Számítógépes grafika

elméleti tananyag ellenőrző kérdései, 2015/16. I. szemeszterétől

Tartalom

2. fejezet

1. Hogyan határozzák meg a számítógépes grafika fogalmát? Mit értünk generatív számítógépes grafika alatt? Mi a számítógépes grafika tárgya?

Legáltalánosabban a **számítógépes grafika** (computer graphics) alatt a számítógép és a grafikus perifériák között megvalósított adatkonvertálási módszereket és eljárásokat értjük.
A **generatív számítógépes grafika** ezek szerint a képi információ tartalmára vonatkozó képleírási adatok (és nem a kép) alapján, algoritmusokkal állít elő pl. a monitor képernyőjén megjeleníthető képeket.
A **számítógépes grafika tárgyát** – az előbbiekre is figyelemmel – a következőképpen határozhatjuk meg: a számítógépes grafika alatt a két- (2D) és három- (3D) dimenziós grafikus objektumok számítógépes generálását, tárolását, feldolgozását és megjelenítését értjük.

2. Mit értünk képfeldolgozás alatt? Mit jelent az alakfelismerés?

A **képfeldolgozás** mindazon számítógépes eljárások és módszerek összességét jelenti, amelyekkel a számítógépen tárolt képek minőségét valamilyen szempont szerint javítani lehet (pl. élek kiemelése), és ezáltal továbbfeldolgozásra alkalmasabbá válnak.
A számítógépes **alakfelismeréssel** a raszteres képeken lévő grafikus objektumok azonosítását végezzük el. Ezekkel – a legtöbb esetben bonyolult matematikai eljárásokkal – a raszteres képből kinyerjük a képleíráshoz szükséges lényegi információkat.

3. Mit jelent a grafikus objektumok modellezése?

A számítógépes grafikában és a képfeldolgozás során nem a valódi objektumokat, hanem azok egy modelljét dolgozzuk fel. A modellalkotás során megpróbáljuk a grafikus objektum lényegi jellemzőit megragadni, és az így absztrakcióval képzett számítógépes modellt algoritmusokkal dolgozzuk fel.

4. Sorolja fel, milyen elemekből épül fel egy grafikus rendszer! Mi a szerepe a szabványos csatolóknak (interface) a grafikus rendszerekben? Mit jelent az API, és mi a feladata a grafikus rendszerekben? Mi a feladata a hardvercsatolónak (device interface), és milyen rendszerelemek tartoznak e csatolóhoz? Mit jelent a GUI, és mikor vált szabvánnyá? Mit nevezünk primitívnek?

Egy grafikus rendszer három fő részből tevődik össze:

- az alkalmazás-specifikus felhasználói programrendszerből (CAD-rendszer, térképészeti rendszer stb.),
- a szabványosított grafikus programcsomagból (pl. DirectX) és
- a grafikus hardverből (pl. grafikus kártya és monitor)

Az egyes rendszerrészeket szabványos **interfészek** vagy **csatolók** kapcsolják össze, ami által az egyes rendszeregységeket (pl. újabb verziójú szoftver vagy fejlettebb grafikus periféria megjelenése esetén) úgy tudjuk lecserélni, hogy a többi rendszerkomponenst nem kell megváltoztatni.
A grafikus programcsomag és a felhasználói programrendszer között helyezkedik el a felhasználói programcsatoló vagy **API** (Application Program Interface). Ez általában azt is biztosítja, hogy a grafikus programcsomaggal különböző programnyelven tarthassanak kapcsolatot a felhasználói programok (language binding).
A grafikus programcsomag egy-egy konkrét grafikus perifériával a hardvercsatolón (**device interface**) keresztül tartja a kapcsolatot. Ide tartoznak a különféle eszközmeghajtók (device driver) és a BIOS megfelelő része (video ROM-BIOS).
A grafikus rendszerrel a felhasználó egy szabványos grafikus felületen tarthat kapcsolatot. Ez a **GUI** (Graphical User Interface), mely az 1980-as évek második felében vált szabvánnyá (X11 és X-Windows 1987).
A grafikus programcsomag a képet saját rendszerében kezelt grafikus objektumokból állítja elő. A legkisebb, a grafikus programcsomag által már további részekre nem bontható elemi grafikus objektumokat **primitíveknek** nevezzük. A grafikus programcsomag ezekből a primitívekből állítja elő a komplexebb objektumokat.
Példák primitívekre:

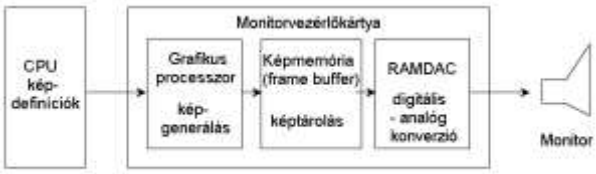
- 3D vektorgrafikában egy térbeli egyenesszakasz,
- rasztergrafikában egy kör.

5. Milyen transzformációk szükségesek a modelladatok képernyőn történő megjelenítéséhez?

A képernyőn történő megjelenítéshez a felhasználói programrendszer modelladatait először a grafikus programcsomag primitívjeiből felépített objektumokká, majd raszteres képpé kell transzformálni.

6. Mit értünk pixel alatt? Mi a monitorvezérlő kártya részegységeinek feladata raszteres képek megjelenítése során? Hogy állítja elő a színes monitor a pixeleket?

A **pixel** (picture element) magyarul a kép elemi, ezért tovább fel nem bontható részét jelenti.



A hardvaregységek közötti feladatmegosztás a raszteres kép megjelenítésében

A folyadékkristályos (**LCD** = Liquid Crystal Display) monitorok esetében a képpontoknak megfelelő piros, zöld és kék színeket egyszerű szűrőkkel állítják elő. A kijelző minden cellájára alkalmazott feszültség külön vezérelhető, amelynek hatására a folyadékkristály-molekulák egy bizonyos irányba állnak be, és így több vagy kevesebb fényt eresztenek át az állandó háttérvilágításból.

A pixelek tulajdonképpen a képernyőn három darab, az alapszíneknek (piros, zöld, kék) megfelelő részpontból állnak, amelyet a **CRT** (Cathode Ray Tube) monitor esetében három elektronsugár „gyújt fel" a monitor foszforrétegén.

7. Jellemezze a színtereket mint vektortereket! Mit értünk additív színkeverés alatt? Mit értünk szubsztraktív színkeverés alatt? Mi az RGB színtér? Mi az CMY színtér? Mi jellemzi a CMYK színteret? Adja meg az RGB és CMY színterek közötti átszámítás szabályait! Mi jellemzi a HSB színteret? Mi a True Color színkódolás?

A színek egy háromdimenziós matematikai struktúrát alkotnak, azaz megfeleltethetők egy 3D-s vektortér vektorainak. Ezt a vektorteret **színtérnek** hívjuk, a tér egyes vektorait színvegyértéknek.

Az összeadó vagy **additív színkeverésnél** a vörös, zöld és kék alapszínekből vett meghatározott mennyiségeket adunk össze, és így kapjuk a különböző színárnyalatokat. Ezzel az úgynevezett elsődleges fényforrások színeit tudjuk előállítani.

A kivonó vagy **szubsztraktív színkeverésnél** az alapszínek komplementereiből (ciánkék, bíborvörös, sárga) állítjuk elő a színeket. Ezzel lehet modellezni a különböző tárgyak által visszavert fényt.

Az **RGB** színtér a vörös, a zöld, a kék (Red, Green, Blue) alapszínekből kikeverhető színeket tartalmazza, az additív színkeverés modellezéséhez használjuk.

A **CMY** színtér a ciánkék, a bíborvörös, a sárga (Cyan, Magenta, Yellow) alapszínekből kikeverhető színeket tartalmazza, a szubtraktív színkeverés modellezéséhez használjuk.

A **CMYK** színtér megegyezik a CMY színtérrel, azzal a különbséggel, hogy a CMY színtér alapszíneihez még hozzáadjuk a „tiszta" fekete színt is. Ennek az az oka, hogy a CMY alapszínek keverésével csak sötétszürke színt tudunk előállítani, és a nyomdatechnikában a teljesen fekete színre is szükségünk van.

```
[c, m, y] = [1, 1, 1] – [r, g, b]
[r, g, b] = [1, 1, 1] – [c, m, y]
```

Egy ilyen a **HSB színtér** , amelyben az RGB alapszínek mellett a színek előállításához a színtelítettség és a megvilágítás erősség értékeit is felhasználhatjuk.

Napjainkban legelterjedtebb a **True Color** (igazi színek) háromcsatornás színkódolás, amikor a három RGB alapszín intenzitását 3x8 = 24 biten kódolják, ami 224, azaz kb. 16 millió színárnyalat megjelenítését teszi lehetővé. Az egyes alapszínek intenzitását a felhasználó ebben az esetben egy 0 és 255 közötti számérték beállításával adhatja meg

8. Definiálja a rasztergrafikus rendszert! Mit nevezünk vektorgrafikus rendszernek? Melyek a vektorgrafika és rasztergrafika közötti legfontosabb különbségek?

A számítógépes grafikában azokat a raszteres, azaz képpontokból álló képet generáló és feldolgozó rendszereket, amelyeknél a képi információ csak képenként kereshető vissza, és a kép tartalma csak a teljes kép felülírásával módosítható, **rasztergrafikus rendszereknek** nevezzük.

A számítógépes grafikában azokat a rendszereket, amelyek a grafikus objektumokat egy lebegőpontos világkoordináta-rendszerben modellezik, **vektorgrafikus rendszereknek** nevezzük.

- A vektorgrafika egy 3D-s (korábbi változatai 2D-s) lebegőpontos világkoordináta-rendszert használ, ezáltal lehetővé teszi a geometriai pontosságú szerkesztést és transzformációkat. Így sokkal alkalmasabb a rasztergrafikánál a mérnöki és tudományos munka támogatására.
- A vektorgrafika absztrakt modelltérbeli tárgyakkal dolgozik. Ezek önálló objektumok (entitások), amelyekkel műveleteket lehet végezni a képernyőn való megjelenítéstől függetlenül is. A felhasználói felület, a grafikus programcsomagok és a megjelenítő hardver szabványosítása ugyanakkor lényegében bármely elterjedt számítógép-konfiguráción lehetővé teszi a modelltérbeli vektorgrafikus objektumok raszteres képen való megjelenítését akár több nézőpontból (kameraállásból) is. Ezzel szemben egy rasztergrafikus képet – lényegét tekintve – csak a kép felülírásával tudjuk módosítani.
- A vektorgrafikában a grafikus objektumokat adatbázisban tárolják, amely lehetővé teszi az egyes testek, tárgyak modelljeinek egyedi visszakeresését, és az ezek közötti kapcsolatok rögzítését és kimutathatóságát. A rasztergrafikában viszont nincs lehetőség az egyes képeken belüli grafikus rajzelemek önálló visszakeresésére és kezelésére.

9. Soroljon fel minél többet a számítógépes grafika tudomány alkalmazási területei közül! Határozza meg a virtuális valóság fogalmát! Mit értünk fotorealisztikus ábrázolás alatt a számítógépes grafikában?

- digitális művészet (digital art)
- grafikai tervezés (graphic design)
- információ megjelenítés (information visualization)
- információ-grafika (infographics)
- oktatás (education)
- racionális gyógyszertervezés (rational drug design)
- számítógépes biológia (computational biology)
- számítógépes fizika (computational physics)
- számítógépes szimuláció (computer simulation)
- számítógéppel segített tervezés (computer-aided design)
- tudományos vizualizáció (scientific visualization)
- videojátékok (video games)
- virtuális valóság (virtual reality)
- webtervezés (web design)

Virtuális valóság alatt elképzelt vagy méretük, illetve távolságuk miatt láthatatlan, veszélyességük miatt megközelíthetetlen világok valósághű, interaktív modellezését és megjelenítését értjük számítógépen.

Fotorealisztikus képábrázolásról akkor beszélünk, ha a számítógépes grafikával generált képeket gyakorlatilag nem lehet megkülönböztetni a fénykép vagy videofelvételtől.

3. fejezet

10. Miből épül fel a raszteres kép? Jellemezze a rasztergrafika modellterét!

A képpontokból (pixelekből) felépülő képet **raszteres képnek** nevezzük, ezek számítógépes feldolgozását pedig rasztergrafikának. Megjegyezzük, hogy a „képpont" megfogalmazás esetenként félrevezető lehet, mivel – helytelenül – a geometriai értelemben kiterjedés nélküli pontra is gondolhatunk.

A **rasztergrafika modelltere** egy kétdimenziós egész koordináta-rendszer, melyben a képpontoknak egészértékű koordinátapontok felelnek meg

11. Mi a Bresenham-algoritmus lényege? Mit jelent a lépcső effektus, és milyen fényerőproblémák jelentkeznek ferde vonalak képernyőn való megjelenítésekor? Mi a super-sampling eljárás?

Bresenham-féle középpontos vonalalgoritmus lényege, hogy a raszteres képen "oszlopirányban" haladva minden egész értékű "x"-koordinátában a matematikai egyeneshez függőlegesen legközelebbi pontot válasszuk ki.

A "pixelesedés" problémájával találjuk magunkat szembe, ha nemcsak vízszintes és függőleges vonalakat húzunk a képernyőn. Nézzük meg például egy ferde egyenes erősen kinagyított képét, amikor az úgynevezett lépcsőeffektus fellép. Ezek a "lépcsős" vonalak már relatíve nem túl nagy felbontásnál is észrevehetők.

A raszteres egyenesszakaszok ábrázolásának további gondja, hogy a ferde és vízszintes vonalak fényereje is eltér egymástól.

A **super-sampling** lényege, hogy például az éleken elhelyezkedő pixeleket felbontják 4×4=16 db további részre, amelyet subpixelnek nevezünk. Ez lényegében azt jelenti, hogy a raszteres képpontokat elvileg egy nagyobb felbontásnak megfelelően számítjuk ki. Egy megjelenített pixel színe vagy szürkeségértéke ezt követően a hozzá tartozó részpixelekhez rendelt értékek átlagolásával kerül kiszámításra.

12. Mit értünk a rasztergrafikus primitívek alatt? Milyen tulajdonságokat lehet hozzárendelni a rajzelemekhez?

A **rasztergrafikus primitívek** olyan, a programcsomagokba beépített rajzelemgenerátorok, amelyekkel a felhasználó tipikus rasztergrafikus objektumokat hozhat létre. Ilyenek például:

- vonalak,
- sokszögek (háromszög, négyzet, téglalap stb.),
- kör és ellipszis,
- szövegek (betűcsalád, betűtípus, például: vastag, dőlt, keskeny, széles, árnyékolt stb.).

A rajzelemekhez tulajdonságokat rendelhetünk hozzá. Ezek lehetnek:

- vonalstílus,
- vonalvastagság,
- szín és
- terület-meghatározó primitívek esetében kitöltő szín, illetve mintázat.

13. Mi a feladata a kiadványszerkesztő programcsomagoknak? Mit értünk DTP alatt? Hogyan határozhatjuk meg rasztergrafikus rendszerekben a szövegek formáját? Mit nevezünk betűcsaládnak? Hogyan kezeli a fontokat a Windows operációs rendszer? Miben különbözhetnek egy családon belül a betűtípusok?

A **kiadványszerkesztő programcsomagok** célja, hogy a felhasználó számára a szöveg szerkesztése mellett olyan funkciókat is biztosítson, amelyekkel a kiadványok nyomdai előállításához szükséges előkészítő munka teljes körűen elvégezhető. Ilyenek például: a nyomdai minőségű szedés, tördelés, több hasáb (kolumna) kezelése, fotók, grafikák négyszínre bontással történő nyomtatása, a szövegelemek helyének tipográfiai pontosságú meghatározása.

Az előbbiekre figyelemmel a nyomdai kiadványszerkesztést (**DTP** = Desk Top Publishing) úgy definiálhatjuk, mint nyomdai anyagok előállítását számítógéppel és a megfelelő szoftverrel.

A formák meghatározásához a rasztergrafikus rendszerek a következő lehetőségeket biztosítják:

- betűcsalád (Times, Helvetica stb.) meghatározása,
- betűtípus (vastag, dőlt stb.) kiválasztása,
- betűnagyság rögzítése.

Betűcsaládnak nevezzük az azonos grafikus jellemzőkkel és formai sajátosságokkal rendelkező betűk összességét.

A betűcsaládok – generálásuk elve szerint – vagy vektorosak vagy raszteresek. A Windows operációs rendszerben például megtalálhatjuk (c:\Windows\Fonts mappában) a

- a TrueType, méretezhető vektorbetűket (kiterjesztésük .ttf) és
- a bittérképes rendszerfontokat (kiterjesztésük .fon).

A betűcsaládokon belül betűtípusokat különböztethetünk meg. A betűtípusok megtartják a család általános grafikai jellemzőit, de néhány tulajdonságukban eltérhetnek egymástól. Ezek:

- a betűkép sötétebb vagy világosabb megjelenése,
- a betűkhöz tartozó vonalak vastagsága (vékony, normál, félkövér, kövér),
- egyenesállású vagy döntött a betű képe.

14. Milyen műveleteket hajthatunk végre a rasztergrafikus felhasználói felületen?

- a kép egy adott részét át lehet helyezni
- nagyításnak csak egész értékeket lehet megadni, és az erősen kinagyított kép eldurvul
- Egy kijelölt ablak tartalmát összenyomhatjuk vagy megnyújthatjuk az X és Y tengely irányában, különböző értékekkel
- Az összenyomás, nyújtás mellett egy szöget is megadhatunk a kép torzításához
- Lehetőség van a teljes rajzelem vagy egy kijelölt részének tükrözésére és 90 fok többszörösével való elforgatására is

15. Mit nevezünk clipartnak, és mire használhatók a clipartok? Mit nevezünk sztereogrammnak? Mit értünk morphing alatt, és mi a morphingalgoritmusok lényege?

Clipartoknak nevezzük azokat a kisméretű képeket, melyeket általában szimbólumonként, emblémaként, logoként vagy egyszerűen csak díszítőelemként alkalmazunk egy képen. Ezek formai megjelenésüket tekintve lehetnek sematikus rajzok, rasztergrafikával generált kis képek vagy beszkennelt (esetleg átalakított) képek.

A **sztereogrammok** olyan 2D-s képek, amelyek a nézőben olyan érzést keltenek, mintha 3D-s képet nézne.

Morphing alatt a számítógépes grafikában azt a transzformációt értjük, amelynek során egy kép alakját folytonosan változtatva „átfolyik” egy másikba.

Ezek az algoritmusok lényegüket tekintve úgy működnek, hogy egy munkahálót feszítünk fel a forrás és a célképre, ezáltal kijelöljük, hogy melyik képrész melyik képrészbe transzformálódjon.

16. Hogyan kereshetjük vissza és módosíthatjuk a raszteres képeket? Milyen fajta raszteres képtípusok vannak? Mit nevezünk színpalettának? Mit jelent a TrueColor, és hány színárnyalatot lehet kódolni ezzel a módszerrel?

A raszteres képen lévő elkülönült grafikus objektumokat egyedileg nem tudjuk visszakeresni. Ha egy objektum egy részletét módosítjuk, akkor a teljes képet meg kell változtatni.

megjeleníthető színek mennyisége alapján négyfajta raszteres képtípust különböztethetünk meg. Ezek lehetnek:

- bittérképes képek (bitmapped image),
- szürkeárnyalatú képek (grayscale image),
- színpalettával indexelt képek (indexed color image),
- valódi színezetű képek (true color image).

A **színpalettával** indexelt képek pixeljeihez egy színindex értéket rendelünk hozzá, amely egy 256 elemű színtáblázatra hivatkozik, amelyet palettának nevezünk. Ezt a rasztergrafikus rendszerek általában a képernyőn is megjelenítik, és így biztosítják a felhasználó számára a szín interaktív (például egérrel való rámutatás) kiválasztását. A színpaletta minden 8 bites indexe tehát egy konkrét színárnyalatot határoz meg egy pixel számára. A rasztergrafikában általában mód van arra, hogy színpalettával indexelt képeknél képenként különböző színpalettát alkalmazzunk (színpalettával indexelt kép).

A valódi színezetű (**True Color**) képek esetében a színtér alapszíneinek megfelelő színcsatornánként adjuk meg az alapszínek intenzitását (True Color raszteres kép). Ez RGB vagy CMY színtér esetén 3x8 = 24 bit, a CMYK színtér esetén pedig 4x8 = 32 bit megadását jelenti. Így például RGB alapszín intenzitások keverésével 224, azaz több mint 16 millió (16 777 216) különböző színárnyalatot tudunk megkülönböztetni.

17. Mikor magas a redundanciája egy közleménynek? Mondjon példákat a kódolási, képi és pszichovizuális redundanciára! Mi alapján lehetséges a veszteségmentes tömörítés? Mi a GIF? Mi teszi lehetővé a veszteséges tömörítést? Mi a JPEG, és milyen tömörítés érhető el felhasználásával? Jellemezze a fraktáltömörítést!

Egy közleménynek akkor **magas a redundanciája**, ha a "megértéséhez" minimálisan szükséges információt kifejező jeleken túlmenően sok felesleges jelet is tartalmaz.

- Kódolási redundancia lép fel akkor, amikor például a fekete-fehér képpontok ábrázolásához 8 bitet, azaz 1 byte-ot használunk fel.
- Képi redundanciát jelent például:
  - ha nagy kiterjedésű, azonos színű objektumok minden egyes képpontját tároljuk a geometriai jellemzők helyett,
  - a mozgóképeknél és az animációknál jelentős redundancia léphet fel akkor, ha az egymást követő képek csak nagyon kis mértékben különböznek egymástól, és ahelyett, hogy csak az eltéréseket tárolnánk, minden képet teljes körűen kódolunk és tárolunk,
  - ha a vektorosan is tárolható alakzatokat raszteresen tároljuk, például egy háromszögnél elegendő a három csúcspont koordinátáit tárolni, és nem kell az összes pixel koordinátáét és színét.
- Pszihovizuális redundanciát jelent például, ha
  - a képminőségbeli eltéréseket (például sok színárnyalattal nagy felbontás) az ember már nem tudja megkülönböztetni,
  - szubjektíve a képnek csak egy része érdekes számunkra, ennek ellenére a teljes képet tároljuk és feldolgozzuk.

A **veszteségmentes képtömörítés** a kép összes információját megőrzi. Ez úgy lehetséges, hogy ezek az eljárások csak a kódolási és a képi redundanciát szüntetik meg.

- A Huffmann-kódolásnál a nagyon gyakori adatokat kódoljuk a legkevesebb bitszámmal. Így például, ha a magyar nyelvű szövegeket kell kódolnunk, a leggyakoribb magánhangzó „E” betűt csak egy bittel jellemezzük.
- A képen azonos jellemzőkkel bíró homogén foltokat kódolhatjuk a geometriai adatokkal és 1 pixel jellemzőivel ahelyett, hogy az összes pixelről tárolnánk az összes információt.
- A mozgóképeknél jelentős tömörítést érhetünk el azzal, ha nem a teljes kép adatait, hanem csak a képpont értékek megváltozott adatait tároljuk.
- A raszteres képeknek az Interneten általánosan elterjedt veszteségmentes képtömörítő eljárása a **GIF** (Graphics Interchange Format). Ez kvázi szabvánnyá a Compuserve hálózat által vált. A GIF tömörítésű fájlok kiterjesztése .gif.

  A **veszteséges tömörítés** a képek pszihovizuális redundanciáját használja ki
- az emberi érzékeléssel ne lehessen észre venni a minőségromlást (ez átlagosan 1:20-tól 1:40 arányú tömörítést tesz lehetővé),
- csak azt kívánjuk, hogy a kép lényeges objektumai felismerhetők legyenek (ekkor kb. 1:200 arányú tömörítést is elérhetünk).
- A veszteséges tömörítés legfontosabb algoritmusa a DFT, azaz a Diszkrét Fourier Transzformáció. (Ezt szokták DCT-nek, azaz diszkrét koszinusz transzformációnak is nevezni.) Ezt az algoritmust alkalmazza az Interneten legelterjedtebb JPEG nevű veszteséges tömörítés, amely a Joint Photographic Experts Group rövidítésből származik. Az eljárás ISO és CCITT szabvány is. A **JPEG** tömörítésű fájlok kiterjesztése .jpg.

Jellemzői :

- a tömörítéssel átlagosan 1:30-as arányt érhetünk el, de ez az eljárást megvalósító szoftverekben paraméterezhető (tehát a minőségromlás árán beállíthatunk nagyobb arányú tömörítést is).
- A tömörített raszteres képeket 24 bites színkódolással is kezelni képes az algoritmus.

A **fraktáltömörítés** alapgondolata, hogy a természetben előforduló képek általában önhasonló mintákat tartalmaznak. Az algoritmus lényege a következőkben foglalható össze: a kép egyes részeit elemi alakzatokkal, fraktálokkal közelítjük, s csak az ezeket a képeket előállító függvények együtthatóit tároljuk. Az adatvesztés ebben az esetben csak a közelítés pontatlanságából adódik. Az alkalmazott módszernek köszönhetően viszont az eredményül kapott kép felbontásfüggetlen lesz.

Mivel várhatóan a fraktáltömörítés a jövőben jelentősen el fog terjedni, fontosabb jellemzőit összefoglaljuk:

- A tömörítésnek jelentős a számításigénye. A tömörített kép kifejtése ennél lényegesen gyorsabb.
- A fraktáltömörítéssel lényegesebben nagyobb tömörítést lehet elérni, mint pl. a JPEG-gel (1:100).
- A fraktáltömörítésnek jobb a kontúr és színárnyalat visszaadása a korábban ismert tömörítő algoritmusokhoz képest.
- A fraktáltömörítésű fájlokat a .fif kiterjesztésről ismerhetjük fel.

18. Melyek a legfontosabb raszteres képfáljformátumok? Soroljon fel néhány vektoros fájlformátumot!

Az általánosan használt (szabványos vagy kvázi szabványos) **raszteres képfájl** típusok a következők:

- BMP = Windows bitmap, független a grafikus kártyától és annak kezelőprogramjától, 24 bites színmélység kezelését biztosítja.
- PCX = A Paintbrush festőprogram fájlformátuma eredetileg, jelenleg az egyik legelterjedtebb formátum. Az RGB színteret a 24 bites színmélységig is lehet kódolni (a CMYK-t nem tudja kezelni).
- TIF = Az ún. TIFF = Tagged Image File Format nagyon elterjedten használt, főleg a DTP területén. Fontos jellemzője az operációs rendszer és a hardverfüggetlenség.
- JPG = A veszteséges képtömörítési szabvány fájlformátum.
- FIF = Veszteséges fraktáltömörítéses fájlformátum, elérhető kb. 1:100 arány is minőségromlás nélkül.
- GIF = A weben legelterjedtebb veszteségmentes tömörítésű fájlok formátuma.
- PDF = Az Adobe cég által kifejlesztett fájlformátum (Portable Document Format). Alapját Postscript lapleírónyelv képezi.
- PCD = A digitális fényképekre a Kodak által kifejlesztett szabvány.
- PNG = Portable Network Graphics, a Compuserve hálózat fájltípusa.

Az általánosan elterjedt vektoros (és esetenként a raszteres képeket is kezelő) fájltípusokat is itt említjük meg:

- EPS (Encapsulated Postscript) = Raszteres és vektoros képek nyomdai munkákhoz való előkészítése során alkalmazhatjuk.
- WMF = Windows Metafile.
- WRL = VRML fájlok formátuma.
- DXF = Huzalvázas vektoros (CAD) objektumokkal kapcsolatos adatcserére használatos (Drawing Exchange Format).
- IGES = (Initial Graphics Exchange Specifications) vektoros fájlszabvány.

Az elterjedt professzionális grafikus programcsomagok fájlformátumai a következők:

- CADL = CADKEY programcsomag fájlformátuma.
- CDR = CorelDRAW fájlformátuma.
- DWG = az AutoCAD fájlformátuma.
- 3DS = a 3D Studio Max fájformátuma.
- PSD = az Adobe Photoshop fájlformátuma.

4. fejezet

19. Jellemezze a vektorgrafika modellterét! Mi jellemzi a 3D-s vektorgrafikus modelltér objektumait? Mit nevezünk világkoordináta-rendszernek?

A **vektorgrafika modelltere** a két- vagy háromdimenziós euklideszi tér.
A valódi 3D-s modellterek objektumait a számítógépen belül teljes értékű háromdimenziós alakzatokként kezeljük. Ezeknek az objektumoknak előállíthatjuk különböző irányú nézeteit, ezek képernyőparancsokkal interaktív módon feldolgozhatók.
Így az olyan tipikus 3D-s műveletek, mint az eltolás, forgatás, tükrözés, másolás a képernyőn való megjelenítéstől teljesen függetlenül végrehajthatók a 3D-s modelltér objektumain.
A számítógépes grafikában a koordinátákat lebegőpontos számként ábrázoló Descartes koordináta-rendszereket **világkoordináta-rendszereknek** nevezik. Ezek a helyvektorok valós koordinátáinak segítségével lehetővé teszik a tárgyak térbeli pozíciójának kifejezését.

20. Hogyan írjuk le matematikailag a térbeli görbéket? Milyen egyenlettel modellezhetők a térbeli felületek?

A térbeli görbék modellezésének legfontosabb eszköze a **paraméteres vektoregyenlet**, amely alkalmazásával a síkbeli és térbeli görbéket azonos módon írhatjuk le.

21. Mi a koordináta-transzformáció, és mire használható a számítógépes grafikában? Mi a ponttraszformáció, és mire használható a számítógépes grafikában? Mit nevezünk affin transzformációnak? Mit nevezünk vetítésnek? Mit nevezünk párhuzamos és középpontos vetítésnek?

Koordináta-transzformációról akkor beszélünk, ha a tárgypontok egy új koordináta-rendszerre vonatkozó koordinátáit határozzuk meg, a régiek ismeretében. Ilyenkor tehát a vizsgált tárgy változatlan, csupán nézőpontunkat változtatjuk meg.
Ponttranszformációról akkor beszélünk, ha a grafikus objektumhoz annak valamilyen értelemben vett hasonmását rendeljük. Tipikus példa erre a fényképezés, ahol a 3D-s tárgyak egyes pontjaihoz egy 2D-s kép pontjait rendeljük hozzá. Ide értjük továbbá a testek elforgatását, elmozgatását stb. is.
Láthattuk, hogy a koordináta-, illetve a ponttranszformációk közül leggyakrabban a számítógépes grafikában a következőket használjuk:
Egybevágósági transzformációk: Ekkor a test képével egybevágó (méretazonos).

- eltolás,
- forgatás,
- centrális tükrözés.

Hasonlósági transzformációk: Ekkor a test képe az eredetihez képest méretarányosan változik meg. Nem változik meg viszont a test alakja és szögei.

- kicsinyítés,
- nagyítás.

Általános léptékváltás: Ekkor a test képe különböző irányokban eltérő módon torzul.

- összenyomás,
- széthúzás.

Ezeket a transzformációkat összefoglaló néven **affin transzformációknak** nevezzük.
Jellemzőjük többek között például a következő:

- pont képe pont, egyenes képe egyenes, sík képe sík,
- metsző térelemek képeinek metszéspontjai megegyeznek az eredeti metszésvonal képével.

Vetítésnek nevezzük az olyan dimenzióveszteséggel járó ponttranszformációt, amelyeknél a tárgypontok és a megfelelő képpontok párhuzamos vagy egy ponton átmenő egyeneseken helyezkednek el. (A fény egyenesvonalú terjedése folytán az optikai leképezések általában ilyen transzformációval egyenértékűek, innen származik a vetítés elnevezés is.)
Párhuzamos vetítésről beszélünk, ha a vetítősugarak egymással párhuzamosak. Ha ezen kívül a vetítősugarak még merőlegesek is a képsíkra, akkor merőleges a vetítés, egyébként pedig a ferde vetítés elnevezést használjuk.
A **középpontos vagy perspektivikus vetítés** esetén a vetítősugarak mindegyike áthalad egy vetítési középponton, a centrumponton (lásd az ábra b) részét). A létrejövő kép igen közel áll az emberi szem, illetve a fényképezőgép által alkototthoz. A perspektivikus hatás elsősorban a tárgy és a centrumpont távolságától függ. Ha ez a távolság minden határon túl nő, a középpontos vetítés párhuzamos vetítésbe megy át.

22. Mi a P pont normalizált homogén koordinátája, ha 3D-s koordinátái x, y, z? Írja fel az affin transzformációk egyenletét homogén koordinátákban!

A P(x,y,z) pont homogén koordinátáinak nevezzük a (w•x, w•y, w•z, w) koordináta négyest, ahol w egy tetszőleges, nemzérus skalár. Ha w értékét 1-nek választjuk, akkor az (x, y, z, 1) koordinátanégyest a P pont **normalizált homogén koordinátáinak** nevezzük.

$$\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} T_{11} & T_{12} & T_{13} & b_x \\ T_{21} & T_{22} & T_{23} & b_y \\ T_{31} & T_{31} & T_{31} & b_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix}$$

23. Mit tartalmaz a geometriai objektumokról a vektorgrafikus adatbázis? Mi a különbség az adat- és eljárásorientált geometriai modellezés között? Milyen kapcsolatok lehetnek a geometriai objektumok között? Milyen tulajdonságokat rendelhetünk a vektorgrafikus geometriai objektumokhoz? Milyen adatokat tartalmaz a vektorgrafikus adatbázis?

A **vektorgrafikus adatbázisban** nem a tárgyak, testek képeit, hanem a megfelelő 3D-s világtérbeli geometriai objektumok matematikai és strukturális adatait dolgozzuk fel. Ezeknek az adatoknak a konkrét tartalma és megjelenési formája (például csak egy poliéder csúcsainak koordinátáit tároljuk, vagy emellett még a lapjainak normálvektorát is) a kiválasztott geometriai modellezési eljárástól függ.

Adatorientált esetben a térbeli alakzat jellemző adatait tároljuk a számítógépes rendszerben (például háromszög esetében a csúcspontokhoz vezető vektorok koordinátáit) míg **eljárásorientált** esetben a térbeli alakzat generáló programját (például körgeneráló rutin a kör egyenlete és a középpontjának és sugarának paraméterei alapján).

A geometriai objektumok közötti kapcsolatok lehetnek:

- alá–fölérendeltségi hierarchikus viszonyok, amelyek jellemző változata a tartalmazás (például: ház → tetőszerkezet → tetőablak),
- mellérendeltségi viszonyok. Ezek közül a legfontosabbak a következők:
  - a szerkezeti jellegű kapcsolatok, amikor a geometriai objektumok hierarchiában azonos felsőbb szintű objektumhoz kapcsolódnak (például egy ház egyik falán található ablakok),
  - az illeszkedési jellegű kapcsolatok, amikor a geometriai objektumok valamilyen formában csatlakoznak egymáshoz (például egy kocka egy csúcsából kiinduló három él),
  - a megjelenítés jellegű kapcsolatok, amikor a geometriai objektumok azonos raszteres képhez, jelenethez (scene) tartoznak.

A vektorgrafikus adatbázis geometriai objektumaihoz különböző – jellemzően megjelenítésorientált – tulajdonságokat is hozzárendelhetünk. Ilyenek például a rasztergrafikához hasonlóan

- a szín,
- a vonalstílus,
- a felületi jellemzők: textúrák, felületre vetített raszteres képek, érdesség stb.,
- a szövegek.

Összefoglalva az eddigieket, a **vektorgrafikus adatbázis** a következő jellegű adatokat tartalmazza:

- a 3D-s világkoordináta-rendszerben meghatározott geometriai objektumok adatait:
  - a modell neve, azonosítói, a geometriai alakzatot felépítő geometriai építőelemek fajtái,
  - az építőelemek kapcsolódásaira vonatkozó adatok,
  - a geometriai alakzatra vonatkozó méret-, nagyságadatok,
  - a geometriai alakzatra vonatkozó hely- és helyzetadatok a modellezési világkoordináta-rendszerben,
  - a geometriai alakzat tulajdonság adatai,
  - a geometriai alakzat megjelenítésének adatai.
- a geometriai objektumok közötti viszonyokat meghatározó strukturális kapcsolatok adatait,
- a modelltér geometriai objektumaihoz rendelt mennyiségi és szervezési információk adatait.

24. Hogyan lehet megoldani a modellezési algoritmusok egységes kezelését? Milyen illeszkedési feladatokat kell megoldani a görbék és felületek modellezés eljárásával? Hogyan biztosítható a modellezés során felhasznált függvények egyszerű kiszámíthatósága? Sorolja fel, milyen követelményeket kell kielégíteni a vektorgrafikus modellezésnek a felhasználóval való interaktív kapcsolattartás miatt!

Az algoritmusokat programtechnikailag egységesen, relatíve nem nagy ráfordítással kell megvalósítani. Ezért olyan modellezési eljárások kellenek, amelyek kizárják az egyedi, egy-egy speciális függvényhez kötődő megoldásokat. Ugyanakkor a modellezési eljárásnak a legkülönbözőbb formájú, alakú görbéket, felületeket is elő kell tudnia állítani.

A modellezési eljárás biztosítsa az előre megadott pontokon áthaladó térgörbék és felületek hatékony generálását, támogassa a különböző térelemek illeszkedésének (metszéspontok, érintés) kezelését.

A modellezési eljárásban felhasznált függvényeknek viszonylag egyszerűen kiszámíthatóaknak kell lenniük. Az egységes programtechnikai kezelés miatt ezek azonos tulajdonságú függvénycsaládokból kerüljenek ki.

25. Milyen fokszámú polinomokat célszerű választani a modellezéshez, és miért? Mivel közelítjük a térgörbéket és a felületeket a számítógépes grafikában?

Q számítógépes grafikában a térgörbék és felületek modellezésére a harmadfokú polinomokat választották. Ezt a döntést az is indokolja, hogy harmadfokú polinomokkal modellezhetők olyan geometriai tulajdonságok, mint az önmetszés, csúcspont vagy az inflexiós pont (ahol az érintő átmetszi a görbét). Tehát a térgörbéket az r = a0 + a1•t + a2•t2 + a2•t3 alakú **köbös paraméteres ívekkel**, a felületeket pedig **kétparaméteres köbös felületekkel** közelítjük (a felületeknél tehát x(u,v), y(u,v), z(u,v) u és v harmadfokú polinomja

26. Milyen két eltérő módszer létezik az interpolációs és approximációs feladat megoldására? Mit értünk spline görbék alatt? Mit értünk spline felületek alatt?

Az interpolálással, illetve approximálással képzett görbék és felületek előállítására két, lényegében eltérő módszer létezik:

- az összes interpolálandó vagy approximálandó pontot figyelembe véve egyetlen görbét vagy felületet határozunk meg,
- az interpoláló vagy approximáló görbét, illetve felületet egymáshoz folytonosan kapcsolódó részekből állítjuk össze.

Ha a modellezendő térgörbét több, egymáshoz folytonosan kapcsolódó ívből állítjuk elő, akkor ezt a görbét spline-nak nevezzük.

Ha egy modellezendő felületet részekből állítunk össze, akkor ezt spline felületnek nevezzük. A spline felületek részeit leggyakrabban köbös polinomokkal generáljuk, ezeket kétparaméteres köbös felületfoltoknak (bicubic patch) nevezzük.

27. Miért előnyös egy kontrollpontokkal generált görbe esetében, ha invariáns az affin transzformációkra?

Az **invariancia** azért előnyös, mert ilyen tulajdonságú függvények összes pontja helyett elegendő az r0, r1 ... rn kontrollpontokat transzformálni, ha például a görbét elmozgatjuk a térben. Ezzel jelentős mennyiségű számítást takaríthatunk meg.

28. Definiálja a Bézier-íveket! Milyen tulajdonságai vannak a Bézier-íveknek? Mire használható a de Casteljau-algoritmus? Milyen részekből épülnek fel a B-spline görbék? Definiálja a B-spline görbéket! Mi a B-spline görbe súlyfüggvénye és Boor-poligonja? Miben különböznek a B-spline és a Bézier-görbék? Mi a Cox–de Boor-algoritmus lényege? Miért vezették be a racionális görbéket a számítógépes grafikában? Minek a rövidítése a NURBS, és mit jelent? Melyek a NURBS legfontosabb tulajdonságai?

A harmadfokú Bézier-íveket [jelölése B3(t)] a következő képlettel definiálhatjuk, ha adott a görbe P0, P1, P2, P3 kontrollpontja az r0, r1, r2 és r3 vektorokkal:

$$\underline{B}_3(t) = \sum_{j=0}^{3} \underline{r}_j \cdot B_j^3(t) \qquad t \in [0,1]$$

ahol $B_i^3(t) = \binom{3}{i} \cdot t^i (1-t)^{3-i} \quad i = 0, 1, 2, 3 \quad 0 \le t \le 1$

A Bézier-ívek néhány, a számítógépes grafikában fontos tulajdonsága:

- A görbe íve mindig a P0, P1, P2, P3 kontrollpontok által meghatározott négyszög belsejében helyezkedik el.
- A Bézier-ívek kontrollpontjaik affin transzformációjával szemben invariánsak. Ez azt jelenti, hogy például a görbe mozgatásakor elegendő a kontrollpontokat transzformálni, amellyel igen sok számítás megtakarítható. (Ez azonban csak az affin transzformációkra igaz, és nem minden esetben teljesül a projektív transzformációkra, például a centrális vetítésre.)
- A Bézier-ívek globálisan változtathatók, azaz ha a kontrollpontok közül egyet elmozgatunk, az az egész görbére kihat.
- Egy egyenes pontosan annyi pontban metszi a Bézier-ívet, ahány metszéspontja van a tartónégyszögével

A Bézier-ívek megjelenítése szempontjából nagyon fontos, hogy létezik-e olyan algoritmus, amely alapján az ív egyenesszakaszokkal közelíthető. Többek között erre a problémára ad választ a **de** Casteljau-**algoritmus**. Ez egy rekurzív eljárást ad a Bézier-ív egy konkrét t paraméterértékhez tartozó pontjának kiszámítására és az ív egyenesszakaszokkal való közelítésére.

Legelterjedtebben a számítógépes grafikában a spline-ok generálásához részívnek úgynevezett B-spline bázisfüggvényeket választanak.

Legyen adva n+1 kontrollpont (P_0, P_1, ... P_n) az r_0, r_1, ... r_n helyvektorokkal. Ekkor egy B-spline görbét az ún. B-spline bázisfüggvények lineáris kombinációjaként a következő képlettel határozhatunk meg:

$$B_{n,k}(t) = \sum_{i=0}^{n} \underline{r}_i \cdot N_{i,k}(t)$$

Tehát a B-spline alappolinomok súlyozott összegzésével jön létre a B-spline görbe, ezért az összefüggésben szereplő $N_{i,k}(t)$ B-spline bázisfüggvényeket **súlyfüggvényeknek** is nevezik.

B-spline görbék esetén a P_0, ... P_n kontrollpontok által meghatározott poligont **Boor-poligonnak** nevezzük, a kontrollpontokat pedig Boor-pontoknak.

Matematikailag bizonyítható, hogy a B-spline görbék rendelkeznek a Bézier-görbék összes előnyös tulajdonságával, és ezen túlmenően **lokálisan is változtathatók**, azaz egy kontrollpont elmozgatása csak a neki megfelelő $[t_i, t_{i+k}]$ intervallumban módosítja a görbe alakját.

A B-spline görbék ugyanúgy, mint a Bézier-görbék, közelíthetők poligonokkal a kontrollpoligon szakaszainak rekurzív felosztásával, és ez az algoritmus elő is állítja a B-spline görbe egy pontját egy konkrét t értékre. Ekkor a közelítő sokszögek a görbéhez tartanak határértékben. Ezzel egy hatékony eljárást kapunk a B-spline-ok raszteres képernyőn megjelenítésére. Ezt az eljárást **Cox–de Boor-algoritmusnak** nevezzük, amely a de Casteljau-algoritmus általánosítása.

A B-spline görbék már majdnem kielégítik a modellezéssel kapcsolatos összes követelményt, de még mindig nem rendelkeznek a centrális vetítésekre vonatkozó invariancia tulajdonságával.

A racionális görbéknek a vetítésekre vonatkozó invariáns tulajdonsága miatt a képernyőn történő ábrázolás előkészítése során elegendő csak a kontrollpontok képét kiszámítani.

A racionális B-spline-ok közül a legfontosabbak a nem uniform (azaz nem egyenközű $[t_i, t_{i+1}]$ felosztással kapott) racionális B-spline görbék, amelyeket **NURBS**-nek szoktak rövidíteni (NURBS = Non Uniform Rational B-Spline).

A NURBS tulajdonságai:

- A NURBS invariáns az affin transzformációkra és a vetítésekre (projektív transzformációkra). Ez azt jelenti, hogy például a NURBS görbe centrális vetítésekor elegendő a Boor-pontokat homogén koordinátákban transzformálni, és ezekből kiszámítani a vetületi görbét.
- Egy kontrollpont megváltoztatása csak a megfelelő paraméter tartományhoz tartozó pontokban módosítja a görbét, azaz a NURBS lokálisan változtatható.
- A NURBS görbe a kontrollpoligon belsejében (konvex burkában) helyezkedik el.

- Ha a kontrollpontok egy egyenesen helyezkednek el, akkor a NURBS egyenesszakasz.
- Ha az egyik r_i kontrollpont w_i súlyát megnöveljük, a görbe határértékben a P_i ponthoz tart .

29. Mit nevezünk vonalfelületnek? Hogyan hozhatunk létre felületeket görbék mozgatásával? Határozza meg a Bézier-felületeket! Határozza meg a B-spline felületeket! Definiálja a NURBS felületeket!

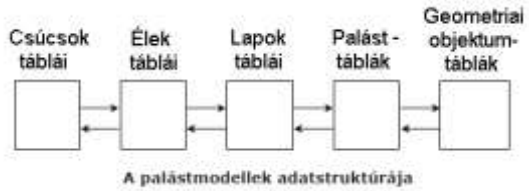
Ha egy felület bármely pontján át húzható olyan egyenes, amelynek pontjai a felülethez tartoznak, akkor ezt a felületet **vonalfelületnek** nevezzük.
Felületeket létrehozhatunk úgy is, hogy egy adott **generáló térgörbét** önmagával párhuzamosan mozgatunk egy vezérgörbe mentén
Bézier-felületnek nevezzük azokat a felületeket, amelyek Bézier-görbék mozgatásával úgy jönnek létre, hogy a mozgatott görbe kontrollpontjai szintén Bézier-görbéken mozognak.
B-spline felületnek nevezzük azokat a felületeket, melyek B-spline görbék mozgatásával úgy jönnek létre, hogy a mozgatott görbe kontrollpontjai szintén B-spline görbéken mozognak.
A **NURBS felületeket** teljesen analóg módon származtatjuk a Bézier és B-spline felületekkel. Ekkor egy racionális, nem uniform B-spline görbét mozgatunk úgy, hogy kontrollpontjai szintén NURBS görbéken mozogjanak.

30. Mit értünk huzalvázmodell alatt? Mi az előnye a huzalvázmodell alkalmazásának? Milyen korlátai vannak a huzalvázmodell alkalmazásának? Mit tárolunk az adatbázisban huzalvázmodell esetén?

A **huzalvázmodell** (wireframe model) a 3D-s geometriai alakzatot csúcsaival és éleivel jellemzi, ennek megfelelően a modell csak a csúcsokat és az ezekhez rendelt összekötő éleket tartalmazza
A huzalvázmodellek legnagyobb **előnye**, hogy számítógépes megvalósításuk algoritmusigénye a többi geometriai modellezőmódszernél lényegesen kisebb, így viszonylag kis erőforrású számítógépen is használhatók.
A huzalvázmodellek legnagyobb **problémája**, hogy egy huzalvázmodellnek több test is megfelelhet. Nem mindig tehető különbség a tömör és üreges test között a modell alapján, és a testet határoló felületek görbültségét sem tudjuk kezelni
A huzalvázmodell **adatstruktúrájának** lényege a csúcs-, az él- és az él-csúcs táblázatok együttese, amelyeket a relációs adatbáziskezelés szabályainak megfelelően építenek fel.

31. Hogyan jellemezzük a geometriai objektumokat palástmodellezésnél? Mi a b-rep? Mit értünk lépésenkénti szerkesztés alatt? Mivel generáljuk a testeket határoló felületeket valósághű palástmodellezésnél? Milyen táblákból áll a palástmodell adatbázisa?

A **palástmodellezésnél** a geometriai objektumokat a vektorgrafikus modelltérben határolófelületeikkel (beleértve e felületek csatlakoztatására vonatkozó adatokat is) jellemezzük.
Ennek a modellezési módszernek a neve az angol szakirodalomban **boundary-representation** vagy röviden b-rep.
Lépésenkénti szerkesztés: a test határoló felületeit egyenként definiáljuk a térbeli felületek csatlakoztatási lehetőségeinek függvényében.
Valósághű palástmodellezés: a testet behatároló felületfoltok generálása és illesztése során a B-spline, a Bézier és NURBS felületfoltok alkalmazását is lehetővé teszi.



A palástmodellek adatstruktúrája

32. Milyen két típusa van a tömör testmodellezésnek? Mi a CSG modellezés lényege? Milyen standard testprimitívek vannak a CSG-ben? Milyen műveletek alkalmazhatók a CSG-ben? Milyen a CSG modellek adatstruktúrája? Miért használható a tömör testmodellezés a műszaki tervezésben?

A **testmodellezési eljárások** közül a számítógépes grafikában két modellezési módszer vált elterjedté:

- elemi testekkel és ezek közötti szabályos halmazműveletekkel való modellezés és
- a testek elemi sejtekből való felépítése.

A véges számú tömör elemi testprimitívből kiinduló és a modellt a metszet, egyesítés, kivonás és ragasztás halmazműveletek egymás utáni felhasználásával megkonstruáló modellezési módszert konstruktív tömör testmodellezésnek nevezzük. Ennek angol elnevezése: Constructive Solid Geometry, vagy röviden CSG.

A standard testprimitívek induló készlete általában a már bemutatott 3D-s primitívekből

- a hasáb,
- a gúla,
- a henger,
- a kúp és
- a gömb

A konstruktív tömör testmodellezéssel létrehozott modellek adatstruktúrájára a bináris fa gráf jellemző, amelynek ágcsomópontjai a halmazalgebrai műveletek, levelei pedig a műveletben részt vevő testek

A konstruktív tömör testmodellezésnek a gyakorlatban történő jól használhatóságára hívják fel a figyelmet azok a vizsgálatok, amelyek eredményei szerint a műszaki tervezés során szükséges testek döntő része előállítható néhány egyszerű geometriai test (például hasáb és henger) megfelelő kombinációjából.

33. Mi a térfogatmodellezés alapelve? Mit nevezünk voxelnek? Hogyan modellezhetjük a testeket voxelekkel?

A térfogatmodellezésnél (cell modeling) egy tömör tárgyat több egymáshoz csatlakozó, de egymást nem metsző kisebb tömör tárgyra, azaz sejtekre bontunk fel. Az elterjedtebb modellezési módszerek a sejtek két típusát kezelik:

- a sejtek azonos típusú alakzatok (például hasábok), de méretük egy paramétertől függően változhat,
- a sejtek azonos típusú és méretű alakzatok, ekkor ezeket (a képponthoz, azaz a pixelhez hasonlóan) **voxelnek** (volument element) nevezzük.

A modellezendő objektumokat a voxelekkel úgy írjuk le, hogy minden egyes, a testhez teljes egészében vagy csak részlegesen hozzátartozó voxel adatait hozzárendeljük a testhez

34. Mit nevezünk sugárzási teljesítménynek? Mi a lumen? Mit nevezünk kisugárzás-erősségnek? Mi a candela? Mi a besugárzás-erősség? Mit mond ki a Lambert-féle távolság- és koszinusztörvény? Mi a megvilágításerősség mértékegysége? Definiálja a sugársűrűséget és a fénysűrűséget! Mi jellemzi a Lambert-fényforrásokat?

A fényforrás által időegység alatt kisugárzott energiát **sugárzási teljesítménynek** vagy fluxusnak nevezzük, egysége a watt [W].

A sugárzási teljesítmény fotometriai megfelelője a fényáram, amelynek mértékegysége a **lumen** [lm].

Ha a pontszerű fényforrás az egységsugarú gömb egységnyi térszögébe időegység alatt I energiát sugároz ki, akkor I-t a **kisugárzás-erősségnek** nevezzük (radiant intensity).

Az I sugárzáserősség megfelelője a fotometriában a fényerősség. Ennek egysége a **candela [cd]**, ami kb. egy gyertya fényének felel meg.

Az egységnyi felületre időegység alatt eső sugárzási energiát **besugárzás-erősségnek** (irradiance) nevezzük.

A távolság- és koszinusztörvényt szétválasztva is megfogalmazhatjuk. Ekkor:

- A **távolság törvény** kimondja, hogy pontszerű fényforrásnál, azonos beesési szögnél a besugárzás erőssége fordítottan arányos a felületnek a fényforrástól mért távolsága négyzetével.
- A **koszinusztörvény** szerint pedig a felület besugárzás-erőssége alfa szög irányában egyenesen arányos a beesési szög koszinuszával, tehát akkor a legnagyobb, ha a fénysugár iránya azonos a felület normálvektorával.

A besugárzás-erősségnek a fotometriai megfelelője a **megvilágítás-erősség**, amelynek egysége a lux [lx].

Sugársűrűség: az egységnyi térszögből az időegység alatt az egységnyi kisugárzó felület vizsgált irányú vetületéről származó sugárzási energiát jelenti.

Fénysűrűség: általános esetben a fénysűrűség a kisugárzási irány (ezt a φ és θ polárszögekkel adhatjuk meg) és a λ hullámhossz függvénye.

Azokat a testeket, melyek oly módon sugároznak ki fényt, hogy a fénysűrűség irányfüggetlen, **Lambert-féle fényforrásoknak** nevezzük.

35. Milyen típusú fényforrásokat adhatunk meg a vektorgrafikus rendszerekben? Jellemezze az egyes fényforrástípusokat!

Vektorgrafikus rendszerekben általában a következő típusú fényforrásokat definiálhatjuk:

- szórt háttérvilágítás, amelynek nincs iránya,
- távoli fényforrások, amelyek irányított párhuzamos fénysugarakat bocsájtanak ki,
- pontszerű fényforrások, amelyek egy térbeli pontból minden irányban kibocsájtanak fénysugarakat,
- reflektor-fényforrások, amelyek a tér egy adott helyétől egy meghatározott irányban, párhuzamos vagy csonka kúp alakú fénynyalábot sugároznak,
- kiterjedt fényforrások, amelyek meghatározott méretű és alakú sugárzó testek.

36. Mit nevezünk fraktálnak? Melyek a fraktálok főbb csoportjai? Mi jellemzi az egyes fraktálcsoportokat? Jellemezze a fraktálok felhasználási területeit a számítógépes grafikában!

A fraktál egy olyan geometriai alakzat, amelynek részei hasonlítanak az egész alakzathoz és a részek a még kisebb részekhez.

A fraktáloknak számos csoportja, alcsoportja van. Ezek közül néhány:

- IFS (iterate function systems, iterációs függvény rendszerek).
- Furcsa attraktorok (strange attractors): Attractor: a labilis és a stabilis tartomány határa. A kitöltött attractor nem divergens. Az attraktorhoz konvergálunk, ha az stabil.
- Lángfraktálok (flame fractals).
- L-rendszer fraktálok (L-system fractals).
- Komplex polinomok iterációjával létrehozott fraktálok (a leghíresebbek).
- Quaternionic és mostanában a hypernionic fraktálok.
- Random fraktálfolyamatokkal generált fraktáltájak.

A fraktálok számos területen alkalmazhatók, többek között:

- Kórszövettani diák osztályozása az orvostudományban
- Fraktál táj vagy partvidék komplexitás
- Enzim/enzimológia (Michaelis-Menten kinetika)
- Új zene generálása
- 2D-s, 3D-s művészi képek, animációk, szobrok
- Jel- és képtömörítés
- Digitális litográfiai nagyítás létrehozása
- Szeizmológia
- Talajmechanika
- Számítógép- és videojáték-tervezés, kiemelten a szerves környezethez és a procedurálisan generált részekhez
- Fractography és törésmechanika
- Fraktál antennák (kis méretű antennák fraktál alakzatok felhasználásával)
- A fraktálisan durva rendszerek kis szög szórási elmélete
- Pólók és egyéb divat
- Álcázáshoz minták generálása
- Digitális napóra
- Ársorozatok technikai elemzése
- Fraktálok hálózatokban

37. Mit nevezünk karakternek és karaktermodellezésnek? Milyen módszerek vannak a karakterek létrehozására a 3D modelltérben? Hogyan készítünk a 3D-ben beszkennelt makettből a grafikus rendszerben kezelhető objektumot? Mi a különbség a bump mapping és a displacement mapping eljárás között?

A számítógépes grafikában az animációkészítés során a modelltérben létrehozott, mozgó, és alakjukat meghatározott törvényszerűségek szerint változtató, ugyanakkor képi jellegzetességük lényegét megőrző 3D objektumokat **karaktereknek** (character), az ezek létrehozásával és mozgatásával összefüggő eljárásokat és módszereket pedig **karaktermodellezésnek** nevezzük.

Ahhoz, hogy a 3D virtuális világban egy "színészt", azaz egy karaktert létrehozzunk, és mozgassunk, a modelltérben meglehetősen összetett és bonyolult műveletsorozatot kell végrehajtani. Ennek egyes elemei általában a következők:

- Létrehozás (modeling)
- Textúrázás (texturing)
- Csontvázrendszer felépítése (skeleton system)
- Beborítás (enveloping)
- Mozgatás (animating)
- Arcanimáció (facial animation)

A **szkenneléshez** először egy (általában kicsinyített) makettet építünk fel az elképzelt lényről. A makettet egy optikailag fényvisszaverő anyaggal vonjuk be, és az adatait egy nagy felbontású, háromdimenziós szkennerrel visszük be a grafikai rendszerbe. A háromdimenziós szkenner a letapogatott tárgyról olyan 3D modellt készít, amely több millió referenciapontból áll. Ezek a referenciapontok határozzák meg a tárgy alakját, körvonalát. A keletkezett pontfelhőből poligonokat készítünk, így kapunk egy könnyebben kezelhető objektumot. A hatékony munkavégzéshez még ez a felbontás is magas, ezért a poligonokat NURBS felületfoltokkal közelítjük (kb. 80–100 felületfolt elegendő egy modell megépítéséhez, persze ez az érték függ a modell részletességétől is).

A texturázással határozzuk meg a karakter felületének jellemzőit. Érdes felületeket a bump mapping vagy a displacement mapping eljárással hozhatunk létre (lásd ábrákat):

- A bump mapping eljárásnál a megjelenítő szoftver egy fekete-fehér intenzitású mintázatot figyelembe véve módosítja a felületi normál-vektorokat, és ezzel kiemelkedéseket és besüllyedéseket szimulál. Ha ezt alkalmazzuk, akkor a modelltérbeli objektum nem változik.
- A NURBS felületek parametrizálhatósága lehetővé teszi, hogy a felületelemeket ténylegesen eltoljuk a normál vektora irányába (displacement mapping). Ez az eljárás általában jobb eredményeket szolgáltat a bump mapping eljárásnál, de sokkal számításigényesebb, és megváltoztatja a modelltérbeli objektumot is.

38. Mi a feladata a karaktermodellezésben a csontváznak? Mit nevezünk kinematikai láncnak? Mi jellemzi a forward kinematikát? Mi jellemzi az inverz kinematikát?

A modell felszínét alkotó építőelemek (spline-felületfoltok, patch-ek) folytonosan illeszkednek egymáshoz, és pontosan leírják az összefüggő testfelszínt. Ahhoz, hogy ezt a felszínt mozgatni tudjuk, egy olyan struktúrára van szükség, amely kisszámú vezérlő pont segítségével az összes elvárt deformációt létre tudja hozni. Ezt a problémát oldják meg az ún. kinematikus rendszerek, amelyekben a karakter részeinek mozgatása, deformálása (az emberi, állati testhez hasonlóan) egy **csontvázrendszer** segítségével történik.

A csontváz (skeleton) olyan részeit, amelyekhez tartozó csontok kapcsolódnak egymáshoz és együtt mozognak, **kinematikai láncnak** nevezzük.

A **forward kinematikában** (forward kinematic) a kinematikus lánc minden egyes tagját (bone) egyenként forgathatjuk a kapcsolódási pontjai (joint) körül. Amikor egy kívánt pózba szeretnénk beállítani a karakterünket, akkor a hierarchia legtetejétől (root) elindulva forgatjuk a csontrendszert – ha szükséges egyenként – a megfelelő helyre. Ha egy csontot elforgatunk, akkor a hozzátartozó (belőle származó), alacsonyabb szinten álló elemek is követik a forgást.

Az **inverz kinematika** (inverse kinematic) alapgondolata az, hogy a kinematikai láncban különböztessünk meg egy vezérlő csontrészt, amelynek mozgatása esetén a kinematikai rendszer képes a szülő rész megfelelő elmozdulását automatikusan kiszámítani.

39. Mit kell elvégezni a karaktermodellezés „beborítás” munkafázisában? Mit nevezünk elsődleges és másodlagos mozgásnak a karakteranimációban?

A **beborítás** vagy bőrözés (enveloping, skining) lépésében a létrehozott karakter felületét alkotó elemeket hozzárendeljük a csontvázrendszerhez, figyelembe véve a mozgással járó felületi deformációkat (pl. embernél izomzat mozgása).

Az **elsődleges mozgás** adja meg a karakter fő mozgásait és cselekvéseit, a **másodlagos** pedig ezek esetleges mellékhatásait, mint például hús, bőr és izomzat deformációit.

40. Mi a fázisanimáció (key framing) lényege? Hogyan készítik el a karakter kulcspozícióit a 3D mozgásrögzítés (motion capture) módszerével? Mire használjuk a klipeket nemlineáris animációnál? Ismertesse az arcanimáció (facial animation) lépéseit!

A **fázisanimáció** (key framing) eljárás a rajzfilmkészítés gyakorlatából származik, ahol először a kulcsfontosságú képkockákat rajzolták meg, és ezek közötti képeket a fázisrajzolók később készítették el.

A testre aktív vagy passzív érzékelőket rögzítenek, amelyek pozícióját a felvétel során nagy pontossággal mérik. Ezeknek a méréseknek az eredményei alapján azután "könnyen" kiszámítható a szereplő csontvázának helyzete és például csuklóinak pontos orientációja.

A **nemlineáris animációkat** (nonlinear animation) ennek az elvnek megfelelően az egyes mozdulatsorozatoknak megfelelő újrafelhasználható (elmentett) részanimációkból, az ún. klipekből állítjuk össze.

Az **arc animációját** a következő lépésekben végezhetjük el:

- Lemodellezzük a karakter arcát. Ez általában kifejezéstelen "pókerarc". Ezt nevezzük bázisobjektumnak.
- Másolatokat készítünk a bázisobjektumról.
- A másolatokat úgy deformáljuk, hogy különböző arckifejezéseket mutassanak (célobjektumok).
- Interpolációval elvégezzük a bázis- és célobjektumok közötti közbeeső állapotok generálását.

41. Hogyan lehet a vektorgrafikus objektumok helyzetére vonatkozó információkat közölni a vektorgrafikus rendszerrel? Hogyan lehet egérrel kijelölni egy vektorgrafikus objektumot?

Ha a felhasználó az alakzatok helyzetére vonatkozó információkat akar közölni a vektorgrafikus rendszerrel, akkor ezt a következő formákban teheti meg:

- a konkrét koordinátaértékek begépelésével billentyűzetről,
- a grafikus kurzor helyzetének megfelelő koordinátaértékekkel, például az egér vagy más relatív koordinátás eszköz használatával,
- abszolútkoordinátás eszköz, például digitalizáló tábla (tablet) használatával,
- a modelltérben már definiált helyzetű objektumokra, vagy a képernyőn látható, a világkoordináta-rendszerben értelmezett segédeszközökre (vonalzók, segédvonalháló = grid) való hivatkozással.

Az **egérrel** a képernyőn rá kell mutatni a kijelölendő objektum képének határoló görbéjére vagy zárt alakzatok belsejére. A grafikus kurzor raszteres képernyőkoordinátáit ekkor először át kell konvertálni a modelltérbe, és meg kell állapítani, hogy a konvertált koordinátaérték melyik grafikus objektumhoz tartozik.

42. Milyen műveleteket végezhetünk a modelltér objektumaival? Hogyan hozunk létre primitívekből geometriai objektumokat a modelltérben? Milyen 2D-s és 3D-s primitíveket ismer?

A vektorgrafikus szoftverek a modelltér objektumaival a következő típusú műveleteket teszik lehetővé:

- új objektum létrehozása,
- egy létező objektum transzformálása, másolása, törlése,
- meglévő objektumokkal végzett halmazalgebrai műveletek,
- struktúra képzés,
- jelenetek megjelenítése.

A modelltérben egy új objektumot általában

- primitívpéldányokra hivatkozással vagy
- szerkesztéssel vagy
- pásztázással vagy
- szkenneléssel

A 2D-s rajzolóprogramok az alakzatokat általában

- a vonal (szakasz),
- a téglalap (négyzet),
- az ellipszis (kör),
- a sokszög

primitívek véges számú kombinációjával állítják elő.

A 3D-s vektorgrafikus rendszerekben ezek általában még a következő térbeli primitívekkel egészülnek ki:

- hasáb (beleértve a téglatestet és a kockát is),
- gúla (csonka gúla),
- henger,
- kúp (csonka kúp),
- gömb,
- tórusz.

43. Mit jelent a szerkesztéssel történő objektumdefiniálás? Mit jelent a pásztázás? Mit jelent a kihúzás? Hogyan hozhatunk létre térbeli alakzatokat forgatással?

Objektum létrehozása szerkesztéssel: a felhasználó adja meg a vektorgrafikus rendszer számára az összes információt, amely alapján a térbeli test összeállítható. Ez a módszer tipikusan jellemző a testek palástfelületekkel való modellezésére (b-rep), amikor a felhasználó egyenként meghatározza az egyes fedőlapok, felületek jellemzőit és az ezek csatlakoztatására vonatkozó adatokat.

Objektum definiálása pásztázással: a 3D-s vektorgrafikus objektumok generálásának fontos módszere a pásztázás (ezt gyakran söprésnek is nevezik, angolul sweep). Pásztázásnál egy 2D-s felületelemet mozgatunk egy vezérgörbe mentén, és ennek során a felületelem által „súrolt" térbeli pontok egy testet határoznak meg.

A **kihúzásnál** a síkra merőlegesen, egy egyenes mentén "kihúzzuk" a 2D-s alakzatokat. Speciális esete ennek, amikor egy 2D-s raszteres kép szürkeségértékeit magasság koordinátákként értelmeztük.

Forgatásnál a vezérgörbe kör, a generáló alakzat síkja a forgatás bármely pillanatában merőleges a vezérgörbére, és a generáló alakzat pontjai is körön mozognak.

44. Mit jelent a vektorgrafikus objektum másolása? Milyen transzformációkat tesznek lehetővé a vektorgrafikus rendszerek? Melyek a vektorgrafikus objektumokkal végrehajtható Boole-algebrai műveleteket? Mi a hatása az összeragasztás műveletének? Mi a lényege a fóliák alkalmazásának?

A **másolás** a kijelölt vektorgrafikus objektum egy eltérő nevű (azonosítójú) példányát hozza létre a világkoordináta-rendszer egy másik helyén.

Objektumok transzformálásához a már megismert ponttranszformációk közül választhatunk. Általában minden vektorgrafikus rendszer lehetővé teszi az objektumoknak

- az eltolását,
- az elforgatását (ide értve a tükrözést is) és
- a léptékváltást (nagyítás, kicsinyítés, összenyomás, széthúzás).

Az általánosan használt és a legtöbb vektorgrafikus rendszerben értelmezett Boole-algebrai műveletek a következők:

- egyesítés (union),
- kivonás ([difference vagy substract),
- metszet (intersection).

A napi gyakorlati munkavégzés komfortjához ezek mellett még szükség van az **összeragasztás** (glue) műveletére is, amelynek során a műveletben résztvevő testeket egy közös felületen csatlakoztatjuk.

A **fóliák** lényege, hogy az objektumok a felhasználó által megadott csoportosításban külön fóliákra kerülhetnek, amelyek a rendszerben külön kezelhetők. Ezt a 2D-s rajzelemek esetében úgy képzelhetjük el, hogy a rajz egyes részeit különálló, teljesen átlátszó pauszra készítjük el. Ezeket külön-külön is megtekinthetjük. Egymásra rakva az átlátszó fóliákat, láthatjuk a teljes rajzot is.

5. fejezet

45. Milyen lépések szükségesek a vektorgrafikus objektumok képének a monitor képernyőjén való megjelenítéséhez?

A vektorgrafikus objektumok képernyőn történő megjelenítéséhez a következő lépésekre van szükség:

- Először azt kell eldönteni, hogy a 3D-s végtelen modelltér melyik részét akarjuk a képen látni. A legtöbbször használt eljárás esetén egy kivágó testet hozunk létre, amely a képernyőn szerepeltetni kívánt objektumokat tartalmazza.
- Ezután a vektorosan kezelt grafikus objektumokat képi megjelenítésükhöz le kell képezni perspektívavetítéssel egy kétdimenziós nézetre, hasonló módon, mint ahogy a fényképezőgép objektíve előállítja a képet a filmkockán.
- A nézetet a képernyőn történő megjelenítéshez ezután vektoros modellből raszteres képpé kell konvertálni, és a frame bufferbe be kell tölteni.
- A frame buffer adatait olvasva állítja elő a monitorvezérlő kártya a jelet, amely vezérli a monitort a kép kirajzolása során. Analóg monitor esetén a monitorvezérlő kártya digitális-analóg konvertere állítja elő a kép kirajzolását végző analóg jelet.

46. Mit nevezünk jelenetnek (scene)? Mit adunk meg a KAMERA paranccsal? Hogyan lehet a vektorgrafikában megvilágítani a modelltér objektumait? Mit jelent a renderelés?

A modelltérnek egy adott nézőpontból látható, és a képi megjelenítés szempontjából összetartozó objektumait **jelenetnek (scene)** nevezzük.

A vektorgrafikus rendszerekben egy jelenet képének meghatározásához szükséges nézőpontot és irányt a világkoordináta-rendszerben általában a **KAMERA** paranccsal definiáljuk. A kamera állásától függően a megfelelő raszteres képek eltérhetnek

Különösen a fotorealisztikus ábrázolásnál fontos, hogy a modelltérben lévő objektumok megvilágítását meghatározzuk. Ezt megtehetjük például azzal, hogy a világkoordináta-rendszerben meghatározott helyű, erősségű és színű fényforrásokat definiálunk. A modelltér egy jelenetéből a raszteres kép előállítását **renderelésnek (rendering)** nevezzük. A fogalom a latin reddere = áthelyez, fordít, átalakít szóból származik, és ilyen értelemben a modelltér objektumainak digitális képpé való átalakítását jelenti.

47. Milyen tipikus formái vannak a vektorgrafikus objektumok megjelenítésének, és mik azok jellemzői?

A modelltérben definiált vektorgrafikus objektumok jeleneteinek megfelelő képek előállításához azt is rögzíteni kell, hogy a képen az objektumok szemléltetése milyen formában történjen. A vektorgrafikus szoftverek erre vonatkozó parancsai általában a következők:

- wireframe: huzalvázas megjelenítés: legegyszerűbb, ugyanakkor a legkevésbé valósághű. Ezesetben a testeket éleikkel ábrázoljuk, azaz a testeket a képen úgy jelenítjük meg, mintha drótból készültek volna. Az ábrán nincsenek takart vonalak, minden él teljes egészében megjelenítésre kerül
- net: poliéder közelítésű vonalhálós megjelenítés;
- hidden: takartvonalas palástmodell;
- shade: felületárnyalt testmodell: a képernyőre nem az objektumok átlátszó vázát, hanem határoló felületeik színnel vagy felületi mintázattal kitöltött képét rajzoljuk ki. Az árnyalással (shading) azt próbáljuk kifejezni, hogy a természetben látható megvilágított felületek a fényforrások és a felület térbeli helyzetéből és anyagi minőségéből, valamint a néző elhelyezkedésétől függően különböző megvilágítottságúnak látszanak. Ekkor már a nézőpont és az objektumok modelltérbeli elhelyezkedése alapján a kirajzolásnál a takarási viszonyokat is figyelembe vesszük. Ez azt jelenti, hogy a képen egy test vagy felület által eltakart élek és testrészek nem fognak megjelenni.

48. Milyen követelményeknek kell megfelelnie egy 2D-s képnek, hogy térhatású legyen? A felületeknek milyen tulajdonságait kell figyelembe venni realisztikus képek készítésekor? Mit kell figyelembe venni az átlátszóság modellezésénél?

A fotorealisztikus megjelenítés egyik fontos követelménye, hogy a 3D-s modelltér jelenete a 2D-s raszteres képen is térhatású legyen. A térhatás elérésének (depth cueing) legfontosabb eszközei a következők:

- A nézőponttól távolodva a képen a párhuzamos egyeneseknek fokozatosan összetartóknak kell lenniük, és a távolabbi testek méreteit arányosan csökkenteni kell. Ezeket a hatásokat a megjelenítés során megfelelő perspektíva transzformációkkal érhetjük el.
- A 2D-s képen szerepeltetett jelenet objektumai egy adott nézőpontból szemlélve takarhatják egymást. Fontos tehát, hogy a képen reálisan ábrázoljuk a tárgyak látható és nem látható éleit, felületeit.
- A nézőtől távoli objektumok már nem látszanak olyan tisztán, mint a közeliek. Ezért a képen a „messzeségbe tűnő" tárgyaknak elmosódottabbaknak és kevésbé részletesen kidolgozottaknak kell lenniük. Ezt a hatást elérhetjük a színintenzitások megfelelő változtatásával, illetve a textúráknak a nézőponttól való távolság függvényében történő alkalmazásával (például mip-mapping).

A realisztikus képek készítése során a látható felületeket saját tulajdonságaiknak (szín, fényvisszaverőképesség, az anyaguknak megfelelő felületi részletek) és a modelltérben elhelyezett fényforrásoknak megfelelően kell árnyalnunk.

Az átlátszóság modellezésénél figyelembe kell vennünk a fénytörést, a diffúz áttetszőséget és a fény intenzitásának csökkenését az átlátszó testen történő áthaladás során.

49. Adja meg a modelltér jeleneteiből a raszteres kép előállításának lépéseit! Mit értünk képgenerálási pipeline alatt?

- Amennyiben a modelltér elemeit már definiáltuk, akkor a képelőállításhoz még:
- Meg kell adni a modelltér objektumait megvilágító fényforrásokat a világkoordináta-rendszerben.
- Rögzítenünk kell a nézőpontot vagy a kameraállást a világkoordináta-rendszerben, ahonnan a jelenetet szemléljük.
- El kell döntenünk, hogy a modelltér milyen objektumait kívánjuk szerepeltetni a generálandó képen. Ehhez egy ablakot (window) kell definiálni a világkoordináta-rendszerben, amelyen keresztül a nézőpontból a jelenetet látjuk. Azok az objektumok, amelyek ezen az „ablakon" kívül esnek, nem vesznek részt a képgenerálásban, azokat kivágjuk. A megjelenítésnek ezt a lépését ablakozásnak és kivágásnak (windowing and clipping) nevezik a szakirodalomban.
- A modelltérnek a jelenetben szereplő objektumait a világkoordináta-rendszerből affin és perspektív transzformációval egy normalizált ábrázolási térbe kell leképeznünk.
- Az ábrázolási térben meg kell határozni az objektumok takarási viszonyait, azaz a nézőpontból látható éleket és felületeket. Ez a látható kép meghatározó algoritmusokkal történik.
- A látható felületelemek képpontjaihoz ezt követően az árnyalási algoritmusokkal hozzárendeljük a fényviszonyoknak és a textúráknak megfelelő színeket.
- A raszteres képernyőn a kiválasztott ablaknak megfelelő pixelekre "vetítjük" a felületelemek képpontjainak színértékeit a monitor fizikai eszközkoordináta-rendszerében.

A vektorgrafikus objektumok definiálásával kezdődő és a képernyőn való megjelenítésükig tartó teljes folyamatot a szakirodalomban **képgenerálási pipelinenak** (viewing pipeline vagy rendering pipeline) nevezik.

50. Mivel közelítjük a görbéket és felületeket a megjelenítéshez? Mi a tesszelláció?

A raszteres kép előállítása során tehát a térbeli görbéket mindig egyenes szakaszokkal, a felületeket pedig sokszöglapokból – legtöbbször háromszögekből – felépített alakzatokkal közelítjük. A modelltérbeli felületek sokszögekre bontásának folyamatát **tesszellációnak** (tessellation) nevezzük.

51. Hogyan definiálhatunk egy kamerát? Hogyan állítjuk elő a jelenet képét az ábrázolósíkban? Hogyan transzformáljuk a világkoordináta-rendszerben definiált objektumokat a normalizált látótérbe?

A kivágás és az ábrázolási térbe történő leképezés lényegét legegyszerűbben egy, a modelltérben elhelyezett fényképezőgéppel szemléltethetjük. A jelenet nézőpontjának definiálásához nyilván meg kell adni annak a pontnak a koordinátáit a világkoordináta-rendszerben, ahova a fényképezőgépet elhelyezzük. Ez azonban a pontos képalkotáshoz még kevés, szükség van a fényképezőgépben lévő film síkjának és az objektív térbeli irányának meghatározására is.

A jelenet képét tehát az (u, v) ábrázolósíkban definiált ablakra történő vetítéssel állítjuk elő. Ez egy középpontos vetítés, amellyel perspektivikus térhatást érhetünk el a keletkező képen.

Miután meghatároztuk a megjelenítő algoritmusok által figyelembe veendő látótért, arra a kérdésre kell keresnünk a választ, hogy az x, y, z világkoordináta-rendszerben definiált objektumok koordinátái hogyan írhatók fel az u, v, n ábrázolási koordináta-rendszerben. (Azaz azt, hogy az u, v, n koordináta-rendszerben hol helyezkednek el a modelltér elemei.) Ehhez egy affin koordináta-transzformációt kell végrehajtanunk, amely az x, y, z koordináta-rendszert forgatással és eltolással átviszi az u, v, n koordináta-rendszerbe.

Ezzel azonban még mindig nem fejezhetjük be a kivágást és az ábrázolási térbe történő transzformálást, mivel különböző esetekben a látótestek mérete és formája még eltérő. (Ha ezen megfelelő egységesítéssel nem tudnánk változtatni, akkor a megjelenítési algoritmusoknak a legkülönbözőbb méretű látótestek kezelésére is fel kellene készülniük.) Ezért egy további affin koordináta-transzformációval a felhasználásspecifikus látóteret egy normalizált látótérbe transzformáljuk.

52. Milyen célt szolgál a w-clipping algoritmus? Mi célt szolgálnak az outkódok a Cohen–Sutherland kivágó algoritmusban?

A figyelembe veendő látótér meghatározása után, a kivágás utolsó lépéseként meg kell határoznunk, hogy a modelltér objektumai közül melyek esnek bele a látótérbe. Ehhez speciális kivágóalgoritmusokat használunk, amelyek közül legismertebb az ún. Cohen–Sutherland w-clipping.

A Cohen–Sutherland w-clipping esetében a síkot az ablakkal együtt 9 részre osztjuk fel az alábbi ábra szerint.

Az outkódok szerint a szakaszokat a következő módon osztályozhatjuk:

- a szakasz az ablakon belül helyezkedik el, ha végpontjainak outkódja = 0,
- a szakasz teljesen az ablakon kívül helyezkedik el, ha végpontjai outkódjának logikai „ÉS” művelettel képzett eredménye nem 0,
- az ettől eltérő esetekben kiszámítjuk a szakasz és az ablakhatárok metszéspontjait, és "eliminájuk" a szakasz ablakon kívüli részeit.

53. Mi a tárgypontosság algoritmusok lényege? Mi a képpontosság algoritmusok lényege?

A **tárgypontosság-algoritmusok** az objektumokat az adott nézőpontra vonatkoztatva közvetlenül hasonlítják össze. Ezek pontosságának csak a gépi számábrázolás szab határt.

A **képpontosság-eljárások** a megjelenítendő kép pixeljei szerint határozzák meg a jelenetek látható részeit. Ezek pontossága nyilvánvalóan függ a megjelenítés eszközének felbontásától is.

54. Mit nevezünk back-face cullingnak? Mennyivel csökkenti a back-face culling a láthatósági algoritmusok műveletigényét?

Ha a poligonok normálvektorait úgy állítjuk be, hogy az objektumból kifelé mutassanak, akkor azok a poligonok, amelyek normálvektorai nem a néző felé mutatnak, biztosan takarva lesznek az objektum közelebbi felületei által. Ezeket a takarásba kerülő, úgymond hátrafelé néző poligonokat back-face poligonoknak nevezzük, és az objektumokat leíró adatbázisból való eltávolításukat eredményező eljárást pedig **back-face cullingnak**.

Egy poligonális objektumot metsző vetítősugár általában annyi back-face poligonon megy át, mint ahány front-face poligonon. A back-face poligonok eltávolítása tehát körülbelül megfelezi a képpontosság algoritmusok által egy-egy pixel esetén megvizsgálandó poligonok számát. Átlagos esetben a jeleneteket alkotó poligonok nagyjából fele hátrafelé néz, így a back-face culling a tárgypontosság algoritmusok által vizsgálandó poligonok számát is nagyjából megfelezi.

55. Ismertesse, hogy mire használhatók a befoglaló testek (bounding volume)?

A térbeli objektumok takarási viszonyainak meghatározását esetenként segítheti, ha ezeket egyszerűbb formájú testekbe zárjuk, amelyek térbeli elhelyezkedését hatékonyabb algoritmusokkal tudjuk tesztelni. Ezeket **befoglaló testeknek (bounding volume)** nevezik. A leggyakrabban alkalmazott befoglaló test a téglatest, de esetenként a céloknak jobban megfelelhet a henger vagy a gömb.

56. Ismertesse, hogy mi a min/max teszt alapgondolata?

A **min/max** eljárás a kivágóablak (x, y) síkjában az objektumok vetületeit teszteli. Ennél az objektumok vetületeit téglalapokkal vesszük körül, és ha ezeknek a téglalapoknak nincs közös része, akkor biztosak lehetünk abban, hogy a vetületeknek megfelelő felületek sem fedik egymást.

57. Ismertesse a mélység rendező algoritmus lényegét!

Ennek a tárgypontosság algoritmusnak az az alapgondolata, hogy a megjelenítendő objektumokat a nézőponttól való távolság függvényében sorba kell állítani (**depth-sort**). Az algoritmus a helyes takarási viszonyokat úgy alakítja ki, hogy a nézőhöz közelebb eső objektum képe felülírja a távolabbi objektum képét.

58. Milyen célra használhatók a BSP-fa algoritmusok? Ismertesse a BSP-fa algoritmusok alapgondolatát!

A háromdimenziós objektumok tetszőleges nézőponthoz tartozó megjelenítésére használt algoritmusok közül az egyik leghatékonyabb a bináris térfelosztó fákat (**BSP-fák, BSP = Binary Space Partitioning**) alkalmazó eljárás.

A BSP-fa algoritmusok azon alapulnak, hogy az ábrázolandó jeleneteket úgy is tekinthetjük, mint objektumcsoportok gyűjteményét. Ha található olyan sík, amely elválaszt egymástól két objektumcsoportot, akkor az a csoport, amelynek az oldalán a nézőpont is van, eltakarhatja a másik csoportot, de a másik csoport által sohasem kerülhet fedésbe.

59. Mi a területfelosztó algoritmusok alapgondolata?

A területfelosztó algoritmusok alapgondolata, hogy néhány esetben (például a képen csak egy objektumot kell ábrázolni), nagyon egyszerűen megállapítható a képernyőn megjelenítendő kép, ezért a bonyolultabb takarási vizsgálatokat célszerű a kép területének kisebb részekre való rekurzív felosztásával visszavezetni az egyszerűen kezelhető esetekre.

60. Milyen puffereket használ a z-buffer algoritmus? Ismertesse a z-buffer algoritmus működését! Mennyiben függ a z-buffer algoritmus a modelltér objektumainak alakjától?

Az algoritmus két tárolóterületet használ:

- a frame-buffert, amely a képernyő pixeljeihez rendelt színértékeket tárolja, induló feltöltése a képernyő háttérszíne,
- a z-buffert, amely az egyes pixelekhez rendelt z értékeket tárolja a normalizált látótérből, kezdeti értéke a hátsó kivágósík z koordinátája (a maximálisan ábrázolható z-érték).

Az algoritmus lényege tehát az, hogy egy raszterponthoz tartozó vetítő sugáron kiválasztjuk az objektumoknak a nézőponthoz legközelebb fekvő döféspontját.

A z-buffer algoritmus a modelltér elemeinek formájától független, így háromszög, poliéder vagy görbült felületek láthatóságának megállapítására egyaránt alkalmas. Alkalmazhatóságának egyetlen feltétele, hogy az objektumok felületi pontjaiban a nézőponttól való z távolság és az árnyalási információk (szín, megvilágítás, textúrák) meghatározhatók legyenek.

61. Mi a scan-line algoritmusok lényege? Hogyan határozzák meg a kirajzolandó pixeleket a scan-line algoritmusok?

A **scan-line algoritmusok** képpontosság műveleteket használva, pixelsorokként készítik a képet, és a poligonok (háromszögek) frame bufferbe történő soronkénti konvertálásának módszerén alapulnak. Megpróbálják az egymást takaró háromszögek vizsgálatát csak a legszükségesebbekre csökkenteni.

A pixelsor és a háromszögek közös részét szegmenseknek nevezzük. Ha a pixelsor egy szegmense csak egy háromszöghöz tartozik, akkor ezt egyszerűen csak meg kell jeleníteni. Ha egy pixel több szegmenshez tartozik, akkor a megfelelő háromszögek mélységi vizsgálatával kell eldönteni, hogy melyik háromszög felületi pontját kell kirajzolni.

62. Ismertesse a sugárkövetéses algoritmust!

A **sugárkövetéses (raytracing)** algoritmusok a felületek láthatóságát a nézőpontból kiinduló, képzeletbeli fénysugarak követésével határozzák meg. Tipikus képpontosság-algoritmusok. Működésükhöz a nézőpontot és egy tetszőleges vetítési síkon felvett ablakot kell definiálni. Az ablak téglalap alakú területét felosztó szabályos rács a képernyő pixeljeinek felel meg.

A raytracing algoritmussal a nézőpontból vetítősugarat indítunk az ablak minden egyes pixelén keresztül a jelenet objektumai felé. Az adott pixel színét a sugár által a nézőponthoz legközelebb metszett objektum színe határozza meg.

63. Mi a szerepe a megvilágítási modelleknek a vektorgrafikában? Mitől függ a modelltér egy objektuma felületi pontjának színe? Milyen hatása van a renderelési időre, ha egy képelőállításnál több megvilágítási modellt is figyelembe veszünk? Hogyan kapjuk meg egy pixel végleges színét?

A **megvilágítási modellekkel** írjuk le a modelltér objektumainak és a fényforrásoknak a kapcsolatát.

A modelltér egy objektumának felületén látható színárnyalatokat a következő tényezők befolyásolják:

- a fényforrás típusa (pontszerű, kiterjedt stb.),
- a fényforrás sugárzásának erőssége és színe,
- a felület fényvisszaverő képessége,
- a felület és a fényforrás távolsága,
- a felület normálisa és a fénysugár által bezárt szög.

Általában egy realisztikus fényviszonyokat tükröző kép megjelenítéséhez több modellt is felhasználunk, viszont minél többet veszünk figyelembe, a képelőállításnak annál nagyobb az erőforrásigénye (azaz annál lassúbb lesz a renderelés).
Egy felületi pontnak megfelelő pixel végleges színét úgy kapjuk meg, hogy a test eredeti alapszínéhez az egyes megvilágítási modellek alapján számított módosító világosság és színértékeket hozzáadjuk.

64. Sorolja fel a fontosabb fényvisszaverődési típusokat és főbb jellemzőiket!

A fontosabb fényvisszaverődési típusok a következők:

- diffúz fényvisszaverődés (diffuse reflection), amely a matt felületekre jellemző;
- tükröző fényvisszaverődés (total reflection), amely tükörként viselkedő felületekre jellemző;
- csillogó (fénylő) fényvisszaverődés (specular reflection), amely fénylő felületekre jellemző;
- átlátszóság (transparency), amely olyan testekre jellemző, amelyeken a fénysugár teljes egészében vagy részben áthalad;
- árnyék (shadow)

65. Mit nevezünk lokális és globális megvilágítási modellnek? Mutassa be az algoritmusok lényegét!

A lokális megvilágítási modellekben az objektumok színét, világosságát más objektumok nem befolyásolják, ezek csak az objektumtól, a fényforrásoktól és a nézőponttól függnek. Ilyenek a háttérmegvilágítás, valamint a diffúz és csillogó fényvisszaverődés.
A globális megvilágítási modellekben az objektumok színe nem csak közvetlenül a fényforrásoktól, hanem a más objektumokon áthaladó, illetve a más objektumok által visszavert fénytől is függ. A modelltér objektumainak kölcsönhatását figyelembe kell venni a tükröződő fényvisszaverésnél, az árnyékképzésnél és az átlátszóság kezelésénél.
A tananyagban bemutatott lokális megvilágítási modellek (local illuminations):

- z-buffer
- scan line
- flat shading
- Gouraud-shading
- Phong-shading.

A tananyagban bemutatott globális megvilágítási modellek (global illuminations):

- radiosity
- rekurzív sugárkövetés
- path tracing
- light tracing
- bi-directional path tracing
- Metropolis light transport
- photon mapping
- ambient occlusion
- image based lighting.

66. Mit jelent a háttérfény (ambient light)? Mi jellemzi a diffúz fényvisszaverődést? Mi az ideálisan tükröző visszaverődés? Mi az irányított diffúz visszaverődés? Mikor kell alkalmazni a Phong-féle megvilágítási modellt?

A **háttérfény** lokális megvilágítási modellben nincs fényforrás, az objektumok "saját" fényüket bocsátják ki. Ez megfelel annak, hogy a jelenetet egy irányfüggetlen, szórt fény (ambient light) világítja meg.

A **diffúz visszaverődés** (diffuse reflection) a matt, illetve durva felületek jellemzője, ezek a Lambert-féle megvilágítási törvényekkel modellezhetők. Ez esetben a visszavert fény sűrűsége független a kilépő sugár irányától, tehát a felület minden szögből egyformán fényesnek tűnik.

Az **ideálisan tükröző visszaverődést** (total reflection) tapasztalhatjuk például egy síktükör esetében. Ebben az esetben a fénysugár a geometriai optikából jól ismert törvénynek megfelelően halad: a belépő és visszavert sugár, valamint a felületi normális egy síkban helyezkedik el, és a belépő és visszavert sugár a tükröző felület normálisával azonos szöget zár be.

Az olyan fényvisszaverődést, amelynek van egy kitüntetett iránya, amely irányban a felület a legtöbb fényt veri vissza, majd ahogy ettől az iránytól távolodunk, a visszavert fény mennyisége ennek megfelelően csökken, **irányított diffúz visszaverődésnek** (specular reflection) nevezzük.

Az irányított diffúz visszaverődés tapasztalati megfigyelése alapján javasolta Phong a róla elnevezett megvilágítási modellt tökéletlenül tükröző testek számára.

67. Miért egyszerűbb poliéderek megvilágítását kiszámolni? Mit nevezünk shadingnek?

Ha a test határolófelülete poligonokból áll, akkor (feltéve, hogy fényfolthatással nem kell számolnunk) elegendő a poligon egy pontjához tartozó szín és intenzitásértéket meghatározni, és ezt követően a poligont ezzel a színnel feltölteni. Így ebben az esetben a megvilágítás kiszámításának műveleti igénye nem a felületi pontok számával, hanem a testet borító poligonok darabszámával lesz arányos.

Az objektumok felületeinek színét meghatározó eljárásokat árnyalási algoritmusoknak (**shading**) is szokták nevezni a szakirodalomban.

68. Mi a flat shading lényege? Milyen megvilágítási modellt használunk a flat shadingnél?

Az eljárás a szórt háttérvilágítás és a diffúz visszaverődés együttes megvilágítási modelljét használja, ezért egy felületi sokszög színének meghatározásához elegendő a poligon normálvektorának kiszámítása (lásd a megvilágítási egyenletet) és a fényforrás jellemzőinek figyelembevétele. Ez alapján számítjuk ki poligononként azt az egy értéket, amely meghatározza a sokszög színét.

Ezt csak a következő két esetben várhatjuk:

- a fényforrás „végtelen” távol van,
- a test valóban egy sokszöglapokkal határolt objektum, és nem egy görbült felület sokszögközelítése.

69. Mi a Gourand shading lényege? Ismertesse az intenzitás-interpoláló árnyalás lépéseit!

Realisztikusabb képek érdekében dolgozták ki az interpolált árnyalással működő algoritmusokat, amelyek a közelítő poligonokat (háromszögeket) nem azonos színnel árnyalják, hanem a sokszög pontjainak intenzitásértékét a csúcspontok intenzitásértékeiből lineáris interpolációval származtatják, ezáltal az éles intenzitáskülönbségeket elsimítják.

A Gouraud shading, vagy más néven intenzitás-interpoláló árnyalás lépései háromszög közelítés esetén a következők:

- Minden csúcspontban, ahol háromszögek találkoznak, kiszámítjuk a találkozó lapokhoz tartozó normálisok átlagát, így kapunk csúcspontonként egy ún. „pszeudo-normális”-t (lásd ábra).
- A szórt háttérvilágítás és a diffúz visszaverődés megvilágítási modellje alapján kiszámítjuk a modelltérben a csúcsok intenzitásértékét a pszeudo-normálisból.
- Vetítjük a háromszögeket a képsíkra, és itt a csúcsok intenzitásértékeit lineárisan interpoláljuk az élekre és a lapokra.

70. Mi a lényege a Phong shadingnak? Mi a legfontosabb különbség az előállított képen a Phong- és a Gourand-árnyalás között?

A normálvektor-interpoláló árnyalás néven is ismert Phong-árnyalás a felületi normálvektorokat interpolálja, nem pedig az intenzitásértékeket.

Specular visszaverődési modell esetén jelentős különbség van a Gouraud- és a Phong-árnyalás között, az utóbbi sokkal élethűbben adja vissza a felületeken megjelenő fényfoltokat. Bár a Gouraud- és a Phong-modell közötti különbség leglátványosabb megnyilvánulása a fényfoltok megjelenítésében tapasztalható, általában is kedvezőbb eredményeket kapunk ez utóbbi algoritmus használatakor, mivel ez minden ábrázolandó ponthoz viszonylag jól közelítő normálvektort állít elő. Viszont a normálvektorok kiszámítása miatt az algoritmus időigénye számottevően növekszik.

71. Mi a radiosity és photon mapping eljárások közös jellemzője? Melyik fotorealisztikus képábrázoló algoritmus modellezi hatékonyabban a tükröződést, illetve a diffúz visszaverődést?

A fényterjedés fizikai törvényeit (fénysűrűség egyenlet, fotonok mozgása) "szimuláló" algoritmusok (radiosity, photon mapping) ezzel szemben teljesen szétválasztják a látható felületek meghatározását és azok árnyalását. Az első, nézőpont független fázisban az objektumok és a rájuk eső fény összes kölcsönhatását képesek modellezni, majd ezt követően a hagyományos látható felületmeghatározó és interpolációs technikákkal készítik el a megadott nézőpontból a képet.
A nézőpontfüggő raytracing algoritmusok könnyedén kezelik a tükröződéseket, de egyes diffúz jelenségek kezelése (például kiterjedt világítótestek) nehézséget okozhat. Ezzel szemben a nézőpontfüggetlen algoritmusok hatékonyan modellezik a diffúz jelenségeket, de a tükröződés figyelembevétele esetenként csak külön eljárásokkal oldható meg.

72. Melyek a raytracinggel készített képek fontosabb jellegzetességei? Mi a rekurzív sugárkövetés lényege? Ismertesse a rekurzív sugárkövetés algoritmusát! Mit értünk backward raytracing alatt? Mikor lesz egy pixelnek elsődleges színe a sugárkövetéses algoritmusnál?

A **raytracing** algoritmussal készített képek jellegzetességei könnyen felismerhetők. Ezek:

- objektumok egymásra vetett árnyékai,
- többszörös tükröződések,
- átlátszóság kezelése fénytöréssel.

A **rekurzív sugárkövetés** algoritmusának alapelve egy mondatban megfogalmazható: a 3D-s modelltérben definiált objektumok és fényforrások alapján a nézőpontból a geometriai optika törvényei szerint követve a többszörösen visszaverődő és megtörő fénysugarakat egy fotóminőségű képet számolunk ki és jelenítünk meg.

A fény terjedésével ellentétben nem a fényforrásokból kiinduló sugarakat kezdi vizsgálni, hanem azokat a nézőpontból indított vetítősugarakat elemzi, amelyek visszafelé eljutnak a fényforrásba. Ezt a megoldást „hátrafelé történő sugárkövetésnek", vagy **backward raytracingnek** nevezik.

Ha egy elsődleges sugár visszaverődve egy felület egy pontjáról „eltalál" egy fényforrást, és nem kell tükröződést illetve fénytörést figyelembe venni, akkor az elsődleges sugár képzésénél felhasznált cellának megfeleltetett pixel színe a megvilágítási modellből közvetlenül számítható. Ez azonban a pixelnek csak **elsődleges színe** lesz, mert előfordulhat, hogy a szóban forgó felületi pontot az algoritmus későbbi lépéseiben újabb sugár is érinteni fogja.

73. Milyen célt szolgálnak az árnyéksugarak? Miként alkalmazzuk a befoglaló testeket a raytracing során?

Ha a fényforrással összekötő sugarak (ezekből annyit kell indítani, ahány fényforrás van definiálva a modelltérben), amelyeket **árnyéksugaraknak** (shadows ray) is neveznek, beleütköznek valamilyen objektumba mielőtt a fényforrást elérnék, akkor a felületi pont árnyékban van, és a megfelelő fényforrást ki kell hagyni a felületi pontot árnyaló megvilágítási egyenletből.

A raytracing leginkább műveletigényes része a vetítősugarak és az objektumok metszéspontjainak kiszámítása, tehát célszerű ezek számát csökkenteni.
Ezt célozza a raytracing során a **befoglaló testek** (bounding volumes) alkalmazása. Ennek során legtöbbször gömböket használunk fel. Ez azt jelenti, hogy olyan legkisebb méretű gömböket definiálunk a modelltérben, amelyek egyes objektumainkat teljes mértékben tartalmazzák.

74. Jellemezze a radiosity algoritmus felhasználhatóságát és fizikai elvét! Mely esetben lassú, és mikor gyors a radiosity eljárás? Hogyan kezeli a fényforrásokat a radiosity eljárás?

A radiosity eljárás az egyik legmodernebb renderelési módszer, amelyet a felületek közötti fényenergiacsere fizikai törvényszerűségeire alapozva dolgoztak ki. Ezért ezzel a módszerrel a kiterjedt fényforrások és a diffúz visszaverődés teljes értékűen, egzakt módon modellezhető. Emiatt a radiosity eljárással teljesen valósághű, fotóminőségű képeket készíthetünk a modelltér jeleneteiről.
A radiosity eljárás alapja a fizikából ismert általános fénysűrűség egyenlet, amelyből bizonyos egyszerűsítésekkel határozzák meg a felületek fénysugárzását. Emiatt a szakirodalomban a radiosity eljárást a fénysűrűséggel renderelésnek is szokták nevezni (rendering with radiance).

A radiosity algoritmus alkalmazása esetén először a jelenet összes objektumának felületére meghatározzák a visszavert és a felület saját sugárzásából adódó fénymennyiséget, amelyből származtatható a globális megvilágítási modell. Ennek erőforrásigénye, és így a gépidőszükséglete általában a többi renderelési algoritmushoz képest többszörös. A fényviszonyokat ugyanakkor csak egy alkalommal kell kiszámítani. Így például egy nézőpont (kameraállás) megváltoztatása esetén – feltéve, hogy a jelenethez tartozó fényforrások és objektumok helyüket nem változtatják – az új kép a megismert egyszerű láthatósági algoritmusokkal már nagyon gyorsan kiszámítható a változatlan megvilágítási modell miatt. (Ez lényegesen gyorsabb, mint egy raytracelt kép előállítása.)

A radiosity eljárásnál viszont minden felületet fénysugárzónak tekintünk, és a fényforrások sem pontszerűek, hanem véges kiterjedésű felületekkel lesznek modellezve.

75. Mit lehet modellezni a photon mapping eljárással? Foglalja össze a photon mapping algoritmus lényegét! Mit tartalmaz a fotontérkép? Hogy számítjuk ki egy pont fényintenzitását a photon mapping alkalmazása esetén?

Ezzel a globális megvilágítási eljárással fotorealisztikusan modellezhetők a közvetett megvilágítás hatásai, valamint az úgynevezett kausztikus optikai jelenségek.
A fotonkövetés alapgondolata, hogy minden fényforrásból a fényforrás intenzitásának megfelelő számú és energiájú fotont lőnek be a modelltérbe, véletlenszerű irányokba. Ezek az objektumok felületével „ütköznek" és visszaverődnek, megtörnek vagy elnyelődnek, amelynek során a fotonok energiája is megváltozik.
A fotonok útját a fényforrásból kiinduló nyomkövetéssel (path tracing) kísérjük figyelemmel, és adataikat ún. foton térkép (photon map) segítségével tartjuk nyilván. (Meg kell jegyezni, hogy ez természetesen kvantummechanikai szempontból nem jelent korrekt szimulációt.)
A globális megvilágítás meghatározását követően a raszteres kép előállítása általában a raytracing renderelő algoritmussal történik, minden felületi pont kiszámításánál figyelembe véve a pontot elérő fotonok energiáját.

76. Miért használhatunk fel látható felület meghatározó algoritmusokat az árnyékok meghatározására? Sorolja fel az árnyékképzéshez felhasználható egyszerűbb algoritmusokat! Hogyan határozhatjuk meg az árnyékokat z-buffer algoritmussal?

A láthatósági algoritmusok azt határozzák meg, hogy mely felületek láthatók a nézőpontból, az árnyékokat előállító algoritmusok pedig azt, hogy mely felületek „láthatók" a fényforrásból. Így a látható felület meghatározó és az árnyékképző algoritmusokat lényegében azonosnak tekinthetjük.

Az egyszerűbb árnyékképző algoritmusok a már megismert eljárásokat alkalmazzák:

- scan-line;
- árnyéktestek használata;
- z-buffer

A z-buffer algoritmussal az alaptechnika kétszeri alkalmazásával (a nézőpontra és a fényforrásra) is lehetőség van árnyékban levő felületek meghatározására. Az algoritmus a fényforrást használva vetítési középpontként, kitölt egy z-buffert, amelyben a fényforrástól mért távolságot rögzítik. Ezután a korábban megismert z-buffer algoritmus meghatározza a látható felületi pontokat, de az egyes pontok árnyalásakor a pontok térbeli koordinátáit áttranszformálja a fényforrás középpontú rendszerbe, és a z koordinátákat összeveti az árnyék z-bufferben lévő távolságértékekkel. Ha a transzformált pont z koordinátája nagyobb, mint a bufferben tárolt érték, akkor a vizsgált pont távolabb van a fényforrástól, tehát árnyékban van.

77. Hogyan használhatjuk a felületi részlet poligonokat? Mit nevezünk textúrának és texelnek? Hogyan tudjuk figyelembe venni a textúrákat a képernyő-koordinátarendszerben?

Egyszerűbb grafikus rendszerekben az objektumainkra jellemző fontosabb részleteket (például egy épület falán levő ajtókat, ablakokat, feliratokat stb.) legegyszerűbben újabb poligonok, ún. felületirészlet-poligonok felhasználásával tudjuk megjeleníteni. Ezeket a poligonokat egyszerűen felvisszük a bázisfelület megfelelő oldalára, és az árnyalás során ezek a poligonok egyszerűen helyettesítik a bázisfelület megfelelő részét az általuk fedett területeken.
Kétdimenziós képek felületekre történő leképzésével a Catmull által kidolgozott textúrázás (texture mapping) néven ismert technika alternatív lehetőséget biztosít objektumaink megjelenítésének javítására. A művelet során felhasznált képet textúrának (texture map), egyes elemeit pedig texelnek hívjuk.

Textúrák alkalmazása esetén a felületnek megfelelő képernyőrész minden egyes pixelje esetén meg kell vizsgálni, hogy a hozzátartozó színértéket a textúrák megfelelő részei mennyiben módosítják. Ehhez többlépcsős leképezéssel egyértelmű kapcsolatot kell létesítenünk a textúra u, v koordinátarendszere és a képernyőkoordináta-rendszer között

78. Mit értünk algoritmikus textúra alatt?

Algoritmikus textúrák esetén a felületre feszített mintázatot programmal állítjuk elő. Erre jó példa egy fraktálok segítségével generált fűszálakból álló rét.

79. Mit értünk edge anti-aliasing és full scene anti-aliasing alatt? Mi a hatása a bilinear filteringnek? Mit jelent a mip-mapping? Hogyan lehet perspektivikus hatást elérni textúrákkal?

A textúraelemekből képzett ferde vonalak lépcsőzetes kialakulásának megakadályozását célozza az **edge anti-aliasing**. Ennek lényege, hogy a vonal széleihez tartozó texelek színét átlagoljuk a vonal melletti texelek színével, és ezt az átlagértéket jelenítjük meg a képen. Ezzel a "simító" eljárással a kapott átmeneti színek elmossák a képen a "lépcsőket". Ezt a technikát esetenként a teljes képre is alkalmazzák (**full scene anti-aliasing**), de ez a képet el is homályosíthatja, illetve ugrásszerűen megemelheti a képelőállítás erőforrásigényét.

Bilinear texture filtering: a textúra egy képpontjának színét a függőleges és vízszintes irányban mellette lévő 4 pixel színértékének átlagából számítjuk ki. Hatására a textúraelemek közötti éles átmenetek elmosódnak

Mip-mapping: ennél az eljárásnál egy textúrát több különböző felbontásban is letárolunk. Ha közeledünk a nézőponttal egy textúrázott objektumhoz, akkor a pixelesedés kizárása érdekében egyre finomabb felbontású változatát feszítjük fel a textúrának az objektum felszínére. Minél közelebb van egy objektum, annál részletesebb a textúra, a távolabbi tárgyak pedig elmosódottabbak.

Perspective correction: a térhatás elérése érdekében a textúrákat a távolság függvényében kisebbítve, ferdítve visszük fel a felületre

80. Hogyan modellezhetjük az átlátszó és áttetsző testeket? Mire és hogyan használható az alpha-blending? Milyen ismert algoritmus kezeli a törő átlátszóságot? Hogyan modellezhető a köd? Hogyan lehet modellezni a füstöt és a lángot?

Az átlátszóság és áttetszőség modellezésére a következő módszerek alakultak ki:

- törésmentes átlátszósági eljárások, amelyek lehetnek interpoláló, illetve szűrő algoritmusok,
- törő átlátszósági eljárások, amelyek a fénytörés törvényei szerint kezelik a testeken

áthaladó fénysugarakat.

Alpha blending: erre legjobb példa az áttetsző víz ábrázolása, amikor a víz felszínének textúráját részlegesen "átlátszóvá tesszük", és "mögötte" megjelenítjük a vízfenék textúráját. Ezt az eljárást alpha-blendingnek nevezzük, és az átlátszó objektumok modellezésére alkalmazhatjuk. Ehhez legtöbbször texelenként az RGB színértékek mellett az áttetszőség mértékét meghatározó α csatorna értéket (RGBA színkódolás) használjuk fel. A képernyőn megjelenítéskor az átlátszó textúra (az alphabitekben meghatározott arányban) és a mögöttes textúra színadatait átlagolják. Így például egy részben átlátszó vízfelületnél átmenetet képezhetünk a vízfelszín és a meder textúrája között.

Ködöt fotorealisztikus képeken úgy tudunk ábrázolni, hogy a ködnek megfelelő színt (általában fehér vagy fehéresszürke) a nézőponttól távolodva egyre nagyobb arányban keverjük a képhez.

Füstöt is modellezhetünk speciális textúrákkal. A füstnek megfelelően mozgatjuk az ábrázoló textúraelemeket, melyek részben átlátszóak, hatásuk elmosódott.
Hasonló módszerrel tudunk előállítani például **lángot** is.

81. Mit értünk objektumok közötti tükröződés alatt? Mi az enviroment mapping lényege? Mit értünk reflection mapping alatt?

Objektumok közti tükröződésről beszélünk, ha egy felületen a jelenet egy másik elemének képe visszatükröződik.
Az **environment mapping** eljárással textúrákkal modellezhetjük egy objektum környezetében lévő tárgyak visszatükröződését az objektum felszínén. Ezt azzal lehet elérni, hogy a visszatükrözött tárgy textúráját használjuk fel a tükröző felülettel rendelkező tárgy textúrájaként.
A tükörszerű visszaverődés modellezésére az ún. **reflection mapping** technikát dolgozták ki. Ennél a megjelenítendő tükröző objektum köré egy gömböt definiálnak, amelyre rávetítik az objektumot körülvevő modelltér képét. A gömb felületét ezután 2 dimenziós textúraként használják fel a tükröző felület képének kialakítása során. Alternatív megoldásként esetenként kockát használnak, és ennek oldalaira végzik el a vetítést. Természetesen a reflection mapping eljárás csak közelíti a valódi tükrözést, ennek ellenére sokszor jól használható realisztikus képeket lehet előállítani vele.

6. fejezet

82. Mi okozta, hogy a hardverportabilitás ma már általánosan elfogadott követelmény a számítógépes grafikában? Sorolja fel, hogy a szabványosítás milyen területekre terjed ki a számítógépes grafikában!

Kezdetben a grafikus szoftvereket eszközfüggően kellett programozni, ez azonban egyre kevésbé felelt meg az igényeknek. Egyrészt a felhasználók már nem fogadták el függő helyzetüket az eszközgyártóktól, másrészt a grafikus szoftvereket fejlesztő szoftverházaknak is elemi érdekévé vált, hogy az egyre bonyolultabb és növekvő fejlesztési költségekkel előállított programcsomagokat minél több hardverplatformon értékesíteni tudják.

Az egységesítés egyre több területre kiterjedt, ezek közül a fontosabbak ma a következők:

- grafikus felhasználói felület szabványai,
- grafikus programcsomag szabványok,
- a felhasználói programok és a grafikus programcsomagok közötti interfészt, az API-t (Application Program Interface) meghatározó szabványok,
- a grafikus rendszerek közötti adatcserét biztosító fájlszabványok,
- a színkezelés, a betűformák szabványai,
- a különböző hardvercsatolókra (BIOS, driverek) és grafikus eszközökre vonatkozó szabványok.

83. Mi az X-Windows System? Az X11 szabvány szerint mit tartalmaz a grafikus munkahely? Mi az X-szerver feladata? Mi a feladata a Windows Manager kliensnek?

Az **X-Windows System** hardverfüggetlen, a raszteres grafikára alapozott ablakozó megjelenítés szabványa.

A grafikus munkahely az X11 szerint képernyőt (screen), billentyűzetet (keyboard) és mutatóeszközt (egér, tablet) kezelő architektúra, amelyen az X-szerver program fut. (Felhívjuk a figyelmet, hogy a szabvány terminológiája eltér pl. a LAN-nál megszokottól.) Ez lehet például egy PC, amely az X-szervert futtatja, vagy egy X-terminál, amely beégetett vagy programként futó X-szervert tartalmaz.

Az **X-szerver** feladata a grafikus megjelenítés (2D-s rajzolás), és a felhasználó inputjainak a fogadása. Az X-szerver hálózati kliensekkel, az úgynevezett alkalmazásokkal (applications) tarthat kapcsolatot, amelyek számára hozzáférést biztosít a grafikus munkahelyhez.

Az X11 felhasználói felülete a Windows-ból jól ismert ablakozó rendszer. Az ablakok méretezését és általában a menedzselését a **Windows Manager** program végzi, mely az X-szerver számára szintén egy kliens.

84. Mi az SRGP? Milyen követelményeket határoztak meg a GKS szabvánnyal szemben? Mi a PHIGS? Mi az OpenGL? Milyen „magasabb szintű”, új renderelő funkciókat tartalmaz az OpenGL a korábbi szabványokhoz képest?

Ezt a szabványt egyszerű rasztergrafikus programcsomagnak, vagy az angol megnevezése után rövidítve SRGP-nek (Simple Raster Graphics Package) nevezik. Az SRGP tulajdonképpen a legalapvetőbb rasztergrafikus funkciókat megvalósító programegységekre vonatkozó szabvány, amely X11 alatt alkalmazható.

Az SRGP legfontosabb alkotóelemei a rasztergrafikus primitívek, ezek

- vonalak,
- ellipszisek,
- sokszögek és
- szövegek

lehetnek. A primitívek olyan képelemgenerátorok, amelyeket felhasználásukkor kell felparaméterezni (például kör esetén a középpont és a sugár megadásával).

A célkitűzések és követelmények, amelyeket a GKS szabvánnyal el akartak érni, a következők voltak:

- egységes illesztési helyet meghatározni a grafikus rendszerek és az egyedi alkalmazások között,
- a felhasználástól független, számítástechnikailag hatékonyan realizálható könyvtár definiálása a vektorgrafikus rendszerek fontosabb funkcióira,
- a grafika területén minél több általánosítható követelményt lefedni a szabvánnyal,
- elválasztani a grafikus rendszerek alap- (ún. mag-) funkcióit a magasabb szintű modellezési funkcióktól,
- a szabvánnyal egy fejlesztési irányt mutatni a készülékgyártók számára.

A **PHIGS** vektorgrafikus programcsomagszabvány létrejötte volt a normakészítők másik válasza a GKS-3D mellett arra a helyzetre, hogy a GKS első változata csak a 2D-s vektorgrafika magfunkcióit szabványosította. A PHIGS a szabvány angol elnevezésének (Programmers Hierarchical Interactive Graphics System) rövidítéséből származik. Ezek szerint a PHIGS a programozók hierarchikus, interaktív grafikus rendszere.

Az 1990-es években, különösen a 3D-s gyorsító chipek fejlődésével egyre nagyobb gondot okozott, hogy a hardverben megvalósított láthatósági, megvilágítási és árnyalási algoritmusokat a szabványok nem tartalmazták. A Silicon Graphics fejlesztői ezekre a problémákra kerestek megoldást, amikor a korábbi IRIS GL grafikus szoftverük tapasztalatainak felhasználásával egy szoftverinterfészt specifikáltak **OpenGL** (Open Graphics Library, 1992.) néven a grafikus kártyák hardveréhez.

Az OpenGL (a GKS-sel és PHIGS-szel szemben) ezen túlmenően a „magasabb szintű” renderelő funkciókat is tartalmaz. Ezek:

- texture mapping, mip mapping,
- alpha blending (átlátszóság),
- légköri effektek (köd, pára, füst),
- anti-aliasing.

85. Ismertesse az OpenGl rendering pipeline-jét! Hogyan adjuk meg a primitíveket az OpenGL-ben? Mikor történik a fényforrások hatásának a figyelembe vétele az OpenGL-ben? Mi történik a raszterizálás során az OpenGL-ben? Mire szolgálnak a képtöredékek (fragments) az OpenGL-ben? Milyen a feladatmegosztás az OpenGL és a Windows között?

A grafikus információkat a rendszerrel **csúcsponti adatok (vertex)** formájában közölhetjük. Ha ezek térbeli görbére vagy felületre vonatkoznak (tartópontok), akkor a **görbe és felületmodellező** ebből OpenGL primitíveket állít elő. A következő lépésben az OpenGL a primitíveket egy **vetítéssel leképezi a látótérbe**. Ebben a fázisban történik a fényforrások kihatásainak figyelembe vételéhez szükséges számítások elvégzése is. Az úgynevezett **raszterizálással** megtörténik az átalakítás 2D-s egész koordinátákra. Ekkor veszi a rendszer figyelembe a különböző bitmap és pixelmezőket, amelyek közé számítják a textúrákat is. A raszterizálás során kapott pontokhoz a színértékek és a pont z értéke is hozzá lesz rendelve. A raszterizálás után kapjuk a **képtöredékeket (fragments)**, amelyek alapján z-buffer eljárással határozzák meg a végső láthatóságot. Az OpenGL megengedi, hogy a frame bufferből adatokat kiolvassunk, illetve egyes tartományokat kimásoljunk.

A rendering pipeline utolsó lépését, a frame bufferből való képkirajzolást a monitor képernyőjére nem az OpenGL, hanem az alkalmazott ablakozó rendszer hajtja végre. Ez összhangban van az OpenGL alapkoncepciójával, amely szerint a konfiguráció és grafikus input készülék vezérlése a Windows feladata. Így például a Windows kezeli a képernyő ablakait, a színtáblázatot stb. (Az ezekkel kapcsolatos információkat viszont az OpenGL természetesen felhasználja a renderelés során.)

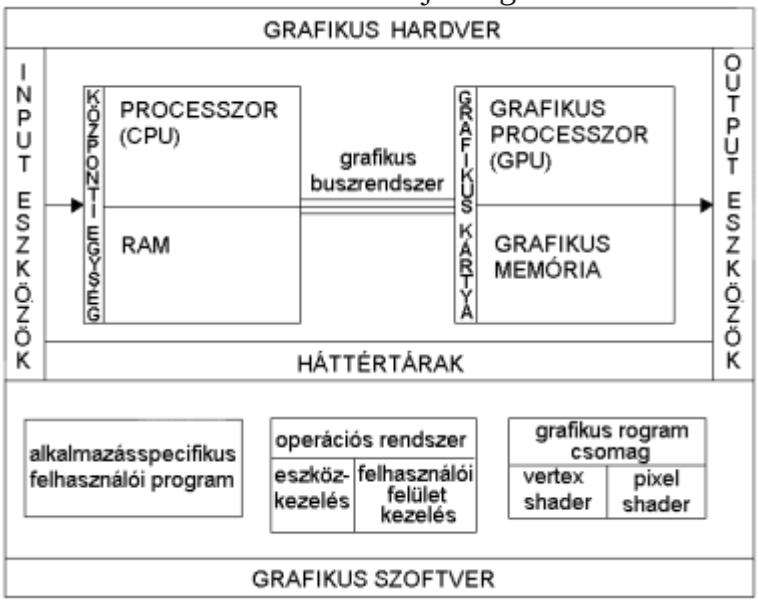
86. Jellemezze a Direct 3D „házi” szabványt!

A Direct X multimédiás és grafikus komponensekből áll, ezek közül a **Direct 3D** a 3D-s vektorgrafikus modellezés és a 3D-s jelenetek képi ábrázolásának szabványa. A Direct 3D API funkcionálisan rendkívül hasonlít az OpenGL-re. Így például a 3D-s objektumokat poligonfelületekkel modellezi, azonosak a modelltér és megjelenítés transzformációs lehetőségei, hasonló a láthatósági, valamint a megvilágítási és árnyalási algoritmusok kezelése, a speciális effektek (pl. köd) biztosítása. A Direct 3D jó animációs lehetőségeket biztosít, erre is visszavezethető, hogy főként a PC-s játékprogramoknál használják elterjedten.A Direct 3D grafikai alkalmazásában jelentős hátrányt jelent, hogy lényegében csak a Windows operációs rendszerek alatt működőképes, emiatt a Direct 3D-re alapozott szoftverek gyakorlatilag más platformokon nem futtathatók.

Az OpenGL-lel összehasonlítva a Direct 3D-nek viszont sokkal folyamatosabb a hardvertámogatása.

7. fejezet

87. Milyen hardver- és szoftverelemek alkotják a grafikus rendszer architektúráját?



88. Nevezzen meg raszter- és vektorgrafikus input és output eszközöket! Fogalmazza meg alapfeladataikat! Mire szolgálnak az adathordozók és háttértárak a grafikus rendszerben? Nevezzen meg adathordozókat és háttértárakat, röviden ismertesse főbb jellemzőiket!

A **raszteres input eszközök** – szkenner, digitális kamera és fényképezőgép (lásd az ábrán) – segítségével fekete-fehér vagy színes képeket, filmeket vihetünk be képpontonként digitális formában a grafikus rendszerbe.

Vektoros input eszköz a digitalizáló tábla (tablet), amely koordinátaadatok bevitelére szolgál, a szálkereszt helyzetét a grafikus rendszer vektorkoordinátaként értelmez.

Raszteres output eszköz a monitor. Biztosítja a grafikus program és a felhasználó kommunikációját, így megjeleníti a modelltér objektumait munka közben képernyőn (pl. huzalvázas forma), és lehetővé teszi a munka végtermékének (pl. kép, film) megtekintését. A nyomtató a grafikus rendszer eredményadatainak nyomtatására használatos raszteres eszköz.

Vektoros adatok (pl. tervrajzok) papírra történő kirajzolására rajzgépet (plotter) használhatunk.

A **háttértárak** (mágneslemez, SSD), RAID tömbök a grafikus szoftver és a hozzátartozó adatállományok (pl. textúrák), a vektorgrafikus adatbázis, valamint a grafikus rendszerrel előállított raszteres eredményadatok (film, képek) hosszabb idejű tárolására szolgálnak.

Nagy mennyiségű adat szállítására megfelelők lehetnek még a gyors, USB portra csatolható hordozható merevlemezek.

Kisebb adatmennyiség szállítására megfelelőek a nagyobb kapacitású pendrive-ok.

Szintén kisebb adatmennyiség ideiglenes tárolására, szállítására használatosak a nagyobb kapacitású memóriakártyák.

Az adatok megbízható, hálózatos tárolására RAID tömböket használnak.

89. Mi a központi processzor szerepe egy vektorgrafikus rendszerben? Mi a feladata a grafikus kártyán található processzornak, és milyen részekből áll?

A központi memóriába (RAM = Random Access Memory) töltődnek be, és a **processzoron** (CPU = Central Processing Unit) futnak a grafikus szoftverek (alkalmazásspecifikus felhasználói program és grafikus programcsomag). Ennek megfelelően a processzor végzi a modelltérben végrehajtott műveleteket, a jelenet objektumainak felbontását háromszögekre, és a jelenet háromszögeinek adatait továbbítja a grafikus kártya processzorához.
Mivel a modelltérben lebegőpontos koordináta-rendszereket alkalmazunk, ezért a grafikus rendszerek teljesítménye szempontjából a processzor lebegőpontos műveletvégző képessége a meghatározó.

A grafikus kártya alapkiépítésben a képmemóriában (frame buffer) tárolt színkódok alapján vezérli a kép kirajzolását a monitoron.
Vektorgrafikus rendszerekben ezen túlmenően a kártyán található **grafikus processzor** (2D, 3D gyorsító csip) a processzortól kapott háromszögadatok alapján elvégzi a megjelenítéshez szükséges geometriai transzformációkat és a raszteres kép előállítását is a képmemóriában (frame buffer). Az előbbi feladatot végrehajtó processzorrészt geometriai vagy vertex processzornak, az utóbbi, vektor-raszter konverziót végző áramköröket raszterprocesszornak is szokták nevezni

90. Milyen szoftverelemekből épül fel egy grafikus rendszer szoftver architektúrája? Milyen együttműködés van a felhasználói program, a grafikus programcsomag és a grafikus processzor programja között? Mi a feladata a modellező szoftvernek? Mi a feladata a renderelő szoftvernek?

Az egyes szoftverelemek fontosabb feladatai a következők:

- Operációs rendszer
  - az ablakozó rendszer kezelése és a frame-bufferből való végső kirajzolás a képernyőre,
  - grafikus I/O eszközkezelés.
- Alkalmazásspecifikus felhasználói program
  - interaktív kapcsolattartás a felhasználóval,
  - alkalmazási területtől függő funkciók,
  - 3D modellezés,
  - objektummozgatás, animáció,
  - objektumstruktúrák kezelése (jelenetgráfok = scene graph),
  - grafikus adatbázis-kezelés,
  - jelenet előkészítése rendereléshez (fényforrás, kameradefiníciók stb.).
- Grafikus programcsomag
  - geometriai alapelemek rajzolása,
  - szín, megvilágítási, árnyalási modellek,
  - transzformációk,
  - speciális effektusok, textúrakezelés,
  - görbe- és felületmodellezés és háromszögekre bontás.
- Grafikus kártya eszközkezelő programja
  - hardverspecifikus illesztés,
  - grafikus processzor vezérlése, adatok átadása.
  - Grafikus processzor programjai (vertex és pixel shader)
  - transzformációk és megvilágítási számítások (vertex shader),
  - raszterizálás (pixel shader).

A felhasználói program a rendereléshez a grafikus programcsomagnak általában háromszögekből álló primitívek formájában adja át az adatokat, vagy vertexek (kontrollpontok) megadásával „megrendeli” a felület- ill. görbemodellezést. A grafikus programcsomag a GPU-nak a háromszögekre vonatkozó vertexek koordinátáit, szín- és textúraadatait adja tovább.

A **modellező szoftverek** feladata a felhasználóval való interaktív kapcsolattartással a 3D modelltér objektumainak létrehozása, képernyőn való bemutatása, mozgatása, komplex objektumok és jelenetek összeállítása.

A **renderelő szoftverek** feladata a modelltér jeleneteiből a raszteres kép előállítása (képszintézis).

91. Melyek a rendering pipeline fázisai, és milyen feladatokat látnak el?

A modelltér jeleneteiből a raszteres képet előállító folyamat, a rendering pipeline, három fő részre bontható:

- Modellezési fázis: e lépésben megtörténik a felhasználói program objektumainak átalakítása a renderelő szoftver adatstruktúrájára (pl. felbontás háromszögekre), illetve ez az adatstruktúra aktualizálásra kerül.
- Geometriai fázis: ekkor valósul meg a jelenet objektumainak (háromszögek) transzformálása (kamera, projektív transzformációk, kivágás stb.) és a megvilágítás figyelembevétele.
- Raszterizációs fázis: e munkafázisban a geometriai primitívekből kiszámításra kerülnek a textúrák figyelembe vételével a pixelek színkódjai.

92. Miért fejlesztették ki az MMX processzorokat? Milyen indokai voltak az SSE utasításkészlet kifejlesztésének? Milyen előnnyel jár a többszálas és többmagos processzorok alkalmazása a számítógépes grafikában?

A Pentium **MMX** (Multimedia Extension) és vele kompatibilis processzorok kifejlesztésének célja az volt, hogy a hardver képes legyen a multimédia egyre inkább növekvő igényeit kiszolgálni.

SSE: az SIMD utasításkészletet bővítették, hogy képes legyen a vektorgrafikus ábrázolást is támogatni.

A vektorgrafika modellterében végrehajtott, és a rendering pipeline-nak a modellezés szintjén működő algoritmusai legtöbbször jól párhuzamosíthatók. (Gondoljunk pl. a raytracing esetében arra, hogy az egyes sugarak követése lépésenként egymástól függetlenül végrehajtható.) Ezért a processzorgyártásban 2004-ben, ill. 2005-ben bevezetett **többszálas** (hyper-threading) és **kétmagos**(dual core) technológia (Intel Prescott, ill. Smithfield) jelentős teljesítménynövekedést eredményezett a számítógépes grafikában is.

93. Milyen részegységekből épül fel a monitorvezérlő kártya? Miben mérjük a monitorvezérlő kártyák teljesítményét? Hogyan számítjuk ki a frame-buffer memóriaigényét?

A grafikus kártya fontosabb részegységeinek feladatai a következők:

- Grafikus síninterfész: puffereken keresztül biztosítja a kártya kapcsolatát az AGP vagy a PCI Express sínnel;
- Videomemória (VRAM): DRAM memória a megjelenítendő digitális kép (frame buffer) és a 2D és 3D gyorsító algoritmusokhoz szükséges adatállományok (pl. textúrák) tárolására;
- Grafikus processzor: vezérli a frame buffer feltöltését, és hardverúton megvalósítja a rendering pipeline geometriai és raszterizációs grafikus algoritmusait (2D és 3D gyorsítás);
- Video-ROM: a kártyához tartozó video ROM-BIOS-t, valamint a grafikus módok és a karakterkészletek definícióit tartalmazza;
- RAMDAC (Random Access Memory Digital-to-Analog Converter): a képernyőn történő megjelenítéshez szükséges digitális-analóg konverziót végzi, előállítva az analóg RGB jeleket a monitor számára. (A grafikus kártya digitális-analóg konvertere, a RAMDAC teljesítményének és a monitor képfrissítési frekvenciájának és felbontásának is meg kell felelnie egymásnak.) Az LCD monitorok használhatnak analóg kimeneti jeleket, digitálisat vagy mindkettőt.
- Output interfészek. (VGA/Video Graphics Array (DE-15), DVI/Digital Visual Interface, VIVOVideo In Video Out az S-Video-hoz, Composite video and Component video, HDMI/High-Definition Multimedia Interface, DisplayPort stb.)

A grafikus kártyák teljesítményét frame/sec-ben (FPS-ben) mérjük, amely a másodpercenkénti teljes raszteres képernyő megjelenítések számát jelenti. Fontos megemlíteni, hogy 25 FPS feletti sebesség kell a folyamatos mozgókép érzékeléshez.

94. Mi a 2D/3D-s gyorsító csipek alkalmazásának lényege? Mi a lényege a stream modellnek? Mi a feladata a geometriai processzornak? Mi a feladata a raszterprocesszornak? Mi a grafikus sínrendszer feladata?

A 2D/3D gyorsítás lényege az, hogy a két-, illetve a háromdimenziós képek előállításához szükséges számítások egy részét a grafikus kártyán lévő csip (GPU) veszi át, ezáltal tehermentesíti a központi processzort (CPU). A feladatmegosztás a 2D és a 3D gyorsító áramkörök között a következő:

- A 2D gyorsító áramkörök feladata a Windows ablakkezelő funkcióihoz (pl. egy ablak megnyitása) szükséges vonalkezelési és színkitöltési feladatok végrehajtása.
- A 3D gyorsító áramkörök a rendering pipeline geometriai és raszterizációs szakaszaihoz tartozó algoritmusokat valósítják meg.

A grafikus processzor felépítésére a **stream modellt** alkalmazzák. Ennek két lényeges eleme van: az adatfolyam vagy stream és az ezt feldolgozó műveletvégző egység, a kernel. A modellben stream alatt egymástól független, azonos típusú adatok sorozatát értjük. A kernel a stream egy elemére egy adott műveletet képes végrehajtani.

A grafikus processzor két alapvető részből épül fel:

- a **geometriai processzor** végzi el vertexfeldolgozó egységeivel a regiszterekben átadott háromszög vertexadatfolyamon először a transzformációs és megvilágítási számításokat, majd ezt követően végrehajtja a kivágást és a képernyőablakra történő vetítést;
- a **raszterprocesszor** a háromszögek csúcsponti értékei alapján interpolációval kiszámítja a belső háromszögpontok szín és megvilágítási értékeit, az így kapott pixelek adatait továbbadja a pixelfeldolgozó egységeknek. Itt megtörténik a végső láthatóság kiszámítása (z-buffer), a textúrázás és a speciális effektusok kihatásainak figyelembevétele.

A **grafikus sínrendszer** feladata a központi egység (CPU és RAM) és a GPU közötti adatátvitel biztosítása, ami napjaink grafikus kártyáinál másodpercenként 500-1000 millió háromszögre vonatkozó vertex, szín és textúra koordináta (4D és 2D vektor) adat átvitelét jelenti. Emellett ennek a sínrendszernek kell biztosítani a grafikus processzor shader programjainak (korábbi GPU-knál az ún. fix eljárások) továbbítását is.

95. Milyen előnnyel jár az OpenGL felhasználása egy grafikus alkalmazás fejlesztésében? Hogyan használjuk egy C programban az OpenGL függvényeket és állapotváltozókat? Sorolja fel az OpenGL fontosabb alapfunkcióit!

Az OpenGL-t használó alkalmazások hordozhatók, több géptípuson és operációs rendszer alatt is futtathatók.

Az egyes OpenGL funkciókat a megfelelő OpenGL függvény meghívásával aktivizálhatjuk. A függvényhívó parancs felépítése: *glFuggvenynev().*

A rendszerben azokat a globális paramétereket nevezzük állapotváltozónak, amelynek értékét figyelembe véve hajtja végre az OpenGL az egyes parancsokat. Ilyen pl. a grafikus objektum színe. Ezt nem kell minden parancsban beállítani, mert a rendszer az állapotváltozót addig használja, amíg azt a programozó meg nem változtatja.

A programfejlesztő által igénybe vehető **OpenGL alapfunkciók** – a teljesség igénye nélkül – a következők:

- Geometriai modellezési parancsok: OpenGL primitívek, gömb, kúp, tórusz, Bézier és NURBS görbék és felületek.
- Szín, árnyalás és megvilágítási parancsok: RGBA színek, Gouraud és flat shading, fényforrások, diffúz, tükröző visszaverődés, csillogás.
- Textúrakezelési parancsok: textúradefiniálás, mip mapping, szűrők.
- Speciális effektusok parancsai: átlátszóság, anti-aliasing, köd.
- Raszteres objektumokat kezelő parancsok: bittérképek és karakterek, pixeles képek (nagyítás, kicsinyítés).
- Pufferkezelő parancsok:
  - színpuffer (itt keletkezik a kép),
  - z-puffer (a z-buffer algoritmus z távolságértékének kezelésére),
  - gyűjtő puffer (több kép összegzésére használható).

96. Milyen lehetőséget biztosítanak a fejlesztőnek a programozható geometriai és raszterprocesszorok? Milyen feladatot lát el a vertex shader program? Mi a feladata a pixel shader programnak?

A **programozható raszter- és geometriai processzorok** esetében a megjelenítési (rendering pipeline) algoritmusok már programmal módosíthatók. Ez azt jelenti, hogy a shader (árnyaló) programokkal a grafikus alkalmazásfejlesztő közvetlenül beavatkozhat a rendering pipeline vertex-transzformációs és megvilágításszámítási, illetve a pixel- és textúraszámítási műveleteibe.

A **vertex shader program** minden egyes vertex esetében megkapja a helykoordinátákat (beleértve a normálvektort is), a textúra-koordinátákat és színvektort, az objektumokra (háromszögekre) vonatkozó állapotadatokat (pl. transzformációs mátrixok, anyagjellemzők stb.). Ezek alapján elvégzi a transzformációs és megvilágítási számításokat, és átadja a megváltoztatott hely, textúra, színkoordinátákat.

A **pixel shader program** (OpenGL terminológiában fragment shader) minden egyes pixelre (fragmentre) megkapja a helykoordinátákat, a szín- és textúrakoordinátákat, elvégzi a végleges szín- és textúraszámításokat és átadja a számított színvektort és „z" értéket.

97. Sorolja fel a számítógépes tervezés néhány részterületét, és adjon meg néhány professzionális tervezőprogramot! Milyen célokra állíthatunk elő grafikákat például a CorelDRAW-val? Milyen programcsomagokat használnak fel a DTP területén? Nevezzen meg legalább három programcsomagot, mellyel a filmipar számára készítenek animációkat!

A professzionális grafikus szoftverek legjellemzőbb felhasználási területei a következők:

- A számítógéppel segített tervezés (CAD), amelynek legfontosabb részterületei a géptervezés és az építészet. Ilyen szolgáltatásokat nyújtó programcsomagok például az AutoCAD, a CADKEY és az ArchieCAD.
- A különböző grafikák, például reklám célra történő számítógéppel történő előállítása. Ezek lehetnek különböző termékismertetési vagy oktatási célú prezentációk, de lehetnek képzőművészek által készített képek is. A vektoros rajzolóprogramok közül a legnagyobb múlttal rendelkezik és a legismertebb a **CorelDRAW**.
- A nyomdai kiadványszerkesztés (**DTP**) ma már elképzelhetetlen a számítógépes grafika felhasználása nélkül. E területen jól ismert és alkalmazott programok az Adobe Photoshop és az Adobe PageMaker.
- A szórakoztató és a filmipar is egyre inkább hasznosítja a számítógépes grafikát (lásd az ábrát). Ezekkel például elképzelt, sci-fi világokat lehet alkotni vagy 3D-s virtuális "szörnyeket" lehet készíteni. E területen ismert programok közül a 3D Studio MAX, a SOFTIMAGE 3D és a Maya említhető meg.

98. Mire használható leginkább a 3D Studio Max? Sorolja fel a 3D Studio Max legfontosabb lehetőségeit!

Az Autodesk cég által kínált 3D Studio Max (rövidebben 3ds Max) egy vektorgrafikus modellező és animációs programcsomag, amelyet leginkább játékfejlesztéshez, ipari és építészeti tervek vizualizációjához, mozifilmek vizuális effektusaihoz és oktatófilmek készítéséhez használhatunk.

A programcsomag egyes lehetőségeit az alábbiak mutatják be:

- Valósághű karakterek létrehozására alkalmas karakterfejlesztési funkciókkal rendelkezik, beleértve a ruha- és fejszimulációs eszközöket is. Támogatja a karakteranimációt.
- Több modellező eszközt kínál, mint például a primitívekkel való modellezés (CSG műveletek lehetőségével), kis- és nagyfelbontású poligonmodellezés, spline-modellezés (NURBS). Poligonmodellezésnél az ún. NURMS (Non-Uniform Rational MeshSmooth) technikának köszönhetően, a poligonháló éleinek geometriáját finoman lehet változtatni.
- Lehetővé teszi egy részecske-kibocsátó objektum létrehozását, amelyet havazás, felhő vagy spray modellezésére használhatunk.
- A felületek reális anyagi mintáknak megfelelő textúrázására (fa, márvány stb.) lehetőség van, biztosítva az UV koordinátákban megadott textúrák pontos illesztését az objektumok geometriájához.
- A programcsomag számtalan fényeffektust képes kezelni, mint például lencsecsillogás, füst, csillagköd, tűz stb.

99. Melyik tervezőprogram a legelterjedtebb a PC alapú rendszerek körében, és melyek a fő felhasználási területei? Milyen szakmaspecifikus szoftverek egészítik ki az AutoCAD alapprogramot?

Az Autodesk cég által kifejlesztett AutoCAD áll az első helyen a CAD tervező programcsomagok piacán, részesedése különösen a PC-alapú rendszereknél meghatározó. Fő felhasználási területe az építészeti, belső építészeti és gépészeti tervezés, de alkalmazzák például a város- és környezettervezésben vagy a térképészetben is.

Az AutoCAD alapprogramot a különböző szakmáknak megfelelő speciális CAD szoftverek egészítik ki. Ezek lehetővé teszik, hogy az általános geometriai objektumok mellett egy olyan elemkészletet is felhasználjunk, amely az adott szakma igényeihez igazodik (például építészeti 3D-s fal-, födém-, tető- és nyílászáróelemek vektorgrafikus objektumai). Ilyen kiegészítő tervezőrendszer a gépészetben az AutoCAD Mechanical, az építészetben az Autodesk Architectural Desktop, vagy a térképészetben az Autodesk Map 3D. Az épülettervezést segítik az épületek mechanikai, villamossági és vízvezetékrendszerének dokumentációját készítő szoftverek (Autodesk Building Systems és Autodesk Civil 3D). Az AutoCAD LT kiegészítő program lehetővé teszi a 2D-s tervek elkészítését, kinyomtatását, közzétételét. Az adatokat DWF (Design Web Format) fájlformátumban gyorsan meg lehet osztani mással az interneten keresztül.

100. Milyen fontosabb részekből áll a Corel programcsomag? Mit tartalmaznak a Corel objektumkönyvtárak, és miért előnyös a használatuk?

A Corel rajzoló, illusztrációkészítő programcsomag legújabb változatai többek között a következő főbb alkotórészeket tartalmazzák:

- CorelDRAW – vektoros rajzoló és oldaltervező programot,
- Corel PHOTO-PAINT – raszteres fényképszerkesztő programot,
- Corel TRACE – raszteres képet vektorossá átalakító programot (lásd az ábrát),
- Corel R.A.V.E. (Real Animated Vector Effects) – internetes vektoralapú animációkészítőt,
- Corel CAPTURE – a képernyő képeinek „elkapására” szolgáló szoftvert.

A programcsomaggal együtt bőséges objektumkönyvtárat is szállítanak, mely több 10000 clipartot és digitális képet, valamint kb. 1000 OpenType fontot tartalmaz.

101. Mire alkalmas az Adobe Photoshop programcsomag? Mutassa be az Adobe Photoshop nyomdatechnikai lehetőségeit? Mondjon példákat az Adobe Photoshop filtereire!

A nyomdai munkában, a kiadványszerkesztésben mindig kiemelt jelentőségű részterület volt a vizuális adatok, képek kezelése, de manapság a digitális képek feldolgozása a fototechnikának is komoly részterületévé vált.

A DTP képfeldolgozás szakterületére számos programcsomag specializálódott, de a legelterjedtebb és a szakma értékítélete szerint legsokoldalúbb mindenképpen az Adobe cég Photoshop nevű raszteres képfeldolgozó programja. Ez alkalmas nyomdába vagy internetre szánt képek előkészítésére, digitális felvételek retusálására, vagy akár montázsgrafikák elkészítésére.

A programcsomag lehetőségeit néhány példával érzékeltetjük:

- A nyomdatechnikában a színes képeket a CMYK színtér komponensei szerint kell felbontanunk (color separation). Ehhez a Photoshop lehetőséget biztosít olyan színkivonatos (color plate) képek előállítására, amelyek a CMYK színtér alapszíneinek megfelelő színek árnyalatait adja vissza. Ezeket a nyomda az ún. színnyomatok készítésénél használhatja fel.
- A szokásos színterek (RGB, CMYK, HSB) mellett a Photoshop beépített színmintákat is felkínál, amelyeket a nyomdai festékgyártók színkódjai alapján is megválaszthatunk.
- A program külön szolgáltatásokkal támogatja a nyomdai kép előkészítéséhez szükséges kalibrációt, azaz a monitoron megjelenő kép színeinek a nyomdai színekhez való igazítását.
- Lehetőségünk van rétegek (layerek) alkalmazására. Ezekben önállóan speciális effektusokat is alkalmazhatunk: például izzás, domborítás. Az új lehetőségeknek köszönhetően egy réteg „fóliáját” nem csak mozgatni, hanem hajlítgatni, gyűrni és torzítani is lehet (ún. 3D-s réteg).
- A képek manipulálását és speciális effektusok előállítását az ún. filterek (szűrők) segítik. Így például a blur filter lágyítja a színek közti átmenetet. Ennek mértékét beállíthatjuk, és akár koncentrikus körök mentén is alkalmazhatjuk. Utóbbi esetében például a Spin (perdület) paranccsal a képen a forgó mozgást szimulálhatjuk. A distort filterrel eltorzíthatjuk a képet például egy másik kép adatai alapján, a ripple filterrel pedig vízben tükröződő képet állíthatunk elő.
- A digitális fényképek finomítása céljából lehetőség van szemcsézettség, illetve vörös szem effektus eltüntetésére. A program képes RAW (nyers digitális képformátum) fájlok feldolgozására, az expozíció, optikai lencse, árnyék, világosság, kontraszt és más paraméterek korrekciójával.
- A Photoshop lehetőséget nyújt a kép nemkívánatos elemeinek az eltüntetésére, a megmaradt terület automatikus kipótlásával.
- A program segítségével készíthetünk weboldalra szánt animált GIF-eket is.

102. Mire használható a Maya programcsomag? Ismertesse a Maya által nyújtott fontosabb lehetőségeket!

A Maya az Autodesk cég által kifejlesztett 3D-s modellező, animációs és renderelő programcsomag. Használják a filmiparban, a játékfejlesztés, a televíziózás, a hirdetés, a nyomdaipar vagy a grafikus tervezés terén. Professzionális funkciókat biztosít többek között a karakteranimáció, a vizuális effektusok és a trükkök területén.

A szoftver által nyújtott néhány fontosabb lehetőség:

- fejlett NURBS- és poligon-modellezési, valamint textúrázási eszközök;
- hierarchikus felületfelosztásos modellezés (Subdivision Surface Modeling);
- rugalmas karakterépítési és karakter-animációs eszközök (megadott kulcspozíciók interpolálása, nemlineáris animáció, direkt és inverz kinematika);
- hatékony deformációs eszközök az izmok feszülése (például beszéd, mosoly) modellezésére;
- integrált részecskedinamika, valamint merev és rugalmas testek kölcsönhatásának kezelése;
- nyomásérzékeny "ecsetek" 2D-s felületre és 3D-s térben való festéshez (Maya Paint Effects, Maya Artisan);
- ruhaanyag, haj, szőrzet realisztikus modellezésének széles skálája;
- légköri, pirotechnikai, folyadékbeli effektusok kezelése;
- több rendelkezésre álló renderelő (raytracing alapú Maya szoftver renderelő, hardver és vektor renderelő, mental ray) közötti választás;
- teljes programozói felhasználói felület (API) és beépített szkript szerkesztő (MEL);
- gyors adatcsere az Adobe Photoshop és Adobe Illustrator termékekkel;
- rendelkezésre állnak kompozitáló eszközök.
- használhatunk előzetes látványtervező és játékprototípus-készítő eszközkészleteket, kiterjesztett szimulációs képességeket és továbbfejlesztett folyamatintegrációt biztosít;
- a Viewport 2.0 fejlesztései: A kiértékelést nagy pontosságú környezetben, teljes képernyős effektusokkal végezhetjük a nagy teljesítményű nézetablakokban.
- Csomópontalapú renderelési lépés: Létrehozhatók csomópontalapú renderelési lépésábrázolások, és közvetlenül a Maya-ban szerkeszthetők a munka ellenőrzéséhez és finomításához.
- Szerkeszthető mozgásútvonalak: Az animációk közvetlenül a nézetablakban szerkeszthetők, nincs szükség gráfszerkesztőre váltsra.
- Sequencer fejlesztései: Átrendezhetők a videók, módosíthatók a belépési és kilépési pontok, és megváltoztathatók a kamerakiosztások a Sequencer lejátszási listában.
- Procedurális anyagtextúrák: Procedurális anyagtextúrát tartalmazó anyagtár használatával többféle megjelenésváltozat érhető el.

103. Mi a VRML, és milyen alapfunkciókat nyújt? Ismertesse a VRML2 új lehetőségeit a VRML1-hez képest! Mit értünk séta (walk) alatt a virtuális térben? Mit jelent a VRML fájlok lejátszása, hogyan változtathatjuk helyzetünket a virtuális térben?

A virtuális valóság modellező nyelve a VRML (Virtual Reality Modeling Language), amely az 1990-es évek második felében vált szabvánnyá (ISO/IEC 14772). A VRML-lel alapvetően térgeometriai objektumokat definiálhatunk. Ezek lehetnek például kockák, kúpok, gömbök stb., amelyeknek meg kell adni a virtuális térbeli koordinátáit és jellemzőit (szín, méret stb.). Ezt egy ASCII szövegfájlban tehetjük meg, amelynek a kiterjesztése a konvenció szerint WRL. A VRML nyelv az alakzatok színeit az RGB alapszínek keverésével állítja elő. A három szín arányát egy számhármassal adhatjuk meg. A virtuális térben fényforrásokat és kamerákat (3D-s nézőpont) is meghatározhatunk, és lehetőség van az objektumok affin transzformálására is.

A VRML első változatát követően megjelent a VRML 2. verziója, amely a modelltérben definiálható objektumokkal kapcsolatos lehetőségeket jelentősen kiszélesítette. A lehetőségek érzékeltetésére csak néhány példát sorolunk fel:

- a virtuális térben szenzorokat helyezhetünk el, amelyekkel meghatározott eseményeket figyelhetünk és ezek bekövetkezésekor akciókat kezdeményezhetünk;
- a virtuális térben már valós fizikai viszonyokat is modellezni lehet, például gravitáció, repülés;
- a VRML fájlban hangforrásokat határozhatunk meg;
- a virtuális térben ütközést is modellezhetünk, át nem járható tárgyak is létezhetnek;
- a virtuális teret a 3D-s grafikában már ismert effektusokkal is felruházhatjuk, például speciális textúrák, köd stb.

A séta (walk) a virtuális valóságban azt jelenti, hogy egy program igénybevételével – melyet lejátszó programnak nevezünk – „bejárhatjuk” a 3D-s modellteret. Lehetőségünk van a definiált objektumokat „körbejárni” és különböző nézeteiket megszemlélni a lejátszó program által a képernyőn visszaadott valódi 3D-s térhatású képen.

A VRML fájlok lejátszásakor a virtuális térbeli helyzetünket, vagyis a nézőpont meghatározását interaktív módon, például egérrel folyamatosan vezéreljük. Ennek hatására a lejátszó program a képernyőn folyamatosan kirajzolja a helyzetünknek megfelelő térhatású képét a modelltér jelenetének. Természetesen ennek feltétele, hogy a program mindig valós időben tudja renderelni a megjelenítendő képet.

104. Mi a POV-Ray? Melyek a POV-Ray fontosabb tulajdonságai? Mit nevezünk 3D-s modellezőknek, és miért fejlesztették ki? Mire használható a Moray program? Mire alkalmas a ppModeler?

A POV-Ray (Persistence of Vision Ray-Tracer) egy nyílt forráskódú, ingyenesen terjeszthető 3D-s fotorealisztikus renderelő program, amely a rendereléshez a raytracing technikát alkalmazza. A program egy saját leírónyelvet használ, ezzel kell definiálni a modelltér objektumait, a kamerát, a fényforrásokat és az effektusokat. A leírónyelven megfogalmazottakat POV kiterjesztésű fájlokban tárolják.

Az alábbiakban összefoglaljuk a POV-Ray fontosabb tulajdonságait:

- egyszerűen használható leírónyelv,
- előre definiált alakzatok, színek és textúrák, példajelenetek terjedelmes könyvtára,
- egyszerű (gömb, kocka, kúp stb.) és komplex (tórusz, poligon, Bézier-folt, forgási felület, fraktál stb.) primitív alakzatok, amelyekkel CSG-műveleteket lehet végrehajtani,
- textúrák réteg- vagy mozaikszerű egyesítése,
- több kameratípus (perspektivikus, ortogonális, halszem stb.),
- photon mapping technika a kausztikus képek kezelésére,
- Phong-féle spekuláris fényvisszaverődés kezelése,
- radiosity algoritmus lehetősége a diffúz jelenségekre,
- légköri effektusok (köd, szivárvány stb.), valamint részecske jelenségek (felhők, por, tűz, gőz) modellezése (lásd ábrákat),
- exportálás több fájlformátumba (BMP, PNG, Targa, PPM).

A POV-Ray hátránya, hogy nincs grafikus interaktív felhasználói felülete, ezért például az objektumok elhelyezkedését a térben előre meg kell adnunk a formális leírónyelven és ennek eredményét csak – a sokszor időigényes – renderelést követően szemlélhetjük meg. Ezért fejlesztettek ki a POV-Ray-hez olyan 3D-s modellezőprogramokat, amelyekkel interaktív módon követni tudjuk a modelltérben az objektumok megtervezését, a megfelelő képek megtekintését és esetleges változtatását.

Moray program: Huzalváz-modellező A program képernyője a jelenet három 2D-s nézetét ábrázolja huzalvázas formában, valamint egy 3Ds renderelt képet a kameraállásból.

pppModeler: Képes az elkészített jelenetet POV formátumban exportálni. Ez a modellező kiválóan alkalmas karaktermodellezésre.

105. Mi a Blender? Ismertesse a Blender legfontosabb funkcióit!

A Blender egy ingyenes, professzionálist megközelítő 3D-s modellező és animáció-készítő program, amellyel főként filmek készítenek. Ezeket interaktív tartalommal is el lehet látni, ami lehetővé teszi például hirdetések interaktív játékként való közzétételét a weben.

A szoftver főbb jellegzetességei közül említünk meg néhányat:

- 3D-s objektumok modellezése poligonhálókkal, NURBS felületekkel, Bezier és B-spline görbékkel, metaball alakzatokkal, vektorgrafikus betűkészlettel,
- fejlett animációs funkciók (direkt és inverz kinematika, automatikus bőrözés, nemlineáris animációkeverés stb.),
- hang és kép szinkronizálásának támogatása,
- gyors beépített sugárkövetéses renderelő, radiosity fényterjedésszámítás,
- több felületárnyló eljárás közötti választás (például Phong),
- grafikus szerkesztő interaktív tulajdonságok meghatározására,
- UV textúraszerkesztő, textúrákkal előállítható effektusok (environment mapping, reflection mapping stb.),
- procedurális textúrák,
- jelenetek réteges kezelése,
- saját tömöríthető fájlformátum (.blend), egyéb fájlformátumok támogatása (TGA, JPG, AVI, TIFF stb.).

A Blenderrel bárki létrehozhat saját gépén egy házi animációs stúdiót, aminek segítségével a legkülönbözőbb felhasználási területekre készíthet animációkat: építészeti bemutatók, weboldalak tervezése, animált hirdetések, termékek modellezése stb. A szoftver különböző operációs rendszerek (mint Windows, Mac, Linux, Sun) alá telepíthető változatokkal rendelkezik.

106. Mire alkalmas a GIMP programcsomag? Sorolja fel a GIMP fontosabb lehetőségeit!

A GIMP (GNU Image Manipulation Program) egy nyílt forráskódú, ingyenesen terjeszthető szoftver, amely alkalmas rajzolásra, grafikus képek előállítására és összeszerkesztésére, digitális fotók retusálására, általában profi szintű képmanipulálásra. Rendelkezik egy plug-in gyűjteménnyel (GIMP Animation Package), amely alkalmassá teszi animációk létrehozására is.

Íme néhány a GIMP nyújtotta lehetőségek közül:

- festőeszközök széles választéka (ecsetek, ceruza, festékszóró stb.),
- szubpixel szintű sampling a lépcsőeffektus csökkentésére (anti-aliasing),
- több réteg használata egy képen (beleértve szerkeszthető szövegréteget is), amelyekkel algebrai kompoziciós műveleteket lehet végezni teljes alfa csatorna (Alpha Channel) támogatással az átlátszóság kezelésére,
- fejlett kijelölő eszközök (téglalap, ellipszis, fuzzy, Bézier stb.),
- transzformációs eszközök (forgatás, skálázás, nyírás, tükrözés),
- csak a rendelkezésre álló lemezterület által korlátozott képméret,
- procedurális adatbázis, amely lehetővé teszi a belső funkciók más programból való hívását,
- fejlett szkriptelési lehetőség, a legegyszerűbb művelettől a komplex képmanipulációs eljárásokig,
- több fájlformátum támogatása (a saját XCF mellet GIF, JPEG, TIFF, TGA, PCX, PDF, BMP stb.),
- nagy számú igénybe vehető plug-in, további filterek és fájlformátumok hozzáadásához.

107. Soroljon fel három kereskedelmi forgalomban kapható, majd három ingyenesen használható grafikus szoftvert. Fogalmazza meg, jellemzően milyen területeken használják azokat, mi az egyedi bennük/miben különböznek az elérhető más szoftverektől.

A professzionális grafikus szoftverek legjellemzőbb felhasználási területei a következők:

- A számítógéppel segített tervezés (CAD), amelynek legfontosabb részterületei a géptervezés és az építészet. Ilyen szolgáltatásokat nyújtó programcsomagok például az AutoCAD, a CADKEY és az ArchieCAD.
- A különböző grafikák, például reklám célra történő számítógéppel történő előállítása. Ezek lehetnek különböző termékismertetési vagy oktatási célú prezentációk, de lehetnek képzőművészek által készített képek is. A vektoros rajzolóprogramok közül a legnagyobb múlttal rendelkezik és a legismertebb a CorelDRAW.
- A nyomdai kiadványszerkesztés (DTP) ma már elképzelhetetlen a számítógépes grafika felhasználása nélkül. E területen jól ismert és alkalmazott programok az Adobe Photoshop és az Adobe PageMaker.
- A szórakoztató és a filmipar is egyre inkább hasznosítja a számítógépes grafikát (lásd az ábrát). Ezekkel például elképzelt, sci-fi világokat lehet alkotni vagy 3D-s virtuális "szörnyeket" lehet készíteni. E területen ismert programok közül a 3D Studio MAX, a SOFTIMAGE 3D és a Maya említhető meg.

3 fitezős: Adobe Photoshop, AutoCAD, Maya
3 ingyenes: Gimp, Blender, POV-Ray