Számítógépes grafika

XVII. rész

A grafikai modellezés

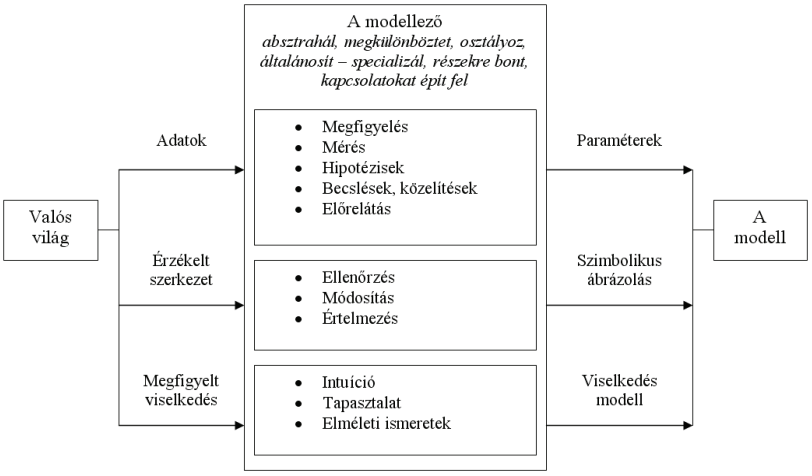
A modellezés

A generatív számítógépes grafikában és a képfeldolgozás során nem a valódi objektumokat (valóságbeli tárgyakat), hanem azok egy *modell*jét dolgozzuk fel.

A *modellezés* során a valós tárgyakból entitásokat absztrahálunk. Az ember a körülötte lévő tárgyakat, valós entitásokat észreveszi, leegyszerűsíti, megkülönbözteti és rendszerezi. A végső cél a bonyolult rendszer megismerése, működésének megértése. A felhasznált eszköz pedig a modellezés. A modellezés során az ember tulajdonképpen egy alapvető, elemi gondolatmenetet (algoritmust) használ, amelynek segítségével absztrahál, megkülönböztet, osztályoz, általánosít – specializál, részekre bont és kapcsolatokat épít fel.

Az *absztrakció* az a szemléletmód, amely segítségével egy végtelenül bonyolult rendszert leegyszerűsítünk úgy, hogy csak a lényegre, a cél elérése érdekében feltétlenül szükséges részekre koncentrálunk. Az absztrahálás tehát azt jelenti, hogy elvonatkoztatunk a számunkra pillanatnyilag nem fontos, közömbös információktól, és kiemeljük az elengedhetetlen fontosságú részleteket.

A megkülönböztetés és az osztályozás szinte automatikus folyamat. Az entitásokat a számunkra lényeges tulajdonságaik, viselkedési módjuk alapján megkülönböztetjük és kategóriákba, osztályokba soroljuk őket, oly módon, hogy a hasonló tulajdonságokkal rendelkező entitások egy osztályba, a különböző vagy eltérő tulajdonságokkal rendelkező entitások pedig külön osztályokba kerülnek. Az osztályozás folyamata tulajdonképpen az *általánosítás* és a *specializálás* műveleteinek segítségével valósul meg. Az entitások között állandóan hasonlóságokat vagy különbségeket keresünk, hogy ezáltal bővebb vagy szűkebb kategóriákba, osztályba soroljuk őket. Minden entitás valamilyen osztály *példánya*, rendelkezik osztályának sajátosságaival, átveszi annak tulajdonságait.



1. ábra

A modellezés folyamata

A generatív számítógépes grafikában a feldolgozott grafikus objektumokat (testek, felületek, alakzatok, görbék stb.) *modell-*, *objektum-,* vagy *színter*ekben (*scene*) írjuk le matematikai eljárások segítségével. A modellterek általában két- vagy háromdimenziós koordináta-rendszerek (2D, 3D).

A pixelgrafika modelltere kétdimenziós egész koordinátarendszer.

A vektorgrafika modelltere két- vagy háromdimenziós valós euklidészi tér – lebegőpontos koordinátaértékekkel.

3D modellezők

A 3D modellezők olyan alkalmazások, amelyek a valóságból vett objektumokat dolgoznak fel. Az objektumok úgy viselkednek, vagy úgy néznek ki, hasonló a struktúrájuk, esetleg néhány fizikai tulajdonsággal rendelkeznek (tömeg, térfogat), mint a valóságban. A 3D modellezők moduláris felépítésűek. Egyik modul kezeli az objektumok térbeli modellezését, egy másik a fizikai tulajdonságokat, egy harmadik a vetítést, megvilágítást (perspektíva, fényforrások, atmoszféra, a fény terjedésének törvényei) és esetleg létezik egy animációs modul is, amely az objektumok vagy fényforrások mozgását valósítja meg. A modellezők legújabb generációi tartalmaznak egy interaktivitás-modult is, amelynek segítségével a felhasználó dinamikusan közbeléphet és interaktívan módosíthatja az objektumok tulajdonságait.

A következő modellező szoftvereket tudjuk megkülönböztetni:

- *felületmodellező*: egy „vázra" rányújtunk egy „bőrt" A váz sokszögekből vagy görbékből állhat. Az eredmény egy struktúra, amelynek térfogata van, de nincsenek fizikai tulajdonságai: tömeg, sűrűség stb.
- *szilárd test modellező*: több modellezési lehetőséget nyújt, viszont a számítási idő észrevehetően megnő. A szilárd test modellezőt nehezebb kezelni mint a felületmodellezőt, viszont tudják kezelni a sűrűséget, tömeget, súrlódási tényezőket, gravitációs erőket, ütközéseket stb., így a 3D objektumok viselkedését közelebbhozzák a valósághoz.
- *sokszög-modellező*: konvex sokszögek összességéből állítja össze a 3D objektumot. A hátránya, hogy a görbe felületeket mindig sík felületekkel közelíti meg. Előnye, hogy egyszerűbb számításokat kell elvégezni, és az algoritmusok is sokkal egyszerűbbek, gyorsabbak.
- *spline-görbéken alapuló modellező*: a 3D objektumokat (felületek vagy szilárd testek) matematikailag könnyen leírható görbékkel közelítik meg (spline görbék). Nagyobb rugalmasságot és pontosságot nyújtanak mint a sokszög-modellezők. A legtöbb modellező program egyenletes spline-okon alapszik, egy modernebb verzió a NURBS (*Non-Uniform Rational B-Splines*) modellező. A NURBS esetében a pontok nem egyenletesen vannak elosztva a görbén, így sokkal jobban meg tudják közelíteni a modellezett objektumokat. A pontosság azonban sok matematikai műveletet igényel és a renderelési algoritmusuk kivitelezése még nehézkes és lassú.

3D testek modellezésének módszerei

Minden 3D modellező szoftver rendelkezik egy pár alaptulajdonsággal, ezeket foglaljuk össze a következőkben:

- *primitívek használata*: minden modellező alkalmazás ismer néhány 3D primitív objektumot: gömböket, kúpokat, hengereket, kockákat és néha komplexebb objektumot is, mint a tórusz, dodekaéder stb. Ezekből a primitívekből komplexebb objektumokat lehet előállítani.
- *extrudálás (extrusion)*: a 3D modellezés egyik alaptechnikája, amely abból áll, hogy egy 2D objektumot (kör, téglalap, sokszög) eltolunk egy görbe vagy egyenes mentén a harmadik dimenzióba, így létrehozunk egy 3D objektumot (pl. egy kör extrudálásával egy egyenes mentén egy hengert kapunk). Az extrudálás parametrizálásával (a 2D objektum méretezése, forgatása) a végső objektum formáját befolyásolhatjuk.
- *Forgástest kialakítása (lathing)*: egy görbe tengely körüli forgatásával generálunk 3D testeket.
- *„bőrözés" (skinning)*: egy több görbéből álló vázra kihúzunk egy felületet (hasonló egy hajó építéséhez, ahol előbb a metszeti síkokat építik meg, majd ezek haránt irányú felületekkel kötik össze). A szerkezet minden *bordája* egy ponthalmaz, amelyen keresztülmegy az adott felület.
- *vertex-szerkesztés*: megadja azt a lehetőséget, hogy a 3D objektum bármely pontjának a pozícióját lehessen változtatni. A vertex-szerkesztés kétfajta: *statikus,* amikor a vertexek egymástól függetlenül mozognak, és *dinamikus*, amikor egy vertex elmozdítása maga után vonja a szomszédos vertexek kisebb mértékbeli elmozdulását (így egy simább felületet kapunk).
- *boole-műveletek*: a legtöbb számítást igénylő műveletek, két vagy több 3D objektum egyesítéséből, különbségéből, metszetéből komplexebb 3D formát állítunk elő.

Képek generálása

A modellről generált képek előállítását, szintézisét *renderelés*nek (*rendering*) nevezzük. Matematikai számítások alapján meghatározzuk a fénysugarak útját a különböző fényforrásoktól, ezek viselkedését a különböző objektumokkal való találkozáskor (az optikai tulajdonságuk függvényében), és végezetül mindezek hatását a megfigyelőre. Ugyanakkor az atmoszférikus effektusok (köd, füst, felhő, szórt háttérfény stb.), valamint a fényforrások alapján az objektumok által vetett árnyékokat is ki kell számítani.

A képek rendeltetésére több algoritmus ismeretes (Fiat, Phong, Gouraud, Metál, globális illumináció, radiosity), de ami a teljesítményt illeti, a *Ray-Tracing (sugárkövetés)* algoritmus a legjobb (a legpontosabb és legtermészetesebb képeket nyújtja), annak ellenére hogy talán a leglassúbb.

A renderelés négy tényező alapján valósul meg:

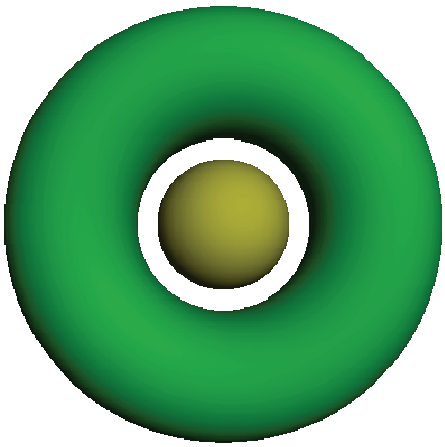
- 3D modellezés,
- a felület tulajdonságainak meghatározása (anyag, tükröződés, áttetszőség, fényesség stb.),
- a fényforrások és a kamera helyzete,
- animáció esetén az elmozdulási görbék, és a különböző objektumok eseményeinek (átmeneteinek) a meghatározása (pl. felület színének a változtatása).

A következő egyszerű példaprogram a POV-Ray 3D modellező leírónyelvben egy gömböt és egy tóruszt definiál. A program elején beállítjuk a kamera (megfigyelő) helyét és irányát, a háttérszínt és egy fényforrást is definiálunk.

A leírónyelv szintaxisa igen egyszerű, parancsokkal és paraméterekkel állíthatjuk be a színtér objektumait.

A POV-Ray rendszer a leírtak (program) alapján képet generál.

```
1.    #include "colors.inc"
2.
3.    camera {
4.        location <0, 0.1, -25>
5.        look_at 0
6.        angle 30
7.    }
8.
9.    background {color Gray75}
10.   light_source {<300, 300, -1000> White}
11.
12.   torus {
13.       3.5, 1.5
14.       rotate -90*x
15.       pigment {Green}
16.   }
17.
18.   sphere {
19.       <0, 0, 0>, 1.5
20.       texture {
21.         pigment {color Yellow}
22.       }
23.   }
```



2. ábra

Testek egyszerű modellje POV-Ray-ben

A képszintézis grafikus *csővezeték (graphics pipeline)* által valósul meg. Megjelenítés céljából a grafikus primitíveken végzendő elemi műveleteket (transzformációk, vetítés, vágás stb.) a rendszer egymás után, sorozatban (csővezetékben) végzi el.

Általánosan azt mondhatjuk, hogy a feldolgozás az alkalmazás–parancs–geometria–raszterizálás–textúrázás–fragmentálás–megjelenítés útvonalon történik.

Az *alkalmazás* szint tartalmazza a szimuláció leírását, az eseménykezelést, az adatstruktúrákat, algoritmusokat, esetleges adatbázisokat, primitívek generálását és más eszközöket.

A *parancs* szint a végrehajtható, értelmezhető grafikus parancsokat tartalmazza.

A *geometria* szint a modell mértani leírását, a mértani műveleteket tartalmazza.

A *raszterizálás* által kapjuk meg a képernyőn ábrázolható pixeleket, a vektorgrafika átalakul pixelgrafikává.

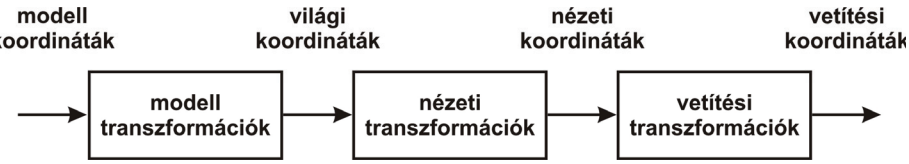
Ez tökéletesedik ki a *textúrázás*sal. A megrajzolt felületekre képeket húzhatunk rá.

A *fragmentálás*sal véglegesen eldől minden pixel színe. Itt alkalmazzuk az atmoszférikus effektusokat (pl. köd), itt határozzuk meg az átlátszóságot stb.

A folyamatot a *megjelenítés* zárja, az előállított kép megjelenik a képernyőn.

A parancs és a geometria szintet összevonva *objektum* vagy *vertex* szintnek nevezzük, a textúrázás és a fragmentálás szintet pedig *kép* vagy *fragmentum* szintnek nevezzük.

Általában minden szintnek más koordinátarendszere van, más tulajdonságok és műveletek érvényesek rá, és másképp támogatja a hardver. Ha a koordinátarendszereket és a transzformációkat szeretnénk csővezetékbe helyezni, akkor a 3. ábrán látható rendszert kapjuk.



3. ábra
Koordinátarendszerek és transzformációk

Szükség szerint a szinteket alszintekre bonthatjuk, ha egy-egy műveletet ki szeretnénk emelni, például a geometria 3D-ben *modell-transzformációk–triviális elvetés–illumináció–nézeti transzformációk–vágás–vetítés* alszintekre bontható.

A modellező szoftverek a hardver grafikus csővezetékére építve saját logikai csővezetékeket állíthatnak fel a saját funkcionalitásuk megvalósítása érdekében, így ezek esetenként eltérhetnek egymástól, csak nagyvonalakban követik az elvi csővezeték modellt.

Kovács Lehel