1.1 Grafikus képformátumok

A grafikus állomány formátumok minden változatát áttekintni szinte lehetetlen feladat. Írásunk adta kereteken belül, csak az általunk legfontosabbnak ítélt, főleg PC DOS környezetben gyakori formátumok kerülnek felsorolásra. A formátum típus név után zárójelben adjuk meg az állomány kiterjesztését.

1.1.1 Állóképek

PCX (*.pcx)

Raszteres típusú Zsoft Corporation fejlesztés.

MS-DOS operációs rendszerkörnyezetben az egyik legelterjedtebb formátum. Eredetileg a Paintbrush programcsomaghoz fejlesztették ki.

TIFF (tag image file format, *.tif)

Raszteres típusú Aldus és Microsoft fejlesztés.

Elsősorban a szkennerrel beolvasott képek belső tárolására használják főleg DTP rendszerekben. Igen elterjedt. Aldus, Microsoft alkalmazásokban és szkennerek szoftvereknél.

GIF (Graphics Interchange format, *.gif)

Raszteres típusú Compuserve Incorporated fejlesztés.

Nagyfelbontású, sokszínű képek tárolására, képernyőn történő megjelenítésekhez használatos. Mivel igen jól tömörített állomány ezért kiválóan alkalmas az Interneten való használatra is.

GEM IMG (*.img)

Raszteres típusú Digital Research fejlesztés.

Főleg a Ventura Publisher applikációhoz fejlesztették ki és jól támogatja a mobilokon megjelenítéket.

TGA (Truevision Targa, *.tga)

Raszteres típusú Truevision Inc. fejlesztés.

Elsősorban a Targa grafikus kártyacsorozat támogatására készült. Mampsag elterjedt szkennereknél és testprogramoknál is.

BMP/DIB (Microsoft Windows Device Independent Bitmap, *.bmp, *.dib)

Raszteres típusú Microsoft fejlesztés.

Alapvető formátum minden windows alatti futó alkalmazás esetén. Nemes más területeken nem támogatott formátum. Windows alatt könnyen kezelhető bitterképes formátum, amely főleg kis méretű képek tárolására alkalmas, mivel RLE tömörítést tartalmaz.

JPEG (Joint Photographic Expert Group, *.jpg)

Főleg fotórealisztikus, valósághű képek tárolására és megjelenítésére használatos tömörített bitterképes formátum. A képeket 8x8 képpontos blokkokra bontja és a blokkok 64 pixelének a bal felső sarokban levő képponthoz való viszonyuk alapján írja le. Ez a tárolás jóval kevesebb helyet igényel, mintha minden képpont jellemzőit külön-külön írnánk le. Kihasználják a tömörítésnél azt is, hogy az emberi szem sokkal érzékenyebb a fényességek változására, mint a színek módosulására, így adatvesztéses tömörítés is alkalmazható. Nagy előnye, hogy ez idáig a legjobban tömörített képformátum, ezért az Interneten a GIF mellett JPEG formátumot használják leginkább. Akár negyedakkora méretű lehet a JPG file, mint egy vele egyenlő minőségű GIF file.

DXF (Drawing Interchange Format, *.dxf)

ASCII formátumú Autodesk Inc. fejlesztés CAD (Számítógéppel Segített Tervezés) programokhoz. Mivel az ASCII szöveg formátum rendkívül lassan dolgozható fel, ezért kifejlesztették egy bináris változatot is. A bináris változat az AutoCAD Release 10 változatával jelent meg, amely körülbelül 25%-kal nagyobb és kb. 5-ször gyorsabb az ASCII változatnál. A DXF formátum kiválóan alkalmas grafikus adatok közvetítésére alkalmazások között, mert a legtöbb számítógépes platformon igen népszerű formátum. Segítségével 3D objektumok, görbék és fontok is leírhatók, ezért elég nehéz feladat DXF formátumú kezelő programot készíteni. Ha ilyenre vállalkoznunk, akkor valójában CAD programot készítünk. Gyakorlati tény, hogy a CAD programok közötti adatcsere szabványává vált ez a formátum.

HP-GL (Hewlett Packard Graphics Language, *.plt)

Hewlett Packard fejlesztésű vektorgrafikus formátum. Eredetileg rajzgépek vezérlésére szolgált, már a lézernyomtatók is vezérelhetők a segítségével. Minden CAD program, a szövegszerkesztők és grafikon készítő programok által is támogatott formátum. Valójában nem képformátum, ha nem egy vonalrajzoló nyelv, amellyel papírmérettől függetlenül tudunk rajzokat megadni. A PCL 5 (Printer Control Language) nyomtatónyelvet ismerő lézer és tintasugaras nyomtatók mind elfogadják a HP-GL/2 változatot. Tehát PCL 5 vezérlő parancsokat használva definiálhatunk a lapon egy olyan téglalap alakú területet, amelyen belül HP-GL/2 kép rajzolható. Másképpen fogalmazva: HP-GL/2 kép beágyazható PCL 5 szövegbe.

PS (Basic PostScript Graphics, *.ps, *.eps)

Vektor és bitterképes formátum, valójában egy lapleíró nyelv, amely hasonlatos a FORTH programnyelvhez. Adobe Systems Inc. fejlesztés eredetileg nyomtatókhoz és egyéb output eszközökhöz tervezték. Az EPS (Encapsulated PostScript) változat megjelenésével különösen alkalmas szines képek és nyomdakész anyagok tárolására. Bár a DTP rendszerekben a PS szabvány, de a nyelv bonyolultsága miatt feldolgozó és PS file-t input-ként elfogadó alkalmazást készíteni meglehetősen nehéz feladat. Adobe's Postcript Language Reference Manual több mint 700 oldal.

WMF (Microsoft Windows Metafile, .wmf)

Eszköz független és jól strukturált file formátum főleg grafikus képek tárolására és Windows alatti futtatható alkalmazások közötti transzportálására. Mivel igen magasszintű struktúrát, és leíratokat benne, ezért jóval tömörebb formátum mint a bitterkép. A WMF file Microsoft Windows grafikus függvények sorozatát tartalmazza, felhasználva grafikus makroutasításokat is. A metafile tartalmaz egy fejlécet (header), melyet néhány rekord követ. A rekordok tartalmaznak egy GDI hívást (Windows graphics device interface), adatokat, függvény azonosítót és paramétereket. Jelenleg a WMF a legtöbb eszközfüggetlen Windows formátum, és minden Microsoft alkalmazás támogatja.

Microsoft Windows Icon (.ico)

Az ikon file 32x32 képpontot ír le általában 4 bit színmélységben. A színleíró táblával együtt 776 byte hosszú az állomány, amelyet a Windows használ ikonként.

Kodak Photo CD (.pcd)

Kodak Precision Color Management System (KPCMS) rendszert használó speciális Kodak fejlesztés, amely igen nagy felbontású és színmélységű képek tárolására alkalmas. A Kodak totó szervizek az általunk elkészített negatív filmekről előhívás után kb. 100 db képet írnak rá egy CD ROM lemezre Kodak CMS Photo CD file formátumban. A legtöbb CD ROM olvasó felismeri ezt a formátumot, így speciális Photo CD lejátszó nélkül is megtekinthetjük képeinket. A KPCMS lehetővé teszi, hogy különböző felbontásokban használjuk a képeket a rendelkezésünkre álló hardver erőforrások függvényében.

A Kodak Photo CD formátumok:

Típus	Felbontás	Szükséges Memória (Mbyte)	Felhasználási terület
Base/16	128 x 192	0.07	Ikon szerűen megjelenő index kép
Base/4	256 x 384	0.28	Elforgatott kép a képernyőn, mivel a Base változat túl nagy méretű a forgatáshoz
Base	512 x 768	1.13	Televízión vagy monitoron való megjelenítés
4 Base	1024 x 1536	4.50	HDTV-n való megjelenítés
16 Base	2048 x 3072	18.00	DTP alkalmazások. Legnagyobb felbontás 35 mm filmről a Photo CD Master Disc számára
64 Base	4096 x 6144	72.00	Elméletileg a legnagyobb felbontás 35 mm-es vagy nagyobb filmről a Pro Photo CD Master Disc

A táblázatban található nagy felbontásokhoz feltétlen meg kell említenünk a Kodak cég speciális scannereit, melyek biztosítják ezeket a minőségeket. A 64 Base felbontáshoz pl. a Professional PCD Film Scanner 4045-öt ajánlják, amely körülbelül 4400 pixels/inch felbontásban képes dolgozni.

1.1.2 Mozgókép formátumok

A mozgókép formátumok rendkívül dinamikusan fejlődnek. Ez a fejlődés egyrészt köszönhető a multimédiás iparág szélesebb körű elterjedésének, másrészt az INTERNET világméretű elterjedésének.

Teljes képernyős lejátszást a legtöbb multimédiás program úgy használja, hogy egy kis ablakban jeleníti meg a videó képet és lehetőség van akár teljes képernyőre váltanunk. Ekkor a felbontás nagy mértékben lecsökken, mert a kis ablak pixelének számát a nagy ablak megfelelő számú pixele kapja meg. Olyan hatású, mintha nagyítón keresztül néznénk a képet.

Teljes mozgású lejátszás (full motion) az emberi szemnek folyamatosnak tűnő mozgó képlejátszás, amelyben 30 képkocka jelenik meg másodpercenként. A gyakorlatban már azonban olyan videófelvételeket is teljes mozgásúnak neveznek, amelyek képváltási frekvenciája csupán 15 képkocka másodpercenként.

Autodesk FLI

Autodesk Animator alkalmazás által használt mozgókép formátum, amely valójában teljesen önálló képek sorozatát tartalmazza.

Grasp GL animáció

Grasp (Gratical System for Presentation) valójában egy slide show, amely a Motortec Industries Inc. fejlesztése. Eredetileg üzleti bemutatókhoz készült, mivel parancs nyelve számtalan vizuális hatást tesz lehetővé pl. áttűnések, forgások, feliratozások stb.

MPEG (Moving Pictures Experts Group)

Egész képernyős, teljes mozgású video lejátszást tesz lehetővé, olcsó 386-4/66-os rendszereken, amelyekben még csak egyszeres sebességű CD-ROM olvasó található. Környezetünkben ez a formátum valhat a mozgóképek szabványává (bejegyzett ISO, International Organization for Standardization szabvány), amelyet a JPEG alapján fejlesztettek ki. MPEG file JPEG képeket tartalmaz, mint vázlat keretet (frames), és a két egymásutáni következő kép közötti különbségeket tárolja a file-ban. Ezzel a megoldással igen nagy mennyiségű képkocka elfér egy nem túl nagy méretű állományban. Mivel elég bonyolult formátumról van szó ezért komoly erőforrásra van szükségünk a zökkenő mentes mozi hatás eléréséhez. Speciális hardver eszközöket fejlesztenek ki az MPEG file-ok lejátszására. Manapság szinte minden grafikus kártyát ellátnak MPEG lejátszó chip-pel, és megjelentek az MPEG formátumban mérve lemezre rögzítő kamerák is. Sőt a mozifilmek is kaphatók úgynevezett Video CD ROM lemezen. Egyenlőre csak kb. 73 perc-nyi film fér el egy CD lemezre, de hamarosan elfogadásra kerülnek új szabványú CD ROM formátumok, amik lehetővé teszik több órás filmek rögzítését video CD-n, video kazetta helyett.

AVI (Audio Video Interleave)

A Widows tehetetlen igen elterjedt formátum, mivel Microsoft fejlesztése a "Video for Windows (VfW)" alkalmazás részeként jelent meg. Hanggal ellátott mozgóképsorozatok lejátszására alkalmazható, ismeretlen ablakban, kb. 15 képkocka másodperc sebességgel. A WIN95 operációs rendszerhez már hozzáadják a VfW-t, így az AVI állományok lejátszhatóak. Az AVI állomány hátránya viszonylag nagy méreteken keresendő.

MOV (QuickTime movies)

Hasonlóan az MPEG formátumhoz ez a formátum is professzionális videó és hang vagy multimédia alkalmazásokhoz készült. Szintén bejegyzett ISO szabvány és bár a Apple Inc. eredetileg Macintosh számítógépekre fejlesztette ki, már létezik nagyszerű PC-s alkalmazása is.

GIF (Graphics Interchange Format)

Az állókép formátuma mellett mozgókép formátuma is van a GIF-nek. Ennek praktikus okai vannak, mivel a CompuServe fejlesztő ezért a saját hálózatán ezt a formátumot támogatja. 1984-ben az UNISYS erősítette a CompuServet, hogy megváltoztatta a Lempel-Ziv-Welch LZW kódolást amelyre épül a GIF is. Az új változatot is megvásárolta a CompuServe és a mozgókép változatban is ezt használja. Ezek után fu olyan

1.2 Vektor- és rasztergrafika összehasonlítása

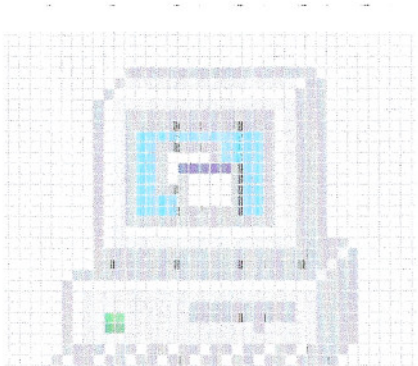
A vektor és raszter (más néven pixel vagy bitmap) grafika két alapvető és lényegesen különböző metodusa a grafikus állományok megjelenítésére és tárolására. A grafikus állományok kezelésére valamelyik módszert vagy esetleg a kettőt együttesen alkalmazzák.

1.2.1 Raszteres grafika

A raszteres alkalmazások igen népszerűek, mert könnyen alkalmazhatóak és használhatóak bármely típusú kép esetén. A raszteres gratika azt jelenti, hogy a kép sorokból (scanline) és azon belül oszlopokba rendezet képpontokból épül fel. Tehát úgy képzelhetjük el, hogy a képet egy rácsszerkezettel osztottuk fel apró elemi részekre. A 1.ábra egy bittérkep kinagyított részletét mutatja, ahol rácspontok egy-egy képpontnak felelnek meg. Végeredményként az adott képpont (pixel) pozicíojat és színet kell tárolnunk. A raszteres grafika alapeleme a képpont. Alkalmazható még a képpontokból álló téglalap (pixmap), melyet főleg betűk, fontok ábrázolásara használnak. Az előző fejezetben láthattuk, hogy nagyon sokféle raszteres állomány létezik. Ezeknek közös jellemzőjük, hogy általában a file elején, a header részben tárolják a színmélységet, színpalettat, tömörítést módot és egyéb a tárolással kapcsolatos információt. A bitterképes állományok egyszerű kezelhetőségük miatt főleg a már kész képek pl. a televíziós képfelvétel és továbbítás területén népszerűek. Meg kell említenünk a vektor grafikával való összehasonlítás miatt, hogy milyen hátrányai vannak a bitterképes grafikának:

- Hatalmas képmeret: Egy nagy felbontású true color kep több megabyte memóriát is felemészthet a tárolaskor, sőt meg többet a megjelenítéskor.
- Komoly hardver erőforrasokat igényel: Az előzőekből is adodik, hogy pl. egy 16 bites és egy 32 bites buszrendszer közötti különbség ebben az esetben drámai különbözőségeket mutat. A megfelelő számítógep egy munkaállomás (workstation) vagy egy igen gyors PC szükséges, hogy elfogadható sebességgel tudjuk feldolgozni a grafikus képeket.
- Alacsony rugalmasság: A képpontoknak nincs kapcsolatuk egymással. Pl. egy képen amely ábrázol egy mezőt és egy házat nem tudjuk beazonositani, hogy melyik pontok a mezőé és melyek a házé, vagyis nem léteznek poligonok és objektumok. Következésképpen, ha módosítani akarjuk a képet pl. megnagyítani vagy látszit megdönteni a házat, akkor ez nem fog sikerülni, vagy igen fáradságos munka az adott terület pontonként való módosítása.
- Felbontási problema: A kép őrzi felbontását, bármely részét nagyítva szembetalálunk magunkat a lépcsőhatas problemájával. Tehát ha például egy körnek látszó alakzatot kinagyítunk, akkor nem egy nagyobb sugarú kört kapunk, hanem egy nagyobb sugarú lépcsős szélű nagy pontokból álló valamit, ami egyre kevesbé emlékeztet minket körre.

Természetesen számtalan kiváló program létezik amellyel a fent említett problémák megoldhatóak (Pl. Adobe PhotoShop, Corel Photo-Paint, stb). Ezeket az alkalmazásokat gyakran fotóretusálo programoknak is nevezik, mivel felhasználhatóságuk ehhez a szakmához áll legközelebb. A Corel Draw programcsomagban található, egy Corel Trace nevű



1. ábra. Nagyító alatt egy ikon

1.2.2 Vektorgrafika

A vektorgrafika azt jelenti, hogy a képet vonalak, alakzatok és görbék sorozataként írjuk le, figyelembe véve, hogy továbbra is lehetnek mintával vagy színnel kitöltött területek. A vektorgrafikát tartalmazó file úgy néz ki mintha egy program file-t vizsgálnánk. Angol parancsszavakat és adatokat tartalmaz ASCII formátumban, ebből következően szabadon szerkeszthető valamilyen editorral. Pl. 2250 mm sugarú kör A4-es (2250 mm és 297 mm középponttal lehet a következő parancs: CIRCLE(100,2250,5000). A

2. ábrán egy HP-GL állomány tartalmát látjuk két hasábban szerkesztve ASCII formátumban, majd a 3. ábra az általa megadott rajzot tartalmazza.

Felismerhető utasítások pl:

- CI = kört rajzol adott sugárral (Circle).
- PU és PD vonalhúzó utasítások (PenUp és PenDown).
- PW = toll szélességet definiálja (PenWith).
- SP = Tollat választ (Select Pen).

1.3 Grafikus képek a WINWORD szövegszerkesztőben és az OLE technika

Az alkalmazások legtöbbje fogadja mind a vektoros mind a raszteres képeket. Az olyan típusú szövegszerkesztő mint pl. Word for Windows képes arra, hogy a dokumentumba beágyazza vagy a dokumentumhoz hozzákapcsolja a raszteres ill. vektoros képeket. Mivel írásunkban főleg PC DOS alkalmazásokkal foglalkozunk, ezért ebben a fejezetben az objektumok beágyazása és kapcsolása alatt a Windows alatt elterjedt OLE technikát fogjuk tárgyalni (Objects Linking and Embedding, OLE). Szövegszerkesztőnek a Word for Windows-t választottuk, mert Magyarországon kétségkívül ez a legelterjedtebb szövegszerkesztő, és célunknak tökéletesen megfelel.

1.3.1 Képek beágyazása (Embedding) és csatolása (Linking)

Ami a számítógépünkben található –program, adatállomány, kép, táblázat, hang, mozgó kép, grafikon– az mind lehet objektum, s ezek az objektumok más objektumokat is létrehozhatnak. Ezentúl csak a képekről fogunk beszélni, bár az elmondottak a többi objektum esetén is szinte teljesen hasonlóak. Amely programot úgy készítenek el, hogy képes legyen kezelni objektumot, az elvileg bármely objektummal tud dolgozni. Persze ez nem ilyen egyszerű, és az OLE technikának nagy ára van, mert ezek a programok nagyon nagy méretűek, s meglehetősen nehézkesen és lassan kezelhetők.

- Amikor hozzákapcsolunk (csatolunk) pl. egy PaintBrush (Rajzoló) képet egy Word szöveghez, akkor a Word-nek megadjuk a BMP állomány nevét és a szövegszerkesztő megjeleníti az állomány tartalmát. A Word használata közben időről időre ellenőrizhetjük a kapcsolatot és frissíthetjük azt a kép esetleges átszerkesztés után, ezzel a módosított kép jelenik már meg a szövegben is.
- Ha beágyazzuk az előbbi képet, akkor a szövegszerkesztő dokumentumában létrejön valami, ami az eredeti BMP kép másolatának nevezhetünk. Nem beszélhetünk állományról mert az operációs rendszer nem tud róla, neve nincs, viszont ha módosítani akarjuk, akkor a Word behívja a Paintbrush-t, s már is szerkeszthetjük a képet.

Lényeges különbség az, hogy kapcsolásnál fizikai kapcsolat létesül a két állomány között, míg beágyazásnál nem. Másképpen fogalmazva, ha a kapcsolt állományt letöröljük vagy megváltoztatjuk fizikai helyét, akkor a dokumentumban csak hibaüzenet jelenik meg a kép helyén. Beágyazásnál fizikailag is bekerül a képállomány másolata, későbbiek során az eredeti állomány le is törölhető, a kép akkor is benne lesz a szövegben, hiszen magában hordozza azt. Kapcsolásnál lényegesen kisebb lesz a dokumentum mérete viszont hordoznunk kell a kapcsolt állományokat is, míg beágyazásnál nagyon nagy méretű állományt kapunk, viszont egyetlen egy állomány tartalmaz mindent. Használhatóságról nehéz véleményt mondani, mert a Word for Windows tartalmaz olyan belső képszerkesztő programot, amely sok grafikus állomány formátumot kezel. Sokszor saját maga önkényesen megszünteti a kapcsolatokat, ha általa ismert állományról van szó. Pl. ha kapcsolunk a szöveghez egy a Word által szerkeszthető kép formátumú állományt, akkor a kép módosításánál nem az eredeti programot hívja be, hanem a saját képszerkesztőjét és megszünteti a kapcsolatot, vagyis beágyazza a képet. A továbbiakhoz tisztáznunk kell két fogalmat:

- Kliens program: a fogadó objektumkezelő programját nevezzük így, a mi esetünkben a Word for Windows szövegszerkesztő.
- Kiszolgáló vagy szerver program: az OLE objektumot létrehozó és kezelő programot nevezzük így, amely speciális objektumkezelő funkciókat tartalmaz. Pl.: PaintBrush.

Amikor egy objektum bekerül egy kapcsolt dokumentumba, akkor összerendelést azzal az alkalmazói, más néven kiszolgáló program létrehozta. Mint tudjuk az összerendelésnek két módja van, a csatolás (linking) és a beágyazás (embedding). Tudnunk kell még a Regisztrációs adatbázis szerkesztése egy olyan speciális eszköz számítógép működésével kapcsolatos információt tartalmaz, beállítások megváltoztatását. Tehát az OLE-val kapcsolatos információkat, vagyis minden újabban telepített program módosíthatja a regisztrációt. A megsértése komoly problémákhoz vezethet, pl. az OLE technika azoknál az állományoknál, ahol eddig működött. Ezek után a grafikus állományainkat beépíteni dokumentumainkba.

1.3.1.1 Meglevő állomány csatolása a dokumentumhoz:

A művelet végrehajtásához általában a következő menüpontot: *Beszúrás•Objektum (Insert•Object)*. *Objektum* helyett speciálisan is választhatjuk a *Kép* almenüt is. Ha nincs *Beszúrás* menü, akkor a *Fájl* menüt. Ezután válasszuk ki a *Létrehozás fájlból* opciót, keressük azt amelyet hozzá akarunk kapcsolni a jelenleg használthoz és jelöljük be a lehetőséget.

1.3.1.2 Meglevő állomány beágyazása a dokumentumba

Ugyanaz a teendő, mint előbb, azzal a különbséggel, hogy a lehetőséget nem jelöljük ki.

1.3.1.3 Még nem létező objektum újként való beágyazása

Hasonlóan járunk el: *Beszúrás•Objektum (Insert•Object)*. *O* Word for Windows-ban választhatjuk a *Kép* almenüt is. Ha nézzük át a *Szerkesztés (Edit)* menüt. Majd az *Új létrehozá* válasszuk ki megadva a kiszolgáló alkalmazás objektumának tí

1.3.1.4 Beágyazott objektum kapcsolttá alakítása:

Kattintsunk rá kétszer a beágyazott objektumra ezzel behoz betöltött alkalmazásban jelöljük ki az objektumot (pl. képet), n megtehetjük pl. a *Szerkesztés•Másolás (Edit•Copy)* menüp vissza a barlová a szövegszerkesztőbe, ezzel visszatérünk bezáródik. Ezek után önállóan futtassuk a kívánt alkalmazást, majd az önállóan működő programba illesszük be a vágólap ta *Szerkesztés•Beillesztés (Edit•Paste)* menüpontokkal, mentsük programból. A WinWordbe visszaterve töröljük a régi objektu tehetjük meg, hogy rákattintunk az objektumra és leütjük a frissen létrehozott állományt a *Meglevő állomány csatolása a d* szerint kapcsoljuk ugyanarra a helyre.

1.3.1.5 Kapcsolt objektum beágyazottá alakítása:

Ez a művelet nagyon egyszerű, csak a kapcsolt állomány *Szerkesztés•Csatolás (Edit•Links)* menüpontok segítségével

a csatolást. Ezután a kívánt helyre beágyazzuk az állományt a *Megjelenő állomány beágyazása a dokumentumba* bejegyzés segítségével.

1.3.1.6 OLE technika hátrányai

Sajnos az OLE igen lassúnak mondható, s nem helytakarékos. Mind a kapcsolásnál, mind a beágyazásnál a forrásobjektum többszörösével nő a kliens állomány mérete. Az OLE használata közben a rendszer erőforrásait igen hamar kimeríti a számtalan szerver alkalmazás betöltése, amit könnyen a rendszer teljes összeomlásához vezethet, különösen Windows 3.1 alatt. Megoldási javaslataink:

- Kerüljük az automatikus frissítéseket. A *Szerkesztés•Csatolás* (*Edit•Links*) menüpontnál használjuk a *Kézi* (*Manual*) frissítést az automatikus helyett, így saját magunk tudjuk vezérelni ezt.
- Hagyjuk nyitva a kiszolgáló alkalmazást. Ha rendszerünk erőforrásai lehetővé teszik az OLE használatát vagyis rendelkezünk elég gyors központi egységgel, legalább 16 Mbyte memóriával és természetesen gyors nagy kapacitású winchesterrel, akkor ne csukjuk be a szerver programokat, mert azok újból és újból való betöltése igen hosszú időt vehet igénybe.
- A kliens program néhány esetben saját maga is tudja olvasni a szerver programok állományait. Ilyen esetekben ne használjuk az OLE-t. Pl. a Word for Windows esetén a *Beszúrás•Kép* (*Insert•Picture*) használatával a grafikus állományokat közvetlenül is olvashatjuk. Különösen jó eredményt érhetünk el bittérképek (pl. BMP) beszúrása esetén. Ez az eset akkor fordul elő, amikor a vágólap tartalma egy bittérkép, és ilyenkor a *Szerkesztés•Irányított beillesztés* (*Edit•Paste*) menüpontokat választjuk. Ekkor lehetőségünk van a Winword számára kezelhetőbb *Kép* megoldást választani, ami kevesebb erőforrást igényel, kisebb lesz az állomány mérete és szebb nyomtatási képet is ad.

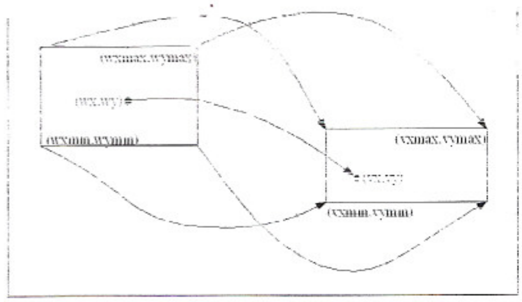
Speciális esetekben előfordul, hogy a koordináta-rendszer kiinduló pontja a képernyő bal alsó sarkában van, vagy a teknőcgrafika esetén középen található. Az általunk megadott raszteres algoritmusok a bal alsó kezdőpontot használják, mintha a matematika füzetünkben dolgoznánk. Ez nem okoz nehézséget, mert egy egyszerű ponttranszformációval áttérhetünk a ?? ábrán bemutatott, egyébként a monitorok kezelésének szokásos rendszerére. A *Transf* eljárás pl. az origóban levő alakzatot transzformálja a ?? ábra által használt koordináta-rendszer esetén a képernyő közepére.

```
Procedure Transf (var x,y : integer);
  Begin
    y := y - (GetMaxY div 2);
    y := -y;
    x := x + (GetMaxX div 2);
  End;
```

Néhány újabb fogalmat is tisztáznunk kell. A ScanLine vagy letapogató vonal az összes azonos y koordinátájú pont halmaza, tehát a képernyő egy sorának felel meg. A raszterpontok fényerőssége az output eszközünk minőségétől függően változtatható. A legtöbb modern grafikus kártya beépített bonyolultabb algoritmusokat is tartalmaz, nem csak a pont színe állítható velük.

4.1 Window-viewport transzformációk

A legtöbb grafikus problémát általában Descartes koordináta-rendszerben oldunk meg, tehát derékszögű úgynevezett világkoordináta-rendszerben adjuk meg az algoritmust. A világkoordináta-rendszer speciális, ezért gyakran transzformálni kell a periféria koordináta-rendszerébe. Általában a világ koordináta-rendszer derékszögű ablakát (Window) transzformáljuk az output rendszer képtartományába (Viewport). Gyors algoritmusra van szükségünk, mivel sok pont kezelésével nagy mennyiségű ismétlődő számítást kell elvégeznünk.



19. ábra

A geometriai input és output adatokat koordináta-rendszerek közötti leképezéseknek vetjük alá. A következő koordináta-rendszereket különböztetjük meg:

- világkoordináta-rendszer (World Coordinates, WC) a felhasználó ebben adja meg illetve ebben kapja vissza a geometriai információkat.
- normalizált eszközkoordináta-rendszer (Normalized Device Coordinates, NDC), a különböző grafikus munkahelyek számára egységes koordináta-rendszer.
- eszközkoordináta-rendszere (Device Coordinates, DC), a munkahely hardverének megfelelő specifikus koordináta-rendszer.

A ?? ábra alapján könnyen előállíthatunk egy olyan függvényt, amely világkoordináta-rendszerben lévő pontot átviszi a NDC koordináta-rendszerbe. A függvénynek rendelkeznie kell a következő tulajdonságokkal:

A) a világ koordináta-rendszer derékszögű ablakának (Window) határvonalpontjait az output rendszer képtartományának (Viewport) határvonalpontjaiba transzformálja.

Tehát a következő megfeleltetés igaz:

$$(wx\min, wy\min, wx\max, wy\max) \rightarrow (vx\min, vy\min, vx\max, vy\max)$$

B) Egyenestartó leképezés, vagyis egyenes képe is egyenes lesz.

C) Az x és y tengelyek a két rendszerben páronként párhuzamosak egymással.

Ebből következik, hogy a leképezés lineáris és a ?? ábra alapján a következő összefüggés vezethető le:

$$(1) \qquad wx = wx\min + a(wx\max - wx\min) \qquad \text{ahol } (0 \le a \le 1)$$

és igaz, hogy

$$(2) \qquad vx = vx\min + a(vx\max - vx\min)$$

(1)-ből adódik, hogy

$$a = \frac{wx - wx\min}{wx\max - wx\min}$$

visszahelyettesítve (2)-be

$$vx = vx\min + \frac{wx - wx\min}{wx\max - wx\min}(vx\max - vx\min) =$$

$$= wx\frac{vx\max - vx\min}{wx\max - wx\min} + vx\min - wx\min\frac{vx\max - vx\min}{wx\max - wx\min}$$

Új jelöléseket bevezetve: $dvx = vx\max - vx\min$, $dwx = wx\max - wx\min$ és hasonlóan számítható *dvy* és *dwy*:

$$vx = \frac{dvx}{dwx} \cdot wx + \left(vx\min - wx\min\frac{dvx}{dwx}\right)$$

ami lineáris leképezés.

A másik koordinátára hasonló kifejezést kapunk:

$$vy = \frac{dvy}{dwy} \cdot wy + \left(vy\min - wy\min\frac{dvy}{dwy}\right)$$

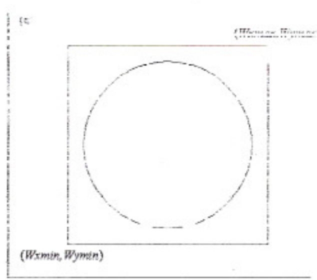
Tehát végeredményként azt kapjuk, hogy

$$vx = A \cdot wx + B$$

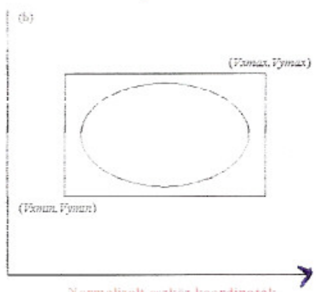
$$vy = C \cdot wy + D$$

Mivel *A,B,C,D* a leképezés során függetlenek a (wx, wy) transzformálandó ponttól ezért ezeket az értékeket csak egyszer kell kiszámítani. Ha $a < 0$ vagy $a > 1$ akkor a transzformálandó pont kívül esik az ablakon ezért nem kell vele foglalkozni.

4.2 Uniform eszköz transzformáció



Világ koordináták



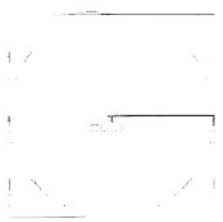
Normalizált eszköz koordináták

20. ábra

A ?? ábra egy Window-Viewport transzformációt illusztrál, ahol a torzítási tényezők (aspect ratio) nem egyeznek meg. Torzítási tényezőn a magasság és a szélesség arányát értjük. Ilyen esetekben a kör képe ellipszis lesz. A leképezési függvényben szereplő (*dvx/dwx*) és (*dvy/dwy*) nem egyenlő. Ez elfogadható, de a programozónak nagyon ügyelnie kell, hogy ne feledkezzen meg a torzulásról. Az *uniform transzformációknál* követelmény az előbbi hányadosok egyenlősége, ie nevezhetjük. Az uniform transzformáció valójában hasonlósági transzformáció is, mivel hasonló alakzatok keletkeznek.

Tekintsünk egy példát erre a problémára Turbo Pascal programozási környezetben.

Készítsük el a 28. ábrának megfelelő rajzot, amelyen egy kört érintő négyzetet láthatunk. A feladat megoldásánál vegyük figyelembe a torzítási tényezőt.



21. ábra

Az algoritmust világ koordináta-rendszerben triviális, de DC-ben a különböző felbontású grafikus üzemmódok esetén figyelembe kell venni a torzítási tényezőt.

Bemenő adatok : x, y, r

```
Procedure Érintő_Négyzet(x,y,r:integer);
Var
      Xasp,Yasp: Word;
 Begin
      GetAspectRatio(Xasp,Yasp);
      Rectangle(x-r,y-Round(r*(Xasp/Yasp)),x+r,y+(Round(r*(Xasp/Yasp)));
      Circle(x,y,r);
 End;
```

Az eljárásban a *Round* függvény paramétereként megadott (*Xasp/Yasp*) hányadost érdemes zárójelbe tenni, mert így elkerülhetjük a egész típusok szorzása során fellépő túlcsordulást. A GetAspectRatio eljárás visszaadja a grafikus meghajtó adott üzemmódjához tartozó torzítási tényezőket, vagyis a téglalap alakú képpont vízszintes ill. függőleges oldalának arányát. Néhány példa ezen értékekre:

Üzemmód	Felbontás	Xasp	Yasp
CgaC0	320x200	8333	10000
VgaLo	640x200	4500	10000
VgaMed	640x350	7750	10000
VgaHi	640x480	10000	10000

A táblázatból kiderül, hogy 640x480-as felbontásban nincs torzítás, ezért ebben a felbontásban nem kell figyelembe vennünk a torzítást. A hiba csak akkor következik be, ha más felbontású monitoron is szeretnénk használni a programot.

4.3 Szakasz rajzolás

Az egyenes szakasz rajzoló algoritmus megadásához elengedhetetlen néhány matemati... fogalom használata. Bresenham algoritmusának levezetése első ránézésre bonyolultn... tűnik, viszont a végső eredmény könnyen programozható.

Néhány kritériumnak eleget kell tennie a számítógéppel húzott szakaszoknak:

A) A szakaszunk „egyenes vonalnak" kell látszania. Raszteres grafika esetén ponto... csak akkor lenne címezhető a szakasz egy pontja, ha 45°-os szögben vagy vízszin... vagy függőleges irányban halad. Egyéb esetben nekünk kell döntenünk, hogy az ide... egyenestől eltérően melyik raszterpontot válasszuk.

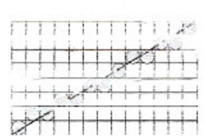
B) Pontosnak kell lennie. Ha nem pontosan a kezdőpontból indul és a végpontig ta... szakasz, akkor poligonvonal rajzolása esetén halmozódóan kicsiny hézagok láthatók... egyik szakasz végpontja és a másik szakasz kezdőpontja között.

C) Fedettsége legyen állandó függetlenül a vonal hosszától és hajlásszögétől. Vo... fedettséget fényességként érzékeljük amikor sötét háttér mellett rajzoljuk a szakaszt... a kritérium csak a már említett kedvező esetekben (45°, 0° és 90°) érhető... tökéletesen. Kis felbontású képernyőnél (pl. 320x200) egyik legismertebb szépséghib... ferdevonalaknál fellépő lépcsőhatás. Ilyen esetekben a vonal fedettségének... megváltoztatása vezethet eredményhez. Tehát a vonal és a környező pon... világosságának változtatásával simán futó vonal látását érhetjük el. Elmosódott ké... kapunk, de a zavaró szögletesség eltüntethető.

D) Gyors legyen. Mivel igen gyakran használják ezeket, ezért a gyorsaság alapvető elv... az algoritmussal szemben.

4.3.1 Egyszerű növekményes módszer

Legegyszerűbbnek tűnő megoldás, ha az egyenes $y = mx + b$ alakú egyenletéből indul... ki. Ekkor rögtön megkapjuk az egyenes y tengelypontját $(0,b)$. Ezek után az $m = dy/...$ segítségével meghatározzuk, hogy mennyit lépjünk balra és mennyit felfelé, hogy megkap... az egyenes következő pontját. Hasonlóan járunk el, mint ahogy azt általános iskola... tanultuk a matematika órán. Észrevétlenül egy növekményes algoritmust kaptunk. ... algoritmus nem ad szép képet, hiszen nekünk kell kerekítenünk a kapott (x,y) értékek... mert az ideális egyenes helyett csak közelítő raszter pontokat kaphatunk. Ezen... hiányosságon próbál segíteni az egyszerű DDA (digitális differenciál-analizátor), mégpe... úgy, hogy a *dx*-et és *dy*-t kedvezőbben választja meg. A nagyobb elmozdulás irányá... legyen a növekmény egy, tehát Max(*dx*,*dy*)=1. A 22. ábra egyszerű DDA-val raj... szakaszt ... mu...



22. ábra

Ha az egyik növekmény 1, akkor ez lehetővé teszi, hogy az egyik növekménynél közönséges számlálót használjunk:

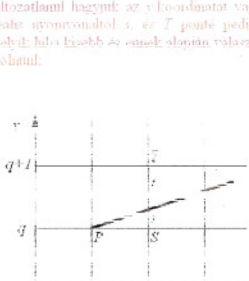
```
Procedure DDA (x1,y1,x2,y2,color:integer);
  Var
      hossz,i:integer;
      x,y,xn,yn:real;
  Begin
      hossz:= abs(x2-x1);
      If abs(y2-y1)> hossz Then hossz:= abs(y2-y1);
      xn:=(x2-x1)/hossz;
      yn:=(y2-y1)/hossz;
      x:= x1+0.5; y:= y1+0.5;
      For i:= 1 to hossz do
         Begin
           PutPixel(Round(x),Round(y),color);
           x:= x+xn;
           y:= y+yn;
         End;
  End;
```

4.3.2 Bresenham-algoritmus

Bresenham, J.E. kifejlesztett egy klasszikus algoritmust, amely kijavítja az egyszerű DDA hibáit és alkalmas hardver implementációra is. Később Midpoint (középpont) technikát fejlesztettek ki, amelyre épülő algoritmust először Pitteway, M.L.W. publikált. A középpont technika kimutathatóan ugyanazokat a pontokat produkálja egyenes szakaszok esetén, mint az eredeti Bresenham-algoritmus. Most az eredeti algoritmust ismertetjük, és a körrajzoló bemutatásánál térünk át a Midpoint algoritmusra.

Induljunk ki ismét az egyenes $y = mx + b$ alakú egyenletéből. A megrajzolandó szakasz kezdő és végpontjával adott. Tehát készítenünk kell egy *Line*(*x1,y1,x2,y2*) eljárást. Legyen $dx = x2 - x1$ és $dy = y2 - y1$. Ekkor $m = dy/dx$. Tegyük fel, hogy az egyenes meredekségére a következő $0 \le m \le 1$ teljesül az egyenesnek a vízszintes tengellyel bezárt ...

irányú elmozdulás nagyobb mint az y tengelyű elmozdulás. A növekményes algoritmus ... érték eggyel való növelésekor kiszámolja, hogy mennyivel kell változtatni az y koordiná... Az egyenes ideális nyomvonala valahol a két y koordináta között halad. Az a feladatu... hogy a két lehetséges y koordináta közül azt válasszuk ki, amelyiknek az ideális egyene... mert eltérése a kisebb. A 23. ábra négyzetháló kereszteződési pontja egy képernyő pixel... felel meg. A *P*(*r,q*) ponton áthalad a szakasz. Tegyük fel, hogy ezt a pontot az (*i*−*1*)-e... lépésben jelöltük ki. Az *i*-edik lépésben az x koordinátát eggyel növeljük. Tehát ... x koordinátája *r*+1 lesz, és megvizsgáljuk, hogy a lehetséges *S* és *T* pont közül melyi... válasszuk, vagyis változatlanul hagyjuk az y koordinátát vagy növeljük eggyel. Legye... pont eltérése az ideális nyomvonaltól *s*, és *T* ponté pedig *t*. A Bresenham-algoritm... megállapítja, hogy melyik hiba kisebb és ennek alapján választ. A 23. ábrán jól látható, h... az *S* pontot kell bejelölnünk.



23. ábra

A számítás egyszerűsége miatt tételezzük fel, hogy $y = mx + b$ egyenes áthalad az orig... ekkor $b = 0$. Felhasználva ezt és a már meghatározott meredekséget az egyenes egyenlet... következő lesz: $y = \frac{dy}{dx}x$

A 23. ábra alapján kiszámítható:

$$s = \frac{dy}{dx}(r+1) - q \qquad t = q + 1 - \frac{dy}{dx}(r+1)$$

Ebből adódik:

$$s - t = 2\frac{dy}{dx}(r+1) - 2q - 1$$

Az $s - t$ pontos értékére nincs szükségünk, mert az előjele alapján eldönthető, hogy mel... pontot válasszuk. Pl. ha $s - t > 0$ akkor $s > t$ és a *T* pontot kell választanunk, ellenk... esetben pedig az *S* pontot.

Tételezzük fel hogy $dx > 0$ és szorozzuk meg az egyenlőtlen részt:

(1) $$dx \cdot (s - t) = 2(r \cdot dy - q \cdot dx) + 2dy - dx$$

Az egyszerűség kedvéért vezessük be a $d(i)$ jelölést, ahol az *i* index a lépésszámot jelöli ...

$$d(i) = dx \cdot (s - t)$$

```
program grafika;
uses crt,graph;
var xx1,yy1,xx2,yy2,color:integer;
{*****************************}
function graf:boolean;
var gv, gm:integer;
begin
   graf:= true;
   detectgraph (gv,gm);
   { vagy gv:=detect;}
   initgraph(gv, gm, '');
   if graphresult <> grOk then
      begin
      writeln('Hiba az inicializ l sakor!: ',grapherrormsg(graphresult));
      graf:=false;
      end;
end;
{*****************************}
procedure DDA(x1,y1,x2,y2,color:integer);
var hossz,i:integer;
    x,y,xn,yn:real;
begin
hossz:=abs(x2-x1);
if hossz< abs(y2-y1) then
   hossz:=abs(y2-y1);
xn:=(x2-x1)/hossz;
yn:=(y2-y1)/hossz;
x:=x1;
y:=y1;
for i:=1 to hossz do
begin
     putpixel(round(x),round(y),color);
     y:=y+yn;
     x:=x+xn;
end;
end;
{*****************************}


begin
clrscr;
if graf then
   begin
   DDA(20,20,200,200,white);
   setcolor(white);
   line(20,50,200,230);
   end;
readln;
closegraph;
end.
```

circle : kör
ellipse : ellipszis
fillellipse : kitöltött ellipszis
bar : kitöltött téglalap
setfillstyle : kitöltésmód beáll.
setcolor : rajzszín
setbkcolor : háttérszín
line : vonal
getx, gety : x,y koord. lekérd.
cleardevice : képernyő törlése
getpixel : képpont színének a lekérdezése
putpixel : pont rajzolása
outtext(x,y) : adott szöveg kiírása