obsahující

- integrované skriptum pro distanční studium obsahující i pokyny ke studiu
- CD-ROM s doplňkovými animacemi vybraných částí kapitol
- harmonogram průběhu semestru a rozvrh prezenční části
- rozdělení studentů do skupin k jednotlivým tutorům a kontakty na tutory
- kontakt na studijní oddělení

Prerekvizity

Předmět nemá žádné prerekvizity.

Cílem předmětu

je seznámení se základními pojmy z oblasti počítačové geometrie a grafiky. Studijní materiál je rozdělen na dvě části – teoretickou a praktickou. V teoretické části se student seznámí se základními principy při vykreslování čar a ploch, vnímání světla a barevných prostorů, gama korekce a mnoho dalšího.

V praktické části jsou řešeny příklady vektorových zobrazení. Pozornost je věnována dvěma programům, na které může student v běžném životě narazit – Corel Draw a Rhinoceros 3D. Řešené příklady jsou rozebrány a postupně řešeny s názornými doprovodnými obrázky.

Přiložené animace názorně ukazují principy a složitější postupy, které se slovně vysvětlují jen obtížně.

Po prostudování modulu by měl student být schopen pochopit metody zobrazení počítačové grafiky a problematiku s tím spojenou. Dále by měl být schopen nakreslit libovolný obraz reálného, nebo imaginárního předmětu ať už ve 2D nebo 3D

Pro koho je předmět určen

Modul je zařazen do bakalářského studia oboru B3922. Automatizace a počítačová technika v průmyslu, FMMI studijního programu 3902R040 a do bakalářského studia oboru B3922 Management jakosti, FMMI studijního programu 3902R041, ale může jej studovat i zájemce z kteréhokoliv jiného oboru.

Skriptum se dělí na části, kapitoly, které odpovídají logickému dělení studované látky, ale nejsou stejně obsáhlé. Předpokládaná doba ke studiu kapitoly se může výrazně lišit, proto jsou velké kapitoly děleny dále na číslované podkapitoly a těm odpovídá níže popsaná struktura.

Při studiu každé kapitoly doporučujeme následující postup:

Čas ke studiu: 0,1 hodin

Na úvod kapitoly je uveden **čas** potřebný k prostudování látky. Čas je orientační a může vám sloužit jako hrubé vodítko pro rozvržení studia celého předmětu či kapitoly. Někomu se čas může zdát příliš dlouhý, někomu naopak. Jsou studenti, kteří se s touto problematikou ještě nikdy nesetkali a naopak takoví, kteří již v tomto oboru mají bohaté zkušenosti.

Cíl: Po prostudování tohoto odstavce budete umět

popsat ...
definovat ...
vyřešit ...

Ihned potom jsou uvedeny cíle, kterých máte dosáhnout po prostudování této kapitoly – konkrétní dovednosti, znalosti.

VÝKLAD

Následuje vlastní výklad studované látky, zavedení nových pojmů, jejich vysvětlení, vše doprovázeno obrázky, tabulkami, řešenými příklady, odkazy na animace.

Shrnutí pojmů

Na závěr kapitoly jsou zopakovány hlavní pojmy, které si v ní máte osvojit. Pokud některému z nich ještě nerozumíte, vraťte se k nim ještě jednou.

Otázky

Pro ověření, že jste dobře a úplně látku kapitoly zvládli, máte k dispozici několik teoretických otázek.

Úlohy k řešení

Protože většina teoretických pojmů tohoto předmětu má bezprostřední význam a využití v databázové praxi, jsou Vám nakonec předkládány i praktické úlohy k řešení. V nich je hlavní význam předmětu a schopnost aplikovat čerstvě nabyté znalosti při řešení reálných situací hlavním cílem předmětu.

Úspěšné a příjemné studium s touto učebnicí Vám přejí autoři výukového materiálu

Ivo Špička a Robert Frischer

OBSAH

1 POČÍTAČOVÁ GRAFIKA A BARVY

Cíl Po nastudování kapitoly se

budete se umět orientovat v problematice barev v počítačové grafice.
Porozumíte fyzikální podstatě světla a barev a jejich reprezentaci v počítači.
Bude schopni vyjmenovat základní barvové modely používané v počítačové grafice.
Seznámíte se a budete schopni aplikovat základní matematické popisy těchto barvových modelů a bude umět převést jeden barevný model na jiný.
Seznámíte se a bude umět v počítačové grafice využívat gama korekce a alfa míchání.

1.1 Světlo a barva

Čas ke studiu: 0,5 hodina

Cíl Po prostudování tohoto odstavce budete umět

Porozumíte fyzikální podstatě světla a barev
Popsat reprezentaci barev v počítači
definovat co je světlo
vyjmenovat základní atributy barvy

Výklad

Jedním z důležitých lidských smyslů je zrak a zrakové vnímání okolního světa. To, že jsme schopni zrakem vnímat okolí je způsobeno šířením světla, jeho částečným odrazem a částečným pohlcením povrchem rozličných předmětů. Oko a jeho optická soustava spolu se sítnicí pak toto světlo zachytí a převede na nervové vzruchy, které zpracuje náš mozek a převede do našeho vnitřního obrazu okolí.

Světlo, či přesněji světelní záření, je elektromagnetické vlnění. Patří do stejné kategorie jako elektromagnetické vlny televizního a rozhlasového vysílání, tepelné záření třeba grilu, rentgenové záření. Čím se tyto jednotlivá vlnění liší. Proč, když se jedná o stejný druh vlnění, vnímáme z celé škály elektromagnetického vlnění jen malou část?

Je to dáno tím, že sítnice oka je citlivá, tedy umí vnímat, jen elektromagnetické záření jistých vlnových délek, respektive jistých frekvencí. Světlo se v prostoru šíří danou rychlostí, označujeme ji c a měříme v metrech za sekundu $[ms^{-1}]$. Rychlost šíření světla je pro dané prostředí konstantní. Vlnovou délku označujeme řeckým písmenem λ, měříme ji v metrech. Frekvence f se udává v hercích [Hz] a vztah mezi těmito fyzikálními veličinami je dám vztahem

$$\frac{f}{c} = \lambda \qquad (1)$$

Světlo je elektromagnetické vlnění s frekvencí v oblasti $10^8 Hz$. Jednotlivým frekvencím pak odpovídají jednotlivé barvy. Nejnižší frekvenci má červené světlo $4{,}3 \times 10^8\ Hz$ a nejvyšší frekvenci, které lidské oko vnímá je asi frekvence $7{,}5 \times 10^8\ Hz$, která odpovídá fialové barvě. Člověk rozliší přibližně 4×10^5 barev a jejich odstínů.

Světlo, které obsahu celou škálu barev je achromatické neboli bílé světlo. Světlo jedné konkrétní barvy, jedné frekvence, je světlo monochromatické.

Světlo má určité vlastnosti, atributy, které je jednoznačně určují:

- Jas určuje intenzitu světla,
- Barva je základní atribut a je dána frekvencí či vlnovou délkou
- Barevná sytost je dána rozsahem frekvencí, které světlo obsahuje. Světlo s nulovou sytostí barvy je světlo bílé. Maximální sytost má světlo, obsahující právě jednu barevnou složku, jednu frekvenci.
- Světlost určuje podíl chromatického podílu světla k určité dominantní barvě, frekvenci.

Co v běžném životě příliš nevnímáme, a pokud nejsme malíři nebo třeba počítačoví grafici, je míchání barev. Každý z nás asi někdy v životě míchal barvy, třeba ve škole při hodinách výtvarné výchovy. Možná jste si položili otázku, zda existují nějaké základní barvy, jejichž kombinací, mícháním, lze získat libovolnou barvu požadované sytosti a odstínu. Ukážeme si, že takové barvy existují, vysvětlíme si princip míchání barev. Abychom byli schopni dosáhnout opakovaného výsledku, vysvětlíme si funkci chromatického diagramu, základních barev a barev doplňkových a funkci jednotlivých barvových modelů.

Shrnutí pojmů

Světlo je elektromagnetické vlnění s frekvencí v oblasti $10^8 Hz$.

- Jas určuje intenzitu světla.
- Barva je základní atribut a je dána frekvencí či vlnovou délkou.
- Barevná sytost je dána rozsahem frekvencí, které světlo obsahuje.
- Světlo s nulovou sytostí barvy je světlo bílé.
- Maximální sytost má světlo, obsahující právě jednu barevnou složku, jednu frekvenci.
- Světlost určuje podíl chromatického podílu světla k určité dominantní barvě, frekvenci.

Otázky

1. ? Co je to elektromagnetické záření.
2. Jaký vztah má vlnová délka a frekvence elektromagnetického záření.
3. Jaké jsou základní atributy světla?
4. Definujte sytost, jas, monochromatické, achromatické světlo.

1.2 Vnímání světla a barevné prostory

Čas ke studiu: 0,5 hodina

Cíl Po prostudování tohoto odstavce budete umět

Porozumíte podstatě lidského vnímání světla
porozumíte významu tyčinek a čípků na sítnici oka
budete umět definovat základní principy míchání barev

Výklad

1.2.1 Způsoby vnímání světla a jeho důsledky

Cit pro barvy, barvocit, je u každého člověka individuální. Individuální je i vnímání barvy. Přesto je možné najít určité standardy, které povedou k stejnému výsledku, ať postup míchání barvy uskuteční kdokoli. Také je možné určit základní barvy, jejichž mísením získáme výslednou barvu.

Existují dva základní způsoby míchání barev, a to:

- aditivní míchání a
- Substraktivní míchání barev.

Zamyslíme-li se nad pojmy aditivní a substantivní, poznáme, že už vlastní názvy tak trochu napovídají, jak a kdy se použije jeden či druhý způsob míchání. Adice znamená přidání, sčítání. Substrakce znamená odebrání, odečtení. Při aditivním míchání přidáváme světlo, barvu, o určité vlnové délce. Při substraktivním míchání barev nějakou část spektra odejmeme.

Zamyslete se, proč vlastně vidíme předměty barevně, nebo ještě obecněji, proč je vůbec vidíme?

Oko, jako orgán zraku, je z fyzikálního hlediska optická soustava, která má za úkol světelné parsky soustředit na sítnici oka. Na sítnici samotné jsou pak umístěny světlocitlivé buňky, tyčinky a čípky. Ty jsou dopadajícím světlem různým způsobem podrážděny a tyto podněty pak jsou nervovými vlákny přenášeny do mozku. Tam vzniká náš vjem obrazu.

- Tyčinky – je jich asi 130 mil. Rozlišují odstíny šedi, vůbec nejsou ve žluté skvrně a jsou citlivější na světlo, umožňují vidění za šera a zajišťují periferní vidění. Jejich činnost umožňuje oční purpur – rodopsin (vit A a bílkovina opsin).
- Čípky – těch je méně asi 7 mil. Umožňují barevné vidění. Jsou tři druhy čípků, citlivé k modré, zelené a červené barvě. Největší nakupení čípků je asi 4 mm od slepé skvrny na mírně vkleslém místě sítnice, tzv. žlutá skvrna (místo nejostřejšího vidění). Slepá skvrna je místo, kde ze sítnice vychází zrakový nerv a jak její název napovídá, nejsou zde ani čípky ani tyčinky a dopadající světlo na tuto plochu není vnímáno.
-

1.2.2 Barvy a jejich míchání

Nyní zpět k otázce, proč vidíme předměty okolo nás. Ve tmě je nevidíme, pokud samy nejsou zdrojem světla jako slunce, žárovka, zářivka nebo například obrazovka monitoru. Ostatní předměty musí být osvětleny. Část dopadajícího světla je povrchem předmětu pohlcena, část odražena. Odražená část světelných paprsků pak může být vnímána naším zrakem. Tím předměty vidíme jako takové. Vlnové délky, které povrch předmětu pohltí, již nejsou k divákovi odraženy. Světelné spektrum se změnilo. To vede k tomu, že různé předměty mají barvu podle toho, jakou část světelného spektra pohltí a jakou část odrazí. Pokud pohltí všechno, nebo většinu světla, pak předmět vnímáme jako tmavý, černý. Pokud odrazí většinu dopadajícího světla, pak se předmět v bílém světle bude jevit bílý. Z dopadajícího světla se tedy odečte jeho část, a proto v případě mísení barev, ve smyslu látky nanášené na povrch například papíru, mluvíme o substraktivním (odečítacím) způsobu míchání barev.

Naopak kombinací světla z různých zdrojů k sobě jednotlivé složky přidáváme, a proto mluvíme o aditivním (součtovém) mísení barev.

Shrneme-li předešlé pak:

- Aditivní míchání - přidáním určité složky světla se změní její světlost a barva. Pokud smícháme všechny složky, dostaneme bílé světlo. Typickým modelem pro aditivní mísení barev je model RGB, kde písmenka znamenají zkratku barvy: Red (červená), Green (zelená) a Blue (modrá) barva.
- Substraktivní míchání barev - z původně obsažených všech vlnových délek původního světla je část vlnových délek částečně či úplně pohlcena. Smícháním všech barev získáme barvu, alespoň teoreticky, černou, prakticky pak nějakou spíše hnědou barvu. Typickým modelem je pak model CMY, písmena označují barvy: C - Cyan (tyrkysová), M - Magenta (fialová, červenomodrá) a Y - Yellow (žlutá). Smíšením dvou barev dostaneme takzvanou doplňkovou barvu, která je právě jedna z trojice modelu RGB. A také naopak smíšením, (sečtením) dvou barev modelu RGB dostaneme jejich doplňkovou barvu modelu CMY.

1.2.3 Barvový model, prostor barev

Výše uvedené dva modely nejsou zatím úplným výčtem barvových modelů, ale mohou nám posloužit proto, abychom definovali barvový model. Je určen:

- množinou základních barev
- pravidly míchání těchto základních barev a
- pravidly, podle nichž se mění barevné charakteristiky.

Mezi základní modely patří model RGB, model CMY(K), model HSB, model HLS, model UWB a LAB. Zhusta se také místo barvový model požívá slova prostor. Toto pojmenování vychází z představy určité barvy jako bodu v prostoru, kdy právě souřadnice určují, jakou výslednou barvu model má. V dalším textu se budeme držet pojmu prostor.

Shrnutí pojmů

Na sítnici oka nalezneme čípky a tyčinky.

Čípky nejsou citlivé na vlnovou délku, barvu světla.

Tyčinky jsou trojího druhu a jsou citlivé k základním barvám, a to červené, modré a zelené.
Základní dva způsoby míchání barev jsou aditivní a substraktivní míchání.
Barevný prostor neboli barvový model je charakteristický množinou základních barev, pravidly míchání těchto základních barev a pravidly, podle nichž se mění barevné charakteristiky.

Otázky

1. Které buňky o v oku jsou citlivé k barvě?
2. K jakým základním barvám je citlivé lidské oko?
3. Co je to barevný prostor?
4. Jaké znáte základní způsoby míchání barev?

Úlohy k řešení

1.3 Chromatické diagramy a gamut

Čas ke studiu: 0,5 hodina

Cíl Po prostudování tohoto odstavce

porozumíte pojmu chromatický diagram
pochopíte pojem a význam stimulu
budete umět definovat chromatičnost barev
porozumíte vztahu mezi chromatickým diagramem a mamutem
budete schopni využít gamut při tisku i práci v grafickém editoru

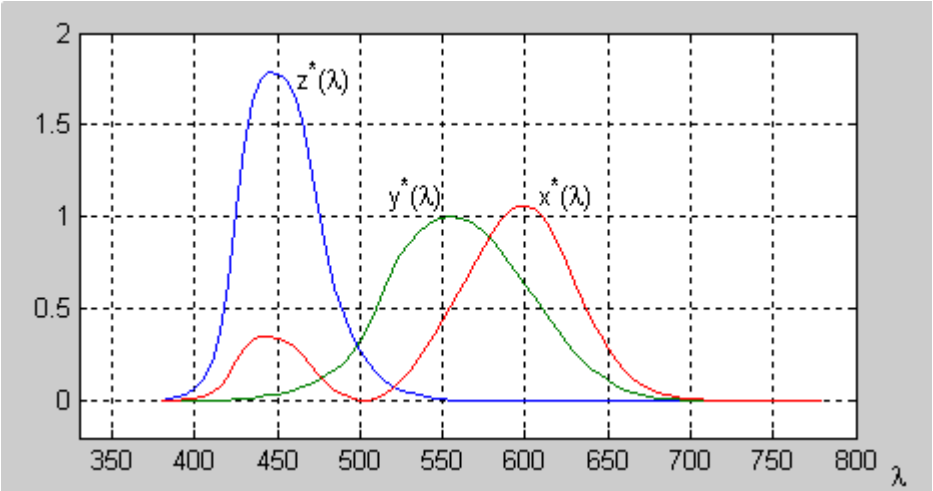
Výklad

Důležitými pojmy v grafice jsou mimo jiné chromatický diagram a gamut.

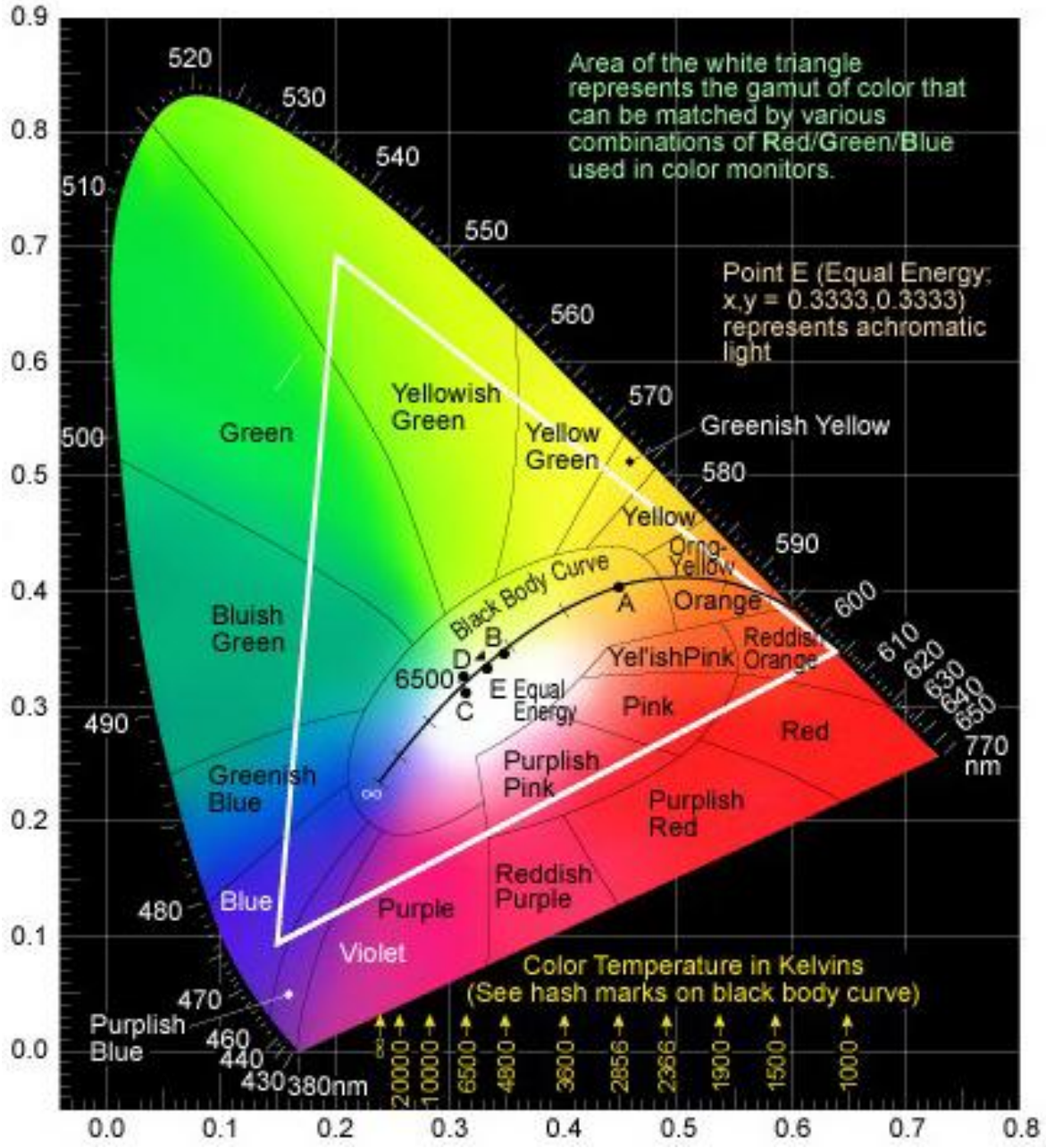
1.3.1 Barevný (chromaticity) diagram

Mezinárodní organizace CIE (Commission Internationale de L'Éclairage) definovala imaginární barvy (spektrální křivky) "červená (red)", "zelená (green)" a "modrá (blue). Tyto křivky byly zkonstruovány na základě měření citlivosti lidského oka na základní barvy. Přesněji řečeno **standardní kolorimetrický pozorovatel CIE 1931** se skládá ze tří tzv. color **matching funkcí**, $x^*(\lambda)$, $y^*(\lambda)$ a $z^*(\lambda)$. Na ose x je vynesena vlnová délka, na ose y jsou uvedeny hodnoty tzv. **color matching funkcí**, $x^*(\lambda)$, $y^*(\lambda)$ a $z^*(\lambda)$. Color matching funkce $x^*(\lambda)$, $y^*(\lambda)$ a $z^*(\lambda)$ slouží ke stanovení speciální trojice čísel, tzv. **XYZ tristimulu**, který umožňuje navzájem porovnávat barvy různých barevných podnětů.

Barevný podnět je světlo vyzařované, odražené či propouštěné pozorovaným objektem, na které naše oči reagují (tj. díky kterému daný objekt vidíme). Toto světlo má jisté spektrální rozložení energie, které se dá popsat pomocí funkce $f(\lambda)$, zde lambda je vlnová délka světla. Laicky řečeno, tato funkce udává, kolik které vlnové délky je v tomto světle obsaženo. Spektrální rozložení energie podnětu určuje to, jakou barvu uvidíme. Vztah mezi spektrálním rozložením energie a námi viděnou barvou je příliš nejednoznačný. Je to dáno mimo jiné tím, že naše vidění je pouze trojbarevné. Mnoho podnětů se zcela odlišnou spektrální funkcí se tak lidskému zraku jeví shodně.



Obr 1. . Standardní kolorimetrický pozorovatel CIE 1931 a průběhy tří tzv. color matching funkcí pro jednotlivé barvy



Obr 2. Chromatický diagram (CIE 1931) zachycující křivku čistých spektrálních barev.

V kalorimetrii, tedy oblasti, která pracuje s barvami, je mnohdy užitečné se zaměřit pouze na **chromatičnost** (barvu jako takovou – její „barevnost“). Nebude nás tedy zajímat intenzita příslušného podnětu neboli jas barvy. Toho je možné dosáhnout normalizací XYZ tristimulu. V kolorimetrii se tradičně používá jednoduchá normalizace

$$x = \frac{X}{X + Y + Z} \tag{2}$$

$$y = \frac{Y}{X + Y + Z} \tag{3}$$

$$z = \frac{Z}{X + Y + Z} \tag{4}$$

Pakliže pro tyto normalizované hodnoty tristimulu platí $x + y + z = 1$, pak je třetí hodnota nadbytečná – lze ji snadno spočíst z předchozích dvou $(z = 1 - x - y)$. K zachycení chromatičnosti barvy stačí jen dvě hodnoty, x a y.

Redukce na pouhé dvě hodnoty umožňuje znázornit mnoho skutečností graficky, a to prostřednictvím **chromatických digramů. Chromatický diagram** je graf, kde *x* se vynáší na vodorovnou a *y* na svislou osu. Příklad takového diagramu vidíme na Obr. 3. Vidíme zde barevnou plochu. Její obvod tvoří čisté spektrální barvy, tedy jak už víme ty obsahující. Vlastní barvy na obrázku jsou, ale jen ilustrativní, neboť jak si řekneme dále žádný monitor ani tiskárna nedokáže přesně reprodukovat plný barevný rozsah Chromaticity diagramu.

Barevná "podkova" začíná na 420nm vlevo dole, otáčí se kolem 520nm ke svému konci kolem 680nm. Uvnitř tohoto diagramu leží všechny barev, které je člověk schopen vnímat.

1.3.2 Gamut primárních barev

Při reprodukci barev se používají tři (nebo někdy i více) zdroje primárních barev. Mícháním primárních barev se vytvářejí požadované barvy.

Položte si sami otázku, proč jsou to právě tři barvy, jejichž mísením získáme požadovanou barvu?

Tři základní barvy jsou minimum, protože naše vidění je, jak už víme, trojbarevné. Spektrální rozložení energie originálního podnětu, který barevně reprodukujeme, se ale zcela neshoduje, vytváří se jiný podnět, který je s originálem metamerický (neboli má stejnou barvu).

Abychom byli schopni namíchat libovolnou barvu z chromatického diagramu pomocí tří základních barev, pak bychom potřebovali použít záporné množství některé z primárních barev. Toto je fyzikálně nerealizovatelné.

Ale proč nemůžeme reálně použít zápornou hodnotu některé barvy?

Všechny fyzikálně realizovatelné barvy, tedy v reálném světě možné, vzniklé mísením tří primárních barev leží uvnitř tzv. **Maxwellova trojúhelníka** (je-li primárních barev více, řekněme n, tak n-úhelníka). Vrcholy tohoto trojúhelníku reprezentující základní barvy, které jsme pro míchání použili. Na Obr. 5 je např. červenou barvou vyznačený Maxwellův trojúhelník pro standardní trojici základních barev definovanou CIE – čistou spektrální červenou o vlnové délce λ= 700 nm, zelenou λ= 546,1 nm a modrou λ= 435,8 nm.

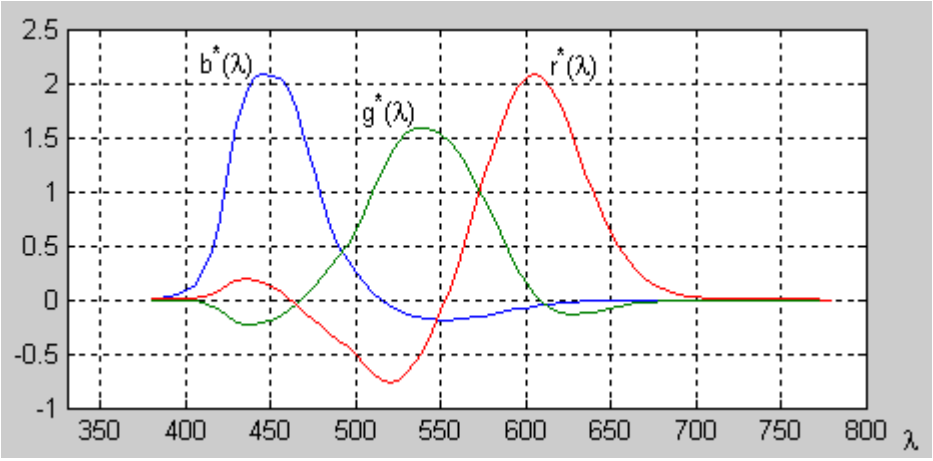
Rozsah barev, které leží uvnitř tohoto trojúhelníku, nazýváme **barevný gamut.**

Řekli jsme si, že nejsme pomocí tří barev získat barvy z celého viditelného spektra. Využijeme jen barvy odpovídající danému gamutu. Bohužel skutečnost je ještě horší. Ignorovali jsme jas. Proto, přes fyzikální opodstatněnost, je Maxwellův trojúhelník pouze teoretická aproximace reálného gamutu daných základních barev. Nejsme totiž schopni přidávat ani neomezeně velká množství světla každé z barev. U reálného gamutu nelze zanedbat intenzitu a omezit se pouze na chromatičnost. Dostáváme se

opět k trojrozměrnému objektu, který má složitější tvar a nejde ho zredukovat do pouhých dvou dimenzí. Ale to již přesahujeme možnosti tohoto textu a zájemce odkazujeme na doporučenou literaturu.

Na obrázku (Obr 3) vidíme color matching funkce luminoforů barevného CRT monitoru. Luminofor je chemická látka, která u obrazovky monitoru při dopadu elektronu vyzařuje světlo určené vlnové délky (barvy).

Na tomto diagramu jsou použity tzv. imaginární barvy, které leží vně Maxwellova trojúhelníku. Jsou to místa, kde křivky přecházejí do záporných hodnot.



Obr 3. Průběh color match funkcí imaginárních barev.

Shrnutí pojmů

Mezinárodní organizace CIE (Commission Internationale de L'Éclairage) definovala imaginární barvy (spektrální křivky) "červená (red)", "zelená (green)" a "modrá (blue).

Standardní kolorimetrický pozorovatel CIE 1931 se skládá ze tří tzv. color **matching funkcí**, $x^*(\lambda)$, $y^*(\lambda)$ a $z^*(\lambda)$.

Barevný podnět je světlo vyzařované, odražené či propouštěné pozorovaným objektem, na které naše oči reagují (tj. díky kterému daný objekt vidíme).

Chromatický diagram je graf, kde *x* se vynáší na vodorovnou a *y* na svislou osu.

Všechny fyzikálně realizovatelné barvy, tedy v reálném světě možné, vzniklé mísením tří primárních barev leží uvnitř tzv. **Maxwellova trojúhelníka**

Rozsah barev, které leží uvnitř **Maxwellova trojúhelníka** trojúhelníku nazýváme **barevný gamut**.

Otázky

1. ?Co definovala Mezinárodní organizace CIE?
2. Z čeho se skládá **standardní kolorimetrický pozorovatel CIE 1931?**
3. Co je to **barevný podnět?**
4. Kde leží všechny fyzikálně realizovatelné barvy?
5. Co je to **barevný gamut**?

1.4 Prostory barev

Čas ke studiu: 0,5 hodina

Cíl Po prostudování tohoto odstavce

budete vyjmenovat základní barevné prostory
budete schopni popsat prostor RGB
porozumíte významu jednotkové barevné krychle
budete se orientovat v prostorech HSB, model HLS, model UWB a LAB

Výklad

Uvedli, jme již základní vlastnosti barvového modelu, který je též nazýván barevným. Víme, že barevný prostor je určen množinou základních barev, pravidly míchání těchto základních barev a pravidly, podle nichž se mění barevné charakteristiky. Mezi základní modely patří model RGB, model CMY(K), model. V dalším textu se budeme držet pojmu prostor.

1.4.1 Prostor RGB

Výsledné barvy jsou v tomto prostoru získány jakou součet, adice, základních barev. Základní barvy jsou vybrány podle největší citlivosti oka, jednotlivých druhů čípků, na tyto barvy. Složkami jsou základní barvy

Red (červená) s vlnovou délkou 630nm,

Green (zelená) s vlnovou délkou 530nm a

Blue (modrá) barva s vlnovou délkou450nm.

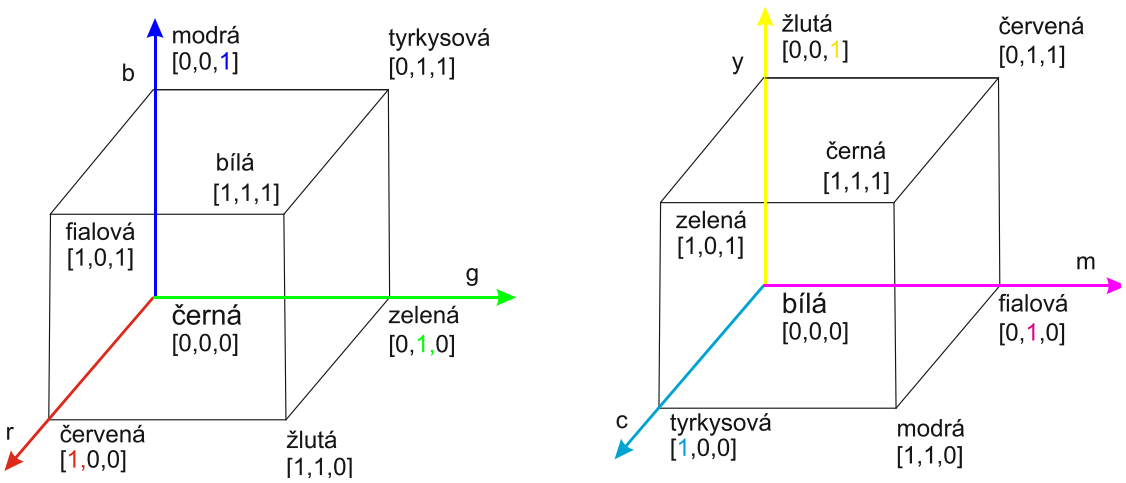
Intenzita každé složky ve výsledném světle je dána číslem z intervalu nula až jedna. Někdy se také používá procentuální vyjádření nula pro nepřítomnou barvu a sto procent pro plný jas základní barvy. V technické praxi, používáme počítače, je interval hodnot nula až jedna převeden na interval hodnot nula až 255. Tento rozsah je právě celá množina celách kladných čísel, která jsou ve dvojkové soustavně zaznamenat na jednom byte (bajtu), tedy pomocí osmi bitů. Lidské oko je schopno vnímat asi sto různých hodnot jedné barvy.

Model se obvykle prezentuje ve formě jednotkové kostky,délka hrany odpovídá délce intervalu intenzity každé základní složky (Obr 4). Jednotlivé barvy pak budeme vyjadřovat jako bod v třírozměrném prostoru jednoho oktantu. Souřadnicové osy pak označíme písmeny základních barev. Počátek soustavy bude mít souřadnice [0, 0, 0], který bude odpovídat četné barvě.

Pokud dodržíme pořadí barev dle písmen pak bod

- [1,0,0] bude vyjadřovat maximální intenzitu červené barvy,
- [0,1,0] bude vyjadřovat maximální intenzitu zelené barvy a

- [0, 0, 1] bude vyjadřovat maximální intenzitu modré barvy.



Obr 4. Barevná krychle modelů RGB a CMY

Prostor RGB je někdy doplněn ještě o složku průhlednosti A nebo taky α. Rozsah hodnot je opět nula až jedna. Této složce se mnohdy říká také alfa kanál. Hodnota nula znamená neprůhledný,hodnota jedna znamená zcela průhledný. Tato možnost se zhusta používá při kombinaci více obrázků. Takové obrázky si můžeme přestavit jako malbu na skle. Pokud obrázek na skle položíme na jiný vytištěný na papíře, pak bude viditelný jak horní tak i spodní obrázek. Nakolik bude spodní obrázek viditelný, závisí právě na průhlednosti toho horního.

1.4.2 Prostor CMY

Jak jsme se již dozvěděli, v tomto modelu jsou barvy vytvářeny substraktivním způsobem. Z hlediska naší praktické zkušenosti se nám tento model jeví jako přirozený. Výsledná barva je dána kombinací základních barev a to:

- *C* - (Cyan) *tyrkysová*,
- *M* - (Magenta) *fialová* a
- *Y* - (Yellow) *žlutá*

Tento model bývá analogicky mnohdy doplněn ještě jednou barvou a to černou. Pak hovoříme o modelu CMYK. Přidání černé je dáno praktickým hlediskem; sytě černý nebo čistě šedý tón je jen obtížně dosažitelný mísením základních barev. Způsobuje to mimo jiné i to, že základní barvy nejsou úplně čisté, obsahují i jiné než svou základní složku.

Mezi modely RGB a CMY jde jednoduše přecházet. Tento přechod je nutný i z toho hlediska, že tiskárny z principu používají model CMY nebo CMYK či jejich obdoby a displeje a interní reprezentace barev v počítači je určena modelem RGB. Z vlastní zkušenosti asi víte, že tiskové náplně jsou obvykle právě ony čtyři: žlutá, purpurová a modrozelená neboli tyrkysová. Monitor zase barvy skládá z červené, modré a zelené barvy světla, pracuje s modelem RGB.

Pokud tedy interní reprezentace barev je RGB a tiskárna využívá model CMY(K), pak je nutno přepočítat hodnot složek jednoho modelu na hodnoty složek druhého modelu tak, aby výsledná barva byla v obou prostorech shodná. Tento přepočet je snadný. Zmínili jsme se, že dvojice základních barev

tvoří doplňkovou barvu, která je základní barvou druhého prostoru. Matematicky přepočet můžeme zapsat jako uspořádané trojice v podobě vektorů:

$$\begin{bmatrix} r \\ g \\ b \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} c \\ m \\ y \end{bmatrix} \tag{5}$$

Vezměme v prostoru RGD červenou a zelenou barvu a smíchejme je, pak obdržíme žlutou. Ta je barvou doplňkovou k barvám červené a modré. Je taká základné barvou v prostoru CMY.

Matematicky pak

$$\begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} c \\ m \\ y \end{bmatrix} \tag{6}$$

$$\begin{bmatrix} c \\ m \\ y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}. \tag{7}$$

Výsledkem je žlutá barva $[0, 0, 1]$ v prostoru CMY.

Je tato barva žlutá i v prostoru RGB?

1.4.3 Model HSB

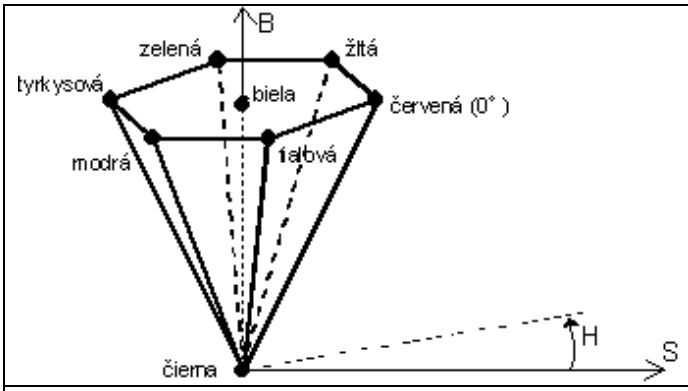
Předchozí dva modely, jak jsme se zmínili, vycházely z požadavků technické praxe. Naše lidské zkušenosti ale jsou jiné. S modely RGB a CMY se nepracuje intuitivně. Při malování jsme zvykli tomu, že jsme si vybrali nějaký barevný tón, stanovili jsme jeho sytost a jas.

Základní složky tohoto modelu jsou:

- *H* - (Hue) odstín barvy,
- *S* - (Saturation) *saturace neboli sytost barvy a*
- *B* - (Brightness) čili *hodnota jasu.*

Barevný prostor je modelován jako šestiboký jehlan, jehož vrchol leží v počátku barevného prostoru (Obr 5). Hodnoty jasu B a nasycení S se obdobně jako u dvou předchozích modelů nabývají hodnot od nuly po jedničku. Barevný tón je udáván jako úhel od počátku, nabývá hodnot od nuly stupňů po 360°. Vrchol, počátek soustavy $[0, 0, 0]$ odpovídá černé barvě, střed podstavy je místo bílé barvy. Sytost barvy je dána vzdáleností od osy jehlanu. Čisté barvy leží ve vrcholech šestiúhelníkové podstavy. Při změně odstínu se při stejné sytosti se nepohybujeme po přirozené kruhové dráze, ale musíme sledovat dráhu rovnoběžnou s pláštěm jehlanu. Musíme se tedy pohybovat po šestiúhelnících.

Graficky je model zachycen na obrázku



Obr 5. Model HSB

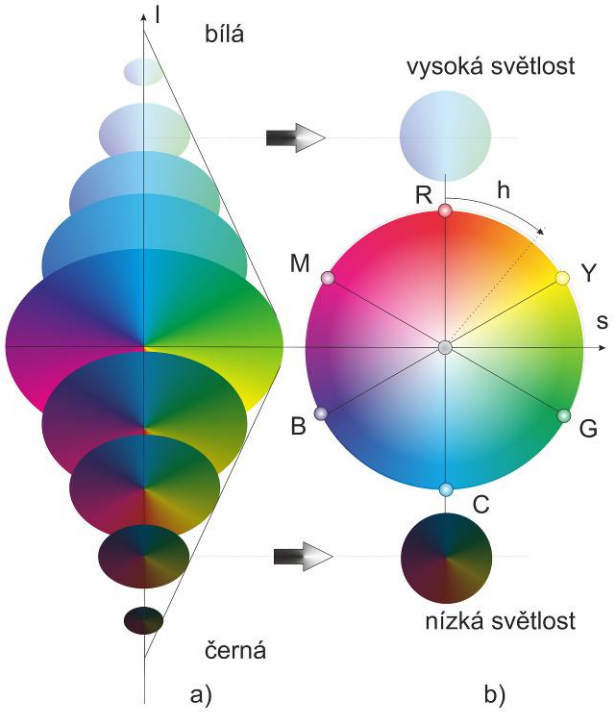
1.4.4 Model HLS

Prostor HLS vychází z prostoru HSB a odstraňuje jeho hlavní nedostatek: místo šestihranného jehlanu prostor modelují dva kužely. Vrchol horního kužele představuje bílou barvu, vrchol spodního naopak barvu černou (Obr 6). Osa kuželů l představuje světlost Lightens, vzdálenost o od osy udává saturaci Saturation. Barevný tón h Hue je určen úhlem z intervalu nula až 360°. Na kružnici, která je okrajem společné podstavy, opět najdeme základní barvy, červenou, žlutou, zelenou, tyrkysovou, modrou a fialovou.

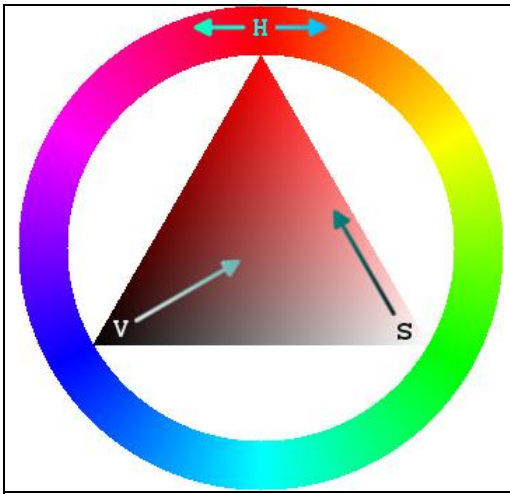
Tento model snad nejvíce odpovídá podstatě lidského vnímání barev. Lidské oko totiž nejvíce odstínů rozeznává asi při střední světlosti. Ta je u tohoto modelu určena hodnotou 0,5 l.

Neutrální barvy - škála šedé od černé barvy až po bílou leží na ose l.

Shrneme-li naše poznatky, pak vidíme, že určení požadované barvy u tohoto modelu je vskutku intuitivní. Vybereme si požadovaný odstín, jeho sytost a jas.



Obr 6. Model HLS Řezy kužely modelu jsou zobrazena v části a). Vybrané řezy, tmavý, řez v základně kuželů a světlá jsou pak zobrazeny v části b). Zde jsou také zobrazeny základní barvy R - červen, Y -žlutá, G - zelená, C - tyrkysová, B - modrá a M - fialová.



Obr 7. Barevné kolo modelu HLS

K výběru barvy v počítači se tento model přímo nehodí. Představme si, že kužely podél osy rozřízneme. Pokud jsme kužely rozřízli v místě, kde na hraně podstav je červená barva, pak tento řez bude tvořit kosodélník. Z tohoto řezu si vybereme jen jednu polovinu, například od osy vpravo. Zůstane nám jen trojúhelník, kdy jeden vrchol tvoří čistá barva, druhý vrchol bílá barva - odpovídá vrcholu jehlanu světlých barev a třetí představuje černou barvu (Obr 7).

1.4.5 Model HLV

Prostor HLV je vymezen je na rozdíl od předchozího modelu HLS vymezen jen jedním jehlanem. Černá barvy je umístěna ve shodě s modelem HLS ve vrcholu kužele, bílá barva je ale modelována uprostřed podstavy jehlanu.

Model YUV

Je používán v technické praxi, při televizním vysílání, ve videokamerách apod. Používají se tři složky - signálové kanály, dvě složky nesou informaci o barvě a třetí o jasu.

1.4.6 Model LAB

LAB model popisuje barvy obdobně jako YUV model. Složka **L** je Luminance shodně s příchozími modely HLS, HLV. Její hodnoty jsou od nuly do jedné. Složky **a** a **b** popisují barvu bodu. Složka a udává barvy na škále červená - zelená a b = na škále modrá zelená. Výhodou modelu LAB je to, že tento model je nezávislý na zařízení. V praxi je tento model užitečné využít pro doostřování digitálních fotografií. Doostřuje se pouze l - jasová složka Předejde se tím vzniku nepěkných barevných artefaktů na hranách při doostřování.

Jelikož cílem této publikace je zejména praktické používání grafických nástrojů, matematický popis převodu jednotlivých modelů mezi sebou navzájem zde nebudeme uvádět. Pro zájemce dobře poslouží literatura….

Shrnutí pojmů

Výsledné barvy **barevného prostoru RGB** jsou získány jakou součet, adice, základních barev.

Základní barvy jsou vybrány podle největší citlivosti oka, jednotlivých druhů čípků, na tyto barvy. Složkami jsou základní barvy Red (červená) s vlnovou délkou 630nm, Green (zelená) s vlnovou délkou 530nm a Blue (modrá) barva s vlnovou délkou450nm.

Barvy v prostoru **CMY** vytvářeny substraktivním způsobem. Výsledná barva je dána kombinací základních barev a to: *C* - (Cyan) *tyrkysová*, *M* - (Magenta) *fialová* a*Y* - (Yellow) *žlutá*

Model **HSB** je blízký lidským zkušenostem

Základní složky tohoto modelu HSB jsou - (Hue) odstín barvy, *S* - (Saturation) *saturace neboli sytost barvy a B* - (Brightness) čili *hodnota jasu.*

Další používané modely jsou HLS, HLV, YUV a LAB.

Výhodou modelu **LAB** je to, že tento model je nezávislý na zařízení. V praxi je tento model užitečné využít pro doostřování digitálních fotografií.

Otázky

1. Vyjmenujte základní barevné prostory.
2. Charakterizujte RGB prostor.
3. Charakterizujte prostor CMY.
4. Popište rozdíl mezi aditivním a substraktivním mícháním barev.
5. K čemu je vhodné používat model LAB.

1.5 Gama-korekce a alfa míchání

Čas ke studiu: 20 minut

Cíl Po prostudování tohoto odstavce

porozumíte důležitosti gama korekce
budete schopni využít alfa míchání

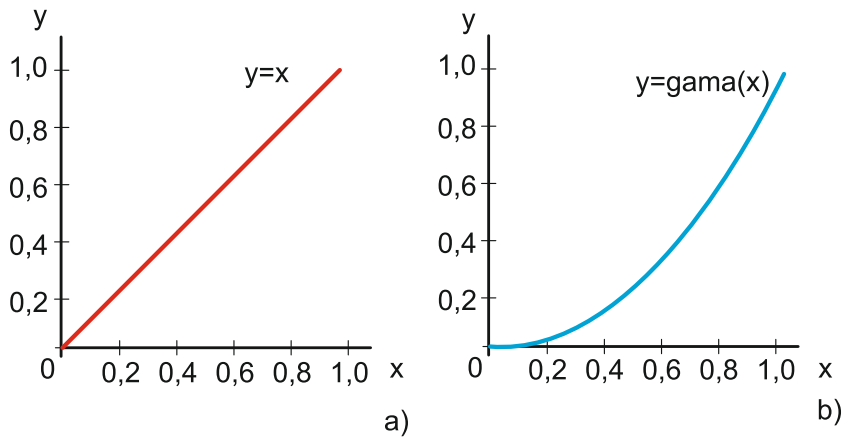
Výklad

Mimo základní barevné prostory je pro práci v grafických editorech a při využívání grafických periférií porozumět pojmům gama-korekce a principu alfa míchání

1.5.1 Gama-korekce

Barvy prezentované v jednotlivých modelech bereme jako lineárně závislé funkce. Zařízení, monitor či tiskárna, zobrazují již ne abstraktní barvy, ale barvy jako fyzikální realitu ve formě viditelného světla nebo naneseného pigmentu (barvy) na papír. Pro jednoduchost uvažujme například modrý subpixel (modrá část celého pixelu) barevného monitoru. Zatím jsme předpokládali, že intenzita jasu tohoto subpixelu - modré složky bude odpovídat lineárně hodnotě b dané barvy např. v modelu RGB. Tuto představu ilustruje (Obr 8a)

Skutečnost ale je jiná jak nám napovídá (Obr 9b). Lineárnímu průběhu intenzity modré barvy v počítači neodpovídá průběh jasy modrého subpixelu. Barvy se nám pak jeví nepřirozeně. Navíc i citlivost lidského oka není lineární funkcí intenzity podnětu.

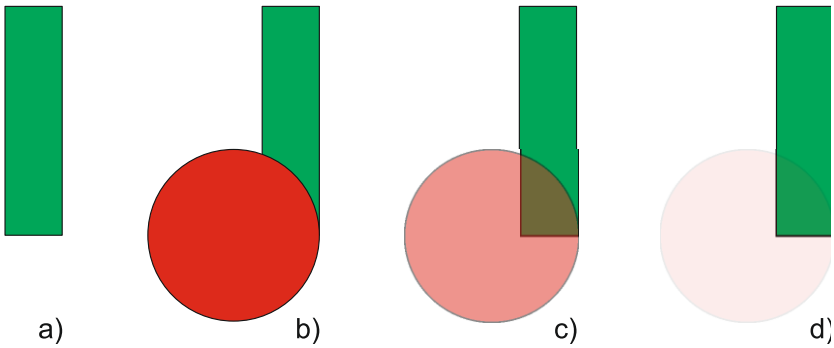


Obr 8. Gama korekce. Na obrázku ad a) je lineární ideální závislost hodnoty veličiny y na x, obrázek b) znázorňuje přibližný průběh gama křivky. Na ose x je normovaná hodnota např., červeného kanálu, na ose y pak normované odpovídající hodnoty jasu červeného subpixelu

Tuto skutečnost korigujeme gama korekcí. Její možný tvar ukazuje obr.

1.5.2 Alfa-míchání

Alfa-míchání *(α-blending*) je funkce pro zobrazování poloprůhledných ploch. Toto míchání je možno provést softwarově nebo je zajistí hardwarové vybavení počítače - grafická karta. Koeficient průhlednosti se označuje jako *alfa-složka, alfa kanál*. Rozsah průhlednosti se udává buď v rozmezí intervalu nula až jedna, nebo počítačově v rozmezí nula až 255. Úplnou průhlednost objektu zajistíme, pokud hodnota *alfa-složky bude nula. Neprůhledný objekt bude mít hodnotu alfa kanálu* rovnu jedné.



Obr 9. Alfa míchání a jeho projevy v grafickém editoru

Na obrázku (Obr 9) vidíme průběh práce v grafickém editoru. Nejprve jsme vložili zelený objekt, pak červený kruh, který je neprůhledný. Hodnota jeho alfa složky je jedna. Pořadí objektů je dáno pořadím jejich vložení na pracovní plochu editoru.Červený kruh nám na obrázku ad b) díky stoprocentní neprůhlednosti překryl zelený obdélník. Na obrázku c) jsme změnili průhlednost vrchního červeného prvku na padesát procent, hodnota alfa složky bude 0,5. Zelený obdélník začíná prosvítat. Také sytost barvy se změnila, jelikož jsme k čisté červené barvě díky průhlednosti přidali bílou barvu podkladu. V části d) jsme ještě zvýšili průhlednost na 90 procent, hodnota alfa složky pak je 0,1 a objekt je téměř průhledný.

Gama křivka vyjadřuje odchylku mezi požadovaným a skutečných průběhem odezvy zařízení.

Gama korekce koriguje tuto odchylku.

Alfa míchání se využívá pro prolínání obrázků.

Otázky

1. Co znamená gama korekce?
2. Co nazýváme alfa-mícháním a popište jeho význam

Otázky k problému barev a barevných prostorů

Charakterizujte problém používaní a zpracování barev v počítačové grafice

Charakterizujte a popište model RGB

Charakterizujte a popište model CMY

Charakterizujte a popište model HSB

Charakterizujte a popište model HLS

Charakterizujte a popište model UWB

Co znamená gama korekce

Co nazýváme alfa-mícháním a popište jeho význam

Shrnutí poznatků o barvách a barvových prostorech

Podle frekvencí, vlnových délek světla, která emituje světelný zdroj je možno světlo rozdělit na **achromatické** a **monochromatické**. Světlo je charakterizované některými svými **atributy: barvou, jasem, sytostí a světlostí.**

Známe dva **základní způsoby kombinace** (míšení) barev: **aditivní a substraktivní**.

Barevný (barvový) model je určen sadou **základních barev**, způsobem jejich **míšení** a **pravidly změny barevných charakteristik.**

Mezi nejčastěji využívané modely dále v počítačové grafice patří: **RGB**, **CMY(K), HSB** či **HSV**, **HLS** a **LAB**.

Základní barvy v počítačové grafice jsou: černá, modrá, zelená, tyrkysová, červená, fialová, žlutá a bílá.

Model založený na fyziologii vnímání lidského oka a který v současnosti často používaný, je model **UWB**.

Je možné přecházet z jednoho modelu do jiného zase zpět. Lze využít například při ostření digitálních fotografií, přechodem z RGB na LAB model a zpět.

Odchylce hodnoty pixelu od skutečného jasu pixelu na obrazovce říkáme **gama faktor** a korekci této skutečnosti potom **gama korekce.**

Alfa míchání je funkce pro zobrazení poloprůhledných ploch. **Koeficient průhlednosti** se označuje **jako alfa-složka** a nejčastěji se udává z intervalu <0,1>.

2 OBRAZ A JEHO REPREZENTACE

Cíl Po nastudování kapitoly se

budete se umět orientovat v problematice digitalizace obrazu.
porozumíte principům ořezávání grafických útvarů
.

2.1 Obraz

Čas ke studiu: 0,5 hodina

Cíl Po prostudování tohoto odstavce budete umět

porozumíte pojmu obrazová funkce
budete umět vysvětlit proces digitalizace obrazu
seznámíte se s pojmem diskrétní Fourierova transformace
budete schopni definovat proces kvantizace
budete schopni definovat proces vzorkování
pochopíte vznik aliasu
porozumíte postupu antialiasignu

Výklad

Jedním z důležitých lidských smyslů je zrak a zrakové vnímání okolního světa. To, že jsme schopni

Obraz obdobně jako mnoho nejobvyklejších pojmů je možné vymezit různě, zůstaneme u intuitivního chápání pojmu obraz, a to jako obraz reálného světa promítaného na sítnici oka, film, monitor, plátno kina… Formálně obraz vymezíme matematicky jako ***obrazovou funkci*** :

$$z = f(x, y) \tag{8}$$

Předpokládáme-li omezené rozměry obrazu, lze definiční obor obrazové funkce zapsat jako kartézský součin dvou spojitých intervalů z oboru reálných čísel, které vymezují rozsah obrazu. Obrazová funkce je tedy zobrazení

$$f : \langle x_{min}, x_{max} \rangle \times \langle y_{min}, y_{max} \rangle \rightarrow H \tag{9}$$

Reálná čísla *x, y, pro* která platí $x : x_{\min} \geq x \geq x_{\max}$ a $y_{\min} \leq y \leq y_{\max}$, jsou souřadnice bodu ve dvourozměrném prostoru. Funkce pro tato čísla nabývá určité hodnoty z z oboru hodnot, tj. z $z \in \mathbf{H}$.

Hodnotou obrazové funkce může být jedno reálné číslo (například jas, intenzita zelené barvy, atp.) V počítačové grafice hodnotu obrazové funkce tvoří obvykle více hodnot. Na monitoru počítače trojice červená, zelená a modrá složka tvoří obrazové informace v modelu RGB. Funkce může nabývat hodnot zapsaných jako uspořádaná n-tice údajů $z = [z_1, z_2, \ldots, z_n]$. Definici funkce zapíšeme ve tvaru

$$f : \langle x_{min}, x_{max} \rangle \times \langle y_{min}, y_{max} \rangle \rightarrow \langle H_1 \times H_2 \times \ldots \times H_n \rangle \quad (10)$$

Navíc v počítačové grafice nemáme k dispozici spojitý definiční obor funkce (2.1). Častěji pracujeme v rastru, (představme si čtverečkovaný papír), složeném z obrazových elementů, které nazýváme *pixely (*z anglického ***pi****cture* ***el****ement).* Obdobné pojmy se používají i pro další grafické elementy, prvek textury se jmenuje *texel*, objemový element *voxel*, povrchový element *surfel* atp.

Obor hodnot funkce (2.1) je různě omezen použitou aplikací, reálnými parametry technického zařízení, které vytváří či pořizují obraz. Nejčastěji pracujeme s přibližně 16 miliony barev, s 256 stupni šedi, s 256 barvami.

V následujícím textu si vysvětlíme základní pojmy - vzorkování, kvantování, pojem alias a jak tento jev zmírníme. Často budeme uvažovat obor hodnot funkce (2.1) tvořený pouze s jedinou hodnotou, například odstínem šedi. Zamyslete se, zda černá a bílá jsou odstíny šedé?

Obdobně při výkladu použijeme jednorozměrný definiční oboru obrazové funkce, budeme hovořit o spojité funkci jedné proměnné *f (x)* a jejím diskrétním (nespojitém) obrazu $I_{i,}$ později o spojité funkci dvou proměnných *f(x,y)* a jí odpovídající diskrétní funkci $I_{i,j}$.

2.1.1 Digitalizace

Většinu modelů v počítačové grafice lze popsat vhodnou spojitou funkcí, jak jsme zvyklí v klasické geometrii. Zmínili jsme se již, že digitální obraz, třeba na obrazovce monitoru, je diskrétní. Výchozí úloha, která se váže k získávání počítačového obrazu, je úloh, jak převést spojitou funkci *f (x, y)* na její diskrétní protějšek, funkci $I_{i,j,}$ a to jak v jejím definičním oboru, tak i v oboru jejích hodnot ***H***. Tuto úlohu musí vyřešit každá digitální videokamera nebo digitální fotoaparát. Obraz reálné scény má teoreticky nekonečný rozsah obrazových hodnot, můžeme si téměř neomezeně vybírat jeho libovolný úsek. Konečným výsledkem bude obrázek s daným množství pixelů a s určitým množstvím barev. ***Digitalizací*** nazýváme proces přechodu od *spojitého obrazu* k jeho *diskrétnímu (digitálnímu)* protějšku. Odehrává se ve dvou krocích, a to ***kvantování*** *a* ***vzorkování.***

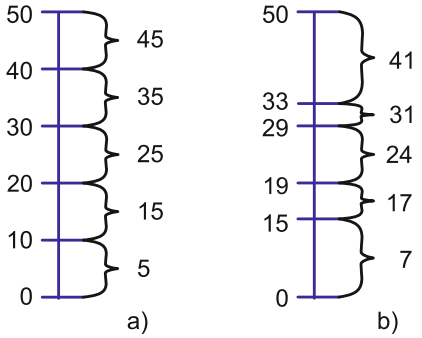
2.1.2 Kvantování

Kvantování probíhá v oboru hodnot obrazové funkce (2.1). Ten je rozdělen na jednotlivé intervaly. Každému intervalu je přidělena jediná zástupná hodnota. Podle způsobu rozdělení původního oboru hodnot rozeznáváme kvantování *uniformní a neuniformní,* viz obrázek 2.1.

Uniformní kvantování používá konstantní délku intervalu. Neuniformní kvantování používá proměnnou délku intervalu. Neuniformní kvantování je méně časté, přestože umožňuje zohlednit nerovnoměrné rozložení hodnot měřené veličiny.

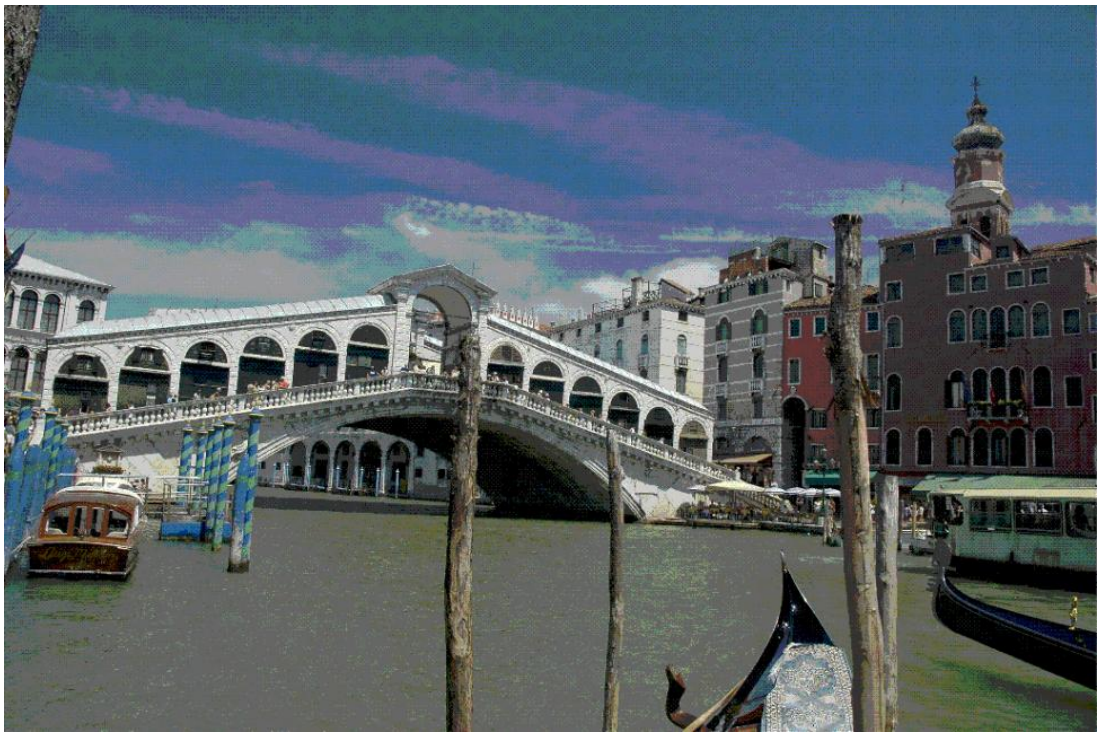
Výběru zástupné hodnoty závisí na mnoha faktorech, používá průměr daného intervalu, jeho vážený průměr, medián, průměr z okrajů, aj.

Při kvantování dochází k chybě, ke ztrátě informace. Množina hodnot, (popřemýšlejte, kolik prvků obsahuje tato množina) je nahrazena jedinou hodnotou. Tuto ztrátu nazýváme *kvantizační chyba* a v počítačové grafice se projevuje na plochách s malou změnou gradientu (odstínu jedné barvy, nejčastěji oblohy) jako náhlý přechod barev. Tento jev se mnohdy nazývá plakátování, jak demonstruje obrázek (Obr 10).



Obr 10. Uniformní (vlevo) a neuniformní kvantování.

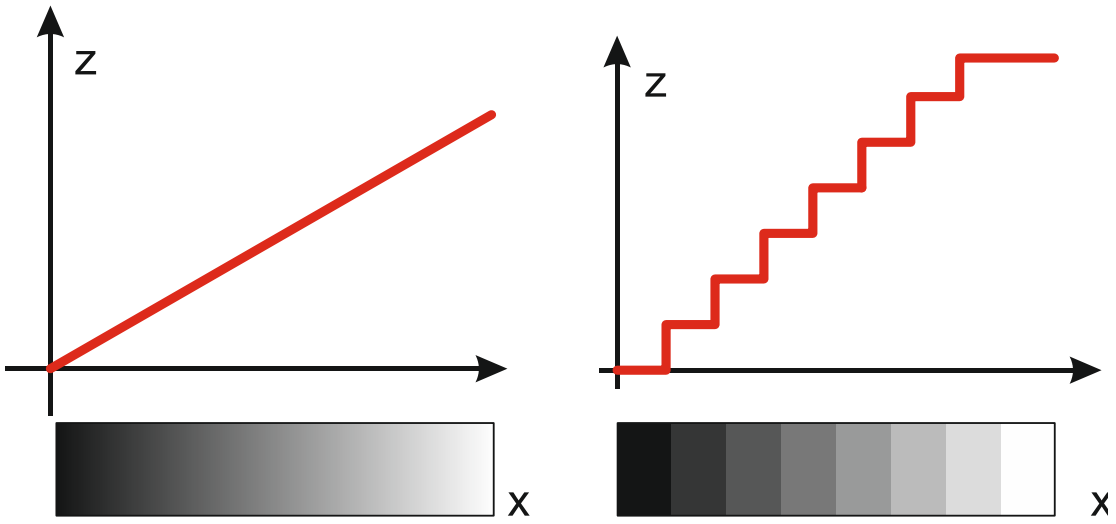
Tento jev byl poprvé popsán na konci osmnáctého století fyzikem Ernstem Machem, ve fyzice mu říkáme Machovy proužky *(Mach-band effect).*



2.1.3 Vzorkování

Vzorkováním *(sampling)* spojité funkce *f (x)* rozumíme zaznamenávání hodnot - vzorků, v předem daných intervalech, tak jak je naznačeno na obr. 2.3. Jednorozměrnou funkci získanou pravidelným vzorkováním budeme označovat I_i a vzdálenost dvou vzorků označíme Δx. Původní spojitá funkce může být definována na libovolném intervalu $x \in \langle x_0, x_1 \rangle$. Vzorky budeme indexovat následujícím způsobem:

$$I_i = f(x_0 + i\Delta x), i = 0, 1, \ldots \tag{11}$$



Obr 11. Kvantizační chyba. Původně hladký barevný přechod (vlevo) je nahrazen skokovým přechodem (vpravo). V obraze vznikají hrany subjektivně hrany, které v původním signálu nebyly přítomné. Viz pruh pod pravým obrázkem.

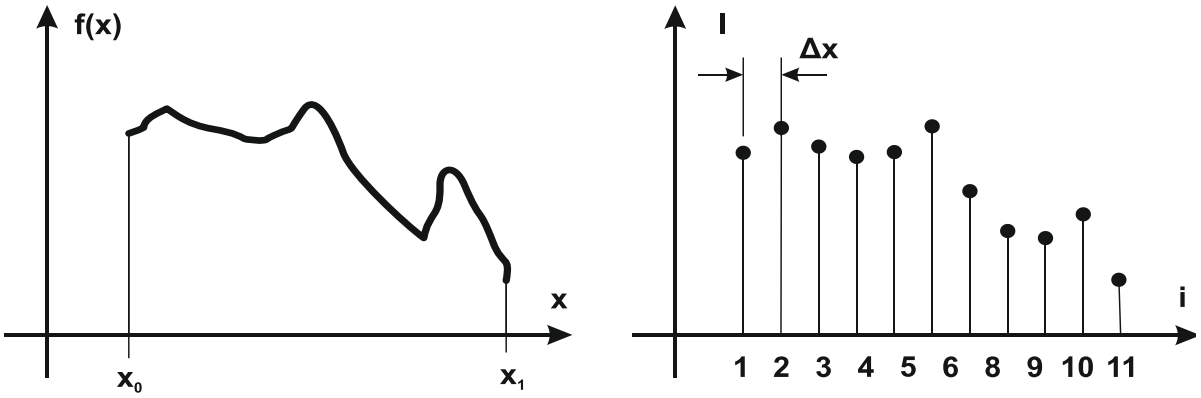
Vzorkování (Obr 12) je běžnou technickou záležitostí: videokamera snímá obraz v diskrétních intervalech, digitální teploměr poskytuje hodnoty v definovaných časových intervalech, hudba ve formě souboru MP3 je posloupnost čísel reprezentujících úroveň signálu snímaného s poměrně vysokou frekvencí. Obraz v počítačové grafice je mnohdy výsledkem vzorkování trojrozměrné scény.

Při vzorkování dochází ke ztrátě informace (Obr 11). Pokud byla například hodnota funkce konstantní a v okamžiku $x_0 + \Delta x/2$ došlo ke změně její hodnoty. Pakliže je funkce vzorkována s krokem Δx, hodnotě $f(x_0)$ odpovídá vzorek I_0 a hodnotě $f(x0 + \Delta x)$ vzorek I_1. Zjistíme, jak se změnila její hodnota v polovině vzorkovaného intervalu? Nezjistíme, jelikož tato skutečnost není v řadě výstupních vzorků zaznamenána.

Tento způsob vzorkování, kdy je hodnota vzorku zaznamenána v pouze jediném bodě, se nazývá bodové vzorkování. Plošné vzorkování *(area sampling)* zaznamenává průměrnou hodnotu funkce během vzorkovacího intervalu Δx a hodnota

vzorku se určí měřením ze všech hodnot jako jejich průměr: $I_i = \frac{1}{\Delta x} \int_{x_0 + i\Delta x}^{x_0 + (i+1)\Delta x} f(t)dt$.

Definovali jsme vzdálenost dvou vzorků jako Δx. *Vzorkovací frekvencí* f_S pak rozumíme



Obr 12. Vzorkování

hodnotu $f_s = \frac{1}{\Delta x}$. Jednotkou je počet vzorků za sekundu času [Hz] či počet vzorků na jednotku vzdálenosti [dpi], zde na palce (dot per inch). Čím bude vyšší vzorkovací frekvence, více vzorků, tím bude paměťová náročnost reprezentace vyšší. V případě obrazu se bude zvětšovat jeho rozlišení.

2.1.4 Fourierův obraz

Fourierova transformace slouží k převodu originálu obrazu do duálního prostoru, ve kterém se určité operace s obrazem provádějí snadněji je v něm snazší pochopit jisté jevy a vlastnosti obrazu. V této kapitole budeme používat dvě zobrazení. Budeme pracovat s obrazem, tak jak ho známe v diskrétní matici pixelů. Tomuto zobrazení budeme říkat *prostorová oblast. Fourierův obraz* je zobrazení původního obrazu *ve frekvenční oblasti.* Fourierův obraz obecně tvoří součet nekonečně mnoha sinusových signálů mající různou amplitudu a různý fázový posun. Nežádoucí šum se projevuje jako vysoké frekvence. Nežádoucí nízkofrekvenční signál se nazývá *alias* a jeho vznik *aliasing*. Pro snazší pochopení budeme většinu následujících pojmů definovat v jednom rozměru; zobecnění do vyšších rozměrů je snadné.

2.1.5 Shannonův vzorkovací teorém a frekvenčně omezená funkce

Frekvenčně *omezená funkce (bandlimited function)* má konečné amplitudové spektrum. V konečném amplitudovém spektru existuje nejvyšší frekvence (označme ji f_{max}) taková, že pro všechny frekvence *u*, pro které platí, že $u > f_{max}$ je $|F(u)| = 0$. Amplituda těchto frekvencí je nulová a tyto frekvence nenesou žádnou energii. Nejjednodušším případem frekvenčně omezené funkce je samozřejmě přímo funkce *sin(x),* které ve Fourierově spektru odpovídá jediný bod.

Nejvyšší nenulové frekvenci f_{max} se říká *Nyquistovo kritérium.* Vzorkovací frekvenci f_s a nejvyšší frekvenci v signálu uvádí do vztahu *Shannonova vzorkovací věta* (též Shannonův vzorkovací teorém), která zní:

Signál spojitý v čase je plně určen posloupností vzorků odebíraných ve stejných intervalech Δx, je-li jejich frekvence $f_s = 1/\Delta x$ větší nežli dvojnásobek nejvyšší frekvence v signálu f_{max} , tj. je-li

$$f_s > 2f_{max} \qquad (12)$$

2.1.6 Alias

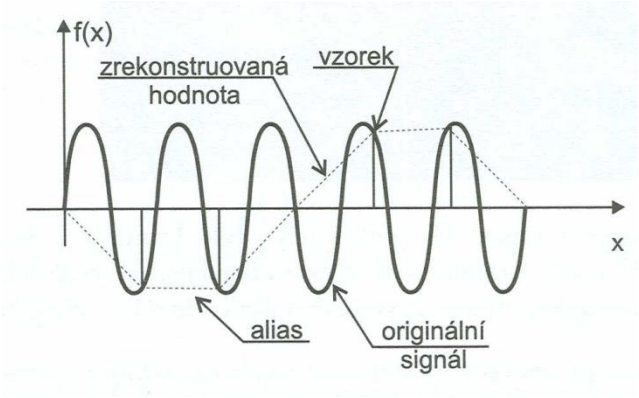
Alias je jedním z nejdůležitějších pojmů počítačové grafiky (Obr 13). Vzniká při rekonstrukci signálu vzorkovaného pod Nyquistovým limitem a projevuje se jako nová, nízkofrekvenční informace, která nebyla v původním signálu přítomna. Jde o případy, kdy originální funkce obsahuje detaily, které není možné v rastru zobrazit. Alias vzniká ve dvou případech:

1. Pokud je původní funkce frekvenčně neomezená. Neexistuje žádná maximální frekvence a funkci není tedy možné v diskrétní mřížce rastru reprezentovat přesně. Zde se jedná například o hrany šikmých stran zobrazovaných předmětů či o šikmé úsečky. Ty se nezobrazí jako hladká hrana, ale jako schodovitá hrana nebo čára.

Je-li původní funkce frekvenčně omezená, tj. v jejím Fourierově spektru existuje určitá maximální frekvence f_{max}, a tuto funkci vzorkujeme s frekvencí menší nežli $2\ f_{max}$, tedy pod Nyquistovým limitem.

Příkladem aliasu v praktickém životě je televizní obrazovka snímaná kamerou. Vzhledem k tomu, že kamera snímá obraz v diskrétních časových intervalech a obrazovka promítá po půlsnímcích, dochází k interferenci frekvencí a následnému aliasu, který se na obrazovce projevuje jako tmavé, různou rychlostí nahoru a dolů se pohybující pruhy či blikání. Jiná Další příklad je například kolo jedoucího auta. Při jisté rychlosti otáčení se zdá, že se otáčí opačným směrem. Tento případ je projevem takzvaného časového aliasu *(temporal alias).*

V počítačové grafice se setkáváme s aliasem v mnoha situacích, například při vzorkování textur a při zobrazování objektů na jejich hranách. Zubatému zobrazení čar a okrajů polygonů



Obr 13. Vznik aliasu

se v počítačové grafice říká „zubatice" *(jaggies).* Při vzorkování šachovnice se může stát, že všechny vzorky zasáhnou pouze černé políčko a výsledkem takového vzorkování pak bude jednobarevná plocha. Při vzorkování textury, která obsahuje hustý pravidelný vzorek, se může někdy alias projevit jako tzv. moire. Jiným příkladem je objevování se a mizení malých objektů; tj. objektů, které jsou svou velikostí po průmětu do obrazu srovnatelné s velikostí pixelu, pro tyto objekty se vžil název *crowlies.*

V počítačové animaci se setkáváme především s časovým aliasem, kde může mít dosti bizarní projevy. *Například* běžící postava se může pohybovat a při tom nehýbat nohama, případně s nimi pohybovat v opačném směru aj.

2.1.7 Antialiasing

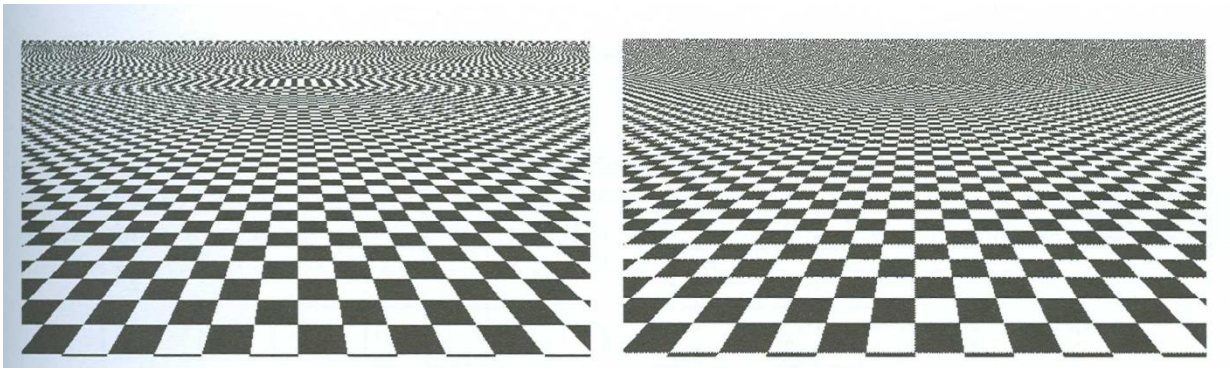
Odstranění či zmenšení aliasu se říká antialiasing. Nejjednodušší cestou je odstranit z obrazu informaci, která *nelze* vzorkovat, tj. odstranit vysoké frekvence. Pokud bude funkce neomezená, je možné její vysoké frekvence eliminovat odříznutím. Toto oříznutí se provede filtrem, který propustí jen nízké frekvence, vysoké frekvence potlačí.U digitálních kamer a fotoaparátů, aby byl potlačen vznik nežádoucího aliasu, moiré, je povrch snímacího čipu potažen mírně matným filtrem, který rozostří vykreslovaný obraz. Takto upravený obraz vnímá pozorovatel jako rozostřený, ostré linie a přechody jsou rozmazané.

Druhou cestou zmenšení aliasu je zvýšení hustoty vzorkování. Alias většinou (výjimkou jsou frekvenčně omezené funkce) nemůžeme odstranit, ale můžeme ho částečně potlačit. V počítačové grafice se nejčastěji používají dvě metody. První spočívá v posunutí aliasu k vyšším frekvencím (použijeme vyšší frekvenci vzorkování), druhá metoda převádí alias na šum. Ten se do obrazu zavede například pomocí generátoru náhodných čísel.

Pravidelné vzorkování s vyšší frekvencí

FSAA, neboli celoobrazovkový antialiasing *(Full Screen AntiAliasing - FSAA)* je dnes běžnou záležitostí ve většině grafických karet. Princip této metody tkví v získání obrazu ve vyšším rozlišení (tedy v jeho přesnější reprezentaci), jeho filtraci a zmenšení. Uvedená metoda neodstraní alias, ale pouze ho posune k vyšším frekvencím. Filtrace se projevuje *rozmazáním (blur)* vysokých frekvencí.

Stochastické vzorkování



Obr 14. Alias

Pravidelné vzorkování přináší do obrazu novou informaci. Filtrace alias částečně rozmaže. Lidské vnímání je tolerantní k šumu. Je daleko citlivější na pravidelné chyby. Převod aliasu na šum je jedna z nejpěknějších technik.

Roztřesení *(jittering)* je patrně nejčastěji používaným algoritmem stochastického antialisingu. Tato metoda generuje dobrou aproximaci optimálního rozložení vzorků. Může se používat přímo, jako bodové vzorkování, nebo jako vzorkování s vyšší frekvencí.

Shrnutí pojmů

Obrazová funkce je zobrazení $f : \langle x_{min}, x_{max} \rangle \times \langle y_{min}, y_{max} \rangle \rightarrow H$

Hodnotou obrazové funkce může být jedno reálné číslo (například jas, intenzita zelené barvy, atp.) V počítačové grafice hodnotu obrazové funkce tvoří obvykle více hodnot.

Digitalizací nazýváme proces přechodu od *spojitého obrazu* k jeho *diskrétnímu (digilánímu)* protějšku. Odehrává se ve dvou krocích, a to ***kvantování a vzorkování.***

Kvantování probíhá v oboru hodnot obrazové funkce. Ten je rozdělen na jednotlivé intervaly. Každému intervalu je přidělena jediná **zástupná hodnota.**

Uniformní kvantování používá konstantní délku intervalu. **Neuniformní kvantování** používá proměnnou délku intervalu.

Vzorkováním *(sampling)* spojité funkce *f (x)* rozumíme zaznamenávání hodnot - vzorků, v předem daných intervalech. Při vzorkování dochází ke ztrátě informace.

Při vzorkování dochází ke **ztrátě informace.**

Fourierova transformace slouží k převodu originálu obrazu do duálního prostoru, ve kterém se určité operace s obrazem provádějí snadněji je v něm snazší pochopit jisté jevy a vlastnosti obrazu

Frekvenčně *omezená funkce* *(bandlimited function)* má konečné amplitudové spektrum.

Nejvyšší nenulové frekvenci f_{max} se říká ***Nyquistovo kritérium*.**

Alias vzniká při rekonstrukci signálu vzorkovaného **pod Nyquistovým limitem** a projevuje se jako nová**, nízkofrekvenční informace**.

V počítačové animaci se setkáváme především s **časovým liasem.**

Antialiasing je proces odstranění či zmenšení aliasu.

Antialiasing lze provést

- odstraněním vysokých frekvencí
- zvýšením hustoty vzorkování

Zvýšení hustoty vzorkování provedeme například **pravidelným vzorkováním s vyšší frekvencí** a stochastickým antialisingem. Příkladem stochastického antialiasingu je **Roztřesení** *(jittering*

Otázky

1. Co znamená proces digitalizace?
2. K čemu se využívá kvantování?
3. Čím se liší uniformní a neuniformní kvantování

4. K čemu využijete vzorkování?
5. K jakým účelům se využívá diskrétní Fourierova transformace?
6. Co je to alias a kdy vzniká.
7. Jaké jsou hlavní postupy při odstraňování aliasu?

3 DVOUROZMĚNÉ OBJEKTY

Čas ke studiu: 20 hodin

Cíl Po prostudování tohoto odstavce budete umět

Definovat. Základní grafické prvky
matematicky popsat základní geometrické prvky…
umět rozlišit mezi vektorovou a rastrovou grafikou
porozumíte procesu rasterizace
budete umět aplikovat základní algoritmy rasterizace
porozumíte procesu ořezávání

3.1 Základní grafické objekty

Čas ke studiu: 0,1 hodiny

Cíl Po prostudování tohoto odstavce budete umět

- Definovat. Základní *grafické prvky*
- rozlišit mezi liniovým a plošným charakterem útvar**u**
- definovat rastrový obraz….

Výklad

Mezi *základní grafické prvky (output primitives)* řadíme úsečky, lomené čáry, kružnice, elipsy, mnohoúhelníky, křivky a textové řetězce říkáme a jsou obsaženy ve většině programů pro kreslení v rovině. Základní prvky mají liniový charakter (úsečky, křivky) nebo plošný charakter, pak u nich rozlišujeme obrys a vnitřek s rozličnými výplněmi.

Výsledkem kreslících algoritmů jsou bud' posloupnosti bodů (pixelů), nebo posloupnosti křivek. V prvním případě tak získáme *rastrový obraz,* ve druhém případě algoritmy vytváří *vektorový* obraz. Vektorový obraz může být buď vykreslován na příslušných kreslicích zařízeních, např. plotterech, nebo převeden do rastrové podoby.

V současné počítačové grafice se povětšinou pracuje s rastrovým obrazem. Pak je třeba nalézt ty pixely, které reprezentují tvar a polohu vykreslovaného grafického prvku a přiřadit jim příslušnou barvu. Rasterizací nazýváme určování pozice a barvy těchto.

V následném textu se budeme věnovat postupy při rasterizaci úsečky, lomené čáry, kružnice a elipsy. Na závěr probereme tématiku vyplňování a oblastí a ořezávání grafických prvků.

Shrnutí poznatků

Mezi z*ákladní grafické prvky (output primitives)* řadíme úsečky, lomené čáry, kružnice, elipsy, mnohoúhelníky, křivky a textové řetězce.

Základní prvky mají **liniový** charakter (úsečky, křivky).

U **plošného** charakteru prvku rozlišujeme obrys a vnitřek s rozličnými výplněmi.

Rasterizací nazýváme určování pozice a barvy těchto pixelů

3.2 Matematický popis elementárních útvarů

Čas ke studiu: 2 hodiny

Cíl Po prostudování tohoto odstavce budete umět

- definovat bod, přímku, kružnici a elipsu v rovině.
- matematicky vyjádřit jejich popis…
- stanovit parametrickou rovnici přímky
- porozumět parametrické rovnici přímky v prostoru

Výklad

3.2.1 Bod

Bodem A v prostou budeme rozumět bod, jehož poloha je určena uspořádanou trojicí čísel $[a_x, a_y, a_z]$, kterou nazýváme souřadnice bodu v prostoru. Čísla a_x, a_y, a_z jsou v pořadí x-ová, y-ová a z-ová souřadnice.

Vzdálenosti bodů A a B v Euklidovském prostoru budeme označovat číslo

$$|AB| = \sqrt{(a_x - b_x)^2 + (a_y - b_y)^2 + (a_z - b_z)^2} \tag{13}$$

Vektor

Vektorem $\mathbf{a} = \overrightarrow{AB}$ v prostoru rozumíme orientovanou úsečku vedenou z bodu A do bodu B. Vektor určuje jeho velikost a směr. Nechť bod A má souřadnice $[A_x, A_y, A_z]$, bod B souřadnice $[B_x, B_y, B_z]$ pak vektor má souřadnice

$$\begin{aligned} A &= [a_x, a_y, a_z] \\ a_x &= B_x - A_x, \\ a_y &= B_y - A_y, \\ a_z &= B_z - A_z. \end{aligned} \tag{14}$$

Velikost vektoru $|\mathbf{a}|$ je určena vztahem

$$|\mathbf{a}| = [a_x^2, a_y^2, a_z^2]. \tag{15}$$

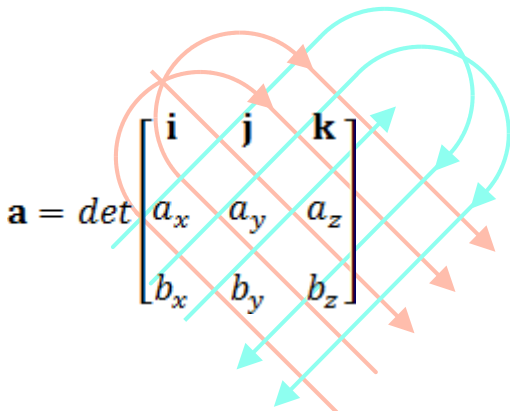
Vektorový součin **a** dvou vektorů **b** *a* **b** je definován takto:

$$\mathbf{a} = \mathbf{b} \times \mathbf{c} = [(a_y b_z - a_z b_y), (a_z b_x - a_x b_z), (a_x b_y - a_y b_x)]. \quad (16)$$

Výsledek vektorového součinu je opět vektor, který je kolmý k oběma vektorům. V pravoúhlé (ortogonální) souřadnicové soustavě orientaci os x, y a z vyjadřujeme pomocí jednotkových vektorů **i**, **j** a **k** ve směru jednotlivých os. Říkáme jim také vektory báze neboli bázové vektory. Tyto vektory definujeme tak, aby vycházely z počátku souřadné soustavy (obr. 16).

Vektorový součin dvou vektorů lze zapsat jako determinant matice, jejíž řádky tvoří postupně vektory báze, souřadnice vektoru **b** a vektoru **c**

$$\mathbf{a} = det \begin{bmatrix} i & j & k \\ a_x & a_y & a_z \\ b_x & b_y & b_y \end{bmatrix}. \quad (17)$$



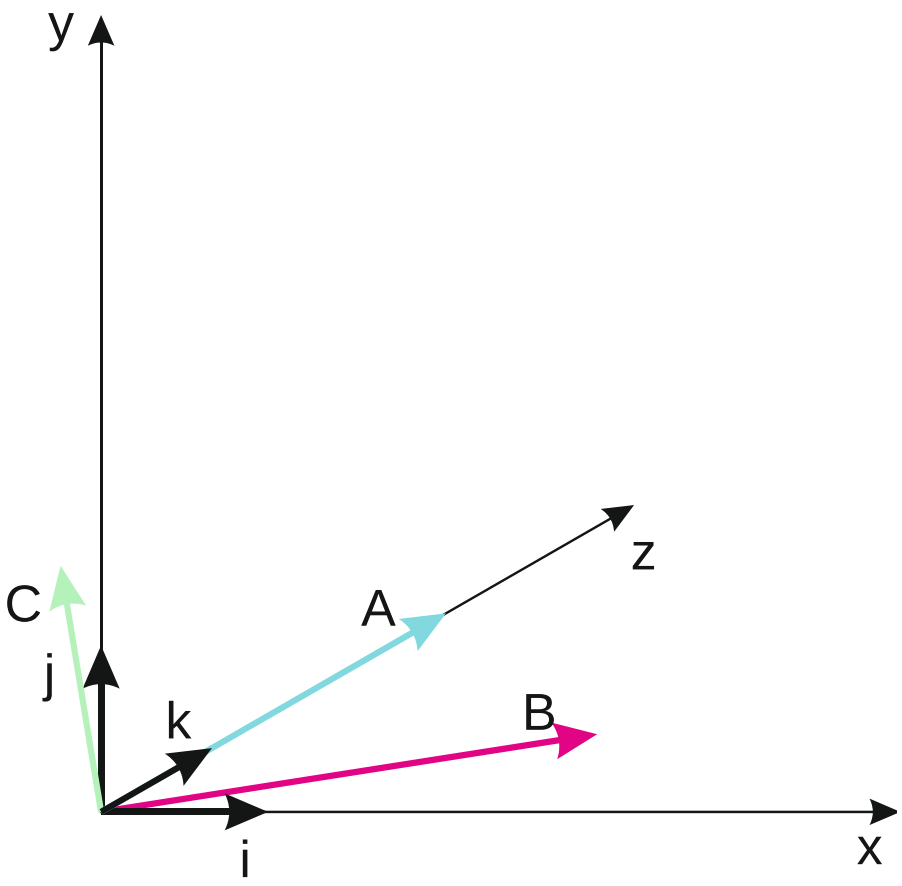
Obr 15. Výpočet determinantu matice 3x3

Na obrázku (Obr 15) jsou červeně označeny hlavní úhlopříčky matice a modře vedlejší úhlopříčky. Členy ležící na každé úhlopříčce vynásobíme mezi sebou, výsledek bude kladný, pokud počítáme členy na hlavních úhlopříčkách a záporný, pokud počítáme členy na vedlejších úhlopříčkách. Výsledek bude součtem všech těchto prvků.

$$\mathbf{a} = ia_y b_z + jb_x a_z + kb_y a_x - ia_z b_y - jb_z a_x - kb_x a_y \quad (18)$$

Vytkneme jednotlivé vektory báze

$$\mathbf{a} = i(a_y b_y - a_z b_y) + j(a_z b_x - a_x b_z) + k(a_x b_y - a_y b_x). \quad (19)$$



Obr 16. Vektorový součin

Porovnáme-li konečnou úpravu s rovnicí, vidíme, že jsme dosáhli shodného výsledku.

Skalární součin vektorů $\mathbf{u}[u_x, u_y, u_z]$ a $\mathbf{v}[v_x, v_y, v_z]$ je definován takto

$$\mathbf{u}.\mathbf{v} = u_x v_x + u_y v_y + u_z v_z$$

Skalární součin dvou vektorů je prosté číslo, skalár.

Geometricky má skalární součin tento význam

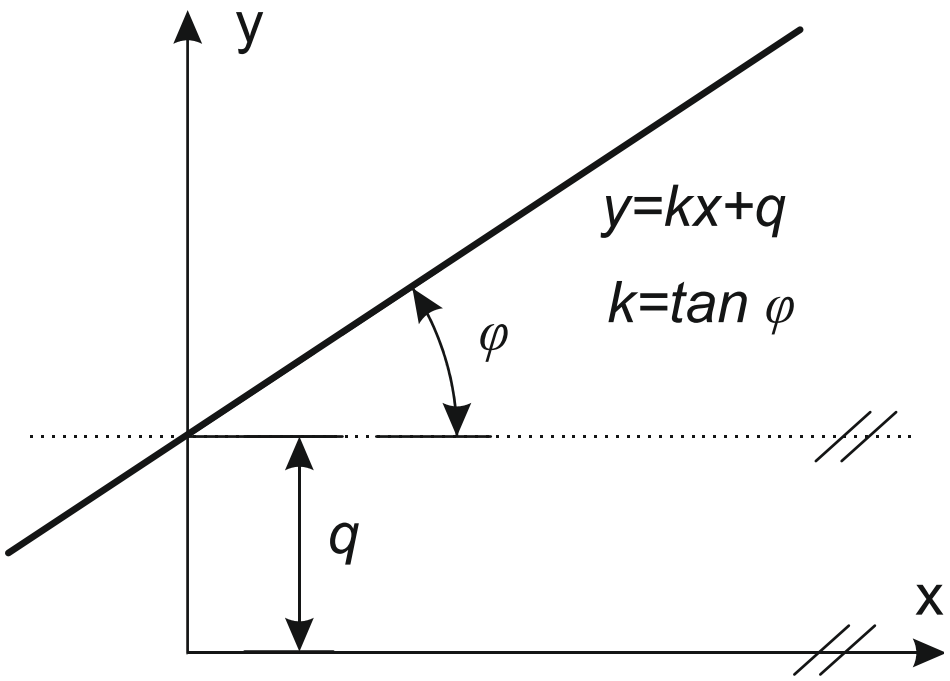
$$\mathbf{u}.\mathbf{v} = |\mathbf{u}||\mathbf{v}|\cos\varphi$$

Kde φ je úhel, který vektory svírají.

3.2.2 Přímka v rovině

Rovnice přímky ve dvojrozměrném prostoru, rovině, má tvar

$$y = kx + q. \tag{20}$$



Obr 17. Rovnice přímky a její parametry

Číslu k říkáme směrnice přímky, q je posunutí přímky vůči počátku souřadné soustavy. Směrnice je rovna tangentě úhlu, která svírá přímka s osou x.

Přímka je plně určena i dvěma body, kterými prochází. Tyto body nechť mají souřadnice $A = [x_a, y_a]$ a $B = [x_b, y_b]$. Směrnic k a posunutí q vypočteme z následujících vztahů

$$k = \frac{y_a - y_b}{x_a - x_b} \tag{21}$$

$$q = \frac{x_b y_a - x_a y_b}{(x_a - x_b)} \tag{22}$$

Rovnice pak má celkový tvar

$$y = \frac{y_a - y_b}{x_a - x_b} x + \frac{x_b y_a - x_a y_b}{x_a - x_b} \tag{23}$$

Případně tvar

$$y(x_a - x_b) - x(y_a - y_b) + x_b y_a - x_a y_b = 0 \tag{24}$$

Parametrickou rovnici přímky můžeme odvodit z následujících úvah. Vektor $\vec{a} = B - A$ směřuje z bodu A do bodu B. Jak jsme se již zmínili, jedná se vlastně o orientovanou úsečku. Přímka prochází

bodem A. Bodu B dosáhneme, pokud k bodu A přičteme vektor a. Tedy $B = A + \vec{a}$. Například bod $B' = A - \vec{a}$ je bod souměrný k bodu B se středem souměrnosti A. Abychom dosáhli bodu B´ vynásobili jsme vektor a číslem mínus jedna a tento vektor přičetli k souřadnicím bodu A. Obdobně: vynásobíme-li vektor a kladným číslem menším než jedna a přičteme výsledný vektor k souřadnicím bodu A, pak obdržíme jistý bod C, který leží na přímce mezi body A a B. Pokud vektor a vynásobíme kladným číslem větším než jedna a výsledný vektor přičteme k souřadnicím bodu A, pak výsledný bod D bude ležet na přímce AB za bodem B. Tedy libovolný bod ležící na přímce AB má souřadnice

$$P_{ab} = A + t\vec{a} = A + t(B - A) \tag{25}$$

3.2.3 Přímka v prostoru

V třírozměrném prostoru se nejčastěji používá parametrický tvar rovnice přímky. Ten je shodný s tvarem rovnice (25). Rozepsanou po složkách pak obdržíme

$$x = q_x + tk_x \tag{26}$$

$$y = q_y + tk_y \tag{27}$$

$$z = q_z + tk_z \tag{28}$$

Zde

$$P_{ab} = [x, y, z] \tag{29}$$

$$A = [q_x, q_y, q_z] \tag{30}$$

$$B - A = [k_x, k_y, k_z] \tag{31}$$

Prochází-li přímka v prostoru dvěma body, pak souřadnice prvého jsou, udávají parametry q, lze je chápat jako posunutí vůči počátku souřadné soustavy. "Směrnice" přímky jsou vtaženy vůči parametru t. Jsou to vlastně prvky vektoru $\vec{a}$

$$k_x = x_b - x_a \tag{32}$$

$$k_y = y_b - y_a \tag{33}$$

$$k_z = z_b - z_a. \tag{34}$$

Bodem A v prostou budeme rozumět **bod**, jehož poloha je určena uspořádanou trojicí čísel $[a_x, a_y, a_z]$, kterou nazýváme **souřadnice bodu** v prostoru.
Vektor $\mathbf{a} = \overrightarrow{AB}$ v prostoru je orientovaná úsečka vedená z bodu A do bodu B
Vektor určuje jeho **velikost a směr**.

Vektorový součin $\mathbf{a}$ dvou vektorů $\mathbf{b}$ a $\mathbf{b}$ je definován takto:

$$\mathbf{a} = \mathbf{b} \times \mathbf{c} = [(a_y b_z - a_z b_y), (a_z b_x - a_x b_z), (a_x b_y - a_y b_x)]$$

Skalární součin vektorů $\mathbf{u}[u_x, u_y, u_z]$ a $\mathbf{v}[v_x, v_y, v_z]$ je definován takto
$\mathbf{u}.\mathbf{v} = u_x v_x + u_y v_y + u_z v_z$

$\mathbf{u}.\mathbf{v} = |\mathbf{u}||\mathbf{v}| \cos\varphi,$

Kde φ je úhel, který vektory svírají.

Rovnice přímky v rovině má tvar $y = kx + q.$

Parametrická rovnice přímky procházející body A a B má tvar $P_{ab} = A + t\vec{a} = A + t(B - A)$

3.3 Úsečka a lomená čára a jejich rasterizace

Čas ke studiu: 4 hodiny

Cíl Po prostudování tohoto odstavce budete umět

- Navrhnout způsob kreslení lomené čáry
- definovat atributy úsečky
- vysvětlit algoritmus DAA
- aplikovat vykreslení úsečky Bresenhamovým algoritmem
- popsat způsob vykreslování přerušované čáry
- vysvětlit záludnosti při kresbách silných čar

Výklad

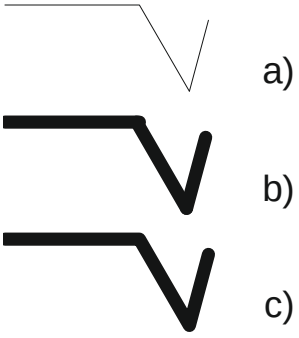
Úsečka patří mezi nejjednodušší grafické prvky. Z úseček lze skládat *lomené čáry (polyline),* jimiž lze nahradit složitější křivočaré obrazce. Protože úsečka náleží k základním stavebním prvkům počítačové grafiky, je vhodné, aby vykreslení úsečky bylo prováděno úsporně. Lomenou čáru pak tvoří posloupnosti úseček, kdy koncový bod jedné úsečky je počátečním bodem úsečky navazující. Lomenou čáru lze popsat posloupností bodů nebo využít následnosti úseček a lomenu čáru definovat vždy počátečním bodem a posloupností relativních přírůstků.

Úsečku popisujeme pomocí koncových bodů $[x_1, y_1]$ *a* $[x_2, y_2]$. *J*e výhodné, aby byly tyto body uspořádány zleva doprava. Je možno také definovat úsečku počátečním bodem $[x_1,y_1]$ a vektorem rozdílů souřadnic $(\Delta x, \Delta z) = (x_2 - x_1, y_2 - y_1)$. Neceločíselné hodnoty jsou před vykreslením v rastru zaokrouhleny.

Popis úsečky doplňují *atributy.* Udávají *sílu* (tloušťku) a typ *přerušované čáry.* Nejtenčí možnou čáru - vlasovou (hair line) značí tloušťka jedna (nebo v některých programech tloušťka nula).

Pro kresbu přerušované čáry se používá *vzor (pattern)* přerušované čáry. Ten je tvořen sudým počtem úseků. Vzor začíná plným úsekem a končí prázdným. Pro přerušovanou čáru vzor tvoří dva úseky, pro čerchovanou má vzor čtyřmi úseky.

Lomenou čáru převedeme na úsečky. V případě, že má lomená čára větší tloušťku (Obr 18), je problematické napojování jednotlivých úseček. Lomenou čáru s různými atributy nazýváme *dráha (path).*

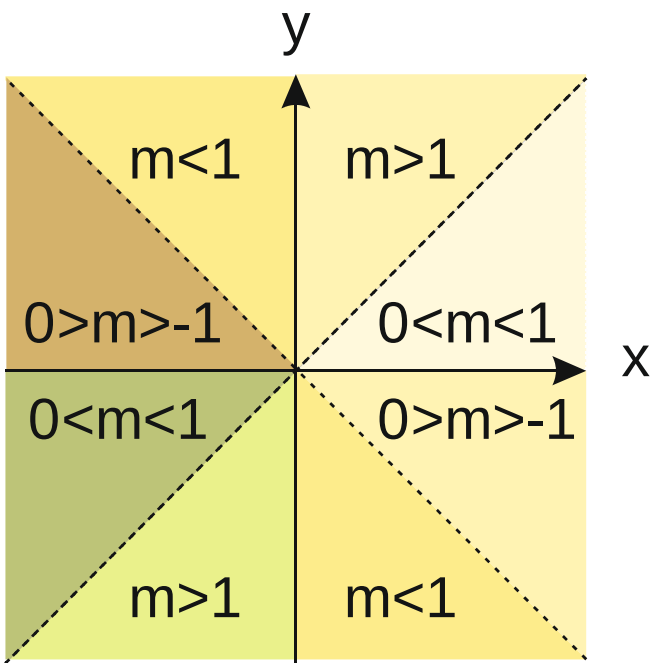


Obr 18. Zakončení lomené čáry

3.3.1 Rasterizace úsečky

Pohybujeme se ve světě, kde se můžeme přemisťovat jen po celých krocích, nemůžeme udělat půlrok. Jako bychom lezli po žebříku, můžeme stoupnout vždy jen na následující příčku. V rastrovém světě se také můžeme pohybovat jen po celých axelech a to jak ve směru osy x tak i ve směru osy y. Také úsečku budeme vykreslovat tak, že bude řešit, který pixel bude vykreslen a který ne. Souřadnice každého bodu (pixelu) tvoří x-ová a y-ová souřadnice, Podle sklonu úsečky je vzorkována s konstantním celým krokem vždy v jedné z os *x a y*. Sklon úsečky je vyjádřen její *směrnicí k* (viz obr. 3.1). Hodnota směrnice je dána podílem rozdílů souřadnic koncových bodů úsečky:

$$k = \frac{\Delta y}{\Delta x} = \frac{y_2 - y_1}{x_1 - x_2}. \tag{35}$$

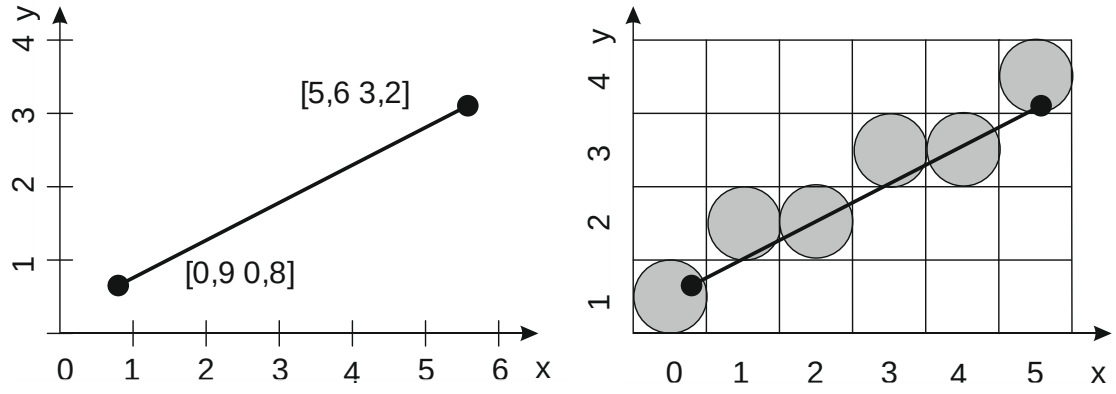


Obr 19. Velikost směrnice dle sklonu přímky

Pokud je $|k| < 1$, úsečka přiléhá k ose *x* a bude proto vzorkována na ose *x* s krokem jeden pixel.

Naopak, na ose *y* jsou vzorkovány ty úsečky, u nichž absolutní hodnota směrnice je větší než jedna. *Řídicí* osa je ta, v jejímž směru se vzorkuje, druhá osa je *vedlejší* osa. Diagonální úsečky (|*k*| = 1) mohou mít řídicí osu libovolnou. Na obr. Obr 19 jsou barevně vyznačeny jednotlivé části roviny společně s možnou velikostí směrnice úsečky.

Úsečka je obecně zadána neceločíselnými souřadnicemi; také směrnice bývá necelé číslo. Většina algoritmů pro kresbu úsečky ovšem předpokládá celočíselné vstupní souřadnice. Chyba zaokrouhlením souřadnic koncových bodů není významná (Obr 18), pokud není porušena návaznost tahu u lomené čáry.



Obr 20. Spojité a rastrové vykreslení úsečky

3.3.2 Algoritmus DDA

Algoritmus DAA (vychází z opakovaného přičítání konstantních přírůstků k oběma souřadnicím. První přírůstek je přičten k počátečnímu bodu úsečky. Přírůstek na řídicí ose volíme roven jednomu pixelu. Neceločíselný přírůstek na druhé ose je roven směrnici k v pokud je řídicí osa *x* a $1/k$ pokud

je řídicí osa y. Souřadnici na vedlejší ose zaokrouhlíme na celé číslo, teprve pak můžeme pixel vykreslit.

Z koncových bodů [x_1, y_1] a [x_2, y_2] urči směrnici m podle vzorce (3.1)

1. Proměnnou bod [*x*, *y*] inicializuj hodnotou [x_1, y_1]
2. Dokud je $x < x_2$, opakuj:

   2a. Vykresli bod $[x, zaokrouhlené(y)]$

   2b. $x = x_{k+1} = x_k + 1$

   2c. $y = y_{k+1} = y_k + m$

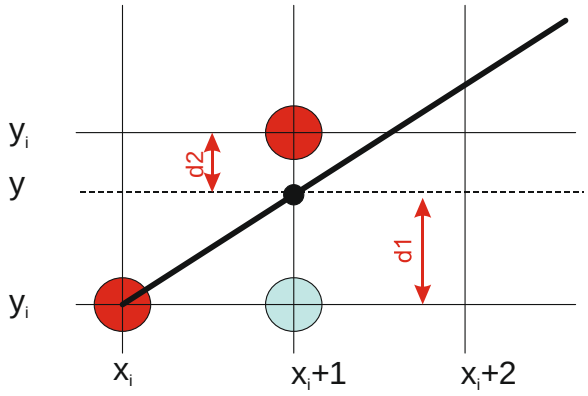
   2d. Opakuj, dokud nedosáhneš koncový bod

Využijeme toho, že můžeme následnou souřadnici vyjádřit pomocí přírůstku a současné hodnoty souřadnice x a y.

Vykreslovací algoritmy mají základ z geometrie a algebry, nevyužívají ale přímo základní vztahy, které jsou obvykle výpočetně náročné, jelikož využívají neceločíselnou aritmetiku. Rychlé algoritmy mají své jádro postaveno na využití celočíselné aritmetiky a nejlépe na operacích přičítání a odečítání.

3.3.3 Vykreslení úsečky Bresenhamovým algoritmem

J. E. Bresenham vyvinul velmi efektivní algoritmus. Při rasterizaci úsečky nachází body ležící nejblíže originální úsečce jen pomocí celočíselných operací. Postup vysvětlíme na příkladu podle obrázku 3.3. Vidíme část rastru, ve kterém mají být vykresleny body úsečky. Řídící osou je zde osa x. Na počátku máme vykreslen levý krajní bod úsečky o souřadnicích $[x_i, y_i]$. Naším úkolem je rozhodnout, zda následující bod má být vykreslen se shodnou souřadnicí y, nebo se souřadnicí $y + 1$. Vybereme ten pixel, jehož vzdálenost od originálu úsečky je menší (Obr 21).



Obr 21. Výběr pixelu při vykreslování úsečky

Pokud již byl vykreslen pixel o souřadnicích [x_i, y_i], pak vpravo existují dva pixely ležící na pozicích $[x_i, +1, y_i]$ a $[x_i + 1, y_i + 1]$. Označíme, ve shodě s obrázkem, d_1 a d_2 rozdíly mezi souřadnicemi y středů uvedených pixelů a souřadnicí y na úsečce v bodě $x_i + 1$. Dosazením souřadnice x_i do obecné rovnice přímky $y = kx + q$ kde k je směrnice a q posun na ose y, získáme souřadnici y

$$y = k(x_i + 1) + q \tag{36}$$

Pak pro rozdíly platí

$$d_1 = y - y_i = (x_i + 1) + q - y_i \; a \tag{37}$$

$$d_2 = y_i + 1 - y = y_i + 1 - k(x_i + 1) - q. \tag{38}$$

Rozdíl těchto dvou vzdáleností vyjádříme jako

$$\Delta d = d_1 - d_2 = 2k(x_i + 1) - 2y_i + 2k - 1. \quad (39)$$

Znaménko proměnné Δd určí, který ze dvou pixelů leží blíže originální úsečce. Pokud leží pixel o souřadnici y_i blíže úsečky, je hodnota Δd záporná, jinak je hodnota Δd kladná a bližší je pixel o souřadnici $y_i + 1$. Důležité pro vykreslování je právě znaménko proměnní Δd a nikoliv hodnota této proměnné.

Směrnice přímky, definovaná jako podíl dvou celých čísel $\Delta y/\Delta x$, je jedinou neceločíselnou proměnnou ve vztahu (3.2. Vynásobíme celou rovnici číslem Δx a po úpravě získáme nový vztah

$$d_i = \Delta x \Delta d = 2\Delta y x_i - 2\Delta x y_i + 2\Delta y + \Delta x(2k - 1) \quad (40)$$

v němž $2\Delta y + \Delta x\ 2k - 1$ je konstanta. Proměnnou d_i na levé straně nazýváme *rozhodovacím členem (decision)*, jelikož podle znaménka jeho+ hodnoty určujeme souřadnice následujících pixelů. Rozhodovací člen d_{i+1}pro další krok zapíšeme shodně

$$d_{i+1} = 2\Delta y x_{i+1} - 2\Delta x y_{i+1} + konst. \quad (41)$$

Odečtením obdržíme

$$\begin{aligned} d_{i+1} - d_i = {} & 2\Delta y x_{i+1} - 2\Delta x y_{i+1} + konst \\ & - 2\Delta y x_i + 2\Delta x y_i - konst \\ & = 2\Delta y(x_{i+1} - x_i) - 2\Delta x(y_{i+1} - y_i). \end{aligned} \quad (42)$$

Jelikož $x_{i+1} - x_i = 1$ (krok v řídicí ose je vždy jedna) můžeme zapsat rozhodovací člen d_{i+1} následující po d_i jako

$$d_{i+1} - d_i = 2\Delta y - 2\Delta x(y_{i+1} - y_i)\ a \quad (43)$$

$$d_{i+1} = d_i + 2\Delta y - 2\Delta x(y_{i+1} - y_i). \quad (44)$$

Navíc, pokud je d_i záporné je $y_i = y_{i+1}$ pak

$$d_{i+1} = d_i + 2\Delta y. \quad (45)$$

Pomocí vztahů počítáme následující hodnotu rozhodovacího členu iteračně z aktuální hodnoty a jejího znaménka. Způsob výpočtu shrnuje následující tabulka

$$d_i \geq 0 \qquad d_{i+1} = d_i + 2\Delta y - 2\Delta x(y_{i+1} - y_i) \quad (46)$$

$$d_i < 0 \qquad d_{i+1} = d_i + 2\Delta y \quad (47)$$

Počáteční hodnotu $d_i = 2\Delta y - \Delta x$ získáme dosazením souřadnic $[x1, y1]$ počátečního bodu úsečky do rovnice (36).

Hodnota rozhodovacího členu p je základním kritériem pro výběr pixelů tvořících obraz úsečky v rastru. Hodnotu p_i aktualizujeme pro každý pixel pouze sčítáním. Pokud je její hodnota záporná, souřadnice *y* následujícího pixelu se nemění. Při kladném znaménku p_i má následující pixel souřadnici *y* o jedničku větší.

Připomeňme, že uvedený postup (dokumentovaný algoritmem 3.2) rasterizuje pouze úsečky, jejichž směrnice je kladná a menší než jedna. Při zpracování libovolných úseček je proto třeba v iniciální fázi algoritmu nejprve rozhodnout, která osa je řídicí a podle toho zaslat úsečku do jedné ze dvou samostatných, analogicky vytvořených částí algoritmu. Vhodnou inicializací konstant docílíme vykreslování v příslušném směru (zleva doprava, zprava doleva apod.)

1. Inicializace algoritmu

   1a. Z koncových bodů [x1, y_1] a [x_2, y_2] urči konstanty:

   1a-i. $k_1 = 2\Delta y$

   1a-ii. $k_2 = 2(\Delta y - \Delta x)$

   1b. Inicializuj rozhodovací člen *d* na hodnotu $2(\Delta y - \Delta x)$

   1c. Inicializuj [*x, y*] jako [x_1, y_1]

   1d. Vykresli bod [*x, y*]

2. Dokud je $x <= x_2$, opakuj:

   2a. $x = x + 1$

   2b. Je-li d kladné,

   2b-i. pak $y = y + 1$ a $d = d + k_2$

   2b-ii. jinak $d = d + k_1$

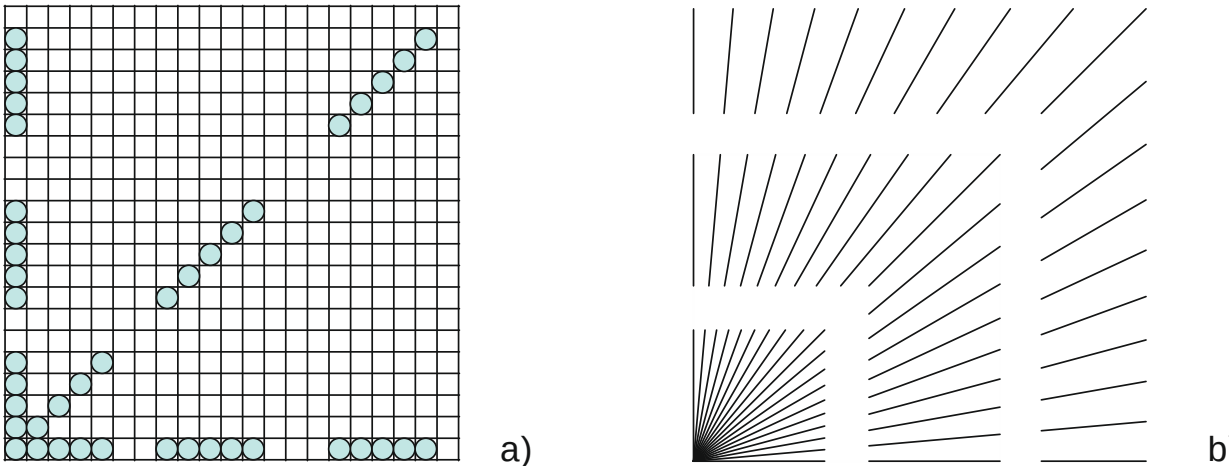
   2c. Vykresli bod [*x, y*]

Algoritmus 3.2: Bresenhamův algoritmus pro řídicí osu x

Bresenhamův algoritmus je příkladem iteračního postupu, v němž se na základě pomocné proměnné (rozhodovacího členu) rozhoduje o podmíněném provedení určité varianty. Algoritmus je velmi jednoduchý, v těle cyklu se používá pouze sčítání a test znaménka.

3.3.4 Kresba přerušované čáry

Algoritmy pro kresbu přerušovaných čar *(dashed line)* můžeme rozdělit do dvou skupin (Obr 22). Algoritmy z prvé skupiny jsou jednodušší, ale jejich výsledky nejsou vždy optimální. Tyto algoritmy postupně vypočítají souřadnice všech pixelů úsečky, například Bresenhamovým algoritmem, a pixel je vykreslen dle toho, zda patří do plného či prázdného úseku čáry.



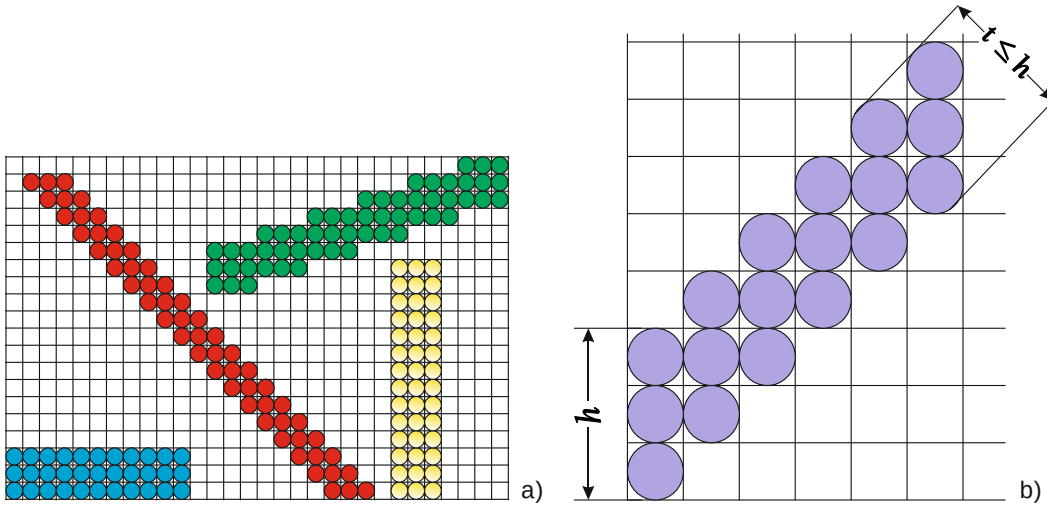
Obr 22. Přerušované úsečky kreslené jednoduchým algoritmem, a) rasterizace čáry, b) úsečky vykreslené pro úhly od 0° do 90°.

Tento způsob generování přerušovaných čar je rychlý, protože pracuje v celočíselné aritmetice, ale vzhled výsledné přerušované úsečky je ovlivňován jejím sklonem. Na obrázku 3.4 vlevo vidíme tři úsečky vycházející ze stejného bodu a vykreslené stejnou čárkovanou čárou. Diagonální úsečka má kreslené úseky zjevně delší oproti vodorovné a svislé úsečce. Tento nepříjemný jev je způsoben vlastnostmi rastru, resp. jeho *metrikou.* V praxi se proto většinou používá přesnější postup, založený na výpočtu délek. Pomocí geometrických výpočtů se určí koncové body kreslených úseků a takto získané elementární úsečky se teprve pak vykreslí běžným algoritmem.

Obdobně jako v technické praxi, je i v počítačové geometrii žádoucí, aby v obou koncových bodech úsečky *byly* zobrazeny plné úseky. Úsečka by se jevila nedokreslená zejména u lomených čar. Možné řešení je vždy vykreslit poslední úsek na úsečce plnou čárou bez ohledu na skutečnou definici vzoru. Existují i složitější algoritmy, které optimalizují jak začátek, tak i konec úsečky tak, aby na obou koncích byla vykreslena alespoň část plného úseku.

3.3.5 Kresba silné čáry

Podobně jako u přerušované čáry rozlišujeme dva typy algoritmů pro kresbu silné čáry *(thick line).* Jednoduchý postup je rychlý, ale vykazuje nepřesnosti. Přesný postup pak vyžaduje náročnější výpočty. Do první třídy metod patří upravený Bresenhamův algoritmus, při kterém se namísto jednoho pixelu v každém kroku kreslí několik pixelů nad sebou či vedle



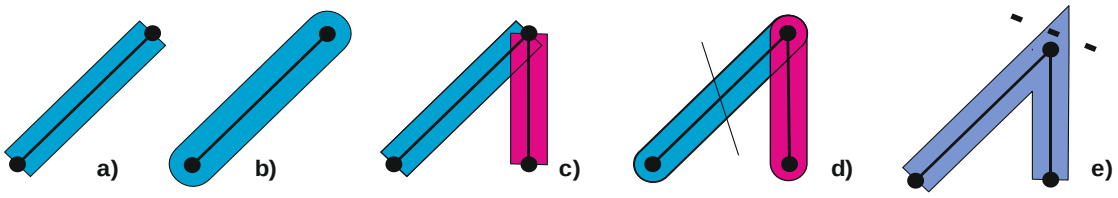
Obr 23. Silné čáry kreslené upraveným Bresenhamovým algoritmem

Na obrázku (Obr 23) je uveden výstup Bresenhamova algoritmu při kreslení silných čar. Vidíme zde dva nedostatky:

1. Síla čáry závisí na sklonu úsečky.
2. Zakončení čar je ostré, svislé nebo vodorovné.

Proč jsou úsečky různé tloušťky? Podívejte se na obrázek ad b). Šikmé čáry jsou tenčí než svislé a vodorovné. Na obrázku kóta h značí požadovanou tloušťku čáry, vidíme, že ji tvoří 3 jednotky – pixely. Skutečnou tloušťku čáry udává kóta t. Kóta h tvoří přeponu myšleného pravoúhlého trojúhelníku a síla čáry t jednu z jeho odvěsen. Přepona pravoúhlého trojúhelníku je vždy větší než kterákoli z odvěsen. Také tloušťka šikmé čáry je při použití těchto jednoduchých algoritmů vždy menší než tloušťka vodorovné či svislé čáry. Tento jev jde korigovat u silnějších čar přepočtem na skutečnou tloušťku, u tenkých čar, tloušťky jednoho pixelu korekci nelze provést. Zkuste se sami zamyslet, proč u nejtenčích čar vykreslovaných Bresenhamovým algoritmem nelze modifikaci provést. Myslíte si, že může existovat (jakkoli složitý) algoritmus, který by tuto nectnost odstranil?

Druhý neduh této metody napravit nelze. U malé tloušťky čar nepůsobí rušivě, u velké je však viditelný.
Kresba silných čar se proto převádí na kresbu ohraničené oblasti s vyplněnou plochou. Nejjednodušší plochou je obdélník, konce úsečky leží ve středu jeho krátkých stran. Pak lze esteticky vyřešit i napojení lomených čar a jejich zakončení (Obr 24). Lze použít i jiná zakončení, například oblouk. U napojení lomených čar je vhodné ošetřit případ, kdy čáry svírají ostrý úhel, na místě spoje se vytvoří dlouhý osten, který je nutno oříznout.



Obr 24. Zakončení silných (a, b) a lomených silných (c, d, e) čar

Shrnutí pojmů

Rasterizace úsečky se provádí dle hlavní osy.

Hlavní osa je ta osa, ke které se úsečky přimyká. Pro směrnici úsečky menší než jedna je to osa x, jinak osa y.

Jednoduchým algoritmem pro rasterizaci úsečky je algoritmus DAA.

Výkonný a rychlý je Bresenhamův algoritmus rasterizace úsečky.

Bresenhamův algoritmus využívá rozhodovacího členu k tomu, zda se má pokračovat se stejnou souřadnicí na vedlejší ose, nebo o jeničku vyšší.

Otázky

Co je hlavní a vedlejší osa při rasterizaci úsečky.

Popište funkci algoritmu DAA.

V čem je výhodný Bresenhamův algoritmus rasterizace úsečky.

K čemu se používá u Bresenhamova algoritmu rozhodovací člen.

3.4 Kružnice a elipsa

Čas ke studiu: 1 hodina

Cíl Po prostudování tohoto odstavce budete umět

- Matematicky popsat kružnici a elipsu.
- Pochopit význam transformací při popisu kružnice a elipsy
-

Výklad

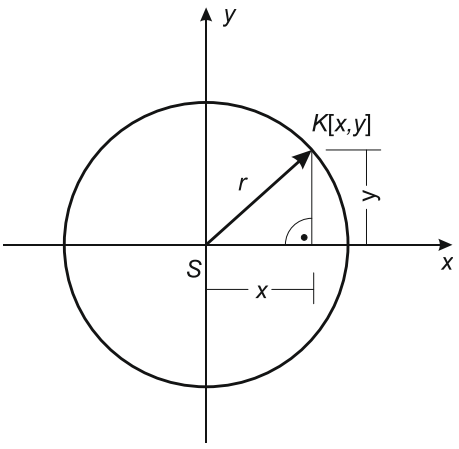
Mezi základní geometrické útvary patří kružnice a elipsa (Obr 25 a Obr 26). Ačkoli to nemusí být na první pohled patrné, kružnice je zvláštní případ elipsy. Proto také v kreslících programech obvykle nenajdeme samostatný nástroj pro kreslení kružnic, ale obvykle nástroj pro kreslení elips pomocí kterého kreslíme i kružnici.*

Kružnici lze definovat jako geometrické místo bodů, které mají stejnou vzdálenost r od jistého bodu S.

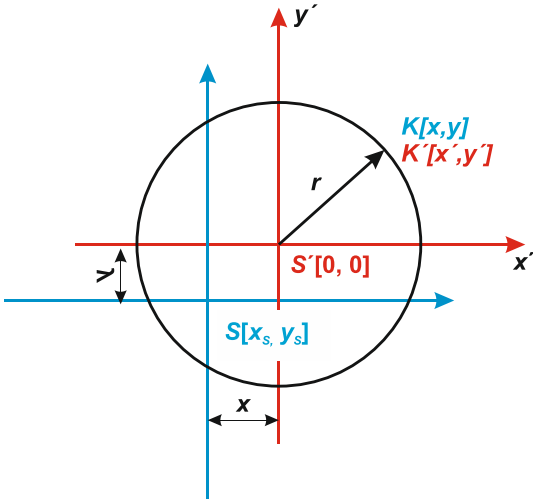
Bodu S říkáme střed kružnice a vzdálenosti r poloměr kružnice.

Základní a nejjednodušší rovnici kružnice obdržíme, pokud její střed $S = [0, 0]$ bude ležet v počátku souřadnicového systému. Rovnice kružnice bude mít tvar

$$x^2 + y^2 = r^2 \tag{48}$$



Obr 25. Středová rovnice kružnice v počátku



Obr 26. Posunutí středu kružnice mimo počátek

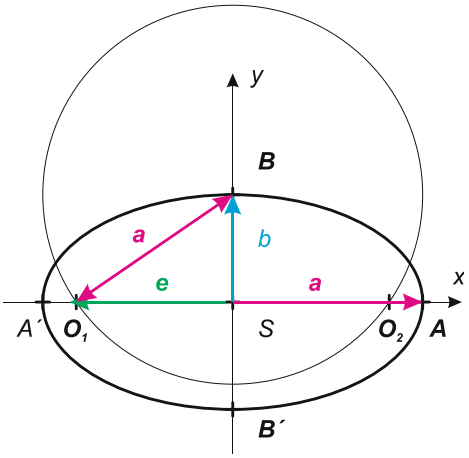
Jaký jiný základní geometrický útvar popisuje obdobná rovnice $a^2 + b^2 = c^2$?

Pokud bude mít kružnice svůj střed mimo počátek souřadnicového systému $S\,[x_s, y_s]$, můžeme využít transformaci souřadnic do nové soustavy souřadnic x' a y' tak, aby střed kružnice v těchto nových souřadnicích opět ležel v počátku této čárkované souřadnicové soustavy $S' = [0,0]$. Souřadnice libovolného bodu $A\,[x_a, y_a\,]$ bude mít v čárkované soustavě souřadnice

$$A' = [x_a', y_a'\,], \tag{49}$$
$$x'_a = x_a - x_s, \tag{50}$$
$$y'_a = x_a - x_s. \tag{51}$$

Aplikujeme-li tuto transformaci na základní rovnici kružnice, pak obdržíme pro kružnice se středem $S\,[x_s, y_s]$ rovnici

$$(x - x_s)^2 + (y - y_s)^2 = r^2. \tag{52}$$



Obr 27. Základní charakteristiky elipsy, ohniska, střed, poloosy a excentricita

Elipsu můžeme definovat jako geometrické místo bodů, které mají od daných bodů O_1, O_2 stejnou vzdálenost (Obr 27). Tyto dva body nazýváme ohniska elipsy. Střed elipsy leží ve středu úsečky spojující tato dvě ohniska. Základní, středovou, rovnici elipsy obdržíme, pokud střed elipsy leží v počátku souřadnicového systému a ohniska leží na ose x.

Pak elipsa protíná osu x v bodech $A[a, 0]$ a $A'[-a, 0]$ a osu y v bodech $B[0, b\,]$ a $B'[0, -b]$. Úsečku $SA = a$ o délce $|SA| = a$ nazýváme hlavní poloosou elipsy a úsečku $SB = b$o délce nazýváme vedlejší poloosou. Rovnici této elipsy říkáme středová rovnice a má tvar

$$\frac{x^2}{a}+\frac{y^2}{b}=1 \tag{53}$$

Pro tuto elipsu platí $a > b$. Další charakteristikou veličinou elipsy je její excentricita

$$e = a - b \tag{54}$$

Ohniska elipsy mají souřadnice

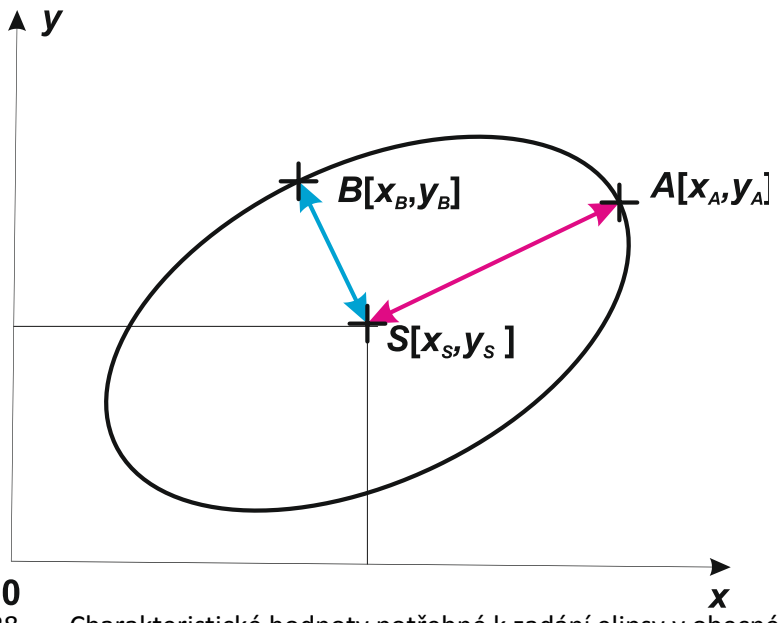
$$O_1 = [-e, 0] \tag{55}$$

a

$$O_2 = [e, 0] \tag{56}$$

Pokud elipsa nemá střed v počátku souřadnicového systému, pak můžeme použít shodný postup jako u kružnice a použít transformaci do čárkovaného souřadnicového systému (obr 28). Pokud elipsa nemá osy rovnoběžné s osami souřadnicového systému, lze využít další transformaci, a to natočení a převést její popis na základní tvar.

Shrneme-li výše uvedené poznatky, vidíme, že kružnice je dostatečně definována souřadnicemi jejího středu $[x_S, y_S]$ a poloměrem r. Elipsu jednoznačně určují souřadnice jejího středu $[x_S, y_S]$ a její poloosy $a = SA$ a $b = SB$. (Obr 28)



Obr 28. Charakteristické hodnoty potřebné k zadání elipsy v obecné poloze

Shodné vlastnosti jako kružnice a elipsa nají kruhové a eliptické oblouky jako části základního geometrického tvaru. U těchto útvarů je nutno navíc definovat bud' dvojici koncových bodů oblouku nebo dvojici úhlů měřených od středu a zvoleného vztažného bodu objektu.

Shrnutí poznatků

Základní **rovnice kružnice** má tvar $x^2 + y^2 = r^2$., kde *r je* **poloměr** kružnice.
Kružnice je určena jejím **středem** a **poloměrem**.
Elipsu charakterizují **délky poloos a její ohniska**.
Středová rovnice elipsy má tvar $\frac{x^2}{a} + \frac{y^2}{b} = 1$
Ohniska elipsy mají souřadnice $O_1 = [-e, 0]$ a $O_2 = [e, 0]$. e udává **excentricitu**. Vztah mezi excentricitou a délkami poloos udává vztah $e^2 = a^2 - b^2$.

Otázky

Napište středovou rovnici kružnice.

Čím je charakterizována kružnice a čím elipsa?

Jaký vzájemný vztah mají elipsa a kružnice?

3.5 Rasterizace kružnice

Čas ke studiu: 1 hodina

Cíl Po prostudování tohoto odstavce budete umět

- Umět vysvětlit symetrii bodů na kružnici a jejich význam pro rasterizaci
- aplikovat vykreslení kružnice Bresenhamovým algoritmem
- popsat modifikaci Bresenhamova algoritmu pro vykreslení elipsy

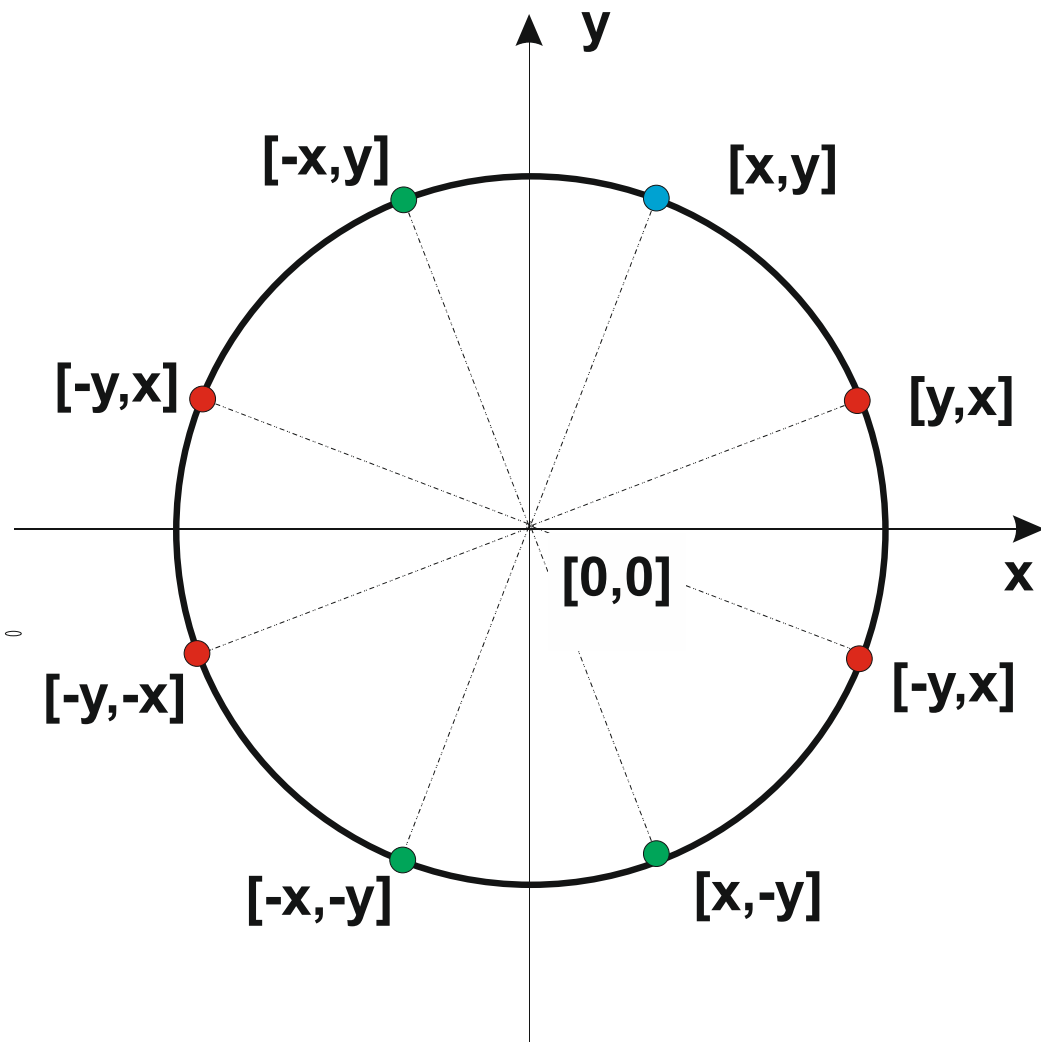
Výklad

Kružnice v rastru lze vykreslit výše popsanými algoritmy pro kresbu úsečky náhradou přesného tvaru kružnice či elipsy mnohoúhelníkem jako lomenou čáru. Jeho vrcholy lze vypočítat iteračním způsobem s využitím transformace otáčení.

Pro jednoduchost většina algoritmů pracuje s kružnicí a elipsou se středem v počátku a v základní pozici. Získané body v základním rastru jsou posunuty do cílové polohy až při vykreslování.

3.5.1 Bresenhamův algoritmus pro kresbu kružnice

Rasterizaci kružnice lze řešit obdobně jako u úsečky jen pomocí celočíselné aritmetiky. Pokud si uvědomíme, že kružnice je středově symetrická, pak můžeme z každého vypočteného bodu odvodit dalších sedm bodů (obr 29). Toto docílíme dvojím způsobem. Zaprvé záměnou x-ové a y-ové souřadnice vypočteného bodu a zadruhé změnou znaménka každé souřadnice. Pro rasterizaci celé kružnice tedy stačí vypočítat hodnoty souřadnic bodů ležících v jednom oktantu, např. v úseku od $x = 0$ do $x = y$. Na obrázku je originální bod modrý, zelené body jsou symetrické, jejich souřadnice jsou získány záměnou jednoho nebo obou znamének souřadnic, souřadnice červených bodů jsme získali kombinací záměny x-ové a y-ové souřadnice a záměnou znamének souřadnic



Obr. 29. Souměrnost bodů na kružnici

J. E. Bresenham publikoval algoritmus, jehož jádro je analogické algoritmu rasterizace úsečky. V literatuře se s tímto algoritmem můžeme setkat pod názvem *midpoint algorithm.* Díky symetrii bodů na kružnici, stačí řešit vykreslení bodů kružnice jen v jedné její osmině – oktantu. Souřadnice zbylých sedmi bodů získáme na jednoduše na základě symetrie, jak jsme ukázali na obrázku… Obdobně jako při vykreslování úsečky, je nutno i při vykreslování křivek, mezi něž patří i kružnice, je nutno dle směrnice tečny v příslušném bodě vybrat příslušnou řídicí osu. U kružnice, pokud vypočítáváme body jen jednoho kvadrantu (zbylých sedm v ostatních kvadrantech jsou symetrické) se řídící osa nemění. Pokud algoritmus začíná v bodě [0, *r*] a končí v průsečíku kružnice s hlavní diagonálou, kdy $x = y$, pak se souřadnice x sousedních bodů rasterizované kružnice liší právě o jeden pixel. Krok v ose *x* je tedy konstantní, vedlejší osou je osa *y*.

I u tohoto algoritmu je využit rozhodovací člen, jehož znaménko určuje, zda se souřadnice na vedlejší ose změní o jedničku či zůstane nezměněna. Nebudeme procházet jednotlivé kroky vedoucí od základní rovnice kružnice k odvození konečného tvaru rozhodovacího členu. Uvedeme jen konečný tvar rozhodovacího členu a načrtneme možný tvar algoritmu.

V i-tém kroku algoritmu bude mít rozhodovací člen hodnotu

$$d_i = (x_i + 1)^2 + \left(y_i - \frac{1}{2}\right)^2 - r^2$$

V následujícím kroku pak analogicky hodnotu

$$d_{i+1} = (x_{i+1} + 1)^2 + \left(y_{i+1} - \frac{1}{2}\right)^2 - r^2$$

Kde pro i-tý a (i+1)-vý krok jsou: x_i a x_{i+1} souřadnice x vykreslovaného bodu, y a y_{i+1} souřadnice y, r je poloměr kružnice. Uvědomme si, že v souřadnice x náleží řídící ose, její velikost se v následujícím kroku zvýší o jedničku, zatímco souřadnice y se v závislosti na znaménku rozhodovacího členu nemění nebo se, pokud je hodnota rozhodovacího členu kladná, shodně se souřadnicí x zvětší také o jedničku. Shodným postupem jako u algoritmu pro vykreslování úsečky, budeme počítat hodnotu rozhodovacího členu iteračně, využijeme vždy již spočtenou hodnotu z aktuálního kroku a budeme ji aktualizovat přičtením jisté hodnoty pro následující krok.

Abychom zjistili, jakou hodnot máme přičíst, odečteme od sebe hodnotu rozhodovacího členu příštího kroku a hodnotu rozhodovacího členu aktuálního kroku.

$$\begin{aligned} d_{i+1} - d_i &= (x_{i+1} + 1)^2 + \left(y_{i+1} - \frac{1}{2}\right)^2 - r^2 - (x_i + 1)^2 - \left(y_i - \frac{1}{2}\right)^2 + r^2 \\ &= x_i^2 + 4x_i + 4 - x_i^2 - 2x_i - 1 + \left(y_{i+1} - \frac{1}{2}\right)^2 - \left(y_i - \frac{1}{2}\right)^2 \\ &= 2x_i + 3 + y_{i+1}^2 - y_{i+1} + \frac{1}{4} - y_i^2 + y_i - \frac{1}{4} \end{aligned}$$

Pro hodnota souřadnice x ve dvou následných krocích, jak jsme se již zmínili, platí $x_{i+1} = x_i + 1$.

Pokud se souřadnice ve dvou následných krocích nemění, pak dosazením do xx obdržíme

$$d_{i+1} - d_i = 2x_i + 3$$

Zmenší-li se souřadnice y o jedničku, pak rozdíl rozhodovacích členů ve dvou následných krocích bude

$$\begin{aligned} d_{i+1} - d_i &= 2x_i + 3 + (y_i - 1)^2 - y_i + 1 - y_i^2 + y_i \\ &= 2x_i + 3 + y_i^2 - 2y_i + 1 - y_i + 1 - y_i^2 + y_i = 2x_i + 5 - 2y_i \end{aligned} \qquad (57)$$

Zamyslete se, proč jsme souřadnici y zmenšili o jedničku. Zdůrazněme, že se jedná o středovou rovnici kružnice a že začínáme z bodu [0, *r*] a končíme v průsečíku kružnice s hlavní diagonálou, tedy když se souřadnice *x a y rovnají.*

Získané poznatky můžeme shrnout do rozhodovací tabulky

$$d_i \leq 0 \qquad d_{i+1} = d_i + 2x_i + 3 \tag{58}$$

$$d_i > 0 \qquad d_{i+1} = d_i + 2x_i + 5 - 2y_i \tag{59}$$

Vzorce pro aktualizaci rozhodovacího členu obsahují násobení. Podíváme-li se pozorně, pak jak souřadnice x tak i souřadnice y zde vystupují vynásobeny číslem dvě. Toho využijeme a místo s prostou hodnotou souřadnic budeme počítat s jejími dvojnásobky. Navíc si uvědomme, že se pohybujeme v diskrétním světě, kde souřadnice, x a y, pro sousední pixely mohou zůstat stejné nebo se změnit o jedničku. Pokud počítáme s jejich dvojnásobky, pak jejich změna bude dvojnásobná. Pro členy $2x_i$ a $2y_i$ zavedeme pomocné proměnné *x2 a y2* jejichž hodnota bude dvojnásobek příslušné souřadnice. Jinými slovy, změní-li se hodnota *x*, resp. *y* o jedničku, aktualizujeme danou pomocnou proměnnou přičtením, resp. odečtením dvojky. Inicializaci provedeme tak, abychom se zbavili konstant tři a pět v těchto vzorcích.

Pro vykreslování kruhového oblouku se nastaví omezující podmínky ve funkci vykreslující osm symetrických bodů. Pokud není splněna podmínka pro vykreslení bodu, kresba se potlačí. Hodnotu rozhodovacího členu budeme stejně jako v případě úsečky určovat během iteračního výpočtu.

1. Inicializace proměnných
   - 1a. Inicializuj pomocné proměnné:
     - 1a-i. $x2 = 3,$
     - 1a-ii. $y2 = 2r - 2$
   - 1b. Inicializuj rozhodovací člen p na hodnotu $1 - r$
   - 1c. Inicializuj bod $[x, y]$ jako souřadnicemi $[0, r]$
2. Dokud je $x \leq y$, opakuj:
   - 2a. vykresli osm bodů symetrických s bodem $[x, y]$
   - 2b. Je-li hodnota d kladná, pak
     - 2b-i. $d = d - y2$
     - 2b-ii. $y2 = y2 - 2$
     - 2b-iii. $y = y - 1$
   - 2c. $d = d + x2$
   - 2d. $x2 = x2 + 2$
   - 2e. $x = x + 1$

Algoritmus 3.3: Bresenhamův algoritmus pro kresbu kružnice

3.5.2 Rasterizace elipsy

Pro vykreslení elipsy lze opět s výhodou použít Bresenhamův algoritmus. Zde můžeme symetrii využít pouze pro tři další body a algoritmus musí vygenerovat body elipsy v jednom celém kvadrantu. Situace je komplikovanější o to, že v průběhu jednoho kvadrantu se mění i řídicí a vedlejší osa. Tento bod ne pro prvý kvadrant dán takovým bodem elipsy, kdy tečna k elipse vedená tímto bodem má

směrnici -1. Uvědomíme-li si, že hodnota směrnice tečny je rovna derivaci rovnice elipsy $y = f(x)$ dle proměnné x, pak pro souřadnice tohoto bodu lze odvodit vztah

$$Z = \left[\frac{a^2}{\sqrt{a^2+b^2}}, \frac{b^2}{\sqrt{a^2+b^2}}\right]$$

Zde a a b jsou délky poloos. V prvém kvadrantu při pohybu zleva doprava je nejprve řídicí osa x, od bodu Z dále pak se role os zamění.

V části s řídicí osou *x* se pro výpočet rozhodovacího členu použijí tato pravidla:

$$d_i \leq 0 \qquad d_{i+1} = d_i + b^2(2x_i + 1) \tag{60}$$

$$d_i > 0 \qquad d_{i+1} = d_i + b^2(2x_i + 1) - 2a^2 y_i \tag{61}$$

Analogické vztahy platí i pro druhou část algoritmu. Vhodnou volbou počáteční inicializace a vhodnou volbou proměnných a jejich významu lze opět celý výpočet řešit celočíselně.

Shrnutí poznatků

J. E. Bresenham publikoval algoritmus jehož jádro je analogické algoritmu rasterizace úsečky. V literatuře se s tímto algoritmem můžeme setkat pod názvem ***midpoint algorithm.***

I u tohoto algoritmu je využit rozhodovací člen, jehož znaménko určuje, zda se souřadnice na vedlejší ose změní o jedničku či zůstane nezměněna.

Kružnici díky symetrii bodů vůči středu a osám **vykreslujeme po oktantech**, čili osminách kružnice.

Rozhodovací člen pro vykreslení kružnice má tvar:

$$d_i \leq 0 \qquad d_{i+1} = d_i + 2x_i + 3 \tag{62}$$

$$d_i > 0 \qquad d_{i+1} = d_i + 2x_i + 5 - 2y_i \tag{63}$$

Pro vykreslení elipsy lze opět s výhodou použít Bresenhamův algoritmus

Rozhodovací člen pro vykreslení elipsy má tvar

$$d_i \leq 0 \qquad d_{i+1} = d_i + b^2(2x_i + 1) \tag{64}$$

$$d_i > 0 \qquad d_{i+1} = d_i + b^2(2x_i + 1) - 2a^2 y_i \tag{65}$$

Elipsu musíme vykreslovat po kvadrantech a navíc musíme během vykreslování **zaměnit řídicí osu s vedlejší**.

Otázky

Proč je možno kružnici při rasterizaci kružnice vypočítávat souřadnice bodů jen v její jedné osmině?

Proč se elipsa musí vykreslit vždy po celém kvadrantu.

Na jaký zádrhel narazíme při sestavování efektivního algoritmu pro vykreslení elipsy.

3.6 Oblast

Čas ke studiu: 1 hodina

Cíl Po prostudování tohoto odstavce budete umět

- definovat oblast jako grafický pojem a její atributy
- definovat hranici oblasti a způsob jejího určení
- popsat druhy výplně oblasti

Výklad

Dosud jsme se setkali s grafickými primitivy liniového (čárového) charakteru. Kresba plošných obrazců využívá obecného a tudíž i univerzálního prvku, který se nazývá oblast (area). Oblast má dvě základní vlastnosti:

- je vždy uzavřená a
- její vnitřek lze různými způsoby vyplnit.

Oblast může být konvexní či konkávní. Práce s konvexním tvarem oblasti je jednodušší, jak při šrafování, vyplňování či ořezávání. Konvexní oblast můžeme definovat jednoduše tak, že libovolná přímka protínající tuto oblast má s její hranicí společné pouze dva body.

Základním definičním prvkem oblasti je její hranice. Tu můžeme popsat dvěma způsoby.

1. Geometricky určená hranice

Definuje polygon (mnohoúhelník) jako posloupnost bodů definujících po řadě jeho vrcholy. Poslední vrchol se vždy spojuje s prvním vrcholem, tím je zajištěno, že je vzniklá hranice uzavřená. Hranici lze definovat pomocí jedné nebo více navazujících křivek. Můžeme ji též definovat jako průnik poloprostorů.

2. Hranice nakreslená v rastru

Tvar takovéto hranice může být libovolný. Pro její určení je třeba znát:

- barvu hranice, nebo
- barvu vnitřních bodů oblasti, případně
- barvu bodů vně oblasti.

Před vyplňováním oblasti uživatel zadává souřadnice vnitřního bodu oblasti.

Vyplňování oblasti můžeme rozdělit na dvě činnosti:

- nalezení vnitřních bodů a
- obarvení vnitřních bodů.

Vnitřek oblasti lze vyplnit:

a) Vykreslení samotné hranice oblasti ***bez vyplnění*** vnitřku se anglicky označuje různě: *empty, void, hollow;*

b) ***jednou barvou*** *(solid fill),*

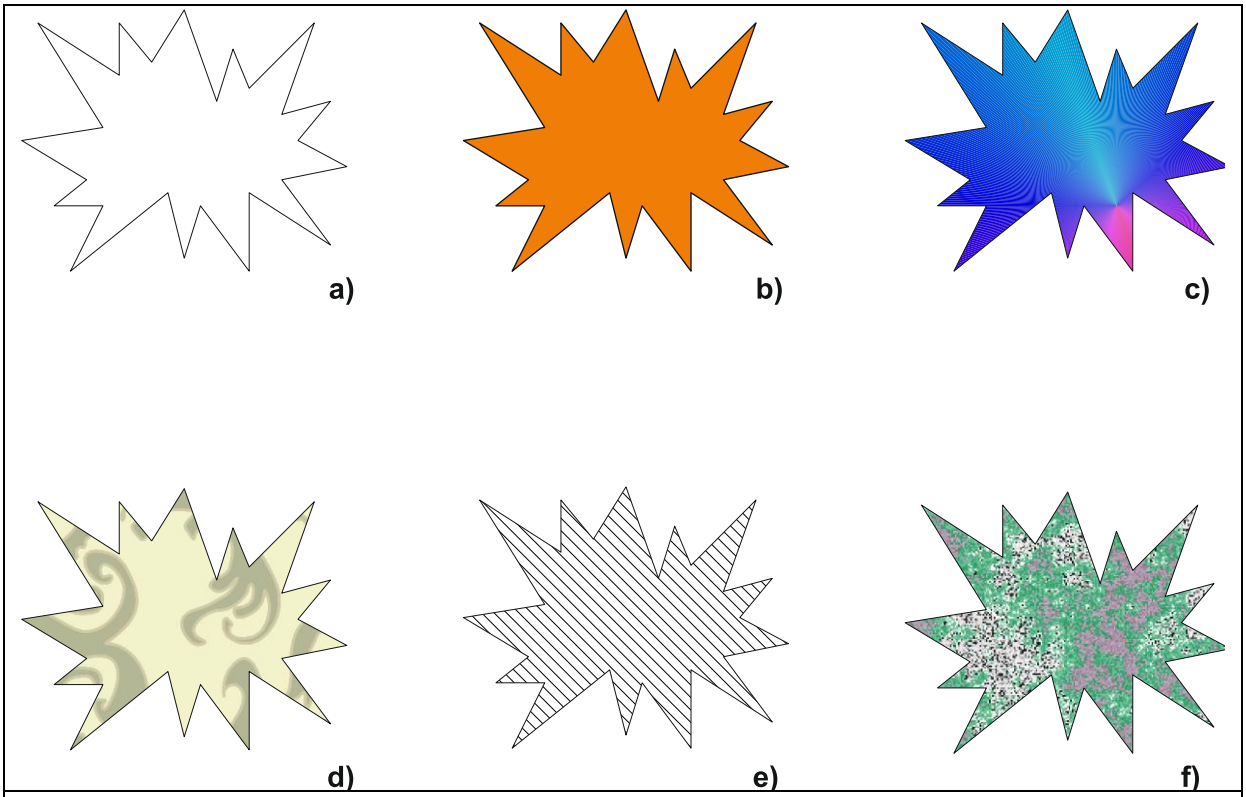
c) případně může být ***barva výplně stanovena výpočtem*** *(fountain fill),*

d) opakovaně nanášeným ***vzorkem*** složeným z více barevných bodů *(pattern tilling),*

e) ***šrafováním*** *(hatch fill),*

f) vyplnění ***texturou*** (*texture fill).*

Jednotlivé možnosti vidíte na obrázku 30.



Obr. 30 Výplně tvaru a) bez výplně, b) jednobarevná výplň, c) vypočítaná výplň, d) vzorek, e) šrafování, f) textura

Shrnutí poznatků

Oblast má **dvě základní** vlastnosti: je vždy **uzavřená** a její vnitřek lze různými způsoby **vyplnit**.

Geometricky určená hranice definuje polygon (mnohoúhelník) jako posloupnost bodů definujících po řadě jeho vrcholy.
Hranice nakreslená v rastru může mít libovolný tvar. Pro její určení je třeba znát: barvu hranice, nebo barvu vnitřních bodů oblasti, případně barvu bodů vně oblasti.
Vnitřek oblasti lze vyplnit: ***bez vyplnění, jednou barvou, barvou výplně stanovenou výpočtem, vzorkem, šrafováním, případně texturou.***

Otázky

Definujte oblast a její vlastnosti.

Čím se liší geometricky definovaná hranice a hranice nakreslená v rastru-

3.7 Vyplňování geometricky určené hranice

Čas ke studiu: 4 hodiny

Cíl Po prostudování tohoto odstavce budete umět

vyjmenovat způsoby vyplňování oblasti
porozumíte a bude umět aplikovat řádkové vyplňování
využít zvláštnosti při vyplňování trojúhelníka

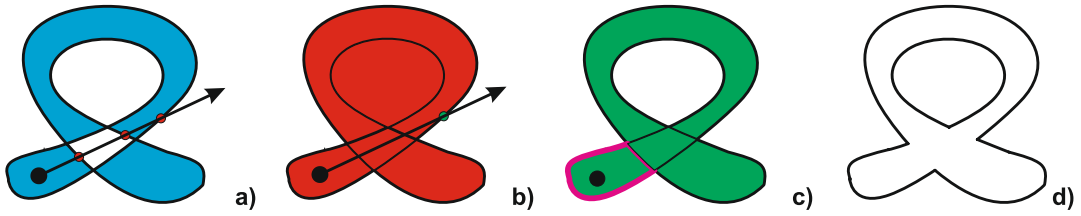
Výklad

Pokud se geometricky zadaná hranice protíná sebe samu, je nutno rozhodnout, které body budeme považovat za vnitřní. Na obrázku 3.11 jsou znázorněny různé možnosti.

a) *Paritní vyplňování*. Hranice je chápána jako entita, která odděluje vyplněný vnitřní a nevyplněný okolní prostor. Za vnitřní bod budeme považovat ten, z něhož vedená polopřímka protne lichý počet hran oblasti.

b) Jsou vyplněny všechny body, které neleží vně hranice. Všechny polopřímky vedené z vnitřního bodu protnou hranici oblasti.

c) Obtočení bodu hranicí, která je určena uzavřeným polygonálním tahem s možností sebeprotínání (obr. 3.llc).



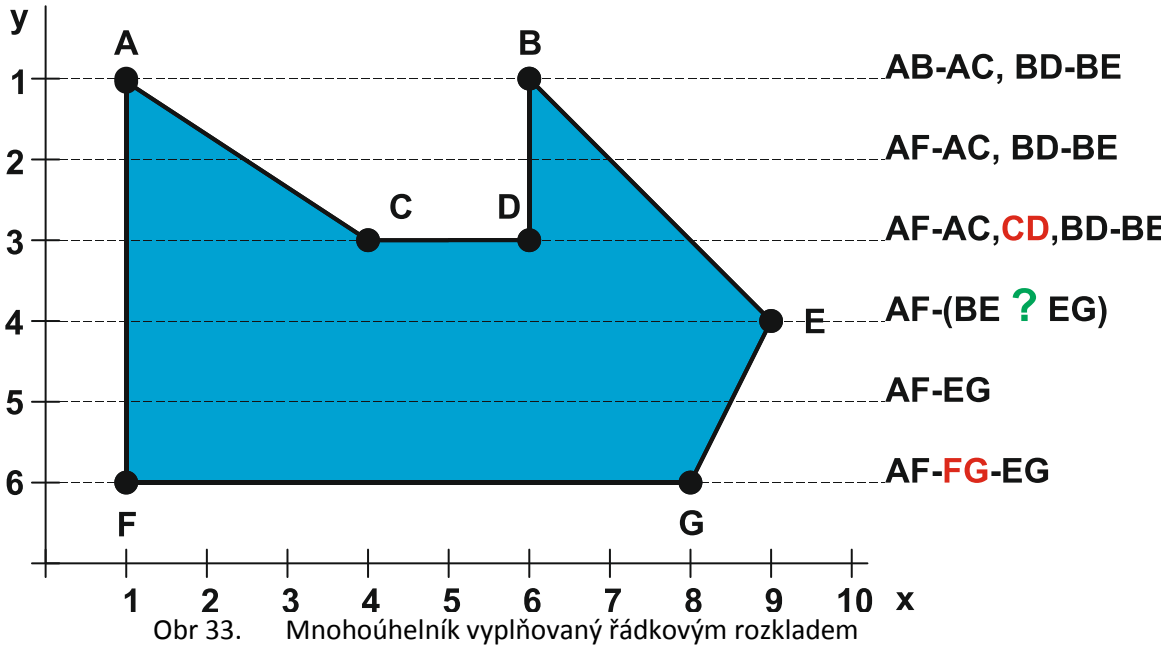
Obr. 31 Varianty vyplňování složitější geometricky určené hranice a) paritní vyplňování b) vnitřní vyplňování c) obtočení bodu d) více hranic

Nejjednodušší a nejméně výpočetně náročné je paritní vyplňování (0). Zbylé algoritmy vyžadují složitější výpočty. V dalším textu budeme věnovat pozornost algoritmům paritního vyplňování. Lze je použít i na oblast určenou několika hranicemi (obr.31).

3.7.1 Řádkové vyplňování

Základní metodou je řádkové vyplňování. Můžeme se setkat s i s názvy *paritní řádkové vyplňování* nebo *vyplňování rozkladovými řádky (scan-line conversion).* Metoda je založena na paritním vyplňování. Představme si například cestu. Přejděme z jednoho chodníku na ten protější přes cestu. Jak poznáme, že jsem na druhé straně? Přejdeme jeden obrubník a jsem na cestě. Přejdeme další obrubník a jsme mimo cestu, řekněme na protějším chodníku. První obrubník – cesta začíná, druhý obrubník – cesta končí. Každou lichou hranicí oblast začíná, každou sudou končí. Proto také název paritní vyplňování. Každým řádkem rastru vedeme myšlenou čáru a hledáme průsečíky této přímky s hranicemi oblasti. Nalezené průsečíky příslušné řádku rastru seřadíme dle souřadnic x. Každá dvojice těchto průsečíků definují úsečky ležící uvnitř oblasti.

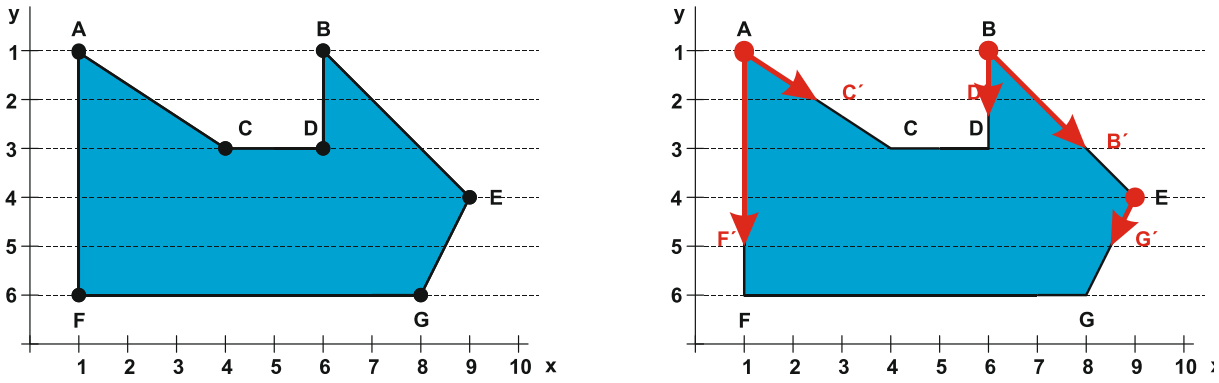
V počítačové grafice je zvykem, že horní levý roh monitoru má souřadnice $[0, 0]$. Také řádky se zpravidla číslují shora dolů. V dalším textu dodržíme tuto konvenci a použijeme orientaci osy *y*, která bude odpovídat číslování řádků. Kladná poloosa *y* bude mířit dolů.



Obr 33. Mnohoúhelník vyplňovaný řádkovým rozkladem

Na obrázku (Obr 33) vidíme jednoduchou mnohoúhelníkovou oblast se sedmi vrcholy ABCDEFG. Vrcholy jsme označili ne v pořadí hran, ale v pořadí řádků. Každý řádek představuje právě jednu řadu pixelů, třeba na monitoru počítače. Toto je si dobré zapamatovat, abychom si uvědomili, že vrcholy a hrany oblasti nemohou ležet mimo tyto řádky. Rozeberme si jednotlivé případy průsečíku rozkladových řádků a hranice oblasti. Vnitřní úsečky budeme hledat odshora dolů. Zamyslete se, zda vykreslením všech vnitřních úseček vyplníme celou oblast. Nebo ji vyšrafujeme? V pravé části obrázku jsou vypsány průsečíky toho kterého řádku s hraniční úsečkou oblasti. První rozkladová řádek se protíná celkem se čtyřmi hraničními úsečkami. Dojde k vykreslení dvou úseček nulové délky, které se ztotožňují s vrcholy A a B. Tyto úsečky mohou být vykresleny jako jeden pixel.

Následující řádek je zpracován obdobně. Ve třetím řádku musí být z výpočtu vyloučena hrana C*D*. Tato hrana má s rozkladovým řádkem geometricky vzato nekonečně mnoho průsečíků, neboť hrana leží na příslušném rozkladovém řádku. V počítačové grafice počet průsečíků bude konečný a bude se rovnat počtu pixelů úsečky. Zbylá čtveřice průsečíků definuje dvě vnitřní úsečky.



Obr 34. Mnohoúhelníková oblast a úpravy jejích hraničních úseček před vyplňováním

Na čtvrtém řádku je nalezen lichý počet průsečíků. Vrchol E je bod společný dvěma hraničním úsečkám. Musíme jednoznačně rozhodnout, i pro další vyhodnocení, kterou úsečku označíme za hraniční. Jednoduše lze tento problém vyřešit tak, že zkrátíme všechny úsečky systematicky o jeden pixel ve směru osy y. To, že zdánlivě rozpojíme hranici, nemá na činnost algoritmu žádný vliv. Jde opravdu o zdánlivé rozpojení hranice, Sjednocení všech pixelů takto zkrácených hraničních úseček nám dá původní množinu pixelů hranice oblasti. Jak jsme totiž konstruovali hraniční úsečky, ty pixely, které tvořily vrcholy mnohoúhelníku, jsme ve skutečnosti započítali dvakrát. Konečné řešení ukazuje obrázek. 34 Navíc ***systematické zkrácení*** všech hran o jeden pixel vede ke snížení počtu průsečíků. Z obrázku (Obr 34) je patrno, že po této úpravě například na třetím řádku zbudou pouze dva průsečíky místo nadbytečných čtyř.

Účelné nalezení všech možných vzájemných poloh rozkladových řádků s hranicí oblasti vyžaduje úpravu všech hraničních úseček:

a) vynecháme vodorovné hraniční úsečky,

b) ostatní pak orientujeme shora dolů,

c) zkrátíme každou úsečku u dolního konce o jeden pixel.

Uvedená metoda je efektivní pro vyplňování oblasti ohraničené mnohoúhelníkem. Lze ji použít i pro oblast ohraničenou obecnými křivkami. Při požadavku na velkou rychlost zpracování může být mnohdy jednodušší nahradit křivky lomenou čárou a použít původní, neupravený algoritmus.

Algoritmus 3.4: Algoritmus vyplňování řádkovým rozkladem

1. Pro každou hraniční úsečku:

   1a. je-li vodorovná, pak ji vynechej (resp. vykresli),

   1b. orientuj ji shora dolů, zkrať zdola o 1 pixel ,

   1c. najdi nejvýše a nejníže umístěné hranice a nastav jejich souřadnice $y_{\max}$ a $y_{\min}$.

2. Pro všechna y od y_{min}. do y_{max} :

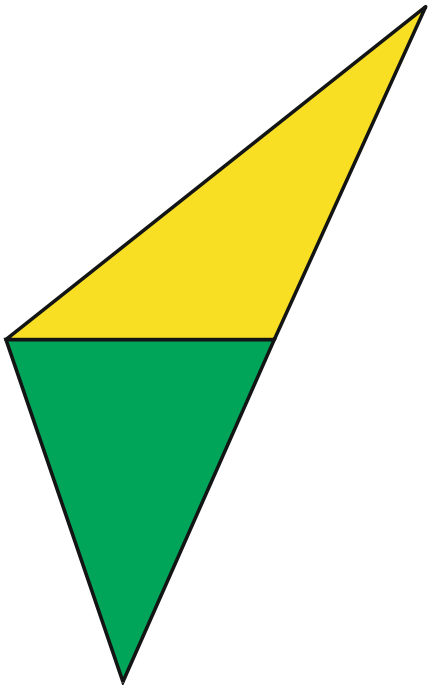
   2a. nalezni průsečíky hraničních úseček s řádkem y,

2b. seřaď nalezené průsečíky dle souřadnice *x*,

2c. vykresli úseky mezi lichými a sudými průsečíky,

3. Vykresli hranici oblasti (je-li třeba).

3.7.2 Vyplňování trojúhelníka



Obr. 33 Rozdělení trojúhelníku na dva elementární trojúhelníky

V počítačové grafice, zejména třírozměrné, je častou úlohou vyplňování trojúhelníků. Je samozřejmě možné použít výše uvedený obecný algoritmus řádkového rozkladu, ale pokud si uvědomíme, že takových elementárních trojúhelníků velké množství, je pro trojúhelník vhodné použít efektivnější metodu. Každý trojúhelník nerozdělit na dvě části – horní a dolní část (). Řez vedeme vrcholem, jehož svislá souřadnice leží mezi nejvyššíma nejnižším vrcholem. Pro tento algoritmus je důležité, aby tento řez byl vodorovný. Dovedete sami zdůvodnit, proč by jinak orientovaná hranice nepřinesla zjednodušení?

Oba vzniklé elementární trojúhelníky jsou popsány právě dvěma hranami. Zbylá, společná strana, je vodorovná a vynecháme ji. Průsečíky již neřadíme. Postupujeme současně po obou hranách shora dolů a vykreslujeme vodorovné spojnice.

Shrnutí poznatků

Pro vyplnění oblasti lze použít: ***paritní vyplňování, vyplnění bodů ležících uvnitř hranice, a obtočit bod hranicí,***

Paritní vyplňování. Hranici je chápána jako entita, která odděluje vyplněný vnitřní a nevyplněný okolní prostor. Za vnitřní bod budeme považovat ten, z něhož vedená polopřímka protne lichý počet hran oblasti.

Vyplnění bodů uvnitř hranice. Jsou vyplněny všechny body, které neleží vně hranice. Všechny polopřímky vedené z vnitřního bodu protnou hranici oblasti.

Obtočení bodu hranicí, která je určena uzavřeným polygonálním tahem s možností sebeprotínání (obr. 3.llc).

Nejpoužívanější je **řádkové vyplňování,** Při jeho aplikaci dojde k **rozpojení hranice**, aby se **redukoval počet vzájemných bodů** mezi řádkem a hranicí oblasti.

Vyplňování trojúhelníku patři mezi velmi efektivní metody vyplňování. Je využíváno v prostorové grafice.

Otázky

Jaké hlavní způsoby vyplňování umíte vyjmenovat.

Proč je vyplňování trojúhelníku efektivní. Kdy je vhodné je použít.

Jakým způsobem se dělí trojúhelník, aby šel efektivně vyplnit?

3.8 Další metody vyplňování polygonů

Čas ke studiu: 1 hodina

Cíl Po prostudování tohoto odstavce budete umět

- popsat princip inverzního vyplňování
- umět modifikovat řádkové vyplňování na vyplňování vzorem
- popsat princip šrafování a těžkosti, se kterými se můžete setkat

3.9 Další metody vyplňování

Výklad

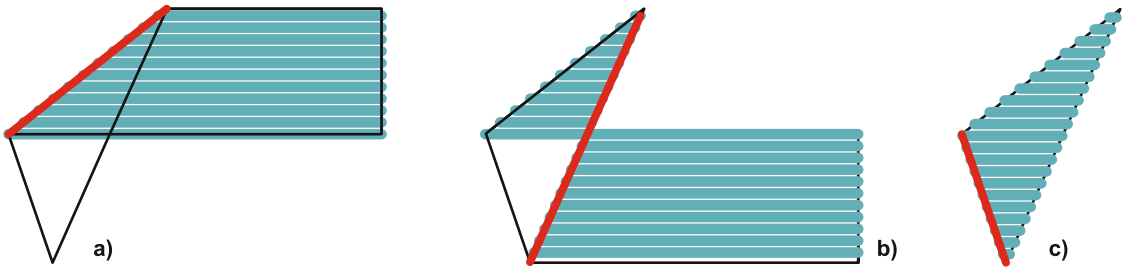
Algoritmus řádkového vyplňování, jak jsme jej poznali, lze geometricky dále zefektivnit.
Pro výpočet průsečíků hrany s následnými řádky lze využít algoritmus DAA. Nejedná se totiž o nic jiného, než o rasterizaci úsečky.

Pokud si dále uvědomíme, že počet protnutých hran se s následujícím řádkem nemění a nemění se pořadí hran, pak si jen zapamatujeme již stanovené pořadí. Případnou změnu pořadí či nové hrany vyřešíme jednoduchým třídicím algoritmem. Naznačeného postupu využívá např. řádkové vyplňování se seznamem aktivních hran.

3.9.1 Inverzní vyplňování a šablona

Slabinou řádkového rozkladu je nutnost řazení průsečíků na každém rozkladovém řádku. Byly proto vyvinuty metody nevyžadující řazení. Jednou z nich je inverzní vyplňování (0). Vykazuje lineární časovou závislost na počtu hraničních úseček. Lineární časová závislost znamená, že čas vykonávání takového algoritmu je přímo úměrný počtu hran.

Šablona *(stencil)* je část paměti reprezentující vyplňovanou oblast. V šabloně bývá pixel obrazové paměti nahrazen jediným bitem. Tento bit obsahuje informaci „vyplněno/nevyplněno". Šablona takto definuje souřadnice pixelů tvořící budoucí výplň oblasti.



Obr. 34 Postup inverzního vyplňování při zpracování hranic, aktivní hrana je vyznačena červeně

Postupujeme následovně:

1. Pro každou hraniční úsečku:
   1a. Zkrátíme ji o jeden pixel zdola.
   1b. Rasterizujeme ji algoritmem DDA s řídicí osou y.
   1c. Od každého vypočítaného pixelu na hraně vedeme vodorovnou polopřímku k pravému okraji šablony.
   1d. Pixely zaznamenáme do šablony.
   1e. Zápis se provedeme invertováním (negací) hodnot v šabloně.

Zpracujeme-li první úsečku, výplň tvoří lichoběžník zleva vymezený danou úsečkou a zprava okrajem šablony. Pokračujeme-li další hranou, část tohoto lichoběžníku bude smazána (zápis se provádí invertováním obsahu šablony), nebo doplněna o nové oblasti. Postup je dokumentován obrázkem 34.

Šablona je běžnou součástí grafických akcelerátorů. Její rozměr je totožný s velikostí obrazovky a slouží pak jako univerzální pomocná paměť *(stencil buffer)*.

3.9.2 Vyplňování vzorem

Metodu řádkového vyplňování lze zobecnit tak, že vnitřní úsečky nejsou vybarveny přímo požadovanou barvou, ale je na ně opakovaně aplikován barevný vzor definovaný v rastru. Vzor je definován v obdélníkové matici $\mathbf{V}_{m,n}$. Jakou hodnotu z matice vybereme, určíme na základě souřadnice vykresleného pixelu. Tato hodnota určí barvu vykreslovaného pixelu. Souřadnice vzoru určíme celočíselným zbytkem po dělení *(modulo)* souřadnice x a y cílového pixelu příslušným rozměrem matice vzoru. Zefektivnit algoritmus lze tím, že do obrazové paměti se opakovaně kopírují celé řádky matice **V.** Zdůvodněte funkčnost tohoto vylepšení. Stačí si uvědomit, že se vyplňování uskutečňuje po řádcích.

3.9.3 Šrafování

V technických aplikacích se často setkáme se šrafováním. Algoritmus řádkového vyplňování upravíme tak, aby se zvýšil krok změny souřadnice *y* z hodnoty 1 na *m (m* je celočíselné a kladné). Takto získáme oblast vyplněnou vodorovnými šrafami s roztečí *m* pixelů.

Při šrafování přerušovanými čárami mohou šrafy při pohledu z dálky vytvořit vzor kopírující levou hranici oblasti. Tento neduh lze eliminovat tak, že šrafy budou vykreslovány vztaženy k pevnému vztažnému bodu. Takovým bodem může být počátek souřadné soustavy.

Šrafování lze nahradit vyplňováním vzorem. U animací je pak nutné, aby se vzor pohyboval spolu s vyplňovanou oblastí.

Šrafování pod obecným úhlem lze vyřešit transformací souřadnic otočením.

3.9.4 Kreslení hranice

Hranici můžeme chápat bud'

- jako součást vyplňované oblasti nebo
- jako samostatnou entitu, která odděluje oblast od zbylého prostoru.

Výše uvedené algoritmy paritního vyplňování označí za vnitřní body i některé body na hranici oblasti. Pokud byla nejprve nakreslena hranice oblasti, vyplnění oblasti jinou barvou hraniční pixely nepřekryje. Hraniční úsečky byly při samostatném vykreslení rasterizovány s řídicí osou x. Hledání vnitřních bodů oblasti probíhá s rasterizací s řídicí osou y. Tím je zaručeno, že získáme odlišné množiny pixelů.

Shrnutí poznatků

Mezi další algoritmy vyplňování oblasti patří inverzní vyplňování a vyplňování pomocí šablony.

Vyplňování pomocí šablony je vhodné pro vyplnění oblasti vzorem.

Otázky

V čem spočívá kouzlo inverzního vyplňování.

Jak lze s výhodou použít vyplňování podle šablony.

3.10 Vyplňování hranice nakreslené v rastru

Čas ke studiu: 4 hodiny

Cíl Po prostudování tohoto odstavce budete umět

- rozlišit rozdíl mezi vyplňováním geometricky a matematiky popsané hranice oblasti
- aplikovat algoritmus semínkového vyplňování
- vysvětlit a použít algoritmus řádkového semínkového vyplňování

Výklad

Metody řádkového vyplňování žádají geometrickou definici hranice. Vyplňování již vykreslené hranice zajišťují algoritmy se společným názvem *semínkové vyplňování (seed fill).*

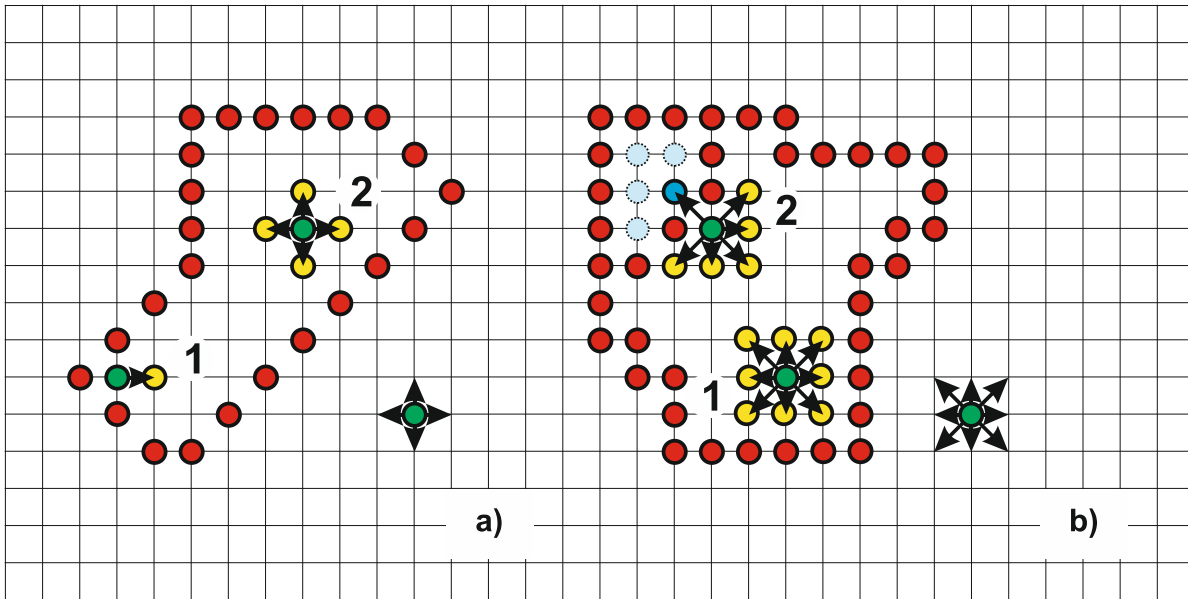
3.10.1Semínkové vyplňování

Vstupním parametrem těchto metod je semínko (vnitřní bod oblasti) a jeho souřadnice. Semínko vyraší, tím zabarví místo kde vyklíčilo. Nová semínka se dostanou do sousedství původního zasazeného semínka. Tam vyklíčí, obarví místo, uzrají, mají nová semínka a tak vegetace rozšíří po celé oblasti. Tam, kde již semínka vyrostla, nová nevyklíčí.

Informace o vyplňované oblasti se získávají čtením přímo z obrazové, rastrové paměti. Od semínka – vstupního bodu - se postupně rozšiřuje prohledávání obrazové paměti. Nalezeným vnitřním bodům se nastaví nová barva.

Jak se semínka šíří, souvisí s tím, jak definujeme hranici. Tato definice závisí na tom, které body v rastru považujeme za sousední. Rastr, jak víme, tvoří plocha složená ze shodných čtverců či obdélníků. Jestliže za sousedy považujeme ty elementy, které sousedí přes hranu, pak mluvíme *4spojité (4souvislé)* oblasti. Pokud za sousedy považujeme i ty, které mají společnou hranici jen ve vrcholu elementu, pomluvíme o 8*spojité (8souvislé).*

Jaký typ hranice *4spojitá či* 8*spojitá* musí vymezovat *4spojitou či* 8*spojitou* oblast. Vezměte si čtverečkovaný papír a tužku nakreslete tu i onu hranici, zasaďte semínko a nechte je růst. Podaří se mu rozšířit do širého světa nebo bude uvězněno za neprostupnou hranicí?



Obr 37. spojitá oblast (a) ohraničená 8spojitou hranicí a 8spojitá oblast (b) ohraničená spojitou hranicí.

Například Bresenhamův algoritmus vytváří 8spojité posloupnosti pixelů, které ohraničují 4spojité oblasti (Obr 37). 8spojitá hranice není neprostupná pro ohraničení 8spojité oblasti. Vyplňovací algoritmy by takovou hranici mohly překonat, což by vedlo k vyplnění rastrových bodů i za takto nakreslenou hranicí. Pro ohraničení 8spojité oblasti musíme použít 4spojitou hranici. Většina systémů pracuje s 8spojitou hranicí a 4spojitou oblastí.

Obrázek 37 zobrazuje dvě oblasti s jejími hranicemi. Oblast a) je 4spojitá ohraničená 8spojitou hranicí. Jsou zde naznačena dvě semínka. Semínko 1 leží těsně u hranice oblasti. Semínko samo je zelené, dostupní sousedé jsou označeni žlutě. Šipkami jsou vyznačeny směry šíření semínka. Vidíme, že ač je hranice oblasti 8smisměrná, semínko se za tuto hranici nevysemení. Nedojde k přebarvení pixelů mimo hranici oblasti. Druhé semínko je v této oblasti umístěno dále od hranice, šíření je možné všemi čtyřmi směry: doleva a doprava, nahoru a dolů.

Druhá oblast, na obrázku 37 označená jako oblast b) je 8spojitá. Hranice takové oblasti musí být 4spojitá. Všimněme si dvou semínek. Semínko číslo jedna je umístěno uvnitř oblasti, směry šíření jsou opět vyznačeny šipkami. Uvnitř 4směrné hranice je část 8spojité oblasti oddělena od zbytku oblasti 8spojitou hranicí. U této 8spojité hranice leží semínko číslo dvě. Místa, kam se semínko může rozšířit, jsou i zde vyznačeny šipkami. Žlutě jsou označeny body ležící uvnitř oblasti ohraničené 8spojitou oblastí. Modře je označen bod vně této hranice. Semínko se zde rozšířilo i za tuto hranici a všechny další body, dostupné z místa seménka číslo dvě jsou vyznačeny tečkovanými světle modře vybarvenými body. Zdůrazněme tedy, že pokud je jen část hranice 8spojitá, je takto třeba chápat celou hranici.

Při semínkovém vyplňování postupujeme od zadaného semínka a zkoumáme, zda sousedé náleží také k vnitřku oblasti. O příslušnosti bodu k oblasti rozhodneme podle určené vlastnosti. Takovou vlastností může být barevný odstín. Vlastní test příslušnosti můžeme provést dvěma způsoby. Testovat, zda bod má odlišnou vlastnost od bodu náležejícímu oblasti, nebo zda je jeho testovaná vlastnost shodná se semínkem.

1. Hraniční vyplňování
Testovaný bod je vnitřní, má-li testovanou vlastnost odlišnou od vlastnosti hranice.

2. Záplavové vyplňování
Testovaný bod je vnitřní, má-li shodnou vlastnost jako zadané semínko. Mluvíme pak o lavinovém vyplňování či *přebarvování.*

Výhodou těchto metod je to, že vlastní test nemusí ověřovat úplnou shodu vlastností, ale můžeme testovat, zda vlastnost přísluší jistému intervalu hodnot. Takovýchto testů se hojně využívá při úpravách digitálních fotografií, kdy například hodláme zaměnit část nevyhovujícího pozadí apod.

Dříve než budete číst dále, zamyslete se, jak byste algoritmus lavinového vyplňování řešili vy?

Řekli jsme, že se semínko rozšíří na všechna dostupná místa. Tedy zasaď semínko, nasaď semínka ke všem dostupným sousedům. Nejjednodušší metoda, z hlediska zápisu velmi elegantní, je rekurzívní algoritmus 3.7. S rekurzí jste se již mnohokrát jistě setkali, například s rekurentními vzorci. Přesto neuškodí připomenout si její princip. Zde se zaměříme na programátorské hledisko. Všimněme si funkce ZasaďSemínko. Parametrem této funkce je pozice bodu, kam semínko umístit. Jak tedy zajistit jeho šíření? Jednoduše. Máme přece k dispozici funkci ZasaďSemínko! Netrapme se tím, že jsme ji ještě nedopsali. To dožene později. Uvnitř funkce ZasaďSemínko zavoláme tuto funkci znovu. Toto volání samotné funkce, ať přímo či nepřímo, nazýváme rekurzivní volání funkce či zkráceně rekurze. Na první pohled elegantní, přehledný zápis je ve skutečnosti téměř nepoužitelný. Šíření semínek do všech směrů má za následek, že každý vnitřní pixel je několikrát testován i když už byl obarven. Čtení z obrazové je přitom pomalé. Celý algoritmus uveden v

1. ZasaďSemínko (x, y)

    1a. pokud je bod [x, *y*] vnitřním bodem a současně nebyl obarven, pak

        1a-i. obarvi bod [*x*, *y*] požadovanou barvou

        1a-ii. ZasaďSemínko (*x*+1, *y*)

        1a-iii. ZasaďSemínko (*x*-1, *y*)

        1a-iv. ZasaďSemínko (*x*, *y*+l)

        1a-v. ZasaďSemínko (*x*, *y*-1)

3.10.2 Řádkové semínkové vyplňování

Vylepšením semínkového vyplňování je algoritmus nazývaný *řádkové semínkové vyplňování (scan-line seed fill).* Je uvedena v algoritmu 3.8. Myšlenkou je vyplňování souvislých vodorovných úseků a prohledávat místa nad a pod těmito úseky. Každá souvislá vodorovná řada vnitřních bodů nad či pod daným úsekem tvoří nový úsek. Ten je pak rekurzívně zpracován. Algoritmus inicializujeme nalezením prvého úseku obsahující zadané semínko.

Na obrázku (Obr 38) je nakreslen stav vyplňované oblasti při postupném provádění algoritmu. Startovním semínkem je bod *A*. Při inicializaci je nalezen úsek obsahující tento bod a končící na hranicích oblasti. Poté je možno zavolat algoritmus 3.8.

Po vyplnění úseku v okolí bodu *A* jsou testovány intervaly nad a pod tímto úsekem. Výsledkem je volání algoritmu pro úseky s bodem *B* (z horní oblasti) a *C* (z dolní oblasti). Po vyplnění úseku s bodem *C* jsou nalezeny volné úseky s body *D*, *E* a *F*. Postupně je vyplněna oblast pod úsekem *E*, nad úsekem *D* a nakonec nad úsekem *B*.

Algoritmus 3.8: Řádkové semínkové vyplňování

1. vyplň pixely v úseku od [x_L, *y*] do [x_R, *y*]
2. v (horním) intervalu mezi [x_L, *y* –1] a [x_R, *y* – 1] hledej souvislé vnitřní úseky. Pro každý i-tý úsek proved':

   2a. VyplňÚsek (*y* - 1, x_{Li}, x_{Ri})
3. v (dolním) intervalu mezi [x_L, *y* + 1] a {x_R, *y*+ 1] hledej souvislé vnitřní úseky. Pro každý j-tý úsek proved:

   3a-i. VyplňÚsek *(y* + 1, x_{Lj}, x_{Rj}*)*

   3a-ii. VyplňÚsek *(y,* x_L, x_R*)*

Obr 38. Postup při řádkovém semínkovém vyplňování

Shrnutí poznatků

Vstupním parametrem **semínkového vyplňování** je **semínko** (vnitřní bod oblasti) a jeho souřadnice.
Informace o vyplňované oblasti se **získávají čtením přímo z obrazové, rastrové paměti**.
*Rozeznáváme **4spojité** (4souvislé)* oblasti a **8***spojité (8souvislé)* oblasti.
Určíme je **podle počtu sousedů** daného bodu.
Čtyři sousedi určují 4spojité *(4souvislé)* oblasti. Pro ohraničení 8spojité oblasti musíme použít 4spojitou hranici.
Pro realizaci semínkového vyplňování **není vhodný rekurzívní** algoritmus. Docházelo by mnohočetným testům již otestovaných bodů.
Vylepšením semínkového vyplňování je algoritmus nazývaný ***řádkové semínkové vyplňování*** *(scan-line seed fill).*

Otázky

Popište ideu semínkového vyplňování.
Proč není rekurzívní algoritmus vhodný pro jeho implementaci?
Jaký efektivní algoritmus semínkového vyplňování znáte a v čem spočívá jeho princip.

3.11 Ořezávání dvourozměrných objektů

Čas ke studiu: 2 hodina

Cíl Po prostudování tohoto odstavce budete umět

- popsat základní činnosti, které vykonávají ořezávací algoritmy
- vysvětlit důležitost ořezávání objektů
- aplikovat algoritmus ořezávání úsečky metodou Cohen-Sutherlanda
- využít výhod parametrického způsoby uřezání úseček
- vyjmenovat těžkosti při ořezávání polygonu

Výklad

Ořezávací *(clipping)* algoritmy jsou mnohem důležitější, než by se na prvý pohled zdálo. Zdaleka nejde jen o výběr oblasti zobrazované na monitoru. Zde se využívají specializované algoritmy. Výřez má totiž strany shodně orientované se souřadným systémem. V dnešní době se velmi rozšířila třírozměrná počítačová grafika. O to, co při vnímání reality je pro nás samozřejmostí, totiž viditelnost jednotlivých předmětů na scéně, je pro počítačové zpracování 3D grafiky stěžejní. A pokus si uvědomíme, že silueta blízkého předmětu „ořeže“ ty viditelnost těch předmětů, které jsou umístěny za tímto přemetem, je snad každému jasné, že metody ořezávání obrazů jsou v počítačové geometrii stěžejní otázkou.

Proto budeme v následujícím textu věnovat pozornost alespoň základním algoritmem ořezávání. Zásadní důležitost zde mají algoritmy pro ořezávání polygonů. Používají se i obecnější algoritmy pro ořezání nekonvexním mnohoúhelníkem.

Ořezávací algoritmy provádějí:

1. Test *polohy bodu*

Testuje se, zda bod patří či nepatří do zadané ořezávací oblasti. Jedná se o univerzální test a využívají jej algoritmy pracující s jednotlivými body.

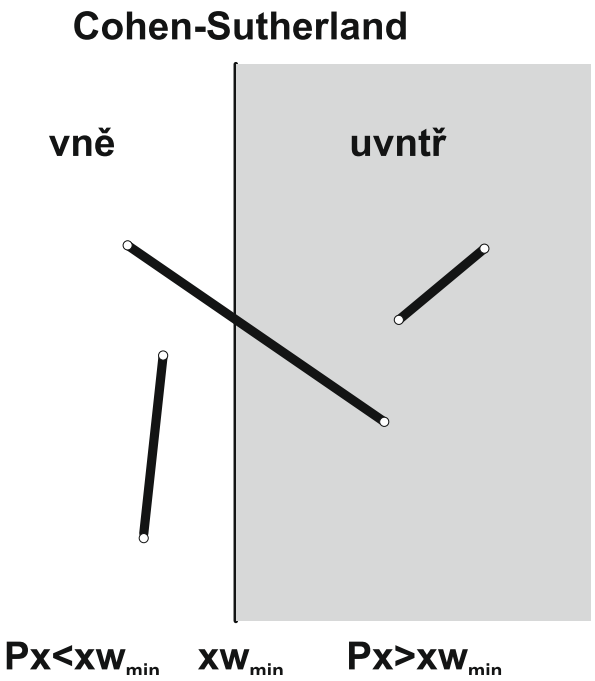
2. *Ořezání úsečky*

Ořezání úsečky použijeme na ořezání liniové objektů ,úsečky, lomené čáry i křivky, kterou jsme ovšem nahradili posloupností úseček.

3. *Ořezání oblasti*

Algoritmy pro ořezání uzavřené oblasti musí pracovat tak, aby výsledkem ořezání byla opět jedna či více o uzavřených oblastí.

Výchozím postupem v našich dalších úvahách bude ořezání obdélníkovým oknem. Jsou-li hrany tohoto okna orientovány ve směru souřadných os, pak toto okno je zcela určeno pravým horním a levým dolním rohem. Tyto souřadnice označme [xo_{max}, yo_{max}] a [xo_{min}, yo_{min}] (Obr 39)



Obr 39. Test pozice bodu vzhledem k hranici

3.11.1 Test polohy bodu vzhledem k oknu

Tento test určuje, zda bod leží ve vnitřní nebo vnější polorovině vymezené hraniční přímkou. Pokud je okno osově orientované dostačuje jednoduché porovnání příslušné souřadnice s hraniční hodnotou, algoritmus je znám jako Cohen-Sutherland.

Pokud je okno orientováno obecně, pak s výhodou využijeme rovnici hraniční přímky, kterou ale zapíšeme ve tvaru

$$F(x,y) = ax + by + c = 0$$

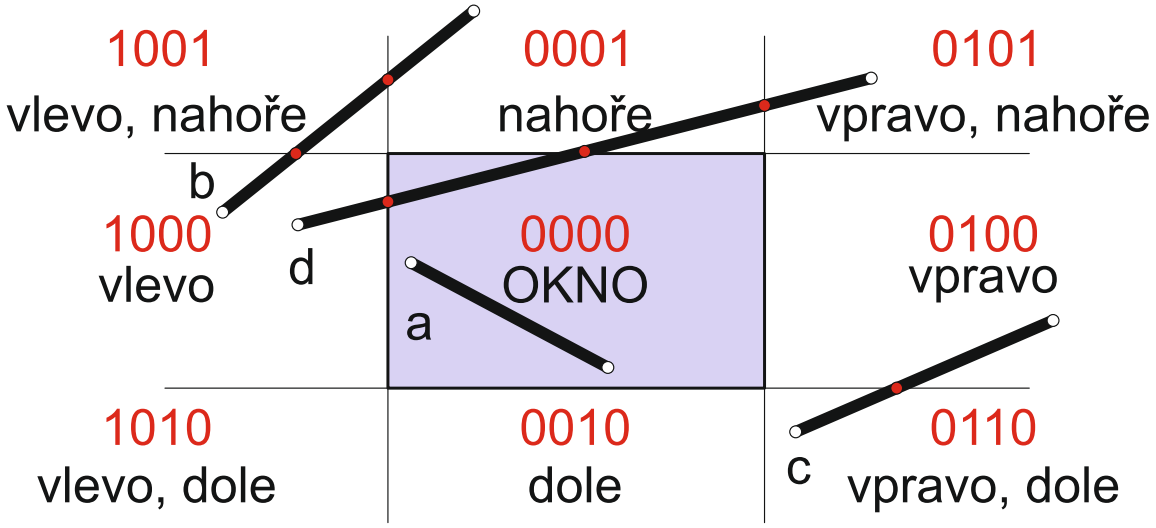
Pozici --bodu otestujeme tak, že jeho souřadnice dosadíme do rovnice xx. Pomocí testu $F(xp, yp) > 0$, $= 0$, < 0 určíme pozici bodu vůči hranici. Algoritmus Cyrus-Beck využívá normálového vektoru hraniční přímky.

3.11.2 Ořezání úsečky

Algoritmus Cohen-Sutherland

Algoritmus předpokládá ořezové okno s hranami rovnoběžnými s osami souřadnicového systému. Algoritmus *Cohen-Sutherland* (Spro68) postupuje následovně:

ohodnotí koncové body úsečky tzv. *hraničními kódy*. Dostačuje čtyřbitová informace. Jednotlivé bity (zleva doprava) určují, zda bod se nachází vlevo, vpravo, dole či nahoře. Rovinu takto rozdělíme na devět oblastí. Obrázek (Obr 40) zobrazuje tyto oblasti společně s příslušným kódem.



Obr 40. Kódy devíti oblastí určených oknem

Úsečka a okno mohou vzájemně zaujímat tři pozice.

1. Úsečka je celá v okně.
2. Úsečka prochází hranou nebo hranami okna.
3. Úsečka celá leží mimo okno.

Koncové body úsečky označme K a L, hodnoty hraničních kódů těchto bodů $Pozice(K)$ a $Pozice(L)$.

Množinové operace sjednocení a průniku zde budou aplikovány po jednotlivých bitech. Pak mohou nastat tyto možnosti:

1. $Pozice(K) \cup Pozice(L) = 0$ úsečka je celá v oknu a ořezávat nebudeme. To je případ úsečky a na obrázku 3.22.
2. $Pozice(K) \cap Pozice(L) \neq 0$ Celá úsečka leží mimo okno a nebudeme ji dále zpracovávat.

Tento test vyřadí úsečky:

- jejichž koncové body leží ve stejné oblasti mimo okno (mají totožné kódy)
- některé úsečky procházející více vnějšími oblasti. Příkladem je úsečka *b* na obrázku 3.22.

- některé úsečky nelze tímto testem vyhodnotit. Příkladem může být úsečka *c*

3. $Pozice(K) \cap Pozice(L) = 0$ Úsečka prochází několika oblastmi a je třeba ji oříznout. Ořezání provedeme takto:
   3a. vybereme koncový bod s nenulovým (neprázdným) hraničním kódem
   3b. podle nastavené jedničky v kódu vybereme hranici a podle této hranice úsečku zkrátíme. Nemusíme počítat průsečík přímek. Hraniční přímky jsou zde totiž rovnoběžné se souřadnicovými osami. Proto jedna z nových souřadnic bude vždy totožná s mezní souřadnicí okna. Například jedna ze svislých hraničních přímek má rovnici $x = 7$. Pak tato souřadnice pak bude i souřadnice x hraničního bodu. Z obecné rovnice přímky pak dopočteme její zbylou souřadnici. Rovnici přímky zadanou dvěma body uvádí rovnice xx.
4. Zopakujeme testy s aktualizovanými koncovými body ořezané úsečky.

Poznamenejme, že postupně vypočítávané koncové body nemusí být skutečnými koncovými body ořezané úsečky. U úsečky d může být například nejprve nalezen průsečík s pravou hranicí okna a teprve při dalších testech průsečík s horní hranicí okna. Obdobně je tomu u úsečky c. Nejprve je zkrácena a teprve poté vyloučena jako zcela vnější.

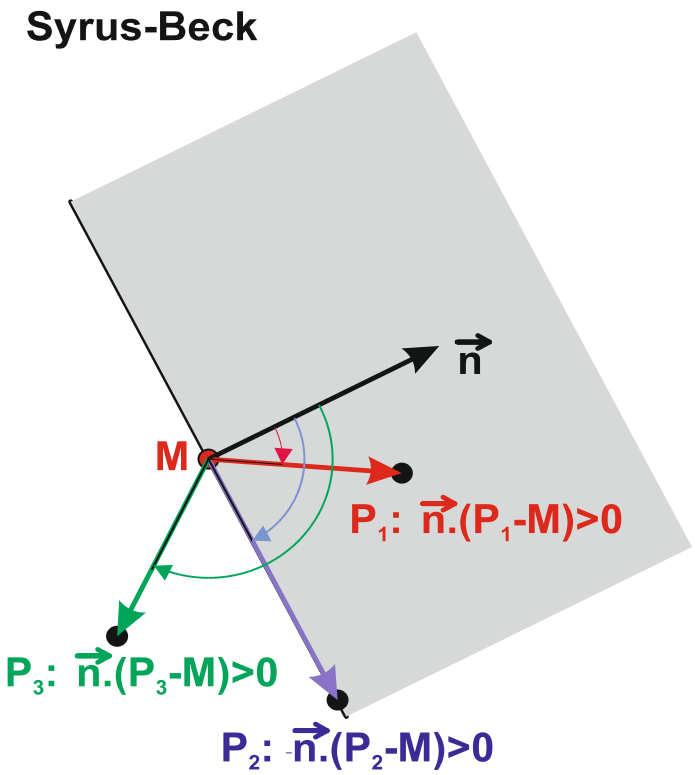
3.11.3 Parametrické ořezávání úseček - algoritmus Cyrus-Beck

Algoritmus Cyrus-Beck
Vyjděme z parametrické rovnice přímky

$$P(t) = P_1 + (P_2 - P_1)t \tag{66}$$

Tato přímka je určena body P_1 a P_2. Pokud parametr t náleží intervalu $t \in \langle 0,1 \rangle$, pak rovnicí (66) je definována úsečka P_1P_2. Abychom mohli tuto úsečku ořezat danou konvekční hranicí, musíme určit, které body přímky $P(t)$ (úsečky P_1P_2 je součástí přímky) leží vně a které uvnitř hranice konvexní oblasti. Pro test využijeme normálový vektor hranice $\mathbf{n}$, který směřuje dovnitř oblasti. Na obrázku 3.21 bodem M prochází hranice oblasti *a* $P(t)$ je testovaný bod. Využijeme zde vlastnosti skalárního součinu dvou vektorů. Vektor $\mathbf{p} = P(t) - M$, Hodnota skalárního součinu $\mathbf{n}.\mathbf{p} = \mathbf{n}.[P(t) - M]$ rozliší tři případy, viz obr 41.

- $\mathbf{n}.\mathbf{p} < 0$ odpovídá bodu P_1, pak vektor $\mathbf{p}$ směřuje z bodu *M* ven z oblasti
- $\mathbf{n}.\mathbf{p} = 0$ odpovídá bodu P_2, pak vektor $\mathbf{p}$ je rovnoběžný s hranicí
- $\mathbf{n}.\mathbf{p} > 0$ odpovídá bodu P_3, pak vektor $\mathbf{p}$ směřuje z bodu *M* dovnitř oblasti.



Obr 41. Určení hranice postupem Syrus-Beck

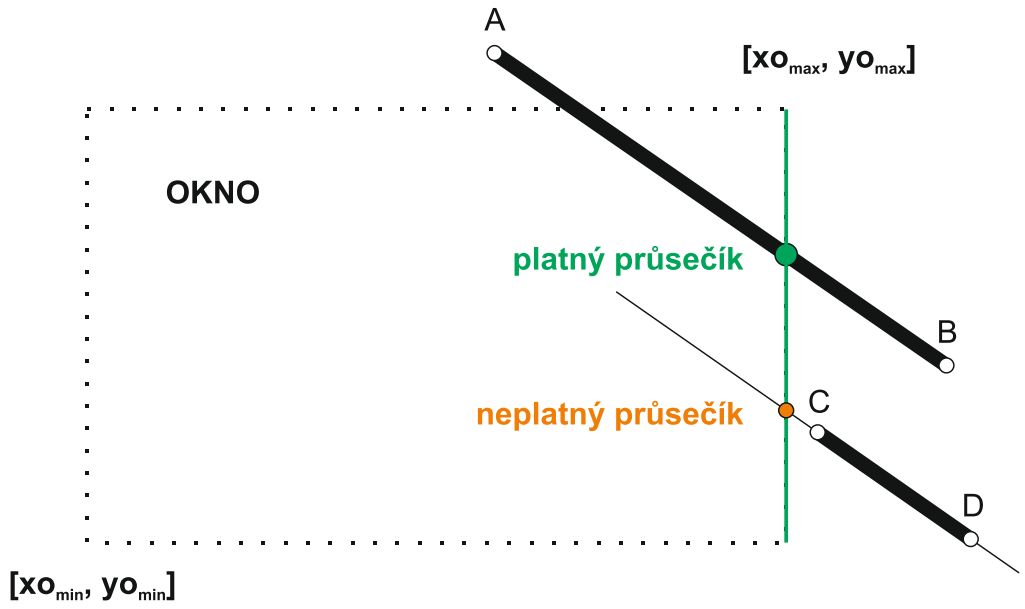
Po dosazení *za* $P(t)$ vypočteme průsečík (určíme hodnotu parametru t parameterické rovnice přímky) pomocí rovnice

$$n.p = n.[P(t) - M] = n.[P_1 + (P_2 - P_1)t - M] = nP_1 + n(P_2 - P_1)t - nM = 0 \quad (67)$$

$$t = \frac{n(M - P_1)}{n(P_2 - P_1)} \quad (68)$$

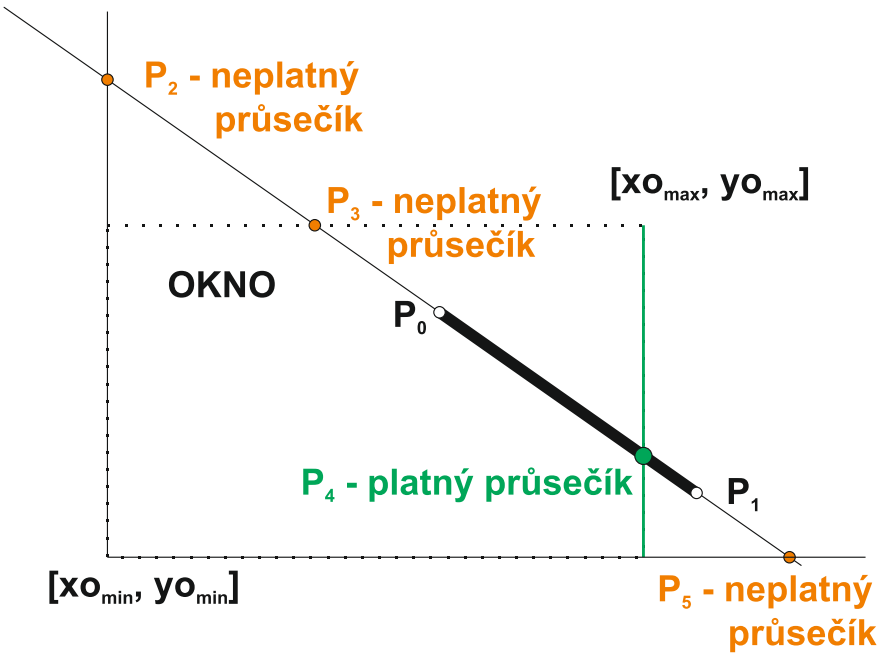
V této rovnici označíme $\mathbf{w} = P_2 - P_1$ jako váhový faktor (weight) a $\mathbf{d} = (M - P_1)$ jako směrový vektor. Ve dvou krocích otestujeme

1. Je-li skalární součin $\mathbf{n}.\mathbf{d} \neq \mathbf{0}$ nenulový, pak existuje průsečík s hraniční přímkou *i*-té hranice.
2. V tomto případě je pak vypočten průsečík přímky s hranicí oblasti (parametr *t*). Dalšími testy je tento bod označen za platný průsečík s hranicí oblasti.



Obr 42. Platný a neplatný průsečík

Abychom nebyli závislí na pořadí hraničních přímek, při hledání průsečíků zároveň hledáme maximální t_{high} a minimální t_{low} hodnotu parametru t. Na obrázku (Obr 43) vidíme, že obecná přímka p s ořezávanou úsečkou protíná všechny čtyři hraniční přímky oblasti. Shora jako neplatné



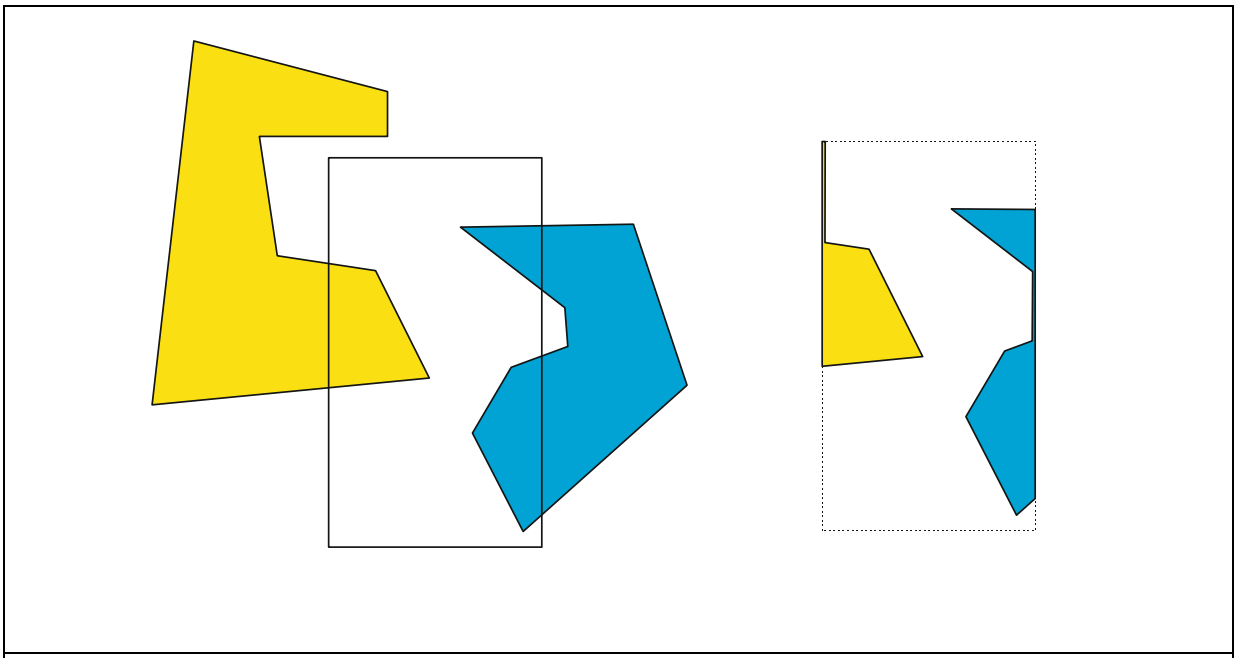
Obr 43. Vyloučení neplatných průsečíků

jsou vyloučeny průsečíky t_2 a t_3. Zdola vyloučíme jako neplatný průsečík t_5 a původní koncový pod úsečky t_0. Jako jediný platný průsečík zůstává průsečík t_4. Tento průsečík se stává novým koncovým bodem úsečky za vyloučený bod t_0. Ořezanou úsečku pak tvoří úsečka $P(t_4)P(t_0)$ neboli $P(t_4)P(0)$.

Výpočetní postup je popsán v algoritmu 3.9.

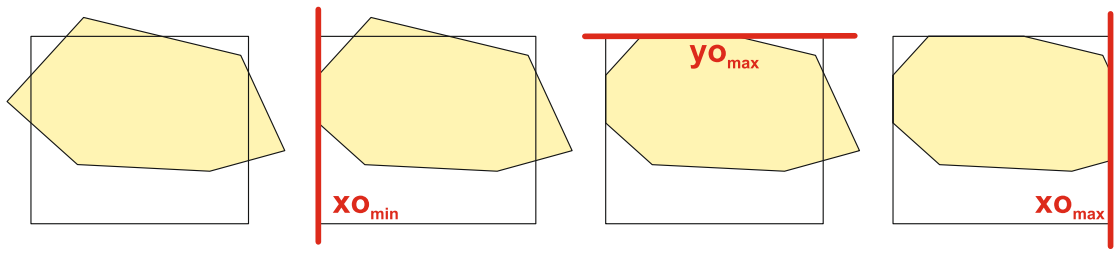
3.11.4 Ořezání polygonu

Ačkoliv algoritmus ořezání polygonů v sobě obsahuje ořezání hranic polygonu oknem, nelze jej provádět jednoduše po jednotlivých hraničních úsečkách polygonu. Tím by mohlo dojít k rozpadu hranice na samostatné, nenavazující úsečky a nebylo by pak například možno polygon vyplnit barvou. Algoritmus proto musí zajistit uzavřenost hranice polygonu případným doplněním nových úseček. Na obrázku (Obr 44) vlevo vidíme dva polygony. Oříznuté polygony vpravo mají čtyři hrany, resp. osm hran. Došlo by k rozpadu hranice na samostatné, nenavazující úsečky a nebylo by pak například možno polygon vyplnit barvou. Algoritmus proto musí zajistit uzavřenost hranice polygonu případným doplněním nových úseček.



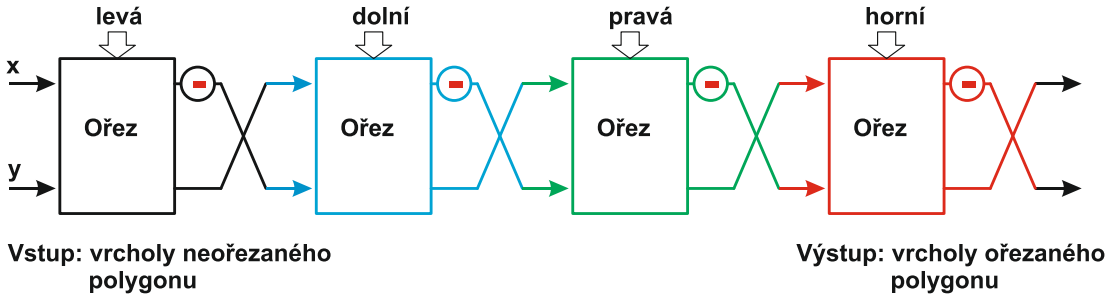
Obr 44. Rozpad hranice ořezaného polynomu

Postup při ořezání polygonu obdélníkovým oknem lze logicky rozdělit do čtyř kroků - v každém s nich se provede ořezání jednou hranicí okna, jak ukazuje obrázek (Obr 45). Při implementaci je vhodné provést následující úpravy:



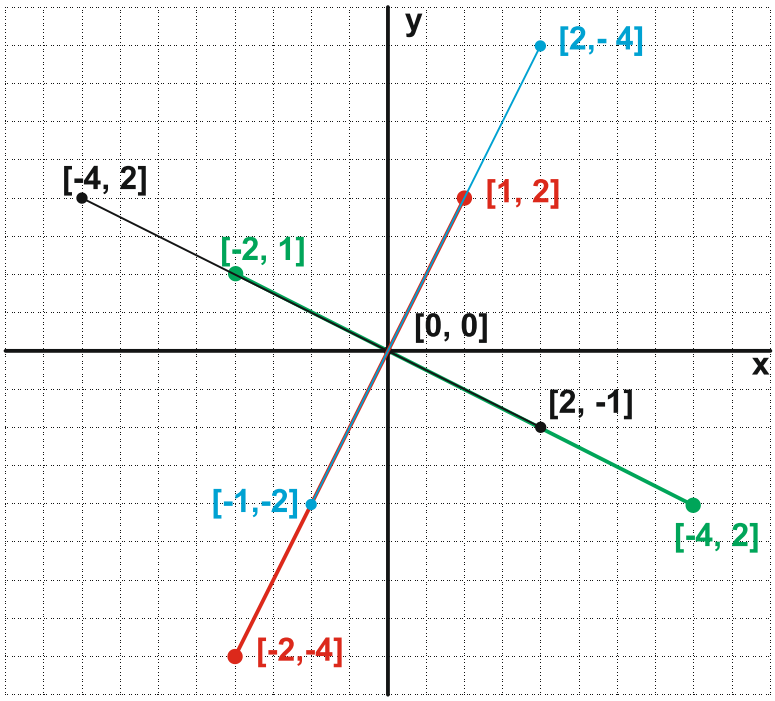
Obr 45. Ořezání polygonů

Abychom nemuseli po dílčím ořezání jednou hranicí zaznamenávat (proměnlivý) počet vrcholů polygonu, je vhodné každou již ořezanou hranu okamžitě oříznout další hranicí. Tento postup je anglicky nazýván *re-entrant polygon clipping* a je znám jako *Sutherland-Hodgrraanův* algoritmus. Zjednodušení testů polohy bodu vůči hranici okna můžeme docílit tím, že budeme stále ořezávat svislou polorovinou. K tomu stačí, abychom vždy před ořezání další hranicí zaměnili souřadnice a



Obr 46. Použití stejného kódu pro ořezání čtyřmi hranicemi s využitím otáčení o 90°

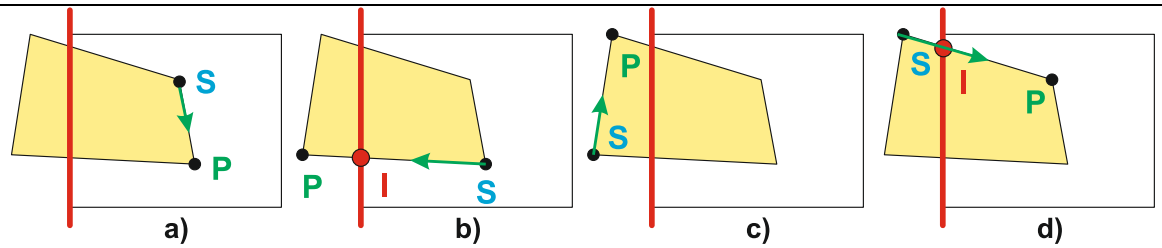
jedno znaménko koncových bodů ořezávané úsečky. Tak provedeme otočení o 90° a můžeme použít stejný kód lišící se pouze jedním parametrem konkrétní hranice. Situace je naznačena na obrázku 3.26.



Obr 47. Změna polohy úsečka při záměně pořadí souřadnic koncových bodů změně jejich znaménka.

Na obrázku 47 je znázorněno, co se bude dít, pokud u úsečky budeme měnit souřadnice koncových bodů. Originální úsečka je černá. Souřadnice x změní znaménko a stane se souřadnicí y otočené úsečky a původní souřadnice y je novou souřadnicí x otočené úsečky. Původní souřadnice koncových bodů úsečky $[-4,2]$ a $[2,-1]$ se změní na $[-2,4]$ a $[-1,-2]$. Na obrázku je to modrá úsečka. Dalším

shodným postupem dostaneme souřadnice koncových bodů zelené úsečka, v dalším kroku červené úsečky. Pokud bychom záměnu provedli ještě jednou, pak bychom obdrželi souřadnice původní, na obrázku černé, úsečky.Sutherland-Hodgmanův algoritmus zpracovává v každém vnitřním kroku právě jeden vrchol *P* ořezávaného polygonu. Tento vrchol spolu s naposledy oříznutým předchozím vrcholem *S* tvoří elementární úsečku. Na obrázku 3.27 jsou znázorněny možné polohy úsečky *SP* vůči svislé hranici (vnitřek okna je vpravo od hranice). Průsečík úsečky s hranicí je označen jako *I*.



Obr 48. Polohy úsečky vůči hranici okna

1. Inicializace:

   1a. $t_1 = 0, t_{high} = 1, \mathbf{d} = P(1) - P(0)$.

   1b. $Ořezaná Úsečka = [P(t_{high}), P(t_{low})]$

2. Pro Všechny hranice i s vnitřní normálou $\mathbf{n_i}$ Dělej:

   2a. Pokud $\mathbf{d}.\mathbf{n_i} \neq 0$

   2a-i. Pak $t = -\frac{\mathbf{n_i}.\mathbf{d}}{\mathbf{n_i}.\mathbf{w}}$

   2a-ii. Pokud $\mathbf{d}.\mathbf{n_i} > 0$ AND $t \leq 1$

   1. Pak $t_{high} = \max(t, t_{high})$ /* oprav průsečík t_{high} */

   2. Jinak Pokud $\mathbf{d}.\mathbf{n_i} > 0$ AND $t \leq 0$

      a. Pak $t_{low} = \min(t, t_{low})$ /*oprav průsečík t_{low}*/

         i. Jinak Pokud $\mathbf{n_i}.\mathbf{w_i} < 0$ Pak UkončiVýpočet;

      b. /* úsečka je zredukována na bod mimo okno */

3. Pokud $t_{high} < t_{low}$

   3a. Pak $Ořezaná Úsečka = [P(t_{high}), P(t_{low})]$

   3b. Jinak *ÚsečkaLežíMimoOkno*

Algoritmus 3.9: Parametrické ořezávání úsečky - algoritmus Cyrus-Beck

Shrnutí pojmů

Ořezávací *(clipping)* algoritmy jsou důležité pro vykreslování 3D scény.

Využívají specializované algoritmy.

Ořezávací algoritmy provádějí: Test ***polohy bodu vůči hranici oblasti, ořezání úsečky a ořezání oblasti***

Výchozím postupem je **ořezání obdélníkovým oknem**

Algoritmus Cohen-Sutherland k posouzení polohy koncových bodů úsečky kódu. Tyto kódy jsou sestaveny tak, aby jednoduše umožnily otestovat polohu úsečky vůči oknu.
Parametrické ořezávání úseček - algoritmus Cyrus-Beck - využívá normálového vektoru hranice oblasti a směrového vektoru úsečky.
Při ořezávání polygonu je nutno doplnit rozpadlé hranice.
Využívá se stejný kód s využitím otáčení úsečky vůči souřadnému systému záměnou znaménka a pořadí souřadnic koncových bodů úsečky.

Otázky

1. Definujte důležitost ořezávacích algoritmů.
2. Jaké metody se využívají k testování polohy úsečky vůči ořezovému oknu.
3. Vyjmenujte základní metody testování polohy úsečky vůči oknu.
4. Lak lze využít otáčení úsečky při ořezávání oblasti polygonu.

Úlohy k řešení

4 KŘIVKY A PLOCHY

Čas ke studiu: 20 hodin

Cíl po prostudování kapitoly budete schopni

- .určit hlavní způsoby zadávání křivek
- znát hlavní druhy interpolačních křivek
- porozumíte problému navazování křivek a jejich hladkosti
- porozumíte hlavním principům modelování křivek
- vědět jak se zadávají Hermitovské kubiky
- orientovat v oblasti aproximačních křivek
- poznáte výhodu Beziérových křivek
- popsat využití B-spline křivek

Výklad

Počítačovou grafiku bez křivek a ploch si snad ani nelze představit. Jsou všudypřítomné. Pomocí křivek jsou definovány grafické fonty. Jsou nezbytné při počítačové animaci pohybu. Obdobně i plochy slouží k modelování rozličných povrchů scény. Požadavky kladené na modelování křivek a ploch jsou případ od případu, aplikace od aplikace různé a tak je tato oblast velice široká. Zdaleka si neklademe za cíl podrobně popsat všechna zákoutí této široké tématiky, pokusíme se ale vybrat ty nejdůležitější aspekty tak, aby student získal základní představu o důležitost křivek ploch v počítačové grafice.

Již v roce 1959 P. de Casteljau u firmy Citroen vypracoval matematický model křivek a ploch tak, aby se snadno zadávaly jejich parametry. V šedesátých letech u firmy Renault skupina kolem P. Béziere vyvinula pro návrh křivek a ploch programový systém UNISURF.

V současné době je snaha o sjednocení různorodých přístupů. Rozšířilo se používání racionálních B-spline křivek a ploch s neuniformní parametrizací (Non-Uniform Rational B-Spline). Pomocí jednotné metody jsme schopni generovat jak klasické geometrické prvky (úsečky, kružnice, elipsy a v prostoru koule, válce atp.) tak i vytvořit křivky a plochy se složitými průběhy a tvary.

4.1 Vlastnosti křivek

Křivky v počítači mohou být zadány trojím způsobem:

1. Explicitně jako spojitá funkce ve tvaru

$$y = f(x) \tag{69}$$

Známým příkladem může být rovnice přímky $y = kx + q$

2. *Implicitně ve* tvaru

$$F(x,y) = 0 \tag{70}$$

Určitě si vzpomeňte na rovnici elipsy $\frac{x^2}{a} + \frac{y^2}{b} - 1 = 0$. Zamyslete se, kde jsme se již setkali s takto zadanými křivkami. Určitě jste se si vzpomněli na rasterizaci kružnice a elipsy. Implicitní zadání se také využívá při testování oblastí vymezených implicitně zadanou křivkou nebo při výpočtu průsečíku paprsku s křivkou.

Parametrický tvar se v počítačové grafice používá nejčastěji (5.1). Souřadnice pohybujícího se bodu jsou funkcemi parametru t (času)

$$x = x(t), y = y(t), z = z(t) \tag{71}$$

Parametr $t \in \langle t_{min}, t_{\max} \rangle$ interval bývá v rozsahu $t \in \langle 0,1 \rangle$

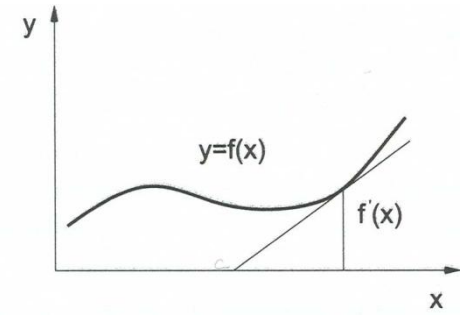
. *Bodovou rovnici* křivky zapíšeme ve tvaru

$$Q(t) = [x(t), y(t), z(t)] \tag{72}$$

Souřadnice $x, y, \text{a}\, z$ lze chápat jako složky polohového vekoru $\vec{q}(t) = Q(t) - [0,0,0]$, *vektorová rovnice má* tvar

$$\vec{q}(t) = \big(x(t), y(t), z(t)\big). \tag{73}$$

Výhodou parametrického zápisu je závislost souřadnic křivky na jediném parametru t, jehož fyzikální interpretace může být čas.



Obr 49. Derivace funkce

Tečný vektor je určen derivací parametrické funkce křivky podle parametru t

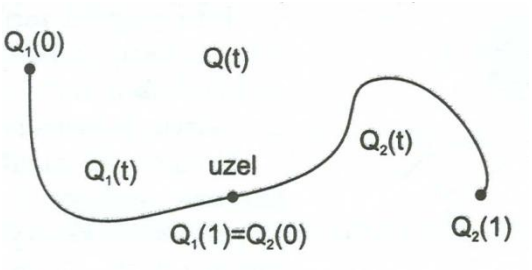
$$\vec{q}'(t) = \frac{\vec{q}(t_0)}{dt_0} = (x'(t_0), y'(t_0), z'(t_0)) = \left(\frac{x(t_0)}{dt}, \frac{y(t_0)}{dt}, \frac{z(t_0)}{dt}\right) \quad (74)$$

Dovedete z předchozích vztahů sami napsat rovnici *tečny v* bodě $Q(t_0)$?, *P*římka je určená bodem, kterým prochází a svým směrovým vektorem $\vec{q}'(t)$.

$$q(t) = \vec{q}'(t)\, t + Q(t_0)$$

Mějme dvě křivky $Q_1(t) a\, Q_2(t)$, které tvoří dvě části (říkáme jim segmenty) jediné křivky $Q(t)$. Křivky $Q_1(t)\, a\, Q_2(t)$ na sebe navazují ve společném bodě $Q_1(1) = Q_2(0)$ (viz obrázek. Společný styčný bod se nazývá *uzel (knot).*

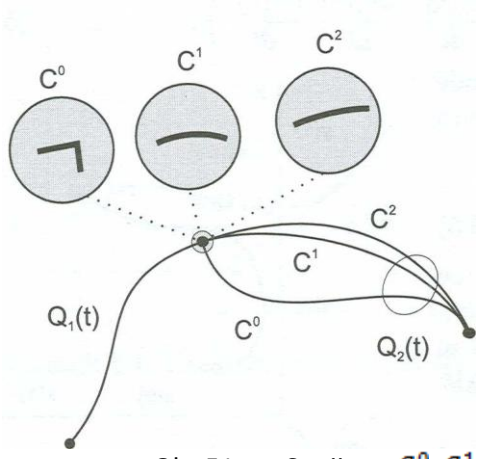
Segmenty $Q_1(t)$ *a* $Q_2(t)$ mohou být definovány různě. Důležitý je ale způsob napojení těchto segmentů. Udává *spojitost (continuity)*- výsledné křivky.



Obr 50. Křivka Q (t) vzniklá spojením dvou segmentů $Q_1(t)$ *a* $Q_2(t)$

Křivka *Q (t) je* třídy C^n *spojitá*, má-li ve všech bodech spojité derivace podle parametru t do řádu *n*. Označení C^n se říká *parametrická spojitost stupně n* (obr. 51).

- Dva segmenty jsou *spojitě* navázány, mají spojení třídy C^0, pokud je koncový bod prvního segmentu počátečním bodem segmentu druhého. Výsledná křivka je spojitá ale v bodě spojitosti může skokem změnit směr.
- Dva segmenty mají spojení C^1, pokud rovnají jejich tečné vektory ve společném bodě, tedy koncovém bodě Q_1 prvého segmentu a počátečním bodě Q_2 druhého. V uzlu je křivka spojitá a zachová rychlost.
- Pro spojitost C_2 je vyžadována rovnost vektoru první a druhé derivace. Tedy křivka je spojitá, nemění rychlost a zrychlení (pokud budeme křivku chápat jako dráhu vykonanou prostoru za nějaký čas.



Obr 51. Spojitost C^0, C^1 a C^2

Hladkost křivek se posuzuje také podle *geometrické spojitosti* označované G^n. Většinou se setkáte s geometrickými spojitostmi G^0 a G^1.

- Dva segmenty křivky $Q(t)$jsou G^0 spojité, pokud je koncový bod Q_1 totožný s počátečním bodem Q_2.
- Dva segmenty jsou G^1 spojité, pokud jsou G^0 spojité a mají v uzlu totožné tečny; tečné vektory $q_1'(1)$ segmentu Q_1 a $q_2'(0)$ segmentu Q_2 jsou souhlasně kolineární, tj. platí:

$$q_1'(1) = kq_2'(0);\ k > 0 \tag{75}$$

Pohybující se bod v uzlu nemůže změnit skokem směr, ale může změnit skokem rychlost, křivka je přesto vizuálně hladká.

4.2 Modelování křivek

Základním druhem parametrických křivek používaných v počítačové grafice jsou křivky *polynomiální*

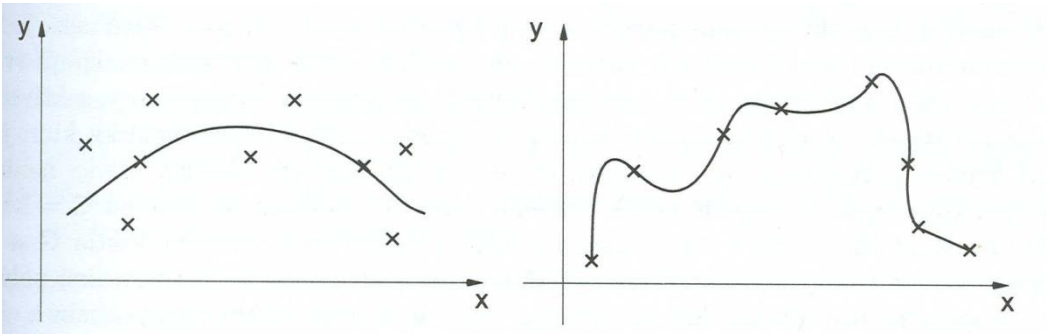
$$Q(t) = a_0 + a_1 t + a_2 t^2 \dots a_{n-1} t^{n-1} + a_n t^n \tag{76}$$

Stupeň polynomu určuje nejvyšší mocnina argumentu, zde proměnné t. Také je zřejmé, že polynom

nemusí obsahovat všechny mocniny t od nulté až po mocninu n-1. Pak hodnota příslušného koeficientu u dané mocniny proměnné je rovna nule. Nultá mocnina proměnné t, tedy jednička, se v zápise obvykle také neuvádí. Výhodou polynomiálních křivek je jejich snadná vyčíslitelnost a jsou jednoduše diferencovatelné. Nejpoužívanější křivky třetího stupně – *kubiky*. Základní vlastnosti kubik:

- jsou dostatečně tvarovatelné,
- jejich výpočet je nenáročný,
- lze s nimi snadno manipulovat,
- je u nich možné zaručit spojitost C2 často požadovaná při modelování v CADU systémech.

Je samozřejmě možné zadávat koeficienty polynomiálních křivek číselně. Běžnější postupem je definice několik *řídicích bodů (řídicí polygon)* a z jejich polohy určit průběh křivky. Pokud požadujeme spojitost a hladkost navázání segmentů je možno křivky zadat i pomocí tečných vektorů.



Obr 52. Aproximační (vlevo) a interpolační křivka a jejich řídicí body

V textu se budeme držet následujících konvencí:

- křivku budeme označovat $Q(t)$,
- její řídicí body P_i .
- při spojení dvou segmentů křivky
  - první segment označíme $Q_1(t)$ a bude zadán řídicími body P_i,
  - druhý segment budeme označovat $Q_2(t)$ a bude zadán řídicími body R_i.
- tečný vektor ke křivce označíme $\vec{p}_i{}'(t)$,
- jeho druhou derivaci pak $\vec{p}_i''(t)$.
- tečné vektory v řídicích bodech nebo uzlech označíme $\vec{r}_i'(t)$, resp. R_i.

Existují dva základní způsoby interpretace řídicích bodů a to *interpolace a* aproximace (viz obr. 52).

Křivka generovaná při interpolaci probíhá danými body, zatímco při aproximaci je řídicími body určen tvar křivky, ta jimi však procházet nemusí. Parametricky zadanou kubiku $Q(t)$ ve tvaru:

$$x(t) = a_x t^3 + b_x t^2 + c_x\, t + d_x \tag{77}$$

$$y(t) = a_y t^3 + b_y t^2 + c_y\, t + d_y \tag{78}$$

$$z(t) = a_z t^3 + b_z t^2 + c_z\, t + d_z \tag{79}$$

můžeme zapsat zkráceně v maticovém tvaru:

$$Q(t) = TC = [t^3\ t^2\ t\ 1]\begin{bmatrix} a_x & a_y & a_z \\ b_x & b_y & b_z \\ c_x & c_y & c_z \\ d_x & d_y & d_z \end{bmatrix} \tag{80}$$

Derivaci $q'(t)$ získáme derivací vektoru **T**

$$q'(t) = \frac{Q(t)}{dt} = \frac{dT}{dt}C = [3t^2\ 2t\ 1\ 0] \tag{81}$$

Kubika v prostoru je určena dvanácti parametry, které tvoří prvky matice **C**. Dosáhnout změny tvaru křivky přímou změnou parametrů je obtížné. Tvarování křivek a ploch se modeluje interaktivně tak, že se parametry spojí s viditelnými prvky: řídicími body, směry a tečnými vektory apod.

Při modelování křivek i ploch je výhodné oddělit individuální charakteristiky dané křivku, od společné vlastností všech křivek shodným způsobem modelovaných.

U kubik lze matici $\boldsymbol{C}$ rozepsat do tvaru

$$C = MG \tag{82}$$

, kde matice konstant **M** je *bázová matice* typu 4 x 4 a čtyřprvkový sloupcový vektor **G** je *vektor geometrických podmínek.* Součin **TM** definuje společnou *polynomiální bázi* (skupinu polynomů). Tato báze je společná pro všechny křivky určitého typu. Vektor geometrických podmínek obsahuje parametry ovlivňující tvar křivky. Graficky to jsou řídicí body a tečné vektory. Kubika je definována vztahem

$$Q(t) = TMG = [t^3\ t^2\ t\ 1]\begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \\ m_{41} & m_{42} & m_{43} & m_{44} \end{bmatrix}\begin{bmatrix} G_1 \\ G_2 \\ G_3 \\ G_4 \end{bmatrix} \tag{83}$$

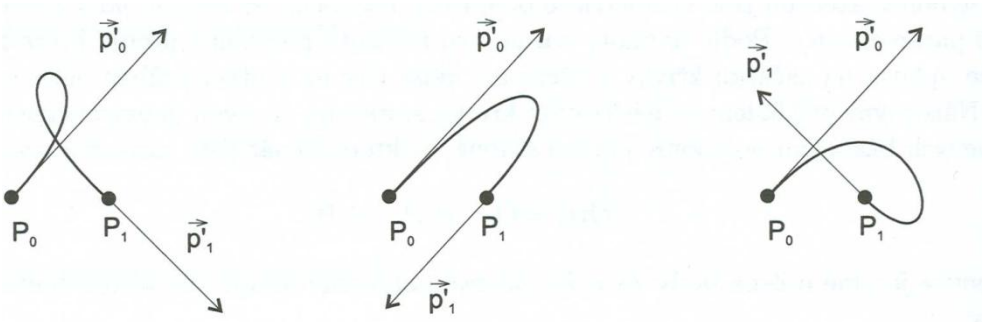
V rozepsaném tvaru nejde o nic jiného než o součet polynomů násobených geometrickými podmínkami

$$Q(t) = (m_{11}t^3 + m_{12}t^2 + m_{13}t + m_{14})G_1 + (m_{21}t^3 + m_{22}t^2 + m_{23}t + m_{24})G_2 + (m_{31}t^3 + m_{32}t^2 + m_{33}t + m_{34})G_3 + (m_{41}t^3 + m_{42}t^2 + m_{43}t + m_{44})G_4 \quad (84)$$

Polynomy, kterými jsou geometrické podmínky násobeny, se uplatní jako proměnlivé váhy řízené parametrem t. Podle hodnoty parametru t bázové polynomy určují, která z podmínek se více uplatní na začátku křivky a která má větší vliv na vnitřní průběh nebo na koncovou část. Názorným příkladem je následující křivka sestavená ze dvou geometrických podmínek násobených bázovými polynomy prvého stupně, ve které čtenář jistě rozpozná rovnici úsečky $Q(t) = (1-t).P_1 + t.P_2$. Geometrie je plně určena body P_1 a P_2, bázové polynomy určují vliv těchto bodů na průběh úsečky.

4.2.1 Hermitovské kubiky

Mezi často používané křivky patří *Hermitovské kubiky*, někdy také označované jako Fergusonovy kubiky. Tyto křivky jsou určeny dvěma řídicími body P_0, P_1 a dvěma tečnými vektory $\vec{p}_0'$ a $\vec{p}_1'$ v v těchto bodech. Body P_0 a P_1 určují polohu křivky, křivka v nich začíná a končí. Směr a velikost tečných vektorů $\vec{p}_0'$ a $\vec{p}_1'$ pak určuje míru jejího vyklenutí. Čím větší je velikost tohoto vektoru, tím více se křivka k němu přimyká. Jsou-li oba vektory nulové, pak se křivka stane úsečkou P_0P_1. Obrázek 5.7 demonstruje změnu tvaru křivky, je-li vektor $\vec{p}_0$ konstantní a vektor $\vec{p}_1$ se mění.



Obr 53. Změna tvaru Hermitovské kubiky

Předpis pro výpočet Hermitovské kubiky ve stylu (5.9) má tvar

$$F(t) = [t^3\ t^2\ t\ 1]\begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} P_0 \\ P_1 \\ \vec{p}_0' \\ \vec{p}_1' \end{bmatrix} \quad (85)$$

Rozepíšeme-li vztah (5.11) do jediné rovnice, dostaneme

$$F(t) = P_0F_1(t) + P_1F_2(t) + \vec{p}_0'F_3(t) + \vec{p}_1'F_4(t) \quad (86)$$

kde F_1, F_2, F_3, F_4 jsou tzv. kubické Hermitovské polynomy ve tvaru (87)

$$F_1(t) = 2t^3 3t^2 + 1 \quad (87)$$
$$F_2(t) - 2t^3 + 3t^2 \quad (88)$$
$$F_3 = t^3 - 2t^2 + t \quad (89)$$
$$F_4(t) = t^3 - t^2 \quad (90)$$

Dosazením $t = 0$, resp. $t = 1$ ověříme, že křivka začíná, resp. končí v bodě P_0, resp. P_1. Přednosti

Hermitovských kubik se projeví při jejich navazování. Právě tečné vektory v jejich koncových umožní velice snadno jejich hladké navázání.

Nevýhodou těchto křivek je poměrně nesnadná editace tečného vektoru ve třech rozměrech. Detailně se Hermitovským kubikám věnuje ().

4.3 Aproximační křivky

Společnou vlastností aproximačních křivek je to, že nemusí procházet body definičního polygonu. Mezi aproximační křivky patří Bézierovy křivky a z jejich rodiny pak nejčastěji Bézierovy kubiky, Coonsovy kubiky, Spline křivky, B-spline křivky a neuniformní racionální spline – NURBS.

4.3.1 Bézierovy křivky

Bézierovy křivky jsou asi nejpopulárnější aproximační křivky. Používají se jak pro modelování ve dvou rozměrech, ale vytváření trojrozměrných objektů. Používají i při definici písma *(fontů).*

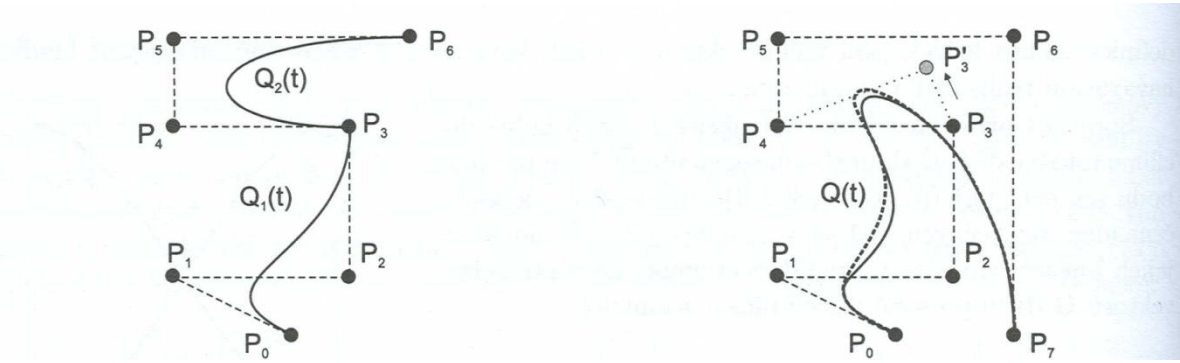
Nejčastěji používanou variantu jsou Bézierovy kubiky. Obecné Bézierovy křivky *n*-tého stupně mají tvar

$$Q(t) = \sum_{i=0}^{n} P_i B_i^n(t) \tag{91}$$

kde B_i^n jsou Bernsteinovy polynomy n-tého stupně

$$B_i^n = \binom{n}{i} t^i (1-t)^{n-i};\ t \in \langle 0,1\rangle;\ i = 0,1 \dots n \tag{92}$$

V tomto vztahu je $\binom{n}{0} = 1$ a $0^0 = 1$.



Obr 54. Bézierovy křivky 3. stupně (vlevo) a 7. stupně (vpravo)

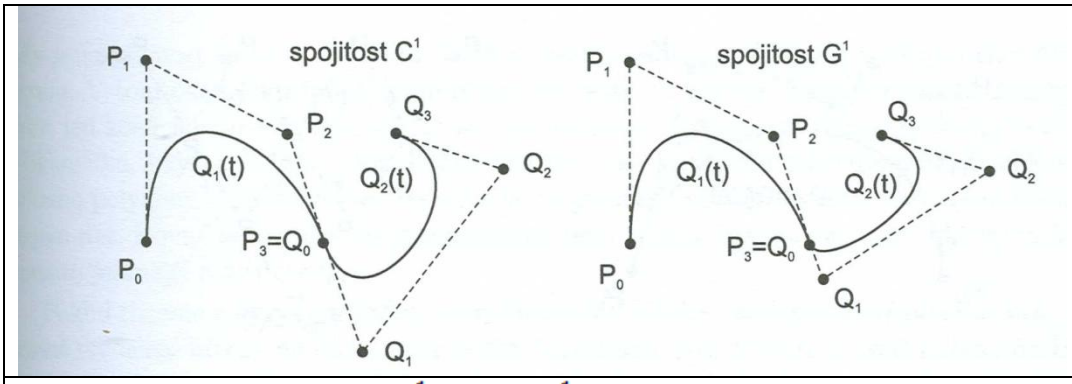
Křivka prochází prvním a posledním bodem řídicího polygonu (viz též obrázek 55). Tečné vektory v krajních bodech se určí ze vztahů:

$$\vec{q}'(0) = n(P_1 - P_0) \tag{93}$$

$$\vec{q}'(1) = n(P_n - P_{n-1}) \tag{94}$$

Bézierovy křivka celkově změní svůj tvar. Proto se většinou používají kubiky a tyto se navazují jedna na druhou.

Spojitost C^0 segmentů Bézierových křivek zajistíme shodou posledního bodu řídicího polygonu segmentu Q1 a prvního bodu segmentu. Spojitost Cl, tedy identita a nenulovost tečných vektorů je zaručena, pokud je bod $P_3 = R_0$ středem úsečky $P_{n-1}R_1$. Geometrickou spojitost zaručí to, že body P_2, P_3 a Q_1 budou v tomto pořadí ležet na jedné přímce na obrázku 55 vpravo.



Obr 55. Spojení C^1 (vlevo) a G^1 (vpravo) dvou Bézierových kubik. První segment je označen $Q_1(t)$ a je určen body P_0 až P_3, druhý je označen $Q_2(t)$ a je určen body Q_0 až Q_3.

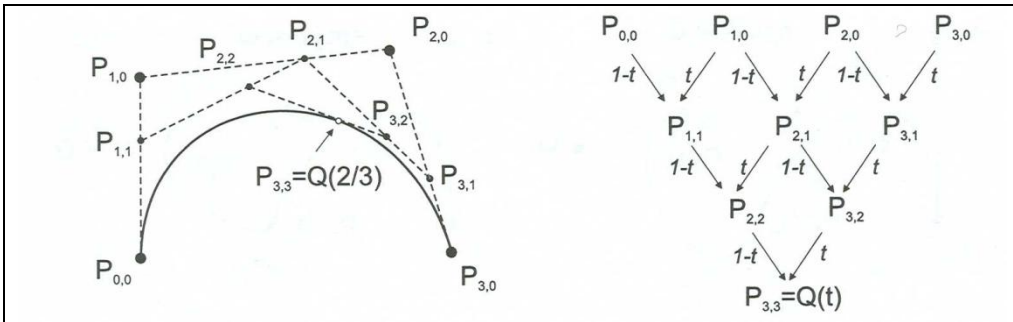
Bernsetinův polynom stupně n lze rekurentně definovat Bernsteinova lineární kombinace dvou po sobě následujících Bernsteinových polynomů stupně n - l.

Vhodnou metodou pro výpočet Bézierovy křivku stupně *n* je rekurzivní *algoritmus de Casteljau*. Bod křivky o souřadnicích $Q(t)$ vypočítáme z rekurentního vztahu

$$P_{j,i}(t) = (1-t)P_{j-1,i-1} + tP_{j,i-1} \qquad (95)$$

kde $i = 1, 2, \ldots, n$ a $j = i, i+1, \ldots, n$.Vstupem tohoto algoritmu jsou body řídicího polygonu. Za $P_{i,0}$ se dosadí po řadě vždy jeden bod P_i a rekurentní vztah (95) postupně vypočítává body $P_{i,j}$ tak, jak ukazuje obrázek 56 vpravo. V posledním kroku výpočtu se koeficient $P_{n,n(t)}$ rovná hodnotě křivky v bodě $Q(t)$. Na obrázku 56 vlevo je ukázán výpočet bodu $Q(3/4)$.

Pomocí algoritmu de Casteljau může být Beziérova křivka v libovolném v bodě určeném hodnotu parametru $t \in \langle 0,1 \rangle$ rozdělena na dvě části .



Obr 56. Algoritmus de Casteljau: Výpočet bodu (a(2/3) (vlevo) a schéma výpočtu

Bézierovy křivky leží v konvexní obálce svého řídicího polygonu. Umíte sami vysvětlit, co si představujete pod pojmem kováni obálka? Změna polohy bodu má vliv na změnu tvaru celé křivky. Proto je vhodnější skládat složitější průběhy po částech ze segmentů Bézierových kubik a v uzlových bodech mezi segmenty využít snadné kontroly jejich spojitosti. Bézierovy křivky jsou invariantní k otáčení, posunu a změně měřítka. Protože Bézierovy křivky uvedené v této části jsou neracionální, tak nejsou invariantní k perspektivnímu promítání. S využitím homogenních souřadnic a s přechodem na racionální vyjádření Bézierových křivek lze i tento nedostatek obejít.

4.3.2 Bézierovy kubiky

Nejčastěji používanými křivkami jsou Bézierovy kubiky, která je určena čtyřmi body P_0, P_1, P_2 a P_3. Vychází z prvního řídicího bodu, končí v posledním a je určena vztahem

$$Q(t) = \sum_{i=0}^{3} P_i B_i(t), \tag{96}$$

Maticový zápis Bézierovy kubiky je

$$B_0(t) = (1-t)^3 \tag{97}$$

$$B_1(t) = 3t(1-t)^2 \tag{98}$$

$$B_2(t) = 3t^2(1-t) \tag{99}$$

$$B_3 = t^3 \tag{100}$$

Tečné vektory v prvním a posledním bodě mají tvar:

$$\vec{p}'(0) = 3(P_1 - P_0) \tag{101}$$

$$\vec{p}'(1) = 3(P_3 - P_2) \tag{102}$$

Postup, jakým převedeme kubiku v Bézierově reprezentaci (čtyři řídicí body) na kubiku v Hermitovské reprezentaci (dva body a dva vektory), vychází ze vztahu pro tečné vektory.

4.3.3 Coonsovy kubiky

Coonsova kubika zaujímá vedle Bézierových křivek významné postavení v historii parametrických křivek a ploch. Je běžně používána při tvorbě po částech skládaných polynomiálních křivek.

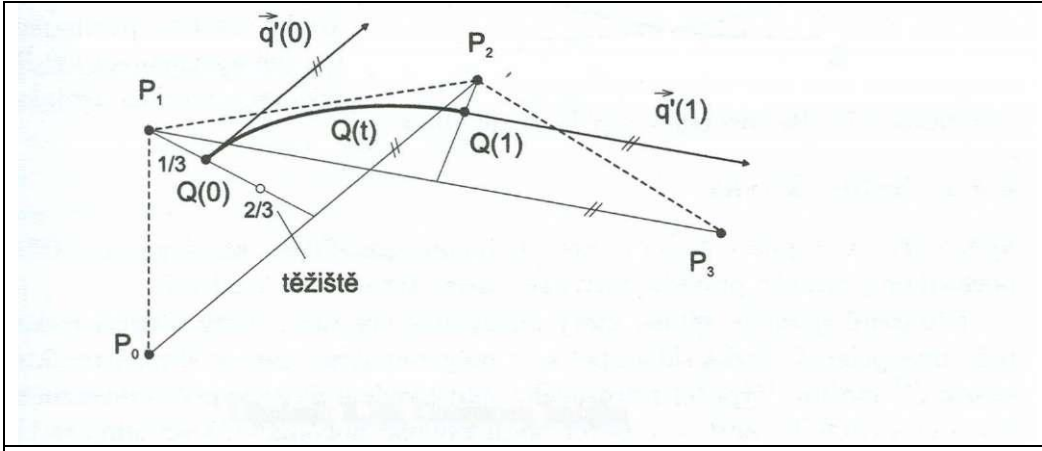
Coonsova kubika, také uniformní neracionální B-spline, zkráceně B-spline, je určena čtyřmi řídicími body P_0, P_1 P_2 a P_3 a předpisem

$$Q(t) = \frac{1}{6}[t^3\ t^2\ t\ 1]\begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix}\begin{bmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{bmatrix} \tag{103}$$

Segment křivky (na rozdíl od Bézierových křivek) obecně *neprochází* krajními body. Tato vlastnost je dána bázovými polynomy Coonsových kubik. Křivka začíná a končí v bodech:

$$Q(0) = \frac{P_0 + 4P_1 + P_2}{6} \tag{104}$$

$$Q(1) = \frac{P_1 + 4P_2 + P_3}{6} \tag{105}$$



Obr 57. Coonsovy kubiky

Geometrický význam těchto vztahů ukazuje obrázek 58. Počáteční bod segmentu $Q(0)$ (viz obr.) leží v první třetině těžnice vycházející z vrcholu P_1 trojúhelníku tvořeného body P_0, P_1 a P_2. Tomuto bodu říkáme antitěžiště. Obdobně i bod $Q(1)$ leží v antitěžišti těžnice vycházející z vrcholu P_2 trojúhelníku tvořeného body P_1, P_2 a P_3.

Pro tečné vektory v bodech $Q(0)$ a $Q(1)$ platí:

$$q'(0) = \frac{P_2 - P_0}{2} \quad (106)$$

$$q'(1) = \frac{P_3 - P_1}{2} \quad (107)$$

Násobnost bodů řídicího polygonu je důležitou vlastností Coonsových kubik. Položíme-li $P_0 = P_1$, vytvoříme *dvojnásobný* bod. Z původního trojúhelníka P_0, P_1 a P_2 zůstane úsečka P_0 a P_2. Křivka pak

začíná v jedné šestině této úsečky. Ztotožníme-li body $P_0 = P_1 = P_2$, vytvoříme úsečku z bodu P_0 a s koncovým bodem v jedné šestině úsečky P_0P_3. Coonsova kubika tedy prochází *trojnásobným* bodem.

4.3.4 Spline křivky

Spline křivka stupně *n* je po částech polynomiální křivka třídy C^n. Vlastnosti těchto křivek mají

napodobovat křivítko – pružné pravítko, kterému se anglicky říká spline.

Přirozený spline interpoluje své řídicí body. Přirozený kubický spline je interpolační křivka, která se skládá z polynomiálních křivek stupně tři, C^2 spojitá ve svých uzlech. Nevýhodou těchto křivek je to, že změnou polohy jediného definičního bodu se změní tvar celé křivky.

V počítačové grafice jsou nejčastěji používané *B-spline kubiky*. Jedná o aproximační křivky a tak tyto *B-spline* křivky nejsou přirozenými spline křivkami. Nejjednodušší jsou uniformní neracionální B-spline. Zobecněním pak jsou neuniformní racionální B-spline, neboli *NURBS*.

Důležitými vlastnostmi B-spline křivek jsou:

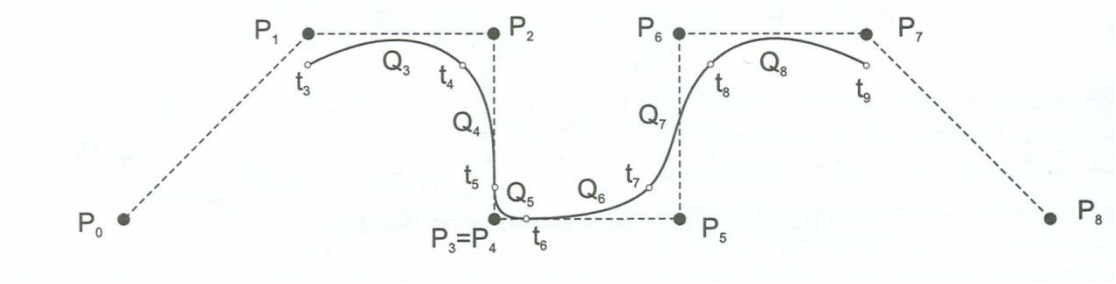
- invariance vůči otáčení, posunutí a změně měřítka,
- B-spline křivka leží celá ve své konvexní obálce,
- její segmenty leží v konvexních obálkách svých řídicích polygonů.

Obecně kubická spline neprochází krajními body svého řídicího polygonu. Pokud chceme zajistit, aby B-spline procházela krajními body polygonu, využijeme další vlastnosti kubiky, a to možnost tvorby násobných bodů. Křivku v tomto případě zadáme posloupností bodů $P_0, P_0, P_0, P_1, \ldots, P_{n-1}, P_n, P_n, P_n$.

První segment bude tvořen úsečkou spojující bod P_0 s bodem ležícím v jedné šestině úsečky $P_0\,P_1$.

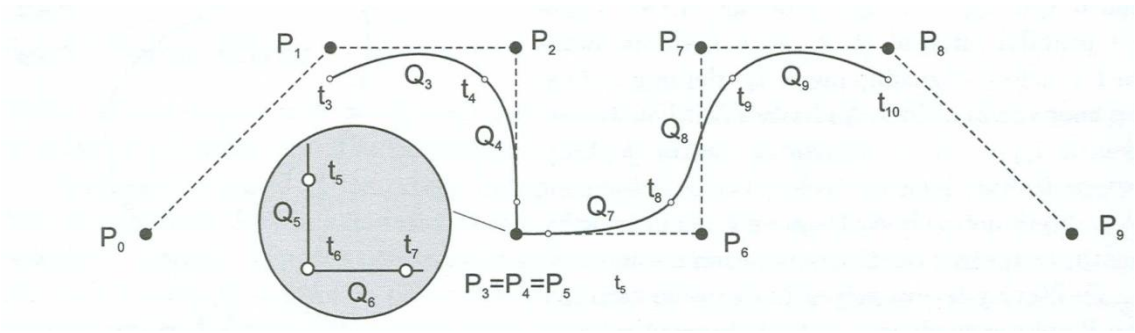
Totéž platí pro poslední segment na konci řídicího polygonu.

Segment křivky leží v konvexní obálce svého řídicího polygonu. Této vlastnosti využijeme pro změnu tvaru segmentu tak, že jednoduché řídicí body znásobíme. Na výchozím obrázku (59) jsou všechny řídicí body jednoduché. Na následujících obrázcích jsou zachyceny změny, když zvyšujeme násobnost bodu P_3.



Obr 58. Dvojnásobný bod

Segment Q_4 je určen uzly t_4 a t_5 a řídicím polygonem P_1, P_2, P_3, P_4 a segment Q_5 uzly t_5 a t_6 a řídicím polygonem P_2, P_3, P_4, P_5. V případě, kdy není žádný bod několikanásobný, jsou všechny řídicí polygony čtyřúhelníky. Díky společným řídicím bodům mají konvexní obálky dvou po sobě následujících segmentů



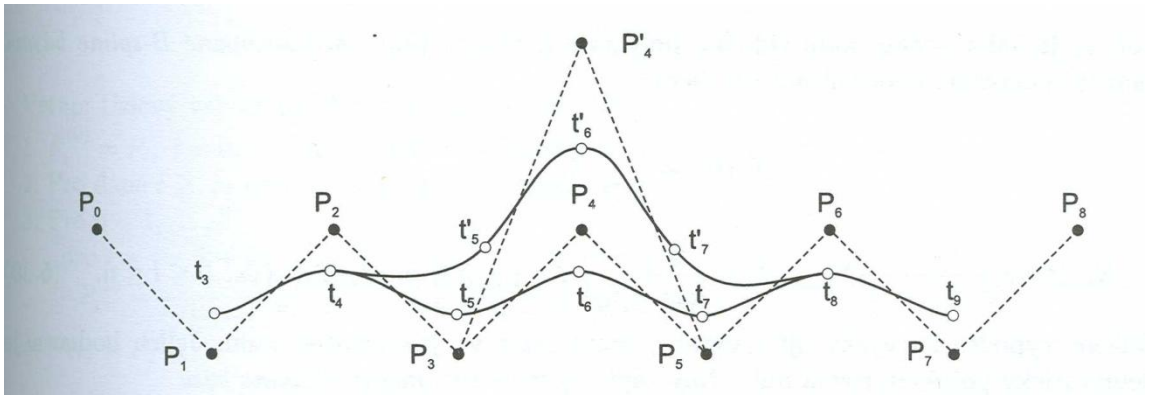
Obr 59. Trojnásobný bod

(pro náš případ řídicí polygony segmentů Q_4 a Q_5 mají společnou plochu vymezenou trojúhelníkem P_2, P_3, P_4. V této oblasti také leží společný uzel t_5. Zdvojením bodu $P\_3 = P\_4$ na obrázku 60

čtyřúhelníky P_1, P_2, P_3, P_4 *a* P_2, P_3, P_4, P_5 se transformují na trojúhelníky. Společnou oblastí je hrana P_2P_3. Zde musí ležet i uzel t_5. Na obrázku 5.21 je bod P_3 trojnásobným bodem. Konvexní obálku P_2 až P_5 tvoří pouze úsečka P_2P_3, segment Q_5 degeneruje na úsečku P_2P_3.

Důležitým pojmem je *lokalita změny:* pokud změníme polohu některého z řídicích bodů, křivka změní tvar jen části určenou tímto bodem. Změníme-li polohu bodu P_i kubiky $Q(t)$, pak změníme tvar čtyř segmentů Q_i, Q_i+1, Q_i+2 a Q_i+3. Tuto skutečnost zachycuje obrázek 5.22. Zde jsou zobrazeny dvě křivky. První je určena řídicím polygonem P_0, P_1, . . ., P_8 Uzly jsou označeny t_3 až t_9. U druhé křivky je bod P_4 je posunut do polohy P'. Vidíme, že změna polohy jediného bodu změnila také polohu uzlů t_5 *až* t_7. Došlo zde ke změně tvaru čtyř segmentů. To proto, že vždy dva uzly určují hranice segmentu,.

Obrázek 61: Lokalita změny tvaru křivky při změně polohy bodu řídicího polygonu. Uzlové vektory jsou označeny prázdnými kolečky.



Obr 60. NURBS

Neuniformní racionální B-spline křivky (NURBS - *non uniform rational B-spline)* jsou dvojím zobecněním B-spline křivek uvedených v předcházející části. Termín *neuniformní* říká, že vzdálenosti uzlů, ve smyslu parametru t, nemusí být u těchto křivek konstantní. Body jsou u těchto křivek reprezentovány *homogenními souřadnicemi* a tuto skutečnost postihuje termín *Racionalita* .

Křivka NURBS je určena:

- $n + 1$ body P_i; $i = 0, \ldots, n$ řídicího polygonu,
- řádem B-spline k (nejvyšší stupeň polynomu je $k - 1$) a
- uzlovým vektorem $\boldsymbol{u}$ délky $n + k + 1$ a . Neklesající posloupnost reálných čísel - uzlových hodnot $t_0 \leq t_1 \leq \ldots \leq t_{n+k}$ tvoří uzlový vektor.

Uzlový vektoru s uniformním rozložením může mít tvar $\mathbf{U}_1 = \{-2, -1, 0, 1, 2, 3, 4, 5\}$ Vektor $\mathbf{U}_2 = \{0, 0, 0, 1, 1, 1\}$ je okrajový uzlový vektor s opakovanými uzlovými hodnotami.

Křivka NURBS je dána vztahem

$$Q(t) = \frac{\sum_{i=0}^{n} w_i \, P_i N_{i,k}(t)}{\sum_{i=0}^{n} w_i N_{i,k}(t)} \tag{108}$$

kde w_i je váha i-tého bodu řídicího polygonu a $N_{i,k}(t)$ jsou *normalizované B-spline bázové funkce.*

Tyto funkce jsou definované rekurentním vztahem

$$N_{i,1}(t) = \begin{cases} 1 \; pro \; t_i \leq t < t_{i+1} \\ 0 \; jinde \end{cases}$$
$$N_{i,k}(t) = \frac{t - t_i}{t_{i+k-1} - t_i} N_{i,k-1}(t) + \frac{t_{i+k} - t}{t_{i+k} - t_{i+1}} N_{i+1,k-1}(t) \tag{109}$$
$$pro \; t_i < t_{i+1+k}, 0 \leq i \leq n$$

Během výpočtu se vyskytují i výrazy, které mají ve jmenovateli nulu. Jejich hodnota je definitoricky položena rovna nule. Jiný zápis využívá *racionální B-spline báze*

$$R(i,k) = \frac{w_i N_{i,k}(t)}{\sum_{j=0}^{n} P_i R_{i,k}(t).} \tag{110}$$

Křivku NURBS podle rovnice (5.29) lze potom zapsat jednodušeji

$$Q(t) = \sum_{i-0}^{n} P_i R_{i,k}(t). \tag{111}$$

Jedním z nejdůležitějších modelovacích prostředků, který poskytují NURBS křivky, je možnost vložení nových uzlových bodů. Namísto zvyšování stupně polynomu umožňují nově vložené uzlové body společně s příslušně pozměněnými řídicími body vymezit ty části křivky, které chceme lokálně ovlivnit.

Křivky NURBS mají tyto vlastnosti:

1. Okrajový uzlový vektor

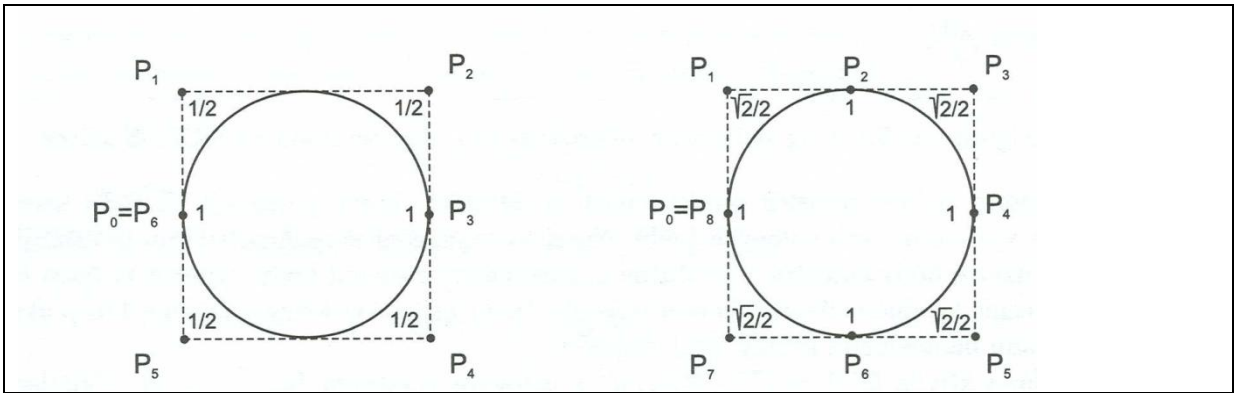
$$\mathbf{U} = \left\{ \underbrace{\alpha, \alpha, \ldots, \alpha}_{k}, \ \underbrace{t_k, \ldots, t_{n-k}}_{n+1-k}, \ \underbrace{\beta, \beta, \ldots, \beta}_{k} \right\}. \tag{112}$$

zajistí, že křivky procházejí prvním a posledním bodem řídicího polygonu.

2. Celá křivka i její jednotlivé segmenty leží v konvexní obálce svých řídicích polygonů. Změna váhy či polohy jednoho bodu má vliv pouze na část křivky.
3. Jsou invariantní vůči transformacím a vůči rovnoběžnému a středovému promítání.
4. Pomocí váhových koeficientů umožňují přesně vyjádřit kuželosečky jako podíl polynomů.
5. Pro uzlový vektor

$$\mathbf{U} = \left\{ \underbrace{0, 0, \ldots, 0}_{k}, \ \underbrace{1, 1, \ldots, 1}_{k} \right\} \tag{113}$$

jsou normalizované B-spline báze rovny Bernsteinovým polynomům stupně k a NURBS je racionální Bézierovou křivkou. Pro hodnoty w_i= 1 je NURBS Bézierovou křivkou tak.



Obr 61. Kružnice jako NURBS: sedm bodů (vlevo) a devět bodů (vpravo)

Zejména čtvrtá vlastnost je důležitá pro tvorbu jednotných datových struktur, které reprezentují jak „klasické" tvary, jako jsou kružnice, elipsa či čtverec, tak volné tvary *(free-form).* Nevýhodou je, že reprezentace „klasických" objektů není jednoduchá. Tak například kružnice může být reprezentována sedmi body s vahami [1, 1/2, 1/2, 1, 1/2, 1/2, 1] (obrázek 5.23 vlevo) a uzlovým vektorem $U = \{0, 0, 0, 1/4, 1/2, 1/2, 3/4, 1, 1, 1\}$, nebo devíti body (čtyřmi 90° kruhovými oblouky) s vahami

$U = \left\{1, \frac{\sqrt{2}}{2,1}, \frac{\sqrt{2}}{2,1}, \frac{\sqrt{2}}{2,1}, \frac{\sqrt{2}}{2,1}\right\}$ (obrázek 5.23 vpravo) a uzlovým vektorem $U = \{0,0,0,1,1,2,2,3,3,4,4,4\}$.

Shrnutí pojmů

Matematický popis křivek je založen na kombinaci polynomů různých řádů.

Nejjednodušší křivky jsou interpolační křivky určené body a polynomem jistého stupně.

- Požadavky kladené na popis křivek jsou: jsou dostatečně tvarovatelné, jejich výpočet je nenáročný, lze s nimi snadno manipulovat, je u nich možné zaručit spojitost C2
- Křivku určujeme řídicími body či vektory.
- Aproximační křivky tvoří základní třídu křivek v počítačové grafice.
- Bézierovy křivky jsou asi nejpopulárnější aproximační křivky. Používají se jak pro modelování ve dvou rozměrech, ale vytváření trojrozměrných objektů. Používají i při definici písma *(fontů).*
- Obecné Bézierovy křivky *n*-tého stupně mají tvar $Q(t) = \sum_{i=0}^{n} P_i B_i^n(t)$
- Vhodným algoritmem pro vykreslení Bézierovy křivky je algoritmus de Casteljau.
- K nejčastěji využívaným křivkám patří Bézierovy kubiky
- Dalším druhem často využívaných křivek jsou Coonsovy kubiky. Segment křivky (na rozdíl od Bézierových křivek) obecně *neprochází* krajními body.

Spline křivka stupně *n* je po částech polynomiální křivka třídy C^n. Vlastnosti těchto křivek mají napodobovat křivítko – pružné pravítko, kterému se anglicky říká spline.

Obecně kubická spline neprochází krajními body svého řídicího polygonu.

Křivka NURBS je zobecněním B-spline křivky.

Otázky

5. Popište interpolační křivky.
6. Které hlavní aproximační křivky znáte?
7. Jaké jsou hlavní požadavky kladení na zadání křivek.
8. Jaký druh křivek se nejčastěji používá.

9. Kdy se začaly používat interaktivně zadávané křivky. Znáte počítačové programy, kde se takto zadané křivky využívají.
10. V čem spočívá výhoda Bézierovy křivky
11. Co určuje B-spline křivku.
12. Jak byste charakterizoval Coonsovy kubiky.
13. Co jsou to Hermitovské kubiky?

5 PLOCHY V POČÍTAČOVÉ GRAFICE

Čas ke studiu: 1 hodina

Cíl Po prostudování tohoto odstavce budete umět

- Po nastudování kapitoly bude student chopen klasifikovat nejpoužívanější typy ploch používaných v počítačové grafice.
- Bude umět vybrat, pro kterou oblast počítačové grafiky je ta která plocha vhodná. Získá přehled o matematické teorii popisující tyto plochy a pude schopen implementace získaných poznatků v širších souvislostech.

Výklad

Podobně jako tomu bylo u křivek je i pro plochy nutné mít k dispozici jejich matematický popis, abychom byli schopni s takovou plochou manipulovat. Těmito manipulacemi rozumíme posuny otáčení, změny měřítka a různé projekce těchto ploch. Vždy se pro popis ploch využijí jen určité klíčové body plochy nebo nějakého vhodného řídicího útvaru.

V systémech CAD, což jsou systémy počítačové podpory projekce, kde je uplatnění křivek a ploch základní úlohou, je možné těmito plochami neprojektovat rozličné tvary s komfortem, který známe u kreslení křivek. Matematický popis těchto rovinných útvarů umožní systému lehký výpočet ploch a objemů těles, která jsou těmito plochami vymezena. V dalším textu se, díky rozsáhlosti dané problematiky, zaměříme jen na vybrané aspekty tohoto tématu a rovněž jen na ty nejpoužívanější plochy počítačové grafiky. Berte tudíž následující text jen jako lehký úvod do jinak široké oblasti počítačových ploch.

5.1 Plochy dané analytickým popisem

Analytický popis ploch je v základě obdobný jako popis křivek ve 2D prostoru. Obdobě i zde, jakmile máme k dispozici matematický popis plochy, je již poměrně snadný úkol tuto plochu vykreslit.

Nejčastěji se využívá parametrická vyjádření plochy ve tvaru tří rovnic se shodnou nezávislou proměnnou, parametrem t. Popis v obecném tvaru vypadá takto:

$$\left.\begin{aligned} x &= f(x) \\ y &= g(y) \\ z &= h(z) \end{aligned}\right\}, t \in \langle t_0, t_1 \rangle \qquad (114)$$

Plochu již jednoduše vykreslíme tak, že za parametr t dosadíme množinu hodnot z použitého intervalu $\langle t_0, t_1 \rangle$. Ke každé hodnotě parametru t získáme souřadnice vykreslovaného bodu ve třírozměrném

prostoru. Získáme tím stejně obsáhlou množinu bodů, zadaných svými souřadnicemi, které nazýváme uzlové body dané plochy.

Druhým krokem, který musíme provést je převést 3D souřadnice na vhodné 2D souřadnice. Vypočteme tedy průmět tohoto bodu do plochy. Pro napojení již stačí provést vhodnou interpolaci s pozicí předchozího uzlového bodu. I když se pohybujeme ve třírozměrném prostoru, ne vždy musíme řešit otázku viditelnosti. Otázku viditelnosti lze řešit také vykreslování plochy směrem odzadu dopředu.

5.2 Coonsovy plochy

Coonsovy plochy patří mezi evergreeny počítačové grafiky. Jejich dlouhodobá oblíbenost je dána jednoduchou teorií, nejsou zvlněny (mají minimum konvexních a konkávních částí, zjednodušeně řešeny hrbů a údolí. Jejich implementace na počítači není příliš obtížná. Coonsovy plochy v praxi tvoří širokou rodinu ploch. Zde uvedeme jen ty nejdůležitější z nich.

5.2.1 Bilineární Coonsova plocha

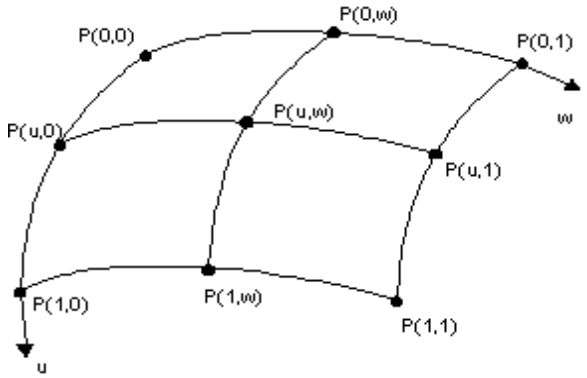
Bilineární Coonsova ploch patří k nejjednodušším typům ploch v počítačové grafice.. Základní rovnicí popisující tuto plochu je rovnice

$$[1-u,-1,u] \times \mathbf{M} \times [1-w,-1,w]^T. \tag{115}$$

Zde u a w jsou parametry, protože úsek plochy, říkáme mu plát, zde mu přisoudíme jméno bilineární plát. Slovíčko bilineární nám říká, že výsledek je lineárně závislý na každém z parametrů, pokud je druhý konstantní. Vlastní plát je určen svým okrajem $P(u,0)$, $P(u,1)$, $P(0,w)$ a $P(1,w)$

Matice $\mathbf{M}$ je čtvercová matice polohových vektorů. Prvky této matice odpovídají odpovídajících uzlových bodů plochy.

$$\boldsymbol{M} = \begin{bmatrix} \boldsymbol{P(0,0)} & \boldsymbol{P(0,w)} & \boldsymbol{P(0,1)} \\ \boldsymbol{P(u,0)} & \boldsymbol{P(u,w)} & \boldsymbol{P(u,1)} \\ \boldsymbol{P(1,0)} & \boldsymbol{P(1,w)} & \boldsymbol{P(1,1)} \end{bmatrix} \tag{116}$$



Obr 62. Bilineární Coonsova plocha

Uprostřed matice leží bod $P(u,w)$. Jeho určuje bod na ploše pro zvolené hodnoty parametrů u a w.

Polohový vektor je řešením implicitní rovnice. Lépe se bude řešit explicitní tvar této rovnice, který můžeme zapsat ve tvaru

$$\begin{aligned} P(u,w) = {} & [1-u,u] \times [P(0,w),P(1,w)]^T + [P(u,0),P(u,1)] \times [1-w,w]^T \\ & - [1-u,u] \times Q \times [1-w,w]. \end{aligned} \tag{117}$$

Matice $\mathbf{Q}$ je matice

$$Q = \begin{bmatrix} P(0,0) & P(0,1) \\ P(1,0) & P(1,1) \end{bmatrix}. \tag{118}$$

Budou- li protilehlé strany plátu úsečky, výsledná plocha bude přímková. Obecná bilineární plocha je obecnějším případem přímkové plochy.

5.2.2 Bikubická Coonsova plocha

Druhým velmi rozšířeným typem Coonsových ploch je plocha bikubická. Představuje rozšíření výše popsané plochy kubické. Tvar a zadávání jejích parametrů je shodné s bilineární plochou. Základní explicitní rovnice je modifikací rovnice bilineární Coonsovy plochy:

$$[F_1(u), -1, F_1(u)] \times M \times [F_1(w), -1, F_2(w)]^T = 0 \tag{119}$$

Matice $\mathbf{M}$ je shodná matice jako u bilineární plochy, funkce F_1 a F_2 jsou Fergusnovy polynomy, jak jsme je již poznali u Fergusonových ploch.

$$\begin{aligned} F_1(t) &= 1t^3 + 3t^2 + 1 \\ F_2(t) &= -2t^3 + 3t^2 \\ F_3(f) &= -1 \end{aligned} \tag{120}$$

Jak vidíte, v implicitním popisu plochy jsme funkce F_3 zapsali přímo.

Pro účely programování ji opět převedeme do explicitního tvaru a získáme obdobou rovnici, jako při výpočtu bilineární plochy.

Při modelování větších celků postupujeme tak, že spojujeme jednotlivé pláty do společného většího plošného útvaru. Je žádoucí, aby přechody mezi pláty byly spojité. Základní spojitosti C_0 lze dosáhnou jednoduše tím, že hraniční křivky dvou navazujících plátů budou shodné. Tím jme, ale nezaručili hladkost přechodu v kolmém směru na tuto společnou hraniční přímku. Z předchozí kapitoly víme, že spojitosti vyššího stupně dosáhneme, pokud se rovnají i společní tečné vektory. Tato podmínka se ale pro tyto plochy splňuje jen obtížně, proto se nepoužívají pro hladké spojení plátů.

5.2.3 Všeobecná Coonsova plocha

Zmínili jsme se, že předchozí typ Coonsovy plochy není vhodný pro plátování. Také jsme se zmínili, že požadavek hladkosti plochy dosáhneme, pokud jsou shodné tečné vektory plochy v místě společné

hraniční křivky dvou plátů. K dosažení hladkého přechodu bude postačovat shodnost tečných vektorů výslední plochy, které jsou kolmé v daném vyšetřovaném bodě na hraniční křivku. Takovou vlastnost má obecná Coonsova plocha.

Musíme tedy zadat nejen hraniční křivky, ale i příčné derivace podél okraje plátu a shodou smíšených derivací ve společných rozích plátů.

Rovnice obecné Coonsovy plochy je

$$U \times C \times W = 0 \tag{121}$$

Jednotlivé matice této explicitní maticové rovnice jsou

$$\boldsymbol{U} = [\boldsymbol{A}(\boldsymbol{u}), -\mathbf{1}, \boldsymbol{B}(\boldsymbol{u}), \boldsymbol{C}(\boldsymbol{u}), \boldsymbol{D}(\boldsymbol{u})] \tag{122}$$

$$\boldsymbol{W} = [\boldsymbol{A}(\boldsymbol{w}), -\mathbf{1}, \boldsymbol{B}(\boldsymbol{w}), \boldsymbol{C}(\boldsymbol{w}), \boldsymbol{D}(\boldsymbol{w})] \tag{123}$$

Matice $\mathbf{C}$ má tvar

$$C = \begin{bmatrix} P(0,0) & P(0,w) & P(0,1) & P_w(0,0) & P_w(0,1) \\ P(u,0) & P(u,w) & P(u,1) & P_w(u,0) & P_w(u,1) \\ P(1,0) & P(1,w) & P(1,1) & P_w(1,0) & P_w(1,1) \\ P_u(0,0) & P_u(0,w) & P_u(0,1) & P_{uw}(0,0) & P_{uw}(0,1) \\ P_u(1,0) & P_u(1,w) & P_u(1,1) & P_{uw}(1,0) & P_{uw}(1,1) \end{bmatrix} \tag{124}$$

Matici $\mathbf{C}$ můžeme rozdělit shora dolů na nás již známou matici M, část tečných vektorů ve směru w a část tečných vektorů ve směru u. Zbývají, mám v levém dolním rohu čtyři prvky matice. Ty tvoří tečné vektoru v rozích plátu, které vyjadřují tzv. krut plátu, říká se jim zkrutné vektory.

Prvek $P_w(0,0)$ je tečný vektor v bodě $[0, 0]$ ke křivce $P(0, w)$. $P_u(0,0)$ je tečný vektor v bodě $[0, 0]$ ke křivce $P(u, 0)$. $P_{wu}(0,0)$ je zkrut (twist), vektor určený smíšenou derivací v bodě $[0, 0]$.

Obdobně budeme rozumět i významu analogických vektorů ve zbylých bodech.

5.3 Beziérovy plochy

Již jsme se zmínili o Beziérových křivkách. Obdobě navrhl i řešení ploch. V šedesátých letech minulého století byly velmi rozšířeny Coonsovy plochy, které se definovaly matematickou rovnicí. Tento popis přestal vyhovovat nástupem interaktivních projektových nástrojů. Obdobně jako u Beziérových křivek, i zde plochu určíme polohou bodů řídicí sítě. Vzpomenete si, jak se toto zadání určovalo u Beziérových křivek?

Plocha definovaná takovou sítí prochází jejími rohovými body. Okrajové křivky se v rozích dotýkají rohových stran sítě. Ostatní body řídicí sítě plocha neobsahuje. Svůj tvar ale přizpůsobuje poloze těchto bodů. Pokud řídicí bod oddalujeme od plochy, plocha se za ním vyboulí. Grafik či inženýr pak posunutím těchto bodů dosáhne kýžený tvar plochy.

5.3.1 Beziérova bikubická plocha

Základní Beziérova plocha se nazývá Beziérova bikubická plocha. Její popis je maticí 4x4 bodů. Plocha je určena šestnácti uzly řídicí sítě. Tyto uzly označme B_{ij} kde indexy i a j nabývají hodnot 0,

1, 2 a 3. Plochu definujeme explicitní maticovou rovnicí tvaru

$$z = [E(u), F(u), G(u), H(u)] \times B \times [E(v), F(v), G(v), H(v)], \tag{125}$$

kde $E(u), F(u), G(u), H(u)$jsou kubické polynomy

$$E(t) = (1-t)^3 \tag{126}$$

$$F(t) = 3t(1-t)^2 \tag{127}$$

$$G(t) = 3t^2(1-t) \tag{128}$$

$$H(t) = t^3 \tag{129}$$

Matice B je matice kót vrcholů řídicí sítě

$$B = \begin{bmatrix} b_{00} & b_{01} & b_{02} & b_{03} \\ b_{10} & b_{11} & b_{12} & b_{13} \\ b_{20} & b_{21} & b_{22} & b_{23} \\ b_{30} & b_{31} & b_{32} & b_{33} \end{bmatrix} \tag{130}$$

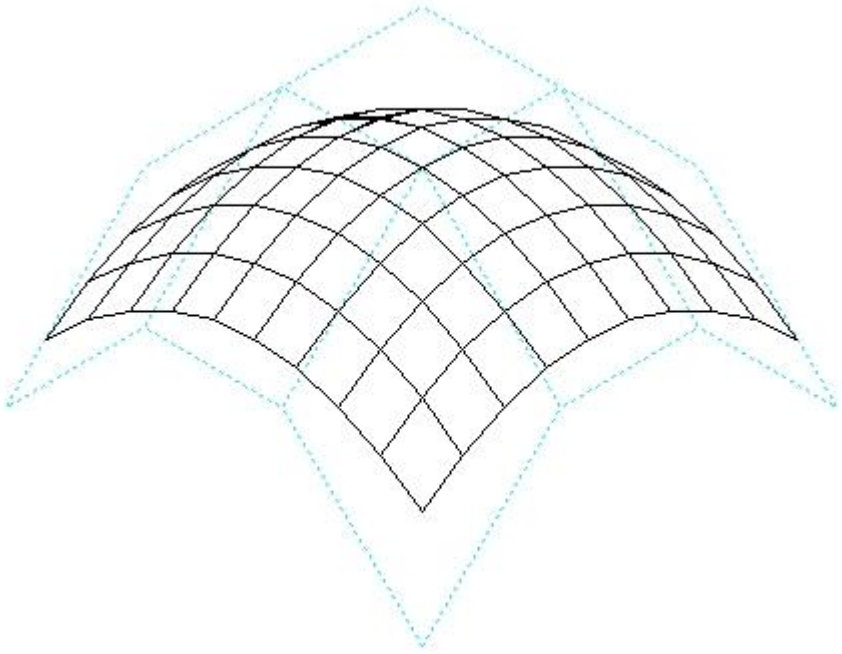
Výhodou použití takto definovaných ploch je, že můžeme měnit tvar plochy bez změny jejího okraje (změnou b_{11}, b_{12} , b_{21} a b_{22}. Při navazovaní jednotlivých ploch (resp. plátů) dosáhneme spojité avšak

ne hladké navázaní, pokus jsou krajní body jednoho plátu identické s krajními body druhého plátu. Hladkého spojení ale dosáhneme jen tehdy, pokud jsou společné vždy poslední dva body jednoho plátu a prvé dva body plátu, která připojujeme. Hladké spojení dosáhneme navíc identitou tečen na

okraji plátu. Identita tečných vektorů je zaručena, pakliže jsou předposlední body obou sousedních plátů symetrické s okrajem.

Vezmeme-li popis ve tvaru matice $\mathbf{B}$ dvou plátů. Levý plát má společnou svou pravou hranu a pravý

plát analogicky svou pravou stranu. Prvky posledního sloupce matice B levého plátu jsou tedy shodné s prvky prvého sloupce matice pravého plátu. Symetrie druhých bodů znamená, že body na okraji plátu leží ve středu úseček, jejichž krajní body jsou bod, jehož souřadnice jsou určeny předposledním prvkem jedné a druhým prvkem druhé matice shodného řádku. Pokud bychom spojovali pláty ve směru svislém, pak totéž platí analogicky pro předposlední řádek prvé a druhý řádek druhé matice.



Obr 63. Beziérová bikubická plocha

Shrnutí pojmů

- Plochy v počítačové grafice se využívají v systémech CAD tj. v návrhových systémech.
- Podle toho, jak jsou plochy zadané, rozdělujeme je :na plochy dané analytickým popisem, interpolační plochy a aproximační, resp. interaktivně vytvářené plochy.
- Pokud je plochu daný analytický předpis, je známa jej rovnice, je lehké ji vykreslit graficky. Nejčastěji se používá parametrické vyjádření plochy.
- Mezi základní plochy používané v počítačové grafice patří přímkové (pravítkové) plochy.
- Všeobecnějšími plochami používanými v počítačové grafice jsou Coonsovy plochy. Nejjednodušší je Coonsova bilineární plocha. Je definovaná maticí 3x3 řídicích bodů. Rozšířením bilineární plochy je bikubická Coonsova plocha.

- Základním problémem Coonsových ploch tohoto typu je velmi obtížné vyjádření přímých tečných vektorů. Jen obtížné se dosáhne hladké spojení dvou Coonsových ploch.
- K základním interaktivně modelovaným plochám patří Beziérova bikubická plocha. Tato je definovaná maticí 4x4 řídicích bodů a oproti výše popsaným Coonsovými plochám umožňuje hladké navazování ploch.

Otázky

14. Charakterizujte plochy používané v počítačové grafice
15. Charakterizujte a popište Coonsovu bilineární plochu
16. Charakterizujte a popište Coonsovu bikubickou a všeobecnou plochu
17. Charakterizujte a popište Beziérovu bikubickou plochu

Další zdroje

Bibliografie

[1] Dalibor, M. (2007). *Matematické principy grafických systémů*. Brno: Littera.

[2] Žára, J., & all., a. (2004). *Moderní počítačová grafika*. Praha: COMPUTER PRESS.

[3] Preparata F. P., Shamos M. I. (1985): *Computational geometry*. Springer-Verlag, New York, USA.

[4] Pelikán J. (1992): PC - *Prostorové modelování*. Grada, Praha.

[5] Farin G. , Hoschek J., Kim M. (2002): *Handbook of Computer Aided Geometric Design*, Elsevier.

[6] ŽÁRA, J. a J. SOCHOR. *Algoritmy počítačové grafiky*. Praha: ČVUT, 1993.

[7] SLAVÍK, P. *Metody zpracování grafické informace*. Praha: ČVUT, 1992.

[8] SOBOTA, B. *Počítačová grafika a jazyk C*. České Budějovice: KOOP, 1995.

[9] SKÁLA, V. *Světlo, barvy a barevné systémy v počítačové grafice*. Praha: ČVUT, 1993.

[10] DRDLA, J. *Metody modelování křivek a ploch v počítačové geometrii*. Olomouc: UP, 1992.

[11] DRS, L. *Plochy ve výpočetní technice*. Praha: ČVUT, 1984.

[12] DRS, L. - JEŽEK, F. - NOVÁK, J. *Počítačová grafika*. Praha: ČVUT, 1995.

[13] GRANÁT, L. - SELECHOVSKÝ, H. *Počítačová grafika*. Praha: ČVUT, 1980.

[14] POLÁČEK, J. , JEŽEK, G. , KOPINCOVÁ, E. *Počítačová grafika*. Praha, 1991.

[15] EGERTON, P. A. , HALL, W. S. : *Computer Graphics – Mathematical first steps*. Pearson Education, 1999.

[16] R. Hartley, A. Zisserman*: Multilple View Geometry in Computer Vision*, Cambridge University Press, 2004

[17] O. Faugeras, Q. Luong: *The Geometry of Multiple Images*, MIT Press, England,2001

[18] G. Farin, J. Hoschek, M. Kim: *Handbook of Computer Aided Geometric Design*, Elsevier, 2002

[19] M. Lávička: KMA/G1, *Geometrie 1, pomocný učební text*, ZČU Plzeň, 2006, [online], dostupné z http://home.zcu.cz/~lavicka , citováno 21. 1. 2012

[20] M. Lávička: KMA/G2, *Geometrie 2, pomocný učební text*, ZČU Plzeň, 2006, [online], dostupné z http://home.zcu.cz/~lavicka, citováno 21. 1. 2012

[21] F. Ježek: Diferenciální geometrie 1.díl, 2.díl, pomocný učební text, ZČU Plzeň, 2006

Rejstřík