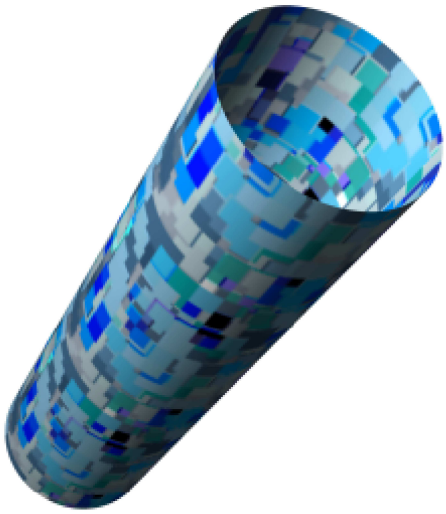
PRINCÍPY ZOBRAZOVANIA 3D OBJEKTOV



Peter Škvarenina
2002

OBSAH

Úvod

Táto kniha vznikla ako pokus podeliť sa o skúsenosti a nápady, ktoré vznikli pri programovaní rozsiahlejšieho systému na zobrazovanie 3D polygonálnych objektov. Jeho úmyslom bolo tiež priblížiť ostatným ľuďom zážitok z tvorby vlastného podobného systému, nevynímajúc úplnych začiatočníkov na poli počítačovej grafiky.

Kniha je rozčlenená na časti, z ktorých každá sa zaoberá základnými oblasťami počítačovej grafiky, ktoré sú zoradené tak, aby čo najviac na seba nadväzovali. Každá z uvedených častí je rozoberaná omnoho podrobnejšie v odbornej literatúre, ktorá však nemusí byť dostupná širokému okruhu ľudí, preto sú niektoré špecifické časti opísané podrobnejšie, kladúc väčšie nároky na matematické myslenie čitateľa.

Štýl väčšej časti informácií v knihe je prispôsobený práve čitateľom, ktorí nie sú matematicky vzdelaní na vysokej úrovni, a preto podstatná väčšina vzťahov je odvodená a popísaná voľnejším, neformálnejším štýlom.

Ďalším cieľom knihy je objasniť používateľom rôznych grafických štandardov princípy, na ktorých sú tieto štandardy postavené, aby čitateľ mohol prejsť od prístupu k štandartu ako k čiernej skrinke s určitými funkciami, k plnému uvedomovaniu si matematickej podstaty funkcionality používaných funkcií, až k spolupráci na tvorbe podobných štandardov.

Kniha sa taktiež venuje niektorým prehliadaným problémom zobrazovania, ako sú chyby najpoužívanejšieho typu perspektívnej projekcie.

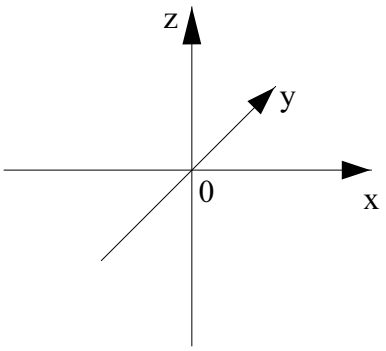
Dúfam, že kniha bude motivovať najmä k vlastnej tvorbe a ďalšiemu rozširovaniu množstva nápadov a tvorivosti, ktoré sú typické pre oblasť programovania.

Súradnicové systémy

Pri opise trojrozmerného priestoru si na začiatku musíme zvoliť súradnicový systém, s ktorým budeme pracovať. Zároveň musíme vedieť, ako môžeme transformovať vzájomne medzi sebou rôzne súradnicové systémy.

Zvyčajne kreslíme súradnicový systém tromi úsečkami zakončenými šípkou, určujúcou smer nárastu danej súradnice. Tieto úsečky predstavujú pre nás bázické vektory priestoru, z ktorých vytvoríme súradnicové osi (každý bázický vektor určuje jednu súradnicu bodu vyjadreného v danej báze). Vo všeobecnosti nám báza rozdeľuje vektorové priestory na dva typy – ľavotočivý a pravotočivý.

Pravotočivý súradnicový systém

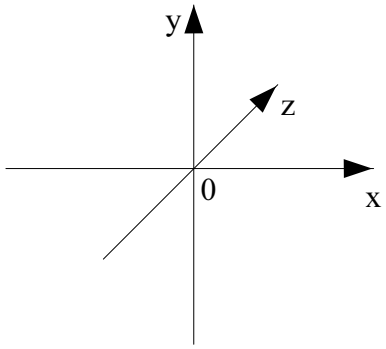


Pravotočivý súradnicový systém identifikujeme pomocou determinantu bázy. Ak je tento determinant kladný, jedná sa o pravotočivý súradnicový systém. Ako pomerne zavádzajúca pomôcka sa používa nasledovné pravidlo: ak osi tvoria vektory $\boldsymbol{x}$, $\boldsymbol{y}$ a $\boldsymbol{z}$, potom musí platiť, že ak sa pozrieme z miesta tretieho vektora na rovinu tvorenú ďalšími dvomi, potom musí prvý vektor prechádzať kratšou cestou proti smeru hodinových ručičiek do druhého vektora podľa vzťahov

$$\frac{\boldsymbol{z}}{\boldsymbol{x}\rightarrow\boldsymbol{y}} \qquad \frac{\boldsymbol{x}}{\boldsymbol{y}\rightarrow\boldsymbol{z}} \qquad \frac{\boldsymbol{y}}{\boldsymbol{z}\rightarrow\boldsymbol{x}}$$

Zápis znamená, že prvý spodný vektor prechádza proti smeru hodinových ručičiek do druhého pri pohľade z horného.

Ľavotočivý súradnicový systém



Opakom pravotočivého je ľavotočivý súradnicový systém. Determinant jeho bázy je záporný. Platia v ňom opačné vzťahy ako v pravotočivom, teda ak sa pozrieme z miesta tretieho vektora na rovinu tvorenú ďalšími dvomi vektormi, potom prvý vektor musí kratšou cestou prechádzať do druhého v smere hodinových ručičiek podľa vzťahov

$$\frac{\boldsymbol{z}}{\boldsymbol{x}\leftarrow\boldsymbol{y}} \qquad \frac{\boldsymbol{x}}{\boldsymbol{y}\leftarrow\boldsymbol{z}} \qquad \frac{\boldsymbol{y}}{\boldsymbol{z}\leftarrow\boldsymbol{x}}$$

Teraz sa vráťme k tomu, prečo je tento spôsob určovania "točivosti" priestoru zavádzajúci. Dôvod je prostý: bázické vektory nemajú žiaden súvis s tým, s akou orientáciou zobrazíme súradnicové osi. "Točivosť" je vlastnosťou bázy, nie spôsobu zobrazenia na papieri.

Kde sa prejavujú odlišné vlastnosti inak "točivých" priestorov ?

Problém vzniká pri vektorovom súčine. Ak sa pokúsime vyjadriť dva vektory $\boldsymbol{a}$, $\boldsymbol{b}$ v pravotočivom a ľavotočivom súradnicovom systéme (teda v príslušných bázach), potom výsledky vektorového súčinu v pravo- a ľavotočivom súradnicovom systéme vyjadrené v rovnakej báze B budú mať opačné znamienka: $(\boldsymbol{a}_r \times \boldsymbol{b}_r)_B = -(\boldsymbol{a}_p \times \boldsymbol{b}_p)_B$.

Vektorový súčin dvoch vektorov $\boldsymbol{a}=(a_x, a_y, a_z)$ a $\boldsymbol{b}=(b_x, b_y, b_z)$ sa dá vyjadriť symbolicky ako determinant $\begin{vmatrix} a_x & a_y & a_z \\ b_x & b_y & b_z \\ \boldsymbol{x} & \boldsymbol{y} & \boldsymbol{z} \end{vmatrix}$, kde ***x***, ***y***, ***z*** sú symbolické vyjadrenia súradnicových osí. Každý determinant regulárnej matice ***A*** rozmeru $\boldsymbol{n}\times\boldsymbol{n}$ zloženej z prvkov $\boldsymbol{a_{ij}}$ môžeme vyjadriť ako $|\boldsymbol{A}| = \sum_{\phi\in M} (-1)^{p(\phi)} a_{1,\phi(1)} a_{2,\phi(2)} \cdots a_{n,\phi(n)}$, kde ***M*** je množina permutácií čísel 1, 2, ... n a p(ϕ) je parita permutácie ϕ. Ak si vezmeme dve bázy, z ktorých je jedna pravo- a druhá ľavotočivá, pričom sa líšia iba tým, že jedná z nich má vymenené poradie riadkov (teda má navzájom zamenené dve súradnicové osi), potom pri výpočte vektorových súčinov v každej z dvoch báz dostávame rovnaké sčítance, akurát vždy s opačným znamienkom (zmenili sme paritu permutácie, v ktorej sa sčítance nachádzajú).

V ďaľších častiach bude súradnicový systém zobrazovaný ako na obrázku pri ľavotočivom priestore. Súradnica ***z*** v ňom bude nadobúdať význam hĺbky obrazu, ***x*** bude označovať šírku a ***y*** výšku obrazu (pohľadu). Ak nebude uvedené inak, "točivosť" priestoru nebude mať pre výpočty žiadny význam.

Báza priestoru

Báza priestoru je sústava vektorov, pomocou ktorých môžeme popísať daný priestor. Týchto vektorov musí byť toľko, koľko majú rozmerov a mali by byť navzájom lineárne nezávislé.

Pozrime sa na zvyčajnú bázu trojrozmerného priestoru $\boldsymbol{E_3}$, ktorú môžeme vyjadriť pomocou matice

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Prvý riadok si nazveme vektorom osi ***x***, druhý vektorom osi ***y*** a tretí vektorom osi ***z***. Tieto vektory sú navzájom nevyjadriteľné (nezávislé) a je ich toľko, koľko je rozmer každého z nich (zároveň sú na seba kolmé). Preto pomocu nich môžeme vyjadriť celý trojrozmerný priestor. Ak máme bod so súradnicami (a, b, c) v tejto báze, jeho skutočnú polohu môžeme rozpísať ako $a\,\boldsymbol{x} + b\,\boldsymbol{y} + c\,\boldsymbol{z}$. Keďže bázické vektory majú taký tvar, aký majú, dostávame opäť (a, b, c). Aké by to bolo pri inej báze ? Vytvorme si inú bázu, napríklad

$$\begin{bmatrix} 1 & 0 & -1 \\ 2 & 1 & 0 \\ 0 & 3 & 2 \end{bmatrix}$$

Ak máme súradnice (a, b, c) bodu v tejto báze, jeho polohu v priestore $\boldsymbol{E_3}$ vyjadríme ako $a\,\boldsymbol{x} + b\,\boldsymbol{y} + c\,\boldsymbol{z} = a(1, 0, -1) + b(2, 1, 0) + c(0, 3, 2)$. Bázu teda môžeme opísať ako sústavu smerov, v ktorých vyjadrujeme bod z priestoru (od počiatku v ***0***). Ako však vyjadriť bod z priestoru v nejakej báze ? Ak je ***r*** vektor súradníc v priestore s bázou ***B***, potom vieme, že jeho obraz ***p*** v báze priestoru $\boldsymbol{E_3}$ je rovný $\boldsymbol{p} = \boldsymbol{r}\,\boldsymbol{B}$.

Prenásobením súradníc bázou teda dostávame súradnice v jednotkovej báze $\boldsymbol{E_3}$. My však potrebujeme získať opačný vzťah. Využijeme preto vlastnosti inverznej matice.

$$p = r\,B$$
$$p\,B^{-1} = r\,B\,B^{-1}$$
$$p\,B^{-1} = r$$

Súradnice v priestore stačí vynásobiť inverznou maticou bázy a dostávame súradnice v báze.

Absolútny súradnicový systém

Pri vytváraní reprezentácie priestoru si zväčša zvolíme súradnicový systém a v ňom sa snažíme vyjadrovať súradnice všetkých objektov. Tento súradnicový systém je pevný, každý bod v ňom má svoje pevné miesto, ktoré sa nemení. Preto ho nazývame *absolútnym súradnicovým systémom*. Práca s ním je náročnejšia, pretože pri manipulácii s objektami vždy musíme brať do úvahy miesto, kde sa objekt nachádza a výsledné transformácie objektov vyzerajú na pohľad zložitejšie.

Relatívny súradnicový systém

Tento systém sa využíva najmä pri určovaní pohľadu na priestor. Absolútny súradnicový systém sa pretransformuje tak, aby miesto pohľadu ležalo v začiatku súradnicovej sústavy, os ***z*** išla smerom "do obrazu" a osi ***x*** a ***y*** boli rovnobežné s okrajmi obrazu. Potom sa transformácie obrazu dajú vyjadriť jednoduchšie, pretože odpadá povinnosť centrovania a nakláňania objektov.

Prechod medzi týmito dvomi systémami je založený na posunutí pôvodných objektov o nový stred, ktorý sa stáva bodom **0** a prenásobení inverznou maticou bázy absolútneho súradnicového systému.

Homogénne súradnice

Pozrime sa, ako prebieha transformovanie bodov priestoru na výsledný obraz. Ak máme priestor zložený z množstva objektov, zrejme budeme najskôr chcieť urobiť nejakú transformáciu, ktorá sa týka iba jedného objektu - napríklad jeho posunutie a otočenie. Túto transformáciu si nazvime $\boldsymbol{M}(\theta)$, kde θ je objekt, ktorý transformujeme. Ďalšiu transformáciu chceme urobiť už so všetkými objektami, pretože vyjadruje spoločnú vlastnosť všetkých objektov. Takouto transformáciou môže byť napríklad natočenie priestoru tak, aby jeho súradnice neboli absolútne, ale aby vyjadrovali súradnice pri pohľade očami. Túto transformáciu si nazvime $\boldsymbol{C}(\xi)$. Nakoniec potrebujeme transformáciu, ktorá nám trojrozmerné súradnice zmení na dvojrozmerné súradnice v obraze - nazvime si ju $\boldsymbol{P}(\varrho)$. Potom pre každý objekt ψ môžeme vyjadriť jeho transformáciu z absolútnych súradníc priestoru do súradníc obrazu ako $\boldsymbol{P} \circ \boldsymbol{C} \circ \boldsymbol{M}(\psi) = \boldsymbol{P}(\boldsymbol{C}(\boldsymbol{M}(\psi)))$. Pozrime sa, ako by vyzerali vzťahy pre transformáciu, ak by transformácie boli lineárne, t.j. dali by sa vyjadriť pomocou matíc. Transformáciu $\boldsymbol{M}(\theta)$ by bolo možné vyjadriť pomocou matice $\boldsymbol{M}$, $\boldsymbol{C}(\xi)$ pomocou matice $\boldsymbol{C}$ a $\boldsymbol{P}(\varrho)$ pomocou matice $\boldsymbol{P}$. Výsledná transformácia by mala tvar $\boldsymbol{M} \cdot \boldsymbol{C} \cdot \boldsymbol{P}$. Ak by sme mali objekt ψ daný vo forme matice tak, že jeho riadky by obsahovali jednotlivé body a stĺpce v týchto riadkoch by vyjadrovali po jednej súradnici priestoru, mohli by sme nové súradnice ζ objektu ψ získané po transformácii vyjadriť ako $\zeta = \psi \cdot \boldsymbol{M} \cdot \boldsymbol{C} \cdot \boldsymbol{P}$. Výpočet týchto súradníc je možný dvomi spôsobmi - najskôr prenásobiť súradnice ψ maticou $\boldsymbol{M}$, potom nové súradnice prenásobiť maticou $\boldsymbol{C}$, a nakoniec vzniknuté súradnice prenásobiť maticou $\boldsymbol{P}$. Týmto spôsobom by sme síce získali korektný výsledok, avšak by sme predlžovali čas úmerne počtu použitých transformácií. Možeme ale využiť vlastnosť matíc, pomocou ktorej získame konštantný čas pre ľubovoľne veľkú transformáciu. Ak si najskôr vypočítame celkovú transformáciu $\boldsymbol{T} = \boldsymbol{M} \cdot \boldsymbol{C} \cdot \boldsymbol{P}$, potom vzťah medzi pôvodnými súradnicami v priestore a v obraze vyjadríme jednoducho ako $\zeta = \psi \cdot \boldsymbol{T}$. Po rozpísaní dostaneme:

$$\begin{bmatrix} \psi_{x0} & \psi_{y0} & \psi_{z0} \\ \psi_{x1} & \psi_{y1} & \psi_{z1} \\ \psi_{x2} & \psi_{y2} & \psi_{z2} \\ \psi_{x3} & \psi_{y3} & \psi_{z3} \\ \psi_{x4} & \psi_{y4} & \psi_{z4} \\ & \cdots & \\ \psi_{xn} & \psi_{yn} & \psi_{zn} \end{bmatrix} \cdot \begin{bmatrix} T_{11} & T_{12} & T_{13} \\ T_{21} & T_{22} & T_{23} \\ T_{31} & T_{32} & T_{33} \end{bmatrix} = \begin{bmatrix} \zeta_{x0} & \zeta_{y0} & \zeta_{z0} \\ \zeta_{x1} & \zeta_{y1} & \zeta_{z1} \\ \zeta_{x2} & \zeta_{y2} & \zeta_{z2} \\ \zeta_{x3} & \zeta_{y3} & \zeta_{z3} \\ \zeta_{x4} & \zeta_{y4} & \zeta_{z4} \\ & \cdots & \\ \zeta_{xn} & \zeta_{yn} & \zeta_{zn} \end{bmatrix}$$

Je teda zrejmé, že je výhodnejšie skladať podtransformácie do jednej transformácie. Je tu však jeden problém. Nie všetky transformácie sa dajú vyjadriť pomocou lineárnych zobrazení (matíc). Napríklad už len obyčajné posunutie nie je možné týmto spôsobom vyjadriť a to je pritom jedna z najpoužívanejších transformácií. Nie je totiž možné pomocou súčinu vektorov s maticami vyjadriť pripočítanie čísla, ktoré nezávisí ani od jednej súradnice vektora. Taktiež perspektívne transformácie si vyžadujú delenie číslom, ktoré naopak závisí od hodnôt zložiek vektora, ktorý transformujeme, a to sa pomocou matíc opäť nedá vyjadriť. Ako to vyriešiť ? Ako nestratiť možnosť skladania transformácií do jednej matice a zároveň byť schopným vyjadriť posunutie a perspektívu ? Jedným z riešení sú homogénne súradnice.

Tieto sú založené na myšlienke rozšírenia 3D súradníc o jednu zvyšnú súradnicu - homogénnej zložky. Dostávame teda štvrtý rozmer a všetky naše transformácie sa zmenia na 4D.

Ak sme pôvodne mali vektor $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$, dostávame nové súradnice $(\boldsymbol{x'}, \boldsymbol{y'}, \boldsymbol{z'}, \boldsymbol{h})$, kde $\boldsymbol{h}$ je homogénna zložka. Ak by sme chceli z homogénnych súradníc vyjadriť súradnice v pôvodnom priestore, tieto by mali tvar $\left(\frac{\boldsymbol{x'}}{\boldsymbol{h}}, \frac{\boldsymbol{y'}}{\boldsymbol{h}}, \frac{\boldsymbol{z'}}{\boldsymbol{h}}\right)$. Pre súradnice, ktoré konvertujeme priamo z 3D priestoru sa $\boldsymbol{h}$ volí rovné 1. Homogénne súradnice potom majú tvar $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}, \boldsymbol{1})$. Pomocou týchto súradníc sme už schopní vyjadriť posunutie ako aj perspektívnu transformáciu. Posunutie by sme mohli vyjadriť pomocou matice

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ p_x & p_y & p_z & 1 \end{bmatrix},$$

ktorá nám zabezpečí posunutie o vektor (p_x, p_y, p_z).

Perspektívu (centrálnu projekciu) by sme mohli vyjadriť pomocou matice

$$\begin{bmatrix} k & 0 & 0 & 0 \\ 0 & l & 0 & 0 \\ r_x & r_y & 1 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Je možné si všimnúť, že homogénnu zložku perspektívy sme dali rovnú súradnici $\boldsymbol{z}$.
Po prenásobení vektora $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}, \boldsymbol{1})$ touto maticou dostávame vektor $(k\,\boldsymbol{x} + \boldsymbol{z}\,\boldsymbol{r}_x, l\,\boldsymbol{y} + \boldsymbol{z}\,\boldsymbol{r}_y, \boldsymbol{z}, \boldsymbol{z})$ a po konverzii do pôvodného 3D priestoru $(k\frac{\boldsymbol{x}}{\boldsymbol{z}} + \boldsymbol{r}_x, l\frac{\boldsymbol{y}}{\boldsymbol{z}} + \boldsymbol{r}_y, \boldsymbol{1})$, čo až na *z*-ovu súradnicu sú vzťahy totožné s centrálnou projekciou.

Ostatné transformácie, ako je napríklad rotácia a zošikmenie vyjadríme opäť pomocou štvorrozmernej matice, pričom táto bude totožná s pôvodnou trojrozmernou maticou, avšak pribudne jeden nulový riadok a nulový stĺpec a do pravého dolného rohu dáme ***1***.

Napríklad maticu otočenia okolo osi $\boldsymbol{z}$ by sme mohli transformovať takto:

$$\begin{bmatrix} \cos\alpha & \sin\alpha & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \longleftrightarrow \begin{bmatrix} \cos\alpha & \sin\alpha & 0 & \mathbf{0} \\ -\sin\alpha & \cos\alpha & 0 & \mathbf{0} \\ 0 & 0 & 1 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} \end{bmatrix}$$

Reprezentácia objektov

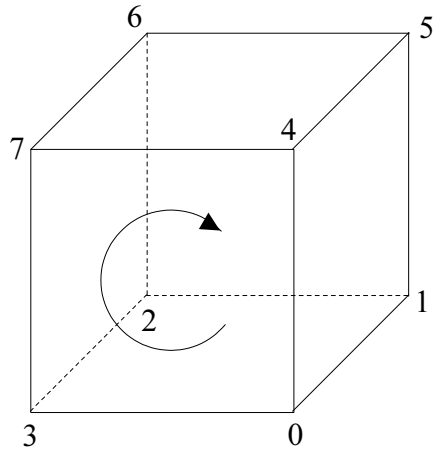
Pre prácu s objektom pomocou počítača je potrebné určitým spôsobom reprezentovať objekt v počítači. Reálne objekty sa zväčša silne zjednodušujú, vyberajú sa z nich iba niektoré body a plochy, zanedbávajú sa nerovnosti povrchu, vystihuje sa len najpodstatnejší tvar objektu a podobne.

Hranová reprezentácia objektu

Ako najjednoduchšia reprezentácia objektu sa využíva hranová reprezentácia. Objekt pozostáva z bodov a hrán, ktoré sú medzi dvojicou bodov z tohto zoznamu. Využíva sa často pri jednoduchých systémoch zobrazovania alebo pri systémoch, ktoré sa snažia modelovať isté typy vlastností objektov.

Plošná reprezentácia objektu

Asi najpoužívanejšou reprezentáciou objektu v systémoch zobrazovania v reálnom čase je plošná reprezentácia objektu. Podobne ako hranová, pozostáva zo zoznamu bodov a zoznamu plôch. Plochy sú určené počtom bodov a ich indexami v zozname bodov. Mnohokrát sa používa priame vyjadrenie bodov plôch pomocou ich súraníc bez existencie zoznamu bodov. Ako by však vyzeralo konkrétne vyjadrenie telesa v plošnej reprezentácii ? Ako príklad môže poslúžiť reprezentácia kocky:



Body ležia symetricky okolo stredu vo vzdialenosti $\sqrt{3}$ od stredu súradnicovej sústavy).

Zoznam bodov

0: (1; -1; -1)
1: (1; -1; 1)
2: (-1; -1; 1)
3: (-1; -1; -1)
4: (1; 1; -1)
5: (1; 1; 1)
6: (-1; 1; 1)
7: (-1; 1; -1)

Zoznam plôch

0: 4 (3; 7; 4; 0)
1: 4 (0; 4; 5; 1)
2: 4 (1; 5; 6; 2)
3: 4 (2; 6; 7; 3)
4: 4 (2; 3; 0; 1)
5: 4 (7; 6; 5; 4)

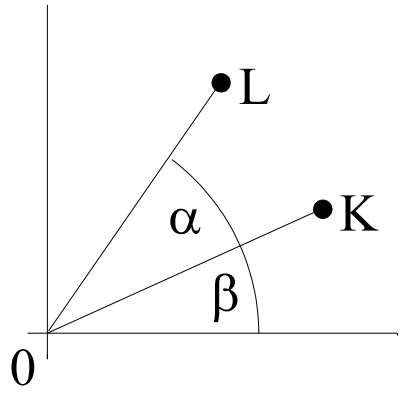
Plochy je vhodné ukladať tak, aby viditeľná strana plochy mala body zoradené v smere hodinových ručičiek. To potom umožňuje jednoduchý výpočet normálového vektora ploch pomocou troch následujúcich bodov mnohouholníka.

Okrem zoznamu bodov a plôch je možné ukladať zoznam normál plôch, normál v bodoch, informácie o intenzite svetla, o súradniciach v bodov štruktúrach mnohouholníkov, alebo je možné rozdeliť objekt na viacero vzájomne nezávislých častí.

Rotácia

Niekedy je potrebné otočiť bod alebo viacero bodov okolo iného bodu (osi). V dvojrozmernom priestore sa bod otáča okolo jedného bodu, v trojrozmernom okolo osi. Os je priamka zložená z bodov, ktoré počas rotácie nemenia svoju polohu.

Rotácia v rovine



Chceme otočiť bod $\boldsymbol{K}=(\boldsymbol{x},\boldsymbol{y})$ o uhol $\boldsymbol{\alpha}$ okolo $\mathbf{0}$. Otočením získame bod $\boldsymbol{L}=(\boldsymbol{x}',\boldsymbol{y}')$. Ako však získame jeho súradnice ?

Súradnice bodu $\boldsymbol{K}$ sa dajú získať aj takto:

$$\boldsymbol{x}=\sqrt{\boldsymbol{x}^2+\boldsymbol{y}^2}\cos(\boldsymbol{\beta})=\boldsymbol{r}\cos(\boldsymbol{\beta})$$

$$\boldsymbol{y}=\sqrt{\boldsymbol{x}^2+\boldsymbol{y}^2}\sin(\boldsymbol{\beta})=\boldsymbol{r}\sin(\boldsymbol{\beta})$$

čo je vlastne vyjadrenie súradníc $\boldsymbol{x}$, $\boldsymbol{y}$ v polárnych súradniciach. ($\boldsymbol{r}$ je vzdialenosť bodu $\boldsymbol{K}$ od stredu súradnicovej sústavy, β je uhol, ktorý zviera vektor $\boldsymbol{K}$ s osou x.

Potom súradnice bodu $\boldsymbol{L}$ môžeme vyjadriť ako $\boldsymbol{x}'=\boldsymbol{r}\cos(\boldsymbol{\alpha}+\boldsymbol{\beta})$ a $\boldsymbol{y}'=\boldsymbol{r}\sin(\boldsymbol{\alpha}+\boldsymbol{\beta})$. Keďže platí $\cos(\alpha+\beta)=\cos(\alpha)\cos(\beta)-\sin(\alpha)\sin(\beta)$ a $\sin(\alpha+\beta)=\sin(\alpha)\cos(\beta)+\cos(\alpha)\sin(\beta)$, súradnice bodu $\boldsymbol{L}$ môžeme vyjadriť ako

$$\boldsymbol{x}'=\boldsymbol{r}\cos(\boldsymbol{\beta})\cos(\boldsymbol{\alpha})-\boldsymbol{r}\sin(\boldsymbol{\beta})\sin(\boldsymbol{\alpha})$$
$$\boldsymbol{y}'=\boldsymbol{r}\cos(\boldsymbol{\beta})\sin(\boldsymbol{\alpha})+\boldsymbol{r}\sin(\boldsymbol{\beta})\cos(\boldsymbol{\alpha})$$

Ak sa pozrieme na predchádzajúce vzťahy, zistíme, že tieto súradnice sa dajú zjednodušiť na tvar

$$\boldsymbol{x}'=\boldsymbol{x}\cos(\boldsymbol{\alpha})-\boldsymbol{y}\sin(\boldsymbol{\alpha})$$
$$\boldsymbol{y}'=\boldsymbol{x}\sin(\boldsymbol{\alpha})+\boldsymbol{y}\cos(\boldsymbol{\alpha})$$

Pomocou matice môžeme výpočet vyjadriť takto

$$(\boldsymbol{x}\ \boldsymbol{y})\begin{bmatrix}\cos\alpha & \sin\alpha\\ -\sin\alpha & \cos\alpha\end{bmatrix}=(\boldsymbol{x}'\ \boldsymbol{y}')$$

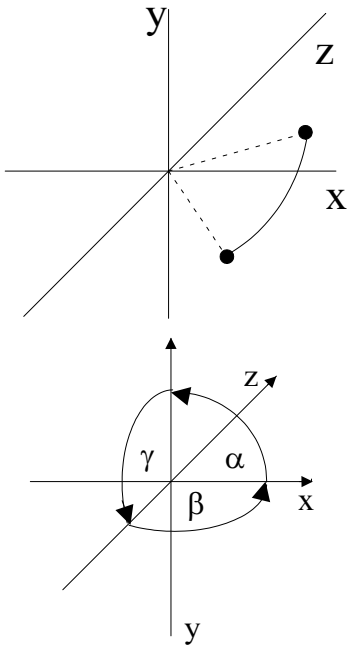
Rotáciu v rovine s klasickou euklidovskou bázou môžeme zovšeobecniť aj pre rotáciu v ľubovoľnej rovine danej bázou $\boldsymbol{B}$ tvorenou dvomi vektormi $\boldsymbol{u}=(u_1,u_2,\dots u_N)$ a $\boldsymbol{v}=(v_1,v_2,\dots v_N)$, kde N je dimenzia priestoru (pre rovinu je 2, pre trojrozmerný priestor 3...). Pomocou týchto dvoch vektorov môžeme napríklad rotovať okolo ľubovoľnej osi v trojrozmernom priestore alebo rotovať po eliptickej dráhe v dvojrozmernom priestore, pokiaľ $\boldsymbol{u}$ a $\boldsymbol{v}$ budú vektormi hlavnej a vedľajšej osi žiadanej elipsy, atď. Všeobecná rotácia bude mať tvar

$$(\boldsymbol{x}\ \boldsymbol{y})\begin{bmatrix}\cos\alpha & \sin\alpha\\ -\sin\alpha & \cos\alpha\end{bmatrix}\boldsymbol{B}+\boldsymbol{S}=(\boldsymbol{x}\ \boldsymbol{y})\begin{bmatrix}\cos\alpha & \sin\alpha\\ -\sin\alpha & \cos\alpha\end{bmatrix}\begin{bmatrix}u_1,u_2,\dots,u_N\\ v_1,v_2,\dots,v_N\end{bmatrix}+\boldsymbol{S}=(z'_1,\dots,z'_N)$$

kde $\boldsymbol{S}$ je bod na osi v priestore, ktorý leží najbližšie k rotovanému bodu ($\boldsymbol{x}$, $\boldsymbol{y}$).

Je dobré všimnúť si, že ***(x, y)*** sú súradnice bodu v báze ***B***, zatiaľ čo $(z'_0, z'_1, \ldots, z'_{N-1})$ sú súradnice bodu v *N*-rozmernom priestore. Rotujeme súradnice v rovine, na ktorú pozeráme, akoby bola bežnou rovinou s Euklidovskou bázou pre dvojrozmerný priestor. Pokiaľ rovina rotácie prechádza bodom ***0*** a súradnice (***x***, ***y***) prenásobíme bázou ***B***, dostávame súradnice v pôvodnom *N*-rozmernom priestore. Báza je spojivo medzi dvomi sústavami súradníc - súradníc v báze a súradníc v priestore. V tomto prípade nám súradnice (***x***, ***y***) hovoria, že $x\,\boldsymbol{u} + y\,\boldsymbol{v} = (z_0, z_1, \ldots, z_{N-1})$.

Rotácia v trojrozmernom priestore



V priestore si najskôr musíme zvoliť os, okolo ktorej chceme otáčať bod. Vyjadrime si najskôr rotácie okolo jednotlivých osí priestoru, osí ***x***, ***y*** a ***z***.

Najskôr sa pokúsime odvodiť vzťahy pre rotáciu okolo osi ***z*** (v rovine danej vektormi ***x*** a ***y***).

Zistíme, že dostávame rovnaký vzťah ako pre rotáciu v rovine, akurát pribudne ***z***-ová súradnica, ktorá sa však počas rotácie nemení.

Výsledná matica rotácie bude teda

$$\begin{bmatrix} \cos\alpha & \sin\alpha & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Podobne, v rovine danej vektormi ***x*** a ***z*** (okolo osi ***y***) bude rotačná matica takáto

$$\begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix}$$

A v rovine danej vektormi ***y*** a ***z*** (okolo osi ***x***) zase takáto

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{bmatrix}$$

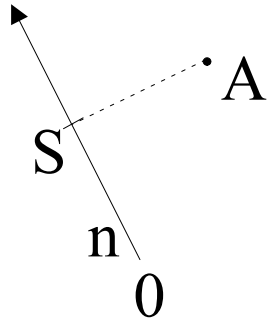
Súradnice otočeného bodu získame tak, že prenásobíme súradnice pôvodného bodu maticou rotácie, napríklad ***Tr***. Potom môžeme vyjadriť $(x, y, z)\,\boldsymbol{Tr} = (x', y', z')$. Rotácie môžeme aj skladať. Ak ***Tx*** je matica rotácie okolo osi ***x***, ***Ty*** okolo osi ***y*** a ***Tz*** okolo osi ***z***, môžeme si vytvoriť ľubovoľnú postupnosť rotácií tak, že budeme postupne násobiť tieto matice, až kým nedostaneme maticu, ktorá nám vyjadrí všetky tieto rotácie ako jedinú transformáciu. Treba si však dať pozor, pretože záleží na poradí vykonávania rotácií (násobení matíc). Napríklad si môžeme vytvoriť maticu ***Txyz*** =***Tz Tx Ty***, ktorá nám zrotuje bod okolo všetkých osí. Táto transformácia sa používa pri modeli kamery, kde dvojica uhlov ukazuje vodorovné a vertikálne natočenie kamery a tretí jej zošikmenie.

Rotácia okolo ľubovoľnej osi

V tejto časti sú opísané dva spôsoby rotácie okolo ľubovoľnej osi. Prvý spôsob využíva všeobecnú rotáciu, druhý zase skladanie rotácií okolo súradnicových osí.

Všeobecná rotácia

Máme bod $\boldsymbol{A}=[A_x, A_y, A_z]$, ktorý chceme otočiť okolo osi roviny danej jednotkovým normálovým vektorom $\boldsymbol{n}=[n_x, n_y, n_z]$ prechádzajúcej **0** o uhol α.. Celé to možno vyjadriť takto:

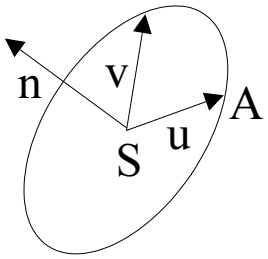


Chceme otočiť bod ***A*** v rovine danej vektorom ***n*** okolo bodu ***S***, ktorý je k bodu ***A*** najbližšie zo všetkých bodov na priamke určenej vektorom ***n*** a bodom **0**. Bod ***S*** potom môžeme vyjadriť ako $\boldsymbol{S}=\boldsymbol{k}\,\boldsymbol{n}\,, k\in R$. Zároveň musí byť vektor ***SA*** kolmý na vektor ***n***, $\boldsymbol{n}\cdot(\boldsymbol{A}-\boldsymbol{S})=0$. Z toho môžeme získať súradnice bodu ***S***, pretože $\boldsymbol{k}=\dfrac{\boldsymbol{A}\cdot\boldsymbol{n}}{\boldsymbol{n}\cdot\boldsymbol{n}}$, čo je vlastne súradnica kolmého priemetu bodu ***A*** do vektora ***n***. Máme teda bod ***S***, okolo ktorého budeme bod ***A*** otáčať. Ako to však urobiť ?

Pozrime sa na to, ako otáčame bod v dvojrozmernom priestore. Máme dané dve na seba kolmé osi, niekde ležiaci bod, ktorý chceme otočiť o nejaký uhol okolo počiatku. Tento bod stačí vynásobiť rotačnou maticou a všetko je hotové. Dá sa niečo podobné urobiť aj v našom prípade ?

Vyjadrenie súradnicových osí roviny rotácie

Zvyčajná rovina v 2D má bázu $\boldsymbol{P}=\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}$. Môžeme vidieť, že báza je tvorená dvomi vektormi, z ktorých každý určuje jednu súradnicu bodu v tejto rovine. Ak máme bod $(\boldsymbol{x}, \boldsymbol{y})$, hovoríme vlastne o tom, že jeho súradnice sú $\boldsymbol{x}[1\quad 0]+\boldsymbol{y}[0\quad 1]$. To je však to isté, ako zápis $\mathbf{1}[x\quad y]+\mathbf{0}[-y\quad x]$. Báza $\boldsymbol{R}=\begin{bmatrix}x & y\\ -y & x\end{bmatrix}$ je opäť ortogonálna, súradnice nášho bodu sú však $(\mathbf{1},\mathbf{0})$. Teda $(\boldsymbol{x}, \boldsymbol{y})\boldsymbol{P}=(\mathbf{1},\mathbf{0})\boldsymbol{R}$. Na rotáciu okolo počiatku potom máme dve možnosti: buď prenásobíme bod $(\boldsymbol{x}, \boldsymbol{y})$ rotačnou maticou, alebo ňou prenásobíme bod $(\mathbf{1},\mathbf{0})$, a následne výsledok prenásobíme bázou ***R***, aby sme zistili, kam sa v skutočnosti posunul bod $(\mathbf{1},\mathbf{0})$. Tým, že báza ***R*** je ortogonálna a jej zložky majú rovnakú veľkosť, máme zaručené, že nedôjde k nežiadúcej deformácii pri otočení bodu. Túto vlastnosť využijeme pri rotácii v rovine v trojrozmernom priestore.



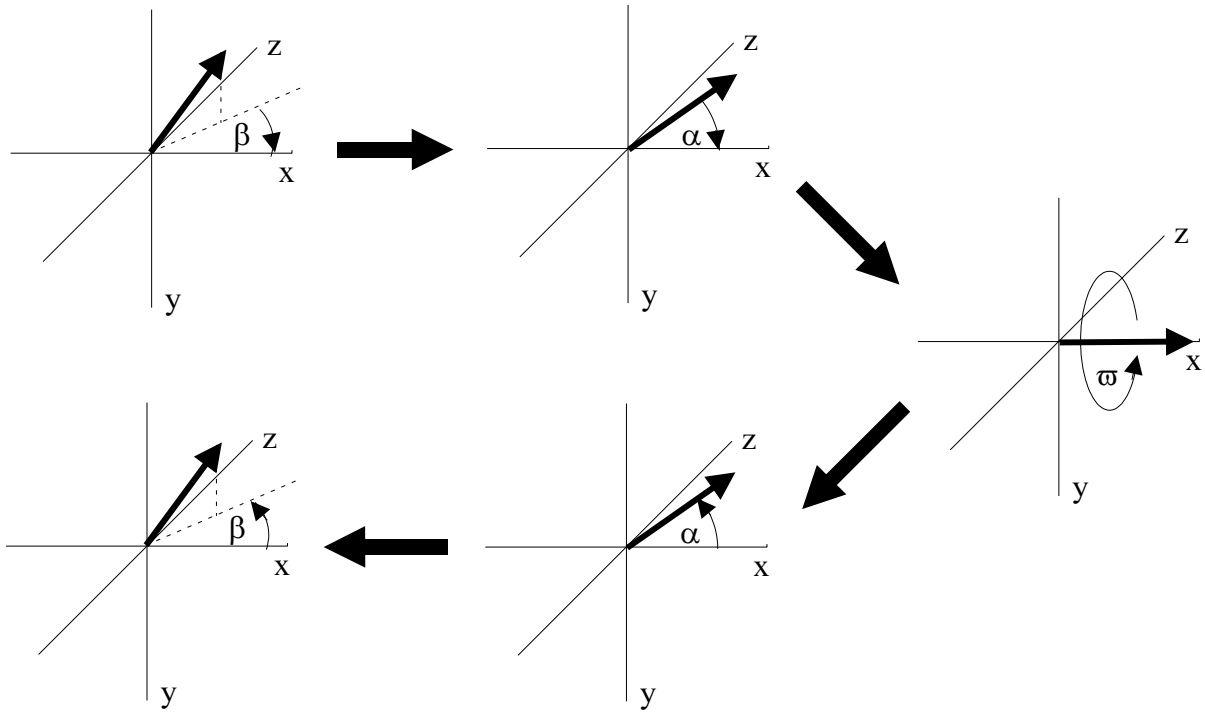
Keď sa pozrieme na našu rovinu, zistíme, že je natočená dvomi smermi, určenými vektorom ***n*** a vektorom ***SA***. Potrebujeme nájsť takú bázu tejto roviny, ktorá je ortogonálna a jej zložky majú rovnakú veľkosť (aby sme ich mohli stotožniť s osou ***x*** a ***y*** roviny v dvojrozmernom priestore) a zároveň, aby bod ***A*** v nej mal súradnicu $(\mathbf{1},\mathbf{0})$. Prvý vektor bázy získame jednoducho - je ním vektor ***SA***. Tento vektor označíme ***u***. Ako však získať druhý vektor bázy ? Musí mať rovnakú veľkosť ako ***u*** a byť naň kolmý, ako aj na vektor ***n***. Ak označíme $\boldsymbol{v}=\boldsymbol{n}\times\boldsymbol{u}$, získame vektor, ktorý je kolmý na oba vektory, a keďže ***n*** je jednotkový vektor, $|\boldsymbol{u}|=|\boldsymbol{v}|$. Dostávame teda bázu $\boldsymbol{Q}=\begin{bmatrix}\boldsymbol{u}\\ \boldsymbol{v}\end{bmatrix}$, bod ***A*** má v nej súradnice $(\mathbf{1},\mathbf{0})$ Rotáciu okolo osi potom môžeme vyjadriť ako $(\mathbf{1}\ \mathbf{0})\begin{bmatrix}\cos\alpha & \sin\alpha\\ -\sin\alpha & \cos\alpha\end{bmatrix}\begin{bmatrix}\boldsymbol{u}\\ \boldsymbol{v}\end{bmatrix}+\boldsymbol{S}=\begin{bmatrix}\cos\alpha & \sin\alpha\end{bmatrix}\begin{bmatrix}\boldsymbol{u}\\ \boldsymbol{v}\end{bmatrix}+\boldsymbol{S}=(\boldsymbol{x}', \boldsymbol{y}', \boldsymbol{z}')$. Dostávame vlastne vyjadrenie pohybu bodu po kružnici v ľubovoľnej báze. Ak by $|\boldsymbol{u}|\neq|\boldsymbol{v}|$, bod by rotoval po

eliptickej dráhe.

Keby ***u*** a ***v*** neboli ortogonálne, bod by rotoval po naklonenej elipse. Tento spôsob rotácie má navyše aj tú výhodu, že pokiaľ nebudeme meniť os rotácie, stačí ***u*** a **v** vypočítať iba raz na začiatku, a potom len meniť uhol α .

Skladanie podrotácií

Ďalšou možnosťou, ako rotovať okolo ľubovoľnej osi je postupne skladať podrotácie okolo jednotlivých osí až kým nedostaneme hľadanú rotáciu. Spôsobov, akými to možno docieliť, je nekonečne veľa, ale výsledok by mal byť vždy rovnaký. Princíp jedného zo spôsobov rotácie spočíva v tom, že sa najskôr pokúsime prerotovať bod tak, aby os rotácie, ktorú chceme vykonať, bola totožná s jednou z troch súradnicových osí. Potom stačí daný bod otočiť v rovine tejto osi o zvolený uhol a inverznými rotáciami vrátiť os rotácie do pôvodnej polohy. Keďže os rotácie je množina bodov, ktoré počas rotácie nemenia svoju polohu, inverznými transformáciami by sa mala dostať na pôvodné miesto.



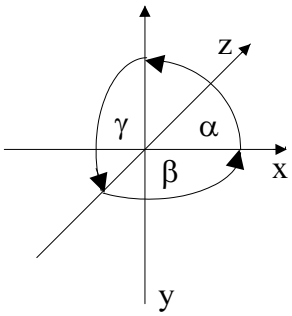
Tu je zobrazený postup skladania rotácií na dosiahnutie otočenia okolo ľubovoľnej osi. Najskôr sa os rotácie ***n*** otočí o uhol -β okolo osi ***y***. Následným otočením okolo osi ***z*** o uhol -α splynie s osou ***x***. Vtedy možno otočiť daný bod okolo osi ***x*** o zvolený uhol ϖ. Po ukončenej rotácii je potrebné vrátiť os otáčania do pôvodného stavu, preto sa vykoná inverzná rotácia okolo osi ***z*** o uhol α a po nej inverzná rotácia okolo osi ***y*** o uhol β.

Ako však určiť uhly α a β ? Ak $\boldsymbol{n} = (\boldsymbol{n}_x, \boldsymbol{n}_y, \boldsymbol{n}_z)$, potom sa dajú vyjadriť týmto spôsobom:

$$\alpha = \arctan\left(\frac{\boldsymbol{n}_y}{\sqrt{\boldsymbol{n}_x^{\,2} + \boldsymbol{n}_z^{\,2}}}\right) \quad \text{a} \quad \beta = \arctan\left(\frac{\boldsymbol{n}_z}{\boldsymbol{n}_x}\right)$$

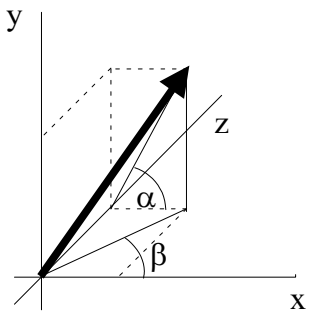
Pre odvodenie symbolických vzťahov je však lepšie vyjadriť si súradnice ***x***, *y* a ***z*** pomocou týchto uhlov. Môžeme ich získať z vyjadrenia jednotkového vektora osi rotácie - ***n*** vo sférických súradniciach.

Sférické súradnice v priestore



Pri bližšom pohľade na súradnicový systém si môžeme všimnúť spôsob počítania uhlov v priestore. Uhol α je uhol v rovine ***xy***, rastie v smere od zväčšujúcej sa hodnoty ***x*** po zväčšujúcu sa hodnotu ***y***. Uhol β je uhol v rovine ***xz***, rastie v smere od zväčšujúcej sa hodnoty ***x*** po zväčšujúcu sa hodnotu ***z***. A nakoniec uhol γ, ktorý je v rovine ***yz*** a rastie v smere od zväčšujúcej sa hodnoty ***z*** po zväčšujúcu sa hodnotu ***y***.

*Ako potom vyjadriť sférické súradnice vektora **n** ?*



Vektor ***n*** môžeme charakterizovať dvomi uhlami - uhlom α v rovine ***xy*** (***yz***), uhlom β v rovine ***xz*** a dĺžkou, ktorá sa rovná jednej. Ak by sme ***n*** položili na os ***x*** a zrotovali v rovine ***xy*** o uhol α, jeho priemet do ***x***-ovej osi by bol **cos(α)**. Po následnom otočení okolo osi ***y*** o uhol β by tento priemet bol rovný **cos(α)cos(β)**. Pokiaľ by sme ***n*** položili na os ***z*** a zrotovali v rovine ***yz*** o uhol α a následne otočili okolo osi ***y*** o uhol β, jeho priemet do osi ***z*** by sa rovnal **cos(α)sin(β)**. Pri otočení ***n*** okolo osi ***x*** alebo ***z*** o uhol α je priemet ***n*** do ***y***-ovej osi rovný **sin(α)**. Odtiaľ dostávame sférické vyjadrenie hodnôt zložiek vektora ***n***.

$$\boldsymbol{n}_x = \mathbf{cos}(\alpha)\,\mathbf{cos}(\beta)$$
$$\boldsymbol{n}_y = \mathbf{sin}(\alpha)$$
$$\boldsymbol{n}_z = \mathbf{cos}(\alpha)\,\mathbf{sin}(\beta)$$

(Uhol α možno chápať ako náklon vektora ***n***, β ako otočenie smeru)

Vyjadrenie rotácie okolo ľubovoľnej osi

Postup možno zapísať týmto spôsobom: Ak máme bod ***b***, jednotkový vektor v smere osi rotácie ***n*** a ***R*ι(δ)** je matica rotácie okolo súradnicovej osi ι o uhol δ, potom pre súradnice otočeného bodu ***b'*** môžeme napísať $\boldsymbol{b'} = \boldsymbol{b}\ \boldsymbol{R}_y(-\beta)\ \boldsymbol{R}_z(-\alpha)\ \boldsymbol{R}_x(\omega)\ \boldsymbol{R}_z(\alpha)\ \boldsymbol{R}_y(\beta)$, kde α (β) sú uhly, ktoré zviera vektor ***n*** s rovinou ***xz*** (***xy***) a ϖ je uhol rotácie.

Môžeme sa pokúsiť vyjadriť celkovú transformáciu $\boldsymbol{R}_n$.

$$\boldsymbol{R}_n(\omega) = \boldsymbol{R}_y(-\beta)\ \boldsymbol{R}_z(-\alpha)\ \boldsymbol{R}_x(\omega)\ \boldsymbol{R}_z(\alpha)\ \boldsymbol{R}_y(\beta)$$

Po postupnom vynásobení matíc získavame jedinú maticu rotácie. Po jej zjednodušení a dosadení vzťahov pre výpočet súradníc vektora ***n*** dostaneme maticu

$$\boldsymbol{R}_n(\omega)=\begin{bmatrix} \boldsymbol{n}_x^{\,2}(\mathbf{1}-\cos\omega)+\cos\omega & \boldsymbol{n}_x\boldsymbol{n}_y(\mathbf{1}-\cos\omega)-\boldsymbol{n}_z\sin\omega & \boldsymbol{n}_x\boldsymbol{n}_z(\mathbf{1}-\cos\omega)+\boldsymbol{n}_y\sin\omega \\ \boldsymbol{n}_x\boldsymbol{n}_y(\mathbf{1}-\cos\omega)+\boldsymbol{n}_z\sin\omega & \boldsymbol{n}_y^{\,2}(\mathbf{1}-\cos\omega)+\cos\omega & \boldsymbol{n}_y\boldsymbol{n}_z(\mathbf{1}-\cos\omega)-\boldsymbol{n}_x\sin\omega \\ \boldsymbol{n}_x\boldsymbol{n}_z(\mathbf{1}-\cos\omega)-\boldsymbol{n}_y\sin\omega & \boldsymbol{n}_y\boldsymbol{n}_z(\mathbf{1}-\cos\omega)+\boldsymbol{n}_x\sin\omega & \boldsymbol{n}_z^{\,2}(\mathbf{1}-\cos\omega)+\cos\omega \end{bmatrix}$$

Táto matica sa dá vyjadriť aj ako $\boldsymbol{R}_n(\omega)=\boldsymbol{E}+\sin\omega\,\boldsymbol{P}+(\mathbf{1}-\cos\omega)\,\boldsymbol{P}^2$, kde $\boldsymbol{P}$ je

$$\begin{bmatrix} \mathbf{0} & -\boldsymbol{n}_z & \boldsymbol{n}_y \\ \boldsymbol{n}_z & \mathbf{0} & -\boldsymbol{n}_x \\ -\boldsymbol{n}_y & \boldsymbol{n}_x & \mathbf{0} \end{bmatrix}$$

Ak označíme $\boldsymbol{d}=\mathbf{1}-\cos\omega$, $\boldsymbol{e}=\boldsymbol{n}_x^{\,2}\boldsymbol{d}$, $\boldsymbol{f}=\boldsymbol{n}_y^{\,2}\boldsymbol{d}$, $\boldsymbol{g}=\boldsymbol{n}_z^{\,2}\boldsymbol{d}$, $\boldsymbol{i}=\boldsymbol{n}_x\boldsymbol{n}_y\boldsymbol{d}$, $\boldsymbol{j}=\boldsymbol{n}_x\boldsymbol{n}_z\boldsymbol{d}$, $\boldsymbol{k}=\boldsymbol{n}_y\boldsymbol{n}_z\boldsymbol{d}$, $\boldsymbol{o}=\boldsymbol{n}_x\sin\omega$, $\boldsymbol{p}=\boldsymbol{n}_y\sin\omega$, $\boldsymbol{r}=\boldsymbol{n}_z\sin\omega$ a $\boldsymbol{c}=\cos\omega$ matica $\boldsymbol{R}_n$ sa zmení na maticu

$$\begin{bmatrix} \boldsymbol{e}+\boldsymbol{c} & \boldsymbol{i}-\boldsymbol{r} & \boldsymbol{j}+\boldsymbol{p} \\ \boldsymbol{i}+\boldsymbol{r} & \boldsymbol{f}+\boldsymbol{c} & \boldsymbol{k}-\boldsymbol{o} \\ \boldsymbol{j}-\boldsymbol{p} & \boldsymbol{k}+\boldsymbol{o} & \boldsymbol{g}+\boldsymbol{c} \end{bmatrix} ,$$

ktorá nám výpočet o niečo urýchli.

Porovnanie spôsobov rotácie

Rotácia pomocou skladania matíc potrebuje na otočenie bodu 9 násobení. Všeobecná rotácia ich potrebuje oveľa viac, avšak ak sa objekt otáča stále okolo jednej osi, možno na začiatku predpočítať bázické vektory a rotácia bude potrebovať iba 6 násobení.

Rotácia skladaním matíc môže na rozdiel od všeobecnej využívať efekt skladania obrazových transformácií a využiť tú skutočnosť, že môže byť súčasťou transformácie obsahujúcej stovky iných transformácií, a pritom všetky tieto transformácie budú vyžadovať vždy rovnaký počet operácií.

Všeobecná rotácia však môže byť užitočná pre objekty, ktoré vykonávajú vlastný pohyb okolo stálej osi, kde sa spojené efekty maticových transformácií veľmi neprejavia. Môže byť tiež výhodná pri zobrazovaní grafov, kde umožní rýchlejšie otáčanie grafu vo zvolenej rovine.

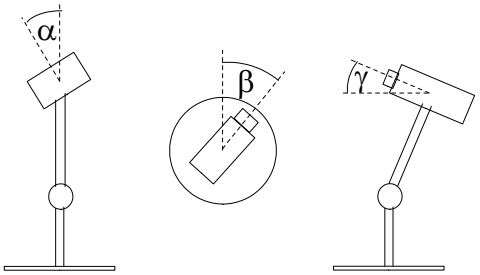
Pohľad

Predstavme si, že stojíme uprostred mesta. Každý bod, ktorý leží v nejakej budove má svoje absolútne súradnice - napríklad $\boldsymbol{B}=(B_x, B_y, B_z)$. My stojíme v bode $\boldsymbol{T}=(T_x, T_y, T_z)$. Keďže máme vo zvyku hýbať hlavou a prípadne ju aj natáčať, môžeme si všimnúť, že v našom pohľade nemajú všetky body budov absolútne súradnice, ale tieto sa menia podľa toho, ako hýbeme hlavou, aké rôzne náklony hlavy nás práve napadnú, alebo ako ďaleko sme prešli po nohách. Náš pohľad sa má teda vo zvyku stále meniť. Stále však platí, že hĺbka pohľadu sa zväčšuje tým smerom, ktorým sa pozeráme a vodorovný so zvislým smerom nezávisí od terajšieho "nakrútenia" hlavy (samozrejme, nemyslí sa tým vodorovný a zvislý smer vzhľadom na povrch, po ktorom sa pohybujeme). Preto pohľad môžeme brať ako relatívny súradnicový systém. Býva mnohokrát výhodnejšie vyjadriť si doposiaľ absolútne súradnice objektov ich relatívnymi súradnicami v pohľade. Napríklad nás mnoho ráz nezaujíma, že budova ***F*** leží 340 m západo-východným smerom a 125 m južne-severným smerom od centra mesta, ale nás možno viac zaujme, že ***F*** je od nás 70 m vpravo a 50 m dopredu. Prvý krok, ktorý musíme urobiť, aby sme zmenili absolútne súradnice na relatívne v našom pohľade, je posunutie všetkých bodov tak, aby sa miesto, kde sa nachádzame stalo začiatkom súradnicovej sústavy. To docielime tým, že od všetkých bodov odčítame miesto, kde sa nachádzame – ***B-T***. Potom musíme všetko nakloniť tak, aby sa to javilo ako v našom pohľade.

Popis pohľadu pomocou troch uhlov

Náš pohľad môžeme charakterizovať tromi uhlami. Prvý uhol, ***natočenie***, je uhol hlavy okolo osi ***y***. Určuje pohyb hlavy, ktorý ľudia zvyèajne vykonávajú, keï s niečím nesúhlasia. Druhý uhol nazveme ***sklon***. Je to uhol, ktorý sa mení okolo osi ***x*** a ľudia ho vykonávajú pohybom vtedy, keď s niečím súhlasia. Posledný uhol nazveme ***vychýlenie***, označuje uhol meniaci sa okolo osi ***z***, u ľudí by sa to dalo opísať pohybom hlavy raz k jednému a raz k druhému ramenu, čo u niektorých ľudí znamená možno. Keďže naša hlava sa otáča pomocou týchto troch uhlov, pohľad sa bude točiť presne naopak.

Ako ďaľší príklad môže poslúžiť kamera na otočnej nohe:



Uhol α predstavuje ***vychýlenie***, uhol β ***natočenie*** a uhol γ ***sklon*** kamery.

Predpokladáme, že začiatočný stav kamery je taký, že kamera má všetky tieto uhly nulové.

K natočeniu pohľadu kamery sa dostaneme tak, že najskôr kameru *vychýlime* okolo osi ***z***. Potom pokračujeme *sklonením* kamery okolo osi ***x***. Nakoniec kameru *otočíme* okolo osi ***y***. Vtedy dostaneme smer pohľadu, ktorý sme vyžadovali. Počiatočný pohľad kamery je totožný s osou ***z***, má teda súradnice $\boldsymbol{P}=(0,0,1)$. Ak $\boldsymbol{R}_{\iota}(\delta)$ je matica rotácie okolo osi ι o uhol δ, potom sa pôvodný vektor pohľadu ***P*** zmení na nový vektor pohľadu $\boldsymbol{P'}=\boldsymbol{P}\,\boldsymbol{R}_z(\alpha)\,\boldsymbol{R}_x(\gamma)\,\boldsymbol{R}_y(\beta)$. Dostali sme vyjadrenie vektora pohľadu v absolútnych súradniciach. My však potrebujeme presný opak - chceme vyjadriť absolútne súradnice v relatívnych súradniciach pohľadu, tak, aby pohľad bol stále rovný $\boldsymbol{P}=(0,0,1)$. Máme vyjadrené, že $\boldsymbol{P}\,\boldsymbol{A}=\boldsymbol{P'}$, kde ***A*** je transformácia z relatívnych do absolútnych súradnic, potrebujeme však vyjadriť $\boldsymbol{P}=\boldsymbol{P'}\,\boldsymbol{A'}$, čiže nájsť transformáciu, ktorá nám zmení absolútne súradnice na relatívne. Keďže platí $\boldsymbol{P}\,\boldsymbol{A}=\boldsymbol{P'}$, platí $\boldsymbol{P}\,\boldsymbol{A}\,\boldsymbol{A}^{-1}=\boldsymbol{P'}\,\boldsymbol{A}^{-1}$ a z toho dostávame $\boldsymbol{P}=\boldsymbol{P'}\,\boldsymbol{A}^{-1}$. ***A*** poznáme, môžeme sa pokúsiť vyjadriť $\boldsymbol{A}^{-1}$.

$$\boldsymbol{P}\ \boldsymbol{R}_z(\alpha)\ \boldsymbol{R}_x(\gamma)\ \boldsymbol{R}_y(\beta) = \boldsymbol{P}'$$
$$\boldsymbol{P}\ \boldsymbol{R}_z(\alpha)\ \boldsymbol{R}_x(\gamma) = \boldsymbol{P}'\ \boldsymbol{R}_y(-\beta)$$
$$\boldsymbol{P}\ \boldsymbol{R}_z(\alpha) = \boldsymbol{P}'\ \boldsymbol{R}_y(-\beta)\ \boldsymbol{R}_x(-\gamma)$$
$$\boldsymbol{P} = \boldsymbol{P}'\ \boldsymbol{R}_y(-\beta)\ \boldsymbol{R}_x(-\gamma)\ \boldsymbol{R}_z(-\alpha)$$

(Využili sme vlastnosť rotácie, že jej inverznou maticou je tá istá matica rotácie s opačným uhlom)

Dostali sme, že $\boldsymbol{A}^{-1} = \boldsymbol{R}_y(-\beta)\ \boldsymbol{R}_x(-\gamma)\ \boldsymbol{R}_z(-\alpha)$. Celkovú transformáciu bodu $\boldsymbol{B}$ vyjadreného v absolútnych súradniciach priestoru v relatívnych súradniciach pohľadu so začiatkom v bode $\boldsymbol{T}$ môžeme vyjadriť ako

$$(\boldsymbol{B} - \boldsymbol{T})\ \boldsymbol{R}_y(-\beta)\ \boldsymbol{R}_x(-\gamma)\ \boldsymbol{R}_z(-\alpha)$$

Ak ***M(V)*** označuje maticu posumutia o bod ***V***, potom môžeme napísať predchádzajúci vzťah ako

$$\boldsymbol{B}\ \boldsymbol{M}(-\boldsymbol{T})\ \boldsymbol{R}_y(-\beta)\ \boldsymbol{R}_x(-\gamma)\ \boldsymbol{R}_z(-\alpha)$$

Ak sa pokúsime vyjadriť inverznú maticu $\boldsymbol{A}^{-1}$, dostávame

$$\begin{bmatrix} \sin\alpha\sin\beta\sin\gamma + \cos\alpha\cos\beta & \sin\alpha\cos\gamma & \cos\alpha\sin\beta - \sin\alpha\cos\beta\sin\gamma \\ \cos\alpha\sin\beta\sin\gamma - \sin\alpha\cos\beta & \cos\alpha\cos\gamma & -\cos\alpha\cos\beta\sin\gamma - \sin\alpha\sin\beta \\ -\sin\beta\cos\gamma & \sin\gamma & \cos\beta\cos\gamma \end{bmatrix}$$

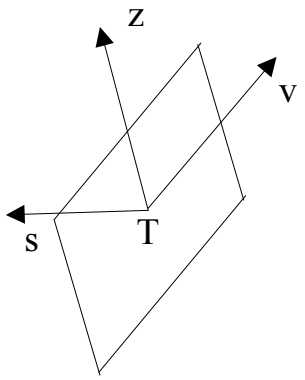
Z nej po spojení s posunom o bod $-\boldsymbol{T} = (-T_x, -T_y, -T_z)$ dostávame celkovú transformáciu

$$\begin{bmatrix} a & d & g & 0 \\ b & e & h & 0 \\ c & f & ch & 0 \\ -aT_x - bT_y - cT_z & -dT_x - eT_y - fT_z & -gT_x - hT_y - chT_z & 1 \end{bmatrix},$$

kde $z = \sin\alpha\sin\beta$, $x = \cos\alpha\cos\beta$, $y = \sin\alpha\cos\beta$, $w = \cos\alpha\sin\beta$, $a = z\sin\gamma + x$, $b = w\sin\gamma - y$, $c = -\sin\beta\cos\gamma$, $d = \sin\alpha\cos\gamma$, $e = \cos\alpha\cos\gamma$, $f = \sin\gamma$, $g = w - y\sin\gamma$, $h = -x\sin\gamma - z$ a $ch = \cos\beta\cos\gamma$.

Určenie pohľadu pomocou trojice vektorov

Určenie relatívnych súradníc pomocou uhlov nie je jediný spôsob, akým to dosiahnuť. Častokrát sa využíva priame vytvorenie bázy pomocou trojice jednotkových vektorov - ***smerového***, ***vodorovného*** a ***zvislého***. ***Smerový*** vektor je orientovaný v smere pohľadu, ***vodorovný*** ukazuje vodorovný smer a ***zvislý*** ukazuje zvislý smer obrazu.



Súradnice v pohľade sú teraz určené bázou, skladajúcou sa z troch navzájom kolmých jednotkových vektorov. Vektor ***s*** je smerovým, vektor ***v*** vodorovným a vektor ***z*** zvislým vektorom.

Ak si ich zvolíme tak, aby platilo $\boldsymbol{v} \times \boldsymbol{z} = \boldsymbol{s}$, môžeme pomocou nich vyjadriť bázu pohľadu $A = \begin{bmatrix} \boldsymbol{v} \\ \boldsymbol{z} \\ \boldsymbol{s} \end{bmatrix}$. Vektor ***v*** akoby nahrádzal súradnicu ***x***, ***z*** súradnicu ***y*** a ***s*** súradnicu ***z*** pôvodného priestoru. Ako v predchádzajúcom prípade, potrebujeme vyjadriť inverznú maticu $\boldsymbol{A}^{-1}$, pomocou ktorej by sme mohli premieňať vzájomne súradnice. Takúto maticu by bolo potrebné počítať pri každom pootočení pohľadu, čo by mohlo veľmi predlžovať čas výpočtov. Našťastie, keďže vektory tvoria ortogonalnu bázu a sú jednotkové, môžeme využiť, že $\boldsymbol{A}^{-1} = \boldsymbol{A}^{\mathbf{T}}$. Vieme, že $\boldsymbol{A}\,\boldsymbol{A}^{-1} = \boldsymbol{E}$, ale pri tejto matici platí aj $\boldsymbol{A}\,\boldsymbol{A}^{T} = \boldsymbol{E}$, a keďže jediná matica, pre ktorú to platí, je inverzná, matica $\boldsymbol{A}^{\mathbf{T}}$ je tiež inverznou maticou. Potom inverznú maticu $\boldsymbol{A}^{-1}$ vyjadríme ako

$$\boldsymbol{A}^{-1} = \begin{pmatrix} \boldsymbol{v}^T & \boldsymbol{z}^T & \boldsymbol{s}^T \end{pmatrix}$$

To, že $\boldsymbol{A}^{\mathbf{T}}$ je zároveň inverznou vyplýva z násobenia $\boldsymbol{A}\,\boldsymbol{A}^{\mathbf{T}}$, kde ide vlastne o vzájomné skalárne súčiny bázických vektorov. Keďže sú na seba kolmé, ľubovoľné dva rôzne vektory majú nulový skalárny súčin - vo výslednej matici vzniknú 0. Ak sa vynásobia dva rovnaké vektory, ich jednotková dĺžka zaručuje, že výsledok bude vždy 1. Jednotky však vzniknú iba na diagonále matice, inde budú samé nuly. Výsledná matica teda bude ***E***.

Celú zmenu súradníc bodu ***B*** z absolútnych súradníc do relatívnych s počiatkom v bode ***T*** s môžeme zapísať ako

$$(\boldsymbol{B} - \boldsymbol{T}) \begin{pmatrix} \boldsymbol{v}^T & \boldsymbol{z}^T & \boldsymbol{s}^T \end{pmatrix}$$

Ak ***M(V)*** opäť označuje maticu posumutia o bod ***V***, predošlý vzťah vyjadríme ako

$$\boldsymbol{B}\ \boldsymbol{M}(-\boldsymbol{T}) \begin{pmatrix} \boldsymbol{v}^T & \boldsymbol{z}^T & \boldsymbol{s}^T \end{pmatrix}$$

Ak $\boldsymbol{v} = (v_x, v_y, v_z)$, $\boldsymbol{z} = (z_x, z_y, z_z)$, $\boldsymbol{s} = (s_x, s_y, s_z)$ a miesto pohľadu je $\boldsymbol{T} = (T_x, T_y, T_z)$, transformačnú maticu vyjadríme v homogénnych súradniciach ako

$$\begin{bmatrix} v_x & z_x & s_x & 0 \\ v_y & z_y & s_y & 0 \\ v_z & z_z & s_z & 0 \\ -v_x T_x - v_y T_y - v_z T_z & -z_x T_x - z_y T_y - z_z T_z & -s_x T_x - s_y T_y - s_z T_z & 1 \end{bmatrix},$$

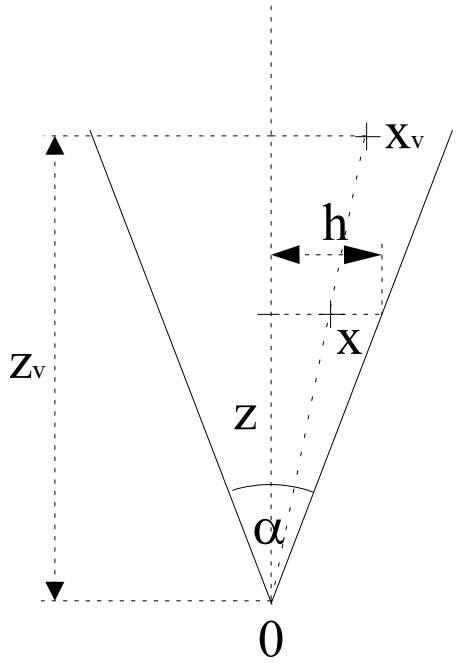
kde sa dá tušiť ekvivalencia s predošlým spôsobom pomocou troch uhlov.

Trojicu bázických vektorov ***v***, ***z***, ***s*** môžeme na začiatku dať totožnú s bázickymi vektormi absolútnych súradníc a aktualizovať ich vždy pri rotácii pohľadu.

Premietanie obrazu

Priestor objektov je potrebné vykresliť po jednotlivých zložkách na obrazovku. Toto si vyžaduje získanie vzťahu medzi zobrazovanými bodmi a ich zobrazeniami na obrazovke. Jednou z možností je použitie perspektívnej transformácie. V tejto časti budú prezentované tri spôsoby výpočtu centrálnej projekcie tak, ako sa vyskytujú v rôznej literatúre.

Centrálna projekcia vyjadrená pomocou rovnoľahlosti

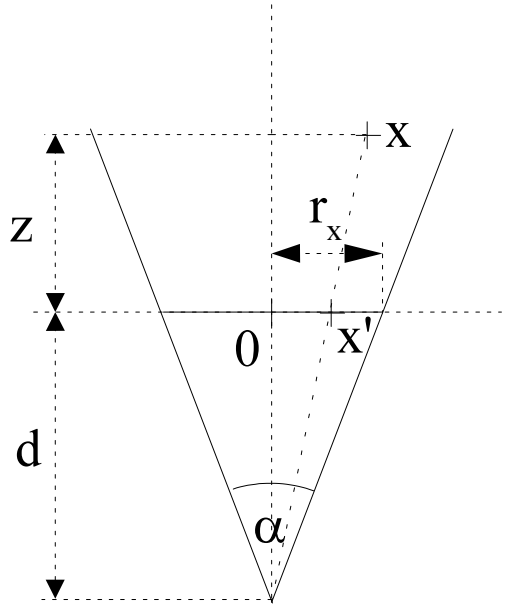


Vlastnosťou tohto typu perspektívy je, že všetky body ležiace na priamke vychádzajúcej z počiatku premieta do jedného bodu. Na obrázku je znázornený priemet dvoch bodov so súradnicami $[x, y, z]$ a $[x_v, y_v, z_v]$ v rovine ***xz***. Keďže ležia na jednej priamke prechádzajúcej počiatkom, mali by sa zobraziť do jedného bodu. Ak chceme zobraziť bod na obrazovku, musíme zistiť, do ktorej časti obrazu patrí. Vidíme, že priamka zachováva proporcie rozdelených častí obrazu. Pre daný bod poznáme jeho ***x***-ovu súradnicu, nepoznáme však rozsah obrazu v danej vzdialenosti. Ak polovicu tohto rozsahu pre daný bod označíme ako ***h*** (je to vzdialenosť od ***z***-ovej osi po viditeľný okraj, kolmá na ***z***-ovu os), môžeme ju vyjadriť ako $h = z \tan\frac{\alpha}{2}$. Ak sa pozrieme na obrázok, zistíme, že pomer $\frac{x}{h}$ a $\frac{x_v}{h_v}$ je rovnaký. Tento pomer je rovnaký pre každý bod na danej priamke a bude pre viditeľné body nadobúdať hodnotu od -1 po 1. Dostali sme teda proporcionálne vyjadrenie polohy bodu v obraze, teraz je ho ešte potrebné dať na obrazovku. Ak označíme polovicu šírky obrazovky ako r_x, potom ***x***-ova súradnica bodu na obrazovke bude $x' = \frac{x}{z} r_x ctg\frac{\alpha}{2} + r_x$. Pre ***y***-ovú súradnicu priemetu platí podobný vzťah, avšak uhol pohľadu je iný. Ak označíme polovicu výšky obrazovky ako r_y, potom uhol pohľadu β pre rovinu ***yz*** môžeme vyjadriť ako $\beta = \alpha \frac{r_y}{r_x} s$, kde ***s*** je korekcia pomeru horizontálneho a vertikálneho rozlíšenia pre dosiahnutie rovnomerného obrazu (využíva sa pri rozlíšeniach, pri ktorých nie je pomer medzi horizontálnym a vertikálnym rozlíšením rovný 4:3 a slúži na jeho dosiahnutie, keďže pri takomto pomere sa bod na obrazovke objaví ako štvorec a nie ako obdĺžnik). Potom pre ***y***-ovú súradnicu platí $y' = \frac{y}{z} r_y ctg\frac{\beta}{2} + r_y$. Ak označíme $k = r_x ctg\frac{\alpha}{2}$ a $l = r_y ctg\frac{\beta}{2}$, vieme, že tieto výrazy sú pri projekcii vždy konštantné. Potom výsledné vzťahy pre projekciu budú

$$x' = k\frac{x}{z} + r_x$$

$$y' = l\frac{y}{z} + r_y$$

Centrálna projekcia vyjadrená pomocou priemetne



Často sa možno stretnúť s projekciou do jedného bodu, ktorý leží na nejakej súradnicovej osi. Tento bod sa nazýva hlavný úbežník a všetky body ležiace na priamke prechádzajúcej úbežníkom sa zobrazia do jedného bodu. Podobne, ako pri predošlej projekcii, môžeme hľadať súradnice v obraze v samostatných rovinách. Ak sa pozrieme na obrázok, zistíme, že obraz všetkých bodov sa premieta na plochu (***priemetňu***), ktorá je umiestnená v počiatku a je kolmá na obe súradnicové roviny (je to vlastne rovina ***xy***). Viditeľná časť je len v okolí bodu **0** do vzdialenosti r_x (čo je opäť polovica šírky obrazovky), čím dostávame rozmery obrazovky. Úbežník leží na osi ***z***, vo vzdialenosti ***d*** od bodu **0**. Body ***x*** a ***x'*** ležia na jednej priamke, preto by sa mali zobraziť do jedného bodu. Bod ***x'*** leží na priemetni (je to vlastne súradnica vo výslednom obraze). Pre tieto dva body musí platiť podobnosť trojuholníkov, $\frac{x}{z+d}=\frac{x'}{d}$. Z toho dostávame $x'=x\frac{d}{z+d}$. Podobne pre ***y***-ovu súradnicu platí $y'=y\frac{d}{z+d}$. Nevýhodou tohto spôsobu premietania je to, že súradnice objektov, ktoré zobrazujeme, nie sú relatívne voči **0**, ale voči bodu $(0,0,-d)$ a pri premietaní teda všetkým bodom musíme zvýšiť ***z***-ovu súradnicu o *d*. Vyriešiť sa to dá aj tak, že posunieme hlavný úbežník a priemetňu o *d* v smere narastajúcej osi ***z***, čím sa úbežník stotožní s bodom **0**. Potom môžeme vyjadriť nové súradnice premietnutých bodov ako $x'=x\frac{d}{z}$ a $y'=y\frac{d}{z}$. Týmto sme dostali ekvivalent ku centrálnej projekcii, keďže $tg\frac{\alpha}{2}=\frac{r_x}{d}$. Samozrejme, výsledné súradnice je potrebné ešte posunúť o šírku r_x a výšku r_y obrazu.

Maticové vyjadrenia perspektívnych transformácií

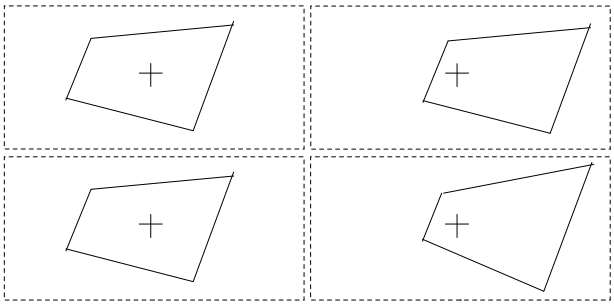
V maticovom tvare by sme tieto tri perspektívne projekcie mohli vyjadriť týmito spôsobmi:

$$\pi_c=\begin{bmatrix} k & 0 & 0 & 0 \\ 0 & l & 0 & 0 \\ r_x & r_y & 1 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \qquad \pi_u\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \frac{1}{d} \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \pi_{cu}=\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \frac{1}{d} \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Pomocou rovnoľahlosti *Pomocou priemetne* *Modifikovaná priemetňa*

Zdalo by sa, že už nič nebráni zobrazeniu trojrozmerných dát na obrazovku, avšak je tu jeden veľký problém. Predchádzajúce perspektívne zobrazenia nie sú správne z hľadiska ľudského vnímania obrazu. Ľudské oko má tendenciu zachovávať uhlové vzdialenosti vnímaných objektov bez ohľadu na natočenie oka, tieto perspektívne zobrazenia však uhly pri zmene pohľadu nezachovávajú. Má to

za následok zvláštne zmeny tvaru objektov pri otáčaní pohľadu bez zmeny polohy objektu a miesta pohľadu. Rozdiel medzi správaním sa projekcií a vnemom ľudského oka je ukázaný na príklade:



Na horných dvoch obrázkoch je obraz, ktorý je podobný vnímaniu oka. Na spodných dvoch je naopak obraz vytvorený projekciou objektov do roviny obrazu. V strede každého obrázku je označený bod, na ktorý sa orientuje pohľad. Ak si porovnáme dva ľavé obrázky, zistíme, že medzi nimi nie je žiadny rozdiel. Vidíme, že v strede sa nachádza akýsi štvoruholník. Ak však otočíme pohľad vľavo, objekt sa na obraze posunie doprava. Dva pravé obrázky ukazujú, ako sa chová vnem oka a perspektívna projekcia. Oko má tendenciu zachovať rozmery a uhly objektu, aj keď sa nenachádza v strede obrazu. Perspektívna transformácia však zdeformuje rozmery telesa. Časti, ktoré sa posunuli bližšie ku stredu obrazu sa zmenšia, tie, ktoré sa vzdialili od stredu, sa zväčšia. Je to práve tým, že každý bod pri perspektívnej transformácii sa snažíme premietnuť do jedného bodu, no pri zmene orientácie pohľadu sa tento bod posúva, objekt sa premieta do tohto nového bodu, a preto sa jeho tvar musí meniť. Podobnú vlastnosť majú aj kamery, ktoré trpia tým, že výsledný obraz sa premieta na rovinu, čo má za následok zväčšenie proporcií obrazu pri väčšej vzdialenosti od stredu. Ľudské oko, naopak, premieta obraz na *zakrivenú* sietnicu, kde sa tieto problémy výrazne zmenšujú. Vnímanie však nie je len záležitosťou samotného oka, ale najmä mozgu, ktorý v súčastnosti stále takmer vôbec nepoznáme.

Doposiaľ nebol vynájdený spôsob zobrazovania, ktorý by riešil tento problém, aj keď niekoľko pokusov sa už vyskytlo. Ako útecha môže poslúžiť fakt, že do 36° uhla pohľadu je centrálna projekcia "veľmi" presná. Problém však vzniká, keď sa pokúsime zobraziť širokoúhle obrazy.

Realistickejší model založený na opise optiky kamery

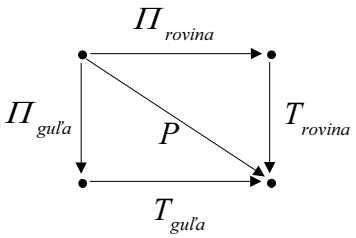
V [Kolb1995] je uvedený spôsob, ako sa viac priblížiť realite vytvorením lepšieho modelu kamery. Doposiaľ používané modely kamery využívali nulovú hrúbku šošovky, z nej potom vychádzali aj vzťahy pre perspektívnu transformáciu. Reálnejší model kamery je založený na opise optiky kamery, na spájaní viacerých šošoviek, ktoré sú **hrubé** (majú nenulovú hrúbku). Obraz vytvárajúci sa šošovkou možno opísať perspektívnou projekciou vzťahmi $x' = x\frac{f'}{z+f'}$, $y' = y\frac{f'}{z+f'}$, pričom hĺbka po prechode šošovkou sa dá vyjadriť ako $z' = \frac{zf' + t(z+f')}{z+f'}$, kde f' je efektívna ohnisková vzdialenosť šošovky a t je efektívna hrúbka šošovky. Pomocou matice možeme transformáciu vyjadriť takto:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1+\frac{t}{f'} & \frac{1}{f'} \\ 0 & 0 & t & 1 \end{bmatrix}$$

Táto projekcia nám tiež umožní vytvoriť efekt zväčšovania a zaostrovania obrazu.

Korekcia geometrických vnemových skreslení v obrazoch

Iný spôsob prístupu k tomuto problému je použitý v [Zorin1995]. Myšlienka je založená na opise vnímania a na minimalizácii štrukturálnych a neštrukturálnych skreslení obrazu. Štrukturálne skreslenia opisujú chyby projekcie, ako sú napríklad prehnutia, skrútenia a slučky vo výslednom obraze, neštrukturálne zase merateľné chyby projekcie. Samotná projekcia je vyjadrená ako zloženie dvoch projekcií - projekcia Π na prechodnú rovinu a zobrazenie T z tejto roviny na rovinu obrazu.



Výslednú projekciu **P** môžeme poskladať napríklad z týchto dvojíc zobazení: Môžeme si vytvoriť projekciu Π_{rovina}, ktorá nám zobrazí priestor do roviny a potom na výsledok aplikovať zobrazenie T_{rovina}, ktorá premietne obraz z prechodnej roviny do roviny obrazu. Alebo vytvoríme projekciu $\Pi_{guľa}$, ktorá premietne priestor dovnútra guľovej plochy, ktorá je spojená so zobrazením $T_{guľa}$. Tá nám premietne guľovú plochu do roviny výsledného obrazu. Oba spôsoby projekcie sú vyjadrené na opis rôznych vlastností obrazu a jeho vzťahu k správnemu vytvoreniu obrazu.

Podmienky správnej projekcie

Podmienky vychádzajú z pozorovaní vnímania obrazu človekom. Prvá podmienka je, aby sa priamky po projekcii zobrazili opäť do priamok. Táto podmienka sa nazýva podmienkou ***zakrivenia***. Ak sa pozrieme na centrálnu projekciu, zistíme, že tejto podmienke vyhovuje. Ak použijeme centrálnu projekciu ako Π_{rovina}, potom zakrivenie výslednej projekcie závisí od zakrivenia T_{rovina}. Vyžadujeme, aby zakrivenie výsledného obrazu bola čo najmenšia. Označíme si druhú mocninu zakrivenia $K(T_{rovina}, x, y)$ zobrazenia T_{rovina} v bode x,y(zakrivenie vychádza zo zakrivenia čiary v bode).

Druhá podmienka sa nazýva podmienkou *priameho pohľadu*. Spočíva v tom, že projekcia bodov obrazu musí byť podobná v každom bode projekcii vlákna projekcie prechádzajúceho daným bodom na rovinu kolmú na toto vlákno. Ide tu vlastne o to, aby sa obraz zobrazil rovnako na každom mieste výsledného obrazu. Túto vlastnosť má v určitom uhlovom okolí projekcia $\Pi_{guľa}$. Preto nás zaujíma veľkosť ***nepriamosti*** zobrazenia $T_{guľa}$. Môžeme preto vytvoriť mieru nepriamosti $D(T_{guľa}, \phi, \psi)$, kde ϕ, ψ sú súradnice na prechodnej guľovej ploche. Pre ďalšiu prácu ju však môžeme vyjadriť v súradniciach zobrazenia T_{rovina}. Dostávame potom $D(T_{rovina}, x, y)$, čo nám ukazuje mieru nepriamosti bodu na súradnici x, y v obraze. Túto mieru nepriamosti si môžeme predstaviť ako zmenu proporcií gule premietnutej do obrazu, kde by sa mala zobraziť ako kruh. Rozdiel pomerov osí v zobrazenom elipsoide a čísla 1 by mohol charakterizovať mieru nepriamosti. Predpokladáme, že ak premietneme guľu na guľovú plochu, získame obraz disku, a ak tento premietneme na rovinu obrazu, mali by sme dostať kruh.

Označíme celkové zakrivenie $K(T_{rovina})$ a celkovú nepriamosť $D(T_{rovina})$, ktoré môžu byť napríklad maximom zo všetkých hodnôt alebo integrálom po celom obraze. Potom nám ostáva nájsť ich minimum. Úloha sa zmení na hľadanie minimalizujúcich funkcionálov.

$$\begin{aligned} &minimalizácia\, K(T_{rovina}) = \nu && minimalizácia\, D(T_{rovina}) = \mu \\ &s\ podmienkou\, D(T_{rovina}) = \mu && s\ podmienkou\, K(T_{rovina}) = \nu \end{aligned}$$

V prvom prípade získame minimalizujúci funkcionál $\nu(\mu)$, v druhom $\mu(\nu)$.
Môžeme sa zbaviť obmedzujúcich podmienok tak, že vytvoríme konvexnú kombináciu K a D. Potom sa minimalizácia zmení na

$$minimalizácia\, F = \lambda K(T_{rovina}) + (1-\lambda) D(T_{rovina}), \quad \lambda \in \langle 0,1 \rangle$$

Riešenie týchto problémov je veľmi zložité, preto je lepšie použiť aproximáciu výsledku. Hodnoty λ sa môžu voľne pohybovať v danom intervale, môžu byť zvolené podľa potreby a líšiť sa v každom bode obrazu.

Použitie v programe

Implementácia je pomerne jednoduchá. Je zložená z aplikovania centrálnej projekcie s ľubovoľným stredom a transformácie vzniknutého obrazu na korigovaný obraz. Predpokladáme, že obraz má stred v bode (0, 0).

Druhá časť transformácie sa dá vyjadriť pomocou algoritmu:

Pre každý bod [i, j] obrazu vypočítame $r = \sqrt{i^2 + j^2}$ a bodu priradíme farbu $\textit{interpolovaná farba}\left(\rho^{-1}(r)\frac{i}{r}, \rho^{-1}(r)\frac{j}{r}\right)$. *Interpolovaná farba* sa vypočíta napríklad priemerovaním alebo filtrovaním.

Je potrebné vyjadriť inverznú funkciu k funkcii ρ. Tú môžeme vyjadriť napríklad pomocou dvoch interpolácií - *geometrickej* a *lineárnej*.

Geometrická interpolácia

Pre geometrickú interpoláciu platí vzťah

$$\rho^{-1}(r) = tg(2-\lambda)\,arctg\left(tg\left(arctg\frac{R}{(2-\lambda)}\right)\frac{r}{R}\right),$$

kde *R* je konštanta zväčšenia.

Ak vyžadujeme, aby λ bola rôzna v obraze, napríklad aby závisela od uhla ϕ, musíme vypočítať inverznú funkciu k funkcii

$$\rho(r) = R\frac{tg\left(\frac{arctg\ r}{2-\lambda(\phi)}\right)}{tg\left(\frac{arctg\,R}{2-\lambda(\phi)}\right)}$$

Lineárna interpolácia

Pre lineárnu interpoláciu platí vzťah

$$\rho(r) = \lambda\frac{r}{R} + (1-\lambda)\frac{R\left(\sqrt{r^2+1}-1\right)}{r\left(\sqrt{R^2+1}-1\right)}$$

Hodnotu inverznej funkcie môžeme vyjadriť pomocou zvolenej numerickej metódy.

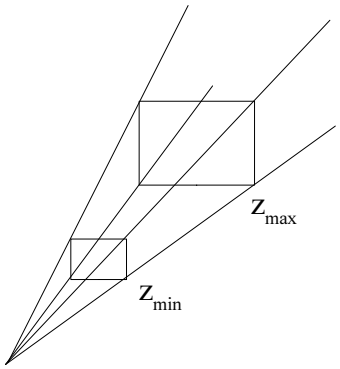
Pri vyžadovanej závislosti λ na ϕ sa výraz zmení na

$$\rho(r) = \lambda(\phi)\frac{r}{R} + (1-\lambda(\phi))\frac{R\left(\sqrt{r^2+1}-1\right)}{r\left(\sqrt{R^2+1}-1\right)}.$$

Výpočet je však veľmi komplikovaný, preto je lepšie využiť geometrickú interpoláciu.

Viditeľná oblasť

Centrálna a úbežníková projekcia v priestore vymedzia oblasť, ktorá je viditeľná. Keďže viditeľná oblasť má v rovine ***xz*** ako i v rovine ***yz*** tvar neuzavretého trojuholníka, v priestore nadobúda tvar neuzavretého štvorbokého ihlanu. Všetky body, ktoré patria do jeho vnútra, budú po projekcii ležať vo vnútri obrazu. Pri takejto viditeľnej oblasti však nastane problém pri výpočte perspektívnej projekcie pri bodoch, ktoré sú príliš blízko stredu premietania (úbežníku). Vtedy by pri delení hĺbkou mohla nastať chyba výpočtu pre príliš veľký výsledok, ktorý sa nezmestí do reprezentácie čísel. Preto sa zvykne pridať blízko stredu premietania rovina, ktorá je kolmá na os ***z*** vo vzdialenosti z_{min}. Táto rovina budeurčovať minimálnu možnú ***z***-ovu súradnicu bodu, ktorý bude ešte viditeľný. Z podobných pohnútok sa zvykne pridávať ďalšia rovina kolmá na os ***z*** do vzdialenosti z_{max}. Jej úlohou je určovať najväčšiu možnú súradnicu ***z*** viditeľných bodov. Jej pridanie je vhodné kvôli chybovosti numerických výpočtov, ktoré sa zvyknú silne prejaviť pri vyšších hodnotách, a taktiež kvôli obmedzeniu viditeľného priestoru kvôli šetreniu pamäťou. Druhý aspekt je využívaný vtedy, keď sa zobrazuje rozľahlý priestor s veľkým počtom objektov, ktorý je však rozdelený na určité časti. Potom stačí udržovať informácie iba o objektoch, ktoré sa nachádzajú vnútri a v bezprostrednom okolí aktuálnej časti priestoru. Kvôli tomu sa zvykne obmedziť dosah pohľadu na minimálny rozmer tejto časti.



Zobrazenie na obrazovke

Súradnice získané po perspektívnom premietaní nemôžeme priamo použiť na vykresľovanie na obrazovku. Výsledný obraz má síce rozmery totožné s veľkosťou obrazovky, avšak ak by sme sa pokúsili vykresliť body, zistili by sme, že obraz je na obrazovke "hore nohami". Súvisí to s tým, že pri všetkých predošlých vzťahoch os ***y*** rástla odspodu nahor, zatiaľ čo na obrazovke to býva zväčša naopak. Vyriešime to tak, že od zvislého rozlíšenia obrazovky odpočítame ***y***-ovu súradnicu bodu, čím dostaneme novú súradnicu ***y***, ktorá bude správne orientovaná. Ak ***v*** je zvislý rozmer (rozlíšenie) obrazovky, potom nová súradnica ***y*** bude mať tvar $y = v - y'$, kde y' je ***y***-ova súradnica bodu, získaná po perspektívnej transformácii. Tento vzťah však môžeme včleniť priamo do matice projekcie, a to tak, že zinvertujeme prvok, ktorý určuje veľkosť y-ovej súradnice. Pri centrálnej projekcii by sme dostali maticu

$$\pi_c = \begin{bmatrix} k & 0 & 0 & 0 \\ 0 & -l & 0 & 0 \\ r_x & r_y & 1 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Vykresľovanie mnohouholníkov

Veľmi dôležitou súčasťou programov pre zobrazovanie trojrozmerných objektov je vykresľovanie mnohouholníkov. Zväčša ide o časovo najnáročnejšiu činnosť, preto sa hľadajú spôsoby ako čo najrýchlejšie vykresľovať mnohouholníky a zároveň ako vykresľovať len tie, ktoré sú skutočne viditeľné. Taktiež sa hľadajú spôsoby, ako obmedziť vzájomné prekrývanie mnohouholníkov a opätovné prekresľovania už zakreslenej oblasti.

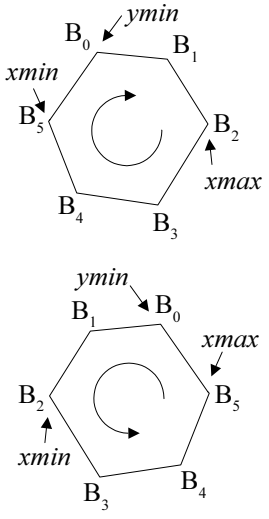
Často používaný a efektívny spôsob vykresľovania mnohouholníkov je založený na ***rozklade mnohouholníka na riadky***. Mnohouholník prechádzame po jednotlivých riadkoch, zistíme si začiatočný a koncový bod riadku a v vykreslíme vodorovnú čiaru medzi týmito bodmi. To, že je tento spôsob veľmi rýchly spočíva v tom, že video pamäť počítača je zväčša organizovaná po riadkoch. Znamená to, že dve následujúce adresy vo video pamäti ukazujú na dva body v susedných stĺpcoch v tom istom riadku (pokiaľ nie je jeden na konci jedného a druhý na začiatku druhého riadku) a vykresľovanie riadku môžeme nahradiť jednou inštrukciou zapisujúcou viacero bytov do pamäti.

Vykresľovanie mnohouholníka riadkovým rozkladom

Nájdeme bod v mnohouholníku, ktorý je najvyššie (na obrazovke má minimálnu ***y***-ovu súradnicu) a dva jeho susedné body, jeden ľavý a jeden pravý. Dostávame dve úsečky - ľavú a pravú. Vypočítame rozdiel ***x***-ovej súradnice pri zmene o jeden riadok v oboch úsečkách (sú to hodnoty, ktoré v každom riadku budeme pripočítavať ku ***x***-ovej súradnici ľavého (začiatočného) a pravého (koncového) bodu). V riadku vykreslíme vodorovnú čiaru medzi týmito dvomi bodmi a posunieme sa na ďalší riadok. Ku bodom potom pripočítame rozdiely ***x***-ovej súradnice. Ak sme sa dostali na koniec jednej z dvoch úsečiek, zoberieme ďalšieho suseda koncového bodu úsečky, čím dostaneme novú, nadväzujúcu úsečku. Taktiež pre ňu vypočítame rozdiel ***x***-ovej súradnice pri zmene o jeden riadok. Toto všetko opakujeme, kým nevykreslíme posledný riadok mnohouholníka.

Algoritmus

Algoritmus vyžaduje, aby sa v zozname bodov mnohouholníka vedľa seba nachádzali navzájom susedné body. Toto je možné dosiahnuť iba tak, že body mnohouholníka usporiadame v smere alebo proti smeru hodinových ručičiek. Body si označíme ako $\boldsymbol{B}_i,\ i = 0..n-1$, kde *n* je počet bodov mnohouholníka a majú tvar ***B*** = *(x, y, V)*, kde *V* sú nejaké ďalšie vlastnosti bodu.

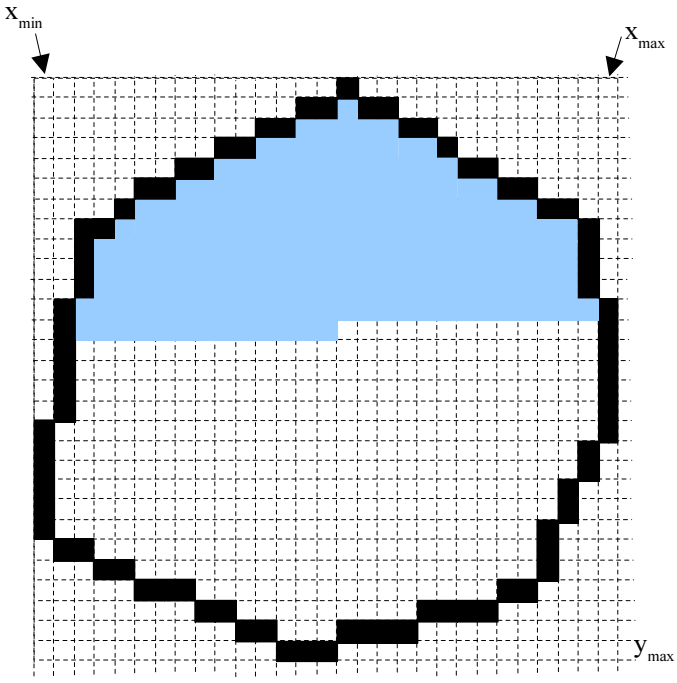


1. Nájdeme index *ymin* vrchola, ktorý má minimálnu hodnotu súradnice ***y***. Nájdeme index *xmin* vrchola s najmenšou a index *xmax* vrchola s najväčšou hodnotou súradnice ***x***. Pomocou týchto troch vrcholov zistíme najskôr, či sú body usporiadané v smere alebo proti smeru hodinových ručičiek, a potom zistíme, ktorý susedný vrchol máme zobrať, aby sme mohli vytvoriť ľavú a pravú hraničnú úsečku, vychádzajúcu z vrchola $\boldsymbol{B}_{ymin}$. Zistíme to tak, že ak vrcholová vzdialenosť (vzdialenosť medzi bodmi v poli so zväčšujúcim sa indexom) medzi bodmi $\boldsymbol{B}_{ymin}$ a $\boldsymbol{B}_{xmin}$ je väčšia ako vzdialenosť medzi $\boldsymbol{B}_{xmax}$ a $\boldsymbol{B}_{ymin}$, potom sú body v zozname usporiadané v smere hodinových ručičiek, inak sú usporiadanú proti smeru. Ak sú usporiadané v smere hodinových ručičiek, potom pravý

susedný bod je $\boldsymbol{B}_{[\,ymin+1\,]}$ a ľavý $\boldsymbol{B}_{[\,ymin-1\,]}$ ([*i*] je operácia, ktorá určuje index v poli tak, aby bol vždy v rozmedzí *0..n-1*, [*i*] = *i* mod *n*). Ak sú body usporiadané proti smeru hodinových ručičiek, pravý susedný bod je $\boldsymbol{B}_{[\,ymin-1\,]}$ a ľavý $\boldsymbol{B}_{[\,ymin+1\,]}$. Vrcholová vzdialenosť je počet vrcholov, ktoré ležia medzi dvomi vrcholmi, avšak nie najmenšia, ale cyklicky v smere zväčšujúceho sa indexu v poli bodov. Vytvoríme si premennú *smer*, ktorá nám bude určovať smer pohybu po vrcholoch v poli, aby tento pohyb bol v smere hodinových ručičiek (bude *1*, ak body sú zoradené v smere hodinových ručičiek, *-1* ak proti smeru). Ďalej si vytvoríme štyri premenné, l^z, l^k, p^z a p^k, ktoré budú určovať indexy začiatočných a koncových bodov ľavej a pravej úsečky. Na začiatku priradíme $l^z = p^z = ymin$, $l^k = xmin$ a $p^k = xmax$. Pridáme ešte premennú *y*, ktorá v sebe obsahuje číslo aktuálneho riadku, $y = \boldsymbol{B}[\,ymin\,] \rightarrow y$ (*y* dáme rovné y-ovej súradnici bodu s minimálnou súradnicou ***y***). Nájdeme ešte y_{max}, maximálnu ***y***-ovú súradnicu bodu.

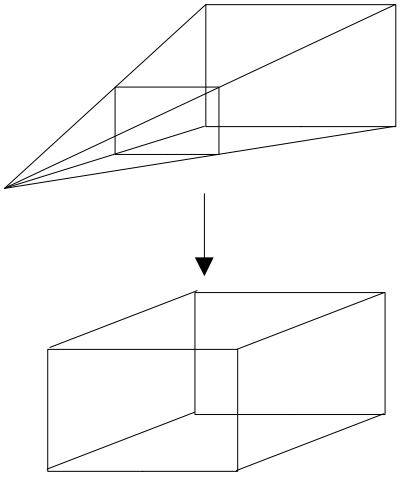
2. Vypočítame $l^d = \dfrac{\boldsymbol{B}[\,l^k\,] \rightarrow x - \boldsymbol{B}[\,l^z\,] \rightarrow x}{\boldsymbol{B}[\,l^k\,] \rightarrow y - \boldsymbol{B}[\,l^z\,] \rightarrow y}$ a $p^d = \dfrac{\boldsymbol{B}[\,p^k\,] \rightarrow x - \boldsymbol{B}[\,p^z\,] \rightarrow x}{\boldsymbol{B}[\,p^k\,] \rightarrow y - \boldsymbol{B}[\,p^z\,] \rightarrow y}$, ktoré nám hovoria, o koľko sa mení ***x***-ova súradnica ľavej a pravej úsečky pri posune na ďalší riadok. Ďalej vytvoríme premenné x^l a x^p, ktoré budú určovať ľavý a pravý okraj mnohouholníka v ***y***-tom riadku ($x^l = \boldsymbol{B}[\,l^z\,] \rightarrow x$ a $x^p = \boldsymbol{B}[\,p^z\,] \rightarrow x$).

3. Vykreslíme vodorovnú čiaru v riadku *y* od bodu x^l po bod x^p a priradíme $x^l = x^l + l^d$ a $x^p = x^p + l^p$. Ak $y = \boldsymbol{B}[\,l^k\,] \rightarrow y$, znamená to, že sme sa dostali na koniec ľavej úsečky a musíme nájsť jej nové koncové vrcholy. Začiatočný bod novej úsečky je totožný s koncovým bodom predošlej úsečky, $l^z = l^k$. Musíme si však dať pozor, aby sme nevybrali dva vrcholy s rovnakou ***y***-ovou súradnicou, pretože by sme pri zisťovaní l^d delili nulou. Tomuto sa vyhneme, pokiaľ sa budeme pohybovať doľava dovtedy, kým nenatrafíme na vrcholy s rôznou ***y***-ovou súradnicou (ľavý susedný vrchol má index [l^k - *smer*], pravý [p^k + *smer*]). Podobne postupujeme aj vtedy, ak sme sa dostali na koniec pravej úsečky. Pre novú úsečku opäť vypočítame l^d alebo p^d. Zvýšime *y* o *1*. Tento krok opakujeme dovtedy, kým $y \leqslant y_{max}$.



Orezávanie mnohouholníkov

Pri zobrazovaní mnohouholníkov sa stáva, že samotný mnohouholník vyčnieva z viditeľnej oblasti. Preto je nutné zmeniť tvar mnohouholníka tak, aby opisoval iba jeho časť, ktorá zasahuje do viditeľnej oblasti.



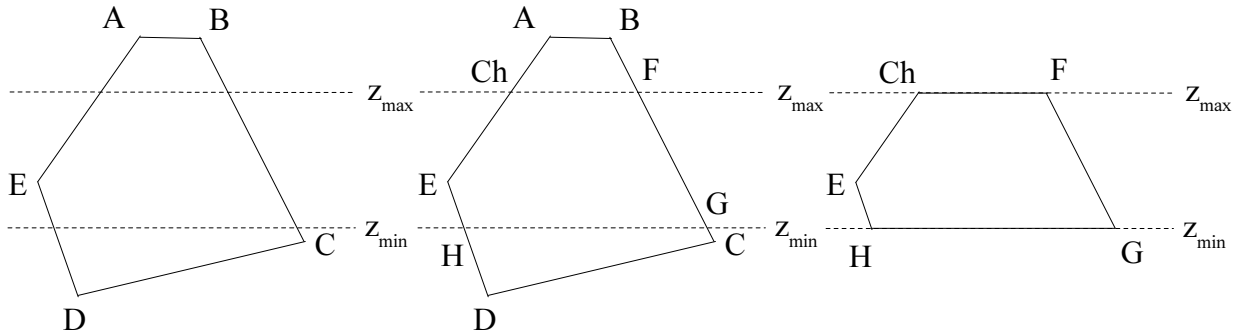
Pri orezávaní mnohouholníkov môžeme postupovať tak, že mnohouholník postupne orezávame s každou rovinou, ktorá ohraničuje viditeľnú oblasť. Toto by však bolo dosť pracné pri orezávaní voči ľavej, pravej, dolnej a hornej rovine, pretože tieto nie sú rovnobežné so súradnicovými osami. Preto môžeme využiť tú skutočnosť, že po perspektívnej transformácii sa viditeľná oblasť transformuje z tvaru zrezaného ihlana na tvar kvádra. Avšak pred perspektívnou transformáciou musíme zaistiť, aby sa v mnohouholníku vyskytovali iba body, ktorých z-ova súradnica leží medzi prednou a zadnou rovinou. Preto orezávanie môžeme rozdeliť na dve časti. Prvá spočíva v orezaní mnohouholníka v pôvodnom priestore vzhľadom na súradnicu z. Druhá časť sa vykonáva po perspektívnej projekcii a mnohouholník sa orezáva vzhľadom na okraje obrazovky (súradnice x' a y').

Hĺbkové orezávanie

Prvé orezávanie spočíva v nájdení okrajových úsečiek mnohouholníka, ktorých jeden bod sa nachádza nad a druhý pod prednou (zadnou) rovinou. Ak takúto úsečku nájdeme, potom ju môžeme rozdeliť na dve úsečky, pretože do mnohouholníka pridáme medzi dva krajné body úsečiek nový bod, ktorý je priesečníkom deliacej roviny a úsečky. Toto budeme vykonávať dovtedy, pokiaľ nenájdeme všetky takéto body. Potom jednoducho všetky body, ktoré sú svojou hĺbkou mimo viditeľnej oblasti z mnohouholníka vyberieme. Ak $\boldsymbol{A} = (A_x, A_y, A_z)$ je prvý a $\boldsymbol{B} = (B_x, B_y, B_z)$ druhý krajný bod úsečky, z_r je vzdialenosť deliacej roviny od počiatku, potom súradnice priesečníka $\boldsymbol{P}$ zistíme ako

$$\boldsymbol{P} = \boldsymbol{A} + \frac{z_r - A_z}{B_z - A_z}(\boldsymbol{B} - \boldsymbol{A}) \quad .$$

(Predpokladáme, že mnohouholník je popísaný vzájomne susediacimi bodmi)



Môžeme si všimnúť, že z pôvodného mnohouholníka ***ABCDE*** vznikol mnohouholník ***FGHECh***.

Orezávanie po perspektívnej transformácii

Perspektívnou transformáciou dosiahneme to, že body, ktoré majú ležať vo viditeľnej oblasti majú teraz súradnice v rozmedzí 0..r_x a 0..r_y, kde r_x je šírka a r_y je výška obrazu. Potom môžeme postupovať takmer úplne zhodne s predchádzajúcim postupom, avšak ak aj po perspektívnej transformácii potrebujeme vyjadriť hĺbku bodu (napríklad pri použití pamäti hĺbok), nemôžeme ju vyjadriť jednoducho ako pri predošlom orezávaní, pretože perspektívnou transformáciou sa stráca linearita hĺbky.

Pri opise pamäti hĺbok je uvedené, že $\frac{1}{z}$ sa v tomto priestore chová lineárne, a teda môže byť využité pri zisťovaní hĺbky priesečníka. Ak máme dva premietnuté body $\boldsymbol{A'} = (A_x', A_y', A_z)$ a $\boldsymbol{B'} = (B_x', B_y', B_z)$ a úsečka $\boldsymbol{AB}$ pretína napríklad priamku $\boldsymbol{x'}$ = r , potom súradnice priesečníka $\boldsymbol{P'} = (P_x', P_y', P_z)$ môžeme vyjadriť ako

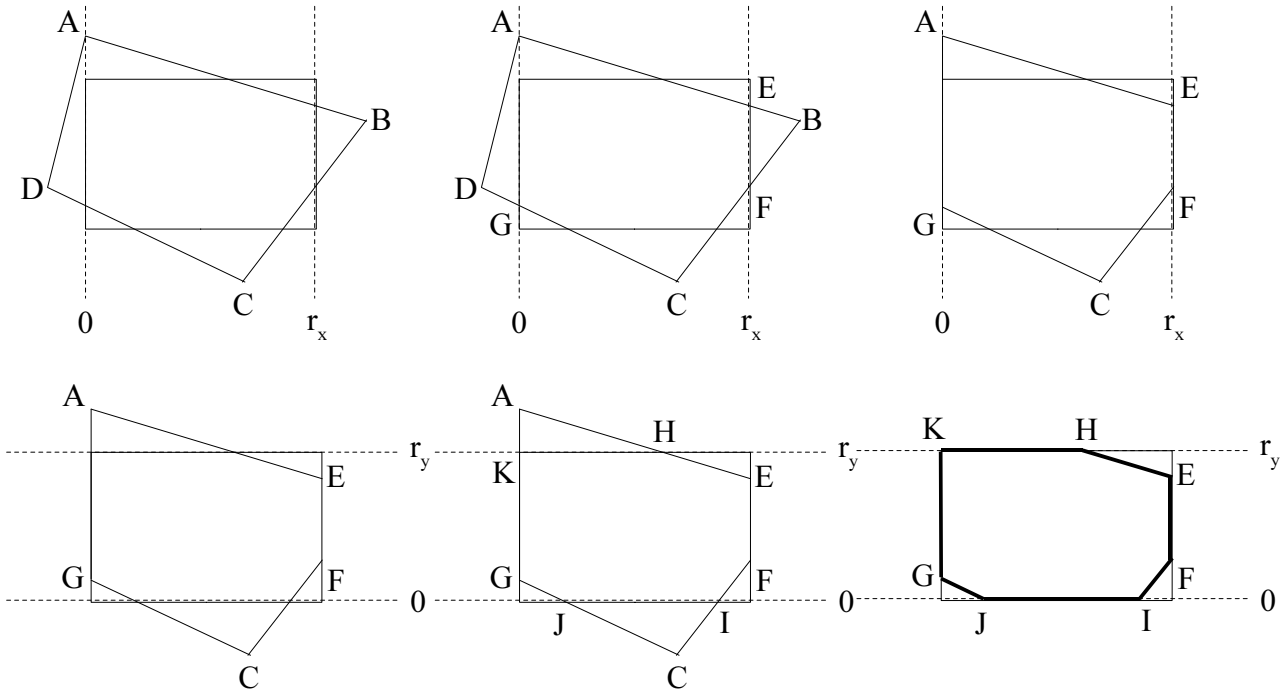
$$P_x' = r$$

$$P_y' = A_y' + \frac{r - A_x'}{B_x' - A_x'}(B_y' - A_y')$$

$$P_z = \frac{B_z A_z (B_x' - A_x')}{B_z (B_x' - A_x') + (A_z - B_z)(r - A_x')}$$

Podobné vzťahy platia aj pre orezávanie vzhľadom na súradnicu $\boldsymbol{y'}$.

Pri orezávaní postupujeme podobne ako pri predošlom orezávaní. Najskôr orezávame podľa jednej zo súradníc a až po nej podľa druhej. Na konci by sme mali dostať mnohouholník, ktorého vrcholy určite ležia vo viditeľnej oblasti a ktorý na obrazovke zaberá správny výrez.



Tieňovanie

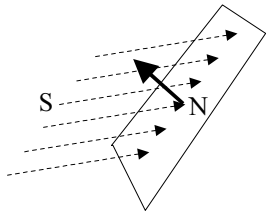
Ak chceme, aby sa obraz objektu aspoň trochu približoval skutočnosti, nemali by sme zabudnúť na svetlo a jeho premeny, ktoré vytvára na povrchu objektu. Môžeme si všimnúť, že priamo osvetlená strana telesa je svetlejšia ako tá, ktorá je šikmejšie otočená voči svetlu, či ako tá, ktorá nie je osvetlená vôbec. Tieňovanie je spôsob, ako sa priblížiť zmenám intenzity svetla na povrchu objektu. Nejde však o vrhanie tieňov na iné objekty, len o správanie sa svetla na povrchu telesa, akoby okolo neboli žiadne iné objekty.

Najčastejšie sa možno stretnúť s týmito spôsobmi **tieňovania** (vo všetkých sa pracuje s plošným svetlom):

1. *Plošné tieňovanie*
2. *Lineárna interpolácia farby*
3. *Interpolácia normály*

Plošné tieňovanie

Založené je na výpočte intenzity farby pre celú plochu mnohouholníka. Túto intenzitu získame tak, že zistíme, aký uhol zviera normála roviny mnohouholníka s vektorom osvetlenia.



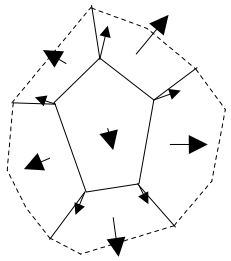
Ak plocha mnohouholníka má normálu $\boldsymbol{N}$ a svetlo sa šíri v smere určenom vektorom $\boldsymbol{S}$ a tieto vektory majú jednotkovú dĺžku, potom uhol, ktorý zvierajú, využijeme na zistenie intenzity svetla. Keďže $\boldsymbol{N} \cdot \boldsymbol{S} = \cos\ \alpha$, kde α je uhol týchto vektorov, využijeme to, že hodnoty cos α sú v intervale $\langle -1\,,\,1 \rangle$, pričom menšie ako 0 sú pri α v intervale $\left(\frac{\pi}{2}\,,\,\frac{3\pi}{2}\right) + 2k\pi\,,\,k \in Z$. Ak svetlo osvetľuje plochu z viditeľnej strany, potom $\boldsymbol{N} \cdot \boldsymbol{S} \leq 0$ a absolútnu hodnotu tohto súčinu môžeme použiť ako intenzitu svetla na povrchu mnohouholníka. Vyplýva to z toho, že $|\boldsymbol{N} \cdot \boldsymbol{S}| \in \langle 0\,,\,1 \rangle$ a z toho, že $|\boldsymbol{N} \cdot \boldsymbol{S}| = 1$ vtedy, keď normála roviny smeruje presne oproti svetlu. Plocha je potom kolmo orientovaná na dopadajúce svetlo a intuitívne sa dá predpokladať, že práve vtedy by mala byť intenzita osvetlenia plochy najvyššia. Podobne, $|\boldsymbol{N} \cdot \boldsymbol{S}| = 0$ vtedy, keď svetlo je kolmé na normálu roviny, potom neosvetľuje rovinu vôbec, keďže je s ňou rovnobežné. Ak svetlo osvetľuje plochu iba z neviditeľnej strany, potom pre viditeľnú stranu môžeme zvoliť intenzitu rovnú 0. Mnohokrát však chceme, aby intenzita svetla aj na neosvetlenej rovine bola nenulová. Ak sa poobzeráme okolo seba, zistíme, že predmety sú osvetlené aj na stranách, ktoré sú ovrátené od svetla. Tie, ktoré sú privrátené, majú vyššiu intenzitu svetla, ktorá sa mení podľa polohy oproti svetlu, tie, ktoré sú odvrátené majú intenzitu podobnú (môžeme ju brať ako rovnakú). Ak si zvolíme minimálnu hodnotu intenzity I_{min} , potom výpočet intenzity svetla $\boldsymbol{I}$ daného vektorom $\boldsymbol{S}$ pre rovinu mnohouhlníka s normálou $\boldsymbol{N}$ môžeme vyjadriť ako

$$\boldsymbol{I} = \begin{cases} |\boldsymbol{N} \cdot \boldsymbol{S}| * (\boldsymbol{1} - I_{min}) + I_{min} & \boldsymbol{N} \cdot \boldsymbol{S} \leq 0 \\ I_{min} & \boldsymbol{N} \cdot \boldsymbol{S} > 0 \end{cases}$$

Výslednú farbu $\boldsymbol{F}$ mnohouholníka v lineárnych farebných modeloch vypočítame prenásobením pôvodnej farby mnohouholníka $\boldsymbol{f}$ intenzitou farby $\boldsymbol{I}$, $\boldsymbol{F = f\,I}$

Lineárna interpolácia farby

Predchádzajúci spôsob tieňovania priraďoval celej ploche rovnakú farbu. Lineárna interpolácia farby nenecháva celú plochu rovnakej farby, ale sa snaží interpolovať farbu vnútri plochy. Opäť sa predpokladá, že celý mnohouholník je jednofarebný. Postup spočíva v tom, že sa vyjadria intenzity farby v krajných bodoch mnohouholníka, vypočítajú sa v nich farby (farba mnohouholníka krát intenzita svetla v bode) a mnohouholník sa vykresľuje s interpolovanými farbami vo vnútri mnohouholníka.



Kľúčovým v tomto spôsobe tieňovania je zistenie intenzity farby v krajných bodoch mnohouholníka. Pre jej zistenie sa najviac využíva priemer z normál rovín mnohouholníkov, do ktorých daný bod patrí. Pre daný bod sa vyberú mnohouholíky, do ktorých patrí, sčítajú sa ich normály, podelia sa ich počtom, výsledný vektor sa znormalizuje a priradí bodu. Intenzita farby a farba bodu sa vypočíta rovnako ako pri plošnom tieňovaní. (na obrázku sú normály rovín označené veľkými šípkami, vrcholov malými).

Nakoniec získavame farbu každého bodu. Ďalším problémom je spôsob vykresľovania mnohouholníka. Ak využijeme kreslenie mnohouholníka riadkovým rozkladom, potom dostaneme nielen začiatočnú x_z a koncovú x_k súradnicu v riadku, ale taktiež aj začiatočnú f_z a koncovú f_k farbu v riadku. Ako teda prejsť od jednej farby k druhej ? Lineárna interpolácia farby je založená na tom, že ku všetkým zložkám začiatoènej farby pripočítava postupne určitú konštantu tak, aby sa na konci riadku z nej stala koncová farba. Pre i-tu zložku farby bude táto konštanta $d_f(i) = \frac{f_k(i) - f_z(i)}{x_k - x_z}$.

Potom pre zložky začiatočnej farby platí $\boldsymbol{f(i)} = f_z(i) + 0\ d_f(i) = \boldsymbol{f_z(i)}$, pre zložky koncovej $\boldsymbol{f(i)} = f_z(i) + (x_k - x_z)\, d_f(i) = f_z(i) + f_k(i) - f_z(i) = \boldsymbol{f_k(i)}$. Algoritmus vykresľovania vodorovnej čiary mnohouholníka teda môžeme vyjadriť takto:

1. Vypočítanie hodnôt $d_f(i)$ pre každú zložku farby, nastavenie hodnoty zložiek aktuálnej farby f(i) na $f_z(i)$. Aktuálna pozícia x bude x_z .
2. Vykreslenie bodu s farbou f. Pripočítanie rozdielov zložiek $d_f(i)$ ku zložkám f(i). Pokiaľ x je menšie alebo rovné x_k , opakujeme vykresľovanie.

Pri vykresľovaní mnohouholníka je ešte potrebné, aby sa pre každú okrajovú úsečku vypočítala zmena farby pri posune o jeden riadok, a táto sa každý riadok pripočítavala ku aktuálnej farbe (ekvivalent ku zmene x-ovej súradnice na jeden riadok).

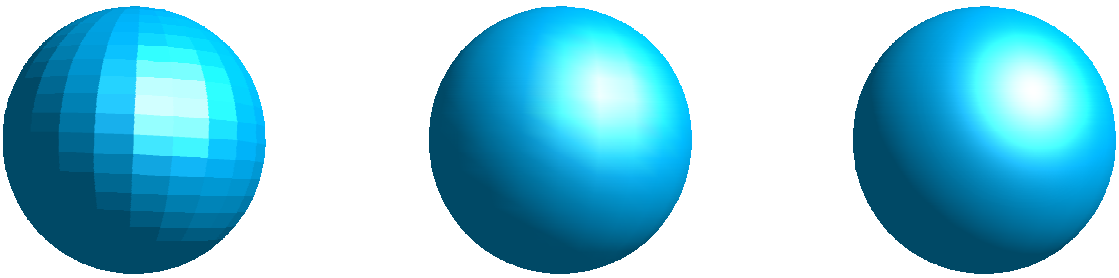
Ak používame farebnú paletu, potom miesto zložiek farby použijeme indexy farieb. Avšak paleta musí byť pripravená na použitie pri tieňovaní (musí byť rozdelená do súvislých blokov postupne klesajúcich alebo stúpajúcich odtieňov základných farieb - napríklad za sebou následujúcich 25 odtieňov červenej...).

Nevýhoda tohto spôsobu tieňovania spočíva v tom, že farba na ploche nemôže nikdy zvýšiť svoju intenzitu, pretože je viazaná na počiatočnú a koncovú farbu a pohybuje sa len v intervale medzi týmito dvomi farbami. Ak však na jednom konci úsečky je uhol medzi normálou plochy a svetlom menší ako 180 stupòov a na druhom väčší, intenzita bude na oboch koncoch menšia ako maximálna a farba medzi týmito bodmi bude mať intenzitu medzi nimi, aj keď v tomto prípade by sme mohli prepokladať, že intenzita bude chvíľku rásť, a potom klesať.

Interpolácia normály

Tento nedostatok sa snaží odstrániť tieňovanie pomocou interpolácie normál. Miesto zložiek farby medzi dvomi bodmi sa snažíme interpolovať zložky normálových vektorov. Normálové vektory v bodoch vypočítame rovnako ako pri lineárnej interpolácii farieb. Na interpoláciu normálového vektoru v riadku môžeme použiť lineárnu interpoláciu. Potom pri kreslení mnohouholníka budeme pripočítavať ku počiatočnej normále vektor prírastkov normály, ktorý je zvolený tak, aby sme na konci riadku dostali koncový normálový vektor. Ak $\boldsymbol{M}$ je počiatočná normála a $\boldsymbol{N}$ je koncová normála v riadku, vektor prírastkov $\boldsymbol{d}$ vypočítame ako $\boldsymbol{d} = \frac{\boldsymbol{M} - \boldsymbol{N}}{x_k - x_z}$.
Postupujeme potom podobne ako pri lineárnej interpolácii farby s tým rozdielom, že pri vykresľovaní bodu vypočítame intenzitu a pomocou nej a farby mnohouholníka vypočítame farbu v danom bode. Lineárna interpolácia normály je síce pomerne rýchla, avšak dosť nepresná, pretože veľkosť normálového vektora vnútri plochy už nemusí byť jednotková. Preto sa používa modifikácia lineárnej interpolácie intenzity, kde sa pri výpočte intenzity normálový vektor znormuje a takto sa použije na jej výpočet.

Pri používaní tieňovania pomocou interpolácií farby a normály si treba uvedomiť, že objekt, ktorý sa snažíme tieňovať musí byť rozložený na vzájomne súvisiace časti z hľadiska tieňovania, inak sa stane, že pri vypočítavaní normály v bode zahrnieme do výpočtu aj normály plôch, ktoré nemajú žiaden súvis s tieňovaním. Napríklad pri tieňovaní valca, ktoré by sme použili kvôli okrúhlemu obalu, by sa nám do výpočtu normál zamiešali aj obe podstavy, čo však pre dané tieňovanie nemá žiaden zmysel, keďže tieto plochy nemajú priamy súvis s intenzitou svetla na obale.



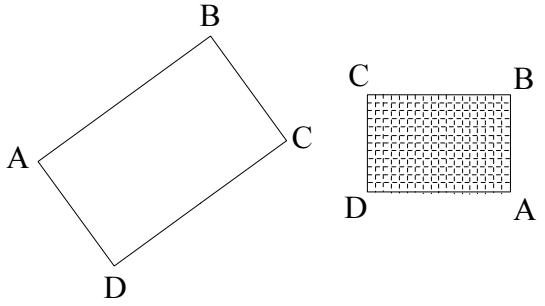
Plošné tieňovanie (Lambertovo tieňovanie)

Lineárna interpolácia farby (Gouraudovo tieňovanie)

Interpolácia normály (Phongovo tieňovanie)

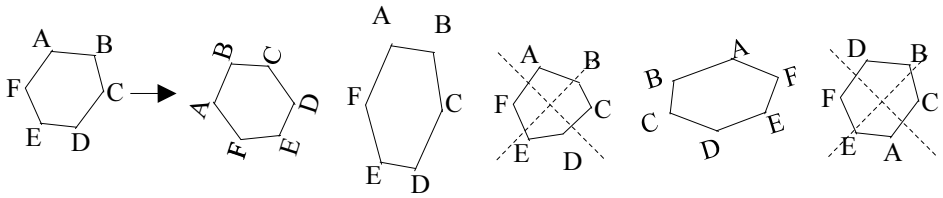
Štruktúra povrchu

Časti telies, zložené z mnohouholníkov vykresľovaných jednou farbou alebo jednofarebne tieňovaných nevyzerajú veľakrát reálne. Ak sa pozrieme na ľubovoľný predmet, u drvivej väčšiny môžeme pozorovať na ich povrchu nejakú farebnú štruktúru. Preto sa namiesto tieňovania alebo v spojení s ním priradí mnohouholníku obrázok štruktúry jeho povrchu (***textúru***). Dosiahne sa tým niekedy realistickejší výzor telesa, ale pribudne aj niekoľko väčších problémov.



Ako priradiť štruktúru mnohouholníku ?

Každému bodu mnohouholníka môžeme priradiť bod, ktorý leží v štruktúre povrchu. Na ľavom obrázku je nakreslený mnohouholník ***ABCD***, ku ktorého bodom sú v štruktúre napravo priradené štyri body. Bez ohľadu na to, aký tvar alebo sklon nadobúda mnohouholník, body v štruktúre sú pevne priradené bodom mnohouholníka. Priradenie by malo byť také, aby body v štruktúre opisovali otočený, zväčšený, pozdĺž osi natiahnutý, posunutý alebo stredovo súmerný obraz mnohouholníka pri pohľade na rovinu v priestore, kde leží mnohouholník. Týmto docielime, aby sa každý bod štruktúry premietol v rovnakom tvare a veľkosti na mnohouholník. Samozrejme, nič nám nebráni, aby sme priradili body ľubovoľne, avšak potom nemáme zaručené, že sa nám obraz nebude meniť v závislosti napríklad od natočenia mnohouholníka, čo poväčšinou nechceme. Na obrázkoch sú ukázané správne a nesprávne priradenia bodov (vľavo od šípky je mnohouholník, vpravo sú priradenia bodov v štruktúre).



Každému bodu $\boldsymbol{B_i} = (B_{ix}, B_{iy}, B_{iz})$ mnohouholníka priradíme teda bod $\boldsymbol{K_i} = (K_{ix}, K_{iy})$ v štruktúre.

Ako potom vykresľovať mnohouholník ? Toto vykresľovanie je založené na výpočte súradníc bodov štruktúry, ktoré ležia vo vnútri mnohouholníku. Tento výpočet je tiež nazývaný ***mapovanie***.

Rovinné mapovanie štruktúry

Toto mapovanie je len modifikáciou lineárnej interpolácii, ktorá bola použitá napríklad interpolácii farby pri tieňovaní. Podobne ako farba, lineárne sa interpolujú súradnice $\boldsymbol{x_s}$ sa $\boldsymbol{y_s}$ v štruktúre. Pre každú krajnú úsečku vypočítame rozdiel ***x***-ovej a ***y***-ovej súradnice v štruktúre pri zmene jedného riadku v obraze. Samotné vykresľovanie riadku zmeníme tak, že vypočítame zmenu týchto súradníc na jeden stĺpec obrazu. Ak počiatočný a koncový bod krajnej úsečky je $\boldsymbol{A} = (A_x, A_y)$ a $\boldsymbol{B} = (B_x, B_y)$, k nim priradený bod v štruktúre je $\boldsymbol{K} = (K_x, K_y)$ a $\boldsymbol{L} = (L_x, L_y)$, potom vektor zmeny súradníc v jednom riadku vypočítame ako

$$\boldsymbol{d} = \frac{\boldsymbol{K} - \boldsymbol{L}}{B_y - A_y}$$

Podobne by sme dostali zmenu súradníc pri prechode na ďalší stĺpec obrazu. Ak x_{min} a x_{max} boli minimálna a maximálna súradnica $\boldsymbol{x}$ v riadku, potom zmena v súradníc jednom stĺpci by bola

$$\boldsymbol{d} = \frac{\boldsymbol{K} - \boldsymbol{L}}{x_{max} - x_{min}}$$

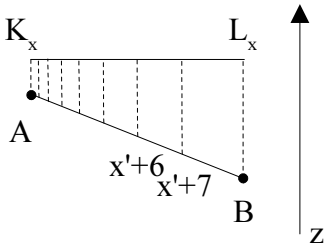
Výsledok použijeme tak, že pri každom kroku pripočítavame ku $\boldsymbol{x}$-ovej a $\boldsymbol{y}$-ovej súradnici v štruktúre rozdiel týchto súradníc v jednom riadku.

Problémom tohto mapovania je, že predpokladá rovnakú hĺbku všetkých bodov. Preto nie je vhodné na zobrazovanie pri perspektívnom premietaní.

Perspektívne mapovanie štruktúry

Pri perspektívnom zobrazovaní sa mení lineárna závislosť odchýliek súradníc v štruktúre od polohy vykresľovaného bodu v obraze. Preto hlavným prostriedkom ku správnemu vykresleniu je zistenie vzťahov medzi susednými bodmi v obraze a ich priradeným hodnotám v štruktúre.

Označíme si dva koncové body úsečky mnohouholníka v priestore ako $\boldsymbol{A} = (x, y, z)$ a $\boldsymbol{B} = (x_b, y_b, z_b)$, k nim prislúchajúce body v štruktúre $\boldsymbol{K} = (k_x, k_y)$ a $\boldsymbol{L} = (l_x, l_y)$, rozdiely $\Delta = \boldsymbol{B} - \boldsymbol{A}$ a $\Delta k = \boldsymbol{L} - \boldsymbol{K}$. Pre súradnice v priestore dvoch vedľajších bodov $\boldsymbol{P}$, $\boldsymbol{R}$ (ležiacich na úsečke $\boldsymbol{AB}$) na obrazovke (o jeden stĺpec obrazu ďalej) platí $\boldsymbol{P} = \boldsymbol{A} + g_1 \Delta$ a $\boldsymbol{R} = \boldsymbol{A} + g_2 \Delta$, kde $g_1 = \frac{x - x' z}{x' \Delta z - \Delta x}$ a $g_2 = \frac{x - (x'+1) z}{(x'+1) \Delta z - \Delta x}$ a $\boldsymbol{x'}$ je $\boldsymbol{x}$-ova súradnica bodu $\boldsymbol{P}$ v obraze (podrobnejšie je to ukázané pri popise pamäti hĺbok). Vieme už vyjadriť polohu dvoch susedných bodov na obrazovke v priestore. Ako sa však budú v týchto dvoch bodoch meniť súradnice v štruktúre ? Bude to zrejme rovnako ako súradnice týchto dvoch bodov $(\boldsymbol{K} + g(x')\Delta)$. To však je veľmi nepríjemná závislosť, keďže g závisí od x'.



Skúsme sa teraz pozrieť len na $\boldsymbol{x}$-ove súradnice v štruktúre. V bode $\boldsymbol{P}$ je súradnica $k_x' = k_x + g_1 \Delta k$, v bode $\boldsymbol{R}$ zase $k_x'' = k_x + g_2 \Delta k$. Pozrime sa však na pomery $\frac{k_x'}{z}'$ a $\frac{k_x''}{z''}$. Môžeme vyjadriť, že $e_1 = \frac{k_x'}{z}' = \frac{x'(k_x \Delta - z \Delta k_x) + x \Delta k_x - k_x \Delta x}{x \Delta z - z \Delta x}$, a tiež aj $e_2 = \frac{k_x''}{z''} = \frac{x'(k_x \Delta - z \Delta k_x) + k_x(\Delta z - \Delta x) + \Delta k_x (x - z)}{x \Delta z - z \Delta x}$. Ak sa pozrieme na e_2-e_1, zistíme, že $e_2 - e_1 = \frac{\Delta k_x (x - z)}{x \Delta z - z \Delta x}$ je konštantný výraz. Preto ho môžeme ho použiť na zistenie súradníc $\boldsymbol{x}$ a $\boldsymbol{y}$ v štruktúre tak, že ho budeme lineárne interpolovať popri úsečke. Vykresľovanie mnohouholníka sa zmení v tom, že budeme počítať miesto zmien súradníc $\boldsymbol{x}$ a $\boldsymbol{y}$ v štruktúre pri prechode o jeden riadok alebo stĺpec zmenu pomerov $\frac{\boldsymbol{k}_x}{\boldsymbol{z}}$ a $\frac{\boldsymbol{k}_y}{\boldsymbol{z}}$. Pri vykresľovaní riadku začneme od začiatočného pomeru a postupným pripočítavaním zmeny sa dostaneme až po koncový pomer v riadku. Ak budeme potrebovať pri vykresľovaní bodu zistiť presnú súradnicu v štruktúre, stačí tento pomer prenásobiť súradnicou $\boldsymbol{z}$ súčasného bodu.

Pamäť hĺbok

Pamäť hĺbok je veľmi prirodzený prostriedok na dosiahnutie správnej viditeľnosti objektov priestoru. Spočíva v tom, že pre každý bod vykresľovaného mnohouholníka vypočítavame okrem súradníc na obrazovke aj informácie o vzdialenosti (hĺbke) tohto bodu. Na odkladanie tejto informácie slúži pole hĺbok, ktoré má rovnaké rozmery ako obrazovka. Zisťovanie viditeľnosti je založené na tom, že vypočítaná hĺbka daného bodu sa porovná s hodnotou v poli hĺbok na pozícii daného bodu, a pokiaľ je hĺbka tohto bodu menšia ako hodnota poľa, bod sa vykreslí a poľu sa priradí hĺbka tohto bodu. Pokiaľ hodnota bodu je väčšia, bod sa jednoducho ignoruje a pokračuje sa ďalším bodom. Týmto spôsobom je zaručené, že objekty priestoru budú vykreslené správne.

Algoritmus vykresľovania

1. Vyplnenie poľa hĺbok najväčšou možnou hodnotou (hĺbkou).
2. V každom mnohouholníku, ktorý sa má vykresľovať, vypočítavame vnútorné body aj s hĺbkami. Ak hĺbka bodu je menšia ako hodnota v poli hĺbok, bod sa vykreslí a jeho hĺbka nahradí doterajšiu hĺbku v poli.

Výpočet hĺbky

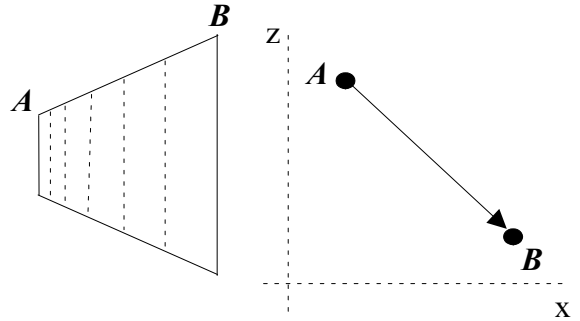
Hĺbku bodu (súradnicu ***z***) máme danú pre všetky krajné body mnohouholníka. Pri vykresľovaní mnohouholníka však potrebujeme zistiť, akú hĺbku majú aj všetky ostatné body tohto útvaru. Ak máme dva body mnohouholníka - $\boldsymbol{A}=(x, y, z)$ a $\boldsymbol{B}=(x_b, y_b, z_b)$, úsečka medzi nimi má rozmery $\boldsymbol{ab}=(\Delta x, \Delta y, \Delta z)=(x_b-x, y_b-y, z_b-z)$. Potom môžme vyjadriť akýkoľvek bod na tejto úsečke ako $\boldsymbol{C}=u\,\boldsymbol{ab}, u\in\langle 0;1\rangle$. Ak vykresľujeme mnohouholník pomocou kreslenia vodorovných čiar, tak nás zaujíma hodnota rozdielu súradnice ***z*** medzi dvomi následujúcimi bodmi v riadku - ***x'*** a ***x'+1***. Potom vykresľovanie vodorovnej čiary mnohouholníka môže byť vyjadrené takto (viaže sa na klasické vykresľovanie mnohouholníka rozkladom na riadky):

1. Vypočítanie dĺžky riadku $dr = x_k - x_z$ a rozdielu medzi počiatočnou hĺbkou v riadku a konečnou, $zr = z_k - z_z$. Zmena hĺbky v jednom bode riadku bude potom $dz=\frac{zr}{dr}$. Nastavíme $x = x_z$ a $z = z_z$.
2. Pokiaľ aktuálna súradnica x je menšia alebo rovná x_k , pozrieme sa do poľa hĺbok na terajšie súradnice x,y a ak pole$[x,y]$ je väčšie ako súčasná hĺbka z, potom pole$[x,y] = z$ a vykreslíme bod.
3. $x=x+1, z=z+dz$ a vrátime sa na 2.

Zdalo by sa, že všetko je v poriadku. Avšak nemusí to byť celkom tak. Tento spôsob práce s hĺbkou by bol funkčný v závislosti od zvoleného premietania. Pri rovnobežnom premietaní by všetko fungovalo správne, no napríklad pri centrálnej projekcii (perspektíve) by to robilo s obrazom akési divné veci. V čom je problém ? Hĺbka pri perspektívnom premietaní nerastie lineárne. Potom nemá zmysel počítať hĺbku z predošlého bodu pripočítavaním konštanty dz. Čo však potom robiť ? Ak nerastie lineárne, je treba počítať hĺbku pomocou nejakého nelineárneho a výpočtovo dlhotrvajúceho vzťahu ? Našťastie, centrálna projekcia má inú užitočnú vlastnosť, ktorá nielenže nespomalí výpočet hĺbky, ale umožní aj to, že nebude potrebné vymazávať pole hĺbok pre každý obraz, ale len po ich určitom počte.

Výpočet hĺbky pri perspektívnom zobrazení

Ak si opäť označíme koncové body úsečky ako ***A*** a ***B*** a vektor medzi nimi označíme ***ab***, potom ľubovoľný bod na úsečke môžeme vyjadriť ako ***A*** + ***u ab***. Vzťahy pre výpočet ***x′***-ovej a ***y′***-ovej súradnice v obraze sa pri centrálnej projekcii takmer zhodujú, preto ďalej budeme vyjadrovať iba súradnicu ***x′***. Ak si zo vzťahu pre ***x′***-ovu súradnicu v centrálnej projekcii vyberieme



iba časť, ktorá nie je konštantná - $x' = \frac{x}{z}$, potom ***x′***-ovu súradnicu ľubovoľného bodu na úsečke ***A*** + *u* ***ab*** môžeme vyjadriť ako $x' = \frac{x + u\Delta x}{z + u\Delta z}, u \in \langle 0\,;1 \rangle$. Zaujíma nás, aký vzťah je medzi hĺbkami dvoch susedných bodov v riadku obrazu (***x′*** a ***x′+1***). Ak si z predchádzajúceho vzťahu vyjadríme

$u_1 = \frac{x - x'z}{x'\Delta z - \Delta x}$ a $u_2 = \frac{x - (x'+1)z}{(x'+1)\Delta z - \Delta x}$, môžeme vyjadriť hĺbky týchto bodov:

$$z_1 = z + u_1\Delta z = z + \frac{x - x'z}{x'\Delta z - \Delta x}\Delta z = \frac{x\Delta z - z\Delta x}{x'\Delta z - \Delta x}$$

$$z_2 = z + u_2\Delta z = z + \frac{x - (x'+1)z}{(x'+1)\Delta z - \Delta x}\Delta z = \frac{x\Delta z - z\Delta x}{(x'+1)\Delta z - \Delta x}$$

Je jasné, že z_2 - z_1 určite nie je konštanta (pretože v menovateli je premenná ***x′***). No ak sa lepšie pozrieme, zistíme, že $\frac{1}{z_2} - \frac{1}{z_1} = \frac{\Delta z}{x\Delta z - z\Delta x}$ už konštantou je. Z toho nám vyplýva, že $\frac{1}{z}$ sa mení v priestore obrazu lineárne a môžeme ju použiť ako kritérium viditeľnosti pri pamäti hĺbok. Samozrejme, algoritmus musíme mierne upraviť, pretože $\frac{1}{z}$ rastie vtedy, keď *z* klesá a naopak.

Nový algoritmus vykresľovania čiary mnohouholníka by mohol vyzerať takto:

1. Vypočítanie dĺžky riadku $dr = x_k - x_z$ a rozdielu medzi počiatočnou "hĺbkou" v riadku a koncovou, $z_r = \frac{1}{z_k} - \frac{1}{z_z}$. Zmena "hĺbky" v jednom bode riadku bude potom $dz = \frac{zr}{dr}$. Nastavíme $x = x_z$ a $z = \frac{1}{z_z}$.
2. Pokiaľ je aktuálna súradnica *x* menšia alebo rovná x_k , pozrieme sa do poľa hĺbok na terajšie súradnice *x,y* a ak pole[*x,y*] je menšie ako súčasná "hĺbka" *z*, potom pole[*x,y*] = *z* a vykreslíme bod.
3. *x=x+1, z=z+dz* a vrátime sa na 2.

("hĺbka" je uvádzaná v úvodzovkách preto, lebo sa nejedná o skutočnú hĺbku, iba o výpočtový ekvivalent hĺbky pôvodného algoritmu)

Zefektívnenie činnosti pamäti hĺbok

Po každom vytvorení obrazu sa musí pamäť hĺbok vymazať, aby bola pripravená na vykresľovanie ďaľšieho obrazu. Ak však použijeme pre reprezentáciu hĺbky číslo s pevnou rádovou čiarkou a vhodne zvolíme vzdialenosť z_{min} prednej orezávacej roviny, stačí pamäť hĺbok vymazávať vždy až po určitom počte obrazov.Pri perspektívnej transformácii ako kritérium "hĺbky" volíme $\frac{1}{z}$. Ak zvolíme z_{min} v tvare 2^{-i}, potom máme zaručené, že žiaden bod vo výslednom obraze nebude mať hodnotu väčšiu ako 2^i. Potom zostávajúce horné bity od í-teho indexu budú pre všetky body nadobúdať hodnotu 0. Pozrime sa však na vykresľovanie mnohouholníka. Kedy budeme vykresľovať bod na obrazovku ? Vypočítame $\frac{1}{z}$ daného bodu a porovnáme s hodnotou na príslušnom mieste v pamäti hĺbok. Ak je hodnota tohto podielu pre zvolený bod väčšia ako hodnota v poli hĺbok, potom ho vykreslíme. Keďže pri zvolenom z_{min} máme určitý počet najvyšších bitov rovný vždy 0, môžeme pomocou nich dosiahnuť to, aby "hĺbky" následujúceho obrazu boli vždy väčšie ako "hĺbky" predošlého obrazu. Body následujúceho obrazu potom vždy dostanú prednosť pred bodmi predošlého obrazu. Horné bity totiž určujú vyššie rády čísiel ako nižšie bity a nastavením ľubovoľného horného bitu v jednom z dvoch rovnakých čísel dosiahneme to, že bude väčšie ako to, kde sme daný horný bit nechali rovný 0. Potom pri každom obraze sa stačí na skupinu horných nulových bitov pozerať ako na samostatné číslo a pre každý ďaľší obraz ho zvýšiť o jedna. Po určitom počte obrazov sa však toto číslo pretečením opäť dostane do 0. Vtedy musíme pamäť hĺbok vymazať. Celkovo však týmto spôsobom môžeme vykresliť 2^i obrazov bez nutnosti vymazávania pamäti hĺbok.

Pamäť úsekov

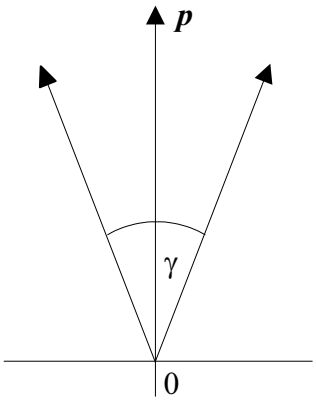
Modifikáciou pamäti hĺbok je pamäť úsekov. Využíva spôsob vykresľovania mnohouholníkov pomocou riadkového rozkladu. Miesto toho, aby sme okamžite vykresľovali body mnohouholníka, pokúsime sa vložiť riadkový úsek mnohouholníka do zoznamu úsekov, ktoré sa nachádzajú v danom riadku obrazovky. Vkladať sa budú tak, že sa budú vzájomne porovnávať polohy krajných bodov zaraďovaného a zaradených úsekov. Pri vkladaní úsekov môže nastať viacero možností. Zaraďovaný úsek môže úplne alebo čiatočne prekryť jeden alebo viacero zaradených úsekov, môže byť nimi naopak čiastočne alebo úplne prekrytý alebo môže byť umiestnený mimo doposiaľ zaradených úsekov. Zaraďujeme ho však tak, aby ***x***-ové intervaly súradnic úsekov tvorili disjunktnú množinu - každý úsek by mal zaberať unikátnu časť riadku. Preto je potrebné úseky navzájom orezávať, zisťovať vzájomné priesečníky a niekedy rozkúskovať pôvodné úseky na viacero nových. Pri zaraďovaní úsekov samozrejme musia byť v zozname len úseky, ktoré sú najbližšie. Pokiaľ zaraďujeme úsek, ktorý je ďalej ako už zaradené úseky, potom po porovnaní krajných bodov zaraďovaného úseku a úsekov, ktoré ležia na mieste, kam by mal byť zaradený, ho môžeme okamžite vylúčiť z ďaľšieho spracovávania. Preto je vhodné pred vykresľovaním mnohouholníkov si tieto pretriediť vzostupne podľa minimálnej hodnoty ***z***-ovej súradnice v mnohouholníku. Takto obmedzíme viacnásobné prekresľovania mnohouholníkov, čo je jednou zo slabých stránok pamäti hĺbok, keďže v nej sa za každých okolností porovnáva vykresľovaný bod s hodnotou nachádzajúcou sa v nej. Avšak pamäť úsekov môže mať problémy s kapacitou pamäti, hlavne pri obrazoch, kde sa nachádza veľmi veľa malých mnohouholníkov. Použitie pamäti úsekov však mnohokrát veľmi silne urýchľuje vytváranie obrazu, dokonca predstihuje rýchlosť špecializovaných procesorov využívajúcich pamäť hĺbok.

Odstránenie neviditeľných mnohouholníkov

Mnoho objektov je zložených z mnohouholníkov, ktoré nemusia byť pri nejakom pohľade na objekt viditeľné. Nastáva niekoľko prípadov. Mnohouholník môže byť orientovaný opačne, smerom od pohľadu (jeho predná strana je voči smeru pohľadu odvrátená, a preto nie je mnohouholník viditeľný), celý objekt môže ležať mimo viditeľnej oblasti alebo do nej len čiastočne zasahovať. Preto sa hľadajú spôsoby, ako neviditeľné mnohouholníky vylúčiť zo spracovávania a tým urýchliť zobrazovanie.

Jeden zo základných spôsobov je založený na zistení uhla normálového vektora roviny mnohouholníka a vektora pohľadu. Pri paralelnej projekcii platí, že ak je tento uhol v intervale $\left(-\frac{\pi}{2}, \frac{\pi}{2}\right)$, pohľad smeruje na neviditeľnú plochu mnohouholníka, a ten potom nie je potrebné vykresľovať. Keïže pohľad v relatívnych súradniciach má smer vektora $\boldsymbol{p} = (0, 0, 1)$ (čo znamená, že pohľad smeruje na zväčšujúcu sa os ***z***), môžeme využiť to, že ak ***z***-ová súradnica normály je väčšia ako 0, mnohouholník nie je viditeľný. Ak si označíme normálu mnohouholníka $\boldsymbol{n} = (n_x, n_y, n_z)$ a predpokladáme, že má jednotkovú dĺžku, potom môžeme vyjadriť $\cos\delta = \boldsymbol{n} \cdot \boldsymbol{p} = (n_x, n_y, n_z) \cdot (0, 0, 1) = n_z$, kde δ je uhol medzi vektormi ***p*** a ***n***. Keďže kosínus je kladný v intervale $\left(-\frac{\pi}{2}, \frac{\pi}{2}\right)$, vyplýva z toho, že skutočne stačí zistiť, či je $n_z > 0$, aby sme mohli o mnohouholníku prehlásiť, že je neviditeľný.

Problém je však v tom, že pri perspektívnej transformácii sa týmto spôsobom nedá zistiť, či je mnohouholník skutočne neviditeľný, keďže aj mnohouholníky s normálami smerujúcimi od pohľadu môžu byť za určitých podmienok viditeľné. Môžeme však s istotou odstrániť tie mnohouholníky, ktorých normály zvierajú s pohľadom uhol menší ako $\frac{\pi - \gamma}{2}$, kde γ je väčší z dvoch priemetov zorných uhlov α alebo β. Vyplýva to z toho, že pri perspektívnej transformácii je možné vidieť plochy, ktorých normála zviera s vektorom pohľadu uhol, ktorý je maximálne polovicou zorného uhla. Pre určenie kritéria porovnávania môžeme využiť podobné vzťahy ako v predošlom prípade. Na to, aby bol mnohouholník neviditeľný však musí platiť $\cos\delta > \cos\frac{\gamma}{2}$. Po rozpísaní dostávame vzťah



$$\cos\delta = \boldsymbol{n} \cdot \boldsymbol{p} = (n_x, n_y, n_z) \cdot (0, 0, 1) = n_z > \cos\frac{\gamma}{2},$$

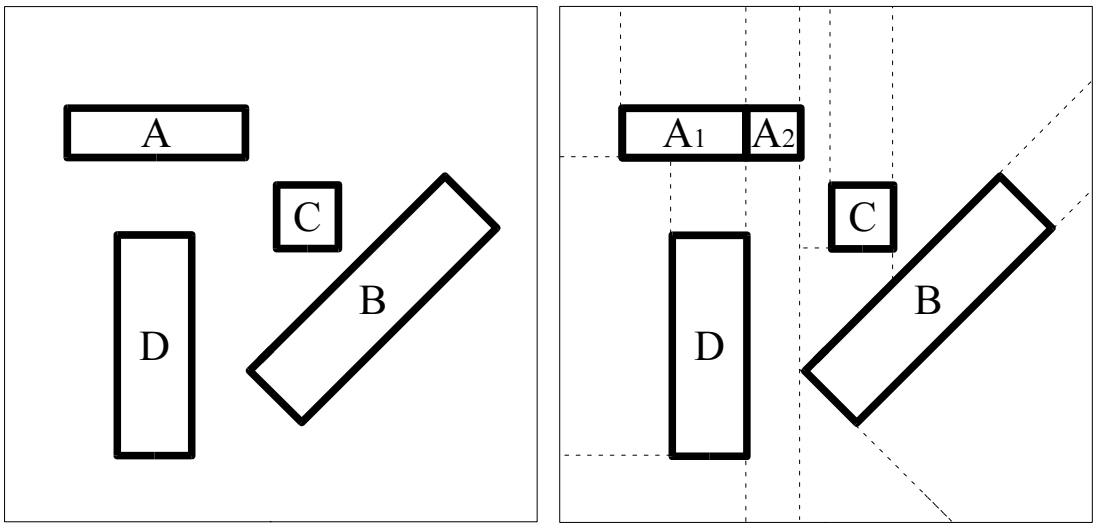
ktorý už môžeme použiť na zistenie viditeľnosti.

Strom binárneho rozdelenia priestoru

Pri zisťovaní viditeľnosti, alebo pri vytváraní tieňov, ktoré sú vrhané objektami na iné objekty je vhodné rozdeliť priestor (objekty a ich zložky) tak, aby práca s ním bola čo najefektívnejšia. Jednou z možností je použiť strom binárneho rozdelenia priestoru (**B**inary **S**pace **P**artitioning **Tree**).

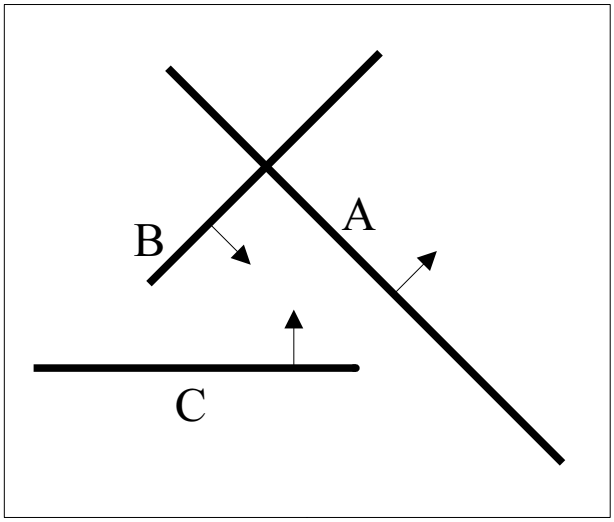
Princíp stromu je postavený na skutočnosti, že ak si zvolíme ľubovoľnú rovinu v zobrazovanom priestore a všetky mnohouholníky objektov v tomto priestore rozdelíme na dve časti podľa toho, v ktorom polpriestore sa nachádzajú (či ležia v *prednom* alebo *zadnom* polpriestore roviny), tak ak najskôr vykreslíme všetky mnohouholníky, ktoré sa nachádzajú v *zadnom* polpriestore, a potom tie, ktoré sú v *prednom* polpriestore, dosiahneme to, že tieto mnohouholníky budú vykreslené v správnom poradí a žiadny mnohouholník nebude prekrytý iným mnohouholníkom tam, kde by mal byť vykreslený sám. Toto však platí len vtedy, pokiaľ sa nám podarí zostrojiť také roviny, ktoré v danom polpriestore nepretínajú žiadny iný mnohouholník. Pokiaľ by však zvolená rovina pretínala ľubovoľný mnohouholník, tento je potrebné rozdeliť na dve časti - na *"prednú"* časť a *"zadnú"* časť. Delenie rovinami na polpriestory je vhodné vykonávať dovtedy, pokiaľ každému mnohouholníku nie je pridelený vlastný polpriestor. Pre potreby zisťovania viditeľnosti sa ukazuje byť vhodným viesť deliacu rovinu tak, aby bola totožná s rovinou zvoleného mnohouholníka. Týmto spôsobom sa môže uchovávať celý objekt, ktorý je zložený z mnohouholníkov a to tak, že vrcholmi stromu budú jednotlivé mnohouholníky (a tiež tie, ktoré sa vytvoria z pôvodných mnohouholníkov pri delení rovinami). V každom vrchole bude teda uvedený mnohouholník, ktorý delí polpriestor, v ktorom sa nachádza, na ďalšie dva polpriestory určené hranicami pôvodného polpriestoru a deliacou rovinou mnohouholníka. Každý vrchol bude ukazovať na ďalšie dva vrcholy, jeden *privrátený* a druhý *odvrátený*, podľa toho, do ktorého nového polpriestoru patria (rozdelenie z hľadiska smerovania normálového vektoru roviny mnohouholníka - *privrátený* - patriaci do polpriestoru od roviny v smere normálového vektoru, *odvrátený* - patriaci do opačného polpriestoru)

Zoberme si napríklad scénu na prvom obrázku. V nej sa nachádzajú 4 objekty - **A**,**B**,**C**,**D**. Po vytvorení stromu dostávame druhý obrázok, kde sú znázornené roviny, ktoré delia polpriestory.



Môžeme si všimnúť, že na druhom obrázku je už objekt A rozdelený na dve časti - **A**1 a **A**2.

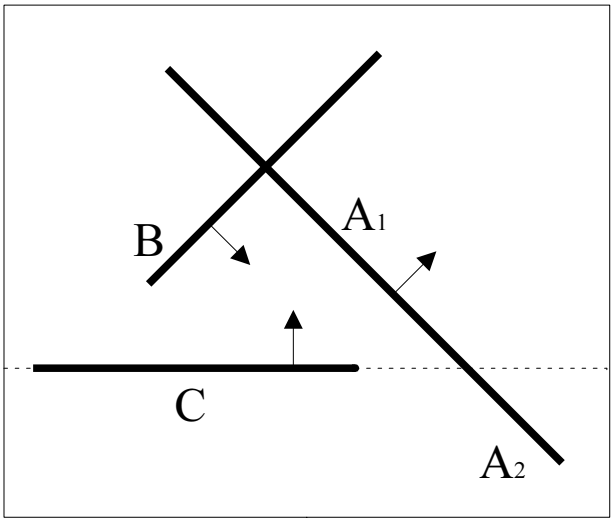
Zoberme si detailnejší príklad, z ktorého bude viac zrejmý postup vytvárania stromu.



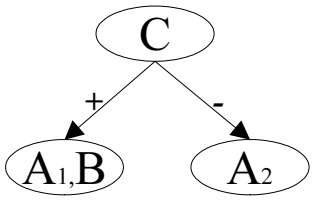
Na obrázku máme zobrazený priestor s tromi mnohouholníkmi - **A**, **B**, **C**. Chceme vytvoriť strom binárneho rozdelenia priestoru, ktorý by opisoval rozloženie týchto mnohouholníkov v priestore (ide o pohľad zhora).

Šípky označujú orientáciu mnohouholníkov v priestore - ich privrátenú stranu (alebo tiež orientáciu normály roviny).

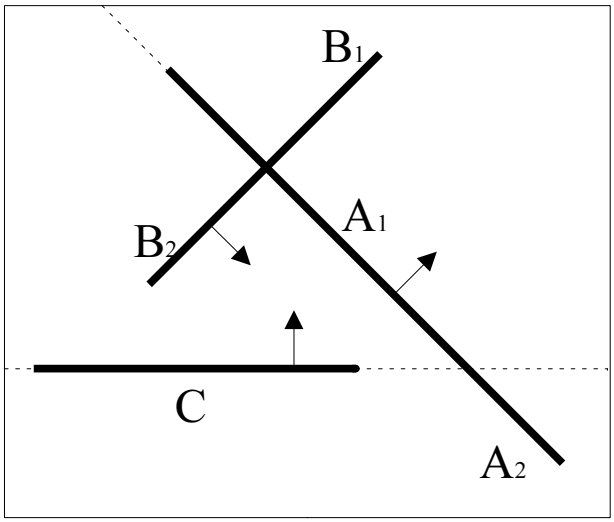
Začneme napríklad mnohouholníkom **C**. Ak ním vedieme deliacu rovinu, zistíme, že táto rovina pretína mnohouholník **A** a delí ho na dve časti - $\mathbf{A}_1$ a $\mathbf{A}_2$.



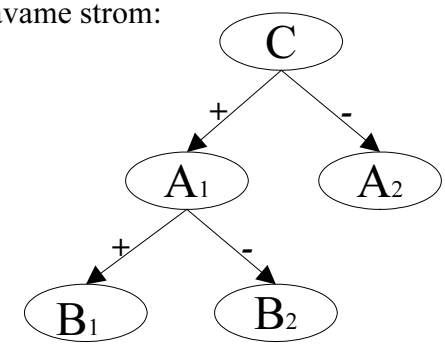
Dostávame prvý strom:



(+ znamená odkaz na privrátenú stranu,
- na odvrátenú stranu deliacej roviny prechádzajúcej mnohouholníkom, z ktorého vychádza šípka)



Teraz môžeme rozdeliť predný polpriestor **C** na dva polpriestory rovinou mnohouholníka $\mathbf{A}_1$.
Dostávame strom:



ktorý už úplne opisuje štruktúru priestoru.

Vytvorenie stromu

V tejto časti bude uvedený náznak algoritmu vytvárania stromu BRP v určitých bodoch, ktoré budú detailnejšie popisované v ďalšom texte.

Algoritmus

1. **Výber mnohouholníka**, ktorý doposiaľ nebol použitý na rozdelenie priestoru. Tým sa aktuálny priestor zmení na polpriestor, v ktorom sa nachádza vybraný mnohouholník (pokiaľ je to prvý vybraný mnohouholník, je to celý priestor).
2. **Rozdelenie mnohouholníkov** v aktuálnom polpriestore na dve skupiny - na tie, ktoré ležia pred rovinou vybraného mnohouholníka a tie, ktoré ležia za ňou.

2a. Pokiaľ **rovina pretína nejaký mnohouholník**, tento sa rozdelí na dve časti, jedna časť sa priradí do privráteného, druhá do odvráteného polpriestoru.

3. Ak **na rozdelenie priestoru boli už použité všetky mnohouholníky**, je strom binárneho rozdelenia priestoru vytvorený. Inak algoritmus opakujeme od začiatku.

Výber deliaceho mnohouholníka:

Pri vyberaní deliaceho mnohouholníka môžeme vyberať ľubovoľný mnohouholník. Avšak je tu veľké riziko, že rovina vybraného mnohouholníka pretína množstvo iných mnohouholníkov, čo by spôsobilo veľký nárast počtu novovytvorených mnohouholníkov. Preto je vhodné zaoberať sa vylepšovaním stromu.

Ideálny strom

Pri tvorbe optimálneho stromu dochádza ku konfliktu dvoch cieľov - vyváženosti stromu a počtu polygónov. Vytvorenie optimálneho stromu je **NPC**, preto sa častokrát od jeho vyhľadania ustupuje a hľadajú sa stromy, ktoré zodpovedajú určitým nárokom, ktoré si vyžaduje ich konkrétna aplikácia. Pre optimálne vyváženie stromu nie je vhodné použiť klasické vyvažovanie stromu, pretože poloha mnohouholníka v strome určuje aj jeho polohu v polpriestore v závislosti od deliacich rovín, a pri presunutí mnohouholníka z jednej časti stromu do druhej by sa táto závislosť porušila a bolo by potrebné rekonštruovať veľkú časť stromu, čo by si napríklad vyžiadalo opätovné spájanie mnohouholníkov a ďalšie rozdeľovanie iných mnohouholníkov, čo by nedokázalo zaručiť úspešný výsledok.

Iné prístupy

*S*ú tu uvedené dva typy prístupov - hľadanie *minimálneho počtu polygónov* a *vyváženie stromu*. Každý z nich má svoje uplatnenie v iných oblastiach počítačovej grafiky.

Hľadanie minimálneho počtu polygónov - zvyčajne sa hľadá pre každý polpriestor taký mnohouholník, ktorého rovina pretína čo najmenší počet mnohouholníkov v danom polpriestore. Samozrejme, že toto nezaručí minimálny počet mnohouholníkov v priestore, ale má šancu byť efektívnejším spôsobom organizovania stromu ako náhodný výber. Využíva sa najmä pri zobrazovaní, pretože zobrazovanie si vyžaduje prechádzať práve raz každým mnohouholníkom, a keď zmenšíme ich počet, tak dosiahneme menší čas zobrazovania.

Algortimus

1. Vyber prvý mnohouholník z polpriestoru
2. Prejdi všetky ostatné mnohouholníky v polpriestore a zisti, na koľko mnohouholníkov sa rozdelia, keď sa bude zisťovať ich poloha voči deliacej rovine.

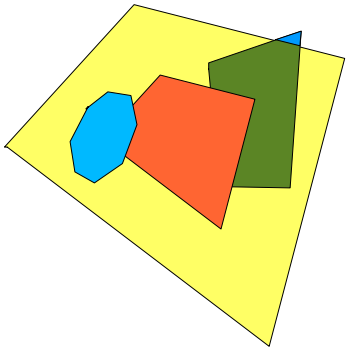
3. Opakujeme dovtedy, kým existujú nevybraté mnohouholníky v polpriestore.
4. Podľa mnohouholníka, ktorý mal najmenší počet novovzniknutých mnohouholníkov delíme polpriestor.

Spôsob výberu podľa najmenšieho počtu rozdelených mnohouholníkov je časovo náročnejší, pretože jeho zložitosť je **O(r²)**, kde **r** je počet mnohouholníkov v danom polpriestore, zatiaľ čo náhodný výber (alebo výber podľa ľubovoľnej postupnosti) má zložitosť **O(1)**. Avšak celú činnosť stačí pre statické objekty vykonávať len raz na začiatku, keď sa vygeneruje strom, ktorý sa už potom nebude meniť.

Vyvažovanie stromu

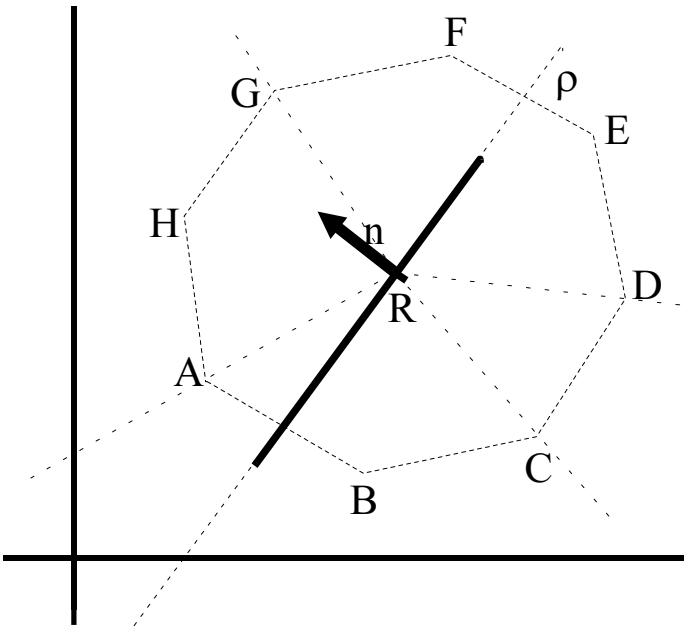
V niektorých prípadoch je lepšie vytvárať vyvážený strom. Sú to najmä 3D editory a programy, ktoré analyzujú povrch. Tieto potrebujú veľmi rýchle vyhľadávanie mnohouholníka v priestore. Toto vyhľadávanie sa môže skrátiť tým, že sa budeme snažiť tvoriť vyvážený strom BRP. Jednou z možností, ako sa k tomu priblížiť je modifikovať predošlý algoritmus tak, že budeme vyberať vždy ten mnohouholník, ktorý nám polpriestor rozdelí na dva polpriestory s rovnakým počtom mnohouholníkov (alebo s najmenšou odchýlkou počtu). Môžeme tiež zohľadniť, aby bol vybraný taký mnohouholník, ktorý delí na najmenší počet mnohouholníkov, zo všetkých, ktoré daný polpriestor rovnomerne rozdeľujú. Pri použití v 3D editore je však nutné dopredu poznať povahu objektu, ktorý chceme vyjadriť pomocou stromu BRP, pretože pri modifikujúcich operáciách sa vyváženosť môže stratiť. Zložitosť algoritmu je opäť **O(r²)**.

Rozdelenie mnohohuholníkov:



Po zvolení deliaceho mnohouholníka je potrebné rozdeliť mnohouholníky na dve skupiny: na tie, ktoré patria do privráteného a tie, ktoré patria do odvráteného polpriestoru. Ako však zistiť, ktorý mnohouholník kam patrí ?

Predstavme si mnohohuholník ***ABCDEFGH***, ktorý pretína rovina ρ deliaceho mnohouholníka. Teraz je zrejmé, že tento mnohouholník nemôžeme jednoznačne priradiť ani do jedného polpriestoru, ale ho musíme rozdeliť. To však zatiaľ nie je dôležité, skúsme sa najskôr pozrieť, ako zistíme polohu jednotilivých bodov, vzhľadom na deliacu rovinu. Vyberme si ľubovoľný bod na rovine ρ, napríklad bod $\boldsymbol{R} = (R_x, R_y, R_z)$. O deliacej rovine vieme, že má nejaký normálový vektor $\boldsymbol{n}$, ktorý je na ňu kolmý. Skúsme sa teda pozrieť, aký uhol zviera vektor od ľubovoľného vrcholu po bod $\boldsymbol{R}$ s normálovým vektorom $\boldsymbol{n} = (n_x, n_y, n_z)$. Ak zoberieme napríklad vrchol $\boldsymbol{A} = (A_x, A_y, A_z)$, potom dostávame vektor $\boldsymbol{AR} = [A_x - R_x, A_y - R_y, A_z - R_z]$ a môžeme zistiť uhol α, ktorý zviera s normálou $\boldsymbol{n}$ podľa vzťahu $\cos(\alpha) = \frac{\boldsymbol{AR} \cdot \boldsymbol{n}}{|\boldsymbol{AR}|\,|\boldsymbol{n}|}$. Zistíme, že ak sa bod $\boldsymbol{A}$ nachádza pred rovinou, tento uhol je z intervalu $\left(\frac{-\pi}{2}, \frac{\pi}{2}\right)$. Keď sa bližšie pozrieme na skalárny súčin $\boldsymbol{AR} \cdot \boldsymbol{n}$, zistíme, že je kladný práve v tomto intervale, inak je záporný (alebo 0). Z toho vyplýva, že môžeme ako indikátor, či sa bod nachádza na jednej alebo druhej strane deliacej roviny



použiť jednoducho skalárny súčin vektora od tohto bodu po ľubovoľný bod na deliacej rovine s normálou deliacej roviny.

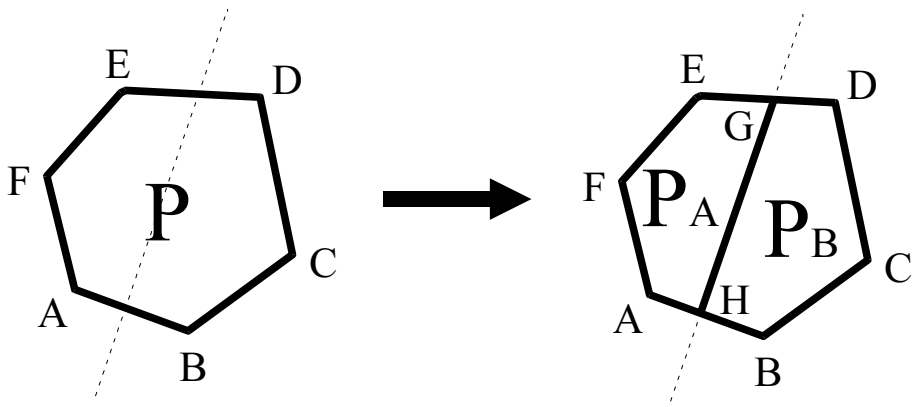
Na tomto obrázku je znázornený osemuholník ***ABCDEFGH***. Môžeme si všimnúť, že vektory z bodov ***A***, ***F***, ***G***, ***H*** do bodu ***R*** zvierajú s normálou roviny ***n*** uhol z intervalu $\left(\frac{-\pi}{2}, \frac{\pi}{2}\right)$.
Body ***B***, ***C***, ***D***, ***E*** zase uhol z intervalu $\langle 0, 2\pi \rangle \setminus \langle \frac{-\pi}{2}, \frac{\pi}{2} \rangle$.

Teraz je už jasné, akým spôsobom sa dajú rozdeliť mnohouholníky do dvoch skupín podľa toho, na ktorej strane deliacej roviny sa nachádzajú. Pre každý bod mnohouholníka stačí zistiť hodnotu skalárneho súèinu normály deliacej roviny a vektora vedeného z bodu mnohouholníka do ľubovoľného bodu na rovine. Ak všetky body mnohouholníka majú kladný výsledok, mnohouholník patrí do privráteného polpriestoru, ak výsledok je u všetkých záporný, potom mnohouholník patrí do odvráteného. Ak však mnohouholník má aj kladné aj záporné body, je potrebné ho rozdeliť na dva nové mnohouholníky. Ak je výsledok pre nejaký bod rovný 0, je možné priradiť ho kamkoľvek, pretože leží v deliacej rovine.

Tomuto spôsobu by sa dalo vytýkať, že hovoríme o uhle vektora vedeného z bodu mnohouholníka do ľubovoľného bodu v rovine a normály roviny, zatiaľ čo na obrázku je uhol s nejakým bodom na priamke, ktorá prechádza rovinou mnohouholníka. Avšak výsledok platí aj pre priestorový uhol.

Rozdelenie mnohouholníka

Ak deliaca rovina pretína mnohouholník, musíme ho rozdeliť na dve časti. V každom takomto mnohouholníku existujú dve dvojice následujúcich vrcholov, z ktorých jeden sa nachádza v privrátenej časti a druhý v odvrátenej. Na ďaľšom obrázku sú takými body ***E-D*** a ***B-A***. Medzi nimi sa vždy nachádza bod, ktorý leží na deliacej rovine. Tieto body sú teda dva. Stačí ich nájsť a vytvoriť dva nové mnohouholníky - jeden bude obsahovať všetky privrátené body a tieto dva nové, a druhý všetky odvrátené spolu s dvomi novými bodmi. Tieto dva nové mnohouholníky budú mať opäť rovnaký normálový vektor ako pôvodný mnohouholník.



Ako zistiť bod, v ktorom pretne priamka rovinu ?

Rovinu máme určenú dvomi vektormi $\boldsymbol{u}=(u_x, u_y, u_z)$, $\boldsymbol{v}=(v_x, v_y, v_z)$ a bodom $\boldsymbol{a}=(a_x, a_y, a_z)$, ktorý leží v tejto rovine. Potom môžeme rovinu vyjadriť ako $\boldsymbol{a}+k\,\boldsymbol{u}+l\,\boldsymbol{v}$, $k, l \in \boldsymbol{R}$. Priamku máme opísanú bodom $\boldsymbol{d}=(d_x, d_y, d_z)$, ktorý na nej leží, a smerovým vektorom $\boldsymbol{m}=(m_x, m_y, m_z)$. Táto priamka sa dá vyjadriť ako $\boldsymbol{d}+t\,\boldsymbol{m}$, $t \in \boldsymbol{R}$. Potom pre bod, ktorý majú rovina a priamka spoločné, platí $\boldsymbol{a}+k\,\boldsymbol{u}+l\,\boldsymbol{v}=\boldsymbol{d}+t\,\boldsymbol{m}$, vo vektorovom vyjadrení:

$$\begin{pmatrix} a_x \\ a_y \\ a_z \end{pmatrix} + \boldsymbol{k}\begin{pmatrix} u_x \\ u_y \\ u_z \end{pmatrix} + \boldsymbol{l}\begin{pmatrix} v_x \\ v_y \\ v_z \end{pmatrix} = \begin{pmatrix} d_x \\ d_y \\ d_z \end{pmatrix} + \boldsymbol{t}\begin{pmatrix} m_x \\ m_y \\ m_z \end{pmatrix}$$

Riešením sústavy lineárnych rovníc získame neznáme ***k***, ***l***, ***t***. Po prepísaní rovníc do tvaru rozšírenej matice môžeme vyjadriť parameter ***t***, ktorý nám stačí na získanie súradnice hľadaného bodu:

$$\left(\begin{array}{ccc|c} u_x & v_x & -m_x & d_x-a_x \\ u_y & v_y & -m_y & d_y-a_y \\ u_z & v_z & -m_z & d_z-a_z \end{array}\right)$$

eliminujeme parameter ***k*** z 2. a 3. rovnice použitím 1.rovnice:

$$\left(\begin{array}{ccc|c} u_x & v_x & -m_x & d_x-a_x \\ 0 & v_y-v_x\frac{u_y}{u_x} & -m_y+m_x\frac{u_y}{u_x} & d_y-a_y-(d_x-a_x)\frac{u_y}{u_x} \\ 0 & v_z-v_x\frac{u_z}{u_x} & -m_z+m_x\frac{u_z}{u_x} & d_z-a_z-(d_x-a_x)\frac{u_z}{u_x} \end{array}\right)$$

a nakoniec eliminujeme parameter ***l*** z 3.rovnice použitím 2.rovnice:

$$\left(\begin{array}{ccc|c} u_x & v_x & -m_x & d_x-a_x \\ 0 & v_y-v_x\frac{u_y}{u_x} & -m_y+m_x\frac{u_y}{u_x} & d_y-a_y-(d_x-a_x)\frac{u_y}{u_x} \\ 0 & 0 & -m_z+m_x\frac{u_z}{u_x}-\frac{v_z-v_x\frac{u_z}{u_x}}{v_y-v_x\frac{u_y}{u_x}}\left(-m_y+m_x\frac{u_y}{u_x}\right) & d_z-a_z-(d_x-a_x)\frac{u_z}{u_x}-\frac{v_z-v_x\frac{u_z}{u_x}}{v_y-v_x\frac{u_y}{u_x}}\left(d_y-a_y-(d_x-a_x)\frac{u_y}{u_x}\right) \end{array}\right)$$

z tohto už môžeme vyjadriť ***t***:

$$t=-\frac{(d_x-a_x)(u_y v_z-u_z v_y)+(d_y-a_y)(u_z v_x-u_x v_z)+(d_z-a_z)(u_x v_y-u_y v_x)}{m_x(u_y v_z-u_z v_y)+m_y(u_z v_x-u_x v_z)+m_z(u_x v_y-u_y v_x)}$$

Keď sa na výsledok lepšie pozrieme, zistíme, že vzťah môžeme napísať vo vektorovom tvare:

$t=-\frac{(\boldsymbol{d}-\boldsymbol{a})\cdot(\boldsymbol{u}\times\boldsymbol{v})}{\boldsymbol{m}\cdot(\boldsymbol{u}\times\boldsymbol{v})}=-\frac{(\boldsymbol{d}-\boldsymbol{a})\cdot(|\boldsymbol{u}||\boldsymbol{v}|\sin(\alpha)\boldsymbol{n})}{\boldsymbol{m}\cdot(|\boldsymbol{u}||\boldsymbol{v}|\sin(\alpha)\boldsymbol{n})}=-\frac{(\boldsymbol{d}-\boldsymbol{a})\cdot\boldsymbol{n}}{\boldsymbol{m}\cdot\boldsymbol{n}}$, kde ***n*** je normála roviny určenej ***u***, ***v***. Súradnice hľadaného bodu potom budú $\boldsymbol{b}=\boldsymbol{d}+t\,\boldsymbol{m}$.

Využitie stromu BRP

Kde všade je možné využiť strom binárneho rozdelenia priestoru ? Niektoré z možností sú:

1. Určovanie viditeľnosti
2. Zisťovanie viditeľných častí priestoru
3. Operácie s 3D objektami
4. Vrhanie tieňov

Určovanie viditeľnosti

Pre nejaký priestor zložený z mnohouholníkov sme už vytvorili strom BRP. Ako ho však využiť na zisťovanie viditeľnosti ? Čo ak sa na priestor raz pozeráme z jedného miesta, inokedy z druhého a vždy pod iným uhlom ? Nebude to robiť problémy ? Našťastie, strom BRP je pripravený práve pre tieto prípady. Predstavme si, že na priestor sa pozeráme z ľubovoľného bodu a pozeráme sa smerom, ktorý opíšeme vektorom pohľadu $\boldsymbol{p}$. Musíme teda v správnom poradí vykresliť mnohouholníky scény. Celé vykresľovanie sa pomocou stromu BRP zmení na prechádzanie stromu v poradí **zadná-predná** strana, čo sa veľmi jednoducho programuje pomocou rekurzie.

Algoritmus

1. Vyberie sa počiatočný mnohouholník stromu BRP
2. Zavoláme vykresľovanie počiatočného mnohouholníka

Vykresľovanie aktuálneho mnohouholníka

1. Ak aktuálny mnohouholník má ešte odkaz na svoj "zadný" mnohouholník, potom sa zavolá vykresľovanie tohto mnohouholníka.
2. Vykreslí sa aktuálny mnohouholník
3. Ak aktuálny mnohouholník má odkaz na "predný" mnohouholník, zavolá vykresľovanie "predného" mnohohuholníka.

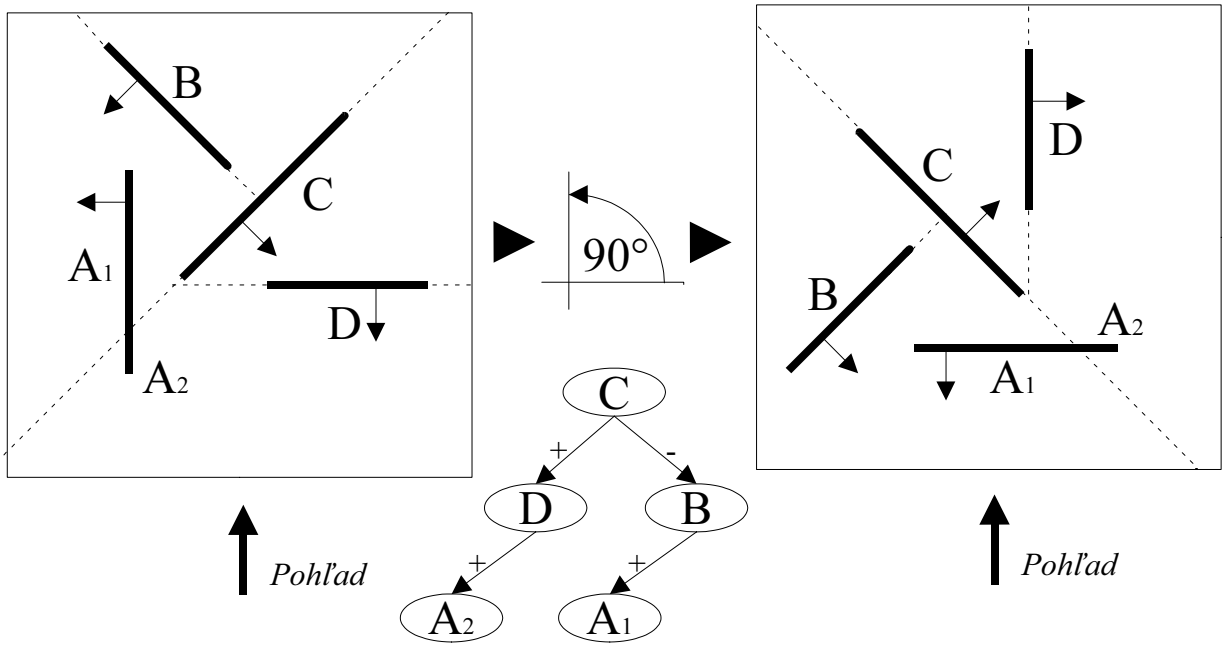
Zostáva už len zistiť, ktorý mnohouholník je "predný", a ktorý "zadný" pri súčasnom pohľade. Vieme, že strom BRP má v každom vrchole informácie o aktuálnom mnohouholníku a odkaz na mnohouholník, ktorý ďalej delí privrátenú časť polpriestoru a odvrátenú časť polpriestoru, vzhľadom na deliacu rovinu. Avšak toto delenie nič nehovorí o tom, ktorá časť je "predná", a ktorá "zadná" pri súčasnom pohľade na priestor.

Ako zistiť, či je polpriestor zadným polpriestorom alebo predným ?

Zistí sa to veľmi jednoducho. Pre každý mnohouholník poznáme normálový vektor $\boldsymbol{n}$, pohľad máme opísaný vektorom $\boldsymbol{p}$, preto nám treba zistiť, aký uhol zvierajú vektory $\boldsymbol{n}$ a $\boldsymbol{p}$. Opäť nám stačí zistiť iba hodnotu skalárneho súčinu týchto dvoch vektorov, $\boldsymbol{p} \cdot \boldsymbol{n}$, podobne ako pri zisťovaní polohy bodu voči deliacej rovine. Ak je výsledok kladný, potom privrátená časť polpriestoru vzhľadom na deliacu rovinu je teraz "zadnou" pri pohľade na polpriestor a naopak. Ak je výsledok záporný, privrátená časť je zároveň "prednou" a odvrátená "zadnou".

Potom už len stačí prechádzať stromom BRP tak, že sa najskôr vykreslia "zadné", a potom "predné" mnohouholníky. Zároveň sa môže ľahko zistiť, či je vôbec potrebné niečo vykresľovať, či by už náhodou vykresľovanie nešlo mimo hranice vytváraného obrazu a spracovávanie by bolo zbytočné.

Ako bude prebiehať vykresľovanie ? Ukážeme si to v dvoch prípadoch na rovnakom priestore, keď sa jednotlivé pohľady líšia miestom pohľadu a tým, že sú v inom uhle (kolmé na seba). Pre oba pohľady pracujeme s rovnakým stromom BRP, meniť sa bude iba vektor pohľadu ***p*** a poradie vykresľovania mnohouholníkov.



Vykresľovanie:

1. Vyberie sa ***C***, zadná rovina obsahuje ***B***
2. Vyberie sa ***B***, ktoré už neobsahuje zadný polpriestor
3. Vykreslí sa ***B***, vyberie sa ***A1***, ktoré sa nachádza v prednom polpriestore ***B***
4. Vykreslí sa ***A1***, vráti sa do ***B***
5. Vráti sa do ***C***, vykreslí ***C***, vyberie sa ***D***
6. ***D*** nemá zadný polpriestor, vykreslí sa, z predného polpriestoru sa vyberie ***A2***
7. ***A2*** nemá zadný polpriestor, vykreslí sa, nemá ani predný polpriestor, vráti sa do ***D***
8. Vráti sa do ***C***
9. Vykresľovanie je ukončené

Mnohouholníky sa vykreslili v poradí:
B, ***A1***, ***C***, ***D***, ***A2***

Vykresľovanie:

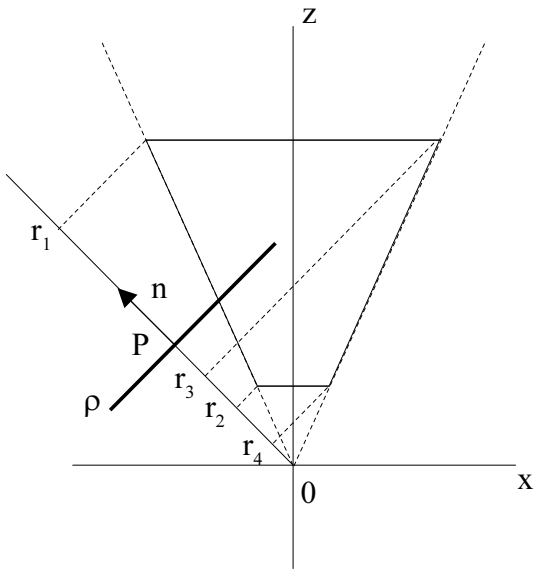
1. Vyberie sa ***C***, zadná rovina obsahuje ***D***
2. Vyberie sa ***D***, ktoré už neobsahuje zadný polpriestor
3. Vykreslí sa ***D***, predný polpriestor obsahuje ***A2***
4. Vykreslí sa ***A2***, vráti sa do ***D***
5. Vráti sa do ***C***, vykreslí ***C***, vyberie sa ***B***
6. ***B*** nemá zadný polpriestor, vykreslí sa, z predného polpriestoru sa vyberie ***A1***
7. ***A1*** nemá zadný polpriestor, vykreslí sa, nemá ani predný polpriestor, vráti sa do ***B***
8. Vráti sa do ***C***
9. Vykresľovanie je ukončené

Mnohouholníky sa vykreslili v poradí:
D, ***A2***, ***C***, ***B***, ***A1***

Možno si všimnúť, že poradie bolo skutočne správne, a že pri vykresľovaní zásluhou delenia mnohouholníkov nevznikali nežiadúce prekrytia mnohouholníkov.

Zisťovanie viditeľných častí priestoru

Pri vykresľovaní obrazu nás napadne, že sa nemusíme zaoberať zbytočne mnohouholníkmi, ktoré nie sú vôbec viditeľné. Pri rozdelení priestoru do stromu BRP to znamená obmedzenie prechádzania stromu tam, kde mnohouholníky určite nie sú viditeľné (nezasahujú do viditeľnej oblasti). Keďže každý vrchol stromu obsahuje mnohouholník, ktorý rozdeľuje daný polpriestor z hľadiska pohľadu na "predný" a "zadný" (polo)polpriestor, ostáva nám zistiť, či tieto vzniknuté polpriestory zasahujú do viditeľnej oblasti. Ak by nejaký z nich nezasahoval, tak aj všetky mnohouholníky, ktoré ležia v tomto polpriestore, nemá zmysel prechádzať, pretože sú neviditeľné. Inými slovami, pokiaľ pri prechádzaní stromu zistíme, že jeden z polpriestorov aktuálneho mnohohuholníka nie je viditeľný, strom týmto smerom už ďalej neprechádzame.



Ako však zistiť, či je polpriestor viditeľný ?

Na obrázku je znázornený prierez viditeľnej oblasti rovinou ***xz***. Deliaci mnohouholník leží v rovine ρ, ktorej normálový vektor je ***n***. Ak bodom ***0*** vedieme priamku, ktorá je rovnobežná s vektorom ***n***, môžeme zistiť priesečník tejto priamky s rovinou ρ - bod ***P***, a taktiež priemety krajných bodov viditeľnej oblasti do tejto priamky - body $\boldsymbol{r_i}$. Potom ak všetky body $\boldsymbol{r_i}$ ležia na rovnakej polpriamke začínajúcej v bode ***P***, je jeden z polpriestorov urèite neviditeľný. Ak všetky body $\boldsymbol{r_i}$ ležia na polpriamke z ***P*** smerom k ***0***, potom je "zadný" polpriestor neviditeľný, ak ležia na polpriamke z ***P*** smerom od ***0***, potom je neviditeľný "predný" polpriestor. Ak sú body $\boldsymbol{r_i}$ na oboch polpriamkach vychádzajúcich z ***P***, sú viditeľné oba polpriestory. Na obrázku je všetko zobrazené v rovine, platí to však aj v trojrozmernom priestore. Keďže viditeľná oblasť je ohraničená ôsmimi bodmi $\boldsymbol{J_i}$ - štyrmi prednými a štyrmi zadnými, musíme urobiť projekciu všetkých týchto bodov do vektora ***n***. Ak máme bod $\boldsymbol{J_i} = (j_x, j_y, j_z)$, potom súradnice $\boldsymbol{j_i}$ jeho priemetu do vektora ***n*** vypočítame ako $\boldsymbol{j_i} = \dfrac{\boldsymbol{J_i} \cdot \boldsymbol{n}}{\boldsymbol{n} \cdot \boldsymbol{n}}$. Normálu máme zvolenú tak, aby $|\boldsymbol{n}| = \boldsymbol{1}$, výsledný vzťah pre zistenie súradnice bodu $\boldsymbol{J_i}$ vo vektore ***n*** bude

$$\boldsymbol{j_i} = \boldsymbol{J_i} \cdot \boldsymbol{n}$$

Podobne súradnicu priesečníka ***P*** roviny ρ a priamky v smere ***n*** prechádzajúcej ***0*** môžeme vyjadriť ako

$$\boldsymbol{p} = \boldsymbol{A} \cdot \boldsymbol{n} \quad ,$$

kde ***A*** je ľubovoľný bod ležiaci na deliacom mnohouholníku (napríklad jeden z jeho krajných bodov). Potom pre viditeľnosť jednotlivých polpriestorov môžeme zapísať

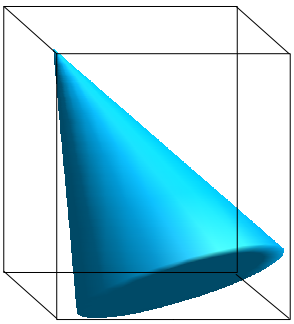
$\boldsymbol{p} \leqslant \boldsymbol{j_i}, i = 1..8$, "predný" polpriestor je neviditeľný
$\boldsymbol{p} \geqslant \boldsymbol{j_i}, i = 1..8$, "zadný" polpriestor je neviditeľný
inak sú oba polpriestory viditeľné

Obálka objektu

Pri narábaní s objektom je výhodnejšie "zabaliť" ho do jednoduchého geometrického útvaru, s ktorým sa ľahšie narába. Všetky transformácie sa potom môžu vykonať na tomto obalovom útvare, a pokiaľpo ich aplikovaní tento obal zasahuje do viditeľnej oblasti, môžu sa vykonať aj na pôvodnom objekte. Je zrejmé, že pri väčšom počte objektov, ktoré sú nerovnomerne roztrúsené po priestore tento spôsob ušetrí mnoho zbytočných výpočtov. Je tu však problém, aký typ obalu vybrať a ako to urobiť. Ak sa budeme snažiť čo najvernejšie opísať tvar objektu, napríklad pomocou konvexného obalu objektu (čiže zjednotením všetkých možných mnohouholníkov medzi bodmi objektu), "obal" objektu bude priveľmi zložitý a počtom bodov, ktoré ho opisujú, sa bude blížiť pôvodnému objektu. Ak sa rozhodneme vybrať jednoduchší tvar, riskujeme, že obal bude priveľmi voľný a jeho veľká časť bude obaľovať voľný priestor mimo telesa.

Zvolenie tvaru obálky

Rozsahový kváder



Najčastejšie sa možno stretnúť s obalením do kvádra, ktorý sa získa tak, že sa nájdu minimálne a maximálne súradnice bodov v jednotlivých rovinách súradnicovej sústavy a z nich sa vygeneruje kváder, ktorého steny sú rovnobežné s týmito rovinami. Tento spôsob však nie je veľmi vhodný, pretože pri náklone objektu vzniká veľký priestor, ktorý by sa pri zisťovaní viditeľnosti musel zbytočne brať do úvahy, aj keď neobaľuje žiadnu časť objektu.

Avšak tvar kvádra je pre svoju jednoduchosť veľmi výhodný, a preto ďalšia metóda bude používať opäť kváder, avšak bude sa ho snažiť orientovať tak, aby čo najvernejšie opísal tvar telesa.

Orientovaný kváder

Orientovaný kváder získame tak, že sa budeme snažiť nájsť trojicu vzájomne kolmých vektorov, ktorá stochasticky najvernejšie vystihuje objekt. Pokúsime sa nájsť nájsť bázu trojrozmerného priestoru, v ktorej má objekt minimálnu možnú koreláciu medzi súradnicami. Po získaní tejto bázy budeme viesť roviny kolmé na tieto vektory, ktoré čo najtesnejšie priliehajú k objektu. Pre získanie hľadanej bázy využijeme následovné princípy.

V Hilbertovom priestore $\boldsymbol{l}_2(\Omega, \phi, P)$ môžeme centrovaný diskrétny náhodný proces $\boldsymbol{f} = (f(\omega,0), f(\omega,1), ...), \omega \in \Omega$, kde (Ω, ϕ, P) je pravdepodobnostným priestorom a skalárny súčin je definovaný ako $\boldsymbol{f} \cdot \boldsymbol{g} = E\left(\sum_{i=0}^{\infty} f(\omega,i)\overline{g(\omega,i)}\right)$, rozložiť do ortogonálnej bázy $\{\boldsymbol{b}_i, i = 0,1,...\}$, pričom $\boldsymbol{f} = \sum_{i=0}^{\infty} c_i \boldsymbol{b}_i$ a $c_i = \frac{\boldsymbol{f} \cdot \boldsymbol{b}_i}{\boldsymbol{b}_i \cdot \boldsymbol{b}_i}$. Ortogonálnu bázu tvorí sústava vzájomne nekorelovaných náhodných procesov $\boldsymbol{b}_i$. Ak bázické náhodné procesy vyjadríme pomocou súčinu deterministického vektora z priestoru $\boldsymbol{l}_2$ a náhodnej premennej, t.j. $\boldsymbol{b}_i = (b_i(\omega,k)) = (\xi_i(\omega) \cdot b_i(k)), \omega \in \Omega, i,k = 0,1,..., b_i(k) \in \boldsymbol{l}_2$ a označíme $C_i(\omega) = c_i \xi_i(\omega)$, rozklad do ortogonálnej bázy bude mať tvar $f(\omega,k) = \sum_{i=0}^{\infty} C_i(\omega) b_i(k), k = 0,1,...$.

Ekvivalent vety G. Browna ml. pre priestor $\boldsymbol{l_2}(\Omega, \phi, P)$ hovorí, že systém $\{ b_i(\omega, k) = C_i(\omega) \cdot b_i(k), i, k = 0, 1, ... \}$ je úplný vtedy, keď $\{ b_i(k), i, k = 0, 1, ... \}$ je bázou priestoru $\boldsymbol{l_2}$ realizácií náhodného procesu ***f***. Navyše, ak je ortogonálna báza priestoru $\boldsymbol{l_2}$, je ortogonálna aj zodpovedajúca báza v priestore $\boldsymbol{l_2}(\Omega, \phi, P)$.

Chceme nájsť bázu $\{ b_i(k), i, k = 0, 1, ... \}$, ktorá zaručuje nekorelovanosť koeficientov $C_i(\omega), C_j(\omega), i \neq j$, čiže $E\{ C_i(\omega) \cdot \overline{C_j(\omega)} \} = \sigma_j^2 \cdot \delta_{ij}$, kde $\delta_{ij} = 0, i \neq j; \delta_{ij} = 1, i = j$ a $\sigma_j^2 = E\{ C_j(\omega)^2 \}$. Ak označíme $E_i = \sum_{k=0}^{\infty} b_i(k) \overline{b_i(k)}, i = 0, 1, ...$, koeficienty rozkladu budú mať tvar $C_i(\omega) = \frac{1}{E_i} \sum_{k=0}^{\infty} f(\omega, k) b_i(k), i = 0, 1, ...$. Ak výsledok dosadíme do vzťahu, vyjadrujúceho nekorelovanosť koeficientov, získavame

$$E\left\{ \frac{1}{E_i} \sum_{k=0}^{\infty} f(\omega, k) \overline{b_i(k)} \overline{\frac{1}{E_j} \sum_{k=0}^{\infty} f(\omega, k) \overline{b_j(k)}} \right\} = \sigma_j^2 \cdot \delta_{ij}$$

a z toho $E_i E_j \sigma_j^2 \cdot \delta_{ij} = E\left\{ \sum_{k=0}^{\infty} f(\omega, k) \overline{b_i(k)} \overline{\sum_{k=0}^{\infty} f(\omega, k) \overline{b_j(k)}} \right\}$. Po ďalších úpravách dostávame

$$E_i E_j \sigma_j^2 \cdot \delta_{ij} = E\left\{ \sum_{k=0}^{\infty} \left(f(\omega, k) \overline{b_i(k)} \sum_{l=0}^{\infty} \overline{f(\omega, l)} b_j(l) \right) \right\},$$

$$E_i E_j \sigma_j^2 \cdot \delta_{ij} = \sum_{k=0}^{\infty} \overline{b_i(k)} \sum_{l=0}^{\infty} E\{ f(\omega, k) \overline{f(\omega, l)} \} b_j(l) \cdot$$

Keďže platí $E_i \delta_{ij} = \sum_{i=0}^{\infty} \overline{b_i(k)} b_j(k)$, po porovnaní s predošlým vzťahom dostávame

$$\sum_{k=0}^{\infty} \overline{b_i(k)} b_j(k) E_j \sigma_j^2 = \sum_{k=0}^{\infty} \overline{b_i(k)} \sum_{l=0}^{\infty} E\{ f(\omega, k) \overline{f(\omega, l)} \} b_j(l), \text{ čiže}$$

$$b_j(k) E_j \sigma_j^2 = \sum_{l=0}^{\infty} E\{ f(\omega, k) \overline{f(\omega, l)} \} b_j(l), k = 0, 1, ...$$

Ak zadefinujeme maticu kovariancie $\boldsymbol{R} = (R_{ij}) = (E\{ f(\omega, i) \cdot \overline{f(\omega, j)} \})$, platí $\sum_{i,j=0}^{\infty} R_{ij} < \infty$ a označíme $E_j \sigma_j^2 = \lambda_j$, potom predošlý vzťah môžeme vyjadriť ako $\lambda_j b_j(k) = b_j R_k, k = 0, 1, ...$, kde R_k je k-ty stĺpec kovariančnej matice. Dostali sme teda návod ako vypočítať zložku vektora. Samotný vektor vypočítame zo vzťahu $\lambda_j \boldsymbol{b_j} = \boldsymbol{b_j R}$. Bázické vektory $\boldsymbol{b_j}$ sú teda vlastnými vektormi kovariančnej matice ***R***.

Získali sme ortogonálnu bázu, ktorá nám stochasticky najlepšie vystihuje náše dáta. Ako ich však využiť pri vytváraní obálky ? V obálke síce ide o konečnorozmerný priestor, ale získané vektory nám dajú napriek tomu bázu s minimálnou chybou.

Vytvorenie orientovaného kvádra

Najskôr vypočítame kovarianciu súradníc jednotlivých bodov objektu. Keďže $\sigma_{XY} = cov(X, Y) = E((X - E(X))(Y - E(Y))) = E(XY) - E(X)E(Y)$ a

$E(XY)=\sum_{i=0}^{n-1}\frac{X_i Y_i}{n}$, matica kovariancie bude mať tvar $\boldsymbol{R}=\begin{pmatrix}\sigma_x^2 & \sigma_{xy} & \sigma_{xz}\\ \sigma_{xy} & \sigma_y^2 & \sigma_{yz}\\ \sigma_{xz} & \sigma_{yz} & \sigma_z^2\end{pmatrix}$. Jedná sa o symetrickú maticu stupňa 3, preto bude mať reálne vlastné čísla (korene charakteristickej rovnice) a reálne vlastné vektory. Korene charakteristickej rovnice sa dajú ľahko vypočítať (napríklad pomocou Cardanových vzorcov) a vlastné vektory by až na niekoľko špeciálnych prípadov nemali tiež robiť problémy. Po ich výpočte získavame tri navzájom kolmé vektory. Tieto využijeme na zostrojenie obálky, a to tak, že objekt vycentrujeme $(E(X)=E(Y)=E(Z)=0)$ a hľadáme kolmé priemety bodov objektu na priamky v smere týchto vektorov. Minimálne a maximálne súradnice priemetov bodov do týchto priamok si odložíme a vedieme v nich roviny kolmé na príslušné priamky. V priesečníkoch týchto rovín vznikne 8 bodov, ktoré dokopy tvoria obálku v tvare kvádra. Tie potom posunieme o pôvodné stredné hodnoty súradníc bodov objektu, aby korešpondovali s pôvodným, nevycentrovaným objektom.

Zistenie priemetov bodov do priamok

Ak máme tri kolmé vektory ***a***, ***b***, ***c*** a chceme nájsť kolmý priemet bodu ***v*** do priamok v smere týchto vektorov, stačí využiť to, že tento priesečník leží na priamke určenej napríklad vektorom ***a*** (dá sa vyjadriť ako ***p*** = *k****a***) a vektor od bodu ***v*** ku bodu ***p*** musí byť kolmý na vektor ***a***.

Teda $(\boldsymbol{v}-\boldsymbol{p})\cdot\boldsymbol{a}=0$ a z toho $k=\frac{\boldsymbol{v}\cdot\boldsymbol{a}}{\boldsymbol{a}\cdot\boldsymbol{a}}$. Podobne to platí aj pre ostatné priemety. Nakoniec získame šesticu bodov, ktoré označujú maximálne a minimálne priemety $\boldsymbol{A}_{min}, \boldsymbol{A}_{max}, \boldsymbol{B}_{min}, \boldsymbol{B}_{max}, \boldsymbol{C}_{min}, \boldsymbol{C}_{max}$. Týmito bodmi vedieme roviny, ktoré sú kolmé na príslušné vektory (tie sú zároveň ich normálami).

Zistenie bodov obálky

Body obálky ležia na priesečníkoch týchto rovín, stačí teda získať tieto priesečníky. Ak máme vektory vytvorené tak, aby platilo $\boldsymbol{a}\times\boldsymbol{b}=\boldsymbol{c}, \boldsymbol{b}\times\boldsymbol{c}=\boldsymbol{a}, \boldsymbol{c}\times\boldsymbol{a}=\boldsymbol{b}, |\boldsymbol{a}|=|\boldsymbol{b}|=|\boldsymbol{c}|=1$ a zvolíme ľubovoľné tri body z priemetov do všetkých priamok - napríklad ***A***, ***B***, ***C***, môžeme označiť polohu priesečníku ***X*** pomocou troch rovníc $(\boldsymbol{X}-\boldsymbol{A})\cdot\boldsymbol{a}=0$, $(\boldsymbol{X}-\boldsymbol{B})\cdot\boldsymbol{b}=0$ a $(\boldsymbol{X}-\boldsymbol{C})\cdot\boldsymbol{c}=0$. Ak označíme $k=\boldsymbol{A}\cdot\boldsymbol{a}$, $l=\boldsymbol{B}\cdot\boldsymbol{b}$ a $m=\boldsymbol{C}\cdot\boldsymbol{c}$, potom môžeme riešiť nasledujúcu sústavu:

$$\left(\begin{array}{ccc|c}a_x & a_y & a_z & k\\ b_x & b_y & b_z & l\\ c_x & c_y & c_z & m\end{array}\right)$$

Riešením dostávame

$$x=\frac{k(b_y c_z-b_z c_y)+l(a_z c_y-a_y c_z)+m(a_y b_z-a_z b_y)}{\boldsymbol{a}\cdot(\boldsymbol{b}\times\boldsymbol{c})}$$

$$y=\frac{k(b_x c_z-b_z c_x)+l(a_z c_x-a_x c_z)+m(a_x b_z-a_z b_x)}{\boldsymbol{a}\cdot(\boldsymbol{b}\times\boldsymbol{c})}$$

$$z=\frac{k(b_x c_y-b_y c_x)+l(a_y c_x-a_x c_y)+m(a_x b_y-a_y b_x)}{\boldsymbol{a}\cdot(\boldsymbol{b}\times\boldsymbol{c})}$$

Po využití predošlých vzťahov ako aj vzájomných vektorových súčinov, dostávame $x=(k,l,m)\cdot(a_x,b_x,c_x)$, $y=(k,l,m)\cdot(a_y,b_y,c_y)$ a $z=(k,l,m)\cdot(a_z,b_z,c_z)$. Ak označíme $\boldsymbol{u}=(k,l,m)$, $\boldsymbol{v_x}=(a_x,b_x,c_x)$, $\boldsymbol{v_y}=(a_y,b_y,c_y)$, $\boldsymbol{v_z}=(a_z,b_z,c_z)$, súradnice priesečníka sú

$$x = \boldsymbol{u} \cdot \boldsymbol{v}_x + E(x)$$
$$y = \boldsymbol{u} \cdot \boldsymbol{v}_y + E(y)$$
$$z = \boldsymbol{u} \cdot \boldsymbol{v}_z + E(z)$$

kde *E(x)*, *E(y)* a *E(z)* sú stredné hodnoty súradníc pôvodného, nevycentrovaného objektu.

Na získanie všetkých bodov obálky stačí tento postup opakovať pre každu kombináciu ***A***, ***B***, ***C***.

Zisťovanie viditeľnosti obálky

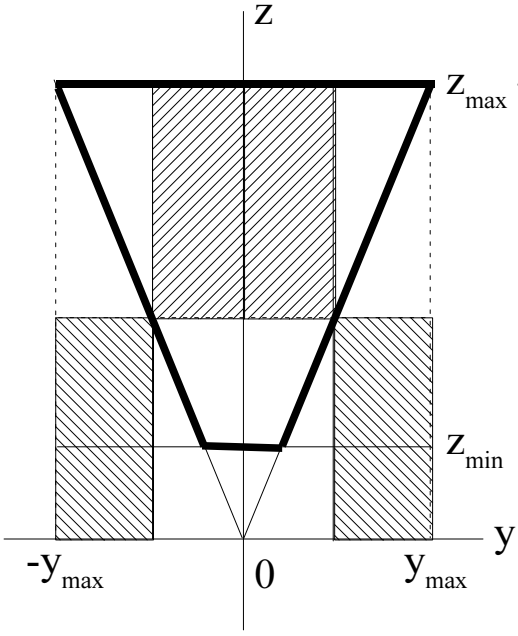
Po obalení telesa obálkou nastáva problém, ako zistiť, či je obálka viditeľná (či zasahuje do viditeľnej oblasti). Toto je možné dosiahnuť viacerými spôsobmi, preto tu budú ukázané tri možnosti:

1. Zisťovanie vzájomných polôh bodov, úsečiek a rovín
2. Premietanie na oddeľujúce osi
3. Rozšírenie viditeľnej oblasti v závislosti na rozmere a tvare obálky

Zisťovanie vzájomných polôh bodov, úsečiek a rovín

Ak chceme zistiť, či sa obálka nachádza nejakou časťou vo vnútri viditeľnej oblasti, stačí nám zistiť, či sa v tejto oblasti nachádza aspoň jeden jej bod. Najskôr sa treba pokúsiť zistiť, či sa aspoň jeden krajný bod obálky nachádza vo viditeľnej oblasti. Ak sa nenachádza, je niekedy nutné zistiť, èi priesečníky úsečiek obálky a rovín viditeľnej oblasti nemajú spoločné body, ktoré by ležali vnútri viditeľnej oblasti. Pokiaľ by ani jeden bod nebol vnútri oblasti, je možné, že celá obálka by obaľovala viditeľnú oblasť a bolo by naopak potrebné zistiť, či sa krajné body viditeľnej oblasti nenachádzajú v obálke.

Zistenie polohy krajných bodov



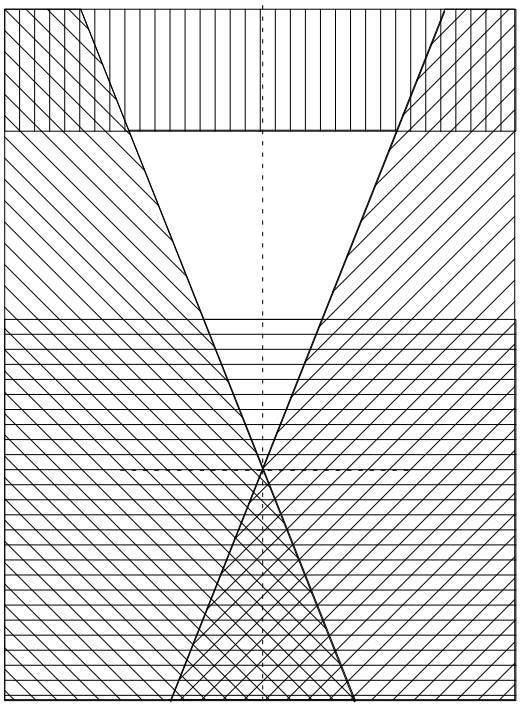
Na obrázku je zobrazený prierez viditeľnou oblasťou v rovine ***yz***, podobný charakter má však aj prierez v rovine ***xz***. Chceme zistiť, či sa aspoň jeden krajný bod obálky nachádza v tejto oblasti. Vidíme, že ak má bod $\boldsymbol{B} = (b_x, b_y, b_z)$ ležať vo viditeľnej oblasti, potom musí $b_z \in \langle z_{min}, z_{max} \rangle$, $|b_x| \leqslant b_z \, \boldsymbol{tg} \frac{\boldsymbol{\alpha}}{\boldsymbol{2}}$ a $|b_y| \leqslant b_z \, \boldsymbol{tg} \frac{\boldsymbol{\beta}}{\boldsymbol{2}}$. Pri týchto testoch však používame násobenie. Aby sme si testovanie o niečo urýchlili, môžeme využiť to, že žiaden viditeľný bod nebude mať $|b_x| > x_{max}$ a $|b_y| > y_{max}$, kde $x_{max} = z_{max} \, \boldsymbol{tg} \frac{\boldsymbol{\alpha}}{\boldsymbol{2}}$ a $y_{max} = z_{max} \, \boldsymbol{tg} \frac{\boldsymbol{\beta}}{\boldsymbol{2}}$. Zároveň môžeme využiť to, že ak pre bod ***B*** platí $(|b_x|, |b_y|, b_z) \in \langle 0, \frac{x_{max}}{2} \rangle \times \langle 0, \frac{y_{max}}{2} \rangle \times \langle \frac{z_{max}}{2}, z_{max} \rangle$, potom určite leží vo viditeľnej oblasti. Ak navyše platí $(|b_x|, |b_y|, b_z) \in \langle \frac{x_{max}}{2}, x_{max} \rangle \times \langle \frac{y_{max}}{2}, y_{max} \rangle \times \langle 0, \frac{z_{max}}{2} \rangle$, bod ***B*** leží mimo viditeľnej oblasti.

Týmto môžeme urýchliť spracovávanie veľkého počtu obálok, keďže pre mnoho bodov budú platiť už tieto zjednodušené podmienky. Takisto môžeme využiť to, že ak všetky body ležia v jednom a

tom istom vonkajšom polpriestore ľubovoľnej zo šiestich rovín, ktoré ohraničujú viditeľnú oblasť, potom je celá obálka mimo viditeľnej oblasti a nemá zmysel robiť ďalej testy polohy úsečiek obálky. Inými slovami, ak každému bodu priradíme čísla vonkajších polpriestorov, do ktorých patrí, potom ak všetky body majú nejaké spoločné číslo zo všetkých im priradených, nachádzajú sa v spoločnom polpriestore.

Zistenie priesečníkov priamok a rovín viditeľnej oblasti

Ak zistíme, že ani jeden bod nepatrí do viditeľnej oblasti, a zároveň body neležia v spoločnom polpriestore, nedá sa vylučiť to, že minimálne jedna úsečka medzi krajnými bodmi môže mať priesečník s hraničnou rovinou viditeľnej plochy a tento bod leží vnútri viditeľnej oblasti.



Na to, aby sme boli schopní efektívne zisťovať možné prieniky úsečiek a rovín, potrebujeme určiť, ktoré roviny môže úsečka vedená z jedného bodu do druhého preťať. Preto si rozdelíme priestor na niekoľko prelínajúcich sa polpriestorov, ktoré budú vyjadrovať vonkajšie strany rovín viditeľnej oblasti. Každému z týchto polpriestorov priradíme unikátne číslo v tvare 2^i. O polpriestoroch delených rovinami s rovnakými z-ovými súradnicami (sú také dve - pri z_{min} a z_{max}) budeme ďalej hovoriť ako o prednom a zadnom polpriestore, o tých, ktoré vznikajú v rovine ***xz*** ako o ľavom a pravom polpriestore a o tých, ktoré vznikajú pri pohľade na rovinu ***yz*** ako o hornom a dolnom polpriestore (na obrázku je nešpecifikovaná rovina, vodorovné a zvislé čiary udávajú predný a zadný polpriestor, šikmo vyšrafované ľavý (dolný) a pravý (horný) polpriestor v závislosti od zvolenej roviny). Každému bodu obálky priradíme číslo, ktoré je súčtom čísel polpriestorov, do ktorých tento bod patrí. Potom ľubovoľná úsečka z jedného bodu do druhého bodu pretína tie roviny, ktorých čísla obsahuje logický súčet čísel priradených koncovým bodom. Napríklad ak si zvolíme čísla polpriestorov (rovín) a vyjadríme ich v dvojkovej sústave takto: predný *1*, zadný *10*, ľavý *100*, pravý *1000*, spodný *10000* a horný *100000*, potom ak máme prvý bod úsečky, ktorý leží v ľavo-prednom a druhý bod v pravo-zadno-hornom polpriestore, ich čísla budú *101* a *101010*, úsečka medzi týmito dvomi bodmi pretína všetky roviny okrem spodnej, keďže logický súčet týchto čísel je *101111*. Môžeme však využiť opäť to, že ak tieto čísla majú logický súčin iný ako *0*, potom sa nachádzajú v spoločnom polpriestore, a preto nemajú prienik s viditeľnou oblasťou. Ďalej sa dá použiť to, že ak majú body úsečky čísla *1* a *10*, *100* a *1000* alebo *10000* a *100000*, ležia vlastne "oproti" sebe a úsečka medzi nimi určite prechádza viditeľnou oblasťou. Ak sa nedá využiť ani jedna z týchto vlastností, musíme zistiť všetky priesečníky úsečky so získanými rovinami a otestovať, či sa tieto nachádzajú vo vnútri viditeľnej oblasti. Pre prednú a zadnú rovinu pre získanie súradníc viac-menej stačí dosadiť z-ovu súradnicu roviny do rovnice úsečky. Pre inú, napríklad pravú rovinu musí platiť $\boldsymbol{A} + e\boldsymbol{v} = (f\,\boldsymbol{tg}\frac{\boldsymbol{\alpha}}{\boldsymbol{2}}, a_y + e v_y, a_z + e v_z)$, kde $\boldsymbol{A} = (a_x, a_y, a_z)$ je začiatočný bod úsečky, $\boldsymbol{v} = (v_x, v_y, v_z)$ je vektor úsečky a *e*, *f* sú hľadané čísla, opisujúce prienik. Tento vzťah stačí riešiť v rovine, keďže jedna súradnica roviny v priestore je závislá od ďalších dvoch. Môžeme však využiť to, že ak *z* je z-ova súradnica priesečníka, platí

$a_x + q\,v_x = z\,\mathbf{tg}\frac{\alpha}{2} = (a_z + q\,v_z)\,\mathbf{tg}\frac{\alpha}{2}$, z toho $q = \dfrac{a_x - a_z\,\mathbf{tg}\frac{\alpha}{2}}{v_z\,\mathbf{tg}\frac{\alpha}{2} - v_x}$ a súradnice priesečníka budú $\boldsymbol{A} + q\,\boldsymbol{v}$. Nakoniec môžeme výpočet q zhrnúť do nasledujúcej tabuľky:

Ľavá rovina: $q = -\dfrac{a_x + a_z\,\mathbf{tg}\frac{\alpha}{2}}{v_z\,\mathbf{tg}\frac{\alpha}{2} + v_x}$ Pravá rovina: $q = \dfrac{a_x - a_z\,\mathbf{tg}\frac{\alpha}{2}}{v_z\,\mathbf{tg}\frac{\alpha}{2} - v_x}$

Spodná rovina: $q = -\dfrac{a_y + a_z\,\mathbf{tg}\frac{\beta}{2}}{v_z\,\mathbf{tg}\frac{\beta}{2} + v_y}$ Horná rovina: $q = \dfrac{a_y - a_z\,\mathbf{tg}\frac{\beta}{2}}{v_z\,\mathbf{tg}\frac{\beta}{2} - v_y}$

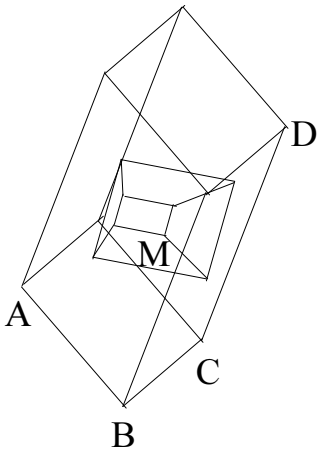
Predná rovina: $q = \dfrac{z_{min} - a_z}{v_z}$ Zadná rovina: $q = \dfrac{z_{max} - a_z}{v_z}$

Súradnice priesečníka budú $\boldsymbol{A} + q\,\boldsymbol{v}$.

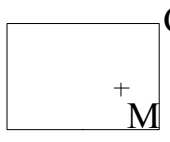
Teraz už stačí iba zistiť, či priesečniky ležia vo viditeľnej rovine, na čo stačí použiť ten istý spôsob, ako pri zisťovaní polohy krajného bodu obálky voči viditeľnej oblasti.

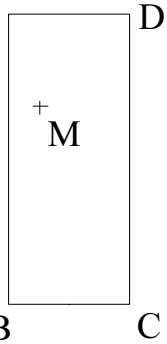
Obsiahnutosť viditeľnej oblasti v obálke

Aj keď všetky predchádzajúce spôsoby nenájdu spoločný bod, môže nastať situácia, že obálka obsahuje celú viditeľnú oblasť. Vtedy nám stačí zobrať ľubovoľný bod viditeľnej oblasti - ***M*** a pokúsiť sa zistiť, či sa nachádza vnútri obálky.



Keďže obálka má tvar kvádra, dá sa využiť skutočnosť, že ak kolmé priemety bodu na ľubovoľné dve navzájom kolmé steny obálky sa nachádzajú vnútri obdĺžnikov s rozmermi stien, potom sa samotný bod nachádza vnútri obálky. Ak si teda zvolíme ľubovoľné štyri body ***A***, ***B***, ***C*** a ***D*** tak, aby určovali dve na seba kolmé roviny, potom úsečky ***AB***, ***BC*** a ***CD*** tvoria ortogonálnu bázu trojrozmerného priestoru a projekcie bodu ***M*** do nich nám dajú jednoznačné, vzájomne nesúvisiace súradnice v tejto báze. Ak všetky súradnice budú z intervalu $\langle 0\,,1 \rangle$, potom bod určite patrí do vnútra obálky.





Jednotlivé súradnice vypočítame pomocou vzťahov

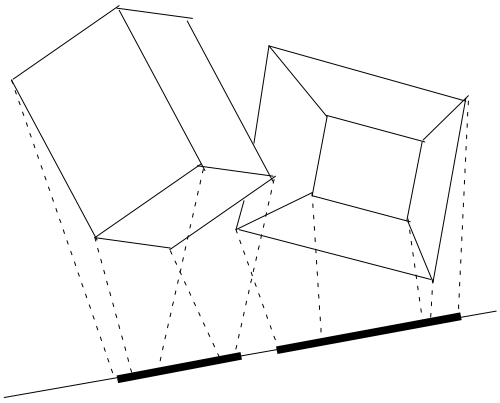
$$M_{AB} = \frac{\boldsymbol{M} \cdot \boldsymbol{AB}}{\boldsymbol{AB} \cdot \boldsymbol{AB}}$$

$$M_{BC} = \frac{\boldsymbol{M} \cdot \boldsymbol{BC}}{\boldsymbol{BC} \cdot \boldsymbol{BC}}$$

$$M_{CD} = \frac{\boldsymbol{M} \cdot \boldsymbol{CD}}{\boldsymbol{CD} \cdot \boldsymbol{CD}}$$

Premietanie na oddeľujúce osi

Predchádzajúci spôsob využíval *presné* súradnice priesečníkov úsečiek s rovinami viditeľnej oblasti a ak sa tieto priesečníky nachádzali vo vnútri viditeľnej oblasti, vyplývalo z toho to, že samotná obálka sa nejakou časťou nachádza vnútri tejto oblasti. Avšak pre zistenie, či sa dva konvexné polyédre pretínajú, nemusíme poznať presné súradnice ich vzájomných priesečníkov. Stačí nám využiť to, že ak sa dva konvexné polyédre nepretínajú, určite existuje rovina, ktorá rozdeľuje priestor na dva polpriestory, z ktorých každý obsahuje jeden z polyédrov. Na zistenie, či rovina oddeľuje dva polyédre stačí premietnuť všetky body polyédrov na priamku rovnobežnú s normálou roviny (oddeľujúcu os). Súradnice bodov polyédra na tejto priamke vytvoria uzavretý interval, každý polyéder jeden. Ak sa tieto intervaly neprekrývajú (nemajú spoločné body), potom rovina je skutočne deliacou rovinou oboch polyédrov (rozdeľuje oba polyédre do dvoch disjunktných polpriestorov) a oba polyédre sa nepretínajú. Ťažším problémom je však nájsť takúto rovinu alebo vymedziť konečný počet rovín, ktoré môžu byť deliacimi rovinami dvoch konvexných polyédrov. Množina týchto rovín musí byť dostatočnou v tom zmysle, že ak ani jedna z týchto rovín nie je deliacou rovinou dvoch konvexných polyédrov, neexistuje ani žiadna iná deliaca rovina.



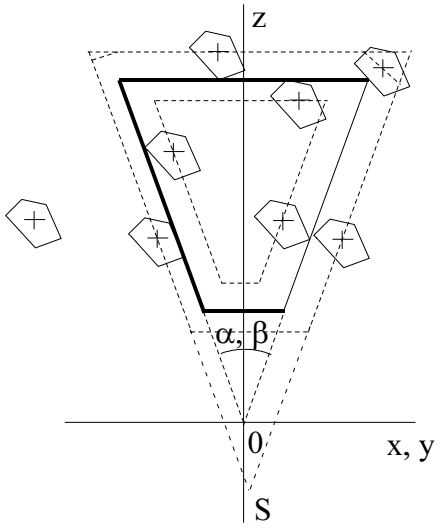
Pre konvexné polyédre sa ukazuje byť dostatočnou množina všetkých povrchových rovín oboch polyédrov a rovín, ktoré sú rovnobežné s ich hranami. Potom určite rovina, ktorá je rovnobežná s nejakou rovinou z tejto množiny rozdeľuje oba polyédre (pokiaľ nemajú spoločný prienik). Ak z každého polyédra použijeme po jednej hrane a z nich vytvoríme rovinu, dostaneme rovinu, ktorá je určite rovnobežná s dvomi hranami, po jednej v každom polyédre (s tými, z ktorých je tvorená), a teda môže byť použitá na testovanie.
Pre prípad obálky a viditeľnej oblasti vidno, že jedinečných povrchových rovín obálky je 3, viditeľnej oblasti 5, unikátnych smerov hrán obálky je 3, viditeľnej oblasti 6. Z týchto smerov hrán môžeme vytvoriť najviac 3 * 6 = 18 rovín, ktoré sú rovnobežné s jednou hranou viditeľnej oblasti a jednou hranou obálky. Spolu teda máme 26 rovín, ktoré máme vyskúšať. Pre každú rovinu musíme vypočítať normálový vektor $\boldsymbol{n}$ (napríklad vektorovým súčinom dvoch hrán, tvoriacich rovinu), a potom urobiť projekciu všetkých bodov polyédrov na priamku rovnobežnú s vektorom $\boldsymbol{n}$.

Ak p_i sú súradnice bodov obálky $\boldsymbol{P_i}$ vo vektore $\boldsymbol{n}$ v tvare $p_i = \frac{\boldsymbol{P}_i \cdot \boldsymbol{n}}{\boldsymbol{n} \cdot \boldsymbol{n}}$ a podobne v_j sú súradnice krajných bodov $\boldsymbol{V_j}$ viditeľnej oblasti vo vektore $\boldsymbol{n}$ v tvare $v_i = \frac{\boldsymbol{V}_i \cdot \boldsymbol{n}}{\boldsymbol{n} \cdot \boldsymbol{n}}$, potom ak p_{min} a p_{max} sú mimimom a maximom z hodnôt p_i, v_{min} a v_{max} sú minimom a maximom z v_i, rovina s normálovým vektorom $\boldsymbol{n}$ navzájom oddeľuje obálku a viditeľnú oblasť vtedy, ak $\langle p_{min}, p_{max} \rangle \cap \langle v_{min}, v_{max} \rangle = \emptyset$. Toto zisťovanie musíme vykonávať s každou rovinou z danej množiny až pokiaľ nenájdeme takú rovinu, pre ktorú by predošlá podmienka vyhovovala. Vtedy by sme našli jednu deliacu rovinu. Pokiaľ by sme prešli všetky roviny z množiny a napriek tomu by

predošlá podmienka nebola splnená ani pri jednej z mich, potom je isté, že obálka a viditeľná oblasť majú spoločný prienik.

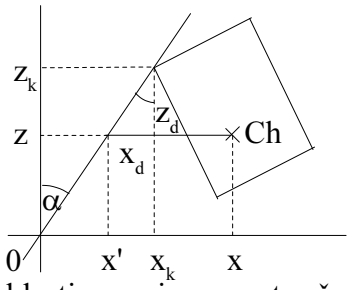
Rozšírenie viditeľnej oblasti v závislosti na rozmeroch a tvare obálky

Ďalší spôsob na rozdiel od ostatných spôsobov umožňuje zistením polohy jediného bodu obálky určiť, či je obálka viditeľná, či leží na okraji alebo či leží mimo viditeľnej oblasti. Ak si zvolíme ľubovoľný bod nachádzajúci sa v obálke a túto obálku potom posúvame po všetkých rovinách viditeľnej oblasti z vnútornej i vonkajšej strany tak, aby sa ich dotýkala, zvolený bod vytvára dve nové oblasti - *vnútornú* a *vonkajšiu*. Potom ak tento bod leží mimo vonkajšej oblasti, celá obálka leží mimo viditeľnej oblasti a s objektom nemusíme vôbec pracovať. Ak leží vo vnútornej oblasti, obálka leží úplne vo viditeľnej oblasti a objekt môžeme vykresľovať bez toho, aby sme museli orezávať mnohouholníky, z ktorých je zložený. Ak leží medzi vnútornou a vonkajšou oblasťou, je veľká pravdepodobnosť, že obálka leží na rozhraní viditeľnej oblasti a mnohouholníky objektu by museli byť podrobené orezávaniu. O obálke však vtedy nemôžeme s úplnou istotou tvrdiť, že vôbec zasahuje do viditeľnej oblasti, pretože vonkajšiu oblasť kvôli uľahčeniu vytvoríme iba zo šiestich vonkajších rovín, ktoré sú rovnobežné s rovinami viditeľnej oblasti. Skutočný tvar vonkajšej oblasti by sa od zjednodušenej" mohol líšiť tým, že by mal "zrezané" rohy. Zjednodušená" vonkajšia oblasť teda ohraničuje celú vonkajšiu oblasť a v rohoch kúsok oblasti, ktorá už do vonkajšej oblasti nepatrí. Vnútorná oblasť tento problém nemá, pretože určite vznikne šesť rovnobežných rovín.



Ak sa pozrieme na obrázok, vidíme, že všetky vzniknuté roviny sú posunuté o určitú hodnotu v smere súradnicovej osi od rovín viditeľnej oblasti. Ostáva nám teda zistiť veľkosť tohto posunu.

Zvoľme si ľubovoľný bod ležiaci v obálke - bod $\boldsymbol{Ch} = (x, y, z)$. Chceme zistiť, aký je veľký posun pravej roviny viditeľnej oblasti. Zaujíma nás vlastne veľkosť x_d , ktorá nám hovorí, na akú vzdialenosť môžeme posunúť bod ***Ch*** doľava, aby sa obálka dotýkala pravej roviny. Na to, aby sme ju zistili, musíme prejsť všetky krajné body obálky a nájsť maximálny posun roviny doprava zo všetkých posunov. Veľkosť tohto posunu od bodu ***Ch*** pri každom krajnom bode obálky zistíme, ak využijeme smer priamky patriacej pravej rovine viditeľnej oblasti v rovine ***xz*** a to, že tento krajný bod na nej musí ležať. Ak (x_k, y_k, z_k) sú súradnice krajného bodu obálky a a je priemet zorného uhla do roviny ***xz***, potom môžeme napísať $\boldsymbol{tg}\frac{\boldsymbol{\alpha}}{\boldsymbol{2}} = \frac{x_k - x'}{z_k - z}$,kde x' je x-ova súradnica bodu na priamke roviny s rovnakou z-ovou súradnicou ako má bod ***Ch***. Z tohto vzťahu môžeme vyjadriť x_d :



$$x_d = max\{x - x'\} = max\left\{ x - x_k + \boldsymbol{tg}\frac{\boldsymbol{\alpha}}{\boldsymbol{2}}(z_k - z) \right\},\ k = 1..8$$

Dostali sme hodnotu, o ktorú musíme posunúť pravú rovinu viditeľnej oblasti, aby sme z nej dostali pravú vonkajšiu rovinu. Podobne môžeme postupovať aj pri ostatných rovinách.

Výpočet týchto hodnôt môžeme zhrnúť do prehľadu vzťahov, ktoré platia pre jednotlivé roviny:

Vonkajšie:

Ľavá: $x_{lm} = max\left\{ x_k - x + \boldsymbol{tg}\frac{\boldsymbol{\alpha}}{\mathbf{2}}(z_k - z) \right\}$ Pravá: $x_{pm} = max\left\{ x - x_k + \boldsymbol{tg}\frac{\boldsymbol{\alpha}}{\mathbf{2}}(z_k - z) \right\}$

Dolná: $y_{dm} = max\left\{ y_k - y + \boldsymbol{tg}\frac{\boldsymbol{\beta}}{\mathbf{2}}(z_k - z) \right\}$ Horná: $y_{hm} = max\left\{ y - y_k + \boldsymbol{tg}\frac{\boldsymbol{\beta}}{\mathbf{2}}(z_k - z) \right\}$

Predná: $z_{rm} = max\{ z_k - z \}$ Zadná: $z_{zm} = max\{ z - z_k \}$

Vnútorné:

Ľavá: $x_{lv} = max\left\{ x - x_k - \boldsymbol{tg}\frac{\boldsymbol{\alpha}}{\mathbf{2}}(z_k - z) \right\}$ Pravá: $x_{pv} = max\left\{ x_k - x + \boldsymbol{tg}\frac{\boldsymbol{\alpha}}{\mathbf{2}}(z_k - z) \right\}$

Dolná: $y_{dv} = max\left\{ y - y_k - \boldsymbol{tg}\frac{\boldsymbol{\beta}}{\mathbf{2}}(z_k - z) \right\}$ Horná: $y_{hv} = max\left\{ y_k - y + \boldsymbol{tg}\frac{\boldsymbol{\beta}}{\mathbf{2}}(z_k - z) \right\}$

Predná: $z_{rv} = max\{ z - z_k \}$ Zadná: $z_{zv} = max\{ z_k - z \}$

Nakoniec stačí zistiť, či sa bod ***Ch*** nachádza vo vonkajšej a vnútornej oblasti. Môžeme použiť rovnaké zisťovanie polohy bodov ako pri prvom spôsobe zisťovania obsiahnutosti obálky vo viditeľnej oblasti, musíme však zistiť nové začiatky vnútornej a vonkajše viditeľnej oblasti, ktoré už teraz nie sú rovné **0**, a taktiež opraviť hodnoty z_{min} a z_{max}.

Pri zisťovaní vonkajšieho prieniku sa nám stredný bod posunie do bodu

$$S = \left(\frac{x_{lm} + x_{pm}}{2}, \frac{y_{dm} + y_{hm}}{2}, -\frac{x_{lm} + x_{pm}}{2}\boldsymbol{ctg}\frac{\boldsymbol{\alpha}}{\mathbf{2}} \right) \quad , \quad z_{min} = z_{rm} \quad \text{a} \quad z_{max} = z_{zm}$$

Pri vnútornom prieniku sa nám stredný bod posunie do bodu

$$S = \left(\frac{x_{lv} + x_{pv}}{2}, \frac{y_{dv} + y_{hv}}{2}, \frac{x_{lv} + x_{pv}}{2}\boldsymbol{ctg}\frac{\boldsymbol{\alpha}}{\mathbf{2}} \right) \quad , \quad z_{min} = z_{rv} \quad \text{a} \quad z_{max} = z_{zv}$$

Bod ***Ch*** a taktiež z_{min} a z_{max} posunieme o bod ***S*** (aby bol ***S*** totožný s bodom **0**) a otestujeme, či sa bod ***Ch*** nachádza vnútri oblasti. Môžu nastať tri prípady:

Ch *sa nenachádza vo vonkajšej oblasti*
- žiaden bod obálky nie je vo vnútri viditeľnej oblasti.

Ch *sa nachádza vo vonkajšej ale nie vo vnútornej oblasti*
- niektoré body obálky sa môžu nachádzať vnútri viditeľnej oblasti, musíme orezať mnohouholníky

Ch *sa nachádza vo vonkajšej aj vo vnútornej oblasti*
- všetky body obálky sa nachádzajú vnútri viditeľnej oblasti, nemusíme orezávať mnohouholníky.

Čísla s pevnou rádovou čiarkou

Na výslednú rýchlosť zobrazovania má veľký vplyv zvolená reprezentácia čísel. Počítač vo všeobecnosti pracuje s dvomi druhmi čísel - s celými číslami a reálnymi číslami s pohyblivou rádovou čiarkou. Na mnohých procesoroch platí, že operácie s celými číslami sú mnohokrát rýchlejšie ako s reálnymi číslami (aj keď sa to nedá zovšeobecňovať), preto je vhodné zamýšľať sa nad tým, či by nebolo možné pomocou celých čísel vyjadriť aspoň nejaké reálne čísla. Jednou z možností je ukladať si do celých čísel celú časť násobku reálneho čísla číslom, ktoré je väčšie ako 1. Ak bude tento násobok v tvare 2^i, potom takéto čísla budú mať na i-tom bite hranicu medzi celou a desatinnou časťou čísla, a preto sa nazývajú č*ísla s pevnou rádovou čiarkou*. Tieto čísla môžeme opísať tromi hodnotami a/b/c - prvá určuje počet bitov celého čísla, druhá počet bitov, ktorú zaberá celá časť čísla a tretia počet bitov, ktorú zaberá vo výslednom čísle desatinná časť pôvodného čísla. Často sa používajú čísla 16/8/8, 32/16/16 alebo 32/24/8. Rozdiel medzi poslednými dvomi spočíva v modifikáciách násobenia a delenia, aby odrážali inú polohu rádovej čiarky operandov.

Ako zapísať pomocou čísla v tvare 32/16/16 číslo 15,2787 ? 32/16/16 nám hovorí, že desatinná časť čísla zaberá 16 bitov, teda pôvodné číslo je nutné najskôr prenásobiť číslom 2^{16}, a potom vziať jeho celú časť. Dostávame číslo 1001304. Ak sa na toto číslo pozrieme v hexadecimálnej sústave, potom vidíme, že jeho tvar je 0F4758H. Keďže každá číslica v tejto sústave opisuje hodnotu 4-och bitov, 4 číslice opisujú hodnotu 16-tich bitov. Keď zoberieme posledné štyri číslice, tie nám dávajú "desatinnú" časť čísla (v skutočnosti ide o šestnástinnú časť). Preto môžeme pred posledné štyri číslice vložiť rádovú čiarku. Dostávame nakoniec číslo F,4758, čo je približným vyjadrením čísla 15,2787 (skutočné číslo, ktoré je týmto zápisom vyjadrené je 15,278686523438...).

Operácie s číslami s pevnou rádovou čiarkou

Ak chceme používať tento spôsob vyjadrenia čísel, mali by sme byť schopný vyjadriť elementárne operácie ako je sčítanie, odčítanie, násobenie a delenie.

Sčítanie a odčítanie

Ak sčítavame alebo odčítavame dve čísla s pevnou rádovou čiarkou, môžeme použiť obyčajné sčítanie alebo odčítanie týchto celých čísel, pretože pri týchto operáciách na úrovni bitov nie je rozdiel medzi týmito reprezentáciami čísel.

Násobenie

Na násobenie takisto použijeme násobenie celých čísel. Problém je však v tom, že násobíme dve čísla, ktoré sú reálne, tak, akoby boli celé. Ak máme násobiť dve čísla 32/16/16, potom pri násobení ich berieme akoby boli 32/32/0. Po násobení dostávame výsledok v tvare 64/64/0 (zdvojnásobí sa počet číslic) a my z neho potrebujeme spätne vytvoriť číslo 32/16/16.

Násobenie dvoch reálnych čísel môžeme napísať ako $a \cdot b = c$. Pomocou násobenia celých čísel však násobíme $a \cdot 2^{16} \cdot b \cdot 2^{16} = c \cdot 2^{32}$. Očakávali by sme však $c \cdot 2^{16}$. Preto výsledok násobenia celých čísel musíme podeliť číslom 2^{16}, aby sme dostali žiadaný výsledok. Toto delenie sa v assembleri vykoná veľmi jednoducho pomocou aritmetického posunu vpravo cez dvojicu registrov.

Pri násobení vzniká veľmi vysoké riziko pretečenia a z toho vyplývajúcej nepresnosti výpočtu.

Delenie

Delenie podobne ako násobenie presahuje rozsah čísel, avšak nie vo výsledku, ale v delenci. Ak delíme dve reálne čísla a : b = c, potom pri reprezentácii v číslach 32/16/16 by sme delili $a \cdot 2^{16} : b \cdot 2^{16} = c$. Takýto tvar výsledku však nepotrebujeme, pretože vyžadujeme výsledok v tvare $c \cdot 2^{16}$. Preto delenec vopred musíme dať do tvaru $c \cdot 2^{32}$, čím však presiahneme počet bitov, určených na číslo. Našťastie, podobné problémy majú aj operácie s celými číslami, preto sa v procesoroch na delenec zvyknú vyhradiť dva registre, z ktorých jeden určuje hornú a druhý dolnú časť delenca. Potom po úprave týchto registrov možno využiť delenie celých čísel.

Pri delení vzniká nebezpeèenstvo delenia veľmi malým číslom alebo nulou, čo má za následok vyvolanie prerušenia a znemožnenie dokončenia výpočtu.

Iné operácie

Medzi ďaľšie typy operácií, ktoré sa často využívajú patria napríklad trigonometrické operácie, ako je sínus alebo kosínus. Rozvoj do ortogonálnych radov a aproximácia pomocou nich je však pri týchto číslach iba veľmi ťažko riešiteľná. Preto sa využíva tabuľkovanie hodnôt s určitým krokom. Pre sínus stačí vložiť do tabuľky funknčné hodnoty zo štvrtiny periódy a zvyšok symetricky dopľňať.

Z ďaľších operácií je často využívaná druhá odmocnina. Tu stačí použiť iteratívny algoritmus, ktorý má veľmi presné výsledky aj pri tejto reprezentácii čísel.

Akumulácia chyby

Keďže reálne čísla popísané číslami s pevnou rádovou čiarkou majú obmedzenú presnosť, vzniká pri práci s nimi chyba. Táto chyba s výrazne zvyšuje pri operáciach násobenia, preto je vhodné, aby sa v programe využívali násobenia čo možno najmenej. Preto treba preferovať výpočty, ktoré majú vlastnosť zlučovania viacerých multiplikatívnych operácií do jednej.

Pri výpočtoch si je tiež potrebné dávať pozor na pretečenie, ktoré síce nespôsobí priame zrútenie programu, môže však byť zdrojom mnohých chýb, ktoré sa veľmi ťažko hľadajú a ich prejavy obtiažne zatrieďujú, avšak majú osudný význam pre funkčnosť programu.

Ako jedna z možností spresnenia výpočtov, minimalizovania chyby a vyhýbania sa pretečeniu sa javí aj spojenie operácie násobenia a delenia do jednej operácie. Najskôr dve čísla vynásobíme, akoby boli celé, ale výsledok neupravujeme. Tento výsledok môžeme brať ako pripravený celočíselný delenec a stačí ho vydeliť deliteľom, pričom dostávame správny výsledok. Ak by sme postupovali samostantne, najskôr vynásobili dve čísla, úpravili výsledok, ten potom upravili do tvaru delenca a vydelili, úpravami medzi násobením a delením by sme riskovali nezachytenie pretečenia pri násobení.

Literatúra

[Assarsson1999] Optimized View Frustum Culling Algorithms, *Ulf Assarsson, Tomas Möller*, Chalmers University of Technology, Department of Computer Engineering, Technical Report 99-3, 1999

[Eberly1999-1] Perspective Mappings in 2D and 3D, *Dave Eberly*, Magic Software 1999

[Eberly1999-2] Perspective Mappings in Quadrilateral, *Dave Eberly*, Magic Software 1999

[Eberly1999-3] Rotations, *Dave Eberly*, Magic Software 1999

[Eberly2000] Intersection of Orthogonal View Frustum and Oriented Bounding Box using Separation Axes Testing, *Dave Eberly*, Magic Software 2000

[Gottschalk1996] OBB Tree: A Hierarchical Structure for Rapid Interference Detection, *S. Gottschalk, M.C. Lin, D. Manocha*, In proceedings of the ACM Siggraph 1996

[Grešák1982] Algebra a geometria, *Pavol Grešák*, Alfa Bratislava 1982

[Katriňák1999] Algebra a teoretická aritmetika 1, *Tibor Katriňák, Martin Gavalec, Eva Gedeonová, Jaroslav Smítal* ,Vydavateľstvo Univerzity Komenského 1999

[Klimo1989] Teória Informácie a prenos dát, *Martin Klimo*, Alfa Bratislava 1989

[Kluvánek1971] Matematika I, *Igor Kluvánek, Ladislav Mišík, Marko Švec*, 4. vydanie, Alfa Bratislava 1971

[Kolb1995] A Realistic Camera Model for Computer Graphics, *Craig Kolb, Don Mitchell, Pat Hanrahan*, In proceedings of the ACM Siggraph 1995

[Omeľčenko1983] Osnovy spektrálnej teórie rozpoznávania signálov, *V. A. Omeľčenko*, Nakladateľstvo pri Charkovskej Ekonomickej Univerzite, Charkov 1983

[Oravec1976] Príručka slovenského pravopisu, *Ján Oravec, Vincent Laca*, Slovenské Pedagogické Nakladateľstvo 1976

[Pecinovský1996] Objektové programování, *Rudolf Pecinovský, Miroslav Virius*, Grada 1996

[Ružický1995] Počítačová grafika a spracovanie obrazu, *Eugen Ružický, Andrej Ferko*, Vydavateľstvo Sapientia 1995

[Virius1997] Pasti a propasti jazyka C++, *Miroslav Virius*, Grada 1997

[Zorin1995] Correction of Geometric Perceptual Distortions in Pictures, *Dennis K. Zorin*, In Partial Fulfilment of the Requirements for the Degree of Master of Science, California Institute of Technology, Pasadena 1995

[Žára1992] Počítačová grafika - principy a algoritmy, *Jiří Žára, Aleš Limpouch, Bedřich Beneš, Tomáš Werner*, Grada 1992

[Žára1998] Moderní počítačová grafika, *Jiří Žára, Bedřich Beneš, Petr Felkel*, Computer Press 1998