Universidade Federal do Ceará – UFC

Departamento de Engenharia de Teleinformática

Programa de Pós-Graduação em Engenharia de Teleinformática

Davyd Bandeira de Melo

ALGORITMOS DE APRENDIZAGEM PARA APROXIMAÇÃO DA CINEMÁTICA INVERSA DE ROBÔS MANIPULADORES: UM ESTUDO COMPARATIVO

Fortaleza/CE

2015

Davyd Bandeira de Melo

ALGORITMOS DE APRENDIZAGEM PARA APROXIMAÇÃO DA CINEMÁTICA INVERSA DE ROBÔS MANIPULADORES: UM ESTUDO COMPARATIVO

Dissertação submetida à Coordenação do Curso de Pós-Graduação em Engenharia de Teleinformática, da Universidade Federal do Ceará, como parte dos requisitos exigidos para obtenção do grau de Mestre em Engenharia de Teleinformática.

Prof. Dr. Guilherme de Alencar Barreto

Fortaleza/CE

2015

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca de Pós-Graduação em Engenharia - BPGE

M477a Melo, Davyd Bandeira de.
Algoritmos de aprendizagem para aproximação da cinemática inversa de robôs manipuladores: um estudo comparativo / Davyd Bandeira de Melo. – 2015.
133 f. : il. color.

Dissertação (mestrado) – Universidade Federal do Ceará, Centro de Tecnologia, Departamento de Engenharia de Teleinformática, Programa de Pós-Graduação em Engenharia de Teleinformática, Fortaleza, 2015.
Área de concentração: Sinais e Sistemas.
Orientação: Prof. Dr. Guilherme de Alencar Barreto.

1. Teleinformática. 2. Redes neurais. 3. Máquinas - Aprendizagem. 4. Cinemática. I. Título.

CDD 621.38

UNIVERSIDADE FEDERAL DO CEARÁ
CENTRO DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA DE TELEINFORMÁTICA

CAMPUS DO PICI, CAIXA POSTAL 6007 CEP 60.738-640
FORTALEZA – CEARÁ - BRASIL
FONE (+55) 85 3366-9467 – FAX (+55) 85 3366-9468

DAVYD BANDEIRA DE MELO

ALGORITMOS DE APRENDIZADO DE MÁQUINAS PARA APROXIMAÇÃO DA CINEMÁTICA INVERSA DE ROBÔS MANIPULADORES: UM ESTUDO COMPARATIVO

Dissertação submetida à Coordenação do Programa de Pós-Graduação em Engenharia de Teleinformática, da Universidade Federal do Ceará, como requisito parcial para a obtenção do grau de Mestre em Engenharia de Teleinformática.
Área de concentração: Sinais e Sistemas.

Aprovada em: 06/07/2015.

BANCA EXAMINADORA

Guilherme de Alencar Barreto
Prof. Dr. GUILHERME DE ALENCAR BARRETO (Orientador)
Universidade Federal do Ceará

Anselmo Frizera Neto
Prof. Dr. ANSELMO FRIZERA NETO
Universidade Federal do Espírito Santo

Prof. Dr. AUZUIR RIPARDO DE ALEXANDRIA
Instituto Federal de Educação, Ciência e Tecnologia do Ceará

Este trabalho é dedicado à minha família,
que sempre me ajudou e incentivou a trilhar uma vida
aplicada aos estudos.

Agradecimentos

Agradeço primeiramente a Deus por todas as oportunidades e bençãos concedidas. Certamente, Ele foi o maior responsável por iluminar o caminho, retirando obstáculos muitas vezes considerados intransponíveis.

Agradeço a minha família, especialmente as minhas mães Maria Serly Bandeira Bezerra e Cândida Maria Bandeira Bezerra, pelo amor incondicional, pela dedicação e por toda educação que não se encontra em nenhum livro. Elas constituem os pilares da minha vida e os maiores exemplos de caráter e superação que conheço. Faço delas um espelho todos os dias.

Agradeço ao amor da minha vida, Samara Semião Freitas, pelo amor, compreensão, confiança e cumplicidade. Nos momentos em que mais precisei, ela sempre esteve lá, me aconselhando, incentivando e sendo, certamente, a maior fonte inspiradora não apenas desta dissertação mas de toda minha vida. Sua presença possibilitou uma trajetória leve, tranquila e recompensadora.

Agradeço ao meu orientador, Prof. Guilherme de Alencar Barreto, pelas valiosas lições transmitidas. Por várias vezes sua compreensão e paciência foram essenciais durante a minha jornada de conciliação entre emprego e pesquisa acadêmica.

Aos professores e colegas do grupo de pesquisa do Centro de Referência em Automação e Robótica (CENTAURO), que sempre estiveram disponíveis para sanar dúvidas e transmitir conhecimento.

Aos colegas de trabalho do Departamento de Engenharia de Teleinformática (DETI), que de diversas formas contribuíram para o desenvolvimento deste trabalho.

"Não vos amoldeis às estruturas deste mundo,
mas transformai-vos pela renovação da mente,
a fim de distinguir qual é a vontade de Deus:
o que é bom, o que Lhe é agradável, o que é perfeito.
(Bíblia Sagrada, Romanos 12, 2)

Resumo

Nesta dissertação são reportados os resultados de um amplo estudo comparativo envolvendo sete algoritmos de aprendizado de máquinas aplicados à tarefa de aproximação do modelo cinemático inverso de 3 robôs manipuladores (planar, PUMA 560 e Motoman HP6). Os algoritmos avaliados são os seguintes: Perceptron Multicamadas (MLP), Máquina de Aprendizado Extremo (ELM), Regressão de Mínimos Quadrados via Vetores-Suporte (LS-SVR), Máquina de Aprendizado Mínimo (MLM), Processos Gaussianos (PG), Sistema de Inferência Fuzzy Baseado em Rede Adaptativa (ANFIS) e Mapeamento Linear Local (LLM). Estes algoritmos são avaliados quanto à acurácia na estimação dos ângulos das juntas dos robôs manipuladores em experimentos envolvendo a geração de vários tipos de trajetórias no volume de trabalho dos referidos robôs. Uma avaliação abrangente do desempenho de cada algoritmo é feito com base na análise dos resíduos e testes de hipóteses são executados para verificar se há diferenças significativas entre os desempenhos dos melhores algoritmos.

Palavras-chaves: Cinemática Inversa, Robôs Manipuladores, Redes Neurais Artificiais, Least Squares Support Vector Regression, Minimal Learning Machine, Extreme Learning Machine.

Abstract

In this dissertation it is reported the results of a comprehensive comparative study involving seven machine learning algorithms applied to the task of approximating the inverse kinematic model of 3 robotic arms (planar, PUMA 560 and Motoman HP6). The evaluated algorithm are the following ones: Multilayer Perceptron (MLP), Extreme Learning Machine (ELM), Least Squares Support Vector Regression (LS-SVR), Minimal Learning Machine (MLM), Gaussian Processes (GP), Adaptive Network-Based Fuzzy Inference Systems (ANFIS) and Local Linear Mapping (LLM). Each algorithm is evaluated with respect to its accuracy in estimating the joint angles given the cartesian coordinates which comprise end-effector trajectories within the robot workspace. A comprehensive evaluation of the performances of the aforementioned algorithms is carried out based on correlation analysis of the residuals. Finally, hypothesis testing procedures are also executed in order to verifying if there are significant differences in performance among the best algorithms.

Key-words: Inverse Kinematics, Manipulators, Artificial Neural Networks, Least Squares Support Vector Regression, Minimal Learning Machine, Extreme Learning Machine.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

ANFIS	*Adaptive Network Fuzzy Inference System*
ARX	*AutoRegresive model with eXternal input*
DH	Parâmetros de Denavit-Hartenberg
DOF	*Degrees of Freedom*
ELM	*Extreme Learning Machine*
EQM	Erro Quadrático Médio
FAC	Função de Autocorrelação
FDA	Função de Distribuição Acumulada
FIS	*Fuzzy Inference System*
ISO	*International Organization for Standardization*
KS	Teste de hipóteses de *Kolmogorov-Smirnov*
LBG	Algoritmo de *Linde-Buzo-Gray*
LLM	*Linear Local Mapping*
LMS	*Least Mean Squares*
LS-SVR	*Least-Squares Support Vector Regression*
MIMO	*Multiple Input Multiple Output*
MISO	*Multiple Inputs Single Outputs*
MLM	*Minimal Learning Machine*
MLP	*Multi Layer Perceptron*
MS	Métrica de Seleção
PG	Processo Gausiano
PUMA	*Programmable Universal Machine for Assembly*
SOM	*Self-Organizing Map*
IFR	*International Federation of Robotics*

Lista de símbolos

$\mathcal{C}$	Espaço de configurações
$\mathcal{W}$	Espaço da tarefa
n_{DOF}	Números de graus de liberdade
$n_{DOF}^{(w)}$	Números de graus de liberdade antes do pulso
$\boldsymbol{\theta}$	Vetor cujas componentes são os ângulos de juntas
$\mathbf{f}$	Mapeamento não-linear desconhecido
$\mathbf{f}^{-1}$	Mapeamento inverso não-linear desconhecido
$\mathbf{x}_d$	Posição do efetuador utilizado como padrão de treinamento
$\{A\}$	Sistema de referência A
${}^{A}P$	Descrição de um ponto P com respeito a um sistema de referência $\{A\}$
${}^{A}_{B}R$	Matriz de rotação que descreve a orientação de $\{B\}$ em relação a $\{A\}$
${}^{A}P_{BORG}$	Vetor que localiza a origem do sistema $\{B\}$ em relação ao sistema $\{A\}$
${}^{A}_{B}T$	Matriz de transformação homogênea
s_i	Seno do ângulo θ_i
c_i	Cosseno do ângulo θ_i
s_{ij}	Seno do ângulo $\theta_i + \theta_j$
c_{ij}	Cosseno do ângulo $\theta_i + \theta_j$
w_{ij}	Peso sináptico da j-ésima entrada do i-ésimo neurônio oculto
$\mathbf{w}_i$	Vetor de pesos do i-ésimo neurônio oculto
$\mathbf{W}$	Matriz dos pesos da camada oculta
m_{ij}	Peso sináptico da j-ésima entrada do i-ésimo neurônio de saída
$\mathbf{m}$	Vetor de pesos do neurônio de saída
$\mathbf{x}$	Padrão de entrada

$\mathbf{X}$	Matriz reunindo N_1 padrões de entrada
h	Número de neurônios na camada oculta
d	Dimensão dos padrões de entrada
g	Número de neurônios na camada de saída
u_i	Ativação do i-ésimo neurônio
$\phi(\cdot)$	Função de ativação
z_i	Saída do i-ésimo neurônio oculto
y_k	Saída do k-ésimo neurônio de saída
α	Taxa de aprendizagem
η	Fator de momento
$k(x_i, x_j)$	Função de *kernel*
$\Im$	Espaço das características (*feature space*)
$\varphi(\cdot)$	Mapeamento para o espaço das características
$\mathbf{D}_x$	Matriz de distâncias no espaço de entrada
$\mathbf{\Delta}_y$	Matriz de distâncias no espaço de saída
$J(\cdot)$	Função de custo
V_i	i-ésima região de Voronoi
$\mathbf{a}_i$	Vetor de coeficientes do i-ésimo modelo linear
n_q	Quantidade de elementos do conjunto de parâmetros
n_r	Número de rodadas de treinamento
ξ_{ij}	EQM da i-ésima rodada para o j-ésimo valor do hiperparâmetro q
j^*	Índice de coluna que representa os parâmetros de menor média sobre as n_r rodadas de treinamento
i^*	Índice de linha que representa a melhor rodada de treinamento para os parâmetros escolhidos por j^*
$\xi_{i^*j^*}$	Menor valor de erro de treinamento para o melhores conjuntos de parâmetros e rodadas de treinamento

$\bar{d}$	Distância médias entre os pontos das as trajetórias desejadas e estimadas
H_0	Hipótese nula
$\hat{F}$	Função de Distribuição Acumulada empírica

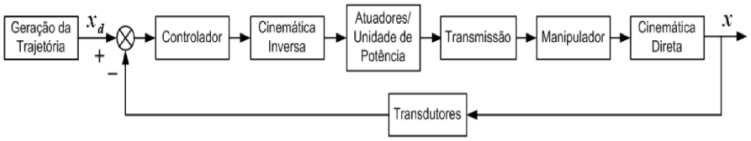
Sumário

1 Introdução

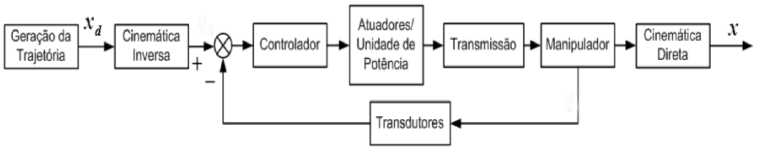
A *International Federation of Robotics* (IFR) estima que o ano de 2014 foi de longe aquele que apresentou o maior volume de vendas de robôs industriais, contabilizando mais de 200 mil robôs instalados em todo o mundo (IFR, 2013 (Acessado em 27/06/2015)). Isso representa em torno de 15% a mais do que o ano anterior, que já tinha sido considerado recorde após uma leve desaceleração nos investimentos no setor em 2012. Todo este crescimento foi alavancado principalmente pelas indústrias automobilística e eletro-eletrônica. A mesma federação estimou que, no final do ano de 2013, estariam funcionando aproximadamente 1,5 milhões de unidades ao redor do mundo. Também de acordo com esta pesquisa, nosso país lidera na aquisição de robôs industriais no continente sul-americano, adquirindo cerca de 1300 unidades até o final de 2013. A IFR também disponibiliza estatísticas bastante promissoras para o mercado de robôs de serviço presentes tanto no âmbito pessoal quanto profissional. Estes números demonstram, de forma concreta, o quão promissor tem sido o setor de robótica e apontam para um contínuo avanço na pesquisa e no desenvolvimento de tecnologia para esta área.

Um manipulador industrial pode ser pensado como um conjunto de corpos rígidos, denominados elos, ligados através de juntas rotacionais ou prismáticas. Ao final da cadeia de elos, em um região chamada de pulso, normalmente está acoplado um efetuador que desempenha de fato as atividades para as quais aquele robô foi designado. Por exemplo, uma atividade comum na indústria é a soldagem de chapas metálicas que é realizada através de um efetuador que produz um arco elétrico e que está acoplado ao pulso. Outro exemplo clássico de efetuador é aquele que apresenta formato de pinça, sendo útil para posicionar ou acoplar objetos. Em muitos casos, o efetuador pode ser substituído na medida que o manipulador poderá realizar atividades diferentes ao longo do tempo. Normalmente esta capacidade de atender múltiplos propósitos é uma característica desejável de um manipulador e sua importância é comprovada na definição dada pela ISO 8373 (ISO, 2012).

Para uma atividade específica, é definido para um manipulador industrial um espaço no qual ele realizará esta tarefa, chamado de *espaço da tarefa* (**task space**). Dentro deste espaço, o efetuador segue trajetórias pré-determinadas que são executadas dentro de um intervalo de tempo. Para realizar o planejamento destas trajetórias, faz-se necessário ter conhecimento das relações entre os atuadores presentes nas juntas e posição resultante do efetuador no seu *espaço de trabalho* (**workspace**). Por exemplo, para um robô de três graus de liberdade (*degrees of freedom* - DOF) pode-se ter três motores fazendo o papel das juntas rotacionais e a posição atual do eixo de rotação (ângulo de junta) de cada um dos motores resulta em uma posição cartesiana diferente.



(a) Esquema de controle no espaço operacional.



(b) Esquema de controle no espaço das juntas.

Figura 1 – Diagrama de blocos ilustrando os tipos de estratégias de controle de robôs manipuladores. Adaptado de Oliveira (2007).

1.1 Contextualização do Problema

Em robótica, Cinemática é o estudo do movimento de robôs sem levar em conta as forças e torques que o geram. Um dispositivo eletrônico chamado *controlador* realiza o movimento do manipulador através da geração do sinal de controle (i.e., comando dos atuadores) que tem origem na diferença entre a posição desejada e a posição atual do robô. Dessa forma, são necessárias leituras do estado do sistema para fins de controle.

As medidas fundamentais de um sistema robótico são as posições das juntas do robô, também chamadas de variáveis de junta. No caso de uma junta rotacional, a variável corresponde ao deslocamento angular (ou ângulo de rotação). No caso de uma junta prismática, a variável é o deslocamento linear.

A tarefa de interesse para um sistema de controle de robôs consiste na geração da trajetória das juntas, i.e. determinar uma sequência de posições e orientações ao longo do tempo que permitam a realização adequada de uma dada tarefa. Uma particularidade no controle de robôs manipuladores é que este pode ser realizado em dois espaços distintos, ou seja, no espaço das juntas ou no espaço operacional. Na Figura 1 são mostrados os diagramas de bloco para estes dois esquemas de controle.

Quando o controle se dá no espaço operacional (Figura 1a), as ações de controle são realizadas em relação à posição cartesiana da extremidade do manipulador (i.e., efetuador). Desta forma, os controladores corrigem os sinais de posição. Já para o controle no espaço das juntas (Figura 1b), as ações de controle são realizadas neste espaço de tal modo que os controladores corrigem sinais de posição angular.

Porém, independente do esquema de controle utilizado, o acionamento efetivo do manipulador sempre acontece no espaço das juntas, pois são os atuadores (e.g. motores elétricos) que realizam os movimentos em última instância, tornando necessária uma

transformação que relaciona as variáveis do espaço operacional (i.e. coordenadas cartesianas) às variáveis no espaço das juntas. Contudo, mesmo que os acionamentos sejam realizados no espaço das juntas, faz-se necessário converter as posições angulares das juntas de volta para o espaço operacional do robô, onde a trajetória do efetuador é efetivamente realizada e avaliada.

Determinar as transformações entre os espaços operacional e das juntas é o objetivo da cinemática de robôs. A depender do sentido da transformação tem-se dois tipos de cinemática: direta e inversa. Posto de modo simples, a cinemática direta transforma do espaço das juntas ao espaço operacional, ou seja, permite a conversão de ângulos das juntas em posição cartesiana e orientação do efetuador do robô. A conversão oposta é chamada de cinemática inversa.

De modo mais formal, quando se necessita obter a posição cartesiana do efetuador a partir das variáveis de junta, tem-se um problema de cinemática direta. A função de cinemática direta é um mapeamento contínuo

$$\mathbf{f} : \mathcal{C} \subseteq \mathbf{\Theta}^n \rightarrow \mathcal{W} \subseteq \mathcal{X}^m \tag{1.1}$$

que mapeia um conjunto de n variáveis das juntas no espaço $\mathcal{C}$ (espaço de configurações) no espaço da tarefa $\mathcal{W}$ de dimensão m. Se $m \leq n$, o robô é dito redundante[1].

Como ilustrado na Figura 1, metas de controle, tais como posicionamento e orientação do efetuador são especificadas em termos de coordenadas no espaço de tarefas. Contudo, o manipulador é tipicamente controlado no espaço das juntas. Desta forma, é importante ser capaz de encontrar algum $\boldsymbol{\theta} \in \mathcal{C}$ tal que $\mathbf{f}(\boldsymbol{\theta})$ resulte na posição e orientação desejada $\mathbf{x}_d$ do efetuador no espaço da tarefa. Este é o problema da cinemática inversa.

O problema da cinemática inversa é mal-posto[2] (*ill-posed*) (OGAWA; KANADA, 2010; DeMers; Kreutz-Delgado, 1992). Se há DOFs redundantes, então o problema é *localmente* mal-posto, porque a solução não é única e consiste em uma variedade[3] não-trivial em $\mathcal{C}$. Com ou sem DOFs redundantes, o problema é em geral *globalmente* mal-posto por causa da existência de um conjunto finito de ramos (*branches*) de solução, o que implica que existirão múltiplas configurações que resultarão na mesma localização no espaço de tarefas. Desta forma, a determinação de uma única função inversa é problemática devido à natureza *muitos-para-um* do mapeamento $\mathbf{f}$.

O problema inverso pode ser resolvido explicitamente, ou seja, em forma fechada, apenas para certos tipos de manipuladores. São deste tipo, por exemplo, manipuladores

[1] Robô como mais graus de liberdade que o estritamente necessário à realização de uma dada tarefa

[2] De acordo com Tikhonov e Arsenin (1977), para um problema ser bem-posto deve ter as seguintes propriedades: (*i*) Existência de solução. (*ii*) Unicidade da solução. (*iii*) A solução depende continuamente das condições iniciais e de contorno, ou seja, pequenas mudanças nas condições iniciais e de contorno devem causar pequenas mudanças na solução.

[3] Geralmente de dimensão $n - m$.

do tipo antropomórficos de 6 DOFs e com efetuador separável do braço/antebraço, em que as primeiras três juntas são usadas para posicionamento e as três últimas para orientação, tal como o robô PUMA 560 (CRAIG, 2005).

Uma primeira alternativa a uma solução em forma fechada é a solução numérica, geralmente usando ou o inverso da matriz jacobiana, que é, na verdade, uma abordagem do tipo Newton, ou gradiente descendente (também um método baseado na matriz jacobiana). Estes métodos, chamados de iterativos, são ensinados em livros-texto introdutórios na área de robótica de manipuladores (CRAIG, 2005; SPONG; HUTCHINSON; VIDYASAGAR, 2006), tendo como principal desvantagem o fato de possuir custo elevado devido à determinação da matriz jacobiana (e de sua inversa) ou do vetor gradiente da função $\mathbf{f}$. Por este motivo, tais métodos não se adequam bem a aplicações de controle em tempo real.

Uma segunda alternativa, que vem sendo explorada desde o final da década de 1980 até os dias atuais, é o uso de redes neurais artificiais (RNAs) (HORNE; JAMSHIDI; VADIEE, 1990; PRABHU; GARG, 1996; BARRETO; ARAÚJO; RITTER, 2003). Modelos de RNAs, devido a bem conhecidas propriedades de aproximação universal de funções (HAYKIN, 2009) podem ser usadas para estimar diretamente a função inversa $\mathbf{f}^{-1}$. Isto é realizado por meio da geração de um conjunto de dados contendo um número elevado de posturas aleatórias no espaço de configuração do robô de interesse, armazenando-se os pares $\{(\boldsymbol{\theta}, \mathbf{x}_d)\} \in \mathcal{C} \times \mathcal{W}$ para posterior treinamento dos modelos de RNAs escolhidos. Usando $\mathbf{x}_d$ como vetor de entrada da rede e $\boldsymbol{\theta}$ como vetor de saída, pode-se treinar um modelo de rede neural para aproximar a função de cinemática inversa com grau de precisão arbitrário. Tipicamente os dados são gerados restringido-se a soluções únicas: para cada $\mathbf{x}_d$) existe uma única solução $\boldsymbol{\theta}$, a fim de sobrepujar o problema das múltiplas soluções possíveis[4].

Uma das vantagens do uso de redes neurais em problemas de controle de robôs reside na possibilidade de dotá-los de maior autonomia. Em outras palavras, pode-se fazer com que robôs sejam capazes de interpretar dados sensoriais adquiridos no ambiente de trabalho, interpretá-los e reagir de forma apropriada, seja em operação normal ou em situações inesperadas. Para que um dado manipulador seja autônomo, este deve ser capaz de aprender através dos dados coletado no ambiente. Tal aprendizado garantirá que mudanças nas condições ambientais serão percebidas e o manipulador se ajustará às novas condições de operação. Através do aprendizado permanente, problemas tais como envelhecimento de componentes eletrônicos, desgaste das juntas, dentre outros; são naturalmente incluídas no novo modelo que deve ser reestimado periodicamente.

Uma desvantagem consiste na dificuldade de se gerar um conjunto de dados com

[4] Tal restrição pode não ser possível ou desejável em algumas aplicações, e pode reduzir consideravelmente a destreza e a *manipulabilidade* do braço robótico. Para os leitores interessados no tratamento simultâneo de múltiplas soluções usando RNAs, recomenda-se a leitura do trabalho de DeMers e Kreutz-Delgado (1992). O tratamento deste tipo de situação está fora do escopo desta dissertação.

amostras em quantidade suficiente grande para representar adequadamente o volume do trabalho do robô e que evite ao mesmo tempo situações com múltiplas soluções e que contenham singularidades.

1.2 Motivação

Embora o uso de redes neurais no controle de robôs manipuladores seja prática relativamente bem disseminada na área de sistemas inteligentes, e seja fácil encontrar referências atuais nesta área de estudo, é raro encontrar na literatura especializada estudos comparativos de desempenho que envolvam as diversas arquiteturas de redes neurais disponíveis, sejam clássicas ou novas. Este fato foi um dos principais motivadores para o desenvolvimento da presente dissertação.

Outro motivo para o desenvolvimento deste trabalho foi a possibilidade de avaliar diferentes paradigmas de construção do modelo neural, a saber, *global* e *local*. A diferença entre estas abordagens ficará mais clara nos próximos capítulos, mas pode-se adiantar que a principal diferença reside no fato de o paradigma local particiona o espaço operacional em subregiões, sendo que a cada subregião associa-se um modelo mais simples (e.g. linear) que usa apenas os dados relativos àquela subregião. Modelos globais aproximam a função $\mathbf{f}^{-1}$ usando todo o conjunto de dados.

A fim de dar um caráter mais amplo e abrangente a este trabalho de pesquisa, resolveu-se por não só comparar os desempenhos de alguns modelos de RNAs clássicos, tais como a rede perceptron multicamadas (*multilayer perceptron*, MLP) (RUMELHART; HINTON; WILLIAMS, 1985) e o mapa auto-organizável (*self-organizing map*, SOM) (KOHONEN, 2013), mas também os de arquiteturas de redes neurais mais recentes, tal como a máquina de aprendizado extremo (*extreme learning machine*, ELM) (HUANG; ZHU; SIEW, 2006), assim como os desempenhos de algoritmos mais modernos de aprendizado de máquinas, tais como modelos de processos gaussianos (*Gaussian process models*, GP) (RASMUSSEN, 2006), regressão de mínimos quadrados via vetores-suporte (*least-squares support vector regression*, LS-SVR) (SUYKENS et al., 2002), e a recente máquina de aprendizado mínimo (*minimal learning machine*, MLM) (JUNIOR et al., 2013). Para fins de completude, inclui-se também nos estudos comparativos um modelo neurofuzzy de amplo uso na comunidade de inteligência computacional, que é o sistema de inferência fuzzy baseado em rede adaptativa (*Adaptive network fuzzy inference system*, ANFIS) (JANG, 1993).

Os sete modelos mencionados no parágrafo anterior cobrem diferentes paradigmas de aprendizado de máquinas. Por exemplo, os modelos MLP, ELM e LLM são arquiteturas de redes neurais artificiais, enquanto o modelo ANFIS é uma arquitetura híbrida neurofuzzy. Já os modelos GP, LS-SVR e MLM podem ser enquandrados como pertencentes ao

paradigma de máquinas de *kernel*.

Por fim, vale ressaltar que um diferencial do estudo comparativo a ser reportado nesta dissertação em relação é seguir uma abordagem ao problema de aproximação de um mapeamento MIMO (*multi-input, multi-output*) distinta da comumente usada para aprendizado da cinemática inversa via RNAs e outros métodos de aprendizado de máquinas. Em vez de usar uma única rede com múltiplas saídas, optou-se por treinar um modelo por variável de junta. Ou seja, se o manipulador tiver $n = 6$ graus de liberdade, serão treinadas 6 redes MLP, uma por junta de saída. Assim, o problema MIMO é decomposto em n subproblemas MISO (*multi-input, single-output*).

Nesta abordagem, embora aumente o custo de seleção dos modelos em alguns casos, ou seja, é preciso estimar os hiperparâmetros (e.g. número de neurônios ocultos) e parâmetros (pesos sinápticos e limiares) de cada submodelo, a complexidade de cada submodelo tende a diminuir porque a complexidade da tarefa diminui. Isto está em sintonia com abordagens mais modernas para controle neural em tempo-real de robôs manipuladores, tal como o controle descentralizado (JIN, 1998; LIU, 1999; HUANG; TAN; LEE, 2003).

1.3 Objetivos Gerais e Específicos

Em virtude do exposto na seção anterior, o objetivo principal desta dissertação é promover um amplo estudo comparativo de desempenho entre modelos de aproximação universal de funções construídos a partir de diferentes paradigmas de aprendizado (e.g. neural, neurofuzzy e máquinas de *kernel*) visando a estimação da cinemática inversa de robôs manipuladores.

Quanto aos objetivos específicos desta dissertação, podem-se listar os seguintes itens:

- Gerar conjuntos de dados para fins de treinamento dos modelos de aprendizado de máquina cujo os desempenhos serão avaliados nesta dissertação na tarefa de aproximação da cinemática inversa de três robôs manipualdores: planar, PUMA 560 e Motoman HP6.
- Comparar o desempenho dos modelos de aprendizado de máquina supracitados com base nos erros médios quadráticos gerados no espaço das juntas dos robôs manipuladores planar, PUMA 560 e Motoman HP6.
- Identificar os modelos mais adequados para aproximação da cinemática inversa através de uma combinação de erros médios quadráticos que representará o comportamento global dos algoritmos.

- Avaliar os desempenhos dos modelos supracitados em tarefas que envolvam a realização de trajetórias específicas.

- Realizar uma análise estatística dos resíduos gerados pelos melhores modelos para fins de validação dos modelos.

- Realizar testes de hipóteses estatísticos a fim de avaliar quão diferentes são os desempenhos dos melhores modelos encontrados.

1.4 Produção Científica

Ao longo do desenvolvimento desta dissertação, foram publicados ou submetidos a congressos ou periódicos os seguintes artigos:

1. **Melo, D. B.** & Barreto, G. A. (2014). "Algoritmos de Aprendizagem para Aproximação da Cinemática Inversa de Estruturas Robóticas: Um Estudo Comparativo", Anais do *XXXV Ibero Latin American Congress on Computational Methods in Engineering* (CILAMCE'2014), Fortaleza-CE, páginas 1-22.

2. Fontinele, H. I. P., **Melo, D. B.** & Barreto, G. A. (2015). "Local Models for Learning Inverse Kinematics of Redundant Robots: A Performance Comparison", submetido ao *11th Workshop on Self-Organizing Maps* (WSOM'2016).

1.5 Resumo dos Capítulos Restantes

O restante deste documento está organizado segundo a lista de capítulos apresentada abaixo:

- **Capítulo 2** - Este capítulo descreve os fundamentos que compõem a robótica dos manipuladores industriais e móveis. Serão apresentados os parâmetros de Denavit-Hartenberg (DH), que descrevem a arquitetura geométrica de um manipulador industrial, e uma convenção através da qual estes parâmetros podem ser obtidos. Além disso, serão apresentados os conceitos de cinemática direta e inversa, que, juntamente com os algoritmos de aprendizagem de máquina, formam a base desta dissertação.

- **Capítulo 3** - Neste capítulo serão apresentados os diversos modelos de regressão adotados. Estes algoritmos advém das mais variadas naturezas apresentando desta forma formulações e características bem distintas. São elas as redes neurais, as arquiteturas neuro-fuzzy, as máquinas de *kernel* e os métodos baseados em distâncias.

- **Capítulo 4** - Neste momento do trabalho, uma ponte é criada entre os tópicos apresentados nos **Capítulos 2** e **3**. Descrevem-se as simulações realizadas bem como o conjunto de passos sistemáticos adotados para obtenção dos resultados, incluindo a descrição dos conjuntos de dados extraídos das equações cinemáticas dos manipuladores. Tabelas e gráficos ilustram o desempenho dos modelos de regressão adotados para cada conjunto de dados apresentados.

- **Capítulo 5** - Este capítulo apresenta os resultados dos testes de hipóteses aplicados aos resíduos gerados pelos melhores modelos de regressão escolhidos no **Capítulo 4**. Além do mais, são apresentados gráficos da função de autocorrelação dos resíduos com o objetivo de validar estatisticamente tais modelos.

- **Capítulo 6** - Este capítulo finaliza o presente trabalho resumindo os resultados obtidos e apresentando as principais contribuições a dissertação. Além do mais, aponta uma série de possíveis direções e trabalhos futuros que seguem a mesma linha de pesquisa e poderão contribuir com o enriquecimento deste trabalho.

2 Introdução à Cinemática de Manipuladores

2.1 Introdução

Este capítulo trata das noções básicas e dos elementos matemáticos essenciais ao estudo da cinemática de manipuladores. Nele é apresentada a convenção de Denavit–Hartenberg, uma metodologia amplamente adotada pela literatura que permite a obtenção de parâmetros numéricos que descrevem a estrutura geométrica dos manipuladores. A aplicação sistemática dessa notação juntamente com as transformações de sistemas de referência possibilita a escrita de equações que relacionem variáveis de junta à posição e orientação do efetuador. Por fim, também são apresentados os modelos cinemáticos dos robôs usados na geração dos conjuntos de dados utilizados nesta dissertação. O material exposto neste capítulo é baseado nos livros de Craig (2005), Spong, Hutchinson e Vidyasagar (2006) e Corke (2011).

2.2 Descrições Espaciais e Transformações

O estudo da robótica reúne em torno de si uma série de conhecimentos acessórios que formam as bases matemáticas e físicas necessárias ao seu completo entendimento. São exemplos destes conhecimentos a álgebra linear e o movimento dos corpos rígidos. Tais áreas de estudo se fazem necessárias devido ao fato de os manipuladores industriais serem composições ou cadeias de corpos rígidos que precisam se localizar e orientar no espaço, tanto em relação a um sistema universal de coordenadas, bem como em relação aos componentes da cadeia. Além do mais, muitas vezes os operadores ou programadores destes dispositivos definem a tarefa a ser executada em um sistema de coordenadas de trabalho que não é o mesmo sistema sobre o qual a cinemática do manipulador foi definida. Para isso, definem-se ferramentas matemáticas que possibilitem realizar conversões entre sistemas de coordenadas diferentes, bem como localizar de maneira mais simples e amigável o espaço de trabalho a ser utilizado.

Nas seções seguintes serão definidos os elementos geométricos básicos que serão utilizados no estudo da robótica, tais como pontos, sistemas de referência, elos, dentre outros. Além do mais, são apresentadas as formas pelas quais estes elementos são descritos juntamente com a notação associada à essas descrições. Outro assunto abordado nesta seção são as transformações que os elementos geométricos podem sofrer quando representados em um espaço cartesiano: translações e rotações. Por fim, é exposto o conceito de transformação homogênea que unifica as operações de translação e rotação em uma única operação matricial. Tal ferramenta será de grande valia nas seções posteriores que tratam de

cinemática de manipuladores.

2.2.1 Descrição da Posição

Dado um sistema de coordenadas, podemos localizar um ponto no espaço através de um vetor P que tem sua base posicionada sobre a origem do sistemas de coordenadas e sua extremidade orientada posicionada sobre o ponto que se deseja representar. Este ponto poderá localizar diversos elementos tais como a ponta do efetuador, uma extremidade de um corpo rígido, a extremidade de um objeto localizado no espaço de trabalho, dentre outros. A quantidade de componentes utilizadas para definir estes vetores dependerá da região em que o manipulador está definido, ou seja, se sua operação se dá num plano de trabalho ou em um espaço de trabalho. A Equação (2.1) apresenta a descrição matemática de um ponto no espaço cartesiano tridimensional:

$$^{A}P = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} \tag{2.1}$$

Para uma descrição mais precisa da posição faz-se necessário explicitar sobre qual sistema de coordenadas um determinado ponto está definido. Para isso será utilizada a notação ^{A}P, explicitando que as coordenadas do ponto P estão sendo tomadas sobre os eixos do sistema de referência $\{A\}$. A Figura 2 apresenta um sistema de coordenadas $\{A\}$, seus vetores unitários e o ponto ^{A}P:

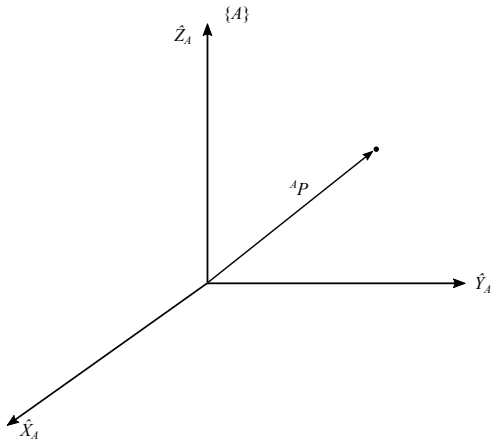


Figura 2 – Ponto relativo a um sistema de referência $\{A\}$.

Um mesmo ponto no espaço pode ser descrito com respeito a mais de um sistema de coordenadas. Por exemplo, pode-se supor a existência de um segundo sistema de referência $\{B\}$ que pode estar transladado e/ou rotacionado em relação ao sistema de referência $\{A\}$. Neste caso, o mesmo ponto P terá duas representações, uma tomada sobre eixos de $\{A\}$ e outra tomada sobre os eixos de $\{B\}$.

Uma das operações mais comuns que podem ser realizadas sobre as descrições dos pontos é a mudança do sistema de referência no qual ele está representado. Esta transformação da representação é definida através da localização e da orientação relativas entre os dois sistemas de coordenadas em questão. As duas seções seguintes detalham como descrever a orientação dos elementos robóticos bem como a descrição dos sistemas de referência. Elas reunem as ferramentas matemáticas necessárias à conversão da representação de um ponto nos diversos sistemas coordenados.

2.2.2 Descrição da Orientação

Para fornecer a localização completa de um corpo no espaço são necessárias duas informações: a posição do corpo e sua orientação. A posição, como descrito anteriormente, será dada por um vetor posição. Para que as orientações dos elos sejam representadas, são anexados a eles sistemas de referência locais. Estes sistemas de referência locais podem sofrer transformações em relação ao sistema de referência universal. Uma dessas transformações é a rotação. A Figura 3 exibe um efetuador cuja posição é dada por um ponto entre suas garras. Neste efetuador foi anexado um sistema de eixos coordenados local que fornece a orientação relativa ao um sistema de referência universal:

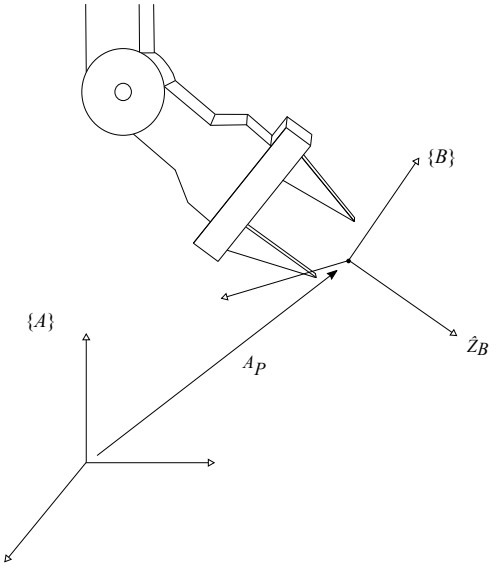


Figura 3 – Sistema de referência definindo a orientação do manipulador.

Dada a necessidade de um componente de orientação, faz-se necessário definir uma descrição matemática genérica, que não gere nem ambiguidades nem descontinuidades. Este trabalho adotará como descrição a projeção dos vetores unitários do sistema de referência de origem sobre os eixos coordenados do sistema de referência de destino.

Desta forma, supõem-se dois sistemas de referência $\{A\}$ e $\{B\}$ que se diferem apenas por suas orientações. Os vetores unitários que compõe o sistema $\{B\}$ são denotados por $\hat{X}_B$, $\hat{Y}_B$ e $\hat{Z}_B$. Quando estes vetores são representados no sistemas de coordenadas $\{A\}$

ele podem ser escritos como ${}^{A}\hat{X}_B$, ${}^{A}\hat{Y}_B$ e ${}^{A}\hat{Z}_B$, respectivamente. As duas representações podem ser relacionadas através de operações de produto escalar como é ilustrado pela equação abaixo:

$$ {}^{A}_{B}R = \begin{bmatrix} {}^{A}\hat{X}_B & {}^{A}\hat{Y}_B & {}^{A}\hat{Z}_B \end{bmatrix} = \begin{bmatrix} \hat{X}_B \cdot \hat{X}_A & \hat{Y}_B \cdot \hat{X}_A & \hat{Z}_B \cdot \hat{X}_A \\ \hat{X}_B \cdot \hat{Y}_A & \hat{Y}_B \cdot \hat{Y}_A & \hat{Z}_B \cdot \hat{Y}_A \\ \hat{X}_B \cdot \hat{Z}_A & \hat{Y}_B \cdot \hat{Z}_A & \hat{Z}_B \cdot \hat{Z}_A \end{bmatrix} \tag{2.2} $$

2.2.3 Descrição de Sistemas de Referência

Frequentemente para descrever determinados elementos em robótica deve-se levar em conta a localização destes elementos juntamente com sua orientação. Anexar um sistema de referência sobre um determinado ponto de um corpo é uma maneira consistente de agrupar essas duas informações. Por exemplo, este tipo de abordagem pode ser utilizada para definir um ponto entre as garras de um efetuador do tipo pinça, informando a localização da pinça juntamente com a sua orientação relativa a um sistema universal de coordenadas.

Um sistema de coordenadas é completamente especificado ao se fornecer quatro vetores: uma para a localização da origem do sistemas de coordenadas e outros três para sua orientação. Outra forma de representar a orientação se dá através do uso de uma matriz de rotação, que define a orientação do sistema que se quer definir em relação a um sistema universal de coordenadas. Isto posto, adota-se um sistema de coordenadas $\{A\}$ que servirá de sistema de referência universal para um sistema $\{B\}$ que se quer definir. O vetor que localiza a origem é denotado por ${}^{A}P_{BORG}$ e a matriz de rotação por ${}^{A}_{B}R$. De maneira mais compacta tem-se $\{B\} = \{{}^{A}_{B}R, {}^{A}P_{BORG}\}$. A Figura 4, ilustra como um sistema de referência $\{B\}$ pode ser representado com respeito a um referencial $\{A\}$:

2.2.4 Transformações Entre Sistemas de Referência

Como explicitado anteriormente, em diversas situações no estudo da robótica, um ponto representado em um sistema de referência $\{B\}$ deve ter sua descrição convertida para outra em um sistemas de referência $\{A\}$ não coicidente com ele. Estes sistemas de referência podem estar tanto transladados bem como rotacionados um em relação ao outro. Sendo assim, deseja-se converter a representação do ponto ${}^{B}P$ em uma descrição ${}^{A}P$ no sistema $\{A\}$. A Figura 4 ilustrou este caso geral para dois sistemas $\{A\}$ e $\{B\}$.

Como exposto por Craig (2005), para converter a descrição do ponto ${}^{B}P$ deve-se primeiramente levá-lo a um sistema de referência intermediário que possua a mesma orientação de $\{A\}$. Este primeiro passo é realizado através da multiplicação da matriz de rotação pelo ponto ${}^{B}P$. Após a operação de rotação, aplica-se a soma vetorial com vetor

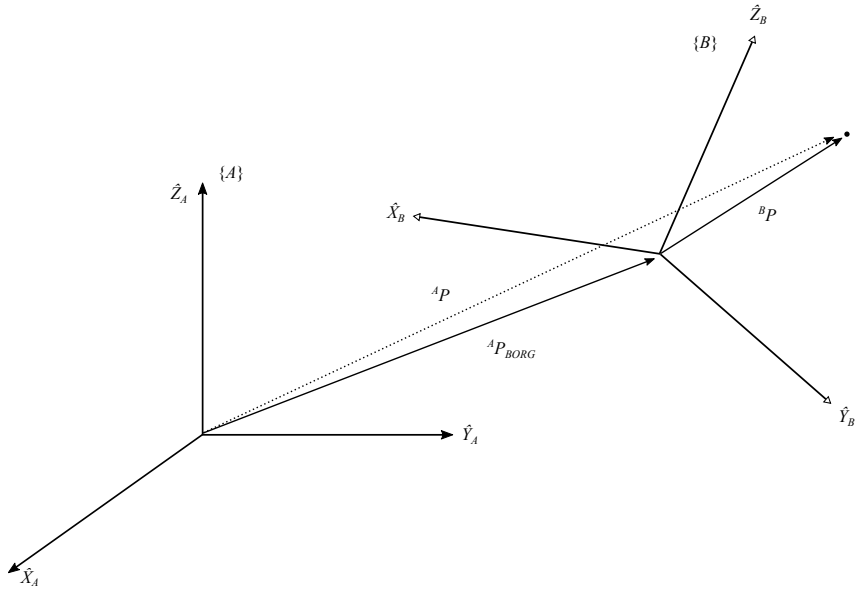


Figura 4 – Representação de um sistema de referência $\{B\}$ em relação a um sistema $\{A\}$. Um ponto P qualquer também é ilustrado, demonstrando a possibilidade de se ter mais de uma representação.

que localiza a origem do sistema $\{B\}$ em relação ao sistema $\{A\}$, ${}^{A}P_{BORG}$. Neste ponto, vale a pena ressaltar novamente a importância da notação adotada, visto que esta facilita o cancelamento dos sobrescritos e subscritos. A Equação (2.3) descreve o mapeamento entre as duas representações:

$$ {}^{A}P = {}^{A}_{B}R\, {}^{B}P + {}^{A}P_{BORG} \tag{2.3} $$

Nos casos mais específicos, em que apenas translações ou rotações estão presentes, algumas considerações podem ser feitas. No primeiro caso, a matriz de rotação envolvida é a matriz identidade, ou seja, ${}^{A}_{B}R = I$. No segundo caso, o vetor posição envolvido é o vetor nulo, ou seja, ${}^{A}P_{BORG} = \mathbf{0}$.

A Equação (2.3) pode ser representada de forma mais simples e clara ao se definir um operador T que agrupe as operações de rotação e translação em uma única matriz. Este operador matricial é chamado de *transformação homogênea* e pode expresso da seguinte forma:

$$ {}^{A}P = {}^{A}_{B}T\, {}^{B}P \tag{2.4} $$

Para aplicar a matriz da transformação homogênea T sobre um vetor representado no sistema de referência $\{B\}$, expresso por ${}^{B}P$, e representá-lo no sistema de referência $\{A\}$, obtendo-se ${}^{A}P$, realiza-se os seguintes passos:

1. Adicionar uma quarta componente ao vetores ${}^{B}P$ e ${}^{A}P$ com valor 1;

2. Concatenar lateralmente o vetor ${}^{A}P_{BORG}$ à direta da matriz ${}^{A}_{B}R$ gerando uma matriz intermediária de dimensão 3×4;

3. Adicionar uma linha "[0 0 0 1]" à matriz intermediária obtida no passo anterior. Neste ponto obtém-se a matriz de transformação homogênea.

Após a aplicação destas regras chega-se à Equação (2.5). Pode-se notar que ao executar a multiplicação dos blocos de T pelo vetor ${}^{B}P$ expandido, esta equação realiza conjuntamente as operações de rotação e translação como nas Equações (2.3) e (2.4).

$$\begin{bmatrix} {}^{A}P \\ 1 \end{bmatrix} = \left[\begin{array}{ccc|c} & {}^{A}_{B}R & & {}^{A}P_{BORG} \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \begin{bmatrix} {}^{B}P \\ 1 \end{bmatrix} \tag{2.5}$$

Além de realizar conversões entre sistemas de referência diferentes, as transformações homogêneas podem ser utilizadas e interpretadas de outras maneiras. Elas também podem ser interpretadas como a descrição de um sistema de referência em relação à outro. Por exemplo, podemos enxergar a matriz ${}^{A}_{B}T$ como a descrição do sistema de referência $\{B\}$ em relação ao sistema $\{A\}$, onde as colunas de ${}^{A}_{B}R$ definem os vetores unitários de $\{B\}$ projetados sobre $\{A\}$ e ${}^{A}P_{BORG}$ localiza a origem de $\{B\}$ em relação à $\{A\}$.

Uma terceira maneira de se utilizar as transformações homogêneas é aplicá-las como operadores de transformação. Neste tipo de aplicação, pontos são rotacionados e transladados em relação a um único sistema de referência. Por exemplo, um ponto ${}^{A}P_1$, representado em $\{A\}$, pode ser rotacionado e/ou transladado para um ponto ${}^{A}P_2$, também representado em $\{A\}$. Craig (2005) e Spong, Hutchinson e Vidyasagar (2006) trazem mais detalhes sobre este tipo de interpretação sobre as transformações homogêneas.

Quando mais de dois sistemas de referência estão envolvidos, as transformações homogêneas podem ser compostas para desenvolver conversões de pontos mais complexas. Por exemplo, a Figura 5 ilustra um ponto ${}^{C}P$ descrito em $\{C\}$.

O sistema $\{C\}$ é descrito em relação à $\{B\}$ através de ${}^{B}_{C}T$ e o sistema $\{B\}$ é descrito em relação à $\{A\}$ através de ${}^{A}_{B}T$. Para se representar ${}^{C}P$ no sistema de referência $\{A\}$ aplica-se a Equação (2.6):

$${}^{A}P = {}^{A}_{B}T\, {}^{B}_{C}T\, {}^{C}P = {}^{A}_{C}T\, {}^{C}P \tag{2.6}$$

$${}^{A}_{C}T = {}^{A}_{B}T\, {}^{B}_{C}T \tag{2.7}$$

Mais uma vez, a notação utilizada por Craig (2005) facilita o cancelamento dos sobrescritos e subscritos, tornando a manipulação das equações mais simples.

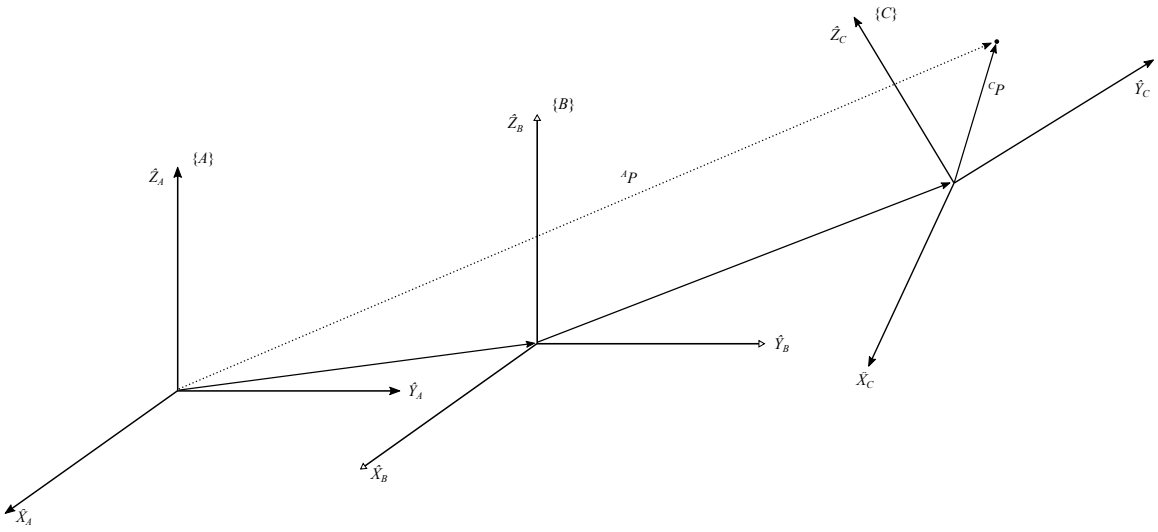


Figura 5 – Ponto P originalmente descrito em um sistema $\{C\}$ é representado em um sistema $\{A\}$ através de um sistema intermediário $\{B\}$.

2.3 Cinemática dos Manipuladores

2.3.1 Introdução

A Cinemática é a ciência que estuda o movimento sem se preocupar com as suas causas, ou seja, as forças ou torques que levam um determinado corpo à adquirir aceleração. Basicamente ela consiste no estudo da posição dos corpos e de suas derivadas de ordem mais elevada, podendo ser estas em relação ao tempo ou qualquer outra grandeza (CRAIG, 2005). Neste trabalho, um estudo sobre a cinemática dos manipuladores é realizado em condições estáticas, ou seja, na ausência de velocidades ou acelerações. Atrávés deste estudo, serão obtidas relações matemáticas entre os diversos sistemas de referência anexados aos elos dos manipuladores, possibilitando a descrição da localização e da orientação do efetuador em função da posição relativa dos elos da cadeia cinemática.

Desta forma, será apresentado um procedimento sistemático através do qual será possível descrever, de forma precisa e sucinta, o arranjo geométrico dos manipuladores industriais. Este procedimento tem como arcabouço os conceitos apresentados nas seções anteriores, notadamente as que tratam de sistemas de referência e suas transformações. Através destes conceitos, serão descritos individualmente os elos, que compõem as cadeias robóticas, bem como suas relações, que são definidas através das juntas que os unem.

Além do mais, será extraído, das definições de elos e juntas, um conjunto de parâmetros geométricos denominados parâmetros de *Denavit-Hartenberg*. Através destes parâmetros, serão desenvolvidas as equações cinemáticas que relacionam a localização e a orientação do efetuador com as variáveis de junta do manipulador. Neste trabalho, foram adotadas apenas cadeias de corpos rígidos ligados através de juntas rotacionais, o que limita as variáveis de junta apenas ao tipo rotacional.

2.3.2 Descrição de um Elo

De acordo com Craig (2005), um elo, ou *link*, "é um corpo rígido que define a relação entre os eixos de duas juntas, vizinhas, de um manipulador". A conexão em cadeia destes elos através de juntas é o que define um manipulador robótico. Cada junta possui um eixo de junta sobre o qual o elo pode rotacionar. A Figura 6 ilustra um elo com duas juntas rotacionais, $\{i-1\}$ e $\{i\}$, e seus respectivos eixos de junta, juntamente com a linha mutuamente perpendicular aos eixos das juntas:

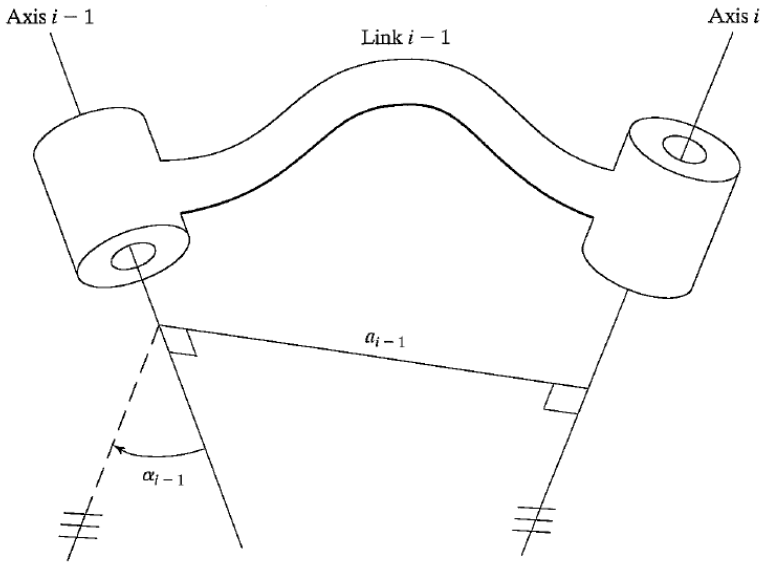


Figura 6 – Elo genérico ilustrando o tamanho e a torção de elo, denotados respectivamente por a e α, que relaciona o movimento de duas juntas rotacionais. Adaptado de Craig (2005)

Para especificar completamente um elo, duas quantidades devem ser especificadas: o comprimento de elo e a torção de elo. O *comprimento de elo*, denotado por a_{i-1} para a junta $\{i-1\}$, é definido pelo comprimento medido sobre a linha que é mutuamente perpendicular aos eixos de junta $\{i-1\}$ e $\{i\}$. Já a *torção de elo*, denotado por α_{i-1}, é definida pelo ângulo formado entre os eixos $\{i\}$ e $\{i-1\}$ sobre o plano cuja a normal é dada pela linha em que se toma o valor de a_{i-1}. A Figura 6 também exibe estes dois valores.

2.3.3 Descrição da Conexão dos Elos

As juntas são os elementos que conectam dois elos, possibilitando o movimento relativo entre eles. A Figura 7 ilustra dois elos intermediários da cadeia conectados através de um junta rotacional, juntamente com os eixos de junta $\{i-1\}$ e $\{i\}$. Esta ilustração também exibe as linhas sobre as quais são medidos os comprimentos de elo a_{i-1} e a_i.

Para descrever esta conexão duas quantidades devem ser especificadas: o deslocamento de elo e o ângulo da junta. O *deslocamento de elo*, denotado por d_i, é a distância,

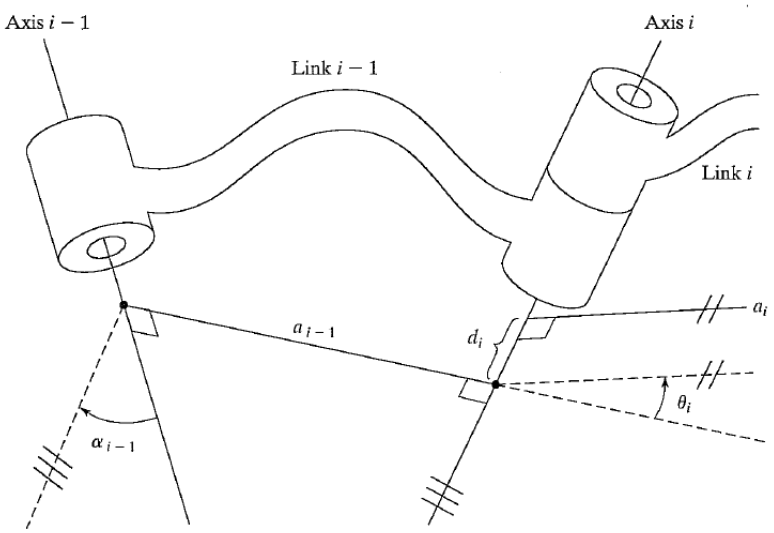


Figura 7 – Deslocamento de *link*, d, e ângulo de junta, θ, descrevendo a conexão entre dois elos.

ao longo do eixo de junta $\{i\}$, da reta que define a_{i-1} até a reta que define a_i. O *ângulo de junta*, denotado por θ_i, é a medida do ângulo de rotação, ao longo do eixo $\{i\}$, da reta que define a_i em relação a reta que define a_{i-1}.

Para as juntas da extremidade da cadeia será convencionado atribuir zero as comprimentos de elo, ou seja, $a_0 = a_n = 0$. Além do mais, para os casos deste trabalho, os valores do deslocamento de elo da primeira e da última junta também serão considerados nulos, isto é, $d_0 = d_n = 0$. Esta convenção permite cálculos mais simples devido a uma maior quantidade de valores nulos.

2.3.4 Fixando Sistemas de Referência aos Elos

De forma a localizar os elos em relação aos seu vizinhos, fixam-se sistemas de referência sobre cada um deles. De posse da descrição matemática destes sistemas de referência e do uso de transformações compostas, constrói-se uma representação matricial que relacione o posicionamento do efetuador com o sistema de referência da base. Como exemplificado nas seções anteriores, esta notação possibilita a rápida manipulação dos índices que indicam um elo e seu antecessor.

Com o objetivo de unir a capacidade de manipulação matemática e a obtenção dos parâmetros de Denavit-Hartemberg, faz-se necessária a adoção de convenções que padronizem o processo de construção das matrizes de transformação homogêneas. A numeração dos sistemas de referências seguirá a mesma adotada para os elos correspondentes.

- **Passo 1**: Identificar os eixos de rotação $\{i\}$ e $\{i+1\}$ das juntas e representá-los como linhas infinitas.

- **Passo 2**: Para cada par de juntas, identificar a linha perpendicular comum a ambas as linhas infinitas.
- **Passo 3**: Definir o eixo $\hat{Z}_i$ do sistema de referência $\{i\}$ apontando ao longo do eixo de sua junta.
- **Passo 4**: Definir o eixo $\hat{X}_i$ do sistema de referência $\{i\}$ apontando na direção da linha perpendicular comum aos eixos $\{i\}$ e $\{i+1\}$
- **Passo 5**: Definir o eixo $\hat{Y}_i$ através da regra da mão direita entre $\hat{X}_i$ e $\hat{Z}_i$.
- **Passo 6**: Definir o sistema de referência $\{0\}$ de forma que este se igualhe a $\{1\}$ quando a primeira variável de junta for zero. Para $\{N\}$, $\hat{X}_N$ é de livre escolha porém, deve-se escolher uma posição que anule uma maior quantidade de parâmetros de acoplamento.

A Figura 8 ilustra o posicionamentos dos *frames* $\{i-1\}$ e $\{i\}$ sobre dois elos de um manipulador genérico seguindo os passos definidos.

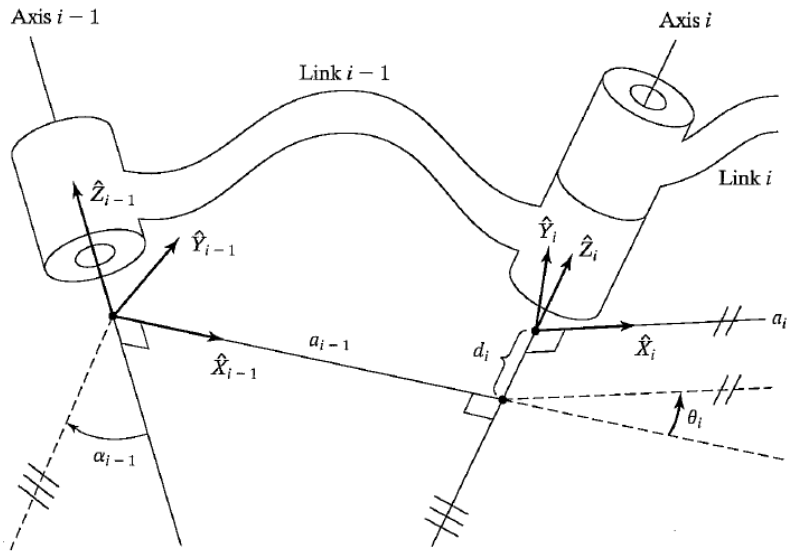


Figura 8 – Sistemas de referência anexados aos elos de acordo com a notação adotada.

Com os sistemas de referência devidamente numerados e dispostos na cadeia de elos do manipulador, define-se formalmente os parâmetros de Denavit-Hartemberg da seguinte maneira:

- **Comprimento de elo** (a_i): distância de $\hat{Z}_i$ a $\hat{Z}_{i+1}$ medida ao longo de $\hat{X}_i$
- **Torção de elo** (α_i): ângulo de $\hat{Z}_i$ a $\hat{Z}_{i+1}$ medido ao longo de $\hat{X}_i$
- **Deslocamento de elo** (d_i): distância de $\hat{X}_{i-1}$ a $\hat{X}_i$ medida ao longo de $\hat{Z}_i$
- **Ângulo de junta** (α_i): ângulo de $\hat{X}_{i-1}$ a $\hat{X}_i$ medido ao longo de $\hat{Z}_i$

Após a obtenção dos parâmetros de Denavit-Hartemberg, derivam-se transformações homogêneas de cada um dos seus elos em relação aos seus vizinhos. Isto é feito através de sistemas de referência auxiliares que possibilitam escrever as transformações de elo como uma multiplicação sucessiva de transformações que são funções de apenas de um dos parâmetros de elo (CRAIG, 2005). Com isso, chega-se a seguinte expressão para a transformação de um elo $\{i\}$ em relação ao elo $\{i-1\}$:

$$ {}^{i-1}_{i}T = \begin{bmatrix} c\theta_i & -s\theta_i & 0 & a_{i-1} \\ s\theta_i c\alpha_{i-1} & c\theta_i c\alpha_{i-1} & -s\alpha_{i-1} & -s\alpha_{i-1}d_i \\ s\theta_i s\alpha_{i-1} & c\theta_i s\alpha_{i-1} & c\alpha_{i-1} & c\alpha_{i-1}d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.8} $$

Após isto, compõem-se estas transformações individuais para que se obtenha a posição do efetuador em relação à base.

$$ {}^{0}_{N}T = {}^{0}_{1}T\,{}^{1}_{2}T\,{}^{2}_{3}T \ldots {}^{N-1}_{N}T \tag{2.9} $$

em que ${}^{0}_{N}T$ é função das variáveis de juntas. As seções seguintes apresentaram os parâmetros de DH e descreverá as equações de cinemática direta manipuladores utilizados neste trabalho, a saber, o robô planar, o manipulador PUMA 560 e o manipulador Motoman HP6.

2.3.5 Cinemática Direta dos Manipuladores

2.3.5.1 Introdução

A *cinemática direta* se ocupa em descobrir a posição e a orientação finais do efetuador a partir de um conjunto de ângulos de junta. De forma alternativa, dados um vetor $\boldsymbol{\theta} = [\theta_1, \ldots, \theta_n]^T$ das variáveis de junta, a cinemática direta estuda mecanismos que possibilitem encontrar um função $\mathbf{T} = f(\boldsymbol{\theta})$ que localize o efetuador através de uma matriz de transformação homogênea. Esta matriz $\mathbf{T}$ é formada por uma submatriz de rotação $\mathbf{R}$, que orienta o efetuador, e um vetor $\mathbf{r}$ que dá a sua posição. Para as simulações executadas neste trabalho, a orientação foi previamente fixada e somente a posição foi levada em conta.

2.3.5.2 Cinemática Direta do Robô Planar

O robô planar é uma estrutura robótica do tipo $2R$, ou seja, é composto de duas juntas rotacionais cujos os eixos são paralelos e perpendiculares ao plano de atuação do robô. Cada elo deste manipulador forma com o plano de trabalho e com o elo anterior ângulos de junta θ_1 e θ_2, respectivamente. A Figura 9 ilustra os elementos que definem a geometria deste manipulador

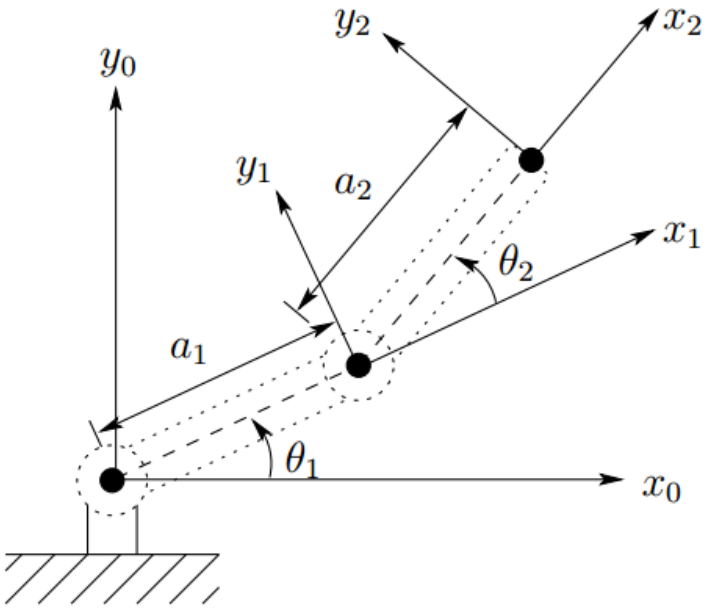


Figura 9 – Manipulador planar de dois *links*.

Devido a simplicidade da cadeia cinemática deste robô, uma abordagem geométrica pode ser utilizada para definir as equações que governam a sua cinemática direta. Os valores de x e y podem ser obtidos através das somas das projeções dos elos sobre os eixos x_0 e y_0, respectivamente. Através da análise da Figura 9, conclui-se que

$$x = L_1 cos(\theta_1) + L_2 cos(\theta_1 + \theta_2) \tag{2.10}$$

$$y = L_1 sen(\theta_1) + L_2 sen(\theta_1 + \theta_2) \tag{2.11}$$

em que θ_1 e θ_2 são os ângulos das juntas e x e y são as coordenadas cartesianas que localizam o efetuador.

2.3.5.3 Cinemática Direta do Robô PUMA 560

O robô PUMA 560 é um mecanismos robótico com seis graus de liberdade e todas as juntas rotacionais, ou seja, um mecanismo do tipo $6R$. A Figura 10 ilustra a cadeia robótica com todos os ângulos de junta iguais a zero e com os alguns sistemas de referência fixados aos elos.

A Tabela 1 reúne de forma esquematizada os parâmetros de DH extraídos da Figura 10

Em seguida, os parâmetros de DH são aplicados seis vezes na Equação (2.8) de forma a se obter seis matrizes que relacionam um elo específico ao seu correspondente anterior. Através da composição destas transformações, obtém-se a transformação homogênea que

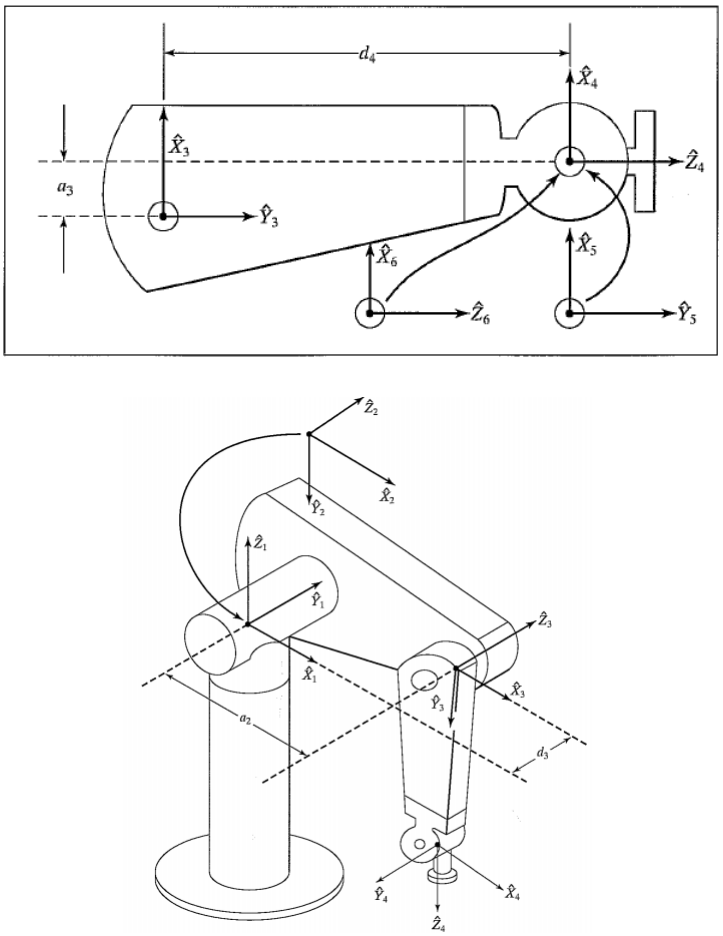


Figura 10 – Manipulador PUMA 560 disposto com seus ângulos de juntas zerados.

i	$\alpha_i - 1$	$a_i - 1$	d_i	θ_i
1	0	0	0	θ_1
2	$-90°$	0	0	θ_2
3	0	a_2	d_3	θ_3
4	$-90°$	a_3	d_4	θ_4
5	$90°$	0	0	θ_5
6	$-90°$	0	0	θ_6

Tabela 1 – Parâmetros de Denavit-Hartenberg para o manipulador PUMA 560.

expressa a localização do manipulador em relação ao sistema de referência fixado à base. Através da análise dos componentes desta transformação, temos:

$$r_{11} = c_1[c_{23}(c_4c_5c_6 - s_4s_5) - s_{23}s_5c_5] + s_1(s_4c_5c_6 + c_4s_6) \tag{2.12}$$
$$r_{21} = s_1[c_{23}(c_4c_5c_6 - s_4s_6) - s_{23}s_5c_6] - c_1(s_4c_5c_6 + c_4s_6) \tag{2.13}$$
$$r_{31} = -s_{23}(c_4c_5c_6 - s_4s_6) - c_{23}s_5c_6 \tag{2.14}$$
$$r_{12} = c_1[c_{23}(-c_4c_5s_6 - s_4c_6) + s_{23}s_5s_6] + s_1(c_4c_6 + s_4c_5s_6) \tag{2.15}$$
$$r_{22} = s_1[c_{23}(-c_4c_5s_6 - s_4c_6) + s_{23}s_5s_6] - c_1(c_4c_6 + s_4c_5s_6) \tag{2.16}$$
$$r_{32} = -s_{23}(-c_4c_5s_6 - s_4c_6) + c_{23}s_5s_6 \tag{2.17}$$
$$r_{13} = -c_1(c_{23}c_4s_5 + s_{23}c_5) - s_1s_4s_5 \tag{2.18}$$
$$r_{23} = -s_1(c_{23}c_4s_5 + s_{23}c_5) + c_1s_4s_5 \tag{2.19}$$
$$r_{33} = s_{23}c_4s_5 - c_{23}c_5 \tag{2.20}$$
$$p_x = c_1[a_2c_2 + a_3c_{23} - d_4s_{23}] - d_3s_1 \tag{2.21}$$
$$p_y = s_1[a_2c_2 + a_3c_{23} - d_4s_{23}] + d_3c_1 \tag{2.22}$$
$$p_z = -a_3s_{23} - a_2s_2 - d_4c_{23} \tag{2.23}$$

onde $\theta_1, \theta_2, \ldots, \theta_6$ são os ângulos de junta. Além do mais, $s_i = \text{sen}\ \theta_i$, $c_i = \cos\theta_i$, $s_{ij} = \text{sen}\ (\theta_i + \theta_j)$ e $c_{ij} = \cos(\theta_i + \theta_j)$.

2.3.5.4 Cinemática Direta do Robô Motoman HP6

Assim como o manipulador PUMA 560, o robô Motoman HP6 (Figura 11) é uma cadeia robótica do tipo $6R$ indicando que todos os seus graus de liberdade são compostos por juntas rotacionais. Devido a este fato, tem-se seis variáveis de junta denotadas por θ_i, onde $i = 1, \ldots, 6$. A Tabela 2 reúne de forma esquematizada os parâmetros DH extraída do modelo de simulação fornecido pelo Robotic Toolbox para Matlab (CORKE, 1996):

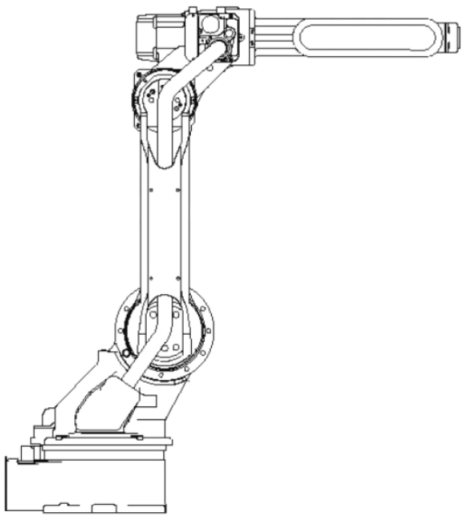


Figura 11 – Manipulador Motoman HP6.

i	$\alpha_i - 1$	$a_i - 1$	d_i	θ_i
1	$-1{,}571$	0,15	0	θ_1
2	$-3{,}142$	0,57	0	θ_2
3	$-1{,}571$	0,155	0	θ_3
4	1,571	0	$-0{,}635$	θ_4
5	$-1{,}571$	0	0	θ_5
6	3,142	0	$-0{,}095$	θ_6

Tabela 2 – Parâmetros de Denavit-Hartenberg para o manipulador Motoman HP6.

Através dos parâmetros DH, aplica-se a Equação (2.8) juntamente com o procedimento de composição expresso em (2.9) para se obter a matriz ${}^6_0T$ que determina a postura final do manipulador:

$$ {}^0_6T = {}^0_1T^1_2T^2_3T^3_4T^4_5T^5_6T = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.24} $$

$$ n_x = c_1\left[c_{23}(c_4c_5c_6 - s_4s_6) - s_{23}s_5c_6\right] + s_1(s_4c_5c_6 + c_4s_6) \tag{2.25} $$

$$ n_y = s_1\left[c_{23}(c_4c_5c_6 - s_4s_6) - s_{23}s_5c_6\right] - c_1(s_4c_5c_6 + c_4s_6) \tag{2.26} $$

$$ n_z = -s_{23}(c_4c_5c_6 - s_4s_6) - c_{23}s_5c_6 \tag{2.27} $$

$$ o_x = c_1\left[c_{23}(-c_4c_5s_6 - s_4c_6) + s_{23}s_5s_6\right] + s_1(c_4c_6 - s_4c_5s_6) \tag{2.28} $$

$$ o_y = s_1\left[c_{23}(-c_4c_5s_6 - s_4c_6) + s_{23}s_5s_6\right] - c_1(c_4c_6 - s_4c_5c_6) \tag{2.29} $$

$$ o_z = -s_{23}(-c_4c_5s_6 - s_4c_6) - c_{23}s_5s_6 \tag{2.30} $$

$$ p_x = c_1(a_2c_2 + a_3c_{23} - d_4s_{23}) - d_2s_1 \tag{2.31} $$

$$ p_y = s_1(a_2c_2 + a_3c_{23} - d_4s_{23}) - d_2c_1 \tag{2.32} $$

$$ p_z = -a_3s_{23} - a_2s_2 - d_4c_{23} \tag{2.33} $$

$$ a_x = -c_1(c_{23}c_4s_5 + s_{23}c_5) - s_1s_4s_5 \tag{2.34} $$

$$ a_y = -s_1(c_{23}c_4s_5 + s_{23}c_5) + c_1s_4s_5 \tag{2.35} $$

$$ a_z = s_{23}c_4s_5 - c_{23}c_5 \tag{2.36} $$

2.3.6 Cinemática Inversa dos Manipuladores

Ao lidar com o problema da cinemática inversa deve-se levar em consideração alguns aspectos de sua solução tais como sua existência, sua multiplicidade e seus método de obtenção.

Primeiramente, devemos considerar a existência ou não de soluções para posição e orientação dadas. Esta análise é bem particular de cada manipulador pois envolve um

estudo detalhado de seu *workspace*. Este *workspace* depende de diversos fatores tais como a quantidade de graus de liberdade do manipulador, os valores dos parâmetros DH e os limites impostos aos ângulos de junta.

Considerando a existência de uma solução específica, deve-se verificar sua multiplicidade. Alguns manipuladores podem atingir a mesma posição e orientação através de conjuntos diferentes de ângulos de junta. Por exemplo, o PUMA 560 pode alcançar certos alvos utilizando oito soluções diferentes. Em aplicações práticas, normalmente adotam-se critérios de seleção das soluções tais como a proximidade em relação a posição atual do efetuador ou a ausência de obstáculos durante o trajeto até uma nova posição. Para manipuladores mais complexos (com muitos parâmetros DH diferentes de zero) o número de soluções pode ser bastante alto.

Uma outra questão a ser levada em conta é o método utilizado para se obter as soluções. Craig (2005) adverte o fato de não existirem métodos gerais para se resolver um conjunto de equações não lineares. Além disso, Craig (2005) considera em sua obra, através da definição dada em Roth (1976), que um *manipulador solucionável* é aquele em que é possível encontrar todos os conjuntos de variáveis de junta associadas a uma posição e orientação dadas. Logo são citados neste trabalho apenas os métodos que são capazes de obter todas as soluções.

A literatura normalmente divide as soluções em duas classes: *soluções de forma fechada* e *soluções numéricas*. Esta última são bem mais custosas do ponto de vista computacional devido a sua natureza iterativa. Além do mais, são bem mais lentas do que as versões correspondentes de forma fechada. Para mais informações sobre soluções numéricas, consultar Tsai e Morgan (1985),Nakamura e Hanafusa (1986) e Baker e Wampler (1988).

As soluções de forma fechada são aquelas que obtém como solução equações analíticas para os ângulos das juntas em função da posição e da orientação. Normalmente, dentro desta categoria podem ser identificados dois tipos de métodos bem similares: o algébrico e o geométrico. O primeiro manipula algebricamente as equações obtidas na análise da cinemática direta para chegar nas soluções. Já a segunda faz intenso uso de técnicas de geometria planar para obter as soluções. Geralmente, as expressões envolvidas durante ambos os procedimento são do tipo trancendentais. Por exemplo, muitas dessas expressões são funções de $sen(\theta)$ e $cos(\theta)$ mas não como funções diretas de θ.

Um resultado conhecido acerca da obtenção das soluções da cinemática inversa é que manipuladores com três eixos de junta vizinhos que se cruzam sempre tem uma solução de forma fechada. Este resultado é chamado de *Solução de Pieper* e pode ser consultado em Peiper (1968). O manipulador PUMA 560, utilizado neste trabalho, é um exemplo clássico deste resultado. As referências Craig (2005), Lee e Ziegler (1984) e Feng, Hu e Shen (2011) trazem as soluções da cinemática inversa para os robôs utilizados neste

trabalho, a saber, o robô planar, o PUMA 560 e o Motoman HP6.

2.4 Conclusão

Este capítulo apresentou os elementos matemáticos fundamentais necessários ao estudo das cinemáticas direta e inversa dos manipuladores industriais. Também foram apresentadas as cinemáticas diretas para três robôs manipuladores comumente citados na literatura. Além disso, foram abordados alguns aspectos importantes da solução para o problema da cinemática inversa bem como os desafios relacionados à sua obtenção. O próximo capítulo apresenta os métodos de regressão não-linear aplicados a este trabalho.

3 Algoritmos de Aprendizagem de Máquina

3.1 Introdução

Neste capítulo são descritos as arquiteturas e os mecanismos de treinamento dos modelos de regressão não-linear aplicados ao problema de aprendizado da cinemática inversa de robôs manipuladores. São apresentados modelos de diversas classes sendo estas as Redes Neurais Artificiais, Modelos de Regressão Baseados em Métodos de Kernel, Sistemas Neuro-Fuzzy, Modelos de Regressão Baseados em Distância e os Modelos Lineares Locais.

3.2 Redes Neurais Artificiais

Haykin (2009) define uma rede neural como um processador maciçamente paralelamente distribuído constituído de unidades de processamento simples, que têm a propensão natural para armazenar conhecimento experimental e torná-lo disponível para uso. Este conhecimento é armazenado nas conexões interneuronais e que são representadas através de pesos sinápticos. A atualização destes pesos sinápticos, através de uma regra de aprendizagem, é o que possibilita as redes neurais a desenvolverem um aprendizado e serem aplicadas em tarefas de generalização. As duas próximas seções abordam duas redes neurais utilizadas neste trabalho: o Perceptron Multicamadas (*Multi Layer Perceptron*, MLP) e a Máquina de Aprendizado Extremo (*Extreme Learning Machine*, ELM).

3.2.1 A Rede MLP e o Algoritmo de Retropropagação dos Erros

Basicamente uma rede MLP é constituída de um conjunto de unidades de entrada, uma ou mais camadas ocultas (ou escondidas) compostas por neurônios não-lineares, e uma camada de saída composta por um ou mais neurônios que podem ser lineares ou não. Os neurônios das camadas intermediárias são chamados de neurônios ocultos ou escondidos pelo fato de não terem acesso direto à saída da rede. Embora a rede MLP possa ter qualquer número de camadas ocultas, o enfoque nesta dissertação é a rede MLP com apenas uma camada de neurônios ocultos.

A Figura 12 mostra a arquitetura de uma rede MLP *feedforward* com uma camada oculta e totalmente conectada, já adaptada ao problema de aprendizado da cinemática inversa de um manipulador qualquer. Isto significa que um neurônio em qualquer camada da rede é conectado a todas as unidades/neurônios da camada anterior, e que um sinal de entrada propaga-se na rede para frente, da esquerda para direita e de camada em camada

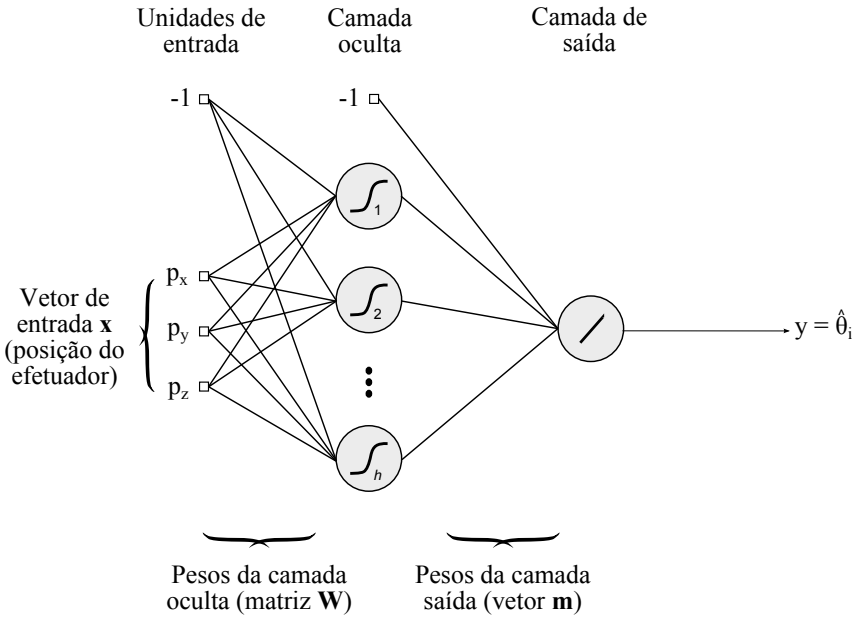


Figura 12 – Rede MLP com uma camada oculta para aprendizado da cinemática inversa. A figura exibe o vetor de entrada com três componentes para garantir uma maior generalidade do problema. A saída da rede fornece uma estimativa ou predição do valor do ângulo da junta i do manipulador.

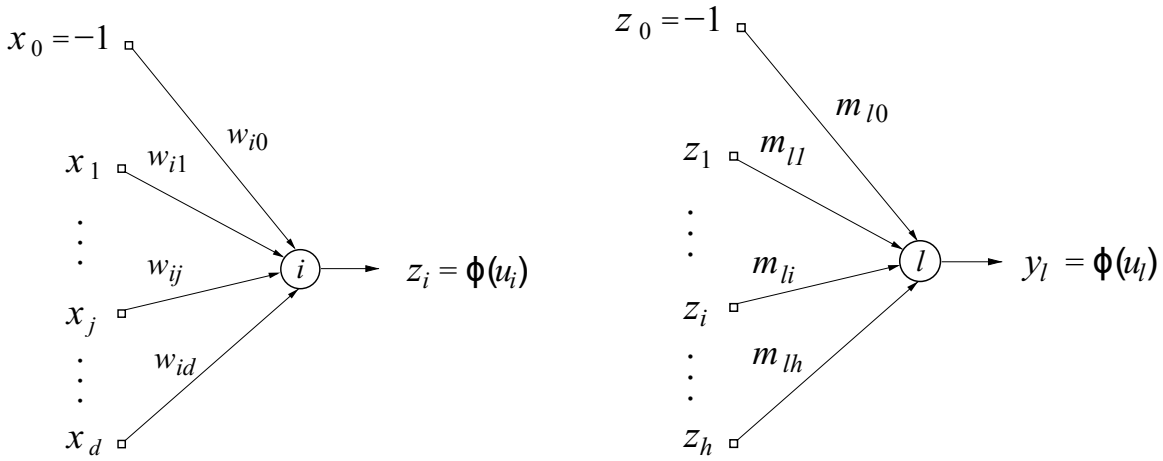


Figura 13 – Entradas e saídas dos neurônios da (a) camada oculta e (b) da camada saída.

(HAYKIN, 2009). A Figura 13 destaca os sinais intermediários produzidos pela camada oculta que servem de entrada para os neurônios da camada de saída. Nesta arquitetura $x_1, x_2, \ldots, x_d$ são as componentes do vetor de entrada $\mathbf{x}$ o qual deve ser acrescido da componente fixa $x_0 = -1$, correspondente ao limiar (*bias*). Portanto, o vetor de entrada da rede MLP, de dimensionalidade $d + 1$ (com a inclusão do limiar), pode então ser definido

como:

$$\mathbf{x} = \begin{pmatrix} -1 \\ x_1 \\ x_2 \\ \vdots \\ x_d \end{pmatrix} = \begin{pmatrix} -1 \\ p_1 \\ p_2 \\ \vdots \\ p_d \end{pmatrix} \tag{3.1}$$

em que $d = 3$ para um robô movendo-se no espaço e $d = 2$ para um robô planar.

É possível definir a matriz $\mathbf{X} \in \mathbb{R}^{(d+1)\times N_1}$ reunindo todos os vetores de entrada correspondendo às N_1 amostras do conjunto de dados utilizado na etapa de treinamento da rede. A matriz $\mathbf{X}$ pode então ser escrita como

$$\mathbf{X} = [\mathbf{x}(1)|\mathbf{x}(2)|\ldots|\mathbf{x}(N_1)]. \tag{3.2}$$

Na Figura 12, h ($2 \leq h < \infty$) representa o número de neurônios da camada oculta, os quais desempenham um papel crucial na rede MLP porque realizam uma projeção não-linear cuja meta é promover uma mudança na representação do problema de um espaço de dimensão $d + 1$ para outro de dimensão h de modo a facilitar a resolução do problema pelo neurônio de saída (HAYKIN, 2009). Conforme o processo de aprendizagem da rede avança, estes neurônios começam gradualmente a "descobrir" as características salientes presentes nos dados de treinamento. Daí então realizam uma transformação não-linear nos dados de entrada para um novo espaço, chamado espaço oculto ou espaço de características.

Na camada de saída de cada rede MLP, a quantidade de neurônios é igual a unidade ($g = 1$) pois, para a tarefa de regressão não-linear, a rede neural aplicada requer somente um único elemento, cuja função de ativação é do tipo linear, a fim de estimar o valor para cada ângulo de junta.

A matriz $\mathbf{W} \in \mathbb{R}^{h\times(d+1)}$ é formada por todas as conexões (ou pesos) sinápticas entre as unidades de entrada e o neurônio da camada oculta. Portanto, cada elemento w_{ij} de $\mathbf{W}$ na Figura 12 representa a conexão sináptica entre a j-ésima entrada e o i-ésimo neurônio da camada oculta. A matriz $\mathbf{W}$ pode então ser representada por

$$\mathbf{W} = \begin{bmatrix} \mathbf{w}_1^T \\ \mathbf{w}_2^T \\ \vdots \\ \mathbf{w}_{h_1}^T \end{bmatrix} \tag{3.3}$$

em que cada linha $\mathbf{w}_i^T = [w_{i0}\ w_{i0}\ \ldots\ w_{id}]$ corresponde aos pesos sinápticos que conectam as $d+1$ unidades de entrada ao i-ésimo neurônio oculto.

Seguindo o mesmo princípio, define-se $\mathbf{m} \in \mathbb{R}^{(h+1)}$ como o vetor formado pelas conexões sinápticas que conectam os neurônios da camada oculta ao neurônio de saída:

$$\mathbf{m} = \begin{bmatrix} m_{10} \\ m_{11} \\ \vdots \\ m_{1h} \end{bmatrix} \tag{3.4}$$

em que cada componente m_{1i} de $\mathbf{m}$ representa a conexão sináptica entre o i-ésimo neurônio oculto e o neurônio de saída. A presença de apenas um neurônio de saída é devida à adoção de uma abordagem MISO (*Multiple Inputs Single Outputs*) em que cada variável de junta é estimada por uma instância de arquitetura MLP.

Uma vez definida a arquitetura da rede MLP com uma camada oculta e apresentados os principais parâmetros, faz-se necessária uma discussão sobre o algoritmo a ser utilizado no treinamento da rede. Para esta finalidade escolheu-se o algoritmo de retropropagação do erro (*Error Backpropagation*) que é o tipo mais comum entre os artigos vistos na literatura. Na seção seguir descreve-se este algoritmo para o caso mais geral, ou seja, para a arquitetura MLP(d, h, m).

3.2.1.1 Algoritmo de Retropropagação do Erro

Os neurônios da rede neural MLP são treinados através do algoritmo de retropropagação do erro (*error backpropagation*) que foi introduzido por Rumelhart, Hinton e Williams (1988). O algoritmo consiste em duas fases de propagação de sinais: sentido direto e sentido inverso.

Sentido direto: Nesta fase um vetor de entrada $\mathbf{x}$ é apresentado à rede e os sinais de entrada gerados por esse vetor percorrem as camadas ocultas da rede neural rumo à camada de saída. Após a apresentação do padrão de entrada, as ativações dos neurônios da camada oculta são calculadas como:

$$u_i = \sum_{j=0}^{d} w_{ij} x_j = \mathbf{w}_i^T \mathbf{x}, \quad i = 1, \ldots, h \tag{3.5}$$

em que h indica o número de neurônios da camada escondida. Em seguida, as saídas correspondentes são calculadas por

$$z_i = \phi(u_i) = \phi\left(\sum_{j=0}^{p} w_{ij} x_j\right) = \phi(\mathbf{w}_i^T \mathbf{x}) \tag{3.6}$$

A função de ativação $\phi(.)$ assume geralmente uma das seguintes formas:

$$\phi(u_i) = \frac{1}{1 + \exp[-u_i]}, \quad \text{(Sigmóide Logística)} \tag{3.7}$$

$$\phi(u_i) = \frac{1 - \exp[-u_i]}{1 + \exp[-u_i]}, \quad \text{(Tangente Hiperbólica)} \tag{3.8}$$

O domínio destas funções é a reta dos números reais. Contudo, a imagem da função sigmóide logística está restrita ao intervalo $[0, 1]$, enquanto a imagem da tangente hiperbólica está restrita a $[-1, 1]$. De um extremo a outro ambas são sempre crescentes.

Procedimento similar é realizado para a camada de saída, onde as entradas desta camada são as saídas dos neurônios da camada anterior, ou seja:

$$u_k = \sum_{i=0}^{h} m_{ki} z_i, \quad k = 1, \ldots, m, \tag{3.9}$$

em que m é o número de neurônios na camada de saída. Em seguida, as saídas dos neurônios da camada de saída são calculadas como

$$y_k = \phi(u_k) = \phi\left(\sum_{i=0}^{q} m_{ki} z_i\right). \tag{3.10}$$

As funções de ativação $\phi(.)$ dos neurônios de saída assumem a forma linear, pois a saída pode assumir qualquer valor entre menos infinito e mais infinito. Após a determinação das saídas y_k para os m neurônios da camada de saída, calculam-se os erros entre a saídas desejadas d_k e y_k:

$$e_k = d_k - y_k, \quad k = 1, \ldots, m. \tag{3.11}$$

Sentido inverso: Após o cálculo do erro na fase anterior, os pesos sinápticos de todos os neurônios são atualizados através da retropropagação dos gradientes locais dos erros da camada de saída. Estes são calculados da seguinte forma:

$$\delta_k = e_k \phi'(u_k) = e_k y'_k, \quad k = 1, \ldots, m, \tag{3.12}$$

em que

$$y'_k = \phi'(u_k) = \frac{dy_k}{du_k}. \tag{3.13}$$

Após o cálculo do gradiente local para a camada de saída, este é retropropagado recursivamente para a camada oculta:

$$\delta_i = \phi'(u_i) \textstyle\sum_k m_{ki} \delta_k, \quad i = 1, \ldots, h. \tag{3.14}$$

Após o término do cálculo dos gradientes locais, estes são utilizados para atualizar os pesos sinápticos de ambas as camadas. Para a camada oculta tem-se

$$\begin{aligned} w_{ij}(n+1) &= w_{ij}(n) + \Delta w_{ij}(n), \\ &= w_{ij}(n) + \alpha \delta_i(n) x_j(n). \end{aligned} \tag{3.15}$$

e para a camada a regra de atualização é dada por

$$\begin{aligned} m_{ki}(n+1) &= m_{ki}(n) + \Delta m_{ki}(n), \\ &= m_{ki}(n) + \alpha \delta_k(n) z_i(n). \end{aligned} \tag{3.16}$$

em que a constante α é chamada de taxa de aprendizagem ou passo de adaptação, e normalmente assume valores entre 0 e 1.

Durante o uso da rede após a etapa de treinamento, espera-se que a mesma seja capaz de generalizar o conhecimento adquirido para novos dados de entrada. Por generalização entende-se a habilidade da rede em utilizar o conhecimento armazenado nos seus pesos e limiares para gerar saídas coerentes para novos vetores de entrada. A generalização é considerada boa quando a rede, durante o treinamento, foi capaz de capturar (aprender) adequadamente a relação entrada-saída do mapeamento de interesse.

As aplicações da rede MLP para aprendizado da cinemática direta/inversa de robôs manipuladores podem ser encontradas nas seguintes referências: Yıldırım e Eski (2006), Hasan et al. (2006) e Duka (2014).

3.2.2 A Rede ELM: Arquitetura e Treinamento

Esta arquitetura de rede neural conhecida como rede ELM (HUANG; ZHU; SIEW, 2006) tem a mesma arquitetura de uma rede MLP de uma camada oculta, porém apresenta uma fase de aprendizado mais rápida que a da rede MLP. Cada neurônio da camada oculta é representado conforme mostrado na Figura 13(a), enquanto os neurônios da camada de saída são representados conforme mostrado na Figura 13(b).

3.2.2.1 Treinamento da Rede ELM

A rede ELM não possui uma etapa de treinamento semelhante ao da rede MLP, cuja atualização dos pesos sinápticos é feita através do algoritmo *backpropagation* por várias épocas de treinamento. No caso da rede ELM o aprendizado é executado de forma não recursiva através de três passos que contribuirão conjuntamente para obtenção dos pesos sinápticos das camadas oculta e de saída. Estes passos são descritos a seguir.

3.2.2.1.1 Passo 1: Inicialização Aleatória dos Pesos da Camada Oculta

Esta etapa de projeto da rede ELM envolve a especificação aleatória dos pesos w_{ij}, $i = 1, \ldots, h$ e $j = 0, \ldots, d$ que conectam as $(d+1)$ entradas aos h neurônios ocultos. Formalmente, pode-se escrever:

$$w_{ij} \sim U(a, b) \text{ ou } w_{ij} \sim N(0, \sigma^2) \tag{3.17}$$

em que $U(a,b)$ é um número uniformemente distribuído no intervalo (a,b), enquanto $N(0,\sigma^2)$ é um número normalmente distribuído com média zero e variância σ^2.

Em ambientes de programação tais como Matlab© e Octave, esta fase é facilmente implementada em apenas uma linha de código. Para isso, precisamos definir uma matriz de pesos $\mathbf{W}$, com h linhas e $d+1$ colunas:

$$\mathbf{W} = \begin{bmatrix} w_{10} & w_{11} & \dots & w_{1d} \\ w_{20} & w_{21} & \dots & w_{2d} \\ \vdots & \vdots & \vdots & \vdots \\ w_{h0} & w_{h1} & \dots & w_{hd} \end{bmatrix}_{h\times(d+1)} = \begin{bmatrix} \mathbf{w}_1^T \\ \mathbf{w}_2^T \\ \vdots \\ \mathbf{w}_h^T \end{bmatrix} \tag{3.18}$$

onde nota-se que a i-ésima linha da matriz $\mathbf{W}$ é composta pelo vetor de pesos do i-ésimo neurônio oculto.

3.2.2.1.2 Passo 2: Acúmulo das Saídas dos Neurônios Ocultos

Este passo corresponde à etapa de treinamento da rede, em que são calculadas as ativações dos neurônios ocultos e suas respectivas saídas. Este passo destina-se à geração de uma matriz formada pelas saídas dos neurônios ocultos para cada vetor de treinamento.

O fluxo de sinais se dá dos neurônios de entrada para os neurônios de saída, passando obviamente pelos neurônios da camada oculta. Por isso, diz-se que a informação está fluindo no sentido direto (*forward*), ou seja,

Entrada → Camada Oculta → Camada de Saída

Em notação matricial, a Equação (3.6) pode ser escrita como:

$$\mathbf{z} = \phi(\mathbf{u}) = \phi(\mathbf{W}\mathbf{x}) \tag{3.19}$$

em que a função de ativação $\phi(\cdot)$ é aplicada a cada um dos h componentes do vetor $\mathbf{u}$.

Para cada vetor de entrada $\mathbf{x}(n)$, $n = 1,\dots,N_1$, tem-se um vetor $\mathbf{y}(n)$ correspondente, que deve ser organizado como uma coluna de uma matriz $\mathbf{Y}$. Esta matriz terá h_1 linhas por N_1 colunas:

$$\mathbf{Z} = [\mathbf{z}(1)|\mathbf{z}(2)|\dots|\mathbf{z}(N_1)] = \begin{bmatrix} y_1(1) & y_1(2) & \dots & y_1(N_1) \\ y_2(1) & y_2(2) & \dots & y_2(N_1) \\ \vdots & \vdots & \vdots & \vdots \\ y_h(1) & y_h(2) & \dots & y_h(N_1) \end{bmatrix}_{h\times N_1} \tag{3.20}$$

Uma linha de "-1" deve ser adicionada à matriz $\mathbf{Y}$, correspondendo à ativação constante associada ao limiar do neurônio de saída, passando a ser escrita como:

$$\bar{\mathbf{Z}} = [\bar{\mathbf{z}}(1)|\bar{\mathbf{z}}(2)|\dots|\bar{\mathbf{z}}(N_1)] = \begin{bmatrix} -1 & -1 & \dots & -1 \\ y_1(1) & y_1(2) & \dots & y_1(N_1) \\ y_2(1) & y_2(2) & \dots & y_2(N_1) \\ \vdots & \vdots & \vdots & \vdots \\ y_h(1) & y_h(2) & \dots & y_h(N_1) \end{bmatrix}_{h \times N_1} \tag{3.21}$$

A matriz $\bar{\mathbf{Y}}$ será usada no Passo 3 para determinar os valores dos pesos do neurônio de saída da rede ELM usando o método dos mínimos quadrados.

3.2.2.1.3 Passo 3: Determinação dos Pesos do Neurônio de Saída

Para o problema de interesse, cada vetor de entrada $\mathbf{x}(n)$, $n = 1, \dots, N_1$, tem-se um escalar de saída desejada $\theta(n)$ correspondente. Ao organizar estes N_1 escalares como um vetor-coluna $\boldsymbol{\theta}$, tem-se então um vetor com N_1 componentes:

$$\boldsymbol{\theta} = \begin{bmatrix} \theta(1) \\ \theta(2) \\ \vdots \\ \theta(N_1) \end{bmatrix}_{N_1 \times 1} \tag{3.22}$$

Já que a ativação do neurônio de saída é linear, pode-se entender o cálculo dos pesos da camada de saída como o cálculo dos parâmetros de um mapeamento linear entre a camada oculta e a camada de saída. O papel de "vetor de entrada" para a camada de saída na iteração n é desempenhado pelo vetor $\bar{\mathbf{z}}(n)$, enquanto a "saída" é representada pelo escalar $\theta(t)$. Assim, busca-se determinar o vetor $\mathbf{m}$ que melhor represente a transformação

$$\theta(n) = \mathbf{m}^T \bar{\mathbf{z}}(n) \tag{3.23}$$

Para isso, pode-se usar o método dos mínimos quadrados, também conhecido como método da pseudoinversa (PRINCIPE; EULIANO; LEFEBVRE, 2000). Assim, de posse da matriz $\bar{\mathbf{Z}}$ e do vetor $\boldsymbol{\theta}$, o vetor de pesos $\mathbf{m}$ é determinado por meio da seguinte expressão:

$$\mathbf{m} = (\bar{\mathbf{Z}}\bar{\mathbf{Z}}^T)^{-1}\bar{\mathbf{Z}}\boldsymbol{\theta} = \bar{\mathbf{Z}}^{\dagger}\boldsymbol{\theta} \tag{3.24}$$

É importante destacar que a matriz inversa $(\bar{\mathbf{Z}}\bar{\mathbf{Z}}^T)^{-1}$ na Equação (3.24) não depende do tamanho do conjunto de treinamento (N_1), mas sim do número de neurônios ocultos (h).

Antes de estimar o vetor $\mathbf{m}$, faz-se necessária uma avaliação do posto da matriz $\bar{\mathbf{Z}}$ pois normalmente esta é de posto incompleto, ou seja, o número de colunas linearmente independentes é diferente do número de neurônios ocultos. Para estes casos, recomenda-se utilizar a função `pinv` do Matlab/Octave a fim de utilizar a Decomposição em Valores Singulares (SVD, sigla em inglês) que torna a obtenção da pseudo-inversa numericamente mais estável. Para maiores informações sobre a SVD pode-se consultar Lay, Camelier e Iório (1997).

3.2.2.1.4 Uso da Rede ELM Após o Treinamento

Uma vez determinada a matriz $\mathbf{W}$ e o vetor $\mathbf{m}$ de pesos, tem-se a rede ELM pronta para uso. Como o neurônio de saída é linear, sua saída é calculada como

$$\hat{\theta}(n) = \mathbf{m}^T \bar{\mathbf{z}}(n) \tag{3.25}$$

em que o vetor de saídas dos neurônios ocultos, $\bar{\mathbf{z}}(n)$, é calculado como na Equação (3.6).

A grande vantagem da rede ELM a ser destacada é a redução do tempo de projeto da rede, pois para este modelo não há treinamento dos pesos dos neurônios ocultos, uma vez que são definidos aleatoriamente (ver Equação (3.17)), enquanto o vetor de pesos da camada de saída ($\mathbf{m}$) é obtido através do método da pseudoinversa (ver Equação (3.24)), tornando o projeto da rede ELM bem mais veloz que o da rede MLP.

Uma aplicação da rede ELM para aprendizado da cinemática direta/inversa de robôs manipuladores pode ser encontrada em Feng, Yao-nan e Yi-min (2014).

3.3 Modelos de Regressão Baseados em Métodos de Kernel

Durante a descrição matemática de alguns algoritmos de aprendizado de máquina aparecem operações envolvendo produtos escalares de funções cuja a propriedade principal é realizar um mapeamento de um espaço de menor dimensão, em que as variáveis de entrada e saída estão relacionadas de modo não-linear, em um espaço de característica de maior dimensão, em que uma relação linear entre as variáveis é estabelecida. Estes produtos escalares de funções são chamadas de funções de *kernel* e o mapeamento não-linear é realizado por meio do chamado truque do *kernel* (*kernel trick*). Por serem um produto escalar, os *kernels* podem ser utilizados como um medida de similaridade entre os pontos e/ou funções aplicadas a eles. Na próxima seção são descritos dois modelos de regressão que fazem uso intensivo de funções de *kernel*: (1) Regressão de Mínimos Quadrados Baseada em Vetores-Suporte (LS-SVR, sigla em inglês) e os Processos Gaussiano (PG).

3.3.1 O Modelo LS-SVR

Máquinas de Vetores de Suporte (*Support Vector Machines*, SVMs), constituem um arcabouço de algoritmos de aprendizado de máquina cuja formulação matemática advém da *Teoria do Aprendizado Estatístico* desenvolvida por Vapnik (2000).

SVM foram introduzidas inicialmente na resolução de problemas de classificação binária onde demonstra-se a formulação básica de problemas de otimização envolvidos na obtenção de um modelo a partir de um conjunto de treinamento. Burges (1998) apresenta este desenvolvimento bem como outros tipos de aplicações deste modelo tais como regressão linear e não-linear. Mais recentemente, foram desenvolvidas diversas extensões das SVMs destinadas a solução de problemas de classificação multiclasses e aproximação de funções. Informações sobre estes últimos tópicos podem ser obtidas em Hsu e Lin (2002) e em Smola e Schölkopf (2004), respectivamente.

Uma destas extensões é de grande interesse para este trabalho e se concentra em uma versão não-linear da aplicação do algoritmo das SVMs, possibilitando a obtenção de modelos capazes de aproximar as funções de interesse para este trabalho. Para isso, realiza-se um pré-processamento sobre os dados chamado de *Truque do Kernel*. Seja X o espaço de entrada de onde são obtidos os exemplos de treinamento e φ um mapeamento tal que $\varphi : X \rightarrow \Im$, onde $\Im$ é o espaço das características (*feature space*). A transformação φ leva os dados de treinamento em um espaço de maior dimensionalidade no qual existe uma alta probabilidade de que uma SVM linear seja capaz de realizar a aproximação da função neste espaço. O truque do Kernel consiste em utilizar o mapeamento φ de forma implícita através de um tipo especial de função denominada $Kernel$, denotada por $K(.,.)$, que é definida através do produto escalar entre dois pontos do conjunto treinamento que são previamente aplicados ao mapeamento φ:

$$K(x_i, x_j) = \varphi(\mathbf{x}_i) \cdot \varphi(\mathbf{x}_j) \tag{3.26}$$

As funções de *kernel* facilitam a obtenção dos modelos SVM devido à simplicidade envolvida nos cálculos dos problemas de optimização quadrática na fase treinamento da SVMs, onde muitas vezes se desconhece o mapeamento $\varphi(.)$ diretamente. São exemplos de *kernels* utilizados na prática os polinomiais, os gaussianos e os sigmoidais. A Tabela 3 lista estes *kernels* e os parâmetros envolvidos em sua utilização:

Tipo de Kernel	**Função** $K(x_i, x_j)$	**Parâmetros**
Polinomial	$(\delta(\mathbf{x}_i \cdot \mathbf{x}_j) + \kappa)^d$	δ, κ e d
Gaussiano	$\exp(-\sigma\|\mathbf{x}_i - \mathbf{x}_j\|^2)$	σ
Sigmoidal	$\tanh(\delta(\mathbf{x}_i \cdot \mathbf{x}_j) + \kappa)$	δ e κ

Tabela 3 – Tipos de função de *kernel*.

Haykin (2009), Suykens et al. (2002) e Scholkopf e Smola (2001) trazem mais informações acerca das propriedades das funções de *kernel* e suas aplicações.

Para o caso da aproximação de funções não lineares através de SVM uma modificação foi proposta por Suykens et al. (2002) na formulação do problema primal de otimização. Esta modificação tem como objetivo simplificar a formulação da SVM sem que esta perca algumas de suas vantagens já conhecidas e bastante exploradas, tal como a convexidade da solução. Através de alguns ajustes na formulação do problema primal, resolve-se um conjunto de equações lineares ao invés de um problema de programação quadrática (PQ).

3.3.1.1 Aplicação em Problemas de Regressão

Dado um conjunto X com N dados de treinamento tal que $X = \{\mathbf{x}_k, y_k\}_{k=1}^{N}$, procura-se através do uso deste conjunto de dados obter um modelo de regressão não linear no qual o truque do Kernel é aplicado, ou seja, o conjunto de dados é mapeado em um espaço de maior de dimensionalidade no qual uma relação linear é suficiente para estimar com acurácia a variável de saída. Considerando um modelo deste tipo tem-se:

$$y(\mathbf{x}) = \mathbf{w}^T \varphi(\mathbf{x}) + b \tag{3.27}$$

onde $\mathbf{x} \in \mathbb{R}^d$ e $\varphi : \mathbb{R}^d \to \mathbb{R}^h$ é a transformação que mapeia o espaço de entrada no espaço de características. Formula-se então o problema primal de minimização no espaço dos pesos:

$$\begin{cases} \min\limits_{\mathbf{w},b,\mathbf{e}} J_p(\mathbf{w}, e) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + \gamma\frac{1}{2}\|e\|^2 = \frac{1}{2}\mathbf{w}^T\mathbf{w} + \gamma\frac{1}{2}\sum_{k=1}^{N} e_k^2 \\ \text{Tal que } y_k = \mathbf{w}^T\varphi(\mathbf{x}_k) + b + e_k, \qquad\qquad k = 1, \ldots, N \end{cases} \tag{3.28}$$

Em determinadas situações a dimensão do vetor de parâmetros $\mathbf{w}$ pode ter dimensão infinita, o que impede a solução direta do problema primal. Logo, a formulação do problema dual e a derivação do Lagrangiano tornam-se necessárias. O Lagrangiano assume a seguinte forma:

$$\mathcal{L}(\mathbf{w}, b, \mathbf{e}, \boldsymbol{\alpha}) = J_P(\mathbf{w}, \mathbf{e}) - \sum_{k=1}^{N} \alpha_k \{\mathbf{w}^T\varphi(\mathbf{x}_k) + b + e_k - y_k\} \tag{3.29}$$

onde α_k são os *multiplicadores de Lagrange*. As condições para encontrar o ótimo do

problema são as seguintes:

$$\begin{cases} \frac{\delta \mathcal{L}}{\delta \mathbf{w}} = \mathbf{0} \rightarrow \mathbf{w} = \sum_{k=1}^{N} \alpha_k \varphi(\mathbf{x}_k) \\ \\ \frac{\delta \mathcal{L}}{\delta b} = 0 \rightarrow \sum_{k=1}^{N} \alpha_k = 0 \\ \\ \frac{\delta \mathcal{L}}{\delta e_k} = 0 \rightarrow \alpha_k = \gamma e_k, \qquad\qquad k = 1, \ldots, N \\ \\ \frac{\delta \mathcal{L}}{\delta \alpha_k} = 0 \rightarrow \mathbf{w}^T \varphi(\mathbf{x}_k) + b + e_k - y_k = 0, \quad k = 1, \ldots, N \end{cases} \tag{3.30}$$

Após a eliminação de $\mathbf{w}$ e $\mathbf{e}$ tem-se a seguinte solução dual:

$$\begin{cases} \textit{Resolver para } \boldsymbol{\alpha}, b \\ \\ \left[\begin{array}{c|c} 0 & \mathbf{1}_v^T \\ \hline \mathbf{1}_v & \boldsymbol{\Omega} + \mathbf{I}/\gamma \end{array}\right] \left[\begin{array}{c} b \\ \hline \boldsymbol{\alpha} \end{array}\right] = \left[\begin{array}{c} 0 \\ \hline \mathbf{y} \end{array}\right] \end{cases} \tag{3.31}$$

onde $\mathbf{y} = [y_1 \ ... \ y_N]^T$, $\mathbf{1}_v = [1 \ ... \ 1]^T$ e $\boldsymbol{\alpha} = [\alpha_1 \ ... \ \alpha_N]^T$. O truque de Kernel sendo aplicado, resulta em:

$$\begin{aligned} \Omega_{kl} &= \varphi(\mathbf{x}_k)^T \varphi(\mathbf{x}_l) \\ &= K(\mathbf{x}_k, \mathbf{x}_l) \qquad k, l = 1, \ldots, N \end{aligned} \tag{3.32}$$

O modelo LS-SVR resultante para estimação de funções é então dado por:

$$y(\mathbf{x}) = \sum_{k=1}^{N} \alpha_k K(\mathbf{x}, \mathbf{x}_k) + b \tag{3.33}$$

onde α_k e b são a solução do sistema linear da Equação (3.31).

O modelo apresentado acima conserva a convexidade do problema de otimização pois apresenta apenas um mínimo global, porém a esparsidade da solução foi perdida devido ao fato de todos os pontos do conjunto de treinamento contribuírem para a obtenção do modelo. Suykens et al. (2002) propõem a utilização de métodos clássicos de poda, tais como o *Optimal Brain Damage* (LECUN et al., 1989) e o *Optimal Brain Surgeon* (HASSIBI; STORK et al., 1993) para que a esparsidade da solução seja retomada através da escolha dos vetores suporte que tem maior contribuição para os índices de performance adotados. Detalhar este tipo de procedimento de poda, porém, está fora do escopo deste trabalho.

3.3.2 Modelo de Regressão Baseado em Processos Gaussianos (PG)

Seja um vetor $\mathbf{y} = [y_1|y_2|\dots|y_N]^T$ composto de N saídas ruidosas que correspondem a um conjunto de vetores de entrada $\mathbf{X} = [\mathbf{x}_1|\mathbf{x}_2|\dots|\mathbf{x}_N]$. De acordo com Murphy (2012), Regressão baseada em Processo Gaussiano é um método de aprendizagem supervisionada não-paramétrico que assume o vetor $\mathbf{y}$ como um ponto N-dimensional amostrado de uma distribuição gaussiana multivariada de dimensão N. Para este trabalho, considera-se que as saídas do modelo são contaminadas por ruído, ou seja, o modelo obtido através dos dados descreve o comportamento global da nuvem de pontos e não uma interpolação dos dados de treinamento. Para cada saída, tem-se:

$$y_n = f(\mathbf{x}_n) + \varepsilon_n \tag{3.34}$$

em que $\varepsilon_n \sim \mathcal{N}(0, \sigma_y^2)$, em que σ_y^2 é a variância da saída. A presença de ruído descrita pela Equação (3.34) pode ser expressa em termos de uma distribuição gaussiana sobre os valores das componentes dos vetores alvos (*targets*) cuja média é a saída ruidosa y_n. Em termos vetoriais, tem-se:

$$p(\mathbf{f}|\mathbf{y}) = \mathcal{N}(\mathbf{f}|\mathbf{y}, \sigma_y^2\mathbf{I}_N) \tag{3.35}$$

onde $\mathbf{I}_N$ é a matriz identidade de dimensão N e $\mathbf{f} = [f(\mathbf{x}_1)\ f(\mathbf{x}_2)\ \dots\ f(\mathbf{x}_N)]$. Por simplicidade, adota-se a seguinte densidade *a priori* para o vetor de saída:

$$p(\mathbf{y}) = \mathcal{N}(\mathbf{y}|0, \mathbf{K}_y) \tag{3.36}$$

em que a matrix de covariância das saídas ruidosas $\mathbf{K}_y$ é dada por

$$\mathbf{K}_y = \mathbf{K} + \sigma_y^2\mathbf{I}_N \tag{3.37}$$

em que

$$\mathbf{K} = k(\mathbf{X}, \mathbf{X}) = \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}_1) & k(\mathbf{x}_1, \mathbf{x}_2) & \dots & k(\mathbf{x}_1, \mathbf{x}_N) \\ k(\mathbf{x}_2, \mathbf{x}_1) & k(\mathbf{x}_2, \mathbf{x}_2) & \dots & k(\mathbf{x}_2, \mathbf{x}_N) \\ \vdots & \vdots & \ddots & \vdots \\ k(\mathbf{x}_N, \mathbf{x}_1) & k(\mathbf{x}_N, \mathbf{x}_2) & \dots & k(\mathbf{x}_N, \mathbf{x}_N) \end{bmatrix} \tag{3.38}$$

onde a matriz $\mathbf{K} = k(\mathbf{X}, \mathbf{X})$ e $k(\cdot, \cdot)$ é chamada de *função de covariância* ou *kernel*. Esta função de covariância expressa a relação de similaridade entre os dois vetores sobre os quais ela é aplicada. Para valores $\mathbf{x}_i$ e $\mathbf{x}_j$ próximos, espera-se que os componentes da matriz $\mathbf{K}$ que correspondam a estes vetores apresentem valores de maior magnitude, indicando uma

maior semelhança entre os valores de y_i e y_j. Isto vai ao encontro da percepção intuitiva que se tem *a priori* da possíveis funções que realizarão a regressão dos dados. Esta percepção indica que, para pontos próximos no espaço de entrada, deve-se ter variações suaves na saída. Diversas funções podem ser utilizadas para este fim, cada uma com o objetivo de capturar certos aspectos da dinâmica de variação dos dados. Uma função de covariância bastante utilizada é o *kernel gaussiano* ou RBF (*Radial Basis Functions*), definida como

$$k(\mathbf{x}_i, \mathbf{x}_j) = \sigma_f^2 \ \exp\left\{\frac{1}{2l^2}(\|\mathbf{x}_i - \mathbf{x}_j\|^2)\right\} + \sigma_y^2 \delta_{ij} \tag{3.39}$$

Nota-se que a primeira parte do segundo membro da equação, ao se desconsiderar brevemente os hiperparâmetros σ_f e l, tende para o valor unitário quando os vetores $\mathbf{x}_i$ e $\mathbf{x}_j$ são próximos e para o valor nulo quando os vetores estão mais distantes.

Consideremos agora o problema de se estimar a saída para um novo padrão de entrada. Seja $\mathbf{x}_*$ um padrão de entrada para o qual se quer determinar o valor de saída, $\mathbf{f}_*$. Devido à suposição que os dados de saída, inclusive o novo valor $\mathbf{f}_*$, advir de uma distribuição gaussiana, toma-se também um distribuição gaussiana conjunta para o vetor composto por $\mathbf{y}$ e $\mathbf{f}_*$:

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{f}_* \end{bmatrix} = \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{K}_y & \mathbf{K}_* \\ \mathbf{K}_*^T & \mathbf{K}_{**} \end{bmatrix}\right) \tag{3.40}$$

onde a média zero foi assumida por questões de simplicidade. Utilizando resultados já conhecidos pela literatura para obtenção da distribuição condicional a partir de uma distribuição gaussiana conjunta, temos:

$$p(\mathbf{f}_*|\mathbf{X}_*, \mathbf{X}, \mathbf{t}) = \mathcal{N}(\mathbf{f}_*|\boldsymbol{\mu}_*, \boldsymbol{\Sigma}_*) \tag{3.41}$$

$$\boldsymbol{\mu}_* = \mathbf{K}_*^T \mathbf{K}_y^{-1} \mathbf{y} \tag{3.42}$$

$$\boldsymbol{\Sigma}_* = \mathbf{K}_{**} - \mathbf{K}_*^T \mathbf{K}_y^{-1} \mathbf{K}_* \tag{3.43}$$

nos quais $\mathbf{K}_* = k(\mathbf{X}_*, \mathbf{X})$ e $\mathbf{K}_{**} = k(\mathbf{X}_*, \mathbf{X}_*)$.

3.3.2.1 Estimação dos Hiperparâmetros

Como dito anteriormente, a função de covariância tem papel fundamental na predição de novos valores de entrada pois o processo de aprendizagem do PG corresponde a obteção dos valores adequados dos hiperparâmetros através dos dados de treinamento. Por exemplo, para o caso em que a saída é ruidosa, a Equação (3.39) possui três parâmetros, reunidos em um vetor $\boldsymbol{\beta} = \{\sigma_f^2, l, \sigma_y^2\}$. Estes parâmetros definem, respectivamente, a máxima variação vertical da função que estima os dados, o parâmetro de escala horizontal do *kernel* e a variância do ruído.

Uma das técnicas mais utilizadas para a obtenção dos hiperparâmentos é a maximização da função log-verossimilhança, definida como (3.44):

$$\log p(\mathbf{y}|\mathbf{X}) = \log \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{K}_y) = -\frac{1}{2}\mathbf{y}\mathbf{K}_y^{-1}\mathbf{y} - \frac{1}{2}\log|\mathbf{K}_y| - \frac{N}{2}\log(2\pi) \tag{3.44}$$

A maximização desta função pode ser realizada através de algortimos de otimização baseados em gradiente. Fletcher (2013) e Wright e Nocedal (1999) apresentam alguns destes algoritmos. Para a otimização não linear, precisa-se também do gradiente da função de verossimilhança logarítmica com respeito ao vetor de parâmetros $\boldsymbol{\beta}$:

$$\frac{\partial}{\partial\beta_i}\log p(\mathbf{y}|\mathbf{X}) = \frac{1}{2}\mathbf{y}^T\mathbf{K}_y^{-1}\frac{\partial\mathbf{K}_y}{\partial\beta_i}\mathbf{K}_y^{-1}\mathbf{y} - \frac{1}{2}\mathrm{tr}(\mathbf{K}_y^{-1}\frac{\partial\mathbf{K}_y}{\partial\beta_i}) \tag{3.45}$$

Após a aplicação de métodos de otimização, obtém-se um vetor ótimo de parâmetros $\boldsymbol{\beta}_{optim}$ cujas componentes são empregadas nas funções de *kernel* gaussianos adotadas para o processo de regressão.

Aplicações de modelos de processos gaussianos no aprendizado de mapeamentos inversos em robótica, dentre eles são ainda escassos na literatura. As poucas referências encontradas são Danafar, Gretton e Schmidhuber (2010) e Grochow et al. (2004) e são voltadas para a área de computação gráfica. Estes trabalhos apresentam um sistema de cinemática inversa baseado em modelos de aprendizado de posturas (i.e. poses) humanas. Dado um conjunto de restrições, o sistema proposto pode produzir, em tempo real, a postura que mais provavelmente satisfaça àquelas restrições. O modelo é representado como uma distribuição de probabilidade sobre um espaço de posturas, sendo usado uma variante de processos gaussianos. A abordagem proposta visa substituir a abordagem clássica de cinemática inversa em problemas de animação e visão computacional.

3.4 Sistemas Neuro-Fuzzy

De acordo com Jang e Sun (1996), os sistemas Neuro-Fuzzy são aqueles que unem a adaptabilidade das redes neurais nas tarefas de reconhecimento de padrões com a capacidade dos sistemas de inferência fuzzy de incorporar conhecimento humano e realizar tomadas de decisões. Esta seção apresenta o ANFIS (Adaptive Network-based Fuzzy Inference System), uma classe de redes adaptativas que funcionam de maneira equivalente a um sistema de inferência fuzzy.

3.4.1 Adaptive-Network-Based Fuzzy Inference System (ANFIS)

Muitos problemas computacionais podem ser bem caracterizados e resolvidos através de conjuntos que representam a pertinência de seus elementos de forma binária.

Estas pertinências são tratadas de forma precisa e isenta de incertezas, ou seja, pode-se afirmar com exatidão se um elemento pertence ou não a determinado conjunto através de valores verdadeiros ou falsos. Por exemplo, pode-se considerar que um homem é alto quando sua altura é maior ou igual a 1,75 m. Neste caso, todos os homens que possuir uma altura maior ou igual a 1,75 m serão ditos altos, caso contrário serão ditos baixos. Tais conjuntos formam a base daquilo que se denomina *raciocínio exato*.

O raciocínio exato se demonstra ineficiente ao tentar caracterizar grandezas imprecisas ou vagas devido ao fato de nem sempre estas grandezas podem ser atribuídas a valores verdadeiros ou falsos. Além do mais, necessita-se tratar formalmente expressões linguísticas complexas que relacionem as grandezas envolvidas. Neste contexto, a *Lógica Fuzzy* e a *Teoria dos Conjuntos Fuzzy* fornecem todos os mecanismos formais para descrever estes tipos de variáveis. Todo raciocínio envolvido nestes casos é dito aproximado, devido a sua similaridade ao modo de raciocínio humano. Para o exemplo citado anteriormente, a Lógica Fuzzy definirá graus de pertinência aos elementos do conjunto de homens altos. Por exemplo, um homem com 1,70 m pertence ao conjunto dos homens altos com grau de pertinência 0,9.

Uma das aplicações da Lógica Fuzzy é a de se estimar saídas através de um conjunto de entradas fornecidas. Sistemas que realizam estes mapeamentos são denominados *Sistemas de Inferência Fuzzy* e suas saídas podem ter diversas finalidades, desde auxílio na tomada de decisões, em problemas de classificação, dentre outros. A Figura 14 ilustra um sistema de inferência fuzzy que é constituído das seguintes partes:

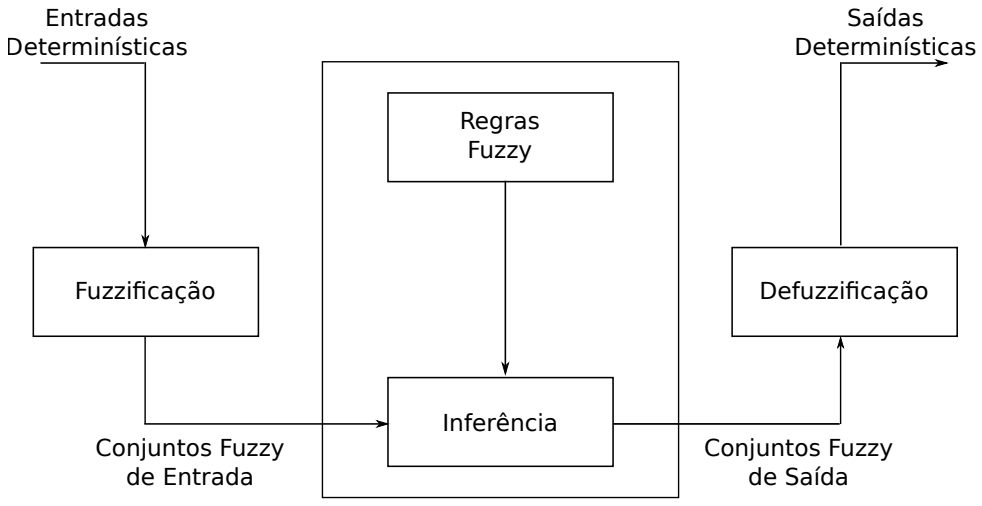


Figura 14 – Componentes de um Sistema de Inferência Fuzzy (do inglês, FIS).

- **Fuzzificador ou Processo de Fuzzificação**: realiza a transformação de uma variável não-fuzzy em variáveis fuzzy através da aplicação de funções de pertinência. Este estágio também é responsável pela remoção de ruído que contamina as variáveis de entrada.

- **Regras Fuzzy**: são relações entre variáveis linguísticas que definem, em conjunto com a base de dados, o comportamento dinâmico do sistema de inferência fuzzy. Elas são baseadas no conhecimento e experiência de especialistas no domínio do problema. Estas regras possuem o formato:

$$\text{SE } \textit{premissa(s)} \text{ ENTÃO } \textit{consequências} \tag{3.46}$$

  A parte das premissas normalmente envolve mais de uma variável linguística que são combinadas através de operadores lógicos. Estas variáveis podem vir acompanhadas de modificadores de vários tipos cuja função é a de intensificar determinado aspecto destas variáveis. Por exemplo, pode-se aplicar um modificador de contraste para aumentar a diferença de grau de pertinência entre os valores mais baixos e mais altos do espaço de discurso de um conjunto fuzzy. Engelbrecht (2007) traz maiores detalhes acerca de modificadores linguísticos. Já a parte consequente das regras normalmente utilizam apenas uma variável linguística porém múltiplas variáveis podem estar presentes.

- **Banco de Dados**: define as funções de pertinência dos conjuntos fuzzy utilizados no conjunto de regras.

- **Mecanismo de Inferência**: processo pelo qual as entradas fuzzyficadas são utilizadas para gerar as saídas para cada regra. Esta saída é chamada de *força de disparo da regra*. Após o cálculo de todas as forças de disparo, estas são acumuladas gerando apenas uma saída fuzzy. Nesta fase é possível atribuir pesos que *a priori* quantificam o grau de confiança da regra. Esta análise de confiança também é feita por um especialista no domínio do problema.

- **Defuzzificador ou Processo de Defuzzificação**: realiza a operação inversa do estágio fuzzyficador, ou seja, transforma a saída fuzzy do FIS em um ponto pertencente a um conjunto ordinário (também conhecido como *crisp*). As referências Engelbrecht (2007) e Jang e Sun (1996) apresentam alguns dos tipos de defuzzificadores mais utilizados, suas vantagens e desvantagens.

Como exposto em Lee (1990), Sistemas de Inferência Fuzzy foram aplicados com sucesso em diversas áreas tais como controle, predição e inferência. Porém, existem algumas questões que são inerentes à aplicação destes sistemas. Uma delas é inexistência de um método padronizado de se mapear o conhecimento e experiência de um especialista em um banco de dados de regras e seus conjuntos Fuzzy associados. Além do mais, faz-se necessário um método efetivo de ajustar os parâmetros das funções de pertinência de forma a minimizar o erro de mapeamento do Sistema de Inferência Fuzzy. Isto posto, Jang (1993) propôs uma arquitetura *feedfoward* híbrida denominada *ANFIS (Adaptive-Network-Based*

Fuzzy Inference System), cujas as características são endereçadas à solução dos entraves citados anteriormente.

Como uma forma de simplificar a descrição da arquitetura do ANFIS, serão utilizado como exemplo um sistema de duas entradas, denominadas x e y, e uma saída z. Além do mais, serão utilizadas regras fuzzy do tipo Takagi e Sugeno (TK) (TAKAGI; SUGENO, 1985), onde as saídas de cada regra são determinadas por uma combinação linear das variáveis de entrada mais uma constante. A Figura 15 ilustra a arquitetura do ANFIS, onde os sistemas retangulares são sistemas adaptativos, ou seja, possuem parâmetros ajustáveis que determinam suas relações entrada-saída. As funções de cada nó na mesma camada são descritas em seguida:

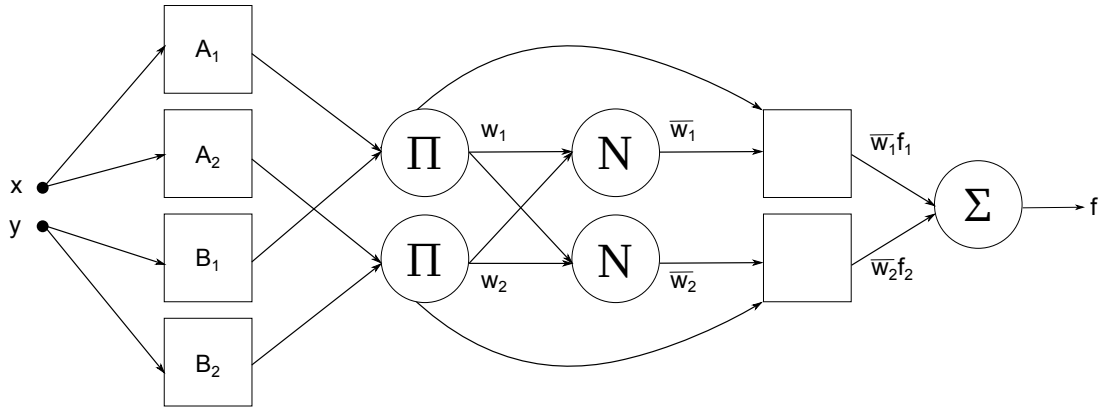


Figura 15 – Diagrama de blocos do ANFIS. Os blocos retangulares dispõem de parâmetros que são adaptados durante a fase de treinamento.

- **Camada 1**: tem como função encontrar uma representação fuzzy dos valores de entrada não fuzzy. Isto é realizado através da aplicação da função de pertinência a um valor obtido do universo de discurso. Qualquer função que siga as restrições básicas de uma função de pertinência pode ser aplicada desde que esta seja diferenciável por partes. Em Jang (1993), funções com formato de sino foram utilizadas. À medida que o treinamento supervisionado se desenrola, os parâmetros são atualizados e, consequentemente, o formato das funções é alterado. Por exemplo, a função:

$$\mu(x) = \frac{1}{1 + [(\frac{x - c_i}{a_i})^2]^{b_i}} \tag{3.47}$$

  altera o seu formato à medida que a_i, b_i e c_i são atualizados. Como tais parâmetros definem a interface entre o espaço de entrada não-fuzzy e o espaço fuzzy, ou seja, a parte *SE* de uma regra *SE-ENTÃO* fuzzy, ele são chamados de *parâmetros de premissa*, onde i denota uma regra fuzzy.

- **Camada 2**: realiza uma operação E lógica caso uma das regras fuzzy possua mais de uma variável linguística associada à parte da premissa de uma regra fuzzy. A

operação de interseção de duas variáveis fuzzy A e B pode ser realizada através de operações matemáticas chamadas T-normas, porém a mais comumente utilizada é a multiplicação:

$$w_i = \mu_A(x) \times \mu_B(x) \tag{3.48}$$

Este bloco tem como saída o que se chama *força de disparo da regra*.

- **Camada 3**: esta camada realiza uma normalização com respeito ao somatório das forças de ativação de todas as regras:

$$\overline{w}_i = \frac{w_i}{\sum_{i=1}^{N} w_i} \tag{3.49}$$

onde N é o número de regras. Os novos valores produzidos são chamados de *forças de disparo normalizadas*.

- **Camada 4**: após o processo de normalização esta camada aplica a parte consequente da regra *SE-ENTÃO* fuzzy. Em (JANG, 1993) a parte consequente é do tipo Takagi-Sugeno (TS), ou seja, a saída de cada regra é uma combinação linear das variáveis de entrada mais um termo constante:

$$O_i = \overline{w}_i f_i = \overline{w}_i(p_i x + q_i y + \cdots + r_i) \tag{3.50}$$

onde o conjunto p_i, q_i, r_i é chamado de *parâmetros consequentes*.

- **Camada 5**: camada de nó único que realiza a acumulação das saídas da camada anterior:

$$O = \sum_{i}^{N} \overline{w}_i f_i \tag{3.51}$$

A partir da descrição da arquitetura fica clara a necessidade de ajustar os parâmetros dos nós adaptativos. Para isso, um algoritmo de treinamento híbrido é empregado, onde a primeira fase é dita *foward* e a segunda é dita *backward*, de forma similar àquela apresentadas para as redes neurais MLP. Na primeira fase o fluxo de sinais se propaga da camada de entrada até a camada de saída, enquanto os parâmetros das premissas são mantidos fixos e os parâmetros da parte conseqüente são ajustados através do algoritmo *Least Mean Squares* (LMS). Na segunda fase, após a determinação dos parâmetros da parte consequente, o fluxo de sinais se propaga da camada de saída em direção a camada de entrada e os parâmetros da premissa são ajustados através de *error backpropagation*. Devido à fixação de parte dos parâmetros, o algoritmo converge mais rapidamente, pois reduz a dimensão do espaço de busca do algoritmo de retropropagação do erro (JANG; SUN, 1996).

As aplicações do modelo ANFIS para aprendizado da cinemática direta/inversa de robôs manipuladores podem ser encontradas nas seguintes referências: Mashhadany e MIEEE (2012), Alavandar e Nigam (2008) e Duka (2015).

3.5 Regressão Baseada em Distância

O algoritmo apresentado nesta seção supõe a existência de um mapeamento entre as configurações geométricas dos pontos do espaço de entrada e as configurações geométricas dos respectivos pontos no espaço de saída. Estas configurações geométricas são representadas através de matrizes de distâncias em relação a pontos de referência tomados de ambos os espaços. A capacidade destes modelos de generalizar a partir de novos pontos advém de um mapeamento linear entre as matrizes de distância. A próxima seção detalhará o *Minimal Learning Machine* (MLM), a formação das matrizes de distância e a obtenção do mapeamento entre elas.

3.5.1 Minimal Learning Machine (MLM)

De acordo com Junior et al. (2013), o modelo MLM consiste de um método de aprendizagem supervisionado em que se assume um mapeamento geométrico entre os pontos do espaço de entrada e os pontos do espaço de saída. Este mapeamento é representado por matrizes de distância de entrada e saída que são calculadas com respeito aos padrões de entrada e saídas desejadas e dois conjuntos de pontos de referência tomados nos espaço de entrada e saída respectivamente. O aprendizado da rede MLM se desenvolve através da determinação de um modelo de regressão linear entre as duas matrizes de distância. Então, dado um ponto no espaço de entrada podemos estimar a localização do ponto de saída através do modelo de regressão aprendido.

Dado um conjunto, com $\mathbf{x}_i \in \mathbb{R}^D$, dos padrões de entrada e um conjunto, com $\mathbf{y}_i \in \mathbb{R}^S$, das saídas correspondentes. Tem-se como objetivo encontrar o mapeamento $f : \mathbf{X} \to \mathbf{Y}$ a partir dos dados utilizando-se o seguinte modelo:

$$\mathbf{Y} = f(\mathbf{X}) + \mathbf{R}, \tag{3.52}$$

tal que as matrizes $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_D]$ e $\mathbf{Y} = [\mathbf{y}_1, ..., \mathbf{y}_D]$ são construídas através da concatenação vertical dos padrões de entrada e saída respectivamente. A matriz $\mathbf{R}$ de dimensão $N \times S$ denota os resíduos.

Para incluir a noção de mapeamento geométrico entre o espaço de entrada e saída, a MLM define um modelo de regressão linear auxiliar que é baseado em matrizes de distância. Dois conjuntos de K pontos de referência, $R = \{\mathbf{m}_k\}_{k=1}^K$ e $T = \{\mathbf{t}_k\}_{k=1}^K$, são tomados do conjunto de entrada e do conjunto de saída respectivamente. As matrizes

de distância relativas a esses pontos de referência são $\mathbf{D}_x$ e $\mathbf{\Delta}_y$, respectivamente, e se relacionam através do seguinte mapeamento linear:

$$\mathbf{\Delta}_y = \mathbf{D}_x\mathbf{B} + \mathbf{E}, \tag{3.53}$$

em que a matriz $\mathbf{B}$ de dimensões $K \times K$ deve ser estimada a partir dos dados. A matriz $\mathbf{E}$ armazena os resíduos.

Normalmente, tem-se $K < N$, ou seja, o número de pontos de referência tomados é menor do que o número de padrões fornecidos pelo conjunto de dados. Baseando-se nesta afirmativa utiliza-se o método dos mínimos quadrados para estimar o mapeamento$\hat{\mathbf{B}}$, ou seja

$$\hat{\mathbf{B}} = (\mathbf{D}_x^T\mathbf{D}_x)^{-1}\mathbf{D}_x^T\mathbf{\Delta}_y, \tag{3.54}$$

Dado um ponto de teste $\mathbf{x} \in \mathbb{R}^D$ no espaço de entrada, calcula-se o vetor $\mathbf{d}(\mathbf{x}, R) = [d(\mathbf{x}, \mathbf{m}_1), ..., d(\mathbf{x}, \mathbf{m}_K)]$ que é composto pelas distâncias do ponto $\mathbf{x}$ ao conjunto R dos pontos de referência $\mathbf{m}_k$ do espaço de entrada. Aplicando o mapeamento $\hat{\mathbf{B}}$ tem-se:

$$\hat{\boldsymbol{\delta}}(\mathbf{y}, T) = \mathbf{d}(\mathbf{x}, R)\hat{\mathbf{B}} \tag{3.55}$$

em que $\hat{\boldsymbol{\delta}}(\mathbf{y}, T)$ é uma estimativa das distâncias do ponto $\mathbf{y}$ ao conjunto T dos pontos $\mathbf{t}_k$ de referência do espaço de saída.

Neste ponto, tem-se um problema de multilateração que consiste encontrar uma estimativa da posição de um ponto $\mathbf{y}$, denominada $\hat{\mathbf{y}}$, dados pontos fixos no espaço e as distâncias até o ponto desejado. Dado que o ponto $\mathbf{y} \in \mathbb{R}^S$, podemos solucionar este problema resolvendo um conjunto sobredeterminado de K equações não lineares correspondendo a hiperesferas de dimensão $(S + 1)$ centradas em $\mathbf{t}_k$ passando através de $\mathbf{y}$. A Figura 16 ilustra esta situação para $S = 2$

Dado o conjunto de $k = 1, \ldots, K$ esferas com cada raio igual a $\hat{\delta}(\mathbf{y}, \mathbf{t}_k)$, temos a seguinte relação:

$$(\mathbf{y} - \mathbf{t}_k)'(\mathbf{y} - \mathbf{t}_k) = \hat{\delta}^2(\mathbf{y}, \mathbf{t}_k) \tag{3.56}$$

Através da Equação (3.56) podemos estimar a saída minimizando a seguinte função de custo:

$$J(\mathbf{y}) = \sum_{k=1}^{K}((\mathbf{y} - \mathbf{t}_k)'(\mathbf{y} - \mathbf{t}_k) - \hat{\delta}^2(\mathbf{y}, \mathbf{t}_k))^2 \tag{3.57}$$

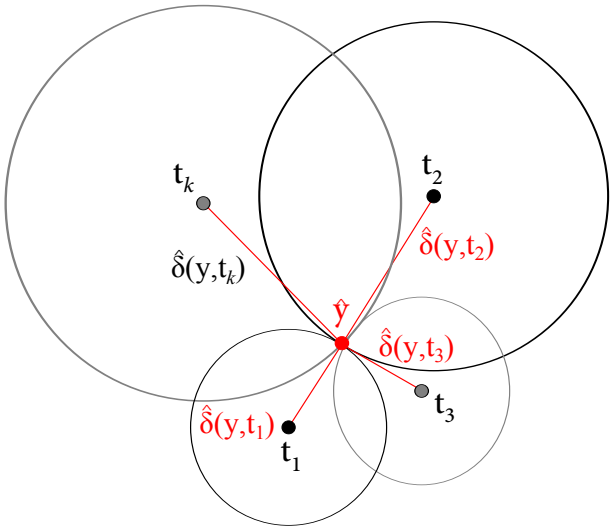


Figura 16 – Problema de multilateração para $S = 2$.

O problema de minimização da função de custo pode ser resolvido utilizando algoritmos de gradiente descendente. Neste trabalho, foi aplicado o algoritmo de Levenberg-Marquardt para encontrar a saída estimada:

$$\hat{\mathbf{y}} = \underset{\mathbf{y}}{\mathbf{argmin}} \sum_{k=1}^{K} ((\mathbf{y} - \mathbf{t}_k)'(\mathbf{y} - \mathbf{t}_k) - \hat{\delta}^2(\mathbf{y}, \mathbf{t}_k))^2 \tag{3.58}$$

Como o modelo MLM foi recentemente proposto na literatura, esta é a primeira vez que o modelo MLM está sendo aplicado no aprendizado dos modelos cinemáticos de robôs manipuladores.

3.6 Modelos Lineares Locais

Modelos locais são aqueles que particionam o domínio do problema em regiões menores de modo que um modelo mais estruturalmente simples seja adotado para cada região. Para o caso especial de aproximação de funções deste trabalho, quantiza-se o domínio do problema e atribui-se um modelo linear a cada uma das regiões obtidas. A seção seguinte é dedicada ao estudo do algoritmo Mapeamento Linear Local (Linear Local Mapping, LLM) bem como o quantizador vetorial K-médias associado a ele. O algoritmo que será apresentado foi adaptado de Walter, Riter e Schulten (1990) através da substituição do quantizador SOM pelo quantizador K-médias.

3.6.1 Linear Local Models (LLM)

3.6.1.1 Introdução

O algoritmo apresentado nesta seção é conhecido como modelos locais, pois particionam os espaços de entrada e de saída em regiões específicas, que são parametrizadas pelo uso de modelos lineares que estimam a dinâmica existente em cada partição. Tais modelos lineares atuam conjuntamente com os vetores-protótipos obtidos através um algoritmo de quantização vetorial chamado K-médias. Estes vetores-protótipos são responsáveis por mapear os vetores de entrada ao modelo local adequado. As duas próxima seções descrevem o processo de quantização vetorial através do algoritmo K-médias e o precedimento de treinamento dos Modelos Lineares Locais, respectivamente.

3.6.1.2 Algoritmo K-médias *Batch*

O primeiro algoritmo baseado em protótipos a ser descrito é o amplamente conhecido algoritmo K-médias (MACQUEEN et al., 1967), também conhecido no campo de quantização vetorial como algoritmo de Linde-Buzo-Gray (LBG) ou algoritmo de Lloyd generalizado (VASUKI; VANATHI, 2006). A aplicação do algoritmo K-médias a um conjunto de N_1 vetores visa encontrar um conjunto de g protótipos, $\{\mathbf{w}_i\}_{i=1}^{g}, g \ll N_1$, que particione os dados de entrada em exatamente g grupos distintos.

A região de influência de determinado protótipo é chamada de partição de Voronoi (ou Dirichlet) daquele protótipo, sendo definida como

$$V_i = \{\mathbf{x} \in \mathbf{R}^d \mid \ \|\mathbf{x} - \mathbf{w}_i\| < \|\mathbf{x} - \mathbf{w}_j\|, \forall j \neq i\} \tag{3.59}$$

em que $||\cdot||$ denota a norma euclidiana. Assim, com g protótipos o espaço de entrada é particionado em g regiões de Voronoi.

O algoritmo K-médias provê um método simples para a obtenção de g protótipos que minimizem a seguinte função-custo:

$$D = \sum_{i=1}^{g} \sum_{\mathbf{x} \in V_i} \|\mathbf{x} - \mathbf{w}_i\|^2 \tag{3.60}$$

também conhecida pelos seguintes nomes: erro de quantização, erro de reconstrução ou ainda distorção. Esta minimização é realizada através da implementação da seguinte sequência de passos:

- **Passo 1** - Seleção aleatória de g vetores do conjunto de dados para funcionar como protótipos iniciais;

- **Passo 2** - Separação do conjunto de dados em g regiões de Voronoi $V_i, i = 1, \ldots, g$, de acordo com a Equação (3.59);

- **Passo 3** - Os novos protótipos são recalculados como as médias aritméticas (centróides) dos dados alocados a cada região de Voronoi V_i, ou seja

$$\mathbf{w}_i = \frac{1}{N_i} \sum_{\mathbf{x} \in V_i} \mathbf{x}, \tag{3.61}$$

em que N_i é o número de vetores pertencentes à célula de Voronoi do i-ésimo protótipo; ou equivalentemente, é o número de vetores de dados para os quais o protótipo $\mathbf{w}_i$ é o mais próximo, segundo a métrica euclidiana.

Os passos 2 e 3 devem ser repetidos até que não haja mudanças substanciais no valor de D (Equação (3.60)) ou um determinado número máximo de iterações tenha sido alcançado. Um critério de parada comumente usado verifica se a taxa de variação da distorção D está abaixo de um limiar de distorção $0 < \epsilon \ll 1$ preestabelecido, ou seja

$$\left| \frac{D(t+1) - D(t)}{D(t+1)} \right| < \epsilon \tag{3.62}$$

em que o operador $|\cdot|$ denota o valor absoluto, e $D(t)$ corresponde ao valor do erro de quantização na t-ésima rodada de ajuste dos protótipos.

3.6.1.3 Modelos Lineares Locais com Quantizador K-médias

Grosso modo, a ideia básica do modelo LLM é associar cada protótipo do algoritmo K-médias a um modelo linear ARX treinado pelo algoritmo LMS (*Least-Mean Squares*). O algoritmo K-médias é usado para quantizar o espaço de entrada em um número reduzido de protótipos (e, assim, de células de Voronoi), enquanto o preditor linear associado a cada neurônio fornece uma estimativa local da saída do mapeamento a ser aproximado.

De maneira mais formal, seja d a dimensão do espaço cartesiano sobre o qual incidirá uma operação de quantização vetorial. O valor apropriado de d será definido de acordo com o modelo cinemático que se quer aproximar. Por exemplo, o robô PUMA 560 possui seu espaço de trabalho definido em três dimensões, implicando $d = 3$. Um elemento qualquer deste espaço é definido como o vetor de entrada $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^d$, também chamado de vetor de regressores.

Para a tarefa de aproximação das funções cinemáticas dos manipuladores robóticos, cada vetor de entrada $\mathbf{x} \in \mathbb{R}^d$ é definido como

$$\mathbf{x}(t) = \begin{pmatrix} p_1 \\ p_2 \\ \vdots \\ p_d \end{pmatrix}. \tag{3.63}$$

Para realizar a quantização vetorial do espaço $\mathcal{X}$, cada protótipo i de um algoritmo K-médias qualquer possui um vetor de pesos $\mathbf{w}_i$, definido como

$$\mathbf{x}(t) = \begin{bmatrix} w_{i1} \\ w_{i2} \\ \vdots \\ w_{id} \end{bmatrix}, \tag{3.64}$$

na qual g é o número total de protótipos do algoritmo. Associado a cada vetor de pesos existe um vetor de coeficientes $\mathbf{a}_i \in \mathcal{X} \subset \mathbb{R}^d$ contendo os coeficientes do i-ésimo modelo ARX, dado por

$$\mathbf{a}_i(t) = \begin{bmatrix} a_{i1} \\ a_{i2} \\ \vdots \\ a_{id} \end{bmatrix}, \tag{3.65}$$

em que d é a dimensão do modelo ARX correspondente. Assim, os parâmetros ajustáveis do modelo LLM são os conjuntos de vetores de pesos $\mathbf{w}_i$ e seus respectivos vetores de coeficientes $\mathbf{a}_i$, para $i = 1, \ldots, g$.

Seguindo a lógica competitiva do algoritmo K-médias, somente um neurônio por vez pode ser usado para estimar a saída do modelo LLM. A seleção do neurônio vencedor se dá com base na distância euclidiana do seu vetor de pesos ao vetor de entrada, ou seja,

$$i^*(t) = \arg\min_{i \in \mathcal{A}} \|\mathbf{x}(t) - \mathbf{w}_i(t)\| \tag{3.66}$$

Uma vez determinado o neurônio vencedor, a saída do modelo LLM é calculada da seguinte maneira:

$$y(t) = \mathbf{a}_{i^*}^T(t)\mathbf{x}(t) \tag{3.67}$$

Resta, portanto, descrever o processo de aprendizagem do modelo LLM. Diferente da rede neural SOM, o algoritmo K-médias não apresenta relações de vizinhança entre seus

protótipos, logo a regra de atualização dos coeficientes dos modelos lineares não leva em consideração nenhuma função de vizinhança. A cada época em que os vetores protótipos são recalculados e as regiões de Voronoi ajustadas, o método da pseudo-inversa é aplicado novamente para obter novos modelos lineares. A Equação (3.68) resume este procedimento:

$$\mathbf{A} = \mathbf{Y}\mathbf{X}^T(\mathbf{X}\mathbf{X}^T)^{-1} \tag{3.68}$$

onde a matriz $\mathbf{X}$ é formada pela concatenação lateral dos vetores coluna de entrada associados a uma região de Voronoi específica e $\mathbf{Y}$ a matriz formada pela concatenação lateral dos vetores de saída.

Neste trabalho, o treinamento é finalizado ao final das n_e épocas fixadas nas simulações, porém pode-se parar o treinamento em um ponto no qual as métricas de desempenho tenham sido satisfeitas. O erro de quantização, obtido através da Equação (3.62), pode ser utilizado para este propósito.

Uma vez apresentado o modelo LLM, é importante e construtivo tecer alguns comentários sobre as principais características deste algoritmo, que o diferencia das abordagens lineares tradicionais.

- O modelo LLM pode ser entendido como um banco de modelos lineares do tipo ARX, um para cada neurônio, cujo o modelo local a ser usado no instante t é escolhido entre os g modelos locais disponíveis. A escolha de qual modelo local e, consequentemente, de qual vetor de coeficientes usar é definida pela Equação (3.66). Se a rede possuir apenas um neurônio (i.e. $g = 1$), o modelo LLM se torna equivalente ao modelo ARX/LMS convencional.

- Do ponto de vista computacional, pode-se dizer que o algoritmo LLM implementa uma aproximação linear por região da função não-linear que se deseja aproximar, ou seja, o sinal de entrada é vetorizado e o conjunto de vetores resultantes é quantizado pelo algoritmo K-médias através de seus vetores-protótipos $\{\mathbf{w}_i\}_{i=1}^{g}$. Estes protótipos definem as coordenadas dos centróides de g regiões do espaço de entrada X, chamadas de células de Voronoi (PRINCIPE; EULIANO; LEFEBVRE, 2000) (ver Figura 17). Por sua vez, cada célula de Voronoi define o campo receptivo ou região de atração do i-ésimo neurônio, ou seja, a região do espaço de entrada para a qual o neurônio i é sempre escolhido vencedor. O que o modelo LLM faz é associar um vetor de coeficientes a_i a cada célula de Voronoi, tal que cada conjunto de coeficientes define um hiperplano aproximador da função não-linear de interesse.

As primeiras aplicações bem sucedidas de redes neurais auto-organizáveis do tipo Mapa de Kohonen no aprendizado de mapeamentos inversos em robótica, entre eles

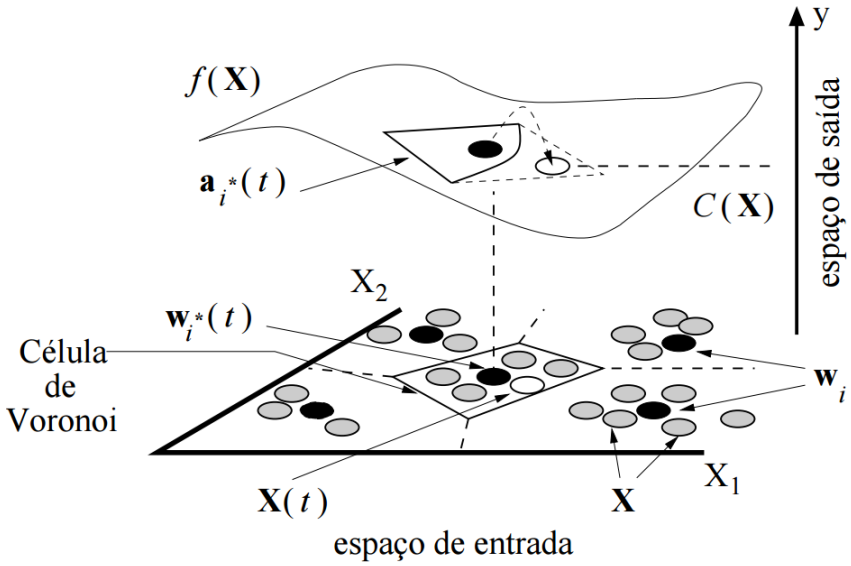


Figura 17 – Representação da arquitetura do modelo LLM.

a cinemática inversa, datam do final da década de 1980 (ANGULO; TORRAS, 1997), (MARTINETZ; RITTER; SCHULTEN, 1990) e (RITTER; MARTINETZ; SCHULTEN, 1989). Uma ampla revisão deste tipo de aplicação pode ser encontrada em Barreto, Araújo e Ritter (2003).

3.7 Conclusão

Este capítulo apresentou os diversos modelos de regressão não-linear utilizados por este trabalho na obtenção das funções que realizam o mapeamento inverso da cinemática dos manipuladores. Para cada modelo, foram apresentadas as particularidades relativas ao treinamento e utilização de um modelo já ajustado aos dados. O próximo capítulo apresenta a metodologia adotada para a justa comparação dos melhores modelos obtidos para três conjuntos de dados gerados pelas cinemáticas diretas de três manipuladores industriais diferentes.

4 Simulações e Resultados

4.1 Introdução

Este capítulo tem como objetivo apresentar os resultados obtidos a partir de simulações realizadas sobre três conjuntos de dados para fins de aproximação das funções que governam a cinemática inversa do robô planar, do manipulador PUMA 560 e do manipulador Motoman HP6. Tais resultados servirão de base para uma análise comparativa entre os desempenhos de modelos não-paramétricos que cobrem diferentes paradigmas de aprendizagem de máquina na tarefa de aprendizado da cinemática inversa.

Será apresentada, de forma geral, a metodologia utilizada na obtenção dos resultados bem como os devidos detalhamentos de cada um dos passos que a compõe. Esta metodologia, através dos resultados da fase de testes, ajudará na escolha dos melhores algoritmos para cada um dos manipuladores supracitados.

A validação dos modelos escolhidos para cada paradigma será realizada através da análise dos resíduos, i.e. erros de predição, para uma grande quantidade de realizações de treinamento, validação e teste. Figuras de mérito, tais como o valor médio ($\mu(\theta_k)$) e o desvio padrão ($\sigma(\theta_k)$) dos erros quadráticos médios (EQM) das diversas realizações, fornecerão meios quantitativos para as comparações entre os modelos.

Por fim, serão realizados testes de hipóteses a fim de comparar as distribuições acumuladas empíricas dos resíduos obtidos para cada conjunto de funções estimadas pelos modelos. Tais testes visam avaliar se há diferenças estatísticas significantes entre os desempenhos dos melhores modelos.

A seguir é descrita a metodologia adotada que norteou o desenvolvimento das simulações realizadas neste trabalho, bem como a coleta de seus resultados.

4.2 Metodologia de Trabalho

Ao se realizar estudos comparativos entre diferentes algoritmos de aprendizado de máquina é de suma importância a utilização de uma metodologia que forneça resultados não enviesados, garantindo assim uma base justa de comparação entre as execuções dos diversos algoritmos. Neste trabalho foi adotado um conjunto de atividades ordenadas que orientaram a obtenção dos resultados. Estas atividade foram executadas na seguinte ordem:

1. **Definição do problema:** foi realizado um profundo estudo dos conceitos de robó-

tica envolvidos no problema de aproximação da cinemática inversa bem como os modelos de simulação que representam as estruturas robóticas utilizadas para gerar os conjuntos de dados empregados nas simulações. O Capítulo 2 traz um apanhado de tais conceitos.

2. **Geração dos conjuntos de dados:** utilizando os conceitos absorvidos no passo anterior, foram gerados três conjuntos de dados contendo pontos gerados no volume de trabalho e que foram escolhidos arbitrariamente, um para cada estrutura robótica. Além do mais, também criou-se conjuntos específicos que correspondem a trajetórias fixas dentro do volume de trabalho dos robôs. A Seção 4.4 descreverá a estrutura destes conjuntos em maiores detalhes.

3. **Particionamento dos dados:** para cada algoritmo, foram executadas $n_r = 50$ rodadas independentes de treinamento. Em cada uma destas rodadas, os conjuntos principais de dados foram divididos em conjuntos de treinamento, validação e teste. Os dois primeiros foram utilizados para estimar os hiperparâmetros dos modelos de cada algoritmo enquanto o último foi utilizado para obter valores de EQMs para pontos realmente não vistos durante as fases de treinamento e validação. Os EQMs também foram utilizados para gerar uma métrica que serviu como critério de avaliação de desempenho e seleção dos melhores algoritmos no espaço das juntas. Na Seção 4.5 descrever-se-á em maiores detalhes a finalidade deste particionamento.

4. **Seleção de parâmetros:** os parâmetros livres de cada algoritmo serão selecionados através dos conjuntos de treinamento e validação advindos da fase anterior. Ao final deste passo, cada variável de junta terá associada a ela sete modelos de regressão, um para cada algoritmo, que aproximam a respectiva função de cinemática inversa. Isto ocorre devido ao fato deste trabalho utilizar uma abordagem MISO, ou seja, as variáveis de saída dos modelos de regressão são estimados de forma separada. Na Seção 4.6 são detalhados os critérios que foram adotados para a escolha dos melhores parâmetros dos regressores. Além do mais, desenvolve-se nesta parte do texto o conceito de erro quadrático médio que é a base do processo comparativo deste trabalho.

5. **Teste dos modelos:** os conjuntos de teste são apresentados a todos os algoritmos e as métricas de comparação são reunidas. Será apresentado o conceito de *Métrica de Seleção* que dá uma noção quantitativa do desempenho global das estimativas entregues pelos modelos, ou seja, quando se considera o desempenho da estimação dos ângulos de junta que mais contribuem para o posionamento do manipulador. Estas métricas são usadas para selecionar os dois melhores algoritmos para cada conjunto de dados. Os algoritmos de melhor desempenho (i.e. de menor Métrica de Seleção) são submetidos à fase seguinte, em que os resíduos de estimação são

analisados. Além do mais, são apresentados gráficos de estimação das trajetórias de teste que ilustram de forma qualitativa o quão boas foram as estimativas para os ângulos de junta que os atuadores rotacionais precisam posicionar para que elas sejam completadas. Na Seção 4.7 são detalhados os resultados obtidos nessa fase.

6. **Análise dos resíduos:** No Capítulo 5 serão executados dois tipos de análise: (*i*) análise da autocorrelação dos resíduos e (*ii*) testes de hipótese de Kolmogorov-Smirnov. Para simplificar a análise, estas duas análises serão aplicadas somente aos dois algoritmos que apresentaram menor métrica de seleção na fase de testes.

4.3 Definição do Problema

Como dito anteriormente no Capítulo 2, a cinemática inversa é de vital importância para a tarefa de controle de um robô manipulador. Porém, a obtenção de uma expressão analítica que relacione coordenadas cartesianas com ângulos de junta não é de fácil derivação, principalmente para robôs mais complexos[1]. Além do mais, a cinemática inversa pode apresentar múltiplas soluções. Pensando na solução deste problema decidiu-se avaliar sete modelos de regressão na tarefa de aprendizado das funções de cinemática inversa para os ângulos de juntas de cada manipulador a partir de padrões de treinamento amostrados de um *task space* específico. A Figura 18 ilustra como o problema foi abordado.

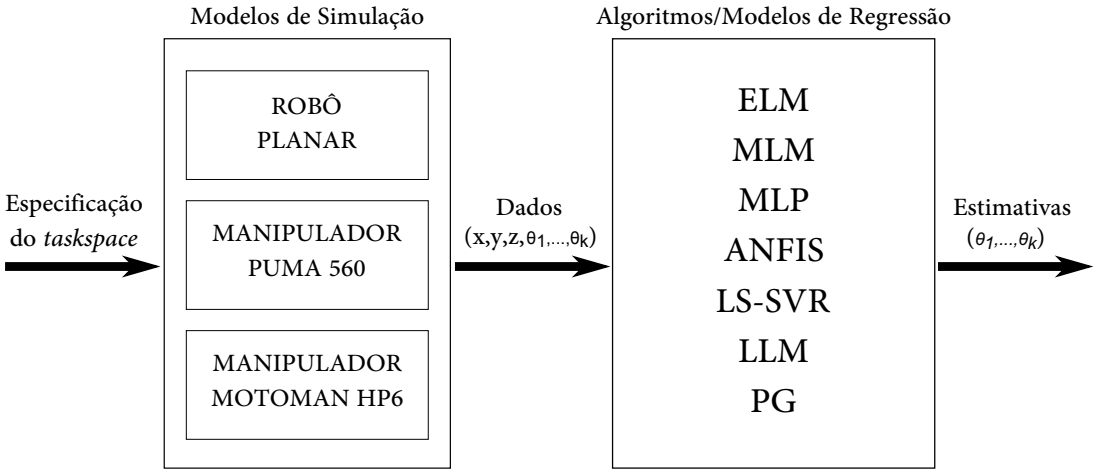


Figura 18 – Regressão das funções de cinemática inversa a partir de dados gerados por modelos de simulação.

[1] Por robôs mais complexos, entende-se robôs que não possuem uma estrutura padrão do tipo robô manipulador industrial. O robô Big Dog (Boston Dynamics - <http://www.bostondynamics.com/robot_bigdog.html>) é um exemplo

4.4 Geração dos Conjuntos de Dados

Nesta seção descreve-se como foram obtidos os conjuntos de dados utilizados na aproximação das funções de cinemática inversa para as três estruturas robóticas escolhidas. O uso de três conjunto de dados se deu pelo fato de os algoritmos implementados necessitarem testes de validação da implementação, onde os conjuntos de dados menos complexos, tais como o do robô planar, tornaram-se uma excelente arcabouço para tal fim. Além disso, é de suma importância apresentar dados de diferentes complexidades para medir a capacidade de generalização dos regressores escolhidos. As próximas subseções detalham como foram definidos os *task spaces* e suas respectivas amostragens, bem como a organização dos conjuntos de dados que foram submetidos às simulações nos passos posteriores.

4.4.1 Dados para o Robô Planar

Conforme descrito na Seção 2.3.5.2, o robô planar possui duas variáveis de junta $\boldsymbol{\theta} = [\theta_1\ \theta_2]^T$ e a localização do seu efetuador é dada por apenas duas componentes cartesianas $\mathbf{r} = [p_x\ p_y]^T$. Os comprimentos dos elos, denotados por l_1 e l_2, determinam como os ângulos de junta se relacionam à localização do efetuador. Obviamente, cada ângulo de junta possui limites físicos para suas faixas de existência.

Com tais informações em mente, foram escolhidas duas faixas para θ_1 e θ_2 dentro dos limites máximos para cada ângulo. O ângulo θ_1 foi tomado no intervalo $T_1 = \{\theta_1 \in \mathbb{R} \mid 0 \leq \theta_1 \leq \pi/2\}$ e o ângulo θ_2 foi tomado no intervalo $T_2 = \{\theta_2 \in \mathbb{R} \mid 0 \leq \theta_2 \leq \pi\}$, ambos com um passo de discretização de 0,1 radianos. Para determinar completamente as funções de cinemática direta, l_1 foi definido como 10 cm e l_2 com 7 cm.

Com as faixas T_1 e T_2 definidas, o conjunto dos pontos dos ângulos de junta foi montado através do produto cartesiano dos conjuntos T_1 e T_2. Tais pares (θ_1,θ_2) foram aplicados às Equações (2.10) e (2.11) para obter o vetor localização $\mathbf{r} = (x, y)$. Após a obtenção dos pares (x, y), o conjunto de dados foi montado através da inserção de 2016 linhas que seguem o formato $(x, y, \theta_1, \theta_2)$. A Figura 19 ilustra no plano cartesiano os pontos que compões o conjunto de dados.

4.4.2 Geração dos Dados para o Manipulador PUMA 560

O manipulador PUMA 560 possui uma topologia $6R$, ou seja, possui seis juntas rotacionais que são descritas por ângulos de junta que foram reunidos em um vetor $\boldsymbol{\theta} = [\theta_1\ \theta_2\ \ldots\ \theta_6]^T$. Estes ângulos, juntamente com as características geométricas dos elos, determinam, através das Equações (2.12) a (2.23), a posição do efetuador representada pelo vetor $\mathbf{r} = [p_x\ p_y\ p_z]^T$. Assim como no robô planar, este robô possui limites físicos sobre as faixas de valores das componentes do vetor $\boldsymbol{\theta}$. Estas restrições implicam diretamente no

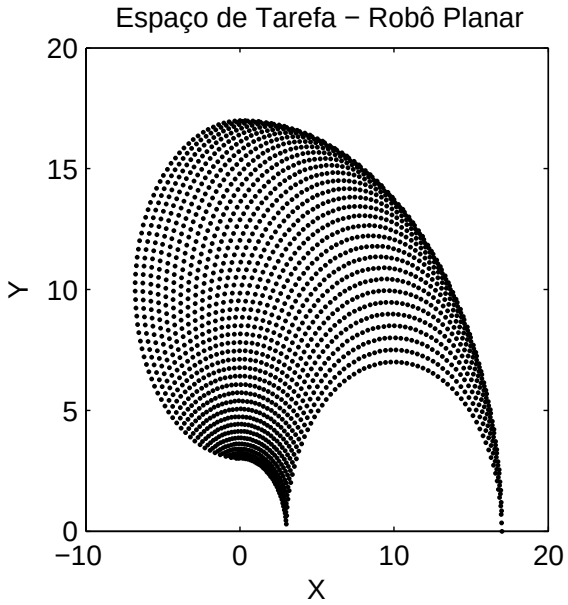


Figura 19 – Pontos do conjunto de dados do robô planar exibidos no espaço cartesiano.

formato do volume de trabalho no qual o manipulador pode atuar, impossibilitando que ele alcance certas posições no espaço cartesiano.

Para gerar o conjunto de dados empregado na estimação da cinemática inversa do robô PUMA 560 utilizou-se o modelo de simulação fornecido pela *Robotics Toolbox* para Matlab. Corke (1996) e Corke (2011) apresentam maiores detalhes acerca deste conjunto de scripts. Neste modelo de simulação estão prontamente disponíveis as equações de cinemática direta e inversa, valores limites para os ângulos de juntas bem, como os parâmetros de Denavit-Hartenberg e as transformações de elo relacionadas.

Por meio do modelo de simulação adotado, gerou-se no espaço cartesiano um conjunto de pontos que correspondem ao produto cartesiano de três intervalos $T_1 \times T_2 \times T_3 = \{\mathbf{r} = (x, y, z) \mid x \in [0{,}25; 0{,}75], y \in [-0{,}25; 0{,}25] \text{ e } z \in [0{,}25; 0{,}75]\}$. O passo de amostragem foi definido com o valor 0,05 para cada um dos eixo coordenados. Através da Figura 20(a) nota-se que estes pontos formam uma região espacial em formato de paralelepípedo. Esta região não corresponde a um espaço da tarefa completamente válido visto que existe a possibilidade de nem todos estes pontos serem alcançáveis pelo manipulador. Para contornar essa situação os pontos do paralelepípedo foram submetidos às equações da cinemática inversa do modelo de simulação e nos casos em que os pontos se mostraram inalcançáveis, estes eram removidos da região. A Figura 20(b) exibe a região obtida sem a presença de tais pontos inalcançáveis.

Após a obtenção somente de pontos (x, y, z) alcançáveis pelo efetuador, os valores correspondentes dos ângulos das juntas $(\theta_1, \theta_2, \ldots, \theta_6)$ são anexados lateralmente aos pontos cartesianos compondo uma nova amostra do conjunto de dados que segue o formato $(x, y, z, \theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6)$. Seguindo este procedimento, foram gerados 1066 amostras de

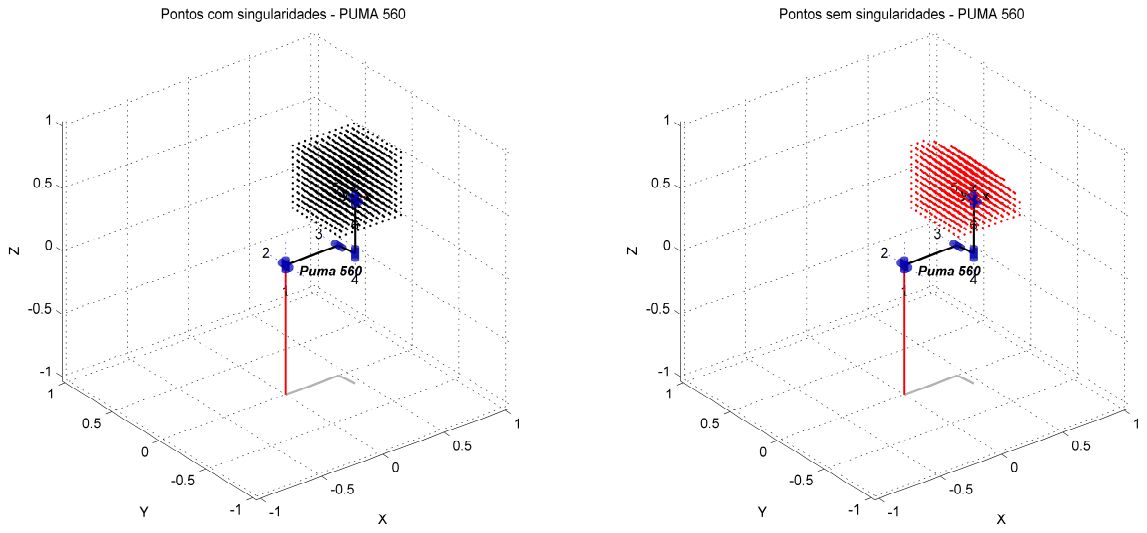


Figura 20 – Dados do robô PUMA 560:(a) Pontos submetidos ao modelo inverso do manipulador PUMA 560 para a remoção de singularidades. (b) Espaço da tarefa do manipulador após a remoção das singularidades.

dados.

Além do conjunto de dados principal, foram gerados dois conjuntos de teste adicionais com trajetórias conhecidas para que fosse possível avaliar o desempenho dos regressores também no espaço cartesiano e não apenas no espaço das juntas. A geração destes conjuntos passou pelos mesmo procedimento de remoção de singularidades, fornecendo apenas trajetórias compostas de pontos alcançáveis. O primeiro deles é uma circunferência de raio $R = 0{,}125\ m$ e de centro $P_c = [0{,}5\ 0\ 0{,}5]^T$. Já o segundo é uma função senoidal de amplitude $A = 0{,}1\ m$ e frequência $\omega = 30\ rad/s$. Ambos os conjuntos tiveram seus pontos posicionados dentro da região na qual os regressores foram treinados, a fim de garantir que os mesmos pudessem ser bem aproximados.

4.4.3 Geração dos Dados para o Manipulador Motoman HP6

Para o manipulador Motoman HP6 procedeu-se de forma similar àquela aplicada ao robô PUMA 560. Tomou-se uma região do espaço através do produto cartesiano de três intervalos $T_1 \times T_2 \times T_3 = \{\mathbf{r} = (x, y, z) \mid x \in [0{,}5; 1{,}25]\,, y \in [-0{,}25; 0{,}25]\ \text{e}\ z \in [0{,}5; 1{,}25]\}$. O passo de discretização dos intervalos também foi de 0,05. Após a remoção das singularidades obteve-se um conjunto de 1018 padrões que seguem o formato $(x, y, z, \theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6)$. Nas Figuras 21(a) e 21(b) são mostrados os pontos do conjunto de dados antes e depois da remoção das singularidades, respectivamente:

Além do mais, também foram geradas trajetórias auxiliares de teste dentro do espaço definido para a tarefa. A primeira delas é uma circunferência de raio $R = 0{,}125\ m$ e de centro $P_c = [0{,}65\ 0\ 0{,}65]^T$. Já a segunda é uma função senoidal de amplitude $A = 0{,}1\ m$

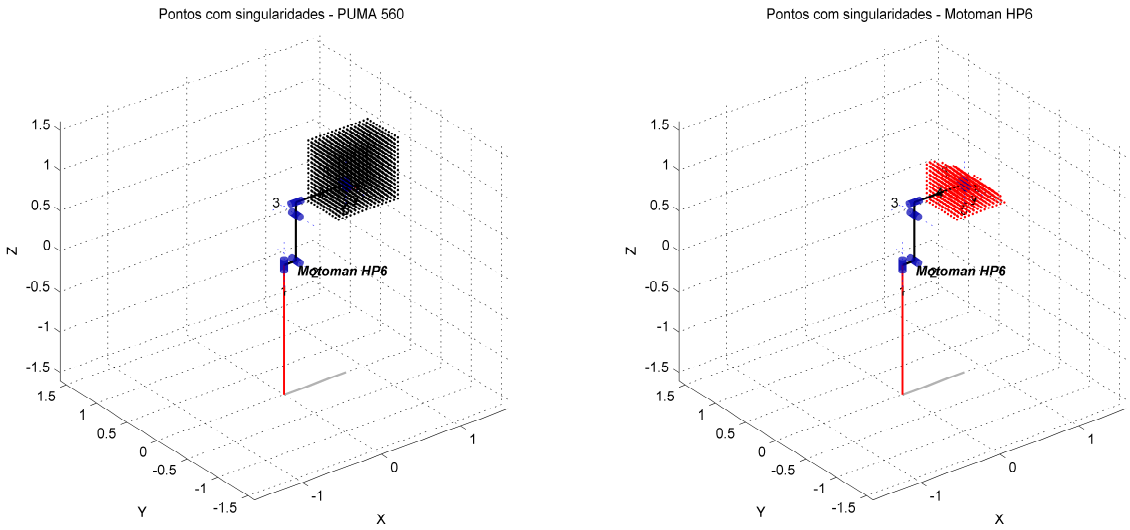


Figura 21 – Dados do robô Motoman HP6:(a) Pontos submetidos ao modelo inverso do manipulador para a remoção de singularidades. (b) Espaço da tarefa do manipulador após a remoção das singularidades.

e frequência $\omega = 30\ rad/s$.

4.5 Particionamento dos Dados

Após reunir os conjuntos de dados necessários à solução do problema de regressão, duas operações foram executadas sobre os conjuntos principais: embaralhamento dos vetores de dados e particionamento em subconjuntos de treinamento, validação e testes.

A operação de embaralhamento tem por objetivo garantir que os subconjuntos de treinamento, validação e teste contemplem amostras de todas as regiões do espaço. Isto é recomendado, quando as amostras não apresentam uma dependência temporal entre elas, uma vez que realizar esta operação antes de particionar os dados é considerada uma boa prática inerente em aprendizado de máquina. Por exemplo, nos conjuntos de dados usados nesta dissertação os vetores de dados foram armazenados em arquivo à medida que iam sendo gerados, causando uma ordenação não desejada que foi removida através do embaralhamento das linhas dos arquivos de dados.

A operação de particionamento visa separar 60% dos vetores de dados para treinamento, 20% para validação e os 20% restantes para teste. A união das duas primeiras parcelas será denominada *conjunto de seleção de modelos* e é reembaralhada e redividida a cada rodada de treinamento. Este procedimento tem como objetivo auxiliar na escolha do melhor modelo de regressão, bem como os melhores parâmetros de cada algoritmo. Já o conjunto de testes permanece intocado até a fase de testes. Ao proceder desta forma, garante-se que o conjunto de teste não tenha envolvimento algum com a fase de seleção

de modelos, não enviesando a escolha dos hiperparâmetros. Após a escolha dos melhores modelos, o conjunto de teste é submetido aos algoritmos para fins de cálculo dos respectivos erros médios quadráticos de teste e das métricas de seleção.

4.6 Seleção de Hiperparâmetros e Treinamento

De posse dos conjuntos de treinamento e validação os parâmetros livres de cada modelo de regressão foram variados, com o objetivo de se encontrar os seus valores ótimos. Por exemplo, para o caso do robô planar tomou-se um intervalo de 5 a 250 neurônios ocultos para a rede neural ELM em incrementos de 5 neurônios. Nestes casos, os limites superiores para os parâmetros livres foram definidos pelo alcance satisfatório do erro médio quadrático obtido por este limite sem que o desempenho do algoritmo seja muito prejudicado ou pela pouca melhoria nos indicadores de desempenho durante o treinamento. Quando o desempenho da rede começou a degradar, parou-se de aumentar o limite superior dos neurônios. Denotou-se por n_q a quantidade de elementos do conjunto de parâmetros utilizado na seleção de modelos. Por exemplo, para o caso supracitado da rede neural ELM $n_q = 50$.

Para a seleção dos hiperparâmetros dos modelos e, consequentemente, dos melhores modelos de regressão, foi adotado o critério do menor valor médio do EQM obtido em relação as n_r rodadas de treinamento. Para esta dissertação adotou-se $n_r = 50$ para todas as simulações. Para um determinado parâmetro livre q fixo, os conjuntos de treinamento e validação são reembaralhados e submetidos ao modelo através de n_r rodadas, onde uma rodada específica será denotada por n. A cada aplicação dos conjuntos de treinamento e validação, novos valores do EQM de treinamento, expressos a partir de agora por ξ, são calculados para cada variável de saída (ângulo de junta). Além disso, todos os pesos e limiares associados ao melhor resultado dentre as n_r rodadas para um dado modelo são armazenados para posterior seleção. Após a seleção, estes pesos e limiares serão resgatados e utilizados na fase de testes.

Ao fim da fase de treinamento obteve-se uma matriz $n_r \times n_q$ para cada ângulo de junta θ_k, onde $k = 1, \ldots, n_{DOF}$, ou seja,

$$\mathbf{EQM}(\theta_k) = \begin{bmatrix} \xi_{11}(\theta_k) & \xi_{12}(\theta_k) & \ldots & \xi_{1n_q}(\theta_k) \\ \xi_{21}(\theta_k) & \xi_{22}(\theta_k) & \ldots & \xi_{2n_q}(\theta_k) \\ \vdots & \vdots & \ldots & \vdots \\ \xi_{n_r1}(\theta_k) & \xi_{n_r2}(\theta_k) & \ldots & \xi_{n_rn_q}(\theta_k) \end{bmatrix} \tag{4.1}$$

em que $\xi_{ij}(\theta_k)$ é o EQM da i-ésima rodada para o j-ésimo valor do hiperparâmetro q para o k-ésimo ângulo de junta ($i = 1, \ldots, n_r;\ j = 1, \ldots, n_q;\ k = 1, \ldots, n_{DOF}$).

Após a determinação da matriz dos erros médios quadráticos na fase de treinamento, calculam-se os vetores de média, $\boldsymbol{\mu}_j(\theta_k)$, e de desvio padrão $\boldsymbol{\sigma}_j(\theta_k)$, para $j = 1, \ldots, n_q$, ou seja, médias e devios padrões sobre as colunas da matriz $\mathbf{EQM}(\theta_k)$

$$\boldsymbol{\mu}_j(\theta_k) = \begin{bmatrix} \mu_1(\theta_k) & \mu_2(\theta_k) & \ldots & \mu_{n_q}(\theta_k) \end{bmatrix} \tag{4.2}$$

$$\boldsymbol{\sigma}_j(\theta_k) = \begin{bmatrix} \sigma_1(\theta_k) & \sigma_2(\theta_k) & \ldots & \sigma_{n_q}(\theta_k) \end{bmatrix} \tag{4.3}$$

em que os elemento destes vetores são definido, respectivamente, pelas equações (4.4) e (4.5)

$$\mu_j(\theta_k) = \frac{1}{n_r} \sum_{i=1}^{n_r} \xi_{ij}(\theta_k), \quad j = 1, \ldots, n_q \tag{4.4}$$

$$\sigma_j(\theta_k) = \sqrt{\frac{1}{n_r} \sum_{i=1}^{n_r} \left[\xi_{ij}(\theta_k) - \mu_j(\theta_k)\right]^2}, \quad j = 1, \ldots, n_q \tag{4.5}$$

Após o cálculo destes vetores, os melhores parâmetros foram escolhidos ao se tomar os índices j^* que representam os parâmetros de menor média $\mu_j(\theta_k)$ para cada variável de saída. De forma mais compacta tem-se

$$j^* = \arg \min_{\forall j} \{\mu_j(\theta_k)\}. \tag{4.6}$$

Em seguida, retorna-se à matriz $\mathbf{EQM}(\theta_k)$ e busca-se na coluna referenciada por j^* o índice i^* da rodada de treinamento que apresentou o menor valor de ξ_{ij^*}. Este valor particular será denotado por $\xi_{i^*j^*}(\theta_k)$. Os parâmetros associados a cada modelo são guardados, através de j^* e i^*, e utilizados posteriormente na fase testes. Por exemplo, no caso das simulações da rede ELM, j^* e i^* apontam para as matrizes dos pesos da camada oculta e da camada de saída que obtiveram melhor resultado de acordo com o critério recém descrito. As próximas subseções apresentam os modelos selecionados para cada conjunto de dados bem como os valores de $\xi_{i^*j^*}(\theta_k)$ que nortearam esta escolha.

Para os casos dos modelos de regressão baseados em LS-SVR e PG, a obtenção de hiperparâmetros se dá através de processos de otimização subjacentes aos próprios algoritmos. Por exemplo, no LS-SVR um procedimento de duas fases é aplicado pela *toolbox* LS-SVMlab[2] na obtenção dos hiperparâmetros (BRABANTER et al., 2011). Na primeira fase utiliza-se o método de otimização *Coupled Simulated Annealing* (SOUZA et al., 2010) para se obter uma estimativa grosseira dos hiperparâmetros. Na fase seguinte, aplica-se o algoritmo *simplex* (NELDER; MEAD, 1965), em que as estimativas grosseiras

[2] Toolbox disponível em <http://www.esat.kuleuven.be/sista/lssvmlab/>

servem como ponto de partida para a realização de ajustes finos. Já no modelo PG, os hiperparametros foram obtidos através da maximização da função log-verossimilhança (Equação (3.44)).

A cada rodada de treinamento, novos conjunto de treinamento e validação são submetidos aos algoritmos e novos hiperparâmetros são calculados. Para cada conjunto de hiperparâmetros calculam-se os valores de ξ. Desta forma, a matriz $\mathbf{EQM}(\theta_k)$ dá lugar a um vetor coluna de dimensão $n_r \times 1$:

$$\mathbf{EQM}_{(j)}(\theta_k) = \begin{bmatrix} \xi_{j1}(\theta_k) & \xi_{j2}(\theta_k) & \dots & \xi_{jn_r}(\theta_k) \end{bmatrix}^T \tag{4.7}$$

A partir deste ponto, a seleção dos melhores parâmetros é realizada através da escolha do índice i^* que representa a rodada de menor ξ, ou seja,

$$i^* = arg \min_j \{\xi_{ij}(\theta_k)\}, \quad i = 1, \dots, n_r \tag{4.8}$$

As subseções seguintes reúnem os hiperparâmetros obtidos nesta etapa do trabalho para cada um dos manipuladores.

4.6.1 Rôbo Planar

Seguindo os procedimentos descritos nas seções anteriores, foram geradas 2016 amostras de dados. Estas amostras foram subdivididas em três partes: conjunto de treinamento, de validação e de teste. Os conjuntos de treinamento e validação contêm 1210 e 403 amostras, respectivamente, restando 403 amostras para a fase de teste.

Para ilustrar melhor o processo de seleção de modelos e o critério adotado para tal fim, as Figuras 22(a)(b) e 23(a)(b) exibem, respectivamente, para cada ângulo de junta as componentes de $\boldsymbol{\mu}_j(\theta_k)$ e $\boldsymbol{\sigma}_j(\theta_k)$ para as faixas de parâmetros adotadas para o algoritmo ANFIS. Este algoritmo apresentou um dos melhores desempenhos globais e por isso seus gráficos são apresentados aqui. Gráficos similares foram traçados para os outros modelos cujos hiperparâmetros são variados empiricamente nas simulações.

Os gráficos anteriores não foram construídos para os algoritmos LS-SVR e PG devido a inexistência de uma fase de seleção de parâmetros empírica pois os mesmos são calculados de forma exata através de processos de otimização convexa. Para estes casos, apenas a apresentação de novos conjuntos de treinamento e validação foi realizada a cada rodada de treinamento, possibilitando a escolha dos melhores modelos correspondentes às rodadas de menor EQM. Estes novos conjuntos foram obtidos do reembaralhamento das amostras dos conjuntos de treinamento e validação da rodada de treinamento anterior. Após o reembaralhamento, estes dados passaram por uma nova divisão, respeitando a proporção de 80% para treinamento e 20% para validação.

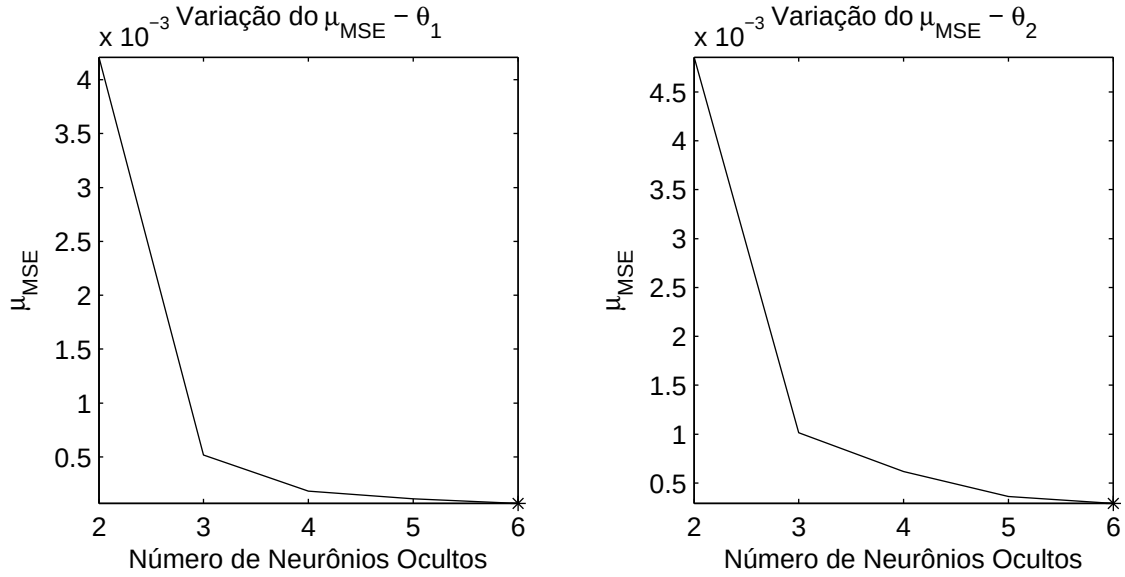


Figura 22 – **ANFIS**: Variações dos valores médios dos EQM de treinamento com o aumento da quantidade de neurônios ocultos: (a) Ângulo θ_1 (b) Ângulo θ_2.

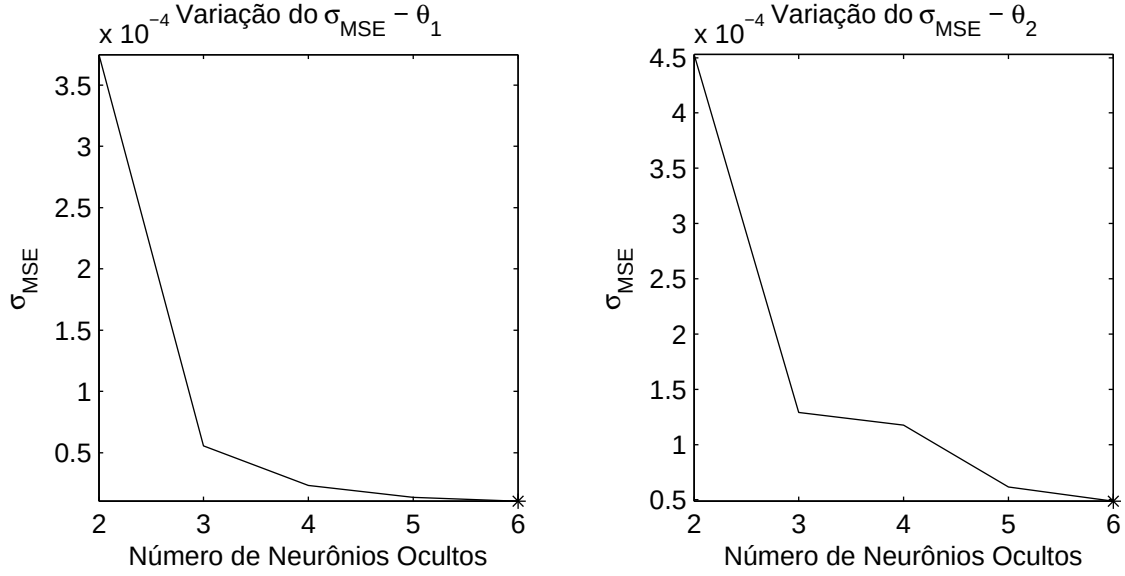


Figura 23 – **ANFIS**: Variações dos desvios padrões dos EQM de treinamento com o aumento da quantidade de neurônios ocultos: (a) Ângulo θ_1 (b) Ângulo θ_2.

Ao observar os gráficos anteriores, pode-se notar que à medida que o número de parâmetros[3] dos modelos aumenta, os valores de $\mu_j(\theta_k)$ tendem a decrescer. Neste momento deve-se fazer uma análise do custo-benefício da adoção de modelos mais complexos em relação ao custo computacional necessário e a possibilidade de *overfitting*. Também foram marcados por um asterisco nos gráficos os pontos correspondentes aos parâmetros selecionados. Os resultados da fase de seleção de modelos são mostrados na Tabela 4. Nesta tabela são apresentados os parâmetros selecionados bem como os respectivos valores

[3] Note que o número de neurônios ocultos é um hiperparâmetro, cujo o aumento, aumenta o número de parâmetros do modelo.

de $\xi_{i^*j^*}(\theta_k)$, $\mu_{j^*}(\theta_k)$ e $\sigma_{j^*}(\theta_k)$, para $k = 1, \ldots, n_{DOF}$:

Modelos	Parâmetros Selecionados		$\xi_{i^*j^*}(\theta_k)$
MLP	θ_1	$h = 40$, $\alpha = 0{,}01$, $\eta = 0{,}1$ e $n_e = 100$	4,8302E-03
	θ_2	$h = 20$, $\alpha = 0{,}01$, $\eta = 0{,}1$ e $n_e = 100$	9,1530E-03
ELM	θ_1	$h = 245$	5,4001E-04
	θ_2	$h = 240$	1,3308E-03
LS-SVR	θ_1	σ = 9,3089E-02, γ = 2,3894E+05, b = 2,1281E+00	2,3575E-05
	θ_2	σ = 1,6698E-01, γ = 2,5289E+05, b = -4,9928E+00	8,9343E-05
ANFIS	θ_1	$r = 6$, $n_e = 100$	4,3116E-05
	θ_2	$r = 6$, $n_e = 100$	1,4425E-04
MLM	θ_1	$k = 500$	1,3778E-04
	θ_2	$k = 500$	4,1524E-04
PG	θ_1	$\sigma_f = 4{,}4001$, $\sigma_y = 0{,}1$, $l = 2{,}3453$	3,8042E-05
	θ_2	$\sigma_f = 59{,}3166$, $\sigma_y = 0{,}1$, $l = 3{,}8697$	1,4003E-04
LLM	θ_1	$g = 100$, $n_e = 100$	3,4265E-04
	θ_2	$g = 80$, $n_e = 100$	9,9170E-04

Tabela 4 – **Robô Planar** - Parâmetros selecionados para os diversos modelos de regressão que estimaram os ângulos de junta.

Na Tabela 4 o número de épocas de treinamento dos algoritmos MLP, ANFIS e LLM, é denotado por n_e. Para a rede neural MLP, a taxa de aprendizagem (α) e o fator de momento (η) foram obtidos de forma empírica partindo-se de valores que são comumente utilizados na literatura. Já na rede neural ELM utilizou-se uma distribuição uniforme $U(-0{,}1; 0{,}1)$ para iniciar os pesos da camada oculta.

4.6.2 Manipulador PUMA 560

De forma análoga ao robô planar, cada junta rotacional do manipulador PUMA 560 possuirá associada a ela sete modelos de regressão, devido à abordagem MISO. Gráficos similares aos das Figura 22(a)(b) e 23(a)(b) também foram traçados para nortear a escolha dos melhores parâmetros, exceto para os modelos LS-SVR e PG, visto que nestes os hiperparâmetros não foram variados empiricamente em tempo de simulação. Estes gráficos novamente foram úteis para visualizar os pontos nos quais os melhores parâmentros foram tomados. Eles serão omitidos nesta seção visto que a utilidade dos mesmos foi apresentada na seção anterior.

Os resultados da fase de seleção de modelos são ilustrados pela Tabela 5. Nesta tabela são apresentados os parâmetros selecionados bem como os respectivos $\xi_{i^*j^*}(\theta_k)$, $\mu_{j^*}(\theta_k)$ e $\sigma_{j^*}(\theta_k)$, onde $k = 1, \ldots, n_{DOF}$:

Assim como no caso dos parâmetros obtidos para o robô planar, o número de épocas de treinamento foi escrito como n_e. Além do mais, a taxa de aprendizagem (α) e o fator de momento (η) da rede neural MLP também foram obtidos após alguma experimentação.

Modelos	Parâmetros Selecionados		$\xi_{i^*j^*}^{(\theta_k)}$
MLP	θ_1	$q = 10$, $\alpha = 0{,}01$, $\eta = 0{,}1$ e $n_e = 100$	9,7244E-03
	θ_2	$q = 250$, $\alpha = 0{,}01$, $\eta = 0{,}1$ e $n_e = 100$	8,8151E-03
	θ_3	$q = 240$, $\alpha = 0{,}01$, $\eta = 0{,}1$ e $n_e = 100$	2,6544E-02
	θ_4	$q = 240$, $\alpha = 0{,}01$, $\eta = 0{,}1$ e $n_e = 100$	6,9911E-00
	θ_5	$q = 90$, $\alpha = 0{,}01$, $\eta = 0{,}1$ e $n_e = 100$	3,0358E-02
	θ_6	$q = 30$, $\alpha = 0{,}01$, $\eta = 0{,}1$ e $n_e = 100$	1,4114E-00
ELM	θ_1	$h = 240$	2,8192E-06
	θ_2	$h = 110$	2,5972E-05
	θ_3	$h = 100$	1,1473E-04
	θ_4	$h = 60$	6,2437E-01
	θ_5	$h = 110$	2,4981E-05
	θ_6	$h = 240$	2,7380E-06
LS-SVR	θ_1	$\sigma = 4{,}0715$E-02, $\gamma = 3{,}9944$E+10, $b =$ -4,97978E-01	1,0030E-09
	θ_2	$\sigma = 2{,}8395$E-01, $\gamma = 1{,}9264$E+13, $b = 8{,}3677$E+03	5,3970E-06
	θ_3	$\sigma = 2{,}5554$E-01, $\gamma = 1{,}7630$E+12, $b =$ -6,3298E+03	2,4954E-05
	θ_4	$\sigma = 7{,}7530$E-01, $\gamma = 3{,}0895$E+00, $b =$ -3,4577E+00	0,0000E-00
	θ_5	$\sigma = 3{,}1632$E-01, $\gamma = 1{,}7427$E+13, $b =$ -1,1672E+04	5,5981E-06
	θ_6	$\sigma = 3{,}0849$E-02, $\gamma = 6{,}0429$E+09, $b = 1{,}9907$E-01	9,7003E-10
ANFIS	θ_1	$r = 3$, $n_e = 100$	9,1912E-08
	θ_2	$r = 4$, $n_e = 100$	1,3630E-05
	θ_3	$r = 4$, $n_e = 100$	6,7026E-05
	θ_4	$r = 4$, $n_e = 100$	4,8023E-07
	θ_5	$r = 3$, $n_e = 100$	2,5094E-05
	θ_6	$r = 3$, $n_e = 100$	1,0159E-07
MLM	θ_1	$k = 450$	1,8674E-05
	θ_2	$k = 450$	1,0587E-04
	θ_3	$k = 450$	3,6224E-04
	θ_4	$k = 500$	3,9905E-01
	θ_5	$k = 500$	1,4381E-04
	θ_6	$k = 450$	1,8139E-05
PG	θ_1	$\sigma_f = 1{,}0715$, $\sigma_y = 0{,}1$, $l = 0{,}2073$	1,0282E-09
	θ_2	$\sigma_f = 6{,}4245$, $\sigma_y = 0{,}1$, $l = 0{,}2791$	1,0130E-05
	θ_3	$\sigma_f = 9{,}8838$, $\sigma_y = 0{,}1$, $l = 0{,}2678$	4,1022E-05
	θ_4	$\sigma_f = 2{,}7811$, $\sigma_y = 0{,}1$, $l = 0{,}0692$	5,6190E-02
	θ_5	$\sigma_f = 3{,}8263$, $\sigma_y = 0{,}1$, $l = 0{,}2551$	1,0247E-05
	θ_6	$\sigma_f = 0{,}4238$, $\sigma_y = 0{,}1$, $l = 0{,}1869$	1,0840E-09
LLM	θ_1	$g = 60$, $n_e = 100$	1,5076E-04
	θ_2	$g = 60$, $n_e = 100$	2,9743E-04
	θ_3	$g = 50$, $n_e = 100$	1,2338E-03
	θ_4	$g = 30$, $n_e = 100$	7,9785E-01
	θ_5	$g = 40$, $n_e = 100$	3,5108E-04
	θ_6	$g = 50$, $n_e = 100$	1,5959E-04

Tabela 5 – **Manipulador PUMA 560** - Parâmetros selecionados para os diversos modelos de regressão que estimaram os ângulos das juntas $\theta_1, \theta_2, \ldots, \theta_6$.

Para a inicialização dos pesos dos neurônios da camada oculta da rede ELM utilizou-se uma distribuição uniforme $U(-0{,}1; 0{,}1)$.

Nota-se que para o ângulo de junta θ_4 a ordem de grandeza de $\xi_{i^*j^*}(\theta_4)$ apresentou valores mais elevados quando comparados aos outros ângulos de junta. Isso ocorreu devido à pouca variabilidade dos valores de θ_4 dentro do espaço da tarefa adotado nas simulações. Pode-se comprovar este fato através da exibição de um histograma para os valores deste ângulo. A Figura 24 exibe dois histogramas, um para θ_1 (Figura 24(a)) e outro para θ_4 (Figura 24(b)), para efeitos comparativos. Nota-se pelo histograma do ângulo θ_4 que a maioria de suas ocorrência se concentram apenas nos valores $\theta_4 = -\pi$ e $\theta_4 = \pi$, enquanto os outros ângulos de junta apresentam uma maior variabilidade nos valores das amostras.

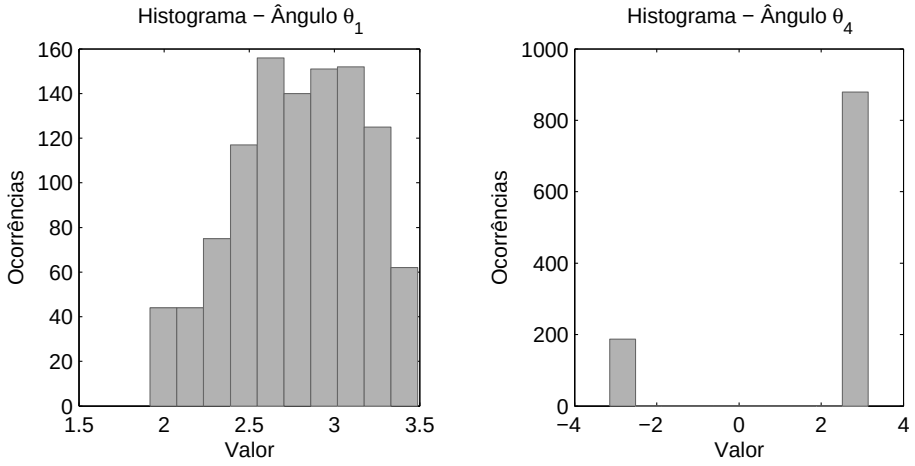


Figura 24 – Ângulo θ_4 (b) apresentando pouca variabilidade em suas amostras quando comparado com o ângulo de junta θ_1 (a).

4.6.3 Manipulador Motoman HP6

De forma semelhante às duas subseções anteriores, esta seção apresenta os parâmetros selecionados para os modelos de regressão adotados. A Tabela 6 reúne estes parâmetros para cada um dos ângulo de junta do manipulador.

Pode-se notar através da Tabela 6 que os valores de $\xi_{i^*j^*}(\theta_4)$ são bem pequenos quando comparados aos valores correspondentes das outras juntas. Valores baixos $\xi_{i^*j^*}$ indicam que os dados de validação puderam ser bem aproximados durante a fase de treinamento e seleção de parâmetros. Esta qualidade mais acentuada da aproximação foi consequência direta da concentração dos valores de θ_4 em torno de zero para o conjunto de treinamento. A Figura 25 exibe dois histogramas, um para θ_1 (Figura 25(a)) e outro para θ_4 (Figura 25(b)), para que possa se comparar a variabilidade dos dados de treinamento ambas as juntas

Modelos	Parâmetros Selecionados		$\xi_{i^*j^*}^{(\theta_k)}$
MLP	θ_1	$q = 110$, $\lambda = 0{,}01$, $\eta = 0{,}1$ e $n_e = 100$	1,8465E-03
	θ_2	$q = 220$, $\lambda = 0{,}01$, $\eta = 0{,}1$ e $n_e = 100$	1,5045E-03
	θ_3	$q = 100$, $\lambda = 0{,}01$, $\eta = 0{,}1$ e $n_e = 100$	3,3417E-03
	θ_4	$q = 250$, $\lambda = 0{,}01$, $\eta = 0{,}1$ e $n_e = 100$	2,9689E-02
	θ_5	$q = 10$, $\lambda = 0{,}01$, $\eta = 0{,}1$ e $n_e = 100$	2,7100E-00
	θ_6	$q = 50$, $\lambda = 0{,}01$, $\eta = 0{,}1$ e $n_e = 100$	3,5159E-01
ELM	θ_1	$h_1 = 245$	4,5640E-09
	θ_2	$h_1 = 245$	8,9203E-08
	θ_3	$h_1 = 245$	2,9075E-07
	θ_4	$h_1 = 75$	2,8522E-17
	θ_5	$h_1 = 245$	6,6834E-08
	θ_6	$h_1 = 235$	4,3924E-09
LS-SVR	θ_1	$\sigma =$ 1,1689E-01, $\gamma =$ 9,7790E+12, $b =$ -3,5158E-02	4,7919E-14
	θ_2	$\sigma =$ 1,0327E-01, $\gamma =$ 1,1821E+13, $b =$ 5,4580E+01	4,9038E-10
	θ_3	$\sigma =$ 1,1962E-01, $\gamma =$ 1,0722E+13, $b =$ 1,0124E+02	1,6512E-09
	θ_4	$\sigma =$ 4,7955E-03, $\gamma =$ 4,4230E+01, $b =$ 1,9493E-02	6,6782E-18
	θ_5	$\sigma =$ 8,9956E-02, $\gamma =$ 5,2079E+12, $b =$ -4,5052E+01	3,7435E-10
	θ_6	$\sigma =$ 1,0767E-01, $\gamma =$ 1,3727E+13, $b =$ -3,3073E-02	5,4336E-14
ANFIS	θ_1	$r = 3$, $n_e = 100$	1,0451E-08
	θ_2	$r = 3$, $n_e = 100$	3,3021E-07
	θ_3	$r = 4$, $n_e = 100$	4,7207E-07
	θ_4	$r = 2$, $n_e = 100$	9,8592E-18
	θ_5	$r = 3$, $n_e = 100$	3,3849E-07
	θ_6	$r = 3$, $n_e = 100$	1,0450E-08
MLM	θ_1	$k = 500$	2,7496E-06
	θ_2	$k = 500$	1,2300E-05
	θ_3	$k = 500$	3,4001E-05
	θ_4	$k = 400$	2,5648E-16
	θ_5	$k = 500$	9,5440E-06
	θ_6	$k = 500$	3,0279E-06
PG	θ_1	$\sigma_f = 0{,}9217$, $\sigma_y = 0{,}1$, $l = 0{,}3733$	9,5413E-14
	θ_2	$\sigma_f = 7{,}1066$, $\sigma_y = 0{,}1$, $l = 0{,}3140$	1,2437E-09
	θ_3	$\sigma_f = 12{,}6798$, $\sigma_y = 0{,}1$, $l = 0{,}3138$	4,6584E-09
	θ_4	$\sigma_f = 0{,}6248$, $\sigma_y = 0{,}1$, $l = 4{,}2045$	1,8506E-16
	θ_5	$\sigma_f = 16{,}0605$, $\sigma_y = 0{,}1$, $l = 0{,}3557$	8,8293E-10
	θ_6	$\sigma_f = 0{,}8749$, $\sigma_y = 0{,}1$, $l = 0{,}3651$	1,1391E-13
LLM	θ_1	$g = 50$, $n_e = 100$	1,8455E-05
	θ_2	$g = 40$, $n_e = 100$	6,0649E-05
	θ_3	$g = 40$, $n_e = 100$	1,1363E-04
	θ_4	$g = 100$, $n_e = 100$	6,6962E-17
	θ_5	$g = 40$, $n_e = 100$	5,0276E-05
	θ_6	$g = 40$, $n_e = 100$	2,1636E-05

Tabela 6 – **Manipulador Motoman HP6** - Parâmetros selecionados para os diversos modelos de regressão que estimaram os ângulos de junta $\theta_1, \theta_2, \ldots, \theta_6$.

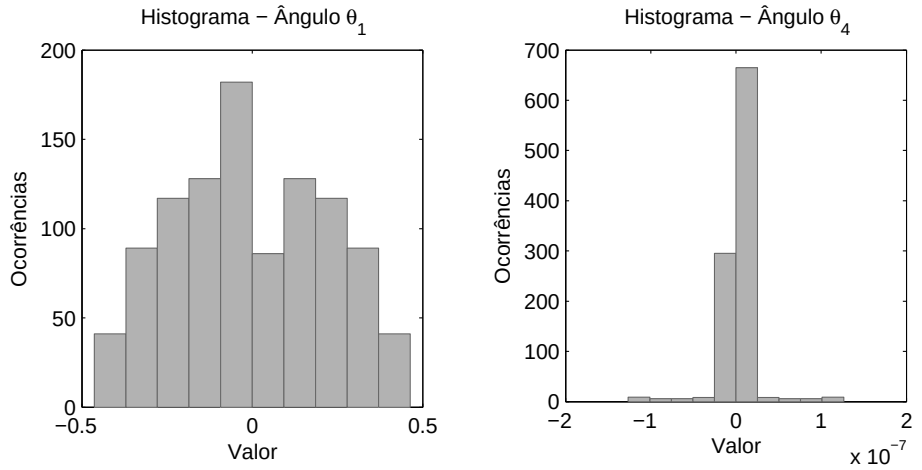


Figura 25 – Ângulo θ_4 (b) apresentando pouca variabilidade em suas amostras quando comparodo com o ângulo de junta θ_1 (a).

4.7 Teste dos Modelos Selecionados

Após a seleção dos melhores modelos, os dados de teste foram submetidos aos mesmos. A utilização destes novos dados permite a obtenção de erros quadráticos médios de teste, denotados por $\xi(\theta_k)_{teste}$, que servirão de base para formação de métricas que nortearão a análise comparativa. Os valores de $\xi(\theta_k)_{teste}$ foram calculados no espaço das juntas, ou seja, são medidos os desvios quadráticos dos ângulos estimados em relação aos ângulos desejados, fornecendo valores em rad^2.

Para selecionar os melhores algoritmos toma-se a média geométrica dos EQMs de teste das juntas que compõem a parte anterior ao punho do manipulador, obtidos do conjunto de teste principal. Utilizou-se esta abordagem pois os ângulos que localizam o punho são os que mais contribuem com o posicionamento do efetuador. Além do mais, houve situações em que a estimação de alguns ângulos de junta localizados após o pulso apresentaram valores de erros tão baixos que prejudicariam a métrica caso esta levasse em conta todos as juntas do manipulador.

Assim, optou-se por multiplicar os $\xi(\theta_k)_{teste}$, para $k = 1, \ldots, n_{DOF}^{(w)}$, onde o sobrescrito w indica que os graus de liberdade tomados são aqueles anteriores ao punho (*wrist*). Posteriormente toma-se a raíz deste produtório cujo índice é dado pelo valor de $n_{DOF}^{(w)}$. Matematicamente, tem-se que a métrica de seleção é dada pela média geométrica dos valores de $\xi(\theta_k)_{teste}$:

$$MS = \sqrt[n_{DOF}^{(w)}]{\prod_{k=1}^{n_{DOF}^{(w)}} \xi(\theta_k)_{teste}} \tag{4.9}$$

Para os manipuladores adotados neste trabalho, $n_{DOF}^{(w)} = 2$ (robô planar) ou $n_{DOF}^{(w)} = 3$ (PUMA 560 e Motoman HP6).

Além dos conjuntos de teste principais, foram gerados conjuntos de testes que definem trajetórias cujas equações paramétricas são conhecidas. Deve-se notar que o posicionamento destes locais geométricos bem como suas dimensões foram escolhidas de forma que os mesmos estivessem contidos na região em que o treinamento dos modelos foi realizado. Por exemplo, utilizou-se a equação paramétrica da circunferência para definir uma trajetória circular dentro do espaço de treinamento do robô planar e observou-se quão bem o modelo selecionado aproximou aquela trajetória.

Para avaliar a qualidade das aproximações no espaço cartesiano, calculam-se as distâncias médias entre as trajetórias desejadas e estimadas, fornecidas em milímetros (mm). Estas distâncias médias, denotadas por $\bar{d}$, foram calculadas ponto a ponto por meio da seguinte equação:

$$\bar{d} = \frac{1}{N} \sum_{n=1}^{N} \sqrt{\left[\mathbf{x}_d(n) - \hat{\mathbf{x}}(n)\right]^T \left[\mathbf{x}_d(n) - \hat{\mathbf{x}}(n)\right]} \tag{4.10}$$

em que $\mathbf{x}_d(n)$ é vetor posição desejado que compõe trajetória original e $\hat{\mathbf{x}}(n)$ o seu valor estimado correspondente. Além disso, o valor de N define quantos pontos compõem a trajetória em questão. As seções seguintes detalham para cada estrutura robótica os resultados obtidos na fase testes.

4.7.1 Robô Planar

Para o robô planar três conjuntos de testes foram submetidos aos algoritmos de aprendizagem de máquina. O primeiro deles adveio do conjunto de dados principal que foi particionado de acordo com o esquema descrito na Seção 4.5. O segundo conjunto descreve uma trajetória circular de centro $P_c = [0{,}5\ 10]^T$ e de raio $r = 2\ cm$. O terceiro conjunto de dados descreve uma espiral de Arquimedes centrada em $(0, 10)$. Estes conjuntos são compostos de 403, 1001 e 501 padrões de teste, respectivamente.

A Tabela 7 exibe para θ_1 e θ_2, respectivamente, os valores de $\xi(\theta_k)_{teste}$ resultantes da apresentação dos padrões de teste do conjunto principal, bem como as respectivas métricas de seleção. Esta tabela está ordenada do menor para o maior valor da MS, ou seja, da melhor para a pior avaliação de desempenho:

Através da Tabela 7 pode-se notar que os algoritmos LS-SVR e ANFIS obtiveram os menores valores de MS, ou seja, melhores desempenhos globais para as amostras do conjunto de teste principal. Neste ponto decidiu-se observar se um bom desempenho no espaço das juntas implicaria em um bom desempenho no espaço cartesiano. Logo, aplicou-se os conjuntos das trajetórias circular e espiral para se obter a Tabela 8, que exibe os respectivos valores de $\bar{d}$. Mais uma vez ordenou-se os resultados do menor para o maior erro.

Algoritmo	$\xi(\theta_1)_{teste}$	$\xi(\theta_2)_{teste}$	MS_{teste}
LS-SVR	**2,65703096E-05**	**9,30629610E-05**	**4,97263682E-05**
ANFIS	**6,17448062E-05**	**2,34180965E-04**	**1,20247488E-04**
PG	1,24763680E-04	3,66688781E-04	2,13891191E-04
MLM	2,04606079E-04	6,16900378E-04	3,55276747E-04
LLM	4,74189426E-04	1,85612631E-03	9,38166015E-04
ELM	8,22831035E-04	1,68373816E-03	1,17704376E-03
MLP	5,10573196E-03	1,38711572E-02	8,41560519E-03

Tabela 7 – **Robô Planar** - Erros quadráticos médios de teste e métricas de seleção para as variáveis de saída dos modelos de regressão.

Algoritmo	$\bar{d}_{círculo}$	$\bar{d}_{espiral}$	MS_{traj}
MLM	**1,79101761E-03**	**2,00137580E-03**	**1,89327740E-03**
LS-SVR	**4,00540802E-03**	**5,51389321E-03**	**4,69950977E-03**
PG	1,19238747E-02	1,42637553E-02	1,30414428E-02
ANFIS	3,53319688E-02	3,27859043E-02	3,40351370E-02
LLM	3,07223748E-02	4,17929692E-02	3,58326564E-02
ELM	9,23026200E-02	1,04414743E-01	9,81720650E-02
MLP	4,74368067E-01	3,90647974E-01	4,30477554E-01

Tabela 8 – **Robô Planar** - Médias das distâncias entre os pontos das trajetórias teóricas e estimadas, e as métricas de seleção correspondentes.

Ao se comparar a ordenação dos algoritmos que obtiveram melhor desempenho no espaço juntas com aqueles que aproximaram melhor as trajetórias teóricas, notou-se que não existe uma correspondência direta entre Tabelas 7 e 8, ou seja, um melhor desempenho no espaço das juntas não necessariamente implica em um melhor desempenho no espaço cartesiano.

Para o caso dos algoritmos LS-SVR e ANFIS, exibe-se, respectivamente, nas Figuras 26 e 27 os gráficos conjuntos dos valores desejados (em linha sólida) e estimados (em linha tracejada) executados pelos ângulos de junta para percorrer as trajetórias de teste.

Os gráficos apresentados nas Figuras 26 e 27 dão uma noção visual da capacidade de generalização dos dois melhores modelos de regressão selecionados. Neste caso, a aproximação se apresentou tão fiel que mal se nota o erro de um gráfico em relação ao outro, visto que os mesmos estão praticamente sobrepostos.

Uma outra forma de visualizar o poder de generalização dos modelos é através da exibição de gráficos dos valores estimados contra os valores desejados. A Figura 28 exibe tais gráficos bem como as retas $\theta_1 = \hat{\theta}_1$ e $\theta_2 = \hat{\theta}_2$ que funcionam como guias visuais e auxiliam na identificação visual da qualidade da aproximação dos modelos.

Em ambos os gráficos pode-se notar que os pontos dos gráficos estão bem concentrados em torno da reta $\theta_{estimado} = \theta_{desejado}$ indicando boas estimações das funções de cinemática inversa. Os espaçamentos entre conjuntos de pontos ao longo da nuvem de

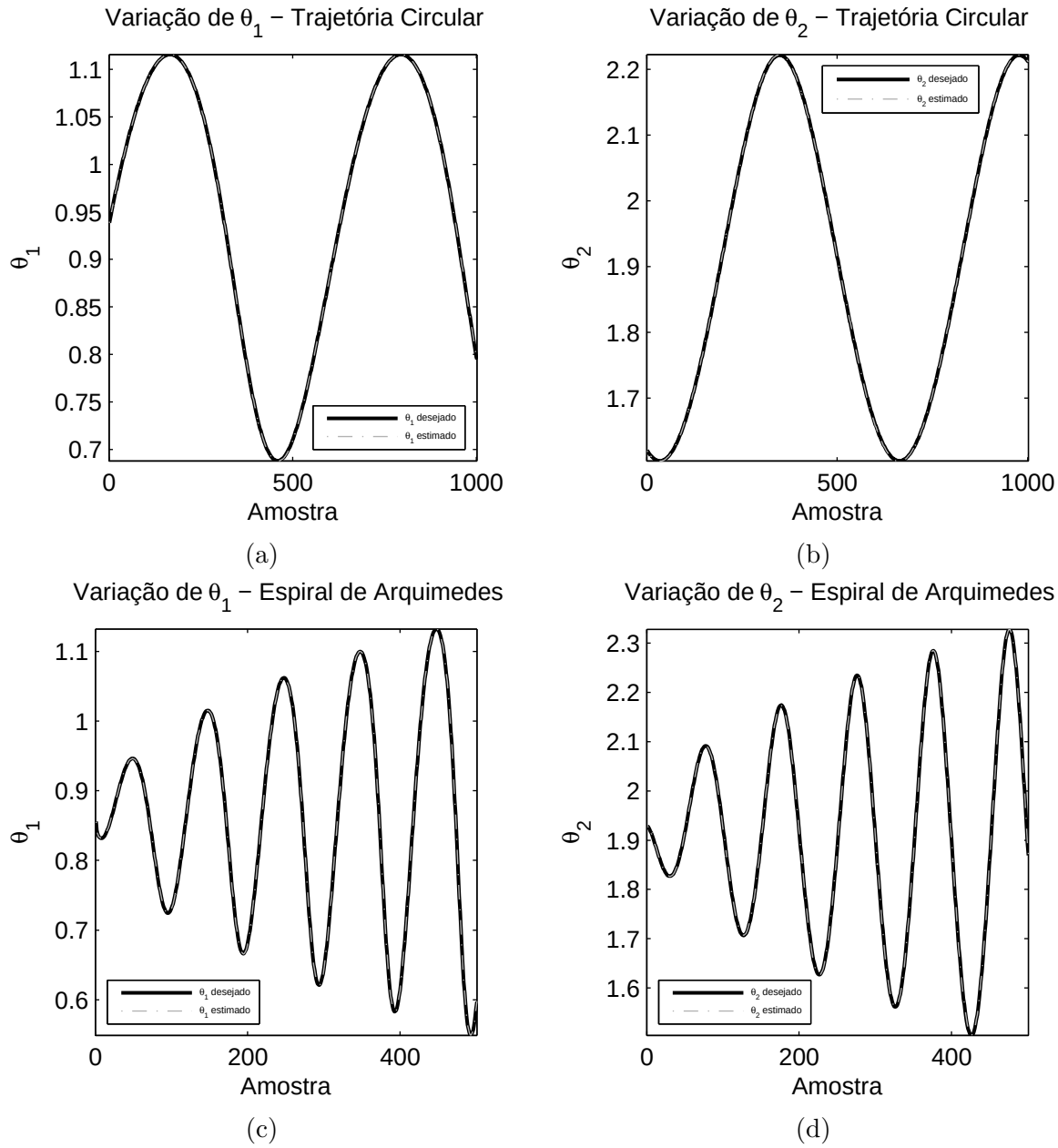


Figura 26 – **Robô Planar - LS-SVR**: (a) e (b) mostram a variação do ângulos de junta ao desenvolver uma trajetória circular. (c) e (d) mostram a variação do ângulos de junta ao desenvolver uma trajetória espiral.

pontos se deve ao processo de geração de dados que realizou uma amostragem do *task space* em passos fixos.

A Figura 29 exibe no plano cartesiano (em linhas tracejadas) as trajetórias desejadas, bem como as trajetórias estimadas pelos modelos LS-SVR e ANFIS (em linhas sólida). As trajetórias estimadas foram obtidas através da aplicação dos valores estimados de θ_1 e θ_2 nas expressões da cinemática direta do robô planar descritas nas Equações (2.10) e (2.11). Desta forma, obteve-se um conjunto de pontos $\hat{\mathbf{x}} = [\hat{p}_x \ \hat{p}_y]^T$ que foram exibidos juntamente como os pontos $\mathbf{x_d}$ desejados.

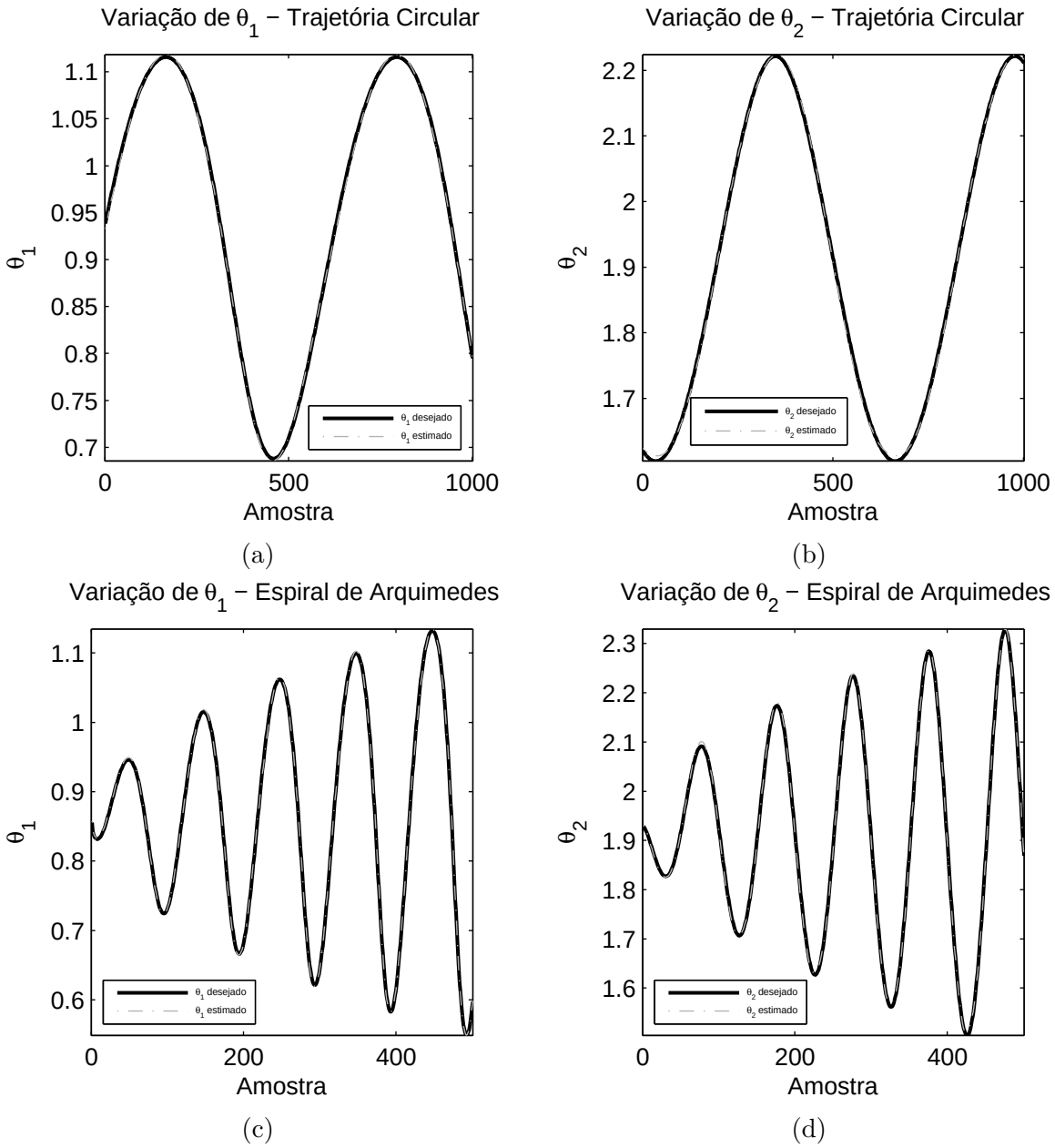


Figura 27 – **Robô Planar - ANFIS**: (a) e (b) mostram a variação do ângulos de junta ao desenvolver uma trajetória circular. (c) e (d) mostram a variação do ângulos de junta ao desenvolver uma trajetória espiral.

4.7.2 Manipulador PUMA 560

Assim como no caso do robô planar, três conjuntos de testes foram submetidos aos melhores modelos de regressão selecionados, a saber, LS-SVR e PG. O primeiro deles foi extraído do conjunto de dados principal particionado anteriormente. O segundo conjunto descreve uma trajetória circular no plano XY centrada no ponto $P_c = [0{,}5\ 0\ 0{,}5]^T$ e de raio $r = 0{,}125\ m$. O terceiro conjunto de dados descreve uma trajetória senoidal de amplitude $A = 0{,}1\ m$ e frequência $\omega = 30\ rad/s$. Estes conjuntos são compostos de 213, 100 e 28 padrões de teste, respectivamente.

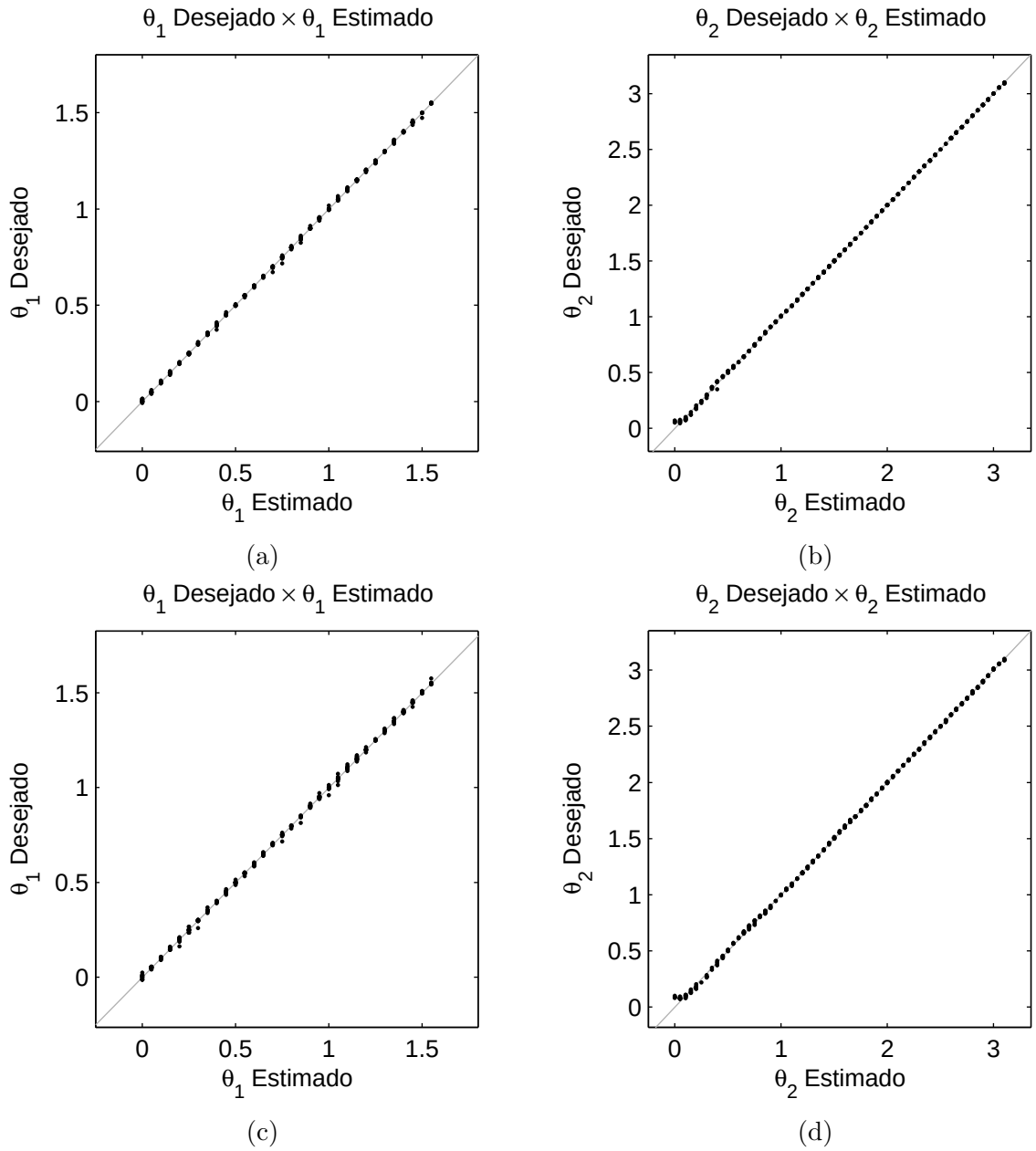


Figura 28 – **Robô Planar** - (a) e (b) mostram os gráficos dos valores desejados contra os valores estimados dos ângulos de junta para o modelo LS-SVR. As Figuras (c) e (d) mostram o mesmo gráfico para o modelo ANFIS.

A Tabela 9 exibe para os ângulos de junta anteriores ao punho o valor de $\xi(\theta_k)$ juntamente com a métrica de seleção. As linhas desta tabela foram ordenadas do melhor para o pior desempenho global.

Ao se analisar a Tabela 9 pode-se notar que os modelos LS-SVR e PG apresentaram o melhor desempenho do ponto de vista do espaço das juntas. Além do mais, realizou-se também a avaliação do desempenho destes modelos perante as amostras das trajetórias circular e senoidal. Uma métrica similar àquela utilizada na ordenação da Tabela 9 foi tomada, MS_{traj}. Esta métrica é obtida através da média geométrica dos valores das

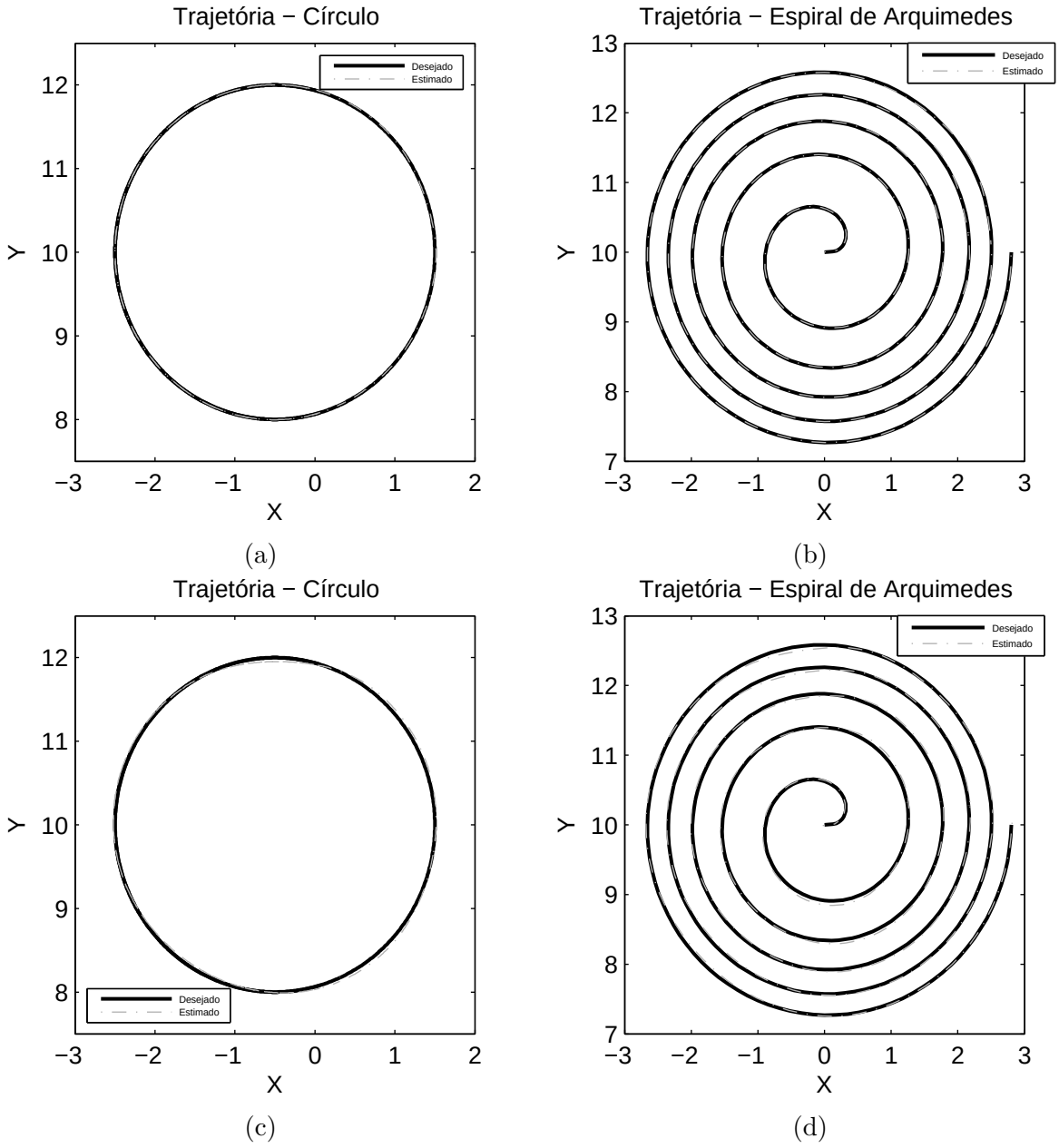


Figura 29 – **Robô Planar** - Resultado da estimação de trajetórias pelos os algoritmos LS-SVR (Figuras (a) e (b)) e ANFIS (Figuras (c) e (d)).

Algoritmo	$\xi(\theta_1)_{teste}$	$\xi(\theta_2)_{teste}$	$\xi(\theta_3)_{teste}$	MS_{teste}
LS-SVR	**2,53661482E-09**	**1,56195937E-05**	**7,28482109E-05**	**1,42379583E-06**
PG	**3,50392133E-09**	**2,02440794E-05**	**8,27697982E-05**	**1,80402032E-06**
ANFIS	9,43975706E-08	2,16546027E-05	7,83295021E-05	5,43015385E-06
ELM	4,24270583E-06	5,12996448E-05	2,04147741E-04	3,54188089E-05
MLM	1,86738900E-05	1,05869200E-04	3,62240100E-04	2,19545610E-04
LLM	2,20988555E-04	6,55363126E-04	2,10820816E-03	6,73372058E-04
MLP	1,16324422E-02	9,77364812E-03	2,82910070E-02	1,47613248E-02

Tabela 9 – **Manipulador PUMA 560** - Erros médios quadráticos de teste e métricas de seleção para as variáveis de saída dos modelos de regressão.

linhas de $\bar{d}_{círculo}$ e $\bar{d}_{senóide}$. A Tabela 10 traz um resumo dos resultados obtidos em termos das médias das distâncias ponto-a-ponto. Esta tabela foi ordenada seguindo os valores crescentes de MS_{traj}.

Algoritmo	$\bar{d}_{círculo}$	$\bar{d}_{senóide}$	MS_{traj}
LS-SVR	**2,96659220E-04**	**4,78389755E-04**	**3,76721026E-04**
PG	**5,74226329E-04**	**6,51319066E-04**	**6,11559119E-04**
MLM	7,86520236E-04	1,09637021E-03	9,28610444E-04
ANFIS	7,34449004E-04	1,76582056E-03	1,13881744E-03
ELM	2,13751864E-03	2,29137106E-03	2,21310830E-03
LLM	1,28162867E-02	1,06907601E-02	1,17053768E-02
MLP	5,06168876E-02	4,80449986E-02	4,93141794E-02

Tabela 10 – **Manipulador PUMA 560** - Médias das distâncias entre os pontos das trajetórias teóricas e estimadas, e as métricas de seleção correspondentes.

Ao se comparar as Tabelas 9 e 10 nota-se que nem todos os algoritmos apresentaram desempenhos similares tanto no espaço das juntas como no espaço cartesiano. Apesar disto, os dois melhores modelos mantiveram a correspondência em seus desempenhos em ambos os espaços.

Para uma melhor visualização do desempenho dos modelos foram traçados gráficos que exibisse conjuntamente as quantidades esperadas e as quantidades estimadas. As Figuras 30 e 31 exibem como variam os ângulos de junta para as trajetórias circular e senoidal, respectivamente.

Ao se observar os gráficos gráficos apresentados nas Figuras 30 e 31 mal percebe-se a diferença entre os valores desejados (curva sólida) e os valores estimados (curva tracejada).

Foi visto antes que outra maneira de visualizar a capacidade de generalização dos modelos é através de gráficos dos valores desejados contra os valores estimados. A Figura 32 exibe estes gráficos para as amostras de teste do conjunto principal. As retas $\theta_k = \hat{\theta}_k$ também foi exibida nestes gráficos para efeitos comparativos. As nuvens de pontos se concentram de forma bastante compacta ao redor das retas auxiliares indicando uma boa aproximação dos ângulos de junta.

Por fim, são exibidas na Figura 33 as trajetórias desejadas (linha sólida) juntamente com as trajetórias estimadas (linhas tracejadas) para os modelos LS-SVR e PG. Utilizou-se a mesma abordagem aplicada ao robô planar para gerar as trajetórias estimadas a partir dos ângulos estimados.

4.7.3 Manipulador Motoman HP6

Para este manipulador também foram aplicados três conjuntos de teste. O primeiro deles é composto de 203 amostras remanescentes do conjunto de dados principal. O segundo é formado de 100 pontos amostrados de um lugar geométrico circular de centro

$P_c = [0{,}65\ 0\ 0{,}65]^T$ e raior $r = 0{,}125\ m$. Já o terceiro conjunto desempenha uma trajetória senoidal de amplitude $A = 0{,}1\ m$ e frequência $\omega = 30\ rad/s$ composta de 28 pontos.

A Tabela 11 exibe para os ângulos de junta anteriores ao punho, o valor de $\xi(\theta_k)$, $k = 1, \ldots, n_{DOF}^{(w)}$), juntamente com a métrica de seleção, para os padrões de teste do conjunto principal. As linhas desta tabela foram ordenadas do melhor para o pior desempenho global

Algoritmo	$\xi(\theta_1)_{teste}$	$\xi(\theta_2)_{teste}$	$\xi(\theta_3)_{teste}$	MS_{teste}
LS-SVR	**1,76394847E-13**	**3,87523660E-10**	**1,74668429E-09**	**4,92416780E-11**
PG	**4,15848331E-13**	**4,76876517E-09**	**1,73845954E-08**	**3,25463154E-10**
ELM	5,59387960E-09	9,49045814E-08	3,22652239E-07	5,55364768E-08
ANFIS	1,73068206E-08	4,02234592E-06	1,10754650E-06	4,25617581E-07
MLM	4,01895046E-06	1,43546103E-05	4,39825684E-05	1,36393879E-05
LLM	1,84548071E-05	6,06491476E-05	1,13629112E-04	5,02891862E-05
MLP	2,25254518E-03	2,07549943E-03	4,85261706E-03	2,83089688E-03

Tabela 11 – **Manipulador Motoman HP6** - Erros médios quadráticos de teste e métricas de seleção para as variáveis de saída dos modelos de regressão.

Ao se observar a Tabela 11 pode-se notar que os algoritmos LS-SVR e PG apresentaram o melhor desempenho. Além disso, verificou-se o desempenho dos modelos para as amostras das trajetórias auxiliares. A Tabela 12 resume os resultados obtidos em termos das médias das distâncias ponto a ponto no espaço cartesiano

Algoritmo	$\bar{d}_{círculo}$	$\bar{d}_{senóide}$	MS_{traj}
PG	**2,66256673E-06**	**2,73889913E-06**	**2,70046324E-06**
LS-SVR	**2,56501913E-06**	**4,01236213E-06**	**3,20808130E-06**
ELM	1,59528264E-04	2,32784244E-04	1,92706166E-04
ANFIS	2,23678397E-04	3,81685173E-04	2,92189541E-04
MLM	5,27705310E-04	4,19095318E-04	4,70275265E-04
LLM	4,99184322E-03	3,89281397E-03	4,40821018E-03
MLP	3,25065752E-02	2,67840701E-02	2,95069210E-02

Tabela 12 – **Manipulador Motoman HP6** - Médias das distâncias entre os pontos das trajetórias teóricas e estimadas, e as métricas de seleção correspondentes.

Uma análise das Tabelas 11 e 12 mostra mais uma vez que nem todos os algoritmos mantém o mesmo desempenho do espaço das juntas no espaço cartesiano, situação esta que é confirmada pela ordenação das duas tabelas. Apesar deste fato, os dois melhores modelos permaneceram nas duas primeiras posições da tabela.

Apenas como uma forma visual de ilustrar o desempenho dos dois melhores modelos de regressão, exibe-se nas Figuras 34 e 35 os gráficos conjuntos dos valores desejados (em linha sólida) e estimados (em linha tracejada) executados pelos três primeiros ângulos de junta durante as trajetórias de teste

A partir das Figuras 34 e 35 nota-se uma sobreposição acentuada dos gráficos dos valores estimados sobre os gráficos dos valores desejados, indicando aproximações bastantes fiéis para as trajetórias simuladas.

Para completar a visualização da capacidade de generalização dos modelos LS-SVR e PG, exibe-se na Figura 36 os pares de pontos formados pelas coordenadas desejadas e estimadas, respectivamente, bem como as retas $\theta_k = \hat{\theta}_k$, para $k = 1, \ldots, n_{DOF}^{(w)}$, que facilitam a verificação da qualidade da aproximação dos modelos.

Observa-se nos gráficos da Figura 36 que os pontos estão bem concentrados em torno da reta de referência indicando boas estimações das funções de cinemática inversa.

A Figura 37 exibe no espaço cartesiano (em linhas sólidas) as trajetórias que se desejou aproximar bem como as trajetórias estimadas pelos modelos LS-SVR e PG (em linhas tracejadas). As trajetórias estimadas foram obtidas através da aplicação dos valores estimados dos ângulos de junta nas equações da cinemática direta do manipulador Motoman HP6 descritas na Seção 2.3.5.4. Desta forma, obteve-se um conjunto de pontos $\hat{\mathbf{x}} = [\hat{p}_x \ \hat{p}_y \ \hat{p}_z]^T$ que foram exibidos juntamente como os pontos $\mathbf{x_d}$ desejados.

4.8 Conclusão

Este capítulo apresentou a metodologia adotada durante o desenvolvimento das simulações e no processo de aquisição dos resultados, bem como foram reunidos os resultados relativos às fases de seleção e testes dos modelos adotados. Nesta última fase, também foi definida uma métrica que possibilitou a escolha dos dois melhores modelos para cada manipulador. Esta escolha norteou a apresentação de trajetórias conhecidas para estes modelos de forma que os seus desempenhos espaço das juntas e no espaço cartesiano pudesse ser comparado. O próximo capítulo realiza uma análise dos resíduos gerados pelos melhores modelos a fim de comparar seus desempenhos estatisticamente bem como uma avaliação das funções de autocorrelação gerados por tais resíduos.

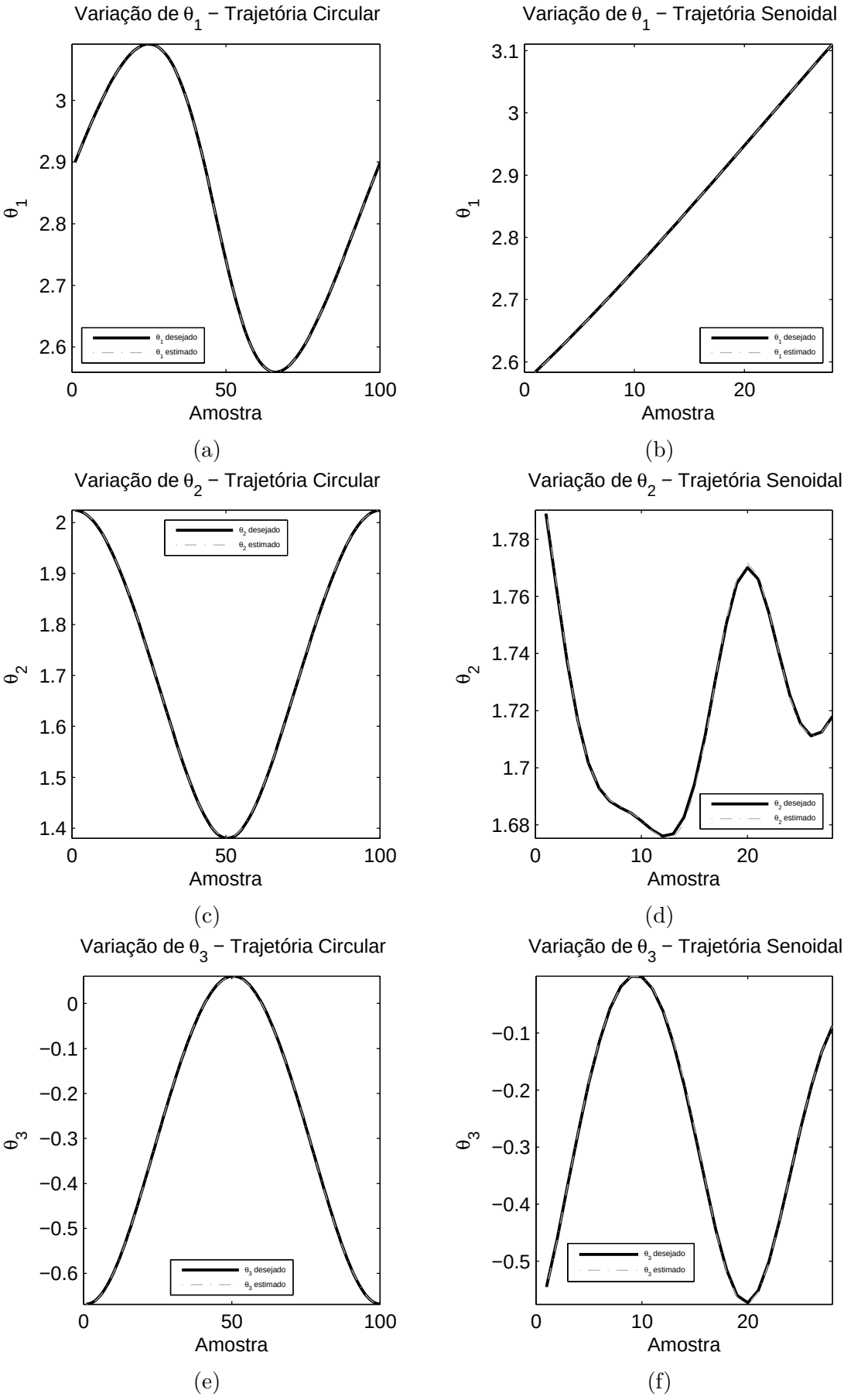


Figura 30 – **Manipulador PUMA 560 - LS-SVR**: (a), (c) e (e) mostram a variação do ângulos de junta ao desenvolver uma trajetória circular. (b), (d) e (f) mostram a variação do ângulos de junta ao desenvolver uma trajetória senoidal.

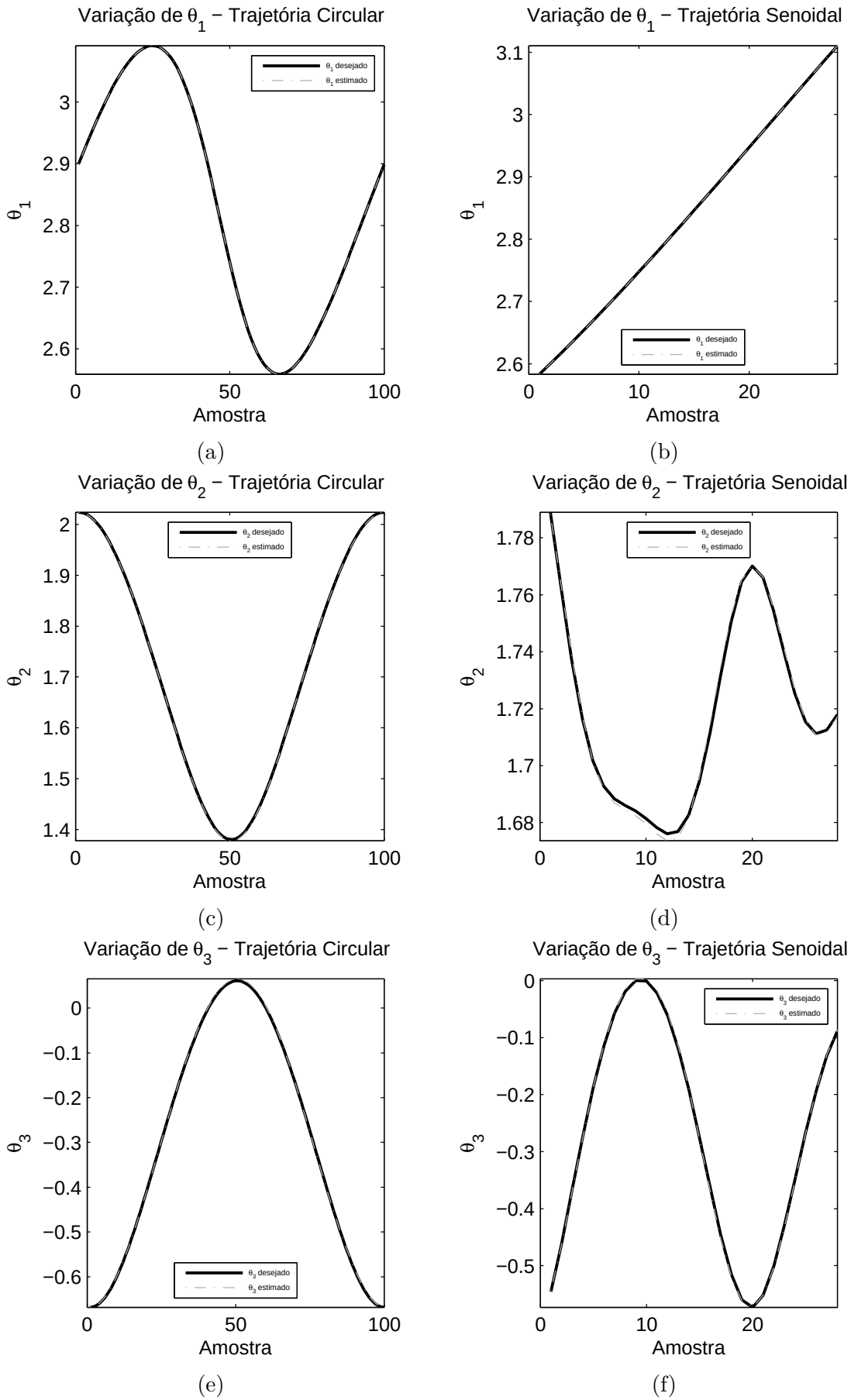


Figura 31 – **Manipulador PUMA 560 - PG**: (a), (c) e (e) mostram a variação do ângulos de junta ao desenvolver uma trajetória circular. (b), (d) e (f) mostram a variação do ângulos de junta ao desenvolver uma trajetória senoidal.

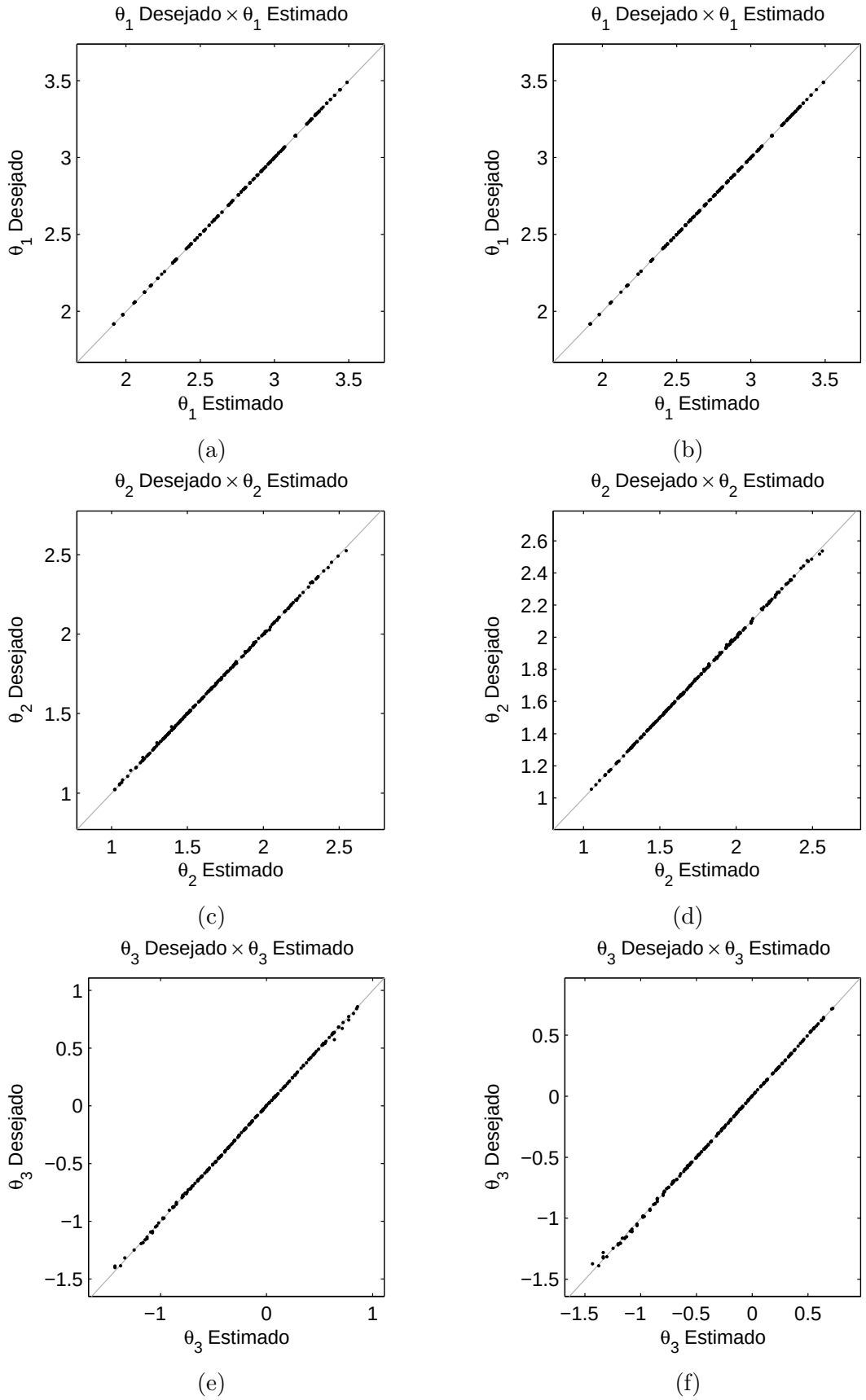


Figura 32 – **Manipulador PUMA 560** - As Figuras (a), (c) e (e) mostram os gráficos dos valores desejados contra os valores estimados dos ângulos de junta para o modelo LS-SVR. As Figuras (b), (d) e (f) mostram o mesmo gráfico para o modelo de regressão baseado em PG.

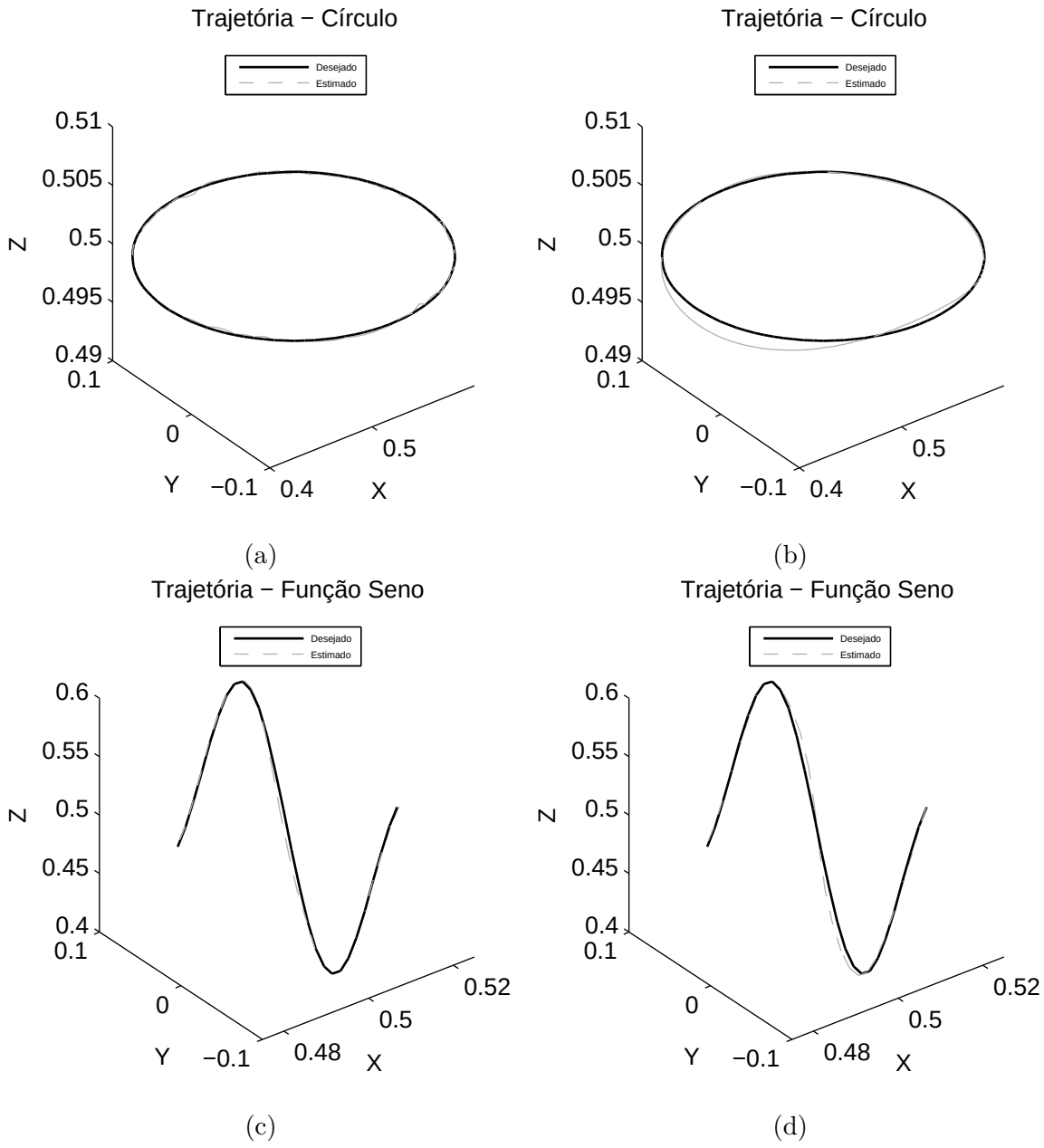


Figura 33 – **Manipulador PUMA 560** - Trajetórias teóricas e estimadas pelos modelos LS-SVR (Figuras (a) e (c)) e PG (Figuras (b) e (d)).

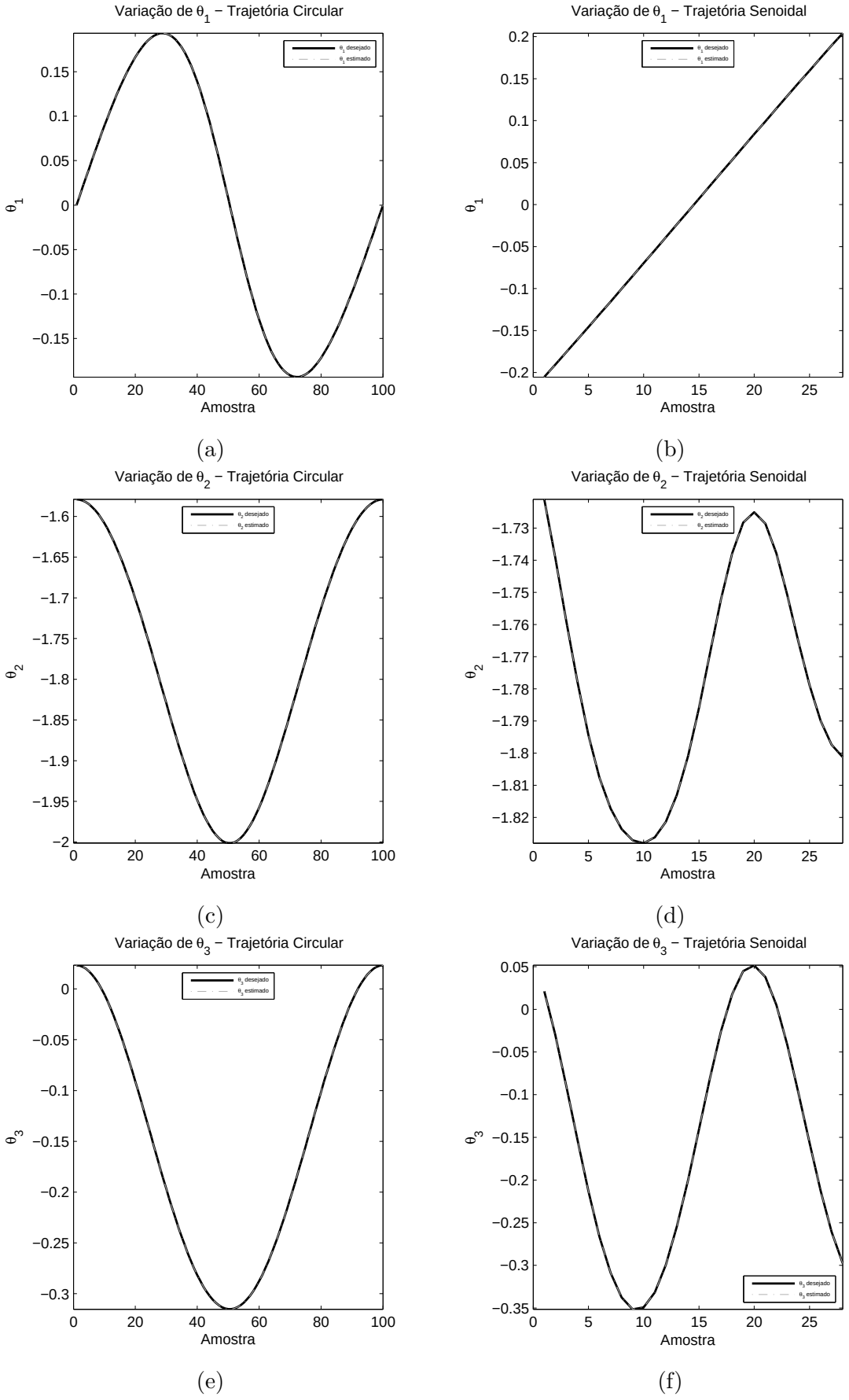


Figura 34 – **Manipulador Motoman HP6 - LS-SVR**: (a), (c) e (e) mostram a variação do ângulos de junta ao desenvolver uma trajetória circular. (b), (d) e (f) mostram a variação do ângulos de junta ao desenvolver uma trajetória senoidal.

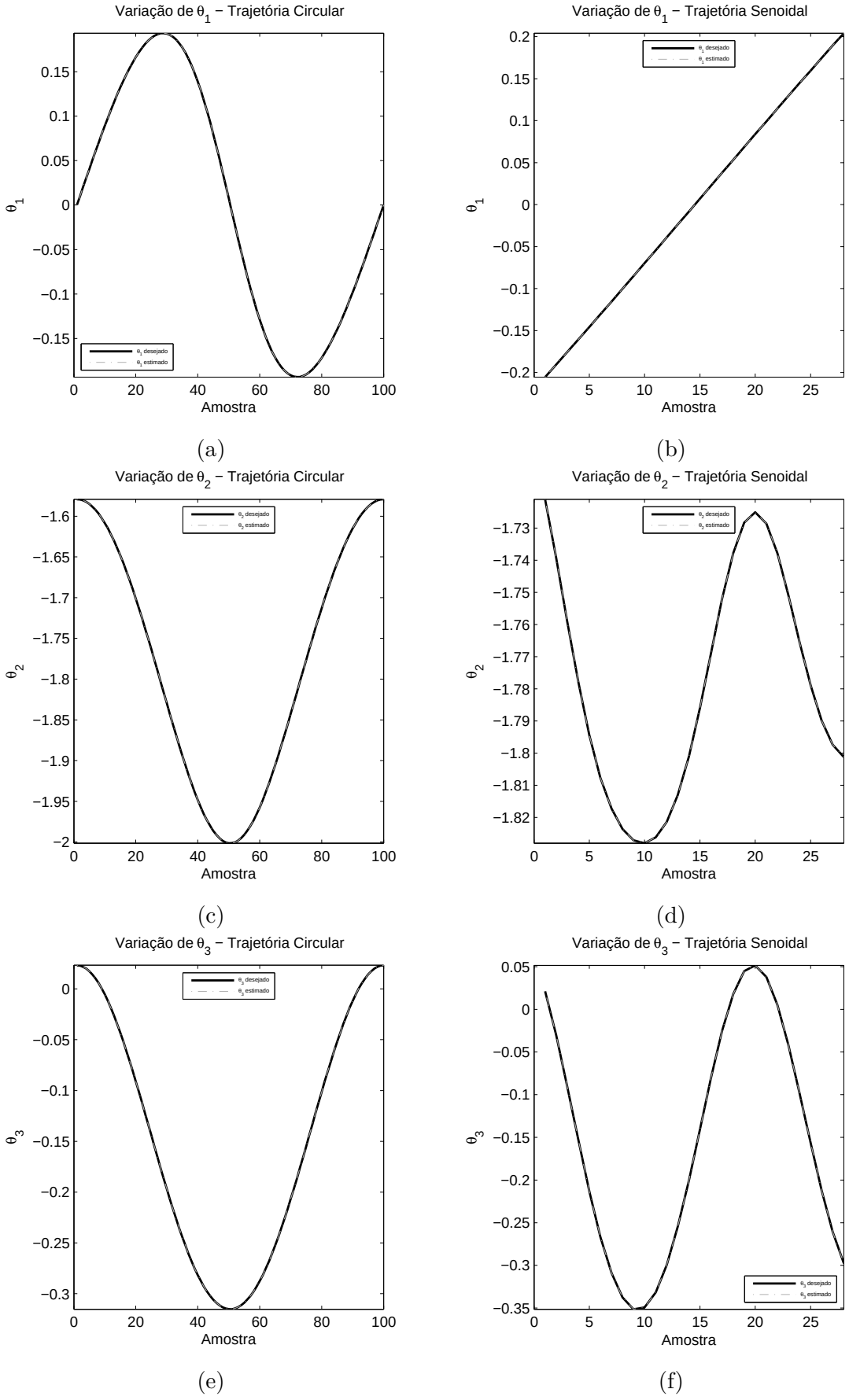


Figura 35 – **Manipulador Motoman HP6 - PG**: (a), (c) e (e) mostram a variação do ângulos de junta ao desenvolver uma trajetória circular. (b), (d) e (f) mostram a variação do ângulos de junta ao desenvolver uma trajetória senoidal.

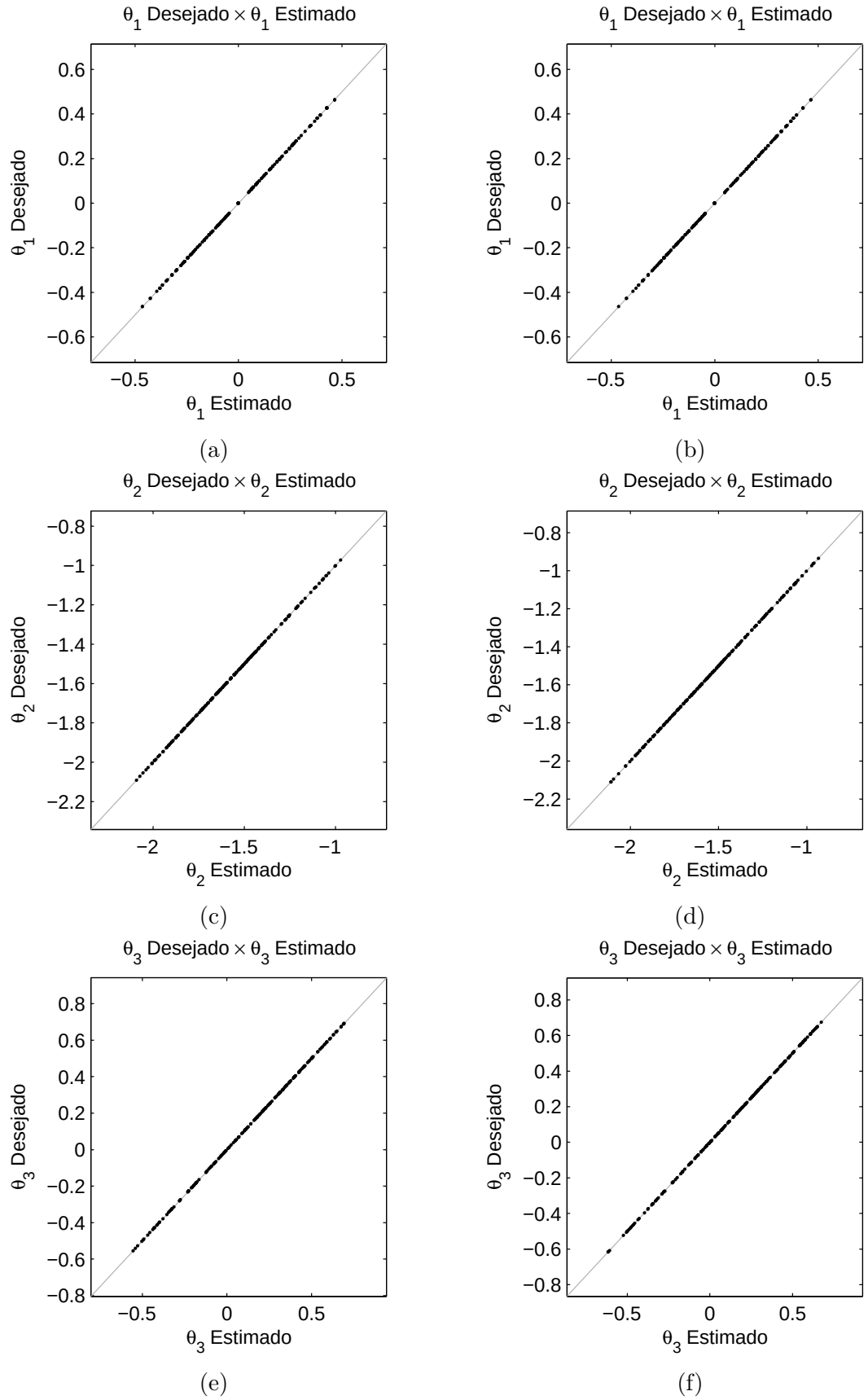


Figura 36 – **Manipulador Motoman HP6** - As Figuras (a), (c) e (e) mostram os gráficos dos valores desejados contra os valores estimados dos ângulos de junta para o modelo LS-SVR. As Figuras (b), (d) e (f) mostram o mesmo gráfico para o modelo de regressão baseado em PG.

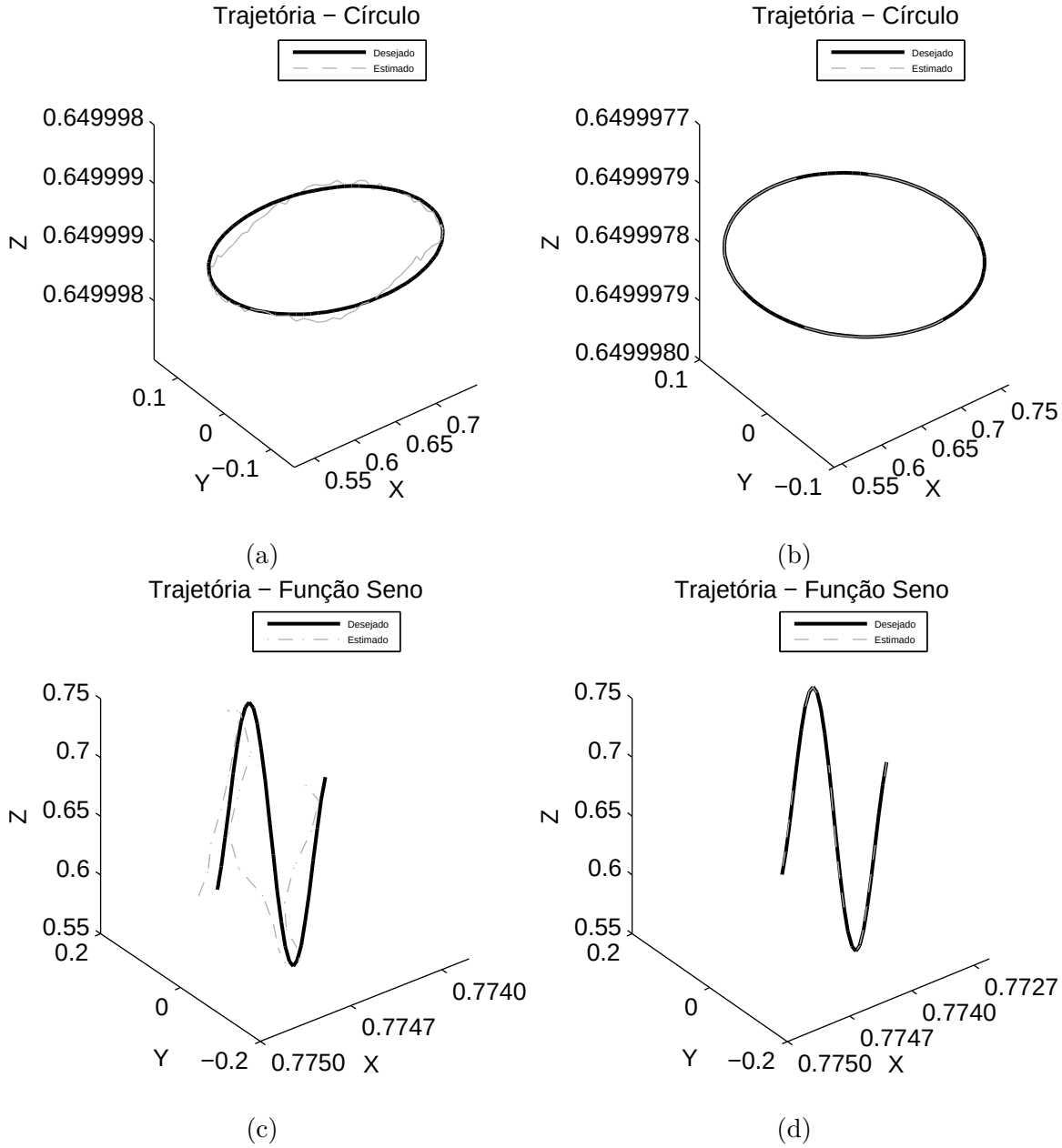


Figura 37 – **Manipulador Motoman HP6** - Trajetórias teóricas e estimadas pelos modelos LS-SVR (Figuras (a) e (c)) e PG (Figuras (b) e (d)).

5 Análise dos Resíduos

5.1 Introdução

Este capítulo apresenta os resultados dos testes de hipóteses aplicados aos resíduos gerados pelos melhores modelos de regressão que foram escolhidos no capítulo anterior. Estes testes têm como objetivo comparar estatisticamente o desempenho destes algoritmos. Além do mais, serão apresentadas as funções de autocorrelação dos resíduos para fins de verificar o quão bem os modelos apreenderam as relações entrada/saída fornecidas pelos dados de treinamento.

5.2 Testes de Hipóteses

Muitas vezes pretende-se realizar afirmações sobre as características de uma determinada população. Em alguns casos, a comprovação dessas afirmações de maneira direta é inviável devido ao fato de existirem diversos elementos na população ou o processo comprobatório ser destrutivo. Por exemplo, não se pode verificar o percentual de defeitos de todas as lâmpadas de um lote pois o procedimento consiste em um ensaio destrutivo. Para solucionar este problema, adota-se uma pequena parcela da população (*amostra*) e espera-se que as análises estatísticas realizadas sobre ela correspondam às características de toda a população. Este processo de generalizar a população a partir das amostras é chamado de *Inferência Estatística* (BUSSAB; MORETTIN, 2010). Outros tipos de problemas tratados pela inferência estatística são os *Testes de Hipóteses*. Grosso modo, um teste de hipótese é uma afirmação sobre a população que é testada contra uma amostra, possibilitando sua confirmação ou refutação baseando-se em parâmetros populacionais.

Testes de hipóteses podem ser classificados de acordo com a quantidade de amostras utilizadas. Nos casos mais simples as análises estatísticas são realizadas apenas sobre uma amostra buscando-se, por exemplo, identificar se a amostra em questão possui determinada média, variância ou advém de uma distribuição de probabilidade específica. Já os testes de amostra dupla realizam indagações sobre a similaridade de uma amostra contra a outra. Por exemplo, procura-se identificar se duas amostras advém da mesma distribuição de probabilidade ou se elas são ambas amostras de uma distribuição normal de mesma média.

Neste trabalho, foram realizados testes de hipóteses apenas para os dois modelos que obtiveram melhor desempenho global através da avaliação da métrica de seleção. O teste de hipóteses adotado foi o teste de *Kolmogorov-Smirnov* de duas amostras, que será detalhado adiante.

5.2.1 Teste de *Kolmogorov-Smirnov*

Por meio do teste de hipóteses de Kolmogorov-Smirnov (KS) (JR, 1951) será verificado se as amostras de resíduo geradas pelos modelos analisados são provenientes de uma mesma distribuição de probabilidade desconhecida. Devido ao fato de o teste KS não assumir nenhuma distribuição específica para os resíduos, ele é classificado com teste *não paramétrico*. A resposta positiva corresponde a aceitação da hipótese nula (H_0), caso contrário ocorre a rejeição. Considerando duas amostras A_1 e A_2, tem-se:

$$\begin{cases} H_0 : \text{As amostras } A_1 \text{ e } A_2 \text{ advém da mesma distribuição de probabilidade.} \\ H_1 : \text{As amostras } A_1 \text{ e } A_2 \text{ advém de distribuições de probabilidade diferentes.} \end{cases} \tag{5.1}$$

O teste KS quantifica a distância entre as Funções de Distribuição Acumuladas (FDA) empíricas de duas sequências de resíduos e através disso busca averiguar se há ou não diferenças significativas do ponto de vista estatístico entre desempenhos dos os diferentes modelos de regressão. Sejam $\hat{F}_1(e)$ e $\hat{F}_2(e)$ as FDAs empíricas das duas amostras comparadas, calcula-se a máxima diferença absoluta entre estas funções:

$$D^* = \max_e \Big(|\hat{F}_1(e) - \hat{F}_2(e)| \Big) \tag{5.2}$$

Através do valor calculado de D^*, juntamente com o nível da significância adotado no teste (5%), obtém-se um outro valor denominado valor-p. O valor-p é definido como a probabilidade de se obter valores iguais ou maiores do que aquele fixado para uma estatística utilizada na hipótese nula (H_0), neste caso, a máxima diferença absoluta entre as FDAs empíricas. Para valores-p maiores que o nível de significância adotado no teste KS, a hipótese nula é aceita, caso contrário, é rejeitada.

A Tabela 13 apresenta os resultados dos testes KS para os resíduos de estimação dos três manipuladores testados.

Para os resíduos dos ângulos de junta do robô planar, todos os testes KS rejeitaram a hipótese nula, ou seja, os resíduos gerados pelos algoritmos LS-SVR e PG não seguem a mesma distribuição de probabilidade. Logo, é possível inferir que estes modelos têm desempenhos estatisticamente distintos. Em outras palavras, pode-se afirmar que a diferença entre os desempenhos dos dois modelos é significativa.

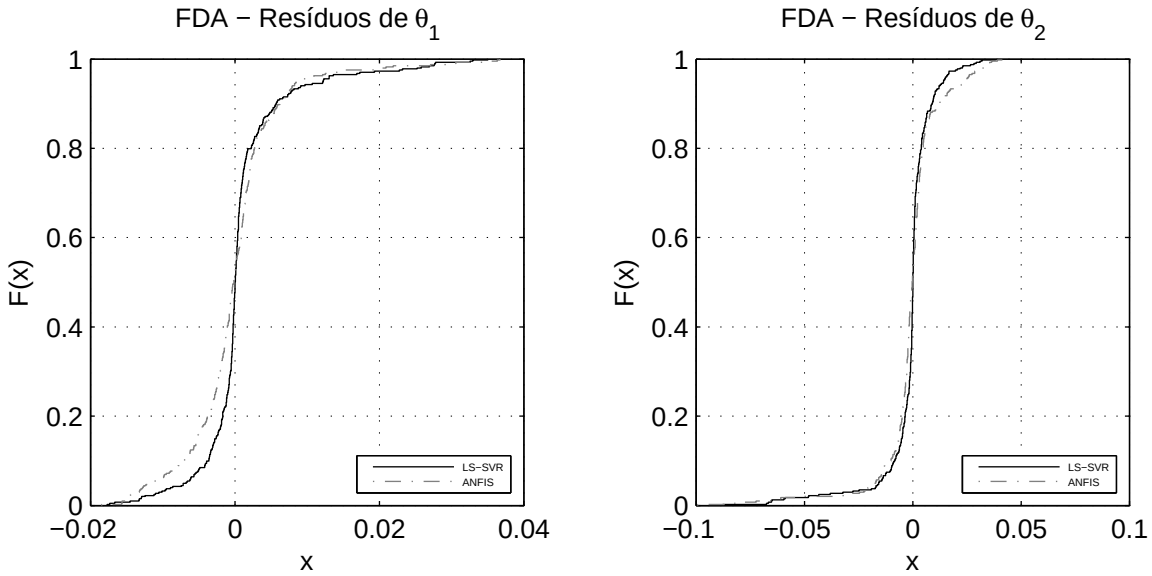
Já para os manipuladores PUMA 560 e Motoman HP6, alguns conjuntos de resíduos apresentaram equivalência estatística. Para estes casos de equivalência, recomenda-se ao usuário escolher o algoritmo que apresente menor $\xi(\theta_k)_{teste}$.

Uma forma bastante útil de visualização dos testes acima é a exibição conjunta dos gráficos das FDAs empíricas dos modelos comparados pelos testes KS. Através destes

Robô Planar - **LS-SVR** × **ANFIS**		
Ângulo de Junta	Hipótese Nula	Valor-p
θ_1	Rejeitada	3,3258E-05
θ_2	Rejeitada	6,4144E-05
PUMA 560 - **LS-SVR** × **PG**		
θ_1	Rejeitada	2,6300E-02
θ_2	Aceita	2,3890E-01
θ_3	Aceita	2,9060E-01
θ_4	Rejeitada	1,5563E-30
θ_5	Aceita	7,7200E-02
θ_6	Aceita	9,8990E-01
Motoman HP6 - **LS-SVR** × **PG**		
θ_1	Rejeitada	1,3624E-06
θ_2	Aceita	6,5600E-02
θ_3	Aceita	4,5830E-01
θ_4	Rejeitada	1,3929E-46
θ_5	Rejeitada	1,1300E-02
θ_6	Aceita	1,0880E-01

Tabela 13 – Resultados dos testes de Kolmogorov-Smirnov para os resíduos gerados pelos melhores modelos de cada manipulador.

gráficos torna-se possível visualizar as diferenças entre cada par de distribuições bem como as máximas distâncias entre elas. As Figuras 38 e 39 trazem estes gráficos para os ângulos de junta anteriores ao pulso dos manipuladores robóticos.



(a) (b)

Figura 38 – **Robô Planar** - FDAs empíricas dos resíduos gerados pelos modelos LS-SVR e ANFIS para os ângulos (a) θ_1 e (b) θ_2.

5.3 Análise de Correlação dos Resíduos

Grosso modo, pode-se enxergar os resíduos de estimação como um subproduto não capturado pelos modelos regressão. Para modelos bem ajustados aos dados espera-se que os resíduos sejam descorrelacionados, ou seja, ao se exibir os gráficos da função de autocorrelação (FAC) para os resíduos espera-se que eles apresentem valores não nulos apenas para o primeiro *lag*, enquanto os outros valores apresentem amplitudes que residem entre os limiares de confiança estabelecidos em 95%. Dentro deste intervalo de confiança as estimativas das funções de autocorrelação são consideradas estatisticamente nulas. Matematicamente, a FAC é definida como

$$\Phi_{ee}(\tau) = E\{e(t-\tau)e(t)\} = \delta(\tau) = \begin{cases} 1, & \text{se } \tau = 0. \\ 0, & \text{caso contrário.} \end{cases} \tag{5.3}$$

onde $E(\cdot)$ é o operador valor esperado e $\delta(\tau)$ é a função delta de Dirac. As seções seguintes apresentam para cada manipulador os gráficos da FAC dos resíduos dos dois melhores modelos.

5.3.1 Resultados da FAC: Robô Planar

Para o robô planar, a análise de autocorrelação dos resíduos do ângulo θ_1 para os algoritmos LS-SVR e ANFIS são mostrados na Figura 40. Por questões de simplicidade, apenas os gráficos para ângulo θ_1 foram apresentados. Ao analisar todos os gráficos gerados pelas simulações constatou-se que os dois melhores modelos são estatisticamente válidos visto que apenas o *lag* zero apresentou-se fora do intervalo de confiança e com valor unitário. Alguns poucos *lags* apresentaram amplitudes fora do intervalo de confiança de 95%, mas este comportamento parece ser fruto do acaso.

5.3.2 Resultados da FAC: Manipulador PUMA 560

De forma similar ao robô planar, adotou-se o critério da Equação (5.3) para validar estatisticamente os dois melhores modelos (LS-SVR e PG) para o manipulador PUMA 560. A Figura 40 ilustra a análise de autocorrelação para os resíduos do ângulo de junta θ_1. Através dos gráficos gerados pelas simulações, conclui-se novamente que os dois melhores modelos novamente são estatisticamente válidos. Alguns poucos *lags* apresentaram amplitudes fora do intervalo de confiança de 95%, mas este comportamento parece ser fruto do acaso.

5.3.3 Resultados da FAC: Manipulador Motoman HP6

Para o manipulador Motoman HP6 procedeu-se de forma similar aos robôs planar e PUMA 560. A análise de autocorrelação dos resíduos do ângulo θ_1 para os algoritmos LS-SVR e PG são mostrados na Figura 40. Adotando-se o mesmo critério anterior, conclui-se, através de todos os gráficos gerados pelas simulações, que os dois melhores modelos também são estatisticamente válidos. Alguns poucos *lags* apresentaram amplitudes fora do intervalo de confiança de 95%, mas este comportamento parece ser fruto do acaso.

5.4 Conclusão

Este capítulo apresentou os resultados dos testes de hipóteses de Kolmogorov-Smirnov aplicados às amostras dos resíduos gerados pelos dois melhores modelos de regressão. Foram exibidos gráficos das FDAs empíricas a fim de evidenciar a máxima distância entre as distribuições. Além do mais, foram exibidos também os gráficos de autocorrelação dos resíduos com o objetivo de validar estatisticamente os modelos.

No próximo capítulo são apresentadas as principais conclusões desta dissertação e também são sugeridos temas para pesquisas futuras na área regressão não-linear aplicada ao aprendizado da cinemática inversa de robôs manipuladores.

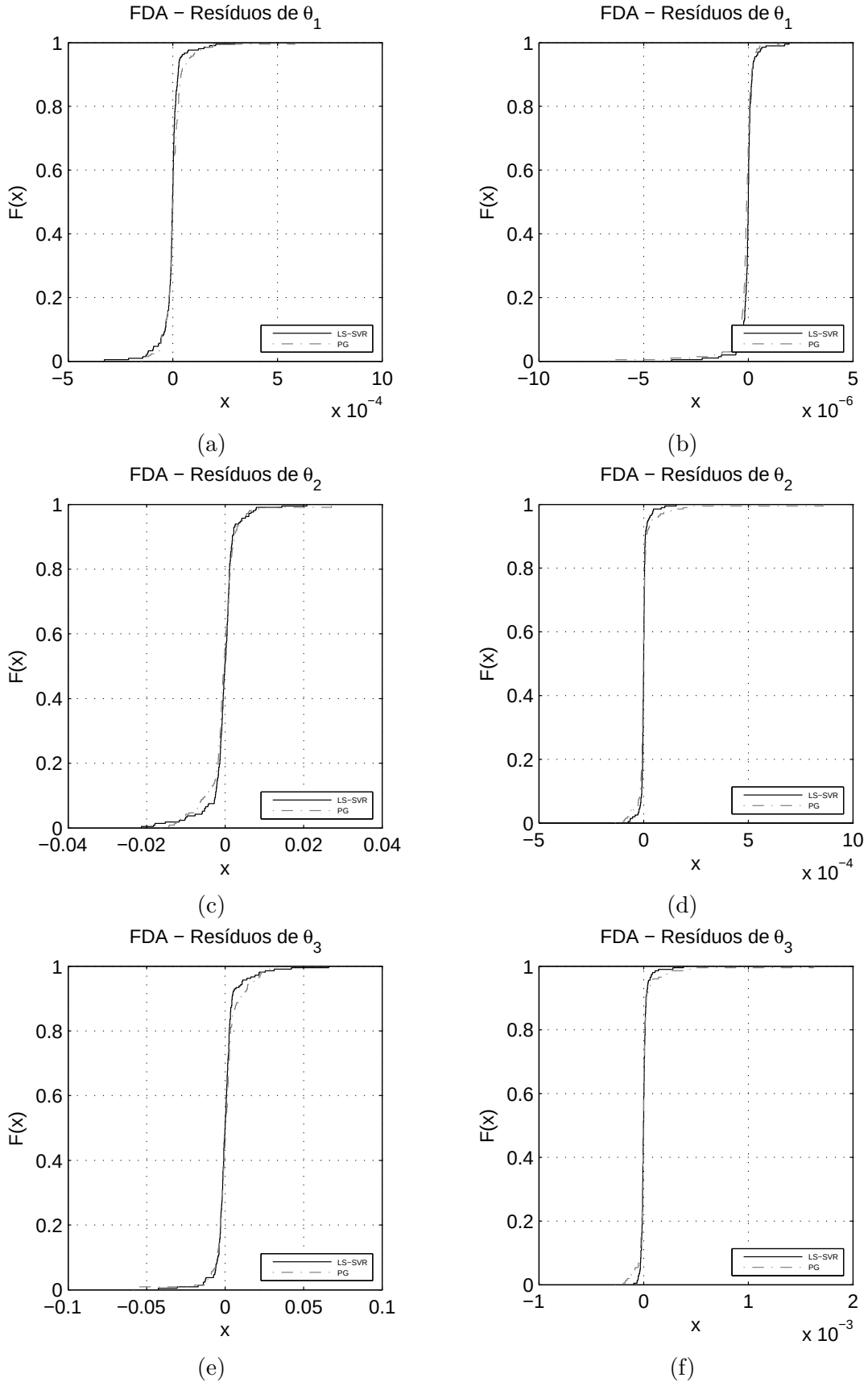
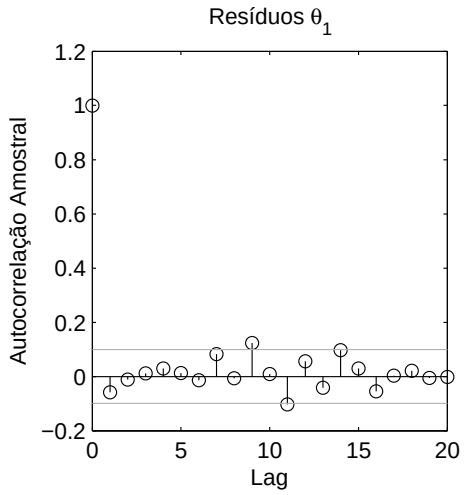
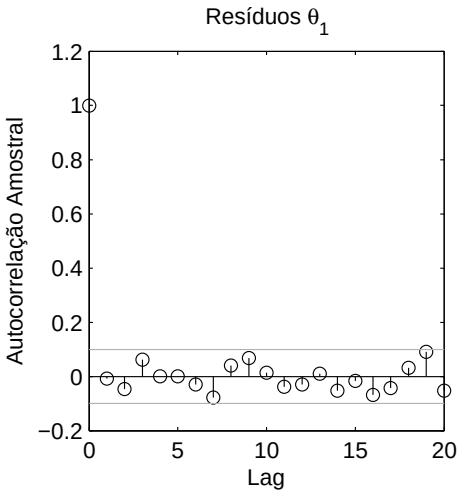


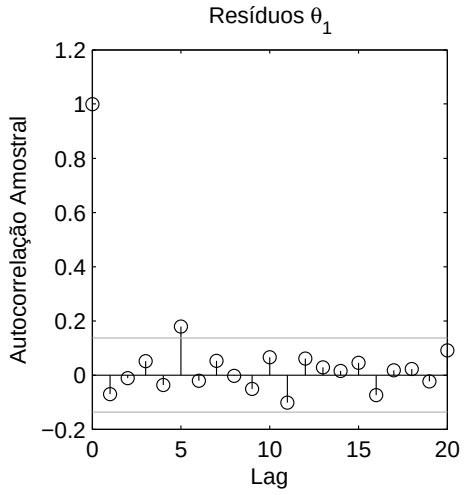
Figura 39 – FDAs empíricas dos resíduos gerados pelos modelos LS-SVR e PG para os três primeiros ângulos de junta para os manipuladores PUMA 560 - Figuras (a),(c) e (e) - e para o Manipulador Motoman HP6 - Figuras (b), (d) e (f).



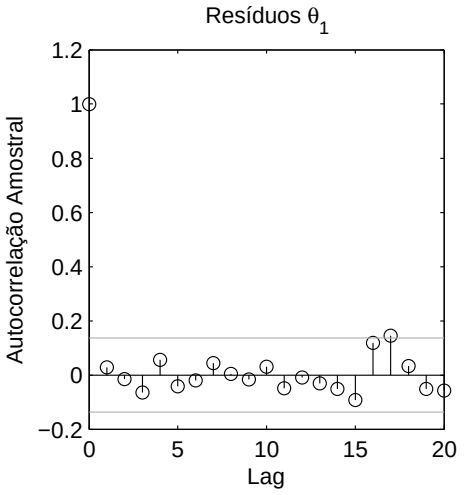
(a) Robô Planar - LS-SVR



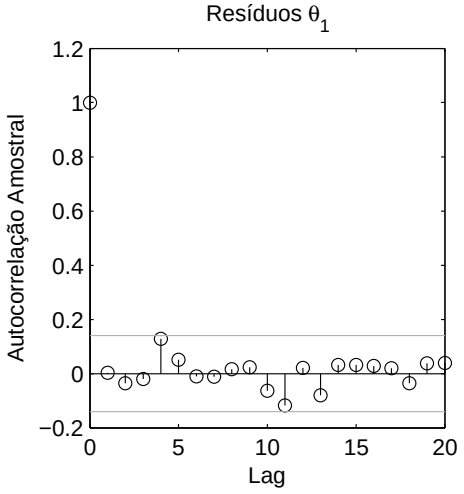
(b) Robô Planar - ANFIS



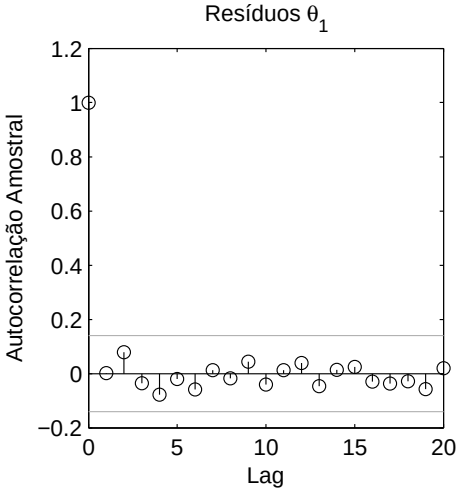
(c) Manipulador PUMA 560 - LS-SVR



(d) Manipulador PUMA 560 - PG



(e) Manipulador Motoman HP6 - LS-SVR



(f) Manipulador Motoman HP6 - PG

Figura 40 – Gráfico de autocorrelação dos resíduos de θ_1 para os robôs planar (Figuras (a) LS-SVR e (b) ANFIS) , PUMA 560 (Figuras (c) LS-SVR e (d) PG) e Motoman HP6 (Figuras (e) LS-SVR e (f) PG)

6 Conclusões

6.1 Introdução

Nesta dissertação, foram realizadas simulações que objetivaram o estudo comparativo de modelos de regressão de diversas classes aplicados à tarefa de aprendizado da cinemática inversa de manipuladores. Tais classes compreendem as Redes Neurais Artificiais (MLP e ELM), Regressão Baseada em Métodos de Kernel (LS-SVR e PG), Modelos Lineares Locais (LLM), Sistemas Neuro-Fuzzy (ANFIS) e Regressão Baseada em Distâncias (MLM).

No Capítulo 4, foram apresentados os passos que constituíram a metodologia adotada nas simulações e aquisição dos resultados que permitiram o estudo comparativo dos modelos de regressão. Esta metodologia tem como um dos fundamentos a utilização do erro quadrático médio calculado sobre as amostras do conjunto de treinamento e de teste. Por meio dos valores EQMs de treinamento, foi possível selecionar os parâmetros dos modelos de regressão. Já os EQMs de teste deram origem a uma métrica de seleção que possibilitou a escolha dos dois melhores modelos para cada um dos robôs manipuladores. Esta métrica forneceu uma noção de desempenho global pois incorporava os EQMs dos ângulos mais relevantes para o posicionamento do efetuador. Desta forma, a análise aplicada fez uma comparação justa entre os modelos, não beneficiando nenhum algoritmo em particular.

Conforme a métrica de seleção adotada, os modelos LS-SVR e ANFIS apresentaram melhores desempenhos globais para os dados do robô planar, enquanto os modelos LS-SVR e PG foram os que apresentaram os menores valores de erro para os dados dos Manipuladores PUMA 560 e Motoman HP6. Vale ressaltar que o modelo PG também apresentou excelentes resultados para os dados do robô planar, aparecendo na terceira posição com respeito a métrica de seleção de teste.

Foram aplicados aos modelos de regressão algumas trajetórias conhecidas dentro do espaço de trabalho dos robôs. Por meio destras trajetórias, procurou-se observar se haveria uma correspondência direta entre os desempenhos no espaço cartesiano e no espaço das juntas. Observou-se que a ordenação dos algoritmos baseada na métrica de seleção não correspondeu a ordenação baseada na distância média observada para todos os pontos das trajetória aplicadas. Logo, para as simulações realizadas, um bom desempenho global no espaço das juntas não necessariamente corresponde a um bom desempenho do espaço cartesiano e vice-versa.

Testes de hipóteses foram realizados sobre os resíduos gerados pelos melhores

modelos de regressão para cada manipulador. Para o caso do robô planar, as hipóteses nulas foram rejeitadas para todos os ângulos de junta estimado, indicando que os desempenhos dos modelos LS-SVR e ANFIS são estatisticamente distintos. Já para os manipuladores PUMA 560 e Motoman HP6 os testes de hipóteses apresentaram resultados mistos, indicando que os modelos LS-SVR e PG são equivalentes apenas para um conjunto de ângulos. Nos casos em que a equivalência estatística ocorre recomenda-se escolher o modelo que apresentou menor erro quadrático médio de teste ($\xi(\theta_k)_{teste}$).

Ao se analisar os gráficos das funções de autocorrelação dos resíduos gerados pelos modelos de regressão, notou-se que todos eles apresentaram validade estatística visto que apenas o *lag* zero apresentou-se fora do intervalo de confiança estipulado de 95%.

A seguir, apresenta-se um resumo das principais contribuições desta tese e discutem-se as conclusões que se pode tirar delas.

6.2 Resumo das Contribuições da Dissertação

De modo mais específico as principais contribuições da tese foram as seguintes:

1. Oferecer à comunidade científica, por meio de uma ampla revisão bibliográfica sobre o tema, uma panorama geral das abordagens disponíveis na literatura sobre o aprendizado da cinemática inversa de robôs manipuladores.

2. Amplo estudo comparativo envolvendo modelos não-paramétricos de diferentes paradigmas de aprendizado na tarefa de aproximação de mapeamentos inversos em robótica, em particular da cinemática inversa.

3. Proposição de uma métrica de seleção que possibilitou a caracterização do desempenho global dos modelos de regressão em estimar os ângulos de junta das estruturas robóticas em tarefas de cinemática inversa.

4. Inclusão de testes de desempenho que envolvessem não só transformações pontuais isoladas, mas também de uma seqüência de pontos (i.e. trajetórias) a fim de ter uma visão mais global do resultado da aproximação.

5. Validação sistemática dos desempenhos por meio de análise de correlação de resíduos e de testes de hipóteses.

6. Aplicação inédita de dois modelos de aprendizado na aproximação da cinemática inversa de robôs manipuladores, a saber: MLM e LS-SVR.

7. Disponibilização em repositórios dos conjuntos de dados dos três robôs avaliados para uso público por parte da comunidade nacional e internacional.

6.3 Trabalhos Futuros

Um grande número de propostas de trabalhos na linha de pesquisa desenvolvida nesta dissertação pode ainda ser desenvolvida. Dentre os potenciais temas de pesquisa futura podem ser listadas as seguintes direções:

1. Realização de testes sistemáticos com diferentes níveis e tipos de ruído nas medidas de posição de entrada e consequente análise do efeito na estimação dos ângulos das juntas.

2. Testes em robôs reais disponíveis no CENTAURO (Centro de Referência em Automação e Robótica), da UFC, como por exemplo: Robô planar (Quanser) e Robô Humanóide NAO.

3. Estudo comparativo dos modelos avaliados nesta dissertação em problemas de aprendizado da dinâmica direta e inversa de robôs manipuladores para fins de controle de posição.

4. Estudo comparativo dos modelos avaliados nesta dissertação em no aprendizado de outros mapeamentos cinemáticos inversos de interesse em robótica, tal como coordenação visuomotora (mapeamento câmeras -> ângulos das juntas).

Referências

ALAVANDAR, S.; NIGAM, M. Inverse kinematics solution of 3dof planar robot using anfis. *Int. J. of Computers, Communication & Control*, v. 3, p. 150–155, 2008. Citado na página 74.

ANGULO, V. R. de; TORRAS, C. Self-calibration of a space robot. *Neural Networks, IEEE Transactions on*, IEEE, v. 8, n. 4, p. 951–963, 1997. Citado na página 81.

BAKER, D. R.; WAMPLER, C. W. On the inverse kinematics of redundant manipulators. *The International Journal of Robotics Research*, Sage Publications, v. 7, n. 2, p. 3–21, 1988. Citado na página 52.

BARRETO, G. D. A.; ARAÚJO, A. F.; RITTER, H. J. Self-organizing feature maps for modeling and control of robotic manipulators. *Journal of Intelligent and Robotic Systems*, Springer, v. 36, n. 4, p. 407–450, 2003. Citado 2 vezes nas páginas 32 e 81.

BRABANTER, K. D. et al. Ls-svmlab toolbox user's guide. *ESAT-SISTA Technical Report*, p. 10–146, 2011. Citado na página 91.

BURGES, C. J. A tutorial on support vector machines for pattern recognition. *Data mining and knowledge discovery*, Springer, v. 2, n. 2, p. 121–167, 1998. Citado na página 64.

BUSSAB, W. d. O.; MORETTIN, P. A. *Estatística básica*. [S.l.]: Saraiva, 2010. Citado na página 117.

CORKE, P. A robotics toolbox for matlab. *Robotics & Automation Magazine, IEEE*, IEEE, v. 3, n. 1, p. 24–32, 1996. Citado 2 vezes nas páginas 50 e 87.

CORKE, P. *Robotics, vision and control: fundamental algorithms in MATLAB*. [S.l.]: Springer Science & Business Media, 2011. Citado 2 vezes nas páginas 37 e 87.

CRAIG, J. J. *Introduction to robotics: mechanics and control*. [S.l.]: Pearson/Prentice Hall Upper Saddle River, NJ, USA:, 2005. Citado 9 vezes nas páginas 15, 32, 37, 40, 42, 43, 44, 47 e 52.

DANAFAR, S.; GRETTON, A.; SCHMIDHUBER, J. Characteristic kernels on structured domains excel in robotics and human action recognition. In: *Machine Learning and Knowledge Discovery in Databases*. [S.l.]: Springer, 2010. p. 264–279. Citado na página 69.

DeMers, D.; Kreutz-Delgado, K. Learning global direct inverse kinematics. In: MOODY, J.; HANSON, S.; LIPPMANN, R. (Ed.). *Advances in Neural Information Processing Systems 4*. [S.l.]: Morgan-Kaufmann, 1992. p. 589–594. Citado 2 vezes nas páginas 31 e 32.

DUKA, A.-V. Neural network based inverse kinematics solution for trajectory tracking of a robotic arm. *Procedia Technology*, Elsevier, v. 12, p. 20–27, 2014. Citado na página 60.

DUKA, A.-V. ANFIS based solution to the inverse kinematics of a 3DOF planar manipulator. *Procedia Technology*, v. 19, n. 0, p. 526 – 533, 2015. ISSN 2212-0173. 8th International Conference Interdisciplinarity in Engineering, INTER-ENG 2014, 9-10 October 2014, Tirgu Mures, Romania. Disponível em: <http://www.sciencedirect.com/science/article/pii/S2212017315000766>. Citado na página 74.

ENGELBRECHT, A. P. *Computational intelligence: an introduction*. [S.l.]: John Wiley & Sons, 2007. Citado na página 71.

FENG, S.; HU, S.; SHEN, J. Kinematic study on motoman hp6 arc welding robot based on unigraphics. *Transactions of Tianjin University*, Springer, v. 17, p. 199–202, 2011. Citado na página 52.

FENG, Y.; YAO-NAN, W.; YI-MIN, Y. Inverse kinematics solution for robot manipulator based on neural network under joint subspace. *International Journal of Computers Communications & Control*, v. 7, n. 3, p. 459–472, 2014. Citado na página 63.

FLETCHER, R. *Practical methods of optimization*. [S.l.]: John Wiley & Sons, 2013. Citado na página 69.

GROCHOW, K. et al. Style-based inverse kinematics. In: ACM. *ACM Transactions on Graphics (TOG)*. [S.l.], 2004. v. 23, n. 3, p. 522–531. Citado na página 69.

HASAN, A. T. et al. An adaptive-learning algorithm to solve the inverse kinematics problem of a 6 DOF serial robot manipulator. *Advances in Engineering Software*, Elsevier, v. 37, n. 7, p. 432–438, 2006. Citado na página 60.

HASSIBI, B.; STORK, D. G. et al. Second order derivatives for network pruning: Optimal brain surgeon. *Advances in neural information processing systems*, Citeseer, p. 164–164, 1993. Citado na página 66.

HAYKIN, S. S. *Neural networks and learning machines*. [S.l.: s.n.], 2009. Citado 5 vezes nas páginas 32, 55, 56, 57 e 65.

HORNE, B.; JAMSHIDI, M.; VADIEE, N. Neural networks in robotics: A survey. *Journal of Intelligent and Robotic Systems*, v. 3, p. 51–66, 1990. Citado na página 32.

HSU, C.-W.; LIN, C.-J. A comparison of methods for multiclass support vector machines. *Neural Networks, IEEE Transactions on*, IEEE, v. 13, n. 2, p. 415–425, 2002. Citado na página 64.

HUANG, G.-B.; ZHU, Q.-Y.; SIEW, C.-K. Extreme learning machine: theory and applications. *Neurocomputing*, Elsevier, v. 70, n. 1, p. 489–501, 2006. Citado 2 vezes nas páginas 33 e 60.

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*, v. 48, n. 5, p. 805–810, 2003. Citado na página 34.

IFR. *International Federation of Robotic, IFR*. [S.l.], 2013 (Acessado em 27/06/2015). Disponível em: <http://www.ifr.org/industrial-robots/statistics/>. Citado na página 29.

ISO. *ISO 8373 Robots and robotic devices - Vocabulary*. [S.l.], 2012. Citado na página 29.

JANG, J.-S. Anfis: adaptive-network-based fuzzy inference system. *Systems, Man and Cybernetics, IEEE Transactions on*, IEEE, v. 23, n. 3, p. 665–685, 1993. Citado 4 vezes nas páginas 33, 71, 72 e 73.

JANG, J.-S. R.; SUN, C.-T. *Neuro-fuzzy and soft computing: a computational approach to learning and machine intelligence*. [S.l.]: Prentice-Hall, Inc., 1996. Citado 3 vezes nas páginas 69, 71 e 73.

JIN, Y. Decentralized adaptive fuzzy control of robot manipulators. *IEEE Transactions on Systems, Man, and Cybernetics - Part B*, v. 28, n. 1, p. 47–57, 1998. Citado na página 34.

JR, F. J. M. The kolmogorov-smirnov test for goodness of fit. *Journal of the American statistical Association*, Taylor & Francis Group, v. 46, n. 253, p. 68–78, 1951. Citado na página 118.

JUNIOR, A. H. de S. et al. Minimal learning machine: a new distance-based method for supervised learning. In: *Advances in Computational Intelligence*. [S.l.]: Springer, 2013. p. 408–416. Citado 2 vezes nas páginas 33 e 74.

KOHONEN, T. Essentials of the self-organizing map. *Neural Networks*, v. 37, p. 52–65, 2013. Citado na página 33.

LAY, D. C.; CAMELIER, R.; IÓRIO, V. de M. *Álgebra linear e suas aplicações*. [S.l.]: LTC, 1997. Citado na página 63.

LECUN, Y. et al. Optimal brain damage. In: *NIPs*. [S.l.: s.n.], 1989. v. 2, p. 598–605. Citado na página 66.

LEE, C. Fuzzy logic in control systems: fuzzy logic controller. i. *Systems, Man and Cybernetics, IEEE Transactions on*, v. 20, n. 2, p. 404–418, Mar 1990. ISSN 0018-9472. Citado na página 71.

LEE, C. G.; ZIEGLER, M. Geometric approach in solving inverse kinematics of puma robots. *Aerospace and Electronic Systems, IEEE Transactions on*, IEEE, n. 6, p. 695–706, 1984. Citado na página 52.

LIU, M. Decentralized control of robot manipulators: Nonlinear and adaptive approaches. *IEEE Transactions on Automatic Control*, v. 44, n. 2, p. 357–363, 1999. Citado na página 34.

MACQUEEN, J. et al. Some methods for classification and analysis of multivariate observations. In: OAKLAND, CA, USA. *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*. [S.l.], 1967. v. 1, n. 14, p. 281–297. Citado na página 77.

MARTINETZ, T. M.; RITTER, H. J.; SCHULTEN, K. J. Three-dimensional neural net for learning visuomotor coordination of a robot arm. *Neural Networks, IEEE Transactions on*, IEEE, v. 1, n. 1, p. 131–136, 1990. Citado na página 81.

MASHHADANY, Y. I. A.; MIEEE, H. ANFIS-inverse-controlled PUMA 560 workspace robot with spherical wrist. *Procedia Engineering*, Elsevier, v. 41, p. 700–709, 2012. Citado na página 74.

MURPHY, K. P. *Machine learning: a probabilistic perspective*. [S.l.]: MIT press, 2012. Citado na página 67.

NAKAMURA, Y.; HANAFUSA, H. Inverse kinematic solutions with singularity robustness for robot manipulator control. *Journal of dynamic systems, measurement, and control*, American Society of Mechanical Engineers, v. 108, n. 3, p. 163–171, 1986. Citado na página 52.

NELDER, J. A.; MEAD, R. A simplex method for function minimization. *The computer journal*, Br Computer Soc, v. 7, n. 4, p. 308–313, 1965. Citado na página 91.

OGAWA, T.; KANADA, H. Solution for ill-posed inverse kinematics of robot arm by network inversion. *Journal of Robotics*, v. 2010, n. ID 870923, p. 1–9, 2010. Citado na página 31.

OLIVEIRA, A. S. *Retrofitting de Robôs Manipuladores com Incorporação de Controle de Posição e Força: Aplicação em um Robô Industrial*. Dissertação (Mestrado) — Universidade Federal de Santa Catarina, Programa de Pós-Graduaçã em Engenharia Mecânica, 2007. Citado 2 vezes nas páginas 15 e 30.

PEIPER, D. L. *The kinematics of manipulators under computer control*. [S.l.], 1968. Citado na página 52.

PRABHU, S. M.; GARG, D. P. Artificial neural network based robot control: An overview. *Journal of Intelligent and Robotic Systems*, v. 15, n. 4, p. 333–365, 1996. Citado na página 32.

PRINCIPE, J. C.; EULIANO, N. R.; LEFEBVRE, W. C. *Neural and Adaptive Systems: Fundamentals Through Simulations*. [S.l.]: John Wiley and Sons, 2000. Citado 2 vezes nas páginas 62 e 80.

RASMUSSEN, C. E. Gaussian processes for machine learning. Citeseer, 2006. Citado na página 33.

RITTER, H. J.; MARTINETZ, T. M.; SCHULTEN, K. J. Topology-conserving maps for learning visuo-motor-coordination. *Neural networks*, Elsevier, v. 2, n. 3, p. 159–168, 1989. Citado na página 81.

ROTH, B. Performance evaluation of manipulators from a kinematic viewpoint. *NBS Special Publication*, v. 459, p. 39–62, 1976. Citado na página 52.

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. *Learning internal representations by error propagation*. [S.l.], 1985. Citado na página 33.

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Learning representations by back-propagating errors. *Cognitive modeling*, v. 5, 1988. Citado na página 58.

SCHOLKOPF, B.; SMOLA, A. J. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. [S.l.]: MIT press, 2001. Citado na página 65.

SMOLA, A. J.; SCHÖLKOPF, B. A tutorial on support vector regression. *Statistics and computing*, Springer, v. 14, n. 3, p. 199–222, 2004. Citado na página 64.

SOUZA, S. Xavier-de et al. Coupled simulated annealing. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, IEEE, v. 40, n. 2, p. 320–335, 2010. Citado na página 91.

SPONG, M. W.; HUTCHINSON, S.; VIDYASAGAR, M. *Robot modeling and control*. [S.l.]: Wiley New York, 2006. Citado 3 vezes nas páginas 32, 37 e 42.

SUYKENS, J. A. et al. *Least squares support vector machines*. [S.l.]: World Scientific, 2002. Citado 3 vezes nas páginas 33, 65 e 66.

TAKAGI, T.; SUGENO, M. Fuzzy identification of systems and its applications to modeling and control. *Systems, Man and Cybernetics, IEEE Transactions on*, IEEE, n. 1, p. 116–132, 1985. Citado na página 72.

TIKHONOV, A. N.; ARSENIN, V. Y. *Solutions of Ill-Posed Problems*. [S.l.]: Winston, New York, 1977. Citado na página 31.

TSAI, L.-W.; MORGAN, A. P. Solving the kinematics of the most general six-and five-degree-of-freedom manipulators by continuation methods. *Journal of Mechanical Design*, American Society of Mechanical Engineers, v. 107, n. 2, p. 189–200, 1985. Citado na página 52.

VAPNIK, V. *The nature of statistical learning theory*. [S.l.]: Springer Science & Business Media, 2000. Citado na página 64.

VASUKI, A.; VANATHI, P. A review of vector quantization techniques. *Potentials, IEEE*, IEEE, v. 25, n. 4, p. 39–47, 2006. Citado na página 77.

WALTER, J.; RITER, H.; SCHULTEN, K. Nonlinear prediction with self-organizing maps. In: IEEE. *Neural Networks, 1990., 1990 IJCNN International Joint Conference on*. [S.l.], 1990. p. 589–594. Citado na página 76.

WRIGHT, S. J.; NOCEDAL, J. *Numerical optimization*. [S.l.]: Springer New York, 1999. Citado na página 69.

YILDIRIM, Ş.; ESKI, İ. A QP artificial neural network inverse kinematic solution for accurate robot path control. *Journal of mechanical science and technology*, Springer, v. 20, n. 7, p. 917–928, 2006. Citado na página 60.