2. előadás

IX. 21.

Tematika:

1; bev.

2; szoftv. és hardverelemek

3; adott progrnyelv környezete → GY

4; kódelmélet (Huffman-kód ... stb)

5; szakaszrajzolás, körrajzolás ....

6; ábraszerkesztő → GY

7; interpoláció és approximáció

8; ponttranszformáció

9; legegyszerűbb láthatósági kérdések tárgyalása

www.ektf.hu/~emod/

felh. név : grafika ; 34Ø2

jelszó : midpoint cg2004faa

irodalomjegyzék:

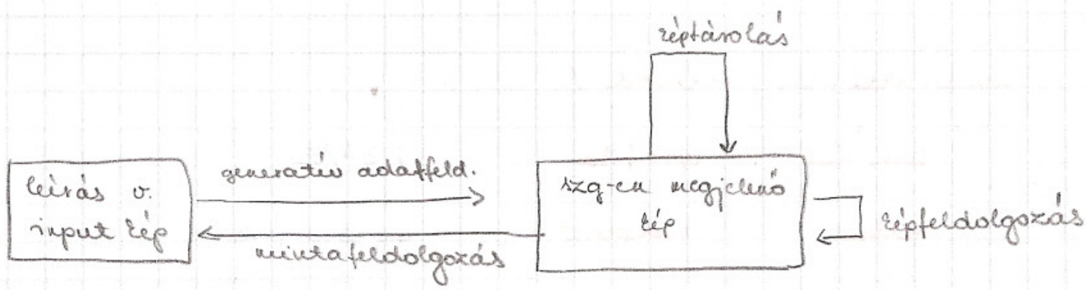
- Várady Lajos - jegyzet
- Word anyagot
- B-spline

Számítógépi grafika tárgya: $\boxed{\text{1. tétel}}$

- a szg-i grafika képpel foglalkozik
- '70-es években keletkezett
- kettévált → cgrafika
  → cgeometria

feladata (→ fő területei)

1) • generatív grafikus adatfeldolgozás, képek és grafikus adatok bevitele, képek előállítása, rajzolása — síkbeli, térbeli objektumok modellezése
2) • grafikus képek tárolása a szg-en és adathordozókon
3) • képfeldolgozás, képek javítása, alakítása
4) • mintafelismerés: képekről megfelelő információk kiolvasása, milyen információt, leírást kapunk.



A szg-i. grafika feladata

- Milyen grafikus file-formátumot használjunk? Kb. 100 ilyen formátum van, ennek vannak verziói is. Ezek a formátumok lehetnek veszteségesek és veszteségmentesek a tömörítés során.

jpeg → igényes tárolás

3; grafikus szoftverek nagy része ilyen

pl.: photo shop, gip (linux)

Az a szoftverek gondja, h. sokmindent ki akar szolgálni →

→ több a hiba.

4., ujjlenyomat - felismerő program → pl.:

- A szg-i grafikához kapcsolódó szoftverek: $\boxed{\text{2. tétel}}$

1) Művészet és animáció (Art and Animation)

- vita: a szg-en készített kép művészet-e.
- Victor Vasarelli, Szász Endre → mindketten használták a szg-et.

Szász Endre: Hollóházi porcelánra teszi rá a mintát szg segítségével (forgástestek)

- Renault emblémát Vasarellivel terveztették meg
- 2D-ban karakterek, hátterek tervezése
- 3D-ban modellek tervezése
- meghalt művészeket, v. nem létező embereket állítanak elő szg-en.
- első egész estés mozifilm: Toy Story → egésze szg-pel készült. (Shrek; Z, a hangya

Maya → kajtervező

szájmozgás a szöveg után lett tervezve

2., Szg-pel segített tervezés és gyártás:

- Computer Aided Design (CAD)

• automatikus keresztés, tervezés, vázlatkészítés, munkadarabok és áramkörök tervezése, kapcsolási rajzok készítése

• kifejezetten tervezéshez használják

- Computer Aided Manufactoring (CAM) → gyártás
- Computer -"- Engineering (CAE)

• mérnöki tervezéshez használják

• ez rokon a CAD-del

• inkább a gyártási rendszerek tervezését célozzák

3.) Bemutató és üzleti grafika (Presentation and Business Graphics)

• piktogramokat és diagramokat előállító táblázatkezelők (Excel)

• bemutató és slideshow készítő programok (Power Point)

4) Tudományos és műszaki szemléltetés és szimuláció (Scientific Visualization and Simulation)

- ide tartozik: fogalmak, jelenségek, törvényszerűségek, folyamatok szemléltetése...

• költségesen vizsgálható dolgok szemléltetése: autók futása, katonai alkalmazások, űrhajós képzés

• közvetlenül nem vizsgálható dolgok: galaxisok mozgása, mikroszkópikus dolgok

• műholdak pályája

• fraktálok és mandelbrot halmazok előállítása és felhasználása

+ szimuláljuk a túl rövid v. hosszú idő alatt végbemenő dolgokat.

5) Képelemzés és feldolgozás (image analysis and processing)

• képek részleteinek felismerése

• tenyérlenyomatot felismerő programok

nagy frekvenciájú rádióhullám működik

• orvosi képfeldolgozás: computer tomográf (CT), MR, ultrahang, digitális radiológia

· karakter felismerő programok (OCR - Recognita, Offic 2003 → tartalmaz magyar karaktereket)

6) Grafikus kezelő felületek (Graphics User Interfaces, GUI)

- ember és szg. közötti interakciót elősegítő programok rendszere

- a grafikát nem a program tárgyaként használják

→ lehet vizuális pontokra pontozni ⇒ átalakítást képek (ábrák), amit megtekintendő

3. előadás IX. 28.

25 másodperc → ma már 10-12 millisec.

Ha nem kicsi a válaszidő rasteres (fodrás lesz a kép)

Sütinyomtató → por és vízzel nyomtat

——— ◦ ———

- hang, mint eszköz is megjelenik

- gyorsabban lehessen az interakciót végrehajtani → legyen grafikus a felh. -i felület

- linuxban lehet váltogatni

- vannak szabványok, amiket be kell tartani. (nem lehet a file menü az alsó sorokban)

- el lehet térni a szabványtól, de nem fogják venni a programot.

- már vannak a WINDOWSBAN ablak nélküli programok (eltűntek)

- a legördülő menüt használjuk

- vmelyik parancsban legyen formában, javasolt a menük sorrendje is. file ... help.

- ajánlás: fájl-ban legyen kilépés v. nyomtatás
- pl.: mediaplayer
- a lejátszó programok valós lejátszóhoz akarnak hasonlítani → itt nincs szabvány
- XP: van tálca, munkaasztal → lehet ide fájlokat, mappákat tenni (pl.: szövegrészletet nem lehet)
- a munkaasztal háttere színes (ez változtatható)
- 2D-ós felület → csináljunk 3D-sat. Lehet, megközelítő zoom-olással.
  ↳ nem biztos, h. megvalósul, de a felhasználók nagy részének nem jó a térlátása

- Paint:
  menü
  eszközök
  munkaterület
  paletta
  státuszsor

Ezek az eszközök kevesek → lebegő eszköztárak (több közül lehessen választani → információk követő kiírása)
Vannak olyan szoftverek, melynek másak a felületeik. (pl.: fotósoké)

7., Virtuális valóság: (Virtual Reality)
- mesterséges szimulált környezetek létrehozása a számítógéppel (+ virtuális sisak, kesztyű)
- lehet interneten át ezt elérni pl.: múzeumban lehet szobrokat nézelődni.
- legelterjedtebbek a játékok.

- jelentős alkalmazásai: oktatás, távműködtetés, játék, internet

8, Multimédia alkalmazások:

- egy átlagos szem. számítógép lehetővé teszi a használatát
- MMX processzor 1. Pentium 1.
- World wide web ⇒ internet multimédiás felülete alkalmazása a szg-i grafikának.
- ezeknek az ismereteknek a tanítása új feladat a közoktatásban

9, Számítógépes játékok:

- a grafikus adatfeldolgozás legelterjedtebb felhasználása ⇒ valósághű programok ⇒ egyre jobbak a grafikus szoftverek
- a játékok azt sugallják a felhasználónak, h. drágább és nagyobb teljesítményű szg-et vásároljon
- DirectX ⇓ PC microsoft, Open GL ⇓ Linux
- internet lehetővé teszi, h. a világ bármely részéről játszhatnak egymással a felhasználók
- bizonyos játékok DirectX 9.0c, mások Open GL alatt futnak
- fejlesztik a DirectX-et ⇒ újabb grafikus kártyák kellenek.

WINDOW/system 32/directX ⇒ diagnosztikai eszköz

- nem csak a grafikáról van szó, hanem a hangról is.

- IGP: alaplapra integrált kártya

Open GL : csövez (3D) sorozatra

Iráplada ⇒ microsoft

Klón funkció ⇒ lemásolja az egyiket a másikra

két monitoron 2 mutatóval van

van egy SVHS kimenet ⇒ TV-re csatlakoztatható

- ha video kép megy, külön be kell állítani.

(Megjelenítés !)

- átfedés egyik ablak alatt a másik elemeit is látjuk
- Z puffer ⇒ láthatósághoz kell.

- Általános az SVGA üzemmód, 256 szín v. 65536 szín használata

(A Grafikai Programozás Fekete Könyve)

2. tétel

Grafikus hardver

Billentyűzet:
- grafikában nevet kapnak
- billentyűkombinációk + funkcióbillentyűzet
- vágólap kezelése szabványos csak

Digitalizáló v. graf. tábla:

ennek álló (A2, A3 méretű)

szálkereszt ⇒ pozícióhoz tartozó koordinátát olvassa le.

mérete 150 x 200 cm

grafikus tábla mérete: 20 x 20 cm – 1 x 1 m.

állványra rakható ill. asztalon is elhelyezhető (graf.)

graf: pálca, fényceruza, korong szálkereszttel, az egér is használható digitalizáló eszközként.

-rajzokat könnyebb készíteni (könnyebb annzával rajzolni), azt nem tudják érzékelni, h. mennyire nyomják le a ceruzát.

Beviteli eszközök 4. előadás X.5.

- egér, trackball, joystick
- trackpad, botkormány ⇒ laptopoknál
- érintőérzékeny ⇒ érzékelő képernyő touchscreen
- fényceruza ⇒ ábrákon
  orvosi diagnosztikában pl. ultrahangon

  lehet nagyítani, kicsinyíteni ... stb.
  ez csak annak jó, aki régen nem tanult számítógépezeslst

- optikai olvasók : image scanners

  - kézi scanner ⇒ kézzel huzogatjuk 10-12 cm-es scannert.
  - lapolvasó ⇒ mechanika olvassa be, biztonságosabb

  • felbontás : DPI hány pontot olvas be inchenként
  • színmélység : ma már mindegyik színes (képen sok 8 bit)
  • sebesség : milyen porton csatlakozik ⇒ USB port
    ⇒ kommunikációs port ⇒ lassabb
    ⇒ SCSI ⇒ gyorsabb, de kártya kell hozzá
  • szoftver : előszűrés ⇒ a k. lap mely területét akarjuk bescannelni

adagolók : legtöbb tud diafilmet v. negatívfilmet scannelni

• vannak már állóscannerek

- vonalkód olvasók : barcode

  mindenütt használják a világon

- digitális fényképezőgép és kamera
optika a fontosabb, nem az elektronika

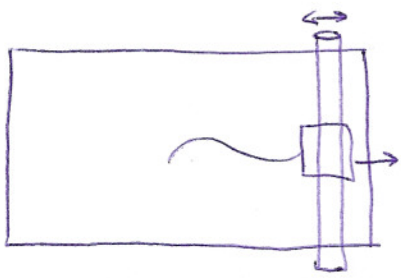
kiviteli eszközök

1) - hard-copy perifériák ⇒ tartós formátumban állítják elő
   - nyomtatók DPI, méret, színes, sebesség, nyomtatószoftver
   - rajzgépek: plotter

tollas
tintasugaras > rajzgép

tollasak ma már nem gyártanak, tintasugaras van

+ hengeres rajzgép
+ asztali rajzgép A4, A3, A2 ... asztalon van, fix a mérete



hengeres rajzgép · jobb a mechanikája

be kell fűzni a papírt, és egy nagy henger mozgatja a papírt

a rajzgépek tintasugaras elven működnek

jell.: felbontóképesség:

+ címezhetőség DPI (dots / inch)
↳ 2,54 cm

ha meg tudja címezni, tud oda pontot tenni

ez nagyobb, mint a valós felbontás

+ pontméret
dot size

04. 12. 14

Grafika előadás

14:53 Laci még nincs itt

15:10 Itt van.

Láthatósági algoritmusok — 8. tétel

Hidden Line (Vektoros) - takart vonalak

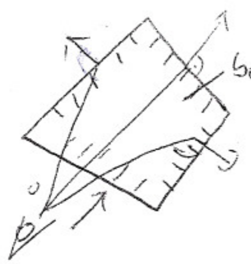
Hidden Surface (Raszteres)

Object space - tárgy tér algoritmus. A tárgy térben eldöntjük a láthatóságot

Image space A képernyőbe eső objektumokról döntjük el a láthatóságot

M Takart / Hátsó | lapok eltávolítása

Megpróbáljuk kiszűrni azokat a lapokat amelyek élei nem látszódnak.



Ez az egy fal látszik a másik 3 nem

Meghatározzuk a lapnormálisokat. Meghatározzuk a nézési irányt és a normálisok szögét.

Konvex poliéder esetén Bármely két pontját összekötjük minden pontja a testen belül van

Ha φ hegyesszög → Látszik a lap | Ha φ derékszög → Élben látszik

- || - tompaszög → Nem látszik

1

Ha a lapnormális nem kifelé hanem befelé mutat akkor is működik ez a megoldás

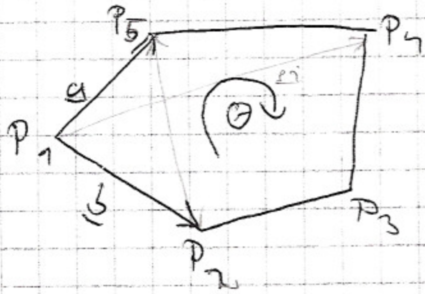
Feladat: (1) A lapnormálisok előállítása

(2) A f meghatározása

(3) Mi van a konkáv esetben

(1) Bármilyen felület normálisát úgy határozom meg, hogy veszek a felületi pontban a két vektort és a két vektor vektoriális szorzatával állítom elő.

a | b két vektor



$\underline{a} = \underline{P}_i - P_1$

$\underline{b} = \underline{P}_n - P_1$

$\underline{a} \times \underline{b}$ → két vektor síkjára merőleges vektor

$\perp$ merőleges a és b síkjára és $\underline{a}, \underline{b}, a \times b$

sorrendben jobbsodrású rendszert alkot.

Tehát ez egy ilyen vektor.

Jobbsodrású amíg jó, mert ha mindig ilyen sorrendben vesszük, az összes vektor vagy kifelé vagy befelé mutat.

$$\underline{h} = \underline{a} \times \underline{b} = \begin{vmatrix} \underline{i} & \underline{j} & \underline{k} \\ a_x & a_y & a_z \\ b_x & b_y & b_z \end{vmatrix}$$

az x, y, z tengelyen (tengelyen) vett egységvektor

ennek a determinánsnak (determinánsnak) az értéke (értéke)

az eredmény egy egységvektor lesz.

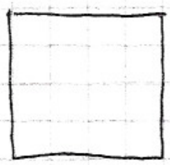
Kifejtés Sarrus (Sarrus) szabály (szabály)

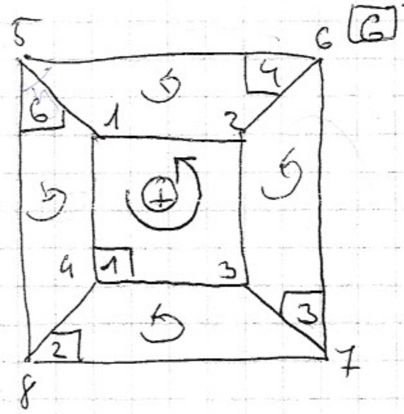
$$\begin{vmatrix} \underline{i} & \underline{j} & \underline{k} \\ a_x & a_y & a_z \\ b_x & b_y & b_z \end{vmatrix} =$$

$$= \overbrace{(a_y b_z - a_z b_y)}^{h_x}\,\underline{i} + \overbrace{(a_z b_x - a_x b_z)}^{h_y}\,\underline{j} + \overbrace{(a_x b_y - a_y b_x)}^{h_z}\,\underline{k}$$

a 3 koordináta

<u>kockánál</u>

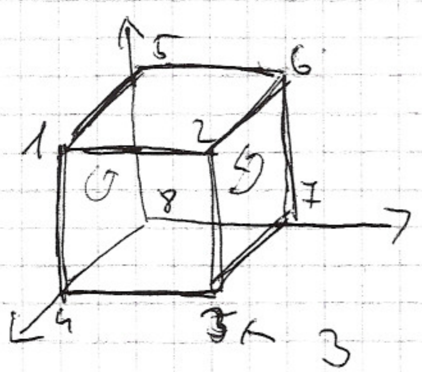




ezen körülötte a körüljárás irány szerint

1. lap	2. lap	3.	4.	5.	6.
1	4	2	1	1	5
4	8	3	2	5	6
3	~~7~~ 7	7	6	8	7
2	3	6	8	4	8

6. lapnál fordított irány amelyik lapot szemléltetjük azt fordítva soroljuk fel



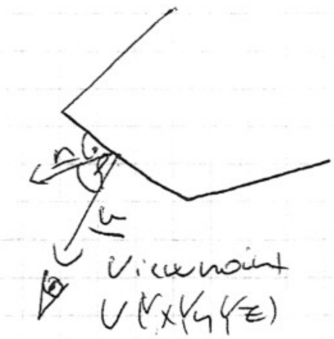
Mi van ha elrontom a körüljárás irányt, akkor azon a lapon az a és b vektor felcserélődik és pont az ellenkező irányba fog mutatni.

Tehát nem befelé hanem kifelé fog nézni.

② φ meghatározása

Ha két alakzat szögét meg akarom határozni akkor skaláris szorzat

$\underline{v}, \underline{n}, \underline{f} = \varphi$



Vizsgálópont úgy határozom meg hogy V koordinátáiból kivonom a lap 1' koordinátáit így kapom meg a (kivonás) vektort

Gyakran ez a pont P_1 = Súlypont

Összeadjuk x, y, z koordinátákat és külön külön elosztjuk a csúcsok számával

Skaláris szorzat

$$(\underline{v}, \underline{n}) = |\underline{v}|\,|\underline{n}| \cdot \cos\varphi$$

Ha a két vektor egységvektor, normalizálva akkor a skaláris szorzat $\cos\varphi$

Normalizáljuk $\underline{v}$ és $\underline{n}$-et elosztjuk a koordinátáit a hosszával

$$\underline{v} = (v_x/h,\ v_y/h,\ v_z/\overset{\text{hossz}}{h}) = \text{koordináták négyzetének az összege}$$

$$h = \sqrt{v_x^2 + v_y^2 + v_z^2}$$ innentől kezdve a $\underline{v}$ vektor hossza 1

$\Rightarrow (v, n) = |\tilde{v}|\,|\tilde{n}| \cdot \cos\varphi = \cos\varphi$ / komponensek skaláris szorzatának a összege

$|\tilde{v}| \quad (\tilde{v}, \tilde{n}) = \tilde{v}_x \cdot \tilde{n}_x + \tilde{v}_y \cdot \tilde{n}_y + \tilde{v}_z \cdot \tilde{n}_z$

mert kell tudni hogy normalizált

Kell e venni φ konkrét értéke NEM

Ha φ hegyes $\cos\varphi > \oplus$ pozitív

φ tompa $\cos\varphi < \ominus$ negatív

tehát elég ennek az összefüggésnek venni az előjelét

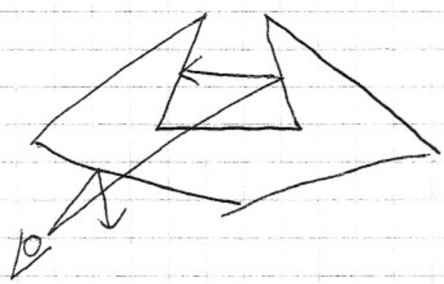
Ha > 0 vagyis a skaláris szorzat az a pozitív

akkor látszik a lap

Ellipszoid a kellene, akkor is érvényes az összefüggés

Akkor lenne $\emptyset$ ha nem lenne

③ Konkáv ... nem minden esetben > 0



Konkáv eset javítására festő algoritmus Painting

– depth sort végsőként rendezés azokat melyik látszik, (hegyesszöget zár be)

5

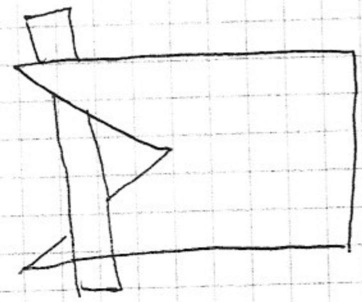
Ha hátulról előre rakom ki a lapokat síkonként legalább őket : fal festésének előnyei a néző után fa → lombok

Ez nem tökéletes megoldás de sok esetben tökéletes

Kell hozzá egy gyors rendező algoritmus

A z koordináta szerint rendezik sorba mert sorban az x, y a sík és z irányából nézzük

Depth sort : A különleges eseteket is kezeljük, amikor nem eldönthető a sorrend.



Ez a két lap kölcsönösen takarja egymást.

Megoldás kellek az eseteket kezelni egyik alakzattal elvágni a másikat

Ez már nem kell ↓

Z-buffer – elrendezik z koordináta szerint kirajzolni a pontokat

<u>Vizsgakérdések</u> :

1. A computer grafika tárgya
(Grafikus adatfeldolgozás mit jelent)

A computer grafikai rendszerek csoportosítása

2. Hardware (computer grafika Hardware)

Videókártyák, felbontások, színmélység

Processzor a kiszolgáló üzemmódok

3. Grafikus file formátumok

Raszteres és vektoros grafika

Grafikus képek kódolása

4, Alapvető Raszteres algoritmusok

a, Window Viewport
b, Szakaszrajzoló DDA
Midpoint
c, Vágás Cohen Sutherland

5. Görbék

Interpoláció : csak az Hermit

Approximáció : Bezier görbe

6. Pont transzformációk

Homogén koordináták 2D-ben 3D-ben

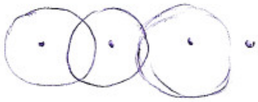
Általános affinitás
Transzformációk egymás utáni végrehajtása sorosan

7 Térbeli transzformációk

Tér leképezések síkra párhuzamos, centrális

8. Láthatóság

Szigorlaton OLE technika alapfogalmak kérdések



megoldás : tesztábra nyomtatása
egybefolyik-e vagy nem

meg lehet nézni, h. 1 inch-en hány vonalat
tud kirajzolni a gép, amikor, h. összefolyna

2) - display : megjelenítők, monitorok

lehet egy pici vékony, ami információt hordoz →
display nem monitor

Alapvető raszteres algoritmusok

4. tétel

1) szakaszrajzoló algoritmus

Pixel → Picture Element

raszter → pontokból rakja ki az alakzatot

kritérium :

a, szakasz egyenesnek kell, h. látszódjon

· megvalósíthatatlan (közelről nézve)

kivéve : 0°, 90°... stb., vízszintes, függőleges

b, fedettség

· fényesség, ha sötét háttéren világossal rajzolunk.

· jó, ha állandó

· ne függjön a szakasz vastagságától

- a lépcsőzetes hatás elkerülése végett képségjavító technikákat használnak: antialiasing

c, gyors legyen

DDA ⇒ gyakorlat

<u>midpoint</u>

- gyors és pontos algoritmus
- ugyanazokat a pontokat állítja elő, mint a Bresenham,

a két algoritmus egymásba transzformálható

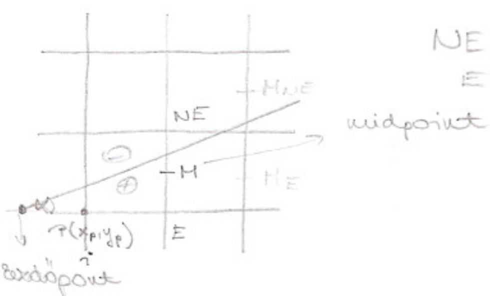
<u>Feltételek:</u>

a) $\alpha < 45°$

$$y = mx + b$$

$$tg\,\alpha = m = \frac{\Delta y}{\Delta x} \qquad \Delta y = y_2 - y_1$$
$$\Delta x = x_2 - x_1$$

x_1, x_2: kezdőpont koordinátái
y_1, y_2: végpont -,,-



NE: north east (ÉK)
E: (K)

Azt világítja ki, amelyik a legközelebb esik az ideális egyenesvonalhoz.

Megnézzük, h. M az ideális nyomvonal alatt v. felett van.

alatt ⇓ NE-t választjuk

felett ⇓ E-t választjuk

$$y = \frac{\Delta y}{\Delta x} \cdot x + b \Rightarrow \emptyset = \Delta y x - \Delta x y + \Delta x \cdot b$$

Csinálunk ebből egy kétváltozós fgv-t.

$$F(x; y) = \Delta y \cdot x - \Delta x \cdot y + \Delta x \cdot b$$

A egyenes alatt ⊕, alatta ⊖ a fgv. Ha behelyettesítjük a midp.-t, és ⊖-t ad, az ellenpontot ~~kegyetlen~~ választjuk, és fordítva.

döntésvált. $d_i = F(x_p + 1, y_p + \frac{1}{2}) = \Delta y\,(x_p + 1) - \Delta x\,(y_p + \frac{1}{2}) +$
$+ \Delta x \cdot b$

Ha $d_i \geq \emptyset \rightarrow NE \quad (x = x + 1;\ y = y + 1)$

vastag vonal esetén egyenlőség-kor a felső csúcsot választjuk, h ne legyen csipkézett

$$d_{i+1} = F\left(x_p + 2, y_p + \frac{3}{2}\right) = \Delta y\,(x_p + 2) - \Delta x\left(y_p + \frac{3}{2}\right) + \Delta x \cdot b.$$

döntési változó növekménye $\Delta NE = d(i+1) - d_i = \underline{\underline{\Delta y - \Delta x}}$

Ha $d_i < \emptyset \rightarrow E \quad (x = x + 1;\ y = y)$

$$d(i+1) = F\left(x_p + 2; y_p + \frac{1}{2}\right) = \Delta y \cdot (x_p + 2) - \Delta x\left(y_p + \frac{1}{2}\right) + \Delta x \cdot b$$

$$\Delta E = d(i+1) - d(i) = \underline{\underline{\Delta y}}$$

Kezdőérték: $x_p = x_1 \quad ; y_p = y_1;$

első középpont $d(1) = F\left(x_1 + 1, y_1 + \frac{1}{2}\right) = \Delta y\,(x_1 + 1) - \Delta x\left(y_1 + \frac{1}{2}\right) + \Delta x \cdot b =$

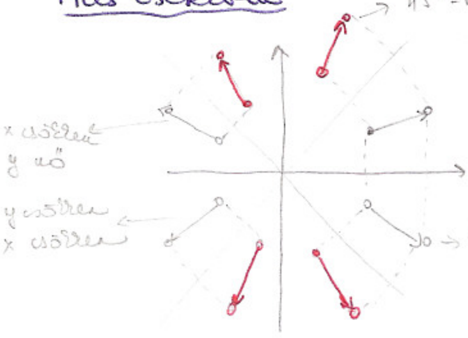
$$= (\Delta y \cdot \underbrace{x_1 - \Delta x \cdot y_1 + \Delta x \cdot b}_{0}) + \Delta y - \Delta x/2 = \Delta y - \Delta x/2$$

algoritmus:

```
d := 2 * dy - dx ;        → 2Δy - Δx
x := x1;     y := y1;

for i := 1 to dx          → (x2 - x1)

begin

putpixel (x, y, color)
if d ≥ 0 then
begin
x := x + 1;
y := y + 1;
d := d + 2*(dy - dx)      a fenti miatt (*)
end

else begin

x := x + 1;
d := d + 2* dy            (*) miatt

end;

end; {for}
```

- Nincs benne osztás → gyorsabb a DDA-nál.

Más esetekben → 45°-nál nagyobb hajlásszögnél x és y szerepe felcserélődik



→ tükrözéssel: x nő, y csökken

MIDPOINT a NETEN!

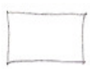
DDA: digitális különbség elemző
digital differencia analizátor

5. előadás x. 12.

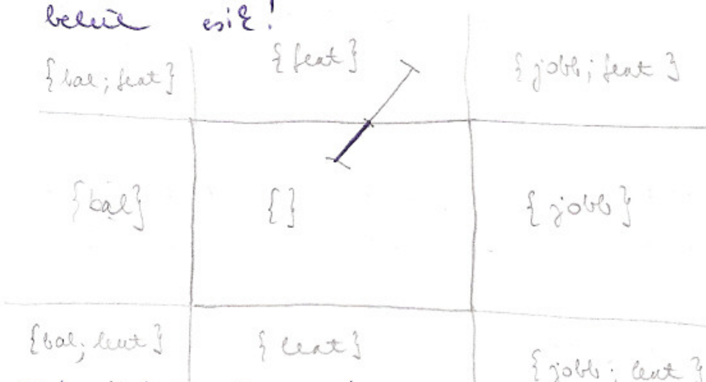
Lehatárolások:

• határoljunk le egy szakaszt egy téglalappal → Cohen-Sutherland-algoritmus

• vágóablak: oldalai páronként ||-os az x és y tengellyel



Határozzuk meg a szakasz azon részét, amely az ablakon belül esik!



Előszűrést kell végezni.

Alg.: - meghosszabbítjuk az oldalakat (a síkot 9 részre osztjuk fel)

1) kódszámítás: szakasz két végpontjához hozzárendelünk egy-egy halmazt.

2) szűrés: megvizsgáljuk, h. ezen halmazok metszete, uniója milyen lesz.

- ha $c_1 \cap c_2 \neq \{\}$ pl.: 

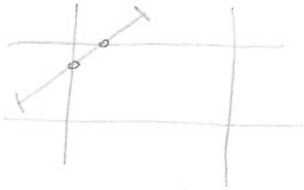
nincs vágás, nem látszik, ugrás a 4.) pontra

- ha $c_1 \cup c_2 = \{\}$ pl.: 
nincs vágás, látszik a szakasz, ugrás a 4.)-re

3) ha egyik feltétel sem teljesül, akkor <u>VÁGÁS</u>
• meg kell nézni, hogy c_1-ben és c_2-ben mi van

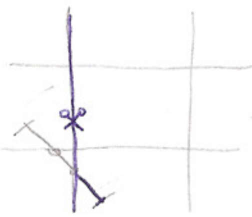
vágás, ugrás 1.)-re

4., vége



→ nem egy menetben határoljuk le, hanem kettőben, de a sorrend lényegtelen.

- akkor lépkemünk ki az algoritmusból, ha látni, v. nem látni → két eset van.

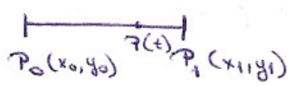
- 

nem egy menetben történik a szűrés
↓
a ∩ üres, az ∪ nem → vágni kell

baloldalival vágunk
- a kis szakasz (~~w~~ maradék) végpontjaihoz kódot rendelünk, mert a határvonal nincs benne. a ∩ ~~nem~~ üres → kilépünk.
2 lépésben történik a „vágás"

Vágás: két egyenes metszéspontja

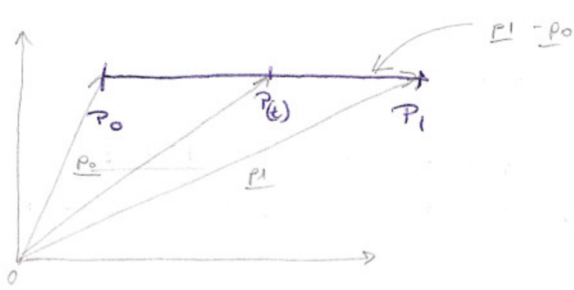
- szakasz megadása



szakasz egyenlete ⇒ meg tudjuk határozni a P-t.

$P(t)$: a szakasz t paraméterű pontja.

$P(t) = \underline{P}_0 + t(\underline{P}_1 - \underline{P}_0)$ $t \in [0;1]$ min 0, max 1.



$P(\emptyset) = P_0$

$P(1) = P_1$

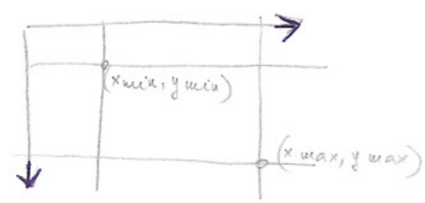
A szakasz megadása ezzel térben is.
A szakasz paraméteres vektoregyenlete.

$$x(t) = x_0 + t(x_1 - x_0) \qquad t \in [0, 1]$$
$$y(t) = y_0 + t(y_1 - y_0)$$

A szakasz paraméteres vektoregyenlet-rendszere térben még: $z(t) = z_0 + t(z_1 - z_0)$

Ebben az esetben a t paraméter tartománya változik.

pl.: fent:



$y = y_{min}$

és ennek keressük meg a vektoregyenlettel a szakasz metszéspontját → meg kell határozni a t-t.

$$y_{min} = y_0 + t(y_1 - y_0) \quad \Rightarrow t = \frac{y_{min} - y_0}{y_1 - y_0}$$

$$x(t) = x_0 + \frac{y_{min} - y_0}{y_1 - y_0}(x_1 - x_0)$$

Ez alapján a többi levezethető!

A megkapott egyenest megkaphatjuk két egyenes metszéspontjának használatával (Négyjegyű)

Ha általánosítunk, a térben egy szakasz és egy sík metszéspontját kell meghatározni.

- ha nem halmazműveletekkel oldjuk meg
- ha nem ||, akkor bonyolódik a kódszámítás és a vágás

A feladat általánosítása térben:

szeretnénk egy térbeli szakaszt lefedni egy téglatesttel.

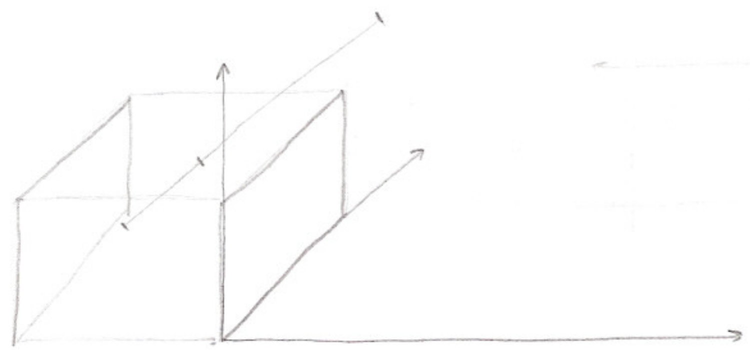
A téglatest a teret 27 részre osztja.

A halmaz számossága lehet 0, 1, 2, 3 : → bal lent mögötte
bal → bal lent

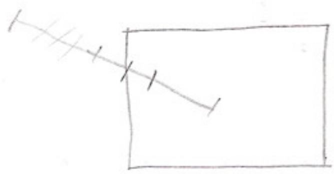
- a vágás ugyanúgy történik, csak síkokkal.

- $y = y_{min} \Rightarrow x$ és z koordinátája lényeges (fent)

A (2.) nem változik.



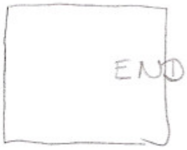
Középpont:



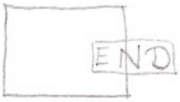
Felezgetéssel is meg lehet határozni a metszéspontot. Ami kívül esik, elhagyjuk

Kövegek lehatárolása:

el lehet dönteni egy magába foglaló téglalappal, ha a széle kívül van, nem látszik.

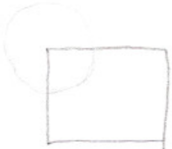


vagy elhagyjuk vagy vágjuk



a magába foglaló téglalap középpontja a téglalapban van ⇒ vágunk, különben elhagyjuk.

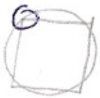
Speciális alakzat vágása:



8 metszéspont van, amely meghatározható a kör egyenletéből.

origó középpontú r sugarú kör egyenlete: $x^2+y^2=r^2$
nem origó középpontú ⇒ $(u; v)$ középpont

Lehet paraméteres egyenlettel felírni a kört.



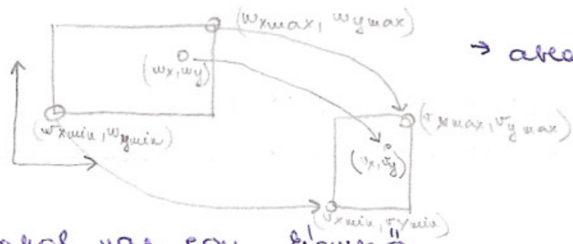
4 pont esetén a pontot kétszer kell számolni

Tempont: nem csak az ívet, hanem a két pontot összekötő szakaszt is meghatározzuk.

módosítása a téglatest és a gömb -re vonatkozik.

WINDOW – VIEWPORT

definiáljuk a problémát.



→ ablak

valahol van egy képernyő

Mi lesz a pont koordinátája, ha ismerjük:

(w_{xmin}, w_{ymin}); (w_{xmax}, w_{ymax})

(v_{xmin}, v_{ymin}) ; (v_{xmax}, v_{ymax})

(w_x, w_y)

Lineáris leképezés:

$v_x = A w_x + B$

$v_y = C w_y + D$

ha a függőleges oldalak aránya = a vízszintes oldalak arányával (kört, körbe viszi át) azt normalizált transzformációnak nevezzük. (Normalizált eszköztranszformáció)

6. előadás

Akkor lineáris a leképezés, ha tudunk olyan összefüggést találni, ami: $v_x = A w_x + B$

$$w_x = w_{xmin} + a(w_{xmax} - w_{xmin}), \text{ ahol } 0 \le a \le 1$$

$$v_x = v_{xmin} + a(v_{xmax} - v_{xmin})$$

$$a = \frac{w_x - w_{xmin}}{w_{xmax} - w_{xmin}}$$

$$v_x = v_{xmin} + \frac{w_x - w_{xmin}}{w_{xmax} - w_{xmin}} * (v_{xmax} - v_{xmin})$$

$$v_x = \underbrace{\frac{v_{xmax} - v_{xmin}}{w_{xmax} - w_{xmin}}}_{A} \cdot w_x + \underbrace{\left(v_{xmin} - \cancel{\frac{w_x w_{xmin}}{w_{xmax} - w_{xmin}}}\right)}_{B} \rightarrow w_{xmin} \frac{v_{xmax} - v_{xmin}}{w_{xmax} - w_{xmin}}$$

$v_y = C w_y + D$ → minden x helyére y-t kell írni.

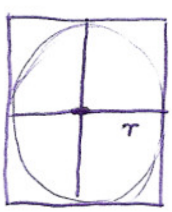
A, B, C, D : konstansok → a kép átellenes sarkainak koordinátái.

Normalizáltnak nevezzük a transzformációt, ha az arány nem változik kör → körbe ... stb.

Ha nem normalizált: négyzet → téglalap lesz.

- ha a<0 v. a>1 → ablakon kívüli pont, de ez a képernyőn kívülre kerül.

<u>Feladat:</u>

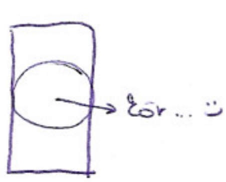


Akkor tudjuk megcsinálni, ha a torzítási tényező 1., azaz, ha a raszter oldalainak az aránya megegyezik. (vagyis a pontok)

$\frac{xasp}{yasp}$ → torzítási tényező

		felb.	xasp	yasp
pl.:	Vga Hi	640 x 480	10000	10000

Ha torzít:



Windowsban a torzítási tényező 1.

<u>Megoldás:</u> A torzítási tényezővel korrigálunk.
A függőleges koordinátákból kivonjuk v. megszorozzuk, valamilyen módon változtatunk.

```
begin
  getaspectratio (xasp, yasp)
  rectangle (x-r, y-round(r*(xasp/yasp)), x+r, y+round(r*(xasp/yasp)))
  circle (x,y,r);
end;
  var  xasp, yasp : word;
```

zárójelezéssel oldjuk meg, h. pontos legyen az eredmény

Cohen (4 paramétert kell megadni, jobb és bal átellenes sarok)

```
       kezdőp   végp.               látszik a szakasz
Cohen(var x1, y1, x2, y1 : longint; var üres

          kimkod

procedure kod (x,y : double; var c : kimkod)
      begin
        c := [];
        if x<min then c:=[bal] else
           if x>max then c:=[jobb];
        if y<min then c:=c+[felso] else
           if y>max then c:=c+[also]

      end;
begin
  kod (x1,y1,c1)
  kod (x2,y2,c2);
while ((c1+c2<>[]) and (c1*c2=[])) do begin
        c := c1;
        if c=[] then c:=c2
                  :
        if c=c1 then begin
```


x és y változott új x, y-nal meghívjuk az új adatokra az eljárást.

különben x2, y2-re tesszük ugyanezt

vizsgáljuk, mert léptünk ki

üres := c1* c2 <> [] Ha igaz, a szakasz nem látszik.

t : segédváltozó, hogy tükrözött -e. Letöltésekben VAN

anonymous pub aries.elte.hu
Computer Grapics

www.elte.hu/~cmod

7. előadás X.26.

Grafikus képek kódolása és tárolása 3. tétel

Tanári letöltésekben van közzé anyag
WORD → grafikus képformátumok.

Ez itt egy kukacska
Át mászhatna rajta!

Letter: 8,5" x 11"
21,54 cm x 27,94 cm

felbontás: 600 dpi

⇓

8,5·11·600·600 = 33 660 000 pont

1 pont ábrázolásához 24 bit szükséges → ≈ 100 Mbyte
(100 980 000 byte)

graf. képek kódolása

① szűrés (mapping)
szűrésekkel redundanciák kiszűrése → kevesebb byteból áll, mint az eredeti kép

② leszűkítés (quantization): az input kép leszűkítése a lehetséges output méretre.
pl.: scannerek fellapjánál, ti. „előszkennelik" a képet. 5100×6600 pont helyett 640×480-as felbontású képernyőn elférő képet hoz létre → megadjuk, h. a kép melyik része kell.

③ kódolás: a kódolt, tömörített adat, a megfelelő kódtábla előállítása

· kódoljuk, de nem vesz el adat
pl.: *.tif, *.bmp

• faxtömörítés ⇒ gyenge minőség, de küldhető adat

Kódolási eljárás:

1) Huffman-kód:

lényege: ami sokszor fordul elő, akkor rövid kódot rendelünk, a kevésszer előfordulókhoz hosszabbat.

folytonosan tároljuk ⇒ szeparálhatóknak kell lennie ⇒ semmelyik eleje nem tartalmazza a másik kód végét és fordítva, vagy egy másik kódot.

ha a kép max. 256 színt használ ⇒ 1 byte-os tárolás

a) előállítjuk az OP halmazt ⇒ előfordulási gyakoriság

b) P_i és P_j az OP halmaz legkisebb elemei

• hozzunk létre egy új csomópontot az N_{ij}-t, amely P_i és P_j apja a fában

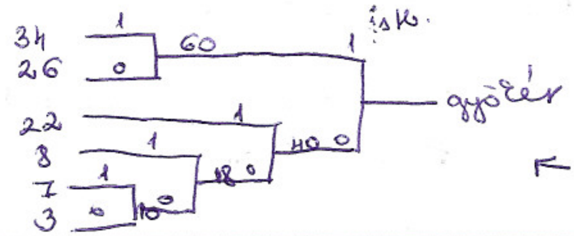
• Az $N_{ij} \Rightarrow P_i$ címkéje legyen 0, az $N_{ij} \Rightarrow P_j$ él címkéje pedig 1.

• Legyen $P(N_{ij}) = P_i + P_j$, a P_i-t és P_j-t töröljük az OP halmazból és $P(N_{ij})$-t felvesszük OP halmazba.

c) ha az OP halmaz 1 elemű, akkor vége, egyébként folytassuk a 2. ponttól.

pl.:

színek	előford. valség	kód
10001010	0,34	11
10100101	0,26	10



← Huffman kód bináris fája

34 kódja	11
26 kódja	10 ... stb.

Szöveges állományokat ezzel kb. a felére lehet tömöríteni.

2) RLE algoritmus (Run-length encoding):

- ismétlődésen alapuló tömörítés

pl.:
input 0000033000344 13 byte
output 5023301324 10 byte
azaz: 5db 0; 2db 3; 3db 0; 1db 3; 2db 4.

- Az első byte tartalmazza, hogy mennyi van, után pedig, hogy miből.

- True Color-nál nem használjuk, mert gyakran hosszabb kódot ad.

- gyors és könnyű a dekódolás

pl.: *.bmp.

- javítása BBS (Background Block Skipping)

abban különbözik az RLE-től, hogy a hátteret nem kódolja s kihagyja a háttérblokkot.

input: 00000 33000 34 4220000000 111112 11
output: $\underbrace{5\,23\,10\,1\,3\,24\,22\,21}$ 511221
ennyiedik pozíciótól kezdve.
hol van háttértől különböző érték

3, LZW (Lempel-Ziv-Welch):

- keresi az ismétlődő karakterláncokat és ezeket azonosító kódra cseréli, ⇒ csak akkor hatékony, ha a karakterláncok ismétlődése nagy.

- kódtábla generálása dinamikus

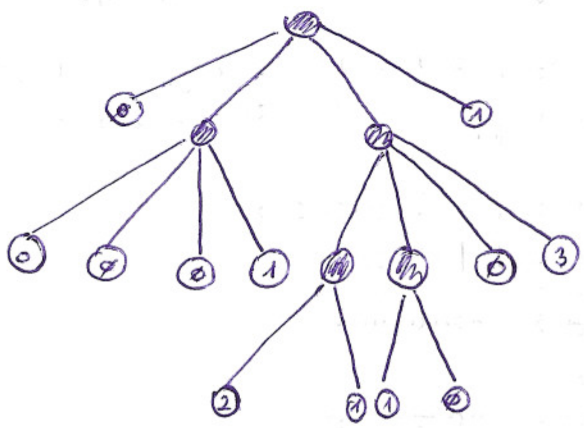
pl. dinamikus kódtábla

1010	j
00	k
000	e

4, Hierarchikus tömörítés:

0	0	0	0	0	0	0	0
0	0	0	0	1	1	0	0
1	1	1	1	2	1	1	0
1	1	1	1	3	3	0	0

· ha egy rész homogén, nem osztjuk tovább, ha nem az, akkor igen.
· Tartozik egy bejárási irány hozzá, ennek megfelelően ezt felépítjük 1 fába.



5, Aritmetikai tömörítés:

- aránylag új
- [0,1) intervallumra leképezzük a különböző tömörítendő szekvenciákat, így különböző hosszúságú törtszámokat kapunk.
- gyakori sorozatokhoz pontosabb törtszámot rendelünk.
- hatásfoka: ✓

6, Adatvesztéses tömörítés:

- az emberi szem nem túl érzékenysége miatt: ha a szín változik, azt érzékeli a szem → a fényességet is, de azt nem, ha 2 pixelből egy hiányzik

  pl.: *.jpg, *.mpeg

- a kép minősége fordított arányban van az adatvesztés mennyiségével.
  pl.: fotó „kockás" lesz, ha túl sokat tömörítjük.

- van olyan, ahol meg lehet mondani, hogy hányszorosan tömörítünk

- ∃ olyan, ahol megadható, hol ne vesszen el adat.

  pl.:

→ lényegtelen

*.gif képet tovább tömöríteni nem érdemes
↳ 256 színből készül

*.jpg → True Colornál

*.png → a böngészők futtatják a programokat ⇒ vírust lehet benne küldeni

kép: mikroprogram, növeg

XI. 16. 9. előadás

Interpoláció, approximáció

$\boxed{\text{5. tétel}}$

adott pontsorozat

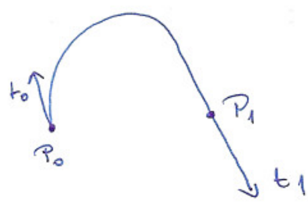
lineáris interpoláció → szakaszokkal kötjük össze → poligont állí-
tunk elő (poli line)

Görbével kötjük össze a pontokat (3-asával vesszük a pontokat → így mindig lehet görbét rajzolni)
↳ nem jó, túl nagy ívek lennének

pub könyvtárban van hozzá anyag!
GRAF

Hermite-interpoláció:

- vegyünk 2 pontot (görbe kezdő- és végpontja)
- adjuk meg az ide húzott érintőket
- görbe paraméteres egyenletét megadjuk



$$\underline{s}(\underline{u}) = \underline{a}_0 u^3 + \underline{a}_1 u^2 + \underline{a}_2 u + \underline{a}_3 \qquad u \in [0,1]$$

u: paraméter

paraméteres harmadfokú egyenlet
síkban ez 2 egyenletet jelent, térben hármat ⇒
→ így lesz belőle pont!

5, Aritmetikai tömörítés:

- aránylag új
- [0;1) intervallumra leképezzük a különböző tömörítendő szekvenciákat, így különböző hosszúságú törtszámokat kapunk.
- gyakori sorozatokhoz pontosabb törtszámot rendelünk.
- hatásfoka:

6, Adatvesztéses tömörítés:

- az emberi nem nem túl érzékenysége miatt: ha a szín változik, azt érzékeli a szem → a fényességet is, de azt nem, ha 2 pixelből egy hiányzik

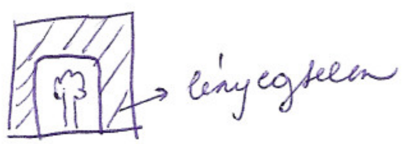
  pl.: *.jpg, *.mpeg

- a kép minősége fordított arányban van az adatvesztés mennyiségével.
  pl.: fotó „kockás" lesz, ha túl sokat tömörítjük.

- van olyan, ahol meg lehet mondani, hogy hányszorosan tömörítünk

- ∃ olyan, ahol megadható, hol ne vesszen el adat.

  pl.:



*.gif képet tovább tömöríteni nem érdemes
↳ 256 színnel készül

*.jpeg → True Colort használ

*.png → a böngészők futtatják a programokat ⇒ vírust lehet benne küldeni

kép: mikroprogram, növeg

XI. 16. 9. előadás

Interpoláció, approximáció

5. tétel

adott pontsorozat

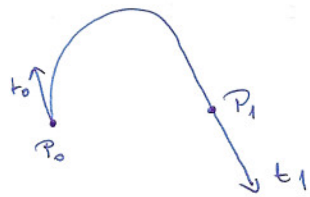
lineáris interpoláció → szakaszokkal kötjük össze → poligont állít kész elő (poli line)

körívekkel kötjük össze a pontokat (3-asával vesszük a pontokat → így mindig lehet körívet rajzolni)
↳ nem jó, kis nagy ívek lesznek

pub könyvtárban van hozzá anyag!
GRAF

Hermite-interpoláció:

- vegyünk 2 pontot (görbe kezdő- és végpontja)
- adjuk meg az ide húzott érintőket
- görbe paraméteres egyenletét megadjuk



$$S(\underline{u}) = \underline{a}_0 u^3 + \underline{a}_1 u^2 + \underline{a}_2 u + \underline{a}_3 \quad u \in [0,1]$$

u: paraméter

paraméteres harmadfokú egyenlet
síkban és 2 egyenletet jelent, térben hármat ⇒
→ így lesz belőle pont!

Ez a görbe térbeli is lehet, lesz egy z tengely (sor) is!

síkban: (x,y) , térben $(x;y;z)$

· h adattal adott a görbe s felírjuk az egyenletet, de nem ismerjük a_0, a_1, a_2, a_3-at ⇒ feladat: határozzuk meg.

· Biztos: P_0, P_1-et tudjuk.

ha u-ba 0-t helyettesítünk s ennek a kezdőpontnak kell lennie $(S(0))$

$S(0) = P_0 = a_3$ → 0-t helyettesítünk

$S(1) = P_1 = a_0 + a_1 + a_2 + a_3$ → 1-et helyettesítünk

$S'(0) = t_0 = a_2$ $\quad S'(0) = t_0$ → derivált, de van x, y, z komponense

$S'(1) = t_1 = 3a_0 + 2a_1 + a_2$

---

t_0 és t_1 az érintők, az első deriváltak ezt adják.

$p_1 = a_0 + a_1 + t_0 + p_0 \Rightarrow a_0 = p_1 - p_0 - t_0 - a_1$

$t_1 = 3a_0 + 2a_1 + t_0$

$(a_0 u^3 + a_1 u^2 + a_2 u + a_3)' = S'(u) =$

M: $\underbrace{= 3a_0 \cdot u^2 + 2a_1 \cdot u + a_2}$

ide kell behelyettesíteni a 0-t és az 1-et.

$t_1 = 3p_1 - 3t_0 - 3\overset{-3p_0}{a_1} + 2a_1 + t_0 \Rightarrow a_1 = 3p_1 - 3p_0 - 2t_0 - t_1$

$\Rightarrow a_0 = 2p_0 - 2p_1 + t_0 + t_1$

<u>A megoldások behelyettesítve:</u>

$S(u) = (2p_0 - 2p_1 + t_0 + t_1) \cdot u^3 + (-3p_0 + 3p_1 - 2t_0 - t_1) \cdot u^2 + t_0 \cdot u + p_0$

Egyenlet rendezése:

$$S(u) = (2u^3 - 3u^2 + 1)\,p_0 + (-2u^3 + 3u^2)\,p_1 + (u^3 - 2u^2 + u)\,t_0 + (u^3 - u^2)\,t_1$$

$$u \in [0, 1]$$

- Az egyenletben szereplő együttható polinomokat Hermite-polinomoknak nevezzük.

- Az u súlyozott Hermite-polinomok jelölése:

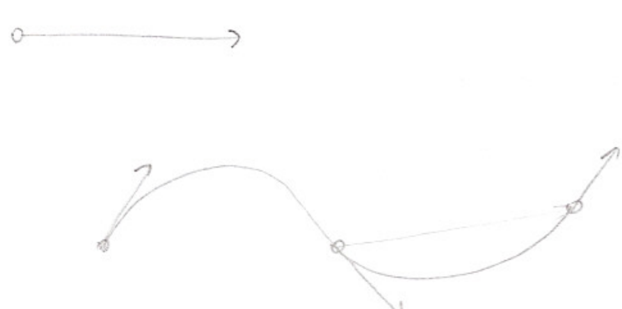
$$H_0 = 2u^3 - 3u^2 + 1$$
$$H_1 = -2u^3 + 3u^2$$
$$H_2 = u^3 - 2u^2 + u$$
$$H_3 = u^3 - u^2$$

- A görbe előnye: mivel harmadfokú, lehet térgörbe, lehet csúcspontja, lehet inflexiós pontja, lehet metszéspontja

Cardinal spline → rugalmas pálca → régen hajók készítéséhez használták

↓

harmadrendű Hermite-görbe (spline)



ftp://.....

Approximáció :

- nem kell, h. a görbe áthaladjon az adott pontokon, csak azt várjuk el, h. megközelítse. (adott sorrendben) →

→ közelítő v. approximáló görbe

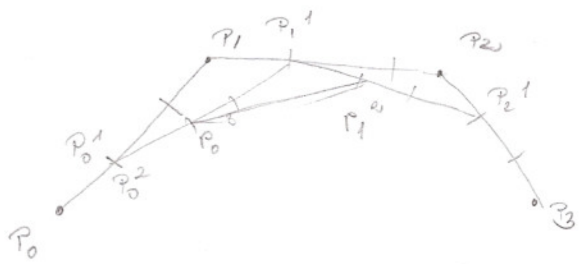
Bezier-görbe :

de Casteljau-algoritmus

$p_i^0 = p_i \quad (i = 0, 1 \dots n)$

$p_i^r(u) = (1-u) \cdot p_i^{r-1}(u) + u\, p_{i+1}^{r-1} \quad (r = 1 \dots n$ és $i = 0, 1 \dots n-r)$

A görbe $u = \frac{1}{3}$ paraméterhez tartozó pontjának megszerkesztése:



kontroll poligon : P_0, P_1, P_2, P_3 kontrolpontok ⇒ meghatározzák a görbe alakját

3 pont által meghatározott Bezier-görbe mindig parabola

$n+1$ db kontrolpontunk van.

Bezier ⇒ Bernstein-polinommal

$s(u) = \sum_{j=0}^{n} p_j \cdot B_j^n(u)$ $\quad u \in [0; 1]$ a Bezier görbe u paraméterhez tartozó pontja, ahol a

$B_j^n(u) = \binom{n}{j} \cdot u^j \cdot (1-u)^{n-j}$ a Bernstein-polinom.

Tulajdonságai:

- el lehet érni önmetszéspontot, csúcspontot

a)- a Bezier-görbe kontrolpontjainak affin transzformációjára invariáns → eltolás, elforgatás ... stb.
elég (csak) a kontrolpontokat transzformálni s újból előállítani a görbét.

b)- Ha $u \in [0,1]$ akkor a Bezier-görbe kontrollpontjainak konvex burkán belül van

 → ilyen nem létezik

c)- Bezier-görbe az első és utolsó kontrollpontokon áthalad

d)- Bezier-görbe szimmetrikus → a kontrollpontok sorrendjét felcserélve ugyanazt a görbét kapjuk.

e)- Ha $u \in [0,1]$ akkor csak görbe kezdő- és végérintője:

$$\underbrace{\frac{d}{du}p(0) = n(p_1 - p_0)}; \quad \underbrace{\frac{d}{du}p(1) = n(p_n - p_{n-1})}$$

c) $B_j^n(u) = \binom{n}{j} u^j \cdot (1-u)^{n-j}$

$n = 3$ (4 kontrollpont van)

$B_0^3(u) = (1-u)^3$

$B_1^3(u) = \binom{3}{1} \cdot u^1 \cdot (1-u)^2 = 3u(1-u)^2$

$B_2^3(u) = \binom{3}{2} \cdot u^2 \cdot (1-u)^1 = 3u^2(1-u)$

$B_3^3(u) = u^3$

$\underline{p}(u) = (1-u)^3 \cdot \underline{p_0} + 3u(1-u)^2 \underline{p_1} + 3u^2(1-u)\underline{p_2} + u^3 \underline{p_3}$

f) Approximáló görbe: a Bezier görbének bármely síkkal legfeljebb annyi metszéspontja van, ahány pontban a sík a kontrollpoligont metszi.

g) belátható, h. n pont esetén n-1-edfokú görbével approximál, a kontrollpontok számának növekedésével, a görbe fokszáma is nő.

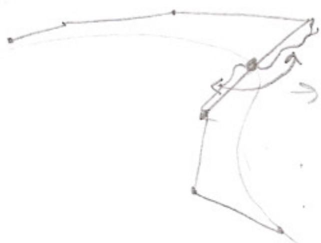
Kapcsolódó Bezier-görbék:

Ha 4 pontnál több van → kapcsolódó görbék jönnek létre

a kapcsolódó pontban sima legyen az átmenet

$\boxed{\text{példaprogram → honlapon!}}$

kapcsolódási pont: az egyik pontba húzott érintő iránya megegyezik a másik-ba húzott érintővel.



→ elsőfokú kapcsolódás

11. előadás

XI.30.

Ponttranszformációk $\boxed{\text{6. tétel}}$

Homogén koordináta:

oka: - egyszerűen lehessen a ~-at tárgyalni
- tudjunk olyan transzformációkat tenni, amiket eddig nem használtunk

a) Síkban: homogén koordinátán egy rendezett számhármast értünk, amely arányossági erejéig van meghatározva, és mindhárom koordináta nem lehet egyszerre 0.

Áttérés: Descartes-koordináta → $P(x,y)$ koordináta
rendezett számpár ↓ → 2. koordináta
első koordináta

Homogén koordináta: $P(x_1, x_2, x_3)$ rendezett számhármas

áttérés:
$x_1 := x$
$x_2 := y$
$x_3 = 1$

a homogén koordináta legyen 1. (ez a gyakoribb)

Pl.: $P(2;3) \rightarrow P(2;3;1) = P(4;6;2)$

Tetszőleges $\lambda \neq 0$ $(\in \mathbb{R})$ számmal megszorozhatók a koordináták, a pont ugyanaz marad.

A sík pontjai és a rendezett számhármasok között kell egy kölcsönösen egyértelmű megfeleltetést létrehozni.

homogén → Descartes

$P(x_1, x_2, x_3)$ → Descartes - koordinátáit akarjuk előállítani

ha $x_3 \neq 0$ $\quad P(x;y)$

$$x = \frac{x_1}{x_3} \quad ; \quad y = \frac{x_2}{x_3}$$

Ha $x_3 = 0$, nem lehet visszalépni.

- Ha $x_3 = 0$ ⇒ végtelen távoli pontoknak hívjuk.

pl: $(1; -3; 0)$

Az egyeneseknek egy végtelen távoli pontja van.

→ egyenlőre illeszkednek végtelen távoli egyenes

b, Térben: Olyan rendezett számnégyes, amely arányossága erejéig van meghatározva, és a koordináták egyszerre nem 0-k.

$P(x_1, x_2, x_3, x_4)$
homogén koordináta.

áttérés, visszatérés: ugyanaz, mint síkban.

A pontok a végtelen távoli síkra illeszkednek.

Ponttranszformációk:

- ponthoz pontot rendel
- kölcsönösen egyértelmű → 2 pontnak nem lehet ua. a képe.

1 : 1 

illeszkedéstartó:

szakasz esetében elég 2 pontot transzformálni, megkapjuk a szakaszt.

a) Síkban:

$\underline{P}' = M \cdot \underline{P}$

↳ transzformáció mátrixa (3x3-as)

$|M| \neq 0$

ha p' és p oszlopvektorral van megadva:

$$\begin{bmatrix} x_1' \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} & & \\ & & \\ & & \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

ponttranszformáció:

homogén formára alakítjuk, megszorozzuk M-mel, és Descartes-f-re átalakítjuk, ha lehet.

1) Egybevágósági transzformációk:

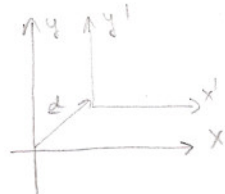
I, mozgási transzformációk

a) helybenhagyás $M = E = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

b) eltolás (transzláció)

$\underline{d}\ (\underline{d}x, \underline{d}y)$ → $x' = x + dx$

$y' = y + dy$



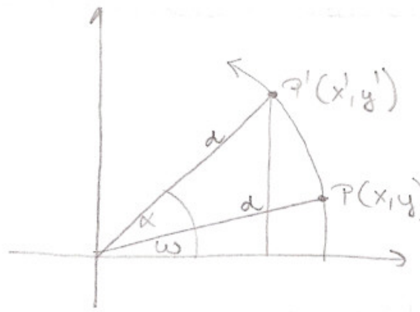
$M = \begin{pmatrix} 1 & 0 & dx \\ 0 & 1 & dy \\ 0 & 0 & 1 \end{pmatrix}$ ✓

c) forgatás (rotate)

Origo körül forgatunk

szög: radiánban α-t $\frac{\pi}{180}$-nal kell beszorozni

letöltések - ben kör.doc



negatív irányba fordatás (2): $(-\alpha)$-t adunk meg

(30) levezetés!!!

$P(x,y) \longrightarrow P'(x',y')$

$$x' = x\cos\alpha - y\cdot\sin\alpha$$
$$y' = x\cdot\sin\alpha + y\cdot\cos\alpha$$

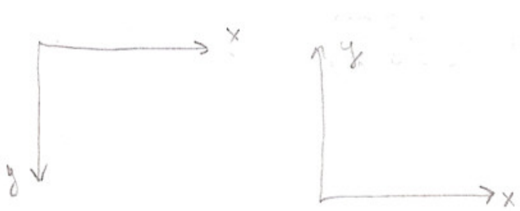
$$M = \begin{pmatrix} \cos\alpha & -\sin\alpha & \emptyset \\ \sin\alpha & \cos\alpha & \emptyset \\ \emptyset & \emptyset & 1 \end{pmatrix}$$

II, tükrözés (reflection)

lehet egyenesre tükrözni pl.: x tengely

$$M_x = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

y - tengely: $M_y = \begin{pmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$



2) Hasonlóság:

x és y koordinátákat ua. a számmal megszorozzuk.

$\lambda \in \mathbb{R}^+$

$$M = \begin{pmatrix} \lambda & 0 & 0 \\ 0 & \lambda & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$\left.\begin{aligned} x' &= \lambda x \\ y' &= \lambda y \end{aligned}\right\} \text{hasonlóság}$$

$\lambda > 0$ és $\lambda < 1$ $\Rightarrow$ kicsinyítés

$\lambda > 1$ $\Rightarrow$ nagyítás

3) Skálázás (scale)

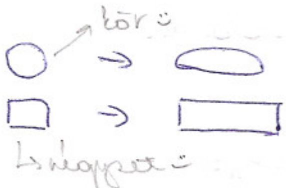
$x' = \lambda_1 x$

$y' = \lambda_2 y$ $\quad \lambda_1, \lambda_2 \in \mathbb{R}^+$

$$M = \begin{pmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

egyik tengely mentén nyújtjuk, vagy a másik mentén összenyomjuk.

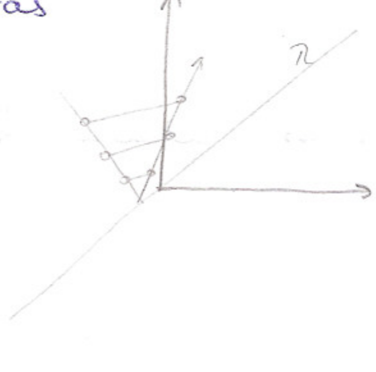
pl.:

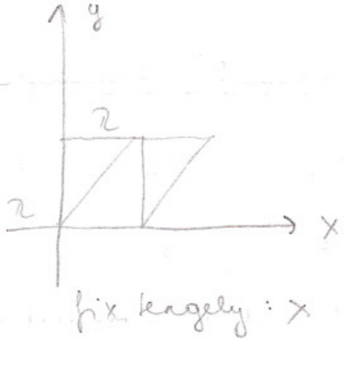


4) általános affin transzformáció

valós ponthoz valós pontot rendel, végtelen távoli-hoz végtelen távolit.

- nyírás





fix tengely: x

$M \Rightarrow$ utolsó sora

$(0; 0; c) \Rightarrow$ valós $\rightarrow$ valós
∞ távoli $\Rightarrow \infty$ távoli

Nem létezik olyan pont, hogy 0 lenne

Descartes-alak:

$$x' = m_{11} x + m_{12} y + dx$$
$$y' = m_{21} x + m_{22} y + dy$$
$\rightarrow$ eltolás

3 pontpár határozza meg az affin transzformációt.

egybeeső pontoknál nem lehet megoldani

5) Projektív transzformáció

valóshoz $\Rightarrow \infty$ távolit és ∞-távolihoz rendelhet valós pontot is.

$|M| \neq \emptyset \Rightarrow$ ez projektív transzformáció

ilyen kapcsolat van a transzformációk között.
(valódi részhalmaza)

4 pontpár kell a meghat-hoz.

<u>Transzformációk szorzása:</u>

Transzformációk szorzása = egymás utáni végrehajtása

$$P' = M_1 \cdot P$$
$$P'' = M_2 \cdot P' = M_2 \cdot (M_1 \cdot P) =$$

mátrixos asszociatív tul. miatt:

$$= \underbrace{(M_2 \cdot M_1)}_{M_3} \cdot P$$

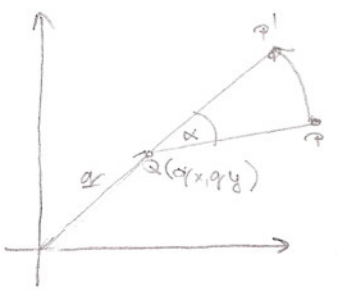
Mátrixos összeszorzása fordított sorrendben ⇒ mátrixok szorzása nem kommutatív.

Mindig balról jobbra szorzunk ⇒ fordított sorrend

ha jobbról balra -"- ⇒ egyenes sorrend

egységelem: helybenhagyás
inverzelem: mátrix inverze $= M^{-1}$

Pl.: tetszőleges pont körüli forgatás α szöggel



mi körül
Rotate At

1) -q-val eltoljuk ($x' = x - q_x$; $y' = y - q_y$)

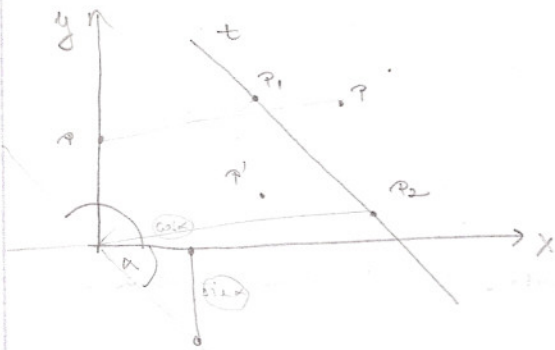
T_q^{-1} ⇒ mátrix (transzláció)

2) R_α ⇒ α szöggel forgatás

3) T_q

$P' = M \cdot P$

$M = T_q \cdot R_\alpha \cdot T_q^{-1}$ Algoritmus 2:2 mátrix szorzása



1) T eltolás
2) R_α
3) R_t^x tükrözés az x tengelyre
4) R_α^{-1}
5) T^{-1}

$\underbrace{\hspace{3em}}$ Tetszőleges tengelyre való tükrözés

egységnyi vektorok esetében: egyik: $\cos\alpha$
másik: $\sin\alpha$

Ponttranszformációk térben: ~~7. tétel~~ →

$\underline{P}' = M \cdot \underline{P}$ $\quad M_{4\times4}$ $\quad |M| \neq \emptyset$

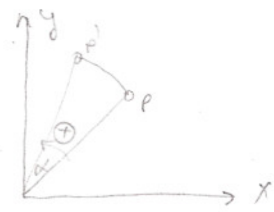
I) Egybevágósági transzformációk
1) mozgási
a) helybenhagyás $\quad M = E$
b) eltolás $\quad \underline{d} = (d_x, d_y, d_z)$

$$M = \begin{pmatrix} 1 & 0 & 0 & d_x \\ 0 & 1 & 0 & d_y \\ 0 & 0 & 1 & d_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

c) forgatás tengely körül α szöggel

M_z : z tengely körül

jobbsodrású



$$M_z = \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 & 0 \\ \sin\alpha & \cos\alpha & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$M_x = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha & 0 \\ 0 & \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$M_y = \begin{pmatrix} \cos\alpha & 0 & \sin\alpha & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\alpha & 0 & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

tükrözni kell a főátlóra. kvadratikus mátrixoknál

d) síkra való tükrözés

koordinátasíkokra tükrözünk

$$M_{xy} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

ekkor minden x y koordináta változatlan, z változik

$$M_{xz} = \begin{pmatrix} 1 & & & \emptyset \\ & -1 & & \\ & & 1 & \\ \emptyset & & & 1 \end{pmatrix} \qquad M_{yz} = \begin{pmatrix} -1 & & & \emptyset \\ & 1 & & \\ & & 1 & \\ \emptyset & & & 1 \end{pmatrix}$$

2) hasonlóság:

$$M = \begin{pmatrix} \lambda & & & \emptyset \\ & \lambda & & \\ & & \lambda & \\ \emptyset & & & 1 \end{pmatrix} \qquad \lambda \in \mathbb{R}^+$$

3) skálázás : scale

$$M = \begin{pmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \lambda_3 & \\ \emptyset & & & 1 \end{pmatrix} \qquad \lambda_1, \lambda_2, \lambda_3 \in \mathbb{R}^+$$

így lehet körből ellipszoidot készíteni

4) Általános affinitás

utolsó sora $(0, 0, 0, c)$

- valóst valósba, ∞ távolit ∞ távoliba visz

$$\left.\begin{aligned} x' &= a_{11}x + a_{12}y + a_{13}\cdot z + dx \\ y' &= a_{21}x + a_{22}y + a_{23}\cdot z + dy \\ z' &= a_{31}x + a_{32}y + a_{33}z + dz \end{aligned}\right\} \text{ált. egyenlet}$$

a ismeretlen van , $dx, dy, dz \rightarrow$ eltolás koordinátái

5) Projektív transzformáció

$\boxed{|M| \neq \emptyset}$

nincs rá megkötés.

Tér leképezése síkra $\boxed{7. \text{tétel}}$

nem kölcsönösen egyértelmű ponttranszformációk

$\underline{P}' = M \cdot \underline{P}$ $\qquad |M| = \emptyset$

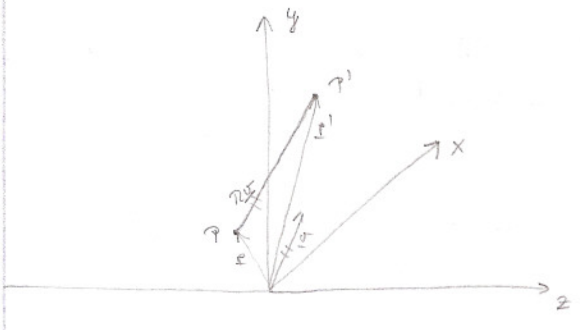
Vetítések $\rightarrow$ 1) párhuzamos
$\rightarrow$ 2) centrális

1)

Adatok: • képsík $[x, y]$ koordinátasík (ide vetítünk, pl: monitor síkja)

• vetítés iránya: $\underline{v}(v_x, v_y, v_z)$

$$P(x, y, z) \xrightarrow{\underline{v}} P'(x', y', z')$$



vesszük a $\underline{v}$-vel ∥ egyenest ⇒ vetítősugár

ezen a vetítősugáron lévő összes pont képe P'

$\underline{p}' = \underline{p} + \lambda \underline{v}$ λ-t kell csak meghatározni

$$x' = x + \lambda v_x$$
$$y' = y + \lambda v_y$$
$$z' = z + \lambda v_z$$

képsík: xy sík – itt minden pont z koordinátája 0.

$$\Downarrow$$
$$\underline{z' = 0} = z + \lambda v_z \Rightarrow \lambda = \frac{-z}{v_z}$$
$$\Downarrow$$
$$\underline{x' = x - z \cdot \frac{v_x}{v_z}}$$
$$\underline{y' = y - z \cdot \frac{v_y}{v_z}}$$

Megjegyzés:

1. át kell térni a másik síkra. pl.: ha nem xy síkra tükrözünk ⇒ át kell térni, de az már koordinátatranszf.

2. ferde vetítés : a z-hez képest

merőleges vetítés : $\underline{v} \perp$ képsíkra
elölnézet, oldalnézet, felülnézet ⇒
innen konstruálható a tárgy

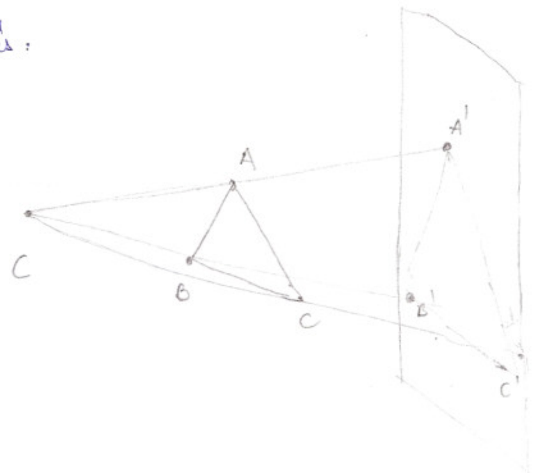
$\underline{v}$ $(0, 0, v_z)$

$x' = x$
$y' = y$
$z' = 0$

úgy kapjuk meg xy koordinátáit, ha elhagyjuk z-t.

kell 4x4-es mátrix, ha megszorozzuk .... megkapjuk a pontokat.

Centrális vetítés:



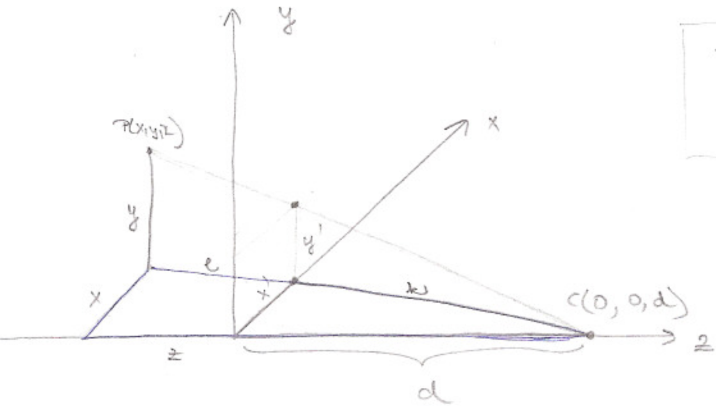
lehetne ⊖ vetíteni (pl.: gömb)
megfordulhat a körbejárás iránya

Spec 1:

<u>Fixcentrumú</u>:

Adatok: képsík: [x, y]
Centrum: z tengely pozitív felén
koordinátái (0, 0, d)
↳ origótól való távolság
térbeli pontok legyenek a síkhoz képest a másik féltérben

P z koordinátái ☺
a sík és a pont külön féltérben van



előbb a $\overline{CP}$ szakaszt rajz. melyeket keresünk egyenes ||-at kihúzunk

$P(x,y,z) \Rightarrow P'(x',y',z')$

$\Delta \sim \Delta$: felírható a megfelelő oldalaik aránya

$$\frac{x'}{x} = \frac{d}{d-z} \Rightarrow \boxed{x' = x \cdot \frac{d}{d-z}}$$

$$\frac{y'}{y} = \frac{k}{l} = \frac{d}{d-z} \Rightarrow \boxed{y' = y \cdot \frac{d}{d-z}}$$

$\boxed{z' = 0.}$ $\Rightarrow$ mert xy síkban vagyunk. (képsík)

Hf: mátrixot előállítani

visszatérés $\Rightarrow$ ha a 4. koord.-ja nem 0.