3D ábrázolás PoVRay (4. rész)

A PoVRay világában létrehozott testek és objektumok általában az origó környékén jöttek létre, illetve egy rájuk jellemző irányban és méretben jönnek létre. Ezen a problémán segít az affin transzformációnak nevezett matematikai eljárás, illetve ennek a barátságosabb megvalósítása.

A transzformációk három komponensből tevődnek össze: elmozgatás, átméretezés, elforgatás. Ez a három komponens egy közös affin mátrix alapján végezhető el *3D* testeken egy lépésben, amely egy összetett matematikai eljárás, s ezt is megoldották helyettünk a *PoVRay* programozói.

Elmozgatás

A legegyszerűbb transzformáció az elmozgatás, amely során a vektorban meghatározott irányba mozdítjuk el a testet, mégpedig a megadott mértékkel. Ez egyszerűen egy `translate` kulcsszót kell megadnunk a meghatározott vektorral. Az *1. ábrán* látott két gömb teljesen azonos méretű és azonos pozíciójú, egyetlen különbség közöttük, hogy a vörös színű testet egy egységgel eltoltuk az `<1,1,1>` vektor mentén.

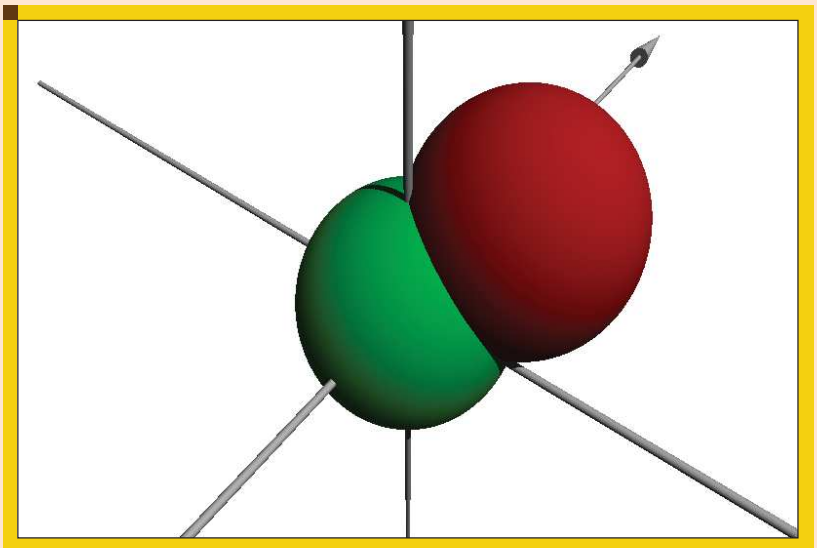
```
sphere{
 <0,0,0>,1.5
 texture{
  pigment{
   color Red}}
 translate <1,1,1>}

sphere{
 <0,0,0>,1.5
 texture{
  pigment{
   color Green}}}
```

Bár a példában a vörös gömb új pozíciója egyezik a megadott eltolással, az eltolás nem abszolút pozíciót jelent, hanem relatív pozíciót. Ennek megfelelően az alább meghatározott gömb pozíciója azonos lesz a fent meghatározott vörös gömb pozíciójával:

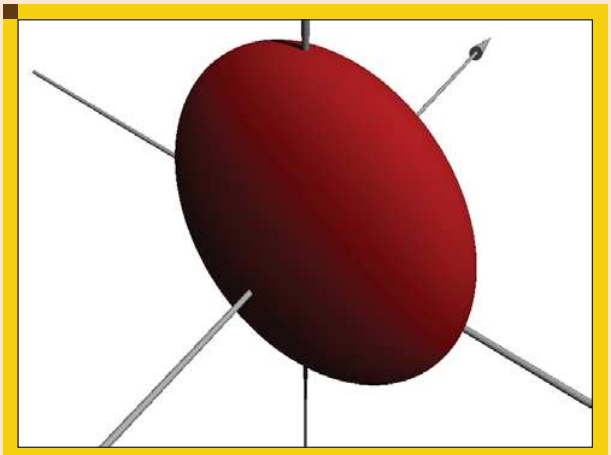
```
sphere{
 <2,2,2>,1.5
 texture{
  pigment{
   color Red}}
 translate <-1,-1,-1>}
```

Ha csak egy irányba kell elmozgatni egy testet, akkor tudjuk használni az előre definiált vektorokat is, tehát az alábbi részlet alapján a gömb új pozíciója a `<5,2,2>`:

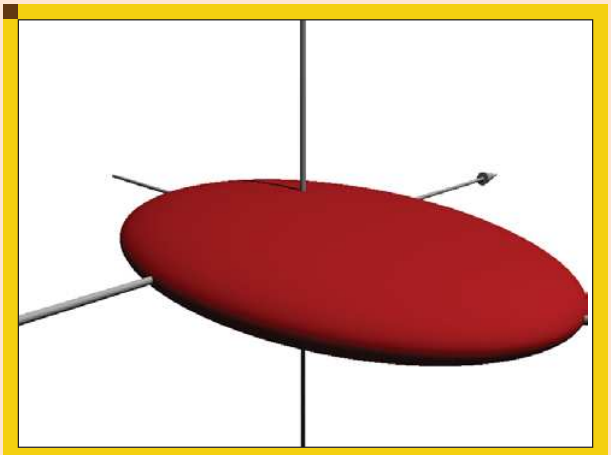
```
sphere{
 <2,2,2>,1.5
 texture{
  pigment{
   color Red}}
 translate 3*x}
```



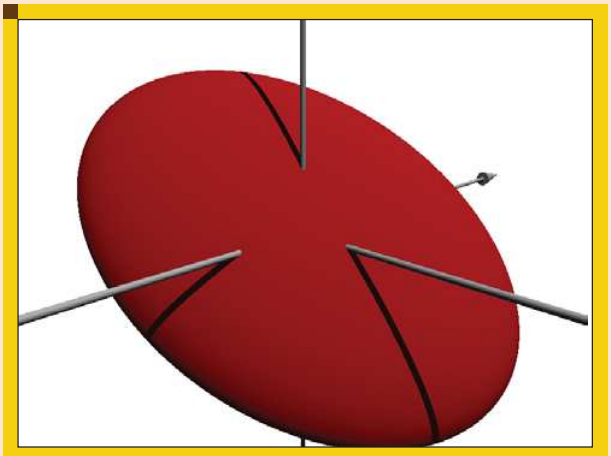
1. ábra Két gömb eltolással



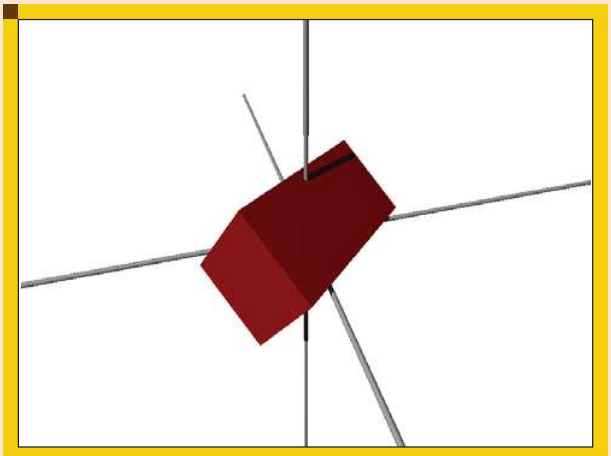
2. ábra Nyújtott gömb = ellipszoid



3. ábra Három tengely mentén átméretezett gömb



4. ábra Egy test elforgatása 30 fokkal



5. ábra Elforgatás X-Y-Z sorrenddel

Átméretezés

Az előző rész megismerése során biztos felmerült a kérdés, hogy miért nincs ovális keresztmetszetű test, más néven ellipszoid, amely egy térben megforgatott ellipszis alapján jön létre. Ennek az az egyszerű oka, hogy az ellipszoid a gömb megnyújtása az egyik tengely mentén, példánkban ez az X tengely (*2. ábra*):

```
sphere{
 <0,0,0>,1.5
 texture{
  pigment{
   color Red}}
 scale <2,1,1>}
```

Ha a megadott vektorban egynél nagyobb számot adunk meg, akkor megnyúlik a test, ha egynél kisebb számot, akkor összenyomódik. A megadott vektor abszolút értéke számít, nyugodtan megadhatunk akár negatív számot is.

```
sphere{
 <0,0,0>,1.5
 texture{
  pigment{
   color Red}}
 scale <3,0.5,2>}
```

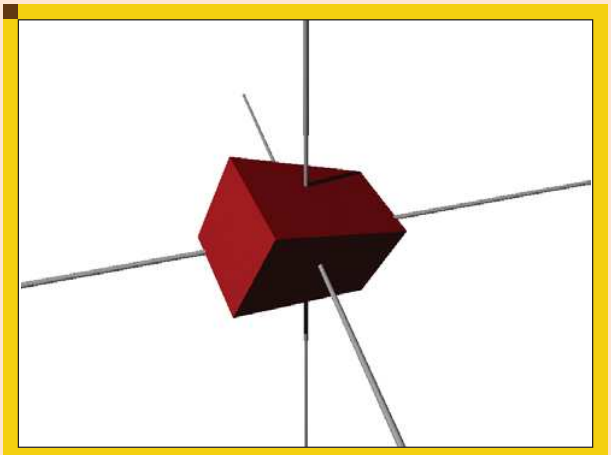
A *3. ábrán* egy gömbből kiindulva egy lapos, kissé ovális korongot készítettünk, egyszerűen az X tengely mentén 3 egységgel megnagyítottuk, az Y tengely mentén felére zsugorítottuk, s a Z tengely mentén 2 egységgel széthúztuk.

Elforgatás

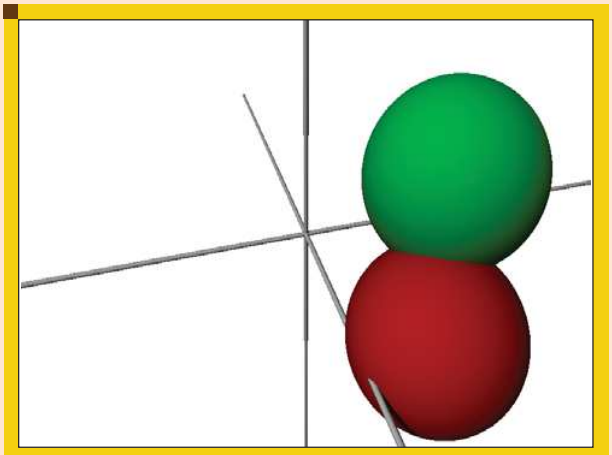
A három transzformáció közül a leginkább problémás, s így a legtöbb gondolkodást igénylő művelet az elforgatás. Például a *3. ábrán* látható testet szeretnénk elforgatni 30 fokkal minden tengely mentén, akkor a következőt kell használnunk:

```
sphere{
 <0,0,0>,1.5
 texture{
  pigment{
   color Red}}
 scale <3,0.5,2>
 rotate <-30,-30,-30>}
```

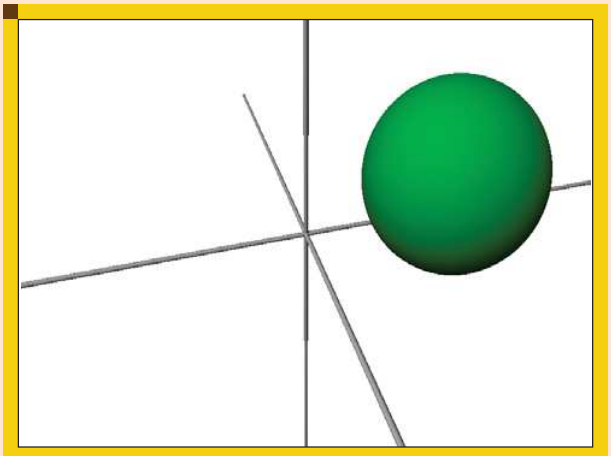
A forgatást megtehetjük több lépésben is, mivel a *PoVRay* az egy vektorban megadott értékek szerint először az X, aztán az Y, s végül a Z tengely körül forgatja meg a testet. Ez forgásszimmetrikus testek esetén nem nagy probléma, de más sorrenddel más-más eredményt kaphatunk (*5.* és *6. ábra*).



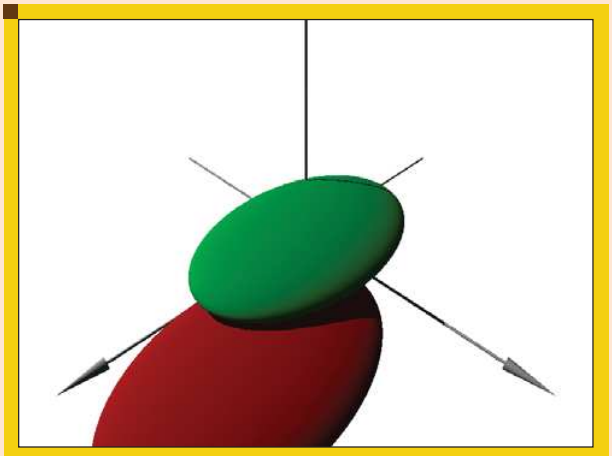
6. ábra Elforgatás Y-X-Z sorrenddel



7. ábra Gömb elforgatása eltolás nélkül



8. ábra Gömb elforgatása eltolással



9. ábra Gömb átméretezése

Ha a PoVRay programra hagyjuk a sorrendet:

```
box{
 <-1,-2,-1>,<1,2,1>
 texture{
  pigment{
   color Red}}
 rotate <-60,-30,-45>}
```

Ezzel szemben saját sorrend esetén a megfelelő sorokat megismételjük, és nullát írunk azon helyekre, amely tengelyek körül nem akarunk forgatni, a sorrend a példában Y-X-Z tengely:

```
box{
 <-1,-2,-1>,<1,2,1>
 texture{
  pigment{
   color Red}}
 rotate <0,-30,0>
 rotate <-60,0,0>
 rotate <0,0,-45>}
```

A különbség – ha nem is látványos – avatatlan szemnek is észrevehető, s az elforgatott tárgy újbóli elforgatásából adódik.

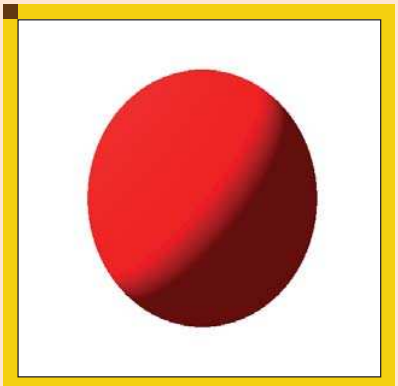
Fontos tudnivalók

Arra kell figyelnünk, hogy az elforgatás nem a test középpontja körül történik, hanem az origó körül. Ez a tulajdonság az *7. ábrán* látható igazán, ahol definiáltunk egy zöld és egy vörös gömböt, amelyek közül a vöröset elforgattuk minden tengely körül 60 fokkal. Mivel a gömb középpontja nem az origóban volt, így a két gömb nem esik egybe

```
sphere{
 <2,2,2>,1.5
 texture{
  pigment{
   color Red}}
 rotate <-60,-60,-60>}
sphere{
 <2,2,2>,1.5
 texture{
  pigment{
   color Green}}}
```

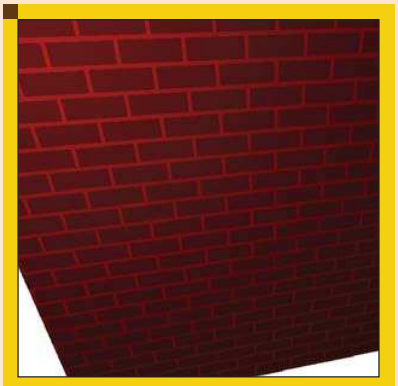
Ennek a problémának a kiküszöbölése egyszerűen annyi, hogy a gömböt az origóban hozzuk létre, majd az elforgatás után toljuk el a végleges helyére (*8. ábra*):

```
sphere{
 <0,0,0>,1.5
 texture{
  pigment{
   color Red}}
 rotate <-60,-60,-60>
 translate <2,2,2>}
```

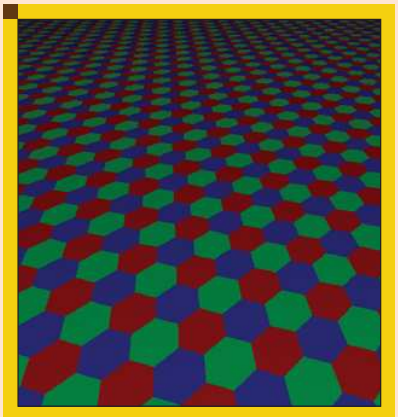


10. ábra Világító vörös szín

11. ábra „Sakktábla” felszín



12. ábra A „téglafal”



13. ábra Hatszög alakú csempék a padlón

A képen csak egy gömböt láthatunk, mivel a gömböt bármerre is forgatjuk: ugyan azt a kinézetű testet láthatjuk, mivel jelenleg csak színeket használtunk, textúrát nem. Célszerű minden testet az origóban létrehozni, majd forgatás és átméretezés után elmozgatni a végleges helyére. Ezzel sok munkától mentesülünk, ha később egy-egy testet el kell forgatnunk.
A fentihez hasonlóan az átméretezés sem a test középpontjához képest történik, hanem az origóhoz képest:

```
sphere{
 <2,2,2>,1.5
 texture{
  pigment{
   color Red}}
 scale <3,0.5,2>}


sphere{
 <0,0,0>,1.5
 texture{
  pigment{
   color Green}}
 scale <3,0.5,2>
 translate <2,2,2>}
```

Bár a *9. ábrán* látható két gömböt azonos pozícióba mozgattuk, az átméretezés okán már mérettel rendelkeztek.
A fentiek alapján a testeket mindig az origón hozzuk létre, végezzük el rajtuk az elforgatást és a méretezést, s csak a végén mozgassuk őket a végső helyükre. Ettől eltérő sorrend esetén nem biztos, hogy a kigondolt helyen és méretben találjuk meg a létrehozott testet.

Egyszerű textúrák

Eddig minden test egyszerű színekkel rendelkezett, amelyek – valljuk be – nem túl realisztikusak. De ne szaladjuk előre, nézzük meg az egyszerűbb színek és textúrák megadási lehetőségeit.
Mint azt már megtapasztaltuk, a színek is a textúra egyfajta megadása, mégpedig a pigment kulcsszó vezeti be őket, majd egy három elemű vektor követi, amely a szín vörös, zöld, illetve kék összetevőjét jelenti:

```
texture{
 pigment{
  color <1.0,0,0>}}
```

Normál esetben a színkomponensek 0.0 és 1.0 közötti értéket vesznek fel, de a ***PoVRay*** lehetőséged ad negatív illetve egynél nagyobb számok megadására is. Negatív szám esetén az adott szín ***nagyon*** sötét lesz, így erősen megvilágítva sem mutat az adott színre vajmi hajlandóságot. Egynél nagyobb színkomponens esetén a test az adott színnel szinte ***világítani*** fog, bár ez a saját fény más tárgyakra nem vetül rá (*10. ábra*):

```
texture{
 pigment{
  color <4.0,0.2,0.2>}}
```

A szín megadásakor egy apró trükkel könnyedén tudunk készíteni sakktáblához hasonló felszínt (*11. ábra*):

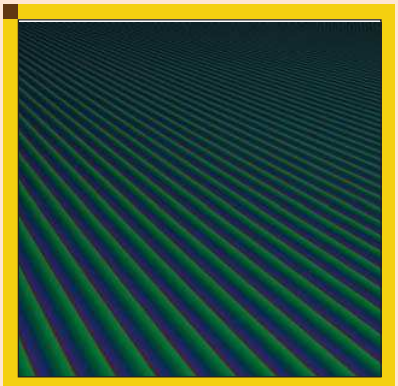
```
texture{
 pigment{
  checker color Red, color
Blue}}}
```

Mivel a gömb felszínére nem feszíthető olyan „sakktábla”, amely mindenhol szépen illeszkedik, kicsit összetörve látjuk viszont az erőfeszítésünket.
Tudunk készíteni „téglafalat” is, ha a `brick` kulcsszót használjuk a színek megadásakor. Ekkor a program a megadott színnel rajzolja a fugákat, s ennél kisebb fényerővel ábrázolja téglákat (*12. ábra*):

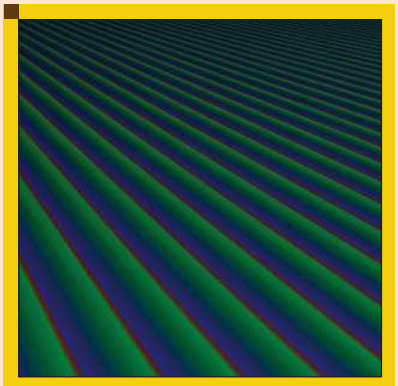
```
texture{
 pigment{
  brick color <1.0,0.0,0.0>}}
```

Megtölthetjük egy test felületét akár hatszögekkel is, amelyeket a program három színnel ábrázol, mégpedig úgy, hogy azonos szín nem lehet szomszédos hatszög (*13. ábra*):

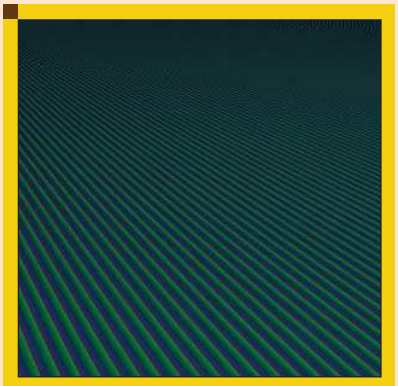
```
texture{
 pigment{
  hexagon color Blue, color
Red, color Green}}
```



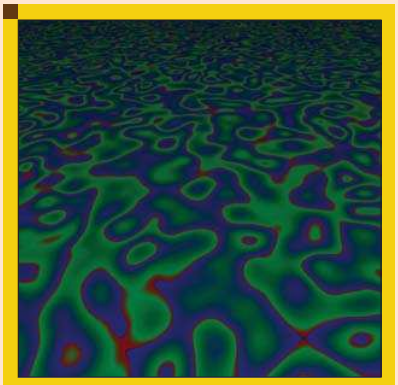
14. ábra Színátmenetek



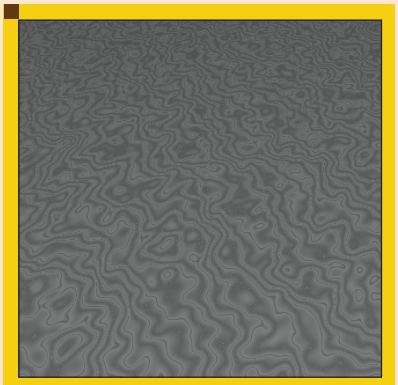
15. ábra Színátmenetek transzformálva



16. ábra Színátmenetek sűrítve



17. ábra „Leopárd-bőr" textúra



18. ábra „Márvány" textúra

19. ábra „Agate" textúra

Kicsit variálva a színekkel képesek leszünk színátmenetekre (`color_map`) is, amelyek egymás követik a gradiensnek megadott irányban (*14. ábra*):

```
texture{
 pigment{
  gradient x
  color_map{
   [0.00 color Red]
   [0.11 color Green]
   [0.99 color Blue]
   [1.00 color Red]}}}
```

A színek előtt megadott szám nulla és egy között lehet, s ezen az egy egységen belül határozza meg az átmenet méretét. A cikk első felében megismert transzformációk alkalmazhatók a textúrákra is, így képesek leszünk megváltoztatni a testek felszínét (*15. ábra*):

```
texture{
 pigment{
  gradient x
  color_map{
   [0.00 color Red]
   [0.11 color Green]
   [0.99 color Blue]
   [1.00 color Red]}
  rotate <45,0,0>
  scale 2.0}}
```

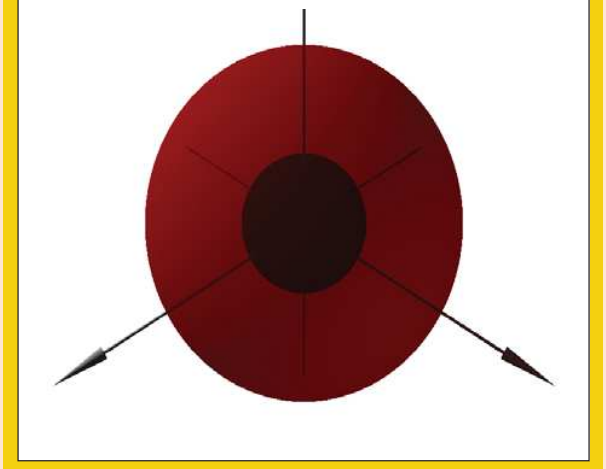
Van néhány módosító kulcsszó a textúráknál, amelyek érdekesebbé vagy mássá varázsolják a felületet. A legelső ilyen a `frequency` is, amelynek hatására a színátmenet sűrűbb vagy ritkább lesz. Ez a módosító ugyanis azt határozza meg, hogy egységnyi távolságon mennyi alkalommal ismétlődjön a megadott minta (*16. ábra*):

```
texture{
 pigment{
  gradient x
  frequency 4
  color_map{
   [0.00 color Red]
   [0.11 color Green]
   [0.99 color Blue]
   [1.00 color Red]}
  rotate <45,0,0>
  scale 2.0}}
```

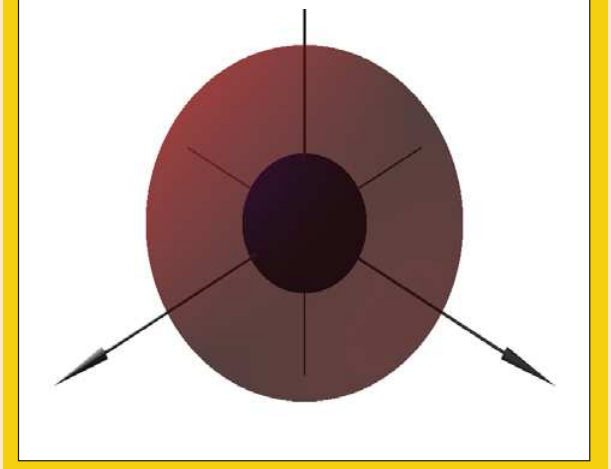
Ha a gradiens helyett kicsit összekuszáljuk a textúrát (`bozo`), akkor egyfajta leopárdmintát fogunk eredményül kapni (kiválóan alkalmazható vízfelszínhez is):

```
texture{
 pigment{
  bozo
  frequency 4
  color_map{
   [0.00 color Red]
   [0.11 color Green]
   [0.99 color Blue]
   [1.00 color Red]}
  rotate <45,0,0>
  scale 2.0}}
```

Lehetőségünk van „márványosítani" a színátmenetet, amely jól jön a faerezet és a kőmintázatok elkészítésekor (*18. ábra*). Az `omega`, a lambda valamint az `octaves` kulcsszavak



20. ábra Szűrő átlátszóság



21. ábra Áteresztő átlátszóság

a turbulencia paraméterezésére szolgálnak, más értékekkel másképp alakul a mintázat:

```
texture{
 pigment{
  marble
  frequency 4
  turbulence 0.5
  omega 0.8
  octaves 5.0
  lambda 1.5
  color_map{
   [0.00 color Gray]
   [0.11 color White]
   [0.99 color Gray]
   [1.00 color White]}
  rotate <45,0,0>
  scale 2.0}}
```

Valahol a `bozo` és `marble` között helyezkedik el az `agate`, amely egyfajta keveréke a kettőnek:

```
texture{
 pigment{
  agate
  frequency 4
  turbulence 0.5
  omega 0.8
  octaves 5.0
  lambda 1.5
  color_map{
   [0.00 color Gray]
   [0.11 color White]
   [0.99 color Gray]
   [1.00 color White]}
  rotate <45,0,0>
  scale 2.0}}
```

Átlátszóság

A testek lehetnek áttetszőek vagy átlátszóak, mint a víz vagy az üveg. Ez a tulajdonság nem okoz azonban automatikusan fénytörést, csak úgy tesz a tárgy, mintha áteresztené a mögötte lévő másik tárgyak fényét. A ***PoVRay*** megkülönböztet áteresztő (***transmittance***) és szűrő (***filtering transparency***) átlátszóságot. A kettő közötti különbséget képeken látni a legjobban (*20-21. ábra*).
A szűrő átlátszóságot az `rgbf` kulcsszó vezeti be, illetve egy négy paraméterből álló vektor, amelynek az első három paramétere a szín három komponense, az utolsó paramétere pedig az átlátszóság (*20. ábra*):

```
sphere{
 <4,4,4>,4
 texture{
  pigment{
   color rgbf <1,0,0,0.5>}}}
```

A szűrés a test saját színére vonatkozik, így a vörös gömb teljes mértékben kiszűri a nem vörös színeket, így a mögötte lévő kék gömb fekete színűnek látszik. Ez igaz a fehér háttérre is, amelyből így csak a vörös komponens marad meg.
Az áteresztő átlátszóságot az előzőhöz hasonlóan az `rgbt` kulcsszóval tudjuk meghatározni.

```
sphere{
 <4,4,4>,4
 texture{
  pigment{
   color rgbt <1,0,0,0.5>}}}
```

Ebben az esetben a tárgy színe nem jelent szűrést, egyszerűen az átlátszóság függvényében átlátszóvá válik, így a mögötte lévő kék gömb színe is átlátszik.
Egészen eddig csak fényeket, színeket és formákat hoztunk létre, ideje a különféle anyagokra jellemző tulajdonságokat is modellezni: a fémek csillanjanak meg, a fának legyen szép erezete, az üvegben törjön meg a fény. Lássuk el textúrával a tárgyakat, de erről már a következő részben!

Auth Gábor
(auth.gabor@enaplo.hu)

Egy pécsi középiskolában informatikát és programozást oktat. Tíz éve botlott először a UNIX rendszerekbe, 7 év Linux használat után kapta el a FreeBSD lázat, amiből máig nem tudott kigyógyulni.

KAPCSOLÓDÓ CÍMEK

A PovRay projekt honlapja
➲ http://www.povray.org
A cikkben említett fájlok
➲ http://user.enaplo.hu/~auth.gabor/pov/