VYUŽITÍ DUÁLNÍHO PROSTORU PRO METODU SLEDOVÁNÍ PAPRSKU

I.Kolingerová
ZČU Plzeň, Americká 42, 306 14 Plzeň
e-mail : kolinger@kron.zcu.cs

1. Úvod

Sledování paprsku (Ray tracing) je jednou ze základních metod zvýšení věrnosti zobrazení v počítačové grafice. Jejímu širšímu využití brání především značná časová náročnost. Proto jsou již delší dobu v centru pozornosti techniky jejího urychlení.

Protože největší časové ztráty přináší výpočet průsečíků paprsku s objekty ve scéně, je snaha jednak zrychlit a zjednodušit tento výpoočet, jednak předem eliminovat ty objekty, se kterými paprsek nemá průsečík. Velmi oblíbené jsou techniky dělení prostoru za pomoci speciálních datových struktur, především stromů, nebo metody využívající koherence.

Nadějnou cestou se zdají být také metody využívající některých méně obvyklých matematických postupů. Tato řešení jsou většinou závislá na typu matematického popisu objektů ve scéně.

Pokud je scéna složena z konvexních polyhedronů, lze využít postup založený na principu duality. Výchozím podnětem pro jeho návrh byla publikace [1], poskytující příslušný matematický aparát.

2. Duální přístup k určení průsečíků polyhedronu s paprskem

Podstatou postupu je využití jednoduchosti detekce průsečíků dvou geometrických objektů ve srovnání s jeho lokalizací. Při vhodném matematickém vyjádření paprsku i objektů ve scéně je možná rychlá eliminace všech objektů, se kterými paprsek nemá průsečík, a následná lokalizace průsečíků se zbývajícími objekty.

Využívá se geometrické duální transformace mezi body a

nadrovinami v d-dimenzionálním euklidovském prostoru E^d. Tato transformace je izomorfismem.

Označme $p = (p_1,p_2,\ldots,p_d)$ bod v E^d

a $H(a,c) = \{ x \in E^d : x \cdot a = c \}$, $a \in E^d - \{ 0 \}$, $c \in E^1$ nadrovinu v E^d. Dále předpokládáme $a_d \neq 0$, tj. nadrovina není vertikální.

Duální obraz bodu p D(p) je nadrovina v E^d daná rovnicí :

$$x_d = -p_1x_1 - p_2x_2 - \cdots p_{d-1}x_{d-1} + p_d$$

Duální obraz nadroviny H D(H) je bod $(b_1,b_2,\ldots,b_d)$ v E^d, kde

$$b_d = c/a_d,$$
$$b_i = - a_i/a_d \ , \ i=1,\ldots,d-1$$

Každý konvexní polyhedron P je v duálním prostoru reprezentován dvěma funkcemi TOP^P a BOT^P: $E^{d-1} \rightarrow E^1$. P a TOP^P, BOT^P jsou izomorfní.

Pro uzavřený konvexní polyhedron jsou funkce TOP^P a BOT^P def. takto :

$$TOP^P(x_1,\ldots,x_{d-1}) = \max_{v \in V_p} F_{D(v)}(x_1,\ldots,x_{d-1})$$
$$BOT^P(x_1,\ldots,x_{d-1}) = \min_{v \in V_p} F_{D(v)}(x_1,\ldots,x_{d-1})$$

kde V_p je množina všech vrcholů polyhedronu P,

$F_{D(v)}: E^{d-1} \rightarrow E^1$ je fce, jejímž grafem je D(v).

Funkce TOP^P, BOT^P jsou po částech lineární, spojité, TOP^P je konvexní a BOT^P konkávní.

Definice lze rozšířit i pro neuzavřené polyhedrony a vertikální nadroviny. Pro využití pro účely sledování paprsku však postačí v tomto tvaru s výše uvedenými omezeními.

Detekce průniku polyhedronu s nadrovinou

Pro detekci průniku polyhedronu s nadrovinou využijeme následující tvrzení : Nadrovina H : D(H) = $[b_1,b_2,\ldots,b_d]$, která není vertikální, protíná uzavřený polyhedron P $\Leftrightarrow$

$$BOT^P(b_1,\ldots,b_{d-1}) \leq b_d \leq TOP^P(b_1,\ldots,b_{d-1}).$$

Teorii ilustrujeme na příkladě pro d=2, viz obr.1, 2 :
Je zadán polygon P množinou vrcholů $V_p = \{v_1, v_2, v_3\}$

$$v_1 = [1,1],\ v_2 = [3,1],\ v_3 = [2,2]$$

a přímka H (a,c), a = [-1,1], c = -1.

Duální obrazy vrcholů polygonu jsou

$$F_{D(v_1)} : x_2 = -x_1 + 1$$

$$F_{D(v_2)} : x_2 = -3x_1 + 1$$

$$F_{D(v_3)} : x_2 = -2x_1 + 2$$

Funkce TOP^P a BOT^P mají následující tvar :

$$TOP^P(x_1) = \max_{v \in V_p} F_{D(v)}(x_1) = \begin{cases} -3x_1 + 1 & x_1 < -1 \\ -2x_1 + 2 & -1 \le x_1 < 1 \\ -x_1 + 1 & x_1 \ge 1 \end{cases}$$

$$BOT^P(x_1) = \min_{v \in V_p} F_{D(v)}(x_1) = \begin{cases} -x_1 + 1 & x_1 < 0 \\ -3x_1 + 1 & x_1 \ge 0 \end{cases}$$

Duální obraz přímky H je D(H) = [1,-1].

Protože $b_1 = 1$, $b_2 = -1$, $BOT^P(b_1) = -2$, $TOP(b_1) = 0$, platí, že

$$BOT^P(b_1) \le b_2 \le TOP^P(b_1)$$

a polygon P má tedy s přímkou H průsečík.

Detekce průsečíku polyhedronu s přímkou

Výše uvedený postup není vhodný pro řešení průniku geometrických útvarů, jejichž dimenze se vzájemně liší o více než o 1. Pro detekci průsečíku polyhedronu s přímkou proto využijeme popisu přímky jako průsečnice 2 rovin a řešíme detekci průsečíku polyhedronu postupně s oběma rovinami. Jen ty polyhedrony, které mají průnik s oběma rovinami, postupují do dalšího zpracování.

Lokalizace průsečíku polyhedronu s přímkou

Eliminací polyhedronů, s nimiž není průsečík, možnosti zkoumání polyhedronu jako celku končí, a dále je nutné se zabývat

polygony tvořícími jeho stěny.

Využijeme následujícího tvrzení : Bod p leží nad (na,pod) nadrovinou H <=> duální obraz H D(H) leží pod (na,nad) duálním obrazem p D(p).

Polygon P' je protínán rovinou H <=> ∃2 vrcholy u,v tohoto polygonu, mezi nimiž rovina prochází <=> D(H) leží mezi D(u) a D(v) (<= z předcházejícího tvrzení).

Pokud polygon splňuje tuto podmínku pro obě roviny, víme, že jej obě tyto roviny protínají. Zjištění, zda průsečík obou průsečnic leží uvnitř tohoto polygonu, již nelze v duálním prostoru efektivně provést, a je výhodnější užít metod "klasické" geometrie. Polygon zařadíme do množiny kandidátů pro výpočet průsečíku roviny, v níž leží polygon, s přímkou a test, zda nalezený průsečík leží uvnitř polygonu. Při vytváření algoritmu byla pro tyto koncové výpočty užívána metoda pyramidy ([2]).

3. Algoritmus pro výpočet průsečíků polyhedronu s paprskem

Podle výše popsané metody byl navržen algoritmus a implementován v jazyce Pascal na PC 386 :

```
procedure INTERSECTION(xa,xb);

{ Algoritmus pro výpočet průsečíku přímky s konvexním polyhedronem
v E3 s převodem do duálního prostoru }
{ 11.10.1992 I.Kolingerová }
{ stěny orientovány tak, aby normály směřovaly ven z tělesa }
{ rovnice roviny [x|1]^T[ni|Di] = Aix + Biy + Ciz + Di = 0 }
{ přímka dána x(t) = xa + s*t }

begin
 s := xb - xa ;
 FIND_DUAL_SPACE_IMAGE;
 if TOP_AND_BOT_P(b11,b12,b13,1) then
   { existuje průsečnice - test opakovat s 2.rovinou }
   if TOP_AND_BOT_P(b21,b22,b23,2) then
      { existuje průsečnice s oběma rovinami - určit průsečík }
      begin
```

```
      for i := 1 to N do
          begin
           { kontrola i-tého polygonu s vrcholy iv1, iv2, iv3 }
           { test vůči 1.rovině }
           p1 := hull[1,iv1] - b13;
           p2 := hull[1,iv2] - b13;
           p3 := hull[1,iv3] - b13;
           sp1 := SGN(p1); sp2 := SGN(p2); sp3 := SGN(p3);
           if (sp1*sp2 <> 1) or (sp1*sp3 <> 1) or (sp2*sp3 <> 1)
              then
              { různá poloha vůči duálu roviny 1 => existuje
                průsečnice s 1.rovinou }
               begin
                { test vůči 2.rovině }
                p1 := hull[2,iv1] - b23;
                p2 := hull[2,iv2] - b23;
                p3 := hull[2,iv3] - b23;
                sp1 := SGN(p1); sp2 := SGN(p2); sp3 := SGN(p3);
                if (sp1*sp2 <> 1) or (sp1*sp3 <> 1) or (sp2*sp3
                    <> 1) then
                    { různá poloha vůči duálu roviny 2 =>
                      existuje průsečnice s 2.rovinou }
                     Zařaď index i do množiny kandidátů pro
                      výpočet průsečíku metodou pyramidy
               end
          end
      end
end; { INTERSECTION }

procedure FIND_DUAL_SPACE_IMAGE;

{ procedura pro přímku najde 2 roviny, které ji určují, najde
jejich duální obraz a uloží jej do bodů B1=(b11,b12,b13) a
B2=(b21,b22,b23) }

begin
 { 1.rovina : výpočet norm. vektoru nn kolmého ke směrovému
 vektoru s přímky }
 if abs(sz) > 0 then begin nnx := 1; nny := 0; nnz := -sx/sz end
```



```
    else
       if abs(sy) > 0 then
          begin nnx := 1; nny := -sx/sy; nnz  :=  0 end
       else
          begin nnx := -sz/sx; nny := 0; nnz := 1 end;
    { výpočet koeficientů rovnice roviny }
    c := s x nn;
    d := c * xa;
    { výpočet duálního obrazu roviny }
    b13:= d/cz; b12 := -cy/cz; b11 := -cx/cz;
    { 2.rovina : výpočet norm. vektoru nn kolmého ke směrovému
      vektoru s přímky }
    if abs(sz) > 0 then begin nnx := 0; nny := 1; nnz := -sy/sz end
    else
       if abs(sy) > 0 then
          begin nnx := 0; nny := -sz/sy; nnz  :=  1 end
       else
          begin nnx := -sy/sx; nny := 1; nnz := 0 end;
    { výpočet koeficientů rovnice roviny }
    c := s x nn;
    d := c * xa;
    { výpočet duálního obrazu roviny }
    b23 := d/cz; b22 := -cy/cz; b21 := -cx/cz
end; { FIND_DUAL_SPACE_IMAGE }

function TOP_AND_BOT_P(var b1,b2,b3:real;no:integer):boolean;

{ Hledají se hodnoty duálního obrazu polyhedronu jako b =
f(x1,x2) pro x1=b1, x2=b2 a ukládají se do pole hull na řádek č.
no podle indexu aktuální roviny (=1,2). Z těchto hodnot se hledá
hodnota fce TOP a BOT jako max, min této fce v daném bodě.
Leží-li b3 mezi TOP a BOT, fce vrací true }

begin
 top := -∞; bot := ∞;
 for j := 1 to number of vertices do
     begin
      { zpracování 1 úseku duální fce }
      b := -x[j]*b1 - y[j]*b2 + z[j];
```

```
      hull[no,j] := b;
      if bot > b then bot := b;
      if top < b then top := b
     end;
 if (bot <= b3) and (b3 <= top) then
    top and bot p := true
 else
    top and bot p := false
end;  { TOP_AND_BOT_P }
```

4. Možnosti dalšího urychlení navržené metody

Jestliže je výše uvedený algoritmus využit pro metodu sledování paprsku, lze jej pro primární paprsky dále urychlit vhodnou volbou rovin určujících paprsek, a to takovou, aby jedna z rovin byla společná pro celý obrazový řádek. Nemá-li polyhedron na aktuálním obrazovém řádku s touto rovinou průnik, pak tento polyhedron na daném řádku není třeba vůbec zkoumat. Podobně by bylo možné druhou rovinu zavést společnou pro celý sloupec obrazu. Při sériovém zpracování po řádcích by však tato volba znamenala pamatovat si v každém sloupci seznam polyhedronů, které mají se svislou rovinou průnik, a má je tedy smysl v aktuálním sloupci uvažovat - např. formou bitového vektoru. Protože byl algoritmus implementován v paměťové omezeném prostředí Turbo Pascalu, nebyla vzhledem k dodatečným paměťovým nárokům tato metoda využita.

Další možností urychlení pro primární paprsky je setřídění těles podle vzdálenosti od pozorovatele. Ani tato metoda nebyla implementována.

Rezervy jsou také v dosud užitém způsobu nalezení duálního obrazu (FIND_DUAL_IMAGE) a detekci průsečíků pomocí TOP_AND_BOT_P. V prvním případě jsou sice normálové vektory voleny s akcentem na lineární nezávislost a snadný výpočet, ale nulové složky nejsou při výpočtu nijak využity. Výpočet koeficientů rovnice roviny lze zapsat tak, aby počet operací byl co nejmenší, tj. místo

```
cx := sy*nnz - sz*nny;
cy := sz*nnx - sx*nnz;
cz := sx*nny - sy*nnx;
```


např. pro nnx = 0, nny = 1 zapsat :

```
cx := sy*nnz - sz;
cy := -sx*nnz;
cz := sx;
```

Ušetříme tedy 8 násobení a 4 sčítání/odčítání pro každý paprsek.

Funkce TOP^P, BOT^P jsou v algoritmu v místě potřeby přímo počítány. Pokud by duální obrazy byly předem připraveny a uloženy ve vhodných datových strukturách, dal by se počet kroků pro nalezení hodnot funkcí TOP_AND_BOT_P snížit. Využití speciálních datových struktur bude předmětem dalšího výzkumu.

5. Porovnání navrhované metody s jinými možnými

Metoda byla implementována v jazyce Pascal jako součást programu pro výpočet osvětlení scény pomocí sledování paprsku.

Metoda byla prověřována ve dvou variantách - bez využití společné roviny na řádku nebo ve sloupci obrazu a s využitím jedné společné roviny procházející aktuálním řádkem. V níže uvedených tabulkách jsou obě modifikace odlišeny indexem - DUAL 2, resp.3. Stejný výpočet proběhl pro referenční metodu PYRAMIDA a metodu CONVEX. První z obou referenčních metod je založena na skutečnosti, že průsečík paprsku s polygonem, pokud existuje, musí ležet uvnitř pyramidy určené počátkem paprsku a vrcholy zkoumaného polygonu. Protože se tato metoda ukázala jako extrémně pomalá, byly testy doplněny ještě o metodu s pracovním názvem CONVEX, která vychází z Cyrus-Beckova algoritmu ořezávání. Byl také zkoumán vliv obalových koulí a testů pomocnou rovinou na urychlení výpočtu.

Program v těchto konfiguracích byl testován na počítačích řady 386 SX a 386. Z důvodů značné časové náročnosti především metody pyramidy byl výpočet omezen na primární sledování paprsku. Bylo počítáno 24 scén obsahujících vždy jeden polyhedron o různém počtu polygonů a různém procentuelním obsazení scény (jako procentuelní obsazení byl chápán poměr počtu pixelů s jinou barvou než má pozadí vůči počtu pixelů s barvou pozadí). Jsme si vědomi, že k důkladnému prověření metody je zapotřebí většího počtu náročnějších scén, avšak již užité scény s maximálním procentuelním obsazením zhruba 10 % a s maximálně 512 polygony se ukázaly na hranicích možností

užívané techniky - výpočet jedné scény pomalejší metodou někdy přesahoval 24 hodin. Další testy budou prováděny na adekvátnější technice, bude-li tato možnost.

Scény byly rozděleny do 4 skupin podle procentuelního obsazení : 1-2%, 4-5%, 6-7% a 9-10%. Každá skupina obsahovala 6 scén lišících se počtem polygonů, a to 24, 48, 72, 108, 384 a 512 polygonů. Naměřené doby výpočtu v s (po přepočtu na typ IBM 386, bez zahrnutí času potřebného pro ukládání vypočteného obrazu na disk a bez času na určení barevné palety) jsou uvedeny v následujících tabulkách :

Obsazení scény 1-2%

počet polygonů / metoda	24	48	72	108	384	512
PYRAMIDA	7533	14929	22370	33384	117615	155957
PYRAMIDA + OBAL	549	1311	1542	2005	6993	10024
PYRAMIDA + ROVINA	1627	3009	4199	5615	20986	26792
CONVEX	1907	4010	5911	8148	30265	41708
DUAL	1628	2437	3179	4312	12912	16846
DUAL + OBAL	664	828	880	1027	2284	3032
DUAL3	487	643	792	1031	3225	4171
DUAL 3 + OBAL	272	323	366	478	1318	1746

Obsazení scény 4-5%

počet polygonů / metoda	24	48	72	108	384	512
PYRAMIDA	7825	15459	23160	33937	119543	158021
PYRAMIDA + OBAL	1143	3733	5356	6344	19576	23751
PYRAMIDA + ROVINA	2686	5362	8371	10107	36014	41770
CONVEX	2026	4199	6206	8147	30264	40905
DUAL	2210	3327	4569	5840	16921	21083
DUAL + OBAL	1036	1652	2111	2423	5710	6669
DUAL3	1010	1459	2113	2449	7045	7973
DUAL 3 + OBAL	616	1031	1187	1403	3582	3998

Obsazení scény 6-7%

počet polygonů / metoda	24	48	72	108	384	512
PYRAMIDA	8122	15441	23545	34255	120900	159420
PYRAMIDA + OBAL	1781	4507	7160	9893	27087	35884
PYRAMIDA + ROVINA	3557	6082	9699	12529	45500	51402
CONVEX	2149	3800	6334	8147	30263	40905
DUAL	2746	3846	5117	6836	18927	24197
DUAL + OBAL	1442	2191	2693	3561	7759	9897
DUAL3	1495	2080	2632	3401	9011	10907
DUAL 3 + OBAL	968	1414	1532	2083	4853	5883

Obsazení scény 9-10%

počet polygonů / metoda	24	48	72	108	384	512
PYRAMIDA	8458	16336	24248	34696	121624	161395
PYRAMIDA + OBAL	2454	6889	9548	13485	41082	54460
PYRAMIDA + ROVINA	4327	8112	11883	15020	52335	63143
CONVEX	2275	4506	6569	8146	30262	40904
DUAL	3238	4567	6058	7938	22161	28208
DUAL + OBAL	1875	2838	3551	4840	11710	15018
DUAL3	2035	2652	3558	4514	12513	15070
DUAL 3 + OBAL	1324	1674	2154	2917	7220	8692

Při porovnání získaných časů je zřejmé, že nejhůře dopadla metoda PYRAMIDA. Její výhodou je pouze to, že prakticky nezávisí na procentuelním obsazení scény. Lepším etalonem se ukázala být metoda CONVEX. Ani tento způsob řešení nezávisí na procentuelním obsazení scény, pouze na počtu polygonů, avšak dosahované doby výpočtu jsou podstatně lepší než při užití metody PYRAMIDA.

Metoda duálního prostoru vychází pro většinu sledovaných

scén nejrychlejší, především ve variantě s jednou rovinou na řádku. Nevýhodou je však podstatně větší závislost na obsazení scény. Tato závislost indikuje, že při využití metod duálního prostoru pro scénu s větším procentuelním obsazením bude pravděpodobně výhodnější metoda vycházející z "klasické" geometrie, tedy např. CONVEX. Podle dosud provedených porovnání i teoretických úvah se zdá, že práh použitelnosti metod založených na duální reprezentaci by se mohl pohybovat kolem 25% obsazení. Jedná se však o odhad na základě pravděpodobnosti zasažení polyhedronu paprskem a na základě chování metody pro dosud sledované jednoduché scény a tento odhad je zapotřebí podložit přesnějšími výpočty a dalšími testy.

Všechny srovnávané metody se dále zrychlily nasazením testu obalové koule. Z časových důvodů nebyly ještě změřeny doby výpočtu algoritmu CONVEX s obalovými tělesy, proto se ve výše uvedených tabulkách tato kombinace nevyskytuje. Ze srovnání časů získaných výpočtem metodou pyramidy s testem pomocnou rovinou a s testem obalovou koulí vychází lépe obalová koule, kde je zapotřebí extrémně malý počet operací. Metoda pomocné roviny však nachází výrazné uplatnění v druhé variantě duálního prostoru, která se při kombinaci s obalovou koulí ukazuje jako nejlepší.

Poměry dob běhu jednotlivých metod v závislosti na počtu polygonů a obsazení scény jsou následující :

n polygonů	24	48	72	108	384	512
PYR.+OBAL / DUAL3+OBAL	1.892	3.746	4.458	4.522	5.511	6.012

obsazení scény	1-2%	4-5%	6-7%	9-10%
PYR.+OBAL / DUAL3+OBAL	4.255	4.319	4.355	4.497

n polygonů	24	48	72	108	384	512
PYRAMIDA / DUAL2	3.434	4.591	5.177	5.734	7.012	7.266
PYRAMIDA / DUAL3	8.201	11.849	13.742	15.999	19.144	20.634
CONVEX / DUAL2	0:893	1.221	1.385	1.376	1.774	1.889
CONVEX / DUAL3	2.119	3.160	3.663	3.857	4.864	5.399

obsazení scény	1-2%	4-5%	6-7%	9-10%
PYRAMIDA / DUAL2	7.316	5.605	4.927	4.295
PYRAMIDA / DUAL3	28.862	13.325	9.985	7.541
CONVEX / DUAL2	1.898	1.443	1.248	1.103
CONVEX / DUAL3	7.484	3.429	2.529	1.933

n polygonů	24	48	72	108	384	512
PYRAMIDA / PYRAM.+OBAL	7.144	5.331	6.165	7.009	7.587	7.404
DUAL2 / DUAL2+OBAL	2.054	2.080	2.346	2.542	3.237	3.260
DUAL3 / DUAL3+OBAL	1.628	1.615	1.828	1.771	2.001	1.993

obsazení scény	1-2%	4-5%	6-7%	9-10%
PYRAMIDA / PYRAM.+OBAL	14.774	5.570	3.941	2.809
DUAL2 / DUAL2+OBAL	4.069	2.474	2.061	1.742
DUAL3 / DUAL3+OBAL	2.156	1.757	1.679	1.631

Podle dosud provedených srovnání se tedy metody využívající duálního prostoru zdají být perspektivní, avšak pravděpodobně nebudou výhodné pro "příliš zaplněné" scény. Pro dosud používané scény byly metody jednoznačně lepší než ostatní užité algoritmy. Otázka, zda budou lepší i pro složitější scény, zůstává ještě otevřená a bude definitivně zodpovězena podle výsledku dalších testů.

6. Další směry výzkumu

Pro další zvýšení účinnosti a rozšíření pole působnosti algoritmu je vhodné se zaměřit především na následující problémy : využití vhodných datových struktur pro uložení hodnot funkcí TOP^P a BOT^P, užití testů jednou rovinou pro jeden řádek nebo sloupec při primárním sledování paprsku, důslednější využití nulových složek normálových vektorů rovin pro zmenšení počtu operací, výběr a vyzkoušení vhodného algoritmu pro rozdělení nekonvexního polyhedronu na konvexní části, využití duálního přístupu pro tělesa tvořená parametricky popsanými plochami, kde by roli polyhedronu mohl hrát konvexní obal tělesa.

7. Závěr

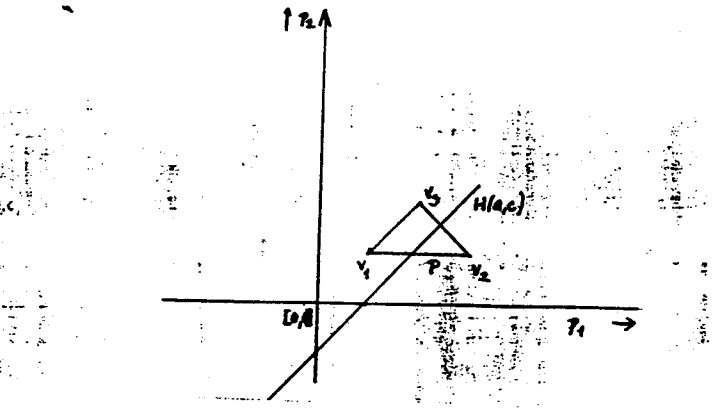
Výše uvedenou metodu lze užít pro zrychlení výpočtu průsečíků scény složené z konvexních polyhedronů s přímkou, např. pro účely sledování paprsku v počítačové grafice. Užití pro složitější typy objektů bude předmětem dalšího výzkumu.

8. Literatura

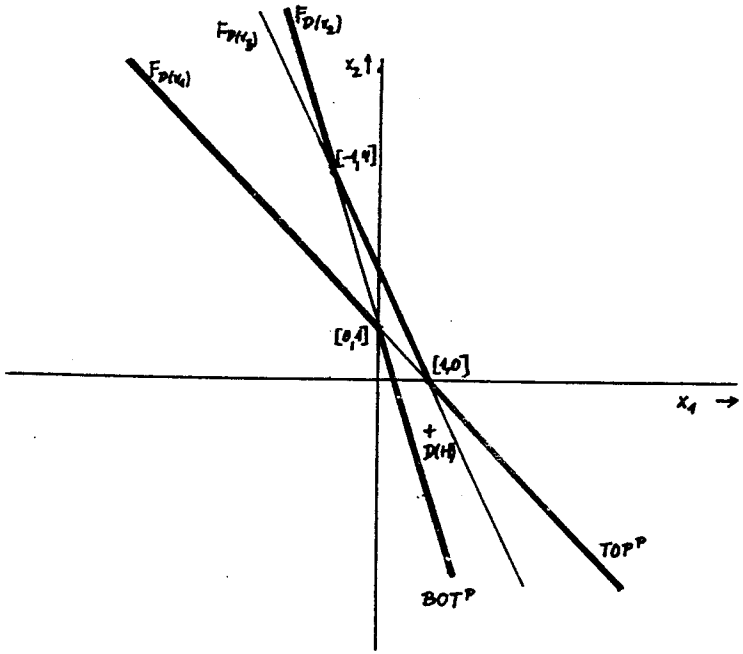
[1] Günther O. : Efficient Data Structures for Geometric Data Management, Springer-Verlag, Berlin Heidelberg 1988.
[2] Skala V. : Algoritmy ořezávání v E_2 a E_3 a jejich srovnání, výzkumná zpráva ke grantu č.151, KIV ZČU, 1992.
[3] Kolingerová I. : Možnosti urychlení metody sledování paprsku (nabídnuto k publikaci pro Sborník přednášek z XII. sympozia o algoritmech Algoritmy'93)

[4] Kolingerová I. : Programová realizace urychlení metody sledování paprsku (nabídnuto k publikaci pro Knihovnu algoritmů z XII. sympozia o algoritmech Algoritmy'93)

Obr.1 : Zadaný polygon P a přímka H



Obr.2 : Polygon P a přímka H po převodu do duálního prostoru



PREHĽAD VÝPOČTOVEJ GEOMETRIE

Roman Galbavý, Andrej Ferko, KAM MFF UK, 842 15 Bratislava

Kľúčové slová: výpočtová geometria, efektívne algoritmy, algoritmické paradigmy

Abstrakt.

Výpočtová geometria (computational geometry) završuje prvé desaťročie svojho prudkého rozvoja. V jej štruktúre sa stabilizovalo päť typov problémov: vyhľadávanie, konvexita, prieniky, proximita (Voronoiov diagram; zovšeobecnenia a aplikácie) a problémy na špeciálnej triede objektov (napr. geometria obdĺžnikov).

Konštrukcia efektívnych algoritmov na riešenie uvedených typov problémov sa líši jednak v algoritmických paradigmách resp. technikách, jednak podľa toho, či je vstup kompletný alebo dostupný postupne (on/line problem). Efektívnosť algoritmov sa hodnotí v štandardnom výpočtovom modeli.

1. Úvod

Existuje viacero možných predstáv o tom, čo je (alebo by mala byť) výpočtová geometria, od numericky orientovanej teórie splajnov, kriviek a povrchov, cez implementačné techniky tvorby softwaru v oblasti počítačovej grafiky, až po oblasť automatického dokazovania geometrických tvrdení. My budeme pod pojmom výpočtová geometria rozumieť disciplínu zaoberajúcu sa analýzou a návrhom efektívnych algoritmov na určovanie vlastností a vzťahov geometrických objektov (bod, priamka, mnohouholník, ...).

Cieľom tejto disciplíny je (v jej praktickej podobe) robiť veci premyslene, uvážene a v konečnom dôsledku efektívne (rýchlo a/alebo s malými pamäťovými nárokmi) a nie bezhlavo programovať prvý (ťažkopádny, pomalý a pamäťovo náročný) algoritmus, ktorý nás po zhliadnutí problému napadne.

Prostriedkami výpočtovej geometrie sú jednak (v teoretickej oblasti) aparát algebraickej, topologickej alebo kombinatorickej geometrie potrebný k tvorbe nástrojov na analýzu problémov, a jednak (v praktickej oblasti) techniky tvorby efektívnych