UNIVERSIDADE FEDERAL DO AMAZONAS
FACULDADE DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

YURI MOTTA LOPES RODRIGUES SILVA

AMBIENTE DE TREINAMENTO POR TELEOPERAÇÃO PARA NOVOS USUÁRIOS DE CADEIRAS DE RODAS MOTORIZADAS BASEADO EM MÚLTIPLOS MÉTODOS DE CONDUÇÃO

MANAUS
2018

UNIVERSIDADE FEDERAL DO AMAZONAS
FACULDADE DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

YURI MOTTA LOPES RODRIGUES SILVA

AMBIENTE DE TREINAMENTO POR TELEOPERAÇÃO PARA NOVOS USUÁRIOS DE CADEIRAS DE RODAS MOTORIZADAS BASEADO EM MÚLTIPLOS MÉTODOS DE CONDUÇÃO

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal do Amazonas, como requisito parcial para obtenção do título de Mestre em Engenharia Elétrica, área de concentração Controle e Automação de Sistemas.

Orientador: Prof. Dr. –Ing. Vicente Ferreira de Lucena Junior

MANAUS
2018

Ficha Catalográfica

Ficha catalográfica elaborada automaticamente de acordo com os dados fornecidos pelo(a) autor(a).

Silva, Yuri Motta Lopes Rodrigues
S586a Ambiente de treinamento por teleoperação para novos usuários de cadeiras de rodas motorizadas baseado em múltiplos métodos de condução / Yuri Motta Lopes Rodrigues Silva. 2018
104 f.: il. color; 31 cm.

Orientador: Vicente Ferreira de Lucena Junior
Dissertação (Mestrado em Engenharia Elétrica) - Universidade Federal do Amazonas.

1. Ambiente de Treinamento. 2. Cadeira de Rodas Motorizada. 3. Eye-Tracker. 4. Teleoperação. I. Lucena Junior, Vicente Ferreira de II. Universidade Federal do Amazonas III. Título

YURI MOTTA LOPES RODRIGUES SILVA

AMBIENTE DE TREINAMENTO POR TELEOPERAÇÃO PARA NOVOS USUÁRIOS DE CADEIRAS DE RODAS MOTORIZADA BASEADO EM MÚLTIPLOS MÉTODOS DE CONDUÇÃO.

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal do Amazonas, como requisito parcial para obtenção do título de Mestre em Engenharia Elétrica na área de concentração Controle e Automação de Sistemas.

Aprovado em 10 de agosto de 2018.

BANCA EXAMINADORA

Prof. Dr. Vicente Ferreira de Lucena Junior, Presidente
Universidade Federal do Amazonas

Prof. Dr. Teodiano Freire Bastos Filho, Membro
Universidade Federal do Espírito Santo

Prof. Dr. Eduardo Neves, Membro
Universidade Federal de Uberlândia

One, remember to look up at the stars and not down at your feet. Two, never give up work. Work gives you meaning and purpose and life is empty without it. Three, if you are lucky enough to find love, remember it is there and don't throw it away.

Stephen Hawking

Agradecimentos

Dedico este trabalho e todo o meu esforço a Deus e a minha família.

Agradeço a minha esposa Yasmin Lima Serrão Motta, minha mãe Leena Motta da Rocha Lopes, meu pai Antonio Carlos Rodrigues Silva e a minha irmã Yanna Motta L. R. S. Oliveira.

Agradeço ao Professor Vicente pelo trabalho de orientação técnica e por todo apoio fornecido durante a jornada do programa.

Agradeço ao Professor Eduardo Naves pelo suporte no desenvolvimento do projeto e experimentação.

Agradeço ao meu amigo César Quiroz que me apoiou durante toda a jornada do curso.

Agradeço aos meus colegas de laboratório Walter Simões, Mario Alves Jr. e aos demais que ajudaram direta e indiretamente no desenvolvimento do trabalho.

Agradeço aos professores do PPGEE que ensinaram categoricamente o conteúdo técnico científico do programa.

Agradeço a CAPES, UFAM, CNPq, FAPEAM e Samsung pelo suporte que viabilizou a pesquisa científica e as publicações.

Resumo

Atualmente, diversos ambientes de treinamento existentes ajudam novos usuários de cadeira de rodas motorizada (CRM) a aprender a comandar, se familiarizar e aprimorar suas habilidades. Vários autores estão desenvolvendo esses ambientes, e a maioria deles está usando CRM virtualmente simulada. Apesar das semelhanças entre a CRM virtual e a real, observou-se que é mais fácil comandar o dispositivo real. Isso ocorre porque nesses ambientes virtuais, a representação do comportamento físico da CRM ainda é um problema. Outro aspecto observado, foi a respeito dos métodos de condução, onde a maioria dos trabalhos utiliza apenas o *joystick*. Porém, esse método não oferece a oportunidade a usuários com deficiência severa de aprender a comandar a partir de novas tecnologias, como por exemplo, o rastreamento ocular. Para superar essas dificuldades, este trabalho propõe, implementa e valida uma abordagem mais realista, a qual é baseada em treinamento por teleoperação e por múltiplos métodos de condução. Foi implementada uma arquitetura que permite ao usuário enviar comandos remotamente para comandar uma CRM real a longas distâncias. Os métodos de condução implementados foram por *joystick*, *eye-tracker* e por meio de uma interface humano-máquina genérica. Para a avaliação do sistema, foram criados cenários considerando diferentes configurações. Os resultados experimentais sugerem que novos usuários podem praticar com segurança utilizando uma CRM real através da Internet, mesmo em uma situação com *delay* de 130,2 ms (média). O sistema proposto mostrou potencial em atender novos usuários de CRM com diferentes tipos de deficiência, bem como de ser uma abordagem de baixo custo com possibilidade de ser aplicada em países em desenvolvimento.

Palavras-chave: Ambiente de Treinamento; Cadeira de Rodas Motorizada; *Eye-Tracker;* Teleoperação.

Abstract

Currently, diverse existing training environments help new users of electric powered wheelchairs (EPW) learn how to drive, acquaint and improve their abilities with these assistive devices. Several authors are developing such environments, and most of them use virtually simulated wheelchairs. Despite the similarities between virtual and real wheelchairs, it is easier to drive the real device because representation of the wheelchair physical behavior is still a problem for virtual simulated environments. Concerning the driving methods, most of them are based on a joystick, which does not give the opportunity for users to test, practice and acquaint themselves with new technologies, such as driving through eye movements. This work implements and tests a more realistic approach for a training environment dedicated to new users of EPW. The proposed system is based on a real EPW controlled by teleoperation, and it is flexible enough to attend to multiple driving methods. An architecture that allows a user to send command messages to control a real EPW through the Internet was implemented to validate the system. The implemented driving methods were conventional joystick, eye-tracker and a generic human-machine interface. For the system's evaluation, scenarios were created considering the implemented driving methods, and also scenarios considering a long distance teleoperation. The experimental results suggest that new users can practice safely using a real EPW through the Internet, even in a situation with a communication delay of 130.2 ms (average). Furthermore, the proposed system showed potential for attending new EPW users with different types of disabilities and to be a low-cost approach that could be applied in developing countries.

Keywords: *Training Environment; Electric-Powered-Wheelchair; Eye-Tracker; Teleoperation.*

Lista de Siglas

ACM	*Association for Computing Machinery*
ADC	*Analog-to-Digital Converter*
API	*Application Program Interface*
ARM	*Advanced RISC Machines*
BPS	*Bits per Second*
BSP	*Board Support Package*
CAN	*Controller Area Network*
CRM	Cadeira de Rodas Motorizada
CT	Centro de Treinamento
DAC	*Digital-to-Analog Converter*
EPW	*Electric Powered Wheelchair*
FFMPEG	*Fast Forward MPEG*
FPV	*First-Person View*
GPIO	*General Purpose Input/Output*
GUI	*Graphical User Interface*
HMD	*Head Mounted Display*
HTTP	*Hypertext Transfer Protocol*
I2C	*Inter-Integrated Circuit*
IBGE	Instituto Brasileiro de Geografia e Estatística
IEEE	*Institute of Electrical and Electronics Engineers*
IHM	Interface Humano-Máquina
IHMG	Interface Humano-Máquina Genérica
IP	*Internet Protocol*
IPCAM	*IP Camera*
IPv4	*Internet Protocol Version 4*
IPv6	*Internet Protocol version 6*
Kbps	*Kilobits per Second*

MIPS	*Microprocessor without Interlocked Pipeline Stages*
MJPEG	*Motion JPEG*
MPEG	*Moving Picture Experts Group*
MPEGTS	*mpeg Transport Stream*
POO	Programação Orientada a Objetos
PowerPC	*Power Performance Computing*
QR Code	*Quick Response Code*
RAM	*Static Random-Access Memory*
RTP	*Real-Time Transport Protocol*
RTT	*Round-trip Time*
SO	Sistema Operacional
SPARC	*Scalable Processor Architecture*
SPI	*Serial Peripheral Interface*
SSH	*Secure Shell*
ST	Sala de Testes
TCP	*Transmission Control Protocol*
TPV	*Third-Person View*
UART	*Universal Asynchronous Receiver/Transmitter*
UDP	*User Datagram Protocol*
UFU	Universidade Federal de Uberlândia

Lista de Figuras

Lista de Tabelas

Sumário

1. Introdução

A cadeira de rodas motorizada (CRM) é um dispositivo de tecnologia assistiva que ajuda pessoas com deficiência a se locomoverem de forma facilitada em relação a cadeira de rodas manual. Os usuários podem comandá-la através de um *joystick*, escolhendo a direção e a velocidade desejada. O dispositivo conta com um sistema de baterias, motores elétricos e um circuito eletrônico para acionamento.

Considerando os tipos de usuários de CRM, esses podem ser diversos, por exemplo, vítimas de acidentes que passaram a depender do dispositivo, pessoas com deficiência congênita nos membros inferiores, bem como, pessoas com deficiências severas, como esclerose lateral amiotrófica, esclerose múltipla, paralisia cerebral e miopatia. Utilizando múltiplos métodos de condução.

Em relação às pessoas com deficiência no Brasil, através do censo do IBGE, foram identificadas 3,2 milhões com deficiência motora (Ibge, 2010). A CRM tem um papel fundamental na vida desses usuários, pois permite que a locomoção seja feita de forma facilitada, auxiliando na melhoria da qualidade de vida e fazendo parte do dia-a-dia dessas. O dispositivo permite aos usuários acesso a shoppings, supermercados, escolas, universidades e hospitais, consequentemente, promove a inclusão social, entretenimento, educação e saúde.

Devido a que muitos usuários não têm familiaridade, nem habilidade de comandar a CRM, surge a necessidade da prática antes de utilizar o próprio dispositivo. A prática permite que os usuários evoluam suas habilidades e promove a sensação de segurança. A forma tradicional de aprender é na qual um usuário recebe auxílio de um instrutor enquanto faz a condução. No entanto, um erro pode causar acidentes como quedas e colisões, trazendo risco à integridade física do usuário e às pessoas envolvidas no treinamento. Além disso, um acidente pode frustrar o usuário e levá-lo até mesmo à desistência do aprendizado.

Vários autores desenvolveram técnicas para facilitar o aprendizado da condução, por exemplo, Morere e seus co-autores criaram um treinamento baseado em virtualização, onde pessoas com deficiência praticavam a partir de simulações computacionais (Morere *et al.*, 2015). De forma similar, outros autores como Rossol e seus co-autores, e Rodriguez, também desenvolveram CRM virtuais (Rossol *et al.*, 2011; Rodriguez, 2015). Não obstante, a evolução desses ambientes baseados em simulação ainda é uma tarefa muito difícil em relação à

representação física do comportamento da cadeira de rodas (Archambault *et al.*, 2011; John *et al.*, 2017).

Atualmente esses ambientes de treinamento têm como método de condução apenas o *joystick.* Sendo esse o método convencional para comandar uma CRM, que abrange usuários com deficiência nos membros inferiores. Contudo, existem métodos alternativos para comandar a CRM, os quais podem ser através de novas tecnologias, como por exemplo, o rastreamento ocular. Em vista disso, usuários com deficiência severa também podem comandar a CRM. Por exemplo, aqueles que não têm movimento das mãos podem comandar a CRM a partir de movimentos oculares.

Dadas as características de contextualização do tema, na subseção seguinte serão apresentados os detalhes sobre a motivação da proposta, onde serão apresentados os desafios encontrados para implementação, bem como o escopo do trabalho. Também será introduzida a ideia generalizada da arquitetura da proposta, de forma a criar uma base prévia à apresentação dos objetivos científicos.

1.1 Motivação

A forma atual dos ambientes de treinamento para novos usuários de CRM pode ser evoluída, por exemplo, através da criação de um método para treinamento que não dependa de virtualização e simulações computacionais. Consequentemente, solucionando o problema de representação do comportamento físico da CRM mencionado por trabalhos relacionados.

Outra característica que pode ser melhorada, é em relação à cobertura de treinamento para diferentes tipos de usuários. Citando caso análogo, atualmente os ambientes englobam apenas as pessoas com deficiência nos membros inferiores. Assim, pode-se desenvolver um modelo que também inclua pessoas com deficiências severas, as quais têm dificuldades ou não conseguem comandar a CRM na forma convencional. Esse modelo pode ser baseado em múltiplos métodos, desde o método convencional (*joystick*), até métodos alternativos como o rastreamento ocular.

No contexto apresentado, esta pesquisa visa a criação de um ambiente de treinamento que possibilite ao usuário praticar sem riscos de acidente, e ter uma experiência realista, sem a utilização de virtualização e simulações computacionais. No ambiente proposto, pretende-se comandar uma CRM real, proporcionando então uma experiência realista ao usuário, dando ao usuário a oportunidade de aprender de forma segura e eficaz, antes de adquirir o próprio dispositivo.

Para que isso seja possível, propõe-se que o ambiente seja dividido em duas partes. A primeira é o Centro de Treinamento (CT), sendo esse o local onde o usuário ficará localizado. O CT poderá ser dentro de hospitais, clínicas de fisioterapia e clínicas de reabilitação física. No CT ficarão todos os dispositivos e equipamentos necessários para a operação, como dispositivos para comandar a CRM, dispositivos para obter o feedback em tempo real e dispositivos para comunicação a longas distâncias.

Um dos conceitos para a criação do CT é permitir que pessoas com diferentes tipos de deficiência possam aprender a comandar a CRM. Por exemplo, caso o usuário seja uma pessoa que sofreu recentemente um acidente e tenha perdido o movimento das pernas, esta poderá aprender a comandar a partir de um *joystick* (forma convencional). Ademais, caso o usuário não tenha o movimento das mãos, esse poderá praticar a partir de métodos alternativos. Pretende-se também a criação de um método genérico, a qual servirá como modelo para facilitar a inclusão de outros métodos.

Previamente à apresentação da segunda parte desta proposta, será introduzido o conceito de teleoperação. Um sistema de teleoperação permite a um usuário imergir em um ambiente distante ou inacessível para realizar tarefas. Esse usuário opera remotamente sem a necessidade de estar localizado onde a operação está ocorrendo.

Isto posto, a segunda parte do ambiente consiste na Sala de Teleoperação (ST). Nesse local ficará localizada a CRM e todos os dispositivos necessários para receber os comandos remotos do usuário. Espera-se que na ST seja feita a comunicação em tempo real com o CT. Almeja-se utilizar uma CRM real para o treinamento, tornando-o mais realista e dando ao usuário um conceito fidedigno de como é comandar uma CRM. Para que isso seja possível, será necessário desenvolver todo o tratamento dos comandos do usuário, de modo a convertê-los em sinais elétricos capazes de movimentar os motores da CRM.

Em se tratando da integração de todas as funcionalidades presentes na ST, como recebimento de mensagens remotas, envio de sinais para movimentação da CRM, comunicação em tempo real, essas são tarefas de alta complexidade, pois ocorrem simultaneamente, e não poderão haver conflitos que bloqueiem o sistema, nem em quesitos de *software* e nem de *hardware*.

A comunicação entre o CT e a ST será um fator fundamental para garantir o bom funcionamento do ambiente de treinamento, pois essa terá como requisito um baixo tempo de resposta entre o envio do comando remoto do usuário e o movimento físico da CRM, garantindo a comunicação em tempo real. Esse quesito será indispensável para que o usuário tenha uma experiência realista de treinamento, pois grandes atrasos entre o envio do comando

do usuário e o movimento real da CRM poderão acarretar em um treinamento ineficaz. Logo, este é um dos principais desafios encontrados nesta pesquisa, o qual consiste na definição de uma arquitetura de telecomunicação capaz de proporcionar ao usuário uma experiência realista no treinamento com baixo tempo de atraso entre a comunicação entre CT e ST, haja vista que o CT e ST serão localizados em regiões distintas, ou em até mesmo em cidades distintas.

Esse conceito de abranger a comunicação entre cidades distintas é de grande importância para a proposta. Por exemplo, a ST pode ser localizada na cidade de Manaus, Amazonas, e o CT pode ser localizado em Uberlândia, Minas Gerais. Esta possibilidade cria proporções maiores de aplicação prática da proposta, e de grande importância, pois permite abranger pessoas com deficiência de todo o país. A partir deste conceito, surge o segundo desafio da proposta, que é de como criar essa arquitetura que cumpra os requisitos observados, e que ao mesmo tempo seja de baixo-custo, e apropriada para a realidade social-financeira do país.

Em continuação à explicação da proposta, outro fator importante é a forma de avaliar o desempenho do usuário em treinamento, a qual consiste em como criar uma forma eficaz de medir a evolução do usuário durante os treinamentos, bem como permitir até mesmo a comparação do desempenho obtido por diferentes métodos de condução, e desse modo determinar o mais adequado para o usuário. Por exemplo, o usuário poderá praticar utilizando métodos distintos, e a partir do desempenho alcançado, determinar qual o mais apropriado para a comandar a CRM segundo as suas dificuldades motoras.

Sintetizando as principais dificuldades sobre as questões técnico-científicas desta pesquisa, a primeira é o aspecto da sincronia das multitarefas de *software* e *hardware*. Citando caso análogo, este sistema poderá incluir múltiplas *threads* em seu *software,* onde essas farão o papel de comunicação a longa distância, bem como, farão os comandos em baixo nível para manipulação do driver dos motores da CRM. Além disso, a sincronia das tarefas deve permitir que o sistema flua sem interrupções indevidas, possibilitando então que o usuário manipule a CRM remotamente, e o treinamento não seja prejudicado por paradas indevidas no funcionamento. Este é então o primeiro grande desafio encontrado.

A segunda dificuldade trata-se de aspectos de comunicação a longas distâncias. Geralmente em cenários onde há comunicação entre cidades distintas, existe um atraso significativo entre o envio e o recebimento de mensagens, por exemplo, tratando um caso *master-slave,* sendo o *master o* usuário, e o *slave* a CRM. Assim, caso exista um atraso de 1 segundo, esse poderá representar uma dificuldade significativa em manipular a CRM, onde o

master realizará o comando remoto, mas não poderá visualizar de forma eficaz a posição atual do *slave*.

A terceira dificuldade é em relação à criação de uma arquitetura capaz de superar os dois problemas anteriores apresentados, e além disso ter um baixo-custo, sendo essa apropriada para a realidade social-financeira dos usuários em países em desenvolvimento como o Brasil. Esse aspecto é fundamental para permitir que o sistema seja replicado em diferentes cidades, visando então melhorar a qualidade de vida de pessoas com deficiência primeiramente dentro do Brasil, e posteriormente, podendo até mesmo possibilitar a aplicação em outros países.

Dentro do contexto apresentado, serão apontados os objetivos desta pesquisa, onde esses foram divididos em geral e específicos, separados de acordo com a implementação das funcionalidades do ambiente de treinamento, e provendo uma visão sobre como será conduzida a pesquisa e o desenvolvimento da proposta.

1.2 Objetivo Geral

Implementar um sistema realista de treinamento para novos usuários de cadeira de rodas motorizada capaz de suportar múltiplos métodos de condução e de ser operado remotamente.

1.2.1 Objetivos Específicos

- Desenvolver um modelo de teleoperação abordando o usuário em treinamento e a CRM.
- Analisar o padrão de sinais utilizados para movimentação da CRM real.
- Implementar um método para gerar os mesmos padrões de sinais originais da CRM e comandá-la remotamente.
- Desenvolver uma arquitetura para comunicação a longas distâncias entre o usuário e a CRM.
- Desenvolver múltiplos métodos de condução para treinamento, incluindo um método genérico.
- Desenvolver uma ferramenta de avaliação do desempenho do usuário em treinamento.

1.3 Organização dos Capítulos

Este trabalho está organizado da seguinte forma, o Capítulo 2 apresenta os principais fundamentos sobre as tecnologias e ferramentas aplicadas, dando ênfase em quesitos teóricos e práticos relacionados ao sistema operacional Linux, redes de computadores e implementação de *software* no paradigma de programação orientada a objeto. O Capítulo 3 descreve o atual estado da arte de ambientes de treinamento de cadeira de rodas, bem como sobre sistemas de teleoperação, onde é feita uma análise detalhada sobre os trabalhos relacionados, com foco nos ambientes virtuais e nas técnicas e ferramentas utilizadas para teleoperação. O Capítulo 4 apresenta a arquitetura desta proposta, e introduz o conceito do sistema de teleoperação de CRM. Em seguida, apresenta os métodos de condução, os sinais elétricos necessários para manipulação remota, *hardware* aplicado, técnicas de comunicação e os métodos de avaliação do desempenho do usuário. O Capítulo 5 descreve a fase de experimentação e discussão, onde primeiramente são apresentados os cenários propostos e logo os resultados obtidos. Na etapa seguinte são discutidos interpretados os resultados e apresentada a contribuição desta pesquisa. Finalmente, o Capítulo 6 aborda as conclusões, considerações finais e trabalhos futuros.

2. Conceitos Fundamentais Sobre as Tecnologias e Ferramentas Aplicadas Nesta Pesquisa

Este capítulo tem como objetivo apresentar os principais conceitos sobre as tecnologias e ferramentas utilizadas nesta pesquisa. A divisão do capítulo ocorre da seguinte forma, primeiramente são abordados conceitos sobre o sistema operacional Linux embarcado, que são fundamentais para o entendimento da arquitetura proposta em termos de interpretadores de linguagem, compilação, e interfaces de comunicação. Em seguida são apresentados fundamentos sobre redes de computadores, que formam a base da comunicação a longas distância da proposta, onde são abordados quesitos como protocolos de comunicação, nomenclaturas de telecomunicações, ferramentas para *streaming* de vídeo e análise do tráfego de rede. Finalmente são apresentados conceitos de desenvolvimento de *software* baseados em POO (Programação Orientada a Objetos), no qual são abordados assuntos como *threads*, ferramentas para criação de interfaces gráficas, e um exemplo prático de servidor e cliente *socket* UDP (*User Datagram Protocol)*.

2.1 Sistema Operacional Linux Embarcado

O sistema operacional Linux oferece confiabilidade e eficiência para os sistemas embarcados. Por ser um sistema *open-source*, recebeu diversas melhorias e otimizações em termos de performance, funcionalidades, drivers e portas de comunicação, oferecendo robustez e versatilidade para ser aplicado a um sistema embarcado, sendo adequado para customização visando atender requisitos específicos. O Linux embarcado pode ser aplicado em microprocessadores de diversas arquiteturas x86, SPARC, ARM, PowerPC, MIPS e SuperH (Hollabaugh, 2002).

A distribuição Linux inclui um framework para desenvolvimento e várias aplicações de *software* desenvolvidas especificamente para esta proposta. Este framework inclui diversas ferramentas como compiladores (gcc, *debuggers*, criadores de imagem (*boot*), navegadores, entre outros (Yaghmour, 2003).

A partir do Linux é possível acessar dispositivos e demais periféricos conectados ao sistema em baixo nível (kernel). Por exemplo, é possível acessar a interface serial Rx/Tx, a partir do protocolo UART, sendo essas acessadas diretamente pelo kernel, criando então uma

vantagem em relação a outros SOs em quesitos de acessibilidade, como portas GPIO, protocolos I2C, SPI, CAN, os quais são comumente aplicados em projetos baseados em sistemas embarcados.

O Linux também possui uma dimensão computacional reduzida em relação a outros SOs. Esse fator é de fundamental importância para sistemas embarcados, pois geralmente há limitações de espaço para memória, sendo possível customizar quais pacotes e funcionalidades serão de fato necessário para a aplicação do sistema, como pacotes de *networking*, mídia, acesso remoto (ssh), interpretadores de linguagem (python), entre outros.

Para o caso de utilizar distribuições baseadas em versões amplamente aplicadas, como por exemplo Linux Ubuntu, é possível obter suporte de repositórios constantemente atualizados, o que facilita a utilização de drivers tanto quanto pacotes da camada de usuário de uma forma geral.

Considerando uma visão generalizada, pode-se dividir a arquitetura do Linux em dois níveis, conforme apresentado na Figura 1.

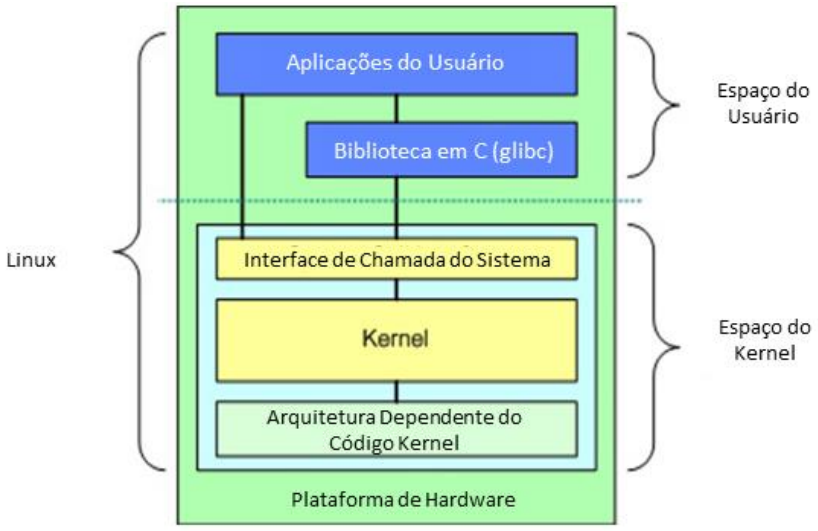


Figura 1 - Arquitetura base de um Sistema Operacional Linux (Ibm, 2007).

Na camada superior está o espaço do usuário, no qual estão as aplicações, assim como as bibliotecas de acesso (glibc) aos dispositivos em baixo nível (kernel). Na camada inferior (espaço do kernel) está presente a interface de chamada do sistema, que pode ser acessada diretamente pelas aplicações do usuário e pelas bibliotecas em C (glibc). Abaixo está o kernel do SO, responsável pela ligação entre a interface de chamada do sistema e a camada dependente da arquitetura do código kernel. Nessa camada, está o chamado (BSP – *Board Support Package*), responsável por inicializar o *hardware* (processador, barramentos, *clock*, memória RAM, *boot loader*, entre outros).

O sistema operacional Linux é a base fundamental para o sistema embarcado da CRM utilizada nesta pesquisa. Onde os conceitos apresentados são de suma importância para a arquitetura do SO desta proposta.

2.2 Rede de Computadores e Telecomunicações

Esta subseção é fundamental à compreensão dos quesitos de comunicação desta pesquisa, já que os conceitos apresentados são utilizados nas etapas de desenvolvimento, tal como nos critérios de avaliação experimental da CRM. Serão primeiramente abordados conceitos teóricos sobre protocolos de comunicação UDP e HTTP (*Hypertext Transfer Protocol*), assim como as nomenclaturas RTT (*round-trip time*) e *throughput*. Em seguida, serão apresentadas ferramentas de análise de tráfego de rede e de implementação da funcionalidade de *streaming* de vídeo.

2.2.1 Conceito sobre UDP (*User Datagram Protocol*)

A Internet utiliza dois principais protocolos na camada de transporte, sendo um orientado à conexão entre servidor e cliente TCP (*Transmission Control Protocol*) e o outro não orientado à conexão (protocolo UDP). Através do protocolo UDP as aplicações enviam datagramas IP encapsulados sem haver a necessidade de uma conexão prévia, ou seja, sem necessidade da ação conhecida como "*handshake"*.

O UDP transmite segmentos que contêm o cabeçalho de 8 bytes, e em seguida a carga útil, conforme apresentado na Figura 2. As duas portas são utilizadas para identificar os pontos extremos nas máquinas de origem e destino. O cabeçalho IP, por sua vez, possui um tamanho mínimo de 20 bytes, devido às opções entre IPv6 e IPv4. O comprimento mínimo do UDP é de 8 bytes, para incluir o cabeçalho, e o comprimento máximo é de 65515 bytes (Wetherall e Tanenbaum, 2011).

Logo que o pacote UDP alcança o destino, o conteúdo é entregue ao processo associado à porta destino. A principal vantagem de utilizar o UDP em relação ao IP bruto é a adição das portas de origem e destino. Sem elas, a camada de transporte não teria propósito funcional aos pacotes recebidos, sendo que o uso delas permite a camada entregar o segmento encapsulado à aplicação correta.

O campo de *Checksum* fornece uma confiabilidade extra, pois valida os dados do cabeçalho UDP e a carga útil. Dessa forma, o *Checksum* considera todas as informações

transmitidas pelo datagrama UDP. Esse cálculo também considera um pseudocabeçalho composto por dados do cabeçalho IP. Este pseudocabeçalho é apresentado na Figura 3.

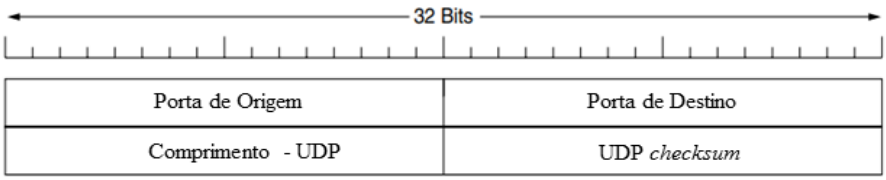


Figura 2 - Cabeçalho (*header*) do UDP (Wetherall e Tanenbaum, 2011).

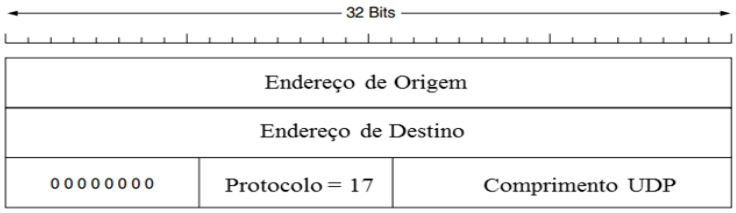


Figura 3 - Pseudocabeçalho do UDP incluído no *Checksum* (Wetherall e Tanenbaum, 2011).

Na Figura 3, o Protocolo 17, refere-se ao número correspondente do UDP. Vale ressaltar que o pseudocabeçalho é utilizado apenas para o cálculo do *Checksum*, não sendo efetivamente transmitido pela rede. Esse mecanismo pode ser desabilitado, quando isso acontece e o valor passado para o destino é 0. Se estiver habilitado e coincidentemente o resultado for 0, então é transmitido 16 bits em nível 1. Apesar do *Checksum* ter a possibilidade de ser desabilitado, essa prática não ocorre, pois se justificaria apenas para melhorar o desempenho em sistemas com *hardware* com performance significativamente baixa, o que não é realidade atual (Carissimi *et al.*, 2009).

2.2.2 Protocolo de Transferência de Hipertexto (HTTP)

O Protocolo de transferência de Hipertexto (HTTP) é utilizado na camada de aplicação, sendo um dos mais comuns, também considerado fundamental para o funcionamento da *World Wide Web* (*Internet)*. Esse protocolo é implementado utilizando cliente e servidor, e ambos estão presentes em sistemas diferentes, comunicando-se por troca de mensagens (Kurose, 2013).

O HTTP utiliza como protocolo subjacente o TCP (*Transmission Control Protocol*), que é um protocolo que garante a integridade da mensagem transferida entre cliente e servidor. Para a troca de mensagens, primeiramente o cliente inicia uma conexão TCP com o servidor. Uma vez que a conexão for estabelecida, o acesso é processado por meio das interfaces *socket*.

O cliente envia solicitações de mensagens para a interface *socket*, e em seguida recebe mensagens nessa mesma interface. O servidor funciona de forma similar, também com interface *socket*. Na Figura 4 é possível observar o funcionamento entre solicitação e resposta HTTP. O exemplo apresenta diferentes *browsers* (*Firefox* e *Internet Explorer*) como clientes de um servidor *Web Apache* (Kurose, 2013).

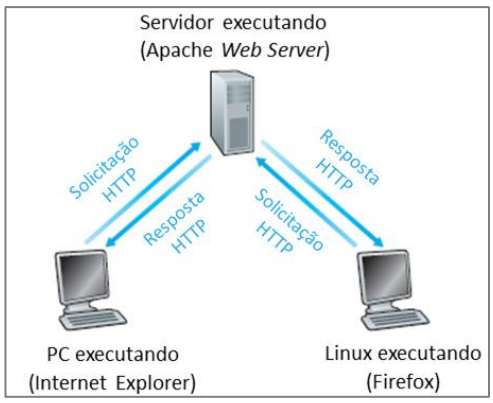


Figura 4 - Exemplo de comportamento do Protocolo HTTP (Kurose, 2013).

2.2.3 Terminologia *Throughput*

Dentro do contexto de rede de computadores, *throughput* é a quantidade de dados que está sendo transferida de um ponto a outro, geralmente representada em bps (*bits per second*). Por exemplo, considere a transferência de um arquivo entre os computadores A e B conectados na mesma rede, o *throughput* instantâneo em qualquer espaço de tempo é a taxa bps no qual o computador B está recebendo o arquivo (Kurose, 2013).

2.2.4 Descrição sobre RTT (*Round-Trip Time*)

O RTT pode ser definido como o tempo necessário para um pacote trafegar de um cliente ao servidor, e em seguida voltar para o cliente, conforme apresentado na Figura 5. O RTT inclui atraso de propagação do pacote, atraso de enfileiramento na camada de roteadores/*switches* e atraso referente ao tempo de processamento do pacote (Kurose, 2013).

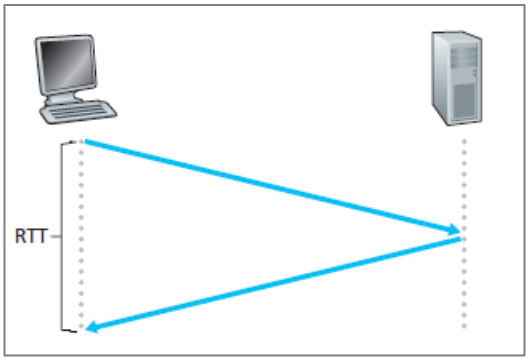


Figura 5 - Exemplo do Tempo de envio e recebimento de um pacote (RTT) (Kurose, 2013).

2.2.5 Ferramenta *Wireshark* para Monitoramento de Tráfego de Rede

O Wireshark é uma ferramenta em código aberto (*open source*) utilizada para o monitoramento de rede (Wireshark, 2018). Esta ferramenta permite filtrar os pacotes que trafegam por determinada placa de rede, seja ela com fio ou sem fio (*wireless)*. Através dessa ferramenta é possível verificar os *headers* dos pacotes, sendo esses os cabeçalhos dos pacotes que permitem identificar endereços de IP, portas, protocolos, e demais informações. O Wireshark também permite medir a quantidade de pacotes que trafegam, tamanho médio dos pacotes, *throughput*, entre outras funcionalidades. Para explicar o funcionamento do *software*, foi criado um exemplo de comando com filtragem por endereço de IP, tipos de protocolo e portas.

No exemplo do comando de filtragem é possível observar a utilização de um endereço IP através do comando `“ip.addr==192.168.1.117”`. Também foram utilizados operadores lógicos, onde “`&&`” faz o papel da porta *AND*. Outra característica é o uso de protocolos e portas, no qual “`udp.port==41897`” representa a filtragem por protocolo UDP, bem como pela porta 41897. Essa ferramenta será de grande importância para o projeto em quesitos de análise do tráfego de pacotes na rede durante a etapa de experimentação.

Comando exemplo do Wireshark para filtragem de pacotes capturados em uma placa de rede

```
ip.addr==192.168.1.117 && udp.port==41897 && ip.addr==200.129.152.97 && udp.port==4444
```

2.2.6 *Streaming* de Vídeo

O *streaming* de vídeo permite que usuários de uma rede de computadores possam assistir a um vídeo sem precisar de download prévio do conteúdo. No procedimento de transmissão, o vídeo é segmentado em diversas partes e compactado para ser enviado ao computador destino.

Para a transmissão de *streaming* são necessários três elementos básicos:

Cliente: São os *plug-ins* ou programas existentes nos computadores dos usuários que permitem a recepção da transmissão.

Servidores: Computador onde se encontra o *hardware* e *software* necessário para fornecer o *streaming*.

Codecs: Os codecs são tecnologias de codificação que possuem dois componentes: o *encoder* codifica o vídeo no formato desejado; o *decoder* decodifica para que possa ser reproduzido. O termo codec é a contração entre as palavras *decoder* e *encoder*. O codec é utilizado para conversão de vídeo para os diversos formatos de reprodução dos programas de *streaming* (Ozer, 2016). É importante ressaltar que para o *streaming* de vídeo o conteúdo após passar pelo *enconder* é armazenado em um *container* para que seja transmitido. Os *containers* são responsáveis por transportar o conteúdo dos vídeos codificados do ponto de origem ao destino.

2.2.6.1 Ferramenta ffmpeg para *Streaming* de Vídeo

O ffmpeg é uma ferramenta aberta (*open source*) composta por várias funcionalidades relacionadas à mídia, como por exemplo, vídeo *streaming*, *codecs*, conversão de formato de vídeo, entre outras (Ffmpeg, 2018). O código fonte dessa ferramenta é fornecido pelo site da plataforma, sendo essa uma vantagem, pois permite a compilação na arquitetura alvo do ambiente de utilização, como por exemplo em um SO Linux baseado em ARM, MIPS, x86 ou demais. Outra vantagem da ferramenta é a documentação detalhada fornecida e o suporte da comunidade de desenvolvedores. A utilização da ferramenta é baseada no uso de *flags*, que são comandos responsáveis por configurar e iniciar as funcionalidades. Foi criado um comando de exemplo para demonstrar o uso do ffmpeg para *streaming* de vídeo.

Primeiramente, utiliza-se a flag "`-i`" para direcionar a *path* da câmera de gravação do vídeo, sendo a *path* o caminho do diretório onde se encontra a câmera. Em seguida a resolução de gravação pode ser escolhida a partir da flag "`-s`", onde no exemplo foi de 640 pixels x 480 pixels. Logo após, é escolhida a taxa de quadros por segundo, do inglês fps (*frames per second*), que, para o exemplo, foi de 15 fps. A próxima *flag* "`-f`" define o formato do container do vídeo,

onde no exemplo foi utilizado *mpegts*. A seguinte "`-fflags`" define o buffer para a transmissão, onde no exemplo foi escolhido sem buffer. Isso significa que todo conteúdo gravado vai ser diretamente transmitido, sem que seja armazenado nenhum conteúdo para uma sincronização prévia com o destino. A flag seguinte "`-listen 1`" significa que o servidor de vídeo streaming está preparado para realizar sincronização com o cliente através do endereço definido. No exemplo é utilizado o protocolo HTTP, seguido pelo endereço IP e pela porta do servidor.

Comando exemplo do ffmpeg para *streaming* de vídeo

```
ffmpeg -i /dev/video2 -s 640x480 -r 15 -f mpegts -fflags nobuffer -listen 1 http://IP:PORTA
```

2.3 Conceitos e Ferramentas de *Software*

Nesta subseção são apresentados quesitos relacionados ao desenvolvimento do *software* da proposta. São abordados conceitos sobre o funcionamento de *threads* no paradigma POO, onde são demonstrados quesitos práticos de implementação. Na etapa subsequente é apresentada uma ferramenta para a criação de interface gráfica, sendo essa utilizada na etapa de interatividade entre o usuário e o ambiente de treinamento.

2.3.1 *Threads* em *Software*

A *thread* é uma sequência única de fluxo de controle dentro de um programa computacional (Microsystems, 2005). Executar várias *thread*s é similar à execução de vários programas paralelamente com os seguintes benefícios:

- Múltiplas *thread*s em um processo compartilham os mesmo espaço de dados com a *thread* principal, portanto podem compartilhar informação ou comunicar-se entre si mais facilmente em relação a dois processos separados.
- As *threads* podem ser tratadas como processos menos custosos, os quais não requisitam da mesma proporção a memória exigida em processos convencionais.

Uma *thread* possui um começo, uma sequência de execução e uma conclusão, sendo que um ponteiro de instrução rastreia onde está ocorrendo a execução.

Outros aspectos relevantes sobre a *thread* são:

- Podem ser preemptadas (interrompidas);
- Podem ser colocadas em modo de espera (*on hold*) enquanto outras estão em execução.

2.3.2 Inicializando uma Nova *Thread*.

Para explicar o funcionamento sobre a inicialização de uma *thread*, é utilizada a linguagem Python 2.7, como também o conceito de POO (Python, 2018a). A base da explicação é o *Script* 1. Nesse script, iniciam-se duas *threads*, onde cada uma apresenta uma marcação de tempo com intervalos de duração diferentes. A primeira *thread* imprime na tela a marcação de tempo em intervalos de 1 s, e a segunda imprime com intervalos de 2 s. Ambas sendo executadas paralelamente.

No *Script* 1, primeiramente foi importado o módulo *threading*, o qual é responsável pela estrutura de uma *thread* em Python (Python, 2018b). Os módulos *time* e *datetime* também foram importados para permitir a funcionalidade de espera (*delay*), assim como obter um *timestamp* (marcação de tempo) e converter esse valor para horas, minutos e segundos. Em seguida foi criada uma classe chamada *Thread_exemplo*, a qual herda as características da classe *Thread* do módulo *threading*.

Na próxima etapa, estão as funções, as quais são iniciadas por "`def`". Foram criadas duas funções: a primeira "`__init__`" é responsável por criar a *thread* com a construção do objeto. A segunda "`run`" é responsável pela tarefa de execução do objeto. Essa recebe como parâmetro de entradas o nome da *thread*, como também, um valor que representa um atraso (*delay*).

A função "`run`" é iniciada com um "`try/except`". Essa estrutura é utilizada para tratar possíveis erros e exceções que venham ocorrer durante a execução do código. Dentro da estrutura "`try`" é obtida a marcação atual de tempo e também é feita a impressão da mensagem contendo o nome da *thread* com a respectiva marcação de tempo.

Após a etapa das funções, está a estrutura principal de execução do código. Nessa etapa são instanciados dois objetos da classe `Thread_exemplo`. Foram criadas duas *threads* a partir desses objetos. A estrutura `threading.Thread(target=obj_1.run, args=("primeira thread", 1))` funciona da seguinte forma: o alvo "`target`" recebe a função de execução do objeto criado. E o "`args`" (*arguments*) recebe os argumentos de entrada da função "`run`" do objeto instanciado. Logo após, cada *thread* é iniciada a partir do método *start*.

Script* 1 - Exemplo prático da criação de *threads

```
import time
import threading
import datetime

class Thread_exemplo(threading.Thread):

    def __init__(self):
        threading.Thread.__init__(self)

    def run(self, thread_name, delay):
                try:
                        while True:
                                ts = time.time()
                                hora = datetime.datetime.fromtimestamp(ts).strftime('%H')
                                minutos = datetime.datetime.fromtimestamp(ts).strftime('%M')
                                segundos = datetime.datetime.fromtimestamp(ts).strftime('%S')
                                print thread_name + " - "+ hora + ":" + minutos + ":" + segundos
                                time.sleep(delay)
                except KeyboardInterrupt:
                        pass


obj_1 = Thread_exemplo()
thread_1 = threading.Thread(target=obj_1.run, args=("primeira thread", 1))
thread_1.start()

obj_2 = Thread_exemplo()
thread_2 = threading.Thread(target=obj_2.run, args=("segunda tread", 2))
thread_2.start()
```

Resultado da execução do *Script* 1

```
primeira thread - 13:33:18
primeira thread - 13:33:19
segunda tread - 13:33:20
primeira thread - 13:33:20
primeira thread - 13:33:21
segunda tread - 13:33:22
primeira thread - 13:33:22
primeira thread - 13:33:23
segunda tread - 13:33:24
primeira thread - 13:33:24
primeira thread - 13:33:25
segunda tread - 13:33:26
primeira thread - 13:33:26
primeira thread - 13:33:27
segunda tread - 13:33:28
```

No resultado do *Script* 1 é possível observar a primeira thread imprimindo na tela valores com intervalos de 1 s, do mesmo modo que a segunda thread imprime com intervalos de 2 s. O conceito desse *script* será na apresentação da arquitetura de *software* do sistema, como por exemplo, para criação de servidor e cliente *socket* UDP no ambiente de treinamento proposto.

2.3.3 Ferramenta Tkinter para Interface Gráfica do Usuário

O conceito de interface gráfica do usuário, ou do inglês GUI (*graphical user interface*) será utilizado no desenvolvimento da pesquisa O objetivo da GUI é permitir a interatividade do usuário com as funcionalidade do ambiente de treinamento. Vale ressaltar que a ferramenta utilizada para criação será o Tkinter, que se trata de um módulo da linguagem de programação Python (Tkinter, 2018). Através do Tkinter é possível criar *widgets*, que são aspectos customizáveis da interface, como por exemplo botões de controle, blocos para entrada de texto, e elementos gráficos para agregar informações à GUI. Para explicação da ferramenta Tkinter, foi criado um *script* simplificado em Python, o qual utiliza um botão de evento.

No *Script* 2, foi primeiramente importado o módulo Tkinter, e em seguida criada a classe *Application*, com parâmetro de entrada *Frame*., o qual é responsável pela criação da janela da GUI. Em seguida são criados os três métodos: "`__init__`" (construção da janela); "`create_widgets`" (criação do botão de eventos, customização dos parâmetros do botão e exposição na janela); e o último método "`update_count`" (atualiza a contagem do número de *clicks* do botão e expõe o valor atualizado na tela). Na etapa seguinte é instanciado o objeto *root* da classe Tk (importada do módulo Tkinter), como também os parâmetros como título da janela e tamanho da resolução da janela em (pixels x pixels). Finalmente é instanciada a aplicação, e colocada em *loop* até que ocorra o fechamento da janela.

***Script* 2 - Exemplo da criação de uma GUI simplificada através do Tkinter**

```
from Tkinter import *

class Application(Frame):

    def __init__(self,master):
        """Inicia o Frame"""
        Frame.__init__(self,master)
        self.grid()
        self.num_clicks = 0  # verifica o numero de clicks do botao
        self.create_widgets()

    def create_widgets(self):
        """Cria um botao para apresentar o numero de clicks"""
        self.botao = Button(self)
        self.botao["text"] = "Total de clicks: 0"
        self.botao["command"] = self.update_count
        self.botao.grid()

    def update_count(self):
        """Aumenta o numero de clicks e apresenta o valor atualizado"""
        self.num_clicks += 1
        self.botao["text"] = "Total de clicks: " + str(self.num_clicks)

root = Tk()
root.title("Botao de evento")
root.geometry("200x100")
app = Application(root)
root.mainloop()
```

Na Figura 6 é apresentado o resultado, no qual é possível observar duas telas distintas. A primeira se refere à tela inicial obtida na execução do *Script 2*. A segunda tela se refere à janela após um *click* no botão, como também ao valor total de *clicks* atualizado. O conceito apresentado no *Script* 2 será utilizado no desenvolvimento do sistema, como por exemplo, na criação da IHM (interface humano-máquina) para condução remota da CRM.

Tela inicial Tela após um click no botão “Total de clicks”

Figura 6 - Resultado da execução do *Script* 2

2.3.4 Comunicação via Rede Através da Interface Socket UDP

O *socket* permite o envio de mensagens de um cliente para um servidor ou vice-versa, onde ambos estão conectados na mesma rede de computadores, podendo ser ela local ou através da Internet (Socket, 2018). O *socket* em conjunto com o protocolo UDP possibilita que as mensagens sejam enviadas sem que haja uma sincronização prévia entre servidor e cliente. Para o envio das mensagens, utiliza-se o endereço de IP e a porta destino. Para demonstração do funcionamento do *socket* foram criados dois *Scripts* (3 e 4), os quais representam o servidor e o cliente, respectivamente. O caso apresentado é um exemplo prático da utilização dessa ferramenta em conjunto com a linguagem de programação Python 2.7. Os objetivos dos *Scripts* 3 e 4 são de realizar o envio de uma mensagem do cliente para o servidor, o qual imprime na tela o IP e porta do cliente, seguido pela mensagem recebida. A explicação detalhada iniciará pelo *Script* 3 (servidor), para em seguida abordar o *Script* 4 (cliente).

No *Script* 3 primeiramente são importados os módulos “`socket`” “`sys`” e “`Thread`”. O módulo “`socket`” contém os métodos necessários para criação, envio e recebimento de mensagens; o módulo “`sys`” contém comandos do sistema que possibilitam, por exemplo, a interrupção da execução do código; o último módulo “`Thread`” foi apresentado em detalhes na subseção anterior. Foi definido o IP do servidor como 127.0.0.1, que representa o *loopback* (mesma máquina). E a porta utilizada foi 4444 (valor arbitrário). A função “`__init__`” tem um papel de iniciar uma thread, instanciar um objeto *socket* e em seguida vincular o objeto ao IP e porta definidos. A função “`run`” é responsável por receber as mensagens do cliente, e imprimir

na tela o conteúdo, o IP e a porta do cliente. Essa função fica em loop infinito até o fechamento da janela. Na etapa final do código é instanciado o objeto e colocado em execução.

No *Script* 4 foi criada uma classe "`socket_client_udp`" baseada em thread que envia uma mensagem para o endereço IP e porta do servidor, e, em seguida, finaliza a execução. Essa classe possui uma estrutura similar à classe do "`Udp_server`" (*Script* 3), no entanto, com uma estrutura mais simplificada. A estrutura dessa classe representa um funcionamento prático de um cliente socket UDP.

***Script* 3 – Exemplo de servidor socket UDP**

```
import socket
import sys
from threading import Thread

HOST = "127.0.0.1"
PORT = 4444

class Udp_server(Thread):
    def __init__(self):
        global socket_server
        """Cria um servidor sockect e conecta-o ao IP e PORTA"""
        Thread.__init__(self)
        try:
            socket_server = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
            print "Servidor socket criado"
        except socket.error, msg:
            print "Falha ao criar o Servidor"
            sys.exit()

        try:
            socket_server.bind((HOST, PORT))
        except socket.error, msg:
            print "Falha ao vincular o servidor a porta"
            sys.exit()

        print "Servidor vinculado ao IP:" + HOST + " Porta:" + str(PORT)

    def run(self):
        """Recebe mensagens"""

        global socket_server

        while 1:
            # Recebe mensagens do cliente (dados, endereco)
            d = socket_server.recvfrom(1024)
            dados = d[0]
            addr = d[1]

            if not dados:
                break

            print 'Mensagem do cliente [' + addr[0] + ':' + str(addr[1]) + '] - ' + dados

server = Udp_server()
server.run()
```

***Script* 4 – Exemplo de cliente socket UDP**

```
import socket
from threading import Thread

UDP_IP = "127.0.0.1"
UDP_PORT = 4444

class socket_client_udp(Thread):
    def __init__(self):
        Thread.__init__(self)
        self.sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

    def send_message(self, message):
        self.sock.sendto(message, (UDP_IP, UDP_PORT))

conn = socket_client_udp()
mensagem = "Eu sou o cliente"
conn.send_message(mensagem)
print "Mensagem enviada"
```

Na Figura 7 é possível observar o resultado gerado após a execução de ambos: cliente e servidor UDP. No *prompt* de comando da direita encontra-se o cliente, e no da esquerda, encontra-se o servidor. A mensagem do cliente recebida é impressa pelo servidor, o qual também apresenta o número de IP e a porta de conexão com o cliente. A estrutura base apresentada sobre as classes de servidor e cliente *socket* UDP baseadas em *Thread* são necessárias para o entendimento sobre a arquitetura de comunicação proposta, a qual possibilita a comunicação remota entre o usuário em treinamento e o ambiente proposto.

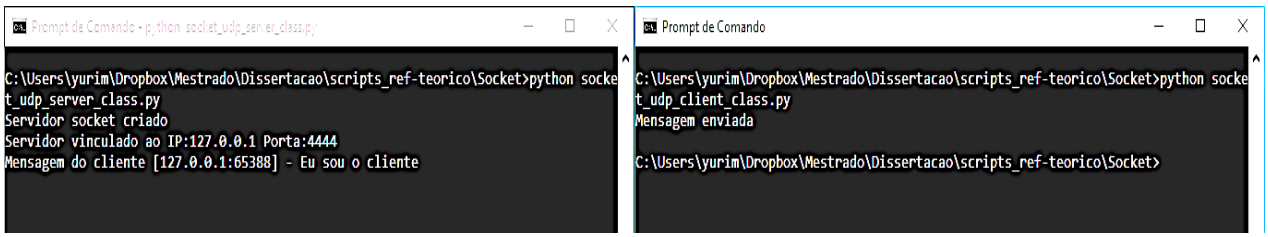


Figura 7 - Resultado da execução dos *Scripts* 3 e 4 para comunicação cliente servidor socket UDP.

2.4 Conclusão do capítulo

Neste capítulo foram apresentados os principais conceitos e fundamentos sobre as tecnologias aplicadas. Foi apresentado o SO Linux embarcado, o qual será aplicado como base para o sistema de teleoperação implementado. Também foram abordados conceitos e nomenclaturas sobre rede de computadores, que formarão a base para a comunicação entre o usuário em treinamento e o ambiente proposto.

Além disso, foi apresentada a biblioteca para *streaming* de vídeo (ffmpeg) e o *software* para análise de qualidade de comunicação (Wireshark). Na etapa final foram apresentados fundamentos necessários para a implementação do *software*, como *threads,* interface gráficas e *socket* UDP. Esses conceitos também serão necessários para entendimento do próximo capítulo, no qual será apresentado o estudo sobre o estado da arte de sistemas baseados em teleoperação, e um estudo sobre os ambientes de treinamento para CRM.

3. Trabalhos Relacionados

Neste capítulo é apresentada a revisão da literatura sobre ambientes de treinamento de CRM com foco no *gap* de pesquisa, e em aspectos que podem contribuir para esta proposta. Ademais, é apresentado um estudo sobre sistemas baseados em teleoperação, onde são verificados os métodos e ferramentas. O objetivo é escolher os mais adequados para as configurações do ambiente proposto, considerando como requisito o baixo custo de implementação e aplicação em ambientes fechados (*indoor*), como por exemplo clínicas de reabilitação física e hospitais. Desse modo, a revisão da literatura foi dividida em duas partes. A primeira tem como foco trabalhos relacionados a ambientes de treinamento para CRM. São analisadas as seguintes questões:

- Quais tipos de ambiente de treinamento para cadeira de rodas motorizada estão sendo utilizados na literatura? Realidade virtual, aumentada ou ambiente real?
- Nestes ambientes, quais métodos de condução estão sendo utilizados?

A segunda parte tem como objetivo verificar o estado da arte sobre sistemas baseados em teleoperação. Serão verificados os seguintes quesitos:

- Quais técnicas estão sendo utilizadas para teleoperação a longas distâncias com baixo custo?
- Dessas técnicas, quais são apropriadas para ambientes fechados (*indoor*), como, por exemplo, hospitais e centros de reabilitação?

A condução da pesquisa dos trabalhos relacionados foi feita utilizando as bases de dados do IEEE *Xplore Digital Library*, *Web of Science*, ACM *Digital Library* e PubMed, as quais são bases renomadas nas áreas das engenharias elétrica, computação, telecomunicação e biomédica (Acm, 2018; Ieee, 2018; Pubmed, 2018; Science, 2018).

3.1 Trabalhos Relacionados sobre Ambiente de Treinamento para Usuários de CRM

Morere e seus co-autores desenvolveram um simulador para treinamento de CRM. Foi criado um ambiente virtual com pessoas e objetos no cenário para representar uma situação mais próxima da realidade. Com o simulador, os usuários puderam praticar de forma segura, mesmo quando foram submetidos a situações de risco. Nos resultados, foi observado que ao cumprirem o treinamento, os usuários sentiram-se mais motivados e confiantes (Morere et al., 2015). No entanto, foram identificadas dificuldades quanto à representação da velocidade da cadeira de rodas, onde, segundo os autores, não era equivalente a uma CRM real. Outro aspecto relevante foi a implementação apenas do método de condução por *joystick*, não permitindo então a prática a partir de outras tecnologias, como, por exemplo, o rastreamento ocular.

Rodriguez desenvolveu um ambiente de treinamento virtual para cadeiras de rodas para crianças. Segundo a autora, o treinamento por simulação apresenta potencial para aplicação nas áreas de reabilitação, pois permite aos usuários que repitam diversas vezes os comandos de manipulação da CRM, possibilitando então o progresso de acordo com o ritmo de aprendizado de cada usuário. Em relação à implementação desse ambiente, para adequá-lo para crianças, foram adaptados aspectos visuais de simulação com o objetivo de despertar a atenção e o interesse delas durante o treinamento. Sobre o método de condução, foi aplicado apenas o *joystick*, não abrangendo métodos alternativos (Rodriguez, 2015). O trabalho apresentado expôs um grande potencial em alcançar um tipo específico de pessoas, como por exemplo, crianças com deficiências. Além disso, foi exposto o potencial na utilização de uma forma de treinamento seguro, na qual os usuários pudessem repetir diversas vezes as ações de acordo com o ritmo de aprendizado de cada um.

John e seus co-autores fizeram um estudo para comparar o desempenho de ambientes de treinamento que utilizam como *feedback* visual o *Desktop* (com monitor convencional para computador) e o *Head Mounted Display* (dispositivo também chamado de óculos de realidade virtual). Foi verificado, através de experimentos, que a evolução no desempenho do grupo que treinou com *HMD* foi 5% maior em comparação ao grupo *Desktop*. Ademais, observou-se que a utilização do *HMD* permitiu aos usuários uma maior liberdade visual para as áreas ao redor da cadeira, auxiliando na execução de manobras que exigissem maior destreza. No entanto, houve um problema em relação à *cybersickness*, que é o enjoo proveniente do uso de *HMD*. Também nesse estudo, comparou-se a condução feita no ambiente simulado e no mundo real. Observou-se que apesar de diferenças sutis, os usuários realizaram um número maior de

manobras no ambiente virtual em comparação com o real. Outra diferença foi um maior tempo necessário para o cumprimento de percursos e um maior número de colisões no ambiente virtual. Foi observado que embora mostrassem potencial, os ambientes virtuais precisavam ser menos desafiadores se quisessem motivar e melhorar a performance dos usuários de maneira efetiva (John *et al.*, 2017).

O trabalho apresentado investigou o uso de outras tecnologias para *feedback* visual do usuário, como o *HMD*, que pode ser utilizado como alternativa para treinamento. Entretanto, foi apresentado um problema dessa tecnologia em relação ao enjoo sentido pelos usuários, chamado de *cybersickness*, podendo então ser um impedimento para esse tipo de treinamento, principalmente no caso de várias repetições, que irão expor os usuários diretamente a essa situação. Outro aspecto importante observado pelos autores foi em relação à dificuldade na representação do comportamento físico da CRM, que mais uma vez vem sendo citado como um problema de ambientes de treinamento virtual.

Archambault e seus co-autores desenvolveram um simulador de CRM, e realizaram um estudo para comparar a condução entre o dispositivo virtual e o real. Os usuários em treinamento precisavam cumprir diferentes tarefas, como, por exemplo, conduzir de ré, conduzir para frente e dobrar 90º, conduzir em trechos com espaço reduzido, entre outras. Os autores chegaram à conclusão que, em média, a metade das tarefas realizadas no ambiente virtual, se mostraram ser mais difíceis, a partir do contexto em que os participantes aumentaram o número de movimentos com o joystick e o tempo de conclusão do percurso. O experimento foi realizado por usuários com expertise na condução de cadeira de rodas e terapeutas ocupacionais, e alguns dos participantes mencionaram sentir como se estivessem jogando um jogo virtual (Archambault *et al.*, 2011). O trabalho deles foi outro exemplo da dificuldade em se criar um ambiente de treinamento mais próximo da realidade, também elucidando o problema de causar a impressão que os usuários estão em um jogo virtual.

Rossol e seus co-autores projetaram uma estrutura para o treinamento virtual de CRM. Foi criado um *software* com a possibilidade de personalizar o cenário de condução, a fim de adicionar obstáculos como cadeiras, cones e mesas. Também foi apresentado um método para alocar automaticamente os objetos de acordo com o nível de habilidade de condução do usuário (Rossol et al., 2011). O trabalho investigou o uso de ambientes de treinamento customizáveis, que possibilitam ajustar os níveis de dificuldade dos percursos de acordo com o nível de habilidade de cada usuário, sendo então um método que apresenta potencial para ambientes virtuais de treinamento No entanto, como (Archambault et al., 2011; Rossol et al., 2011; Morere et al., 2015; Rodriguez, 2015; John et al., 2017) , esses autores apenas desenvolveram

o *joystick* como um método de condução, impossibilitando o treinamento a partir de novas tecnologias adequadas para pessoas com deficiência severa.

Caetano e seus co-autores apresentaram uma proposta em fase inicial sobre ambiente baseado em realidade aumentada, utilizando um robô móvel para representação de uma CRM. A realidade aumentada utiliza o mundo real combinado com objetos virtuais, no qual o usuário visualiza a imagem real do local, com adição de elementos virtuais. Para possibilitar essa tecnologia, os autores utilizaram marcadores visuais no mundo real, contendo QR *code* (código de barras 2D). A partir desses marcadores, foram utilizados métodos de visão computacional para localizar os objetos, bem como métodos de computação gráfica para adicionar elementos virtuais, por exemplo, cones. Para representar uma CRM foi utilizado uma espécie de carro em miniatura, combinado com um celular exercendo o papel de câmera (Caetano *et al.*, 2014).

O trabalho despertou um grande potencial de utilizar CRM real para o treinamento, manipulada remotamente, possibilitando um treinamento mais realista que um virtual e igualmente seguro. Todavia, foram apresentados apenas resultados iniciais, nos quais a fase de experimentação foi feita a curta distância, não sendo avaliados quesitos como qualidade de comunicação, atraso entre o envio de comandos e movimento real do dispositivo em caso de condução remota a longas distâncias. Tampouco foi utilizada uma CRM real no ambiente proposto. Outrossim, o método de condução implementado foi apenas por *joystick*.

Tao desenvolveu um ambiente virtual de treinamento para cadeira de rodas que utiliza um *joystick* e um manipulador por movimentos das mãos. Nesse ambiente, usuários completavam um trajeto, e paralelamente precisavam alcançar objetos virtuais com a mão, a partir do manipulador. Foram medidos o tempo de cumprimento do trajeto, a quantidade de movimentos realizados no *joystick*, assim como o tempo e a quantidade de movimentos para alcançar os objetos virtuais com a mão. O resultado mostrou baixa representação cinemática da cadeira de rodas no ambiente virtual, porém com estratégias de condução similares. Verificou-se que no ambiente virtual foi necessário um maior tempo para cumprir as tarefas (Tao e Archambault, 2016). Esse trabalho investigou a adição de um método para interatividade entre o usuário e o ambiente, onde além de praticar as habilidades de condução, os usuários podem praticar o alcance de objetos dentro do ambiente virtual. Apesar disso, também foram mencionados problemas quanto à representação cinemática da CRM.

Como forma de sintetizar todos esses estudos, foram criadas duas tabelas que apresentam esses ambientes investigados. A Tabela 1 apresenta imagens dos simuladores, de modo a permitir uma visualização sobre os ambientes virtuais de treinamento implementados. A Tabela 2 apresenta uma comparação sobre esses ambientes, facilitando uma visão resumida

e generalizada das tecnologias aplicadas e critérios utilizados para avaliar o desempenho do usuário durante o treinamento.

Tabela 1 - Ambientes de treinamento para cadeira de rodas encontrados na literatura.

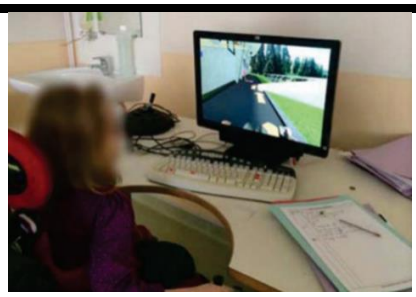
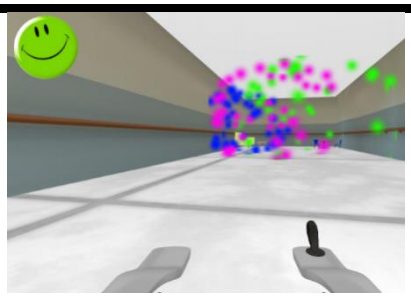
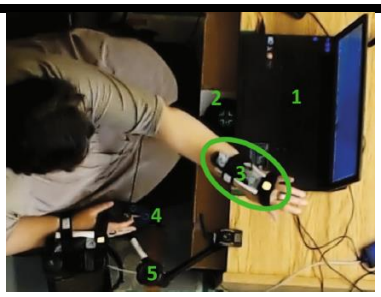
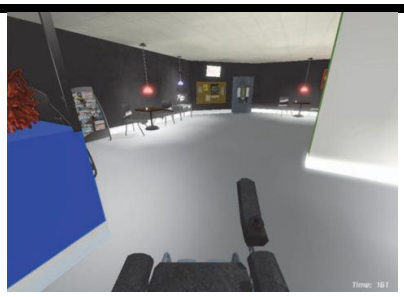
 Simulador ViEW (Virtual Electric Wheelchair) (Morere *et al.*, 2015)	 Usuário em treinamento no simulador ViEW (Morere *et al.*, 2015)
 Framework para treino virtual adaptativo (Rossol *et al.*, 2011)	 Simulador de cadeira de rodas para crianças (Rodriguez, 2015)
 Treinamento utilizando realidade aumentada (Caetano *et al.*, 2014)	 Ambiente de treinamento que utiliza *HMD (John et al., 2017)*
 Treinamento em ambiente virtual utilizando joystick e controlador por movimento das mãos (Tao e Archambault, 2016)	 Simulador miWE (McGill Immersive Wheelchair) (Archambault *et al.*, 2011)

Tabela 2 - Comparação entre ambientes de treinamento para cadeira de rodas.

Referência	Tipo de Ambiente de Treinamento	Feedback visual	Critério para avaliar o desempenho do usuário
(Morere *et al.*, 2015)	Virtual	Monitor	Para avaliação, foi definida uma pontuação baseada em critérios de condução (atravessar ruas, considerar elementos físicos do ambiente, não parar no meio de ruas, entre outros).
(Rossol *et al.*, 2011)	Virtual	Monitor	Foi utilizada uma abordagem probabilística para estimar o nível de destreza baseado em variáveis medidas como o tempo de percurso e o número de colisões. A partir deste nível, o sistema ajustava automaticamente o grau de dificuldade do percurso.
(Rodriguez, 2015)	Virtual	Monitor	Não apresentado.
(Caetano *et al.*, 2014)	Realidade Aumentada	*HMD*	Não apresentado.
(John *et al.*, 2017)	Virtual	Monitor/*HMD*	Foi medido o tempo de conclusão de percursos. E foi adicionada penalidade caso fossem atingidos obstáculos.
(Tao e Archambault, 2016)	Virtual	Monitor	Questionário para medir o nível de presença no ambiente virtual, bem como a quantidade de movimentos do joystick, o tempo de cumprimento dos percursos, além do tempo e quantidade de movimentos para alcançar objetos com o manipulador.
(Archambault *et al.*, 2011)	Virtual	Monitor	Foi medido o tempo do cumprimento de percursos e a quantidade de movimentos no joystick, além do nível de presença do usuário a partir de questionário.

3.1.2 Aspectos Observados sobre o Estado da Arte dos Ambientes de Treinamento para Usuários de Cadeira de Rodas Motorizada

Nesta subseção são respondidas as questões de origem à revisão de trabalhos relacionados, onde são observados os principais aspectos sobre ambientes de treinamento para novos usuários de CRM. São avaliados os aspectos positivos desses ambientes e é apresentado o *gap* de pesquisa no qual este trabalho será direcionado.

Sobre a primeira pergunta que diz respeito a quais tipos de ambiente de treinamento estão sendo usados na literatura, observou-se que a maioria trata de ambientes virtuais de treinamento. Nesses ambientes, o usuário utiliza um computador, no qual participa de percursos pré-estabelecidos, comandando uma CRM virtual. Foi observado um grande potencial nesses ambientes virtuais, pois possuem um baixo custo de infraestrutura, e permitem que o usuário possa evoluir as habilidades de condução, também o ajudando a melhorar aspectos como a autoconfiança e a familiarização com o dispositivo. Outro conceito importante observado foi que esses ambientes podem ser direcionados a diferentes tipos de usuários, como por exemplo, crianças com deficiência. Apenas um dos artigos encontrados tratava de um ambiente de realidade aumentada, entretanto, estava em etapa inicial de desenvolvimento.

Apesar desses pontos positivos sobre ambientes virtuais, verificou-se a dificuldade em relação à representação do comportamento físico da CRM, onde foram citados problemas em relação a criação de um ambiente realista. Também foi notado que na comparação entre comandar uma CRM virtual e uma real foram necessários mais comandos no dispositivo virtual, além da ocorrência de um número maior de colisões. Logo, se observa que esse ponto pode ser explorado, a fim de criar um ambiente de treinamento que seja mais próximo da realidade, eliminando o problema de simulação do comportamento da CRM, e dando ao usuário uma experiência mais realista.

Sobre a segunda pergunta que trata dos métodos de condução, observou-se que os trabalhos relacionados utilizaram *joystick* como base para o treinamento. Esse método de condução é de grande importância, pois é a forma convencional de condução de CRM, sendo opção primária de escolha dos usuários.

Apesar da grande importância do método de condução por *joystick*, existem usuários que têm grandes dificuldades ou até mesmo não conseguem comandar por esse método, devido a deficiências severas, como por exemplo, miopatia, paralisia cerebral, esclerose lateral amiotrófica, esclerose múltipla, entre outras. Essas deficiências podem afetar as capacidade de movimentação das mãos.

Assim, verifica-se a importância de abranger métodos alternativos de condução nos ambientes de treinamento. Esse aspecto pode ser observado para a criação de ambientes de treinamento, pois a inclusão ajuda a abranger um maior número de possíveis usuários para praticar no ambiente.

3.2 Trabalhos Relacionados à Sistemas Baseados em Teleoperação

Esta seção apresenta o atual estado da arte dos sistemas baseados em teleoperação. Para isso, alguns autores utilizaram diferentes arquiteturas de comunicação e infraestrutura de rede, como rede de celular (3G / 4G) e Internet de banda larga cabeada. Além disso, são apresentadas as atuais ferramentas de *streaming* de vídeo, os protocolos de rede e a métrica para avaliar a comunicação.

Xu e seus co-autores desenvolveram um sistema de teleoperação para um robô humanóide. Em seu trabalho, utilizaram a Internet através de uma rede 3G para a comunicação entre o operador (*master*) e o robô (*slave*). Para fornecer ao *master* um *feedback* em tempo real do *slave*, os autores aplicaram o *streaming* de vídeo. Portanto, para reduzir o *throughput* de troca de dados, foi aplicado o codec h.264. A ferramenta para o codec e *streaming* de vídeo foi o ffmpeg, que é uma biblioteca de código aberto. O protocolo de rede foi o RTP / UDP-IP (Xu *et al.*, 2012).

Ha e seus co-autores desenvolveram uma arquitetura para controlar uma cadeira de rodas à distância para ambientes externos. As mensagens para o esquema de comunicação *master-slave* foram enviadas através da Internet por uma rede 4G. O *streaming* de vídeo foi através do Skype. Para avaliar a comunicação, os autores mediram o tempo de ida e volta (*Round-Trip Time* - RTT) e a intensidade do sinal da rede celular. O *software* para realizar essa avaliação foi o Wireshark. Os protocolos de rede foram TCP / UDP-IP. Os autores usaram um computador desktop para o *slave* (Ha *et al.*, 2016).

Gonzalez e seus co-autores projetaram um sistema de telepresença para um helicóptero de controle remoto. O sistema utiliza dois *smartphones*, o primeiro, transmite o vídeo em tempo real usando o Skype, e o segundo captura fotos do alvo de acordo com os comandos do operador. Para a comunicação, foi aplicada uma infraestrutura celular 3G (Gonzalez *et al.*, 2012).

Manzi e seus coautores apresentaram um sistema genérico de teleoperação em nuvem para um robô móvel. Foi usado o Azure *Cloud Platform* e uma interface baseada na *Web* aplicável a dispositivos móveis (*smartphones* e *tablets*). O acesso à Internet foi feito através de uma rede 4G (Manzi *et al.*, 2016).

Shenai e seus co-autores criaram um modelo de interação telecirúrgica em tempo real. Seu trabalho permite que os participantes remotos tenham a mesma visão do médico durante o procedimento cirúrgico, sendo que esses participantes podem ajudar o médico através de

opiniões médicas, vídeo e interação de realidade aumentada. Foi utilizada uma conexão com Internet cabeada e um dispositivo de teleconferência comercial (Shenai *et al.*, 2014).

Kato e seus co-autores criaram uma teleoperação para sistemas pneumáticos, utilizando um braço robótico 4 graus de liberdade com garra para pegar objetos. Foi usado o Skype para transmissão de vídeo, UDP para comandos, e uma conexão com Internet cabeada. A distância entre os locais entre *master-slave* foi de 30 km (Kato *et al.*, 2010).

3.2.1 Escolha das Ferramentas e Técnicas para o Desenvolvimento da Arquitetura de Teleoperação

Em relação ao estado da arte da teleoperação, observou-se que é possível controlar em tempo real uma plataforma robótica através da Internet. Alguns trabalhos contam com infraestrutura celular, 3G e 4G. No entanto, devido à dependência do sinal, pode não ser a melhor opção para locais fechados, como hospitais. Um método mais apropriado para locais fechados é a conexão com Internet cabeada, que foi aplicada por alguns autores (Shenai *et al.*, 2014; Manzi *et al.*, 2016). Porém, para a comunicação de curta distância, é necessário adicionar dispositivos extras de comunicação sem fio.

Considerando o *feedback* de *streaming* de vídeo do *slave* para o *master*, observou-se que os autores aplicaram *codecs* com o intuito de reduzir o *throughput*. Nesta pesquisa, optou-se por aplicar esse método, de modo a diminuir os recursos de largura de banda necessários para o treinamento. Para isso, foi utilizada a biblioteca ffmpeg, que permite realizar o *streaming* de vídeo e o codec, além de ser uma ferramenta de código aberto.

Os autores usaram diferentes protocolos de rede, como UDP, TCP, RTP e HTTP. Este trabalho aplicará o UDP para os comandos, pois esse não requer confirmação de mensagem e consequentemente fornece uma resposta mais rápida. Para o *streaming* de vídeo, foi escolhido o HTTP, pois facilita o processo de sincronização de solicitação e resposta entre o servidor e cliente. Para avaliar a comunicação, alguns autores aplicaram o RTT como parâmetro e o *software* Wireshark para a análise do tráfego de rede. Este trabalho usará os dois métodos, pois o RTT facilita a métrica de referência com trabalhos relacionados. Outro fator relevante é o Wireshark, um *software* aberto que facilita a aplicação nesta proposta.

Geralmente, os trabalhos encontrados na literatura utilizam um *smartphone* ou computador *desktop* no *slave* para receber dados do *master*. Já que um dos objetivos desta pesquisa é desenvolver um sistema de baixo custo que possa ser mais facilmente aplicado aos

países em desenvolvimento, este trabalho aplica um sistema embarcado de baixo custo para realizar esse papel.

3.3 Contribuição Esperada desta Pesquisa

A contribuição esperada é um ambiente de treinamento para novos usuários de CRM realista, que não tenha o problema de representação do comportamento cinemático, que é encontrado em trabalhos relacionados, permitindo ao usuário uma experiência segura e realista de treinamento, que possibilite a prática de forma segura. O ambiente esperado permitirá aos usuários praticar à distância, mesmo que esses estejam localizados em cidades diferentes de onde se encontra a CRM. Logo, essa característica facilita a replicação do ambiente proposto, pois com uma infraestrutura de treinamento contendo a CRM, usuários poderão praticar à distância a partir de hospitais e centros de reabilitação, tendo como requisito um computador com conexão à Internet e pelo menos um dos métodos de condução.

Em relação os métodos de condução, o ambiente esperado permitirá não só o treinamento a partir do *joystick*, mas através de novas tecnologias, como, por exemplo, o rastreamento ocular. Desta forma, espera-se incluir pessoas com deficiência severa como possíveis usuários para o treinamento. Ademais, no ambiente proposto será implementado um método genérico para condução, onde o objetivo é criar um modelo flexível para servir como base para inclusão de novos métodos de condução; esse método se chamará IHMG (interface humano-máquina genérica). Por exemplo, para incluir outros métodos, poderá ser utilizado toda a arquitetura da IHMG.

A partir do estudo e identificação das vantagens e desvantagens dos trabalhos relacionados, a Tabela 3 apresenta as características pretendidas nesta pesquisa.

Tabela 3 - Características pretendidas para esta pesquisa.

Esta proposta		
Característica		**Justificativa**
Tipo de Ambiente de Treinamento	Teleoperação	Treinamento realista, onde não há problema de representação do comportamento cinemático
Método de condução	Joystick (I), Eye-tracker (II), IHMG (III)	Abranger usuários de diferentes níveis de deficiência (I e II). Método genérico que facilite inclusão de novas formas de condução (III).
Feedback visual	Monitor	Facilidade na utilização e não causa sensação *cybersickness* (enjoo).
Critério para avaliar o desempenho do usuário	Usuário irá cumprir percursos pré-estabelecidos, onde serão medidos o tempo de cumprimento, bem como a quantidade de comandos utilizados.	Comparar performances de treinamento para os diferentes métodos de condução e verificar o mais adequado para cada usuário
Infraestrutura de comunicação pela Internet	Banda larga cabeada	Adequada para ambientes *indoor* (hospitais e clínicas de reabilitação)
Streaming de vídeo e codec	Ffmpeg	Ferramenta de código aberto (*open source*), que permite utilizar codecs para diminuir *throughput* e então reduzir os requisitos de largura de banda
Protocolos de Rede	UDP (I) / HTTP (II)	Curto tempo de resposta (I) e facilidade de sincronização entre cliente servidor (II)
Métrica para avaliar a comunicação	RTT	Será utilizado para medir o atraso (*delay*) de teleoperação entre o envio de comandos e recebimento de *feedback* visual
Software para análise do tráfego de rede	Wireshark	Ferramenta de código aberto (*open source*). Permite filtrar pacotes e calcular parâmetros como *throughput*
Hardware (função de computador) do lado *slave*	Sistema Embarcado	Baixo custo de implementação. Facilidade para replicar o ambiente em diferentes cidades

3.4 Conclusão do capítulo

Neste capítulo foi realizado um estudo sobre os trabalhos relacionados a ambientes de treinamento para usuários de CRM, e sobre sistemas baseados em teleoperação. Foram investigados os tipos de ambientes de treinamento que estão sendo utilizados, os métodos de condução, com também quais dificuldades quanto ao estado da arte desses ambientes.

Em relação à teleoperação, foram investigadas técnicas apropriadas para atender aos requisitos da arquitetura do ambiente proposto, como por exemplo, permitir a teleoperação a longa distância com baixo custo de implementação, e fazer a escolha de uma tecnologia que fosse adequada a locais fechados (*indoor*).

Deste modo, este capítulo teve a importância de apresentar o *background* do tema abordado e o *gap* da pesquisa científica. Da mesma forma, foi apresentada a contribuição esperada, além das características e justificativas para a escolha das tecnologias utilizadas como base para a realização desta pesquisa.

4. Arquitetura e Implementação do Ambiente de Treinamento Proposto

Neste capítulo são apresentadas as partes do ambiente de treinamento proposto, no que se refere ao local do usuário em treinamento, o local da CRM, e os métodos desenvolvidos para permitir a comunicação entre ambos. São apresentados os conceitos de cada uma dessas partes detalhadamente, descrevendo o funcionamento, ferramentas utilizadas, métodos e materiais aplicados.

Desse modo, o capítulo está organizado da seguinte forma: primeiramente na seção 4.1 é apresentada uma visão geral do ambiente. Em seguida, na seção 4.2 são explicadas as técnicas utilizadas para o local de treinamento do usuário. A seção 4.3 aborda o local da CRM, apresentado os materiais e técnicas aplicadas na implementação, assim como é abordado o modelo de comunicação implementado. Na seção 4.4 são apresentadas as técnicas para avaliação do desempenho do usuário em treinamento. Finalmente, na seção 4.5 são apresentadas as configurações e fluxogramas do *software* implementado.

4.1 Visão Geral do Ambiente

O ambiente proposto possui duas grandes partes. A primeira se trata do Centro de Treinamento (CT), onde ficam localizados todos os acessórios e dispositivos necessários para que os usuários realizem o treinamento. Os usuários poderão frequentar o CT para aprender a comandar a CRM, e contarão com múltiplas opções de métodos para condução. A explicação detalhada sobre o CT é descrita ao longo deste capítulo.

A segunda parte do ambiente proposto é a Sala de Teleoperação (ST), onde fica localizada a CRM e toda a infraestrutura necessária para permitir a teleoperação. Na ST todos os comandos serão recebidos e executados, e o vídeo do local será transmitido em tempo real para o CT. A descrição detalhada sobre a ST também será apresentada no decorrer do capítulo.

Nesta pesquisa, foram criados CTs em diferentes cidades do País, são elas: Manaus, AM; Uberlândia, MG e Campinas, SP. Essas cidades foram escolhidas para permitir a representação de diferentes configurações de treinamento, além disso, permitir a avaliação do ambiente proposto no quesito de teleoperação a longas distâncias. A distância linear entre

Manaus e Uberlândia é de 2172 km, no caso de Manaus e Campinas é de 2596 km. A Figura 8 apresenta o diagrama conceitual da proposta com CT e a ST.

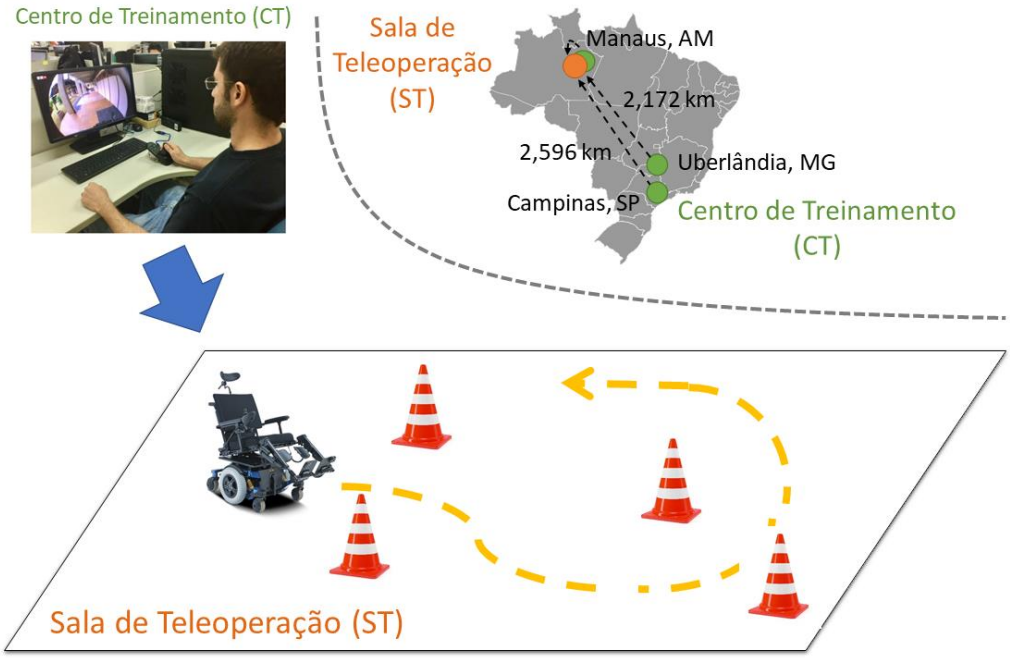


Figura 8 - Diagrama conceitual do sistema de treinamento em teleoperação.

O funcionamento do sistema consiste na constante comunicação entre CT e ST. Enquanto o usuário utiliza os métodos de condução (também chamados de interfaces de controle), são gerados comandos remotos a serem enviados à ST. Ao alcançarem o destino, esses comandos são traduzidos de mensagens computacionais para sinais elétricos. Esses sinais são responsáveis por movimentarem a CRM, a qual foi adaptada com um sistema embarcado conectado diretamente ao circuito de interface com os motores (driver dos motores), conforme apresentado na Figura 9.

Paralelamente, na ST, duas câmeras gravam e transmitem o vídeo da CRM em tempo real (utilizando a técnica de vídeo *streaming*). Esse vídeo é reproduzido no CT para que o usuário possa receber em tempo real um *feedback* visual dos seus comandos, permitindo então que a CRM seja teleoperada apropriadamente. Esse conceito também é apresentado na Figura 9.

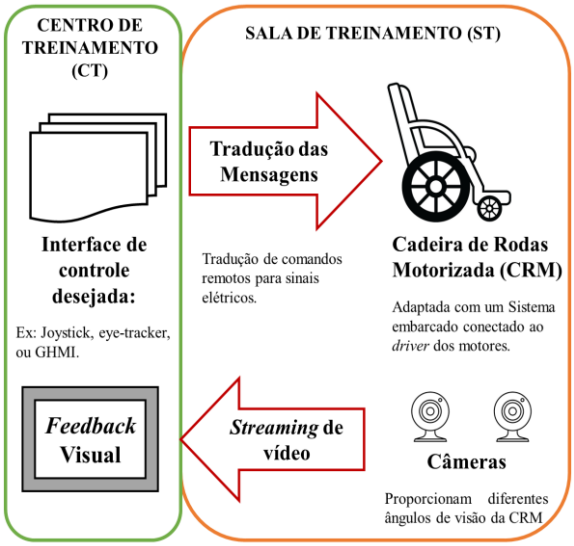


Figura 9 - Visão geral de como o sistema opera.

4.2 Centro de Treinamento (CT)

O Centro de Treinamento (CT) consiste em um local onde os novos usuários podem frequentar para aprender a comandar a CRM e melhorar suas habilidades, esses usuários contam como opção diferentes métodos de condução. Os métodos podem ser escolhidos de acordo com as limitações de movimentos do usuário, podendo ser usuários com deficiência nos membros inferiores, como também usuários com deficiência severa, os quais possuem limitações dos movimentos das mãos. Logo, existem três tipos de interfaces de controle: *joystick*, *eye-tracker* e interface humano-máquina genérica (IHMG), que serão descritas ao longo desta seção. O requisito de *hardware* do CT é ter um computador conectado a um *gateway* com uma *Internet* de banda larga cabeada, e pelo menos uma das interfaces de controle conectadas ao computador. Os detalhes sobre as configurações da Internet e do funcionamento das interfaces de controle serão apresentados nas próximas seções.

4.2.1 Método de Condução por *Joystick*

O primeiro método de condução a ser apresentado é o *joystick*, o qual representa a maneira convencional de dirigir uma CRM. Para desenvolver este método, testes e laboratório foram realizados com o intuito de verificar os níveis de tensão gerados pelo *joystick* original da CRM, modelo *Freedom Carbon,* disponibilizado pela Universidade Federal do Amazonas (Freedom, 2018). Observou-se que o sistema de comando consiste em um vetor de magnitude e ângulo gerado a partir do movimento do *joystick*, conforme apresentado na Figura 10. A

magnitude corresponde à velocidade, e o ângulo corresponde à direção. Matematicamente, de forma simplificada, o comportamento é interpretado de acordo com as equações 1 e 2 .

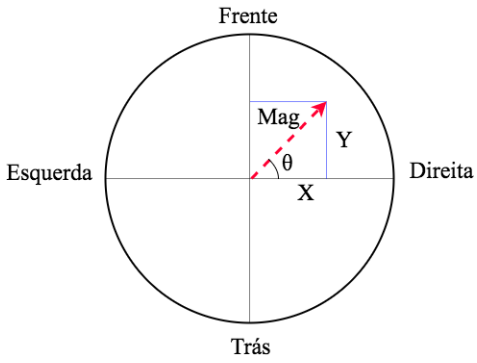


Figura 10 - Representação matemática do comportamento do *joystick*.

$$\boldsymbol{Mag = \sqrt{X^2 + Y^2}} \quad (1)$$

$$\boldsymbol{\theta = \tan^{-1}\left(\frac{Y}{X}\right)} \quad (2)$$

- Mag: Magnitude (intensidade da velocidade) escolhida pelo usuário;
- θ: Ângulo (direção) escolhida pelo usuário, medida a partir do eixo X (lado positivo) até o ponto de interesse.

A implementação desta interface de controle foi feita a partir do *joystick* original da CRM conectado a um sistema embarcado (Arduino Uno), conforme apresentado na Figura 11 (Arduino, 2018). Esse sistema embarcado recebe como entrada os sinais analógicos de tensão utilizando dois ADCs (*Analog to Digital Converters*) com 8 bits de resolução cada, sendo destinado aos sinais do eixo X e outro ao eixo Y. Em seguida as mensagens na forma digital são enviadas do Arduino Uno para o computador através do protocolo UART por meio da porta USB, com uma taxa de transmissão de 9600 bps. Essas mensagens de comando são transmitidas a uma frequência de 13,33 Hz.

Figura 11 - *Joystick* original da CRM adaptado para teleoperar à distância.

4.2.2 Interface de Controle por *Eye-Tracker*

Esta interface de controle foi projetada com o intuito de atender pessoas com deficiências severas, ou seja, aquelas pessoas que não podem comandar a cadeira de rodas pelo método convencional (*joystick*). Com a interface de controle por *eye-tracker*, os usuários podem comandar usando movimentos dos olhos. Para criar essa interface, é utilizado o dispositivo Tobii 4c, o qual é um *eye-tracker* que possui uma API (*Application Program Interface*) para facilitar o desenvolvimento do *software (Tobii, 2018)*. Neste método de condução, o usuário olha em direção a uma das setas referente ao comando desejado (frente, trás, esquerda, direita e parada), sendo essas setas projetadas em um monitor de computador.

Durante a teleoperação o *software* desenvolvido captura a posição do olhar do usuário no monitor do computador. Para evitar comandos não intencionais, esses são gerados somente quando o usuário foca em uma das setas por, pelo menos, 500 milissegundos. Esse valor foi escolhido de forma empírica através de diversos testes realizados durante a fase de implantação do *software.*

Foi construído um mapa de pixels com um intervalo específico de posição para cada comando. Este mapa é uma imagem com a mesma resolução da tela do computador *full* HD (1920 x 1080). A Figura 12 mostra essa interface e os valores referentes às posições de pixels das linhas (X) e colunas (Y).

Quando um comando é gerado, o computador reproduz um som de confirmação. Por exemplo, se o usuário olhar para a seta para frente, o computador reproduzirá o som "*forward*", que significa "para frente" em inglês. Além do mais, o mesmo acontece para as outras direções. As posições das setas foram baseadas na técnica de rastreamento ocular de Ktena, chamada *screen-based*, que apresentou melhor desempenho em relação a outras técnicas de rastreamento ocular (Ktena *et al.*, 2015).

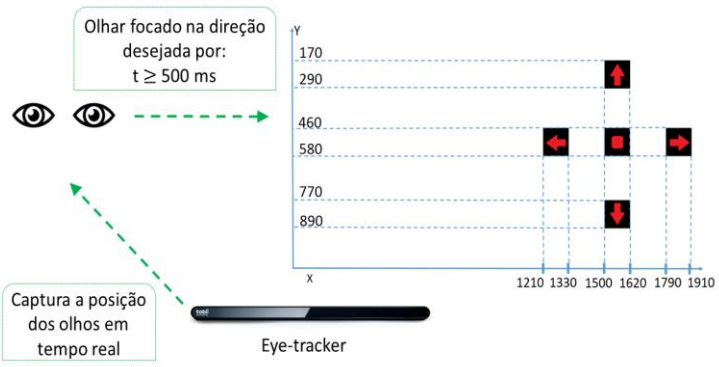


Figura 12 - Interface de rastreamento ocular com as faixas de posições de pixel para gerar comandos para CRM.

4.2.3 Interface Humano-Máquina Genérica

A interface humano-máquina genérica (IHMG) consiste em uma interface gráfica do usuário (GUI) com comandos discretos representando os movimentos da cadeira de rodas. O usuário opera selecionando os botões correspondentes à direção desejada. Quando isso ocorre, a CRM se move continuamente, mas o usuário pode parar selecionando o botão quadrado (no centro das setas). A Figura 13 apresenta essa interface.

Para o reaproveitamento da arquitetura desenvolvida, futuros desenvolvedores não terão que se preocupar com toda a complexidade de criar a comunicação entre *master-slave.* Por exemplo, não será necessário converter as mensagens digitais em sinais elétricos para mover a CRM, ou em estabelecer um *streaming* de vídeo para *feedback*, além de outros aspectos. Por exemplo, se for desejado incluir sinais musculares como um método de condução, é possível incluí-los reutilizando a IHMG. O requisito é simplesmente importar o *software* implementado como a estrutura principal do código e seguir o padrão de comandos, o qual foi apresentado na Tabela 3. Esse aspecto torna essa uma das interfaces de controle mais importantes desta pesquisa.

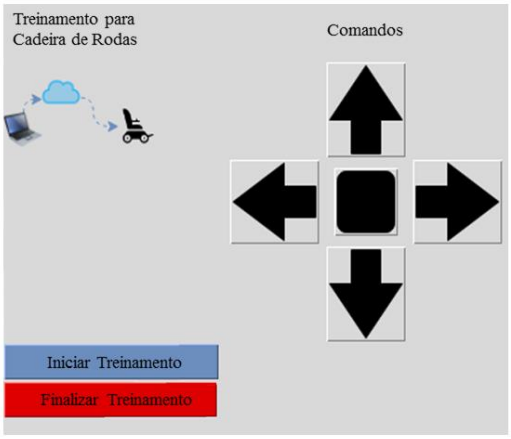


Figura 13 - Interface humano-máquina genérica (IHMG) que pode ser facilmente adaptada para novos métodos de condução.

4.3 Sala de Teleoperação (ST)

A Sala de Teleoperação (ST) consiste no local onde a CRM ficará localizada, juntamente com toda a infraestrutura necessária para suportar acesso remoto do usuário em treinamento. Nesse local, são traduzidas as mensagens recebidas pelo usuário para sinais elétricos responsáveis pela movimentação da CRM, cujo padrão de mensagens é apresentado nesta seção. Também são apresentadas as ferramentas e configurações aplicadas para o vídeo *streaming* transmitido da ST para o CT. Outrossim, é descrita a arquitetura de comunicação

entre a ST e CT, onde são abordados quesitos como *throughput*, protocolos de rede e finalmente uma visão completa sobre o *hardware* do ambiente implementado.

4.3.1 Sinais Elétricos para Movimentação da CRM

Com o intuito de comandar a CRM a partir de mensagens computacionais, foi implementado um driver responsável por gerar níveis de tensão analógicos baseados no comando gerado pelo usuário. Para isso foi criado um protocolo de mensagens de comando que funciona da seguinte forma: cada eixo pode variar entre –100% e 100%, os quais representam os valores mínimos e máximo, respectivamente. Caso os valores de ambos os eixos X e Y sejam 0 (zero)%, significa a posição neutra (CRM parada).

A partir desse protocolo são gerados níveis de tensão analógica que variam entre 1,5V e 3,5V, no qual –100% equivale ao menor valor, e 100% ao maior, e no caso de 0% é gerado 2,5V. Partindo desse protocolo e do conceito aplicado nas equações (1) e (2) apresentadas na subseção 4.4.1, foi criada a Tabela 4, a qual apresenta exemplos de comandos de movimentação da CRM.

Para permitir a implementação dessa ferramenta, existe um sistema embarcado com dois Conversores Digital-para-Analógico, ou do inglês *(Digital to Analog Converters, DACs)*, um para o eixo X e outro para Y, no qual cada DAC tem resolução de 8 bits. O sistema embarcado aplicado na CRM é a placa UDOO-Dual (Udoo, 2018). Esta placa vem com um processador ARM, um adaptador de rede WiFi e um Arduino DUE integrado (usado para os DACs). O processador ARM se comunica com a placa Arduino DUE através do protocolo serial.

Tabela 4 - Exemplos de mensagens de comandos e tensão analógica enviada ao driver da CRM.

Comando	Coordenada Polar		Tensão Gerada (V)		Movimento da CRM
	θ	*Mag.*	*X*	*Y*	
X=0%,Y=100%	90°	100	2,5	3,5	Frente
X=0%,Y= -100%	-90°	100	2,5	1,5	Trás
X=100%,Y=0%	0°	100	3,5	2,5	Direita
X= -100%,Y=0%	180°	100	1,5	2,5	Esquerda
X=0%,Y=0%	0°	0	2,5	2,5	Parada

Entretanto, foi necessário aplicar um condicionador de sinal para amplificar o nível de tensão. Isso ocorreu porque o nível máximo de tensão gerado pelo DAC é de 3,3V, mas o nível necessário para movimentar a cadeira de rodas é de 3,5V. A equação 3 mostra o cálculo do ganho do amplificador.

O condicionador de sinal projetado é o circuito eletrônico apresentado na Figura 14, que contém amplificadores operacionais não-inversores baseados no circuito integrado LM324N (Ti, 2018). O ajuste fino da tensão é realizado via *software*. A Figura 15 apresenta os componentes de *hardware* usados para adaptar a CRM para a teleoperação.

$$A_{V\,(\text{não}-inversor)} = 1 + \frac{R2}{R1} = 1 + \frac{1K}{5K} = 1,2 \qquad (3)$$

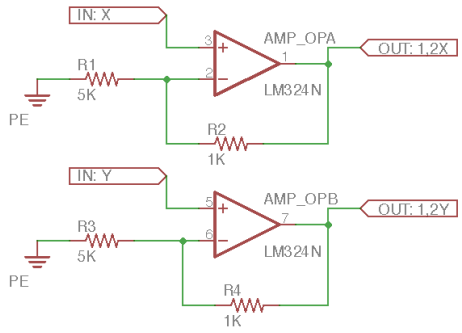


Figura 14 - Circuito de condicionamento de sinal para o driver do motor da CRM.

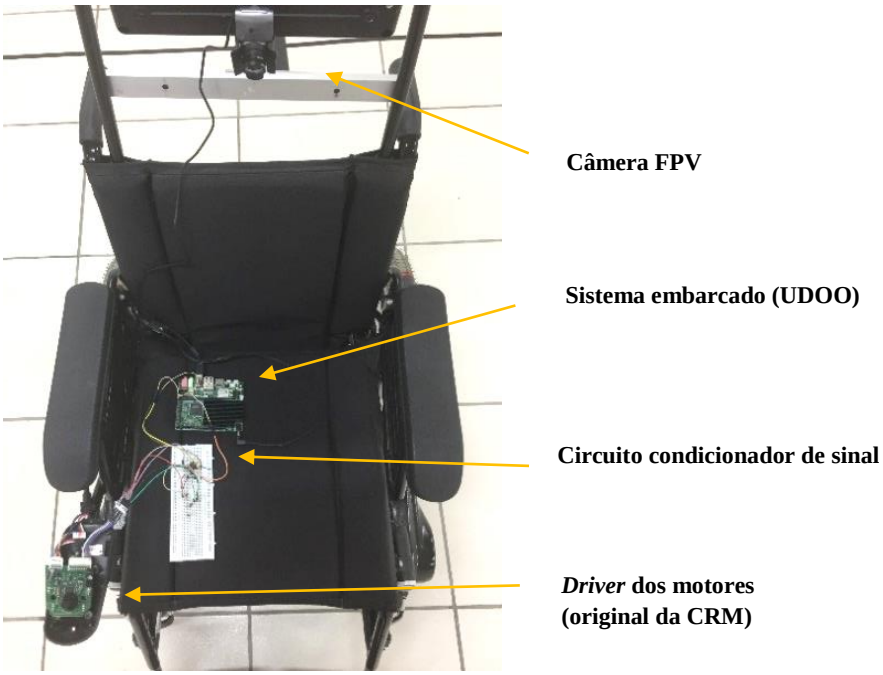


Figura 15 - Componentes de *hardware* para adaptar a CRM para a teleoperação.

4.3.2 *Streaming* de Vídeo

O *streaming* de vídeo fornece ao usuário um *feedback* de vídeo em tempo real da ST através de duas câmeras diferentes. A primeira fornece uma visão em primeira pessoa, também chamada de (*First Person View* - FPV), que representa a visão de uma pessoa sentada na CRM. A segunda fornece uma visão de terceira pessoa (*Third Person View* - TPV), que possibilita ao usuário uma visão de referência do ambiente de treinamento.

A câmera FPV está diretamente conectada ao sistema embarcado (UDOO) através da porta USB. O reconhecimento do dispositivo é feito de forma automática a partir do próprio SO do sistema embarcado. Para realizar o vídeo *streaming*, é compilada a biblioteca ffmpeg diretamente no sistema embarcado, de forma a ser adequada à arquitetura do processador ARMv7. Para isso foram instalados pacotes no Linux embarcado como, por exemplo, *build-essential*, que contêm ferramentas necessárias para compilação.

A câmera TPV é baseada em uma câmera IP (IPCAM), que possui o próprio *software* do fabricante que permite configurar aspectos como resolução, protocolo de transmissão e formato de compressão. As resoluções das câmeras FPV e TPV é de 800 x 600 pixels e 320 x 240 pixels, respectivamente. A taxa de quadros por segundo de ambas é de 20 (fps - *frames per second*). Foi adicionada uma lente olho-de-peixe (*fisheye lens*) para a câmera FPV, com objetivo de fornecer uma visão mais ampla ao usuário. Para reduzir o *throughput* foram utilizados os encoders mpeg4 e mjpeg para as câmeras FPV e TPV, respectivamente.

De modo permitir a transmissão, foram criados dois servidores HTTP, um para cada câmera. Cada um possui uma porta específica, havendo sido escolhida a porta 4444 para a FPV, e 4445 para a TPV. Outro aspecto importante é que, de forma empírica, foi observado que reduzir o *buffer* de vídeo *streaming* diminui significativamente o atraso visualizado na reprodução, logo, foi optado por não utilizar *buffer*. No que se refere à reprodução de vídeo no lado do cliente, é utilizado um reprodutor de mídia portátil (ffplay). Os passos necessários para transmitir e receber o vídeo são apresentados na Figura 16.

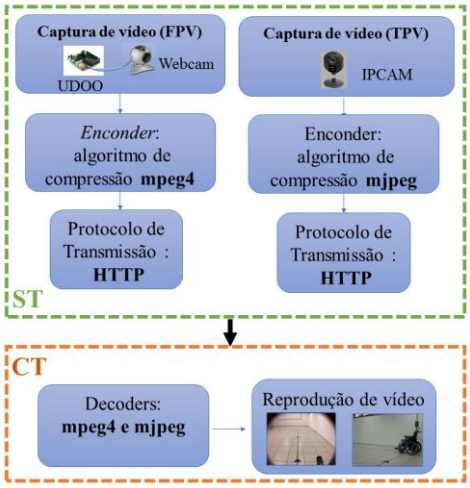


Figura 16 - Transmissão de vídeo *streaming* e processo de reprodução.

4.3.3 Arquitetura de Comunicação

A comunicação entre o CT e a ST é realizada através da Internet, na qual os comandos do usuário em treinamento são enviados a partir de *socket* UDP. O servidor fica localizado na ST, enquanto o cliente no CT. Para o caso do *streaming* de vídeo, foram utilizados dois servidores HTTP, sendo um para cada câmera, os quais ficam localizados na ST. Além disso, existem dois clientes para *streaming* de vídeo, os quais ficam no CT.

No lado da ST foi utilizado um roteador para realizar o redirecionamento de portas. Por exemplo, caso chegue um pacote UDP para o endereço público do roteador com a porta 4444, esse pacote é redirecionado para o IP local do sistema embarcado através da conexão WiFi, o qual possui um servidor socket UDP aguardando mensagens nessa porta. De forma similar acontece para as câmeras; em caso de requisição de acesso HTTP ao IP público do roteador, com a porta 4445, essa comunicação é redirecionada ao servidor HTTP presente no sistema embarcado, responsável pelo vídeo *streaming* da câmera FPV. Por último, caso haja uma requisição à porta 4444 com protocolo HTTP, esse acesso é redirecionado à câmera IP, a qual faz o vídeo *streaming* TPV. A Figura 17 apresenta a arquitetura de comunicação do sistema e o fluxo de dados.

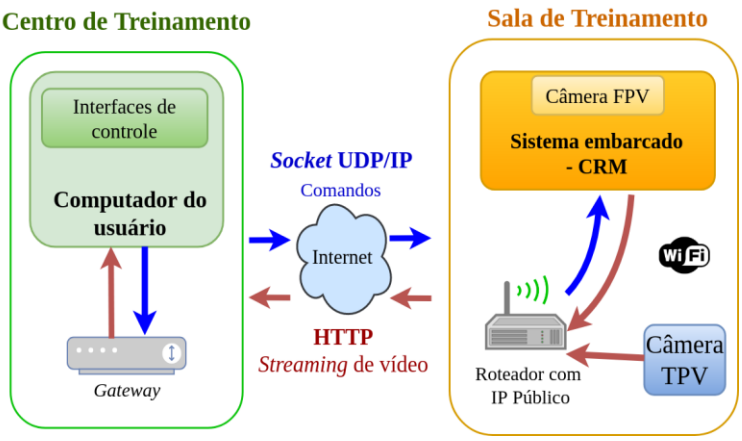


Figura 17 - Arquitetura de comunicação e fluxo de dados para a teleoperação.

Para fim de análise do tráfego de rede, foi monitorado o fluxo de mensagens de comando através do *software* Wireshark, e similarmente foi monitorado o *streaming* de vídeo de ambas as câmeras (FPV e TPV). Foi verificado que o *throughput* médio para os pacotes de vídeo é de 865 kbits/s, considerando as duas câmeras. Para o caso das mensagens de comando, foi obtida uma média de 22 kbits/s. Logo, a soma das médias de ambas as câmeras resulta em 887 kbits/s.

A Figura 18 apresenta o gráfico com os *throughputs* de cada caso individual e também da soma de todos. No eixo Y é apresentada a taxa em bits/segundo, e no eixo X é apresentado o tempo no qual o pacote foi capturado. O tempo total do experimento foi de aproximadamente 5 minutos, possibilitando observar as variações dos tamanhos dos pacotes, e obter referência adequada para calcular as médias. A Tabela 5 apresenta um resumo das médias calculadas.

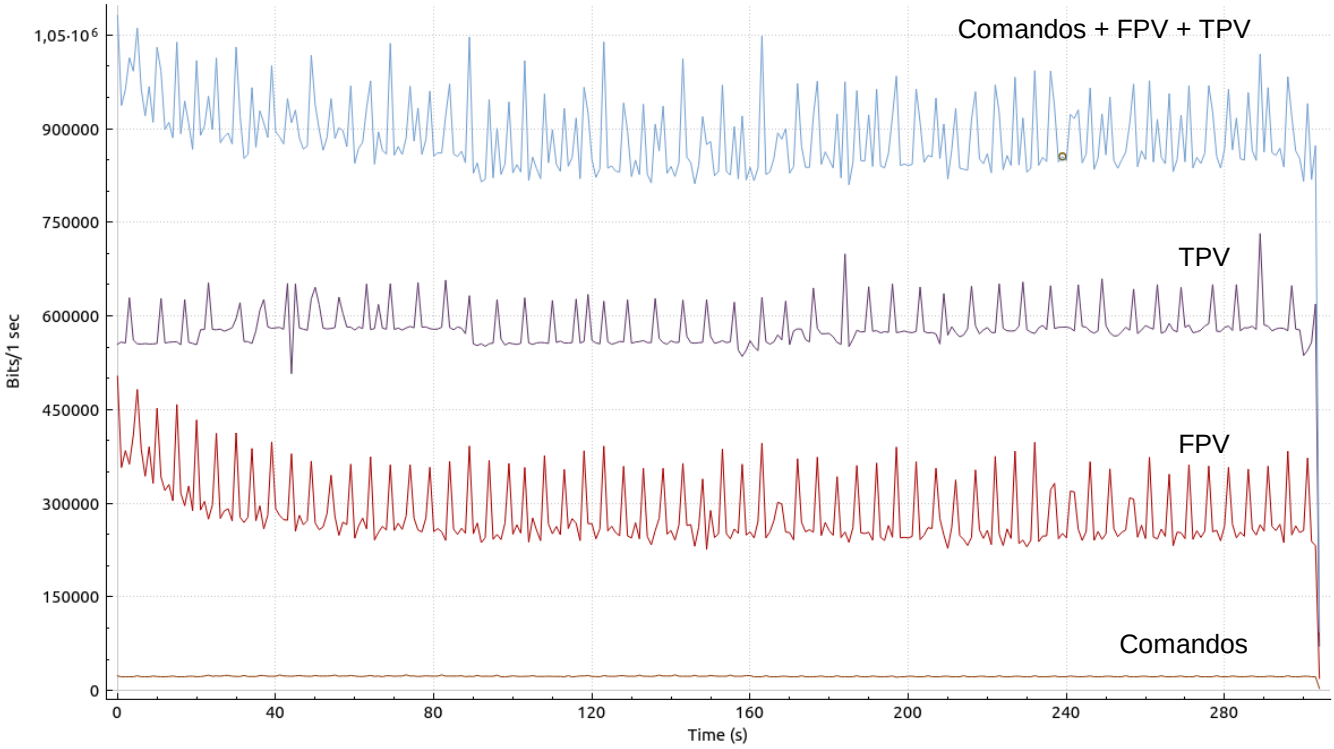


Figura 18 - Gráficos de *throughput* referente às câmeras de *streaming* de vídeo e comandos do usuário.

Tabela 5 - Análise dos dados de *throughput* para o tráfego de rede (CT/ST).

Dados	**Média (kbits / seg.)**
Comandos	22
FPV	284
TPV	581
TOTAL	887

Para as fases de validação, a qualidade da comunicação entre o CT e a ST é avaliada enquanto o treinamento está ocorrendo. O parâmetro utilizado nessa avaliação é o RTT, que consiste no tempo de resposta de ida e volta de um pacote. Em cada segundo do treinamento, esse parâmetro é medido e salvo em um arquivo de log. Em seguida, os resultados são analisados estatisticamente para calcular valores médios, desvio padrão, máximos e mínimos. Essa análise será apresentada na próxima seção, sendo importante mencionar que a avaliação de comunicação foi realizada para a validação do sistema, embora, para uso futuro, isso não seja mais necessário.

A Figura 19 mostra uma visão completa dos componentes de *hardware* do ambiente proposto.

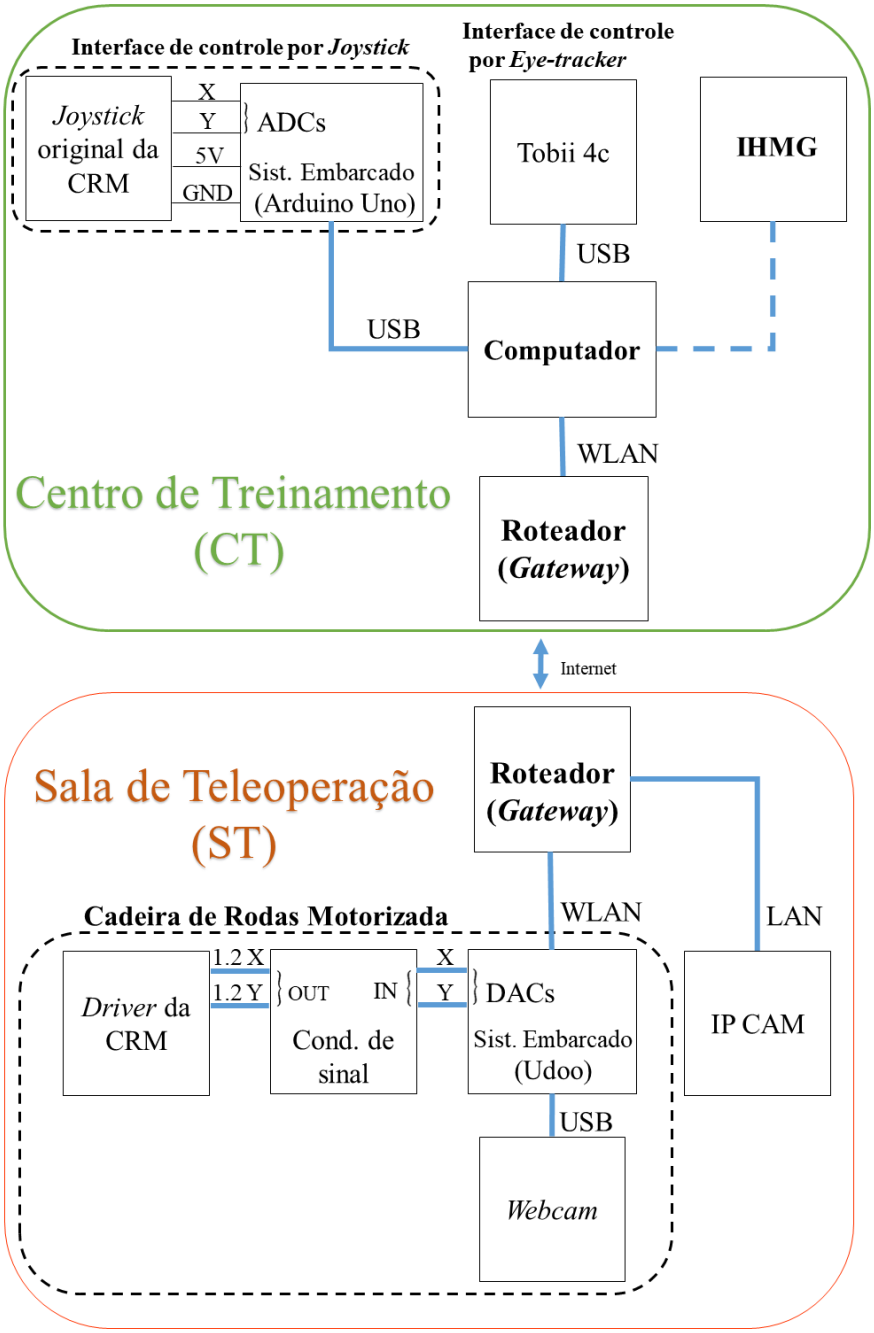


Figura 19 - Visualização completa dos componentes de *hardware* do sistema de treinamento em teleoperação.

4.4 Avaliação do Desempenho do Usuário em Treinamento

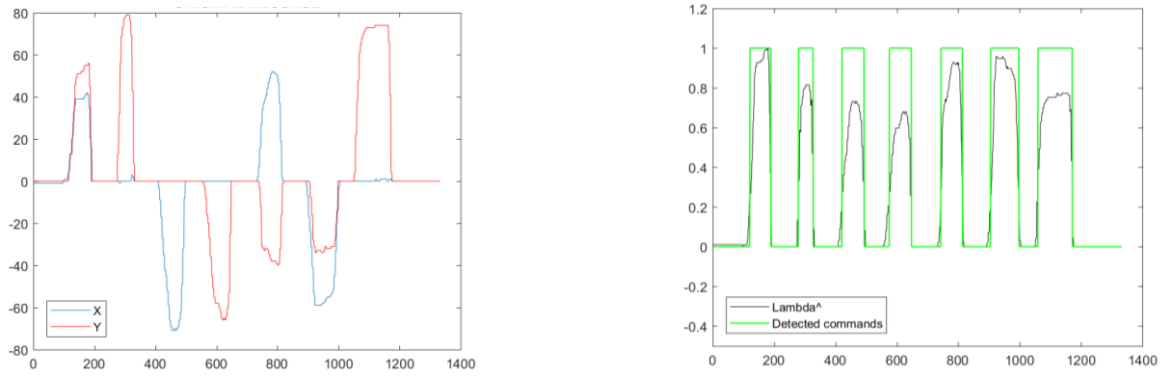
Para determinar como foi o desempenho do usuário durante o treinamento, duas variáveis de interesse são medidas. A primeira é o tempo utilizado para terminar o percurso pré-estabelecido, e a segunda é o número de comandos gerados durante esse percurso. O tempo de conclusão dos percursos é obtido automaticamente pela diferença de marcações de tempo (*timestamps*) do início e fim. Para obter automaticamente o número de comandos, os dados X e Y das mensagens do usuário são registrados em um arquivo de log. Para isso, foi criado um algoritmo no MATLAB. As Equações 4 e 5 apresentam os primeiros procedimentos de cálculo. A variável λ representa a soma dos módulos de X e Y. Em seguida, o resultado é normalizado e se torna $\hat{\lambda}$.

$$\boldsymbol{\lambda} = |\boldsymbol{X}| + |\boldsymbol{Y}| \qquad (4)$$

$$\hat{\lambda} = \frac{\lambda}{máx(\lambda)} \qquad (5)$$

A próxima etapa de obtenção do número de comandos é aplicar a função de limiar do MATLAB em $\hat{\lambda}$. Essa função transforma a saída em binário, onde são considerados como "1" os valores maiores que o limiar, e "0" os menores. O valor apropriado para o limiar foi escolhido após vários testes. Observou-se que quanto menor o valor, maior a sensibilidade na detecção de comandos. Um exemplo de dados coletados de um comando do *joystick* é mostrado na Figura 20 (a). O gráfico mostra a amplitude de X e Y com o número da amostra. O número de comandos é contado automaticamente usando a função de número de regiões do MATLAB, que verifica o número de regiões para os valores iguais a "1". A Figura 20 (b) apresenta a aplicação do algoritmo de detecção de comandos. O *Script* 5 apresenta a implementação do algoritmo a partir do MATLAB.

O objetivo da contagem de comandos é comparar o desempenho do usuário ao longo das sessões de treinamento. Por exemplo, pela primeira vez realizando um percurso, um usuário precisou de 30 comandos para completar, mas depois de praticar, o número de comandos foi reduzido para 21. Esse exemplo mostrou uma melhoria de 30% do desempenho em comparação com os valores iniciais. O algoritmo de contagem do comando também pode ajudar a determinar qual interface de controle é mais apropriada para o novo usuário comandar a CRM. O usuário pode testar diferentes interfaces de controle e escolher a mais eficiente (menor número de comandos).



(a) **(b)**

Figura 20 – (a) Exemplo de dados de uma mensagem gerada com o *joystick*. (b) Gráfico do algoritmo de detecção de comandos.

Script 5 – Algoritmo de detecção de comandos

```
clear
clc
close all

%% leitura dos comandos
all_data = importdata('C:\Users\AmbIntLab\Dropbox\Mestrado\Dissertacao\analise_comandos-
rtt\experiments\hmi\Uberlandia-Manaus\Commands_hmi_Uberlandia-Manaus\commands (1).dat',' ');
dados = all_data.data;

X = all_data.textdata(:,4);
Y = all_data.textdata(:,5);

% conversao cell para num
X = all_data.textdata(:,4);
[M N] = size(X);
for i = 1:M
    X_int(i,1)=str2num((cell2mat(X(i))));
end
Y = all_data.textdata(:,5);
[M N] = size(Y);
for i = 1:M
    Y_int(i,1)=str2num((cell2mat(Y(i))));
end
X = X_int;
Y = Y_int;

%% duracao do treinamento
ano = all_data.data(:,2);
mes = all_data.data(:,3);
dia = all_data.data(:,4);
hora = all_data.data(:,5);
minutos = all_data.data(:,6);
segundos = all_data.data(:,7);
data = datetime(ano,mes,dia,hora,minutos,segundos);
duracao_treinamento = data(end) - data(1)

%% PRODUTO UNITARIO DE |X| e |Y|
U_prod = abs(X)+abs(Y);
Unit_U_prod = U_prod/max(U_prod);

%% LIMIARIZACAO E CONTAGEM DE COMANDOS
thresh_max = 0.25;
U_prod_limiar = (Unit_U_prod >= thresh_max);
%funcao para contar as regioes
[labeledData, numberOfRegions] = bwlabel(U_prod_limiar);
numero_de_comandos = numberOfRegions
```

4.5 Configurações de *Software*

Para possibilitar o funcionamento dos *softwares* do ambiente proposto, são necessárias algumas configurações prévias nos dispositivos, como, por exemplo, a instalação de pacotes no SO, bem como a configuração de redirecionamento de portas no roteador, e o suporte para determinadas linguagens de programação. Para apresentar os detalhes sobre essas configurações foi criada a Tabela 6.

Tabela 6 – Configurações de necessárias para execução dos *softwares* implementados.

Dispositivo	Sistema Operacional (SO)	Requisitos de *Software*	Suporte de Linguagem de programação
IHMG e *joystick*	Ubuntu 16.04 (Ubuntu, 2018a)	**Pacotes Python**: *pip, python-tk, socket, subprocess, threading, time, datetime, time, pyserial e pillow* *(Pillow, 2018; Pyserial, 2018; Python, 2018b; Tkinter, 2018)* **Pacotes Linux:** ffmpeg	Python 2.7
Interface por rastreamento ocular	Windows 7	**Pacotes Python**: *pip, python-tk, socket, subprocess, threading, time, datetime, time, pyserial e pillow* **Pacotes Windows:** API Tobii 4c instalada (Windows)	Python 2.7 C# (.NET)
Sistema embarcado (UDOO)	UDOObuntu (Udoobuntu, 2018)	**Pacotes Python:** *pip, socket, sys, regex, subprocess, threading, time, datetime* e *pyserial* (Regex, 2018) **Pacotes Linux:** *build-essential (Ubuntu, 2018b)* **Compilação**: Source code da biblioteca ffmpeg,	Python 2.7 C
Roteador	-	**IP público** e acesso liberado às portas: 4444, 4445 **Redirecionamento de pacotes:** Portas: 4444 / UDP (comandos) => IP UDOO 4444 / HTTP (FPV) => IP UDOO 4445 / HTTP (TPV) => IPCAM **Desbloqueio firewall**: ICMP (ping) => IP Público	-
IP CAM	-	**Modelo** Linksys WVC54GCA (Linksys, 2018) **Configuração** de servidor HTTP para vídeo *streaming* com portas 4445 e codec mjpeg	-

4.5.1 *Softwares* Implementados

Nesta subseção serão apresentados os algoritmos do ambiente proposto, os quais foram separados em interfaces de controle e algoritmos utilizados no sistema embarcado da CRM. A ideia de apresentar o fluxograma é demonstrar o funcionamento e detalhar como ocorre o fluxo de informações entre a geração de comandos, reprodução de vídeo em tempo real, até a etapa de movimentação da CRM. O conceito apresentado é a base de todo *software* implementado nesta pesquisa referente às funcionalidades de teleoperação.

Nos algoritmos apresentados, foram utilizados conceitos previamente introduzidos, como por exemplo POO, *threads*, cliente e servidor *socket UDP*, vídeo *streaming* HTTP, RTT, formato padrão de envio de mensagens (Tabela 4), mapa de pixels do rastreamento ocular (Figura 12), ADC, DAC, entre outros.

Os fluxogramas são apresentados nas Figuras 21 e 22, os quais foram separados em clientes: IHMG, interface por rastreamento ocular, interface por joystick. Primeiro são apresentadas as características distintas de cada interface e logo é apresentada a parte comum a todas as interfaces de controle. Conseguinte, também é apresentado o fluxograma do servidor (UDOO), no qual primeiramente é apresentada a etapa de tratamento de mensagens em alto nível e, em seguida, é demonstrado o funcionamento do baixo nível, no qual ocorre a conversão das mensagens para níveis de tensão analógico para movimentação da CRM.

No que se refere à última etapa do fluxograma, é apresentado um exemplo de conversão em nível analógico, o qual segue o princípio apresentado na Tabela 4.

Ex: Caso o valor de entrada de um dos eixos X ou Y seja máximo, ou seja, valor igual a 100, ocorre o seguinte:

$$Valor\_entrada = 100 \qquad Valor\_saída = \frac{100\,.225}{100} = 225$$

Sabe-se que a tensão máxima da UDOO é 3.3V e equivale ao valor 255 no DAC.

$$\text{Logo, 225 equivale a: } \frac{225\,.3{,}3}{255} = 2{,}9117\ V$$

Como foi aplicado um circuito condicionador de sinal, esse valor é amplificado por um fator 1,2. Portanto:

$$Tensão\_saída = 2{,}9117\,.\,1{,}2 = 3{,}4940 \sim 3{,}5\ V$$

Esta tensão é apropriada para controlar os motores da CRM, considerando o valor máximo de acionamento, conforme Tabela 4. O mesmo princípio é aplicado para os outros valores de entrada desse sistema, podendo a tensão variar de 1,5V até 3,5V.

Figura 21- Fluxograma cliente e servidor *socket.*

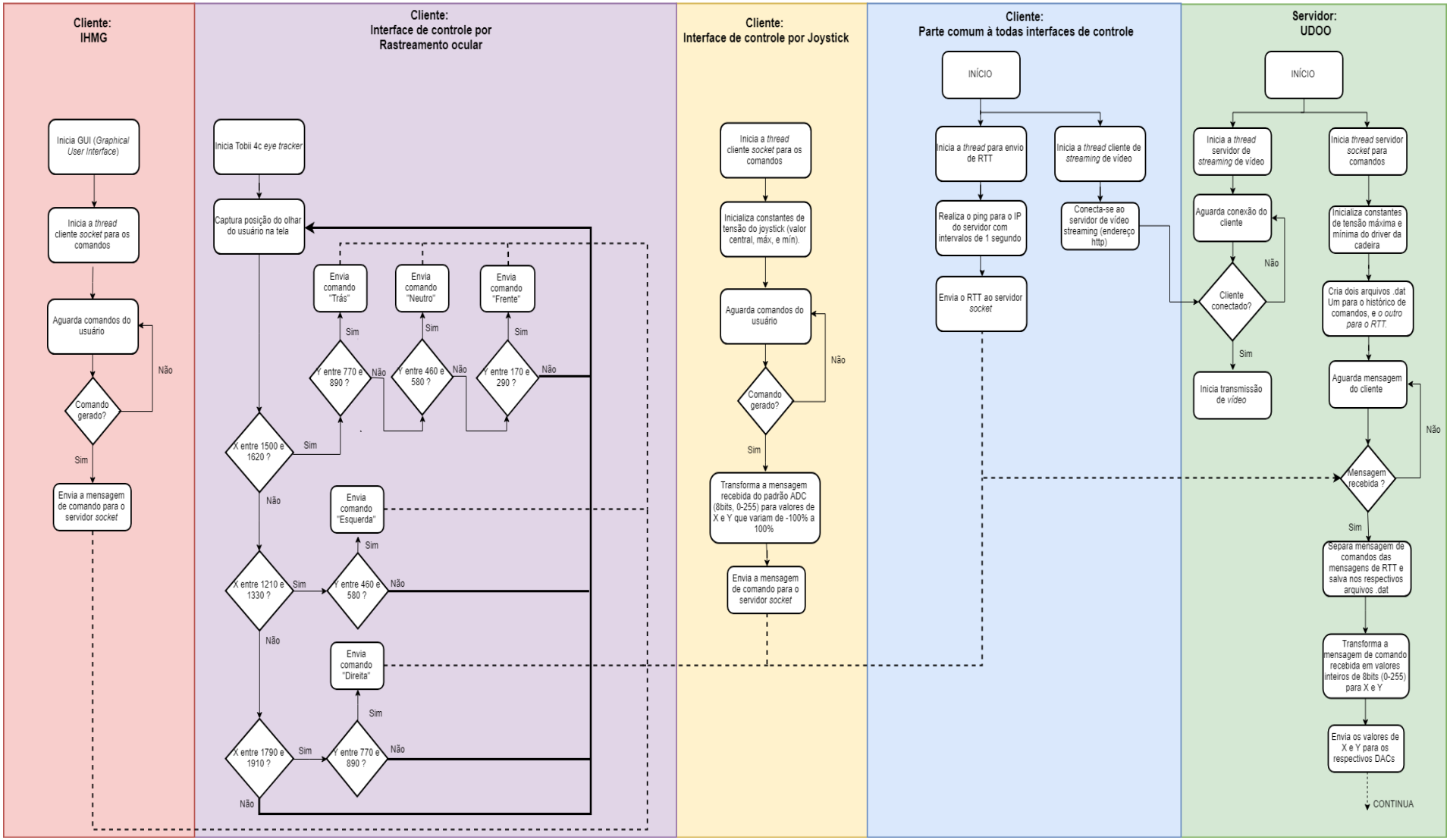
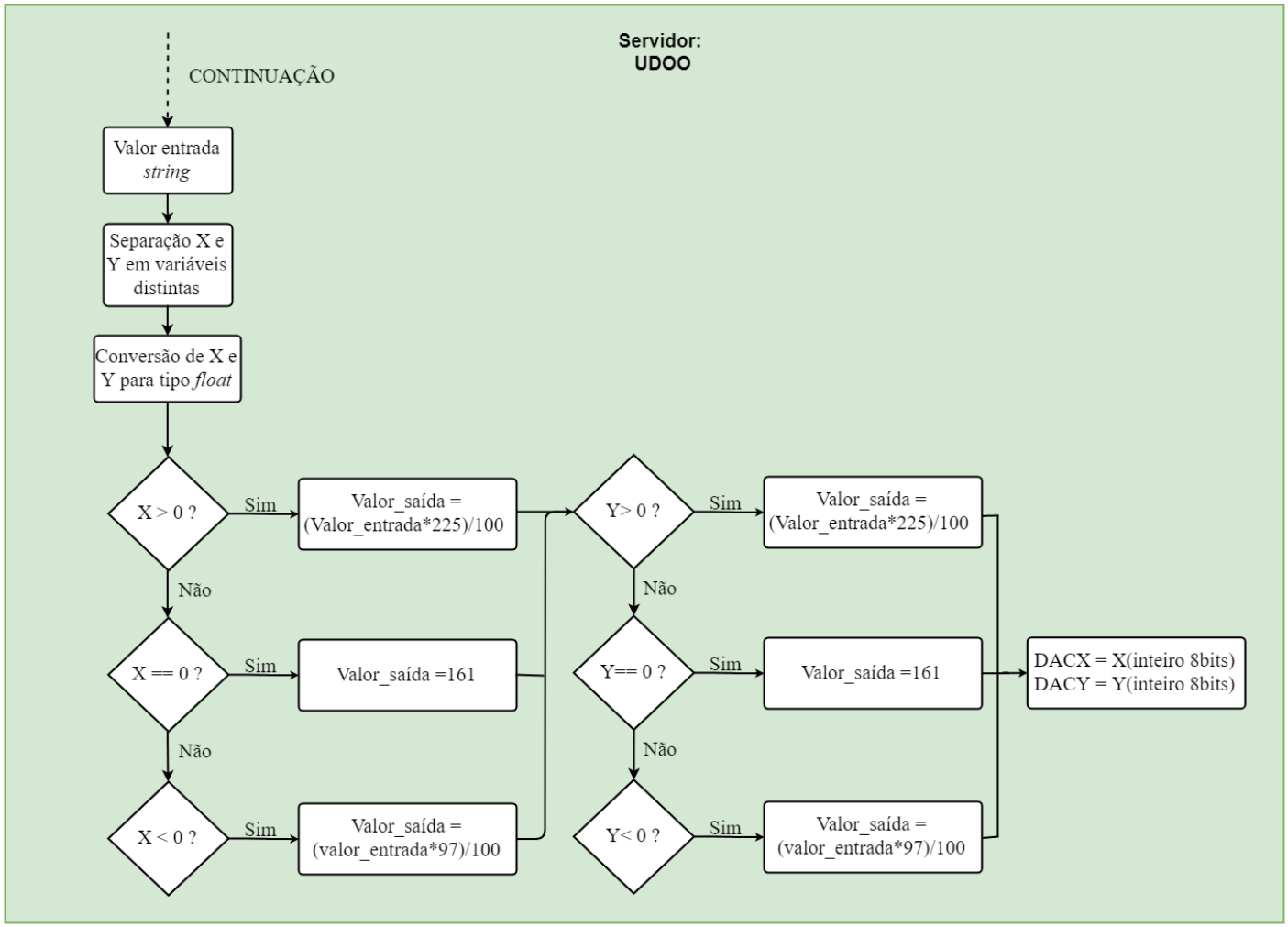


Figura 22 - Fluxograma cliente e servidor *socket: continuação.*



4.6 Conclusão do Capítulo

Este capítulo descreveu o modelo proposto de ambiente de treinamento, conjuntamente com a implementação. Inicialmente foi apresentada uma visão geral sobre o funcionamento, também foi descrita detalhadamente cada etapa. Essas foram divididas em técnicas aplicadas, arquitetura de comunicação, métodos de condução, sinais elétricos para operação da CRM, protocolo de mensagens desenvolvido, análise do desempenho do treinamento, e finalmente, configurações e fluxogramas dos softwares implementados.

Os resultados alcançados neste capítulo serão utilizados na etapa de experimentação que será abordada no Capítulo seguinte, no qual são exibidos os diferentes cenários propostos para validação, como também os resultados e a interpretação desses. Por último, é apresentada a contribuição desta pesquisa.

5. Resultados Experimentais e Discussão

Este capítulo tem como objetivo realizar a prova de conceito do funcionamento do ambiente proposto seguindo as especificações e implementações apresentadas no Capítulo 4. São apresentados os cenários criados, como também a justificativa para a configuração escolhida. Ainda, são apresentadas imagens do ambiente desenvolvido, seguido pelos resultados quantitativos, a análise estatística e gráficos que expõem de forma unificada o desempenho.

5.1 Configuração dos Experimentos

A configuração dos experimentos foi dividida em três partes. Na primeira, o conceito é validar a teleoperação a partir de múltiplos métodos de condução. Para isso, os comandos de todas as interfaces de controle precisam ser traduzidos para sinais elétricos que permitam comandar a cadeira de rodas remotamente. Logo, foram criados cenários com os métodos de condução IHMG, rastreamento ocular e *joystick*.

Na segunda parte, o algoritmo proposto é validado para contagem automática de comandos. Todos os arquivos de dados de comandos gerados a partir dos experimentos foram submetidos ao algoritmo, sendo apresentados os resultados. A ideia é verificar o funcionamento independentemente da interface de controle.

Finalmente, na terceira parte, o objetivo é validar a teleoperação a longas distâncias. O estudo consiste em comparar os desempenhos através de conexões na mesma cidade e em cidades distintas. Foi verificado se o atraso presente na teleoperação a longas distâncias afetava consideravelmente o treinamento. Para esse fim, foram criados cenários nos quais o CT e a ST são conectados através de rede local, e também através da Internet. As características dos cenários são apresentadas na Tabela 7.

Para validar esses conceitos, diferentes cenários foram criados. O primeiro, segundo e terceiro foram baseados nos métodos de condução por *joystick*, *eye-tracker* e IHMG, respectivamente. O quarto e o quinto foram baseados na IHMG considerando uma teleoperação a longa distância a partir da Internet. Vale observar que a distância linear entre CT e a ST no cenário 4 foi de aproximadamente 2.596 km, e para o cenário 5 foi de 2172 km.

Cada cenário teve o mesmo percurso a ser cumprido pelo usuário em treinamento, que é apresentado na Figura 23. O número total de experimentos foi 40, e cada cenário foi realizado 8 vezes. Em todos os experimentos, foram coletados os dados de comandos, o RTT e o tempo de percurso.

Os experimentos foram conduzidos pelos pesquisadores participantes deste projeto (UFAM, UFU e Unicamp, no total de 3 pessoas rígidas), e não houve a preocupação em criar um grupo de controle, pois esse não era o objetivo deste trabalho.

Tabela 7 - Cenários para avaliar o sistema de treinamento em teleoperação.

<table>
<tr><th>Cenário</th><th>Número de Experimentos por cenário (Total 40)</th><th>Método de Condução</th><th>CT</th><th>ST</th><th>Tipo da Conexão</th><th>Largura de Banda</th></tr>
<tr><td>1</td><td>8</td><td>Joystick</td><td colspan="2" rowspan="3">Manaus</td><td rowspan="3">Local</td><td rowspan="3">100 Mbps</td></tr>
<tr><td>2</td><td>8</td><td>Eye-tracker</td></tr>
<tr><td>3</td><td>8</td><td rowspan="3">IHMG</td></tr>
<tr><td>4</td><td>8</td><td>Campinas</td><td rowspan="2">Manaus</td><td rowspan="2">Internet</td><td>70 Mbps</td></tr>
<tr><td>5</td><td>8</td><td>Uberlândia</td><td>40 Mbps</td></tr>
</table>

Figura 23 - Percurso para os experimentos de treinamento de teleoperação.

A Figura 24 apresenta a visão do usuário, praticando com a IHMG. É possível visualizar a câmera de visão em primeira pessoa (FPV), assim como a visão de terceira pessoa (TPV). A IHMG está no canto inferior direito.

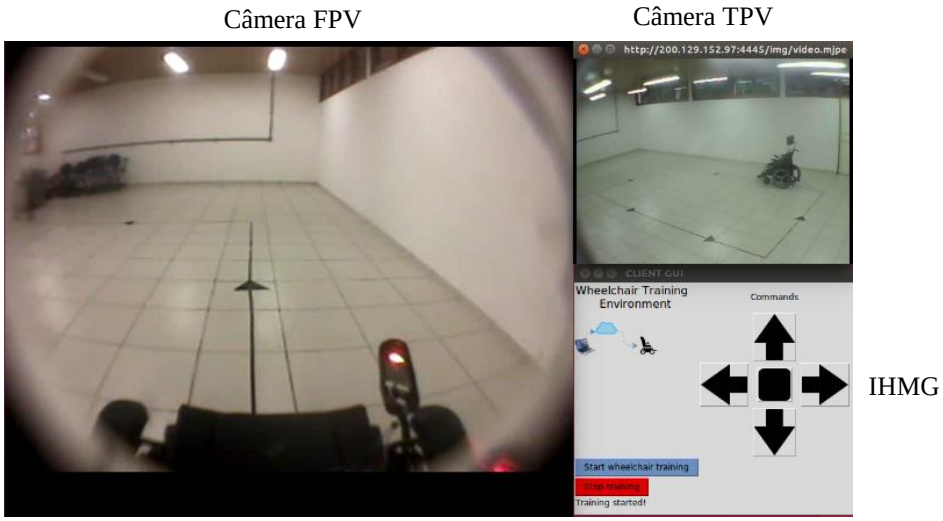


Figura 24 - Teleoperação pela IHMG.

A Figura 25 apresenta a interface de controle do *joystick*. É possível visualizar o *joystick* original da CRM adaptado com o sistema embarcado (Arduino) e o *streaming* de vídeo de ambas as câmeras FPV e TPV.

Figura 25 – Experimento de teleoperação pela interface de controle por *joystick*.

A Figura 26 apresenta a prática a partir da interface de controle por rastreamento ocular. O *eye-tracker* (Tobii 4c) está fixado ao monitor do computador.

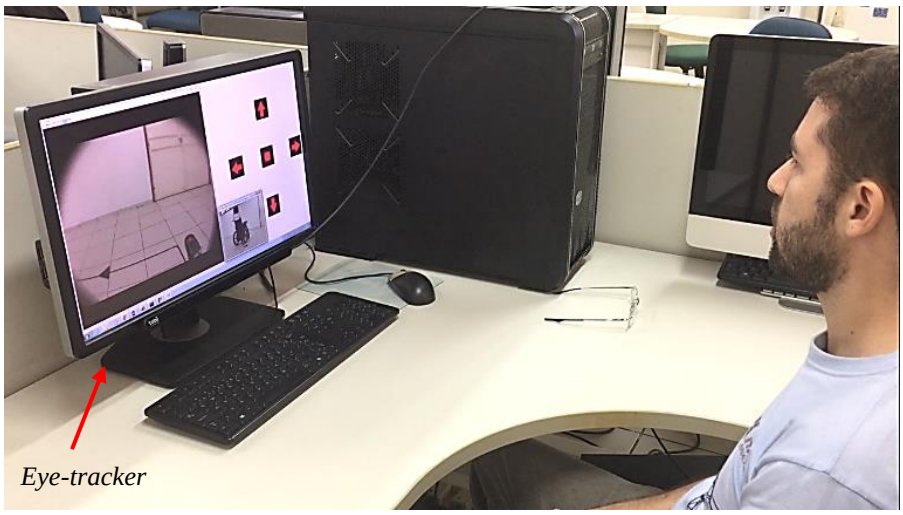


Figura 26 - Experimento de teleoperação pela interface de controle por rastreamento ocular.

5.2 Resultados Obtidos

Foram criados gráficos que apresentam o tempo de percurso, o número de comandos e o RTT (*delay*). Nos primeiros gráficos são apresentados os resultados individuais de cada um dos experimentos, como também os resultados considerando a média dos valores obtidos em cada cenário. Em seguida, são apresentadas tabelas com a análise estatística desses valores, contendo o desvio padrão, máximos e mínimos. Na etapa final, é apresentada a amostragem com todos os RTTs obtidos para teleoperação e, em seguida, é apresentado o histograma sobre os RTTs, considerando a teleoperação realizada a longas distâncias.

A Figura 27 apresenta o tempo de percurso nos experimentos de cada cenário. Logo após, a Figura 28 mostra o número de comandos em todos os experimentos.

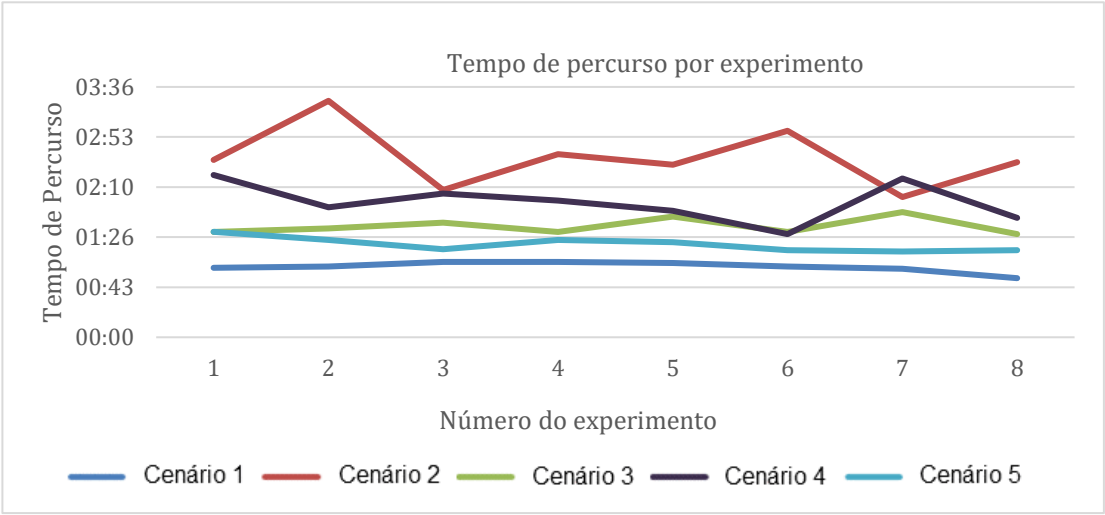


Figura 27 - Tempo de percurso nos experimentos.

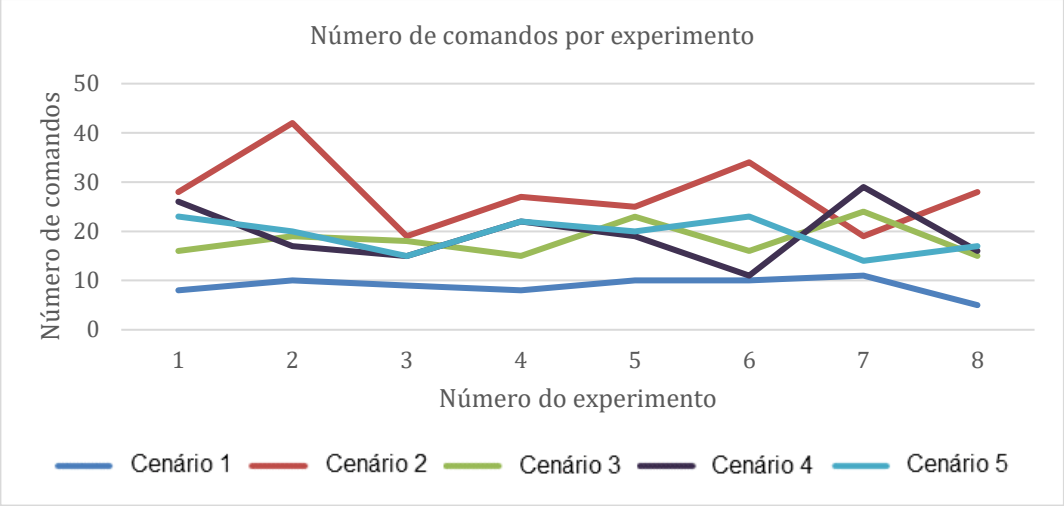


Figura 28 - O número de comandos ao longo dos experimentos.

A Figura 29 mostra a visão geral do tempo médio e percurso de cada cenário. Em seguida, a Figura 30 apresenta o número médio de comandos.

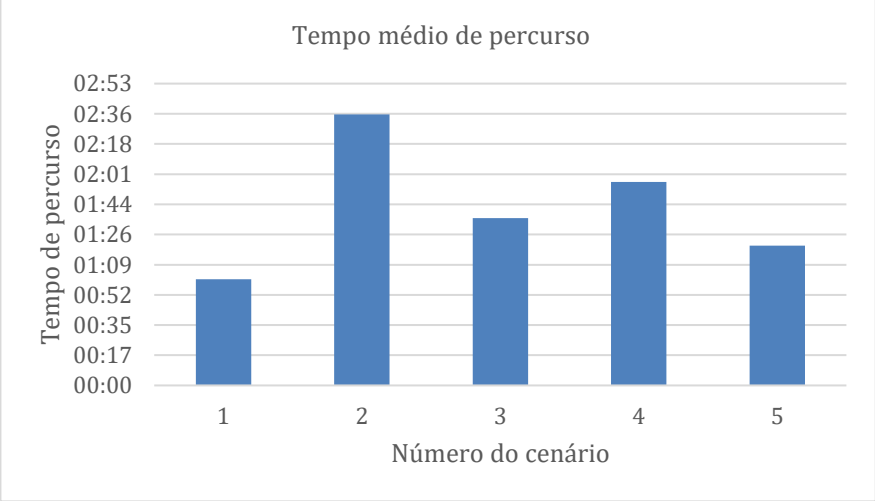


Figura 29 - Visão geral da média do tempo de percurso.

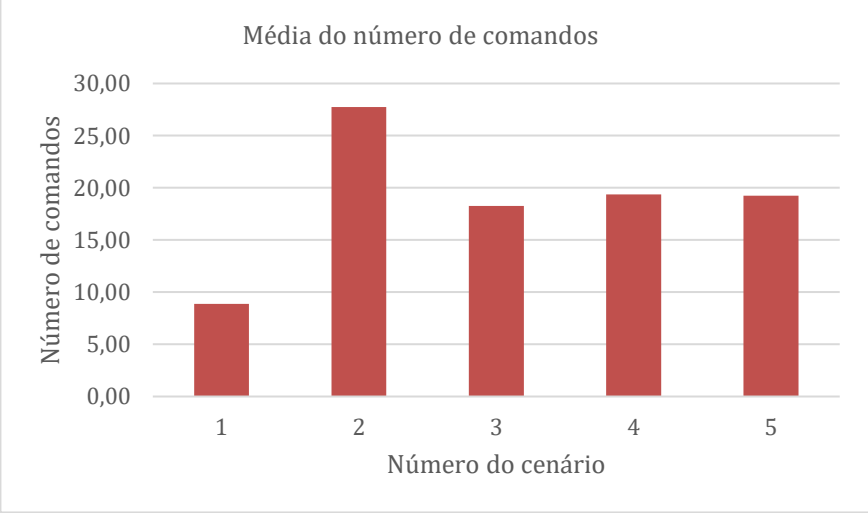


Figura 30 – Visão geral da média do número de comandos por cenário.

A Tabela 8 fornece os detalhes da análise estatística do tempo de percurso. A Tabela 9 apresenta as estatísticas do número de comandos. A Tabela 10 mostra a análise estatística do RTT .

Tabela 8 - Análise estatística do tempo de percurso de todos os cenários.

Tempo de percurso (min:seg)					
Cenário	**Média**	**±**	**Desvio Padrão**	**Máx.**	**Mín.**
1	01:01	±	00:04	01:05	00:51
2	02:35	±	00:25	03:24	02:01
3	01:36	±	00:07	01:48	01:29
4	01:57	±	00:16	02:20	01:29
5	01:20	±	00:06	01:31	01:14

Tabela 9 - Análise estatística do número de comandos em todos os cenários.

Número de comandos					
Cenário	**Média**	**±**	**Desvio Padrão**	**Máx.**	**Mín.**
1	8.88	±	1.76	11	5
2	27.75	±	7.10	42	19
3	18.25	±	3.31	24	15
4	19.38	±	5.59	29	11
5	19.25	±	3.31	23	14

Tabela 10 - Análise estatística do *delay* (RTT) em todos os cenários.

***Delay* - RTT (ms)**					
Cenário	**Média**	**±**	**Desvio Padrão**	**Máx.**	**Mín.**
1	2.24	±	1.01	23.10	1.49
2	1.89	±	3.38	49.41	0.99
3	2.07	±	4.05	92.23	1.44
4	85.27	±	24.49	532.06	76.68
5	130.12	±	8.12	266.62	122.20

A Figura 31 apresenta um gráfico das amostras dos RTTs de todos os experimentos realizados nos cenários 4 e 5. A Figura 32 apresenta os histogramas dos RTTs da teleoperação a longas distâncias (cenários 4 e 5).

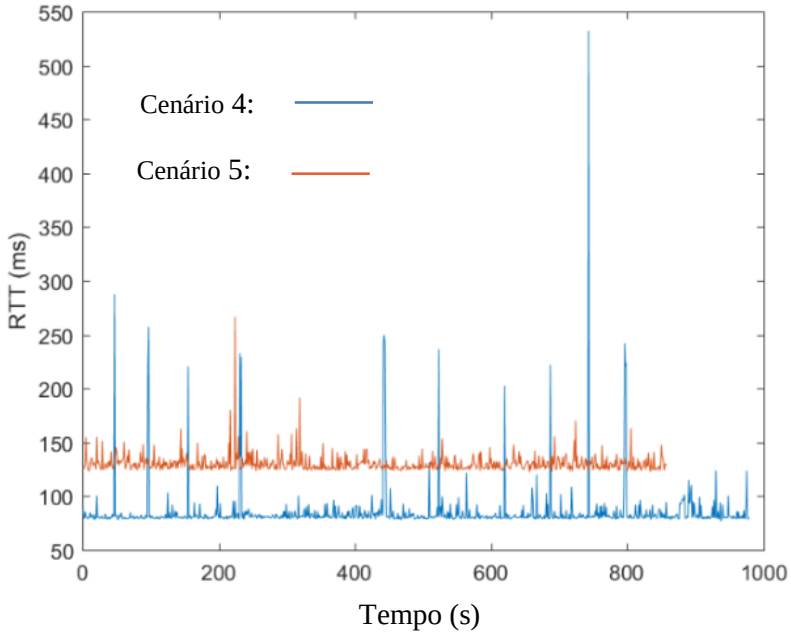


Figura 31 - Gráfico de todas as amostras de RTT dos experimentos realizadas pela Internet.

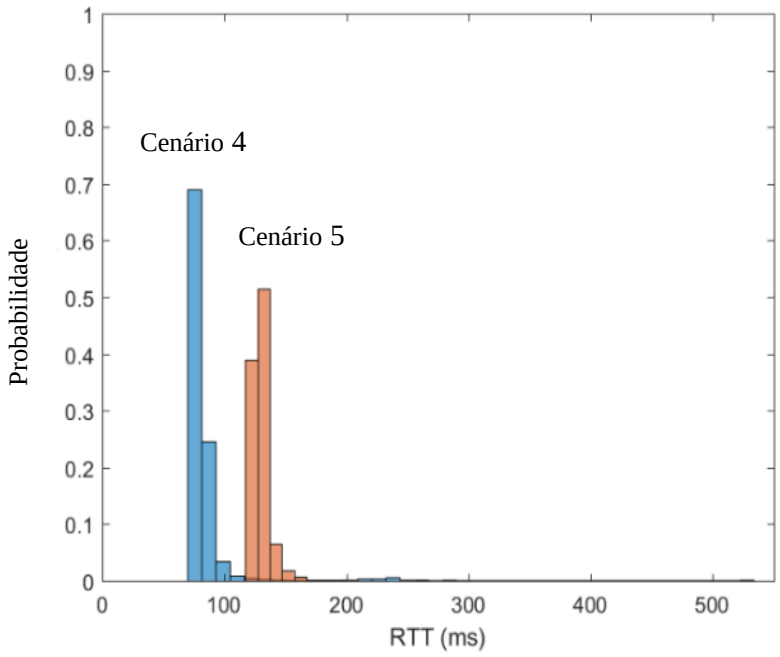


Figura 32 - Histogramas do *Delay* (RTT) dos experimentos executados a partir da Internet.

Os gráficos e tabelas apresentados neste Capítulo demonstram os resultados de forma agrupada, de modo a facilitar a visualização do desempenho do sistema. No Apêndice, são apresentados os resultados individuais de cada cenário, onde é possível visualizar os detalhes referentes aos valores obtidos em cada um dos experimentos.

5.3 Discussão

Esta seção apresenta os aspectos interpretados a partir dos resultados experimentais. Primeiramente, é feita uma análise qualitativa, onde são analisados os conceitos que deram origem à experimentação. Em seguida, é feita uma análise quantitativa, que trata sobre as principais observações feitas a partir dos valores observados.

5.3.1 Análise Qualitativa

O primeiro conceito validado foi a teleoperação a partir de múltiplos métodos de condução. Observou-se que, independentemente do método de condução utilizado, a CRM respondeu apropriadamente aos comandos do usuário. Esse conceito prova que a arquitetura funciona para múltiplos métodos de condução, tendo como requisito a utilização do padrão de mensagens proposto na Tabela 4.

O segundo conceito verificado foi o algoritmo de detecção automática do número de comandos. Foi verificado que para todos os experimentos foi possível extrair o número de comandos a partir do arquivo log gerado em cada experimento, independentemente do método de condução utilizado, oferecendo assim aos usuários em treinamento uma maneira eficaz de escolher o método de condução mais apropriado.

O terceiro conceito foi validar a teleoperação a longas distâncias. Foi verificado que foi possível praticar a condução de CRM mesmo em situações onde o CT e a ST estavam localizados em cidades diferentes. Por exemplo, foi possível realizar a teleoperação entre uma distância linear de 2596 km entre o usuário e a CRM, validando que a arquitetura proposta foi capaz de permitir essa comunicação.

5.3.2 Análise Quantitativa

Sobre a análise quantitativa, foi observado que o *joystick* (cenário 1) obteve o tempo médio de percurso (01min: 01s) e o número médio de comandos (8,88). Para a performance da interface por rastreamento ocular (cenário 2), obteve-se o tempo médio de percurso (02min: 35s), e o número médio de comandos (27,75). Essa diferença se dá pelo fato da dificuldade em comandar a CRM a partir do rastreamento ocular, fato que era esperado. Embora a condução por rastreamento ocular não seja uma tarefa fácil, possa ser a única opção para alguns usuários com deficiências severas.

Comparando os desempenhos dos treinamentos a partir da conexão local (cenário 3) com a conexão pela Internet (cenários 4 e 5), notou-se que não houve uma diferença significativa nos tempos médios de percurso, obteve-se: (1min: 36s), (1min: 20s), e (1min: 57 s). O número médio de comandos nesses cenários foi muito semelhante; no cenário 3 foram (18,25 comandos), no cenário 4 (19,38 comandos) e no cenário 5 (19,25 comandos), os quais representam praticamente o mesmo número médio de comandos. Essas semelhanças sugerem que o atraso na comunicação não afetou o treinamento. À vista disso, esses resultados contribuem para uma melhor compreensão do uso da teleoperação para treinamento de CRM, verificando que é possível praticar a condução de CRM a longas distâncias apesar do atraso presente nas conexões.

5.3.3 Contribuições

Outros autores criaram ambientes de treinamento para CRM, conforme mencionado no Capítulo 3 (Trabalhos Relacionados). As abordagens foram focadas em ambientes virtuais, com o *joystick* como método de condução. Não obstante, este trabalho propôs um ambiente de treinamento com múltiplos métodos de condução, o que é uma nova abordagem. O ambiente permite não só o treinamento a partir do *joystick*, mas através de novas tecnologias, como por exemplo o rastreamento ocular. Desta forma, também são incluídas pessoas com deficiência severa como possíveis usuários para o treinamento. Ademais, foi implementado um método genérico para condução (IHMG), que pode ser utilizado como base para inclusão de novos métodos de condução. Desse modo, possibilita novos trabalhos para a continuidade desta pesquisa.

Outro aspecto diferenciado em relação aos trabalhos relacionados é a criação de um ambiente de treinamento realista, no qual não ocorre o problema de representação do comportamento cinemático. Essa dificuldade foi mencionada por diversos autores que desenvolveram ambientes de treinamento virtuais, nos quais foram observados problemas quanto à representação da velocidade da CRM, e em serem fidedignos ao comportamento real da CRM. Assim sendo, na arquitetura proposta, elimina-se essa dificuldade, e passa-se a permitir a usuários uma experiência mais realista de treinamento, onde ao invés de comandar um dispositivo virtual (semelhante a um jogo), é comandada a própria CRM. Possibilitando que seja gerada uma maior motivação para que os usuários possam aprender em uma CRM real, porém, sem o risco de acidentes provindos da prática.

Finalmente, outra contribuição desta pesquisa é possibilitar que novos usuários possam praticar a longas distâncias, mesmo que esses estejam localizados em cidades diferentes de onde se encontra a CRM. Logo, esta característica facilita a replicação do ambiente proposto, pois com uma infraestrutura de treinamento contendo a CRM, usuários poderão praticar à distância a partir de hospitais e centros de reabilitação, tendo como requisito um computador com conexão à Internet e pelo menos um dos métodos de condução.

5.4 Conclusão do Capítulo

O objetivo principal deste capítulo foi realizar a prova de conceito do funcionamento do ambiente de treinamento proposto. Na primeira etapa foi apresentada a configuração dos experimentos detalhando sobre a justificativa de cada cenário. Em seguida, foram demonstrados gráficos considerando a integração dos resultados de forma comparativa. Similarmente, foram apresentadas tabelas com análise estatística.

Finalmente, foi apresentada a discussão, na qual foram abordados aspectos qualitativos e quantitativos, e, por último, foram apontadas as contribuições desta pesquisa. Esses conceitos servirão como base para o próximo capítulo, o qual irá expor as conclusões, considerações finais e trabalhos futuros.

6. Conclusões e Trabalhos Futuros

O objetivo desse trabalho foi implementar um ambiente realista de treinamento para novos usuários de CRM capaz de suportar múltiplos métodos de condução e ser operado remotamente. Esse ambiente permite que usuários pratiquem, sem riscos de acidente, tendo uma experiência realista, sem a utilização de virtualização e simulações computacionais. Ainda, o ambiente abrange usuários de diferentes tipos, desde pessoas com deficiências nos membros inferiores, a usuários com deficiência severa, os quais dependem de métodos alternativos de condução.

Em vista disso, o ambiente proporciona opção de praticar com o *joystick*, rastreamento ocular, e também possui uma IHMG, a qual tem toda infraestrutura necessária para incluir novos métodos de condução, dando oportunidade para que os novos usuários testem os diferentes métodos e escolham o mais adequado de acordo com o melhor desempenho que obtiveram durante o treinamento, possibilitando Criando aos novos usuários a possibilidade de aprendizado prévio à obtenção da própria CRM.

Para possibilitar o desenvolvimento desta pesqusia, partiu-se de um estudo sobre o estado da arte sobre os ambientes de treinamento para CRM. Foram analisadas as principais abordagens, verificando os tipos de ambiente e os métodos de condução que estão sendo utilizados. Também foi realizado um estudo sobre a literatura de sistemas de teleoperação, onde foram verificados métodos de baixo custo utilizados para teleoperação a longas distâncias, assim como métodos adequados para ambientes *indoor* (fechados), como exemplo hospitais e clínicas de reabilitação física. A análise dessas abordagens permitiu a identificação dos aspectos mais relevantes sobre os ambientes de treinamento atuais, e também permitiu analisar diferentes técnicas aplicadas para teleoperação.

Após o estudo dos aspectos mais relevantes analisados nessas abordagens, especificou-se um modelo de ambiente de treinamento. Logo, esse ambiente foi dividido em duas partes:

- **Centro de Treinamento (CT):** local no qual usuários frequentam para realizar o treinamento, e onde ficam localizados todos os acessórios e dispositivos necessários para condução por teleoperação.
- **Sala de Teleoperação (ST):** local no qual fica localizada a CRM e infraestrutura de comunicação necessária para permitir a teleoperação. Aqui todos os comandos são

recebidos e executados, paralelamente, o vídeo do local é transmitido em tempo real para o CT.

Cada uma dessas partes foi subdivida em partes menores. Primeiramente no CT foram desenvolvidos:

- **Condução por *Joystick:*** foi criado um método de condução por joystick, para isso, foram analisados os sinais elétricos gerados pelo modelo original da CRM, verificando o comportamento matemático. A partir disso, foi criado um modelo baseado em um vetor de magnitude e angulação.
- **Condução por rastreamento ocular:** foi utilizado um modelo de *eye-tracker* para captura da posição do olhar do usuário na tela do computador. Foi desenvolvido um modelo baseado em mapa de pixels, onde os comandos são gerados de acordo com um padrão específico criado.
- **Condução por IHMG:** foi criado um método genérico para gerar comandos discretos e comandar a CRM de forma simplificada. Esse método foi fundamental para a avaliação do sistema, assim como é peça chave para inclusão de outras formas de condução no ambiente.

Na ST foram desenvolvidos:

- **Padrão de sinais elétricos para condução remota:** foi desenvolvido um padrão de mensagens baseados nos eixos X e Y para comandar a CRM remotamente. Esse padrão permite que mensagens do tipo *string* sejam traduzidas para sinais elétricos analógicos responsáveis por comandar os motores da CRM.
- ***Streaming* de vídeo:** foi criado um servidor de *streaming* de vídeo em tempo real que permite que o usuário em treinamento possa visualizar dois pontos de vistas diferentes da CRM. Nesse método foram aplicadas técnicas de diminuição do consumo de banda necessário, como também técnicas para diminuição do *delay* presente no *streaming* a longa distância.
- **Arquitetura de comunicação:** foi desenvolvida uma arquitetura de comunicação a longas distâncias que permite a todos os dispositivos presentes no CT comunicarem-se de forma apropriada com a ST. A arquitetura foi baseada em diferentes protocolos de rede e a partir do redirecionamento de mensagens durante tráfego de pacotes entre CT e ST.

- **Arquitetura para avaliação do desempenho do usuário:** foi desenvolvido uma arquitetura para analisar o desempenho do usuário a partir do histórico de comandos gerados durante o treinamento. Para isso, foi utilizado um algoritmo que permite verificar a evolução do usuário, auxiliando na determinação do método de condução no qual foi obtido o melhor desempenho.
- ***Softwares* implementados:** foi criado um modelo baseado em multitarefas para permitir que comandos fossem gerados, recebidos, interpretados e convertidos em sinais elétricos. Paralelamente, esse modelo permite que sejam executadas as tarefas de vídeo *streaming* e a gravação do histórico de comandos.

Na etapa de análise do modelo proposto, foram utilizadas diferentes configurações. Para isso, foram criados cenários considerando os diferentes métodos de condução desenvolvidos, tal como considerando as diferentes situações de teleoperação para curta e longa distância. Os resultados obtidos foram analisados de forma qualitativa e quantitativa, apresentando-se os principais aspectos observados, e, desse modo, apresentando a contribuição desta pesquisa.

Portanto, os objetivos definidos no início deste trabalho foram alcançados com o desenvolvimento do modelo proposto.

Ademais, com o desenvolvimento da ferramenta, os problemas descritos no início da pesquisa foram tratados da seguinte maneira

- **Dificuldade em representar o comportamento físico da CRM:** para solucionar esse problema, ao invés de utilizar ambientes virtuais e simulações computacionais, foi desenvolvido um modelo a partir de teleoperação. Esse modelo é baseado em uma CRM real comandada remotamente e dá ao usuário uma experiência realista de treinamento.
- **Opção limitada sobre o método de condução em ambientes de treinamento:** para eliminar o problema do número limitado de opções para praticar, que geralmente é apenas por joystick, foi criado um modelo baseado em múltiplos métodos de condução. No modelo proposto, utiliza-se *joystick* e rastremaneto ocular, além de contar com um método genérico (IHMG) que facilita a inclusão de novas formas de condução.

Como trabalhos futuros, espera-se incluir novos métodos de condução ao ambiente implementado. Por exemplo, poderá ser incluído o método de condução por sinais musculares, o qual seria baseado em sEMG (*Surface Electromygraphy*), do português eletromiografia de superfície. Essa técnica conta com eletrodos posicionados na superfície do músculo, os quais são responsáveis por capturar o sinal elétrico gerado a partir da movimentação muscular. Para inclusão desse método de condução, poderia ser reutilizada a estrutura da IHMG.

Adicionalmente, poder-se-ia considerar características dinâmicas ao ambiente de treinamento, por exemplo, o peso do usuário, desse modo, seria aumentado o nível de realismo do ambiente. Finalmente, outro trabalho seria avaliar o ambiente desenvolvido considerando diferentes grupos de pessoas, como, por exemplo, usuários com deficiência nos membros inferiores e usuários com deficiência severa.

Por fim, as publicações geradas com esta pesquisa, estão apresentadas no Apêndice F.

Referências

ACM. Digital Library. 2018. Disponível em: < https://dl.acm.org/ >. Acesso em: Agosto de 2018.

ARCHAMBAULT, P. S. et al. Comparison of powered wheelchair driving performance in a real and in a simulated environment. Virtual Rehabilitation (ICVR), 2011 International Conference on, 2011, IEEE. p.1-7.

ARDUINO. Open Source Electronics Platform. 2018. Disponível em: < https://www.arduino.cc/ >. Acesso em: Agosto de 2018.

CAETANO, D. et al. DEMO On the Use of Augmented Reality Techniques in a Telerehabilitation Environment for Wheelchair Users' Training. 2014 Ieee International Symposium on Mixed and Augmented Reality (Ismar) - Science and Technology, p. 329-330, 2014. ISSN 1554-7868. Disponível em: < <Go to ISI>://WOS:000349548300085 >.

CARISSIMI, A. D. S.; ROCHOL, J.; GRANVILLE, L. Z. Redes de computadores. Bookman, 2009.

FFMPEG. Video streaming cross-platform. 2018. Disponível em: < https://www.ffmpeg.org/ >. Acesso em: Agosto de 2018.

FREEDOM. Cadeira de Rodas Motorizada. 2018. Disponível em: < http://www.freedom.ind.br/ >. Acesso em: Agosto de 2018.

GONZALEZ, J. D. T. et al. On internet based supermedia enhanced telepresence via cellular data network. IECON 2012-38th Annual Conference on IEEE Industrial Electronics Society, 2012, IEEE. p.2750-2755.

HA, V. K. L.; NGUYEN, T. N.; NGUYEN, H. T. A Telepresence Wheelchair Using Cellular Network Infrastructure in Outdoor Environments. 38th Annual International Conference of the

IEEE-Engineering-in-Medicine-and-Biology-Society (EMBC), 2016, Orlando, FL. Ieee, Aug 16-20. p.5352-5355.

HOLLABAUGH, C. Embedded Linux: hardware, software, and interfacing. Addison-Wesley Professional, 2002. ISBN 0672322269.

IBGE, I. B. D. G. E. E.-. Censo Demográfico. 2010. Disponível em: < https://biblioteca.ibge.gov.br/visualizacao/periodicos/94/cd_2010_religiao_deficiencia.pdf >. Acesso em: Agosto de 2018.

IBM. Kernel Linux 2007.

IEEE. XPlore. 2018. Disponível em: < http://ieeexplore.ieee.org >. Acesso em: Agosto de 2018.

JOHN, N. W. et al. The Implementation and Validation of a Virtual Environment for Training Powered Wheelchair Manoeuvres. IEEE Transactions on Visualization and Computer Graphics, 2017. ISSN 1077-2626.

KATO, T.; HIGASHI, T.; SHIMIZU, K. Teleoperation of a robot arm system using pneumatic artificial rubber muscles: Teleoperation over the internet using UDP and a web camera. Broadband, Wireless Computing, Communication and Applications (BWCCA), 2010 International Conference on, 2010, IEEE. p.714-718.

KTENA, S. I. et al. A virtual reality platform for safe evaluation and training of natural gaze-based wheelchair driving. 7th Annual International IEEE EMBS Conference on Neural Engineering (NER), 2015, Montpellier, FRANCE. Ieee, Apr 22-24. p.236-239.

KUROSE, J. Computer Networking: A Top-Down Approach /James F. Kurose, Keith W. Ross. 2013.

LINKSYS. WVC54GCA Documentation. 2018. Disponível em: < http://downloads.linksys.com/downloads/userguide/1224639055972/WVC54GCA-EU_V10_UG_A-WEB.pdf >. Acesso em: Agosto de 2018.

MANZI, A. et al. Use Case Evaluation of a Cloud Robotics Teleoperation System (Short Paper). Cloud Networking (Cloudnet), 2016 5th IEEE International Conference on, 2016, IEEE. p.208-211.

MICROSYSTEMS, S. Essential Java Classes 2005. Disponível em: < https://www.math.uni-hamburg.de/doc/java/tutorial/essential/threads/definition.html >. Acesso em: Agosto de 2018.

MORERE, Y. et al. ViEW, a wheelchair simulator for driving analysis. 2015 International Conference on Virtual Rehabilitation Proceedings (Icvr), p. 100-105, 2015. Disponível em: < <Go to ISI>://WOS:000380459800005 >.

OZER, J. Video Encoding by the Numbers: Eliminate the Guesswork from Your Streaming Video. Doceo Publishing, 2016. ISBN 9780998453002.

PILLOW. Python Imaging Library. 2018. Disponível em: < https://pillow.readthedocs.io/en/latest/ >. Acesso em: Agosto de 2018.

PUBMED. Library of Medicine. 2018. Disponível em: < https://www.ncbi.nlm.nih.gov/pubmed/ >. Acesso em: Agosto de 2018.

PYSERIAL. Python Module for Serial Communication. 2018. Disponível em: < https://pythonhosted.org/pyserial/ >. Acesso em: Agosto de 2018.

PYTHON. Programming Language. 2018a. Disponível em: < https://www.python.org/ >. Acesso em: Agosto de 2018.

_____. Threading Documentation. 2018b. Disponível em: < https://docs.python.org/2/library/threading.html >. Acesso em: Agosto de 2018.

REGEX. Regular Expressions Python Module. 2018. Disponível em: < https://pypi.org/project/regex/ >. Acesso em: Agosto de 2018.

RODRIGUEZ, N. Development of a Wheelchair Simulator for Children with Multiple Disabilities. 2015 3rd IEEE VR International Workshop on Virtual and Augmented Assistive Technology (VAAT), p. 19-21, 2015. Disponível em: < <Go to ISI>://WOS:000380551200005 >.

ROSSOL, N. et al. A framework for adaptive training and games in virtual reality rehabilitation environments. Proceedings of the 10th International Conference on Virtual Reality Continuum and Its Applications in Industry, 2011, ACM. p.343-346.

SCIENCE, W. O. Web of Science. 2018. Disponível em: < http://webofknowledge.com >. Acesso em: Agosto de 2018.

SHENAI, M. B. et al. Virtual interactive presence for real-time, long-distance surgical collaboration during complex microsurgical procedures Technical note. Journal of Neurosurgery, v. 121, n. 2, p. 277-284, Aug 2014. ISSN 0022-3085. Disponível em: < <Go to ISI>://WOS:000339473000006 >.

SOCKET, M. Low-level Networking Interface for Python. 2018. Disponível em: < https://docs.python.org/2/library/socket.html >. Acesso em: Agosto de 2018.

TAO, G.; ARCHAMBAULT, P. S. Powered wheelchair simulator development: implementing combined navigation-reaching tasks with a 3D hand motion controller. Journal of Neuroengineering and Rehabilitation, v. 13, p. 13, Jan 2016. ISSN 1743-0003. Disponível em: < <Go to ISI>://WOS:000368392200001 >.

TI. Texas Instruments Amplifier. 2018. Disponível em: < http://www.ti.com/lit/ds/snosc16d/snosc16d.pdf >. Acesso em: Agosto de 2018.

TKINTER. Graphical User Interface (GUI) Package for Python. 2018. Disponível em: < https://wiki.python.org/moin/TkInter >. Acesso em: Agosto de 2018.

TOBII. API (C#) Tobii 4c. 2018. Disponível em: < https://developer.tobii.com/tag/c-sharp/ >. Acesso em: Agosto de 2018.

UBUNTU. 16.04 Documentation. 2018a. Disponível em: < https://www.ubuntu.com >. Acesso em: Agosto de 2018.

_____. Build-essential Package for Ubuntu. 2018b. Disponível em: < https://packages.ubuntu.com/xenial/build-essential >. Acesso em: Agosto de 2018.

UDOO. Udoo-Dual Board. 2018. Disponível em: < https://www.udoo.org/udoo-dual-and-quad/ >. Acesso em: Agosto de 2018.

UDOOBUNTU. Image and Documentation. 2018. Disponível em: < https://www.udoo.org/udoobuntu-the-official-udoo-linux-operating-system/ >. Acesso em: Agosto de 2018.

WETHERALL, J.; TANENBAUM, A. Redes de Computadores. 5ª edição. Rio de Janeiro: Editora Campus, 2011.

WIRESHARK. Network Protocol Analyzer. 2018. Disponível em: < https://www.wireshark.org/ >. Acesso em: Agosto de 2018.

XU, Q. et al. Teleoperation of Humanoid Motion Using 3G Communication Network. 2012 Ieee International Conference on Robotics and Biomimetics (Robio 2012), p. 6, 2012. Disponível em: < <Go to ISI>://WOS:000321004000229 >.

YAGHMOUR, K. Building embedded Linux systems. O'Reilly Japan, 2003. ISBN 4873111617.

Anexo

Dados do IBGE sobre pessoas com deficiência

Tabela 11 - Distribuição percentual da população residente, por tipo de deficiência, segundo o sexo e os grupos de idade (Ibge, 2010).

Sexo e grupos de idade	Distribuição percentual da população residente (%)						
	Total (1) (2)	Tipo de deficiência					
		Pelo menos uma das deficiências enumeradas (1)	Visual	Auditiva	Motora	Mental ou intelctual	Nenhuma destas deficiências (3)
Total	**100,0**	**23,9**	**18,8**	**5,1**	**7,0**	**1,4**	**76,1**
0 a 14 anos	100,0	7,5	5,3	1,3	1,0	0,9	92,5
15 a 64 anos	100,0	24,9	20,1	4,2	5,7	1,4	75,0
65 anos ou mais	100,0	67,7	49,8	25,6	38,3	2,9	32,3
Homens	**100,0**	**21,2**	**16,0**	**5,3**	**5,3**	**1,5**	**78,8**
0 a 14 anos	100,0	7,3	4,8	1,4	1,0	1,0	92,7
15 a 64 anos	100,0	22,2	17,1	4,5	4,5	1,6	77,8
65 anos ou mais	100,0	64,6	47,3	28,2	30,9	2,8	35,4
Mulheres	**100,0**	**26,5**	**21,4**	**4,9**	**8,5**	**1,2**	**73,5**
0 a 14 anos	100,0	7,8	5,9	1,3	1,0	0,7	92,2
15 a 64 anos	100,0	27,6	23,1	4,0	6,8	1,2	72,4
65 anos ou mais	100,0	70,1	51,7	23,6	44,0	3,0	29,9

Fonte: IBGE, Censo Demográfico 2010.

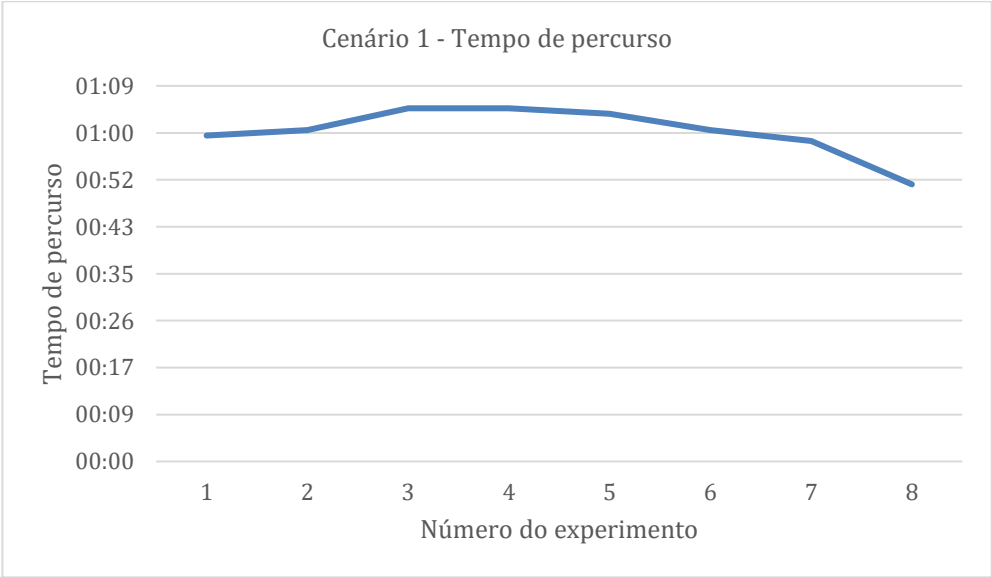
(1) As pessoas incluídas em mais de um tipo de deficiência foram contadas apenas uma vez. (2) Inclusive as pessoas sem declaração destas deficiências. (3) Inclusive a população sem qualquer tipo de deficiência.

Apêndice

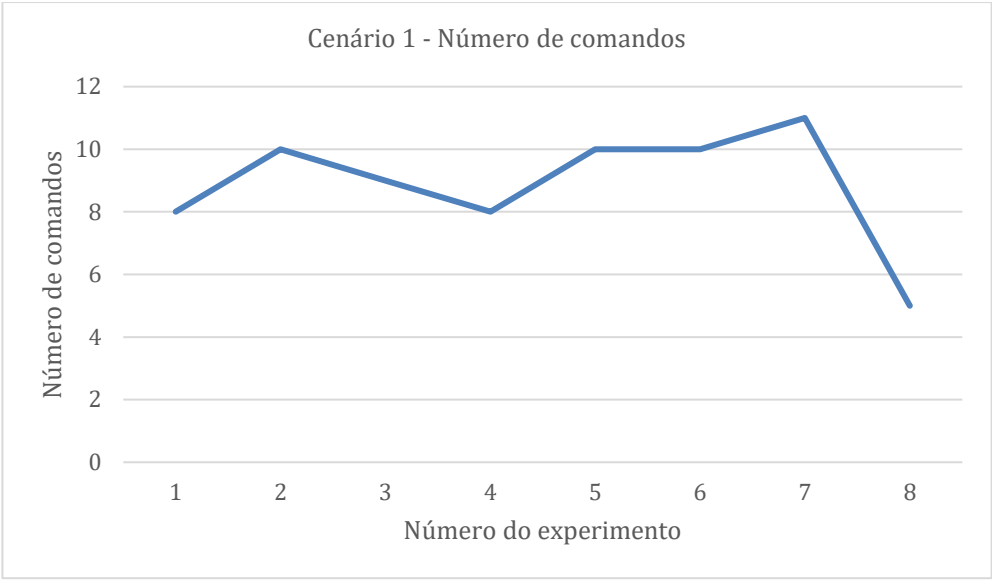
Apêndice A - Resultados individuais dos experimentos do Cenário 1

Tabela 12 - Análise estatística de todos os experimentos realizados no Cenário 1.

CENÁRIO 1 – *JOYSTICK*						
Performance de treinamento			Avaliação da comunicação - *Delay*/RTT (ms)			
N. do Exp.	N. de Comandos	Tempo gasto	Média	σ	Máx	Mín
1	8	01:00	2,3048	1,1407	11,581	1,493
2	10	01:01	2,2187	0,4001	4,161	1,664
3	9	01:05	2,1054	0,26	3,004	1,534
4	8	01:05	2,2436	0,7409	8,395	1,639
5	10	01:04	2,2026	0,5131	5,261	1,564
6	10	01:01	2,1372	0,3229	3,17	1,567
7	11	00:59	2,4112	2,1282	23,101	1,646
8	5	00:51	2,2728	0,6564	7,156	1,659

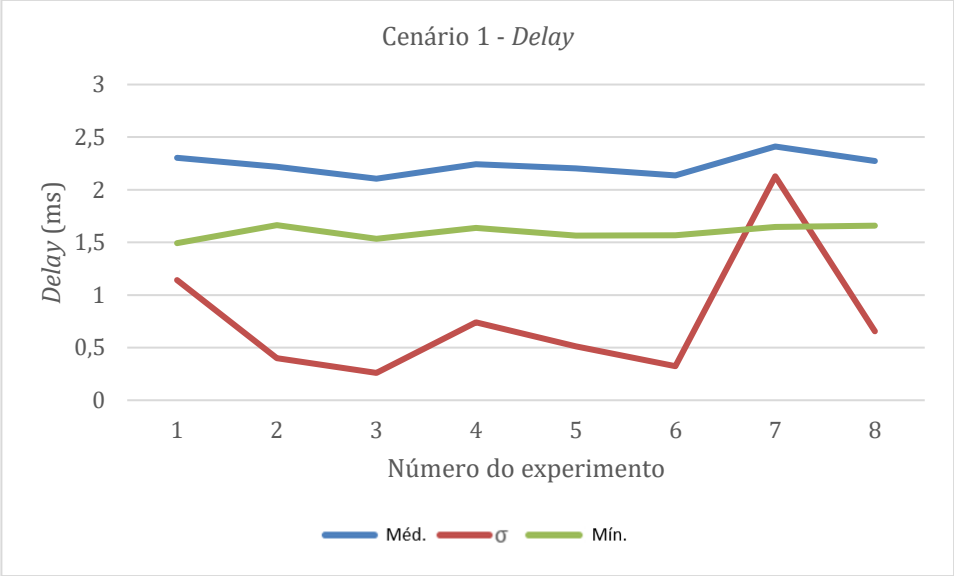


(a)

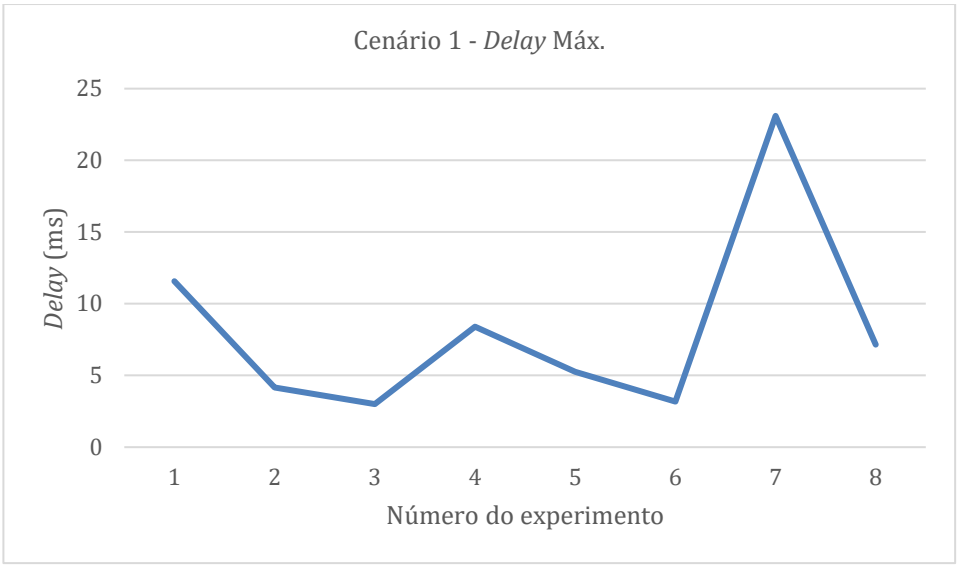


(b)

Figura 33 - (a) Tempo gasto nos experimentos do Cenário 1. (b) Número médio de comandos desse Cenário.



(a)



(b)

Figura 34 - (a) *Delay* médio em todos os experimentos do Cenário 1. (b) *Delay* máximo nos experimentos desse Cenário.

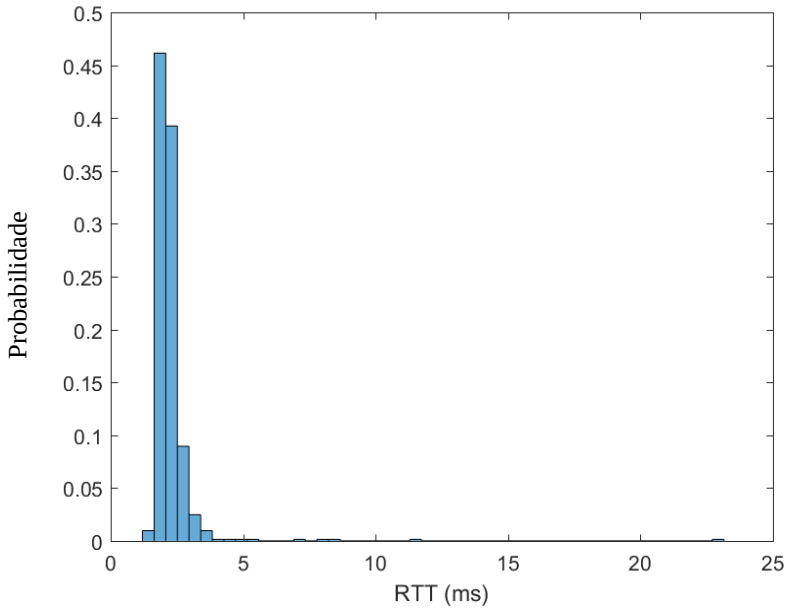
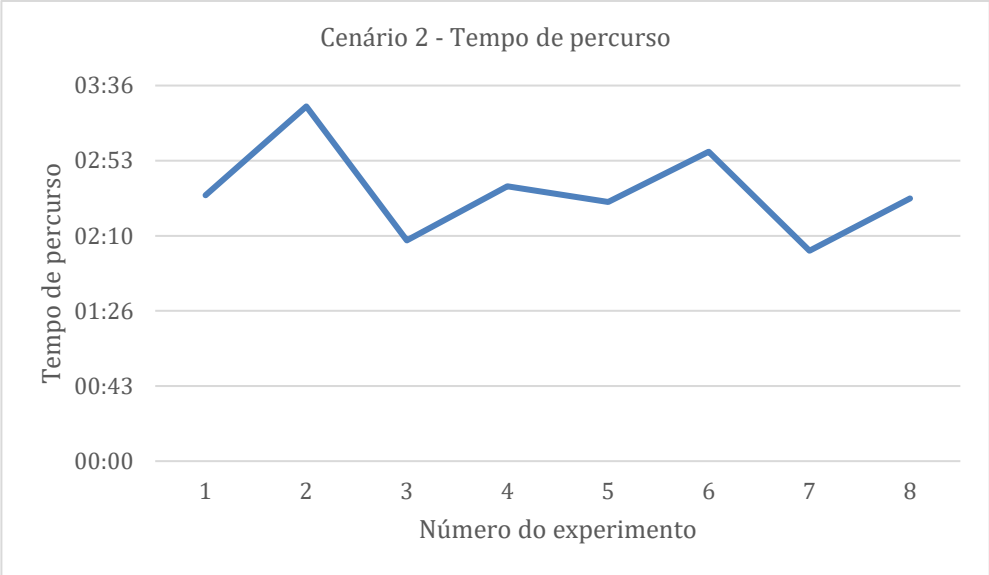


Figura 35 - Histogramas do *Delay* (RTT) dos experimentos do Cenário 1.

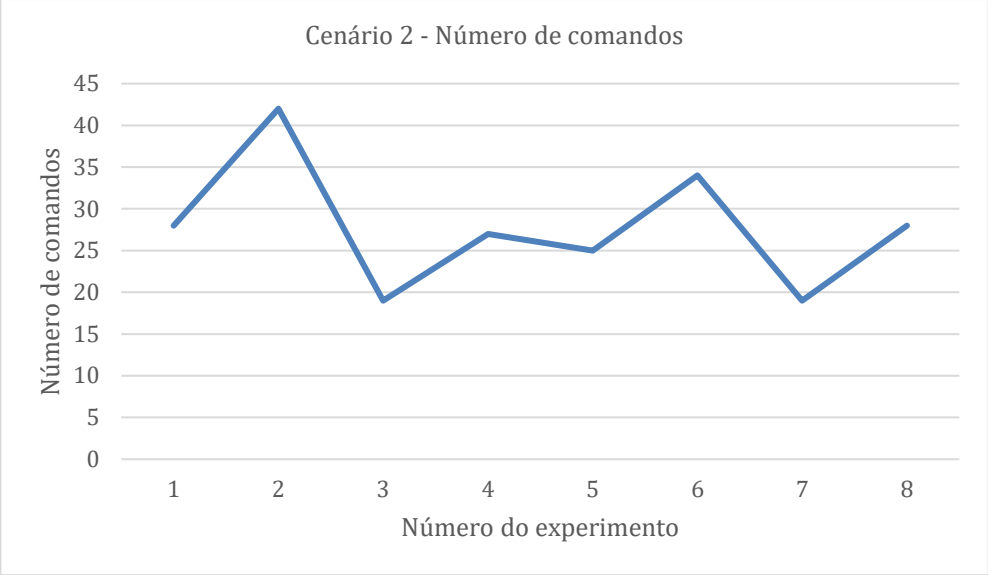
Apêndice B - Resultados individuais dos experimentos do Cenário 2

Tabela 13 - Análise estatística de todos os experimentos realizados no Cenário 2.

CENÁRIO 2 - *EYE-TRACKER*						
	Performance de treinamento		Avaliação da comunicação - *Delay*/RTT (ms)			
N. do Exp.	N. de Comandos	Tempo gasto	Média	σ	Máx	Mín
1	28	02:33	1,7387	0,4051	2,85	1,23
2	42	03:24	1,5017	0,3279	2,97	1
3	19	02:07	1,5663	0,4201	3,4	1,12
4	27	02:38	2,1023	3,055	28,32	1,08
5	25	02:29	3,0143	7,7345	49,41	1,2
6	34	02:58	1,5948	0,5464	5,38	1,19
7	19	02:01	1,5666	0,3923	3,35	1,03
8	28	02:31	1,5554	0,4971	4,74	0,99

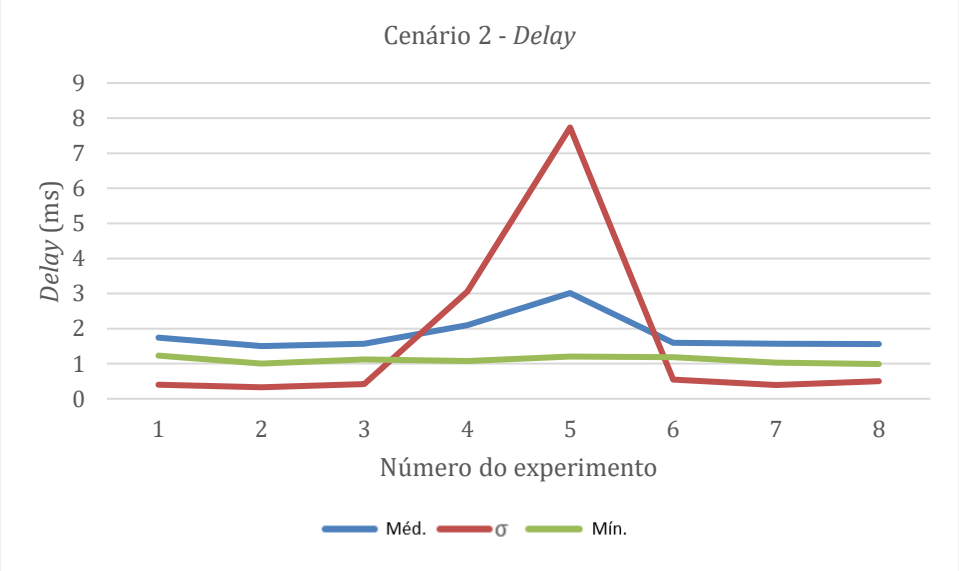


(a)

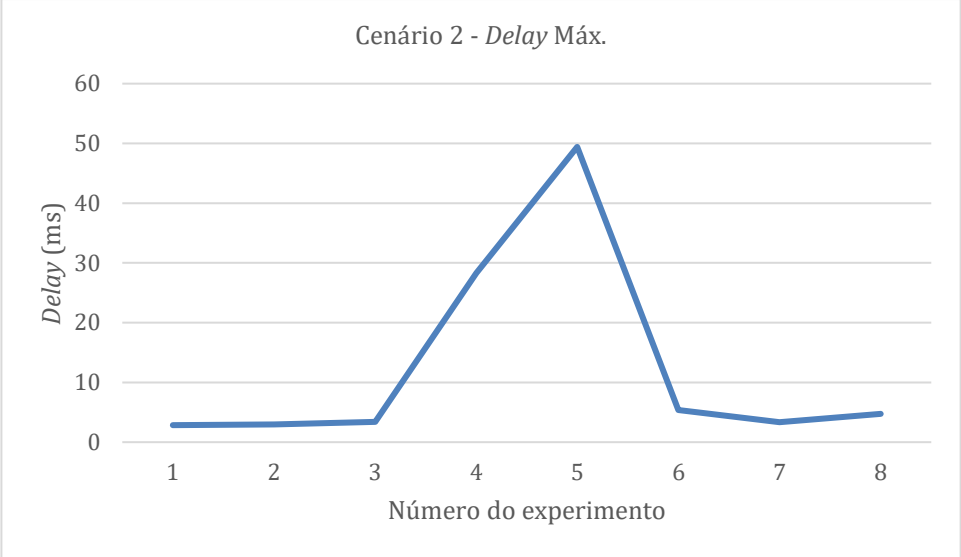


(b)

Figura 36 - (a) Tempo gasto nos experimentos do Cenário 2. (b) Número médio de comandos desse Cenário.



(a)



(b)

Figura 37 - (a) *Delay* médio em todos os experimentos do Cenário 2. (b) *Delay* máximo nos experimentos desse Cenário.

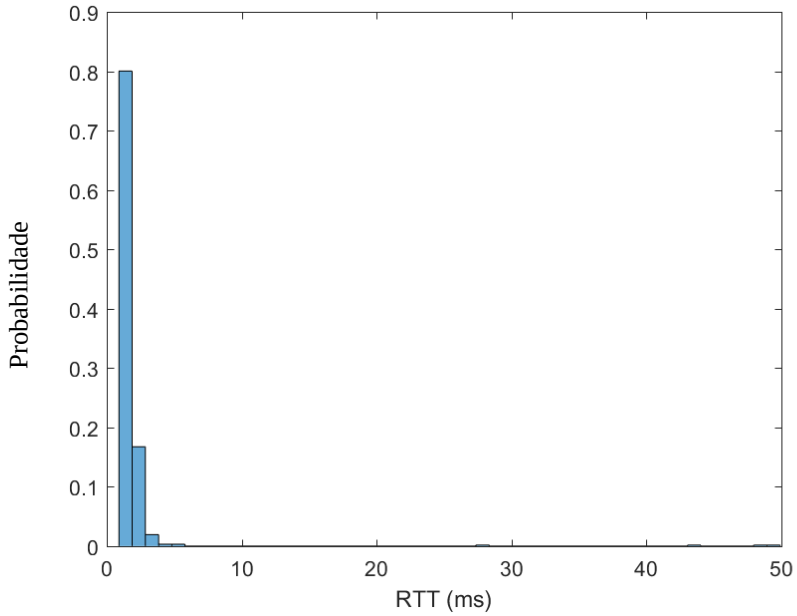
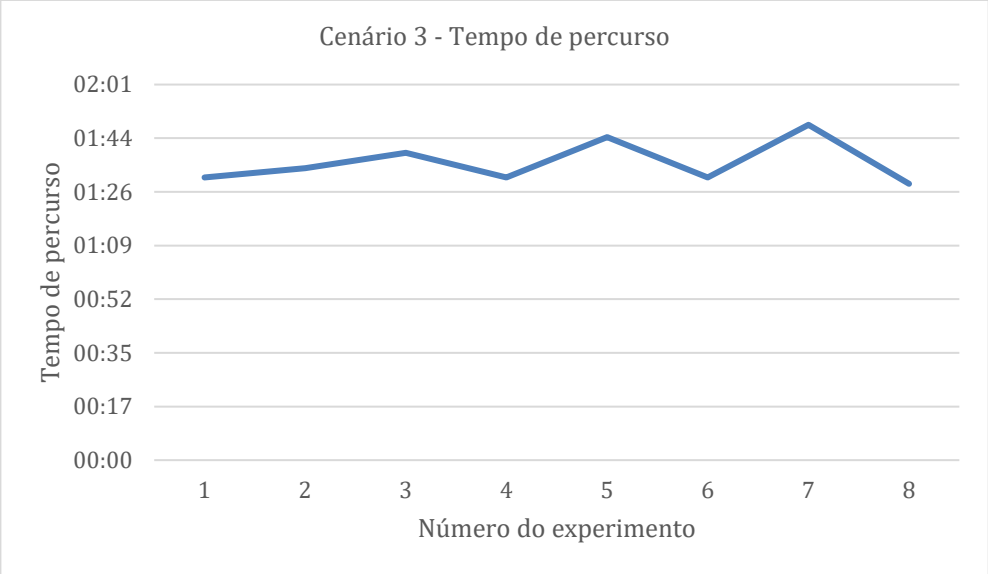


Figura 38 - Histogramas do *Delay* (RTT) dos experimentos do Cenário 2.

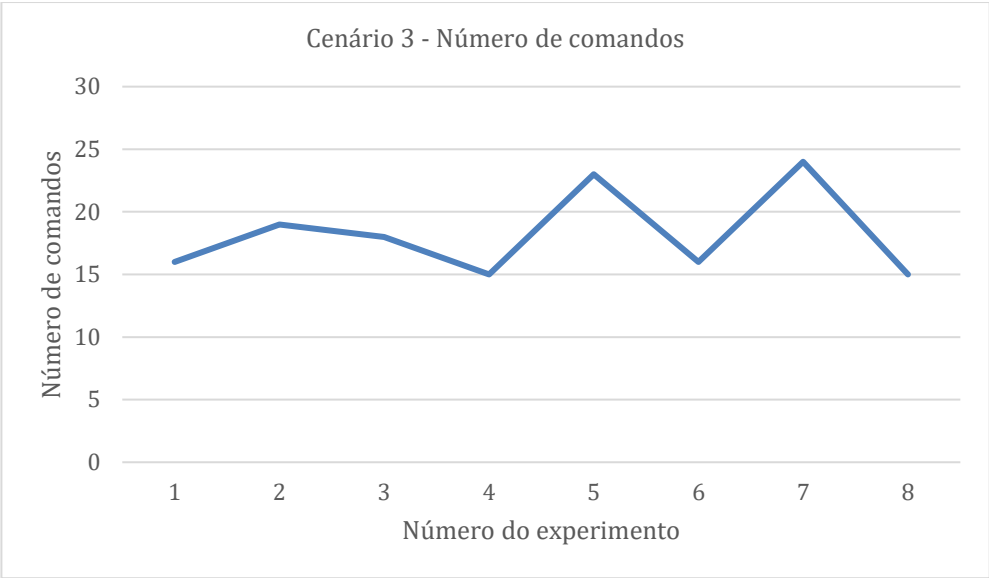
Apêndice C - Resultados individuais dos experimentos do Cenário 3

Tabela 14 - Análise estatística de todos os experimentos realizados no Cenário 3.

CENÁRIO 3 – IHMG (MANAUS - MANAUS)						
	Performance de treinamento		Avaliação da comunicação - *Delay*/RTT (ms)			
N. do Exp.	N. de Comandos	Tempo gasto	Média	σ	Máx	Mín
1	16	01:31	1,7994	0,4542	5,244	1,521
2	19	01:34	2,7639	5,8449	49,828	1,456
3	18	01:39	2,7897	9,1217	92,228	1,438
4	15	01:31	1,7675	0,3616	4,165	1,485
5	23	01:44	1,9356	1,6185	17,023	1,456
6	16	01:31	1,7419	0,3341	4,419	1,45
7	24	01:48	2,008	2,4781	26,115	1,477
8	15	01:29	1,7512	0,3264	4,153	1,532

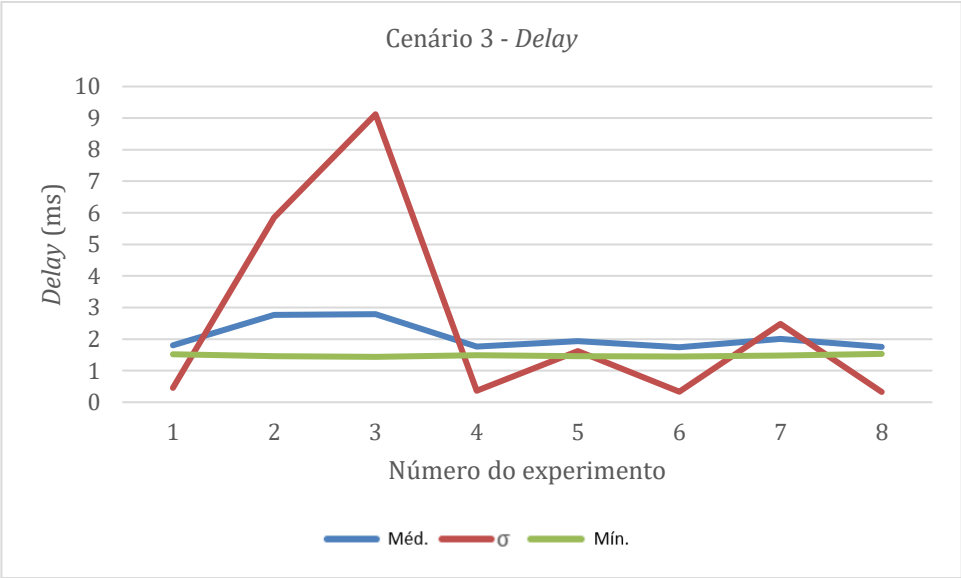


(a)

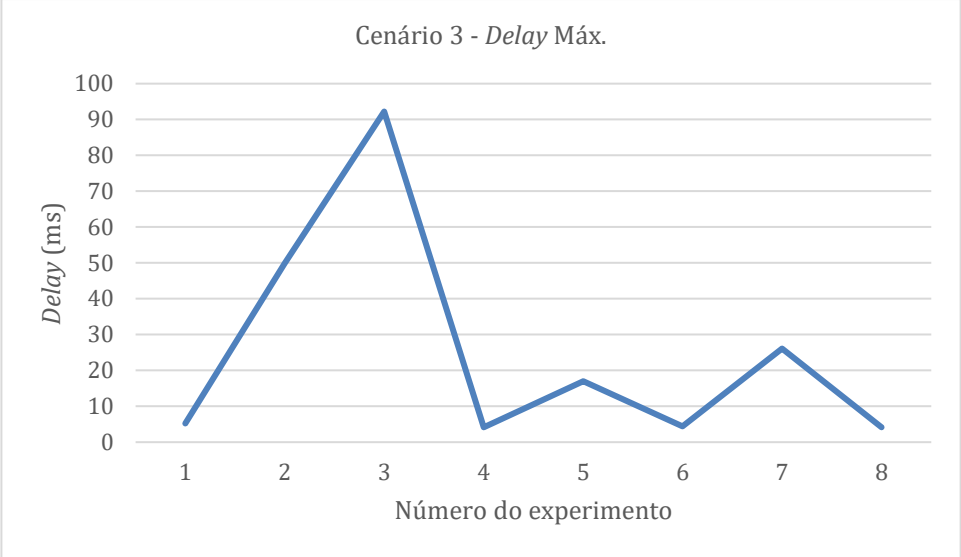


(b)

Figura 39 - (a) Tempo gasto nos experimentos do Cenário 3. (b) Número médio de comandos desse Cenário.



(a)



(b)

Figura 40 - (a) *Delay* médio em todos os experimentos do Cenário 3. (b) *Delay* máximo nos experimentos desse Cenário.

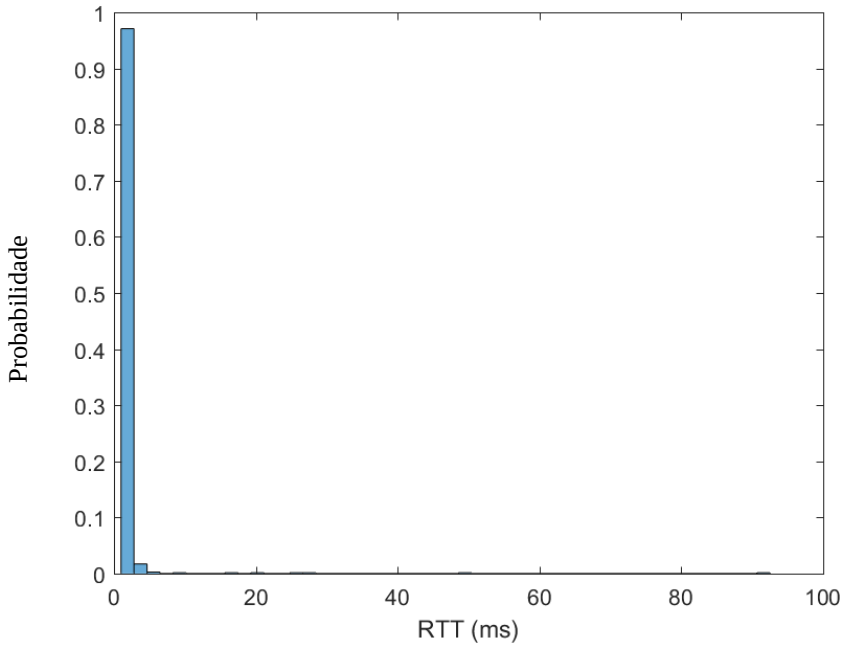
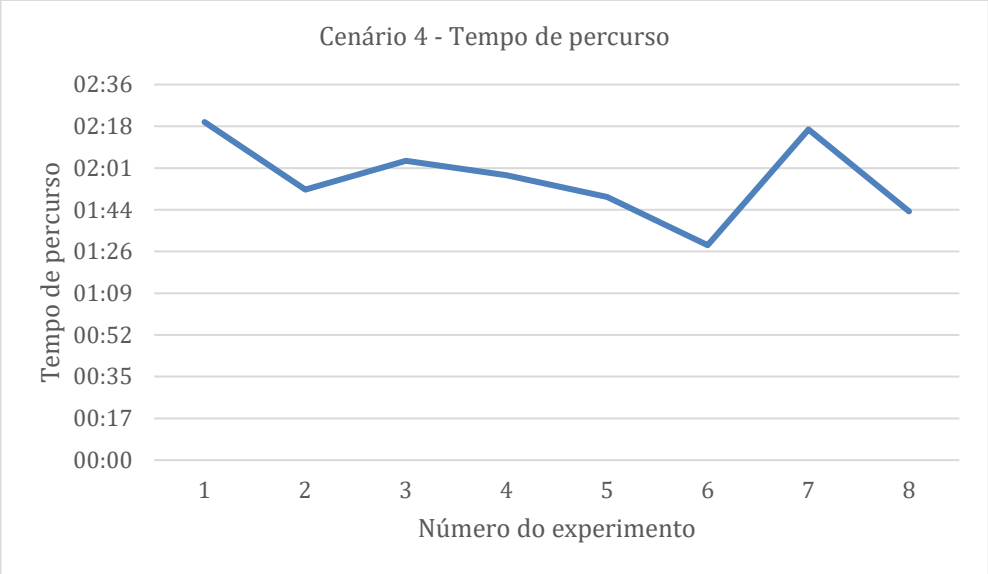


Figura 41 - Histogramas do *Delay* (RTT) dos experimentos do Cenário 3.

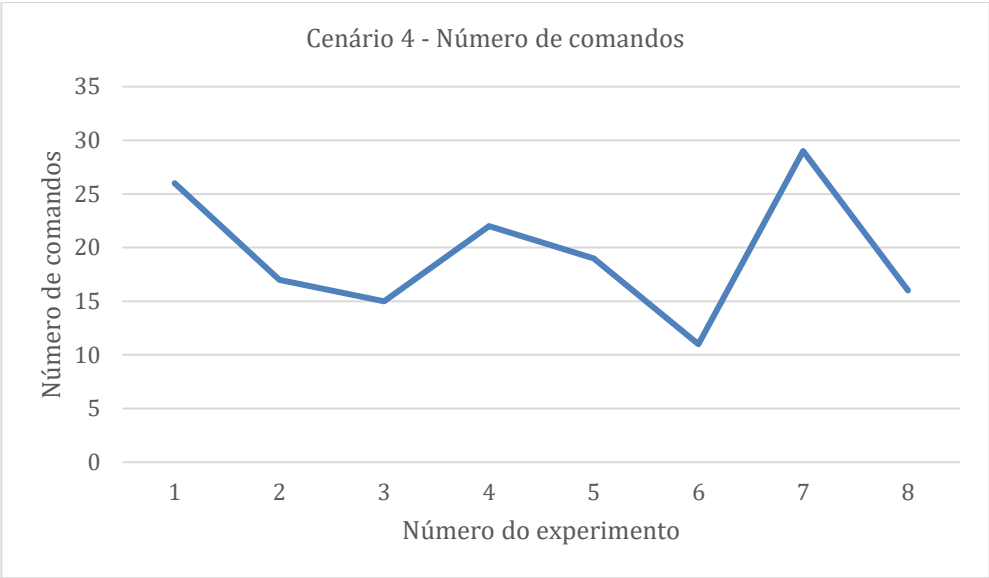
Apêndice D - Resultados individuais dos experimentos do Cenário 4

Tabela 15 - Análise estatística de todos os experimentos realizados no Cenário 4.

CENÁRIO 4 - IHMG (CAMPINAS - MANAUS)						
	Performance de treinamento		Avaliação da comunicação - *Delay*/RTT (ms)			
N. do Exp.	N. de Comandos	Tempo gasto	Média	σ	Máx	Mín
1	26	02:20	86,1277	22,9992	249,751	78,427
2	17	01:52	87,9913	46,5564	532,064	78,666
3	15	02:04	85,1742	28,2493	287,342	78,152
4	22	01:58	83,9134	16,4319	236,511	78,169
5	19	01:49	83,0203	13,2439	202,588	78,355
6	11	01:29	83,0712	8,0533	123,705	76,678
7	29	02:17	87,3602	23,2916	241,712	78,291
8	16	01:43	83,2343	15,0019	220,486	78,581

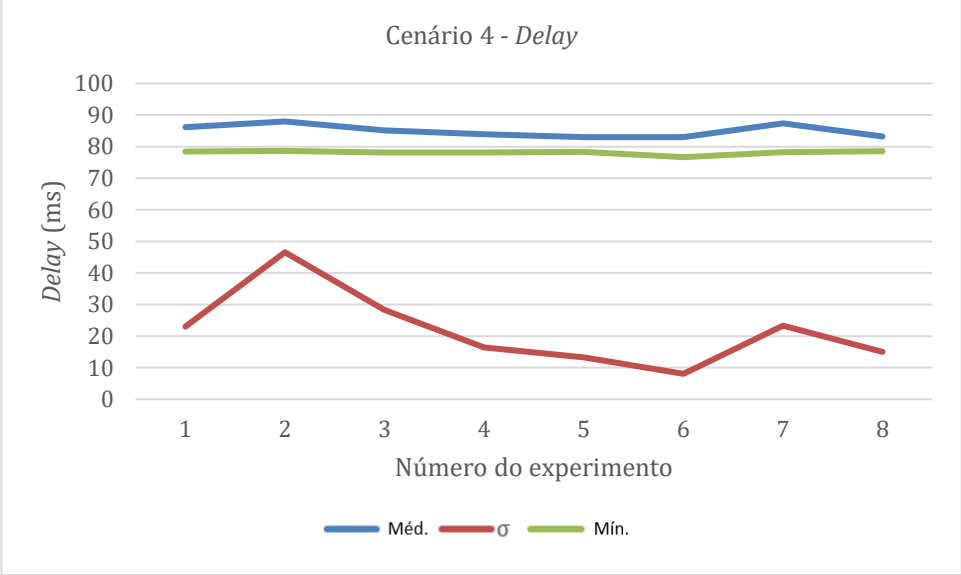


(a)

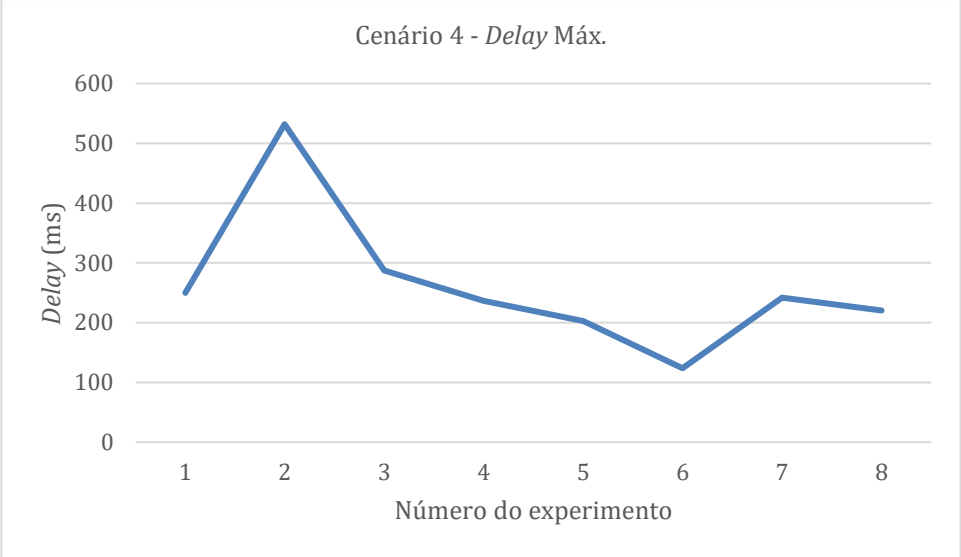


(b)

Figura 42 - (a) Tempo gasto nos experimentos do Cenário 4. (b) Número médio de comandos desse Cenário.



(a)



(b)

Figura 43 - (a) *Delay* médio em todos os experimentos do Cenário 4. (b) *Delay* máximo nos experimentos desse Cenário.

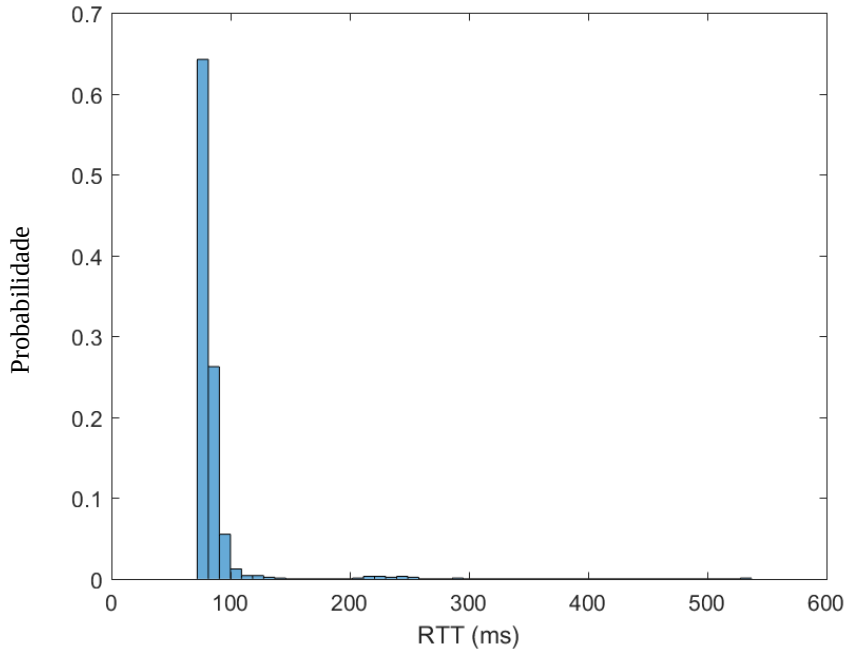
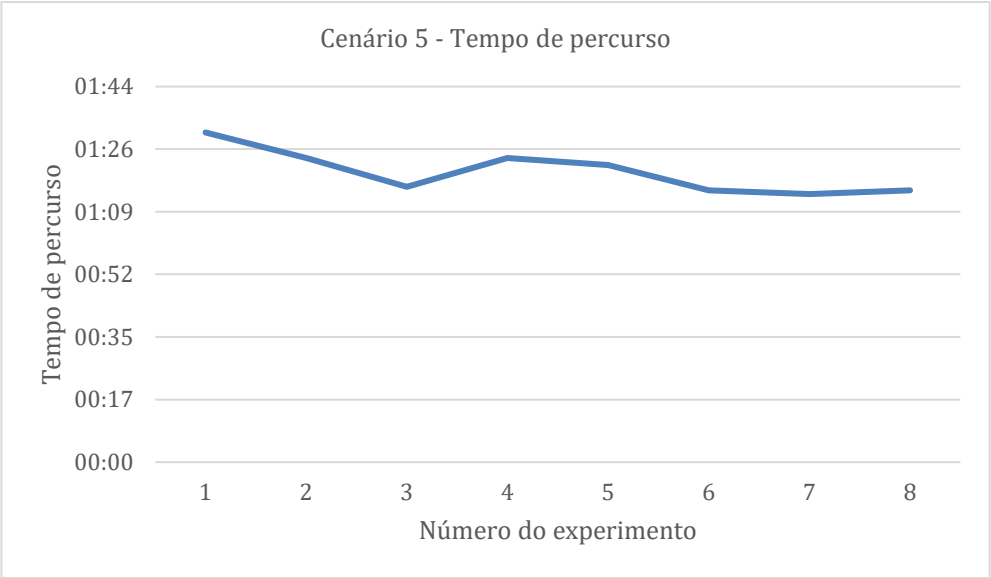


Figura 44 - Histogramas do *Delay* (RTT) dos experimentos do Cenário 4.

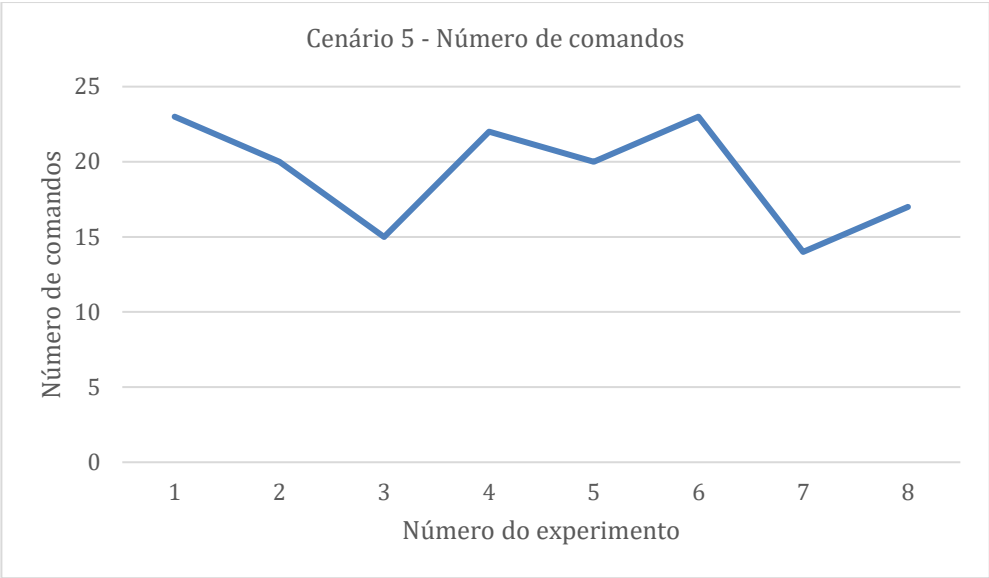
Apêndice E - Resultados individuais dos experimentos do Cenário 5

Tabela 16 - Análise estatística de todos os experimentos realizados no Cenário 5.

CENÁRIO 5 -IHMG (UBERLANDIA - MANAUS)						
Performance de treinamento			Avaliação da comunicação - (ms)			*Delay*/RTT
N. do Exp.	N. de Comandos	Tempo gasto	Média	σ	Máx	Mín
1	23	01:31	131,6572	191,477	191,477	123,458
2	20	01:24	131,0808	7,7424	170,072	123,164
3	15	01:16	129,8602	6,4332	162,597	124,003
4	22	01:24	129,0857	5,171	153,455	123,473
5	20	01:22	129,1722	4,5956	147,673	122,201
6	23	01:15	131,5799	6,2288	151,154	123,884
7	14	01:14	129,6309	6,4507	163,076	122,718
8	17	01:15	129,83	4,8121	144,108	123,962

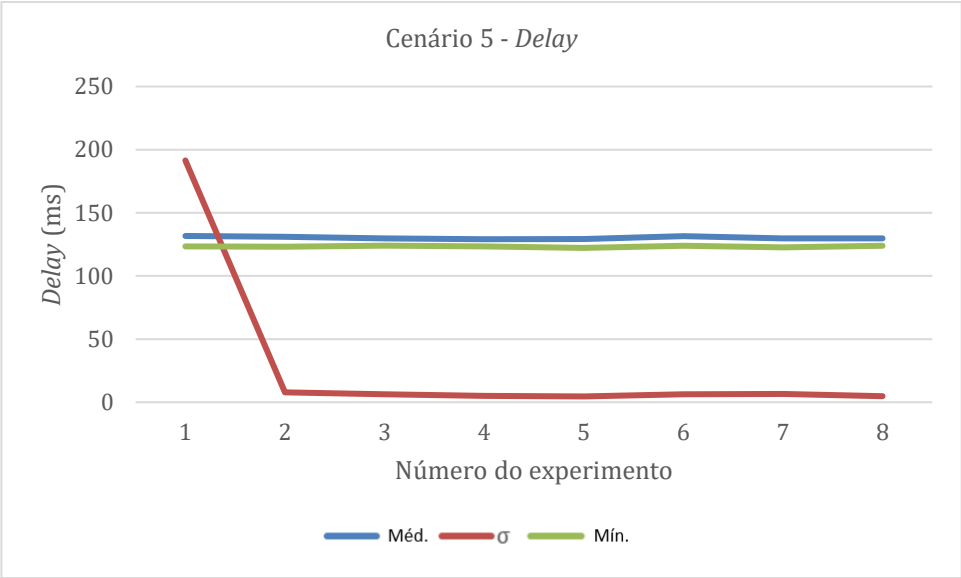


(a)

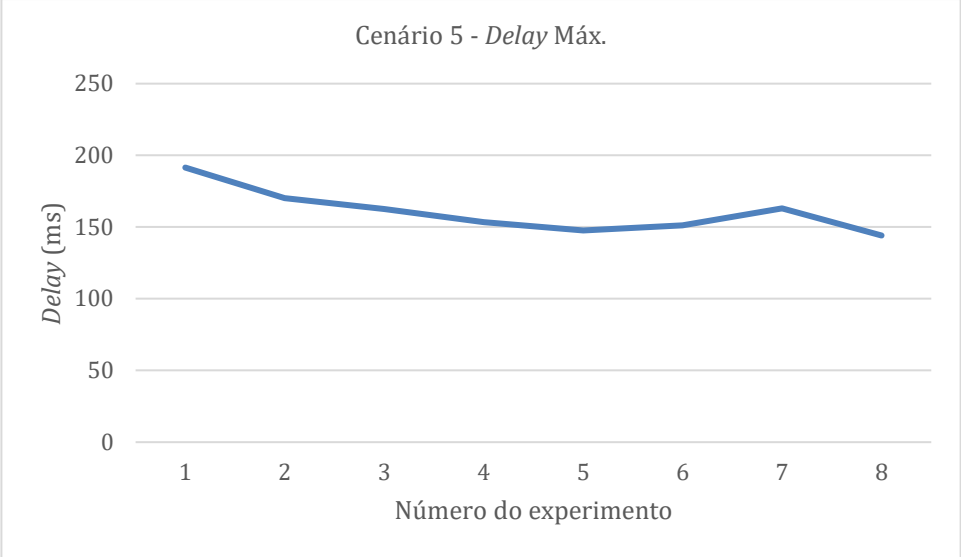


(b)

Figura 45 - (a) Tempo gasto nos experimentos do Cenário 5. (b) Número médio de comandos desse Cenário.



(a)



(b)

Figura 46 - (a) *Delay* médio em todos os experimentos do Cenário 5. (b) *Delay* máximo nos experimentos desse Cenário.

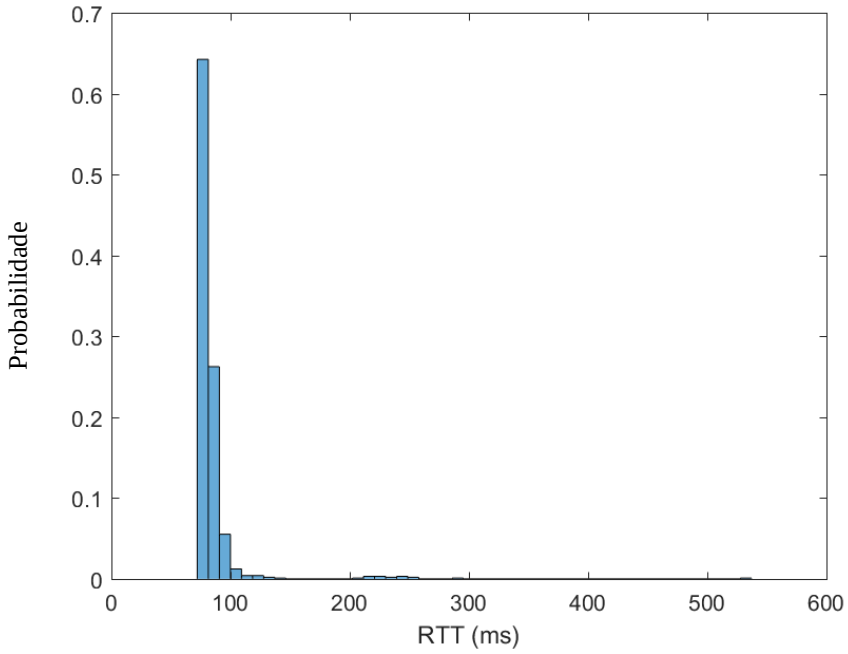


Figura 47 - Histogramas do *Delay* (RTT) dos experimentos do Cenário 5.

Apêndice F - Publicações

Artigos Publicados em Conferências Internacionais

- SILVA, Y. *et al.* Training environment for electric powered wheelchairs using teleoperation through a head mounted display. IEEE International Conference on Consumer Electronics (ICCE), 2018, IEEE. p.1-2.

- LIMA, M; KURKA, P; SILVA, Y; LUCENA JR, VICENTE. Indoor Visual Localization of a Wheelchair Using Shi-Tomasi and KLT. IEEE 30th Canadian Conference on Electrical and Computer Engineering (CCECE 2017), 2017, pp. 1055–1058.

- NAVES, E; BASTOS, T; BOURHIS, G; SILVA, Y. M. L. R.; SILVA, V; LUCENA, V. Virtual and augmented reality environment for remote training of wheelchairs users: Social, mobile, and wearable technologies applied to rehabilitation. 2016 IEEE 18th International Conference on e-Health Networking, Applications and Services (Healthcom), 2016, (pp. 1-4).

Artigo Submetido a Periódico Científico

- SILVA, Y; SIMÕES, W; NAVES, E; BASTOS, T; LUCENA JR., VICENTE. Teleoperation Training Environment for New Users of Electric Powered Wheelchair Based on Multiple Driving Methods. IEEE Access, 2018.