Trabalho Prático - R-type

Pedro O.S. Vaz de Melo

December 16, 2021

1 Descrição do Problema

O objetivo deste trabalho é fazer com que o aluno utilize as técnicas de programação aprendidas na disciplina para desenvolver um jogo eletrônico gráfico semelhante às versões clássicas (antigas) da franquia *R-type*. R-Type [a] é um videogame de fliperama de rolagem horizontal desenvolvido e lançado pela Irem em 1987. O jogador controla uma nave estelar, a R-9 "Arrowhead", em seus esforços para destruir o Bydo, uma poderosa raça alienígena curvada em exterminar toda a humanidade. A versão arcade foi distribuída pela Nintendo na América do Norte; é o último título de arcade distribuído pela Nintendo. É o primeiro jogo da série R-Type. O jogo é composto por vários níveis sequenciais, com um chefe inimigo no final de cada um. O jogador controla uma pequena nave espacial e deve navegar pelo terreno e lutar contra os inimigos usando as várias armas da nave. A nave espacial do jogador tem, por padrão, uma arma principal fraca, mas de disparo rápido; e uma arma mais poderosa chamada canhão de ondas, que exige que o jogador segure o fogo para aumentar a força do canhão.

No jogo original, o jogador pode obter um dispositivo auxiliar denominado Força. Isso se assemelha a uma bola laranja brilhante. A Força pode ser anexada à frente ou atrás da espaçonave do jogador, ou desconectada para voar livremente. Quando anexada, a Força fornece uma das três diferentes armas poderosas, além do canhão principal e do canhão de ondas. Quando destacadas, essas armas não podem ser usadas, mas a Força irá, em vez disso, recorrer a um conjunto secundário de armas, que podem ser disparadas pelo jogador mesmo se a Força estiver distante da espaçonave. A Força tem um uso secundário como escudo; é completamente indestrutível e pode bloquear a maioria das coisas disparadas contra ele, bem como danificar ou destruir inimigos com o contato.

Você pode jogar a versão original do *R-type* através deste link: `https://www.retrogames.cc/arcade-games/r-type-us.html`. Um vídeo com todo o *gameplay* do jogo pode ser visto através deste link: `https://www.youtube.com/watch?v=pVWtI0426mU`.

Nesse trabalho, você vai implementar uma versão bem básica e rudimentar desse jogo. Nessa versão, o jogador controla a nave que possui dois tipos de tiros apenas. O primeiro, que é o mais simples, é um tiro único que, quando acerta um inimigo, o destrói e também é destruído. O segundo, mais poderoso, é ativado quando o jogador segura o botão de tiro (tecla espaço) até que o raio do tiro atinja o seu valor máximo. Quando o tiro é disparado nessa condição, o tiro destrói todos os inimigos que encontrar na sua trajetória, e só é destruído quando ultrapassar os limites da tela. Em ambos os casos, apenas um tiro pode ser disparado por vez, ou seja, um segundo tiro só poderá ser disparado depois que o primeiro não existir mais, ou porque colidiu, ou porque ultrapassou os limites da tela.

Durante o jogo, o jogador perde (ou deve ser penalizado) quando colide com obstáculos. Há dois tipos de obstáculos: inimigos e blocos. Inimigos são objetos de diferentes tamanhos e cores que passam pela tela da esquerda para a direita em diferentes velocidades. No seu jogo, você deve ter que gerenciar pelo menos 10 inimigos ao mesmo tempo e o jogo deve conter situações em que pelo menos 6 inimigos estejam na tela ao mesmo tempo. Blocos são objetos maiores, retangulares, que passam pela tela de forma mais lenta. Blocos devem ter tamanhos e posições variáveis, mas devem ocupar, no mínimo, um quinto da altura da tela e devem ter comprimento pelo menos equivalente à largura da tela. Enquanto inimigos podem ser destruídos, blocos não podem. Inimigos podem ser destruídos (1) por tiros do jogador e (2) por colidirem com outros inimigos ou com (3) blocos. Nessa versão básica do jogo, você só precisa controlar um único bloco por vez.

O jogador obtém pontos sempre que ele destruir inimigos. A pontuação obtida deve variar de inimigo para inimigo, por exemplo, inimigos maiores devem dar mais pontos. Se a pontuação for a maior pontuação

obtida até o momento, uma mensagem indicando "novo recorde" deve ser exibida e a pontuação deve ser registrada em arquivo. Veja o exemplo do jogo disponibilizado pelo professor no KIT_DEV_ALLEGRO (`https://www.dropbox.com/s/5ngnjsvlb4iohw5/KIT_DEV_ALLEGRO.zip?dl=0`).

2 Critérios de Avaliação

Este jogo pode ser tão complexo quanto você deseja, mas há uma versão básica que lhe garante os 20 pontos do trabalho prático. Abaixo as funcionalidades que devem ser implementadas na versão básica do jogo:

- **Controle preciso dos movimentos da nave.** O jogador deve ter acesso a teclas que controlam a movimentação da nave e essa deve responder aos comandos do jogador. *(2 pontos)*
- **Tiro básico.** O jogador deve ter acesso a uma tecla que dispara o tiro do canhão do tanque. Na sua versão básica, o tiro não deve mudar de direção depois que ele foi lançado e só deve para depois que ele sair da tela ou colidir 1) com o tanque inimigo ou 2) com um obstáculo do cenário. Além disso, na versão básica do jogo um canhão só pode disparar 1 tiro por vez. No exemplo fornecido pelo professor, use a tecla `espaço` para atirar.*(2 pontos)*
- **Tiro avançado.** Ao segurar o botão de tiro, o tiro deve aumentar de tamanho e, quando disparado no seu tamanho máximo, deve ser capaz de eliminar todos os inimigos que estejam em sua rota.*(2 pontos)*
- **Blocos.** O cenário deve conter pelo menos um bloco que elimina o jogador e inimigos que colidem com ele. Além disso, blocos devem ter tamanhos e posições variáveis, mas devem ocupar, no mínimo, um quinto da altura da tela e devem ter comprimento pelo menos equivalente à largura da tela. Enquanto inimigos podem ser destruídos, blocos não podem. *(2 pontos)*
- **Controle de colisões.** Os objetos presentes no cenário não podem se sobrepor. Tiros não podem atravessar obstáculos e nem inimigos. Da mesma forma, os movimentos dos inimigos e da nave do jogador devem ser limitados aos espaços vazios do cenário, ou seja, não podem sair da tela e não podem atravessar blocos. *(4 pontos)*
- **Pontuação.** Cada inimigo destruído deve aumentar a pontuação do jogador. Inimigos devem oferecer valores em pontuação diferentes, de acordo com algum critério (ex: tamanho). O cenário deve exibir os pontos ganhos pelos jogadores. *(2 pontos)*
- **Fim de jogo.** O jogo deve terminar quando o jogador colidir com um inimigo ou com um bloco. *(1 ponto)*
- **Recorde.** A maior pontuação registrada (recorde) deve ser armazenada em um arquivo. O valor do recorde deve ser exibido sempre que o personagem terminar o jogo de forma vitoriosa. Se além disso ele também bater o recorde, uma mensagem informativa deve ser apresentada para ele. *(3 pontos)*
- **Documentação.** Deve conter o manual de uso, que descreve como operar o jogo, e detalhes da implementação, que descreve brevemente os trechos de código e as estruturas de dados desenvolvidas por você. Exemplos de documentação podem ser baixados na página da disciplina ou através do link `https://drive.google.com/open?id=1KPl5y2DVEZqTW-Rrhor5SFhGLffGOJ0r`. *(2 pontos)*

2.1 Conhecimento do Código

Conhecimento do aluno sobre o código apresentado será verificado via prova oral, que será dada no formato de uma entrevista. Sua nota total será multiplicada pela sua nota da prova oral, que vale 1. Assim, se você tirar 0.5 na prova oral, sua nota será dividia por 2.

2.2 Pontos Extras

Além dos pontos acima, o professor pode atribuir até 10 pontos a mais caso o aluno implemente extras, tais como:

1. Usar imagens e animações do tipo *sprite*;
2. Gerar diferentes tipos de cenários;
3. Permitir diferentes tipos de ataques, que produzem efeitos diferentes;
4. Colocar sons e músicas;
5. Implementar animações para o movimento das ações;
6. Implementar fases;
7. Implementar vidas ou pontos de vida;
8. Criar *addons* e *power-ups* que podem, por exemplo, restaurar os pontos de vida do jogador ou ataques mais poderosos;
9. Implementar modo de dois jogadores;
10. Implementar opção de salvar o jogo;
11. Implementar mais de um tipo de nave;
12. Implementar menus e diferentes níveis de dificuldade;
13. Possibilitar que mais de bloco esteja na tela ao mesmo tempo;
14. Implementar tiros para os inimigos;
15. Implementar chefões finais;
16. Implementar diferentes tipos de monstros, com ataques diferentes;
17. **Qualquer outro extra que você ache interessante!**

IMPORTANTÍSSIMO: Pontos extras só serão dados aos alunos que obtiveram mais de 50% **dos pontos nas provas, ou seja, mais de** 36 **no somatório das três provas.**

3 Como eu faço?

Apesar da descrição fazer o trabalho parecer complicado, ele é bastante simples. Tudo que o aluno precisa saber para desenvolver este jogo são os conhecimentos adquiridos na disciplina e um pequeno entendimento de desenvolvimento de aplicações gráficas. Assim como são necessárias bibliotecas novas para a utilização de funções não nativas da linguagem C, como a math.h, uma biblioteca também é necessária para que se utilize funções gráficas. Para este trabalho, pede-se que se utilize a biblioteca Allegro5, que fornece inúmeras funções que podem ajudar no desenvolvimento deste trabalho. Os vídeos abaixo ensinam como instalar a biblioteca Allegro5 em um ambiente Windows com o MingW instalado:

`https://www.youtube.com/watch?v=AezxBP687n8`
`https://www.youtube.com/watch?v=cgqjzJzm00w`

4 Roteiro de Desenvolvimento Sugerido

Como o jogo é complexo, identificar a sequência de funcionalidades que devem ser desenvolvidas pode ser um problema. Assim, a seguir estão descritas etapas de desenvolvimento sugeridas, colocadas em ordem cronológica.

1. Implementar o desenho do cenário do jogo;
2. Implementar o desenho da nave e implementar o seu movimento;
3. Implementar o bloco e o seu movimento;
4. Implementar a colisão entre a nave e o bloco;
5. Implementar o tiro básico da nave e a sua colisão com o bloco;
6. Implementar os inimigos e os seus movimentos;
7. Implementar a colisão entre os inimigos e a nave;
8. Implementar a colisão entre os inimigos e o bloco;
9. Implementar a colisão entre o tiro e os inimigos;
10. Implementar a colisão entre o tiro e o bloco;
11. Implementar o segundo tipo de tiro;
12. Implementar o sistema de pontuação;
13. Implementar o armazenamento do recorde;
14. Divirta-se implementando funcionalidades extras;
15. Escrever a documentação.

Um vídeo tutorial ensinando a desenvolver alguns dos primeiros itens desse roteiro estará disponível na página da disciplina a partir do dia 17 de dezembro de 2021.