UNIVERSIDADE FEDERAL DO CEARÁ

CAMPUS SOBRAL

DEPARTAMENTO DE ENGENHARIA DA COMPUTAÇÃO

CURSO DE GRADUAÇÃO EM ENGENHARIA DA COMPUTAÇÃO

JOSÉ MATHEUS SOARES FERREIRA

APROXIMAÇÃO DA CINEMÁTICA INVERSA DE UM ROBÔ MANIPULADOR DIDÁTICO ATRAVÉS DE ALGORITMOS DE APRENDIZADO DE MÁQUINA

SOBRAL

2024

JOSÉ MATHEUS SOARES FERREIRA

APROXIMAÇÃO DA CINEMÁTICA INVERSA DE UM ROBÔ MANIPULADOR DIDÁTICO ATRAVÉS DE ALGORITMOS DE APRENDIZADO DE MÁQUINA

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia da Computação do Campus Sobral da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Engenharia da Computação.

Orientador: Prof. Me. David Nascimento Coelho.

SOBRAL

2024

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Sistema de Bibliotecas
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

S654a Soares Ferreira, José Matheus.
Aproximação da cinemática inversa de um robô manipulador didático através de algoritmos de aprendizado de máquina / José Matheus Soares Ferreira. – 2024.
77 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Sobral, Curso de Engenharia da Computação, Sobral, 2024.
Orientação: Prof. Me. David Nascimento Coelho.

1. Cinemática inversa. 2. Robô manipulador. 3. Aprendizado de Máquina. I. Título.
CDD 621.39

JOSÉ MATHEUS SOARES FERREIRA

APROXIMAÇÃO DA CINEMÁTICA INVERSA DE UM ROBÔ MANIPULADOR DIDÁTICO ATRAVÉS DE ALGORITMOS DE APRENDIZADO DE MÁQUINA

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia da Computação do Campus Sobral da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Engenharia da Computação.

Aprovada em: 02 de outubro de 2024

BANCA EXAMINADORA

Prof. Me. David Nascimento Coelho (Orientador)
Universidade Federal do Ceará (UFC)

Profa. Dr. Jermana Lopes de Moraes
Universidade Federal do Ceará (UFC)

Prof. Me. Humberto Ícaro Pinto Fontinele
Universidade da Integração Internacional da Lusofonia Afro-Brasileira (UNILAB)

RESUMO

O cálculo da cinemática inversa é um desafio na robótica, especialmente para manipuladores robóticos, e métodos tradicionais muitas vezes não são eficazes. Este trabalho propôs a utilização de algoritmos de aprendizado de máquina, com foco em abordagens de modelos locais e globais, para aproximar a cinemática inversa de um robô manipulador didático. Especificamente, foram empregadas redes neurais Perceptron Multicamadas e o algoritmo *K-means* para os modelos locais segmentarem o espaço operacional. Inicialmente, a modelagem da cinemática direta é realizada, utilizando a notação de Denavit-Hartenberg, de modo a gerar um banco de dados contendo informações do espaço cartesiano e do espaço das juntas do manipulador. Este banco de dados é então filtrado, removendo amostras inalcançáveis e amostras próximas entre si no espaço cartesiano. A fase de otimização de hiperparâmetros para os modelos globais envolveu ajustar a melhor arquitetura das redes neurais, e para os modelos locais abrangeu determinar o número k de *clusters* que melhor dividisse o espaço operacional. A avaliação de desempenho dos algoritmos é feita com base nas medidas R^2 e MSE, tanto no espaço das juntas como no espaço cartesiano, e os resultados obtidos evidenciam que ambas as abordagens se mostraram viáveis para a tarefa em questão, com a abordagem local superando a global.

Palavras-chave: Robô Manipulador; Cinemática Inversa; Cinemática Direta; Perceptron Multicamadas; Modelos Locais; Modelos Globais.

ABSTRACT

The calculation of inverse kinematics is a challenge in robotics, especially for robotic manipulators, and traditional methods are often ineffective. This work proposed the use of machine learning algorithms, focusing on local and global model approaches, to approximate the inverse kinematics of an educational robotic manipulator. Specifically, Multilayer Perceptron neural networks and the K-means algorithm were employed for the local models to segment the operational space. Initially, direct kinematics modeling is carried out using Denavit-Hartenberg notation to generate a database containing information about the Cartesian space and the manipulator's joint space. This database is then filtered, removing unreachable samples and samples that are close to each other in Cartesian space. The hyperparameter optimization phase for the global models involved adjusting the best architecture of the neural networks, while for the local models, it included determining the optimal number k of clusters that best divided the operational space. The performance evaluation of the algorithms is based on R^2 and MSE measures, both in joint space and in Cartesian space, and the results obtained demonstrate that both approaches proved viable for the task at hand, with the local approach outperforming the global one.

Keywords: Robotic Manipulator; Inverse kinematics; Direct kinematics; Multilayer Perceptron; Local models; Global Models.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

R^2	Coeficiente de Determinação
tanh	Tangente Hiperbólica
AM	Aprendizado de Máquina
ANFIS	*Adaptive Network Fuzzy Inference System*
CNN	*Convolutional Neural Networks*
D-H	Denavit-Hartenberg
ELM	*Extreme Learning Machine*
GDL	Graus de Liberdade
IA	Inteligência Artificial
ISO	*International Standard Organization*
LLM	*Linear Local Mapping*
LS-SVR	*Least-Squares Support Vector Regression*
MIMO	*Multiple Inputs, Multiple Outputs*
MLM	*Minimal Learning Machine*
MLP	*Multi Layer Perceptron*
MP	McCulloch e Pitts
MSE	*Mean Squared error*
PG	Processo Gausiano
PLA	*Polylactic Acid*
PS	Perceptron Simples
ReLU	*Rectified Linear Unit*
RIA	*Robot Institute of America*
RNA	Redes Neurais Artificiais
RNN	*Recurrent Neural Networks*
SVM	*Support Vector Machine*

LISTA DE SÍMBOLOS

θ_i	Ângulo de rotação da junta
d_i	Deslocamento do elo
a_i	Tamanho do elo
α_i	Ângulo de torção do elo
$\{A\}$	Sistema de referência (*frame*) A
$\{B\}$	Sistema de referência (*frame*) B
${}^A_B T$	Matriz de transformação homogênea
${}^B P$	Descrição de um ponto P definido sobre o sistema $\{B\}$
${}^A P$	Descrição de um ponto P definido sobre o sistema $\{A\}$
${}^A P_{BORG}$	Vetor que localiza a origem do sistema $\{B\}$ em relação ao sistema $\{A\}$
$\varphi(\cdot)$	Função de ativação
b_k	Limiar de ativação (Bias) do k-ésimo neurônio
x_m	Vetor de atributos da m-ésima entrada
w_{km}	Peso do k-ésimo neurônio da m-ésima entrada
u_k	Ativação do k-ésimo neurônio
$w(t)$	Conhecimento atual dos neurônios
$w(t+1)$	Acréscimo de conhecimento dos neurônios
$x(t)$	Informação fornecida pelo vetor de entrada
$e(t)$	Erro entre a saída desejada e a saída gerada pela rede
η	Passo de aprendizado da rede neural
V	Tensão
d_s	Distância de similaridade entre as amostras
n_r	Número de iterações de treinamento dos modelos

SUMÁRIO

1 INTRODUÇÃO

A primeira vez que o termo “robô” foi utilizado ocorreu na dramaturgia em 1920 durante uma peça teatral tcheca de título “R.U.R - Rossum’s Universal Robots", onde rapidamente se popularizou no imaginário da sociedade. No entanto, somente em 1954, devido aos avanços tecnológicos do século XX, foi possível efetivamente desenvolver o primeiro robô patenteado por George C. Devol, Jr. (1912-2011) (SPONG *et al.*, 2020).

Desde então, a robótica desenvolveu-se e incorporou-se à sociedade, desempenhado um papel crucial em diversos setores que vão desde a indústria manufatureira e automobilística, exploração subaquática, saúde e medicina, exploração espacial, monitoramento ambiental, agricultura, até na assistência domiciliar oferecendo serviços às pessoas (CORKE, 2017).

Em grande parte, isso deve-se aos robôs possuírem uma maior eficiência, agilidade, precisão, flexibilidade e capacidade de executar tarefas perigosas ou de risco à saúde humana, o que gera grandes vantagens quanto ao seu uso, como reduzir custos de trabalho, eliminar trabalhos perigosos, aumentar a taxa de produção, melhorar a qualidade do produto, reduzir desperdício e custos de capital (CORKE, 2017).

Especificamente, para tratar movimentos de um robô manipulador, é necessário desenvolver técnicas para modelar a localização do braço no espaço cartesiano ao longo tempo, portanto, definindo o comportamento cinemático do robô. Nesse caso, há uma relação entre o espaço das juntas e o espaço operacional do manipulador, e por isso a cinemática é dividida entre direta e inversa (OLIVEIRA, 2007).

A cinemática inversa possui uma complexidade maior comparada a cinemática direta, já que é entendida como um problema mal posto, o que faz o sistema possuir várias soluções ou nenhuma solução para determinar a posição final do efetuador de um manipulador (OGAWA; KANADA, 2010).

O uso de manipuladores didáticos no ensino de robótica tem se mostrado uma ferramenta eficaz para a aprendizagem ativa, proporcionando aos estudantes uma compreensão mais concreta dos conceitos complexos da cinemática dos robôs. Esses manipuladores permitem que os alunos interajam diretamente com os sistemas robóticos, facilitando a visualização e a experimentação dos princípios teóricos em situações práticas. Além disso, a utilização de ferramentas práticas no processo educacional gera um maior engajamento e a retenção do conhecimento (ALIMISIS, 2013).

À medida que o manipulador possui maiores Graus de Liberdade (GDL), surgem

também diversas complexidades que envolvem a cinemática do robô. Para lidar com essa complexidade, nas últimas décadas, vem sendo explorado o uso de algoritmos de Aprendizado de Máquina (AM) para estimar a cinemática inversa de manipuladores (HUANG *et al.*, 2003).

O AM é uma área da inteligência artificial que tem origem na década de 1940, quando Warren McCulloch e Walter Pitts definiram o modelo de neurônio artificial, motivando o surgimento de áreas que se dedicam em desenvolver sistemas que simulem a inteligência humana. A ideia é dotar os computadores de inteligência e autonomia na tomada de decisões com base nas experiências passadas.

Desde então, o aprendizado de máquina continua evoluindo e possui diversas aplicações práticas como é o caso do uso de *Recurrent Neural Networks* (RNN) em sistemas de reconhecimento de voz (GRAVES *et al.*, 2013), na detecção de atividades fraudulentas em transações bancárias (CHANDOLA *et al.*, 2009) e na análise de imagens médicas auxiliando o diagnóstico médico de doenças (ESTEVA *et al.*, 2017).

Um exemplo de algoritmo AM utilizado para solucionar a modelagem da cinemática de manipuladores são as Redes Neurais Artificiais (RNA), as quais, segundo algumas literaturas, têm tido uma abordagem promissora nesta aplicação. A partir destas, não há mais dependência de soluções analíticas mais complexas, visto que RNA são conhecidas por serem aproximadoras universais de funções (HORNIK *et al.*, 1989).

1.1 Objetivos

O objetivo principal do trabalho consiste na aproximação da cinemática inversa de um robô manipulador didático através de algoritmos de aprendizado de máquina comparando as abordagens de modelos globais e locais.

1.1.1 Objetivos Específicos

A partir dos objetivos gerais, os seguintes objetivos específicos foram traçados:

- Estudar métodos para geração da cinemática direta de um manipulador robótico.
- Investigar dados relevantes que relacionem os ângulos das juntas com a posição cartesianas para fins de treinamento dos modelos de aprendizado de máquina.
- Pesquisar modelos de regressão para aproximação de cinemática inversa.
- Investigar métodos de avaliação para os modelos que solucionam a cinemática inversa.

2 TRABALHOS RELACIONADOS

Uma variedade de estudos tem explorado a aplicação de algoritmos de aprendizado para resolver a cinemática inversa de manipuladores robóticos, enfrentando desafios complexos na modelagem desses sistemas. Abordagens como RNA, algoritmos genéticos, RNN e *Support Vector Machine* (SVM) têm sido utilizadas para estabelecer a relação entre as coordenadas cartesianas e os ângulos das juntas. Essas metodologias destacam-se pela capacidade de lidar com a não linearidade e a alta dimensionalidade do problema, resultando em soluções precisas e eficientes.

Junior (2014) propôs solucionar a cinemática inversa de um manipulador didático servo controlado utilizando redes neurais e algoritmos genéticos, analisando dois cenários: um simplificado, variando todas as juntas exceto a última, e um completo, variando todas as juntas. A dinâmica do sistema e o controle de trajetória não foram considerados. As redes neurais usaram como entrada os resultados das equações da modelagem direta usando a notação Denavit-Hartenberg (D-H) e, no cenário simplificado, obtiveram um erro de 10^{-2} mm na trajetória de teste e, no cenário completo, 1 mm. Os algoritmos genéticos, com busca inicial em todo o espaço das juntas e subsequente otimização em torno do ponto próximo encontrado, apresentaram erros de 10^{-4} mm no cenário simplificado, e 10^{-3} mm no completo. Ambas as técnicas mostraram-se viáveis e eficazes para resolver a cinemática inversa do manipulador em questão.

Nos estudos comparativos de Melo (2015), são reportados os resultados obtidos na tentativa de solucionar a cinemática inversa de três manipuladores: Motoman HP6, PUMA 560 e o Robô Planar. Foram avaliados sete algoritmos de aprendizado de máquina quanto ao desempenho, utilizando análise de resíduos, *Mean Squared error* (MSE) e teste de hipótese, em um contexto de modelagem global. Os algoritmos avaliados incluem Redes Neurais Artificiais (*Multi Layer Perceptron* (MLP) e *Extreme Learning Machine* (ELM)), Regressão Baseada em Métodos de Kernel (*Least-Squares Support Vector Regression* (LS-SVR) e Processo Gausiano (PG)), Modelos Lineares Locais (*Linear Local Mapping* (LLM)), Sistemas Neuro-Fuzzy (*Adaptive Network Fuzzy Inference System* (ANFIS)) e Regressão Baseada em Distâncias (*Minimal Learning Machine* (MLM)). A cinemática direta dos robôs foram modeladas pela notação de D-H, limitando o espaço de trabalho e espaço das juntas. Para simplificar a escolha do melhor algoritmo, foram comparados apenas LS-SVR e PG, que apresentaram a melhor MSE. Os testes de hipóteses indicaram resultados mistos, sugerindo que os modelos escolhidos são equivalentes apenas para um conjunto específico de ângulos. O estudo concluiu que tanto LS-SVR

quanto PG mostraram-se eficazes para a aproximação da cinemática inversa dos manipuladores, destacando-se pela precisão em cenários específicos e contribuindo para o avanço das técnicas de aprendizado de máquina aplicadas à robótica.

No trabalho de Fontinele (2015), outro estudo comparativo é conduzido também na tarefa de aproximação do modelo cinemático inverso de 3 manipuladores (Motoman HP6, PUMA 560 e o Robô Planar). Porém, neste trabalho, seis modelos locais são avaliados: rede de funções de base radial (RBFN), rede de modelos locais (LMN), mapeamento linear local baseado em SOM (LLM), mapeamento linear local usando K vencedores (KSOM), regressão local ponderada (LWR) e rede counterpropagation (CP). Os modelos usaram como entrada os resultados das equações da modelagem direta usando a notação D-H, que foram filtrados para excluir plurivocidades decorrentes das redundâncias dos motores Motoman HP6 e PUMA 560. Para avaliar estes modelos, os erros quadráticos médios no espaço das juntas foram calculados, de modo a verificar a diferença entre os ângulos estimados em relação aos ângulos desejados. Também foram gerados conjuntos de testes baseados em trajetórias no espaço cartesiano, e calculou-se a distância média entre as trajetórias desejadas e estimadas. Uma contribuição deste trabalho foi mostrar que, não necessariamente um melhor desempenho de um determinado algoritmo no espaço das juntas implica em um melhor desempenho no espaço cartesiano. Além disso, os modelos RBF e LMN apresentaram os melhores desempenhos globais para os três manipuladores estudados.

Em Nunes (2016) a cinemática inversa de um manipulador servo controlado de cinco GDL foi resolvida utilizando redes neurais artificiais. Os modelos foram treinados offline e utilizados para prever trajetórias e pontos em tempo real. O mapeamento do volume de trabalho do manipulador foi realizado com a modelagem da cinemática direta utilizando a notação de D-H. Duas abordagens de treinamento foram seguidas: gerando trajetórias para limitar o espaço operacional e definindo um conjunto de pontos para o volume de trabalho. Já as redes neurais foram configuradas de duas maneiras: uma MLP simples e outra MLP em paralelo seguindo a abordagem de modelos locais, ambas com entradas cartesianas. A configuração de modelos locais obteve um desempenho superior, reduzindo o erro em cerca de $87,8\%$ para trajetórias e 80% para o espaço de trabalho completo em comparação com a rede neural simples. Esses resultados sugerem que o uso de redes neurais em paralelo é uma abordagem promissora na cinemática inversa de manipuladores robóticos, contribuindo para a precisão e eficiência da solução da cinemática inversa de aplicações robóticas.

Em Demby's *et al.* (2019), é investigado a resolução da cinemática inversa de robôs seriais utilizando RNA e Sistemas de Inferência ANFIS. Os modelos foram treinados para mapear os espaços de trabalho completos de robôs seriais com 4, 5, 6 e 7 graus de liberdade, utilizando duas configurações: *All-Joints-ANN*, que prevê todas as saídas das juntas dadas as entradas desejadas, e *Individual-Joints-ANN*, que prevê cada saída de junta separadamente. Embora os modelos utilizassem variáveis cartesianas e números quaternions para representar as orientações, o treinamento focou na predição das poses dos efetuadores finais sem considerar as orientações. Os resultados indicam que tanto as RNA quanto as ANFIS foram capazes de aprender a cinemática inversa dos robôs com um grau significativo de precisão, apesar de a performance variar conforme a complexidade do robô e o número de GDL. As RNA e ANFIS demonstraram ser ferramentas viáveis para resolver problemas de cinemática inversa, embora desafios como erros de aproximação e tempo de treinamento ainda persistam. O estudo sugere que, apesar de promissores, esses métodos devem ser comparados e possivelmente combinados com abordagens tradicionais para otimizar sua aplicação prática.

Gao (2020) propõe a solução da cinemática inversa de um robô manipulador de seis GDL utilizando um algoritmo de redes neurais aprimorado. O modelo emprega retro-propagação baseada em uma função de excitação de processamento adaptativo, que resolve as limitações de convergência dos algoritmos tradicionais da cinemática inversa limitando uma região do volume de trabalho do manipulador. A abordagem inovadora foca na melhoria da precisão e eficiência da solução da cinemática inversa, adaptando a função de excitação para otimizar a convergência do algoritmo. Os resultados indicam que o algoritmo aprimorado supera os métodos tradicionais de AM em termos de velocidade e precisão, reduzindo significativamente os erros de aproximação. O autor conclui que o uso de redes neurais adaptativas oferece uma solução robusta e eficiente para a cinemática inversa de robôs manipuladores, destacando-se como uma ferramenta promissora para aplicações avançadas na modelagem da cinemática robótica.

Embora o uso de algoritmos de aprendizado de máquina na solução da cinemática inversa de manipuladores seja relativamente comum, especialmente com a aplicação de redes neurais, não são comuns estudos que mapeiem todo o volume de trabalho alcançável pelo manipulador, como proposto neste trabalho. Isso se deve, especialmente, à característica da não linearidade e singularidade da cinemática inversa, que limita os estudos a treinarem modelos para uma sub-região do espaço operacional, a fim de reduzir a complexidade da solução. Essa característica também impactou na geração de dados de treinamento do presente trabalho, no

qual se propôs uma abordagem para o tratamento de dados redundantes e inalcançáveis por meio de uma técnica de similaridade.

Além disso, muitos estudos se restringem a utilizar apenas atributos de posição cartesiana ou todos os elementos da matriz de transformação homogênea, o que, na prática, torna a solução inviável. Neste trabalho, foram estruturados atributos de posição e orientação, e definidas duas abordagens de métricas para avaliar o erro dos modelos, a fim de compreender melhor seu comportamento. Uma dessas métricas avalia o erro no espaço das juntas, enquanto a outra se concentra nos erros no espaço operacional. Por fim, destaca-se que, devido à estrutura física específica do manipulador didático, os resultados obtidos são reais e exclusivos para esse modelo.

3 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo, serão apresentados os fundamentos necessários para a compreensão do trabalho em questão. Primeiramente, será explorada a cinemática de manipuladores, abordando a definição de manipuladores, a modelagem da posição e orientação, bem como a descrição espacial de objetos em relação a sistemas de referência. O foco recai sobre o movimento dos manipuladores, considerando a posição e orientação do efetuador e as configurações das juntas, sem levar em conta as forças envolvidas. Também serão apresentadas as notações utilizadas na modelagem cinemática de manipuladores.

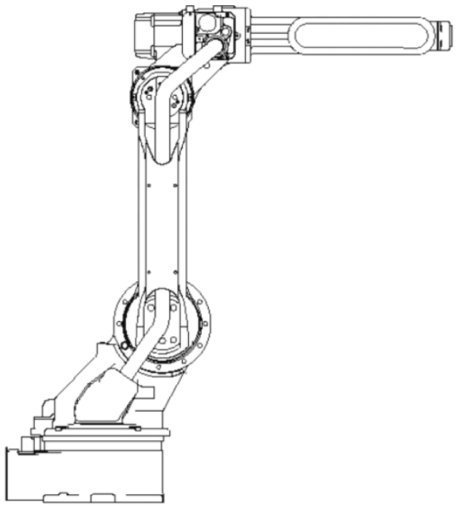
Em seguida, será abordado os algoritmos de aprendizado de máquina, com ênfase em modelos inspirados no sistema nervoso humano, como as redes neurais artificiais, e seus modelos de regressão. Além de modelos que segmentam o espaço de trabalho do manipulador.

3.1 Comportamento Cinemática de Manipuladores

Conforme a *International Standard Organization* (ISO) (ISO, 2012) e a *Robot Institute of America* (RIA), um robô pode ser definido como um manipulador reprogramável e multifuncional, projetado com vários graus de liberdade com base móvel ou fixada. Este pode ser capaz de sentir o ambiente e sobre ele manipular instrumentos, seguindo uma trajetória a fim de alcançar algum objetivo (RIASCOS, 2010). Além disso, um robô pode ser entendido como um dispositivo eletromecânico constituído de um circuito eletrônico de controle computadorizado e um mecanismo articulado, chamado de manipulador, capaz de executar uma variedade de funções (CORKE, 2017).

Existem distintas configurações geométricas de um robô e, para cada uma delas, há diferentes arranjos e características entre a estrutura física e as articulações utilizadas. Os robôs manipuladores, como exemplo do robô manipulador Motoman HP6 ilustrado na Figura 1, oferecem seis graus de liberdade otimizados em comprimento e geometria para eficiência, e por isso são amplamente utilizados atualmente. Estes, ao contrário dos robôs móveis, possuem base fixa, e, por isso, não se movem no ambiente. Um manipulador é composto por elos que constituem o corpo rígido do robô, interligados por juntas ou articulações. Além destes componentes, o elemento final de controle, denominado efetuador, refere-se à parte do robô que atua sobre o ambiente, sendo responsável por executar tarefas específicas como soldar, agarrar, entre outras ações (CRAIG, 2006).

Figura 1 – Robô Manipulador Motoman HP6



Fonte: (MELO, 2015)

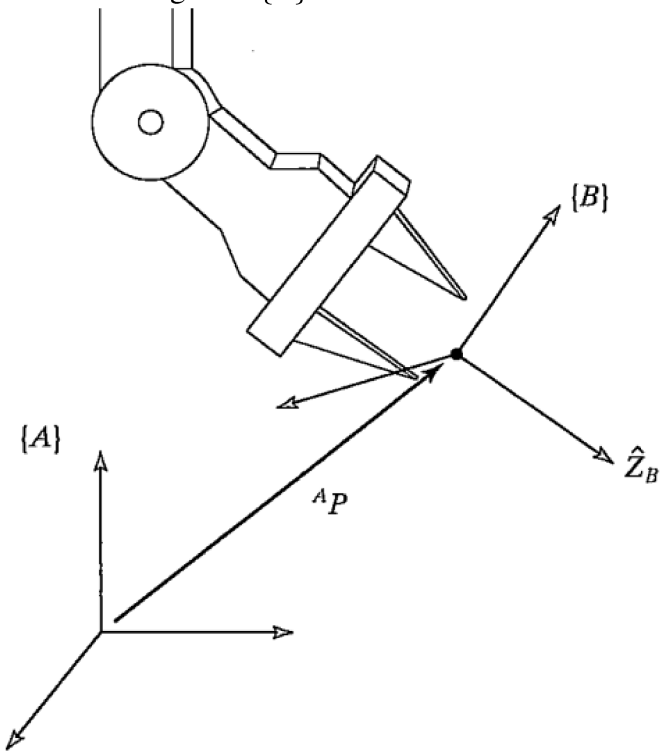
Dentro da robótica de manipuladores, é essencial compreender como os elementos pertencentes ao ambiente do robô estão posicionados e orientados. Isso inclui tanto os componentes do próprio manipulador quanto os elementos da cena ao seu redor. Portanto, é necessário adotar convenções que relacionem os elementos do robô, a fim de modelar sua posição e orientação (SICILIANO *et al.*, 2009). A modelagem da posição e orientação em sistemas robóticos é comumente abordada na literatura através de transformações homogêneas, cinemática direta e inversa, e no uso de descrições espaciais.

3.1.1 Descrições Espaciais e Transformações

Dentro da representação de um modelo espacial, a localização é definida em função de dois atributos fundamentais: a posição e a orientação, os quais são associados a um sistema de referência, conforme ilustrado o efetuador localizado no sistema de referência $\{B\}$ na Figura 2. A descrição desses atributos, seja para uma ferramenta, partes do robô, o próprio robô e os demais elementos do ambiente, constitui uma área de estudo da robótica (CRAIG, 2006).

A posição refere-se à localização de um objeto nas coordenadas espaciais em relação a um sistema de referência, o que define um ponto no espaço tridimensional (SICILIANO *et al.*, 2009). Uma vez estabelecido o sistema de referência $\{A\}$, a posição é descrita por um vetor de

Figura 2 – Órgão terminal com orientação e posição no sistema de coordenadas $\{B\}$ associados aos sistema de referência global $\{A\}$



Fonte: (SICILIANO *et al.*, 2009)

posição tridimensional $(\mathbb{R}^3)$, dado por:

$$^AP = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix}. \tag{3.1}$$

Por outro lado, quando o corpo rígido analisado possui dimensões, é necessário também representar sua orientação. A orientação descreve a direção e o alinhamento do objeto em relação a um sistema de referência, ou seja, expressa as combinações de rotações em torno dos eixos de coordenadas de dois sistemas de referências (SICILIANO *et al.*, 2009). Dessa maneira, o elemento é descrito a partir da fixação de um sistema de coordenadas no próprio corpo e relacionando-o a outro sistema de coordenadas (CRAIG, 2006).

Na Figura 2 ilustrado o sistema de referência $\{B\}$ fixado sobre um elemento e definido em relação ao sistema de coordenadas $\{A\}$. Os vetores unitários dos eixos principais do sistema de referência $\{B\}$ em termos de $\{A\}$ são definidos como $^A\hat{X}_B$, $^A\hat{Y}_B$ e $^A\hat{Z}_B$. A orientação

do sistema de coordenadas $\{B\}$ em relação a $\{A\}$, é descrita pela matriz rotacional como:

$$ {}^{A}_{B}R = \begin{bmatrix} {}^{A}\hat{X}_B & {}^{A}\hat{Y}_B & {}^{A}\hat{Z}_B \end{bmatrix} = \begin{bmatrix} \hat{X}_B \cdot \hat{X}_A & \hat{Y}_B \cdot \hat{X}_A & \hat{Z}_B \cdot \hat{X}_A \\ \hat{X}_B \cdot \hat{Y}_A & \hat{Y}_B \cdot \hat{Y}_A & \hat{Z}_B \cdot \hat{Y}_A \\ \hat{X}_B \cdot \hat{Z}_A & \hat{Y}_B \cdot \hat{Z}_A & \hat{Z}_B \cdot \hat{Z}_A \end{bmatrix}, \tag{3.2} $$

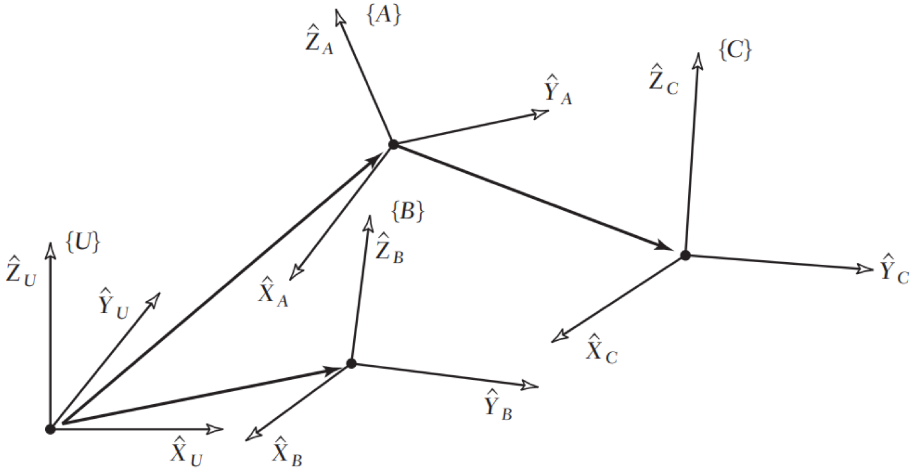
a qual representa as projeções dos diferentes vetores unitários dos dois sistemas de coordenadas (CRAIG, 2006).

O sistema de coordenadas, também denominado de sistemas de referência ou *frame*, descreve a posição e a orientação em relação a outro sistema de coordenadas. Ele fornece uma estrutura de referência fixa na qual os cálculos de posição e orientação são realizados. O sistema de referência $\{B\}$ pode ser descrito por:

$$ \{B\} = \left\{ {}^{A}_{B}R, {}^{A}P_{BORG} \right\}. \tag{3.3} $$

Um sistema de referência é denotado por $\{A\}$, e seus eixos principais são nomeados com $\hat{X}_A$, $\hat{Y}_A$ e $\hat{Z}_A$, representados por vetores unitários. Na Figura 3, são ilustrados exemplos de sistemas de referência locais com suas respectivas notações $\{A\}$,$\{B\}$ e $\{C\}$, associados a um sistema de coordenadas universal $\{U\}$. Existem diversos tipos de referenciais, entre os mais conhecidos estão o sistema de coordenadas global, de um robô, universal ou inercial (CRAIG, 2006).

Figura 3 – Exemplos de sistemas de coordenadas $\{A\}$,$\{B\}$ e $\{C\}$ associados ao sistema de coordenadas universal $\{U\}$



Fonte: (CRAIG, 2006)

Dada a descrição espacial de objetos, torna-se necessário estabelecer um mapeamento entre os distintos sistemas de coordenadas. Esse mapeamento deve descrever as translações e rotações existentes entre as diferentes referências.

Craig (2006) define que para dois sistemas de coordenadas $\{A\}$ e $\{B\}$ com translações distintas, onde um ponto ${}^{B}P$ é definido sobre o sistema de referência $\{B\}$. O mapeamento que expressa esse ponto em termos do sistema de coordenadas $\{A\}$ é dado por:

$$ {}^{A}P = {}^{B}P + {}^{A}P_{BORG}. \tag{3.4} $$

Já para os mesmos sistemas de coordenadas, agora rotacionados entre si e sem translação, a rotação do sistema $\{B\}$ sobre $\{A\}$, descreve as diferentes projeções dos vetores unitários de cada referencial e é dada por,

$$ {}^{A}P = {}^{A}_{B}R\,{}^{B}P. \tag{3.5} $$

Devido a necessidade de lidar com diferentes sistemas de coordenadas de forma uniforme, é definida a notação de transformações homogêneas. Por meio dessa representação, é possível concatenar várias transformações lineares de maneira simplificada. Dessa maneira, a notação combina rotação, translação e escala para implementar a orientação e a posição dos *frames* que compõem o sistema robótico por meio de uma notação matricial da transformação homogênea tridimensional (SICILIANO *et al.*, 2009).

Na Figura 4, são apresentados os sistemas de coordenadas denotados por $\{A\}$ e $\{B\}$. No sistema de coordenadas $\{B\}$, é definido um ponto ${}^{B}P$ conhecido. A matriz de transformação homogênea que mapeia o sistema de coordenadas $\{B\}$ em relação ao sistema de coordenadas $\{A\}$ é descrita como:

$$ {}^{A}_{B}T = \begin{bmatrix} {}^{A}_{B}R & {}^{A}P_{BORG} \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} R_{11} & R_{12} & R_{13} & p_x \\ R_{21} & R_{22} & R_{23} & p_y \\ R_{31} & R_{32} & R_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{3.6} $$

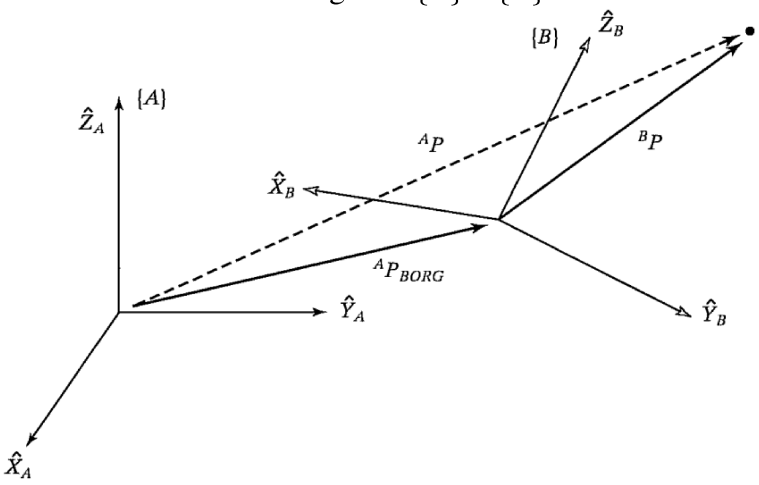
em que T_B^A mapeia a transformação geral do vetor ${}^{B}P$, percorrendo a descrição do seu *frame* $\{B\}$ para o $\{A\}$ seguinte (CRAIG, 2006).

Portanto, a notação da matriz de transformações homogêneas do ponto ${}^{B}P$ em relação ao sistema de referência $\{A\}$ é dado por:

$$ \begin{bmatrix} {}^{A}P \\ 1 \end{bmatrix} = \begin{bmatrix} {}^{A}_{B}R & & & {}^{A}P_{BORG} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} {}^{B}P \\ 1 \end{bmatrix}. \tag{3.7} $$

Destaca-se que os nove elementos da matriz de transformação rotacional não são equações independentes, uma vez que as colunas são mutuamente ortogonais e vetores unitários.

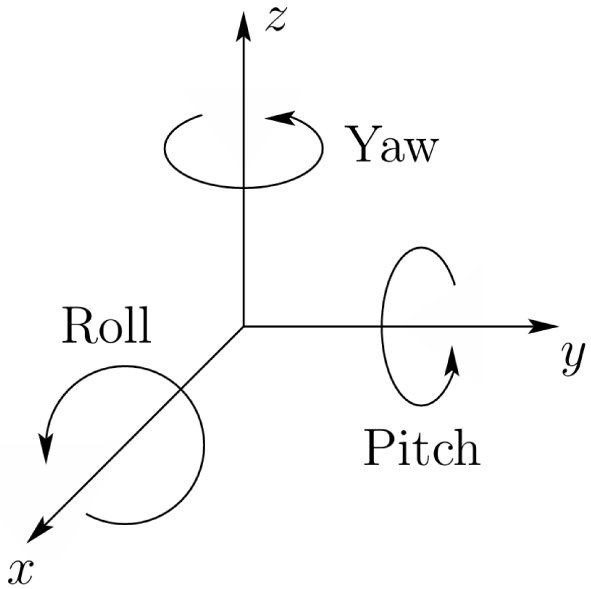
Figura 4 – Representação das transformações envolvendo sistemas de coordenadas gerais $\{A\}$ e $\{B\}$



Fonte: (SICILIANO *et al.*, 2009)

Essas restrições determinam a existência de apenas três variáveis independentes. Na prática, um corpo rígido possui, no máximo, três graus de liberdade rotacionais para especificar sua orientação. Um método comum para definir uma matriz de rotação em termos dessas três variáveis é o uso dos ângulos *roll*, *pitch* e *yaw*, denotados por ψ, β e ϕ, conforme ilustrado na Figura 5 (SPONG *et al.*, 2020).

Figura 5 – Ângulos *roll*, *pitch* e *yaw*



Fonte: (SPONG *et al.*, 2020) (adaptada)

Essa representação é mais compacta e define a transformação de orientação como um produto de rotações sucessivas em torno dos eixos principais x, y e z do sistema de coordenadas locais, seguindo a ordem específica. Primeiro uma rotação em torno do eixo de rotação em z

(*yaw*) para o ângulo ψ, em seguida um rotação no eixo *y* (*pitch*) gerando o ângulo β, por fim uma rotação em torno de *x* (*yaw*) para o ângulo ϕ, sendo expressas por:

$$R_{XYZ} = R_{z,\psi} R_{y,\beta} R_{x,\phi} \tag{3.8}$$

$$= \begin{bmatrix} c_\psi & -s_\psi & 0 \\ s_\psi & c_\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c_\beta & 0 & s_\beta \\ 0 & 1 & 0 \\ -s_\beta & 0 & c_\beta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & c_\phi & -s_\phi \\ 0 & s_\phi & c_\phi \end{bmatrix} \tag{3.9}$$

$$= \begin{bmatrix} c_\psi c_\beta & c_\psi s_\beta s_\phi - s_\psi c_\phi & c_\psi s_\beta c_\phi + s_\psi s_\phi \\ s_\psi c_\beta & s_\psi s_\beta s_\phi + c_\psi c_\phi & s_\psi s_\beta c_\phi - c_\psi s_\phi \\ -s_\beta & c_\beta s_\phi & c_\beta c_\phi \end{bmatrix}, \tag{3.10}$$

onde $s_\beta = \sin(\beta)$, $c_\beta = \cos(\beta)$, $s_\psi = \sin(\psi)$, $c_\psi = \cos(\psi)$ e $c_\phi = \cos(\phi)$, $s_\phi = \sin(\phi)$. Cada uma dessas rotações pode ser representada por uma matriz de rotação, e a sequência completa de rotações é expressa como o produto dessas matrizes individuais (SPONG *et al.*, 2020).

Considerando a matriz de rotação descrita na Equação 3.6, os três ângulos ψ, β e ϕ, podem ser obtidos através de um método que relaciona os elementos da matriz com esses ângulos. Esse método descreve as seguintes equações:

$$\phi = atan2(r_{32}, r_{33}) \tag{3.11}$$

$$\psi = atan2(r_{21}, r_{11}) \tag{3.12}$$

$$\beta = \begin{cases} atan2\left(-r_{31}, \frac{r_{21}}{\sin(\psi)}\right), & \text{se } \cos(\psi) = 0 \\ atan2\left(-r_{31}, \frac{r_{11}}{\cos(\psi)}\right), & \text{caso contrário} \end{cases} \tag{3.13}$$

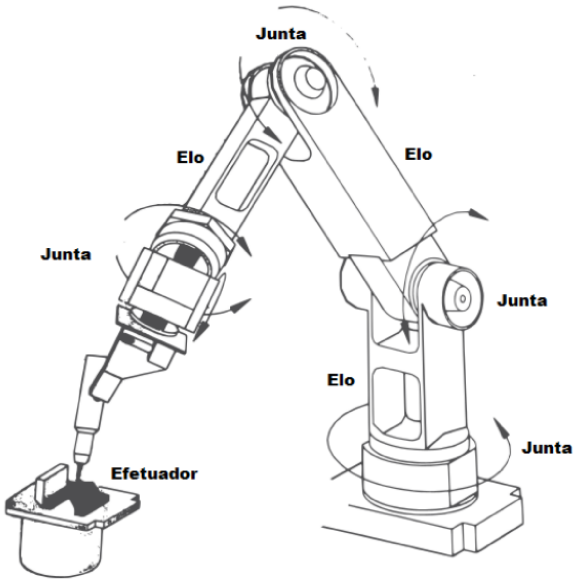
3.1.2 Descrição de Elos e Conexões

Um manipulador robótico é composto por uma cadeia de elos, interligados por juntas, conforme ilustrado na Figura 6. Os elos são os corpos rígidos do robô, enquanto as juntas conectam dois ou mais elos e são responsáveis pela movimentação relativa entre eles. Esses componentes permitem a realização dos movimentos do robô (NUNES, 2016).

A maioria dos manipuladores é equipado com juntas que podem ser de dois tipos principais: rotacionais ou deslizantes, estas últimas também conhecidas como prismáticas (CRAIG, 2006).

As juntas rotacionais permitem que um elo gire em relação ao elo vizinho em torno de um eixo rotacional, esses deslocamento é denominado de ângulo de junta. Já as juntas

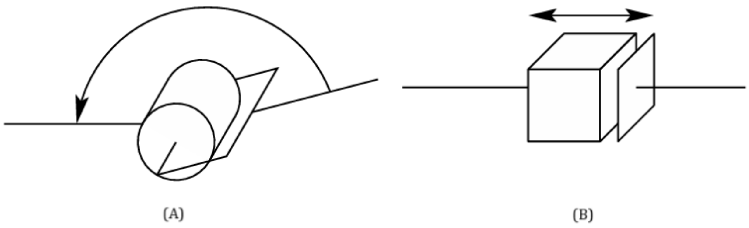
Figura 6 – Manipulador composto de elos, juntas e do efetuador



Fonte: (JESUS, 2019)

prismáticas possibilitam a movimentação linear de um elo em relação ao seu vizinho, permitindo um movimento retilíneo entre dois elos (NUNES, 2016). Dessa maneira, um manipulador pode ser representado por *n* corpos rígidos móveis e um corpo fixo, ligados por *n* articulações, defindindo a forma de estrutura da cadeia cinemática.

Figura 7 – Representação de juntas (A) rotacionais e (B) prismáticas



Fonte: (SPONG *et al.*, 2020) (adaptada)

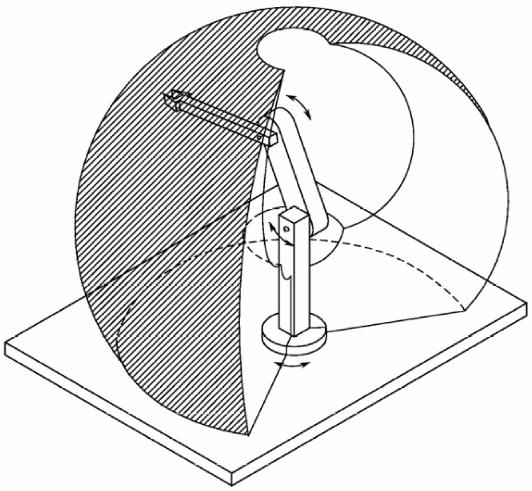
Os elos de um manipulador robótico são enumerados progressivamente, partindo da base fixa até o efetuador, em ordem crescente, começando pelo elo 0. As juntas, por sua vez, são enumeradas a partir do seu primeiro grau de liberdade, iniciando pela junta 1 (CRAIG, 2006).

Os Graus de Liberdades (GDL) de um manipulador robótico referem-se aos diferentes movimentos que o robô pode executar. Cada junta no braço do robô permite um desses movimentos, possibilitando a realização de rotações e translações ao redor de diferentes eixos. A

quantidade total de GDL determina a complexidade e a capacidade do robô para atingir variadas posições e executar diversas tarefas (CRAIG, 2006).

O espaço de trabalho de um robô, definido pelos pontos que o efetuador pode alcançar, é determinado pelos GDL do manipulador, ilustrado na Figura 8. O volume de trabalho do manipulador é, portanto, limitado pela sua construção e pelo projeto do robô (NUNES, 2016). Dentro desta região, o espaço de trabalho alcançável é aquele em que o efetuador consegue localizar-se com pelo menos uma orientação e posição.

Figura 8 – Volume de trabalho de um manipulador robótico



Fonte: (SICILIANO *et al.*, 2009)

No estudo de manipuladores com múltiplos GDL, a posição de cada elo pode ser descrita por um conjunto de variáveis conhecido como vetor de juntas. Este conjunto forma o chamado espaço das juntas. Por outro lado, o termo espaço operacional é usado para se referir ao espaço cartesiano, onde a posição e a orientação são medidas em relação a eixos ortogonais e de acordo com convenções estabelecidas na Seção 3.1.1.

3.1.3 Cinemática de Manipuladores

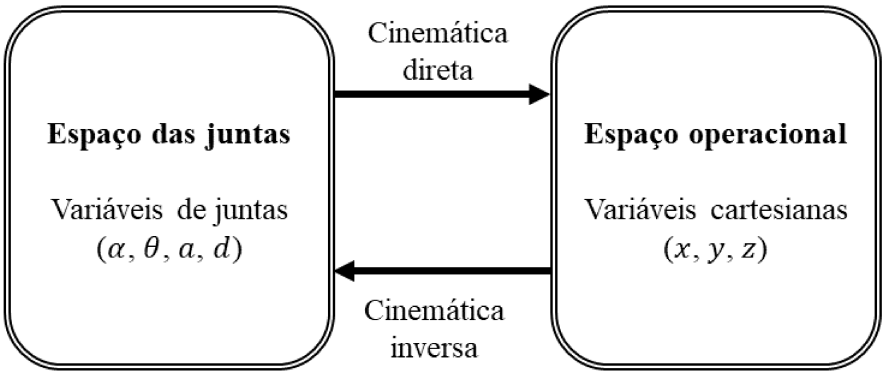
A cinemática robótica é a ciência que estuda o movimento gerado pelo manipulador sem se preocupar com as forças que agem sobre o mecanismo. Em outras palavras, considera-se apenas a posição, a velocidade e a aceleração do sistema, bem como as propriedades geométricas do sistema robótico. O objetivo é equacionar a posição e a orientação do elemento final de controle a cada instante de tempo, de acordo com os deslocamentos das juntas (CRAIG, 2006).

Dessa maneira, por meio da geometria física do manipulador, é possível determinar

a localização do elemento final de controle considerando os ângulos de cada junta. Alternativamente, pode-se calcular os ângulos das juntas necessários para que o elemento final de controle esteja em uma determinada posição e orientação (CRAIG, 2006).

Portanto, para manipuladores robóticos, há uma relação entre o espaço das juntas, dado pela posição angular das articulações, e o espaço operacional, dado pela posição e orientação do elemento final de controle em referência ao espaço cartesiano. Nesse sentido, a cinemática é dividida em direta e inversa (MELO, 2015). Para a modelagem da cinemática do robô, é necessário estabelecer uma correspondência entre o espaço das variáveis das juntas e o espaço das coordenadas finais do elemento de controle, conforme a ilustrado na Figura 9.

Figura 9 – Relação entre os espaço operacional e o espaço das juntas



Fonte: Autoral

3.1.4 Cinemática Direta

A cinemática direta permite determinar o posicionamento e a orientação do efetuador a partir dos ângulos das juntas e das dimensões geométricas dos elos, tomando como referência o *frame* da base do robô. Dessa forma, a cinemática direta é utilizada para localizar o efetuador com base na posição angular das articulações, através de um processo sequencial e progressivo, onde cada elo e junta é articulado desde a base até alcançar o elemento final de controle (CRAIG, 2006).

O equacionamento da cinemática direta utilizando apenas o formato matricial, como foi discutido na Seção 3.1.1, é adequado para sistemas mais simples (NIKU, 2020). Para configurações de robôs manipuladores mais complexos, com maior número de graus de liberdade, é necessário empregar a notação de D-H (DENAVIT; HARTENBERG, 1955).

A notação D-H tem como objetivo modelar as translações e rotações produzidas pelos elos e juntas, padronizando as coordenadas dos sistemas de referências das ligações espaciais por meio de um conjunto de passos a serem seguidos, os quais facilitam a obtenção da cinemática direta. Ela simplifica a modelagem dos parâmetros geométricos e facilita a derivação das equações que descrevem o movimento do robô. Embora existam outras convenções e métodos para representar a geometria e cinemática de um robô, a notação D-H é amplamente difundida na comunidade científica devido à sua eficiência e clareza (CRAIG, 2006).

O método D-H consiste em associar quatro parâmetros em cada junta do robô, bem como fixar o sistema de referência aos elos e às juntas da cadeia cinemática espacial. Assim, Craig (2006) define que, para representar a movimentação de uma junta em relação a outra, é necessário definir os parâmetros descritos abaixo:

- **Ângulo de rotação da junta** (θ_i): Ângulo de rotação medido sobre z_i entre o eixo x_{i-1} da junta atual em relação ao eixo x_i da junta seguinte.
- **Deslocamento do elo** (d_i): Distância medida ao longo do eixo z_i entre o eixo do elo atual x_{i-1} com o eixo do elo seguinte x_i.
- **Tamanho do elo** (a_i): Distância ao longo do eixo x_i entre o eixo da junta atual z_{i-1} e a junta seguinte z_i.
- **Ângulo de torção do elo** (α_i): Ângulo de torção medido sobre o eixo x_i entre o eixo da junta atual z_{i-1} e a junta seguinte z_i.

A transformação de uma junta em relação a junta anterior é dado pela notação ${}_{i-1}^{i}T$. A Notação D-H descreve essa transformação como o produto de quatro transformações elementares: uma rotação em torno do eixo z, uma translação ao longo do eixo z, uma translação ao longo do eixo x, e uma rotação em torno do eixo x, isso é descrito na Equação 3.14. Ao realizar a parametrização dos movimentações nos eixos x e z, o método modela as transformações espaciais em todos os eixos cartesianos do sistema de referência. Os parâmetros sugeridos pela notação são usados para construir a matriz de transformação homogênea genérica, conforme a Equação 3.15, onde $s_{\theta_i} = \sin(\theta_i)$, $c_{\theta_i} = \cos(\theta_i)$, $s_{\alpha_i} = \sin(\alpha_i)$ e $c_{\alpha_i} = \cos(\alpha_i)$. Para cada elemento da cadeia cinemática do manipulador, é necessário obter uma matriz de transformação

específica (SPONG *et al.*, 2020).

$$
{}_{i-1}^{i}T = Rot_{z,\theta_i} Trans_{z,d_i} Trans_{x,a_i} Rot_{x,\alpha_i} \tag{3.14}
$$

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

$$
= \begin{bmatrix} c_{\theta_i} & -s_{\theta_i} c_{\alpha_i} & s_{\theta_i} s_{\alpha_i} & a_i c_{\theta_i} \\ s_{\theta_i} & c_{\theta_i} c_{\alpha_i} & -c_{\theta_i} s_{\alpha_i} & a_i s_{\theta_i} \\ 0 & s_{\alpha_i} & c_{\alpha_i} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.15}
$$

Para definir os parâmetros D-H é necessário seguir uma série de etapas. Dessa maneira, é possível modelar a cinemática direta de um manipulador. O algoritmo a ser seguido inclui as seguintes etapas:

1. Numerar os elos e juntas do manipulador.
2. Definir os sistemas de coordenadas locais em cada elemento do manipulador. Cada corpo elementar da cadeia cinemática deve ser fixado em um *frame*. Cada sistema é numerado de acordo com o elo ao qual está associado.
3. Localizar os eixos das articulações. O eixo das juntas fará parte do sistema de coordenadas associado ao elo, convencionando-se que seja o eixo de coordenada z_i de acordo com a rotação das juntas.
4. Localizar a origem do sistema $\{i\}$ no ponto O_i onde a normal comum entre os eixos z_{i-1} e z_i intercepta z_i.
5. Estabelecer o eixo x_i do sistema de coordenadas locais. A partir da origem O_i, estabelecer o x_i ao longo da normal aos eixos z_{i-1} e z_i.
6. Encontrar os parâmetros D-H de cada elemento.
    - θ_i: Rotacionar o eixo x_{i-1} em torno de z_i até que fique paralelo ao eixo x_i.
    - d_i: Deslocar o eixo de coordenadas $\{i-1\}$ ao longo de z_{i-1} até a intersecção entre os eixos z_{i-1} e x_i.
    - a_i: Deslocar o eixo de coordenadas $\{i\}$ ao longo de x_i até a intersecção entre os eixos z_{i-1} e x_i.
    - α_i: Rotacionar o eixo z_{i-1} em torno de x_i até que fique paralelo ao eixo z_i.

7. Formar a matriz de transformação homogênea para cada *frame*, dada na Equação 3.15. Os parâmetros encontrados devem ser substituídos nas matrizes para obter as transformações homogêneas específicas para cada elemento.
8. Multiplicar em série as transformações específicas para obter a matriz homogênea de transformação total.

Após calcular as matrizes das diferentes coordenadas locais, é gerada uma matriz que relaciona as coordenadas das articulações desde a base até às coordenadas do efetuador. Essa matriz de transformação total é o passo inicial para implementar a análise e o controle preciso do manipulador robótico.

3.1.5 Cinemática Inversa

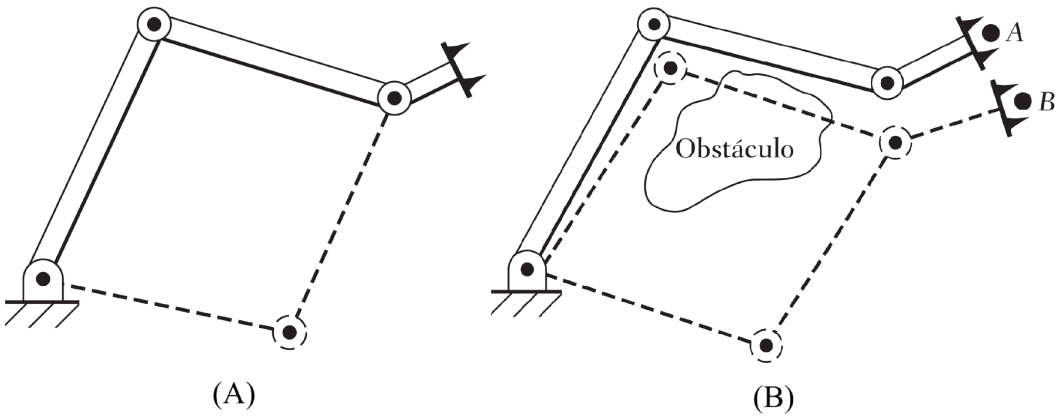
A cinemática inversa na robótica lida com a determinação das configurações angulares das juntas para alcançar uma posição e orientação desejada do elemento final de controle (CRAIG, 2006). Assim, a cinemática inversa permite controlar o manipulador robótico de forma mais intuitiva. Ao passo que, fornece uma especificação que relaciona as coordenadas desejadas no espaço de trabalho. Bem como, faz-se compreender quais as configurações do espaço das juntas correspondentes necessárias para alcançar a coordenadas desejada ao efetuador.

A cinemática inversa possui uma complexidade maior comparada à cinemática direta, já que envolve equações não lineares e por isso é entendida como um problema mal posto. Isto significa que, para o sistema, pode haver múltiplas soluções, nenhuma solução ou soluções não admissíveis para determinar a posição do efetuador final, e, por isso, não é possível encontrar uma solução de forma fechada (OGAWA; KANADA, 2010). A Figura 10, representa as múltiplas soluções de configurações do manipulador para alcançar um determinado ponto no espaço. Existem ainda situações em que a solução não é admissível por conta de obstáculos.

Além disso, fatores como limites de movimento das juntas, restrições físicas e colisões também devem ser considerados. O número de soluções depende do número de graus de liberdade do manipulador e os parâmetros de notação de D-H. Há algumas técnicas e abordagens tradicionais de solução da cinemática inversa, desde métodos analíticos ou soluções fechadas até técnicas numéricas, como o método da Matriz Jacobiano inverso e o método iterativo de otimização.

Para saber se existe solução, é necessário conhecer o espaço de trabalho do manipulador, o qual é definido como a região do espaço tridimensional no qual o manipulador é capaz de

Figura 10 – Representação da cinemática inversa de um robô manipulador com múltiplas soluções (A) e soluções inalcançáveis (B)



Fonte: (CRAIG, 2006) (adaptada)

posicionar seu efetuador e realizar uma tarefa. Então, se a coordenada de referência do efetuador está no espaço de trabalho, existe alguma solução para o problema da cinemática inversa, caso contrário o problema não é solucionável. Assim, um manipulador é considerado solucionável se existir um algoritmo que determine todos as possíveis múltiplas soluções dos conjuntos de variáveis das juntas (ROTH, 1976).

Tradicionalmente, as soluções da cinemática inversa são divididas entre: soluções na forma fechada e soluções numéricas. As soluções fechadas utilizam de conceitos das identidades trigonométricas e álgebra linear para resolver as equações não lineares que definem o problema. As soluções numéricas se baseiam na utilização de métodos iterativos para resolver o problema da não linearidade das equações. A diferença das métricas é dada pela precisão, já que para o caso da forma fechada o resultado é exato, enquanto a solução numérica o resultado é aproximado (CRAIG, 2006).

As soluções fechadas são muito mais restritivas em suas soluções e bem aplicáveis em soluções de manipuladores mais simples. Para sistemas com muitos graus de liberdade e mais complexos a solução numérica é mais aplicada (ZHANG; XIAO, 2019).

3.2 Aprendizado de Máquina

O Aprendizado de Máquina (AM) é um campo de estudo da Inteligência Artificial (IA) que permite aos computadores a capacidade de aprender e melhorar a execução de uma determinada tarefa, sem serem explicitamente programados. Isso é feito com base nas experiências passadas e no reconhecimento de padrões dos conjuntos de dados obtidos de um experimento. Para tanto, o AM envolve diversas áreas do conhecimento, como Probabilidade e Estatística,

Álgebra Linear, Ciências da Computação, Bioinformática, Reconhecimento de Padrões, entre outras (FACELI *et al.*, 2011).

Existem diferentes tipos de aprendizados de máquina, cada um com modelos característicos e abordagem específicas, classificados em Aprendizado Supervisionado, Aprendizado Não Supervisionado e o Aprendizado por Reforço. Para o aprendizado supervisionado, o algoritmo é treinado baseado em uma relação de entrada e saída obtida por um conjunto de dados rotulados. No aprendizado não supervisionado, o algoritmo é treinado usando um conjunto de dados não rotulados, ou seja, ocorre o aprendizado por meio do conhecimento dos padrões da entrada sem o uso das respostas corretas conhecidas. Por fim, no aprendizado por reforço, o algoritmo aprende por meio de recompensas e punições, gerando uma política que maximize as recompensas geradas (RUSSELL; NORVIG, 2016).

Os modelos preditivos têm como objetivo realizar previsões a partir de dados rotulados, ou seja, exemplos previamente conhecidos. Eles são divididos em dois tipos principais: classificação, quando a variável de saída assume valores discretos, e regressão, quando a saída é contínua. Em ambos os casos, o modelo busca aprender a partir de um conjunto de treinamento e depois generalizar para novos exemplos, definindo superfícies de decisão que separam as classes ou ajustam uma função de previsão (RUSSELL; NORVIG, 2016).

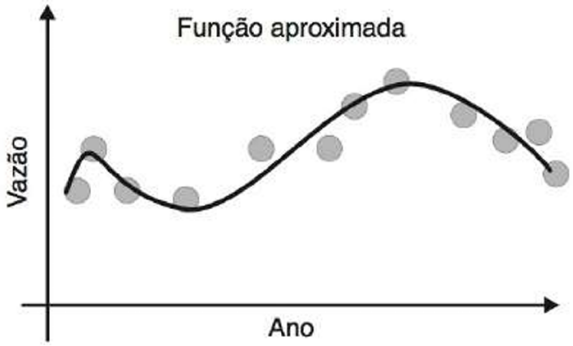
A regressão é uma técnica fundamental no aprendizado supervisionado, especialmente em problemas onde a variável de saída é contínua. O objetivo da regressão é aproximar uma função desconhecida $y_i = f(x) \in \mathbb{R}$, a partir de um conjunto de dados rotulados, permitindo a previsão de novos exemplos. Isso é obtido através da escolha de uma hipótese h, que deve minimizar o erro entre os valores reais e os valores previstos (FACELI *et al.*, 2011).

Outro aspecto importante da regressão é a sua aplicação em contextos com variáveis contínuas. Diferente da classificação, onde as saídas são discretas, a regressão busca prever valores em um intervalo contínuo (RUSSELL; NORVIG, 2016). As técnicas de regressão podem ser usadas para prever tendências futuras ou identificar padrões subjacentes em um conjunto de dados, caso do modelo de regressão da Figura 11 que aproxima uma função que relacione as variáveis ano e vazão de água de um rio.

3.2.1 Redes Neurais Artificiais

As Redes Neurais Artificiais (RNA) são modelos de aprendizado de máquina inspirados na neurociência, área da medicina que estuda o sistema nervoso e suas funcionalidades, a

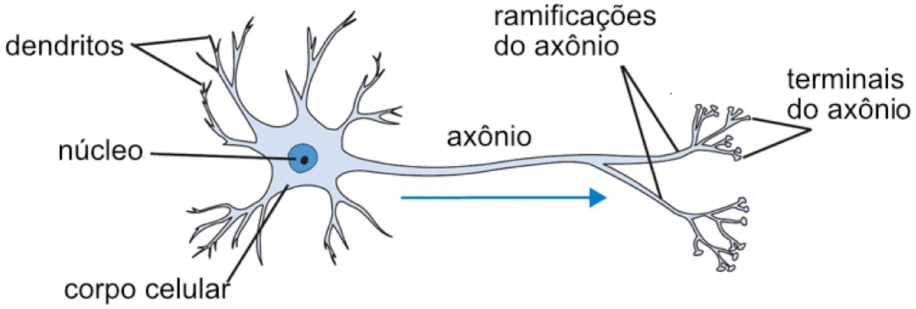
Figura 11 – Modelo de regressão para predizer vasão de água do rio a partir do atributo ano



Fonte: (FACELI *et al.*, 2011)

qual define que as atividades mentais ocorrem por meio das redes neurais (RUSSELL; NORVIG, 2016). Como ilustrado na Figura 12, o neurônio é composto de várias partes: a parte principal do corpo celular, na qual se localiza o núcleo; ramificações denominadas de dendritos, por onde chegam as informações; o axônio, dado por uma única extensão longa responsável por transmitir os sinais; e as sinapses, responsáveis por conectar os neurônios.

Figura 12 – Representação do neurônio biológico simplificado

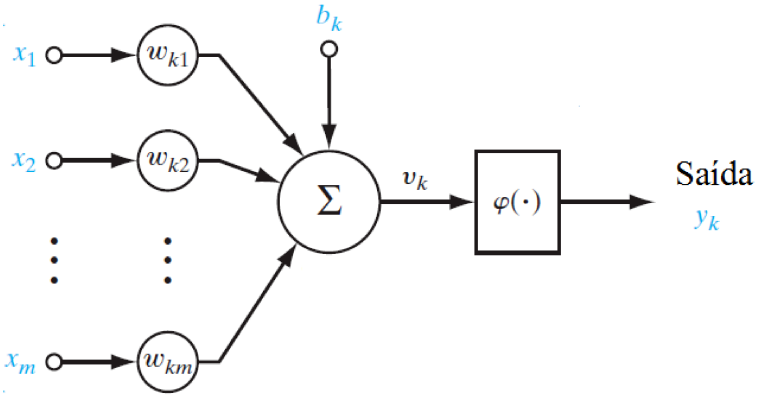


Fonte: (BEZERRA, 2016)

A Figura 13 representa um modelo simplificado do neurônio de McCulloch e Pitts (MP) desenvolvido em 1943 e compõe a unidade básica de uma RNA. Este busca modelar aspectos relevantes ao processamento de informação de um neurônio real. Nesta figura é possível observar que os dendritos são modelados como ramos contendo o canal de comunicação por onde fluem as informações de entrada $\{x_1, x_2, ..., x_m, b_k\}$. Bem como, define-se b_k como um *bias* ou limiar de ativação. A eficiência das conexões sinápticas da árvore dendrítica é ponderada por um fator ou peso $\{w_{k1}, w_{k2}, ..., w_{km}\}$, cuja função é modelar a força do sinal passada em cada ramo. O corpo celular possui a função de realizar o acúmulo energético por meio do somatório das entradas e pesos sinápticos $u_k = w_{k1}x_1 + w_{k2}x_2 + ... + w_{km}x_m - b_k$. Por fim, o axônio é modelado como aplicação de uma função de ativação $\varphi(\cdot)$ sobre u_k, gerando a saída do neurônio

y_k (HAYKIN *et al.*, 2009).

Figura 13 – Representação matemática de um neurônio artificial



Fonte: (HAYKIN *et al.*, 2009)

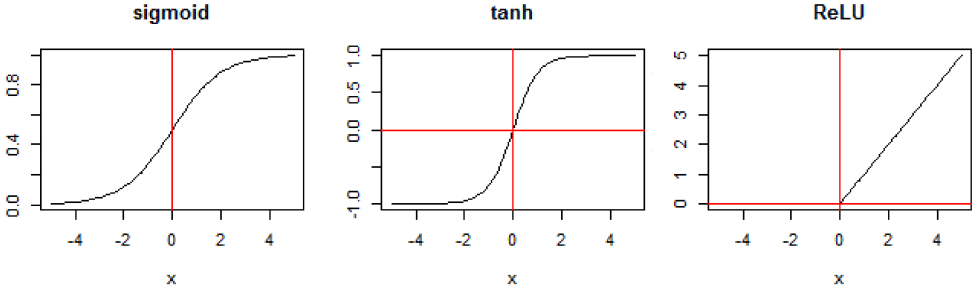
A rede Perceptron Simples (PS) foi o primeiro algoritmo de rede neural, proposto por Frank Rosenblatt em 1958. A rede PS é composta pelo neurônio MP e o processo de aquisição de conhecimento do algoritmo é dado por uma regra de aprendizado, a qual consiste de uma função que altera os pesos e do limiar de ativação (*bias*). A Equação

$$w(t+1) = w(t) + \eta e(t)x(t), \tag{3.16}$$

define a regra de aprendizado como uma relação entre o conhecimento atual $w(t)$, vetor de entrada $x(t)$, erro entre a saída desejada e a saída gerada pela rede $e(t)$ e o passo de aprendizado η.

As funções de ativação são componentes que introduzem não linearidade ao modelo neural, permitindo que ele capture padrões complexos. Diversas funções de ativação têm sido propostas na literatura. Entre as mais utilizadas, destacam-se a Tangente Hiperbólica (tanh), *Rectified Linear Unit* (ReLU) e a Sigmóide Logística, como ilustrado na Figura 14.

Figura 14 – Funções de ativações dos neurônios



Fonte: (METAQUOTES, 2020) (adaptada)

A função tanh é definida no intervalo $[-1,1]$ e possui a propriedade de centralizar os dados. O gráfico dessa função é mostrado na Figura 14, e sua equação é dada por:

$$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}. \tag{3.17}$$

Essa característica permite que os valores de saída fiquem distribuídos em torno de zero, o que pode acelerar o processo de aprendizado. No entanto, a função tanh pode sofrer com o problema de gradientes pequenos, o que dificulta o ajuste dos pesos durante o treinamento de redes muito profundas (FACELI *et al.*, 2011).

A função sigmoide logística, semelhante à tangente hiperbólica, é uma função suave que mapeia as entradas para o intervalo $[0,1]$, conforme ilustrado na Figura 14. Sua equação é dada por:

$$S(x) = \frac{1}{1 + e^{-x}}. \tag{3.18}$$

No entanto, essa função também enfrenta o problema dos gradientes desaparecendo, já que nas extremidades a função tende a se aproximar de zero, o que pode paralisar o processo de aprendizado em camadas mais profundas (BENGIO *et al.*, 2016).

Por outro lado, a função de ativação ReLU tornou-se a mais popular em redes neurais profundas devido à sua simplicidade e eficácia em lidar com o problema do gradiente próximos de zero. Seu gráfico é ilustrado na Figura 14, e sua equação é dada por:

$$ReLU(x) = \begin{cases} 0, & \text{se } x \leq 0 \\ x, & \text{se } x > 0 \end{cases}. \tag{3.19}$$

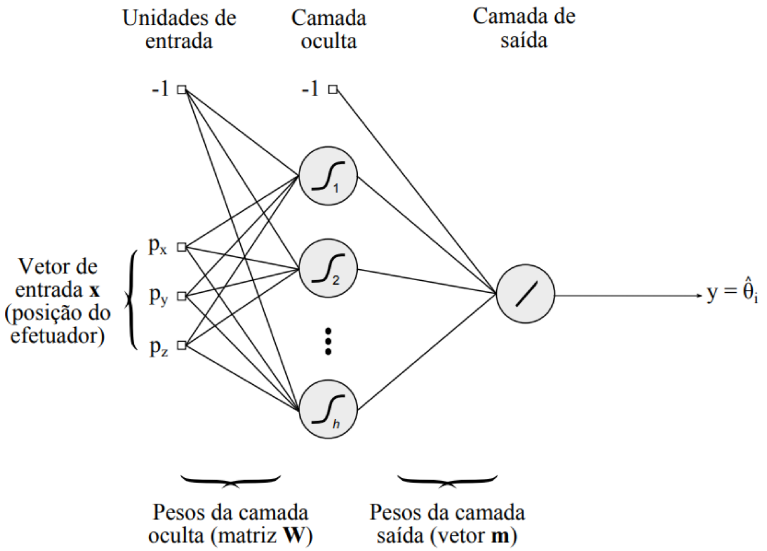
Por retornar zero para valores negativos e a própria entrada para valores positivos, o gradiente não se satura, tornando o processo de aprendizado mais eficiente em redes profundas. No entanto, a ReLU pode enfrentar o problema dos "neurônios mortos", que ocorre quando valores negativos persistem e impedem a atualização de determinados pesos (BENGIO *et al.*, 2016).

Uma RNA é composta por um ou mais neurônios, os quais podem estar distribuídos em uma ou mais camadas de neurônios interconectados. Nesse caso, os neurônios recebem como entrada valores oriundos das saídas dos neurônios das camadas anteriores e enviam seus valores para os neurônios seguintes à sua camada. Esta rede de múltiplas camadas de neurônios é denominada *Multi Layer Perceptron* (MLP).

A Figura 15 representa uma rede MLP que possui neurônios artificiais, uma camada oculta e uma camada de saída. Uma rede neural do tipo MLP possui, além do neurônio de MP, os seguintes elementos:

- **Unidade de entrada**: É a primeira camada da rede responsável por captar as entradas e enviar para as camadas seguintes.
- **Camadas ocultas**: São camadas intermediárias compostas por vários neurônios e são responsáveis por receber as saídas de outros neurônios, processá-las e propagá-las para as camadas seguintes.
- **Camada de saídas**: É a última camada da rede neural e é responsável por fornecer a saída do modelo.

Figura 15 – Rede neural de múltiplas camadas treinando a cinemática inversa de um manipulador



Fonte: (MELO, 2015)

Para esse tipo de rede há uma complexidade maior nos seus parâmetros e na definição da regra de aprendizado. Nesse caso, os pesos da rede são treinados por meio de um algoritmo de retropropagação do erro (*error backpropagation*) definido por Rumelhart *et al.* (1986) com o objetivo de minimizar o erro de predição da rede. O processo de retro propagação envolve as fases de propagação para frente e a propagação para trás. Na propagação para frente, os dados são inseridos na rede percorrendo as camadas ocultas até a camada de saída. Após calcular o gradiente do erro em relação aos pesos das conexões e dos bias, os novos pesos são ajustados desde a camada de saída em direção à camada de entrada.

3.2.2 Modelos Globais e Locais

Em aprendizado de máquina, os modelos globais e locais representam duas abordagens distintas para a modelagem de funções e padrões a partir de dados. Essas abordagens

refletem como o algoritmo lida com a relação entre as variáveis de entrada e saída, seja capturando padrões globais em todo o conjunto de dados ou ajustando modelos especializados para diferentes regiões do espaço de entrada (MELO, 2015). A escolha entre um modelo local ou global depende da complexidade do problema e das características dos dados. Ambos os métodos possuem vantagens e limitações, sendo aplicados conforme a natureza da tarefa de predição.

Modelos globais buscam ajustar um único modelo a todos os dados disponíveis, assumindo que a relação entre as variáveis de entrada e saída pode ser capturada por uma função abrangente. Nessa abordagem, o algoritmo tenta otimizar o desempenho em todo o conjunto de dados, ignorando variações locais que possam existir. Isso é adequado quando a função que rege os dados é relativamente simples ou linear, ou quando há uma grande quantidade de dados disponível para capturar padrões globais (BOTTOU; VAPNIK, 1992).

Exemplos comuns de modelos globais incluem regressão linear, redes neurais profundas e SVM. Esses modelos tratam os dados de maneira uniforme e buscam encontrar padrões que generalizem bem para todo o espaço de entrada. O objetivo é minimizar o erro em toda a distribuição de dados, sem focar nas variações locais (FACELI *et al.*, 2011). No entanto, uma limitação dos modelos globais é a dificuldade de capturar detalhes ou variações significativas em regiões específicas do espaço de entrada, o que pode ocorrer em problemas altamente não lineares. Nesses casos, pode ocorrer o subajuste (*underfitting*), onde o modelo não consegue capturar a complexidade dos dados, levando a um desempenho inferior (BOTTOU; VAPNIK, 1992).

Por outro lado, modelos locais dividem o espaço de entrada em regiões menores, ajustando submodelos distintos para cada uma dessas áreas. A premissa é que uma única função global pode não ser adequada para representar a complexidade dos dados, especialmente quando o comportamento da relação entre entrada e saída varia substancialmente entre diferentes regiões. Modelos locais são ajustados em subconjuntos dos dados, oferecendo maior flexibilidade para lidar com variações e não linearidades locais (MELO, 2015).

Em termos de aplicações, os modelos globais são amplamente utilizados em tarefas como predição de séries temporais, classificação de imagens e predição de valores contínuos em grandes bases de dados. Já os modelos locais são frequentemente aplicados em problemas como reconhecimento de padrões em sistemas dinâmicos, incluindo a cinemática inversa de robôs (MELO, 2015) e sistemas de recomendação, onde a personalização baseada em vizinhanças locais pode melhorar a precisão das predições.

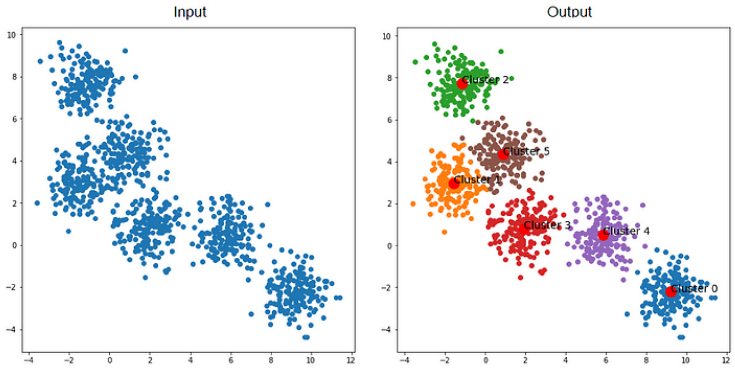
Bottou e Vapnik (1992) descrevem essa abordagem como particularmente útil em problemas onde há uma variação substancial no comportamento dos dados em diferentes regiões, e onde um único modelo global seria incapaz de capturar essas diferenças. No entanto, modelos locais podem ser mais propensos ao sobreajuste (*overfitting*) se houver poucos dados disponíveis ou se o modelo for muito sensível às flutuações locais dos dados.

3.2.3 Algoritmo de Agrupamento K-means

O aprendizado não supervisionado envolve técnicas que buscam identificar padrões em dados sem a necessidade de rótulos ou supervisão explícita. Ao contrário da classificação e da regressão, no agrupamento (*clustering*), a definição das categorias do modelo não é previamente conhecida através de uma base de treinamento. Entre essas técnicas, o objetivo do *clustering* é categorizar ou agrupar amostras de dados em grupos (*clusters*), onde dados semelhantes são agrupados em uma mesma categoria, enquanto dados distintos são separados em diferentes grupos (JAIN, 2010). Essa abordagem é amplamente aplicada em áreas como reconhecimento de padrões, mineração de dados, segmentação de imagens e compressão de dados.

O método *K-means* é uma técnica de aprendizado não supervisionado amplamente utilizada para a tarefa de clusterização de dados. Seu objetivo principal é particionar um conjunto de dados em k grupos distintos, conforme ilustrado na Figura 16, minimizando a variabilidade dentro de cada grupo e maximizando as diferenças entre eles. O *K-means* busca encontrar centróides (centros dos *clusters*) que representem o conjunto de dados de forma compacta, agrupando os dados com base em sua proximidade a esses centróides (JAIN, 2010).

Figura 16 – Agrupamento de classes com a técnica *K-means*



Fonte: (ICHI, 2020)

O algoritmo *K-means* inicia com a definição do número de *clusters* k, ao passo que os k centróides são selecionados de forma aleatória ou heurística. Cada ponto de dados é, então,

associado ao centróide mais próximo, formando os *clusters* iniciais. O processo é iterativo: os centróides são recalculados como a média dos pontos atribuídos a cada *cluster*, e os pontos são reatribuídos ao centróide mais próximo após cada recalculação. Esse ciclo continua até que os centróides se estabilizem ou que a variação de um critério de convergência seja inferior a um determinado limiar, ou ainda até que um número máximo de iterações seja atingido (BISHOP, 2006).

A equação que define a distância euclidiana, comumente usada como métrica no *K-means*, é expressa como:

$$d(x_i, \mu_j) = \sqrt{\sum_{n=1}^{p} (x_{in} - \mu_{jn})^2}, \tag{3.20}$$

onde x_i é o ponto de dados, μ_j é centróide do *cluster* j, e p representa o número de características de cada ponto de dados (FACELI *et al.*, 2011).

Basicamente, o *k-means* busca minimizar a função objetivo das variâncias intra-*cluster*, que é dada pela soma das distâncias quadradas entre os pontos e seus respectivos centróides. A função é representada por:

$$J = \sum_{i=1}^{k} \sum_{x \in C_i} ||x - \mu_i||^2. \tag{3.21}$$

Onde C_i representa o conjunto de pontos pertencentes ao *cluster* i e μ_i é o centróide do *cluster* i. O objetivo do algoritmo é minimizar J, o que implica em *clusters* mais compactos e bem definidos.

4 MATERIAIS E MÉTODOS

Neste capítulo, é apresentado o desenvolvimento prático do trabalho proposto, juntamente com os métodos e materiais que viabilizam a sua execução. Para isso, foram adotadas uma série de atividades que direcionaram o progresso e a obtenção de resultados. Essas atividades foram executadas na seguinte ordem:

- **Definição do problema**: estudo da literatura de robótica envolvendo a cinemática direta e o problema da aproximação da cinemática inversa, a fim de modelar a estrutura geométrica do manipulador didático utilizado deste trabalho e empregá-lo na simulação para gerar o conjunto de dados.
- **Geração do padrão de treinamento**: Criação de um conjunto de dados abrangendo o espaço de trabalho do robô didático, mapeando a relação entre o espaço das juntas e o espaço operacional para aproximar a cinemática inversa.
- **Particionamento e Seleção de Hiperparâmetros**: Os dados foram divididos em conjuntos de treinamento e validação para cada modelo de AM, utilizando o método de validação cruzada com 5 iterações n_r para uma avaliação do desempenho. Em seguida, os hiperparâmetros de cada modelo foram selecionados usando os conjuntos de treinamento e validação, identificando quais necessitavam de ajustes e determinando valores possíveis para cada um. Ao final, foram definidas duas abordagens para resolver o problema de cinemática inversa.

4.1 Robô Manipulador Didático

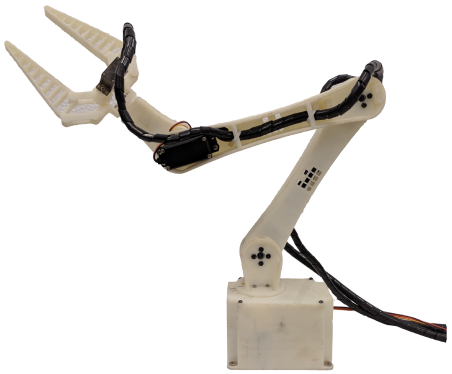
O braço robótico, base deste trabalho, é mostrado na Figura 17. Trata-se de um robô do tipo manipulador, desenvolvido como um módulo didático pelo grupo de pesquisa em robótica do IFCE[1]. Sua fabricação foi realizada utilizando uma impressora 3D com filamento de *Polylactic Acid* (PLA), um material de baixo custo amplamente empregado em impressões tridimensionais.

A estrutura mecânica do braço é composta por cinco partes interligadas, como ilustrado na Figura 18: a base (elo 0), o braço (elo 1), o antebraço (elo 2), o punho (elo 3) e a garra (elo 4). As dimensões de cada elo do manipulador estão descritas na Tabela 1.

O robô possui cinco juntas rotacionais que permitem o movimento relativo entre os

[1] https://robotica.ifce.edu.br/

Figura 17 – Robô Manipulador Didático



Fonte: Autoral

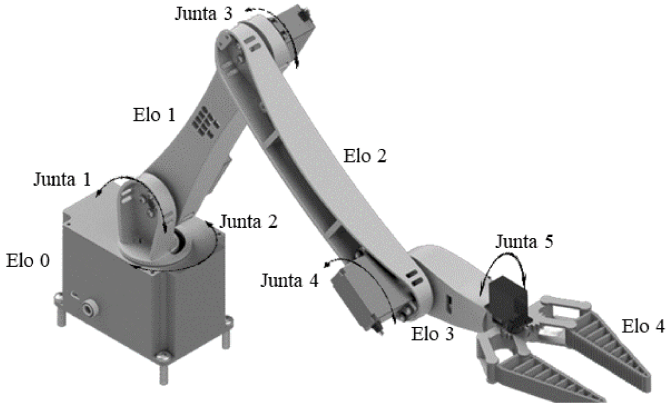
Tabela 1 – Dimensões dos elementos do manipulador didático

	Comprimento (cm)	Largura (cm)	Altura (cm)
Base	9,9	6,4	10,0
Braço	18,0	2,8	4,0
Antebraço	18,0	2,8	4,0
Punho	6,0	3,0	3,3
Garra	12,0	6,6	2,0

Fonte: Autoral

elos. Portanto, trata-se de uma cadeia cinemática constituída por 5 GDL, definindo o robô como um modelo 5R.

Figura 18 – Representação da estrutura do manipulador Robótico Didático

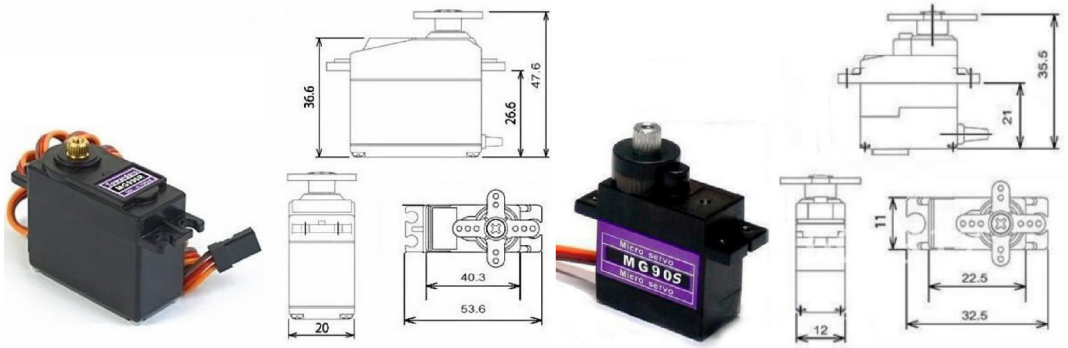


Fonte: (IFCE, 2022) (adaptada)

4.1.1 Atuadores

Os atuadores selecionados para gerar a força motriz que movimenta o braço são os servomecanismos MG996R[2] e os micro servos MG90S[3], ilustrados na Figura 19. Para produzir o movimento no manipulador, são utilizados quatro atuadores do tipo MG996R e dois MG90S, totalizando seis unidades de servo motores.

Figura 19 – (a) servo motor MG996R (b) micro servo motor MG90S



(a) (b)

Fonte: (ALLDATASHEET, 2003) (adaptada)

A escolha desses motores deve-se à simplicidade no controle de posição angular, ao tamanho que facilita o acoplamento à estrutura robótica, à precisão no controle da posição e da velocidade angulares, além de apresentarem uma faixa de operação de torques satisfatória para atender às demandas do sistema robótico.

As especificações técnicas dos atuadores estão detalhadas na Tabela 2, juntamente com suas respectivas juntas de fixação. Todos os motores foram instalados no manipulador de modo que seus eixos de rotação sejam paralelos ou antiparalelos aos eixos das juntas. Essa disposição facilita a modelagem da cinemática do manipulador e minimiza as perdas de torque.

Tabela 2 – Especificações técnicas dos atuadores

Modelo do Motor	Junta	Torque ($Kg \cdot cm$)	Tensão (V)	Rotação ($^\circ$)	Corrente (mA)
PowerPro MG996R	1, 2, 3 e 4	$9,4 - 11,0$	$4,8 - 7,2$	$0 - 180$	$500 - 900$
Microservo MG90S	5 e Pinça	$1,8 - 2.2$	$4,8 - 6,0$	$0 - 180$	$140 - 650$

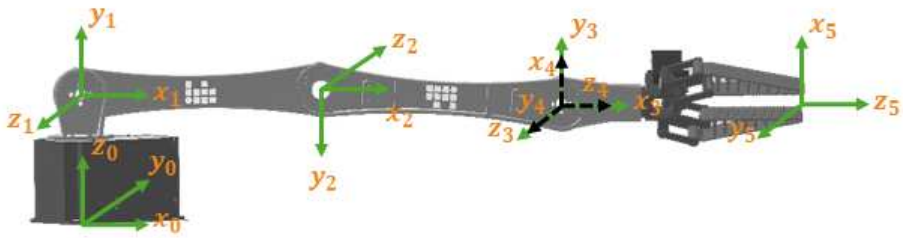
Fonte: Autoral

[2] https://www.alldatasheet.com/datasheet-pdf/pdf/1131873/ETC2/MG996R.html
[3] https://www.alldatasheet.com/datasheet-pdf/pdf/1132104/ETC2/MG90S.html

4.2 Cinemática Direta do Manipulador Didático

Nesse trabalho, o método de solução da cinemática inversa do manipulador didático passa inicialmente pela modelagem da cinemática direta. Nesse sentido, utilizando as regras e etapas do método de Denavit-Hartenberg (Seção 3.1.4), são fixados os sistemas de referência locais em cada elemento do robô, conforme ilustrado na Figura 20.

Figura 20 – Sistema de coordenadas locais do manipulador didático



Fonte: Autoral

Devido ao mecanismo robótico ser constituído por 5 GDL de juntas rotativas, a cadeia cinemática possui cinco variáveis de juntas θ_i, para $i = 1, ..., 5$. A descrição da cinemática direta desse sistema está apresentada na Tabela 3, na qual define os parâmetros D-H de cada elemento do manipulador.

Tabela 3 – Parâmetros de Denavit-Hartenberg do Manipulador Didático

i	θ (°)	a (cm)	d (cm)	α (°)
1	θ_1	0	Elo 0	90
2	θ_2	Elo 1	0	180
3	θ_3	Elo 2	0	-180
4	$\theta_4 + 90$	0	0	90
5	θ_5	0	Elo 3 + Elo 4	0

Fonte: Autoral

A partir dos parâmetros D-H da Tabela 3, aplica-se a Equação 3.15 para construir as cinco matrizes de transformação homogênea específicas que relacionam os sistemas de coordenadas locais de um elo em relação ao elo anterior correspondente.

Posteriormente, essas cinco matrizes de transformação homogênea específicas são multiplicadas para obter uma matriz total que relaciona o sistema de referência da base ao sistema

de referência do efetuador final, conforme ilustrado na Equação 4.1.

$$T_5^0 = T_1^0 T_2^1 T_3^2 T_4^3 T_5^4 = \begin{bmatrix} R_{11} & R_{12} & R_{13} & p_x \\ R_{21} & R_{22} & R_{23} & p_y \\ R_{31} & R_{32} & R_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.1}$$

Sabendo que $s_{\theta_i} = \sin(\theta_i)$ $c_{\theta_i} = \cos(\theta_i)$, pode-se definir os elementos desta matriz:

$$R_{11} = s_{\theta_1} s_{\theta_5} - s_{\theta_2-\theta_3+\theta_4} c_{\theta_1} c_{\theta_5} \tag{4.2}$$

$$R_{12} = s_{\theta_1} c_{\theta_5} + s_{\theta_5} s_{\theta_2-\theta_3+\theta_4} c_{\theta_1} \tag{4.3}$$

$$R_{13} = c_{\theta_1} c_{\theta_2-\theta_3+\theta_4} \tag{4.4}$$

$$p_x = 18(c_{\theta_2} + c_{\theta_2-\theta_3} + c_{\theta_2-\theta_3+\theta_4}) c_{\theta_1} \tag{4.5}$$

$$R_{21} = -s_{\theta_1} s_{\theta_2-\theta_3+\theta_4} c_{\theta_5} - s_{\theta_5} c_{\theta_1} \tag{4.6}$$

$$R_{22} = s_{\theta_1} s_{\theta_5} s_{\theta_2-\theta_3+\theta_4} - c_{\theta_1} c_{\theta_5} \tag{4.7}$$

$$R_{23} = s_{\theta_1} c_{\theta_2-\theta_3+\theta_4} \tag{4.8}$$

$$p_y = 18(c_{\theta_2} + c_{\theta_2-\theta_3} + c_{\theta_2-\theta_3+\theta_4}) s_{\theta_1} \tag{4.9}$$

$$R_{31} = c_{\theta_5} c_{\theta_2-\theta_3+\theta_4} \tag{4.10}$$

$$R_{32} = -s_{\theta_5} c_{\theta_2-\theta_3+\theta_4} \tag{4.11}$$

$$R_{33} = s_{\theta_2-\theta_3+\theta_4} \tag{4.12}$$

$$p_z = 18 s_{\theta_2} + 18 s_{\theta_2-\theta_3} + 18 s_{\theta_2-\theta_3+\theta_4} + 10 \tag{4.13}$$

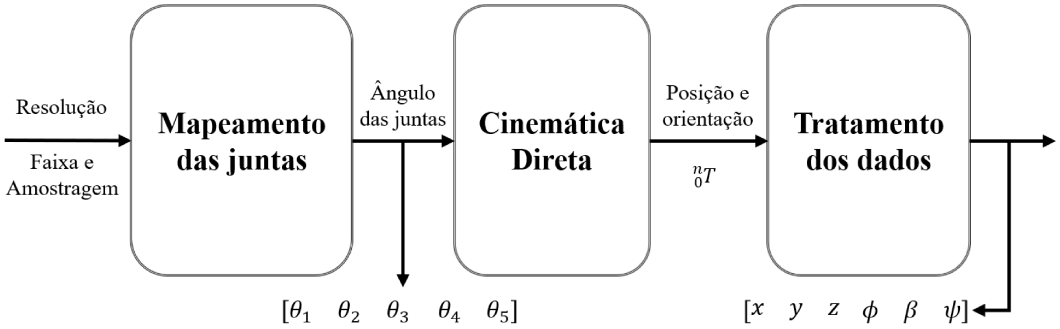
4.3 Geração e Pré-processamento dos Padrões de Treinamento

A criação de uma base de dados dos padrões de treinamento é uma etapa crucial no desenvolvimento de sistemas de AM baseados em aprendizado supervisionado. Para que o modelo aprenda de forma eficaz a realizar as tarefas desejadas, é necessário expor o agente a uma ampla gama de exemplos que ilustrem a relação entre as entradas e as saídas corretas (RUSSELL; NORVIG, 2013). Ao estabelecer uma base de dados, garantimos que o agente desenvolva uma função de mapeamento generalizável, essencial para seu desempenho em situações reais e variadas.

O método para aproximar a cinemática inversa do robô didático descrito neste trabalho começa com o conhecimento prévio da cinemática direta (Seção 4.2). A modelagem do manipulador, apresentada na Equação 4.1, foi então utilizada para a geração dos padrões de

treinamento dos algoritmos de AM. Nesse sentido, a criação dos padrões de treinamento dos modelos foi realizada pelo mapeamento das relações entre o espaço operacional e o espaço das juntas que circundam o manipulador. Através dessa relação, é possível criar uma base de dados que permita representar todo o volume de trabalho e o comportamento dos ângulos das juntas do robô. A Figura 21 ilustra o fluxograma necessário para o mapeamento dos espaços proposto neste trabalho, destaca-se os principais passos e decisões envolvidos no processo.

Figura 21 – Fluxograma da geração dos padrões de treinamento



Fonte: Autoral

A geração dos padrões do espaço das juntas é estabelecida por meio da resolução, que define a faixa de trabalho e o número de amostras dos ângulos de cada junta (NUNES, 2016). Para garantir que os dados gerados reflitam a realidade do manipulador, são aplicadas restrições ao mapeamento do espaço das juntas. Devido à presença de limites físicos nos ângulos das juntas dos motores, foram estabelecidas as faixas de atuação dos ângulos de cada junta, conforme demonstrado na Tabela 4. Essas restrições condicionam a forma do volume de trabalho final do manipulador, impossibilitando-o de alcançar determinadas poses no espaço operacional (MELO, 2015).

Tabela 4 – Resolução do espaço das juntas

Junta	Faixa de ângulo (°)	Número de amostras por GDL	Resolução (graus/pontos)
1	$0 \leq \theta_1 \leq 120$	25	5
2	$0 \leq \theta_2 \leq 120$	11	12
3	$0 \leq \theta_3 \leq 120$	11	12
4	$0 \leq \theta_4 \leq 120$	11	12
5	$\theta_5 = 0$	1	1

Fonte: Autoral

Cabe destacar que optou-se por mapear somente o conjunto de ângulos, excluindo a influência da junta do pulso. Apesar de o sistema robótico possuir 5 GDL, considerou-se θ_5 como um ângulo constante. Essa restrição deve-se ao fato de que o último grau de liberdade não

influencia a posição do efetuador, mas sim sua orientação. E dado que o objetivo do trabalho é desenvolver um algoritmo capaz de posicionar o efetuador no espaço de trabalho desejado sem considerar sua orientação, a inclusão desta junta resultaria em redundâncias adicionais na base de dados e aumentaria o tempo de processamento dos algoritmos de aprendizado de máquina (NUNES, 2016).

Para que o modelo aprenda a cinemática inversa, é necessário determinar o número de amostras de treinamento dos ângulos das juntas. Dependendo da resolução adotada, o número de redundâncias, a carga computacional e o tamanho da base de dados podem aumentar, o que pode diminuir a capacidade do modelo de aprender e generalizar. Além disso, uma base de dados muito grande ou muito pequena não necessariamente resulta em uma redução do erro ou em uma melhoria no aprendizado. Encontrar o número mínimo de amostras dos ângulos é, portanto, uma etapa essencial na metodologia do trabalho (LIM; LEE, 2019).

Adotou-se um número de amostras diferentes para cada ângulo, conforme ilustrado na Tabela 4. Observou-se que, devido à estrutura geométrica do braço, θ_1 possui maior relevância para os eixos x e y, tornando pertinente explorar um número maior de amostras para este grau de liberdade. Em contrapartida, os demais ângulos têm mais influência no eixo z e por isso suas amostras foram mais distribuídas entre si.

A geração dos dados que mapeiam o espaço operacional do manipulador deu-se pela aplicação dos ângulos indicados na Tabela 4, na modelagem da cinemática direta. Nesse processo, substituíram-se os valores dos ângulos das juntas, obtidos anteriormente no mapeamento, na matriz de transformação homogênea calculada por meio da notação D-H (Equação 4.1), o que permitiu a obtenção do conjunto de matrizes de transformações homogêneas alcançáveis pelo efetuador.

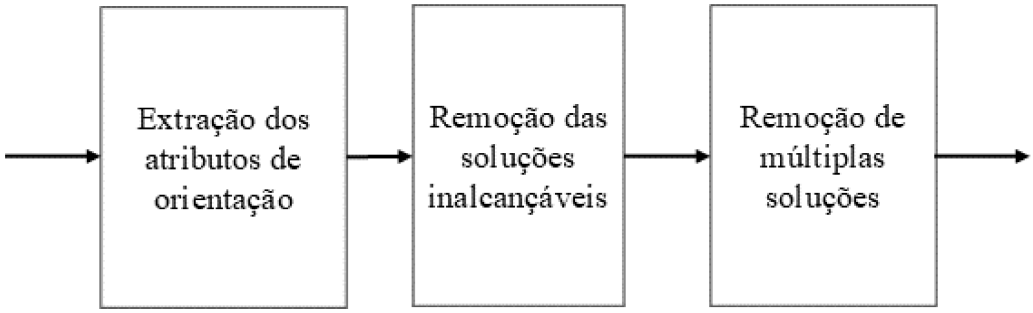
Por fim, gerou-se uma base de dados brutas com 33275 observações. As amostras foram estruturados no formato, onde as linhas contêm todas as amostras e as colunas compõem o vetor de atributos. Esses dados incluem as saídas dos ângulos das juntas e os atributos compostos dos elementos das matrizes de transformação homogênea alcançadas pela garra do robô. Os dados foram armazenados, no que se definiu como a base de dados bruta, no formato de entrada e saída padrão expressos pelo vetor $[\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, x, y, z, R_{11}, R_{12}, R_{13}, R_{21}, R_{22}, R_{23}, R_{31}, R_{32}, R_{33}]$.

4.3.1 Tratamento da Base de Dados

Os algoritmos de AM desenvolvidos com o intuito de predizer, por meio da extração de conhecimento de bases de dados, têm seu desempenho diretamente afetado pela qualidade dos dados. A ausência de tratamento adequado desses dados pode dificultar o uso dos algoritmos, gerando a construção de modelos pouco fiéis à realidade, aumento dos custos computacionais e dificuldade do modelo em entender os padrões de treinamento (FACELI *et al.*, 2011).

Para mitigar esses problemas, técnicas de pré-processamento de dados são frequentemente empregadas para tornar os dados mais adequados ao uso dos algoritmos. Uma boa técnica de tratamento dos dados não apenas melhora a acurácia dos modelos, mas também assegura a reprodutibilidade e a robustez dos experimentos conduzidos (FACELI *et al.*, 2011). Desta forma, na Figura 22 é ilustrado os passos seguidos no pré-processamento da base de dados, que inclui a extração de atributos relevantes, a remoção dos pontos pertencentes ao semi-plano negativo de z, e a eliminação de redundâncias.

Figura 22 – Diagrama do pré-processamento da base de dados



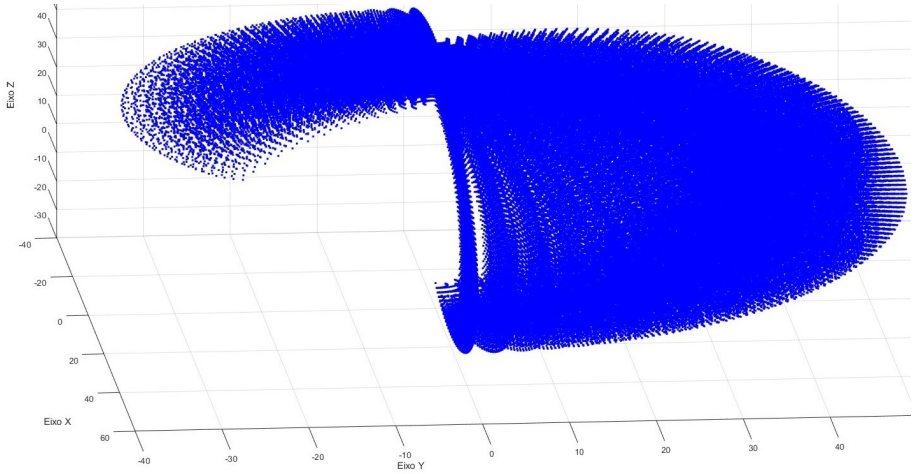
Fonte: Autoral

Os atributos de entrada dos modelos de AM são elementos fundamentais, pois representam as variáveis ou características extraídas dos dados brutos, nas quais o modelo se baseia para aprender e fazer previsões. A escolha e a qualidade desses atributos influenciam diretamente o desempenho do modelo. A extração e seleção dos atributos é, portanto, uma etapa crucial do pré-processamento da base de dados.

Nesse contexto, inicialmente, considerando o objetivo de mapear a cinemática inversa do manipulador, foram definidos como atributos as coordenadas cartesianas (x, y, z) que compõem a base de dados bruta. O volume de trabalho delimitado pelo robô manipulador didático é ilustrado nas Figuras 23 e 24. Em cada amostra da base de dados construída, é plotada

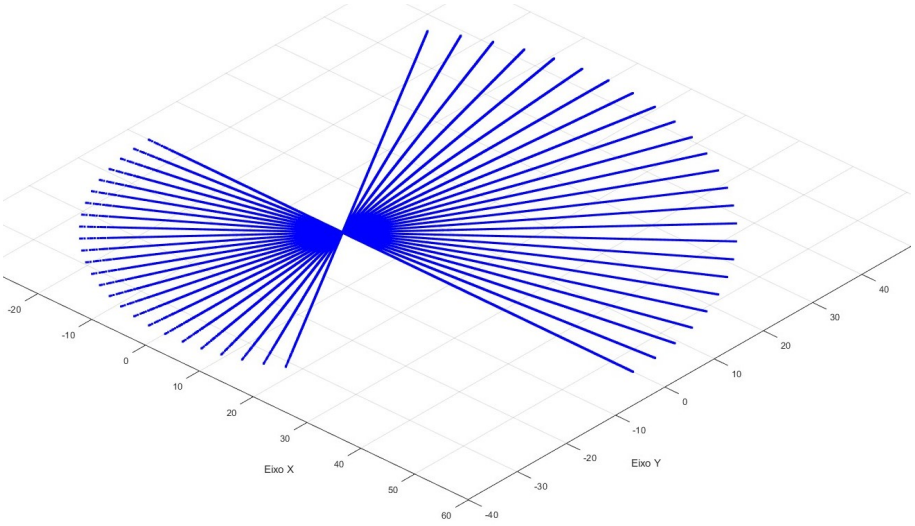
a posição do efetuador no espaço tridimensional, sem analisar a orientação associada.

Figura 23 – Volume de trabalho do manipulador didático



Fonte: Autoral

Figura 24 – Vista superior do volume de trabalho do manipulador didático



Fonte: Autoral

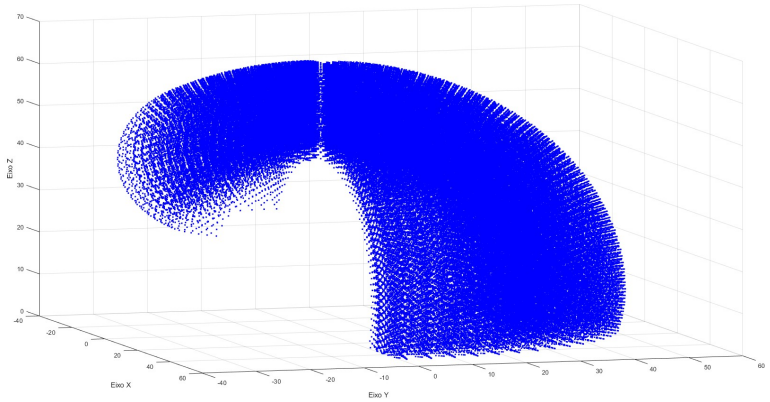
Embora o foco do trabalho seja mapear a posição no espaço de trabalho, houve a necessidade de realizar uma extração de atributos relacionados à orientação, provenientes dos padrões da base de dados que definem os elementos da matriz de transformações homogêneas mapeadas. Para tanto, optou-se por adicionar como atributos os ângulos $Roll(\phi)$, $Pitch(\beta)$ e $Yaw(\psi)$ para expressar a orientação do efetuador (DEMBY'S *et al.*, 2019). Isso exigiu uma transformação dos elementos de rotação da base de dados bruta $(R_{11}, R_{12}, R_{13}, R_{21}, R_{22}, R_{23}, R_{31}, R_{32}, R_{33})$. Definindo as entradas do modelo como o vetor padrão de atributos dado por $[x, y, z, \phi, \beta, \psi]$.

Ao lidar com a cinemática inversa, é importante considerar os aspectos não lineares

das equações que levam a singularidades, entendidas como soluções mal postas (Seção 3.1.5). Assim, para encontrar a posição do efetuador, podem existir múltiplas soluções, nenhuma solução ou soluções não admissíveis. Essa análise é particular para cada manipulador e envolve conhecer as soluções do espaço de trabalho e, por consequência, a base de dados.

A solução de pontos inalcançáveis refere-se a situações em que não é possível encontrar configurações de juntas que permitam ao manipulador alcançar uma posição ou orientação desejada no espaço de trabalho. Quando as equações não têm solução, ou as soluções obtidas não estão dentro dos limites operacionais do manipulador, o ponto desejado é considerado inalcançável (MELO, 2015). Na Figura 23, é possível observar regiões que não correspondem ao volume de trabalho completamente válido, pois há pontos inacessíveis pelo manipulador. Isso ocorre especialmente nos pontos pertencentes ao semiplano negativo do eixo Z, região que representa o espaço ocupado pelo suporte do robô e a superfície onde ele está fixado.

Figura 25 – Limitação física no eixo z do manipulador didático



Fonte: Autoral

A Tabela 5 descreve a faixa de atuação dos eixos cartesianos no espaço operacional. Para contornar essa problemática, cada amostra da base de dados é submetida a um filtro que restringe o espaço operacional do manipulador. Esse filtro verifica, para cada ponto do espaço de trabalho, se há uma componente no semiplano negativo do eixo z. Além disso, em todos as amostras da base de dados é avaliado se a posição do sistema de referência local de cada elo viola essa regra. Os pontos que se mostram inalcançáveis são removidos da base de dados.

Após avaliar a existência de uma solução, é necessário analisar sua multiplicidade, já que um conjunto diferente de ângulos de juntas pode permitir ao manipulador atingir a mesma posição. Uma amostra da base de dados é considerada redundante quando seus atributos x, y e z têm valores muito semelhantes a outras amostras da base de dados. A presença dessas

Tabela 5 – Faixa de atuação do espaço operacional

Eixos cartesiano	Mínimo (cm)	Máximo (cm)
x	-36	54
y	-36	54
z	0	54

Fonte: Autoral

observações redundantes no conjunto de dados cria uma falsa impressão para o modelo de que o perfil desses dados é mais importante do que outros. Isso ocorre porque essas amostras participam mais vezes no processo de ajuste dos parâmetros, contribuindo mais do que outras no aprendizado do algoritmo. Esse fenômeno resulta em um aumento do tempo de treinamento do modelo e em algoritmos menos precisos (FACELI *et al.*, 2011).

Para eliminar os pontos redundantes, a base de dados passa por uma segunda filtragem que verifica e remove padrões de treinamento semelhantes. Nesse contexto, pontos redundantes são definidos como aqueles que estão próximos uns dos outros no espaço operacional e como posições angulares distantes no espaço das juntas. A fim de calcular a distância entre cada amostra e as demais da base de dados, utiliza-se a equação da Distância Euclidiana:

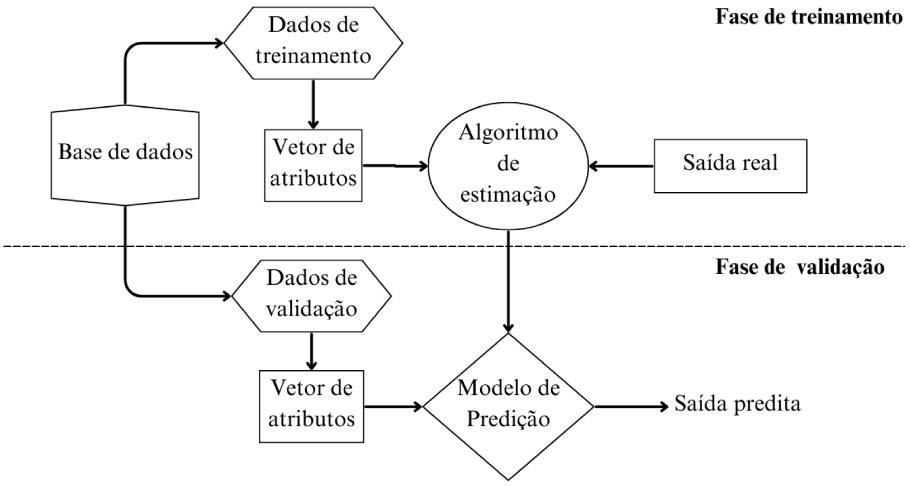
$$d = \sqrt{(x_i - x_1)^2 + (y_i - y_1)^2 + (z_i - z_1)^2}. \tag{4.14}$$

Essa métrica calcula a proximidade entre pontos no espaço tridimensional, estimando a similaridade entre as amostras dentro de um raio r_s. Para selecionar uma amostra do conjunto de pontos semelhantes, desenvolve-se uma função de custo que maximize as posições angulares θ_2 e θ_4, e minimize as posições angulares θ_3. Após a seleção do ponto que melhor representa as amostras semelhantes, os demais pontos redundantes são removidos. Para simulação optou-se por escolher os seguintes raios de similaridade $r_s = \{0.5, 0.7, 1, 1.25, 1.5\}$ cm, e obter no final do processo cinco bases de dados filtradas.

4.4 Treinamento, Seleção dos Hiperparâmetros e Validação dos Modelos

É de fundamental importância definir sequência de etapas para o treinamento, validação e teste dos modelos de aprendizado de máquina. Isso implica em implementar modelos capazes de realizar previsões acertadas com base nos dados fornecidos. Para este trabalho, o fluxo de etapas de treinamento e validação dos modelos é ilustrado na Figura 26, e para cada abordagem de modelo, são seguidos os passos desse fluxograma.

Figura 26 – Fluxograma das etapas de treinamento e validação do modelos



Fonte: Autoral

Essencialmente, os dados da base de dados obtidos na Seção 4.3, são separados em conjuntos de dados para treinamento, validação e teste. Em seguida, é aplicada uma normalização ou padronização em cada conjunto de dados gerados na etapa anterior, com o objetivo de ajustar os dados para terem a mesma escala ou distribuição.

Na fase de treinamento, o conjunto de dados de treino é usado para ensinar o modelo a identificar os padrões da cinemática inversa. Posteriormente, na fase de validação, os modelos são avaliados usando o conjunto de dados de validação. A performance do modelo é avaliada usando uma métrica adequada ao problema de regressão, como é o caso da medida do erro quadrático médio da saída predita de cada modelo.

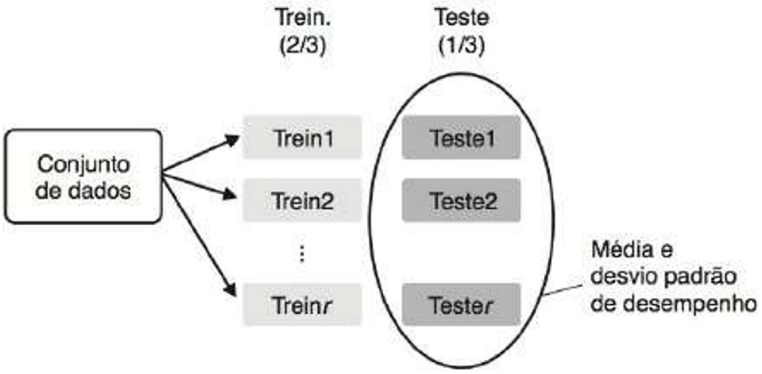
As avaliações dos modelos são usadas para ajustar e encontrar as melhores combinações dos hiperparâmetros de cada modelo. Portanto, para definir a arquitetura dos modelos e a otimização dos seus hiperparâmetros , o fluxo de treino e validação da Figura 26 ocorreu em n_r iterações.

4.4.1 Particionamento dos Dados

Após obter um conjunto de dados válido e representativo do volume de trabalho do manipulador didático, procedeu-se à operação de particionamento da base de dados em subconjuntos mutuamente exclusivos. O conjunto de treinamento será utilizado para treinar o modelo de aprendizado de máquina, o conjunto de validação será empregado para ajustar os hiperparâmetros, e o conjunto de teste será usado na etapa de avaliação do desempenho dos modelos.

A técnica utilizada neste trabalho para dividir os dados de treinamento e teste é o *Random Subsampling*, conforme ilustrado na Figura 27. Esse método consiste em repetir o processo de divisão aleatória em várias iterações, reservando uma proporção maior das amostras para o treinamento e o restante para testes no modelo, garantindo que os subconjuntos sejam mutuamente exclusivos (FACELI *et al.*, 2011). O modelo é treinado e testado com esses diferentes conjuntos, e o erro de regressão é avaliado pela média da taxa de erro quadrático obtida em cada iteração.

Figura 27 – Divisão da base de dados por amostragem aleatória



Fonte: (FACELI *et al.*, 2011)

Esse processo de particionamento envolveu a separação de 70% das amostras da base de dados para treinamento, 20% para validação e os 10% dos dados restante para testes. Essa divisão permite calcular métricas de desempenho médio mais precisas e compreende melhor a variabilidade do modelo para grandes conjuntos de dados. A escolha aleatória dos subconjuntos garante a ausência de uma ordem específica das amostras, evitando qualquer viés que possa ser introduzido se os dados estiverem ordenados de alguma maneira.

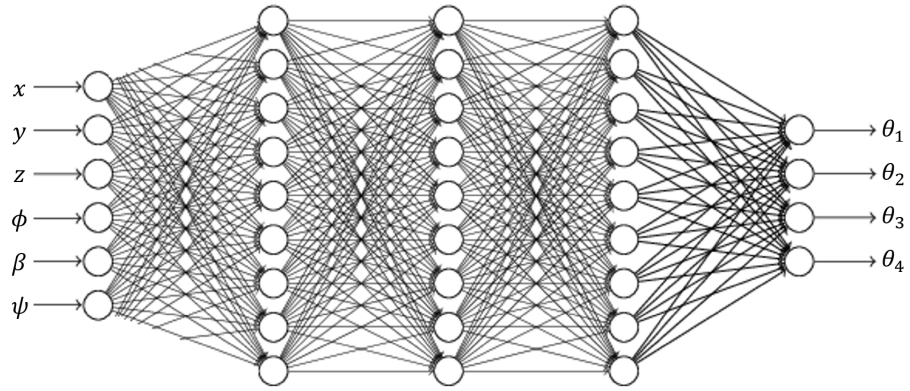
4.4.2 Modelos de Aprendizado de Máquina

Com o objetivo de solucionar o problema da cinemática inversa do manipulador didático, foram avaliadas e implementadas duas abordagens de modelos de regressão baseadas na teoria apresentada na Seção 3.2: as arquiteturas de modelo neural global e local. Cada modelo tem a função de aprender o comportamento da cinemática inversa para os ângulos (θ_1, θ_2, θ_3, θ_4) do volume de trabalho do robô, de acordo com os dados obtidos na Seção 4.3. O paradigma global aproxima a cinemática inversa mapeando todo o espaço operacional alcançável pelo manipulador, enquanto o paradigma local particiona o espaço operacional em sub-regiões, e para

cada uma dessas regiões, é definido um modelo inteligente específico (FACELI *et al.*, 2011).

A primeira abordagem consistiu na definição de um modelo de MLP global configurado em um sistema de Múltiplas Entradas e Múltiplas Saídas, conhecido como *Multiple Inputs, Multiple Outputs* (MIMO), conforme ilustrado na Figura 28. Esse modelo foi desenvolvido para estimar simultaneamente os ângulos $(\theta_1, \theta_2, \theta_3, \theta_4)$ a partir dos vetores de atributos $[x, y, z, \phi, \beta, \psi]$, que contêm informações de posição e orientação no espaço de trabalho. Nessa arquitetura, a rede neural processa diversas entradas através de camadas neurais, gerando múltiplas saídas correspondentes aos ângulos desejados. Esse formato permite à rede estimar todos os pontos no espaço de trabalho, aprendendo padrões e relações entre os dados do volume de trabalho e produzindo previsões simultâneas para todos os ângulos (MELO, 2015).

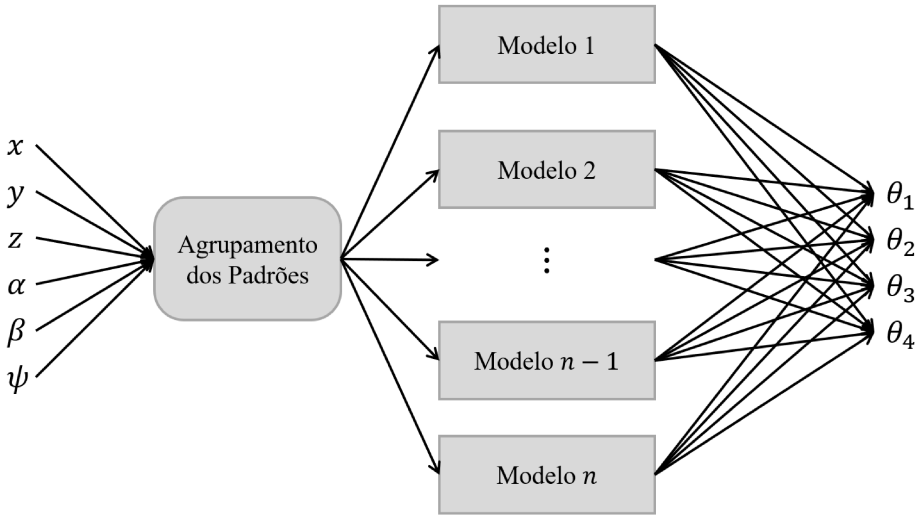
Figura 28 – Modelo global das redes MLP configuradas em sistemas MIMO



Fonte: Autoral

A segunda abordagem definiu k modelos de MLP locais, também configurados em sistemas MIMO, conforme descrito na Figura 29. Cada modelo local foi desenvolvido para estimar simultaneamente os ângulos $(\theta_1, \theta_2, \theta_3, \theta_4)$ a partir dos vetores de atributos $[x, y, z, \phi, \beta, \psi]$ das sub-regiões particionadas do volume de trabalho, que contêm informações de posição e orientação. O desenvolvimento dessa abordagem envolveu a partição do espaço alcançável do manipulador em sub-regiões, utilizando a técnica de agrupamento *k-means* (Seção 3.2.3). Com essa técnica, foi possível adicionar um algoritmo que direcionou os padrões de treinamento de cada sub-região para o respectivo modelo MLP local, permitindo o treinamento e, posteriormente, a predição dos ângulos das juntas a partir de suas entradas.

Figura 29 – Modelos locais de redes MLP configurada em subsistemas MIMO



Fonte: Autoral

4.4.3 Seleção de Hiperparâmetros e Treinamento

A seleção de hiperparâmetros é um processo importante no desenvolvimento de modelos de aprendizado de máquina, influenciando diretamente o desempenho do modelo. O uso de métodos adequados para escolher os hiperparâmetros é relevante para otimizar essa performance. Diferente dos parâmetros ajustados durante o treinamento, os hiperparâmetros são definidos previamente e afetam diretamente a capacidade de generalização do modelo. A escolha correta de hiperparâmetros, como a taxa de aprendizado e o número de camadas em redes neurais, pode maximizar o desempenho do modelo, evitando problemas como *overfitting* e *underfitting*, além de melhorar a acurácia (BERGSTRA; BENGIO, 2012).

Devido à complexidade desse processo, é necessário utilizar técnicas que gerem modelos com bom desempenho em dados desconhecidos. Métodos como *Grid Search*, *Randomized Search* e algoritmos mais recentes como o *Bayesian Optimization*, são amplamente utilizados para otimizar hiperparâmetros. O uso dessas técnicas não só economiza tempo, mas também aumenta as chances de encontrar um conjunto de hiperparâmetros que maximize a acurácia do modelo (HASTIE *et al.*, 2009). A escolha dos hiperparâmetros dos modelos de regressão global e local do tipo MLP adotados neste trabalho é o foco desta seção.

Este estudo utilizou a técnica *Random Search* para a busca de hiperparâmetros, variando o número de camadas ocultas, número de neurônios em cada camada e as funções de ativação. Bergstra e Bengio (2012) demonstram que esta técnica apresenta vantagens sobre a *Grid Search* em termos de eficiência computacional e resultados alcançados. Em vez de testar

exaustivamente todas as combinações possíveis, a *Random Search* seleciona aleatoriamente um subconjunto de combinações, permitindo encontrar boas soluções com menor custo computacional. Portanto, essa abordagem, combinada com a validação cruzada discutida anteriormente (Seção 4.4.1), visa maximizar o desempenho dos modelos MLP adotados neste trabalho.

As principais configurações dos parâmetros fixos nos modelos local e global, ou seja, parâmetros que não participaram da otimização, estão descritas na Tabela 6, e incluem o número máximo de épocas, a tolerância para o critério de parada e a possibilidade de parada antecipada, com o objetivo de prevenir o *overfitting* durante o treinamento. Esses parâmetros foram configurados para equilibrar o tempo de execução e a qualidade da busca, sendo definidos por meio de alguns testes.

Tabela 6 – Parâmetros fixos dos modelos inteligentes

Parâmetros	Valor
Número máximo de épocas	1000
Erro tolerado	1×10^{-4}
Número de parada sem mudança	5
Fator de *momentum*	0.9
Regularização (α)	1×10^{-4}

Fonte: Autoral

Já como métricas de avaliação, foram utilizados o Coeficiente de Determinação (R^2) e o erro quadrático médio (MSE), sendo esta última a métrica utilizada para ranquear os melhores conjuntos de hiperparâmetros. A validação cruzada foi implementada por meio da técnica de *Random Subsampling*, conforme descrito na Seção 4.4.1, que divide o conjunto de dados de treino e validação em partições, garantindo que o modelo seja testado em diferentes subconjuntos de dados durante o processo de seleção de hiperparâmetros.

Para os parâmetros otimizados do modelo global, foi definido um espaço de busca focado em múltiplas arquiteturas de MLP, com diferentes números de neurônios por camada e funções de ativação, como ilustrado na Tabela 7. O número de neurônios variou entre camadas, permitindo que a arquitetura da rede se ajustasse de acordo com as características dos dados. As funções de ativação consideradas no espaço de busca foram ReLU e tanh. No total, foram utilizadas $n_i = 2000$ número de iterações para explorar o espaço de busca das combinações de hiperparâmetros.

Já para a otimização dos hiperparâmetros dos modelos locais, o processo começou com a divisão das regiões do espaço operacional do robô. Nesse caso, aplicou-se o algoritmo

Tabela 7 – Hiperparâmetros do espaço de busca do modelo global

Número de Camadas	Mínimo de Neurônios	Máximo de Neurônios	Passo
1	20	6000	5
2	100	500	10
3	100	500	35
4	100	400	55

Funções de Ativação: ReLU, tanh

Fonte: Autoral

K-means, que agrupou o espaço em regiões com base nos atributos (x, y, z). Foi definido um espaço de busca com foco nos *clusters* dos subespaços operacionais, variando o número de grupos entre $k = [2, .., 14]$ combinações de agrupamentos do espaço. Para determinar o número ideal de grupos para dividir as regiões do espaço operacional, aplicaram-se duas técnicas de avaliação: o coeficiente de silhueta e a inércia (Seção 4.4.4). Cada métrica foi armazenada para uma análise criteriosa, a fim de selecionar o melhor valor do hiperparâmetro k.

4.4.4 Métricas de Avaliação dos Modelos de Regressão e Agrupamento

O Erro Médio Quadrático (MSE), é uma das principais métricas utilizadas para avaliar o desempenho de modelos de regressão. Essa métrica calcula a média dos quadrados das diferenças entre os valores previstos pelo modelo e os valores reais observados nos dados (FACELI *et al.*, 2011). Na Equação 4.15, o erro da hipótese ξ_i de um modelo $\hat{f}$ é dado pela distância entre o valor de saída real y_i e aquele predito pelo modelo $\hat{f}(x_i)$, portanto $\xi_i = y_i - \hat{f}(x_i)$. O MSE é dado por:

$$\text{MSE}(\hat{f}) = \frac{1}{n}\sum_{i=1}^{n}(\xi_i)^2 \tag{4.15}$$

Essa métrica penaliza erros maiores, pois cada erro é elevado ao quadrado antes de ser somado e normalizado pelo número total de observações. Quanto menor o valor do MSE, melhor o modelo se ajusta aos valores reais do manipulador robótico. Portanto, um MSE baixo indica uma melhor capacidade de generalização do modelo.

Além disso, para compreender melhor o comportamento dos erros, é utilizado o desvio padrão dos erros, dado por:

$$\sigma(\xi) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\xi_i - \mu(\xi))^2} \tag{4.16}$$

Essa métrica é importante porque valores altos de desvio padrão indicam uma grande variação nos erros preditos pelo modelo, sugerindo que ele é sensível às amostras avaliadas (FACELI *et al.*, 2011).

No presente trabalho, foram definidas duas abordagens para o cálculo do MSE. A primeira aborda os erros no espaço das juntas, onde são avaliadas diretamente as predições dos ângulos das juntas gerados pelos modelos. Para cada junta $\{\theta_k | k = 1,..,4\}$, o erro $\xi_i(\theta_k)$ é calculado por meio da métrica MSE. A segunda abordagem considera os erros no espaço operacional, avaliando a diferença entre a posição cartesiana desejada e a posição cartesiana reconstruída a partir dos ângulos preditos pelo modelo na modelagem da cinemática direta. O MSE operacional é descrito em termos do erro $\xi_i(\mathbb{R}^3)$ no espaço cartesiano (DEMBY'S *et al.*, 2019).

Outra métrica utilizada foi o Coeficiente de Determinação (R^2), amplamente empregado para avaliar a performance de modelos de regressão. O R^2 é definido como a proporção da variância na variável dependente que pode ser explicada pelas variáveis independentes. Ele varia entre $[0,1]$, e é dado por:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}. \tag{4.17}$$

Um valor de R^2 igual a 1 indica que o modelo explica completamente a variância dos dados, enquanto um valor de 0 sugere que o modelo não é melhor do que uma simples média dos valores (FACELI *et al.*, 2011).

A avaliação de modelos de agrupamento em aprendizado de máquina, por outro lado, difere dos modelos supervisionados, onde há uma métrica de desempenho claramente definida. Nesse caso, como não existem rótulos previamente conhecidos, torna-se necessário o uso de métricas internas e externas específicas. Métricas internas, como o coeficiente de silhueta e a inércia, avaliam a coesão e a separação dos grupos com base nas características dos dados. Já as métricas externas comparam o resultado do agrupamento com uma estrutura de referência, se disponível.

No método *k-Means*, cujo objetivo principal é dividir os dados em k grupos distintos, minimizando a variabilidade interna dos grupos e maximizando a separação entre eles, a validação do número adequado de grupos k e a qualidade dos agrupamentos gerados são realizadas com a análise do coeficiente de silhueta e da inércia (FACELI *et al.*, 2011).

O coeficiente de silhueta é uma métrica que avalia a coesão e a separação dos grupos gerados pelo *K-Means*. Para cada ponto i, o coeficiente de silhueta $s(i)$ e a largura média de silhueta são dadas por:

$$s = \frac{1}{N}\sum_{i=1}^{N}\frac{b(i) - a(i)}{\max(a(i), b(i))}. \tag{4.18}$$

Onde $a(i)$ é a distância média entre o ponto *i* e todos os demais pontos do mesmo *cluster*, e $b(i)$ é a menor distância média entre o ponto *i* e todos os pontos do *cluster* mais próximo ao qual *i* não pertence. O coeficiente de silhueta varia entre $[-1, 1]$. Valores próximos de 1 indicam que o ponto está bem ajustado ao seu próprio grupo e longe dos grupos vizinhos, enquanto valores próximos de -1 indicam que o ponto foi mal classificado (FACELI *et al.*, 2011).

O método do cotovelo (*elbow method*) é uma técnica usada para identificar o número ideal de grupos *k*, analisando a soma das variâncias internas dos *clusters*, também chamada de inércia. A inércia é a soma das distâncias quadradas das observações ao seu centróide, conforme a Equação 3.21. À medida que o número de grupos aumenta, a inércia tende a diminuir, pois mais *clusters* resultam em distâncias internas menores. O ponto em que a taxa de diminuição começa a se estabilizar forma o "cotovelo"na curva da inércia, indicando o número adequado de *clusters* (FACELI *et al.*, 2011).

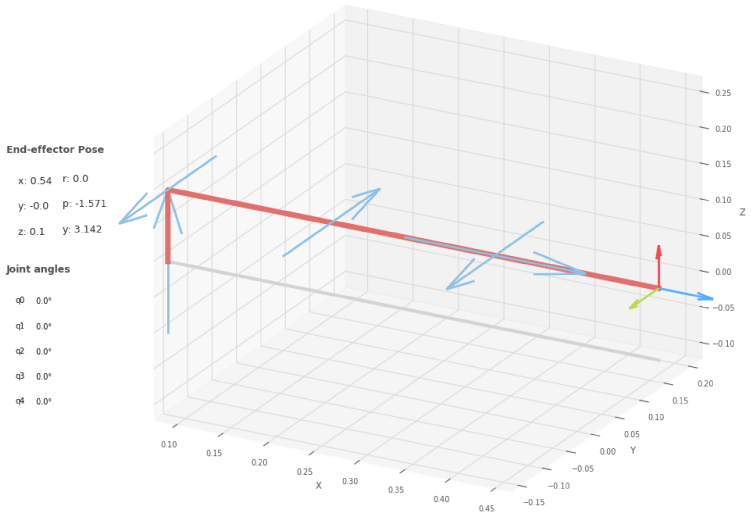
5 RESULTADOS

Neste capítulo, são apresentados os resultados obtidos da modelagem da cinemática inversa do robô manipulador para aproximar a solução da cinemática inversa. Inicialmente, é avaliada a modelagem direta teórica, conforme descrito nas seções anteriores, a qual serve de referência para geração dos padrões de treinamento. Em seguida, os resultados do processo de ajuste de hiperparâmetros e a performance dos modelos globais e locais são discutidos em termos de precisão, generalização e eficiência, utilizando métricas quantitativas para validar a solução proposta.

5.1 Cinemática Direta

A validação da cinemática direta foi conduzida utilizando a parametrização D-H, conforme descrita na Seção 4.2, aplicada a um modelo de braço robótico desenvolvido na biblioteca *Robotics Toolbox for Python* (CORKE; HAVILAND, 2021). O uso dessa biblioteca permitiu a visualização e análise das configurações articulares, bem como a disposição espacial no espaço cartesiano dos sistemas de referências locais. Os parâmetros D-H foram diretamente inseridos no modelo, e a simulação gerou a Figura 30 que representa as posições e orientações do braço robótico com base nas configurações articulares iniciais, nas quais todos os ângulos das juntas foram zerados.

Figura 30 – Simulação da modelagem direta do manipulador didático

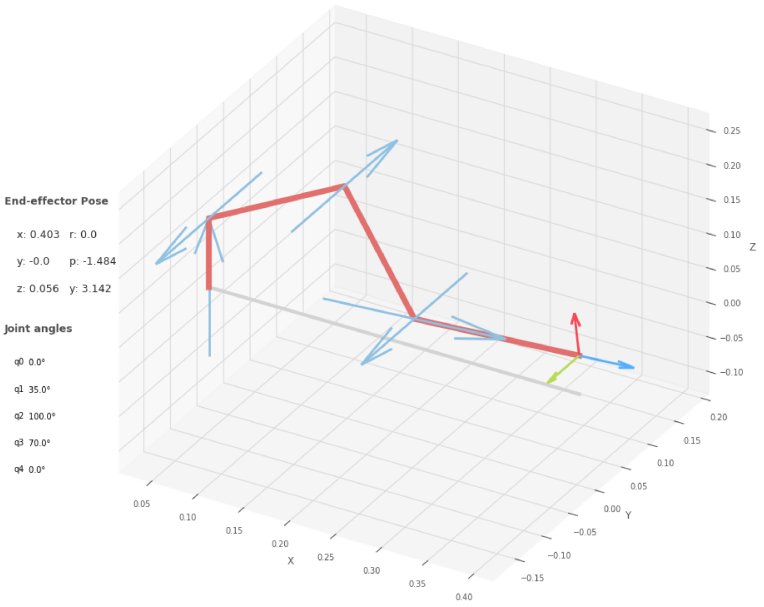


Fonte: Autoral

Após a inserção dos parâmetros D-H, a validação demonstrou a consistência do

modelo teórico, ao calcular com precisão a posição final do efetuador a partir das coordenadas articulares. Na Figura 30, observa-se a correspondência direta dos sistemas de coordenadas locais, previamente definidos nas etapas da modelagem direta representada na Figura 20. Já as Figuras 31 e 32 apresentam configurações com diferentes valores de ângulos nas articulações, e os resultados obtidos para a posição e orientação do efetuador coincidiram com as previsões teóricas.

Figura 31 – Simulação da modelagem direta do manipulador didático para ângulos das juntas $\theta_i = [0°, 35°, 100°, 70°, 0°]$



Fonte: Autoral

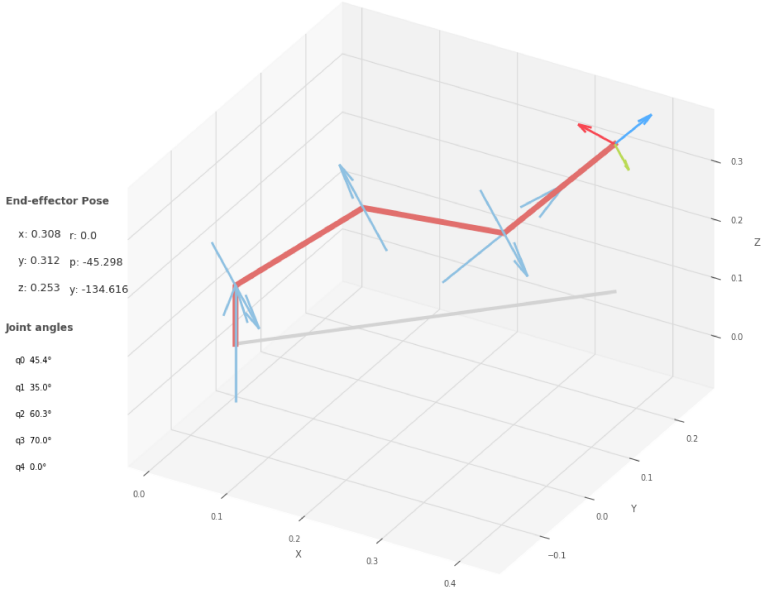
Os resultados indicaram que a modelagem dos parâmetros D-H foi eficaz em representar o comportamento esperado do robô, confirmando que o braço robótico simulado atingiu as posições e orientações corretas, em conformidade com os cálculos teóricos da cinemática direta na Tabela 8. As posições simuladas, ilustradas nas Figuras 31 e 32, demonstram que a implementação é adequada para simulações de tarefas de manipulação robótica.

Tabela 8 – Saídas da modelagem direta teórica para diferentes ângulos

Ângulos $(\theta_1, ..., \theta_5)$	x (cm)	y (cm)	z (cm)
$(0, 35, 100, 70, 0)$	$40,283$	$0,000$	$5,580$
$(45, 35, 60, 70, 0)$	$30,962$	$30,962$	$25,445$

Fonte: Autoral

Figura 32 – Simulação da modelagem direta do manipulador didático para ângulos das juntas $\theta_i = [45°, 35°, 60°, 70°, 0°]$



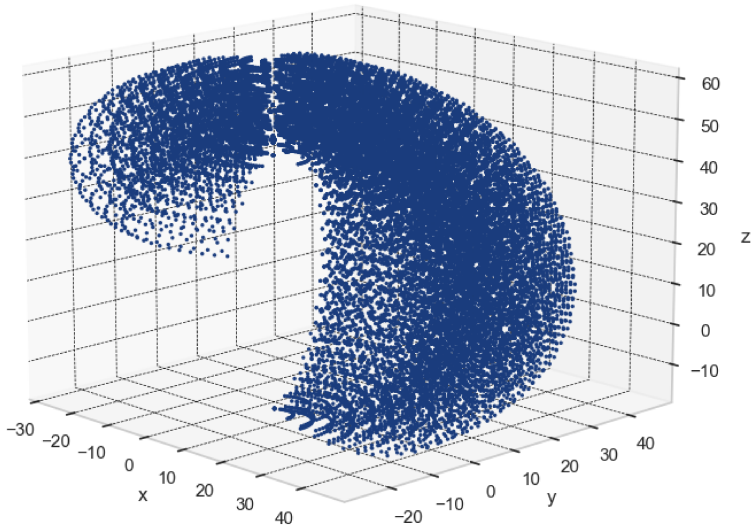
Fonte: Autoral

5.2 Padrões de Treinamento

De posse da cinemática direta e com base nas restrições impostas pela geometria e pelos limites físicos das juntas (Tabela 4), foi possível gerar uma base de dados com 33275 observações, contendo informações detalhadas sobre as configurações angulares e as posições espaciais do efetuador no ambiente cartesiano, conforme ilustrado na Figura 33 o espaço de trabalho alcançável pelo manipulador. A decisão de fixar o valor de θ_5, devido à sua influência apenas na orientação e não na posição do efetuador, simplificou a geração dos padrões de treinamento sem comprometer a precisão do mapeamento do volume de trabalho.

A distribuição diferenciada do número de amostras por junta, conforme evidenciado na Tabela 4, permitiu uma cobertura eficaz dos eixos x, y, e z. A maior densidade de amostras para o ângulo θ_1 otimizou a exploração do espaço bidimensional (x,y), enquanto os outros ângulos foram balanceados de maneira a cobrir adequadamente o eixo z. Em testes anteriores, considerando um número fixo de 25 amostras por GDL, obteve-se um total de 456976 pontos. Com as resoluções adotadas na presente abordagem, houve uma redução de $91,5\%$ no total de amostras da base de dados, portanto diminuindo a geração de dados redundantes. A extração dos atributos essenciais focou nas coordenadas cartesianas (x,y,z), que mapeiam a posição do efetuador no espaço tridimensional, e nos ângulos (ϕ,β,ψ) que descrevem a orientação do efetuador. A inclusão dos ângulos *Roll*, *Pitch* e *Yaw*, derivados da matriz de rotação, aumentou o número de atributos relevantes, facilitando o aprendizado dos modelos.

Figura 33 – Volume de trabalho total do manipulador didático



Fonte: Autoral

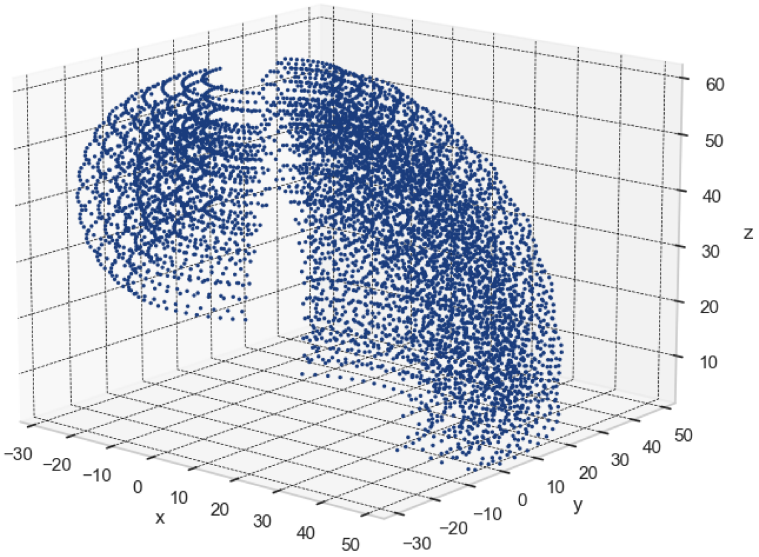
Uma análise dos limites operacionais do manipulador revelou que determinadas posições no espaço de trabalho eram inatingíveis devido a restrições físicas, especialmente aquelas localizadas no semi-plano negativo do eixo z, onde se encontram o suporte e a base fixa do robô. Para evitar que o modelo fosse treinado com dados inviáveis, foi aplicado um filtro que removeu 1089 amostras inatingíveis. Essa etapa garantiu que a base de dados fosse limitada ao volume de trabalho fisicamente viável, assegurando maior realismo no treinamento do modelo.

Outro passo importante no pré-processamento foi a eliminação de redundâncias. A Figura 34 ilustra o volume de trabalho após essa etapa e as demais do processo de pré-processamento da base de dados. Pontos próximos entre si no espaço tridimensional, que não agregavam novas informações ao modelo, foram identificados como redundantes. Utilizando a métrica da distância euclidiana, foram removidas as amostras cuja proximidade se encontrava dentro de um raio de similaridade de $r_s = 1\text{cm}$.

Já a Figura 35, é mostrado o histograma do espaço das juntas após a remoção das redundâncias, indicando que a função custo gerou um espaço de ângulos que maximiza as posições angulares de θ_2.

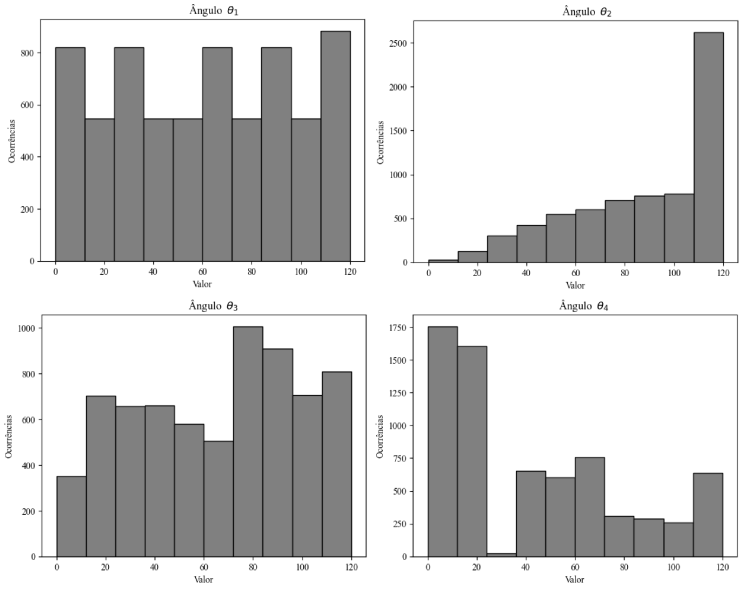
Com essa abordagem, a base de dados final passou a conter 6890 amostras, representando uma redução de $20,7\%$ dos dados originais, sem comprometer a variabilidade. A Figura 36 destaca os pontos azuis, selecionados como não redundantes, em contraste com os pontos vermelhos removidos. Como resultado, obteve-se um conjunto de dados mais enxuto e eficiente, permitindo que os algoritmos de aprendizado de máquina se concentrassem em padrões

Figura 34 – Volume de trabalho do manipulador sem redundâncias



Fonte: Autoral

Figura 35 – Histograma do espaço das juntas após a remoção das redundâncias



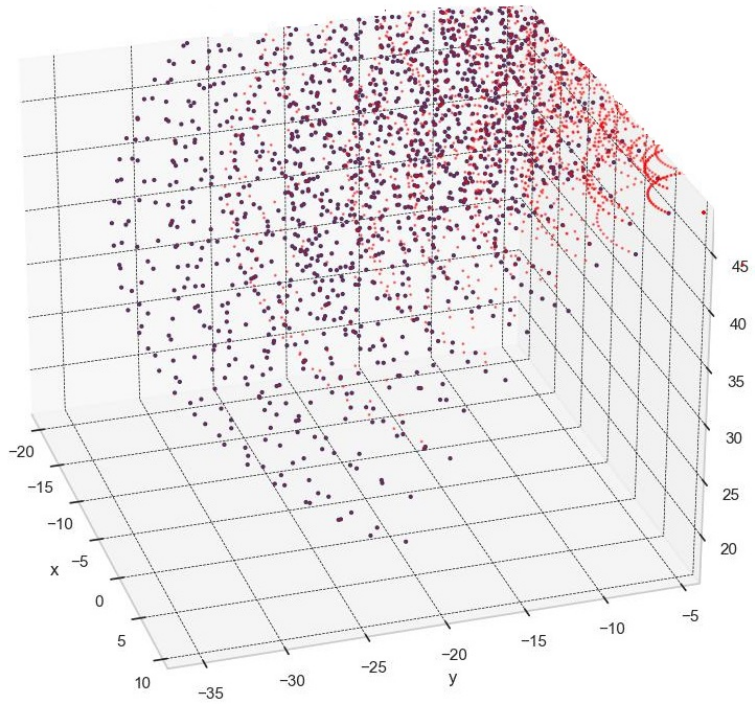
Fonte: Autoral

relevantes, com menor tempo de processamento e melhoria na capacidade de aprendizado e precisão dos resultados.

5.3 Seleção de Hiperparâmetros, Treinamento e Validação dos Modelos

Na fase de otimização de hiperparâmetros, os parâmetros ajustados no modelo global incluíram o número de camadas ocultas, o número de neurônios por camada e as funções de ativação. A seleção dos hiperparâmetros foi realizada por meio da técnica de *Random Search*

Figura 36 – Comparação entre as amostras retiradas da base de dados em uma região do volume de trabalho



Fonte: Autoral

com validação cruzada, focando na otimização das arquiteturas de redes neurais do tipo MLP. O objetivo principal foi encontrar uma configuração ideal que maximizasse a capacidade preditiva do modelo na resolução do problema de cinemática inversa do manipulador robótico, conforme detalhado nas seções anteriores.

O espaço de busca do modelo global foi definido conforme a Tabela 7, onde foi definido com base em diferentes combinações de funções de ativação (ReLU e tanh), além de variações no número de camadas e neurônios por camada, como descrito na Seção 4.4.3. Durante a busca, foram realizados 2000 experimentos distintos, variando os hiperparâmetros dentro do espaço de busca predefinido. Durante esse processo, aplicou-se a validação cruzada via *Random Subsampling*, garantindo uma divisão eficiente entre os conjuntos de treino e validação, o que minimizou o risco de *overfitting* e maximizou a capacidade de generalização.

A Tabela 9 presenta os dez melhores resultados obtidos em termos de MSE e o R^2 para os dados normalizados. O modelo global foi configurado para prever simultaneamente os ângulos do manipulador robótico em todo o volume de trabalho. O tempo médio de treinamento foi de 12 segundos, com o tempo máximo de 97 segundos. A técnica de parada antecipada foi ativada com sucesso em diversas execuções, prevenindo o *overfitting* e garantindo eficiência no treinamento.

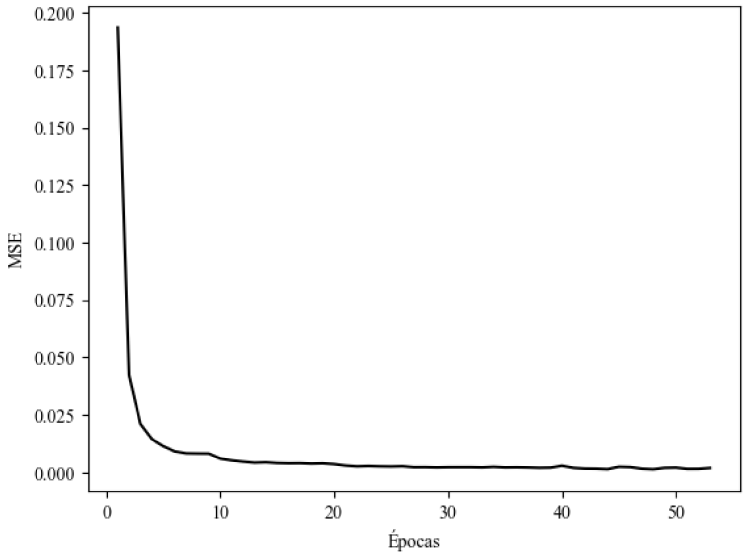
Tabela 9 – Resultados ranqueados dos dez melhores hiperparâmetros do modelo global

Classificação	$\text{MSE}(\overline{\theta_i})$		$R^2(\overline{\theta_i})$	
	Treinamento	Validação	Treinamento	Validação
1	$2,22 \times 10^{-3}$	$2,70 \times 10^{-3}$	$9,978 \times 10^{-1}$	$9,973 \times 10^{-1}$
2	$2,30 \times 10^{-3}$	$2,90 \times 10^{-3}$	$9,977 \times 10^{-1}$	$9,971 \times 10^{-1}$
3	$2,50 \times 10^{-3}$	$3,00 \times 10^{-3}$	$9,975 \times 10^{-1}$	$9,970 \times 10^{-1}$
4	$2,50 \times 10^{-3}$	$3,00 \times 10^{-3}$	$9,975 \times 10^{-1}$	$9,970 \times 10^{-1}$
5	$2,60 \times 10^{-3}$	$3,00 \times 10^{-3}$	$9,974 \times 10^{-1}$	$9,970 \times 10^{-1}$
6	$2,50 \times 10^{-3}$	$3,20 \times 10^{-3}$	$9,975 \times 10^{-1}$	$9,968 \times 10^{-1}$
7	$2,70 \times 10^{-3}$	$3,30 \times 10^{-3}$	$9,973 \times 10^{-1}$	$9,967 \times 10^{-1}$
8	$2,60 \times 10^{-3}$	$3,30 \times 10^{-3}$	$9,974 \times 10^{-1}$	$9,967 \times 10^{-1}$
9	$2,90 \times 10^{-3}$	$3,30 \times 10^{-3}$	$9,971 \times 10^{-1}$	$9,967 \times 10^{-1}$
10	$2,50 \times 10^{-3}$	$3,30 \times 10^{-3}$	$9,975 \times 10^{-1}$	$9,967 \times 10^{-1}$

Fonte: Autoral

A técnica de *Random Search* identificou uma configuração de hiperparâmetros otimizada, com uma arquitetura de quatro camadas ocultas e neurônios nas camadas $(320, 375, 265, 155)$, utilizando a função de ativação ReLU. A Figura 37 ilustra a curva de aprendizado do melhor conjunto de hiperparâmetro selecionado. Esse modelo alcançou um $R^2(\overline{\theta_i}) = 9,973 \times 10^{-1}$ e um $\text{MSE}(\overline{\theta_i}) = 2,70 \times 10^{-3}$, destacando-se pelo menor erro, e por consequência o mais preciso entre os testados. O segundo melhor modelo, com camadas de $(155, 375, 155, 375)$ neurônios e também utilizando a função ReLU, apresentou desempenho semelhante, com $R^2(\overline{\theta_i}) = 9,971 \times 10^{-1}$ e $\text{MSE}(\overline{\theta_i}) = 2,90 \times 10^{-3}$.

Figura 37 – Curva de aprendizado do melhor conjunto de hiperparâmetro do modelo global



Fonte: Autoral

Para avaliar o erro real das juntas afim de compreende-los melhor, o MSE das

predições dos ângulos reais foi calculado, conforme ilustrado na Tabela 10. Onde é dado o MSE para o conjunto específico de hiperparâmetros que apresentou o menor erro durante o processo de validação cruzada. Para avaliar a qualidade da regressão, é avaliado o erro de predição dos modelos para dados que não participaram em nenhum momento do treinamento do modelo. A análise dos dados de teste revelou que o modelo global generalizou bem as predições para dados desconhecidos.

Tabela 10 – Erros dos ângulo em graus das juntas preditos pelo modelo global

Junta	MSE(θ_i) Treinamento ($^\circ$)	MSE(θ_i) Teste ($^\circ$)
θ_1	$0,9382$	$0,9774$
θ_2	$1,6871$	$1,7533$
θ_3	$4,4431$	$4,2210$
θ_4	$3,0766$	$3,3884$

Fonte: Autoral

Além disso, foi analisado como os erros de predição dos ângulos das juntas impactam o espaço operacional, tendo em vista o objeto de estudo do presente trabalho. Desta maneira, as predições dos ângulos das juntas foram utilizadas na matriz de transformação total (Equação 4.1) oriunda da modelagem direta para calcular as coordenadas cartesianas preditas pelos modelos no espaço operacional. A Tabela 11 apresenta os erros de predição no espaço operacional.

Tabela 11 – Erros do espaço operacional preditos pelo modelo global

Coordenadas	MSE($\mathbb{R}^3$) Treinamento (cm)	MSE($\mathbb{R}^3$) Teste (cm)
x	$2,372 \times 10^{-1}$	$2,406 \times 10^{-1}$
y	$1,763 \times 10^{-1}$	$1,810 \times 10^{-1}$
z	$1,842 \times 10^{-1}$	$1,939 \times 10^{-1}$

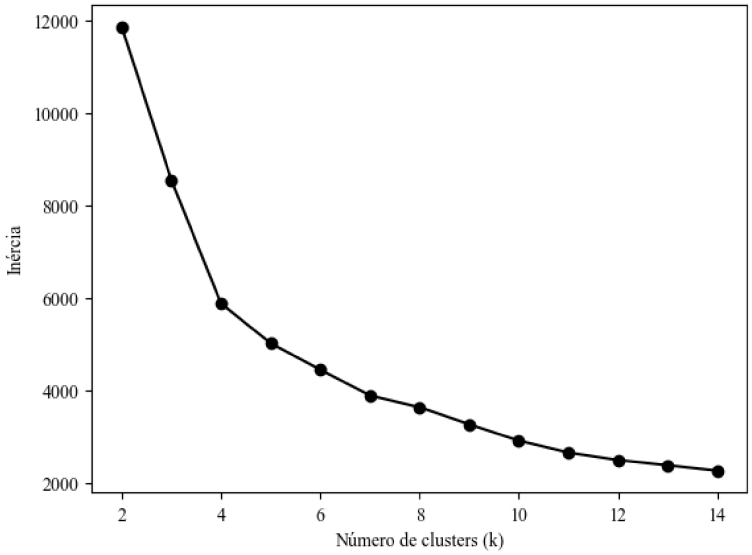
Fonte: Autoral

Para o modelo local, os hiperparâmetros ajustados incluíram o número de *clusters* k, responsáveis por subdividir o espaço operacional em regiões. Para tanto, é aplicado o algoritmo *K-means* para agrupar o conjunto de dados em grupos que sejam mais semelhantes entre si do que com os dados de outros grupos. A otimização foi realizada com o *K-means*, com o objetivo de determinar o número ideal de clusters que melhor separaria as regiões tridimensionais (x, y, z) do espaço de trabalho do manipulador. O espaço de busca para k variou entre 2 a 14, e a avaliação foi feita com as métricas de inércia e coeficiente de silhueta.

O gráfico da Figura 38 ilustra a inércia, que representa a soma das distâncias qua-

dradas entre os pontos e o centro de seus respectivos *clusters* para cada valor de k. No gráfico, é possível observar o comportamento dessa métrica conforme o número de *clusters* aumentou. A análise da inércia revelou uma queda acentuada entre $k = 2$ e $k = 4$, com pequenos ganhos após esse ponto. Esse padrão sugere que um número de *clusters* maior que 4 oferece ganhos marginais em termos de compactação dos grupos, indicando que o valor de $k = 4$ pode ser um ponto ótimo, pois a inércia começa a estabilizar após essa quantidade de *clusters*.

Figura 38 – Gráfico do método do cotovelo aplicado no espaço operacional do manipulador

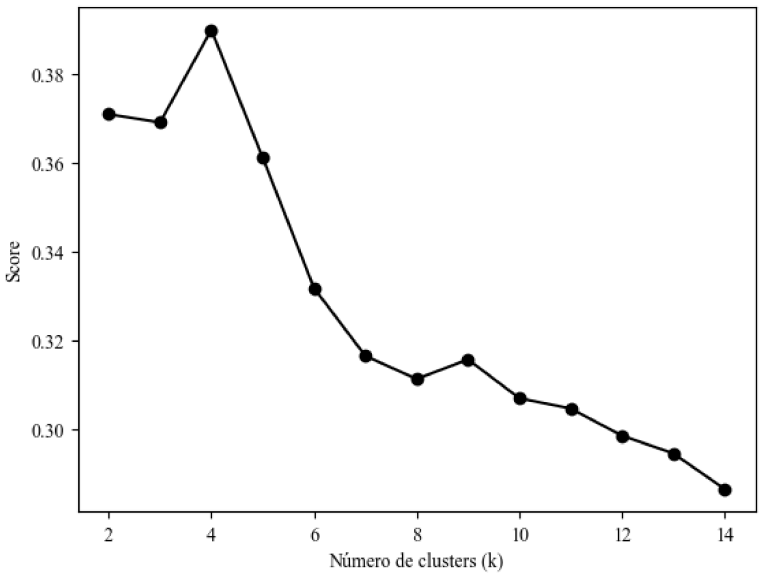


Fonte: Autoral

O coeficiente de silhueta, por sua vez, que mede o quão próximos os pontos estão dos seus próprios *clusters* em relação aos demais *clusters*, é representado no gráfico da Figura 39. O coeficiente atingiu seu valor máximo em $k = 4$, sugerindo que essa quantidade de *clusters* resulta na melhor separação e coesão dos grupos. Para valores de k superiores, o coeficiente de silhueta diminuiu gradualmente, indicando uma perda de qualidade na separação dos *clusters*, possivelmente devido ao aumento de sobreposição entre grupos.

Assim, a escolha de $k = 4$ garantiu uma segmentação eficaz do espaço operacional, como ilustrado nas Figuras 38 e 39. A inércia, embora continue a diminuir para valores maiores de k, sugere que $k = 4$ é o ponto de saturação, onde aumentos adicionais no número de *clusters* resultam em pouca melhora prática na compactação dos grupos. Além disso, o coeficiente de silhueta reforça essa conclusão, uma vez que apresenta seu melhor desempenho nesse ponto. Portanto, a escolha de $k = 4$ gerou a divisão do espaço operacional conforme ilustrado na Figura 40. Esse Número de sub regiões garante uma boa segmentação do espaço operacional, conforme

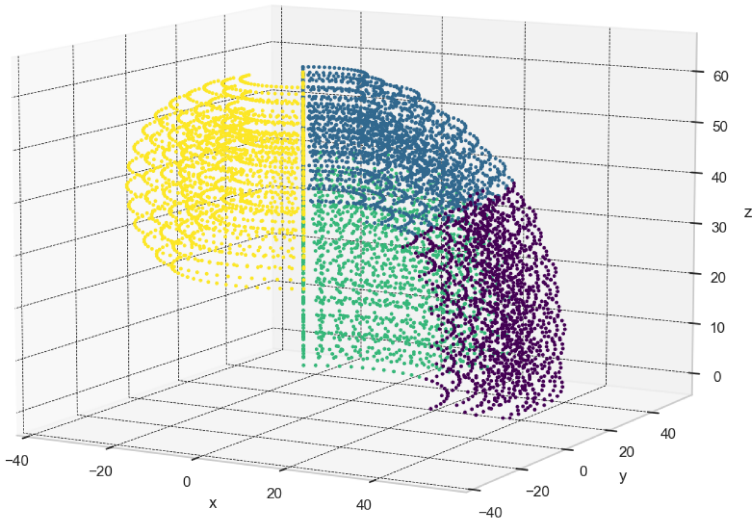
Figura 39 – Gráfico do método da silhueta aplicado no espaço operacional do manipulador



Fonte: Autoral

ilustrado no gráfico de dispersão das amostras em cada sub-região da Figura 41, minimizando a variabilidade interna as sub-regiões ao mesmo tempo que maximiza a distinção entre eles.

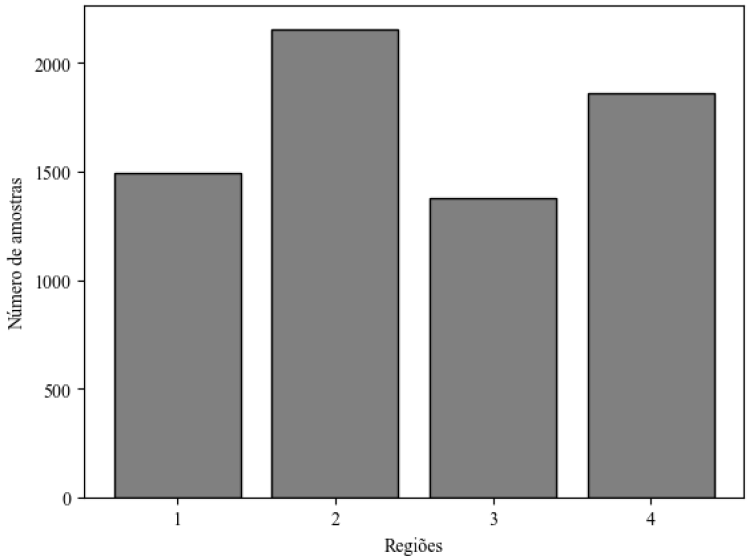
Figura 40 – Espaço operacional do manipulador dividido em sub regiões



Fonte: Autoral

Após a definição das regiões, as arquiteturas dos modelos de regressão foram configuradas para cada sub-região. Utilizou-se redes MLP para cada modelo local com quatro camadas ocultas e neurônios dispostos em $(320, 375, 265, 155)$ e função de ativação ReLU. A Figura 42 ilustra as curvas de aprendizado desses modelos para o conjunto de hiperparâmetro selecionado. Os modelos locais em cada região obtiveram o erro, para os dados normalizados,

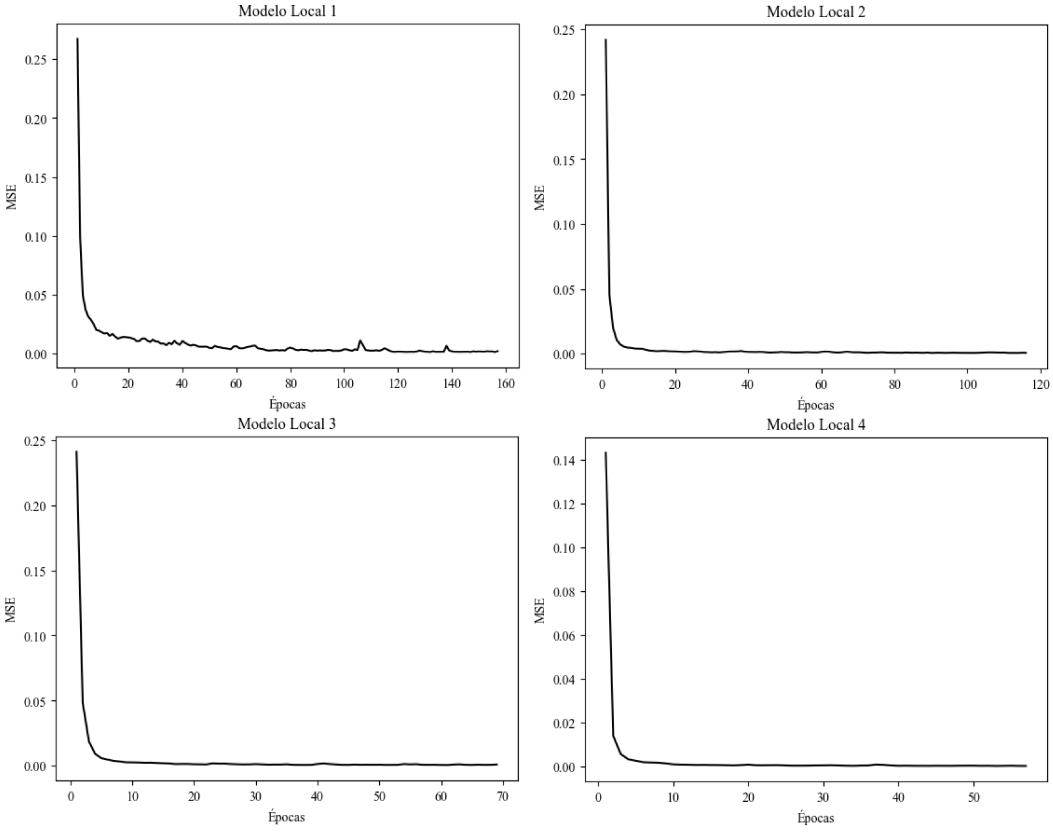
Figura 41 – Gráfico de barras da dispersão das amostras nas sub- regiões



Fonte: Autoral

iguais a $\text{MSE}(\theta_i) = (2.132 \times 10^{-3}, 6.656 \times 10^{-4}, 9.771 \times 10^{-4}, 3.457 \times 10^{-4})$, assim no geral o modelo local teve um $\text{MSE}(\theta_i) = 1.030 \times 10^{-3}$.

Figura 42 – Curva de aprendizado dos modelos locais



Fonte: Autoral

Assim como no caso do modelo global, é de interesse compreender melhor os erros

reais de predição das juntas, portanto é definido a Tabela 12, onde, para avaliar os modelos locais de regressão é apresentado os MSE dos dados que participaram do treinamento bem como os dados de testes que não participaram do treinamento. Os resultados apresentado demonstram os erros médios quadráticos MSE obtidos durante o treinamento e teste de quatro modelos locais para predição dos ângulos das juntas, indicados por θ_1, θ_2, θ_3 e θ_4.

Tabela 12 – Erros dos ângulo em graus das juntas preditos pelo modelos locais

Modelo Local	θ	MSE(θ_i) Treinamento ($^\circ$)	MSE(θ_i) Teste ($^\circ$)
1	θ_1	$0,342$	$0,308$
	θ_2	$1,634$	$0,572$
	θ_3	$4,625$	$2,011$
	θ_4	$1,296$	$0,831$
2	θ_1	$0,055$	$0,064$
	θ_2	$0,788$	$0,795$
	θ_3	$1,803$	$1,862$
	θ_4	$1,677$	$2,108$
3	θ_1	$0,069$	$0,054$
	θ_2	$0,554$	$0,421$
	θ_3	$1,574$	$1,054$
	θ_4	$0,648$	$0,544$
4	θ_1	$0,254$	$0,259$
	θ_2	$0,135$	$0,143$
	θ_3	$0,265$	$0,242$
	θ_4	$0,452$	$0,598$

Fonte: Autoral

Os quatro modelos locais apresentaram variações quanto ao seu desempenho. O Modelo 1 mostrou bons resultados para θ_1, mas apresentou maior erro para θ_3, com MSE de $4,625^\circ$ no treinamento e $2,011^\circ$ no teste, indicando dificuldades em prever corretamente esse ângulo. O Modelo 2 apresentou baixo erro para θ_1, mas mostrou maiores dificuldades para θ_3 e θ_4, cujos erros chegaram a $1,862^\circ$ e $2,108^\circ$, respectivamente. O Modelo 3 foi mais equilibrado, com desempenho satisfatório em θ_1 e θ_2, e uma redução significativa no erro de θ_3, com MSE de $1,054^\circ$ no teste. Por fim, o Modelo 4 apresentou o melhor desempenho geral, com erros baixos e consistentes em todas as juntas, sugerindo uma boa capacidade de generalização. Para todos os casos, os modelos locais apresentaram uma maior vantagem comparada ao modelo global.

Além disso, a Tabela 13 apresenta os erros MSE($\mathbb{R}^3$) dos modelos locais na predição dos ângulos que levam o manipulador a atingir a pose desejada no espaço operacional. O erro é medido em cada um dos três eixos cartesianos (x, y, z) e avalia a precisão dos modelos ao determinar os ângulos corretos para alcançar as coordenadas de referência.

Tabela 13 – Erros do espaço operacional preditos pelos modelos locais

Modelo Local	Eixo	MSE($\mathbb{R}^3$) Treinamento (cm)	MSE($\mathbb{R}^3$) Teste (cm)
1	x	$3,8 \times 10^{-2}$	$4,3 \times 10^{-2}$
	y	$3,2 \times 10^{-2}$	$3,4 \times 10^{-2}$
	z	$2,5 \times 10^{-2}$	$2,5 \times 10^{-2}$
2	x	$2,6 \times 10^{-2}$	$2,8 \times 10^{-2}$
	y	$2,2 \times 10^{-2}$	$2,4 \times 10^{-2}$
	z	$5,8 \times 10^{-2}$	$6,1 \times 10^{-2}$
3	x	$3,1 \times 10^{-2}$	$2,7 \times 10^{-2}$
	y	$2,4 \times 10^{-2}$	$2,0 \times 10^{-2}$
	z	$7,8 \times 10^{-2}$	$6,3 \times 10^{-2}$
4	x	$2,1 \times 10^{-2}$	$2,2 \times 10^{-2}$
	y	$3,0 \times 10^{-2}$	$2,9 \times 10^{-2}$
	z	$1,9 \times 10^{-2}$	$1,8 \times 10^{-2}$

Fonte: Autoral

Analisando o desempenho dos quatro modelos, observa-se que cada um apresenta uma boa taxa de erro MSE, apesar de apresentarem variações dependendo do eixo analisado. O modelo 1 teve um desempenho equilibrado nos três eixos, com erros de teste ligeiramente maiores para x e y. O modelo 2 apresentou bom desempenho em x e y, com erros próximos de $2,4 \times 10^{-2}$ cm, mas apresentou um aumento no erro para z ($6,1 \times 10^{-2}$ cm). Já o modelo 3 teve resultados semelhantes ao modelo 2, com uma melhora no eixo x no teste, mas um erro elevado para o eixo z. Por fim, o Modelo 4 apresentou o menor erro em z ($1,8 \times 10^{-2}$ cm) e resultados consistentes nos demais eixos. Esses modelos foram treinados para aprender áreas específicas do espaço operacional do robô, com cada um apresentando maior precisão em diferentes regiões, contribuindo para uma cobertura mais abrangente do comportamento do manipulador.

6 CONCLUSÕES E TRABALHOS FUTUROS

Neste trabalho, buscou-se aproximar a cinemática inversa de um robô manipulador didático utilizando algoritmos de aprendizado de máquina, por meio das abordagens de modelos locais e globais. Especificamente, foram empregadas redes neurais MLP e o algoritmo *K-means* para os modelos locais, segmentando o espaço operacional. A fase de otimização de hiperparâmetros, tanto para o modelo global quanto para os modelos locais, foi necessário para melhorar a capacidade de predição e generalização. Esse processo envolveu ajustes nos parâmetros das redes neurais, como número de camadas ocultas, número de neurônios por camada e funções de ativação. Para os modelos locais, foi crucial determinar o número k de *clusters* que melhor dividisse o espaço operacional. Esses ajustes visaram obter a configuração mais adequada para prever os ângulos do robô.

A validação da cinemática direta, utilizando a parametrização D-H na biblioteca *Robotics Toolbox for Python*, comprovou o modelo teórico ao simular o comportamento do braço robótico. Os resultados demonstraram que a modelagem foi capaz de gerar corretamente as posições e orientações do efetuador, confirmando a adequação da implementação para simulações de manipulação robótica. As simulações mostraram correspondência entre os resultados teóricos e práticos, validando a eficácia da modelagem direta.

A geração da base de dados a partir da modelagem direta resultou em 6.890 amostras relevantes, após a aplicação de critérios de pré-processamento que eliminaram redundâncias e amostras inatingíveis. Essa abordagem garantiu que o conjunto de dados mantivesse a variabilidade necessária para o treinamento eficaz dos modelos de aprendizado de máquina, ao mesmo tempo em que reduziu o volume total de dados em $20,7\%$. A seleção de atributos, incluindo coordenadas cartesianas e ângulos de orientação, enriqueceu as informações disponíveis para os algoritmos, aumentando a eficiência no aprendizado e potencializando a precisão dos resultados, algo fundamental para a aproximação da cinemática inversa do robô manipulador didático.

No modelo global, foram conduzidos 2000 experimentos utilizando *Random Search*, com validação cruzada, o que permitiu identificar a arquitetura mais eficaz, composta por quatro camadas ocultas com ativações ReLU. O modelo alcançou um $R^2(\overline{\theta_i}) = 9,973 \times 10^{-1}$ e um $\text{MSE}(\overline{\theta_i}) = 2,70 \times 10^{-3}$, destacando-se por sua boa precisão preditiva. No entanto, ao aplicar o modelo a dados desconhecidos, observou-se que, embora generalizasse bem, o erro em graus das juntas revelou algumas limitações ao prever a posição no espaço operacional.

Para os modelos locais, o *K-means* foi empregado para segmentar o espaço operacio-

nal, identificando que $k = 4$ *clusters* proporcionava a melhor compactação e separação dos dados, conforme as métricas de inércia e coeficiente de silhueta. Após essa segmentação, os modelos de regressão local, também baseados em MLP, apresentaram um erro significativamente menor em comparação ao modelo global, principalmente ao prever os ângulos em áreas específicas do espaço operacional. Os modelos local apresentaram um $\text{MSE}(\theta_i) = 1.030 \times 10^{-3}$, com melhor desempenho na previsão do eixo z e resultados consistentes para os demais eixos cartesianos.

Os resultados obtidos demonstram que tanto o modelo global quanto os modelos locais possuem boa capacidade preditiva na resolução do problema de cinemática inversa do robô manipulador. O modelo global apresentou um bom desempenho, com valores satisfatórios de R^2 e MSE. No entanto, os modelos locais mostraram-se superiores ao lidar com regiões específicas do espaço operacional, alcançando maior precisão, especialmente em eixos cartesianos críticos, como o eixo z. Isso reforça a ideia de que a segmentação do espaço operacional é uma estratégia eficaz para aumentar a acurácia de modelos preditivos complexos, como no caso de robôs manipuladores.

6.1 Trabalhos Futuros

Com base nos resultados obtidos, diversas oportunidades de expansão e aprimoramento surgem para futuras pesquisas, visando aumentar a eficiência e robustez na solução da cinemática inversa do manipulador didático. Dentre as possíveis linhas de pesquisa para trabalhos futuros, destacam-se as seguintes:

1. **Ampliar a predição para incluir a orientação do efetuador**: Essa abordagem permitiria um controle mais completo e preciso do manipulador robótico, especialmente em aplicações que demandam manipulação de objetos, onde é essencial o controle preciso dos ângulos de rotação, além da posição.
2. **Explorar novos modelos de aprendizado de máquina**: Testar e comparar diferentes abordagens, como *Convolutional Neural Networks* (CNN), RNN, SVM e aprendizado por reforço, com o objetivo de melhorar a precisão das predições na cinemática inversa.
3. **Desenvolver a modelagem dinâmica do manipulador**: Incorporar a modelagem das forças e torques atuantes no sistema, a fim de aprimorar o controle de trajetória e posição, considerando as propriedades físicas do manipulador, como massas, inércias e atritos.
4. **Implementar os modelos em hardware**: Embarcar os modelos locais e globais no manipulador didático, possibilitando testes em tempo real, o que viabilizará ajustes e

otimizações no desempenho do sistema para operações em ambientes físicos reais.

REFERÊNCIAS

ALIMISIS, D. Educational robotics: Open questions and new challenges. **Themes in Science and Technology Education**, ERIC, v. 6, n. 1, p. 63–71, 2013.

ALLDATASHEET. **Electronic Components Datasheet Search**. 2003. Acessado: 14-09-2024. Disponível em: https://www.alldatasheet.com.

BENGIO, Y.; GOODFELLOW, I.; COURVILLE, A. **Deep Learning**. Cambridge, MA, USA: MIT Press, 2016. v. 1.

BERGSTRA, J.; BENGIO, Y. Random search for hyper-parameter optimization. **Journal of machine learning research**, v. 13, n. 2, 2012.

BEZERRA, S. G. T. de A. Reservoir computing com hierarquia para previsão de vazões médias diárias. 2016.

BISHOP, C. M. **Pattern Recognition and Machine Learning**. New York, NY, USA: Springer, 2006.

BOTTOU, L.; VAPNIK, V. Local learning algorithms. **Neural computation**, MIT Press, v. 4, n. 6, p. 888–900, 1992.

CHANDOLA, V.; BANERJEE, A.; KUMAR, V. Anomaly detection: A survey. **ACM computing surveys (CSUR)**, ACM New York, NY, USA, v. 41, n. 3, p. 1–58, 2009.

CORKE, P. **Robotics, Vision and Control**. Berlin Heidelberg: Springer, 2017.

CORKE, P.; HAVILAND, J. Not your grandmother's toolbox–the robotics toolbox reinvented for python. In: IEEE. **2021 IEEE International Conference on Robotics and Automation (ICRA)**. [*S. l.*], 2021. p. 11357–11363.

CRAIG, J. J. **Introduction to robotics**. USA: Pearson Educacion, 2006.

DEMBY'S, J.; GAO, Y.; DESOUZA, G. N. A study on solving the inverse kinematics of serial robots using artificial neural network and fuzzy neural network. USA, p. 1–6, 2019.

DENAVIT, J.; HARTENBERG, R. S. A kinematic notation for lower-pair mechanisms based on matrices. American Society of Mechanical Engineers, 1955.

ESTEVA, A.; KUPREL, B.; NOVOA, R. A.; KO, J.; SWETTER, S. M.; BLAU, H. M.; THRUN, S. Dermatologist-level classification of skin cancer with deep neural networks. **nature**, Nature Publishing Group, v. 542, n. 7639, p. 115–118, 2017.

FACELI, K.; LORENA, A. C.; GAMA, J.; CARVALHO, A. C. d. **Inteligência Artificial: uma abordagem de aprendizado de máquina**. Rio de Janeiro: Editora LTC, 2011.

FONTINELE, H. Í. P. **Modelos locais para aproximação da cinemática inversa de robôs redundantes: um estudo comparativo**. 2015.

GAO, R. Inverse kinematics solution of robotics based on neural network algorithms. **Journal of Ambient Intelligence and Humanized Computing**, Springer, v. 11, n. 12, p. 6199–6209, 2020.

GRAVES, A.; MOHAMED, A.-r.; HINTON, G. Speech recognition with deep recurrent neural networks. In: IEEE. **2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)**. Vancouver, Canada, 2013. p. 6645–6649.

HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. H. **The elements of statistical learning: data mining, inference, and prediction**. New York: Springer, 2009. v. 2.

HAYKIN, S. *et al.* Neural networks and learning machines pearson upper saddle river. **NJ, USA**, v. 3, 2009.

HORNIK, K.; STINCHCOMBE, M.; WHITE, H. Multilayer feedforward networks are universal approximators. **Neural networks**, Elsevier, v. 2, n. 5, p. 359–366, 1989.

HUANG, S.; TAN, K. K.; LEE, T. H. Decentralized control design for large-scale systems with strong interconnections using neural networks. **IEEE Transactions on Automatic Control**, IEEE, v. 48, n. 5, p. 805–810, 2003.

ICHI. **Rede Neural Centroid: Um Algoritmo de Clustering Eficiente e Estável**. 2020. Acessado: 15-09-2024. Disponível em: https://ichi.pro/pt/rede-neural-centroid-um-algoritmo-de-clustering-eficiente-e-estavel-196789172644391.

IFCE, R. **Projeto de robótica em instituições de ensino fundamental e médio**. 2022. Acessado: 14-09-2024. Disponível em: https://robotica.ifce.edu.br/.

ISO. **ISO 8373: Robots and robotic devices**. Geneva, Switzerland, 2012.

JAIN, A. K. Data clustering: 50 years beyond k-means. **Pattern recognition letters**, Elsevier, v. 31, n. 8, p. 651–666, 2010.

JESUS, R. C. d. O. Desenvolvimento de um controlador cinemático com ponderação por parâmetros dinâmicos para robôs manipuladores. Universidade Federal de Sergipe-Pró-reitoria de Pós-Graduação e Pesquisa , 2019.

JUNIOR, F. E. F. **Estudo e implementação de redes neurais e algoritmos genéticos para resolução de cinemática inversa de um manipulador robótico com 5 graus de liberdade**. Tese (Doutorado) – [UNICAMP], 2014.

LIM, D.-W.; LEE, Y.-K. On the number of training samples for inverse kinematics solutions by artificial neural networks. In: IEEE. **2019 16th International Conference on Ubiquitous Robots (UR)**. Jeju, South Korea, 2019. p. 61–64.

MELO, D. B. **Algoritmos de aprendizagem para aproximação da cinemática inversa de robôs manipuladores: um estudo comparativo**. 2015.

METAQUOTES. **Redes Neurais Profundas (Parte IV). Criação, treinamento e teste de um modelo de rede neural**. 2020. Acessado: 14-09-2024. Disponível em: https://www.mql5.com/pt/articles/3473.

NIKU, S. B. **Introduction to robotics: analysis, control, applications**. USA: John Wiley & Sons, 2020.

NUNES, R. F. Mapeamento da cinemática inversa de um manipulador robótico utilizando redes neurais artificiais configuradas em paralelo. Universidade Estadual Paulista (Unesp), 2016.

OGAWA, T.; KANADA, H. Solution for ill-posed inverse kinematics of robot arm by network inversion. **Journal of Robotics**, Hindawi, v. 2010, 2010.

OLIVEIRA, A. S. Retrofitting de robôs manipuladores com incorporação de controle de posição e força: aplicação em um robô industrial. Florianópolis, SC, 2007.

RIASCOS, L. A. **Fundamentos de robótica**. São Paulo: Plêiade, 2010.

ROTH, B. Performance evaluation of manipulators from a kinematic viewpoint. **NBS Special Publication**, v. 459, p. 39–62, 1976.

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Learning representations by back-propagating errors. **nature**, Nature Publishing Group UK London, v. 323, n. 6088, p. 533–536, 1986.

RUSSELL, S.; NORVIG, P. **Artificial Intelligence: A Modern Approach**. 3rd. ed. Boston, MA, USA: Pearson, 2016.

RUSSELL, S. J.; NORVIG, P. **Inteligência artificial**. Rio de Janeiro: Elsevier, 2013.

SICILIANO, B.; SCIAVICCO, L.; VILLANI, L.; ORIOLO, G. **Robotics Modelling, Planning and Control**. Berlin: Springer, 2009.

SPONG, M. W.; HUTCHINSON, S.; VIDYASAGAR, M. **Robot modeling and control**. USA: John Wiley & Sons, 2020.

ZHANG, L.; XIAO, N. A novel artificial bee colony algorithm for inverse kinematics calculation of 7-dof serial manipulators. **Soft Computing**, Springer, v. 23, p. 3269–3277, 2019.