



filhote3D

Meu primeiro RayCaster com miniLibX

Resumo: Este projeto é inspirado no mundialmente famoso jogo Wolfenstein 3D, que foi o primeiro FPS de todos os tempos. Isso permitirá que você explore a projeção de raios. Seu objetivo será fazer uma visão dinâmica dentro de um labirinto, no qual você terá que encontrar o seu caminho.



Versão: 10

Conteúdo

I	Prefácio	2
II	Metas	4
III	Instruções Comuns	5
4	Parte obrigatória - cub3D	7
V	Parte bônus	11
VI	Exemplos	12
VII	Envio e avaliação por pares	15



Capítulo I

Prefácio

Desenvolvido porSoftware de identificaçãodos super famosos John Carmack e John Romero, publicado em 1992 pelaSoftware Apogee, Wolfenstein 3Dé o primeiro verdadeiro “tiro em primeira pessoa” da história dos videogames.



Figura I.1:John Romero (esquerda) e John Carmack (direita) posando para a posteridade.

Wolfenstein 3Dé o ancestral de jogos comoDoom (software de identificação,1993),Doom II (software de identificação,1994),Duke Nukem 3D (reino 3D,1996) eQuake (Software de identificação, 1996), que são marcos eternos adicionais no mundo dos videogames.

Agora é a sua vez de reviver a História...



O jogo Wolfenstein 3D se passa originalmente na Alemanha nazista, o que pode ser eventualmente perturbador. As fotos e a história deste jogo são trazidas a você apenas por motivos técnicos e de cultura pop/geek, já que o jogo foi considerado uma obra-prima para ambos.

Capítulo II

Metas

Os objectivos deste projecto são semelhantes a todos os objectivos deste primeiro ano: Rigor, utilização de C, uso de algoritmos básicos, pesquisa de informações etc.

Como um projeto de design gráfico, filhote3D permitirá que você melhore suas habilidades nestas áreas: janelas, cores, eventos, formas de preenchimento, etc.

Concluir filhote3D é um playground notável para explorar as aplicações práticas lúdicas da matemática sem ter que entender os detalhes.

Com a ajuda dos inúmeros documentos disponíveis na internet, você utilizará a matemática como ferramenta para criar algoritmos elegantes e eficientes.



Se for conveniente para você, você pode testar o jogo original antes de iniciar este projeto:

<http://users.atw.hu/wolf3d/>

Capítulo III

Instruções Comuns

- Seu projeto deve ser escrito em C.
- Seu projeto deve ser escrito de acordo com a Norma. Se você tiver arquivos/funções bônus, eles serão incluídos na verificação de norma e você receberá um0se houver um erro de norma dentro.
- Suas funções não devem encerrar inesperadamente (falha de segmentação, erro de barramento, liberação dupla, etc.), exceto por comportamentos indefinidos. Se isso acontecer, seu projeto será considerado não funcional e receberá uma0durante a avaliação.
- Todo o espaço de memória alocado no heap deve ser liberado adequadamente quando necessário. Nenhum vazamento será tolerado.
- Se a disciplina exigir, você deverá enviar umMakefileque compilará seus arquivos de origem para a saída necessária com os sinalizadores -Parede, -Wextrae -erro,use cc, e seu Makefile não deve ser vinculado novamente.
- SeuMakefiledeve conter pelo menos as regras \$(NOME), tudo, limpo, fclean e ré.
- Para entregar bônus ao seu projeto, você deve incluir uma regra bônus no seu Makefile, 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}se o assunto não especificar mais nada. A avaliação da parte obrigatória e da parte bônus é feita separadamente.
- Se o seu projeto permitir que você use seulibft,você deve copiar suas fontes e seus associadosMakefileem umlibftpasta com seu Makefile associado. O seu projeto Makefile deve compilar a biblioteca usando seuMakefile,em seguida, compile o projeto.
- Nós encorajamos você a criar programas de teste para o seu projeto, mesmo que este trabalho **não precisará 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 é livre para usar seus testes e/ou os testes do colega que está avaliando.
- Envie seu trabalho para o repositório git atribuído. Somente o trabalho no repositório git será avaliado. Se Deepthought for designado para avaliar seu trabalho, isso será feito

após suas avaliações por pares. Se ocorrer um erro em qualquer seção do seu trabalho durante a avaliação do Deepthought, a avaliação será interrompida.

Capítulo IV

Parte obrigatória - cub3D

Nome do programa	filhote3D
Entregar arquivos	Todos os seus arquivos
Makefile	tudo, limpo, fclean, re, bônus
Argumentos	um mapa no formato *.cub
Funções externas.	<ul style="list-style-type: none">• abrir, fechar, ler, escrever, printf, malloc, grátis, perror, strerror, sair, gettimeofday• Todas as funções da biblioteca matemática (-lm man man 3 math)• Todas as funções do MinilibX
Libft autorizado	Sim
Descrição	Você deve criar uma representação gráfica 3D “realista” do interior de um labirinto a partir de uma perspectiva de primeira pessoa. Você deve criar esta representação usando os princípios de Ray-Casting mencionados anteriormente.

As restrições são as seguintes:

- Você deve usar o miniLibX. A versão 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 Instruções Comunspapel.
- A gestão da sua janela deve permanecer suave: mudar para outra janela, minimizar, etc.
- Exiba diferentes texturas de parede (a escolha é sua) que variam dependendo do lado da parede (Norte, Sul, Leste, Oeste).

- Seu programa deve ser capaz de definir as cores do piso e do teto para duas cores diferentes.
 - O programa exibe a imagem em uma janela e respeita as seguintes regras:
 - As teclas de seta esquerda e direita do teclado devem permitir que você olhe para a esquerda e para a direita no labirinto.
 - As teclas W, A, S e D devem permitir que você move o ponto de vista pelo labirinto.
 - 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 encerrar o programa de forma limpa.
 - O uso de imagens domínio libX é fortemente recomendado.
 - Seu programa deve tomar como primeiro argumento um arquivo de descrição de cena com a extensão .filhote extensão.
 - O mapa deve ser composto por apenas 6 caracteres possíveis: 0 para um espaço vazio, 1 para uma parede e N, S, E ou C para a posição inicial do jogador e orientação de desova.
- Este é um mapa simples e válido:
- ```
111111
100101
101001
1100N1
111111
```
- O mapa deve estar fechado/cercado por paredes, caso contrário o programa deverá retornar um erro.
  - Exceto pelo conteúdo do mapa, cada tipo de elemento pode ser separado por uma ou mais linhas vazias.
  - Exceto o conteúdo do mapa que sempre deve ser o último, cada tipo de elemento pode ser definido em qualquer ordem no arquivo.
  - Com exceção do mapa, cada tipo de informação de um elemento pode ser separada por um ou mais espaços.
  - O mapa deve ser analisado conforme aparece no arquivo. Os espaços são uma parte válida do mapa e cabe a você administrá-los. Você deve ser capaz de analisar qualquer tipo de mapa, desde que respeite as regras do mapa.

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

- \* Textura Norte:

```
NÃO ./path_to_the_north_texture
```

- identificador:**NÃO**
- caminho para a textura norte

- \* Textura Sul:

```
SO ./path_to_the_south_texture
```

- identificador:**ENTÃO**
- caminho para a textura sul

- \* Textura Oeste:

```
NÓS ./path_to_the_west_texture
```

- identificador:**NÓS**
- caminho para a textura oeste

- \* Textura Leste:

```
EA ./path_to_the_east_texture
```

- identificador:**EA**
- caminho para a textura leste

- \* Cor do piso:

```
F 220.100,0
```

- identificador:**F**
- Cores R,G,B na faixa [0,255]:**0, 255, 255**

\* Cor do teto:

C 225,30,0

- identificador:**C**
- Cores R,G,B na faixa [0,255]:**0, 255, 255**

- Exemplo da parte obrigatória com um minimalista.**filhotecena**:

```
NÃO ./path_to_the_north_texture
SO ./path_to_the_south_texture
NÓS ./path_to_the_west_texture
EA ./path_to_the_east_texture

F 220.100,0
C 225,30,0

11111111111111111111111111111111
100000000011000000000001
101100000111000000000001
100100000000000000000001
11111111011000011100000000000001
100000000110000111011111111111
11110111111101110000010001
1111011111110111010100 10001
11000000110101011100000010001
10000000000000001100000010001
10000000000000001101010010001
110000011101010111110111110111
11110111 1110101 101111010001
11111111 11111111 111111111111
```

- Se qualquer tipo de configuração incorreta for encontrada no arquivo, o programa deverá sair corretamente e retornar "Erro\n" seguido por uma mensagem de erro explícita de sua escolha.

# Capítulo V

## Parte bônus



Os bônus serão avaliados somente 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 errados etc. Significa que se sua parte obrigatória não obtiver TODOS os pontos durante a avaliação, seus bônus serão inteiramente IGNORADO.

Lista de bônus:

- Colisões de paredes.
- Um sistema de minimapa.
- Portas que podem abrir e fechar.
- sprite animado.
- Gire o ponto de vista com o mouse.



Você poderá criar jogos melhores mais tarde, não perca muito tempo!



Você pode usar outras funções ou adicionar símbolos no mapa 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!

# Capítulo VI

## Exemplos



Figura VI.1:Wolfenstein3Dréplica do design original, usando RayCasting.

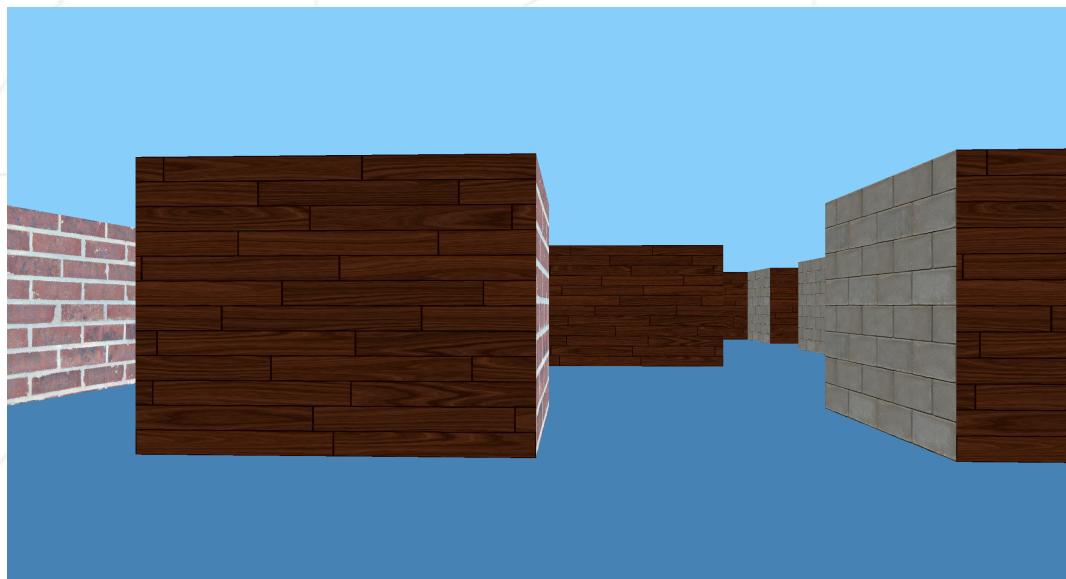


Figura VI.2:Exemplo de como seria o seu projeto conforme a parte obrigatória.



Figura VI.3: Exemplo da parte bônus com minimapa, texturas de piso e teto e um famoso sprite animado de ouriço.



Figura VI.4: Outro exemplo de bônus com HUD, barra de saúde, efeito de sombra e arma que pode atirar

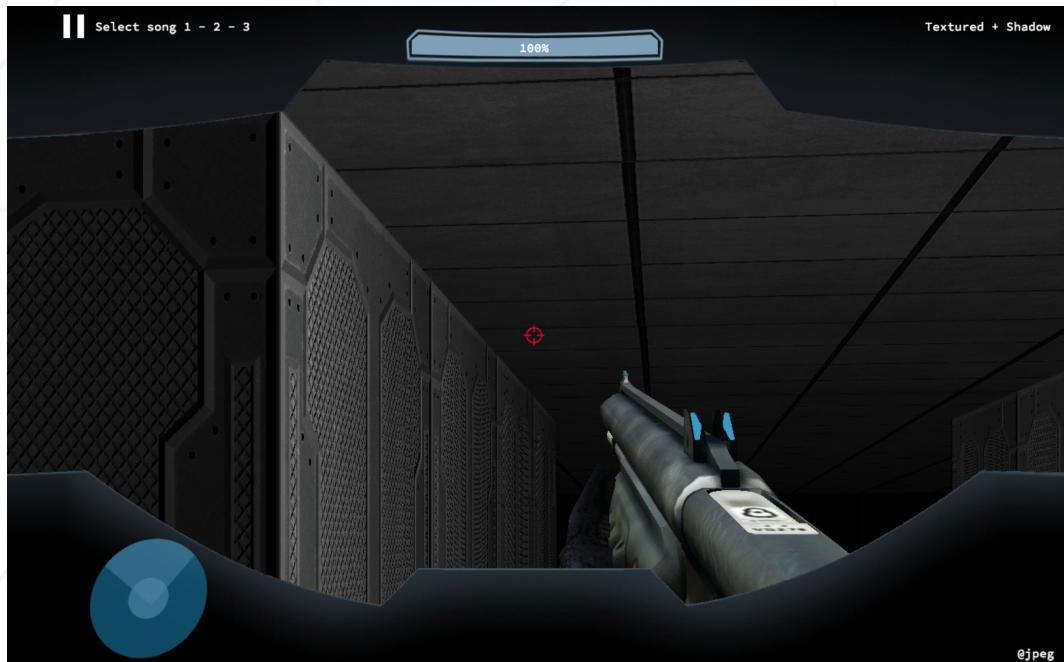


Figura VI.5:Outro exemplo de jogo bônus com uma arma de sua escolha e o jogador olhando para o teto

Nota: as imagens acima podem conter imagens protegidas por direitos autorais, assim como o documento inteiro. Consideramos que este assunto, criado para fins educacionais, se enquadra nas diretrizes de Fair Use.

## **Capítulo VII**

# **Envio e avaliação por pares**

Entregue sua tarefa em seu Gitrepository como de costume. Somente o trabalho dentro do seu repositório será avaliado durante a defesa. Não hesite em verificar os nomes dos seus arquivos para garantir que estão corretos.



???????????? XXXXXXXXXX = \$3\$\$796ba5a53df1352e06cc7b0f3ad2a41d