



# miniRT

Meu primeiro RayTracer com miniLibX

*Resumo: Este projeto é uma introdução ao belo mundo do Raytracing. Depois de concluído, você será capaz de renderizar imagens simples geradas por computador e você nunca mais terá medo de implementar fórmulas matemáticas novamente.*

# **Conteúdo**

<b>eu</b>	<b>Introdução</b>	<b>2</b>
<b>II</b>	<b>Instruções Comuns</b>	<b>3</b>
<b>III</b>	<b>Parte obrigatória - parte do</b>	<b>4</b>
<b>4</b>	<b>bônus miniRT</b>	<b>9</b>
<b>V</b>	<b>Exemplos</b>	<b>11</b>

# Capítulo I

## Introdução

Quando se trata de renderizar imagens tridimensionais geradas por computador, existem 2 abordagens possíveis: " Rasterização ", Que é usado por quase todos os motores gráficos devido à sua eficiência e" Ray Tracing. "

O " Ray Tracing "Método, desenvolvido pela primeira vez em 1968 (mas aprimorado desde então) é ainda hoje mais caro em computação do que o" Rasterização "Método. Como resultado, não é bem adaptado a casos de uso em tempo real, mas produz um grau muito mais alto de realismo visual.



Figura I.1: As imagens acima são renderizadas com a técnica de rastreamento de raio. Impressionante, não é?

Antes mesmo de começar a produzir gráficos de alta qualidade, você deve dominar o básico: o miniRT é o seu first ray tracer codificado em C, normalizado e humilde, mas funcional.

O objetivo principal de miniRT é para provar a si mesmo que você é capaz de implementar quaisquer fórmulas matemáticas ou físicas sem ser um matemático, nós apenas implementaremos os recursos básicos de traçado de raios aqui, então mantenha a calma, respire fundo e não entre em pânico! Após este projeto, você poderá mostrar fotos de boa aparência para justificar a quantidade de horas que passa na escola.

# Capítulo II

## Instruções Comuns

- Seu projeto deve ser escrito de acordo com a Norma. Se você tiver arquivos / funções de bônus, eles serão incluídos na verificação de normas e você receberá um 0 se houver um erro de norma interno.
- Suas funções não devem encerrar inesperadamente (falha de segmentação, erro de barramento, double free, etc) além de comportamentos indefinidos. Se isso acontecer, seu projeto será considerado não funcional e receberá um 0 durante a avaliação.
- Todo o espaço de memória alocado no heap deve ser liberado adequadamente quando necessário. Nenhum vazamento será tolerado.
- Se o assunto exigir, você deve enviar um Makefile que irá compilar seus arquivos de origem para a saída necessária com as bandeiras -Wall, -Wextra e -Werror, e seu arquivo Make não deve vincular novamente.
- Seu Makefile deve pelo menos conter as regras \$(NOME), tudo, limpo, limpo e ré.
- Para entregar bônus ao seu projeto, você deve incluir uma regra bônus ao seu arquivo Make, que irá adicionar todos os vários cabeçalhos, bibliotecas ou funções que são proibidas na parte principal do projeto. Os bônus devem estar em um arquivo diferente \_bônus. {c / h}.  
A avaliação da parte obrigatória e da parte bônus é feita separadamente.
- Se o seu projeto permite que você use o seu libft, você deve copiar suas fontes e seus associados Makefile em um libft pasta com seu arquivo Make associado. Do seu projeto Makefile deve compilar a biblioteca usando seu Makefile, em seguida, compile o projeto.
- Nós encorajamos você a criar programas de teste para o seu projeto, embora este trabalho **não terá que ser enviado e não será avaliado**. Isso lhe dará a chance de testar facilmente seu trabalho e o de seus colegas. Você achará esses testes especialmente úteis durante sua defesa. Na verdade, durante a defesa, você está livre para usar seus testes e / ou os testes do colega que está avaliando.
- Envie seu trabalho para o repositório git atribuído. Apenas o trabalho no repositório git será avaliado. Se Deepthought for designado para avaliar seu trabalho, isso será feito após as avaliações de seus pares. Se ocorrer um erro em qualquer seção do seu trabalho durante a avaliação do Deepthought, a avaliação será interrompida.

# Capítulo III

## Parte obrigatória - miniRT

<b>Nome do programa</b>	miniRT
<b>Entregar arquivos</b>	Todos os seus arquivos
<b>Criar arquivo</b>	all, clean, fclean, re, bonus a scene
<b>Argumentos</b>	in format * .rt
<b>Funções externas.</b>	<ul style="list-style-type: none"><li>• abrir, fechar, ler, escrever, imprimir, malloc, livre, perror, strerror, sair</li><li>• Todas as funções da biblioteca matemática (-lm man man 3 math)</li><li>• Todas as funções do MinilibX</li></ul>
<b>Libft autorizado</b>	sim
<b>Descrição</b>	O objetivo do seu programa é gerar imagens usando o protocolo Raytracing. Cada uma dessas imagens geradas por computador representará uma cena, vista de um ângulo e posição específicos, definidos por objetos geométricos simples, e cada uma com seu próprio sistema de iluminação.

As restrições são as seguintes:

- Vocês devem usar o miniLibX. A versão que está disponível no sistema operacional ou em suas fontes. Se você optar por trabalhar com as fontes, precisará aplicar as mesmas regras para o seu libft como aqueles escritos acima em Comum Instruções papel.
- O gerenciamento de sua janela deve permanecer suave: mudar para outra janela, minimizar, etc.
- Você precisa de pelo menos 5 objetos geométricos simples: plano, esfera, cilindro, quadrado e triângulo.

- Se aplicável, todas as intersecções possíveis e o interior do objeto devem ser manuseados corretamente.
- Seu programa deve ser capaz de redimensionar as propriedades exclusivas do objeto: diâmetro para uma esfera, tamanho do lado para um quadrado e a largura e altura para um cilindro.
- Seu programa deve ser capaz de aplicar a transformação de translação e rotação a objetos, luzes e câmeras (exceto para esferas, triângulos e luzes que não podem ser girados).
- Gerenciamento de luz: brilho do ponto, sombras fortes, iluminação ambiente (os objetos nunca estão completamente no escuro). As luzes coloridas e com vários pontos devem ser tratadas corretamente.
- No caso de Pensamento profundo tem olhos um dia para avaliar seu projeto e se você quiser ser capaz de renderizar lindos papéis de parede de mesa ...  
Em vez de abrir uma janela, seu programa deve salvar a imagem renderizada em bmp formato quando seu segundo argumento é " --Salve • ".
- Se nenhum segundo argumento for fornecido, o programa exibirá a imagem em uma janela e respeitará as seguintes regras:
  - Pressionando ESC deve fechar a janela e sair do programa de forma limpa.
  - Clicar na cruz vermelha na moldura da janela deve fechar a janela e sair do programa de forma limpa.
  - Se o tamanho declarado da cena for maior do que a resolução da tela, o tamanho da janela será definido de acordo com a resolução da tela atual.
  - Se houver mais de uma câmera, você deve ser capaz de alternar entre elas pressionando as teclas do teclado de sua escolha.
  - O uso de imagens do minilibX é fortemente recomendado.
- Seu programa deve tomar como primeiro argumento um arquivo de descrição de cena com o. rt extensão.
  - Irá conter a janela / imagem renderizada Tamanho, o que implica o seu miniRT deve ser capaz de renderizar em qualquer tamanho positivo.
  - Cada tipo de elemento pode ser separado por uma ou mais quebras de linha.
  - Cada tipo de informação de um elemento pode ser separado por um ou mais espaços.
  - Cada tipo de elemento pode ser definido em qualquer ordem no arquivo.
  - Elementos que são definidos por uma letra maiúscula só podem ser declarados uma vez na cena.

- Cada informação do primeiro elemento é o identificador de tipo (composto por um ou dois caracteres), seguido por todas as informações específicas para cada objeto em uma ordem estrita, como:

- \* Resolução:

```
R 1920 1080
```

- Identificador: **R**
- Taxa de resolução x
- Taxa de resolução y

- \* Iluminação ambiente:

```
A 0,2 255.255.255
```

- Identificador: **UMA**
- Relação de iluminação ambiente no intervalo [0,0,1,0]: **0,2**
- Cores R, G, B no intervalo [0-255]: **255, 255, 255**

- \* Câmera:

```
c -50,0,0,20 0,0,1 70
```

- Identificador: **c**
- Coordenadas x, y, z do ponto de vista: **0.0,0.0,20.6**
- Vetor de orientação 3D normalizado. No intervalo [-1,1] para cada eixo x, y, z: **0.0,0.0,1.0**
- FOV: Campo de visão horizontal em graus no intervalo [0,180] \*
- Luz:

```
eu -40,0,50,0,0,0,0,6 10.0.255
```

- Identificador: **eu**
- Coordenadas x, y, z do ponto de luz: **0.0,0.0,20.6**
- A taxa de brilho da luz no intervalo [0,0,1,0]: **0,6**
- Cores R, G, B no intervalo [0-255]: **10, 0, 255**

- \* Esfera:

```
sp 0,0,0,0,20.6 12.6 10.0.255
```

- Identificador: **sp**
- Coordenadas x, y, z do centro da esfera: **0.0,0.0,20.6**
- O diâmetro da esfera: **12,6**
- Cores R, G, B no intervalo [0-255]: **10, 0, 255**

\* Avião:

```
pl 0.0,0.0, -10.0 0.0,1.0,0.0 0,0,225
```

- Identificador: **pl**
- Coordenadas x, y, z: **0.0,0.0, -10.0**
- Vetor de orientação 3D normalizado. No intervalo [-1,1] para cada eixo x, y, z: **0.0,0.0,1.0**
- Cores R, G, B no intervalo [0-255]: **0, 0, 255**

\* Quadrado:

```
sq 0.0,0.0,20.6 1.0,0.0,0.0 12.6 255.0.255
```

- Identificador: **sq**
- Coordenadas x, y, z do centro do quadrado: **0.0,0.0,20.6**
- Vetor de orientação 3D normalizado. No intervalo [-1,1] para cada eixo x, y, z: **1.0,0.0,0.0**
- Tamanho lateral: **12.6**
- Cores R, G, B no intervalo [0-255]: **255, 0, 255**

\* Cilindro:

```
cy 50.0,0.0,20.6 0.0,0.0,1.0 10.0.255 14.2 21.42
```

- Identificador: **cy**
- Coordenadas x, y, z: **50.0,0.0,20.6**
- Vetor de orientação 3D normalizado. No intervalo [-1,1] para cada eixo x, y, z: **0.0,0.0,1.0**
- O diâmetro do cilindro: **14.2**
- A altura do cilindro: **21.42**
- Cores R, G, B no intervalo [0,255]: **10, 0, 255**

\* Triângulo:

```
tr 10.0,20.0,10.0 10.0,10.0,20.0 20.0,10.0,10.0 0,0,255
```

- Identificador: **tr**
- Coordenadas x, y, z do primeiro ponto: **10.0,20.0,10.0**
- Coordenadas x, y, z do segundo ponto: **10.0,10.0,20.0**
- Coordenadas x, y, z do terceiro ponto: **20.0,10.0,10.0**
- Cores R, G, B no intervalo [0,255]: **0, 255, 255**

- Exemplo de peça obrigatória com minimalistica. **rt** cena:

```
R 1920 1080
UMA 0,2
c -50,0,20 0,0,0 70 255.255.255
eu -40,0,30 0,7 255.255.255
pl 0,0,0 0,1,0,0 255.0.225
sp 0,0,20 20 255,0,0
sq 0,100,40 0,0,1,0 30 42,42,0
cy 50,0,0,0,20,6 0,0,1,0 14,2 21,42 10,0,255
tr 10,20,10 10,10,20 20,10,10 0,0,255
```

- Se alguma configuração incorreta de qualquer tipo for encontrada no arquivo, o programa deve sair corretamente e retornar "Erro \ n" seguido por uma mensagem de erro explícita de sua escolha.
- Para a defesa, o ideal seria você ter todo um conjunto de cenas com foco no que é funcional, para facilitar o controle dos elementos a criar.

# Capítulo IV

## Parte bônus

Obviamente o Ray-Tracing técnica poderia lidar com muito mais coisas, como reflexão, transparência, refração, objetos mais complexos, sombras suaves, cáusticas, iluminação global, bump mapping, renderização de arquivo .obj etc.

Mas para o miniRT projeto, queremos manter as coisas simples para seu primeiro raytracer e seus primeiros passos em CGI.

Portanto, aqui está uma lista de alguns bônus simples que você pode implementar. Se quiser fazer bônus maiores, recomendamos enfaticamente que recodifique um novo ray-tracer mais tarde em sua vida de desenvolvedor, depois que este pequeno estiver concluído e totalmente funcional.



Figura IV.1: Um ponto, uma skybox espacial e uma esfera brilhante de textura terrestre com bump-maping



Os bônus serão avaliados apenas se sua parte obrigatória for PERFEITA. Por PERFEITO queremos dizer naturalmente que precisa ser completo, que não pode falhar, mesmo em casos de erros desagradáveis como usos incorretos, etc. Basicamente, significa que se sua parte obrigatória não obtiver TODOS os pontos durante a avaliação, seus bônus irão ser totalmente IGNORADO.

Lista de bônus:

- Interrupção normal, por exemplo, usando seno que dá um efeito de onda.
- Ruptura de cor: tabuleiro de damas.
- Quebra de cor: efeito do arco-íris usando o normal do objeto.
- Luz paralela seguindo uma direção precisa.
- Elemento composto: Cubo (6 quadrados).
- Elemento composto: Pirâmide (4 triângulos, 1 quadrado).
- Colocando tampas em cilindros de tamanho limitado.
- Um outro objeto de 2º grau: Cone, Hiperbolóide, Parabolóide ..
- Filtro de uma cor: Sépia, filtros R / G / B ..
- Anti-aliasing.
- Estereoscopia simples (como óculos vermelhos / verdes).
- Renderização multithread.
- Texturização de esfera: mapeamento UV.
- Lidar com texturas de mapa de relevo.
- Um lindo camarote.
- Interatividade do teclado (translação / rotação) com câmera.
- Interatividade do teclado (translação / rotação) com objetos.
- Alterando a rotação da câmera com o mouse.



Você tem permissão para usar outras funções para completar a parte bônus, desde que seu uso seja justificado durante sua avaliação. Você também pode modificar o formato de arquivo de cena esperado para atender às suas necessidades. Seja esperto!



Para ganhar todos os pontos de bônus, você precisa validar pelo menos 14 deles, então escolha com sabedoria, mas tome cuidado para não perder seu tempo!

# **Capítulo V**

## **Exemplos**

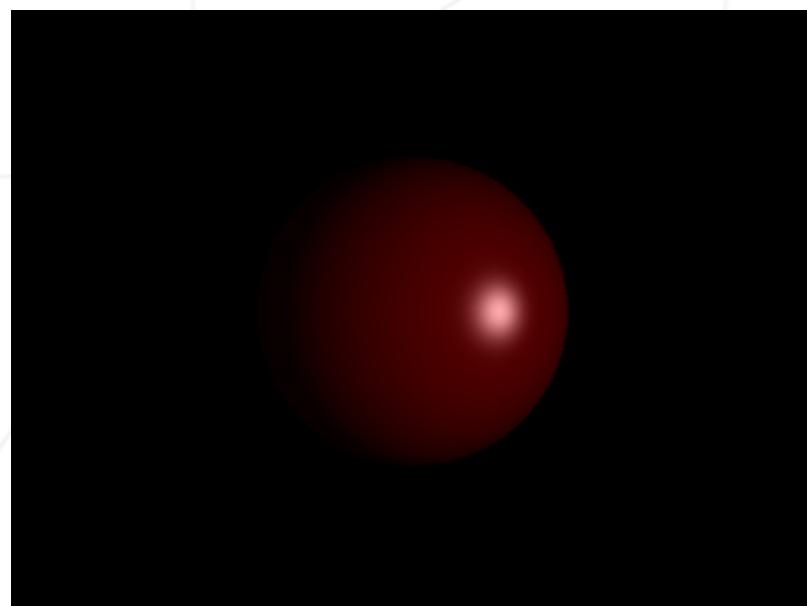


Figura V.1: Uma esfera, um ponto, algum brilho (opcional)



Figura V.2: Um cilindro, um ponto

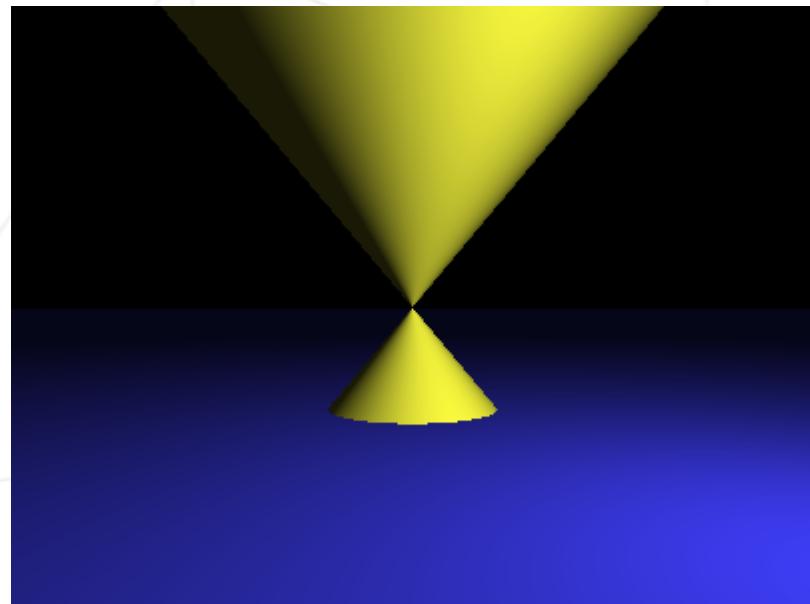


Figura V.3: Um cone (opcional), um plano, um ponto

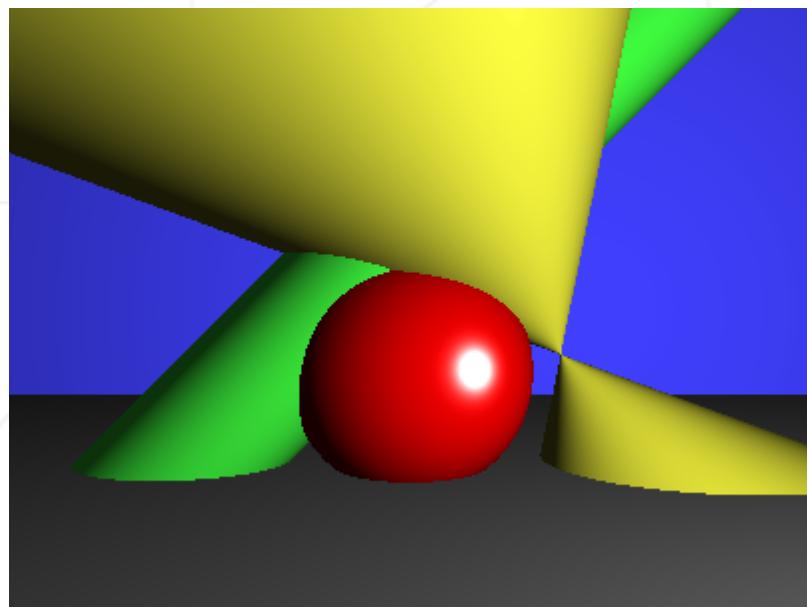


Figura V.4: Um pouco de tudo, incluindo 2 aviões

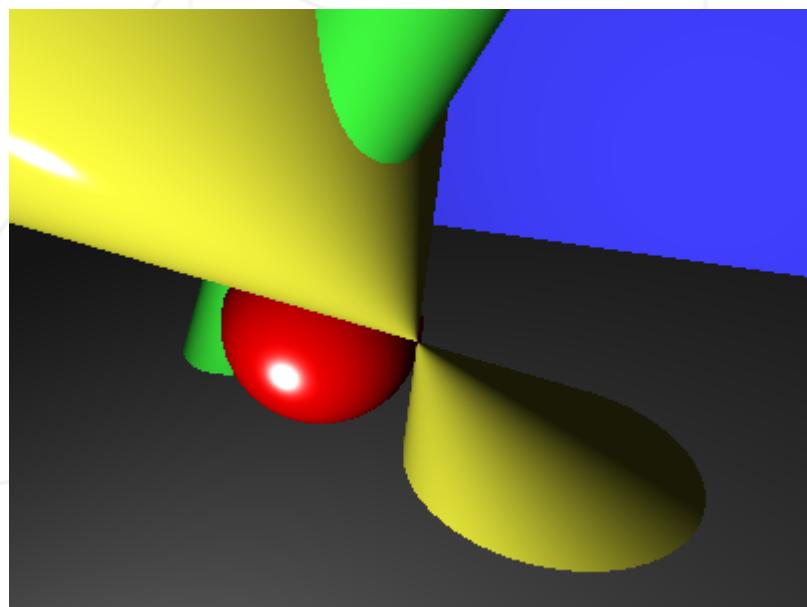


Figura V.5: mesma cena de câmera diferente

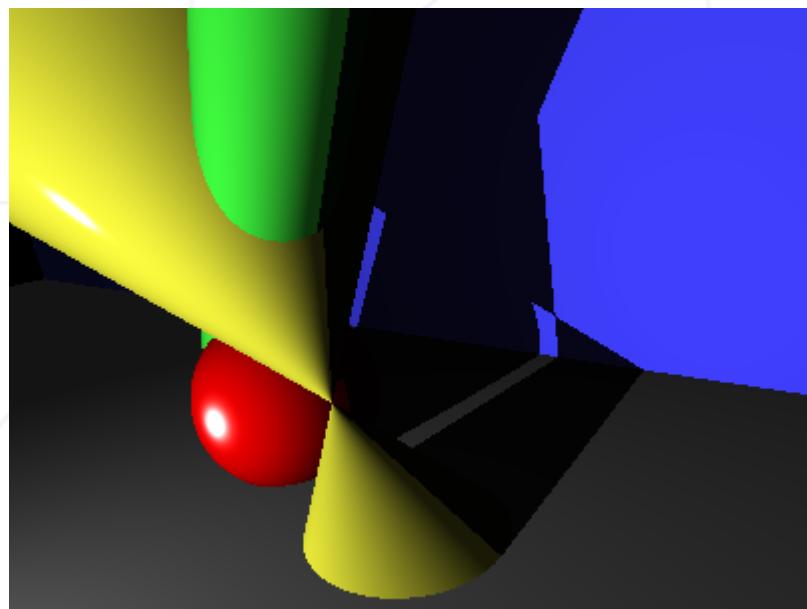


Figura V.6: Desta vez com sombras

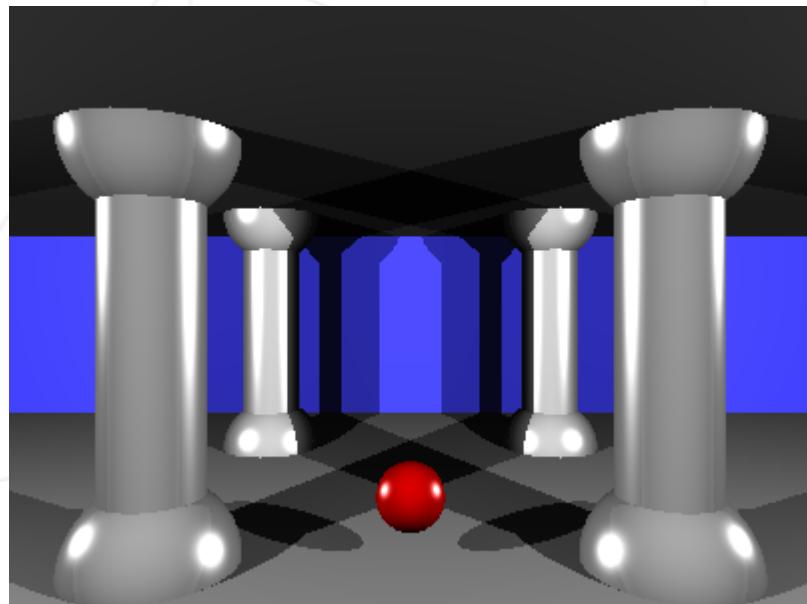


Figura V.7: Com vários pontos

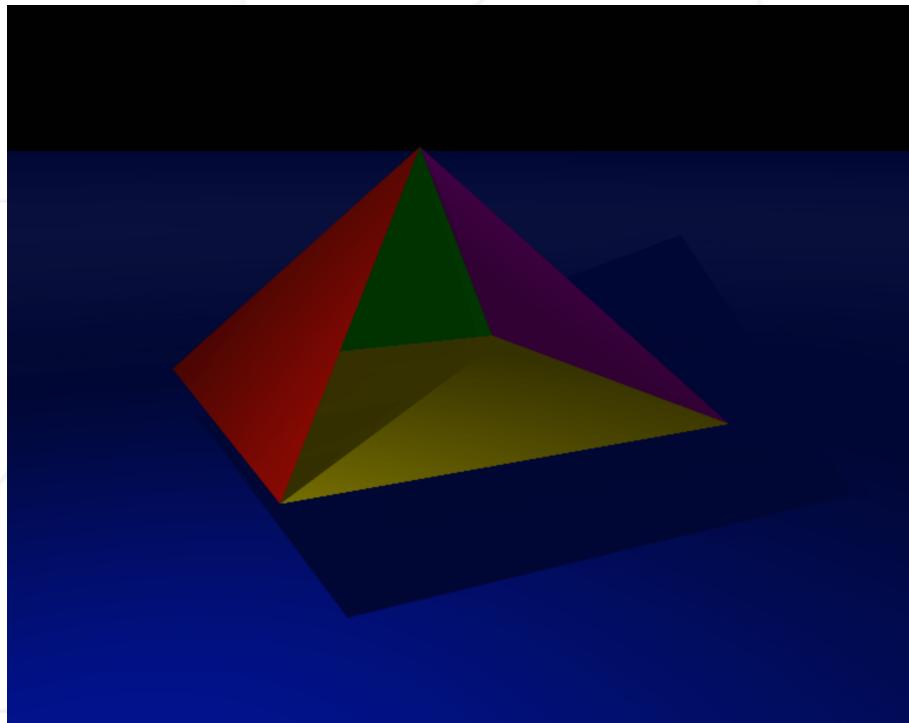


Figura V.8: Um plano, 3 triângulos, 1 quadrado e um ponto de baixo brilho à esquerda

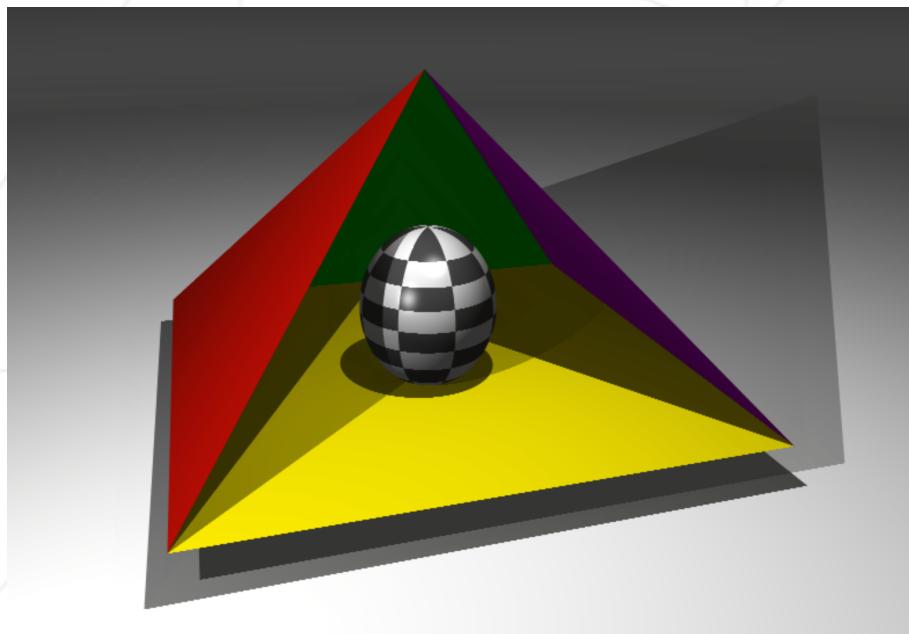


Figura V.9: E finalmente com vários pontos e uma esfera quadriculada brilhante (opcional) no meio