INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SANTA CATARINA – CAMPUS ITAJAÍ
DEPARTAMENTO ACADÊMICO DE ELETROELETRÔNICA
CURSO SUPERIOR DE ENGENHARIA ELÉTRICA

FABRICIO SIMÕES FONSECA HERMES

IMPLEMENTAÇÃO DE UM SISTEMA DE FASES E TAREFAS UTILIZANDO O SIMULADOR 3D DE EMPILHADEIRA DO TIPO *REACH STACKER* PARA TREINAMENTO E RECICLAGEM DE OPERADORES

ITAJAÍ – SC
SETEMBRO, 2022

FABRICIO SIMÕES FONSECA HERMES

IMPLEMENTAÇÃO DE UM SISTEMA DE FASES E TAREFAS UTILIZANDO O SIMULADOR 3D DE EMPILHADEIRA DO TIPO *REACH STACKER* PARA TREINAMENTO E RECICLAGEM DE OPERADORES

Trabalho de Conclusão de Curso submetido ao Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina, como parte dos requisitos para a obtenção do Título de Engenheiro Eletricista.

Orientador: Prof. Sergio A. B. Petrovcic, Dr. Eng.

ITAJAÍ – SC

SETEMBRO, 2022

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca do IFSC.

Hermes, Fabricio
IMPLEMENTAÇÃO DE UM SISTEMA DE FASES E TAREFAS UTILIZANDO O SIMULADOR 3D DE EMPILHADEIRA DO TIPO REACH STACKER PARA TREINAMENTO E RECICLAGEM DE OPERADORES / Fabricio Hermes ; orientador, Sergio Petrovcic, 2022.
63 p.

Trabalho de Conclusão de Curso (graduação) - Instituto Federal de Santa Catarina, Campus Itajaí, Graduação em Engenharia Elétrica , Itajaí, 2022.

Inclui referências.

1. Engenharia Elétrica . 2. simulação 3D. 3. Empilhadeira de grande porte. 4. Reach stacker. 5. Unity3D. 6. Treinamento. I. Petrovcic, Sergio . II. Instituto Federal de Santa Catarina. Graduação em Engenharia Elétrica . III. Título.

FABRICIO SIMÕES FONSECA HERMES

IMPLEMENTAÇÃO DE UM SISTEMA DE FASES E TAREFAS UTILIZANDO O SIMULADOR 3D DE EMPILHADEIRA DO TIPO *REACH STACKER* PARA TREINAMENTO E RECICLAGEM DE OPERADORES

Este trabalho foi julgado adequado para obtenção do título em Engenharia Elétrica, pelo Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina, e aprovado na sua forma final pela comissão avaliadora abaixo indicada.

Itajaí, 08 de setembro, 2022

IFSC

Documento assinado digitalmente
SERGIO AUGUSTO BITENCOURT PETROVCIC
Data: 27/09/2022 10:00:20-0300
CPF: 026.649.159-67
Verifique as assinaturas em https://v.ifsc.edu.br

Prof. Sergio A. B. Petrovcic, Dr. Eng.

Orientador

IFSC

Documento assinado digitalmente
GUILHERME RANZOLIN PIAZZETTA
Data: 27/09/2022 10:08:16-0300
CPF: 048.696.069-25
Verifique as assinaturas em https://v.ifsc.edu.br

Prof. Guilherme Ranzolin Piazzetta, Msc.

GOV.BR

Documento assinado digitalmente
ENIO DOS SANTOS SILVA
Data: 27/09/2022 10:12:00-0300
CPF: ***.778.252-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Ênio dos Santos Silva, Msc.

Dedico este Trabalho de Conclusão de Curso a todas as pessoas que estiveram me apoiando nos últimos anos para obtenção do título em Engenharia Elétrica

AGRADECIMENTOS

Agradeço à minha mãe Patrícia Simões e ao meu pai Fábio Fonseca por todo o apoio que foi me dado para chegar até aqui, tanto no sentido financeiro, como no sentido emocional e intelectual, sem vocês nada seria possível. Vocês são a minha base e sempre serão um exemplo para mim.

Agradeço aos meus familiares, principalmente minha irmã Letícia Simões pelo apoio, espero sempre servir como um bom exemplo para você.

Agradeço a minha namorada Kelly Nitz pelo apoio emocional, pela paciência e pela compreensão nos diversos momentos de dificuldade que foram surgindo durante o curso e este projeto de TCC.

Agradeço aos meus amigos, tanto aos mais próximos como aos que estão mais distantes, em especial ao Gabriel Vaillant, por todo o suporte emocional durante toda essa caminhada.

Agradeço ao IFSC e aos seus professores e funcionários. Em especial ao meu Professor e orientador Sergio Petrovcic por todo apoio e incentivo nos últimos anos para desenvolvimento deste projeto de TCC e entre outros projetos.

Agradeço também ao Grupo Impulsus, empresa na qual trabalho, por todo o apoio e colaboração nos últimos meses para que eu pudesse finalizar este projeto.

RESUMO

Com o advento da tecnologia, a utilização de simuladores para o treinamento de profissionais se torna cada vez mais comum. O presente trabalho tem como intuito implementar um sistema de fases e tarefas em simulador 3D para empilhadeiras do tipo *Reach Stacker*, através da plataforma *Unity3D*, visando auxiliar no treinamento prático e reciclagem dos operadores. O simulador apresenta um *cockpit* montado de maneira similar à cabine da máquina, apresentando volante e pedais, trazendo uma maior imersão ao usuário. O ambiente virtual apresenta um modelo 3D da empilhadeira, contendo as principais funções que a máquina realiza. O sistema de fases tem como intuito facilitar o primeiro contato com a máquina através do simulador, e o sistema de tarefas busca apresentar um *feedback* ao usuário, conforme as tarefas vão sendo concluídas. Foi possível verificar que o simulador auxilia no aprendizado dos profissionais, além de representar uma certa economia, sendo necessário deslocar a empilhadeira por menos tempo das suas funções principais para exclusivamente o treinamento.

Palavras-chave: simulação 3D, empilhadeira de grande porte, *reach stacker*, *Unity3D*, treinamento.

ABSTRACT

With the advent of technology, the use of simulators for professional training becomes increasingly common. The work aims to implement a system of phases and tasks in a 3D simulator for forklifts of the Reach Stacker type, through the Unity3D platform, helping in the practical training and recycling of operators. The features a larger simulator and a prototype alongside the assembled machine. The virtual environment presents a 3D model of the structure, containing the performed machine as its main functions. The phase system is intended to facilitate the first contact with the machine through the simulator, and the task system seeks to provide feedback to the user as the tasks are completed. The simulator was able to assist in the professionals' learning, verifying the representation of a necessary time, in addition to allowing the maintenance of a tool for less important functions for exclusive training.

Keywords: 3D simulation, large forklift, reach stacker, *Unity3D*, training.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

RV – Realidade virtual

RS – *Reach Stacker*

2D – Duas dimensões

3D – Três dimensões

IE – Instituições de Ensino

NR – Normas Regulamentadoras

SUMÁRIO

1. INTRODUÇÃO

Uma empilhadeira é um equipamento móvel que tem como função primária empilhar produtos ou cargas em armazéns, fábricas ou portos, em ambientes abertos ou fechados. Para executar sua função primária, a empilhadeira precisa transportar e manobrar a carga, por percursos que podem variam conforme a demanda e em superfícies e espaços apropriados. As empilhadeiras podem ser separadas em diversos grupos, levando em consideração o seu porte e a maneira com a qual irá transportar a carga. O foco desse trabalho se dará nas empilhadeiras de grande porte, que são aquelas que possuem capacidade de levantamento (*lift*) superior a 10 toneladas, sendo muito utilizadas para deslocamento de diversos tipos de cargas, sendo a mais comum os contêineres (INCATEP, 2020).

O uso de empilhadeiras de grande porte deve muito às características funcionais que a mesma apresenta, como possibilitar em um único equipamento o auto carregamento e transporte da carga. Assim como a facilidade de manobrar a carga, aceitando alterações de percursos e pontos de carga de descarga (CAMPOS, 2019).

As empilhadeiras de grande porte podem ser divididas em 4 grandes grupos:

- Empilhadeiras de garfo: possibilitam a elevação e transporte da carga através das patolas (garfos);
- Empilhadeiras de mesa: funcionamento similar às empilhadeiras de garfo, já que também possui um eixo de elevação vertical, entretanto, são especializadas para contêineres e possuem os grafos virados para o solo;
- Empilhadeiras de mesa frontal: sendo um grupo bem específico, utilizado somente para o transporte de contêineres vazios, acoplando somente os dois *locks* dianteiros ao mesmo;
- Empilhadeiras de lança: também chamadas de *Reach Stacker* (RS), é o modelo de empilhadeira mais versátil se tratando de movimentação de contêineres, devido a sua lança telescópica, permitindo deslocamento da carga tanto na vertical como na horizontal (INCATEP, 2020).

As empilhadeiras tipo *reach stacker* (Figura 1) são empilhadeiras de grande porte que possuem a capacidade de manobrar contêineres, descarregados ou carregados, com rapidez e eficiência em espaços estreitos (BORGES, 2011).

Figura 1: *Reach stacker* modelo DRG450.

Fonte: (*TECHNICAL INFORMATION KALMAR CONTAINER HANDLER*, 2020).

As empilhadeiras apresentam diversas configurações de montagem que as diferenciam dos automóveis comuns, como para sua operação, sendo necessário estar com uma mão no volante e outra nos comandos que executam as funções da máquina, diferentemente de um automóvel, onde o correto é estar a maior parte do tempo com as duas mãos no volante. Seu funcionamento se baseia em deslocamento de carga por distâncias curtas, perdendo economia para distâncias acima de 100 metros. Apresentam tração dianteira e direção traseira, possibilitando um giro das rodas de até 90°, sendo outro ponto divergente dos automóveis que em sua maioria apresentam direção dianteira e tração traseira. Deslocam-se com a mesma velocidade tanto para frente, como para trás e possuem como ponto de apoio as rodas dianteiras (CAMPOS, 2019).

Além das configurações de montagem, apresentam certos fatores de risco próprios, além dos riscos que um automóvel comum apresenta, como: ruptura do sistema de elevação devido a sobrecargas, ausência de sinalização necessária na área de operação, necessidade de entendimento da sinalização, falta de padrões operacionais pré-determinados, falta de equipamento de sinalização nas máquinas, pisos irregulares, entre outros diversos fatores (CAMPOS, 2019).

1.1 Justificativa

O porto de Itajaí, em Santa Catarina (Figura 2), administrado pela empresa APM Terminals, possui 14 empilhadeiras tipo RS, que em conjunto com outras empilhadeiras e máquinas presentes no porto, foram responsáveis por manobrar cerca de 4,5 mil contêineres por dia, se destacando pelo recorde histórico de movimentação, conforme trecho abaixo:

> "O Complexo Portuário de Itajaí encerrou 2021 com recorde histórico na movimentação de contêineres em TEU's (unidade de medida equivalente a um contêiner de 20 pés). De janeiro a dezembro, foram movimentados 1.643.152 TEU's e 18.945.270 toneladas contra 1.419.082 TEU's e 15.655.812 toneladas no mesmo período do anterior. O dado representa um aumento de 16% em TEU's e 21% na tonelagem. Em 2021, foi registrado crescimento de 2% nas atrações, com 1.066 navios recebidos no complexo." (MUNICÍPIO DE ITAJAÍ, 2022)

Figura 2: Porto de Itajaí (SC).

Fonte: (*FLICKR APM TERMINALS, 2020)*

A informação citada anteriormente, serve para ilustrar como a empilhadeira RS podem movimentar uma quantidade muito grande de produtos, dos mais diversos tipos, dentro dos contêineres, e devido a isso, podem apresentando riscos e complexidades de operação únicos. Cada pequeno acidente pode causar danos significativos, tanto à vida, como financeiramente,

já que a não disponibilidade de uma RS impacta em um custo de R$ 9.300,00 por hora, considerando que cada movimentação, tem por tabela pública, um custo de R$ 310,00 e a máquina faz um movimento a cada 2 minutos (APM, 2022).

Para poder operar a RS, é necessário realizar um curso específico de operador de empilhadeira de grande porte, oferecido em Instituições de Ensino (IE) credenciadas. O curso possui carga horária teórica, abrangendo as Normas Regulamentadoras (NR) aplicáveis, sendo elas: a NR-06, de Equipamentos de Proteção Individual, a NR-11 que trata sobre Transporte, Movimentação, Armazenagem e Manuseio de Materiais, e a NR-12, sobre Segurança no Trabalho em Máquinas e Equipamentos. Outros temas são abordados como questões de segurança, princípio de funcionamento da máquina, partes do equipamento, rotina de transporte (empilhamento e desempilhamento) e, por fim, o exame teórico final.

Na parte prática, é feito o primeiro contato com o equipamento e aulas de operação, abordando pontos como:

- Controle da máquina;
- Painel de instrumentos;
- Técnicas de operação;
- Mudança de direção e velocidade;
- Estacionamento da máquina;
- Procedimento de testes operacionais;
- Técnicas operacionais;
- Entre outros demais pontos.

Tendo em mente o que foi pontuado nos parágrafos anteriores, a utilização de simuladores para auxiliar no treinamento de operadores novos, para diminuir o nervosismo presente no primeiro contato com a RS e aperfeiçoamento de operadores antigos, contribuindo para a diminuição de acidentes.

Simuladores virtuais podem suprir, em parte, o treinamento prático. De acordo com (TORI, 2018), a utilização de simuladores envolve um custo elevado e muito restrito a certos nichos, devido às complexidades envolvidas nas tecnologias que vão ser empregadas. Entretanto, com o passar dos anos, a popularização, barateamento e a acessibilidade de certas ferramentas de desenvolvimento, tornou-se possível o desenvolvimento de simuladores de custo inferior que diversas áreas podem utilizar para treinamento.

O potencial presente nos simuladores para a educação e treinamento, transborda para projetos e aplicações eficazes e específicas, que podem contribuir para o treinamento de profissionais de diversas áreas, como já ocorre com pilotos de avião que devem cumprir um determinado período de treinamento em simuladores. Outro exemplo, são os países que adotaram os simuladores de automóveis para o treinamento de motoristas (Tori et al., 2018).

1.2 Objetivos

1.2.1 Objetivo geral

O desenvolvimento de um simulador de RS surgiu na disciplina de projeto Integrador 2, utilizando um software de simulação distinto. Foi dado continuidade na disciplina de Projeto Integrador 3 em conjunto com um projeto de pesquisa, esse já utilizando o software de simulação que foi utilizado para este Trabalho de Conclusão de Curso (TCC). Sendo assim, o objetivo geral deste TCC é:

- Implementar as funções da empilhadeira tipo *reach stacker* em simulador 3D, baseadas em fases e tarefas para treinamento de operadores.

1.2.2 Objetivos específicos

Como objetivos específicos têm-se:

- Apresentar as principais funções da máquina e reproduzi-las no simulador;
- Criar interface gráfica no simulador contando com indicadores de velocidade, lista de tarefas, cronômetro, entre outros;
- Criar cenas dentro do simulador, seguindo a ideia de fases como um jogo, para o entendimento das funções da máquina, tornando o aprendizado mais intuitivo;
- Disponibilizar o simulador para utilização de pessoas que nunca tiveram contato com a máquina e pessoas que já tiveram para obter o nível de satisfação na utilização do mesmo.

2. REVISÃO DE LITERATURA

Neste capítulo serão apresentados conceitos teóricos envolvendo os simuladores 3D, apresentando uma definição de conceitos como: realidade virtual, *hardware*, *software* e *game engine*, com o intuito de facilitar o entendimento no que se refere à parte técnica do trabalho.

Apresentará também exemplos de simuladores 3D utilizados para treinamento de profissionais de diversas áreas, assim como exemplos de jogos que utilizam o conceito de simulação para o entretenimento.

2.1 Realidade virtual

Com o decorrer dos tempos, o ser humano sempre cultivou maneiras de representar a realidade a sua volta ou até mesmo suas imaginações, passando desde pinturas rupestres até o cinema. Incluindo os jogos, as pinturas, o teatro e outras formas de representações artísticas. Textos, imagens, sons, vídeos e animações permitiram, em conjunto com o avanço dos computadores, desenvolver ambientes não lineares e interativos de navegação.

Com o auxílio da popularização dos *vídeos games*, a integração entre o ambiente real e o virtual foi se tornando cada vez mais natural, não se limitando a um plano de interação, mas sim expandindo para ambientes tridimensionais (3D) em tempo real, ficando conhecido como "realidade virtual" (TORI E HOUNSELL, 2006).

A RV envolve avançadas tecnologias de interface, permitindo ao usuário realizar imersões, navegações e interações em um ambiente artificial 3D, desenvolvido em computador (KIRMER E PINHO, 1996).

Segundo (VINCE, 2004), o ambiente virtual possibilita a sua visualização a partir de qualquer ponto de vista conforme vão sendo feitas alterações em tempo real, e para ele isso é uma grande vantagem da RV comparada a outras formas de interações do homem com o computador.

Valores, atributos, comportamentos e resultados podem ser atribuídos aos objetos do ambiente, possibilitando uma simulação do real no ambiente virtual. Utilizando de ferramentas e/ou dispositivos pertencentes e integrados ao ambiente virtual, como: controles, luvas, volantes, *joysticks*, pedais e entre outros, é possível simular estes tipos de interações e observar os resultados. Além disso, essas ferramentas trazem ao usuário a imersão ao virtual, utilizando de objetos reais para interagir com o ambiente 3D em tempo real, possibilitando a exploração e manipulação desse ambiente artificial (STUART, 1996; VINCE, 2004).

Existem diversas definições para o termo “realidade virtual”. Para (TORI E KIRMER, 2006) pode ser definido como:

> “A Realidade Virtual (RV) é, antes de tudo, uma “interface avançada do usuário” para acessar aplicações executadas no computador, tendo como características a visualização de, e movimentação em, ambientes tridimensionais em tempo real e a interação com elementos desse ambiente. Além da visualização em si a experiência do usuário de RV pode ser enriquecida pela estimulação dos demais sentidos como tato e audição”.

2.1.1 Hardware

Para (KIRMER E SISCOUTTO, 2007), o *hardware* de um computador se trata do núcleo para qualquer aplicação, possuindo os recursos necessários para sua utilização. Apresenta características que são responsáveis para o melhor custo-benefício, com base nas necessidades da aplicação.

O *hardware* de um sistema de RV abrange uma diversidade de dispositivos de entrada, que tem como função auxiliar o usuário a comunicar-se com o sistema. Dispositivos, como: reconhecedores de voz, luvas eletrônicas, *mouses* 3D, teclados, *joysticks*, rastreadores, entre outros (TORI, KIRMER E SISCOUTTO, 2006).

Os *displays* de sistemas de RV são elementos sensoriais de saída, envolvendo mais sentido do que unicamente a visão. Podemos classificá-los como: *displays* de áudio, *displays* de vídeo e *displays* hápticos (TORI, HOUNSELL E KIRMER, 2018).

Outro componente muito importante para a RV são os processadores. Devido a constantes avanços da tecnologia e das tendências presentes no mercado de *video games*, os processadores acompanham esses avanços. Estão divididos entre processadores principais e processadores de apoio em placas gráficas, sonoras ou de outra função específica (TORI, HOUNSELL E KIRMER, 2018).

Segundo (COSTA, 2018), o desempenho de um sistema de RV que compõe um dispositivo está relacionado à qualidade envolvida nos componentes presentes nele, citando exemplos como “...sofisticação da resolução das imagens, precisão dos rastreadores, nível e controle da capacidade sonora.”. Eles devem possuir a qualidade de registrar, tratar e fornecer esses dados sensoriais, detectados pelos softwares de entrada, em tempo real, para que, dessa maneira, a interação dentro da RV seja de maneira fluida e natural, não causando desconforto ao usuário. De maneira geral ela completa dizendo:

> "A integração harmoniosa dos componentes de um ambiente virtual exige vários tipos de controle de software e de equipamentos. Cada modalidade sensorial requer um controle específico, enquanto que uma ação integrada coordena e sincroniza as várias modalidades sensoriais envolvidas." (COSTA, 2018)

2.1.2 Software

Devido a sua imersão entre o real e o virtual, os sistemas devem responder de maneira rápida às alterações de comando efetuadas em tempo real pelo usuário. Sendo assim, os *softwares* de RV são divididos em dois: os que atuam na fase de preparação do sistema, como *software* de autoria de ambientes 3D, e os que atuam na fase de execução, como tempo de execução (TORI, KIRMER E SISCOUTTO, 2006).

O *software* de autoria está ligado à resposta do simulador com base na alteração realizada pelo usuário, e pode envolver: linguagens de programação, bibliotecas gráficas ou mesmo *game engines*.

Segundo (TORI, KIRMER E SISCOUTTO, 2006), como tempo de execução, o *software* de um sistema de RV tem que: "...interagir com os dispositivos especiais; cuidar da interface com o usuário; tratar de visualização e interação; controlar a simulação/animação do ambiente virtual; e implementar a comunicação em rede para aplicações colaborativas remotas."

As *game engines* (motores de desenvolvimento de jogos) fazem parte do *software* e são comumente utilizados para simplificar e abstrair o desenvolvimento de jogos eletrônicos ou outras aplicações com gráficos em tempo real. Existem diversos *game engines* que podem ser utilizados para apoiar o desenvolvimento de ambientes virtuais.

Game engines têm sido a opção preferida dos desenvolvedores, principalmente *Unreal* e *Unity 3D*, dada a facilidade propiciada por seus ambientes de desenvolvimento, por oferecerem suporte para a maioria dos dispositivos e óculos de realidade virtual do mercado, e por gerarem aplicativos e executáveis para diferentes plataformas e sistemas operacionais. Esses dois motores citados são comerciais mas oferecem licenciamento gratuito para uso pessoal e/ou sem fins lucrativos. A preparação dos ambientes virtuais envolve modelagem 3D, preparação e manipulação de texturas, manipulação de som, elaboração de animações etc.

2.2 Simuladores 3D para treinamento profissional

Na aviação, além das aulas teóricas e práticas é de muita importância o uso dos simuladores, segundo é possível afirmar durante algumas horas de simulador se o aluno será ou não um piloto, pois ali é de imediato a percepção dos instrutores sobre os alunos pelo desenvolvimento de cada um. (SALVATORE, 2007).

Já para as escolas de aviação, ter um simulador, mesmo sendo de pequeno porte e baixo custo, causa um aumento na motivação dos alunos em fase de aprendizado, criando um ambiente seguro de treinamento, pois os alunos obtêm um maior conhecimento da aeronave e suas atitudes de voo (MORENO, 2003).

Além disso, o treinamento em simulador de voo minimiza os custos da hora de voo, segundo (MORENO, 2003). os simuladores devem ser eficientes e ter um custo operacional baixo, para que treine pilotos com segurança, realismo e principalmente economia. Segundo o mesmo autor, os treinamentos em simuladores auxiliam a realizar e treinar manobras a baixa altura com realismo e segurança.

2.3 Características físicas da empilhadeira reach stacker

As empilhadeiras têm como ponto de apoio as rodas dianteiras, sendo o peso da carga contrabalançando por um contrapeso existente na parte traseira do veículo, por isso devemos carregar a empilhadeira respeitando sua capacidade de carga.

A capacidade de carga de uma empilhadeira está condicionada a dois fatores: peso de carga e distância entre o centro de gravidade da carga e o ponto de apoio da empilhadeira. Quando essa distância aumenta, reduz a capacidade de carga a ser transportada, pois aumenta a tendência da perda de estabilidade da empilhadeira.

A empilhadeira é baseada sob o princípio de dois pesos colocados em sentidos opostos, como uma espécie de gangorra. A localização do centro de gravidade da máquina e da carga é um princípio básico para levantar a carga. Ou seja, a capacidade da empilhadeira possui relação direta com o centro de gravidade, estabilizado frontal e lateralmente. Vale ressaltar que a capacidade de carga varia conforme o modelo de RS e de contêiner.

A estabilidade frontal da máquina está relacionada com a capacidade que a máquina tem para suportar uma carga sem risco de capotamento frontal ou de elevar as rodas traseiras durante o processo de içamento do contêiner.

A localização do centro de gravidade é um aspecto importante para a estabilidade da empilhadeira. Sendo o centro de gravidade definido como o ponto onde há o equilíbrio do

objeto, é necessário que sua localização seja tal que permita a movimentação vertical ou horizontal da carga sem perigo de tombamento da máquina.

Considerando somente a empilhadeira, ou seja, sem carga, o ponto correspondente ao centro de gravidade está, aproximadamente, localizado próximo ao motor da máquina. Porém quando a empilhadeira está deslocando uma carga, o centro de gravidade se altera, já que seu posicionamento se modifica conforme a movimentação feita com a carga.

Para manter a estabilidade da empilhadeira, o centro de carga do conjunto carga mais a empilhadeira deve permanecer dentro da base da máquina, representado por um triângulo, formado pelo eixo da tração dianteira e o ponto central do eixo de direção das rodas traseiras.

Para o desenvolvimento deste simulador, foi necessário implementar todas as movimentações e funções realizadas pela empilhadeira para o deslocamento do contêiner. Cada função será descrita com um pouco mais de profundidade nos tópicos a seguir, onde serão apresentados os principais componentes da máquina.

Na figura 3 estão apresentadas duas configurações que a *reach stacker* pode apresentar para deslocamento e transporte do contêiner.

Figura 3: Configurações de movimentação da RS.

Fonte: (*TECHNICAL INFORMATION KALMAR CONTAINER HANDLER*, 2020).

2.3.1 Sistema de direção e tração das rodas

Conforme já foi comentado neste trabalho, ao contrário dos automóveis comuns, a RS possui a tração nas rodas dianteiras, onde apresenta um conjunto de 4 rodas. Já as rodas traseiras são responsáveis pelo sistema de direção da máquina. Na figura 4, estão destacadas as rodas presentes na empilhadeira.

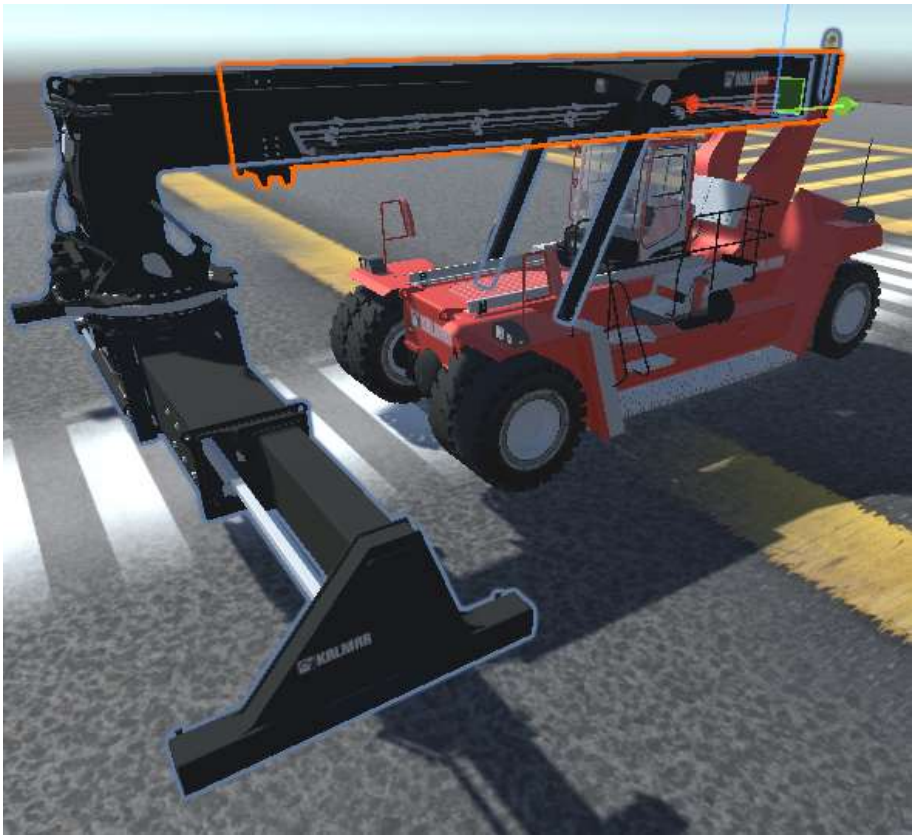
Figura 4: Rodas presentes na RS.

Fonte: Autor.

2.3.2 Lift boom

O *lift boom*, ou simplesmente *boom*, apresentado na figura 5, é a lança telescópica com elevação responsável por movimentar o contêiner quando o mesmo já está acoplado à máquina. Este componente é responsável pelas principais funções no que se diz respeito a pegar e empilhar os contêineres.

Figura 5: *Lift boom* da RS.



Fonte: Autor.

As funções realizadas por esse grupo são erguer e abaixar a lança, conforme figura 6, que possibilita pegar o contêiner no chão e colocá-lo no alto de uma pilha de contêineres, ou retirá-lo da pilha e colocá-lo em um caminhão para transporte.

Figura 6: B*oom* da RS levantado.

Fonte: Autor.

Já na figura 7, é possível observar outra função realizada pelo *boom*, a de esticar e recolher a lança. Essa função também é muito importante no momento de empilhamento dos contêineres, já que assim a empilhadeiras pode formar ou desfazer pilhas de cotêineres que estão mais distantes, atrás de objetos ou até mesmo atrás de outras pilhas

Figura 7: B*oom* da RS estendido.

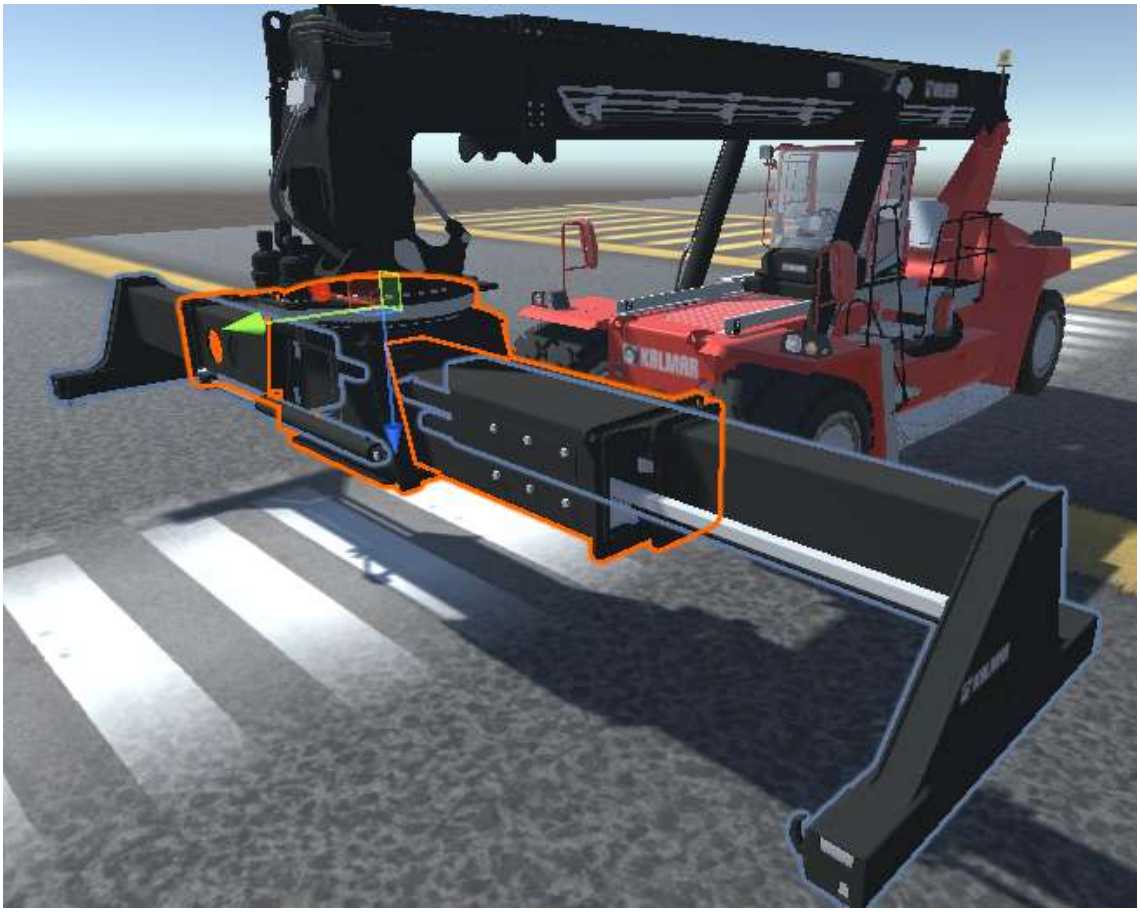
Fonte: Autor.

2.3.3 Spreader

O *spreader* (Figura 8) é um conjunto de componentes que faz parte do *lift boom*. Esse grupo realiza diversas funções na máquina, além de ser o componente que efetivamente se acopla ao contêiner, o segurando durante a movimentação.

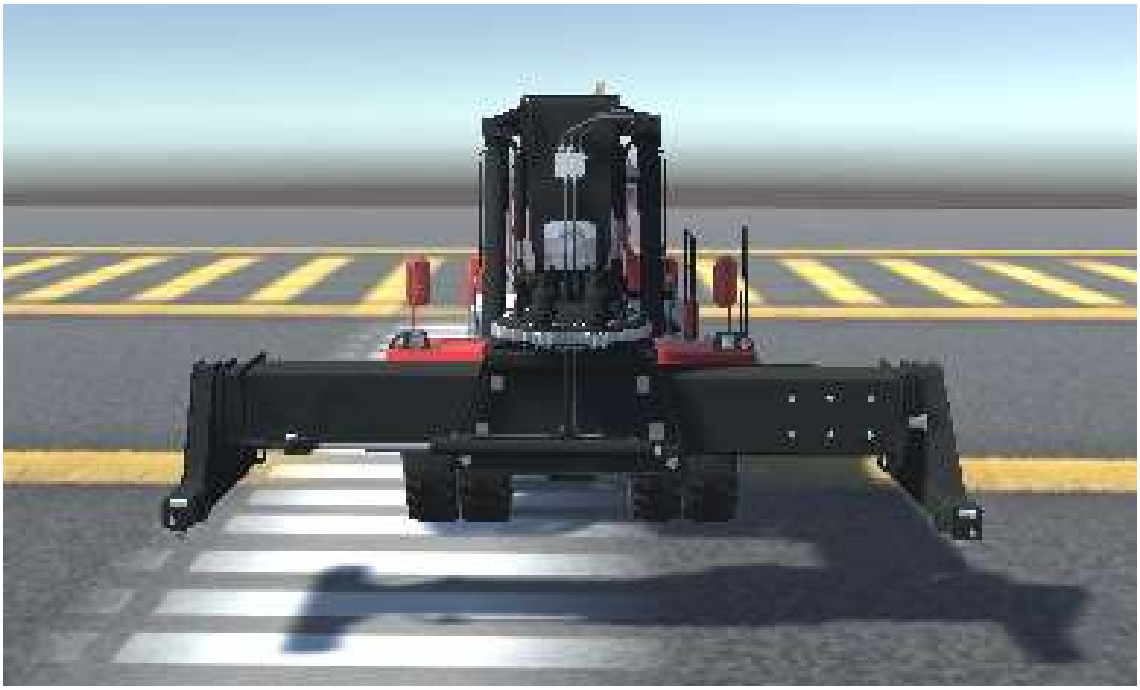
O *spreader* possui dois modos de configuração para acoplar os contêineres, sendo na configuração para contêineres de 20’ e para contêineres de 40’, e essa alteração de formato do *spreader* está representado nas figuras 9 e 10, respectivamente.

Figura 8: Conjunto central do *Spreader* da RS.



Fonte: Autor.

Figura 9: *Spreader* para a configuração de 20’ pés.



Fonte: Autor.

Figura 10: *Spreader* para a configuração de 40’ pés.

Fonte: Autor.

Apresenta outras duas funções que são de extrema importância para a movimentação dos contêineres que são a de deslocar o *spreader* para esquerda e para direita (Figura 11), possibilitando um ajuste fino no momento de empilhar ou retirar contêineres e a função que possibilita rotacionar o *spreader*.

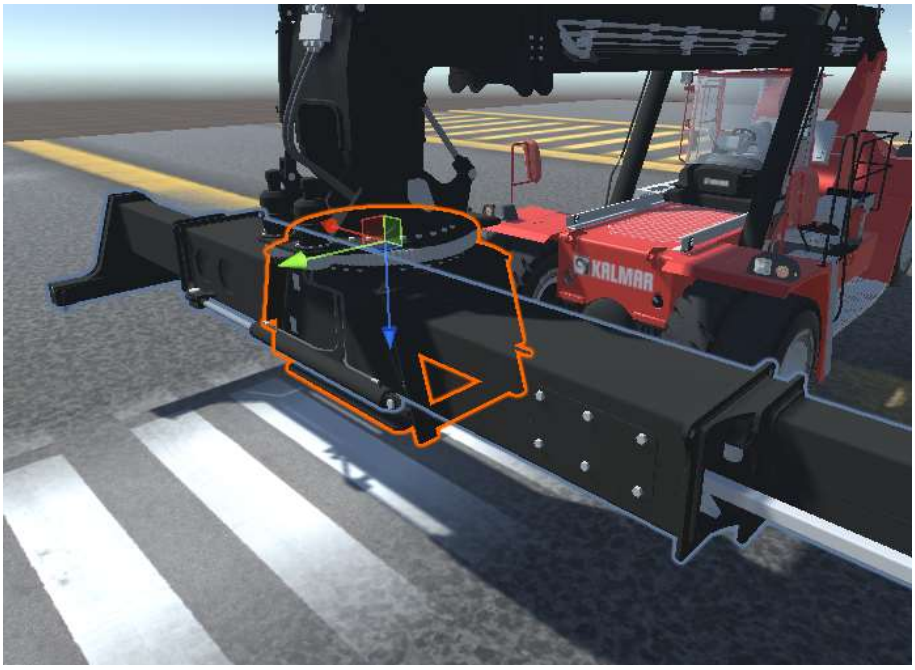
Figura 11: *Spreader* da RS deslocado para a direita.

Fonte: Autor.

2.3.4 Rotador do spreader

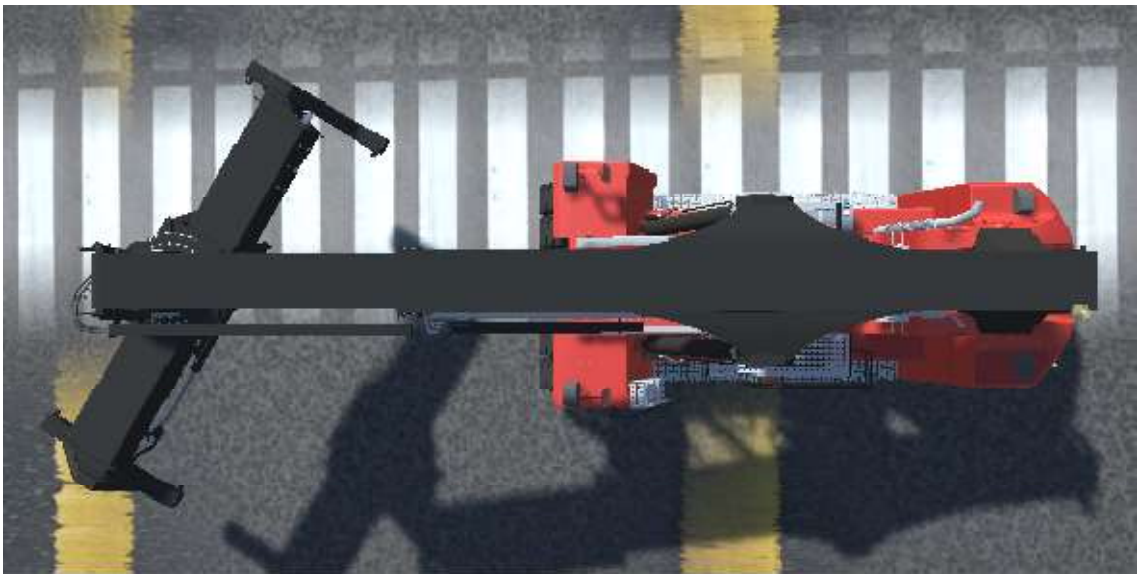
Sendo mais específico, a função de rotacionar o *spreader* é realizada pelo rotador do *spreader* em conjunto com motores e engrenagens presentes no *spreader* (Figura 12). A rotação pode ser tanto no sentido anti-horário como no sentido horário (Figura 13) e serve para facilitar o acoplamento do contêiner à máquina, não precisando ficar necessariamente um de frente ao outro.

Figura 12: Componente central do *Spreader* da RS.



Fonte: Autor.

Figura 13: *Spreader* da RS rotacionado.



Fonte: Autor.

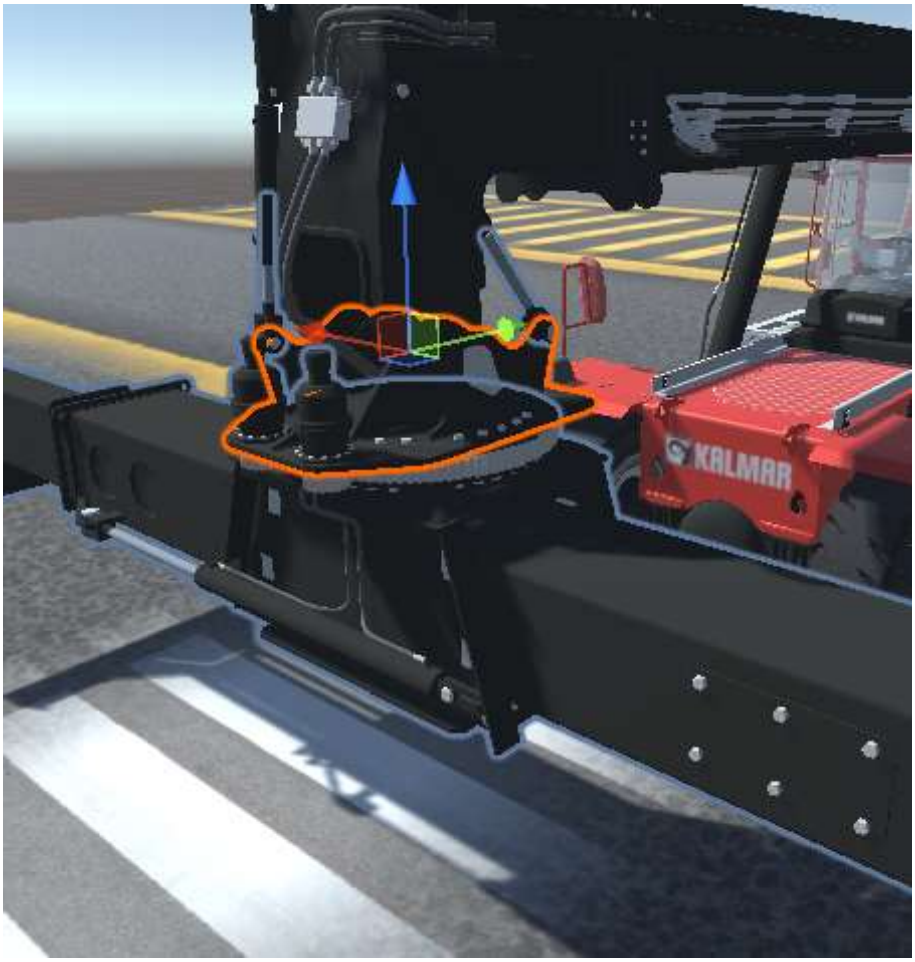
Além de facilitar no momento de empilhar e retirar contêineres, não tendo a necessidade de parar perpendicularmente à pilha, possui uma função muito importante no momento de deslocamento do contêiner quando a empilhadeira está se deslocando para distâncias médias, onde é comum e recomendado que o contêiner esteja alinhado com a empilhadeira.

Desta maneira, ajuda a evitar colisões do contêiner que está sendo movido com outras pilhas, construções, objetos ou equipamentos presentes no mesmo ambiente.

2.3.5 Eixo de inclinação do spreader

Na figura 14 é possível observar o componente que é responsável pela função de inclinação do *spreader*. O movimento é realizado pelos dois pistões que compõem este conjunto, entretanto, é muito comum, principalmente em portos e centros logísticos, que essa função seja desabilitada por constantemente apresentar defeitos devido a estresse. Vale ressaltar que desabilitar esta função não acarreta de maneira significativa na eficácia da movimentação dos contêineres e que, apesar de desabilitada, o eixo continua apresentando o balanço durante a movimentação devido ao peso do contêiner.

Figura 14: Junção do *Spreader* da RS.

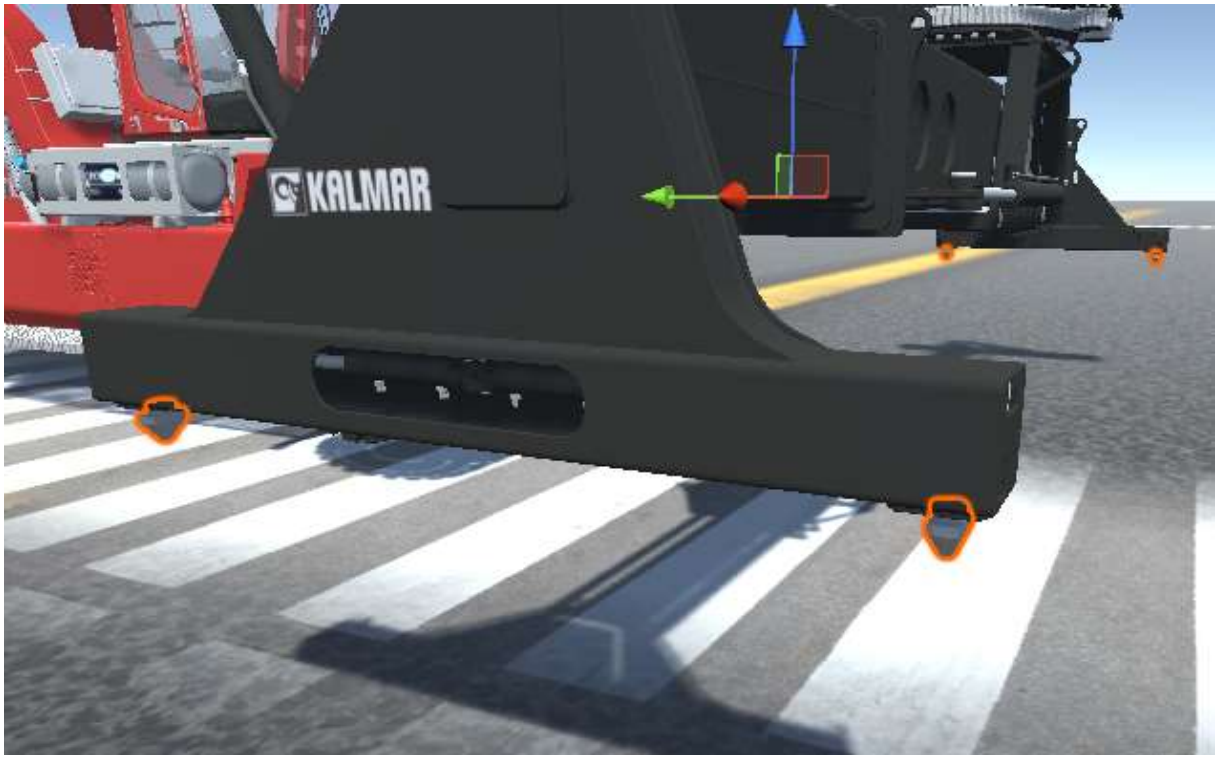


Fonte: Autor.

2.3.6 Pino lock

O pino *lock* é um conjunto de quatro peças que fazem parte do *spreader* e são efetivamente os responsáveis por acoplar o contêiner à máquina. Na figura 15 é possível observar os 4 pinos que compõem o *spreader*, dois em cada lateral do mesmo.

Figura 15: Pinos *Locks* da RS.



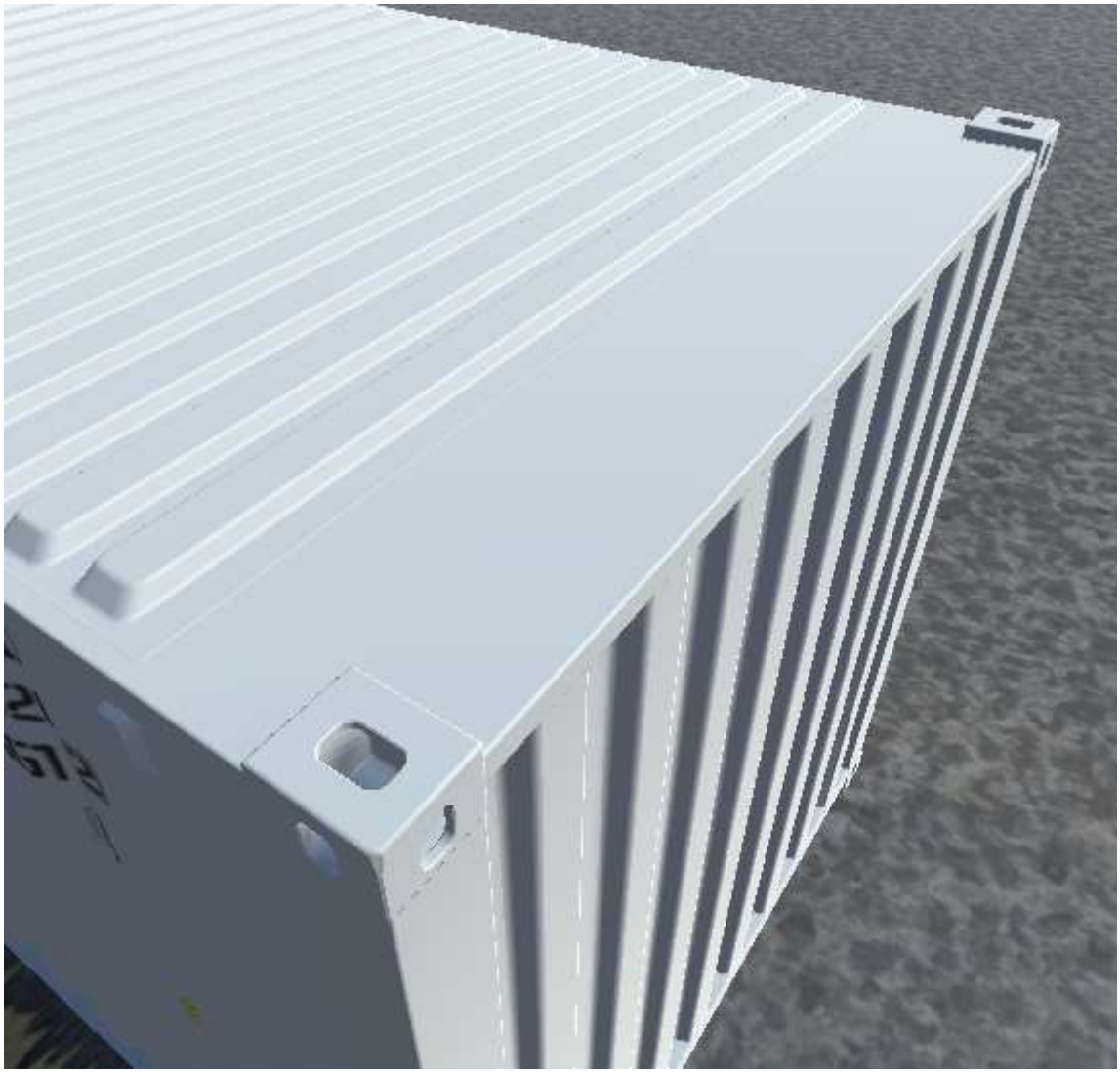
Fonte: Autor.

Seu acionamento ocorre a partir do *joystick* da máquina, assim como todas as outras funções que foram citadas, com exceção da movimentação da máquina, comandada pelo volante e pelos pedais.

Para facilitar o acoplamento do contêiner ao *spreader* através dos pinos, existe uma indicação luminosa na máquina, chamada de três Marias, composto por 3 indicadores luminosos: vermelho, laranja e verde, cada uma indicando um *status* ao operador. O vermelho indica que os *locks* estão destravados, o branco indica que o posicionamento dos *locks* está correto para acoplamento ao contêiner e o verde indica que os *locks* estão travados.

Então, com o auxílio dos indicadores luminosos, os quatro *locks* devem ser posicionados internamente nas quatro “bolsas” do contêiner (figura 16) e acionados. Desta maneira, eles rotacionam e encaixam o contêiner ao *spreader* da empilhadeira.

Figura 16: Bolsas dos contêineres.



Fonte: Autor.

3. MATERIAIS E MÉTODOS

Neste capítulo serão abordados os materiais e métodos que foram utilizados para o desenvolvimento deste trabalho, contemplando tanto componentes virtuais, no caso programas como o *Unity*, 3DS *Max* e o *Visual studio*, como componentes físicos, no caso, os que compõem o *cockpit*.

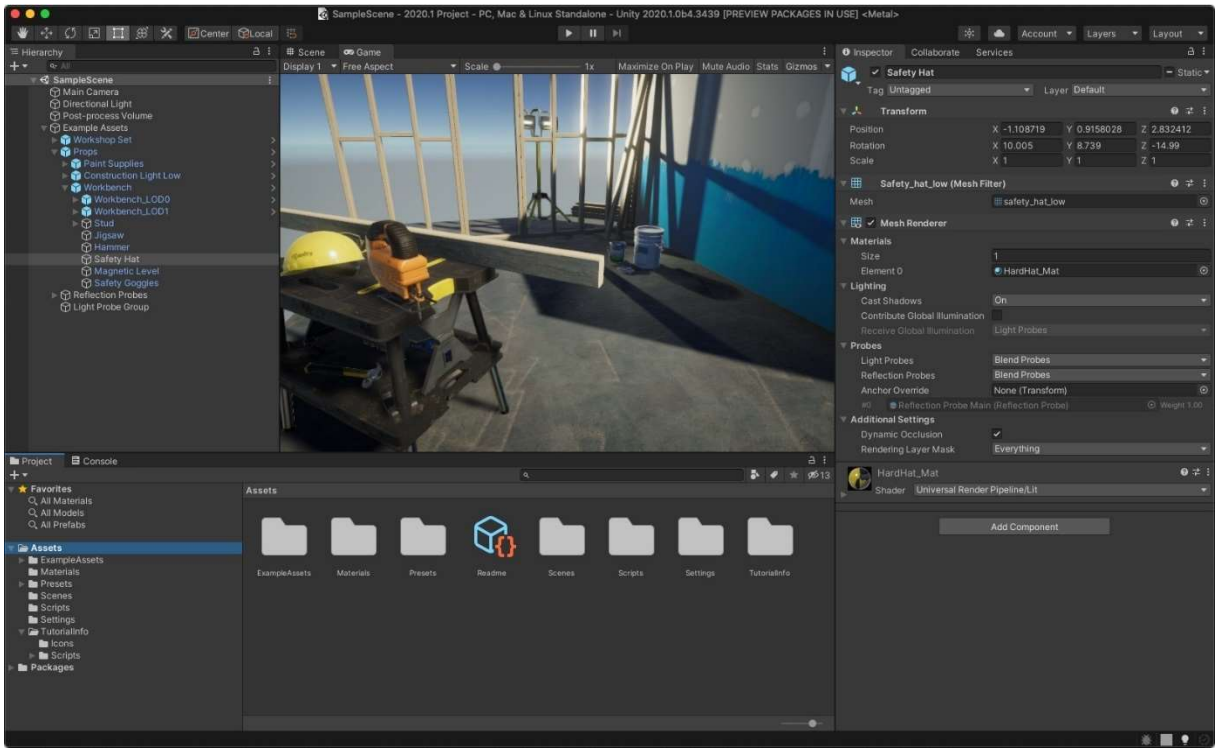
3.1 Unity

O *Unity* é uma *game engine*, desenvolvida pela *Unity Technologies*, que fornece funcionalidades para a criação de jogos e outros conteúdos interativos. É possível utilizar o *Unity* para desenvolver gráficos e recursos em cenas e ambientes 2D (Figura 17) ou 3D (Figura 18), adicionar física, editar e testar simultaneamente o ambiente e publicar em diversas plataformas, tais como computadores, celulares e consoles.

Figura 17: Imagem do editor *Unity* para 2D.

Fonte: (*UNITY*, 2022).

Figura 18: Imagem do editor *Unity* para 3D.



Fonte: (*UNITY,* 2022).

O *Unity* é bem documentado e possui uma ampla comunidade de desenvolvedores. É considerado uma das *game engines* mais populares entre os desenvolvedores de jogos e as ferramentas digitais, conforme é possível observar alguns exemplos, como na figura 19.

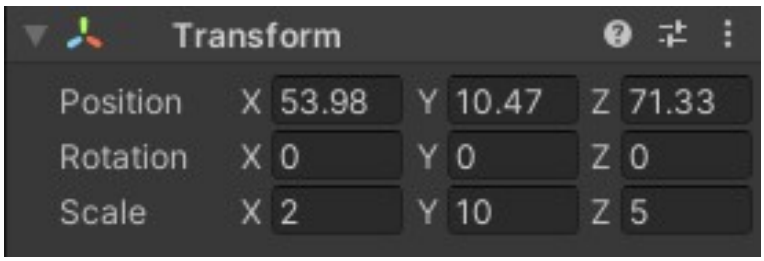
Figura 19: Desenvolvedoras que utilizam a *Unity*.

Fonte: (*UNITY,* 2022).

O *Unity* foi escolhido para este projeto devido a sua qualidade na representação da física, fácil importação de modelos 3D possibilitando integração com outros programas e ferramentas, como o *3DS MAX* e o *Visual Studio*.

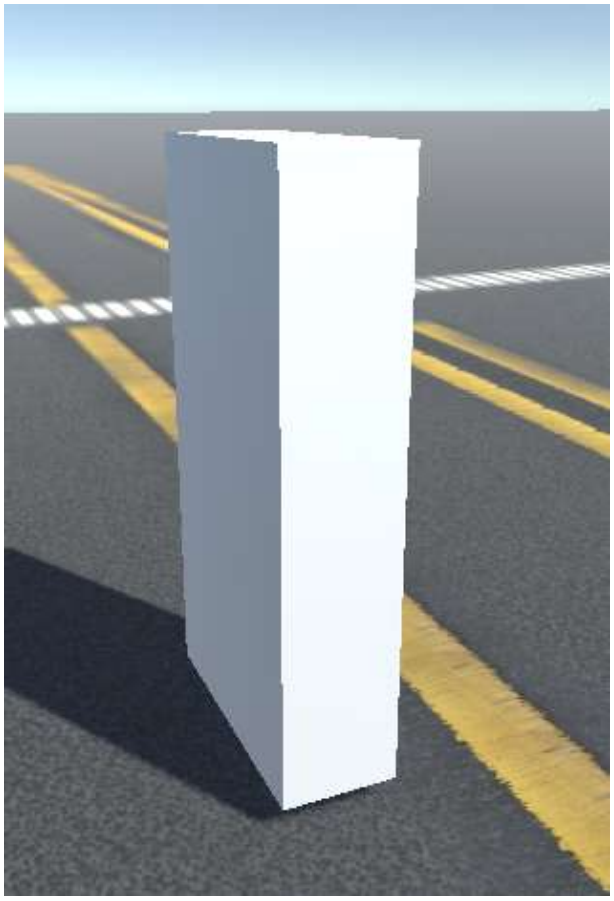
Todo objeto na *Unity* apresenta pelo menos um componente, o "*Transform*". Ele armazena e apresenta as características relacionadas a posição, rotação e escala do objeto. Quando se fala de posição, está relacionado as coordenadas, em x, y e z, daquele objeto dentro do espaço da *Unity*. Já a rotação diz respeito a rotação angular em x, y e z do objeto. E por último, a escala tem relação com o tamanho do objeto, por exemplo: na *Unity*, uma unidade representa um metro, ou seja, caso o objeto esteja numa escala de (1,1,1), o mesmo possui 1 metro de largura, 1 metro de altura e 1 metro de comprimento. Caso o objeto tenha como valor de escala (2,10,5), o mesmo terá 2 metros de largura, 10 de altura e 5 de comprimento. O exemplo do objeto está apresentado nas figuras 20 e 21.

Figura 20: Valores dos componentes Transform.



Fonte: Autor.

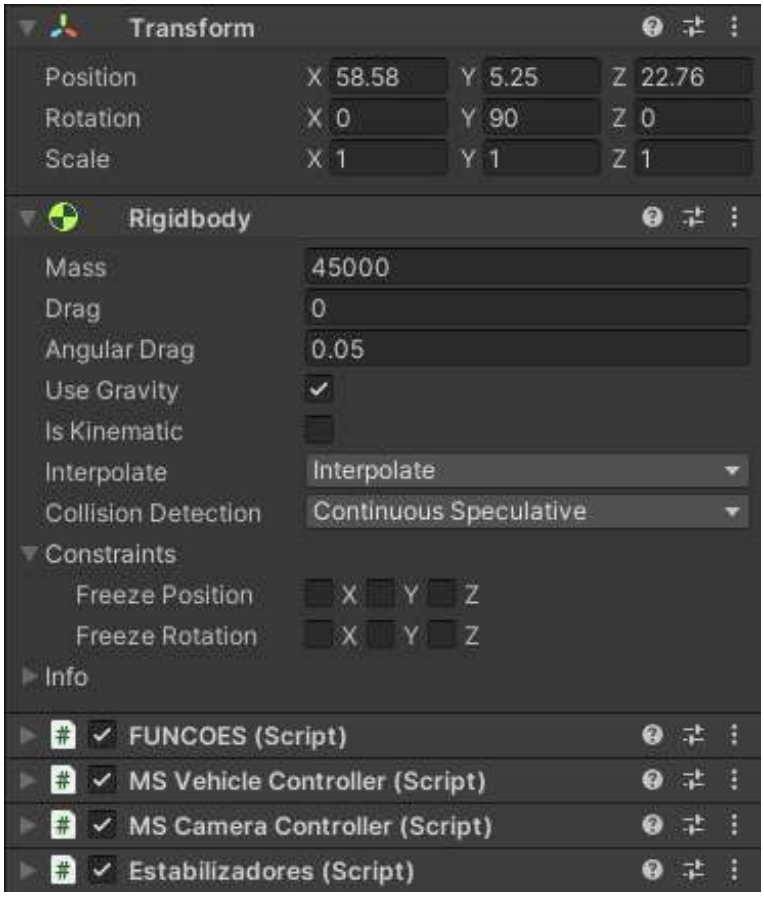
Figura 21: Objeto exemplo.



Fonte: Autor.

Outro componente muito importante no desenvolvimento desse trabalho é o *Rigidbody* que serve para adicionar massa ao objeto, optar por interagir ou não com a gravidade, entre outras configurações, e apresenta também os *scripts* do simulador que conforme já foi explicado anteriormente, os scripts são o "roteiro" de alguma função, neles que estão a programação. Foi escolhido colocar todos os scripts no veículo, já que o mesmo é o grupo principal da simulação, entretanto poderia ter sido em qualquer outro objeto, porém resultaria em uma complexidade maior de programação. Os componentes presentes no objeto Veículos estão apresentados na figura 22.

Figura 22: Componentes presentes no objeto Veículo.

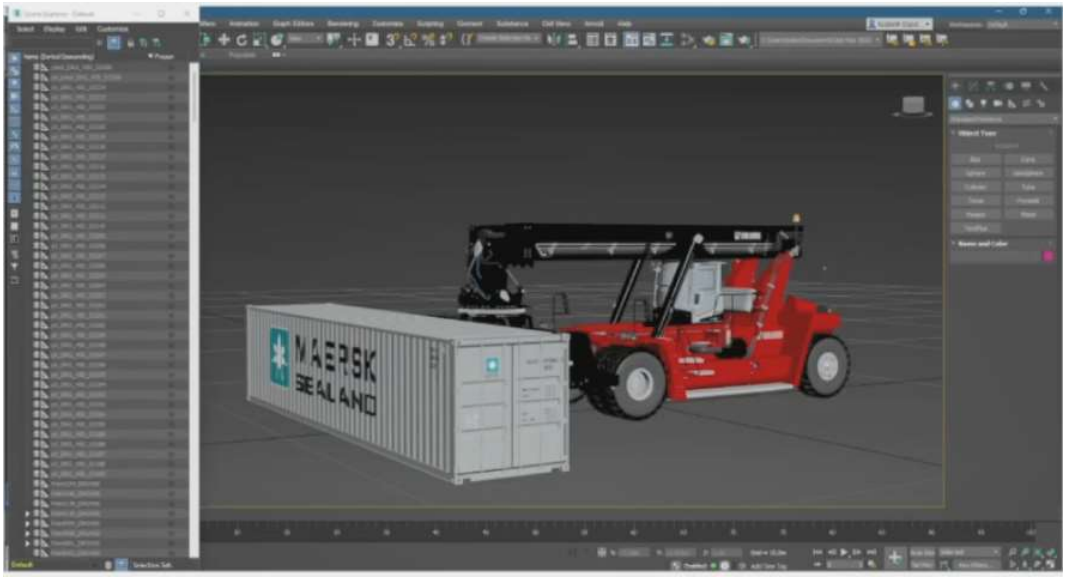


Fonte: Autor.

3.2 3DS MAX

O modelo 3D da RS (Figura 23) não foi modelado para este projeto, mas sim comprado através do projeto de pesquisa PVITJ2439/2021 do edital interno 38/2021/PROPPI do Instituto Federal de Santa Catarina – Câmpus Itajaí. É fácil entender a razão para isso, já que esse projeto se propõe a desenvolver um simulador, não sendo necessário modelá-lo desde o ponto inicial, o que seria muito difícil tendo em vista o tempo disponível para desenvolvimento. Além de ser contraproducente, já que mesmo apesar de muitos esforços para realizar a modelagem 3D, dificilmente apresentaria uma qualidade melhor do que o modelo que foi adquirido, já que foi produzido por profissionais da área.

Figura 23: Modelo 3D inicial da RS no 3DS MAX.

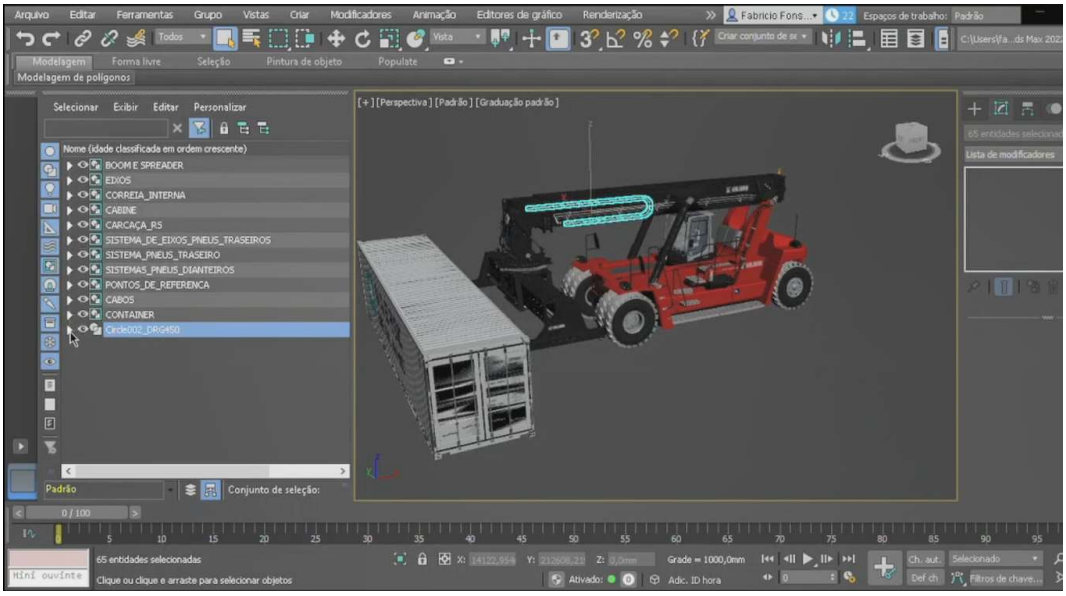


Fonte: Autor.

Primeiro, vale ressaltar que foi apenas utilizada a parte gráfica do modelo adquirido, sendo essa a razão para a utilização do *3DS MAX*. Como se trata de um programa focado em modelagem e renderização 3D, foi utilizado para agrupar melhor os objetos da máquina, já que o modelo original se encontrava com todos os objetos listados individualmente.

Para facilitar o trabalho de programação no *Unity*, os objetos foram reagrupados um a um, conforme é possível ser observado na figura 24, além de corrigir o eixo de direção de alguns e desmembrar outros.

Figura 24: Modelo 3D da RS no 3DS MAX com objetos separados por grupos.



Fonte: Autor.

3.3 Visual Studio

O *Visual Studio* é uma IDE, ou seja, um ambiente de desenvolvimento integrado. Toda a parte de codificação dos *scripts* (Figura 25) da máquina foi realizada no *Visual Studio*. *Scripts* são arquivos de textos que são interpretados pelo *Unity* como ordens, ou como o próprio nome diz, um roteiro a ser seguido a partir de alguma ação ou interação.

Todos os *scripts* implementados no simulador foram escritos em C#, utilizando o *Visual Studio* como ambiente de desenvolvimento e programação.

Figura 25: Exemplo de código no *Visual Studio*.

```
public class ConfiGeral : MonoBehaviour
{
    public GameObject menu;
    Menu menuInicial;
    static public bool estaPausado;
    static public float tempoRestante;

    // Start is called before the first frame update
    void Start()
    {
        menu = Instantiate(menu, menu.transform.position, menu.transform.rotation) as GameObject;
        menuInicial = menu.GetComponent<Menu>();
        Pause(false);

        tempoRestante = 30;
    }

    void Pause(bool statusPause)
    {
        estaPausado = statusPause;
        menuInicial.mostrarCronometro(estaPausado);
        if (estaPausado)
        {
            Time.timeScale = 0;
        }
        else
        {
            Time.timeScale = 1;
        }
    }
```

Fonte: Autor.

3.4 Cockpit

Para que o simulador ficasse o mais próximo de uma cabine de RS, foi adquirido através da verba disponibilizada para o projeto de pesquisa que o autor desse TCC também faz parte, um *cockpit*, conforme apresentando na figura 26.

Figura 26: *Cockpit Virtual Experience* - VE.3 Estação Completa - Banco vermelho.

Fonte: (*EXTREME SIMRACING, 2022.)*

O *cockpit* é feito em estrutura de chapas e tubos em carbono com espessura entre 1,5 mm e 3 mm, com pintura eletrostática a pó com tratamento antiferrugem, assoalho com acabamento em metal, base do volante com regulagem de altura, profundidade e inclinação, base dos pedais com ajuste de inclinação e distância, apoio de copo e câmbio ao lado da base do volante, banco com encosto dobrável e estrutura que suporta um monitor/televisão de até 55”, console de vídeo game, computador, teclado, mouse, câmbio, freio de mão e *joysticks*.

3.5 Volante e pedal

Para o controle de movimentação e direção da máquina, foi adquirido o kit da *Logitech*, G29 *Driving Force*, que é reconhecido pela *Unity* como input externo, sendo necessário apenas

realizar as configurações do *drive* e as calibrações do volante e pedais que compõem o produto, conforme apresentado na figura 27.

Figura 27: Volante e pedal.

Fonte: (*LOGITECH,* 2022).

3.6 Computador e monitor

A computador e o monitor serão instalados no *cockpit*, assim como todos os outros componentes. Ambos compõem o *hardware*, sendo o computador o responsável por rodar o software do simulador. O monitor será tanto o display de vídeo, apresentando para o usuário, visualmente, os resultados de suas interações através das ferramentas de interação, como o *display* de áudio, apresentando respostas auditivas ou alertas ao usuário.

Para facilitar a navegação durante os menus de navegação, será necessário utilizar um *mouse* e um teclado. O *mouse* terá uma função de interação com o simulador para realizar a movimentação das câmeras. Caso o usuário queira olhar para cima, direita, esquerda ou qualquer outra direção, ele deverá realizar esse movimento com o auxílio do *mouse*.

No caso do teclado, além da navegação nos menus, ele terá uma função um pouco mais importante, quando se trata da movimentação das partes móveis da máquina. Como o *joystick* é um componente caro, e que nem sempre estará à disposição do usuário para manipulação no simulador, todos os movimentos realizados pelo *joystick*, poderão também ser realizados pelo teclado, como se fossem comandos auxiliares ou de reserva.

4. DESENVOLVIMENTO

Neste capítulo será abordado como prosseguiu o desenvolvimento do simulador, apresentando a montagem do *cockpit* e traçando paralelos entre o *cockpit* montado, a cabine real da máquina e a cabine do modelo 3D.

Posteriormente será explanado com relação a integração do hardware e software do simulador e será aprofundado com relação a programação desenvolvida para funcionamento do simulador.

Por fim será apresentado os sistemas de fases e sistemas de tarefas presentes no simulador que foram desenvolvidos para possibilitar a utilização do simulador também como ferramenta auxiliar para possíveis treinamentos reais.

5.1 Montagem do cockpit

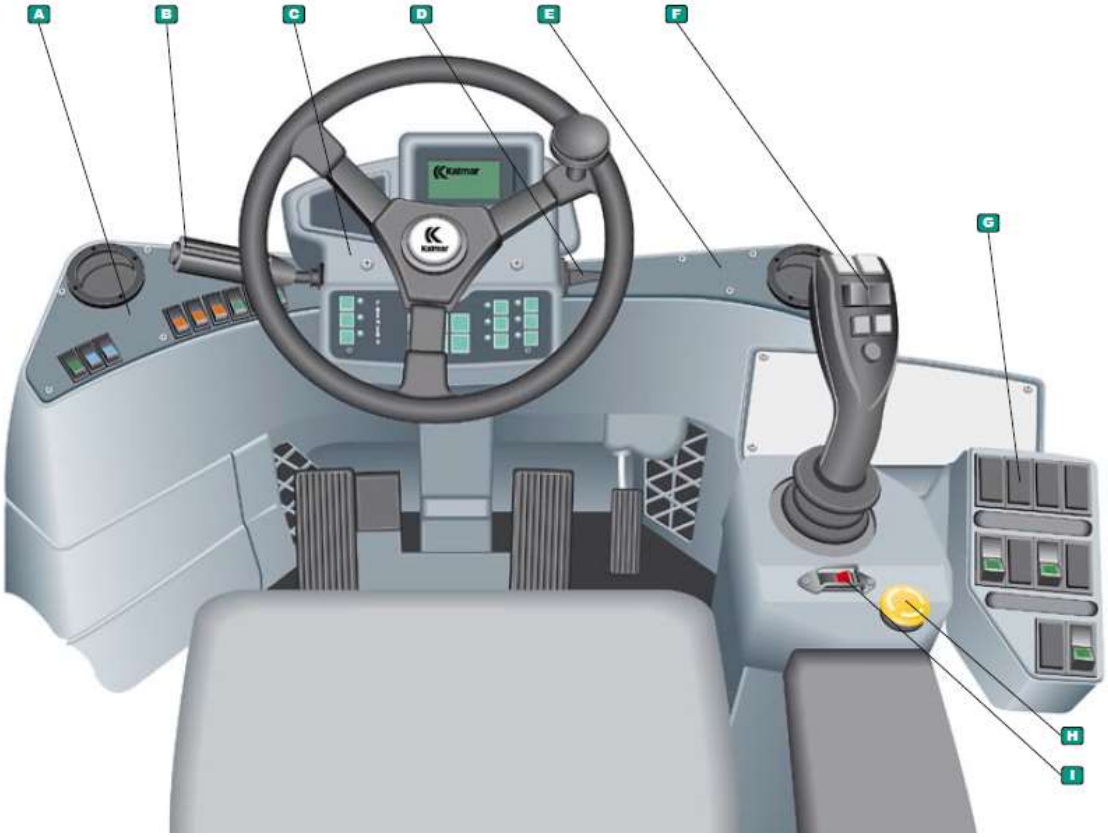
Para que o simulador seja o mais fiel possível com a realidade, a simulação 3D deve ser condizente com a realidade, no sentido de que o modelo 3D deve ser parecido com a empilhadeira real, como a física empregada no simulador deve ser programada e configurada respeitando a realidade.

O mesmo se aplica para o *cockpit*, na medida dos recursos disponíveis, o *layout* montado para o projeto deve ser o mais próximo possível do *layout* de uma cabine real da máquina.

5.1.1 Cabine de uma reach stacker

Dentro da cabine de uma RS é de onde saem todos os comandos que a máquina irá realizar. Com base na figura 28, a tabela abaixo indica os principais componentes que fazem parte de uma cabine de RS.

Figura 28: Informativo de cabine de RS.



Fonte: (*TECHNICAL INFORMATION KALMAR CONTAINER HANDLER*, 2020).

Tabela 1 – Componentes do painel de uma RS.

Índice	Componente
A	Painel de instrumentos esquerdo
B	Seletor de marcha e alavanca multifuncional
C	Painel do volante
D	Indicadores de direção
E	Preparado para terminais e anexo do painel
F	Controles hidráulicos
G	Painel para funções hidráulicas
H	Interruptor de emergência
I	Trava de mão

Nas figuras 29 e 30 estão apresentadas imagens reais das cabines, podendo-se ter uma maior noção de como é a vista do interior da cabine para fora, ou seja, a vista do operador, assim como o espaço que o mesmo tem dentro da cabine.

Figura 29: Painel de uma RS.

Fonte: (KALMAR GLOBAL, 2014)

Figura 30: Interior cabine do modelo 3D.

Fonte: Autor.

5.1.2 Processo de montagem do cockpit

Na figura 31 está apresentado o processo de montagem do *cockpit*, integrando os componentes do hardware, como o monitor, o computador, o teclado e o mouse montados no *cockpit*.

Figura 31: Processo de montagem do *cockpit*.

Fonte: Autor.

Na figura 32, a estrutura do *cockpit* está completamente montada. É possível observar onde ficaram os principais componentes do simulador, como o volante que ficará abaixo da tela e de frente para o usuário, podendo ser regulado; a posição dos pedais, na base logo abaixo do volante; o espaço destinado ao teclado, a esquerda do usuário e o local para o mouse, a direta do usuário.

Figura 32: Estrutura do *cockpit* montada

Fonte: Autor.

5.2 Organização do modelo 3D

O desenvolvimento do simulador se baseou na configuração, manipulação e, principalmente, na programação do modelo 3D da empilhadeira *reach stacker*. O primeiro passo necessário para iniciar a configuração e programação do simulador foi organizar a hierarquia do modelo 3D da máquina.

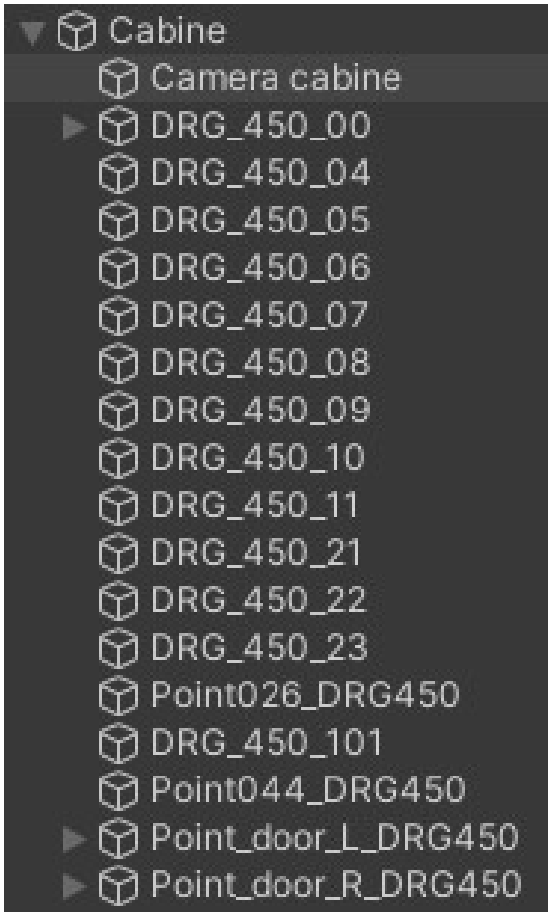
Primeiro foi necessário organizar todos os objetos da máquina em grupos, facilitando a visualização e entendimento. Estando todos os objetos pertencendo a um grupo da máquina, todos esses grupos foram inseridos no grupo principal, ou grupo pai, que foi chamado de Veiculo. O grupo Veiculo e seus "filhos" estão apresentados na figura 33.

Figura 33: Hierarquia do grupo Veículo.

Fonte: Autor.

No grupo correspondente à cabine estão todos os objetos que compõem a cabine com a adição de um objeto câmera, para simular a visão do operador no interior da cabine. A movimentação da câmera da cabine durante a simulação é realizada pelo *mouse*, possibilitando um maior campo de visão durante a simulação. Os objetos que fazem parte do grupo Cabine podem ser observados na figura 34.

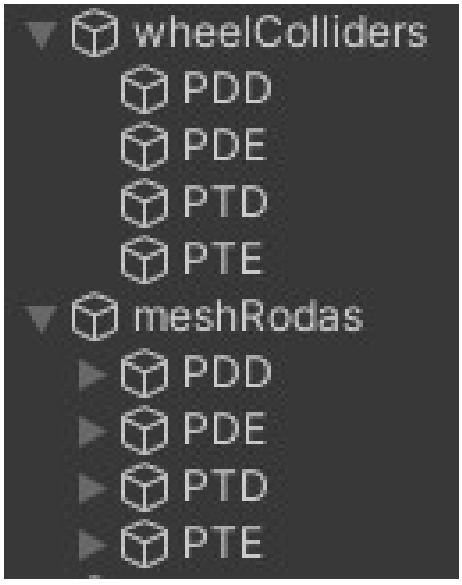
Figura 34: Hierarquia do grupo Cabine.



Fonte: Autor.

Os grupos *wheelColliders* e *meshRodas* estão primariamente relacionados com a movimentação da máquina e simulação do movimento das rodas.

Figura 35: Hierarquia dos grupos *wheelColliders* e *meshRodas*.



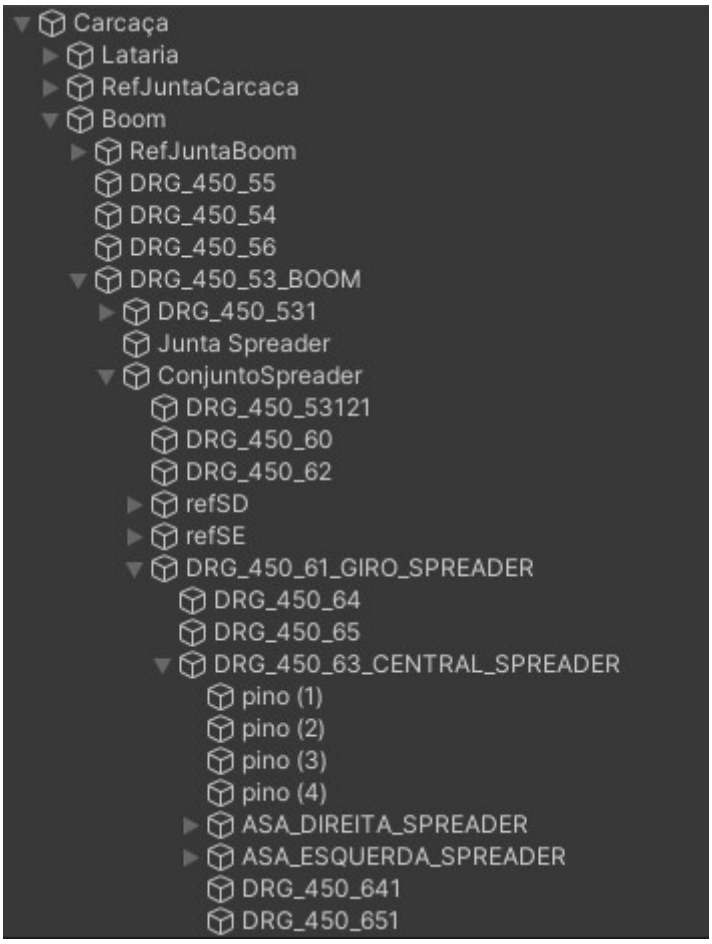
Fonte: Autor.

É a interação que ocorre entre os *colliders* e as *meshes*, através do *script* de movimentação da empilhadeira, que realiza a simulação do movimento da roda e dos pneus, tanto rotacionando como virando a roda para realizar as curvas. Como existe física implementada, no caso o atrito dos pneus com o solo durante toda essa simulação resulta na movimentação da empilhadeira como um todo.

É importante ressaltar que a programação dos *scripts* que realizam as funções de movimentação da câmera e movimentação da máquina foram desenvolvidos em um trabalho paralelo, não sendo desenvolvido pelo autor do presente trabalho. Entretanto, foi necessário realizar toda a configuração dos parâmetros pertencentes ao *script* para que a simulação ocorresse de maneira correta.

O último grupo pertencente ao grupo do Veiculo está nomeado como Carcaça. Este grupo apresenta diversos outros subgrupos e objetos, devido a isso é o maior grupo presente no modelo 3D, conforme pode ser observado na figura 36.

Figura 36: Grupos e subgrupos que pertencem ao grupo Carcaça.



Fonte: Autor.

5.3 Programação

No grupo Lataria estão presentes todos os objetos que compõem a parte externa da empilhadeira e que não realizam nenhum tipo de movimentação.

O objeto RefJuntaCarcaca é um ponto de referência que possui como filho na hierarquia o pistão que faz a elevação do *boom*, e está desta maneira pois na programação é utilizado comando *"transform.LookAt"*. Esse comando é utilizado para que o elemento interno do pistão esteja sempre "olhando" para o pistão, simulando desta maneiro o funcionamento do pistão. O trecho do *script* está apresentando na figura37.

Figura 37: *Script* do funcionamento dos pistões.

```
//////////   FUNCIONAMENTO DOS PISTÕES   //////////

pistaoBoom.transform.LookAt(pistaoCarcaca);
pistaoCarcaca.transform.LookAt(pistaoBoom);

pistaoSpreaderDireito.transform.LookAt(pistaoBoomDireito);
pistaoBoomDireito.transform.LookAt(pistaoSpreaderDireito);

pistaoSpreaderEsquerdo.transform.LookAt(pistaoBoomEsquerdo);
pistaoBoomEsquerdo.transform.LookAt(pistaoSpreaderEsquerdo);
```

Fonte: Autor.

Na imagem anterior também estão os comandos para outros dois pistões presentes na máquina, além do que conecta o *boom* à carcaça. Os outros dois conectam o eixo do *spreader* ao final da lança do *boom*.

O grupo Boom é composto por todos os objetos que realizam as funções da máquina, que já foram comentadas anteriormente em outros capítulos.

O objeto "JuntaSpreader" é na verdade um ponto de referência que foi implementado para simular a junção entre a lança e o conjunto do *spreader*. A configuração dos parâmetros dessa junta é realizada no grupo ConjuntoSpreader através do componente *Hinge Joint*.

A implementação desse componente e do objeto de referência se fez necessária para que durante a simulação fosse possível simular o balanço do contêiner durante sua movimentação, além de trazer mais fidelidade ao simulador, já que em operação com a máquina real o contêiner não fica estático conforme a máquina vai se locomovendo, o que aconteceria na simulação caso a junta não tivesse sido implementada.

Para implementar a programação das partes móveis da máquina que realizam funções específicas, dois comandos foram os mais utilizados, o "*transform.Rotate*" e o "*transform.localPosition*".

O comando *transform.Rotate* (Figura 38) pode ser utilizado para situações em que é necessário rotacionar algum objeto da máquina. Esse comando foi implementado para realizar as funções de levantar e abaixar o boom, tendo como eixo de rotação o início do boom, e foi utilizado também para implementar a função de rotacionar o *spreader*, tanto para sentido horário como anti-horário.

Figura 38: Função da máquina que foi utilizada o comando *transform.Rotate*.

```
//////////   GIRO SPREADER   //////////

if (Input.GetKey(KeyCode.O))
{
    giroSpreader.transform.Rotate(0, 0, rotacao * Time.deltaTime);

}
if (Input.GetKey(KeyCode.U))
{
    giroSpreader.transform.Rotate(0, 0, -rotacao * Time.deltaTime);
}
```

Fonte: Autor.

O comando *transform.localPosition* (Figura 39) pode ser utilizado para situações em que é necessário deslocar o objeto com base na sua posição. Esse comando foi utilizado para realizar as funções de esticar e recolher o boom, deslocar o *spreader* e para alterar a configuração do *spreader* para acoplar contêineres de 20’ para contêineres de 40’.

Figura 39: Função da máquina que foi utilizado o comando *transform.localPosition*.

```
//////////   ABRE E FECHA SPREADER   //////////

if (Input.GetKeyDown(KeyCode.E))
{
    spreaderEsquerdoNewPos = new Vector3(-4f, 0, -0.16f);
    spreaderDireitoNewPos = new Vector3(4f, 0, -0.16f);
}
if (Input.GetKeyDown(KeyCode.Q))
{
    spreaderEsquerdoNewPos = new Vector3(-1.2f, 0, -0.16f);
    spreaderDireitoNewPos = new Vector3(1.2f, 0, -0.16f);
}

spreaderEsquerdo.transform.localPosition = Vector3.MoveTowards(spreaderEsquerdo.transform.localPosition, spreaderEsquerdoNewPos, Time.deltaTime);
spreaderDireito.transform.localPosition = Vector3.MoveTowards(spreaderDireito.transform.localPosition, spreaderDireitoNewPos, Time.deltaTime);
```

Fonte: Autor.

Para a função acoplar o *spreader* foi necessário criar um comando dentro da programação com o intuito de tornar um objeto filho ao outro na hierarquia, comando chamado de “*SetParent*” (Figura 40). Foi realizado desta maneira pois no momento que a empilhadeira acopla o contêiner ao *spreader*, ele precisa acoplar a movimentação do *spreader* que consequentemente é influenciado por outras funções e o próprio deslocamento da RS.

Figura 40: Comando *SetParent()*.

```
1 referência
public void SetParent(GameObject novoParente)
{
    container.transform.parent = novoParente.transform;

    Debug.Log("Player's Parent: " + container.transform.parent.name);

    if (novoParente.transform.parent != null)
    {
        Debug.Log("Player's Grand parent: " + encaixe1.transform.parent.parent.name);
        Debug.Log("Player's Grand parent: " + encaixe2.transform.parent.parent.name);
        Debug.Log("Player's Grand parent: " + encaixe3.transform.parent.parent.name);
        Debug.Log("Player's Grand parent: " + encaixe4.transform.parent.parent.name);
    }
}
```

Fonte: Autor.

Entretanto, o comando *SetParent()* tem a funcionalidade no que se diz respeito a hierarquia, podendo ser acionado a qualquer momento, o que é incorreto. Sendo assim, foi necessário então implementar no *script* uma lógica de programação que fosse capaz de somente tornar filho quando o *spreader* estivesse na posição correta, com os pinos *locks* no interior das bolsas do contêiner, para que assim no momento do acoplamento o contêiner estivesse na posição correta.

Para isso foi utilizado o comando "*Vector3.Distance"* em conjunto com o "*transform.position*". Vale observar que desta vez foi utilizado a coordenada "*position*" ao invés do "*localPosition"*, já que o intuito dessa implementação é medir a distância de objetos que não estão na mesma hierarquia, logo são posições globais. Tendo então o valor da distância entre os pinos locks e as bolsas através do *Vector3.Distance*, foi necessário criar uma condição de acionamento do comando *SetParent*.

Utilizando o condicional "*if*", foi estipulada uma distância mínima de acionamento de 0,1 metros, ou seja, o acionamento do comando *SetParent* (Figura 41) só poderá ser realizado quando cada um dos quatro pinos *locks* estiver dentro dessa distância mínima até a bolsa do contêiner correspondente. Atendendo a essa condição, o contêiner será hierarquizado como filho do conjunto do *spreader*.

Figura 41: Função acoplar.

```
//////////   ACOPLAR   //////////

distancia1 = Vector3.Distance(encaixe1.transform.position, pino1.transform.position);
distancia2 = Vector3.Distance(encaixe2.transform.position, pino2.transform.position);
distancia3 = Vector3.Distance(encaixe3.transform.position, pino3.transform.position);
distancia4 = Vector3.Distance(encaixe4.transform.position, pino4.transform.position);

Debug.Log(distancia1);


if (Input.GetKeyDown(KeyCode.T) && distancia1 < 0.1f && distancia2 < 0.1f && distancia3 < 0.1f && distancia4 < 0.1f)
{
    SetParent(spreader);
}
```

Fonte: Autor.

Falando com relação a lógica de programação, foi comentado anteriormente que todas as funções da máquina relacionadas ao contêiner estão no mesmo script e que esse script está no objeto pai da hierarquia. Essa decisão foi tomada para que o código possa ser utilizado em outras RS, tendo a necessidade apenas de vincular cada objeto ao espaço da memória alocado para aquele objeto, isso ocorre devido a declaração dos objetos em código, que está apresentada na Figura 42.

Figura 42: Declaração dos objetos.

```
public GameObject spreaderEsquerdo;
public GameObject spreaderDireito;
public GameObject spreader;
public GameObject giroSpreader;

public GameObject container;

public GameObject pino1;
public GameObject pino2;
public GameObject pino3;
public GameObject pino4;

public GameObject encaixe1;
public GameObject encaixe2;
public GameObject encaixe3;
public GameObject encaixe4;

public GameObject boom;

public GameObject eixoBoom;

public Transform pistaoCarcaca;
public Transform pistaoBoom;
public Transform pistaoSpreaderDireito;
public Transform pistaoBoomDireito;
public Transform pistaoSpreaderEsquerdo;
public Transform pistaoBoomEsquerdo;
```

Fonte: Autor.

5.4 Sistema de fases e tarefas

As fases devem suceder umas às outras respeitando a dificuldade de execução de cada uma, de maneira que as fases com menor dificuldade, ou seja, funções mais básicas da empilhadeira, devem ser as primeiras. Sendo assim, conforme o usuário for avançando de fase, funções mais complexas são apresentadas para aprendizado e execução;

O sistema foi dividido em quatro fases, as primeiras duas fases estão relacionadas a movimentação da máquina e as duas últimas estão relacionadas a movimentação do contêiner. Todas as fases possuem um cromômero, para inserir uma certa dificuldade e desafio ao usuário.

A primeira fase apresenta as funções da máquina responsáveis pela movimentação da empilhadeira. Ou seja, apresenta as funções de ir para frente, para trás, para direita, para esquerda e freio. A fase possui um tempo inicial fixo e decrescente no cronômetro, que pode ser ajustado pelo usuário conforme a dificuldade. Conforme o usuário for acionando as funções corretamente, uma quantidade de tempo é adicionada ao tempo do cronômetro, uma forma de recompensar o usuário pelo acerto.

Na segunda fase é necessário deslocar a empilhadeira de um ponto ao outro, ainda sem o contêiner, sendo necessário colocar em prática as funções aprendidas na fase anterior. Essa fase também possui um cronômetro, e a fase deve ser finalizada até o fim do cronômetro para que o usuário possa avançar de fase

A terceira fase possui dinâmica parecida com a primeira fase, entretanto os comandos estão relacionados a movimentação do contêiner. Sendo assim, os comandos são os de erguer a lança, abaixar a lança, esticar a lança, recolher a lança, girar o spreader, abrir o spreader, fechar o spreader e acoplar o contêiner. Assim como a primeira fase, conforme o usuário for acionando as funções corretamente, uma quantidade de tempo é adicionada ao tempo do cronômetro

E por fim, na quarta fase é necessário deslocar o contêiner de um ponto até outro. Para ser mais específico, é necessário retirar o contêiner do chão e posicioná-lo na pilha de contêineres. Na última fase então, todos os comandos da empilhadeira apresentados nas fases anteriores serão necessários, seguindo então a ideia de ir dificultando conforme o usuário vai avançando de fase e ir utilizando o que foi aprendido nas fases anteriores.

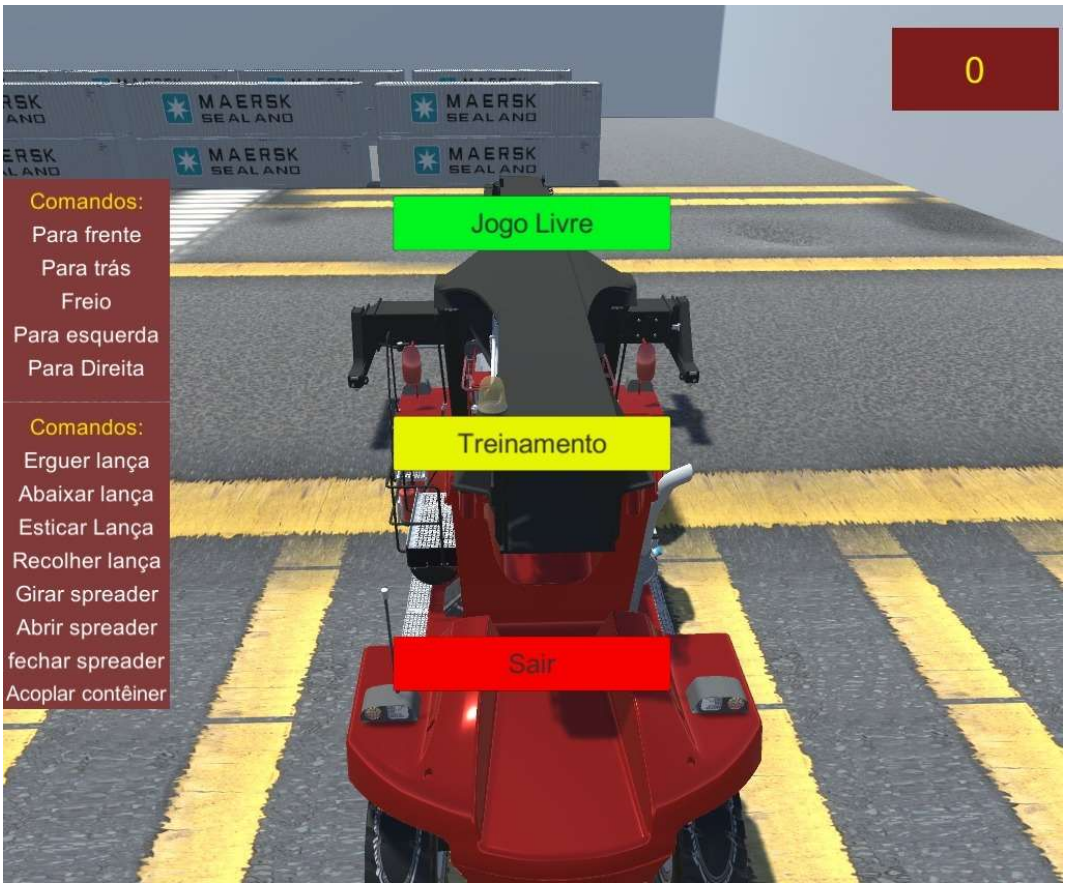
Na figura 43 é possível observar os comandos que compõem o sistema de tarefas, o sistema de fases e o cronômetro responsável por fazer o controle do limite de tempo de cada fase.

Figura 43: tarefas do sistema.

Fonte: Autor.

O desenvolvimento do menu, apresentado na figura 44, auxilia no treinamento, sendo possível reiniciar as tarefas, trocar para o cenário de movimentação livre e sair do simulador.

Figura 44: Menu do simulador.



Fonte: Autor.

O simulador foi utilizado por cerca de 8 pessoas que nunca haviam dirigido uma empilhadeira do tipo *reach stacker* anteriormente. Durante a utilização dessas pessoas foi possível observar como a máquina oferece dificuldade em seu primeiro contato, devido aos vários comandos presentes na mesma. Entretanto, após alguns minutos utilizando o simulador, com o auxílio do sistema de fases e tarefas, o aprendizado de cada comando da máquina foi bem intuitivo, auxiliando no entendimento com relação ao funcionamento da máquina.

O simulador também foi experimentado por uma pessoa que já haviam pilotado uma RS anteriormente. A maior dificuldade apresentada foi com relação a quais comandos do teclado realizam as funções da empilhadeira no simulador. Entretanto, após memorizar cada tecla de comando, o simulador demonstrou ser bem fiel a realidade, possibilitando uma operação fluída tanto para quem nunca havia dirigido uma *reach stacker*, tanto para quem já teve contato com a máquina real.

5. RESULTADOS E CONCLUSÕES

Após muitos ajustes, o simulador funcionou de maneira satisfatória, tanto no que diz respeito à fidelidade dos componentes da máquina, como aos comandos que a mesma deve executar em sincronia com os componentes do *cockpit*.

Os principais objetivos propostos pelo trabalho foram atendidos, já que o simulador está funcional e respeita o sistema de tarefas e fases configuradas, desafiando o usuário ao aprendizado e dando como alternativa também a movimentação livre com a máquina.

O menu foi de grande valia tanto para as pessoas que nunca haviam dirigido uma RS, assim como ao que já havia, possibilitando uma melhor navegação dentro do simulador, dando a possibilidade de reiniciar o mesmo caso seja necessário e trazendo a possibilidade de alterar o modo de utilização do simulador.

Em seu primeiro uso, o simulador apresenta bastante dificuldade ao usuário, devido à complexidade da máquina, porém foi possível observar que as tarefas ajudam no entendimento da máquina e possibilitam um melhor aprendizado.

Possíveis melhorias que poderiam ser implementadas são o desenvolvimento de uma quantidade maior de fases, voltadas para situações específicas do dia a dia de um operador e a utilização de inteligência artificial para uma maior dificuldade durante a simulação, contando com a presença de carros, pessoas e outras máquina interagindo na cena durante a simulação, sendo necessário a utilização de outros modelos 3D em conjunto no cenário, cujos exemplos seriam: caminhões, porteineres, navios e carros.

Futuramente, com o aperfeiçoamento do trabalho, poderiam ser implementadas melhorias no modelo 3D, como retrovisores, as três Marias, movimentação da cabine, entre outros fatores.

E como sugestão para prosseguimento do trabalho, seria necessário testar o simulador por mais vezes com pessoas que já possuem experiência com a empilhadeira, de preferência com os operadores reais da máquina. Assim seria possível extrair um feedback real do simulador, já que seria utilizada por pessoas que utilizam a empilhadeira real diariamente.

As principais dificuldades apresentadas no desenvolvimento desse simulador estão relacionadas com o aprendizado da linguagem de programação em conjunto com o software de simulação utilizado, já que grande parte das funções utilizadas para a programação dos comandos são próprios da *Unity*. Além da dificuldade com relação ao aprendizado do software de simulação utilizado, a manipulação do modelo 3d para que o mesmo estivesse pronto para ser programado também foi uma tarefa árdua.

De maneira geral os resultados apresentados pelo simulador foram satisfatórios, principalmente aos que nunca haviam dirigido uma empilhadeira de grande porte anteriormente. Pode-se concluir que o simulador ainda tem muito a evoluir, apesar de estar funcional. Com mais tempo de desenvolvimento pode-se tornar cada vez mais complexo e realista.

REFERÊNCIAS

[1] INCATEP - INSTITUTO DE CAPACITAÇÃO PROFISSIONAL. **Apostila Empilhadeira de Grande Porte**. Disponível em: <https://pt.slideshare.net/JGC026/apostila-empilhadeira-de-grande-porte?from_action=save>. Acesso em: 29 jul. 2022.

[2] INCATEP - INSTITUTO DE CAPACITAÇÃO PROFISSIONAL. **Reach Stacker**. Disponível em: <https://pt.slideshare.net/JGC026/reach-stacker-137551523>. Acesso em: 29 jul. 2022.

[3] INCATEP - INSTITUTO DE CAPACITAÇÃO PROFISSIONAL. **USO DE SIMULADORES INCATEP**. Disponível em: <https://pt.slideshare.net/JGC026/uso-de-simuladores-incatep>. Acesso em: 29 jul. 2022.

[4] RIBEIRO, M. REACH STACKERS KALMAR. www.academia.edu, [s.d.].

[5] INCATEP - INSTITUTO DE CAPACITAÇÃO PROFISSIONAL. **EMPILHADEIRA DE LANÇA / REACH STACKER**. Disponível em: <https://pt.slideshare.net/JGC026/empilhadeira-de-lana-reach-stacker>. Acesso em: 29 jul. 2022.

[6] **Kalmar - Essential Range Reachstackers**. Disponível em: <https://www.kalmarglobal.com/efit/reachstackers/>. Acesso em: 29 jul. 2022.

[7] ITAJAÍ, P. DE. **Complexo Portuário de Itajaí bate recorde e movimenta 1,6 milhão de contêineres em 2021 | Município de Itajaí**. Disponível em: <https://itajai.sc.gov.br/noticia/28024/complexo-portuario-de-itajai-bate-recorde-e-movimenta-16-milhao-de-conteineres-em-2021#.Yt809UHMKiO>. Acesso em: 29 jul. 2022.

[8] **Operação de Reach Stacker**. Disponível em: <http://www.msoequipamentos.com.br/treinamentos/operacao-de-reach-stacker/>. Acesso em: 29 jul. 2022.

[9] **O que são as Game Engines ou Motores de Jogos?** Disponível em: <http://www.universoprofissional.com.br/blog/game-design/o-que-sao-game-engines/>. Acesso em: 29 jul. 2022.

[10] TECHNOLOGIES, U. **Tem curiosidade em saber o que é Unity? Descubra quem somos, o que fizemos e para onde vamos | Unity**. Disponível em: <https://unity.com/pt/our-company>. Acesso em: 29 jul. 2022.

[11] **Introdução ao desenvolvimento de games com Unity3D**. Disponível em: <https://www.devmedia.com.br/unity-3d-introducao-ao-desenvolvimento-de-games/30653>. Acesso em: 29 jul. 2022.

[12] ONLINE, R. **Visual Studio Code: confira as principais funções da ferramenta - Remessa Online**. Disponível em: <https://www.remessaonline.com.br/blog/visual-studio-code-confira-as-principais-funcoes-da-ferramenta/>. Acesso em: 29 jul. 2022.

[13] **Visual Studio IDE 2022 – Ferramenta de Programação para Desenvolvedores de Software**. Disponível em: <https://visualstudio.microsoft.com/pt-br/vs/>.

[14] ECOMMERCE, S. I. S. **Cockpit VE3 Estação Completa para volantes simuladores vermelho**. Disponível em: <https://loja.cockpitextremeracing.com.br/produto/cockpit-virtual-experience-3-estacao-completa-vermelho/33309>. Acesso em: 29 jul. 2022.

[15] **Volante Logitech Driving Force G29 Para PS4 / PS3 / PC**. Disponível em: <https://www.logitechstore.com.br/volante-para-jogos-logitech-g29#>. Acesso em: 29 jul. 2022.

[16] **Unity 2020.1b**. Disponível em: <https://unity3d.com/pt/beta/2020.1b>. Acesso em: 29 jul. 2022.

[17] **Get a list of scripts attached to an object?** Disponível em: <https://forum.unity.com/threads/get-a-list-of-scripts-attached-to-an-object.453632/>. Acesso em: 29 jul. 2022.

[18] **Complexo Portuário de Itajaí e Navegantes excede a marca de 1,6 Milhão de contêineres (TEU'S) movimentados em 2021.** Disponível em: <https://www.portoitajai.com.br/noticia/1591/complexo-portuario-de-itajai-e-navegantes-excede-a-marca-de-1-6-milhao-de-conteineres-teu-s-movimentados-em-2021->. Acesso em: 29 jul. 2022.

[19] TORI, R.; KIRNER, C.; SISCOUTTO, R. (EDS.). **Fundamentos e tecnologia de realidade virtual e aumentada** Editora SBC – Sociedade Brasileira de Computação, 2006.

[20] NETO, M. et al. (EDS.). ANÁLISE DA PRODUTIVIDADE E EFICIÊNCIA EM TERMINAIS DE CONTÊINERES POR MEIO DE SIMULAÇÃO. **CONECTA**, 2019.

[21] TEICHRIEB, V. et al. **Realidade virtual e aumentada na prática.** Recife: Gráfica E Copiadora Nacional, 2008.

[22] TORI, Romero; HOUNSELL, Marcelo da Silva (org.). **Introdução a Realidade Virtual e Aumentada**. Porto Alegre: Editora SBC, 2018.

[23] KIRNER, C.; SISCOUTTO, R. (EDS.). **Realidade Virtual e Aumentada Conceitos, Projeto e Aplicações**. Editora SBC – Sociedade Brasileira de Computação, 2007.