Departamento de Ciência da COMPUTAÇÃO

UDESC Joinville

POV-Ray

André Tavares da Silva

andre.silva@udesc.br

www.povray.org

Para você acessar o material do POV-Team, acesse o tutorial em www.povray.org

Sintaxe

- A Linguagem de Descrição de Cena do POV-Ray possui sintaxe similar à da Linguagem C, com comentários, estruturas definindo objetos, blocos definidos por { } e diretivas de inclusão de arquivos iniciadas por #include.
- Exemplo: iniciando com a inclusão de um arquivo de definição de cores:

```
#include "colors.inc"
```

Definindo câmera

- Para renderizar algo, é necessário definir a posição da janela.
- Para isso utiliza-se a definição de uma câmera, que possui implícitas as definições de ângulo de abertura para projeção em perspectiva, etc.

```
// Camera posicionada em x=0, y=2, z=-5 voltada para
// direção apontada pelo vetor x=0, y=1, z=2.
camera {
    location <0, 2, -5>
    look_at  <0, 1,  2>
}
```

Cor de fundo

- A cor de fundo representa a cor que será utilizada quando um raio não interceptar nenhum objeto.
- São utilizadas as definições de cores carregadas anteriormente (no include).

```
// Cor de fundo. Nao é objeto de cena.
background { color Black }
```

Fonte de Luz

- Para podermos visualizar objetos, é necessário que eles sejam iluminados por alguma fonte de luz.

```
// Fonte de Luz Branca posicionada
// em x=20, y=4 e z=-13

light_source { <20, 4, -13> color White}
```

- Este comando define uma fonte de luz pontual. POV-Ray possui muitos tipos de fontes de luz (veja manual).

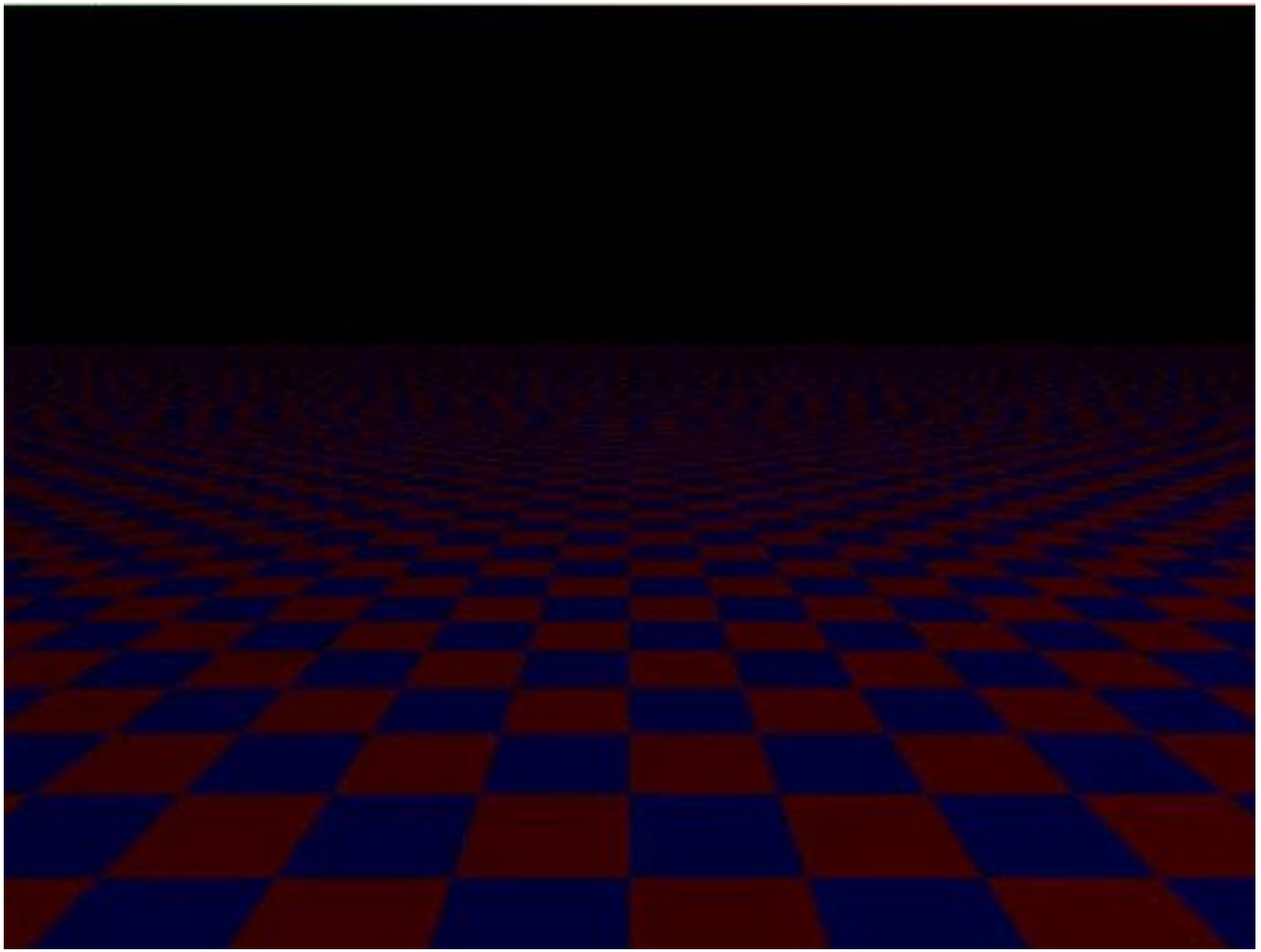
Objetos de Dimensões Infinitas

- Para definir "chão" de uma cena ou abóbada celeste, o POV-Ray oferece vários objetos geométricos de dimensões infinitas, como por exemplo, o plano:

```
// Plano com textura em xadrez para chão
plane { <0, 1, 0>, -1
    pigment {
        checker color Red, color Blue
    }
}
```

- O atributo *pigment* permite definir características de cor ou textura do objeto. Neste caso, o atributo *checker* define uma estrutura xadrez e não depende de uma textura definida como *bitmap*, apenas da definição das suas duas cores.

Objetos de Dimensões Infinitas

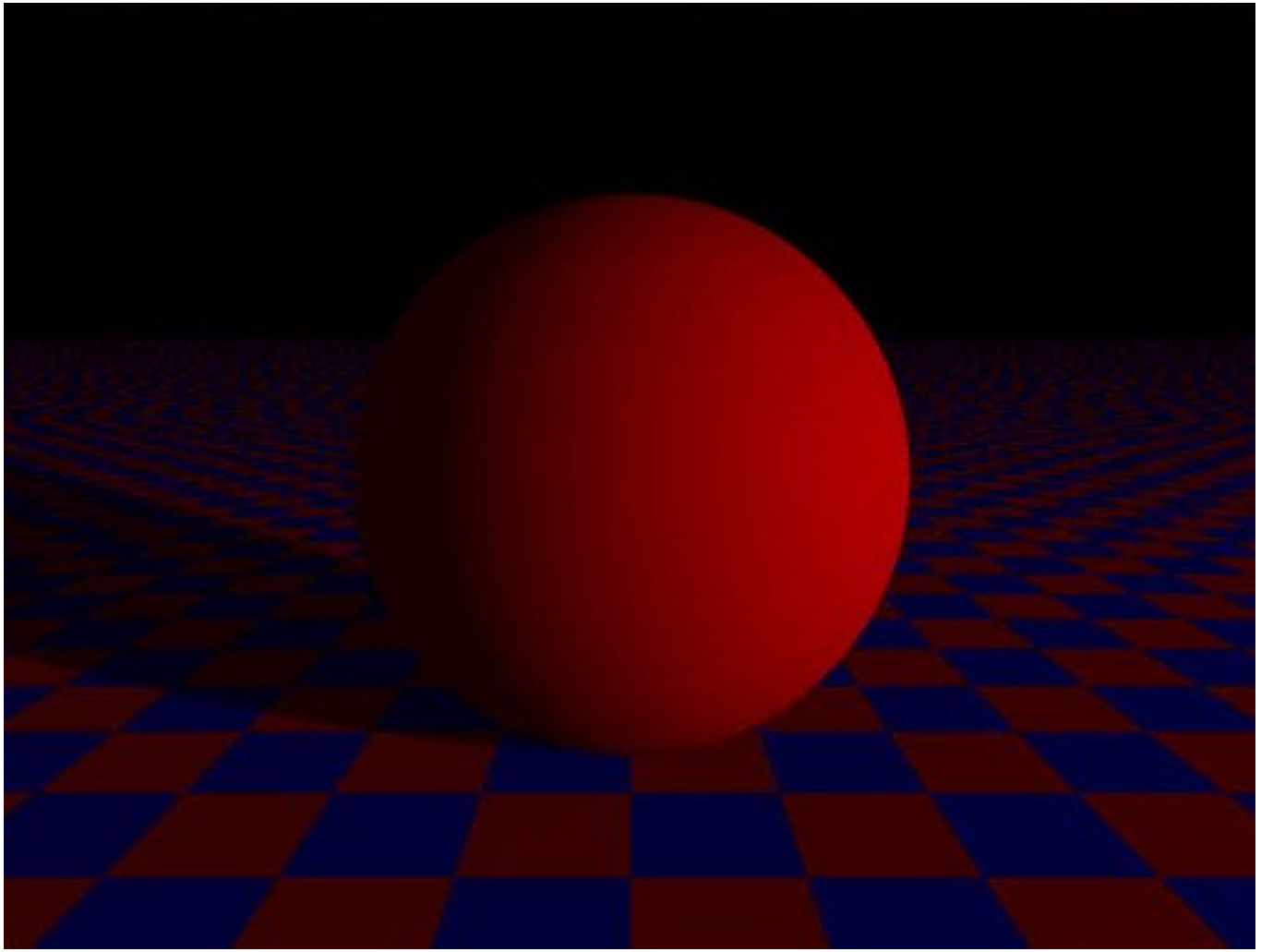


Objetos Geométricos Simples

- POV-Ray define vários objetos descritos por primitivas geométricas, como esferas, cilindros, cubos, cones, etc.

```
// Esfera em x=0, y=1, z=2 com raio=2
sphere {
    <0, 1, 2>, 2
    texture {
      pigment {
        color Red
      }
    }
  }
```

Objetos Geométricos Simples



Objetos Geométricos Simples

- Não foi definido nenhum tipo de reflectividade ao objeto e ele portanto possui somente reflexão difusa, fácil de ver pela forma como o objeto reage à luz, que vem da direita e ilumina a face direita do objeto, ficando mais escuro à medida que se avança para seu lado esquerdo.
- Observe que o POV-Ray também renderiza a sombra do objeto sobre o plano xadrez definido anteriormente.

Aparência e Textura

- POV-Ray permite vários tipos de "acabamento" para os objetos da cena. Os principais são texturas e definição de características de reflectância.
- Para incluir no início um arquivo de definição de aparência e de texturas que já vem com o POV-Ray:

```
#include "textures.inc"
```

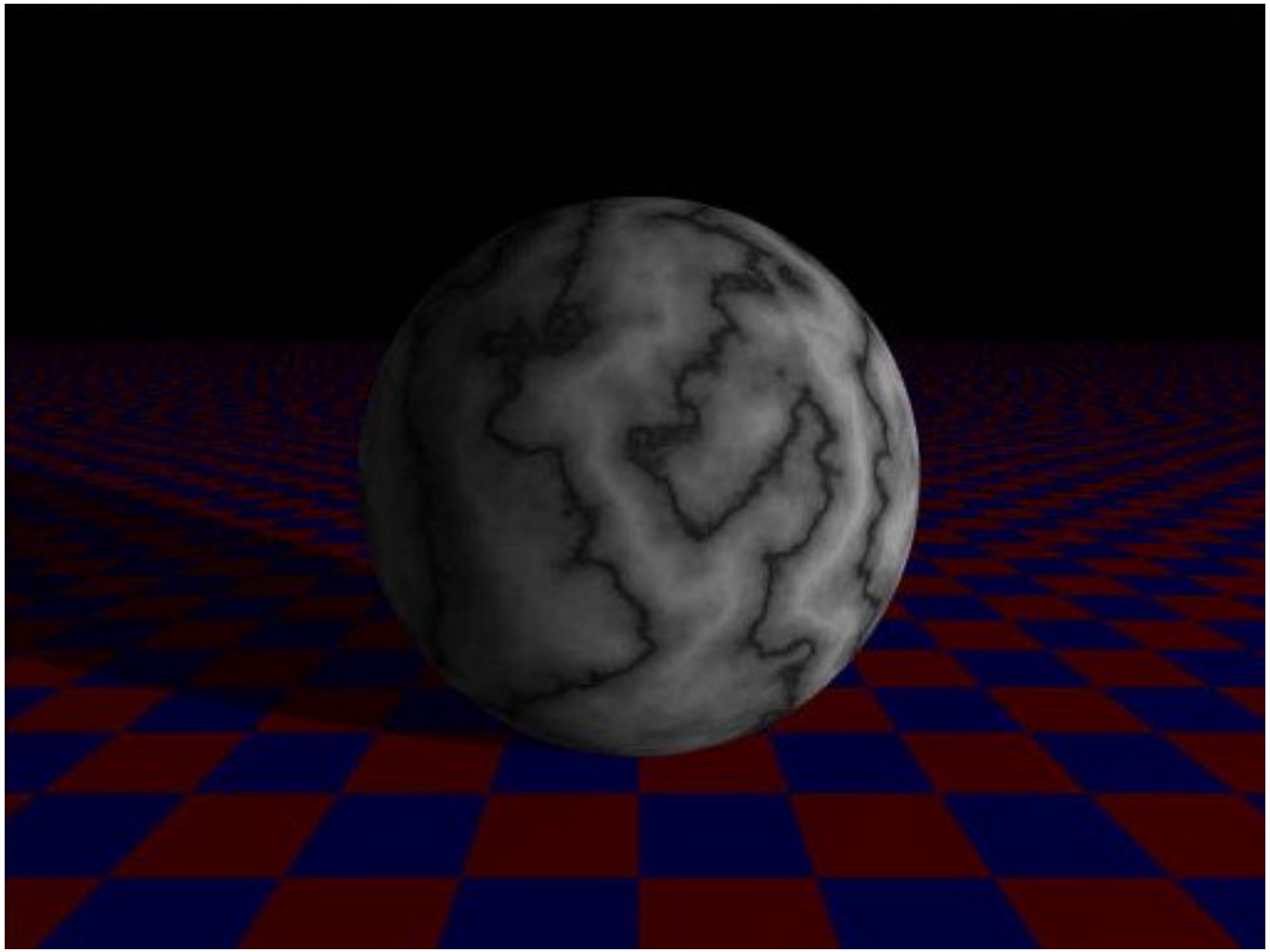
- Obs.: veja os arquivos no diretório include do POV-Ray.

Aparência e Textura

- Para dar à esfera uma aparência de madrepérola, redefinido a sua textura usando "stones":

```
// Esfera em x=0, y=1, z=2 com raio=2
sphere {
   <0, 1, 2>, 2
   texture {
     pigment {
       White_Marble // em textures.inc
       scale 1   // fator de escala da textura
     }
   }
}
```

Aparência e Textura

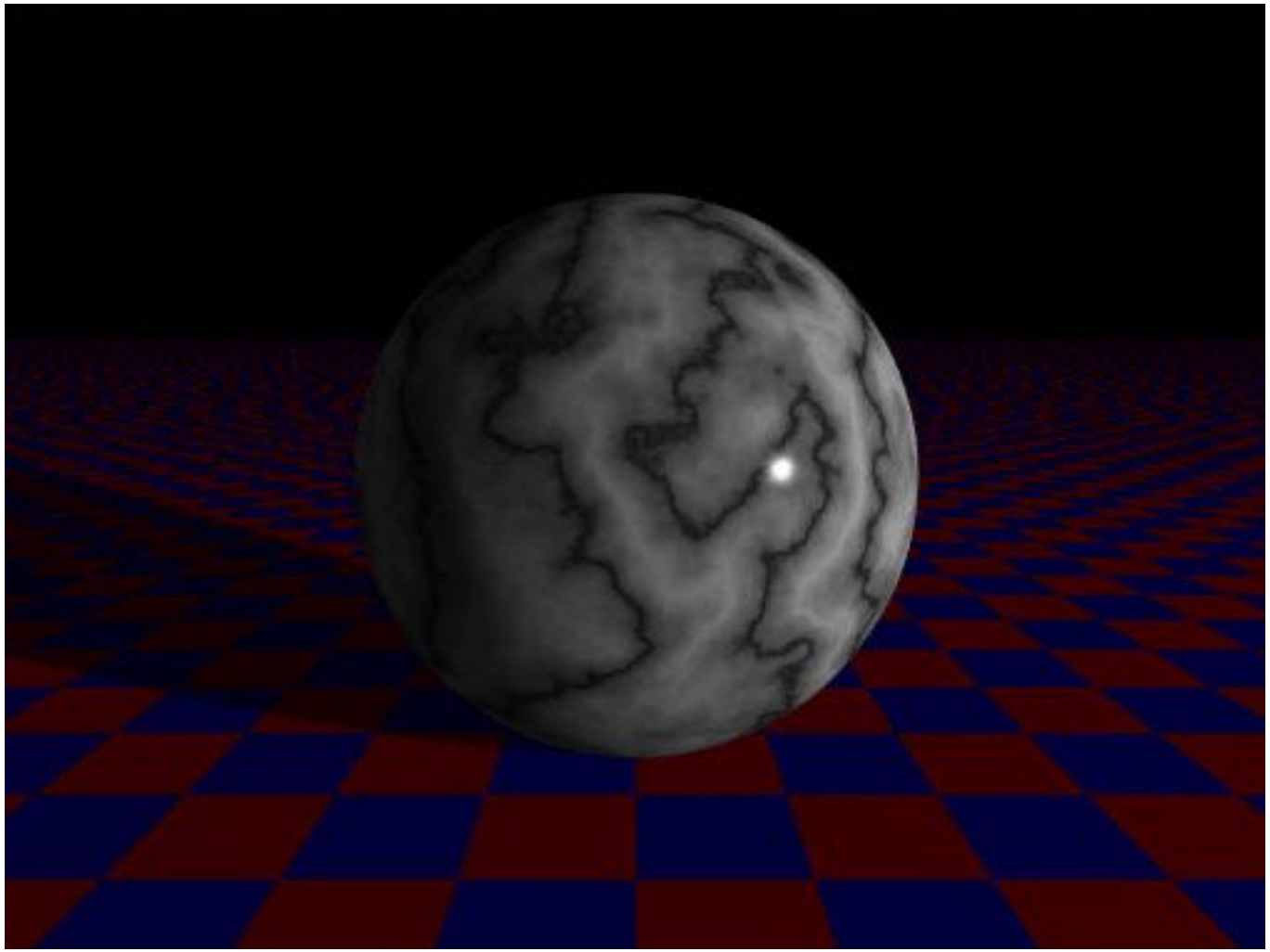


Aparência e Textura

- O que nos falta é realismo na parte de reflexos de luz. Uma bola de pedra é polida e brilhante.

```
// Esfera em x=0, y=1, z=2 com raio=2
sphere {
   <0, 1, 2>, 2
   texture {
     pigment {
       White_Marble // em textures.inc
       scale 1   // fator de escala da textura
     }
     finish { Shiny } // em finish.inc
   }
}
```

Aparência e Textura

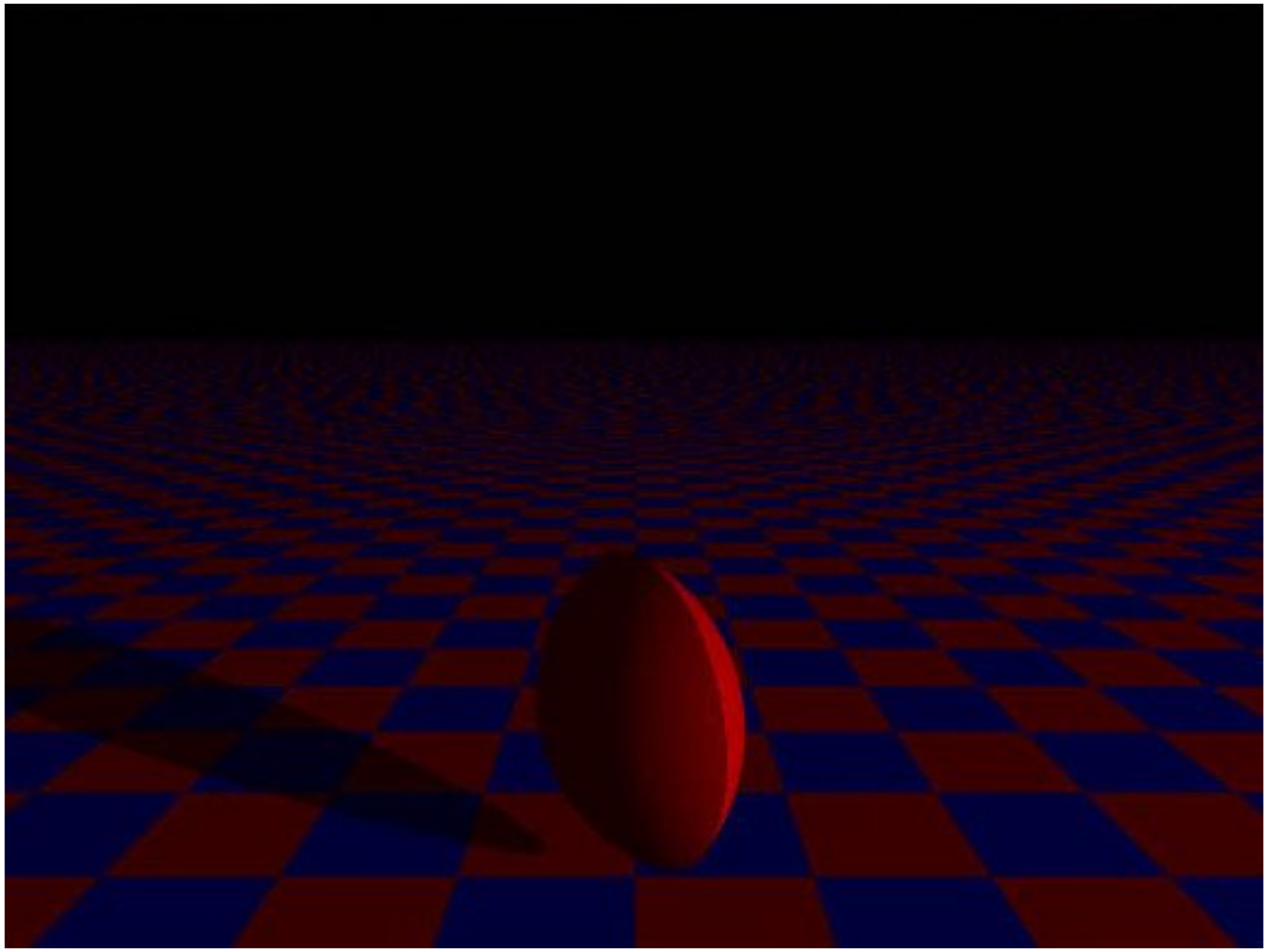


CSG

- Vimos a CSG nas aulas de Modelagem 3D. POV-Ray dá suporte completo à CSG, possibilitando a criação de diversos objetos.
- Exemplo de operação de intersecção:

```
intersection {
   sphere { <0, 0, 0>, 1
      translate -0.5*x
   }
   sphere { <0, 0, 0>, 1
      translate 0.5*x
   }
   pigment { Red }
   rotate -30*y   // para visualizar disco "de lado"
   finish { Shiny }
}
```

CSG

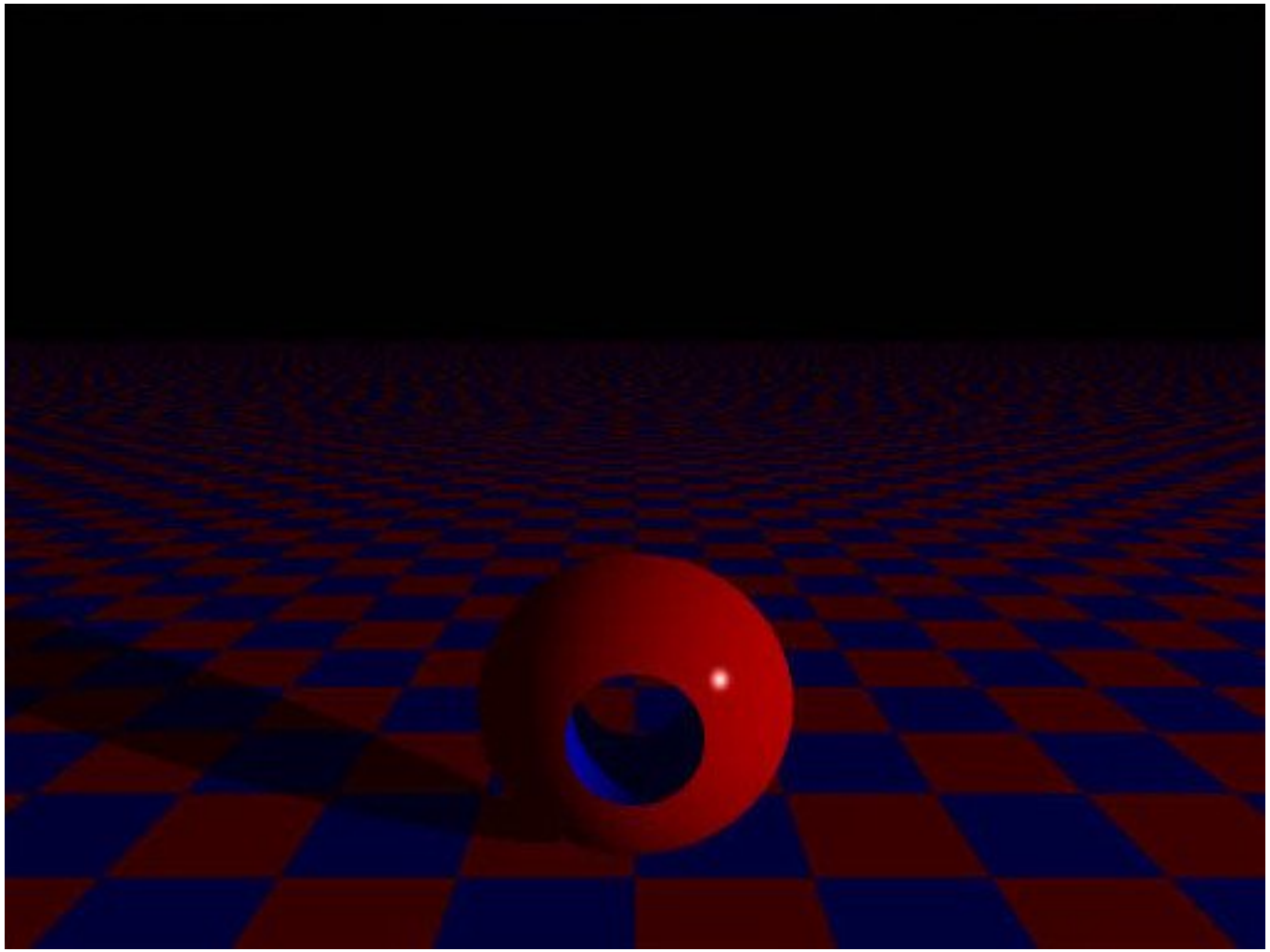


CSG

- Fazendo esta intersecção, mas com rotação de 90 graus para deixar a pastilha de frente, vamos calcular a diferença dela com um cilindro azul:

```
difference {
   intersection {
      sphere { <0, 0, 0>, 1
         translate -0.5*x
      }
      sphere { <0, 0, 0>, 1
         translate 0.5*x
      }
      pigment { Red }
      rotate 90*y
      finish { Shiny }
   }
   cylinder { <0, 0, -1> <0, 0, 1>, .35
      pigment { Blue }
   }
}
```

CSG

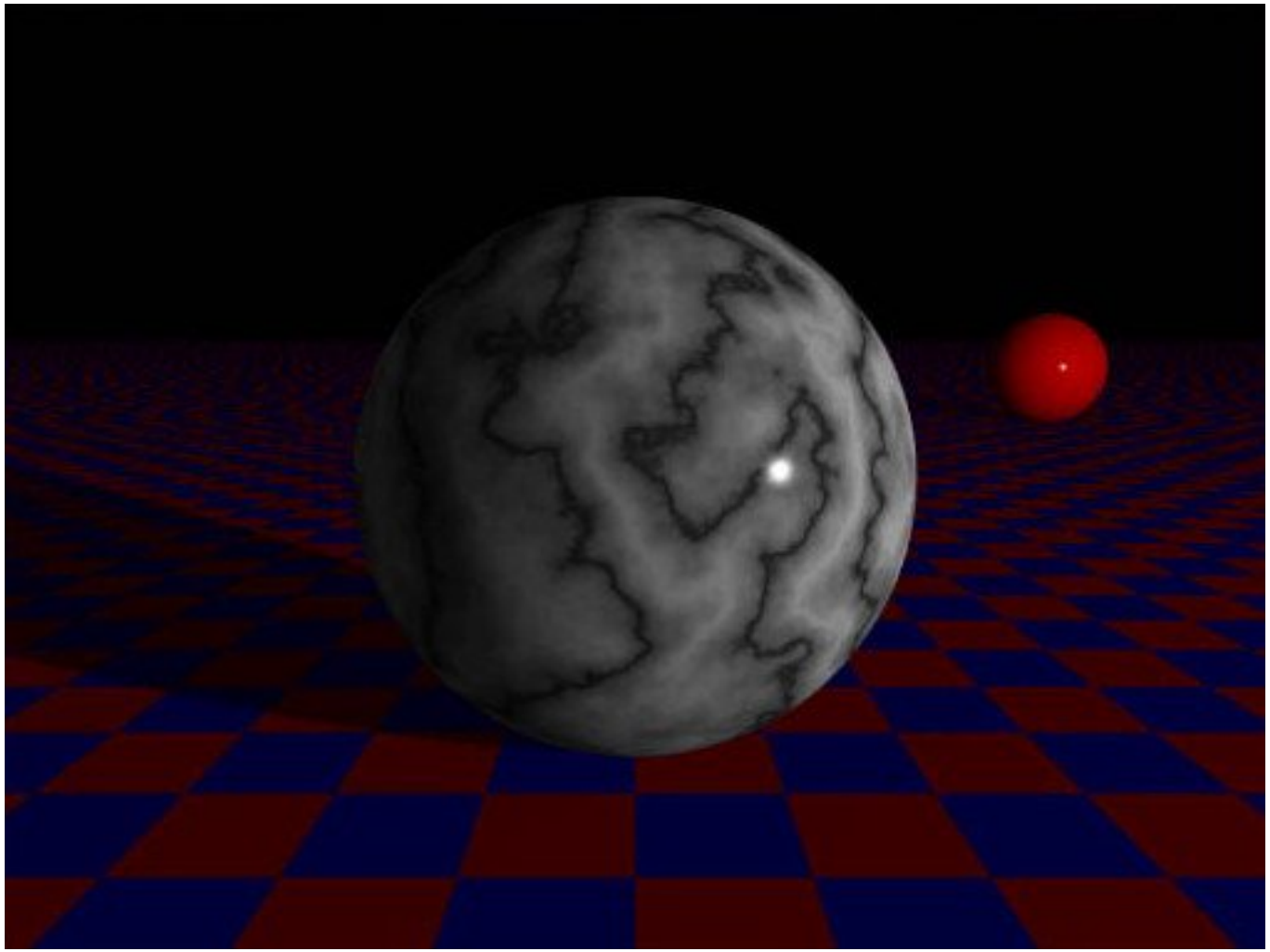


Efeitos Atmosféricos

- Através do de uma implementação simples do método da radiância, POV-Ray pode simular muitos efeitos atmosféricos, como névoa, fumaça, etc.

```
// Esfera ao fundo para mais tarde demonstrar efeitos de nevoa
sphere {
   <15, 1, 30>, 2
   texture {
      pigment {Red}
      finish { Shiny }
   }
}
```

Efeitos Atmosféricos

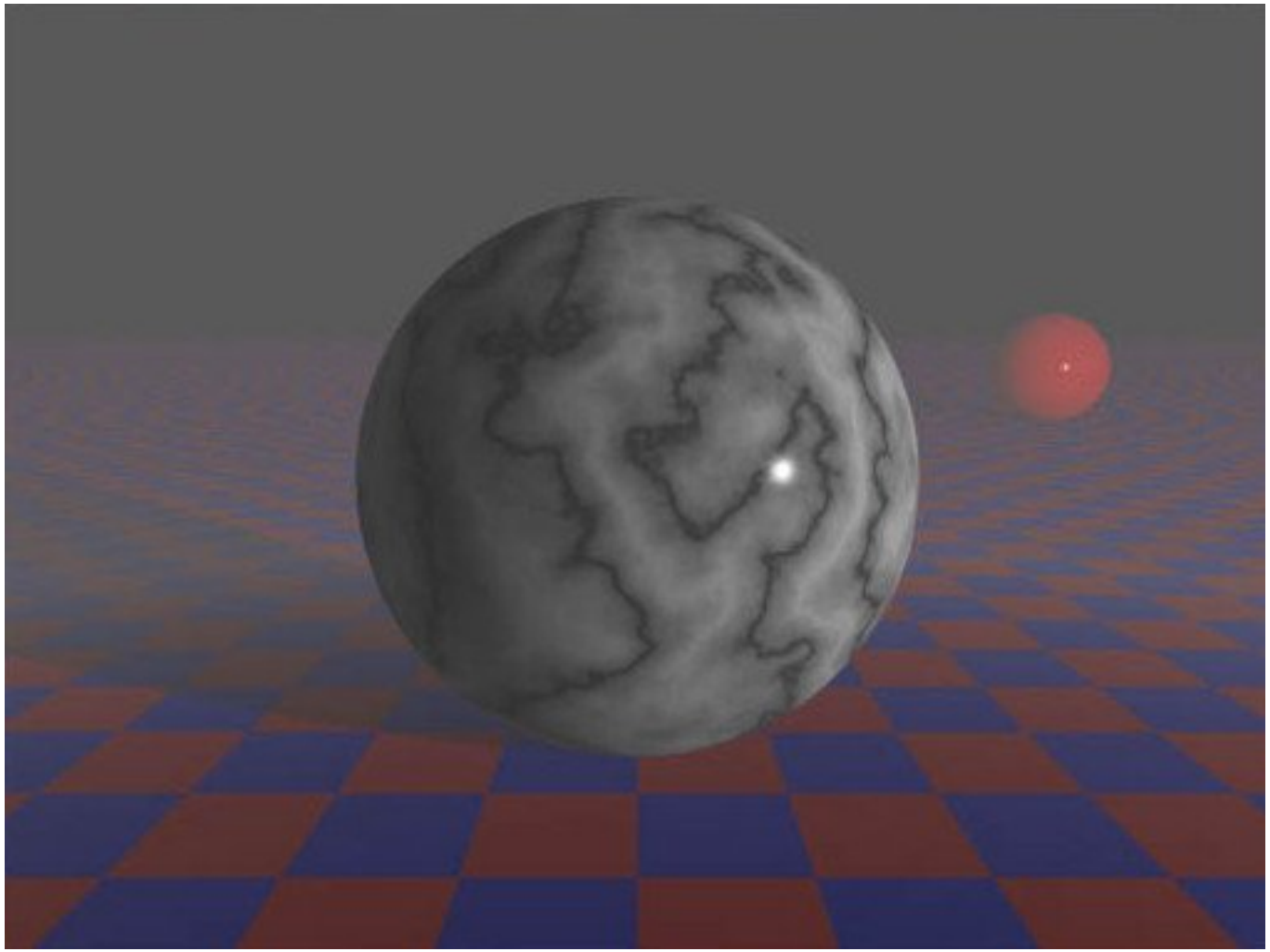


Efeitos Atmosféricos

- Definindo uma frente de névoa:

```
// Efeito de névoa iniciando-se a 20 unidades
// da câmera com cor cinza 70% e transparência
//50% (definida em RGBA - POV-Ray chama de RGBT)
fog {
    distance 20
    color rgbt <0.7 0.7 0.7 0.5>
}
```

Efeitos Atmosféricos



Efeitos Atmosféricos

- O resultado é um fundo enevoado, com a segunda esfera envolta na névoa.

- Observe que por causa do efeito de radiância, as sombras se tornam menos nítidas, como naturalmente acontece.

Malha de Polígonos / Triângulos

- Para a definição de objetos "facetados" descritos por malhas de polígonos, POV-Ray possui três objetos:
  - ***mesh*** - para triângulos
  - ***mesh2*** - para triângulos gerados por programas de construção geométrica ou reconsrução 3D, é mais rápido do que mesh mas a sintaxe é complicada de se fazer "na mão"
  - ***polygon*** - para malhas de polígonos de qualquer número de pontos.

Malha de Polígonos / Triângulos

```
polygon {
   30,
   <-0.8, 0.0>, <-0.8, 1.0>,    // Letter "P"
   <-0.3, 1.0>, <-0.3, 0.5>,    // outer shape
   <-0.7, 0.5>, <-0.7, 0.0>,
   <-0.8, 0.0>,
   <-0.7, 0.6>, <-0.7, 0.9>,    // hole
   <-0.4, 0.9>, <-0.4, 0.6>,
   <-0.7, 0.6>
   <-0.25, 0.0>, <-0.25, 1.0>,  // Letter "O"
   < 0.25, 1.0>, < 0.25, 0.0>,  // outer shape
   <-0.25, 0.0>,
   <-0.15, 0.1>, <-0.15, 0.9>,  // hole
   < 0.15, 0.9>, < 0.15, 0.1>,
   <-0.15, 0.1>,
   <0.45, 0.0>, <0.30, 1.0>,    // Letter "V"
   <0.40, 1.0>, <0.55, 0.1>,
   <0.70, 1.0>, <0.80, 1.0>,
   <0.65, 0.0>,
   <0.45, 0.0>
   pigment { color rgb <1, 0, 0> }
   translate x*2
}
```

Malha de Polígonos / Triângulos