INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO PIAUÍ
CAMPUS PARNAÍBA
LICENCIATURA EM FÍSICA

VINICIUS SEREJO PINHEIRO

PRODUTO EDUCACIONAL: FÍSICA NA CRIAÇÃO DE JOGOS ELETRÔNICOS EM UNITY 3D

PARNAÍBA - PI
2018

VINICIUS SEREJO PINHEIRO

PRODUTO EDUCACIONAL: FÍSICA NA CRIAÇÃO DE JOGOS ELETRÔNICOS EM UNITY 3D

Trabalho de Conclusão de Curso (monografia) apresentado como exigência parcial para obtenção do diploma do Curso de Licenciatura em Física do Instituto Federal de Educação, Ciência e Tecnologia do Piauí, Campus Parnaíba.

Orientador: Prof. Msc. Bruno Pires Sombra

PARNAÍBA
2018

VINICIUS SEREJO PINHEIRO

PRODUTO EDUCACIONAL: FÍSICA NA CRIAÇÃO DE JOGOS ELETRÔNICOS EM UNITY 3D

Trabalho de Conclusão de Curso (monografia) apresentado como exigência parcial para obtenção do diploma do Curso de Licenciatura em Física do Instituto Federal de Educação, Ciência e Tecnologia do Piauí. Campus Parnaíba.

Aprovada em: ___/___/_____.

BANCA EXAMINADORA

_____

Prof. Msc. Bruno Pires Sombra (Orientador)
Instituto Federal do Piauí (IFPI)

_____

Prof. Prof. Msc. Deymes Silva de Aguiar
Instituto Federal do Piauí (IFPI)

_____

Prof. Prof. Msc. Lucas Izídio de Sousa Sampaio
Instituto Federal do Piauí (IFPI)

À minha noiva, Jessica, que me inspirou a ser a melhor pessoa que eu puder.

Agradecimentos

Agradeço aos meus pais, por sustentarem todos os meus sonhos e necessidades e me amar incondicionalmente.

Agradeço a todos os professores e servidores do IFPI do Campus Parnaíba, que tornaram possíveis a minha plena formação e aprendizagem neste curso, em especial a meu orientador, Bruno Pires Sombra.

Agradeço a minha família, amigos, e todos os outros que me ajudaram a tornar-me a pessoa que sou hoje.

Por fim, agradeço em especial todos os colegas de curso que tive, pois sem a nossa união, eu teria muito mais barreiras no meu trajeto, e a minha noiva, que me motivou a perseverar por um trabalho do qual posso me orgulhar.

“Se você quer construir um navio, não chame as pessoas para juntar madeira ou atribua-lhes tarefas e trabalho, mas sim os ensine a desejar a infinita imensidão do oceano.”

Antoine de Saint-Exupéry

Resumo

Os jogos eletrônicos atuais, principalmente os títulos de grande sucesso, evoluíram de forma excepcionalmente acelerada em comparação às suas origens. Dentre estas evoluções, destaca-se a sua fidelidade gráfica e funcional cada vez maior em relação ao mundo real. Neste trabalho, foi utilizado um motor de jogo de livre uso, o *Unity 3D*, para investigar o processo de criação de um cenário de jogo que permitisse o uso do conhecimento da Física, especificamente da Mecânica, na construção de um jogo no qual se dirige um veículo em um circuito fechado, com o objetivo de demonstrar um método de ajustes e manipulação do sistema para a finalidade de representar um sistema físico real, de forma simplificada, verificando se isto é possível com o *software* escolhido. Após um breve estudo sobre o histórico da Física nos jogos eletrônicos desde seu nascimento e apresentação do conhecimento necessário, isto foi realizado através do cálculo físico e uso de valores reais, para comparar e substituir no *software* de modo que produzisse o comportamento desejado.

Palavras-chave: Jogos Eletrônicos. Unity 3D. Física.

Abstract

The current vídeo games, especially the highly successful titles, have evolved exceptionally fast when compared to their origins. Among these evolutions, their graphic and functional fidelity to the real world stands out. In this work, a freeware game engine, *Unity 3D*, was used to investigate the process of creating a game scene that allowed the use of the knowledge of Physics, specifically of Mechanics, in the construction of a game in which a vehicle is driven in a closed circuit, with the objective of demonstrating a method of adjustments and manipulation of the system for the purpose of representing a real physical system, in a simplified way, verifying whether this is possible with the chosen software. After a brief study of the history of physics in video games since its birth and presentation of the necessary knowledge, this was accomplished through physical calculations and use of real values, to compare and replace in the software so that it produced the desired behavior.

Keywords: Video Games. Unity 3D. Physics.

Lista de Ilustrações

Lista de Tabelas

Lista de Símbolos

τ	Torque
ρ	Densidade do ar
μ	Coeficiente de atrito
θ	Ângulo

Sumário

1 INTRODUÇÃO

Com a rápida evolução da tecnologia de computadores nos últimos anos, a capacidade de processamento e geração de dados, imagens, vídeos e outros aumentou imensamente em relação às décadas passadas. Por exemplo, a renderização 3D, isto é, a formação de uma imagem ou sequência de imagens na tela (2D) a partir de um processo de modelagem em 3D, somente começou a ser utilizada por programas de PC em meados dos anos 90, menos de 25 anos atrás, apesar de ser algo que pode ser considerado indispensável para a representação realista do mundo em que vivemos.

Dentre todas as aplicações possíveis da modelagem 3D, as mais exigentes são aquelas que usam a formação de imagem ou vídeo em tempo real, como simuladores e *video games* (em inglês, jogos eletrônicos). Este último foi bastante impactado pelo avanço da tecnologia de 3D, impulsionando a busca por gráficos que cada vez mais representam o mundo real e aprimorar a imersão dos jogadores, radicalmente afetando o mercado de entretenimento dos anos seguintes.

Na era presente, as placas de vídeo já possuem capacidade de processamento e armazenamento mais de mil vezes maior que as primeiras placas de vídeo 3D, como a *S3 ViRGE* (lançada em 1996), o que, resumindo, significa que os modelos possuem muito mais polígonos na sua composição, mais quadros de movimento por segundo, e melhor iluminação e sombreamento, dando muito mais detalhe à sequência de imagens gerada pelos *video games*, o que os torna cada vez mais indistinguíveis de uma fotografia ou vídeo gravado de objetos e localidades reais. Estas produções também dependem da aplicação de fenômenos físicos para resultar em uma imagem fiel, principalmente fenômenos ópticos, mas estes não são parte do âmbito deste trabalho.

Mas de nada adianta reproduzir a imagem da realidade se estes objetos não se comportarem de forma similar à realidade. Como representar estes objetos da mesma forma que se movimentariam no mundo real? Nos primeiros jogos 2D ou semi-2D, o movimento de objetos ou efeitos em geral ocorre através de regras programadas pelos criadores para este jogo. Estas regras, programadas para simular situações que se comportem de modo consistente e interativo, são chamadas de *game physics* (física de jogos), e incluem diversas reproduções do

comportamento de corpos nas áreas da Física, como a Mecânica Clássica, o estudo de colisão de objetos, dentre outras. A partir dos jogos 3D em diante, no entanto, isto se torna uma tarefa monumental, com inúmeras partes e possibilidades de interação e movimento. Então, começam a ser utilizadas as *game engines* (motores de jogo), que são, de forma resumida, um "universo" com suas próprias regras de funcionamento, especialmente físico, para servir de base para a criação de jogos.

O presente trabalho buscou conhecer o funcionamento de um destes sistemas, verificando os métodos de transferência de teorias físicas para a solução de problemas e limitações de *software*, para enfim aplicá-los na criação de um sistema físico simulado no jogo, constituindo-se assim um exercício prático de uso da Física, apresentando um sistema físico normalmente oculto ao público em geral, mas de grande complexidade e importância para uma experiência imersiva e impressionante para os jogadores.

Com este propósito em mente, o objetivo central do trabalho foi verificar se a simulação da Física do *software* escolhido adequadamente segue as leis e teorias da Mecânica Clássica após os ajustes necessários, o que foi feito a partir da comparação dos resultados obtidos na simulação pelo programa com aqueles obtidos pela aplicação teórica das equações e fórmulas físicas em diversas situações propostas. Para o desenvolvimento do jogo, foi utilizada a plataforma *Unity 3D*, desenvolvida pela Unity Technologies, onde foi criado o cenário de teste a partir de um pacote de simulação de veículos disponibilizado pela empresa.

Sendo a criação de um jogo realista a reprodução do mundo, há uma imensa gama de fenômenos físicos, muitos dos quais são interações mais complexas e minuciosas, como a suspensão dos carros, por exemplo, ou a danificação dos veículos por colisões. No entanto, este projeto limitou-se à reprodução de fenômenos que podem ser diretamente estudadas pelas equações da Mecânica, mas ainda assim são partes essenciais da representação do movimento de um veículo.

Este trabalho encontra-se dividido em quatro seções: A Física no contexto dos jogos, Fundamentação Teórica, Criação do cenário de jogo, e Resultados, Ajustes e Discussões.

Na primeira seção, foi abordada a presença da Física para o desenvolvimento de jogos de forma geral, estudando seu uso frente aos avanços dos jogos eletrônicos ao longo da história recente.

A segunda seção traz a fundamentação teórica, com a base de conhecimento da Física necessário para a compreensão dos processos utilizados neste trabalho, como a Mecânica Clássica, dinâmica de veículos, colisões, entre outros.

A terceira seção trata sobre os passos necessários para o estudo das propriedades e fenômenos físicos atuantes sobre o sistema de jogo, o que inclui a análise de todas as variáveis, objetos e componentes do sistema utilizado, e a construção do circuito no qual os testes do trabalho foram aplicados.

Na quarta e última seção, o cenário de jogo criado foi executado, primeiramente para ajustes e manipulações dos componentes para que o comportamento inicial do veículo seja adequado ao esperado, e em seguida para a aplicação do objetivo do trabalho de fazer a comparação dos resultados obtidos na simulação e na teoria, em três situações de uso em uma pista: Em uma reta, em subidas e descidas, e finalmente, em uma colisão, observando se ele cumpre o propósito do trabalho de representar um veículo real onde possível através da comparação do cenário executado e cálculos e equações da Mecânica, e caso contrário, propor alternativas ou discutir o comportamento resultante.

2 A FÍSICA NO CONTEXTO DOS JOGOS

A Física se encontra presente nos jogos eletrônicos mesmo desde quando elas eram simples jogos 2D, sendo uma importante contribuição para uma boa experiência de jogo. Portanto, para compreender a situação e estado da arte da aplicação das leis da Física na atualidade, é necessário fazer um estudo da história de seu uso desde o nascimento dos primeiros jogos eletrônicos nas décadas passadas. (HENRY et al, 2008).

Durante as primeiras épocas de desenvolvimento dos computadores, os jogos eletrônicos em geral representavam apenas demonstrações das capabilidades do computador, ou utilizados para treinamento ou objeto de pesquisa, pois, sendo o computador um investimento multimilionário, as comunidades acadêmicas e militares, responsáveis por este desenvolvimento, não tinham o entretenimento em mente como objetivo (SMITH, 2014).

Uma exceção notável a este regra foi o jogo *Tennis for Two*, sendo o primeiro jogo de computador projetado especificamente para entretenimento. Criado em 1958 por William Higinbotham, ele consistia de uma tela de osciloscópio, na qual um pequeno ponto servia como bola de tênis, e esta era arremessada com controles de alumínio pelos jogadores. Ele não foi, porém, criado para distribuição comercial, mas para que, além de servir de entretenimento dos pesquisadores, servisse demonstrar que os trabalhos científicos tinham relevância para a sociedade. Assim, o seu criador acidentalmente iniciou a indústria gigante de jogos eletrônicos que conhecemos hoje. Mesmo com o grande sucesso que este jogo teve na sua exibição ao público, nas quais visitantes podiam testar o jogo na mostra anual do *Brookhaven National Laboratory*, em Nova York, ele foi desmontado cerca de um ano depois (KALNING, 2008).

Durante esta mesma época, no Massachusetts Institute of Technology (MIT), foi criado um sistema de computação interativo que podia ser acessado por quase qualquer indivíduo associado à esta instituição, não sendo limitado à pesquisadores trabalhando em projetos específicos. Inevitavelmente, havia certos usuários experientes de computadores que estavam mais interessados em se divertir do que realizar as pesquisas para os quais os computadores eram disponibilizados. O resultado de suas atividades foi o jogo *Spacewar!*. Este foi um dos primeiros a ser

bem-sucedidos em utilizar as leis da Física para criar um jogo bem-recebido e de possibilidade de distribuição. (HENRY et al, 2008).

Desenvolvido em 1962 por Steve Russell e colaboradores, este jogo, diferentemente do anterior, foi um software programado para um computador, o PDP-1, sendo assim possível sua distribuição para outras máquinas. Ele constitui-se de um plano bidimensional onde duas espaçonaves, controladas por dois jogadores diferentes, batalham entre si lançando torpedos ao pressionar um botão. Neste cenário, há diversas estrelas que possuem atração gravitacional que afeta as naves, que podem interferir na trajetória das naves, ou em pior caso, capturá-las no seu poço gravitacional. Elas obedecem as leis do movimento Newtoniano, como a lei da inércia, mas os torpedos não. (GRAETZ, 1981).

Uma década depois, em 1972, foi lançado um dos primeiros jogos eletrônicos de acesso ao público em geral, intitulado *Pong*. Baseado em um jogo que seu criador, Allan Alcorn, havia testado em uma visita a *Magnavox Profit Caravan* na Califórnia, ele é um jogo de tênis para dois jogadores, em que duas plataformas em cada lado da tela podem se movimentar para interceptar a trajetória de uma bola, assim retornando-a ao oponente. Seu ângulo de retorno dependia do ângulo e posição da plataforma na qual ela impactava, mas sua velocidade não reduzia, pelo contrário, ela gradativamente aumentava para elevar o nível de dificuldade do jogo com o tempo (KENT, 2001).

Enquanto que a maioria dos desenvolvedores de jogos na época criavam jogos em ambientes surrealistas ou abstratos, uma companhia lançou um jogo que destacou um protagonista humano: A Nintendo, com o jogo *Donkey Kong*. Apesar de ainda bastante simplista, a Física deste jogo e seus sucessores criam um ambiente de jogo com uma sensação muito mais realista, em que o protagonista sofre a ação do atrito, momento, gravidade, e diversos outros aspectos que tornam a experiência muito mais intuitiva. O exemplo mais popular é o jogo *Super Mario Bros.*, lançado em 1985, protagonizando um personagem que é capaz de pular e andar/correr. Se ele estivesse correndo ao invés de andando antes do pulo, ele consegue pular uma distância maior. Seu movimento é acelerado ou desacelerado gradualmente dependendo das circunstâncias (HENRY et al, 2008).

Durante o avanço destes tipos de jogos, que rapidamente viam mais e mais novas funções e características com os novos títulos lançados a partir daí, outros desenvolvedores focavam em outro aspecto: A exibição de imagens 3D. No entanto,

os consoles de jogos eletrônicos neste tempo ainda não haviam poder de processamento suficiente para gerar imagens 3D reais ainda. Era possível, no entanto, criar a ilusão de 3D através de certas técnicas utilizando imagens 2D, criando assim jogos conhecidos como 2.5D.

Os primeiros jogos a aplicar esta metodologia utilizavam um processo chamado *ray-tracing*. Da Silva (1994) explica, sucintamente, que este é um algoritmo que traça raios que partem do observador (neste caso, a câmera de jogo) e avançam em linha reta até encontrar um pixel do objeto a ser desenhado. Dependendo da distância percorrida, este pixel recebe uma cor definida, de modo que esta simule o sombreamento que teria à essa distância caso fosse um ponto em um objeto 3D real. Repete-se então este processo para cada pixel até a imagem final ser construída.

Um destes pioneiros foi o jogo *3D Monster Maze*, desenvolvido por Malcolm Evans em 1981 para computadores. Ele consiste em um labirinto com um monstro, o Tiranossauro Rex, que persegue o jogador enquanto este tenta escapar deste labirinto. O diferencial deste em relação à outros da mesma época é que o ponto de vista do jogador é em primeira pessoa. Assim como diversos outros jogos de conceitos inovadores apresentados anteriormente, este começou como simples treinamento de programação e exercício de teste das capabilidades do computador, culminando em um produto inesperadamente popular (KROUWEL, 2007). Na figura 1, que mostra uma captura de tela durante o jogo, é possível ver a diferença de construção das paredes, onde as paredes mais distantes são descoloradas comparadas às mais próximas.

Figura 1: Captura de tela do jogo 3D Monster Maze. Fonte: Vassilii Khachaturov

Como este processo necessita gerar cada pixel do objeto um a um, múltiplas

vezes por segundo devido à mudança constante de perspectiva inerente aos jogos eletrônicos, ele necessita de relativamente alto poder de processamento para criar cenários de gráficos complexos e detalhados. Para contornar este problema, a empresa desenvolvedora de jogos eletrônicos Nintendo utilizou uma tecnologia diferente para criar uma experiência pseudo-3D. Intitulada *Mode* 7, ela é simplesmente um artifício que implementa imagens 2D de forma rotacionada tal que elas se estendam ao plano de fundo, e permitindo que objetos movam-se “através” da tela, diminuindo ou aumentando seu tamanho, dando a ilusão de perspectiva. (HENRY et al, 2008).

O último passo na história dos jogos eletrônicos antes da criação de um jogo verdadeiramente 3D, mas que ainda assim foi um dos predecessores mais importantes para a popularização deste tipo de tecnologia, foi o lançamento do jogo *Wolfenstein*. Apesar de ainda possuir interações físicas primitivas, ele trazia uma experiência ainda mais imersiva com uma aprimoração do método *ray-tracing*, o *ray-casting*, que agora permitia a aplicação de texturas sobre colunas inteiras de pixels e redimensionamento de imagens, otimizando e acelerando o processo (PERMADI, 1996).

Somente em 1996 foi criado o primeiro jogo eletrônico amplamente popular que utilizava gráficos 3D, com o lançamento de *Quake,* pela *id software*. Neste jogo, tanto os ambientes e os personagens eram constituídos de modelos 3D poligonais, permitindo a aplicação da Física em um espaço tridimensional. Alguns exemplos incluem o recuo de objetos atingidos por uma explosão, a fragmentação de inimigos destruídos e espalhados no cenário pelo impacto, e movimentos mais realistas. A criação de um sistema físico geral para controlar os movimentos gerais do jogo levaram inclusive os jogadores a movimentar-se de forma inesperada pelos desenvolvedores, ressaltando a enorme liberdade e flexibilidade no mundo dos jogos eletrônicos nunca antes vista (HENRY et al, 2008).

Tal sistema, chamado de *game engine*, ou motor de jogo, é o que torna esta experiência possível. Devido à grande popularidade dos primeiros jogos 3D utilizando complexos motores de jogo, criados especificamente para estes, este começou a ser um processo comum a partir de então – criar um motor de jogo robusto para a montagem do conteúdo do jogo em si sobre ele. Porém, justamente devido à esta complexidade, a dificuldade e custo de desenvolvimento de jogos aumentou significantemente. A solução lógica, sendo um motor de jogo uma base, é

de reutilizá-la para outros jogos. Além disso, a partir daí, surgiram empresas que especializavam não em criar jogos, mas simplesmente criar estes motores, para então comercializá-las para os desenvolvedores. Estas empresas são conhecidas como *middleware providers* (WARD, 2008).

Ambas as escolhas são comuns; em geral, desenvolvedores que têm como objetivo um sistema fácil de manipular e de baixo custo procuram motores de jogo pré-fabricados, para ter certeza que a construção do jogo será consistente e com mínima solução de problemas. Alguns exemplos de softwares deste tipo, na atualidade, são o *Unity*, *OGRE*, e *Torque*. No entanto, estes motores, por serem feitos para uso geral, não possuem a especialização e flexibilidade desejadas por empresas de jogos de alto nível. Mesmo com a possibilidade de ser confrontado com um longo e problemático período de desenvolvimento, portanto, diversas empresas ainda buscam criar seu próprio motor de jogo para criar algo realmente inovador (WARD, 2008).

O primeiro foco do desenvolvimento da tecnologia 3D foi a aprimoração dos gráficos, inclusive com a adaptação de jogos 2D para o formato 3D. Inicialmente, a maioria dos produtores de jogos não houveram grande êxito nestas tentativas. Foi o jogo *Super Mario 64*, da Nintendo, que liderou e serviu como marca de referência para todos os jogos 3D desenvolvidos nos anos seguintes, com um controle de câmera dinâmico de 360° (GERSTMANN, 2006). Este processo culminou no jogo *Half-Life 2*, da empresa *Valve*, em 2004. Contando com um dos sistemas físicos mais avançados da época, incluindo simulações precisas da gravidade, peso e massa, magnetismo e flutuabilidade em quase todos os objetos dos cenários, ele marcou o ponto na história dos jogos eletrônicos em que a corrida por gráficos mais poderosos ficou em segundo plano em relação à interatividade com o mundo e os objetos (HENRY et al, 2008).

Isto não significa, porém, que toda tentativa de criar um jogo realista através do uso da Física é bem-sucedida, porém. Muito antes do *Half-Life 2*, em 1998, um jogo altamente ambicioso foi lançado – o *Jurassic Park: Trespasser*, da *Electronic Arts*. Com o objetivo de ser um jogo revolucionário, ele tentou criar uma experiência ultra-realista: Todos os movimentos seriam controlados por interações físicas, ou seja, sem animações pré-programadas; ele não possuía HUD (*heads-up display,* informações relevantes ao jogador mostradas na tela a todo momento); o método de controle do jogo consistia em uma mão virtual que representava a mão da

personagem; um mundo aberto e extenso; dentre outros (WYCKOFF, 1999).

Ele foi, no entanto, um fracasso: os controles eram complexos e irritantes, as simulações físicas tinham erros constantes, como na representação de atrito e colisões, e o jogo frequentemente apresentava lentidão. Apesar de ser bem-sucedido em criar algo novo e interessante, os sistemas computacionais desta época simplesmente não estavam preparados para lidar com um projeto deste tamanho, e, principalmente, ele falhou em criar um jogo divertido. Jogos mais recentes, como *Oblivion* e *Crysis*, tiveram sucesso onde aquele não teve simplesmente devido aos avanços tecnológicos dos anos seguintes, aplicando conceitos similares (HENRY et al, 2008).

É possível ver, então, que não basta aplicar a física da forma mais aderente à realidade possível: é necessário observar tanto as limitações do sistema com que se trabalha, como ter o bom-senso de não permitir que isto torne o jogo, de modo geral, menos satisfatório para os consumidores.

3 FUNDAMENTAÇÃO TEÓRICA

Nesta seção, apresentam-se os conceitos essenciais para o estudo do movimento de um veículo, incluindo, além da Física, uma descrição do programa utilizado para este propósito.

3.1 A mecânica clássica

A mecânica clássica é a parte da Física que estuda o movimento, as variações de energia, e as forças atuantes sobre um corpo, e constitui todo o estudo da Mecânica existente antes da Mecânica Relativista postulada por Albert Einstein (1879 – 1955). Ela envolve a Mecânica Newtoniana, Lagrangeana e Hamiltoniana (DE AGUIAR, 2011). Segundo Taylor (2013), essas duas últimas são equivalentes à Newtoniana, mas seu propósito principal é tornar diversos problemas muito mais simples de serem resolvidos. A Mecânica Clássica continua a ser estudada e utilizada mesmo após o desenvolvimento da Mecânica Relativista, pois, afirma ele, que aquela é uma boa aproximação das equações desta, mais gerais, quando as velocidades envolvidas são baixas (muito menores que a velocidade da luz).

Sendo assim, ela poderá ser utilizada para o estudo do movimento de veículos terrestres comuns. Tomando como base a Mecânica Newtoniana, Neto (2004) divide-a em duas partes: Na primeira, estuda-se os conceitos independentes das forças que regem o movimento, chamada de Cinemática; na segunda, são apresentadas as Leis de Newton e como elas demonstram os princípios da Dinâmica.

3.1.1 Cinemática

O movimento nada mais é que a mudança de posição de um objeto ao longo do tempo. A figura 2, a seguir, exemplifica uma situação como esta. A partir dela, é possível introduzir os conceitos básicos para o estudo da Cinemática. Observa-se que ela representa a trajetória de uma partícula movendo-se em um sistema ortogonal de três dimensões x, y, z. Este movimento descreve uma curva em relação ao referencial escolhido (neste caso, os eixos ortogonais), iniciando-se em um tempo t no ponto A e culminando no ponto B, num instante de tempo $t + \Delta t$.

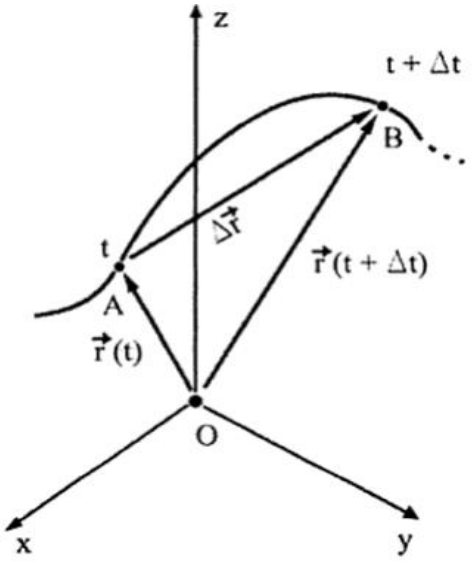


Figura 2: Trajetória de uma partícula. Fonte: Neto (2004).

Define-se a velocidade média como a razão da distância total percorrida com o tempo gasto para tal. Utilizando o caso geral representado pela figura 2, a variação de tempo total seria a diferença o tempo final $t + \Delta t$ e o inicial t, ou seja, Δt. Para a distância total percorrida representada, de $\Delta\vec{r}$, a velocidade média é:

$$\overrightarrow{v_m} = \frac{\Delta\vec{r}}{\Delta t} \tag{3.1}$$

Esta velocidade, no entanto, não representa as velocidades que a partícula obteve em cada instante deste movimento, especialmente considerando que ela percorreu a curva AB, e não o percurso $\Delta\vec{r}$. É possível aproximar este cálculo para variações de espaço cada vez menores, tal que um deslocamento $\Delta\vec{r}$ aproxime-se de parte do percurso curvilíneo. A figura 3 esclarece este procedimento.

Figura 3: Deslocamentos menores entre A e B. Fonte: Neto (2004).

Caso seja escolhida uma seção infinitamente pequena, pode-se utilizar o limite para calcular a velocidade da partícula nele:

$$\vec{v}(t) = \lim_{\Delta t \to 0} \frac{\Delta\vec{r}}{\Delta t} = \frac{d\vec{r}}{dt} \tag{3.2}$$

Conclui-se, então, que a velocidade é a derivada do vetor posição em relação ao tempo. Para uma distância infinitesimal, o tempo também deve sê-lo, portanto, esta velocidade é exatamente a daquele instante medido, intitulando-a assim como velocidade instantânea. A partir da figura 3, é possível observar que esta velocidade é tangente á curva da trajetória em cada ponto.

De maneira análoga, define-se a aceleração. Sendo a aceleração um valor que altera a velocidade em função do tempo, há valores iniciais e finais da velocidade tal que resulte em uma variação $\Delta\vec{v}$ desde um tempo inicial t até o tempo final $t + \Delta t$. A aceleração média é, então, dada por:

$$\overrightarrow{a_m} = \frac{\Delta\vec{v}}{\Delta t} \tag{3.3}$$

e a aceleração instantânea da partícula num instante t é dada por:

$$\vec{a}(t) = \lim_{\Delta t \to 0} \frac{\Delta\vec{v}}{\Delta t} = \frac{d\vec{v}}{dt} \tag{3.4}$$

A aceleração pode ser variável com o tempo assim como a velocidade, como denotado nas equações (3.3) e (3.4). Considerando um caso em que ela é um valor constante, é possível reescrever a eq. (3.2) da seguinte forma, para:

$$\frac{d\vec{r}}{dt} = v_0 + at \tag{3.5}$$

Isolando o termo $d\vec{r}$ na eq. (3.6), pode-se fazer a integração desta. Utilizando-se limites para um caso geral, ela resulta na equação a seguir:

$$\int_{r_0}^{r} d\vec{r} = \int_{t_0}^{t} [v_0 + at]\, dt \tag{3.6}$$

Resolvendo a integral e atribuindo-se os limites, a eq. (3.6) resulta na seguinte fórmula:

$$\vec{r}(t) = r + v_0(t - t_0) + \frac{1}{2}a(t - t_0)^2 \tag{3.7}$$

Esta fórmula, que relaciona a posição de um objeto com as variáveis que regem seu movimento, é conhecida como a equação do movimento retilíneo uniformemente variado. Caso necessário obter este valor sem utilizar o tempo, é possível combinar a equação (3.7) com a (3.5) para eliminar o tempo, resultando na chamada fórmula de Torricelli.

3.1.2 Dinâmica

A Dinâmica é a parte da Mecânica que estuda o movimento dos corpos e as causas deste movimento. Nussenzveig (1981) afirma que os princípios da Dinâmica

foram formulados por Isaac Newton e Galileu Galilei, mas foi o primeiro que enunciou-os da forma utilizada na Física atual, conhecida como as Leis de Newton. Estas três leis estão descritas a seguir.

3.1.2.1 Primeira Lei de Newton

Conhecida como a Lei da Inércia, ela é definida pelo seguinte enunciado de Isaac Newton: *Uma partícula livre de interações está em repouso ou movimento retilíneo de velocidade constante.* Interação, define Neto (2004), é a atuação de forças não-fictícias sobre a partícula. É importante fazer esta distinção, pois há forças que não provém dos quatro tipos fundamentais de interação – gravitacional, eletromagnética, forte e fraca – que surgem apenas quando o referencial não é inercial, como exemplo, aquela sentida por um passageiro de um veículo acelerado em curva. Estas forças são chamadas de fictícias, pois só existem em certos referencias.

Mas o que é o referencial inercial? É justamente isso que a Primeira Lei busca definir. A conclusão trivial de que um objeto na qual nenhuma força age tem velocidade constante pode ser retirada da Segunda Lei, a seguir, mas a Primeira Lei torna-se necessária para criar a condição em que o referencial seja livre de forças fictícias, o que permite que os princípios da Dinâmica de Newton sejam aplicáveis.

3.1.2.2 Segunda Lei de Newton

Para modificar a quantidade de movimento de um corpo, é necessário que este corpo interaja com outros. Esta interação, como visto, é regida pelas quatro forças fundamentais, que aplicam uma alteração da quantidade de movimento proporcional à sua intensidade, na mesma direção em que se aplica a força. Newton, definindo o que hoje é conhecido como a sua Segunda Lei, afirma que a resultante da soma de todas as forças atuantes sobre o corpo ($\overrightarrow{F_R}$) é igual à sua variação da quantidade de movimento no tempo. Esta definição pode ser escrita da seguinte forma:

$$\overrightarrow{F_R} = \frac{d\vec{p}}{dt} \tag{3.8}$$

onde $\vec{p}$ representa a quantidade de movimento, também conhecida como momento linear, definida como:

$$\vec{p} = m\vec{v} \tag{3.9}$$

estabelecendo, assim, a Segunda Lei de Newton como:

$$\overrightarrow{F_R} = \vec{v}\frac{d\vec{m}}{dt} + m\frac{d\vec{v}}{dt} \tag{3.10}$$

Para a resolução de problemas comuns, onde a massa m mantém-se constante, a (3.10) torna-se:

$$\overrightarrow{F_R} = m\vec{a} \tag{3.11}$$

É importante notar que esta lei não descreve as forças em si, isto é, sua natureza e origem. Assim, a Segunda Lei não é suficiente para descrever o movimento da partícula por completo, necessitando seu estudo a partir de todo o conhecimento Físico geral.

3.1.2.3 Terceira Lei de Newton

Newton, através da sua Segunda Lei, aborda o comportamento mecânico de um corpo sob a ação de forças. Sendo estas forças resultado de uma interação, deve haver, no mínimo, um outro corpo envolvido nela. Como estes se relacionam nesta interação? Através de sua terceira e última lei, também conhecida como Lei da Ação e Reação, ele descreve o que ocorre. Nas palavras de Neto (2004), “Quando duas partículas interagem, a força numa delas possui o mesmo módulo, direção e sentido contrário à força que atua na outra”. Isto significa que, qualquer que seja a origem da força atuante sobre um objeto, o outro também sente a mesma força, mas oposta, de modo que esta não surja “do nada”, mas sim mantendo o equilíbrio de forças em todo o sistema referencial.

É importante notar que este par de forças requer a interação entre dois corpos diferentes. Assim, a Terceira Lei não se aplica no estudo das forças atuantes em apenas um corpo, ou seja, uma força aplicada em um objeto analisado não necessariamente possui uma força igual contrária para equilibrá-la, e se possuir, ele não é um par ação-reação. Por exemplo, num objeto em repouso sobre uma mesa, a força Peso é igual em valor e de sentido contrário á força Normal, mas este não é um par ação-reação. Finalmente, vale notar que forças fictícias não possuem um par, mais uma vez reforçando a importância da Primeira Lei e do estudo em um referencial inercial.

Dentre as forças que podem atuar no movimento, há duas forças comuns quase sempre presentes na análise de objetos em situações cotidianas. Elas são a

resistência do ar e o atrito, descritas em detalhe nos subcapítulos a seguir. Há ainda forças rotacionais, imprescindíveis no estudo do movimento de um veículo por suas rodas, que são melhor descritas pelo conceito de torque.

Serway (2003) descreve o torque, de símbolo τ, como um vetor que indica a magnitude de uma força rotacional agindo sobre um objeto, aplicando neste uma torção. Ele é definido como o produto vetorial do vetor distância $\vec{r}$ do ponto de aplicação da força e o centro de massa deste com a força rotacional $\vec{F}$, a seguir:

$$\vec{\tau} = \vec{r} \times \vec{F} \tag{3.12}$$

Observa-se que, naturalmente, deve haver um ângulo diferente de zero entre os vetores que determinam o ponto de aplicação e a força para haver rotação, pois caso contrário, esta se torna uma força comum que acelera o objeto linearmente.

3.2 Resistência do ar

A resistência do ar constitui a força atuante sobre um corpo movimentando-se através do ar, sendo assim um exemplo de resistência fluidodinâmica ocorrente em objetos em movimento através de um fluido. Falkovich (2011) afirma que esta força sempre é contrária ao movimento, e possui dois tipos: Laminar e Turbulenta.

No caso laminar, geralmente presente em baixas velocidades, o fluido corre em camadas paralelas e ordenadas, sem misturar-se entre si. Neste caso, a força de desaceleração atuante em um objeto em movimento é dependente da velocidade do objeto e da viscosidade do fluido, que determina a quantidade de atrito gerada.

Em geral, no entanto, casos naturais, incluindo o ar terrestre, possuem fluxo turbulento. Assim, o cálculo da resistência não mais depende da viscosidade do fluido, apenas de sua densidade ρ. A equação (3.12), a seguir, indica o cálculo da força de resistência F nesta situação:

$$F = \frac{\rho C_D A}{2} v^2 \tag{3.13}$$

sendo A a área de contato do objeto na direção do movimento, v a sua velocidade, e C_D o coeficiente de resistência aerodinâmica do objeto. Este último depende do seu formato, e é o principal fator que define a aerodinâmica de um objeto. Isto ocorre porque há diversos tipos de interações entre este e o fluido, das quais as duas mais relevantes para este estudo são a resistência de pressão, devido ao choque do fluido com a superfície do objeto, e a resistência de fricção, que se deve ao atrito do fluido paralelamente ao objeto. A figura 4 ilustra alguns exemplos de objetos e a

resistência que sofrem. Encontrar este coeficiente depende, no entanto, de testes práticos, então será necessário aceitar e utilizar os valores pesquisados para o cálculo desta força.

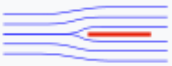
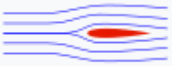
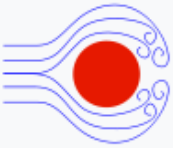
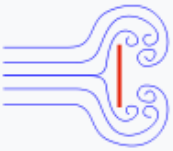
Shape and flow	Form Drag	Skin friction
	0%	100%
	~10%	~90%
	~90%	~10%
	100%	0%

Figura 4: Relação entre formato do objeto e quantidade de resistência de cada tipo atribuído a ele. Fonte: Adaptação de Media Commons

3.3 Força de Atrito

A força de atrito é uma força que atua quando dois corpos fazem contato entre si. Segundo Beer (1996), ela atua quando estes objetos têm tendência ao movimento relativos entre si, sendo originada da rugosidade da superfície de contato dos objetos, adesão entre eles, deformação das superfícies, dentre outros fatores. No nível microscópico, esta força é gerada por interações eletromagnéticas entre os átomos constituintes das superfícies. Ela é sempre paralela à direção do movimento, opondo-se a este, mesmo que apenas iminente. Há diversos tipos de atrito presentes na natureza, mas aqui será discutido apenas o atrito entre sólidos.

Neste caso, a intensidade da força de atrito F_{at} é dada por:

$$F_{at} = \mu N \tag{3.14}$$

com μ representando o coeficiente de atrito entre as duas superfícies, e N a intensidade da força normal sobre o corpo na qual está medindo-se a força de atrito. Resnick e Halliday (2003) afirmam que o valor de μ, além de variar com o tipo e condição dos materiais constituintes das superfícies, também assume dois valores dependendo da situação do movimento. Se o objeto encontra-se apenas na iminência do movimento, isto é, sofrendo forças mas mantendo-se ainda estático,

utiliza-se o coeficiente de atrito estático μ_e. Caso ele já tenha iniciado o movimento, o cálculo da força de atrito é feito com o coeficiente de atrito dinâmico μ_d. Em geral, o coeficiente de atrito dinâmico é menor que o estático, indicando que o atrito é menor quando há movimento.

Na situação específica do atrito entre os pneus de um carro e uma superfície, ele funciona de modo diferente. Aqui, o ponto de atrito fornece uma força de resistência estática que permite que os pneus do carro rotacionem ao invés de deslizar, isto é, ela traciona o veículo para a frente como força de reação à ação da força gerada pelo motor. Assim, contrário ao esperado, a força de atrito é responsável por impulsionar o carro a partir do torque gerado pelo motor do carro. Isto é possível através da deformação sofrida pelo pneu, e em menor grau, a pista.

Apesar disto minimizar a perda de energia e velocidade por atrito, pois o movimento não leva à um arrasto entre as superfícies, ou seja, a força de atrito estático não é contrária ao movimento, Hibbeler (2007) aponta fatores primariamente ligados à inelasticidades dos materiais, incluindo deformação permanente (plástica), que causam uma força contrária ao movimento nesta situação, ainda que muito menor que a força de atrito estático, chamada de resistência ao rolamento. Similarmente ao atrito estático, esta força de resistência F_R é uma função linear dependente da força normal N:

$$F_R = C_{rr}N \tag{3.15}$$

O coeficiente de resistência ao rolamento C_{rr} depende de diversos fatores como textura da superfície, diâmetro das rodas, estado de conservação das rodas, entre outros. Mas assim como o coeficiente de atrito comum, ele possui valores tabelados para diversos casos de uso obtidos empiricamente. Como exemplo, Gillespie (1992) determina o valor de $C_{rr} = 0{,}01$ para um carro comum sobre asfalto.

Em geral, pode-se considerar que um veículo comum para uso diário não desliza, a não ser que esteja freando ou sobre superfície de baixo atrito, como pista molhada ou gelo. Quando isto ocorre, no entanto, a derrapagem do carro é compensada pelo material do pneu, que se estica, aumentando a área de contato e, assim, mantendo a estabilidade do carro a partir do aumento da força de atrito até seu valor máximo. Caso isto não seja suficiente para evitar o deslizamento, o veículo começa a perder contato com a superfície, evitando que o pneu consiga esticar-se para aumentar o atrito, e assim este torna-se o valor básico de contato entre o pneu

relaxado e a pista. Este comportamento é representado pela curva da figura 5.

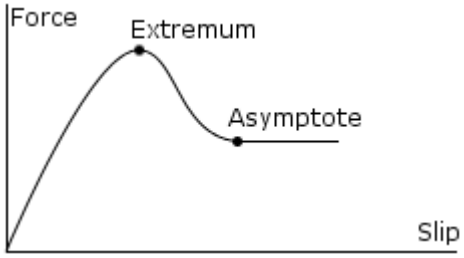


Figura 5: Representação de um gráfico Força de atrito x Derrapagem para um veículo. Fonte: Manual do *Unity* (2018)

3.4 Colisões

O estudo de colisões faz parte da Dinâmica, sendo, em geral, caracterizadas por uma interação entre dois ou mais corpos por um período do tempo relativamente curto, independente da magnitude destas forças. Neste tipo de interação, ao menos um dos corpos move-se em encontro aos outros (TIPLER, 2009). A distribuição resultante do momento, pode se configurar de formas diferentes dependendo do tipo de colisão, que podem ser elásticas, inelásticas, ou parcialmente elásticas.

Nas colisões perfeitamente elásticas e inelásticas, o momento resultante obedece à Lei da Conservação do Momento de Newton, um princípio fundamental que permitiu-o formular suas leis, e eventualmente levou à definição da Lei de Conservação de energia. Aquele pode ser resumido na seguinte expressão:

$$\frac{dp_T}{dt} = 0 \tag{3.16}$$

que indica que o momento, ou quantidade de movimento total em um sistema fechado não sofre mudança ao analisar-se todo e cada objeto participante na interação. A principal diferença entre estes dois tipos apresenta-se na configuração da distribuição de energia cinética. Na elástica, a energia se conserva de forma que objetos transferem sua energias entre si, resultando numa velocidade relativa de afastamento (após a colisão) igual a de aproximação. No caso inelástico, os corpos seguem a movimentar-se como se houvessem se amalgamado em apenas um corpo após a colisão, mas ainda mantendo a quantidade de movimento. Pode-se ver na Figura 6 uma ilustração destes dois tipos de colisões.

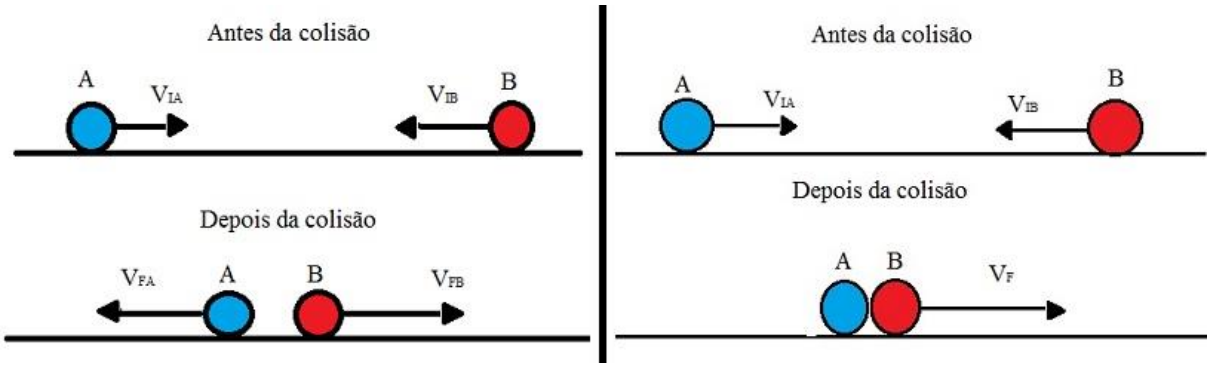


Figura 6: Representação de uma colisão elástica, à esquerda, e uma colisão inelástica, à direita. Fonte: Teixeira (2018).

Estes representam, no entanto, casos ideais em que a dissipação da energia cinética total do sistema não é presente, por fatores como deformações e atrito. Quando isto ocorre, o comportamento dos objetos apresenta-se de forma intermediária aos casos anteriores, caracterizando uma colisão parcialmente elástica (ou parcialmente inelástica). O fator que determina qual a razão entre a o efeito elástico da colisão e o inelástico é chamado de coeficiente de restituição, representado por e. Ele é determinado pela seguinte equação:

$$e = \frac{v_{2f} - v_{1f}}{-v_{2i} - v_{1i}} \tag{3.17}$$

Isto pode ser observado tomando em conta o caso da colisão elástica. É possível afirmar que a soma das velocidades em um objeto deverá ser igual àquelas atuantes no segundo, para uma colisão entre dois objetos. Assim:

$$v_{1i} + v_{1f} = v_{2i} + v_{2f} \tag{3.18}$$

Esta equação pode ser escrita da seguinte forma:

$$v_{2f} - v_{1f} = -v_{2i} - v_{1i} \tag{3.19}$$

$$\frac{v_{2f} - v_{1f}}{-v_{2i} - v_{1i}} = 1 \tag{3.20}$$

Este nada mais é que um caso particular da colisão parcialmente elástica, onde $e = 1$. Nota-se, então, que o coeficiente de restituição é um número entre 0 e 1. Quanto maior este valor, mais elástica é a colisão, e vice-versa. Utilizando a eq. (3.15) aplicada na análise de conservação do momento, podemos finalmente obter as velocidades finais dos objetos após a colisão, dadas pelas equações abaixo.

$$v_{1f} = \frac{(1+e)m_2 v_{2i}}{m_1 + m_2} + \frac{v_{1i}(m_1 - m_2 e)}{m_1 + m_2} \tag{3.21}$$

$$v_{2f} = \frac{(1+e)m_1 v_{1i}}{m_1 + m_2} + \frac{v_{2i}(m_2 - m_1 e)}{m_1 + m_2} \tag{3.22}$$

3.5 Sobre o *Unity 3D*

Unity 3D, ou simplesmente *Unity*, é um software de criação de jogos eletrônicos multi-plataforma, desenvolvido pela *Unity Technologies*. Ele foi desenvolvido por três colegas: David Helgason, Nicholas Francis, e Joachim Ante na Dinamarca a partir de agosto de 2004, após o seu primeiro projeto juntos de um jogo não ser bem-sucedido. Sua primeira versão foi lançada em 6 de junho de 2005. Desde então, o *Unity* sofreu diversas atualizações, e se tornou um dos mais populares e bem-sucedidos *softwares* deste tipo, com seu website reportando mais de 5 bilhões de downloads e 770 milhões de jogadores de jogos criados em Unity, ao final de 2016. Ele possui três versões, de acordo com a necessidade do usuário de recursos mais avançados, sendo uma delas gratuita.

Segundo seus desenvolvedores, seu objetivo era criar um *software* de criação de jogos acessível e com ferramentas profissionais para criadores de jogos amadores, democratizando a indústria de desenvolvimento de jogos (HAAS, 2014). Sua característica notável é de possuir a possibilidade de exportar o produto final para dezenas de plataformas possíveis, incluído plataformas móveis como o Android, o que sem dúvida é parte do sucesso do programa, como indica o seu website, apontando cerca de 2,4 bilhões de dispositivos móveis possuindo jogos feito nele.

O Unity é programado em uma mistura de C++ e C#, em diferentes partes, mas a sua interface de criação de jogos suporta diversas linguagens de programação, incluindo sua própria linguagem específica, o *UnityScript*. Esta é uma linguagem similar ao Javascript, simples de utilizar e compreender, gere grande parte do processo por conta própria, e é a mais utilizada pelos utilizadores do programa. Portanto, ela é recomendada para iniciantes, e apesar de não oferecer o controle preciso que outras linguagens podem proporcionar, ela possui uma vasta gama de exemplos prontos e de assistência disponível (HAAS, 2014).

Utilizar estes recursos é indispensável para acelerar o processo de montagem dos cenários, objetos, e sistemas do jogo para simplificar e agilizar esta parte do desenvolvimento, permitindo focar na análise e utilização dos programas e métodos que introduzem a Física neste projeto. Ao iniciar o *Unity*, é possível encontrar diversos tutoriais e recursos de criação de jogos na aba *Learn* (aprender, em inglês). Dentre eles, será utilizado o recurso *Vehicle Tools*, desenvolvido pela própria

empresa, pois é o que mais se adequa ao propósito deste trabalho. Ele possui um guia em PDF, um modelo de jogo pré-construído, e diversas configurações que facilitam a construção de um protótipo de veículo para iniciar o trabalho.

A empresa também disponibiliza um guia em PDF online de um dos seus componentes principais na elaboração de um jogo com veículos, o *WheelCollider*. Este trata do funcionamento da Física de uma massa sobre rodas, e também será utilizado neste projeto. A figura 7 mostra a interface do *Unity*.

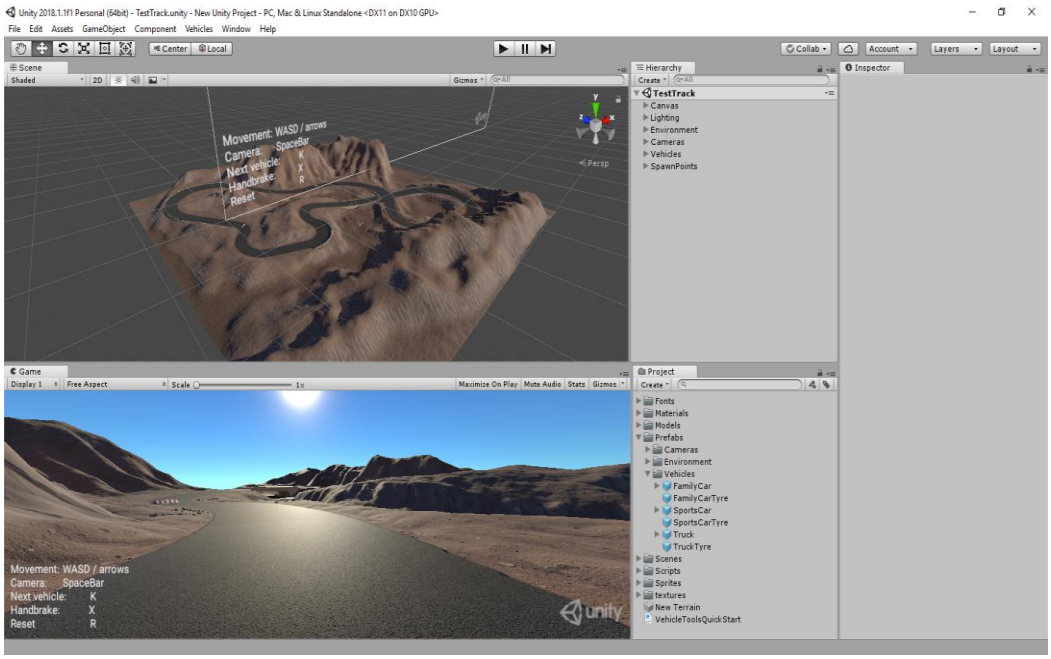


Figura 7: Interface do programa Unity. Na parte superior, encontra-se o editor de componentes, que permite mover, inserir, modificar e visualizar os objetos e a área de jogo, com a janela à direita permitindo editar os valores, componentes, e comportamento destes objetos. Na parte inferior, encontra-se a área de visualização de teste do jogo, com a janela à direita listando os recursos disponíveis para ser utilizados. Fonte: Autoria própria

A partir do *scene* (cenário de jogo) *TestTrack*, que é o cenário carregado no *Unity* acessível ao obter o recurso *VehicleTools* mostrado na figura 7, realizou-se, primeiramente, diversas comparações e análises sobre o uso do programa para a preparação do cenário de estudo a ser construído neste trabalho. Quando a medição do tempo foi necessária, foi utilizado para este propósito o programa *XNote Stopwatch*, desenvolvido pela *dnSoft Research Group*, que permite sobrepor o cronômetro na tela e ativá-lo pelo teclado, permitindo medições mais precisas.

4 CRIAÇÃO DO CENÁRIO DE JOGO

Esta seção é destinada à criação do cenário de jogo proposto neste trabalho, e é dividida em duas partes: Na primeira, detalha-se o uso do software *Unity* na manipulação de valores, fórmulas e sistemas físicos, a partir do cenário *TestTrack*. Na segunda, mostra-se o processo de construção de um novo cenário, onde foi possível simular o comportamento de um veículo real escolhido, de forma simplificada, para tornar preciso o estudo de cada parte.

4.1 Análise do cenário TestTrack

Ao iniciar o *TestTrack*, mostrado na figura 7 da seção anterior, podemos ver o que constitui este cenário: Uma pista, com alguns obstáculos e barreiras nas suas laterais; um terreno montanhoso; o veículo; e finalmente, o objeto que constitui a interface do jogo. Na aba intitulada *Hierarchy*, é possível visualizar as propriedades e os valores destas ao selecionar cada objeto, como mostra a figura 8.

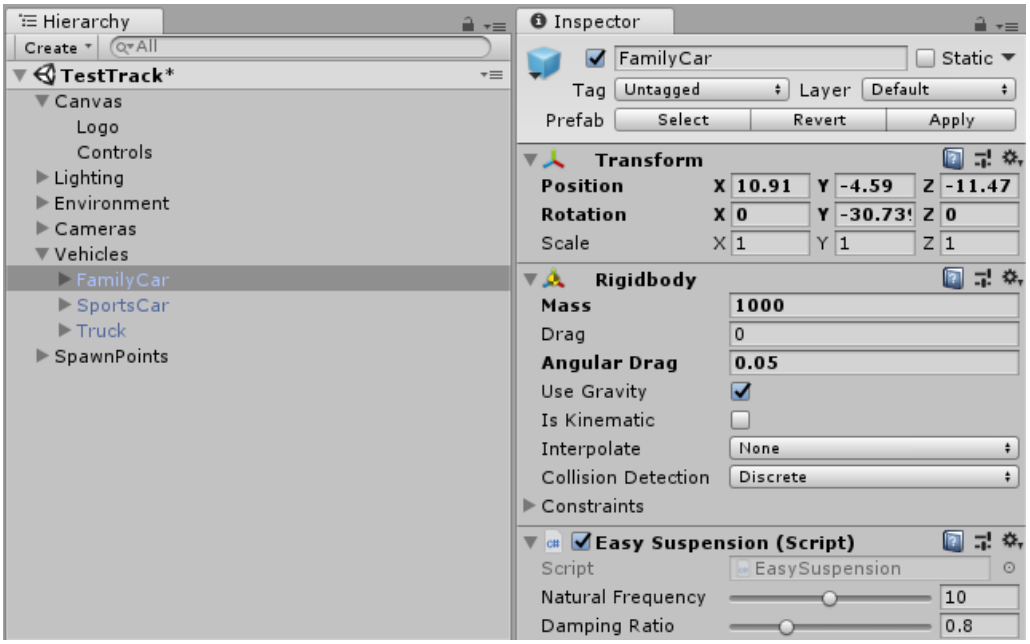


Figura 8: Aba *Hierarchy* e parte da aba *Inspector* do editor do Unity. Fonte: Autoria Própria

Nas subseções a seguir, é feita a análise de cada objeto, observando cada componente destes a partir da aba Inspector (mostrada na figura 8). Todas as definições utilizadas para descrever estes componentes estão disponíveis na documentação fornecida nos diretórios do *Unity*, especificamente em *ScriptReference*, também acessível pelo website docs.unity3d.com

4.1.1 O veículo

Parte central de qualquer jogo eletrônico de corrida, o veículo é o objeto controlável pelo jogador e seus oponentes. A maioria das interações entre objetos o involverá, e seu funcionamento correto e preciso é indispensável para a construção de um bom jogo, portanto, este foi o foco principal deste projeto.

O recurso *VehicleTools* disponibiliza três modelos de veículos pré-fabricados: Um carro comum, um carro esportivo, e um caminhão. Este projeto utilizou apenas o carro comum, da figura 9, por ser o mais adequado à adaptação à um caso real.

Figura 9: Vista frontal e lateral do veículo. Fonte: Autoria Própria

Ao selecionar o carro na aba *Hierarchy*, onde ele é intitulado *FamilyCar*, podemos ver seus componentes físicos. Acima dos componentes, temos as opções:
Tag: Determina qual o tipo de objeto de jogo. Não é necessário para o veículo.
Layer: Manipula características visuais do objeto. A opção *Default* (Padrão) é utilizada para objetos em geral.
Prefab: Permite selecionar opções de objetos pré-fabricados, isto é, modelos prontos.

Os componentes deste objeto são, em ordem:
Transform: Script que aponta e manipula a posição atual do objeto, assim como suas dimensões.
Rigidbody: Permite ao objeto ser influenciado por diversos efeitos físicos. Estes são:
- *Mass*: Análogo à massa física real, é um valor utilizado para cálculo de seu

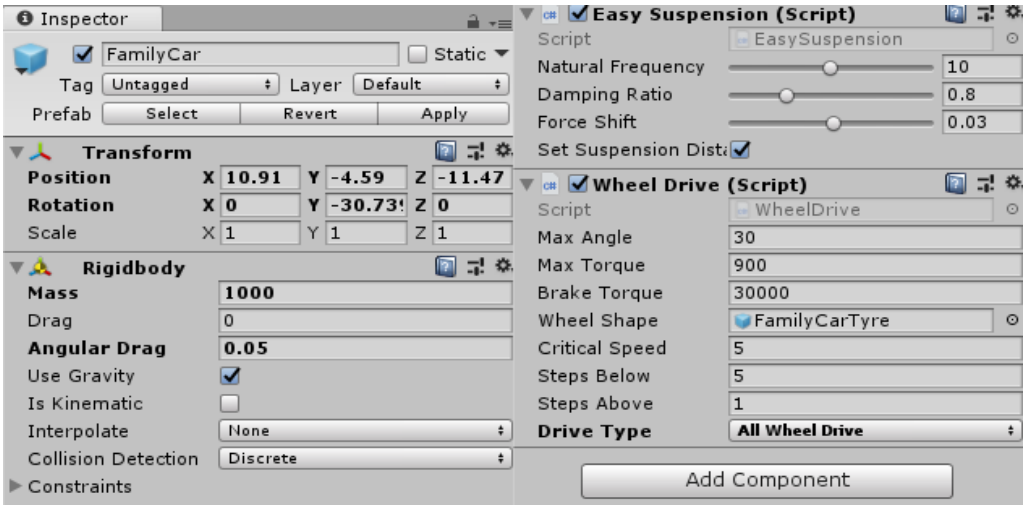


Figura 10: Aba Inspector para o objeto FamilyCar, dividida horizontalmente. Fonte: Autoria Própria

comportamento mecânico.

- *Drag*: Valor da resistência do ar. Na prática, isto indica o valor da força que será aplicada contrária ao movimento do objeto a todo momento.
- *Angular Drag*: Similar ao anterior, mas atuante apenas em rotações.
- *Use Gravity*: Seleciona se o objeto sofre influência da gravidade ou não, o que, na prática, é uma aceleração constante para baixo. Esta opção não permite manipulação do valor, mas análise do script mostra que ele é de 9,81.
- *Is Kinematic*: Indica se o objeto pode ser movido por interações físicas.
- *Interpolate*: Esta opção aproxima o movimento do objeto baseado nas suas posições anteriores e seguintes esperadas, para prevenir movimentos bruscos.
- *Collision Detection*: Possui opções de detecção de colisão entre objetos, caso a opção padrão (*Discrete*) esteja permitindo objetos atravessarem-se.
- *Constraints*: Permite desabilitar o movimento em certas direções.

Destes componentes, já é possível notar que o valor padrão para a resistência do ar é nula, por enquanto. A opção *Interpolate* deve ser desabilitada para prevenir alterações de comportamento do veículo de modo que não siga as leis da física. As outras opções não necessitarão de alteração, pois não são relacionadas à simulação física. Em seguida, temos:

Easy Suspension: Rege o funcionamento da suspensão do carro. São utilizados os valores padrão, por simplicidade.

Wheel Drive: Este é o script que controla o funcionamento e condução do carro. A partir do documento fornecido pelo pacote *VehicleTools*, define-se os valores do *script* como:

- *Max Angle*: Máximo ângulo de rotação das rodas dianteiras.
- *Max Torque*: Torque máximo aplicado pelo veículo sobre as rodas. Vale notar que, quando se trata de torque máximo do motor, este não necessariamente corresponde ao torque transmitido por inteiro às rodas, mas depende do estado da transmissão e a marcha.
- *Brake Torque*: Torque de freio, ou seja, o poder de frenagem do sistema sobre as rodas. Diferentemente do torque do motor, este não resulta em uma força de aceleração aplicada no movimento do veículo, mas sim sobre o movimento de rolamento das rodas. Assim, a força de desaceleração sobre o veículo é resultante, na verdade, do surgimento da força de atrito contrária ao movimento.
- *Critical Speed*, *Steps Below* e *Steps Above*: Número de passos realizados pelo sistema para calcular o resultado das forças aplicadas sobre as rodas a cada interação. É possível escolher dois valores diferentes para velocidades maiores e velocidades menores, separadas pela velocidade crítica (*Critical Speed*).
- *Drive Type*: Define o tipo de direção, entre rodas dianteiras, traseiras, ou ambas.

A manipulação dos valores de passos de computação pode ser analisada através do gráfico do componente de UI, detalhado mais adiante. Em seguida, ao expandir o submenu do objeto, pode-se verificar as partes que o constituem.

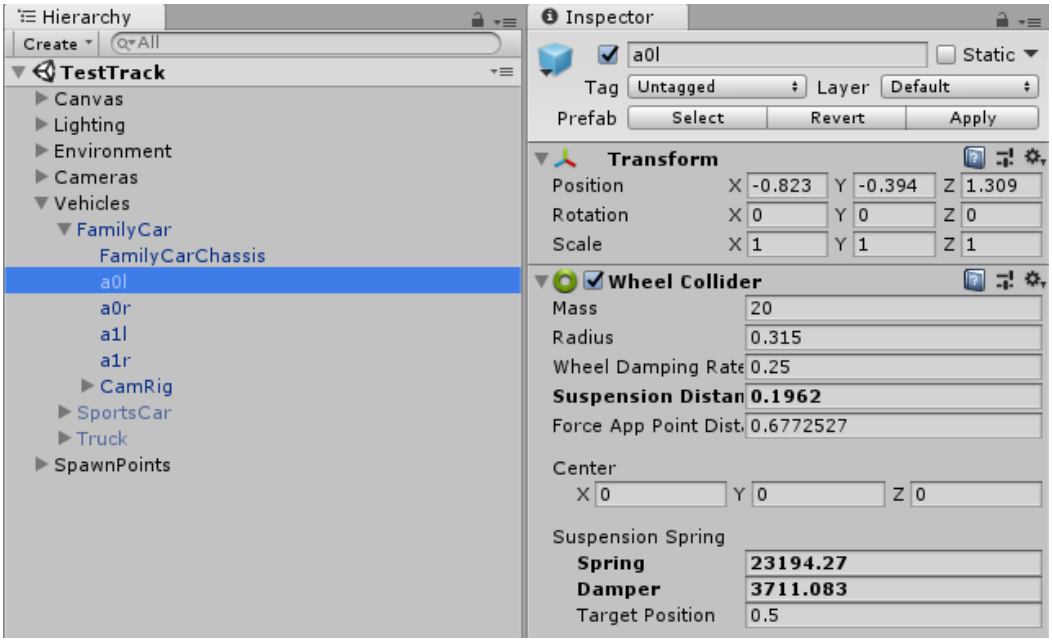


Figura 11: Aba *Hierarchy* com o objeto *FamilyCar* expandido, à esquerda. À direita, parte da aba *Inspector* para a parte a0l deste objeto. Fonte: Autoria Própria

Como ilustrado na figura 11, o objeto FamilyCar possui seis partes. Elas são o chassi, as quatro rodas, e a câmera de visualização do jogo, respectivamente. Os

componentes do chassi limitam-se a dar forma, textura e áreas de colisão para o carro, portanto, é de interesse apenas como este objeto interage com o meio.

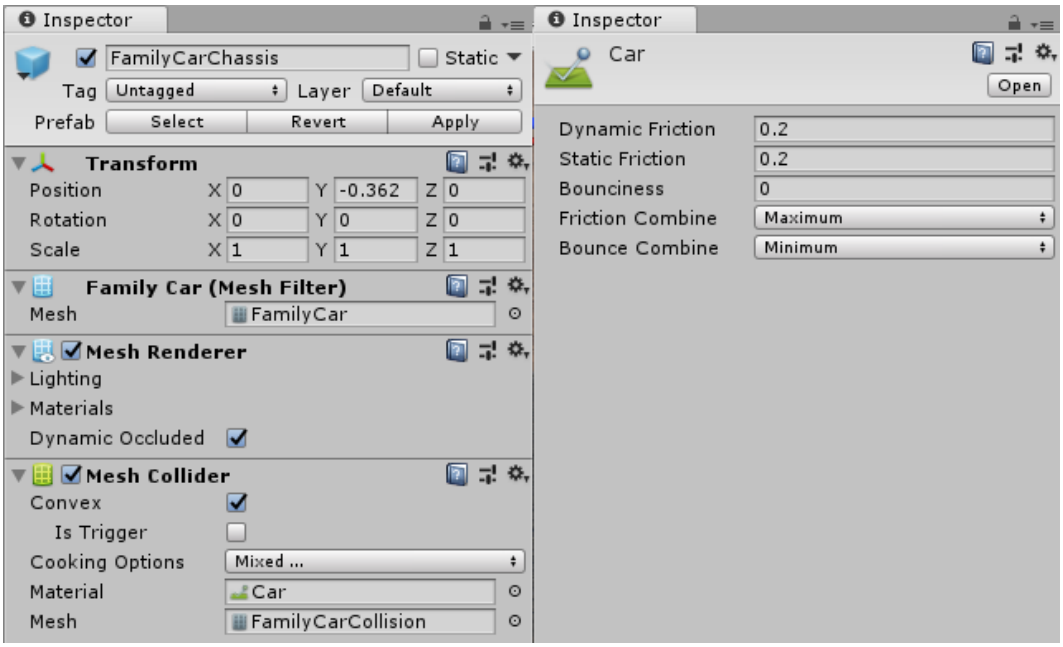


Figura 12: Componentes da parte *FamilyCarChassis*, à esquerda. À direita, propriedades do material *Car* que é aplicado à esta parte. Fonte: Autoria Própria

Através do componente *Mesh Collider*, pode-se aplicar um material ao objeto tal que este modifique as interações de colisão entre este e outros objetos. Na figura 12, pode-se identificar estas propriedades:

Dynamic Friction: Atrito dinâmico, valor de 0,2.

Static Friction: Atrito estático, valor de 0,2.

Bounciness: Coeficiente de restituição, valor 0.

As opções abaixo destes indicam o modo de combinação destes valores com os do objeto com o qual ele colide: *Maximum* torna o maior valor entre os dois o escolhido, enquanto *Minimum* faz o inverso. Nota-se que os valores padrão possuem valores iguais para o atrito estático e dinâmico, e nulo para o coeficiente de restituição de colisão.

Os objetos a0l, a0r, a1l, e a1l, isto é, as rodas do veículo, têm funcionamento idêntico, o que pode ser confirmado ao inspecionar cada uma destas partes, que possuem valores iguais, exceto da posição relativa ao carro. Além do componente que determina sua posição e dimensão, estes contém o *WheelCollider*.

WheelCollider: Este é o script que rege a interação entre as rodas e o terreno. Grande parte do funcionamento destre script é substituído pelo *Wheel* Drive, sendo este um script criado especificamente para aprimorar o controle e

compreensão dos atributos do carro, como cita seu documento de ajuda.

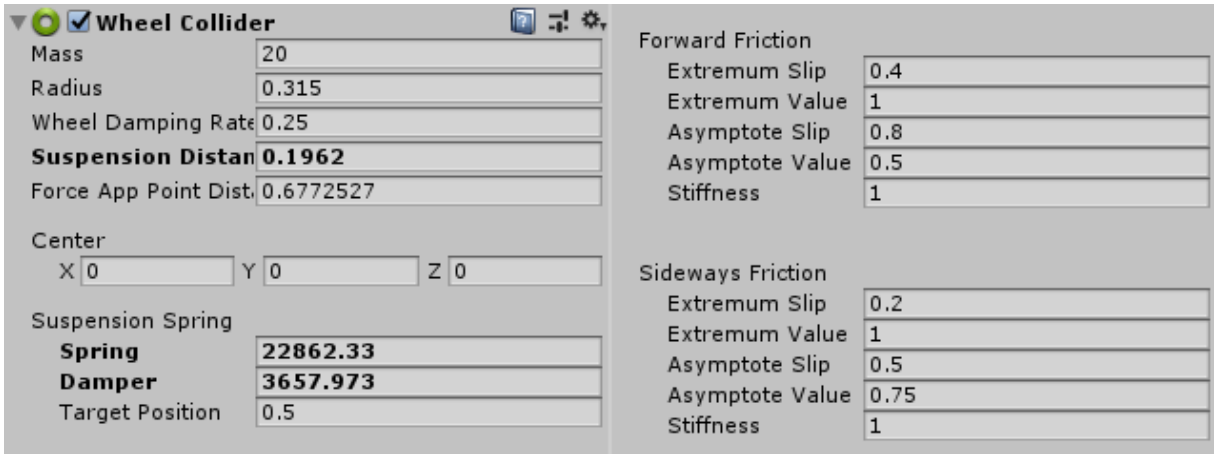


Figura 13: Componente *WheelCollider* do *Unity*. Fonte: Autoria Própria

Das propriedades ilustradas na figura 13, aquelas de interesse são:

- *Mass*: Massa da roda.

- *Radius*: Tamanho do raio da roda.

- *Forward Friction, Sideways Friction*: Diferentemente do atrito entre dois objetos comuns, o *Unity* utiliza um sistema especial de atrito para representar a interação entre os pneus do veículo e o terreno. Pode-se selecionar valores para as coordenadas de *Extremum* (valor máximo onde o atrito começa a diminuir) e *Asymptote* (valor mínimo do atrito a altas velocidades) tanto para *Forward Friction* e *Sideways Friction*, as quais tratam do atrito de movimento para a frente e lateral, respectivamente, criando a curva de atrito da Figura 5 vista anteriormente a partir destes pontos. Por fim, o valor *Stiffness* é um valor multiplicador dos outros, o que efetivamente o torna o controlador da intensidade do atrito.

4.1.2 O cenário

Além dos componentes do veículo, pode-se visualizar na figura 8 as outras subcategorias além do veículo que constituem o cenário. Elas são, em ordem:

Canvas: Esta é a interface do jogo que é sobreposta na tela a todo momento, representada na figura 14.

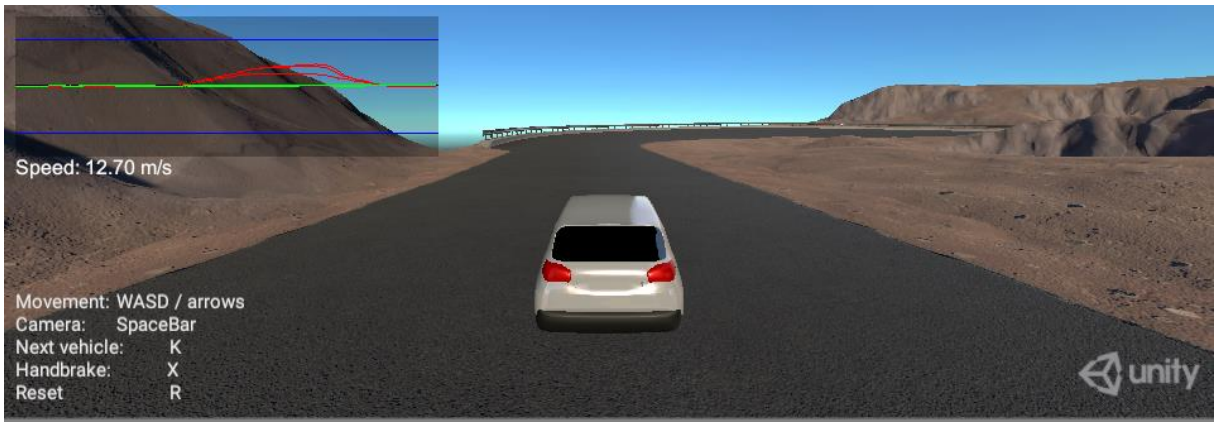


Figura 14: Modo *Play* do cenário de jogo, mostrando sua interface. Fonte: Autoria Própria

Na parte inferior, a interface contém os controles do jogo, à esquerda, e o logotipo da *Unity*, à direita. Na parte superior-esquerda, há o medidor de velocidade do veículo, e o gráfico que mede a derrapagem. Este é um gráfico Derrapagem x tempo, onde a linha vermelha representa a derrapagem longitudinal, e a verde a lateral. Curvas suaves, como na figura 14, indicam que a simulação está funcionando normalmente. Quando elas oscilam rapidamente, isto significa que o jogo não está conseguindo calcular os valores apropriados das forças a tempo. Um exemplo desta situação é disponibilizada no manual do *VehicleTools*,

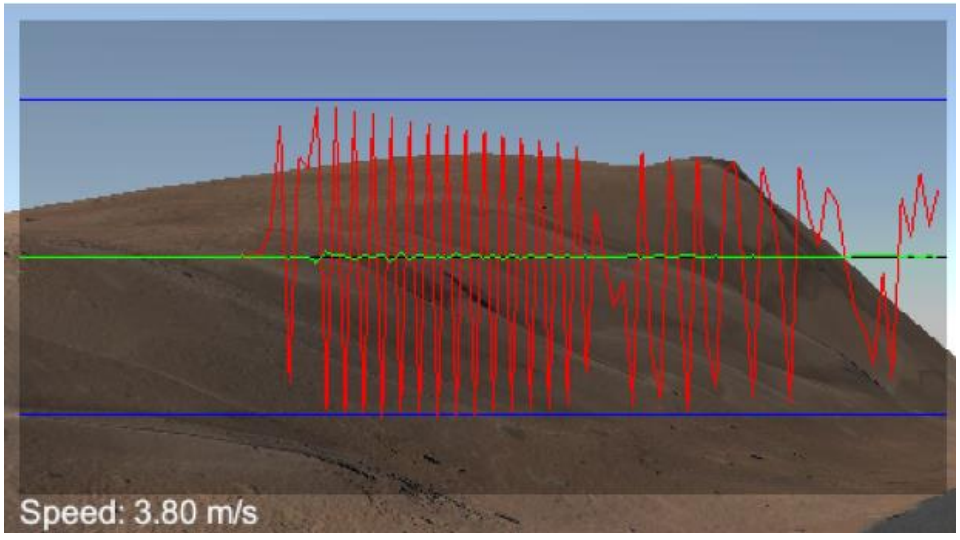


Figura 15: Gráfico da derrapagem do *VehicleTools*. Fonte: Manual do *VehicleTools*, *Unity*

Lighting: Objeto que produz iluminação ao cenário. Sua posição determina a orientação, intensidade e tamanho da formação de sombras, por exemplo.

Environment: Esta seção engloba todos os outros objetos com o qual o veículo pode interagir e colidir. Isto inclui a pista, as montanhas, e diversos obstáculos ao longo do caminho, como rampas, lombadas, e pedras.

Cameras: Contém a câmera principal, que determina o que o jogador verá do

cenário ao jogar.

SpawnPoints: Determina a posição inicial do veículo no cenário, e as posições nas quais ele pode ser posto ao reiniciar o jogo a partir do botão R (Reset).

Além dos elementos estudados, é possível adicionar novos scripts ao jogo para realizar outras funções desejadas, se necessário, através do botão *Add Component*. Tais scripts podem ser selecionados da base do Unity, obtidos na internet, ou programados pelo elaborador do jogo. Estes elementos serão utilizados na criação da nova pista de teste para o veículo escolhido. Agora que seu funcionamento é compreendido, pode-se iniciar a construção de um novo ambiente para análise física para o carro.

4.2 Construção da pista de teste

Para construir a pista de teste, é preciso primeiramente identificar seus objetivos. São eles:

- Permitir a locomoção do carro ao longo de uma pista plana e reta, e em pista com aclive e declive;
- Verificar o que ocorre quando o veículo colide com um obstáculo não-fixo.

Sendo assim, foi construído um circuito fechado retangular, no qual parte da pista consiste de uma rampa que leva a uma seção elevada da pista, com um objeto posicionado no meio da seção da pista sem elevação, como obstáculo. Além da pista, adiciona-se uma base plana simples como referencial. Este processo será iniciado ao criar um objeto 3D do tipo *Terrain*, com um clique direito sobre a aba *Hierarchy*, e modificando sua textura para contrastar com a cor da interface.

Para iniciar a construção do circuito, cria-se a primeira parte da pista sobre a base a partir de um objeto plano (*Plane*), de dimensões 3, 1, e 25 nas direções X, Y, e Z, constituindo largura, espessura e comprimento, respectivamente. A partir daí, cria-se uma cópia deste plano, e, aplicando-lhe uma rotação de -10 graus em X, produz-se a primeira parte inclinada do circuito. Ao posicionar esta cópia em frente ao primeiro objeto, o circuito encontra-se no estado ilustrado pela figura 16.

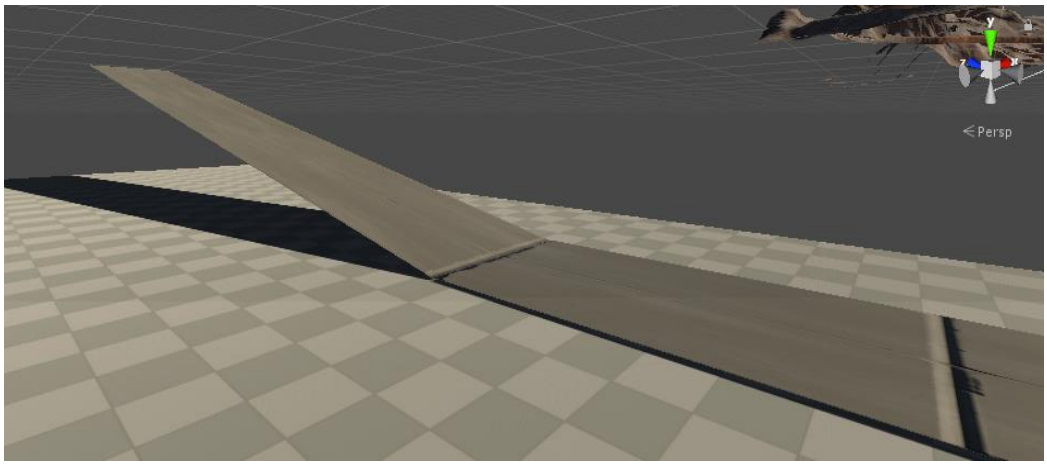


Figura 16: Primeiros passos da construção do circuito de teste, mostrando os três primeiros componentes da pista: A base, a pista horizontal, e a pista inclinada. Fonte: Autoria Própria

É possível facilmente completar o circuito copiando os componentes já feitos. Após copiá-los, basta mudar sua posição X para posicionar as cópias paralelamente aos originais. Para interligá-las, basta criar uma cópia da pista horizontal e posicioná-la perpendicular à primeira pista nas pontas, o que pode ser feito com uma rotação de 90 graus em Y e arrastando o objeto à posição apropriada. Fazendo o mesmo para a outra extremidade, obtém-se o circuito mostrado na figura 17.

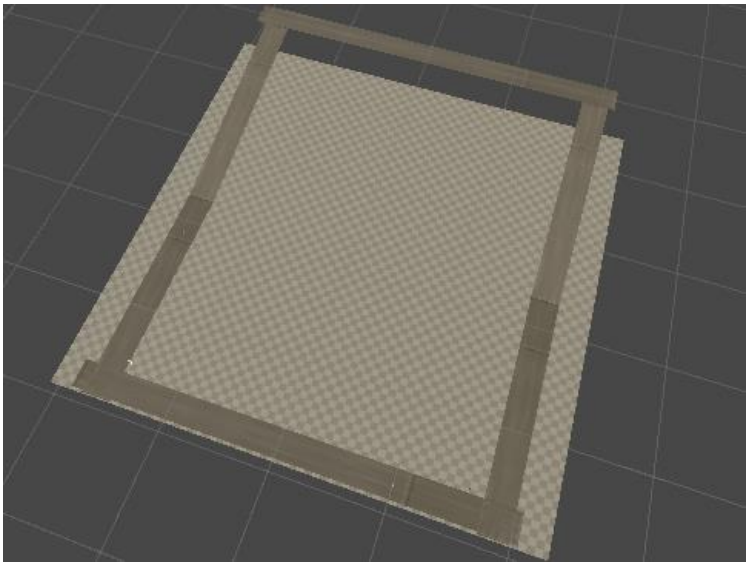


Figura 17: Formato final da pista do circuito de teste. Fonte: Autoria Própria

Com a pista construída, resta apenas adicionar o obstáculo móvel para o teste de colisão. Este objeto é um simples bloco com uma textura chamativa, que permite distinguí-lo facilmente da pista, de dimensões 4m x 2m x 2m, posicionado em (-400, -72, -60). Aplica-se sobre ele um *Physic Material* customizado para este teste, intitulado Caixa. Este será utilizado para determinar algumas propriedades físicas deste objeto, por exemplo, sua massa (1000Kg) e atrito.

5 RESULTADOS, AJUSTES E DISCUSSÕES

Com o novo cenário agora construído. é possível, agora, verificar o comportamento inicial do veículo ao pressionar *Play*. Primeiramente, deve-se ajustar os valores que alteram as propriedades físicas do carro vistas na seção anterior, de forma que o *Unity* possa simular uma situação real da melhor forma possível. Feito isto, o cenário final resultante é executado nas três situações escolhidas e seus resultados serão discutidos, comparando o resultado físico esperado pela teoria e o comportamento exibido no programa.

5.1 Estudo físico do veículo para ajustes de parâmetros

O primeiro teste realizado foi acelerar o veículo continuamente, ao segurar a tecla W. Observa-se, na figura 18, que o veículo obtém, dessa forma, uma velocidade bastante elevada. Isto ocorre porque, como visto na seção anterior, não há nenhuma resistência do ar aplicada neste veículo pelo jogo, por enquanto.

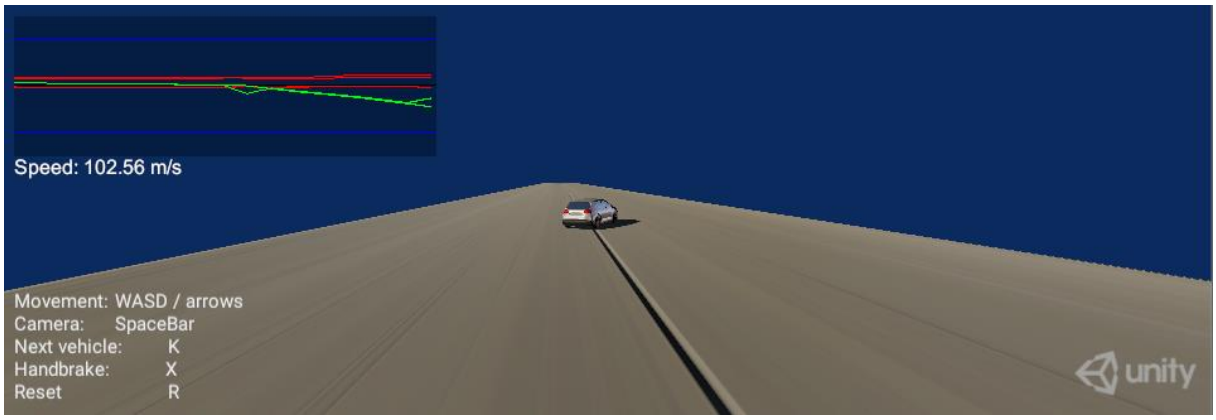


Figura 18: Modo *Play* do Unity com veículo sobre a pista, movimentando-se a altas velocidades. Fonte: Autoria Própria

Utilizando a equação (3.13), é possível determinar um valor aproximado para a resistência do ar. Para definir o valor de C_D, escolheu-se um modelo de veículo que adequadamente representa aquele do jogo. O modelo utilizado aqui é o 2003 Ford Focus C-Max, pois ele se assemelha ao formato do veículo do jogo, e possui diversas informações técnicas disponibilizadas pelo fabricante. Conforme afirma a empresa *Ford of Europe*, o valor de C_D para este carro é de 0,29, sua altura é de 1,588m, e sua largura é de 1,825m. Usando o valor de 1,225 kg/m^3 para a resistência do ar em condições normais calculada por Graham (2006), pode-se, então, calcular a função da resistência do ar com a velocidade:

$$F = \frac{\rho C_D A}{2} v^2 = \frac{1{,}225 \times 0{,}29 \times (1{,}588 \times 1{,}825)}{2} \times v^2 = 0{,}515 v^2 \quad (5.1)$$

Se o veículo se encontrasse em queda livre com sua parte posterior voltada para baixo, sua velocidade terminal seria, para um peso aproximado de 1300kg:

$$v = \sqrt{\frac{m \times g}{0{,}515}} = \sqrt{\frac{1300 \times 9{,}81}{0{,}515}} = 157{,}36 m/s \quad (5.2)$$

Realizando o mesmo teste no jogo, no entanto, resulta em uma velocidade terminal de aproximadamente 33,63m/s. Nota-se, então, que o jogo utiliza uma fórmula diferente para o cálculo da resistência do ar. Realizando testes com diversos valores, verifica-se a relação entre a constante de resistência ao movimento (*drag*) no *Unity* e a velocidade terminal de queda, representados na tabela 1.

Tabela 1: Velocidade terminal de queda do veículo para diversos valores de *drag*

Drag	Velocidade terminal (m/s)
0,1	97,95
0,2	48,95
0,4	24,43
0,8	12,21

Nota-se que a velocidade é inversamente proporcional ao valor da constante de resistência, diferentemente do comportamento real denotado pela equação (3.13), que demonstra uma relação quadrática inversa. Isto indica que o programa assume que o fluxo do ar é laminar. Será necessário, então, criar um *script* que adequadamente simule a resistência do ar.

Exibido na figura 19, o *script* que possui o funcionamento desejado, adaptado de um escrito por Foddy (2015), aplica uma força sempre contrária ao vetor velocidade do objeto. Esta força possui intensidade igual ao produto da variável *drag*, que possui o valor atribuído de 0,515 no *script,* com a velocidade, e então novamente multiplicado pela velocidade, resultando numa força aplicada igual a $0{,}515v^2$. Repetindo o teste anterior, de fato obtém-se um valor muito mais próximo daquele obtido pela equação (5.2), de cerca de 153m/s. Deve-se criar uma cópia deste *script* para aplicar no bloco com o qual o teste de colisão será realizado, com um coeficiente distinto. Intitulado ResArB, seu coeficiente, para um bloco de massa 1000Kg, será:

$$k = \frac{\rho C_D A}{2} = \frac{1{,}225 \times 1{,}05 \times (2 \times 2)}{2} = 2{,}5725 \quad (5.3)$$

ResAr

Assembly Information

Filename Assembly-CSharp.dll

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class ResAr : MonoBehaviour {

            // Use this for initialization
public float thrust;
   public Rigidbody rb;

   void Start()
   {
      rb = GetComponent<Rigidbody>();
   }

   void FixedUpdate()
   {
      float drag = -0.515f; //(or some other negative
number, you'll need to tune it)
Vector3 force = drag *
GetComponent<Rigidbody>().velocity.normalized *
GetComponent<Rigidbody>().velocity.sqrMagnitude;
GetComponent<Rigidbody>().AddForce(force);
   }
}
```

Figura 19: Script intitulado ResAr, que gera a resistência do ar no veículo no cenário. Fonte: Autoria Própria

Em seguida, modificou-se o atrito do veículo. Como visto na seção anterior, a perda de velocidade origina-se da resistência ao rolamento. Como o *Unity* calcula os parâmetros de atrito a partir de valores da curva da figura 5, estes deverão ser preenchidos por um caso real desta curva. Um estudo realizado por HasanKhansari et al (2015) indica as seguintes curvas para a derrapagem na figura 20, sendo a curva superior sobre asfalto seco.

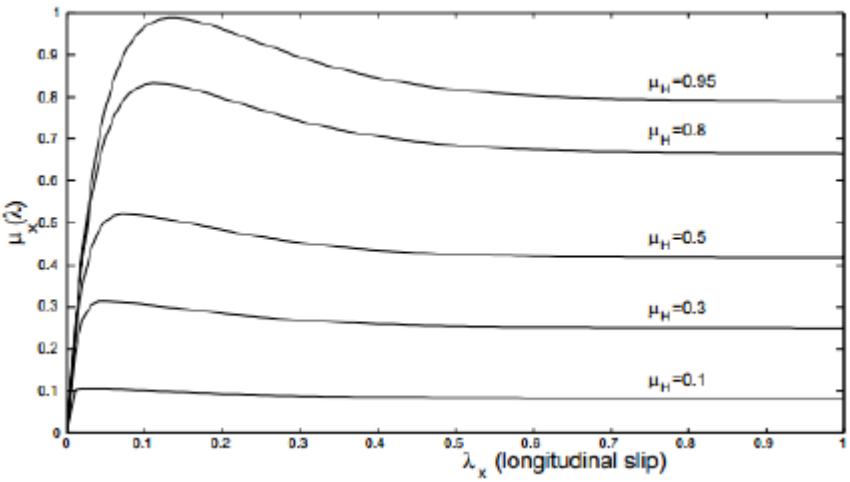


Figura 20: Gráfico Força de atrito x Derrapagem para um veículo comum sobre diversos terrenos. Fonte: HasanKhansari et al (2015)

Observa-se, então, que a curva obtém seu valor máximo para o atrito (aproximadamente 1) em cerca de 0,15 de derrapagem, e estabiliza-se em uma reta em 0,6 de derrapagem, para um valor do atrito de 0,8, valores que então são adicionados no *Unity.* Mas como saber se estes valores produzem o resultado

esperado? Segundo Petersen (2017), veículos sobre pista molhada geralmente perdem o controle a partir da velocidade de cerca de 72 a 93 km/h. Inserindo os valores da segunda curva da figura 21 no *Unity*, será realizado um teste para observar se o veículo continua estável acima da velocidade na qual um veículo real derraparia.

Ao executar o teste, observa-se que o veículo já começa a deslizar em tentativas de curvas em 20 m/s (72 km/h), condizente com o valor citado acima. Retornando para os valores para a primeira curva, o carro não perde mais o controle, até tentativas de curvas do dobro desta velocidade, isto é, 40 m/s. É possível, então, concluir que esta simulação de atrito é precisa o suficiente para este propósito.

Para finalmente ser possível realizar a execução do jogo para verificar seu comportamento final, os parâmetros do veículo e dos materiais foram modificados para valores conhecidos. Primeiramente, para os materiais, tem-se:

- *Friction*: Aproximadamente 0,6 estático e 0,5 dinâmico para o chassi do veículo, e 0,5 estático e 0,4 dinâmico para um obstáculo feito de pedra. Devido ao cálculo do atrito do *Unity* considerando superfícies extensas como dois pontos de contato, estes valores são duplicados. Portanto, deve-se inserir apenas metade destes valores no programa.

- *Bounciness*: Um estudo de Batista (2006) afirma que o coeficiente de restituição em uma colisão automobilística comum gira em torno de 0,45.

Em seguida, para os parâmetros do carro do script do *VehicleTools*:

- *Max Torque*: A ficha técnica do carro simulado, o 2003 Ford Focus C-Max, indica um torque máximo de 320 Nm. Este valor será utilizado para os primeiros testes.

Com os parâmetros relevantes inseridos, agora é possível iniciar os testes e discussões sobre estes e seus resultados.

5.2 Execução do jogo e comparação com a Física

Nesta última seção, o procedimento para comparar o sistema físico virtual criado no capítulo anterior com as leis da Física terá o seguinte formato: Calcula-se o comportamento esperado do veículo nas diversas situações e locais da pista escolhidas através das equações da Física apresentadas neste trabalho, e em

seguida, observa-se se o veículo do jogo mostra valores similares durante o teste. Caso hajam divergências significativas entre os valores, deverão ser apresentadas justificativas, e se necessário, soluções para o ocorrido. Estas divergências foram apresentadas na forma de porcentagem. Este processo foi dividido entre as partes do cenário criado: A pista reta, as subidas e descidas, a curva, e finalmente, a colisão com o bloco.

Posicionando o veículo nas coordenadas (-700, -73, 155), ele se encontra no início da pista reta horizontal do percurso, permitindo a verificação da sua aceleração e desaceleração de freio. O primeiro passo é determinar a força resultante atuante sobre o veículo através da soma das forças existentes neste. Como ilustrado na figura 21, é possível determinar esta soma a partir de um diagrama de forças.

Sendo a força de resistência ao rolamento $\overrightarrow{F_{rr}}$ e a força de resistência do ar $\overrightarrow{F_d}$ sempre contrárias ao movimento, este causado pela força $\vec{F}$ gerada pelo motor, a equação que determina a força resultante é:

$$\overrightarrow{F_R} = \vec{F} - \overrightarrow{F_{rr}} - \overrightarrow{F_d} \tag{5.4}$$

Figura 21: Diagrama de forças horizontais do veículo na seção horizontal plana do circuito. Fonte: Autoria Própria

Basta determinar, então, o valor de cada uma destas forças. Iniciando pela força $\vec{F}$, resultante da soma das quatro forças de impulso $\overrightarrow{F_W}$ idênticas geradas pelas rodas:

$$\vec{F} = 4\overrightarrow{F_W} \tag{5.5}$$

Estas, por sua vez, são originadas do torque gerado pelo motor. Utiliza-se a eq. (3.12) para determinar esta força:

$$\vec{\tau} = \vec{r} \times \vec{F} \ \therefore F_W = \frac{\tau}{r \operatorname{sen} \theta} \tag{5.6}$$

Substituindo os valores de r = 0,3m (raio da roda, pois o ponto de aplicação da força é na sua extremidade), τ = 320 Nm e θ = 90°, pois o eixo de rotação das

rodas é perpendicular ao movimento do veículo:

$$\vec{F} = 4\overrightarrow{F_W} = 4\left(\frac{320}{0{,}3\ \text{sen}\,90°}\right) = 4267N \tag{5.7}$$

Agora, para $\overrightarrow{F_{rr}}$, será utilizada a eq. (3.15). Sendo a massa m do carro igual a 1300kg, a aceleração da gravidade g igual a 9,81 m/s^2, e C_{rr} = 0,01, tem-se:

$$\overrightarrow{F_{rr}} = C_{rr}\mathrm{N} = C_{rr}mg = 0{,}01 \times 1300 \times 9{,}81 = 127{,}53N \tag{5.8}$$

Finalmente, para a resistência do ar, ela é determinada pela equação (5.1) para este veículo. Nota-se, no entanto, que ela depende do quadrado da velocidade. A equação da força resultante é, então:

$$\overrightarrow{F_R} = 4267 - 127{,}53 - 0{,}515v^2 = 4139{,}47 - 0{,}515v^2 \tag{5.9}$$

Como a força resultante é variável, será necessário obter o valor médio desta função no intervalo desejado para calcular a aceleração média. Isto é realizado através da seguinte equação:

$$M = \frac{1}{b-a}\int_a^b f(x)dx$$

(5.10)

Para uma medição da velocidade de 0m/s até 15m/s, o valor médio da força resultante deverá ser:

$$F_R = \frac{1}{15}\int_0^{15} 4139{,}47 - 0{,}515v^2 dv = \frac{4139{,}47 \times 15 - 0{,}515\frac{(15^3)}{3}}{15} = 4100N \quad (5.11)$$

A partir daí, utiliza-se a Segunda Lei de Newton (eq. 3.11) para determinar a aceleração:

$$a = \frac{F_R}{m} = \frac{4100}{1300} = 3{,}16m/s^2 \quad (5.12)$$

Para verificar se o veículo de fato possui esta aceleração, mede-se o tempo necessário para o veículo do *Unity* obter a velocidade de 15m/s com o *XNote Stopwatch*. Este teste é mostrado na figura 22.

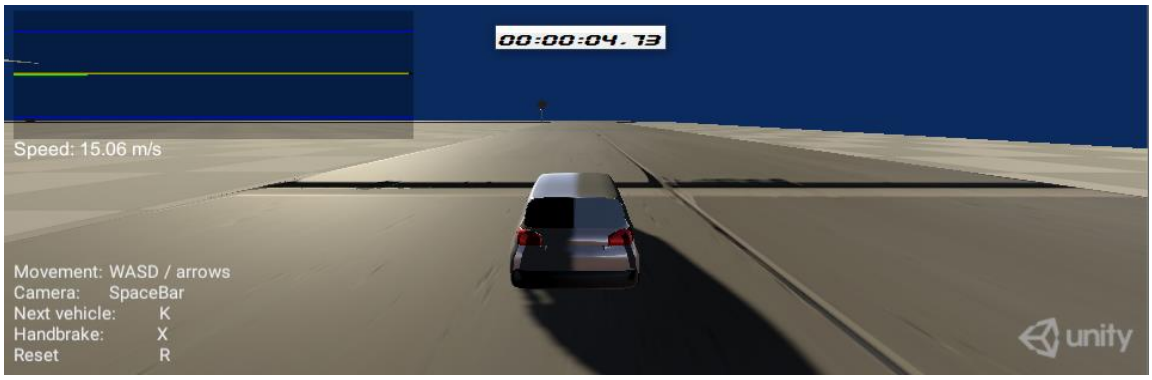


Figura 22: Medição do tempo levado para o carro obter a velocidade de 15m/s. Fonte: Autoria Própria

Conclui-se, então, que a aceleração média deste veículo é:

$$a = \frac{\Delta v}{\Delta t} = \frac{15{,}06}{4{,}73} = 3{,}18m/s^2 \quad (5.13)$$

O valor obtido pelo teste é, então, 0,6% maior que o esperado. Esta minúscula diferença entre os valores pode ser atribuída a variações causadas pela medida e aproximação de valores. No caso da frenagem do veículo, a força de desaceleração é originada da mudança da forma de movimento do veículo: Como o sistema de frenagem impede o rolamento das rodas, este começa a deslizar sobre a pista, possuindo o movimento comum de um objeto atritando-se com a superfície. Assim, a soma das forças resultantes é:

$$\overrightarrow{F_R} = \overrightarrow{F_{at}} + \vec{F}_d \quad (5.14)$$

Sabendo que o coeficiente de atrito dinâmico μ_d entre os pneus e a pista é 0,6, determina-se esta força pela equação (3.14). Utilizando o mesmo intervalo de velocidade que no caso anterior, já se obtém o módulo de $\vec{F}_d$ de 39,47N. A força

$$\overrightarrow{F_R} = 0{,}6 \times 1300 \times 9{,}81 + 39{,}47 = 7691{,}27N$$

resultante é, então:

(5.15)

Daí, o módulo da desaceleração do carro é:

$$a = \frac{7691,27}{1300} = 5,91 m/s^2 \tag{5.16}$$

Freando o veículo no *Unity* a partir de 15m/s até o instante que ele chega ao repouso e marcando o tempo necessário para isto com o cronômetro, obtém-se o valor mostrado na figura 23.

Figura 23: Medição do tempo levado para o carro chegar ao repouso a partir da velocidade de 15m/s usando o freio. Fonte: Autoria Própria

Determinado o tempo, podemos encontrar a desaceleração média do veículo no ambiente de jogo:

$$a = \frac{15}{2,56} = 5,86 m/s^2 \tag{5.17}$$

Isso indica que o valor teórico é apenas 0,85% maior. Agora trasladando o veículo para a posição (-700, -37, 370) localizada no início da rampa descente, poderá ser realizado um teste em condições na qual a gravidade pode afetar a aceleração do carro. Nesta primeira condição, a figura 24 mostra as possíveis forças que podem atuar no movimento do veículo.

Figura 24: Diagrama de possíveis forças horizontais a atuar no veículo na posição (-700, -37, 370) sobre a rampa. Fonte: Autoria Própria

A força resultante é, então, encontrada através da seguinte equação:

$$\overrightarrow{F_R} = \overrightarrow{P_x} + \vec{F} - \overrightarrow{F_{rr}} - \overrightarrow{F_d} \tag{5.18}$$

Caso a medição seja novamente feita de 0 a 15m/s, a única força ainda desconhecida é a componente horizontal do peso $\overrightarrow{P_x}$. Como mostra a figura 24, o ângulo da rampa em relação à horizontal $\theta = 10°$. Calcula-se, então, o módulo desta força:

$$P_x = mg\,\text{sen}\,\theta = 1300 \times 9{,}81 \times \text{sen}\,10° = 2215N \tag{5.19}$$

Da equação (5.11), já é conhecido o valor do módulo da força quando $\overrightarrow{P_x}$ não é presente. A aceleração resultante é, finalmente:

$$a = \frac{4100 + 2215}{1300} = 4{,}86m/s^2 \tag{5.20}$$

Executando o jogo e acelerando o carro, obtém-se os valores da figura 25 para a velocidade e tempo, buscando alcançar um valor próximo de 15m/s para a velocidade. Utilizando mais uma vez a equação (3.3), encontra-se a aceleração do veículo do jogo:

$$a = \frac{14{,}85}{3{,}08} = 4{,}82m/s^2 \tag{5.21}$$

Figura 25: Medição do tempo para o carro obter uma velocidade de 15m/s na descida. Fonte: Autoria Própria

Desta vez, a diferença é de 0,8%, ainda insignificante. Para realizar este procedimento na subida, basta utilizar-se a equação (5.20) para um valor negativo da componente horizontal peso, pois agora ele age na direção oposta ao movimento. Aqui, a aceleração deverá ser:

$$a = \frac{4100 - 2215}{1300} = 1{,}45m/s^2 \tag{5.22}$$

Para facilitar a realização do teste de medida da aceleração do veículo nas condições desejadas, mais uma vez determina-se uma posição inicial que permita

simplificar os comandos necessários para chegar à esta situação. Neste caso, posiciona-se o carro nas coordenadas (-300, -65, 200), rotacionado em 180° em relação ao caso anterior, isto é, o início da pista ascendente. Realizando o teste pela execução do jogo, obtém-se os dados exibidos na figura 26. Daí, a aceleração do veículo no cenário de jogo é:

(5.22)

O valor obtido no teste é 4,8% maior que o esperado. Apesar de ser um diferença significantemente maior que as outras porcentagens de erro obtidas neste trabalho até aqui, este ainda é satisfatoriamente próximo daquele esperado pelo cálculo físico.

Figura 26: Medição do tempo para o carro obter uma velocidade de 15m/s na subida. Fonte: Autoria Própria

O último teste a ser realizado sobre este circuito neste trabalho refere-se à colisão entre o veículo e o bloco de 1000Kg situado no meio da seção menor da pista. Para ir de encontro a este em colisão frontal, na qual é possível aplicar as equações (3.21) e (3.22) no estudo destas colisões, o veículo é inserido nas coordenadas (-510, -72, -60,5), o que garante isto se ele for acelerado em linha reta apenas. Logo antes da colisão, buscará-se obter a velocidade de 20m/s para o carro, enquanto que o bloco permanece em repouso.

Utilizando estas equações. as respectivas velocidades esperadas são, então:

$$v_{1f} = \frac{(1+0{,}45)\times 1000\times 0}{1000+1300} + \frac{20(1300-1000\times 0{,}45)}{1000+1300} = 7{,}39 m/s \quad (5.23)$$

$$v_{2f} = \frac{(1+0{,}45)\times 1300\times 20}{1000+1300} + \frac{0(1000-1300\times 0{,}45)}{1000+1300} = 16{,}39 m/s \quad (5.24)$$

O jogo não possui, no entanto, medidor de velocidade para o bloco, portanto

não é possível verificar este valor. Analisando suas coordenadas no *Unity,* no entanto, é possível determinar a distância percorrida após a colisão. Encontra-se, primeiro, sua desaceleração, sabendo que as forças atuantes são a força de atrito $\overrightarrow{F_{at}}$ e a força de resistência do ar $\overrightarrow{F_d}$, contrárias ao movimento:

$$a = \frac{F_R}{m} = \frac{-F_d - F_{at}}{m} \tag{5.25}$$

Para este objeto, $k = 2.5725$, $m = 1000Kg$ e $\mu_d = 0{,}4$. A força de resistência do ar média é:

$$\overline{F_d} = \frac{1}{16{,}39 - 0}\int_0^{16{,}39} 2{,}5725v^2 dv = 230{,}35N \tag{5.26}$$

e a força de atrito é:

$$F_{at} = 0{,}4 \times 1000 \times 9{,}81 = 3924N \tag{5.27}$$

resultando em uma desaceleração igual a:

$$a = \frac{-3924 - 230{,}35}{1000} = -4{,}15m/s^2 \tag{5.28}$$

A distância percorrida pelo bloco deve ser, finalmente:

$$d = \frac{v^2 - {v_0}^2}{2a} = \frac{0 - 16{,}39^2}{-2 \times 4{,}15} = 32{,}36m \tag{5.29}$$

Agora, acelera-se o veículo até o velocidade mais próxima possível de 20m/s, e registra-se a velocidade que ele apresenta após esta colisão. As figuras 27 e 28 apresentam os resultados obtidos.

Figura 27: Velocidade do veículo imediatamente após colisão com bloco, com velocidade inicial do veículo de 20m/s. Fonte: Autoria Própria

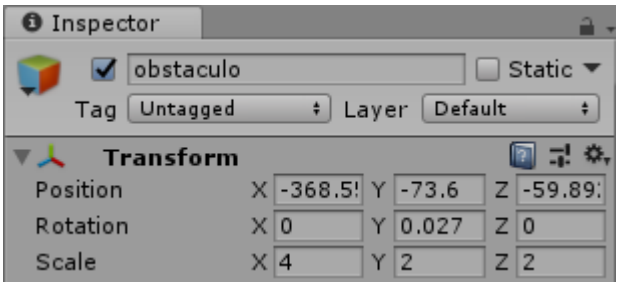


Figura 28: Coordenadas finais do bloco com o qual o veículo colidiu, após chegar novamente ao repouso. Fonte: Autoria Própria

Como a coordenada X indica a direção na qual houve o movimento e possuía valor inicial de -400 para o obstáculo, conclui-se que seu deslocamento foi de 31,45m, então, o valor calculado é 2,9% maior que este obtido. Por sua vez, a velocidade do veículo após a colisão é apenas 0,7% maior no cálculo teórico em relação àquele obtido na simulação. A maior divergência do primeiro pode ser justificada pelo fato de que a colisão não é perfeitamente unidimensional, resultando num pequeno deslocamento perpendicular ao eixo analisado por este experimento, isto é, no eixo Z do *Unity*. Mais uma vez, então, obtiveram-se valores de pequena divergência com os valores calculados pelas equações físicas, indicando que o sistema criado no *Unity* adequadamente simula as leis da Física no casos estudados.

Conclusão

Através do estudo do programa *Unity* e suas funcionalidades, foi possível criar um ambiente de jogo que permitiu a demonstração de um sistema físico, validado pela análise através das leis da Mecânica. Portanto, este trabalho permitiu uma introspecção sobre a Física aplicada no desenvolvimento de jogos eletrônicos modernos, vista como indispensável para uma jogabilidade e experiência do nível esperado pelos consumidores de tais produtos no cenário atual.

Os conceitos tratados neste trabalho constituem, no entanto, uma pequena porção da imensa gama de fenômenos físicos tanto presentes no mundo real como passíveis de representação nestes jogos, não apenas da Física em geral, mas inclusive da Mecânica em si, que poderiam até mesmo serem realizados neste mesmo cenário. O estudo feito pode servir como introdução básica ao método de implementação de sistemas físicos representativos da realidade nestes programas ou até simuladores, de fato representando os fenômenos considerados de forma simples.

Tendo isso em vista, o trabalho cumpriu seu objetivo de demonstrar este processo, mesmo que de forma não-extensiva e com as limitações do programa utilizado. Por fim, nota-se que esta pode ser uma oportunidade de uso das equações da Física para a construção de algo que qualquer pessoa pode fazer e reproduzir, indicando possíveis usos educacionais, onde o aluno pode aplicar seu conhecimento de modo talvez ainda mais interativo do que o simples uso dos programas simuladores disponibilizados prontos para o uso.

Referências

BATISTA, M. *On the mutual coefficient of restitution in two car collinear collisions*. arXiv preprint physics/0601168, 2006.

BEER et al. *Vector Mechanics for Engineers Sixth ed*. McGraw-Hill. p. 397, 1996.

DA SILVA,F. W*. Introdução ao Ray Tracing,* COPPE/ UFRJ, 1994.

DE AGUIAR, M. *Tópicos de mecânica clássica*. Livraria da Fısica*,* 1ª edição, 2011.

FALKOVICH, G. (2011). *Fluid Mechanics (A short course for physicists)*. Cambridge University Press, 2011.

FORD OF EUROPE. All-new Ford Focus C-Max. Disponível em: <https://www.fordforums.com/f349/all-new-ford-focus-c-max-32598> Acesso em: 21 de julho de 2018.

GERSTMANN, J. *Super Mario 64 Virtual Console Review*. Disponível em: <https://www.gamespot.com/articles/super-mario-64-virtual-console-review/1100-6162119/> Acesso em 14 de junho de 2018.

GILLESPIE, T. D. *Fundamentals of Vehicle Dynamics*, SAE International, 1992.

GRAETZ, J. M. *The origin of Spacewar*. Creative Computing. Vol. 6 no. 8. pp. 56–67. *ISSN 0097-8140, 1981.*

GYATT, G. The Standard Atmosphere. *NASA, Washington, United States*, 1976.

HAAS, J. A History of the Unity Game Engine. *Worcester, UK: Worcester Polytechnic Institute,* 2014.

HASANKHANSARI et al. *Independent Model Generalized Predictive Controller Design for Antilock Braking System.* International Journal of Computer Applications, p. 18-23, 2015.

HENRY, C. *Physics in Mass Market Games*. Disponível em: <http://www.gamecareerguide.com/features/534/physics_in_mass_market_.php> Acesso em: 14 de junho de 2018.

HIBBELER, R.C. *Engineering Mechanics: Statics & Dynamics (Eleventh ed.).* Pearson, Prentice Hall. P. 441–442, 2007.

KALNING, K. *The anatomy of the first vídeo game.* Disponível em: <http://www.nbcnews.com/id/27328345/> Acesso em: 14 de junho de 2018.

KENT, S. *And Then There Was Pong*. Ultimate History of Video Games, *Three Rivers Press*. pp. 40–43. *ISBN 0-7615-3643-4*, 2001.

KROUWEL, A. *The Making of... 3D Monster Maze*. Disponível em: <https://web.archive.org/web/20070513045033/http:/www.edge-online.co.uk/archives/2006/04/the_making_of_3_1.php> Acesso em: 14 de junho de 2018.

NETO, J. B. *Mecânica Newtoniana, Lgrangiana e Hamiltoniana*. Editora Livraria da Física, 2004.

NICKLIN, R. C. *Kinematics of Tailgating*. The Physics Teacher, Vol.35, p. 78, 1996.

NUSSENZVEIG, H. M. *Curso de Física Básica. 1 – Mecânica*. São Paulo: Edgard Blücher. 3ª edição, p. 65 a 69, 1981.

PERMADI, F. *Ray-Casting Tutorial For Game Development And Other Purposes*. Disponível em: <https://permadi.com/1996/05/ray-casting-tutorial-table-of-contents/> Acesso em: 14 de junho de 2018.

PETERSEN, G. Best and Worst Tires in All Weather Conditions. Disponível em: *<https://www.consumerreports.org/cro/tires/best-and-worst-tires-in-all-weather-conditions>* Acesso em 14 de junho de 2018.

RESNICK et al. *Física 1.* LTC, p. 110, *2003.*

SERWAY et al. *Physics for Scientists and Engineers 6th Ed*. Brooks Cole, 2003.

TAYLOR, J. R. *Mecânica clássica*. Bookman Editora, 2013.

UNITY TECHNOLOGIES. Factos Rápidos. Disponível em: <https://unity3d.com/pt/public-relations> Acesso em: 02 de junho de 2018.

UNITY TECHNOLOGIES. Manual do Unity. Disponível em:<https://docs.unity3d.com/Manual/> Acesso em: 14 de junho de 2018.

WARD, J. *What is a Game Engine?.* Disponível em: *<http://www.gamecareerguide.com/features/529/what_is_a_game_.php?page=2> Acesso em: 14 de junho de 2018.*

WEST, N. *The Next Generation 1996 Lexicon A to Z: Mode 7*. *Next Generation*. No. 15. Imagine Media p. 37, 1996.

WYCKOFF, R. *Postmortem: DreamWorks Interactive's Trespasser*. Disponível em: *<http://www.gamasutra.com/view/feature/131746/postmortem_dreamworks_.php?page=2> Acesso em: 14 de junho de 2018.*