Érdekes informatika feladatok

XXXII. rész

A Cohen–Sutherland-féle szakaszvágás

A generatív számítógépes grafika segítségével előállított grafikus modell a teljes színteret magába foglalja, a számítógép képernyőjén azonban lehet, hogy ennek csak egy része fog látszani, egy része pedig kilóg a képből.

POV-Ray-ben előállított színtér bevágása az ablakba

A modellből mindig csak annyit szabad mutatni, amennyi látszik. Sem a takart, sem a képből kilógó részek nem szabad felkerüljenek a generált képre.

A következőkben Kuba Attila: *Számítógépes grafika* (http://www.inf.u-szeged.hu/oktatas/jegyzetek/KubaAttila/grafika_html/szgrafika/) alapján összefoglaljuk a vágás módszereit:

- a modellt vágjuk le a megjelenítés előtt, azaz számítsuk ki a metszéspontokat és az új végpontokkal rajzoljunk;
- *ollózás*: pásztázzuk a teljes modellt, de csak a látható képpontokat jelenítjük meg (ellenőrizzük minden (x, y)-ra);
- a teljes modell legenerálása egy munkaterületre, majd innen való átmásolása a megfelelő látható résznek.

A pontok vágása egyszerű, egy $P(x, y)$ pont akkor látható, ha $x_{\min} \le x \le x_{\max}$, és $y_{\min} \le y \le y_{\max}$, ahol $(x_{\min}, y_{\min}), (x_{\max}, y_{\max})$ az ablakot (képernyőt) körbefogó téglalap bal-felső és jobb-alsó sarkának a koordinátái.

Vonalak, szakaszok esetén elegendő a végpontokat vizsgálni. Ha mindkét végpont belül van, akkor a teljes vonal belül van, nem kell vágni. Ha csak egy végpont van belül, akkor ki kell számítani a metszéspontot a téglalappal és vágni kell. Ha mindkét végpont kívül van, akkor lehet, hogy nincs közös része a vágási téglalappal, további vizsgálat szükséges. A vágási téglalap minden élére megvizsgáljuk, hogy van-e az élnek közös ré-

sze az egyenessel (egyenesek metszéspontjának a meghatározása, élen belül van-e a metszéspont?).

Vonalak vágására Cohen és Sutherland adott optimális vágási algoritmust bitkódok segítségével (SHOAFF, William D.: *Cohen-Sutherland Algorithm Source Code*, http://www.cs.fit.edu/~wds/classes/cse5255/thesis/lineClip/ code.html).

A Cohen–Sutherland-féle szakaszvágáshoz előzetes vizsgálatokra és kódolásra van szükség.

Legyen a szakasz két végpontja (x_1, y_1) és (x_2, y_2). Mindkét végpont a következő táblázat szerint kódot kap ($kód_1$, $kód_2$), annak megfelelően, hogy melyik tartományban van:

1001	1000	1010
0001	0000	0010
0101	0100	0110

- Ha a végpontok belül vannak, akkor nincs mit vágni, mindkét végpont kódja: $kód_1 = kód_2 = 0000$ (*triviális elfogadás*).
- Ha mindkét végpont a vágási téglalapon kívül van, ugyanazon az oldalon, vagyis ha $x_1, x_2 < x_{min}$ (***1), vagy ha $x_1, x_2 > x_{max}$ (**1*), vagy ha $y_1, y_2 < y_{min}$ (*1**), vagy ha $y_1, y_2 > y_{max}$ (1***), akkor a szakasz nem látható, bittenként $kód_1$ ÉS $kód_2 = 1$ (*triviális elvetés*).
- Különben a szakasz metszi a vágási téglalap valamelyik oldalát, tehát egy részét ki kell rajzolni (el kell fogadni), egy részét pedig nem (el kell vetni). Ekkor vegyünk egy külső végpontot (legalább az egyik az, ha mindkettő az, válasszuk az elsőt felülről lefelé és jobbról balra haladva), számítsuk ki a metszéspontot, a két részre bomlott szakasz egyik fele pedig az előbbi pont alapján triviálisan elvethető.

Körök, ellipszisek vágására fogjuk körbe az alakzatot egy kerettel (írjuk be a kört egy négyzetbe, az ellipszist egy téglalapba). Ha a keret belül van, akkor a kör vagy ellipszis is belül van, nem kell vágni. Ha a keret kívül van, akkor az alakzat is kívül van, nincs mit vágni. Különben: körnegyedekre (nyolcadokra) megismételjük a kör és él metszéspont meghatározó eljárást, majd pásztázunk.

Sokszögek vágásánál sok esetet kell megvizsgálni. A Sutherland-Hodgman algoritmus szerint sorba vesszük az éleket és egyenként vágunk.

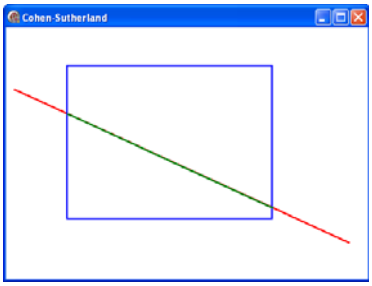
A Cohen–Sutherland-féle szakaszvágás gyakorlati megvalósítását a következő *Borland Delphi* program tartalmazza – Vardy, Andrew: *Point and line clipping* (http://www.cs.mun.ca/~av/courses/cg/notes/raster_clip.pdf) alapján:

```
unit uMain;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes,
  Graphics, Controls, Forms, Dialogs;

type
  TfrmMain = class(TForm)
```

```
    procedure FormPaint(Sender: TObject);
  end;

  TEdge = (teLEFT, teRIGHT, teBOTTOM, teTOP);
  TOutcode = set of TEdge;

var
  frmMain: TfrmMain;

implementation
{$R *.dfm}

procedure TfrmMain.FormPaint(Sender: TObject);
var
  x1, y1, x2, y2, xmin, xmax, ymin, ymax: real;

    procedure CompOutCode(x, y: real;
                          var code: TOutcode);
    begin
      code := [];
      if y > ymax then include(code, teTOP)
      else if y < ymin then
        include(code, teBOTTOM);
      if x > xmax then include(code, teRIGHT)
      else if x < xmin then include(code, teLEFT);
    end;

var
  accept, done: boolean;
  outcode1, outcode2, outcodeOut: TOutcode;
  x, y: real;
begin
   x1 := 10;
   y1 := 80;
   x2 := 450;
   y2 := 280;
   xmin := 80;
   ymin := 50;
   xmax := 350;
   ymax := 250;
   with Canvas do
     begin
        Pen.Width := 2;
        Pen.Color := clRed;
        MoveTo(Round(x1), Round(y1));
        LineTo(Round(x2), Round(y2));
        Pen.Color := clBlue;
        Brush.Style := bsClear;
        Rectangle(Round(xmin), Round(ymin),
                  Round(xmax), Round(ymax));
     end;
   accept := false;
   done := false;
   CompOutCode(x1, y1, outcode1);
```

```
CompOutCode(x2, y2, outcode2);
repeat
  if (outcode1=[]) and (outcode2=[]) then
    begin
      { triviális elfogadás }
      accept := true;
      done := true;
    end
  else if (outcode1*outcode2) <> [] then
    { triviális elvetés }
    done := true
  else
    begin
      if outcode1 <> [] then
        outcodeOut := outcode1
      else outcodeOut := outcode2;
      { metszéspont meghatározása }
      if teTOP in outcodeOut then
        begin
          x := x1 + (x2 - x1) *
               (ymax - y1) / (y2 - y1);
          y := ymax;
        end
      else if teBOTTOM in outcodeOut then
        begin
          x := x1 + (x2 - x1) *
               (ymin - y1) / (y2 - y1);
          y := ymin;
        end;
      if teRIGHT in outcodeOut then
        begin
          y := y1 + (y2 - y1) *
              (xmax - x1) / (x2 - x1);
          x := xmax;
        end
      else if teLEFT in outcodeOut then
        begin
          y := y1 + (y2 - y1) *
               (xmin - x1) / (x2 - x1);
          x := xmin;
        end;
      if (outcodeOut = outcode1) then
        begin
          x1 := x;
          y1 := y;
          CompOutCode(x1, y1, outcode1);
        end
      else
        begin
           x2 := x;
           y2 := y;
           CompOutCode(x2, y2, outcode2);
        end;
     end;
```

```
    until done;
    if accept then
      with Canvas do
        begin
          Pen.Color := clGreen;
          MoveTo(Round(x1), Round(y1));
          LineTo(Round(x2), Round(y2));
        end;
end;

end.
```



Kovács Lehel István

Katedra

A kérdéseken alapuló oktatás

Inquired Based Learning (IBL, avagy az irányított felfedeztetés)

A módszertanilag megalapozott kérdés alapú oktatás valójában a tudományos kutatási módszer lépéseit követi. Ez az információfeldolgozási modell lehetővé teszi, hogy a gyermekek felfedezzék az információ jelentését egy sor lépésen keresztül, ami egy bizonyos következtetés megfogalmazásához, illetve az új ismereten történő reflektáláshoz vezet. Leggyakrabban a tanár az ún. irányított kérdezésmódot alkalmazza annak érdekében, hogy elősegítse a tanulási tapasztalatok megszerzését, és hogy a kérdéseket strukturálja a tanítás sajátos céljai alapján. A kérdezésen alapuló oktatás egyaránt fejleszti a kritikai-, a kreatív-, valamint a problémamegoldó gondolkodást.

A kérdéseken alapuló oktatási módszer lépései:

- A probléma meghatározása (a kutatott témával kapcsolatos kérdés megfogalmazása)
- Adatgyűjtés (további kérdések megfogalmazása a vizsgált témával kapcsolatos információk begyűjtésére)