beat'n strike

Orientador:
Flávio Soares Correa da Silva

Alunos:
Alexandre Barbosa de Macedo
Daniel Grecco Machado
Fábio Tsuguta Matsumoto

Introdução

Jogos eletrônicos são um dos tipos de software que mais abrangem as áreas da computação. Em função desta complexidade, uma das áreas mais importantes para o desenvolvimento de jogos é a engenharia de software, pois um planejamento e modelagem adequados são fundamentais para unir harmoniosamente todas estas diferentes áreas afim de criar jogos de qualidade. A idéia do projeto é desenvolver um jogo, desde seu conceito até um demo jogável, que envolva várias das competências do curso de Computação, além de explorar alguns aspectos desta área em específico.

O Jogo

Beat'n Strike consiste de uma arena com um personagem principal, neste caso um cometa, que possui o objetivo de colidir com obstáculos. Os obstáculos por sua vez possuem formas (formas geométricas de diferentes tamanhos) e cores diferentes. O jogador pode movimentar o cometa livremente pela arena além de poder alterar a cor do mesmo, escolhendo entre quatro cores: vermelho, azul, amarelo e verde. Para ganhar pontos o jogador deve colidir o cometa com obstáculos da sua própria cor, e perde pontos caso colida com obstáculos de cores diferentes. Quanto maior o número de obstáculos de mesma cor atingidos em curto intervalo de tempo, maior será a pontuação obtida. Tudo com uma música tocando ao fundo.

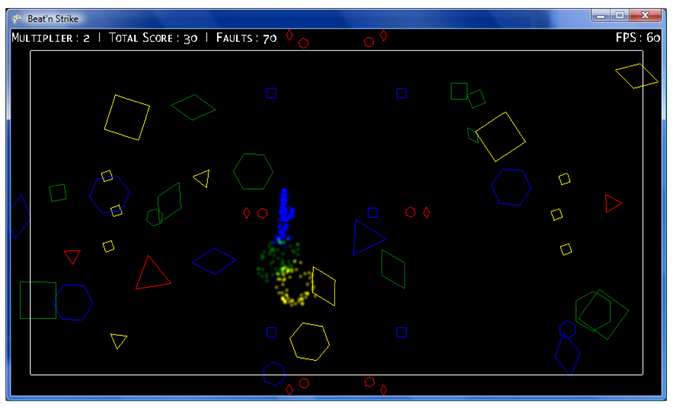

Figura 1: Cometa e obstáculos

A música é um fator muito importante para o Beat 'n Strike. O jogador tem livre escolha sobre a música de entrada, neste caso um arquivo MP3. Este arquivo é analisado e os obstáculos são gerados a partir da análise das batidas ao longo da música.

Ferramentas

O Beat'n Strike foi implementado utilizando XNA[1], um *framework* específico para o desenvolvimento de jogos para a plataforma Windows e XBOX360[2]. A linguagem utilizada foi o C#, e a IDE foi o Visual Studio 2008. Também foi utilizado o HLSL[3] (High Level Shader Language), uma linguagem de *shader* desenvolvida também pela Microsoft.

Desenvolvimento

O projeto foi concebido em 3 fases. Na primeira foi desenvolvida uma biblioteca de áudio. A fase seguinte foi implementar os componentes gráficos e a física do jogo. Finalmente, na última fase, integramos os módulos das fases anteriores, adicionando algoritmos para a geração dos obstáculos e desenvolvemos a interface do jogo.

Implementação

Para que os obstáculos pudessem ser gerados a partir do áudio escolhido pelo usuário selecionamos dois atributos característicos da música para analisá-los e representá-los na interface do jogo. O ritmo, que faz parte da organização temporal da música e a altura (frequência) que é característica do som que a compõe. Somente a onda sonora (Figura 2) não fornecia diretamente tais informações.

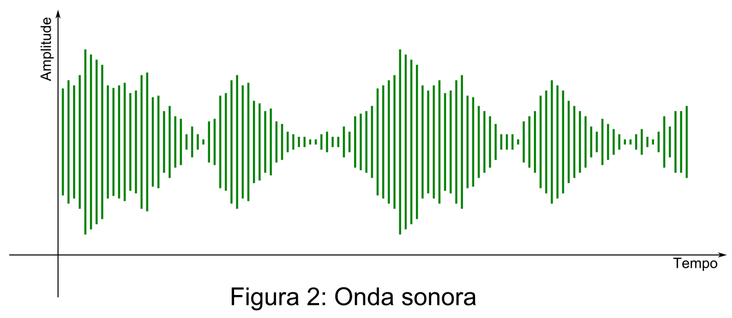

Figura 2: Onda sonora

Durante o *encoding* do mp3 a onda sonora é transformada em um espectro de frequências (Figura 3) de maneira que no processo de *decoding* podemos capturá-lo e a partir dele extrair a altura e o ritmo.

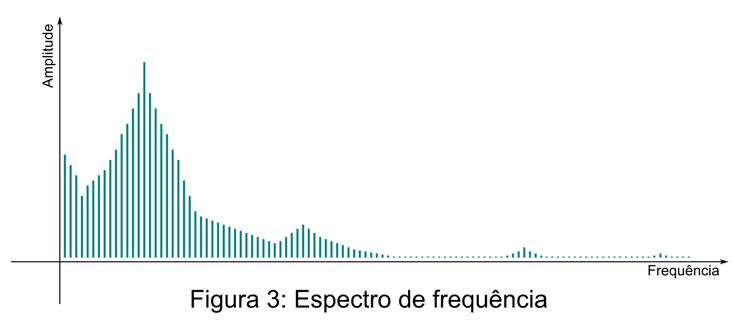

Figura 3: Espectro de frequência

O passo seguinte foi repartir o domínio em bandas de tamanhos convenientes, classificadas segundo o modelo da audição humana.

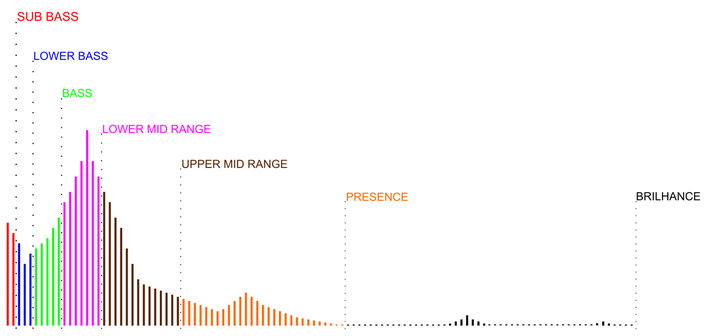

Figura 4: Classificação do espectro

Quando condensamos estes sete valores através do cálculo das médias o que obtivemos foi um gráfico semelhante ao que pode ser visualizado em muitos *players* populares.

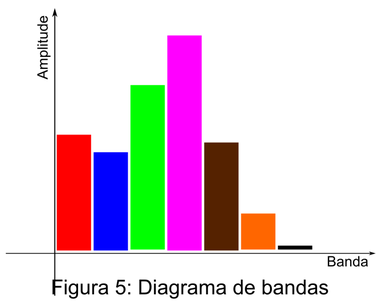

Figura 5: Diagrama de bandas

Cada valor pode ser utilizado para definir diversas partes do jogo, por exemplo: geração de obstáculos, *background*, animações e pontuações variadas.

A outra característica, o ritmo, deve ser capturada no decorrer do tempo. Para determinar o ritmo contamos o número de batidas separadamente em cada banda.

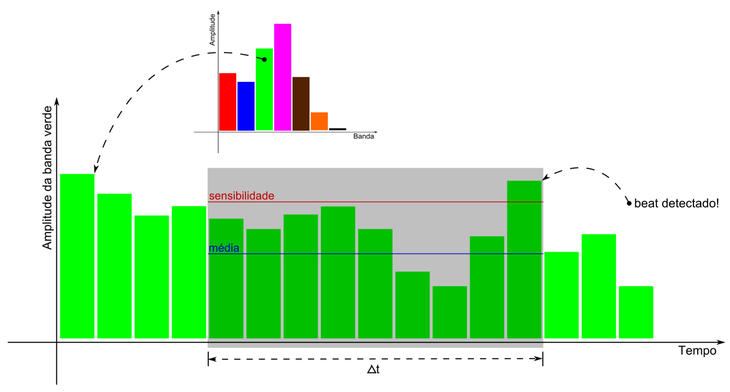

Figura 6: Captura da batida da música

Neste caso, um buffer guarda um histórico das médias de cada banda relativo a um período pré-determinado em segundos. A cada instante, a média gerada a partir do histórico de uma certa banda no último período, determina se o valor atual da banda é uma batida ou não.

Uma constante diferente para cada banda é aplicada sobre a média do histórico para ajustar a sensibilidade do detector de batidas.

Desta maneira podemos determinar qual banda gera mais batidas e definir perfis para cada estilo musical. A Figura 6 ilustra o processo descrito.

Além do módulo de manipulação de áudio, implementamos uma engine de partículas para a criação do cometa e efeitos das explosões, e também uma biblioteca de física simples para a geração da cauda do cometa.

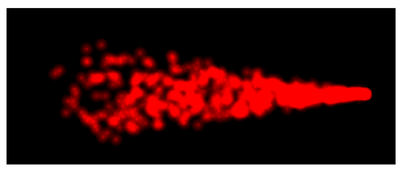

Figura 7: Cometa de partículas

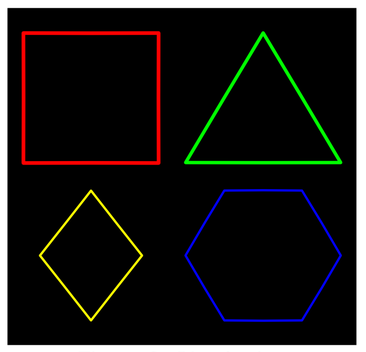

Figura 8: Obstáculos

O jogo se utiliza basicamente de figuras geométricas para a formação da arena e dos obstáculos, tornando o tratamento de colisões mais simples. As figuras 7 e 8 ilustram alguns dos componentes do jogo

Resultados

Como resultado obtivemos, além de um jogo completo que captura elementos da música, uma *engine* simples para criação de partículas, e uma biblioteca de áudio que pode ser utilizada para o desenvolvimento de outros jogos e aplicações. Abaixo temos alguns *screenshots* do jogo.

Figura 9: Tela inicial

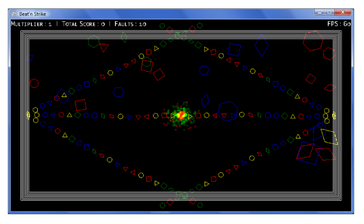

Figura 10: *Bass frame* em ação

Figura 11: Tela de instruções

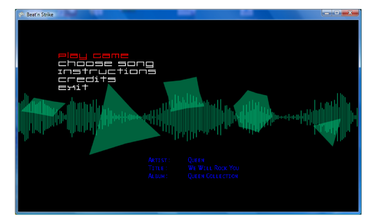

Figura 12: Menu principal

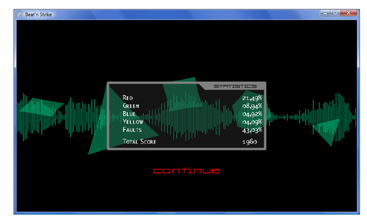

Figura 13: Tela de estatísticas

Referências

[1] http://www.xna.com
[2] http://www.xbox.com
[3] http://en.wikipedia.org/wiki/High_Level_Shader_Language