**Trabalho prático I**

**Computação Gráfica**

**Lincoln Mateus Konig - 2014129589**

**1. Introdução**Breakout foi um jogo de bastante sucesso desenvolvido por Steve Wozniak onde o objetivo era destruir todos os "tijolos" controlando-se uma barra e uma bola em movimento. Neste trabalho, foi feita uma versão moderna desse jogo a fim de se aprofundar os conhecimentos a respeito de OpenGL.

**2. Compilação e execução**O jogo foi desenvolvido em linguagem C#, devido à familiaridade com a mesma e a facilidade de portabilidade entre sistemas operacionais. A biblioteca utilizada para expor as funções do OpenGL é a OpenTK. O projeto necessita pelo menos OpenGL 3.3.

Para a compilação do projeto, será disponibilizado um makefile, mas a instalação do SDK do .NET Core 3.1 é necessária. Também serão disponibilizado dois arquivos prontos para execução: um executável windows e um programa compatível com linux (ambos somente x64). Nenhum argumento é necessário para a execução, o jogo se iniciará automaticamente na resolução de 612x648.

**3. Decisões de implementação**Para o desenvolvimento desse jogo, diversas decisões foram tomadas a fim de se maximizar o aprendizado a respeito de OpenGL.

**Shaders**: Shaders foram utilizados a fim de possibilitar uma maior flexibilidade na renderização de objetos na tela. Os dois principais motivos pelo qual se optou por usar shaders é aprender como funcionam códigos da GPU e a exibição de texturas.

**Sons:** Para se possibilitar uma experiência mais agradável, foram implementados sons utilizando a biblioteca NAudio.

**Powerups:** Para tornar a experiência mais divertida e desafiadora, powerups foram colocados no jogo. Alguns possuem um efeito positivo na experiência do jogador, enquanto outros a tornam mais difícil. Uma interface foi implementada a fim de se exibir quais powerups estão ativos no momento.

**Dificuldade:** 3 dificuldades foram implementadas. A dificuldade escolhida modifica diversos atributos do jogo, como a velocidade da bolinha e do paddle, frequência de powerups e quantidade de vidas.

**PostProcessing:** Efeitos de pós-processamento da imagem foram implementados a fim de se aprender mais sobre a utilização de framebuffers e shaders

**Partículas:** Efeito de partículas foram implementadas a fim de se entender a diferença entre objetos do jogo e partículas, bem como a diferença de performance de processamento entre um e outro.

**Níveis:** Diferentes níveis foram implementados, com graus de dificuldade variados.

**Texto:** A renderização de texto foi implementada a fim de se entender mais sobre como funciona a renderização de fontes com OpenGL. A biblioteca utilizada para leitura de fontes foi a SharpFont, que se baseia na biblioteca freetype6.dll

**Menu:** Ume menu onde se é poossível escolher diferentes fases foi implementado. Nesse menu também e possível acessar uma tela informativa onde existem instruções básicas sobre como jogar.

**Início do jogo:** Ao invés do jogo começar pausado, devido a implementação de um menu, optou-se por iniciar o jogo no menu para a escolha de fases. Quando uma fase é escolhida, o jogo se inicia com a bolinha travada na barra, e a barra livre para se mover. Ao se pressionar espaço, a bolinha é lançada em alguma direção aleatória para cima.

**Blocos indestrutíveis:** Blocos indestrutíveis foram implementados para diversificar as fases. Eles não rpecisam

ser destruídos para se ganhar.

**Reflexão da bola:** A bola reflete na barra com uma variação na velocidade do eixo horizontal dela, dependendo do quão distante do centro da barra ela bate.

**Vitória e derrota:** A vitória e a derrota possuem suas respectivas telas, e são atingidas ao se destruir todos os blocos quebráveis, ou ao se perder todas as vidas, respectivamente.

**Detalhes técnicos:** Conforme exigido na documentação, "Q" sai do jogo a qualquer momento e "R" reinicia o jogo (retornando ao menu). O botão esquerdo do mouse pausa e despausa o jogo, e o botão direito pausa e exibe informações a respeito de todos os objetos na tela. Se o jogo já estiver pausado, o botão direito executa um loop e exibe as informações na tela. Essas informações podem ultrapassar o tamanho da janela devido a quantidade de "tijolos" na tela, mas são exibidas também no console, caso disponível.

**4.Considerações finais**O desenvolvimento do trabalho foi de grande valia para o aprendizado de OpenGL e as diversas funcionalidades da linguagem. O projeto foi baseado nas informações disponíveis no site learnopengl.com, o que possibilitou um maior aprofundamento na utilização de shaders e framebuffers, e a implementação de de efeitos como pós-processamento de imagens e partículas. O código fonte encontra-se disponível em <https://github.com/bootzin/tp1>