UNIVERSIDADE FEDERAL DE MINAS GERAIS
ESCOLA DE ENGENHARIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA MECÂNICA

KÁSSIO MACIEL KIENITZ

TRADUÇÃO EM TEMPO REAL DE CINEMÁTICA NO COMANDO DE MOVIMENTO DE ROBÔS

Belo Horizonte

Fevereiro de 2023

KÁSSIO MACIEL KIENITZ

TRADUÇÃO EM TEMPO REAL DE CINEMÁTICA NO COMANDO DE MOVIMENTO DE ROBÔS

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Mecânica do Escola de Engenharia da Universidade Federal de Minas Gerais como requisito parcial para a obtenção do grau de Mestre em Engenharia Mecânica.

Orientador: Eduardo José Lima II

Belo Horizonte

Fevereiro de 2023

Kienitz, Kássio Maciel.
K47t Tradução em tempo real de cinemática no comando de movimento de robôs [recurso eletrônico] / Kássio Maciel Kienitz. - 2023.
1 recurso online (126 f. : il., color.) : pdf.

Orientador: Eduardo José Lima II.

Dissertação (mestrado) - Universidade Federal de Minas Gerais, Escola de Engenharia.

Apêndices: f. 121-126.

Bibliografia: f. 115-118.
Exigências do sistema: Adobe Acrobat Reader.

1. Engenharia mecânica - Teses. 2. Robótica - Teses. 3. Cinemática - Teses. 4. Robôs - Teses. 5. Robôs - Movimento - Teses. 6. Robôs - Sistemas de controle - Teses. I. Lima II, Eduardo José. II. Universidade Federal de Minas Gerais. Escola de Engenharia. III. Título.

CDU: 621 (043)

Ficha catalográfica elaborada pela bibliotecária Ângela Cristina Silva CRB/6 2361
Biblioteca Prof. Mário Werneck, Escola de Engenharia da UFMG

UNIVERSIDADE FEDERAL DE MINAS GERAIS
ESCOLA DE ENGENHARIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA MECÂNICA

FOLHA DE APROVAÇÃO

"TRADUÇÃO EM TEMPO REAL DE CINEMÁTICA NO COMANDO DE MOVIMENTO DE ROBÔS"

KÁSSIO MACIEL KIENITZ

Dissertação submetida à Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em Engenharia Mecânica da Universidade Federal de Minas Gerais, constituída pelos Professores: Dr. Eduardo José Lima II (Orientador – Departamento de Engenharia Mecânica/UFMG), Dr. Antônio Augusto Torres Maia (Departamento de Engenharia Mecânica/UFMG) e Dr. Alessandro Corrêa Victorino (Sorbonne Universités - Université de Technologie de Compiègne), como parte dos requisitos necessários à obtenção do título de **"Mestre em Engenharia Mecânica"**, na área de concentração de **"Projeto e Sistemas".**

Dissertação aprovada no dia 16 de fevereiro de 2023.

Documento assinado eletronicamente por **Eduardo Jose Lima Ii**, **Professor do Magistério Superior**, em 24/02/2023, às 19:34, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Alessandro Correa Victorino**, **Usuário Externo**, em 02/03/2023, às 05:50, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Antonio Augusto Torres Maia**, **Professor do Magistério Superior**, em 02/03/2023, às 09:31, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

A autenticidade deste documento pode ser conferida no site https://sei.ufmg.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **2068421** e o código CRC **59B01529**.

Dedico esse trabalho a todos os trabalhadores que perdem sua saúde executando tarefas perigosas que podem ser executadas por robôs e máquinas.

Agradecimentos

Agradeço a:

Minha esposa Luiza, que me apoiou nas horas mais difíceis de forma incondicional.

Aos meus pais por muita paciência e compreensão.

Aos meus amigos que foram perfeitos nas horas de desabafo.

A FAR-X Tecnologia por prover o laboratório para o desenvolvimento dos trabalhos.

A meu orientador por instigar sempre o meu desenvolvimento.

"Eles são mecanicamente mais perfeitos do que nós, têm uma capacidade intelectual espantosa, mas não têm alma."

(Karel Čapek, R.U.R, 1920)

Resumo

O presente trabalho tem como propósito o desenvolvimento de um sistema de tradução em tempo real de soluções cinemáticas. Realizando um estudo de caso de aplicação na tradução de uma cinemática cartesiana para a cinemática de um robô híbrido paralelo com uma extensão serial. A relevância do trabalho está no fato de sistemas de comando numérico se basearem no controle de máquinas de arquitetura cartesiana. Para o comando de robôs não cartesianos é necessária a tradução dos comandos de movimento do espaço cartesiano para o espaço de juntas, levando à necessidade de alterações na programação do software de comando. O sistema de tradução em tempo real atua no set-point de movimento possibilitando que CNCs possam ser utilizados como controladores de movimento de robôs com geometrias mais complexas sem perda de recursos. A geometria híbrida do robô estudado consiste em um robô planar paralelo de quatro barras na configuração pantográfica. Então a esta cadeia fechada de juntas rotativas é acoplado um terceiro link serial formando uma cadeia aberta em sua extremidade. A atuação do robô é totalmente concêntrica em sua base, e sua ferramenta se movimenta no plano XY e tem sua orientação do giro no eixo Z controlada. As soluções das cinemáticas são encontradas utilizando uma metodologia híbrida unindo o método do desacoplamento cinemático com as soluções paramétricas K-K e D-H, e o método geométrico. Ao final obtém-se uma solução totalmente analítica para as configurações do robô. O sistema de tradução foi implementado em um em um modelo simulado desenvolvido no GNU Octave, de forma que os parâmetros fossem estudados em um ambiente controlado. Foi utilizada a metodologia DoE, design de experimento, para desenvolver um experimento multifatorial de estudo das interações entre os parâmetros de interesse do modelo matemático. Após este estudo seus resultados foram utilizados para o desenvolvimento de uma aplicação real embarcada. Isto para possibilitar o teste de eficácia do sistema de tradução utilizando um CNC real pré-existente no mercado. Este provém o comando de movimento em forma de frequência para cada eixo cartesiano. O sinal de comando é introduzido no tradutor de cinemática, que utilizando a solução das cinemáticas desenvolvidas para gerar um set-point de movimento de acordo com o espaço de juntas do robô. Para se avaliar a eficácia do sistema o movimento comandado no espaço cartesiano e o movimento traduzido e aplicado ao modelo do robô são comparados. Ao final do desenvolvimento conclui-se que é possível realizar a tradução embarcada em tempo real das cinemáticas do robô em questão com uma resposta que atende as expectativas de um sistema robótico. Sendo possível a utilização deste sistema como ferramenta no auxílio do desenvolvimento e estudo de novas soluções e geometrias robóticas.

Palavras-Chaves: Robótica,Cinemática,Sistemas Embarcados,Comando Numérico

Abstract

The purpose of this work is to develop a real-time translation system for kinematic solutions. A case study is conducted on the translation of a Cartesian kinematic to the kinematic of a hybrid parallel robot with a serial extension. The relevance of this work lies in the fact that numerical control systems are based on Cartesian architecture machines. For the command of non-Cartesian robots, it is necessary to translate the motion commands from Cartesian space to joint space, leading to the need for changes in the control software programming. The real-time translation system acts on the motion set-point, enabling CNCs to be used as motion controllers for robots with more complex geometries without loss of features. The hybrid geometry of the studied robot consists of a four-bar planar parallel robot in a pantographic configuration. Then, a third serial link is coupled to this closed chain of rotary joints, forming an open chain at its end. The robot's actuation is entirely concentric at its base, and its tool moves in the XY plane, with its tool orientation over the Z-axis rotation. The kinematic solutions are found using a hybrid methodology combining the kinematic decoupling method with the K-K and D-H parametric solutions and the geometric method. Finally, a fully analytical solution is obtained for the robot's configurations. The translation system was implemented in a simulated model developed in GNU Octave so that the parameters could be studied in a controlled environment. The DoE methodology, design of experiment, was used to develop a multifactorial experiment to study the interactions between the parameters of interest of the mathematical model. After this study, its results were used to develop a real embedded application. This was done to enable testing of the translation system's effectiveness using an existing CNC on the market, which provides motion command in the form of frequency for each Cartesian axis. The command signal is introduced into the kinematic translator, which uses the developed kinematic solutions to generate a motion set-point according to the robot's joint space. To evaluate the effectiveness of the system, the commanded motion in Cartesian space and the translated motion applied to the robot model are compared. At the end of the development, it is concluded that it is possible to perform real-time embedded translation of the kinematics of the robot with a response that meets the expectations of a robotic system. Thus this system can be used as a tool to aid in the development and study of new robotic solutions and geometries.

Keywords: Robotics, Kinematics, Embeded Systems, Numeric Comand.

Lista de Figuras

Lista de Tabelas

Sumário

1 Introdução

O crescente desenvolvimento dos sistemas abertos de comando de máquinas de manufatura automática, como impressoras 3D, routers, fresas CNC, aumentando o acesso destas tecnologias à pesquisadores, estudantes e entusiastas. Assim, há uma crescente interação entre diferentes áreas de desenvolvimento e as máquinas automáticas, o qual possibilita novas áreas de pesquisa. Este trabalho propõem o desenvolvimento de uma ferramenta que permitirá a utilização de sistemas de comando de movimento de arquitetura aberta (desenvolvidos para uma cinemática cartesiana) para comandar máquinas que tenham geometrias diversas.

Foi demonstrado por [Alvares et al., 2018] a possibilidade de se utilizar um controlador CNC baseado em Linux com a implementação das soluções cinemáticas diretamente no gerador de movimento em um PC, porém, isso limita a solução a um software específico. Com a presente proposta parte de um controlador CNC genérico, desenvolvido para controlar uma máquina de três eixos prismáticos, e o faz capaz de para controlar um robô com eixos não cartesianos. Isso mantendo a programação de movimentação normal do software e possibilitando que operações especiais do sistema como: *teaching*, auto calibração, *JOG*, programação *off-line* baseada em CAD, e modificações de trajetória durante a execução do trabalho, possam sem utilizados.

Esta maior versatilidade de cinemáticas a serem utilizadas facilita a criação e desenvolvimento de novas geometrias e novos manipuladores, pois não será mais necessário re-desenvolver o gerador de movimento desde o início e sim somente desenvolver a transformação de soluções cinemática e implementá-la na plataforma proposta.

O desenvolvimento é prosseguimento dos trabalhos realizados por [Kienitz, 2012] e [JF et al., 2009], no estudo de sistemas de comando para máquinas automáticas de baixo custo. Também sendo sequência dos trabalhos de [De Lima et al., 2010] e [Bomfim et al., 2012], na área de tradução de cinemática offline para utilização de controladores CNC em robôs antropomórficos. Pode-se traçar um paralelo entre os trabalhos desenvolvidos por [De Lima et al., 2010] e [Bomfim et al., 2012], e a plataforma proposta no presente trabalho sendo que os três trabalhos tratam de tradução de soluções cinemáticas e comando de movimento.:

O primeiro ([De Lima et al., 2010]) desenvolveu um sistema de comando no qual a cinemática era resolvida no software de programação matemática Matlab. Enviando os comandos aos drivers de controle de movimento através de uma rede digital. Porém, o sistema não utilizava uma programação de movimento flexível e consolidada para os comandos. Isto dificultaria sua utilização em um processo fabril real, onde para traba-

lhos repetitivos, ou com programação *off-line*, é necessária a utilização de uma lista de comandos encadeadas para a realização de diversos movimentos em sequência;

Já o segundo ([Bomfim et al., 2012]) desenvolveu um sistema que traduz a solução da cinemática anteriormente ao comando de movimento, atuando entre o gerador de trajetória, o qual gera uma lista de comandos de movimento encadeados, e o interpretador de trajetória pelo comando de movimento. Para isto foi realizada a segmentação do movimento em semi-retas com um número menor de pontos, perdendo assim resolução de movimento e inserindo aberrações discretas de re-amostragem do movimento. Desta forma foi obtida uma transformação offline da cinemática cartesiana em um movimento de acordo com a solução cinemática específica do robô, e anterior à geração de movimento;

O presente trabalho desenvolve um tradutor atuando entre a geração de movimento e o driver de controle das juntas. Atuando diretamente no setpoint de movimentação das juntas, com o objetivo de manter a mesma resolução de discretização utilizada no sistema de comando. Propõem-se uma transformação em tempo real de espaço de juntas após a geração de movimento. Pode-se ver na Fig.1.1 os pontos de atuação das propostas anteriores e proposta deste trabalho. Como robô alvo para tradução foi escolhido um robô paralelo planar com uma configuração pantográfica de quatro barras e uma extensão serial para orientação da ferramenta.

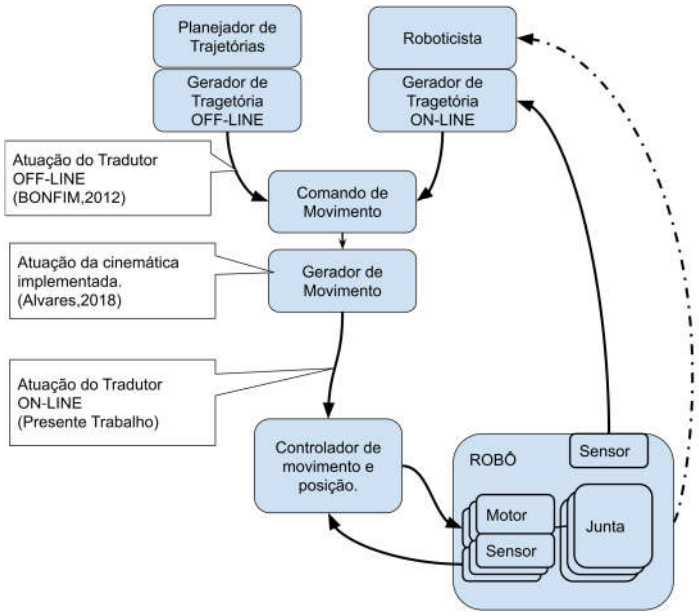

Figura 1.1. Descrição da estrutura de comando de uma sistema robótico

1.1 Trabalhos regressos

O desenvolvimento desta pesquisa baseia-se nos trabalhos realizados no LRSS (Laboratório de Robótica Soldagem e Simulação) no Departamento de Engenharia mecânica da Universidade Federal de Minas Gerais - UFMG. Estes têm início com o desenvolvimento de[Lages et al., 2003] e dando sequência por [Lima II et al., 2004] e [JF et al., 2009] tendo como efeito uma proposta de sistema de comando de movimento para utilização de comandos numéricos de baixo custo para reforma de robôs, porém não se observando a cinemática dos mesmos.

A implementação de [De Lima et al., 2010] demonstra o comando de movimento baseado em rede *CAN*, PLC e o Matlab. Traduzindo a solução da cinemática do robô *ABB IRB2000* para comandos a serem executados pelo PLC afim de que o robô executar o movimento desejado.

Utilizando da proposta de [JF et al., 2009] e acrescentando a tradução da cinemática no pós-processamento da trajetória, [Bomfim et al., 2012] gerou uma lista encadeada de segmentos de reta que realizam o movimento desejado por um robô *ASEA IRB6*, porém utilizando um software comando de movimento sem alterações em suas soluções cinemáticas.

Em paralelo [Kienitz, 2012] desenvolve um comando de movimento baseado em pulsos controlado por FPGAs e CPLDs (hardware eletrônico programável) para a implementação embarcada de comando de movimento para robôs de aplicação específica uma demanda demonstrada por [Ramalho Filho et al., 2010].

O trabalho de [Alvares et al., 2018] evolui, utilizando um sistema de comando de baixo custo. E realiza a implementação da solução cinemática de um robô *ASEA IRB6-S2* no Software de comando numérico aberto CNC Linux. Assim realizando a geração de movimento já com a cinemática correta do robô e sem uma segunda segmentação do movimento.

Como robô alvo utilizado neste trabalho será utilizado o robô publicado no artigo [Kienitz et al.,] que é um pantógrafo de quatro barras com uma extensão serial para correção do tcp em relação a orientação da ferramenta. Porém, assim como os artigos de [Ramalho Filho et al., 2010] e [Lima et al., 2020], não descreve o sistema de comando de movimento responsável pelo controle do robô estudado, por serem robôs não cartesianos e necessitarem de sistemas de comando que lidem com as cinemáticas específicas desenvolvidas para suas geometrias.

1.2 Justificativa

O desenvolvimento de robôs com uma geometria específica para uma classe de trabalhos, ou seja, projeto de um robô orientado ao trabalho, pode ser dificultado caso:

- Não esteja disponível uma plataforma flexível para o teste computacional das soluções de cinemáticas destes, a fim de se demonstrar a viabilidade, ou não, da solução pelo sistema de comando e controle;
- Não haja um sistema de comando de movimento flexível o qual essas soluções possam ser implementadas.

Assim justifica-se o desenvolvimento de ferramentas que possam traduzir soluções cinemáticas implementadas nos comandos de movimento padrões de mercado para soluções cinemáticas específicas dos robôs alvos em tempo real. Assim como o desenvolvimento de ferramentas de simulação e experimentação das soluções destas cinemáticas facilitando a validação das mesmas.

Em resumo a relevância do trabalho está no fato de sistemas de comando de máquias CNC se basearem no comando de máquinas de arquitetura cartesiana. Para o comando de robôs não cartesianos é necessária a tradução dos comandos de movimentos do espaço cartesiano para o espaço de juntas, levando à necessidade de alterações na programação do software de comando.

1.3 Objetivos

O objetivo do trabalho proposto é desenvolver a tradução de solução cinemática em tempo real atuando diretamente no setpoint de movimento e posição das juntas do robô. Para isso, será necessário que os seguintes objetivos específicos sejam alcançados:

- Escolha de geometria e desenvolvimento de solução de cinemáticas, ou seja, a escolha do robô e estudo da solução de sua cinemática.
- Escolha de uma arquitetura de comando e desenvolvimento de uma plataforma computacional simulada que represente esta.
- Escolha e desenvolvimento e implementação da solução em uma plataforma computacional real embarcada.
- Análise das influências dos parâmetros do sistema de comando na eficácia da solução desenvolvida.

2 Revisão Bibliográfica

Neste capitulo, serão colocados os conceitos e definições necessárias para o desenvolvimento da pesquisa e sua melhor compreensão.

2.1 Robôs Industriais e suas Cinemáticas

> "A definição oficial do que são Robôs vem do Instituto Americano de Robótica (RIA): Robô é um manipulador multifuncional reprogramável desenvolvido para mover materiais, partes, ferramentas ou dispositivos especiais por uma sequência de movimentos programados para executar uma variedade de tarefas."[Spong et al., 2006]

.

Em sua definição de robôs [Spong et al., 2006] também coloca o surgimento dos sistemas robóticos modernos como a fusão das máquinas CNC e das máquinas tele operadas, ambas desenvolvidas para a realização de funções de alta precisão ou de elevada perigosidade.

Este trabalho coloca robôs como máquinas multifuncionais desenvolvidas para movimentação de peças ou ferramentas através de uma programação para a realização de um trabalho.

2.1.1 Robôs cartesianos e Máquinas CNC

Os robôs cartesianos são aqueles constituídos somente por juntas prismáticas que coincidem com os eixos do espaço de trabalho como demonstrado na Fig. 2.1 e Fig. 2.2 Sendo assim o espaço de juntas e o espaço de trabalho geralmente são congruentes facilitando muito o entendimento de posicionamento pelo operador.

Essa característica pode ser observada também em máquinas de comando numérico onde o operador enxerga diretamente a posição e orientação da porta ferramenta, dos eixos de atuação, e do offset da ferramenta. Sua programação do trabalho pode ser executada

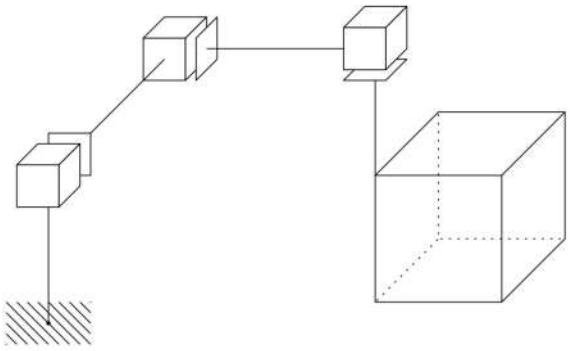

Figura 2.1. Figura da representação do espaço de trabalho de um robô cartesiano [Spong et al., 2006]

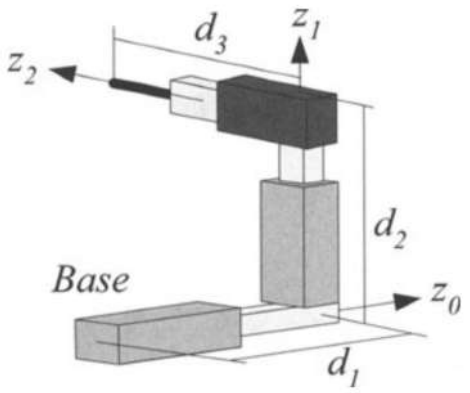

Figura 2.2. Figura da representação de um robô cartesiano de [Jazar, 2010]

de forma direta e no plano cartesiano, onde cada comando descreve o movimento explicito a ser executado pelas juntas do robô.

Soluções Cinemáticas diretas de robôs cartesianos

A cinemática direta dos robôs cartesianos é basicamente a conversão do movimento rotacional para movimento linear, pode-se dizer que é a cinemática dos atuadores pura, ou seja é um fator de conversão de Ângulo para comprimento equivalente a soma das reduções do atuador mecânico. Por exemplo, um motor acoplado com uma redução de polias 3:1 a um fuso de passo 5mm por revolução, logo 360° de rotação do motor equivale a 5mm/3, ou seja, são 0,00463mm de movimento linear para cara grau de movimento rotacional do motor. No caso de CNCs com eixos rotacionais a conversão é ainda mais simples sendo somente a razão de redução entre o motor e a junta, assim como no caso de uso de motores lineares onde não é necessária nenhuma a conversão.

Outra característica desse sistema é que ao se requerer a movimentação em um eixo

do espaço de trabalho somente um atuador realizará o movimento, ou seja, se é necessário realizar o movimento no eixo X de 1mm, somente o motor referente a esse eixo realizará o movimento angular equivalente a distância linear de 1mm.

Soluções Cinemáticas inversas de robôs cartesianos

A cinemática inversa de robôs cartesianos na maioria dos casos não é necessária, e se é, é na verdade um offset linear de posições entre as juntas e a ferramenta. Por exemplo: A posição $x = 10mm$ literalmente quer dizer que o eixo x deverá se posicionar a 10mm da referência sem que outros eixos sejam atuados; Ou sabendo que existe um offset de 2mm em x na ferramenta, ao se comandar a ferramenta para $x = 10mm$ o eixo X devera se movimentar até a posição 8mm. Para CNCs, mesmo com eixos rotacionais, esta afirmação continua válida, por exemplo: O comando para $x = 2mm$, $y = 3mm$ e $a = 10^o$, literalmente quer dizer que o eixo x deverá se movimentar 2mm o eixo y se movimentar 3mm e o eixo a girar em 10^o.

Resolução de Jacobianos de velocidade e aceleração

Os robôs cartesianos têm suas cinemáticas simplificadas, assim o cálculo das relações entre as velocidades e acelerações das juntas em função da velocidade e aceleração também o são. As velocidades e acelerações de cada eixo são desacopladas umas das outras assim a matriz jacobiana será identitária, ou seja, as forças e acelerações envolvidas são linearmente independentes.

2.1.2 Robôs Seriais não cartesianos

Robôs seriais são baseados em cadeias de links e atuadores em série, ou seja, a posição da junta posterior depende necessariamente da junta anterior. Exemplos comuns de robôs seriais não cartesianos são os antropomórficos Fig.2.3 e os SCARA Fig.2.4. O primeiro é mimético ao braço humano e por isso apresenta excelente flexibilidade de movimentação e trabalho; já o segundo foi desenvolvido especialmente para montagem de componentes em planos paralelos e demonstra-se especialmente compacto veloz e preciso.

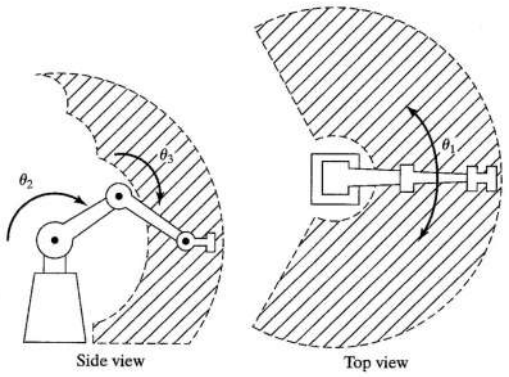

Figura 2.3. Figura da representação de um robô antropomórfico Fig. de [Craig, 2009]

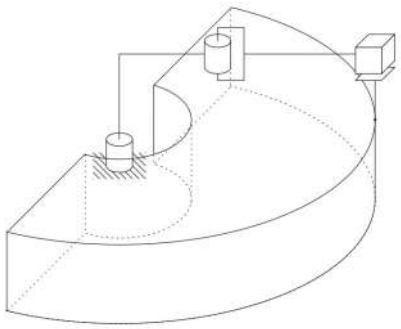

Figura 2.4. Figura da representação de um robô scara Fig. [Spong et al., 2006]

Descrição de cinemática: Denavit-Hartenberg

A utilização das definições de [Hartenberg & Denavit, 1955] para descrever os parâmetros cinemáticos de um robô serial não cartesiano é demonstrada por [Craig, 2009] e [Spong et al., 2006]. Convenciona quatro parâmetros para esta descrição da cinemática de cada link de um robô:

- $a_i =$ distância entre as juntas no eixo x_i comprimento do link

- $\alpha_i =$ ângulo entre os eixos $Z_i e Z_{i+1}$ medido em relação ao eixo x_i torção do link
- $d_i =$ distância entre as juntas no eixo z_i link offset
- $\theta_i =$ ângulo entre os eixos $x_{i-1} e x_i$ medido em relação ao eixo z_i ângulo de junta

Para definir estes parâmetros de forma a correta é necessário de posicionar o eixos XYZ_i em relação aos links de uma determinada forma. Para isso utiliza-se os seguintes passos:

- Identificar as juntas e os eixos de movimento das mesmas.
- Identificar a posição dos eixos das juntas onde existem relações perpendiculares comum entre eles.
- Posicionar o eixo Z da junta em questão de forma que ele coincida com o eixo de revolução da mesma.
- Posicionar o eixo X da junta em questão de forma que ele coincida com o eixo que conecta às juntas anterior e posterior, ou no caso de um link reto fique alinhado ao mesmo.
- Posicionar o eixo y de forma a completar o sistema da mão direita.
- Indicar i=0 para a base do robô.

Tomando com exemplo o robô de dois links rotativos da Fig.2.5 pode-se notar a aplicação das diretivas supracitadas.

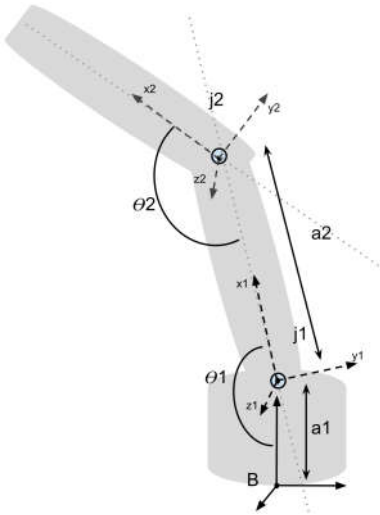

Figura 2.5. Descrição D-H das juntas de um robô

Posiciona-se os parâmetros dos links juntos em uma tabela organizada como pode-se ver na Tabela2.1:

$Link$	a_i	α_i	d_i	θ_i
1	a_1	0	0	θ_1
2	a_2	0	0	θ_2

Tabela 2.1. Tabela organizadora das variáveis da metodologia DH

Transformações homogêneas:

Uma transformação homogênea é a descrição da posição da junta em destaque em relação a junta anterior a esta acoplada, realizando então um encadeamento em série de movimentos e posições. Trata-se cada elo desse encadeamento como um sub-problema da solução da cinemática direta do robô. A resolução destas parcelas vem da utilização das quatro variáveis definidas por D-H para cada link. Assim, montando as matrizes necessárias para resolver o problema completo, denominado ${}^{i-1}_{i}T$ por [Craig, 2009]. Essa é a transformação homogênea e pode ser representada pelo produto de quatro transformações básicas [Spong et al., 2006]:

$$A_i = Rot_{z,\theta_i} \cdot Trans_{z,d_i} \cdot Trans_{x,a_i} \cdot Rot_{x,\alpha_i} \tag{2.1}$$

Onde:

$$Rot_{z,\theta_i} = \begin{bmatrix} c_{\theta_i} & -s_{\theta_i} & 0 & 0 \\ s_{\theta_i} & c_{\theta_i} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.2}$$

$$Trans_{z,d_i} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.3}$$

$$Trans_{x,a_i} = \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.4}$$

$$Rot_{x,\alpha_i} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c_{\alpha_i} & -s_{\alpha_i} & 0 \\ 0 & s_{\alpha_i} & c_{\alpha_i} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.5}$$

Assim:

$$A_i = \begin{bmatrix} c_{\theta_i} & -s_{\theta_i}c_{\alpha_i} & s_{\theta_i}s_{\alpha_i} & a_i c_{\theta_i} \\ s_{\theta_i} & c_{\theta_i}c_{\alpha_i} & -c_{\theta_i}s_{\alpha_i} & a_i s_{\theta_i} \\ 0 & s_{\alpha_i} & c_{\alpha_i} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.6}$$

A matriz A_i configura uma função de uma variável só, geralmente θ_i para juntas rotativas ou d_i para juntas prismáticas. Logo têm-se A_i como a solução da cinemática direta da junta em evidência. E sendo atendidas as restrições do procedimento D-H, a solução da cinemática direta de um robô serial é produto das i matrizes A_i.

$$T_i^0 = A_1 \times A_2 \times ...A_i \tag{2.7}$$

Soluções Cinemáticas diretas de robôs seriais

A solução das cinemáticas diretas de robôs seriais baseia-se em métodos analíticos, dentre estes destaca-se especialmente o método de D-H. A seguir, demonstra-se, a solução da cinemática direta de um robô com duas juntas rotacionais em série e perpendiculares ao eixo dos links representado na Fig.2.6. A aplicação da metologia D-H se inicia pela organização das variáveis na Tabela2.2 onde * indica as variáveis manipuladas nas juntas.

Link	a_i	α_i	d_i	θ_i
1	a_1	0	0	θ_1^*
2	a_2	0	0	θ_2^*

Tabela 2.2. Tabela de variáveis DH para um robô 2 links RR

$$A_1 = \begin{bmatrix} c_1 & -s_1 & 0 & a_1 \cdot c_1 \\ s_1 & c_1 & 0 & a_1 \cdot s_1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.8}$$

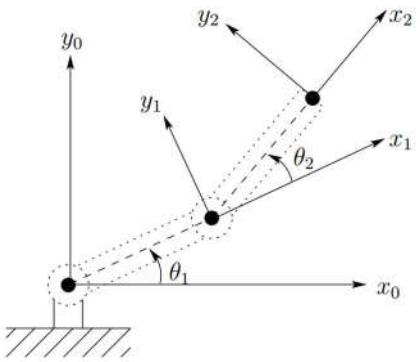

Figura 2.6. Figura da representação de um robô cartesiano de [Spong et al., 2006]

$$A_2 = \begin{bmatrix} c_2 & -s_2 & 0 & a_2 \cdot c_2 \\ s_2 & c_2 & 0 & a_2 \cdot s_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.9}$$

As transformações homogêneas se dão por:

$$T_1^0 = A_1 \tag{2.10}$$

$$T_2^1 = A_2 \tag{2.11}$$

$$T_2^0 = A_1 \times A_2 \tag{2.12}$$

$$T_2^0 = \begin{bmatrix} c_{12} & -s_{12} & 0 & a_1 \cdot c_1 + a_2 \cdot c_{12} \\ s_{12} & c_{12} & 0 & a_1 \cdot s_1 + a_2 \cdot s_{12} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.13}$$

Onde $sen(\theta_1) = s_1$ e assim para c_1, s_2c_2 e $sen(\theta_1 + \theta_2) = s_{12}$ o mesmo para c_{12}.

Com efeito, o resultado analítico é:

$$x = a_1 \cdot c_1 + a_2 \cdot c_{12} \tag{2.14}$$

$$y = a_1 \cdot s_1 + a_2 \cdot s_{12} \tag{2.15}$$

Soluções Cinemáticas inversas de robôs seriais

A solução cinemática inversa é responsável por informar a configuração do robô para que esse posicione a ferramenta em um dado ponto. O problema geral é descrito por [Spong et al., 2006] utilizando uma transformação homogênea $4x4$ baseada nos ângulos de Euler:

$$H = \begin{bmatrix} R & o \\ 0 & 1 \end{bmatrix} \tag{2.16}$$

Onde:

$$H = A_1(q_1) \times A_2(q_2) \times ...A_n(q_n) \tag{2.17}$$

Sendo que H representa a configuração desejada da ferramenta, logo basta encontrar os valores corretos de q_n. Porém, a solução de H se desdobra um sistema de 12 equações não lineares com n variáveis a serem encontradas onde geralmente quatro são triviais e o restante não triviais. Como citado por [Spong et al., 2006] resolver esse sistema é algo complexo e trabalhoso para ser resolvido diretamente. Então o mesmo explora técnicas a serem utilizadas para encontrar uma solução viável para a cinemática inversa. E essa solução se existir deve ser única, viável e utilizável e não necessariamente uma solução matemática geral fechada. Uma solução viável depende bastante de como as limitações das juntas são descritas. Por exemplo, [Spong et al., 2006] sugere limitar as juntas sem giro livre para ângulos menores do que 360° de rotação, isso nos ajuda a garantir que a solução H seja fisicamente possível de ser atingida pelo manipulador.

Solução de Cinemática Inversa por Desacoplamento cinemático: O desacoplamento cinemático consiste em separar o problema da cinemática inversa em dois problemas separados: Um para resolver a posição do efetuador e outro para resolver a orientação do mesmo. Sendo então a cinemática inversa de posição e cinemática inversa de orientação como exemplificado por [Spong et al., 2006] e demonstrado no Fig.2.7.

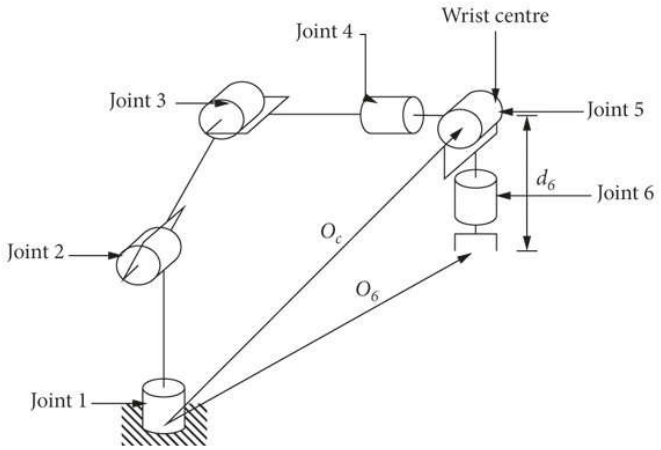

Figura 2.7. Demonstração de desacoplamento cinemático de [Spong et al., 2006]

Solução da cinemática inversa de posição: A solução da cinemática inversa de posição é primeira etapa da resolução do desacoplamento cinemático descrito por [Spong et al., 2006] e trata-se de calcular a posição do ponto onde os eixos do efetuador se encontram ,o_c. A descrição dos eixos a responsáveis pela posição pode ser observada na Fig. 2.8.

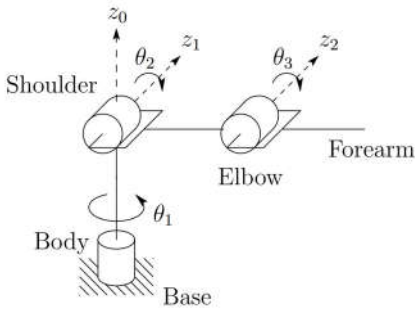

Figura 2.8. Descrição das juntas responsáveis pela posição de [Spong et al., 2006]

Então separa-se as variáveis relativas a posição e orientação obtendo-se então a seguinte equação:

$$H = A_1(q_1) \times A_2(q_2) \times ...A_n(q_n) \tag{2.18}$$

ou

$$H = T_n^0(q_1, q_2, ...q_n) \tag{2.19}$$

e sim duas equações:

$$R = R_n^0(q_1, q_2, ...q_n) \tag{2.20}$$

$$o = o_n^0(q_1, q_2, ...q_n) \tag{2.21}$$

Onde: R e o são respectivamente a posição e a orientação do efetuador no plano de trabalho.

Define-se então quais juntas alteram ou não posição e orientação. Por exemplo [Spong et al., 2006] cita que em um manipulador de 6 eixos antropomórfico, os eixos $z_3, z_4 e z_5$ ($z_5 = z_6$) que interceptam o ponto o_c no punho, não alteram a posição do efetuador e sim só sua orientação. Já as três primeiras juntas serão responsáveis pela posição do efetuador, mas não sua orientação.

Na matriz de transformação homogênea a terceira coluna representa a direção do Z do efetuador e assim tem-se:

$$o_c^0 = o - d_i \cdot R \cdot \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{2.22}$$

Onde tem-se que: d_i é a distância entre a ferramenta e o ponto o_c^0; E a orientação do plano $o_i x_i y_i z_i$ é dado por R.

As componentes do ponto o são denominados por [Spong et al., 2006] como: $o_x o_y o_z$, e as componentes do ponto O_c^0 como: $x_c y_c z_c$. Com efeito obtém-se a seguinte relação:

$$\begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = \begin{bmatrix} 0_x - d_i \cdot r_{13} \\ 0_y - d_i \cdot r_{23} \\ 0_z - d_i \cdot r_{33} \end{bmatrix} \tag{2.23}$$

onde $(r_j i)$ são elementos de R.

Assim pode-se encontrar transformação de rotação R_i^0 dependente somente das juntas anteriores ao ponto O_c^0. E com efeito encontrar a orientação da ferramenta relativa ao plano $o_i x_i y_i z_i$ pela expressão:

$$R = R_i^0 \cdot R_n^i \tag{2.24}$$

logo:

$$R_n^i = (R_i^0)^{-1} \cdot R = (R_i^0)^T \cdot R \tag{2.25}$$

Solução da cinemática inversa de Orientação: A solução da cinemática inversa de orientação é a segunda etapa do desacoplamento cinemático, inicia-se sua solução calculando os ângulos de Euler do efetuador no ponto o_c^0. Na Fig. 2.9, o eixo "Yaw"representa $_4$, "Pitch"o $_5$ e "Roll"$_6$.

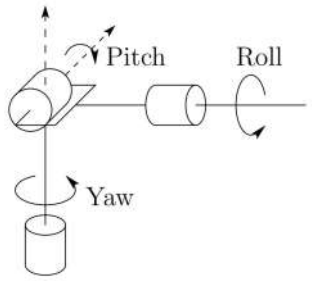

Figura 2.9. Descrição das juntas responsáveis pela orientação do efetuador de [Spong et al., 2006]

$$\theta_4 = \phi$$
$$\theta_5 = \theta$$
$$\theta_6 = \psi$$

. (2.26)

$$R_{XYZ} = R_{z,\phi} R_{y,\theta} R_{z,\psi} \tag{2.27}$$

$$R_{XYZ} = \begin{bmatrix} c_\phi & -s_\phi & 0 \\ s_\phi & c_\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} c_\theta & 0 & s_\phi \\ 0 & 1 & 0 \\ -s_\phi & 0 & c_\theta \end{bmatrix} \times \begin{bmatrix} c_\psi & -s_\psi & 0 \\ s_\psi & c_\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.28}$$

$$R_{XYZ} = \begin{bmatrix} c_\phi c_\theta c_\psi - s_\psi s_\psi & -c_\phi c_\theta s_\psi - s_\psi s_\psi & c_\psi s_\theta \\ s_\phi c_\theta c_\psi + c_\psi s_\psi & -s_\phi c_\theta s_\psi + c_\psi c_\psi & s_\psi s_\theta \\ -s_\phi c_\psi & s_\theta s_\psi & c_\theta \end{bmatrix} \tag{2.29}$$

Pode-se então utilizar a solução dos arcos tangentes de Euler:

$$\theta = atan2(r_{33}, \pm\sqrt{1 - r_{33}^2}) \tag{2.30}$$

$$\phi = atan2(\pm r_{13}, \pm r_{23}) \tag{2.31}$$

$$\psi = atan2(\pm r_{31}, \pm r_{32}) \tag{2.32}$$

Método geométrico para resolução da cinemática inversa de posição: A parcela da posição da matriz o_c^0 é relativamente mais simples de ser resolvida com a abordagem geométrica, encontrando as vaiáveis $q_1 q_2 q_3$. Porém, esta abordagem é limitada a cinemáticas relativamente simples. A complexabilidade dos manipuladores cresce com o número de links com parâmetros D-H diferentes de zero, como citado por [Spong et al., 2006]. Por exemplo um robô serial com mais de três links que atuam sobre a posição do ponto O_c^0 dificilmente terão uma solução geométrica. Quando os termos a_i, d_i são iguais a zero e α_i é zero ou $\pm\pi/2$ a solução geométrica geralmente é a mais simples, e é baseada em realizar as projeções do manipulador em $x_{i-1} - y_{i-1}$ como um problema de trigonometria achando assim a variável q_i.

Por exemplo um robô antropomórfico de 5 eixos dificilmente teria sua cinemática inversa resolvida diretamente, porém ao separar posição do efetuador (eixos 1,2,3) da orientação do efetuador (eixos 4,5) é possível resolver a posição pelo método geométrico e a orientação pela inversa da transformação homogênea.

Método numérico para resolução da cinemática inversa: Quando a complexibilidade do robô impede que sua cinemática inversa possa ser resolvida por métodos analíticos, métodos de resolução numérica devem ser utilizados. Basicamente realiza-se a modelagem matemática do robô e resolve-se a cinemática direta do mesmo. Então são aplicados valores sistêmicos a cada variável da cinemática direta e calcula-se a posição esperada do efetuador para cada conjunto de variáveis. Assim monta-se uma tabela com as configurações possíveis do robô (Espaço de configurações) e o estado do efetuador correspondente para cada configuração. Com essa tabela de correspondências pode-se partir para dois tipos de abordagem, ou a modelagem polinômios baseados nas entradas e saídas ou a utilização da tabela bruta. Os polinômios têm a vantagem de se diminuir o custo de memória do controlador do robô, porém podem ser de alto custo computacional e a precisão do controle depende do quão bem ajustados foram os parâmetros; já a tabela bruta tem uma resolução de configurações finitas e têm alto uso de memória.

2.1.3 Robôs Paralelos

A definição geral de robôs paralelos segundo [Merlet, 2006] é: Manipulador que tem seu atuador final de força ligado à sua base por várias cadeias fechadas de cinemática independentes. Essa definição é muito abrangente e o próprio [Merlet, 2006] também cita que algumas características são interessantes para se facilitar o estudo dos robôs paralelos, estas são:

- Quando existem mais cadeias cinemáticas do que graus de liberdade de movimento do robô, criando assim uma configuração redundante;
- Um manipulador paralelo precisa de pelo menos duas cadeias de cinemáticas ligando o efetuador a base, com cada uma dessas cadeia sendo ativamente atuada;
- O número de atuadores ativo é no mínimo o mesmo do número de graus de liberdade de movimento do efetuador;
- Ao se travar os atuadores a o manipulador não pode ter movimento livre;

Ainda dentro das definições e classificação de robôs totalmente paralelos [Merlet, 2006] separa estes em duas classes maiores: robôs planares e o caso geral. Robôs paralelos planares geralmente são constituídos de um número de atuadores igual ao número de graus de liberdade com dois ou três atuadores; já o caso geral é aplicado em casos em que o robô tem m graus de liberdade sendo que se n_1 é o número de links e n_2 é p número de graus de liberdade das juntas em cada cadeia, logo:

$$m = 6 + 6mn_1 - 5mn_2 \tag{2.33}$$

Este trabalho trata somente de robôs paralelos planares por simplificação, mas a metodologia não tem restrições a ser aplicada no caso geral.

Solução de Cinemáticas inversas em robôs paralelos

A resolução de cinemáticas inversas de robôs paralelos como definição não se difere da solução da cinemática inversa de robô seriais, por [Merlet, 2006] ela consiste em encontrar para uma certa pose de um corpo rígido e o conjunto de parâmetros que definem esta pose em relação a o ponto de origem do plano cartesiano.

Solução de cinemáticas inversas pelo método analítico em robôs paralelos: Partindo de um robô planar onde o início das cadeias é representado A é ligado a base e o final das cadeias ligado a atuador de força, punho ou plataforma, a ser movimentada

e representado por B. Assim as coordenadas de A são fixas e conhecidas, e que B são variáveis a serem encontradas e definem a pose da plataforma. Logo o vetor AB é definido pela equação:

$$AB = AO + OB = H_1(X) \tag{2.34}$$

Onde:

- AB É o vetor entre o ponto fixo do atuador na base e o ponto móvel na plataforma, atuador final.
- AO É o vetor entre o ponto fixo do atuador e a origem (referência).
- OB É o vetor entre o ponto móvel do atuador na plataforma e o ponto de origem.

Estas definições podem ser observadas de forma gráfica na Fig. 2.10:

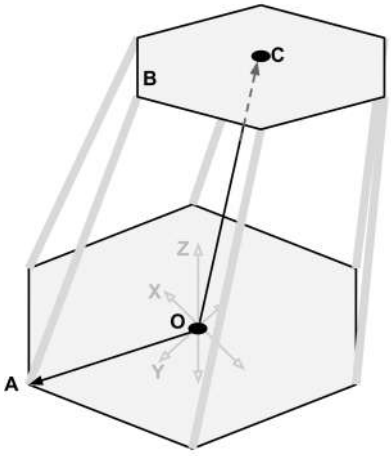

Figura 2.10. Descrição dos pontos A,B,C,O de [Merlet, 2006]

Assim obtêm-se os pontos das extremidades atuadas da plataforma. As coordenadas nas juntas também podem ser necessárias para o cálculo de X e neste caso a solução para encontrar essas se dá por:

$$H_1(X) = H_2(X, \Theta) \tag{2.35}$$

Solução de cinemáticas inversas pelo método geométrico em robôs paralelos: O método geométrico para a solução da cinemática inversa, proposto por [Merlet, 2006] baseia-se em: Considerar que as extremidades A, B de cada link são conhecidas no espaço de referência 3d, então definir um ponto médio M dos links separando-os em dois mecanismos M_A e M_B definindo duas cadeias A, M e B, M. A aplicação deste método no exemplo da Fig. 2.10 pode ser observado na Fig. 2.11.

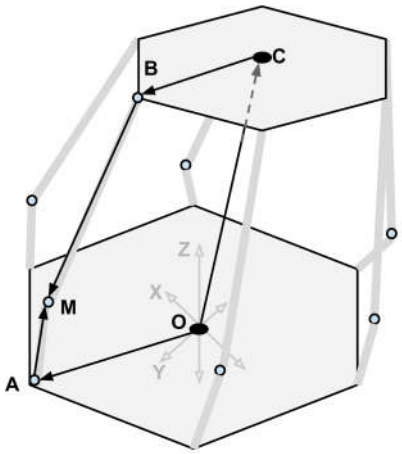

Figura 2.11. Descrição dos pontos A,B,C,O,M de [Merlet, 2006]

A movimentação livre das juntas destas duas cadeias fica neste ponto médio. Sendo que M pertencente a M_i e sua variação é algébrica e definida pela distância d_i. A solução da cinemática inversa será a intercessão destas distâncias, formando assim um sistema de equações que deverá ser linearmente independente para que o controle do robô seja possível.

Para um robô paralelo planar baseado em juntas rotacionais, dada uma posição do ponto C e rotação ϕ, é preciso calcular a rotação a posição e rotação dos pontos B_i pela equação:

$$AB = AO + OC + RCB_r = H1(X) \tag{2.36}$$

Onde CB_r é a coordenada conhecida de B_i no efetuador. Então B_i pertence a uma cadeia serial A_i, M_i, B_i com as variáveis de θ e α, logo a equação se torna:

$$AB = AO + OM_i(\theta_i) + M_iB_i(\alpha_i) = H2(X, \phi) \tag{2.37}$$

Então para a solução da cinemática inversa utiliza-se a Eq. 2.35 com as variáveis θ_i e α_i. Então pelo método geométrico escolhendo o ponto M_i o mecanismo A_i, M_i naturalmente impõem a restrição de M representando um círculo com o centro em A_i e raio l_1^i. Igualmente o mecanismo B_i, M_i também forma um círculo de centro B_i com

raio l_2^i. Finalizando a solução do sistema será a intercessão destes dois círculos. Esta representação geométrica pose ser observada na Fig 2.12.

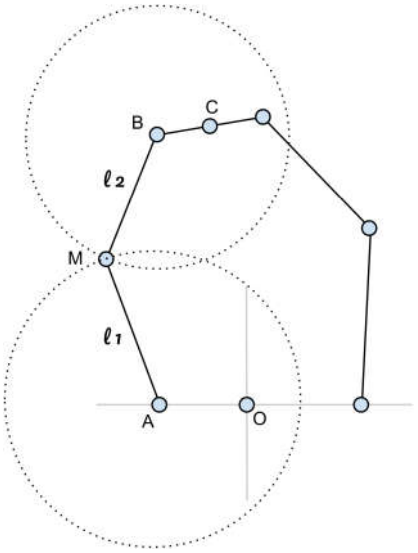

Figura 2.12. Solução geométrica de robôs planares por [Merlet, 2006]

Método Khalil-Klenfinger ou Denavit-Hartenberg Modificado, MDH: Para robôs paralelos pode-se utilizar adaptação do método D-H para a resolução de sua cinemática como demonstrado por [Briot et al., 2015]. Utilizando assim o MDH para reduzir as ambiguidades geradas pela aplicação dos parâmetros de D-H para resolução de cadeias ramificadas e fechadas.

Parametrização de cadeias ramificadas por MDH: Uma estrutura ramificada consiste em uma cadeia cinemática com $n+1$ links sejam esses reais ou virtuais. Para resolver algumas das ambiguidades o MDH aplica o conceito de links e juntas virtuais sobrepostas a elementos reais, enumerados por $\mathfrak{B}_\mathrm{i}$ que antecede $\mathfrak{B}_\mathrm{j}$. Cada link tem um plano fixo local $\mathfrak{F}_\mathrm{i}$ e uma origem local O_i e seguem o seguinte conjunto de regras:

- O plano $\mathfrak{F}_\mathrm{i} = (O_i, x_i, y_i, z_i)$ é fixo no link $\mathfrak{B}_\mathrm{i}$;
- O eixo z_i é ao longo da junta i
- O eixo x_i pode ser:
  - Quando serial: Perpendicular a Z_i e Z_j
  - Quando ramificado: Escolher um dos links sucessores para ser perpendicular. No caso dos outros links um vetor adicional u_j fixo em $\mathfrak{B}_\mathrm{i}$ será criado.

- O eixo y_i é definido pela regra da mão direita.

Após a definição dos planos dos links são definidas as variáveis MDH e estas são:

- Y_i Ângulo entre x_i e u_j em relação z_i;
- b_j Distância entre x_i e u_j no eixo z_i;
- α_j Ângulo entre z_i e z_j em relação a u_j;
- d_j Distância entre z_i e z_j no eixo u_j;
- θ_j ângulo entre u_j e x_j em relação z_j;
- r_j distancia entre u_j e x_j no eixo z_j;

Caso o eixo x_i seja perpendicular a ao eixo z_j o vetor adicional u_j não precisa ser definido e seus valores serão nulos reduzindo os parâmetros necessários a somente os quatro parâmetros D-H tradicionais (α_j, d_j, θ_j e r_j).

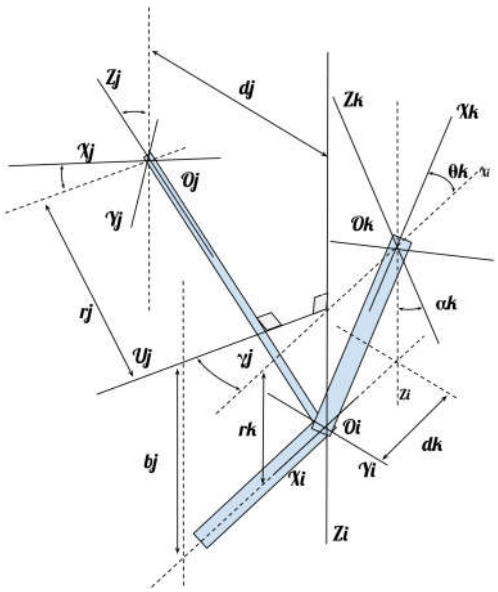

Figura 2.13. Representação isométrica da descrição MDH

A transformação homogênea utilizando os parâmetros supracitados é descrita por [Briot et al., 2015] nos seguintes equações:

$$^{i}T_j = Rot_{(z,y_i)} \times Trans_{(z,b_j)} \times Rot_{(a,\alpha_j)} \times Trans_{(x,d_j)} \times Rot_{(z,\theta_j)} \times Trans_{(z,r_j)} \quad (2.38)$$

$$
{}^{i}T_{j} = \begin{bmatrix} C_{y_j}C_{\theta_j} - S_{y_j}C_{\alpha_j}S_{\theta_j} & -C_{y_j}S_{\theta_j} - S_{y_j}C_{\alpha_j}C_{\theta_j} & S_{y_j}S_{\alpha_j} & d_jC_{y_j} + r_jS_{y_i}S_{\alpha_j} \\ S_{y_j}C_{\theta_j} + C_{y_j}C_{\alpha_j}S_{\theta_j} & -S_{y_j}S_{\theta_j} - C_{y_j}C_{\alpha_j}C_{\theta_j} & -C_{y_j}S_{\alpha_j} & d_jS_{y_j} - r_jC_{y_i}S_{\alpha_j} \\ S_{\alpha_j}S_{\theta_j} & S_{\alpha_j}C_{\theta_j} & C_{\alpha_j} & r_jC_{\alpha_j} + b_j \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.39}
$$

$$
{}^{i}T_{j} = \left| \begin{array}{cccc} & & & d_jC_{y_j} + r_jS_{y_i}S_{\alpha_j} \\ & {}^{i}R_j & & d_jS_{y_j} - r_jC_{y_i}S_{\alpha_j} \\ & & & r_jC_{\alpha_j} + b_j \\ 0 & 0 & 0 & 1 \end{array} \right| \tag{2.40}
$$

Logo a transformação inversa se dará por:

$$
{}^{i}T_{j}^{-1} = {}^{j}T_{i} = \left| \begin{array}{cccc} & & & -d_jC_{\theta_j} - b_jS_{\theta_i}S_{\alpha_j} \\ & {}^{i}R_j^T & & d_jS_{\theta_j} - b_jC_{\theta_i}S_{\alpha_j} \\ & & & -b_jC_{\alpha_j} - r_j \\ 0 & 0 & 0 & 1 \end{array} \right| \tag{2.41}
$$

Utilização do MDH em soluções cinemáticas fechadas e paralelas: Quando a cinemática tem mais juntas (reais ou virtuais) do que links então, provavelmente sua cinemática contêm elos fechados com redundância de atuação ou não. Para utilizar o MDH proposto por [Briot et al., 2015] em sistemas de cadeias de cinemática fechadas é necessária uma variável adicional para informar se a junta é ativa ou passiva.

$$
\mu_j = 1 \text{ se a junta j por ativa} \tag{2.42}
$$

$$
\mu_j = 0 \text{ se a junta j for passiva} \tag{2.43}
$$

São necessários também a adição de planos virtuais adicionais às juntas existentes para que seja possível solucionar as equações de cadeias fechadas, para isso utiliza-se os seguintes conceitos para estes:

- Estabelecer uma estrutura ramificada equivalente: Selecionar uma das juntas da cadeia, se possível passiva e conectada ao efetuador, nessa faz-se um corte virtual adicionando um plano virtual.

- Definição do plano do primeiro link ligados a junta virtual: para cada corte de junta k adicionar um plano $\mathfrak{F}_{\mathfrak{k}}$ fixo no primeiro link da junta o qual denomina-se j. O eixo Z_k é disposto colinear ao eixo da junta k e x_k é normal a z_k e z_j simultaneamente.

O outro link adjacente a k denomina-se i e $i = a(k)$ e a transformação entre $\mathfrak{F}_{\mathfrak{k}}$ e $\mathfrak{F}_{\mathfrak{i}}$ será obtida com os parâmetros: $y_k, b_k, \alpha_k, d_{k,k}\, er_k$.

- Definição do plano da junta cortada em relação ao link adjacente i: O plano $\mathfrak{F}_{\mathfrak{k}}$ é fixo no link j logo a transformação $\mathfrak{F}_{\mathfrak{k}}$ para $\mathfrak{F}_{\mathfrak{j}}$ é constante. [Briot et al., 2015] define essa transformação como ${}^{j}T_{k+B}$, onde B é o número de cadeias fechadas e $j = a(k + B)$. O plano $\mathfrak{F}_{\mathfrak{k}+\mathfrak{B}}$ é alinhado com o plano $\mathfrak{F}_{\mathfrak{k}}$ de forma a $Z_{k+B} = Z_k$ na junta de corte, e $X_{k+B} = X_k$ simplificando assim os parâmetros dos planos de corte ${}^{j}T_{k+B}$ em $y_{k+B} e b_{k+B}$. Permitindo assim alinhar X_j com X_{k+B}, α_{k+B} e d_{k+B} vão alinhar Z_j com Z_{k+B} e θ_{k+B} e r_{k+B} serão nulos.

Metodologia de solução de cinemática inversa de um manipulador paralelo utilizando MDH: .

Premissas Iniciais:

- Manipulador com uma base fixa, nomeada como $\mathfrak{B}_{\mathfrak{o}}$;
- O plano de referência de $\mathfrak{B}_{\mathfrak{o}}$ será também o plano de trabalho $\mathfrak{F}_{\mathfrak{o}}(O, x_0, y_0, z_0)$;
- O efetuador é nomeado como $\mathfrak{B}_{\mathfrak{p}}$ ligado a n cadeias cinemáticas;
- Cada cadeia cinemática pode ser serial, ramificada ou fechada, e é composta por m_j juntas;
- Cada junta m_j é localizada nos pontos A_{ij} onde $i = \{1, ..., n\}$ e $j = \{1, ..., m_i\}$;
- Cada junta tem seu plano denominado $\mathfrak{B}_{\mathfrak{ij}}$ localizado no ponto A_{ij} e parametrizado na MDH pelos parâmetros q_{ij};

Equacionamento do problema:

- As variáveis das juntas atuadas são organizadas no vetor q_a;
- As variáveis das juntas passivas são organizadas no vetor q_d;
- AS variáveis de posição do efetuador $\mathfrak{F}_{\mathfrak{p}}(P, x_p, y_p, z_p)$ serão organizadas no vetor X_p onde somente as variáveis de graus de liberdade serão independentes e são denominadas por X e correlacionadas por $C_p = (X, X_p) = 0$;
- O tamanho do vetor q_a deverá ser igual ou maior aos graus de liberdade do efetuador;

Assim o problema da cinemática inversa consiste em encontrar os valores de q_a como funções de X:

$$q_a = \mathfrak{H}(X)$$

Para robôs paralelos usuais esse problema é relativamente simples de ser resolvido de forma analítica (*closed form solution*) como descrito por[Briot et al., 2015]. A abordagem do problema consiste em separar as cadeias cinemáticas que ligam a base ao efetuador em i cadeias seriais do ponto O ao ponto P. Utilizando a seguinte transformação:

$${}^0T_p = {}^0T_{i1(q_{i1})} \cdot \prod_{j=2}^{m_i} ({}^{i(j-1)}T_{ij(q_{ij})}) \cdot {}^{im_i}T_p \tag{2.44}$$

Para: $i = \{1, ..., n\}$ Onde:

- ${}^0T_p(X)$ é a matriz de transformação homogênia de $\mathfrak{F}_0$ para $\mathfrak{F}_p$:

$${}^0T_p = \begin{vmatrix} & & & x \\ & {}^0R_p & & y \\ & & & z \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

  E x, y, z são os componentes de translação de X_p;

- ${}^0T_{i1(q_{i1})}$ é a matriz de transformação homogênea entre a base do robô: $\mathfrak{F}_0$ e o primeiro link: $\mathfrak{F}_1$;

- ${}^{i(j-1)}T_{ij(q_{ij})}$ é a matriz de transformação homogênea entre um link subsequente e adjacente, sendo que q_{ij} é constante para este caso.

- ${}^{im_i}T_p$ é a matriz de transformação homogênea entre o último link e o efetuador;

Aplicação da metodologia MDH a um robô planar com dois graus de liberdade: A Descrição do robô dado como exemplo pode ser visualizada na Fig. 2.14:

- A primeira cadeia é constituída de 3 juntas rotativas paralelas, localizadas em $A_11, A_12 e A_13$ sendo somente a junta A_11 ativa;

- A segunda cadeia é construída de duas juntas rotativas paralelas localizadas em $A_21 e A_22$ sendo somente a junta A_12 ativa;

- O vetor das coordenadas ativas é $q_a^T = [q_1 1 q_2 1]$;

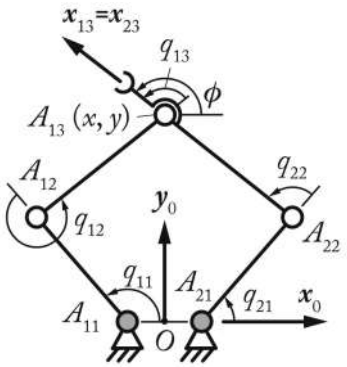

Figura 2.14. Robô 5 barras de [Briot et al., 2015]

- O efetuador está localizado em A_13 e suas cordeadas serão x, y

  Os parâmetros MDH são:

ij	$a(ij)$	μ_{ij}	σ_{ij}	γ_{ij}	b_{ij}	α_{ij}	d_{ij}	θ_{ij}	r_{ij}
11	0	1	0	0	0	0	$d_{11} = -l_{OA_{11}}$	q_{11}	0
12	11	0	0	0	0	0	$d_{12} = l_{A_{11}A_{12}}$	q_{12}	0
13	12	0	0	0	0	0	$d_{13} = l_{A_{12}A_{13}}$	q_{13}	0
21	0	1	0	0	0	0	$d_{21} = l_{OA_{21}}$	q_{21}	0
22	21	0	0	0	0	0	$d_{22} = l_{A_{21}A_{22}}$	q_{22}	0
23	22	0	2	0	0	0	$d_{23} = l_{A_{22}A_{13}}$	0	0

Figura 2.15. Parâmetros MDH para solução de uma robô 5 barras Tabela de [Briot et al., 2015]

Pose-se notar que os planos $\mathfrak{F}_23$ e $\mathfrak{F}_13$ são coincidentes porém com antecedentes diferentes.

A equação relacional $C_p = (X, X_p) = 0$ descreve a relação entre o ângulo ϕ de orientação do efetuador e suas coordenadas x, y e pode ser descrita como:

$$tan\phi = \frac{y - yA_{22}}{x - A_{22}} = \frac{y - d_{22} \cdot senq_{21}}{x - d_{21} - d_{22} \cdot cosq_{21}} \tag{2.45}$$

Com isso, $[xA_{22}yA_{22}]^T$ é a posição do ponto A_{22} em relação a origem. E determina-se q_{21} como função de x, y.

Tem-se que:

$${}^0T_p = \begin{bmatrix} C\phi & -S\phi & 0 & x \\ S\phi & C\phi & 0 & y \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.46}$$

Sendo assim:

$$\left|\begin{matrix} x \\ y \end{matrix}\right| = \left|\begin{matrix} d_{i1} + d_{i2} \cdot Cq_{i1} + d_{i3} \cdot C(q_{i1} + q_{i2} \\ d_{i2} \cdot Sq_{i1} + d_{i3} \cdot S(q_{i1} + q_{i2}) \end{matrix}\right| \tag{2.47}$$

Para: $i = \{1, 2\}$ $d_{23} = l_{A_{22}A_{13}}$ $\phi = q_{11} + q_{12} + q_{13}$ p/ cadeia 1 $\phi = q_{21} + q_{22}$ p/ cadeia 2 logo:

$$\left|\begin{matrix} x - d_{i1} - d_{i2} \cdot Cq_{i1} \\ y - d_{i2} \cdot Sq_{i1} \end{matrix}\right| = \left|\begin{matrix} d_{i3} \cdot C(q_{i1} + q_{i2} \\ d_{i3} \cdot S(q_{i1} + q_{i2}) \end{matrix}\right| \tag{2.48}$$

Equacionando os dois lados e elevando ao quadrado as partes, obtém-se as equações das cadeias:

$$h_p(X, q_a) = \left|\begin{matrix} (x - d_{i1} - d_{12} \cdot Cq_{12})^2 + (y - d_{12} \cdot Sq_{11})^2 - d_{13}^2 \\ (x - d_{i1} - d_{21} \cdot Cq_{22})^2 + (y - d_{22} \cdot Sq_{21})^2 - d_{23}^2 \end{matrix}\right| = 0 \tag{2.49}$$

onde: $a_i = -2d_{i2}(x - d_{i1})$ $b_i = -2d_{i2y}$ $c_i = (x - d_{i1})^2 + y^2 + d_{i2}^2 - d_{i3}^2$

Aplicando a formula de arco metade:

$$Cq_{ij} = \frac{1 - t_{ij}^2}{1 + t_{ij}^2} \tag{2.50}$$

$$Sq_{ij} = \frac{2 \cdot t_{ij}}{1 + t_{ij}^2} \tag{2.51}$$

onde: $t_{ij} = tan(\frac{q_{ij}}{2})$

Aplicando isto a equação de cadeia obtém-se:

$$h_p(X, q_a) = \left|\begin{matrix} (c_1 - a_1)t_{11}^2 + 2b_1t_{11} + (a_1 + c_1) \\ ((c_2 - a_2)t_{21}^2 + 2b_2t_{21} + (a_2 + c_2) \end{matrix}\right| = 0 \tag{2.52}$$

Logo:

$$t_{i1} = \frac{-b_i \pm \sqrt{b_i^2 - c_i^2 + a_i^2}}{c_i - a_i} \tag{2.53}$$

Com efeito:

$$q_{i1} = 2tan^{-1}\left(\frac{-b_i \pm \sqrt{b_i^2 - c_i^2 + a_i^2}}{c_i - a_i}\right) \tag{2.54}$$

Sendo que o $\pm$ na equação significa que existem dois modos de trabalho para cada cadeia, isso porque do ponto de vista geométrico essas equações são o equivalente a resolver

a intercepção de dois círculos, que fora de simetrias e singularidades resultam em dois pontos. Como nota-se na Fig.2.16.

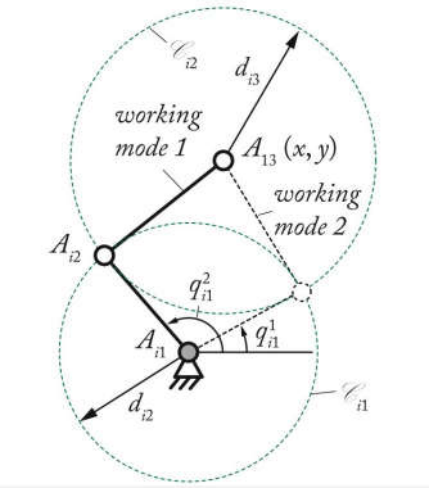

Figura 2.16. Descrição das duas configuração de um robô RR 2links Fig. de [Briot et al., 2015]

Os valores de q_{i2} podem ser obtidos utilizando a equação 2.55 de forma:

$$q_{i2} = arcotan(y - d_{i2} \cdot Sq_{i1}, x - d_{i1} - d_{i2} \cdot Cq_{i1}) - q_{i1} \tag{2.55}$$

Para: $i = \{1, 2\}$
E para finalizar:

$$q_{13} = q_{21} + q_{22} - q_{11} - q_{12} \tag{2.56}$$

Solução de Cinemáticas diretas em robôs paralelos

Como em robôs seriais a cinemática direta de robôs paralelos consiste em encontrar qual a posição dos atuadores para que o robô se mova para uma certa configuração, a diferença se dá na possibilidade de atuação redundante existente em cadeias cinemáticas fechadas.

A definição de [Briot et al., 2015] para o problema é: A resolução da equação $x = \mathfrak{H}^{-1}(q_a) = \mathfrak{G}_{\mathfrak{a}}$ onde x são as coordenadas do efetuador. E a resolução do problema adicional para obter as coordenadas q_d, variáveis de juntas passivas em uma cadeia fechada, $q_d = \mathfrak{G}_d(q_a)$. Para simplificar o problema [Briot et al., 2015] desconecta virtualmente as cadeias do efetuador, sem perder a generalização do ponto A_{1m_1} que é o ponto da junta entre a cadeia e o efetuador. Com efeito a retirada de uma cadeia supre o efetuador com mais um grau de liberdade. Ficando assim livre para realizar movimentos dentro da configuração LOCI $\mathfrak{L}$ onde os outros atuadores estão fixos. Já a junta A_{1m_1} que pertence a cadeia desconectada pode realizar movimentos dentro da configuração LOCI $\mathfrak{C}$

denominada de espaço de vértice da cadeia. A intercepção das configurações LOCI $\mathfrak{L}e\mathfrak{C}$ é a solução da cinemática direta de um robô paralelo. Como existem redundâncias mais de uma solução pode ser encontrada, cada uma dessas é chamada de modos de configuração do robô. Cada uma representa uma configuração específica o qual o robô pode ser montado de acordo com suas juntas ativas.

Aplicando o método MDH para resolução de cinemáticas diretas em um robô planar de dois graus de liberdade: Considerando a equação de cadeia encontrada por [Briot et al., 2015] na resolução da cinemática inversa:

$$h_p(X, q_a) = \begin{vmatrix} (c_1 - a_1)t_{11}^2 + 2b_1t_{11} + (a_1 + c_1) \\ ((c_2 - a_2)t_{21}^2 + 2b_2t_{21} + (a_2 + c_2) \end{vmatrix} = 0 \tag{2.57}$$

Realizando uma substituição de variáveis tem-se que:

$$h_p(X, q_a) = \begin{vmatrix} x^2 + y^2 + a_1x + b_1y + c_1 \\ x^2 + y^2 + a_2x + b_2y + c_2 \end{vmatrix} = 0 \tag{2.58}$$

Onde: $a_i = -2(d_{i1}+d_{i2}\cdot Cq_{i1})$ $b_i = -2(d_{i2}\cdot Sq_{[}i1])$ $c_i = (d_{i1+d_{i2}\cdot Cq_{i1}})+(d_{i2}^2\cdot S^2q_{i1})-d_{i3}^2$

Do ponto de vista geométrico aparece novamente a intercepção de dois círculos, porém agora os centros destes se encontram nas juntas intermediárias passivas $A_{12}eA_{22}$ e não no efetuador e na base. Como pode-se observar na Fig.2.17.

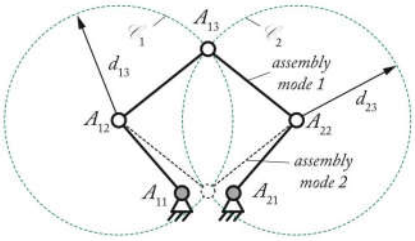

Figura 2.17. Descrição das cinemática direta RR cinco barras Fig. de [Briot et al., 2015]

Com efeito a resolução das coordenadas se dá por: se b_1 e b_2 são diferentes:

$$x = \frac{-f_2 \pm \sqrt{f_2^2 - 4f - 1 \cdot f_3}}{2 \cdot f_1} \tag{2.59}$$

$$y = e_1 \cdot x + e_2 \tag{2.60}$$

onde:

$f_1 = 1 + e_1^2$

$f_2 = 2e_1e_2 + a_1 + b1e_1$
$f_3 = e_2^2 + b_1e_2 + c_1$
$e_1 = -d_1/d_2$
$e_2 = -d_3/d_2$
$d_1 = a_2 - a_1$
$d_2 = b_2 - b_1$
$d_3 = c_2 - c_1$
ou se b_1 e b_2 são iguais:

$$x = \frac{c_2 - c_1}{a_2 - a_1} \tag{2.61}$$

$$y = \frac{-b_1 \pm \sqrt{b_1^2 - 4(x^2 + a_1x + c - 1}}{2} \tag{2.62}$$

Jacobianos de velocidade e aceleração, singularidades e similaridades

As relações de velocidade entre as juntas ativas e a torque do efetuador podem ser encontradas derivando as equações de cadeia. Porém por [Briot et al., 2015], isso seria computacionalmente custoso, e propõe uma metodologia alternativa. Logo, considerando a transformação ${}^0T_p$ e as velocidades envolvidas ($\dot{q_i}$) e aplicando os vetores jacobianos observa-se que:

$${}^0T_p = {}^0J_{pi} \cdot \dot{q_i} = [\,{}^0\$_{i1}, ..., {}^0\$_{im_i}] \cdot \dot{q_i} \tag{2.63}$$

Onde ${}^0\$_{ik}$ representa o deslocamento do efetuador quando a junta ik está movendo e mi é o número de juntas na cadeia. Separando estas variáveis em dois grupos, para juntas ativas: ${}^0\$_{ia}$ e velocidades $\dot{q_i a}$; E ${}^0\$_{id}$ e velocidades $\dot{q_i d}$ para juntas passivas;

Aplicando a metodologia em um robô planar de dois graus de liberdade:

$${}^0T_p = \begin{vmatrix} {}^0\$_{11} & {}^0\$_{12} & {}^0\$_{13} \end{vmatrix} \cdot \dot{q_1} \tag{2.64}$$

$${}^0T_p = \begin{vmatrix} {}^0\$_{21} & {}^0\$_{22} \end{vmatrix} \cdot \dot{q_2} \tag{2.65}$$

onde:

$${}^0\$_{11} = \begin{vmatrix} -d_{12} \cdot Sq_{11} - d_{13} \cdot S_{121} & d_{12} \cdot Cq_{11} - d_{13} \cdot C_{121} & 0 & 0 & 0 & 1 \end{vmatrix}^T \tag{2.66}$$

$${}^0\$_{12} = \begin{vmatrix} -d_{13} \cdot S_{121} & d_{13} \cdot C_{121} & 0 & 0 & 0 & 1 \end{vmatrix}^T \tag{2.67}$$

$$^{0}\$_{13} = \left|0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1\right|^{T} \tag{2.68}$$

$$^{0}\$_{21} = \left|-d_{22} \cdot Sq_{21} - d_{23} \cdot S_{122} \quad d_{22} \cdot Cq_{21} - d_{23} \cdot C_{122} \quad 0 \quad 0 \quad 0 \quad 1\right|^{T} \tag{2.69}$$

$$^{0}\$_{22} = \left|-d_{23} \cdot S_{122} \quad d_{23} \cdot C_{122} \quad 0 \quad 0 \quad 0 \quad 1\right|^{T} \tag{2.70}$$

$$^{0}T_{p} = \left|\dot{x} \quad \dot{y} \quad 0 \quad 0 \quad 0 \quad \dot{\phi}\right|^{T} \tag{2.71}$$

$$\dot{q_1} = \left|\dot{q_{11}} \quad \dot{q_{12}} \quad \dot{q_{13}}\right|^{T} \tag{2.72}$$

$$\dot{q_2} = \left|\dot{q_{21}} \quad \dot{q_{22}}\right|^{T} \tag{2.73}$$

$$C_{12i} = C(q_{i1} + q_{i2}) \tag{2.74}$$

$$S_{12i} = S(q_{i1} + q_{i2}) \tag{2.75}$$

Para a mão de força sobre o efetuador:

Logo:

$$\zeta_{i}^{T} \cdot {}^{0}\$_{id} = 0 \tag{2.76}$$

$$\zeta_{i}^{T} \cdot {}^{0}\$_{ia} \neq 0 \tag{2.77}$$

$$^{0}\$_{a1} = {}^{0}\$_{11} \tag{2.78}$$

$$^{0}\$_{a2} = {}^{0}\$_{21} \tag{2.79}$$

$$^{0}\$_{d1} = \left|{}^{0}\$_{12} \quad {}^{0}\$_{13}\right| \tag{2.80}$$

$$^{0}\$_{d2} = {}^{0}\$_{22} \tag{2.81}$$

Então:

$$\zeta_1 = \begin{vmatrix} C_{121} & S_{121} & 0 & 0 & 0 & 0 \end{vmatrix}^T \tag{2.82}$$

$$\zeta_2 = \begin{vmatrix} C_{122} & S_{122} & 0 & 0 & 0 & 0 \end{vmatrix}^T \tag{2.83}$$

Nota-se que ζ_1 é uma força pura aplicada diretamente a $A_{12}\vec{A}_{13}$ e ζ_2 é uma força pura aplicada diretamente a $A_{22}\vec{A}_{13}$.

$$A = \begin{vmatrix} \zeta_1^T \\ \zeta_2^T \end{vmatrix} = \begin{vmatrix} C_{121} & S_{121} & 0 & 0 & 0 & 0 \\ C_{121} & S_{121} & 0 & 0 & 0 & 0 \end{vmatrix} \tag{2.84}$$

$$B = \begin{vmatrix} \zeta_1^T \cdot {}^0\$_{a1} & 0 \\ 0 & \zeta_2^T \cdot {}^0\$_{a2} \end{vmatrix} \tag{2.85}$$

Como resultado o jacobiano de primeira ordem da cinemática inversa se dá por:

$${}^0t_r = -A_r^{-1} \cdot B \cdot \dot{q_a} \tag{2.86}$$

O jacobiano de primeira ordem da cinemática direta se dá por:

$$\dot{q_a} = -B^{-1} \cdot A_r \cdot {}^0t_r \tag{2.87}$$

onde:

$$A_r = \begin{vmatrix} C_{121} & S_{121} \\ C_{121} & S_{122} \end{vmatrix} \tag{2.88}$$

Estudo das singularidades: As singularidades encontradas com as matrizes A e B são classificadas por [Briot et al., 2015] em três tipos:

- Tipo 1: Quando a matriz B têm uma deficiência de posto, ou seja, alguma de suas linhas não é linearmente independente. O significado físico disso é o robô chegou a algum limite de movimentação de alguma direção do espaço de trabalho.

- Tipo 2: Quando a matriz A_r têm uma deficiência de posto. Neste tipo de singularidade o robô atinge um ponto do espaço de trabalho onde seu movimento é imprevisível, $\dot{q_a} = 0$, ou seja, os atuadores ficam travados e o robô não tem rigidez a esforços externos. O sistema linear neste ponto tem infinitas soluções. Outra informação importante é que nas imediações destes pontos o torque dos atuadores sobe consideravelmente.

- Tipo 3: Quando as duas matrizes têm deficiência de posto. Neste tipo o robô está restringido de movimento em alguma direção e com movimento imprevisível em outro.

Singularidades do robô planar de dois graus de liberdade: Quando $q_{12} = 0$ ou π ou seja quando a cadeia está totalmente esticada ou dobrada a matriz B torna-se deficiente de posto e o robô chegou a um limite de movimento. Como ilustrado na Fig.2.18.

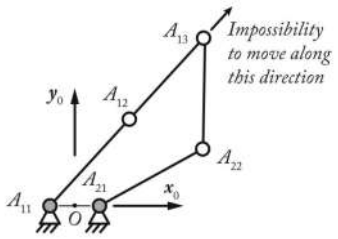

Figura 2.18. Descrição da singularidade robô RR 2links de [Briot et al., 2015]

Quando $q_{11} + q_{12} - q_{21} - q_{22} = 0$ ou π a matriz A torna-se deficiente de posto, ou seja quando os pontos A_{12}, A_{13}, A_{22} estão alinhados. Como indicado na Fig.2.19.

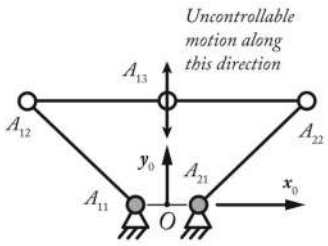

Figura 2.19. Descrição da singularidade para robô RR 2links de [Briot et al., 2015]

2.2 Arquitetura de planejamento e comando de trajetória e movimento

Os sistemas robóticos no geral são compostos pelos seguintes blocos, segundo [JF et al., 2009]:

- Gerador de Trajetória,*on-line ou off-line*;
- Gerador de Movimento;
- Servo-atuadores com controle de movimento;
- Sensores internos e externos;
- Braços e links mecânicos;
- Ferramenta de trabalho;
- Sistemas de limite e segurança do operador e do robô.

Esta arquitetura está intimamente ligada a acurácia e a repetibilidade de um sistema robótico. Isto através da escolha de características destes blocos, como:

- O tipo, posição, quantidade e resolução de sensores;
- Tipo das reduções mecânicas, e suas folgas;
- Rigidez dos links e a geometria do robô;
- Modelos computacionais das solução das cinemáticas;

A qualidade de movimento de um robô será a soma das folgas e falhas de desenvolvimento de cada bloco de seu sistema. Não obstante um bloco pode compensar características de outros blocos. Por exemplo, um robô com necessidade se ser extremamente leve, pode abrir mão da rigidez de seus links e ter um desenvolvimento computacional que modele e corrija essa falta de rigidez. O ônus desta escolha seria um custo computacional e operacional mais elevado. Em contrapartida a maioria dos robôs industriais preza por estruturas muito robustas e rígidas, junto a servo-atuadores de elevada precisão para que possam ter uma acurácia e repetitividade muito alta sem elevados custos computacionais e operacionais.

2.2.1 Geradores e Planejadores de Trajetória

O planejamento ou geração de trajetória é definido por [Spong et al., 2006] como: O processo que gera o caminho a ser seguido pelo robô entre dois pontos de interesse. Isto levando em consideração obstáculos, limites dos espaços de trabalho e junta e limitações dinâmicas do robô. Assim a trajetória deve conter a descrição geométrica do movimento desejado, porém sem a descrição completa da dinâmica do mesmo, somente um parâmetro temporal de escala deverá ser incluso.

A classificação dos tipos os geração de trajetória feita por [Kumar et al., 1999] define-os em:

- **Explícitos:** Geram uma sequência de movimento que é previamente calculada. A maioria dos robôs fixos industriais utilizam deste modo de geração de trajetória.
- **Implícitos:** O próximo movimento é calculado baseado na pose atual e em informações sensoriais e leis gerais de controle previamente definidas. Por exemplo, robôs móveis que observam o ambiente para planejar seu próximo movimento.

E o tipo de trajetória gerada em:

- **Contínuas:** Uma função contínua é utilizada como referência como entrada de controle em malha aberta. Esta é normalmente aplicada em atuadores muito simples que normalmente executam movimentos repetitivos e com baixa necessidade de flexibilização.
- **Discretas:** Sequências de posições discretas, a serem utilizadas dentro do espaço de configurações entre um ponto de interesse e outro. Dentro das trajetórias discretas, estas ainda podem ser divididas em dois tipos:
  - **Ponto a Ponto:** Sequência de movimento que utiliza somente o ponto inicial e o final, corroborado por [Craig, 2009]. Não leva em consideração o espaço de configurações, ou seja, tenta realizar o menor caminho possível, ou o de menor tempo, entre os dois pontos sem observar possíveis obstáculos. Geralmente quando a programação de um robô é feita *on-line* sua trajetória é ponto a ponto.
  - **Caminho contínuo:** Sequência de movimento com vários pontos intermediários entre os pontos iniciais e finais, onde esses são contidos no espaço de configurações. De modo geral quando a programação de robôs é realizada *off-line* utilizando de softwares CAM (*Computer Aided Manufacturing*) sua trajetória é de caminho contínuo dentro do espaço de configurações.

Geração de trajetória explícita discreta de caminho contínuo: Esta é uma trajetória baseada em vários pontos intermediários entre a posição inicial e final e respeitando o espaço de configuração.

Estes pontos ainda podem passar por algoritmos de otimização e suavização de curvas. Logo, transformando retas entre pontos intermediários em um número maior de pequenos segmentos de reta.

Constituindo assim um caminho contínuo, porém ainda sujeito a discretização da taxa de atualização de caminho, e mesmo que seja muito pequena deve ser levada em consideração. Os pontos intermediários segundo [Kumar et al., 1999] podem ser definidos por algoritmos de planejamento de movimento que utilizam o espaço de configurações e técnicas, como:

- Campos potenciais artificiais como descrita por [Spong et al., 2006];
- Mapas de rotas, funções específicas que conectam duas configurações adjacentes, como um segmento de reta ou um arco;
- Decomposição celular e mapas probabilísticos, gerando grafos entre configurações adjacentes não sobrepostas e suas relações;

Contudo, caso na aplicação robótica existam obstáculos internos ao espaço de trabalho, ou o robô aplicado tenha múltiplas soluções de cinemática para alguma configuração, passa ser fundamental o planejamento de movimento junto a geração de trajetória, como explicado por [Kumar et al., 1999]. Por exemplo, robôs paralelos ou robôs não holonômicos.

Este trabalho é baseado geradores de trajetória discretos e de caminho contínuo do tipo mapa de rotas. Onde as configurações são ligadas por segmentos de reta de comprimento definido pelo parâmetro de escala temporal. Ou seja, a taxa de atualização de caminho é fixa e a velocidade do movimento é proporcional ao tamanho do segmento. Nesta configuração, quando ocorrem inflexões de velocidade e aceleração no movimento, força-se o equipamento e o sistema de controle ao máximo. Isto causa a deterioração da performance em cantos afiados no plano cartesiano como citado por [Munasinghe & Nakamura, 2006]. Para se evitar essas aberrações de movimento, são utilizadas metodologias para se tentar suavizar esses cantos. Porém, isto implica em uma programação complicada, extensa e dificilmente aplicada em programação online. E caso o movimento comandado explorar algum limite dinâmico do robô, uma sobrecorreção do sistema de controle apareceria. Assim ao passar por inflexões, possíveis oscilações e vibrações seriam causadas no sistema.

Na maioria das aplicações industriais, utiliza-se um compensador do tipo avançado (*feed-foward*), que antevem a inflexão aplicando uma diretiva de controle que altera o movimento de forma ao sistema de controle conseguir atuar de forma crítica, mesmo com a presença de atrasos dinâmicos consideráveis como citado por [Munasinghe & Nakamura, 2006].

2.2.2 Gerador de Movimento

As trajetórias são armazenadas em forma de uma lista de comandos a serem interpretados pelo gerador de movimento. A definição de [Spur & Uhlmann, 2005], é que este módulo tem a responsabilidade de:

- Realizar a interface com o operador;
- Interpretar os comandos;
- Encadear a sequência de movimentos de uma tarefa;
- Aplicar as descrições e limitações dinâmicas do robô à trajetória gerada;

O gerador de movimento também realiza as alterações *on-line* desta trajetória, ou *on-the-fly*, utilizando de sistemas adicionais de sensoriamento e regras de trajetória implícitas. Por exemplo sistemas de visão ou medição espacial a laser, podem ser utilizados para fechar o *loop* de pose da ferramenta diretamente no gerador de movimento, aumentando a precisão e corrigindo desvios difíceis de serem previstos.

Para facilitar a configuração da resposta dinâmica do robô muitos geradores de movimento utilizam curvas predefinidas de aceleração como: Curvas de aceleração linear (trapezoidal), e curvas "S"polinomiais de 3^a e 5^a ordem. A grande diferença entre elas está na suavidade da aceleração no início do movimento, por exemplo a inflexão no início da curva trapezoidal acaba por criar um ponto de impulso infinito no início e no final do movimento causando possíveis trancos; já uma curva S de 5^a ordem já não causa impulsos elevados, como demonstram [Spong et al., 2006] e [Spur & Uhlmann, 2005].

A utilização de curvas pré-definidas parece ser de certa forma menos eficiente do que uma curva baseada nas características exatas dos atuadores do robô. porém, essa dinâmica muda de acordo com o passar do tempo e utilização do equipamento. Outro ponto são as variações da fabricação de cada robô, e por último essa solução genérica possibilita a utilização do mesmo controle de movimento para robôs que tenham dinâmica similares, mas não exatamente iguais.

A curva de velocidade Trapezoidal consiste em três fases, a inicial com aceleração constante, a intermediária com velocidade constante, e a final com desaceleração constante. Apesar da curva trapezoidal ter a desvantagem de criar pontos de alto impulso,

sua simplicidade e o baixo custo computacional acaba por fazê-la muito comum, principalmente em sistemas de custo efetivo baixo.

Segundo [Craig, 2009] os valores de posição, velocidade e aceleração são recalculados de forma discreta na periodicidade da taxa de atualização de movimento (variando entre 60 e 2000Hz). Esse período é chamado de janela temporal de interpolação por [Spur & Uhlmann, 2005]. Logo dentro desta janela estas variáveis dinâmicas estão congeladas, porém o movimento não. Assim, entre uma atualização e outra, o movimento é constante em velocidade e aceleração. Isto causa uma segmentação secundária de resolução mais alta do que a encontrada na geração de trajetória. Como pode-se notar na Fig.2.20 onde [Wüst, 2018] exemplifica três tipos de discretização e onde elas ocorrem dentro do sistema de comando e controle de movimento de um robô. A terceira segmentação é a de controle de posição e movimento pelo driver de controle causada pela resolução discreta dos sensores deste loop.

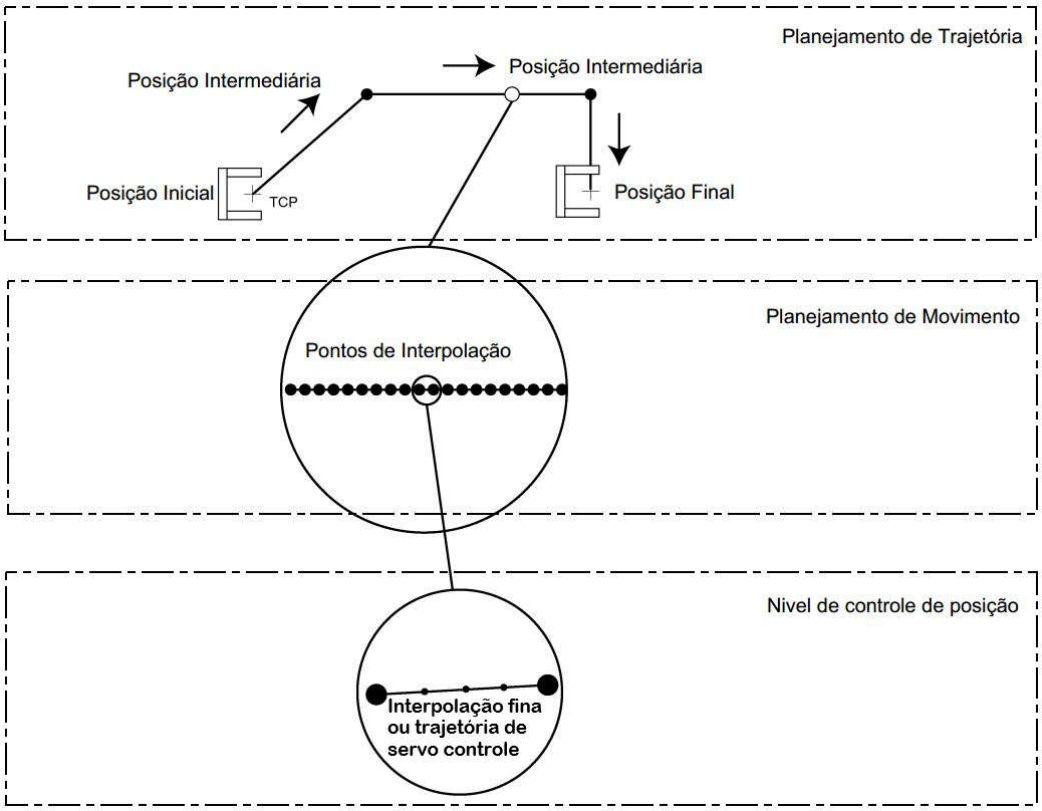

Figura 2.20. Diferentes discretizações do comando e controle de movimento Fig. de [Wüst, 2018]

Apesar do comando de movimento de um robô ser calculado de forma contínua, mesmo que segmentado em uma sequência de semirretas. A implementação dos controladores de movimento das juntas é realizada de forma discreta. Ou seja, as variáveis do sistema de controle são atualizadas de acordo com o um estanciamento temporal dis-

creto provido pelo sistema de processamento. [Lewis et al., 2003]. Como pode-se notar na Fig.2.21.

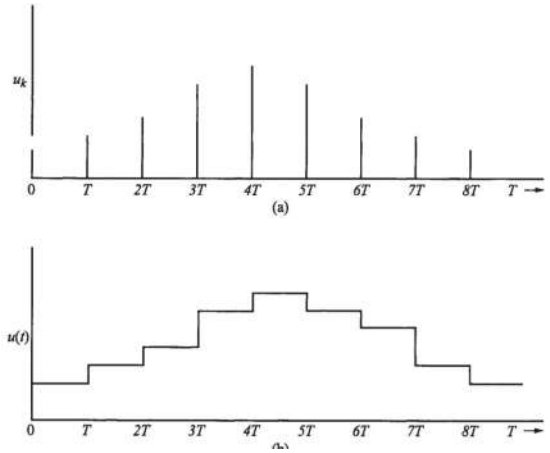

Figura 2.21. Representação do setpoint de movimento discretizado de [Lewis et al., 2003]

A saída do gerador de movimento é o setpoint dos controladores de posição das juntas. Essa passagem de informação pode ser executada de várias maneiras, como:

- Sinais analógicos em controladores muito antigos;
- Sinais de frequência;
- Por redes digitais, como por exemplo as redes tipo CAN.

Porém, controladores que utilizam redes ainda tem seu custo alto para aplicações didáticas ou produção que tenham custo efetivo sensível, assim inviabilizando sua utilização. Para plataformas simples é comum a utilização de sinais de frequência em forma de pulsos para se transmitir o sinal do setpoint de movimento. As duas metodologias mais aplicadas são:

- Pulso e Direção;
- Diferencial ou FASE AB;

Pulso e Direção: É o sistema que tem a vantagem de ser mais simples e com um custo computacional menor, pois, para cada junta é necessário somente um sinal de frequência. Cada pulso deste sinal é um quantum de distância a ser movimento pela junta [Le & Jeon, 2007]. Assim que o número de pulsos representa a posição a ser movimentada na janela temporal. Com efeito a quantidade de pulsos por unidade de tempo, ou frequência, representa a velocidade do movimento. Em consequência a variação da frequência por unidade de tempo representa a aceleração do movimento. Já a orientação do movimento é dada por um sinal digital onde um estado é positivo e outro negativo sendo esse chamado de sinal de direção. [Kienitz, 2012] [Le & Jeon, 2007].

Sistema Diferencial, Fase AB: É o sistema que Utiliza da mesma estrutura de sinal de frequência de pulsos, porém utiliza dois sinais de frequência defasados de 1/2 período [Takahashi & Goetz, 2004]. Isto dobra o custo computacional da geração de pulsos, contudo provem redundância do sinal e controle de erros de ligação entre os blocos do sistema. Sendo então muito indicado para sistemas onde apesar de o custo seja uma variável sensível a necessidade de segurança no movimento também seja sensível. Para informar a direção do movimento este sistema utiliza o desfasamento positivo ou negativo de um dos dois sinais, sendo que eles sempre estão defasados de ou +1/2 ou -1/2 período no tempo.

Este trabalho enfoca o sistema Pulso e Direção pois o hardware para sua utilização é o mais comum no mercado, principalmente devido a maioria das impressoras 3D de hardware aberto utilizarem esta configuração.

Gerador de Pulsos

O módulo gerador de pulsos é geralmente integrado ao gerador de movimento, e é responsável por transformar a quantidade de movimento a ser executada por cada junta, em um intervalo da janela temporal de interpolação, em pulsos de movimento com uma certa resolução espacial definida. Ele realiza uma sobre-discretização do segmento de movimento em pulsos de movimento. Pode-se notar isso na Fig.2.20, na passagem do nível de planejamento de movimento para o nível de controle de posição. Pode-se dizer que o gerador de movimento define o número de pulsos de movimento necessários a serem executados dentro de uma janela temporal por cada junta, e o gerador de pulsos executa estes pulsos.

Dentro de uma janela temporal a frequência, ou seja, a velocidade de movimento, é fixa. Espera-se que a taxa de atualização de movimento seja diretamente proporcional a resolução do movimento. A frequência máxima de geração de pulsos é dezenas ou centenas de vezes maior do que a taxa de atualização de movimento. Isso implica um custo computacional alto e muitas vezes é limitado por pelos recursos disponíveis no próprio hardware

do gerador de movimento. As entradas dos controladores de movimento mais utilizados estão limitadas entre 100 e 300Khz, ([All, 2014], [Tos, 2016], [Instruments, 2014]. Para contornar isso, sistemas sofisticados utilizam dispositivos externos ao processamento de movimento e especializados em geração de pulsos de alta frequência ou até mesmo *FPGAs* programados em hardware para essa função.

Driver de controle de movimento:

O driver de controle de movimento é o responsável pelo controle da malha de controle de movimento dos motores. Assim transformando a informação de pulsos de movimento em sinais de tensão, corrente e fases para os motores.

Segundo [Le & Jeon, 2007], na entrada de sinal de setpoint de um driver de controle de movimento se encontra um contador de pulso que funciona como ZOH (*Zero Order Holder*). Este integra os pulsos de movimento discreto em uma posição discreta a ser atingida pelo motor naquele instante de controle, como também descrito por [Lewis et al., 2003].

Na Fig. 2.22 nota-se que na saída da malha de controle também existe um ZOH. Isto ocorre porque apesar do controle de posição ser discreto a dinâmica física do motor e suas reduções acabam por filtrar esse comportamento, transformando assim o movimento novamente para uma função contínua em seu limite. Ou seja, a posição de um motor é a soma de todos seus movimentos anteriores.

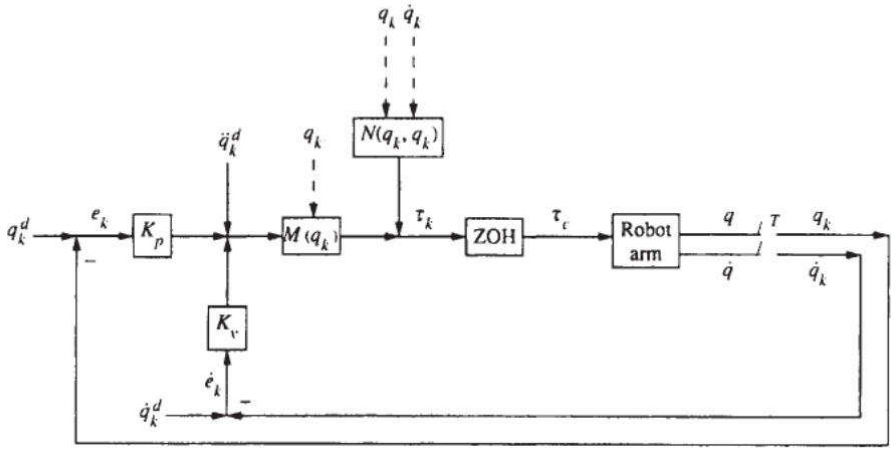

Figura 2.22. Diagrama do controlador de movimento discreto nos drivers de atuação [Lewis et al., 2003]

"A resolução de controle significa o menor incremento de movimento que o controlador pode perceber. Esta resolução pode ser calculada pela distância total percorrida pela junta dividida por $2n$ onde n é o número de pulsos de acurácia do encoder. "[Spong et al., 2006]

2.3 Plataformas computacionais embarcadas

Sistemas computacionais embarcados são aqueles que foram desenvolvidos para serem configurados para uma aplicação específica. Sendo assim muitos destes têm uma capacidade computacional limitada, porém, com grande versatilidade de acesso a seus recursos de hardware. Na maioria das vezes também possuem um vasto conjunto de periféricos específicos, voltados para sistemas de controle, aquisição de dados, e interconexão com outros hardwares com aplicação específica. Como vantagens pode-se citar que esses sistemas permitem grande otimização de seus recursos proporcionando níveis assertivos de confiabilidade a baixo custo, [Arora, 2016].

Para cada módulo do sistema de controle de um robô uma série de características de performance do sistema computacional são requeridas. Uma destas características é o tempo mínimo requerido para execução das ações o qual cada módulo é responsável, como definido por [Dubey, 2008]:

- IHM e Interfaces com indústria 4.0: centenas de milissegundos;
- Gerador de trajetória: dezenas de milissegundos;
- Sensores Analógicos e de Visão: dezenas de milissegundos;
- Gerador de movimento: milissegundos;
- Controle de movimento: centenas de microssegundos;
- Sensores discretos: centenas de microssegundos;
- Controle de torque: dezenas de microssegundos;
- Controle de posição parada: microssegundos;
- Atuação de semicondutores de controle: menos que microssegundo;

2.3.1 SBC, Computadores Embarcados

Geralmente constituídos de sistema computacionais completos em uma placa *(Single Board Computer)*. São compatíveis com sistemas operacionais e conexões usualmente utilizadas em computadores convencionais, porém, com dimensões e consumo energético reduzidos. São muito indicados para quando existe a necessidade de trabalho com grande volume de dados, em relação a outros sistemas embarcados. Assim como a utilização de bancos de dados corelacionais, necessidade de processamento gráfico, ou portabilidade de um sistema já desenvolvido em um PC convencional para uma plataforma convencional. Como principal exemplo pode-se citar o *Raspberry Pi*, que consiste em um projeto

de hardware aberto utilizando processadores do tipo ARM para rodar sistemas operacionais usuais e completos adaptados para essa plataforma como LINUX e EMBEDED WINDOWNS. [Cicolani, 2018].

Apesar do relativo grande poder computacional dos SBC, a utilização de plataformas usuais de sistemas operacionais atrapalha o determinismo do sistema em geral. Ou seja, o sistema nunca atuará em tempo real, sempre haverá atrasos importantes entre um evento no mundo real e a detecção deste evento pelo sistema e sempre existirá uma variação na dimensão deste atraso, o que dificulta sua predição. Com a otimização dos SOs utilizados esses atrasos podem ser minimizados formando assim um sistema quase em tempo real, possibilitando assim a utilização destes sistemas como:

- Interface entre o operador e o sistema (IHM);
- Interface entre o sistema unitário de produção e sistemas especialistas de controle de fábrica (MES);
- Planejador de trajetória;
- Gerador de trajetória *off-line* ou *on-line (on the fly)*;
- Sistemas de Visão;

porém sua utilização como gerador de movimento e interpretador de trajetória torna-se muito difícil pois estas operações exigem um determinismo temporal em níveis acimas dos fornecidos por eles, como citado por [Cicolani, 2018].

2.3.2 SoC, Sistemas Encapsulados

Os sistemas encapsulados SoC (*System on Chip*) são sistemas computacionais dedicados com periféricos embutidos e grande capacidade de resolução de problemas de controle e baixo nível. São similares aos Microcontroladores ou algumas vezes baseados nestes. Tem sua programação gravada em memória física, e costumam ter um SO dedicado onde a aplicação do usuário roda. Estes SOs são do tipo RTOS (*Real Time Operational System*) que é muito diferente de um SO usual, como o utilizado nos SBCs. Estes são desenvolvidos para serem determinísticos, flexibilizam a utilização de recursos do sistema em relação aos uC (microcontroladores), e melhoram a confiabilidade dos sistemas deixando-os mais tolerante a falhas. Os SoC podem ser considerados sistemas híbridos entre os SBCs e os uC, pois: Mantêm o determinismo, baixo consumo e custo comparáveis aos uCs; mas com uma capacidade de memória, processamento e conectividade maior se aproximando dos SBC, como demostrado por [Arora, 2016].

2.4 Síntese do desenvolvimento teórico

O presente trabalho descreve e desenvolve um robô híbrido constituído de: Duas cadeias paralelas fechadas, formando um pantógrafo de quatro barras com sua atuação concentrada na base; E uma cadeia aberta acoplada de forma serial na extremidade da cadeia paralela. Porém, apesar deste trabalho tratar somente de robôs paralelos planares, a metodologia utilizada não tem restrições a ser aplicadas em outros casos.

Para a solução de suas cinemáticas, primeiro é utilizado o método do desacoplamento cinemático, assim separando o problema em duas etapas. Na primeira etapa resolve-se a cinemática da cadeia serial aberta na extremidade, aplicando as diretivas K-K criando um link virtual, possibilitando assim a utilização da solução D-H a cinemática; já na segunda etapa, utiliza-se a solução K-K para um robô paralelo planar de cinco barras aplicada a um robô pantógrafo de quatro barras, demonstrando que este é um caso especial desta solução. Como prova desta solução utiliza-se a solução geométrica analítica de intersecção de círculos para chegar aos mesmos resultados.

No que tange ao sistema de comando de movimento, foi utilizado um planejador de trajetória do tipo discreto de caminho contínuo por mapa de rotas. Este é acoplado a um planejador de movimento que utiliza curvas de aceleração trapezoidais como lei dinâmica de conexão entre as configurações adjacentes na trajetória. Realiza-se assim a segmentação do movimento na frequência de interpolação de movimento em segmentos de retas. Este movimento segmentado é então separado em suas componentes em cada eixo cartesiano, gerando o planejamento de movimento em cada direção e orientação. O setpoint de movimento utilizado é do tipo digital por frequência na configuração pulso e direção. Este é gerado por um microcontrolador dedicado responsável por discretizar o segmento de movimento planejado para cada eixo no nível de controle de posição a ser interpretado pelos drivers de controle dos atuadores.

O sistema de tradução de cinemática atua utilizando o sinal a nível de controle de posição para reconstituir o segmento de movimento planejado interpolado nos eixos cartesianos. Então, utilizando das soluções cinemáticas específicas do robô estudado, é gerado, com uma frequência de interpolação superior à original, um novo planejamento de movimento transformado para o plano de juntas e atuadores. Por fim o set-point de movimento angular é gerado também com a estrutura de frequência em pulso e direção para que o sistema de controle de posição possa atuar de forma condizente com a geometria do robô.

3 Metodologia

O presente capítulo pretende descrever as características do sistema desenvolvido e os procedimentos utilizados no estudo e implementação deste desenvolvimento. Também são descritas neste capítulo os métodos de simulação e experimentação utilizados para estudar e avaliar o comportamento do tradutor de movimento simulado e implementado. Este deverá possibilitar que um gerador de trajetória e movimento do tipo comando numérico e geometria cartesiana possa controlar o movimento de um robô que possui uma geometria mais complexas em tempo real. O estudo de método em questão está dividido em quatro pontos principais, citados abaixo e descritos nas próximas sessões:

- Desenvolvimento: Escolha do robô e estudo e solução da sua geometria e cinemática;
- Simulação: Desenvolvimento de uma plataforma de simulação do sistema de controle e a implementação da solução desenvolvida;
- Experimentação: Aplicação da metodologia de DOE (*Design of Experiment*, Design de Experimento) para estudar a influência dos parâmetros na acurácia do sistema simulado;
- Implementação: Teste do sistema de tradução num sistema embarcado real e comprovação de funcionalidade.

3.1 Desenvolvimento

Nesta sessão descreve-se a geometria de robô escolhida e as soluções matemáticas de suas cinemáticas indiretas, diretas e de atuadores a serem utilizadas na tradução. É fundamental que o tipo de robô escolhido e sua geometria sejam bem conhecidos, para que os resultados possam ser analisados mais facilmente.

3.1.1 Seleção da Geometria

Segundo a classificação proposta por [Spong et al., 2006], os robôs tratados neste estudo são do subgrupo: Robôs alimentados por fonte de energia elétrica, servo controlados com programação de trajetória contínua. Já [Merlet, 2006] descreve sua geometria como robôs paralelos, planares, não redundantes. É importante ressaltar que a metodologia aplicada neste estudo não se limita a esta classe de robôs podendo ser aplicada a outros tipos de manipuladores mais complexos. Porém, a escolha de um tipo de robô onde os conceitos estudados possam ser facilmente visualizados é fundamental para o contexto didático do mesmo. Assim a geometria trabalhada pode-se observada na Fig.3.1.

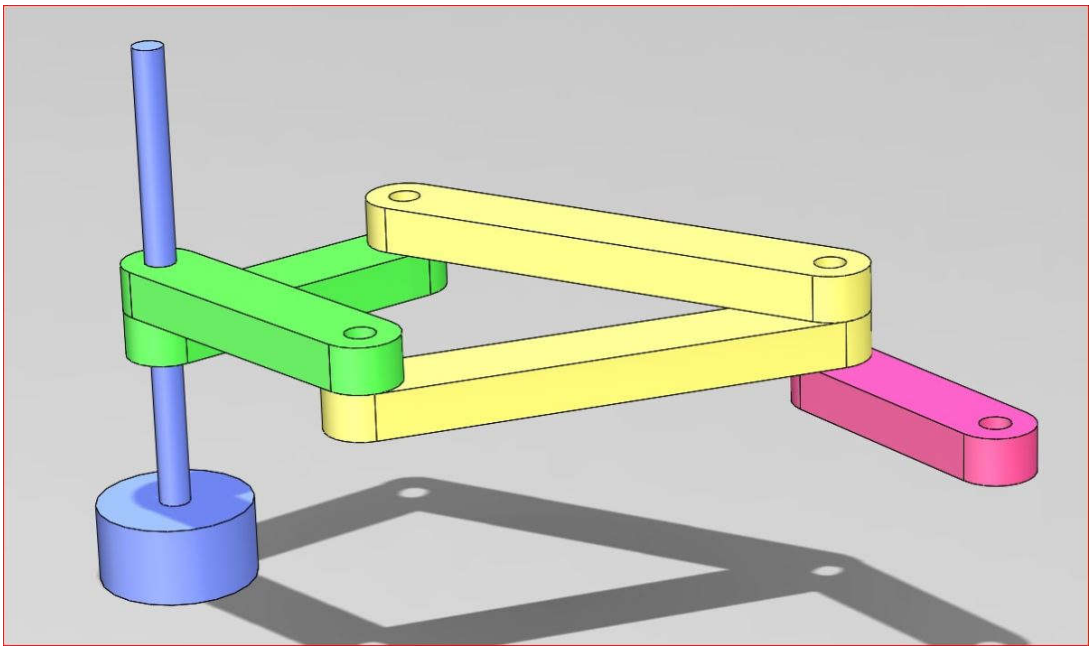

Figura 3.1. Pantografo de quatro barras e atuação concêntrica

Pantógrafo quatro barras de atuação concêntrica: A geometria do robô escolhida é um pantógrafo de quatro barras com atuação concêntrica no ponto de origem. A motivação para a utilização desta geometria está na ampla literatura descrevendo a resolução de cinemáticas semelhantes. Seja de forma analiticamente direta, considerando como uma cinemática paralela; ou por métodos como D-H, a considerando como uma cinemática serial. A segunda opção é possibilitada pela sua simetria em relação aos vetores centrais de sua movimentação. Com efeito a solução do robô planar de dois links e duas juntas rotativas ativas pode ser utilizada. Isto, considerando que um par de links do pantógrafo como a solução positiva (links B1,B2 da Fig.3.2) e o outro par como a solução negativa (links A1,A2 da Fig.3.2). Essa solução será demonstrada posteriormente nesta secção.

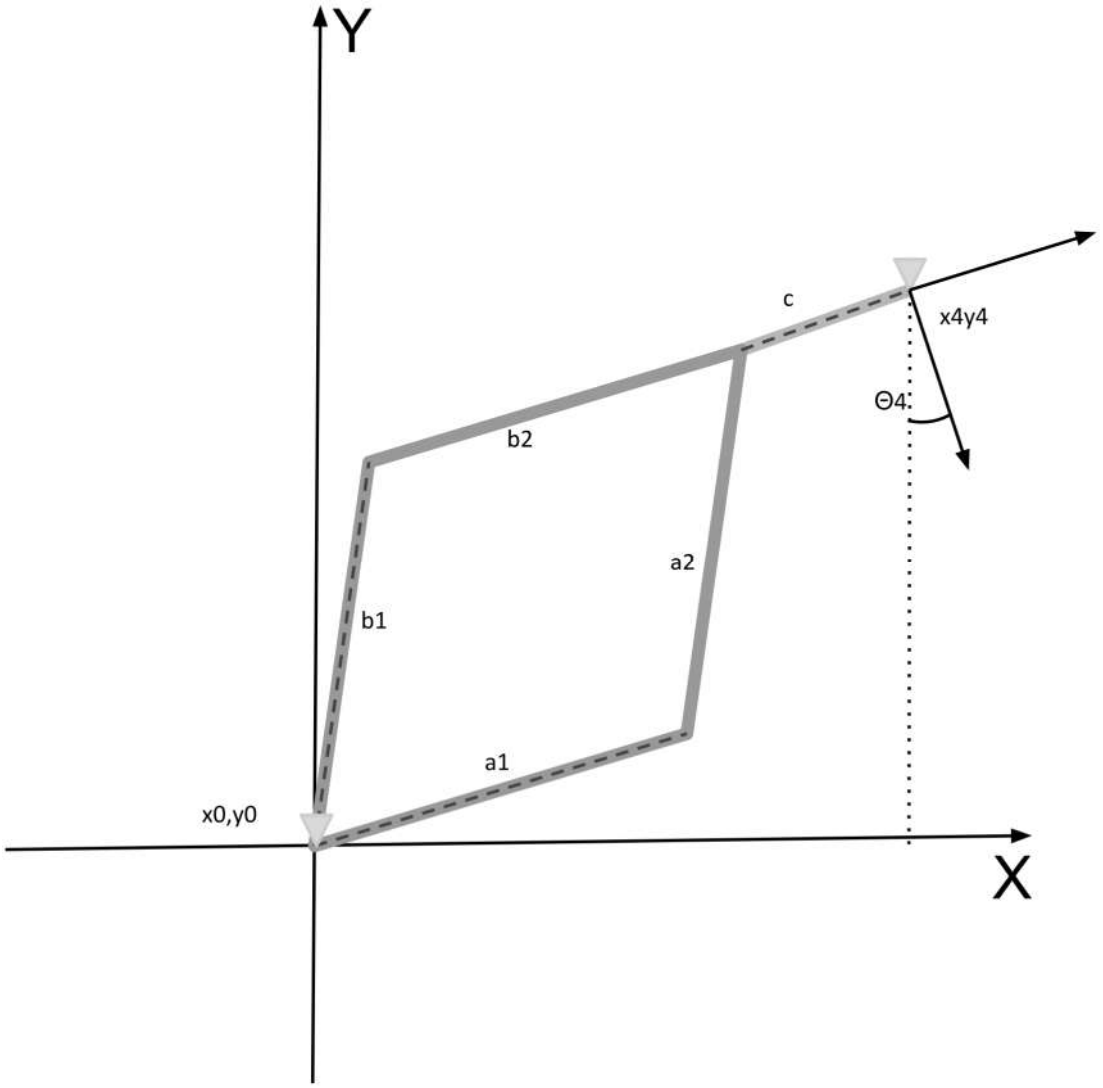

Figura 3.2. Diagrama da geometria do robô escolhido

Pode-se observar vários modelos didáticos baseados em pantógrafos de quatro e cinco barras ou robôs seriais de juntas rotativas com dois e três links. Assim o desenvolvimento do controle de movimento será executado sobre uma plataforma bem conhecida.

A resolução das cinemáticas será dividida em duas partes, posicionamento da ferramenta e orientação da mesma. Devido a caraterística de simetria do robô escolhido, é possível então resolver o link de orientação da ferramenta, eixo "C", como se o robô fosse um manipulador planar de dois links. Considerando para isso um link equivalente virtual, que pode ser observado na Fig. 3.3 e que é coincidente com o vetor entre a origem e o final do pantógrafo, ponto P3(a1a2, b1b2) da Fig.3.2, ou o vetor OC da descrição de [Merlet, 2006].

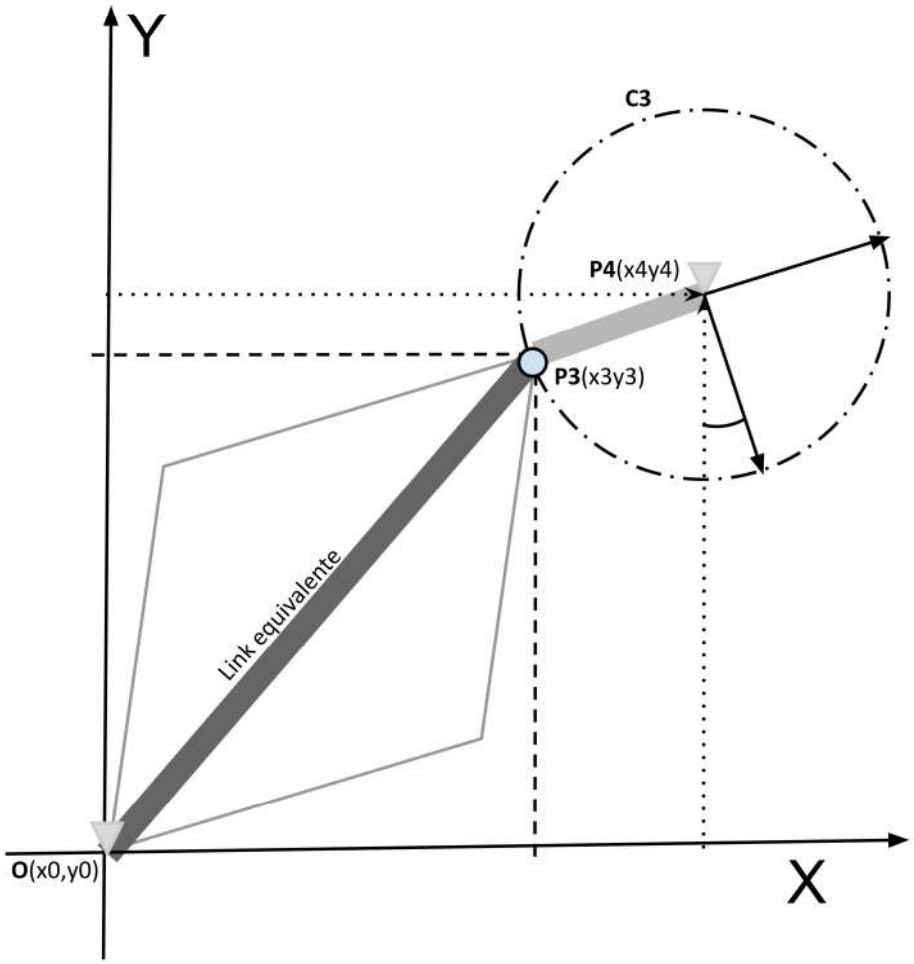

Figura 3.3. Cinemática Inversa Link3

3.1.2 Resolução da cinemática inversa

A solução cinemática inversa é responsável por definir a partir de uma dada posição e orientação da ferramenta, $P4(x_4,y_4)$, as posições dos pontos $P3(X_3,Y_3)$,$P2(X_2,Y_2)$ e $P1(X_1,Y_2)$, como pode-se observar na Fig. 3.4.

Observando a Fig. 3.5 como referência, inicia-se a solução partindo de $P4,(X_4,Y_4)$ e considerando o um link equivalente entre $P0,(Origem)$ e $P3,(X_3,Y_3)$. Pode-se definir a posição de $P3$ de acordo com a posição e orientação de $P4$. Sendo o plano de $P4$ colinear com o link C, então o ângulo entre o eixo X do plano cartesiano e de C será θ_4. Pode-se então calcular $P3$ (x_3,y_3) como sendo um ponto pertencente ao círculo $C4$ com o link C como o raio desse círculo, e a posição de $P3$ é uma transformação direta de senos e cossenos.

$$x_3 = x_4 - c \cdot cos(\theta_4) \tag{3.1}$$

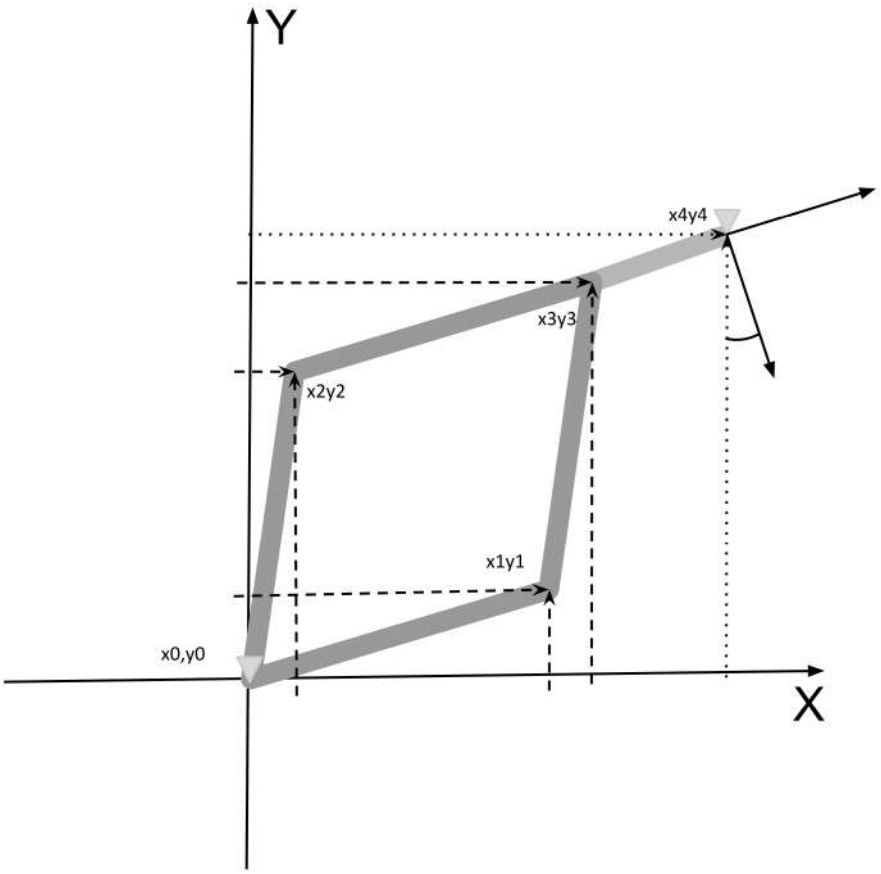

Figura 3.4. Cinemática Inversa Links 1 e 2

$$y_3 = y_4 - c \cdot sen(\theta_4) \tag{3.2}$$

Definindo o ponto de intercessão dos links adjacentes ($b2$ e $a2$) e $P3$, e co-relacionando isto a rotação de $P4$ em no eixo Z do espaço de ferramentas. Assim pode-se ou: Manter um *TCP* (*Tool Center Point*, ponto central da ferramenta) fixo na posição no espaço cartesiano de ferramenta e mudar somente sua orientação; ou alterar somente posição do *TCP* sem alterar sua orientação. Pode-se fazer um paralelo com o método de desacoplamento cinemático demonstrado no item 2.1.2 do capítulo de revisão teórica. Onde, a orientação e a posição do efetuador são resolvidas separadamente e possibilitando assim resolvê-las de forma analítica.

A resolução das cinemáticas do pantógrafo ($P0$,$P1$,$P2$,$P3$ da Fig. 3.5) também será executada de forma analítica. Os pontos de posição das juntas serão calculados através da intercessão entre os círculos. Este tipo de abordagem normalmente nos daria n soluções para uma mesma posição, onde n representa o número de juntas. Porém, ao se utilizar a configuração pantográfica simétrica, as duas soluções encontradas já coincidem com as

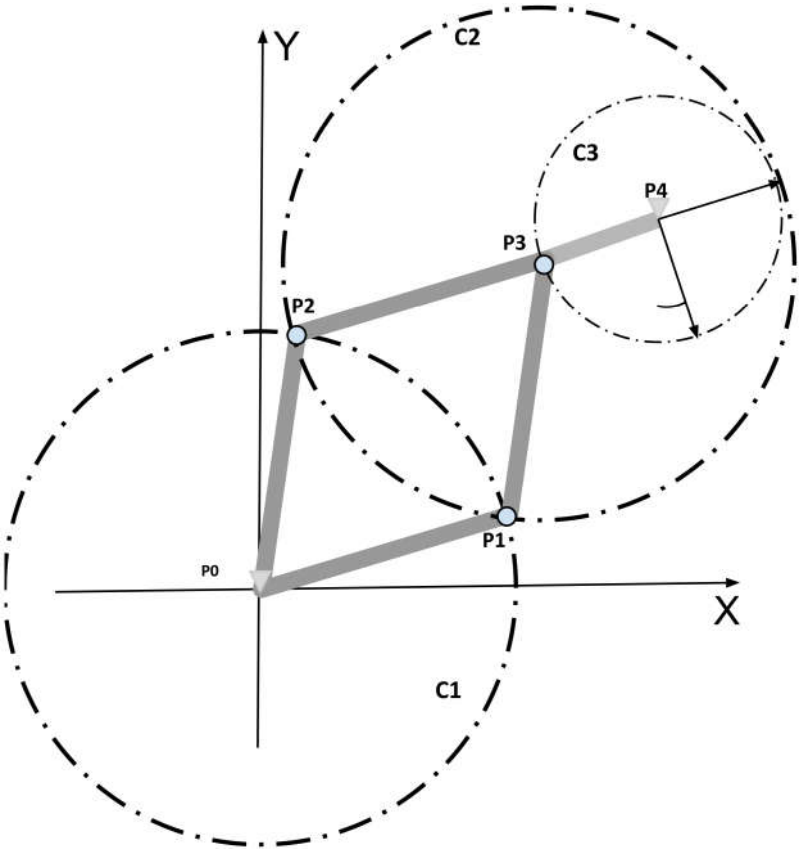

Figura 3.5. Resultado da solução da cinemática Inversa Links 1 e 2

posições das juntas para cada par de links ($a1$,$a2$ e $b1$,$b2$). Assim simplificando o cálculo por não ser necessário a aplicações de diretivas para escolha da melhor solução.

A resolução da cinemática indireta do pantógrafo resume-se a um sistema de equações que representa a interseção de dois círculos. Um com centro em $P3$ e raio igual a $A2$ e $B2$, Fig.3.2 ($C3$, Fig.3.5), e outro com centro na origem e com raio igual a $A1$ e $B1$, Fig.3.2 ($C1$, Fig.3.5).

$$X^2 - 2x_3X + x_3^2 + y2 - 2y_3Y + y_3^2 = \sqrt{a_1} \tag{3.3}$$

$$X^2 - 2x_0X + x_0^2 + y2 - 2y_0Y + y_0^2 = \sqrt{a_2} \tag{3.4}$$

Para resolver este sistema, utiliza-se uma separação de parâmetros e etapas discretas. Estes parâmetros são os índices de cada elemento do sistema acima. Essa técnica previne que o algoritmo tenha uma função matemática compacta indivisível e com um alto custo operacional. Como o alvo deste desenvolvimento é ser implementado de fato em um sistema embarcado, a separação de grandes funções matemáticas em vários passos

intermediários é fundamental para que se consiga gerenciar as interrupções temporais de forma mais eficaz. Assim tem-se a seguir a sequência de solução das etapas intermediárias:

$$p_{11} = -2x_3 \tag{3.5}$$
$$p_{12} = x_3^2 \tag{3.6}$$
$$p_{13} = -2y_3 \tag{3.7}$$
$$p_{14} = y_3^2 \tag{3.8}$$
$$p_{15} = \sqrt{a_1} \tag{3.9}$$
$$p_{21} = -2x_0 \tag{3.10}$$
$$p_{22} = x_0^2 \tag{3.11}$$
$$p_{23} = -2y_0 \tag{3.12}$$
$$p_{24} = y_0^2 \tag{3.13}$$
$$p_{25} = \sqrt{a_2} \tag{3.14}$$

Logo realizando a subtração das equações:

$$p_{31} = p_{21} - p_{11} \tag{3.15}$$
$$p_{32} = p_{22} - p_{12} \tag{3.16}$$
$$p_{33} = p_{23} - p_{13} \tag{3.17}$$
$$p_{34} = p_{24} - p_{14} \tag{3.18}$$
$$p_{35} = p_{25} - p_{15} \tag{3.19}$$

$$p_{31}X + p_{32} + p_{33}y + p_{34} = p_{35} \tag{3.20}$$
$$X = \frac{p_{35} - (p_{32} + p_{34}) - p_{33}Y}{p_31} \tag{3.21}$$
$$p_{41}X = \frac{p_{35} - p_{32}p_{34}}{p_31} \tag{3.22}$$
$$p_{42}X = \frac{-p_{33}Y}{p_{31}} \tag{3.23}$$
$$X = p_{41} + p_{42}Y \tag{3.24}$$

Substituindo na equação:

$$((p_{42}^2)+1)X^2+((2p_{41}p_{42})+(p_{11}p_{42})+p_{13})X+((p_{41}^2)+(p_{11}p_{41})+p_{14}+p_{12}-\sqrt{a_1})=0 \quad (3.25)$$

Resolvendo esta equação do 2º grau tem-se x_1 e x_2 e como $X = p_{41} + p_{42}Y$ calcula-se y_1 e y_2.

3.1.3 Resolução da cinemática direta

Uma vez conhecidos os pontos de posição de cada junta, a cinemática direta deste robô trona-se simples. O cálculo de atuação para posicionar as juntas do pantógrafo em $P1$ e $P2$ (Fig 3.5) se dá pela transformação direta de arco-senos e arco-cossenos dos ângulos dos dois atuadores concêntricos. Estes ângulos podem ser vistos na Fig. 3.6:

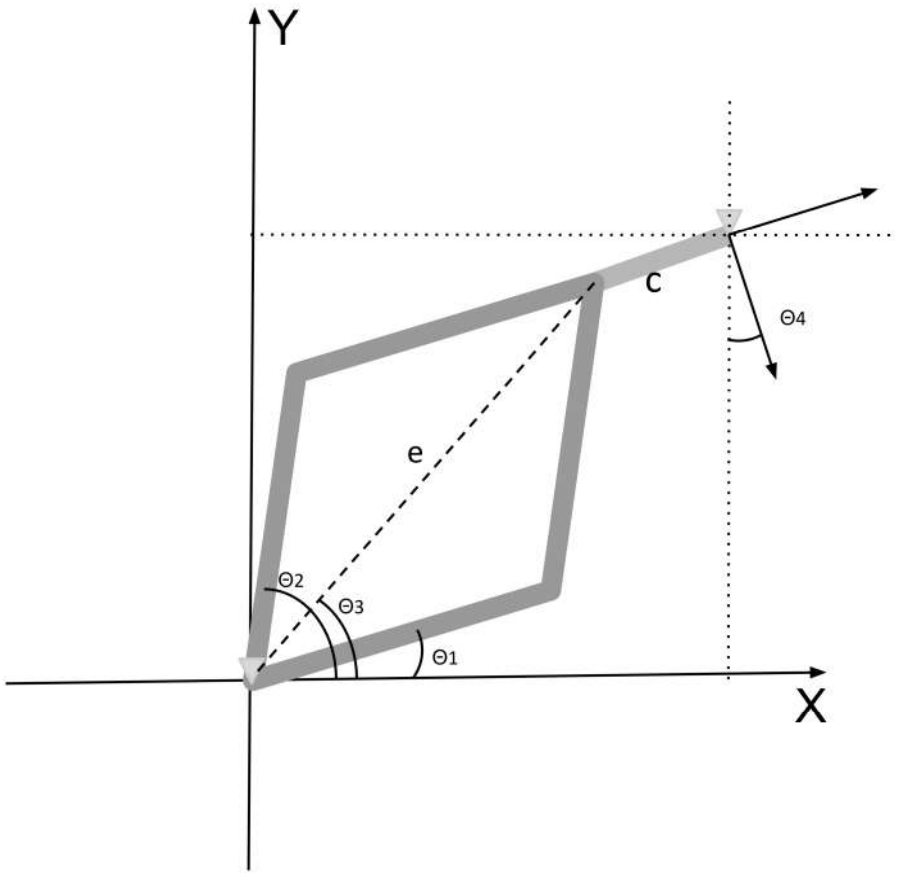

Figura 3.6. Cinemática Direta

Já a cinemática direta do terceiro link pode ser resolvido de duas formas: Analiticamente ou utilizando as transformações homogenias de D-H, como pode ser observado na Tabela3.1.

Link	a_i	α_i	d_i	θ_i
e	e	0	0	θ_3
c	c	0	0	θ_4

Tabela 3.1. Tabela de parâmetros D-H

$$A_1 = \begin{vmatrix} C_1 & -S_1 & 0 & eC_1 \\ S_1 & C_1 & 0 & eS_1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \tag{3.26}$$

$$A_2 = \begin{vmatrix} C_2 & -S_2 & 0 & cC_2 \\ S_2 & C_2 & 0 & cS_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \tag{3.27}$$

$$T_3^0 = A_1 \tag{3.28}$$

$$T_4^0 = A_1 A_2 = \begin{vmatrix} C_{12} & -S_{12} & 0 & eC_1 + cC_{12} \\ S_{12} & C_{12} & 0 & eS_1 + cC_{12} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \tag{3.29}$$

$$x = eC_1 + cC_{12} \tag{3.30}$$

$$y = eS_1 + cS_{12} \tag{3.31}$$

3.2 Simulação

A seguinte sessão descreve a plataforma de simulação utilizada para testar a solução matemática de cinemáticas junto a outros fatores encontrados na arquitetura de controle como, por exemplo, discretização temporal e espacial.

Tomou-se como exemplo a se basear o fluxograma da Fig.3.7 definido por [Lewis et al., 2003]. Porém, etapas adicionais foram acrescentadas para simular as su-

cessivas re-amostragens e integrações em cada etapa do sistema de comando e controle de movimento.

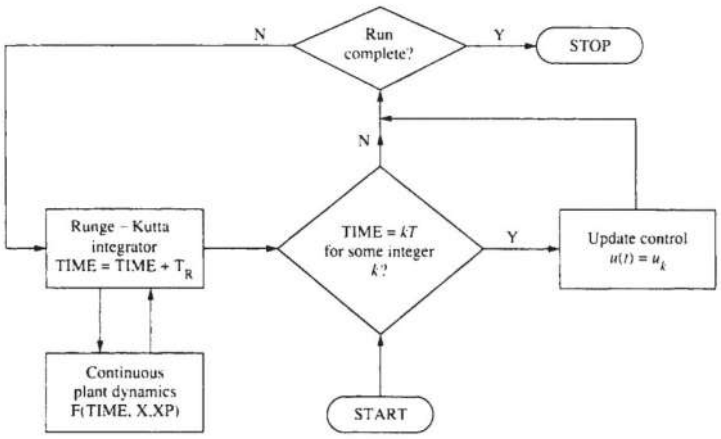

Figura 3.7. Fluxograma de simulação de comando de robô por [Lewis et al., 2003]

O modelo computacional simulado tem como objetivo a prova de conceito da lógica matemática envolvida no processo de captura do setpoint de movimento, tradução e geração do novo setpoint traduzido. Este modelo é uma simplificação para teste das funções e soluções matemáticas, e não contempla os dispositivos utilizados na implementação computacional em hardware.

Para essa etapa são utilizados softwares de computação matemática científica, por exemplo Matlab, Scilab e Octave. Estes são extremamente versáteis na implementação e relativamente simples de se depurar e observar as variáveis. Como escolha para este trabalho optou-se pelo GNU Octave por ser um software *open source* e ter as ferramentas necessárias para simular a matemática envolvida no processo em estudo.

A Fig. 3.8 descreve os módulos do sistema simulado, onde este é divido em três partes principais:

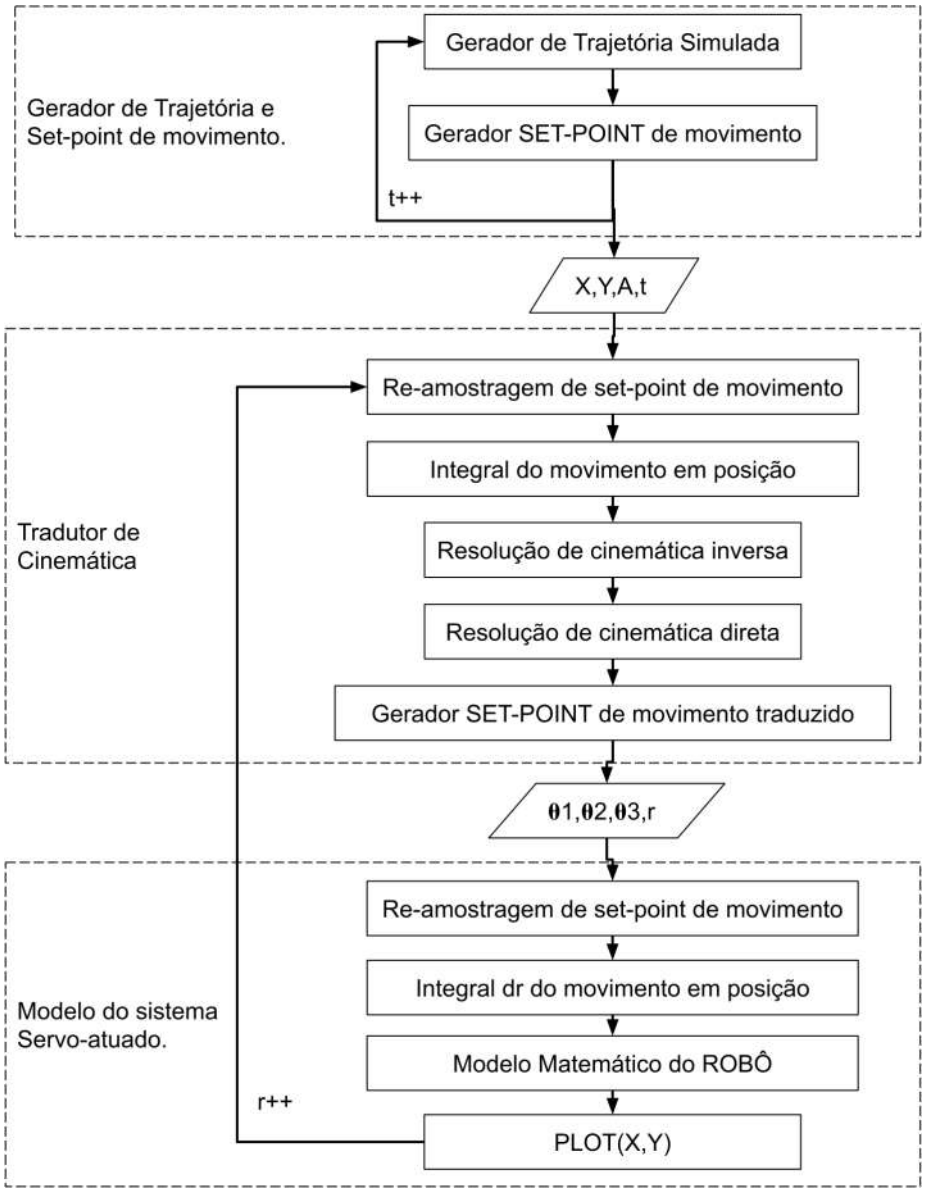

Figura 3.8. Modelo simulado do sistema

- A simulação de trajetória e geração de setpoint de movimento cartesiano: modelo simples de interpolação de três eixos X,Y,A e geração de um vetor contendo a informação em forma de pulsos;
- O Tradutor de cinemática: realiza a re-amostragem do setpoint de movimento e a integral do sinal dentro de uma janela temporal. Reconstitui-se assim a posição inicial e final de um segmento de movimento. Em seguida realiza a tradução

da cinemática inversa e direta e geração de novo vetor de setpoint de movimento angular.

- Simulação do Robô: Transformação do setpoint de movimento angular em posição angular dos atuadores e através da cinemática direta e de atuação. Realizando assim o cálculo da configuração dos componentes do robô e dando como saída a representação gráfica destes componentes no espaço cartesiano.

Pode-se notar que existem duas bases pseudo temporais no sistema, t e r. Isso foi feito, porque, no sistema real, a amostragem temporal do gerador de movimento cartesiano e do tradutor são assíncronas. Realiza-se a passagem de dados entre os módulos através vetores contendo um índex temporal e a informação. Ou seja, informando assim se houve ou não um pulso de movimento naquela instancia temporal indexada. No modulo subsequente essa informação é re-amostrada e integrada transformando-se de um quantum de pulsos (velocidade) para posição.

3.2.1 Modelo simulado de Geração de trajetória

Essa função é responsável por criar segmentos de reta interpolados de x, y e a. Ela encadeia a partir de um ponto inicial, um incremento de posição para cada eixo, até que um número específico de interações temporais seja atingido. Observa-se na Fig.3.9 os parâmetros de entrada: número de pulsos, frequência e ponto inicial, e a informação de saída do mesmo: vetores de posição X,Y, e tempo e. No gráfico 3.10 demonstra-se um exemplo de trajetória gerada. O diagrama 3.9 descreve a implementação do modelo:

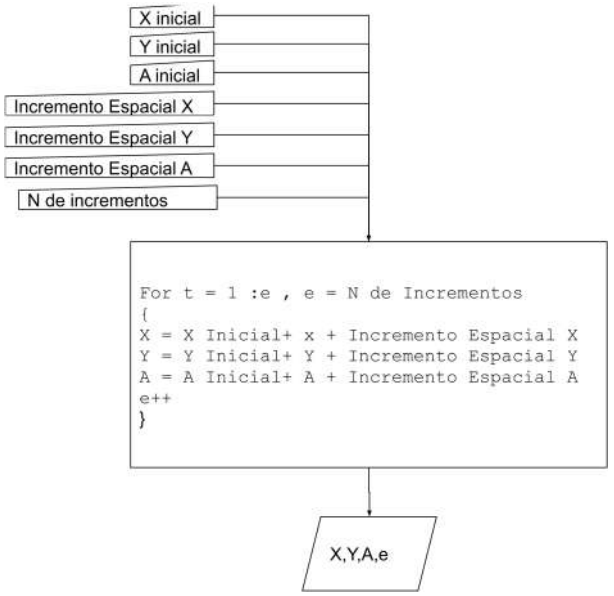

Figura 3.9. Diagrama do gerador de trajetória simulado

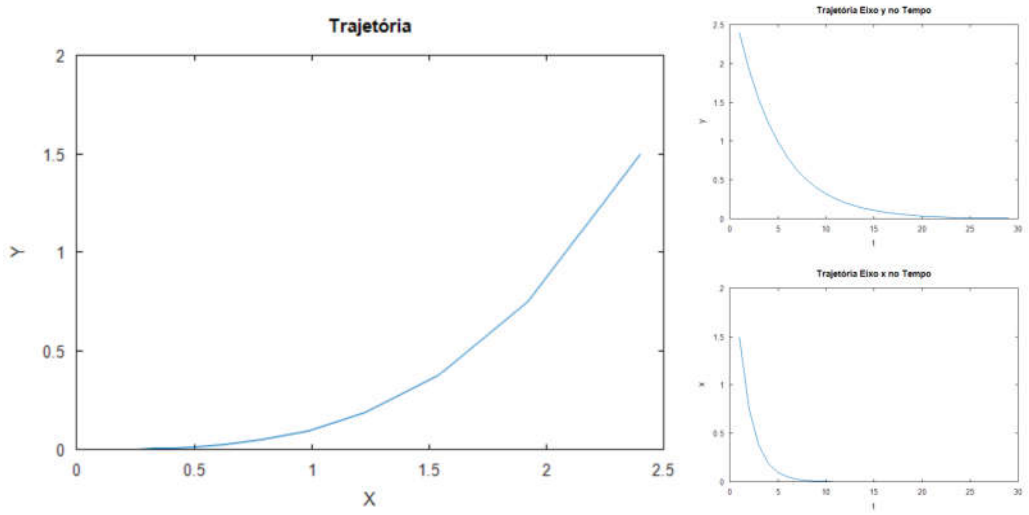

Figura 3.10. Exemplo de Trajetória Gerada

3.2.2 Geração de setpoint de movimento

Essa função tem com objetivo ser prova de conceito do sistema de geração de pulsos controlados, seus parâmetros de entrada são:

- Frequência base: Resolução máxima (metade da frequência máxima) dos pulsos a serem gerados, essa resolução temporal será igual a $Fatiatemporal/FrequnciaBase$ sendo esse valor a miníma largura de um pulso possível.
- Resolução espacial: O número de pulsos a serem dados por unidade de posição;
- Quantidade de movimento a ser executada por eixo e o índex temporal (X,Y,A,t);

O diagrama 3.11 descreve a implementação do modelo:

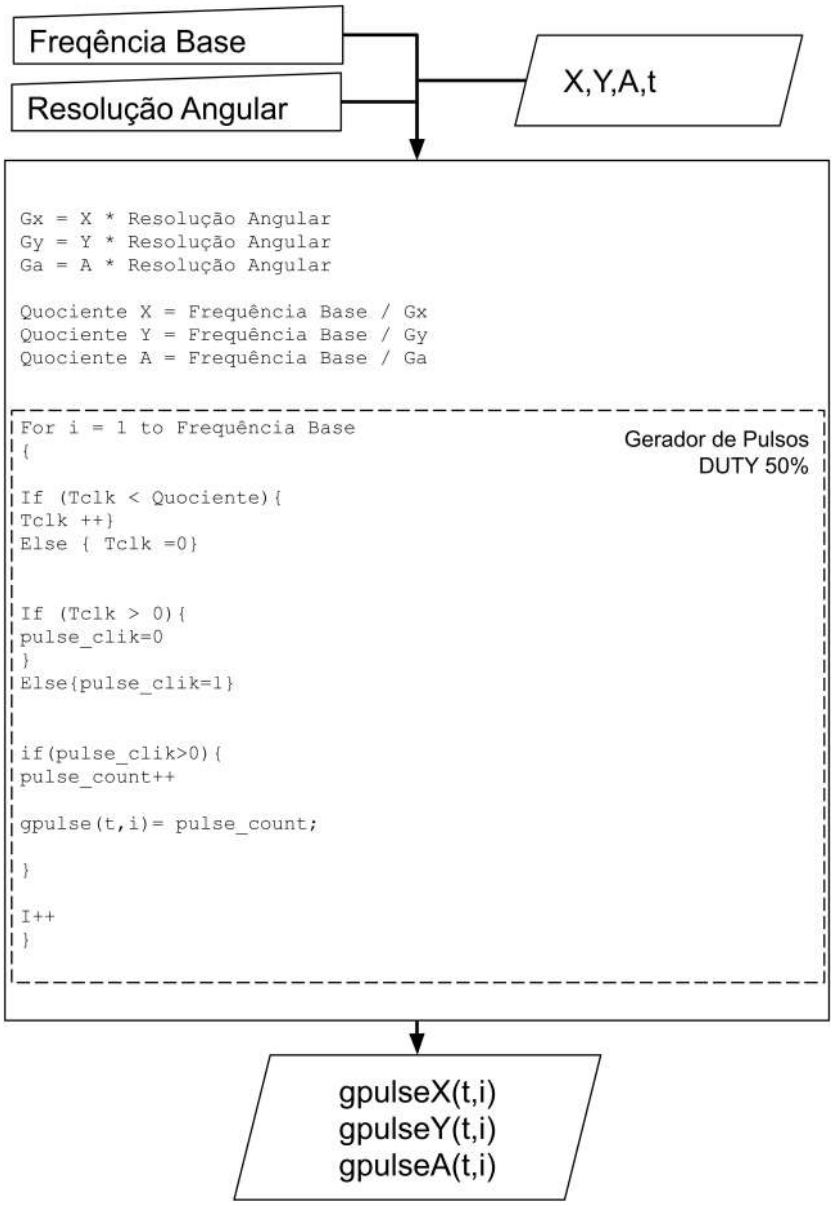

Figura 3.11. Gerador de setpoint de movimento

A saída deste módulo é um vetor contendo, valores nulos onde não existem pulsos, e onde existem pulsos os espaços do vetor são preenchidos com os valores da contagem do número de pulsos realizados na janela temporal em questão. Cada pulso equivale a um incremento da resolução espacial e o número total de pulsos é o diferencial (Δ) espacial a ser movimentado na janela em questão. A frequência base representa a resolução temporal destes pulsos e quanto maior esta, menor a sobra de divisão. Com efeito, menor é o erro de conversão do valor escalar do movimento para o vetor de pulsos. Pode-se comparar as semelhanças das Fig.s 3.12 com a Fig. 2.21 da revisão teórica, para confirmar que os resultados das técnicas são semelhantes.

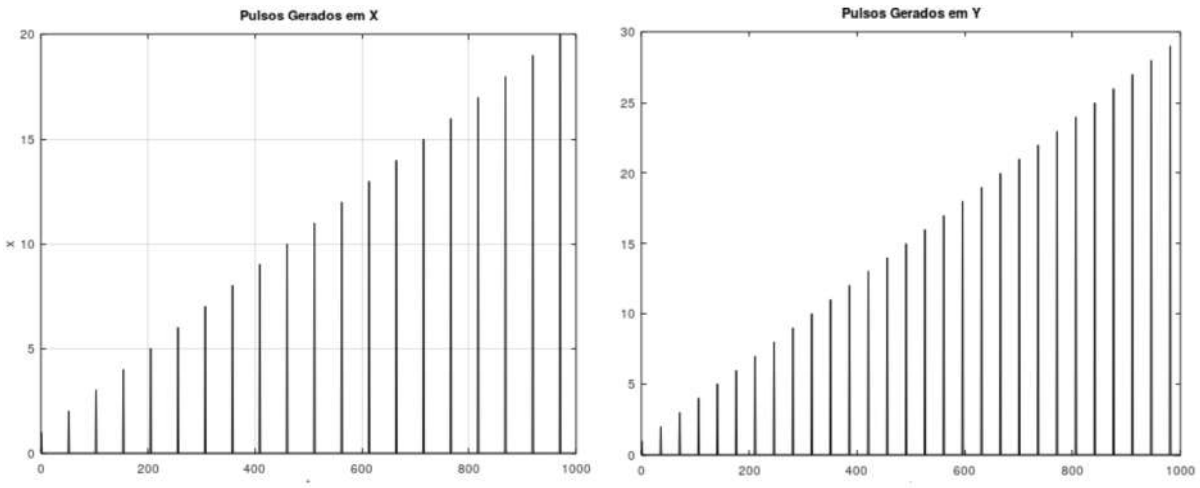

Figura 3.12. Exemplo da saída do gerador de pulsos

Na Fig.3.12 pode-se ver um movimento $\Delta X = 2$ e $\Delta Y = 3$ para uma resolução espacial de $10 pulsos/unidade$ e uma $Frequnciabase = 1000 pulsos/janelatemporal$. Nota-se que estes pulsos são equidimensionados na janela temporal o que é essencial para a realização de um encadeamento de movimento suave entre janelas. Pontua-se que a velocidade dentro de uma janela temporal é fixa e que esta deverá ser de dimensão compatível com as curvas de aceleração utilizadas. Com efeito, a variação de velocidade de movimento entre uma janela e a sua subsequente não deve ser grande o suficiente a ponto do controlador de posição do atuador não consiga resolver, evita-se assim impulsos (derivada da aceleração) que possam sobrecarregar o sistema de atuação.

3.2.3 Tradutor de Cinemática

O modulo de tradução de cinemática é responsável por:

- Re-amostrar o vetor contendo os pulsos de movimento cartesiano (setpoint de movimento) realizando a integral e transformando de vetor de pulsos para um valor escalar de delta de movimento;
- Traduzir o delta de movimento cartesiano para um delta de movimento angular de entrada da cinemática de atuação que vá posicionar o robô em numa configuração o qual coloca o efetuador na posição cartesiana desejada, isto através da solução analítica das cinemáticas inversa e direta;
- Gerar um novo setpoint de movimento em formato de vetor de pulsos dentro de uma nova janela temporal de interpolação (*time slice*).

Re-amostragem de setpoint de movimento e integrador

Este submódulo, descrito abaixo no diagrama 3.13, é responsável por detectar e somar os pulsos contidos num vetor de pulsos em uma janela temporal fornecendo como saída o valor escalar desse *quantum* de movimento.

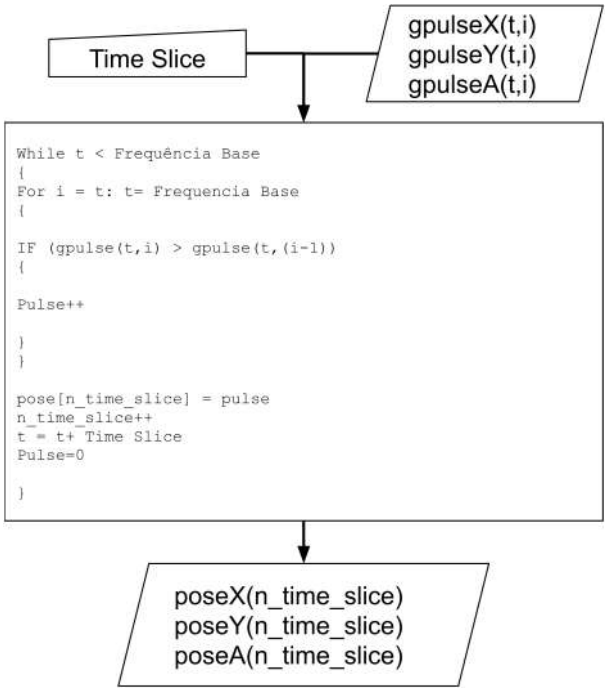

Figura 3.13. Captura de pulsos simulada

Pode-se indagar como será tratada a aceleração do movimento neste ponto, na verdade não se trata. Isto com o objetivo de avaliar se uma janela temporal curta o bastante é capaz de realizar uma trajetória dentro dos limites aceitáveis de aberração de movimento sem que esta precise ser tratada. Isso faz parte do conceito de janelamento do tempo, quando se fatia o tempo em pequenos pedaços trata-se o movimento dentro dessa fatia como um movimento retilíneo uniforme. Assim, um delta de posição a cada delta de tempo (largura da janela temporal) ou seja, a velocidade é constante dentro da janela, mas não o é entre as janelas. Assim a variação do delta de movimento entre duas janelas adjacentes é a aceleração, e o delta de posição dentro de uma janela é a velocidade do movimento. Espera-se que se a janela de re-amostragem da tradução for pequena o bastante a variação o residual entre duas janelas de entrada cartesianas também o será.

Cinemática inversa e direta

O módulo da cinemática inversa e direta á responsável por receber a posição alvo da ferramenta vinda da re-amostragem e integração do setpoint de movimento cartesiano. E então, calcular a posição das juntas $P1$,$P2$ e $P4$ para que essa posição alvo seja atingida. Em seguida, calcular a cinemática direta informando qual o ângulo o qual os links $a1$,$b1$ e c devem estar em relação ao eixo X para que essa configuração alvo seja atingida, mais bem visualizada na Fig.3.18. Pode-se observar o diagrama do gerador de fatia temporal em 3.14.

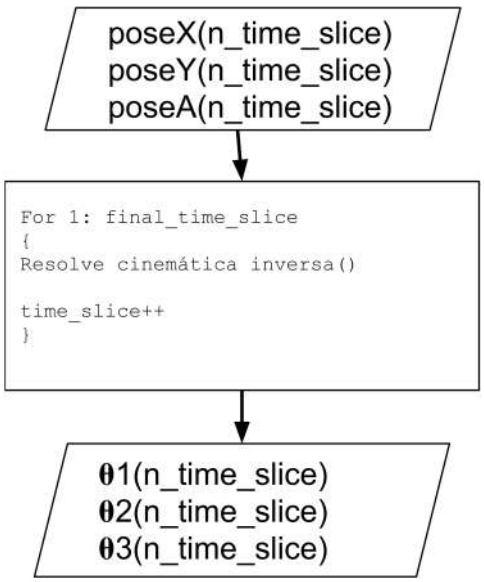

Figura 3.14. Gerador de fatia temporal de tradução

Pode-se notar na Fig.3.15 os pontos de interseção das circunferências características dos links que compõem o robô, ressaltando a posição final dos links anteriores a elas. Ou seja, a solução da cinemática inversa do robô capaz de atingir o alvo.

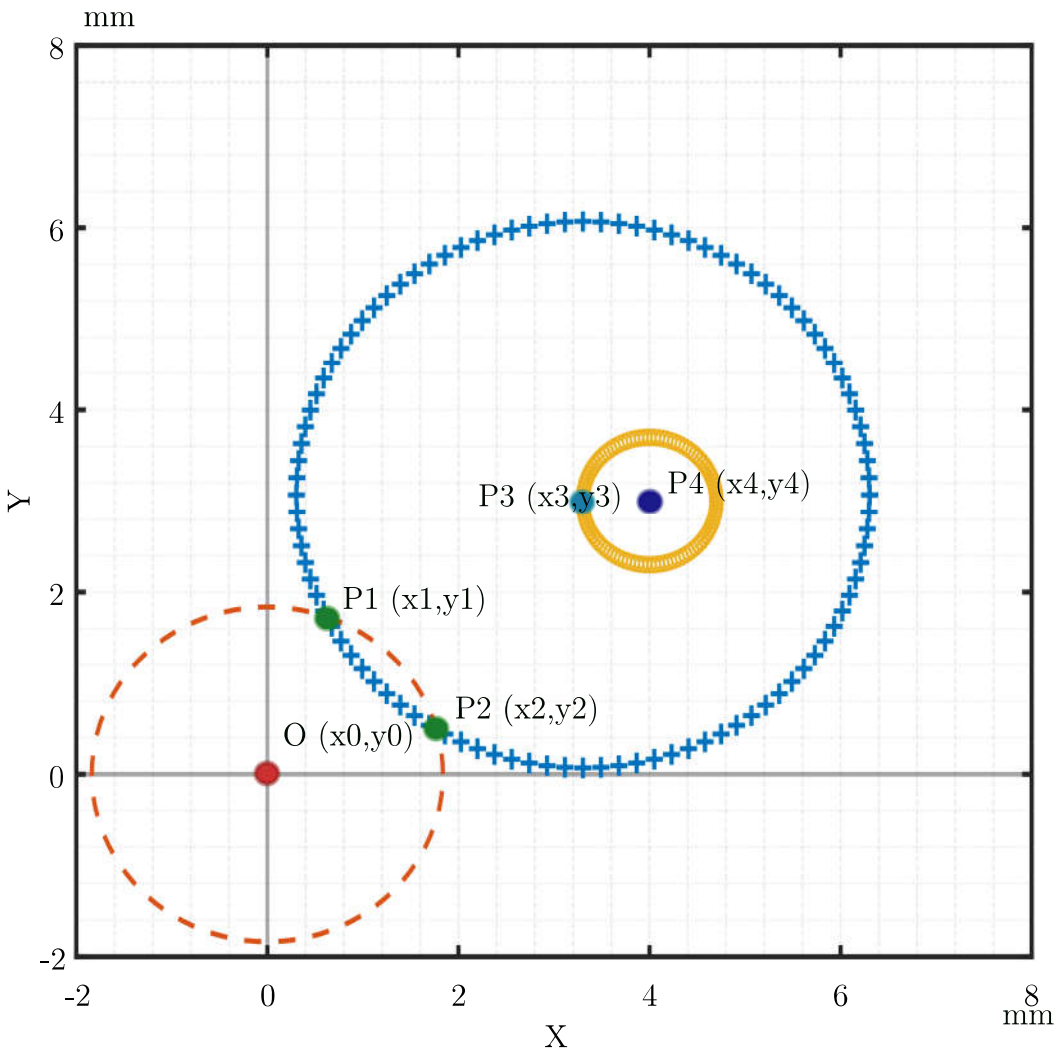

Figura 3.15. Plot do resultado da tradução de cinemática, posição das juntas

Aplicando a solução cinemática direta pode-se saber quais os ângulos de atuação

dos links e assim saber qual a configuração do robô para que a ferramenta se posicione e oriente no ponto comandado. Observa-se na Fig.3.16 a posição dos links obtida pela resolução da cinemática direta.

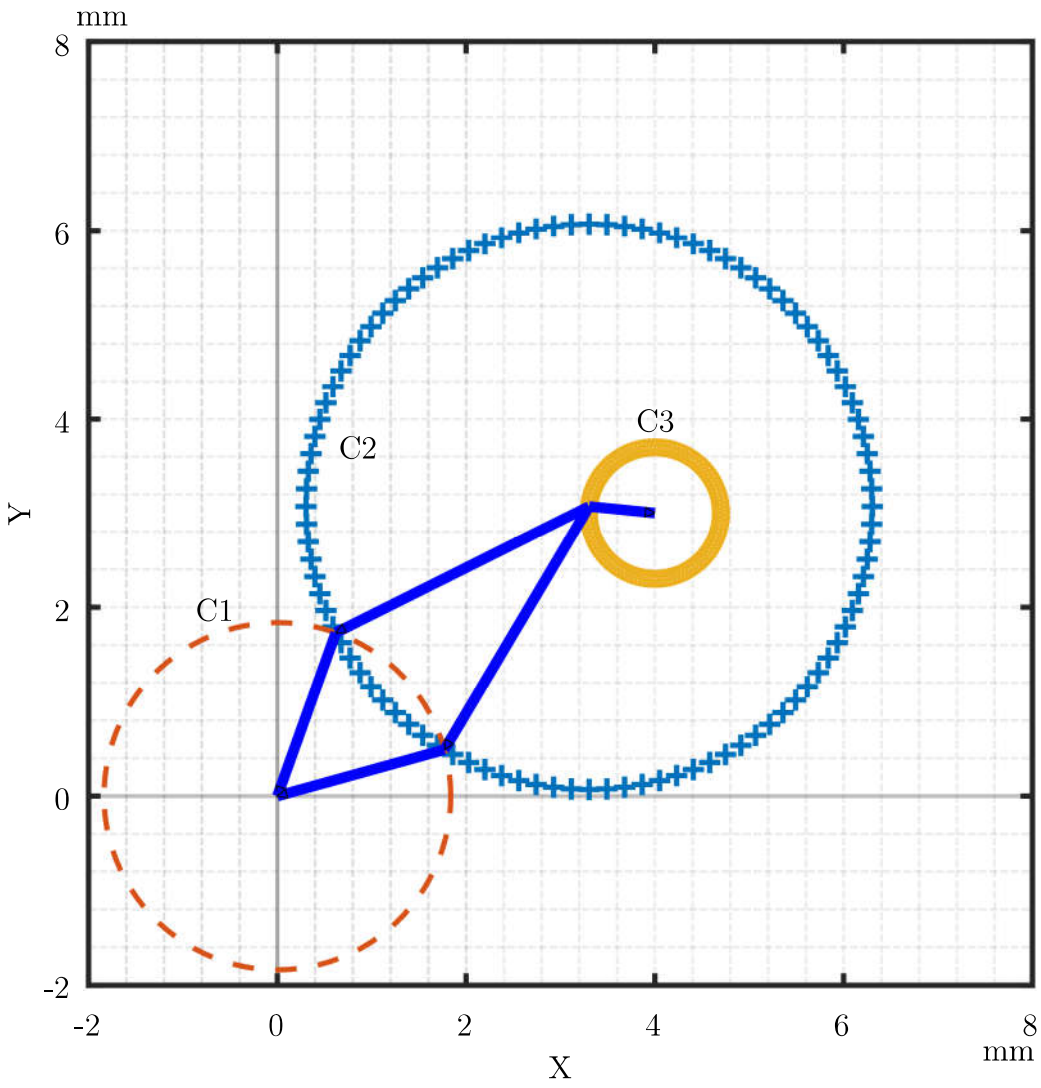

Figura 3.16. Plot do resultado da tradução de cinemática, posição dos links.

Cinemática de atuação de juntas e Geração de setpoint de movimento de saída

A solução da cinemática de atuação de juntas trata-se da transformação da posição angular alvo dentro da fatia temporal em um quantum de setpoint de movimento dentro dessa fatia. Assim re-discretizando o a posição em uma quantidade de movimento num sinal de pulsos e direção equivalente a essa quantidade específica de movimento. Como o movimento do atuador é angular (por exemplo, redução mecânica à correia síncrona) e o movimento das juntas também é angular, não existem transformações adicionais de movimento. Pode-se notar na Fig.3.17 que assim como no gerador de pulsos do gerador de trajetória cartesiana, a entra da no módulo de geração de pulsos do setpoint traduzido tem como entrada um delta de movimento, uma frequência base e uma resolução espacial.

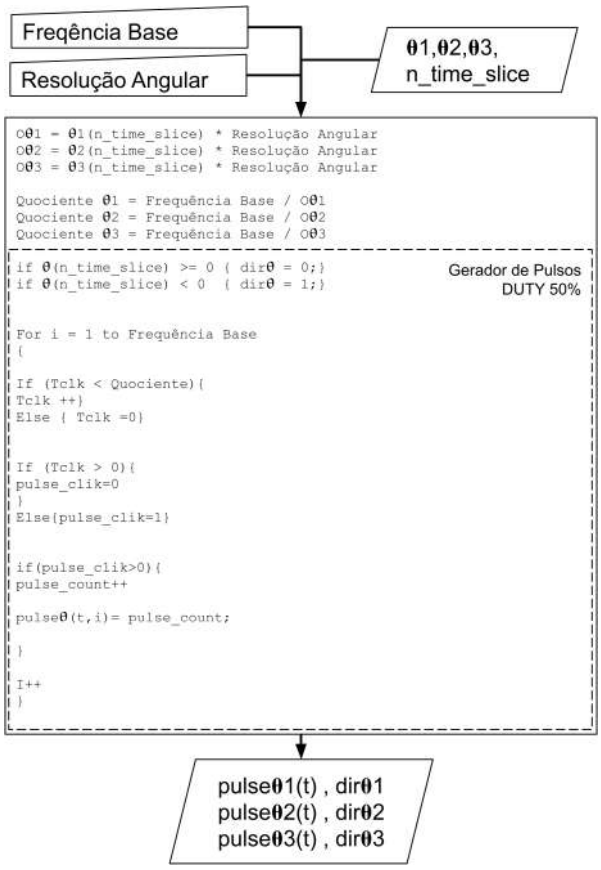

Figura 3.17. Gerador de setpoint de movimento de saída do tradutor

Na Fig. 3.18 pode-se ver a representação gráfica do resultado da tradução após a solução da cinemática de atuação:

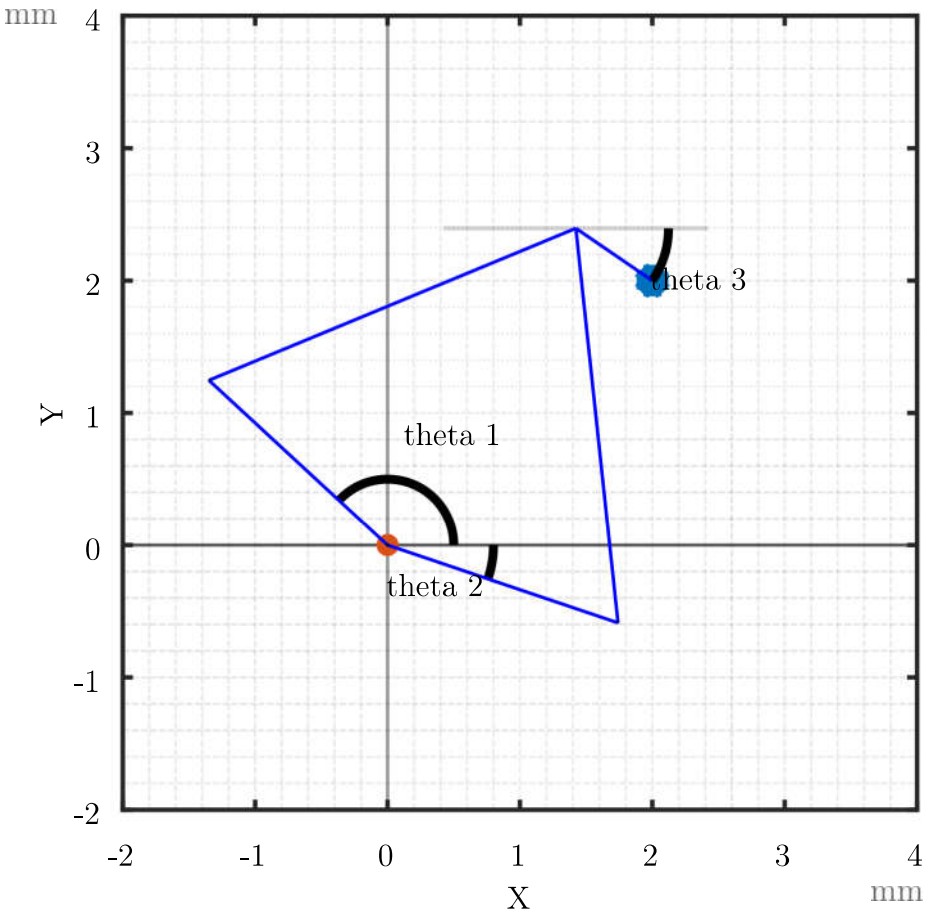

Figura 3.18. Plot do resultado da tradução de cinemática de atuação

3.2.4 Modelo Matemático do Robô

No modelo matemático do robô simula-se: A re-amostragem da entrada do driver de controle de movimento da junta; A integração desse quantum de movimento na configuração do robô; Aplica-se novamente a cinemática de atuação juntas para calcular-se a o Ângulo de posição de cada atuador; E a cinemática direta para calcular a posição de cada junta e cada link do robô; Pode-se assim por fim calcular a posição e orientação final da ferramenta pelo setpoint de movimento recebido do tradutor de cinemática. Finalizando na comparação com a posição alvo comandada pelo gerador simulado de trajetória cartesiana. Observa-se o resultado da simulação do robô na Fig.3.20 onde é demonstrado um movimento de um giro completo da orientação da ferramenta mantendo o *TCP* fixo. O diagrama do modelo pode ser observado em 3.19.

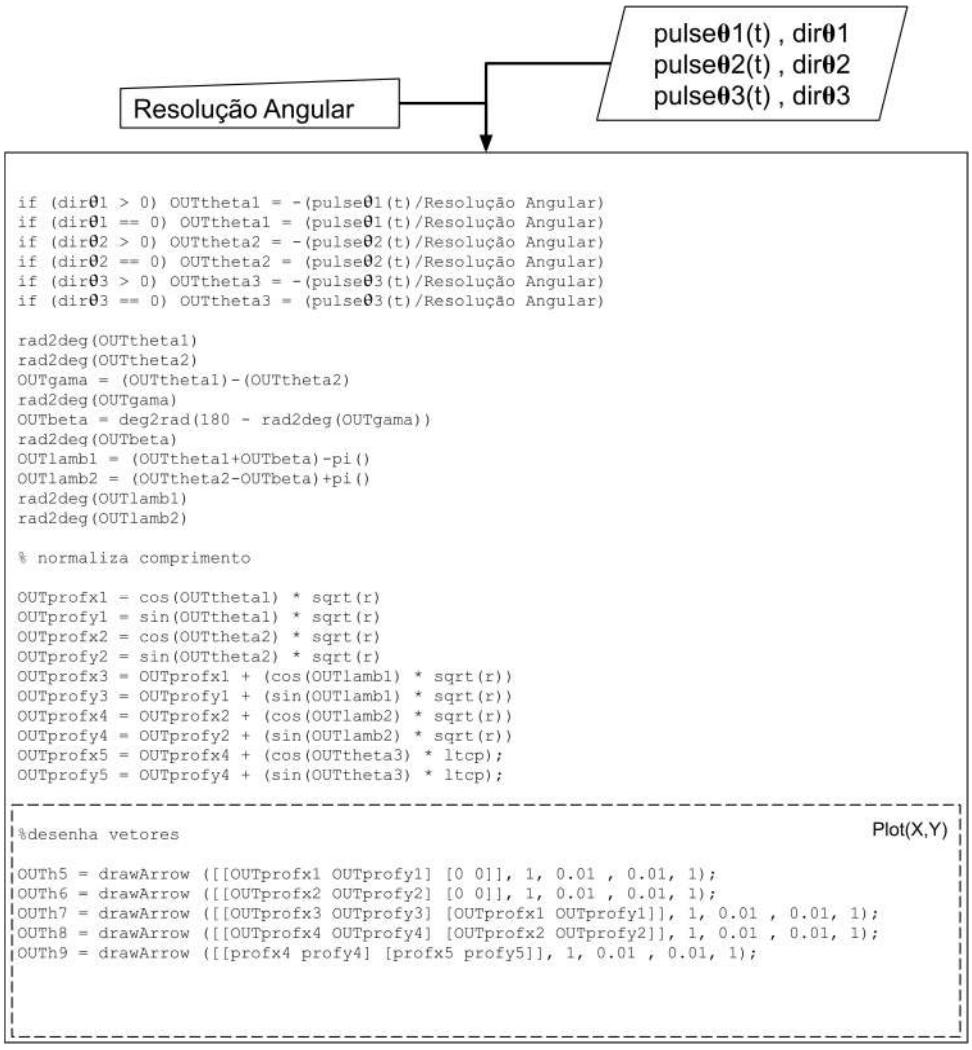

Figura 3.19. Modelo matemático do robô, baseado na cinemática direta do mesmo.

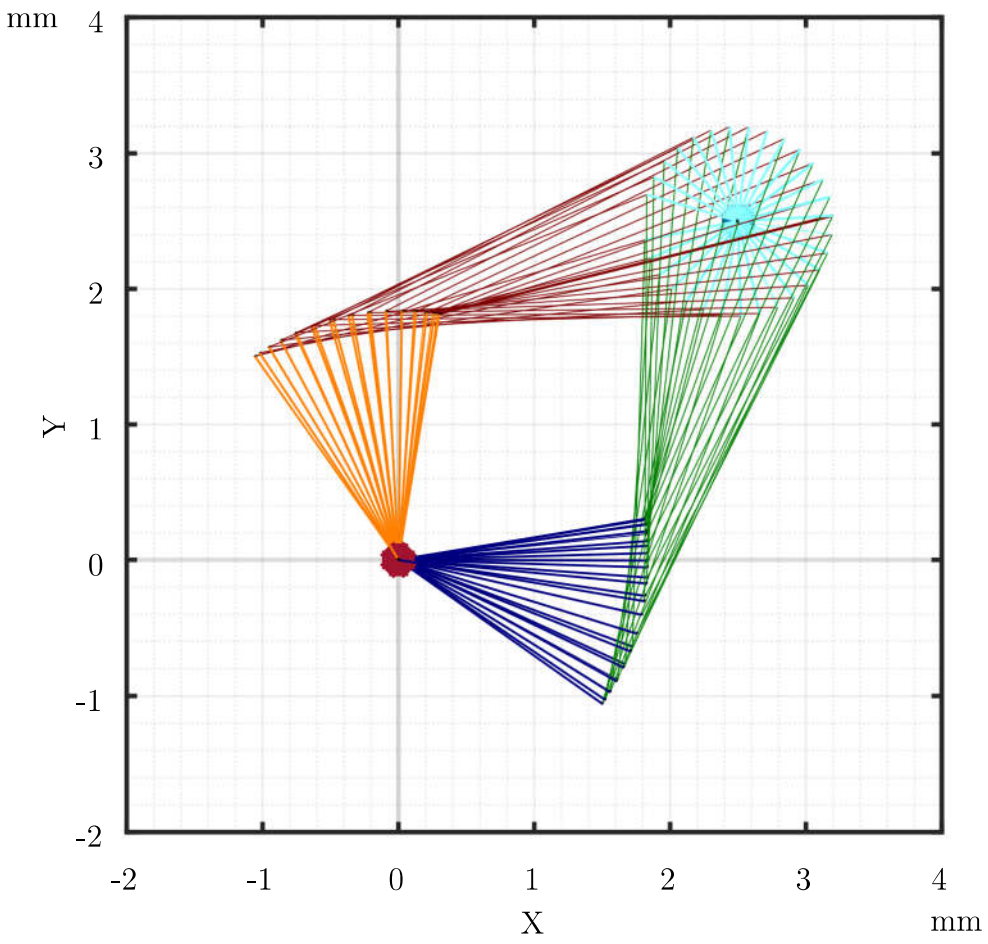

Figura 3.20. Resultado da tradução de cinemática, de um giro completo na orientação da ferramenta

Análise de erros e aberrações de posição e movimento

Com a posição e orientação final do robô pode-se então comparar a trajetória realizada em relação a trajetória gerada na entrada do sistema. Para isso utiliza-se primeiramente uma função de distância entre os pontos desejados e os executados. Devido a existência de um atraso em relação as variáveis de entrada e saída relacionado ao tempo de processamento e re-amostragens, é necessária uma normalização da trajetória de entrada e saída em relação ao tempo de execução. Uma análise do tempo morto entre o início e final do movimento será realizada em separado, mas atrasos durante a execução da trajetória gerando aberrações de movimento estão inclusos na análise de erro e são esperados principalmente em pontos de inflexão de aceleração onde as re-amostragens podem causar um efeito de suavização da trajetória.

A função erro implementada realiza a regressão polinomial de 6ª ordem tanto na trajetória setpoint gerada como na trajetória executada pelo algoritmo de simulação, isto para que o número de pontos de entrada e saída possa ser normalizado e os dois movimentos possam ser comparados, já que é executado um *upscale*, ou seja um aumento da resolução do movimento pelo algoritmo de tradução.

3.3 Experimentação

Neste capítulo define-se a metodologia dos testes e experimentos os quais será submetido o desenvolvimento, para assim realizar análises e comparações do desempenho do sistema.

Inicialmente realiza-se testes de desempenho de cada módulo do sistema, a fim de se levantar os limites de cada módulo. E assim definir quais parâmetros chaves de configuração a serem testados. E qual a influência esperada destes paramentos na performance do sistema. Posteriormente são definidas as variáveis de entrada e saída do sistema para que se compare os diferentes conjuntos de parâmetros. Os testes são baseados numa sequência aleatória de variação dos parâmetros para uma mesma entrada do sistema. Com as respostas a cada conjunto de parâmetros pode-se realizar a análise de covariância entre os parâmetros e a influência de cada paramento na reposta do sistema.

3.3.1 Metodologia DOE

O planejamento de experimento ou DOE (*Design of Experiment*) tem a seguinte definição:

> "É uma técnica utilizada para se planejar experimentos, ou seja, para definir quais dados, em que quantidade e em que condições devem ser coletados durante um determinado experimento, buscando, basicamente, satisfazer dois grandes objetivos: a maior precisão estatística possível na resposta e o menor custo. É, portanto, uma técnica de extrema importância para a indústria pois seu emprego permite resultados mais confiáveis economizando dinheiro e tempo, parâmetros fundamentais em tempos de concorrência acirrada. A sua aplicação no desenvolvimento de novos produtos é muito importante, onde uma maior qualidade dos resultados dos testes pode levar a um projeto com desempenho superior seja em termos de suas características funcionais como também sua robustez."[Tahara S., 2014].

O planejamento dos experimentos a seguir é baseado no passos definidos por [Coleman & Montgomery, 1993] :

- Definição do problema: Objetivos e ambiente;
- Definição das variáveis de entrada: variáveis de controle e constantes;
- Definição das variáveis avaliadas: variáveis de resposta e de distúrbio;
- Escolha do tipo de modelo experimento
- Experimentação: interações e restrições;
- Análise de dados
- Conclusões

Definição do problema: Objetivos e Ambiente

Segundo [Coleman & Montgomery, 1993], os objetivos do experimento devem procurar ser sem vieses, específicos, mensuráveis e uma consequência prática do experimento em si. Isso permite que as conclusões sejam uma consequência dos dados obtidos, ou seja, de fácil verificação. A análise do ambiente de experimentação deverá ser feita de forma a suportar os objetivos, e devem incluir o estudo do estado da arte do sistema a ser experimentado. Assim fornecendo a informação necessária para a formação do contexto do experimento deixando claro qual a justificativa de se realizar o experimento, ou seja, qual a informação importante deseja-se obter realizando-o.

Sendo assim o objetivo do experimento em questão é verificar a influência dos parâmetros de configuração do gerador de setpoint de movimento na resposta do sistema de tradução on-line de cinemática implementado de forma a obter-se os limites e o erro máximo do sistema em função destas configurações.

Definição das variáveis de entrada: variáveis de controle e constantes;

As variáveis de controle são as que serão manipuladas no experimento, estas devem ser listadas assim como a previsão da influência dessas no experimento e os limites de atuação devem ser estabelecidos previamente a realização do experimento.

Tem-se que [Coleman & Montgomery, 1993] define que os limites das variáveis de controle devem partir de valores historicamente usuais para que o modelo de experimento possa ser validado. Numa segunda etapa, esses limites devem ser colocados de forma que, um efeito observável possa surgir nas variáveis de resposta. O procedimento de medição das variáveis deve ser discutido previamente ao experimento, assim como sua exatidão e

precisão. Essas informações devem ser utilizadas para realizar o cálculo da propagação de incertezas do ambiente de experimentação.

As variáveis de entrada do experimento são:

- Posição inicial da trajetória escolhida;
- Período da janela de geração de setpoint cartesiano de movimento;
- Resolução temporal de comando do gerador de setpoint cartesiano de movimento;
- Resolução espacial de comando do gerador de setpoint cartesiano de movimento;
- Período da janela de atualização de tradução de trajetória;
- Resolução temporal de comando do gerador de setpoint angular de movimento;
- Resolução espacial de comando do gerador de setpoint angular de movimento.

As constantes do ambiente de experimentação devem ser discutidas previamente, para que os possíveis distúrbios causados por essas possam ser preditos e identificados com maior facilidade. No geral as constantes são fatores controláveis, ou que não podem ser variados, ou o efeito de sua variação não são objetos de interesse do experimento. Segundo [Coleman & Montgomery, 1993] é uma boa prática manter essas variáveis em valores usuais ou até mesmo nominais.

As constantes do experimento são:

- A trajetória a ser executada;

Definição das variáveis avaliadas: variáveis de resposta e de distúrbio

As variáveis de resposta de um experimento segundo [Tahara S., 2014] são aquelas que se deseja medir a variação de acordo com a manipulação das variáveis de entrada. A escolha prévia de uma variável de resposta pode ser algo complexo caso o experimento não seja realizado num processo maduro e bem conhecido como dito por [Coleman & Montgomery, 1993]. E de preferência deve ser uma variável que seja alterada continuamente pelas variáveis de entrada, respostas discretas são difíceis de serem preditas. As variáveis de resposta devem informar o máximo sobre um fator de interesse e serem quantizadas numa escala condizente com o processo, podendo assim serem comparadas com algum tipo de objetivo. Os graus de incerteza das variáveis de resposta devem ser os mais constantes possíveis dentro dos limites do experimento e se possível preditos pela propagação de incerteza do experimento.

As respostas de um experimento devem variar, é normal que o sistema de medição tenha a imprecisão associada e essa variação cause distúrbios nas variáveis de resposta. Segundo [Coleman & Montgomery, 1993] o importante em um experimento é que essa variação esteja dentro dos limites esperados. Os distúrbios no geral são variáveis que não podem ser controladas no ambiente do experimento. Estes não devem ter sua influência grande o bastante para alterar os resultados do experimento. Caso um distúrbio cause grande influência no experimento ele deverá ser analisado como co-variável na análise, e um cuidado adicional deve ser tomado pois a variação imprevisível de um distúrbio de grande influência torna-se um risco de experimento e pode elevar muito a incerteza do experimento.

As variáveis de saída do experimento são:

- O erro relativo entre a trajetória planejada e a trajetória executada, calculado através de uma média do desvio da distância entre a trajetória programada e executada;

Experimentação: interações e restrições

Interações entre variáveis são geralmente difíceis de ser estimadas intuitivamente, principalmente quando se trata de um experimento ou processo novo. A análise de covariância e tabelas e gráficos de interação como exemplificado por [Coleman & Montgomery, 1993], podem ser uma excelente ferramenta para evidenciar o comportamento interativo entre duas variáveis, indicando assim correlações não obvias à primeira vista. Tem-se que [Coleman & Montgomery, 1993] cita que experimentos teóricos, modelagens computacionais, e experimentos práticos são situações muito diferentes e deve-se dar importância a estudar as restrições aplicadas aos três modelos que normalmente são bem distintas e fazem muita diferença nos resultados.

Escolha do tipo de modelo experimento

Segundo [Tahara S., 2014] a escolha do tipo de planejamento deve considerar o tamanho da amostra o tempo e recurso para se realizar as rodadas e as restrições do processo a ser experimentado. A escolha do modelo de experimento é uma consequência do planejamento supracitado e os modelos se diferenciam em geral de acordo com:

- Quantidade de fatores e efeitos desses fatores;
- A separação desses fatores em grupos e blocos ou não, principalmente para estudos com grande número de fatores;
- Quanto a aleatoriedade ou não do experimento;

- Se algum dado deverá ser inferido, estudos com número de fatores muito grande ou impossibilidade de medição do fator.

- Quanto a necessidade de hierarquizar os fatores.

- Se existe a necessidade da interação de mais de dois fatores ser indicada de forma gráfica.

Em um estudo anterior [Enescu & Alexandru, 2014]P629 testou os seguintes modelos de experimento para estudo de dinâmica de robôs industriais: Fatorial, *Planckett-Burman* e *D-Optimal*. Sendo que o melhor resultado encontrado foi com o modelo fatorial. Citando também que este modelo foi o de mais fácil compreensão da interação dos fatores do experimento.

A descrição de [Tahara S., 2014] para o modelo fatorial é: o modelo indicado para análise de vários fatores, em dois ou mais níveis, onde a interação entre os fatores é importante. A cada rodada de experimento todas as combinações possíveis entre os fatores são tratadas. A alteração das variáveis de entrada em cada rodada de experimentação é feita de forma aleatória. Com isso obtém-se: Estimativas e comparação dos efeitos dos fatores nos resultados; estimativa dos efeitos de interação dos fatores; A estimativa da covariância.

Devido ao bom resultado prévio com experimentos em processos semelhantes. Sendo que os fatores a estudados são compatíveis com a as informações que podem ser obtidas. Opta-se pelo modelo fatorial para o planejamento do experimento.

3.3.2 Planejamento de experimento:

O objetivo do experimento é: Definir a influência dos parâmetros do sistema no desempenho deste; entender como se dá a interação entre as amostragens e re-amostragens realizadas durante a tradução de cinemática; detectar através da análise de covariância dos parâmetros uma região onde o desempenho seja aceitável.

O experimento será realizado na plataforma simulada pois é um ambiente extremamente controlado, isolando o sistema de imprevistos e variações não controladas. Para o funcionamento efetivo de um experimento do tipo fatorial é extremamente importante que os distúrbios do sistema sejam conhecidos e mantidos dentro de limites bem estabelecidos. No atual estágio do desenvolvimento não está disponível uma montagem física real do robô com o grau de exatidão e repetibilidade exigida para isso. Porém testes e experimentos reais serão realizados com a plataforma computacional embarcada real, a fim de se comprovar a eficácia do sistema.

Parâmetros de entrada:

Os parâmetros escolhidos para realização dos experimentos serão:

- A - Resolução temporal da geração de pulsos do gerador de setpoint de trajetória:

  Estes parâmetros define a resolução pulsos/segmentos do gerador de pulsos, porém como o segmento temporal simulado é fixo ele representa a frequência máxima o qual os pulsos podem ser gerados. De acordo com os testes preliminares pode-se definir os limites desta variável entre 100 a 1000 pulsos/ segmento.

- B - Resolução espacial do gerador de setpoint de trajetória:

  Variável que define a razão entre pulsos/unidade de medida, ou seja, define qual é a distância a ser percorrida a cada pulso de movimento. Nos testes preliminares obteve-se os limites de 50 a 500 pulsos/unidade.

- C - Posição Inicial da trajetória:

  Parâmetro define o ponto de início da trajetória a ser executada, espera-se que a análise da variação do desempenho do sistema em diferentes pontos iniciais da execução de uma mesma trajetória possa indicar se existe algum tipo de falha na cinemática inversa e direta. Estipula-se seu limite de P(1,2 ; 1,2) a P(2,1 ; 2,1) unidades, para que se não ultrapasse os limites da área de trabalho do robô.

- D - Período da janela temporal de tradução

  Variável que define o tamanho da janela de re-amostragem, integração e processamento dos segmentos de tradução da trajetória.

  A análise desta variável e sua contribuição na variação do desempenho do sistema, é um ponto fundamental para validar e otimizar o sistema de tradução. Esse período, devido a simulação estática matemática, se dá em números de interações temporais da frequência base do gerador de setpoint. Assim sendo, ela não está totalmente independente desse fator, ou seja, representa uma razão entre frequência máxima o número de instantes dela. Isso diminui muito o volume de dados necessários para se executar a simulação do sistema, realizando uma compressão temporal dos dados. É realizado um encadeamento de todos os segmentos gerados pelo gerador de setpoint cartesiano, mantêm-se assim o desacoplamento de sincronismo entre os segmentos de comando de movimento cartesiano e as janelas temporais de tradução.

  Nos testes preliminares obteve-se os limites de teste desta varável entre 100 e 1000 instantes da frequência base de geração de pulsos do gerador de setpoint de trajetória.

- E - Resolução temporal de geração de pulsos da saída do tradutor de trajetória

  Esta resolução é a frequência máxima o qual pode-se gerar pulsos dentro da janela temporal de tradução. A geração traduzida é dessincronizada do gerador de setpoint cartesiano, podendo-se, inclusive, realizar *upscaling* da resolução do setpoint para o driver de controle de movimento. Assim, elevando a frequência máxima e podendo proporcionando a utilização de sistemas com *microstep* menores e movimentos mais suaves.

  Nos testes preliminares obteve-se os valores limites de 100 a 1000 pulsos/janela de tradução.

- F - Resolução espacial do gerador de pulsos de saída do tradutor de trajetória.

  Assim como no gerador de setpoint de trajetória cartesiana, necessita-se de se ajustar a resolução espacial de saída do sistema de tradução. Principalmente porque a resolução original era linear e a traduzida é angular, logo deve ser ajustada de acordo com a resolução de controle do driver de movimento e as reduções dos atuadores das juntas rotativas.

  Nos testes preliminares obteve-se os valores limites de 100 a 1000 pulsos/rad.

Modelo de Experimento

O modelo de experimento escolhido é o e fatorial randomizado de dois níveis, e para a criação da tabela de experimento e análise será utilizado o Software Minitab® 19.2020.1 (64-bit) da LLC©. A análise da covariância multifatorial também será realizada nesta plataforma.

A configuração do experimento é a seguinte:

- Modelo: Fatorial de dois níveis Randomizado
- Número de Fatores: 6
- Número de pontos centrais: 10
- Número de blocos: 1
- Variável de saída: Erro médio da trajetória executada;

Essa configuração gerou uma tabela que contém os valores dos fatores e serem utilizados em cada rodada, gerando assim um valor da variável de saída para cada rodada. No total serão realizadas 74 rodadas de experimentação, 64 rodadas em dois níveis e 10 com pontos centrais. Apesar da literatura previa ter obtido resultados ótimos com o modelo Fatorial Completo o número de fatores torna inviável a utilização deste modelo pois seria necessário um número no mínimo 10x maior de rodadas de experimentação.

3.4 Implementação

Esta sessão descreve como a solução desenvolvida e simulada deverá ser implementada em uma plataforma computacional real afim de se testar a funcionalidade do sistema. A Fig.3.21 descreve a estrutura do sistema onde a solução será implementada e testada.

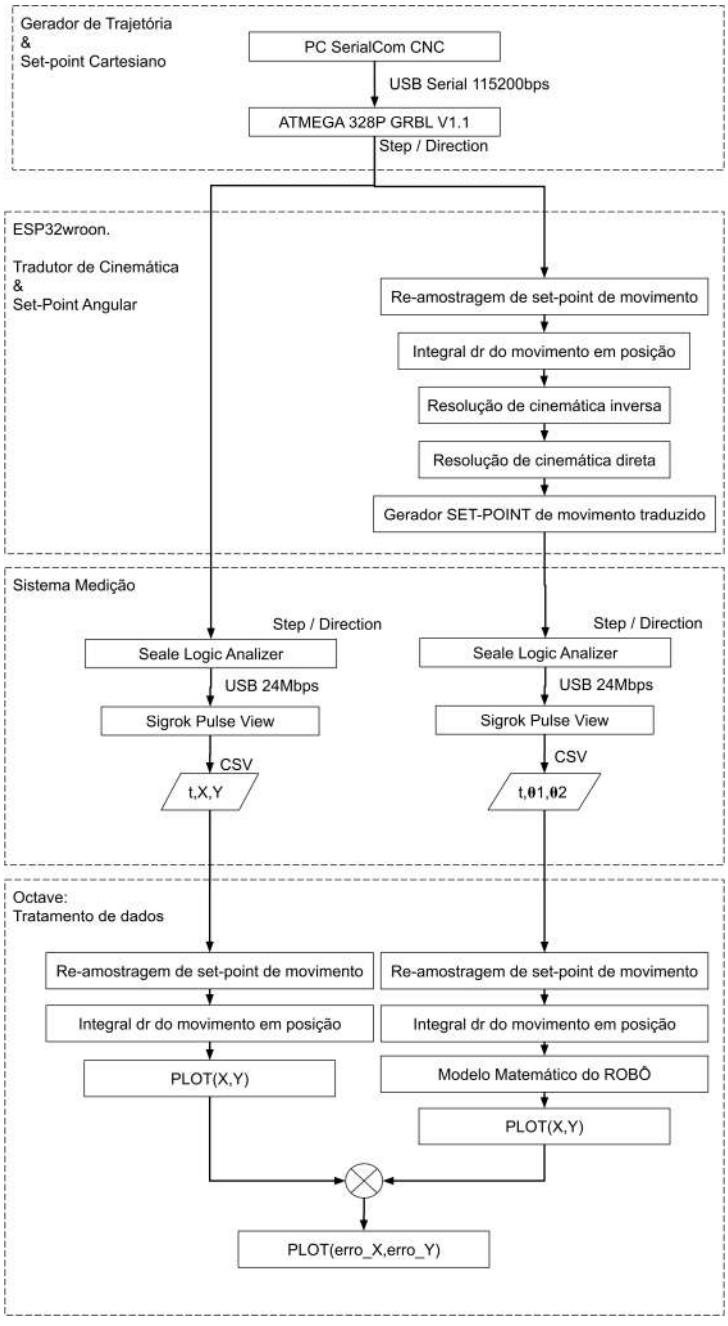

Figura 3.21. Diagrama do modelo computacional implementado

E a Fig.3.22 demonstra como esta estrutura se encaixa no sistema de comando e controle de um robô completo real.

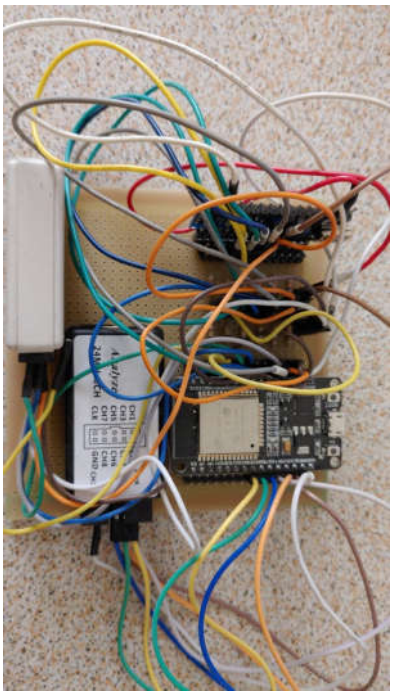

Figura 3.22. Modelo computacional real implementado

3.4.1 Seleção da arquitetura de hardware

A arquitetura baseia-se em:

- Um gerador de trajetória que interpreta um caminho contínuo e gera uma lista de comandos encadeados definindo a trajetória a ser executada;
- Um gerador de movimento que interpreta os comandos e interpola o movimento separado de cada eixo para a execução do movimento global no espaço cartesiano. Gerando assim o setpoint de movimento cartesiano;
- Um tradutor de movimento que interpreta o movimento no espaço cartesiano de ferramenta e o traduz para o espaço de juntas. Traduzindo o o setpoint de movimento cartesiano para o o setpoint de movimento angular apropriado ao robô;
- Um conjunto de servo atuadores que recebem o setpoint de movimento angular do tradutor e executando-o.

A fim de simplificar a montagem prática do modelo, esta não contará com a malha final de atuação (drivers e motores e atuadores). Com efeito espera-se que os recursos

sejam focados no estudo dos erros e distorções adicionados ao sistema somente pelo sistema de tradução de trajetória. A inclusão da malha de controle final incluiria também erros e distorções como: Re-amostragem do sinal digital na entrada do driver, parâmetros de controle da malha de controle do motor e folgas e erros mecânicos do robô. Estas folgas e erros são de difícil segregação caso somente o erro no movimento real efetivo do robô seja medido, isso tornar-se um distúrbio considerável e não controlado no experimento. O diagrama da Fig. 3.23 descreve a arquitetura de controle escolhida:

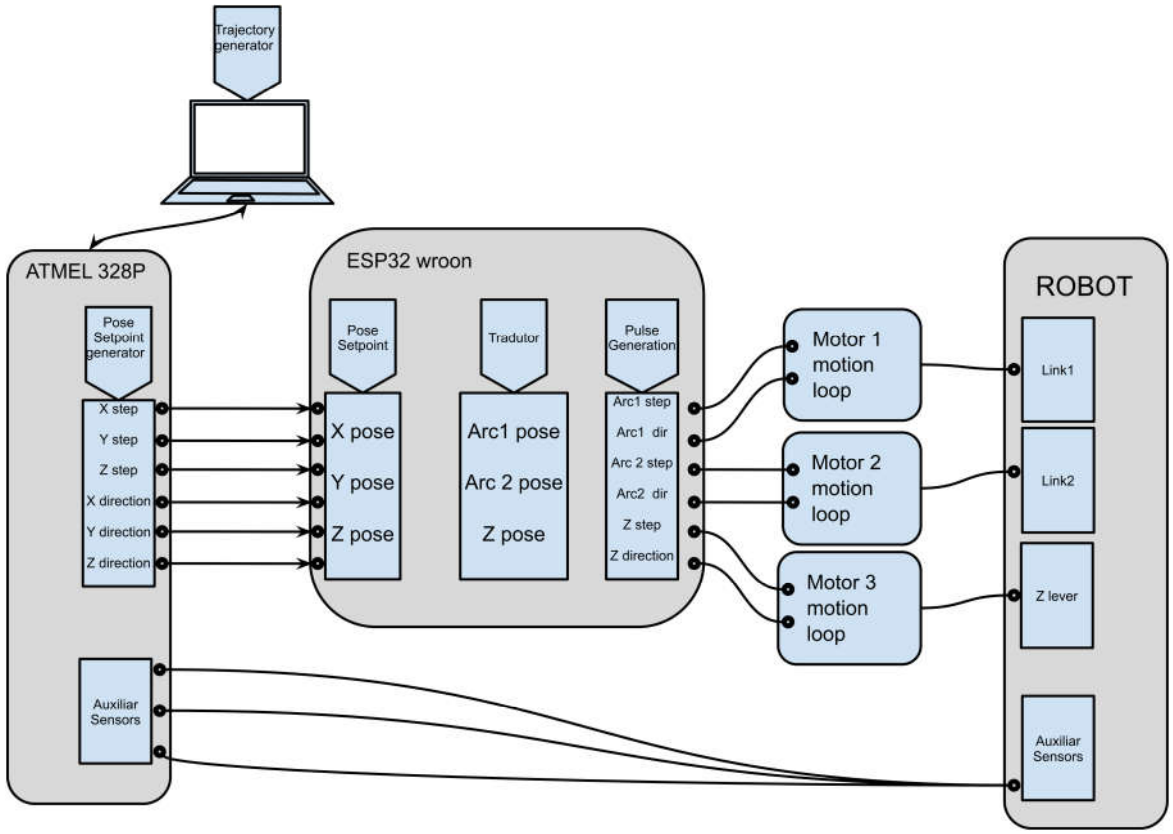

Figura 3.23. Arquietura de controle

3.4.2 Gerador de Trajetória

3.4.3 Gerador setpoint de Movimento

O gerador de setpoint de movimento escolhido é o *SerialCommCNC Gmbl*, como interpretador de trajetória e IHM, conectado a um *uC Atmega328p* com *GRBL Firm V1.1*, como gerador de pulsos. Esta escolha baseia-se na plataforma aberta e comumente utilizada na comunidade *maker*. Sendo assim uma opção aberta de baixo custo, podendo ser facilmente replicada. Apesar de utilizar a placa do *Arduino Nano* o *firmware* utilizado não é desenvolvido na plataforma *Arduino*, ele substitui completamente o *bootloader* do *uC* para que seja possível extrair o máximo de desempenho do hardware.

Especificações técnicas do softwae *SerialComCNC* por [Maassen, 2019] :

- Interface nativa de comunicação: *serial over USB (115200 Baud) com GRBL.*
- Capacidade para encadear e interfacear programas em G-Code para GRBL.
- Interface do tipo preventiva, onde os comandos são enviados antes de serem executados pelo interpretador e gerador de pulsos.
- Capacidade de envio de comandos *ON-the-fly* (durante um movimento);
- Gráfico com posição estimada da ferramenta.
- Interface com *pendant* de controle manual.
- Capacidade de troca de parâmetros de controle entre a execução de comandos;

Especificações técnicas da placa *Arduino Nano (uC Atmega328p)* com *firmware GRBL* V1.1:

- Até 30kHz de frequência de pulsos de saída para 3 eixos independentes e sem *jitter.*
- Interpreta a maioria dos comandos *G-code*, inclusive interpolações de arcos e hélice.
- *Buffer* de comandos de até 18 comandos armazenados antes da execução;
- Capacidade de espelhamento de eixos e referênciamento especial para atuação com dupla motorização.

A Fig. 3.24 mostra o hardware montado do comando de movimento.

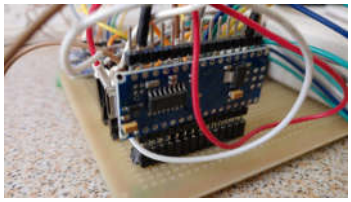

Figura 3.24. Gerador de setpoint de movimento cartesiano

3.4.4 Tradutor de setpoint de movimento

A plataforma computacional escolhida para a implementação da tradução de movimento é o *SoC ESP32wroom-32* da *Espressif*, isto devido a sua grande versatilidade e poder computacional com um custo baixo. Essa plataforma contém quatro características fundamentais para a implementação:

- Periférico de captura de pulsos de alta velocidade;
- Alta frequência de *clock* de trabalho, 240MHz;
- Boa capacidade computacional matemática, incluindo registradores de 32bits,
- Possibilidade de utilização de interrupções temporais determinísticas.

Na Fig.3.25 pode-se observar a arquitetura interna do *SoC* com todos seus periféricos. A montagem do hardware do tradutor pode ser visualizada na Fig. 3.26.

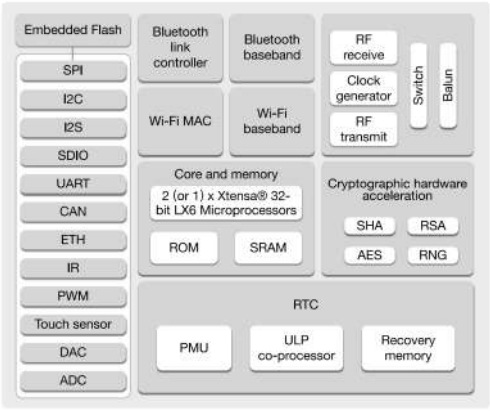

Figura 3.25. Diagrama Funcional, [Espressif, 2019]

O tradutor de movimento será dividido em três blocos de tarefas principais:

- **Bloco de captura de setpoint de movimento (pulsos e direção)**, obtendo assim a distância, velocidade, aceleração e direção do movimento em cada eixo

Figura 3.26. Tradutor de movimento

cartesiano do espaço de ferramenta, eixos X e Y, mais o ângulo de orientação da ferramenta referente ao eixo Z da ferramenta, eixo A;

- **Bloco de tradução**, neste bloco a quantidade de movimento no segmento é utilizada como entrada das transformações cinemáticas inversas e diretas, transformando o movimento cartesiano no espaço de ferramentas em um movimento angular no espaço de juntas.
- **Bloco de geração de movimento**, assim como no gerador de movimento cartesiano do interpretador de comandos de trajetória, após a tradução da cinemática é necessário gerar novamente o setpoint de movimento, agora no espaço de juntas, para que os servo-atuadores possam executar o movimento desejado.

Pode-se notar na Fig.3.27 como se dá a estrutura de agendamento temporal interna ao *SoC*, mesclando tarefas de baixa e alta prioridade, preemptivas e não preemptivas, de forma a se conseguir a performance requerida do sistema.

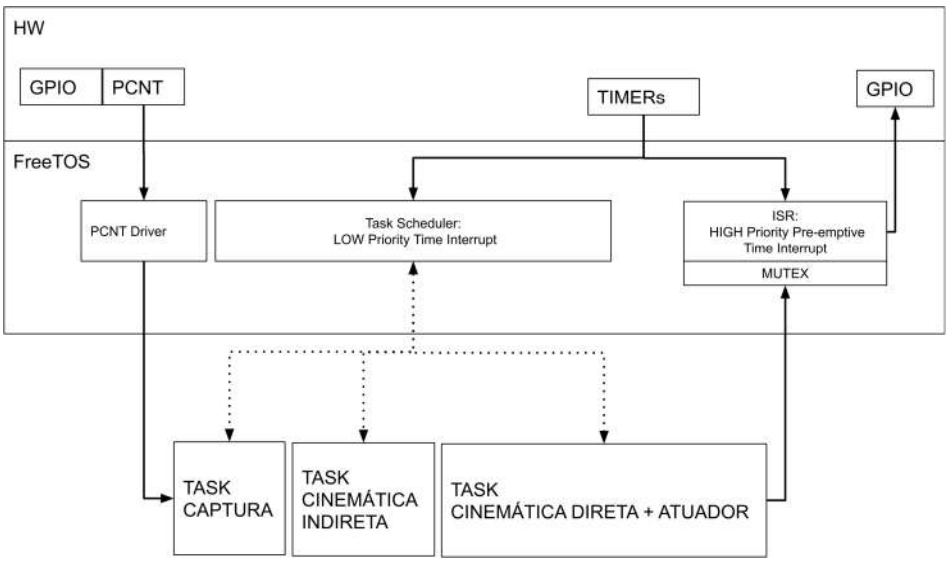

Figura 3.27. Diagrama de utilização de recursos de Hardware do SoC ESP3-wroon

Captura de setpoint de movimento cartesiano

É utilizada para essa tarefa o dispositivo nativo do *SoC ESP-32* para captura de pulsos, *(PCNT)*. Onde realiza-se a contagem da quantidade de pulsos ocorridos dentro de um determinado intervalo de tempo. Sabendo que a frequência de geração de pulsos dos geradores de movimento usualmente não ultrapassa 200kHz, limita-se a frequência de re-amostragem do sinal ao dobro desta frequência. Seguindo assim a diretiva de que para se capturar um sinal digital sem perda de informação, deve-se utilizar uma frequência de amostragem maior ou igual ao dobro da frequência máxima do sinal amostrado.

O objetivo da captura é que a ao final de cada ciclo da janela temporal de tradução, mantenha-se atualizada, a posição e orientação da ferramenta no plano de trabalho cartesiano, integrando assim os vetores de movimento adquiridos a cada janela.

Pode-se observar no diagrama da Fig.3.28 que a utilização de um dispositivo independente ao processamento simplifica bastante a lógica do sistema. Pois, não é necessário que interrupções externas sejam utilizadas, o que causariam variações aleatórias no tempo de excussão do programa reduzindo por efeito o determinismo do sistema.

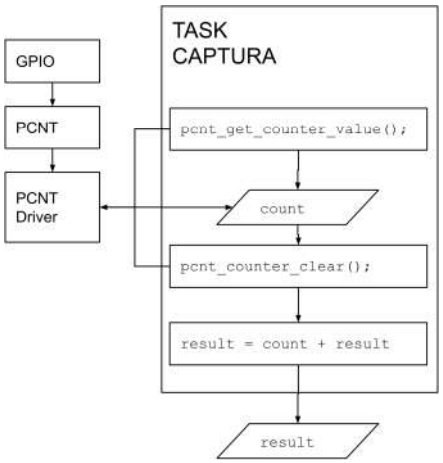

Figura 3.28. Fluxograma de captura

O dispositivo *PCNT* demonstra-se muito eficiente para realizar a aquisição de entrada, pois, tem uma frequência de trabalho muito superior a frequência máxima usual do sinal amostrado, 80MHz versus 200Khz. Isto, inclusive, poderia se tornar um problema, porque com uma frequência tão acima do necessário poderiam ser amostrados como pulsos de sinal: ruídos, harmônicos e repiques. porém, o próprio dispositivo já tem um filtro interno que limita a largura mínima dos pulsos a serem amostrados.

Uma vantagem do dispositivo em questão, é que ele já faz a soma e subtração da posição através do sinal de controle. Assim, ele nativamente já faz a aquisição de posição positiva e negativa dos sinais de pulso e direção. porém, tem-se que tomar o cuidado para que a janela temporal de aquisição do setpoint de movimento não seja muito longa. Caso contrário, podem ocorrer aberrações quando houver uma mudança de sinal de movimento. Contudo, geralmente toda mudança de sinal é precedida de uma curva de desaceleração e aceleração afim de se manter a continuidade e fluidez de movimento, e proteger os mecanismos de atuação contra impulsos elevados. Logo espera-se que, mesmo havendo uma aberração, está deve ser pequena e na forma de um adiantamento da inflexão do movimento. O exemplo do comportamento do modulo pode ser visto na Fig.3.29.

Outra possibilidade interessante é que se pode utilizar o próprio contador interno do dispositivo como acumulador de posição. Assim evitando possíveis perdas de pulsos durante a execução de interrupções já que o dispositivo é um hardware dedicado e não depende de compartilhar recursos com outras funções do *SoC*. Contudo, caso algum pulso de movimento seja perdido na hora da leitura do contador ele será introduzido na próxima janela temporal, e não descartado.

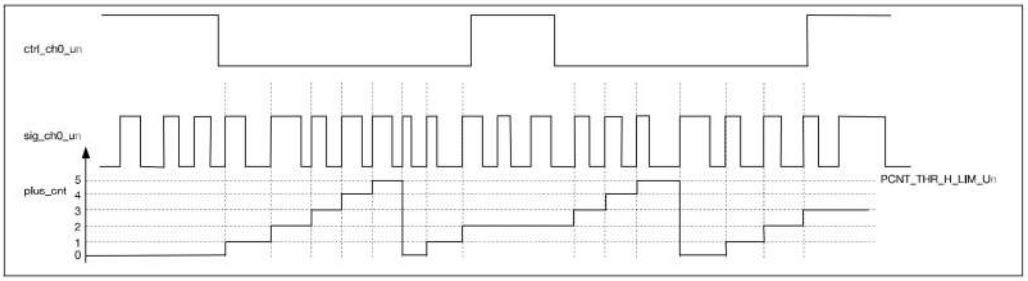

Figura 3.29. Exemplo de captura de pulsos executada pelo dispositivo PCNT, [Espressif, 2019]

Tradução do movimento

Utilizando os vetores obtidos na captura de setpoint, pode-se calcular qual a posição alvo do segmento no espaço de juntas. Através da solução das cinemáticas inversa e direta como observa-se nos fluxogramas das Fig.3.30 e Fig.3.31. Assim, partindo do ponto atual para o próximo ponto, tem-se a quantidade de movimento a ser executado dentro da janela temporal de tradução no espaço de juntas. Realizando a tradução do movimento interpolado no espaço cartesiano para o movimento interpolado no espaço de juntas e atuadores.

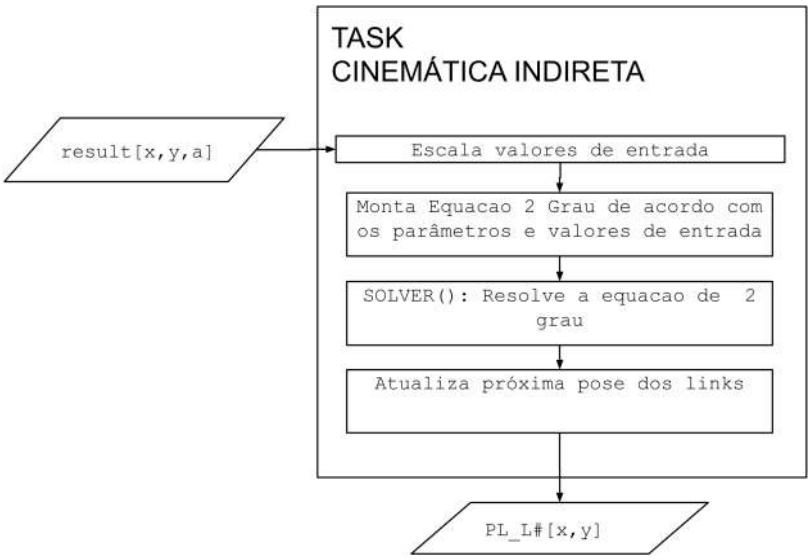

Figura 3.30. Fluxograma da solução analítica de cinemática indireta

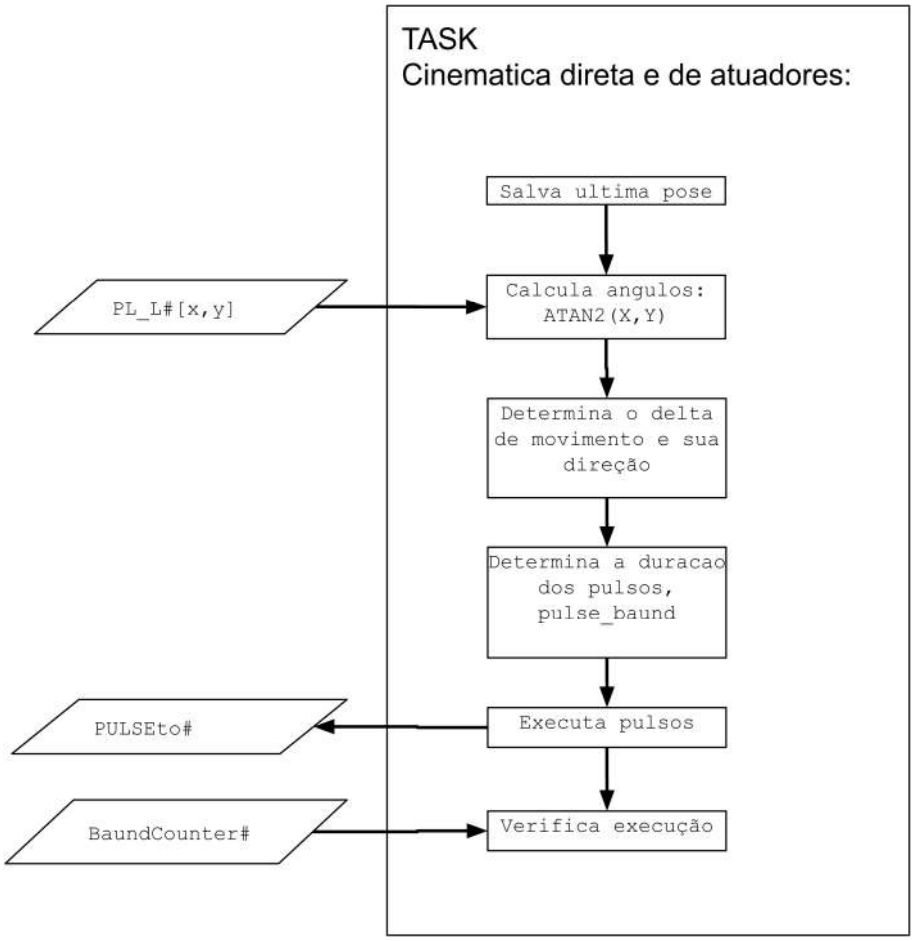

Figura 3.31. Fluxograma da solução analítica de cinemática direta

Geração do movimento traduzido

Sabendo-se a quantidade de movimento necessária para a movimentação de cada junta, pode-se realizar a geração de movimento executando um movimento síncrono das juntas. Ou seja, dentro de um período de tempo de interpolação serão gerados os pulsos referentes ao movimento de cada junta dentro do segmento de movimentação. Sendo que estes movimentos se iniciam e terminam juntos, cada qual com sua velocidade e direção específica.

Essa função tem também o objetivo ser prova de conceito do sistema de geração de pulsos controlados sempre a metade do ciclo. Ou seja, tempos iguais de nível alto e nível baixo do sinal digital. Essa funcionalidade atrela a frequência máxima de trabalho à velocidade do movimento. Isto elimina o parâmetro de largura de pulso do gerador de pulsos do gerador de trajetória.

Como o intuito do tradutor é, ser um componente atrelado ao robô e sua cinemática. Podendo assim, ser utilizado com qualquer comando de movimento com saída de setpoints de movimento em forma de pulso e direção. É importante que a frequência de saída do mesmo não precise ser ajustada com a troca do comando de movimento cartesiano.

Pode-se observar a função da interrupção temporal de alta prioridade responsável pela geração dos pulos de comando de movimento traduzido no fluxograma na figura3.32:

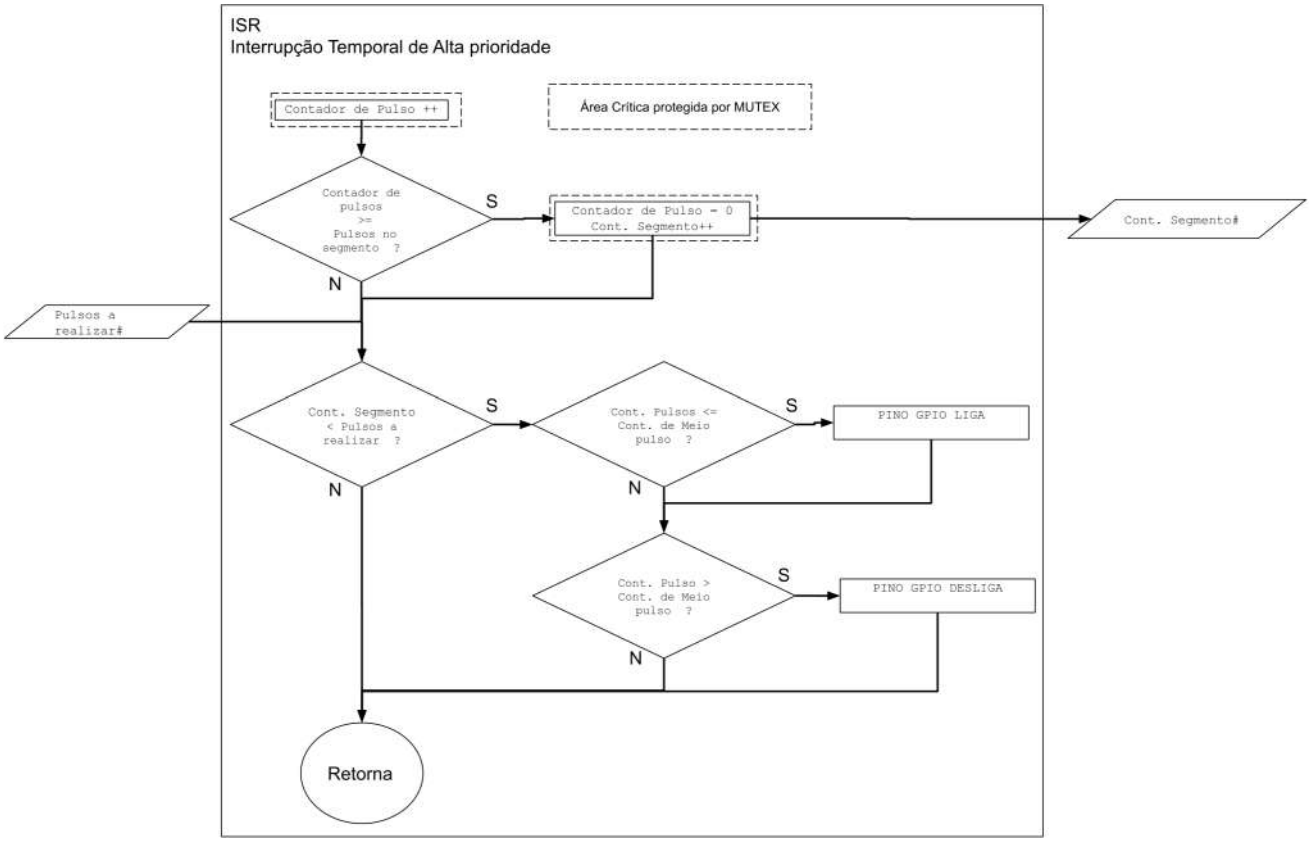

Figura 3.32. Gerador de pulsos de comando de movimento traduzido

3.4.5 Aquisição de sinais

Para se realizar a aquisição dos sinais de setpoint de entrada e saída do tradutor de trajetória será utilizado um analisador de lógica *Saleae Logic Analyzer* baseado na Arquitetura *FX2LP*, que utiliza o chip textitCypress CY7C68013 para realizar a aquisição de sinais discretos até 24Mhz em 8 canais e interface USB até 480Mbps.

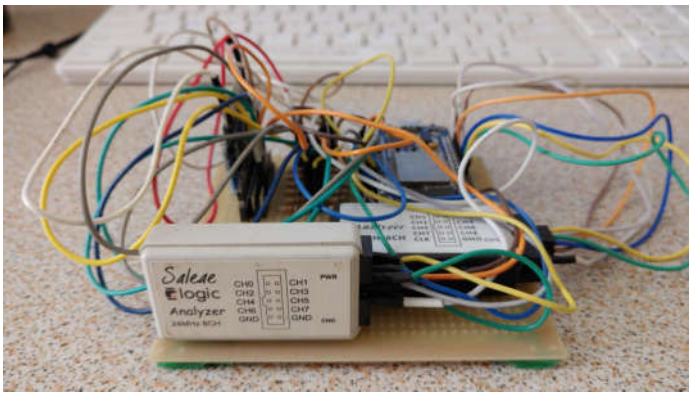

Figura 3.33. Dispositivo de aquisição de sinais

Para realizar o controle e consolidação da aquisição utiliza-se o Software *Sigrok Pulse View* que se auto define como um projeto que pleita a criação de um software aberto, portável e multiplataforma para análise de sinais, que suporta vários hardwares diferentes e sob a licença da *GNU GLP V3*.

4 Resultados

Neste capítulo são apresentadas as informações obtidas através dos dados consolidados das rodadas de experimentação no modelo simulado no *Octave* e do teste funcional do sistema embarcado real.

4.1 Experimentos no modelo simulado

Para análise dos dados do experimento simulado foi utilizado o software *Minitab*. Este foi responsável por calcular:

- Ajuste dos dos dados;
- Covariâncias dos fatores;
- Polinômio característico de resposta do erro em relação aos fatores de entrada;
- Índices de força de efeito isolado dos fatores no erro médio;
- Efeitos de interação entre os fatores de entrada;

4.1.1 Valores Ajustados

Pode-se observar os valores obtidos como saída do experimento ajustados no gráfico da Fig.4.1.

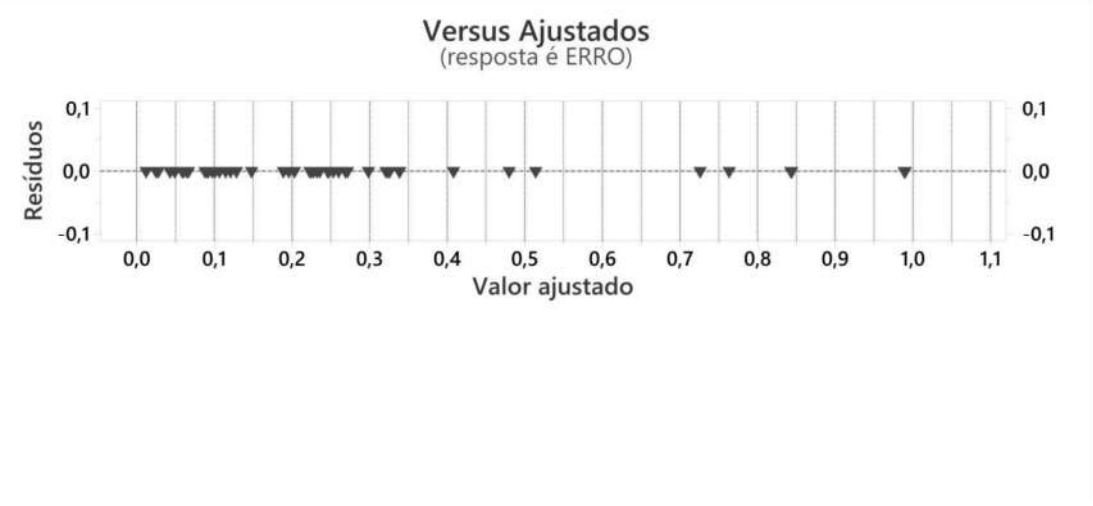

Figura 4.1. Valores ajustados de saída do experimento

Segundo o manual científico do [LCC, 2019] o no gráfico "Versus Ajustado"o eixo X representa como os dados estão distribuídos no range de medição, e o eixo Y representa como as mesmas medições de um ponto variam. Pontos muito separados dos outros em X são mais representativos do que pontos próximos para os cálculos de causa e efeito dos fatores e a variável analisada. Estes valores são ajustados entre o máximo e mínimo medidos. Os valores dos resíduos não variam no eixo "Y"pois o experimento foi realizado num ambiente simulado, logo os pontos onde os parâmetros de entrada se repetem os valores de saída também se repetem.

4.1.2 Polinômio característico

Polinômio característico da média de erro, equação de regressão em unidades não codificadas, obtida pelo software Minitab:

$$\begin{aligned} ERRO = {} & 0,2633 - 0,000120A - 0,000794B - 0,06495C \\ & +0,000893D - 0,000281E + 0,000239F + 0,000003AB \\ & +0,000107AC - 0,000001AD + 0,000329BC + 0,000001BD \\ & +0,000002BF + 0,00006CD + 0,000080CE - 0,000018CF \\ & -0,000001DE - 0,000001ABC - 0,000001BCF - 0,2259PtCt \end{aligned}$$

. (4.1)

Este polinômio representa a função ero em relação os fatores de entrada. Assim para experimentos futuros pode ser feita uma estimativa mais assertiva destes valores de entrada. Por consequência pode-se obter uma janela de dados mais refinada.

4.1.3 Efeito dos fatores no erro médio

Esta análise define quais variáveis demonstram-se ser relevantes na variação da média do erro, ou seja obtiveram valores de α maior do que 0.05 no gráfico de Pareto 4.2.

Pode-se observar que quase todos os fatores escolhidos têm um valor de α acima de 0.05, logo pode-se concluir que estes são de fato relevantes ao experimento. Ou seja, quando alterados causam efeitos relevantes na variável de saída. O único fator que não teve efeito relevante foi aposição inicial da trajetória fator C. Isto evidencia que a posição da trajetória dentro do espaço de trabalho não alterou significativamente o valor do erro de execução desta. OU seja, a solução matemática das cinemáticas demonstra-se robusta dentro do espaço de trabalho.

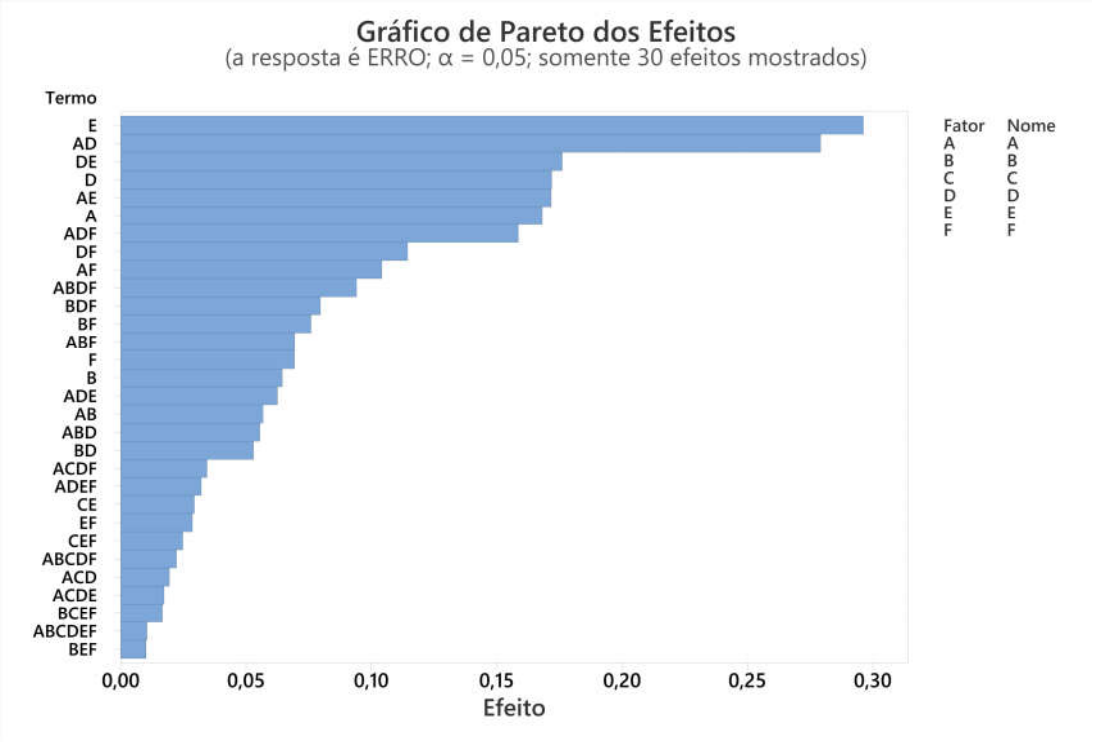

Figura 4.2. Gráfico de Pareto dos fatores do experimento

A Fig.4.3 é o gráfico de efeitos principais onde pode-se observar como cada variável de entrada influencia no erro médio da trajetória e qual a magnitude desta influência. Os pontos de "extremidade"indicam o máximo e mínimo do range de variação dos fatores de entrada do experimento; já o "centro"é o ponto médio deste range.

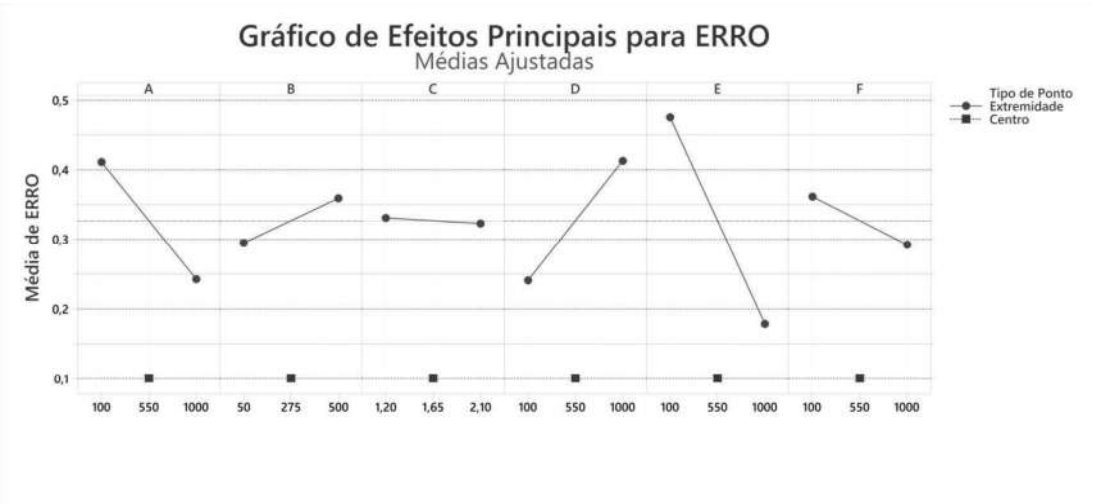

Figura 4.3. Efeitos dos fatores na média de erro

A análise da influência de cada fator de entrada pode então ser realizada:

- A - Frequência de geração de pulsos do gerador de setpoint de trajetória:

  Nota-se uma diminuição erro com uma maior frequência base de geração de pulsos, ou seja, uma maior resolução temporal contribui para um menor erro de movimento.

- B - Resolução espacial do gerador de setpoint de trajetória:

  Houve um aumento do erro médio com o aumento da resolução espacial, o que não era esperado. Isto indica que existe uma influência forte de outro fator deslocando esta resposta. Neste caso específico, a análise co-fatorial indicou uma influência grande do fator "A"sobre o fator "B"mostrada pela figura 4.4.

- C - Posição Inicial da trajetória:

  Dos fatores escolhidos esse foi o que apresentou a menor influência no erro, o que evidencia que o erro não se altera dentro do espaço de trabalho.

- D - Período da janela temporal de tradução

  Espera-se que quanto menor a janela temporal mais suave seria o movimento final, e o resultado corrobora com esta expectativa sendo que quanto menor a janela temporal menor o erro.

- E - Frequência de geração de pulsos da saída do tradutor de trajetória

  Assim como na geração de pulsos de setpoint de trajetória de movimento, a frequência máxima do gerador de pulsos de saída tem grande influência no erro final. Porém, a saída do tradutor de movimento tem uma maior influência no erro.

- F - Resolução espacial do gerador de pulsos de saída do tradutor de trajetória.

  A resolução especial pulsos/ângulo da saída do sistema de tradução demonstrou-se inversamente proporcional ao erro.

4.1.4 Efeitos de interação dos fatores

Na Fig.4.4 observa-se a análise co-fatorial da interação dos fatores de entrada agrupados dois a dois. Esta possibilita a avaliação se existe acoplamento entre este par de fatores. O a figura é uma matriz de gráficos de interação, onde o eixo "X"representa a média do erro e o eixo "Y"os pontos de extremidade e médios do range de entrada dos valores. O subtítulo de cada bloco está indica qual a interação em análise, assim "A x B"é a influência de "A"em "B"e assim por diante.

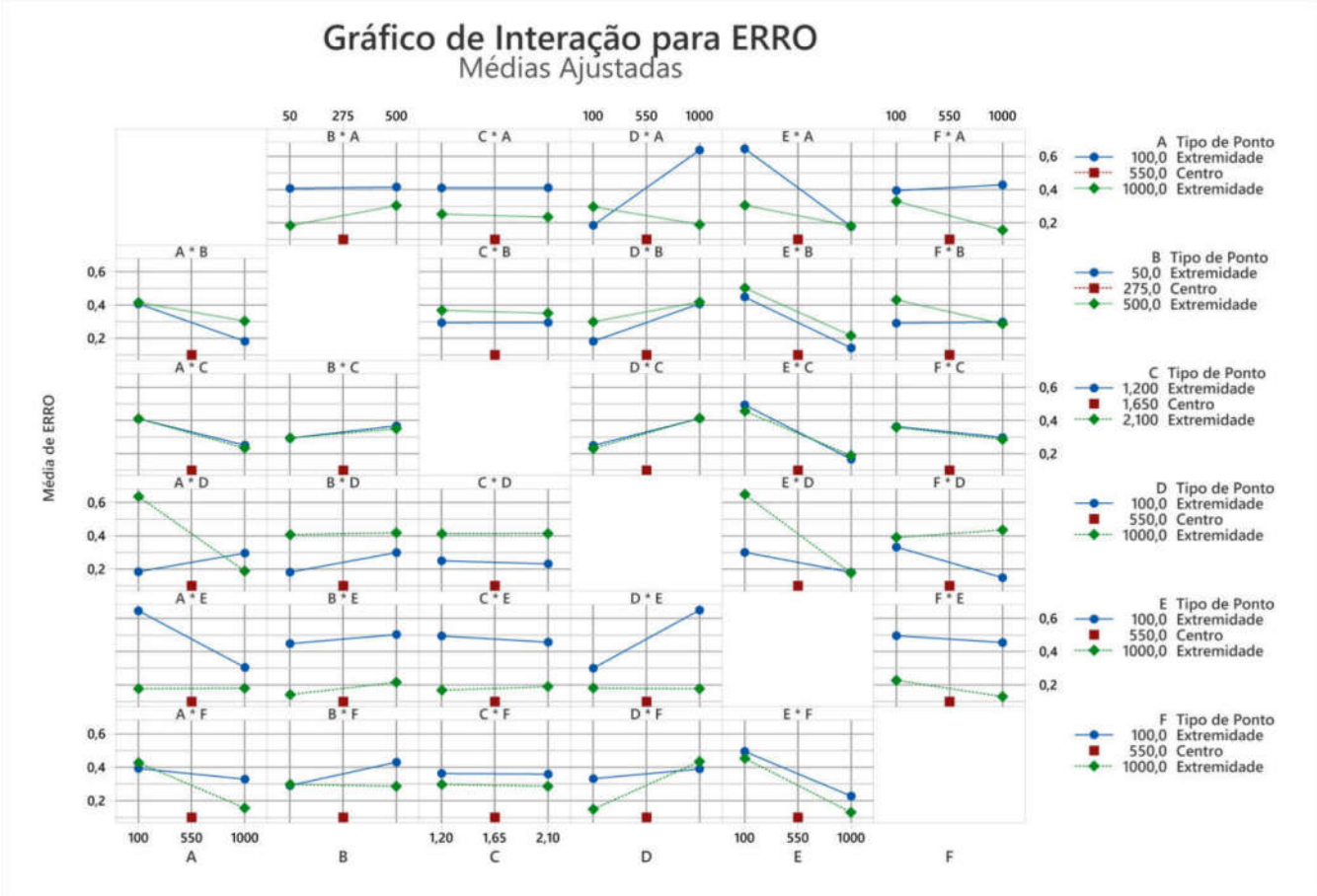

Figura 4.4. Interação entre os fatores

No geral quando as linhas de interação são paralelas os efeitos tendem a ser desacoplados, porém quando tem derivadas muito diferentes isso indica uma maior influência de um fator no outro. Como já descrito anteriormente, espera-se um acoplamento entre a os fatores A e B devido a forma como o sistema simulado foi formulado.

- A x B - Nota-se que valores maiores de A potencializam a influência de B na média de erro. A influência de A em B é forte, isto explica por que se observou uma curva de influência inesperada de B na Fig.4.3.

- A x C - Não se observa influência entre esses fatores.
- A x D - Pode-se observar que maiores valores de período da janela temporal, D, potencializam o efeito de A. O que já se esperava, pois, isso reduz a resolução da tradução.
- A x E - Como esperava-se a frequência máxima de geração de pulsos de saída, E, potencializa a influência de A quando tem seus valores mais baixos. Em valores mais altos, evidencia-se a possibilidade do *upscaling* entre a entrada e a saída do sistema, no que tange a frequência máxima de geração de pulsos. Pode-se notar que a influência de A quase não se altera quando E está no seu limite superior, reta verde do quadro A*E da Fig.4.4.
- A x F - Observa-se que no limite inferior da resolução espacial angular de saída, F, a influência de A é reduzida, e no limite superior o aumento de A causa uma notável redução do erro médio.
- B x C - Não se observa influência entre esses fatores.
- B x D - Nota-se uma pequena influência de D em B, elevando o erro associado quando o período da janela de tradução é muito curto. Provavelmente devido ao fato do período muito curto acaba por diminuir muito o quantum espacial a ser traduzido, levando os valore para mais próximo do limite mínimo de processamento.
- B x E - Não se observa influência entre esses fatores.
- B x F - Observa-se que a resolução espacial saída F tem sua influência aumentada quando a resolução espacial de entrada está no seu limite inferior.
- C x D - Não se observa influência entre esses fatores.
- C x E - Não se observa influência entre esses fatores.
- C X F - Não se observa influência entre esses fatores.
- D X E - Nota-se que valores menores de E potencializam a influência do Período da janela de tradução (D)
- D X F - Observa-se um leve aumento da influência de D em valores mais altos de E.
- E X F - Não se observa influência entre esses fatores.

4.2 Teste de Funcionalidade do sistema embarcado real

Na Fig.4.5 observa-se uma trajetória semelhante a que foi executada no teste. O objetivo da trajetória é de seguir uma linha reta unitária, porém com uma mudança de ângulo de orientação do efetuador de um radiano. Nota-se que na cinemática inversa e direta teóricas e sem as re-amostragens não são perceptíveis aberrações e desvios.

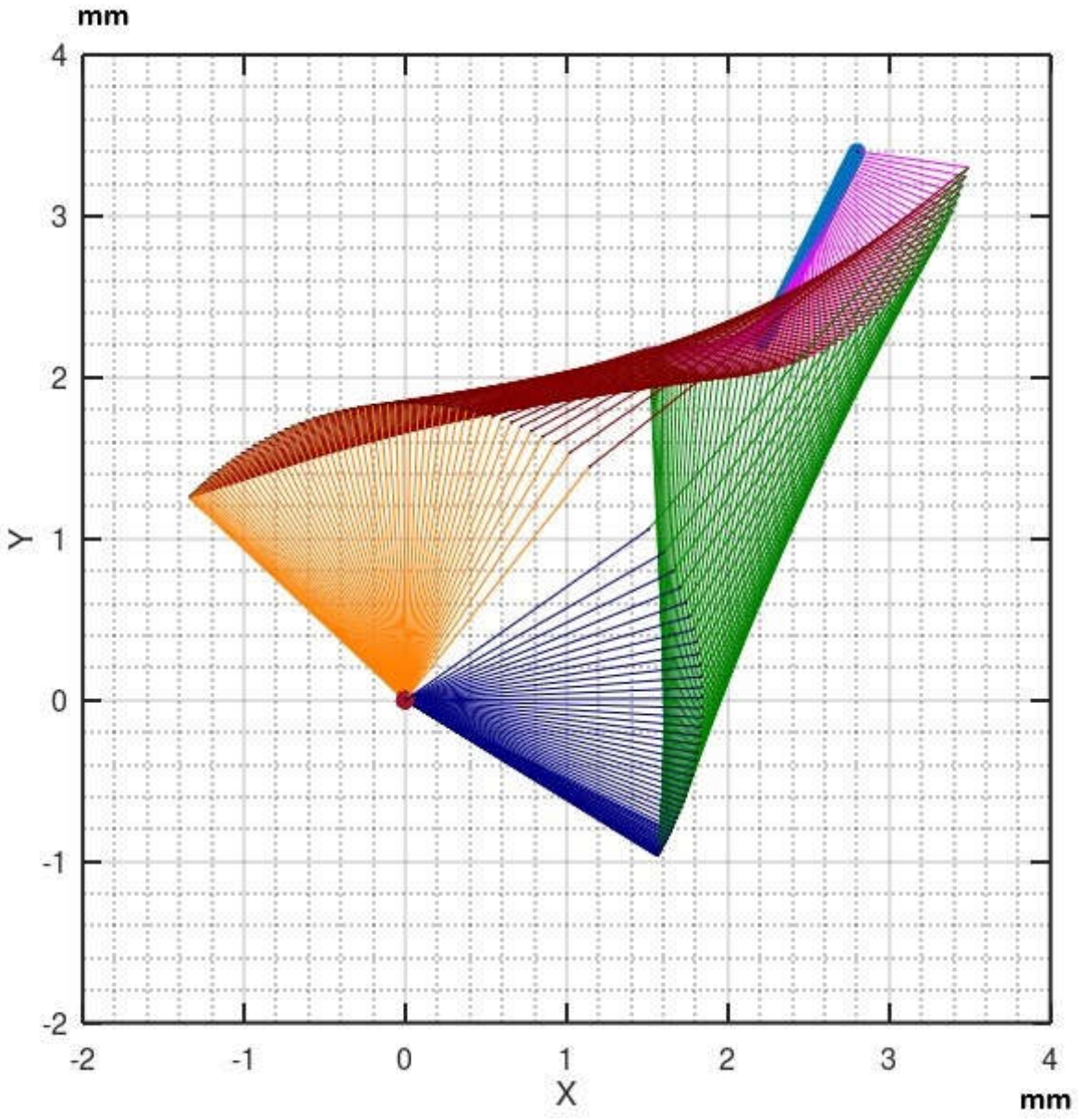

Figura 4.5. Trajetória exemplo para realização do teste funcional

A Fig.4.6 mostra a trajetória capturada pelo sistema de aquisição gerada pelo gerador de movimento cartesiano setpoint de entrada do tradutor. Na Fig.4.6 a seta grande em azul representa a trajetória programada. E em negro estão vetores representando a posição e orientação do efetuador.

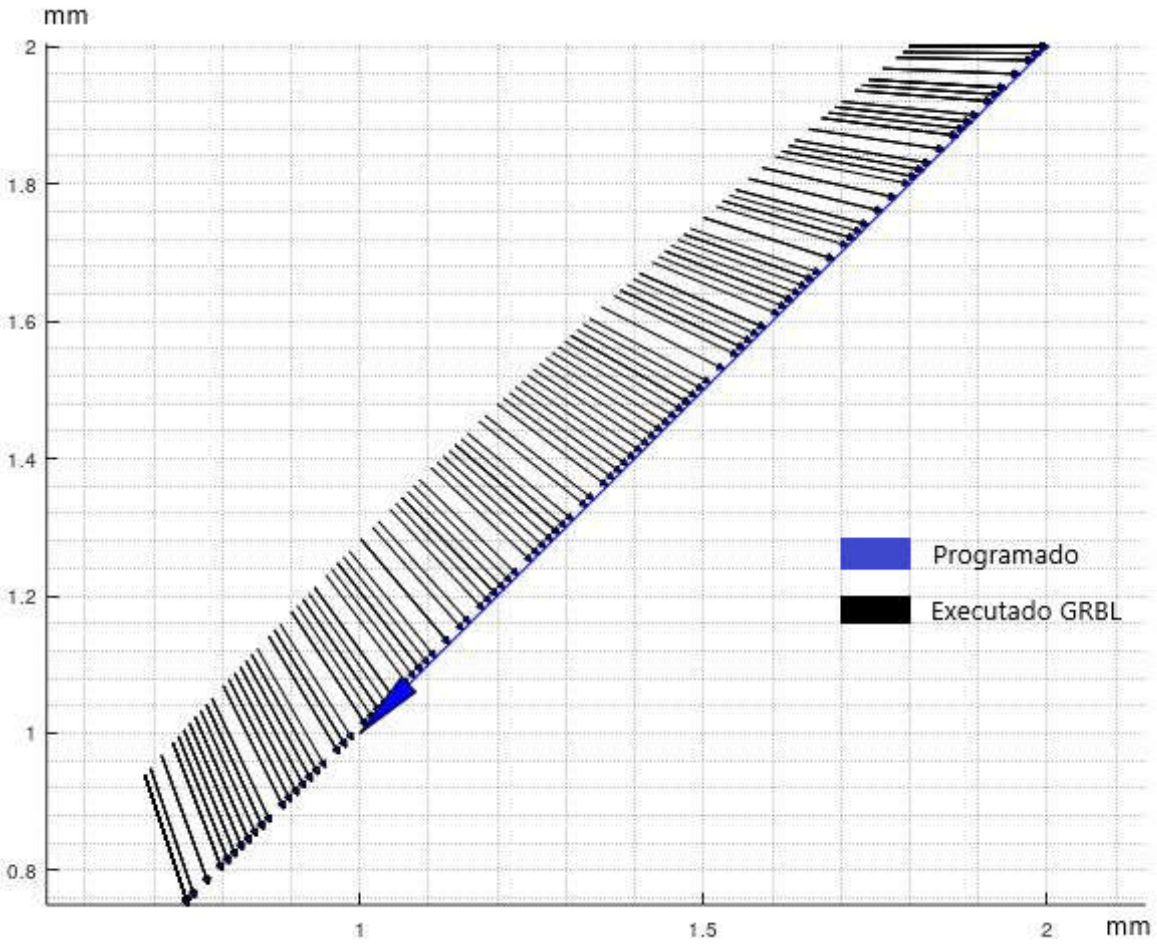

Figura 4.6. Trajetória setpoint do gerador de movimento cartesiano

Nota-se que a trajetória programada era originalmente unitária, porém o sistema GRBL executou uma reta com comprimento um pouco maior do que programado, isso devido a configuração de *ant-backslash* utilizada no gerador de movimento. O comprimento pequeno da trajetória foi escolhido para se observar melhor os erros de re-amostragens, pois estes são mais críticos em movimentos menores.

Para entender melhor o movimento discretizado pode-se observar no gráfico 4.7 o deslocamento por janela temporal de 10ms. A linha amarela e verde estão sobrepostas e são idênticas.

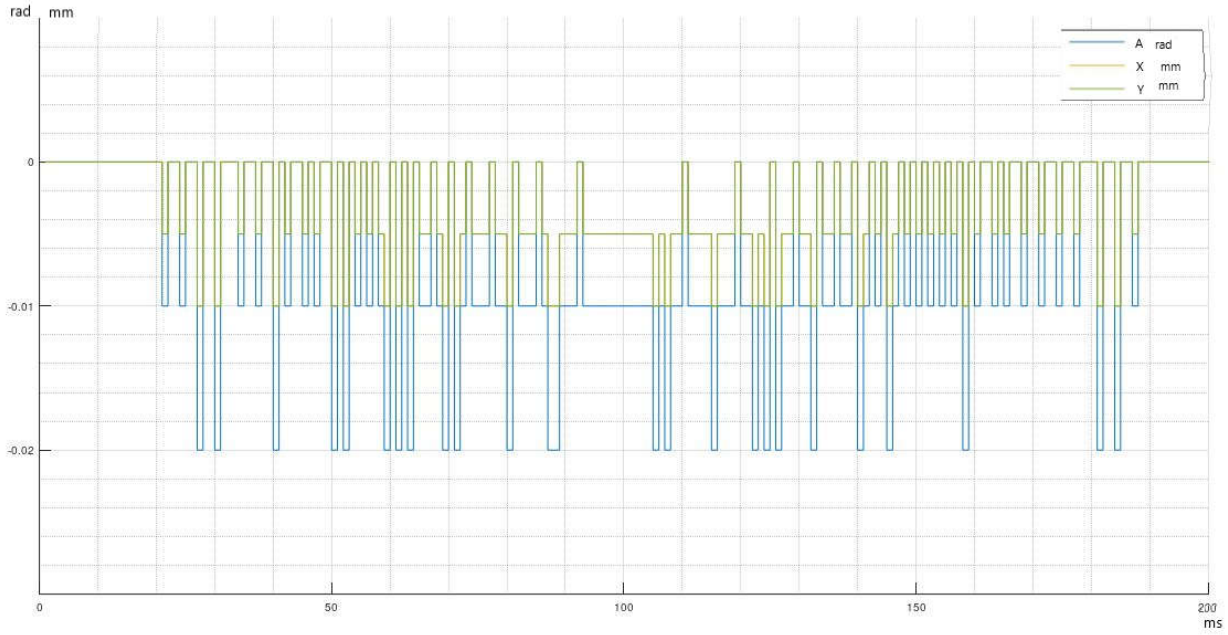

Figura 4.7. Pulsos no tempo gerados pelo setpoint do gerador de movimento cartesiano

Teoricamente as três linhas deveriam ser idênticas já que representam o mesmo número de pulsos. porém, pequenos defasamentos na geração de pulsos e interpolação acabam por deslocar alguns pulsos de uma janela de aquisição para a próxima, gerando assim dissimilaridades. Lembra-se que a janela de interpolação do GRBL é de 50ms.

No gráfico 4.8 observa-se a trajetória traduzida no sistema embarcado real, adquirida com o mesmo sistema e com as mesmas configurações, e pós processada pela mesma cinemática de juntas utilizada no modelo matemático do robô. Os vetores em vermelho representam a posição e orientação do efetuador.

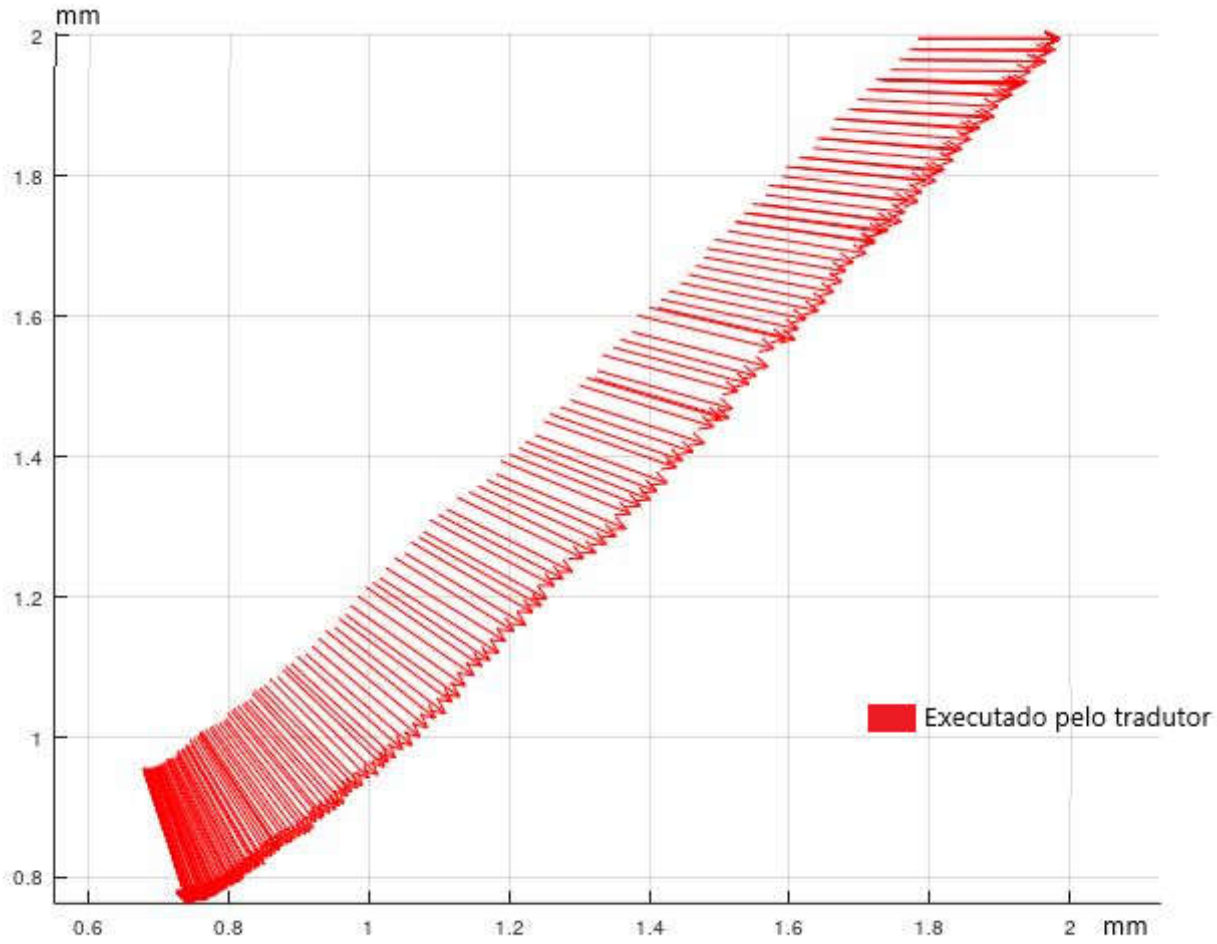

Figura 4.8. Trajetória executada pelo tradutor, obtida na saída do tradutor embarcado.

Pode-se notar no gráfico 4.9 o perfil de aceleração e desaceleração do movimento sendo mantido. A janela de interpolação do tradutor é de 1ms, ou seja, cinquenta vezes mais rápida do que a do gerador de setpoint.

Evidencia-se a suavização do movimento, ao notar-se que um menor número de pontos de deslocamento nulo são observados durante o movimento. Assim como, também pode ser notado um menor número de patamares de velocidade constante durante o processo de aceleração e desaceleração, isto em comparação com o perfil da Fig.4.7, como pode se ver melhor na Fig.4.10.

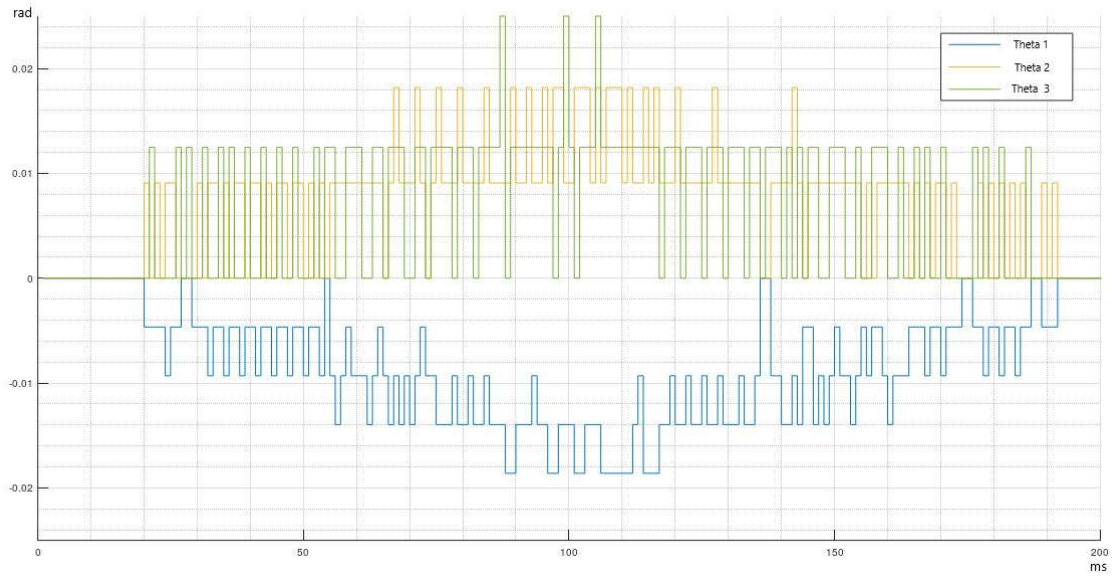

Figura 4.9. Pulsos no tempo gerados pelo tradutor

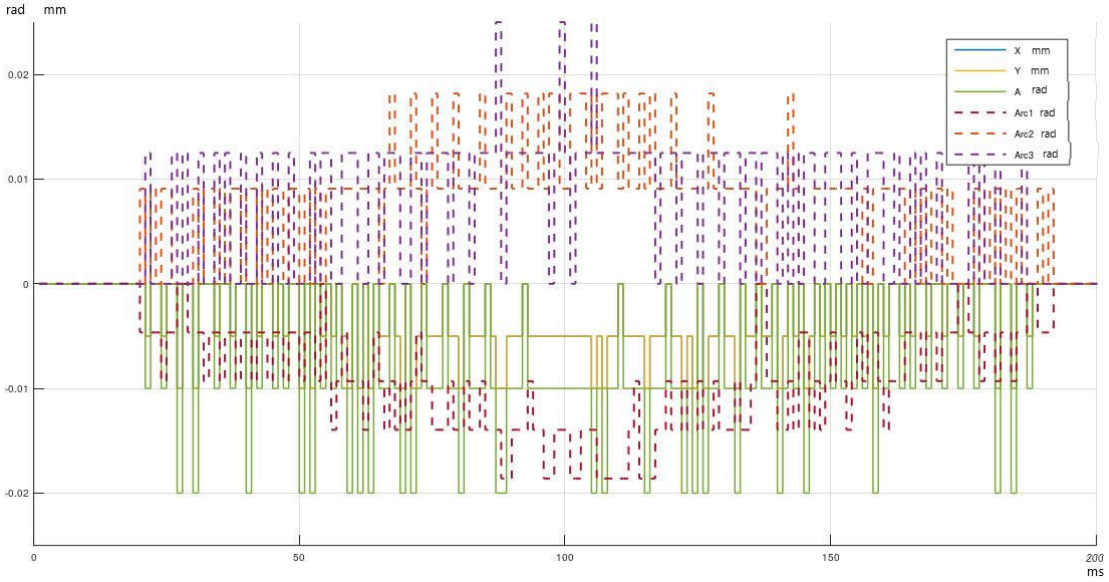

Figura 4.10. Comparativo do perfil de discretização de movimento

Observa-se na Fig 4.11 a comparação das trajetórias comandadas, executada pelo gerador de set-point cartesiano e a traduzida pelo sistema embarcado. Nota-se que este existe um erro entre as duas últimas, visualizado pela diferença entre os vetores pretos e vermelhos. Esse erro é chamado de aberrações de movimento, onde o ponto inicial e final da trajetória não apresentam diferenças, porém no meio da trajetória existem. Utilizando a distância entre os vetores de entrada e saída a cada instante calculou-se um erro médio de 0,02mm e um erro máximo de 0,05mm. A implementação realizada do

sistema embarcado previne erros acumulativos o que explica porque o erro ao final da trajetória é reduzido até ser nulo.

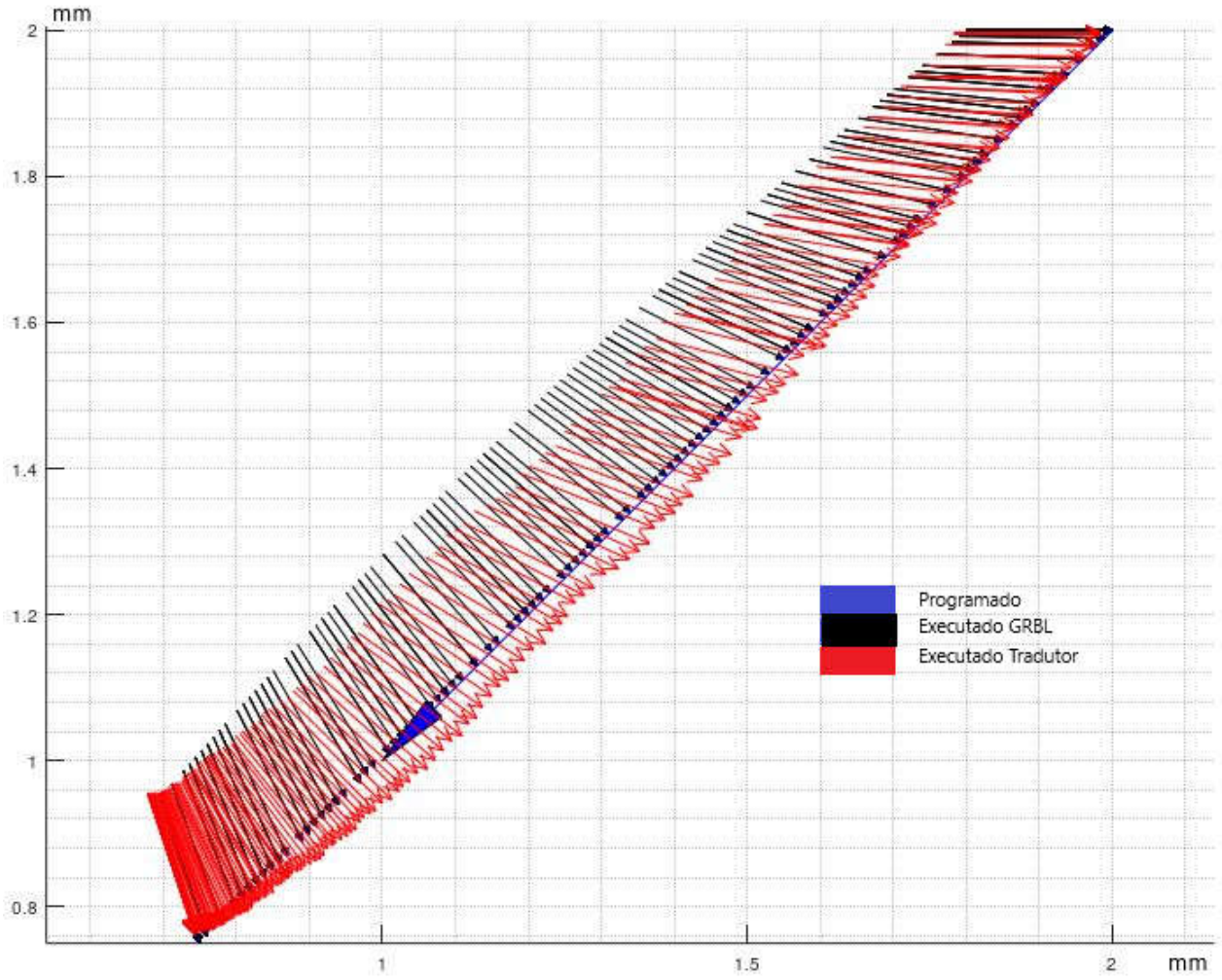

Figura 4.11. Comparativo de trajetória setpoint e executada

Pode-se notar na Fig. 4.12 que houve a ocorrência de um atraso de aproximadamente 20ms entre: o início da geração do setpoint de movimento cartesiano pelo GRBL, e o início de geração dos pulsos após a tradução da cinemática pelo ESP. Isso era previsto como possível de ocorrer devido a necessidade de se ter o mínimo de 2 pontos de movimento para se possa calcular a quantidade de movimento a ser percorrida dentro da janela temporal de processamento do tradutor.

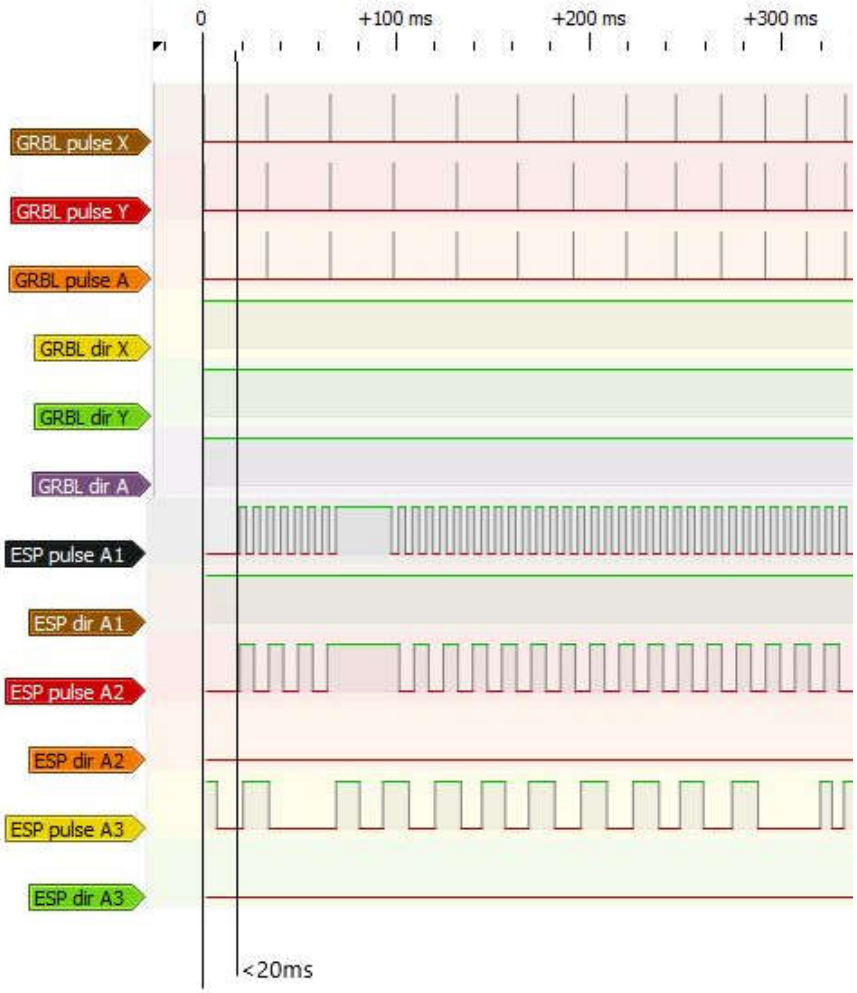

Figura 4.12. Comparação do tempo dos pulsos entre a entrada e saída do sistema embarcado.

5 Conclusão

O Objetivo principal do trabalho era desenvolver a tradução em tempo real da cinemática de um robô cartesiano possibilitando seu comando de movimento por um controle numérico cartesiano. Com os resultados obtidos conclui-se que a tradução foi realizada de forma satisfatória, com um erro médio de execução de 0,02mm para uma trajetória de um milímetro de comprimento. Este erro foi causado por uma falta de sincronia no gerador de set-point de saída do tradutor entre os três arcos, θ_1,θ_2 e θ_3 . O atraso de sincronismo foi menor do que 20ms, sendo que o período de interpolação do gerador de set-point cartesiano é de 50ms, porém, foi o bastante para gerar uma aberração no movimento durante a execução da trajetória. Ressalta-se que este erro não é acumulativo e não é relativo ao comprimento do movimento. Assim, movimentos mais extensos não tem maiores valores de erros absolutos e ao final do movimento esse erro é nulo. Porém, em movimentos com maiores velocidades relativas à orientação da ferramenta, porém causar maiores aberrações durante a trajetória. Isto deve ser corrigido aplicando-se uma diretiva rígida de sincronismo entre os três ângulos.

Quanto ao objetivo específico de se escolher um robô e desenvolver a solução de suas cinemáticas, conclui-se que este foi atingido. O robô escolhido, pantógrafo de quadro barras com extensão serial de orientação da ferramenta e atuação concêntrica, demonstrou-se uma plataforma de estudo sólida, com uma literatura abrangente e muitos estudos de casos validados. Robôs planares com duas juntas rotativas seriais têm a condição desfavorável de possuírem duas configurações possíveis para cada posição da ferramenta. A utilização da geometria do tipo pantográfica de quatro barras remove essa ambiguidade. Isto facilita o planejamento de trajetória pois, não é necessário a aplicação de diretivas adicionais para a escolha da configuração desejada dentro de várias soluções possíveis. A atuação concêntrica permitida por esse robô tem a vantagem de excluir a necessidade de uma unta atuada no segundo link do manipulador reduzindo a complexibilidade mecânica e a inércia do mesmo. As soluções das cinemáticas envolvidas na geometria em questão podem ser obtidas por vários métodos, inclusive pela junção destes como foi executada neste trabalho. Isto possibilita traçar paralelos entre os métodos para verificação da eficácia da solução já nas primeiras etapas do desenvolvimento. Foi observado que as singularidades e similaridades encontradas na geometria e nas soluções matemáticas são facilmente visualizadas, facilitando muito o estudo de sua significância matemática e mecânica. A forma analítica das soluções das cinemáticas encontradas e com equações de no máximo segunda ordem, agilizou o estudo e a implementação das mesmas tanto na plataforma simulada como no sistema real embarcado.

A plataforma simulada foi desenvolvida utilizando o GNU Octave que se demonstrou uma ferramenta capaz de executar a modelagem e com a vantagem de ser gratuito. Por ser uma ferramenta aberta e com amplo suporte pela comunidade de usuários as dificuldades encontradas foram resolvidas de forma facilitada. A simplicidade, pouca exigência de recursos e flexibilidade encontradas nesta plataforma evidenciam que ela possa ser utilizada como base para uma plataforma didática mais geral em trabalhos futuros. Unindo-se o modelo matemático simulado, a metodologia DoE e a análise multifatorial forma-se uma ferramenta muito útil para o estudo das interações entre parâmetros do sistema de comando de movimento de robôs. Isto, porque, este sistema é inerentemente multivariável o que dificulta o isolamento e observação destas influências. Sua utilização foi fundamental para que existisse uma base prévia dos ajustes dos parâmetros a serem utilizados no sistema embarcado real.

Conclui-se também que o *SoC ESP32-wroom-32* é uma ferramenta capaz de realizar as etapas necessárias para a tradução, inclusive com o *upscaling* da janela temporal de interpolação. Assim o resultado obtido com a geração de uma trajetória real com curvas de aceleração foi satisfatório como observa-se na Fig.4.8. Isto evidência que a tradução não altera de forma significativa a dinâmica do movimento, desde que esse movimento não atinja limites de velocidade, aceleração e posição do arranjo dos atuadores na cinemática final. A singularidade artificial encontrada na plataforma simulada não foi observada no sistema embarcado real. Isto porque por mais similares que sejam as funções utilizadas na plataforma simulada e real, o SoC é um sistema com processamento paralelo e por isso tem características que não foram implementadas no modelo simulado.

A complexidade matemática envolvida na solução dos modelos cinemáticos propostos é simples o bastante para que essas possam ser implementadas em sistemas de controle embarcados atuais, o que evita a utilização de soluções numéricas limitadas. Logo, comparando com os trabalhos anteriores, não foi necessário o desenvolvimento de um pós-processador específico para converter o Gcode da trajetória desejada para um código que realiza o movimento necessário pela cinemática direta, o que impede a programação online do sistema robótico, como proposto por [Bomfim et al., 2012]. Assim o Gcode da trajetória cartesiana planejada pode ser utilizado diretamente no controlador de movimento, ou seja, um mesmo programa utilizado por um robô cartesiano poderia ser utilizado para controlar o robô híbrido em questão.

Era desejável que a trajetória testada fosse mais extensa e complexa. porém, é necessário que todo o tempo de excussão seja amostrado numa frequência acima do dobro da frequência máxima de geração de pulsos. Isto porque, o sinal gerado não é periódico, e no caso do GRBL ainda tem uma largura fixa de pulso e com uma frequência mais alta ainda do que a frequência máxima de geração de pulsos. Somando ao fato, que apesar

de muito eficaz, a análise multifatorial exige um número de experimento relativamente alto. O resultado é que o tempo de de excussão e o volume de dados gerados para as simulações e testes tornaram-se grandes o bastante para limitar que fossem efetuados testes mais extensos dentro do tempo deste trabalho.

Considerando que para uma trajetória curtas (um mm), os erros provenientes das sucessivas: amostragem, reconstrução (ZOH) e re-discretização tendem a ser mais perceptíveis. Isto pois, a escala do movimento está próxima da escala da resolução de comando. Se tem que desvios e aberrações observados foram pequenos quando os parâmetros estavam dentro de uma janela de trabalho ótima. De qualquer forma esses erros não são acumulativos, e tendem a ter amplitude fixa. E em trajetórias maiores a relação entre movimento e erro ficaria ainda menor, durante a execução. E em relação ao ponto inicial e ao ponto final da trajetória não houve erro.

6 Discussão e propostas de trabalhos futuros

Neste capítulo coloca-se as considerações sobre pontos de interesse levantados no trabalho, e a relevância destes para a continuidade do estudo, propondo trabalhos futuros sobre estes.

6.1 Robô como plataforma didática:

A geometria robótica escolhida demonstrou-se ser baseada numa plataforma de estudo madura, com ampla literatura desenvolvida e vários estudos de casos de fácil aplicação, tanto para robôs seriais como paralelos. Isto corrobora com uma aplicação didática onde estudantes podem acessar as informações de várias fontes e diferentes pontos de vista. Robôs seriais de dois eixos rotacionais tem como inconveniente ter duas soluções possíveis para cada ponto alvo. Uma positiva e outra negativa, como pode ser observado nas figuras: 6.1, 6.2, 6.3. A utilização do robô pantógrafo de 4 barras retira essa ambiguidade, sendo uma solução aplicada a um par de links e a outra para o outro par de links, assim uma completando a cadeia cinemática fechada. Demonstrou-se possível a solução da cinemática inversa, por métodos de cinemática seriais (D-H), híbridos (MDH, onde pode-se incluir cadeias abertas, fechadas e paralelas ao método D-H) e por métodos de solução cinemática geométrica. Indica-se assim a possibilidade de se fazer um paralelo entre os métodos, facilitando a compreensão destes e suas aplicações. Evidencia-se que a solução analítica baseada em intercessão de círculos, e provada pelo método MDH, aplicada a geometrias que formam 4 barras, ou seja, configurando um pantógrafo, possa ser mais geral do que o esperado. Pode-se então especular que seja uma classe de robôs de geometria híbrida: Seriais com cadeias fechadas, ou cadeias fechadas em série; E com uma solução de cinemática inversa geral em comum. porém, a prova desta hipótese fica como proposta para trabalhos futuros. Como exemplo, observa-se os robôs nas figuras 6.1, 6.2, 6.3 e 6.4 que são robôs de quatro barras formando um pantógrafo em várias configurações.

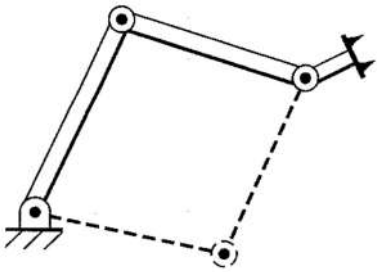

Figura 6.1. As duas soluções formando um pantógrafo de 4 barras [Craig, 2009]

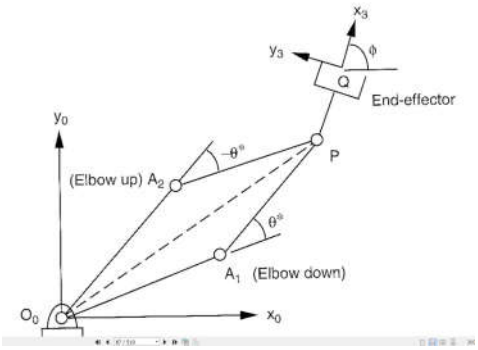

Figura 6.2. As duas soluções formando um pantógrafo de 4 barras [Tsai, 1999]

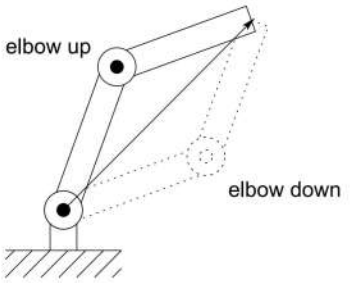

Figura 6.3. As duas soluções formando um pantógrafo de 4 barras [Spong et al., 2006]

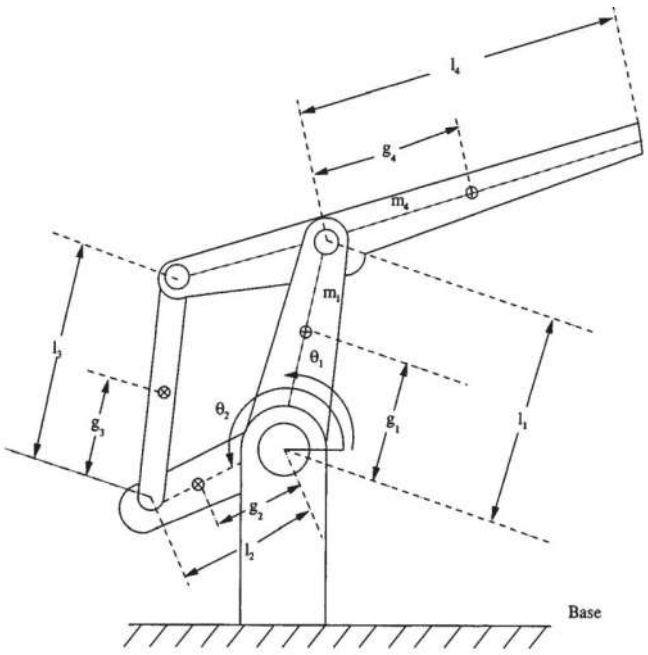

Figura 6.4. Exemplo de robô baseado em pantografo com revolução [Lewis et al., 2003]

6.2 Sistema Computacional como plataforma didática.

O sistema escolhido demonstrou-se adequado ao projeto e não foram encontradas grandes barreiras para realizar a programação embarcada. A utilização de plataformas integradas de desenvolvimento de código aberto (Arduíno IDE, Sigrock, Octave) facilitou a resolução do problema proposto. De forma que os recursos envolvidos puderam ser direcionados para o problema em si: re-amostragem de posição, tradução de cinemática e re-geração de movimento; ao invés da resolução de problemas relacionados a deficiências de bibliotecas, plataformas computacionais e de formação acadêmica na área de programação.

Espera-se que essa facilidade e o baixo custo da plataforma desenvolvida, assim como a ausência de necessidade de licenças de software, e a escolha de uma plataforma com ampla utilização da comunidade *Maker*, possam possibilitar a sua utilização para o ensino de robótica. Para isto a proposta de trabalhos futuros é o desenvolvimento de uma interface de configuração amigável para o sistema, facilitando: A configuração de parâmetros de uma solução cinemática; ou até mesmo uma construção modular baseadas em círculos para solucionar a cinemática de diversas geometrias.

6.3 Re-amostragem e *upscaling.*

O sistema embarcado utilizado foi capaz de realizar um *upscaling*, aumento aparente de resolução. Sendo que a janela de interpolação do GRBL é de 50ms e a janela de interpolação utilizada no experimento foi de 1ms, foi possível amostrar não só de forma eficiente a posição alvo setpoint do gerador de movimento do GRBL; como também foram capturadas as nuances de variação de velocidade e aceleração do movimento.

Assim o movimento traduzido manteve um perfil de velocidade e aceleração semelhante ao original. Evidenciando-se que para o tipo de tradução aplicada, onde se reintegra a posição alvo a cada intervalo de interpolação do tradutor, a velocidade e aceleração no plano cartesiano não são alteradas. porém, essas são modificadas somente no plano de juntas e atuadores.

Esse torna-se um fator importante para aplicação desse sistema junto a controladores de movimento com limitação de frequência máxima de saída, e curvas de aceleração do tipo trapezoidal. Possibilitando então que seja feita uma multiplicação real desta frequência máxima. Gerando assim, uma suavização de movimento final com a utilização de drivers de controle de movimento com maior resolução.

Para cinemáticas que utilizam juntas rotativas isso se torna um fator crítico, pois, existe normalmente uma amplificação entre a resolução do movimento do atuador e a extremidade do link. Atuadores de baixa resolução acabam por causar movimentos bruscos e oscilações indesejadas. Ou utilizam grandes reduções causando perda de velocidade

máxima e folgas mecânicas no robô.

Uma possibilidade que fica em aberta a ser desenvolvida em trabalhos futuros é a utilização do sistema para resolver paralelismos que possam ser difíceis de se resolver de forma mecânica. Assim realizando a expansão do número de eixos ativos, porém não diretamente comandados. Sendo uma ferramenta interessante caso deseje-se utilizar um comando holonômico para se posicionar uma ferramenta não holonômica, por exemplo: uma faca que só corte numa orientação específica; ou uma tocha de solda com um ângulo de solda fixo em relação a direção do movimento; ou até mesmo um robô manipulador de *pallets* que necessita de "nivelar"sua ferramenta em relação ao piso.

6.4 Plataforma totalmente simulada como ferramenta de otimização.

A simulação do robô em ambiente de programação matemática (*GNU Octave*) possibilitou desenvolver toda metodologia matemática da tradução antes de se começar a desenvolver na plataforma computacional embarcada. Assim ao passar para essa etapa já se sabia da eficácia da solução matemática da cinemática inversa, direta e de atuadores.

O Software |*GNU Octave* possuí as ferramentas necessárias para se fazer a modelagem do sistema discreto de comando. E assim foi possível entender mais a fundo as limitações dessa discretização ao se simular totalmente o sistema, tendo o controle de todas as variáveis envolvidas antes de se passar para a montagem real que é sujeita a variações e a respostas muitas vezes não documentadas pelos fabricantes desenvolvedores de bibliotecas.

Outro ponto importante do sistema simulado foi a facilidade de se reproduzir e diferenciar respostas a ajustes realizados. Possibilitando que os parâmetros de controle fossem alterados de forma isolada sem os distúrbios aleatórios normais vistos em sistemas reais. Assim podendo-se realizar um número de testes muito maior e em menos tempo, acelerando a delimitação dos limites do sistema e seus parâmetros ótimos.

6.5 Análise fatorial e de covariância como ferramentas de validação e otimização.

A análise co-fatorial de variância foi fundamental para se entender melhor a influência dos parâmetros do sistema na resposta do mesmo. Assim como a influência dos parâmetros entre si, além de indicar onde possivelmente onde é provável o sistema operar de forma mais satisfatória. Não foram realizadas múltiplas interações de otimização utilizando esta metodologia, mas, evidenciou-se que esta seja eficaz para essa tarefa. Por exemplo: Realizando-se experimentos adicionais com um número maior de rodadas, e sempre reduzindo os limites dos parâmetros de entrada em busca de pontos ótimos. porém um

estudo adicional das técnicas de otimização é necessária para a implementação de uma metodologia robusta para esse fim, o que foge do escopo deste trabalho, mas é uma boa proposta para trabalhos futuros.

Referências Bibliográficas

[All, 2014] (2014). *DMOS Microstepping Driver with Translator And Overcurrent Protection*. Allegro MicroSystems,. Rev. 5.

[Tos, 2016] (2016). *TOSHIBA BiCD Integrated Circuit Silicon Monolithic*. Allegro MicroSystems,. Rev. 1.

[Alvares et al., 2018] Alvares, A. J.; Toquica, J. S.; Lima, E. J. & Bomfim, M. H. (2018). Retrofitting of the irb6-s2 robotic manipulator using computer numerical control-based controllers. *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, 40(3):149.

[Arora, 2016] Arora, M. (2016). *Embedded System Design: Introduction to SoC System Architecture*. Learning Bytes Publishing.

[Bomfim et al., 2012] Bomfim, M. H. S.; Gontijo, R.; Bracarense, A. Q. & Lima II, E. J. (2012). Overhauling of a asea robot ir6 with open architecture. Em *2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications*, pp. 482--489. IEEE.

[Briot et al., 2015] Briot, S.; Khalil, W. et al. (2015). *Dynamics of parallel robots*. Springer.

[Cicolani, 2018] Cicolani, J. (2018). *Beginning Robotics with Raspberry Pi and Arduino: Using Python and OpenCV*. Apress.

[Coleman & Montgomery, 1993] Coleman, D. E. & Montgomery, D. C. (1993). A systematic approach to planning for a designed industrial experiment. *Technometrics*, 35(1):1--12.

[Craig, 2009] Craig, J. J. (2009). *Introduction to robotics: mechanics and control, 3/E*. Pearson Education India.

[De Lima et al., 2010] De Lima, A.; Poubel, L.; Lizarralde, F.; Leite, A. & Gleizer, G. (2010). Atualização de harware e software de um robô industrial. Em *Congresso Brasileiro de Automática*, volume 18, pp. 4403--4410.

[Dubey, 2008] Dubey, R. (2008). *Introduction to embedded system design using field programmable gate arrays*. Springer Science & Business Media.

[Enescu & Alexandru, 2014] Enescu, M. & Alexandru, C. (2014). Optimal design of the control system for an industrial robot using doe technique and regression model. *Applied Mechanics and Materials*, 658(1):626–631.

[Espressif, 2019] Espressif, S. (2019). Esp32 technical reference manual,

. *www.espressif.com*.

[Hartenberg & Denavit, 1955] Hartenberg, R. S. & Denavit, J. (1955). A kinematic notation for lower pair mechanisms based on matrices. *Journal of applied mechanics*, 77(2):215--221.

[Instruments, 2014] Instruments, T. (2014). Drv8825 stepper motor controller ic. *contrôleur de moteur pas-àpas DRV8825*.

[Jazar, 2010] Jazar, R. N. (2010). *Theory of applied robotics: kinematics, dynamics, and control*. Springer Science & Business Media.

[JF et al., 2009] JF, C.; KM, K.; MHS, B.; AJA, S.; FA, R. F.; EJ, L. I. & AQ, B. (2009). Development of control cabinet for industrial robots up to six degrees of freedom. Em *20 th International Congress of Mechanical Engineering, Gramado, Rio Grande Do Sul, Brazil*.

[Kienitz, 2012] Kienitz, KM & Ramalho F, F. . C. J. . N. R. . L. I. E. (2012). Development of a low cost integrated control system based on the fpga technology. *ABCM Symposium Series in Mechatronics*, Section VII - Robotics:1015–1022.

[Kienitz et al.,] Kienitz, K. M.; Lima II, E. J.; Gerais, M. & Brasil, G. Soluçao de cinemática hıbrida para robô pantógrafo de três graus de liberdade.

[Kumar et al., 1999] Kumar, V.; Zefran, M. & Ostrowski, J. (1999). Motion planning and control of robots. Em *Handbook of industrial robotics*, pp. 295--315. Wiley.

[Lages et al., 2003] Lages, W. F.; Henriques, R. V. B. & Bracarense, A. (2003). Arquitetura aberta para retrofitting de robôs. Em *Manet Notes Workshop*.

[LCC, 2019] LCC, M. (2019). *Interpretar os principais resultados para Gráfico de linha ajustada*. https://support.minitab.com/pt-br/minitab/18/help-and-how-to/modeling-statistics/regression/how-to/fitted-line-plot/interpret-the-results/key-results/.

[Le & Jeon, 2007] Le, N. Q. & Jeon, J. W. (2007). An open-loop stepper motor driver based on fpga. Em *2007 International Conference on Control, Automation and Systems*, pp. 1322--1326. IEEE.

[Lewis et al., 2003] Lewis, F. L.; Dawson, D. M. & Abdallah, C. T. (2003). *Robot manipulator control: theory and practice*. CRC Press.

[Lima et al., 2020] Lima, E. J.; de Matos Souza, H. A.; Kienitz, K. M. & Ramalho Filho, F. A. (2020). Closed-form solution for direct and inverse kinematic models of a parallel manipulator for robotic orbital welding. *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, 42(1):1--14.

[Lima II et al., 2004] Lima II, E. J.; Torres, G. C. F.; Castro, C. A. C.; Bracarense, A. Q.; Henriques, R. V. B. & Lages, W. F. (2004). Sensoring for retrofitting of an industrial robot. *IFAC Proceedings Volumes*, 37(4):545--550.

[Maassen, 2019] Maassen, U. A. (2019). *SerialComCNC Manual + Referenz*. Mönchengladbach, www.serialcominstruments.com.

[Merlet, 2006] Merlet, J.-P. (2006). *Parallel robots*, volume 128. Springer Science & Business Media.

[Munasinghe & Nakamura, 2006] Munasinghe, S. & Nakamura, M. (2006). *Trajectory planning and control of industrial robot manipulators*. INTECH Open Access Publisher.

[Ramalho Filho et al., 2010] Ramalho Filho, F. A.; Bracarense, A. Q.; Lima II, E. J.; Kienitz, K. M. & Ribeiro, E. B. (2010). Development of robots for the pipeline industry. Em *ISR 2010 (41st International Symposium on Robotics) and ROBOTIK 2010 (6th German Conference on Robotics)*, pp. 1--6. VDE.

[Shiller & Dubowsky, 1991] Shiller, Z. & Dubowsky, S. (1991). On computing the global time-optimal motions of robotic manipulators in the presence of obstacles. *IEEE Transactions on Robotics and Automation*, 7(6):785--797.

[Spong et al., 2006] Spong, M. W.; Hutchinson, S.; Vidyasagar, M. et al. (2006). *Robot modeling and control*.

[Spur & Uhlmann, 2005] Spur, G. & Uhlmann, E. (2005). Industrieroboter. Em *Dubbel*, pp. T106--T112. Springer.

[Tahara S., 2014] Tahara S., R. H. (2014). Planejamento de experimentos (doe).

[Takahashi & Goetz, 2004] Takahashi, T. & Goetz, J. (2004). Implementation of complete ac servo control in a low cost fpga and subsequent assp conversion. Em *Nineteenth Annual IEEE Applied Power Electronics Conference and Exposition, 2004. APEC'04.*, volume 1, pp. 565--570. IEEE.

[Tsai, 1999] Tsai, L.-W. (1999). *Robot analysis: the mechanics of serial and parallel manipulators*. John Wiley & Sons.

[Wüst, 2018] Wüst, K. (2018). *Grundlagen der Robotik*. https://homepages.thm.de/ hg6458/Robotik/Robotik.pdf.

A Definições de Robótica Aplicadas a este trabalho:

Espaço cartesiano, operacional ou de tarefa:

O espaço cartesiano, também designado como espaço operacional ou espaço de tarefa é o espaço de referência de ferramenta, ou seja, espaço cartesiano global em que a ferramenta está inserida. Tem como ponto de origem normalmente um ponto de referência do trabalho a ser executado pelo robô, ou um ponto de referência na própria base do robô. Também pode ser chamado de espaço de tarefas ou espaço operacional ([Craig, 2009], P4)([Spong et al., 2006], P23). Por exemplo, pode-se indicar posições de interesse partindo da origem na base do robô, com vetores: Descrevendo a posição da primeira junta; A posição do alvo do objeto a ser manipulado; ou mesmo a posição e orientação da ferramenta na extremidade do manipulador como colocado por([Craig, 2009]). É o espaço de referência onde a cinemática inversa é calculada. É neste espaço de coordenadas onde os pontos de interesse do sistema são descritos de forma normalizada para que o entendimento de seu posicionamento possa ser feito. Nota-se na Fig.A.1 os pontos: de origem(O), da posição da ferramenta(t), da base do robô(b), das juntas (j1 e j2) e do objeto a ser manipulado(p).

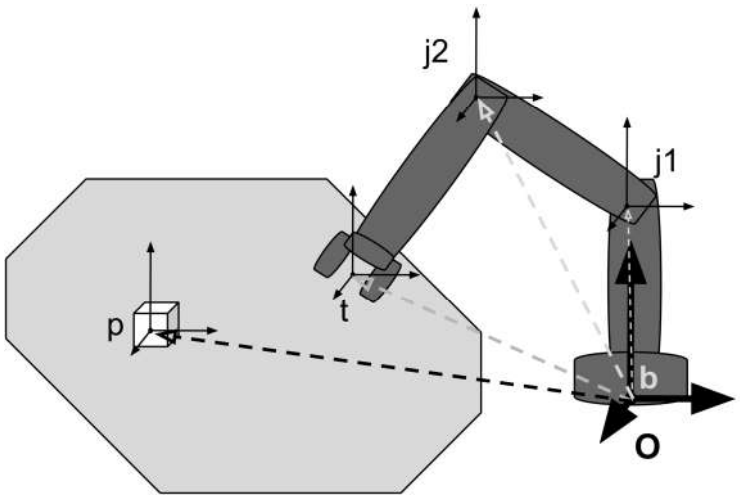

Figura A.1. Espaço cartesiano

Espaço de Juntas:

O espaço de juntas é a referência dos parâmetros de posição e orientação (vetores de junta) das juntas do robô, geralmente tem como referência a origem do plano cartesiano como descrito por [Craig, 2009]. Ou seja, pode-se definir a posição da próxima junta em relação a junta atual, sendo fundamental para o cálculo da cinemática direta. Pode-se ver essa representação nos vetores da Fig.A.2.

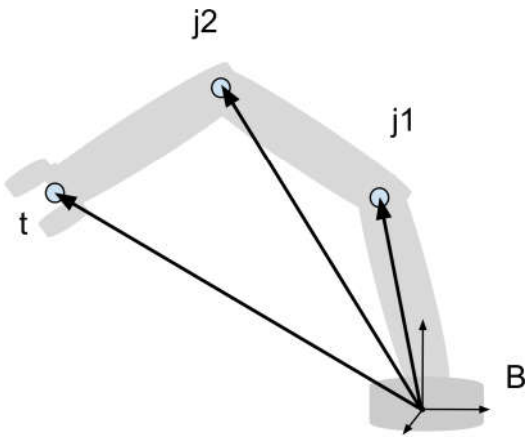

Figura A.2. Representação dos Vetores das juntas no espaço cartesiano

Espaço de atuadores:

O espaço de atuadores é o espaço de referência de posição dos atuadores, na maioria das vezes os sensores de posição dos robôs encontram-se acoplados aos servomecanismos de atuação (por exemplo: encoders nos motores). Este espaço tem como referência um ponto na movimentação do próprio atuador, por exemplo um Ângulo específico de um eixo.

Espaço de Trabalho:

O espaço de trabalho representa o volume total o qual o atuador final, ou punho, do robô pode atingir com todos os valores possíveis de posição das juntas. Para isso leva-se em consideração as limitações impostas pela construção do robô. Têm como subespaço o chamado espaço destro que consiste em todo o volume alcançável pela ferramenta com uma orientação qualquer, sendo que esta está acoplada ao punho podendo assim realizar extensões do mesmo. Pode-se ver sua representação na Fig.A.3.

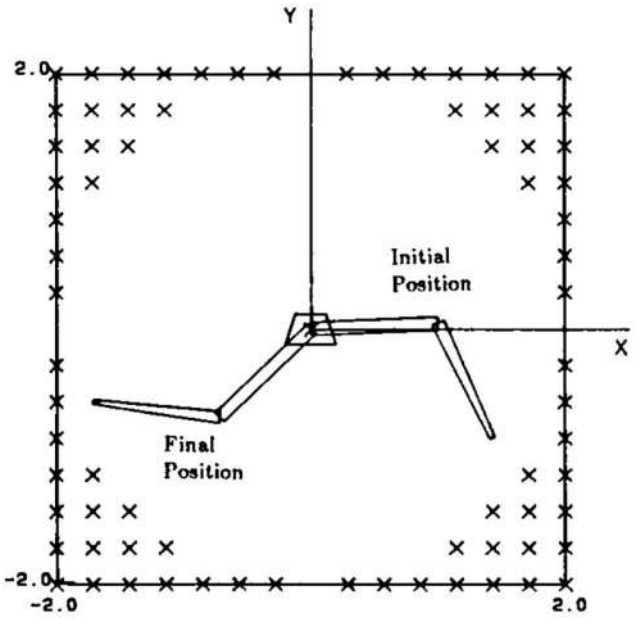

Figura A.3. Descrição do espaço de trabalho de um robô RR 2links de [Shiller & Dubowsky, 1991]

Configuração:

Configuração é a descrição da posição e orientação instantânea das juntas do robô em relação ao espaço cartesiano, porém sem descrever parâmetros dinâmicos do robô [Spong et al., 2006]. A configuração é muito utilizada quando existem similaridades (várias possíveis soluções de cinemática para uma mesma posição alvo), singularidades (ponto alvo o qual não tem solução de cinemática) ou obstáculos a serem desviados dentro do espaço de trabalho. Também pode ser chamada de "pose do robô", em analogia a postura de um modelo artístico, o qual fica parado em uma posição estática.

Espaço de Configurações:

O espaço de configurações é a especificação completa das localizações possível de todos os pontos do robô no espaço de juntas de movimentação [Spong et al., 2006]. O Conjunto de

todas as configurações possíveis por um robô, sendo um subespaços contido no espaço de trabalho. É similar ao espaço destro, porém leva em consideração obstáculos e periféricos como: Suporte de fixação de peças e dispositivos acessórios ao trabalho a ser executado. É muito útil para alguns tipos de planejamento de movimento, pois define um espaço onde o robô está livre de colisões e ajuda a executar diretivas especiais para que ele passe por pontos de singularidade e similaridade. Também é utilizado para planejamentos implícitos como mapas topológicos e campos potenciais artificiais. Pode-se ver sua representação na Fig.A.4, onde os "X"representam o espaço ao entorno no espaço de trabalho, e os "+"representam as configurações que se tornaram inalcançáveis por causa dos obstáculos.

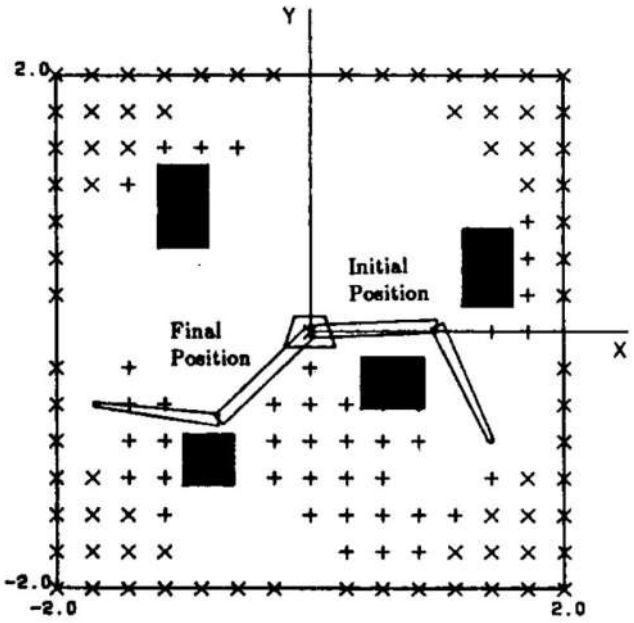

Figura A.4. Descrição do espaço de configuração de um robô RR 2linksde [Shiller & Dubowsky, 1991]

Solução de cinemática Inversa:

A solução da cinemática inversa é a relação matemática que descreve as soluções possíveis para o posicionamento das juntas e links formando uma configuração possível para o robô atingir uma posição alvo, como explicado por [Craig, 2009]. Em outras palavras é o conjunto de funções matemáticas necessárias para se encontrar, em relação posição da ferramenta no plano de tarefas, os pontos de posição das juntas e dos links necessários para que o robô possa estar nesta posição. Estes pontos de posição se tornarão os alvos para a solução cinemática direta como mostrado na Fig. A.5.

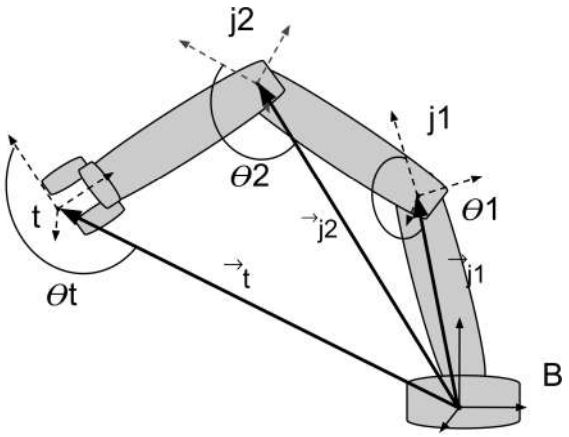

Figura A.5. Descrição da posição e orientação das juntas no espaço cartesiano

Solução Cinemática Direta:

A solução da cinemática direta é a descrição matemática a qual define a partir da base do robô a posição relativa entre as juntas para posicioná-las formando a configuração necessária para que o robô chegue à posição alvo. [Craig, 2009] demonstra isso na Fig. A.6 onde as posições de θ_1,θ_2 e θ_3 descrevem a pose desejada.

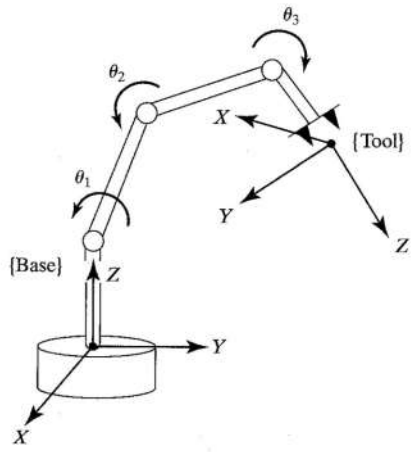

Figura A.6. Figura da representação da cinemática direta Fig. de [Craig, 2009]

Solução Cinemática de Atuadores:

A solução cinemática de atuadores é a relação matemática necessária para posicionar a junta da forma definida pela cinemática direta. Como exemplo simplificado pode-se colocar a conversão de movimento angular de um motor elétrico para uma posição linear

de um fuso de esferas como explicado por [Craig, 2009]P6, ou reduções utilizadas em movimentos angulares.

Estado e Espaço de Estados:

O estado do robô é o conjunto de variáveis que junto a descrição dinâmica e a entrada de comando determina qual é o próximo estado [Spong et al., 2006]. Similar a configuração, mas com os parâmetros dinâmicos de velocidade e aceleração. O Espaço de estados é o conjunto de todos os estados possíveis de serem movimentados pelo robô.

Jacobiano:

O Jacobiano é a relação matemática que descreve a transformação das velocidades angulares e lineares entre os espaços de junta e de tarefa. De outra forma é uma matriz das derivadas parciais dos parâmetros que descrevem as posições das juntas no espaço de estados quando o robô se movimenta entre duas configurações.

Dentre as utilidades do estudo da função jacobiana da solução cinemática direta de um robô está a determinação dos pontos de singularidade e similaridade do robô, utilizando a determinante da matriz jacobiana. Com isso pode-se determinar movimentos e posições a serem restringidas no espaço de configurações: Por serem geometricamente possíveis, porém dinamicamente problemáticas, requerer um torque infinito dos motores; ou ter múltiplas soluções geométricas, sendo necessária a utilização de diretivas adicionais para escolha de uma solução.

Lagrangiano:

As relações lagrangianas, também chamadas de equações Euler-Lagrangianas, são as derivadas das equações da segunda lei de Newton aplicadas a cada grau de liberdade do robô. São utilizadas para calcular as energias potenciais e cinéticas de cada link e junta do robô em sua dinâmica. sua utilização é cada vez mais importante no controle de juntas de um robô. Pois, possibilita o conceito de trabalho virtual, que tanto pode ser aplicado à otimização de trajetória, no aumento de estabilidade de juntas por otimização de controle, e para criar os limites de força e torque em robôs colaborativos.