UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
CURSO DE CIÊNCIA DA COMPUTAÇÃO

DESENVOLVIMENTO DE UM PROTÓTIPO DE SISTEMA DE ÁUDIO PARA UM SIMULADOR DE MOTOCICLETA

TRABALHO DE GRADUAÇÃO

Vanderlan Dupont de Oliveira

Santa Maria, RS, Brasil

2014

DESENVOLVIMENTO DE UM PROTÓTIPO DE SISTEMA DE ÁUDIO PARA UM SIMULADOR DE MOTOCICLETA

Vanderlan Dupont de Oliveira

Trabalho de Graduação apresentado ao Curso de Ciência da Computação da Universidade Federal de Santa Maria (UFSM, RS), como requisito parcial para a obtenção do grau de

Bacharel em Ciência da Computação

Orientador: Prof. Dr. Raul Ceretta Nunes

Co-orientador: Prof. Dr. Cesar Tadeu Pozzer

Trabalho Nº: 387
Santa Maria, RS, Brasil

2014

Universidade Federal de Santa Maria
Centro de Tecnologia
Curso de Ciência da Computação

A Comissão Examinadora, abaixo assinada,
aprova o Trabalho de Graduação

DESENVOLVIMENTO DE UM PROTÓTIPO DE SISTEMA DE ÁUDIO PARA UM SIMULADOR DE MOTOCICLETA

elaborado por
Vanderlan Dupont de Oliveira

como requisito parcial para obtenção do grau de
Bacharel em Ciência da Computação

COMISSÃO EXAMINADORA:

Raul Ceretta Nunes, Prof. Dr.
(Presidente/Orientador)

Ana Trindade Winck, Profª. Drª. (UFSM)

Giliane Bernardi, Profª. Drª. (UFSM)

Santa Maria, 04 de Dezembro de 2014.

AGRADECIMENTOS

Agradeço primeiramente aos meus pais, Vasconcelos e Sueli pelo apoio, a oportunidade e o incentivo que me deram durante toda a vida, e a minha irmã Maura pelo companheirismo de sempre.

A minha avó Margarida, que hoje infelizmente já não está mais entre nós, pela pessoa maravilhosa que era, a qual eu tive a felicidade e o prazer de ter como avó e conviver durante muitos anos. Pelo incentivo que me deu durante a vida e por sempre acreditar que eu pudesse alcançar meus objetivos. Saudades, vó! :(

Agradeço aos meus bons e velhos amigos de Alegria-RS, que apesar da distância sempre me apoiaram e me deram a maior força durante todo esse tempo, desde a época de ensino médio, em especial ao Buza, Jefe e Laura.

Aos meus grandes colegas e amigos, que o curso de Ciência da Computação na UFSM me deu a oportunidade e o prazer de conhecer, principalmente ao Tiago, Ricardo, Maia, Giane, Liza e Jéssica, os quais a amizade e parceria levarei pra toda a vida.

Ao professor Raul, e ao professor Pozzer por todo ensino e suporte desde as disciplinas ministradas até a orientação durante o desenvolvimento deste trabalho, pessoas com quem eu posso dizer que tive o prazer de trabalhar.

À professora Andrea, por todo suporte prestado tanto no papel de professora como no de coordenadora do curso de Ciência da Computação.

E aos demais professores, colegas e amigos, que de uma forma ou outra possibilitaram essa conquista. Obrigado! :)

RESUMO

Trabalho de Graduação
Curso de Ciência da Computação
Universidade Federal de Santa Maria

DESENVOLVIMENTO DE UM PROTÓTIPO DE SISTEMA DE ÁUDIO PARA UM SIMULADOR DE MOTOCICLETA

AUTOR: VANDERLAN DUPONT DE OLIVEIRA
ORIENTADOR: RAUL CERETTA NUNES
CO-ORIENTADOR: CESAR TADEU POZZER
Local da Defesa e Data: Santa Maria, 04 de Dezembro de 2014.

Os simuladores de direção veícular estão se tornando cada vez mais presentes no treinamento e formação de condutores, mas ainda são bastante limitados na representação do mundo real. Esta ferramenta proporciona um treinamento mais direcionado com a vantagem de não ao usuário em risco quanto a possíveis acidentes. Para se caracterizar um simulador, e poder ser utilizado na formação de condutores, é necessário que esse, em âmbito geral, represente o máximo possível a realidade. A sincronização dos sons com as imagens do cenário reproduzidas na tela de visualização é um aspecto importante para a simulação de automóveis e/ou motocicletas, proporcionando ao usuário uma maior sensação de realismo ao interagir com o simulador. Alcançar o sincronismo desejado é uma tarefa desafiadora e este trabalho explora os aspectos de manipulação e sincronização, em tempo real, dos sons, imagens e eventos que ocorrem num protótipo de simulador de motocicleta. Pela complexidade do trabalho, faz-se necessário a utilização de uma engine de jogos, a Unity 3D, a qual fornece diversas ferramentas de apoio para o desenvolvimento de jogos e/ou simuladores, e o áudio é um aspecto que pode ser utilizado e manipulado. Este trabalho implementa um protótipo de sistema de áudio, com foco na sincronização entre áudio, imagem e eventos que ocorrem em um ambiente virtual de simulação, utilizando efeitos e técnicas de áudio tais como Áudio 3D e Surround Sound.

Palavras-chave: Simulação. Motocicleta. Unity3D. Sincronização. Áudio.

ABSTRACT

Undergraduate Final Work
Undergraduate Program in Computer Science
Federal University of Santa Maria

DEVELOPMENT OF AN AUDIO SYSTEM PROTOTYPE FOR A MOTORCYCLE SIMULATOR

AUTHOR: VANDERLAN DUPONT DE OLIVEIRA
ADVISOR: RAUL CERETTA NUNES
COADVISOR: CESAR TADEU POZZER
Defense Place and Date: Santa Maria, December 04th, 2014.

The vehicular driving simulators are becoming increasingly present in the education and training of drivers, but are still very limited in the real world representation. This tool provides a more targeted training with the advantage of not the user at risk for possible accidents. To characterize a simulator, and can be used in driver training, it is necessary that, at the general level, representing as much as possible reality. The synchronization of sounds with images of the scene reproduced in the preview screen is an important aspect for the simulation of cars and/or motorcycles, providing the user with a greater sense of realism that interacts with the simulator. Achieve the desired timing is a challenging task and this work explores aspects of handling and synchronization, real-time, sounds, images and events that occur in a motorcycle simulator prototype. By the complexity, it is necessary to use a game engine, the Unity 3D, which provides various support tools for game development and/or simulators, and audio is an aspect that can be used and manipulated. This work implements an audio system prototype, focusing on synchronization between audio, images and events that occur in a virtual environment simulation using effects and audio techniques such as 3D Sound and Surround Sound.

Keywords: Simulation. Motorcycle. Unity3D. Synchronization. Audio.

LISTA DE FIGURAS

SUMÁRIO

1 INTRODUÇÃO

A evolução das tecnologias voltadas à produção de jogos 3D vem acontecendo cada vez mais rápido. O uso de novas tecnologias, como no caso de simuladores de direção, é uma realidade no mundo todo, e o Brasil vem se destacando na melhoria contínua dos processos de formação de condutores (Rota Simuladores, 2014). De acordo com a empresa Rota Simuladores (2014) acredita-se que o uso desta tecnologia contribuirá para capacitar estas pessoas, auxiliando na redução dos acidentes e na qualificação de quem está dirigindo nas ruas e vias.

Os simuladores ainda são bastante limitados na representação do mundo real, mas eles são considerados importantes na formação de condutores por proporcionarem opções de reações quando ocorrem condições adversas. Contudo, a tendência é melhorar significativamente a representação da realidade pelos simuladores, mas isso necessita muito investimento e testes nesta área.

O desenvolvimento de simuladores de direção, para que sejam utilizados por pessoas que estão passando por treinamento, é de fundamental importância, pois a simulação de direção é um passo importante na adaptação dos usuários a situações cotidianas ou de perigo que ocorrem na vida real ao se conduzir um veículo. O objetivo da construção de simuladores, em especial os de motocicleta, está em favorecer o usuário na capacitação e adaptação às situações de risco em situações reais. O desafio que surge no contexto da implementação, é fazer com que o usuário se sinta imerso no ambiente virtual, obtendo a sensação de estar dirigindo uma motocicleta real.

Este trabalho está relacionado à construção de um protótipo de simulação de direção voltado para a categoria A, o de motocicleta, onde os sons emitidos possam representar algumas situações reais que acontecem quando se conduz de uma motocicleta. A imagem mostrada na tela do simulador deve estar devidamente sincronizada com o som, ou seja, o som deve ser executado a partir de eventos/estímulos, que devem também corresponder à imagem mostrada. Deste modo, o trabalho tem seu foco, não na construção de um simulador de motocicleta em si mas, nos mecanismos de sincronização entre imagem, eventos e sons reproduzidos pelo protótipo de simulação. Conforme há alterações na imagem e a ocorrência de eventos, no ambiente virtual e/ou no ambiente de interação homem-máquina (estímulos gerados pelo usuário), tal como aceleração, troca de marchas, frenagens, dentre outros, os sons produzidos/reproduzidos deverão estar sincronizados. Para isso será feita a simulação através da utilização de sons tridimensionais (Áudio 3D) e a técnica de *Surround Sound*, que envolvem o usuário, e ajudam a

melhor simular os sons produzidos por uma motocicleta real.

A sincronização dos sons com a imagem reproduzida na tela de visualização é um aspecto importante para a simulação de automóveis e/ou motocicletas, dando maior sensação de realismo quando o usuário interage com o simulador. Neste sentido, alcançar o sincronismo desejado é uma tarefa desafiadora e que exige conhecimentos de programação de jogos, sistemas distribuídos e de interface gráfica. Este trabalho visa explorar estes aspectos, manipulando e sincronizando, em tempo real, os eventos e sons que ocorrem num protótipo de simulador de motocicleta. Quanto melhor for a sincronia entre o cenário e os sons reproduzidos, melhor o realismo e consequente auxílio no processo capacitação do usuário.

1.1 Objetivos

1.1.1 Objetivo Geral

O objetivo geral desse trabalho consiste em explorar a sincronização de Áudio 3D num protótipo de simulador de motocicleta. Através da utilização de uma *Engine* de Jogos, onde o áudio possa ser um aspecto a ser utilizado e manipulado, visa-se neste trabalho implementar um protótipo de sistema de áudio que melhor se adapte ao simulador e a sua arquitetura de áudio, sincronizando imagens e sons em tempo real.

1.1.2 Objetivo Específico

- Definir *Engine(s)* para o desenvolvimento do simulador e a manipulação de Áudio 3D, com foco na renderização de áudio.
- Mapear cenário(s) de simulação para treinamento com vista à habilitação Categoria A (motocicleta), onde o áudio seja um elemento importante a ser testado.
- Criar um Ambiente Virtual de Simulação, ou seja, o cenário de treinamento onde será implementado o protótipo de simulação e feita a sincronização de áudio, utilizando a engine escolhida.
- Dado um case (cenário de treinamento), especificar um protótipo de sistema de Áudio 3D para o Simulador de Motocicleta.
- Implementar e testar a(s) funcionalidade(s) de sincronização do protótipo de simulação

com Áudio 3D.

1.2 Justificativa

A sincronização do áudio com a parte gráfica do simulador é fundamental para o aprendizado do usuário. Para isso, optou-se por desenvolver esse trabalho, o qual objetiva a imersão do usuário ao cenário de treinamento, através da simulação dos sons reproduzidos, com a utilização de Áudio 3D.

Este trabalho é parte de um projeto maior para simulação de motocicletas, com vistas à capacitação de motoristas, no qual também está sendo desenvolvido um mecanismo de sincronização entre o simulador e uma plataforma de movimentos (motion) (PAUL, 2014).

1.3 Organização do Trabalho

Este trabalho está estruturado da seguinte maneira: o Capítulo 2 apresenta uma revisão bibliográfica sobre simuladores de direção e as tecnologias envolvidas, especialmente a Unity3D, que é a *engine* utilizada no desenvolvimento do trabalho. Ainda, destaca algumas características sobre os sons e as ténicas de áudio utilizadas em simulação. O capítulo 3 descreve a proposta para a realização do trabalho, incluindo os requisitos para a sequência de implementação, a arquitetura da aplicação que será desenvolvida e a metodologia de desenvolvimento. O capítulo 4 trata dos aspectos de desenvolvimento do trabalho, detalhando cada etapa de implementação e do uso da engine Unity3D e seus componentes, desde a criação dos cenários de simulação, criação e movimentação da motocicleta até a parte de utilização e sincronização de áudio. O capítulo 5 relata os experimentos e os resultados obtidos no trabalho e, por fim, o capítulo 6 conclui o trabalho, retomando alguns pontos importantes e resultados alcançados, citando também o que pode ser feito como trabalhos futuros.

2 FUNDAMENTAÇÃO E REVISÃO BIBLIOGRÁFICA

Este capítulo apresenta definições e conceitos teóricos sobre a simulação, simuladores de direção, a importância dos sons (Áudio 3D e *Surround Sound*) na simulação, bem como uma visão geral sobre as ferramentas e componentes utilizados neste trabalho.

2.1 Simuladores

A partir da constante evolução dos jogos, surgiram também ferramentas que buscam a maior proximidade com a realidade, como os simuladores de veículos, por exemplo. A busca pelo realismo é uma das características que tem muita importância em um simulador, desde implementação e sincronização de vários elementos como movimentos, sons e cenário gráfico até a interação do usuário com a ferramenta. As características dos simuladores de direção são muito parecidas com a de um jogo 3D, pois o usuário interage com a ferramenta e a mesma trata os comandos conforme eles são recebidos (Ferrazin et al., 2001). A imersão do usuário no cenário que foi desenvolvido é considerado o principal aspecto em um simulador desse tipo.

2.1.1 Simuladores de Direção

De acordo com Stedmon et al. (2011), os simuladores devem oferecer uma abstração do mundo real, no qual os usuários podem experimentar características de um sistema real. Simuladores de direção podem criar situações que permitem realizar um treinamento e oferecer maior segurança na capacitação de motoristas. Pereira (2009), diz que além de ser uma importante ferramenta para oferecer esse tipo de treinamento, os simuladores podem trabalhar muito próximo da realidade. Incidentes ou situações podem ser simulados e a maneira de agir do usuário poderá ser monitorada, a fim de avaliar se o resultado foi o esperado ou não. Desta forma, o usuário do simulador poderá adquirir experiência, conhecimento e habilidades, e poderá ser capaz de tratar alguma situação de difícil reprodução na vida real.

Simuladores de direção oferecem muitas vantagens para a educação no trânsito. Segundo Fuller (2008), eles oferecem uma exposição mais rápida a uma ampla variedade de situações, melhorando possibilidades de respostas do veículo sob diferentes perspectivas, possibilita a repetição ilimitada de momentos educativos, avaliações objetivas, demonstração de manobras e prática segura dentro do ambiente virtual de simulação.

2.1.2 Simulador de Motocicleta

No simulador de motocicleta, o piloto deve experimentar as mesmas sensações sonoras da condução ou operação de uma motocicleta real. Um dos desafios da implementação de um simulador é sincronizar adequadamente os sons com o que aparece na tela e o que é percebido e sentido pelos movimentos, acelerações e desacelerações. São considerados os movimentos de controle do veículo, ou as interações físicas que surgem com a estrutura mecânica de um simulador, que devem estar em sincronia com os sons. Um exemplo de Simulador de Motocicleta pode ser visto na Figura 2.1, onde o simulador Lander é composto de uma moto, uma plataforma de movimentos e três telas para visualização do cenário, obtendo um alto grau de imersão, onde as sensações percebidas pelo motorista (som, imagem, vibrações, movimentos, etc.) são como na realidade (LANDER, 2014).

Figura 2.1 – Lander Simulator (Lander, 2014).

2.2 Engines

Segundo Clua (2005), uma *engine* é o principal elemento para a criação e o funcionamento de um jogo ou simulador. Ela abstrai questões de baixo nível em programação para quem está desenvolvendo um simulador, auxiliando o programador a desenvolver um trabalho complexo de alto nível, em um período menor de tempo, ou seja, uma *engine* pode ser integrada com o hardware gráfico, pode controlar os modelos para serem renderizados, trata dos sons que serão produzidos e executados, das entradas de dados do usuário, de todo o processamento de baixo nível e outros aspectos que o desenvolvedor normalmente não pode despender muito tempo.

Normalmente, segundo Zerbst et al. (2004), uma *engine* é composta por diversas ferramentas, tais como módulos de visualização, módulos de física, módulo de áudio e de inteligência artificial, por exemplo, cada um responsável por alguma etapa do processo de criação de um jogo. Pode-se afirmar que uma *engine* na realidade é composta por diversas *"subengines"*, sendo cada um responsável por tratar um tipo de problema envolvido em jogos (CLUA, 2005). Algumas delas tem papel importantíssimo na construção de jogos/simuladores. A seguir são apresentados mais detalhes sobre duas delas, a de Física e a de Som.

- ***Engine* de Física:** Grande parte da interatividade de um jogo ou simulador se deve ao funcionamento de algumas leis da física sobre o mundo virtual criado. Assim, segundo Clua (2005), alguns cálculos básicos de física podem ser feitos num jogo, como: (i) Simulação de colisão, onde objetos 3D podem colidir com outros e (ii) aplicação de forças, onde os objetos podem se movimentar no mundo virtual devido a aplicação de diversas forças sobre o mesmo. Ao acelerar a motocicleta, sua velocidade deverá ir crescendo gradualmente e não abruptamente, por exemplo.

- ***Engine* de Som:** Este componente de áudio permite o controle sobre os arquivos de áudio da biblioteca de recursos (*Assets*) do jogo. Normalmente, utiliza-se scripts para manipular este tipo de arquivo na *engine*. Além de permitir abrir e tocar arquivos de áudio de diferentes formatos, tais como .mp3, .wav, etc, estas *engines* permitem um controle de som posicional, permitindo que objetos da cena emitam sons e estes se comportem conforme o posicionamento do objeto na cena e o posicionamento da câmera (CLUA, 2005).

2.3 Unity3D

A *engine* gráfica Unity3D é uma ferramenta de desenvolvimento projetado para a criação de jogos 3D para as mais diferentes plataformas: Mac OS, iPhone, Android, Play Station, etc. Ela é uma *engine* completa, pois atende todos os critérios necessários (como os vistos na seção 2.2) para a contrução de um jogo ou simulador. A interface permite que sejam controlados e gerenciados todos os elementos e aspectos do jogo, como: efeitos de luz, texturas, simulações físicas, áudio 3D e até a implementação de rede para jogos *multiplayer* onde, a partir de uma interface gráfica bastante intuitiva, pode-se visualizar a animação em tempo real. Também tem um sistema de scripts que permite que se construa uma lógica ligada a todos os aspectos do

jogo, podendo controlar tanto objetos da cena quanto o áudio inseridos no jogo.

A Unity3D é baseada em três linguagens de programação: JavaScript, C# e Boo (uma variação do Python), que também são executadas na plataforma de desenvolvimento (PROCACCINI, 2013). A interface de desenvolvimento da Unity é composta de cinco janelas com guias: *Hierarchy, Inspector, Game, Scene e Project* (Ashaolu, 2012), como pode ser visto na Figura 2.2.

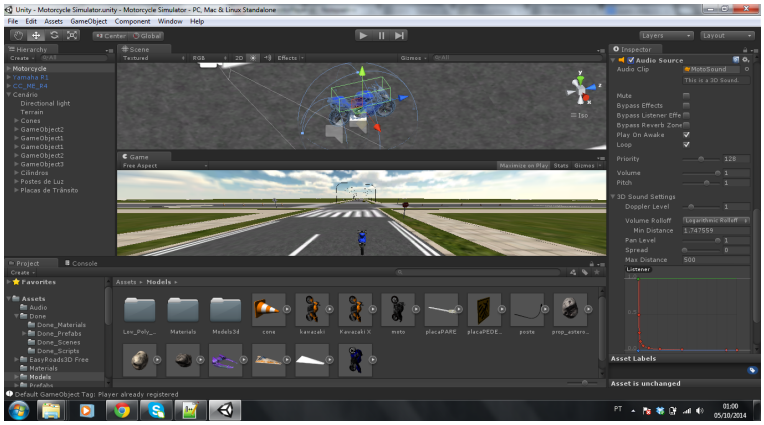

Figura 2.2 – Screenshot da Interface de Desenvolvimento da Unity3D.

2.3.1 Componentes Unity

A *engine* de jogos Unity3D é composta por vários subsistemas que permitem que sejam gerenciados todos os aspectos do jogo. Todo o conceito é baseado no uso de *"GameObjects"*, que podem ser facilmente manipulados no interior da *engine*, a partir de um simples "arrastar e soltar"de *Assets* (Objetos ou Modelos em 3D) no ambiente virtual. Ela fornece uma série de componentes que auxiliam no desenvolvimento de um jogo, ou simulador, como é o caso. Cada objeto criado na Unity pode possuir diversas propriedades que são utilizadas para simular as características que deseja-se que esses objetos tenham. Essas características são apresentadas através da classe *Component*, a qual contém todas as referências para qualquer propriedade que esteja ligada a um *GameObject*.

Dentre as principais e mais importantes destacam-se os componentes gráficos, que auxiliam na criação dos cenários. Os componentes de física, que tornam objetos 3D mais parecidos com os reais. E os componentes de áudio, que permitem a adição de sons e/ou efeitos sonoros dentro do ambiente de simulação. Tudo isso pode ser controlado pelo uso de scripts. Esses componentes são abordados nos itens a seguir.

- **Gráficos:** É o lado visual da Unity, o que inclui câmeras e iluminação. Ela possui diversos recursos gráficos que permitem a criação dos mais diversos e elaborados cenários gráficos. É o elemento chave que propicia ao programador as ferramentas necessárias para o desenvolvimento de jogos simples até os mais realistas, sejam eles de 2 ou 3 dimensões (UNITY, 2014). Isso é fundamental quando se fala na criação de um simulador de direção, pois é crucial que a representação gráfica dos cenários e os componentes do mesmo sejam a mais realista possível.

- **Física:** Como visto em Hu Wenfeng (2012), cada *GameObject* tem um comportamento diferente, de acordo com as leis da física, graças à *Engine* de Física na Unity3D. O componente *"rigidbody"* representa o pai de todos os outros componentes físicos que estão conectados. É possível simular colisões entre objetos (*Colliders*), adicionar forças (*Constant Force*), etc. Através do uso e manipulação desses componentes conectados à *rigidbody* pode-se simular qualquer efeito físico sobre os objetos no ambiente virtual.

- **Áudio:** Um jogo ou simulador seria incompleto sem algum tipo de áudio, seja ele música de fundo ou efeitos sonoros. O Sistema de áudio da Unity é flexível e poderoso, capaz de importar arquivos de áudios de formatos de padrão da Unity, tais como .wav, .mp3, etc e manipulá-los da forma que se deseja. Pode-se, opcionalmente, aplicar efeitos como eco, por exemplo, através da utilização de sons 3D, pois a Unity possui recursos sofisticados para a reprodução de sons no espaço tridimensional (UNITY, 2014).

- **Scripts:** Os *Scripts* são essenciais em todos os jogos. Mesmo o jogo mais simples precisa de scripts para responder à entrada do jogador e mandar comandos para o jogo e os eventos acontecerem no tempo certo. Além disso, os scripts podem ser usados para criar efeitos gráficos, controlar o comportamento físico de objetos, manipular e aplicar efeitos sonoros e até mesmo implementar um sistema de inteligência artificial personalizado para personagens ou objetos do jogo (UNITY, 3D).

2.4 Sons

Em simulação, a sintonia correta dos sons reproduzidos é fundamental e deve-se explorar técnicas e/ou ferramentas para ajustá-las corretamente. Para Cossalter et al. (2010), a simulação de uma motocicleta pode ser realizada dando atenção especial aos aspectos de:

- percepção da velocidade;
- sensação de frenagem e o sentimento ao andar em estradas com obstáculos;
- capacidade de resposta do veículo durante mudanças de marcha;
- experiência de andar em baixa velocidade, aceleração e desaceleração, etc.

Um aspecto tecnológico específico, que deve ser destacado na simulação e na produção de simulador, é o áudio, ou seja, a sincronização ddo som com a imagem reproduzida. Segundo Clua (2005), para incrementar a imersividade no simulador é fundamental adicionar a percepção sonora. O som 3D dá maior sensação de imersão sonora e traz uma noção de profundidade e maior sensação de realismo. Este é tratado num processo de "renderização"de áudio, ou seja, a produção de áudio em tempo de execução.

2.4.1 Técnicas de Áudio

Segundo Tang (2014), Áudio 3D e *Surround Sound* são duas técnicas diferentes de uso do som, mas que muitas vezes são confundidas. As diferenças entre elas podem ser destacadas a seguir:

2.4.1.1 Áudio 3D

Áudio 3D é muitas vezes usado em jogos ou outras aplicações interativas e ajuda a criar um ambiente mais realista. As tecnologias de áudio 3D foram criadas para simular os sons que podem ser ouvidos na vida real. Comparado com áudio *surround*, o áudio 3D é um sistema de áudio mais complexo e mais imersivo. Áudio 3D é usado para criar uma projeção mais realista do ambiente para o ouvinte. Muito parecido com o áudio *surround*, o ouvinte é capaz de dizer de qual direção o som está vindo. Além da direção, o ouvinte pode compreender a distância de onde o som vem, muito parecido com o que podemos ouvir realmente. Em resumo, as características do Áudio 3D são (THORN, 2013):

- Volume e Atenuação: Dá ao ouvinte uma ideia da distância.
- Movimento e Direção: Dá ao ouvinte uma ideia da direção.
- Reverberação e Acústica: Dá ao ouvinte uma ideia de ambiente imersivo.

2.4.1.2 *Surround Sound*

O sistema de áudio *surround* é muitas vezes usado para entretenimento em casa ou em cinemas. Segundo o estudo de Tang (2014) a ideia por trás do sistema surround é fazer o ouvinte compreender a orientação básica do som para melhorar a experiência imersiva. Ou seja, ouvir de que parte do cenário o som está sendo reproduzido, como por exemplo, a partir da parte traseira, da parte dianteira ou de um dos lados. O ouvinte não pode realmente entender o quão longe ou quão perto o som está sendo reproduzido.

Tang (2014) ainda cita que dependendo da situação, os dois tipos de sistemas de áudio apresentados anteriormente são usados de forma diferente. Em um filme, na TV ou outras formas de entretenimento estática, onde a imersão é limitada, o sistema de áudio surround é mais utilizado. O motivo é porque não há necessidade do espectador ou o ouvinte interagir com o que é apresentado. No entretenimento dinâmico no entanto, como é o caso de jogos, simuladores ou realidade virtual, onde imersão e interatividade desempenham um grande papel, deve-se pensar em utilizar um sistema com áudio 3D.

O áudio 3D tem melhor capacidade de simular o áudio que podemos ouvir no dia-a-dia, tornando a ilusão de realidade muito mais próxima da real do que somente utilizando um sistema *surround*, contudo, essas duas técnicas combinadas fornecem um melhor desempenho quando trata-se de imersão sonora.

A Figura 2.3 mostra as diferenças entre essas técnicas. Na esquerda, é ilustrado um sistema *surround* onde, utilizando apenas caixas de som com áudio normal (ou Áudio 2D), o ouvinte consegue apenas ouvir de qual direção o som está sendo emitido. Na imagem da direita, com o sistema utilizando Áudio 3D o usuário pode determinar com maior precisão a "posição do som", ou seja, a que distância ele está sendo emitido.

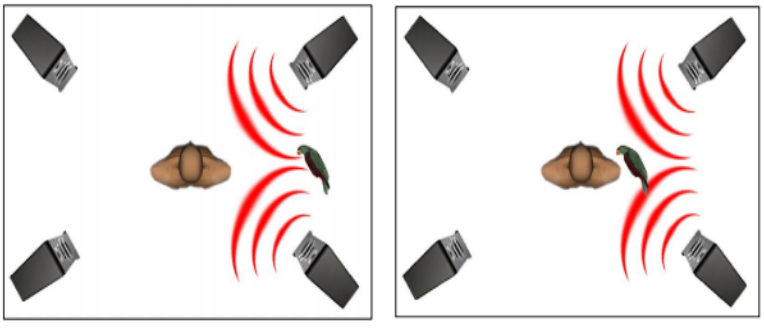

Figura 2.3 – Sistema Surround e Áudio 3D (Tang, 2014).

Ainda, segundo Benyon (2011) o sistema visual e auditivo desenvolveram-se para trabalhar juntos. A visão proporciona uma imagem do mundo estreita, voltada para a frente e rica em detalhes, enquanto a audição nos proporciona informação que está em toda a nossa volta. Por exemplo, o som de um carro se aproximando atrás de nós, faz com que nos viremos para olhar. Tanto o som quanto a visão permitem nos orientarmos no mundo (BENYON, 2011).

Essas duas técnicas são utilizadas no projeto de sincronização apresentado por este trabalho, através da *engine* de jogos Unity3D, o qual é desenvolvido para justamente sincronizar os eventos de áudio e imagem, e simular a imersão do usuário no ambiente virtual criado para a simulação, dando ao usuário também a possibilidade da percepção sonora, como já citado anteriormente.

3 PROPOSTA DE SINCRONIZAÇÃO DOS SONS

Conforme a introdução e justificativas apresentadas no capítulo 1 e as fundamentações apresentadas no capítulo 2, este trabalho propõe a utilização e sincronização de sons em um ambiente de simulação. A abordagem proposta envolve o uso de uma *engine* de jogos (a Unity3D) para o desenvolvimento do protótipo de simulação e de sincronização dos sons. Diferentes técnicas e algoritmos de manipulação dos arquivos de áudio, através de scripts, serão definidas e utilizadas neste trabalho.

3.1 Definição dos Requisitos

O objetivo principal do trabalho é sincronizar os eventos da motocicleta com o áudio reproduzido, como visto na seção 1.1. Para isso será utilizada a engine de jogos Unity3D, fundamental para a construção de cenários e uso de técnicas de áudio em ambientes de simulação. Os requisitos fundamentais para o desenvolvimento do trabalho, são justamente, a partir da *engine* Unity3D, utilizar e manipular arquivos de Áudio 3D para obter o sincronismo desejado entre a imagem e os sons, tornando o aspecto sonoro condizente com a imagem visualizada no protótipo de simulação. A partir daí, alguns pontos são importantes a serem tratados, e o áudio pode ajudar de forma significativa quando se trata de imersão ao cenário de simulação, que devem ser implementados e demonstrados.

Em um simulador de motocicleta, diversos aspectos são fundamentais quando se trata do aprendizado e da imersão do usuário ao cenário de treinamento, através dos sons. No entanto, no escopo deste trabalho, são avaliados e tratados alguns aspectos sonoros resultantes de algumas ações básicas de pilotagem. O áudio 3D utilizado para isso deve ser manipulado de tal forma que alcance a sincronia desejada simulando estas ações, executadas durante a condução da motocicleta, onde o usuário pode percebê-las. Resumindo, o protótipo deve ser capaz de simular, através dos sons, as seguintes ações:

- Som de aceleração, desaceleração e troca de marchas;
- Som de frenagens e derrapagens;
- Som da Buzina;
- Simulação do som de ultrapassagens por veículos.

Para cada tipo de ação, utiliza-se um determinado tipo de de efeito de sonoro. Por isso, é inserido ao ambiente e a motocicleta os componentes *Audio Source*, que são descritos na seção 4.4.1, fundamentais para o acesso e o controle de áudio dentro do ambiente de simulação, criado através da Unity3D.

3.2 Ambiente de Execução do Protótipo de Simulação

O ambiente de execução, do protótipo de simulação de motocicleta, é composto por alguns módulos fundamentais para o seu funcionamento como: (i) uma *engine* de jogos responsável por todo o controle físico e de áudio da simulação, (ii) um dispositivo de entrada para o usuário controlar os movimentos da motocicleta, (iii) um componente para visualização do ambiente de simulação e (iv) caixas acústicas para a emissão dos sons. Na Figura 3.1 é representado esse ambiente da aplicação. O usuário interage a partir dos controles de um teclado e visualiza o ambiente virtual através de um monitor de computador. A interação do usuário com o teclado resulta no envio de dados para a engine de jogos, que por sua vez os processará para controlar os movimentos da motocicleta mostrados na tela de visualização e por consequência emitir os sons produzidos por cada ação. Na engine são definidos o cenário de simulação, a motocicleta, scripts de movimentação da motocicleta (*GameObject*) e de manipulação e controle de áudio (*AudioFile*). Resumindo, o funcionamento da aplicação se dá a partir da interação do usuário com a ferramenta, enviando comandos a partir do teclado para fazer a movimentação da motocicleta e consequentemente a reprodução dos sons correspondentes a cada tipo de ação realizada.

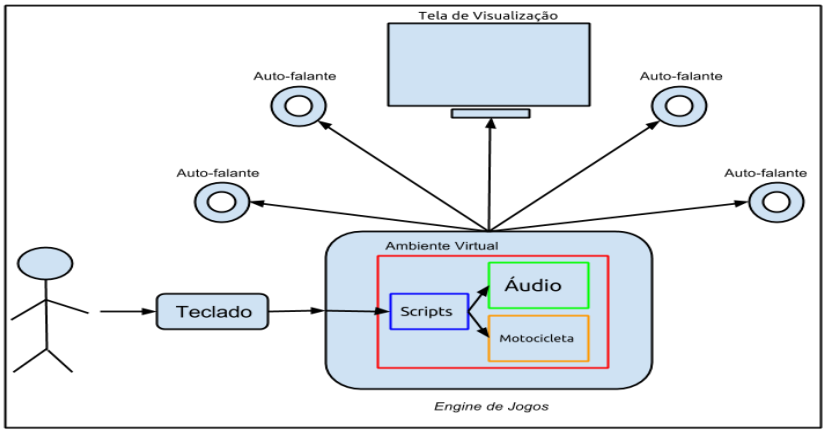

Figura 3.1 – Ambiente de Execução do Protótipo de Simulação

3.3 Metodologia de Implementação

A metodologia de implementação segue uma sequência de etapas que servem para atender aos requisitos propostos na seção 3.1. Inicialmente busca-se entender os conceitos fundamentais de programação de jogos e simulação, peças chave para o desenvolvimento do projeto. Ao dar início ao projeto, mas antes de adicionar e poder sincronizar o áudio com os eventos que ocorrerem no protótipo de simulação, são realizadas uma série de etapas de implementação, de acordo com os requisitos. Primeiramente é definido e criado um ambiente virtual onde o cenário de simulação representa alguns elementos encontrados no mundo real tais como ruas, estradas, quebra-molas, placas, etc. Feito isso, parte-se para a criação da motocicleta e simulação dos seus movimentos, essencial para quando aplicar os sons e sincronizá-los de acordo com cada movimento e ações que acontecerem no ambiente virtual.

Assim, após as funcionalidades de movimentação da motocicleta no protótipo de simulação estarem funcionando, inicia-se a utilização e manipulação dos sons a serem sincronizados. A implementação do protótipo de sincronização de áudio é desenvolvida utilizando arquivos de Áudio 3D, no formato .wav, e a técnica de *Surround*, a qual a engine possui suporte para teste. Por fim, são realizados os experimentos e o trabalho é concluído. Tudo o que foi citado anteriormente, ou seja, a sequência de implementação do trabalho e o cumprimento dos requisitos, por etapas, pode ser vista na Figura 3.2.

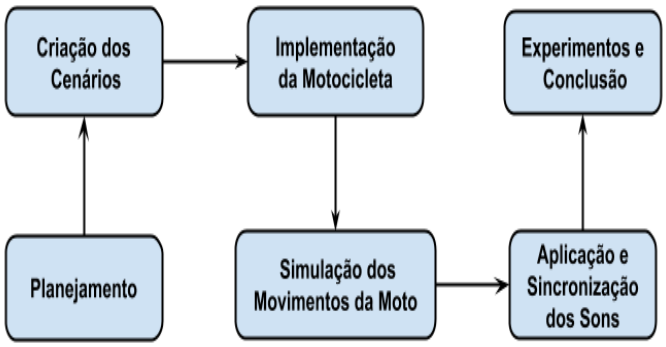

Figura 3.2 – Apresentação da Metodologia de Desenvolvimento

Para poder modelar o protótipo, foram utilizados alguns diagramas UML (Linguagem de Modelagem Unificada), tais como Diagrama de Casos de Uso e Diagrama de Sequência. Sendo assim, é possível definir alguns casos de uso para a aplicação, como é mostrado na Figura 3.3. O Usuário primeiramente inicia a aplicação, após isso ele deve enviar os comandos, a partir do teclado, que fazem o controle e a movimentação da motocicleta. Esses comandos

que fazem o controle da motocicleta executam especificadamente as seguintes ações: acelerar, freiar, direcionar a motocicleta, trocar de marchas e buzinar. Após isso, caso deseja-se parar, o usuário encerra a aplicação.

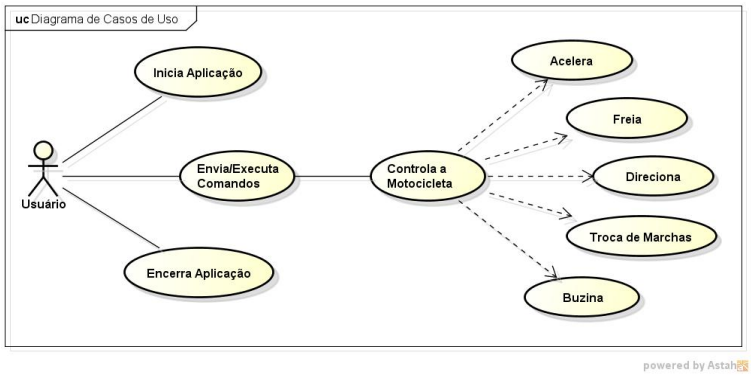

Figura 3.3 – Diagrama de Casos de Uso

Como já dito anteriormente, o funcionamento da aplicação se dá a partir da interação do usuário com a ferramenta, enviando comandos a partir do teclado para fazer a movimentação da motocicleta e consequentemente a reprodução dos sons correspondente a cada tipo de ação realizada. O diagrama de sequência visto na Figura 3.4 mostra como é feita a sequência de interação com a aplicação, de tal forma que primeiramente o usuário irá iniciar a aplicação, a partir dai pode enviar os comandos pelo teclado, que são recebidos pelos scripts que controlam a motocicleta e os sons, e executam as ações no cenário.

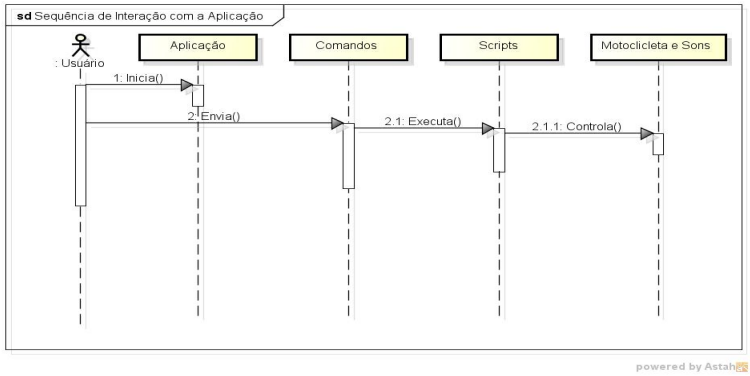

Figura 3.4 – Diagrama de Sequência da Aplicação

4 DESENVOLVIMENTO

Este capítulo descreve as etapas de desenvolvimento realizadas para atingir os objetivos desse trabalho, tratando dos aspectos e tudo que envolve a implementação do mesmo, descrevendo também algumas das principais ferramentas e componentes utilizados nas respectivas etapas de desenvolvimento.

O trabalho está dividido em quatro etapas, dada a complexidade de desenvolvimento em questão. Essa decisão foi tomada visando primeiramente entender o contexto ao qual o protótipo de simulação é implementado e de que forma isso deve ser feito, implementando os cenários de simulação e a motocicleta, deixando a etapa de implementação do sistema de áudio e a sincronização dos sons para o final, quando todos os pré-requisitos, necessários para a utilização correta do áudio na simulação, já estiverem concluídos.

A primeira etapa visou à instalação e configuração da *engine* de jogos Unity3D, bem como o entendimento da mesma para aprofundar o conhecimento sobre o ambiente de desenvolvimento. Na segunda etapa foi feita a criação do ambiente virtual de simulação, ou seja, a implementação da parte gráfica e física de um cenário na Unity3D (ruas, obstáculos, etc), visto que o objetivo do trabalho é sincronizar o sons com estímulos gerados pelo usuário e as imagens mostradas no ambiente virtual. A terceira etapa foi destinada a criação de uma motocicleta utilizando alguns componentes da engine, e a criação de scripts para fazer a movimentação da mesma, nesta etapa, propriedades físicas são adicionadas a motocicleta como forças para simular a aceleração, frenagem e inclinação em curvas. E finalmente, a quarta etapa é o desenvolvimento do protótipo de sistema de áudio, o qual foi implementado e testado para ver se atende aos requisitos de sincronização de áudio propostos para este trabalho. As seções a seguir explicam cada etapa de desenvolvimento.

4.1 Ambiente de Desenvolvimento

Antes de iniciar o desenvolvimento, com base nos requisitos, foi realizado um estudo para entender o conceitos e aplicações envolvidos com simulação (seção 2.1). A partir daí buscou-se por uma *engine* que atendesse aos propósitos do trabalho (seções 2.2 e 2.3). Depois de construir uma base sólida, decidiu-se começar o projeto utilizando a *engine* escolhida, podendo aplicar os conceitos estudados.

4.1.1 Escolha e Configuração da Engine

A *engine* de jogos escolhida para o desenvolvimento do projeto foi a Unity3D, versão 4.5.4, com licença livre. O seu uso se justifica pelo fato de que ela é uma ferramenta que possui inúmeros recursos para implementação de um protótipo de simulação e sincronização dos sons, proporcionando flexibilidade ao desenvolvedor quanto às maneiras de utilizá-la, além de ser uma das engines mais utilizadas atualmente, tanto para a criação de ambientes virtuais como para a criação de jogos e/ou simuladores. A documentação disponível em seu site é bastante farta e dessa forma auxilia os programadores nas mais diversas tarefas, durante o desenvolvimento de um projeto.

Para poder utilizar a Unity é necessário apenas instalá-la em um computador com o Sistema Operacional Windows. Sua instalação é simples e ela já vem com todas bibliotecas necessárias para o funcionamento. A máquina utilizada para o desenvolvimento é um Notebook HP com 3GB de RAM, processador Intel Core i5- 4200U de 1.60 GHz de 64 bits, executando Microsoft Windows 7 Ultimate.

4.2 Ambiente Virtual de Simulação

Tudo que é criado e desenvolvido em um simulador precisa estar inserido dentro de um mundo virtual, tornando possível a interação do usuário com o ambiente criado. Os cenários virtuais tem grande importância na área de simulação, pois é onde o usuário tem uma visão do que está acontecendo, auxiliando assim nas decisões tomadas. Basicamente, não existe um jogo ou simulador sem um cenário. Conforme Clua (2005) a criação da parte artística dos cenários, que compreende os elementos que serão usados para sua montagem: modelos 3D, texturas, terrenos, sons, músicas e arquivos de configuração, é a primeira e uma das principais partes implementadas no simulador, pois ela engloba todos os aspectos necessários para realizar a simulação, como trafegar em estradas, ruas, passar por obstáculos, fazer curvas, etc.

Para o desenvolvimento inicial do trabalho, foi necessário definir alguns cenários de treinamento para poder posteriormente criar todo o ambiente de virtual de simulação dentro da engine utilizada. A partir dai, desenvolver a parte física do simulador, como simulação de colisões, criação de obstáculos, criação e movimentação da motocicleta, etc. Todos os aspectos de criação do cenários serão detalhados nesta seção.

4.2.1 Criação do Cenário de Simulação

O objetivo do projeto é a sincronização de áudio com a parte gráfica do simulador, mas para isso é necessário a criação de cenários gráficos com algumas características como a representação de ruas e/ou estradas, contendo alguns tipos de obstáculos como quebra molas, aclives, declives, etc. Isto não é o foco do trabalho, mas uma capacitação via simulador para habilitar usuários para dirigir motocicletas exige a percepção de sons de diferentes ações básicas de pilotagem, como aceleração, desaceleração, frenagens, etc. Neste sentido os cenários são importantes.

Em primeiro lugar, foi criado um terreno utilizando o componente *Terrain*, o qual permite que seja adicionado vários tipos de paisagens ao cenário. O próximo passo foi a criação de algumas ruas e obstáculos como quebra-molas, por exemplo. Estas ruas delimitam por onde o usuário vai trafegar, sendo posicionadas de forma que fossem formadas retas, curvas, aclives e declives. Como parte cenográfica do ambiente, foram adicionadas algumas placas de trânsito e alguns cones, representando um cenário próximo do encontrado no mundo real. Para dar um aspecto mais realista, também são inseridas alguns tipos de iluminação na cena, denominada *DirectionalLights*, e algumas propriedades de renderização, como a textura do céu.

Esta etapa foi feita a partir da importação de *GameObjects*, que são objetos já modelados e disponíveis gratuitamente para download na própria AssetStore da Unity3D. Outros objetos utilizados, como as placas de transito e cones, foram importadas do Google SketchUp 8, que é uma ferramenta fornecida pela Google para a criação de modelos em 3D (Google SkecthUp, 2014). A construção das ruas, obstáculos, paisagens, ou seja, de todo o cenário de simulação, foi feito desta maneira. O cenário de simulação criado pode ser visto na Figura 4.1.

Figura 4.1 – Cenário Virtual de Simulação

4.3 Criação da Motocicleta

Para a criação da motocicleta no ambiente virtual foi necessário a importação, para a Unity3D, de um modelo de motocicleta 3D obtido através do SketchUp, pois a Unity3D não dispõe de ferramenta de modelagem de objetos 3D, exceto alguns modelos primitivos tais como cilindros, cubos, etc. A motocicleta é constituída por partes, ou seja, uma hierarquia de objetos onde cada componente dela é independente um do outro, mas juntos eles compõem o conjunto do modelo da moto, o que permite fazer com que as rodas girem e a movimentação da moto seja implementada de forma mais natural, podendo aplicar diversas técnicas físicas na mesma. O modelo de motocicleta utilizado pode ser visualizado na Figura 4.2.

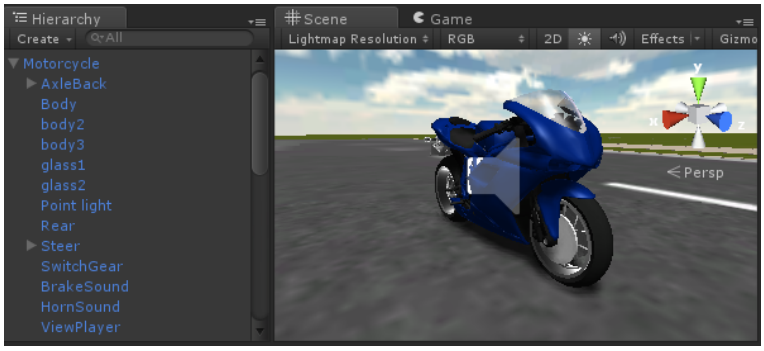

Figura 4.2 – Modelo 3D da Motocicleta

Cada objeto criado dentro do ambiente virtual de simulação da Unity, inclusive a motocicleta, pode possuir diversas propriedades e componentes, que são utilizadas para simular as características que deseja-se ter nesses objetos. Essas características são apresentadas através da classe *Component*, ela contém todas as referências para qualquer propriedade que esteja ligada a um *GameObject*, como pode ser visto na Figura 4.3. Algumas delas tais como *Terrain Collider* e *Wheel Collider* são adicionadas tanto ao cenário quanto à motocicleta, respectivamente.

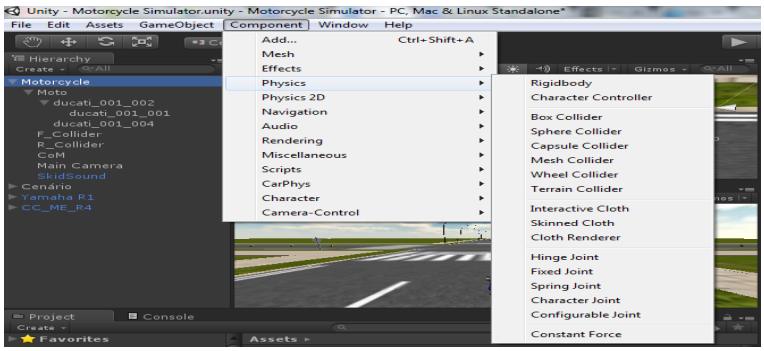

Figura 4.3 – Atributos da classe Component

4.3.1 Componentes Físicos da Motocicleta

O modelo de motocicleta é composto por uma série de componentes que permitem que sejam feitas as mais diversas interações físicas. O componente Rigidbody presente na motocicleta, permite o objeto agir de acordo com diversos parâmetros. Ele habilita o objeto a receber forças de outros objetos e a se mover de uma forma mais realista, de acordo com características como massa, gravidade, resistência ao ar, etc.

Os componentes *Colliders* são responsáveis pela movimentação e pela maioria das interações que acontecem entre os elementos de um jogo. Eles trabalham juntamente com os *Rigidbody* trazendo as características físicas do mundo real para os objetos. Na Figura 4.4 é possível observar que o objeto que representa a motocicleta possui um *Rigidbody*, e o componente *Box Collider* o que significa que ela está sujeito à física, à gravidade e interferência de outros objetos que tenham propriedades de colisão, esses componentes podem ser controlados por scripts.

Figura 4.4 – Componentes da Motocicleta

4.3.2 Movimentação da Motocicleta

O comportamento dos *GameObjects* é controlado pelos componentes que estão ligados a eles. Embora os componentes integrados do Unity possam ser muito versáteis, é necessário ir além do que eles podem oferecer para implementar suas próprias características de jogabilidade e/ou movimentação. A Unity3D permite que sejam manipuladas essas características usando scripts. Estes permitem desencadear eventos do simulador, modificar as propriedades do componente ao longo do tempo e responder à entrada e iterações do usuário, de diversas formas.

Diferentes linguagens de programação foram avaliadas para serem utilizadas neste trabalho, especialmente para o controle físico e para a movimentação da motocicleta. A Unity3D, como já vimos, suporta três linguagens de script, C#, JavaScript e Boo. Mas por causa do nível de acesso a alguns componentes físicos como o *rigidbody*, por exemplo, C# foi escolhida como a linguagem de programação de scripts para fazer a movimentação da motocicleta.

4.3.3 Script de Movimentação

Basicamente, foram criados Scripts de movimentação e controle físico da motocicleta. A Unity possui uma IDE de desenvolvimento integrada, a MonoDevelop, a qual possibilita a edição de scripts, principalmente os suportados pela engine. Para que a movimentação da motocicleta seja realizada, é necessário adicionar o componente Script ao modelo para que o script criado possa atuar sobre os componentes *Rigidbody* e *Colliders* que o objeto possui, e a partir da iteração do usuário com a ferramenta, utilizando os comandos vindos do teclado, possa movimentar a motocicleta no cenário.

O Script atua sobre o objeto a partir da obtenção de uma referência para o mesmo, pelo código do script. A partir dessa referência, é possível calcular e ajustar o controle de movimentação da moto, pois o centro de massa habilita a motocicleta a receber forças e ter um ponto de equilíbrio fazendo ela ganhar estabilidade para se manter na posição vertical. As linhas de código abaixo mostram como é obtida a referência ao centro de massa do objeto(a motocicleta):

```
var CoM : Transform; // centro de massa
rigidbody.centerOfMass = Vector3(CoM.localPosition.x, CoM.localPosition.y, CoM.localPosition.z);
```

Um exemplo de aplicação e utilização de forças físicas pode ser visto na função Update(), mostrada na Figura 4.5, onde são visualizados os cálculos das forças que podem ser aplicadas nas rodas da motocicleta. Esta função é chamada a cada frame, durante a execução do do projeto criado e utilizado na Unity3D. Nesta função, **motorTorque** é utilizado para aplicar força (torque) no eixo da roda traseira da motocicleta, o valor pode ser positivo ou negativo dependendo do sentido das teclas pressionadas no teclado, o que faz e motocicleta se movimentar. Contudo, são utilizados apenas valores positivos, pois a motocicleta se desloca apenas para frente. Já para simular a ação do freio, outra força é aplicada à roda da motocicleta, essa força é chamada **brakeTorque**. As variáveis **WheelF** e **WheelR** são do tipo *WheelCollider*, as quais

recebem referência das rodas da motocicleta. Desta forma, as rodas passam a se movimentar, e por consequência movimentam o resto da moto, a partir da aplicação destas forças.

```
161 function Update () {
162     //calcula a velocidade da moto
163     CurSpeed = rigidbody.velocity.magnitude*2.0;
164 
165     //Aceleraçao da moto
166     if(Input.GetAxis("Vertical") > 0){
167         WheelR.motorTorque = Torque * Input.GetAxis ("Vertical");
168     }else{//freio da moto
169         WheelR.brakeTorque = Freio * Input.GetAxis("Vertical");
170     }
171     //direçao da moto
172     WheelF.steerAngle = SteerAngle * Input.GetAxis ("Horizontal");
173 
174 }
```

Figura 4.5 – Exemplo de Uso de Forças Físicas (Função Update).

A direção da motocicleta é calculada através da adição de uma propriedade na roda dianteira, chamada **SteerAngle**. Conforme o usuário muda de direção, a variável aumenta seu valor e o ângulo da roda é alterado fazendo a moto se direcionar para um dos lados, esquerdo ou direito.

Desta forma, terminada essa etapa e concluída a parte gráfica dos cenários e movimentação da motocicleta, os requisitos principais para poder aplicar os sons no ambiente de simulação já estão prontos. Então, deu-se início ao próximo passo que é a implementação do sistema de áudio da motocicleta e sincronização dos sons 3D, e será descrito a partir da seção seguinte.

4.4 Sistema de Áudio e Sincronização dos Sons

A Unity fornece mecanismos de controle e manipulação de áudio permitindo a inclusão arquivos de áudio, importação e configurações de som em ambiente tri-dimensional, etc. A teoria básica por trás do áudio da Unity diz que a forma como um som é percebido depende de alguns fatores, tais como: (i) Um ouvinte pode dizer mais ou menos a direção que o som está vindo e pode também obter algum sentido da distância, da intensidade e da qualidade do mesmo. (ii) Uma fonte sonora em movimento rápido, como uma ambulância passando, por exemplo, vai mudar a intensidade conforme ela se move, resultado do efeito Doppler (UNITY, 2014). Serão demonstrados, nesta seção, os componentes, os cálculos e algoritmos principais utilizados na aplicação e sincronização dos sons.

4.4.1 Componentes de Áudio Fundamentais

Para simular alguns efeitos sonoros, a Unity3D permite que sejam adicionados arquivos de áudio ao componente *AudioSource*, que são ligados aos objetos. Os sons emitidos podem então serem ouvidos por um *AudioListener*, que também é ligado a outro objeto, na maioria das vezes esse componente é adicionado à câmera principal. É possível simular os efeitos da distância de um *Source* para a o objeto *Listener* e reproduzí-los ao usuário de acordo com os parâmetros estipulados. A velocidade relativa entre os objetos *Listener* e *Source* pode também ser usado para simular o efeito de Doppler, e aumentar o realismo (UNITY, 2014). Esses componentes de áudio são destacados nos itens a seguir e na Figura 4.6:

- **Audio Source:** Este componente de áudio é capaz de reproduzir um clipe de áudio(Audio Clip) no cenário. Se o Audio Clip é um arquivo de som 3D, a fonte sonora é reproduzida em uma determinada posição da cena, podendo ser controlada a atenuação e a distância.
- **Audio Listener:** O Audio Listener atua como se fosse um microfone, recebendo a entrada de qualquer Audio Source presente na cena e transfere esses sons para os auto-falantes. Como citado anteriormente, esse componente é utilizado normalmente na câmera principal do jogo.

Figura 4.6 – Componentes Áudio Sources e Listener (Unity, 2014)

A Unity possui um sistema de coordenadas 3D onde qualquer *GameObject* sendo ele um objeto, um efeito ou um som reproduzido na cena, pode estar posicionado em qualquer ponto desse sistema. A Figura 4.7 retrata esse cenário mostrando as coordenadas dos objetos na cena. A Unity reproduz o áudio posicionalmente, assim como faz graficamente com os objetos, de modo que seja possível perceber de que parte do cenário o som está sendo reproduzido, se ele está mais distante, mais perto e a direção, por exemplo, sendo essas as características do áudio 3D.

Alguns parâmetros podem ser alterados de modo que o áudio tenha suas propriedades definidas da forma que se deseja que ele seja reproduzido, até que distância ele poderá ser ouvido, o volume, etc. A Unity3D simula todos os aspectos de áudio 3D fundamentais para a utilização e sincronização dos sons, sendo necessário somente desenvolver os algoritmos e as técnicas de sincronização dos sons com a imagem, propostos para esse trabalho.

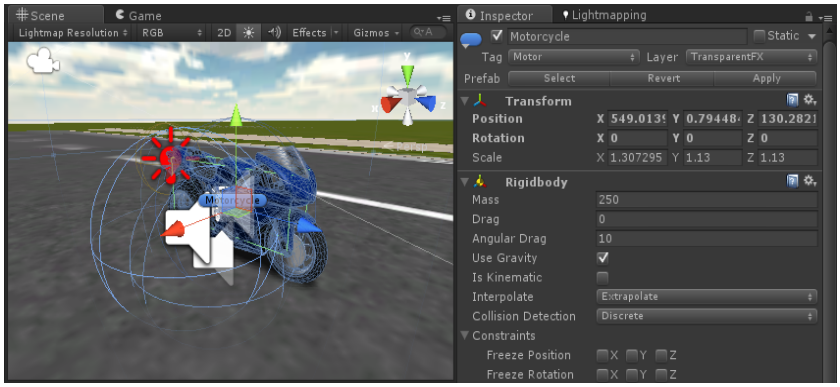

Figura 4.7 – Coordenadas dos objetos na cena(Componente Transform)

4.4.2 Som de Aceleração e Desaceleração

O áudio que é reproduzido no cenário, possui diversas características que podem ser visualizadas através do *Inspector* da Unity3D, como mostra a Figura 4.8. O controle do arquivo de áudio 3D adicionado ao componente *Audio Source* (que é ligado a motocicleta), utilizado na simulação do som de aceleração e desaceleração, é feito a partir do cálculos sobre da variável Pitch. Alguns parâmetros são atribuídos para que o áudio continue executando mesmo que a moto esteja parada, e a execução do som começe quando a aplicação for inicializada. Para isso, marcamos como verdadeiros os campos *Loop* e *Play On Awake*, também vistos na Figura 4.8.

Algumas propriedades são ainda mais específicas de cada arquivo de áudio utilizado, (isso vale para cada efeito de som e sincronização aplicado) justamente para gerar o efeito tridimensional que o som ganha quando adicionado ao ambiente virtual de simulação. Características como Volume, Distância, Nível de Efeito Doppler, *Pan* e *Spreed*, são específicas do Áudio 3D, e podem ser visualizadas graficamente na Figura 4.8, abaixo do item *3D Sound Settings*. Essas funções entre Distância e Volume, Pan, Spread e filtros de áudio Low-Pass bem como a distância atual do *Audio Source* para o *Audio Listener* também é mostrada no gráfico. Conforme o objeto emissor dos sons se aproxima do *Audio Listener*, o som torna-se mais intenso, ou seja, o volume aumenta dando a sensação de que o objeto está realmente se aproximando.

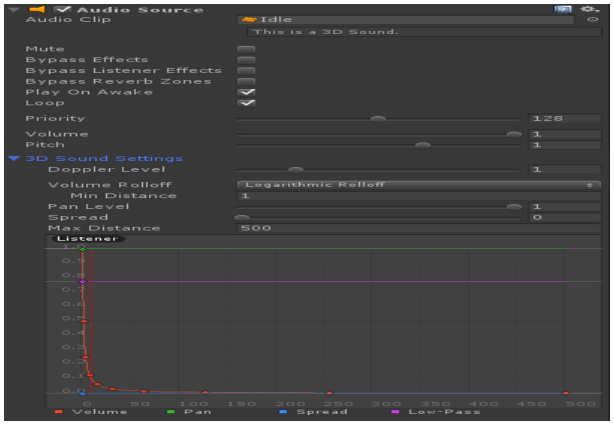

Figura 4.8 – Parâmetros do Audio Source e Propriedades do Som 3D

O cálculo básico para a simulação do áudio de aceleração e desaceleração é mostrado na Figura 4.9, onde é verificado se existe um arquivo de áudio 3D conectado ao componente *Audio Source*, aí então é feito o cálculo a partir das variáveis utilizadas na movimentação da motocicleta. A sincronia do som depende muito da aceleração da moto e o torque aplicado na aceleração. Conforme a aceleração aumenta, a taxa de rotações por minuto (rpm) da motocicleta também é alterada, modificando o valor do *Pitch*, que é a variável de mudança de tom devido à desaceleração/aceleração do *Audio Clip* (arquivo de áudio), que é a referência para o arquivo áudio que será reproduzido. A medida que esse valor aumenta ou diminui, dá a sensação que a moto está acelerando ou diminuindo sua velocidade. O valor do Pitch varia de 0 a 3, onde 0 (zero) significa que a frequência de reprodução do som é mínima, 3(três) é máxima e valor 1 é a velocidade de reprodução normal (UNITY, 2014).

```
// Se tiver um AudioSource (Idle) ajusta o pitch usando rpm
if (audioSource != null){
    // calcula o pitch (mantem dentro dos limites razoaveis)
    float pitch = Mathf.Clamp(1.0f + ((motoRPM - idleRPM) / (shiftGearUpRPM - idleRPM) * 1.0f), 1.0f, 3.0f);
    audioSource.pitch = pitch;

    if (!audioSource.isPlaying) {
        audioSource.Play();
    }
}
```

Figura 4.9 – Controle do Som de Aceleração através da Variável Pitch

4.4.3 Som da Troca de Marchas

Para a reprodução do som referente a troca de marchas da motocicleta, também foram feitos alguns cálculos no script utilizado para a movimentação da motocicleta, pois depende de alguns parâmetros e variáveis utilizadas e que são interligadas para que o funcionamento se dê corretamante, o trecho de código onde é feita a parte de som das marchas é visto na Figura 4.10.

```
//mudança de marcha para cima
public void ShiftGearUp(){
    float time = Time.timeSinceLevelLoad;

    //se o tempo for menor que o delay entre as trocas de marcha, nao troca
    if (time < shiftGearDelay) return;

    //verifica se e possivel trocar a marcha para cima, entao executa o som
    if (currentGear < gears.Length - 1){
        if (!switchGear.isPlaying){
            switchGear.audio.Play();
        }

        currentGear++;

        //deixa um delay de 1 segundo entre as trocas de marchas
        shiftGearDelay = time + 1.0f;
    }
}
```

Figura 4.10 – Função que Dispara o Som da Troca de Marchas

É verificado se o tempo atual de execução da aplicação (time) é menor que o delay estipulado para o intervalo entre as trocas de marchas, se verdadeiro não executa o som. Por definição, foi decidido que o tempo entre as trocas de marchas demora pelo menos 1 segundo, assim, se a marcha atual for menor que o tamanho do vetor estipulado de tamanho 5 para as marchas, ele executa o som através do comando **switchGear.audio.Play()**. Quando *currentGear* for maior que o tamanho do vetor (currentGear < gears.length-1) , mesmo que se tente executar a troca de marcha isso não irá ocorrer devido a essa condição. Temos algumas outras condições para que ocorra o som da troca de marchas, tanto se a moto for automática ou for com troca de marchas normal, como podemos ver na Figura 4.11. A troca de marchas automática depende de alguns fatores, como a marcha atual (currentGear) e a aceleração (acceleration), já para a troca normal, são recebidos como entrada comandos do teclado, assim é disparado a troca de marcha e o respectivo som. O arquivo de áudio responsável pelo efeito da troca de marchas também é inserido no script através de um *GameObject* que contém um componente *AudioSource*, responsável pela emissão do som.

```
//troca de marchas automatica
if (automatic && (currentGear == 1) && (acceleration < 0)){
    if (speed < 1.0)
        ShiftGearDown(); //neutro
}
else if (automatic && (currentGear == 0) && (acceleration > 0)){
    if (speed < 5.0)
        ShiftGearUp(); //troca no neutro para a primeira marcha
}
else if (automatic && (motoRPM > shiftGearUpRPM) && (acceleration > 0) && !brake){
    // if (speed > 20)
    ShiftGearUp();
}
else if (automatic && (motoRPM < shiftGearDownRPM) && (currentGear > 1)){
    ShiftGearDown();
}

//troca de marchas manual
if(!automatic && Input.GetKey (KeyCode.PageUp)){
    ShiftGearUp();
}
else if(!automatic && Input.GetKey (KeyCode.PageDown)){
    ShiftGearDown();
}
```

Figura 4.11 – Trecho de Código que faz a Troca de Marchas Automática e Normal

4.4.4 Som das Frenagens e Derrapagens

Algumas implementações requerem a utilização de *Prefabs*, que é um tipo de *Asset* que permite guardar um *GameObject* completo com componentes e propriedades. Os *Prefabs* atuam como um modelo a partir do qual é possível criar novas instâncias de *GameObjects* na cena, sejam eles objetos 3D, efeitos visuais ou sonoros. Quaisquer edições feitas em um *Prefab* são imediatamente refletidas em todas as instâncias produzidas a partir dele, mas também é possível substituir os componentes e as configurações para cada instância individualmente.

Dessa forma, diferentemente do som utilizado para reproduzir o motor da motocicleta, que executa repetidamente, mudando a frequência com que é reproduzida, os sons da ação de frenagem e derrapagem devem ser disparados somente quando acionar o freio ou desacelerar de forma mais brusca. Utiliza-se um componente *Audio Source* que é adicionado ao *Prefab* para que o som respectivo seja reproduzido a cada vez que o usuário executar a ação de frenagem.

Quando a aplicação é iniciada, a execução do som de frenagem é feita somente quando for acionado o comando de freio ou de desaceleração. O algoritmo e os cálculos principais de sincronização utilizado para isso é mostrado no trecho código mostrado na Figura 4.12, a seguir.

```
if (brakePrefab){
    if (Wheel[WheelAtual] == null){
        Wheel[WheelAtual] = Instantiate(brakePrefab, w.transform.position, Quaternion.identity) as GameObject;
        Wheel[WheelAtual].transform.parent = transform;
        Wheel[WheelAtual].AddComponent<AudioSource>().clip = brakeSound;
    }

    if (((brake || acceleration < 0.0f) && speed > 20) && currentGear > 0){
        if ((acceleration < 0.0f) || (brake)){
            if (!Wheel[WheelAtual].audio.isPlaying)
                Wheel[WheelAtual].audio.Play();
                Wheel[WheelAtual].audio.volume = Mathf.Clamp((speed / 45), 0, 1.0f);
        }
    }

}
```

Figura 4.12 – Trecho de Código que faz a Execução do Som das Frenagens e Derrapagens

Verifica-se primeiramente se existe o *Prefab*, então ele é adicionado ao vetor Wheel[WheelAtual] que representa as rodas, e que recebe a força do freio que executa a ação de desaceleração e parada da moto. No mesmo vetor, é adicionado um componente de áudio *Audio Source*, cuja referência é feita através da variável *brakeSound*, que irá executar o arquivo de áudio assim que as condições de frenagem forem satisfeitas. Se o freio é acionado, a velocidade for maior que 10 e a marcha atual for maior que zero, por exemplo, o áudio de simulação da derrapagem é acionado, a partir do comando **Wheel[WheelAtual].audio.Play()**. A intensidade da reprodu-

ção do som varia conforme a velocidade, ou seja, quanto mais rápido a moto estiver, mais alto será a reprodução do som quando acionado o freio, e vice-versa.

4.4.5 Som da Buzina

A simulação da buzina é feita de forma mais simples, inserindo um arquivo de áudio no formato .wav ao componente *Audio Source* que será adicionado a motocicleta da aplicação, o script pega a referência para esse *Audio Source* e dispara o efeito sonoro quando um botão do teclado for pressionado. O tempo de execução desse arquivo de som é bem curto, cerca de 0.1 segundos, então, o som fica executando em *loop* enquanto o botão for pressionado, simulando o som de uma buzina. A execução do som da buzina é realizado através do comando **hornSound.audio.play**().

Todos os arquivos de Áudio 3D utilizados para sincronizar e simular as ações foram obtidos através de bibliotecas de efeitos sonoros disponíveis na internet, e estes são inseridos no script a partir de *GameObjects*. Pode-se destacar que foram escolhidos os sons que pudessem representar os sons reproduzidos por uma motocicleta, não buscando a fidelidade e a qualidade do som, mas que desse para fazer a sincronização destes com as ações executadas e as imagens mostradas.

Variáveis são criadas no script para receber cada *GameObject* referente ao áudio específico. Cada variável criada no script, aparece no *Inspector* da Unity para que o *GameObject* seja utilizado, e o script controle cada evento gráfico e sonoro realizado no cenário. Essas variáveis dos scripts e os *GameObjetcs* utilizados pelos scripts são mostrados na Figura 4.13.

Figura 4.13 – Variáveis e Componentes Utilizadas pelo Script

Desta forma, os requisitos da aplicação e sincronização dos sons propostos para este trabalho foram satisfeitos, considerando alguns dos sons principais reproduzidos por uma motocicleta.

5 EXPERIMENTOS E RESULTADOS

Neste capítulo são fornecidas informações sobre os experimentos realizados e os resultados alcançados, juntamente com a descrição dos cenários utilizados para tais experimentos.

5.1 Cenários de Experimentação

Para a demonstração dos experimentos, montou-se um cenário onde é possível interagir com o protótipo desenvolvido, sentir os estímulos sonoros gerados por essa interação e perceber a sincronização dos sons com a imagem mostrada na tela de visualização. Ele é mostrado na Figura 5.1, e corresponde ao do ambiente descrito na seção 3.2, onde observa-se um monitor de visualização, onde visualiza-se o cenário de simulação e a motocicleta, um teclado, onde o usuário pode executar os comandos para a movimentação e controle da moto, e as caixas de som, onde os efeitos de áudio são reproduzidos conforme a posição dos sons e objetos na cena.

Desta forma, para simular um sistema *Surround* foi utilizado nesse cenário, caixas de som estéreo conectadas ao computador e dispostas de modo que seja possível perceber as características e os efeitos do som 3D aplicado. Ainda, para ter uma melhor distribuição do som, foi instalado uma Placa de Som Surround PCI 7.1 Canais da Encore (Encore Eletronics, 2014) no desktop que foi utilizado para executar a aplicação e fazer os experimentos, pois ela torna possível a reprodução dos sons, pelas caixas acústicas, com uma boa qualidade. Como a simulação de um ambiente *Surround* depende muito da tecnologia utilizada, a maneira mais barata e possível que se encontrou para simular essa técnica foi utilizando caixas de som e a placa de *surround* instalada no computador, a qual distribui o áudio para as caixas de som conectadas conforme fazendo que elas reproduzam os sons conforme as características de posição do áudio no cenário desenvolvido.

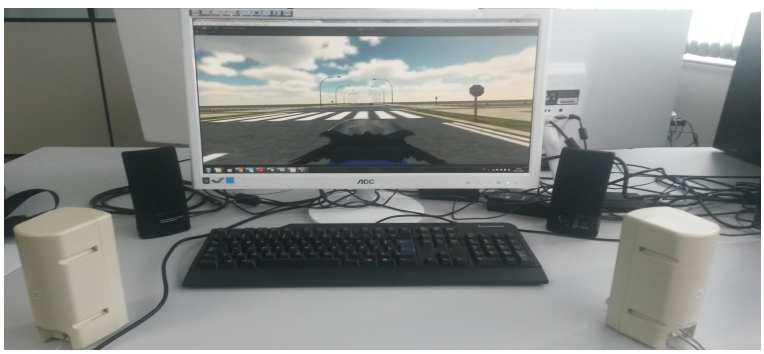

Figura 5.1 – Cenário de Experimentação Utilizando Caixas de Som

Outro cenário importante que foi testado, ao invés de utilizar caixas de som, foi utilizando fones de ouvido estéreos, os quais são capazes de reproduzir melhor os aspectos de áudio 3D adicionado ao cenário, sendo possível o usuário perceber melhor os sons, e também as carecterísticas de som da técnica de *Surround*, dando a sensação de estar imerso ao ambiente de simulação. A Figura 5.2 ilustra como é simulado o *Surround* utilizando fones de ouvido.

Figura 5.2 – Cenário de Experimentação Utilizando Fones de Ouvido

Para a definição de quantos canais de áudio são utilizados, habilita-se alguns parâmetros, na Unity 3D, para que os auto-falantes reproduzam os sons de acordo com as suas características, e seja possível fazer essa simulação. Através do componente *AudioManager*, visualizado na figura 5.3, o parâmetro *Default Speaker Mode* é habilitado para o modo *surround*, que inclui a utilização de até 5 auto-falantes.

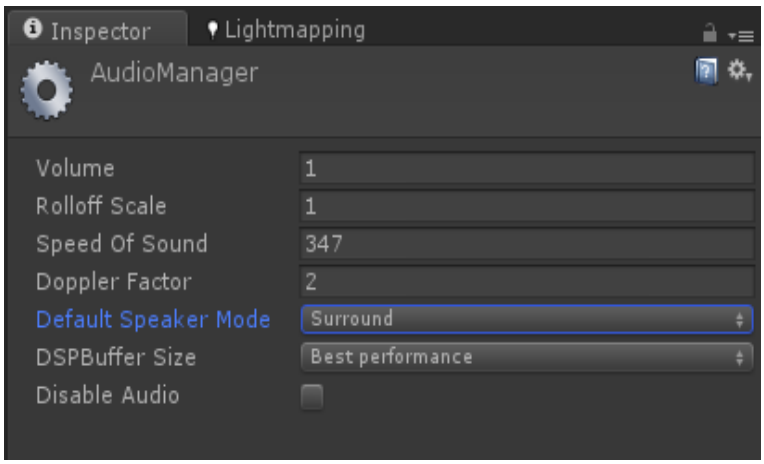

Figura 5.3 – Parâmetros Componente Audio Manager.

5.2 Demonstrações e Resultados das Funcionalidades

Cada etapa de sincronização implementada é descrita a seguir, a fim de explicar como realmente ocorre cada evento e como o som se comporta no cenário de simulação a cada estí-

mulo gerado pela interação com o protótipo. Dessa forma, é possível realizar experimentos que envolvem o que foi citado no capítulo de proposta e descrito no capítulo de desenvolvimento. As etapas são discutidas nas subseções, a seguir.

5.2.1 Aceleração, desaceleração e troca de marchas

Este experimento teve por objetivo testar a resposta da motocicleta em relação ao sincronismo dos sons de aceleração, desaceleração e troca de marchas. O experimento consistiu em andar com a motocicleta pelo cenário e submeter a situações onde pudessem ser testados esses casos. Momentos de aceleração e desaceleração foram alternados, de modo a perceber a sincronia dos sons. Durante a pilotagem da moto, a troca de marchas também se mostrou eficiente, e dois casos foram testados: (i) a troca de marchas automática, onde, a medida que a moto for aumentando sua velocidade a troca de marchas vai acontecendo e (ii) a troca de marchas normal, onde a partir de um comando enviado pelo usuário através do teclado é possível executar a troca de marchas. Os dois casos mostraram-se satisfatórios tratando-se de sincronização e efeito dos sons.

5.2.2 Frenagens e Derrapagens

Utilizando o protótipo desenvolvido, observou-se nesse experimento que o áudio reproduzido pela ação de frenagem foi semelhante ao que podemos ouvir na realidade. Pecebe-se o som da frenagem e/ou derrapagem quando se aplica a ação do freio. O usuário acelera a moto, e quando diminui a velocidade ou freia bruscamente, pode ouvir o som da derrapagem da roda no 'asfalto'. Quando diminui a velocidade da moto, o som é disparado no momento em que a moto está quase parando. Se for utilizado o freio de forma brusca, o som é logo disparado, dando a sensação de derrapagem.

5.2.3 Simulação de Ultrapassagem por Veículos

Foi simulado a ultrapassagem da moto por outros objetos que estejam reproduzindo algum tipo de som, ou seja, outras fontes sonoras, no caso objetos que representam outros veículos. Na ultrapassagem por outros veículos é possível perceber de qual lado e qual a proximidade relativa a motocicleta, o som está sendo reproduzido. Essa é uma situação que pode ocorrer no dia-a-dia, quando muitas vezes o piloto não visualiza o automóvel fazendo a ultrapassagem mas

percebe pelo som que o mesmo está se aproximando. O Momento da ultrapassagem pode ser visualizada na Figura 5.4.

Figura 5.4 – Simulação de Ultrapassagem por outro Veículo

O efeito produzido pela ultrapassagem da moto por outros veículos é chamado de efeito Doppler. O usuário é capaz de perceber, através do som, outro objeto se aproximando. O volume e a intensidade ficam maiores a medida que o objeto *Source* se aproxima do objeto *Listener*, no momento que ambos se encontram, temos o pico de maior intensidade do som, a medida que eles se afastam novamente o som vai diminuindo de intensidade. Como introduzido na seção 4.4.1, esse efeito é causado pela velocidade relativa entre os componentes *source* e *listener*, dando ao ouvinte a sensação de efeito Doppler. Podemos ver no gráfico da Figura 5.5 que a medida em que o *source* se aproxima do *listener*, o volume do som vai aumentando, dando essa sensação.

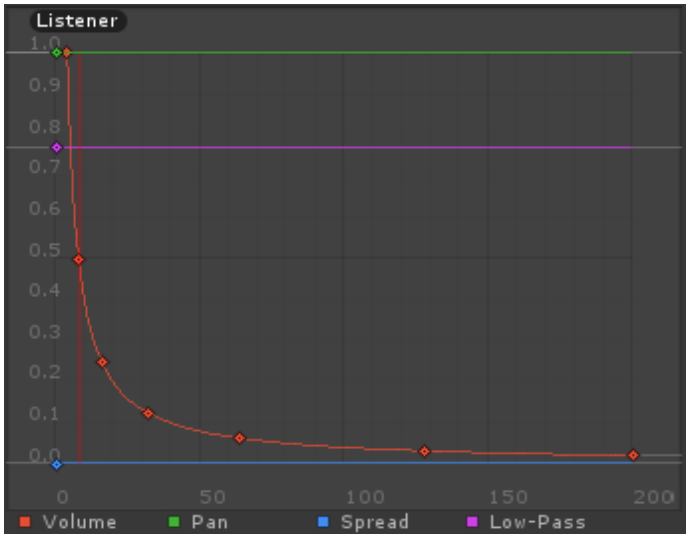

Figura 5.5 – Relação entre a distância do objeto Source para o Listener e o Volume do Som

Em todos os experimentos realizados no ambiente virtual de simulação e discutidos nas subseções enteriores, foi possível perceber a sincronia entre o controle da moto e os sons reproduzidos. Observou-se com clareza todas essas ações, mesmo que com algum atraso em alguns casos, como no momento da arrancada da moto, o som demora um pouco a ser executado conforme aceleração da moto, mostrada na imagem pela tela de visualização. A percepção sonora, dá a sensação de estar realmente pilotando uma moto, e sentir-se imerso no ambiente de simulação.

Com a utilização de fones de ouvido para a percepção dos sons, foi posível ter uma melhor percepção do áudio reproduzido. Contudo, com a utilização do primeiro cenário, simulando um ambiente *Surround* através das caixas de som, também foi posssível perceber os efeitos sonoros reproduzidos pelo protótipo.

6 CONCLUSÃO E TRABALHOS FUTUROS

Neste trabalho foi descrito o processo de desenvolvimento de um protótipo de simulação de motocicleta, com foco nos mecanismos de áudio, que teve por objetivo a sincronização dos sons com a imagem reproduzida numa tela de visualização, para auxíliar no aprendizado do usuário e melhorar a sensação de imersão do mesmo ao cenário de simulação. Durante o processo de estudo foi identificado que a utilização de Áudio 3D e a técnica de *Surround Sound* melhoraria esse aspecto, dando a sensação de realismo para o usuário, podendo obter a experiência da percepção sonora ao interagir com a ferramenta desenvolvida. Contudo, antes de trabalhar com o som em si, foi necessário criar uma base, ou seja, um cenário onde fosse possível conduzir uma moto, e a propria motocicleta, onde o aspecto físico foi fundamental para a implementação do resto do trabalho. Para isso, foi utilizada a *engine* de jogos Unity3D, a qual pôde ser bastante explorada, devido a sua vasta documentação disponível em seu site, facilitando a implementação da aplicação. Tudo feito utilizando objetos pré-fabricados, (Prefabs ou objetos 3D) o que também agilizou o processo de implementação.

Assim, com esse trabalho podemos concluir que o protótipo atende as funcionalidades principais definidas no capítulo 3 e as características básicas de pilotagem de uma motocicleta podem ser percebidos com naturalidade através dos sons, os quais o protótipo é capaz de simular. A questão de distância, e as características 3D dos efeitos sonoros empregados ao trabalho, satisfazem aquelas citadas na seção 2, onde o Áudio 3D e a técnica de Surround Sound desempenham um importante papel para a imersão do usuário ao cenário de treinamento. Em outras palavras, podemos dizer que é possível criar experiências imersivas de som em simuladores com a utilização de som 3D.

Devido ao pouco tempo e a dificuldade de avaliar alguns cenários, voltados à capacitação de motoristas, foram demonstrados e testados as funcionalidades implementadas e a simulação de ultrapassagem da moto por outros veículos, somente, as quais correspondem aos objetivos propostos. A seguir há algumas sugestões de trabalhos futuros:

- Aplicar outros tipos de som ao protótipo, como o som do pisca(seta), colisões, etc.
- Criar cenários mais realistas para poder simular melhor um ambiente encontrado no dia-a-dia.
- Fazer integração com uma plataforma de movimentos(motion), etc.

REFERÊNCIAS

ASHAOLU, P. Development of an Interactive 3-D Virtual Environment Test Bed For Motorbikes. **Electrical and Information Technology Research Unit, Savonia University of Applied Sciences**, [S.l.], 2012.

BALBINOT, A. B.; TIMM, M. I.; ZARO, M. A. Aplicação de Jogos e Simuladores como Instrumentos para Educação e Segurança no Trânsito. **Novas Tecnologias na Educação**, [S.l.], v.7, n.1, 2009.

BENYON, D. Interação Humano-Computador. 2ª Edição. , [S.l.], 2011.

CLUA, E. W. G.; BITTENCOURT, J. R. Desenvolvimento de Jogos 3D: concepção, design e programação. **Anais da XXIV Jornada de Atualização em Informática do Congresso da Sociedade Brasileira de Computação**, [S.l.], p.1313–1356, 2005.

COSSALTER, V. et al. A motorcycle riding simulator for assessing the riding ability and for testing rider assistance systems. **DSC**, [S.l.], p.48–58, 2006.

ELETRONICS, E. **Encore Eletronics**. Disponível em: http://www.encore-usa.com/br/product/ENM232-8CMI, Acesso em Outubro de 2014.

FERRAZZIN, D. et al. THE MORIS MOTORCYCLE SIMULATOR: an overview. **SAE Technical Paper**, [S.l.], 2001.

FULLER, R. Driver training and assessment: implications of the task-difficulty homeostasis model. **Proceedings of the Third International Conference on Driver Behaviour and Training**, [S.l.], v.3, p.337–348, 2008.

Google SketchUp. **SketchUp**. Disponível em: http://www.sketchup.com, Acesso em Setembro de 2014.

NILSSON, L. . Behavioural research in an advanced driving simulator-experiences of the VTI system. **Proceedings of the Human Factors and Ergonomics Society 37th Annual Meeting 37**, [S.l.], p.612–616, 1993.

PAUL, R. L. Desenvolvimento de um case para simulação de motocicleta. , [S.l.], 2014.

PEREIRA, J. A. Estudo sobre a Influência dos Jogos de Corrida no Comportamento de Motoristas. **VIII Brazilian Symposium on Games and Digital Enterteinment**, [S.l.], p.30–33, 2009.

PROCACCINI, M. Evaluation of the perceived sense of speed in a driving simulator. **Master's thesis, Högskolan i Skövde**, [S.l.], 2013.

SAUVE, L. et al. Distinguishing between games and simulations: a systematic review. **Educational Technology e Society**, [S.l.], v.10, n.3, p.247–256, 2007.

SIMULADORES, R. **Rota Simuladores**. Disponível em: http://www.rotasimuladores.com.br, Acesso em Julho de 2014.

STEDMON, A. W. et al. MotorcycleSim: an evaluation of rider interaction with an innovative motorcycle simulator. **The Computer Journal**, [S.l.], v.54, p.1010–1025, 2011.

TANG, A. 3D Sound Aplication for Game Environments. **Malardalens Universit in Vasteras**, [S.l.], 2014.

THORN, A. Game Developement Principles (1st ed.). **Boston: Cengage Learning**, [S.l.], 2013.

Unity Techonogies. **UNITY 3D, Manual.** Disponível em: http://docs.unity3d.com/Manual/, Acesso em Setembro de 2014.

WENFENG, H.; ZHOUQING, Q.; XIAOYUAN, Z. A New Approach of Mechanic Simulation Based on Game Engine. **Fifth Internation Joint Conference on Computational Sciences and Optimization**, [S.l.], 2012.

YAMASAKI, G. et al. Development of motorcycle training simulator. **JSAE**, [S.l.], p.81–85, 1998.

ZERBST, S.; DüVEL, O. 3D Game Engine Programming. **Thomson Course Technology, Premier press**, [S.l.], 2004.