

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO
INF01047 – FUNDAMENTOS DE COMPUTAÇÃO GRÁFICA
TURMA 2016/1

DIEGO DASSO MIGOTTO

TRABALHO FINAL DE FCG
JOGO DESENVOLVIDO BASEADO EM DIG DUG II

Relatório de Desenvolvimento
Prof. Marcelo Walter

Porto Alegre
2016

SUMÁRIO

1 O JOGO.....	3
2 CRONOLOGIA DO DESENVOLVIMENTO.....	4
3 MANUAL DO JOGO.....	5
4 IMPLEMENTAÇÃO DOS REQUISITOS.....	11
5 CONSIDERAÇÕES FINAIS.....	12

1 O JOGO

A proposta do trabalho final da turma do primeiro semestre de 2016 da disciplina de Fundamentos de Computação Gráfica do Instituto de Informática da UFRGS era o desenvolvimento de um jogo baseado no título Dig Dug II(NES), individualmente ou em dupla. Havia também a possibilidade de desenvolvimento livre atentando ao enquadramento da aplicação nos conceitos de computação gráfica requisitados na especificação do trabalho.

Esse relatório busca expor o desenvolvimento do adaptação proposta (Dig Dug II) utilizando a API OpenGL e linguagem C. A baixo, um breve resumo das especificações do jogo previstas na proposta do professor.

- Carregar níveis através de dois arquivos bitmap 20x20 separados(nível inferior e superior), com cada pixel representando um objeto no mundo tridimensional, composto de: posição inicial do jogador, posição inicial de até quatro inimigos, blocos representando o chão, obstáculos, rachaduras e buracos;
- Jogador pode se movimentar, criar rachaduras e empurrar inimigos;
- Inimigos com inteligência artificial simples;
- Buracos sobre o qual o jogador deve estar para criar rachaduras;
- Obstáculos que não podem ser ultrapassados pelo jogador/inimigos;
- Desmoronamento de partes do cenário cercadas por água e/ou rachaduras;

O jogo desenvolvido recebeu o título de “Dig Dug II: Apples and Oranges”, inspirado no modelo de esfera laranja utilizado em práticas anteriores da disciplina.

2 CRONOLOGIA DO DESENVOLVIMENTO

A especificação do trabalho foi liberada na semana final de abril, com entrega prevista para o dia 26/06, entretanto o desenvolvimento foi iniciado somente no dia 22/06.

No dia 22/06 foi criado um novo projeto na IDE codeblocks espelhado no projeto da prática 3 da disciplina, que se resumia a uma câmera com movimentação simples, em um plano, com uma esfera no centro. Neste dia foi feita a familiarização com o código já existente e a remoção de funções que não seriam utilizadas, como funções de input do mouse. No final do dia restou um programa com a implementação de uma câmera em primeira pessoa e sua movimentação e iluminação.

No dia 23/06 foi desenvolvida uma função para carregar os bitmaps do mapa e posicionar e renderizar o jogador em sua posição inicial, plataformas representando os blocos nos quais os atores poderiam se movimentar e a água no restante do cenário. Além dessas funcionalidades, foi tentado o desenvolvimento de uma “skybox” cercando o cenário, entretanto não foi atingido o resultado esperado e essa feature foi “empurrada” para mais adiante.

No dia 24/06, as funções desenvolvidas anteriormente foram aperfeiçoadas e foram implementados os obstáculos, representados por caixas carregadas de um modelo .obj disponibilizado pelo professor. Foram feitas as rachaduras, buracos e o desmoronamento do terreno ao serem formado dois ciclos de rachaduras/água/buraco e colisão do jogador com obstáculos. Os modos de câmera em terceira pessoa e topdown também foram desenvolvidos.

No dia 25/06, foi desenvolvido o restante das funcionalidades: inimigos com inteligência artificial parcial, um overlay de texto com tempo da partida e inimigos restantes. minimapa, menu principal, lógica de vitória e derrota, uma skybox e suporte para vários níveis. Além disso, para propósito de aprendizagem, foi desenvolvido um modelo para o braço dos personagens no software de modelagem Blender, que precisou ser simplificado devido a problemas de performance.

Por fim, o dia 26/06 foi reservado para a escrita do relatório.

3 MANUAL DO JOGO

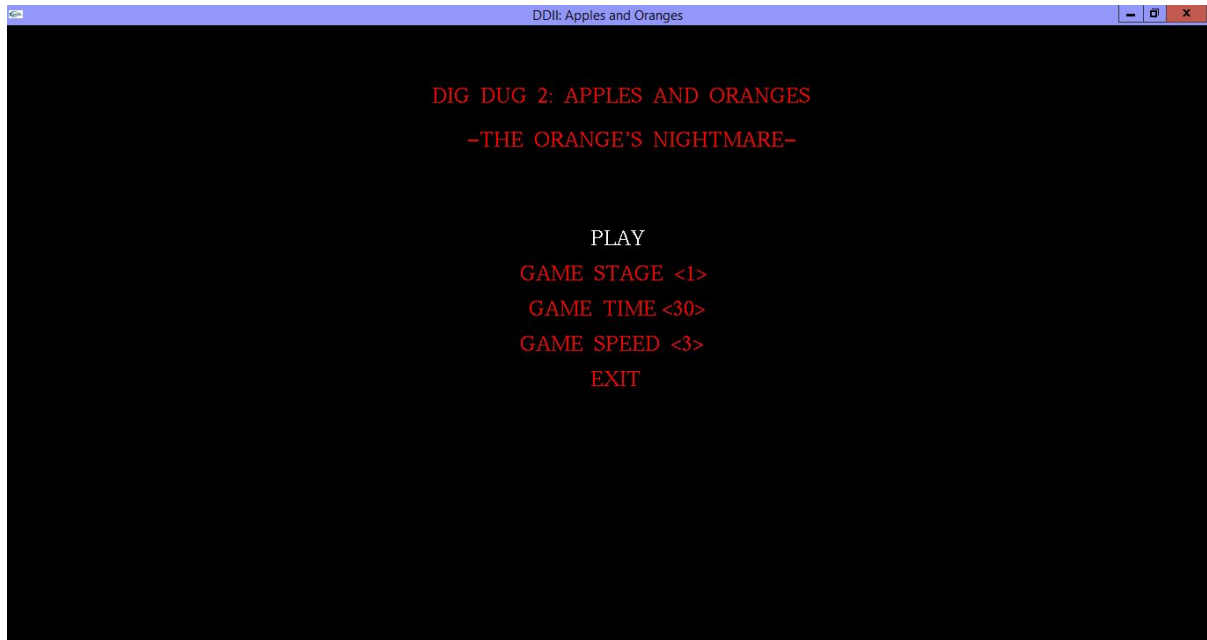
DIG DUG II: Apples and Oranges - The orange's Nightmare

A temática do jogo de laranjas contra maçãs foi inspirada no modelo de esfera utilizado na prática 3 com uma textura laranja. Foi levado em consideração que seria um modelo fácil de trabalhar, que não afetaria muito a performance da aplicação e que, se atrelado a uma temática condizente deixaria o jogo com uma aparência razoável.

O jogador controla o personagem principal, uma laranja. Cada fase é um pesadelo da laranja no qual ela deve afundar todas as maçãs sem cabeça para poder escapar. Para afundar as maçãs, a laranja deve pisar nos buracos dispostos pelas fases e criar rachaduras, que quando separarem duas partes do terreno, farão com que a menor parte desmorone, levando junto consigo para água quaisquer objetos ou inimigos que estiverem sob ela. O jogo vem com 4 fases prontas e 5 espaços para fases de autoria do jogador.

Controles

- w, a, s, d e as setas do teclado - controlam a movimentação do personagem e o menu;
- Enter - para selecionar as opções PLAY e EXIT do menu, cima e baixo para trocar entre as opções, e esquerda e direita para alterar os parâmetros;
- r - para correr dentro do jogo;
- v - troca entre os modos de câmera;
- z, x - diminui/aumenta distância da câmera para o personagem;
- Barra de espaço - arma de rachadura;
- Esc - sair do jogo;



Menu Principal

- Play - Inicia o jogo com as configurações escolhidas;
- Game Stage - Seleciona a fase desejada: 1 a 4 vem com o jogo, outros 5 a 9 reservadas para o usuário desenvolver próprias fases;
- Game Time - Tempo disponível para terminar a fase: 30, 45 ou 60 segundos;
- Game Speed - Velocidade do jogo: 1 2 ou 3, o usuário deve testar as velocidades e escolher a mais adequada ao seu computador;
- Exit - Sai do jogo.

Para criar o próprio nível, o jogador deve editar os arquivos lowerGridX.bmp e upperGridX.bmp. A legenda para a lowerGrid é:

Preto(0,0,0) BLOCO;

Branco(255,255,255) ÁGUA.

A legenda para a upperGrid é:

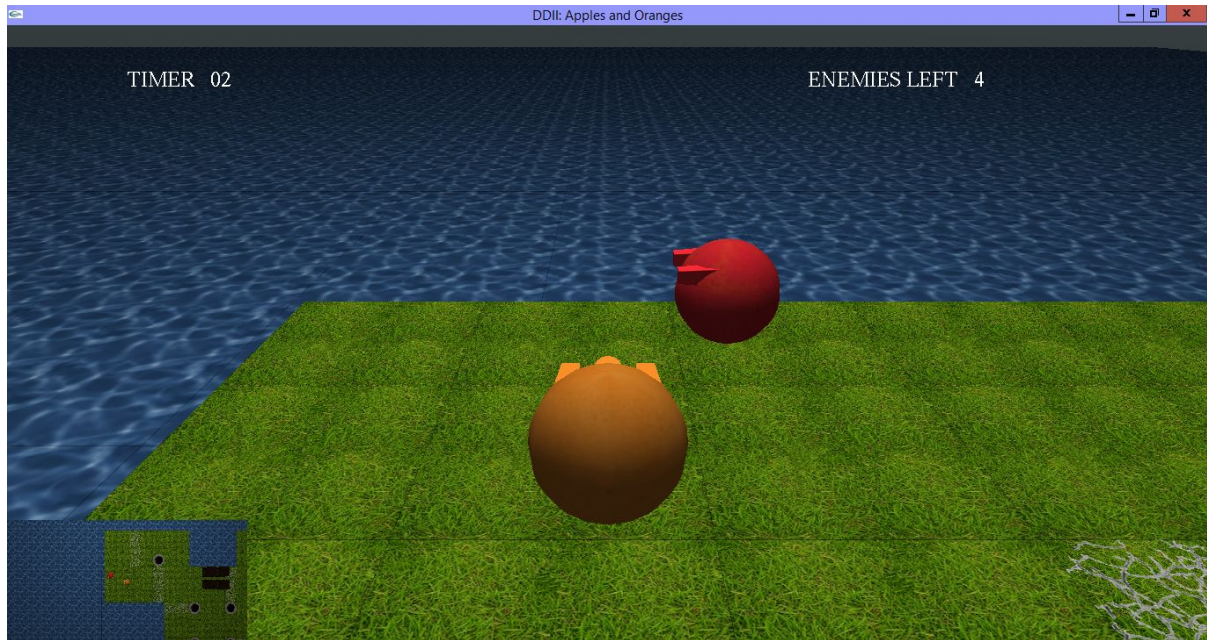
Azul(63,72,204) POSIÇÃO JOGADOR;

Vermelho(237,28,36) POSIÇÃO INIMIGO;

Marrom(185,122,87) RACHADURA;

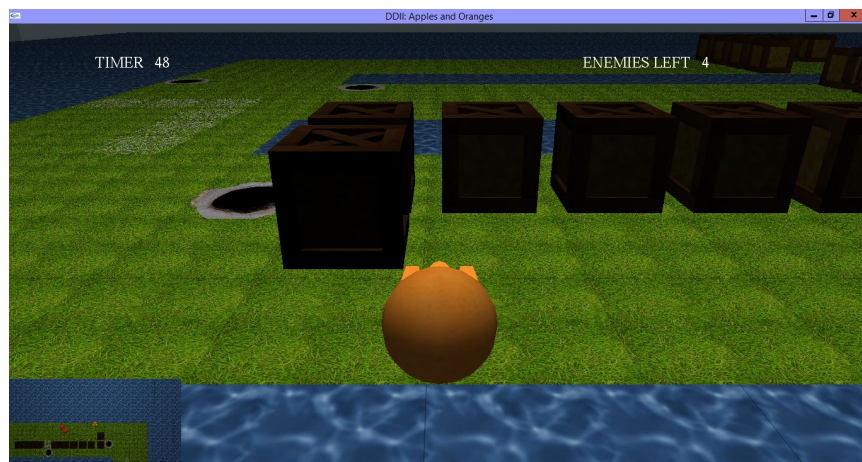
Marrom Escuro(136,0,21) BURACO;

Verde(181,230,29) OBSTÁCULO;

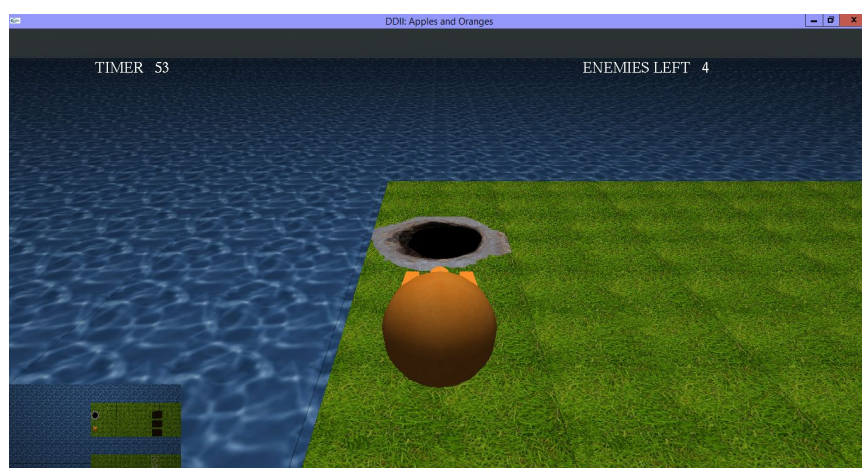


Tela do jogo em andamento

- No canto superior esquerdo está o TIMER, indicando o tempo restante para terminar a fase, na imagem indica que se o nível não for completado em 2 segundos será Game Over, congelado caso o usuário consiga eliminar todos os inimigos;
- No canto superior direito está ENEMIES LEFT, indicando quantos inimigos ainda estão vivos, quando nenhum inimigo estiver vivo, o jogador vence a fase;
- No centro da tela, em laranja, está o modelo do jogador, uma laranja, com cabeça e braços primitivos;
- Próximo ao modelo da laranja, em vermelho, está o modelo da maçã, sem cabeça e com braços primitivos. O inimigo pode apenas caminhar em direção a plataformas verdes, não podendo caminhar sobre buracos, rachaduras, ultrapassar obstáculos ou outros inimigos. Caso o jogador encoste no inimigo, ele perde, para derrotar um inimigo, ele o deve derrubar na água utilizando a arma de rachaduras. Há no máximo 4 inimigos por fase;
- No canto inferior esquerdo está o minimapa, o entorno da fase visto de cima;



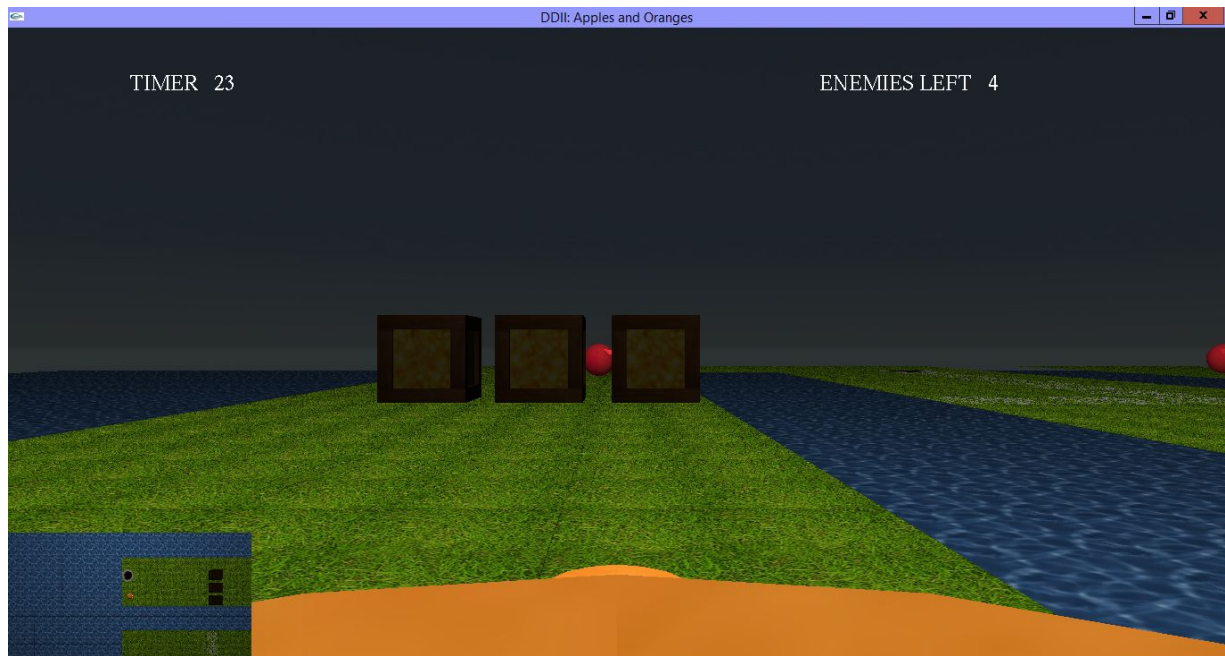
Na imagem ao lado é possível ver várias caixas, esses são os obstáculos - caixas de suco de laranja. Nem o jogador nem os inimigos podem passar pelos obstáculos.



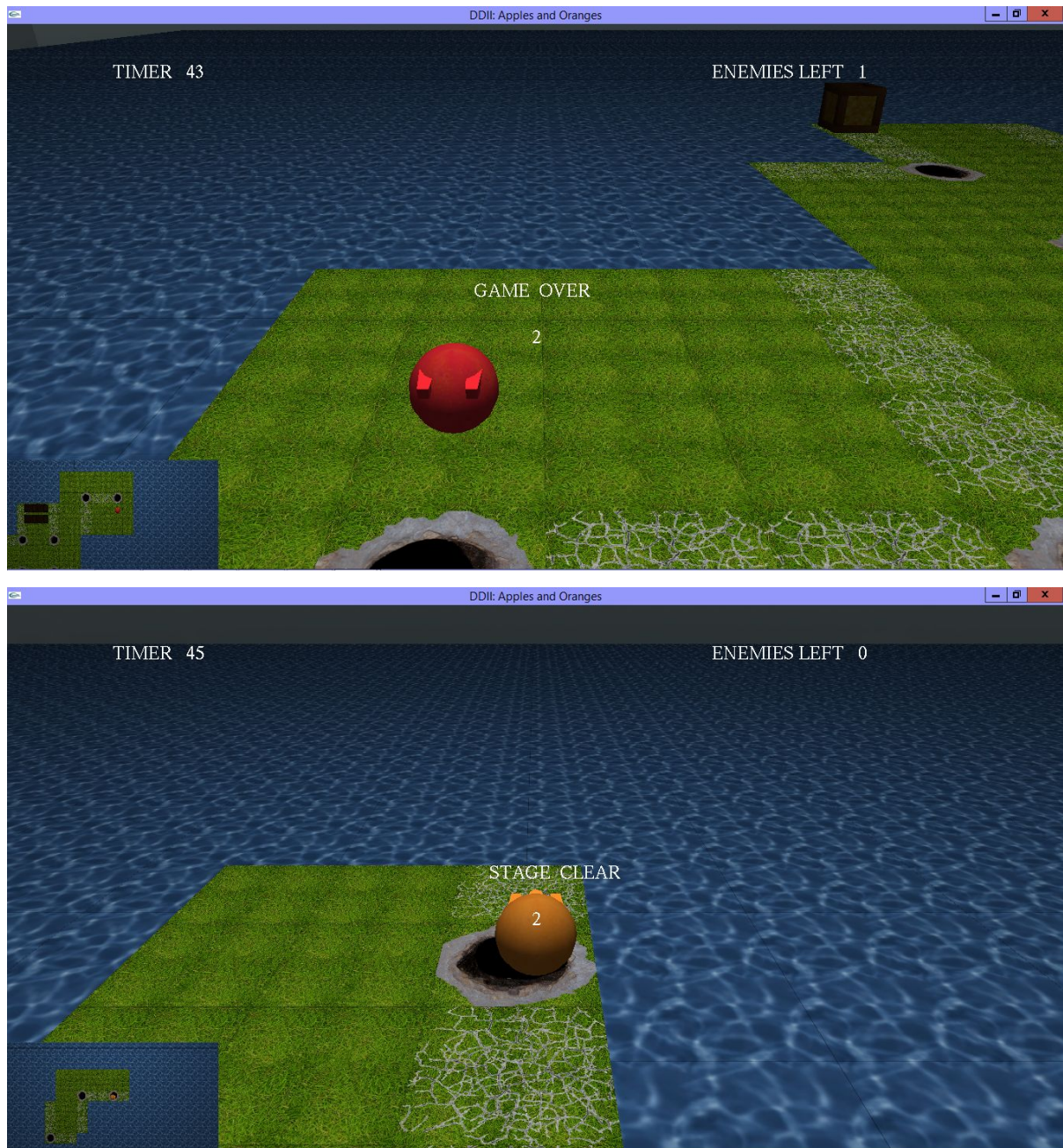
Na imagem seguinte está o buraco, para o jogador gerar uma rachadura deve estar posicionado em cima dele.



Como mostra a terceira imagem, se ao gerar uma rachadura o jogador separar o mapa em dois, fará com que o menor terreno dentre as duas partes desmorone.



Acima estão duas imagens com os modos de câmeras ainda não mostrados, em primeira pessoa e visto de cima.



Acima estão as telas de GAME OVER e STAGE CLEAR. Quando o personagem cai na água, encosta em um inimigo ou acaba o tempo, o jogador perde. Será travado o teclado, a tela, e o jogador receberá a mensagem de GAME OVER juntamente a um contador de 3 segundos. Ao fim da contagem será levado de volta ao Menu. Caso o jogador elimine todos os inimigos, ele receberá a mensagem de STAGE CLEAR e será levado ao próximo estágio.

4 IMPLEMENTAÇÃO DOS REQUISITOS

A seguir será detalhado de forma resumida algumas das formas utilizadas para implementar os requisitos nos quais foi encontrado alguma dificuldade. Dentre os requisitos básicos do projeto, um não foi implementado, que é a tecla F para empurrar inimigos, e a inteligência artificial foi implementada parcialmente - os inimigos se movem aleatoriamente, e ignoram o jogador. Dentre requisitos extras foram implementados um tempo para a fase, um algoritmo que elimina rachaduras e buracos cercados por água e interfaces gráficas. Algo que não foi especificado no trabalho nem nos requisitos básicos, mas foi implementado, foi uma “skybox” rudimentar.

- Implementação do minimapa, feita através do uso das funções `glScissor` e `glViewport` para renderizar primeiro a cena principal e depois o minimapa;
- Implementação de uma skybox rudimentar, através de 4 quads posicionados nos extremos do mapa texturizados. Necessário aumentar o alcance da visão na função `gluPerspective`. Não foi alcançado exatamente o efeito desejado devido a iluminação ter influenciado de forma diferente nos 4 quads;
- Implementação do menu através do uso da função `glutBitmapCharacter`, seria interessante exemplificar o uso dessa função em alguma prática dos próximos semestres já que é extremamente útil no desenvolvimento do trabalho final;
- Iluminação através de luz ambiente extremamente fraca, e outro foco de luz difusa/especular;
- Utilização da função `glFog` abandonada devido a implementação da skybox;
- Chão, água, buracos e rachaduras implementados como quads texturizados. Obstáculos, personagem principal, e inimigos implementados como modelos 3D .obj importados através da função fornecida na disciplina;
- Colisão do personagem principal com obstáculos e com fim do mapa para ser derrubado na água feita através da utilização de um mapa dos objetos no jogo. O inimigo tem uma espécie de colisão falsa, ele nunca ira se mover em direção a um obstáculo buraco, rachadura ou se jogar na água.

5 CONSIDERAÇÕES FINAIS

Olhando para projetos realizados anteriormente no curso, pode ser que esse seja um dos mais extensos já desenvolvidos. Apesar da quantidade de trabalho, após pegar o ritmo e entender a maioria das funções da API OpenGL, o desenvolvimento começa a fluir rapidamente. E da mesma maneira que é extenso, é extremamente gratificante ver o jogo funcionar antes de entregar o trabalho.

Apesar de no início, a utilização de uma API tão primitiva, da qual não se espera resultados visualmente impressionantes, ser desencorajadora, no final se percebe que é a escolha correta. A utilização dessa versão antiga do OpenGL, permite ao aluno realmente executar os conceitos aprendidos na disciplina, sem ser necessário se estender além do que foi aprendido. Não acredito que algum dia utilizaria essa versão do OpenGL de novo, mas com certeza a realização do trabalho foi extremamente útil para entender como os conceitos funcionam na prática.

Em geral, o resultado foi bastante satisfatório, até melhor do que o esperado no início do desenvolvimento. O único aspecto negativo foi a dificuldade de encontrar modelos no formato .obj com texturas na internet. Seria interessante disponibilizar um conversor de outros formatos para .obj ou uma função que carregasse modelos de outros formatos.