Motor de Jogos e Multimídia

Sistema de áudio e imagem de uma *game engine*

Slides por:
Gustavo Ferreira Ceccon (gustavo.ceccon@usp.br)

Este material é uma criação do
Time de Ensino de Desenvolvimento de Jogos Eletrônicos (TEDJE)
Filiado ao grupo de cultura e extensão Fellowship of the Game (FoG), vinculado ao ICMC - USP

Este material possui licença CC By-NC-SA. Mais informações em:
https://creativecommons.org/licenses/by-nc-sa/4.0/

Objetivos

- ➔ Introduzir como áudio e imagem são representados e armazenados
- ➔ Explicar como áudio e imagem são utilizados em jogos
- ➔ Mostrar os diferentes formatos de compressão de áudio e imagem

Objetivos

- Apresentar diferentes tipos de efeitos sonoros utilizados em jogos
  - Low Pass Filter
  - High Pass Filter
  - Efeito Doppler
  - Reverberação

Índice

1. Introdução

1. Introdução

- Representação de áudio e imagem (em disco)
  - Tamanho
  - Qualidade
- Representação de áudio e imagem (em memória)
  - Tamanho
  - *Streaming*

1. Introdução

- ➔ Carregamento
  - ◆ Síncrono
  - ◆ Assíncrono
    - *Loading screen*
  - ◆ Carregamento do disco é demorado
    - Como aproveitar o tempo

1. Introdução

- Disco
  - Muitos arquivos são problemáticos, além de poder desperdiçar memória (devido aos formatos), demoram para buscar os dados (*seek*)
  - Solução: compactar arquivos
  - Problema: compactar de forma esperta

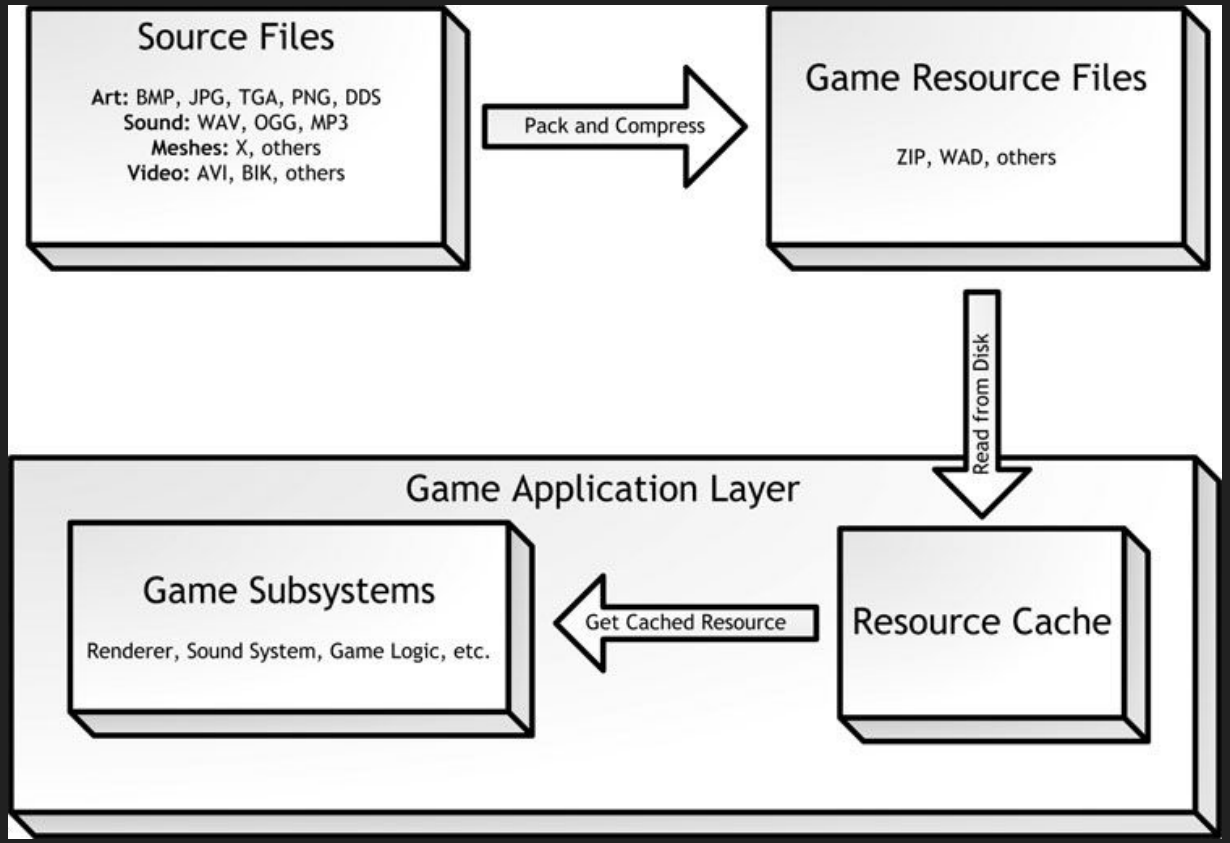
1. Introdução

- ➔ Compactação
  - Não há perda de dados
  - Deixando os arquivos em um arquivo contínuo
  - Pode ou não haver compressão
  - Não é uma boa ideia compactar todo os arquivos
    - Usar *game design* ao seu favor

1. Introdução

- Cache de recursos
  - Guardar recursos que estão sendo mais utilizados
  - Cache cheia
    - Algoritmos de substituição
    - *Cache-miss*
  - Recursos podem ter prioridades

Cache de Recursos



McShaffry, Mike & Graham, David. Game Coding Complete 4th Edition

1. Introdução

- ➔ Tipos de carregamento
  - ◆ Tudo de uma vez
    - Mario
  - ◆ Apenas em pontos estratégicos
    - FPS
  - ◆ Constantemente
    - Mundo aberto

1. Introdução

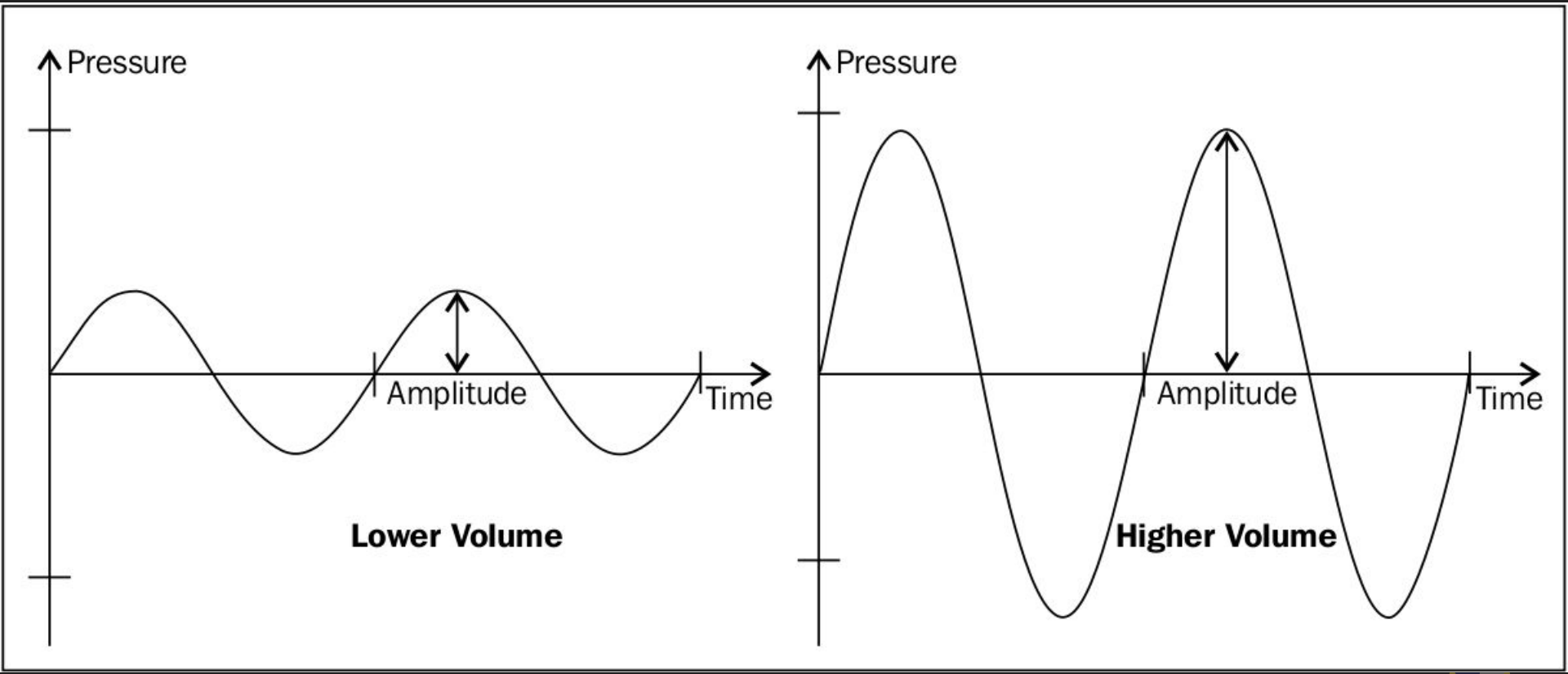
- Gerenciador de memória
  - Aumenta a performance se bem implementada
  - Complicada de implementar
  - Podem ser implementadas ferramentas como *garbage collector*

2. Áudio

2. Áudio

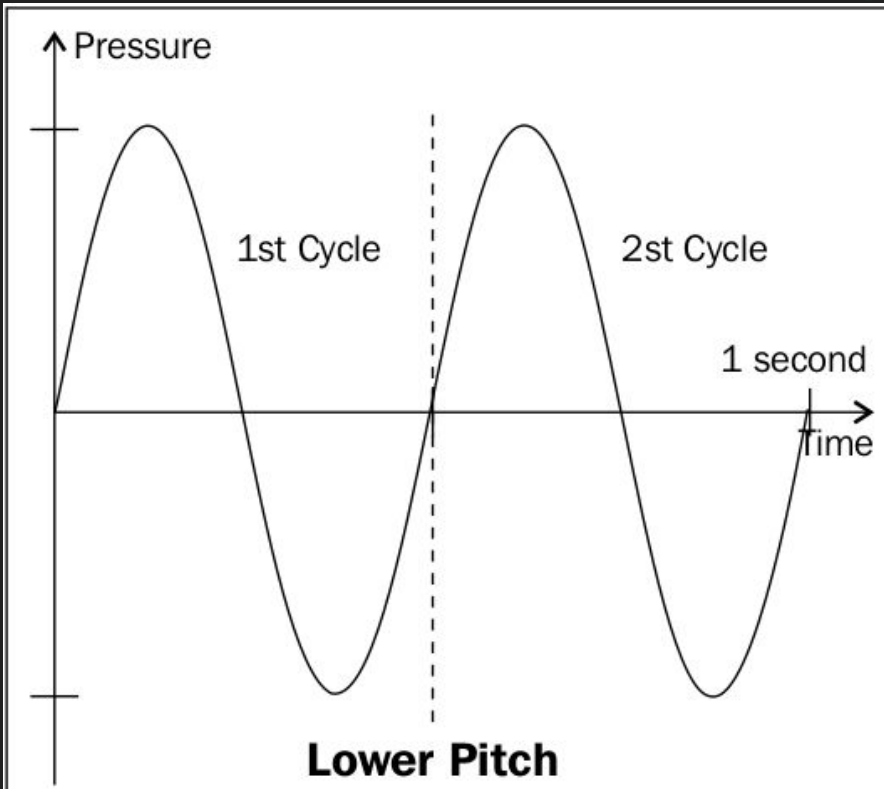
- Representação de áudio
  - Sinal
    - Onda seno: frequência (*pitch*), amplitude (volume) e fase
  - Sinal digital
    - Amostragem
    - Quantização

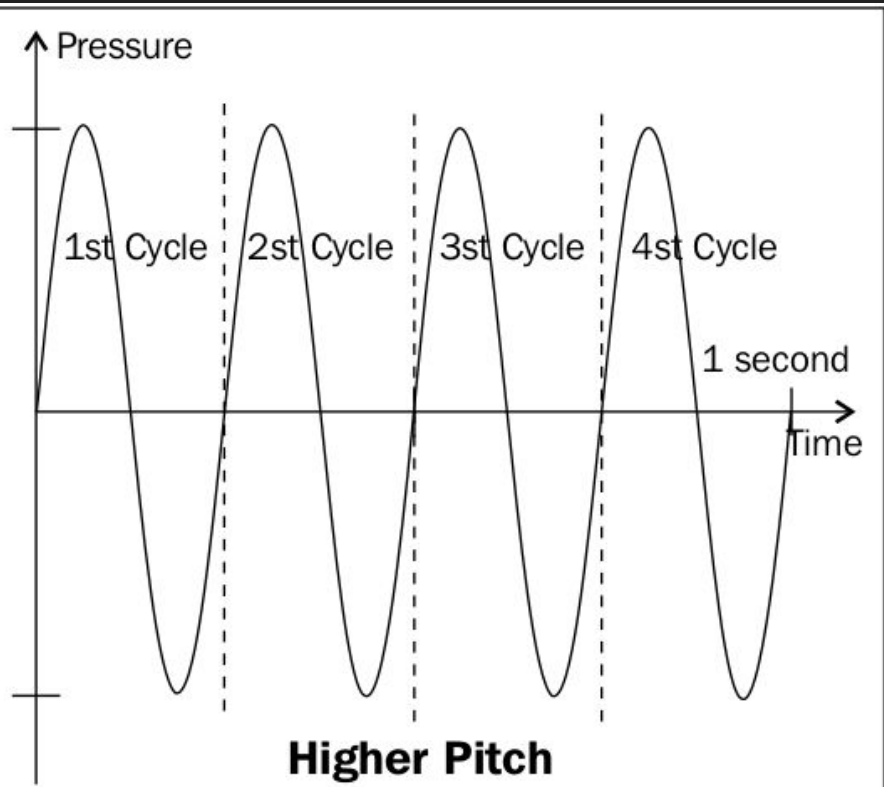
Volume



Gouveia, David. Getting Started with C++ Audio Programming for Game Development

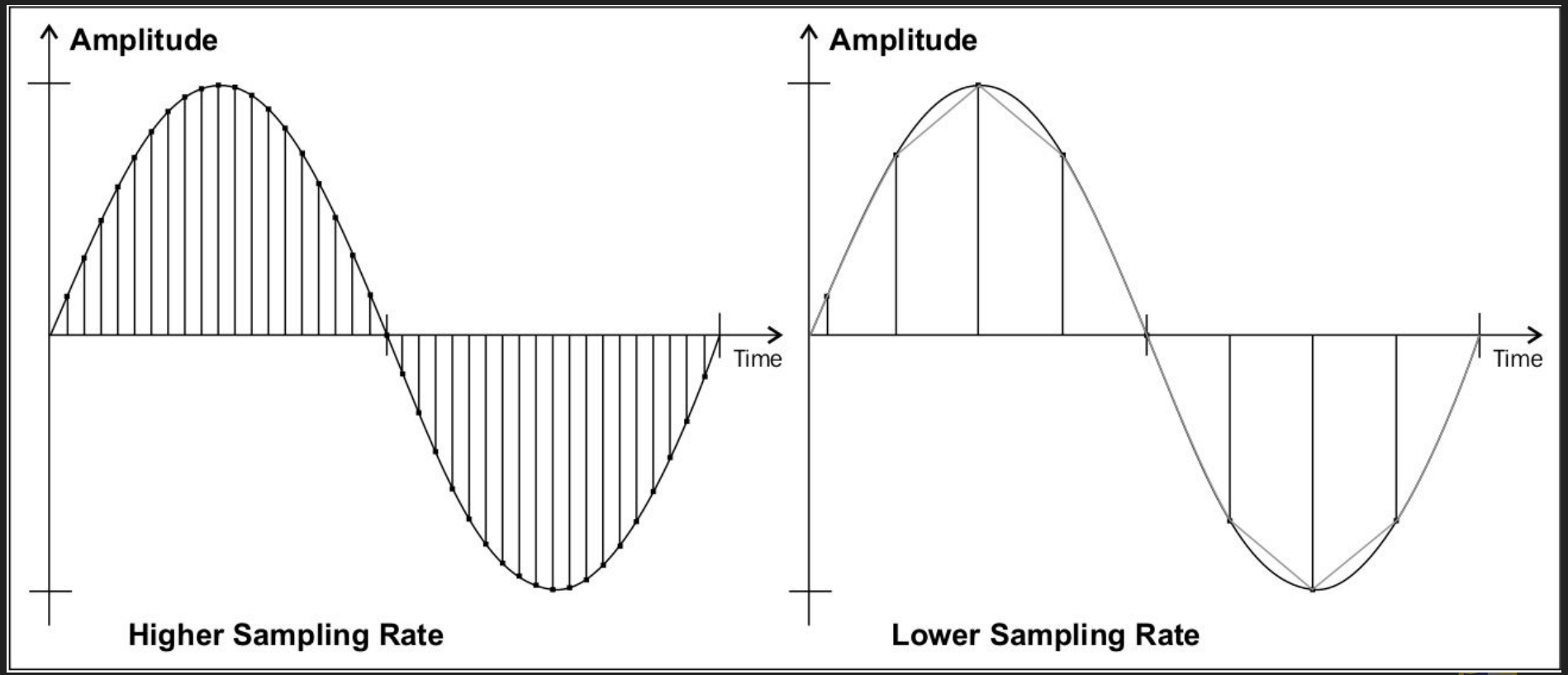
Frequência





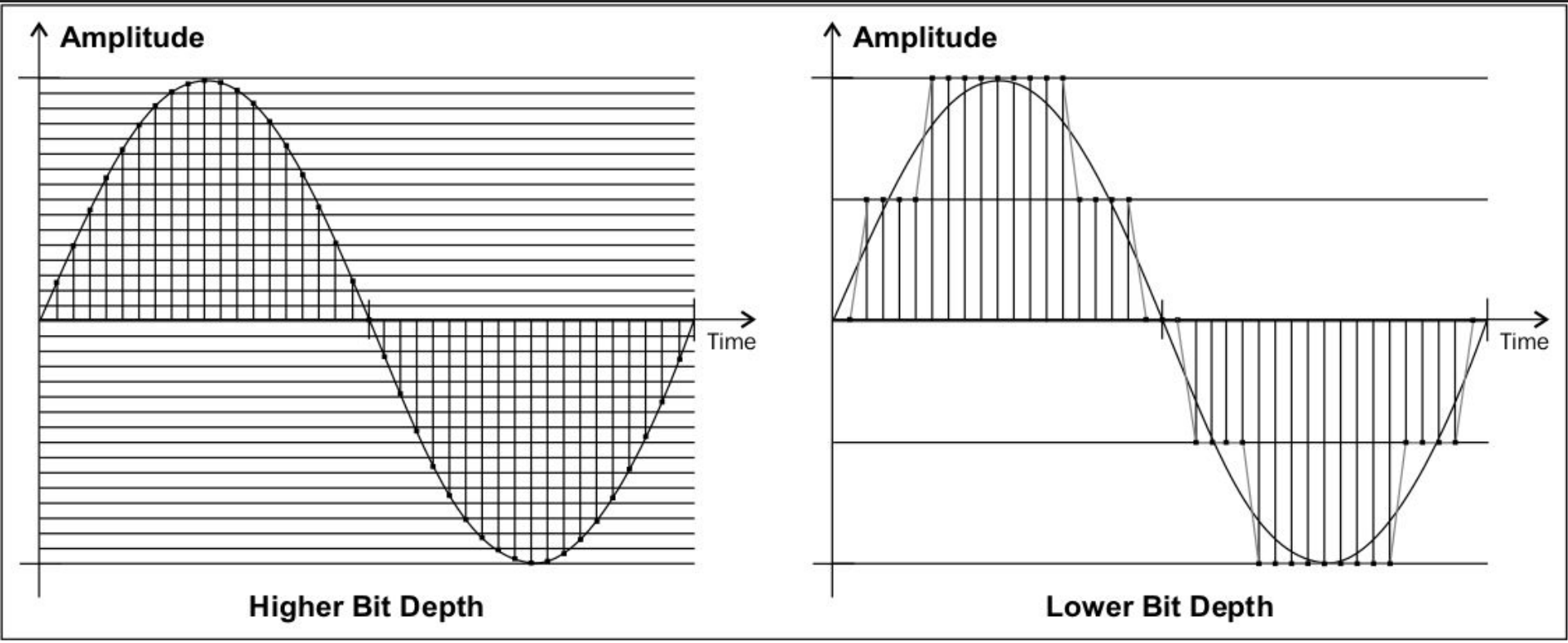
Gouveia, David. Getting Started with C++ Audio Programming for Game Development

Taxa de amostragem



Gouveia, David. Getting Started with C++ Audio Programming for Game Development

Profundidade de Bit



Gouveia, David. Getting Started with C++ Audio Programming for Game Development

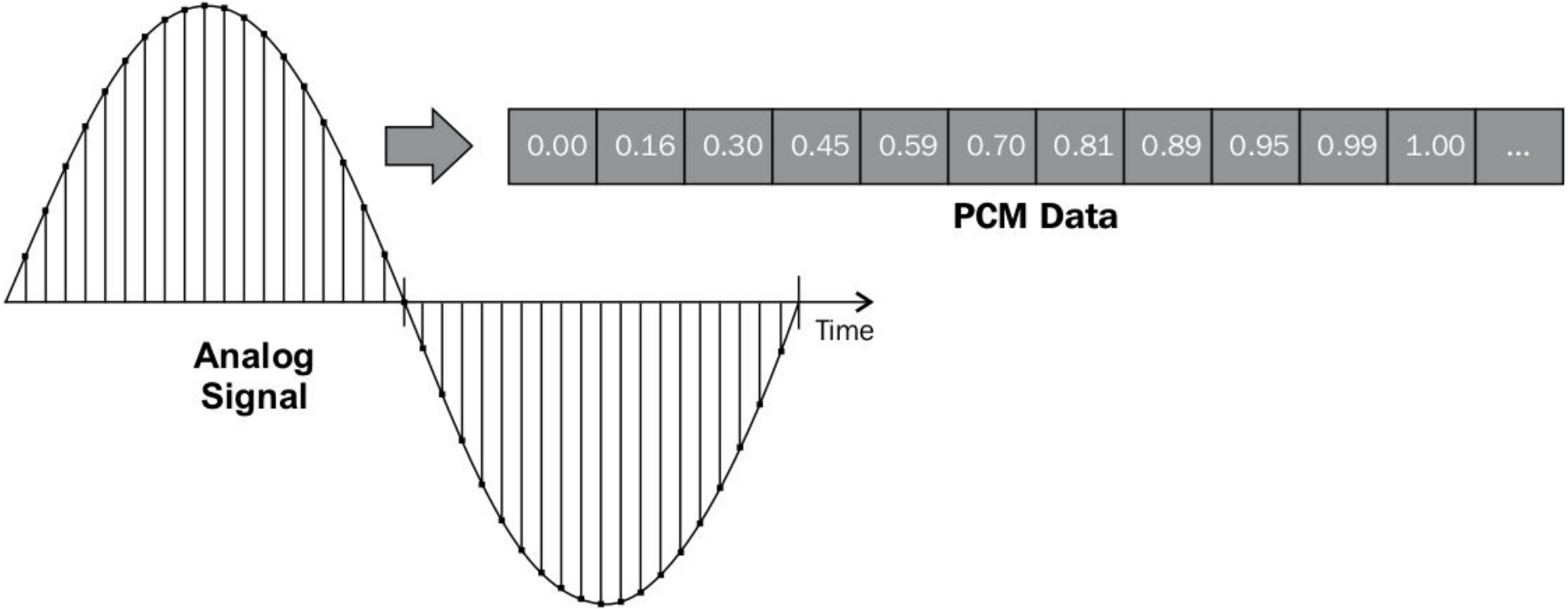
2. Áudio

- Problemas
  - Amostragem: teorema de Nyquist-Shannon
  - Quantização: distorção de som

2. Áudio

- PCM - *Pulse Code Modulation*
  - Técnica anterior
  - Quantização em dois formatos (geralmente)
    - Inteiro (16 bits)
    - Ponto flutuante (32 bits)
      - -1.0 a 1.0 (necessária conversão)

PCM

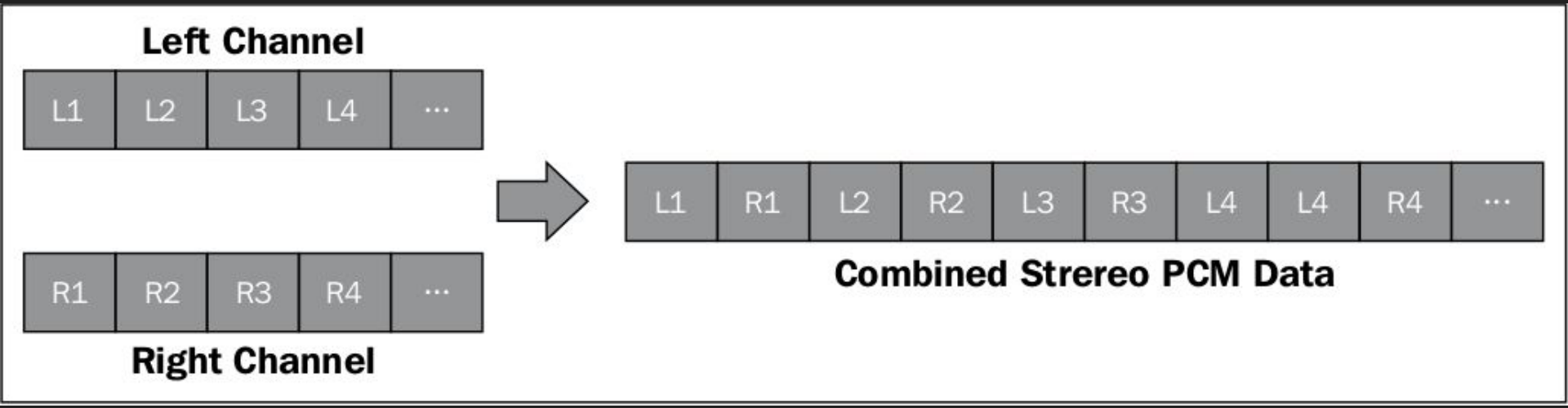


Gouveia, David. Getting Started with C++ Audio Programming for Game Development

2. Áudio

- ➔ PCM - *Pulse Code Modulation*
  - ◆ Canais
    - Mono (1.0)
    - Stereo (2.0)
    - Surrond (5.1)
    - Representação em PCM

Multicanais

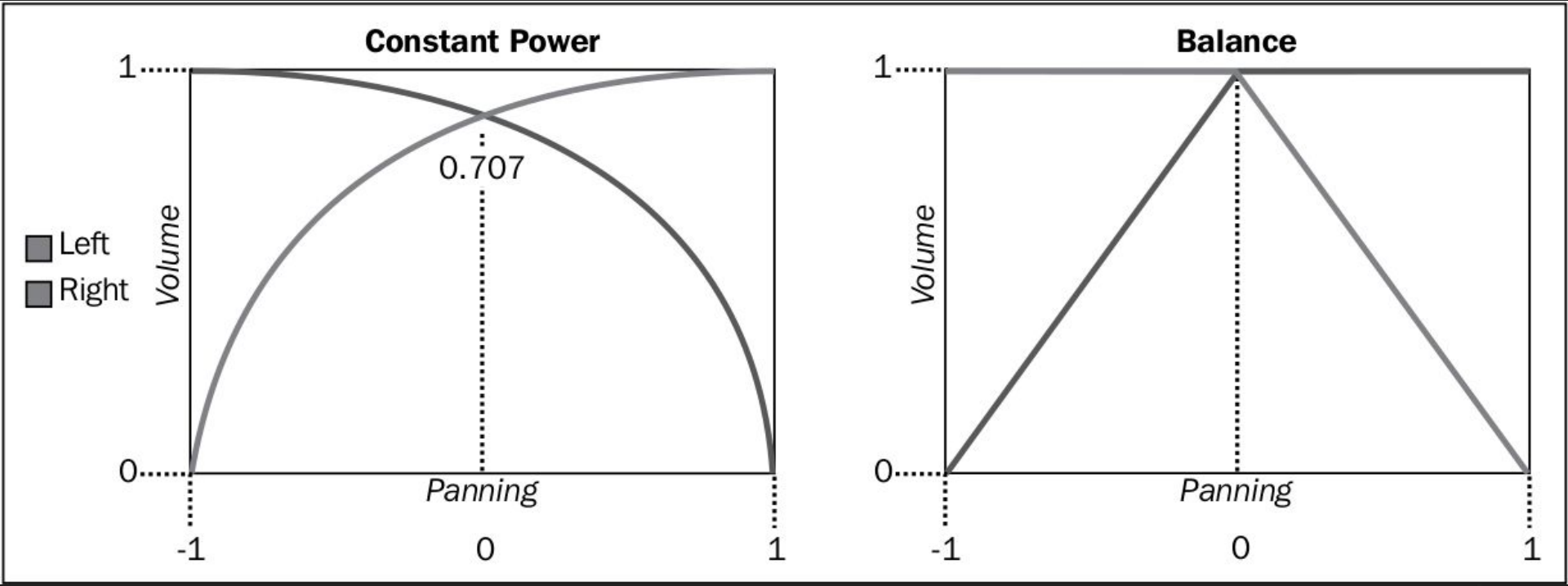


Gouveia, David. Getting Started with C++ Audio Programming for Game Development

2. Áudio

- ➔ Diferenças de 2D e 3D
  - ◆ *Panning*
    - Linear
    - Coordenadas polares

Panning

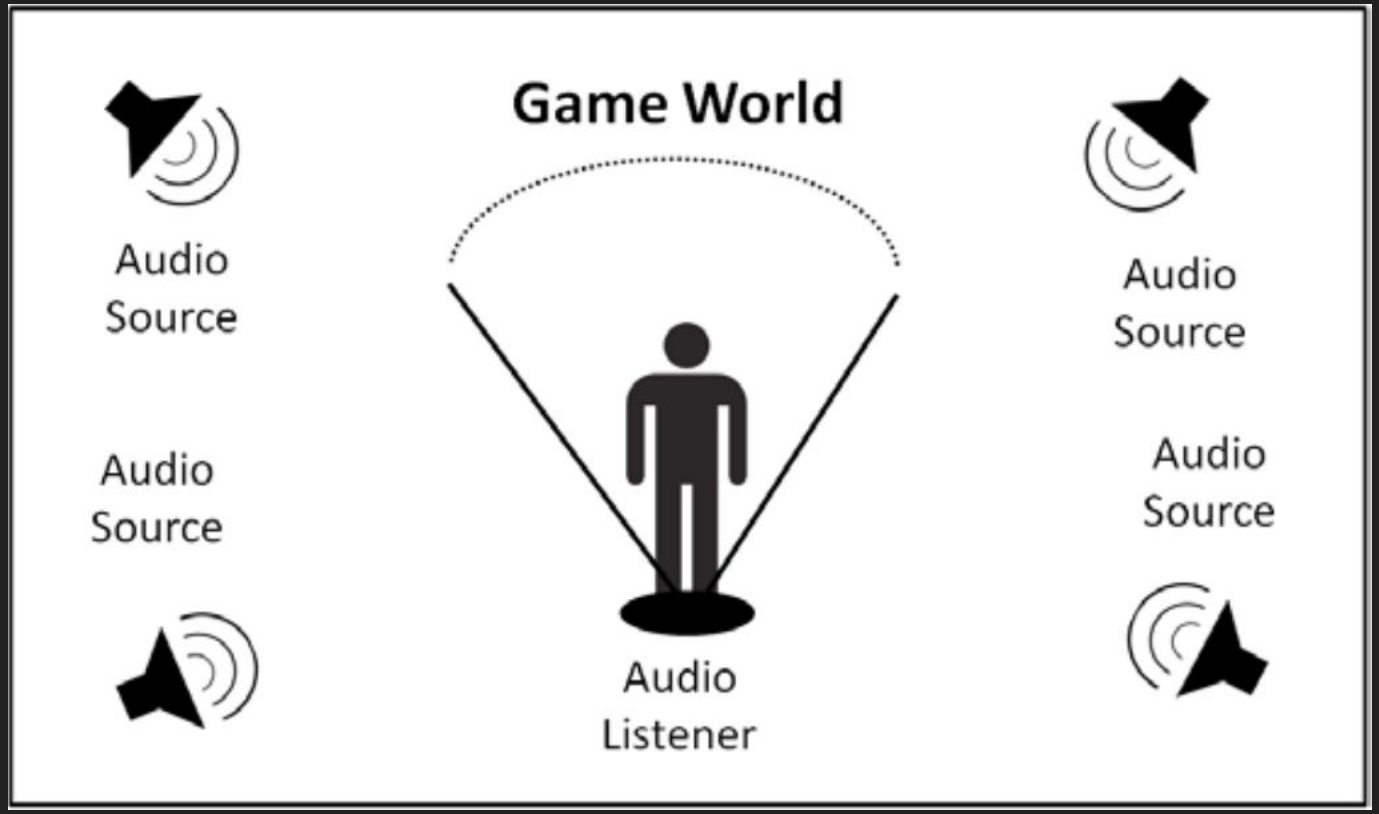


Gouveia, David. Getting Started with C++ Audio Programming for Game Development

2. Áudio

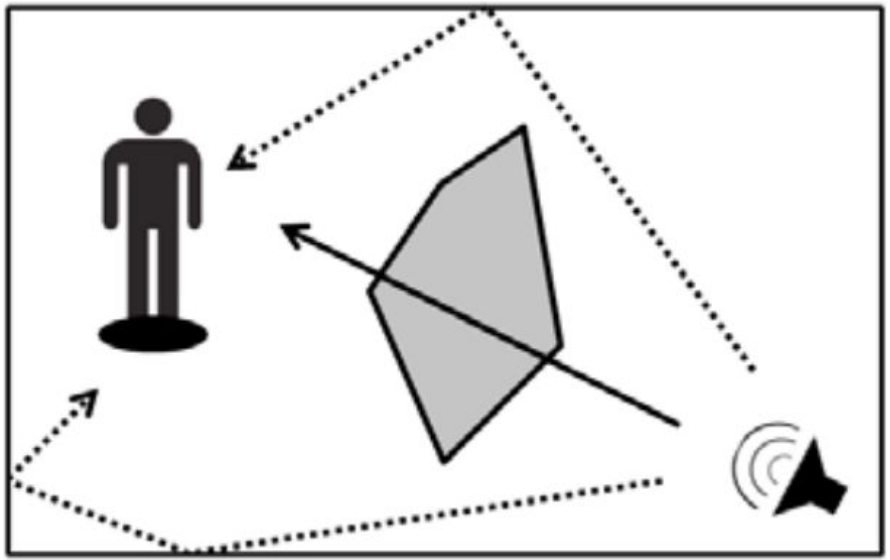
- 3D
  - Obstrução
  - Oclusão
  - Baseado em geometrias
  - Geralmente filtros quebram o galho
  - Audio 3D às vezes é essencial: FPS

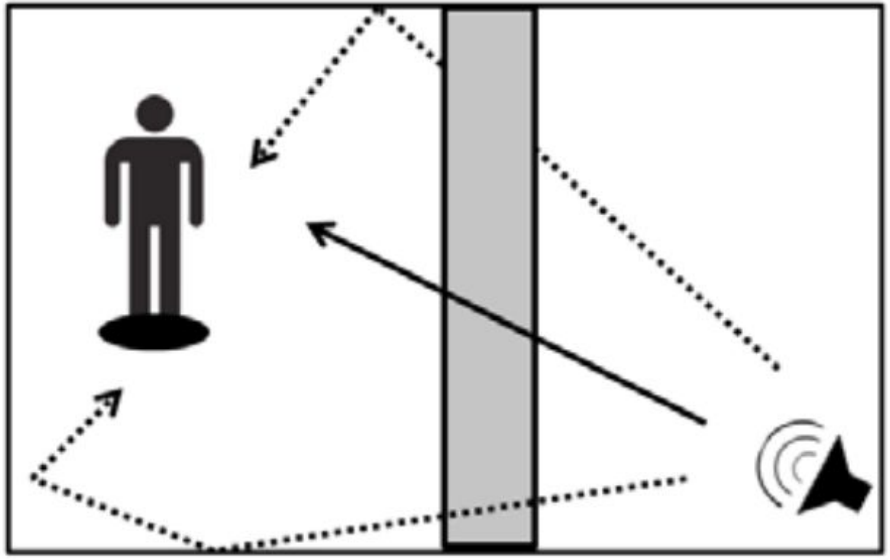
Áudio 3D



Gouveia, David. Getting Started with C++ Audio Programming for Game Development

Obstrução e Oclusão





Gouveia, David. Getting Started with C++ Audio Programming for Game Development

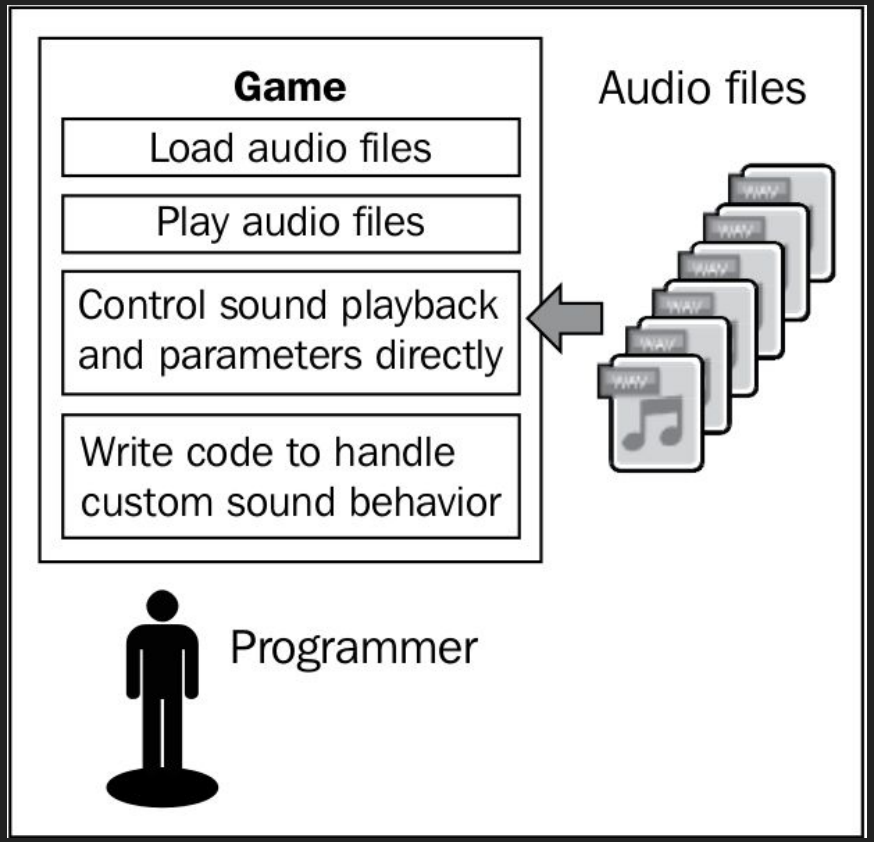
2. Áudio

- Ouvinte (*Listener*)
  - Posição (*panning*, distância)
  - Velocidade (Doppler)
  - Vetor cima (*panning*)
  - Vetor frente (*panning*)

2. Áudio

- Fonte (*Source*)
  - Posição (*panning*, distância)
  - Velocidade (Doppler)
  - Direção (fontes com propagação em formato de cone)
  - Intervalos (Min / Max)
    - Variação do volume: *rolloff model* (i.e logaritmo)

Pipeline de áudio no jogo



Gouveia, David. Getting Started with C++ Audio Programming for Game Development

2. Áudio

- ➔ Mixer
  - ◆ Agrupar áudios em canais
  - ◆ Aplicação de efeitos, volume etc.
  - ◆ Canais ligando em canais
    - ● *Side chain*

2. Áudio

- Transição
  - Importante para *game design*
  - Criar efeitos de transições entre cenas, ou músicas
  - Interpolação entre áudios, parâmetros, controladores etc.

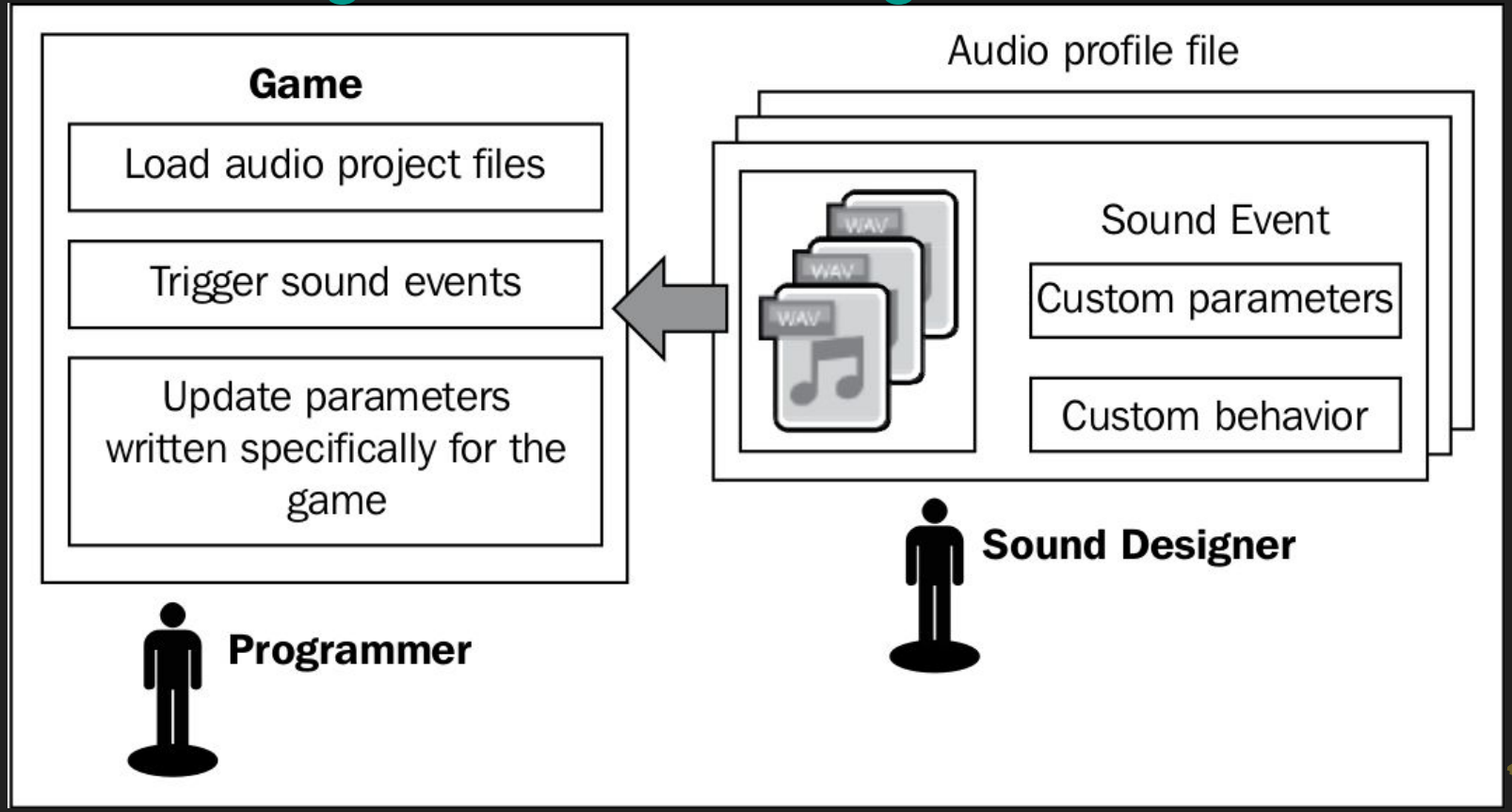
2. Áudio

- ➔ Áudio inteligente
  - ◆ Sistema de eventos
  - ◆ Música interativa
    - ● *Cues*
    - ● Sincronização

FMOD

Gouveia, David. Getting Started with C++ Audio Programming for Game Development

Programador e Designer de Som

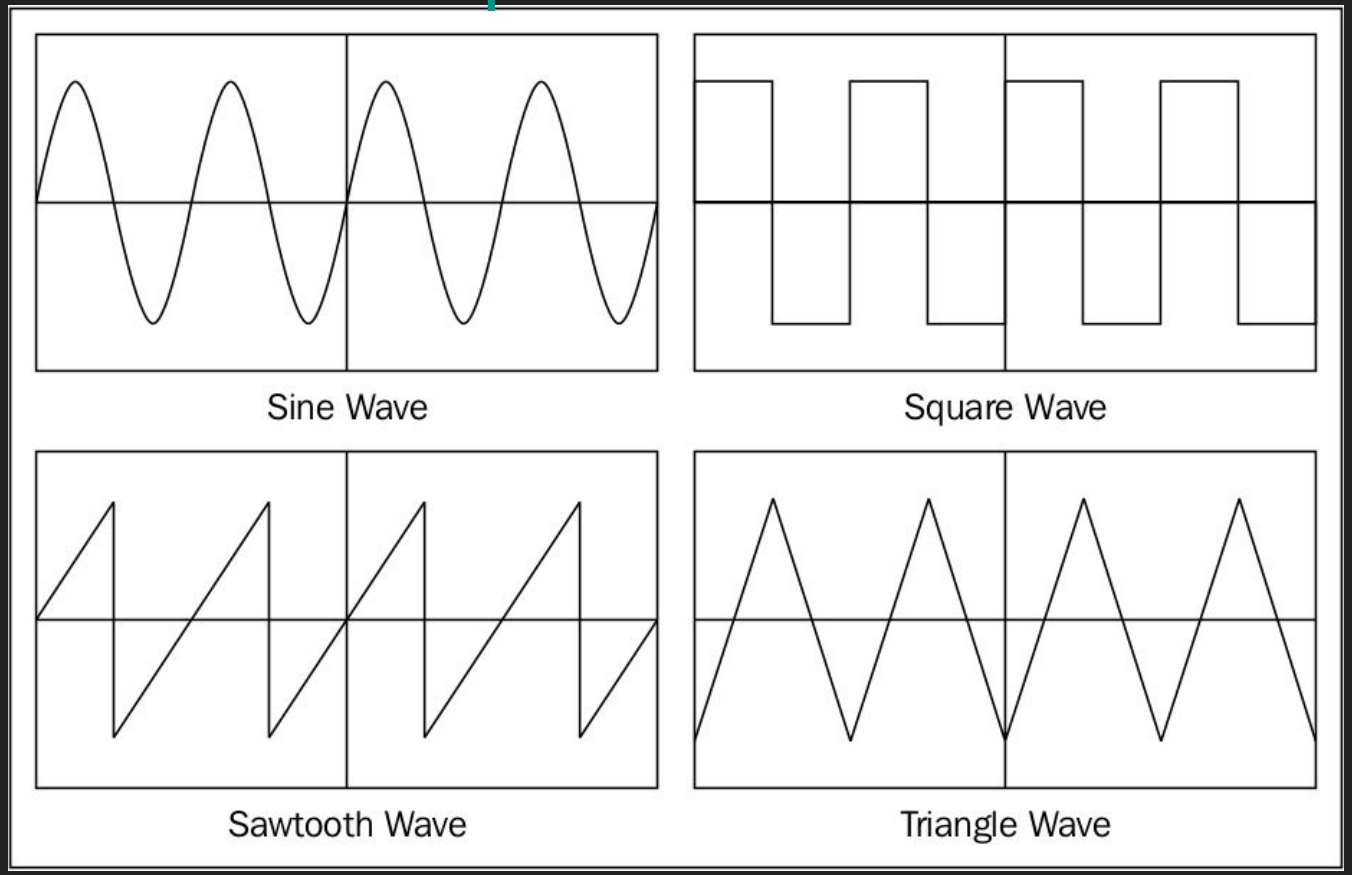


Gouveia, David. Getting Started with C++ Audio Programming for Game Development

2. Áudio

- ➔ Sintetizadores
  - Tipos de ondas
  - Instrumentos: sintetização aditiva, *subtractive etc.*
  - Modulação: FM, AM etc.

Tipos de Onda



Gouveia, David. Getting Started with C++ Audio Programming for Game Development

3. Formatos de Áudio

3. Formatos de Áudio

- ➔ Introdução
  - ◆ Diferentes algoritmos de compressão
  - ◆ Codecs - codificadores e decodificadores
    - Compressão ⇔ PCM
  - ◆ SFX (*sound effects*) geralmente já estão em PCM
  - ◆ *Streaming* é usado em músicas

3. Formatos de Áudio

- Compressão
  - Sem compressão
    - RAW, WAVE
  - Compressão sem perda (*lossless*)
    - FLAC, ZIP
  - Compressão com perda (*lossy*)
    - MP3, OGG

3. Formatos de Áudio

- Música sequencial
  - “Como” e não “O quê”
  - Partituras e MIDI
  - Eventos MIDI

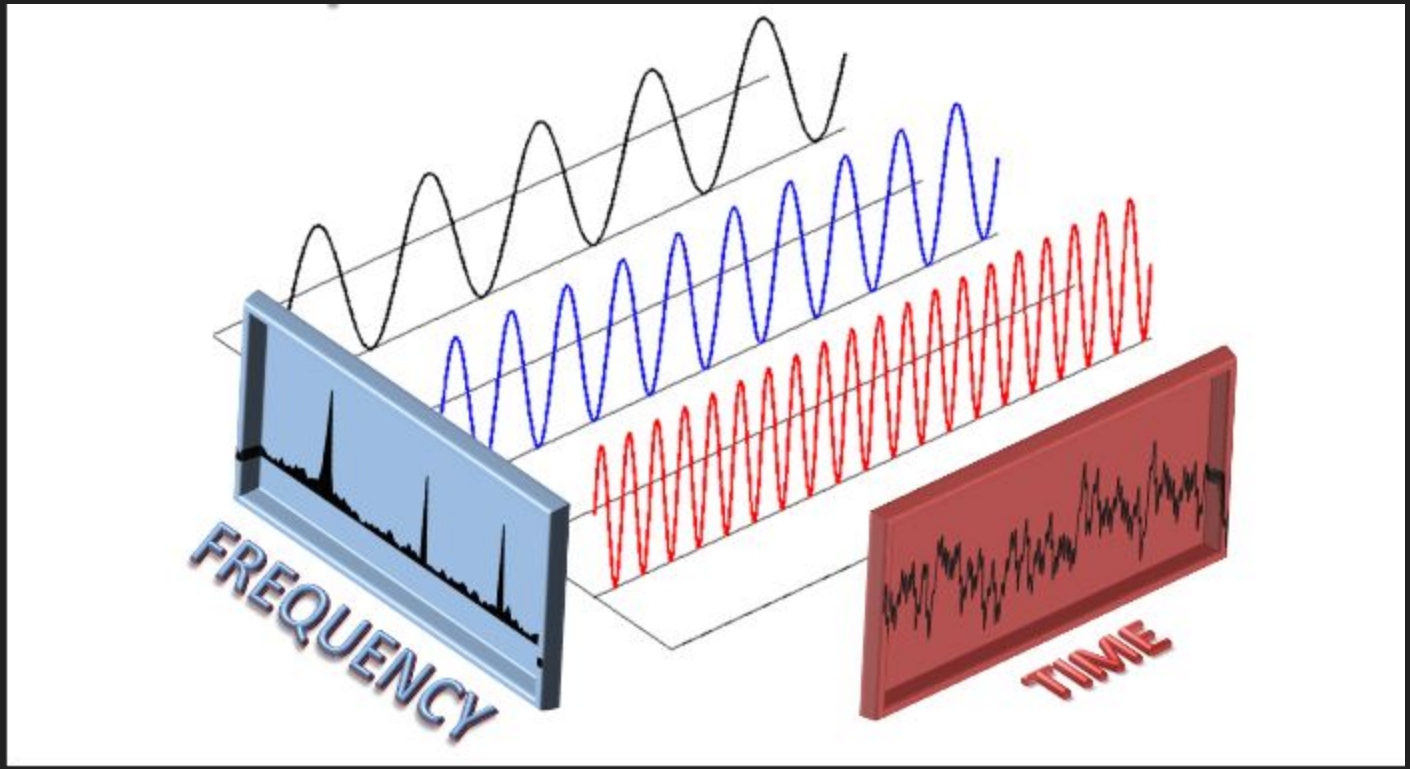
4. Efeitos sonoros

4. Efeitos sonoros

- Introdução
  - *Digital Signal Processing*
    - Algoritmos que passam pelo PCM
    - Podem ser transformadas

4. Efeitos sonoros

- ➔ Transformada de Fourier
  - ◆ Domínio de tempo ⇔ frequências
  - ◆ Resultado complexo (frequência, fase)
- ➔ DSP e FT podem ser pesadas
  - ◆ FT: n^2; FFT: n*log(n)
  - ◆ DSP: n, n^2 etc.


FREQUENCY
TIME

4. Efeitos sonoros

- FMOD
  - Normalização: normaliza -1 a 1
  - Compressor: comprime partes com volume alto
  - Distorção: distorce o som (similar ao *clipping*)
  - *Low-pass filter:* atenua frequências altas (abafa)
  - *High-pass filter:* atenua frequências baixas (fino)

4. Efeitos sonoros

- FMOD
  - *Parametric EQ:* equalizador com diversos parâmetros
  - *Delay:* delay curto (não perceptível)
  - *Echo:* delay longo (perceptível)
  - *Flanger:* modulador de delay
  - *Chorus:* modulador de delay e *pitch*

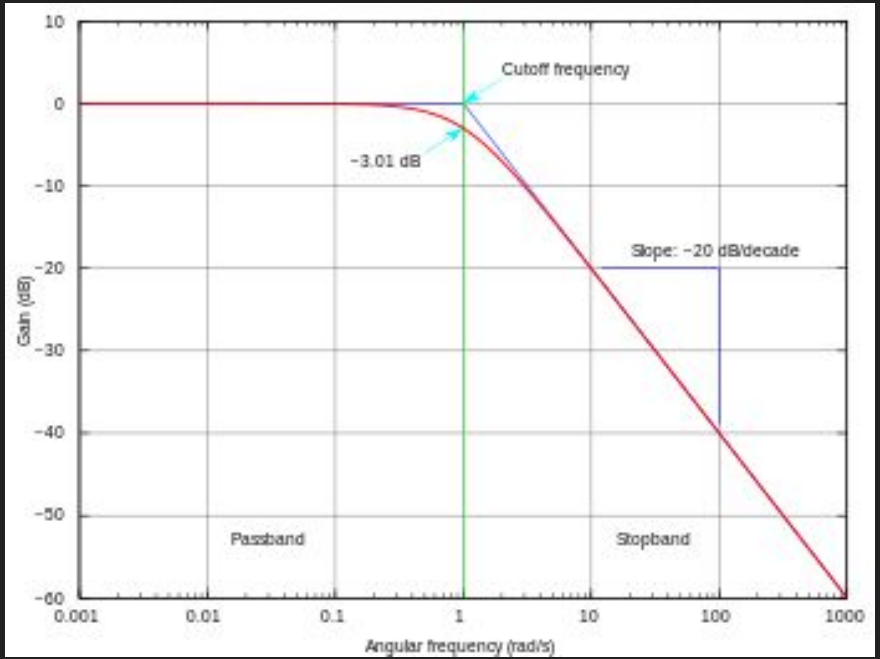
4. Efeitos sonoros

- FMOD
  - *Pitch shift:* altera a frequência sem alterar o tempo
  - Remoção de ruído: remove ruídos (*Noise gate*)

4. Efeitos sonoros

- ➔ *Low-pass filter*
  - Simula perda de energia
    - Som passa por um objeto ou reflete
  - Bom para obstrução e oclusão
  - Água

Low-pass Filter



Low-pass Filter

$$y_i = \alpha x_i + (1 - \alpha) y_{i-1}$$

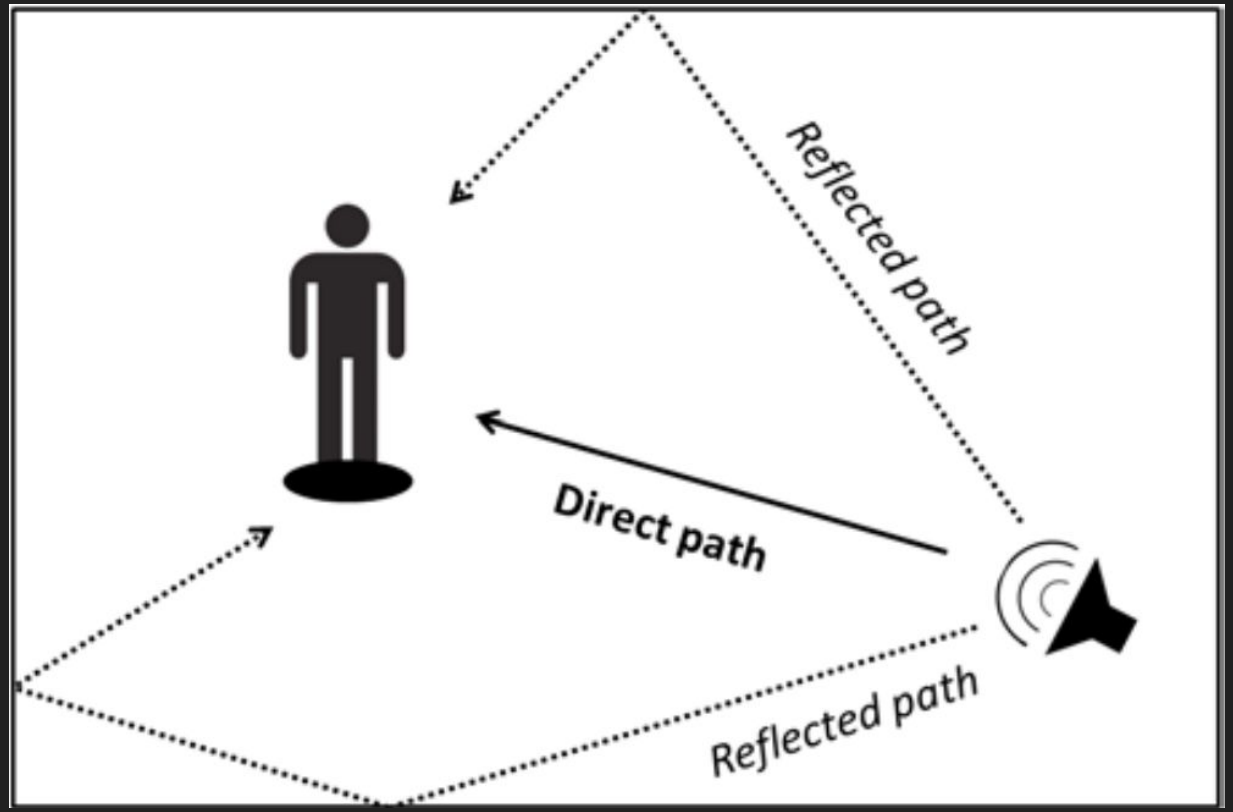
Low-pass Filter

$$\alpha = \frac{2\pi\Delta_T f_c}{2\pi\Delta_T f_c + 1}$$

4. Efeitos sonoros

- Reverberação
  - Importante para simular ambiente
  - Duas ou três *delay lines*
    - Filtros (LPF)
  - Tamanho do *delay* é definido pelo tamanho do espaço

Reverberação

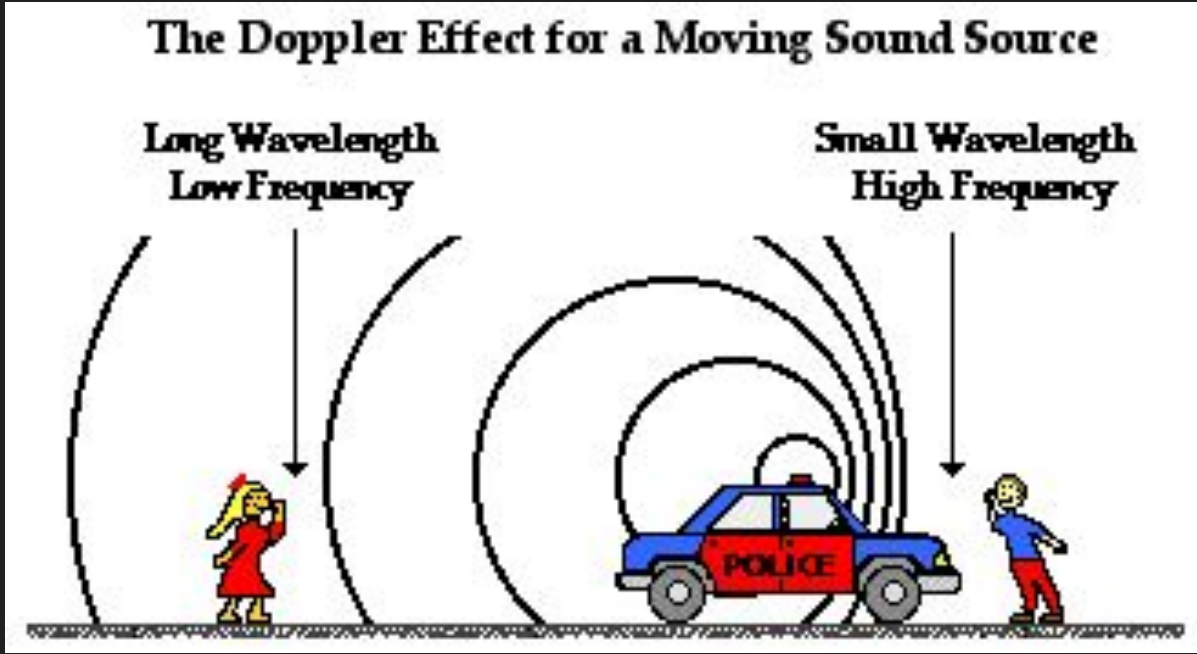


Gouveia, David. Getting Started with C++ Audio Programming for Game Development

4. Efeitos sonoros

- Efeito Doppler
  - Diferença de *pitch* quando objetos estão que estão em movimento

Efeito Doppler



5. Imagem

5. Imagem

- Representação
  - *Pixel*
  - Cor
    - RGBA /ARGB
    - CMKY
    - YCbCr

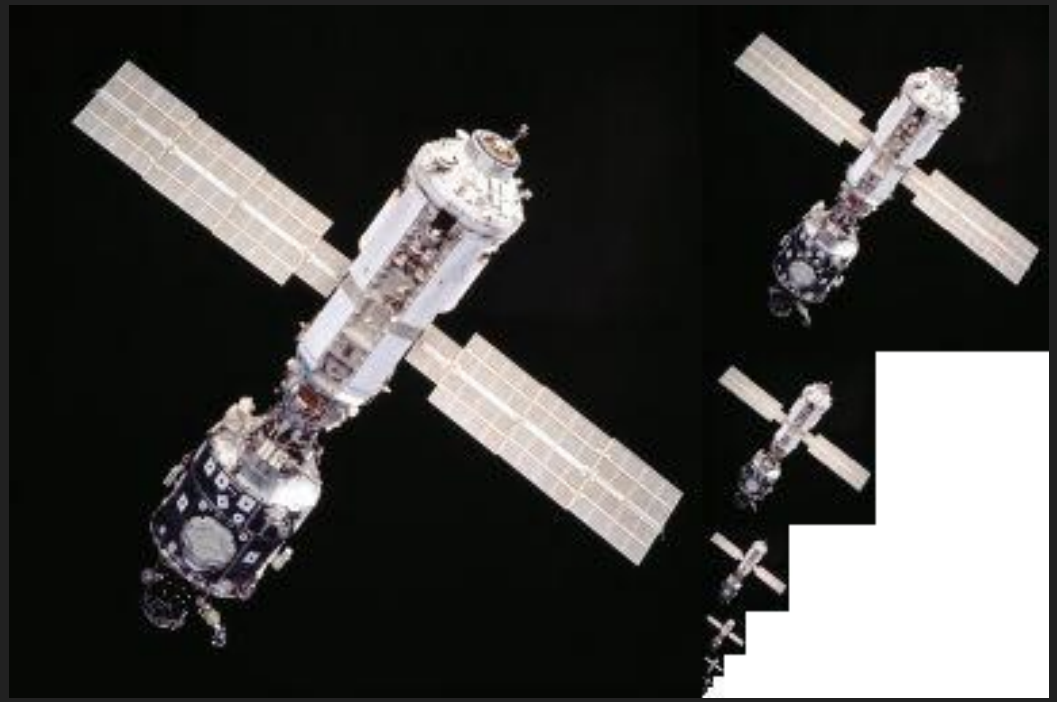
5. Imagem

- ➔ *Streaming*
  - Comum em textura pela *web*
  - Mais fácil carregar e passar para placa
  - Texturas
    - Resoluções: 512x512, 1024x1024

5. Imagem

- ➔ *Mipmap*
  - Versões de imagem (tamanhos de diferentes)
  - Podem ser criadas em *runtime* (potência de 2)

Mipmap



6. Formatos de Imagem

6. Formatos de Imagem

- Compressão
  - Sem compressão
    - Bitmap
  - Compressão sem perda
    - PNG, TIFF
  - Compressão com perda
    - JPG

Compressão JPG

Dúvidas?

Referências

Referências

[1] Game Coding Complete, Fourth Edition (2012) - Mike McShaffry, David Graham
[2] Gouveia, David. Getting Started with C++ Audio Programming for Game Development
[3] https://en.wikipedia.org/wiki/Nyquist-Shannon_sampling_theorem
[4] https://en.wikipedia.org/wiki/Fourier_transform
[5] http://groups.csail.mit.edu/netmit/wordpress/projects/sparse-fourier-transform/
[6] https://en.wikipedia.org/wiki/Doppler_effect
[7] http://www.physicsclassroom.com/class/waves/Lesson-3/The-Doppler-Effect
[8] https://en.wikipedia.org/wiki/Mipmap
[9] https://en.wikipedia.org/wiki/JPEG