# Protótipo de um simulador de movimentos para jogos de corrida

**Pablo Borges Pimentel[1], Sandro Roberto Fernandes[2]**

[1]Instituto Federal de Educação, Ciência e Tecnologia do Sudeste de Minas Gerais – *Campus* Juiz de Fora

[2]Núcleo de Informática - Instituto Federal de Educação, Ciência e Tecnologia do Sudeste de Minas Gerais – *Campus* Juiz de Fora

borgespablo1991@gmail.com, sandro.fernandes@ifsudestemg.edu.br

***Abstract.*** *The purpose of this paper is to present a motion simulator prototype, capable of simulating the movement of a car from a racing game. The prototype works combining three different scopes: the setup of SimTools software, responsible for extracting data from the racing game; the development of an electronic project using Arduino and servo motors, which will receive the data extracted by the software and a mechanical structure which will effectively move when receiving the servo motor's actuation.*

***Resumo.*** *O objetivo deste artigo é apresentar um protótipo de um simulador de movimentos que seja capaz de simular a movimentação de um automóvel de um jogo de corrida. O funcionamento do protótipo, se dá através da união de três âmbitos distintos: a parametrização do software SimTools, que captura dados do jogo de corrida; o desenvolvimento de um projeto eletrônico utilizando Arduino e servo motores, que recebe os dados capturados pelo software e a criação de uma estrutura mecânica, que efetivamente se movimenta através da atuação dos servo motores.*

## 1. Introdução

A indústria dos jogos eletrônicos cresceu de forma muito expressiva na última década, e especialmente no Brasil, a indústria dos *games* já movimenta mais de 1,5 bilhão de dólares por ano (MENDES, 2019), ou seja, o dobro em relação à década anterior. O número de equipamentos e plataformas para jogar também cresce a cada dia, e dentro da infinidade de categorias e estilos de jogos disponíveis, existem os simuladores de corrida, que possuem a proposta de aproximar o jogador o máximo possível da sensação de pilotar um automóvel real.

Os simuladores mais recentes, além de apresentarem visuais gráficos realísticos e física elaborada, também permitem ao jogador utilizar uma série de periféricos e controles que imitam a realidade, como volantes e pedais que transmitem as sensações de um

automóvel verdadeiro (perda de tração nas rodas, trepidações em relevos irregulares, entre outros). Nos últimos anos, a imersão proporcionada pelos simuladores, atingiu um novo patamar, combinando a utilização desses periféricos com a realidade virtual, que está mudando totalmente a forma com que o jogador interage e percebe o mundo digital ao seu redor.

A proposta deste trabalho é criar um protótipo em miniatura, de uma plataforma que simule os movimentos de um automóvel de um jogo de corrida. Este protótipo, irá servir de base para o desenvolvimento futuro de uma plataforma em tamanho real, que possa ser efetivamente utilizada por um jogador, a fim de aumentar o realismo e a imersão na experiência de jogo.

## 2. Revisão Sistemática

O escopo da Revisão Sistemática iniciou pela busca de artigos e patentes em sítios de indexação acadêmicos: Periódicos da Capes e Google Scholar. O levantamento realizado encontrou resultados relevantes e proporcionou esclarecimento para as principais questões referentes ao desenvolvimento do protótipo. Abaixo, podem ser visualizadas uma lista, com as principais perguntas a serem respondidas pela revisão sistemática e também a Tabela 1, contendo os principais artigos e patentes utilizados como referência:

1. É possível a construção de um simulador de movimentos para jogos de corrida?
2. Quais softwares podem ser utilizados para capturar os dados de movimento (telemetria) dos carros em jogos eletrônicos?
3. Qual a melhor opção de design mecânico: *full frame* ou *seat shaker/mover*?
4. Quantos eixos de movimentação ou graus de liberdade *(degrees of freedom)* se adequam melhor a um simulador de movimentos para jogos de corrida?
5. Quais os materiais necessários para construção de um simulador de movimentos?
6. Qual placa controladora é mais indicada para o desenvolvimento de um simulador de movimentos?
7. Qual tipo motor elétrico é mais indicado para o desenvolvimento de um simulador movimentos?

**Tabela 1 – Principais artigos e patentes selecionados na Revisão Sistemática.**

| Título do artigo/patente e autores | Resumo |
|---|---|
| **SIMULADOR DE DIREÇÃO COM TRÊS GRAUS DE LIBERDADE (2010)**<br><br>**Denver Marchese Orsolin**<br>**Eduardo R. Rampelotto** | O artigo apresenta o desenvolvimento e construção de um cockpit completo para simuladores de corrida/direção, que consiste em um banco, volante, pedais, cambio e 3 monitores LCD montados sobre uma plataforma que possui três graus de liberdade de movimentação (pitch, roll e yaw). O trabalho discorre sobre todo referencial teórico necessário para simular os movimentos de um automóvel real (Leis de Newton, Lei de Hooke, forças atuantes num automóvel etc.) e também como é dada a percepção humana desses movimentos (Sistema Vestibular, responsável pelo equilíbrio). Ao longo do artigo são detalhados todos os materiais e componentes utilizados na construção da plataforma (estrutura mecânica, motores elétricos e placas controladoras). |
| **MOTION PLATFORM VIDEO GAME RACING AND FLIGHT SIMULATOR (2012)**<br><br>**Robert Childress** | A patente analisada apresenta um projeto mecânico detalhado para a construção de uma plataforma de movimentos para simuladores de direção, com 2 graus de liberdade de movimentação (pitch e roll), onde podem ser montados sobre ela, um banco, volante, pedais, cambio e uma TV/monitor LCD. O texto destaca que o número de graus de liberdade de movimentação de uma plataforma, está diretamente associado ao custo e a complexidade do projeto. Ou seja, quanto maior o grau de liberdade, maior será o custo e também a fidelidade dos movimentos simulados. Porém também é explicado que utilizando apenas 2 graus de liberdade reais (pitch e roll) é possível transmitir a percepção de outros 3 (heave, surge e sway) ao condutor/piloto. Isso se deve aos movimentos de inclinação, associados ao posicionamento do condutor/piloto e à força da gravidade. |

Baseado nas informações presentes nos artigos e patentes selecionados, foi possível responder as principais questões referentes à construção de um simulador de movimentos

para jogos de corrida, bem como concluir que sua construção é viável, tendo em vista a existência de projetos similares.

## 3. Componentes

Os componentes necessários para a execução deste projeto foram definidos durante sua Revisão Sistemática e divididos em três âmbitos diferentes. Entretanto, ao longo do desenvolvimento, novos componentes foram adicionados de acordo com o design do modelo. Na Tabela 2 abaixo, estão descritos os requisitos fundamentais para a criação do protótipo:

**Tabela 2 – Definição dos três âmbitos do projeto e suas descrições.**

| Âmbitos | Descrição |
|---|---|
| **Software** | O software deve ser capaz de captar a telemetria do carro escolhido no jogo e transmitir essas informações para a unidade eletrônica controladora. Também deve possibilitar a configuração e ajustes necessários para se adequar ao projeto eletrônico. |
| **Projeto Eletrônico** | O projeto eletrônico é responsável por fazer a comunicação entre o software que captura a telemetria e a estrutura mecânica. Os componentes a serem utilizados no projeto eletrônico incluem placa controladora, motores elétricos, e cabos. |
| **Projeto Mecânico** | Deve ser criada uma estrutura mecânica em miniatura, que se movimente através da atuação dos motores elétricos. |

### 3.1. Software

O software foi o primeiro componente definido por ser o ponto inicial da comunicação com os outros elementos do protótipo. O SimTools foi o software escolhido, primeiramente por se adequar ao levantamento de requisitos, permitindo a captura da telemetria do carro escolhido no jogo de corrida, e também por ser capaz de se comunicar com diferentes tipos de placas eletrônicas controladoras. Porém, além de suprir as necessidades fundamentais, o SimTools possui algumas características que favoreceram a sua escolha, como a permissão de uso gratuito, interface amigável, documentação bem definida e suporte a novos jogos através de plugins (XSIMULATOR, 2021).

O SimTools funciona através de 2 interfaces gráficas distintas, o *Game Manager*, responsável por se conectar ao jogo escolhido e capturar os dados, e o *Game Engine*, que faz a comunicação com a placa eletrônica controladora e permite a parametrização de todos os sinais que serão enviados a ela, bem como a possibilidade de testar o modelo. Nas figuras 1 e 2 a seguir, podem ser visualizadas ambas interfaces:


**Figura 1 – Interface gráfica do SimTools Game Manager. Fonte: O autor.**

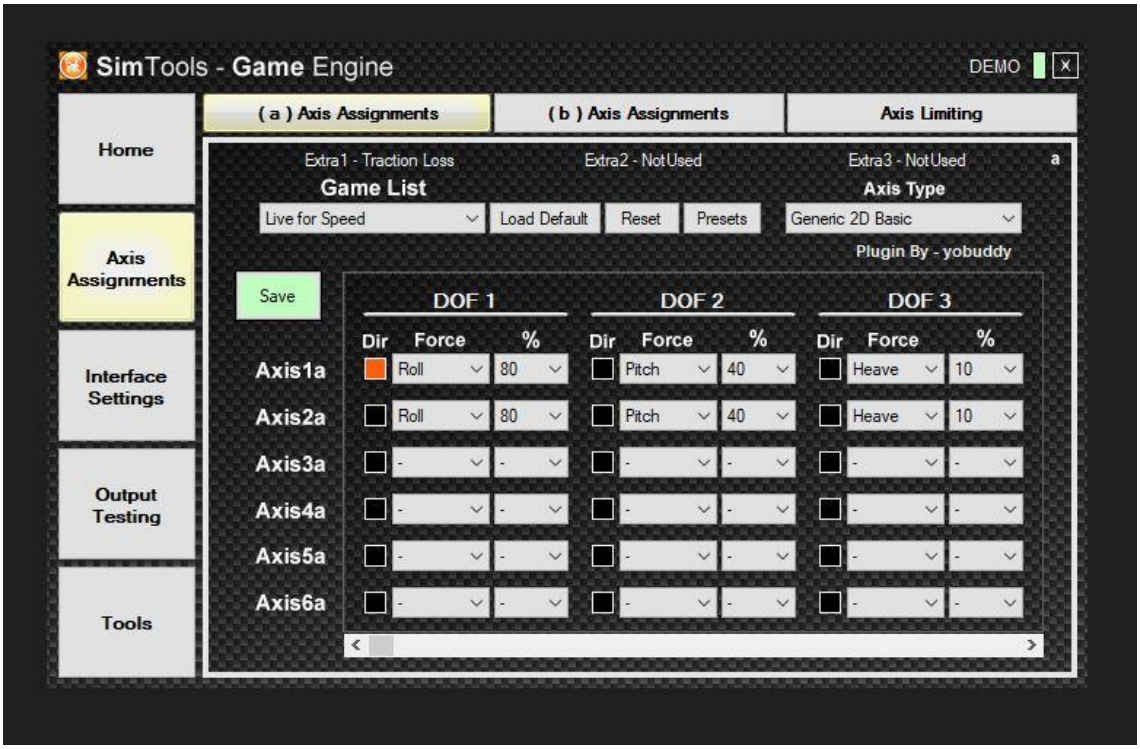



**Figura 2 – Interface gráfica do SimTools Game Engine. Fonte: O autor.**

## 3.2. Projeto Eletrônico

O Projeto Eletrônico é o elemento intermediário do protótipo, sendo o elo de ligação entre o software e a estrutura mecânica. Para que essa comunicação seja feita, o componente a ser definido é a placa eletrônica controladora, que tem a finalidade de receber e interpretar os dados enviados pelo SimTools e ao mesmo tempo converter e transmitir estes dados em impulsos elétricos que movimentam os motores.

A placa controladora escolhida foi o Arduino modelo UNO R3, primeiramente por ser compatível com o SimTools e suprir todos os requisitos, porém, também foi importante o fato de ser codificado utilizando uma linguagem de programação similar ao C e possuir uma documentação extensa disponível na internet (ARDUINO, 2021). Outro ponto decisivo para a sua escolha, observado durante a Revisão Sistemática, foi a existência de outros projetos de simuladores de movimentos que utilizaram o Arduino de forma satisfatória. A Figura 3, exibe uma foto do Arduino UNO R3:

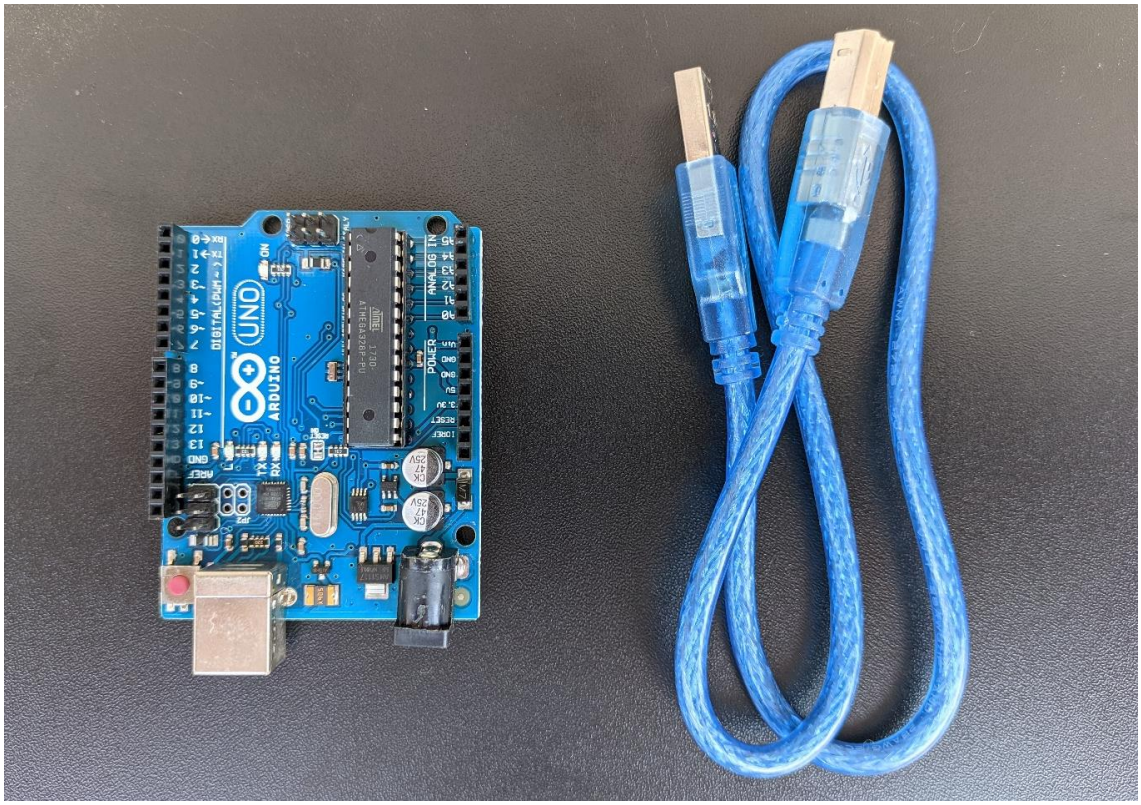



**Figura 3 – Foto Arduino UNO R3 + cabo USB. Fonte: O autor.**

A partir da escolha do Arduino como placa controladora, os componentes do projeto eletrônico precisaram obrigatoriamente ser compatíveis com esta placa, como os motores elétricos. É importante destacar, que para simular a movimentação de um automóvel de forma eficaz, são necessários pelo menos dois atuadores, que vão gerar deslocamento em dois eixos de movimentação diferentes da estrutura mecânica. Portanto, se fez necessária a aquisição de dois servo motores, que serão responsáveis por mover os atuadores acoplados a eles.

> Servo Motor é um dispositivo eletromecânico utilizado para movimentar, com precisão, um objeto, permitindo-o girar em ângulos ou distâncias específicas, com garantia do posicionamento e garantia da velocidade. Possui este nome porque não tem rotação livre e de forma contínua, como um motor convencional. Ele obedece a um comando estabelecido, ou seja, “serve” a um procedimento determinado. (CRAVO, 2019)

Os servo motores escolhidos foram do modelo Tower Pro SG90, um micro servo de baixo custo, largamente utilizado em projetos que envolvem braços robóticos automatizados e aeromodelos de helicópteros e aviões. O Tower Pro SG90 possui o peso de apenas 9g, tamanho compacto (32 x 30 x 12 mm) e capacidade de gerar torque de até 1,6 kg-cm. Os kits adquiridos incluem também três parafusos de fixação e três braços mecânicos diferentes, como pode ser visto na Figura 4:


**Figura 4 – Foto de dois Kits do Micro Servo Tower Pro SG90. Fonte: O autor.**

Por fim, para realizar as conexões necessárias entre o Arduino e os servo motores, foi adquirido um kit contendo quarenta unidades de jumpers (cabos) machos de 20 cm de comprimento cada, identificados pela Figura 5:

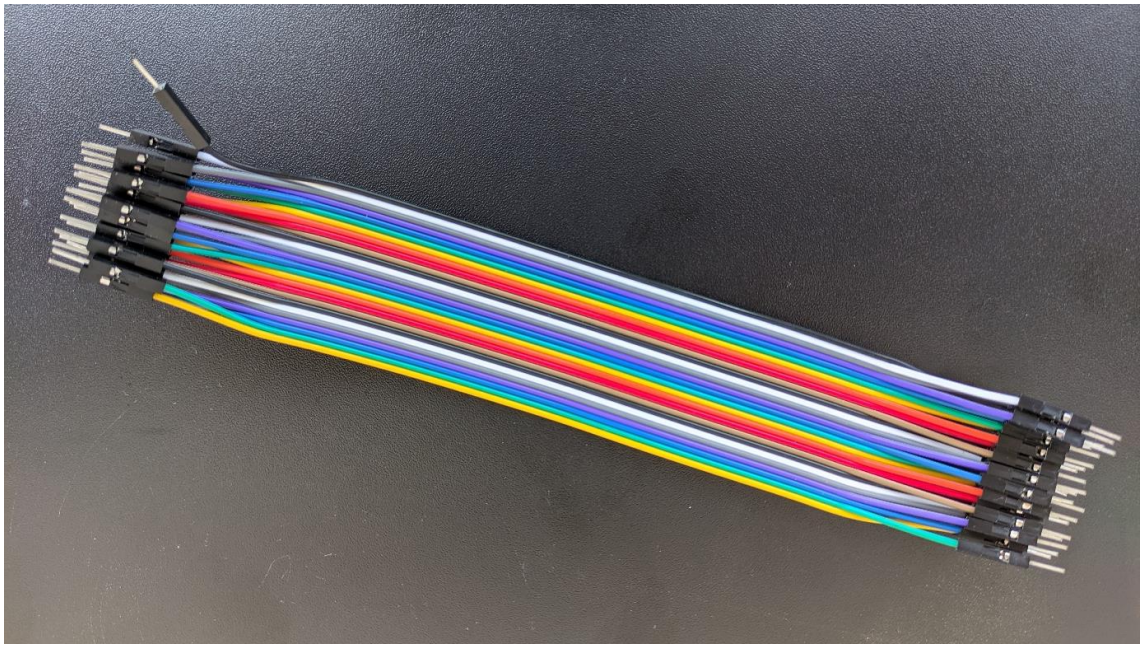


**Figura 5 – Foto do Kit de jumpers. Fonte: O autor.**

## 3.3. Componentes do projeto mecânico

O projeto mecânico consiste na elaboração de uma estrutura que seja capaz de se movimentar ao receber a atuação dos servos motores. Seu design, apesar de permitir diversas possibilidades, precisa estar alinhado com a proposta de realizar uma movimentação que se assemelhe a de um automóvel real.

O modelo idealizado para o projeto, pode ser categorizado como um padrão utilizado no universo dos simuladores de corrida conhecido como 2DOF, que é a sigla em inglês para *2 degrees of freedom*. O significado desta sigla, em termos práticos, informa o número de eixos diferentes que a estrutura mecânica pode se mover.

A construção da estrutura foi planejada para utilizar o mínimo de materiais possível e preferencialmente, materiais de fácil aquisição e baixo custo, desde que o funcionamento do protótipo não fosse comprometido. Ao todo, a estrutura mecânica é composta por sete componentes essenciais e um componente opcional, além da utilização de parafusos e fita adesiva para fixação dos componentes. A seguir, é exibida a Tabela 3, contendo a lista de componentes utilizados e também as Figuras 6, 7, 8 e 9, identificando cada um deles:

**Tabela 3 – Lista de componentes do Projeto Mecânico.**

| Componentes | Quantidade |
|---|---|
| Chapa de MDF:  30 x 21 x 0,5 cm | 01 (um) |
| Chapa de MDF: 15 x 21 x 0,5 cm | 02 (dois) |
| Suporte articulado para câmera de segurança | 02 (dois) |
| Haste de plástico: 7,5 cm | 02 (dois) |

| | |
|---|---|
| Miniatura de Fórmula 1 | 01 (um) |

**Figura 6 – Foto das três chapas de MDF. Fonte: O autor.**

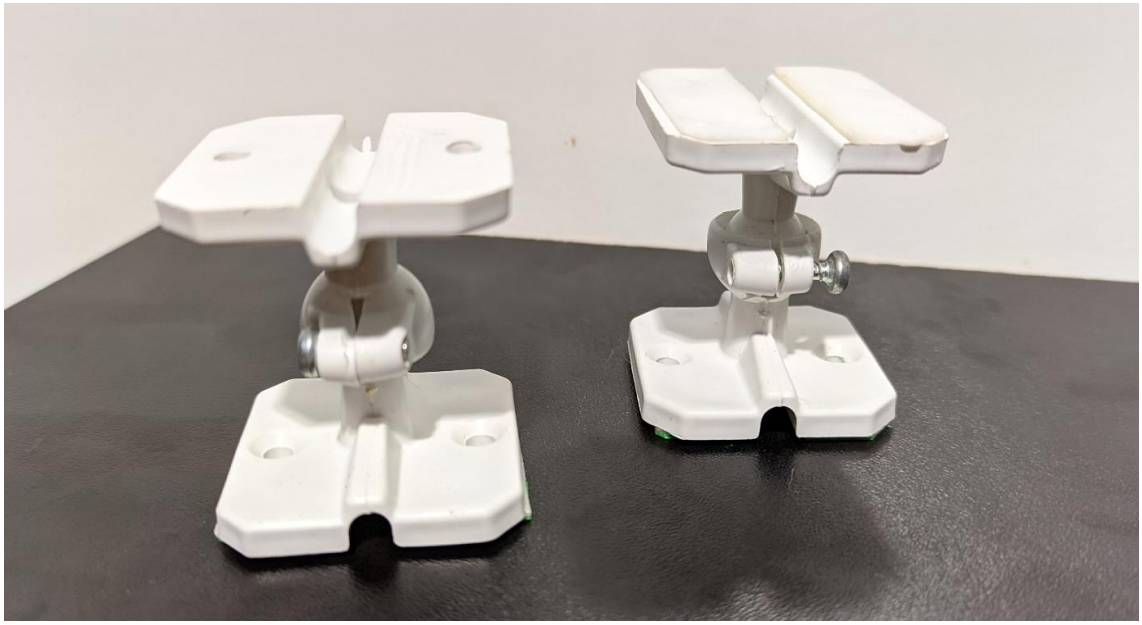


**Figura 7 – Foto dos dois suportes articulados para câmera de segurança. Fonte: O autor.**

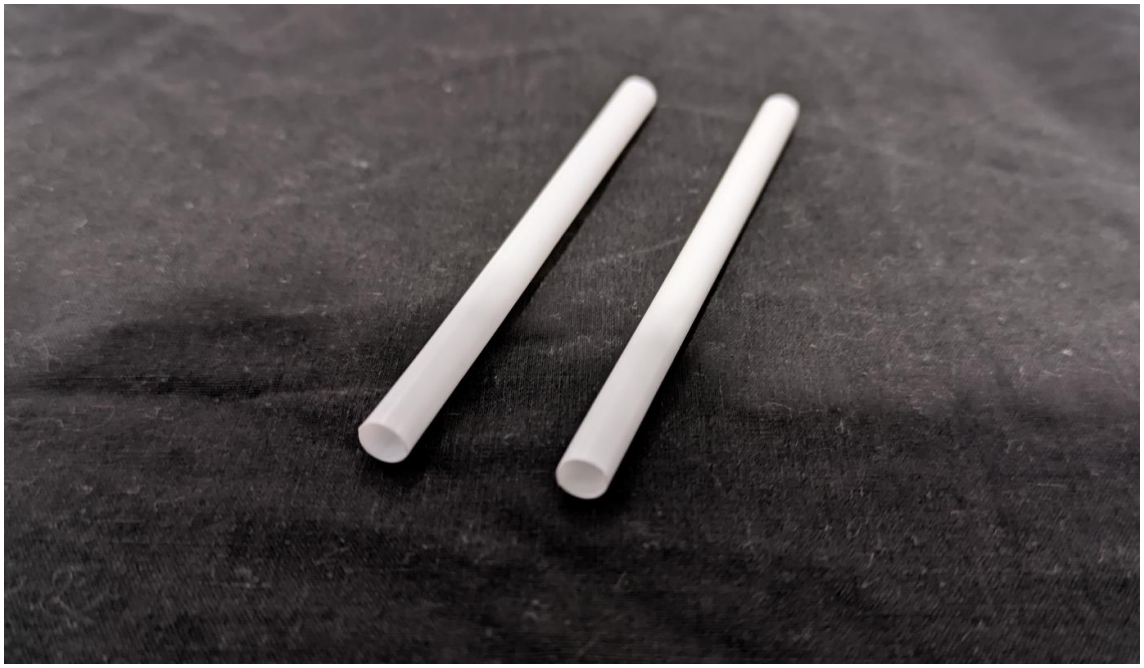


**Figura 8 – Foto das duas hastes de plástico. Fonte: O autor.**


**Figura 9 – Foto da miniatura de Fórmula 1. Fonte: O autor.**

### 3.4. Modelo do protótipo

Nas Figuras 10 e 11 abaixo, é possível visualizar o protótipo montado, com os componentes do projeto eletrônico e mecânico:

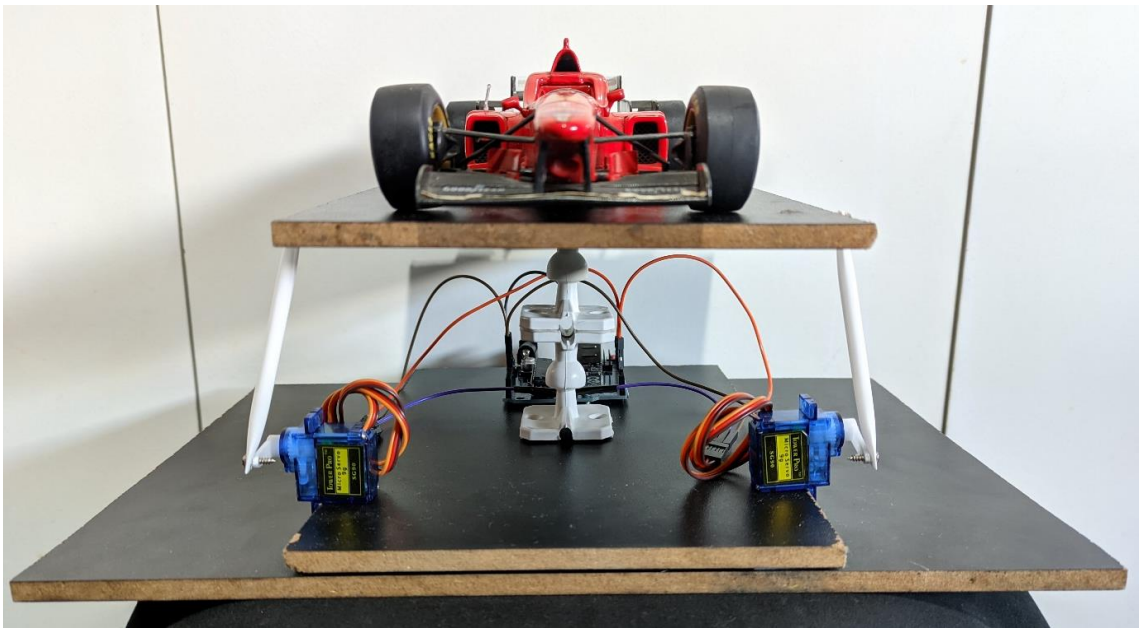


**Figura 10 – Foto frontal do protótipo montado. Fonte: O autor.**

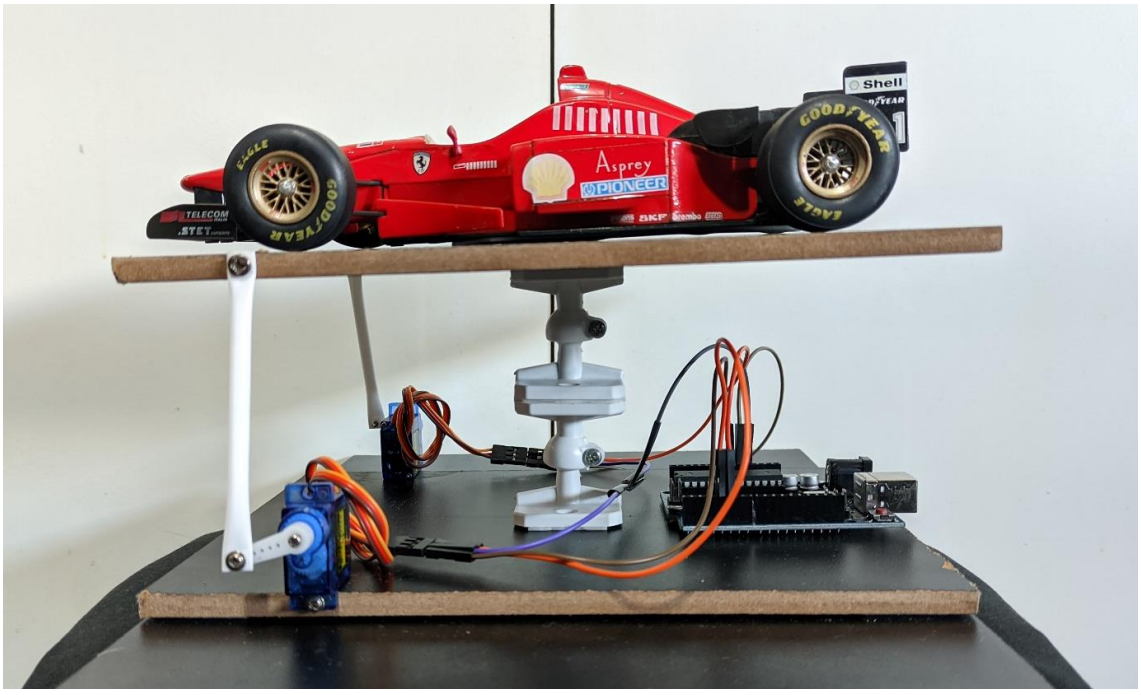



**Figura 11 – Foto lateral do protótipo montado. Fonte: O autor.**

A chapa de MDF maior serve como base de sustentação para toda a estrutura, ou seja, os demais componentes são todos montados sobre ela. As duas chapas de MDF menores,

possuem medidas iguais e são utilizadas como plataforma inferior e superior, para a fixação dos componentes do projeto eletrônico e mecânico. A plataforma inferior não possui movimento e é fixada à base, através de fita adesiva. A plataforma superior é móvel e é fixada na junta universal articulada, também com fita adesiva.

O primeiro suporte articulado para câmeras, é utilizado de forma fixa, apenas como adição de altura à estrutura mecânica. Foi fixado sobre plataforma de MDF inferior e acoplado ao suporte articulado número dois. O suporte número dois, é utilizado efetivamente como junta universal articulada, sendo fixado sob a plataforma de MDF superior e permitindo a sua movimentação. Todos estes elementos foram também fixados com fita adesiva.

Por motivos de praticidade, o Arduino UNO R3 foi inserido na parte central traseira da plataforma de MDF inferior. Desta forma, é possível transportar o protótipo sem a necessidade de desfazer as conexões por cabos entre o Arduino e os motores. Sua fixação foi realizada por parafusos

Os dois servo motores foram posicionados paralelos entre si e parafusados nas laterais frontais da plataforma de MDF inferior. Também com a utilização de parafusos, foram fixados os braços mecânicos aos eixos de rotação dos servo motores.

Duas hastes de plástico, com 7,5 cm de comprimento foram utilizadas com a função de atuadores lineares. Uma de suas extremidades é fixada através de parafusos no braço mecânico do servo motor e a outra extremidade é parafusada na plataforma de MDF superior. Estas hastes são responsáveis por converter a rotação dos servos motores em movimento linear, transmitindo a força destes, à plataforma de MDF superior.

A miniatura de Fórmula 1 é um componente opcional e não interfere no funcionamento do protótipo, porém, tem extrema importância em fornecer compreensão visual da movimentação da estrutura mecânica. Desta forma, permite avaliar com clareza o funcionamento do protótipo como um todo. A miniatura foi fixada no topo da plataforma de MDF superior e consequentemente irá se mover de acordo com o movimento da plataforma.

## 4. Metodologia

O software SimTools foi configurado de acordo com o projeto eletrônico e mecânico construídos. A configuração foi feita através da parametrização de algumas seções da interface gráfica *Game Engine*, iniciando pelo menu *Axis Assignments* do software, onde foi definido o número de atuadores utilizados no protótipo e quantos eixos de movimentação eles irão simular ou emular.

No protótipo, foram utilizados dois atuadores, definidos no SimTools como *Axis1a* e *Axis2a*, combinados, eles são capazes de simular efetivamente apenas dois eixos de movimentação, *Pitch* e *Roll*, porém, também são capazes de emular outros três eixos de movimentação, *Heave*, *Sway* e *Surge*, que serão explicados na Figura 14 e Tabela 4 apresentadas no decorrer deste artigo.

É importante destacar, que o termo simular é utilizado para definir a representação verdadeira do movimento. Um exemplo seria caso o carro se incline para a direita no jogo, o protótipo se inclina da mesma maneira no mundo real. Já o termo emular é utilizado para definir representações alternativas dos movimentos. Um exemplo seria que ao realizar uma

frenagem no jogo, deveria ocorrer um deslocamento linear da estrutura mecânica, trazendo a sensação de inércia ao piloto. Porém, como não existe um atuador dedicado para este movimento, este eixo de movimentação é representado no mundo real de forma alternativa, inclinando levemente o protótipo para a frente e proporcionando uma sensação similar à frenagem verdadeira. As Figuras 12 e 13 abaixo, apresentam as parametrizações realizadas no SimTools:


**Figura 12 – Parametrização dos eixos de movimentação Roll, Pitch e Heave. Fonte: O autor.**


**Figura 13 – Parametrização dos eixos de movimentação Sway e Surge. Fonte: O autor.**

Para entender melhor como cada um dos eixos de movimentação atuam no modelo, podemos utilizar como referência a Figura 14 e a Tabela 4, abaixo:

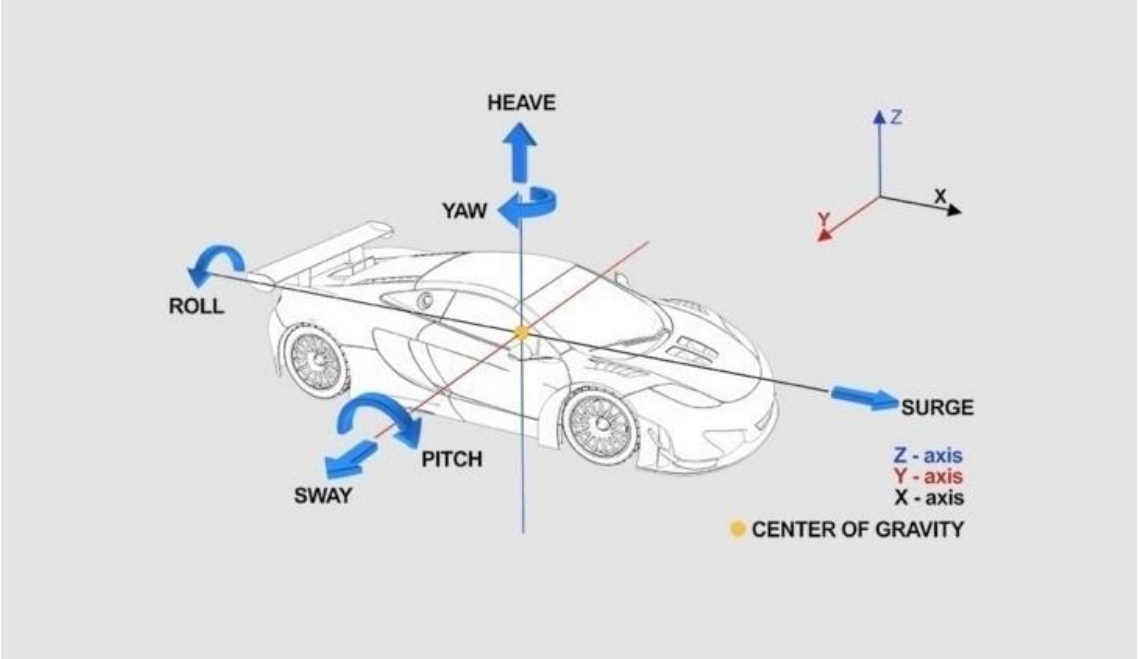



**Figura 14 – Imagem referencial dos eixos de movimentação. Fonte: XSimulator.**

**Tabela 4 – Eixos de movimentação e suas descrições.**

| Eixo de movimentação | Descrição |
|---|---|
| **Pitch** | Movimento de rotação sobre o eixo Y. Proporcionando ao protótipo movimentos de inclinação para cima e para baixo, simulando subidas e descidas. |
| **Roll** | Movimento de rotação sobre o eixo X. Proporciona ao protótipo movimentos de inclinação lateral, simulando curvas com inclinação, elevação da suspensão ao passar sobre a zebra, etc. |
| **Heave** | Deslocamento linear sobre o eixo Z. Proporciona ao protótipo deslocamento linear para cima e para baixo, emulando textura de relevos, trepidações, buracos e ondulações na pista. O protótipo não possui um atuador dedicado para este eixo de movimentação, mas é possível reproduzi-lo de forma satisfatória utilizando os eixos Pitch e Roll. |

| | |
|---|---|
| **Surge** | Deslocamento linear sobre o eixo X. Proporciona ao protótipo deslocamento linear para frente e para trás, emulando aceleração, frenagem, batidas frontais e traseiras e trocas de marcha. O protótipo não possui um atuador dedicado para este eixo de movimentação, mas é possível reproduzi-lo de forma satisfatória utilizando o eixo Pitch. |
| **Sway** | Deslocamento linear sobre o eixo Y. Proporciona ao protótipo deslocamento linear para a direita e para a esquerda, emulando a força G lateral quando o automóvel faz curvas e também batidas nas laterais. O protótipo não possui um atuador dedicado para este eixo de movimentação, mas é possível reproduzi-lo de forma satisfatória utilizando o eixo Roll. |
| **Yaw** | Movimento de rotação sobre o eixo Z. Proporciona ao protótipo movimento de rotação em torno do seu próprio eixo, simulando perda de tração nas rodas, derrapagens e deslizamento. O protótipo não possui um atuador dedicado para este eixo de movimentação, e também não é possível reproduzi-lo com o projeto atual. |

Após a parametrização dos eixos de movimentação, é necessário configurar outra seção da interface gráfica *Game Engine*, o menu *Interface Settings*. Este menu é responsável por realizar a comunicação com o Arduino. Nele definimos que os parâmetros do atuador *Axis1a* farão referência ao servo motor direito e os parâmetros do atuador *Axis2a* farão referência ao servo motor esquerdo. Também são parametrizadas as saídas de dados do software como é mostrado na Figura 15 abaixo:

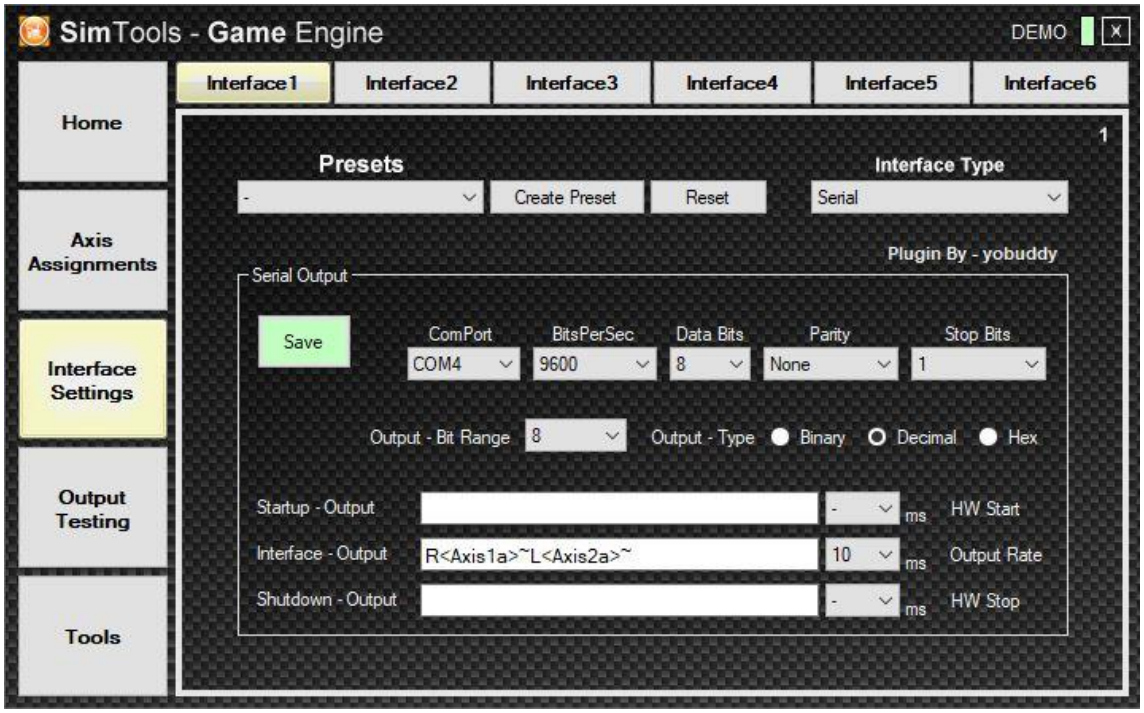



**Figura 15 – Parametrização dos eixos de movimentação Sway e Surge. Fonte: O autor.**

Para finalizar as configurações no SimTools, é necessário definir qual jogo de corrida será utilizado para a captura da telemetria do carro. Para cada jogo diferente, é necessário instalar um plugin específico, que realiza a captura de dados apenas para aquele determinado jogo. Utilizaremos o game *Live for Speed*, *freeware* e que possui plugin para uso com o SimTools.

A configuração de captura da telemetria é feita na interface gráfica *Game Manager*. Primeiramente é selecionado o jogo escolhido no menu *Game Selection* e depois indicamos o caminho para a captura de dados através da opção *Patch* e clicando no botão *Patch Game*, visualizado na Figura 16 a seguir. Em seguida, basta indicar o diretório onde o jogo foi instalado e o procedimento estará concluído.

Após indicar o diretório de instalação, a opção *Profile Editor* poderá ser configurada. Nesta interface, é possível visualizar quais dados de telemetria estão disponíveis no jogo e definir quais serão capturados, bem como a intensidade de transmissão destes dados. Como pode ser visto na Figura 17.


**Figura 16 – Configuração do Game Manager - Patch. Fonte: O autor.**

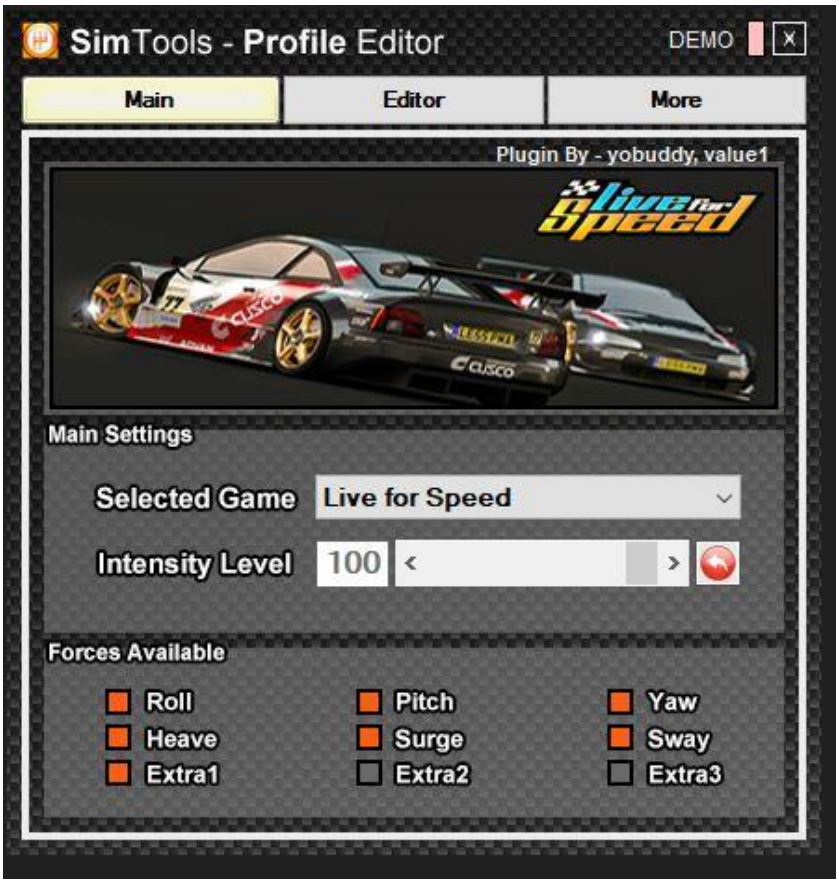
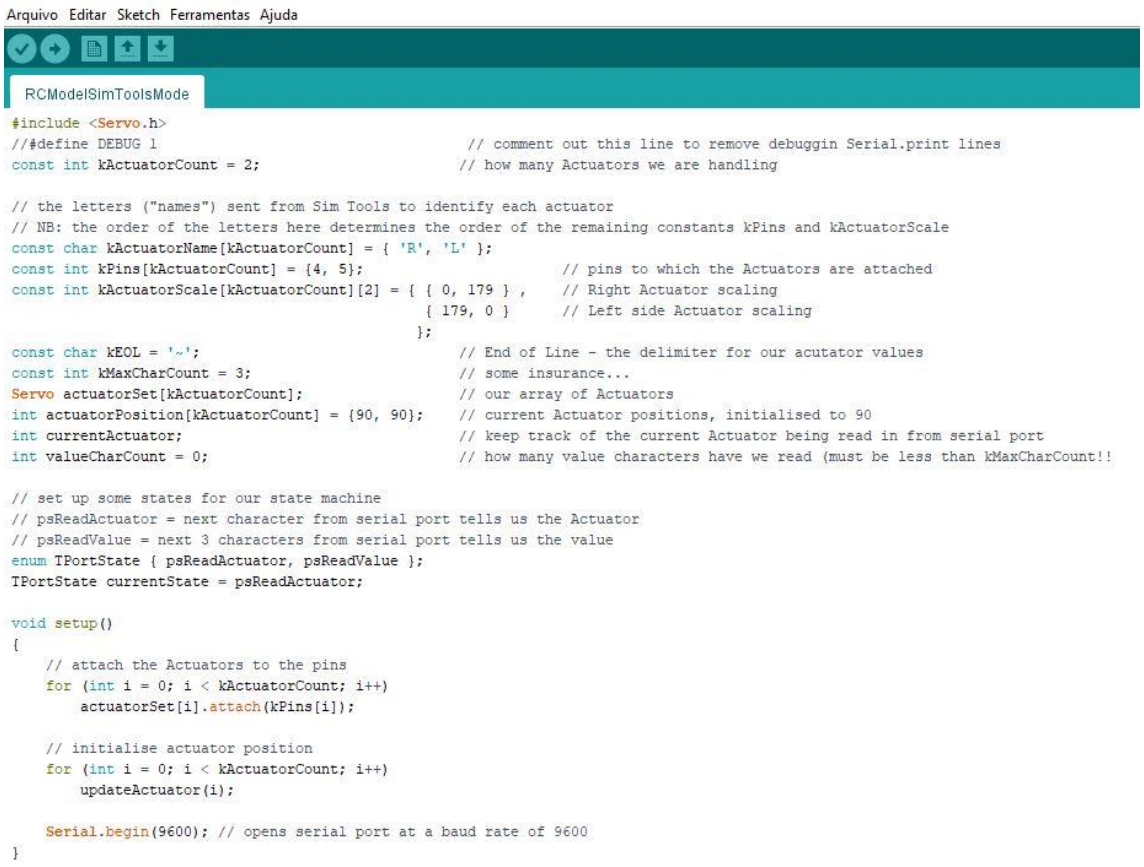



**Figura 17 – Configuração do Game Manager – Profile Editor. Fonte: O autor.**

Após finalizar as configurações do SimTools, o próximo passo é codificar o Arduino para receber e interpretar os dados enviados pelo software. Para executar tal ação, o Arduino deve ser conectado ao computador através de um cabo USB. Para codificá-lo, é utilizado um ambiente de programação próprio, o Arduino IDE, que pode ser instalado gratuitamente. Após o carregamento do código, não há necessidade de manter a IDE em funcionamento, pois o mesmo fica salvo na memória do Arduino. Abaixo, na Figura 18, podemos ver uma parte do código utilizado:




```
#include <Servo.h>
//#define DEBUG 1                                  // comment out this line to remove debuggin Serial.print lines
const int kActuatorCount = 2;                      // how many Actuators we are handling

// the letters ("names") sent from Sim Tools to identify each actuator
// NB: the order of the letters here determines the order of the remaining constants kPins and kActuatorScale
const char kActuatorName[kActuatorCount] = { 'R', 'L' };
const int kPins[kActuatorCount] = {4, 5};                       // pins to which the Actuators are attached
const int kActuatorScale[kActuatorCount][2] = { { 0, 179 } ,    // Right Actuator scaling
                                                { 179, 0 }      // Left side Actuator scaling
                                              };
const char kEOL = '~';                             // End of Line - the delimiter for our acutator values
const int kMaxCharCount = 3;                       // some insurance...
Servo actuatorSet[kActuatorCount];                 // our array of Actuators
int actuatorPosition[kActuatorCount] = {90, 90};   // current Actuator positions, initialised to 90
int currentActuator;                               // keep track of the current Actuator being read in from serial port
int valueCharCount = 0;                            // how many value characters have we read (must be less than kMaxCharCount!!

// set up some states for our state machine
// psReadActuator = next character from serial port tells us the Actuator
// psReadValue = next 3 characters from serial port tells us the value
enum TPortState { psReadActuator, psReadValue };
TPortState currentState = psReadActuator;

void setup()
{
    // attach the Actuators to the pins
    for (int i = 0; i < kActuatorCount; i++)
        actuatorSet[i].attach(kPins[i]);

    // initialise actuator position
    for (int i = 0; i < kActuatorCount; i++)
        updateActuator(i);

    Serial.begin(9600); // opens serial port at a baud rate of 9600
}
```

**Figura 18 – Amostra do código utilizado no Arduino. Fonte: O autor.**

Para que o Arduino se comunique com os servo motores, devem ser conectados através dos jumpers. Cada servo motor possui três conexões, uma de alimentação (cabo vermelho), uma de sinal (cabo laranja) e uma de aterramento (cabo marrom). O Arduino UNO R3 dispõe apenas de uma saída de 5V, utilizada para alimentação, portanto para evitar a necessidade do uso de um *protoboard*, foi realizada uma emenda num dos jumpers, criando um cabo em formato T, que permitiu a conexão de dois servo motores, numa única saída de alimentação de 5V. Os cabos de sinal foram conectados às portas digitais 4 e 5, que foram definidas no código utilizado no Arduino. Os cabos de aterramento, foram ligados às portas GND. O diagrama de conexões entre o Arduino e os servo motores pode ser visto na Figura 19, abaixo:

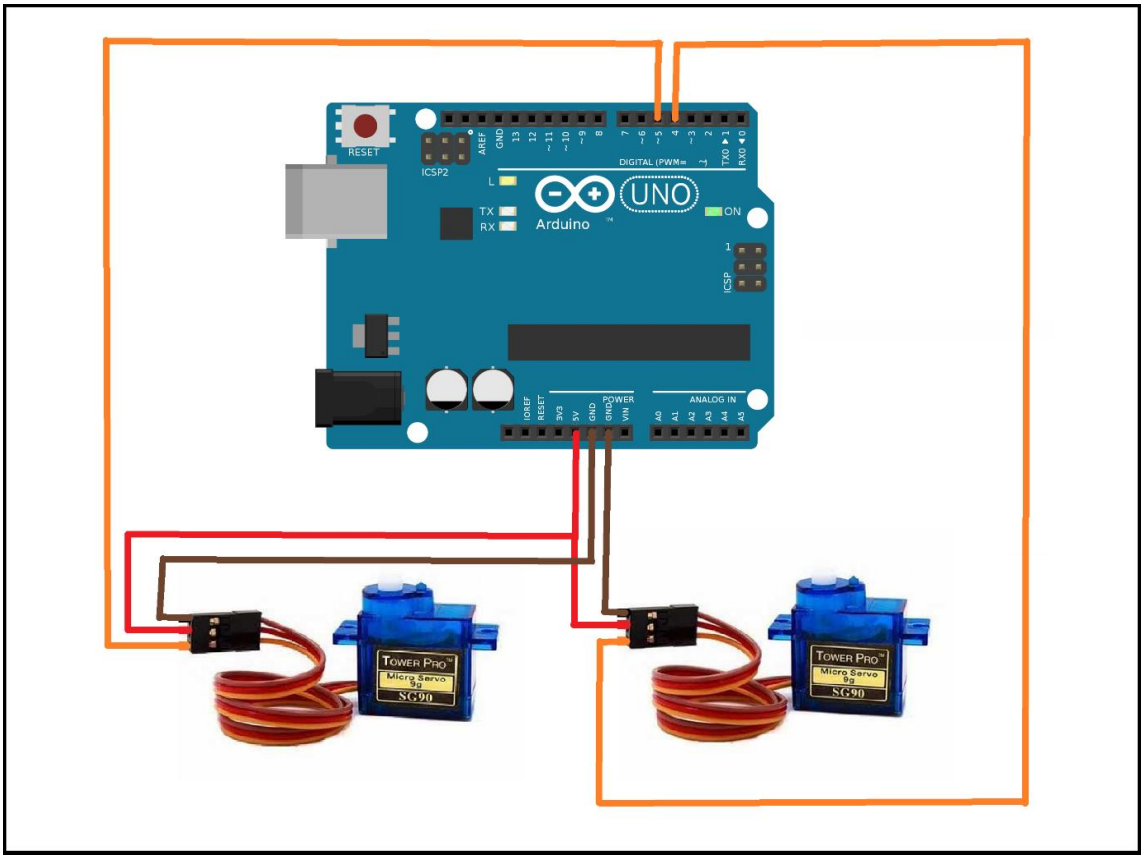



**Figura 19 – Diagrama de conexões entre o Arduino e os servo motores. Fonte: O autor.**

## 5. Funcionamento do Protótipo

O funcionamento do protótipo como um todo, se dá a partir da configuração e conexão entre software, projeto eletrônico e projeto mecânico. Para testar o funcionamento dos eixos de movimentação, o SimTools possui a opção *Output Testing*, que faz parte da interface gráfica *Game Engine*. A função principal desta opção, é verificar o funcionamento de cada um dos eixos de movimentação de forma individual, evitando que algum problema ocorra durante a utilização efetiva do protótipo, onde a todos os eixos de movimentação serão utilizados simultaneamente. Para realizar os testes, basta clicar no botão *Turn On* e utilizar as barras de rolagem para movimentar cada um dos eixos.

Após realizar os testes individuais e verificar o funcionamento correto de cada um dos eixos de movimentação, basta abrir o jogo *Live for Speed*, para utilizar o protótipo. Um vídeo demonstrando o funcionamento completo, pode ser encontrado na seguinte URL: https://youtu.be/_vVuIUErsw8. Abaixo, podem ser visualizadas a Figura 20 que exibe a opção *Output Testing* e a Figura 21, que exibe uma captura de tela do vídeo demonstrando o funcionamento do protótipo:

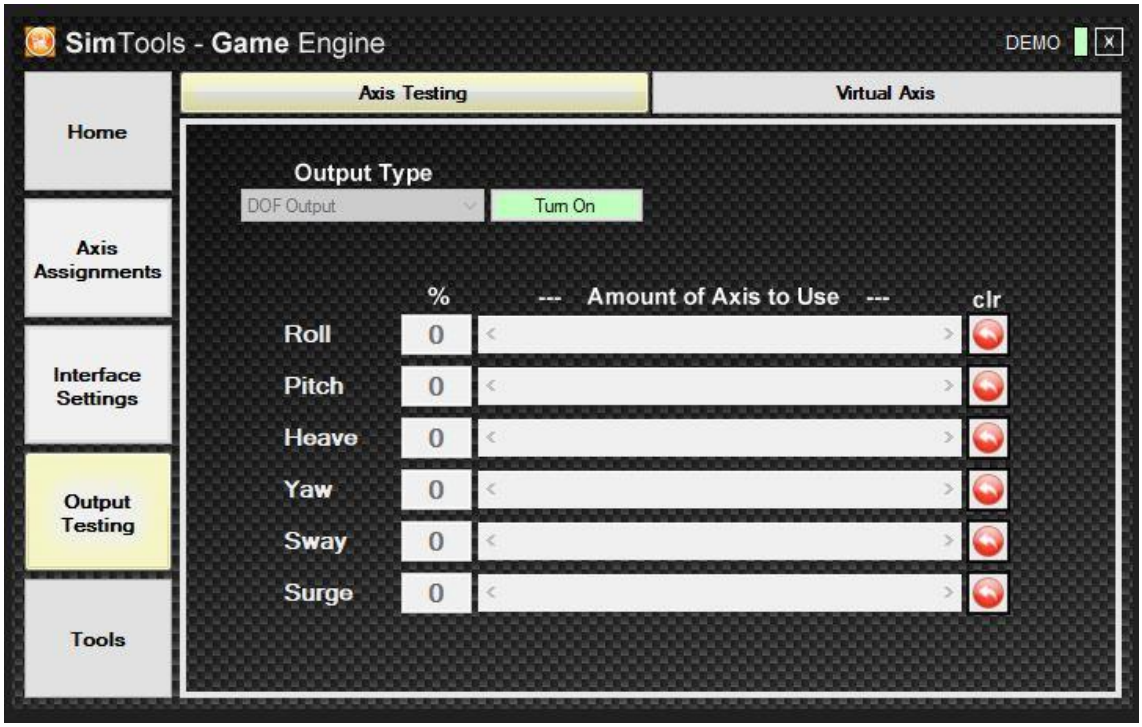



**Figura 20 – Interface da opção Output Testing. Fonte: O autor.**


**Figura 21 – Captura de tela do vídeo, demonstrando o funcionamento do protótipo. Fonte: O autor.**

## 6. Resultados e discussões

O protótipo cumpriu os requisitos estabelecidos para o projeto e seu funcionamento correspondeu ao esperado. O SimTools demonstrou ser capaz de realizar a captura da telemetria do jogo *Live for Speed* e permitiu a parametrização de todos os eixos de

movimentação presentes no protótipo. A comunicação com o Arduino, também ocorreu de forma eficiente, pois, não foi possível detectar nenhum tipo de atraso nas transmissões dos dados movimento para os servo motores, o que demonstra que a placa controladora supriu as necessidades de forma adequada. O design elaborado para a estrutura mecânica, utilizou apenas materiais de baixo custo e fácil aquisição, proporcionando ao protótipo uma estrutura capaz de representar os movimentos de um automóvel.

Unindo o software, projeto eletrônico e projeto mecânico, o protótipo foi testado de forma plena com o jogo *Live for Speed*, e foi possível constatar que com a utilização de apenas dois eixos de movimentação efetivos, *Pitch* e *Roll*, o mesmo foi capaz de simular de forma satisfatória os movimentos de um automóvel, incluindo também os eixos emulados *Heave*, *Sway* e *Surge*. As nuances da pista, como inclinações, ondulações e a zebra, puderam ser observadas de forma clara na movimentação da plataforma, bem como as forças atuantes sobre o carro durante as curvas, aceleração, frenagem e colisões.

## 7. Conclusões

O protótipo foi construído de acordo com os requisitos e através do seu funcionamento, foi possível observar a simulação dos movimentos de um automóvel de um jogo de corrida. A construção foi realizada utilizando materiais de baixo custo para os projetos mecânico e eletrônico, totalizando um valor aproximado de 200 reais, com exclusão da miniatura de Fórmula 1, que se trata de um item de coleção e possui valor variável para sua aquisição. É possível concluir, que o protótipo desenvolvido pode ser utilizado como base para um projeto futuro, em tamanho real. Também foram identificados pontos de melhoria durante o desenvolvimento, porém, seria necessária uma mudança no escopo e aquisição de novos componentes, aumentando consideravelmente o custo do projeto.

## 8. Trabalhos Futuros

Durante o desenvolvimento do protótipo, foi possível observar um ponto de melhoria, que iria trazer um nível extra de autenticidade à simulação dos movimentos de um automóvel. A adição de um terceiro servo motor, para atuar no eixo de movimentação *Yaw*. Esta adição proporcionaria ao protótipo o movimento de rotação em torno do seu próprio eixo, simulando perda de tração nas rodas, derrapagens e deslizamento. Para que esta melhoria pudesse ser incluída, seria necessário planejar um novo projeto mecânico, que permita o movimento de rotação, bem como a aquisição de componentes extras para a composição da estrutura.

Outro ponto de destaque, é que uma das propostas iniciais para o desenvolvimento deste projeto, foi a possibilidade de construir um protótipo em miniatura, que pudesse servir como base para o desenvolvimento futuro de um simulador de movimentos em tamanho real. Tendo em vista o funcionamento adequado do protótipo, a ideia se mostra viável e poderia seguir os mesmos padrões de design implementados na miniatura. Porém, para prosseguir, será necessário aprofundar os estudos da parte mecânica e eletroeletrônica, pois na construção de um protótipo em tamanho real, alguns fatores que podem ser negligenciados na miniatura, possuem extrema importância para o funcionamento e também para a segurança. A

potência dos motores, resistência dos materiais e capacidade da estrutura para sustentar o peso de um jogador adulto, são itens que necessitam de atenção e dimensionamento adequado, para que o funcionamento do simulador não seja comprometido.

## REFERÊNCIAS


MENDES, Jaqueline. **Indústria de games cresce e se profissionaliza cada vez mais.** 16 out. 2019. Disponível em: https://www.em.com.br/app/noticia/economia/2019/10/16/internas_economia,1093076/industria-de-games-cresce-e-se-profissionaliza-cada-vez-mais.shtml. Acesso em: 10 ago. 2021.

CRAVO, Edilson. **O que é um Servo Motor, como funciona e quais as vantagens?** Disponível em: https://blog.kalatec.com.br/o-que-e-servo-motor/. Acesso em: 10 ago. 2021.

XSIMULATOR: **Simtools complete Documentation.** Disponível em: https://www.xsimulator.net/simtools-complete-documentation/. Acesso em: 10 ago. 2021.

ARDUINO: **Software Downloads.** Disponível em: https://www.arduino.cc/en/software/. Acesso em: 10 ago. 2021.

ARDUINO: **Documentation.** Disponível em: https://www.arduino.cc/reference/en/. Acesso em: 10 ago. 2021.

ORSOLIN, Denver Marques; RAMPELOTTO, Eduardo R.. **Simulador de Direção com Três Graus de Liberdade.** 2010.

CHILDRESS, Robert. **Motion Platform Video Game Racing and Flight Simulator.** 2012.

NODARI, Christine Tessele; VERONEZ, Mauricio Roberto; BORDIN, Fabiane; JR, Luiz Gonzaga da Silveira; DE OLIVEIRA, Maurício Castilhos; LAROCCA, Ana Paula Camargo; FRAMARIM, Carlo. **Avaliação do Realismo e da Sensação de Mal Estar (Simulator Sickness) no Uso de Simulador Imersivo de Direção.** 2017.

SIAM, M. F. Mohd; BORHAN, N.; SUKARDI, A.. **Driving Simulator Development with Two Degrees of Freedom Motion for Driver Behavior Study**. 2017.

LO, Patrick K.; DIETRICH, Robert A.. **Portable and Compact Motion Simulator.** 1999.