Relatório final de INF2607 - Animação por Computador e Jogos

Rafael Diniz
Lab. Telemídia, PUC-Rio
rafaeldiniz@telemidia.puc-rio.br

3 de fevereiro de 2014

Resumo

Este é o relatório final do trabalho desenvolvido para a cadeira sobre jogos de computador do professor Bruno Feijó (INF2607).

O trabalho consistiu na implementação de um jogo de computador de nave espacial 2D (de plataforma) cujo objetivo do jogador é destruir as naves inimigas utilizando tiros. Pontos são somados a cada inimigo abatido e o número de vidas é decrementado quando o jogador é abatido.

O relatório contém um *Game Design Document*, uma descrição dos diversos aspectos do jogo além das instruções de uso e instalação.

O trabalho utilizou e extendeu o motor javaPlay e o jogo descrito no capítulo 9 do livro[1] de Bruno Feijó.

Sumário

1 Introdução

O jogo detalhado nesse relatório utiliza o motor javaPlay, que acompanha o livro de Bruno Feijó[1] sobre o tema, com algumas modificações que serão detalhadas na seção específica sobre a implementação do jogo. O código do jogo extende o código apresentado no capítulo 9 do mesmo livro.

Na segunda seção deste relatório, após essa introdução, é apresentado o *Game Design Document*. Na terceira seção deste texto o jogo é apresentado, estando incluso imagens, um breve manual de jogo e instruções de instação.

Já na quarta seção é feita uma discussão à respeito a implementação do jogo, as modificações feitas ao javaPlay e uma breve descrição das classes do jogo.

Finalmente uma seção final contém algumas considerações e perspectivas a respeito da usabilidade do jogo e da possibilidade da adaptação do jogo para ser executado em receptores de TV Digital através do uso da API Ginga-J e em aparelhos celular através do uso da API Java do Android (Dalvik VM).

2 Game Design Document

2.1 Título

_DinaBodic

2.2 Descrição

O jogo **_DinaBodic** é um jogo 2D de plataforma no qual o jogador controla uma nave equipada com arma de fogo. Naves inimigas surgem em perseguição à nave do jogador, que deve atirar contra as naves inimigas com o objetivo de destruí-las. O jogador deve se esquivar dos tiros das naves inimigas. Cada vez que a nave do jogador é acertada, uma vida é descontada, de um total inicial de 5 vidas. Cada vez que uma nave inimiga é abatida, pontos são somados para o jogador.

O objetivo do jogador é destruir todas as naves inimigas, que ao serem todas destruídas, o jogo se encerra com vitória. No caso o jogador peder as 5 vidas, o jogo termina com derrota.

2.3 Gênero do Jogo

Jogo 2D de plataforma estilo *Space Invaders*, com eixo de movimentação principal horizontal.

2.4 Cenário

O jogo possui um ambiente que remonta o espaço sideral e naves que se baseiam em modelos que reais ou fictícios.

2.5 Game Play

Os controles disponíveis para o usuário durate o jogo são as setas direcionais para cima, para baixo, para esquerda e para direita, que fazem a nave se movimentar para cima, baixo, esquerda e direita respectivamente. A barra de espaço faz com que um tiro seja disparado.

Na tela de apresentação e tela final existe um botão que deve ser clicado com o botão esquerdo do mouse e inicia ou sai do jogo, respectivamente.

2.6 Navegação e informações presentes na tela

Assim que o jogo é executado uma tela inicial é apresentada com o título do jogo e um botão com o texto *Single Player*, que quando pressionado com o botão esquerdo do mouse leva ao início da ação do jogo.

No canto superior esquerda da tela estão duas informações, acima está o número de vidas, e abaixo o número de pontos marcados.

2.7 Requisitos de vídeo

O jogo necessita de um computador com adaptador gráfico ajustado para a resolução de 1280x720 e um monitor que suporte a resolução de 1280x720, também conhecido como resoluçao "HD". Pelo fato do jogo rodar em tela cheia, caso a resolução não esteja ajustada para 1280x720, a tela do jogo ou não ocupará a totalidade da tela ou não caberá na tela, tendo em vista que o jogo não suporta trocar a resolução do sistema automaticamente.

2.8 Requisitos de áudio

O jogo necessita de uma interface de áudio que suporte a reprodução de amostras de som com 16 bits a uma taxa de 48kHz (efeitos de áudio) e 44,1kHz (som de fundo).

2.9 Requisitos de CPU e memória

Uma CPU com clock de 2GHz e 2GB de memória RAM é recomendado.

2.10 Requisitos da Máquina Virtual Java

Java 7 é recomendado.

3 Instruções de instalação

O jogo é distribuído em um arquivo do tipo ZIP, que quando descompactado deverá criar uma pasta de nome "DinaBodic". Dentro dessa pasta o jogo poderá ser executado de duas formas. Pela interface gráfica do sistema operacional deve-se somente clicar sobre o arquivo "DinaBodic.jar". Pela linha de comando, deve-se executar o comando "java -jar DinaBodic.jar".

4 O jogo _DinaBodic

O jogo inicia com uma tela de apresentação, como mostrada na Figura 1.

Figura 1. Tela de início do jogo.

Clicando com o botão esquerdo do mouse em "Single Player" inicia o jogo.

As Figuras 2 e 3 explicitam telas do jogo, sendo que com a barra de espaços o jogador atira contra os inimigos e com as setas direcionais a nave se movimenta pelo mapa.

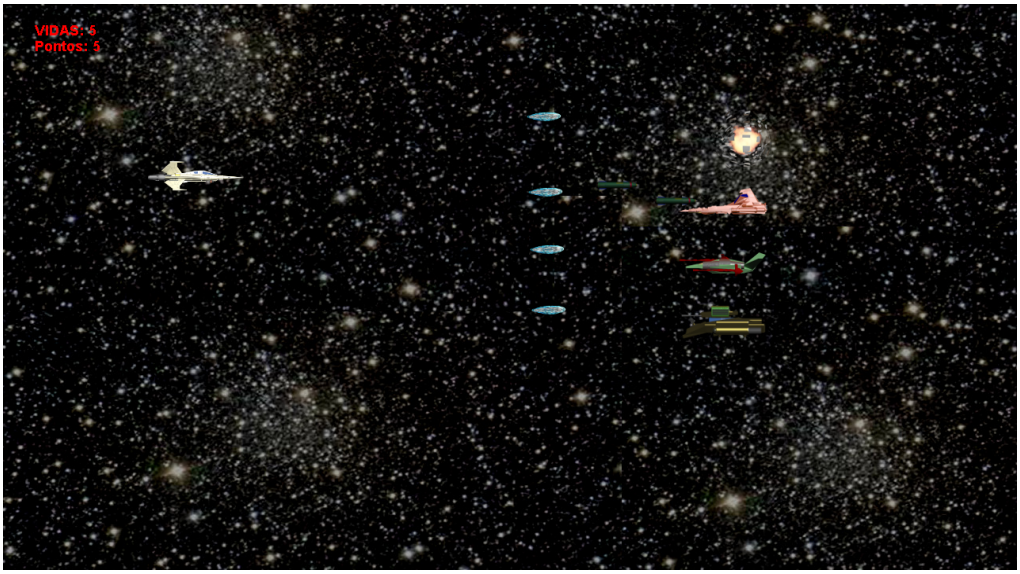

Figura 2. Tela do jogo em ação.

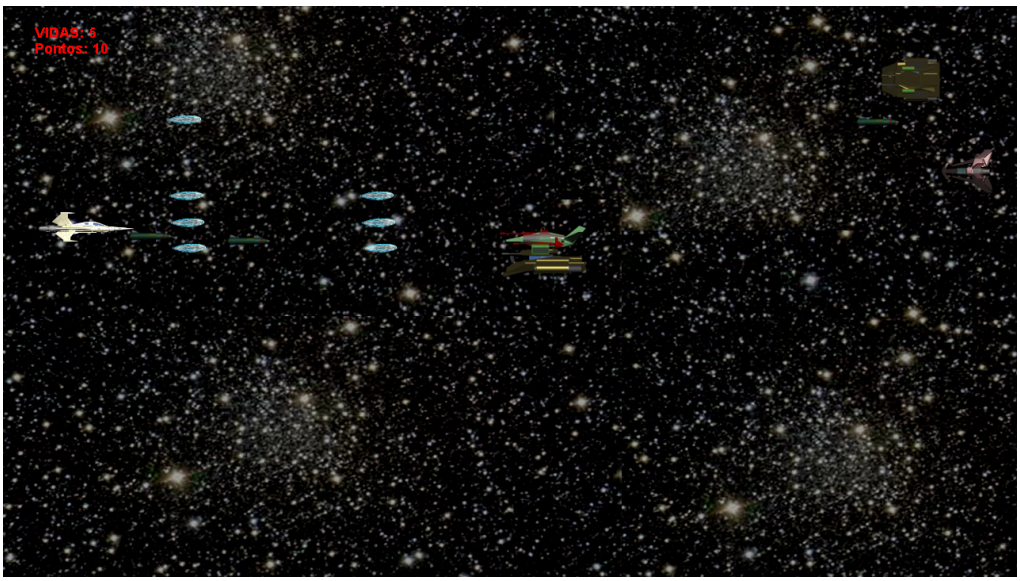

Figura 3. Tela do jogo em ação.

No caso do jogador destruir todas as naves inimigas e perder menos que 5 vidas, a tela de vitória é mostrada assim como na Figura 4.

Figura 4. Tela de vitória.

Caso o jogador perca as 5 vidas, aparece a tela de derrota assim como mostrada na Figura 5.

Figura 5. Tela de derrota.

5 Implementação do _DinaBodic

O jogo utiliza o motor javaPlay com algumas modificações, sendo as mais importantes:

- Na classe GameCanvas foi adicionado o suporte a tela cheia e foi feita uma pequena otimização de performance.
- Na classe GameEngine o método run() foi reescrito de forma a permitir que haja um controle do número de quadros por segundo/chamadas ao método step() interno ao motor, facilitando a programação do jogo em si.
- Na classe Scene, no método draw(), o desenho dos "Tiles" foi comentado do código pois o mesmo se apresentou como um problema de performance que impedia o jogo de rodar na velocidade adequada (num Core i3 com 4Gb de RAM). Consequentemente não foram utilizados "Tiles" no jogo. No entanto o suporte a colisão com os "Tiles" não se apresentou como um problema de performance, fato que permitiria, por exemplo, o desenho das barreiras diretamente no backdrop e o mapeamento dessas barreiras no arquivo de descrição de cena, mas essa técnica não foi utilizada no jogo. A realização da composição das cenas utilizando aceleração gráfica também poderia ter sido experimentado para superar os problemas de performance.
- Todos os métodos que faziam carregamento de imagem utilizavam uma API assíncrona e um laço com um número arbitrário de tentativas com o objetivo de esperar a imagem ser carregada, fato que fazia com que o motor falhasse em computadores antigos (inclusive no computador utilizado no desenvolvimento do jogo). Todos esses métodos foram alterados para utilizar métodos bloqueantes para carregamento das imagens, eliminando o problema.
- Foi adicionado um novo construtor para a classe "Sprite" no qual é possível carregar o sprite a partir de múltiplos arquivos de imagem, sendo passado somente o prefixo do nome dos arquivos que devem estar no formato *imagem-x.png*, onde x deve iniciar com 1 e ser incrementado até o número de quadros do sprite.

Tanto o codigo fonte do javaPlay modificado quanto do jogo acompanham este relatório.

Com relação às classes desenvolvidas para o jogo, as mesmas estão apresentadas abaixo com uma breve descrição:

- EndScreenController: Controlador que apresenta a tela de encerramento (vitória ou derrota).
- Enemy: Possui a lógica de movimentação das naves inimigas e disparo dos tiros.

- EnemyFire: Possui a lógica de movimentação dos tiros das naves inimigas.
- Explosion: Possui a lógica das explosões.
- Global: Algumas constantes globais como tamanho do tela e identificadores dos controladores de estado do jogo.
- Main: Possui o ponto de entrada do programa, é onde são realizados o registro dos controladores de estado do jogo.
- Player: Contém a lógica de movimentação e animação do jogador e a lógica de disparo dos tiros.
- PlayerFire: Implementa a lógica de movimentação do tiro do jogador.
- Score: Implementa os métodos que operam sobre a pontuação e número de vidas.
- SimpleController: Controlador que implementa a lógica do jogo em si. É o bloco de código mais importante do jogo, dentre outras funcionalidades implementa as colisões entre jogador e inimigos, entre tiro e inimigo, tiro e jogador, coordena a atribuição de pontos e o decremento de vidas, a reprodução dos áudios e contém as estrutura de dados dos objetos em cena.
- StartMenuController: Controlador que apresenta a tela inicial do jogo.

6 Considerações finais

O jogo desenvolvido aplicou conceitos aprendidos em aula, tanto relativos à modelagem 3D, para os sprites, como para a lógica do jogo em si. No que tange o uso do motor javaPlay, seu uso se mostrou eficiente e seu código relativamente fácil de ser entendido e extendido.

Com relação às possibilidades de portabilidade do código, pelo fato da linguagem Java ser utilizada, pode-se facilmente adaptar o aplicativo para o ambiente de TV Digital utilizando a API LWUIT (Lightweight User Interface Toolkit) ao invés da API AWT utilizada no javaPlay. É possível também, com um pouco mais de esforço, adaptar o motor javaPlay para Android utilizando a API provida pelo SDK do sistema Android.

Ainda é pretendido pelo autor adicionar um chefe de fase, animar o backdrop e adicionar um caminho com rochas nas extremidades superior e inferior do mapa para posterior publicação o jogo (na verdade, uma "demo") na plataforma Steam[1], sem custos.

7 Agradecimentos

Agradecimentos aos amigos Samir Piccolotto Issa (Unicamp), pela ajuda com os modelos 3D feitos no 3DS Max que foram utilizados para os sprites da nave principal e naves inimigas, e a Felipe Ridolfi (UFRJ) pelo áudio de fundo do jogo.

Referências

[1] Feijó, Bruno, Flávio Soares Corrêa da Silva, and Esteban Clua.
Introdução À Ciência da Computação Com Jogos.
Editora Elsevier, 2010.

[1] Steam: http://steamcommunity.com/