Konference
Algoritmy 85, JSMaF při ~~ČS~~ SAV - OSVS Bratislava
15.- 19.4.1985, Štrbské Pleso

RESPEKTOVÁNÍ VIDITELNOSTI - ZÁKLADNÍ GRAFICKÁ INSTRUKCE

Václav Skala
Katedra technické kybernetiky, VŠSE , Plzeň

1.Úvod

Řešení mnoha technických problémů má jako výsledek funkce dvou proměnných, které jsou buď vyjádřeny explicitním popisem nebo tabulkou funkčních hodnot. Funkce byly obvykle kresleny bez respektování viditelnosti. Podprogramy pro kreslení funkcí dvou proměnných s respektováním viditelností nebyly právě jednoduché (viz [6] - [8]), ačkoliv viditelnost může být řešena poměrně jednoduše na fyzické úrovni kreslení, pokud předpokládáme rastrové grafické výstupní zařízení. Bresenhamův algoritmus pro kreslení přímek může být jednoduše modifikován tak, abychom mohli kreslit funkce dvou proměnných s ohledem na viditelnost.

Williamson [7] řešil daný problém s tím,že neumožňuje obecnou rotaci, Watkinson [6] poukázal pouze, že pro některé kombinace natočení nebude jím předložený algoritmus správně eliminovat neviditelné části a Boutlandova metoda [1] je založena na pevném natočení. V předkládané metodě je podstatné pořadí kreslení a v [3], [4] je uvedena obecná metoda nalezení správného pořadí kreslení pro obecné natočení.

2.Úvod do problematiky

Mějme explicitně vyjádřenou funkci dvou proměnných x a y

$$z = f(x, y)$$

kde: $x \in \langle ax,bx \rangle \qquad y \in \langle ay,by \rangle$

a chtějme zobrazit tuto funkci použitím grafického rastrového displaye. Pro mnoho technických problémů je postačující ukázat chování této funkce kreslením funkčních řezů vzhledem k ose x a ose y, tj. křivky:

$$z = f(x, y_i) \qquad i=1,\dots,n$$

kde: $x \in \langle ax,bx \rangle \qquad a \; ay=y_1 < y_2 < \dots < y_n = by$

a křivky:

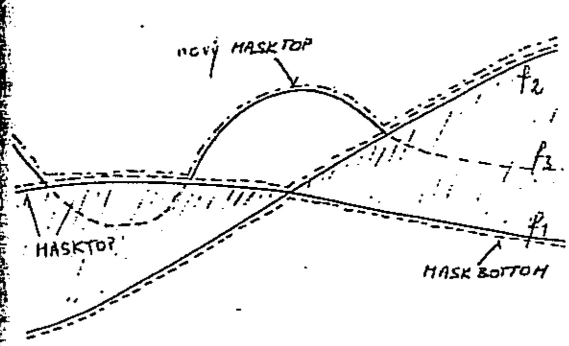
$$z = f(x, y) \qquad j=1,\dots,m$$

kde: $y \in \langle ay,by \rangle \qquad a \; ax=x_1 < x < \dots < x_m = bx$

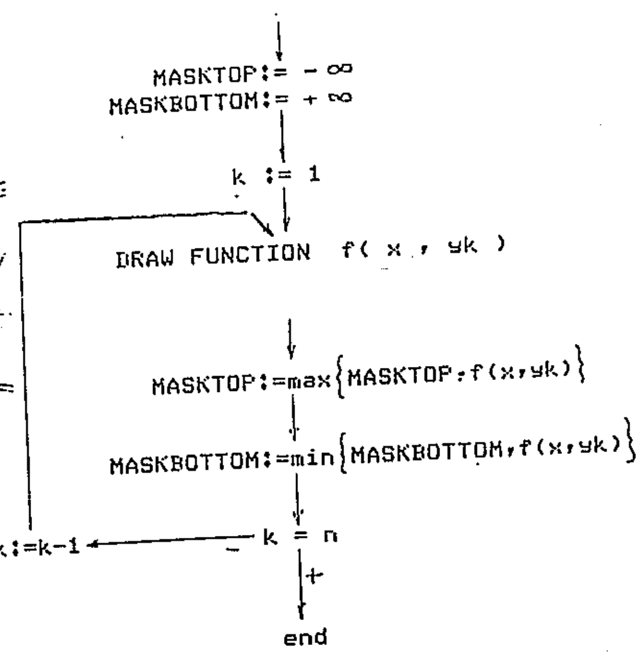
Daná funkce může být reprezentována buď specifikací funkce nebo tabulkou hodnot pro uzlové body sítě v rovině x-y. Je-li funkce složitá, je obtížné si představit chování funkce,neboť jsou kresleny i části, které jsou neviditelné při daném natočení. Úspěšně tento problém řešili Watkins[6],Williamson [7] a Boutland [1]. Princip řešení je obecně velmi jednoduchý.Jestliže nakreslíme první dva řezy rovnoběžné s osou x, pak se nám mezi těmito křivkami vytvoří pás neviditelnosti.Předpokládejme, že budeme nejdříve kreslit křivky bližší, později pak vzdálenější. Jestliže budeme chtít kreslit třetí křivku, je zřejmé, že části které prochází tímto pásem neviditelnosti jsou tímto pásem zakryté a tedy neviditelné (viz. obr.1). Jestliže budeme analyzovat podrobně daný problém, vidíme, že budeme potřebovat nějakým způsobem reprezentovat horní a dolní hranici pásu neviditelnosti, což může být uděláno funkcemi MASKTOP a MASKBOTTOM. Skutečnou reprezentaci těchto funkcí zatím opomineme. Nyní se problém kreslení křivek s ohledem na viditelnost stává jednoduchým (viz. algoritmus 1.),neboť budeme kreslit jen ty části řezů funkce,kdy body křivky jsou mimo pás neviditelnosti.

Problém jak reprezentovat funkce MASKTOP a MASKBOTTOM byl vyřešen Watkinsonem [6] zavedením maskovacích vektorů. V tomto případě musí být řešeny následující problémy:

- podle čeho rozhodneme, zda nastavíme i-tý nebo i+1 -ní prvky maskovacích vektorů, je-li $x_i < x < x_{i+1}$
- vektory MASKTOP a MASKBOTTOM by měly být nastaveny pro všechny body křivky. To znamená, že musíme použít nějaké interpolace s "vhodným" krokem interpolace.
- musíme řešit speciální případ, kdy řez funkce je paralelní s osou z.



obr.

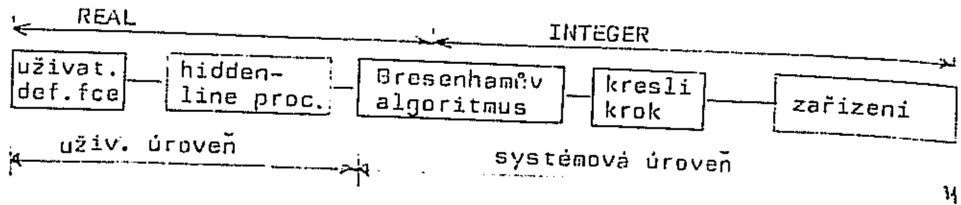


algoritmus 1.

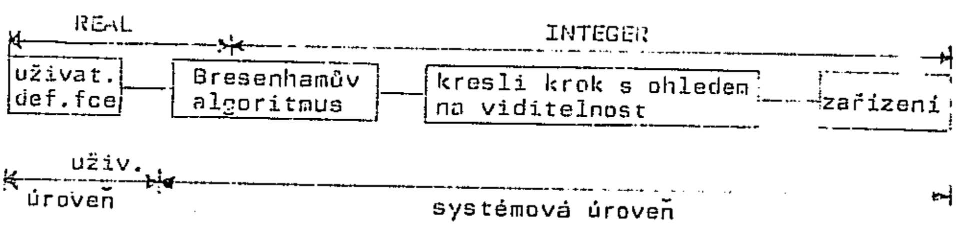
3.Navržená metoda

V [6] jsou funkce MASKTOP a MASKBOTTOM reprezentovány pomocí vektorů v pohyblivé řádové čárce. Proces kreslení lze pak znázornit obr.2. Nyní se můžeme ptát, zda existuje nějaká možnost zvýšení efektivnosti řešení daného problému. Jenou z možností je kombinace Watkinsonovy metody s Bresenhamovým algoritmem (viz. [2]) pro kreslení úsečky v rastrovém prostředí přímo na fyzické úrovni kreslení. Vzhledem k tomu, že pracujeme s rastrovým zařízením na úrovni fyzického kroku zbavíme se všech výše uvedených problémů.

Řešení viditelnosti je nyní velmi jednoduché, neboť musíme pouze modifikovat metodu proceduru DRAW STEP, která generuje kód pro fyzický pohyb kurzoru. Procedura DRAW STEP kreslí pouze jeden fyzický krok a tudíž musíme pouze kontrolovat, zda koncový bod kroku je uvnitř nebo vně pásu neviditelnosti. Předkládaná metoda je znázorněna na obr.3.



obr. 2.



obr. 3.

Je zřejmé, že nyní potřebujeme pouze celočíselnou reprezentaci pro maskovací vektory MASKTOP a MASKBOTTOM. Zjednodušené řešení je znázorněno algoritmem2.

```
{GLOBÁLNÍ PROMĚNNÉ}
VAR xo,yo: REAL; {Absolutní souřadnice}
    masktop,maskbottom: ARRAY [0..1023] OF INTEGER;
    PROCEDURE DRAWSTEP (dx,dy,i: INTEGER );
    VAR flag: BOOLEAN;
    BEGIN xo:=xo+dx;
          yo:=yo+dy;
          flag:=i=1; {posuv nebo čára ?}
          IF flag THEN
          BEGIN flag:=FALSE;
                IF masktop[xo]<=yo THEN
                BEGIN masktop[xo]:=yo; flag:=TRUE; END;
                IF maskbottom[xo]>=yo THEN
                BEGIN maskbottom[xo]:=yo; flag:=TRUE; END;
          END;
          IF flag THEN physline(dx,dy)
                 ELSE physmove(dx,dy);
    END;
```

Algoritmus 2.

4. Závěr

Předložený algoritmus pro kreslení funkcí dvou proměnných s respektováním viditelnosti je určen pro implementaci v periferních zařízeních rastrového typu, neboť algoritmus lze realizovat asi 20 strojovými instrukcemi bez použití pohyblivé řádové čárky. Nyní lze rozšířit soubor instrukcí grafických periférií rastrového typu o instrukce:

- inicializuj maskovací vektory
- kresli čáry s ohledem na viditelnost

V [3],[4] lze nalézt úplné řešení i s algoritmem pro obecné natočení při dodržení správného pořadí kreslení a modifikace základního algoritmu,která eliminuje chyby vzniklé skládáním dvou obrazů jež vznikly kreslením řezů paralelních s osou x a y. Podstatnou výhodou uvedeného přístupu je, že se uživatel nemusí zabývat volbou vhodného kroku apod. Daný algoritmus byl implementován na mikropočítačích Crommenco Z2H a Apple II, přičemž byl postatně rychlejší než Watkinsonův algoritmus a Williamsův algoritmus. Pokud daná grafická periférie bude vybavena mikroprocesorem a pamětí, není problémem soubor instrukcí periférie rozšířit o instrukci:

- Kresli funkci dvou proměnných s ohledem na viditelnost při daném natočení pro úhly α a β (v tomto případě by asi fukce byla dána tabulkou hodnot funkce v uzlových bodech sítě).

5. Literatura

[1] Boutland J.: Surface Drawing Made Simple, Computer Aided Design 11(1) January 1979,pp.19-22

[2] Bresenham J.E.: Algorithm for Computer Control of Digital Plotter, IBM Syst. J. 4(1), 1965, pp.25-30

[3] Skala V.: Hidden - line Processor, CSTR/29, Computer Sci. Dept. Brunel University, Middlesex, 1984

[4] Skala V.: Hidden - line Processor, CSTR 209- -84, Katedra Technické Kybernetiky, VŠSE , Plzeň, 1984

[5] Sowerbutts W.T.: A Surface-Plotting Program Suitable for Microcomputers, Computer Aided Design 15(6), November 1983, pp.324-327

[6] Watkins S.L.: Masked Three-Dimensional Plot Program with Rotation, CACM 17(9), September 1974, pp.520-523

[7] Williamson H.: Hidden-Line Plotting Program, CACM 15(2), February 1972, pp.100-103

[8] Wright W.A.: A Two-Space Solution for the Explicit Function of Two Variables, IEEE Trans. on Comp.22(1), January 1973,pp.28-33