Vastag primitívek rajzolása

Képpontok ismétlése

Mozgó ecset

Területkitöltés

Közelítés vastag szakaszokkal

Vastag primitívek rajzolása

Több képpontnyi vastagságú vonalak

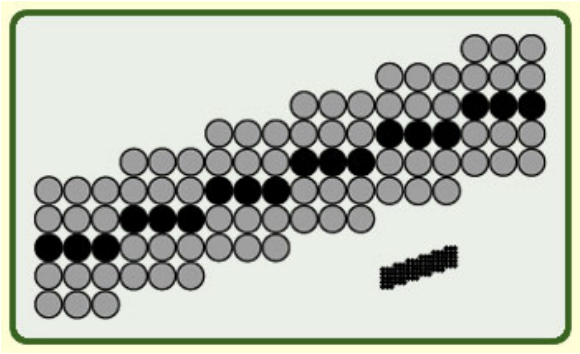
Milyen alakú legyen az ecset?

- Kör?
- Téglalap?
- Forduljon a vonallal?

Képpontok ismétlése

A pásztázó vonalas algoritmus kiterjesztése:

ha $-1 < m < 1$,	akkor a képpontokat többszörözzük meg az oszlopokban
különben	a sorokban töbszörözzük meg a képpontot



Képpontok ismétlése

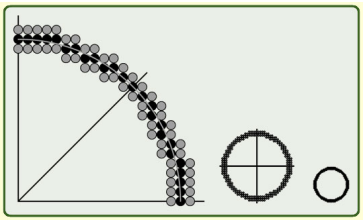
Tulajdonságai:

a) gyors,

b) a vonal végek mindig vízszintesek vagy függőlegesek

c) a vonal vastagsága függ a meredekségtől

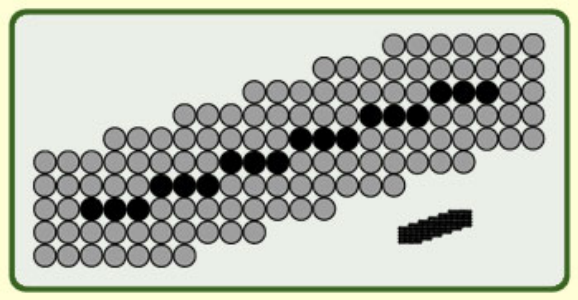
d) a duplázás nem megy: a vonal valamelyik oldala felé vastagabb.



Jó módszer, ha nem túl vastag a vonal.

Mozgó ecset

Téglalap alakú „ecset”, aminek a középpontja (vagy a csúcspontja) az 1 pixel vastag vonalon mozog (az ecset nem „forog”).



Mozgó ecset

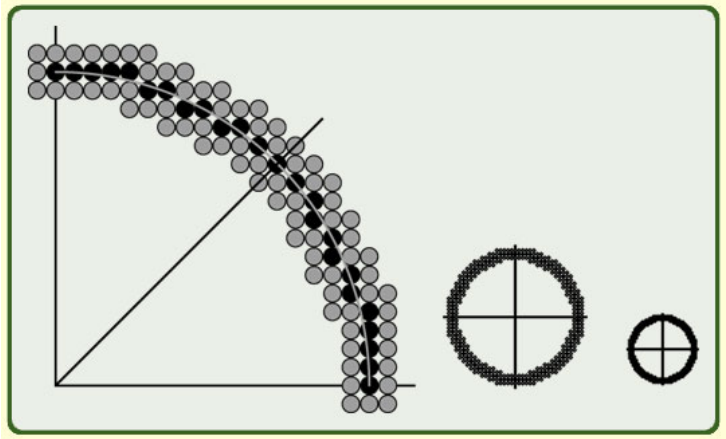
Tulajdonságok:
hasonló a *Képpontok ismétlésé*hez, de

a) a végpontok „nagyobbak”

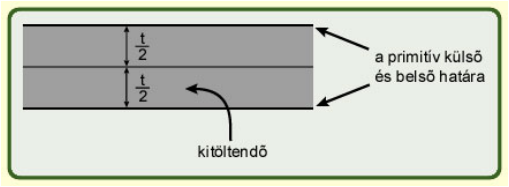
b) a vonal vastagsága függ a meredekségtől és az ecset alakjától. Jobb a kör alakú ecset.

Implementáció: ecset (=minta) másolása az 1 pixel vastag vonal minden pontjába.

Mozgó ecset



Területkitöltés



Terület primitíveknél a külső határvonalhoz használjuk az eredeti határvonalat, elegendő tehát a belsőt meghatározni.

Területkitöltés

Tulajdonságok:

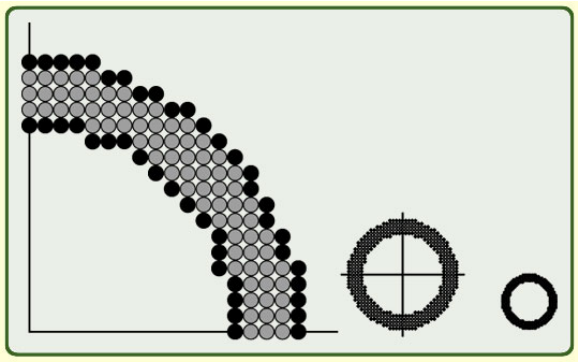
a) ugyanolyan jó páros és páratlan vastagságra

b) a vonal vastagsága nem függ a meredekségtől

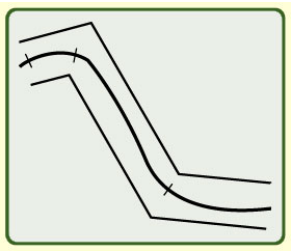
Kör esetén: külső és belső kör
Ellipszis esetén:

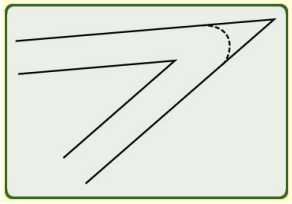
$$\left.\begin{array}{ll} a-\frac{t}{2}, b-\frac{t}{2} & \text{belső} \\ a+\frac{t}{2}, b+\frac{t}{2} & \text{külső} \end{array}\right\} \text{ellipszisek}$$

Területkitöltés



Közelítés vastag szakaszokkal





Szakaszonként lineáris approximáció

a) szép

b) vastag vonalakat símán kell illeszteni

Pont mérete - OpenGL

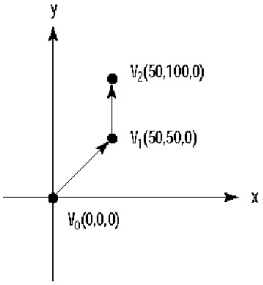
```
void glPointSize(GLfloat size);
```
Nem minden méretet támogatnak az implementációk:
```
GLfloat sizes[2]; // meret tartomany
GLfloat step; // tamogatott lepes

glGetFloatv(GL_POINT_SIZE, sizes);
glGetFloatv(GL_POINT_SIZE_GRANULARITY, & step);
```

Szakaszok rajzolása - OpenGL

Független szakasz (`GL_LINES`):
Az elsőként specifikált két csúcspont határozza meg az első szakaszt, a második két csúcspont a második szakaszt, ... (nincsenek összekötve).

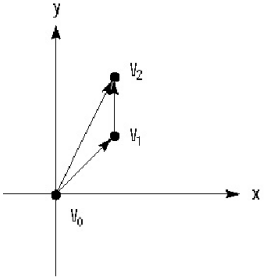
Szakaszok rajzolása - OpenGL

Szakasz sorozat (`GL_LINE_STRIP`):
Egy vagy több összekötött szakasz specifikálása a végpontok sorozatának megadásával. Pl.:




```
glBegin(GL_LINE_STRIP);
glVertex3d(0,0,0);
glVertex3d(50,50,0);
glVertex3d(50,100,0);
glEnd();
```

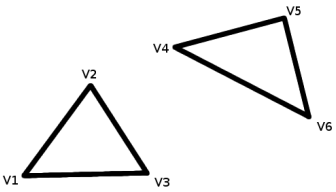
Szakasz hurok (`GL_LINE_LOOP`):
Ugyanaz, mint a `LINE_STRIP`, de az utolsóként specifikált csúcspontot összekötjük az elsőként specifikált csúcsponttal. Pl.:




```
glBegin(GL_LINE_LOOP);
glVertex3d(0,0,0);
glVertex3d(50,50,0);
glVertex3d(50,100,0);
glEnd();
```

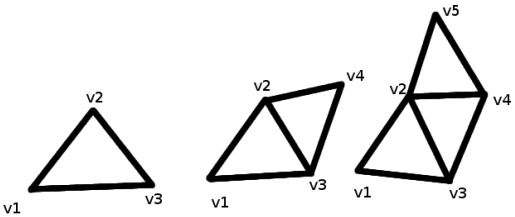
Háromszögek rajzolása - OpenGL

Független háromszögek (`GL_TRIANGLES`):
Az elsőként specifikált három csúcspont határozza meg az első háromszöget, a második három a második háromszöget, ...



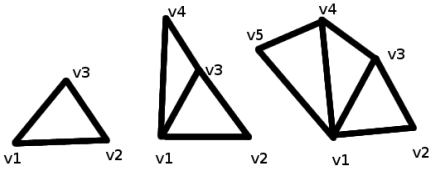
Háromszögek rajzolása - OpenGL

Háromszög sorozat (`GL_TRIANGLE_STRIP`):
Egy vagy több szomszédos háromszög specifikálása a csúcspontok sorozatának megadásával. Pl.:



Háromszögek rajzolása - OpenGL

Háromszög legyező (`GL_TRIANGLE_FAN`):
Egy vagy több szomszédos háromszög specifikálása a csúcspontok sorozatának megadásával. Pl.:



Vonal vastagsága - OpenGL

```
void glLineWidth(GLfloat width);
```

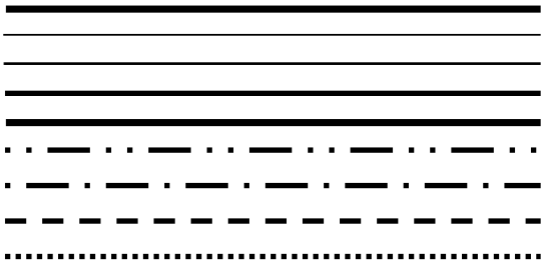
Nem minden vastagságot támogatnak az implementációk:

```
GLfloat sizes[2]; // vastagsag tartomany
GLfloat step; // tamogatott lepes
glGetFloatv(GL_LINE_WIDTH_RANGE, sizes);
glGetFloatv(GL_LINE_WIDTH_GRANULARITY,&step);
```

Szakaszok élsimítása - OpenGL

A szakaszok élsimítását engedélyezni a `GL_LINE_SMOOTH` argumentummal meghívott `glEnable`, tiltani a `glDisable` függvénnyel lehet.
Ha az élsímítás engedélyezett, akkor nem egész szélességek is megadhatók, és ekkor a szakasz szélén kirajzolt képpontok intenzitása kisebb, mint a szakasz közepén lévő képpontoké.

Vonal stílus



Vonal stílus

Primitívek attribútumai:

- vonal vastagság
- szín
- vonal stílus
- stb. ...

Vonal stílus:

- folytonos
- szaggatott
- pontozott
- a felhasználó által definiált

Vonal stílus

Stílus:
8 vagy 16 bites maszk írja le, hogy mely biteknek megfelelő pontok legyenek kirajzolva, mint a vonal pontjai

Pl: 11111111 = folytonos
11101110 = szaggatott

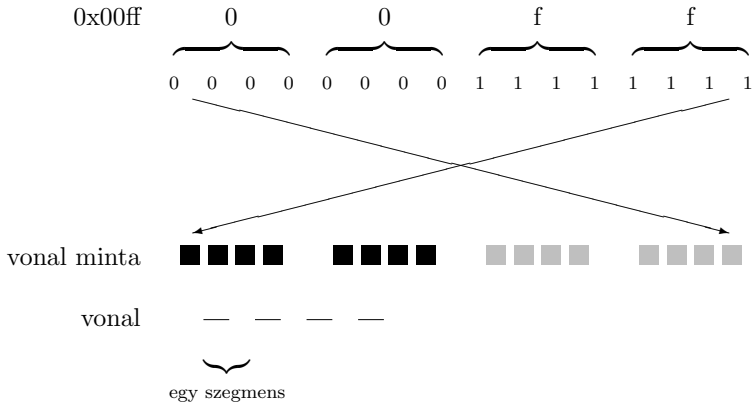
Rajzolás: (maszkolással)
Hátránya:
A szaggatások távolsága függ a meredekségtől.
Megoldás:
A távolságot számolva rajzolni a szakaszokat.

Szakasz stílus - OpenGL

```
void glLineStipple(int factor, ushort pattern);
```

Pattern (maszk): 16 bites bináris jelsorozat
factor: A *pattern*-ben lévő minden bit *factor*-szor kerül alkalmazásra.

Szakasz stílus - OpenGL



Szakasz stílus - OpenGL

Pl.:
`glLineStipple(1, 0x3F07);`
`glEnable(GL_LINE_STIPPLE);`
A minta: 0011111100000111
(az alacsony helyértékű bittel kezdünk).

`glLineStipple(2, 0x3F07);`
`glEnable(GL_LINE_STIPPLE);`
A minta: 00001111111111110000000000111111

Szakasz stílus
`glEnable(GL_LINE_STIPPLE)` engedélyezése
`glDisable(GL_LINE_STIPPLE)` tiltása

Szakasz stílus - OpenGL

Megoldandó feladat:
Rajzoljunk ötszöget olyan egyenes szakaszokból, amelyek a következő mintákból épülnek fel: