

**MINISTÉRIO DA DEFESA
EXÉRCITO BRASILEIRO
DEPARTAMENTO DE CIÊNCIA E TECNOLOGIA
INSTITUTO MILITAR DE ENGENHARIA
CURSO DE MESTRADO EM ENGENHARIA MECÂNICA**

FELYPPE LABORDE MARINHO SANTOS

**MODELAGEM E SIMULAÇÃO DE ROBÔS SERIAIS
BASEADAS EM HELICOIDES**

Rio de Janeiro
2017

INSTITUTO MILITAR DE ENGENHARIA

FELYPPE LABORDE MARINHO SANTOS

**MODELAGEM E SIMULAÇÃO DE ROBÔS SERIAIS BASEADAS EM
HELICOIDES**

Dissertação de Mestrado apresentada ao Curso de Mestrado em Engenharia Mecânica do Instituto Militar de Engenharia, como requisito parcial para obtenção do título de Mestre em Ciências em Engenharia Mecânica.

Orientador: Cel Luiz Paulo Gomes Ribeiro - Dr. Eng.

Rio de Janeiro
2017

INSTITUTO MILITAR DE ENGENHARIA
Praça General Tibúrcio, 80 - Praia Vermelha
Rio de Janeiro-RJ CEP 22290-270

Este exemplar é de propriedade do Instituto Militar de Engenharia, que poderá incluí-lo em base de dados, armazenar em computador, microfilmar ou adotar qualquer forma de arquivamento.

É permitida a menção, reprodução parcial ou integral e a transmissão entre bibliotecas deste trabalho, sem modificação de seu texto, em qualquer meio que esteja ou venha a ser fixado, para pesquisa acadêmica, comentários e citações, desde que sem finalidade comercial e que seja feita a referência bibliográfica completa.

Os conceitos expressos neste trabalho são de responsabilidade do autor e do orientador.

620.1 Santos, Felyppe Laborde Marinho
S237m Modelagem e simulação de robôs seriais baseadas em helicoides / Felyppe Laborde Marinho Santos; orientado por Luiz Paulo Gomes Ribeiro. – Rio de Janeiro: Instituto Militar de Engenharia, 2017.

125 p.:il.

Dissertação (mestrado) – Instituto Militar de Engenharia – Rio de Janeiro, 2017.

1. Curso de Engenharia Mecânica – teses, dissertações. 2. Realidade Virtual. 3. Helicoides. I. Ribeiro, Luiz Paulo Gomes. II. Modelagem e Simulação de Robôs Seriais Baseadas em Helicoides. III. Instituto Militar de Engenharia.

INSTITUTO MILITAR DE ENGENHARIA

FELYPPE LABORDE MARINHO SANTOS

**MODELAGEM E SIMULAÇÃO DE ROBÔS SERIAIS BASEADAS EM
HELICOIDES**

Dissertação de Mestrado apresentada ao Curso de Mestrado em Engenharia Mecânica do Instituto Militar de Engenharia, como requisito parcial para obtenção do título de Mestre em Ciências em Engenharia Mecânica.

Orientador: Cel Luiz Paulo Gomes Ribeiro - Dr. Eng.

Aprovada em 04 de setembro de 2017 pela seguinte Banca Examinadora:

Cel Luiz Paulo Gomes Ribeiro - Dr. Eng. do IME - Presidente

TC Antonio Eduardo Carrilho da Cunha - D.Sc. do IME

Max Suell Dutra - Dr. Ing. da UFRJ

Rio de Janeiro
2017

Esta obra é dedicada a todos que desejam alcançar os seus sonhos, independente do quanto tenham que trabalhar para atingi-los.

AGRADECIMENTOS

Agradeço a Deus, pois sem ele nada disso seria possível.

A minha família, minha mãe, Maria José, meu pai, José Eugênio, e minha irmã, Laryssa Laborde, pelo suporte integral.

A Andrezza Bonfim, por todo amor, carinho e incentivo em todos os momentos desta etapa.

Ao meu orientador Cel Luiz Paulo Gomes Ribeiro, pelos diversos ensinamentos, pelas palavras de apoio, pela paciência em ensinar e ouvir, por indicar o melhor caminho a ser seguido e por sempre me mostrar que eu era capaz.

Ao IME, pela estrutura oferecida e pela oportunidade que me foi dada de ingressar e me formar no mestrado.

A todos do IDR Lab, pelo apoio, pela união e pelos momentos de descontração que se fizeram necessários.

Aos professores que me fizerem crescer, intelectual e pessoalmente, através de seus ensinamentos.

A todos os funcionários do Instituto Militar de Engenharia que, de alguma forma, colaboraram com esta etapa que se encerra.

A CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior), pelo suporte financeiro durante o mestrado.

A todos que de alguma maneira contribuíram para o meu crescimento durante esta caminhada.

SUMÁRIO

LISTA DE ILUSTRAÇÕES	8
LISTA DE TABELAS	11
LISTA DE ABREVIATURAS E SÍMBOLOS	12
1 INTRODUÇÃO	17
1.1 Motivação	17
1.2 Formulação do Problema	18
1.3 Objetivos da Dissertação	19
1.4 Organização da Dissertação	20
2 ROBÓTICA INDUSTRIAL	22
2.1 Surgimento da Robótica	22
2.2 A Robótica	25
2.3 Robô Industrial	26
2.4 Desenvolvimento da Robótica Industrial	27
2.5 Classificação dos Robôs Industriais Seriais	30
2.6 Estatísticas da Robótica Industrial	33
2.6.1 Principais Mercados de Robôs Industriais	33
2.6.2 Panorama do Mercado Brasileiro de Robôs Industriais	34
2.6.3 Custos do Sistema Robótico	35
2.6.4 Expectativas para os Robôs Industriais	35
2.7 Tipos de Programação de Robôs Industriais	37
2.7.1 Programação <i>On-line</i> de Robôs Industriais	37
2.7.1.1 Etapas de desenvolvimento da Programação <i>On-line</i>	40
2.7.2 Programação <i>Off-line</i> de Robôs Industriais	41
2.8 Softwares associados à modelagem, simulação e programação de robôs	42
2.8.1 Software de Simulação de Robótica do Pegasus (RSS)	43
2.8.2 Virtual Robot Experimentation Platform (V-REP)	44
3 METODOLOGIAS EXISTENTES	48
3.1 Cinemática de Robôs Seriais	48

3.2	Convenção de Denavit-Hartenberg	49
3.3	Robótica Baseada em Helicoides	56
3.4	Validação dos Métodos	60
3.5	Comparação entre os Métodos	64
4	CARPA: MODELAGEM, SIMULAÇÃO E PROGRAMAÇÃO	66
4.1	Modelagem e Simulação	66
4.2	O Sistema CAPRON	67
4.3	O Sistema CARPA	74
4.3.1	Inserir um robô modelado	75
4.3.2	Inserir novos objetos	78
4.3.3	Associar cor aos objetos.....	83
4.3.4	Posicionar objetos.....	84
4.3.5	Programação de um robô	87
5	CONCLUSÕES E PERSPECTIVAS	94
5.1	Recapitulação Sintetizada	94
5.2	Contribuições Científicas	95
5.3	Conclusão	96
5.4	Perspectivas	97
6	REFERÊNCIAS BIBLIOGRÁFICAS	98
7	<u>APÊNDICES</u>	101
7.1	APÊNDICE 1: ROBÓTICA BASEADA EM HELICOIDES	102
7.1.1	Localização de um Corpo Rígido	102
7.1.1.1	Teorema de Euler - Representação da Orientação	104
7.1.1.2	Teorema de Chasles – Representação da Localização	106
7.1.2	Teoria dos Helicoides	108
7.1.2.1	Representação da Rotação por Meio de Eixo de Helicoide	108
7.1.2.2	Deslocamento Helicoidal	114
7.1.3	Método dos Deslocamentos dos Helicoides Sucessivos	119
7.1.4	Jacobiano de um Manipulador Serial	123

LISTA DE ILUSTRAÇÕES

FIG.2.1	Cena da peça de teatro R.U.R. (SICILIANO & KHATIB, 2008)	22
FIG.2.2	Comparação entre custos do robô e do homem-hora na indústria de fabricação nos EUA (adaptado de UNECE (2004))	23
FIG.2.3	Efeito dos objetivos de desempenho (adaptado de SLACK et al. (2002))	24
FIG.2.4	Fatores associados à robótica (MACEDO & RIBEIRO, 2015)	25
FIG.2.5	Primeiro robô industrial, o <i>Unimate</i> (SICILIANO & KHATIB, 2008)	28
FIG.2.6	Robô Paralelo (cadeia fechada) (SICILIANO et al., 2009)	30
FIG.2.7	Tipos de Manipuladores Robóticos e sua área de trabalho (Autoria Própria) 32	
FIG.2.8	Novas unidades instaladas de robôs industriais por ano (adaptado de IFR (2016))	33
FIG.2.9	Comparativo entre Brasil e Mundo de novas unidades de robôs industriais instaladas entre 2010 e 2015 (adaptado de IFR (2016))	34
FIG.2.10	Comparação entre os valores do robô e do sistema de <i>software</i> em 2015 (adaptado de IFR (2016))	35
FIG.2.11	Célula de Trabalho durante a Programação <i>On-line</i> no IDR Lab (Autoria própria)	37
FIG.2.12	<i>Teach Pendant (TP)</i> do robô Pegasus instalado no IDR Lab (Autoria própria)	39
FIG.2.13	Relação entre os Tempos de Programação e o de Execução da Tarefa (Autoria Própria)	39
FIG.2.14	Tela Inicial do RSS do Pegasus (Autoria Própria)	43
FIG.2.15	Tela Inicial do V-REP (Autoria Própria)	44
FIG.2.16	V-REP - inserir robô (Autoria Própria)	46
FIG.2.17	V-REP - inserir cada corpo (Autoria Própria)	46
FIG.2.18	V-REP - base inserida (Autoria Própria)	47
FIG.2.19	V-REP - robô inserido (Autoria Própria)	47
FIG.3.1	Identificação dos elos do robô Pegasus, conforme Denavit-Hartenberg (Autoria Própria)	49
FIG.3.2	Identificação das juntas do robô Pegasus, conforme Denavit-Hartenberg (Autoria Própria)	50

FIG.3.3	Identificação do eixo z_i do robô Pegasus, conforme Denavit-Hartenberg (Autoria Própria)	50
FIG.3.4	Determinação do sistema de coordenadas absoluto por D-H (Autoria Própria)	51
FIG.3.5	Determinação dos sistemas O_i e O'_i por D-H (Autoria Própria)	52
FIG.3.6	Determinação dos eixos x_i e y_i por D-H (Autoria Própria)	53
FIG.3.7	Determinação dos eixos $z_{i'}$, $x_{i'}$ e $y_{i'}$ por D-H (Autoria Própria)	53
FIG.3.8	Determinação do sistema n e n' por D-H (Autoria Própria)	54
FIG.3.9	Identificação de elos e juntas do Robô Pegasus (Autoria Própria)	57
FIG.3.10	Representação do parâmetro s no Robô Pegasus (Autoria Própria)	57
FIG.3.11	Representação do parâmetro s_0 no Robô Pegasus (Autoria Própria)	58
FIG.3.12	Referencial da ferramenta do robô Pegasus (Autoria Própria)	59
FIG.3.13	Postura de <i>Home</i> com variáveis de junta $[0\ 0\ 0\ 0]^T$ (Autoria Própria)	60
FIG.3.14	Postura com variáveis de junta $[45\ 0\ 30\ 0\ 0]^T$ (Autoria Própria)	62
FIG.3.15	Referenciais utilizados por (a) Teoria dos Helicoides e (b) D-H (Autoria Própria)	65
FIG.4.1	Tela Inicial do CARPA (Autoria Própria)	66
FIG.4.2	Tela Inicial do CAPRON (Autoria Própria)	67
FIG.4.3	FMS do robô Pegasus, localizado no IDR Lab (Autoria Própria)	67
FIG.4.4	TP do robô Pegasus real e virtual (SANTOS et al. (2017a))	68
FIG.4.5	Sequência de acesso às funcionalidades do TP (SANTOS et al. (2017a))	70
FIG.4.6	Função TCH (SANTOS et al. (2017a))	71
FIG.4.7	Função Pmove - PMV (SANTOS et al. (2017a))	71
FIG.4.8	Alunos de graduação utilizando o CAPRON (Autoria Própria)	72
FIG.4.9	Posições inicial e final das peças no local de armazenamento (Autoria Própria)	73
FIG.4.10	Tempos de programação dos usuários por grupo (SANTOS et al., 2017a) 74	
FIG.4.11	Robôs genéricos modelados no CARPA (Autoria Própria)	74
FIG.4.12	Inserir robô modelado: evento botão "Arquivo" (Autoria Própria)	76
FIG.4.13	Inserir robô modelado: evento botão "Abrir" na seção "Robôs" (Autoria Própria)	77
FIG.4.14	Inserir robô modelado: evento escolher o robô (Autoria Própria)	77

FIG.4.15	Inserir robô modelado: robô IRB120 selecionado (Autoria Própria)	78
FIG.4.16	Inserir novos objetos: informar o número de juntas do objeto (Autoria Própria)	79
FIG.4.17	Inserir novos objetos: parâmetros no <i>treeview</i> (Autoria Própria)	80
FIG.4.18	Inserir novos objetos: evento botão "IMG/S" (Autoria Própria)	80
FIG.4.19	Inserir novos objetos: objeto inserido no ambiente virtual (Autoria Própria)	
	81	
FIG.4.20	Inserir novos objetos: demais objetos inseridos (Autoria Própria)	81
FIG.4.21	Inserir novos objetos: informar o número de juntas do robô (Autoria Própria)	82
FIG.4.22	Inserir novos objetos: parâmetros do robô inseridos (Autoria Própria)	83
FIG.4.23	Associar cor ao objeto: evento botão "COR" (Autoria Própria)	84
FIG.4.24	Associar cor ao objeto: escolher as tonalidade de cor (Autoria Própria)	84
FIG.4.25	Posicionar um robô inserido: definindo o valor de P_z (Autoria Própria)	85
FIG.4.26	Posicionar um robô inserido: definindo o valor de P_y (Autoria Própria)	86
FIG.4.27	Posicionar um robô inserido: definindo o valor de P_x (Autoria Própria)	86
FIG.4.28	Ensinando o ponto 0 (zero) (Autoria Própria)	89
FIG.4.29	Ensinando o ponto 1 (Autoria Própria)	89
FIG.4.30	Ensinando os pontos 2, 3 e 4 (Autoria Própria)	90
FIG.4.31	Tela com os códigos implementados do robô Pegasus (Autoria Própria)	90
FIG.4.32	Comando de <i>output</i> do robô Pegasus (Autoria Própria)	91
FIG.4.33	Procedimento para programação do robô Pegasus (Autoria Própria)	92
FIG.4.34	Procedimento para salvar o programa do robô Pegasus (Autoria Própria)	93
FIG.7.1	Referenciais absoluto e móvel (TSAI, 1999)	103
FIG.7.2	Representação de P em dois referenciais com origem coincidente (SICILIANO et al., 2009)	104
FIG.7.3	Rotação em torno do eixo do helicoide (TSAI, 1999)	109
FIG.7.4	Projeção de $S_P P_2^r$ sobre $S_P P_1$ (TSAI, 1999)	110
FIG.7.5	Rotação e translação em torno de um único eixo (TSAI, 1999)	115
FIG.7.6	Deslocamento de um corpo por dois helicoides sucessivos (TSAI, 1999)	119
FIG.7.7	Deslocamento de um corpo por n helicoides sucessivos (TSAI, 1999)	121
FIG.7.8	Fluxograma da Cinemática Direta por Helicoides (Autoria Própria)	122

LISTA DE TABELAS

TAB.3.1	Parâmetros de Denavit-Hartenberg do robô Pegasus	55
TAB.3.2	Parâmetros s e s_0 da Teoria dos Helicoides do robô Pegasus	58

LISTA DE ABREVIATURAS E SÍMBOLOS

ABREVIATURAS

ABB	-	<i>Asea Brown Boveri</i>
CAD	-	<i>Computer-Aided Design</i>
CAPRON	-	<i>Computer-aided Pegasus Robot On-line Programming</i>
CARPA	-	<i>Computer-Aided Serial Robots Programming, Modelling and Simulation Analysis</i>
CRIA	-	<i>China Robot Industry Alliance</i>
END	-	Estratégia Nacional de Defesa
FANUC	-	<i>Fuji Automatic Numerical Control</i>
FMS	-	<i>Flexible Manufacturing System</i>
IBM	-	<i>International Business Machines</i>
IDR Lab	-	<i>Industrial and Defense Robotics Laboratory</i>
IDE	-	<i>Integrated Development Environment</i>
IES	-	Instituições de Ensino Superior
IFR	-	<i>International Federation of Robotics</i>
IHM	-	Interface Homem-Máquina
IME	-	Instituto Militar de Engenharia
IoT	-	<i>Internet of Things</i>
JIRA	-	<i>Japan Industrial Robot Association</i>
KUKA	-	<i>Keller und Knappich Augsburg</i>
MCL	-	<i>Manufacturing Control Language</i>
MIT	-	<i>Massachusetts Institute of Technology</i>
MTH	-	Matriz de Transformação Homogênea
NC	-	<i>Numerical Control</i>
OOP	-	<i>Object Oriented Programming</i>
PD&I	-	Pesquisa, Desenvolvimento e Inovação
PME	-	Pequenas e Médias Empresas
PND	-	Política Nacional de Defesa
PPGMEC	-	Programa de Pós-Graduação em Engenharia Mecânica
PUMA	-	<i>Programmable Universal Machine for Assembly</i>
RFID	-	<i>Radio-Frequency Identification</i>

RIA	- <i>Robot Institute of America</i>
RSS	- <i>Pegasus Robotics Simulation Learning System</i>
R.U.R.	- <i>Rossum's Universal Robots</i>
SCARA	- <i>Selective Compliance Assembly Robot Arm</i>
TCP	- <i>Tool Center Point</i>
TI	- Tecnologia da Informação
TP	- <i>Teach Pendant</i>
VR	- <i>Virtual Reality</i>
V-REP	- <i>Virtual Robot Experimentation Platform</i>

SÍMBOLOS

A	- <i>Referencial Absoluto</i>
B	- <i>Referencial Móvel</i>
(x, y, z)	- <i>Sistema de eixos coordenados do Referencial Absoluto</i>
(u, v, w)	- <i>Sistema de eixos coordenados do Referencial Móvel</i>
O	- <i>Origem do Referencial Absoluto</i>
Q	- <i>Origem do Referencial Móvel</i>
p^A	- <i>Ponto em B em relação ao Referencial Absoluto</i>
p^B	- <i>Ponto em B em relação ao Referencial Móvel</i>
R_B^A	- <i>Matriz de Rotação</i>
q_x^A	- <i>Translação em relação ao Referencial Absoluto em torno do eixo na direção x</i>
q_y^A	- <i>Translação em relação ao Referencial Absoluto em torno do eixo na direção y</i>
q_z^A	- <i>Translação em relação ao Referencial Absoluto em torno do eixo na direção z</i>
\tilde{p}^A	- <i>Ponto em B em relação ao Referencial Absoluto representado na forma homogênea</i>
\tilde{p}^B	- <i>Ponto em B em relação ao Referencial Móvel representado na forma homogênea</i>
T_B^A	- <i>Matriz de Transformação Homogênea</i>
s	- <i>Direção do eixo de cada junta em relação ao referencial absoluto</i>
s_0	- <i>Vetor posição (instantânea) em relação ao referencial absoluto</i>
\dot{d}	- <i>Movimento de translação ao longo do eixo do helicoide</i>
$\$_i$	- <i>Helicoide da i-ésima junta</i>
\dot{x}	- <i>Vetor velocidades angular e linear</i>
\dot{q}	- <i>Vetor composto pelas velocidades das juntas do robô</i>

- v_x *Velocidade linear em torno do eixo na direção x*
- v_y *Velocidade linear em torno do eixo na direção y*
- v_z *Velocidade linear em torno do eixo na direção z*
- J_H *Jacobiano baseado em helicoides*
- v *Vetor das velocidades generalizadas do efetuador final*
- $\dot{\theta}$ *Movimento de rotação em torno do eixo do helicoide*
- ω_x *Velocidade angular em torno do eixo na direção x*
- ω_y *Velocidade angular em torno do eixo na direção y*
- ω_z *Velocidade angular em torno do eixo na direção z*

RESUMO

O efeito da Globalização de Mercados trouxe novos desafios e necessidades de aumento da competitividade dos produtos, imprimindo uma busca incessante por novas soluções que tornem os sistemas de produção cada vez mais eficientes, inovadores, robustos e econômicos. Ganhar e manter vantagens competitivas passou a ser palavras de ordem entre empresas concorrentes, devendo para isso priorizar o cumprimento de datas de entrega, reduzir os tempos e custos de fabricação, aumentar a flexibilidade que permita incluir rapidamente inovações aos produtos, sem perda de qualidade e de padronização. Neste contexto desafiador, a automação dos processos de fabricação utilizando os adventos dos Sistemas Mecatrônicos tornou-se uma aliada estratégica. O robô industrial é, na atualidade, o Sistema Mecatrônico mais vendido e utilizado no mundo, sendo considerado estratégico por muitos países industrialmente desenvolvidos, por potencialmente proporcionar, em uma primeira análise: redução do custo de fabricação, aumento da produtividade, padronização da qualidade dos produtos, eliminação de postos de trabalhos humanos em tarefas insalubres, perigosas e repetitivas, e ao ser humano, destinar a execução de tarefas mais nobres. Contudo, também existem desafios que precisam ser vencidos, de modo a viabilizar a implementação de robôs industriais, como o planejamento e a sincronização, além da necessidade de programação que demanda tempo e mão-de-obra devidamente treinada e qualificada à eficiente utilização das peculiaridades de cada manipulador robótico. A presente dissertação tem por objetivo contribuir com uma solução que auxilie e facilite a programação de robôs industriais seriais, os quais possuem cadeia cinemática aberta, possibilitando modelar e simular em cenários de Realidade Virtual, além de obter a resolução cinemática baseada em helicoides desses robôs, por meio de um protótipo computacional intitulado CARPA (*Computer-Aided Serial Robots Programming, Modelling and Simulation Analysis*) em um ambiente amigável, simples e robusto, que possibilite dispensar a necessidade de um conhecimento aprofundado por parte do usuário/programador da Teoria dos Helicoides aplicada à robótica. Um estudo de caso, utilizando o modelo do FMS da Amatrol, contendo o robô Pegasus, é apresentado, onde o CARPA possibilita gerar automaticamente o código de programação na linguagem MCL, nativa deste robô.

Palavras-chave: Robô Industrial, Programação, Realidade Virtual, Helicoides, CARPA.

ABSTRACT

The Market Globalization effects brought new challenges and a need to increase the products' competitiveness, impelling an incessant search for new solutions that make the production systems more profitable, innovative, robust and economical. Winning and maintaining competitive advantages have became a slogan among competing companies. For this reason, they have to prioritize the fulfillment of delivery dates, reduce the manufacturing times and costs and increase flexibility, in order to quickly include innovations to the products, without loss of quality and standardization. In this challenging context, the manufacturing processes automation, using the advents of the Mechatronic Systems, has became a strategic ally. Currently, the industrial robot is the best-selling and most used Mechatronic System in the world, being considered strategic by many industrially developed countries. This is because it is able to potentially provide, in a first analysis: reduction of manufacturing costs, increase in productivity, standardization of products quality, elimination of human work stations in unhealthy, dangerous and repetitive tasks, and assignment of the nobler tasks execution to the human being. However, there are also challenges that need to be overcome in order to enable the implementation of industrial robots, such as the planning and synchronization, and the need for programming, which demands time and a properly trained and qualified workforce for the efficient use of the peculiarities of each robotic manipulator. The present dissertation aims to contribute to a solution that helps and facilitates the programming of serial industrial robots, which has open kinematic chain , allowing to model and simulate in Virtual Reality scenarios, besides obtaining the kinematic resolution based on the Screw Theory of these robots, through a computational prototype entitled CARPA (*Computer-Aided Serial Robots Programming, Modelling and Simulation Analysis*), in a friendly, simple and robust environment that makes it possible to dispense a deep user/programmer knowledge of the Screw Theory applied to robotics. A case study, using the Amatrol FMS model, containing the Pegasus robot, is presented, where CARPA allows the automatic generation of the programming code in the MCL language, native to this robot.

Keywords: Industrial Robot, Programming, Virtual Reality, Screw Theory, CARPA.

1 INTRODUÇÃO

Esta dissertação apresenta a modelagem cinemática e a simulação de robôs seriais e, mais especificamente, contribui para o planejamento e à programação de tarefas do robô Pegasus, tendo como base a Teoria dos Helicoides.

Esse capítulo inicia-se com a motivação dessa dissertação, na seção 1.1, em que se encontra uma contextualização, relacionando as vantagens da utilização de robôs industriais na automação dos processos e os desafios encontrados. Na seção 1.2, são apresentadas algumas dificuldades e limitações, referentes à programação de sistemas robóticos em processos de fabricação. Na seção 1.3, estão apresentados os objetivos gerais e específicos da dissertação. Por fim, a seção 1.4 apresenta a estrutura e a organização dos capítulos da dissertação.

1.1 MOTIVAÇÃO

Por se tratar de pesquisa realizada no Programa de Pós-Graduação em Engenharia Mecânica (PPGMEC) do Instituto Militar de Engenharia (IME), mais especificamente, no Laboratório de Robótica Industrial e de Defesa (IDR Lab), esta seção visa apresentar a motivação desta dissertação, estabelecendo um raciocínio que relate as vantagens estratégicas da Robótica Industrial, como Tecnologia Dual, para produzir bens de uso civil ou militar, contribuindo para o desenvolvimento da economia do país e da Defesa Nacional.

Cabe, portanto, uma análise dos principais instrumentos orientadores da Defesa no Brasil: a Política Nacional de Defesa (PND) e a Estratégia Nacional de Defesa (END). Enquanto a primeira fixa os objetivos da Defesa Nacional, a segunda estabelece como fazer o que foi estabelecido.

A PND (2012) define segurança como a condição que permite: preservar a soberania e a integridade territorial do País; promover os interesses nacionais, livre de pressões e ameaças; e garantir aos cidadãos o exercício dos direitos e deveres constitucionais. Segundo este documento de Estado, o conceito de segurança ampliou-se gradualmente e requer medidas de largo espectro para preservá-la, envolvendo além da defesa externa, políticas econômica, educacional e industrial. Mediante avaliação dos ambientes internacional, regional e entorno, emergem 11 (onze) Objetivos Nacionais de Defesa PND (2012), sendo que cabem ser transcritos, para fundamentar a motivação desta dissertação, especificamente o seguinte Objetivo: *IX. desenvolver*

a indústria nacional de defesa, orientada para a obtenção da autonomia em tecnologias indispensáveis.

Outro ponto que merece destaque para fundamentar a argumentação em foco encontra-se nas Orientações da PND, mais especificamente o seguinte ponto: *7.7. Os setores governamental, industrial e acadêmico, voltados à produção científica e tecnológica e para a inovação, devem contribuir para assegurar que o atendimento às necessidades de produtos de defesa seja apoiado em tecnologias sob domínio nacional obtidas mediante estímulo e fomento dos setores industrial e acadêmico. A capacitação da indústria nacional de defesa, incluindo o domínio de tecnologias de uso dual, é fundamental para alcançar o abastecimento de produtos de defesa.*

A END (2012) objetiva contribuir para o desenvolvimento das ações de médio e longo prazos, e organiza-se em torno de três eixos estruturantes, dos quais, o segundo é um dos pilares de argumentos que busca relacionar com a Robótica Industrial, pois o mesmo se refere à reorganização da Base Industrial de Defesa, para assegurar que o atendimento às necessidades de equipamento das Forças Armadas apoie-se em tecnologias sob domínio nacional, preferencialmente, as de emprego dual (militar e civil).

A Robótica Industrial é considerada como tecnologia estratégica por países de economia fortemente baseada no setor manufatureiro, e vem sendo alvo de altos e crescentes investimentos na automação dos processos de fabricação. Na atualidade, o robô industrial é o sistema mecatrônico mais vendido e utilizado no mundo e vem ganhando mais atenção a cada ano, por auferir robustez aos produtos fabricados em processos de manufatura automatizados, favorecendo o aumento produtividade, a padronização da qualidade e redução de custos de fabricação.

Diante da dependência tecnológica dos produtos presentes, tanto para emprego civil, quanto militar, uma base manufatureira automatizada, capaz de responder com rapidez e qualidade conforme a demanda, sendo de alta produtividade e confiabilidade ganha contornos cada vez mais acentuados, passando a ser considerada estratégica e dual, ou seja, tanto para garantir produtos competitivos na busca por maiores fatias de mercado, bem como, para produzir Produtos de Defesa, com qualidade e em quantidade, se preciso for, para garantir o êxito no campo de batalha.

1.2 FORMULAÇÃO DO PROBLEMA

Embora existam muitas vantagens no emprego do robô industrial na automação dos processos de fabricação, surgem também algumas questões que devem ser tratadas com atenção, a fim de viabilizar a sua integração aos processos, bem como, torná-los realmente aplicáveis e eficientes.

Esses sistemas mecatrônicos demandam pessoal qualificado e bem treinado, além de consumir um tempo relativamente alto para serem integrados.

Assim sendo, surge um problema ou oportunidade de melhoria, que demanda Pesquisa, Desenvolvimento e Inovação (PD&I) que possam **contribuir para a diminuição da complexidade e do tempo associados às fases de planejamento, programação, sincronização, validação e testes de programação** de robôs industriais seriais.

Essa dissertação, intitulada: "Modelagem e Simulação de Robôs Seriais baseadas em Helicoides", enquadra-se com a pesquisa vigente no Laboratório de Robótica Industrial e de Defesa do Instituto Militar de Engenharia (IME).

1.3 OBJETIVOS DA DISSERTAÇÃO

Com base na Motivação e diante das Problemas associados à implementação de robôs industriais, esta dissertação tem como objetivos **gerais** contribuir para:

1. reduzir a complexidade da implementação e programação de robôs industriais, de modo que possam auferir vantagens estratégicas às indústrias; e,
2. reduzir os tempos necessários ao treinamento de pessoal, e às fases de planejamento e programação de robôs industriais, de modo a difundir e aumentar a quantidade e qualidade do pessoal envolvido, de forma a conscientizá-los da importância do robô industrial para o aumento da robustez dos produtos fabricados.

A partir dos supracitados, a dissertação tem como objetivos **específicos**:

1. obter um sistema auxiliado por computador que facilite a modelagem, simulação e análise integradas de cenários contendo robôs industriais, da forma mais simples e intuitiva possível, sem perda de características de robustez e generalidade;
2. obter a resolução cinemática do robô modelado, utilizando a Teoria dos Helicoides, dispensando a necessidade de um conhecimento aprofundado por parte do usuário/programador desta Teoria aplicada à robótica; e,
3. reduzir o tempo de programação e de treinamento de iniciantes, possibilitando a programação assistida e a validação da obtenção do código de maneira automatizada para o caso do robô Pegasus.

Como resultado desta dissertação, é apresentado um protótipo computacional intitulado CARPA (*Computer-Aided Serial Robots Programming, Modelling and Simulation Analysis*). É mostrado um estudo de caso específico para o robô Pegasus, existente no IDR Lab, de forma a demonstrar a efetividade na diminuição do tempo de treinamento, utilizando o equipamento real, e na obtenção do respectivo código de maneira automatizada, reduzindo o tempo de programação *on-line*.

1.4 ORGANIZAÇÃO DA DISSERTAÇÃO

Esta seção descreve a organização do presente trabalho. A presente dissertação está estruturada da seguinte forma.

O Capítulo 1 se destina a introdução, em que se encontram estruturadas a motivação, a formulação do problema e como esta dissertação pretende contribuir em nível de solução com PDI, que estão explicitados nos objetivos gerais e específicos desta dissertação, bem como os resultados esperados que utiliza metodologia de implementação computacional orientada a objetos para a modelagem e simulação da cinemática baseada em helicoides de robôs seriais.

O Capítulo 2 apresenta uma revisão bibliográfica dos fundamentos da robótica industrial, apresentando desde suas vantagens e desvantagens em sua aplicação, passando por sua evolução histórica, até o panorama no mundo e sua situação atual no Brasil.

No Capítulo 3, são descritas e apresentadas as principais metodologias existentes (Convenção de Denavit-Hartenberg e Teoria dos Helicoides) para resolução cinemática, mostrando as etapas para determinação da cinemática direta e comparando-as, a fim de exaltar suas vantagens e desvantagens para implementação computacional e utilização no CARPA, para que seja de fácil modelagem pelo usuário.

No Capítulo 4, o sistema CARPA é apresentado, mostrando as fases de desenvolvimento, desde o início, passando por uma etapa intermediária, até a particularização capaz de gerar o código de forma automática para o robô Pegasus da Amatrol, e implementação para cenários genéricos. A etapa intermediária consiste na utilização do CARPA para um cenário específico, nas fases iniciais de treinamento de operação do referido robô, intitulado CAPRON (*Computer-Aided Pegasus Robot On-line Programming*). Ao longo do capítulo, são ressaltadas as funcionalidades da interface do protótipo, bem como a possibilidade de inserir robôs previamente modelados, e novos objetos.

Por fim, o Capítulo 5 apresenta um resumo das discussões abordadas ao longo da dissertação, as conclusões, vantagens e limitações encontradas, ressaltando as contribuições científica

publicadas e sugestões para trabalhos futuros.

Foram incluídos no Apêndice A, a Teoria dos Helicoides aplicada à Robótica, metodologia implementada computacionalmente, ressaltando o Método dos Deslocamentos dos Helicoides Sucessivos na determinação da cinemática direta de um manipulador serial (cadeia cinemática aberta) e o cálculo do Jacobiano baseado em helicoides.

2 ROBÓTICA INDUSTRIAL

Nesse capítulo, é feita uma revisão bibliográfica da robótica industrial, apresentando sua evolução histórica, suas vantagens e limitações em sua aplicação, bem como uma análise do panorama de sua situação, no Brasil e no Mundo.

2.1 SURGIMENTO DA ROBÓTICA

Antes de qualquer descrição do cenário atual, envolvendo os robôs industriais, é necessário voltar para o início do século XX, quando as palavras robô e robótica foram inventadas por escritores de ficção científica, conforme NOF (1999). A primeira foi criada por *Karel Capek*, em 1922, na peça de teatro *Rossum's Universal Robots (R.U.R.)*, sendo representada, na FIG. 2.1, uma de suas cenas. Já a segunda foi dada por *Isaac Asimov*, no começo dos anos de 1940, a fim de descrever a arte e a ciência, em que os roboticistas estão inseridos atualmente.



FIG. 2.1: Cena da peça de teatro R.U.R. (SICILIANO & KHATIB, 2008)

TSAI (1999) afirma que, desde a revolução industrial, tem havido uma crescente demanda para melhorar a qualidade dos produtos e reduzir os custos de fabricação. Antigamente, essa qualidade dependia, altamente, do artesão, na chamada Produção Manual. Em seguida, surge o conceito de Produção em Massa, em que a maioria dos processos era realizada por máquinas

especiais, o que diminuía drasticamente os custos de fabricação, tornando os produtos manufaturados, acessíveis à população em geral, principalmente, na época, os automóveis. Entretanto, como cada máquina era projetada para realizar uma determinada tarefa, toda a linha de produção tinha que ser parada e reequipada, sempre que um novo modelo era introduzido, o que significava altos investimentos. Essa era a chamada Automação Rígida. A partir disso, os robôs manipuladores começam a ser inseridos nas indústrias de fabricação para desempenhar certas tarefas de produção, tais como manuseio de material, soldagem a ponto, pintura por spray e montagem. Como os robôs podem ser reprogramados para diferentes tarefas, surge o tipo Automação Flexível.

Segundo CRAIG (2005), a história da automação industrial é caracterizada por períodos de rápida mudança nos métodos populares. Seja por causa, seja por efeito, tais períodos de transformação das técnicas de automação guardam relação direta com o desenvolvimento da economia mundial. O emprego de robôs industriais, os quais foram identificados como o único dispositivo mecatrônico nos anos de 1960, junto com o sistema CAD (*Computer-Aided Design*) e CAM (*Computer-Aided Manufacturing*), foi caracterizado como a última tendência na automação dos processos de manufatura.

Ainda segundo CRAIG (2005), a maior razão para o crescimento do uso dos robôs industriais foi o declínio do preço dos mesmos. Conforme a FIG. 2.2, nos Estados Unidos, além dessa diminuição, ao longo dos anos de 1990, houve também um aumento no custo das horas de trabalho do homem.

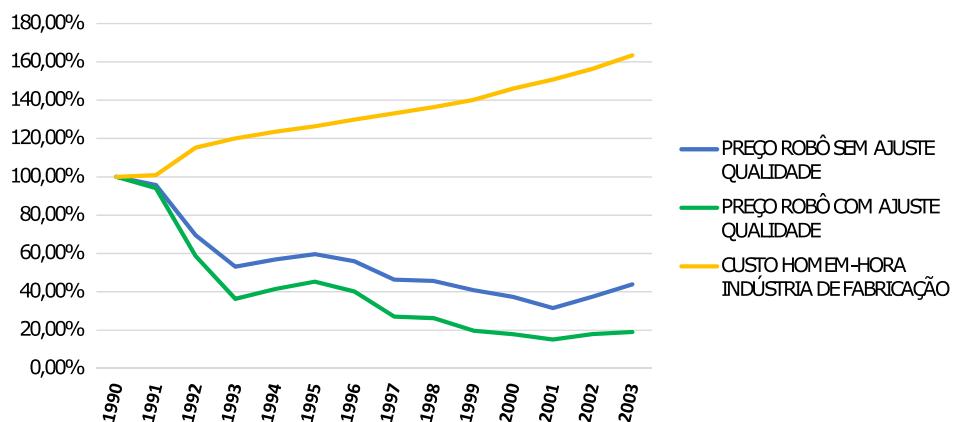


FIG. 2.2: Comparação entre custos do robô e do homem-hora na indústria de fabricação nos EUA (adaptado de UNECE (2004))

Além dos robôs estarem ficando mais baratos, os mesmos estão se tornando mais eficientes - mais rápidos, mais precisos, mais flexíveis. Segundo SICILIANO & KHATIB (2008), es-

sas melhorias proporcionaram um retorno mais rápido dos investimentos, particularmente, para produção de lotes pequenos e de curto prazo. Como pode ser verificado na FIG. 2.2, se esses ajustes de qualidade citados fossem levados em consideração, o preço do robô irá cair ainda mais rápido do que o preço pago pelos mesmos. Além disso, vão surgindo diversos fabricantes de robôs industriais, a fim de aperfeiçoar essa ferramenta tecnológica, aumentando a concorrência e, com o tempo, diminuindo ainda mais o preço do robô.

Assim, a necessidade de implementação rápida dos adventos tecnológicos nos processos de fabricação, na busca contínua em auferir vantagens competitivas aos produtos manufaturados, está se tornando cada vez mais estratégica por nações, que reconhecem na industrialização crescente, uma linha de ação relevante ao desenvolvimento de suas economias, como pode ser visto na seção 2.6. A utilização desses adventos confere vantagens competitivas estratégicas aos produtos fabricados, no sentido de diminuir a sua vulnerabilidade.

Segundo SLACK et al. (2002), a vulnerabilidade de um produto começa a despontar quando surge no mercado um produto similar de uma empresa concorrente, que desempenha a mesma função e possui um preço de mercado menor. Essa situação agrava-se bastante quando o produto concorrente possui ainda melhor qualidade e confiabilidade, menor tempo de entrega, melhor adequação às necessidades do cliente ou ainda, quando proporciona ao cliente, inovações a cada versão produzida. Assim, uma forma importante de melhorar o desempenho dos Sistemas de Fabricação é promover uma sinergia entre os objetivos de desempenho da indústria, como ilustrado na FIG. 2.3.

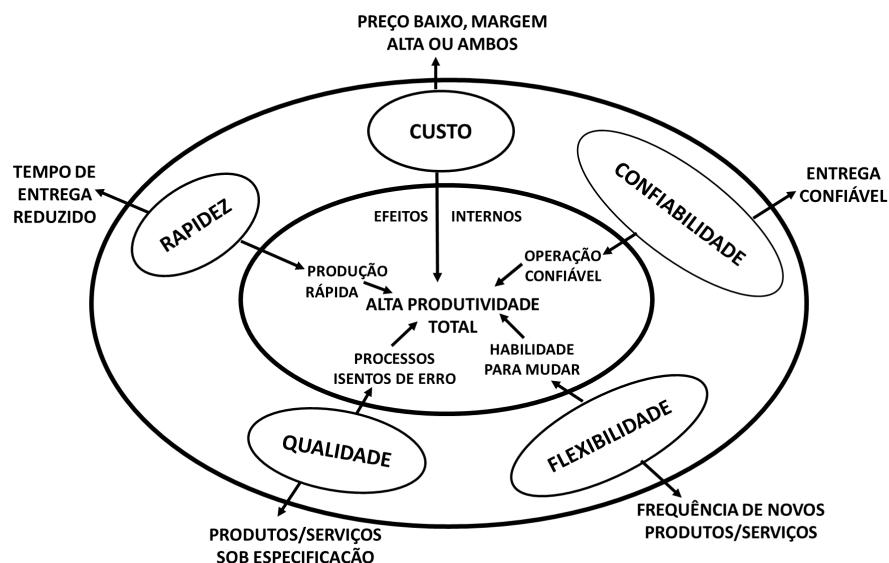


FIG. 2.3: Efeito dos objetivos de desempenho (adaptado de SLACK et al. (2002))

2.2 A ROBÓTICA

Antigamente, a robótica era conhecida como a ciência da ficção. No entanto, atualmente, os avanços tecnológicos tornaram-na uma das ferramentas mais avançadas, sendo muito utilizada nas indústrias de manufatura. Conforme CRAIG (2005), a entrada da robótica proporcionou uma melhora na produtividade, segurança, qualidade e eficiência dessas companhias.

Conforme SICILIANO & KHATIB (2008), a robótica pode ser definida como a ciência que estuda a conexão "inteligente" da percepção (sistema de medição) para a ação (sistema de atuação). Além disso, pode ser vista como o estudo das máquinas que podem substituir as tarefas executadas pelo homem, tanto no que tange aos aspectos físicos das atividades, quanto ao processo de tomada de decisão. A robótica é uma área de conhecimentos interdisciplinares e integrados, resultando da sinergia de diversas áreas, como matemática, a física, as engenharias mecânica e eletrônica e de computação, a tecnologia da informação e a automação e controle. Isso leva à necessidade de conjugar um conhecimento amplo dessas disciplinas, além de uma grande interação entre profissionais de formação distinta. A FIG. 2.4 mostra a sinergia entre as áreas do conhecimento e as de aplicação da robótica, mostrando a diversidade de possibilidades de aplicação em potencial e de contribuição em diferentes setores.

Por outro lado, sua aplicação real está, diretamente, relacionada ao tipo de ambiente, o qual pode ser: estruturado, parcialmente estruturado e não-estruturados. Quanto mais estruturado for um ambiente, mais conhecida é a sua descrição física ou geométrica, como por exemplo, o "chão de fábrica".

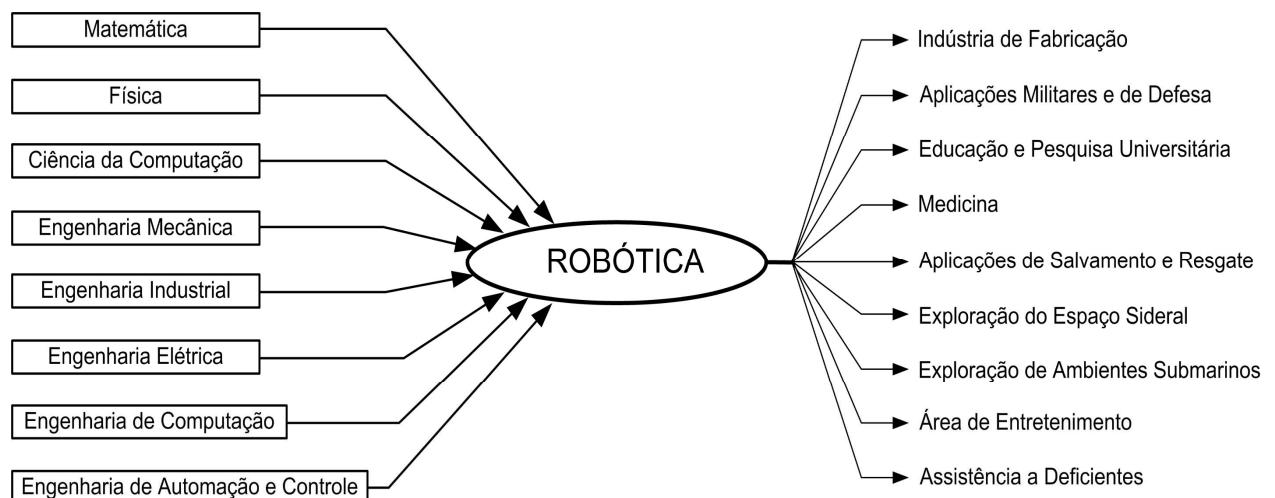


FIG. 2.4: Fatores associados à robótica (MACEDO & RIBEIRO, 2015)

Diante do alto potencial de aplicação, conforme SICILIANO & KHATIB (2008), genericamente, é possível dividir a robótica em:

- robótica avançada - estuda como os robôs atuam em ambientes pouco estruturados, ou seja, em locais hostis, como o espaço sideral ou em aplicação nuclear ou militar, onde é exigido maior grau de autonomia do robô; e,
- robótica industrial - estuda a atuação dos robôs em ambientes estruturados, onde o grau de complexidade e de necessidade de autonomia é relativamente menor, sendo uns dos motivos, pelos quais esta apresenta maior maturidade tecnológica do que a primeira.

2.3 ROBÔ INDUSTRIAL

Segundo a RIA (2015) - *Robot Institute of America*, um robô industrial é um manipulador multifuncional reprogramável, projetado para mover materiais, peças, ferramentas ou dispositivos especializados através de movimentos variáveis, programados para executar, adequadamente, uma variedade de tarefas.

Conforme SICILIANO & KHATIB (2008), o robô industrial é o Sistema Mecatrônico mais vendido e utilizado no mundo, sendo considerado estratégico para as indústrias por proporcionar, potencialmente, em uma primeira análise:

- redução do custo de fabricação;
- aumento da produtividade;
- padronização da qualidade dos produtos; e,
- eliminação de postos de trabalhos humanos em tarefas insalubres, perigosas e repetitivas.

Além destas vantagens, admite-se como consenso em países industrialmente desenvolvidos, as Leis da Aplicabilidade dos Robôs, enunciadas por NOF (1999), que tratam das atividades que os robôs devem substituir os seres humanos, tais como:

- **perigosas**, em que existam níveis excessivos de ruído, temperatura, pressão ou radiação; em locais com atmosferas tóxicas; onde haja radiação, e sempre que riscos físicos e outras periculosidades estejam presentes;

- **indesejáveis ou impossíveis**, onde o ser humano não tenha capacidade de atuar diretamente, tais como trabalhos repetitivos ou complexos e em escalas nano, micro ou macrométricas; e,
- **onde são mais produtivos e econômicos**, ou seja, embora sejam passíveis de desgaste mecânico, os robôs padronizam o nível de qualidade do produto, trabalham noite e dia (24 horas por dia, 7 dias na semana), sem serem afetados por desgastes emocionais e problemas familiares.

Devido à capacidade do robô industrial ser programável, é um componente típico dos Sistemas Automáticos Programáveis. Entretanto, também são passíveis de serem utilizados nos Sistemas Automáticos Rígidos e nos Flexíveis (SICILIANO et al., 2009). Assim, um mesmo robô pode realizar um conjunto de movimentos ou operações, completamente, diferentes, se a célula de trabalho for ajustada ou as tarefas desejadas forem alteradas. O programa de um robô deve ser capaz de possibilitar a obtenção de uma variedade de aplicações, devendo ser flexível, a fim de permitir uma sequência dinâmica de operações. A flexibilidade do robô é governada por uma extensa quantidade de tipos de movimentos e operações que podem estar programados dentro da unidade de controle e pela facilidade com que o programa pode ser introduzido e/ou modificado (SICILIANO & KHATIB, 2008).

Ao considerar um Sistema Automatizado Programável, vale ressaltar que os robôs manipuladores são a menor parte de um processo automatizado. O conjunto de equipamentos, o qual pode ser composto por um ou mais manipuladores, além de sistemas de transporte, de alimentação, peças e outros dispositivos elétricos, é conhecido como célula de trabalho. Esse conjunto deve estar interconectado na rede de comunicação da fábrica para que o controle central possa monitorá-lo. Dessa forma, é considerada uma variedade de máquinas interconectadas em um sistema celular de trabalho automatizado.

2.4 DESENVOLVIMENTO DA ROBÓTICA INDUSTRIAL

Conforme SICILIANO & KHATIB (2008), a invenção do robô industrial remonta ao ano de 1954, quando George Devol solicitou uma patente chamada "Transferência de Artigo Programado". Em 1956, após parceria com Joseph Engelberger, foi fundada a *Unimation Incs Technology* (fusão das palavras *universal* e *automation*), primeira empresa de robôs industriais, em *Connecticut* (USA). MURRAY et al. (1994) admite que, em 1961, foi implementado o primeiro robô industrial, o *Unimate*, em operação em uma fábrica da *General Motors*, para

manipular peças em um processo de fundição, conforme FIG. 2.5. Ainda conforme MURRAY et al. (1994), a inovação chave foi a programação da máquina, pois a mesma podia ser reequipada e reprogramada, com um baixo custo relativo, permitindo o robô executar uma variedade de tarefas. A construção mecânica do braço manipulador representou uma mudança do projeto, no qual utilizou uma cadeia cinemática aberta com vários graus de liberdade.



FIG. 2.5: Primeiro robô industrial, o *Unimate* (SICILIANO & KHATIB, 2008)

Segundo NOF (1999), em 1968, Engelberger assinou um contrato de licenciamento com a companhia japonesa *Kawasaki Heavy Industries (KHI)* para fabricar e vender os robôs da *Unimation* para o mercado asiático. Assim, até 1971, essa tecnologia se propagou, formando a primeira associação de robôs mundial, no Japão, conhecida como *Japan Industrial Robot Association (JIRA)*, a qual foi formada, inicialmente, por 46 empresas representantes, tendo influenciado a comunidade industrial do restante do mundo. Depois disso, o restante da indústria mundial começou a se despertar. Além da gigante japonesa, a KHI, algumas empresas começaram a se desenvolver e implementar a robótica em suas instalações, tais como: *General Motor*, *General Electric*, *Westinghouse*, *International Business Machines (IBM)* e *United Technologies*, nos Estados Unidos; *Siemens*, na Alemanha; *Renault*, na França; *FIAT*, na Itália. SICILIANO & KHATIB (2008) afirma que, nos anos posteriores, os manipuladores foram vendidos para manipulação de peças e para soldagem a ponto nas carrocerias de automóveis, tendo esta se tornado a principal aplicação para os robôs industriais, pois era uma tarefa exaustiva, insalubre e perigosa para o ser humano.

Conforme SICILIANO & KHATIB (2008), antigamente, os robôs industriais eram usados quase que exclusivamente no ramo automobilístico e na produção em massa. Atualmente,

através do contínuo aprimoramento desse advento tecnológico, a aplicação passou a ser feita em muitos outros ramos. Assim, muitos países se desenvolveram a ponto de não somente utilizá-los em suas indústrias, mas também de fabricá-los. A partir disso, surgem diversos fabricantes de robôs industriais, tais como: *ABB Robotics*, *Adept Technologies*, *COMAU Robotics*, *FANUC Robotics*, *General Electric* e *General Motors* que se associaram a *FANUC*, *KUKA Robotics*, *REIS Robotics*, *STAUBLI Robotics*, *YASKAWA MOTOMAN*. A seguir são melhor abordados os principais fabricantes, com seus respectivos países sede, além de sua atuação no Brasil.

- **ABB (Asea Brown Boveri)** - formada em 1988, através da fusão de duas empresas: a sueca ASEA e a suíça BBC (*Brown, Boveri Cie*), com sede em Zurique, Suíça. Em 2016, encontra-se presente em, aproximadamente, 100 países, empregando cerca de 135.000 funcionários. Dentre esses países, está o Brasil, onde possui uma planta fabril, localizada em Osasco, São Paulo, para a produção de soluções de automação, desde 1957 (ABB-BRASIL (2016)). Segundo consulta feita ao Departamento de Vendas da ABB do Brasil, em 2014, a empresa possuía 24% do *Market Share* no Brasil e, caso sejam desconsiderados os manipuladores cartesianos, os quais a ABB não fabrica, esse valor se aproxima de 29%;
- **FANUC (Fuji Automatic Numerical Control)** - criada em 1956, no Japão, quando o Dr. *Seiuemon Inaba* descobriu o conceito de Controle Numérico (CN). Atualmente, após 60 anos da fundação, foram instalados uma faixa de 400.000 robôs em todo o mundo (Brasil, América do Norte, Europa, Ásia e África), possuindo mais de 5.200 funcionários (FANUC (2016)). No Brasil, desde 1999, está localizada em São Paulo;
- **KUKA** - fundada em 1898, na Alemanha, o nome KUKA surgiu das letras iniciais da designação da empresa "*Keller und Knappich Augsburg*", de seus fundadores Hans Keller e Jakob Knappich. Hoje, a empresa opera com o nome de *KUKA Werkzeugbau Schwarzenberg GmbH (KWS)*, constituindo um grupo internacional com cerca de 3.000 funcionários. A empresa, atualmente, possui filiais espalhadas pelo mundo (África, América do Norte, América do Sul, Europa, Ásia e Oceania). No Brasil, está presente desde 1996, situada na cidade de São Paulo (KUKA, 2016); e,
- **YASKAWA Motoman** - fundada em 1989, no Japão. Essa empresa é umas das líderes de vendas de robôs industriais nas Américas, com mais de 300.000 unidades vendidas. Além disso, possui escritórios em 28 países e, aproximadamente, 14.000 funcionários no mundo

todo. No Brasil, a YASKAWA iniciou suas atividades em 1974 (MOTOMAN (2016)), se estabelecendo em 1999, por meio de uma subsidiária, a Motoman Robótica do Brasil, localizada em Diadema, São Paulo. Em consulta realizada, segundo o Diretor Geral da YASKAWA Elétrico do Brasil, a empresa possui um *Market Share* correspondente a cerca de 20% do Mercado Brasileiro em robótica industrial.

2.5 CLASSIFICAÇÃO DOS ROBÔS INDUSTRIAIS SERIAIS

Um robô industrial pode ser de dois tipos: serial, ou seja, possui cadeia cinemática aberta, existindo apenas uma sequência de corpos, conectando as duas extremidades da cadeia; ou paralelo, que apresenta cadeia cinemática fechada, em que a sequência de corpos forma um *loop*, como pode ser verificado na FIG. 2.6. No entanto, o estudo feito nesta dissertação se restringe a robôs industriais seriais.

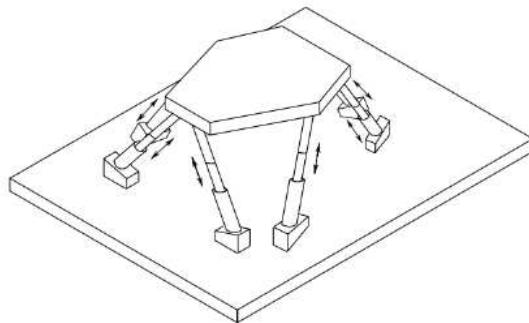


FIG. 2.6: Robô Paralelo (cadeia fechada) (SICILIANO et al., 2009)

Segundo SICILIANO et al. (2009), a estrutura mecânica de um robô manipulador consiste em uma sequência de corpos rígidos (multicorpos), interconectados por juntas. Um manipulador é caracterizado por apresentar um braço que garante mobilidade, um punho que confere destreza e um efetuador-final que realiza as tarefas exigidas. Os graus de mobilidade devem ser, corretamente, distribuídos ao longo da estrutura mecânica, a fim de fornecer os graus de liberdade necessários para a execução de uma determinada tarefa. No caso mais geral de uma tarefa, consistindo, de forma arbitrária, no posicionamento e na orientação de um objeto, em um espaço tridimensional, são necessários seis graus de liberdade - três para posicionamento do objeto e três para orientação do mesmo, em relação ao referencial de coordenadas (SICILIANO & KHATIB, 2008).

O espaço de trabalho representa a parte, no ambiente, que o efetuador do manipulador pode

alcançar. Seu formato e volume dependem da estrutura do manipulador, assim como, da presença de limites da junta mecânica. A tarefa do braço é posicionar o punho, o qual tem como finalidade, orientar o efetuador final. Nesse processo, pelo menos, três graus de liberdade são necessários no espaço de trabalho tridimensional. O tipo e a sequência dos graus de liberdade do braço do robô, começando da junta de base, permitem classificar os manipuladores seriais, conforme pode ser visto na FIG. 2.7 (SICILIANO & KHATIB, 2008):

- **Manipulador Cartesiano (PPP):** é constituído por três juntas prismáticas, das quais os eixos são mutuamente ortogonais, conforme FIG. 2.7 (a). Devido a sua geometria simples, cada grau de liberdade da junta corresponde ao grau de mobilidade, no espaço cartesiano, o que permite movimentos retos no espaço. Esse tipo de estrutura oferece boa rigidez mecânica. A garra possui precisão constante em todo espaço de trabalho, porém, pelo fato de todas as juntas serem prismáticas, o manipulador cartesiano possui baixa destreza. Por fim, a direção de aproximação, durante a manipulação dos objetos, é feita lateralmente. Por outro lado, se a aproximação de um objeto for feita pelo topo, o manipulador cartesiano pode ser construído através de uma estrutura *gantry*, como ilustrado na FIG. 2.7 (b), que proporciona uma área de trabalho de grande volume, além de manipular objetos pesados e de grandes dimensões. Esse tipo é empregado para o manuseio e montagem de materiais.
- **Manipulador Cilíndrico (RPP):** esse tipo difere do manipulador cartesiano, em relação à primeira junta prismática, que é substituída por uma de rotação. Se a tarefa é descrita em coordenadas cilíndricas, cada grau de mobilidade corresponde a um grau de liberdade. A estrutura cilíndrica oferece boa rigidez mecânica, porém a precisão do posicionamento do punho diminui, à medida que o curso horizontal aumenta. O espaço de trabalho é uma porção de um cilindro oco, como ilustrado na FIG. 2.7 (c). A junta prismática horizontal faz com que o punho de um manipulador cilíndrico seja adequado para acessar cavidades horizontais. São, principalmente, empregados para transporte de objetos, até mesmo, os de grandes dimensões;
- **Manipulador Esférico (RRP):** a diferença, em relação ao manipulador cilíndrico, se refere à troca da segunda junta prismática por uma de revolução, conforme apresentado na FIG. 2.7 (d). A rigidez mecânica é menor do que a dos manipuladores cartesiano e cilíndrico, e sua construção mecânica é mais complexa. A precisão do posicionamento do punho diminui, conforme o curso radial aumenta. O espaço de trabalho é uma porção

de uma esfera oca. Caso seja incluída um suporte de apoio do manipulador, será possível a manipulação de objetos no chão. No entanto, tem como principal função, o processo de usinagem;

- **Manipulador SCARA (RRP):** sua geometria especial é obtida, dispondendo duas juntas de revolução e uma prismática, de forma que todos os eixos de movimento sejam paralelos entre si. A sigla *SCARA* significa *Selective Compliance Assembly Robot Arm* e caracteriza os aspectos mecânicos de uma estrutura, oferecendo alta rigidez a cargas verticais e conformidade a cargas horizontais. A estrutura SCARA é própria para tarefas de montagem vertical. A precisão do posicionamento do punho diminui, conforme a distância, do punho à primeira junta, aumenta. A típica área de trabalho está ilustrada na FIG. 2.7 (e). Esse manipulador é adequado para a manipulação de pequenos objetos;
- **Manipulador Antropomórfico (RRR):** esse tipo é constituído por três juntas de rotação. O eixo de revolução da primeira junta é ortogonal aos eixos das outras duas juntas, os quais são paralelos, entre si. A estrutura antropomórfica é a que proporciona maior destreza, pelo fato de todas as juntas serem de revolução. O espaço de trabalho é, aproximadamente, uma porção de uma esfera oca, segundo ilustrado na FIG. 2.7 (f). Seu volume é grande se comparado ao manipulador esférico. Possui uma ampla faixa de aplicações industriais.

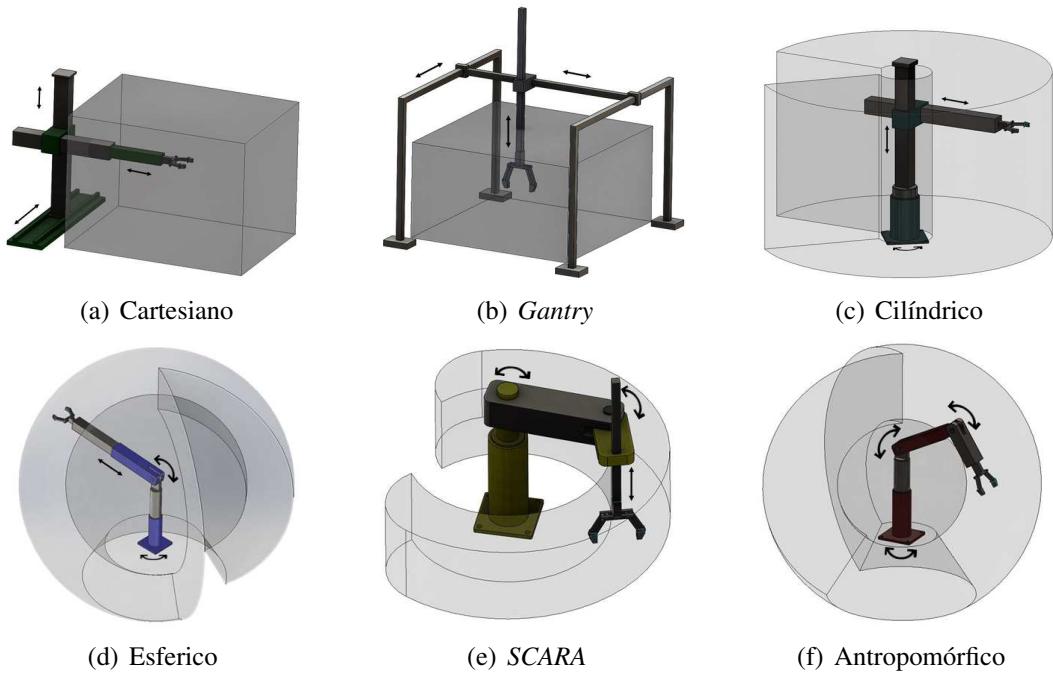


FIG. 2.7: Tipos de Manipuladores Robóticos e sua área de trabalho (Autoria Própria)

2.6 ESTATÍSTICAS DA ROBÓTICA INDUSTRIAL

De acordo com a *International Federation of Robotics* - IFR (2016), no ano de 2015, foi obtido, de longe, o maior volume de robôs industriais vendidos já registrado, quando as vendas de robôs industriais aumentaram em 15% em relação a 2014, alcançando um novo recorde 253.748 unidades, conforme mostrado na FIG. 2.8. Devido à constante tendência em relação à automação e à contínua melhoria técnica inovadora dos robôs industriais, desde 2010, a demanda por esses manipuladores tem acelerado consideravelmente.

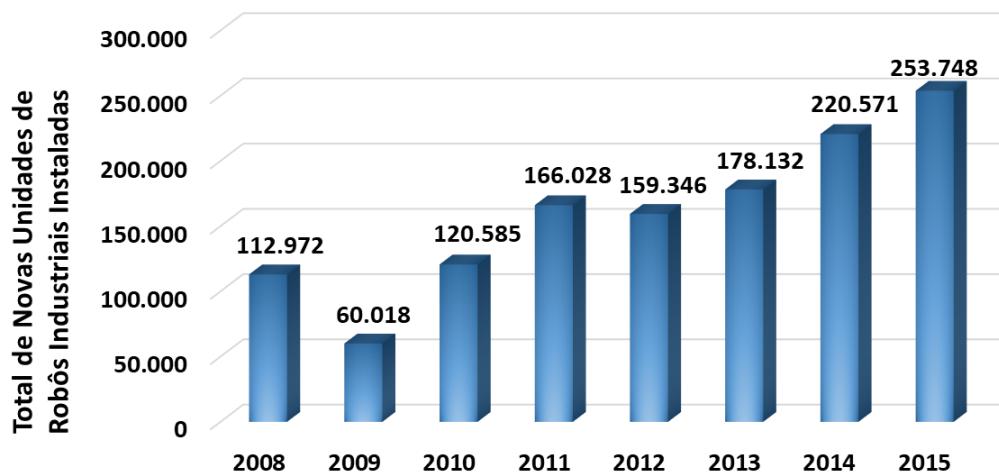


FIG. 2.8: Novas unidades instaladas de robôs industriais por ano (adaptado de IFR (2016))

2.6.1 PRINCIPAIS MERCADOS DE ROBÔS INDUSTRIAIS

É importante destacar a predominância de cinco países, como China, República da Coreia, Japão, Estados Unidos e Alemanha, sobre a maior parte do mercado, alcançando em torno de 75% do volume total das instalações, em 2015. Entretanto, a China foi o que mais se destacou com uma parcela de cerca de 27% do total instalado, em 2015, ou seja, 68.556 robôs industriais foram vendidos nesse país, 20% mais do que em 2014, ultrapassando sozinha o volume total de vendas para a Europa.

A República da Coreia, o segundo maior mercado de robô em 2015, aumentou em 55%, se comparado a 2014, instalando 38.285 novas unidades, aproximadamente, 15% do total, o maior nível alcançado por esse país.

No Japão, o terceiro maior em instalação em 2015, com cerca de 35.023 novos robôs industriais, em torno de 13% do total, representando um aumento de 20% em relação a 2014, alcançando

o maior nível de instalação nesse país, desde 2007, quando se atingiu 36.100 unidades.

No caso dos Estados Unidos, o quarto maior mercado, a instalação de robôs, em 2015, atingiu 27.504 unidades, cerca de 10% do total, 5% a mais do que em 2014. O propulsor desse crescimento, desde 2010, foi a contínua tendência para produção automatizada.

Por fim, o quinto país, dentre os principais mercados de instalação de novas unidades de robôs, é a Alemanha. Em 2015, houve um pequeno aumento das instalações de robôs, inferior a 1%, em relação a 2014, alcançando a marca de 20.105 unidades, o que representa por volta de 8% do todo, seu novo recorde de instalação registrado em um único ano.

2.6.2 PANORAMA DO MERCADO BRASILEIRO DE ROBÔS INDUSTRIAIS

Baseado nas estatísticas da IFR (2015), é possível fazer uma análise que conduz a uma reflexão: enquanto, no mundo, foram instaladas 253.748 novas unidades de robôs industriais, somente 1.407 foram instaladas no Brasil, o que representa cerca de 0,55% do total. A FIG. 2.9 mostra a baixa participação de instalação de novas unidades comparada com o total, no período de 2010 a 2015, indicando uma maior necessidade de atenção por parte dos tomadores de decisão para automatizar os processos de fabricação, a fim de auferir vantagens estratégicas ao parque fabril brasileiro.

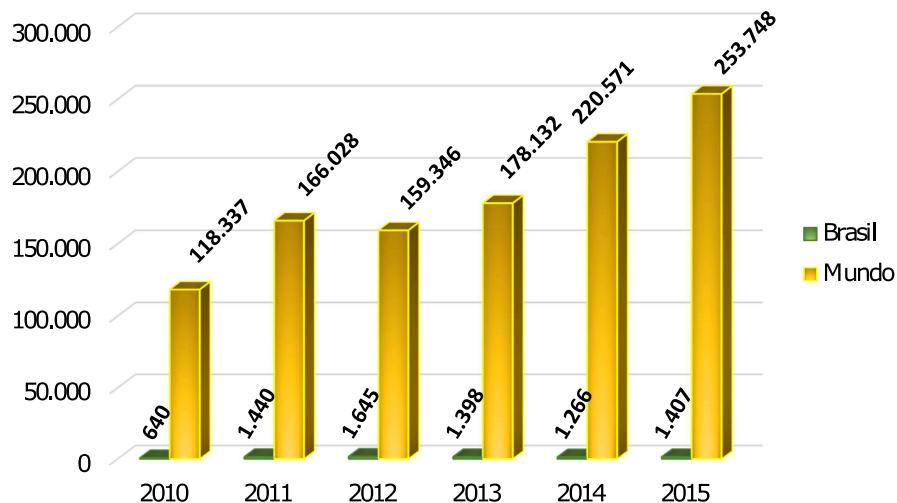


FIG. 2.9: Comparativo entre Brasil e Mundo de novas unidades de robôs industriais instaladas entre 2010 e 2015 (adaptado de IFR (2016))

Finalmente, cabe ressaltar que em 2015, os principais impulsionadores desse crescimento, foram as indústrias automotiva, eletroeletrônica e de metal.

2.6.3 CUSTOS DO SISTEMA ROBÓTICO

Segundo a IFR (2016), o valor das vendas aumentou em 9% em 2015, alcançando um novo pico de USD 11.1 bilhões. É importante ressaltar que o valor mencionado não engloba o custo de software, periféricos e de engenharia, que se incluídos, resultariam em um valor de mercado dos sistemas robóticos cerca de três vezes maior, em torno de USD 35 bilhões, conforme FIG. 2.10.

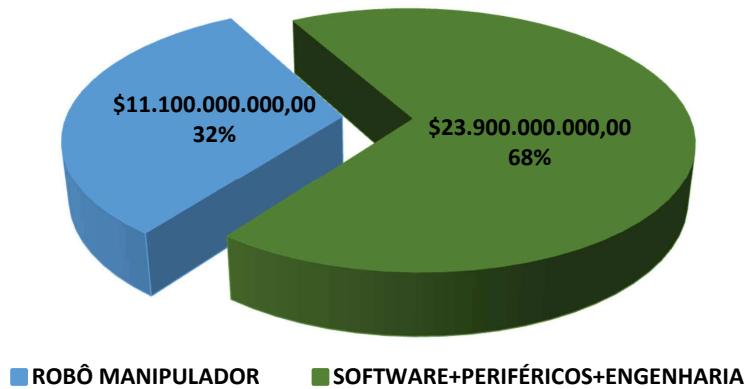


FIG. 2.10: Comparaçāo entre os valores do robô e do sistema de *software* em 2015 (adaptado de IFR (2016))

2.6.4 EXPECTATIVAS PARA OS ROBÔS INDUSTRIAIS

A integração de robôs aos processos de fabricação é uma atividade desafiadora e complexa diante da quantidade de escolhas possíveis, quando se analisa o número de fabricantes de robôs no mercado mundial, os atributos de cada robô, seus custos, face às exigências do processo a ser automatizado.

Conforme o Departamento de Estatística da IFR (2016), a implementação de robôs industriais tem se acentuado ao longo dos anos. Cada vez mais, os robôs irão substituir os seres humanos em tarefas perigosas, tediosas ou em trabalhos em que não sejam possíveis ou seguros para os seres humanos realizarem.

Entre 2016 e 2019, é estimado pela IFR (2016), que cerca de 1.400.000 novos robôs industriais sejam instalados nas fábricas do mundo todo, sendo esperada a concretização de diversas realizações, tais como:

- avanço da colaboração homem-robô;

- modernização contínua das instalações de produção devido ao aumento da competição global;
- expansão das capacidades de produção devido ao crescimento de mercados consumidores;
- aumento contínuo da qualidade dos produtos;
- Pequenas e Médias Empresas (PME) usarão, cada vez mais, os robôs industriais;
- a China permanecerá como o principal impulsionador do crescimento de robôs industriais instalados, expandindo seus domínios. Até 2019, quase 40% do fornecimento global será instalado na China;
- crescimento contínuo de instalações de robôs na maioria dos mercados asiáticos, como República da Coréia, Japão e Taiwan; e,
- crescimento acelerado das vendas de robôs na Europa Central e Oriental.

Além disso, o conceito de Indústria 4.0 está emergindo, a Quarta Revolução Industrial, que, conforme a IFR (2016), visa unir a realidade de fábrica com a realidade virtual, desempenhando um papel cada vez mais importante na produção global. De acordo com VENTURELLI (2014), entendendo o conceito dessa revolução como a evolução dos sistemas produtivos industriais, existem alguns benefícios que a mesma poderá trazer, como:

- redução de custos;
- economia de energia;
- aumento da segurança;
- conservação industrial;
- redução de erros;
- fim do desperdício;
- transparência nos negócios; e,
- inovação contínua.

VENTURELLI (2014) afirma que a tecnologia base responsável por esse conceito é o *Internet of Things* (Internet das Coisas) - IoT, em que a ideia é a conexão lógica de todos os dispositivos e meios, relacionados ao ambiente produtivo em questão.

Essa evolução exigirá um novo profissional, com formação multidisciplinar, capacidade de adaptação, senso de urgência, além do já necessário e cada vez mais preciso bom relacionamento interpessoal. De acordo com Osvaldo Maia, gerente de inovação e tecnologia do SENAI de São Paulo, "o avanço da tecnologia afetará todo mundo, do chão de fábrica ao alto escalão" (SIEMENS (2015)).

2.7 TIPOS DE PROGRAMAÇÃO DE ROBÔS INDUSTRIAIS

Segundo NOF (1999), a programação de um robô industrial pode ser alcançada basicamente por dois métodos: *on-line* e *off-line*. Assim, ao longo dessa seção, são apresentadas as definições e as vantagens e desvantagens de cada tipo.

2.7.1 PROGRAMAÇÃO *ON-LINE* DE ROBÔS INDUSTRIAIS

A programação *on-line* encontra-se no processo produtivo e envolve uma célula de trabalho, conforme mostrado FIG. 2.11.



FIG. 2.11: Célula de Trabalho durante a Programação *On-line* no IDR Lab (Autoria própria)

NOF (1999) afirma que a programação *on-line* é também conhecida como a técnica de mostrar e ensinar, ou seja, nela, ocorre a movimentação manual do robô para cada posição requerida. As informações do movimento e demais dados necessários podem ser armazenados pelo controlador do robô, quando este é guiado, através da trajetória desejada, durante o processo de

aprendizado. A edição do programa pode ser utilizada para adicionar dados, corrigir um ponto incorreto ou modificar algum ponto, devido a troca da tarefa desenvolvida. Durante o processo de aprendizado, o operador deve repetir vários segmentos do programa, a fim de verificar, visualmente, o movimento ou as operações.

A programação por aprendizado depende dos algoritmos de controle usados para mover o robô. Estes são caracterizados por três tipos: ponto a ponto, contínuo e trajetória controlada. No primeiro tipo, o movimento do ponto de origem ao ponto de destino não considera a trajetória percorrida pelo manipulador. Geralmente, cada eixo percorre a sua taxa máxima ou limitada, até alcançar a posição desejada. Embora todos os eixos comecem a se mover, simultaneamente, não significa que os mesmos irão completar os movimentos ao mesmo tempo. O segundo envolve a repetição de aproximação dos pontos de posição de cada junta que são gravados pela unidade de controle, em uma base de tempo constante, digitalizando os codificadores (*encoders*) dos eixos, durante o movimento do robô. O terceiro implica no controle das coordenadas do movimento de todas as juntas para garantir a trajetória desejada entre dois pontos programados. Nesse último método, cada eixo se move suavemente e proporcionalmente, a fim de garantir um movimento de trajetória controlado (NOF, 1999).

A principal vantagem da programação *on-line* é que o robô é programado de acordo com a posição atual do equipamento e dispositivos. Porém, existem outras, tais como:

- capacidade de se verificar o resultado imediatamente;
- o controle do robô limita a sua velocidade, tornando a programação mais segura;
- baixo custo inicial; e,
- é de fácil aprendizado e não necessita de conhecimento técnico específico.

Por outro lado, a desvantagem mais marcante é que a produção fica suspensa durante a programação por um tempo relativamente alto. Entretanto, existem outras: pouca flexibilidade e a qualidade da programação depende, diretamente, da capacidade técnica do programador.

Os robôs podem ser programados por um conjunto de métodos denominado de Métodos por Aprendizagem, os quais são realizados de duas formas: direta e indireta. Na primeira forma, o robô é guiado, fisicamente, pelo operador do ponto de origem até o de destino. Já na segunda, esse deslocamento até o ponto final é feito, por meio de uma caixa de comando, conhecida como *Teach Pendant (TP)* que controla todos os atuadores das juntas, conforme a FIG. 2.12, a qual exemplifica o TP do robô Pegasus.



FIG. 2.12: *Teach Pendant (TP)* do robô Pegasus instalado no IDR Lab (Autoria própria)

Uma tarefa com um grau de complexidade elevado por ter uma diferença muito grande entre o tempo de programação e o de execução da mesma. Segundo PAN et al. (2012), a programação *on-line* de uma grande linha automotiva de soldagem a arco pode levar mais de 8 meses, enquanto que o tempo de execução do processo de soldagem é de apenas 16 horas. Para o caso citado, o tempo de programação é 360 vezes maior que o tempo de execução. No entanto, em uma atividade de *pick-and-place*, um caso relativamente mais simples, também pode haver uma discrepância acentuada. Essa tarefa, realizada no Laboratório de Robótica de Defesa e Industrial (IDR Lab - sigla em inglês) do IME, foi publicada, na revista do XXXV Congresso da SODEBRAS, em forma de artigo. Conforme abordado por SANTOS et al. (2016), em uma atividade desse tipo, o tempo de programação foi de, aproximadamente, 32 horas (1955 minutos) e o tempo total necessário à realização da tarefa pelo robô Pegasus foi de 10 minutos, ou seja, o tempo de programação foi 192 vezes maior do que o tempo de execução da tarefa. Esses dados podem ser observados na FIG. 2.13, além de um comparação com demais grupos, que serviram como ponto de partida desse artigo.

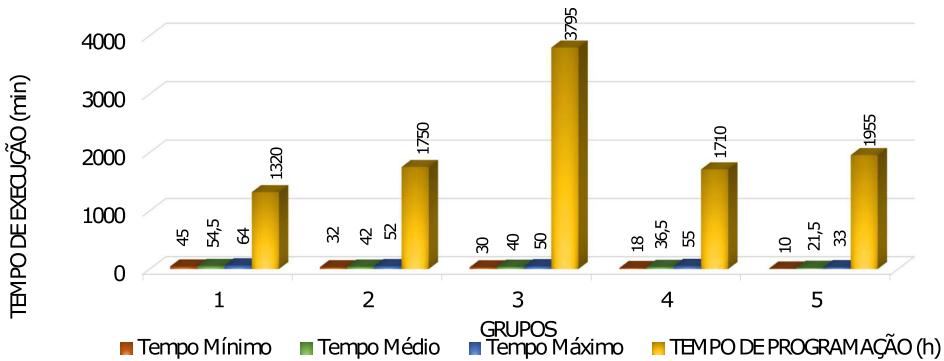


FIG. 2.13: Relação entre os Tempos de Programação e o de Execução da Tarefa (Autoria Própria)

Outra atividade relativamente mais simples do que a de soldagem, a de montagem, também realizada no IDR Lab do IME, foi publicada, na revista do XXXVI Congresso da SODEBRAS, em forma de artigo. Conforme abordado por SANTOS et al. (2017b), em uma atividade desse tipo, o tempo de programação foi de, aproximadamente, 27 horas (1620 minutos) e o tempo total necessário à realização da tarefa pelo robô Pegasus foi de 12 minutos, ou seja, o tempo de programação foi 135 vezes maior do que o tempo de execução da tarefa, o que representa ainda uma diferença muito grande entre esses tempos.

2.7.1.1 ETAPAS DE DESENVOLVIMENTO DA PROGRAMAÇÃO *ON-LINE*

Segundo a AMATROL (2007a), recomenda-se que se utilize um procedimento que facilite o processo de programação do robô, o qual pode ser dividido nas seguintes etapas:

1. Conhecer plenamente a tarefa desejada;
2. Diagrama Pictórico - um desenho da célula de trabalho do robô, em escala ou pelo menos, nas proporções corretas;
3. Sequência Geral de Operações - uma descrição geral dos passos a serem executados pelo robô para realizar a tarefa;
4. Lista de Entradas e Saídas - deve-se incluir o nome, função, tipo de dispositivos de *input* e *output* (I/O) necessários;
5. Sequência Detalhada de Operações - listar, passo-a-passo, toda ação ou processo de comunicação que o robô deverá executar para concluir a tarefa;
6. Lista de pontos - apresenta os pontos necessários para se alcançar a sequência detalhada de operações, bem como a função de cada um;

Finalmente, é possível escrever o programa, iniciando com um fluxograma para facilitar. Utilizam-se um ou mais comandos de programa para cada passo da sequência detalhada de operações. Essa abordagem, além de facilitar a escrita dos programas, proporciona uma documentação detalhada da aplicação para outras pessoas que tenham participado do desenvolvimento do programa.

A partir desse ponto, é necessário fazer a Validação e Teste para garantir que o processo esteja conforme desejado.

Segundo NOF (1999), a programação de um robô por aprendizado pode consumir um tempo que, frequentemente, cresce, em grande proporção, com o aumento da complexidade da tarefa. Assim, como o robô permanece fora de produção, sua utilidade pode ser reduzida substancialmente e a viabilidade econômica da programação *on-line* passa a ser questionada.

2.7.2 PROGRAMAÇÃO OFF-LINE DE ROBÔS INDUSTRIAIS

Na programação *off-line*, o computador e os modelos substituem o sistema robótico, ou seja, os programas para os robôs são desenvolvidos, parcialmente ou completamente, sem a necessidade do uso do robô, incluindo a geração de pontos de coordenadas, dados funcionais e ciclo lógico, todavia a Validação e Teste ainda se fazem necessários. Existem, atualmente, consideráveis atividades nos métodos de programação *off-line*, as quais são muito empregadas na indústria de manufatura. Em alguns casos, os programas podem ser criados pela reutilização de dados existentes de programas CAD, tornando a programação relativamente mais rápida e efetiva. Esses programas são verificados em simulação, e qualquer erro é, possível de ser corrigido em uma primeira análise.

Para as aplicações de robô serem possíveis em campos de produção de pequeno e médio lotes, onde os tempos de programação podem ser substanciais, torna-se mais acentuado o emprego da programação *off-line*. Devido ao aumento da complexidade das aplicações dos robôs, particularmente, em relação aos trabalhos de montagem, as vantagens associadas à programação *off-line* tornam-se mais atrativas, tais como:

- a utilização de um programa computacional desenvolvido em uma linguagem de alto nível facilita a programação do robô para atividades mais complexas;
- verificação do programa através de simulação e visualização. *Softwares* de simulação podem ser empregados para provar que as tarefas estão livres de colisões;
- retirada do programador de ambientes potencialmente hostis e redução no tempo da sua exposição ao risco de comportamentos estranhos do robô, já que a maior parte da programação é feita longe do robô;
- a programação é feita sem precisar parar, parcialmente ou completamente, a produção, pois não ocupa totalmente o equipamento de produção. O robô pode ainda estar na produção, enquanto a próxima tarefa está sendo programada, o que capacita utilizar a flexibilidade do robô com mais eficiência;

- Um sistema único de programação *off-line* pode ser usado para simular uma variedade de robôs, sem a necessidade de conhecer as particularidades de cada controlador do robô.

Entretanto, a programação *off-line* necessita de um modelo matemático do robô e de seu ambiente de trabalho, a fim de simular o comportamento do robô no mundo real. Assim, sua implementação esbarra em algumas desvantagens, tais como:

- dificuldades no desenvolvimento de um sistema generalizado de programação, o qual seja independente dos robôs e das aplicações;
- demanda investimentos em um sistema de programação *off-line*, tanto tecnológico, quanto de mão de obra especializada.

Para se obter um sistema de programação *off-line* bem-sucedido, existem alguns requerimentos que são considerados importantes, tais como:

- um modelo tridimensional do ambiente de trabalho, isto é, dados da descrição geométrica dos componentes e suas relações no espaço de trabalho;
- conhecimento do processo ou da tarefa a ser programada;
- conhecimento da geometria, da cinemática (incluindo os limites das juntas e suas velocidades) e da dinâmica do robô;
- verificação e validação dos programas produzidos, a fim de conferir as violações dos limites das juntas e detectar possíveis colisões no espaço de trabalho;
- interface apropriada, a qual permita comunicação entre os dados de controle de sistemas *off-line* e os controladores dos robôs; e,
- Interface Homem-Máquina (IHM) amigável, a qual permita a transferência da habilidade do programador para o sistema *off-line*. Para isso, é importante um ambiente de programação intuitivo e robusto.

2.8 SOFTWARES ASSOCIADOS À MODELAGEM, SIMULAÇÃO E PROGRAMAÇÃO DE ROBÔS

Com a evolução dos robôs industriais e sua implementação em atividades mais complexas, tornou-se necessário o emprego de ferramentas que auxiliassem o programador/usuário na exe-

cução dessas tarefas. Conforme YANG et al. (2016), o ambiente de simulação robótico possibilita o usuário projetar, controlar e testar o robô de forma segura, o que faz com que o sistema computacional tenha um papel importante no campo da robótica. A partir disso, podem ser exemplificadas algumas plataformas de simulação que foram úteis no desenvolvimento do CARPA.

2.8.1 SOFTWARE DE SIMULAÇÃO DE ROBÓTICA DO PEGASUS (RSS)

O software de simulação do Robô Pegasus da Amatrol, o *Pegasus Robotics Simulator Learning System (RSS)*, possui uma interface, como pode ser visualizado na FIG. 2.14. Segundo a AMATROL (2007b), o RSS permite programar o robô Pegasus simulado e testar seu programa antes de carregá-lo no robô real. Entretanto, pôde-se verificar que o RSS disponível pelo fabricante, para realizar a programação *off-line* do robô, possui algumas restrições. Dentre elas, a principal é que o *software* apresenta apenas seis configurações fixas de área de trabalho, isto é, não podem ser modificadas. Além disso, nenhuma destas representa exatamente a área de trabalho real do robô presente no IDR Lab, uma vez que não é considerada a junta prismática, sobre a qual a base do robô é montada. Dessa forma, esse conjunto de fatores impossibilita o uso do RSS no sistema robótico disponível.

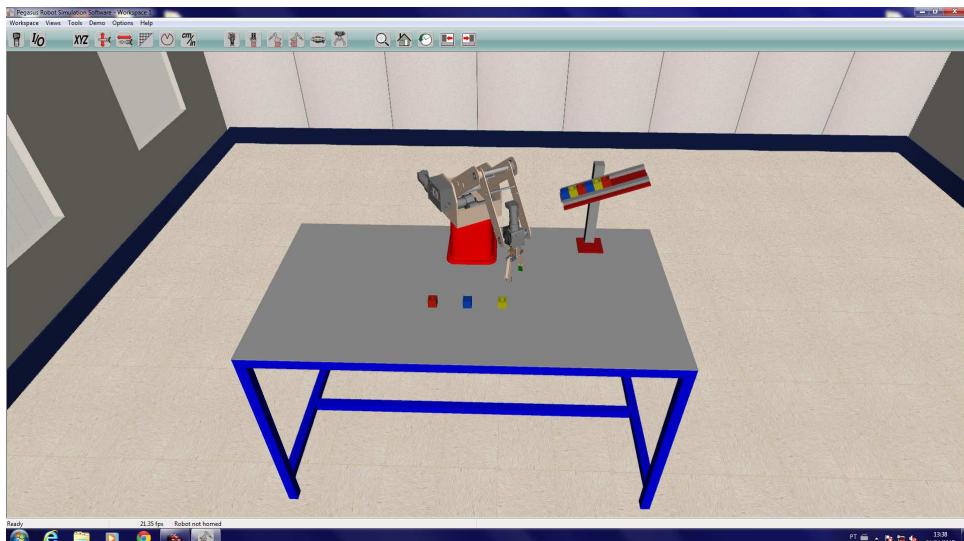


FIG. 2.14: Tela Inicial do RSS do Pegasus (Autoria Própria)

Por outro lado, existem algumas funcionalidades importantes do RSS que, se usado em um dos seis cenários modelados, traz grandes benefícios, tais como:

- possibilidade de verificar a posição da ponta da ferramenta (garra), facilitando seu posicionamento simétrico com relação ao objeto. Diferentemente, na programação *on-line*, o operador necessita posicionar o próprio rosto próximo ao local, onde será o contato entre o objeto e a garra, podendo ter um risco e ser pouco prático em algumas posições na mesa de trabalho; e,
- possibilidade de exibir a projeção vertical da localização do atuador e o eixo Z do sistema de referência da ferramenta. Esses recursos combinados auxiliam na centralização da ferramenta no objeto, de maneira correta.

2.8.2 VIRTUAL ROBOT EXPERIMENTATION PLATFORM (V-REP)

De acordo com FREES et al. (2013), o V-REP é um simulador de robô multipropósito, com um ambiente de desenvolvimento integrado, conforme FIG. 2.15. No V-REP, existe uma grande quantidade de modelos de robôs, sensores e atuadores. O software está disponível nos Sistemas Operacionais Windows, Mac e Linux.

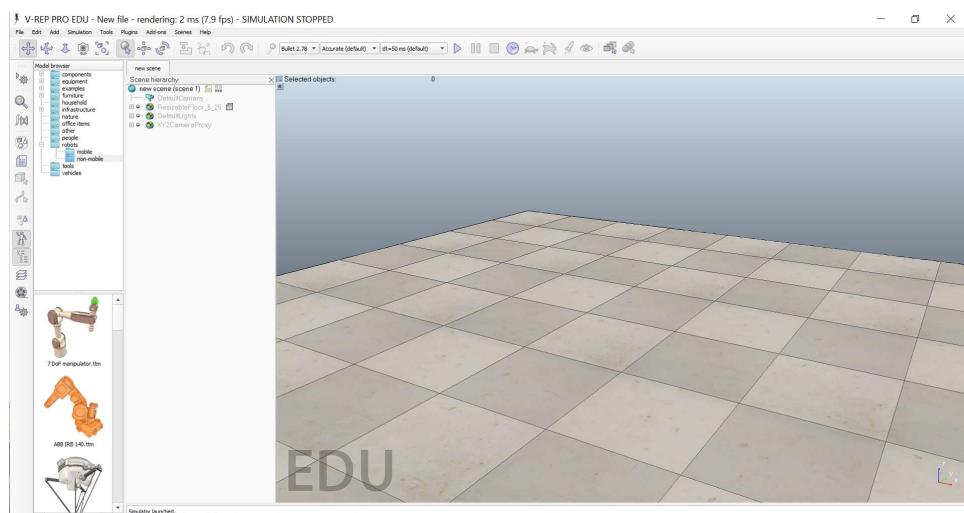


FIG. 2.15: Tela Inicial do V-REP (Autoria Própria)

O V-REP foi lançado ao público conforme a COPPELIA-ROBOTICS (2016a), em Zermatt, na Suíça, em março de 2010, tendo sido desenvolvido durante muitos anos, possuindo mais de cinquenta versões, com auxílio de diversas pessoas (pesquisadores) e empresas (que atuam nas áreas de automação industrial, pesquisas em robótica, desenvolvimento de softwares e/ou simulação), tais como: ABB (disponibilizando os modelos e projetos em CAD); Laboratórios

da KUKA GmbH; HiBot; Blue Work Force; Cubictek; Scientif Formosa INC (SFI); Siwat; Reflexxes GmbH.

Entretanto, a companhia *Coppelia Robotics* passou a desenvolvê-lo apenas em abril de 2014, quando a mesma foi fundada por Marc Andreas Freese, na Suíça, sendo definida como Pequena e Média Empresa, atuando no campo de simulação de robótica.

O V-REP é dividido em três versões, as quais podem ser citadas com as seguintes particularidades (COPPELIA-ROBOTICS, 2016b):

- **V-REP PRO EDU** - pode ser utilizado por estudantes, professores, escolas e universidades de forma gratuita e sem limitações (totalmente funcional), não necessitando de registro. Entretanto, não pode ser empregado em aplicações comerciais, instituições de pesquisa e organizações sem fim lucrativos;
- **V-REP PRO** - pode ser empregado sem limitações (totalmente funcional) em empresas, instituições de pesquisa e organizações sem fim lucrativos, tendo um custo/valor associado. Além disso, pode ser usado em aplicações comerciais; e,
- **V-REP PLAYER** - pode ser usado por qualquer pessoa de forma gratuita para qualquer aplicação. No entanto, com capacidade limitada de edição, em que a opção "salvar" está desabilitada.

Tanto o V-REP, quanto o RSS, foram estudados para entender como são tratadas as trocas de informações com o usuário, de modo a levantar pontos fortes e oportunidades de melhoria para fundamentar a forma de interação do CARPA com o usuário.

As FIG. 2.16 à 2.19 apresentam uma forma de inserir um robô no V-REP. O primeiro passo dessa tarefa é clicar em "*File*", em seguida, em "*Import*", e depois em "*Mesh...*" (FIG. 2.16). O segundo passo é escolher o arquivo do corpo do robô e clicar em "Abrir" (FIG. 2.17). Na (FIG. 2.18), é possível ver o primeiro corpo inserido. Analogamente, esses passos podem ser feitos para os demais (FIG. 2.19). Essa atividade foi realizada na fase de pesquisa desse sistema que permitiu conhecer uma de suas maneiras de inclusão de objetos, além de poder operar nele.

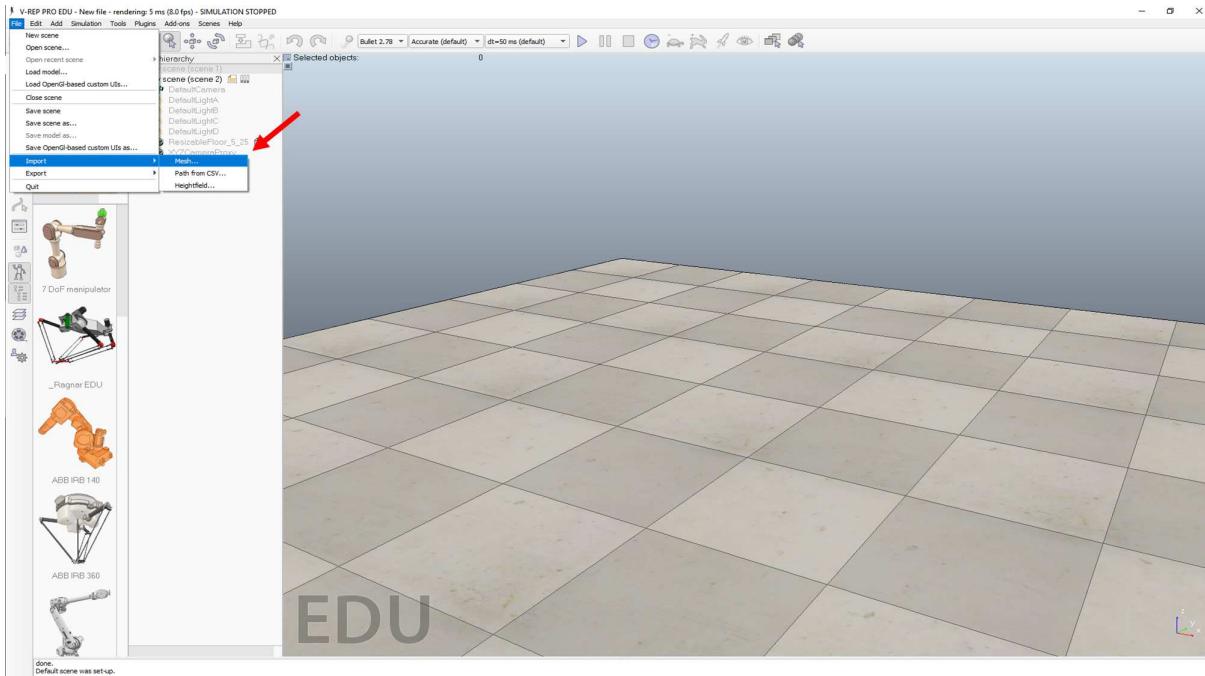


FIG. 2.16: V-REP - inserir robô (Autoria Própria)

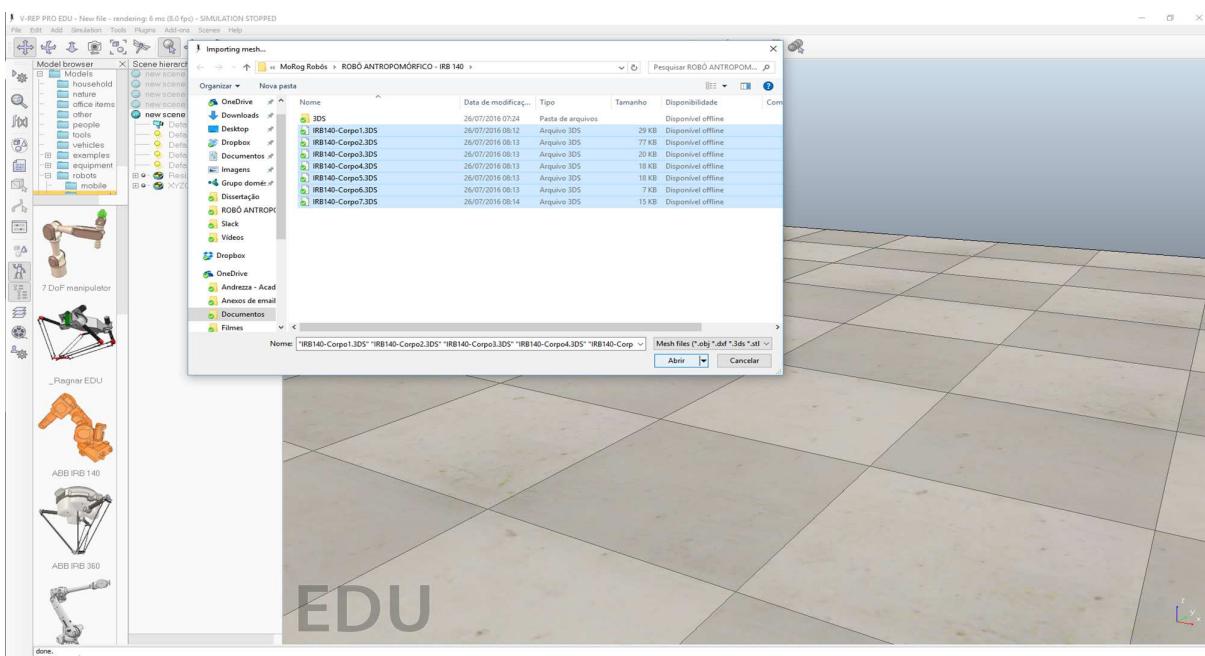


FIG. 2.17: V-REP - inserir cada corpo (Autoria Própria)

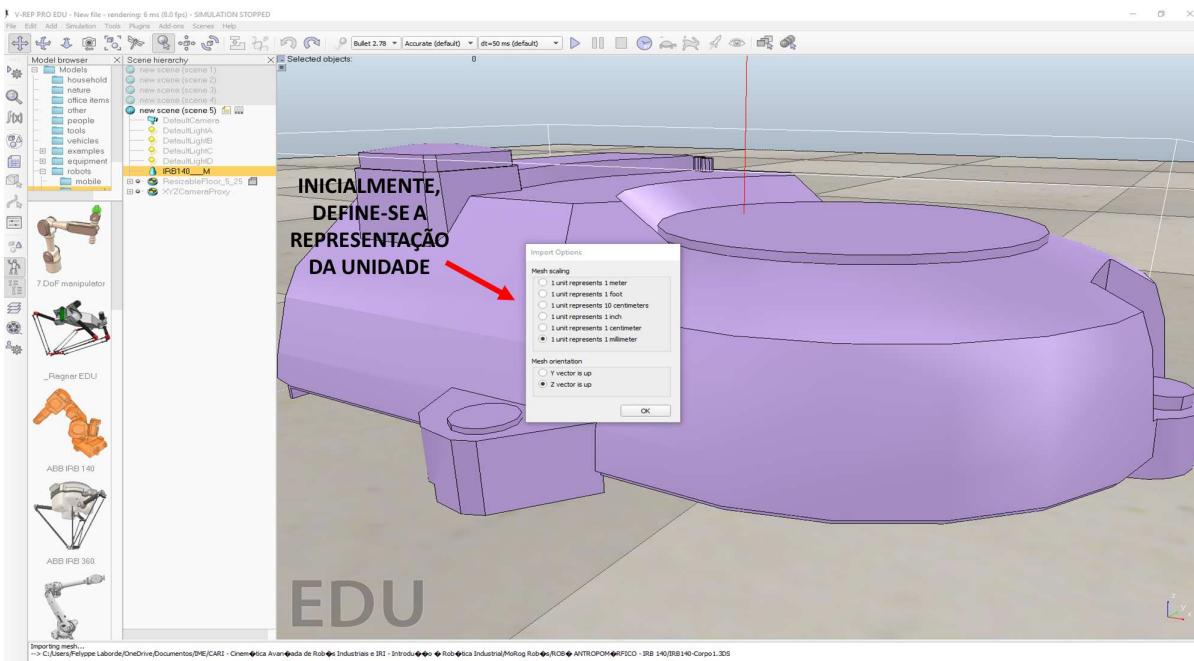


FIG. 2.18: V-REP - base inserida (Autoria Própria)

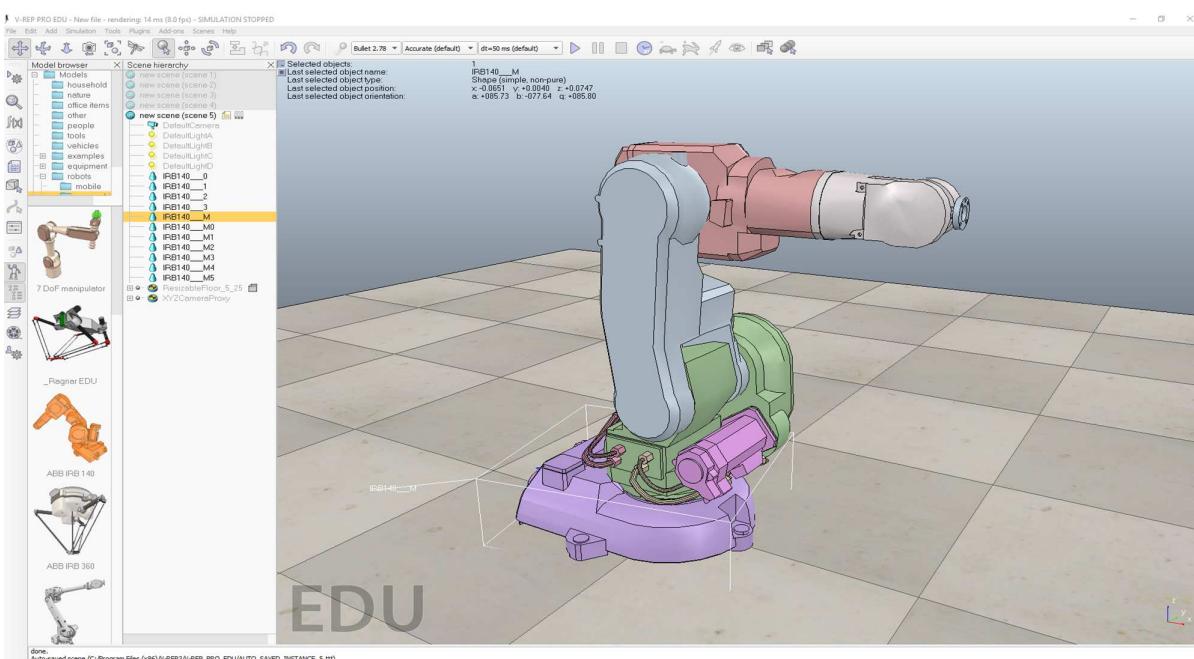


FIG. 2.19: V-REP - robô inserido (Autoria Própria)

3 METODOLOGIAS EXISTENTES

Nesse capítulo, é feita uma apresentação e comparação de duas das principais metodologias encontradas na literatura, a Teoria dos Helicoides e a Convenção de Denavit-Hartenberg, para a modelagem cinemática de robôs. Para isso, utiliza-se o robô Pegasus como estudo de caso, e por fim, é explicado porque foi adotada a Teoria dos Helicoides para a resolução da cinemática direta, mais especificamente, o Método dos Deslocamentos dos Helicoides Sucessivos.

3.1 CINEMÁTICA DE ROBÔS SERIAIS

Conforme SICILIANO et al. (2009), a modelagem de robôs industriais seriais é de extrema importância para a resolução da cadeia cinemática, a qual é formada por multicorpos (corpos rígidos), conectados por juntas (geralmente, de translação ou de revolução) que compõem o robô. A cadeia inicia-se na base do robô e termina no efetuador-final. A análise cinemática consiste na descrição matemática da capacidade que a variação da posição de cada junta possui em alterar a posição e orientação do efetuador final, em relação a um referencial fixo. Esse procedimento é conhecido como Cinemática Direta.

Existe uma outra forma de analisar a cinemática de um robô serial, a qual chama-se Cinemática Inversa, em que as posições de cada uma de suas juntas são determinadas em função da posição e orientação desejadas para o efetuador-final.

Além destas, existe a Cinemática Diferencial, a qual determina e relaciona as velocidades generalizadas do efetuador-final, em função das velocidades das juntas (linear ou angular). Essa relação das velocidades é descrita pelo Jacobiano (EQ. 7.106).

Para a resolução cinemática de robôs seriais, existem alguns métodos que podem ser empregados. Nas seções 3.2, 3.3, 3.4 e 3.5, é apresentado o procedimento para resolução da cinemática direta, por meio do principal método, a Convenção de Denavit-Hartenberg, e de uma metodologia alternativa que utiliza a Teoria dos Helicoides, mais especificamente, o Método dos Deslocamentos dos Helicoides Sucessivos.

3.2 CONVENÇÃO DE DENAVIT-HARTENBERG

A Convenção de Denavit-Hartenberg (D-H), segundo ROCHA et al. (2011), é um dos métodos mais utilizados pela comunidade científica na área de robótica, para resolução cinemática de robôs manipuladores. Com o objetivo de analisar o potencial de implementação, são mostrados procedimentos de forma genérica e um estudo de caso, utilizando o robô Pegasus da Amatrol. A Convenção de D-H, tem como objetivo apresentar uma série de regras para se definir a posição relativa e a orientação entre dois elos consecutivos. Desta forma, é possível estabelecer um procedimento (SICILIANO et al., 2009), a fim de operacionalizar a determinação da cinemática direta por esse método, com n elos e $n - 1$ juntas, conforme a seguir:

1. Identificar as juntas, elos e eixos:

- Numerar as juntas de 1 até n , conforme FIG. 3.1;
- Numerar os elos de zero (base) até n , ou seja, existem $n + 1$ elos, como pode ser visto na FIG. 3.2;
- Escolher o eixo z_i sobre o eixo da junta $i + 1$, para $i = 0, \dots, n - 1$, segundo a FIG. 3.3 .

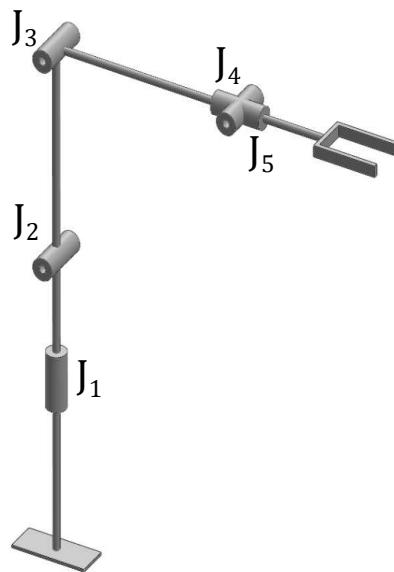


FIG. 3.1: Identificação dos elos do robô Pegasus, conforme Denavit-Hartenberg (Autoria Própria)

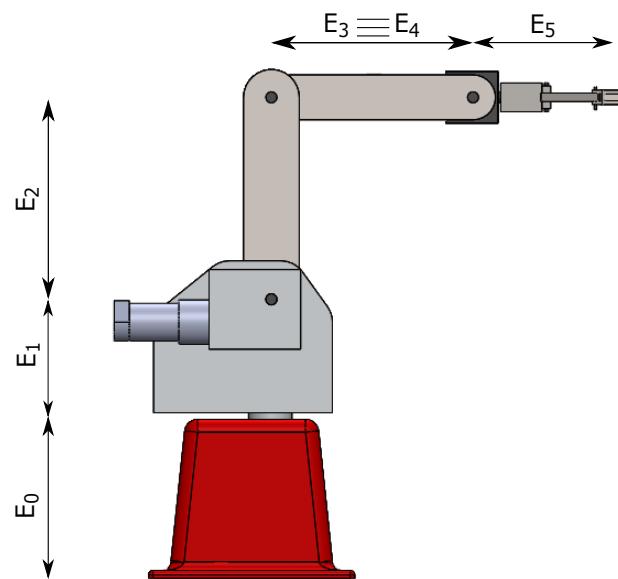


FIG. 3.2: Identificação das juntas do robô Pegasus, conforme Denavit-Hartenberg (Autoria Própria)

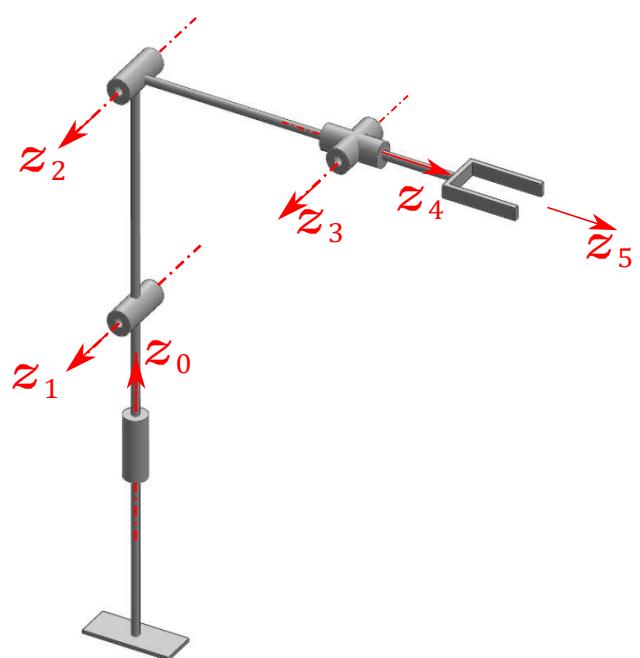


FIG. 3.3: Identificação do eixo z_i do robô Pegasus, conforme Denavit-Hartenberg (Autoria Própria)

2. Determinar o sistema de coordenadas absoluto 0 (zero), conforme pode ser verificado na FIG. 3.4, utilizando o esquemático do robô Pegasus:

- O_0 é definido sobre a direção do eixo z_0 , na menor distância possível entre z_0 e z_1 . Quando $z_0 \perp z_1$, O_0 estará na interseção dos dois eixos. Quando não for possível utilizar essa regra, coincidir O_0 com a interseção do eixo z_0 com a base do robô;
- Escolher o sentido de z_0 sempre da base para a junta 1 e x_0 e y_0 seguindo a regra da mão-direita.

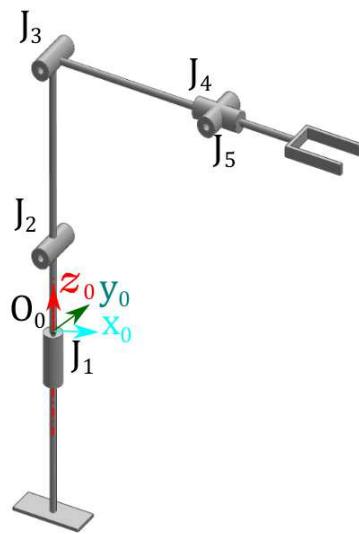


FIG. 3.4: Determinação do sistema de coordenadas absoluto por D-H (Autoria Própria)

Os passos de 3 a 9 a seguir devem ser repetidos para $i = 1, \dots, n-1$.

3. Determinar o ponto O'_i (sistema linha) na interseção do eixo z_{i-1} com a normal comum dos eixos z_{i-1} e z_i , observando se:

- z_i for colinear a $z_{(i-1)}$ e a junta $i + 1$ for de translação, definir O'_i no limite da junta $i+1$, ou seja na sua posição de *home*;

4. Determinar as origens dos sistemas coordenados O_i na interseção do eixo z_i com a normal comum aos eixos z_{i-1} e z_i , observando se:

- $z_{i-1} \parallel z_i$, e a junta i for de rotação, escolher O_i , tal que $d_i = 0$ (ou seja, coordenada de O_i' sobre o eixo z_{i-1} ;

- $z_{i-1} \parallel z_i$, e a junta i for de translação, escolher O_i , em uma posição de limite da junta (na posição de zero da junta).

Os passos 3 e 4 estão representados na FIG. 3.5.

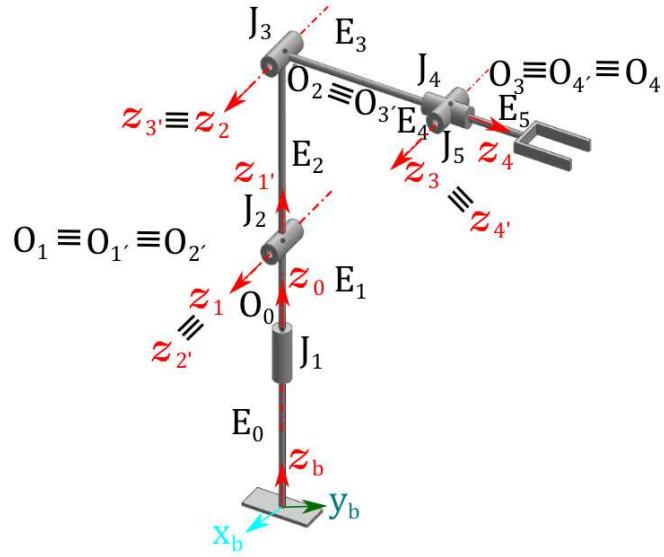


FIG. 3.5: Determinação dos sistemas O_i e O'_i por D-H (Autoria Própria)

5. Escolher o eixo x_i sobre a normal comum aos eixos z_{i-1} e z_i ;

6. Escolher o eixo y_i , obedecendo a regra da mão direita;

A FIG. 3.6 representa os passos 5 e 6.

7. Definir $z_{i'}$, sabendo que este é colinear (ou seja, os eixos devem estar sobre uma mesma reta) a z_{i-1} ;

8. Definir $x_{i'}$ sobre a normal comum aos eixos z_{i-1} e z_i com sentido da junta i para a junta $i + 1$, sabendo que $x_{i'}$ é colinear a x_i ;

9. Definir $y_{i'}$ de acordo com a regra da mão direita;

Os passos 7, 8 e 9 são representados pela FIG. 3.7, utilizando o esquemático do robô Pegasus.

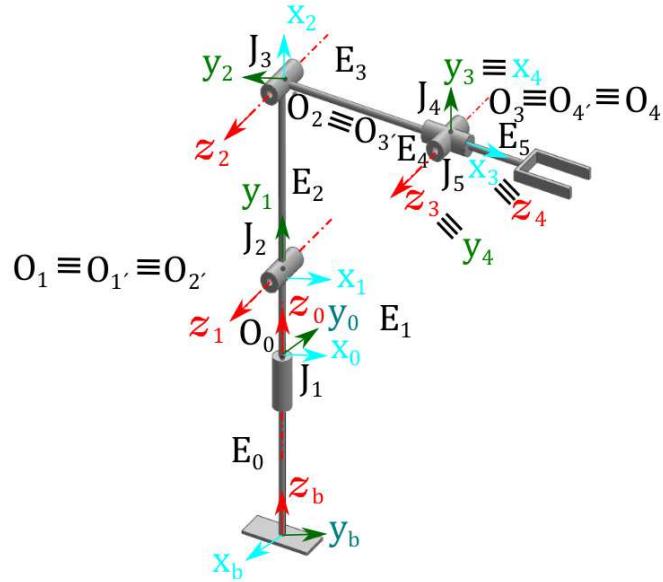


FIG. 3.6: Determinação dos eixos x_i e y_i por D-H (Autoria Própria)

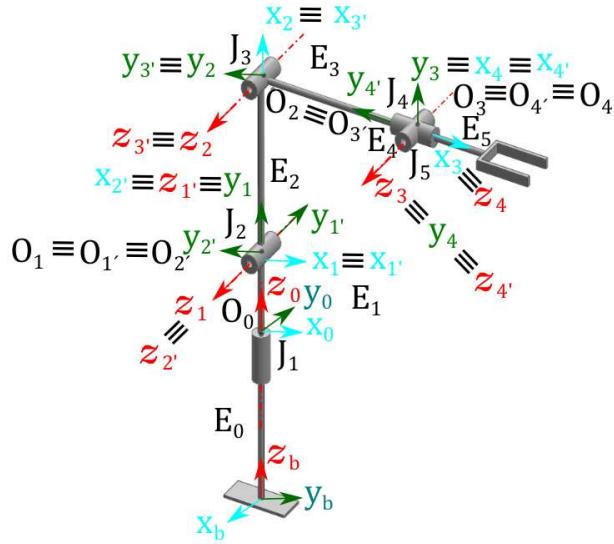


FIG. 3.7: Determinação dos eixos z_i' , x_i' e y_i' por D-H (Autoria Própria)

10. Definir o sistema n da seguinte forma:

(a) Determinar o eixo z_n , observando se a junta $n - 1$ for:

- de rotação, alinhar z_n com z_{n-1} ;

- prismática, escolher z_n de forma arbitrária.
- (b) Determinar x_n , conforme o passo 5;
- (c) Determinar y_n , conforme o passo 6.
- (d) Determinar $z_{n'}$, conforme o passo 7;
- (e) Determinar $x_{n'}$, conforme o passo 8;
- (f) Determinar $y_{n'}$, conforme o passo 9.

Assim, a representação do passo 10 encontra-se na FIG. 3.8.

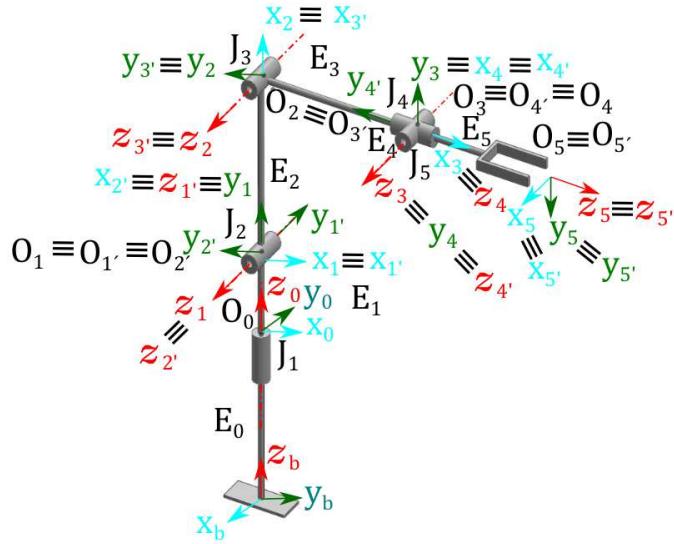


FIG. 3.8: Determinação do sistema n e n' por D-H (Autoria Própria)

11. Definir os quatro parâmetros de Denavit-Hartenberg construindo uma tabela para $i = 1, \dots, n$, onde:

- a_i - distância entre O_i e $O_{i'}$;
- d_i - coordenada de $O_{i'}$ sobre o eixo z_{i-1} ;
- α_i - ângulo entre os eixos z_{i-1} e z_i em torno do eixo x_i ; e,
- ϑ_i - ângulo entre os eixos x_{i-1} e x_i em torno do eixo z_{i-1} .

É importante mencionar que dentre os 4 parâmetros, 2 são sempre constantes (a_i e α_i). Em relação aos outros dois, um varia de acordo com o tipo de junta que conecta o elo $i - 1$ ao elo i . Isto é:

- Se a junta i for de rotação, a variável é ϑ_i , tendo um parâmetro fixo, calculado pela regra supracitada, adicional de θ_i que representa a rotação da junta i ;
- Se a junta i for prismática, a variável é d_i , tendo um valor fixo, calculado pela regra supracitada, com um adicional de t_i que representa a translação de junta i .

Assim, para o robô Pegasus, os parâmetros podem ser definidos, conforme TAB. 3.1.

TAB. 3.1: Parâmetros de Denavit-Hartenberg do robô Pegasus

JUNTAS	a_i	d_i	α_i	ϑ_i
1	0	5,25	90°	ϑ_1
2	9	0	0	$\vartheta_2 + 90^\circ$
3	9	0	0	$\vartheta_3 - 90^\circ$
4	0	0	90°	$\vartheta_4 + 90^\circ$
5	0	6	0	$\vartheta_5 + 90^\circ$

12. Determinar as Matrizes de Transformação Homogêneas (MTH's), para $i = 1, \dots, n$:

$$\mathbf{T}_i^{(i-1)}(q_i) = \begin{bmatrix} c\vartheta_i & -s\vartheta_i c\alpha_i & s\vartheta_i s\alpha_i & a_i c\vartheta_i \\ s\vartheta_i & c\vartheta_i c\alpha_i & -c\vartheta_i s\alpha_i & a_i s\vartheta_i \\ 0 & s\alpha_i & c\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.1)$$

13. Determinar a composição das Matrizes de Transformação Homogêneas:

$$\mathbf{T}_n^0(q) = \mathbf{T}_1^0(q_1) \mathbf{T}_2^1(q_2) \dots \mathbf{T}_i^{(i-1)}(q_i) \dots \mathbf{T}_n^{(n-1)}(q_n) \quad (3.2)$$

Para o caso de estudo desse artigo:

$$\mathbf{T}_5^0(q) = \mathbf{T}_1^0(q_1) \mathbf{T}_2^1(q_2) \mathbf{T}_3^2(q_3) \mathbf{T}_4^3(q_4) \mathbf{T}_5^4(q_5) \quad (3.3)$$

14. Determinar as MTH's T_0^b entre o referencial absoluto e a base escolhida, e T_e^n entre o efetuador-final e a última junta:

$$T_e^b(q) = T_0^b(q) T_n^0(q) T_e^n(q) \quad (3.4)$$

Para o caso específico:

$$T_e^b(q) = T_0^b(q) T_5^0(q) T_e^5(q) \quad (3.5)$$

Assim, de forma genérica, ou seja, sem atribuir valores a ϑ , utilizando os parâmetros de D-H da TAB. 3.1 para o robô Pegasus:

$$T_0^b = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 7,25 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.6) \quad T_1^0 = \begin{bmatrix} c_1 & 0 & s_1 & 0 \\ s_1 & 0 & -c_1 & 0 \\ 0 & 1 & 0 & 5,25 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.7)$$

$$T_2^1 = \begin{bmatrix} c_2 & -s_2 & 0 & 9c_2 \\ s_2 & c_2 & 0 & 9s_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.8) \quad T_3^2 = \begin{bmatrix} c_3 & -s_3 & 0 & 9c_3 \\ s_3 & c_3 & 0 & 9s_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.9)$$

$$T_4^3 = \begin{bmatrix} c_4 & 0 & s_4 & 0 \\ s_4 & 0 & -c_4 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.10) \quad T_5^4 = \begin{bmatrix} c_5 & -s_5 & 0 & 0 \\ s_5 & c_5 & 0 & 0 \\ 0 & 0 & 1 & 6 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.11)$$

É importante ressaltar que é feita uma rotação de 90° em Z , apresentada na matriz T_0^b , a fim de seguir o mesmo referencial do fabricante do robô. A matriz T_e^5 é uma identidade, não tendo assim necessidade de inseri-la no cálculo.

3.3 ROBÓTICA BASEADA EM HELICOIDES

Pela Teoria dos Helicoides, também é possível seguir um procedimento de atividades necessárias para a resolução da cinemática direta, apresentando as Matrizes de Transformação Homogênea de forma genérica, e, como estudo de caso, os cálculos e resultados são obtidos utilizando o robô Pegasus da Amatrol, como enumerado a seguir. Vale ressaltar que todos os detalhes acerca desse método estão apresentados no Apêndice 7.1.

- Identificar os elos e juntas, conforme pode ser visto na FIG. 3.9.

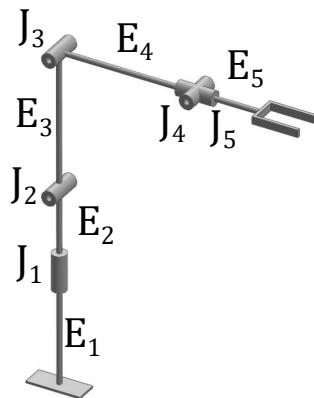


FIG. 3.9: Identificação de elos e juntas do Robô Pegasus (Autoria Própria)

- Encontrar os parâmetros S e S_0 de cada junta, ou seja, o eixo em torno do qual as juntas rotacionam ou ao longo do qual as mesmas transladam, e as distâncias em relação ao referencial absoluto, respectivamente. Analisando as FIG. 3.10 e 3.11, é possível verificar, respectivamente, cada um destes.

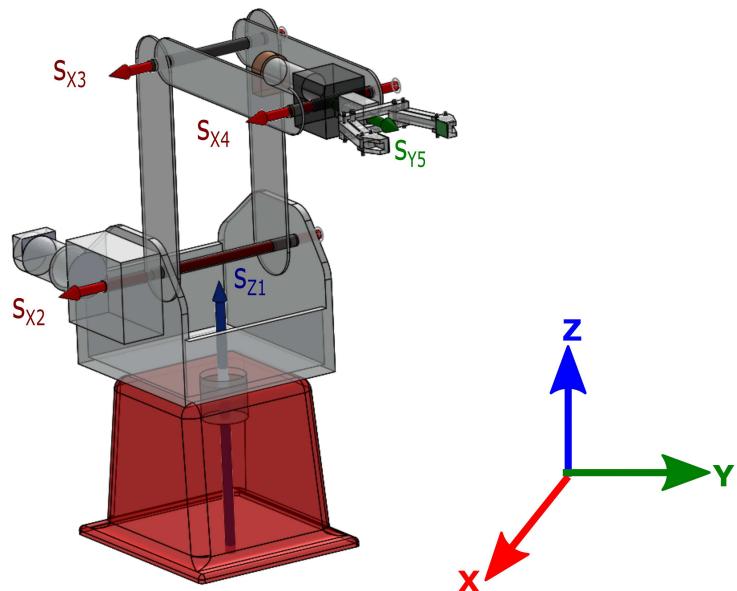


FIG. 3.10: Representação do parâmetro s no Robô Pegasus (Autoria Própria)

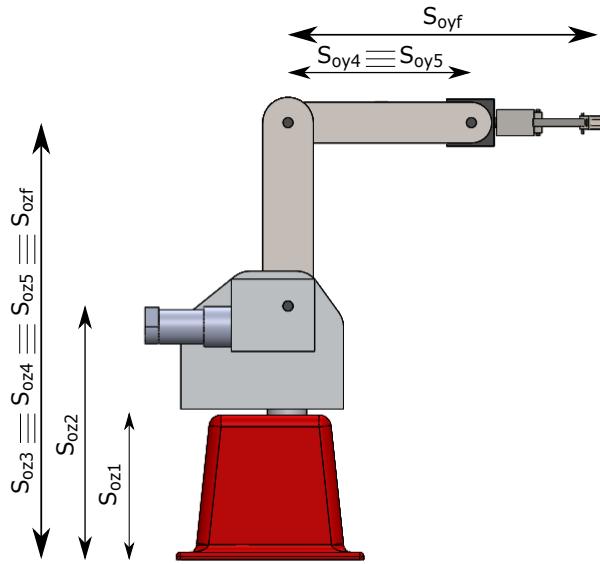


FIG. 3.11: Representação do parâmetro s_0 no Robô Pegasus (Autoria Própria)

Na TAB. 3.2, são mostrados os valores de cada um desses parâmetros. Esses dados foram obtidos a partir do próprio robô real, instalado no IDR Lab.

TAB. 3.2: Parâmetros s e s_0 da Teoria dos Helicoides do robô Pegasus

JUNTAS	S			So (em polegadas)		
	Sx	Sy	Sz	Sox	Soy	Soz
1	0	0	1	0	0	7,25
2	1	0	0	0	0	12,50
3	1	0	0	0	0	21,50
4	1	0	0	0	9	21,50
5	0	1	0	0	9	21,50
f	1	0	0	0	15	21,50

- Montar a Matriz de Transformação Homogênea (EQ. 7.83) para cada junta. Para isso, deve-se determinar as componentes da matriz de rotação (EQ. 7.54) de cada junta. Em seguida, associar os termos da translação q (EQ. 7.84, 7.85, 7.86) para cada junta. Como o robô Pegasus não possui junta de translação, todos os parâmetros t serão iguais a zero. Os parâmetros para realizar esse passo estão localizados na TAB. 3.2.

Utilizando o robô Pegasus, é possível montar as matrizes de transformação homogênea de cada junta, sem alterar os ângulos de rotação (θ), conforme a seguir:

$$T_1 = \begin{bmatrix} c_1 & -s_1 & 0 & 0 \\ s_1 & c_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.12)$$

$$T_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c_2 & -s_2 & 12,5s_2 \\ 0 & s_2 & c_2 & 12,5c_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.13)$$

$$T_3 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c_3 & -s_3 & 21,5s_3 \\ 0 & s_3 & c_3 & 21,5 - 21,5c_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.14)$$

$$T_4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c_4 & -s_4 & 21,5s_4 - 9c_4 + 9 \\ 0 & s_4 & c_4 & 21,5 - 9s_4 - 21,5c_4 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.15)$$

$$T_5 = \begin{bmatrix} c_5 & 0 & s_5 & -21,5s_5 \\ 0 & 1 & 0 & 0 \\ -s_5 & 0 & c_5 & 21,5 - 21,5c_5 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.16)$$

4. É necessário obter a MTH para ajustar a orientação do referencial da ferramenta para que esteja conforme o robô Pegasus, que tem o eixo z do efetuador final como mostra a FIG. 3.12.

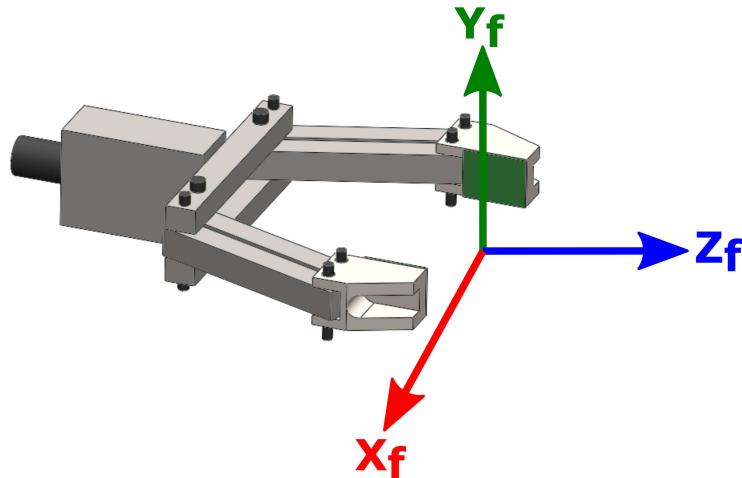


FIG. 3.12: Referencial da ferramenta do robô Pegasus (Autoria Própria)

O resultado é a matriz T_f que equivale a uma rotação elementar de -90° em torno de X em relação ao referencial absoluto.

$$T_f = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 15 \\ 0 & -1 & 0 & 21,5 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.17)$$

5. Aplicar o Método de Deslocamentos dos Helicoides Sucessivos para obter o produtório das matrizes de transformação homogênea de cada junta, a posição e a orientação da ferramenta. Assim, para o robô em questão, a matriz final pode ser representada como $T=T_1T_2T_3T_4T_5T_f$. A forma numérica das matrizes, para determinadas posições das juntas, será apresentada na seção 3.4.

3.4 VALIDAÇÃO DOS MÉTODOS

Com o objetivo de validar a cinemática direta do Robô Pegasus, por meio das duas metodologias, foram posicionadas as juntas do manipulador em locais conhecidos. A primeira posição foi com todas as juntas em posição de ***Home***, ou seja, com os ângulos iguais a zero grau, conforme a FIG. 3.13. Assim, as matrizes de transformação homogênea de cada junta, bem como, a MTH resultante (T), obtida através da produtório dessas matrizes, utilizando a Convenção de Denavit-Hartenberg, estão representadas a seguir:

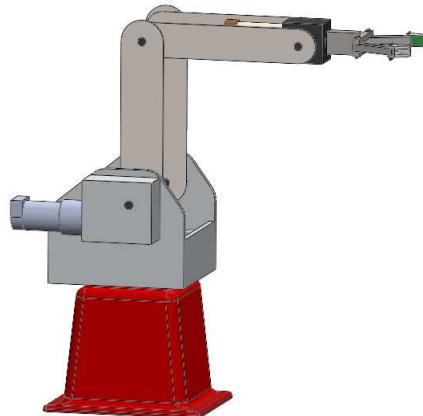


FIG. 3.13: Postura de *Home* com variáveis de junta $[0\ 0\ 0\ 0]^T$ (Autoria Própria)

$$T_0^b = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 7,25 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.18) \quad T_1^0 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 5,25 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.19)$$

$$T_2^1 = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 9 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.20) \quad T_3^2 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & -9 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.21)$$

$$T_4^3 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.22) \quad T_5^4 = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 6 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.23)$$

Como:

$$T = T_0^b T_1^1 T_2^1 T_3^2 T_4^3 T_5^4 T_f^5 \quad (3.24)$$

Logo:

$$T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 15 \\ 0 & -1 & 0 & 21,5 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.25)$$

Pelo Método dos Deslocamentos dos Helicoides Sucessivos, as matrizes de cada junta do manipulador, bem como a MTH resultante (T), podem ser verificadas pelas EQ. 3.26 a 3.31:

$$T_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.26) \quad T_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.27)$$

$$T_3 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.28) \quad T_4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.29)$$

$$T_5 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.30) \quad T_f = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 15 \\ 0 & -1 & 0 & 21,5 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.31)$$

Como:

$$T = T_1 T_2 T_3 T_4 T_5 T_f \quad (3.32)$$

Logo:

$$T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 15 \\ 0 & -1 & 0 & 21,5 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.33)$$

Já o segundo posicionamento de cada junta ficou determinado como: ângulo da junta 1 igual a 45° , das juntas 2, 4 e 5 iguais a 0° e da junta 3 igual a 30° , como pode ser visto na FIG. 3.14. Dessa forma, pela Convenção de Denavit-Hartenberg, foram obtidos os seguintes resultados:

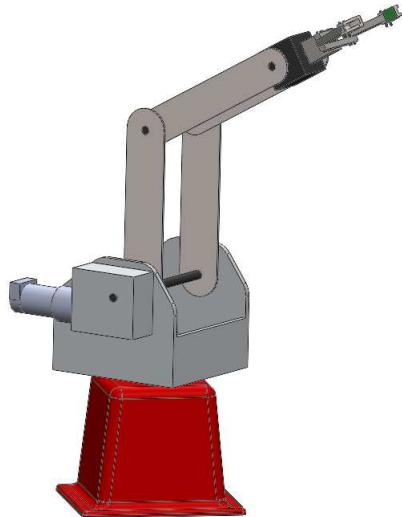


FIG. 3.14: Postura com variáveis de junta $[45 \ 0 \ 30 \ 0 \ 0]^T$ (Autoria Própria)

$$T_0^b = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 7,25 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.34) \quad T_1^0 = \begin{bmatrix} 0,7071 & 0 & 0,7071 & 0 \\ 0,7071 & 0 & -0,7071 & 0 \\ 0 & 1 & 0 & 5,25 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.35)$$

$$T_2^1 = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 9 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.36) \quad T_3^2 = \begin{bmatrix} 0,5 & 0,866 & 0 & 4,5 \\ -0,866 & 0,5 & 0 & -7,7942 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.37)$$

$$T_4^3 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.38) \quad T_5^4 = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 6 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.39)$$

$$T = \begin{bmatrix} 0,7071 & -0,3536 & -0,6124 & -9,1856 \\ 0,7071 & 0,3536 & 0,6124 & 9,1856 \\ 0 & -0,866 & 0,5 & 29 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.40)$$

Para o Método dos Deslocamentos dos Helicoides Sucessivos, foram encontrados os seguintes valores:

$$T_1 = \begin{bmatrix} 0,7071 & -0,7071 & 0 & 0 \\ 0,7071 & 0,7071 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.41) \quad T_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.42)$$

$$T_3 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0,866 & -0,5 & 10,75 \\ 0 & 0,5 & 0,866 & 2,8805 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.43) \quad T_4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.44)$$

$$T_5 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.45) \quad T_f = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 15 \\ 0 & -1 & 0 & 21,5 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.46)$$

$$T = \begin{bmatrix} 0,7071 & -0,3536 & -0,6124 & -9,1856 \\ 0,7071 & 0,3536 & 0,6124 & 9,1856 \\ 0 & -0,866 & 0,5 & 29 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.47)$$

3.5 COMPARAÇÃO ENTRE OS MÉTODOS

A partir dos dados apresentados nas seções 3.2, 3.3, 3.4, pode-se fazer uma comparação entre os resultados apresentados com as duas metodologias, a Convenção de Denavit-Hartenberg, e o Método dos Deslocamentos dos Helicoides Sucessivos, apontando as vantagens e desvantagens do seu emprego.

A Convenção de Denavit-Hartenberg possui MTH's mais simples do que as de Helicoide, além de ser o método mais utilizado pela comunidade científica.

Entretanto, o método do Deslocamentos dos Helicoides Sucessivos apresenta algumas vantagens sobre o primeiro:

- necessita de menos etapas para a resolução cinemática, apenas 5, enquanto que a Convenção de Denavit-Hartenberg exige 14 etapas;
- apesar de ser necessário determinar 2 vetores (s e s_0) e as magnitudes t e θ para a sua resolução, enquanto para D-H devem ser encontrados apenas 4 parâmetros (a, d, α, ϑ), os mesmos são facilmente obtidos a partir do *Datasheet* do Robô, porém por D-H não é tão intuitivo quanto, sendo necessário realizar os 11 passos para encontrá-los;
- possui somente 2 referenciais, um de base e outra da ferramenta (FIG. 3.15 (a)), enquanto que D-H exige, pelo menos, $2n + 1$ (n = número de juntas) referenciais (FIG. 3.15 (b)); e,
- o deslocamento de cada junta é sempre em relação ao referencial de base, o que torna mais fácil sua verificação. Já para D-H, o deslocamento é relativo ao referencial da junta anterior.

Finalmente, é importante ressaltar que a complexidade para serem encontrados os parâmetros de D-H, fazem com que, em um primeiro momento, este se torne um método complicado de se entender quando modelado por um iniciante, podendo propiciar erros mais facilmente. Em contrapartida, os parâmetros necessários para a utilização da teoria dos Helicoides podem ser encontrados com mais facilidade, diminuindo assim as chances de erros.

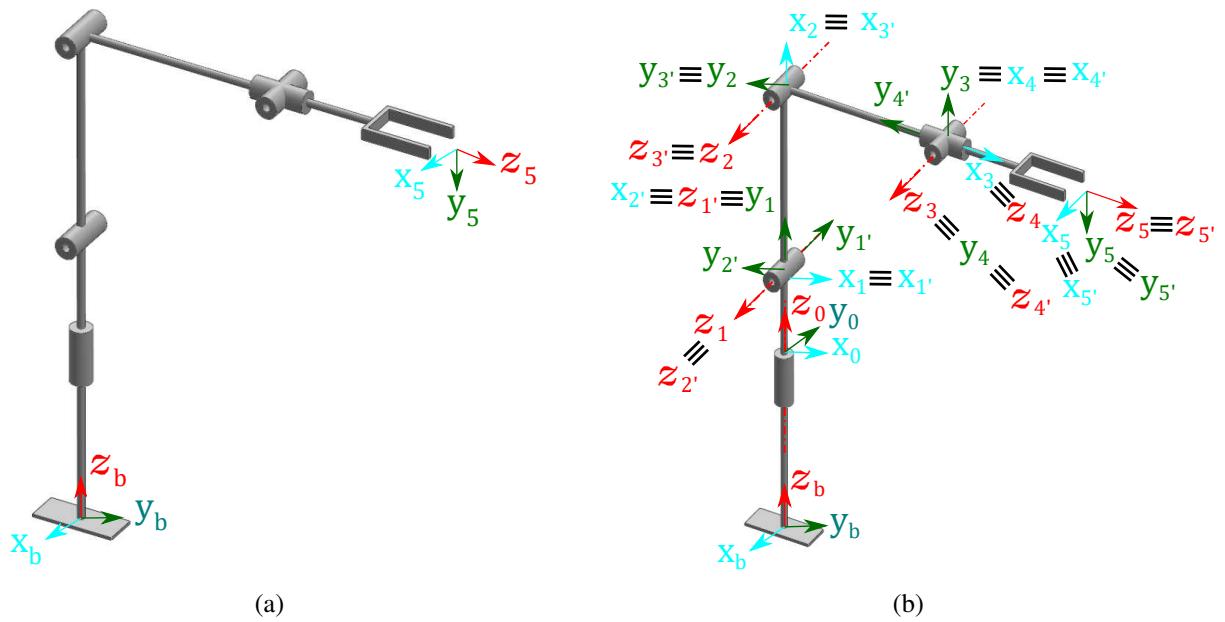


FIG. 3.15: Referenciais utilizados por (a) Teoria dos Helicoides e (b) D-H (Autoria Própria)

4 CARPA: MODELAGEM, SIMULAÇÃO E PROGRAMAÇÃO

Nesse capítulo, são mostradas as etapas de modelagem e simulação do CAPRON, específico para o robô Pegasus, e o procedimento de utilização do CARPA e suas principais funcionalidades.

4.1 MODELAGEM E SIMULAÇÃO

Foi desenvolvida e implementada computacionalmente uma metodologia que auxilia a modelagem cinemática e a programação de robôs industriais, possibilitando que as etapas associadas possam ser auxiliadas, por meio de computador, com uma interface amigável e intuitiva. Inicialmente, encontra-se disponível um cenário virtual de um laboratório composto por modelos de multicorpos, representando piso, paredes e iluminação, gerados em CAD 3D. Os modelos dos robôs podem ser inseridos, retratando o equipamento real, e a movimentação junta a junta foi implementada, utilizando o Métodos dos Deslocamentos dos Helicoides Sucessivos. A tela inicial do CARPA, considerando que não existe ainda nenhum robô modelado, pode ser vista na FIG. 4.1.

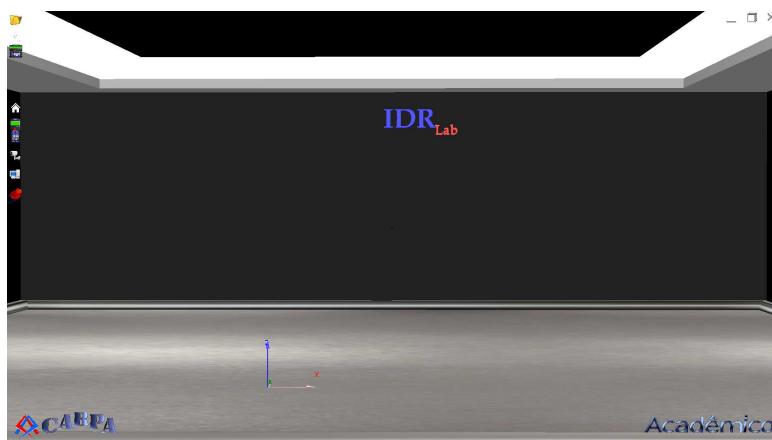


FIG. 4.1: Tela Inicial do CARPA (Autoria Própria)

O resultado é um protótipo computacional, intitulado CARPA-Pegasus, que, além de permitir a modelagem e a simulação de qualquer robô industrial, particulariza a programação para o robô Pegasus, pois possibilita a geração automática dos arquivos de programação para a execução da tarefa de *pick-and-place*.

4.2 O SISTEMA CAPRON

Um objetivo secundário surgiu naturalmente ao constatar que o CARPA-Pegasus é efetivo para o treinamento de iniciantes das funcionalidades do *Teach Pendant*. Nessa função, o sistema recebe nome de CAPRON (*Computer-Aided Pegasus Robot On-line Programming*). A interface inicial desse simulador é mostrada na FIG. 4.2. Porém, este é específico para o FMS Pegasus da empresa AMATROL, conforme a FIG. 4.3, visto que é o sistema robótico disponível no IDR Lab.

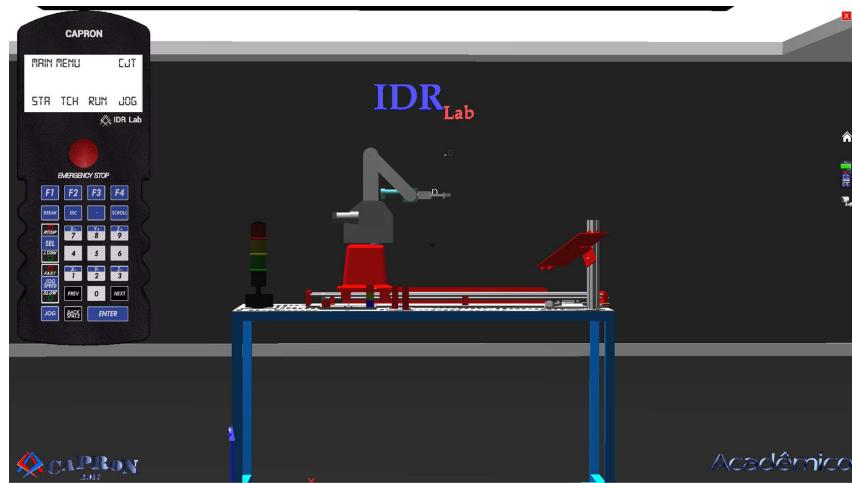


FIG. 4.2: Tela Inicial do CAPRON (Autoria Própria)

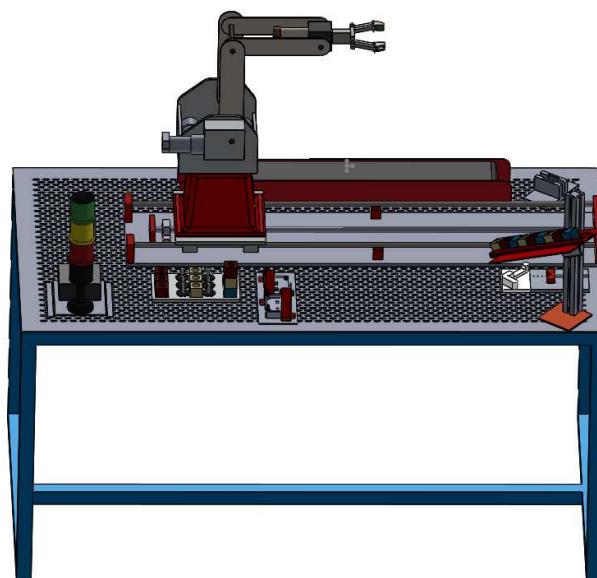


FIG. 4.3: FMS do robô Pegasus, localizado no IDR Lab (Autoria Própria)

Para permitir simular os movimentos e atualizar a localização dos corpos do robô Pegasus no CAPRON, foi implementado o Método dos Deslocamentos de Helicoides Sucessivos na resolução da cinemática direta, por meio do modelo de um Teach Pendant, no qual a disposição do teclado real e do virtual são rigorosamente as mesmas, por se tratar de um simulador para treinamento, porém o invólucro sofreu adaptações, como pode ser verificado na FIG. 4.4. Sua modelagem foi feita por meio da versão acadêmica do 3DS Max, tendo a lógica de atuação sobre o robô virtual implementada computacionalmente em DELPHI, após um cuidadoso mapeamento das funcionalidades do TP real.



FIG. 4.4: TP do robô Pegasus real e virtual (SANTOS et al. (2017a))

O TP é composto por um *display*, 25 teclas multifuncionais e um botão de emergência. O display, alterna 7 (sete) menus distintos de opções de comandos. Assim sendo, existe uma variedade de funcionalidades, que precisam ser bem conhecidas e, com o tempo de uso, são memorizadas, permitindo a correta operação do robô real. Todavia, para o iniciante, é natural ser complicado navegar entre teclas e menus para achar determinada funcionalidade, o que particularmente torna as primeiras programações um processo ainda mais demorado. Com o tempo de manipulação do TP, a dificuldade inicial dá lugar a um automatismo mental por parte do programador, que passa a acessar naturalmente as funcionalidades do TP.

Para a explicação das funcionalidades do TP, as diversas possibilidades de menu são apresentadas a seguir:

tadas na FIG. 4.5, dividida em índices de (a) até (j). A seguir, encontra-se explicado cada menu, com o respectivo índice referente à FIG. 4.5 associado ao texto para facilitar o entendimento da sequência de acesso às funcionalidades do TP.

Ao ligar o controlador, os motores elétricos das juntas do robô encontram-se desenergizados e o display do TP apresenta ao usuário o menu inicial (a).

O primeiro passo é energizar os motores das juntas, pressionando a tecla F1 (a), que se relaciona com a função STA de Start. Em seguida, F2 (b), ativa o robô, por meio da função ENA (c) de Enabled.

Para retornar à tela inicial, visando escolher outras funcionalidades, é necessário pressionar o botão ESC (d).

O segundo passo é informar ao controlador as velocidades máxima (Fast) e mínima (Slow), nas quais o usuário deseja mover o robô. Para isso, deve-se pressionar o botão F4 (e), referente à função JOG. As velocidades de default (f) para Fast e Slow são 200 e 10, respectivamente, sendo que a maior velocidade possível a ajustar é 255.

Para alterar o Fast, pressiona-se o botão F1 (g), de forma que o respectivo campo relativo para alterar os valores da velocidade Fast fique editável. Em seguida, define-se a velocidade, por meio das teclas alfanuméricas e, por fim, basta pressionar a tecla ENTER (g) para confirmar o valor desejado. É possível observar que o valor de Fast foi alterado de 200 (f) para 225 (h).

Procedimento análogo pode ser adotado para a alterar a velocidade Slow, porém a tecla a ser pressionada passa a ser F2 (i), onde em (j) passa de 10 para 20.

Com as velocidades definidas, é necessário zerar os encoders, colocando o Pegasus na posição de HOME, por meio da tecla Esc (d), seguida de F1, que passa a representar o HOME (c).

O terceiro passo, para permitir o movimento do robô, é pressionar o botão JOG, a fim de habilitar as teclas SEL e JOG SPEED. Assim sendo, a cada acionamento da tecla SEL, é possível alterar o sentido (Right/Up ou Left/Down) da junta a ser movida. Analogamente, pressionando alternadamente a tecla JOG SPEED, a velocidade Fast ou Slow é empregada para movimentar as juntas do robô. Tais procedimentos podem ser visualizados na Figura FIG. 4.5 de (e) à (j).

Após cumprir as três etapas, por meio das teclas alfanuméricas, pode-se controlar cada uma das cinco juntas do Robô Pegasus, pressionando as teclas de 1 a 5, além da tecla referente ao número 7 para a junta prismática que provê translação à base do Pegasus.

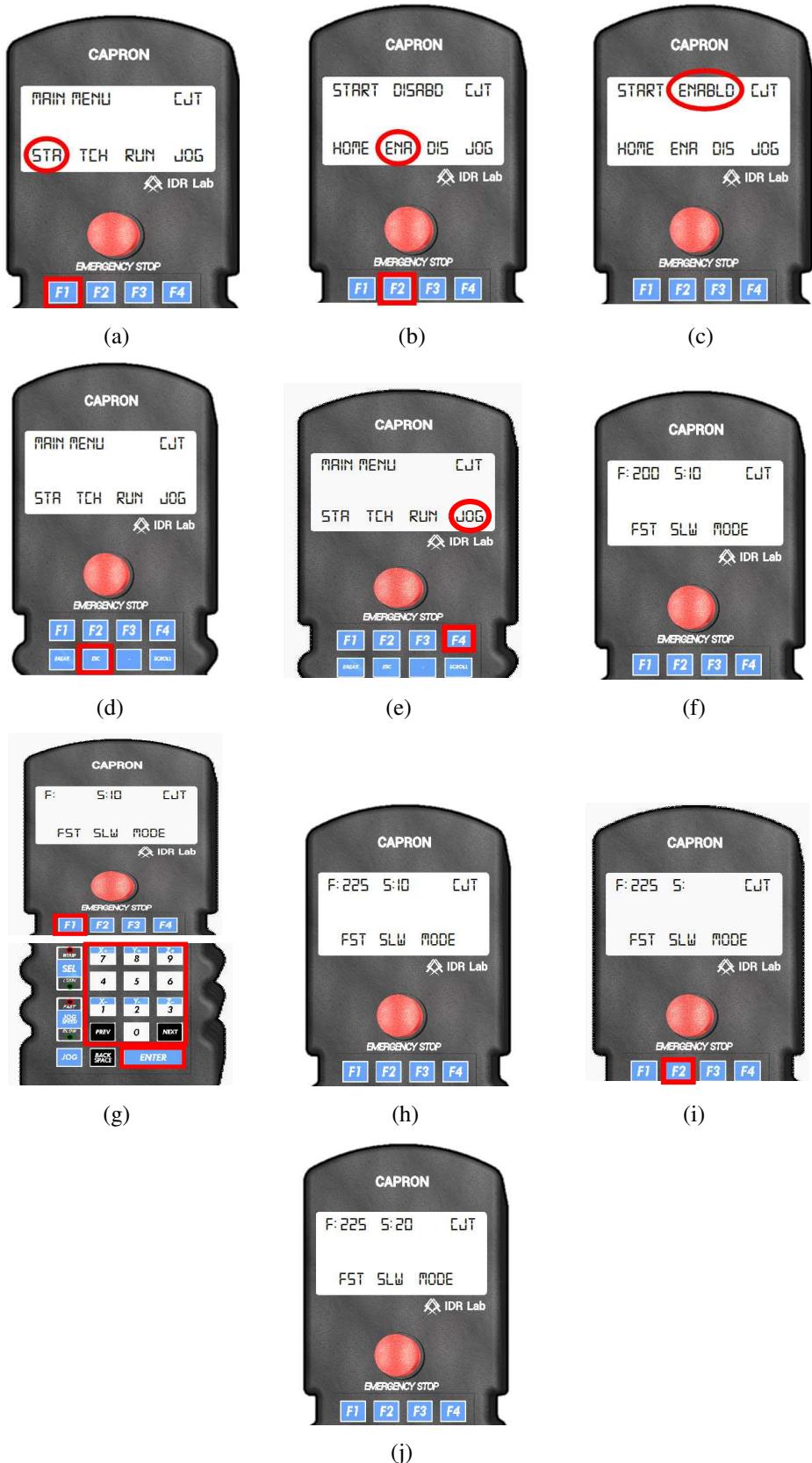


FIG. 4.5: Sequência de acesso às funcionalidades do TP (SANTOS et al. (2017a))

Após movimentar a garra do robô para uma posição desejada, surge a etapa quatro, correspondente ao TEACH POINT, ou seja, a posição de cada encoder é gravada na memória do controlador, a fim de que as varáveis de junta fiquem armazenadas no arquivo do programa. Para isso, pressiona-se a tecla F2, como mostrado na FIG. 4.5 (d), e, em seguida, F1, referente à função TCH de Teach, que pode ser visualizado na FIG. 4.6. Entretanto, é possível escolher a posição em que se deseja ensinar o ponto. Para isso, existem os botões Prev de Previous e Next, os quais permitem ir para a posição anterior ou para a posterior, respectivamente.



FIG. 4.6: Função TCH (SANTOS et al. (2017a))

A etapa cinco se refere à movimentação entre pontos ensinados para validação e testes via TP, o que ocorre ao pressionar a tecla F4, referente à Pmove (PMV), como pode ser visualizado na FIG. 4.7, onde um novo menu aparece no display, possibilitando digitar o número do ponto para o qual se quer mover, por meio das teclas alfanuméricas, estando a função JOG desabilitada. Por fim, é preciso acionar o botão ENTER para confirmar a movimentação para o ponto digitado.

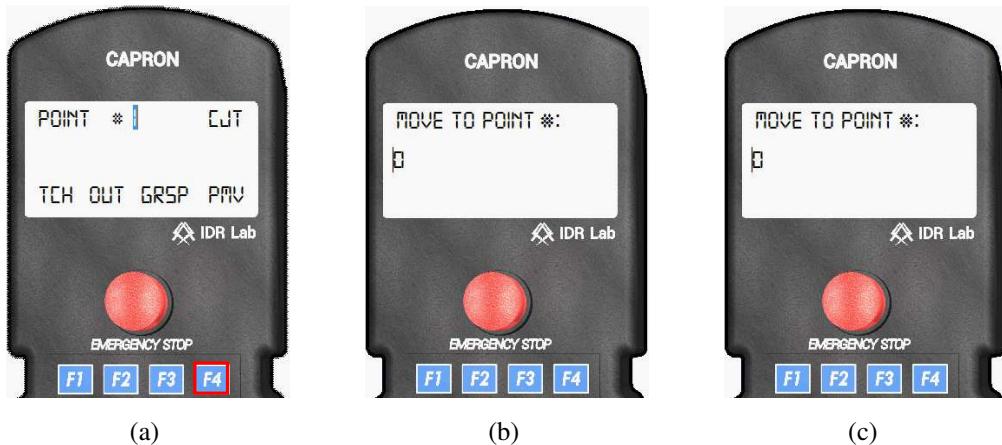


FIG. 4.7: Função Pmove - PMV (SANTOS et al. (2017a))

Diante das inúmeras etapas supracitadas, é natural que o programador iniciante leve tempo para localizar as funções e menus, o que demanda treinamento com o TP, motivando o desenvolvimento de um simulador.

Assim, para validação da efetividade do CAPRON, conforme SANTOS et al. (2017a) foram treinados 14 alunos do 5º ano da graduação de Engenharia Mecânica de 2017 do IME, no IDR Lab, como pode ser visto na FIG. 4.8, utilizando uma metodologia, conhecida como *Coding Dojo*. Inicialmente, Dave Thomas propôs a ideia de *Code Kata* como um exercício (SATO et al., 2008), em que programadores poderiam escrever códigos descartáveis, a fim de praticar seus rascunhos fora do ambiente de trabalho. Posteriormente, conforme SATO et al. (2008), surge o conceito de *Coding Dojo*, no qual um grupo de programadores se reúne para resolver o *Code Kata* juntos. A palavra Dojo é de origem japonesa, que significa local de treinamento. Assim, *Coding Dojo* é o Local de Treinamento de Programação (BONFIM, 2016).



FIG. 4.8: Alunos de graduação utilizando o CAPRON (Autoria Própria)

De acordo com SATO et al. (2008), o principal fundamento desse método é criar um ambiente seguro, que seja colaborativo, inclusivo e não competitivo, em que as pessoas possam estar em um aprendizado contínuo. BONFIM (2016) completa que, além do princípio anterior, deve-se promover o *networking* e o compartilhamento de ideias entre os membros da equipe, tendo como objetivo desafiar os programadores com novos problemas, buscando novas soluções, de forma a aprimorar as práticas de programação, saindo da zona de conforto. O conhecimento obtido, durante a realização do *Coding Dojo*, pode ser utilizado pelos programadores em outras atividades, aumentando a qualidade do trabalho no dia-a-dia.

Conforme BONFIM (2016), existem alguns formatos que podem ser usados para a implementação do *Coding Dojo*, porém foram utilizadas apenas duas:

- Kata - nesse formato, existe um apresentador que deve demonstrar uma solução previa-

mente desenvolvida. O objetivo é ensinar aos participantes todos os passos necessários e permitir que todos possam reproduzir o mesmo resultado. Neste, é permitido realizar interrupções para sanar dúvidas a qualquer momento; e,

- Randori - nesse formato, há a participação de todos. É proposto um problema a ser resolvido e a programação é realizada por pares, em apenas uma máquina, a qual todos têm acesso visual ao mesmo tempo.

Assim, foi proposto um estudo de caso, que permitiu detectar oportunidades de melhorias e levantar índices quantitativos de redução do tempo de programação: foram divididos quatorze alunos, em dois grupos de sete, para realizar o experimento que consiste em programar o manipulador, disponível no IDR LAB do IME, em uma operação de *pick-and-place* de três peças, em que as mesmas devem estar posicionadas no *pallet* (local de armazenamento), inicialmente, de baixo para cima, na ordem azul, amarela e vermelha, e atingir a ordem amarelo, vermelho e azul, conforme a FIG. 4.9.

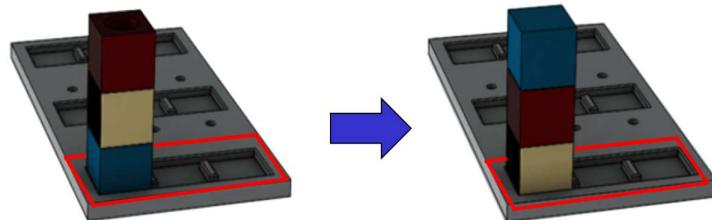


FIG. 4.9: Posições inicial e final das peças no local de armazenamento (Autoria Própria)

Enquanto o grupo 1 utilizou diretamente o sistema robótico real, tendo apenas os treinamentos de segurança e de programação do robô Pegasus, o grupo 2, adicionalmente, usou o sistema de auxílio CAPRON, antes de usar o robô real para realizar a sua programação, com o intuito de ganhar destreza nas funcionalidades e manuseio do *Teach Pendant* e de verificar a importância do simulador para o aprendizado do programador, em relação ao conhecimento das múltiplas funções das teclas do TP, que possibilitam acessar diversos menus.

Após o término das atividades pelos dois grupos, foi possível verificar, conforme a FIG. 4.10, que houve uma redução média de, aproximadamente, 48% do tempo de programação entre os grupos, ou seja, enquanto o primeiro grupo necessitou, em média, de 2 horas e 21 minutos, o grupo 2, de apenas 1 hora e 13 minutos. Além da redução do tempo de utilização de robô, houve também diminuição no tempo necessário de auxílio e supervisão do instrutor, um aluno de mestrado, que pôde utilizar esse tempo para realizar pesquisas e demais atividades associadas.

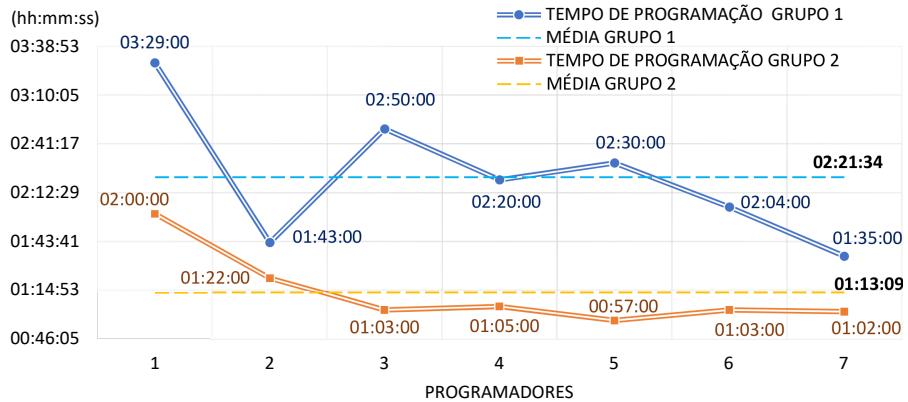


FIG. 4.10: Tempos de programação dos usuários por grupo (SANTOS et al., 2017a)

4.3 O SISTEMA CARPA

Assim como o CAPRON, a implementação computacional do CARPA foi feita utilizando a versão acadêmica do ambiente de desenvolvimento integrado (IDE) de softwares, o RAD Studio Berlin. Nessa ferramenta, foi utilizado Delphi, incorporado à biblioteca livre *GLScene*, permitindo criar e manipular objetos e cenas 3D, em tempo de execução em cenário de Realidade Virtual (VR - *Virtual Reality*). Os modelos em CAD 3D utilizaram as versões acadêmicas dos softwares SolidWorks e 3DS Max. Quando empresas fabricantes, como a ABB, KUKA, YASKAWA Motoman e Stäubli disponibilizam os modelos dos robôs nos seus sites, os mesmos podem ser usados. Na FIG. 4.11, é possível verificar a modelagem no CARPA de alguns desses manipuladores robóticos, tais como o IRB120 da ABB, o Motoman MH5F, o KR6 da KUKA e o TX90 da Staubli, além do Pegasus da Amatrol.



FIG. 4.11: Robôs genéricos modelados no CARPA (Autoria Própria)

A concepção da metodologia, com implementação computacional, buscou amparo e inspiração em trabalhos anteriores na literatura específica, apresentados na seção 2.8, além de utilizar modelos em CAD 3D, feitos pela equipe que atualmente trabalha/estuda no IDR Lab. Essas atividades possibilitaram a obtenção do protótipo computacional ao longo dos dois anos destinados ao mestrado.

Implementado no RAD Studio Berlim, o CARPA utilizou conceitos de OOP e programação orientada a eventos. Além disso, o sistema foi implementado para o ambiente Windows.

A fundamentação teórica da resolução cinemática dos robôs industriais seriais utilizou como base a Teoria dos Helicoides, mais especificamente, o Método dos Deslocamentos dos Helicoides Sucessivos para a cinemática direta.

Para a programação do robô Pegasus, utilizou-se os comando da linguagem MCL da Amatrol, nativo desse robô, visando possibilitar a validação prática. Os arquivos foram gerados através dos fundamentos de criação de arquivos em Delphi, guardando as particularidades dessa linguagem, assim como as extensões adequadas à leitura pelo controlador do Pegasus.

O CARPA é definido como um sistema que permite a modelagem e simulação de diversos tipos de robôs industriais seriais, além de possibilitar a programação específica do robô Pegasus, visto que a geração automática do código é totalmente dependente da linguagem de programação de cada robô. Além disso, possui uma interface amigável, desenvolvida para o Sistema Operacional Windows, herdando a filosofia de orientação a eventos, de fácil entendimento, de forma que o usuário não tenha necessidade de conhecer, profundamente, a Teoria dos Helicoides para resolução cinemática, implementada computacionalmente. De forma a instruir o uso do CARPA, algumas de suas principais funcionalidades encontram-se explicadas e exemplificadas a seguir.

4.3.1 INSERIR UM ROBÔ MODELADO

O CARPA possui um base de dados de robôs industriais seriais, previamente, modelados, em que se buscou englobar o máximo de fabricantes possíveis, retirando a atividade de modelagem do usuário. Além dos robôs já contidos na biblioteca, permite ainda inserir uma quantidade ilimitada destes. Assim, o procedimento para utilização desses manipuladores é conforme as FIG. 4.12 à 4.15, em que mostrado um exemplo com o robô IRB120 da empresa ABB. Primeiro, deve-se clicar no ícone "Arquivos" (FIG. 4.12). Em seguida, clicar sobre "Abrir" na seção "Robôs" (FIG. 4.13). Na sequência, irá aparecer uma aba, em que se poderá escolher o robô desejado, clicando uma vez sobre o mesmo (FIG. 4.14). Por fim, é possível verificar todos os

parâmetros do robô (FIG. 4.15).

Conforme mostrado na FIG. 4.12, pode-se verificar que, em um cenário, existe um referencial absoluto, com coordenadas (X, Y, Z) nulas. Assim, tudo que for inserido no ambiente será posicionado, inicialmente, nesse ponto, podendo-se alterar a sua localização em relação a esse referencial, caso o usuário deseje.

Como pode ser visto na FIG. 4.15, o robô industrial possui 2 referenciais, um de base, o qual possui um corpo que liga a sua origem até a primeira junta, e um da ferramenta, que liga a última junta ao *Tool Center Point (TCP)*. Além disso, é composto por $n + 1$ multicorpos, unidos por n juntas, as quais possuem um vetor s e um vetor s_0 . Cada junta possui um grau de liberdade, que está associado à variável de junta - se for do tipo rotação, a variável é θ , caso seja de translação, é t . Cada junta tem sua respectiva Matriz de Transformação Homogênea baseada em Helicoides (transparente ao usuário), além de um corpo a frente que a mesma irá influenciar. Esse conceito permite que seja aplicado o Método do Deslocamento dos Helicoides Sucessivos, para se obter a Cinemática Direta do robô, de forma automatizada, dispensando um conhecimento profundo por parte do usuário acerca do método.

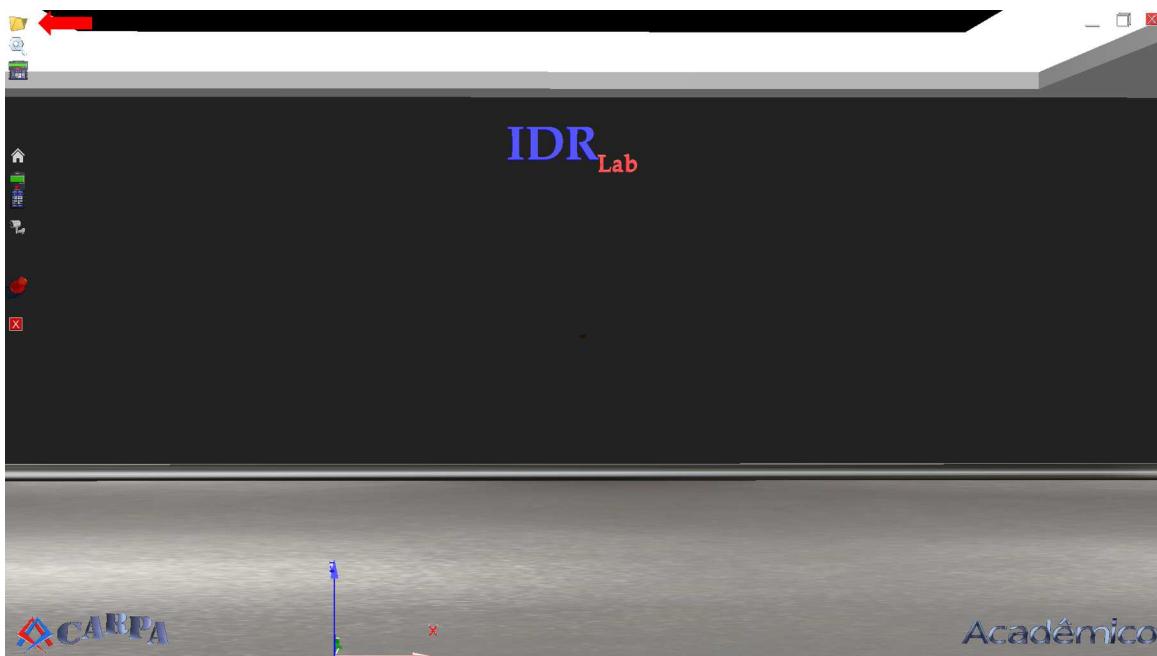


FIG. 4.12: Inserir robô modelado: evento botão "Arquivo" (Autoria Própria)



FIG. 4.13: Inserir robô modelado: evento botão "Abrir" na seção "Robôs" (Autoria Própria)



FIG. 4.14: Inserir robô modelado: evento escolher o robô (Autoria Própria)

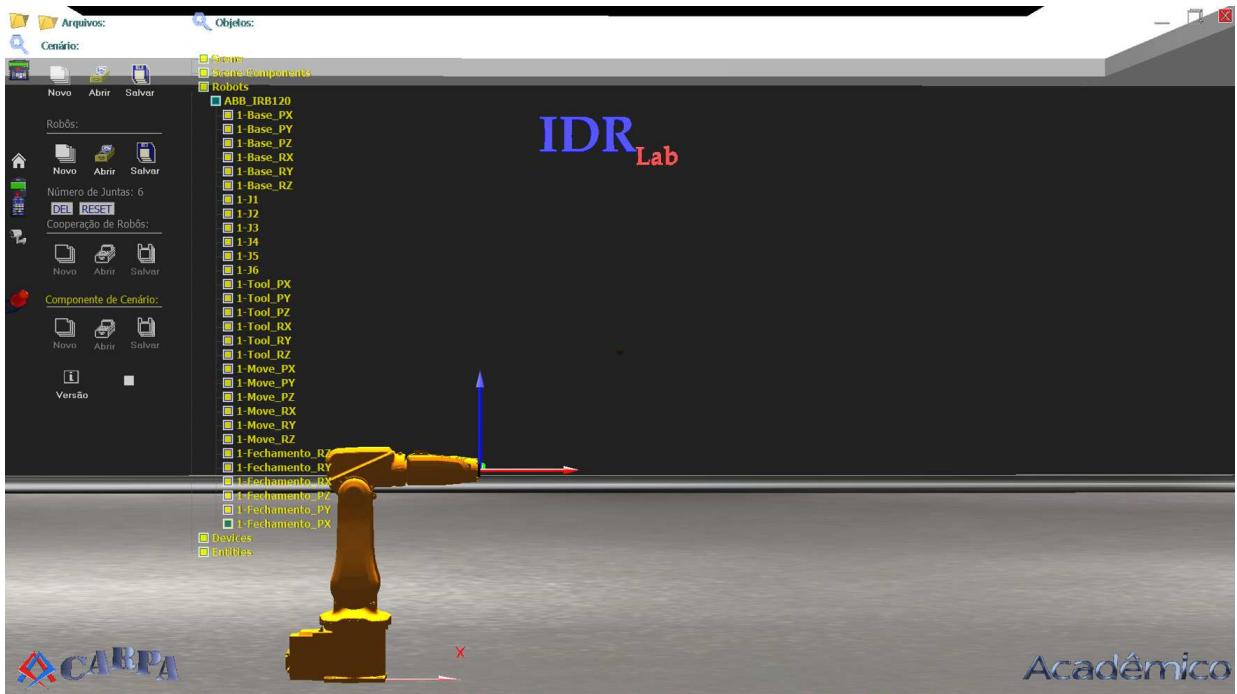


FIG. 4.15: Inserir robô modelado: robô IRB120 selecionado (Autoria Própria)

4.3.2 INSERIR NOVOS OBJETOS

Outra funcionalidade do CARPA é a possibilidade de inserir novos objetos, de acordo com as necessidades do usuário, dos quais a representação gráfica em CAD 3D deve estar em arquivos no formato 3DS, oriundo do *software* 3DS Max. Nas FIG. 4.16 à 4.22, é representado como executar essa atividade, utilizando, como exemplo, o sistema real do robô Pegasus, encontrado no IDR Lab (FIG. 4.3).

Inicialmente, deve-se inserir os dispositivos do sistema, como a mesa, as peças cúbicas e o alimentador de peças, informando o número de juntas (FIG. 4.16). Cada um desses objetos deve ser considerado como se fosse um robô de uma junta. Em seguida, irá aparecer um *treeview* com os seus parâmetros (FIG. 4.17). A partir disso, deve-se abrir o arquivo desejado da seguinte forma: clicar no item "1–Junta_1" do *treeview*; irá surgir uma aba, em que se deve pressionar o botão "IMG/S", a fim de que surja uma janela, onde será possível escolher o arquivo desejado (FIG. 4.18). Assim, na FIG. 4.19, é possível verificar o objeto inserido, o qual está posicionado exatamente no referencial absoluto. Analogamente, com exceção do robô, é feito para todos os componentes do sistema, os quais são posicionados, de acordo com o sistema robótico real (FIG. 4.20).

Após todo o sistema ter sido inserido no ambiente virtual, pode-se colocar o robô. O valor *default* do número de juntas é igual a 6, pois a maioria dos robôs industriais possui essa quantidade (FIG. 4.21), porém esse valor pode ser alterado, se necessário, bastando que o usuário digite um outro valor que confira ao número de juntas. Após definido o total de juntas (no exemplo, é igual a 6), abrirá novamente o *treeview*, mas agora com os dados referentes ao robô, representando automaticamente sua Cadeia Cinemática.

De forma análoga ao que foi feito com os dispositivos, para cada um dos itens do robô selecionados no *treeview*, foi inserido um arquivo, representando seus respectivos corpos.

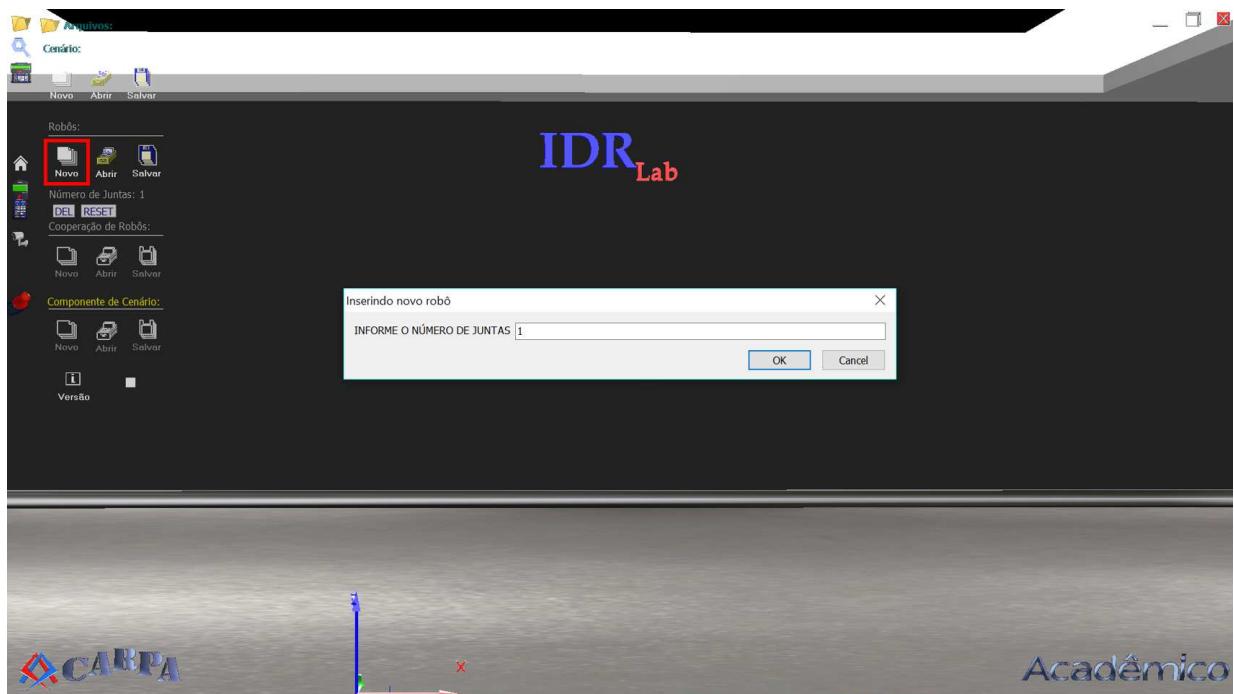


FIG. 4.16: Inserir novos objetos: informar o número de juntas do objeto (Autoria Própria)

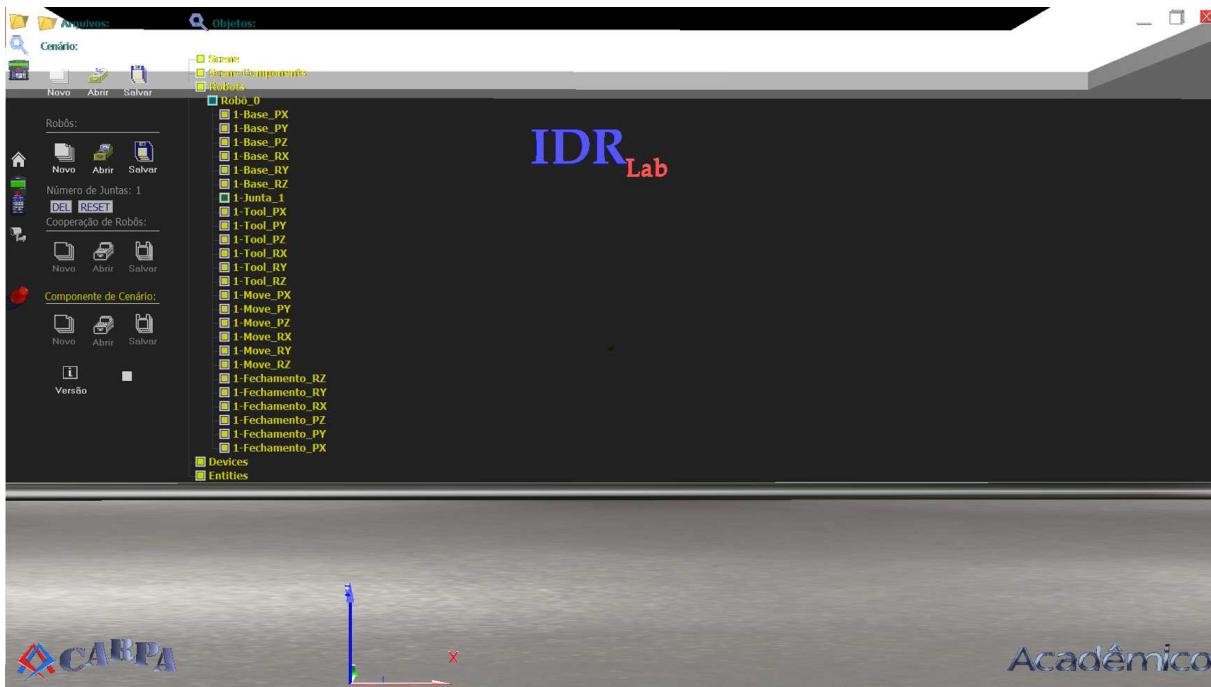


FIG. 4.17: Inserir novos objetos: parâmetros no *treeview* (Autoria Própria)

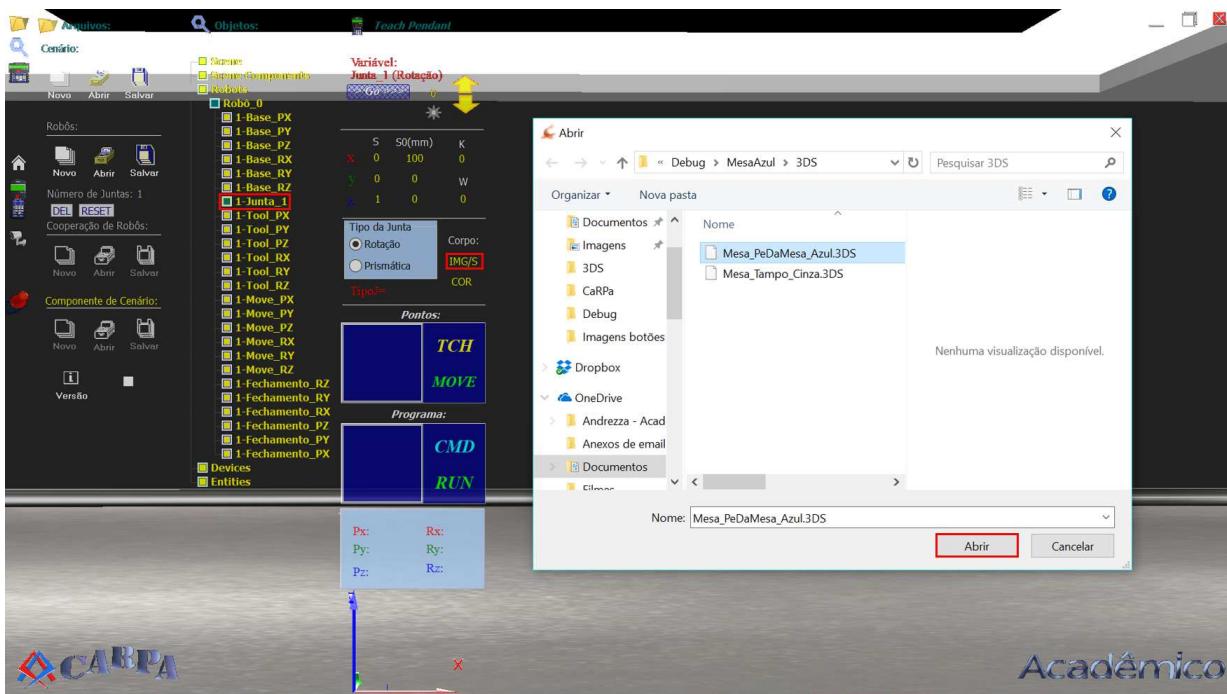


FIG. 4.18: Inserir novos objetos: evento botão "IMG/S" (Autoria Própria)

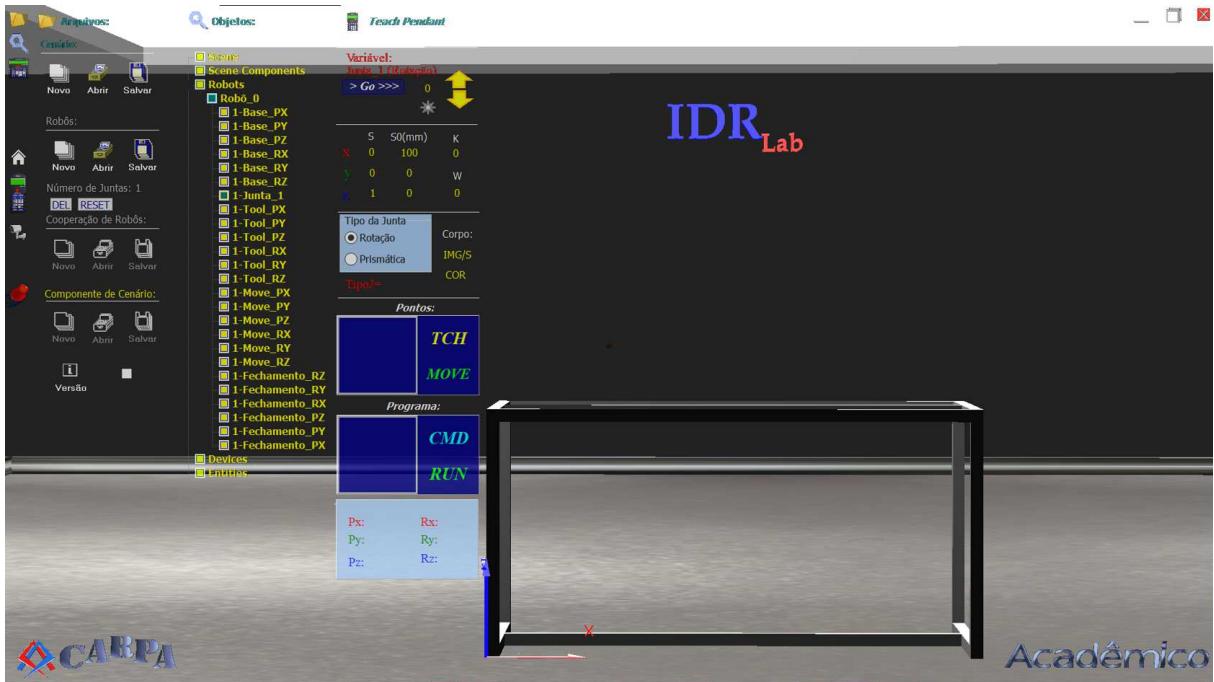


FIG. 4.19: Inserir novos objetos: objeto inserido no ambiente virtual (Autoria Própria)



FIG. 4.20: Inserir novos objetos: demais objetos inseridos (Autoria Própria)

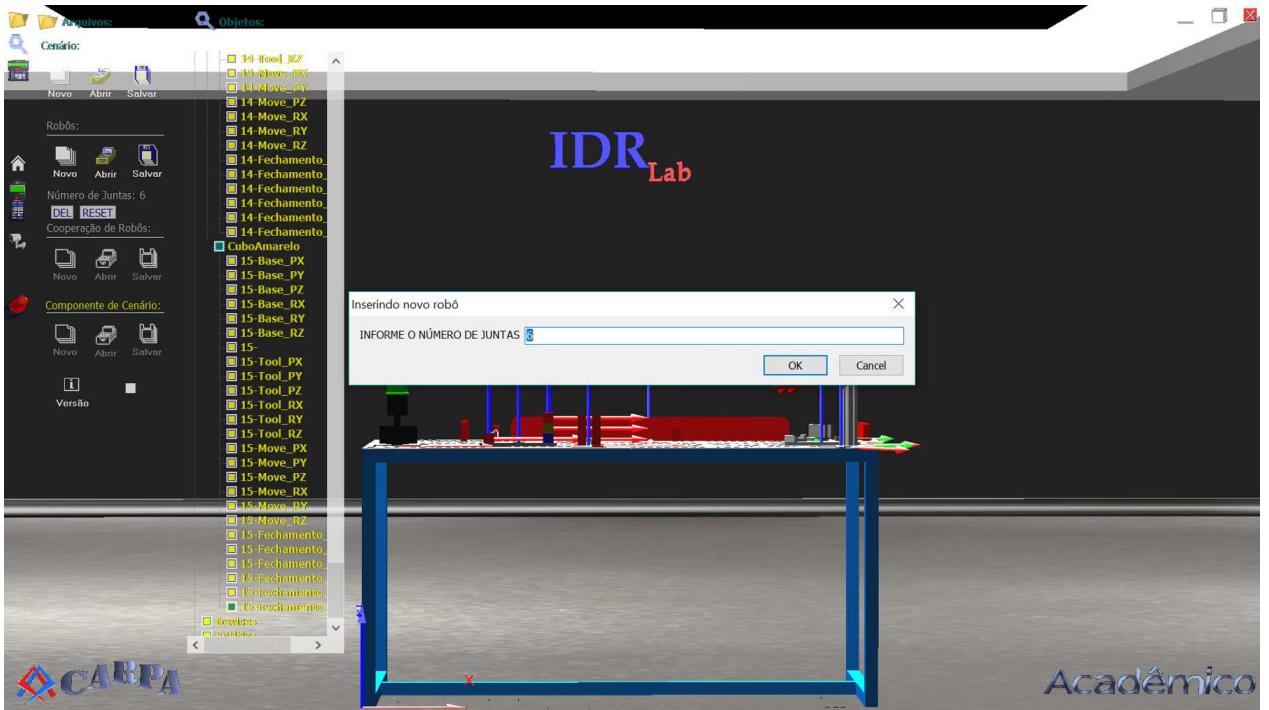


FIG. 4.21: Inserir novos objetos: informar o número de juntas do robô (Autoria Própria)

Basta o usuário definir o vetor s , a direção da junta em torno ou ao longo do qual o helicoide rotaciona ou translada, respectivamente, e o vetor s_0 , distância em (X , Y e Z) do vetor da base até qualquer ponto do eixo da junta. No caso do robô Pegasus, esses parâmetros podem ser encontrados na TAB. 3.2. Seguindo os fundamentos do método dos Deslocamentos dos Helicoides Sucessivos, cada junta está associada ao corpo anterior, uma vez que a junta i interfere no posicionamento do corpo $i + 1$ (FIG. 4.22). Logo, é possível encontrar o mesmo resultado para Matriz de Transformação Homogênea final, realizando tanto a multiplicação por matrizes elementares ($T=R_z.R_y.R_x.P_z.P_y.P_x$), quanto o produtório por Helicoide ($T=T_n.T_{n+1}...T_{n-1}.T_n$), teoria implementada no CARPA.

$$T_{Rx} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(r_x) & -\sin(r_x) & 0 \\ 0 & \sin(r_x) & \cos(r_x) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad T_{Ry} = \begin{bmatrix} \cos(r_y) & 0 & \sin(r_y) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(r_y) & 0 & \cos(r_y) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_{Rz} = \begin{bmatrix} \cos(r_z) & -\sin(r_z) & 0 & 0 \\ \sin(r_z) & \cos(r_z) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad T_{Px} = \begin{bmatrix} 1 & 0 & 0 & P_x \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_{Py} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & P_y \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad T_{Pz} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



FIG. 4.22: Inserir novos objetos: parâmetros do robô inseridos (Autoria Própria)

4.3.3 ASSOCIAR COR AOS OBJETOS

Pelo CARPA, é possível também dar cores a cada corpo do objeto inserido. Para isso, basta clicar no botão "COR" que, em seguida, irá abrir uma tela com diversas tonalidades de cores, podendo escolher alguma delas e aplicar, conforme FIG. 4.23 e 4.24.

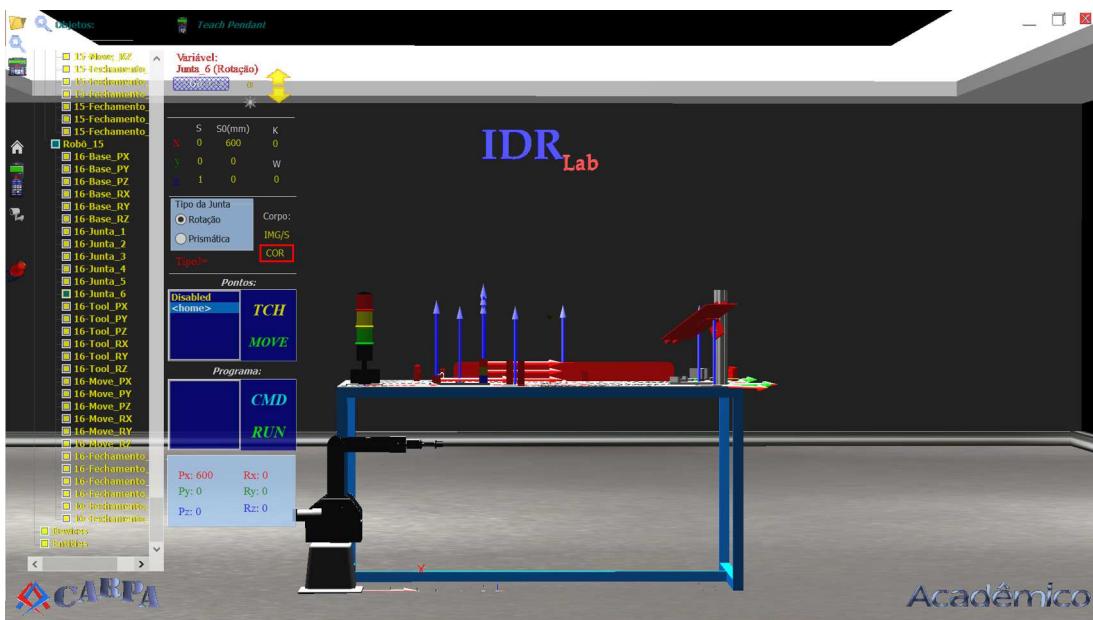


FIG. 4.23: Associar cor ao objeto: evento botão "COR" (Autoria Própria)

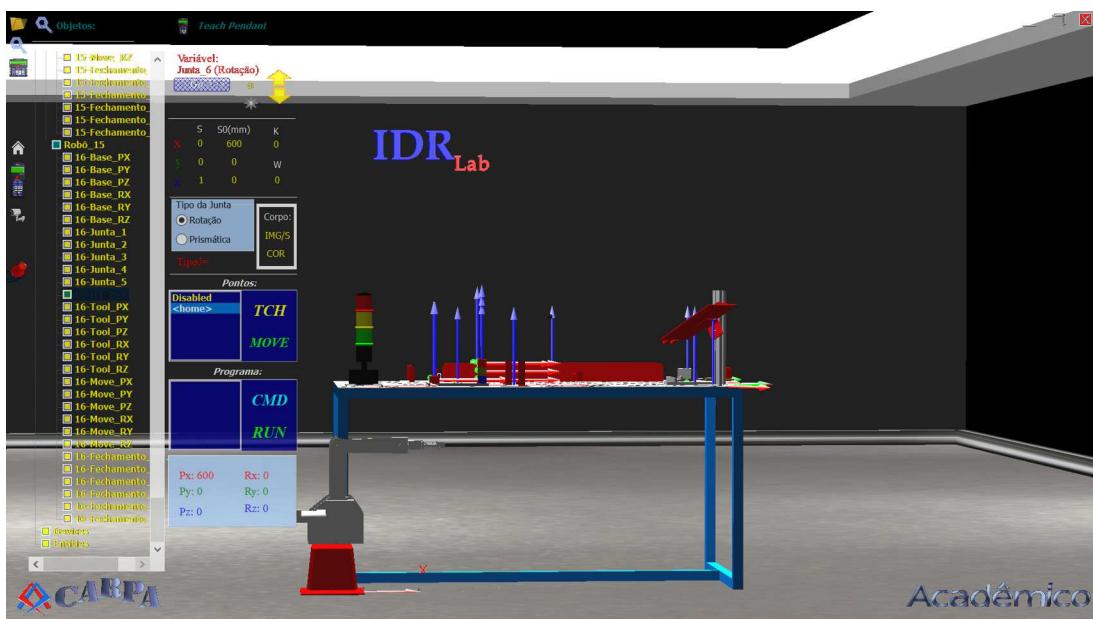


FIG. 4.24: Associar cor ao objeto: escolher as tonalidade de cor (Autoria Própria)

4.3.4 POSICIONAR OBJETOS

Além do que foi apresentado acima, o CARPA permite posicionar o robô no espaço tridimensional, como o usuário desejar. Dessa maneira, após verificada a posição do robô no sistema

real, foi realizado o correto posicionamento do robô virtual, definindo, por exemplo, os valores de P_z igual 838,53 mm (FIG. 4.25), P_y igual 455,3 mm (FIG. 4.26) e P_x igual a 547,92 mm (FIG. 4.27), os quais representam o deslocamento ao longo de seus respectivos eixos . Porém, poderia ainda realizar rotações em torno de cada eixo, em R_x , R_y e R_z , caso fosse necessário.

A fase de posicionamento é uma necessidade de todo *software* que pretende representar um cenário real por modelos, visto que a correta localização é fundamental para os passos de geração automática do código.

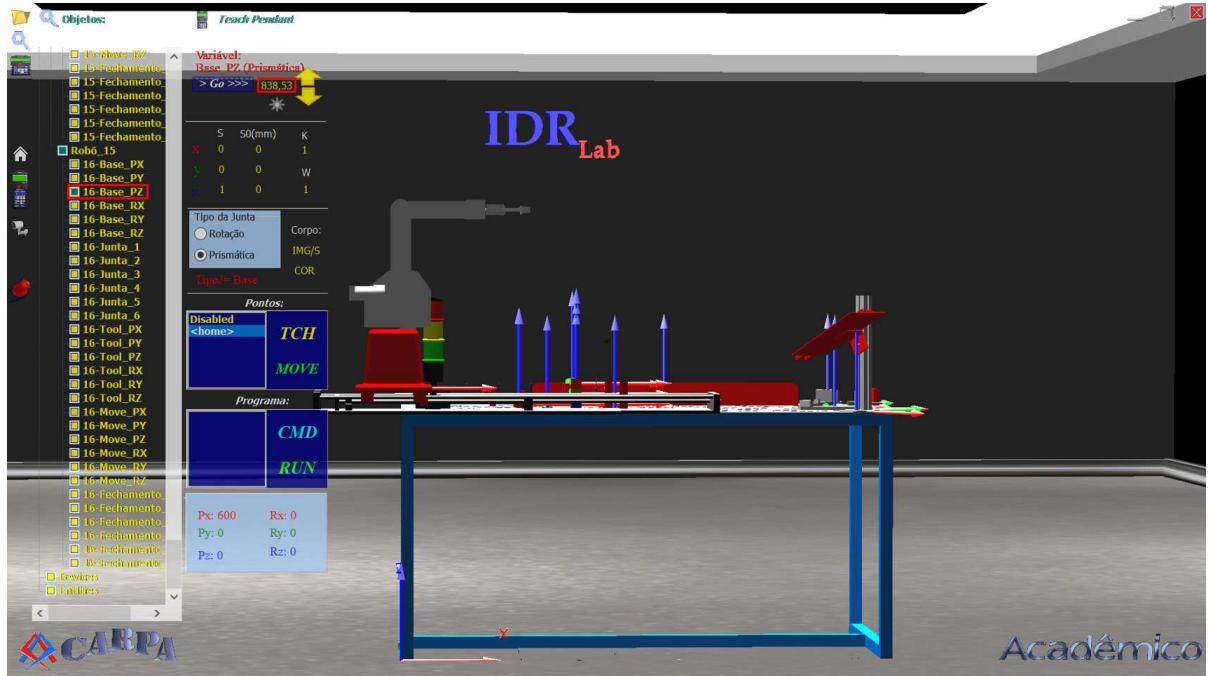


FIG. 4.25: Posicionar um robô inserido: definindo o valor de P_z (Autoria Própria)

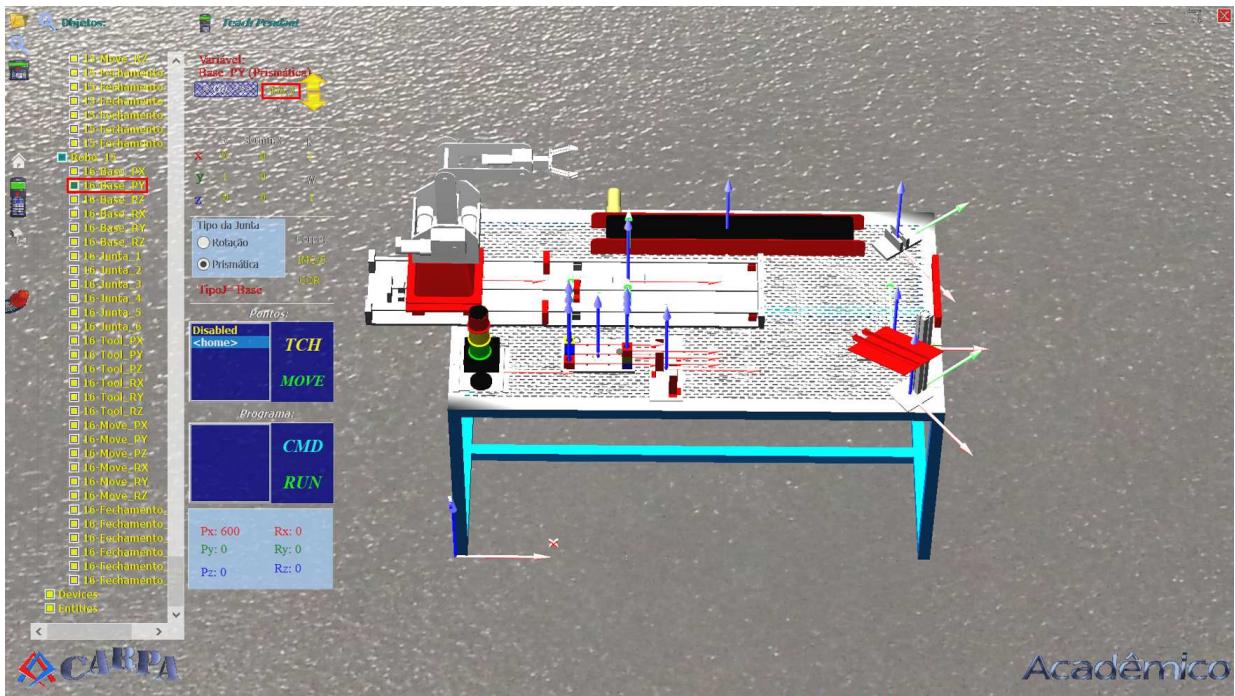


FIG. 4.26: Posicionar um robô inserido: definindo o valor de P_y (Autoria Própria)

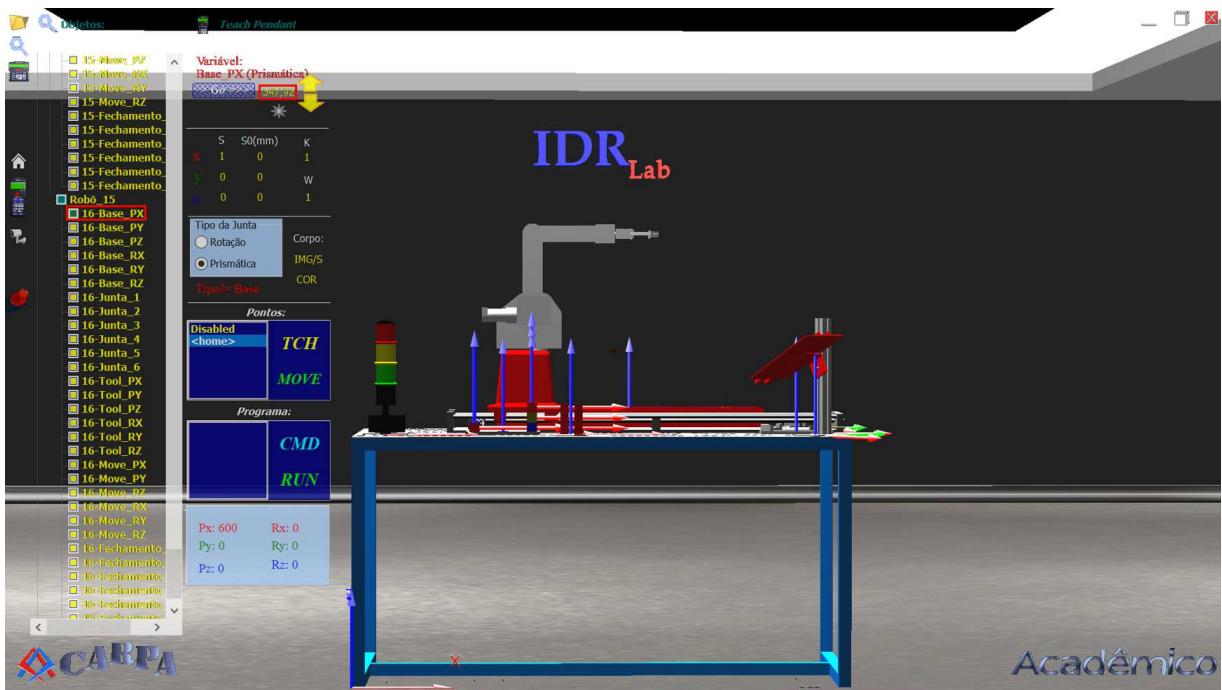


FIG. 4.27: Posicionar um robô inserido: definindo o valor de P_x (Autoria Própria)

4.3.5 PROGRAMAÇÃO DE UM ROBÔ

Antes da criação do CARPA, foi necessário conhecer qual é a linguagem do robô real a ser programado virtualmente, visto que, como já abordado anteriormente, a programação é dependente da linguagem do robô, ou seja, para que a mesma seja possível, é necessário um conhecimento prévio da linguagem de um robô real, a fim de gerar o(s) arquivo(s) de programação. Para isso, foi escolhido o robô Pegasus da Amatrol, com linguagem MCL, visto que é o robô que se encontra instalado no IDR Lab. Dessa forma, diversos estudos e atividades foram feitos, utilizando esse robô, a fim de conhecer com profundidade as características dessa linguagem e poder implementá-la computacionalmente. A partir disso, pode-se apresentar uma das principais características do CARPA, a programação específica desse robô.

O primeiro passo é definir e ensinar os pontos, pelos quais o robô irá percorrer (FIG. 4.28 à 4.30). Em seguida, abre-se uma tela com os principais comandos implementados do robô real (FIG. 4.31). Uma programação simples foi definida, em que a primeira linha de código é referente ao comando de ligar o "*output 16*" (liga ou desliga uma saída digital. A saída permanece neste estado até ser mudada por um outro comando "WRITEO"), o qual é representado, pela linguagem MCL, como "Writeo Output_16, On", como mostrado nas FIG. 4.32 (a) e (b).

Na sequência, foram inseridos no programa os pontos ensinados "*Point_1*", utilizando o código da linguagem MCL "Pmove Point_1"; *Point_2*, por meio de "Pmove Point_2"; *Point_3*, por "Pmove Point_3"; e *Point_4*", por "Pmove Point_4", como pode ser verificado nas FIG. 4.33 (a) e (b). O comando "Pmove" move o robô para um ponto, através de uma trajetória desconhecida.

Por fim, finaliza-se a atividade, inserindo o comando de desligar o "*output 16*", de forma análoga a como foi feito para ligá-lo, sendo representado por "Writeo Output_16, Off", como consta na FIG. 4.33 (b).

Alguns outros comandos, que estão no CARPA e não foram utilizados, são expostos e explicados a seguir:

- GRASP

- Fecha a garra do robô;
 - Sintaxe: GRASP.

- RELEASE

- Abre a garra do robô;

- Sintaxe: RELEASE.
- LMOVE
 - Faz com que o robô se movimente a um ponto específico, utilizando uma trajetória em linha reta. Não realiza o movimento do eixo transversal;
 - Sintaxe: LMOVE Point_1, ou seja, move o robô para o ponto 1 da lista de pontos.
- SPEED
 - Altera a velocidade do robô para todos os comandos subsequentes;
 - Sintaxe: SPEED 50, ou seja, a velocidade do robô de um ponto ao outro é igual a 50. Vale ressaltar que a velocidade varia de 0 a 255.
- WAITI
 - Espera uma entrada digital ser igual a um determinado valor;
 - Sintaxe: WAITI entrada, valor. "Entrada" é o nome ou o número de uma entrada, enquanto "valor" é o valor a ser comparado. O programa irá parar de executar até o estado da entrada igualar o valor determinado. Exemplo: WAITI 12, 0, ou seja, espera até a entrada 12 estar desligada.

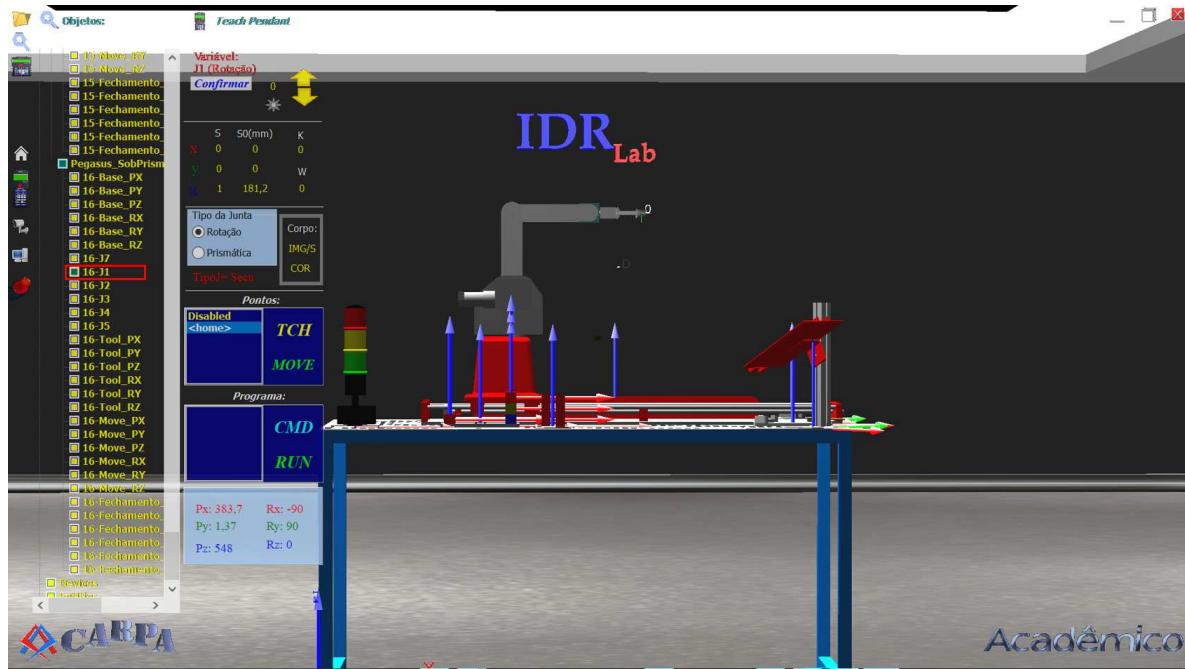


FIG. 4.28: Ensinando o ponto 0 (zero) (Autoria Própria)

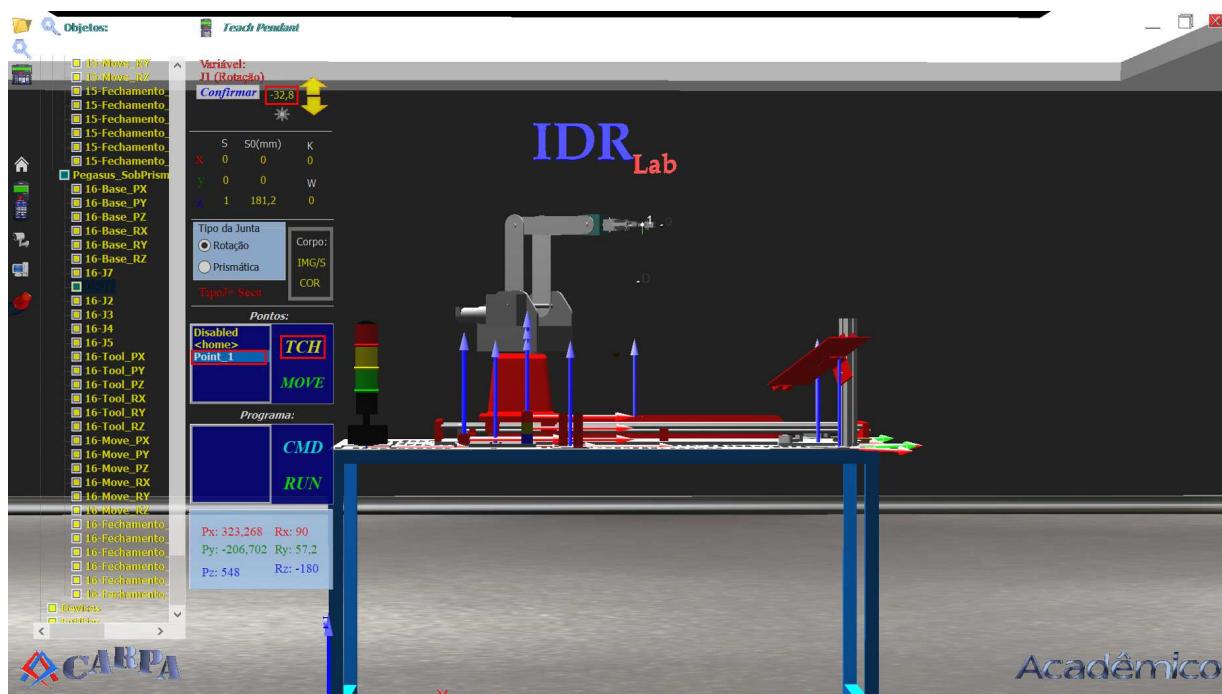


FIG. 4.29: Ensinando o ponto 1 (Autoria Própria)

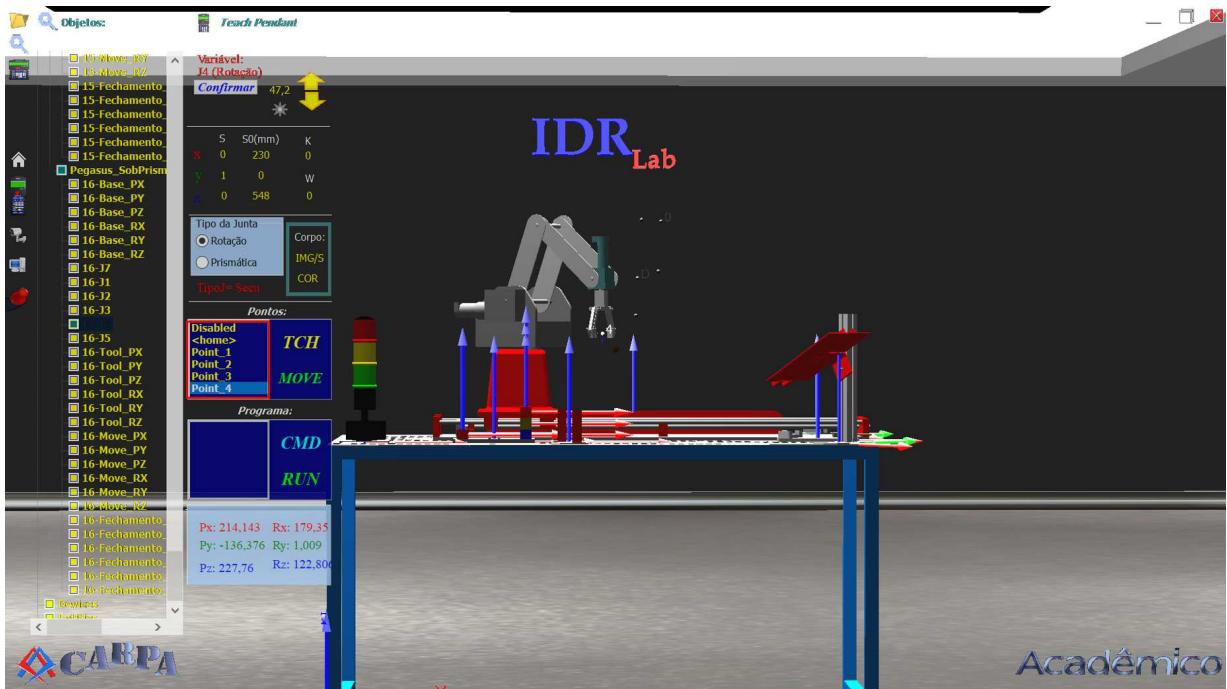


FIG. 4.30: Ensinando os pontos 2, 3 e 4 (Autoria Própria)



FIG. 4.31: Tela com os códigos implementados do robô Pegasus (Autoria Própria)

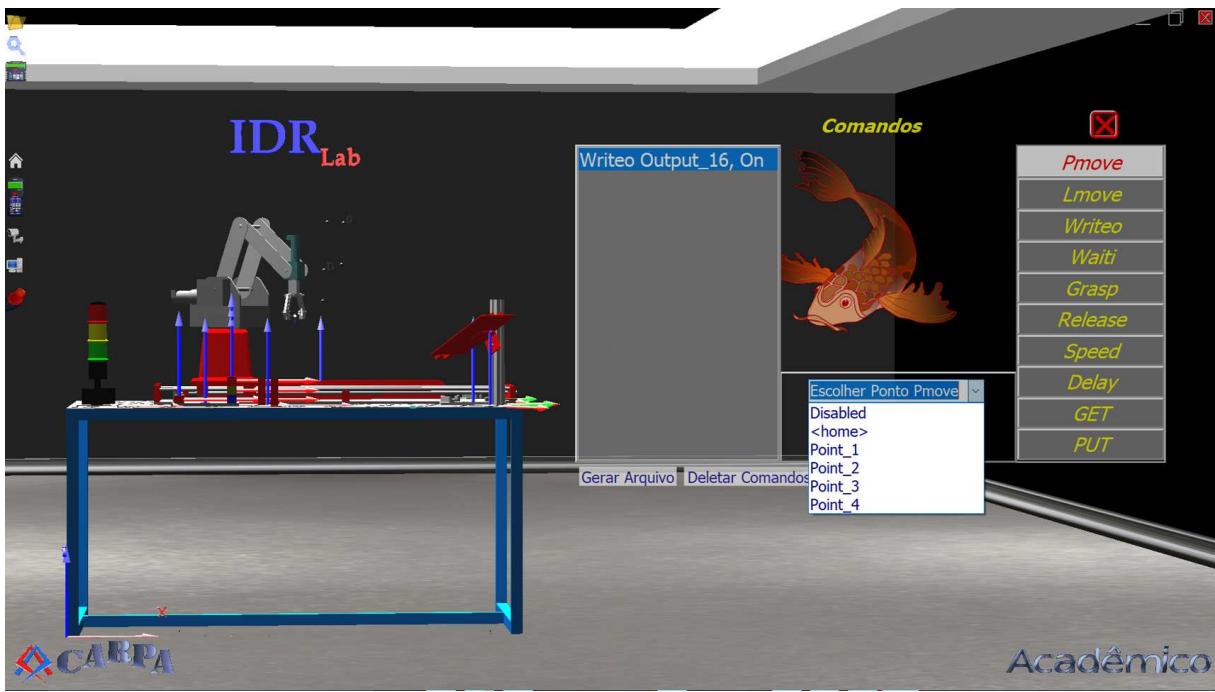


(a)



(b)

FIG. 4.32: Comando de *output* do robô Pegasus (Autoria Própria)



(a)



(b)

FIG. 4.33: Procedimento para programação do robô Pegasus (Autoria Própria)

Após todas as etapas anteriores terem sido finalizadas, é necessário salvar um arquivo, em que estejam contidas todas as linhas de código da programação. Para isso, deve-se clicar em "Gerar Arquivo", o que abrirá uma tela para escolher o local onde será salvo o arquivo, nomeá-lo e

salvar, conforme FIG. 4.34 (a). Na FIG. 4.34 (b), é possível verificar o arquivo com toda a programação salva na sequência pré-definida na linguagem MCL.

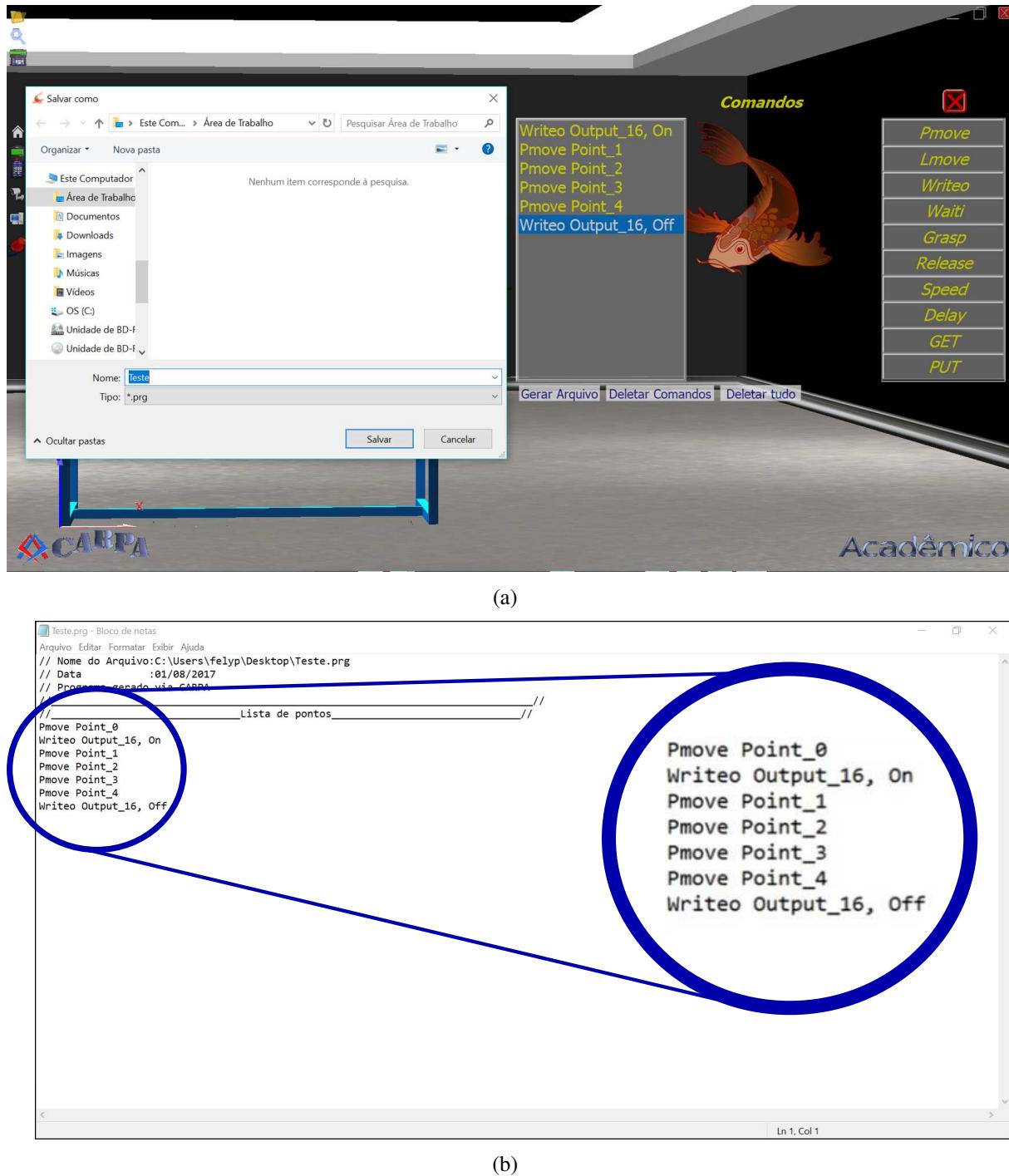


FIG. 4.34: Procedimento para salvar o programa do robô Pegasus (Autoria Própria)

5 CONCLUSÕES E PERSPECTIVAS

Neste capítulo, são apresentadas, formalmente, as conclusões da dissertação, iniciando com a seção 5.1 que apresenta um resumo dos capítulos anteriores, a fim de ressaltar os argumentos e fundamentos que resultaram na PD&I em Robótica Industrial como tecnologia dual, as vantagens e a complexidade associada a sua utilização, o que motivou a PD&I desta dissertação. Em seguida, na seção 5.2, são ressaltados os resultados obtidos, com as contribuições científicas desta dissertação. Por fim, na seção 5.3, são expostas as conclusões finais, apresentando as perspectivas para trabalhos futuros.

5.1 RECAPITULAÇÃO SINTETIZADA

De modo a ratificar e fundamentar a escolha da área de concentração da pesquisa, o Capítulo 1 apresentou o contexto, no qual essa dissertação está inserida; a importância da Robótica Industrial como Tecnologia Dual, auferindo vantagens tanto para o emprego civil como militar; levantou-se as desvantagens associadas ao emprego dos manipuladores industriais na automação dos processos de fabricação, resultando na formulação do problema que a dissertação visa contribuir com uma solução, que consiste, em última análise, diminuir o tempo e a complexidade associados à programação de robôs, pela modelagem, simulação e resolução cinemática de um manipulador industrial genérico, através de uma metodologia auxiliada por computador, de maneira amigável, consistente e robusta, além de ser de fácil entendimento para o usuário.

Após a etapa de determinação do problema, fez-se, no Capítulo 2, uma revisão bibliográfica, apresentando: a evolução histórica da robótica industrial, a fim de mostrar o seu surgimento e, posteriormente, o seu desenvolvimento; campos de aplicação da robótica; definição e classificação de robôs industriais; uma conjuntura, em termos quantitativos, em que a robótica industrial se encontra, tanto no Brasil, quanto no Mundo, fazendo um comparativo entre os dois cenários; tipos de programação de manipuladores, ressaltando suas vantagens e desvantagens; e, finalmente, os *softwares* robóticos existentes;

No Capítulo 3, foram apresentadas duas metodologias importantes (a robótica baseada na Teoria dos Helicoides e a Convenção de Denavit-Hartenberg). Foram apresentados os procedimentos de cada método, de forma genérica e específica, utilizando o robô Pegasus, a fim de operacionalizar a determinação da cinemática direta de um robô industrial, mostrando suas

vantagens e desvantagens.

O Capítulo 4 apresentou o sistema CARPA, desenvolvido para o sistema operacional Windows, seguindo a filosofia de programação orientada a eventos, ressaltando as funcionalidades disponíveis, exemplificando como implementar um novo robô. Segue a particularização do sistema CARPA para o Sistema Flexível de Manufatura, contendo o robô Pegasus da Amatrol, que é o primeiro sistema de aprendizagem em robótica disponível no IDR Lab, onde, em 2017, todos os alunos do 5º ano de graduação em Engenharia Mecânica do IME foram treinados. O CARPA-PEGASUS resultou em um simulador intitulado CAPRON, que foi experimentado e mostrou efetividade na fase inicial de treinamento para redução do tempo de utilização com o equipamento real, pelo fato de facilitar e fixação do conhecimento pelo aprendiz das funcionalidades do TP na movimentação do Pegasus.

No Apêndice 7.1, foi feita uma revisão dos fundamentos da Teoria dos Helicoides aplicada à robótica, base para a modelagem e resolução cinemática de robôs seriais genéricos, que recebeu implementação computacional, além de permitir a geração automatizada dos arquivos de programa específicos para o robô Pegasus.

5.2 CONTRIBUIÇÕES CIENTÍFICAS

Diante de um cenário de mercados globalizados e de alta competitividade entre produtos, a implementação eficiente da automação, utilizando robôs industriais, ganha relevância. Surge uma necessidade crescente de difundir e aumentar a quantidade e a qualidade de pessoal envolvido, qualificado e consciente da importância do robô industrial para o aumento da robustez dos produtos fabricados, favorecendo uma base consistente para aumentar a quantidade e a eficiência de robôs instalados no país, de modo a auferir vantagens estratégicas às indústrias.

Esta dissertação apresenta como contribuições a obtenção de uma metodologia de modelagem, simulação, análise e programação de robôs industriais seriais, tendo como base a Teoria dos Helicoides, implementada computacionalmente, o que resultou no protótipo CARPA (*Computer-Aided Serial Robots Programming, Modelling and Simulation Analysis*), com os seguintes requisitos:

- ser um ambiente amigável de modelagem e simulação de robôs, dispensando a necessidade de um conhecimento aprofundado por parte do usuário/programador da Teoria dos Helicoides aplicada à robótica;

- ser genérico, ou seja, que permita modelar, simular e analisar qualquer robô industrial serial; e,
- possuir código aberto, orientado a objetos e, devidamente, documentado, de modo a permitir a sua manutenção e o desenvolvimento de trabalhos futuros.
- facilita a modelagem, simulação e análise integradas de cenários contendo robôs industriais, de forma mais simples e intuitiva possível sem perda de características de robustez e generalidade;
- reduz o tempo necessário de programação *on-line* utilizando o equipamento real, pois todas as etapas de planejamento podem ser simuladas em ambiente virtual;
- apresenta resultados promissores em aumento de motivação e de efetividade na redução do tempo necessário ao treinamento de aprendizes, que antes utilizavam somente o equipamento real; e,
- possibilita ser utilizado como simulador das funcionalidades do TP, de modo a prover movimentação em modelos dos robôs, favorecendo o treinamento e reduzindo o tempo com o equipamento real.

Cabe citar que, no decorrer desta dissertação, foram feitas 3 publicações (SANTOS et al., 2016), (SANTOS et al., 2017b) e (SANTOS et al., 2017a), as quais serviram de fundamento para a pesquisa realizada.

5.3 CONCLUSÃO

Os requisitos supracitados foram alcançados e são estratégicos para a continuidade da PD&I no IDR Lab.

O protótipo CARPA permite que novos robôs sejam modelados, necessitando que usuário defina: o número de juntas e os vetores s e s_0 , ou seja, vetor direção do eixo da junta e vetor posição de um ponto qualquer pertencente ao eixo da junta em relação ao referencial de base, respectivamente; além do arquivo em CAD 3D de cada corpo no formato 3DS.

Toda a movimentação fica automaticamente determinada, de forma transparente ao usuário, bastando que o mesmo modifique o valor da variável de junta que deseja mover. Recursos de visualização 3D ficam disponibilizados, podendo girar o cenário para obter diversas vistas e aplicar *zoom* nos modelos que estão sendo apresentados na tela do computador. Pontos podem

ser ensinados e a movimentação entre os mesmos pode ser feita pelo clique do mouse em um botão.

Diversos robôs podem ser incluídos simultaneamente, bem como objetos de cenário. Cada um pode ser localizado adequadamente, de modo a representar o *layout* real. Com isso, o cenário virtual torna-se um modelo geométrico do real, com o mínimo de abstração.

A versão atual do CARPA foi concebida com, aproximadamente, 19.000 linhas de código e 3.000 horas-homem de implementação computacional.

5.4 PERSPECTIVAS

Com continuidade de PD&I voltada ao aperfeiçoamento do CARPA, algumas funcionalidades poderão ser adicionadas, como a identificação de colisão, o que permitirá torná-la uma ferramenta de aprendizagem mais útil, facilitadora e multiplicadora do conhecimento em Robótica Industrial. Além disso, poderá ser disponibilizada para Instituições de Ensino Fundamental, Médio e Superior, em níveis de dificuldades compatíveis e progressivos a cada uma dessas fases do ensino, motivando o gosto pela pesquisa em uma tecnologia que tem tudo para ocupar posição de destaque no futuro.

Outra demanda que pode ser atendida é estabelecer parcerias com Pequenas e Médias Empresas (PME), fortalecendo ainda mais os elos entre as instituições de ensino e as indústrias, além de poder ser empregado em aplicações duais, tanto para as indústrias, meio civil, quanto ao militar. Pelo fato do código aberto estar disponível, pode ser adaptado e customizado conforme a necessidade de cada parceiro.

Igualmente importante é permitir estreitar parcerias com grupos de pesquisas de outras Instituições de Ensino Superior (IES), permitindo que protocolos de comunicação de dados sejam criados, tornando o CARPA de uso ainda mais amplo.

Finalmente, cabe ressaltar que o resultado obtido nesta dissertação já está sendo utilizado por alunos de pós-graduação, com devidas adaptações para modelagem e simulação de Sistemas Robóticos Cooperativos e para criação do simulador de treinamento do robô industrial Motoman MH5F, disponível no Laboratório de Mecatrônica do IME.

6 REFERÊNCIAS BIBLIOGRÁFICAS

- ABB-BRASIL. Quem somos. Disponível em: <<http://new.abb.com/br/empresa/brasil>>. Acesso em 09/08/2016, 2016.
- AMATROL. Operação de equipamento automatizado 1. desenvolvimento de aplicações. 1. ed. 2007a.
- AMATROL. Operação de equipamento automatizado 1. programação básica de um robô. 1. ed. 2007b.
- BONFIM, M. Code kata: How to become a better developer, abril 2016. URL <http://www.devmedia.com.br/o-que-e-o-coding-dojo/30517>.
- COPPELIA-ROBOTICS. Acknowledgments and credits. disponível em: <<http://www.coppeliarobotics.com/helpfiles/en/acknowledgments.htm>>. Acesso em 03/10/2016, 2016a.
- COPPELIA-ROBOTICS. Disponível em: <<http://www.coppeliarobotics.com/v-repoviewpresentation.pdf>>. Acesso em: 05/10/2016, 2016b.
- CRAIG, J. J. Introduction to robotics: mechanics and control, volume 3. Upper Saddle River: Pearson Prentice Hall, 2005.
- END. Estratégia nacional de defesa. Ministério da Defesa. Brasília, 2012.
- FANUC. História da fanuc. disponível em: <<http://www.fanuc.eu/pt/pt/quem-somos/fanuc-history>>. acesso em 09/08/2016. 2016.
- FREES, M., ROHMER, E. e SINGH, S. P. N. Vrep: a versatile and scalable robot simulation framework. Proc. of The International Conference on Intelligent Robots and Systems (IROS), 2013.
- IFR. Executive summary: Industrial robots. Disponível em: <<http://www.ifr.org/downloads/>>. Acesso em 15/11/2015, 2015.
- IFR. Executive summary world robotics 2016 industrial robots. Disponível em: <<http://www.ifr.org/downloads/>>. Acesso em 27/10/2016, 2016.
- MACEDO, E. e RIBEIRO, L. P. Robótica industrial: vantagem estratégica e desafios. Revista Sodebras [on line]. v. 10, n.116, Ago./2015, p. 139-144. ISSN 1809-3957, 2015.
- MOTOMAN. Decades among the leading robotics companies. disponível em: <<http://www.motoman.com/brasil>>. acesso em 09/08/2016. 2016.
- MURRAY, R. M., LI, Z., SASTRY, S. S. e SASTRY, S. S. A mathematical introduction to robotic manipulation. California, USA: CRC press, 1994.

- NOF, S. Y. *Handbook of industrial robotics*. Indiana, USA: John Wiley & Sons, 1999.
- PAN, Z., POLDEN, J., LARKIN, N., VAN DUIN, S. e NORRISH, J. Recent progress on programming methods for industrial robots. *Robotics and Computer-Integrated Manufacturing*, 28(2):87–94, 2012.
- PND. Política nacional de defesa. Ministério da Defesa. Brasília, 2012.
- RIA. Robot institute of america. disponível em: <<http://www.fem.unicamp.br/hermini/robotica/livro/cap.1.pdf>>. Acesso em 25/11/2015, 2015.
- RIBEIRO, L. P. G. Modelagem cinemática de sistemas robóticos cooperativos: Proposta de um jacobiano de cooperação. 2010.
- ROCHA, C., TONETTO, C. e DIAS, A. A comparison between the denavit-hartenberg and the screw-based methods used in kinematic modeling of robot manipulators. *Robotics and Computer-Integrated Manufacturing*, 2011.
- SANTOS, F. L. M., SANTOS, R. R., DE AZEVEDO MACEDO, F. A., DE OLIVEIRA MARQUES, L. e RIBEIRO, L. P. G. Capron: Avaliação da eficiência do simulador na aprendizagem de programação on-line do robô pegasus. *Revista Sodebras [on line]*. v. 12, n.141, Set./2017, p. 193-198. ISSN 1809-3957, 2017a.
- SANTOS, F. L. M., SANTOS, R. R. e RIBEIRO, L. P. G. Fatores de influência da programação on-line no tempo de execução de uma tarefa realizada pelo robô pegasus. *Revista Sodebras [on line]*. v. 11, n.129, Set./2016, p. 174-179. ISSN 1809-3957, 2016.
- SANTOS, R. R., DE AZEVEDO MACEDO, F. A., SANTOS, F. L. M. e RIBEIRO, L. P. G. O uso de cad 3d no planejamento da programação on-line de uma operação de montagem usando robô pegasus. *Revista Sodebras [on line]*. v. 12, n.135, mar/2017, p. 185-190. ISSN 1809-3957, 2017b.
- SATO, D. T., CORBUCCI, H. e BRAVO, M. V. Coding dojo: An environment for learning and sharing agile practices. Agile, 2008. AGILE'08. Conference. IEEE, 2008.
- SICILIANO, B. e KHATIB, O. *Springer Handbook of Robotics*. California, USA: Springer Science & Business Media, 2008.
- SICILIANO, B., SCIavicco, L., VILLANI, L. e ORIOLO, G. *Robotics: Modelling, Planning and Control*. London, England: Springer Science & Business Media, 2009.
- SIEMENS. Indústria 4.0 exigirá um novo profissional. disponível em: <<http://exame.abril.com.br/publicidade/siemens/conteudo-patrocinado/industria-4-0-exigira-um-novo-profissional>>. Acesso em 13/06/2016, 2015.
- SLACK, N., CHAMBERS, S. e JOHNSTON, R. Administração da Produção. São Paulo: Atlas, 2002.
- TENENBAUM, R. A. *Fundamentals of Applied Dynamics*. New York, USA: Editora Springer, 1997.

TSAI, L.-W. Robot analysis: the mechanics of serial and parallel manipulators. Maryland, USA: John Wiley & Sons, 1999.

UNECE. World robotics survey. disponível em: <<http://www.unece.org/fileadmin/dam/press/pr2004/04stat-p01e.pdf>>. Acesso em 02/06/2016, 2004.

VENTURELLI, M. Indústria 4.0: Uma visão da automação industrial. disponível em: <<http://www.automacaoindustrial.info/industria-4-0-uma-visao-da-automacao-industrial/>>. Acesso em 13/06//2016., 2014.

YANG, C., MA, H. e FU, M. Advanced Technologies in Modern Robotic Applications. Beijing, China: Science Press and Springer Science+Business Media Singapore, 2016.

7 APÊNDICES

7.1 APÊNDICE 1: ROBÓTICA BASEADA EM HELICOIDES

Neste apêndice, é feita uma revisão dos fundamentos teóricos necessários à modelagem e resolução cinemática de um robô serial, valendo-se da Teoria dos Helicoides. Inicialmente é apresentado o modelo matemático de representação da localização de um corpo rígido no espaço. Em seguida, é apresentada a modelagem de juntas elementares, baseada em Helicoides. Posteriormente, faz-se a representação da rotação em torno de um ponto fixo por meio de eixo de helicoide seguido do deslocamento helicoidal. Ainda é descrito o Método dos Deslocamentos dos Helicoides Sucessivos para determinação da cinemática direta de um manipulador serial. Finalmente, é feita a determinação do eixo do helicoide para um movimento instantâneo genérico, o que permite analisar o Jacobiano baseado em helicoides.

7.1.1 LOCALIZAÇÃO DE UM CORPO RÍGIDO

Antes de iniciar qualquer tratativa de localização de um corpo rígido no espaço tridimensional, nessa seção, é pertinente definir um corpo rígido, conforme a seguir.

Definição 7.1.1 – *Corpo Rígido – sistema contínuo, tal que a distância, entre dois pontos quaisquer, pertencentes a esse corpo, não varia com o tempo. Dessa forma, o corpo rígido é um modelo da mecânica que constitui uma abstração, tal que suas deformações, ou seja, os deslocamentos relativos possam ser desprezados frente a seu movimento global (TENENBAUM, 1997).*

A localização de um corpo rígido é determinada, em relação ao sistema de eixos coordenados de referência, quando a posição de todos os pontos desse corpo for conhecida no espaço tridimensional. Para isso, é necessária a determinação de seis parâmetros independentes, segundo TSAI (1999), os quais são medidos em função de um referencial. Assim, deve-se verificar as seguintes definições.

Definição 7.1.2 – *Referencial – é definido como um conjunto de pontos não colineares, que guardam entre si distâncias invariantes com o tempo, podendo se associar a um sistema de eixos cartesianos (TENENBAUM, 1997).*

Definição 7.1.3 – Referencial Absoluto – é definido como um sistema fixo de eixos coordenados de referência, permitindo a localização de um corpo rígido no espaço tridimensional (TENENBAUM, 1997).

Definição 7.1.4 – Referencial Relativo ou Móvel – é definido como um sistema de eixos coordenados que se movimenta, em relação ao referencial fixo ou a outro referencial móvel, sendo útil na determinação da localização de sistemas com múltiplos corpos ligados por juntas. Esses sistemas coordenados se movem usualmente solidários a elos.

A FIG. 7.1 mostra o sistema de eixos coordenados (x, y, z) assumido como referencial absoluto (A) e o sistema (u, v, w) como referencial móvel (B).

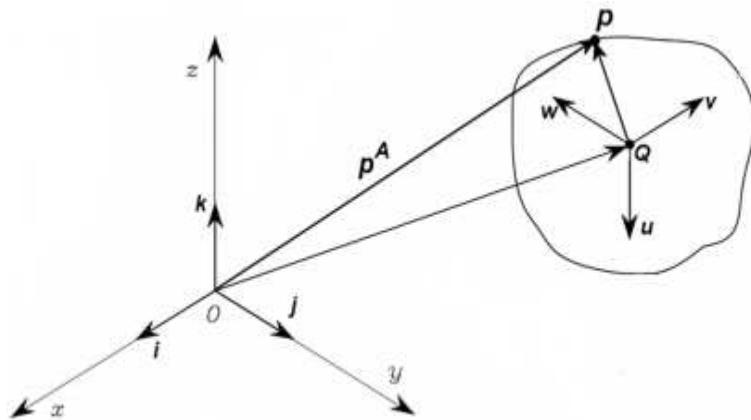


FIG. 7.1: Referenciais absolutos e móveis (TSAL, 1999)

As posições de todos os pontos pertencentes ao corpo rígido podem ser determinadas quando se conhece a localização do referencial móvel, em relação ao referencial absoluto. Essa localização relativa pode ser considerada como a composição da **posição** do ponto Q que é a origem do referencial móvel, com a **orientação** deste referencial em relação ao referencial absoluto.

Para facilitar a análise, em um primeiro momento, considere que os dois referenciais tenham origens coincidentes. Ao analisar a FIG. 7.1, a posição do ponto P , com respeito ao referencial absoluto, pode ser descrita por um vetor (3×1) , no seguinte modo:

$$p^A = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} \quad (7.1)$$

Notação 7.1.1 – Cabe ressaltar que a representação com um super-índice A significa que o vetor p tem coordenadas do ponto P relativas ao referencial A .

7.1.1.1 TEOREMA DE EULER - REPRESENTAÇÃO DA ORIENTAÇÃO

Na orientação de um corpo rígido, um referencial móvel é associado ao corpo rígido , visto que acompanha os movimentos desse corpo, sendo solidário ao corpo. Na representação da orientação de um corpo rígido, considera-se que a movimentação entre o referencial móvel e o absoluto se dá com um ponto em comum (0), ou seja, a origem em comum. Isto é conhecido como rotação ou movimento esférico, conforme mostrado na FIG. 7.2.

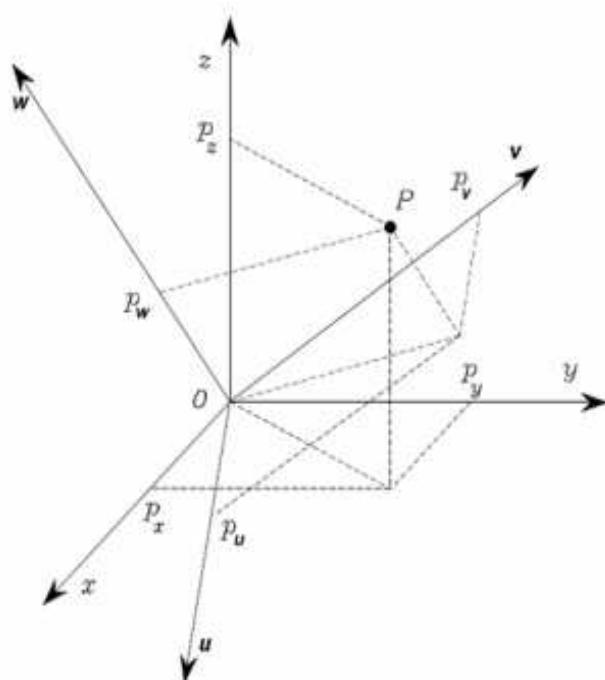


FIG. 7.2: Representação de P em dois referenciais com origem coincidente (SICILIANO et al., 2009)

A orientação pode ser representada de diversas maneiras, tais como (TSAI, 1999): por cossenos diretores, por eixos de helicoides ou por ângulos de Euler.

Assumindo-se que o referencial absoluto (A) possua vetores unitários (i, j, k) e o móvel (B), vetores unitários (u, v, w) , é possível representar (u, v, w) em função de (i, j, k) , conforme a seguir:

$$u^A = u_x i + u_y j + u_z k \quad (7.2)$$

$$v^A = v_x i + v_y j + v_z k \quad (7.3)$$

$$w^A = w_x i + w_y j + w_z k \quad (7.4)$$

É possível representar o ponto P em ambos os referenciais:

$$p^A = p_x i + p_y j + p_z k \quad (7.5)$$

$$p^B = p_u u + p_v v + p_w w \quad (7.6)$$

Substituindo os vetores unitários (u, v, w) , em função das suas projeções no referencial fixo (i, j, k) , passa-se a ter o valor do vetor p nesse último referencial, ou seja:

$$p^A = p_u(u_x i + u_y j + u_z k) + p_v(v_x i + v_y j + v_z k) + p_w(w_x i + w_y j + w_z k) \quad (7.7)$$

$$p^A = (p_u u_x + p_v v_x + p_w w_x)i + (p_u u_y + p_v v_y + p_w w_y)j + (p_u u_z + p_v v_z + p_w w_z)k \quad (7.8)$$

Dessa maneira, comparando as Eqs. 7.6 e 7.8, é possível escrever:

$$p_x = p_u u_x + p_v v_x + p_w w_x \quad (7.9)$$

$$p_y = p_u u_y + p_v v_y + p_w w_y \quad (7.10)$$

$$p_z = p_u u_z + p_v v_z + p_w w_z \quad (7.11)$$

Agrupando na forma matricial, tem-se:

$$\begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} = \begin{bmatrix} u_x & v_x & w_x \\ u_y & v_y & w_y \\ u_z & v_z & w_z \end{bmatrix} \begin{bmatrix} p_u \\ p_v \\ p_w \end{bmatrix} \quad (7.12)$$

Assim, todo deslocamento de um corpo rígido sobre um ponto fixo pode ser representado como uma rotação em torno de um único eixo e pode ser determinado através de uma transformação ortogonal que é a própria Matriz de Rotação, o que é conhecido como Teorema de Euler, podendo ser representado da seguinte forma.

$$p^A = R_B^A \ p^B \quad (7.13)$$

onde,

$$R_B^A = \begin{bmatrix} u_x & v_x & w_x \\ u_y & v_y & w_y \\ u_z & v_z & w_z \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \quad (7.14)$$

A matriz R_B^A é denominada Matriz de Rotação que move o referencial B em relação ao referencial A , representando completamente a orientação de B em relação a A , sendo formada por 3 vetores coluna unitários ortogonais, o que faz com que a mesma seja ortonormal, em que as seguintes condições podem ser aplicadas:

$$u^2 = u^T u = 1 \quad (7.15)$$

$$v^2 = v^T v = 1 \quad (7.16)$$

$$w^2 = w^T w = 1 \quad (7.17)$$

$$u^T v = v^T w = w^T u = 0 \quad (7.18)$$

$$u \times v = w = -(v \times u) \quad (7.19)$$

$$v \times w = u = -(w \times v) \quad (7.20)$$

$$w \times u = v = -(u \times w) \quad (7.21)$$

$$\det(R_B^A) = 1 \quad (7.22)$$

$$R_B^A = \begin{bmatrix} u^A & v^A & w^A \end{bmatrix} = \begin{bmatrix} i^{B^T} \\ j^{B^T} \\ k^{B^T} \end{bmatrix} \quad (7.23)$$

$$R_A^B = R_B^{A^{-1}} = R_B^{A^T} \quad (7.24)$$

7.1.1.2 TEOREMA DE CHASLES – REPRESENTAÇÃO DA LOCALIZAÇÃO

Admite-se, pelo Teorema de Chasles, que todo deslocamento espacial de um corpo rígido pode ser representado pela soma vetorial das parcelas de: (1) uma rotação em torno do eixo do helicoide e (2) uma translação ao longo do eixo do helicoide.

$$p^A = R_B^A p^B + q^A \quad (7.25)$$

$$\begin{bmatrix} p_x^A \\ p_y^A \\ p_z^A \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} p_x^B \\ p_y^B \\ p_z^B \end{bmatrix} + \begin{bmatrix} q_x^A \\ q_y^A \\ q_z^A \end{bmatrix} \quad (7.26)$$

Assim, o Teorema de Chasles pode ser representado, de forma compacta, pela Matriz de Transformação Homogênea (4x4).

$$T_B^A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & q_x^A \\ a_{21} & a_{22} & a_{23} & q_y^A \\ a_{31} & a_{32} & a_{33} & q_z^A \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} R_B^A & q^A \\ 0 & 1 \end{bmatrix} \quad (7.27)$$

O vetor $p^A = [p_x^A \ p_y^A \ p_z^A]^T$ ganha uma dimensão, sendo representado na forma homogênea $\tilde{p}^A = [p_x^A \ p_y^A \ p_z^A \ 1]^T$. Assim sendo, tem-se:

$$\tilde{p}^A = T_B^A \tilde{p}^B \quad (7.28)$$

A matriz de transformação homogênea possui inversa, embora não seja ortogonal, ou seja:

$$[T_B^A]^{-1} \neq [T_B^A]^T \quad (7.29)$$

A determinação de T_B^A pode ser feita pré-multiplicando a EQ. 7.25 pela inversa da matriz de rotação, ou seja:

$$[R_B^A]^{-1} p^A = [R_B^A]^{-1} R_B^A p^B + [R_B^A]^{-1} q^A \quad (7.30)$$

Visto que a matriz de rotação é ortogonal, sua inversa é igual a sua transposta (EQ. 7.24), o que permite que a EQ. 7.30 seja reescrita, isolando p^B , como:

$$p^B = [R_B^A]^T p^A - [R_B^A]^T q^A \quad (7.31)$$

Adotando a forma homogênea \tilde{p} do vetor p , a EQ. 7.31 pode ser reescrita como:

$$\tilde{p}^B = T_B^{A^{-1}} \tilde{p}^A \quad (7.32)$$

onde,

$$[T_B^A]^{-1} = \begin{bmatrix} [R_B^A]^T & -[R_B^A]^T q^A \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{21} & a_{31} & -a_{11}q_x^A - a_{21}q_y^A - a_{31}q_z^A \\ a_{12} & a_{22} & a_{32} & -a_{12}q_x^A - a_{22}q_y^A - a_{32}q_z^A \\ a_{13} & a_{23} & a_{33} & -a_{13}q_x^A - a_{23}q_y^A - a_{33}q_z^A \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (7.33)$$

7.1.2 TEORIA DOS HELICOIDES

Conforme RIBEIRO (2010), considera-se o helicoide um ente geométrico definido por:

1. uma reta no espaço que representa a direção do seu eixo, designada pelo respectivo vetor unitário s ;
2. um vetor s_0 , ligando a origem do referencial a qualquer ponto pertencente ao eixo do helicoide;
3. um ângulo θ , cuja variação caracteriza o movimento de rotação em torno do eixo do helicoide; e,
4. um comprimento d , cuja variação representa a translação ao longo do eixo do helicoide.

7.1.2.1 REPRESENTAÇÃO DA ROTAÇÃO POR MEIO DE EIXO DE HELICOIDE

Conforme visto nas seções anteriores, um corpo submetido à rotação, em torno de um ponto fixo, possui um eixo de rotação conhecido como eixo do Helicoide. Foi visto também que um ponto sobre esse eixo não sofre mudanças de coordenadas (x, y, z) , e, dada a matriz de rotação, entre duas posições distintas, consegue-se definir o ângulo θ em torno do eixo do Helicoide, correspondente ao giro do corpo rígido.

A seguir, são determinadas as direções do eixo do Helicoide s , ou seja, (s_x, s_y, s_z) em função das componentes a_{ij} da matriz de rotação. Para isso, considere a FIG. 7.3, onde encontram-se duas posições sucessivas P_1 e P_2^r , correspondente à rotação em torno do eixo do Helicoide s .

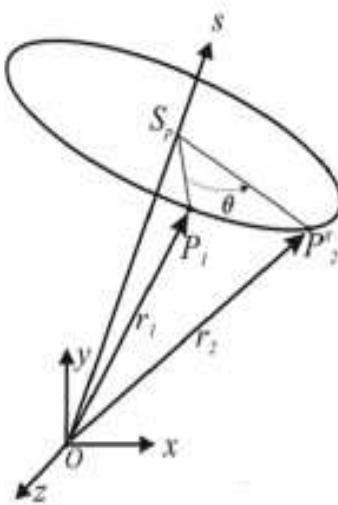


FIG. 7.3: Rotação em torno do eixo do helicoide (TSAI, 1999)

Seja P_1 a posição inicial de um ponto P pertencente ao corpo rígido e P_2^r corresponda à posição final do ponto P entre dois deslocamentos sucessivos. Note que ocorre uma rotação em torno do eixo do helicoide s da posição P_1 para P_2^r . Seja S_P um ponto sobre o eixo do helicoide pertencente ao plano perpendicular e que contenha os pontos P_1 e P_2^r . Analisando a geometria da FIG. 7.3, as seguintes expressões vetoriais podem ser escritas como:

$$OP_1 = OS_P + S_P P_1 \quad (7.34)$$

$$OP_1 = P_1 \quad (7.35)$$

$$OS_P = S_P = (P_1 \cdot s)s \quad (7.36)$$

Assim sendo, tem-se que:

$$S_P P_1 = P_1 - (P_1 \cdot s)s \quad (7.37)$$

Analogamente:

$$S_P P_2^r = P_2^r - (P_2^r \cdot s)s \quad (7.38)$$

Agora, considera-se a interseção do plano perpendicular à s , passando pelos pontos P_1 e P_2^r , sendo determinado o ponto S_p . N é a projeção de $S_P P_2^r$ sobre $S_P P_1$, de acordo com a FIG. 7.4. Logo, $NP_2^r \perp S_P P_1$ e $S_P N = S_P P_2^r \cos \theta$. No entanto, $S_P P_2^r = S_P P_1$, pois equivale a uma

rotação de θ em torno de S_P . Logo, $S_P N = S_P P_1 \cos \theta$.

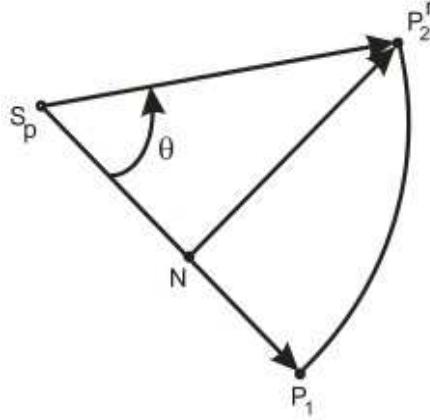


FIG. 7.4: Projeção de $S_P P_2^r$ sobre $S_P P_1$ (TSAI, 1999)

Sabendo ainda que:

$$\begin{aligned}
 s \times S_P P_2^r &= s \times [P_1 - (P_1 \cdot s) s] = \\
 &= s \times P_1 - s \times (P_1 \cdot s) s = \\
 &= s \times P_1 - (P_1 \cdot s) (s \times s) = \\
 &= s \times P_1
 \end{aligned} \tag{7.39}$$

pois $(s \times s) = 0$.

Deseja-se escrever a seguinte soma vetorial:

$$S_P P_2^r = S_P N + N P_2^r \tag{7.40}$$

Para isso, tem-se que:

$$N P_2^r = |N P_2^r| \left[s \times \frac{S_P P_1}{|S_P P_1|} \right] \tag{7.41}$$

Como:

$$|N P_2^r| = |S_P P_2^r| \sin \theta = |S_P P_1| \sin \theta \tag{7.42}$$

Assim:

$$N P_2^r = (s \times S_P P_1) \sin \theta \tag{7.43}$$

Os outros dois termos são:

$$S_P N = S_P P_1 \cos \theta \quad (7.44)$$

$$S_P P_2^r = P_2^r - (P_2^r \cdot s)s \quad (7.45)$$

Logo, a soma vetorial desejada fica:

$$S_P P_2^r = S_P N + N P_2^r \quad (7.46)$$

$$P_2^r - (P_2^r \cdot s)s = S_P P_1 \cos \theta + (s \times S_P P_1) \sin \theta \quad (7.47)$$

Como:

$$S_P P_1 = P_1 - (P_1 \cdot s)s \quad (7.48)$$

$$P_1 \cdot s = P_2^r \cdot s \quad (7.49)$$

Tem-se que:

$$P_2^r - (P_1 \cdot s)s = (P_1 - (P_1 \cdot s)s) \cos \theta + (s \times (P_1 - (P_1 \cdot s)s)) \sin \theta \quad (7.50)$$

Isolando P_2^r , tem-se:

$$P_2^r = (P_1 \cdot s)s - (P_1 \cdot s)s \cos \theta + P_1 \cos \theta + (s \times P_1 - (P_1 \cdot s)(s \times s)) \sin \theta \quad (7.51)$$

$$P_2^r = P_1 \cos \theta + (1 - \cos \theta)(P_1 \cdot s)s + (s \times P_1) \sin \theta \quad (7.52)$$

A EQ. 7.52 é conhecida como fórmula de Rodrigues para o deslocamento esférico.

É possível colocar na forma do Teorema de Euler ($p^A = R_B^A p^B$), de modo a determinar as componentes da matriz de rotação, fazendo $P_1 = p^B$ e $P_2^r = p^A$:

$$p^A = p^B \cos \theta + (1 - \cos \theta)(p^B \cdot s)s + (s \times p^B) \sin \theta \quad (7.53)$$

$$\begin{bmatrix} p_x^A \\ p_y^A \\ p_z^A \end{bmatrix} = \begin{bmatrix} p_x^B \\ p_y^B \\ p_z^B \end{bmatrix} \cos \theta + (1 - \cos \theta) \left(\begin{bmatrix} p_x^B \\ p_y^B \\ p_z^B \end{bmatrix} \cdot \begin{bmatrix} s_x \\ s_y \\ s_z \end{bmatrix} \right) \begin{bmatrix} s_x \\ s_y \\ s_z \end{bmatrix} + \left(\begin{bmatrix} s_x \\ s_y \\ s_z \end{bmatrix} \times \begin{bmatrix} p_x^B \\ p_y^B \\ p_z^B \end{bmatrix} \right) \sin \theta =$$

$$= \begin{bmatrix} p_x^B \\ p_y^B \\ p_z^B \end{bmatrix} \cos \theta + (1 - \cos \theta) (p_x^B s_x + p_y^B s_y + p_z^B s_z) \begin{bmatrix} s_x \\ s_y \\ s_z \end{bmatrix} + \begin{vmatrix} i & j & k \\ s_x & s_y & s_z \\ p_x^B & p_y^B & p_z^B \end{vmatrix} \sin \theta =$$

$$\begin{aligned} &= \begin{bmatrix} p_x^B \\ p_y^B \\ p_z^B \end{bmatrix} \cos \theta + (1 - \cos \theta) (p_x^B s_x s_x + p_y^B s_y s_x + p_z^B s_z s_x) i + \\ &\quad + (1 - \cos \theta) (p_x^B s_x s_y + p_y^B s_y s_y + p_z^B s_z s_y) j + \\ &\quad + (1 - \cos \theta) (p_x^B s_x s_z + p_y^B s_y s_z + p_z^B s_z s_z) k + \\ &\quad + ((s_y p_z^B - s_z p_y^B) i + (s_z p_x^B - s_x p_z^B) j + (s_x p_y^B - s_y p_x^B) k) \sin \theta = \end{aligned}$$

$$\begin{aligned} &= \begin{bmatrix} p_x^B \\ p_y^B \\ p_z^B \end{bmatrix} \cos \theta + (1 - \cos \theta) \begin{bmatrix} p_x^B s_x^2 + p_y^B s_y s_x + p_z^B s_z s_x \\ p_x^B s_x s_y + p_y^B s_y^2 + p_z^B s_z s_y \\ p_x^B s_x s_z + p_y^B s_y s_z + p_z^B s_z^2 \end{bmatrix} + \begin{bmatrix} s_y p_z^B - s_z p_y^B \\ s_z p_x^B - s_x p_z^B \\ s_x p_y^B - s_y p_x^B \end{bmatrix} \sin \theta = \\ &= \begin{bmatrix} p_x^B \cos \theta + p_x^B s_x^2 (1 - \cos \theta) + p_y^B s_y s_x (1 - \cos \theta) + \dots \\ p_y^B \cos \theta + p_x^B s_x s_y (1 - \cos \theta) + p_y^B s_y^2 (1 - \cos \theta) + \dots \\ p_z^B \cos \theta + p_x^B s_x s_z (1 - \cos \theta) + p_y^B s_y s_z (1 - \cos \theta) + \dots \\ \dots + p_z^B s_z s_x (1 - \cos \theta) + s_y p_z^B \sin \theta - s_z p_y^B \sin \theta \\ \dots + p_z^B s_z s_y (1 - \cos \theta) + s_z p_x^B \sin \theta - s_x p_z^B \sin \theta \\ \dots + p_z^B s_z^2 (1 - \cos \theta) + s_x p_y^B \sin \theta - s_y p_x^B \sin \theta \end{bmatrix} = \end{aligned}$$

$$\begin{aligned} &= \begin{bmatrix} p_x^B (\cos \theta + s_x^2 (1 - \cos \theta)) + p_y^B (s_y s_x (1 - \cos \theta) - s_z \sin \theta) + \dots \\ p_x^B (s_x s_y (1 - \cos \theta) + s_z \sin \theta) + p_y^B (\cos \theta + s_y^2 (1 - \cos \theta)) + \dots \\ p_x^B (s_x s_z (1 - \cos \theta) - s_y \sin \theta) + p_y^B (s_y s_z (1 - \cos \theta) + s_x \sin \theta) + \dots \\ \dots + p_z^B (s_z s_x (1 - \cos \theta) + s_y \sin \theta) \\ \dots + p_z^B (s_z s_y (1 - \cos \theta) - s_x \sin \theta) \\ \dots + p_z^B (\cos \theta + s_z^2 (1 - \cos \theta)) \end{bmatrix} = \end{aligned}$$

$$= \begin{bmatrix} \cos \theta + s_x^2 (1 - \cos \theta) & s_y s_x (1 - \cos \theta) - s_z \sin \theta & s_z s_x (1 - \cos \theta) + s_y \sin \theta \\ s_x s_y (1 - \cos \theta) + s_z \sin \theta & \cos \theta + s_y^2 (1 - \cos \theta) & s_z s_y (1 - \cos \theta) - s_x \sin \theta \\ s_x s_z (1 - \cos \theta) - s_y \sin \theta & s_y s_z (1 - \cos \theta) + s_x \sin \theta & \cos \theta + s_z^2 (1 - \cos \theta) \end{bmatrix} \begin{bmatrix} p_x^B \\ p_y^B \\ p_z^B \end{bmatrix}$$

Como:

$$\begin{bmatrix} p_x^A \\ p_y^A \\ p_z^A \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} p_x^B \\ p_y^B \\ p_z^B \end{bmatrix} = R_B^A \begin{bmatrix} p_x^B \\ p_y^B \\ p_z^B \end{bmatrix} \quad (7.54)$$

É possível determinar os termos da matriz de rotação em função da direção s do Helicoide e do ângulo θ , conforme a seguir:

$$a_{11} = \cos \theta + s_x^2 (1 - \cos \theta) \quad (7.55)$$

$$a_{12} = s_y s_x (1 - \cos \theta) - s_z \sin \theta \quad (7.56)$$

$$a_{13} = s_z s_x (1 - \cos \theta) + s_y \sin \theta \quad (7.57)$$

$$a_{21} = s_x s_y (1 - \cos \theta) + s_z \sin \theta \quad (7.58)$$

$$a_{22} = \cos \theta + s_y^2 (1 - \cos \theta) \quad (7.59)$$

$$a_{23} = s_z s_y (1 - \cos \theta) - s_x \sin \theta \quad (7.60)$$

$$a_{31} = s_x s_z (1 - \cos \theta) - s_y \sin \theta \quad (7.61)$$

$$a_{32} = s_y s_z (1 - \cos \theta) + s_x \sin \theta \quad (7.62)$$

$$a_{33} = \cos \theta + s_z^2 (1 - \cos \theta) \quad (7.63)$$

A partir disso, conhecendo-se o helicoide, é possível determinar a matriz de rotação e vice-versa, ou seja, combinando, corretamente, as componentes da matriz de rotação, pode-se isolar as componentes s_x , s_y e s_z , assim como o ângulo θ , como mostrado a seguir:

$$s_x = \frac{a_{32} - a_{23}}{2 \sin \theta}; \quad (7.64)$$

$$s_y = \frac{a_{13} - a_{31}}{2 \sin \theta}; \quad (7.65)$$

$$s_z = \frac{a_{21} - a_{12}}{2 \sin \theta} \quad (7.66)$$

$$2 \sin \theta = \sqrt{4 - (a_{11} + a_{22} + a_{33} - 1)^2} \quad (7.67)$$

$$\theta = \arccos \left(\frac{a_{11} + a_{22} + a_{33} - 1}{2} \right) \quad (7.68)$$

$$s_x = \frac{a_{32} - a_{23}}{\sqrt{4 - (a_{11} + a_{22} + a_{33} - 1)^2}} \quad (7.69)$$

$$s_y = \frac{a_{13} - a_{31}}{\sqrt{4 - (a_{11} + a_{22} + a_{33} - 1)^2}} \quad (7.70)$$

$$s_z = \frac{a_{21} - a_{12}}{\sqrt{4 - (a_{11} + a_{22} + a_{33} - 1)^2}} \quad (7.71)$$

7.1.2.2 DESLOCAMENTO HELICOIDAL

Uma forma mais completa do Teorema de Chasles enuncia como o corpo rígido desloca-se da posição inicial para a posição final, ou seja, afirma que o deslocamento pode ser feito por meio de uma combinação de uma rotação e uma translação em torno de um único eixo, a qual é conhecida como deslocamento helicoidal.

Tem-se o ponto P em três posições distintas: P_1 , P_2^r e P_2 , conforme representado na FIG. 7.5. Existe uma rotação θ entre P_1 e P_2^r , e uma translação t entre P_2^r e P_2 . A direção do eixo do helicoide é dada pelo vetor s , onde ocorrerá a translação e a rotação. É necessária ainda a determinação de um outro vetor, denominado s_0 que aponte da origem para qualquer ponto do eixo do helicoide, de modo que esse eixo fique determinado no espaço tridimensional. Dessa forma, o deslocamento helicoidal fica perfeitamente determinado por s , s_0 , θ e t .

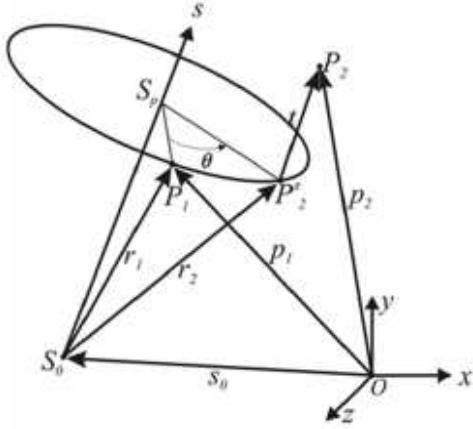


FIG. 7.5: Rotação e translação em torno de um único eixo (TSAI, 1999)

As seguintes somas vetoriais podem ser escritas:

$$P_1 = s_0 + s_0 P_1 \quad (7.72)$$

$$P_2 = s_0 P_2^r + s_0 + ts \quad (7.73)$$

É possível substituir na equação de Rodrigues, na forma do teorema de *Chasles*, de modo a determinar as componentes da matriz de rotação, fazendo $s_0 P_1 = r_1$ e $s_0 P_2^r = r_2$.

$$\begin{aligned} s_0 P_2^r &= s_0 P_1 \cos \theta + (1 - \cos \theta) (s_0 P_1 \cdot s) s + (s \times s_0 P_1) \sin \theta \\ P_2 - s_0 - ts &= (P_1 - s_0) \cos \theta + (1 - \cos \theta) ((P_1 - s_0) \cdot s) s + (s \times (P_1 - s_0)) \sin \theta \\ P_2 &= s_0 + ts + (P_1 - s_0) \cos \theta + (1 - \cos \theta) ((P_1 - s_0) \cdot s) s + (s \times (P_1 - s_0)) \sin \theta \\ P_2 &= P_1 \cos \theta + (1 - \cos \theta) (P_1 \cdot s) s + (s \times P_1) \sin \theta + \\ &\quad + s_0 + ts - s_0 \cos \theta - (1 - \cos \theta) (s_0 \cdot s) s - (s \times s_0) \sin \theta \end{aligned} \quad (7.74)$$

Colocando na forma do teorema de *Chasles*, fazendo $P_1 = p^B$ e $P_2 = p^A$.

$$\begin{aligned} p^A &= R_B^A p^B + q^A \\ p^A &= p^B \cos \theta + (1 - \cos \theta) (p^B \cdot s) s + (s \times p^B) \sin \theta + \\ &\quad + s_0 + ts - s_0 \cos \theta - (1 - \cos \theta) (s_0 \cdot s) s - (s \times s_0) \sin \theta \end{aligned} \quad (7.75)$$

O termo da rotação é idêntico ao analisado na rotação em torno de um ponto, o que implica

que as componentes da matriz de rotação são as mesmas, o que não poderia deixar de ser, visto que é uma soma vetorial:

$$\begin{aligned}
 & \begin{bmatrix} {}^A p_x \\ {}^A p_y \\ {}^A p_z \end{bmatrix} = \\
 & = \begin{bmatrix} \cos \theta + s_x^2 (1 - \cos \theta) & s_y s_x (1 - \cos \theta) - s_z \sin \theta & s_z s_x (1 - \cos \theta) + s_y \sin \theta \\ s_x s_y (1 - \cos \theta) + s_z \sin \theta & \cos \theta + s_y^2 (1 - \cos \theta) & s_z s_y (1 - \cos \theta) - s_x \sin \theta \\ s_x s_z (1 - \cos \theta) - s_y \sin \theta & s_y s_z (1 - \cos \theta) + s_x \sin \theta & \cos \theta + s_z^2 (1 - \cos \theta) \end{bmatrix} \cdot \\
 & \quad \cdot \begin{bmatrix} p_x^B \\ p_y^B \\ p_z^B \end{bmatrix} + \begin{bmatrix} {}^A q_x \\ {}^A q_y \\ {}^A q_z \end{bmatrix} \tag{7.76}
 \end{aligned}$$

Associando ainda os termos da translação, é possível encontrar:

$$\begin{aligned}
 & \begin{bmatrix} {}^A q_x \\ {}^A q_y \\ {}^A q_z \end{bmatrix} = s_0 + t s - s_0 \cos \theta - (1 - \cos \theta) (s_0 \cdot s) s - (s \times s_0) \sin \theta = \\
 & = \begin{bmatrix} s_{0x} \\ s_{0y} \\ s_{0z} \end{bmatrix} + t \begin{bmatrix} s_x \\ s_y \\ s_z \end{bmatrix} - \begin{bmatrix} s_{0x} \\ s_{0y} \\ s_{0z} \end{bmatrix} \cos \theta + \\
 & \quad - (1 - \cos \theta) (s_x s_{0x} + s_y s_{0y} + s_z s_{0z}) \begin{bmatrix} s_x \\ s_y \\ s_z \end{bmatrix} - \begin{vmatrix} i & j & k \\ s_x & s_y & s_z \\ s_{0x} & s_{0y} & s_{0z} \end{vmatrix} \sin \theta =
 \end{aligned}$$

$$\begin{aligned}
 & = \begin{bmatrix} s_{0x}(1 + \cos \theta) + s_x (t - (1 - \cos \theta) (s_x s_{0x} + s_y s_{0y} + s_z s_{0z})) - (s_y s_{0z} - s_z s_{0y}) \sin \theta \\ s_{0y}(1 + \cos \theta) + s_y (t - (1 - \cos \theta) (s_x s_{0x} + s_y s_{0y} + s_z s_{0z})) - (s_z s_{0x} - s_x s_{0z}) \sin \theta \\ s_{0z}(1 + \cos \theta) + s_z (t - (1 - \cos \theta) (s_x s_{0x} + s_y s_{0y} + s_z s_{0z})) - (s_x s_{0y} - s_y s_{0x}) \sin \theta \end{bmatrix} = \\
 & = \begin{bmatrix} t s_x + s_{0x}(1 + \cos \theta) - (1 - \cos \theta) (s_x^2 s_{0x} + s_x s_y s_{0y} + s_x s_z s_{0z}) - (s_y s_{0z} - s_z s_{0y}) \sin \theta \\ t s_y + s_{0y}(1 + \cos \theta) - (1 - \cos \theta) (s_y s_x s_{0x} + s_y^2 s_{0y} + s_y s_z s_{0z}) - (s_z s_{0x} - s_x s_{0z}) \sin \theta \\ t s_z + s_{0z}(1 + \cos \theta) - (1 - \cos \theta) (s_z s_x s_{0x} + s_z s_y s_{0y} + s_z^2 s_{0z}) - (s_x s_{0y} - s_y s_{0x}) \sin \theta \end{bmatrix} =
 \end{aligned}$$

$$\begin{aligned}
&= \left[\begin{array}{l} t s_x + s_{0x} (1 - (\cos \theta + (1 - \cos \theta) s_x^2)) \\ t s_y - s_{0x} (s_z \sin \theta + (1 - \cos \theta) s_y s_x) \\ t s_z - s_{0x} (-s_y \sin \theta + (1 - \cos \theta) s_z s_x) \end{array} \right] + \\
&\quad + \left[\begin{array}{l} -s_{0y} ((1 - \cos \theta) s_x s_y - s_z \sin \theta) \\ s_{0y} (1 - (\cos \theta + (1 - \cos \theta) s_y^2)) \\ -s_{0y} ((1 - \cos \theta) s_z s_y + s_x \sin \theta) \end{array} \right] + \\
&\quad + \left[\begin{array}{l} -s_{0z} ((1 - \cos \theta) s_x s_z + s_y \sin \theta) \\ -s_{0z} ((1 - \cos \theta) s_y s_z - s_x \sin \theta) \\ s_{0z} (1 - (\cos \theta + (1 - \cos \theta) s_z^2)) \end{array} \right] = \\
&= \left[\begin{array}{l} t s_x + s_{0x} (1 - a_{11}) - s_{0y} a_{12} - s_{0z} a_{13} \\ t s_y - s_{0x} a_{21} + s_{0y} (1 - a_{22}) - s_{0z} a_{23} \\ t s_z - s_{0x} a_{31} - s_{0y} a_{32} + s_{0z} (1 - a_{33}) \end{array} \right] = \\
&= \left[\begin{array}{l} t s_x + s_{0x} - s_{0x} a_{11} - s_{0y} a_{12} - s_{0z} a_{13} \\ t s_y + s_{0y} - s_{0x} a_{21} - s_{0y} a_{22} - s_{0z} a_{23} \\ t s_z + s_{0z} - s_{0x} a_{31} - s_{0y} a_{32} - s_{0z} a_{33} \end{array} \right] = \\
&= \left[\begin{array}{l} t s_x \\ t s_y \\ t s_z \end{array} \right] + \left[\begin{array}{l} s_{0x} \\ s_{0y} \\ s_{0z} \end{array} \right] - \left[\begin{array}{ccc} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{array} \right] \left[\begin{array}{l} s_{0x} \\ s_{0y} \\ s_{0z} \end{array} \right] = \\
&= t s + s_0 - R_B^A s_0
\end{aligned}$$

Assim:

$$q^A = t s + [I - R_B^A] s_0 \quad (7.77)$$

Para calcular s_0 , tem-se:

$$q^A - t s = [I - R_B^A] s_0 \quad (7.78)$$

$$s_0 = [I - R_B^A]^{-1} [q^A - t s] \quad (7.79)$$

Assim, para o deslocamento helicoidal, tem-se que a equação do Teorema de *Chasles* pode

ser reescrita como:

$$p^A = R_B^A p^B + q^A \quad (7.80)$$

$$q^A = t s + [I - R_B^A]s_0 \quad (7.81)$$

$$p^A = R_B^A p^B + t s + [I - R_B^A]s_0 \quad (7.82)$$

A matriz de transformação homogênea (4x4) passa a ser reescrita como:

$$\begin{aligned} T_B^A &= \begin{bmatrix} a_{11} & a_{12} & a_{13} & q_x^A \\ a_{21} & a_{22} & a_{23} & q_y^A \\ a_{31} & a_{32} & a_{33} & q_z^A \\ 0 & 0 & 0 & 1 \end{bmatrix} = \\ &= \begin{bmatrix} R_B^A & t s + [I - R_B^A]s_0 \\ 0 & 1 \end{bmatrix} \end{aligned} \quad (7.83)$$

Onde:

$$q_x^A = t s_x + s_{0x} - s_{0x} a_{11} - s_{0y} a_{12} - s_{0z} a_{13} \quad (7.84)$$

$$q_y^A = t s_y + s_{0y} - s_{0x} a_{21} - s_{0y} a_{22} - s_{0z} a_{23} \quad (7.85)$$

$$q_z^A = t s_z + s_{0z} - s_{0x} a_{31} - s_{0y} a_{32} - s_{0z} a_{33} \quad (7.86)$$

A forma homogênea do vetor $p^A = [p_x^A, p_y^A, p_z^A]^T$ é representada por $\tilde{p}^A = [p_x^A, p_y^A, p_z^A, 1]^T$.

Assim, tem-se que:

$$\tilde{p}^A = T_B^A \tilde{p}^B \quad (7.87)$$

A Matriz de Transformação Homogênea possui inversa, embora não seja ortogonal, ou seja, $[T_B^A]^{-1} \neq [T_B^A]^T$, e pode ser calculada pré-multiplicando a fórmula do teorema de Chasles pelo inverso da matriz de rotação, como segue:

$$[R_B^A]^{-1} p^A = [R_B^A]^{-1} R_B^A p^B + [R_B^A]^{-1} q^A \quad (7.88)$$

Como a matriz de rotação é ortogonal, sua inversa é igual a sua transposta:

$$[R_B^A]^{-1} = [R_B^A]^T \quad (7.89)$$

$$p^B = [R_B^A]^{-1} p^A - [R_B^A]^{-1} q^A \quad (7.90)$$

$$\tilde{p}^B = [T_B^A]^{-1} \tilde{p}^A \quad (7.91)$$

Logo:

$$[T_B^A]^{-1} = \begin{bmatrix} [R_B^A]^T & -t[R_B^A]^T s - [[R_B^A]^T - I] s_0 \\ 0 & 1 \end{bmatrix} \quad (7.92)$$

7.1.3 MÉTODO DOS DESLOCAMENTOS DOS HELICOIDES SUCESSIVOS

Seja um corpo rígido σ composto de dois pares cinemáticos representados por $\$_1$ e $\$_2$. O primeiro par cinemático conecta o “elo móvel 1” ao eixo fixo, e o segundo par cinemático conecta o “elo móvel 2” ao eixo móvel, que é solidário ao elo móvel 1, como mostra a FIG. 7.6.

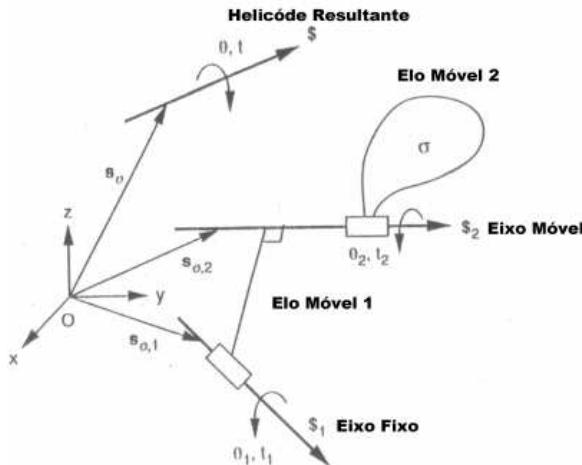


FIG. 7.6: Deslocamento de um corpo por dois helicoides sucessivos (TSAI, 1999)

Portanto, o corpo rígido pode se deslocar sobre dois eixos, que são completamente definidos pelos vetores s e s_0 de cada um dos helicoides $\$_1$ e $\$_2$. O deslocamento pode ser simultâneo, ou seja, o corpo pode girar de um ângulo θ_1 e θ_2 em torno dos helicoides $\$_1$ e $\$_2$, bem como, pode ocorrer a translação de d_1 e d_2 sobre cada um dos eixos de juntas, ou seja, sobre cada um dos helicoides $\$_1$ e $\$_2$.

O teorema de Chasles para deslocamento helicoidal (EQ. 7.82) pode ser reescrito, baseado na matriz de transformação homogênea verificada na EQ. 7.83. Logo, o deslocamento (θ_1, d_1) sobre o eixo $(s \text{ e } s_0)$ do Helicoide $\$1$, fica representado devidamente pelo próprio Helicoide $\$1$, ou seja, o Helicoide $\$1 (s, s_0, \theta_1, d_1)$ terá a sua matriz de transformação homogênea dada por A_1 .

$$A_1 = \begin{bmatrix} R_{1(3x3)} & d_1 s_{\$1(1x3)} + (I_{(3x3)} - R_{1(3x3)})_{(3x3)} s_{0\$1(1x3)} \\ 0 \ 0 \ 0 & 1 \end{bmatrix} \quad (7.93)$$

onde:

$$R_1 = \begin{bmatrix} c\theta_1 + s_x^2(1 - c\theta_1) & s_y s_x (1 - c\theta_1) - s_z s\theta_1 & s_z s_x (1 - c\theta_1) + s_y s\theta_1 \\ s_x s_y (1 - c\theta_1) + s_z s\theta_1 & c\theta_1 + s_y^2 (1 - c\theta_1) & s_z s_y (1 - c\theta_1) - s_x s\theta_1 \\ s_x s_z (1 - c\theta_1) - s_y s\theta_1 & s_y s_z (1 - c\theta_1) + s_x s\theta_1 & c\theta_1 + s_z^2 (1 - c\theta_1) \end{bmatrix} \quad (7.94)$$

$$d_1 s_{\$1(1x3)} + (I_{(3x3)} - R_{1(3x3)}) s_{0\$1(1x3)} = \begin{bmatrix} d_1 s_x + s_{0x} (1 - a_{11}) - s_{0y} a_{12} - s_{0z} a_{13} \\ d_1 s_y - s_{0x} a_{21} + s_{0y} (1 - a_{22}) - s_{0z} a_{23} \\ d_1 s_z - s_{0x} a_{31} - s_{0y} a_{32} + s_{0z} (1 - a_{33}) \end{bmatrix} \quad (7.95)$$

Partindo de uma posição inicial, ocorrem deslocamentos sucessivos, ou seja, primeiro há um giro de θ_2 em torno do helicoide $\$2$ e uma translação sobre o mesmo, que tem associada a Matriz de Transformação Homogênea A_2 . Em seguida, ocorre outro giro de θ_1 e uma translação de t_1 sobre o helicoide $\$1$, que, por sua vez, tem associada a Matriz de Transformação Homogênea A_1 , que é capaz de influenciar os corpos que o sucedem.

Assim, existe um helicoide $\$$ resultante que tem associada a matriz de transformação homogênea, dada por A_r , resultado do produtório das respectivas matrizes relacionadas aos deslocamentos de helicoides sucessivos, respeitando a ordem em que ocorrem, conforme a sequência da cadeia cinemática. Nesse caso, tem-se:

$$A_r = A_1 A_2 \quad (7.96)$$

Esta metodologia é adequada para determinação da cinemática direta de manipuladores seriais de mais de um elo, como mostra a FIG. 7.7. Conforme TSAI (1999), é usual numerar os

helicoides em ordem crescente da base em direção à ferramenta.

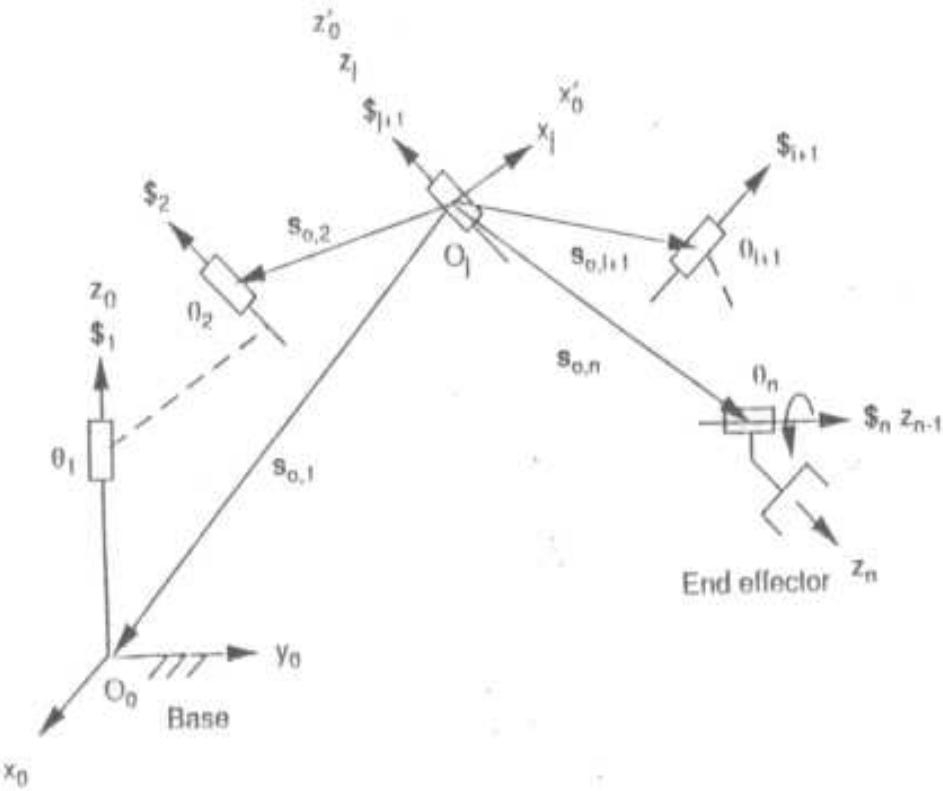


FIG. 7.7: Deslocamento de um corpo por n helicoides sucessivos (TSAI, 1999)

Em um manipulador serial de n -elos, deseja-se conhecer o efeito que a variação das variáveis das juntas causam na ferramenta, localizada no último elo, a qual sofre a influência dos elos anteriores. Iniciando uma análise do último Helicoide em direção ao primeiro, ou seja, da ferramenta em direção à base, tem-se que o Helicoide $\$_n$ é passível de sofrer influência do Helicoide anterior e, assim por diante, o que implica que sua Matriz de Transformação Homogênea A_n deverá ser pré-multiplicada pela anterior (A_{n-1}) e esse resultado ($A_{n-1} A_n$), pela anterior (A_{n-2}) e assim, sucessivamente, até a multiplicação por A_1 que corresponde ao Helicoide que tem seu eixo ligado à base, ou seja:

$$A_r = A_1 A_2 \cdots A_{n-2} A_{n-1} A_n \quad (7.97)$$

Primeiramente, deve ser definida a posição de referência para o manipulador ou posição zero, que geralmente é escolhida em uma localização de fácil definição dos vetores determinantes do

eixo das juntas, ou seja, os vetores s e s_0 do helicoide $\$_i$, com i variando de 1 até n .

Geralmente, a posição desejada é chamada de posição alvo, segundo TSAI (1999). O manipulador irá se deslocar da posição de referência para a posição alvo, por meio de uma série de deslocamentos de helicoides finitos sucessivos sobre os eixos das juntas, ou seja, a rotação do n -ésimo eixo da junta, seguido por outro sobre o $(n - 1)$ -ésimo eixo da junta, e assim sucessivamente. O deslocamento helicoidal resultante é obtido pela pré-multiplicação das matrizes de transformação homogênea dos helicoides sucessivos presentes, conforme a EQ. 7.97.

De modo a facilitar a implementação computacional da cinemática direta, a FIG. 7.8 apresenta o fluxograma relativo ao Métodos dos Deslocamentos dos Helicoides Sucessivos.

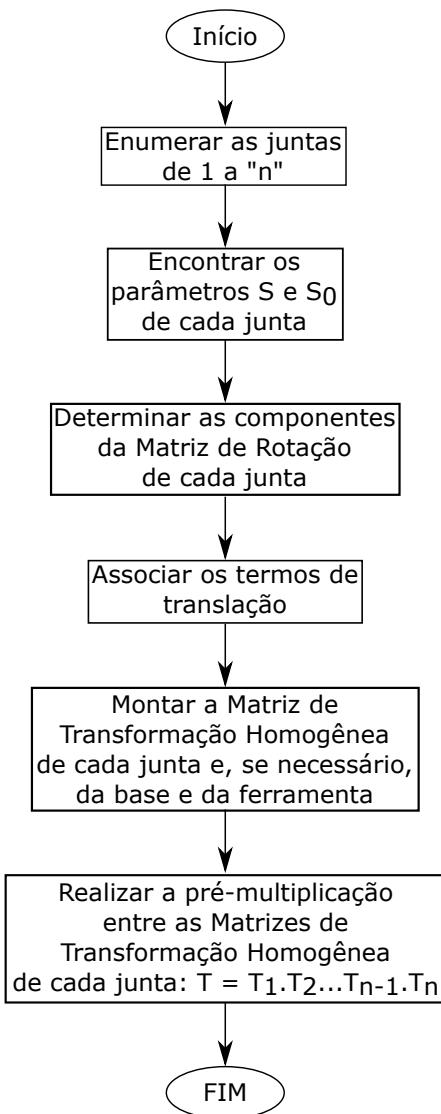


FIG. 7.8: Fluxograma da Cinemática Direta por Helicoides (Autoria Própria)

7.1.4 JACOBIANO DE UM MANIPULADOR SERIAL

O Jacobiano de um manipulador serial é a transformação linear entre o vetor das velocidades das juntas \dot{q} com dimensão n , que corresponde ao número de juntas do manipulador, e o vetor velocidade generalizada \dot{x} de um ponto pertencente ao corpo, composto pelas velocidades angular e linear, em relação a um referencial absoluto. Assim, o vetor velocidade generalizada desse ponto pode ser representado por um par de vetores composto por:

$$\dot{x} \equiv \begin{bmatrix} w_x \\ w_y \\ w_z \\ v_x \\ v_y \\ v_z \end{bmatrix} \quad (7.98)$$

Com seis coordenadas, a velocidade instantânea do corpo rígido fica determinada, em que cada uma dessas componentes é função de todas as variáveis de junta que influenciam o ponto pertencente ao sistema mecânico, que, no caso de um manipulador serial, corresponde à ferramenta ou efetuador-final.

Cada junta do manipulador pode ser representada por um helicoide $\$$, sendo $\hat{\$}$ o seu respectivo helicoide unitário. É possível relacioná-los pela multiplicação de uma amplitude ou intensidade \dot{q} . Dessa forma:

$$\$ = \dot{q} \hat{\$} \quad (7.99)$$

Assume-se que, para a junta de revolução: $\dot{q} = \dot{\theta}$. Para a junta de translação: $\dot{q} = \dot{d}$.

O helicoide unitário é, apropriadamente, representado por um par de vetores, da seguinte forma:

$$\hat{\$} = \begin{bmatrix} s \\ s_0 \times s + h s \end{bmatrix} = \begin{bmatrix} s_x \\ s_y \\ s_z \\ (s_{0y}s_z - s_{0z}s_y) + hs_x \\ (s_{0z}s_x - s_{0x}s_z) + hs_y \\ (s_{0x}s_y - s_{0y}s_x) + hs_z \end{bmatrix} \quad (7.100)$$

Para a junta de rotação, a translação é nula, ou seja, $d = 0$ o que implica que o passo é nulo,

$h = 0$. Então, o helicoide unitário para uma junta de rotação é dado por:

$$\hat{\$} = \begin{bmatrix} s \\ s_0 \times s \end{bmatrix} = \begin{bmatrix} s_x \\ s_y \\ s_z \\ s_z s_{0y} - s_y s_{0z} \\ s_x s_{0z} - s_z s_{0x} \\ s_y s_{0x} - s_x s_{0y} \end{bmatrix} \quad (7.101)$$

O vetor $(s_0 \times s)$ define o momento do eixo do Helicoide em torno da origem, e como o vetor s é ortogonal a $(s_0 \times s)$, o produto escalar entre eles é nulo.

Para a junta prismática, tem-se somente translação, ou seja, a rotação é nula, $\theta = 0$, resultando em um passo infinito. Assim, o helicoide unitário é dado por:

$$\hat{\$} = \begin{bmatrix} 0 \\ s \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ s_x \\ s_y \\ s_z \end{bmatrix} \quad (7.102)$$

Em cada helicoide, as três primeiras coordenadas representam a velocidade angular, e as três últimas, a velocidade linear de um ponto que se move solidário ao corpo rígido e, instantaneamente, encontra-se coincidente com a origem do referencial fixo. Trata-se de uma abstração, imaginando que uma extensão desse corpo passasse instantaneamente pela origem do referencial absoluto, no qual é medida sua velocidade linear.

Dessa maneira, o vetor de estado de velocidades instantâneas do efetuador final de um manipulador composto por n juntas pode ser enunciado da seguinte forma:

$$\dot{x} = \$_1 + \$_2 + \dots + \$_{n-1} + \$_n \quad (7.103)$$

$$\dot{x} = \hat{\$}_1 \dot{q}_1 + \hat{\$}_2 \dot{q}_2 + \dots + \hat{\$}_{n-1} \dot{q}_{n-1} + \hat{\$}_n \dot{q}_n \quad (7.104)$$

$$\dot{x} \equiv \begin{bmatrix} w_x \\ w_y \\ w_z \\ v_x \\ v_y \\ v_z \end{bmatrix} = \begin{bmatrix} \hat{\$}_1 & \hat{\$}_2 & \dots & \hat{\$}_{n-1} & \hat{\$}_n \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \vdots \\ \dot{q}_{n-1} \\ \dot{q}_n \end{bmatrix} \quad (7.105)$$

TSAI (1999) denomina o Jacobiano baseado em helicoides, J_H , por meio da matriz:

$$J_H = \begin{bmatrix} \hat{\$}_1 & \hat{\$}_2 & \dots & \hat{\$}_{n-1} & \hat{\$}_n \end{bmatrix} \quad (7.106)$$

onde cada coluna da matriz é composta por um helicoide unitário representativo de sua respectiva junta.

Vale destacar que os helicoides unitários para as juntas de revolução e de translação encontram-se, respectivamente, representados nas EQ. 7.101 e 7.102. Além disso, para a junta rotacional $\dot{q} = \dot{\theta}$ e para a prismática $\dot{q} = \dot{d}$.

A EQ. 7.105 pode ser reescrita da seguinte forma:

$$v = J_H(q)\dot{q} \quad (7.107)$$

onde:

- v é o vetor das velocidades generalizadas do efetuador final com dimensão ($r \times 1$);
- r é o número do espaço operacional necessário à realização de uma dada tarefa; e,
- \dot{q} é o vetor das velocidades das juntas, com dimensão ($n \times 1$), onde n é o número do grau de liberdade do manipulador, proporcionado pelas n juntas.