
UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA DE AUTOMAÇÃO E SISTEMAS

Luiz Felipe Baldo Marques

# TELEOPERAÇÃO BILATERAL DE ROBÔS MANIPULADORES INDUSTRIAIS

Florianópolis
2019

Luiz Felipe Baldo Marques

# TELEOPERAÇÃO BILATERAL DE ROBÔS MANIPULADORES INDUSTRIAIS

Dissertação submetida ao Programa de Pós-graduação em Engenharia de Automação e Sistemas para obtenção do Grau de Mestre em Engenharia de Automação e Sistemas.
Orientador: Edson R. De Pieri, PhD
Coorientador: Henrique Simas, Dr. Eng.

Florianópolis
2019

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

Marques, Luiz Felipe Baldo
Teleoperação bilateral de robôs manipuladores industriais / Luiz Felipe Baldo Marques ; orientador, Edson Roberto de Pieri, coorientador, Henrique Simas, 2019.
97 p.

Dissertação (mestrado) - Universidade Federal de Santa Catarina, Centro Tecnológico, Programa de Pós-Graduação em Engenharia de Automação e Sistemas, Florianópolis, 2019.

Inclui referências.

1. Engenharia de Automação e Sistemas. 2. Teleoperação bilateral. 3. Robôs industriais. 4. Restrições virtuais. I. Pieri, Edson Roberto de. II. Simas, Henrique. III. Universidade Federal de Santa Catarina. Programa de Pós Graduação em Engenharia de Automação e Sistemas. IV. Título.

Luiz Felipe Baldo Marques

**Teleoperação Bilateral de Robôs Manipuladores Industriais**

O presente trabalho em nível de mestrado foi avaliado e aprovado por banca examinadora composta pelos seguintes membros:

Prof. Daniel Martins, Dr.
Universidade Federal de Santa Catarina

Prof. Ubirajara Franco Moreno, Dr.
Universidade Federal de Santa Catarina

Prof. Marcelo Ricardo Stemmer, PhD.
Universidade Federal de Santa Catarina

Certificamos que esta é a **versão original e final** do trabalho de conclusão que foi julgado adequado para obtenção do título de Mestre em Engenharia de Automação e Sistemas.

Assinado de forma digital por Werner Kraus Junior:53108523953
Dados: 2019.10.22 15:17:08 -03'00'

Werner Kraus Junior, PhD.
Coordenador do Programa

Documento assinado digitalmente
Edson Roberto de Pieri
Data: 22/10/2019 15:02:46-0300
CPF: 002.042.498-13

Edson R. De Pieri, PhD.
Orientador

Florianópolis, 2019.

## RESUMO

Neste trabalho é proposto um modelo de simulação de teleoperação bilateral de robôs manipuladores industriais. A teleoperação bilateral consiste em um dispositivo mestre que comanda um dispositivo escravo remotamente, sendo que escravo é capaz de fornecer uma percepção do ambiente de operação ao mestre. Atualmente os problemas de teleoperação bilateral de robôs são solucionados em contextos onde as propriedades das tecnologias no mundo real, como velocidade de comunicação e limitação de programação, não são totalmente consideradas . A presente dissertação aborda então a teleoperação bilateral considerando as limitações de robôs industriais, e tem por base o desenvolvimento de dois modelos de simulação, um para validação de um conceito de restrições virtuais e outro para analisar o comportamento de robôs industriais teleoperados. Primeiramente é simulado um modelo de teleoperação no domínio do tempo contínuo, no qual um robô mestre comanda um robô escravo. Este último opera em um ambiente virtual que impõe restrições de movimento, sendo que a percepção dessas restrições por parte do mestre é feita através de realimentação de força, obtida através de uma impedância virtual. A segunda simulação conta com o modelo no tempo discreto de dois robôs industriais idênticos. O mestre recebe comados de movimento por meio de uma admitância virtual alimentada por medições de um sensor de força real. Nesta segunda simulação é o mestre quem opera em um ambiente virtual, estimando a interação do escravo com as restrições. O movimento do mestre é copiado então pelo escravo através de um sistema de comunicação de dados com atraso, calculado de acordo com o tempo de resposta do escravo. Por fim foi concluído que o sistema final proposto possui maiores vantagens quando operado com pequenas forças e velocidades, sendo que para magnitudes relativamente maiores, o sistema tende a apresentar maiores atrasos de comunicação, perdendo a precisão e a confiabilidade.



**Palavras-chaves:** teleoperação bilateral, robôs industriais, restrições virtuais.

**ABSTRACT**

In this work it is proposed a simulation model of bilateral teleoperation of industrial robot manipulator. Bilateral teleoperation consists of a master device that commands a slave device remotely, being that the slave is able to provide a perception of the operation environment to the master. Actually the problems in robots bilateral teleoperation are solved in contexts where real-world technology properties, like communication speed and programming limitations are not fully considered. The present thesis then approaches the bilateral teleoperation considering the limitations of industrial robots, and is based on the development of two simulation models, one to validate a concept of virtual constraints and the other to analyze the behavior of teleoperated industrial robots. Firstly is simulated a teleoperation model in the continuous time domain, in which a master robot commands a slave robot. The latter one operates in a virtual environment that imposes movement restrictions, and the perception of these restrictions by the master is made through of force feedback, obtained through a virtual impedance. The second simulation features the discrete time model of two identical industrial robots. The master receives motion commands through a virtual admittance fed by measurements of a real force sensor. In this second simulation is the master who operates in a virtual environment, estimating the interaction of the slave with the constraints. The movement of the master is then copied by the slave via a delayed data communication system, calculated according to the slave's response time. Finally it was concluded that the proposed final system has greater advantages when operated with small forces and speeds, and for relatively larger magnitudes, the system tends to present greater communication delays, losing accuracy and reliability.

**Keywords:** bilateral teleoperation, industrial robots, virtual constraints.

**SUMÁRIO**


| | | |
|---|---|---|
| | **Lista de símbolos** | **i** |
| | **Lista de tabelas** | **iii** |
| | **Lista de ilustrações** | **iv** |
| **1** | **INTRODUÇÃO** | **1** |
| **1.1** | **Revisão bibliográfica** | **2** |
| **1.2** | **Problemática** | **7** |
| **1.3** | **Proposição** | **8** |
| **1.3.1** | **Objetivos** | **9** |
| **1.4** | **Escopo do trabalho** | **9** |
| **1.5** | **Organização do trabalho** | **10** |
| **2** | **ROBÔS MANIPULADORES INDUSTRIAIS** | **11** |
| **2.1** | **Modelagem cinemática de robôs manipuladores** | **12** |
| **2.1.1** | **Cinemática direta** | **13** |
| **2.1.2** | **Cinemática inversa** | **14** |
| **2.2** | **Princípio básico de estática de robôs** | **15** |
| **2.3** | **Dispositivos hápticos** | **16** |
| **3** | **CONTROLE DE FORÇA E POSIÇÃO** | **19** |
| **3.1** | **Ambiente de operação** | **19** |
| **3.1.1** | **Ambiente virtual** | **20** |
| **3.1.2** | **Monitoramento** | **21** |
| **3.2** | **Controle de impedância/admitância** | **23** |
| **3.2.1** | **Admitância do mestre** | **24** |
| **3.2.2** | **Impedância do ambiente** | **24** |
| **4** | **SIMULAÇÃO DE TELEOPERAÇÃO HÁPTICA** | **25** |
| **4.1** | **Modelagem cinemática do mestre e escravo** | **26** |
| **4.2** | **Fatores de escala e transformação** | **29** |
| **4.3** | **Ambiente de simulação** | **30** |
| **4.4** | **Resultados e considerações** | **32** |
| **5** | **TELEOPERAÇÃO DE UM ROBÔ INDUSTRIAL** | **37** |
| **5.1** | **Robô ABB IRB1600** | **37** |
| **5.1.1** | **Controlador IRC5** | **37** |
| **5.1.2** | **RAPID e RobotStudio** | **39** |
| **5.2** | **Sensor de força** | **40** |
| **5.2.1** | **Aquisição de dados** | **41** |
| **5.3** | **Topologia de comunicação e controle** | **43** |

**5.4 Operação do manipulador mestre . . . . . 45**
**5.5 Comando do manipulador escravo . . . . . 46**
**5.6 Comunicação . . . . . 47**
**5.7 Simulações e Resultados . . . . . 48**
**5.7.1 Parâmetros de simulação . . . . . 48**
**5.7.2 Manipulação do mestre sem realimentação . . . . . 50**
**5.7.3 Teleoperação sem realimentação . . . . . 53**
**5.7.4 Teleoperação com realimentação . . . . . 57**
**5.7.5 Análise do atraso . . . . . 62**

**6 CONCLUSÃO . . . . . 67**
**6.1 Contribuição . . . . . 68**
**6.2 Trabalhos futuros . . . . . 69**

**REFERÊNCIAS . . . . . 71**

**APÊNDICES . . . . . 75**

**A CÓDIGO PARA LEITURA DO SENSOR DE FORÇA JR3 EM LUNGUAGEM C++ . . . . . 77**

**B MATRIZES JACOBIANAS DE POSIÇÃO DOS ROBÔS DE TRÊS GRAUS DE LIBERDADE . . . . . 81**
**B.1 Dispositivo RRR . . . . . 81**
**B.2 SCARA . . . . . 81**

**C CÓDIGO DA CENTRAL DE CONTROLE EM LINGUAGEM PYTHON . . 83**

**D VARIÁVEIS DO MODELO DE OPERAÇÃO DO MANIPULADOR MESTRE DA SIMULAÇÃO EM TEMPO DISCRETO . . . . . 87**

**E RESULTADOS DAS SIMULAÇÕES 1 E 2 UTILIZANDO O ROBOTSTUDIO 89**

**F ESTIMAÇÃO DO *CYCLETIME* POR MÍNIMOS QUADRADOS ORDINÁRIOS (MQO) . . . . . 97**

## LISTA DE SÍMBOLOS

$A_j^i$ Matriz de transformação homogênea da posição e orientação do sistema de coordenada $j$ em relação ao sistema de coordenadas $i$.

$R_j^i$ Matriz de rotação do sistema de coordenada $j$ em relação ao sistema de coordenadas $i$.

$p_j^i$ Vetor de posição do sistema de coordenada $j$ em relação ao sistema de coordenadas $i$.

$p_{x,y,z}$ Coordenadas $x$, $y$ e $z$ do ponto $p$ do efetuador final do manipulador em relação à base.

$\alpha, \beta, \gamma$ Ângulos de Euler do tipo ZYX do efetuador final do manpulador.

$q$ Vetor de variáveis de junta do manipulador.

$P$ Vetor de postura do efetuador final no sistema global de coordenadas.

$\Theta$ Vetor de orientação do efetuador final.

$v$ Vetor de velocidades do efetuador final.

$v_{x,y,z}$ Vetor de velocidades de translação do efetuador final nas direções $x$, $y$ e $z$.

$\omega_{x,y,z}$ Vetor de velocidades de rotação do efetuador final nas direções $x$, $y$ e $z$.

$J$ Matriz Jacobiana.

$\tau$ Vetor de forças e torques das juntas do manipulador.

$f$ Vetor de forças e torques do efetuador final espaço de operação do manipulador.

$p_o$ Origem do sistema de coordenadas d ambiente virtual de operação.

$h_z$ Altura do cilindro no eixo $z$.

$r_{xy}$ Raio do ambiente virtual de operação.

$p_r$ Vetor de posição do efetuador final em relação ao centro do ambiente virtual de operação.

$\sigma$ Ângulo do efetuador final em relação ao eixo $x$ do ambiente virtual de operação.

$p_{lim}$ Vetor de posição limite relativa do ambiente virtual de operação no plano $xy$.

$x_{lim}$ Limite no eixo $x$ relativo do ambiente virtual de operação.

$y_{lim}$ Limite no eixo $y$ relativo do ambiente virtual de operação.

$z_{min}$ Limite no eixo $z$ inferior do ambiente virtual de operação.

$z_{max}$ Limite no eixo $z$ superior do ambiente virtual de operação.

$M$ Constante de massa.

$B$ Constante de amortecimento.

$K$ Constante de rigidez.

$F_{int}$ Força de interação.

$B_m$ Constante de amortecimento do mestre.

$B_a$ Constante de amortecimento do ambiente virtual de operação.

$F_m$ Força de estímulo do mestre.

$F_a$ Força de interação com o ambiente virtual de operação.

$F_s$ Leitura de força no sensor.

$F_x$, $F_y$, $F_z$ Componentes da leitura de força do sensor nas direções $x$, $y$ e $z$.

$\theta$ Ângulo de junta rotativa.

$d$ Distância de junta prismática.

$\psi_{1,2}$ Ângulos auxiliares para cálculo da cinemática inversa do SCARA.

$t_s$ *Cycletime* de operação do mestre.

$V_m$ Velocidade do mestre no domínio da frequência.

$v_d$ Velocidade desejada do mestre.

$v_p$ Velocidade parametrizada do comando *MoveL*

$\lambda$ Vetor de parâmetros do modelo do *cycletime*.

$\hat{\lambda}$ Vetor estimado de parâmetros do modelo do *cycletime*.

$\lambda_0$ Coeficiente linear no modelo do *cycletime*.

$\lambda_1$ Parâmetro relativo à saída anterior no modelo do *cycletime*.

$\lambda_2$ Parâmetro relativo à velocidade desejada no mestre do modelo do *cycletime*.

$\lambda_2$ Parâmetro relativo à velocidade base de deslocamento do mestre do modelo do *cycletime*.

$Y$ Vetor de saídas reais do *cycletime*.

$\hat{Y}$ Vetor de saídas estimadas do *cycletime*.

$X$ Matriz de regressores do sistema do *cycletime* a ser estimado.

$s^2$ Variância da estimação de $\hat{Y}$ em relação a $Y$.

## LISTA DE TABELAS

Tabela 1 – Parâmetros D-H do mestre. . . . . 27
Tabela 2 – Parâmetros D-H do escravo. . . . . 27
Tabela 3 – Dimensões das zonas de operação do escravo. . . . . 32
Tabela 4 – Limitações cinemáticas do Robô IRB1600. . . . . 38
Tabela 5 – Especificações técnicas do sensor de força-torque. . . . . 42
Tabela 6 – Quadro de dados dos filtros de medição. . . . . 43
Tabela 7 – Canais de comunição do sensor JR3. . . . . 43
Tabela 8 – Variáveis de operação do mestre. . . . . 46
Tabela 9 – Parâmetros das simulações. . . . . 49
Tabela 10 – Resultado da estimação de *cycletime*. . . . . 64

## LISTA DE ILUSTRAÇÕES


Figura 1 – Produção científica em teleoperação bilateral de robôs. . . . . 3
Figura 2 – Sistema de teleoperação do tipo *4-channel*. . . . . 4
Figura 3 – Transformação de onda para sistemas *4-channel*. . . . . 5
Figura 4 – Teleoperação mediada por modelo de referência. . . . . 6
Figura 5 – Teleoperação com realimentação virtual. . . . . 7
Figura 6 – Robô industrial com 7 DOFs. . . . . 11
Figura 7 – Transformação de sistemas de coordenadas. . . . . 13
Figura 8 – Cinemática inversa e direta. . . . . 15
Figura 9 – Dispositivo háptico Phantom Omni. . . . . 16
Figura 10 – Controle de impedância/admitância do sistema de teleoperação. . . . . 19
Figura 11 – Robô no ambiente de operação virtual. . . . . 20
Figura 12 – Geometria do ambiente de operação virtual. . . . . 21
Figura 13 – Monitoramento em x e y. . . . . 22
Figura 14 – Monitoramento em z. . . . . 22
Figura 15 – Sistema mecânico de segunda ordem. . . . . 23
Figura 16 – Sistemas de Teleoperação: a) Unilateral; b) Bilateral. . . . . 25
Figura 17 – Modelos em CAD do mestre e escravo. . . . . 26
Figura 18 – Configuração do mestre. . . . . 26
Figura 19 – Configuração do escravo. . . . . 27
Figura 20 – Arquitetura proposta de um sistema ideal de teleoperação bilateral. . . . . 30
Figura 21 – Ilustração de manipulação do dispositivo mestre. . . . . 31
Figura 22 – Zonas de operação do escravo. . . . . 31
Figura 23 – Deslocamento e forças de reação no eixo X. . . . . 33
Figura 24 – Deslocamento e forças de reação no eixo y. . . . . 34
Figura 25 – Deslocamento e forças de reação no eixo z. . . . . 34
Figura 26 – Torques nas juntas do mestre. . . . . 34
Figura 27 – Resposta do mestre sem atraso. . . . . 35
Figura 28 – Resposta do mestre com 100 ms de atraso. . . . . 35
Figura 29 – Resposta do mestre com 500 ms de atraso. . . . . 36
Figura 30 – Resposta do mestre com 1 s de atraso. . . . . 36
Figura 31 – Robô IRB1600. . . . . 37
Figura 32 – Área de trabalho do Robô IRB1600. . . . . 38
Figura 33 – Controlador IRC5. . . . . 39
Figura 34 – Exemplo de programa em RAPID. . . . . 39
Figura 35 – Área de trabalho do ABB RobotStudio. . . . . 41
Figura 36 – Sensor de força/torque JR3. . . . . 41
Figura 37 – Manípulo de operação. . . . . 42
Figura 38 – Modelo de teleoperação de robô industrial. . . . . 44
Figura 39 – Sistema de teleoperação proposto. . . . . 44
Figura 40 – Fluxograma de operação do mestre. . . . . 45

Figura 41 – Comunicação entre os subsistemas de teleoperação. . . . . 47
Figura 42 – Aplicação de forças no sensor. . . . . 49
Figura 43 – Simulação 1 - Posição no eixo x. . . . . 50
Figura 44 – Simulação 1 - Velocidade no eixo x. . . . . 51
Figura 45 – Simulação 1 - Relação entre força e atraso. . . . . 51
Figura 46 – Simulação 2 - Posição no eixo x. . . . . 52
Figura 47 – Simulação 2 - Velocidade no eixo x. . . . . 52
Figura 48 – Simulação 2 - Relação entre força e atraso. . . . . 53
Figura 49 – Simulação 3: Posição do Mestre. . . . . 53
Figura 50 – Simulação 3: Posição do Escravo. . . . . 54
Figura 51 – Simulação 3: Velocidades do Mestre. . . . . 54
Figura 52 – Simulação 3: Velocidades do Escravo. . . . . 54
Figura 53 – Simulação 3: Erro de seguimento de posição. . . . . 55
Figura 54 – Simulação 3: Relação entre força e atraso. . . . . 55
Figura 55 – Simulação 4: Velocidades do Mestre. . . . . 56
Figura 56 – Simulação 4: Velocidades do Escravo. . . . . 56
Figura 57 – Simulação 4: Erro de seguimento de posição. . . . . 57
Figura 58 – Simulação 4: Relação entre força e atraso. . . . . 57
Figura 59 – Diagrama de blocos da realimentação de força virtual. . . . . 58
Figura 60 – Resposta da admitância à entrada degrau (estável). . . . . 58
Figura 61 – Simulação 5: Posições do mestre e escravo no eixo z. . . . . 59
Figura 62 – Simulação 5: Velocidades do mestre e escravo no eixo z. . . . . 59
Figura 63 – Simulação 5: Erro no seguimento de posição no eixo z. . . . . 60
Figura 64 – Simulação 5: Atraso do mestre e escravo. . . . . 60
Figura 65 – Resposta da admitância à entrada degrau (instável). . . . . 61
Figura 66 – Simulação 6: Posições do mestre e escravo no eixo z. . . . . 61
Figura 67 – Simulação 6: Velocidades do mestre e escravo no eixo z. . . . . 62
Figura 68 – Simulação 6: Erro no seguimento de posição no eixo z. . . . . 62
Figura 69 – Simulação 6: Atraso do mestre e escravo. . . . . 63

# 1 INTRODUÇÃO

Robôs manipuladores foram introduzidos na indústria na segunda metade do século XX, sendo que sua invenção pode ser datada em 1954, quando Geroge Devol registrou a patente do *Programmed Article Transfer* (HÄGELE; NILSSON; PIRES, 2008). Devol então se aliou a Joseph Engelberg, e fundaram a primeira empresa de robótica chamada Unimation, em 1961, ano em que também instalaram o primeiro robô na indústria, em uma planta da General Motors. Na época, essas máquinas eram usadas basicamente em tarefas de manipulação e solda ponto (HÄGELE; NILSSON; PIRES, 2008).

No cenário atual a robótica cresceu, e ainda cresce, segundo a pesquisa realizada pela Federação Internacional de Robótica (*International Federation of Robotics*) IFR (2018). Do ano de 2016 para 2017 foi obsevado um crescimento de 30% no fornecimento de robôs industriais, ou seja de 294 mil unidades em 2016 para 381 mil unidades em 2017. Ainda estima-se que até 2021 exista uma taxa média de crescimento de 14% ao ano no fornecimento de unidades (IFR, 2018). Atualmente os países orientais lideram a produção e o fornecimento de robôs industriais, sendo que em 2017, a China liderou o *ranking* de fornecedores, com 137,9 mil unidades fornecidas, seguida pelo Japão com 45,6 mil unidades, República da Coreia com 39,7 mil unidades, Estados Unidos com 33,2 mil unidades e Alemanha com 21,4 mil unidades (IFR, 2018).

Ainda na mesma pesquisa da IFR (2018), foram identificados os cinco segmentos industriais que mais adquiriram robôs industriais nos últimos anos: automotivo, produtos eletrônicos, metalmecânico, produtos plásticos e químicos, alimentos e bebida. No ano de 2017, foi estimado que cada um desses segmentos adquiriu 126 mil unidades (22% a mais que 2016), 121 mil unidades (33% a mais que 2016), 45 mil unidades (55% a mais que 2016), 21 mil unidades (9% a mais que 2016) e 10 mil unidades (19% a mais que 2016), respectivamente.

Em outro cenário industrial, na década de 1940 começou a se utilizar o termo de teleoperação, que consistia na operação de sistemas de forma remota (HOKAYEM; SPONG, 2006). Esse tipo de tecnologia tem evoluído em função dos avanços na computação e comunicação de dados, e foi utilizado inicialmente em operação de usinas nucleares e outras aplicações onde se tornava perigosa ou inviável a presença de um operador humano (MIYAZAKI et al., 1986). E como os robôs foram concebidos com o intuito de reproduzir o comportamento humano, não demorou muito para que se começasse a estudar a possibilidade do uso de robôs em diversas aplicações de teleoperação.

Atualmente a teleoperação de robôs é utilizada em aplicações tais como exploração sapcial e submarina, cirurgias, manipulação em ambientes perigosos, manipulação de objetos delicados e/ou perigosos, manipulação de cargas pesadas, aplicações com drones, e diversas outras situações em que um humano poderia ser substituido por um robô (HÄGELE; NILSSON; PIRES, 2008). Desta forma, cientistas e empresas de todo o mundo vêm estudando e desenvolvendo cada vez mais as tecnologias de teleoperação, buscando sempre melhorar a sua confiabilidade e desempenho.

Uma vez que a maioria dos sistemas de teleoperação de robôs manipuladores envolve a interação física do manipulador ou do efetuador final com o ambiente de operação, é importante que exista uma forma de monitorar como o robô está interagindo com este ambiente. Assim se faz necessário uma realimentação no sistema de teleoperação, surgindo então o conceito de teleoperação bilateral.

No contexto da teleoperação bilateral ocorre uma reação por parte do manipulador escravo em relação à sua interação com o ambiente de operação, realimentando o operador do mestre de tal forma

que permita a ele perceber, de forma exata ou relativa, o que o escravo "sente"durante a sua operação (ANDERSON; SPONG, 1989).

A teleoperação já se tornou uma das aplicações mais clássicas em robótica, como visto em Hokayem e Spong (2006) e Niemeyer, Preusche e Hirzinger (2008). Porém isso não significa que não existam oportunidades de melhoria nesse tipo de sistema. Portanto, integrar equipamentos de alta tecnologia é uma atividade extremamente interessante dos pontos de vista de engenharia e pesquisa.

Neste trabalho são propostos modelos de simulação para sistemas robóticos de teleoperação bilaterais. A contribuição da dissertação está no desenvolvimento de dois modelos desenvolvidos de acordo com critérios de desempenho baseados na variável de tempo. Primeiramente foi feita uma simulação de teleoperação bilateral com modelos virtuais de robôs ideiais, cuja matemática utilizada no comando e nas interações opera no tempo contínuo. Este primeiro modelo de simulação contou com um ambiente virtual de operação do dispositivo escravo, cuja interação com esse ambiente resulta em reações de força que são transmitidas ao robô mestre.

Um segundo modelo de simulação de um sistema de teleoperação bilateral é proposto com conceito similar à primeira simulação, utilizando um ambiente específico de programação para robôs industriais. Para o segundo modelo foram utilizados modelos virtuais de dois robôs de cadeia aberta com seis graus de liberdade da fabricante ABB, modelo IRB1600. Um sensor de força e torque foi usado e incorporado ao sistema e ao modelo do sistema, através do qual um operador humano comanda o dispositivo mestre. Nessa simulação porém, toda a matemática foi incorporada de forma discreta no tempo.

Os modelos desenvolvidos possibilitam a percepção do comportamento de um sistema de teleoperação bilateral de robôs que possui, em sua arquitetura, a implementação de alguns conceitos já estudados, como a utilização de ambientes virtuais e o controle de impedância, que são explicados na sequência deste trabalho. Esses conceitos foram então aplicados a um modelo de teleoperação baseado nas tecnologias já existentes de controle de robôs industriais. Portanto esse trabalho demonstra uma avanço na área, uma vez que simula uma previsão dos efeitos da aplicação de conceitos de teleoperação em robôs reais.

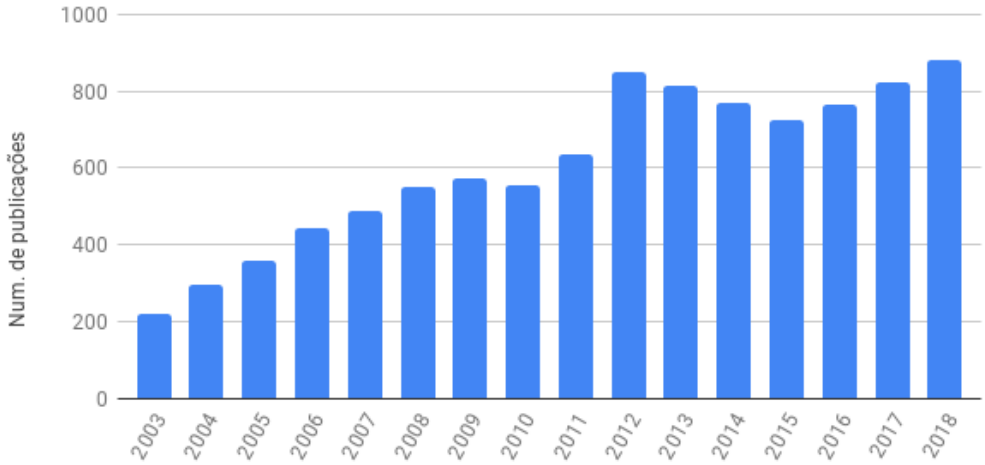
## 1.1 REVISÃO BIBLIOGRÁFICA

O desenvolvimento de pesquisas e o material científico acerca do assunto de teleoperação tem se mostrado constante no meio acadêmico e técnico nos últimos anos. Em uma rápida pesquisa no site do Google Acadêmico, utilizando as palavras chave *"bilateral+teleoperation+robot"*, se chega ao resultado de 7.930 artigos entre outras publicações científicas, entre os anos de 2003 e 2018. O crescimento da produção de material científico sobre o assunto pode ser visto na Figura 1.

Obviamente todo esse material abrange vários assuntos e aspectos que vão além do abordado neste trabalho. A teleoperação de robôs está presente em ambientes não industriais, como a utilização de robôs móveis, drones, robôs cirúrgicos, robôs de serviço etc (NIEMEYER; PREUSCHE; HIRZINGER, 2008).

Nota-se nesse levantamento então, que houve um significativo crescimento da atividade acadêmica no assunto em questão, que pode ser atribuído a vários motivos, como o maior interesse comercial em sistemas de teloperação, a popularização e barateamento dos robôs disponníeis no mercado e

Figura 1 – Produção científica em teleoperação bilateral de robôs.




Fonte: Google Acadêmico (2019).

a evolução dos sistemas computacionais e de processamento digital. Portanto os clássicos problemas e obstáculos da teleoperação em robótica vem sendo resolvidos e a utilização de tais sistemas torna-se cada vez mais viável.

No que se refere à teleoperação de robôs industriais, alguns dos temas que têm despertado mais interesse dos pesquisadores são:

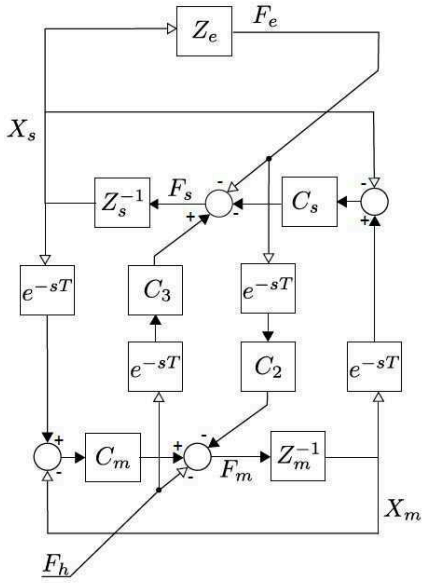
- Controle de força transmitida com atrasos no tempo (COLONNESE; OKAMURA, 2017; HECK et al., 2018; PECLY; SOUZA; HASHTRUDI-ZAAD, 2018; TASHIRO et al., 2018);
- Transparência em sistemas com atraso no tempo (PERERA; ABEYKOON, 2014; REBELO; SCHIELE, 2015);
- Estabilidade em sistemas com atraso no tempo (CHEN et al., 2018; HUANG et al., 2017; REBELO; SCHIELE, 2015);
- Controle de força em ambientes desconhecidos (LIU et al., 2017; RAHIMIFARD; TALEBI; MOHAMMADI, 2016; ZHAO et al., 2017);
- Otimização na comunicação de teleoperação (DALVAND; NAHAVANDI, 2014; TIAN; GAO; ZHANG, 2017).

Um dos maiores problemas encontrados nos sistemas de teleoperação é o atraso na comunicação, que pode levar a instabilidade tanto no controle de força como na impedância e forças transmitidas do escravo para o mestre (HU et al., 2011). Em função disso, diversos autores realizaram pesquisas e desenvolveram métodos matemáticos e computacionais para tentar compensar os atrasos na transmissão de informações em sistemas bilateriais.

Huang et al. (2017) propôs um modelo para sistemas de teleoperação baseado em amostragem e atrasos no tempo, o qual pode ser controlado utilizando *buffers* e um controlador PD. Também foram considerados os critérios de estabilidade baseados no método funcional de Lyapunov-Krasovskii. O sistema proposto então foi testado e validado em um sistema de teleoperação de um grau de liberdade.

Rebelo e Schiele (2015) por sua vez, apresentam uma análise do desempenho de um sistema de teloperação bilateral utilizando uma arquitetura de *4-channel* (quatro canais) de comunicação, para sistemas com escravos controlados por impedância. Nesse tipo de sistema, as variáveis transmitidas bilateralmente entre mestre e escravo são velocidade e força, com as quais é possível obter as informações para compor o modelo de impedância/admitância dos sistemas. A arquitetura desse tipo de sistema pode ser visto na Figura 2.

Figura 2 – Sistema de teleoperação do tipo *4-channel*.




Fonte: Rebelo e Schiele (2015).

Na Figura 2 tem-se que $C_i$ são os controladores utilizados na transmissão de informação, $Z_i$ são as impedâncias dos susbsistemas, $X_i$ são as posições dos manipuladores e $F_i$ são as forças de atuação de cada subsistema. Os índices são identificados como: $m$ = mestre; $s$ = escravo (*slave*); $h$ = manipulação (*handling*); $e$ = ambiente (*environment*); 2 = canal escravo-mestre; 3 = canal mestre-escravo. Por fim, $e^{-sT}$ é o atraso na comunicação bilateral (REBELO; SCHIELE, 2015).

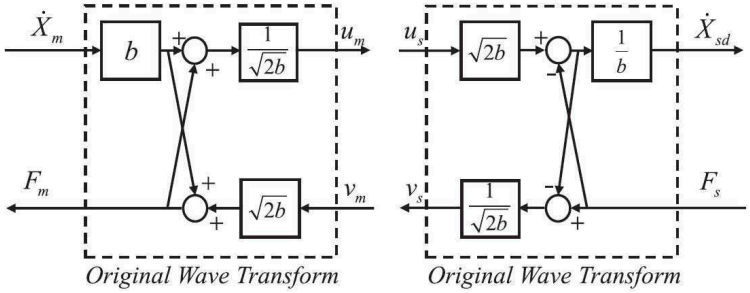
O conceito de controle e transmissão de impedância é muito utilizado, pois a partir de Hogan (1984), se tornou um método mais eficiente de modelar a interação entre sistemas mecânicos, quando comparado a métodos nos quais, por exemplo, se mede a potência fornecida pelos atuadores do sistema, que está diretamente relacionada com a potência trocada com o ambiente de operação.

Colonnese e Okamura (2017) analisaram como ocorre a transmissão de impedância efetiva em sistemas bilaterais de troca de posição, levando em consideração todos os problemas de atraso na comunicação. Foi concluído então que o efeito do atraso na comunicação causa os seguintes efeitos nos parâmetros de impedância transmitidos: a rigidez transmitida sempre será menor que a rigidez real do ambiente, mas de uma forma pouco significativa; o amortecimento transmitido pode ser substancialmente maior que o amortecimento real do ambiente, aumentando proporcionalmente ao aumento do atraso; a massa transmitida, por fim, sofre pouca influência do atraso na comunicação.

Uma outra técnica que atraiu a atenção de diversos autores, é a compensação do problema de atraso no tempo através de uma variável de onda, que transmite as variáveis de força e velocidade de forma energética, conforme visto na Figura 3. Este método foi proposto por Niemeyer e Slotine

(1991a), Niemeyer e Slotine (1991b). Este trabalho foi a base para várias pesquisas futuras no âmbito de teleoperação bilateral.

Figura 3 – Transformação de onda para sistemas *4-channel*.




Fonte: Chen et al. (2018).

Na Figura 3, as variáveis $F_i$ e $\dot{X}_i$, são as forças e velocidades dos manipuladores $m$ e $s$, sendo que $\dot{X}_sd$ se refere à velocidade desejada do escravo. Já $u_m$, $u_s$, $v_m$ e $v_s$ são os canais de comunicação por variável de onda, e $b$ é o parâmetro de impedância, que pode ser constante ou variável (CHEN et al., 2018).

Chen et al. (2018) propuseram uma melhoria no sistema de Niemeyer e Slotine (1991a), pois havia uma dificuldade em balancear a estabilidade do sistema de controle com a transparência das informações transmitidas. Foi utilizado a arquitetura *4-channel*, e utilizando um compensador de passividade baseado em atraso no tempo e realimentação, assim foi possível reduzir a distorção causada pela reflexão de onda, aumentando a performance geral do sistema.

Muitos outros autores publicaram recentemente trabalhos que envolvem controle e reflexão de força e sistemas com atraso no tempo, como Heck et al. (2018), Yuan, Wang e Guo (2018), Tashiro et al. (2018). Todos eles tiveram abordagens diferentes para tentar atender requisitos similares de estabilidade e transparência dos sistemas através de modelagem e simulação matemática, e até mesmo utilizando protótipos e plataformas de testes experimentais com modelos reduzidos, como sistemas mecânicos de teleoperação bilateral com apenas um grau de liberdade.

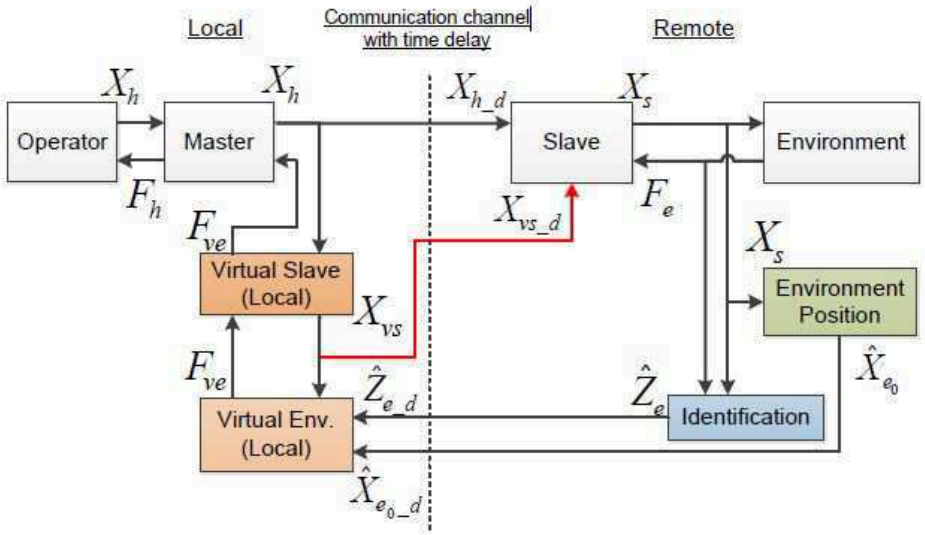
Já no âmbito de sistemas sujeitos a ambientes desconhecidos ou com dinâmica desconhecida, também foram encontrados alguns trabalhos. Rahimifard, Talebi e Mohammadi (2016) abordaram a sincronização de sistemas bilaterais de teleoperação sujeitos a não-linearidades desconhecidas. Através de modelos passivos e não-passivos, e do controle de impedância, foi possível melhorar o seguimento de posição em modelos teóricos.

Pecly, Souza e Hashtrudi-Zaad (2018) propuseram um sistema que trata tanto de ambientes desconhecidos quanto do problema do atraso no tempo. Ao invés de utilizar ferramentas para compensar o atraso na comunicação, foi desenvolvido um sistema de identificação de sistemas. A partir da aquisição de dados de sensores de força no escravo, é possível obter um modelo aproximado do ambiente. Portanto a realimentação do sistema de teleoperação é proveniente de um ambiente virtual, onde a comunicação com o dispositivo mestre ocorre de forma local, sem o atraso da teleoperação.

Observa-se portanto que é possível prever as reações do escravo ao ambiente virtual, antes do escravo interagir com o ambiente real de operação, e então o operador é capaz de reagir a obstáculos e demais restrições de forma mais rápida do que se aguardasse a realimentação dos valores reais da

interação do escravo com o ambiente real. A topologia desse sistema pode ser visto na Figura 4, onde $X_i$, $F_i$, $Z_i$ são as posições, forças e impedâncias, respectivamente, e os índices $i$ são: h = manipulação; $s$ = escravo; $vs$ = escravo virtual (*virtual slave*); $ve$ = ambiente virtual (virtual environment; $e$ = ambiente; $e_0$ = referencial do ambiente [1].

Figura 4 – Teleoperação mediada por modelo de referência.




Fonte: Pecly, Souza e Hashtrudi-Zaad (2018).

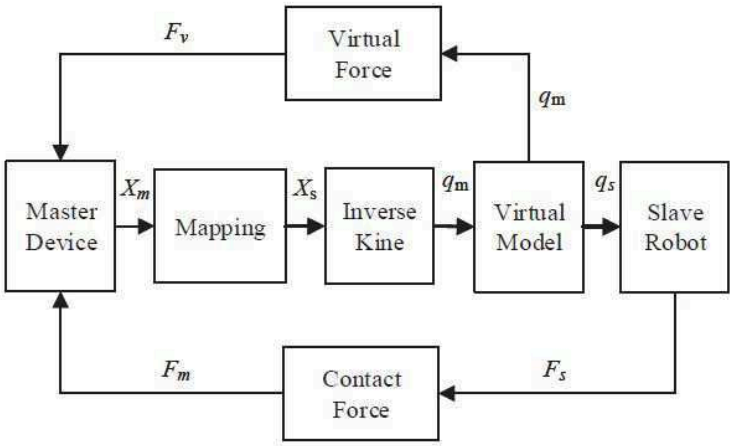
Um trabalho similar ao de pecly2018 foi publicado por Zhao et al. (2017), onde é utilizado um modelo virtual para se obter uma realimentação de força virtual do escravo, que pode ser explicado pela Figura 5, onde a nomenclatura das variáveis segue o mesmo padrão das Figuras 2, 3 e 4, com adição a parâmetro $q$, que são as variáveis de junta.. Nessa pequisa foi considerado um sistema no qual o mestre e o escravo são construtivamente diferentes, o que leva à necessidade de um mapeamento e transformação de coordenadas e movimento. Aliado a isso, o sistema de realimentação de força virtual opera em diferentes zonas: orientação, repulsão e proibição.

Ainda nesse contexto, Liu et al. (2017) também utilizou o conceito de restrições virtuais através da identificação por imagens. Foram utilizados dois dispositivos hápticos Phantom Omni®, como mestre e escravo, em conjunto com o sistema de câmera Kinect®, que realiza a identificação de obstáculos e restrições, gerando então uma realimentação de força que serve como um "alerta"ao operador de que o escravo está próximo aos obstáculos.

A questão de meios de comunicação em sistema de teloperação também é um assunto consideravelmente pesquisado. Tian, Gao e Zhang (2017) desenvolveram um sistema de teleoperação bilateral *wireless* de três canais de comunicação. Um sistema de chaveamento dos três canais alterna entre a transmissão de informações de força para a realimentação de contato do escravo, e o envio de dados de posição, para o controle de movimento. Chegou-se à conclusão, que com esse sistema, foi possível de se obter estabilidade ao custo da diminuição da transparência de realimentação de força nas situações de contato de escravo.

[1] O acento circunflexo representa os sistemas e variáveis estimadas, e o o índice $d$ representa as informações com atraso de comunicação.

Figura 5 – Teleoperação com realimentação virtual.




Fonte: Zhao et al. (2017).

O trabalho desenvolvido por Foletto (2013) utiliza o UKFI (*Unscented Kalman Filter Intermittent*) para reduzir o erro em estimações de sistemas de controle via rede (*Networked Control Systems*) com perdas de pacotes na comunicação entre sistemas mestre e escravo. Essa reformulação foi aplicada em um sistema de teleoperação de um robô cartesiano. Chegou-se a conclusão de ocorre a degradação de um sistema de teleoperação via rede sem fio quando é utilizado um estimador que não considera a perda de pacotes em seu modelo. Consequentemente, modelos de estimação que consideram a perda de pacotes aumentam o desempenho do sistema.

Uma pesquisa utilizando robôs industriais em um sistema de teleoperação bilateral foi realizada por Lima et al. (2018), no qual foi utilizado um dispostivo háptico Geomagic® como mestre e uma mão robótica com sensores de força acoplada ao efetuador de um robô industrial ABB de pequeno porte como o sistema escravo. Desta forma foi possível utilizar uma realimentação real de força de interação do escravo com o ambiente de operação. O sistema de controle de teleoperação neste trabalho foi implementado por intermédio de um ambiente virtual de simulação, programado nos *softwares* ROS (*Robot Operating System*) e V-REP (*Virtual Robot Experimentation Platform*). Neste sistema era visualizada a movimentação do sistema virtual, e então com a utilização do *software* MoveIt!, foi feita a geração de trajetória que foi enviada até o controlador do escravo, realizando a manipulação.

Dalvand e Nahavandi (2014) conceberam um sistema de implementação simplificada utilizando um robô industrial ABB como dispositivo escravo, e um *joystick* comum como dispositivo mestre, sem realimentação de força, ou seja, trata-se de um sistema de teleoperação unilateral. Ainda assim, foi possível desenvolver um sistema de teleoperação de performance satisfatória, utilizando métodos de controle e programação desenvolvidos pelo próprio fabricante do robô, e com tecnologia de processamento computacional de baixo custo.

## 1.2 PROBLEMÁTICA

De acordo com o que foi exposto na revisão bibliográfica, nota-se que a teleoperação de robôs é um tema que possui grande destaque no meio técnico e científico. Isso também é impulsionado pelo

aumento de robôs sendo utilizados no mundo, em diversas aplicações. Entretanto, mesmo que exista uma grande atividade de desenvolvimento nessa área, notou-se que é pequeno o número de trabalhos que consideram as limitações dos robôs industriais e seus respectivos controladores, no que se refere a comunicação de dados e programação.

A maioria dos artigos e trabalhos encontrados levam em conta modelos e tecnologias com comportamentos ideais, sem considerar todas as suas limitações quando aplicadas em cenários reais. Mesmo nos trabalhos nos quais se vê testes e aplicações práticas, a maioria dos equipamentos utilizados possuem arquitetura de controle aberta, o que se torna muito vantajoso para pesquisas e resultados laboratoriais, porém pode ser restritivo para aplicações mais práticas, como as industriais, onde se requer um custo mais baixo possível e que não sejam superdimensionados.

Em termos de aplicações práticas, o estudo da teleoperação de manipuladores industriais pode levar a contribuições muito significativas para a área de robótica. A teleoperação não precisa ser utilizada exclusivamente para manipulação industrial. Diversas outras áreas se correlacionam com a teleoperação bilateral, tendo como alguns exemplos: reconhecimento de superfícies, programação de robôs por aprendizagem, detecção e identificação de ambientes de operação, cooperação homem-máquina, direção assistida etc.

Levando em consideração esses fatores, é de interesse técnico e científico a pesquisa e o desenvolvimento de sistemas de teleoperação que levem em consideração as limitações e requisitos operacionais de equipamentos de uso comum e já difundidos no mercado. Assim, surge uma possibilidade de contribuição para a área, uma vez que este trabalho pode proporcionar uma nova visão sobre os desafios de se aplicar as técnicas e conceitos já desenvolvidos, como a realimentação de força e o controle de impedância, na teleoperação bilateral de equipamentos e sistemas reais disponíveis no mercado.

## 1.3 PROPOSIÇÃO

Este trabalho busca explorar os conceitos e dificuldades de teleoperação bilateral quando aplicados em um sistema industrial. Utilizando *softwares* de programação *offline* de robôs industriais, é possível ter uma estimativa de como seria o comportamento real de um robô teleoperado e interagindo com restrições e ambientes virtuais, que reagem ao movimento do robô de acordo com um modelo pré-determinado.

A proposta deste trabalho é desenvolver um modelo de simulação de teleoperação bilateral de robôs manipuladores industriais, considerando restrições virtuais de movimento. É proposta também a utilização de ferramentas de simulação disponibilizadas por uma plataforma de desenvolvimento e programação *offline* e utilizando uma linguagem de programação de robôs, ambos desenvolvidos pelo próprio fabricante do robô em questão. Esse sistema é então comandado externamente por aplicações em *software* desenvolvidas durante o trabalho, e cuja interface de comando do robô pode ser tanto a manipulação direta do efetuador final ou através de um sensor de força, que transforma medições de esforço provenientes da manipulação por um operador humano em comandos de movimento, que são enviados para o *software* de simulação do robô, que então interpreta e executa os comandos recebidos.

Propõe-se nesta dissertação dois modelos de simulação de teleoperação bilateral. O primeiro modelo considera o comportamento no tempo contínuo com o intuito de simular o comportamento

ideal do sistema de impedância virtual que compõe o ambiente virtual de operação do robô escravo. A partir da observação do comportamento desejado de operação nesse primeiro modelo de operação, é proposto um segundo modelo cuja concepção é similar a do primeiro modelo, mas com a adição do comportamento no tempo discreto, que é o domínio no qual operam os sistemas de controle de robôs industriais, pelo fato de utilizarem controladores digitais e não analógicos.

### 1.3.1 Objetivos

O objetivo geral deste trabalho é desenvolver um modelo matemático de simulação de teleoperação bilateral de robôs manipuladores industriais sujeitos a restrições virtuais de movimento no espaço de trabalho.

Já os objetivos específicos deste trabalho são:

- Desenvolver um sistema de comando do manipulador mestre utilizando um sensor de força real;
- Desenvolver um modelo matemático da interação do manipulador mestre com as restrições virtuais no domínio do tempo discreto;
- Avaliar o desempenho do sistema com e sem as restrições virtuais;
- Analisar o comportamento e propriedades dos atrasos de resposta no sistema no tempo discreto.
- Analisar propriedades, caracaterísticas e comportamentos da teleoperação bilateral em ambientes de simulação discretos e contínuos.

## 1.4 ESCOPO DO TRABALHO

O desenvolvimento desse trabalho ocorreu da seguinte forma: primeiramente foi feito um modelo de simulação de teleoperação biletareal no tempo contínuo. Esse primeiro modelo foi desenvolvido na plataforma Simulink do MatLab, onde foi possível modelar dois manipuladores (mestre e escravo) de três graus de liberdade com características construtivas distintas. O robô mestre opera de forma similar a um dispositivo háptico, pois é capaz de aplicar forças em quem o opera. O segundo robô, um SCARA, é comandado pela movimentação do mestre, e interage com um ambiente virtual que nele são impostas restrições de movimento. Essas restrições se comportam como uma impedância, gerando forças de reação virtuais que são enviadas para o mestre.

Na segunda parte do trabalho é desenvolvido um modelo de simulação composto por equações no tempo discreto. É utilizado o *software* de simulação *offline* de robôs da empresa ABB, o RobotStudio. Este simulador é capaz de reproduzir exatamente o comportamento de robôs reais, contanto que as informações do ambiente virtual sejam conhecidas. Neste simulador existem os modelos de dois robôs IRB 1600, operando como mestre e escravo, e seus respectivos controladores. O simulador é capaz de se comunicar via TCP/IP com dispositivos e aplicativos externos, como outros computadores ou outros *softwares* sendo executados na mesma máquina. Em função disto foi possível comandar o movimento virtual do mestre através de um sensor de força real conectado ao mesmo computador no qual é excutado o RobotStudio.

Através de um admitância virtual, as forças lidas no sensor são transformadas em comandos de deslocamento, que são enviados para o controlador virtual do robô mestre, também via TCP/IP. Nesta simulação porém, é o mestre quem está sujeito às restrições de um ambiente virtual, que foi pré-definido e programado em Python. Essa aplicação também se comunica via TCP/IP com o simulador, trocando informações de movimento e interação do robô com o ambiente virtual, bem como os comandos de movimento proveninentes do sensor de força e admitância virtual. Conforme o mestre interage com as restrições, baseadas também em uma impedância virtual, surgem forças de reação, que são descontadas dos valores medidos do sensor de força, constituindo então a realimentação, e consequentemente, a bilateralidade do sistema de teleoperação proposto. Por fim o robô escravo recebe as informações de deslocamento do mestre, tentando reproduzí-las, porém sem interagir diretamente com nenhum ambiente ou restrição.

Duas aplicações diferentes do mesmo simulador são executadas ao mesmo tempo na mesma máquina, ou seja, os modelos dos robôs mestre e escravo não operam no mesmo ambiente de simulação. Portanto é possível simular problemas de atraso de comunicação, mesmo que usando uma conexão TCP/IP interna da máquina. O objetivo de separar os dois robôs na simulação é de que a comunicação mestre-escravo seja intermediada pela aplicação em Python, constituindo um sistema centralizado de comunicação de todo o sistema.

## 1.5 ORGANIZAÇÃO DO TRABALHO

Este trabalho está organizado da seguinte forma: no Capítulo 2 são abordados os fundamentos utilizados no que se refere a robôs manipuladores industriais. O Capítulo 3 mostra os conceitos que basearam a interação mecânica entre os sistemas e as medições de forças, bem como a proposta de ambiente virtual de operação dos manipuladores.

O primeiro modelo de simulação é proposto e testado no Capítulo 4. Esse sistema opera no tempo contínuo e em situações ideias de comunicação e controle, ou seja, sem levar em consideração o comportamento real de equipamentos reais. Já o segundo sistema de teleoperação que utiliza modelos virtuais de robôs e controladores reais é proposto considerando o tempo discreto e simulado no Capítulo 5. Por fim, a conclusão do trabalho e perspectivas futuras são apresentadas no Capítulo 6.

# 2 ROBÔS MANIPULADORES INDUSTRIAIS

Uma definição geral de robôs industriais, de acordo com a IFR (2018), é a de que:

> Robôs industriais conforme definido pela ISO 8373-2012: um manipulador multiuso controlado automaticamente, reprogramável, programável em três ou mais eixos, que pode ser tanto fixo ou móvel para uso em aplicações de automação industrial.

Robôs podem ser classificados em duas classes distintas: robôs industriais e robôs de serviço (IFR, 2018). Robôs industriais são, na maioria dos casos, manipuladores utilizados em diversas atividades de fabricação, como montagem, soldagem, usinagem, movimentação etc. Os robôs de serviço são utilizados em diferentes tarefas, incluindo algumas tarefas na indústria, e são na sua maioria representados por drones, robôs móveis, VANTs (Veículo Aéreo Não Tripulado), robôs domésticos, robôs cirúrgicos, robôs de limpeza, etc. Um exemplo de robô industrial pode ser visto na Figura 6.

Figura 6 – Robô industrial com 7 DOFs.

Fonte: Siciliano e Khatib (2008).

Segundo Scheinman e McCarthy (2008), a estrutura mecânica de um robô consiste de uma série de elos conectados entre si, formando uma cadeia cinemática. Consequentemente a estrutura possui duas formas básicas: uma única cadeia serial, se configurando em um robô serial, ou um conjunto de cadeias seriais suportando um efetuador final, o que consiste então em um robô paralelo.

Adicionalmente, Hägele, Nilsson e Pires (2008) classificam algumas topologias básicas de robôs manipuladores conforme sua construção mecânica, espaço de trabalho e movimentação, como segue abaixo:

- **Cartersianos**: robôs que possuem apenas juntas prismáticas;
- **Articulados**: robôs constituidos normalmente apenas por juntas rotatativas, ligadas de forma serial;
- **SCARA (*Selective Compliance Assembly Robot Arm*)**: um modelo específico de robô, que possui duas ou três juntas rotativas e uma junta prismática, tendo um espaço de trabalho cilíndrico.

Já em relação à performance de robôs manipuladores, Scheinman e McCarthy (2008) afirma que "a vantagem mecânica de uma máquina é o inverso de sua taxa de velocidade". Considerando

a máquina "robô", pode se dizer em outras palavras, que quanto maior a velocidade de um robô manipulador, menor será sua capacidade de exercer esforço de forma adequada e estável.

O princípio de funcionamento de sistemas robóticos é baseado essencialmente na cinemática, estática e dinâmica do sistema mecânico que constituem o modelo matemático de um robô (SICILIANO; KHATIB, 2008). A cinemática descreve o posicionamento e movimentação do robô, relativo tanto ao seu espaço de trabalho quanto às suas variáveis e características construtivas (elos e juntas) (WALDRON; SCHMIEDELER, 2008). A estática é derivada da modelagem cinemática de posição do robô, pela qual são calculadas as forças e torques necessários para que o robô consiga exercer uma força ou momento estáticos no espaço operacional. A dinâmica por sua vez relaciona todas as características da cinemática (posição, velocidade e aceleração) com as forças e torques necessários para realizar a movimentação (FEATHERSTONE; ORIN, 2008).

Neste trabalho foram considerados apenas os comportamentos cinemáticos e estáticos de robôs manipuladores. A dinâmica de robôs manipuladores não foi abordada nesta trabalho pelo fato de que, nos testes e simulações, foram utilizados robôs industriais com arquitetura de controle fechada, ou seja, a solução do problema de dinâmica dos robôs utilizados já está resolvida e incorporada ao controlador do robô, cujo fabricante é o mesmo do manipulador.

## 2.1 MODELAGEM CINEMÁTICA DE ROBÔS MANIPULADORES

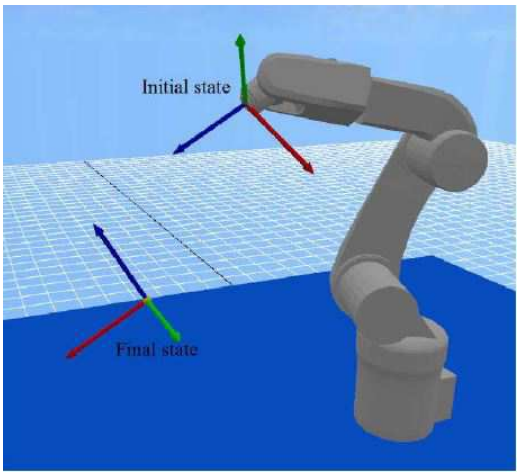
Conforme Waldron e Schmiedeler (2008), "cinemática de robôs descreve a postura, velocidade, aceleração e todas as derivadas de alta ordem da postura dos corpos que abrangem um mecanismo". Dito isto, as equações da cinemática permitem que se possa mapear o posicionamento e orientação de um conjunto de corpos rígidos, interligados por juntas, de forma paralela, serial ou mista.

As coordenadas e orientação de um robô e seus elos são mapeados em sistemas de coordenadas, que se relacionam através de transformações rígidas (WALDRON; SCHMIEDELER, 2008). Essas transformações se dão por meio de translações e rotações. As translações alteram a posição dos sistemas de coordenadas $x$, $y$ e $z$, enquanto as rotações alteram a orientação do sistema de coordenadas, também nos três eixos principais. Um exemplo de transformação de sistema de coordenadas pode ser visto na Figura 7, onde há o sistema que representa a postura atual do efetuador final de um manipulador industrial, e a postura final desejada para o mesmo. É importante salientar também que normalmente o sistema global de coordenadas de um robô manipulador é fixado na base da estrutura do manipulador, e os demais sistemas de coordenadas são referenciados a esse sistema global.

As transformações nos sistemas de coordenadas podem ser representadas matematicamente de diversas formas: matriz de transformação (rotação por ângulos de Euler ou ângulos de base fixa) (WALDRON; SCHMIEDELER, 2008), eixo-ângulo (WALDRON; SCHMIEDELER, 2008), helicóides (DAVIDSON; HUNT; HUNT, 2004) e quatérnios (HAMILTON, 1848; PHAM et al., 2010).

A Equação 2.1 mostra uma matriz de transformação homogênea $A$ entre dois sistemas de coordenadas $i$ e $j$, que é composto por: um vetor de posição $p \in \mathbb{R}^3$ que mostra a translação entre os dois sistemas de coordenadas; e uma matriz matriz de rotação $R \in \mathbb{R}^{3x3}$ que define a orientação do sistema de coordenadas $j$ em relação ao sistema de coordenadas (SICILIANO et al., 2010).

Figura 7 – Transformação de sistemas de coordenadas.




Fonte: Pham et al. (2010).

$$A_j^i = \begin{bmatrix} R^i_{j(3x3)} & p^i_{j(3x1)} \\ 0_{(1x3)} & 1 \end{bmatrix} \tag{2.1}$$

Quando um robô ou mecanismo possui dois ou mais corpos rígidos interconectados em série, pode se obter a matriz de transformação homogênea entre o primeiro e último sistema de coordenadas através da multiplicação sucessiva das matrizes de transformação do robô (WALDRON; SCHMIEDELER, 2008). Para um robô com n elos (consequentemente n sistemas de coordenadas), a matriz de transformação entre o sistema de coordenadas 0 e $n$ pode ser calculada através da Equação 2.2. Como o robô se movimenta através da atuação de suas juntas, a postura e velocidade do efetuador final dependem das variáveis de junta do robô.

$$A_n^0 = A_1^0 A_2^1 A_3^2 ... A_n^{n-1} \tag{2.2}$$

### 2.1.1 Cinemática direta

As equações de cinemática direta de um robô são utilizadas para calcular a postura (posição e orientação) do efetuador final, ou qualquer outro ponto específico do conjunto de corpos rígidos de um robô ou mecanismo, em relação ao sistema de coordenadas localizado na base do robô. Desta forma, os vetores de postura $P$ e velocidades $v$ de um robô com $n$ juntas, podem ser representados conforme as Equações 2.3 e 2.4, onde q são as variáveis de junta e $\Theta$ são os ângulos de RPY da matrix de rotação.

$$P(q) = \begin{bmatrix} p(q) \\ \Theta(q) \end{bmatrix} = \begin{bmatrix} p_x(q) \\ p_y(q) \\ p_z(q) \\ \alpha(q) \\ \beta(q) \\ \gamma(q) \end{bmatrix} \tag{2.3}$$

$$v(q,\dot{q}) = \dot{P}(q) = \begin{bmatrix} \dot{p}(q) \\ \dot{\Theta}(q) \end{bmatrix} = \begin{bmatrix} v_x(q,\dot{q}) \\ v_y(q,\dot{q}) \\ v_z(q,\dot{q}) \\ \omega_x(q,\dot{q}) \\ \omega_y(q,\dot{q}) \\ \omega_z(q,\dot{q}) \end{bmatrix} \tag{2.4}$$

Um das ferramentas mais utilizadas em cinemática de robôs é a cinemática diferencial, que pode ser calculada pela matriz Jacobiana, que depende das posições e velocidades instantâneas das juntas (BONILLA, 2004). Este método permite calcular as velocidades no sistema global de coordenadas a partir de um vetor de velocidades de junta e de uma matriz Jacobiana, que representa a contribuição de cada junta para cada posição e orientação no sistema global de coordenadas, conforme mostra a Equação 2.5. Uma das formas dessa matriz é a Jacobiana analítica, que é obtida derivando parcialmente cada equação da postura do manipulador em relação a cada uma das variáveis de junta, conforme a Equação eq:jacob. Aplicando (2.3) em (2.6), e considerando um robô com n juntas independentes, é possível obter a matriz jacobiana conforme a Equação 2.7. Existe também a matriz Jacobiana geométrica, que é explicada mais detalhadamente em sciavicco, porém não é utilizada nesse trabalho.

$$v(q,\dot{q}) = J(q)\dot{q} \tag{2.5}$$

$$J(q) = \frac{\partial P(q)}{\partial q} \tag{2.6}$$

$$J(q) = \begin{bmatrix} \frac{\partial p_x(q)}{\partial q_1} & \frac{\partial p_x(q)}{\partial q_2} & \cdots & \frac{\partial p_x(q)}{\partial q_n} \\ \frac{\partial p_y(q)}{\partial q_1} & \frac{\partial p_y(q)}{\partial q_2} & \cdots & \frac{\partial p_y(q)}{\partial q_n} \\ \frac{\partial p_z(q)}{\partial q_1} & \frac{\partial p_z(q)}{\partial q_2} & \cdots & \frac{\partial p_z(q)}{\partial q_n} \\ \frac{\partial \alpha(q)}{\partial q_1} & \frac{\partial \alpha(q)}{\partial q_2} & \cdots & \frac{\partial \alpha(q)}{\partial q_n} \\ \frac{\partial \beta(q)}{\partial q_1} & \frac{\partial \beta(q)}{\partial q_2} & \cdots & \frac{\partial \beta(q)}{\partial q_n} \\ \frac{\partial \gamma(q)}{\partial q_1} & \frac{\partial \gamma(q)}{\partial q_2} & \cdots & \frac{\partial \gamma(q)}{\partial q_n} \end{bmatrix} \tag{2.7}$$

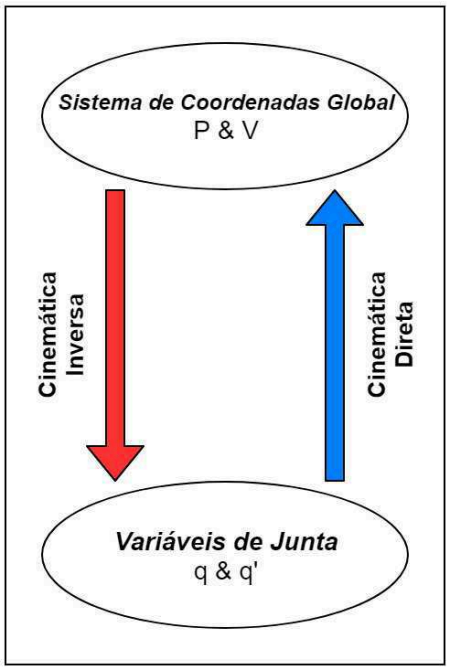
Em aplicações práticas de robôs manipuladores, a cinemática direta é utilizada para calcular a postura real do efetuador final de um robô a partir da leitura em tempo real das variáveis de junta do robô, que é feita com o uso de sensores de posição e velocidade posicionados na montagem mecânica das juntas. Robôs manipuladores industriais normalmente possuem controladores próprios e dedicados que fazem todos os cálculos de cinemática, não sendo necessário a execução dos mesmos pelo usuário ou por um controlador externo.

### 2.1.2 Cinemática inversa

A cinemática inversa é utilizada para se obter as variáveis de junta de um robô (WALDRON; SCHMIEDELER, 2008), tendo como argumentos de entrada uma postura pré-determinada, conforme

mostra a Figura 8, que em geral é medida no efetuador final. Um dos métodos de calcular a cinemática inversa de um robô é utilizando a Equação 2.5, isolando o vetor de velocidades das juntas, resultando na Equação 2.8 (WHITNEY; PEIPER, 1972, 1968 apud WALDRON; SCHMIEDELER, 2008).

Figura 8 – Cinemática inversa e direta.




Fonte: do autor.

$$\dot{q} = J(q)^{-1} v(q, \dot{q}) \tag{2.8}$$

É importante salientar que para que se calcule a cinemática inversa com a matriz Jacobiana, é necessário que esta seja inversível, ou seja, quando for uma matriz quadrada e de posto completo. No caso de não haver possibilidade de inversão, podem ser utilizados métodos para se calcular pseudo-inversas e outras aproximações para a cinemática inversa (CHIAVERINI; ORIOLO; WALKER, 2008; WALDRON; SCHMIEDELER, 2008).

## 2.2 PRINCÍPIO BÁSICO DE ESTÁTICA DE ROBÔS

Similar à cinemática inversa de robôs, a estática pode ser resolvida também com a utilização da matriz Jacobiana (SICILIANO et al., 2010). Dado um vetor $f$ de forças e torques a serem exercidos pelo efetuador final do robô no sistema global de coordenadas, é possível calcular o vetor $\tau$ de forças e torques necessários em cada junta $j$ do robô (força $f_j$ para juntas prismáticas e torque $\tau_j$ para juntas rotativas), conforme a Equação 2.9.

$$\tau(q) = J(q)^T f \tag{2.9}$$

$$f = \begin{bmatrix} f_x \\ f_y \\ f_z \\ M_x \\ M_y \\ M_z \end{bmatrix} \tag{2.10}$$

O conceito de estática de robôs permite o controle de força da robôs, o que se torna muito útil em aplicações onde existe a interação física do manipulador com o ambiente de operação (WALDRON; SCHMIEDELER, 2008). Em situações de teleoperação bilateral isso é de grande importância, pois permite a realimentação de força e percepção do ambiente de forma tátil.

## 2.3 DISPOSITIVOS HÁPTICOS

Sistemas hápticos são aqueles em que são fornecidas experiência de toque em operadores humanos, onde a interface é realizada com equipamentos mecatrônicos que utilizam sensores e atuadores a fim reproduzir interações mecânicas (HANNAFORD; OKAMURA, 2016). Esses equipamentos são chamados de dispositivos hápticos, e de acordo com Xu et al. (2017) são capazes de reproduzir sensações cinetostáticas e táteis provenientes da interação entre um operador e um ambiente real ou virtual.

Um exemplo de dispositivo háptico pode ser visto na Figura 9, que contém um Phantom Omni (SANSANAYUTH; NILKHAMHANG; TUNGPIMOLRAT, 2012). Esse dispositivo eletromecânico possui movimentação e mapeamento de posição e velocidade em seis graus de liberdade, capaz de fornecer uma realimentação de força em três graus de liberdade, ou seja, é capaz de aplicar forças nos três eixos ortogonais $x$, $y$ e $z$. Ele pode ser utilizado em diversas aplicações, como entretenimento, realidade virtual e teleoperação.

Figura 9 – Dispositivo háptico Phantom Omni.

Fonte: Sensable Technologies, 2008.

Apesar de estarem disponíveis no mercado, eles ainda estão muito mais presentes em cenários de pesquisa, simulações e testes em laboratório. Como este trabalho visa a aplicação de tecnologias

de uso industrial, se torna interessante a utilização de robôs industriais, até pela sua presença e disponibilidade nas indústrias. E como os dispositivos hápticos tem o mesmo conceito de funcionamento de robôs manipuladores, automaticamente os manipuladores industriais podem ser capazes de operar como dispositivos hápticos. Além disso, a tecnologia utilizada em sistemas hápticos de robótica pode ser aproveitada em sistemas onde há a operação cooperativa de humanos e manipuladores em tarefas industriais (BICCHI; PESHKIN; COLGATE, 2008).

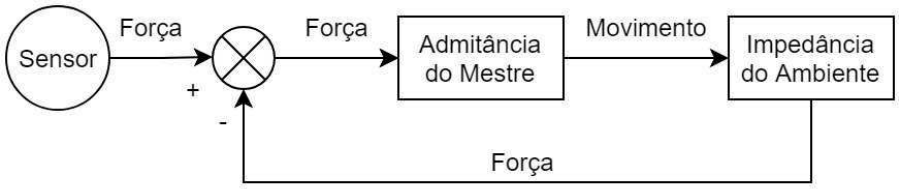
# 3 CONTROLE DE FORÇA E POSIÇÃO

Como os sistemas de teleoperação de robôs normalmente atuam em contato com o meio físico, seja ele o operador ou o ambiente de operação, as forças de contato entre ambiente e manipulador são variáveis de grande importância para o sistema. Portanto é necessária a medição, o comando e o controle dessas forças para o sistema operar de forma autônoma e segura.

A interação entre manipuladores e ambiente exige que o sistema seja capaz medir ou estimar as forças de contato. No primeiro caso utiliza-se sensores de força e de torque que normalmente são acoplados no efetuador final do robô. No caso de estimação de força, pode-se, entre outras formas de medição, medir o deslocamento do efetuador final em contato com o meio e, a partir da característica do meio representado pelo coeficiente da impedância do meio, estimar a força exercida. O controle e a relação entre as forças medidas ou estimadas, e as velocidades e posições dos robôs são calculadas através do método de controle de impedância e admitância (PECLY; SOUZA; HASHTRUDI-ZAAD, 2018).

Neste trabalho, como o mestre irá responder com deslocamento a estímulos de força, medidos por um sensor, ele se comporta como uma admitância. O processo de reação ocorre através de uma admitância virtual do mestre, que é arbitrada na programação do robô, e que é utilizada para calcular o deslocamento do mestre a uma entrada de força, que no caso é a força medida no sensor. Já o ambiente, que restringe o movimento do manipulador escravo, possui um comportamento de impedância, que fornece uma força de reação, considerada como a realimentação de força. Esta força é calculada utilizando os dados de movimento do mestre, que são as entradas da impedância virtual de um ambiente de operação também virtual, no qual o manipulador está inserido. O sistema proposto de controle de impedância/admitância pode ser visto na Figura 10.

Figura 10 – Controle de impedância/admitância do sistema de teleoperação.




Fonte: do autor

## 3.1 AMBIENTE DE OPERAÇÃO

Ambientes de interação mecânica podem ser classificados em dois tipos: os ambientes passivos e ambientes ativos. Os ambientes passivos são aqueles que não geram energia durante a interação, sendo normalmente compostos por elementos dissipadores de energia. Já os ambientes ativos, são aqueles em que o ambiente é capaz de fornecer energia ao sistema, através de atuadores e outros elementos similares (NIEMEYER; PREUSCHE; HIRZINGER, 2008).

Robôs manipuladores normalmente operam em ambientes passivos, que normalmente são compostos por sistemas equivalentes a sistemas de mola, massa e amortecedor. Esses ambientes podem

ser divididos em: ambientes cinemáticos, ambientes dinâmicos e ambientes flexíveis (AMARAL; PIERI; GUENTHER, 2000).

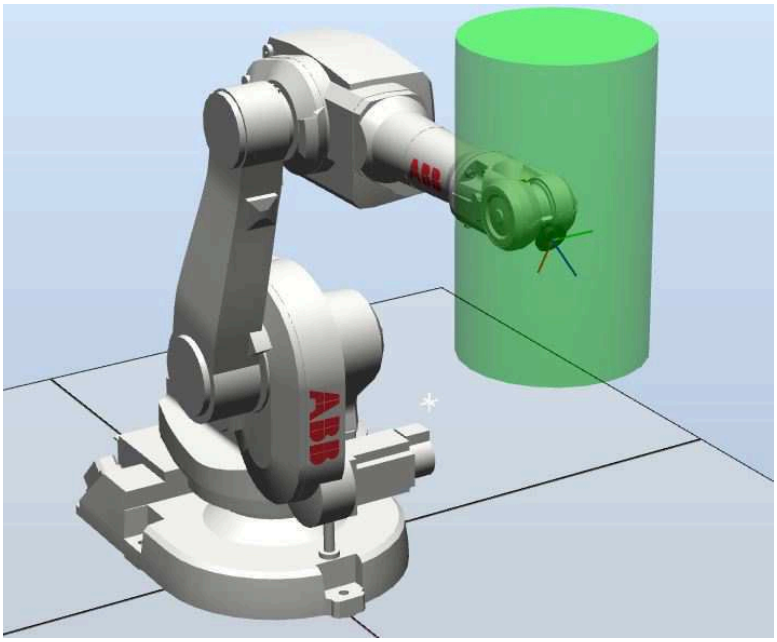
- Ambientes cinemáticos: possuem apenas restrições geométricas, permitindo movimento puramente tangencial do efetuador nas superfícies e bloqueando o movimento em direções normais às superfícies.
- Ambientes dinâmicos: são ambientes que possuem uma resposta dinâmica quando sujeitos a movimento em uma ou mais direções. Nas demais direções, o ambiente se comporta de forma cinemática.
- Ambientes flexíveis: apresentam flexibilidade em uma ou mais direções, se comportando de forma cinemática ou dinâmica nas demais direções.

### 3.1.1 Ambiente virtual

Neste trabalho, foi proposto um ambiente virtual de operação no qual os manipuladores são inseridos em simulação. A intenção dessa proposição foi simular a interação do efetuador final dos robôs manipuladores com restrições e obstáculos que possuem comportamento dinâmico, reagindo passivamente às interações de contato mecânico entre robô e ambiente.

O ambiente foi concebido para simular um ambiente genérico com restrições dinâmicas, sendo dimensionado como um volume cilíndrico. Este é virtualmente centralizado na posição inicial de operação dos robôs, conforme visto na Figura 11 e Figura 12.

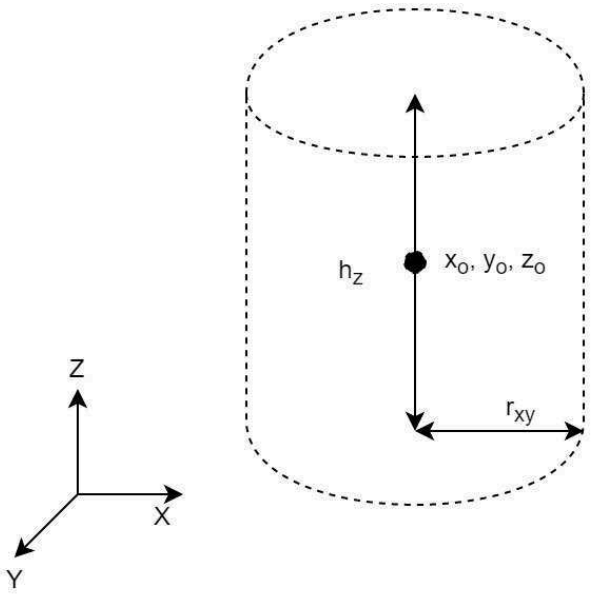
Figura 11 – Robô no ambiente de operação virtual.




Fonte: do autor.

A interação entre o manipulador e o ambiente virtual é feito através do mapeamento da posição do manipulador. A realimentação de força só ocorre se o efetuador final do manipulador sair de dentro da zona cilíndrica. O sistema de teleoperação detecta então que o robô está entrando em uma zona

Figura 12 – Geometria do ambiente de operação virtual.




Fonte: do autor.

restritiva, e fornece ao controle de impedância os dados de movimento para que seja calculada a força virtual de realimentação para o mestre.

### 3.1.2 Monitoramento

O monitoramento da posição é feito de duas formas diferentes. Para as coordenadas $x$ e $y$, é considerado o deslocamento radial em relação ao centro da base do cilindro. Já para a coordenada $z$, é considerado apenas o deslocamento linear vertical em relação aos limites inferior e superior da área cilíndrica.

Para os eixos $x$ e $y$, primeiramente é calculada a posição relativa $p_r$ do efetuador final em relação ao centro do cilíndro ($x_o$ e $y_o$), para que então seja possível calcular o ângulo de deslocamento relativo $\sigma$, com o qual se avalia os limites $x_{lim}$ e $y_{lim}$. Esses limites são a base para verificar o deslocamento $dx$ e $dy$ do efetuador na zona restritiva. Todo o cálculo é desenvolvido então com base na Figura 13 e nas equações a seguir.

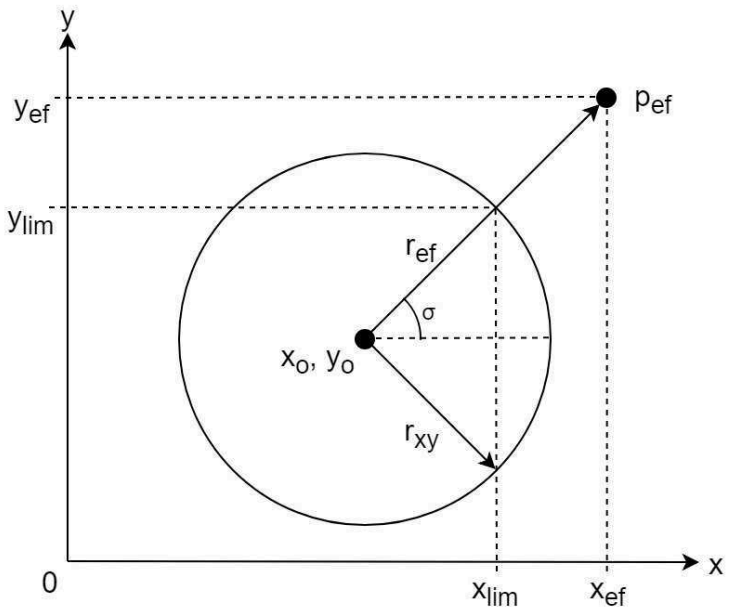
$$p_r = p_{ef} - p_o = \begin{bmatrix} x_{ef} - x_o \\ y_{ef} - y_o \end{bmatrix} = \begin{bmatrix} x_r \\ y_r \end{bmatrix} \tag{3.1}$$

$$\sigma = atan2(y_r, x_r) \tag{3.2}$$

$$p_{lim} = \begin{bmatrix} x_{lim} \\ y_{lim} \end{bmatrix} = \begin{bmatrix} r_{xy} cos(\sigma) \\ r_{xy} sen(\sigma) \end{bmatrix} \tag{3.3}$$

$$dp = \begin{bmatrix} dx \\ dy \end{bmatrix} = \begin{bmatrix} x_r - x_{lim} \\ y_r - y_{lim} \end{bmatrix} \tag{3.4}$$

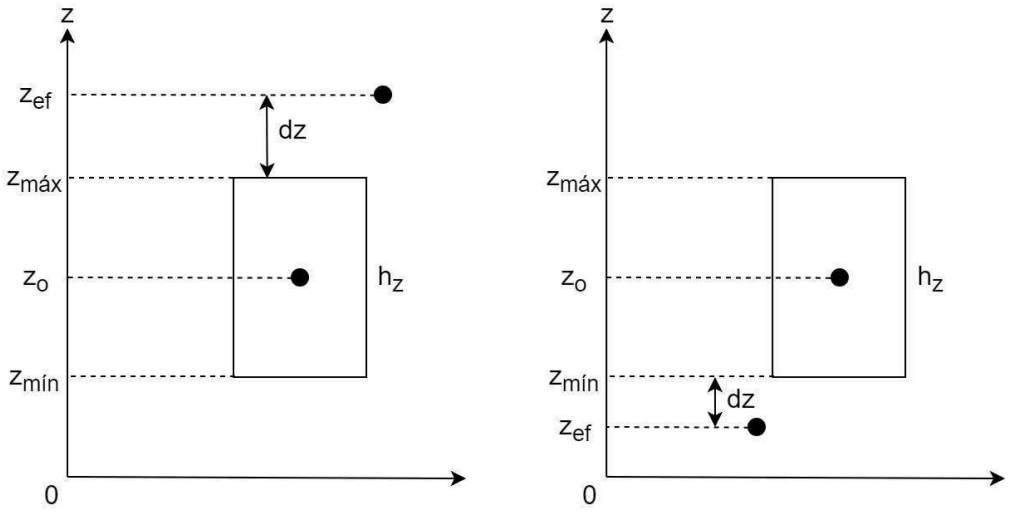
Figura 13 – Monitoramento em x e y.




Fonte: do autor.

Para o eixo $z$, o monitoramento se torna bem mais simples, uma vez que é apenas verificado se a posição $z_{ef}$ do efetuador final está dentro dos limites $z_{max}$ e $z_{min}$ de altura do cilindro, de acordo com a Figura 14.

Figura 14 – Monitoramento em z.




Fonte: do autor.

Os deslocamentos $dx$, $dy$ e $dz$ não são considerados para calcular a velocidade do manipulador, apenas o deslocamento total nos três eixos em cada ciclo de amostragem. É feito isso pelo fato de o monitoramento do tempo ser feito nas medições iniciais e finais da posição do ciclo, então a velocidade no intervalo é a mesma velocidade de deslocamento dentro do zona de restrição, não importando se o efetuador final iniciou o ciclo fora da zona e terminou dentro dela, ou vice-versa.

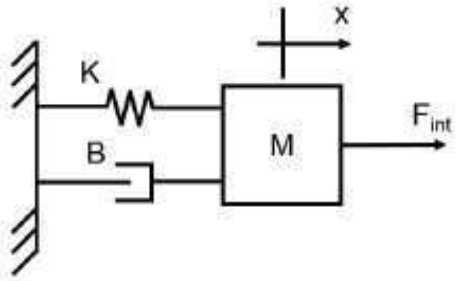
## 3.2 CONTROLE DE IMPEDÂNCIA/ADMITÂNCIA

Este método de controle foi proposto inicialmente por Craig e Raibert (1979), onde o sistema acoplado robô-ambiente é considerado como um sistema de impedância, onde as entradas são variáveis fluxo, como velocidade e corrente elétrica, e as saídas são variáveis de força e torque, que então compõem a potência do sistema. No caso da admitância têm-se o inverso, as entradas são representadas pelas variáveis de força e torque, e a saída pelas variáveis representando o fluxo. Neste tipo de controle, o sistema é tratado como um sistema amortecido de segunda ordem (por exemplo um sistema massa mola), e normalmente a variável de controle é o fluxo (velocidades) e a variável controlada é a força (LAHR et al., 2016). Já Villani e Schutter (2008) definem o controle de impedância como um método de controle indireto de força, uma vez que a variável a ser controlada é a força/torque, enquanto a ação de controle é feita nas variáveis de posição e velocidade.

De uma forma simplificada, o manipulador é tratado como uma impedância ou admitâcia, ou seja, um sistema de segunda ordem composto por uma massa, uma mola e um amortecedor, conforme a Figura 15, com a implementação do algoritmo se dando de forma virtual, onde se calcula posição necessária para atingir a força de contato desejada, ou o contrário, onde se regula o torque atuante nas juntas, para compensar o erro de posição desejada do efetuador final.

A força de interação dos sistemas $F_{int}$ pode ser descrita pela Equação 3.5, onde $M$ é a massa do sistema, $B$ é o amortecimento e $K$ é a rigidez. Esses parâmetros constantes se relacionam respectivamente, coma a aceleração, velocidade e deslocamento, resultando em parcelas individuais da força de interação.

Figura 15 – Sistema mecânico de segunda ordem.




Fonte: Lahr et al. (2016)

$$M\ddot{x}(t) + B\dot{x}(t) + Kx(t) = F_{int}(t) \tag{3.5}$$

Neste trabalho, o controle de impedância é utilizado em duas aplicações diferentes. Tanto na simulação em tempo contínuo como no tempo discreto, as reações do ambiente virtual dinâmico são obtidas através de uma impedância doa ambiente. Já a segunda aplicação é utilziada apenas na simulação no tempo discreto, onde o dispositivo mestre possui uma admitância (inverso da impedância) através do qual ele reage com movimento a partir de estímulos de força, oriundos da medição de um sensor de força.

### 3.2.1 Admitância do mestre

Tanto a operação quanto a realimentação virtual de força para o operador dependem da admitância do definida para o mestre. A manipulação do dispositivo mestre se dá da seguinte forma: o operador manipula o efetuador final do robô, no qual está instalado o sensor de força. Este sensor mede as forças em cada uma das três direções, $x$, $y$ e $z$, reagindo aos esforços com movimento, que é resultado da admitância.

O mestre deve reagir aos estímulos de força $F_m$ apenas com velocidade de deslocamento $\dot{x}_m$[1], ou seja, a impedância do mestre possui apenas o coeficiente de amortecimento $B_m$, conforme a Equação 3.6. O coeficiente de rigidez $K_m$ é desconsiderado para que o mestre permaneça na posição atual caso não existe nenhuma força atuando no sensor de força acoplado ao efetuador final do mestre.

$$B_m\dot{x}_m(t) = F_m(t) \tag{3.6}$$

### 3.2.2 Impedância do ambiente

A realimentação virtual de força se origina na interação do mestre com o ambiente virtual de operação. A posição atual do mestre é monitorada, e então é calculado o deslocamento do mesmo na zona restritiva, que por sua vez possui uma impedância conhecida. O ambiente é considerado como fixo, ou seja, não é considerado o parâmetro de massa na impedância do ambiente, que é composta então apenas pela rigidez $B_a$, que reage a velocidade de movimento relativa do manipulador $\dot{x}_a$ dentro do ambiente virtual, e rigidez $K_a$, que reage ao deslocamento relativo $x_a$ do manipulador dentro do ambiente. Portanto a força de reação do ambiente $F_a$ é calculada conforme a Equação 3.7.

$$B_a\dot{x}_a(t) + K_a x_a(t) = F_a(t) \tag{3.7}$$

Considerando que a força $F_m$ é a força $F_s$ lida no sensor menos a força de realimentação $F_a$ do ambiente, pode-se substituir a Equação 3.8 na Equação 3.6, resultando na Equação 3.10.

$$F_m(t) = F_s(t) - F_a(t) \tag{3.8}$$

$$B_m\dot{x}_m(t) = F_s(t) - F_a(t) \tag{3.9}$$

$$B_m\dot{x}_m(t) = F_s(t) - K_a x_a(t) - B_a\dot{x}_a(t) \tag{3.10}$$

A equação 3.10 é referente a uma situação na qual existe total transparência na realimentação de força. Normalmente o vetor de forças de realimentação pode ser multiplicado por um vetor de proporção $S$, para aumentar ou diminuir a impedância transmitida para o mestre, com a finalidade de aumentar a sensibilidade háptica, ou aumentar a segurança do operador humano, respectivamente. Porém desta maneira o sistema já não opera de forma totalmente transparente, uma vez que a realimentação de força não é composta pela força real de interação do robô com o ambiente.

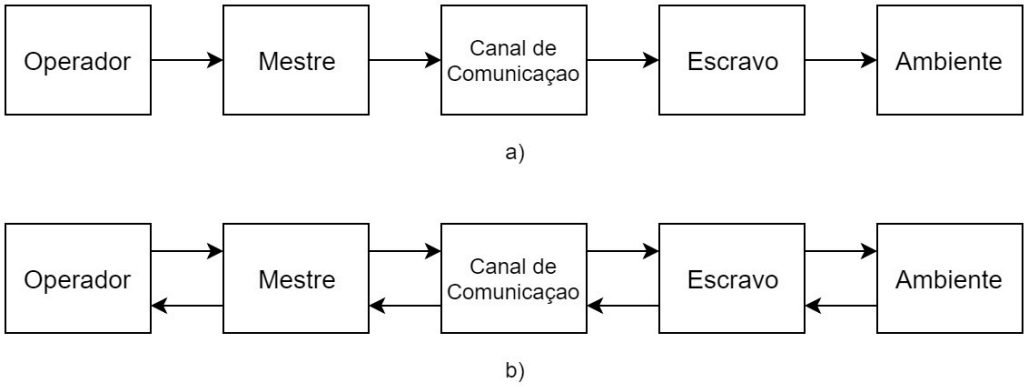
[1] Aqui $x$ representa uma posição ou um deslocamento unidimensional qualquer, não se referindo especificamente para a coordenada $x$ de um sistemas de coordenadas $xyz$.

# 4 SIMULAÇÃO DE TELEOPERAÇÃO HÁPTICA

Teleoperação consiste em um operador local enviando comandos para um dispositivo remoto, com o intuito de realizar uma tarefa específica em um ambiente distante do operador (NATH; TATLICIOGLU; DAWSON, 2009). Sistemas de teleoperação normalmente possuem dispositivos mestres, que comandam a operação, e dispositivos escravos, que executam a tarefa remotamente, e que são interligados pro um meio de comunicação, de acordo com Anderson e Spong (1989) e que podem ser vistos na Figura 16(a).

Sistemas bilaterais de teleoperação por sua vez, possuem a característica de que não apenas o mestre exerce influência, de forma remota, sobre o meio de operação do escravo, mas as reações e sensações sentidas pelo escravo no ambiente de operação são transmitidas exata ou proporcionalmente ao mestre e operador (PERERA; ABEYKOON, 2014), conforme mostra a Figura 16(b). Neste contexto são utilizados os dispositivos hápticos, que se tornam os dispositivos mestres no sistema de teleoperação bilateral, e são responsáveis por transformar o toque humano em comandos, e também por transmitir realimentação de toque ao operador humano (MCLAUGHLIN; SUKHATME; HESPANHA, 2001).

Figura 16 – Sistemas de Teleoperação: a) Unilateral; b) Bilateral.




Fonte: do autor.

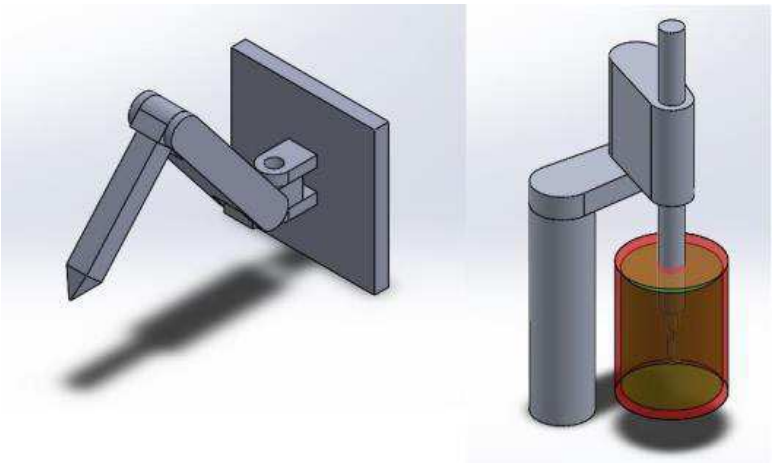
Durante o desenvolvimento do trabalho foram feitas simulações para compreender melhor os conceitos de teleoperação bilateral de robôs, de uma forma genérica. Foram modelados então dois robôs de diferentes configurações e características construtivas para serem usados em um sistema de teleoperação bilateral ideal, ou seja, onde a transferência de comando de força e posição entre mestre e escravo é instantânea, e não estão sujeitos às restrições de tecnologias reais de controle de robôs.

Foi utilizado um mecanismo em série com três juntas rotativas como dispositivo mestre, enquanto o dispositivo escravo é um SCARA de três graus de liberdade. A operação do mestre é feita por deslocamento do efeatuador final, cuja posição é virtualmente mapeada pela cinemática direta e enviada como comando para o escravo depois de ser modificada por uma matriz de transformação. A realimentação de força do sistema ocorre quando o escravo interage com um ambiente virtual modelado conforme o Capítulo 3.

## 4.1 MODELAGEM CINEMÁTICA DO MESTRE E ESCRAVO

Como o objetivo da simulação era obter uma teleoperação apenas da posição dos robôs (três graus de liberdade), foi escolhido um configuração de robô espacial com três graus de liberdade para ser o mestre. Já o dispositivo escravo escolhido foi o SCARA, por possuir uma cinemática simples, espaço de trabalho espacial e três graus de liberdade. A Figura 17 mostra os modelos em CAD feitos no *software* SolidWorks do mestre (esquerda) e escravo (direita).

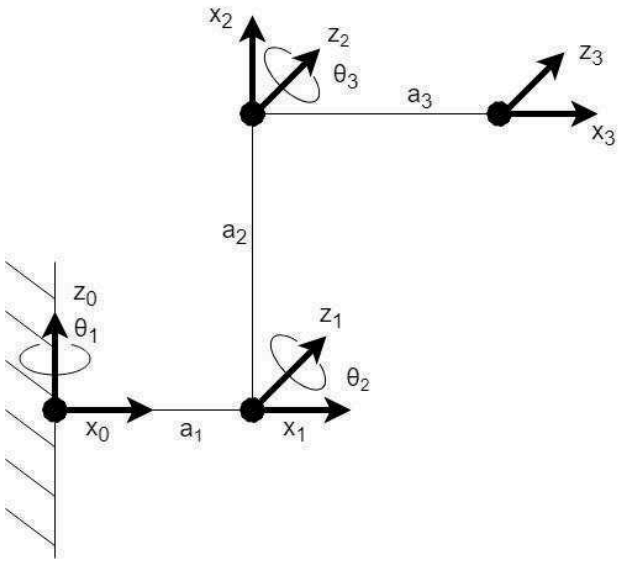
Figura 17 – Modelos em CAD do mestre e escravo.



Fonte: do autor.

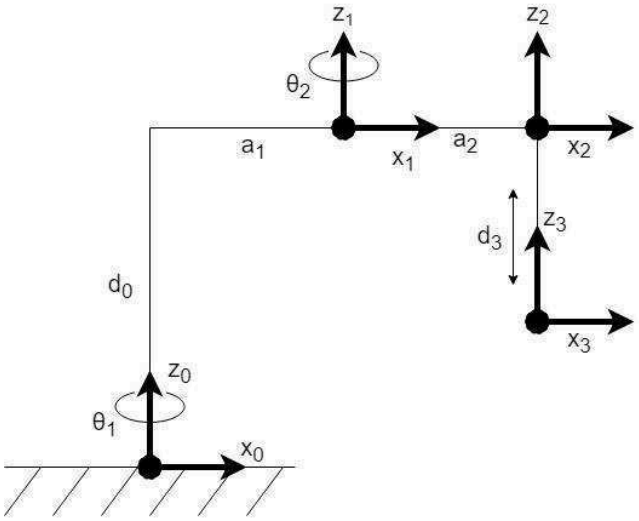
Para a obtenção dos modelos cinemáticos, foram obtidos os paramâmetros para o método de Denavit-Hartenberg (D-H), para cada um dos robôs, seguindo a metodologia de Siciliano et al. (2010) e com base nas configurações mostradas nas Figuras 18 e 19. Os parâmetros obtidos são mostrados nas Tabelas 18 e 19.

Figura 18 – Configuração do mestre.




Fonte: do autor.

Figura 19 – Configuração do escravo.




Fonte: do autor.

Tabela 1 – Parâmetros D-H do mestre.

| Junta (i) | $d_i$(mm) | $a_i$(mm) | $\alpha_i$ | $\theta_i$ |
|---|---|---|---|---|
| 1 | 0 | 30 | -$\pi$/2 | $\theta_1$ |
| 2 | 0 | 200 | 0 | $\theta_2$ |
| 3 | 0 | 200 | 0 | $\theta_3$ |

Fonte: do autor.

Tabela 2 – Parâmetros D-H do escravo.

| Junta (i) | $d_i$(mm) | $a_i$(mm) | $\alpha_i$ | $\theta_i$ |
|---|---|---|---|---|
| 1 | 665 | 250 | 0 | $\theta_1$ |
| 2 | 0 | 250 | 0 | $\theta_2$ |
| 3 | $d_3$ | 0 | 0 | 0 |

Fonte: do autor.

$$q_m = \begin{bmatrix} \theta_{1m} \\ \theta_{2m} \\ \theta_{3m} \end{bmatrix} \tag{4.1}$$

$$q_e = \begin{bmatrix} \theta_{1e} \\ \theta_{2e} \\ d_{3e} \end{bmatrix} \tag{4.2}$$

A partir da definição dos parâmetros, foi possível obter as matrizes de transformação dos robôs, de acordo com a Equação 4.3, com as quais se calcula a matriz de transformação homogênea entre dois elos subsequentes do robô, utilizando os parâmetros D-H, conforme explicado em Siciliano et

al. (2010). A multiplicação sequencial das matrizes permite se chegar a uma matriz de transformação que relaciona a pose do efetuador final com o sistema de coordenadas da base do robô, conforme a Equação 2.2. Como ambos os robôs possuem três juntas e três elos, suas matrizes de transformação finais podem ser calculadas conforme a Equação 4.4.

$$A_i^{i-1} = \begin{bmatrix} c_{\theta_i} & -s_{\theta_i}c_{\alpha_i} & s_{\theta_i}s_{\alpha_i} & a_i c_{\theta_i} \\ s_{\theta_i} & c_{\theta_i}c_{\alpha_i} & -c_{\theta_i}s_{\alpha_i} & a_i s_{\theta_i} \\ 0 & s_{\alpha_i} & c_{\theta_i} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.3}$$

$$A_3^0 = A_1^0 A_2^1 A_3^2 \tag{4.4}$$

Aplicando 4.3 nas Equações 2.1 e 2.2, é possível obter os vetores de posição do mestre e do escravo, $p_m$ e $p_e$ respectivamente, demonstrados nas Equações 4.5 e 4.6. Essas equações são utilizadas no cálculo das matrizes jacobianas analíticas $J_m$ e $J_e$, cuja obtenção é explicada mais detalhadamente no Apêndice B.

$$p_m = \begin{bmatrix} x_m \\ y_m \\ z_m \end{bmatrix} = \begin{bmatrix} c_{\theta_{1m}}[30 + 200(c_{\theta_{2m}} + c_{\theta_{2m}}c_{\theta_{3m}} - s_{\theta_{2m}}s_{\theta_{3m}})] \\ s_{\theta_{1m}}[30 + 200(c_{\theta_{2m}} + c_{\theta_{2m}}c_{\theta_{3m}} - s_{\theta_{2m}}s_{\theta_{3m}})] \\ -200(s_{\theta_{2m}} + c_{\theta_{2m}}s_{\theta_{3m}} - s_{\theta_{2m}}c_{\theta_{3m}}) \end{bmatrix} \tag{4.5}$$

$$p_e = \begin{bmatrix} x_e \\ y_e \\ z_e \end{bmatrix} = \begin{bmatrix} 250(c_{\theta_{1e}} + c_{\theta_{1e}}c_{\theta_{2e}} - s_{\theta_{1e}}s_{\theta_{2e}}) \\ 250(s_{\theta_{1e}} + s_{\theta_{1e}}c_{\theta_{2e}} + c_{\theta_{1e}}s_{\theta_{2e}}) \\ 665 - d_{3e} \end{bmatrix} \tag{4.6}$$

O dispositivo mestre realiza o mapeamento da posição do efetuador final utilizando a cinemática direta do mecanismo da Equação 4.5. Essas coordenadas de posição são transformadas por uma matriz de escala $T$, para adequar a cinemática às características construtivas do escravo.

O escravo então recebe uma posição desejada $p_e$, que é a entrada para o cálculo da cinemática inversa, a fim de determinar as variáveis de junta necessárias para atingir essa posição. Como o comando cinemático do escravo é feito por posição e não por velocidade, não foi utilizado aqui a matriz inversa de $J_e$, pois as varíaveis de saída da operação utilizando a Equação 2.8 seriam de velocidade e não de posição. Em função disso foram utilizadas as equações de cinemática inversa do robô SCARA, adaptadas do modelo especificado em Golin (2002).

$$\begin{aligned} q_e(p_e) = \begin{bmatrix} \theta_{1e}(p_e) \\ \theta_{2e}(p_e) \\ d_{3e}(p_e) \end{bmatrix} &= \begin{bmatrix} \psi_1 - \psi_2 \\ cos^{-1}\left(\frac{x_e^2 + y_e^2 - a_{1e}^2 - a_{2e}^2}{2a_{1e}a_{2e}}\right) \\ z_e - d_{1e} \end{bmatrix} = \\ &= \begin{bmatrix} \psi_1 - \psi_2 \\ cos^{-1}[8x10^{-6}(x_e^2 + y_e^2) - 1] \\ z_e - 665 \end{bmatrix} \end{aligned} \tag{4.7}$$

$$\psi_1 = tan^{-1}\left(\frac{y_e}{x_e}\right) \tag{4.8}$$

$$\psi_2 = tan^{-1}\left(\frac{a_{2e}s_{\theta_{2e}}}{a_{1e}+a_{2e}c_{\theta_{2e}}}\right) = tan^{-1}\left(\frac{s_{\theta_{2e}}}{1+c_{\theta_{2e}}}\right) \tag{4.9}$$

## 4.2 FATORES DE ESCALA E TRANSFORMAÇÃO

Uma vez que o dispositivo mestre e escravo possuem configurações mecânicas diferentes é necessário que haja uma adaptação na transferência de força e movimento entre os dois agentes do sistema (PERERA; ABEYKOON, 2014). Em sistemas de teleoperação bilateral essas transformaçãoes são usadas, por exemplo, para aumentar a segurança do operador durante uma realimentação de força, a precisão e suavidade do escravo em tarefas delicadas, ou a capacidade de percepção do mestre, aumentar ou reduzir a área de trabalho absoluta do escravo, ou até mesmo alterar os sistemas de coordenadas.

Em função disso foi utilizada, para a transformação de posição do mestre para o escravo, uma matriz $T$ que multiplica a posição $p_m$, e um vetor de *offset* $\delta$ que é somado à essa multiplicação, conforme a Equação 4.10.

$$p_e = T\,p_m + \delta \tag{4.10}$$

No caso do sistema a ser simulado, por mais que mestre e escravo possuam configurações diferentes, a escala de posição foi mantida unitária, apenas com a inversão dos eixos $x$ e $y$, conforme a Equação 4.11. Já a o vetor de *offset* considerou um pequeno deslocamento do ponto de origem de operação, conforme visto na Equação 4.12, em função de que a modelagem utilizada para o SCARA determina o ponto de origem orignal como sendo a base do robô, posição que não pode ser mecanicamente alcançada em uma situação real que utilize esse tipo de robô.

$$T = \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.11}$$

$$\delta = \begin{bmatrix} 0,3 \\ 0 \\ 0,3 \end{bmatrix} \tag{4.12}$$

Como o o sistema de teleoperação bilateral demanda uma ação do escravo no mestre, foi modelado um sistema de realimentação de força, no qual o escravo opera em um ambiente virtual dinâmico, cujo modelo é explicado no Capítulo 3. As reações do ambiente virtual em relação ao escravo são enviadas como comandos de força para o escravo, que atua no suposto operador com forças impostas nos três eixos.

Para se obter essas forças no efetuador final do mestre, é feito o cálculo do torque necessário nas juntas rotativas, utilizando a Equação 2.9 para estática[1] de robôs manipuladores. O vetor de forças reais de reação $f_r$ transferido para o escravo é multiplicado por uma matriz diagonal $S$, gerando o vetor

[1] Nesta simulação as reações de força não foram calculadas pelas equações diâmicas, devido ao aumento na complexidade do modelo. Em função disso foi feita uma aproximação para o modelo estático, porém com a restrição de não utilizar velocidades e acelerações elevadas na simulação, de acordo com a suposição quase-estática (KAO; LYNCH; BURDICK, 2008).

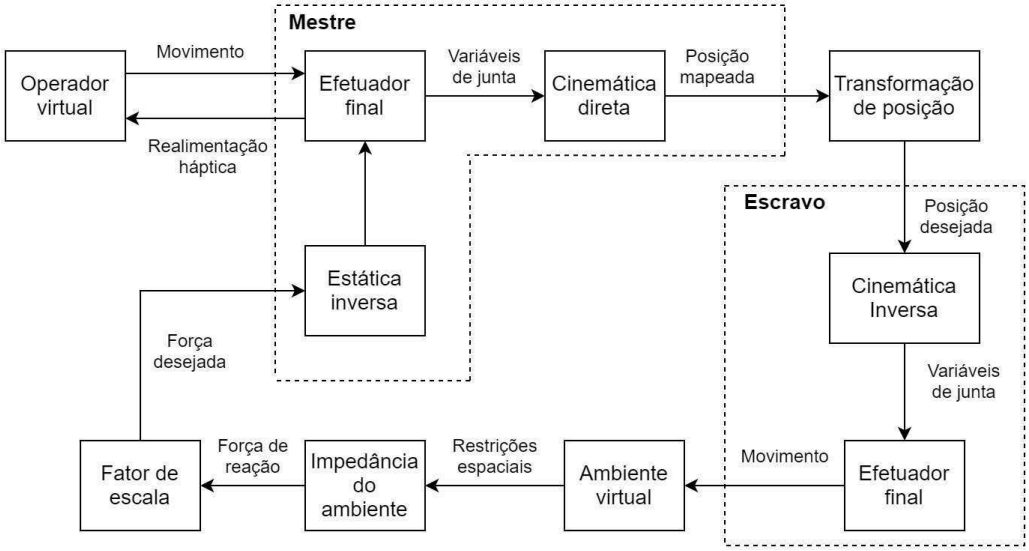
de forças desejadas no mestre $f_d$. Os torques necessários nas juntas são então calculados utilizando a Equação 2.9 e a transposta da matriz $J_m$ calculada anteriormente.

$$\tau_m = J_m^T f_d = J_m^T (S f_r) \tag{4.13}$$

## 4.3 AMBIENTE DE SIMULAÇÃO

Após ser feita a modelagem cinemática dos dispositivos, foram feitos modelos em CAD dos mesmos para possibilitar a visualização tridimensional dos mesmos no ambiente de simulação. O modelos em CAD gerados foram então exportados para o ambiente Simulink do *software* MatLab, através do pacote de ferramentas Simscape Mechanics, que permite a simulação de sistemas mecânicos, incluindo cadeias cinemáticas com diversas opções de atuação e medição de juntas. A Figura 20 mostra a arquitetura geral proposta do sistema de teleoperação desta primeira simulação.

Figura 20 – Arquitetura proposta de um sistema ideal de teleoperação bilateral.

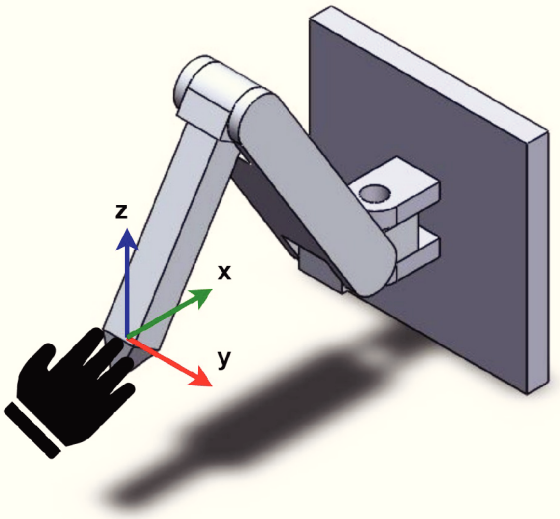



Fonte: do autor.

Mestre e escravo foram colocados no mesmo ambiente de simulação. O dispositivo mestre teve então suas juntas subatuadas, para que fosse possível a movimentação do mecanismo em função de atuações externas do operador. Para simular a operação manual do mestre, foram feitos comandos de deslocamento no efetuador final, conforme exemplificado na Figura 21, fazendo com que as juntas do mesmo rotacionassem para permitir o movimento do efetuador final. Utilizando então as equações de cinemática direta do mestre, é possível mapear a posição do mesmo. Esta posição atual do mestre é então remapeada para a posição desejada do escravo pela Equação 4.10 e enviada para este, que então calcula as variáveis de junta necessárias para se deslocar até tal posição, através das equações de cinemática inversa do escravo.

O efetuador final do escravo opera dentro de um ambiente virtual que é modelado conforme detalhado no Capítulo 3, sendo que o movimento dentro da área com restrições desencadeia forças de reação a serem transferidas para o dispositivo mestre. Essas forças são multiplicadas por uma matriz de escala $S$ e enviadas para o mestre como forças de atuação desejadas no operador, tendo-se então a

Figura 21 – Ilustração de manipulação do dispositivo mestre.

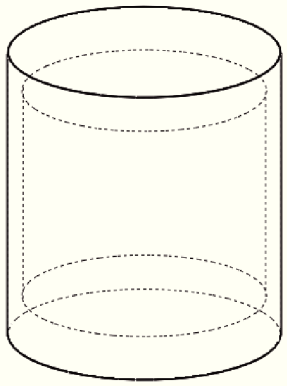



Fonte: do autor.

simulação de uma realimentação háptica. Os torques necessários nas juntas rotativas do mestre para realizar a força desejada no operador são calculadas através da estática.

O ambiente virtual de operação do escravo é baseado no que foi descrito na Seção 3.1.1, porém possui dois cilindros, conforme visto na Figura 22, um menor dentro de outro maior. O volume do cilindro menor, em linhas tracejadas, representa a zona operação livre do escravo, onde não há restrições nem reações de força, enquanto a zona fora do volume do cilindro maior, em linhas sólidas, representa a zona operação proibida. O volume que compreende a subtração do cilindro menor do maior é a zona indesejada.

Figura 22 – Zonas de operação do escravo.



Fonte: do autor.

As dimensões limites dos cilindros/zonas, de acordo com os parâmetros da Seção 3.1.1, são definidas na Tabela 3.

O ambiente virtual de operação, nas zonas indesejada e proibida, foi considerado como tendo comportamento dinâmico passivo, ou seja, ele reage ao sistema mecânico atuante através da sua

Tabela 3 – Dimensões das zonas de operação do escravo.

| Zona | $r_{xy}$ (mm) | $z_{min}$ (mm) | $z_{max}$ (mm) |
|---|---|---|---|
| Indesejada | 135 | 165 | 405 |
| Proibida | 150 | 120 | 450 |

Fonte: do autor.

impedância. Nessa simulação o ambiente foi concebido para possuir as características de rigidez $K$ e amortecimento $B$. Como a ideia é de que se trate de um ambiente fixo, o mesmo não possui o parâmetro de massa $M$.

A geração de reações ao movimento do efetuador final do escravo dentro das zonas indesejada e proibida depende de dois fatores: em qual das zonas o efetuador está e a direção na qual ele se desloca. Foi escolhida uma impedância, para o ambiente, com os parâmetros de $K$ = 1 kN/m e $B$ = 100 Ns/m.

Quando o efetuador se encontra fora dos limites da zona livre, é considerado que o mesmo está em contato com a zona indesejada ou proibida, e então é calculada a força de reação com base nas variáveis de movimento, conforme explicado na Seção 3.1.2. Se a zona de interação é a proibida, essa força de reação é multiplicada por 10. Uma pecualiaridade para essa simulação, é que o parmâmetro $B$ da impedância só é considerado se o efetuador se move, dentro das zonas indesejada ou proibida, afastando-se da origem do ambiente virtual de operação, caso contrário o ambiente é considerado como tendo apenas a rigidez $K$.

$$B = \begin{cases} 0, & \frac{dx}{dt} \leq 0 \\ 100, & \frac{dx}{dt} > 0 \end{cases} \tag{4.14}$$

Sob essa lógica, só ocorre reação à velocidade de deslocamento, se o mesmo estiver adentrando nas zonas restritivas, caso contrário, se o efetuador se move em direção à zona livre, a reação de força é relativa apenas à rigidez do ambiente. Portanto, o operador será capaz pode perceber, em função da realimentação háptica, que se os comandos de movimento que executa fazem o escravo operar dentro ou fora, ou se afastando ou se aproximando de regiões indesejadas ou proibidas de operação, bem como perceber a transição de uma região para outra.

## 4.4 RESULTADOS E CONSIDERAÇÕES

Esta simulação foi dividida em duas partes. Primeiramente foi feita a moviamentação do mestre nas três direções $x$, $y$ e $z$. O efetuador final foi movimentado com velocidade constante em uma direção defina por um vetor $v = [v_x\ v_y\ v_z]^T$. A trajetória então é do tipo retilínea, fazendo o efetuador final se deslocar pelo ambiente virtual, saindo da zona livre, adentrando a zona indesejada e indo até a zona proibida, onde fica parado por alguns segundos, e então retorna a origem do ambiente virtual, com a mesma velocidade em módulo. O ponto inicial da simulação, que também é considerada a origem do ambiente virtual, é definido na Equação 4.15, em relação ao sistema de cordenadas base do robô

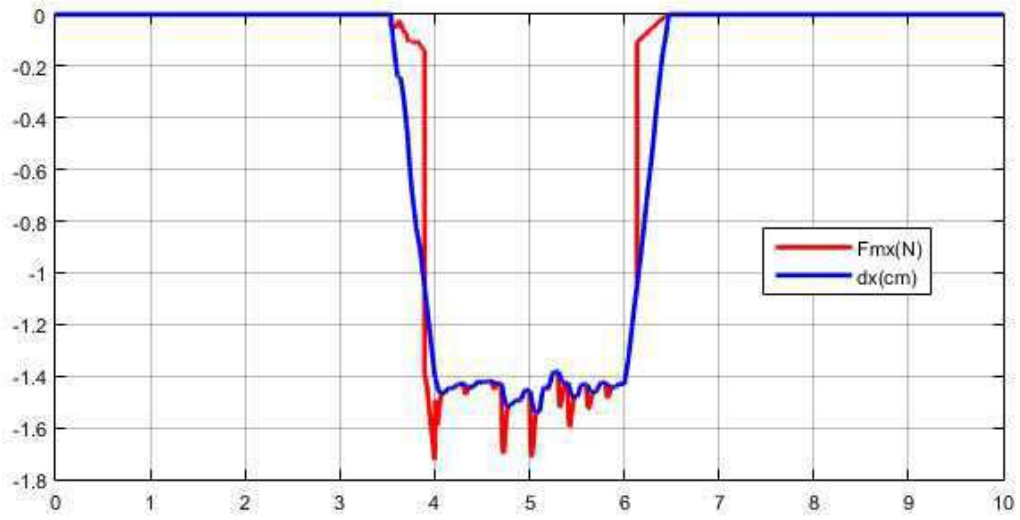
escravo.

$$ {}^{o}p_{base} = \begin{bmatrix} {}^{o}x_{base} \\ {}^{o}y_{base} \\ {}^{o}z_{base} \end{bmatrix} = \begin{bmatrix} 300 \\ 0 \\ 300 \end{bmatrix} \tag{4.15} $$

Para aproximar o comportamento de um operador humano, foi adicionado ao comando de deslocamento do efetuador final do mestre um ruído (em cada componente *xyz*) de 100 Hz com valores aleatórios de amplitude de $\pm 1$ mm. Esse sinal é então é filtrado por um filtro de segunda ordem para atenuar velocidades muito grandes, uma vez que os valores são gerados de forma discreta a cada ciclo da simulação.

Os resultados da primeira parte dessa simulação estão mostrado nas Figuras 23, 24 e 25 , que mostram a relação de deslocamento do escravo dentro das zonas proibidas e indesejadas em função do tempo (em segundos). A Figura 26 mostra então os torques[2] nas juntas do mestre para os comandos de reação de força em função do tempo.

Figura 23 – Deslocamento e forças de reação no eixo X.




Fonte: do autor.

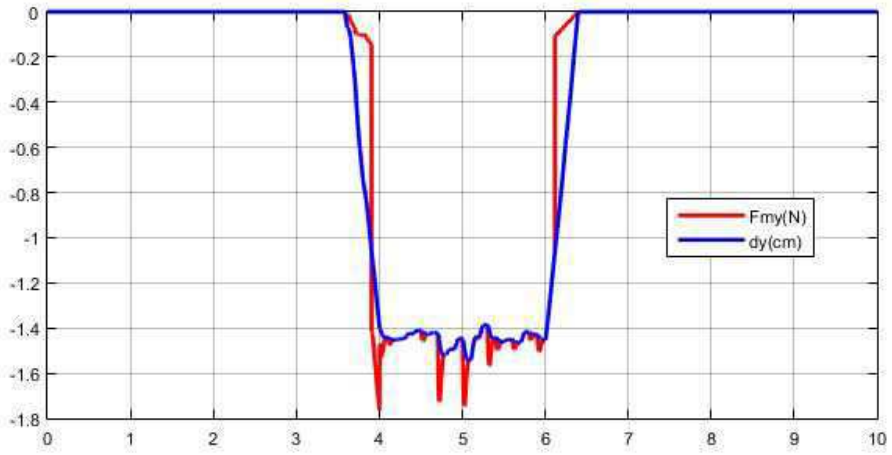
Em todos os gráficos é possível notar os momentos de transição entre as zonas livre, indesejada e proibida, pelas curvas de força de reação. Nos instantes em que o efetuador final se encontra parado, as oscilações de força são causadas pelo ruído inserido no sinal de deslocamento de operação do mestre.

A segunda parte dessa simulação contou com a presença de atrasos nas comunicações mestre-escravo e escravo-mestre, $t_1$ e $t_2$, respectivamente. Nesta etapa foi realizado um movimento senoidal no eixo $z$, fazendo com que o efetuador adentre nas zonas indesejadas e proibida. Foi aplicado o mesmo ruído de posição utilizado na primeira parte da simulação.

Os atrasos $t_1$ e $t_2$ são iguais, e tiveram quatro valores atribuídos para simular o efeito de atrasos diferentes no sistema. Os valores foram de 0, 50 ms, 250 ms e 500 ms, resultando em atrasos totais

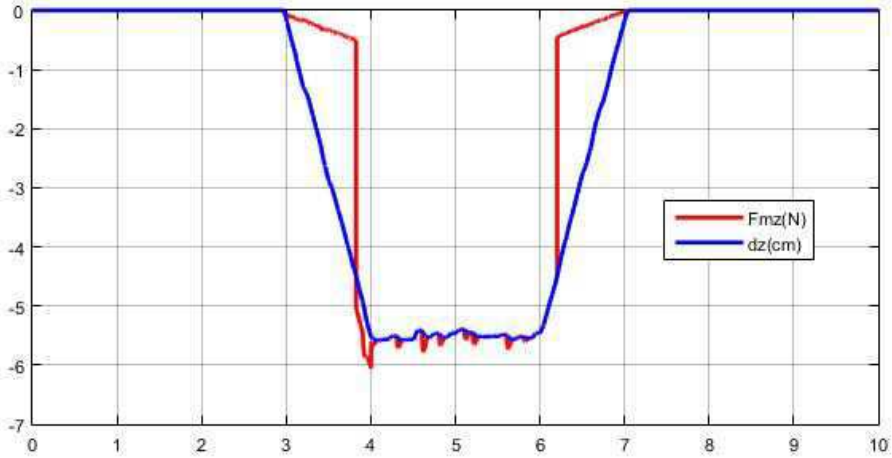
[2] Aqui são mostrados apenas os torques relativos as forças de reação e reaelimentação háptica. Os torques referentes a atuação do operador no efetuador final e de sustentação estática da massa dos elos foi desprezada nessa simulação.

Figura 24 – Deslocamento e forças de reação no eixo y.



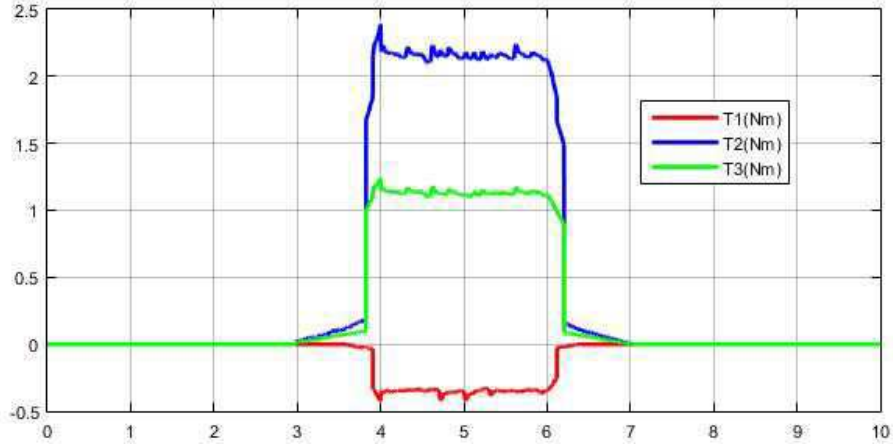

Fonte: do autor.

Figura 25 – Deslocamento e forças de reação no eixo z.




Fonte: do autor.

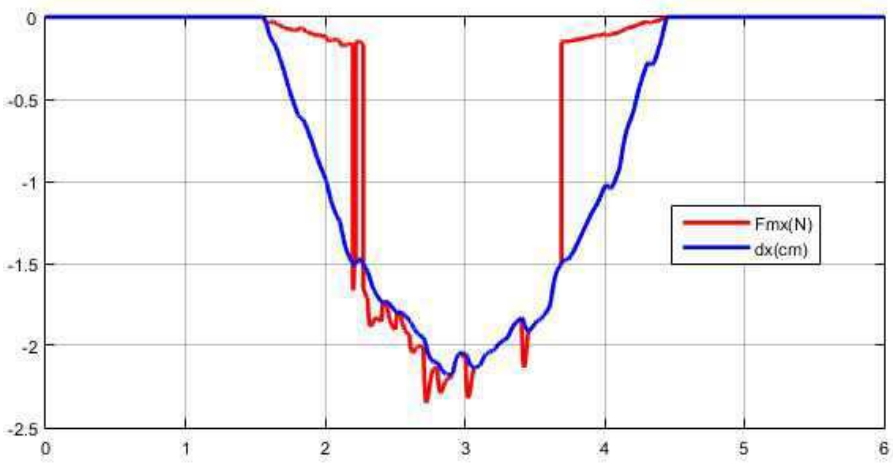
Figura 26 – Torques nas juntas do mestre.




Fonte: do autor.

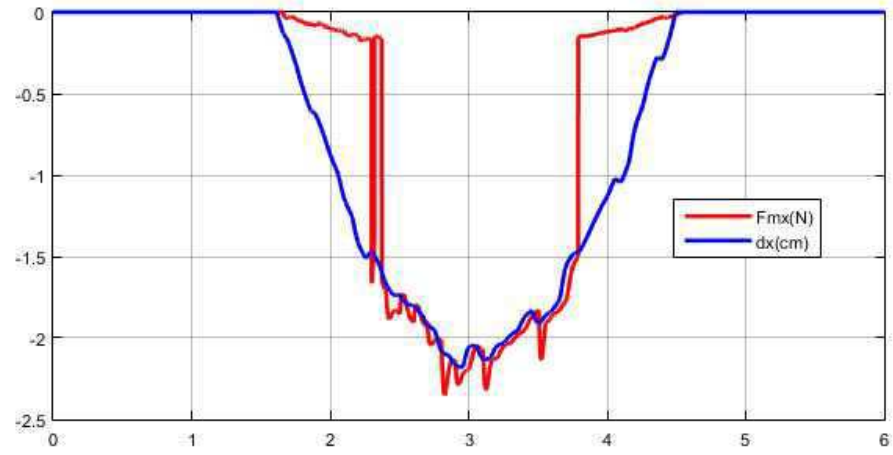
no sistema de 0, 100 ms, 500 ms e 1 s, respectivamente. Os resultados desta parte da simulação são mostrados nas Figuras 27 a 30.

Figura 27 – Resposta do mestre sem atraso.




Fonte: do autor.

Figura 28 – Resposta do mestre com 100 ms de atraso.

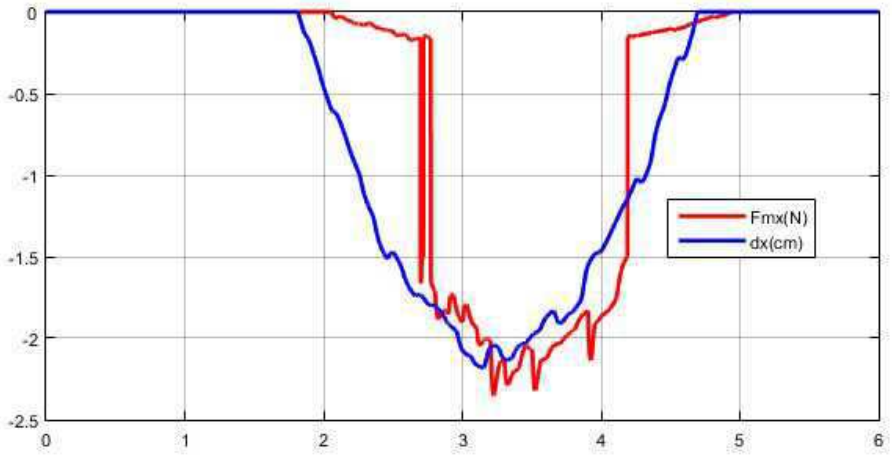



Fonte: do autor.

O comportamento do sistema nessa segunda parte da simulação é similar à primeira. É nítida a transição entre as zonas, porém desta vez existe uma oscilação maior de força no instante em que o escravo está na transição da zona indesejada para a zona proibida, causada pelo sinal de ruído injetado na posição. Já em relação ao atraso, não houve nenhum comportamento inesperado ou efeitos de maior importância, uma vez que o atraso na comunicação apenas atrasa a atuação das forças de reação no mestre. Isso se dá pelo fato dessa realimentação ocorrer em malha aberta, não exercendo nenhum influência na entrada do sistema que é o comando de posição. Caso contrário seriam esperados comportamentos instáveis do sistema.

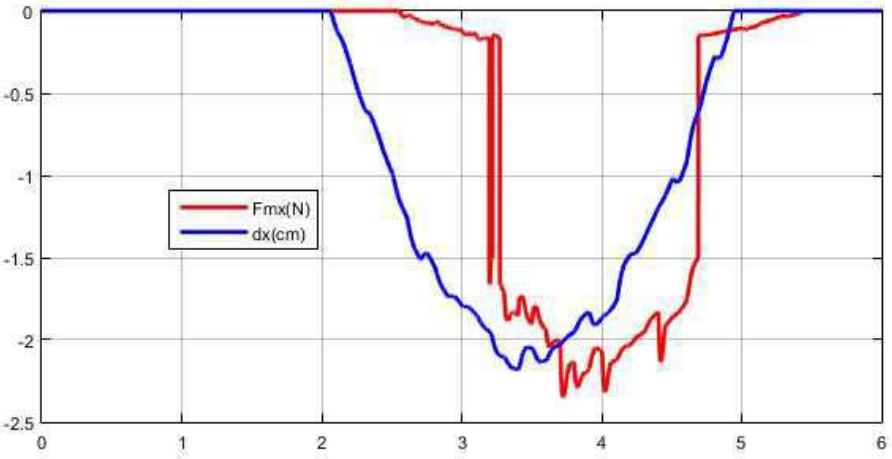
Por fim pode-se dizer que houve um comportamento satisfatório do sistema, mostrando que é interessante considerar esse método de bilateralidade em um sistema de teleoperação, dada a sua simplicidade de implementação, não exigindo a utilização de muitos sensores, apenas um monitoramento virtual do ambiente de operação. Claro que essa implementação deve ser aperfeiçoada a um algoritmo

Figura 29 – Resposta do mestre com 500 ms de atraso.




Fonte: do autor.

Figura 30 – Resposta do mestre com 1 s de atraso.




Fonte: do autor.

que permita seu funcionamento em sistemas reais, considerando as limitações de programação e de processamento de informações mecânicas, especialmente de movimento, em sistemas discretos, pois a esta simulação foi feita utilizando matemática contínua no tempo, e não discreta.

# 5 TELEOPERAÇÃO DE UM ROBÔ INDUSTRIAL

Neste capítulo é detalhado o segundo e o principal sistema de teleoperação proposto neste trabalho. A intenção dessa proposição é simular um sistema de teleoperação bilateral que utilize robôs industriais. No capítulo anterior é mostrado um sistema de comportamento ideal de teleoperação de robôs, onde há uma comunicação direta e instantânea tanto dos comandos de movimento nos sinais de realimentação. Aqui toda a formulação e modelagem do sistema são feitas no domínio do tempo discreto, e simulado em ambiente que possui atrasos de comunicação e controle, gerando efeitos diferentes daqueles vistos anteriormente.

Ao final do capítulo são mostrados e explicados os resultados obtidos. Também é feita uma estimação de um possível modelo linear do atraso de resposta do escravo aos comandos do mestre.

## 5.1 ROBÔ ABB IRB1600

O sistema de teleoperação proposto nesta etapa do trabalho utiliza o modelo virtual de um robô industrial da marca ABB modelo IRB1600, mostrado na Figura 31. Este robô possui seis graus de liberdade (DOF - *Degrees of Freedom*), com seis juntas rotativas, e foi projetado para tarefas clássicas de manipulação e robótica industrial, como montagem, soldagem, usinagem, limpeza, pintura e empacotamento (ABB, 2018a). Ele possui capacidade de manipular cargas nominais de até 10 kg, com um alcance de aproximadamente 1,2 m, conforme a Figura 32.

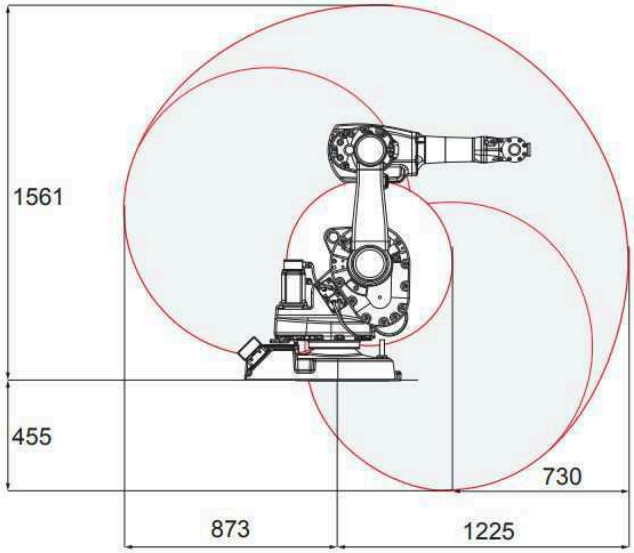
Figura 31 – Robô IRB1600.


Fonte: ABB (2018a).

### 5.1.1 Controlador IRC5

O modelo virtual do robô IRB 1600 é controlado por um controlador virtual também fabricado pela ABB, do modelo IRC5, visto na Figura 33. Este controlador possui a capacidade de controlar diversos robôs da marca ABB, e conta com ferramentas de programação de fácil utilização, permitindo a programação *online* e *offline* dos robôs. Ele possui a capacidade de controlar até quatro robô simul-

Figura 32 – Área de trabalho do Robô IRB1600.




Fonte: ABB (2018a).

Tabela 4 – Limitações cinemáticas do Robô IRB1600.

| Eixo | Limites de trabalho (°) | Velocidade máxima ($°/s$) |
|---|---|---|
| 1 | -180 até +180 | 150 |
| 2 | -63 até +136 | 160 |
| 3 | -235 até +55 | 170 |
| 4 | -200 até +200 (rev. de $\pm 190$) | 320 |
| 5 | -115 até +115 | 400 |
| 6 | -400 até +400 (rev. de $\pm 288$) | 460 |

Fonte: ABB (2018a).

taneamente através de uma linguagem própria da ABB de alto nível, desenvolvida para programação de robôs industriais, chamada de RAPID (ABB, 2018b).

Todas as tarefas de cálculo da cinemática inversa e direta, controle de posição, controle de força, controle de motores, e demais funções para o funcionamento seguro e confiável do robô, são realizados pelo controlador IRC5 a partir dos parâmetros e comandos definidos pela programação. Portanto não é necessário nenhum dispositivo ou controlador externo para realizar o controle mecânico e elétrico do robô em baixo nivel, cabendo ao controldor apenas receber comandos em alto nível de programação.

Neste trabalho não há um controlador IRC5 real. Todas as propriedades do controlador estão inseridas no ambiente de simulação RobotStudio da ABB. Ou seja, tanto os recursos quanto as limitações de um controlador IRC5 real estão disponíveis no *software* de programação e simulação *offline*.

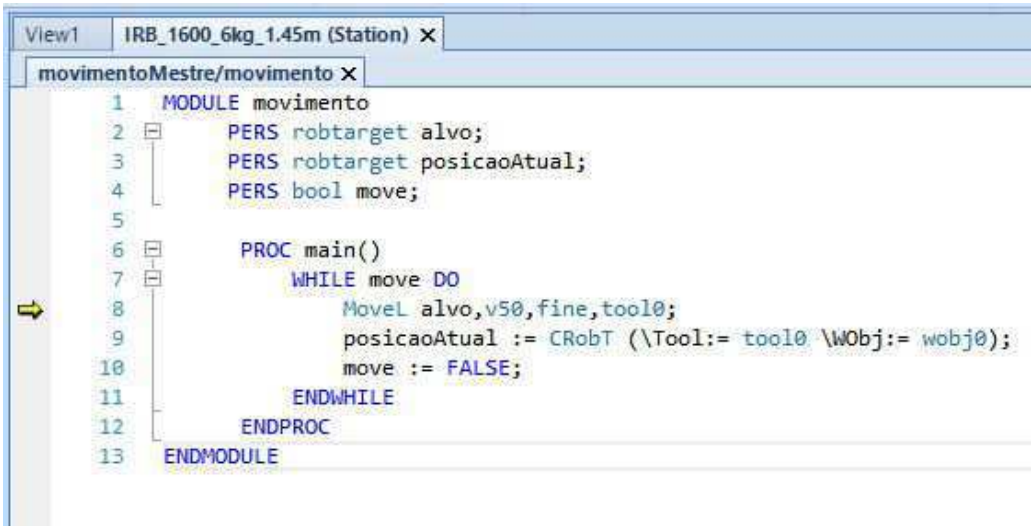
Figura 33 – Controlador IRC5.

Fonte: ABB (2018b).

### 5.1.2 RAPID e RobotStudio

Os robôs e controladores da ABB são programados utilizando uma linguagem de alto nível desenvolvida pela ABB, chamada de RAPID. Esta linguagem possui funções específicas para controle e programação de robôs, como comandos de posição, movimento e trajetória, leitura de posição do efetuador final e de juntas, leitura de sensores, comunicação com dispositivos externos via rede, etc. (ABB, 2007; ABB, 2010b). Um exemplo de programa pode ser visto na Figura 34, onde é ordenada a movimentação de um robô.

Figura 34 – Exemplo de programa em RAPID.




Fonte: *software* ABB RobotStudio.

Nas programações em RAPID utilizadas neste trabalho, as principais funções utilizadas foram (ABB, 2010b):

- *MoveL*: move o robô em linha reta da sua postura atual até uma postura alvo, com velocidade e

precisão pré-determinadas;

- *robtarget*: define uma postura alvo para o robô;
- *SocketCreate*: cria um *socket* para comunicação por TCP/IP;
- *SocketBind*: vincula um *socket* a uma porta e endereço IP para comunicação por TCP/IP;
- *SocketListen*: inicia a comunicação TCP/IP, "escutando"o emissor;
- *SocketSend*: envia dados pela comunicaçõ TCP/IP pré-estabelecida;
- *SocketReceive*: recebe dados pela comunicaçõ TCP/IP pré-estabelecida;
- *SocketClose*: desliga o *socket* pré-estabelecido.

O sistema de teleoperação proposto neste trabalho foi simulado em um *software* da ABB chamado RobotStudio, onde é possível realizar as mesmas ações de programação e controle dos robôs da ABB que são realizadas no ambiente físico. Este *software* permite simular todos os robôs, controladores e sensores da ABB, obtendo cenários fiéis à realidade. Também é possível criar ambientes físicos com os quais os robôs interagem, permitindo então a detecção de colisão, geração de trajetória e todas as outras aplicações de um ambiente real (ABB, 2010a).

Esses recursos permitem então que não seja necessária uma estação real de robô + controlador + ambiente para realizar ensaios e experimentos, permitindo que um sistema real seja testado com maior segurança e de forma mais rápida, facilitando então a implementação em um sistema real que utiliza exatamente a mesma programação e parâmetros do simulador. A interface de desenvolvimento e simulação do RobotStudio pode ser vista na Figura 35.

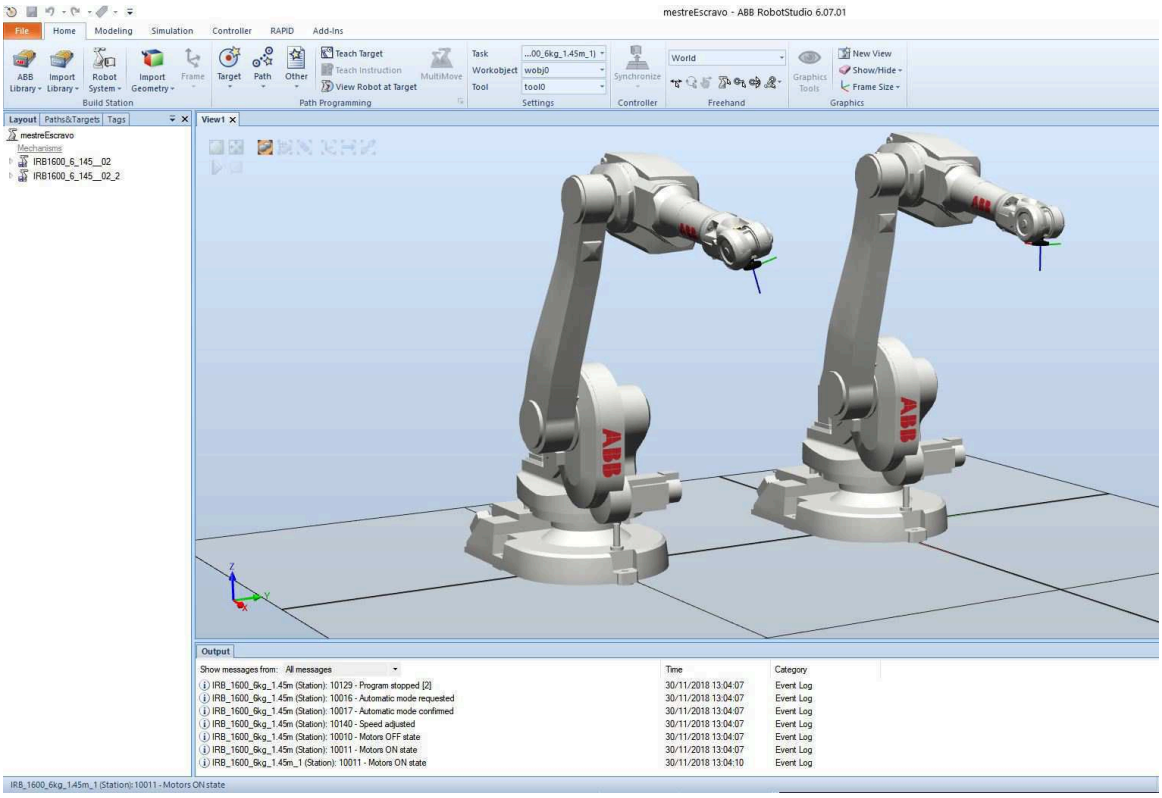
## 5.2 SENSOR DE FORÇA

O dispotivo utilizado na medição de força do sistema de teleoperação proposto neste capítulo é um Sensor de Força-Torque Multi-Eixos fabricado pela JR3 Inc. mostrado na Figura 36. Esse sensor é capaz de medir as forças nos eixos $x$, $y$ e $z$, bem como o torque em cada um desses eixos, sendo que tais medições são feitas através da estrutura metálica monolítica do sensor, que possui transdutores resistivos de tensão conectados em pontes de Wheastone para transformar as forças medidas em sinais elétricos.

Em aplicações reais o sensor é montado no punho do robô, entre o efetuador final e o elo mais extremo do manipulador, a fim de realizar a medição o mais próximo possível do ponto de contato do manipulador/ferramenta com o meio de operação. O sensor realiza a conversão do sinal analógico do transdutor para um sinal digital que é enviado via comunicação serial para uma placa PCI de aquisição instalada em um microcomputdaor. Neste trabalho o sensor não é montado no punho de um robô, mas é operado desacoplado a qualquer sistema. É utilizado apenas para gerar medições de força, para simular a comunicação das informações medidas ao sistema de controle do(s) robô(s).

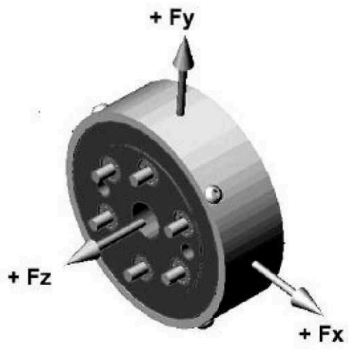
O modelo de sensor utilizado é 100M40A3 200N, que possui as características e limites de medição conforme a Tabela 5.

Figura 35 – Área de trabalho do ABB RobotStudio.




Fonte: *software* ABB RobotStudio.

Figura 36 – Sensor de força/torque JR3.




Fonte: JR3 Inc. (2003).

Para tornar possível a manipulação e o estímulo de força no sensor, foi fabricado em plástico um manípulo, preso mecanicamente ao sensor, permitindo uma interação mais ergonômica com o dispositivo. A montagem do manípulo ao sensor pode ser vista na Figura 37.

### 5.2.1 Aquisição de dados

Uma vez que o sensor possua uma saída digital, ele deve ser lido por um sistema computacional. A aquisição de dados é realizada através de um cabo RJ-11 que liga o sensor a uma placa PCI Express conectada a uma palca mãe de um PC do tipo Desktop.

Tabela 5 – Especificações técnicas do sensor de força-torque.

| Eixos | $X, Y$ | $Z$ |
|---|---|---|
| Faixa de medição de força (N) | ±200 | ±400 |
| Faixa de medição de torque (Nm) | ±20 | ±20 |
| Força máxima (N) | 1200 | 5100 |
| Torque máximo (Nm) | 110 | 87 |
| Rigidez (N/m) | 11,5x10^6 | 110x10^6 |
| Rigidez (N/rad) | 0,098x10^6 | 0,026x10^6 |
| Resolução (%) | ±1 | ±1 |

Fonte: JR3 Inc. (2003).

Figura 37 – Manípulo de operação.

Fonte: do autor.

O sistema de medição foi implementado através de uma aplicação em Sistema Operacional Linux, utilizando uma biblioteca específica para placas de aquisição de dados fornecido gratuitamente pela Comedi.org (COMEDI, 2012). Esta biblioteca disponibiliza funções necessárias para a interface entre a placa de aquisição do sensor JR3 e o Sistema Operacional. O código utilizado para a leitura do sensor foi baseado em linguagem C++, e pode ser visto no Apêndice A.

O sensor possui sete filtros com oito canais cada, totalizando 56 canais (0 - 55) de comunicação e dois canais de identificação, conforme visto na Tabela 7. Através dos filtros é possível realizar diferentes medições com períodos de amostragem diferentes, sendo que os dados são transmitidos do primeiro ao sexto canal, e os dois últimos canais não são utilizados, conforme visto na Tabela 6.

Tabela 6 – Quadro de dados dos filtros de medição.

| Canais do filtro | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| Informação | Fx | Fy | Fz | Mx | My | Mz | N/A | N/A |

Fonte: https://github.com/jhu-saw/sawJR3ForceSensor/wiki/JR3-Hardware.

Tabela 7 – Canais de comunição do sensor JR3.

| Canais | Filtro | Freq. de amostragem (Hz) |
|---|---|---|
| 0 - 7 | N/A | 8000 |
| 8 - 15 | 1 | 8000 |
| 16 - 23 | 2 | 500 |
| 24 - 31 | 3 | 125 |
| 32 - 39 | 4 | 31,25 |
| 40 - 47 | 5 | 7,813 |
| 48 - 55 | 6 | 0,4883 |
| 56 | Número do modelo | |
| 57 | Nímero serial | |

Fonte: https://github.com/jhu-saw/sawJR3ForceSensor/wiki/JR3-Hardware.

## 5.3 TOPOLOGIA DE COMUNICAÇÃO E CONTROLE

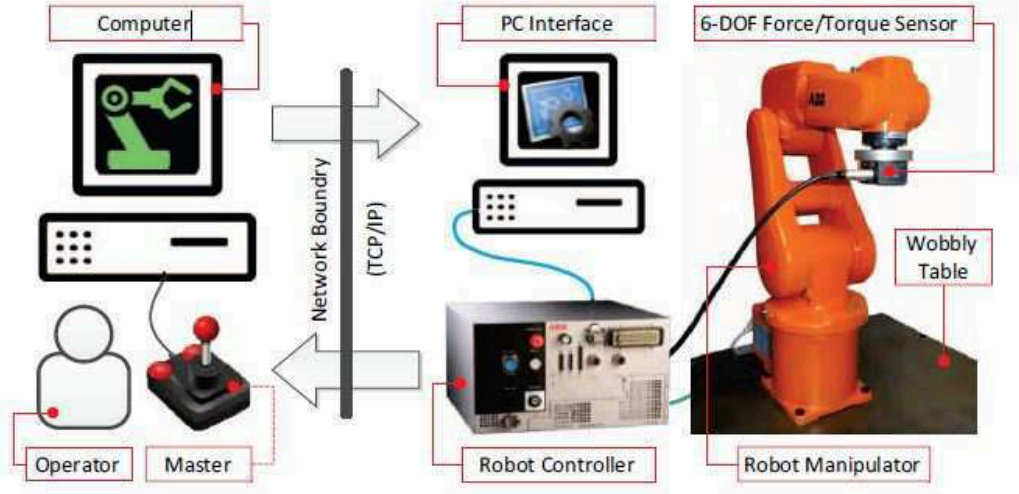
Dalvand e Nahavandi (2014) propuseram um sistema melhorado de teleoperação sem acesso ao baixo nível de controle, que é o caso do sistema proposto neste trabalho, no qual são utilizados robôs industriais com tecnologias próprias e fechadas de controle e acionamento, conforme visto na Seção 5.1. Esse sistema consiste de um dispositivo háptico, manipulado por um operador humano, que se comunica com uma interface PC que realiza o a comunicação com o robô escravo, que possui um sensor de força instalado no seu efetuador final. A topologia desse sistema pode ser visto na Figura 38.

Nesse trabalho foram utilizados, na parte do robô escravo, comandos de movimentação ponto a ponto, atavés do uso do comando *MoveL* da linguagem RAPID de programação de robôs da ABB. O deslocamento do escravo foi feito utilizando tanto velocidades fixas de deslocamento ponto a ponto quanto velocidades variáveis, no comando *MoveL*.

Com base no modelo proposto em Dalvand e Nahavandi (2014), foi concebida a topologia de teleoperação para este trabalho, com duas modificações principais. Primeiramente, ao invés de um dispositivo háptico, foi utilizado um robô industrial como dispositivo mestre, operado através de um sensor de força, para que o controle e bilateralidade ocorresse através de sinais de força e admitância. A segunda diferença se deu na não utilização de um sensor de força no robô escravo, sendo que todo o monitoramento da interação do escravo com o ambiente é realizado de forma virtual, com o mapeamento da posição do escravo dentro do ambiente e restrições virtuais pré-estabelecidas.

Por fim, a estrutura do sistema de teleoperação proposto neste trabalho, foi baseada na con-

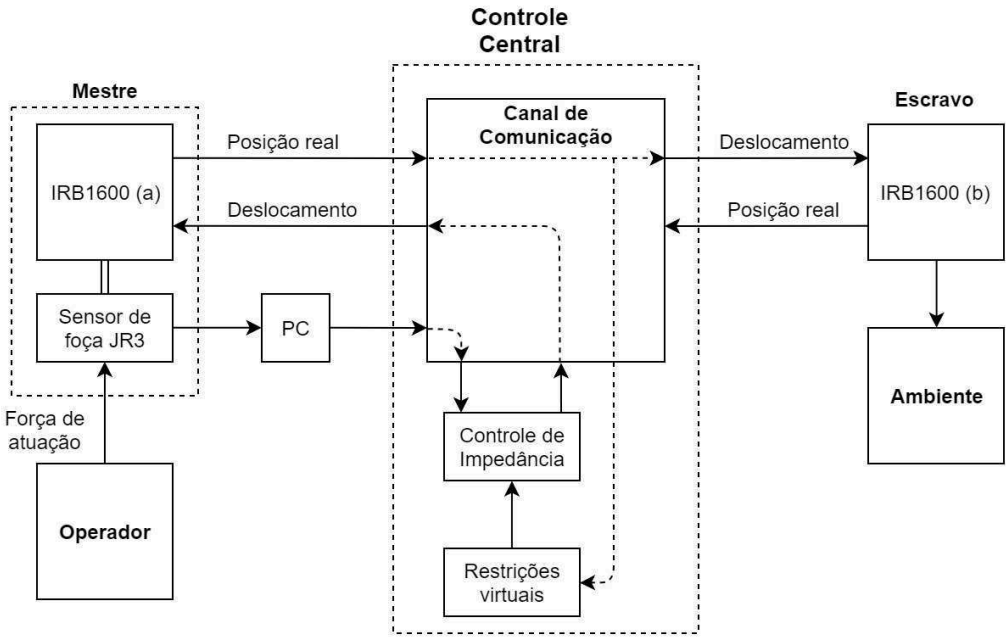
Figura 38 – Modelo de teleoperação de robô industrial.




Fonte: Dalvand e Nahavandi (2014).

cepção de uma topologia controle centralizado, que realiza as funções de comunicação, comando e controle em alto nível dos dispositivos que compõe o sistema, como pode ser visto na Figura 39. Esta topologia conta com um computador central que executa todas as funções de gerenciamento e controle do sistema: comunicação entre mestre e escravo; cálculo da impedância utilizada na manipulação do mestre; monitoramento da operação do mestre no ambiente e nas zonas com restrições virtuais.

Figura 39 – Sistema de teleoperação proposto.




Fonte: do autor.

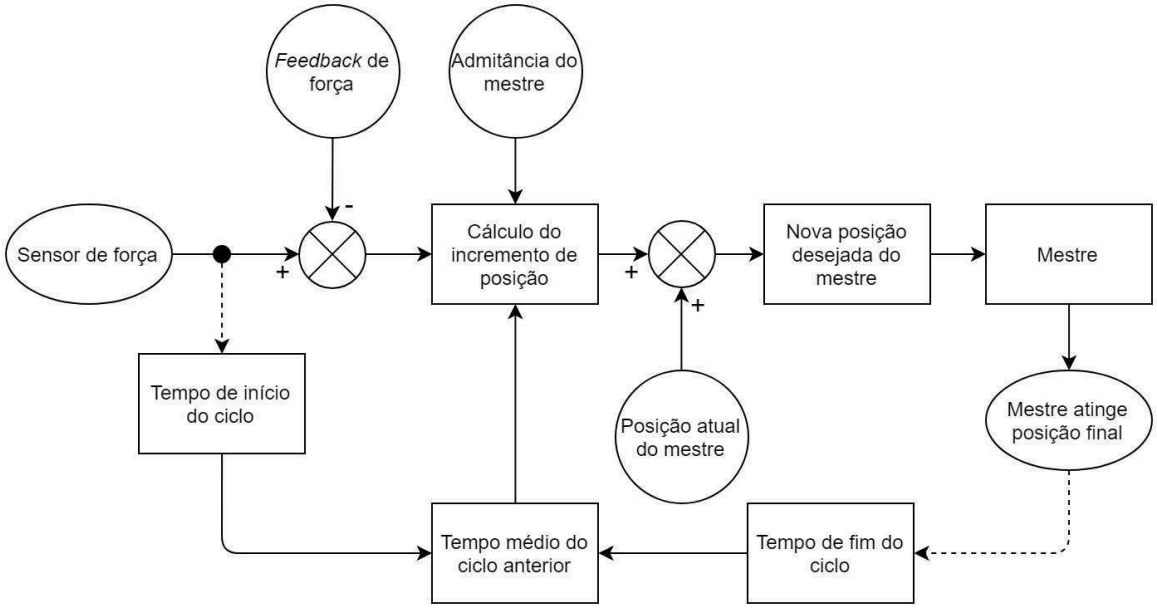
## 5.4 OPERAÇÃO DO MANIPULADOR MESTRE

Como foram utilizados robôs manipuladores industriais da ABB nas funções de dispositivo mestre e escravo, eles não possuem nenhuma funcionalidade ou módulo de controle que adapte o robô para interagir diretamente com um operador. Em função disto, foi desenvolvido um sistema para o robô mestre reagir a forças medidas no sensor de força JR3.

Através da admitância do mestre, explicada na Seção 3.2.1, é calculada a velocidade e o deslocamento desejados em cada uma das três direções x, y e z. Porém na aplicação desejada, a programação em RAPID não permite que seja enviado um comando de velocidade pura, ou seja, o comando de movimentação sempre requer uma posição alvo final do movimento, pois a programação da trajetória dos robôs, em RAPID é feita ponto a ponto. Portanto foi desenvolvido um algoritmo que calcula o deslocamento desejado em função da velocidade calculada pela admitância do mestre.

A Figura 40 mostra o funcionamento do algoritmo, a partir do qual é calculado o incremento de posição do mestre na forma de uma aproximação discreta, baseada na Equação 3.9, substituindo deslocamento unidimensional $x$ pelo vetor de deslocamento tridimensional $p$. A Equação 5.1 consiste no modelo discreto, onde existe um tempo de amostragem variável $t_s$, que por sua vez também é a duração do ciclo anterior de operação do mestre, utilizado para calcular as velocidades discretizadas.

Figura 40 – Fluxograma de operação do mestre.




Fonte: do autor.

$$B_m v_m(k) = F_s(k) - K_a p_a(k) - B_a p_a(k) \tag{5.1}$$

$$v_a(k) = \frac{p_a(k) - p_a(k-1)}{t_s(k)} \tag{5.2}$$

Os blocos do diagrama da Figura 40 foram implementados seguindo o que foi definido no Capítulo 3. O código em Python utilizado para a operação do mestre está incorporado ao código da central de controle do sistema e é mostrado no Anexo C.

Como $K_a$, $B_a$ são constantes previamente definidas e $p_a$ e $v_a$ são variáveis, obtidas e calculadas, respectivamente, pelo controle central, é possível calcular a parcela de deslocamento desejado referente à força $F_s$ medida no sensor, subtraída da força $F_a$ oriunda da interação do mestre com a impedância ambiente de operação. Essa força resultante é aplicada então na admitância do mestre, resultando no incremento de posição $\Delta p_m(k)$ que é calculado conforme a Equação 5.5, onde $t_s$ é o tempo do ciclo anterior de operação do mestre, e as demais variáveis são explicadas na Seção 3.2.

$$v_m(k) = \frac{\Delta p_m(k)}{t_s(k)} = \frac{p_m(k+1) - p_m(k)}{t_s(k)} \tag{5.3}$$

$$p_m(k+1) = p_m(k) + \Delta p_m(k) \tag{5.4}$$

$$\Delta p_m(k) = B_m^{-1} t_s(k)[F_s(k) - K_a p_a(k) - B_a v_a(k)] \tag{5.5}$$

A posição final desejada $p_m(k+1)$ é enviada então para o controlador do robô mestre, que está programado em RAPID para comandar o mestre para se movimentar utilizando a função *MoveL*. Um dos parâmetros desta função é a velocidade de deslocamento entre o ponto inicial e final, que pode ser um valor constante ou variável a cada ciclo. A aproximação discreta proposta tem como objetivo tornar os cálculos e processamentos da central de controle mais simples e rápidos.

Com exceção da variável de *cycletime* $t_s$, todas as outras variáveis que compõem as equações descritas nesse modelo são na verdade vetores e matrizes de componentes $x$, $y$ e $z$, que são explicados mais detalhadamente no Apêndice D e na Tabela 8.

Tabela 8 – Variáveis de operação do mestre.

| Variável | Descrição |
|---|---|
| $p_m$ | Vetor de posição do manipulador mestre |
| $v_m$ | Vetor de velocidade do manipulador mestre |
| $p_a$ | Vetor de posição do manipulador mestre no ambiente virtual |
| $v_a$ | Vetor de velocidade do manipulador mestre no ambiente virtual |
| $F_s$ | Vetor de forças medidas no sensor |
| $B_m$ | Matriz de constantes de amortecimento do mestre |
| $B_a$ | Matriz de constantes de amortecimento do ambiente virtual |
| $K_a$ | Matriz de constantes de rigidez do ambiente virtual |

Fonte: do autor.

## 5.5 COMANDO DO MANIPULADOR ESCRAVO

A operação do manipulador escravo é feita de forma semelhante ao do mestre, porém com maior simplicidade, uma vez que o escravo apenas obedece a comandos de posição enviados pelo mestre e o controle central. Por se tratar de um robô idêntico ao mestre, a movimentação do escravo é comandada da mesma forma que o mestre, através de linguagem RAPID e com um controlador ABB.

É utilizado o mesmo comando de *MoveL*, cuja posição alvo é a posição atual real medida no mestre (não é necessário um escalonamento, fator de proporção, ou transformações de coordenadas

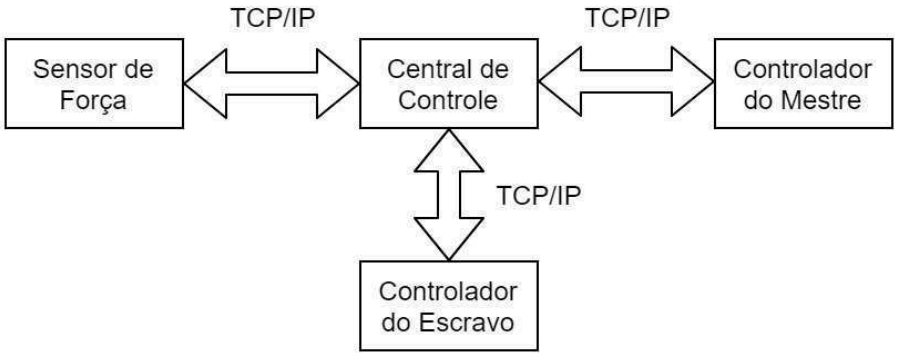
e orientação, pois os robôs são iguais). O parâmetro de velocidade de deslocamento do comando também é o mesmo utilizado no mestre, a fim de tentar reproduzir o movimento do mestre da forma mais precisa possível.

A estimação da força de interação do escravo com o ambiente é feita considerando o movimento do mestre ao invés do escravo, pois permite uma reação mais rápida por parte do sistema bilateral às interações virtuais do escravo com o ambiente de operação, no caso em que o ambiente de operação é conhecido, quanto a sua geometria e características físicas. Se o monitoramento fosse feito com base no escravo, a realimentação estaria sempre com o atraso de um ciclo de operação.

## 5.6 COMUNICAÇÃO

Toda a comunicação entre os subsistemas foi centralizada na central de controle, em uma topologia em estrela. A arquietetura proposta é do tipo mestre e escravo, onde a central de controle atua como mestre, e o sensor de força, controladores do robô mestre e robô escravo operam como escravos. Aqui o conceito de robô mestre/escravo é diferente do de dispositivo mestre/escravo em redes de comunicação. A arquitetura de comunicação pode ser vista na Figura 41.

Figura 41 – Comunicação entre os subsistemas de teleoperação.




Fonte: do autor.

São consideradas no sistema de comunicação proposto apenas as camadas 4, 2 e 1 do sistema OSI (KUROSE; ROSS; ZUCCHI, 2007). A camada 3 (rede) não é considerada pois não existe comunicação direta entre os dispositivos escravos, ou seja, todos eles se comunicam direta e exclusivamente com a central de controle. Desta forma não se fazem necessários os recursos dessa camada, como por exemplo, o roteamento e controle de fluxo de mensagens, uma vez que apenas o mestre pode solicitar comunicação. Abaixo seguem as definições das camadas 4, 2 e 1 para o sistema proposto:

- Camada 4 - Transporte: é utilizado o protocolo TCP/IP para a comunicação com os escravos (pontas da topologia estrela). Cada conexão ocorre através de uma porta física separada, sendo que cada uma possui um *socket* e cada estação possui um IP diferente.

- Camada 2 - Enlace: conforme citado anteriormente, o enlace entre os elementos é feito em uma topologia estrela, ou seja, os dispositivos escravos não se conectam entre si. O MAC (*Medium Access Control*) é feito de forma simples pela central de controle, onde a comunicação com cada escravo é feita de forma cíclica e sequencial, sem a concorrência de acesso ao meio de

comunicação, uma vez que a comunicação ocorre mediante solicitação do mestre (central de controle).

- Camada 1 - Física: para a comunicação com o PC que realiza a leitura e condicionamento do sensor de força, é utilizada conexão com cabo *Ethernet* RJ-45. Como a conexão entre a central de controle e os robôs é feita através de um simulador que opera na mesma máquina que a central de controle, não existe camada física na comunicação nessas conexões.

A central de controle é programada em linguagem Python, cujo código pode ser visto no Apêndice C. O PC que faz a leitura do sensor de força é programado em C++ em um sistema operacional Linux. O código de aquisição de dados do sensor de força pode ser visto no Apêndice A.

## 5.7 SIMULAÇÕES E RESULTADOS

Nesta seção são mostrados e discutidos os resultados das simulações feitas com base no sistema de teleoperação proposto neste capítulo. Os robôs manipuladores IRB1600 (mestre e escravo) são simulados no *software* de programação *offline* RobotStudio da ABB, permitindo se ter uma noção aproximada de como seria o comportamento de robôs reais.

### 5.7.1 Parâmetros de simulação

As simulações foram divididas em três fases, cada uma com foco e objetivos diferentes quanto a análise dos dados coletados:

A. Manipulação do mestre sem realimentação: aqui as simulações foram feitas com o intuito de analisar o comportamento básico do mestre quando estimulado por forças do operador, e sem considerar o manipulador escravo ou a realimentação de força virtual;

B. Teleoperação sem realimentação: essas simulações foram feitas com o intuito de avaliar o desempenho do escravo em relação aos comandos oriundos do mestre, especialmente em relação ao seguimento do movimento;

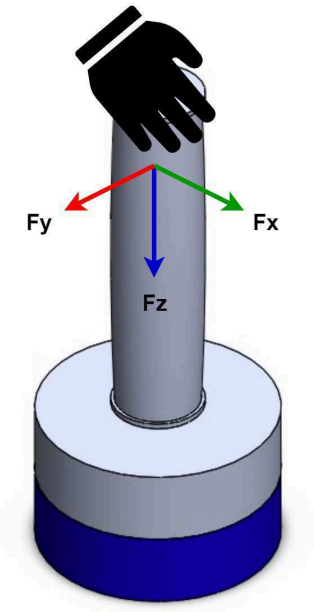
C. Teleoperação com realimentacão: aqui é testado o sistema como um todo, onde se analisa os efeitos da realimentação virtual de força, e seus efeitos no que diz respeito à sensação háptica do operador.

Em todas as simulações, foram utilizados valores fixos de admitância do mestre arbritrados em função de testes anteriores às simulações, nos quais foram definidos de forma empírica valores que produziriam velocidades de magnitude adequadas para as simulações. Uma admitância muito baixa resultaria em velocidades muito altas fariam com que o sistema de controle dos robôs bloqueasse os movimentos por questões de segurança. Já o uso de admitâncias muito altas eixigiriam forças muito elevadas para pequenos deslocamentos.

Para gerar os comandos de movimento do mestre, foram aplicadas forças reais manualmente ao conjunto do do sensor de força e manípulo de operação, conforme visto na Figura 42. O vetor de forças $F_s$ é medido pelo sensor e enviado para o controle central, que calcula a velocidade de

deslocamento do mestre utilizando a Equação 5.5. As forças aplicadas nas primeiras duas fases de simulações são aleatórias e variantes nos três eixos, enquanto na última fase é aplicada uma força constante no eixo $z$ apenas.

Figura 42 – Aplicação de forças no sensor.




Fonte: do autor.

Tabela 9 – Parâmetros das simulações.

| Sim. | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Fase | A | A | B | B | C | C |
| $v_{pm}$ (mm/s) | 50 | 100 | 50 | 50-500 | 80 | 80 |
| $v_{pe}$ (mm/s) | - | - | 50 | 50-500 | 80 | 80 |
| $B_m$ (Ns/mm) | 0,5 | 0,5 | 0,4 | 0,4 | 0,4 | 0,4 |
| $B_a$ (Ns/mm) | - | - | - | - | 0,1 | 1 |
| $K_a$ (N/mm) | - | - | - | - | 0,1 | 1 |

Fonte: do autor.

Nas simulações foram utilizadas velocidades parametrizadas $v_{pm}$ e $v_{pe}$, do mestre e escravo respectivamente, na função *MoveL* da linguagem RAPID, sendo que em uma das simulações é testado um sistema que envia para esse comando um parâmetro de velocidade diferente a cada ciclo de operação. Já os parâmetros do $B_a$ e $K_a$ do ambiente virtual de operação foram definidos apenas para as simulações onde se utilizava a realimentação de força. Os parâmetros de simulação estão mostrados na Tabela 9.

Na Simulação 4, onde há uma faixa de velocidade para $v_m$ e $v_s$, a velocidade enviada para ser utilizada no comando *MoveL* é baseada na velocidade calculada pela admitância do mestre. Nas demais simulações, foram arbitrados valores de velocidade que fossem maiores que as velocidades

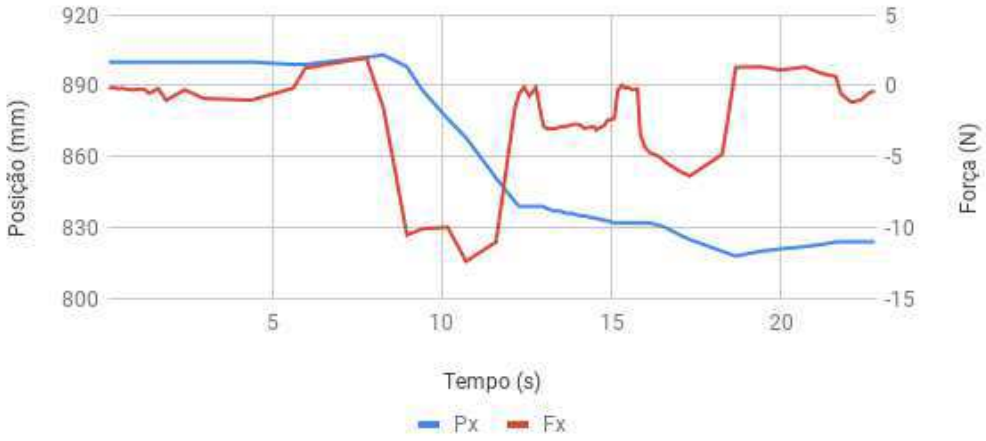
esperadas na saída da admitância do mestre. Quanto aos estímulos de força, nas fases A e B, foram aplicadas forças aleatórias no sensor de força, e na fase C forma aplicadas forças constantes (mas ainda assim com uma leve oscilação) ao sensor.

Na fase C, onde o sistema interage no ambiente virtual, as dimensões da mesma, de acordo com a Seção 3.1.1, são definidas como: $r_{xy}$ = 150mm, $z_{max}$ = 1100 mm e $z_{min}$ = 700.

### 5.7.2 Manipulação do mestre sem realimentação

Primeiramente são analisadas as respostas no deslocamento do mestre em função da força de contato no sensor. Foram gerados gráficos dos resultados nos eixos $x$,$y$ e $z$, porém nesta etapa da simulação são mostradas apenas os gráficos do eixo $x$, que são suficientes para mostrar objetivamente os resultados. Os demais gráficos podem ser vistos no Apêndice E.

Figura 43 – Simulação 1 - Posição no eixo x.

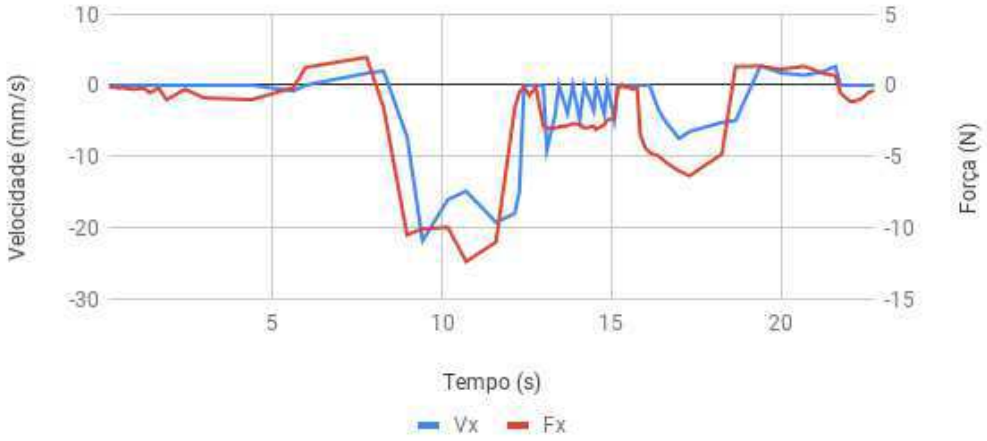



Fonte: do autor.

Inicialmente é possível notar que o deslocamento do mestre nos três eixos ocorre proporcionalmente às forças medidas no sensor, de acordo com a Figura 43. Já na Figurs 44 é possível visualizar o desempenho do sistema à velocidade de deslocamento do mestre. As velocidades foram calculadas com base no deslocamento entre dois ciclos subsequentes e seus *cycletimes*. É importante salientar que a velocidade mostrada nos gráficos é a velocidade calculada aproximandamente após o movimento do robô, e não a velocidade comandada pelo controle central.

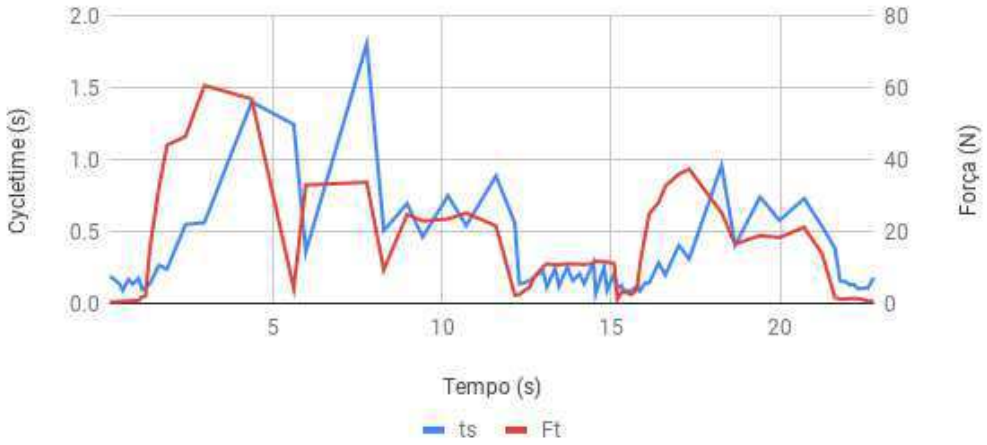
Ainda na Figura 44 notam-se dois comportamentos principais: em primeiro lugar, percebe-se um atraso nas curvas de velocidade quando a força atinge valores mais altos, isso se dá pelo fato de que o sistema de controle do mestre só inicia um novo ciclo de comando de deslocamento quando, no ciclo anterior, o mestre atinge a posição desejada. Como o manipulador efetua deslocamentos maiores para forças maiores, e como no controlador ABB e na linguagem RAPID a velocidade de deslocamento ponto-a-ponto foi constante (no valor de 50 mm/s) nesta simulação, é natural que ele demore mais a chegar ao ponto final, fazendo com que exista um atraso na próxima leitura de força e comando de deslocamento, como pode ser visto na Figura 45, onde é possível relacionar os picos de *cycletime* com os instantes em que o sensor teve as maiores medições de força. Resumidamente, quanto maior a velocidade de deslocamento do manipulador mestre, maior o *cycletime* de controle.

Figura 44 – Simulação 1 - Velocidade no eixo x.




Fonte: do autor.

Figura 45 – Simulação 1 - Relação entre força e atraso.

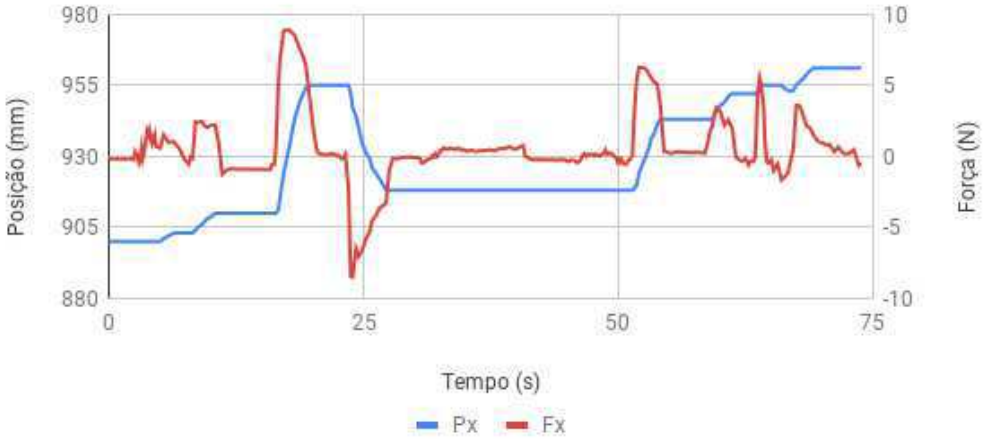



Fonte: do autor.

Como o *cycletime* é proporcional às velocidades de deslocamento do mestre, surgiram oscilações de velocidade nos pontos onde as forças no sensor estavam próximas de zero. Isso se dá pelo fato de que quando a força é mínima, o *cycletime* é mínimo, porém ainda assim ocorre um pequeno deslocamento do manipulador, e então a velocidade de deslocamento é calculada através da Equação 5.3. Com isso surgem pequenos picos de velocidade devido ao cálculo ser baseado no *cycletime* passado, e não reflete precisamente a velocidade desejada calculada através da admitância do mestre.

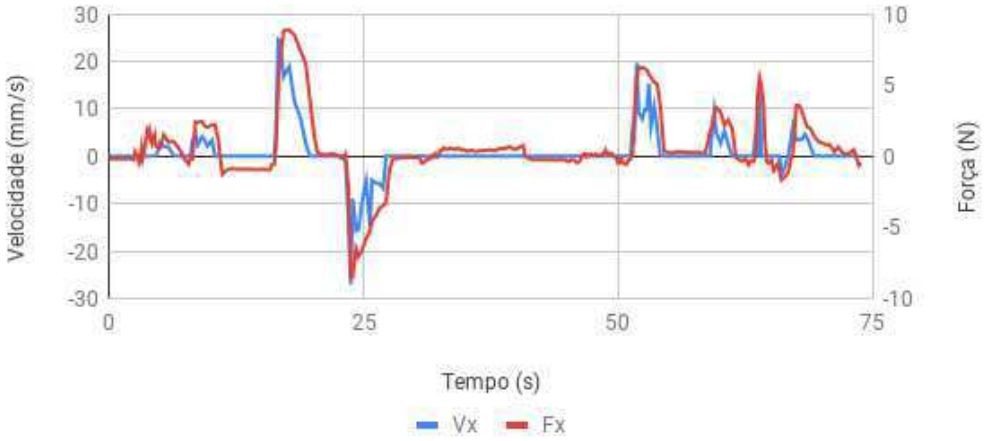
A Simulação 2 foi feita de forma similar à Simulação 1, porém desta vez utilizando a velocidade constante de 100 mm/s para o parâmetro de velocidade do comando *MoveL*. A Figura 46 mostra o deslocamento do mestre quando é submetido a estímulos de força. A velocidade de deslocamento do mestre em relação às forças pode ser vista na Figura 47.

Figura 46 – Simulação 2 - Posição no eixo x.




Fonte: do autor.

Figura 47 – Simulação 2 - Velocidade no eixo x.

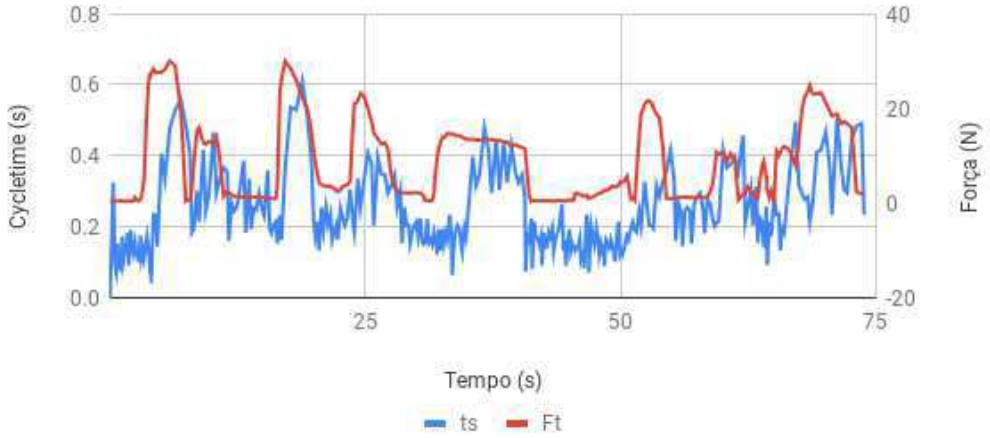



Fonte: do autor.

Já na Figura 48 é possível verificar as forças de estímulo e a variação do *cycletime* do mestre. Nota-se que, da mesma forma que ocorreu na Simulação 1, o *cycletime* tem uma curva que se assemelha, em formato, com a curva da soma dos módulos das forças nos três eixos. Novamente constata-se, que quanto maior a força de estímulo, maior o atraso do ciclo. Porém nesta simulação, como foi utilizado o parâmetro de velocidade do comando *MoveL* constante de 100 mm/s (para o mestre e escravo), os valores de *cycletime* atingiram valores mais baixos em relação à Simulação 1.

Em contrapartida, notou-se que a curva das velocidades do mestre, calculadas da mesma forma que foi feito na Simulação 1, atingiram valores maiores e mais instáveis, gerando curvas mais abruptas e oscilantes, conforme pode ser visto na Figura 47. Isto certamente se deu pela velocidade do *MoveL*, que faz o mestre se movimentar relativamente mais rápido nos incrementos de posição oriundos do cálculo da admitância do mestre, gerando também *cycletimes* menores, e por fim faz com que as velocidades calculadas sejam maiores.

Figura 48 – Simulação 2 - Relação entre força e atraso.

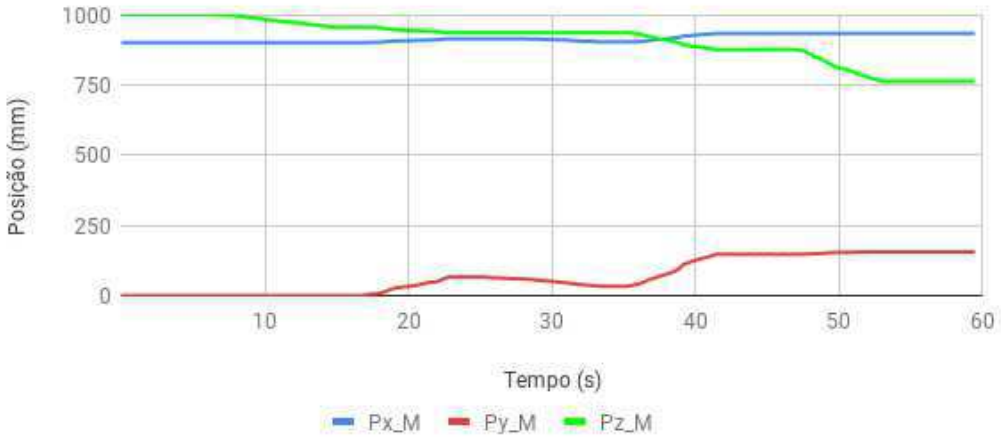



Fonte: do autor.

### 5.7.3 Teleoperação sem realimentação

As posições do mestre e escravo, resultado da Simulação 3, podem ser vistas nas Figuras 49 e 50, respectivamente. Já as velocidades do mestre e escravo estão mostradas nas Figuras 51 e 52, respectivamente.

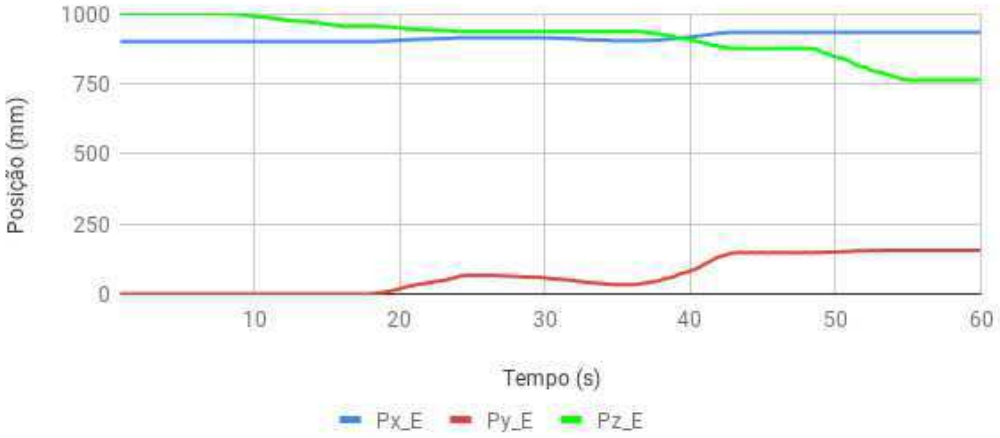
Figura 49 – Simulação 3: Posição do Mestre.




Fonte: do autor.

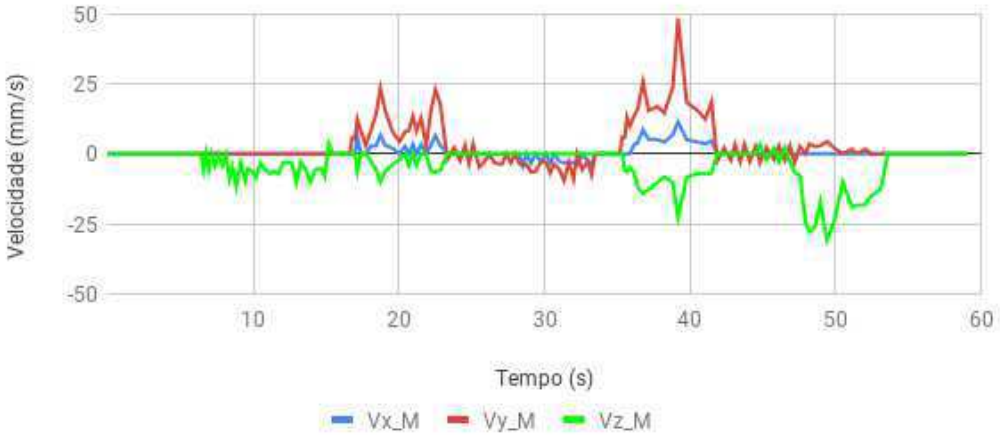
Foi analisado também o erro do seguimento de posição do escravo em relação ao mestre, mostrado na Figura 53. Este último gráfico mostra que o erro atinge valores mais elevados quando as velocidades do mestre e escravo são mais elevadas.

Figura 50 – Simulação 3: Posição do Escravo.



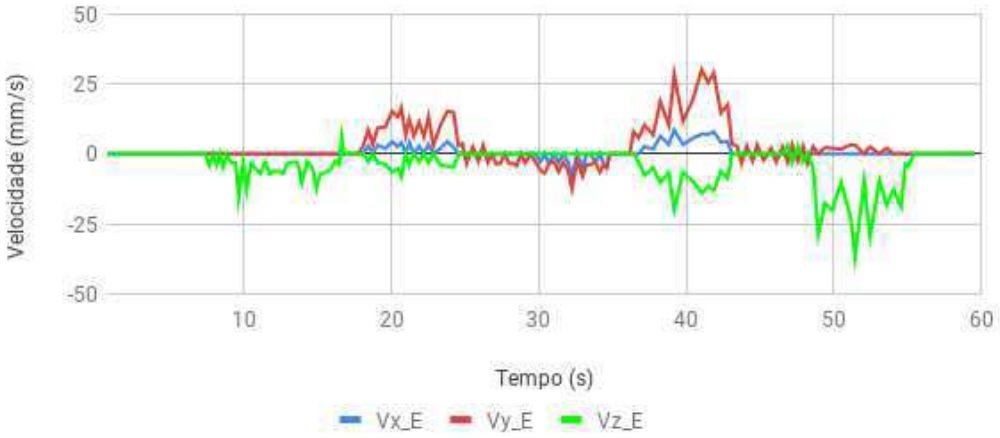

Fonte: do autor.

Figura 51 – Simulação 3: Velocidades do Mestre.



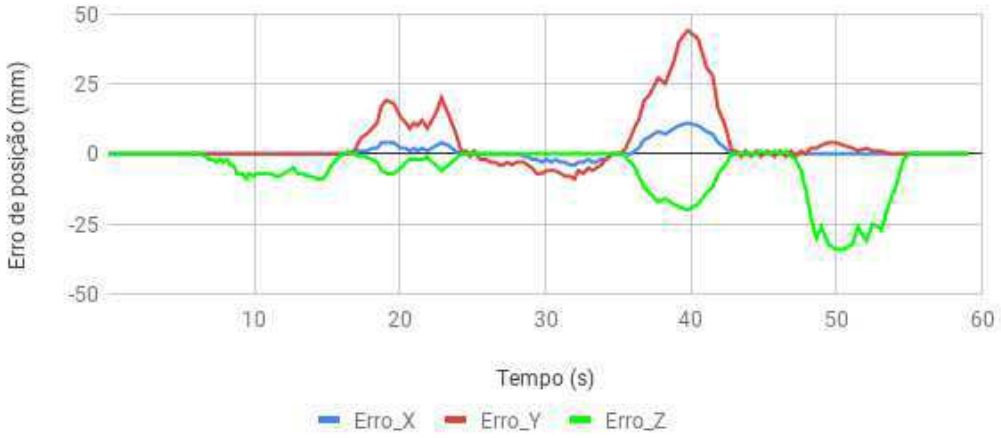

Fonte: do autor.

Figura 52 – Simulação 3: Velocidades do Escravo.




Fonte: do autor.

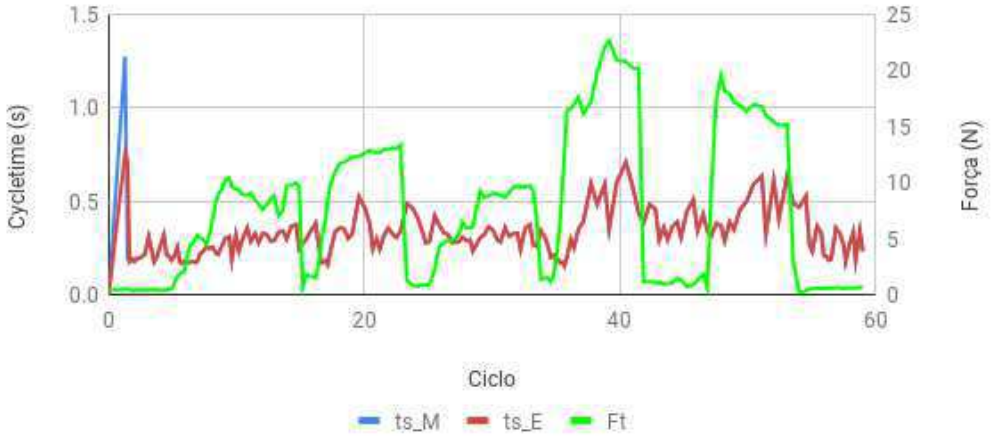
Figura 53 – Simulação 3: Erro de seguimento de posição.




Fonte: do autor.

A Figura 54 mostra também a relação entre as forças de estímulo e os *cycletimes* do mestre e escravo, para cada ciclo de operação. Nota-se que, com exceção dos picos de valores iniciais, provavelmente causados pela inicialização do sistema, os valores de atraso se mantiveram abaixo de 0,5 segundos, e oscilando pouco quando ocorriam maiores valores absolutos de força. É interessante notar também que os *cycletimes* do mestre e escravo são os mesmos em cada ciclo, o que demonstra que o escravo possui o mesmo desempenho do mestre.

Figura 54 – Simulação 3: Relação entre força e atraso.




Fonte: do autor.

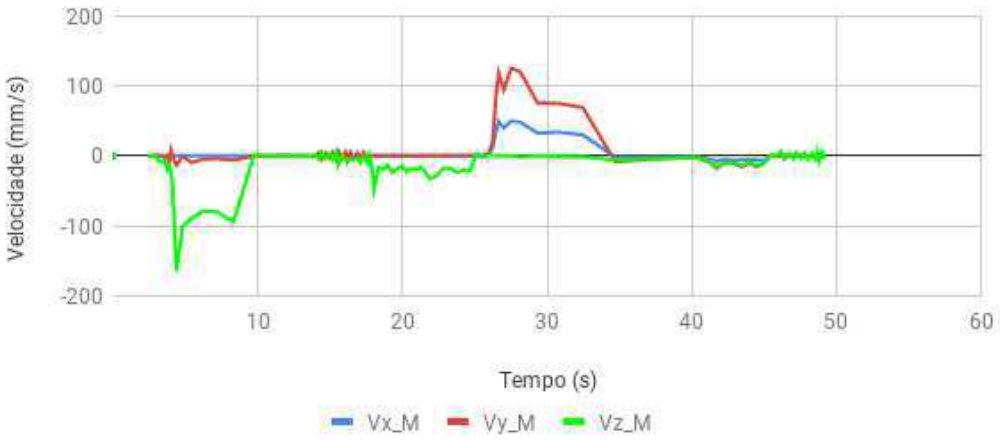
A Simulação 4 por sua vez foi feita utilizando um algoritmo que utiliza uma velocidade variável para o comando *MoveL*. Neste sistema, a velocidade calculada conforme a Equação 5.3, que utilizada para calcular o incremento de posição do mestre, também é utilizada para definir a velocidade de avanço do robô na linguagem RAPID. A velocidade, em mm/s é transformada em uma porcentagem de um valor base pré-definido de velocidade. Essa porcentagem é então enviada para o controlador do robô, junto com os dados de incremento de posição, sendo usada então como argumento para a função *MoveL*. Tomou-se o cuidado também em estabelecer limites para essas velocidades, pois isso

pode gerar problemas no *software* de controle dos manipuladores.

Os limites inferiores e superiores de velocidade utilizados nesse experimento foram de 50 e 500 mm/s, respectivamente, enquanto a velocidade base definida, tanto para o mestre quanto para o escravo, foi de 500 mm/s, ou seja, a porcentagem de movimento enviada varia sempre entre 10 e 100% do valor base. É importante notar que mesmo que nunca seja computada uma velocidade inferior a 50 mm/s, isso não significa que o manipulador estará sempre em movimento, pois além do argumento de velocidade, o *MoveL* também necessita de um argumento de distância percorrida, e se ela for zero, a velocidade pouco importa, pois não haverá deslocamento.

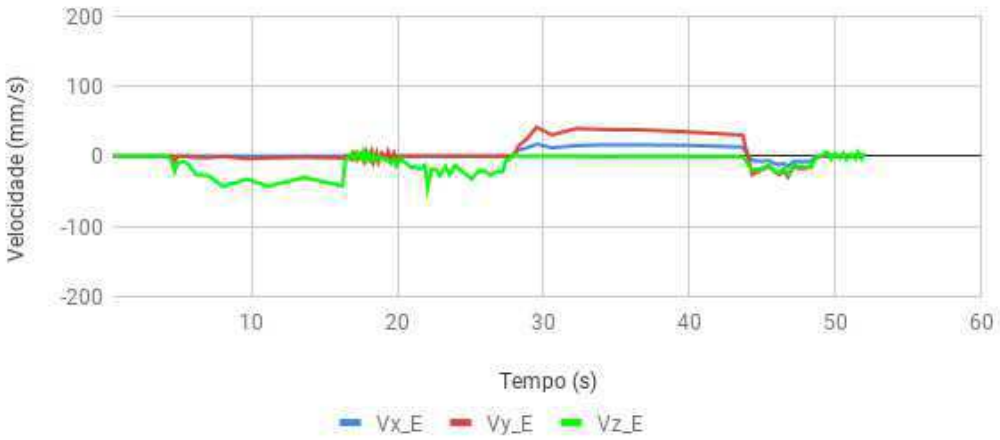
As Figuras 55 e 56 mostram as velocidades do mestre e escravo, respectivamente. É possível notar que há um atraso considerável, especialmente a partir de 30 segundos, quando o escravo demora para retornar às velocidades próximas de zero. Isso pode ser relacionado com o aumento da das forças aplicadas no sensor.

Figura 55 – Simulação 4: Velocidades do Mestre.




Fonte: do autor.

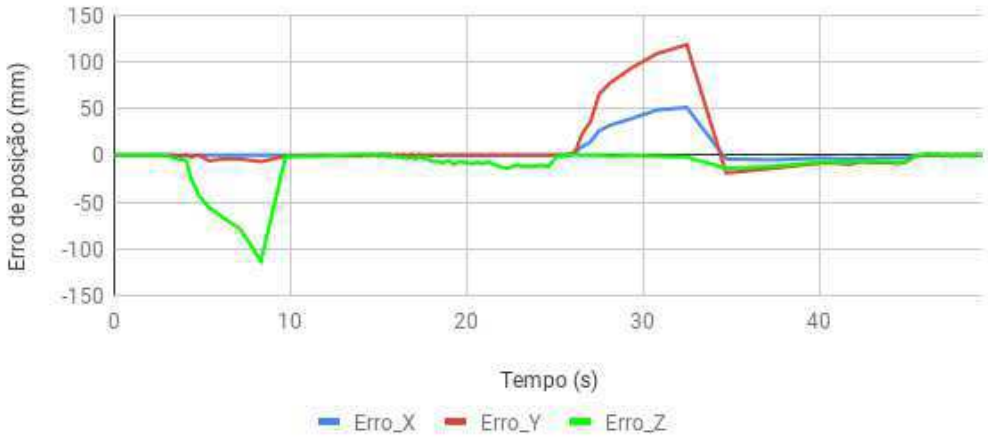
Figura 56 – Simulação 4: Velocidades do Escravo.




Fonte: do autor.

O erro no seguimento de posição pode ser visto na Figura 57. Assim como na Simulação 3, o erro aumenta conforme aumenta a velocidade do mestre.

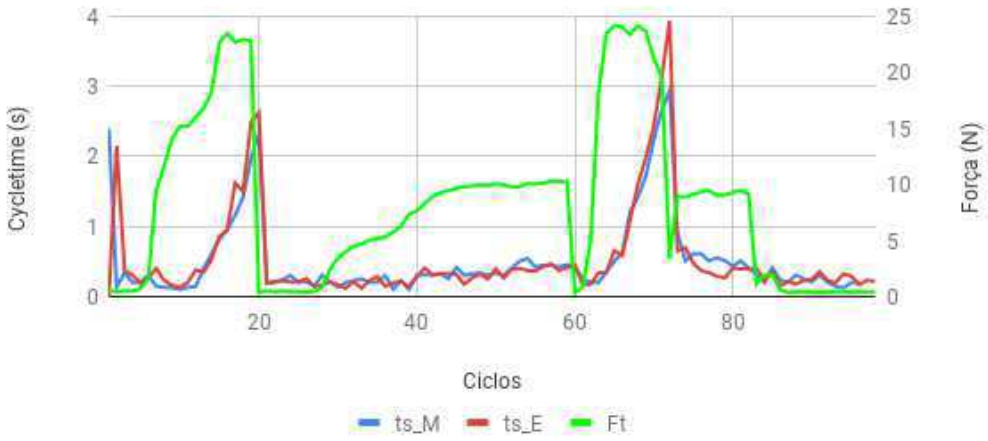
Figura 57 – Simulação 4: Erro de seguimento de posição.




Fonte: do autor.

Por fim a Figura 58 mostra a relação entre as forças de estímulo e os *cycletimes* do mestre e escravo. Mesmo que as curvas sejam bastante similares para o *cycletimes*, os valores para cada ciclo se somam para se obter o atraso total, nesse caso sendo muito maior do que nas simulações anteriores.

Figura 58 – Simulação 4: Relação entre força e atraso.




Fonte: do autor.

### 5.7.4 Teleoperação com realimentação

Na Simulação 5, foram estipuladas as restrições virtuais conforme visto no Capítulo 3.1. A geometria cilíndrica da zona de operação foi arbritrada com a altura de 400mm com raio de 150mm, posicionada com seu centro nas coordenadas x = 900 mm, y = 0 mm e z = 1000 mm, que coincide com a posição inicial dos manipuladores no sistema de coordenadas global.

Para a definição da impedância do ambiente virtual de operação, foi primeiramente feita a análise do sistema dinâmico que corresponderia a interação do mestre com o ambiente, no domínio

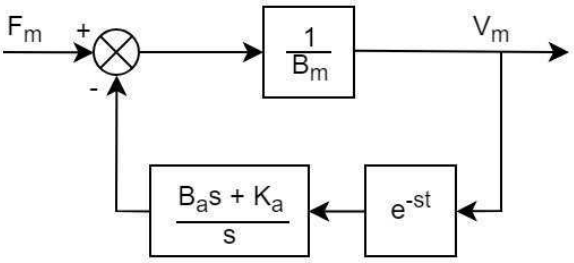
do tempo. A partir da Equação 3.10, é possível montar a equação do sistema equivalente, onde a velocidade $\dot{x}_a(t)$ pode ser considerada igual a $\dot{x}_m(t)$, uma vez que uma vez que o manipulador adentre as zonas restritivas, as duas variáveis são as mesmas. O sistema de uma entrada e uma saída pode ser mostrado na Equação 5.6, cuja representação no domínio da frequência é dada pela Equação 5.7.

$$(B_m + B_a)\dot{x}_m(t) + K_a x_m(t) = F_m(t) \tag{5.6}$$

$$(B_m + B_a)V_m(s) + K_a \frac{V_m(s)}{s} = F_m(s) \tag{5.7}$$

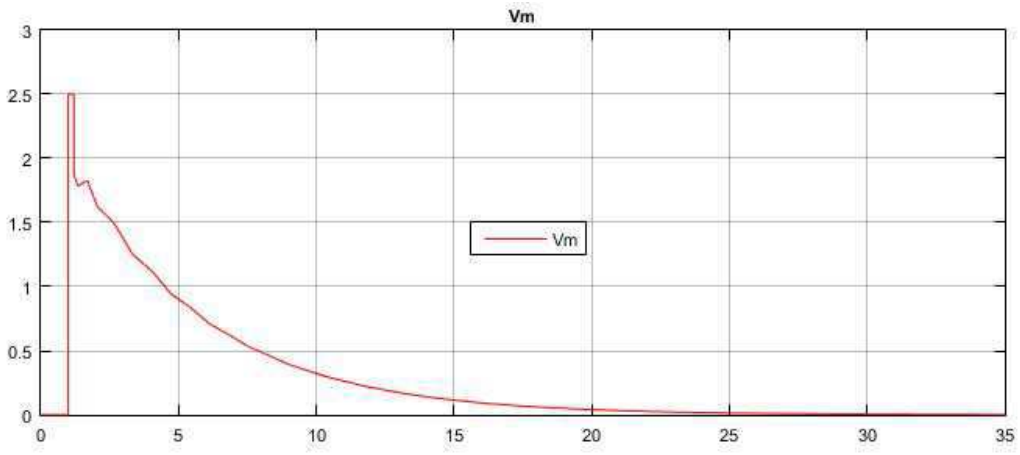
A partir dessas equações, é possível montar o diagrama de blocos do sistema, e considerando também o atraso na realimentação da força, o diagrama geral do sistema é mostrado na Figura 59. Trata-se de um sistema de primeira ordem com atraso no tempo. Uma vez que a admitância do mestre é composta apenas do amortecimento $B_m$, a mesma se comporta apenas como um ganho, não alterando oferecendo efeito dinâmico no sistema. Portanto, no momento em que o manipulador adentra na zona restritiva, os sistema começa a atuar com uma entrada degrau (no caso de uma força constante do sensor). O comportamento esperado do sistema para uma entrada degrau do sistema pode ser visto na Figura 60, onde foram utilzados os mesmos parâmetros da Simulação 5 e um tempo de atraso fixo de 0,2 segundos.

Figura 59 – Diagrama de blocos da realimentação de força virtual.




Fonte: do autor.

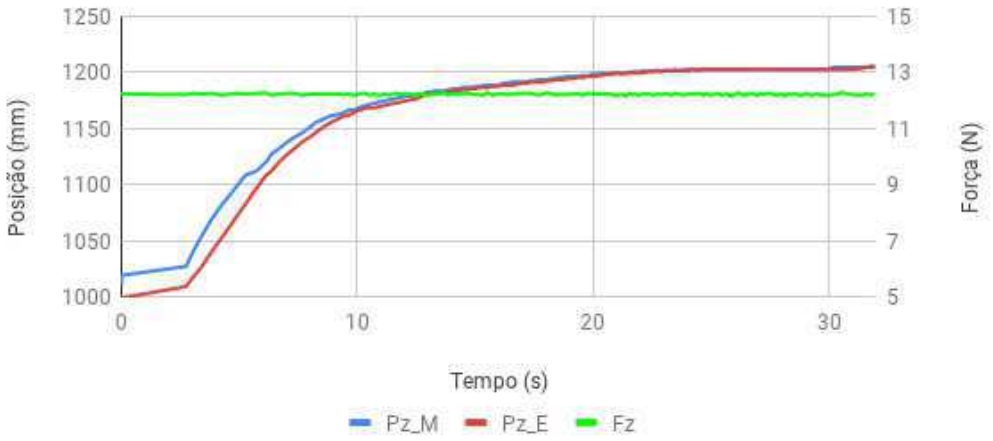
Figura 60 – Resposta da admitância à entrada degrau (estável).




Fonte: do autor.

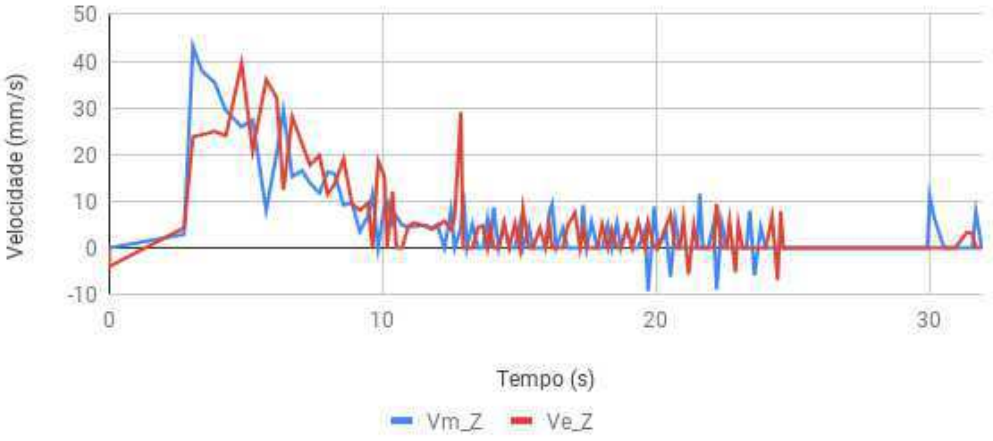
Os resultados da Simulação 5 podem ser vistos nas Figuras 61, onde é mostrada as curvas de posição, força, posição, velocidade e *cycletime* em função do tempo, quando o sensor é submetido a uma força constante de 12 N no eixo Z. O primeiro fato a se notar é a estabilizacão da saída em regime permanente, levando a velocidade a zero, demonstrando um comportamento quase idêntico com a resposta ao degrau mostrado na Figura 60.

Figura 61 – Simulação 5: Posições do mestre e escravo no eixo z.




Fonte: do autor.

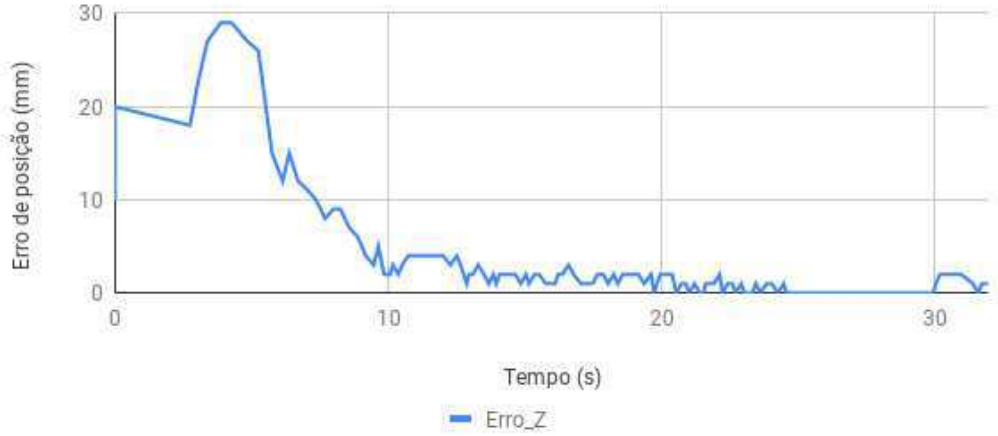
Figura 62 – Simulação 5: Velocidades do mestre e escravo no eixo z.




Fonte: do autor.

A Figura 63 mostra o erro de seguimento de posição do escravo em relação ao mestre no eixo z. Percebe-se que existe um elevado valor do erro no início da simulação. Isso se dá pelo fato de os dois manipuladores não estarem na mesma posição inicial no início da simulação. Porém após alguns segundos, o erro de posição é minimizado, conforme o sistema se estabiliza, seguindo o comportamento esperado.

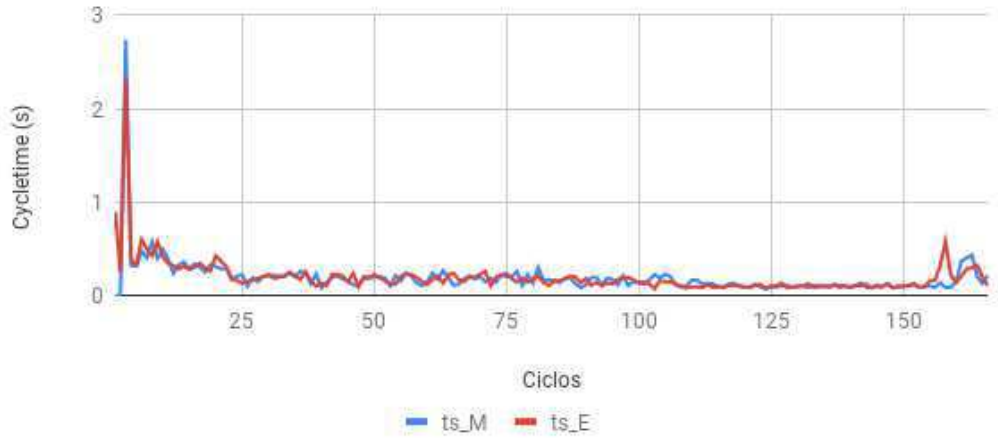
Figura 63 – Simulação 5: Erro no seguimento de posição no eixo z.




Fonte: do autor.

Esse comportamento mostra que, se o operador desejasse manter a velocidade constante do manipulador, mesmo com as restrições, ele deveria aumentar a força aplicada ao sensor. Isso equivale à sensação háptica do sistema. O manipulador não exerce força diretamente no operador, mas subtraindo a realimentação de força virtual da força medida no sensor, têm-se um equivalência matemática de uma força de reação. Já o *cycletime* mostrado na Figura 64 se mantém em um valor mais baixo e estável, se comparado com as simulações anteriores.

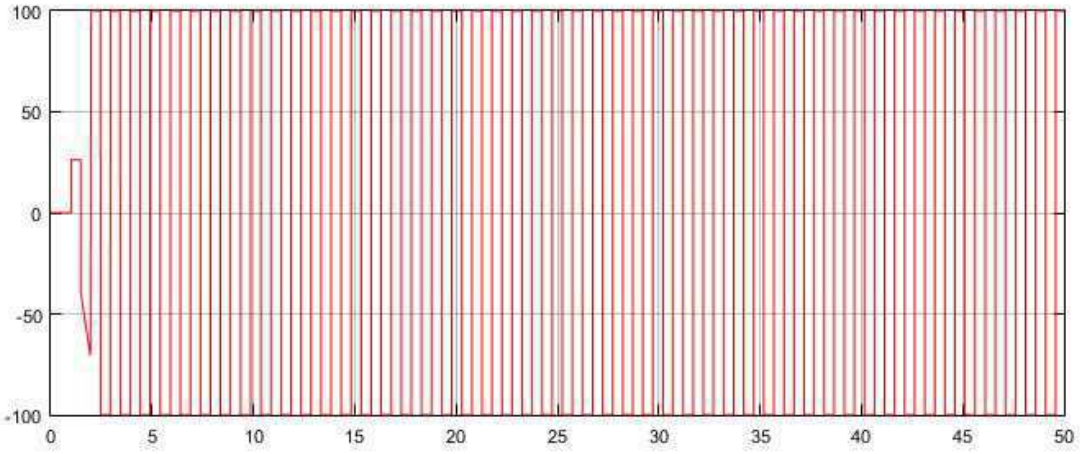
Figura 64 – Simulação 5: Atraso do mestre e escravo.




Fonte: do autor.

A Simulação 6 teve como objetivo analisar o sistema nos casos de instabilidade. O procedimento foi feito de forma similar à Simulação 5, porém com parâmetros diferentes para o ambiente. Para definir os parâmetros do ambiente para se obter um sistema instável, também foi gerada uma resposta degrau ao sistema com diferentes valores para os parâmetros de $B_a$ e $K_a$, seguindo o sistema da Figura 59, e então de forma empírica, foram utilizados os valores de 1 Ns/mm e 1 N/mm, respectivamente, e com um atraso de 0,48 segundos na realimentação. A resposta de velocidade a uma entrada degrau de 10,5 N é mostrada na Figura 65.

Figura 65 – Resposta da admitância à entrada degrau (instável).

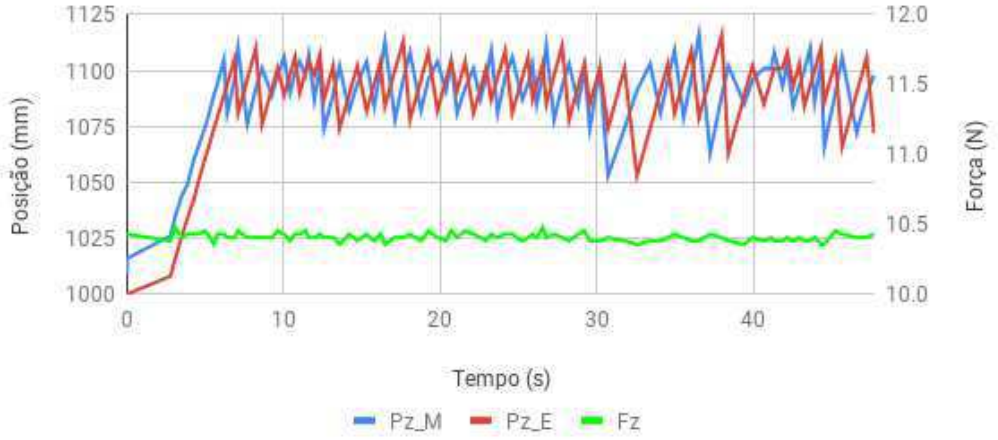



Fonte: do autor.

Rapidamente é visto que o sistema tem um comportamento instável em regime permanente. Nessa simulação preliminar no Matlab, foi adicionado um bloco de saturação, pois o sistema atinge valores altíssimos na saída, gerando uma curva que dificulta a comparação com um sistema real.

Para a Simulação 6 foi então aplicada uma forca constante de 10,5 N no sensor de força. As Figuras 66 e 67 mostram as posições e velocidades, respectivamente, do mestre e do escravo no eixo *z*.

Figura 66 – Simulação 6: Posições do mestre e escravo no eixo z.

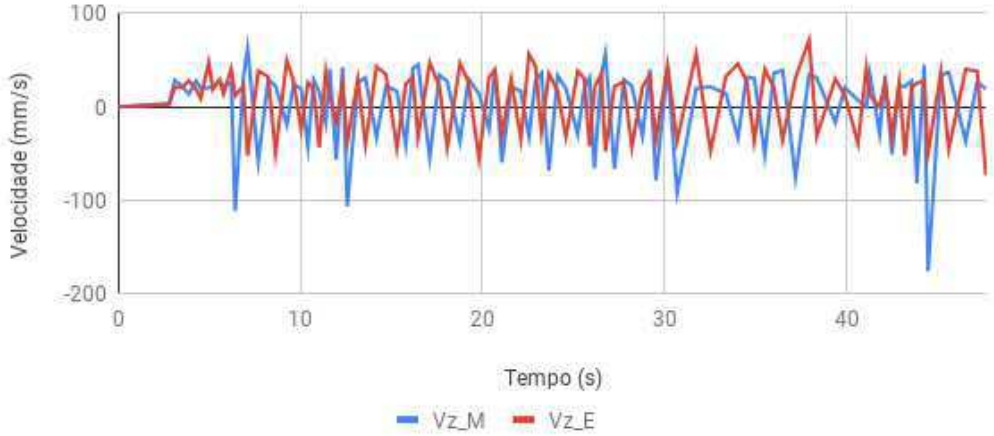



Fonte: do autor.

Observa-se claramente que o sistema se comporta de forma instável em regime permanente. É importante salientar também, que tanto o manipulador mestre quanto o escravo não atingiram valores maiores de velocidade, pois os controladores dos robôs possuem valores máximos de velocidade permitidos, e na própria programação em RAPID foi definida a faixa de operação da velocidade.

As Figuras 68 e 69 mostram o erro de seguimento de posição no eixo z e o *cycletimes*, respectivamente. O erro obviamente também se comporta de forma instável, porém os *cycletimes* mantêm um comportamento relativamente mais estável, oscilando em valores baixos, se comparados com outras

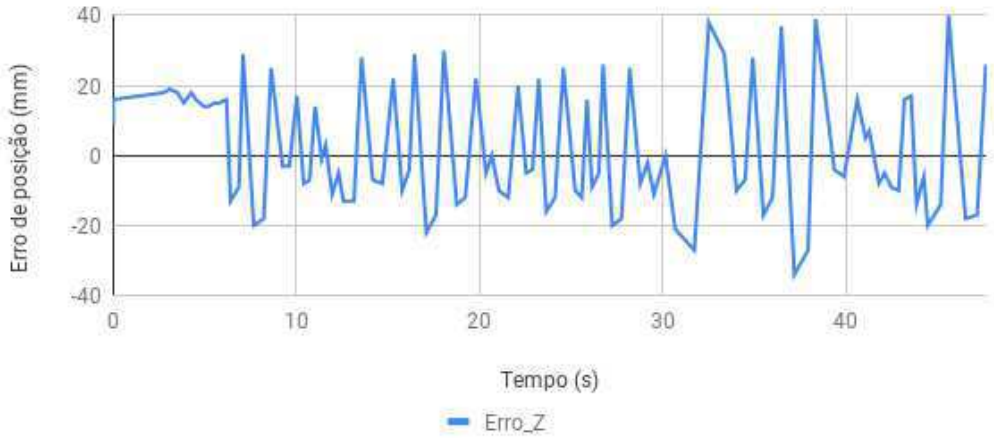
Figura 67 – Simulação 6: Velocidades do mestre e escravo no eixo z.




Fonte: do autor.

simulações. Porém isso se deve aos pequenos valores de força considerados no sistema de admitância do ambiente, que resultam da subtração das forças de reação do ambiente virtual dos valores de força medidos no sensor.

Figura 68 – Simulação 6: Erro no seguimento de posição no eixo z.



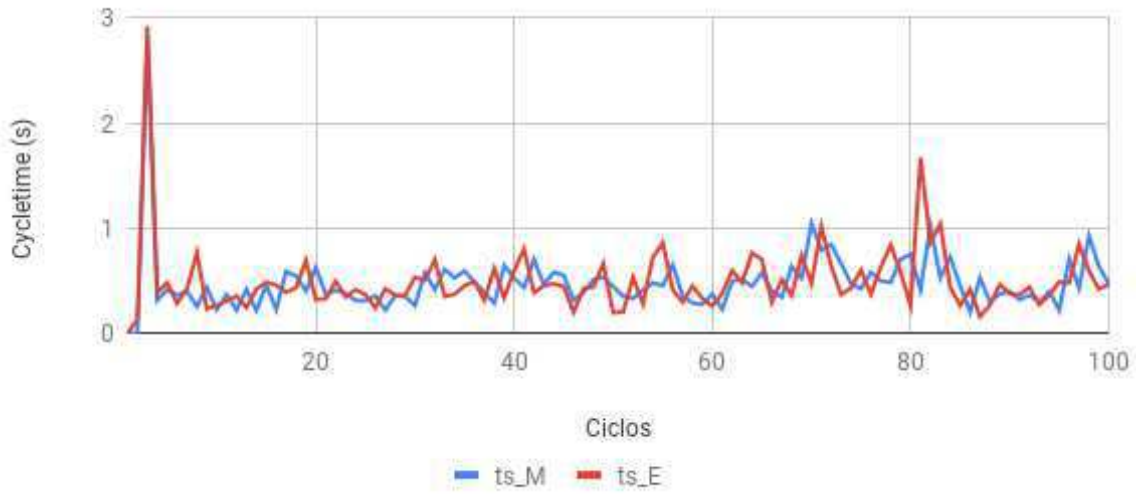

Fonte: do autor.

### 5.7.5 Análise do atraso

Os resultados das simulações referentes ao atraso e *cycletimes* do mestre e escravo foram analisadas em conjunto, para ser ter uma panorama geral do comportamento dessas variáveis. Trabalhou-se inicialmente com a hipótese de que o atraso dos ciclos de operação de teleoperação bilateral poderiam ser causados pelos seguintes fatores:

- Congestionamento da rede de comunicação de teleoperação;
- Processadores dos microcomputadores que simulam os controladores de robô não serem capazes de processar de forma rápida todos dados e operações necessárias;

Figura 69 – Simulação 6: Atraso do mestre e escravo.




Fonte: do autor.

- Comandos de movimentação mestre/escravo resultam em grandes distâncias de deslocamento durante um mesmo ciclo, aumentando o "tempo de espera"para se iniciar um novo ciclo.

Foi possível desconsiderar o fator do congestionamento da rede, uma vez que o sistema simulado possui uma rede local de comunicação, com poucas interconexões e um tráfego de mensagens pequeno. As mensagens trocadas entre controlador, mestre e escravo eram relativamente pequenas, possuindo apenas poucos valores numéricos de posição e velocidade.

O segundo fator é bastante plausível, uma vez que, como foi utilizado o *software* RobotStudio para simular a operação de robôs e controladores de robôs reais, todo o processamento dos mesmos ficou sob responsabilidade do microcomputador onde o *software* opera. Ou seja, não se pôde contar com um sistema fisicamente distribuído de controle e processamento. Porém é muito importante salientar que em aplicações reais, os controladores dos robôs são projetados para processar e comandar de forma eficiente os robôs manipuladores, ou seja, em sistemas de teleoperação reais, as causas de atraso pouco seriam causadas pelos controladores dedicados dos robôs, e sim no sistema de comunicação e central de processamento do sistema como um todo.

A terceira suposição foi a mais considerada para o sistema simulado, pois nesse caso é possível coletar as variáveis de tempo e tentar estimar a relação entre as forças de contato no sensor, que por sua vez são diretamente proporcionais a velocidade desejada do mestre e escravo, gerando então intervalos maiores de deslocamento. Em função disso, foi feita uma estimação de uma equação que relaciona-se as forças de estímulo do mestre com o seu tempo de ciclo.

Para essa estimação, foram consideradas apenas as simulações das Fases A e B, uma vez que elas possuem relação exclusiva entre a força medida no sensor com o deslocamento dos manipuladores, e ainda assim, foram utilizadas as Simulações 1, 2 e 3. A Simulação 4 foi desconsiderada, pelo fato de ter apresentado um comportamento pouco satisfatório, em função da velocidade base variável para cada ciclo.

No caso das simulações da Fase C, a realimentação de força faz com que o deslocamento do mestre e escravo não seja proporcional às forças medidas. A estimação foi feita utilizando uma

modelo de regressão linear, onde foram considerados quatro fatores que, em uma hipótese inicial, poderiam influenciar no tempo de atras do ciclo $t_s$, que são: uma constante, que representa um atraso mínimo do sistema; o atraso do ciclo anterior, que é utilizado no algoritmo do cálculo do incremento de posição do mestre; a velocidade desejada $v_d$, que é calculada de acordo com o amortecimento virtual do mestre $B_m$ e a força medida no sensor $F_s$; e a velocidade parametrizada $v_p$ de deslocamento do comando *MoveL* em RAPID. O modelo é mostrado nas equações abaixo.

$$t_s(k) = \lambda_0 + \lambda_1 t_s(k-1) + \lambda_2 v_d(k) + \lambda_3 v_p(k) \tag{5.8}$$

$$v_d(k) = |B_m^{-1} F_s(k)| = \sqrt{v_x^2(k) + v_y^2(k) + v_z^2(k)} \tag{5.9}$$

$$|F_s(k)| = \sqrt{F_{sx}^2(k) + F_{sy}^2(k) + F_{sz}^2(k)} \tag{5.10}$$

$$\lambda = \begin{bmatrix} \lambda_0 & \lambda_1 & \lambda_2 & \lambda_3 \end{bmatrix}^T \tag{5.11}$$

O objetivo da estimação é obter o vetor de parâmetros $\lambda$, que representam a influência de cada fator no *cycletime*. Neste modelo, as variáveis definidas são a matriz $B_m$ e a velocidade parametrizada $v_p$. O vetor de forças $F_s$ e os *cycletimes* são conhecidos em função da medição no sensor e medição do intervalos entre os ciclos, respectivamente. Por fim as variáveis indefinidas da equações são os parâmetros do vetor $\lambda$.

Foi utilizado o método dos Mínimos Quadrados Ordinários para se obter o vetor de parâmetros, conforme descrito no Apêndice F. O resultado da estimação é mostrado na Tabela 10.

Tabela 10 – Resultado da estimação de *cycletime*.

| Parâmetro | $\lambda_0$ | $\lambda_1$ | $\lambda_2$ | $\lambda_3$ | $s^2$ |
|---|---|---|---|---|---|
| Valor | 0,1615 | 0,4351 | 0,0027 | -0,007 | 0,0161 |

Fonte: do autor.

A primeira informação a ser considerada na estimação realizada é a variância obtida, que é um valor relativamente elevado, considerando o intervalo de valores da amostra. Isso indica que a estimação pode não ser confiável. Porém isso não elimina o fato de que existe influência dos fatores cogitados no valor do *cycletime* do sistema proposto.

Mesmo que a estimação não tenha sido confiável o bastante para estipular o modelo proposto como uma aproximação viável do *cycletime*, a mesma ainda permite que se relacione a influência dos fatores considerados. Nota-se inicialmente que fator que aparentemente mais influência no *cycletime* da amostra corrente, é o *cycletime* da amostra anterior, representado por $\lambda_1$. Já o intercepto $\lambda_0$ é a segunda maior influência, indicando que normalmente um *offset* do cycletime. Por fim, com uma influência menor, $\lambda_2$ e $\lambda_3$ representam a influência da velocidade desejada calculada para o ciclo e a velocidade base do deslocamento do robô, respectivamente.

O relativo alto valor de $\lambda_1$ mostra que o atraso do sistema opera de uma forma cumulativa, com uma certa memória, ou seja, um cycletime alto indica que o ciclo seguinte pode vir a ter um valor maior ainda. Isto pode ser um problema, pois em casos de picos de *cycletime*, entende-se que pode haver uma demora para essa variável retornar a valores mais baixos, mesmo que o sistema volte a operar em uma zona em que normalmente haveria menores atrasos.

Os valores mais baixos para $\lambda_2$ e $\lambda_3$ mostram que a questão da velocidade do manipulador pode não exercer tanta influência no atraso como o esperado. O valor de $\lambda_2$ é considerável, pois maiores valores da velocidade desejada calculada implicam em maiores incrementos de velocidade, que também dependem de $t_s(k-1)$. Já o valor negativo de $\lambda_3$ também é plausível, uma vez que maiores velocidades do manipulador implicam em um trajeto mais rápido ao percorrer o incremento de posição desejado, finalizando o ciclo de operação em um menor tempo.

Por fim, o intercepto $\lambda_0$ é discutível, uma vez que em algumas das simulações, se obtiveram valores menores do que 0,1615 segundos. Isso implicaria que esse seria o valor mínimo esperado para o valor do *cycletime*, e que poderia ser reduzido apenas pela influência da velocidade base do comando *MoveL*, mas que na prática não exerce influência se o controlador do robô possui algum comando de incremento de posição.

# 6 CONCLUSÃO

Ao final deste trabalho, foi possível estabelecer algumas conclusões em função da metodologia utilizada, do processo de desenvolvimento do sistema proposto e dos resultados obtidos nas simulações. Também foram considerados fatores externos ao trabalho para argumentar a favor dessas conclusões, como os equipamentos e tecnologias disponíveis fisicamente no ambiente da universidade, para realizar a pesquisa e aplicação: as limitações de *hardware* e *software* das ferramentas utilizadas, a disponibilidade de tempo para a realização do trabalho, etc.

Inicialmente, pode-se afirmar que ambos os sistemas se comportaram de forma satisfatória e esperada. Os resultados foram condizentes com as limitações das ferramentas utilizadas e as características previstas para a aplicação desta tecnologia. Obviamente foram identificadas possibilidades de melhoria ou correção do sistema, assim como pontos que não se mostraram suficientemente úteis para o sistema.

O segundo ponto a ser observado, foi a complexidade de se implementar um sistema de teleoperação que utilize equipamentos industriais, especialmente aqueles que não foram concebidos especificamente para operar em teleoepração ou interação homem-máquina. Uma vez que a tecnologia de teleoperação bilateral possui requisitos específicos, pelo fato de operar em contato com operadores humanos, se torna bastante desafiador implementar tal sistema utilizando equipamentos, neste caso robôs e sensores, que não foram projetados para essa função, e sim para tarefas gerais na indústria. Entretanto o fato de um equipamento não ter sido projetado especificamente para uma aplicação, não exclui a possibilidade de usar o mesmo em um novo tipo de aplicação, e isto é muito interessante do ponto de vista da pesquisa em tecnologia.

O que tornou desafiadora a implementação do sistema de teleoperação em robôs da ABB, foi o fato da linguagem de programação RAPID não possuir uma função de comando de velocidade linear do robôs no espaço operacional. Os comandos disponíveis apenas permitiam movimentos em intervalos definidos de espaço, com velocidade definida da trajetória "ponto-a-ponto"(WALDRON; SCHMIEDELER, 2008), porém na teleoperação, como a trajetória é gerada em tempo real pelo operador humano, se torna inadequado o movimento "ponto-a-ponto". A abordagem mais adequada seria um comando de movimento que a cada ciclo atualizasse a velocidade e a direção de deslocamento do manipulador. Isso aumentaria a precisão do movimento dos dispositivos mestre e escravo, uma vez que foi proposto uma reação de movimento em função de um sistema de admitância, cuja variável de saída é a velocidade, e não a posição.

Foi necessária então a adaptação do sistema para que se executasse o movimento entre pequenos incrementos de posição. Porém não houve muita precisão nesse aspecto, uma vez que a definição dos incrementos é proveniente da velocidade calculada no sistema de admitância, multiplicada pelo *cycletime* de operação do mestre. Como o *cycletime* se mostrou bastante variável, e a estimação do mesmo não é confiável, tornou-se impreciso o cálculo dos incrementos, especialmente quando a força de contato do operador foi mais elevada.

Isso leva à outra questão de grande importância. O sistema de comunicação utilizado não é determinista, o que dificulta as funções de controle de força e de movimento do mestre e escravo. Em função disso, uma topologia de comunicação determinística se faz necessária, pois permite melhorar a precisão, estabilidade e controlabilidade do sistema, uma vez que os atrasos entre os ciclos e seus

efeitos podem ser estimados.

Um comportamento interessante que pôde ser notado, foi que a sutileza e continuidade dos movimentos dos manipuladores na teleoperação foi inverso à velocidade de deslocamento. Quando foram aplicadas forças maiores ao sensor, resultando em maiores velocidades de deslocamento, o movimento dos manipuladores tornou-se abrupto entre os deslocamentos dos incrementos de posição calculados. Quando foram aplicadas forças menores, o movimento dos manipuladores se comportou de forma mais suave e precisa. Aliado a isso, os *cycletimes* se mostraram menores para forças menores, o que gerou uma melhor performance do sistema.

Em função disso, foi possível concluir que o sistema proposto é mais adequado para manipulações de pequenas acelerações, velocidades e deslocamentos. Isso é bastante positivo, uma vez que muitas das aplicações de teloperação bilateral incluem tarefas de maior delicadeza e precisão. Também entende-se que qualquer tarefa de robótica que inclua movimentos em grande magnitude de velocidade e aceleracão, possui maiores desafios e problemas no que diz respeito ao controle de força e movimento, pois nesses casos a influência do comportamento dinâmico do manipulador é muito maior, aumentando a complexidadade matemática do sistema.

Obviamente, essas constatações são relevantes enquanto os robôs e controladores são da fabricante ABB. É possível que equipamentos de outros fabricantes possuam outras caracterísitcas de *hardware* e *software* que facilitem a aplicação de sistemas de teleoperação bilateral, ou que talvez possuam ainda mais limitações que a tecnlogia desenvolvida pela ABB.

Em relação à realimentação virtual de força, ela se comportou de forma satisfatória, mostrando-se capaz de fornecer a sensação háptica para o operador humano, conforme o mestre ou escravo interage com o ambiente dinâmico virtual. É importante realçar que a utilização de um ambiente virtual é útil como uma abordagem generalista de interação do sistema de teleoperação com uma meio qualquer de operação. Esse ambiente virtual poderia representar um ambiente real, um ambiente híbrido, com restrições virtuais e reais, ou qualquer tipo de cenário onde os manipuladores operam.

A forma como foi implementada a realimentação de força, mostra que para tarefas cujas interações mecânicas possuem certa simplicidade, um algoritmo simples pode ser implementado para produzir uma sensação háptica, considerando as limitações de comunicação e acionamento dos robôs manipuladores. Com a utilização de outros instrumentos capazes de obter informação sobre a interação dos robôs com o ambiente de operação, a performance da percepção háptica pode ser melhorada.

Por fim, em relação às causas de atrasos e tempo de operação do sistema de teleoperação, chegou-se a mesma conclusão dos demais autores. Com a análise do *cycletime* e a tentativa de estimação de um modelo do mesmo, viu-se que o os atrasos e tempos de operação pouco dependiam das ações de comando e controle de força e movimento. Presume-se portanto que, nesse sistema, o atraso é mais influenciado pela performance da comunicação e do processamento computacional do sistema.

## 6.1 CONTRIBUIÇÃO

De modo geral, destaca-se que a contribuição deste trabalho para a área de robótica se deu na questão do levantamento de características, requisitos e desafios da utilização de robôs e tecnologias industriais em aplicações de teleoperação bilateral. A orginalidade do trabalho se deu no fato de terem

sido usadas ferramentas em um sistema de teleoperação, que não foram projetadas para este tipo de aplicação, com o intuito de mostrar para o meio técnico e acadêmico que podem ser utilizados equipamentos e tecnologias comuns do meio industrial para conceber um sistema de teleoperação, pois essas ferramentas normalmente são, em relação a tecnologias dedicadas para sistemas de teleoperação, mais baratas, mais acessíveis e possuem uma gama maior de opções no que se refere a fabricantes e fornecedores. Isso mostra a significância dessa pesquisa em relação ao ganho econômico para as aplicações aqui discutidas para a indústria e outros setores não industriais.

Foi demonstrado, portanto, que é possível conceber um sistema de teleoperação bilateral utilizando robôs manipuladores industriais tanto nas tarefa de escravo como na de mestre. Como este trabalho abordou um tema pouco explorado no meio acadêmico, no quesito de implementação e aplicação prática da teleoperação bilateral de robôs industriais, entende-se que esta pesquisa assume um papel mais provocativo e fomentador de outras pesquisas e desenvolvimento de sistemas de teleoperação bilateral e de cooperação homem-máquina que utilizem robôs fabricados com finalidades industriais.

## 6.2 TRABALHOS FUTUROS

Com base nos resultados obtidos e desafios identificados neste trabalho, foram levantados as seguintes questões, temas e sugestões a serem desenvolvidas em trabalhos futuros relacionados com esta pesquisa:

- Utilização de robôs e controladores de robôs industriais de variados fabricantes e características construtivas, e que utilizem diferentes linguagens e métodos de programação, em sistemas de teleoperação bilateral;
- Substituir a realimentação virtual de força por um sensor de força acoplado ao efetuador final do escravo, a fim de se obter valores reais de força de interação com o ambiente operacional do manipulador;
- Mesclar a realimentação real com a realimentação virtual de força, a fim de diminuir os atrasos da realimentação, utilizando um modelo intermediário que estime as reações oriundas da interação do escravo com o ambiente de operação;
- Aplicar métodos de compensação ou estimação de atraso na comunicação e *cycletime* do sistema;
- Monitorar o desempenho do processamento computacional do sistema, a fim de aprimorar o *hardware* e o desempenho geral do sistema.

# REFERÊNCIAS

ABB. *Introduction to RAPID*. Västerås, 2007. Disponível em: <http://www.oamk.fi/~eeroko/Opetus/Tuotantoautomaatio/Robotiikka/Introduction_to_RAPID_3HAC029364-001_rev-_en.pdf>. Acesso em: 30 nov. 2018.

ABB. *Operating manual: RobotStudio*. Västerås, 2010. Disponível em: <https://library.e.abb.com/public/244a8a5c10ef8875c1257b4b0052193c/3HAC032104-001_revD_en.pdf>. Acesso em: 30 nov. 2018.

ABB. *RAPID Instructions, Functions and Data types*. Västerås, 2010. Disponível em: <https://library.e.abb.com>. Acesso em: 30 nov. 2018.

ABB. *IRB 1600 Data sheet*. Västerås, 2018. Disponível em: <https://new.abb.com/products/robotics/industrial-robots/irb-1600/irb-1600-data>. Acesso em: 29 nov. 2018.

ABB. *IRC5 Industrial Robot Controller Data Sheet*. Västerås, 2018. Disponível em: <https://new.abb.com/products/robotics/controllers/irc5>. Acesso em: 30 nov. 2018.

AMARAL, S. do; PIERI, E. R. de; GUENTHER, R. Controle a estrutura variável de robôs manipuladores interagindo com ambientes cinemáticos. *Revista Controle & Automação*, v. 11, p. 117–127, 2000.

ANDERSON, R. J.; SPONG, M. W. Bilateral control of teleoperators with time delay. *IEEE Transactions on Automatic control*, IEEE, v. 34, n. 5, p. 494–501, 1989.

BICCHI, A.; PESHKIN, M. A.; COLGATE, J. E. Safety for physical human–robot interaction. In: *Springer handbook of robotics*. Heidelberg: Springer-Verlag Berlin, 2008. p. 1335–1348.

BONILLA, A. A. C. *Cinemática diferencial de manipuladores empregando cadeias virtuais*. 2004. Dissertação de Mestrado. Universidade Federal de Santa Catarina. Florianópolis, SC.

CHEN, Z. et al. An improved wave-variable based four-channel control design in bilateral teleoperation system for time-delay compensation. *IEEE Access*, IEEE, v. 6, p. 12848–12857, 2018.

CHIAVERINI, S.; ORIOLO, G.; WALKER, I. D. Kinematically redundant manipulators. In: *Springer handbook of robotics*. Heidelberg: Springer-Verlag Berlin, 2008. p. 245–268.

COLONNESE, N.; OKAMURA, A. M. Analysis of effective impedance transmitted to the operator in position-exchange bilateral teleoperation. In: IEEE. *World Haptics Conference (WHC), 2017 IEEE*. 2017. p. 328–333.

COMEDI. *The Control and Measurement Device Interface handbook for Comedilib 0.10.0*. 2012. Disponível em: <http://www.linux-usb-daq.co.uk/comedi/pdf/comedilib.pdf>. Acesso em: 25 nov. 2018.

CRAIG, J. J.; RAIBERT, M. H. A systematic method of hybrid position/force control of a manipulator. In: IEEE. *Computer Software and Applications Conference, 1979. Proceedings. COMPSAC 79. The IEEE Computer Society's Third International*. 1979. p. 446–451.

DALVAND, M. M.; NAHAVANDI, S. Improvements in teleoperation of industrial robots without low-level access. In: IEEE. *Systems, Man and Cybernetics (SMC), 2014 IEEE International Conference on*. 2014. p. 2170–2175.

DAVIDSON, J. K.; HUNT, K. H.; HUNT, K. H. *Robots and screw theory: applications of kinematics and statics to robotics*. : Oxford University Press on Demand, 2004.

FEATHERSTONE, R.; ORIN, D. E. Dynamics. In: *Springer Handbook of Robotics*. Heidelberg: Springer-Verlag Berlin, 2008. p. 35–65.

FOLETTO, T. d. C. *PROPOSTA DE ESTIMADOR NÃO LINEAR INTERMITENTE PARA SISTEMAS DE CONTROLE VIA REDE SEM FIO*. 2013. Dissertação de Mestrado. Universidade Federal de Santa Catarina. Florianópolis, SC.

GOLIN, J. F. *Implementação de controle de força e compensação de atrito em um robô industrial*. 2002. Universidade Federal de Santa Catarina. Florianópolis, SC.

HÄGELE, M.; NILSSON, K.; PIRES, J. N. Industrial robotics. In: *Springer handbook of robotics*. Heidelberg: Springer-Verlag Berlin, 2008. p. 963–986.

HAMILTON, W. R. Xi. on quaternions; or on a new system of imaginaries in algebra. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, Taylor & Francis, v. 33, n. 219, p. 58–60, 1848.

HANNAFORD, B.; OKAMURA, A. M. Haptics. In: *Springer Handbook of Robotics*. Heidelberg: Springer-Verlag Berlin, 2016. p. 1063–1084.

HECK, D. et al. Direct force-reflecting two-layer approach for passive bilateral teleoperation with time delays. *IEEE Transactions on Robotics*, Institute of Electrical and Electronics Engineers (IEEE), v. 34, n. 1, p. 194–206, 2018.

HOGAN, N. Impedance control: An approach to manipulation. In: IEEE. *American Control Conference, 1984*. 1984. p. 304–313.

HOKAYEM, P. F.; SPONG, M. W. Bilateral teleoperation: An historical survey. *Automatica*, Elsevier, v. 42, n. 12, p. 2035–2057, 2006.

HU, L. et al. Trajectory tracking compensation for teleoperation with transmission delays. *Robotica*, Cambridge University Press, v. 29, n. 6, p. 863–871, 2011.

HUANG, S. et al. Stability analysis of bilateral teleoperation system based on data sampling with time delay. In: IEEE. *Intelligent Control and Information Processing (ICICIP), 2017 Eighth International Conference on*. 2017. p. 287–292.

IFR. *World Robotics 2018*. 2018.

JR3 Inc. *Multi-axis force-torque sensor technical specifications*. Woodland, 2003. Disponível em: <http://www.jr3.com/uploads/4/5/7/0/45700429/___spec_sheet_100m40a3_si_200n400n.pdf>. Acesso em: 12 nov. 2018.

KAO, I.; LYNCH, K.; BURDICK, J. W. Contact modeling and manipulation. Springer, p. 647–669, 2008.

KUROSE, J. F.; ROSS, K. W.; ZUCCHI, W. L. *Redes de Computadores ea Internet: uma abordagem top-down*. : Pearson Addison Wesley, 2007.

LAHR, G. J. et al. Understanding the implementation of impedance control in industrial robots. In: IEEE. *2016 XIII Latin-American Robotics Symposium and IV Brazilian Robotics Symposium (LARS/SBR)*. 2016. p. 269–274.

LIMA, A. T. et al. Teleoperation of an abb irb 120 robotic manipulator and barretthand bh8-282 using a geomagic touch x haptic device and ros. In: IEEE. *2018 Latin American Robotic Symposium, 2018 Brazilian Symposium on Robotics (SBR) and 2018 Workshop on Robotics in Education (WRE)*. 2018. p. 188–193.

LIU, M. et al. Bilateral control of teleoperation manipulator based on virtual force aware guidance. In: IEEE. *Cybernetics and Intelligent Systems (CIS) and IEEE Conference on Robotics, Automation and Mechatronics (RAM), 2017 IEEE International Conference on*. 2017. p. 231–236.

MCLAUGHLIN, M. L.; SUKHATME, G.; HESPANHA, J. *Touch in virtual environments: haptics and the design of interactive systems*. : Prentice Hall PTR, 2001.

MIYAZAKI, F. et al. A new control methodology toward advanced teleoperation of master-slave robot systems. In: IEEE. *Robotics and Automation. Proceedings. 1986 IEEE International Conference on*. 1986. v. 3, p. 997–1002.

NATH, N.; TATLICIOGLU, E.; DAWSON, D. M. Teleoperation with kinematically redundant robot manipulators with sub-task objectives. *Robotica*, Cambridge University Press, v. 27, n. 7, p. 1027–1038, 2009.

NIEMEYER, G.; PREUSCHE, C.; HIRZINGER, G. Telerobotics. In: *Springer handbook of robotics*. Heidelberg: Springer-Verlag Berlin, 2008. p. 741–757.

NIEMEYER, G.; SLOTINE, J.-J. Stable adaptive teleoperation. *IEEE Journal of oceanic engineering*, IEEE, v. 16, n. 1, p. 152–162, 1991.

NIEMEYER, G.; SLOTINE, J.-J. Transient shaping in force-reflecting teleoperation. In: IEEE. *Advanced Robotics, 1991.'Robots in Unstructured Environments', 91 ICAR., Fifth International Conference on*. 1991. p. 261–266.

PECLY, L. S.; SOUZA, M. L.; HASHTRUDI-ZAAD, K. Model-reference model-mediated control for time-delayed teleoperation systems. In: IEEE. *Haptics Symposium (HAPTICS), 2018 IEEE*. 2018. p. 72–77.

PEIPER, D. L. *The kinematics of manipulators under computer control*. Tese (Doutorado) — Stanford University, 10 1968.

PERERA, G. A.; ABEYKOON, A. H. S. Review on bilateral teleoperation with force, position, power and impedance scaling. In: IEEE. *Information and Automation for Sustainability (ICIAfS), 2014 7th International Conference on*. 2014. p. 1–7.

PHAM, H.-L. et al. Position and orientation control of robot manipulators using dual quaternion feedback. In: IEEE. *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*. 2010. p. 658–663.

RAHIMIFARD, S.; TALEBI, H.; MOHAMMADI, A. D. Impedance control of non-passive bilateral teleoperation systems with uncertain dynamics. In: IEEE. *2016 24th Iranian Conference on Electrical Engineering (ICEE)*. 2016. p. 1931–1936.

REBELO, J.; SCHIELE, A. Performance analysis of time-delay bilateral teleoperation using impedance-controlled slaves. In: IEEE. *Advanced Robotics (ICAR), 2015 International Conference on*. 2015. p. 28–33.

SANSANAYUTH, T.; NILKHAMHANG, I.; TUNGPIMOLRAT, K. Teleoperation with inverse dynamics control for phantom omni haptic device. In: IEEE. *SICE Annual Conference (SICE), 2012 Proceedings of*. 2012. p. 2121–2126.

SCHEINMAN, V.; MCCARTHY, J. M. Mechanisms and actuation. In: *Springer Handbook of Robotics*. Heidelberg: Springer-Verlag Berlin, 2008. p. 67–86.

SICILIANO, B.; KHATIB, O. *Springer handbook of robotics*. Heidelberg: Springer-Verlag Berlin, 2008.

SICILIANO, B. et al. *Robotics: modelling, planning and control*. : Springer Science & Business Media, 2010.

TASHIRO, T. et al. Time delay compensation for force controller in bilateral teleoperation system under time delay. In: IEEE. *Advanced Motion Control (AMC), 2018 IEEE 15th International Workshop on*. 2018. p. 649–654.

TIAN, D.; GAO, H.; ZHANG, L. Wireless ethernet haptic transmission based on a switching three-channel bilateral control. In: IEEE. *Industrial Electronics Society, IECON 2017-43rd Annual Conference of the IEEE*. 2017. p. 7251–7256.

VILLANI, L.; SCHUTTER, J. D. Force control. In: *Springer handbook of robotics*. Heidelberg: Springer-Verlag Berlin, 2008. p. 161–185.

WALDRON, K. J.; SCHMIEDELER, J. Kinematics. In: *Springer Handbook of Robotics*. Heidelberg: Springer-Verlag Berlin, 2008. p. 9–33.

WHITNEY, D. E. The mathematics of coordinated control of prosthetic arms and manipulators. *Journal of Dynamic Systems, Measurement, and Control*, American Society of Mechanical Engineers, v. 94, n. 4, p. 303–309, 1972.

XU, K. et al. Design of a haptic master device for teleoperation applications. In: IEEE. *2017 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. 2017. p. 1436–1441.

YUAN, Y.; WANG, Y.; GUO, L. Force reflecting control for bilateral teleoperation system under time-varying delays. *IEEE Transactions on Industrial Informatics*, IEEE, 2018.

ZHAO, J. et al. Virtual force feedback in teleoperation for enhanced manipulator performance. In: IEEE. *Real-time Computing and Robotics (RCAR), 2017 IEEE International Conference on*. 2017. p. 511–516.

# APÊNDICES

# A CÓDIGO PARA LEITURA DO SENSOR DE FORÇA JR3 EM LUNGUAGEM C++

```
#include <stdio.h>
#include <string.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <stdlib.h>
#include <comedilib.h>
#include <time.h>
#include <ctype.h>
#include <math.h>

int main(int argc,char *argv[])
{
  if(argc < 3)
  {
    printf("Inclua no comando o IP e a porta\n");
    return  1;
  }

  int sock;
  struct sockaddr_in server;
  char resposta[10];
  char ip[strlen(argv[1]+1)];
  char *mensagem;
  comedi_t *it;
  lsampl_t data[5];
  comedi_range * range_info;
  lsampl_t maxdata;
  int retval;
  int i = 0;
  char Fx[10];
  char Fy[10];
  char Fz[10];
  char Mx[10];
  char My[10];
  char Mz[10];

  sock = socket(AF_INET,SOCK_STREAM,0);
  if(sock ==  1)
  {
    printf("Erro na criacao do socket");
  }
  memcpy(ip,argv[1],strlen(argv[1])+1);
  server.sin_addr.s_addr = inet_addr(ip);
  server.sin_family = AF_INET;
  server.sin_port = htons(atoi(argv[2]));
  if(connect(sock,(struct sockaddr*)&server,sizeof(server))<0)
  {
```

```
49    perror("Conex o falhou");
50    return 1;
51  }
52  it = comedi_open("/dev/comedi0");
53  if(it == NULL)
54  {
55    comedi_perror("comedi_open");
56    return  1;
57  }
58  while(1)
59  {
60    printf("\e[1:1H\e[2J");
61    for(i = 0;i<6;i++)
62    {
63            retval = comedi_data_read(it,0,i,0,AREF_GROUND,& data[i]);
64            if(retval < 0)
65      {
66                  comedi_perror("comedi_data_read");
67                  return 1;
68            }
69      comedi_set_global_oor_behavior(COMEDI_OOR_NAN);
70      range_info = comedi_get_range(it,0,i,0);
71      maxdata = comedi_get_maxdata(it,0,i);
72      printf("Canal %d: ",i);
73      printf("%lu\n",(unsigned long)data[i]);
74    }
75    snprintf(Fx,10,"%lu",(unsigned long)data[0]);
76    snprintf(Fy,10,"%lu",(unsigned long)data[1]);
77    snprintf(Fz,10,"%lu",(unsigned long)data[2]);
78    snprintf(Mx,10,"%lu",(unsigned long)data[3]);
79    snprintf(My,10,"%lu",(unsigned long)data[4]);
80    snprintf(Mz,10,"%lu",(unsigned long)data[5]);
81    if((mensagem = calloc(1,70))!= NULL)
82    {
83      mensagem[0]='\0';
84      strcat(mensagem,Fx);
85      strcat(mensagem,";");
86      strcat(mensagem,Fy);
87      strcat(mensagem,";");
88      strcat(mensagem,Fz);
89      strcat(mensagem,";");
90      strcat(mensagem,Mx);
91      strcat(mensagem,";");
92      strcat(mensagem,My);
93      strcat(mensagem,";");
94      strcat(mensagem,Mz);
95    }
96    else
97    {
98      perror("Alocacao falhou!\n");
```

```
99          return  1;
100       }
101       if(send(sock,mensagem,strlen(mensagem),0)<0)
102       {
103         puts("Falha no envio");
104         return 1;
105       }
106       if(recv(sock,resposta,80,0)<0)
107       {
108         puts("Sem resposta");
109         close(sock);
110       }
111       sleep(1);
112       free(mensagem);
113         }
114         close(sock);
115         return 0;
116 }
```

# B MATRIZES JACOBIANAS DE POSIÇÃO DOS ROBÔS DE TRÊS GRAUS DE LIBERDADE

## B.1 DISPOSITIVO RRR

$$q_m = \begin{bmatrix} \theta_{1m} \\ \theta_{2m} \\ \theta_{3m} \end{bmatrix} \tag{B.1}$$

$$p_m = \begin{bmatrix} x_m \\ y_m \\ z_m \end{bmatrix} = \begin{bmatrix} c_{\theta_{1m}}[30+200(c_{\theta_{2m}}+c_{\theta_{2m}}c_{\theta_{3m}}-s_{\theta_{2m}}s_{\theta_{3m}})] \\ s_{\theta_{1m}}[30+200(c_{\theta_{2m}}+c_{\theta_{2m}}c_{\theta_{3m}}-s_{\theta_{2m}}s_{\theta_{3m}})] \\ -200(s_{\theta_{2m}}+c_{\theta_{2m}}s_{\theta_{3m}}-s_{\theta_{2m}}c_{\theta_{3m}}) \end{bmatrix} \tag{B.2}$$

$$J_m = \frac{\partial p_m}{\partial q_m} = \begin{bmatrix} \frac{\partial x_m}{\partial q_m} & \frac{\partial y_m}{\partial q_m} & \frac{\partial z_m}{\partial q_m} \end{bmatrix} \tag{B.3}$$

$$\frac{\partial x_m}{\partial q_m} = \begin{bmatrix} -s_{\theta_{1m}}[30+200(c_{\theta_{2m}}+c_{\theta_{2m}}c_{\theta_{3m}}-s_{\theta_{2m}}s_{\theta_{3m}})] \\ -200c_{\theta_{1m}}(s_{\theta_{2m}}+s_{\theta_{2m}}c_{\theta_{3m}}+c_{\theta_{2m}}s_{\theta_{3m}})] \\ -200c_{\theta_{1m}}(c_{\theta_{2m}}s_{\theta_{3m}}+s_{\theta_{2m}}c_{\theta_{3m}})] \end{bmatrix} \tag{B.4}$$

$$\frac{\partial y_m}{\partial q_m} = \begin{bmatrix} c_{\theta_{1m}}[30+200(c_{\theta_{2m}}+c_{\theta_{2m}}c_{\theta_{3m}}-s_{\theta_{2m}}s_{\theta_{3m}})] \\ -200s_{\theta_{1m}}(s_{\theta_{2m}}+s_{\theta_{2m}}c_{\theta_{3m}}+c_{\theta_{2m}}s_{\theta_{3m}}) \\ -200s_{\theta_{1m}}(c_{\theta_{2m}}s_{\theta_{3m}}+s_{\theta_{2m}}c_{\theta_{3m}}) \end{bmatrix} \tag{B.5}$$

$$\frac{\partial z_m}{\partial q_m} = \begin{bmatrix} 0 \\ -200(c_{\theta_{2m}}-s_{\theta_{2m}}s_{\theta_{3m}}-c_{\theta_{2m}}c_{\theta_{3m}}) \\ -200(c_{\theta_{2m}}c_{\theta_{3m}}+s_{\theta_{2m}}s_{\theta_{3m}}) \end{bmatrix} \tag{B.6}$$

## B.2 SCARA

$$q_e = \begin{bmatrix} \theta_{1e} \\ \theta_{2e} \\ d_{3e} \end{bmatrix} \tag{B.7}$$

$$p_e = \begin{bmatrix} x_e \\ y_e \\ z_e \end{bmatrix} = \begin{bmatrix} 250(c_{\theta_{1e}}+c_{\theta_{1e}}c_{\theta_{2e}}-s_{\theta_{1e}}s_{\theta_{2e}}) \\ 250(s_{\theta_{1e}}+s_{\theta_{1e}}c_{\theta_{2e}}+c_{\theta_{1e}}s_{\theta_{2e}}) \\ 665-d_{3e} \end{bmatrix} \tag{B.8}$$

$$J_e = \frac{\partial p_e}{\partial q_e} = \begin{bmatrix} \frac{\partial x_e}{\partial q_e} & \frac{\partial y_e}{\partial q_e} & \frac{\partial z_e}{\partial q_e} \end{bmatrix} \tag{B.9}$$

$$\frac{\partial x_e}{\partial q_e} = \begin{bmatrix} -250(s_{\theta_{1e}}+s_{\theta_{1e}}c_{\theta_{2e}}+c_{\theta_{1e}}s_{\theta_{2e}}) \\ -250(c_{\theta_{1e}}s_{\theta_{2e}}+s_{\theta_{1e}}c_{\theta_{2e}}) \\ 0 \end{bmatrix} \tag{B.10}$$

$$\frac{\partial y_e}{\partial q_e} = \begin{bmatrix} 250(c_{\theta_{1e}}+c_{\theta_{1e}}c_{\theta_{2e}}-s_{\theta_{1e}}s_{\theta_{2e}}) \\ -250(s_{\theta_{1e}}s_{\theta_{2e}}-c_{\theta_{1e}}c_{\theta_{2e}}) \end{bmatrix} \tag{B.11}$$

$$\frac{\partial z_e}{\partial q_e} = \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix} \tag{B.12}$$

# C CÓDIGO DA CENTRAL DE CONTROLE EM LINGUAGEM PYTHON

```
import socket
import random
import time
from threading import Thread
from queue import Queue
import os
import math

socketPC = '127.0.0.1'
socketComunicacao = '200.135.72.8'
portaMestre = 9900
portaComunicacao = 9901
portaEscravo = 9902
BUFFER = 80

#Admitancia do mestre
Bxm = 0.4 #Ns/mm
Bym = 0.4
Bzm = 0.4

#Impedancia do ambiente virtual
Bxv = 0.1
Byv = 0.1
Bzv = 0.1
Kxv = 0.1
Kxy = 0.1
Kzv = 0.1

tempoGlobal = time.time()
Vbase = 500

#Restri  es virtuais
xo = 900
yo = 0
zo = 1000
rXY = 150
zm = 700
zM = 1100


def mestre(resposta,valores,forca,tempoMedio,tempoInicioMestre,posicaoRecebida):
    tcpMestre = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
    tcpMestre.connect((socketPC,portaMestre))
    tempoMedio.put(0)
    tempoInicio = time.time()
    respostaMestre = tcpMestre.recv(BUFFER).decode('utf 8')
    posicaoRecebida.put(respostaMestre)
    tcpMestre.send(respostaMestre.encode('utf 8'))
```

```
49      resposta.put('OK')
50      posicoes = respostaMestre.split(';')
51      pxei = int(posicoes[0])
52      pyei = int(posicoes[1])
53      pzei = int(posicoes[2])
54      while(True):
55          respostaMestre = tcpMestre.recv(BUFFER).decode('utf 8')
56          posicaoRecebida.put(respostaMestre)
57          posicoes = respostaMestre.split(';')
58          tempoFim = time.time()
59          tempo = tempoFim    tempoInicio
60          tempoMedio.put(tempoInicio    tempoGlobal)
61          forcas = leitura.get().split(';')
62          Fxs = int(forcas[0]) * 0.0122    200
63          Fys = int(forcas[1]) * 0.0122    200
64          Fzs =  (int(forcas[2]) * 0.0244    400)
65          forca.put(str(Fxs) + ';' + str(Fys) + ';' + str(Fzs))
66          pxef = int(posicoes[0])
67          pyef = int(posicoes[1])
68          pzef = int(posicoes[2])
69          if(math.sqrt((pow((pxef    xo),2)) + (pow((pyef    yo),2))) > rXY):
70              angulo = math.atan2((pyef    yo),(pxef    xo))
71              lx = rXY*math.cos(angulo)
72              ly = rXY*math.sin(angulo)
73              if(abs(pxef    xo) > abs(lx)):
74                 dxe = abs(pxef    xo)    abs(lx)
75                 vxe = (pxef    pxei)/tempo
76                 Fxv = Bxv*vxe + Kxv*dxe
77              else:
78                  Fxv =0
79              if(abs(pyef    yo) > abs(ly)):
80                 dye = abs(pyef    yo)    abs(ly)
81                 vye = (pyef    pyei)/tempo
82                 Fyv = Byv*vye + Kyv*dye
83              else:
84                  Fyv = 0
85          else:
86              Fxv = 0
87              Fyv = 0
88
89          if (pzef > zM) or (pzef < zm):
90              if pzef > zM:
91                 dze = pzef    zM
92              if pzef < zm:
93                 dze = zM    pzef
94              vze = (pzef    pzei)/tempo
95              Fzv = Bzv*vze + Kzv*dze
96          else:
97              Fzv = 0
98
```

```
99          Vx = (Fxs    Fxv)/Bxm
100         Vy = (Fys    Fyv)/Bym
101         Vz = (Fzs    Fzv)/Bzm
102         Vref = math.sqrt(pow(Vx,2) + pow(Vy,2) + pow (Vz,2))
103         if float(Vref/Vbase) < 0.1:
104             PercVel = float(0.1*100)
105         elif(float(Vref/Vbase) > 1):
106             Vx = 300
107             Vy = 300
108             Vz = 300
109             PercVel = float(1*100)
110         else:
111             PercVel = float((Vref/Vbase)*100)
112         pxei = pxef
113         pyei = pyef
114         pzei = pzef
115         dx = Vx*tempo
116         dy = Vy*tempo
117         dz = Vz*tempo
118         px = int(dx)
119         py = int(dy)
120         pz = int(dz)
121         mensagem = str(int(posicoes[0]) + px) + ';' + str(int(posicoes[1]) + py)
        + ';' + str(int(posicoes[2]) + pz) + ';' + posicoes[3] + ';' + posicoes[4] +
        ';' + posicoes[5] + ';' + posicoes[6] + ';' + posicoes[7] + ';' + posicoes
        [8] + ';' + posicoes[9] + ';' + posicoes[10] + ';' + str(PercVel)
122         valores.put(mensagem)
123         tempoInicio = time.time()
124         tcpMestre.send(mensagem.encode('utf 8'))
125         resposta.put('OK')
126
127 def escravo(posicaoRecebida, posicaoEscravo,tempoInicioMestre, tempoEscravo):
128     tcpEscravo = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
129     tcpEscravo.connect((socketPC,portaEscravo))
130     while(True):
131         posicao = posicaoRecebida.get()
132         tcpEscravo.send(posicao.encode('utf 8'))
133         respostaEscravo = tcpEscravo.recv(BUFFER).decode('utf 8')
134         tempoFimEscravo = time.time()    tempoGlobal
135         tempoEscravo.put(tempoFimEscravo)
136         posicaoEscravo.put(respostaEscravo)
137
138 def monitoramento(leitura,resposta):
139     tcpComunicacao = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
140     tcpComunicacao.bind((socketComunicacao,portaComunicacao))
141     tcpComunicacao.listen()
142     comunicacao = tcpComunicacao.accept()[0]
143     while(True):
144         forcas = comunicacao.recv(BUFFER).decode('utf 8')
145         leitura.put(forcas)
```

```
146         comunicacao.send(resposta.get().encode('utf 8'))
147
148 def salvarValores(valores,posicaoEscravo,forca,tempoMedio,tempoEscravo):
149     arquivo = open('valoresPosicaoForca.txt','a')
150     arquivo.write('TempoMestre;TempoEscravo;Px;Py;Pz;Pxe;Pye;Pze;Fx;Fy;Fz\n')
151     arquivo.close()
152     while(True):
153         arquivo = open('valoresPosicaoForca.txt','a')
154         valoresPosicao = valores.get().split(';')
155         valoresEscravo = posicaoEscravo.get().split(';')
156         valoresForcas = forca.get().split(';')
157         tempo1 = tempoMedio.get()
158         tempo2 = tempoEscravo.get()
159         arquivo.write(str(tempo1).replace('.',',') + ';' + str(tempo2).replace('
    .',',') + ';' + valoresPosicao[0].replace('.',',') + ';' + valoresPosicao[1].
    replace('.',',') + ';' + valoresPosicao[2].replace('.',',') + ';' +
    valoresEscravo[0].replace('.',',') + ';' + valoresEscravo[1].replace('.',',')
     + ';' + valoresEscravo[2].replace('.',',') + ';' + str(float(valoresForcas
    [0])).replace('.',',') + ';' + str(float(valoresForcas[1])).replace('.',',')
    + ';' + str(float(valoresForcas[2])).replace('.',',') + '\n')
160         arquivo.close()
161
162 leitura = Queue()
163 posicao = Queue()
164 resposta = Queue()
165 valores = Queue()
166 forca = Queue()
167 tempoMedio = Queue()
168 tempoInicioMestre = Queue()
169 tempoEscravo = Queue()
170 posicaoRecebida = Queue()
171 posicaoEscravo = Queue()
172
173 threadMestre = Thread(target=mestre,args=[resposta,valores,forca,tempoMedio,
    tempoInicioMestre,posicaoRecebida])
174 threadMonitoramento = Thread(target=monitoramento,args=[leitura,resposta])
175 threadSalvar = Thread(target=salvarValores,args=[valores,posicaoEscravo,forca,
    tempoMedio,tempoEscravo])
176 threadEscravo = Thread(target=escravo,args=[posicaoRecebida,posicaoEscravo,
    tempoInicioMestre,tempoEscravo])
177 threadMestre.start()
178 threadMonitoramento.start()
179 threadSalvar.start()
180 threadEscravo.start()
```

# D VARIÁVEIS DO MODELO DE OPERAÇÃO DO MANIPULADOR MESTRE DA SIMULAÇÃO EM TEMPO DISCRETO

$$p_m(k) = \begin{bmatrix} p_{mx}(k) \\ p_{my}(k) \\ p_{mz}(k) \end{bmatrix} \tag{D.1}$$

$$v_m(k) = \begin{bmatrix} v_{mx}(k) \\ v_{my}(k) \\ v_{mz}(k) \end{bmatrix} \tag{D.2}$$

$$p_a(k) = \begin{bmatrix} p_{ax}(k) \\ p_{ay}(k) \\ p_{az}(k) \end{bmatrix} \tag{D.3}$$

$$v_a(k) = \begin{bmatrix} v_{ax}(k) \\ v_{ay}(k) \\ v_{az}(k) \end{bmatrix} \tag{D.4}$$

$$F_s(k) = \begin{bmatrix} F_{sx}(k) \\ F_{sy}(k) \\ F_{sz}(k) \end{bmatrix} \tag{D.5}$$

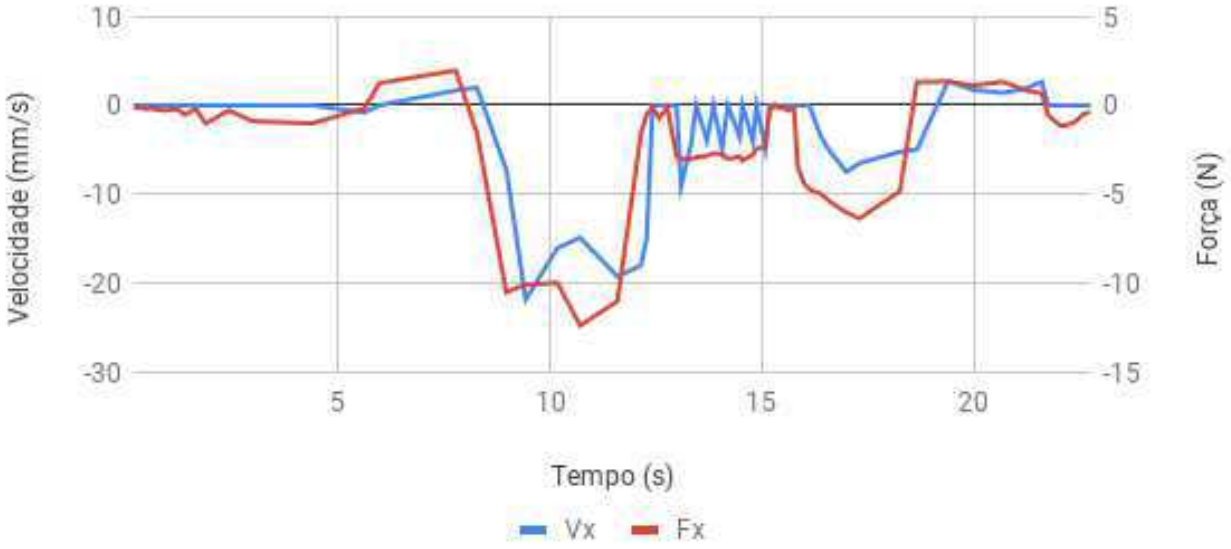
$$B_m = \begin{bmatrix} B_{mx} & 0 & 0 \\ 0 & B_{my} & 0 \\ 0 & 0 & B_{mz} \end{bmatrix} \tag{D.6}$$

$$B_a = \begin{bmatrix} B_{ax} & 0 & 0 \\ 0 & B_{ay} & 0 \\ 0 & 0 & B_{az} \end{bmatrix} \tag{D.7}$$

$$K_a = \begin{bmatrix} K_{ax} & 0 & 0 \\ 0 & K_{ay} & 0 \\ 0 & 0 & K_{az} \end{bmatrix} \tag{D.8}$$

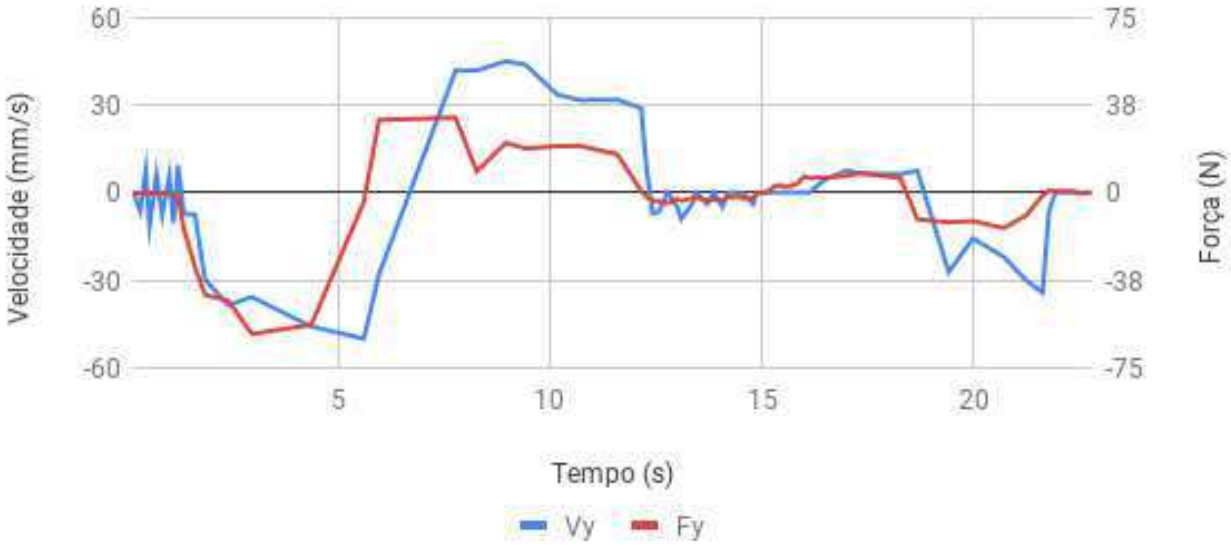
# E RESULTADOS DAS SIMULAÇÕES 1 E 2 UTILIZANDO O ROBOTSTUDIO

Figura 70 – Simulação 1 - Velocidade no eixo x.



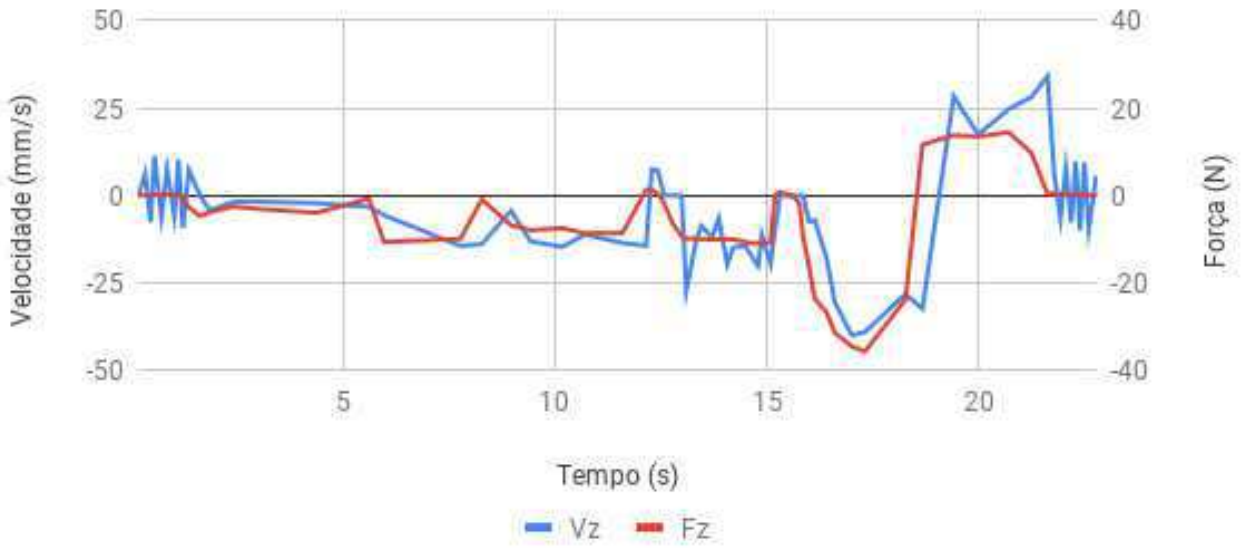

Fonte: do autor.

Figura 71 – Simulação 1 - Velocidade no eixo y.



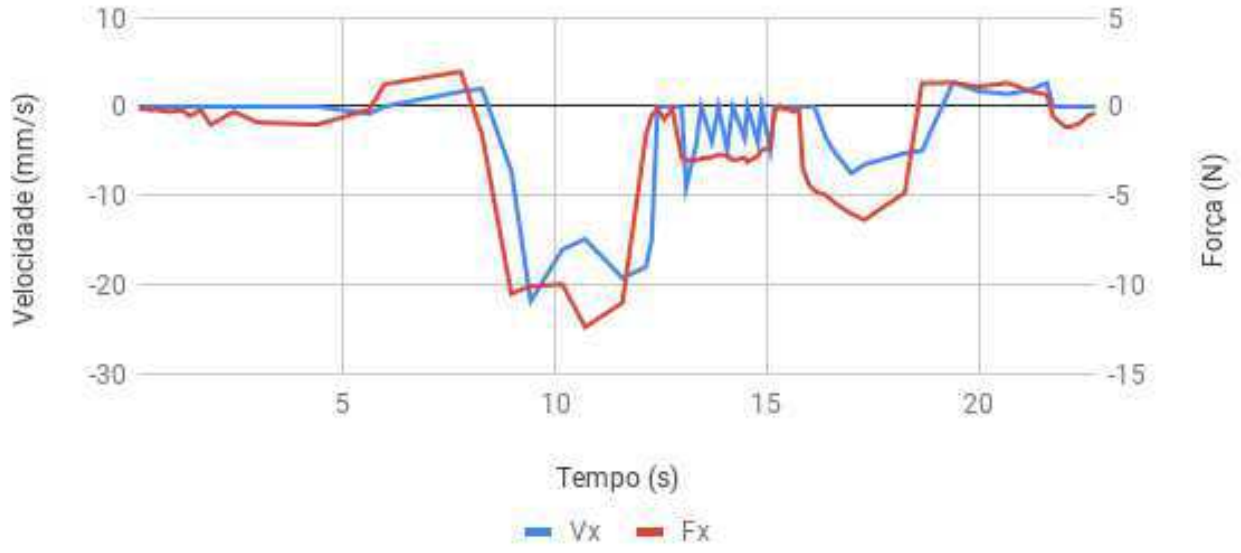

Fonte: do autor.

Figura 72 – Simulação 1 - Velocidade no eixo z.



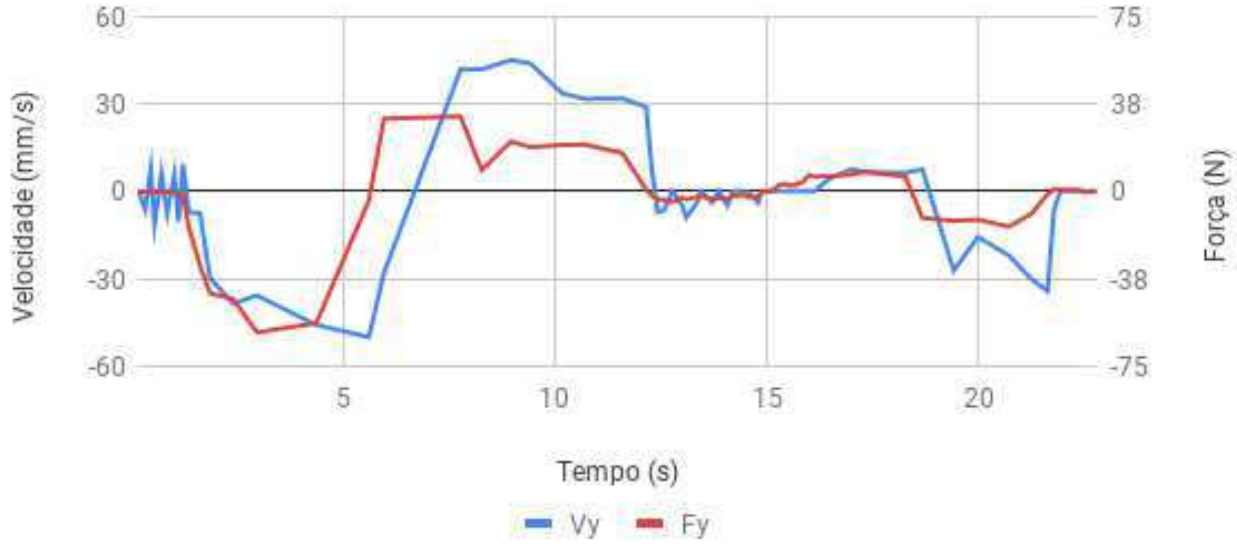

Fonte: do autor.

Figura 73 – Simulação 1 - Velocidade no eixo x.



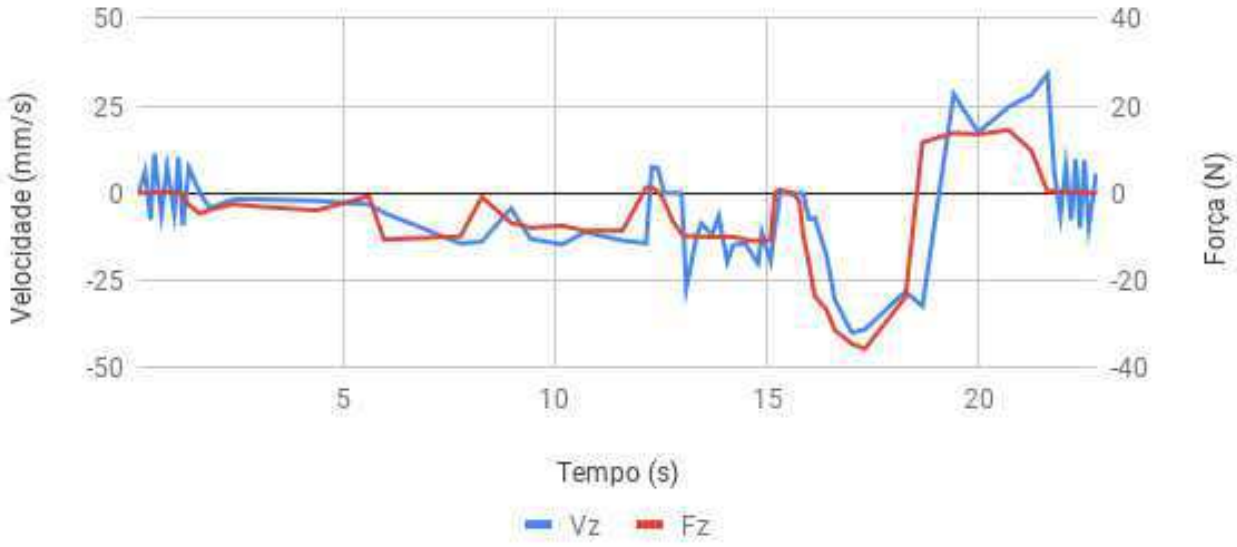

Fonte: do autor.

Figura 74 – Simulação 1 - Velocidade no eixo y.



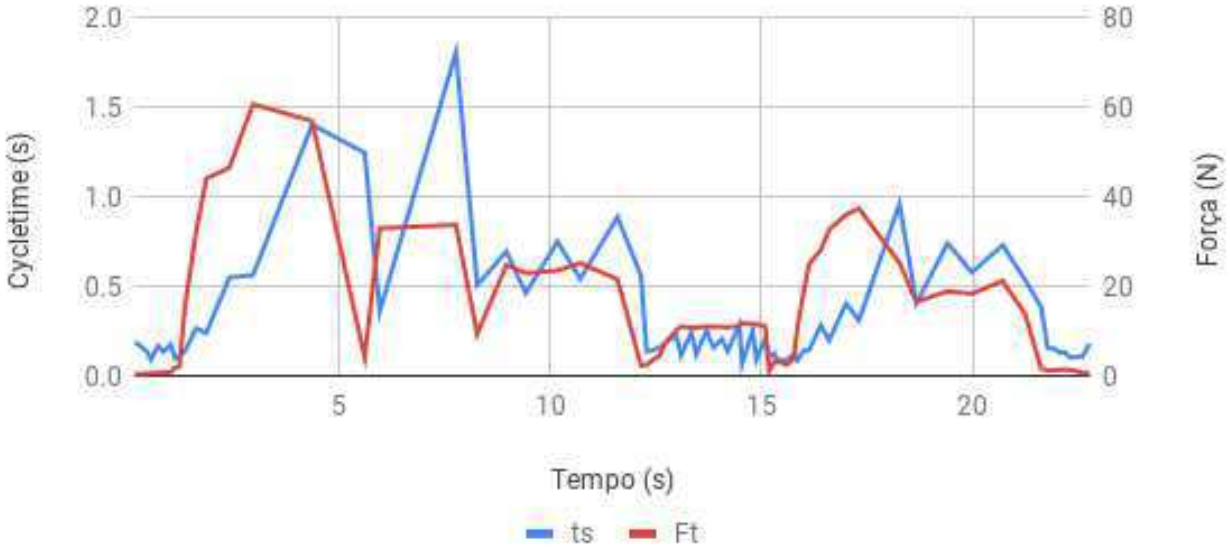

Fonte: do autor.

Figura 75 – Simulação 1 - Velocidade no eixo z.



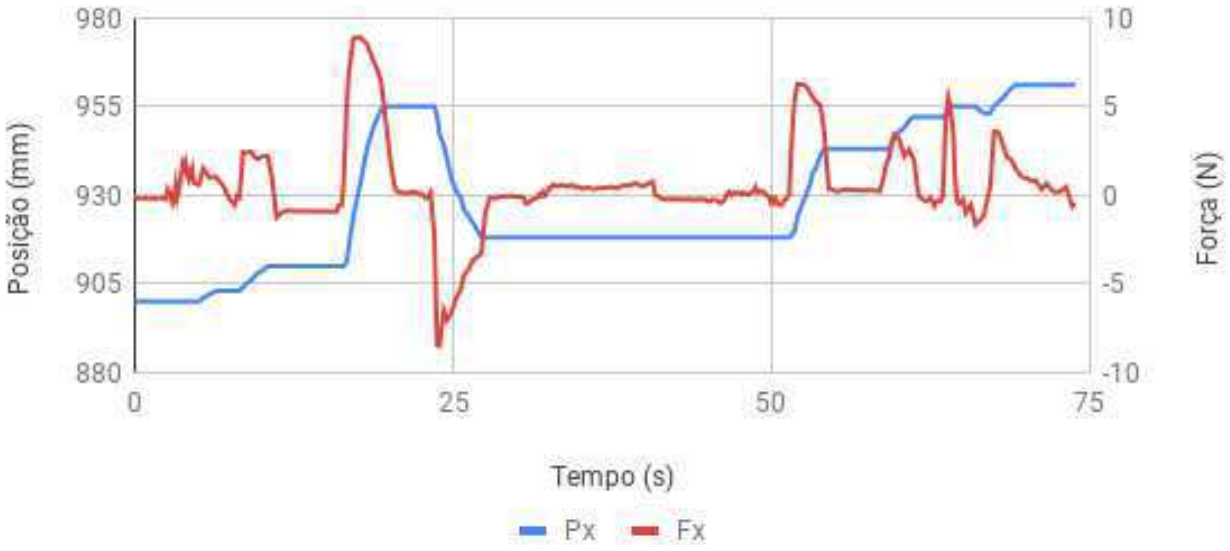

Fonte: do autor.

Figura 76 – Simulação 1 - Relação entre força e atraso.



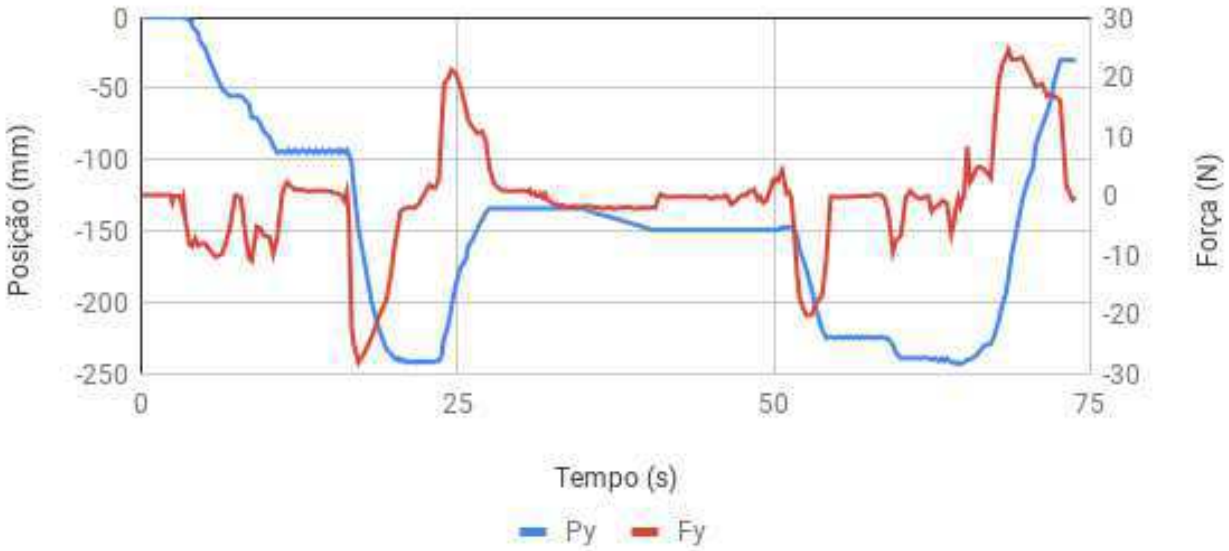

Fonte: do autor.

Figura 77 – Simulação 2 - Posição no eixo x.



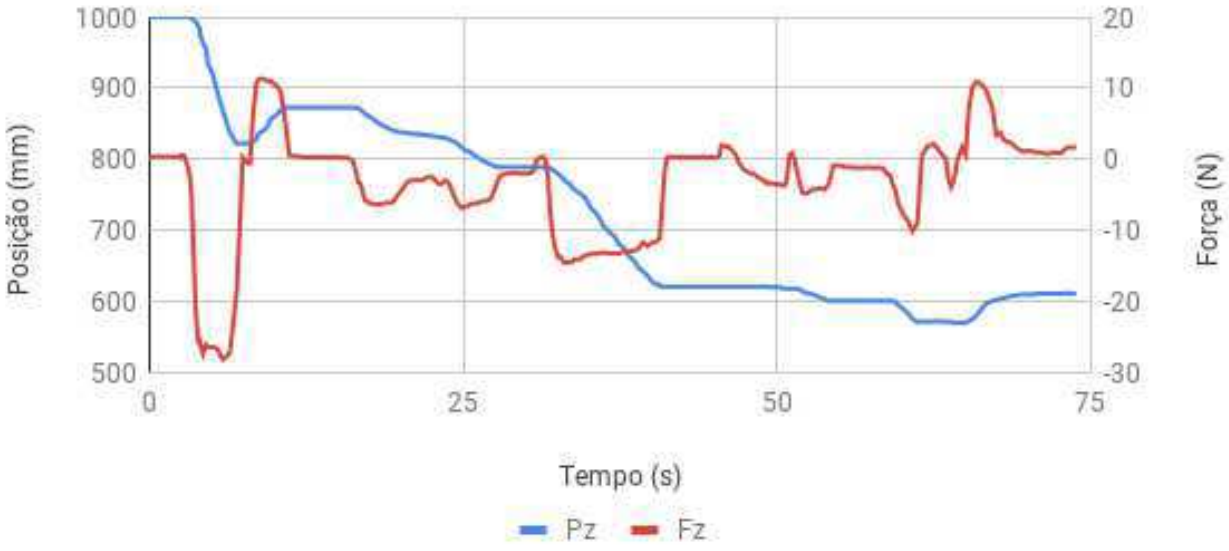

Fonte: do autor.

Figura 78 – Simulação 2 - Posição no eixo y.



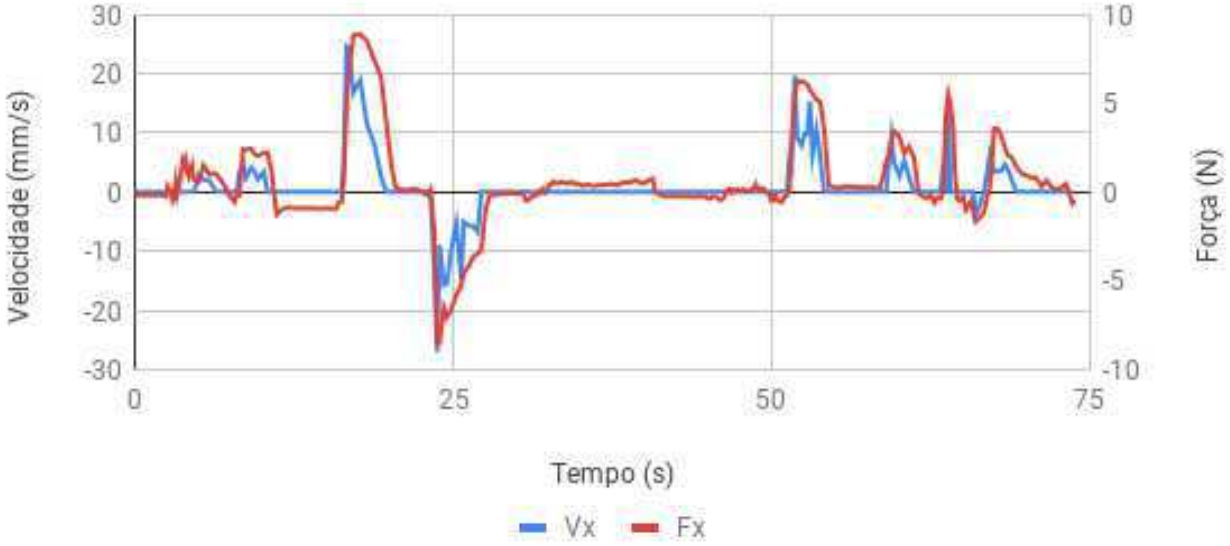

Fonte: do autor.

Figura 79 – Simulação 2 - Posição no eixo z.



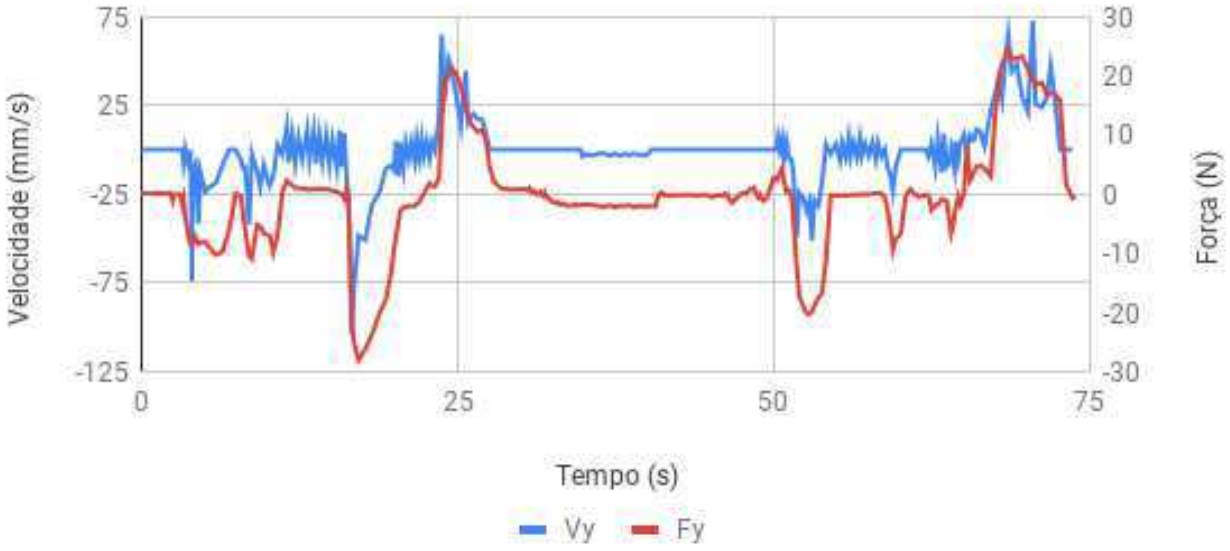

Fonte: do autor.

Figura 80 – Simulação 2 - Velocidade no eixo x.



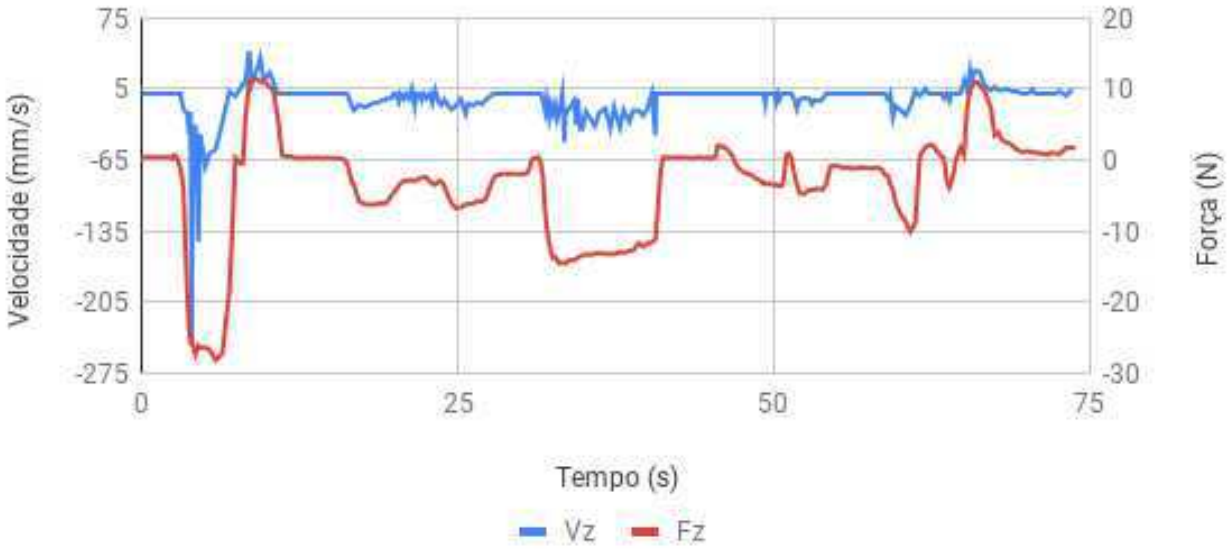

Fonte: do autor.

Figura 81 – Simulação 2 - Velocidade no eixo y.



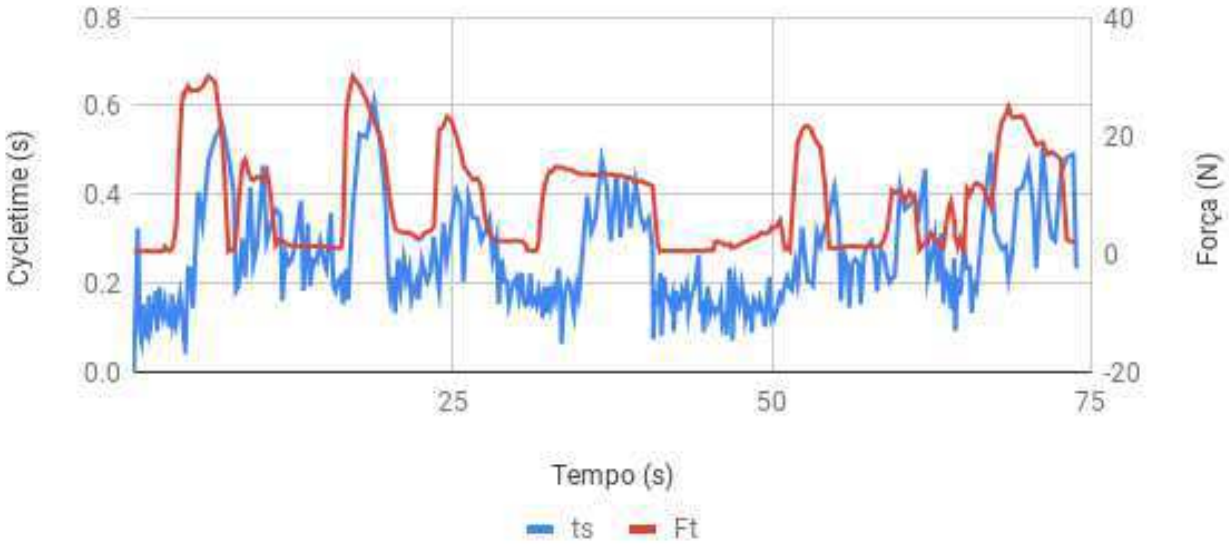

Fonte: do autor.

Figura 82 – Simulação 2 - Velocidade no eixo z.




Fonte: do autor.

Figura 83 – Simulação 2 - Relação entre força e atraso.




Fonte: do autor.

# F ESTIMAÇÃO DO *CYCLETIME* POR MÍNIMOS QUADRADOS ORDINÁRIOS (MQO)

A estimação do *cycletime* foi feita utilizando o método dos Mínimos Quadrados Ordinários, e considerando um modelo de regressão linear de quatro parâmetros, conforme a Equação F.1. Foram utilizadas 578 amostras, oriundas dos parâmetros e resultados das simulações feitas no trabalho.

$$t_s(k) = \lambda_0 + \lambda_1 t_s(k-1) + \lambda_2 v_d(k) + \lambda_3 v_b(k) \tag{F.1}$$

$$k = 1, 2, ..., n \tag{F.2}$$

$$\lambda = \begin{bmatrix} \lambda_0 & \lambda_1 & \lambda_2 & \lambda_3 \end{bmatrix}^T \tag{F.3}$$

$$\hat{\lambda} = (X^T X)^{-1} XY \tag{F.4}$$

$$Y = \begin{bmatrix} t_s(1) & t_s(2) & ... & t_s(n) \end{bmatrix}^T \tag{F.5}$$

$$X = \begin{bmatrix} 1 & t_s(0) & v_d(1) & v_b(1) \\ 1 & t_s(1) & v_d(2) & v_b(2) \\ \vdots & \vdots & \vdots & \vdots \\ 1 & t_s(n-1) & v_d(n) & v_b(n) \end{bmatrix} \tag{F.6}$$

$$\hat{Y} = X\hat{\lambda} \tag{F.7}$$

$$s^2 = \frac{1}{n-1} \sum_{k=0.8}^{n} (\hat{Y}(k) - \overline{Y})^2 \tag{F.8}$$